

---

# Traffic Control Documentation

*Release 2.1-dev*

**Apache Software Foundation**

**Nov 15, 2022**



---

## Contents

---

<b>1</b>	<b>CDN Basics</b>	<b>3</b>
1.1	CDN Basics . . . . .	3
<b>2</b>	<b>Traffic Control Overview</b>	<b>11</b>
2.1	Traffic Control Overview . . . . .	11
<b>3</b>	<b>Administrator's Guide</b>	<b>21</b>
3.1	Administrator's Guide . . . . .	21
<b>4</b>	<b>Developer's Guide</b>	<b>131</b>
4.1	Developer's Guide . . . . .	131
<b>5</b>	<b>FAQ</b>	<b>505</b>
5.1	FAQ . . . . .	505
<b>6</b>	<b>Indices and Tables</b>	<b>509</b>
6.1	Glossary . . . . .	509
	<b>Index</b>	<b>511</b>



The vast majority of today's Internet traffic is media files being sent from a single source to many thousands or even millions of destinations. Content Delivery Networks make that one-to-many distribution possible in an economical way.

Traffic Control is an Open Source implementation of a Content Delivery Network.

The following documentation sections are available:



A review of the basic functionality of a Content Delivery Network.

## 1.1 CDN Basics

Traffic Control is a CDN control plane, see the topics below to familiarize yourself with the basic concepts of a CDN.

### 1.1.1 Content Delivery Networks

The vast majority of today's Internet traffic is media files (often video or audio) being sent from a single source (the *Content Provider*) to many thousands or even millions of destinations (the *Content Consumers*). Content Delivery Networks are the technology that make that one-to-many distribution possible in an economical way. A Content Delivery Network (CDN) is a distributed system of servers for delivering content over HTTP. These servers are deployed in multiple locations with the goal of optimizing the delivery of content to the end users, while minimizing the traffic on the network. A CDN typically consists of the following:

- **Caching Proxies** The proxy (cache or caching proxy) is a server that both proxies the requests and caches the results for reusing.
- **Content Router** The Content Router ensures that the end user is connected to the optimal cache for the location of the end user and content availability.
- **Health Protocol** The Health Protocol monitors the usage of the caches and tenants in the CDN.
- **Configuration Management System** In many cases a CDN encompasses hundreds of servers across a large geographic area. The Configuration Management System allows an operator to manage these servers.
- **Log File Analysis System** Every transaction in the CDN gets logged. The Log File Analysis System aggregates all of the log entries from all of the servers to a central location for analysis and troubleshooting.

### 1.1.2 HTTP 1.1

For a comprehensive look at Traffic Control, it is important to understand basic HTTP 1.1 protocol operations and how caches function. The example below illustrates the fulfillment of an HTTP 1.1 request in a situation without CDN or proxy, followed by viewing the changes after inserting different types of (caching) proxies. Several of the examples below are simplified for clarification of the essentials.

For complete details on HTTP 1.1 see [RFC 2616 - Hypertext Transfer Protocol – HTTP/1.1](#).

Below are the steps of a client retrieving the URL `http://www.origin.com/foo/bar/fun.html` using HTTP/1.1 without proxies:

1. The client sends a request to the Local DNS (LDNS) server to resolve the name `www.origin.com` to an IPv4 address.
2. If the LDNS does not have this name (IPv4 mapping cached), it sends DNS requests to the `.`, `.com`, and `.origin.com` authoritative servers until it receives a response with the address for `www.origin.com`. Per the DNS SPEC, this response has a Time To Live (TTL), which indicates how long this mapping can be cached at the LDNS server. In the example, the IP address found by the LDNS server for `www.origin.com` is 44.33.22.11.

---

**Note:** While longer DNS TTLs of a day (86400 seconds) or more are quite common in other use cases, in CDN use cases DNS TTLs are often below a minute.

---

3. The client opens a TCP connection from a random port locally to port 80 (the HTTP default) on 44.33.22.11, and sends this (showing the minimum HTTP 1.1 request, typically there are additional headers):

```
GET /foo/bar/fun.html HTTP/1.1
Host: www.origin.com
```

4. The server at `www.origin.com` looks up the `Host:` header to match that to a configuration section, usually referred to as a virtual host section. If the `Host:` header and configuration section match, the search continues for the content of the path `/foo/bar/fun.html`, in the example, this is a file that contains `<html><body>This is a fun file</body></html>`, so the server responds with the following:

```
HTTP/1.1 200 OK
Content-Type: text/html; charset=UTF-8
Content-Length: 45

<html><body>This is a fun file</body></html>
```

At this point, HTTP transaction is complete.

### 1.1.3 Caching Proxies

The main function of a CDN is to proxy requests from clients to origin servers and cache the results. To proxy, in the CDN context, is to obtain content using HTTP from an origin server on behalf of a client. To cache is to store the results so they can be reused when other clients are requesting the same content. There are three types of proxies in use on the Internet today which are described below.



#### Reverse Proxy

A reverse proxy acts on behalf of the origin server. The client is mostly unaware it is communicating with a proxy and not the actual origin. All EDGE caches in a Traffic Control CDN are reverse proxies. To the end user a Traffic Control based CDN appears as a reverse proxy since it retrieves content from the



origin server, acting on behalf of that origin server. The client requests a URL that has a hostname which resolves to the reverse proxy's IP address and, in compliance with the HTTP 1.1 specification, the client sends a `Host :` header to the reverse proxy that matches the hostname in the URL. The proxy looks up this hostname in a list of mappings to find the origin hostname; if the hostname of the `Host` header is not found in the list, the proxy will send an error (404 Not Found) to the client. If the supplied hostname is found in this list of mappings, the proxy checks the cache, and when the content is not already present, connects to the origin the requested `Host :` maps to and requests the path of the original URL, providing the origin hostname in the `Host` header. The proxy then stores the URL in cache and serves the contents to the client. When there are subsequent requests for the same URL, a caching proxy serves the content out of cache thereby reducing latency and network traffic.

**See also:**

[ATS documentation on reverse proxy.](#)

To insert a reverse proxy into the previous HTTP 1.1 example, the reverse proxy requires provisioning for `www.origin.com`. By adding a remap rule to the cache, the reverse proxy then maps requests to this origin. The content owner must inform the clients, by updating the URL, to receive the content from the cache and not from the origin server directly. For this example, the remap rule on the cache is: `http://www-origin-cache.cdn.com` `http://www.origin.com`.

---

**Note:** In the previous example minimal headers were shown on both the request and response. In the examples that follow, the origin server response is more realistic.

---

```
HTTP/1.1 200 OK
Date: Sun, 14 Dec 2014 23:22:44 GMT
Server: Apache/2.2.15 (Red Hat)
Last-Modified: Sun, 14 Dec 2014 23:18:51 GMT
ETag: "1aa008f-2d-50a3559482cc0"
Content-Length: 45
Connection: close
Content-Type: text/html; charset=UTF-8

<html><body>This is a fun file</body></html>
```

The client is given the URL `http://www-origin-cache.cdn.com/foo/bar/fun.html` (note the different hostname) and when attempting to obtain that URL, the following occurs:

1. The client sends a request to the LDNS server to resolve the name `www-origin-cache.cdn.com` to an IPv4 address.
2. Similar to the previous case, the LDNS server resolves the name `www-origin-cache.cdn.com` to an IPv4 address, in this example, this address is 55.44.33.22.
3. The client opens a TCP connection from a random port locally, to port 80 (the HTTP default) on 55.44.33.22, and sends the following:

```
GET /foo/bar/fun.html HTTP/1.1
Host: www-origin-cache.cdn.com
```

4. The reverse proxy looks up `www-origin-cache.cdn.com` in its remap rules, and finds the origin is `www.origin.com`.
5. The proxy checks its cache to see if the response for `http://www-origin-cache.cdn.com/foo/bar/fun.html` is already in the cache.
- 6a. If the response is not in the cache:

1. The proxy uses DNS to get the IPv4 address for `www.origin.com`, connect to it on port 80, and sends:

```
GET /foo/bar/fun.html HTTP/1.1
Host: www.origin.com
```

2. The origin server responds with the headers and content as shown:

```
HTTP/1.1 200 OK
Date: Sun, 14 Dec 2014 23:22:44 GMT
Server: Apache/2.2.15 (Red Hat)
Last-Modified: Sun, 14 Dec 2014 23:18:51 GMT
ETag: "1aa008f-2d-50a3559482cc0"
Content-Length: 45
Connection: close
Content-Type: text/html; charset=UTF-8

<html><body>This is a fun file</body></html>
```

3. The proxy sends the origin response on to the client adding a `Via:` header (and maybe others):

```
HTTP/1.1 200 OK
Date: Sun, 14 Dec 2014 23:22:44 GMT
Last-Modified: Sun, 14 Dec 2014 23:18:51 GMT
ETag: "1aa008f-2d-50a3559482cc0"
Content-Length: 45
Connection: close
Content-Type: text/html; charset=UTF-8
Age: 0
Via: http/1.1 cache01.cdn.kabletown.net (ApacheTrafficServer/4.2.1_
→[uScSsSfUpSeN:t cCSi p sS])
Server: ATS/4.2.1

<html><body>This is a fun file</body></html>
```

- 6b. If it *is* in the cache:

The proxy responds to the client with the previously retrieved result:

```
HTTP/1.1 200 OK
Date: Sun, 14 Dec 2014 23:22:44 GMT
Last-Modified: Sun, 14 Dec 2014 23:18:51 GMT
ETag: "1aa008f-2d-50a3559482cc0"
Content-Length: 45
Connection: close
Content-Type: text/html; charset=UTF-8
Age: 39711
Via: http/1.1 cache01.cdn.kabletown.net (ApacheTrafficServer/4.2.1_
→[uScSsSfUpSeN:t cCSi p sS])
Server: ATS/4.2.1

<html><body>This is a fun file</body></html>
```



## Forward Proxy

A forward proxy acts on behalf of the client. The origin server is mostly unaware of the proxy, the client requests the proxy to retrieve content from a particular origin server. All MID caches in a Traffic Control based CDN are forward proxies. In a forward proxy scenario, the client is explicitly configured to use

the the proxy's IP address and port as a forward proxy. The client always connects to the forward proxy for content. The content provider does not have to change the URL the client obtains, and is unaware of the proxy in the middle.

**See also:**

[ATS documentation on forward proxy.](#)

Below is an example of the client retrieving the URL `http://www.origin.com/foo/bar/fun.html` through a forward proxy:

1. The client requires configuration to use the proxy, as opposed to the reverse proxy example. Assume the client configuration is through preferences entries or other to use the proxy IP address 99.88.77.66 and proxy port 8080.
2. To retrieve `http://www.origin.com/foo/bar/fun.html` URL, the client connects to 99.88.77.66 on port 8080 and sends:

```
GET http://www.origin.com/foo/bar/fun.html HTTP/1.1
```

**Note:** In this case, the client places the entire URL after GET, including protocol and hostname (`http://www.origin.com`), but in the reverse proxy and direct-to-origin case it puts only the path portion of the URL (`/foo/bar/fun.html`) after the GET.

3. The proxy verifies whether the response for `http://www-origin-cache.cdn.com/foo/bar/fun.html` is already in the cache.
- 4a. If it is not in the cache:

1. The proxy uses DNS to obtain the IPv4 address for `www.origin.com`, connects to it on port 80, and sends:

```
GET /foo/bar/fun.html HTTP/1.1
Host: www.origin.com
```

2. The origin server responds with the headers and content as shown below:

```
HTTP/1.1 200 OK
Date: Sun, 14 Dec 2014 23:22:44 GMT
Server: Apache/2.2.15 (Red Hat)
Last-Modified: Sun, 14 Dec 2014 23:18:51 GMT
ETag: "1aa008f-2d-50a3559482cc0"
Content-Length: 45
Connection: close
Content-Type: text/html; charset=UTF-8

<html><body>This is a fun file</body></html>
```

3. The proxy sends this on to the client adding a `Via:` header (and maybe others):

```
HTTP/1.1 200 OK
Date: Sun, 14 Dec 2014 23:22:44 GMT
Last-Modified: Sun, 14 Dec 2014 23:18:51 GMT
ETag: "1aa008f-2d-50a3559482cc0"
Content-Length: 45
Connection: close
Content-Type: text/html; charset=UTF-8
Age: 0
Via: http/1.1 cache01.cdn.kabletown.net (ApacheTrafficServer/4.2.1
→[uScSsSfUpSeN:t cCSi p sS])
```

(continues on next page)

(continued from previous page)

```
Server: ATS/4.2.1

<html><body>This is a fun file</body></html>
```

4b. If it *is* in the cache:

The proxy responds to the client with the previously retrieved result:

```
HTTP/1.1 200 OK
Date: Sun, 14 Dec 2014 23:22:44 GMT
Last-Modified: Sun, 14 Dec 2014 23:18:51 GMT
ETag: "1aa008f-2d-50a3559482cc0"
Content-Length: 45
Connection: close
Content-Type: text/html; charset=UTF-8
Age: 99711
Via: http/1.1 cache01.cdn.kabletown.net (ApacheTrafficServer/4.2.1
→[uScSsSfUpSeN:t cCSi p sS])
Server: ATS/4.2.1

<html><body>This is a fun file</body></html>
```



## Transparent Proxy

Neither the origin nor the client are aware of the actions performed by the transparent proxies. A Traffic Control based CDN does not use transparent proxies. If you are interested you can learn more about transparent proxies on [wikipedia](http://wikipedia.org).

### 1.1.4 Cache Control Headers and Revalidation

The [HTTP/1.1 spec](http://http11.org) allows for origin servers and clients to influence how caches treat their requests and responses. By default, the Traffic Control CDN will honor cache control headers. Most commonly, origin servers will tell the downstream caches how long a response can be cached:

```
HTTP/1.1 200 OK
Date: Sun, 14 Dec 2014 23:22:44 GMT
Server: Apache/2.2.15 (Red Hat)
Last-Modified: Sun, 14 Dec 2014 23:18:51 GMT
ETag: "1aa008f-2d-50a3559482cc0"
Cache-Control: max-age=86400
Content-Length: 45
Connection: close
Content-Type: text/html; charset=UTF-8

<html><body>This is a fun file</body></html>
```

In the above response, the origin server tells downstream caching systems that the maximum time to cache this response for is 86400 seconds. The origin can also add a `Expires:` header, explicitly telling the cache the time this response is to be expired. When a response is expired it usually doesn't get deleted from the cache, but, when a request comes in that would have hit on this response if it was not expired, the cache *revalidates* the response. In stead of requesting the object again from the origin server, the cache will send a request to the origin indicating what version of the response it has, and asking if it has changed. If it changed, the server will send a 200 OK response, with the new data. If it has not changed, the origin server will send back a 304 Not Modified response indicating the response

is still valid, and that the cache can reset the timer on the response expiration. To indicate what version the client (cache) has it will add an `If-Not-Modified-Since:` header, or an `If-None-Match:` header. For example, in the `If-None-Match:` case, the origin will send an `ETag` header that uniquely identifies the response. The client can use that in a revalidation request like:

```
GET /foo/bar/fun.html HTTP/1.1
If-None-Match: "1aa008f-2d-50a3559482cc0"
Host: www.origin.com
```

If the content has changed (meaning, the new response would not have had the same `ETag`) it will respond with 200 OK, like:

```
HTTP/1.1 200 OK
Date: Sun, 18 Dec 2014 3:22:44 GMT
Server: Apache/2.2.15 (Red Hat)
Last-Modified: Sun, 14 Dec 2014 23:18:51 GMT
ETag: "1aa008f-2d-50aa00feadd"
Cache-Control: max-age=604800
Content-Length: 49
Connection: close
Content-Type: text/html; charset=UTF-8

<html><body>This is NOT a fun file</body></html>
```

If the Content did not change (meaning, the response would have had the same `ETag`) it will respond with 304 Not Modified, like:

```
304 Not Modified
Date: Sun, 18 Dec 2014 3:22:44 GMT
Server: Apache/2.2.15 (Red Hat)
Last-Modified: Sun, 14 Dec 2014 23:18:51 GMT
ETag: "1aa008f-2d-50a3559482cc0"
Cache-Control: max-age=604800
Content-Length: 45
Connection: close
Content-Type: text/html; charset=UTF-8
```

Note that the 304 response only has headers, not the data.



---

### Traffic Control Overview

---

An introduction to the Traffic Control architecture, components, and their integration.

## 2.1 Traffic Control Overview

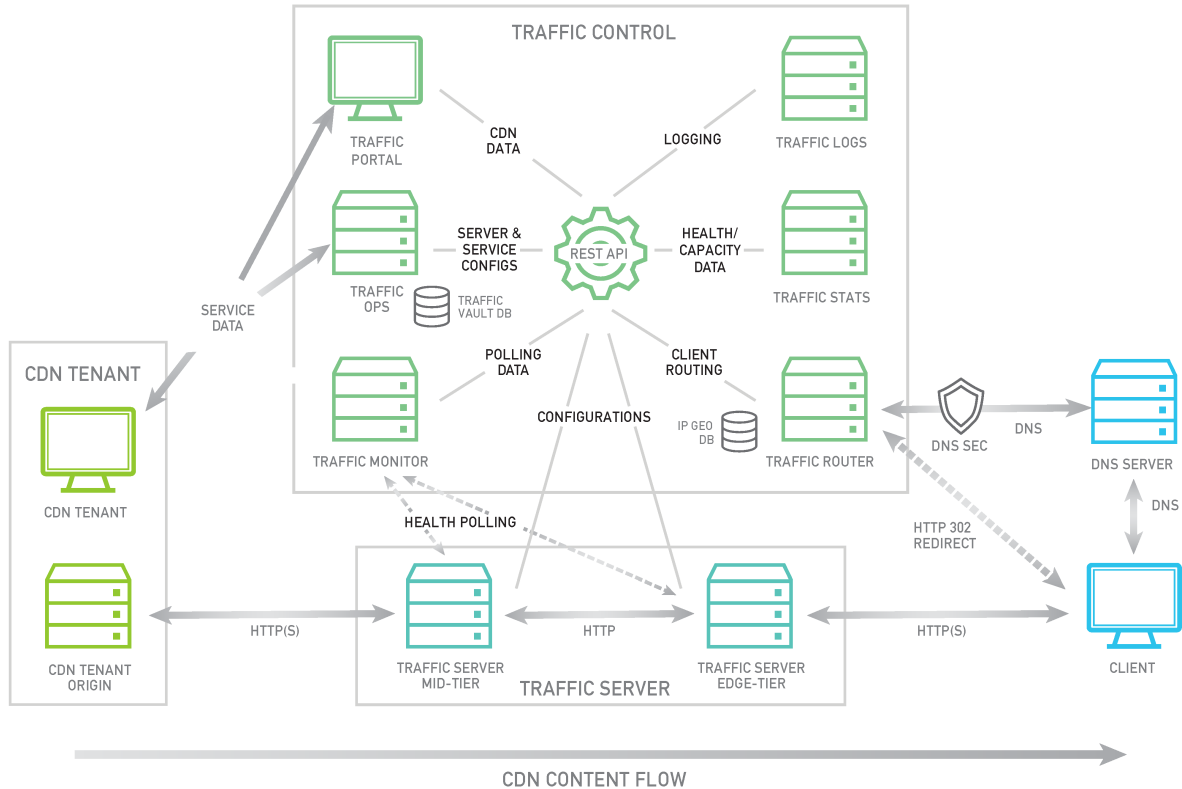
Introduces the Traffic Control architecture, components, and their integration.

### 2.1.1 Introduction

Traffic Control is a caching server control plane application which is used to aggregate caching servers into a Content Delivery Network (CDN). The CDN caching software chosen for Traffic Control is [Apache Traffic Server](#) (ATS). Although the current release only supports ATS as a cache, this may change with future releases.

Traffic Control was first developed at Comcast for internal use and released to Open Source in April of 2015. Traffic Control moved into the Apache Incubator in August of 2016.

Traffic Control implements the elements illustrated in green in the diagram below.



## Traffic Ops

- **Traffic Ops** is used to configure caching servers and CDN delivery services. It also contains APIs used to access CDN data.

## Traffic Router

- **Traffic Router** is used to route clients requests to the closest healthy cache by analyzing the health, capacity, and state of the caching servers and relative distance from each cache group to the location of the client.

## Traffic Monitor

- **Traffic Monitor** does health polling of the caching servers on a very short interval to keep track of which servers should be kept in rotation.

## Traffic Stats

- **Traffic Stats** collects real time traffic statistics aggregated from each of the caching servers. This data is used by the Traffic Router to assess the available capacity of each caching server which it uses to balance traffic load and prevent overload.

## Traffic Portal

- **Traffic Portal** is a web application which leverages the Traffic Ops APIs to present CDN data through a web interface.

## Traffic Logs

- Traffic Logs is currently under construction and is intended to aggregate Traffic Server request/response logs as well as other server logs. Logs will be parsed and indexed for search.

In the next sections each of these components will be explained further.



## 2.1.2 Traffic Ops

Traffic Ops is the tool for administration (configuration and monitoring) of all components in a Traffic Control CDN. The CDN administrator uses Traffic Ops to manage servers, cache groups, delivery services, etc. In many cases, a configuration change requires propagation to several, or even all, caches and only explicitly after or before the same change propagates to Traffic Router. Traffic Ops takes care of this required consistency between the different components and their configuration. Traffic Ops exposes its data through a series of HTTP APIs and has a user interface that is interactive and viewable using a standard web browser.

Traffic Ops uses a MySQL or PostgreSQL database to store the configuration information, and the [Mojolicious framework](#) to generate the user interface and APIs. Not all configuration data is in this database however; for sensitive data, like SSL private keys or token based authentication shared secrets, a separate key-value store is used, allowing the operator to harden the server that runs this key-value store better from a security perspective (i.e only allow Traffic Ops access it with a cert). The Traffic Ops server, by design, needs to be accessible from all the other servers in the Traffic Control CDN.

Traffic Ops generates all the application specific configuration files for the caches and other servers. The caches and other servers check in with Traffic Ops at a regular interval (default 15 minutes) to see if updated configuration files require application.

Traffic Ops also runs a collection of periodic checks to determine the operational readiness of the caches. These periodic checks are customizable by the Traffic Ops admin using extensions.

Traffic Ops is in the process of migrating from Perl to Go, and currently runs as two applications. The Go application serves all endpoints which have been rewritten in the Go language, and transparently proxies all other requests to the old Perl application. Both applications are installed by the RPM, and both run as a single service. When the project has fully migrated to Go, the Perl application will be removed, and the RPM and service will consist solely of the Go application.



### Traffic Ops Extension

Traffic Ops Extensions are a way to enhance the basic functionality of Traffic Ops in a custom manner. There are three types of extensions:

- Check Extensions - Allows you to add custom checks to the “Health->Server Checks” view.
- Configuration Extension - Allows you to add custom configuration file generators.
- Data source Extensions - Allows you to add data sources for the graph views and usage APIs.

## 2.1.3 Traffic Portal

Traffic Portal is an [AngularJS 1.x](#) client served from a [Node.js](#) web server designed to consume the Traffic Ops API. It is the official replacement for the legacy Traffic Ops UI.

Features include:

- CDN Monitoring
- CDN Administration
- Delivery Service Configuration
- Cache Maintenance

See *Traffic Portal - Using*

## 2.1.4 Traffic Router

Traffic Router's function is to send clients to the most optimal cache. Optimal in this case is based on a number of factors:

- Distance between the cache and the client (not necessarily measured in meters, but quite often in layer 3 network hops). Less network distance between the client and cache yields better performance, and lower network load. Traffic Router helps clients connect to the best performing cache for their location at the lowest network cost.
- Availability of caches and health / load on the caches. A common issue in Internet and television distribution scenarios is having many clients attempting to retrieve the same content at the same time. Traffic Router helps clients route around overloaded or down caches.
- Availability of content on a particular cache. Reusing of content through cache HITs is the most important performance gain a CDN can offer. Traffic Router sends clients to the cache that is most likely to already have the desired content.

Traffic routing options are often configured at the Delivery Service level.



### Delivery Service

As discussed in the basic concepts section, the EDGE caches are configured as reverse proxies, and the Traffic Control CDN looks from the outside as a very large reverse proxy. Delivery Services are often referred to a reverse proxy remap rule. In most cases, a Delivery Service is a one to one mapping to a FQDN that is used as a hostname to deliver the content. Many options and settings regarding how to optimize the content delivery, which is configurable on a Delivery Service basis. Some examples of these Delivery Service settings are:

- Cache in RAM, cache on disk, or do not cache at all.
- Use DNS or HTTP Content routing (see below).
- Limits on transactions per second and bandwidth.
- Protocol (http or https).
- Token based authentication settings.
- Header rewrite rules.

Since Traffic Control version 2.1 `deliveryservices` can optionally be linked to a *Profile*, and have parameters associated with them. The first feature that uses `deliveryservice` parameters is the *Multi Site Origin* configuration. Delivery Services are also for use in allowing multi-tenants to coexist in the Traffic Control CDN without interfering with each other, and to keep information about their content separated.

## ➔ Localization

Traffic Router uses a JSON input file called the *coverage zone map* to determine what *cachegroup* is closest to the client. If the client IP address is not in this coverage zone map, it falls back to *geo*, using the maxmind database to find the client's location, and the geo coordinates from Traffic Ops for the cachegroup.

Traffic Router is inserted into the HTTP retrieval process by making it DNS authoritative for the domain of the CDN delivery service. In the example of the reverse proxy, the client was given the `http://www-origin-cache.cdn.com/foo/bar/fun.html` url. In a Traffic Control CDN, URLs start with a routing name, which is configurable per-Delivery Service, e.g. `http://foo.mydeliveryservice.cdn.com/fun/example.html` with the chosen routing name `foo`.

## ➔ DNS Content Routing

For a DNS delivery service the client might receive a URL such as `http://foo.dsname.cdn.com/fun/example.html`. When the LDNS server is resolving this `foo.dsname.cdn.com` hostname to an IP address, it ends at Traffic Router because it is the authoritative DNS server for `cdn.com` and the domains below it, and subsequently responds with a list of IP addresses from the eligible caches based on the location of the LDNS server. When responding, Traffic Router does not know the actual client IP address or the path that the client is going to request. The decision on what cache IP address (or list of cache IP addresses) to return is solely based on the location of the LDNS server and the health of the caches. The client then connects to port 80 on the cache, and sends the `Host: foo.dsname.cdn.com` header. The configuration of the cache includes the remap rule `http://foo.dsname.cdn.com http://origin.dsname.com` to map the routed name to an origin hostname.

## ➔ HTTP Content Routing

For an HTTP delivery service the client might receive a URL such as `http://bar.dsname.cdn.com/fun/example.html`. The LDNS server resolves this `bar.dsname.cdn.com` to an IP address, but in this case Traffic Router returns its own IP address. The client opens a connection to port 80 on the Traffic Router's IP address, and sends:

```
GET /fun/example.html HTTP/1.1
Host: bar.dsname.cdn.com
```

Traffic Router uses an HTTP 302 to redirect the client to the best cache. For example:

```
HTTP/1.1 302 Moved Temporarily
Server: Apache-Coyote/1.1
Location: http://atsec-nyc-02.dsname.cdn.com/fun/example.html
Content-Length: 0
Date: Tue, 13 Jan 2015 20:01:41 GMT
```

The information Traffic Router can consider when selecting a cache in this case is much better:

- The client's IP address (the other side of the socket).
- The URL path the client is requesting, excluding query string.
- All HTTP 1.1 headers.

The client follows the redirect and performs a DNS request for the IP address for `atsec-nyc-02.dsname.cdn.com`, and normal HTTP steps follow, except the sending of the `Host:` header when connected to the cache is `Host: atsec-nyc-02.dsname.cdn`, and the configuration of the cache includes the remap rule (e.g., “`http://atsec-nyc-02.dsname.cdn http://origin.dsname.com`”).

Traffic Router sends all requests for the same path in a delivery service to the same cache in a cache group using consistent hashing, in this case all caches in a cache group are not carrying the same content, and there is a much larger combined cache in the cache group.

In many cases DNS content routing is the best possible option, especially in cases where the client is receiving small objects from the CDN like images and web pages.

Traffic Router is redundant and horizontally scalable by adding more instances into the DNS hierarchy using NS records.

## 2.1.5 Traffic Monitor

Traffic Monitor is an HTTP service that monitors the caches in a CDN for a variety of metrics. These metrics are for use in determining the overall health of a given cache and the related delivery services. A given CDN can operate a number of Traffic Monitors, from a number of geographically diverse locations, to prevent false positives caused by network problems at a given site.

Traffic Monitors operate independently, but use the state of other Traffic Monitors in conjunction with their own state, to provide a consistent view of CDN cache health to upstream applications such as Traffic Router. Health Protocol governs the cache and Delivery Service availability.

Traffic Monitor provides a view into CDN health using several RESTful JSON endpoints, which are consumed by other Traffic Monitors and upstream components such as Traffic Router. Traffic Monitor is also responsible for serving the overall CDN configuration to Traffic Router, which ensures that the configuration of these two critical components remain synchronized as operational and health related changes propagate through the CDN.

### Cache Monitoring

Traffic Monitor polls all caches configured with a status of `REPORTED` or `ADMIN_DOWN` at an interval specified as a configuration parameter in Traffic Ops. If the cache is set to `ADMIN_DOWN` it is marked as unavailable but still polled for availability and statistics. If the cache is explicitly configured with a status of `ONLINE` or `OFFLINE`, it is not polled by Traffic Monitor and presented to Traffic Router as configured, regardless of actual availability.

Traffic Monitor makes HTTP requests at regular intervals to a special URL on each EDGE cache and consumes the JSON output. The special URL is a plugin running on the Apache Traffic Server (ATS) caches called `astats`, which is restricted to Traffic Monitor only. The `astats` plugin provides insight into application and system performance, such as:

- Throughput (e.g. bytes in, bytes out, etc).
- Transactions (e.g. number of 2xx, 3xx, 4xx responses, etc).
- Connections (e.g. from clients, to parents, origins, etc).
- Cache performance (e.g.: hits, misses, refreshes, etc).
- Storage performance (e.g.: writes, reads, frags, directories, etc).
- System performance (e.g: load average, network interface throughput, etc).

Many of the application level statistics are available at the global or aggregate level, some at the Delivery Service (remap rule) level. Traffic Monitor uses the system level performance to determine the overall health of the cache by evaluating network throughput and load against values configured in Traffic Ops. Traffic Monitor also uses throughput and transaction statistics at the remap rule level to determine Delivery Service health.

If `astats` is unavailable due to a network related issue or the system statistics have exceeded the configured thresholds, Traffic Monitor will mark the cache as unavailable. If the delivery service statistics exceed the configured thresholds, the delivery service is marked as unavailable, and Traffic Router will start sending clients to the overflow destinations for that delivery service, but the cache remains available to serve other content,

#### See also:

For more information on ATS Statistics, see the [ATS documentation](#)



## Health Protocol

Redundant Traffic Monitor servers operate independently from each other but take the state of other Traffic Monitors into account when asked for health state information. In the above overview of cache monitoring, the behavior of Traffic Monitor pertains only to how an individual instance detects and handles failures. The Health Protocol adds another dimension to the health state of the CDN by merging the states of all Traffic Monitors into one, and then taking the *optimistic* approach when dealing with a cache or Delivery Service that might have been marked as unavailable by this particular instance or a peer instance of Traffic Monitor.

Upon startup or configuration change in Traffic Ops, in addition to caches, Traffic Monitor begins polling its peer Traffic Monitors whose state is set to `ONLINE`. Each `ONLINE` Traffic Monitor polls all of its peers at a configurable interval and saves the peer's state for later use. When polling its peers, Traffic Monitor asks for the raw health state from each respective peer, which is strictly that instance's view of the CDN's health. When any `ONLINE` Traffic Monitor is asked for CDN health by an upstream component, such as Traffic Router, the component gets the health protocol influenced version of CDN health (non-raw view).

In operation of the health protocol, Traffic Monitor takes all health states from all peers, including the locally known health state, and serves an optimistic outlook to the requesting client. This means that, for example, if three of the four Traffic Monitors see a given cache or Delivery Service as exceeding its thresholds and unavailable, it is still considered available. Only if all Traffic Monitors agree that the given object is unavailable is that state propagated to upstream components. This optimistic approach to the Health Protocol is counter to the “fail fast” philosophy, but serves well for large networks with complicated geography and or routing. The optimistic Health Protocol allows network failures or latency to occur without affecting overall traffic routing, as Traffic Monitors can and do have a different view of the network when deployed in geographically diverse locations. Short polling intervals of both the caches and Traffic Monitor peers help to reduce customer impact of outages.

It is not uncommon for a cache to be marked unavailable by Traffic Monitor - in fact, it is business as usual for many CDNs. A hot video asset may cause a single cache (say cache-03) to get close to its interface capacity, the health protocol “kicks in”, and Traffic Monitor marks cache-03 as unavailable. New clients want to see the same asset, and now, Traffic Router will send these customers to another cache (say cache-01) in the same cachegroup. The load is

now shared between cache-01 and cache-03. As clients finish watching the asset on cache-03, it will drop below the threshold and gets marked available again, and new clients will now go back to cache-03 again.

It is less common for a delivery service to be marked unavailable by Traffic Monitor, the delivery service thresholds are usually used for overflow situations at extreme peaks to protect other delivery services in the CDN from getting impacted.

### 2.1.6 Traffic Stats

Traffic Stats is a program written in [Golang](#) that is used to acquire and store statistics about CDNs controlled by Traffic Control. Traffic Stats mines metrics from Traffic Monitor's JSON APIs and stores the data in [InfluxDb](#). Data is typically stored in InfluxDb on a short-term basis (30 days or less). The data from InfluxDb is then used to drive graphs created by [Grafana](#) - which are linked to from Traffic Ops - as well as provide data exposed through the Traffic Ops API. Traffic Stats performs two functions: first it gathers stat data for Edge Caches and Delivery Services at a configurable interval (10 second default) from the Traffic Monitor API's and stores the data in InfluxDb; second it summarizes all of the stats once a day (around midnight UTC) and creates a daily rollup containing the Max Gbps served and the Total Bytes served.

Stat data is stored in three different databases:

- `cache_stats`: The `cache_stats` database is used to store data gathered from edge caches. The [measurements](#) stored by cache are: `bandwidth`, `maxKbps`, and `client_connections` (`ats.proxy.process.http.current_client_connections`). Cache Data is stored with [tags](#) for `hostname`, `cachegroup`, and `CDN`. Data can be queried using tags.
- `deliveryservice_stats`: The `deliveryservice_stats` database is used to store data for delivery services. The measurements stored by delivery service are: `kbits`, `status_4xx`, `status_5xx`, `tps_2xx`, `tps_3xx`, `tps_4xx`, `tps_5xx`, and `tps_total`. Delivery Service stats are stored with tags for `cachegroup`, `CDN`, and `Deliveryservice xml_id`.
- `daily_stats`: The `daily_stats` database is used to store summary data for daily activities. The stats that are currently summarized are Max Bandwidth and Bytes Served and they are stored by `CDN`.

---

Traffic Stats does not influence overall CDN operation, but is required in order to display charts in Traffic Ops and Traffic Portal.

### 2.1.7 Traffic Server

The caches in a Traffic Control CDN are servers running the Apache Traffic Server software. See [ATS documentation](#) for more information. Caches in a Traffic Control CDN are deployed in cache groups.

#### Cache Group

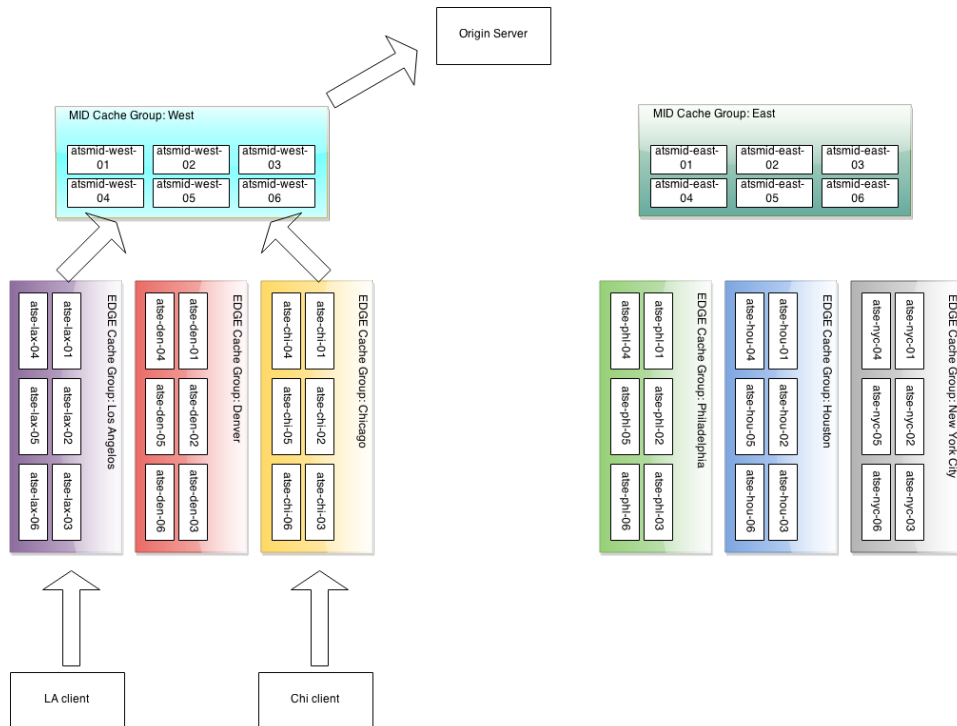
A cache group is a logical group of caches that Traffic Router tries to use as a combined cache. Traffic Router treats all servers in a cache group as though they are in the same physical location, though they are in fact only in the same region (network). A cache group has one single set of geographical coordinates even if the caches that make up the cache group are in different physical locations. The caches in a cache group are not aware of the other caches in the group - there is no clustering software or communications between caches in a cache group.

There are two types of cache groups: `EDGE` and `MID`. Traffic Control is a two tier system, where the clients get directed to the `EDGE` cache group. On cache miss, the cache in the `EDGE` cache group obtains content from a `MID` cache group, rather than the origin, which is shared with multiple `EDGE`s. `EDGE` cache groups are configured to have one single parent cache group.

**Note:** Often the EDGE to MID relationship is based on network distance, and does not necessarily match the geographic distance.

A cache group serves a particular part of the network as defined in the coverage zone file. See *The Coverage Zone File and ASN Table*.

Consider the example CDN below:



There are two MID tier cache groups, each assigned with three EDGES. The lax, den and chi EDGE locations are configured with the West MID as their parent, and the nyc, phl, and hou EDGES, are configured with the East MID as their parent. On a cache miss, the EDGES use their assigned parent.

All caches (and other servers) are assigned a Profile in Traffic Ops.

## ➔ Profile

A Profile is a set of configuration settings and parameters, applied to a server or deliveryservice. For a typical cache there are hundreds of configuration settings to apply. The Traffic Ops parameter view contains the defined settings, and bundled into groups using Profiles. Traffic Ops allows for duplication, comparison, import and export of Profiles.

### 2.1.8 Traffic Vault

Traffic Vault is a keystore used for storing the following types of information:

- **SSL Certificates**
  - Private Key
  - CRT

- CSR
- **DNSSEC Keys**
  - **Key Signing Key**
    - \* private key
    - \* public key
  - **Zone Signing Key**
    - \* private key
    - \* public key
- URL Signing Keys

As the name suggests, Traffic Vault is meant to be a “vault” of private keys that only certain users are allowed to access. In order to create, add, and retrieve keys a user must have admin privileges. Keys can be created via the Traffic Ops UI, but they can only be retrieved via the Traffic Ops API. The keystore used by Traffic Vault is [Riak](#). Traffic ops accesses Riak via https on port 8088. Traffic ops uses Riak’s rest API with username/password authentication. Information on the API can be found [here](#).



---

## Administrator's Guide

---

How to deploy and manage a Traffic Control CDN.

### 3.1 Administrator's Guide

Traffic Control is distributed in source form for the developer, but also as a binary package. This guide details how to install and configure a Traffic Control CDN using the binary packages, as well as how to perform common operations running a CDN.

When installing a complete CDN from scratch, a sample recommended order is:

1. Traffic Ops
2. Traffic Vault (Riak)
3. Traffic Monitor
4. Apache Traffic Server Mid-Tier Caches
5. Apache Traffic Server Edge Caches
6. Traffic Router
7. Traffic Stats
8. Traffic Portal

Once everything is installed, you will need to configure the servers to talk to each other. You will also need Origin server(s), which the Mid-Tier Cache(s) get content from. An Origin server is simply an HTTP(S) server which serves the content you wish to cache on the CDN.

#### 3.1.1 Traffic Ops - Installing

##### System Requirements

The user must have the following for a successful minimal install:

- CentOS 7
- 2 VMs with at least 2 vCPUs, 4GB RAM, 20 GB disk space each
- Access to Centos Base and epel repositories
- Access to [The Comprehensive Perl Archive Network \(CPAN\)](#)

As of version 2.0 only Postgres is supported as the database. This documentation assumes CentOS 7.2 and Postgresql 9.6.3. For a production install

## Navigating the Install

To begin the install:

### 1. Install Postgres

For a production install it is best to install postgres on it's own server/VM. To install postgres, on the postgres host (pg)

```
pg-$ sudo su -
pg-# yum -y update
pg-# yum -y install https://download.postgresql.org/pub/repos/yum/9.6/redhat/
    ↪ rhel-7-x86_64/pgdg-centos96-9.6-3.noarch.rpm
pg-# yum -y install postgresql96-server
pg-$ su - postgres
pg-$ /usr/pgsql-9.6/bin/initdb -A md5 -W # -W forces the user to provide a
    ↪ superuser (postgres) password
```

Edit `/var/lib/pgsql/9.6/data/pg_hba.conf` to allow your traffic ops app server access. For example if you are going to install traffic ops on `99.33.99.1` add:

```
host    all         all             99.33.99.1/32 md5
```

to the appropriate section of this file. Edit the `/var/lib/pgsql/9.6/data/postgresql.conf` file to add the appropriate `listen_addresses` or `listen_addresses = '*'`, set `timezone = 'UTC'`, and start the database:

```
pg-$ exit
pg-# systemctl enable postgresql-9.6
pg-# systemctl start postgresql-9.6
pg-# systemctl status postgresql-9.6
```

### 2. Build Traffic Ops

Build a Traffic Ops rpm using the instructions under the [Building Traffic Control](#) page.

### 3. Install Postgresql

Install the postgresql 9.6 yum repository access.

```
to-$ sudo su -
to-# yum -y install https://download.postgresql.org/pub/repos/yum/9.6/redhat/
    ↪ rhel-7-x86_64/pgdg-centos96-9.6-3.noarch.rpm
```

### 4. Install the rpm built in step 2.

```
to-# yum -y install ./dist/traffic_ops-2.0.0-xxxx.yyyyyyy.el7.x86_64.rpm
```

Install some additional packages that it depends on that were not installed as dependencies in the previous step (these are for the 2.0.0 install, this may change, but the pre-installs won't hurt):

```
to-# yum -y install git
to-# wget -q https://storage.googleapis.com/golang/go1.8.3.linux-amd64.tar.gz
to-# tar -C /usr/local -xzf go1.8.3.linux-amd64.tar.gz
to-# PATH=$PATH:/usr/local/go/bin          # go bins are needed in the
→path for postinstall
to-# go get bitbucket.org/liamstask/goose/cmd/goose
```

At this point you should be able to login to the database from the `to` host to the `pg` host like:

```
to-# psql -h 99.33.99.1 -U postgres
Password for user postgres:
psql (9.6.3)
Type "help" for help.

postgres=#
```

Use this connectivity to create the user and database. In this example, we use user: `traffic_ops`, password: `tcr0cks`, database: `traffic_ops`:

```
to-# psql -U postgres -h 99.33.99.1 -c "CREATE USER traffic_ops WITH
→ENCRYPTED PASSWORD 'tcr0cks';"
Password for user postgres:
CREATE ROLE
to-# createdb traffic_ops --owner traffic_ops -U postgres -h 99.33.99.1
Password:
to-#
```

Now, run the following command as root: `/opt/traffic_ops/install/bin/postinstall`

The `postinstall` will first get all packages needed from CPAN. This may take a while, expect up to 30 minutes on the first install. If there are any prompts in this phase, please just answer with the defaults (some CPAN installs can prompt for install questions).

When this phase is complete, you will see:

```
Complete! Modules were installed into /opt/traffic_ops/app/local
```

Some additional files will be installed, and then it will proceed with the next phase of the install, where it will ask you about the local environment for your CDN. Please make sure you remember all your answers and the database answers match the database information previously used to create the database.

Example output:

```
===== /opt/traffic_ops/app/conf/production/database.conf =====
Database type [Pg]:
Database type: Pg
Database name [traffic_ops]:
Database name: traffic_ops
Database server hostname IP or FQDN [localhost]: 99.33.99.1
Database server hostname IP or FQDN: 99.33.99.1
Database port number [5432]:
Database port number: 5432
Traffic Ops database user [traffic_ops]:
Traffic Ops database user: traffic_ops
Password for Traffic Ops database user:
```

(continues on next page)

(continued from previous page)

```

Re-Enter Password for Traffic Ops database user:
Writing json to /opt/traffic_ops/app/conf/production/database.conf
Database configuration has been saved
=====opt/traffic_ops/app/db/dbconf.yml=====
Database server root (admin) user [postgres]:
Database server root (admin) user: postgres
Password for database server admin:
Re-Enter Password for database server admin:
Download Maxmind Database? [yes]:
Download Maxmind Database?: yes
=====opt/traffic_ops/app/conf/cdn.conf=====
Generate a new secret? [yes]:
Generate a new secret?: yes
Number of secrets to keep? [10]:
Number of secrets to keep?: 10
Not setting up ldap
=====opt/traffic_ops/install/data/json/users.json=====
Administration username for Traffic Ops [admin]:
Administration username for Traffic Ops: admin
Password for the admin user:
Re-Enter Password for the admin user:
Writing json to /opt/traffic_ops/install/data/json/users.json
=====opt/traffic_ops/install/data/json/openssl_configuration.
↪json=====
Do you want to generate a certificate? [yes]:
Country Name (2 letter code): US
State or Province Name (full name): CO
Locality Name (eg, city): Denver
Organization Name (eg, company): Super CDN, Inc
Organizational Unit Name (eg, section):
Common Name (eg, your name or your server's hostname):
RSA Passphrase:
Re-Enter RSA Passphrase:
=====opt/traffic_ops/install/data/json/profiles.json=====
Traffic Ops url [https://localhost]:
Traffic Ops url: https://localhost
Human-readable CDN Name. (No whitespace, please) [kabletown_cdn]: blue cdn
Human-readable CDN Name. (No whitespace, please): blue cdn
DNS sub-domain for which your CDN is authoritative [cdn1.kabletown.net]: ↵
↪blue-cdn.supercdn.net
DNS sub-domain for which your CDN is authoritative: blue-cdn.supercdn.net
Writing json to /opt/traffic_ops/install/data/json/profiles.json
Downloading Maxmind data
--2017-06-11 15:32:41-- http://geolite.maxmind.com/download/geoip/database/
↪GeoLite2-City.mmdb.gz
Resolving geolite.maxmind.com (geolite.maxmind.com)... ↵
↪2400:cb00:2048:1::6810:262f, 2400:cb00:2048:1::6810:252f, 104.16.38.47, ...
Connecting to geolite.maxmind.com (geolite.maxmind.
↪com) |2400:cb00:2048:1::6810:262f|:80... connected.

... much SQL output skipped

Starting Traffic Ops
Restarting traffic_ops (via systemctl): [ OK ]
Waiting for Traffic Ops to restart
Success! Postinstall complete.

```

(continues on next page)

(continued from previous page)

to-# ifconfig

Explanation of the information that needs to be provided:

Field	Description
Database type	Pg
Database name	The name of the database Traffic Ops uses to store the configuration information
Database server hostname IP or FQDN	The hostname of the database server
Database port number	The database port number
Traffic Ops database user	The username Traffic Ops will use to read/write from the database
Password for traffic ops	The password for the above database user
Database server root (admin) user name	Privileged database user that has permission to create the database and user for Traffic Ops
Database server root (admin) user password	The password for the above privileged database user
Traffic Ops url	The URL to connect to this instance of Traffic Ops, usually <a href="https://&lt;traffic ops host FQDN&gt;/">https://&lt;traffic ops host FQDN&gt;/</a>
Human-readable CDN Name	The name of the first CDN traffic Ops will be managing
DNS sub-domain for which your CDN is authoritative	The DNS domain that will be delegated to this Traffic Control CDN
Administration username for Traffic Ops	The Administration (highest privilege) Traffic Ops user to create; use this user to login for the first time and create other users
Password for the admin user	The password for the above user

Traffic Ops is now installed!

**To complete the Traffic Ops Setup See:** [Traffic Ops - Default Profiles](#)

## Upgrading Traffic Ops

To upgrade:

1. Enter the following command:`service traffic_ops stop`
2. Enter the following command:`yum upgrade traffic_ops`
3. Enter the following command from the `/opt/traffic_ops/app` directory: `PERL5LIB=/opt/traffic_ops/app/lib:/opt/traffic_ops/app/local/lib/perl5 ./db/admin.pl --env production upgrade`
4. See [Traffic Ops - Installing](#) to run `postinstall`.
5. Enter the following command:`service traffic_ops start`

### 3.1.2 Traffic Ops - Default Profiles

Traffic Ops has the concept of *Parameters and Profiles*, which are an integral function within Traffic Ops. To get started, a set of default Traffic Ops profiles need to be imported into Traffic Ops to get started to support Traffic Control components Traffic Router, Traffic Monitor, and Apache Traffic Server.

[Download Default Profiles from here](#)

#### Minimum Traffic Ops Profiles needed

- EDGE\_ATS\_<version>\_<platform>\_PROFILE.traffic\_ops
- MID\_ATS\_<version>\_<platform>\_PROFILE.traffic\_ops
- TRAFFIC\_MONITOR\_PROFILE.traffic\_ops
- TRAFFIC\_ROUTER\_PROFILE.traffic\_ops
- TRAFFIC\_STATS\_PROFILE.traffic\_ops

#### Steps to Import a Profile

1. Sign into Traffic Ops
2. Navigate to 'Parameters->Select Profile'
3. Click the "Import Profile" button at the bottom
4. Choose the specific profile you want to import from your download directory
5. Click 'Submit'
6. Continue these steps for each *Minimum Traffic Ops Profiles needed* above

### 3.1.3 Traffic Ops - Migrating from 1.x to 2.x

In Traffic Ops 2.x MySQL was removed and Postgres was replaced as the database of choice for the unforeseen future. A Docker-based migration tool was developed to help with that conversion using an open source Postgres tool called [pgloader](#). The following instructions will help configuring the Migration tool

#### System Requirements

The user must have the following for a successful minimal install:

- CentOS 7.2+
- Docker installed (this migration was tested against version **docker-engine-selinux-17.05.0.ce-1.el7.centos.noarch.rpm**)
- Postgres has been installed according to [Traffic Ops - Installing](#)

#### Setup the traffic\_ops\_db directory

Modify /opt dir permission to make it writable and owned by postgres:postgres

```
$ sudo chmod 755 /opt
```

Download the Traffic Control tarball for 2.0.0

```
$ cd /opt
$ wget https://dist.apache.org/repos/dist/release/incubator/trafficcontrol/
↳<tarball_version>
```

Extract the **traffic\_ops\_db** dir to **/opt/traffic\_ops\_db**

```
$ tar -zxvf trafficcontrol-incubating-<version>.tar.gz --strip=1_
↳trafficcontrol-incubating-<version>/traffic_ops_db
$ sudo chown -R postgres:postgres /opt/traffic_ops_db
```

## Migration Preparation

Be sure there is connectivity between your MySQL server's IP address/port and your Postgres server's IP address/port.

## Navigating the Database Migration

Begin the database migration after settings up the **/opt/traffic\_ops\_db** directory

Switch to the postgres user so permissions stay intact.

```
$ su - postgres
$ cd /opt/traffic_ops_db/
```

1. Configure the **/opt/traffic\_ops\_db/pg-migration/mysql-to-postgres.env** migration for your source MySQL and target Postgres settings
2. Run the migration, watch the console output for any errors (it may take some time)

```
$ ./migrate.sh
```

Your MySQL data should now be ported into your new instance of Postgres!

## 3.1.4 Traffic Ops - Migrating from 2.0 to 2.2

### Apache Traffic Server 7.x (Cachekey Plugin)

In Traffic Ops 2.2 we have added support for Apache Traffic Server 7.x. With 7.x comes support for the new cachekey plugin which replaces the cacheurl plugin which is now deprecated. While not needed immediately it is recommended to start replacing cacheurl usages with cachekey as soon as possible because ATS 6.x already supports the new cachekey plugin.

It is also recommended to thoroughly vet your cachekey replacement by comparing with an existing key value. There are inconsistencies in the 6.x version of cachekey which have been fixed in 7.x (or require this patch([cachekeypatch](#)) on 6.x to match 7.x). So to ensure you have a matching key value you should use the xdebug plugin before fully implementing your cachekey replacement.

First if you are currently using a regex for your delivery service you will have to remove that existing value. Then you will need to make a new DS profile and assign parameters in it to the cachekey.config file.

Some common parameters are

```
static-prefix      - This is used for a simple domain replacement
separator         - Used by cachekey and in general is always a single space
remove-path       - Removes path information from the URL
remove-all-params - Removes parameters from the URL
capture-prefix-uri - This is usually used in combination with remove-path and remove-
↳all-params.
                    Capture-prefix-uri will let you use your own full regex value,
↳for non simple cases
```

## Examples of Cacheurl to Cachekey Replacements

### Original regex value:

```
http://test.net/(.*) http://test-cdn.net/$1
```

### Cachekey parameters:

Parameter	File	Value
static-prefix	cachekey.config	http://test-cdn.net/
separator	cachekey.config	(empty space)

### Original regex value:

```
http://([?]+)(?:?|$) http://test-cdn.net/$1
```

### Cachekey parameters:

Parameter	File	Value
remove-path	cachekey.config	true
remove-all-params	cachekey.config	true
separator	cachekey.config	(empty space)
capture-prefix-uri	cachekey.config	/https?:\\/(\\([?]*\\)/http:\\/\\/test-cdn.net\\/\$1/

Also note the `s?` used here so that both http and https requests will end up with the same key value

### Original regex value:

```
http://test.net/([?]+)(?:\\?|$) http://test-cdn.net/$1
```

### Cachekey parameters:

Parameter	File	Value
static-prefix	cachekey.config	http://test-cdn.net/
separator	cachekey.config	(empty space)
remove-all-params	cachekey.config	true

Further documentation on the cachekey plugin can be found at [ApacheTrafficServerDocs](#)

## 3.1.5 Traffic Ops - Configuring

Follow the steps below to configure the newly installed Traffic Ops Instance.



## Installing the SSL Cert

By default, Traffic Ops runs as an SSL web server, and a certificate needs to be installed.

### Self-signed Certificate (Development)

Example Procedure:

```
$ openssl genrsa -des3 -passout pass:x -out localhost.pass.key 2048
Generating RSA private key, 2048 bit long modulus
...
$ openssl rsa -passin pass:x -in localhost.pass.key -out localhost.key
writing RSA key
$ rm localhost.pass.key

$ openssl req -new -key localhost.key -out localhost.csr
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [XX]:US<enter>
State or Province Name (full name) []:CO<enter>
Locality Name (eg, city) [Default City]:Denver<enter>
Organization Name (eg, company) [Default Company Ltd]: <enter>
Organizational Unit Name (eg, section) []: <enter>
Common Name (eg, your name or your server's hostname) []: <enter>
Email Address []: <enter>

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []: pass<enter>
An optional company name []: <enter>
$ openssl x509 -req -sha256 -days 365 -in localhost.csr -signkey localhost.
→key -out localhost.crt
Signature ok
subject=/C=US/ST=CO/L=Denver/O=Default Company Ltd
Getting Private key
$ sudo cp localhost.crt /etc/pki/tls/certs
$ sudo cp localhost.key /etc/pki/tls/private
$ sudo chown trafops:trafops /etc/pki/tls/certs/localhost.crt
$ sudo chown trafops:trafops /etc/pki/tls/private/localhost.key
```

### Certificate from Certificate Authority (Production)

---

**Note:** You will need to know the appropriate answers when generating the certificate request file *trafficopss.csr* below.

---

Example Procedure:

```
$ openssl genrsa -des3 -passout pass:x -out trafficopss.pass.key 2048
Generating RSA private key, 2048 bit long modulus
```

(continues on next page)

(continued from previous page)

```

...
$ openssl rsa -passin pass:x -in trafficops.pass.key -out trafficops.key
writing RSA key
$ rm localhost.pass.key

Generate the Certificate Signing Request (CSR) file needed for Certificate Authority
↳ (CA) request.

$ openssl req -new -key trafficops.key -out trafficops.csr
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [XX]: <enter country code>
State or Province Name (full name) []: <enter state or province>
Locality Name (eg, city) [Default City]: <enter locality name>
Organization Name (eg, company) [Default Company Ltd]: <enter organization name>
Organizational Unit Name (eg, section) []: <enter organizational unit name>
Common Name (eg, your name or your server's hostname) []: <enter server's hostname
↳ name>
Email Address []: <enter e-mail address>

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []: <enter challenge password>
An optional company name []: <enter>
$ sudo cp trafficops.key /etc/pki/tls/private
$ sudo chown trafops:trafops /etc/pki/tls/private/trafficops.key

You must then take the output file trafficops.csr and submit a request to your
↳ Certificate Authority (CA).
Once you get approved and receive your trafficops.crt file:

$ sudo cp trafficops.crt /etc/pki/tls/certs
$ sudo chown trafops:trafops /etc/pki/tls/certs/trafficops.crt

If necessary, install the CA certificates .pem and .crt in /etc/pki/tls/certs.

You will need to update the file /opt/traffic_ops/app/conf/cdn.conf with the
↳ following changes:
...
e.g. given trafficops.crt and trafficops.key
'hypnotoad' => ...
'listen' => 'https://[::]:443?cert=/etc/pki/tls/certs/trafficops.crt&key=/
↳ etc/pki/tls/private/trafficops.key&ca=/etc/pki/tls/certs/localhost.ca&verify=0x00&
↳ ciphers=AES128-GCM-SHA256:HIGH:!RC4:!MD5:!aNULL:!EDH:!ED'
...

```

## Content Delivery Networks

## Profile Parameters

Many of the settings for the different servers in a Traffic Control CDN are controlled by parameters in the parameter view of Traffic Ops. Parameters are grouped in profiles and profiles are assigned to a server or a deliveryservice. For a typical cache there are hundreds of configuration settings to apply. The Traffic Ops parameter view contains the defined settings. To make life easier, Traffic Ops allows for duplication, comparison, import and export of Profiles. Traffic Ops also has a “Global profile” - the parameters in this profile are going to be applied to all servers in the Traffic Ops instance, or apply to Traffic Ops themselves. These parameters are:

Name	Con-fig file	Value
tm.url	global	The URL where this Traffic Ops instance is being served from.
tm.rev_proxy	global	Not required. The URL where the Traffic Ops Config file cache instance is being served from. Requires Traffic Ops ORT 2.1 and above. When configured, ORT will request configuration files via this fqdn, which should be setup as a reverse proxy to the Traffic Ops host or hosts. Suggested cache lifetime for these files is ~3 minutes or less. This setting allows for greater scaleability of a CDN maintained by Traffic Ops by caching configuration files of profile and cdn scope.
tm.toolname	global	The name of the Traffic Ops tool. Usually “Traffic Ops”. Used in the About screen and in the comments headers of the files generated.
tm.infourl	global	This is the “for more information go here” URL, which is visible in the About page.
tm.logourl	global	This is the URL of the logo for Traffic Ops and can be relative if the logo is under traf-ops/app/public.
tm.instanceid	global	The name of the Traffic Ops instance. Can be used when multiple instances are active. Visible in the About page.
tm.traffic_monitor_proxy	global	When collecting stats from Traffic Monitor, Traffic Ops uses this forward proxy to pull the stats through. This can be any of the MID tier caches, or a forward cache specifically deployed for this purpose. Setting this variable can significantly lighten the load on the Traffic Monitor system and it is recommended to set this parameter on a production system.
ge-olo-ca-tion.polling.url	CR-Con-fig.json	The location to get the GeoLiteCity database from.
ge-olo-ca-tion6.polling.url	CR-Con-fig.json	The location to get the IPv6 GeoLiteCity database from.

These parameters should be set to reflect the local environment.

After running the postinstall script, Traffic Ops has the following profiles pre-loaded:

Name	Description
EDGE1	The profile to be applied to the latest supported version of ATS, when running as an EDGE cache
TR1	The profile to be applied to the latest version of Traffic Router
TM1	The profile to be applied to the latest version of Traffic Monitor
MID1	The profile to be applied to the latest supported version of ATS, when running as an MID cache
RIAK_ALL	Riak profile for all CDNs to be applied to the Traffic Vault servers

**Note:** The Traffic Server profiles contain some information that is specific to the hardware being used (most notably the disk configuration), so some parameters will have to be changed to reflect your configuration. Future releases of

Traffic Control will separate the hardware and software profiles so it is easier to “mix-and-match” different hardware configurations.

Below is a list of cache parameters that are likely to need changes from the default profiles shipped with Traffic Ops:

Name	Config file	Description
allow_ip	as-tats.config	This is a comma separated list of IPv4 CIDR blocks that will have access to the astats statistics on the caches. The Traffic Monitor IP addresses have to be included in this, if they are using IPv4 to monitor the caches.
allow_ip6	as-tats.config	This is a comma separated list of IPv6 CIDR blocks that will have access to the astats statistics on the caches. The Traffic Monitor IP addresses have to be included in this, if they are using IPv6 to monitor the caches.
Drive_Prefix	storage.config	The device path start of the disks. For example, if you have /dev/sda through /dev/sdf set this to /dev/sd
Drive_Letters	storage.config	The letter part of the disks, in the same example as above set this to a, b, c, d, e, f
purge_allow_ip	ip_allow.config	The IP address range that is allowed to execute the PURGE method on the caches (not related to <i>Invalidate Content</i> )
coalesce_masklen_v4	ip_allow.config	The masklen to use when coalescing v4 networks into one line using <a href="http://search.cpan.org/~miker/NetAddr-IP-4.078/IP.pm">http://search.cpan.org/~miker/NetAddr-IP-4.078/IP.pm</a>
coalesce_number_v4	ip_allow.config	The number to use when coalescing v4 networks into one line using <a href="http://search.cpan.org/~miker/NetAddr-IP-4.078/IP.pm">http://search.cpan.org/~miker/NetAddr-IP-4.078/IP.pm</a>
coalesce_masklen_v6	ip_allow.config	The masklen to use when coalescing v6 networks into one line using <a href="http://search.cpan.org/~miker/NetAddr-IP-4.078/IP.pm">http://search.cpan.org/~miker/NetAddr-IP-4.078/IP.pm</a>
coalesce_masklen_v6	ip_allow.config	The masklen to use when coalescing v6 networks into one line using <a href="http://search.cpan.org/~miker/NetAddr-IP-4.078/IP.pm">http://search.cpan.org/~miker/NetAddr-IP-4.078/IP.pm</a>
health.threshold.loadavg	cache.properties	The Unix load average at which Traffic Router will stop sending traffic to this cache
health.threshold.availableBandwidthInKbps	cache.properties	The amount of bandwidth that Traffic Router will try to keep available on the cache. For example: “”>1500000” means stop sending new traffic to this cache when traffic is at 8.5Gbps on a 10Gbps interface.

Below is a list of Traffic Server plugins that need to be configured in the parameter table:

Name	Config file	Description	Details
as-tats_over_http	package	The package version for the astats_over_http plugin.	<a href="#">as-tats_over_http</a>
trafficserver	package	The package version for the trafficserver plugin.	<a href="#">trafficserver</a>
regex_revalidate	plugin.config	The config to be used for regex_revalidate. For example: -config regex_revalidate.config	<a href="#">regex_revalidate</a>
remap_stats	plugin.config	The config to be used for remap_stats. Value is left blank.	<a href="#">remap_stats</a>

Below is a list of cache parameters for special configuration, which are unlikely to need changes, but may be useful in particular circumstances:

Name	Config file	Description
not_a_parent	ent.config	This is a boolean flag and is considered true if it exists and has any value except 'false'. This prevents servers with this parameter in their profile from being inserted into the parent.config generated for servers with this server's cachegroup as a parent of their cachegroup. This is primarily useful for when edge caches are configured to have a cachegroup of other edge caches as parents (a highly unusual configuration), and it is necessary to exclude some, but not all, edges in the parent cachegroup from the parent.config (for example, because they lack necessary capabilities), but still have all edges in the same cachegroup in order to take traffic from ordinary delivery services at that cachegroup's geo location. Once again, this is a highly unusual scenario, and under ordinary circumstances this parameter should not exist.

## Regions, Locations and Cache Groups

All servers have to have a *location*, which is their physical location. Each location is part of a *region*, and each region is part of a *division*. For Example, *Denver* could be a location in the *Mile High* region and that region could be part of the *West* division. Enter your divisions first in *Misc->Divisions*, then enter the regions in *Misc->Regions*, referencing the divisions entered, and finally, enter the physical locations in *Misc->Locations*, referencing the regions entered.

All servers also have to be part of a *cache group*. A cache group is a logical grouping of caches, that don't have to be in the same physical location (in fact, usually a cache group is spread across minimally 2 physical Locations for redundancy purposes), but share geo coordinates for content routing purposes. JvD to add more.

## Configuring Content Purge

Content purge using ATS is not simple; there is no file system to delete files/directories from, and in large caches it can be hard to delete a simple regular expression from the cache. This is why Traffic Control uses the [Regex Revalidate Plugin](#) to purge content from the system. We don't actually remove the content, we have a check that gets run before each request on each cache to see if this request matches a list of regular expressions, and if it does, we force a revalidation to the origin, making the original content inaccessible. The `regex_revalidate` plugin will monitor its config file, and will pick up changes to it without a `traffic_line -x` signal to ATS. Changes to this file need to be distributed to the highest tier (MID) caches in the CDN before they are distributed to the lower tiers, to prevent filling the lower tiers with the content that should be purged from the higher tiers without hitting the origin. This is why the `ort` script (see [Configuring Traffic Server](#)) will by default push out config changes to MID first, confirm that they have all been updated, and then push out the changes to the lower tiers. In large CDNs, this can make the distribution and time to activation of the purge too long, and because of that there is the option to not distribute the `regex_revalidate.config` file using the `ort` script, but to do this using other means. By default, Traffic Ops will use `ort` to distribute the `regex_revalidate.config` file.

Content Purge is controlled by the following parameters in the profile of the cache:

Name	Con-fig file	Description	Details
lo-ca-tion	regex_revalidate.config	Where the file should be in on the cache	The presence of this parameter tells ort to distribute this file; delete this parameter from the profile if this file is distributed using other means.
maxRevalDuration-Days	regex_revalidate.config	The longest time a purge can be active	To prevent a build up of many checks before each request, this is longest time the system will allow
regex_revalidate	regex_revalidate.config	The config be used for regex_revalidate. For example: <code>-config regex_revalidate.config</code>	<code>regex_revalidate</code>
use_reval_pending	global.config	Configures Traffic Ops to use separate reval_pending flag for each cache.	When this flag is in use ORT will check for a new regex_revalidate.config every 60 seconds in syncds mode during the dispersal timer. This will also allow ORT to be run in revalidate mode, which will check for and clear the reval_pending flag. This can be set to run via cron task. Enable with a value of 1. Use of this feature requires Traffic Ops 2.1 and above. Parameter should be assigned to the GLOBAL profile.

Note that the TTL the administrator enters in the purge request should be longer than the TTL of the content to ensure the bad content will not be used. If the CDN is serving content of unknown, or unlimited TTL, the administrator should consider using `proxy-config-http-cache-guaranteed-min-lifetime` to limit the maximum time an object can be in the cache before it is considered stale, and set that to the same value as `maxRevalDurationDays` (Note that the former is in seconds and the latter is in days, so convert appropriately).

## Creating the CentOS Kickstart File

The kickstart file is a text file, containing a list of items, each identified by a keyword. You can create it by using the Kickstart Configurator application, or writing it from scratch. The Red Hat Enterprise Linux installation program also creates a sample kickstart file based on the options that you selected during installation. It is written to the file `/root/anaconda-ks.cfg`. This file is editable using most text editors that can save files as ASCII text.

To generate ISO, the CentOS Kickstart is necessary:

1. Create a kickstart file.
2. Create a boot media with the kickstart file or make the kickstart file available on the network.
3. Make the installation tree available.
4. Start the kickstart installation.

Create a `ks.src` file in the root of the selection location. See the example below:

```
mkdir newdir
cd newdir/
cp -r ../centos65/* .
vim ks.src
vim isolinux/isolinux.cfg
```

(continues on next page)

(continued from previous page)

```
cd vim osversions.cfg
vim osversions.cfg
```

This is a standard kickstart formatted file that the generate ISO process uses to create the kickstart (ks.cfg) file for the install. The generate ISO process uses the ks.src, overwriting any information set in the Generate ISO tab in Traffic Ops, creating ks.cfg.

---

**Note:** Streamline your install folder for under 1GB, which assists in creating a CD.

---

### See also:

For in-depth instructions, please see [Kickstart Installation](#)

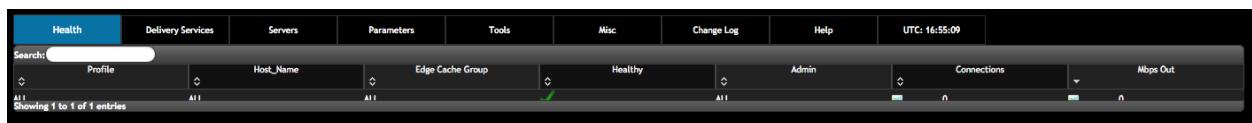
## Configuring the Go Application

Traffic Ops is in the process of migrating from Perl to Go, and currently runs as two applications. The Go application serves all endpoints which have been rewritten in the Go language, and transparently proxies all other requests to the old Perl application. Both applications are installed by the RPM, and both run as a single service. When the project has fully migrated to Go, the Perl application will be removed, and the RPM and service will consist solely of the Go application.

By default, the postinstall script configures the Go application to behave and transparently serve as the old Perl Traffic Ops did in previous versions. This includes reading the old `cdn.conf` and `database.conf` config files, and logging to the old `access.log` location. However, if you wish to customize the Go Traffic Ops application, you can do so by running it with the `-oldcfg=false` argument. By default, it will then look for a config file in `/opt/traffic_ops/conf/traffic_ops_golang.config`. The new config file location may also be customized via the `-cfg` flag. A sample config file is installed by the RPM at `/opt/traffic_ops/conf/traffic_ops_golang.config`. If you wish to run the new Go Traffic Ops application as a service with a new config file, the `-oldcfg=false` and `-cfg` flags may be added to the `start` function in the service file, located by default at `etc/init.d/traffic_ops`.

## 3.1.6 Traffic Ops - Using

### The Traffic Ops Menu



The following tabs are available in the menu at the top of the Traffic Ops user interface.

- **Health**

Information on the health of the system. Hover over this tab to get to the following options:

Option	Description
Table View	A real time view into the main performance indicators of the CDNs managed by Traffic Control. This view is sourced directly by the Traffic Monitor data and is updated every 10 seconds. This is the default screen of Traffic Ops. See <a href="#">The Health Table</a> for details.
Graph View	A real graphical time view into the main performance indicators of the CDNs managed by Traffic Control. This view is sourced by the Traffic Monitor data and is updated every 10 seconds. On loading, this screen will show a history of 24 hours of data from Traffic Stats See <a href="#">Graph View</a> for details.
Server Checks	A table showing the results of the periodic check extension scripts that are run. See <a href="#">Server Checks</a>
Daily Summary	A graph displaying the daily peaks of bandwidth, overall bytes served per day, and overall bytes served since initial installation per CDN.

- **Delivery Services**

The main Delivery Service table. This is where you Create/Read/Update/Delete Delivery Services of all types. Hover over to get the following sub option:

Option	Description
Federations	Add/Edit/Delete Federation Mappings.

- **Servers**

The main Servers table. This is where you Create/Read/Update/Delete servers of all types. Click the main tab to get to the main table, and hover over to get these sub options:

Option	Description
Upload Server CSV	Bulk add of servers from a csv file. See <a href="#">Bulk Upload Server</a>

- **Parameters**

Parameters and Profiles can be edited here. Hover over the tab to get the following options:

Option	Description
Global Profile	The table of global parameters. See <a href="#">Profile Parameters</a> . This is where you Create/Read/Update/Delete parameters in the Global profile
All Cache Groups	The table of all parameters <i>that are assigned to a cachegroup</i> - this may be slow to pull up, as there can be thousands of parameters.
All Profiles	The table of all parameters <i>that are assigned to a profile</i> - this may be slow to pull up, as there can be thousands of parameters.
Select Profile	Select the parameter list by profile first, then get a table of just the parameters for that profile.
Export Profile	Profiles can be exported from one Traffic Ops instance to another using ‘Select Profile’ and under the “Profile Details” dialog for the desired profile
Import Profile	Profiles can be imported from one Traffic Ops instance to another using the button “Import Profile” after using the “Export Profile” feature
Orphaned Parameters	A table of parameters that are not associated to any profile of cache group. These parameters either should be deleted or associated with a profile of cache group.

- **Tools**



Tools for working with Traffic Ops and it's servers. Hover over this tab to get the following options:

Option	Description
Generate ISO	Generate a bootable image for any of the servers in the Servers table (or any server for that matter). See <a href="#">Generate ISO</a>
Queue Updates	Send Updates to the caches. See <a href="#">Queue Updates and Snapshot CRConfig</a>
DB Dump	Backup the Database to a .sql file.
Snapshot CR-Config	Send updates to the Traffic Monitor / Traffic Router servers. See <a href="#">Queue Updates and Snapshot CRConfig</a>
Invalidate Content	Invalidate or purge content from all caches in the CDN. See <a href="#">Invalidate Content</a>
Manage DNSSEC keys	Manage DNSSEC Keys for a chosen CDN.

- **Misc**

Miscellaneous editing options. Hover over this tab to get the following options:

Option	Description
CDNs	Create/Read/Update/Delete CDNs
Cache Groups	Create/Read/Update/Delete cache groups
Users	Create/Read/Update/Delete users
Profiles	Create/Read/Update/Delete profiles. See <a href="#">Parameters and Profiles</a>
Net-works(ASNs)	Create/Read/Update/Delete Autonomous System Numbers See <a href="#">The Coverage Zone File and ASN Table</a>
Hardware	Get detailed hardware information (note: this should be moved to a Traffic Ops Extension)
Data Types	Create/Read/Update/Delete data types
Divisions	Create/Read/Update/Delete divisions
Regions	Create/Read/Update/Delete regions
Physical Locations	Create/Read/Update/Delete locations

- **ChangeLog**

The Changelog table displays the changes that are being made to the Traffic Ops database through the Traffic Ops user interface. This tab will show the number of changes since you last visited this tab in (brackets) since the last time you visited this tab. There are currently no sub menus for this tab.

- **Help**


Help for Traffic Ops and Traffic Control. Hover over this tab to get the following options:

Option	Description
About	Traffic Ops information, such as version, database information, etc
Release Notes	Release notes for the most recent releases of Traffic Ops
Logout	Logout from Traffic Ops

## Health

## The Health Table

The Health table is the default landing screen for Traffic Ops, it displays the status of the EDGE caches in a table form directly from Traffic Monitor (bypassing Traffic Stats), sorted by Mbps Out. The columns in this table are:

- **Profile:** the Profile of this server or ALL, meaning this row shows data for multiple servers, and the row shows the sum of all values.
- **Host Name:** the host name of the server or ALL, meaning this row shows data for multiple servers, and the row shows the sum of all values.
- **Edge Cache Group:** the edge cache group short name or ALL, meaning this row shows data for multiple servers, and the row shows the sum of all values.
- **Healthy:** indicates if this cache is healthy according to the Health Protocol. A row with ALL in any of the columns will always show a , this column is valid only for individual EDGE caches.
- **Admin:** shows the administrative status of the server.
- **Connections:** the number of connections this cache (or group of caches) has open (`ats.proxy.process.http.current_client_connections` from ATS).
- **Mbps Out:** the bandwidth being served out if this cache (or group of caches)

Since the top line has ALL, ALL, ALL, it shows the total connections and bandwidth for all caches managed by this instance of Traffic Ops.

## Graph View

The Graph View shows a live view of the last 24 hours of bits per seconds served and open connections at the edge in a graph. This data is sourced from Traffic Stats. If there are 2 CDNs configured, this view will show the statistics for both, and the graphs are stacked. On the left-hand side, the totals and immediate values as well as the percentage of total possible capacity are displayed. This view is updated every 10 seconds.

## Server Checks

The server checks page is intended to give an overview of the Servers managed by Traffic Control as well as their status. This data comes from [Traffic Ops extensions](#).



Name	Description
ILO	Ping the iLO interface for EDGE or MID servers
10G	Ping the IPv4 address of the EDGE or MID servers
10G6	Ping the IPv6 address of the EDGE or MID servers
MTU	Ping the EDGE or MID using the configured MTU from Traffic Ops
FQDN	DNS check that matches what the DNS servers responds with compared to what Traffic Ops has.
DSCP	Checks the DSCP value of packets from the edge server to the Traffic Ops server.
RTR	Content Router checks. Checks the health of the Content Routers. Checks the health of the caches using the Content Routers.
CHR	Cache Hit Ratio in percent.
CDU	Total Cache Disk Usage in percent.
ORT	Operational Readiness Test. Uses the ORT script on the edge and mid servers to determine if the configuration in Traffic Ops matches the configuration on the edge or mid. The user that this script runs as must have an ssh key on the edge servers.

## Daily Summary

Displays daily max gbps and bytes served for all CDNs. In order for the graphs to appear, the 'daily\_bw\_url' and 'daily\_served\_url' parameters need to be created, assigned to the global profile, and have a value of a grafana graph. For more information on configuring grafana, see the [Traffic Stats](#) section.

## Server

This view shows a table of all the servers in Traffic Ops. The table columns show the most important details of the

server. The **IPAddr** column is clickable to launch an `ssh://` link to this server. The  icon will link to a Traffic Stats graph of this server for caches, and the  will link to the server status pages for other server types.

## Server Types

These are the types of servers that can be managed in Traffic Ops:

Name	Description
EDGE	Edge Cache
MID	Mid Tier Cache
ORG	Origin
CCR	Traffic Router
RASCAL	Rascal health polling & reporting
TOOLS_SERVER	Ops hosts for management
RIAK	Riak keystore
SPLUNK	SPLUNK indexer search head etc
TRAFFIC_STATS	traffic_stats server
INFLUXDB	influxDb server

## Bulk Upload Server

TBD

## Delivery Service

The fields in the Delivery Service view are:

Name	Description
Active	When this is set to false Traffic Router will not serve DNS or HTTP responses for this delivery service.
Cache URL expression	Cache URL rule to apply to this delivery service. See <a href="#">ATS documentation on cacheurl</a> . <sup>1</sup>
CDN	The CDN in which the delivery service belongs to.
Check Path	A path (ex: /crossdomain.xml) to verify the connection to the origin server with. This can be used by Check Extension scripts to do periodic health checks against the delivery service.

Continued on next page

Table 1 – continued from previous page

Name	Description
Content Routing Type	The type of content routing this delivery service will use. See <i>Delivery Service Types</i> .
Deep Caching	(for HTTP routed delivery services only) When to do deep caching for this delivery service (see <i>Deep Caching</i> for more details): <ul style="list-style-type: none"> <li>• NEVER</li> <li>• ALWAYS</li> </ul>
Delivery Service DNS TTL	The Time To Live on the DNS record for the Traffic Router A and AAAA records (<routing-name>.<deliveryservice>.<cdn-domain>).
Delivery Services URLs	(Read Only) An example of how the delivery URL may start. This could be multiple rows if multiple HOST_REGEX entries have been entered.
Display Name	A human-readable name for the delivery service.
DNS Bypass CNAME	(For DNS routed delivery services only) This is a CNAME to respond to DNS requests with when a Delivery Service is unavailable.
DNS Bypass Ip	(For DNS routed delivery services only) This is the address to respond to A requests with when the max Bps or Max Tps for this delivery service are exceeded.
DNS Bypass IPv6	(For DNS routed delivery services only) This is the address to respond to AAAA requests with when the max Bps or Max Tps for this delivery service are exceeded.
DNS Bypass TTL	(For DNS routed delivery services only) This is the TTL to use for DNS Bypass responses when a delivery service is unavailable.
DSCP Tag	The DSCP value to mark IP packets to the client with.
Edge Header Rewrite Rules	Header Rewrite rules to apply for this delivery service at the EDGE tier. See <i>Header Rewrite Options and DSCP</i> . <sup>1</sup>
Geo Limit?	Some services are intended to be limited by geography. The possible settings are are: <ul style="list-style-type: none"> <li>• None - Do not limit by geography.</li> <li>• CZF only - If the requesting IP is not in the Coverage Zone File, do not serve the request.</li> <li>• CZF + US - If the requesting IP is not in the Coverage Zone File or not in the United States, do not serve the request.</li> </ul>
Geo Limit Redirect URL	(for HTTP routed delivery services only) This is the URL Traffic Router will redirect to when Geo Limit Failure. See <i>GeoLimit Failure Redirect feature</i>
Geo Limit Countries	A comma delimited list of Country Codes that will be allowed to make requests from the Delivery Service
Geo Miss Default Latitude	Default Latitude for this delivery service. When client localization fails for both Coverage Zone and Geo Lookup, the client will be routed as if it was at this lat.

Continued on next page

Table 1 – continued from previous page

Name	Description
Geo Miss Default Longitude	Default Longitude for this delivery service. When client localization fails for both Coverage Zone and Geo Lookup, the client will be routed as if it was at this long.
Geolocation Provider	An ordered list of Geolocation Providers to be used in make geo location decisions.
Global Max Mbps	The maximum bits per second this delivery service can serve across all EDGE caches before traffic will be diverted to the bypass destination. For a DNS delivery service, the Bypass Ipv4 or Ipv6 will be used (depending on whether this was a A or AAAA request), and for HTTP delivery services the Bypass FQDN will be used.
Global Max TPS	The maximum transactions this delivery service can serve across all EDGE caches before traffic will be diverted to the bypass destination. For a DNS delivery service, the Bypass Ipv4 or Ipv6 will be used (depending on whether this was a A or AAAA request), and for HTTP delivery services the Bypass FQDN will be used.
HTTP Bypass FQDN	(for HTTP routed delivery services only) This is the FQDN Traffic Router will redirect to (with the same path) when the max Bps or Max Tps for this delivery-service are exceeded.
Info URL	Info URL for this delivery service. To be consumed from the APIs by downstream tools (Portal).
Initial Dispersion	For HTTP delivery services - the number of caches that will be used by Traffic Router. Traffic Router will consistently return the same caches as long as they are healthy.
IPv6 Routing Enabled	When set to yes, the Traffic Router will respond to AAAA DNS requests for the routed name of this delivery service. Otherwise, only A records will be served.
Key (XML ID)	A unique string that identifies this delivery service.
Logs Enabled?	Whether or not Traffic Logs is enabled for the delivery service. This feature is currently incomplete without Traffic Logs
Long Description	Long description for this delivery service. To be consumed from the APIs by downstream tools (Portal).
Long Description 2	Another Long Description
Long Description 3	Another Long Description
Max DNS Answers	For DNS delivery services - the max number of DNS answers that will be returned by Traffic Router.
Mid Header Rewrite Rules	Header Rewrite rules to apply for this delivery service at the MID tier. See <a href="#">Header Rewrite Options and DSCP</a> . <sup>1</sup>
Multi Site Origin	Enable the Multi Site Origin feature for this delivery service. See <a href="#">Multi Site Origin</a>
Origin Server Base URL	The Origin Server's base URL. This includes the protocol (http or https). Example: <code>http://movies.origin.com</code>
Origin Shield (Pipe Delimited String)	Origin Shield string.
Profile	The profile for this delivery service.

Continued on next page

Table 1 – continued from previous page

Name	Description
Protocol	<p>The protocol to serve this delivery service to the clients with:</p> <ul style="list-style-type: none"> <li>• 0 http</li> <li>• 1 https</li> <li>• 2 both http and https</li> </ul>
Query String Handling	<p>How to treat query strings:</p> <ul style="list-style-type: none"> <li>• 0 use in cache key and hand up to origin - this means each unique query string is treated as a unique URL.</li> <li>• 1 Do not use in cache key, but pass up to origin - this means 2 URLs that are the same except for the query string will match, and cache HIT, while the origin still sees original query string in the request.</li> <li>• 2 Drop at edge - this means 2 URLs that are the same except for the query string will match, and cache HIT, while the origin will not see original query string in the request.</li> </ul> <p><b>Note:</b> Choosing to drop query strings at the edge will preclude the use of a Regex Remap Expression. See <a href="#">Regex Remap Expression</a>. To set the qstring without the use of regex remap, or for further options, see <a href="#">Qstring Handling</a>.</p>
Range Request Handling	<p>How to treat range requests:</p> <ul style="list-style-type: none"> <li>• 0 Do not cache (ranges requested from files that are already cached due to a non range request will be a HIT)</li> <li>• 1 Use the <code>background_fetch</code> plugin.</li> <li>• 2 Use the <code>cache_range_requests</code> plugin.</li> </ul>
Raw Remap Text	For HTTP and DNS deliveryservices, this will get added to the end of the remap line on the cache verbatim. For ANY_MAP deliveryservices this is the remap line. <sup>1</sup>
Regex Remap Expression	<p>Regex Remap rule to apply to this delivery service at the Edge tier. See <a href="#">ATS documentation on regex_remap</a>.<sup>1</sup></p> <p><b>Note:</b> you will not be able to save a Regex Remap Expression if you have Query String Handling set to drop query strings at the edge. See <a href="#">Regex Remap Expression</a>.</p>
Regional Geo-Blocking	<a href="#">See documentation here</a>
Routing Name	The routing name to use for the delivery FQDN, i.e. <code>&lt;routing-name&gt;.&lt;deliveryservice&gt;.&lt;cdn-domain&gt;</code> . It must be a valid hostname without periods. <sup>2</sup>
Signing Algorithm	
SSL Key Version	The most recent version for SSL keys, this is incremented each time new certificates are generated or pasted in traffic ops
Tenant	By selecting a tenant for the delivery service, you restrict access of that delivery service to only those users that belong to the same tenant or a parent tenant.

Continued on next page

Table 1 – continued from previous page

Name	Description
Traffic Router Request Headers	Additional request header to be logged by Traffic Router
Traffic Router Response Headers	Reponse headers to be returned by Traffic Router for each request

## Delivery Service Types

One of the most important settings when creating the delivery service is the selection of the delivery service *type*. This type determines the routing method and the primary storage for the delivery service.

Name	Description
HTTP	HTTP Content Routing - The Traffic Router DNS auth server returns its own IP address on DNS queries, and the client gets redirected to a specific cache in the nearest cache group using HTTP 302. Use this for long sessions like HLS/HDS/Smooth live streaming, where a longer setup time is not a problem.
DNS	DNS Content Routing - The Traffic Router DNS auth server returns an edge cache IP address to the client right away. The client will find the cache quickly but the Traffic Router can not route to a cache that already has this content in the cache group. Use this for smaller objects like web page images / objects.
HTTP_NO_CACHE	HTTP Content Routing, but the caches will not actually cache the content, they act as just proxies. The MID tier is bypassed.
HTTP_LIVE	HTTP Content routing, but where for “standard” HTTP content routing the objects are stored on disk, for this delivery service type the objects are stored on the RAM disks. Use this for linear TV. The MID tier is bypassed for this type.
HTTP_LIVE_NATNL	HTTP Content routing, same as HTTP_LIVE, but the MID tier is NOT bypassed.
DNS_LIVE	DNS Content routing, but where for “standard” DNS content routing the objects are stored on disk, for this delivery service type the objects are stored on the RAM disks. Use this for linear TV. The MID tier is NOT bypassed for this type.
DNS_LIVE_NATNL	DNS Content routing, same as DNS_LIVE_NATNL, but the MID tier is bypassed.
ANY_MAP	ANY_MAP is not known to Traffic Router. For this deliveryservice, the “Raw remap text” field in the input form will be used as the remap line on the cache.
STEERING	The Delivery Service will be used to route to other delivery services. The target delivery services and the routing weights for those delivery services will be defined by an admin or steering user. For more information see the <a href="#">steering feature</a> documentation
CLIENT_STEERING	STEERING except that a client can send a request to Traffic Router with a query param of <i>trred=false</i> and Traffic Router will return a HTTP 200 reponse with a body that contains a list of Delivery Service URLs that the client connect to. Therefore the client is doing the steering, not the Traffic Router.

**Note:** Once created, the Traffic Ops user interface does not allow you to change the delivery service type; the drop down is greyed out. There are many things that can go wrong when changing the type, and it is safer to delete the delivery service, and recreate it.

<sup>1</sup> These fields are not validated by Traffic Ops to be correct syntactically, and can cause Traffic Server to not start if invalid. Please use with caution.

<sup>2</sup> It is not recommended to change the Routing Name of a Delivery Service after deployment because this changes its Delivery FQDN (i.e. `<routing-name>.<deliveryservice>.<cdn-domain>`), which means that SSL certificates may need to be updated and clients using the Delivery Service will need to be transitioned to the new Delivery URL.

## Federations

Federations allow for other (federated) CDNs (at a different ISP, MSO, etc) to add a list of resolvers and a CNAME to a delivery service Traffic Ops. When a request is made from one of federated CDN's clients, Traffic Router will return the CNAME configured in the federation mapping. This allows the federated CDN to serve the content without the content provider changing the URL, or having to manage multiple URLs.

Before adding a federation in the Traffic Ops UI, a user with the federations role needs to be created. This user will be assigned to the federation and will be able to add resolvers to the federation via the Traffic Ops [Federation API](#).

## Header Rewrite Options and DSCP

Most header manipulation and per-delivery service configuration overrides are done using the [ATS Header Rewrite Plugin](#). Traffic Control allows you to enter header rewrite rules to be applied at the edge and at the mid level. The syntax used in Traffic Ops is the same as the one described in the ATS documentation, except for some special strings that will get replaced:

Traffic Ops Entry	Gets Replaced with
<code>__RETURN__</code>	A newline
<code>__CACHE_IPV4__</code>	The cache's IPv4 address

The `deliveryservice` screen also allows you to set the DSCP value of traffic sent to the client. This setting also results in a `header_rewrite` rule to be generated and applied to at the edge.

---

**Note:** The DSCP setting in the UI is *only* for setting traffic towards the client, and gets applied *after* the initial TCP handshake is complete, and the HTTP request is received (before that the cache can't determine what `deliveryservice` this request is for, and what DSCP to apply), so the DSCP feature can not be used for security settings - the TCP SYN-ACK is not going to be DSCP marked.

---

## Token Based Authentication

Token based authentication or *signed URLs* is implemented using the Traffic Server `url_sig` plugin. To sign a URL at the signing portal take the full URL, without any query string, and add on a query string with the following parameters:

**Client IP address** The client IP address that this signature is valid for.

`C=<client IP address>`

**Expiration** The Expiration time (seconds since epoch) of this signature.

`E=<expiration time in secs since unix epoch>`

**Algorithm** The Algorithm used to create the signature. Only 1 (HMAC\_SHA1) and 2 (HMAC\_MD5) are supported at this time

`A=<algorithm number>`

**Key index** Index of the key used. This is the index of the key in the configuration file on the cache. The set of keys is a shared secret between the signing portal and the edge caches. There is one set of keys per reverse proxy domain (fqdn).



K=<key index used>

**Parts** Parts to use for the signature, always excluding the scheme (<http://>). parts0 = fqdn, parts1..x is the directory parts of the path, if there are more parts to the path than letters in the parts param, the last one is repeated for those. Examples:

1: use fqdn and all of URI path 0110: use part1 and part 2 of path only 01: use everything except the fqdn

P=<parts string (0's and 1's)>

**Signature** The signature over the parts + the query string up to and including “S=”.

S=<signature>

**See also:**

The url\_sig [README](#).

## Generate URL Sig Keys

To generate a set of random signed url keys for this delivery service and store them in Traffic Vault, click the **Generate URL Sig Keys** button at the bottom of the delivery service details screen.

## Parent Selection

Parameters in the Edge (child) profile that influence this feature:

Name	File-name	Default	Description
CONFIG proxy.config.http.parent_proxy_routing_enable	records.config	INT 1	enable parent selection. This is a required setting.
CONFIG proxy.config.url_remap.remap_required	records.config	INT 1	required for parent selection.
CONFIG proxy.config.http.no_dns_just_forward_to_parent	records.config	INT 0	See
CONFIG proxy.config.http.uncacheable_requests_bypass_parent	records.config	INT 1	
CONFIG proxy.config.http.parent_proxy_routing_enable	records.config	INT 1	
CONFIG proxy.config.http.parent_proxy.retry_time	records.config	INT 300	
CONFIG proxy.config.http.parent_proxy.fail_threshold	records.config	INT 10	
CONFIG proxy.config.http.parent_proxy.total_connect_attempts	records.config	INT 4	
CONFIG proxy.config.http.parent_proxy.per_parent_connect_attempts	records.config	INT 2	
CONFIG proxy.config.http.parent_proxy.connect_attempts_timeout	records.config	INT 30	
CONFIG proxy.config.http.forward.proxy_auth_to_parent	records.config	INT 0	
CONFIG proxy.config.http.parent_proxy_routing_enable	records.config	INT 0	
CONFIG proxy.config.http.parent_proxy.file	records.config	STRING parent.config	
CONFIG proxy.config.http.parent_proxy.connect_attempts_timeout	records.config	INT 3	
algorithm	parent.config	urlhash	The algorithm to use.

Parameters in the Mid (parent) profile that influence this feature:

Name	Filename	Default	Description
domain_name	CRConfig.json	.	Only parents with the same value as the edge are going to be used as parents (to keep separation between CDNs)
weight	parent.config	1.0	The weight of this parent, translates to the number of replicas in the consistent hash ring. This parameter only has effect with algorithm at the client set to "consistent_hash"
port	parent.config	80	The port this parent is listening on as a forward proxy.
use_ip_address	parent.config	0	1 means use IP(v4) address of this parent in the parent.config, 0 means use the host_name.domain_name concatenation.

## Qstring Handling

Delivery services have a Query String Handling option that, when set to ignore, will automatically add a regex remap to that delivery service's config. There may be times this is not preferred, or there may be requirements for one delivery service or server(s) to behave differently. When this is required, the `psel.qstring_handling` parameter can be set in either the delivery service profile or the server profile, but it is important to note that the server profile will override ALL delivery services assigned to servers with this profile parameter. If the parameter is not set for the server profile but is present for the Delivery Service profile, this will override the setting in the delivery service. A value of "ignore" will not result in the addition of regex remap configuration.

Name	Filename	Default	Description
psel.qstring_handling	parent.config	.	Sets qstring handling without the use of regex remap for a delivery service when assigned to a delivery service profile, and overrides qstring handling for all delivery services for associated servers when assigned to a server profile. Value must be "consider" or "ignore".

## Multi Site Origin

---

**Note:** The configuration of this feature changed significantly between ATS version 5 and  $\geq 6$ . Some configuration in Traffic Control is different as well. This documentation assumes ATS 6 or higher. See [Configure Multi Site Origin](#) for more details.

---

Normally, the mid servers are not aware of any redundancy at the origin layer. With Multi Site Origin enabled this changes - Traffic Server (and Traffic Ops) are now made aware of the fact there are multiple origins, and can be configured to do more advanced failover and loadbalancing actions. A prerequisite for MSO to work is that the multiple origin sites serve identical content with identical paths, and both are configured to serve the same origin hostname as is configured in the deliveryservice *Origin Server Base URL* field. See the [Apache Traffic Server docs](#) for more information on that cache's implementation.

With This feature enabled, origin servers (or origin server VIP names for a site) are going to be entered as servers in to the Traffic Ops UI. Server type is "ORG".

Parameters in the mid profile that influence this feature:

Name	Filename	De- fault	Description
CONFIG proxy.config.http.parent_proxy_routing_enable	records.config	INT 1	enable parent selection. This is a required setting.
CONFIG proxy.config.url_remap.remap_required	records.config	INT 1	required for parent selection.

Parameters in the deliveryservice profile that influence this feature:

Name	Filename	Default	Description
mso.parent_retry	parent.config	-	Either <code>simple_retry</code> , <code>dead_server_retry</code> or both.
mso.algorithm	parent.config	consistent_hash	<p>The algorithm to use. <code>consistent_hash</code>, <code>strict</code>, <code>true</code>, <code>false</code>, or <code>latched</code>.</p> <ul style="list-style-type: none"> <li>• <code>consistent_hash</code> - spreads requests across multiple parents simultaneously based on hash of content URL.</li> <li>• <code>strict</code> - strict Round Robin spreads requests across multiple parents simultaneously based on order of requests.</li> <li>• <code>true</code> - same as <code>strict</code>, but ensures that requests from the same IP always go to the same parent if available.</li> <li>• <code>false</code> - uses only a single parent at any given time and switches to a new parent only if the current parent fails.</li> <li>• <code>latched</code> - same as <code>false</code>, but now, a failed parent will not be retried.</li> </ul>
mso.unavailable_server_retry_response_codes	parent.config	"503"	Quoted, comma separated list of HTTP status codes that count as a <code>unavailable_server_retry_response_code</code> .
mso.max_unavailable_server_retries	parent.config	1	How many times an unavailable server will be retried.
mso.simple_retry_response_codes	parent.config	"404"	Quoted, comma separated list of HTTP status codes that count as a simple retry response code.
mso.max_simple_retries	parent.config	1	How many times a simple retry will be done.

see *Configure Multi Site Origin* for a *quick how to* on this feature.

## Traffic Router Profile

Name	Config_file	Description
location	dns.zone	Location to store the DNS zone files in the local file system of
location	http-log4j.properties	Location to find the log4j.properties file for Traffic Router.
location	dns-log4j.properties	Location to find the dns-log4j.properties file for Traffic Router.
location	geolocation.properties	Location to find the log4j.properties file for Traffic Router.
CDN_name	rascal-config.txt	The human readable name of the CDN for this profile.
CoverageZoneJsonURL	CRConfig.xml	The location (URL) to retrieve the coverage zone map file in JS
geolocation.polling.url	CRConfig.json	The location (URL) to retrieve the geo database file from.
geolocation.polling.interval	CRConfig.json	How often to refresh the coverage geo location database in ms
coveragezone.polling.interval	CRConfig.json	How often to refresh the coverage zone map in ms
coveragezone.polling.url	CRConfig.json	The location (URL) to retrieve the coverage zone map file in JS
deepcoveragezone.polling.interval	CRConfig.json	How often to refresh the deep coverage zone map in ms
deepcoveragezone.polling.url	CRConfig.json	The location (URL) to retrieve the deep coverage zone map file
tld.soa.expire	CRConfig.json	The value for the expire field the Traffic Router DNS Server wi
tld.soa.minimum	CRConfig.json	The value for the minimum field the Traffic Router DNS Serve
tld.soa.admin	CRConfig.json	The DNS Start of Authority admin. Should be a valid support e
tld.soa.retry	CRConfig.json	The value for the retry field the Traffic Router DNS Server will
tld.soa.refresh	CRConfig.json	The TTL the Traffic Router DNS Server will respond with on A
tld.ttls.NS	CRConfig.json	The TTL the Traffic Router DNS Server will respond with on N
tld.ttls.SOA	CRConfig.json	The TTL the Traffic Router DNS Server will respond with on S
tld.ttls.AAAA	CRConfig.json	The Time To Live (TTL) the Traffic Router DNS Server will re
tld.ttls.A	CRConfig.json	The TTL the Traffic Router DNS Server will respond with on A
tld.ttls.DNSKEY	CRConfig.json	The TTL the Traffic Router DNS Server will respond with on I
tld.ttls.DS	CRConfig.json	The TTL the Traffic Router DNS Server will respond with on I
api.port	server.xml	The TCP port Traffic Router listens on for API (REST) access.
api.cache-control.max-age	CRConfig.json	The value of the <code>Cache-Control: max-age=</code> header in
api.auth.url	CRConfig.json	The API authentication URL ( <a href="https://\${tmHostname}/api/1.1/u">https://\${tmHostname}/api/1.1/u</a> )
consistent.dns.routing	CRConfig.json	Control whether DNS Delivery Services use consistent hashing
dnssec.enabled	CRConfig.json	Whether DNSSEC is enabled; this parameter is updated via the
dnssec.allow.expired.keys	CRConfig.json	Allow Traffic Router to use expired DNSSEC keys to sign zone
dynamic.cache.primer.enabled	CRConfig.json	Allow Traffic Router to attempt to prime the dynamic zone cac
dynamic.cache.primer.limit	CRConfig.json	Limit the number of permutations to prime when dynamic zone
keystore.maintenance.interval	CRConfig.json	The interval in seconds which Traffic Router will check the key
keystore.api.url	CRConfig.json	The keystore API URL ( <a href="https://\${tmHostname}/api/1.1/cdns/n">https://\${tmHostname}/api/1.1/cdns/n</a> )
keystore.fetch.timeout	CRConfig.json	The timeout in milliseconds for requests to the keystore API
keystore.fetch.retries	CRConfig.json	The number of times Traffic Router will attempt to load keys b
keystore.fetch.wait	CRConfig.json	The number of milliseconds Traffic Router will wait before a r
signaturemanager.expiration.multiplier	CRConfig.json	Multiplier used in conjunction with a zone's maximum TTL to
zonemanager.threadpool.scale	CRConfig.json	Multiplier used to determine the number of cores to use for zon
zonemanager.cache.maintenance.interval	CRConfig.json	The interval in seconds which Traffic Router will check for zon
zonemanager.dynamic.response.expiration	CRConfig.json	A string (e.g.: 300s) that defines how long a dynamic zone
DNSKEY.generation.multiplier	CRConfig.json	Used to determine when new keys need to be regenerated. Ke
DNSKEY.effective.multiplier	CRConfig.json	Used when creating an effective date for a new key set. New ke

## Regex Remap Expression

The regex remap expression allows to use a regex and resulting match group(s) in order to modify the request URIs that are sent to origin. For example:

```
^/original/(.*) http://origin.example.com/remapped/$1
```

**Note:** If **Query String Handling** is set to 2 Drop at edge, then you will not be allowed to save a regex remap expression, as dropping query strings actually relies on a regex remap of its own. However, if there is a need to both drop query strings **and** remap request URIs, this can be accomplished by setting **Query String Handling** to 1 Do not use in cache key, but pass up to origin, and then using a custom regex remap expression to do the necessary remapping, while simultaneously dropping query strings. The following example will capture the original request URI up to, but not including, the query string and then forward to a remapped URI:

```
^/([^\?]*).* http://origin.example.com/remapped/$1
```

## Delivery Service Regexp

This table defines how requests are matched to the delivery service. There are 3 type of entries possible here:

Name	Description	DS Type	Status
HOST_REGEXP	This is the regular expresion to match the host part of the URL.	DNS and HTTP	Supported
PATH_REGEXP	This is the regular expresion to match the path part of the URL.	HTTP	Beta
HEADER_REGEXP	This is the regular expresion to match on any header in the request.	HTTP	Beta

The **Order** entry defines the order in which the regular expressions get evaluated. To support CNAMEs from domains outside of the Traffic Control top level DNS domain, enter multiple HOST\_REGEXP lines.

**Example:** Example foo.

**Note:** In most cases is is sufficient to have just one entry in this table that has a HOST\_REGEXP Type, and Order 0. For the *movies* delivery service in the Kabletown CDN, the entry is simply single HOST\_REGEXP set to `.*\.movies\..*`. This will match every url that has a hostname that ends with `movies.cdn1.kabletown.net`, since `cdn1.kabletown.net` is the Kabletown CDN's DNS domain.

## Static DNS Entries

Static DNS entries allow you to create other names *under* the delivery service domain. You can enter any valid hostname, and create a CNAME, A or AAAA record for it by clicking the **Static DNS** button at the bottom of the delivery service details screen.

## Server Assignments

Click the **Server Assignments** button at the bottom of the screen to assign servers to this delivery service. Servers can be selected by drilling down in a tree, starting at the profile, then the cache group, and then the individual servers. Traffic Router will only route traffic for this delivery service to servers that are assigned to it.

## The Coverage Zone File and ASN Table

The Coverage Zone File (CZF) should contain a cachegroup name to network prefix mapping in the form:

```
{
  "coverageZones": {
    "cache-group-01": {
      "coordinates": {
        "latitude": 1.1,
        "longitude": 2.2
      },
      "network6": [
        "1234:5678::/64",
        "1234:5679::/64"
      ],
      "network": [
        "192.168.8.0/24",
        "192.168.9.0/24"
      ]
    },
    "cache-group-02": {
      "coordinates": {
        "latitude": 3.3,
        "longitude": 4.4
      },
      "network6": [
        "1234:567a::/64",
        "1234:567b::/64"
      ],
      "network": [
        "192.168.4.0/24",
        "192.168.5.0/24"
      ]
    }
  }
}
```

The CZF is an input to the Traffic Control CDN, and as such does not get generated by Traffic Ops, but rather, it gets consumed by Traffic Router. Some popular IP management systems output a very similar file to the CZF but in stead of a cachegroup an ASN will be listed. Traffic Ops has the “Networks (ASNs)” view to aid with the conversion of files like that to a Traffic Control CZF file; this table is not used anywhere in Traffic Ops, but can be used to script the conversion using the API.

The script that generates the CZF file is not part of Traffic Control, since it is different for each situation.

---

**Note:** The "coordinates" section is optional and may be used by Traffic Router for localization in the case of a CZF “hit” where the zone name does not map to a Cache Group name in Traffic Ops (i.e. Traffic Router will route to the closest Cache Group(s) geographically).

---



## The Deep Coverage Zone File

The Deep Coverage Zone File (DCZF) format is similar to the CZF format but adds a `caches` list under each `deepCoverageZone`:

```
{
  "deepCoverageZones": {
    "location-01": {
      "coordinates": {
        "latitude": 5.5,
        "longitude": 6.6
      },
      "network6": [
        "1234:5678::/64",
        "1234:5679::/64"
      ],
      "network": [
        "192.168.8.0/24",
        "192.168.9.0/24"
      ],
      "caches": [
        "edge-01",
        "edge-02"
      ]
    },
    "location-02": {
      "coordinates": {
        "latitude": 7.7,
        "longitude": 8.8
      },
      "network6": [
        "1234:567a::/64",
        "1234:567b::/64"
      ],
      "network": [
        "192.168.4.0/24",
        "192.168.5.0/24"
      ],
      "caches": [
        "edge-02",
        "edge-03"
      ]
    }
  }
}
```

Each entry in the `caches` list is the hostname of an edge cache registered in Traffic Ops which will be used for “deep” caching in that Deep Coverage Zone. Unlike a regular CZF, coverage zones in the DCZF do not map to a Cache Group in Traffic Ops, so currently the deep coverage zone name only needs to be unique.

If the Traffic Router gets a DCZF “hit” for a requested Delivery Service that has Deep Caching enabled, the client will be routed to an available “deep” cache from that zone’s `caches` list.

---

**Note:** The `"coordinates"` section is optional.

---

## Parameters and Profiles

Parameters are shared between profiles if the set of { `name`, `config_file`, `value` } is the same. To change a value in one profile but not in others, the parameter has to be removed from the profile you want to change it in, and a new parameter entry has to be created (**Add Parameter** button at the bottom of the Parameters view), and assigned to that profile. It is easy to create new profiles from the **Misc > Profiles** view - just use the **Add/Copy Profile** button at the bottom of the profile view to copy an existing profile to a new one. Profiles can be exported from one system and imported to another using the profile view as well. It makes no sense for a parameter to not be assigned to a single profile - in that case it really has no function. To find parameters like that use the **Parameters > Orphaned Parameters** view. It is easy to create orphaned parameters by removing all profiles, or not assigning a profile directly after creating the parameter.

### See also:

*Profile Parameters* in the *Configuring Traffic Ops* section.

## Tools

### Generate ISO

Generate ISO is a tool for building custom ISOs for building caches on remote hosts. Currently it only supports Centos 6, but if you're brave and pure of heart you MIGHT be able to get it to work with other unix-like OS's.

The interface is *mostly* self explanatory as it's got hints.

Field	Explanation
Choose a server from list:	This option gets all the server names currently in the Traffic Ops database and will autofill known values.
OS Version:	There needs to be an <code>_osversions.cfg_</code> file in the ISO directory that maps the name of a directory to a name that shows up here.
Hostname:	This is the FQDN of the server to be installed. It is required.
Root password:	If you don't put anything here it will default to the salted MD5 of "Fred". Whatever put is MD5 hashed and writte to disk.
DHCP:	if yes, other IP settings will be ignored
IP Address:	Required if DHCP=no
Netmask:	Required if DHCP=no
Gateway:	Required if DHCP=no
IPV6 Address:	Optional. /64 is assumed if prefix is omitted
IPV6 Gateway:	Ignored if an IPV4 gateway is specified
Network Device:	Optional. Typical values are <code>bond0</code> , <code>eth4</code> , etc. Note: if you enter <code>bond0</code> , a LACP bonding config will be written
MTU:	If unsure, set to 1500
Specify disk for OS install:	Optional. Typical values are "sda".

When you click the **Download ISO** button the folling occurs (all paths relative to the top level of the directory specified in `_osversions.cfg_`):

1. Reads `/etc/resolv.conf` to get a list of nameservers. This is a rather ugly hack that is in place until we get a way of configuring it in the interface.
2. Writes a file in the `ks_scripts/state.out` that contains directory from `_osversions.cfg_` and the `mkisofs` string that we'll call later.
3. Writes a file in the `ks_scripts/network.cfg` that is a bunch of `key=value` pairs that set up networking.

4. Creates an MD5 hash of the password you specify and writes it to `ks_scripts/password.cfg`. Note that if you do not specify a password “Fred” is used. Also note that we have experienced some issues with webbrowsers autofilling that field.
5. Writes out a disk configuration file to `ks_scripts/disk.cfg`.
6. `mkisofs` is called against the directory configured in `_osversions.cfg` and an ISO is generated in memory and delivered to your webbrowser.

You now have a customized ISO that can be used to install Red Hat and derivative Linux installations with some modifications to your `ks.cfg` file.

Kickstart/Anaconda will mount the ISO at `/mnt/stage2` during the install process (at least with 6).

You can directly include the password file anywhere in your `ks.cfg` file (usually in the top) by doing `%include /mnt/stage2/ks_scripts/password.cfg`

What we currently do is have 2 scripts, one to do hard drive configuration and one to do network configuration. Both are relatively specific to the environment they were created in, and both are *probably* wrong for other organizations, however they are currently living in the “misc” directory as examples of how to do things.

We trigger those in a `%pre` section in `ks.cfg` and they will write config files to `/tmp`. We will then include those files in the appropriate places using `%pre`.

For example this is a section of our `ks.cfg` file:

```
%include /mnt/stage2/ks_scripts/packages.txt



```
python /mnt/stage2/ks_scripts/create_network_line.py
bash /mnt/stage2/ks_scripts/drive_config.sh

```


```

These two scripts will then run `_before_` anaconda sets up it’s internal structures, then a bit further up in the `ks.cfg` file (outside of the `%pre %end` block) we do an

```
%include /mnt/stage2/ks_scripts/password.cfg
...
%include /tmp/network_line

%include /tmp/drive_config
...
```

This snarfs up the contents and inlines them.

If you only have one kind of hardware on your CDN it is probably best to just put the drive config right in the `ks.cfg`.

If you have simple networking needs (we use bonded interfaces in most, but not all locations and we have several types of hardware meaning different ethernet interface names at the OS level etc.) then something like this:

```
#!/bin/bash
source /mnt/stage2/ks_scripts/network.cfg
echo "network --bootproto=static --activate --ipv6=$IPV6ADDR --ip=$IPADDR --netmask=
↪$NETMASK --gateway=$GATEWAY --ipv6gateway=$GATEWAY --nameserver=$NAMESERVER --mtu=
↪$MTU --hostname=$HOSTNAME" >> /tmp/network.cfg
# Note that this is an example and may not work at all.
```

You could also put this in the `%pre` section. Lots of ways to solve it.

We have included the two scripts we use in the “misc” directory of the git repo:

- `kickstart_create_network_line.py`

- `kickstart_drive_config.sh`

These scripts were written to support a very narrow set of expectations and environment and are almost certainly not suitable to just drop in, but they might provide a good starting point.

## Queue Updates and Snapshot CRConfig

When changing delivery services special care has to be taken so that Traffic Router will not send traffic to caches for delivery services that the cache doesn't know about yet. In general, when adding delivery services, or adding servers to a delivery service, it is best to update the caches before updating Traffic Router and Traffic Monitor. When deleting delivery services, or deleting server assignments to delivery services, it is best to update Traffic Router and Traffic Monitor first and then the caches. Updating the cache configuration is done through the *Queue Updates* menu, and updating Traffic Monitor and Traffic Router config is done through the *Snapshot CRConfig* menu.



### Queue Updates

Every 15 minutes the caches should run a *syncds* to get all changes needed from Traffic Ops. The files that will be updated by the syncds job are:

- `records.config`
- `remap.config`
- `parent.config`
- `cache.config`
- `hosting.config`
- `url_sig_(.*).config`
- `hdr_rw_(.*).config`
- `regex_revalidate.config`
- `ip_allow.config`

A cache will only get updated when the update flag is set for it. To set the update flag, use the *Queue Updates* menu - here you can schedule updates for a whole CDN or a cache group:

1. Click **Tools > Queue Updates**.
2. Select the CDN to queue updates for, or All.
3. Select the cache group to queue updates for, or All
4. Click the **Queue Updates** button.
5. When the Queue Updates for this Server? (all) window opens, click **OK**.

To schedule updates for just one cache, use the "Server Checks" page, and click the  in the *UPD* column. The *UPD* column of Server Checks page will change show a  when updates are pending for that cache.

### Snapshot CRConfig

Every 60 seconds Traffic Monitor will check with Traffic Ops to see if a new CRConfig snapshot exists; Traffic Monitor polls Traffic Ops for a new CRConfig, and Traffic Router polls Traffic Monitor for the same file. This is necessary to ensure that Traffic Monitor sees configuration changes first, which helps to ensure that the health and state of caches

and delivery services propagates properly to Traffic Router. See [Traffic Router Profile](#) for more information on the CRConfig file.

To create a new snapshot, use the *Tools > Snapshot CRConfig* menu:

1. Click **Tools > Snapshot CRConfig**.
2. Verify the selection of the correct CDN from the Choose CDN drop down and click **Diff CRConfig**. On initial selection of this, the CRConfig Diff window says the following:  
  
There is no existing CRConfig for [cdn] to diff against... Is this the first snapshot??? If you are not sure why you are getting this message, please do not proceed! To proceed writing the snapshot anyway click the 'Write CRConfig' button below.  
  
If there is an older version of the CRConfig, a window will pop up showing the differences between the active CRConfig and the CRConfig about to be written.
3. Click **Write CRConfig**.
4. When the This will push out a new CRConfig.json. Are you sure? window opens, click **OK**.
5. The Successfully wrote CRConfig.json! window opens, click **OK**.

## Invalidate Content

Invalidating content on the CDN is sometimes necessary when the origin was mis-configured and something is cached in the CDN that needs to be removed. Given the size of a typical Traffic Control CDN and the amount of content that can be cached in it, removing the content from all the caches may take a long time. To speed up content invalidation, Traffic Ops will not try to remove the content from the caches, but it makes the content inaccessible using the *regex\_revalidate* ATS plugin. This forces a *revalidation* of the content, rather than a new get.

---

**Note:** This method forces a HTTP *revalidation* of the content, and not a new *GET* - the origin needs to support revalidation according to the HTTP/1.1 specification, and send a 200 OK or 304 Not Modified as applicable.

---

To invalidate content:

1. Click **Tools > Invalidate Content**
2. Fill out the form fields:
  - Select the **Delivery Service**
  - Enter the **Path Regex** - this should be a [PCRE](#) compatible regular expression for the path to match for forcing the revalidation. Be careful to only match on the content you need to remove - revalidation is an expensive operation for many origins, and a simple `/.*` can cause an overload condition of the origin.
  - Enter the **Time To Live** - this is how long the revalidation rule will be active for. It usually makes sense to make this the same as the `Cache-Control` header from the origin which sets the object time to live in cache (by `max-age` or `Expires`). Entering a longer TTL here will make the caches do unnecessary work.
  - Enter the **Start Time** - this is the start time when the revalidation rule will be made active. It is pre-populated with the current time, leave as is to schedule ASAP.
3. Click the **Submit** button.

## Manage DNSSEC Keys

In order to support [DNSSEC](#) in Traffic Router, Traffic Ops provides some actions for managing DNSSEC keys for a CDN and associated Delivery Services. DNSSEC Keys consist of a Key Signing Keys (KSK) which are used to sign other DNSKEY records as well as Zone Signing Keys (ZSK) which are used to sign other records. DNSSEC Keys are stored in [Traffic Vault](#) and should only be accessible to Traffic Ops. Other applications needing access to this data, such as Traffic Router, must use the Traffic Ops [DNSSEC APIs](#) to retrieve this information.

### To Manage DNSSEC Keys:

1. Click **Tools -> Manage DNSSEC Keys**
2. Choose a CDN and click **Manage DNSSEC Keys**
  - If keys have not yet been generated for a CDN, this screen will be mostly blank with just the **CDN** and **DNSSEC Active?** fields being populated.
  - If keys have been generated for the CDN, the Manage DNSSEC Keys screen will show the TTL and Top Level Domain (TLD) KSK Expiration for the CDN as well as DS Record information which will need to be added to the parent zone of the TLD in order for DNSSEC to work.

The Manage DNSSEC Keys screen also allows a user to perform the following actions:

### Activate/Deactivate DNSSEC for a CDN

Fairly straight forward, this button set the `dnssec.enabled` param to either **true** or **false** on the Traffic Router profile for the CDN. The Activate/Deactivate option is only available if DNSSEC keys exist for CDN. In order to active DNSSEC for a CDN a user must first generate keys and then click the **Active DNSSEC** button.

### Generate Keys

Generate Keys will generate DNSSEC keys for the CDN TLD as well as for each Delivery Service in the CDN. It is important to note that this button will create a new KSK for the TLD and, therefore, a new DS Record. Any time a new DS Record is created, it will need to be added to the parent zone of the TLD in order for DNSSEC to work properly. When a user clicks the **Generate Keys** button, they will be presented with a screen with the following fields:

- **CDN:** This is not editable and displays the CDN for which keys will be generated
- **ZSK Expiration (Days):** Sets how long (in days) the Zone Signing Key will be valid for the CDN and associated Delivery Services. The default is 30 days.
- **KSK Expiration (Days):** Sets how long (in days) the Key Signing Key will be valid for the CDN and associated Delivery Services. The default is 365 days.
- **Effective Date (GMT):** The time from which the new keys will be active. Traffic Router will use this value to determine when to start signing with the new keys and stop signing with the old keys.

Once these fields have been correctly entered, a user can click Generate Keys. The user will be presented with a confirmation screen to help them understand the impact of generating the keys. If a user confirms, the keys will be generated and stored in Traffic Vault.

### Regenerate KSK

Regenerate KSK will create a new Key Signing Key for the CDN TLD. A new DS Record will also be generated and need to be put into the parent zone in order for DNSSEC to work correctly. The **Regenerate KSK** button is only available if keys have already been generated for a CDN. The intent of the button is to provide a mechanism for generating a new KSK when a previous one expires or if necessary for other reasons such as a security breach. When a user goes to generate a new KSK they are presented with a screen with the following options:

- **CDN:** This is not editable and displays the CDN for which keys will be generated
- **KSK Expiration (Days):** Sets how long (in days) the Key Signing Key will be valid for the CDN and associated Delivery Services. The default is 365 days.

- **Effective Date (GMT):** The time from which the new KSK and DS Record will be active. Since generating a new KSK will generate a new DS Record that needs to be added to the parent zone, it is very important to make sure that an effective date is chosen that allows for time to get the DS Record into the parent zone. Failure to get the new DS Record into the parent zone in time could result in DNSSEC errors when Traffic Router tries to sign responses.

Once these fields have been correctly entered, a user can click Generate KSK. The user will be presented with a confirmation screen to help them understand the impact of generating the KSK. If a user confirms, the KSK will be generated and stored in Traffic Vault.

Additionally, Traffic Ops also performs some systematic management of DNSSEC keys. This management is necessary to help keep keys in sync for Delivery Services in a CDN as well as to make sure keys do not expire without human intervention.

### Generation of keys for new Delivery Services

If a new Delivery Service is created and added to a CDN that has DNSSEC enabled, Traffic Ops will create DNSSEC keys for the Delivery Service and store them in Traffic Vault.

### Regeneration of expiring keys for a Delivery Service

Traffic Ops has a process, controlled by cron, to check for expired or expiring keys and re-generate them. The process runs at 5 minute intervals to check and see if keys are expired or close to expiring (withing 10 minutes by default). If keys are expired for a Delivery Service, traffic ops will regenerate new keys and store them in Traffic Vault. This process is the same for the CDN TLD ZSK, however Traffic Ops will not re-generate the CDN TLD KSK systematically. The reason is that when a KSK is regenerated for the CDN TLD then a new DS Record will also be created. The new DS Record needs to be added to the parent zone before Traffic Router attempts to sign with the new KSK in order for DNSSEC to work correctly. Therefore, management of the KSK needs to be a manual process.

## 3.1.7 Managing Traffic Ops Extensions

Each script is a separate bash script located in `$TO_HOME/bin/checks/`.

The extensions must be registered with Traffic Ops in order to display a column on the Server Check page. The list of currently registered extensions can be listed by running `/opt/traffic_ops/app/bin/extensions -a`.

The below extensions are automatically registered with the Traffic Ops database (`to_extension` table) at install time (see `traffic_ops/app/db/seeds.sql`). However, cron must still be configured to run these checks periodically.

The scripts are called as follows:

```
$TO_HOME/bin/checks/To<name>Check.pl -c "{\"base_url\": \"https://\",<traffic_ops_ip>
↪\", \"check_name\": \"<check_name>\"}" -l <log level>
where:

<name> is the type of check script
<traffic_ops_ip> is the IP address of the Traffic Ops Server
<check_name> is the name of the check. For example: CDU, CHR, DSCP, MTU, etc...
<log_level> is between 1 and 4, with 4 being the most verbose. This field is optional
```

### Example Cron File

Edit with `crontab -e`. You may need to adjust the path to your `$TO_HOME` to match your system.

```

PERL5LIB=/opt/traffic_ops/app/local/lib/perl5:/opt/traffic_ops/app/lib

# IPv4 ping examples - The 'select: ["hostName","domainName"]' works but, if you want
↳to check DNS resolution use FQDN.
*/15 * * * * root /opt/traffic_ops/app/bin/checks/ToPingCheck.pl -c "{\"base_url\": \
↳\"https://localhost\", \"check_name\": \"10G\", \"select\": [\"hostName\", \
↳\"domainName\"]}" >> /var/log/traffic_ops/extensionCheck.log 2>&1
*/15 * * * * root /opt/traffic_ops/app/bin/checks/ToPingCheck.pl -c "{\"base_url\": \
↳\"https://localhost\", \"check_name\": \"10G\", \"select\": \"ipAddress\"}" >> /var/
↳log/traffic_ops/extensionCheck.log 2>&1
*/15 * * * * root /opt/traffic_ops/app/bin/checks/ToPingCheck.pl -c "{\"base_url\": \
↳\"https://localhost\", \"check_name\": \"10G\", \"name\": \"IPv4 Ping\", \"select\": \
↳\"ipAddress\", \"syslog_facility\": \"local0\"}" > /dev/null 2>&1

# IPv6 ping examples
*/15 * * * * root /opt/traffic_ops/app/bin/checks/ToPingCheck.pl -c "{\"base_url\": \
↳\"https://localhost\", \"check_name\": \"10G6\", \"name\": \"IPv6 Ping\", \"select\
↳\": \"ip6Address\", \"syslog_facility\": \"local0\"}" >/dev/null 2>&1
*/15 * * * * root /opt/traffic_ops/app/bin/checks/ToPingCheck.pl -c "{\"base_url\": \
↳\"https://localhost\", \"check_name\": \"10G6\", \"select\": \"ip6Address\"}" >> /
↳var/log/traffic_ops/extensionCheck.log 2>&1

# iLO ping
18 * * * * root /opt/traffic_ops/app/bin/checks/ToPingCheck.pl -c "{\"base_url\": \
↳\"https://localhost\", \"check_name\": \"ILO\", \"select\": \"iloIpAddress\"}" >> /
↳var/log/traffic_ops/extensionCheck.log 2>&1
18 * * * * root /opt/traffic_ops/app/bin/checks/ToPingCheck.pl -c "{\"base_url\": \
↳\"https://localhost\", \"check_name\": \"ILO\", \"name\": \"ILO ping\", \"select\": \
↳\"iloIpAddress\", \"syslog_facility\": \"local0\"}" >/dev/null 2>&1

# MTU ping
45 0 * * * * root /opt/traffic_ops/app/bin/checks/ToPingCheck.pl -c "{\"base_url\": \
↳\"https://localhost\", \"check_name\": \"MTU\", \"select\": \"ipAddress\"}" >> /var/
↳log/traffic_ops/extensionCheck.log 2>&1
45 0 * * * * root /opt/traffic_ops/app/bin/checks/ToPingCheck.pl -c "{\"base_url\": \
↳\"https://localhost\", \"check_name\": \"MTU\", \"select\": \"ip6Address\"}" >> /var/
↳log/traffic_ops/extensionCheck.log 2>&1
45 0 * * * * root /opt/traffic_ops/app/bin/checks/ToPingCheck.pl -c "{\"base_url\": \
↳\"https://localhost\", \"check_name\": \"MTU\", \"name\": \"Max Trans Unit\", \
↳\"select\": \"ipAddress\", \"syslog_facility\": \"local0\"}" > /dev/null 2>&1
45 0 * * * * root /opt/traffic_ops/app/bin/checks/ToPingCheck.pl -c "{\"base_url\": \
↳\"https://localhost\", \"check_name\": \"MTU\", \"name\": \"Max Trans Unit\", \
↳\"select\": \"ip6Address\", \"syslog_facility\": \"local0\"}" > /dev/null 2>&1

# FQDN
27 * * * * root /opt/traffic_ops/app/bin/checks/ToFQDNCheck.pl -c "{\"base_url\": \
↳\"https://localhost\", \"check_name\": \"FQDN\"}" >> /var/log/traffic_ops/
↳extensionCheck.log 2>&1
27 * * * * root /opt/traffic_ops/app/bin/checks/ToFQDNCheck.pl -c "{\"base_url\": \
↳\"https://localhost\", \"check_name\": \"FQDN\", \"name\": \"DNS Lookup\", \"syslog_
↳facility\": \"local0\"}" > /dev/null 2>&1

# DSCP
36 * * * * root /opt/traffic_ops/app/bin/checks/ToDSCPCheck.pl -c "{\"base_url\": \
↳\"https://localhost\", \"check_name\": \"DSCP\", \"cms_interface\": \"eth0\"}" >> /
↳var/log/traffic_ops/extensionCheck.log 2>&1
36 * * * * root /opt/traffic_ops/app/bin/checks/ToDSCPCheck.pl -c "{\"base_url\": \
↳\"https://localhost\", \"check_name\": \"DSCP\", \"name\": \"Delivery Service\", \
↳\"cms_interface\": \"eth0\", \"syslog_facility\": \"local0\"}" > /dev/null 2>&1

```

(continues on next page)



(continued from previous page)

```
# RTR
10 * * * * root /opt/traffic_ops/app/bin/checks/ToRTRCheck.pl -c "{\"base_url\": \
↪ "https://localhost\", \"check_name\": \"RTR\"}" >> /var/log/traffic_ops/
↪ extensionCheck.log 2>&1
10 * * * * root /opt/traffic_ops/app/bin/checks/ToRTRCheck.pl -c "{\"base_url\": \
↪ "https://localhost\", \"check_name\": \"RTR\", \"name\": \"Content Router Check\", \
↪ \"syslog_facility\": \"local0\"}" > /dev/null 2>&1

# CHR
*/15 * * * * root /opt/traffic_ops/app/bin/checks/ToCHRCheck.pl -c "{\"base_url\": \
↪ "https://localhost\", \"check_name\": \"CHR\"}" >> /var/log/traffic_ops/
↪ extensionCheck.log 2>&1

# CDU
20 * * * * root /opt/traffic_ops/app/bin/checks/ToCDUCheck.pl -c "{\"base_url\": \
↪ "https://localhost\", \"check_name\": \"CDU\"}" >> /var/log/traffic_ops/
↪ extensionCheck.log 2>&1

# ORT
40 * * * * ssh_key_edge_user /opt/traffic_ops/app/bin/checks/ToORTCheck.pl -c "{\
↪ "base_url\": \"https://localhost\", \"check_name\": \"ORT\"}" >> /var/log/traffic_
↪ ops/extensionCheck.log 2>&1
40 * * * * ssh_key_edge_user /opt/traffic_ops/app/bin/checks/ToORTCheck.pl -c "{\
↪ "base_url\": \"https://localhost\", \"check_name\": \"ORT\", \"name\": \
↪ \"Operational Readiness Test\", \"syslog_facility\": \"local0\"}" > /dev/null 2>&1
```

### 3.1.8 Traffic Portal Administration

The following are requirements to ensure an accurate set up:

- CentOS 6.7 or 7
- Node.js 6.0.x or above

#### Installing Traffic Portal

- Download the Traffic Portal RPM from [Apache Jenkins](#) or build from source (./pkg traffic\_portal\_build).
- Copy the Traffic Portal RPM to your server
- curl -silent -location [https://rpm.nodesource.com/setup\\_6.x](https://rpm.nodesource.com/setup_6.x) | sudo bash -
- sudo yum install -y nodejs
- sudo yum install -y <traffic\_portal rpm>

#### Configuring Traffic Portal

- update /etc/traffic\_portal/conf/config.js (if upgrade, reconcile config.js with config.js.rpmnew and then delete config.js.rpmnew)
- update /opt/traffic\_portal/public/traffic\_portal\_properties.json (if upgrade, reconcile traffic\_portal\_properties.json with traffic\_portal\_properties.json.rpmnew and then delete traffic\_portal\_properties.json.rpmnew)
- [OPTIONAL] update /opt/traffic\_portal/public/resources/assets/css/custom.css (to customize traffic portal skin)

#### Starting Traffic Portal

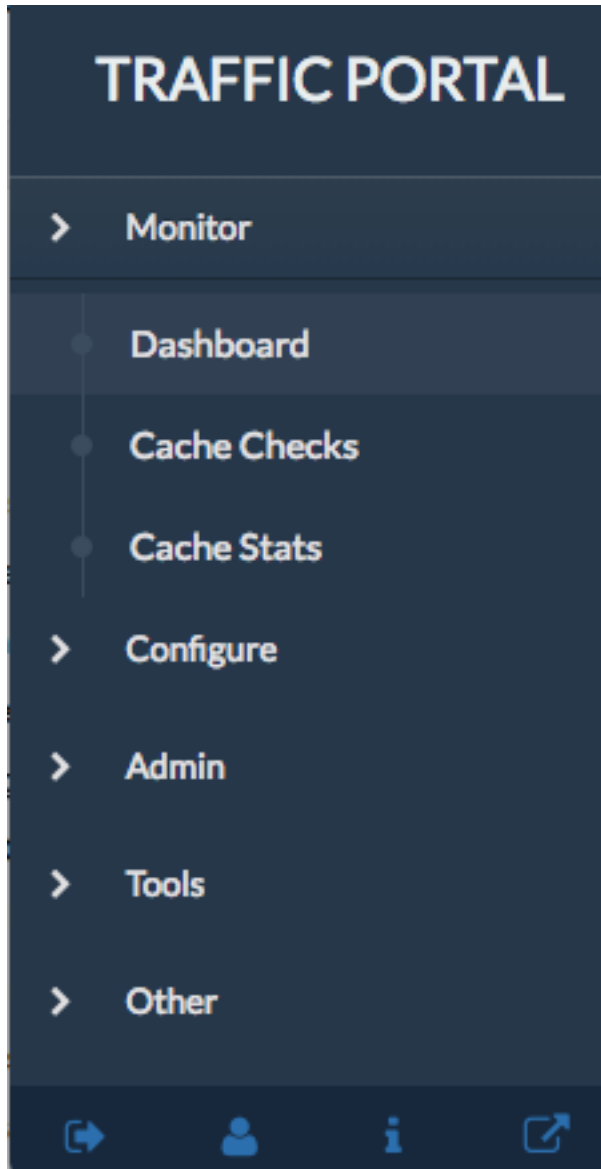
- `sudo service traffic_portal start`

### Stopping Traffic Portal

- `sudo service traffic_portal stop`

## 3.1.9 Traffic Portal - Using

### Traffic portal Menu



Traffic Portal is a replacement for the Traffic Ops UI. The following tabs are available in the Traffic Portal Menu.

- **Monitor**

Information on the health of the system. Click the Monitor tab to get to the following options:

Option	Description
Dash-board	A real time view into the main performance indicators of the CDNs managed by Traffic Control. This view is sourced directly by the Traffic Monitor data and is updated every 10 seconds. This is the default screen of Traffic Portal. See <a href="#">Dashboard</a> for details.
Cache Checks	A real time view into the status of each cache. This view is sourced by the Traffic Monitor data and is updated on demand. See <a href="#">Cache Checks</a> for details.
Cache Stats	A table showing the results of the periodic check extension scripts that are run. See <a href="#">Cache Stats</a>

- **Configure**

The main Delivery Service table. This is where you Create/Read/Update/Delete Delivery Services of all types. Hover over to get the following sub option:

Option	Description
Cache Groups	Add/Edit/Delete Cache Groups.
Delivery Services	Add/Edit/Delete Delivery Services.
Servers	Add/Edit/Delete Servers. Queue/Clear Server updates.

- **Admin**

The main Servers table. This is where you Create/Read/Update/Delete servers of all types. Click the main tab to get to the main table, and hover over to get these sub options:

Option	Description
ASNs	Create/Read/Update/Delete Autonomous System Numbers See <a href="#">The Coverage Zone File and ASN Table</a>
CDNs	Create/Read/Update/Delete CDNs, Manage Federations( <a href="#">Federations</a> ), Manage DNSSEC Keys( <a href="#">Manage DNSSEC Keys</a> ), View Delivery Services, Profiles, and Servers
Divisions	Create/Read/Update/Delete divisions
Jobs	View/Create Invalidate Content Jobs
Phys Locations	Create/Read/Update/Delete locations
Parameters	Create/Read/Update/Delete Parameters
Profiles	Create/Read/Update/Delete profiles. See <a href="#">Parameters and Profiles</a>
Regions	Create/Read/Update/Delete regions
Statuses	Create/Read/Update/Delete server statuses
Tenants	Create/Read/Update/Delete CDN tenants
Types	Create/Read/Update/Delete data types
Users	Create/Read/Update/Delete users

- **Tools**

Tools for working with Traffic Ops and it's servers. Hover over this tab to get the following options:

Option	Description
Generate ISO	Generate a bootable image for any of the servers in the Servers table (or any server for that matter). See <a href="#">Generate ISO</a>

- **Other**

Miscellaneous editing options. Hover over this tab to get the following options:

Option	Description
Grafana	Link to your Grafana instance
Docs	Link to <a href="https://trafficcontrol.apache.org">trafficcontrol.apache.org</a>

- **ChangeLog**

Displayed at the top right of every page. The Changelog table displays the changes that are being made to the Traffic Ops database through the Traffic Portal. This tab will show the number of changes since you last visited this tab in (brackets) since the last time you visited this tab. There are currently no sub menus for this tab.

- **Bottom Menu Icons**



Hover over each icon to see the following options:

Option	Description
Logout	Logout of Traffic Portal
User Profile	Read/Update your user information
Release Info	Release notes for the most recent releases of Traffic Portal
Popout	Open the current window in a new tab. 0

## Monitor

### Dashboard

The Dashboard is the default landing screen for Traffic Portal, it displays various statistics about the overall health of your CDN.

- **Current Bandwidth:** The current bandwidth of all of your CDNs.
- **Current Connections:** The current number of connections to all of your CDNs.
- **Healthy Caches:** Number of healthy caches returned by the `/cdns/health` endpoint. Click the link to view the healthy caches on the cache stats page.
- **Unhealthy Caches:** Number of unhealthy caches returned by the `/cdns/health` endpoint. Click the link to view the unhealthy caches on the cache stats page.
- **Online Caches:** Number of online caches. Traffic Monitor will not monitor the state of ONLINE servers. True health is unknown.
- **Reported Caches:** Number of caches with REPORTED status.
- **Offline Caches:** Number of caches with OFFLINE status.
- **Admin Down Caches:** Number of caches with ADMIN\_DOWN status.

## Cache Checks

The cache checks page is intended to give an overview of the Servers managed by Traffic Control as well as their status. This data comes from the servers/checks api endpoint.

Name	Description
Host-Name	Cache host name
Profile	The name of the profile to be applied to the cache
Status	The status of the cache (ONLINE, REPORTED..)
UPD	Config updates pending for an edge or mid
RVL	Content invalidation requests are pending for an edge or mid
ILO	Ping the iLO interface for EDGE or MID servers
10G	Ping the IPv4 address of the EDGE or MID servers
FQDN	DNS check that matches what the DNS servers responds with compared to what Traffic Ops has.
DSCP	Checks the DSCP value of packets from the edge server to the Traffic Ops server.
10G6	Ping the IPv6 address of the EDGE or MID servers
MTU	Ping the EDGE or MID using the configured MTU from Traffic Ops
RTR	Content Router checks. Checks the health of the Content Routers. Checks the health of the caches using the Content Routers.
CHR	Cache Hit Ratio in percent.
CDU	Total Cache Disk Usage in percent.
ORT	Operational Readiness Test. Uses the ORT script on the edge and mid servers to determine if the configuration in Traffic Ops matches the configuration on the edge or mid. The user that this script runs as must have an ssh key on the edge servers.

## Cache Stats

Displays health by cache group/profile.

Name	Description
Profile	Name of the profile applied to the edge or mid cache
Host	ALL or grouped by profile
CacheGroup	Cache Group Name
Healthy	true/false
Status	Status of the cache or cacheGroup
Connections	Number of connections
MbpsOut	MbpsOut

## Delivery Services

The fields in the Delivery Service view are:

Name	Description
Active	When this is set to no Traffic Router will not serve DNS or HTTP responses for this delivery service.

Continued on next page

Table 3 – continued from previous page

Name	Description
Content Routing Type	The type of content routing this delivery service will use. See <i>Delivery Service Types</i> .
Key(XML ID)	A unique string that identifies this delivery service.
Display Name	Delivery Service Name.
Tenant	Dropdown to choose the appropriate tenant for your delivery service.
CDN	Dropdown to choose the CDN that the delivery service will use.
Origin Server Base URL	The Origin Server's base URL. This includes the protocol (http or https). Example: <code>http://movies.origin.com</code>
Protocol	The protocol to serve this delivery service to the clients with: <ul style="list-style-type: none"> <li>• 0 http</li> <li>• 1 https</li> <li>• 2 both http and https</li> <li>• 3 http to https</li> </ul>
Long Description	Long description for this delivery service. To be consumed from the APIs by downstream tools (Portal).
Delivery Service URLs	List of URLs derived from Protocol + Routing Name + XML ID + CDN Domain
<b>Advanced</b>	
Routing Name	The routing name to use for the delivery FQDN, i.e. <code>&lt;routing-name&gt;.&lt;deliveryservice&gt;.&lt;cdn-domain&gt;</code> . It must be a valid hostname without periods. <sup>2</sup>
DSCP Tag	The DSCP value to mark IP packets to the client with.
IPv6 Routing Enabled?	When set to yes, the Traffic Router will respond to AAAA DNS requests for the routed name of this delivery service. Otherwise, only A records will be served.
Range Request Handling	(experimental) How to treat range requests: <ul style="list-style-type: none"> <li>• 0 Do not cache (ranges requested from files that are already cached due to a non range request will be a HIT)</li> <li>• 1 Use the <code>background_fetch</code> plugin.</li> <li>• 2 Use the <code>cache_range_requests</code> plugin.</li> </ul>

Continued on next page

Table 3 – continued from previous page

Name	Description
Query String Handling	<p>How to treat query strings:</p> <ul style="list-style-type: none"> <li>• 0 use in cache key and hand up to origin -this means each unique query string Is treated as a unique URL.</li> <li>• 1 Do not use in cache key, but pass up to origin - this means a 2 URLs that are the same except for the query string will match, and cache HIT, while the origin still sees original query string in the request.</li> <li>• 2 Drop at edge - this means a 2 URLs that are the same except for the query string will match, and cache HIT, while the origin will not see original query string in the request.</li> </ul> <p><b>Note:</b> Choosing to drop query strings at the edge will preclude the use of a Regex Remap Expression. See <a href="#">Regex Remap Expression</a>. To set the qstring without the use of regex remap, or for further options, see <a href="#">Qstring Handling</a>.</p>
Use Multi Site Origin Feature	Enable the Multi Site Origin feature for this delivery service. See <a href="#">Multi Site Origin</a>
Logs Enabled	
Geo Provider	ip geolocation provider
Geo Miss Default Latitude	Default Latitude for this delivery service. When client localization fails for both Coverage Zone and Geo Lookup, this the client will be routed as if it was at this lat.
Geo Miss Default Longitude	Default Longitude for this delivery service. When client localization fails for bot Coverage Zone and Geo Lookup, this the client will be routed as if it was at this long.
Geo Limit?	<p>Some services are intended to be limited by geography. The possible settings are are:</p> <ul style="list-style-type: none"> <li>• None - Do not limit by geography.</li> <li>• CZF only - If the requesting IP is not in the Coverage Zone File, do not serve the request.</li> <li>• CZF + US - If the requesting IP is not in the Coverage Zone File or not in the United States, do not serve the request.</li> </ul>
Geo Limit Countries	(for HTTP routed delivery services only) Do not serve the request to specific countries.
Geo Limit Redirect URL	(for HTTP routed delivery services only) This is the URL Traffic Router will redirect to when Geo Limit Failure. See <a href="#">GeoLimit Failure Redirect feature</a>
Signing Algorithm Signing Algorithm	See <a href="#">Token Based Authentication</a> . None URL Signature Keys URI Signing Keys
Bypass Ipv4	(For DNS routed delivery services only) This is the address to respond to A requests with when the the max Bps or Max Tps for this delivery service are exceeded.

Continued on next page

Table 3 – continued from previous page

Name	Description
Bypass IPv6	(For DNS routed delivery services only) This is the address to respond to AAAA requests with when the the max Bps or Max Tps for this delivery service are exceeded.
DNS Bypass Cname	
DNS Bypass TTL	
Max DNS Answers	
Delivery Service DNS TTL	The Time To Live on the DNS record for the Traffic Router A and AAAA records (<routing-name>.<deliveryservice>.<cdn-domain>).
Delivery Service Profile	The profile for this delivery service.
Global Max Mbps	The maximum bits per second this delivery service can serve across all EDGE caches before traffic will be diverted to the bypass destination. For a DNS delivery service, the Bypass Ipv4 or Ipv6 will be used (depending on whether this was a A or AAAA request), and for HTTP delivery services the Bypass FQDN will be used.
Global Max TPS	The maximum transactions per se this delivery service can serve across all EDGE caches before traffic will be diverted to the bypass destination. For a DNS delivery service, the Bypass Ipv4 or Ipv6 will be used (depending on whether this was a A or AAAA request), and for HTTP delivery services the Bypass FQDN will be used.
Signed URLs	Use Signed URLs? See <i>Token Based Authentication</i> .
Edge Header Rewrite Rules	Header Rewrite rules to apply for this delivery service at the EDGE tier. See <i>Header Rewrite Options and DSCP</i> . <sup>1</sup>
Mid Header Rewrite Rules	Header Rewrite rules to apply for this delivery service at the MID tier. See <i>Header Rewrite Options and DSCP</i> . <sup>1</sup>
Traffic Router Additional Response Headers	
Traffic Router Log Request Headers	
Regex Remap Expression	Regex Remap rule to apply to this delivery service at the Edge tier. See <i>ATS documentation on regex_remap</i> . <sup>1</sup> <b>Note:</b> you will not be able to save a Regex Remap Expression if you have Query String Handling set to drop query strings at the edge. See <i>Regex Remap Expression</i> .
Cache URL expression	Cache URL rule to apply to this delivery service. See <i>ATS documentation on cacheurl</i> . <sup>1</sup>
Raw remap text	For HTTP and DNS deliveryservices, this will get added to the end of the remap line on the cache verbatim. For ANY_MAP deliveryservices this is the remap line. <sup>1</sup>
Long Description 2	Customer description for this delivery service. To be consumed from the APIs by downstream tools (Portal).
Long Description 3	Service description for this delivery service. To be consumed from the APIs by downstream tools (Portal).
Info URL	Info URL for this delivery service. To be consumed from the APIs by downstream tools (Portal).

Continued on next page



Table 3 – continued from previous page

Name	Description
Check Path	A path (ex: /crossdomain.xml) to verify the connection to the origin server with. This can be used by Check Extension scripts to do periodic health checks against the delivery service.
Origin Shield (Pipe Delimited String)	Experimental. Origin Shield string.

## Delivery Service Types

One of the most important settings when creating the delivery service is the selection of the delivery service *type*. This type determines the routing method and the primary storage for the delivery service.



Name	Description
HTTP	HTTP Content Routing - The Traffic Router DNS auth server returns its own IP address on DNS queries, and the client gets redirected to a specific cache in the nearest cache group using HTTP 302. Use this for long sessions like HLS/HDS/Smooth live streaming, where a longer setup time is not a problem.
DNS	DNS Content Routing - The Traffic Router DNS auth server returns an edge cache IP address to the client right away. The client will find the cache quickly but the Traffic Router can not route to a cache that already has this content in the cache group. Use this for smaller objects like web page images / objects.
HTTP_NO_CACHE	HTTP Content Routing, but the caches will not actually cache the content, they act as just proxies. The MID tier is bypassed.
HTTP_LIVE	HTTP Content routing, but where for “standard” HTTP content routing the objects are stored on disk, for this delivery service type the objects are stored on the RAM disks. Use this for linear TV. The MID tier is bypassed for this type.
HTTP_LIVE_NATNL	HTTP Content routing, same as HTTP_LIVE, but the MID tier is NOT bypassed.
DNS_LIVE	DNS Content routing, but where for “standard” DNS content routing the objects are stored on disk, for this delivery service type the objects are stored on the RAM disks. Use this for linear TV. The MID tier is NOT bypassed for this type.
DNS_LIVE_NATNL	DNS Content routing, same as DNS_LIVE_NATNL, but the MID tier is bypassed.
ANY_MAP	ANY_MAP is not known to Traffic Router. For this deliveryservice, the “Raw remap text” field in the input form will be used as the remap line on the cache.
STEERING	The Delivery Service will be used to route to other delivery services. The target delivery services and the routing weights for those delivery services will be defined by an admin or steering user. For more information see the <a href="#">steering feature</a> documentation

**Note:** Once created, the Traffic Ops user interface does not allow you to change the delivery service type; the drop down is greyed out. There are many things that can go wrong when changing the type, and it is safer to delete the delivery service, and recreate it.

<sup>2</sup> It is not recommended to change the Routing Name of a Delivery Service after deployment because this changes its Delivery FQDN (i.e. <routing-name>.<deliveryservice>.<cdn-domain>), which means that SSL certificates may need to be updated and clients using the Delivery Service will need to be transitioned to the new Delivery URL.

<sup>1</sup> These fields are not validated by Traffic Ops to be correct syntactically, and can cause Traffic Server to not start if invalid. Please use with caution.

## Servers

This view shows a table of all the servers in Traffic Ops. The table columns show the most important details of the server. The **IPAddr** column is clickable to launch an `ssh://` link to this server. The  icon will link to a Traffic Stats graph of this server for caches, and the  will link to the server status pages for other server types.

## Server Types

These are the types of servers that can be managed in Traffic Ops:

Name	Description
EDGE	Edge Cache
MID	Mid Tier Cache
ORG	Origin
CCR	Traffic Router
RASCAL	Rascal health polling & reporting
TOOLS_SERVER	Ops hosts for management
RIAK	Riak keystore
SPLUNK	SPLUNK indexer search head etc
TRAFFIC_STATS	traffic_stats server
INFLUXDB	influxDb server

## Federations

Federations allow for other (federated) CDNs (at a different ISP, MSO, etc) to add a list of resolvers and a CNAME to a delivery service Traffic Portal. When a request is made from one of federated CDN's clients, Traffic Router will return the CNAME configured in the federation mapping. This allows the federated CDN to serve the content without the content provider changing the URL, or having to manage multiple URLs.

Before adding a federation in the Traffic Portal UI, a user with the federations role needs to be created. This user will be assigned to the federation and will be able to add resolvers to the federation via the Traffic Ops [Federation API](#).

## Header Rewrite Options and DSCP

Most header manipulation and per-delivery service configuration overrides are done using the [ATS Header Rewrite Plugin](#). Traffic Control allows you to enter header rewrite rules to be applied at the edge and at the mid level. The syntax used in Traffic Ops is the same as the one described in the ATS documentation, except for some special strings that will get replaced:

Traffic Ops Entry	Gets Replaced with
<code>__RETURN__</code>	A newline
<code>__CACHE_IPV4__</code>	The cache's IPv4 address

The `deliveryservice` screen also allows you to set the DSCP value of traffic sent to the client. This setting also results in a `header_rewrite` rule to be generated and applied to at the edge.

**Note:** The DSCP setting in the UI is *only* for setting traffic towards the client, and gets applied *after* the initial TCP handshake is complete, and the HTTP request is received (before that the cache can't determine what deliveryservice this request is for, and what DSCP to apply), so the DSCP feature can not be used for security settings - the TCP SYN-ACK is not going to be DSCP marked.

---

## Token Based Authentication

Token based authentication or *signed URLs* is implemented using the Traffic Server `url_sig` plugin. To sign a URL at the signing portal take the full URL, without any query string, and add on a query string with the following parameters:

**Client IP address** The client IP address that this signature is valid for.

`C=<client IP address>`

**Expiration** The Expiration time (seconds since epoch) of this signature.

`E=<expiration time in secs since unix epoch>`

**Algorithm** The Algorithm used to create the signature. Only 1 (HMAC\_SHA1) and 2 (HMAC\_MD5) are supported at this time

`A=<algorithm number>`

**Key index** Index of the key used. This is the index of the key in the configuration file on the cache. The set of keys is a shared secret between the signing portal and the edge caches. There is one set of keys per reverse proxy domain (fqdn).

`K=<key index used>`

**Parts** Parts to use for the signature, always excluding the scheme (<http://>). `parts0` = fqdn, `parts1..x` is the directory parts of the path, if there are more parts to the path than letters in the parts param, the last one is repeated for those. Examples:

1: use fqdn and all of URL path 0110: use part1 and part 2 of path only 01: use everything except the fqdn

`P=<parts string (0's and 1's)>`

**Signature** The signature over the parts + the query string up to and including "S=".

`S=<signature>`

**See also:**

The `url_sig` [README](#).

## Generate URL Sig Keys

To generate a set of random signed url keys for this delivery service and store them in Traffic Vault, click the **Generate URL Sig Keys** button at the bottom of the delivery service details screen.

## Parent Selection

Parameters in the Edge (child) profile that influence this feature:

Name	File-name	Default	Description
CONFIG proxy.config.http.parent_proxy_routing_enable	records.config	INT 1	enable parent selection. This is a required setting.
CONFIG proxy.config.url_remap.remap_required	records.config	INT 1	required for parent selection.
CONFIG proxy.config.http.no_dns_just_forward_to_parent	records.config	INT 0	See
CONFIG proxy.config.http.uncacheable_requests_bypass_parent	records.config	INT 1	
CONFIG proxy.config.http.parent_proxy_routing_enable	records.config	INT 1	
CONFIG proxy.config.http.parent_proxy.retry_time	records.config	INT 300	
CONFIG proxy.config.http.parent_proxy.fail_threshold	records.config	INT 10	
CONFIG proxy.config.http.parent_proxy.total_connect_attempts	records.config	INT 4	
CONFIG proxy.config.http.parent_proxy.per_parent_connect_attempts	records.config	INT 2	
CONFIG proxy.config.http.parent_proxy.connect_attempts_timeout	records.config	INT 30	
CONFIG proxy.config.http.forward.proxy_auth_to_parent	records.config	INT 0	
CONFIG proxy.config.http.parent_proxy_routing_enable	records.config	INT 0	
CONFIG proxy.config.http.parent_proxy.file	records.config	STRING parent.config	
CONFIG proxy.config.http.parent_proxy.connect_attempts_timeout	records.config	INT 3	
algorithm	parent.config	urlhash	The algorithm to use.

Parameters in the Mid (parent) profile that influence this feature:

Name	Filename	Default	Description
domain_name	CRConfig.json	.	Only parents with the same value as the edge are going to be used as parents (to keep separation between CDNs)
weight	parent.config	1.0	The weight of this parent, translates to the number of replicas in the consistent hash ring. This parameter only has effect with algorithm at the client set to "consistent_hash"
port	parent.config	80	The port this parent is listening on as a forward proxy.
use_ip_address	parent.config	0	1 means use IP(v4) address of this parent in the parent.config, 0 means use the host_name.domain_name concatenation.

## Qstring Handling

Delivery services have a Query String Handling option that, when set to ignore, will automatically add a regex remap to that delivery service's config. There may be times this is not preferred, or there may be requirements for one delivery service or server(s) to behave differently. When this is required, the `psel.qstring_handling` parameter can be set in either the delivery service profile or the server profile, but it is important to note that the server profile will override ALL delivery services assigned to servers with this profile parameter. If the parameter is not set for the server profile but is present for the Delivery Service profile, this will override the setting in the delivery service. A value of "ignore" will not result in the addition of regex remap configuration.

Name	Filename	Default	Description
psel.qstring_handling	parent.config	.	Sets qstring handling without the use of regex remap for a delivery service when assigned to a delivery service profile, and overrides qstring handling for all delivery services for associated servers when assigned to a server profile. Value must be "consider" or "ignore".

## Multi Site Origin

**Note:** The configuration of this feature changed significantly between ATS version 5 and  $\geq 6$ . Some configuration in Traffic Control is different as well. This documentation assumes ATS 6 or higher. See [Configure Multi Site Origin](#) for more details.

---

Normally, the mid servers are not aware of any redundancy at the origin layer. With Multi Site Origin enabled this changes - Traffic Server (and Traffic Ops) are now made aware of the fact there are multiple origins, and can be configured to do more advanced failover and loadbalancing actions. A prerequisite for MSO to work is that the multiple origin sites serve identical content with identical paths, and both are configured to serve the same origin hostname as is configured in the deliveryservice *Origin Server Base URL* field. See the [Apache Traffic Server docs](#) for more information on that cache's implementation.

With This feature enabled, origin servers (or origin server VIP names for a site) are going to be entered as servers in to the Traffic Ops UI. Server type is "ORG".

Parameters in the mid profile that influence this feature:

Name	Filename	De- fault	Description
CONFIG proxy.config.http.parent_proxy_routing_enable	records.config	INT 1	enable parent selection. This is a required setting.
CONFIG proxy.config.url_remap.remap_required	records.config	INT 1	required for parent selection.

Parameters in the deliveryservice profile that influence this feature:

Name	Filename	Default	Description
mso.parent_retry	parent.config	-	Either <code>simple_retry</code> , <code>dead_server_retry</code> or both.
mso.algorithm	parent.config	consistent_hash	<p>The algorithm to use. <code>consistent_hash</code>, <code>strict</code>, <code>true</code>, <code>false</code>, or <code>latched</code>.</p> <ul style="list-style-type: none"> <li>• <code>consistent_hash</code> - spreads requests across multiple parents simultaneously based on hash of content URL.</li> <li>• <code>strict</code> - strict Round Robin spreads requests across multiple parents simultaneously based on order of requests.</li> <li>• <code>true</code> - same as <code>strict</code>, but ensures that requests from the same IP always go to the same parent if available.</li> <li>• <code>false</code> - uses only a single parent at any given time and switches to a new parent only if the current parent fails.</li> <li>• <code>latched</code> - same as <code>false</code>, but now, a failed parent will not be retried.</li> </ul>
mso.unavailable_server_retry_response_codes	parent.config	"503"	Quoted, comma separated list of HTTP status codes that count as a <code>unavailable_server_retry_response_code</code> .
mso.max_unavailable_server_retries	parent.config	1	How many times an unavailable server will be retried.
mso.simple_retry_response_codes	parent.config	"404"	Quoted, comma separated list of HTTP status codes that count as a simple retry response code.
mso.max_simple_retries	parent.config	1	How many times a simple retry will be done.

see *Configure Multi Site Origin* for a *quick how to* on this feature.

## Traffic Router Profile

Name	Config_file	Description
location	dns.zone	Location to store the DNS zone files in the local file system of
location	http-log4j.properties	Location to find the log4j.properties file for Traffic Router.
location	dns-log4j.properties	Location to find the dns-log4j.properties file for Traffic Router.
location	geolocation.properties	Location to find the log4j.properties file for Traffic Router.
CDN_name	rascal-config.txt	The human readable name of the CDN for this profile.
CoverageZoneJsonURL	CRConfig.xml	The location (URL) to retrieve the coverage zone map file in JS
geolocation.polling.url	CRConfig.json	The location (URL) to retrieve the geo database file from.
geolocation.polling.interval	CRConfig.json	How often to refresh the coverage geo location database in ms
coveragezone.polling.interval	CRConfig.json	How often to refresh the coverage zone map in ms
coveragezone.polling.url	CRConfig.json	The location (URL) to retrieve the coverage zone map file in X
tld.soa.expire	CRConfig.json	The value for the expire field the Traffic Router DNS Server wi
tld.soa.minimum	CRConfig.json	The value for the minimum field the Traffic Router DNS Serve
tld.soa.admin	CRConfig.json	The DNS Start of Authority admin. Should be a valid support e
tld.soa.retry	CRConfig.json	The value for the retry field the Traffic Router DNS Server will
tld.soa.refresh	CRConfig.json	The TTL the Traffic Router DNS Server will respond with on A
tld.ttls.NS	CRConfig.json	The TTL the Traffic Router DNS Server will respond with on N
tld.ttls.SOA	CRConfig.json	The TTL the Traffic Router DNS Server will respond with on S
tld.ttls.AAAA	CRConfig.json	The Time To Live (TTL) the Traffic Router DNS Server will re
tld.ttls.A	CRConfig.json	The TTL the Traffic Router DNS Server will respond with on A
tld.ttls.DNSKEY	CRConfig.json	The TTL the Traffic Router DNS Server will respond with on D
tld.ttls.DS	CRConfig.json	The TTL the Traffic Router DNS Server will respond with on D
api.port	server.xml	The TCP port Traffic Router listens on for API (REST) access.
api.cache-control.max-age	CRConfig.json	The value of the <code>Cache-Control: max-age=</code> header in
api.auth.url	CRConfig.json	The API authentication URL ( <a href="https://\${tmHostname}/api/1.1/u">https://\${tmHostname}/api/1.1/u</a>
consistent.dns.routing	CRConfig.json	Control whether DNS Delivery Services use consistent hashing
dnssec.enabled	CRConfig.json	Whether DNSSEC is enabled; this parameter is updated via the
dnssec.allow.expired.keys	CRConfig.json	Allow Traffic Router to use expired DNSSEC keys to sign zone
dynamic.cache.primer.enabled	CRConfig.json	Allow Traffic Router to attempt to prime the dynamic zone cac
dynamic.cache.primer.limit	CRConfig.json	Limit the number of permutations to prime when dynamic zone
keystore.maintenance.interval	CRConfig.json	The interval in seconds which Traffic Router will check the key
keystore.api.url	CRConfig.json	The keystore API URL ( <a href="https://\${tmHostname}/api/1.1/cdns/n">https://\${tmHostname}/api/1.1/cdns/n</a>
keystore.fetch.timeout	CRConfig.json	The timeout in milliseconds for requests to the keystore API
keystore.fetch.retries	CRConfig.json	The number of times Traffic Router will attempt to load keys b
keystore.fetch.wait	CRConfig.json	The number of milliseconds Traffic Router will wait before a r
signaturemanager.expiration.multiplier	CRConfig.json	Multiplier used in conjunction with a zone's maximum TTL to
zonemanager.threadpool.scale	CRConfig.json	Multiplier used to determine the number of cores to use for zon
zonemanager.cache.maintenance.interval	CRConfig.json	The interval in seconds which Traffic Router will check for zon
zonemanager.dynamic.response.expiration	CRConfig.json	A string (e.g.: 300s) that defines how long a dynamic zone
DNSKEY.generation.multiplier	CRConfig.json	Used to determine when new keys need to be regenerated. Ke
DNSKEY.effective.multiplier	CRConfig.json	Used when creating an effective date for a new key set. New ke

## Regex Remap Expression

The regex remap expression allows to to use a regex and resulting match group(s) in order to modify the request URIs that are sent to origin. For example:



```
^/original/(.*) http://origin.example.com/remapped/$1
```

**Note:** If **Query String Handling** is set to 2 Drop at edge, then you will not be allowed to save a regex remap expression, as dropping query strings actually relies on a regex remap of its own. However, if there is a need to both drop query strings **and** remap request URIs, this can be accomplished by setting **Query String Handling** to 1 Do not use in cache key, but pass up to origin, and then using a custom regex remap expression to do the necessary remapping, while simultaneously dropping query strings. The following example will capture the original request URI up to, but not including, the query string and then forward to a remapped URI:

```
^/([?]*).* http://origin.example.com/remapped/$1
```

## Delivery Service Regexp

This table defines how requests are matched to the delivery service. There are 3 type of entries possible here:

Name	Description	DS Type	Status
HOST_REGEXP	This is the regular expression to match the host part of the URL.	DNS and HTTP	Supported
PATH_REGEXP	This is the regular expression to match the path part of the URL.	HTTP	Beta
HEADER_REGEXP	This is the regular expression to match on any header in the request.	HTTP	Beta

The **Order** entry defines the order in which the regular expressions get evaluated. To support CNAMEs from domains outside of the Traffic Control top level DNS domain, enter multiple HOST\_REGEXP lines.

**Example:** Example foo.

**Note:** In most cases is is sufficient to have just one entry in this table that has a HOST\_REGEXP Type, and Order 0. For the *movies* delivery service in the Kabletown CDN, the entry is simply single HOST\_REGEXP set to `.*\.movies\..*`. This will match every url that has a hostname that ends with `movies.cdn1.kabletown.net`, since `cdn1.kabletown.net` is the Kabletown CDN's DNS domain.

## Static DNS Entries

Static DNS entries allow you to create other names *under* the delivery service domain. You can enter any valid hostname, and create a CNAME, A or AAAA record for it by clicking the **Static DNS** button at the bottom of the delivery service details screen.

## Server Assignments

Click the **Server Assignments** button at the bottom of the screen to assign servers to this delivery service. Servers can be selected by drilling down in a tree, starting at the profile, then the cache group, and then the individual servers. Traffic Router will only route traffic for this delivery service to servers that are assigned to it.

## The Coverage Zone File and ASN Table

The Coverage Zone File (CZF) should contain a cachegroup name to network prefix mapping in the form:

```
{
  "coverageZones": {
    "cache-group-01": {
      "network6": [
        "1234:5678::\64",
        "1234:5679::\64"
      ],
      "network": [
        "192.168.8.0\24",
        "192.168.9.0\24"
      ]
    }
    "cache-group-02": {
      "network6": [
        "1234:567a::\64",
        "1234:567b::\64"
      ],
      "network": [
        "192.168.4.0\24",
        "192.168.5.0\24"
      ]
    }
  }
}
```

The CZF is an input to the Traffic Control CDN, and as such does not get generated by Traffic Ops, but rather, it gets consumed by Traffic Router. Some popular IP management systems output a very similar file to the CZF but in stead of a cachegroup an ASN will be listed. Traffic Ops has the “Networks (ASNs)” view to aid with the conversion of files like that to a Traffic Control CZF file; this table is not used anywhere in Traffic Ops, but can be used to script the conversion using the API.

The script that generates the CZF file is not part of Traffic Control, since it is different for each situation.

## Parameters and Profiles

Parameters are shared between profiles if the set of { name, config\_file, value } is the same. To change a value in one profile but not in others, the parameter has to be removed from the profile you want to change it in, and a new parameter entry has to be created (**Add Parameter** button at the bottom of the Parameters view), and assigned to that profile. It is easy to create new profiles from the **Misc > Profiles** view - just use the **Add/Copy Profile** button at the bottom of the profile view to copy an existing profile to a new one. Profiles can be exported from one system and imported to another using the profile view as well. It makes no sense for a parameter to not be assigned to a single profile - in that case it really has no function. To find parameters like that use the **Parameters > Orphaned Parameters** view. It is easy to create orphaned parameters by removing all profiles, or not assigning a profile directly after creating the parameter.

### See also:

*Profile Parameters* in the *Configuring Traffic Ops* section.

## Tools

## Generate ISO

Generate ISO is a tool for building custom ISOs for building caches on remote hosts. Currently supports Centos 6 and 7, but if you're brave and pure of heart you MIGHT be able to get it to work with other unix-like OS's.

The interface is *mostly* self explanatory as it's got hints.

Field	Explanation
Choose a server from list:	This option gets all the server names currently in the Traffic Ops database and will autofill known values.
OS Version:	There needs to be an <code>_osversions.cfg_</code> file in the ISO directory that maps the name of a directory to a name that shows up here.
Hostname:	This is the FQDN of the server to be installed. It is required.
Root password:	If you don't put anything here it will default to the salted MD5 of "Fred". Whatever put is MD5 hashed and writte to disk.
DHCP:	if yes, other IP settings will be ignored
IP Address:	Required if DHCP=no
Netmask:	Required if DHCP=no
Gateway:	Required if DHCP=no
IPV6 Address:	Optional. /64 is assumed if prefix is omitted
IPV6 Gateway:	Ignored if an IPV4 gateway is specified
Network Device:	Optional. Typical values are bond0, eth4, etc. Note: if you enter bond0, a LACP bonding config will be written
MTU:	If unsure, set to 1500
Specify disk for OS install:	Optional. Typical values are "sda".

When you click the **Download ISO** button the folling occurs (all paths relative to the top level of the directory specified in `_osversions.cfg_`):

1. Reads `/etc/resolv.conf` to get a list of nameservers. This is a rather ugly hack that is in place until we get a way of configuring it in the interface.
2. Writes a file in the `ks_scripts/state.out` that contains directory from `_osversions.cfg_` and the `mkisofs` string that we'll call later.
3. Writes a file in the `ks_scripts/network.cfg` that is a bunch of key=value pairs that set up networking.
4. Creates an MD5 hash of the password you specify and writes it to `ks_scripts/password.cfg`. Note that if you do not specify a password "Fred" is used. Also note that we have experienced some issues with webrowsers autofilling that field.
5. Writes out a disk configuration file to `ks_scripts/disk.cfg`.
6. `mkisofs` is called against the directory configured in `_osversions.cfg_` and an ISO is generated in memory and delivered to your webbrowser.

You now have a customized ISO that can be used to install Red Hat and derivative Linux installations with some modifications to your `ks.cfg` file.

Kickstart/Anaconda will mount the ISO at `/mnt/stage2` during the install process (at least with 6).

You can directly include the password file anywhere in your `ks.cfg` file (usually in the top) by doing `%include /mnt/stage2/ks_scripts/password.cfg`

What we currently do is have 2 scripts, one to do hard drive configuration and one to do network configuration. Both are relatively specific to the environment they were created in, and both are *probably* wrong for other organizations, however they are currently living in the "misc" directory as examples of how to do things.

We trigger those in a %pre section in ks.cfg and they will write config files to /tmp. We will then include those files in the appropriate places using %pre.

For example this is a section of our ks.cfg file:

```
%include /mnt/stage2/ks_scripts/packages.txt



```
%pre
  python /mnt/stage2/ks_scripts/create_network_line.py
  bash /mnt/stage2/ks_scripts/drive_config.sh
%end
```


```

These two scripts will then run `_before_` anaconda sets up it's internal structures, then a bit further up in the ks.cfg file (outside of the %pre %end block) we do an

```
%include /mnt/stage2/ks_scripts/password.cfg
...
%include /tmp/network_line

%include /tmp/drive_config
...
```

This snarfs up the contents and inlines them.

If you only have one kind of hardware on your CDN it is probably best to just put the drive config right in the ks.cfg.

If you have simple networking needs (we use bonded interfaces in most, but not all locations and we have several types of hardware meaning different ethernet interface names at the OS level etc.) then something like this:

```
#!/bin/bash
source /mnt/stage2/ks_scripts/network.cfg
echo "network --bootproto=static --activate --ipv6=$IPV6ADDR --ip=$IPADDR --netmask=
↪$NETMASK --gateway=$GATEWAY --ipv6gateway=$GATEWAY --nameserver=$NAMESERVER --mtu=
↪$MTU --hostname=$HOSTNAME" >> /tmp/network.cfg
# Note that this is an example and may not work at all.
```

You could also put this in the %pre section. Lots of ways to solve it.

We have included the two scripts we use in the “misc” directory of the git repo:

- kickstart\_create\_network\_line.py
- kickstart\_drive\_config.sh

These scripts were written to support a very narrow set of expectations and environment and are almost certainly not suitable to just drop in, but they might provide a good starting point.

## Queue Updates and Snapshot CRConfig

When changing delivery services special care has to be taken so that Traffic Router will not send traffic to caches for delivery services that the cache doesn't know about yet. In general, when adding delivery services, or adding servers to a delivery service, it is best to update the caches before updating Traffic Router and Traffic Monitor. When deleting delivery services, or deleting server assignments to delivery services, it is best to update Traffic Router and Traffic Monitor first and then the caches. Updating the cache configuration is done through the *Queue Updates* menu, and updating Traffic Monitor and Traffic Router config is done through the *Snapshot CRConfig* menu.



## Queue Updates

Every 15 minutes the caches should run a *syncds* to get all changes needed from Traffic Ops. The files that will be updated by the syncds job are:

- records.config
- remap.config
- parent.config
- cache.config
- hosting.config
- url\_sig\_(.\*).config
- hdr\_rw\_(.\*).config
- regex\_revalidate.config
- ip\_allow.config

A cache will only get updated when the update flag is set for it. To set the update flag, use the *Queue Updates* menu - here you can schedule updates for a whole CDN or a cache group:

1. Click **Tools > Queue Updates**.
2. Select the CDN to queue updates for, or All.
3. Select the cache group to queue updates for, or All
4. Click the **Queue Updates** button.
5. When the Queue Updates for this Server? (all) window opens, click **OK**.

To schedule updates for just one cache, use the “Server Checks” page, and click the  in the *UPD* column. The *UPD* column of Server Checks page will change show a  when updates are pending for that cache.

## Snapshot CRConfig

Every 60 seconds Traffic Monitor will check with Traffic Ops to see if a new CRConfig snapshot exists; Traffic Monitor polls Traffic Ops for a new CRConfig, and Traffic Router polls Traffic Monitor for the same file. This is necessary to ensure that Traffic Monitor sees configuration changes first, which helps to ensure that the health and state of caches and delivery services propagates properly to Traffic Router. See [Traffic Router Profile](#) for more information on the CRConfig file.

To create a new snapshot, use the *Tools > Snapshot CRConfig* menu:

1. Click **Tools > Snapshot CRConfig**.
2. Verify the selection of the correct CDN from the Choose CDN drop down and click **Diff CRConfig**. On initial selection of this, the CRConfig Diff window says the following:

There is no existing CRConfig for [cdn] to diff against. . . Is this the first snapshot??? If you are not sure why you are getting this message, please do not proceed! To proceed writing the snapshot anyway click the ‘Write CRConfig’ button below.

If there is an older version of the CRConfig, a window will pop up showing the differences between the active CRConfig and the CRConfig about to be written.

3. Click **Write CRConfig**.

4. When the This will push out a new CRConfig.json. Are you sure? window opens, click **OK**.
5. The Successfully wrote CRConfig.json! window opens, click **OK**.

### Invalidate Content

Invalidating content on the CDN is sometimes necessary when the origin was mis-configured and something is cached in the CDN that needs to be removed. Given the size of a typical Traffic Control CDN and the amount of content that can be cached in it, removing the content from all the caches may take a long time. To speed up content invalidation, Traffic Ops will not try to remove the content from the caches, but it makes the content inaccessible using the *regex\_revalidate* ATS plugin. This forces a *revalidation* of the content, rather than a new get.

---

**Note:** This method forces a HTTP *revalidation* of the content, and not a new *GET* - the origin needs to support revalidation according to the HTTP/1.1 specification, and send a 200 OK or 304 Not Modified as applicable.

---

To invalidate content:

1. Click **Tools > Invalidate Content**
2. Fill out the form fields:
  - Select the **Delivery Service**
  - Enter the **Path Regex** - this should be a **PCRE** compatible regular expression for the path to match for forcing the revalidation. Be careful to only match on the content you need to remove - revalidation is an expensive operation for many origins, and a simple `/.*` can cause an overload condition of the origin.
  - Enter the **Time To Live** - this is how long the revalidation rule will be active for. It usually makes sense to make this the same as the `Cache-Control` header from the origin which sets the object time to live in cache (by `max-age` or `Expires`). Entering a longer TTL here will make the caches do unnecessary work.
  - Enter the **Start Time** - this is the start time when the revalidation rule will be made active. It is pre-populated with the current time, leave as is to schedule ASAP.
3. Click the **Submit** button.

### Manage DNSSEC Keys

In order to support **DNSSEC** in Traffic Router, Traffic Ops provides some actions for managing DNSSEC keys for a CDN and associated Delivery Services. DNSSEC Keys consist of a Key Signing Keys (KSK) which are used to sign other DNSKEY records as well as Zone Signing Keys (ZSK) which are used to sign other records. DNSSEC Keys are stored in **Traffic Vault** and should only be accessible to Traffic Ops. Other applications needing access to this data, such as Traffic Router, must use the Traffic Ops **DNSSEC APIs** to retrieve this information.

**To Manage DNSSEC Keys:**

1. Click **Tools -> Manage DNSSEC Keys**
2. Choose a CDN and click **Manage DNSSEC Keys**
  - If keys have not yet been generated for a CDN, this screen will be mostly blank with just the **CDN** and **DNSSEC Active?** fields being populated.
  - If keys have been generated for the CDN, the Manage DNSSEC Keys screen will show the TTL and Top Level Domain (TLD) KSK Expiration for the CDN as well as DS Record information which will need to be added to the parent zone of the TLD in order for DNSSEC to work.

The Manage DNSSEC Keys screen also allows a user to perform the following actions:

#### Activate/Deactivate DNSSEC for a CDN

Fairly straight forward, this button set the **dnssec.enabled** param to either **true** or **false** on the Traffic Router profile for the CDN. The Activate/Deactivate option is only available if DNSSEC keys exist for CDN. In order to active DNSSEC for a CDN a user must first generate keys and then click the **Active DNSSEC** button.

#### Generate Keys

Generate Keys will generate DNSSEC keys for the CDN TLD as well as for each Delivery Service in the CDN. It is important to note that this button will create a new KSK for the TLD and, therefore, a new DS Record. Any time a new DS Record is created, it will need to be added to the parent zone of the TLD in order for DNSSEC to work properly. When a user clicks the **Generate Keys** button, they will be presented with a screen with the following fields:

- **CDN:** This is not editable and displays the CDN for which keys will be generated
- **ZSK Expiration (Days):** Sets how long (in days) the Zone Signing Key will be valid for the CDN and associated Delivery Services. The default is 30 days.
- **KSK Expiration (Days):** Sets how long (in days) the Key Signing Key will be valid for the CDN and associated Delivery Services. The default is 365 days.
- **Effective Date (GMT):** The time from which the new keys will be active. Traffic Router will use this value to determine when to start signing with the new keys and stop signing with the old keys.

Once these fields have been correctly entered, a user can click Generate Keys. The user will be presented with a confirmation screen to help them understand the impact of generating the keys. If a user confirms, the keys will be generated and stored in Traffic Vault.

#### Regenerate KSK

Regenerate KSK will create a new Key Signing Key for the CDN TLD. A new DS Record will also be generated and need to be put into the parent zone in order for DNSSEC to work correctly. The **Regenerate KSK** button is only available if keys have already been generated for a CDN. The intent of the button is to provide a mechanism for generating a new KSK when a previous one expires or if necessary for other reasons such as a security breach. When a user goes to generate a new KSK they are presented with a screen with the following options:

- **CDN:** This is not editable and displays the CDN for which keys will be generated
- **KSK Expiration (Days):** Sets how long (in days) the Key Signing Key will be valid for the CDN and associated Delivery Services. The default is 365 days.
- **Effective Date (GMT):** The time from which the new KSK and DS Record will be active. Since generating a new KSK will generate a new DS Record that needs to be added to the parent zone, it is very important to make sure that an effective date is chosen that allows for time to get the DS Record into the parent zone. Failure to get the new DS Record into the parent zone in time could result in DNSSEC errors when Traffic Router tries to sign responses.

Once these fields have been correctly entered, a user can click Generate KSK. The user will be presented with a confirmation screen to help them understand the impact of generating the KSK. If a user confirms, the KSK will be generated and stored in Traffic Vault.

Additionally, Traffic Ops also performs some systematic management of DNSSEC keys. This management is necessary to help keep keys in sync for Delivery Services in a CDN as well as to make sure keys do not expire without human intervention.

#### Generation of keys for new Delivery Services

If a new Delivery Service is created and added to a CDN that has DNSSEC enabled, Traffic Ops will create DNSSEC keys for the Delivery Service and store them in Traffic Vault.

#### Regeneration of expiring keys for a Delivery Service

Traffic Ops has a process, controlled by cron, to check for expired or expiring keys and re-generate them. The process runs at 5 minute intervals to check and see if keys are expired or close to expiring (withing 10 minutes by default). If keys are expired for a Delivery Service, traffic ops will regenerate new keys and store them in Traffic Vault. This process is the same for the CDN TLD ZSK, however Traffic Ops will not re-generate the CDN TLD KSK systematically. The reason is that when a KSK is regenerated for the CDN TLD then a new DS Record will also be created. The new DS Record needs to be added to the parent zone before Traffic Router attempts to sign with the new KSK in order for DNSSEC to work correctly. Therefore, management of the KSK needs to be a manual process.

### 3.1.10 Traffic Monitor Administration (Legacy)

- These instructions are for the legacy Java Traffic Monitor, for the new Golang version, see [here](#).

#### Installing Traffic Monitor

The following are requirements to ensure an accurate set up:

- CentOS 6
  - 4 vCPUs
  - 8GB RAM
  - Successful install of Traffic Ops
  - Tomcat
  - Administrative access to the Traffic Ops
  - Physical address of the site
  - perl-JSON
  - perl-WWW-Curl
1. Add the Traffic Monitor server into Traffic Ops using ‘Servers’ -> ‘Add Server’. Set the ‘Type’ field to ‘RAS-CAL’.
  2. Make sure the FQDN of the Traffic Monitor is resolvable in DNS.
  3. Get the Traffic Monitor RPM.

Sample command:

```
wget http://traffic-control-cdn.net/downloads/1.7.0/RELEASE-1.7.0/traffic_monitor-1.7.0-3908.5b77f60f.el6.x86_64.rpm
```

4. Install Traffic Monitor and Perl modules:

```
sudo yum -y install traffic_monitor-*.rpm perl-JSON perl-WWW-Curl
```

5. Take the config from Traffic Ops:

```
sudo /opt/traffic_monitor/bin/traffic_monitor_config.pl https://<traffic-ops-URL>  
↪<traffic-ops-user>:<traffic-ops-password> prompt
```

Sample session:



```

traffic_mon # /opt/traffic_monitor/bin/traffic_monitor_config.pl https://traffic-
↳ops.cdn.kabletown.net admin:kl0tevox prompt
DEBUG: traffic_ops selected: https://traffic-ops.cdn.kabletown.net
DEBUG: traffic_ops login: admin:kl0tevox
DEBUG: Config write mode: prompt
DEBUG: Found profile from traffic_ops: RASCAL_CDN
DEBUG: Found CDN name from traffic_ops: kabletown_cdn
DEBUG: Found location for rascal-config.txt from traffic_ops: /opt/traffic_
↳monitor/conf
WARN: Param not in traffic_ops: allow.config.edit
↳description: Allow the running configuration to be edited through the UI
↳
↳Using default value of:
↳false
WARN: Param not in traffic_ops: default.accessControlAllowOrigin
↳description: The value for the header: Access-Control-Allow-Origin for
↳published jsons... should be narrowed down to TMs
↳Using default
↳value of: *
WARN: Param not in traffic_ops: default.connection.timeout
↳description: Default connection time for all queries (cache, peers, TM)
↳
↳Using default value of:
↳2000
WARN: Param not in traffic_ops: hack.forceSystemExit
↳description: Call System.exit on shutdown
↳
↳Using default value of:
↳false
WARN: Param not in traffic_ops: hack.peerOptimistic
↳description: The assumption of a caches availability when unknown by peers
↳
↳Using default value of:
↳true
WARN: Param not in traffic_ops: hack.publishDsStates
↳description: If true, the delivery service states will be included in the
↳CrStates.json
↳Using default value
↳of: true
WARN: Param not in traffic_ops: health.ds.interval
↳description: The polling frequency for calculating the deliveryService states
↳
↳Using default value of:
↳1000
WARN: Param not in traffic_ops: health.ds.leniency
↳description: The amount of time before the deliveryService disregards the last
↳update from a non-responsive cache
↳Using default value of:
↳30000
WARN: Param not in traffic_ops: health.event-count
↳description: The number of historical events that will be kept
↳
↳Using default value of:
↳200
WARN: Param not in traffic_ops: health.polling.interval
↳description: The polling frequency for getting the states from caches
↳
↳Using default value of:
↳5000
WARN: Param not in traffic_ops: health.startupMinCycles
↳description: The number of query cycles that must be completed before this
↳Traffic Monitor will start reporting
↳Using default value
↳of: 2
WARN: Param not in traffic_ops: health.timepad
↳description: A delay between each separate cache query
↳
↳Using default value of: 10
WARN: Param not in traffic_ops: peers.polling.interval
↳description: Polling frequency for getting states from peer monitors
↳
↳Using default value of:
↳5000

```

(continues on next page)

(continued from previous page)

```

WARN: Param not in traffic_ops: peers.polling.url
↳description: The url for current, unfiltered states from peer monitors
↳
↳http://${hostname}/publish/CrStates?raw
WARN: Param not in traffic_ops: peers.threadPool
↳description: The number of threads given to the pool for querying peers
↳
↳Using default value of: 1
WARN: Param not in traffic_ops: tm.auth.url
↳description: The url for the authentication form
↳
↳Using default value of:
↳https://${tmHostname}/login
WARN: Param not in traffic_ops: tm.crConfig.json.polling.url
↳description: Url for the cr-config (json)
↳
↳Using default value of:
↳https://${tmHostname}/CRConfig-Snapshots/${cdnName}/CRConfig.json
WARN: Param not in traffic_ops: tm.healthParams.polling.url
↳description: The url for the heath params (json)
↳
↳Using default value of:
↳https://${tmHostname}/health/${cdnName}
WARN: Param not in traffic_ops: tm.polling.interval
↳description: The polling frequency for getting updates from TM
↳
↳Using default value of:
↳10000
DEBUG: allow.config.edit needed in config, but does not exist in config on disk.
DEBUG: cdnName value on disk () does not match value needed in config (kabletown_
↳cdn).
DEBUG: default.accessControlAllowOrigin needed in config, but does not exist in
↳config on disk.
DEBUG: default.connection.timeout needed in config, but does not exist in config
↳on disk.
DEBUG: hack.forceSystemExit needed in config, but does not exist in config on
↳disk.
DEBUG: hack.peerOptimistic needed in config, but does not exist in config on disk.
DEBUG: hack.publishDsStates needed in config, but does not exist in config on
↳disk.
DEBUG: health.ds.interval needed in config, but does not exist in config on disk.
DEBUG: health.ds.leniency needed in config, but does not exist in config on disk.
DEBUG: health.startupMinCycles needed in config, but does not exist in config on
↳disk.
DEBUG: health.timepad value on disk (20) does not match value needed in config
↳(10).
DEBUG: peers.polling.interval needed in config, but does not exist in config on
↳disk.
DEBUG: peers.threadPool needed in config, but does not exist in config on disk.
DEBUG: tm.auth.password value on disk () does not match value needed in config
↳(kl0tevax).
DEBUG: tm.auth.username value on disk () does not match value needed in config
↳(admin).
DEBUG: tm.hostname value on disk () does not match value needed in config
↳(traffic-ops.cdn.kabletown.net).
DEBUG: Proposed traffic_monitor_config:
{
  "traffic_monitor_config":{
    "default.accessControlAllowOrigin": "*",
    "health.startupMinCycles": "2",
    "tm.auth.password": "kl0tevax",
    "tm.auth.url": "https://${tmHostname}/login",

```

(continues on next page)

(continued from previous page)

```

    "tm.healthParams.polling.url":"https://${tmHostname}/health/${cdnName}",
    "allow.config.edit":"false",
    "tm.crConfig.json.polling.url":"https://${tmHostname}/CRConfig-Snapshots/${
↪{cdnName}/CRConfig.json",
    "tm.auth.username":"admin",
    "peers.polling.url":"http://${hostname}/publish/CrStates?raw",
    "health.timepad":"10",
    "hack.publishDsStates":"true",
    "default.connection.timeout":"2000",
    "health.ds.interval":"1000",
    "peers.polling.interval":"5000",
    "hack.forceSystemExit":"false",
    "health.ds.leniency":"30000",
    "cdnName":"kabletown_cdn",
    "peers.threadPool":"1",
    "tm.polling.interval":"10000",
    "health.polling.interval":"5000",
    "health.event-count":"200",
    "hack.peerOptimistic":"true",
    "tm.hostname":"traffic-ops.cdn.kabletown.net"
  }
}
-----
----OK to write this config to disk? (Y/n) [n]y
-----
----OK to write this config to disk? (Y/n) [n]Y
-----
DEBUG: Writing /opt/traffic_monitor/conf/traffic_monitor_config.js
traffic_mon #

```

6. Update the 'allow\_ip' and 'allow\_ip6' parameters in the profiles of all caches defined in traffic ops, both edge and mid, with the address of the traffic monitor being installed, so that the traffic servers will allow this Traffic Monitor to access the astats plugin. For details see [Profile Parameters](#) in the *Configuring Traffic Ops* section.
7. Start Tomcat: `sudo service tomcat start`

```

Using CATALINA_BASE: /opt/tomcat
Using CATALINA_HOME: /opt/tomcat
Using CATALINA_TMPDIR: /opt/tomcat/temp
Using JRE_HOME: /usr
Using CLASSPATH:/opt/tomcat/bin/bootstrap.jar
Using CATALINA_PID:/var/run/tomcat/tomcat.pid
Starting tomcat [ OK ]

```

8. Configure tomcat to start automatically: `sudo chkconfig tomcat on`
9. Verify Traffic Monitor is running by pointing your browser to port 80 on the Traffic Monitor host:
  - The 'Cache States' tab should display all Mid and Edge caches configured in Traffic Ops.
  - The 'DeliveryService States' tab should display all delivery services configured in Traffic Ops.
10. In Traffic Ops servers table, click 'Edit' for this server, then click 'Online'.

## Configuring Traffic Monitor

## Configuration Overview

Traffic Monitor is configured using its JSON configuration file, `/opt/traffic_monitor/conf/traffic_monitor_config.js`. This file is created by `traffic_monitor_config.pl` script, and among other things, it contains the Traffic Ops URL and the `user:password` specified during the invocation of that script.

When started, Traffic Monitor uses this basic configuration to download its configuration from Traffic Ops, and begins polling caches. Once a configurable number of polling cycles completes, health protocol state is available via RESTful JSON endpoints.

## Troubleshooting and log files

Traffic Monitor log files are in `/opt/traffic_monitor/var/log/`, and tomcat log files are in `/opt/tomcat/logs/`.

### 3.1.11 Traffic Monitor Administration

- These instructions are for the Golang Traffic Monitor, for the legacy Java version, see [here](#).

## Installing Traffic Monitor

The following are requirements to ensure an accurate set up:

- CentOS 6
- 8 vCPUs
- 16GB RAM
- Successful install of Traffic Ops
- Administrative access to the Traffic Ops
- Physical address of the site

1. Enter the Traffic Monitor server into Traffic Ops
2. Make sure the FQDN of the Traffic Monitor is resolvable in DNS.
3. Install Traffic Monitor: `sudo yum -y install traffic_monitor`
4. Configure Traffic Monitor. See [here](#)
5. Start the service: `sudo service traffic_monitor start`

```
Starting traffic_monitor:
```

6. Verify Traffic Monitor is running by pointing your browser to port 80 on the Traffic Monitor host.

## Configuring Traffic Monitor

### Configuration Overview

Traffic Monitor is configured via two JSON configuration files, `traffic_ops.cfg` and `traffic_monitor.cfg`, by default located in the `conf` directory in the install location.

The `traffic_ops.cfg` config contains Traffic Ops connection information. Specify the URL, username, and password for the instance of Traffic Ops for which this Traffic Monitor is a member.

The `traffic_monitor.cfg` config contains log file locations, as well as detailed application configuration variables, such as processing flush times and initial poll intervals.

Once started with the correct configuration, Traffic Monitor downloads its configuration from Traffic Ops and begins polling caches. Once every cache has been polled, health protocol state is available via RESTful JSON endpoints.

## Troubleshooting and log files

Traffic Monitor log files are in `/opt/traffic_monitor/var/log/`.

### 3.1.12 Traffic Router Administration

#### Contents

- *Traffic Router Administration*
  - *Installing Traffic Router*
  - *Configuring Traffic Router*
  - *DNSSEC*
  - *Troubleshooting and log files*
  - *Event Log File Format*
  - *GeoLimit Failure Redirect feature*
  - *Deep Caching - Deep Coverage Zone Topology*
  - *Steering feature*
  - *HTTPS for Http Type Delivery Services*
  - *Tuning Recommendations*

## Installing Traffic Router

The following are requirements to ensure an accurate set up:

- CentOS 6
- 4 vCPUs
- 8GB RAM
- Successful install of Traffic Ops
- Successful install of Traffic Monitor
- Administrative access to Traffic Ops

---

**Note:** Hardware requirements are generally doubled if DNSSEC is enabled

---

1. If no suitable profile exists, create a new profile for Traffic Router.

2. Enter the Traffic Router server into Traffic Ops, assign it to a Traffic Router profile, and ensure that its status is set to `ONLINE`.
3. Ensure the FQDN of the Traffic Router is resolvable in DNS. This FQDN must be resolvable by the clients expected to use this CDN.
4. Install a traffic router: `sudo yum install traffic_router`.
5. **Edit `/opt/traffic_router/conf/traffic_monitor.properties` and specify the correct online Traffic Monitor.**  
# traffic\_monitor.properties: url that should normally point to this file traffic\_monitor.properties=file:/opt/traffic\_router/conf/traffic\_monitor.properties  
  
# Frequency for reloading this file # traffic\_monitor.properties.reload.period=60000
6. **Start Tomcat: `sudo service tomcat start`, and test lookups with `dig` and `curl` against that server.**  
To restart, `sudo service tomcat stop`, kill the traffic router process, and `sudo service tomcat start` Also, `crconfig` previously recieved will be cached, and needs to be removed manually to actually be reloaded `/opt/traffic_router/db/cr-config.json`
7. Snapshot CRConfig; See [Snapshot CRConfig](#)

---

**Note:** Once the CRConfig is snapshotted, live traffic will be sent to the new Traffic Routers provided that their status is set to `ONLINE`.

---

8. Ensure that the parent domain (e.g.: `kabletown.net`) for the CDN's top level domain (e.g.: `cdn.kabletown.net`) contains a delegation (NS records) for the new Traffic Router, and that the value specified matches the FQDN used in step 3.

## Configuring Traffic Router

---

**Note:** Starting with Traffic Router 1.5, many of the configuration files under `/opt/traffic_router/conf` are only needed to override the default configuration values for Traffic Router. Most of the given default values will work well for any CDN. Critical values that must be changed are hostnames and credentials for communicating with other Traffic Control components such as Traffic Ops and Traffic Monitor.

---

---

**Note:** Pre-existing installations having configuration files in `/opt/traffic_router/conf` will still be used and honored for Traffic Router 1.5 and onward.

---

For the most part, the configuration files and parameters that follow are used to get Traffic Router online and communicating with various Traffic Control components. Once Traffic Router is successfully communicating with Traffic Control, configuration is mostly performed in Traffic Ops, and is distributed throughout Traffic Control via the CR-Config snapshot process. See [Snapshot CRConfig](#) for more information. Please see the parameter documentation for Traffic Router in the Using Traffic Ops guide documented under [Traffic Router Profile](#) for parameters that influence the behavior of Traffic Router via the CRConfig.

## Configuration files

File name	Parameter	Description	Default Value
traffic_monitor.properties	traffic_monitor.bootstrap_hosts	Traffic Monitor FQDNs and port if necessary, separated by a semicolon (;)	N/A
	traffic_monitor.bootstrap_logfile	Use only the Traffic Monitors specified in config file	false
	traffic_monitor.properties	Path to the traffic_monitor.properties file; used internally to monitor the file for changes	/opt/traffic_router/traffic_monitor.properties
	traffic_monitor.properties.reloadperiod	The interval in milliseconds which Traffic Router will reload this configuration file	60000
dns.properties	dns.tcp.port	TCP port that Traffic Router will use for incoming DNS requests	53
	dns.tcp.backlog	Maximum length of the queue for incoming TCP connection requests	0
	dns.udp.port	UDP port that Traffic Router will use for incoming DNS requests	53
	dns.max-threads	Maximum number of threads used to process incoming DNS requests	1000
	dns.zones.dir	Path to auto generated zone files for reference	/opt/traffic_router/var/auto-zones
traffic_ops.properties	traffic_ops.username	Username to access the APIs in Traffic Ops (must be in the admin role)	admin
	traffic_ops.password	Password for the user specified in traffic_ops.username	N/A
cache.properties	cache.geolocation.database	Full path to the local copy of the MaxMind geolocation binary database file	/opt/traffic_router/db/GeoIP2-City.mmdb
	cache.geolocation.database.refreshperiod	The interval in milliseconds which Traffic Router will poll for a new geolocation database	604800000
	cache.czmap.database	Full path to the local copy of the coverage zone file	/opt/traffic_router/db/czmap.json
	cache.czmap.database.refreshperiod	The interval in milliseconds which Traffic Router will poll for a new coverage zone file	10800000
	cache.dczmap.database	Full path to the local copy of the deep coverage zone file	/opt/traffic_router/db/dczmap.json
	cache.dczmap.database.refreshperiod	The interval in milliseconds which Traffic Router will poll for a new deep coverage zone file	10800000
	cache.health.json	Full path to the local copy of the health state	/opt/traffic_router/db/health.json
	cache.health.json.refreshperiod	The interval in milliseconds which Traffic Router will poll for a new health state file	1000
	cache.config.json	Full path to the local copy of the CRConfig	/opt/traffic_router/db/cr-config.json
	cache.config.json.refreshperiod	The interval in milliseconds which Traffic Router will poll for a new CRConfig	60000
log4j.properties	various parameters	Configuration of log4j is documented on their site; adjust as necessary based on needs	N/A

## DNSSEC

## Overview

Domain Name System Security Extensions (DNSSEC) is a set of extensions to DNS that provides a cryptographic mechanism for resolvers to verify the authenticity of responses served by an authoritative DNS server.

Several RFCs (4033, 4044, 4045) describe the low level details and define the extensions, RFC 7129 provides clarification around authenticated denial of existence of records, and finally RFC 6781 describes operational best practices for administering an authoritative DNSSEC enabled DNS server. The authenticated denial of existence RFC describes how an authoritative DNS server responds in NXDOMAIN and NODATA scenarios when DNSSEC is enabled.

Traffic Router currently supports DNSSEC with NSEC, however, NSEC3 and more configurable options will be provided in the future.

## Operation

Upon startup or a configuration change, Traffic Router obtains keys from the keystore API in Traffic Ops which returns key signing keys (KSK) and zone signing keys (ZSK) for each delivery service that is a subdomain off the CDN's top level domain (TLD), in addition to the keys for the CDN TLD itself. Each key has timing information that allows Traffic Router to determine key validity (expiration, inception, and effective dates) in addition to the appropriate TTL to use for the DNSKEY record(s). All TTLs are configurable parameters; see the [Traffic Router Profile](#) documentation for more information.

Once Traffic Router obtains the key data from the API, it converts each public key into the appropriate record types (DNSKEY, DS) to place in zones and uses the private key to sign zones. DNSKEY records are added to each delivery service's zone (e.g.: mydeliveryservice.cdn.kabletown.net) for every valid key that exists, in addition to the CDN TLD's zone. A DS record is generated from each zone's KSK and is placed in the CDN TLD's zone (e.g.: cdn.kabletown.net); the DS record for the CDN TLD must be placed in its parent zone, which is not managed by Traffic Control.

The DNSKEY to DS record relationship allows resolvers to validate signatures across zone delegation points; with Traffic Control, we control all delegation points below the CDN's TLD, **however, the DS record for the CDN TLD must be placed in the parent zone (e.g.: kabletown.net), which is not managed by Traffic Control.** As such, the DS record (available in the Traffic Ops DNSSEC administration UI) must be placed in the parent zone prior to enabling DNSSEC, and prior to generating a new CDN KSK. Based on your deployment's DNS configuration, this might be a manual process or it might be automated; either way, extreme care and diligence must be taken and knowledge of the management of the upstream zone is imperative for a successful DNSSEC deployment.

## Rolling Zone Signing Keys

Traffic Router currently follows the zone signing key pre-publishing operational best practice described in [section 4.1.1.1 of RFC 6781](#). Once DNSSEC is enabled for a CDN in Traffic Ops, key rolls are triggered via Traffic Ops via the automated key generation process, and Traffic Router selects the active zone signing keys based on the expiration information returned from the keystore API in Traffic Ops.

## Troubleshooting and log files

Traffic Router log files are in `/opt/traffic_router/var/log`, and Tomcat log files are in `/opt/tomcat/logs`. Application related logging is in `/opt/traffic_router/var/log/traffic_router.log`, while access logs are written to `/opt/traffic_router/var/log/access.log`.

## Event Log File Format



## Summary

All access events to Traffic Router are logged to the file `/opt/traffic_router/var/log/access.log`. This file grows up to 200Mb and gets rolled into older log files, 10 log files total are kept (total of up to 2Gb of logged events per traffic router).

Traffic Router logs access events in a format that largely following [ATS event logging format](#).

## Sample Message

Items within brackets below are detailed under the HTTP and DNS sections

```
144140678.000 qtype=DNS chi=192.168.10.11 ttms=789 [Fields Specific to the DNS
↳request] rtype=CZ rloc="40.252611,58.439389" rdtl=- rerr="-" [Fields Specific to
↳the DNS result]
144140678.000 qtype=HTTP chi=192.168.10.11 ttms=789 [Fields Specific to the HTTP
↳request] rtype=GEO rloc="40.252611,58.439389" rdtl=- rerr="-" [Fields Specific to
↳the HTTP result]
```

**Note:** The above message samples contain fields that are always present for every single access event to Traffic Router.

**Message Format** - Each event that is logged is a series of space separated key value pairs except for the first item. - The first item is always the epoch in seconds with a decimal field precision of up to milliseconds - Each key value pair is in the form of unquoted string, equals character, optionally quoted string - Values that are quoted strings may contain space characters - Values that are not quoted should not contains any space characters

**Note:** Any value that is a single dash character or a dash character enclosed in quotes represents an empty value

## Fields Always Present

Name	Description	Data
qtype	Whether the request was for DNS or HTTP	Always DNS or HTTP
chi	The IP address of the requester	Depends on whether this was a DNS or HTTP request, see below sections
ttms	The amount of time in milliseconds it took Traffic Router to process the request	A number greater than or equal to zero
rtype	Routing Result Type	One of ERROR, CZ, DEEP_CZ, GEO, MISS, STATIC_ROUTE, DS_REDIRECT, DS_MISS, INIT, FED
rloc	GeoLocation of result	Latitude and Longitude in Decimal Degrees
rdtl	Result Details Associated with unusual conditions	One of DS_NOT_FOUND, DS_NO_BYPASS, DS_BYPASS, DS_CZ_ONLY
rerr	Message about internal Traffic Router Error	String

**rtype meanings**

Name	Meaning
ERROR	An internal error occurred within Traffic Router, more details may be found in the rerr field
CZ	The result was derived from Coverage Zone data based on the address in the chi field
DEEP_CZ	The result was derived from Deep Coverage Zone data based on the address in the chi field
GEO	The result was derived from geolocation service based on the address in the chi field
MISS	Traffic Router was unable to resolve a DNS request or find a cache for the requested resource
STATIC_ROUTE	_DNS Only*_ No DNS Delivery Service supports the hostname portion of the requested url
DS_MISS	_HTTP Only*_ No HTTP Delivery Service supports either this request's URL path or headers
DS_REDIRECT	The result is using the Bypass Destination configured for the matched Delivery Service when that Delivery Service is unavailable or does not have the requested resource
FED	_DNS Only*_ The result was obtained through federated coverage zone data outside of any delivery service

### rdtl meanings

Name	Meaning
DS_NOT_FOUND	Always goes with rtypes STATIC_ROUTE and DS_MISS
DS_BYPASS	Used Bypass Destination for Redirect of Delivery Service
DS_NO_BYPASS	No valid Bypass Destination is configured for the matched Delivery Service and the delivery service does not have the requested resource
DS_CZ_ONLY	The selected Delivery Service only supports resource lookup based on Coverage Zone data
DS_CLIENT_GEO_UNSUPPORTED	Traffic Router did not find a resource supported by coverage zone data and was unable to determine the geolocation of the requesting client
GEO_NO_CACHE_FOUND	Traffic Router could not find a resource via geolocation data based on the requesting client's geolocation

## HTTP Specifics

### Sample Message

```
1452197640.936 qtype=HTTP chi=69.241.53.218 url="http://foo.mm-test.jenkins.cdnlab.
↪comcast.net/some/asset.m3u8" cqhm=GET cqhv=HTTP/1.1 rtype=GEO rloc="40.252611,58.
↪439389" rdtl=- rerr="-" pssc=302 ttms=0 rurl="http://odol-atsec-sim-114.mm-test.
↪jenkins.cdnlab.comcast.net:8090/some/asset.m3u8" rh="Accept: */*" rh="myheader:
↪asdasdasdasfasg"
```

### Request Fields

Name	Description	Data
url	Requested URL with query string	String
cqhm	Http Method	e.g GET, POST
cqhvh	Http Protocol Version	e.g. HTTP/1.1
rh	One or more of these key value pairs may exist in a logged event and are controlled by the configuration of the matched Delivery Service	Key value pair of the format "name: value"

### Response Fields

Name	Description	Data
rurl	The resulting url of the resource requested by the client	A URL String

## DNS Specifics

### Sample Message

```
144140678.000 qtype=DNS chi=192.168.10.11 ttms=123 xn=65535 fqdn=www.example.com.
↪type=A class=IN ttl=12345 rcode=NOERROR rtype=CZ rloc="40.252611,58.439389" rdt1=-
↪rerr="-" ans="192.168.1.2 192.168.3.4 0:0:0:0:0:0:ffff:c0a8:102
↪0:0:0:0:0:0:ffff:c0a8:304"
```

### Request Fields

Name	Description	Data
xn	The ID from the client DNS request header	a number from 0 to 65535
fqdn	The qname field from the client DNS request message (i.e. The fully qualified domain name the client is requesting be resolved)	A series of DNS labels/domains separated by ‘.’ characters and ending with a ‘.’ character (see <a href="#">qname</a> )
type	The qtype field from the client DNS request message (i.e. the type of resolution that’s requested such as IPv4, IPv6)	<b>Examples are A (IPv4), AAAA (IPv6), NS (Name Server), SOA (Start of Authority), and CNAME, (see <a href="#">qtype</a>)</b>
class	The qclass field from the client DNS request message (i.e. The class of resource being requested)	<b>Either IN (Internet resource) or ANY (Traffic router rejects requests with any other value of class)</b>

### Response Fields

Name	Description	Data
ttl	The ‘time to live’ in seconds for the answer provided by Traffic Router (clients can reliably use this answer for this long without re-querying traffic router)	A number from 0 to 4294967295
rcode	The result code for the DNS answer provided by Traffic Router	One of NOERROR (success), NOTIMP (request is not supported), REFUSED (request is refused to be answered), or NXDOMAIN (the domain/name requested does not exist)

## GeoLimit Failure Redirect feature

### Overview

This feature is also called ‘National GeoBlock’ feature which is short for ‘NGB’ feature. In this section, the acronym ‘NGB’ will be used for this feature.

In the past, if the Geolimit check fails (for example, the client ip is not in the ‘US’ region but the geolimit is set to ‘CZF + US’), the router will return 503 response; but with this feature, when the check fails, it will return 302 if the redirect url is set in the delivery service.

The Geolimit check failure has such scenarios: 1) When the GeoLimit is set to ‘CZF + only’, if the client ip is not in the the CZ file, the check fails 2) When the GeoLimit is set to any region, like ‘CZF + US’, if the client ip is not in such region, and the client ip is not in the CZ file, the check fails

### Configuration

To enable the NGB feature, the DS must be configured with the proper redirect url. And the setting lays at ‘Delivery Services’->Edit->‘GeoLimit Redirect URL’. If no url is put in this field, the feature is disabled.

The URL has 3 kinds of formats, which have different meanings:

1. URL with no domain. If no domain is in the URL (like ‘vod/dance.mp4’), the router will try to find a proper cache server within the delivery service and return the redirect url with the format like ‘<http://<cache server name>.<delivery service's FQDN>/<configured relative path>>’
2. URL with domain that matches with the delivery service. For this URL, the router will also try to find a proper cache server within the delivery service and return the same format url as point 1.
3. URL with domain that doesn’t match with the delivery service. For this URL, the router will return the configured url directly to the client.

### Deep Caching - Deep Coverage Zone Topology

#### Overview

Deep Caching is a feature that enables clients to be routed to the closest possible “deep” edge caches on a per Delivery Service basis. The term “deep” is used in the networking sense, meaning that the edge caches are located deep in the network where the number of network hops to a client is as minimal as possible. This deep caching topology is desirable because storing content closer to the client gives better bandwidth savings, and sometimes the cost of bandwidth usage in the network outweighs the cost of adding storage. While it may not be feasible to cache an entire copy of the CDN’s contents in every deep location (for the best possible bandwidth savings), storing just a relatively small amount of the CDN’s most requested content can lead to very high bandwidth savings.

#### Getting started

What you need:

1. Edge caches deployed in “deep” locations and registered in Traffic Ops
2. A Deep Coverage Zone File (DCZF) mapping these deep cache hostnames to specific network prefixes (see *The Deep Coverage Zone File* for details)
3. Deep caching parameters in the Traffic Router Profile (see *Traffic Router Profile* for details):
  - `deepcoveragezone.polling.interval`
  - `deepcoveragezone.polling.url`
4. Deep Caching enabled on one or more HTTP Delivery Services (i.e. `deepCachingType = ALWAYS`)

## How it works

Deep Coverage Zone routing is very similar to that of regular Coverage Zone routing, except that the DCZF is preferred over the regular CZF for Delivery Services with DC (Deep Caching) enabled. If the client requests a DC-enabled Delivery Service and their IP address gets a “hit” in the DCZF, Traffic Router will attempt to route that client to one of the available deep caches in the client’s corresponding zone. If there are no deep caches available for a client’s request, Traffic Router will “fall back” to the regular CZF and continue regular CZF routing from there.

## Steering feature

### Overview

A Steering delivery service is a delivery service that is used to “steer” traffic to other delivery services. A Steering delivery service will have target delivery services configured for it with weights assigned to them. Traffic Router uses the weights to make a consistent hash ring which it then uses to make sure that requests are routed to a target based on the configured weights. This consistent hash ring is separate from the consistent hash ring used in cache selection.

Special regular expressions called Filters can also be configured for target delivery services to pin traffic to a specific delivery service. For example, if a filter called `.*news/.*` for a target called `target-ds-1` is created, any requests to traffic router with ‘news’ in them will be routed to `target-ds-1`. This will happen regardless of the configured weights.

A client can bypass the steering functionality by providing a header called `X-TC-Steering-Option` with the `xml_id` of the target delivery service to route to. When Traffic Router receives this header it will route to the requested target delivery service regardless of weight configuration.

Some other points of interest:

- Steering is currently only available for HTTP delivery services that are a part of the same CDN.
- A new role called `STEERING` has been added to the traffic ops database. Only users with Admin or Steering privileges can modify steering assignments for a Delivery Service.
- A new API has been created in Traffic Ops under `/internal`. This API is used by a Steering user to add filters and modify assignments. (Filters can only be added via the API).
- Traffic Router uses the steering API in Traffic Ops to poll for steering assignments, the assignments are then used when routing traffic.

A couple simple use cases for steering are:

1. Migrating traffic from one delivery service to another over time.
2. Trying out new functionality for a subset of traffic with an experimental delivery service.
3. Load balancing between delivery services.

## Configuration

The following needs to be completed for Steering to work correctly:

1. Two target delivery services are created in Traffic Ops. They must both be HTTP delivery services part of the same CDN.
2. A delivery service with type `STEERING` is created in Traffic Ops.
3. Target delivery services are assigned to the steering delivery service using Traffic Ops.
4. A user with the role of Steering is created.
5. Using the API, the steering user assigns weights to the target delivery services.

6. If desired, the steering user can create filters for the target delivery services.

For more information see the [steering how-to guide](#).

## HTTPS for Http Type Delivery Services

Starting with version 1.7 Traffic Router added the ability to allow https traffic between itself and clients on a per http type delivery service basis.

**Warning:** The establishing of an HTTPS connection is much more computationally demanding than an HTTP connection. Since each client will in turn get redirected to ATS, Traffic Router is most always creating a new HTTPS connection for all HTTPS traffic. It is likely to mean that an existing Traffic Router will have some decrease in performance depending on the amount of https traffic you want to support. As noted for DNSSEC, you may need to plan to scale Traffic Router vertically and/or horizontally to handle the new load.

The summary for setting up https is to:

1. Select one of 'https', 'http and https', or 'http to https' for the delivery service
2. Generate private keys for the delivery service using a wildcard domain such as `*.my-delivery-service.my-cdn.example.com`
3. Obtain and import signed certificate chain
4. Snapshot CR Config

Clients may make HTTPS requests delivery services only after Traffic Router receives the certificate chain from Traffic Ops and the new CR Config.

## Protocol Options

**https only** Traffic Router will only redirect (send a 302) to clients communicating with a secure connection, all other clients will receive a 503

**http and https** Traffic Router will redirect both secure and non-secure clients

**http to https** Traffic Router will redirect non-secure clients with a 302 and a location that is secure (i.e. starting with 'https' instead of 'http'), secure clients will remain on https

**http** Any secure client will get an SSL handshake error. Non-secure clients will experience the same behavior as prior to 1.7

## Certificate Retrieval

**Warning:** If you have https delivery services in your CDN, Traffic Router will not accept **any** connections until it is able to fetch certificates from Traffic Ops and load them into memory. Traffic Router does not persist certificates to the java keystore or anywhere else.

Traffic Router fetches certificates into memory:

- At startup time
- When it receives a new CR Config

- Once an hour from whenever the most recent of the last of the above occurred

---

**Note:** To adjust the frequency when Traffic Router fetches certificates add the parameter ‘certificates.polling.interval’ to CR Config and setting it to the desired time in milliseconds.

---

---

**Note:** Taking a snapshot of CR Config may be used at times to avoid waiting the entire polling cycle for a new set of certificates.

---

**Warning:** If a snapshot of CR Config is made that involves a delivery service missing its certificates, Traffic Router will ignore **ALL** changes in that CR-Config until one of the following occurs: \* It receives certificates for that delivery service \* Another snapshot of CR Config is created and the delivery service without certificates is changed so it’s HTTP protocol is set to ‘http’

## Certificate Chain Ordering

The ordering of certificates within the certificate bundle matters. It must be:

1. Primary Certificate (e.g. the one created for \*.my-delivery-service.my-cdn.example.com)
2. Intermediate Certificate(s)
3. Root Certificate from CA (optional)

**Warning:** If something is wrong with the certificate chain (e.g. the order of the certificates is backwards or for the wrong domain) the client will get an SSL handshake. Inspection of /opt/tomcat/logs/catalina.out is likely to yield information to reveal this.

To see the ordering of certificates you may have to manually split up your certificate chain and use openssl on each individual certificate

## Suggested Way of Setting up an HTTPS Delivery Service

Do the following in Traffic Ops:

1. Select one of ‘https’, ‘http and https’, or ‘http to https’ for the protocol field of a delivery service and click ‘Save’.
2. Click ‘Manage SSL Keys’.
3. Click ‘Generate New Keys’.
4. Copy the contents of the Certificate Signing Request field and save it locally.
5. Click ‘Load Keys’.
6. Select ‘http’ for the protocol field of the delivery service and click ‘Save’ (to avoid preventing other CR Config updates from being blocked by Traffic Router)
7. Follow your standard procedure for obtaining your signed certificate chain from a CA.
8. After receiving your certificate chain import it into Traffic Ops.

9. Edit the delivery service.
10. Restore your original choice for the protocol field and click save.
11. Click 'Manage SSL Keys'.
12. Click 'Paste Existing Keys'.
13. Paste the certificate chain into the CRT field.
14. Click 'Load Keys'.
15. Take a new snapshot of CR Config.

Once this is done you should be able to test you are getting correctly redirected by Traffic Router using curl commands to https destinations on your delivery service.

A new testing tool was created for load testing traffic router, it allows you to generate requests from your local box to multiple delivery services of a single cdn. You can control which cdn, delivery services, how many transactions per delivery service, and how many concurrent requests. During the test it will provide feedback about request latency and transactions per second.

While it is running it is suggested that you monitor your Traffic Router nodes for memory and CPU utilization.

## Tuning Recommendations

The following is an example of `/opt/tomcat/bin/setenv.sh` that has been tested on a multi core server running under HTTPS load test requests. This is following the general recommendation to use the G1 garbage collector for JVM applications running on multi core machines. In addition to using the G1 garbage collector the `InitiatingHeapOccupancyPercent` was lowered to run garbage collection more frequently which improved overall throughput for Traffic Router and reduced 'Stop the World' garbage collection. Note that setting the min and max heap settings in `setenv.sh` will override init scripts in `/etc/init.d/tomcat`.

`/opt/tomcat/bin/setenv.sh:`

```
#!/bin/sh
export CATALINA_OPTS="$CATALINA_OPTS -server"
export CATALINA_OPTS="$CATALINA_OPTS -Xms2g -Xmx2g"
export CATALINA_OPTS="$CATALINA_OPTS -XX:+UseG1GC"
export CATALINA_OPTS="$CATALINA_OPTS -XX:+UnlockExperimentalVMOptions"
export CATALINA_OPTS="$CATALINA_OPTS -XX:InitiatingHeapOccupancyPercent=30"
```

### 3.1.13 Traffic Stats Administration

Traffic Stats consists of three separate components: Traffic Stats, InfluxDB, and Grafana. See below for information on installing and configuring each component as well as configuring the integration between the three and Traffic Ops.

## Installation

### Installing Traffic Stats:

- See the [downloads](#) page for Traffic Control to get the latest release.
- Follow our build [intructions](#) to generate an RPM.
- Copy the RPM to your server
- perform the following command: `sudo rpm -ivh <traffic_stats rpm>`



### Installing InfluxDB:

**As of Traffic Stats 1.8.0, InfluxDb 1.0.0 or higher is required. For InfluxDb versions less than 1.0.0 use Traffic Stats 1.7.x**

In order to store traffic stats data you will need to install [InfluxDB](#). While not required, it is recommended to use some sort of high availability option like [Influx enterprise](#), [Influxdb Relay](#), or another [high availability option](#).

### Installing Grafana:

Grafana is used to display Traffic Stats/InfluxDB data in Traffic Ops. Grafana is typically run on the same server as Traffic Stats but this is not a requirement. Grafana can be installed on any server that can access InfluxDB and can be accessed by Traffic Ops. Documentation on installing Grafana can be found on the [Grafana website](#).

## Configuration

### Configuring Traffic Stats:

Traffic Stats' configuration file can be found in `/opt/traffic_stats/conf/traffic_stats.cfg`. The following values need to be configured:

- *toUser*: The user used to connect to Traffic Ops
- *toPasswd*: The password to use when connecting to Traffic Ops
- *toUrl*: The URL of the Traffic Ops server used by Traffic Stats
- *influxUser*: The user to use when connecting to InfluxDB (if configured on InfluxDB, else leave default)
- *influxPassword*: That password to use when connecting to InfluxDB (if configured, else leave blank)
- *pollingInterval*: The interval at which Traffic Monitor is polled and stats are stored in InfluxDB
- *statusToMon*: The status of Traffic Monitor to poll (poll ONLINE or OFFLINE traffic monitors)
- *seelogConfig*: The absolute path of the seelog config file
- *dailySummaryPollingInterval*: The interval, in seconds, at which Traffic Stats checks to see if daily stats need to be computed and stored.
- *cacheRetentionPolicy*: The default retention policy for cache stats
- *dsRetentionPolicy*: The default retention policy for deliveryservice stats
- *dailySummaryRetentionPolicy*: The retention policy to be used for the daily stats
- *influxUrls*: An array of influxdb hosts for Traffic Stats to write stats to.

### Configuring InfluxDB:

As mentioned above, it is recommended that InfluxDb be running in some sort of high availability configuration. There are several ways to achieve high availability so it is best to consult the high availability options on the [InfluxDB website](#).

Once InfluxDB is installed and configured, databases and retention policies need to be created. Traffic Stats writes to three different databases: `cache_stats`, `deliveryservice_stats`, and `daily_stats`. More information about the databases and what data is stored in each can be found on the [overview](#) page.

To easily create databases, retention policies, and continuous queries, run `create_ts_databases` from the `/opt/traffic_stats/influxdb_tools` directory on your Traffic Stats server. See the [InfluxDb Tools](#) section below for more information.

### Configuring Grafana:

In Traffic Ops the Health -> Graph View tab can be configured to display grafana graphs using influxDb data. In order for this to work correctly, you will need two things 1) a parameter added to traffic ops with the graph URL (more information below) and 2) the graphs created in grafana. See below for how to create some simple graphs in grafana. These instructions assume that InfluxDB has been configured and that data has been written to it. If this is not true, you will not see any graphs.

- Login to grafana as an admin user [http://grafana\\_url:3000/login](http://grafana_url:3000/login)
- Choose Data Sources and then Add New
- Enter the necessary information to configure your data source
- Click on the 'Home' dropdown at the top of the screen and choose New at the bottom
- Click on the green menu bar (with 3 lines) at the top and choose Add Panel -> Graph
- Where it says 'No Title (click here)' click and choose edit
- Choose your data source at the bottom
- You can have grafana help you create a query, or you can create your own. Here is a sample query:

```
SELECT sum(value)*1000 FROM "monthly"."bandwidth.cdn.
1min" WHERE $timeFilter GROUP BY time(60s), cdn
```

- Once you have the graph the way you want it, click the 'Save Dashboard' button at the top
- You should now have a new saved graph

In order for Traffic Ops users to see Grafana graphs, Grafana will need to allow anonymous access. Information on how to configure anonymous access can be found on the configuration page of the [Grafana Website](#).

Traffic Ops uses custom dashboards to display information about individual delivery services or cache groups. In order for the custom graphs to display correctly, the `traffic_ops_*.js` files need to be in the `/usr/share/grafana/public/dashboards/` directory on the grafana server. If your Grafana server is the same as your Traffic Stats server the RPM install process will take care of putting the files in place. If your grafana server is different from your Traffic Stats server, you will need to manually copy the files to the correct directory.

More information on custom scripted graphs can be found in the [scripted dashboards](#) section of the Grafana documentation.

### Configuring Traffic Ops for Traffic Stats:

- The influxDb servers need to be added to Traffic Ops with profile = InfluxDB. Make sure to use port 8086 in the configuration.
- The traffic stats server should be added to Traffic Ops with profile = Traffic Stats.
- Parameters for which stats will be collected are added with the release, but any changes can be made via parameters that are assigned to the Traffic Stats profile.

### Configuring Traffic Ops to use Grafana Dashboards

To configure Traffic Ops to use Grafana Dashboards, you need to enter the following parameters and assign them to the GLOBAL profile. This assumes you followed the above instructions to install and configure InfluxDB and Grafana. You will need to place 'cdn-stats', 'deliveryservice-stats', and 'daily-summary' with the name of your dashboards.

parameter name	parameter value
all_graph_url	<a href="https://&lt;grafana_url&gt;/dashboard/db/deliveryservice-stats">https://&lt;grafana_url&gt;/dashboard/db/deliveryservice-stats</a>
cachegroup_graph_url	<a href="https://&lt;grafanaHost&gt;/dashboard/script/traffic_ops_cachegroup.js?which=">https://&lt;grafanaHost&gt;/dashboard/script/traffic_ops_cachegroup.js?which=</a>
deliveryservice_graph_url	<a href="https://&lt;grafanaHost&gt;/dashboard/script/traffic_ops_devdeliveryservice.js?which=">https://&lt;grafanaHost&gt;/dashboard/script/traffic_ops_devdeliveryservice.js?which=</a>
server_graph_url	<a href="https://&lt;grafanaHost&gt;/dashboard/script/traffic_ops_server.js?which=">https://&lt;grafanaHost&gt;/dashboard/script/traffic_ops_server.js?which=</a>
visual_status_panel_1	<a href="https://&lt;grafanaHost&gt;/dashboard-solo/db/cdn-stats?panelId=2&amp;fullscreen&amp;from=now-24h&amp;to=now-60s">https://&lt;grafanaHost&gt;/dashboard-solo/db/cdn-stats?panelId=2&amp;fullscreen&amp;from=now-24h&amp;to=now-60s</a>
visual_status_panel_2	<a href="https://&lt;grafanaHost&gt;/dashboard-solo/db/cdn-stats?panelId=1&amp;fullscreen&amp;from=now-24h&amp;to=now-60s">https://&lt;grafanaHost&gt;/dashboard-solo/db/cdn-stats?panelId=1&amp;fullscreen&amp;from=now-24h&amp;to=now-60s</a>
daily_bw_url	<a href="https://&lt;grafanaHost&gt;/dashboard-solo/db/daily-summary?panelId=1&amp;fullscreen&amp;from=now-3y&amp;to=now">https://&lt;grafanaHost&gt;/dashboard-solo/db/daily-summary?panelId=1&amp;fullscreen&amp;from=now-3y&amp;to=now</a>
daily_served_url	<a href="https://&lt;grafanaHost&gt;/dashboard-solo/db/daily-summary?panelId=2&amp;fullscreen&amp;from=now-3y&amp;to=now">https://&lt;grafanaHost&gt;/dashboard-solo/db/daily-summary?panelId=2&amp;fullscreen&amp;from=now-3y&amp;to=now</a>

## InfluxDb Tools

Under the Traffic Stats source directory there is a directory called `influxdb_tools`. These tools are meant to be used as one-off scripts to help a user quickly get new databases and continuous queries setup in influxdb. They are specific for traffic stats and are not meant to be generic to influxdb. Below is a brief description of each script along with how to use it.

**create/create\_ts\_databases.go** This script creates all [databases](#), [retention policies](#), and [continuous queries](#) required by traffic stats.

### How to use create\_ts\_databases:

Pre-Requisites:

1. Go 1.7 or later
2. configured \$GOPATH (e.g. export GOPATH=~/.go)

Using create\_ts\_databases.go

1. go to the `traffic_stats/influxdb_tools/create` directory
2. build it by running `go build create_ts_databases.go` or simply `go build`
3. **Run it:**

- `./create_ts_databases -help` or `./create -help`
- **optional flags:**
  - `url` - The influxdb url and port
  - `replication` - The number of nodes in the cluster
  - `user` - The user to use
  - `password` - The password to use
- **example:** `./create_ts_databases -url=localhost:8086 -replication=3 -user=joe -password=mysecret` or `./create -url=localhost:8086 -replication=3 -user=joe -password=mysecret`

**sync\_ts\_databases** This script is used to sync one influxdb environment to another. Only data from continuous queries is synced as it is downsampled data and much smaller in size than syncing raw data. Possible use cases are syncing from Production to Development or Syncing a new cluster once brought online.

### How to use sync\_ts\_databases:

#### Pre-Requisites:

1. Go 1.7 or later
2. configured \$GOPATH (e.g. export GOPATH=~/go)

#### Using sync\_ts\_databases.go:

1. go to the traffic\_stats/influxdb\_tools/create directory
2. build it by running `go build sync_ts_databases.go` or simply `go build`
3. **Run it**

- `./sync_ts_databases -help` or `./sync -help`

#### • required flags:

- `source-url` - The URL of the source database
- `target-url` - The URL of the target database

#### -optional flags:

- `database` - The database to sync (default = sync all databases)
- `days` - Days in the past to sync (default = sync all data)
- `source-user` - The user of the source database
- `source-pass` - The password for the source database
- `target-user` - The user of the target database
- `target-pass` - The password for the target database
- example: `./sync -source-url=http://idb-01.foo.net:8086 -target-url=http://idb-01.foo.net:8086 -database=cache_stats -days=7 -source-user=admin source-pass=mysecret`

## 3.1.14 Traffic Server Administration

### Installing Traffic Server

1. Build the Traffic Server RPM. The best way to do this is to follow the Traffic Server documents:

```
https://docs.trafficserver.apache.org/en/latest/getting-started/index.en.html  
↪ #installation
```

2. Build the astats RPM using the appropriate version number:

```
https://github.com/apache/incubator-trafficcontrol/tree/<version>/traffic_server
```

Sample link:

```
https://github.com/apache/incubator-trafficcontrol/tree/master/traffic_server
```

3. Install Traffic Server and astats:

```
sudo yum -y install trafficserver-*.rpm astats_over_http*.rpm
```

4. Add the server using the Traffic Ops web interface:

1. Select **Servers**.
2. Scroll to the bottom of the page and click **Add Server**.
3. Complete the “Required Info:” section:
  - Set ‘Interface Name’ to the name of the interface from which traffic server delivers content.
  - Set ‘Type’ to ‘MID’ or ‘EDGE’.
4. Click **Submit**.
5. Click **Save**.
6. Click **Online Server**.
7. Verify that the server status is now listed as **Reported**
5. Install the ORT script and run it in ‘badass’ mode to create the initial configuration, see [Configuring Traffic Server](#)
6. Start the service: `sudo service trafficserver start`
7. Configure traffic server to start automatically: `sudo systemctl enable trafficserver`
8. Verify that the installation is good:
  1. Make sure that the service is running: `sudo systemctl status trafficserver`
  2. Assuming a traffic monitor is already installed, browse to it, i.e. <http://<trafficmonitorURL>>, and verify that the traffic server appears in the “Cache States” table, in white.

## Configuring Traffic Server

All of the Traffic Server application configuration files are generated by Traffic Ops and installed by way of the `traffic_ops_ort.pl` script. The `traffic_ops_ort.pl` should be installed on all caches (by puppet or other non Traffic Ops means), usually in `/opt/ort`. It is used to do the initial install of the config files when the cache is being deployed, and to keep the config files up to date when the cache is already in service. The usage message of the script is shown below:

```
$ sudo /opt/ort/traffic_ops_ort.pl
=====
Usage: ./traffic_ops_ort.pl <Mode> <Log_Level> <Traffic_Ops_URL> <Traffic_Ops_Login>
↳[optional flags]
  <Mode> = interactive - asks questions during config process.
  <Mode> = report - prints config differences and exits.
  <Mode> = badass - attempts to fix all config differences that it can.
  <Mode> = syncds - syncs delivery services with what is configured in Traffic Ops.
  <Mode> = revalidate - checks for updated revalidations in Traffic Ops and applies
↳them. Requires Traffic Ops 2.1.

  <Log_Level> => ALL, TRACE, DEBUG, INFO, WARN, ERROR, FATAL, NONE

  <Traffic_Ops_URL> = URL to Traffic Ops host. Example: https://trafficops.company.net

  <Traffic_Ops_Login> => Example: 'username:password'

[optional flags]:
  dispersion=<time>      => wait a random number between 0 and <time> before
↳starting. Default = 300.
  login_dispersions=<time> => wait a random number between 0 and <time> before
↳login. Default = 0.
  retries=<number>      => retry connection to Traffic Ops URL <number> times.
↳Default = 3.
(continues on next page)
```

(continued from previous page)

```
wait_for_parents=<0|1> => do not update if parent_pending = 1 in the update json.
↪Default = 1, wait for parents.
=====
$
```

## Installing the ORT script

1. Build the ORT script RPM from the Apache Build Server and install it:

```
https://builds.apache.org/view/S-Z/view/TrafficControl/
```

Sample command:

```
sudo wget https://builds.apache.org/view/S-Z/view/TrafficControl/job/incubator-
↪trafficcontrol-2.1.x-build/lastSuccessfulBuild/artifact/dist/traffic_ops_ort-2.
↪1.0-6807.1dcd512f.el7.x86_64.rpm
sudo yum install traffic_ops_ort*.rpm
```

2. Install modules required by ORT if needed: `sudo yum -y install perl-JSON perl-Crypt-SSLeay`
3. For initial configuration or when major changes (like a Profile change) need to be made, run the script in “badass mode”. All required rpm packages will be installed, all Traffic Server config files will be fetched and installed, and (if needed) the Traffic Server application will be restarted.

Example run below:

```
$ sudo /opt/ort/traffic_ops_ort.pl --dispersion=0 badass warn https://ops.
↪$tcDomain admin:admin123
```

**Note:** First run gives a lot of state errors that are expected. The “badass” mode fixes these issues. Run it a second time, this should be cleaner. Also, note that many ERROR messages emitted by ORT are actually information messages. Do not panic.

4. Create a cron entry for running ort in ‘syncds’ mode every 15 minutes. This makes traffic control check periodically if ‘Queue Updates’ was run on Traffic Ops, and if so, get the updated configuration.

Run `sudo crontab -e` and add the following line

```
*/15 * * * * /opt/ort/traffic_ops_ort.pl syncds warn https://traffops.kabletown.
↪net admin:password --login_dispersion=30 --dispersion=180 > /tmp/ort/syncds.log
↪2>&1
```

Changing `https://traffops.kabletown.net`, `admin`, and `password` to your CDN URL and credentials.

**Note:** By default, running ort on an edge traffic server waits for its parent (mid) servers to download their configuration before it downloads its own configuration. Because of this, scheduling ort for running every 15 minutes (with 5 minutes default dispersion) means that it might take up to ~35 minutes for a “Queue Updates” operation to affect all traffic servers. To customize this dispersion time, use the command line option `-dispersion=x` where `x` is the number of seconds for the dispersion period. Servers will select a random number from within this dispersion period to begin pulling down configuration files from Traffic Ops. Another option, `-login_dispersion=x` can be used. This option creates a dispersion period after the job begins during which

ORT will wait before logging in and checking Traffic Ops for updates to the server. This defaults to 0. If `use_reval_pending`, a.k.a. Rapid Revalidate is enabled, edges will NOT wait for their parents to download their configuration before downloading their own.

---

**Note:** In ‘syncds’ mode, the `ort` script updates only configurations that might be changed as part of normal operations, such as:

- Delivery Services
  - SSL certificates
  - Traffic Monitor IP addresses
  - Logging configuration
  - Revalidation requests (By default. If Rapid Revalidate is enabled, this will only be checked by using a separate `revalidate` command in ORT.)
- 

5. If Rapid Revalidate is enabled in Traffic Ops, create a second cron job for revalidation checks. ORT will not check revalidation files if Rapid Revalidate is enabled. This setting allows for a separate check to be performed every 60 seconds to verify if a revalidation update has been made.

Run `sudo crontab -e` and add the following line

```
*/1 * * * * /opt/ort/traffic_ops_ort.pl revalidate warn https://traffops.  
↪kabletown.net admin:password --login_dispersion=30 > /tmp/ort/syncds.log 2>&1
```

## 3.1.15 Traffic Vault Administration

### Installing Traffic Vault

In order to successfully store private keys you will need to install Riak. The latest version of Riak can be downloaded on the Riak [website](#). The installation instructions for Riak can be found [here](#).

Production is currently running version 2.0.5 of Riak, but the latest version should suffice.

### Configuring Traffic Vault

The following steps were taken to configure Riak in our environments.

#### Riak configuration file configuration

The following steps need to be performed on each Riak server in the cluster:

- Log into riak server as root
- `cd` to `/etc/riak/`
- **Update the following in `riak.conf` to reflect your IP:**
  - `nodename` = `riak@a-host.sys.kabletown.net`
  - `listener.http.internal` = `a-host.sys.kabletown.net:8098` (can be 80 - This endpoint will not work with sec enabled)
  - `listener.protobuf.internal` = `a-host.sys.kabletown.net:8087` (can be different port if you want)

- listener.https.internal = a-host.sys.kabletown.net:8088 (can be 443)
- **Updated the following conf file to point to your cert files**
  - ssl.certfile = /etc/riak/certs/server.crt
  - ssl.keyfile = /etc/riak/certs/server.key
  - ssl.cacertfile = /etc/pki/tls/certs/ca-bundle.crt
- **Add a line at the bottom of the config for tlsv1**
  - tls\_protocols.tlsv1 = on
- **Once the config file has been updated restart riak**
  - /etc/init.d/riak restart
- **Validate server is running by going to the following URL:**
  - <https://<serverHostname>:8088/ping>

## Riak-admin configuration

Riak-admin is a command line utility that needs to be run as root on a server in the riak cluster.

### Assumptions:

- Riak 2.0.2 or greater is installed
- SSL Certificates have been generated (signed or self-signed)
- Root access to riak servers

### Add admin user and riakuser to riak

- Admin user will be a super user
- Riakuser will be the application user

Login to one of the riak servers in the cluster as root (any will do)

1. Enable security

```
riak-admin security enable
```

2. Add groups

```
riak-admin security add-group admins
```

```
riak-admin security add-group keysusers
```

3. Add users

---

**Note:** username and password should be stored in /opt/traffic\_ops/app/conf/<environment>/riak.conf

---

```
riak-admin security add-user admin password=<AdminPassword>  
groups=admins
```

```
riak-admin security add-user riakuser  
password=<RiakUserPassword> groups=keysusers
```

4. Grant access for admin and riakuser



```
riak-admin security add-source riakuser 0.0.0.0/0 password
riak-admin security add-source admin 0.0.0.0/0 password
```

#### 5. Grant privs to admins for everything

```
riak-admin security grant riak_kv.list_buckets,riak_kv.
list_keys,riak_kv.get,riak_kv.put,riak_kv.delete on any to
admins
```

#### 6. Grant privs to keysuser for ssl, dnssec, and url\_sig\_keys buckets only

```
riak-admin security grant riak_kv.get,riak_kv.put,riak_kv.
delete on default ssl to keysusers

riak-admin security grant riak_kv.get,riak_kv.put,riak_kv.
delete on default dnssec to keysusers

riak-admin security grant riak_kv.get,riak_kv.put,riak_kv.
delete on default url_sig_keys to keysusers

riak-admin security grant riak_kv.get,riak_kv.put,riak_kv.
delete on default cdn_uri_sig_keys to keysusers
```

#### See also:

For more information on security in Riak, see the [Riak Security documentation](#).

#### See also:

For more information on authentication and authorization in Riak, see the [Riak Authentication and Authorization documentation](#).

## Traffic Ops Configuration

There are a couple configurations that are necessary in Traffic Ops.

### 1. Database Updates

- A new profile for Riak needs to be added to the profile table
- A new type of Riak needs to be added to the type table
- The servers in the Riak cluster need to be added to the server table

---

**Note:** profile and type data should be pre-loaded by seeds sql script.

---

### 2. Configuration updates

- /opt/traffic\_ops/app/conf/<environment>/riak.conf needs to be updated to reflect the correct username and password for accessing riak.

## Configuring Riak Search

In order to more effectively support retrieval of SSL certificates by Traffic Router and Traffic Ops ORT, Traffic Vault uses [Riak search](#). Riak Search uses [Apache Solr](#) for indexing and searching of records. The following explains how to enable, configure, and validate Riak Search.

## Riak Configuration

On Each Riak Server:

1. **If java is not already installed on your Riak server, install Java**
  - To see if Java is already installed: `java -version`
  - To install Java: `yum install -y jdk`
2. **enable search in riak.conf**
  - `vim /etc/riak/riak.conf`
  - look for search and change `search = off` to `search = on`
3. **Restart Riak so search is on**
  - `service riak restart`

One time configuration:

1. **On one of the Riak servers in the cluster run the following riak-admin commands**

```
riak-admin security grant search.admin on schema to admin
riak-admin security grant search.admin on index to admin
riak-admin security grant search.query on index to admin
riak-admin security grant search.query on index sslkeys to admin
riak-admin security grant search.query on index to riakuser
riak-admin security grant search.query on index sslkeys to riakuser
riak-admin security grant riak_core.set_bucket on any to admin
```

2. **Add the search schema to Riak. This schema is a simple Apache Solr configuration file which will index all records on cdn**

- Get the schema file by either cloning the project and going to *traffic\_ops/app/config/misc/riak\_search* or from [github](#).
- Use `curl` to add the schema to riak: `curl -kvs -XPUT "https://admin:pass@riakserver:8088/search/schema/sslkeys" -H 'Content-Type:application/xml' -d @sslkeys.xml`

3. **Add search index to Riak**

- run the following `curl` command: `curl -kvs -XPUT "https://admin:pass@riakserver:8088/search/index/sslkeys" -H 'Content-Type:application/json' -d '{"schema":"sslkeys"}'`

4. **Associate the sslkeys index to the ssl bucket in Riak**

- run the following `curl` command: `curl -kvs -XPUT "https://admin:pass@riakserver:8088/buckets/ssl/props" -H 'content-type:application/json' -d '{"props":{"search_index":"sslkeys"}}'`

Riak Search (using Apache Solr) will now index all NEW records that are added to the “ssl” bucket. The `cdn`, `deliveryservice`, and `hostname` fields are indexed and when a search is performed riak will return the indexed fields along with the `crt` and `key` values for a `ssl` record. In order to add the indexed fields to current records and to get the current records added, a standalone script needs to be run. The following explains how to run the script.

1. Get script from github either by cloning the project and going to *traffic\_ops/app/script* or from [here](#)

2. Run the script by performing the following command `./update_riak_for_search.pl -to_url=https://traffic-ops.kabletown.net -to_un=user -to_pw=password`

Validate the search is working by querying against Riak directly: `curl -kvs "https://admin:password@riakserver:8088/search/query/sslkeys?wt=json&q=cdn:mycdn"`

Validation can also be done by querying Traffic Ops: `curl -Lvs -H "Cookie: $COOKIE" https://traffic-ops.kabletown.net/api/1.2/cdns/name/mycdn/sslkeys.json`

### 3.1.16 Quick How To Guides

Traffic Control is a complicated system, and documenting it is not trivial. Sometimes a picture says more than a thousand words, so here are some screen shot based tutorials on how to use some of the more involved features.

#### Configure Multi Site Origin

- 1) Create cachegroups for the origin locations, and assign the appropriate parent-child relationship between the mid and org cachegroups (click the image to see full size). Each mid cachegroup can be assigned a primary and secondary origin parent cachegroup. When the mid cache parent configuration is generated, origins in the primary cachegroups will be listed first, followed by origins in the secondary cachegroup. Origin servers assigned to the delivery service that are assigned to neither the primary nor secondary cachegroups will be listed last.

Name	Short Name	Type	Magnet Latitude	Magnet Longitude	Parent Loc	Last Updated
mid-east	east	MID_LOC	0	0	origin-east	2015-06-25 20:52:37
mid-west	west	MID_LOC	0	0	origin-west	2015-06-25 20:52:37
origin-east	org-east	ORG_LOC	0	0		2015-06-25 20:52:37
origin-west	org-west	ORG_LOC	0	0		2015-06-25 20:52:37

- 2) Create a profile to assign to each of the origins:

Profile name	Parameter name	Config file name	Value
ORG1_CDN1	domain_name	CRConfig.xml	cdn1.kabletown.net
ORG1_CDN1	GeolocationURL	CRConfig.xml	http://www.cdnlab.kabletown.net:8080/GeoLiteCity.dat.gz
ORG1_CDN1	weight	parent.config	1.0
ORG2_CDN1	domain_name	CRConfig.xml	cdn1.kabletown.net
ORG2_CDN1	GeolocationURL	CRConfig.xml	http://www.cdnlab.kabletown.net:8080/GeoLiteCity.dat.gz
ORG2_CDN1	weight	parent.config	1.0

- 3) Create server entries for the origination vips:

Host Name	Domain Name	Cache group	Location	IPAddr	IPv6 Addr	Status	Profile	ILO
org1	kabletown.net	origin-east	plocation-nyc-1	10.11.10.2		CCR_IGNORE	ORG1_CDN1	
org2	kabletown.net	origin-west	plocation-nyc-1	10.11.12.2		CCR_IGNORE	ORG2_CDN1	

- 4) Check the multi-site check box in the delivery service screen:

* CDN	cdn1
* Content Routing Type	HTTP
* Protocol	http
* DSCP Tag	0 - Best Effort
Signed URLs	No
Query String Handling	0 - use qstring in cache key, and pass up
Geo Limit?	None
Bypass FQDN	
Initial Dispersion	1 - Off
IPv6 Routing Enabled?	No
Range Request Handling	0 - Don't cache
Delivery Service DNS TTL	3600
* Origin Server Base URL	http://movies.origin.kabletown.net
* Use Multi Site Origin Feature	Yes

- 5) Assign the org servers to the delivery service that will have the multi site feature. Org servers assigned to a delivery service with multi-site checked will be assigned to be the origin servers for this DS.

**Delivery Service Server Assignments for movies-c1 (http://movies.origin.kabletown.net)**

- ☐ EDGE1\_CDN1\_421\_SSL
  - ☒ us-ca-losangeles
  - ☒ us-co-denver
  - ☐ us-il-chicago
    - ☒ atsec-chi-00
    - ☒ atsec-chi-01
    - ☒ atsec-chi-02
    - ☒ atsec-chi-03
    - ☐ good-host
  - ☒ us-ny-newyork
  - ☒ us-pa-philadelphia
  - ☒ us-tx-houston

Save

Close

**Note:** “Origin Server Base URL” uniqueness: In order to enable MID caches to distinguish delivery services by different MSO algorithms while performing parent failover, it requires that “Origin Server Base URL” (OFQDN) for

each MSO enabled delivery service is unique unless the exceptions listed afterwards. This means that the OFQDN of a MSO enabled delivery service should be different with the OFQDNs of any other delivery service, regardless of whether they are MSO enabled or not. The exceptions are: 1) If there are multiple CDNs created on the same Traffic Ops, delivery services across different CDNs may have the same OFQDN configured. 2) If several delivery services in the same CDN have the same MSO algorithm configured, they may share the same OFQDN. 3) If delivery services are assigned with different MID cache groups respectively, they can share the same OFQDN. 4) This OFQDN must be valid - ATS will perform a DNS lookup on this FQDN even if IPs, not DNS, are used in the parent.config. 5) The OFQDN entered as the “Origin Server Base URL” will be sent to the origins as a host header. All origins must be configured to respond to this host.

- 6) For ATS 5.x, configure the mid\_hdr\_rewrite on the delivery service, example:

```
cond %{REMAP_PSEUDO_HOOK} __RETURN__ set-config proxy.config.http.parent_origin.  
↳dead_server_retry_enabled 1 __RETURN__ set-config proxy.config.http.parent_  
↳origin.simple_retry_enabled 1 __RETURN__ set-config proxy.config.http.parent_  
↳origin.simple_retry_response_codes "400,404,412" __RETURN__ set-config proxy.  
↳config.http.parent_origin.dead_server_retry_response_codes "502,503" __RETURN__  
↳set-config proxy.config.http.connect_attempts_timeout 2 __RETURN__ set-config_  
↳proxy.config.http.connect_attempts_max_retries 2 __RETURN__ set-config proxy.  
↳config.http.connect_attempts_max_retries_dead_server 1 __RETURN__ set-config_  
↳proxy.config.http.transaction_active_timeout_in 5 [L] __RETURN__
```

- 7) Create a delivery service profile. This must be done to set the MSO algorithm. Also, as of ATS 6.x, multi-site options must be set as parameters within the parent.config. Header rewrite parameters will be ignored. See *ATS parent.config* <<https://docs.trafficserver.apache.org/en/6.2.x/admin-guide/files/parent.config.en.html>> for more details. These parameters are now handled by the creation of a delivery service profile.

- a) Create a profile of the type DS\_PROFILE for the delivery service in question.

- b) Click “Show profile parameters” to bring up the parameters screen for the profile. Create parameters for the following:

Parameter Name	Config File Name	Value	ATS parent.config value
mso.algorithm	parent.config	true, false, strict, consistent_hash	round_robin
mso.parent_retry	parent.config	simple_retry, both, unavailable_server_retry	parent_retry
mso.unavailable_server_retry_responses	parent.config	list of server response codes, eg "500,502,503"	defaults to the value in records.config when unused.
mso.max_simple_retries	parent.config	Number of retries made after a 4xx error	defaults to the value in records.config when unused.
mso.max_unavailable_server_retries	parent.config	Number of retries made after a 5xx error	defaults to the value in records.config when unused.

Profile name	Parameter name	Config file name	Value	Secure
DS_example	mso.algorithm	parent.config	false	no
DS_example	mso.parent_retry	parent.config	both	no
DS_example	mso.unavailable_server_retry_responses	parent.config	"500,502,503"	no

c) In the delivery service page, select the newly created DS\_PROFILE and save the delivery service.

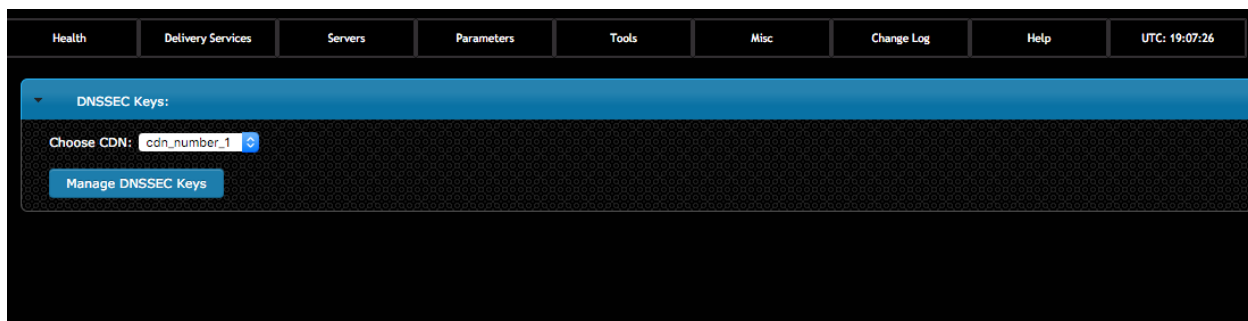
11) Turn on parent\_proxy\_routing in the MID profile.

## Configure DNSSEC

**Note:** In order for Traffic Ops to successfully store keys in Traffic Vault, at least one Riak Server needs to be configured in Traffic Ops. See the [Traffic Vault admin page](#) for more information.

**Note:** Currently DNSSEC is only supported for DNS delivery services.

1) Go to Tools->Manage DNSSEC Keys choose a CDN and click Manage DNSSEC Keys



2) Generate keys for a CDN by clicking Generate Keys then entering the following information:

- Expiration in days for the Zone Signing Key (ZSK)
- Expiration in days for the Key Signing Key (KSK)
- Effective Date

Once the required information has been entered click on the 'Generate Keys' button.

Depending upon the number of Delivery Services in the CDN, generating DNSSEC keys may take several seconds.

- 3) In order for DNSSEC to work properly, the DS Record information needs to be added to the parent zone of the CDN's domain (e.g. If the CDN's domain is 'cdn.kabletown.net' the parent zone is 'kabletown.net').

If you control your parent zone you can enter this information yourself, otherwise you will need to work with your DNS team to get the DS Record added to the parent zone.

- 4) Once DS Record information has been added to the parent zone, DNSSEC needs to be activated for the CDN so that Traffic Router will sign responses.

Click on Tools -> Manage DNSSEC Keys -> Choose your CDN -> On the Manage DNSSEC Keys page click the activate DNSSEC Keys button.

This will add a 'dnssec.enabled = "true"' entry to CRConfig for the chosen CDN.

Manage DNSSEC Keys

**DNSSEC Key Information:**

CDN Name:

TTL:

DNSSEC Active?:

KSK Expiration:

**DS Record Information:**

Algorithm:

Digest Type:

Digest:

- 5) DNSSEC should now be active on your CDN and Traffic Router should be signing responses.

A dig command with +dnssec added should show you the signed responses.

```
dig edge.cdn.kabletown.net. +dnssec
```

- 6) When KSK expiration is approaching (default 365 days), it is necessary to manually generate a new KSK for the TLD (Top Level Domain) and add the DS Record to the parent zone. In order to avoid signing errors, it is suggested that an effective date is chosen which allows time for the DS Record to be added to the parent zone before the new KSK becomes active.

A new KSK can be generated by clicking the 'Regenerate KSK' button on the Manage DNSSEC Keys screen (see screenshot above).

## Configure Federations

- 1) Create a user with a federations role (Misc -> Users -> Add User). This user will have the ability to perform the following actions:
  - Edit the federation
  - Delete the federation
  - Add IPV4 resolvers
  - Add IPV6 resolvers

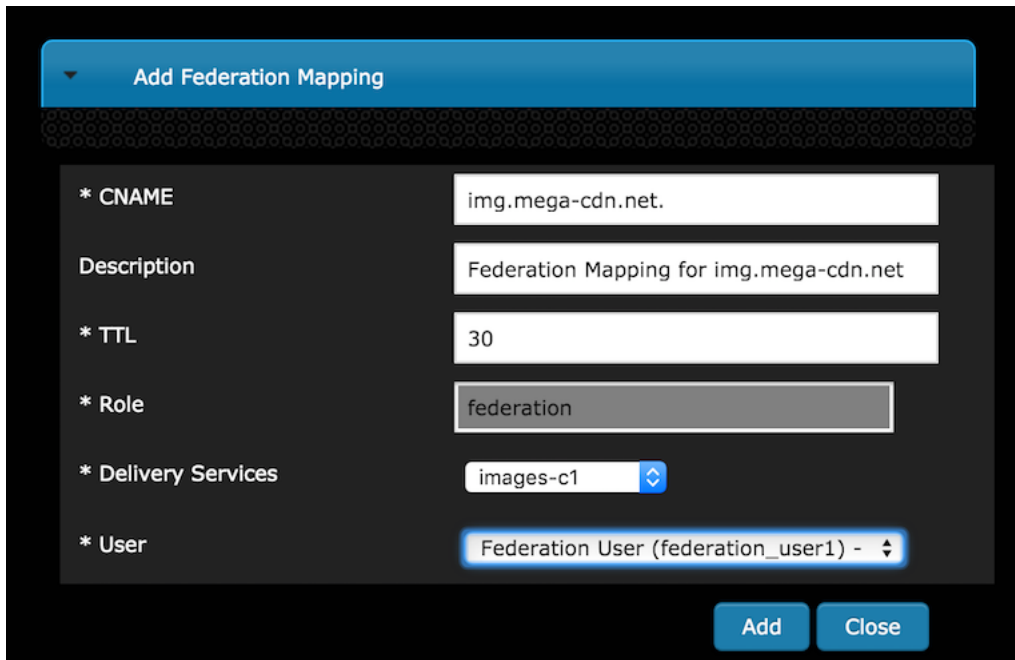


**Add User**

* Full Name	Federation User
* Username	federation_user1
* Email	federation_user@mega-cdn.net
* Password	.....
* Confirm Password	.....
Role	federation
Phone Number	
Company	
Address Line 1	
Address Line 2	
City	
State or Province	
Postal Code	
Country	
UID	
GID	
Last Updated	

**Add User** **Close**

- 2) As a user with admin privileges, create a Federation Mapping by going to Delivery Services -> Federations and then clicking 'Add Federation Mapping'
- 3) Choose the Delivery Service for the federation to be mapped to and assign it to the Federation User; click Add.



**Add Federation Mapping**

\* CNAME: img.mega-cdn.net.

Description: Federation Mapping for img.mega-cdn.net

\* TTL: 30

\* Role: federation

\* Delivery Services: images-c1

\* User: Federation User (federation\_user1)

Add Close

- 4) After the Federation is added, Traffic Ops will display the Federation.

Changes can be made at this time or the Federation can be deleted. Notice that no resolvers have been added to the federation yet. This can only be done by the federation user created in step 1.

If no further action is necessary, the Close button will close the window and display the list of all Federations.

The Federation user can now add resolvers to the Federation Mapping in Traffic Ops.

- 5) The federation user logs to traffic ops and stores the mojolicious cookie. The mojolicious cookie can be obtained manually using the debug tools on a web browser or via curl.

Example:

```
$ curl -i -XPOST "http://localhost:3000/api/1.1/user/login" -H "Content-Type:
↪application/json" -d '{ "u": "federation_user1", "p": "password" }'

HTTP/1.1 200 OK
Date: Wed, 02 Dec 2015 21:12:06 GMT
Content-Length: 65
Access-Control-Allow-Credentials: true
Content-Type: application/json
Access-Control-Allow-Methods: POST,GET,OPTIONS,PUT,DELETE
Set-Cookie:
↪mojolicious=eyJleHBpcmVzIjoxNDQ5MTA1MTI2LCJhdXRoX2RhGEiOiJmZWRLcmF0aW9uX3VzZXIhIn0-
↪--06b4f870d809d82a91433e92eae8320875c3e8b0; expires=Thu, 03 Dec 2015 01:12:06 GMT; path=/; HttpOnly
```

(continued from previous page)

```

Server: Mojolicious (Perl)
Access-Control-Allow-Headers: Origin, X-Requested-With, Content-Type, Accept
Cache-Control: no-cache, no-store, max-age=0, must-revalidate
Connection: keep-alive
Access-Control-Allow-Origin: http://localhost:8080

{"alerts":[{"level":"success","text":"Successfully logged in."}]}
```

6) The federation user sends a request to Traffic Ops to add IPV4 and/or IPV6 resolvers

Example:

```

$ curl -ki -H "Cookie:
↪mojolicious=eyJleHBpcmVzIjoxNDQ5MTA1MTI2LCJhdXRoX2RhdGEiOiJmZWRLcmF0aW9uX3VzZXIxIn0-
↪--06b4f870d809d82a91433e92eae8320875c3e8b0;" -XPUT 'http://
↪localhost:3000/api/1.2/federations' -d '
    {"federations": [
      {
        "deliveryService": "images-cl",
        "mappings":
          { "resolve4": [ "8.8.8.8/32", "8.8.4.4/32
↪" ],
          "resolve6": ["2001:4860:4860::8888/128",
↪"2001:4860:4860::8844"]
        }
      }
    ]
  }'
```

```

HTTP/1.1 200 OK
Set-Cookie:
↪mojolicious=eyJleHBpcmVzIjoxNDQ5MTA1OTQyLCJhdXRoX2RhdGEiOiJmZWRLcmF0aW9uX3VzZXIxIn0-
↪--b42be0749415cefd1d14e1a91bb214845b4de556; expires=Thu, 03 Dec 2015
↪01:25:42 GMT; path=/; HttpOnly
Server: Mojolicious (Perl)
Date: Wed, 02 Dec 2015 21:25:42 GMT
Content-Length: 74
Access-Control-Allow-Credentials: true
Content-Type: application/json
Access-Control-Allow-Methods: POST,GET,OPTIONS,PUT,DELETE
Cache-Control: no-cache, no-store, max-age=0, must-revalidate
Access-Control-Allow-Origin: http://localhost:8080
Connection: keep-alive
Access-Control-Allow-Headers: Origin, X-Requested-With, Content-Type,
↪Accept

{"response":"federation_user1 successfully created federation resolvers."}
```

7) The resolvers added by the federation user will now visible in Traffic Ops.

- 8) Any requests made from a client that resolves to one of the federation resolvers will now be given a CNAME from Traffic Router.

Example:

```
$ dig @tr.kabletown.net foo.images-c1.kabletown.net

; <<>> DiG 9.7.3-RedHat-9.7.3-2.el6 <<>> @tr.kabletown.net foo.images-c1.
↪kabletown.net
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 45110
;; flags: qr rd; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0
;; WARNING: recursion requested but not available

;; QUESTION SECTION:
;foo.images-c1.kabletown.net.    IN A
```

(continues on next page)

(continued from previous page)

```
;; ANSWER SECTION:
foo.images-cl.kabletown.net.      30 IN CNAME img.mega-cdn.net.

;; Query time: 9 msec
;; SERVER: 10.10.10.10#53(10.10.10.10)
;; WHEN: Wed Dec  2 22:05:26 2015
;; MSG SIZE rcvd: 84
```

## Configure Regional Geo-blocking (RGB)

---

**Note:** RGB is only supported for HTTP delivery services.

---

### 1) Prepare RGB configuration file

RGB uses a configuration file in JSON format to define regional geo-blocking rules for delivery services. The file needs to be put on an HTTP server accessible to Traffic Router. An example of the JSON is as follows:

```
{
  "deliveryServices":
  [
    {
      "deliveryServiceId": "hls-live",
      "urlRegex": ".*live4\\.m3u8",
      "geoLocation": {"includePostalCode": ["N0H", "L9V", "L9W"]},
      "redirectUrl": "http://third-party.com/blacked_out.html"
    },
    {
      "deliveryServiceId": "hls-live",
      "urlRegex": ".*live5\\.m3u8",
      "ipWhiteList": [185.68.71.9/22, "142.232.0.79/24"],
      "geoLocation": {"excludePostalCode": ["N0H", "L9V"]},
      "redirectUrl": "/live5_low_bitrate.m3u8"
    }
  ]
}
```

- The value of “deliveryServiceId” shall be equal to the “XML ID” field of the intended delivery service defined on Traffic Ops.
- “urlRegex” is to match request URLs. The URLs matching the regex are applicable to the rule.
- “geoLocation” currently supports “includePostalCode” and “excludePostalCode” attribute. When “includePostalCode” attribute is used, only the clients whose FSAs are in the “includePostalCode” list are able to view the content represented by “urlRegex”. When “excludePostalCode” is used, any client whose FSA are not in the “excludePostalCode” list are allowed to view the content. “includePostalCode” and “excludePostalCode” are mutually exclusive in one rule. (FSA: Forward Sortation Area, first three postal characters of Canadian postal codes)
- “redirectUrl” is the URL that will be responded to the blocked clients. Without a domain name in the URL, the URL will still be served in the same delivery service. Thus Traffic Router will redirect the client to a chosen cache server assigned to the delivery service. If the URL includes a domain name, Traffic Router simply redirects the client to the defined URL. In the later case, the redirect URL must not match the “urlRegex” part to avoid HTTP 302 loop on Traffic Router.

- “ipWhiteList” is an optional element. It includes a list of CIDR (Classless Inter-Domain Routing) blocks indicating the IPv4 subnets that are allowed by the rule. If this list exists and the value is not empty, client IP will be matched against the CIDR list, and if there is any match, the request will be allowed and no postal code matching logic is needed. If there is no match in the white list, postal code matching logic will be processed further.

## 2) Add RGB parameters on Traffic Ops

The two new parameters in following table are required to be added into CRConfig.json:

- “regional\_geoblocking.polling.url”: the HTTP URL of RGB configuration file. Traffic Router will fetch the file from this URL.
- “regional\_geoblocking.polling.interval”: the interval that Traffic Router polls the RGB configuration file.

Health	Delivery Services	Servers	Parameters	Tools
Misc	Change Log	Help	UTC: 5:55:41	
Search: regional				
	Profile name	Parameter name	Config file name	Value
<a href="#">Edit</a>	CCR_CDN	regional_geoblock.polling.interval	CRConfig.json	86400000
<a href="#">Edit</a>	CCR_CDN	regional_geoblock.polling.url	CRConfig.json	http://10.75.168.90/ftp/regional_geoblock.json
Showing 1 to 2 of 2 entries (filtered from 433 total entries)				
Add Parameter				

## 3) Enable RGB for a delivery service

Regex remap expression

Long Description

Customer

Service

Info URL

Check Path

Origin Shield (Pipe Delimited String)

Regional Geoblocking: ☒ Disabled ☐ Enabled

\* Active: ☐ No ☐

**Regular expressions for this delivery service:**

Type	Order	Regular Expression
HOST_REGEX	0	

**Add a new Regex:**

Select...

Save Close

#### 4) Make configuration effective

Go to Tools->Snapshot CRConfig, perform “Diff CRConfig” and click “Write CRConfig”.

CRConfig Diff:

Choose CDN:

#### 5) Traffic Router access log with RGB

RGB extends the field of “rtype” and adds a new field “rgb” in Traffic Router access.log to help to monitor the working of this feature.

For “rtype”, RGALT indicates that a request is redirected to an alternate URL by RGB; RGDENY indicates that a



request is denied by RGB because there is no matching rule in JSON for this request.

For “rgb”, when RGB is enabled, it will be non-empty with following format:

```
{FSA}:{allowed/disallowed}:{include/exclude postal}:{fallback config/current config}:
↪{allowed by whitelist/otherwise}
```

- {FSA}: It is the client’s FSA part of its postal code, which is retrieved from geo-location database. If FSA is empty, dash (“-“) is filled in.
- {allowed/disallowed}: This flag shows if a request is allowed or disallowed by RGB (1 for yes, and 0 for no).
- {include/exclude postal}: It shows that when a rule in JSON is matched for a request, it is an include or exclude list of postal codes (i.e. FSAs). “I” for include, and “X” for exclude. If no rule matches, dash (“-“) is filled in.
- {fallback config/current config}: when TR fails to parse an RGB JSON, TR will handle requests with latest valid JSON configuration, but will set {fallback config} flag to 1. If the new JSON is valid, then the flag is set to 0.
- {allowed by whitelist/otherwise}: If a request is allowed by whitelist, this flag is set to 1; for all other cases, it is 0.

Example:

```
1446442214.685 qtype=HTTP chi=129.100.254.79 url="http://foo.geo2.cdn.com/live5.m3u8"
↪cqhm=GET cqhv=HTTP/1.1 rtype=GEO rloc="-" rdtl=- rerr="-" rgb="N6G:1:X:0:0"
↪pssc=302 ttms=3 rurl=http://cent6-44.geo2.cdn.com/live5.m3u8 rh="-"

1446442219.181 qtype=HTTP chi=184.68.71.9 url="http://foo.geo2.cdn.com/live5.m3u8"
↪cqhm=GET cqhv=HTTP/1.1 rtype=RGALT rloc="-" rdtl=- rerr="-" rgb="-:0:X:0:0"
↪pssc=302 ttms=3 rurl=http://cent6-44.geo2.cdn.com/low_bitrate.m3u8 rh="-"

1446445521.677 qtype=HTTP chi=24.114.29.79 url="http://foo.geo2.cdn.com/live51.m3u8"
↪cqhm=GET cqhv=HTTP/1.1 rtype=RGDENY rloc="-" rdtl=- rerr="-" rgb="L4S:0:-:0:0"
↪pssc=520 ttms=3 rurl="-" rh="-"
```

## Configure Delivery Service Steering

- 1) Create two target delivery services in Traffic Ops. They must both be HTTP delivery services that are part of the same CDN.

Health

Delivery Services

Servers

Parameters

Tools

Misc

Change Log (2)

Help

UTC: 19:06:37

Search: target

XML ID	Origin Server	CDN	Profile	CCR TTL	Active	Type	DSCP	Signed	QSH	Geo Limit	Protocol	IPv6	RR
target-deliveryservice-1	http://target-ds-origin.com	cdn_number_1	CCR_CDN1	3600	no	HTTP	0	no	0	none	http	no	0
target-deliveryservice-2	http://target-ds-2-origin.com	cdn_number_1	CCR_CDN1	3600	no	HTTP	0	no	0	none	http	no	0

Showing 1 to 2 of 2 entries (filtered from 10 total entries)

Add Delivery Service

- 2) Create a delivery service with type STEERING in Traffic Ops.

Add Delivery Service

**Delivery Service Info:**

\* XML ID

steering-ds

\* Display Name

Steering Deliveryservice

\* CDN

cdn\_number\_1

\* Content Routing Type

STEERING

IPv6 Routing Enabled?

Yes

\* CCR profile

CCR\_CDN1 (Comcast Content Router for cdn1.cdn.net)

Long Description

Customer

Service

Info URL

Check Path

\* Active

Yes

\* Logs Enabled

No

**Regular expressions for this delivery service:**

Type:

HOST\_REGEX

Order:

1

Regular Expression:

.\*\steering-ds\.\*

Save

Close

- 3) Click the 'Manage Steering Assignments' button on the delivery service screen to assign targets.

126

Chapter 3. Administrator's Guide

\* Content Routing Type

STEERING

IPv6 Routing Enabled?

\* CCR profile

Long Description

Customer

Service

Info URL

Check Path

\* Active

\* Logs Enabled

Last Updated

Number of edges assigned:

Number of static DNS entries:

Example delivery URL

**Steering Assignments**

**Steering Information:**

DeliveryService Name:

Target Name:

Target Name:

**Regular expressions for this delivery service:**

Type	Order	Regular Expression
HOST_REGEX	0	.*\steering-ds\..*

4) Create a user with the role of Steering.

- 5) As the steering user, assign weights or orders to target delivery services. Assignments must either have a value for weight or order, but not both. The value of weight must be a positive integer, while the value of order can be any integer. This will require logging in to Traffic Ops first via `http://to.kabletown.net/api/1.2/user/login` and storing the `mojolicious` cookie.

Sample cURL: `curl -H "Cookie: mojolicious=xxxxxy" -XPUT "https://to.kabletown.net/internal/api/1.2/steering/steering-ds" -d @/tmp/steering.json`

Sample JSON body:

```
{
  "targets": [
    {
      "weight": "1000",
      "deliveryService": "target-deliveryservice-1"
    },
    {
      "weight": "9000",
      "deliveryService": "target-deliveryservice-2"
    },
    {
      "order": -1,
      "deliveryService": "target-deliveryservice-3"
    }
  ]
}
```

(continues on next page)

(continued from previous page)

```

    "order": 3,
    "deliveryService": "target-deliveryservice-4"
  }
]
}

```

- 6) If desired, the steering user can create filters for the target delivery services.

Sample cURL: `curl -H "Cookie: mojolicious=xxxxxyy" -XPUT "https://to.kabletown.net/internal/api/1.2/steering/steering-ds" -d @/tmp/steering.json`

Sample JSON body:

```

{
  "filters": [
    {
      "pattern": ".*\\gototarget1\\.\\.\\.\"",
      "deliveryService": "target-deliveryservice-1"
    }
  ],
  "targets": [
    {
      "weight": "1000",
      "deliveryService": "target-deliveryservice-1"
    },
    {
      "weight": "9000",
      "deliveryService": "target-deliveryservice-2"
    },
    {
      "order": -1,
      "deliveryService": "target-deliveryservice-3"
    },
    {
      "order": 3,
      "deliveryService": "target-deliveryservice-4"
    }
  ]
}

```

- 7) Any requests to Traffic Router for the steering delivery service should now be routed to target delivery services based on configured weight or order. Example: `curl -Lvs http://foo.steering-ds.cdn.kabletown.net/bar`



A guide to the various internal and external APIs, and an introduction for the Traffic Control developer.

## 4.1 Developer's Guide

Use this guide to start developing applications that consume the Traffic Control APIs, to create extensions to Traffic Ops, or work on Traffic Control itself.

### 4.1.1 Building Traffic Control

#### Build using pkg

This is the easiest way to build all the components of Traffic Control; all requirements are automatically loaded into the image used to build each component.

#### Requirements

- `docker` (<https://docs.docker.com/engine/installation/>)
- `docker-compose` (<https://docs.docker.com/compose/install/>) (optional, but recommended)

If `docker-compose` is not available, the `pkg` script will automatically download and run it in a container. This is noticeably slower than running it natively.

#### Usage

```
$ ./pkg -?
Usage: ./pkg [options] [projects]
  -q      Quiet mode. Supresses output.
  -v      Verbose mode. Lists all build output.
  -l      List available projects.

If no projects are listed, all projects will be packaged.
Valid projects:
  - traffic_portal_build
  - traffic_router_build
  - traffic_monitor_build
  - source
  - traffic_ops_build
  - traffic_stats_build
```

If any project names are provided on the command line, only those will be built. Otherwise, all projects are built.

All artifacts (rpms, logs, source tar ball) are copied to dist at the top level of the incubator-trafficcontrol directory.

## Example

```
$ ./pkg -v source traffic_ops_build
Building source.
Building traffic_ops_build.
```

## Build using docker-compose

If the pkg script fails, docker-compose can still be used directly.

## Usage

```
$ docker-compose -f ./infrastructure/docker/build/docker-compose.yml down -v
$ docker-compose -f ./infrastructure/docker/build/docker-compose.yml up --build_
↪source traffic_ops_build
$ ls -l dist/
build-traffic_ops.log
traffic_ops-2.1.0-6396.07033d6d.el7.src.rpm
traffic_ops-2.1.0-6396.07033d6d.el7.x86_64.rpm
traffic_ops_ort-2.1.0-6396.07033d6d.el7.src.rpm
traffic_ops_ort-2.1.0-6396.07033d6d.el7.x86_64.rpm
trafficcontrol-incubating-2.1.0.tar.gz
```

## 4.1.2 Traffic Ops

### Introduction

Traffic Ops uses a Postgres database to store the configuration information, and the [Mojolicious framework](#) to generate the user interface and REST APIs.



## Software Requirements

To work on Traffic Ops you need a \*nix (MacOS and Linux are most commonly used) environment that has the following installed:

- [Carton 1.0.12](#)
  - cpan JSON
  - cpan JSON::PP
- [Go 1.8.3](#)
- Perl 5.10.1
- Git
- Postgres 9.6.6
- [Goose](#)

Additionally, the installation of the following RPMs (or equivalent) is required:

- All RPMs listed in *[Traffic Ops - Migrating from 1.x to 2.x](#)*

## Traffic Ops Project Tree Overview

### /opt/traffic\_ops/app

- bin/ - Directory for scripts, cronjobs, etc
- conf/
  - /development - Development (local) specific config files.
  - /misc - Miscellaneous config files.
  - /production - Production specific config files.
  - /test - Test (unit test) specific config files.
- db/ - Database related area.
  - /migrations - Database Migration files.
- lib/
  - /API - Mojo Controllers for the /API area of the application.
  - /Common - Common Code between both the API and UI areas.
  - /Extensions
  - Fixtures/ - Test Case fixture data for the 'to\_test' database. \* /Integration - Integration Tests.
  - /MojoPlugins - Mojolicious Plugins for Common Controller Code.
  - Schema/ - Database Schema area. \* /Result - DBIx ORM related files.
  - /Test - Common Test.
  - /UI - Mojo Controllers for the Traffic Ops UI itself.
  - Utils/ \* /Helper - Common utilities for the Traffic Ops application.
- log/ - Log directory where the development and test files are written by the app.
- public/

- `css/` - Stylesheets.
- `images/` - Images.
- `js/` - Javascripts
- `script/` - Mojo Bootstrap scripts.
- `t/` - Unit Tests for the UI.
- `api/` - Unit Tests for the API.
- `t_integration/` - High level tests for Integration level testing.
- `templates/` - Mojo Embedded Perl (.ep) files for the UI.

## Perl Formatting Conventions

Perl tidy is for use in code formatting. See the following config file for formatting conventions.

```
edit a file called $HOME/.perltidyrc

-l=156
-et=4
-t
-ci=4
-st
-se
-vt=0
-cti=0
-pt=1
-bt=1
-sbt=1
-bbt=1
-nsfs
-nolq
-otr
-aws
-wls="= + - / * . \"
-wrs=\"= + - / * . \"
-wbb=\"% + - * / x != == >= <= =~ < > | & **= += *= &= <<= &&= -= /= |= + >>= || = . =
↪ %= ^= x=
```

## Database Management

The `admin.pl` script is for use in managing the Traffic Ops database tables. Below is an example of its usage.

```
$ db/admin.pl
```

Usage: `db/admin.pl [-env (development|test|production)] [arguments]`

Example: `db/admin.pl --env=test reset`

Purpose: This script is used to manage the database. The environments are defined in the `dbconf.yml`, as well as the database names.

- To use the `admin.pl` script, you may need to add `traffic_ops/lib` and `traffic_ops/local/lib/perl5` to your `PERL5LIB` environment variable.

Arguments	Description
create	Execute db 'create' the database for the current environment.
down	Roll back a single migration from the current version.
drop	Execute db 'drop' on the database for the current environment.
redo	Roll back the most recently applied migration, then run it again.
reset	Execute db drop, create, load_schema, migrate on the database for the current environment.
seed	Execute SQL from db/seeds.sql for loading static data.
setup	Execute db drop, create, load_schema, migrate, seed on the database for the current environment.
status	Print the status of all migrations.
upgrade	Execute migrate then seed on the database for the current environment.

## Installing The Developer Environment

To install the Traffic Ops Developer environment:

1. Clone the traffic\_control repository from [github.com](https://github.com).
2. Install the local dependencies using Carton (cpanfile).

```
$ cd traffic_ops/app
$ carton
```

3. Set up a role (user) in Postgres

See Postgres instructions on initdb [https://wiki.postgresql.org/wiki/First\\_steps](https://wiki.postgresql.org/wiki/First_steps)

4. Enter db/admin.pl --env=<environment name> setup to set up the traffic\_ops database(s).
  - Unit test database: \$ db/admin.pl --env=test setup
  - Development database: \$ db/admin.pl --env=development setup
  - Integration database: \$ db/admin.pl --env=integration setup

Running the the admin.pl script in setup mode should look like this:

```
master $ db/admin.pl --env=development setup
Using database.conf: conf/development/database.conf
Using database.conf: conf/development/database.conf
Using database.conf: conf/development/database.conf
Using database.conf: conf/development/database.conf
Using database.conf: conf/development/database.conf
Using database.conf: conf/development/database.conf
Executing 'drop database to_development'
Executing 'create database to_development'
Creating database tables...
Warning: Using a password on the command line interface can be insecure.
Migrating database...
goose: migrating db environment 'development', current version: 0, target: 20150210100000
↪20150210100000
OK    20141222103718_extension.sql
OK    20150108100000_add_job_deliveryservice.sql
OK    20150205100000_cg_location.sql
```

(continues on next page)

(continued from previous page)

```
OK    20150209100000_cran_to_asn.sql
OK    20150210100000_ds_keyinfo.sql
Seeding database...
Warning: Using a password on the command line interface can be insecure.
```

5. (Optional) To load temporary data into the tables: `$ perl bin/db/setup_kabletown.pl`
6. Run the postinstall script: `traffic_ops/install/bin/postinstall`
7. To start Traffic Ops, enter `$ bin/start.pl`

The local Traffic Ops instance uses an open source framework called morbo, starting following the start command execution.

Start up success includes the following:

```
[2015-02-24 10:44:34,991] [INFO] Listening at "http://*:3000".
Server available at http://127.0.0.1:3000.
```

8. Using a browser, navigate to the given address: `http://127.0.0.1:3000`
9. For the initial log in:
  - User name: admin
  - Password: password
10. Change the log in information.

## Test Cases

Use prove to execute test cases. Execute after a carton install:

- To run the Unit Tests: `$ local/bin/prove -qrp t/`
- To run the Integration Tests: `$ local/bin/prove -qrp t_integration/`

## The KableTown CDN example

The integration tests will load an example CDN with most of the features of Traffic Control being used. This is mostly for testing purposes, but can also be used as an example of how to configure certain features. To load the KableTown CDN example and access it:

1. Run the integration tests
2. Start morbo against the integration database: `export MOJO_MODE=integration; ./bin/start.pl`
3. Using a browser, navigate to the given address: `http://127.0.0.1:3000`
4. For the initial log in:
  - User name: admin
  - Password: password

## Extensions

Traffic Ops Extensions are a way to enhance the basic functionality of Traffic Ops in a custom manner. There are three types of extensions:

1. Check Extensions

These allow you to add custom checks to the “Health->Server Checks” view.

2. Configuration Extensions

These allow you to add custom configuration file generators.

3. Data source Extensions

These allow you to add statistic sources for the graph views and APIs.



Extensions are managed using the `$TO_HOME/bin/extensions` command line script. For more information see [Managing Traffic Ops Extensions](#).

## Check Extensions

In other words, check extensions are scripts that, after registering with Traffic Ops, have a column reserved in the “Health->Server Checks” view and that usually run periodically out of cron.

It is the responsibility of the check extension script to iterate over the servers it wants to check and post the results.

An example script might proceed by logging into the Traffic Ops server using the HTTPS `base_url` provided on the command line. The script is hardcoded with an auth token that is also provisioned in the Traffic Ops User database. This token allows the script to obtain a cookie used in later communications with the Traffic Ops API. The script then obtains a list of all caches to be polled by accessing Traffic Ops’ `/api/1.1/servers.json` REST target. This list is walked, running a command to gather the stats from that cache. For some extensions, an HTTP GET request might be made to the ATS `astats` plugin, while for others the cache might be pinged, or a command run over SSH. The results are then compiled into a numeric or boolean result and the script POSTs the result back to the Traffic Ops `/api/1.1/servercheck/` target.

A check extension can have a column of ’s and ’s (`CHECK_EXTENSION_BOOL`) or a column that shows a number (`CHECK_EXTENSION_NUM`). A simple example of a check extension of type `CHECK_EXTENSION_NUM` that will show 99.33 for all servers of type EDGE is shown below:

```
Script here.
```

Check Extension scripts are located in the `$TO_HOME/bin/checks` directory.

Currently, the following Check Extensions are available and installed by default:

**Cache Disk Usage Check - CDU** This check shows how much of the available total cache disk is in use. A “warm” cache should show 100.00.

**Cache Hit Ratio Check - CHR** The cache hit ratio for the cache in the last 15 minutes (the interval is determined by the cron entry).

**DiffServe CodePoint Check - DSCP** Checks if the returning traffic from the cache has the correct DSCP value as assigned in the delivery service. (Some routers will overwrite DSCP)

**Maximum Transmission Check - MTU** Checks if the Traffic Ops host (if that is the one running the check) can send and receive 8192 size packets to the `ip_address` of the server in the server table.

**Operational Readiness Check - ORT** See *Configuring Traffic Server* for more information on the ort script. The ORT column shows how many changes the traffic\_ops\_ort.pl script would apply if it was run. The number in this column should be 0.

**Ping Check - 10G, ILO, 10G6, FQDN** The bin/checks/ToPingCheck.pl is to check basic IP connectivity, and in the default setup it checks IP connectivity to the following:

**10G** Is the ip\_address (the main IPv4 address) from the server table pingable?

**ILO** Is the ilo\_ip\_address (the lights-out-mangement IPv4 address) from the server table pingable?

**10G6** Is the ip6\_address (the main IPv6 address) from the server table pingable?

**FQDN** Is the Fully Qualified Domain name (the concatenation of host\_name and . and domain\_name from the server table) pingable?

**Traffic Router Check - RTR** Checks the state of each cache as perceived by all Traffic Monitors (via Traffic Router). This extension asks each Traffic Router for the state of the cache. A check failure is indicated if one or more monitors report an error for a cache. A cache is only marked as good if all reports are positive. (This is a pessimistic approach, opposite of how TM marks a cache as up, “the optimistic approach”)

## Configuration Extensions

NOTE: Config Extensions are Beta at this time.

## Data Source Extensions

Traffic Ops has the ability to load custom code at runtime that allow any CDN user to build custom APIs for any requirement that Traffic Ops does not fulfill. There are two classes of Data Source Extensions, private and public. Private extensions are Traffic Ops extensions that are not publicly available, and should be kept in the /opt/traffic\_ops\_extensions/private/lib. Public extensions are Traffic Ops extensions that are Open Source in nature and free to enhance or contribute back to the Traffic Ops Open Source project and should be kept in /opt/traffic\_ops/app/lib/Extensions.

## Extensions at Runtime

The search path for extensions depends on the configuration of the PERL5LIB, which is preconfigured in the Traffic Ops start scripts. The following directory structure is where Traffic Ops will look for Extensions in this order.

## Extension Directories

PERL5LIB Example Configuration:

```
export PERL5LIB=/opt/traffic_ops_extensions/private/lib/Extensions:/opt/traffic_ops/
↳ app/lib/Extensions/TrafficStats
```

## Perl Package Naming Convention

To prevent Extension namespace collisions within Traffic Ops all Extensions should follow the package naming convention below:

Extensions::**<ExtensionName>**

Data Source Extension Perl package name example Extensions::TrafficStats Extensions::YourCustomExtension

## TrafficOpsRoutes.pm

Traffic Ops accesses each extension through the addition of a URL route as a custom hook. These routes will be defined in a file called TrafficOpsRoutes.pm that should live in the top directory of your Extension. The routes that are defined should follow the Mojolicious route conventions.

## Development Configuration

To incorporate any custom Extensions during development set your PERL5LIB with any number of directories with the understanding that the PERL5LIB search order will come into play, so keep in mind that top-down is how your code will be located. Once Perl locates your custom route or Perl package/class it ‘pins’ on that class or Mojo Route and doesn’t look any further, which allows for the developer to *override* Traffic Ops functionality.

## API

The Traffic Ops API provides programmatic access to read and write CDN data providing authorized API consumers with the ability to monitor CDN performance and configure CDN settings and parameters.

## Response Structure

All successful responses have the following structure:

```
{
  "response": <JSON object with main response>,
}
```

To make the documentation easier to read, only the <JSON object with main response> is documented, even though the response and version fields are always present.

## Using API Endpoints

1. Authenticate with your Traffic Portal or Traffic Ops user account credentials.
2. Upon successful user authentication, note the mojolicious cookie value in the response headers.
3. Pass the mojolicious cookie value, along with any subsequent calls to an authenticated API endpoint.

Example:

```
[jvd@laika ~]$ curl -H "Accept: application/json" http://localhost:3000/api/1.1/usage/
↪ asns.json
{"alerts":[{"level":"error","text":"Unauthorized, please log in."}]}
```

```
[jvd@laika ~]$
[jvd@laika ~]$ curl -v -H "Accept: application/json" -v -X POST --data '{ "u":"admin",
↪ "p":"secret_passwd" }' http://localhost:3000/api/1.1/user/login
* Hostname was NOT found in DNS cache
* Trying ::1...
* connect to ::1 port 3000 failed: Connection refused
* Trying 127.0.0.1...
```

(continues on next page)

(continued from previous page)

```

* Connected to localhost (127.0.0.1) port 3000 (#0)
> POST /api/1.1/user/login HTTP/1.1
> User-Agent: curl/7.37.1
> Host: localhost:3000
> Accept: application/json
> Content-Length: 32
> Content-Type: application/x-www-form-urlencoded
>
* upload completely sent off: 32 out of 32 bytes
< HTTP/1.1 200 OK
< Connection: keep-alive
< Access-Control-Allow-Methods: POST,GET,OPTIONS,PUT,DELETE
< Access-Control-Allow-Origin: http://localhost:8080
< Access-Control-Allow-Headers: Origin, X-Requested-With, Content-Type, Accept
< Set-Cookie: mojolicious=eyJleHBpcmVzIjoxNDI5NDYmMjAxLCJhdXRoX2RhdGEiOiJhZG1pbiJ9--
↪f990d03b7180blece97c3bb5ca69803cd6a79862; expires=Sun, 19 Apr 2015 00:10:01 GMT;↪
↪path=/; HttpOnly
< Content-Type: application/json
< Date: Sat, 18 Apr 2015 20:10:01 GMT
< Access-Control-Allow-Credentials: true
< Content-Length: 81
< Cache-Control: no-cache, no-store, max-age=0, must-revalidate
* Server Mojolicious (Perl) is not blacklisted
< Server: Mojolicious (Perl)
<
* Connection #0 to host localhost left intact
{"alerts":[{"level":"success","text":"Successfully logged in."}]}
[jvd@laika ~]$

[jvd@laika ~]$ curl -H'Cookie:↪
↪mojolicious=eyJleHBpcmVzIjoxNDI5NDYmMjAxLCJhdXRoX2RhdGEiOiJhZG1pbiJ9--
↪f990d03b7180blece97c3bb5ca69803cd6a79862;' -H "Accept: application/json" http://
↪localhost:3000/api/1.1/asns.json
{"response":{"asns":[{"lastUpdated":"2012-09-17 15:41:22", .. asn data deleted .. },}
[jvd@laika ~]$

```

## API Errors

### Response Properties

Parameter	Type	Description
alerts	array	A collection of alert messages.
>level	string	Success, info, warning or error.
>text	string	Alert message.

The 3 most common errors returned by Traffic Ops are:

**401 Unauthorized** When you don't supply the right cookie, this is the response.

```

[jvd@laika ~]$ curl -v -H "Accept: application/json" http://localhost:3000/api/1.
↪1/usage/asns.json
* Hostname was NOT found in DNS cache
* Trying ::1...
* connect to ::1 port 3000 failed: Connection refused

```

(continues on next page)



(continued from previous page)

```

* Trying 127.0.0.1...
* Connected to localhost (127.0.0.1) port 3000 (#0)
> GET /api/1.1/usage/asns.json HTTP/1.1
> User-Agent: curl/7.37.1
> Host: localhost:3000
> Accept: application/json
>
< HTTP/1.1 401 Unauthorized
< Cache-Control: no-cache, no-store, max-age=0, must-revalidate
< Content-Length: 84
* Server Mojolicious (Perl) is not blacklisted
< Server: Mojolicious (Perl)
< Connection: keep-alive
< Access-Control-Allow-Methods: POST,GET,OPTIONS,PUT,DELETE
< Access-Control-Allow-Headers: Origin, X-Requested-With, Content-Type, Accept
< Access-Control-Allow-Origin: http://localhost:8080
< Date: Sat, 18 Apr 2015 20:36:12 GMT
< Content-Type: application/json
< Access-Control-Allow-Credentials: true
<
* Connection #0 to host localhost left intact
{"alerts":[{"level":"error","text":"Unauthorized, please log in."}]}
[jvd@laika ~]$

```

**404 Not Found** When the resource (path) is non-existent Traffic Ops returns a 404:

```

[jvd@laika ~]$ curl -v -H'Cookie:
↪mojolicious=eyJleHBpcmVzIjoxNDI5NDYmMjAxLCJhdXRhZGEiOiJhZGlubiJ9--
↪f990d03b7180blece97c3bb5ca69803cd6a79862;' -H "Accept: application/json" http://
↪localhost:3000/api/1.1/asnsjj.json
* Hostname was NOT found in DNS cache
* Trying ::1...
* connect to ::1 port 3000 failed: Connection refused
* Trying 127.0.0.1...
* Connected to localhost (127.0.0.1) port 3000 (#0)
> GET /api/1.1/asnsjj.json HTTP/1.1
> User-Agent: curl/7.37.1
> Host: localhost:3000
> Cookie: mojolicious=eyJleHBpcmVzIjoxNDI5NDYmMjAxLCJhdXRhZGEiOiJhZGlubiJ9--
↪f990d03b7180blece97c3bb5ca69803cd6a79862;
> Accept: application/json
>
< HTTP/1.1 404 Not Found
* Server Mojolicious (Perl) is not blacklisted
< Server: Mojolicious (Perl)
< Content-Length: 75
< Cache-Control: no-cache, no-store, max-age=0, must-revalidate
< Content-Type: application/json
< Date: Sat, 18 Apr 2015 20:37:43 GMT
< Access-Control-Allow-Credentials: true
< Set-Cookie:
↪mojolicious=eyJleHBpcmVzIjoxNDI5NDYmMjAxLCJhdXRhZGEiOiJhZGlubiJ9--
↪8a5a61b91473bc785d4073fe711de8d2c63f02dd; expires=Sun, 19 Apr 2015 00:37:43 GMT;
↪ path=/; HttpOnly
< Access-Control-Allow-Methods: POST,GET,OPTIONS,PUT,DELETE
< Connection: keep-alive
< Access-Control-Allow-Headers: Origin, X-Requested-With, Content-Type, Accept

```

(continues on next page)

(continued from previous page)

```
< Access-Control-Allow-Origin: http://localhost:8080
<
* Connection #0 to host localhost left intact
{"alerts":[{"text":"Resource not found.,"level":"error"}]}
[jvd@laika ~]$
```

**500 Internal Server Error** When you are asking for a correct path, but the database doesn't match, it returns a 500:

```
[jvd@laika ~]$ curl -v -H'Cookie:
↪mojolicious=eyJleHBpcmVzIjoxNDI5NDYmJmAxLCJhdXRoX2RhdGEiOiJhZGlpciJ9--
↪f990d03b7180blece97c3bb5ca69803cd6a79862;' -H "Accept: application/json" http://
↪localhost:3000/api/1.1/servers/hostname/jj/details.json
* Hostname was NOT found in DNS cache
* Trying ::1...
* connect to ::1 port 3000 failed: Connection refused
* Trying 127.0.0.1...
* Connected to localhost (127.0.0.1) port 3000 (#0)
> GET /api/1.1/servers/hostname/jj/details.json HTTP/1.1
> User-Agent: curl/7.37.1
> Host: localhost:3000
> Cookie: mojolicious=eyJleHBpcmVzIjoxNDI5NDYmJmAxLCJhdXRoX2RhdGEiOiJhZGlpciJ9--
↪f990d03b7180blece97c3bb5ca69803cd6a79862;
> Accept: application/json
>
< HTTP/1.1 500 Internal Server Error
* Server Mojolicious (Perl) is not blacklisted
< Server: Mojolicious (Perl)
< Cache-Control: no-cache, no-store, max-age=0, must-revalidate
< Content-Length: 93
< Set-Cookie:
↪mojolicious=eyJhdXRoX2RhdGEiOiJhZGlpciIsImV4cGlyZXMiOjE0MDJkMDQzMjZ9--
↪1b08977e91f8f68b0ff5d5e5f6481c76ddfd0853; expires=Sun, 19 Apr 2015 00:45:06 GMT;
↪ path=/; HttpOnly
< Content-Type: application/json
< Date: Sat, 18 Apr 2015 20:45:06 GMT
< Access-Control-Allow-Credentials: true
< Access-Control-Allow-Methods: POST,GET,OPTIONS,PUT,DELETE
< Connection: keep-alive
< Access-Control-Allow-Headers: Origin, X-Requested-With, Content-Type, Accept
< Access-Control-Allow-Origin: http://localhost:8080
<
* Connection #0 to host localhost left intact
{"alerts":[{"level":"error","text":"An error occurred. Please contact your
↪administrator."}]}
[jvd@laika ~]$
```

The rest of the API documentation will only document the 200 OK case, where no errors have occurred.

## Traffic Ops API Routes

### API Routes

1.0	1.1	1.2
/asns	/api/1.1/asns	/api/1.2/asns
/availablebds	/api/1.1/deliveryservices	/api/1.2/deliveryservices
<i>Not Implemented</i>	<i>Not Implemented</i>	/api/1.2/cache_stats
/datacrans	/api/1.1/crans.json	/api/1.2/crans.json
/datacrans/orderby/:field	/api/1.1/crans.json	/api/1.2/crans.json
/datadeliveryservice	/api/1.1/deliveryservices	/api/1.2/deliveryservices
/datadeliveryserviceserver	/api/1.1/deliveryserviceserver.json	/api/1.2/deliveryserviceserver.json
/datadomains	/api/1.1/cdns/domains.json	/api/1.2/cdns/domains.json
<i>Not Implemented</i>	<i>Not Implemented</i>	/api/1.2/deliveryservice_stats
/datahwinfo	/api/1.1/hwinfo	/api/1.2/hwinfo
/datalinks	/api/1.1/deliveryserviceserver.json	/api/1.2/deliveryserviceserver.json
/datalinks/orderby/:field	/api/1.1/deliveryserviceserver.json	/api/1.2/deliveryserviceserver.json
/datalogs	/api/1.1/logs	/api/1.2/logs
/datalocation/orderby/id	/api/1.1/cachegroups	/api/1.2/cachegroups
/datalocationparameters	/api/1.1/cachegroups	/api/1.2/cachegroups
/dataparameter	/api/1.1/parameters	/api/1.2/parameters
/dataparameter/:parameter	/api/1.1/parameters/profile/:parameter.json	/api/1.2/parameters/profile/:parameter.json
/dataphys_location	/api/1.1/phys_locations	/api/1.2/phys_locations
/dataprofile	/api/1.1/profiles	/api/1.2/profiles
/dataprofile/orderby/name		
/dataregion	/api/1.1/regions	/api/1.2/regions
/datarole	/api/1.1/roles	/api/1.2/roles
/datarole/orderby/:field	/api/1.1/roles	/api/1.2/roles
/dataserver	/api/1.1/servers	/api/1.2/servers
/dataserver/orderby/:field	/api/1.1/servers	/api/1.2/servers
/dataserverdetail/select/:hostname	/api/1.1/servers/hostname/:hostname/details.json	/api/1.2/servers/hostname/:hostname/details.json
/datastaticdnsentry	/api/1.1/staticdnsentries	/api/1.2/staticdnsentries
/datastatus	/api/1.1/statuses	/api/1.2/statuses
/datastatus/orderby/name	/api/1.1/statuses	/api/1.2/statuses
/datatype	/api/1.1/types	/api/1.2/types
/datatype/orderby/:field	/api/1.1/types	/api/1.2/types
/datauser	/api/1.1/users	/api/1.2/users
/datauser/orderby/:field	/api/1.1/users	/api/1.2/users
<i>Not Implemented</i>	<i>Not Implemented</i>	/api/1.2/servers/:hostname/configfiles/ats

## API 1.1 Reference

### ASN

#### /api/1.1/asns

#### GET /api/1.1/asns

Authentication Required: Yes

Role(s) Required: None

#### Response Properties

Parameter	Type	Description
asns	array	A collection of asns
>lastUpdated	string	The Time / Date this server entry was last updated
>id	string	Local unique identifier for the ASN
>asn	string	Autonomous System Numbers per APNIC for identifying a service provider.
>cachegroup	string	Related cachegroup name

**Response Example**

```
{
  "response": {
    "asns": [
      {
        "lastUpdated": "2012-09-17 21:41:22",
        "id": "27",
        "asn": "7015",
        "cachegroup": "us-ma-woburn"
      },
      {
        "lastUpdated": "2012-09-17 21:41:22",
        "id": "28",
        "asn": "7016",
        "cachegroup": "us-pa-pittsburgh"
      }
    ]
  },
}
```

**Cache Group****/api/1.1/cachegroups****GET /api/1.1/cachegroups**

Authentication Required: Yes

Role(s) Required: None

**Response Properties**

Parameter	Type	Description
id	string	Local unique identifier for the Cache Group
lastUpdated	string	The Time / Date this entry was last updated
latitude	string	Latitude for the Cache Group
longitude	string	Longitude for the Cache Group
name	string	The name of the Cache Group entry
parentCachegroupId	string	Parent cachegroup ID.
parentCachegroupName	string	Parent cachegroup name.
secondaryParentCachegroupId	string	Secondary parent cachegroup ID.
secondaryParentCachegroupName	string	Secondary parent cachegroup name.
shortName	string	Abbreviation of the Cache Group Name
typeId	string	Unique identifier for the 'Type' of Cache Group entry
typeName	string	The name of the type of Cache Group entry

### Response Example

```
{
  "response": [
    {
      "id": "21",
      "lastUpdated": "2012-09-25 20:27:28",
      "latitude": "0",
      "longitude": "0",
      "name": "dc-chicago",
      "parentCachegroupId": null,
      "parentCachegroupName": null,
      "secondaryParentCachegroupId": null,
      "secondaryParentCachegroupName": null,
      "shortName": "dcchi",
      "typeName": "MID_LOC",
      "typeId": "4"
    },
    {
      "id": "22",
      "lastUpdated": "2012-09-25 20:27:28",
      "latitude": "0",
      "longitude": "0",
      "name": "dc-chicago-1",
      "parentCachegroupId": null,
      "parentCachegroupName": null,
      "secondaryParentCachegroupId": null,
      "secondaryParentCachegroupName": null,
      "shortName": "dcchi",
      "typeName": "MID_LOC",
      "typeId": "4"
    }
  ],
}
```

GET /api/1.1/cachegroups/trimmed

Authentication Required: Yes

Role(s) Required: None

### Response Properties

Parameter	Type	Description
name	string	

### Response Example

```
{
  "response": [
    {
      "name": "dc-chicago"
    },
    {
      "name": "dc-cmc"
    }
  ],
}
```

## GET /api/1.1/cachegroups/:id

Authentication Required: Yes

Role(s) Required: None

### Response Properties

Parameter	Type	Description
id	string	Local unique identifier for the Cache Group
lastUpdated	string	The Time / Date this entry was last updated
latitude	string	Latitude for the Cache Group
longitude	string	Longitude for the Cache Group
name	string	The name of the Cache Group entry
parentCachegroupId	string	Parent cachegroup ID.
parentCachegroupName	string	Parent cachegroup name.
secondaryParentCachegroupId	string	Secondary parent cachegroup ID.
secondaryParentCachegroupName	string	Secondary parent cachegroup name.
shortName	string	Abbreviation of the Cache Group Name
typeId	string	Unique identifier for the 'Type' of Cache Group entry
typeName	string	The name of the type of Cache Group entry

### Response Example

```
{
  "response": [
    {
      "id": "21",
```

(continues on next page)

(continued from previous page)

```

    "lastUpdated": "2012-09-25 20:27:28",
    "latitude": "0",
    "longitude": "0",
    "name": "dc-chicago",
    "parentCachegroupId": null,
    "parentCachegroupName": null,
    "secondaryParentCachegroupId": null,
    "secondaryParentCachegroupName": null,
    "shortName": "dcchi",
    "typeName": "MID_LOC",
    "typeId": "4"
  }
],
}

```

**GET /api/1.1/cacheGroup/:parameter\_id/parameter**

Authentication Required: Yes

Role(s) Required: None

**Request Route Parameters**

Name	Required	Description
parameter_id	yes	

**Response Properties**

Parameter	Type	Description
cacheGroups	array	
>name	string	
>id	string	

**Response Example**

```

{
  "response": {
    "cacheGroups": [
      {
        "name": "dc-chicago",
        "id": "21"
      },
      {
        "name": "dc-cmc",
        "id": "22"
      }
    ]
  },
}

```

**GET /api/1.1/cacheGroupParameters**

Authentication Required: Yes

Role(s) Required: None

#### Response Properties

Parameter	Type	Description
cachegroupParameters	array	A collection of cache group parameters.
>parameter	string	
>last_updated	string	
>cachegroup	string	

#### Response Example

```
{
  "response": {
    "cachegroupParameters": [
      {
        "parameter": "379",
        "last_updated": "2013-08-05 18:49:37",
        "cachegroup": "us-ca-sanjose"
      },
      {
        "parameter": "380",
        "last_updated": "2013-08-05 18:49:37",
        "cachegroup": "us-ca-sanjose"
      },
      {
        "parameter": "379",
        "last_updated": "2013-08-05 18:49:37",
        "cachegroup": "us-ma-woburn"
      }
    ]
  },
}
```

#### GET /api/1.1/cachegroups/:parameter\_id/parameter/available

Authentication Required: Yes

Role(s) Required: None

#### Request Route Parameters

Name	Required	Description
parameter_id	yes	

#### Response Properties

Parameter	Type	Description
name		
id		



**Response Example**

```
{
  "response": [
    {
      "name": "dc-chicago",
      "id": "21"
    },
    {
      "name": "dc-cmc",
      "id": "22"
    }
  ],
}
```

**CDN****/api/1.1/cdns****GET /api/1.1/cdns**

Authentication Required: Yes

Role(s) Required: None

**Response Properties**

Parameter	Type	Description
id	string	CDN id.
name	string	CDN name.
dnssecEnabled	bool	DNSSEC enabled.
lastUpdated	string	

**Response Example**

```
{
  "response": [
    {
      "id": "1"
      "name": "over-the-top",
      "dnssecEnabled": false,
      "lastUpdated": "2014-10-02 08:22:43"
    },
    {
      "id": "2"
      "name": "cdn2",
      "dnssecEnabled": true,
      "lastUpdated": "2014-10-02 08:22:43"
    }
  ]
}
```

**GET /api/1.1/cdns/:id**

Authentication Required: Yes

Role(s) Required: None

**Request Route Parameters**

Name	Required	Description
id	yes	CDN id.

**Response Properties**

Parameter	Type	Description
id	string	CDN id.
name	string	CDN name.
dnssecEnabled	bool	DNSSEC enabled.
lastUpdated	string	

**Response Example**

```
{
  "response": [
    {
      "id": "2"
      "name": "cdn2",
      "dnssecEnabled": false,
      "lastUpdated": "2014-10-02 08:22:43"
    }
  ]
}
```

**GET /api/1.1/cdns/name/:name**

Authentication Required: Yes

Role(s) Required: None

**Request Route Parameters**

Name	Required	Description
name	yes	CDN name.

**Response Properties**

Parameter	Type	Description
id	string	CDN id.
name	string	CDN name.
dnssecEnabled	bool	DNSSEC enabled.
lastUpdated	string	

**Response Example**

```
{
  "response": [
    {
      "id": "2"
      "name": "cdn2",
      "dnssecEnabled": false,
      "lastUpdated": "2014-10-02 08:22:43"
    }
  ]
}
```

## Health

### GET /api/1.1/cdns/health

Retrieves the health of all locations (cache groups) for all CDNs.

Authentication Required: Yes

Role(s) Required: None

#### Response Properties

Parameter	Type	Description
totalOnline	int	Total number of online caches across all CDNs.
totalOffline	int	Total number of offline caches across all CDNs.
cachegroups	array	A collection of cache groups.
>online	int	The number of online caches for the cache group
>offline	int	The number of offline caches for the cache group.
>name	string	Cache group name.

#### Response Example

```
{
  "response": {
    "totalOnline": 148,
    "totalOffline": 0,
    "cachegroups": [
      {
        "online": 8,
        "offline": 0,
        "name": "us-co-denver"
      },
      {
        "online": 7,
        "offline": 0,
        "name": "us-de-newcastle"
      }
    ]
  },
}
```

**GET /api/1.1/cdns/:name/health**

Retrieves the health of all locations (cache groups) for a given CDN.

Authentication Required: Yes

Role(s) Required: None

**Request Route Parameters**

Name	Required	Description
name	yes	

**Response Properties**

Parameter	Type	Description
totalOnline	int	Total number of online caches across the specified CDN.
totalOffline	int	Total number of offline caches across the specified CDN.
cacheGroups	array	A collection of cache groups.
>online	int	The number of online caches for the cache group
>offline	int	The number of offline caches for the cache group.
>name	string	Cache group name.

**Response Example**

```
{
  "response": {
    "totalOnline": 148,
    "totalOffline": 0,
    "cacheGroups": [
      {
        "online": 8,
        "offline": 0,
        "name": "us-co-denver"
      },
      {
        "online": 7,
        "offline": 0,
        "name": "us-de-newcastle"
      }
    ]
  },
}
```

**GET /api/1.1/cdns/usage/overview**

Retrieves the high-level CDN usage metrics.

Authentication Required: Yes

Role(s) Required: None

### Response Properties

Parameter	Type	Description
currentGbps	number	
tps	int	
maxGbps	int	

### Response Example

```
{
  "response": {
    "currentGbps": 149.368167,
    "tps": 36805,
    "maxGbps": 3961
  }
}
```

## GET /api/1.1/cdns/capacity

Retrieves the aggregate capacity percentages of all locations (cache groups) for a given CDN.

Authentication Required: Yes

Role(s) Required: None

### Response Properties

Parameter	Type	Description
availablePercent	number	
unavailablePercent	number	
utilizedPercent	number	
maintenancePercent	number	

### Response Example

```
{
  "response": {
    "availablePercent": 89.0939840205533,
    "unavailablePercent": 0,
    "utilizedPercent": 10.9060020300395,
    "maintenancePercent": 0.0000139494071146245
  }
}
```

## Routing

### GET /api/1.1/cdns/routing

Retrieves the aggregate routing percentages of all locations (cache groups) for a given CDN.

Authentication Required: Yes

Role(s) Required: None

### Response Properties

Parameter	Type	Description
staticRoute	number	Used pre-configured DNS entries.
miss	number	No location available for client IP.
geo	number	Used 3rd party geo-IP mapping.
err	number	Error localizing client IP.
cz	number	Used Coverage Zone geo-IP mapping.
dsr	number	Overflow traffic sent to secondary CDN.

### Response Example

```
{
  "response": {
    "staticRoute": 0,
    "miss": 0,
    "geo": 37.8855391018869,
    "err": 0,
    "cz": 62.1144608981131,
    "dsr": 0
  }
}
```

## Metrics

### GET /api/1.1/cdns/metric\_types/:metric/start\_date/:start/end\_date/:end

Retrieves edge metrics of one or all locations (cache groups).

Authentication Required: Yes

Role(s) Required: None

### Request Route Parameters

Name	Required	Description
metric_type	yes	ooff, origin_tps
start	yes	UNIX time, yesterday, now
end	yes	UNIX time, yesterday, now

### Response Properties

Parameter	Type	Description
stats	hash	
>count	string	
>98thPercentile	string	
>min	string	
>max	string	
>5thPercentile	string	
>95thPercentile	string	
>mean	string	
>sum	string	
data	array	
>time	int	
>value	number	
label	string	

### Response Example

```
{
  "response": [
    {
      "stats": {
        "count": 1,
        "98thPercentile": 1668.03,
        "min": 1668.03,
        "max": 1668.03,
        "5thPercentile": 1668.03,
        "95thPercentile": 1668.03,
        "mean": 1668.03,
        "sum": 1668.03
      },
      "data": [
        [
          1425135900000,
          1668.03
        ],
        [
          1425136200000,
          null
        ]
      ],
      "label": "Origin TPS"
    }
  ],
}
```

## Domains

### GET /api/1.1/cdns/domains

Authentication Required: Yes

Role(s) Required: None

#### Response Properties

Parameter	Type	Description
profileId	string	
parameterId	string	
profileName	string	
profileDescription	string	
domainName	string	

#### Response Example

```
{
  "response": [
    {
      "profileId": "5",
      "parameterId": "404",
      "profileName": "CR_FOO",
      "profileDescription": "Content Router for foo.domain.net",
      "domainName": "foo.domain.net"
    },
    {
      "profileId": "8",
      "parameterId": "405",
      "profileName": "CR_BAR",
      "profileDescription": "Content Router for bar.domain.net",
      "domainName": "bar.domain.net"
    }
  ],
}
```

## Topology

### GET /api/1.1/cdns/:cdn\_name/configs

Retrieves CDN config information based upon the provided cdn name.

Authentication Required: Yes

Role(s) Required: None

#### Request Route Parameters

Name	Required	Description
cdn_name	yes	Your cdn name or, all

#### Response Properties



Parameter	Type	Description
id	string	
value	string	
name	string	
config_file	string	

**Response Example**

TBD

**GET /api/1.1/cdns/:name/configs/monitoring**

Retrieves CDN monitoring information.

Authentication Required: Yes

Role(s) Required: None

**Request Route Parameters**

Name	Required	Description
name	yes	CDN name

**Response Properties**

Parameter	Type	Description
trafficServers	array	A collection of Traffic Servers.
>profile	string	
>ip	string	
>status	string	
>cacheGroup	string	
>ip6	string	
>port	int	
>hostName	string	
>fqdn	string	
>interfaceName	string	
>type	string	
>hashId	string	
cacheGroups	array	A collection of cache groups.
>coordinates	hash	
>>longitude	number	
>>latitude	number	
>name	string	
config	hash	
>hack.ttl	int	
>tm.healthParams.polling.url	string	
>tm.dataServer.polling.url	string	
>health.timepad	int	

Continued on next page

Table 2 – continued from previous page

Parameter	Type	Description
>tm.polling.interval	int	
>health.threadPool	int	
>health.polling.interval	int	
>health.event-count	int	
>tm.crConfig.polling.url	number	
>CDN_name	number	
trafficMonitors	array	A collection of Traffic Monitors.
>profile	string	
>location	string	
>ip	string	
>status	string	
>ip6	string	
>port	int	
>hostName	string	
>fqdn	string	
deliveryServices	array	A collection of delivery services.
>xmlId	string	
>totalTpsThreshold	int	
>status	string	
>totalKbpsThreshold	int	
profiles	array	A collection of profiles.
>parameters	hash	
>>health.connection.timeout	int	
>>health.polling.url	string	
>>health.threshold.queryTime	int	
>>history.count	int	
>>health.threshold.availableBandwidthInKbps	string	
>>health.threshold.loadavg	string	
>name	string	
>type	string	

**Response Example**

TBD

**GET /api/1.1/cdns/:name/configs/routing**

Retrieves CDN routing information.

Authentication Required: Yes

Role(s) Required: None

**Request Route Parameters**

Name	Required	Description
name	yes	

**Response Properties**

Parameter	Type	Description
trafficServers	array	A collection of Traffic Servers.
>profile	string	
>ip	string	
>status	string	
>cacheGroup	string	
>ip6	string	
>port	int	
>deliveryServices	array	
>>xmlId	string	
>>remaps	array	
>>hostName	string	
>fqdn	string	
>interfaceName	string	
>type	string	
>hashId	string	
stats	hash	
>trafficOpsPath	string	
>cdnName	string	
>trafficOpsVersion	string	
>trafficOpsUser	string	
>date	int	
>trafficOpsHost	string	
cacheGroups	array	A collection of cache groups.
>coordinates	hash	
>>longitude	number	
>>latitude	number	
>name	string	
config	hash	
>tld.soa.admin	string	
>tcoveragezone.polling.interval	int	
>geolocation.polling.interval	int	
>tld.soa.expire	int	
>coveragezone.polling.url	string	
>tld.soa.minimum	int	
>geolocation.polling.url	string	
>domain_name	string	
>tld.ttls.AAAA	int	
>tld.soa.refresh	int	
>tld.ttls.NS	int	
>tld.ttls.SOA	int	
>geolocation6.polling.interval	int	
>tld.ttls.A	int	
>tld.soa.retry	int	
>geolocation6.polling.url	string	
trafficMonitors	array	A collection of Traffic Monitors.
>profile	string	
>location	string	
>ip	string	

Continued on next page

Table 3 – continued from previous page

Parameter	Type	Description
>status	string	
>ip6	string	
>port	int	
>hostName	string	
>fqdn	string	
deliveryServices	array	A collection of delivery services.
>xmlId	string	
>ttl	int	
>geoEnabled	string	
>coverageZoneOnly	boolean	
>matchSets	array	
>>protocol	string	
>>matchList	array	
>>>regex	string	
>>>matchType	string	
>bypassDestination	hash	
>>maxDnsIpsForLocation	int	
>>ttl	int	
>>type	string	
>ttls	hash	
>>A	int	
>>SOA	int	
>>NS	int	
>>AAAA	int	
>missCoordinates	hash	
>>longitude	number	
>>latitude	number	
>soa	hash	
>>admin	string	
>>retry	int	
>>minimum	int	
>>refresh	int	
>>expire	int	
trafficRouters	hash	
>profile	int	
>location	string	
>ip	string	
>status	string	
>ip6	string	
>port	int	
>hostName	string	
>fqdn	string	
>apiPort	int	

**Response Example**

:: TBD

## DNSSEC Keys

### GET /api/1.1/cdns/name/:name/dnsseckeys

Gets a list of dnsseckeys for CDN and all associated Delivery Services. Before returning response to user, check to make sure keys aren't expired. If they are expired, generate new ones. Before returning response to user, make sure dnssec keys for all delivery services exist. If they don't exist, create them.

Authentication Required: Yes

Role(s) Required: Admin

#### Request Route Parameters

Name	Required	Description
name	yes	

#### Response Properties

Parameter	Type	Description
cdn name/ds xml_id	string	identifier for ds or cdn
>zsk/ksk	array	collection of zsk/ksk data
>>ttl	string	time-to-live for dnssec requests
>>inceptionDate	string	epoch timestamp for when the keys were created
>>expirationDate	string	epoch timestamp representing the expiration of the keys
>>private	string	encoded private key
>>public	string	encoded public key
>>name	string	domain name
version	string	API version

#### Response Example

```
{
  "response": {
    "cdn1": {
      "zsk": {
        "ttl": "60",
        "inceptionDate": "1426196750",
        "private": "zsk private key",
        "public": "zsk public key",
        "expirationDate": "1428788750",
        "name": "foo.kabletown.com."
      },
      "ksk": {
        "name": "foo.kabletown.com.",
        "expirationDate": "1457732750",
        "public": "ksk public key",
        "private": "ksk private key",
        "inceptionDate": "1426196750",
        "ttl": "60"
      }
    },
    "ds-01": {
      "zsk": {
        "ttl": "60",
        "inceptionDate": "1426196750",
```

(continues on next page)

(continued from previous page)

```
    "private": "zsk private key",
    "public": "zsk public key",
    "expirationDate": "1428788750",
    "name": "ds-01.foo.kabletown.com."
  },
  "ksk": {
    "name": "ds-01.foo.kabletown.com.",
    "expirationDate": "1457732750",
    "public": "ksk public key",
    "private": "ksk private key",
    "inceptionDate": "1426196750"
  }
},
... repeated for each ds in the cdn
},
}
```

### GET /api/1.1/cdns/name/:name/dnsseckey/delete

Delete dnssec keys for a cdn and all associated delivery services.

Authentication Required: Yes

Role(s) Required: Admin

#### Request Route Parameters

Name	Required	Description
name	yes	name of the CDN for which you want to delete dnssec keys

#### Response Properties

Parameter	Type	Description
response	string	success response

#### Response Example

```
{
  "response": "Successfully deleted dnssec keys for <cdn>"
}
```

### POST /api/1.1/deliveryservices/dnsseckey/generate

Generates zsk and ksk keypairs for a cdn and all associated delivery services.

Authentication Required: Yes

Role(s) Required: Admin

### Request Properties

Parameter	Type	Description
key	string	name of the cdn
name	string	domain name of the cdn
ttl	string	time to live
kskExpirationDays	string	Expiration (in days) for the key signing keys
zskExpirationDays	string	Expiration (in days) for the zone signing keys

### Request Example

```
{
  "key": "cdn1",
  "name": "ott.kabletown.com",
  "ttl": "60",
  "kskExpirationDays": "365",
  "zskExpirationDays": "90"
}
```

### Response Properties

Parameter	Type	Description
response	string	response string
version	string	API version

### Response Example

```
{
  "response": "Successfully created dnssec keys for cdn1"
}
```

## Change Logs

### /api/1.1/logs

#### GET /api/1.1/logs.json

Authentication Required: Yes

Role(s) Required: None

#### Response Properties

Parameter	Type	Description
ticketNum	string	Optional field to cross reference with any bug tracking systems
level	string	Log categories for each entry, examples: 'UICHANGE', 'OPER', 'APICHANGE'.
lastUpdated	string	Local unique identifier for the Log
user	string	Current user who made the change that was logged
id	string	Local unique identifier for the Log entry
message	string	Log detail about what occurred

### Response Example

```
{
  "response": [
    {
      "ticketNum": null,
      "level": "OPER",
      "lastUpdated": "2015-02-04 22:59:13",
      "user": "userid852",
      "id": "22661",
      "message": "Snapshot CRConfig created."
    },
    {
      "ticketNum": null,
      "level": "APICHANGE",
      "lastUpdated": "2015-02-03 17:04:20",
      "user": "userid853",
      "id": "22658",
      "message": "Update server odol-atsec-nyc-23.kbaletown.net_
↪status=REPORTED"
    },
  ],
}
```

### GET /api/1.1/logs/:days/days.json

Authentication Required: Yes

Role(s) Required: None

#### Request Route Parameters

Name	Required	Description
days	yes	Number of days.

#### Response Properties

Parameter	Type	Description
ticketNum	string	
level	string	
lastUpdated	string	
user	string	
id	string	
message	string	

#### Response Example

```
{
  "response": [
    {
      "ticketNum": null,
      "level": "OPER",
      "lastUpdated": "2015-02-04 22:59:13",
```

(continues on next page)



(continued from previous page)

```
    "user": "userid852",
    "id": "22661",
    "message": "Snapshot CRConfig created."
  },
  {
    "ticketNum": null,
    "level": "APICHANGE",
    "lastUpdated": "2015-02-03 17:04:20",
    "user": "userid853",
    "id": "22658",
    "message": "Update server odol-atsec-nyc-23.kabletown.net_
↪status=REPORTED"
  }
],
}
```

### GET /api/1.1/logs/newcount.json

Authentication Required: Yes

Role(s) Required: None

#### Response Properties

Parameter	Type	Description
newLogcount	int	

#### Response Example

```
{
  "response": {
    "newLogcount": 0
  }
}
```

## Delivery Service

### /api/1.1/deliveryservices

#### GET /api/1.1/deliveryservices

Retrieves all delivery services. See also [Using Traffic Ops - Delivery Service](#).

Authentication Required: Yes

Role(s) Required: None

#### Response Properties

Parameter	Type	Description
active	bool	true if active, false if inactive.
cacheurl	string	Cache URL rule to apply to this delivery service.
ccrDnsTtl	string	The TTL of the DNS response for A or AAAA queries requesting the IP address of the tr. host.
cdnId	string	Id of the CDN to which the delivery service belongs to.
cdnName	string	Name of the CDN to which the delivery service belongs to.
checkPath	string	The path portion of the URL to check this deliveryservice for health.
displayName	string	The display name of the delivery service.
dnsBypassIp	string	The IPv4 IP to use for bypass on a DNS deliveryservice - bypass starts when serving more than the globalMaxMbps traffic on this deliveryservice.
dnsBypassIp6	string	The IPv6 IP to use for bypass on a DNS deliveryservice - bypass starts when serving more than the globalMaxMbps traffic on this deliveryservice.
dnsBypassTtl	string	The TTL of the DNS bypass response.
dscp	string	The Differentiated Services Code Point (DSCP) with which to mark downstream (EDGE -> customer) traffic.
edgeHeaderRewrite	string	The EDGE header rewrite actions to perform.
geoLimitRedirectUrl	string	
geoLimit	string	<ul style="list-style-type: none"> <li>• 0: None - no limitations</li> <li>• 1: Only route on CZF file hit</li> <li>• 2: Only route on CZF hit or when from USA</li> </ul> <p>Note that this does not prevent access to content or makes content secure; it just prevents routing to the content by Traffic Router.</p>
geoLimitCountries	string	
geoProvider	string	

Continued on next page

Table 4 – continued from previous page

Parameter	Type	Description
globalMaxMbps	string	The maximum global bandwidth allowed on this deliveryservice. If exceeded, the traffic routes to the dnsByPassIp* for DNS deliveryservices and to the httpBypassFqdn for HTTP deliveryservices.
globalMaxTps	string	The maximum global transactions per second allowed on this deliveryservice. When this is exceeded traffic will be sent to the dnsByPassIp* for DNS deliveryservices and to the httpBypassFqdn for HTTP deliveryservices
httpBypassFqdn	string	The HTTP destination to use for bypass on an HTTP deliveryservice - bypass starts when serving more than the globalMaxMbps traffic on this deliveryservice.
id	string	The deliveryservice id (database row number).
infoUrl	string	Use this to add a URL that points to more information about that deliveryservice.
initialDispersion	string	
ipv6RoutingEnabled	bool	false: send IPv4 address of Traffic Router to client on HTTP type del.
lastUpdated	string	
logsEnabled	bool	
longDesc	string	Description field 1.
longDesc1	string	Description field 2.
longDesc2	string	Description field 2.
>>type	string	The type of MatchList (one of :ref:to-api-v11-types use_in_table='regex').
>>setNumber	string	The set Number of the matchList.
>>pattern	string	The regexp for the matchList.
maxDnsAnswers	string	The maximum number of IPs to put in a A/AAAA response for a DNS deliveryservice (0 means all available).
midHeaderRewrite	string	The MID header rewrite actions to perform.
missLat	string	The latitude to use when the client cannot be found in the CZF or the Geo lookup.

Continued on next page

Table 4 – continued from previous page

Parameter	Type	Description
missLong	string	The longitude to use when the client cannot be found in the CZF or the Geo lookup.
multiSiteOrigin	bool	Is the Multi Site Origin feature enabled for this delivery service (0=false, 1=true). See <a href="#">Multi Site Origin</a>
multiSiteOriginAlgor	bool	Is the Multi Site Origin feature enabled for this delivery service (0=false, 1=true). See <a href="#">Multi Site Origin</a>
orgServerFqdn	string	The origin server base URL (FQDN when used in this instance, includes the protocol ( <a href="#">http://</a> or <a href="#">https://</a> ) for use in retrieving content from the origin server.
originShield	string	
profileDescription	string	The description of the Traffic Router Profile with which this deliveryservice is associated.
profileId	string	The id of the Traffic Router Profile with which this deliveryservice is associated.
profileName	string	The name of the Traffic Router Profile with which this deliveryservice is associated.
protocol	string	<ul style="list-style-type: none"> <li>• 0: serve with <a href="#">http://</a> at EDGE</li> <li>• 1: serve with <a href="#">https://</a> at EDGE</li> <li>• 2: serve with both <a href="#">http://</a> and <a href="#">https://</a> at EDGE</li> </ul>
qstringIgnore	string	<ul style="list-style-type: none"> <li>• 0: no special query string handling; it is for use in the cache-key and pass up to origin.</li> <li>• 1: ignore query string in cache-key, but pass it up to parent and or origin.</li> <li>• 2: drop query string at edge, and do not use it in the cache-key.</li> </ul>

Continued on next page

Table 4 – continued from previous page

Parameter	Type	Description
rangeRequestHandling	string	How to treat range requests: <ul style="list-style-type: none"> <li>• 0 Do not cache (ranges requested from files that are already cached due to a non range request will be a HIT)</li> <li>• 1 Use the <code>background_fetch</code> plugin.</li> <li>• 2 Use the <code>cache_range_requests</code> plugin.</li> </ul>
regexRemap	string	Regex Remap rule to apply to this delivery service at the Edge tier.
regionalGeoBlocking	bool	Regex Remap rule to apply to this delivery service at the Edge tier.
remapText	string	Additional raw remap line text.
signed	bool	<ul style="list-style-type: none"> <li>• false: token based auth (see <code>:ref:token-based-auth</code>) is not enabled for this deliveryservice.</li> <li>• true: token based auth is enabled for this deliveryservice.</li> </ul>
signingAlgorithm	string	<ul style="list-style-type: none"> <li>• null: token based auth (see <code>:ref:token-based-auth</code>) is not enabled for this deliveryservice.</li> <li>• “url_sig”: URL Sign token based auth is enabled for this deliveryservice.</li> <li>• “uri_signing”: URI Signing token based auth is enabled for this deliveryservice.</li> </ul>
sslKeyVersion	string	
trRequestHeaders	string	
trResponseHeaders	string	
type	string	The type of this deliveryservice (one of <code>:ref:to-api-v11-types</code> use <code>in_table='deliveryservice'</code> ).
typeId	string	The type of this deliveryservice (one of <code>:ref:to-api-v11-types</code> use <code>in_table='deliveryservice'</code> ).

Continued on next page

Table 4 – continued from previous page

Parameter	Type	Description
xmlId	string	Unique string that describes this deliveryservice.

**Response Example**

```
{
  "response": [
    {
      "active": true,
      "cacheurl": null,
      "ccrDnsTtl": "3600",
      "cdnId": "2",
      "cdnName": "over-the-top",
      "checkPath": "",
      "displayName": "My Cool Delivery Service",
      "dnsBypassCname": "",
      "dnsBypassIp": "",
      "dnsBypassIp6": "",
      "dnsBypassTtl": "30",
      "dscp": "40",
      "edgeHeaderRewrite": null,
      "exampleURLs": [
        "http://edge.foo-ds.foo.bar.net"
      ],
      "geoLimit": "0",
      "geoLimitCountries": null,
      "geoLimitRedirectURL": null,
      "geoProvider": "0",
      "globalMaxMbps": null,
      "globalMaxTps": "0",
      "httpBypassFqdn": "",
      "id": "442",
      "infoUrl": "",
      "initialDispersion": "1",
      "ipv6RoutingEnabled": true,
      "lastUpdated": "2016-01-26 08:49:35",
      "logsEnabled": false,
      "longDesc": "",
      "longDesc1": "",
      "longDesc2": "",
      "matchList": [
        {
          "pattern": ".*\\.foo-ds\\.\\.\\.*",
          "setNumber": "0",
          "type": "HOST_REGEX"
        }
      ],
      "maxDnsAnswers": "0",
      "midHeaderRewrite": null,
      "missLat": "41.881944",
      "missLong": "-87.627778",
      "multiSiteOrigin": false,
      "multiSiteOriginAlgorithm": null,
      "orgServerFqdn": "http://baz.boo.net",
      "originShield": null,
    }
  ]
}
```

(continues on next page)

(continued from previous page)

```

    "profileDescription": "Content Router for over-the-top",
    "profileId": "5",
    "profileName": "ROUTER_TOP",
    "protocol": "0",
    "qstringIgnore": "1",
    "rangeRequestHandling": "0",
    "regexRemap": null,
    "regionalGeoBlocking": false,
    "remapText": null,
    "signed": false,
    "signingAlgorithm": null,
    "sslKeyVersion": "0",
    "trRequestHeaders": null,
    "trResponseHeaders": "Access-Control-Allow-Origin: *",
    "type": "HTTP",
    "typeId": "8",
    "xmlId": "foo-ds"
  }
  { .. },
  { .. }
]
}

```

**GET /api/1.1/deliveryservices/:id**

Retrieves a specific delivery service. See also [Using Traffic Ops - Delivery Service](#).

Authentication Required: Yes

Role(s) Required: None

**Response Properties**

Parameter	Type	Description
active	bool	true if active, false if inactive.
cacheurl	string	Cache URL rule to apply to this delivery service.
ccrDnsTtl	string	The TTL of the DNS response for A or AAAA queries requesting the IP address of the tr. host.
cdnId	string	Id of the CDN to which the delivery service belongs to.
cdnName	string	Name of the CDN to which the delivery service belongs to.
checkPath	string	The path portion of the URL to check this deliveryservice for health.
displayName	string	The display name of the delivery service.

Continued on next page

Table 5 – continued from previous page

Parameter	Type	Description
<code>dnsBypassIp</code>	string	The IPv4 IP to use for bypass on a DNS deliveryservice - bypass starts when serving more than the <code>globalMaxMbps</code> traffic on this deliveryservice.
<code>dnsBypassIp6</code>	string	The IPv6 IP to use for bypass on a DNS deliveryservice - bypass starts when serving more than the <code>globalMaxMbps</code> traffic on this deliveryservice.
<code>dnsBypassTtl</code>	string	The TTL of the DNS bypass response.
<code>dscp</code>	string	The Differentiated Services Code Point (DSCP) with which to mark downstream (EDGE -> customer) traffic.
<code>edgeHeaderRewrite</code>	string	The EDGE header rewrite actions to perform.
<code>exampleURLs</code>	array	Entry points into the CDN for this deliveryservice.
<code>geoLimitRedirectUrl</code>	string	
<code>geoLimit</code>	string	<ul style="list-style-type: none"> <li>• 0: None - no limitations</li> <li>• 1: Only route on CZF file hit</li> <li>• 2: Only route on CZF hit or when from USA</li> </ul> <p>Note that this does not prevent access to content or makes content secure; it just prevents routing to the content by Traffic Router.</p>
<code>geoLimitCountries</code>	string	
<code>geoProvider</code>	string	
<code>globalMaxMbps</code>	string	The maximum global bandwidth allowed on this deliveryservice. If exceeded, the traffic routes to the <code>dnsByPassIp*</code> for DNS deliveryservices and to the <code>httpBypassFqdn</code> for HTTP deliveryservices.
<code>globalMaxTps</code>	string	The maximum global transactions per second allowed on this deliveryservice. When this is exceeded traffic will be sent to the <code>dnsByPassIp*</code> for DNS deliveryservices and to the <code>httpBypassFqdn</code> for HTTP deliveryservices

Continued on next page



Table 5 – continued from previous page

Parameter	Type	Description
httpBypassFqdn	string	The HTTP destination to use for bypass on an HTTP deliveryservice - bypass starts when serving more than the global-MaxMbps traffic on this deliveryservice.
id	string	The deliveryservice id (database row number).
infoUrl	string	Use this to add a URL that points to more information about that deliveryservice.
initialDispersion	string	
ipv6RoutingEnabled	bool	false: send IPv4 address of Traffic Router to client on HTTP type del.
lastUpdated	string	
logsEnabled	bool	
longDesc	string	Description field 1.
longDesc1	string	Description field 2.
longDesc2	string	Description field 2.
matchList	array	Array of matchList hashes.
>>type	string	The type of MatchList (one of :ref:to-api-v11-types use_in_table='regex').
>>setNumber	string	The set Number of the match-List.
>>pattern	string	The regexp for the matchList.
maxDnsAnswers	string	The maximum number of IPs to put in a A/AAAA response for a DNS deliveryservice (0 means all available).
midHeaderRewrite	string	The MID header rewrite actions to perform.
missLat	string	The latitude to use when the client cannot be found in the CZF or the Geo lookup.
missLong	string	The longitude to use when the client cannot be found in the CZF or the Geo lookup.
multiSiteOrigin	bool	Is the Multi Site Origin feature enabled for this delivery service (0=false, 1=true). See <a href="#">Multi Site Origin</a>
multiSiteOriginAlgor	bool	Is the Multi Site Origin feature enabled for this delivery service (0=false, 1=true). See <a href="#">Multi Site Origin</a>

Continued on next page

Table 5 – continued from previous page

Parameter	Type	Description
orgServerFqdn	string	The origin server base URL (FQDN when used in this instance, includes the protocol ( <a href="http://">http://</a> or <a href="https://">https://</a> ) for use in retrieving content from the origin server.
originShield	string	
profileDescription	string	The description of the Traffic Router Profile with which this deliveryservice is associated.
profileId	string	The id of the Traffic Router Profile with which this deliveryservice is associated.
profileName	string	The name of the Traffic Router Profile with which this deliveryservice is associated.
protocol	string	<ul style="list-style-type: none"> <li>• 0: serve with <a href="http://">http://</a> at EDGE</li> <li>• 1: serve with <a href="https://">https://</a> at EDGE</li> <li>• 2: serve with both <a href="http://">http://</a> and <a href="https://">https://</a> at EDGE</li> </ul>
qstringIgnore	string	<ul style="list-style-type: none"> <li>• 0: no special query string handling; it is for use in the cache-key and pass up to origin.</li> <li>• 1: ignore query string in cache-key, but pass it up to parent and or origin.</li> <li>• 2: drop query string at edge, and do not use it in the cache-key.</li> </ul>
rangeRequestHandling	string	<p>How to treat range requests:</p> <ul style="list-style-type: none"> <li>• 0 Do not cache (ranges requested from files taht are already cached due to a non range request will be a HIT)</li> <li>• 1 Use the <a href="#">background_fetch</a> plugin.</li> <li>• 2 Use the <a href="#">cache_range_requests</a> plugin.</li> </ul>
regexRemap	string	Regex Remap rule to apply to this delivery service at the Edge tier.

Continued on next page

Table 5 – continued from previous page

Parameter	Type	Description
regionalGeoBlocking	bool	Regex Remap rule to apply to this delivery service at the Edge tier.
remapText	string	Additional raw remap line text.
signed	bool	<ul style="list-style-type: none"> <li>• false: token based auth (see :ref:token-based-auth) is not enabled for this deliveryservice.</li> <li>• true: token based auth is enabled for this deliveryservice.</li> </ul>
signingAlgorithm	string	<ul style="list-style-type: none"> <li>• null: token based auth (see :ref:token-based-auth) is not enabled for this deliveryservice.</li> <li>• “url_sig”: URL Sign token based auth is enabled for this deliveryservice.</li> <li>• “uri_signing”: URI Signing token based auth is enabled for this deliveryservice.</li> </ul>
sslKeyVersion	string	
trRequestHeaders	string	
trResponseHeaders	string	
type	string	The type of this deliveryservice (one of :ref:to-api-v11-types use_in_table='deliveryservice').
typeId	string	The type of this deliveryservice (one of :ref:to-api-v11-types use_in_table='deliveryservice').
xmlId	string	Unique string that describes this deliveryservice.

**Response Example**

```
{
  "response": [
    {
      "active": true,
      "cacheurl": null,
      "ccrDnsTtl": "3600",
      "cdnId": "2",
      "cdnName": "over-the-top",
      "checkPath": "",
      "displayName": "My Cool Delivery Service",
      "dnsBypassCname": "",
      "dnsBypassIp": "",

```

(continues on next page)

(continued from previous page)

```

    "dnsBypassIp6": "",
    "dnsBypassTtl": "30",
    "dscp": "40",
    "edgeHeaderRewrite": null,
    "exampleURLs": [
        "http://edge.foo-ds.foo.bar.net"
    ],
    "geoLimit": "0",
    "geoLimitCountries": null,
    "geoLimitRedirectURL": null,
    "geoProvider": "0",
    "globalMaxMbps": null,
    "globalMaxTps": "0",
    "httpBypassFqdn": "",
    "id": "442",
    "infoUrl": "",
    "initialDispersion": "1",
    "ipv6RoutingEnabled": true,
    "lastUpdated": "2016-01-26 08:49:35",
    "logsEnabled": false,
    "longDesc": "",
    "longDesc1": "",
    "longDesc2": "",
    "matchList": [
        {
            "pattern": ".*\\.foo-ds\\.\\.\\.\\.\"",
            "setNumber": "0",
            "type": "HOST_REGEX"
        }
    ],
    "maxDnsAnswers": "0",
    "midHeaderRewrite": null,
    "missLat": "41.881944",
    "missLong": "-87.627778",
    "multiSiteOrigin": false,
    "multiSiteOriginAlgorithm": null,
    "orgServerFqdn": "http://baz.boo.net",
    "originShield": null,
    "profileDescription": "Content Router for over-the-top",
    "profileId": "5",
    "profileName": "ROUTER_TOP",
    "protocol": "0",
    "qstringIgnore": "1",
    "rangeRequestHandling": "0",
    "regexRemap": null,
    "regionalGeoBlocking": false,
    "remapText": null,
    "signed": false,
    "signingAlgorithm": null,
    "sslKeyVersion": "0",
    "trRequestHeaders": null,
    "trResponseHeaders": "Access-Control-Allow-Origin: *",
    "type": "HTTP",
    "typeId": "8",
    "xmlId": "foo-ds"
}
]

```

(continues on next page)

(continued from previous page)

```
}
```

## Health

### GET /api/1.1/deliveryservices/:id/state.json

Retrieves the failover state for a delivery service.

Authentication Required: Yes

Role(s) Required: None

#### Response Properties

Parameter	Type	Description
failover	hash	
>locations	array	
>destination	string	
>configured	boolean	
>enabled	boolean	
enabled	boolean	

#### Response Example

```
{
  "response": {
    "failover": {
      "locations": [ ],
      "destination": null,
      "configured": false,
      "enabled": false
    },
    "enabled": true
  }
}
```

### GET /api/1.1/deliveryservices/:id/health.json

Retrieves the health of all locations (cache groups) for a delivery service.

Authentication Required: Yes

Role(s) Required: None

#### Response Properties

Parameter	Type	Description
totalOnline	int	Total number of online caches across all CDNs.
totalOffline	int	Total number of offline caches across all CDNs.
cachegroups	array	A collection of cache groups.
>online	int	The number of online caches for the cache group
>offline	int	The number of offline caches for the cache group.
>name	string	Cache group name.

### Response Example

```
{
  "response": {
    "totalOnline": 148,
    "totalOffline": 0,
    "cachegroups": [
      {
        "online": 8,
        "offline": 0,
        "name": "us-co-denver"
      },
      {
        "online": 7,
        "offline": 0,
        "name": "us-de-newcastle"
      }
    ]
  }
}
```

### GET /api/1.1/deliveryservices/:id/capacity.json

Retrieves the capacity percentages of a delivery service.

Authentication Required: Yes

Role(s) Required: None

#### Request Route Parameters

Name	Required	Description
id	yes	delivery service id.

### Response Properties

Parameter	Type	Description
availablePercent	number	The percentage of server capacity assigned to the delivery service that is available.
unavailablePercent	number	The percentage of server capacity assigned to the delivery service that is unavailable.
utilizedPercent	number	The percentage of server capacity assigned to the delivery service being used.
maintenancePercent	number	The percentage of server capacity assigned to the delivery service that is down for maintenance.

**Response Example**

```
{
  "response": {
    "availablePercent": 89.0939840205533,
    "unavailablePercent": 0,
    "utilizedPercent": 10.9060020300395,
    "maintenancePercent": 0.0000139494071146245
  },
}
```

**GET /api/1.1/deliveryservices/:id/routing.json**

Retrieves the routing method percentages of a delivery service.

Authentication Required: Yes

Role(s) Required: None

**Request Route Parameters**

Name	Required	Description
id	yes	delivery service id.

**Response Properties**

Parameter	Type	Description
staticRoute	number	The percentage of Traffic Router responses for this deliveryservice satisfied with pre-configured DNS entries.
miss	number	The percentage of Traffic Router responses for this deliveryservice that were a miss (no location available for client IP).
geo	number	The percentage of Traffic Router responses for this deliveryservice satisfied using 3rd party geo-IP mapping.
err	number	The percentage of Traffic Router requests for this deliveryservice resulting in an error.
cz	number	The percentage of Traffic Router requests for this deliveryservice satisfied by a CZF hit.
dsr	number	The percentage of Traffic Router requests for this deliveryservice satisfied by sending the client to the overflow CDN.

### Response Example

```
{
  "response": {
    "staticRoute": 0,
    "miss": 0,
    "geo": 37.8855391018869,
    "err": 0,
    "cz": 62.1144608981131,
    "dsr": 0
  },
}
```

## Metrics

**GET /api/1.1/deliveryservices/:id/server\_types/:type/metric\_types/start\_date/:start/end\_date/:end.json**

Retrieves detailed and summary metrics for MIDs or EDGEs for a delivery service.

Authentication Required: Yes

Role(s) Required: None

### Request Route Parameters

Name	Re-quired	Description
id	yes	The delivery service id.
server_types	yes	EDGE or MID.
metric_types	yes	One of the following: “kbps”, “tps”, “tps_2xx”, “tps_3xx”, “tps_4xx”, “tps_5xx”.
start_date	yes	UNIX time
end_date	yes	UNIX time

### Request Query Parameters

Name	Required	Description
stats	no	Flag used to return only summary metrics

### Response Properties



Parameter	Type	Description
stats	hash	
>>count	int	
>>98thPercentile	number	
>>min	number	
>>max	number	
>>5thPercentile	number	
>>95thPercentile	number	
>>median	number	
>>mean	number	
>>stddev	number	
>>sum	number	
data	array	
>>item	array	
>>time	number	
>>value	number	
label	string	

### Response Example

```
{
  "response": [
    {
      "stats": {
        "count": 988,
        "98thPercentile": 16589105.55958,
        "min": 3185442.975,
        "max": 17124754.257,
        "5thPercentile": 3901253.95445,
        "95thPercentile": 16013210.034,
        "median": 8816895.576,
        "mean": 8995846.31741194,
        "stddev": 3941169.83683573,
        "sum": 333296106.060112
      },
      "data": [
        [
          1414303200000,
          12923518.466
        ],
        [
          1414303500000,
          12625139.65
        ]
      ],
      "label": "MID Kbps"
    }
  ],
}
```

## Server

GET /api/1.1/deliveryserviceserver.json

Authentication Required: Yes

Role(s) Required: Yes

**Request Query Parameters**

Name	Required	Description
page	no	The page number for use in pagination.
limit	no	For use in limiting the result set.

**Response Properties**

Parameter	Type	Description
lastUpdated	array	
server	string	
deliveryService	string	

**Response Example**

```
{
  "page": 2,
  "orderby": "deliveryservice",
  "response": [
    {
      "lastUpdated": "2014-09-26 17:53:43",
      "server": "20",
      "deliveryService": "1"
    },
    {
      "lastUpdated": "2014-09-26 17:53:44",
      "server": "21",
      "deliveryService": "1"
    }
  ],
  "limit": 2
}
```

**SSL Keys****GET /api/1.1/deliveryservices/xmlId/:xmlid/sslkeys.json**

Authentication Required: Yes

Role(s) Required: Portal

**Request Route Parameters**

Name	Required	Description
xmlId	yes	xml_id of the desired delivery service

**Request Query Parameters**

Name	Required	Description
version	no	The version number to retrieve

### Response Properties

Parameter	Type	Description
crt	string	base64 encoded crt file for delivery service
csr	string	base64 encoded csr file for delivery service
key	string	base64 encoded private key file for delivery service
businessUnit	string	The business unit entered by the user when generating certs. Field is optional and if not provided by the user will not be in response
city	string	The city entered by the user when generating certs. Field is optional and if not provided by the user will not be in response
organization	string	The organization entered by the user when generating certs. Field is optional and if not provided by the user will not be in response
hostname	string	The hostname entered by the user when generating certs. Field is optional and if not provided by the user will not be in response
country	string	The country entered by the user when generating certs. Field is optional and if not provided by the user will not be in response
state	string	The state entered by the user when generating certs. Field is optional and if not provided by the user will not be in response
version	string	The version of the certificate record in Riak

### Response Example

```
{
  "response": {
    "certificate": {
      "crt": "crt",
      "key": "key",
      "csr": "csr"
    },
    "businessUnit": "CDN_Eng",
    "city": "Denver",
    "organization": "KableTown",
    "hostname": "foober.com",
    "country": "US",
    "state": "Colorado",
    "version": "1"
  }
}
```

### GET /api/1.1/deliveryservices/hostname/:hostname/sslkeys.json

Authentication Required: Yes

Role Required: Admin

### Request Route Parameters

Name	Required	Description
hostname	yes	pristine hostname of the desired delivery service

### Request Query Parameters

Name	Required	Description
version	no	The version number to retrieve

### Response Properties

Parameter	Type	Description
crt	string	base64 encoded crt file for delivery service
csr	string	base64 encoded csr file for delivery service
key	string	base64 encoded private key file for delivery service
businessUnit	string	The business unit entered by the user when generating certs. Field is optional and if not provided by the user will not be in response
city	string	The city entered by the user when generating certs. Field is optional and if not provided by the user will not be in response
organization	string	The organization entered by the user when generating certs. Field is optional and if not provided by the user will not be in response
hostname	string	The hostname entered by the user when generating certs. Field is optional and if not provided by the user will not be in response
country	string	The country entered by the user when generating certs. Field is optional and if not provided by the user will not be in response
state	string	The state entered by the user when generating certs. Field is optional and if not provided by the user will not be in response
version	string	The version of the certificate record in Riak

### Response Example

```
{
  "response": {
    "certificate": {
      "crt": "crt",
      "key": "key",
      "csr": "csr"
    },
    "businessUnit": "CDN_Eng",
    "city": "Denver",
    "organization": "KableTown",
    "hostname": "foober.com",
    "country": "US",
    "state": "Colorado",
    "version": "1"
  }
}
```

GET /api/1.1/deliveryservices/xmlId:xmlid/sslkeys/delete.json

Authentication Required: Yes

Role Required: PORTAL

### Request Route Parameters

Name	Required	Description
xmlId	yes	xml_id of the desired delivery service

### Request Query Parameters

Name	Required	Description
version	no	The version number to retrieve

### Response Properties

Parameter	Type	Description
response	string	success response

### Response Example

```
{
  "response": "Successfully deleted ssl keys for <xml_id>"
}
```

## POST /api/1.1/deliveryservices/sslkeys/generate

Generates SSL crt, csr, and private key for a delivery service

Authentication Required: Yes

Role(s) Required: Portal

### Request Properties

Parameter	Type	Description
key	string	xml_id of the delivery service
version	string	version of the keys being generated
hostname	string	the <i>pristine hostname</i> of the delivery service
country	string	
state	string	
city	string	
org	string	
unit	boolean	

### Request Example

```
{
  "key": "ds-01",
  "businessUnit": "CDN Engineering",

```

(continues on next page)

(continued from previous page)

```
"version": "3",
"hostname": "tr.ds-01.ott.kabletown.com",
"certificate": {
  "key": "some_key",
  "csr": "some_csr",
  "crt": "some_crt"
},
"country": "US",
"organization": "Kabletown",
"city": "Denver",
"state": "Colorado"
}
```

### Response Properties

Parameter	Type	Description
response	string	response string
version	string	API version

### Response Example

```
{
  "response": "Successfully created ssl keys for ds-01"
}
```

## POST /api/1.1/deliveryservices/sslkeys/add

Allows user to add SSL crt, csr, and private key for a delivery service.

Authentication Required: Yes

Role(s) Required: Portal

### Request Properties

Parameter	Type	Description
key	string	xml_id of the delivery service
version	string	version of the keys being generated
csr	string	
crt	string	
key	string	

### Request Example

```
{
  "key": "ds-01",
  "version": "1",
  "certificate": {
    "key": "some_key",
    "csr": "some_csr",
    "crt": "some_crt"
  }
}
```

### Response Properties

Parameter	Type	Description
response	string	response string
version	string	API version

### Response Example

```
{
  "response": "Successfully added ssl keys for ds-01"
}
```

## POST /api/1.1/deliveryservices/request

Allows a user to send delivery service request details to a specified email address.

Authentication Required: Yes

Role(s) Required: None

### Request Properties

Parameter	Type	Re- quired	Description
emailTo	string	yes	The email to which the delivery service request will be sent.
details	hash	yes	Parameters for the delivery service request.
>customer	string	yes	Name of the customer to associated with the delivery service.
>deliveryProtocol	string	yes	Eg. http or http/https
>routingType	string	yes	Eg. DNS or HTTP Redirect
>serviceDesc	string	yes	A description of the delivery service.
>peakBPSEstimate	string	yes	Used to manage cache efficiency and plan for capacity.
>peakTPSEstimate	string	yes	Used to manage cache efficiency and plan for capacity.
>maxLibrarySizeEstimate	string	yes	Used to manage cache efficiency and plan for capacity.
>originURL	string	yes	The URL path to the origin server.
>hasOriginDynamicRemap	bool	yes	This is a feature which allows services to use multiple origin URLs for the same service.
>originTestFile	string	yes	A URL path to a test file available on the origin server.
>hasOriginACLWhitelist	bool	yes	Is access to your origin restricted using an access control list (ACL or whitelist) of Ips?
>originHeaders	string	no	Header values that must be passed to requests to your origin.
>otherOriginSecurity	string	no	Other origin security measures that need to be considered for access.
>queryStringHandling	string	yes	How to handle query strings that come with the request.
>rangeRequestHandling	string	yes	How to handle range requests.
>hasSignedURLs	bool	yes	Are Urls signed?
>hasNegativeCachingCustomization	bool	yes	Any customization required for negative caching?
>negativeCachingCustomizationNotes	string	yes	Negative caching customization instructions.
>serviceAliases	array	no	Service aliases which will be used for this service.
>rateLimitingGBPS	int	no	Rate Limiting - Bandwidth (Gigabits per second)
>rateLimitingTPS	int	no	Rate Limiting - Transactions/Second
>overflowService	string	no	An overflow point (URL or IP address) used if rate limits are met.
>headerRewriteEdge	string	no	Headers can be added or altered at each layer of the CDN.
>headerRewriteMid	string	no	Headers can be added or altered at each layer of the CDN.
>headerRewriteRedirect	string	no	Headers can be added or altered at each layer of the CDN.
>notes	string	no	Additional instructions to provide the delivery service provisioning team.

### Request Example

```
{
  "emailTo": "foo@bar.com",
```

(continues on next page)



(continued from previous page)

```

"details": {
  "customer": "XYZ Corporation",
  "contentType": "video-on-demand",
  "deliveryProtocol": "http",
  "routingType": "dns",
  "serviceDesc": "service description goes here",
  "peakBPSEstimate": "less-than-5-Gbps",
  "peakTPSEstimate": "less-than-1000-TPS",
  "maxLibrarySizeEstimate": "less-than-200-GB",
  "originURL": "http://myorigin.com",
  "hasOriginDynamicRemap": false,
  "originTestFile": "http://myorigin.com/crossdomain.xml",
  "hasOriginACLWhitelist": true,
  "originHeaders": "",
  "otherOriginSecurity": "",
  "queryStringHandling": "ignore-in-cache-key-and-pass-up",
  "rangeRequestHandling": "range-requests-not-used",
  "hasSignedURLs": true,
  "hasNegativeCachingCustomization": true,
  "negativeCachingCustomizationNote": "negative caching instructions",
  "serviceAliases": [
    "http://alias1.com",
    "http://alias2.com"
  ],
  "rateLimitingGBPS": 50,
  "rateLimitingTPS": 5000,
  "overflowService": "http://overflowcdn.com",
  "headerRewriteEdge": "",
  "headerRewriteMid": "",
  "headerRewriteRedirectRouter": "",
  "notes": ""
}
}

```

## Response Properties

Parameter	Type	Description
alerts	array	A collection of alert messages.
>level	string	Success, info, warning or error.
>text	string	Alert message.
version	string	

## Response Example

```

{
  "alerts": [
    {
      "level": "success",
      "text": "Delivery Service request sent to foo@bar.com."
    }
  ]
}

```

(continues on next page)

(continued from previous page)

```
]
}
```

## Hardware Info

### /api/1.1/hwinfo

#### GET /api/1.1/hwinfo.json

Authentication Required: Yes

Role(s) Required: None

#### Response Properties

Parameter	Type	Description
serverId	string	Local unique identifier for this specific server's hardware info
serverHostName	string	Hostname for this specific server's hardware info
lastUpdated	string	The Time and Date for the last update for this server.
val	string	Freeform value used to track anything about a server's hardware info
description	string	Freeform description for this specific server's hardware info

#### Response Example

```
{
  "response": [
    {
      "serverId": "odol-atsmid-cen-09",
      "lastUpdated": "2014-05-27 09:06:02",
      "val": "D1S4",
      "description": "Physical Disk 0:1:0"
    },
    {
      "serverId": "odol-atsmid-cen-09",
      "lastUpdated": "2014-05-27 09:06:02",
      "val": "D1S4",
      "description": "Physical Disk 0:1:1"
    }
  ],
}
```

## Parameter

### /api/1.1/parameters

#### GET /api/1.1/parameters.json

Authentication Required: Yes

Role(s) Required: None

### Response Properties

Parameter	Type	Description
last_updated	string	The Time / Date this server entry was last updated
secure	boolean	When true, the parameter is accessible only by admin users. Defaults to false.
value	string	The parameter value, only visible to admin if secure is true
name	string	The parameter name
config_file	string	The parameter config_file

### Response Example

```
{
  "response": [
    {
      "last_updated": "2012-09-17 21:41:22",
      "secure": 0,
      "value": "foo.bar.net",
      "name": "domain_name",
      "config_file": "FooConfig.xml"
    },
    {
      "last_updated": "2012-09-17 21:41:22",
      "secure": 0,
      "value": "0,1,2,3,4,5,6",
      "name": "Drive_Letters",
      "config_file": "storage.config"
    },
    {
      "last_updated": "2012-09-17 21:41:22",
      "secure": 0,
      "value": "STRING __HOSTNAME__",
      "name": "CONFIG proxy.config.proxy_name",
      "config_file": "records.config"
    }
  ],
}
```

### GET /api/1.1/parameters/profile/:name.json

Authentication Required: Yes

Role(s) Required: None

### Request Route Parameters

Name	Required	Description
profile_name	yes	

### Response Properties

Parameter	Type	Description
last_updated	string	The Time / Date this server entry was last updated
secure	boolean	When true, the parameter is accessible only by admin users. Defaults to false.
value	string	The parameter value, only visible to admin if secure is true
name	string	The parameter name
config_file	string	The parameter config_file

### Response Example

```
{
  "response": [
    {
      "last_updated": "2012-09-17 21:41:22",
      "secure": 0,
      "value": "foo.bar.net",
      "name": "domain_name",
      "config_file": "FooConfig.xml"
    },
    {
      "last_updated": "2012-09-17 21:41:22",
      "secure": 0,
      "value": "0,1,2,3,4,5,6",
      "name": "Drive_Letters",
      "config_file": "storage.config"
    },
    {
      "last_updated": "2012-09-17 21:41:22",
      "secure": 0,
      "value": "STRING __HOSTNAME__",
      "name": "CONFIG proxy.config.proxy_name",
      "config_file": "records.config"
    }
  ],
}
```

## Physical Location

### /api/1.1/phys\_locations

#### GET /api/1.1/phys\_locations

Authentication Required: Yes

Role(s) Required: None

#### Response Properties

Parameter	Type	Description
address	string	
city	string	
comments	string	
email	string	
id	string	
lastUpdated	string	
name	string	
phone	string	
poc	string	
region	string	
regionId	string	
shortName	string	
state	string	
zip	string	

### Response Example

```
{
  "response": [
    {
      "region": "Mile High",
      "region": "4",
      "poc": "Jane Doe",
      "lastUpdated": "2014-10-02 08:22:43",
      "name": "Albuquerque",
      "comments": "Albuquerque",
      "phone": "(123) 555-1111",
      "state": "NM",
      "email": "jane.doe@email.com",
      "city": "Albuquerque",
      "zip": "87107",
      "id": "2",
      "address": "123 East 3rd St",
      "shortName": "Albuquerque"
    },
    {
      "region": "Mile High",
      "region": "4",
      "poc": "Jane Doe",
      "lastUpdated": "2014-10-02 08:22:43",
      "name": "Albuquerque",
      "comments": "Albuquerque",
      "phone": "(123) 555-1111",
      "state": "NM",
      "email": "jane.doe@email.com",
      "city": "Albuquerque",
      "zip": "87107",
      "id": "2",
      "address": "123 East 3rd St",
      "shortName": "Albuquerque"
    }
  ]
}
```

**GET /api/1.1/phys\_locations/trimmed.json**

Authentication Required: Yes

Role(s) Required: None

**Response Properties**

Parameter	Type	Description
name	string	

**Response Example**

```
{
  "response": [
    {
      "name": "Albuquerque"
    },
    {
      "name": "Ashburn"
    }
  ]
}
```

**GET /api/1.1/phys\_locations/:id**

Authentication Required: Yes

Role(s) Required: None

**Request Route Parameters**

Name	Required	Description
id	yes	Physical location ID.

**Response Properties**

Parameter	Type	Description
address	string	
city	string	
comments	string	
email	string	
id	string	
lastUpdated	string	
name	string	
phone	string	
poc	string	
region	string	
regionId	string	
shortName	string	
state	string	
zip	string	

### Response Example

```
{
  "response": [
    {
      "region": "Mile High",
      "region": "4",
      "poc": "Jane Doe",
      "lastUpdated": "2014-10-02 08:22:43",
      "name": "Albuquerque",
      "comments": "Albuquerque",
      "phone": "(123) 555-1111",
      "state": "NM",
      "email": "jane.doe@email.com",
      "city": "Albuquerque",
      "zip": "87107",
      "id": "2",
      "address": "123 East 3rd St",
      "shortName": "Albuquerque"
    }
  ]
}
```

**PUT /api/1.1/phys\_locations/:id** Update a physical location

Authentication Required: Yes

Role(s) Required: admin or oper

#### Request Route Parameters

Name	Type	Description
id	int	Physical location id.

#### Request Properties

Parameter	Required	Description
address	yes	Physical location address.
city	yes	Physical location city.
comments	no	Physical location comments.
email	no	Physical location email.
name	yes	Physical location name.
phone	no	Physical location phone.
poc	no	Physical location point of contact.
regionId	no	Physical location region ID.
shortName	yes	Physical location short name.
state	yes	Physical location state.
zip	yes	Physical location zip.

### Request Example

```
{
  "regionId": "1",
  "poc": "Jane Doesssss",
  "name": "Albuquerque",
  "comments": "Albuquerque",
  "phone": "(123) 555-1111",
  "state": "NM",
  "email": "jane.doe@email.com",
  "city": "Albuquerque",
  "zip": "87107",
  "address": "123 East 9rd St",
  "shortName": "Albuquerque"
}
```

### Response Properties

Parameter	Type	Description
address	string	
city	string	
comments	string	
email	string	
id	string	
lastUpdated	string	
name	string	
phone	string	
poc	string	
region	string	
regionId	string	
shortName	string	
state	string	
zip	string	

### Response Example



```
{
  "alerts": [
    {
      "level": "success",
      "text": "Physical location update was successful."
    }
  ],
  "response": [
    {
      "region": "Mile High",
      "region": "4",
      "poc": "Jane Doe",
      "lastUpdated": "2014-10-02 08:22:43",
      "name": "Albuquerque",
      "comments": "Albuquerque",
      "phone": "(123) 555-1111",
      "state": "NM",
      "email": "jane.doe@email.com",
      "city": "Albuquerque",
      "zip": "87107",
      "id": "2",
      "address": "123 East 3rd St",
      "shortName": "Albuquerque"
    }
  ]
}
```

**POST /api/1.1/regions/:region\_name/phys\_locations** Create physical location.

Authentication Required: Yes

Role(s) Required: admin or oper

region\_name: the name of the region to create physical location into.

#### Request Route Parameters

Name	Required	Description
region_name	yes	The name of the physical location

#### Request Properties

Parameter	Required	Description
name	yes	The name of the location
shortName	yes	The short name of the location
address	yes	
city	yes	
state	yes	
zip	yes	
phone	no	
poc	no	Point of contact
email	no	
comments	no	

### Request Example

```
{
  "name" : "my physical location1",
  "shortName" : "myphylocation1",
  "address" : "",
  "city" : "Shanghai",
  "state": "SH",
  "zip": "200000",
  "comments": "this is physical location1"
}
```

### Response Properties

Parameter	Type	Description
id	string	The id of the physical location created.
name	string	The name of the location
shortName	string	The short name of the location
regionName	string	The region name the physical location belongs to.
regionId	string	
address	string	
city	string	
state	string	
zip	string	
phone	string	
poc	string	Point of contact
email	string	
comments	string	

### Response Example

```
{
  "response": {
    'shortName': 'myphylocati',
    'regionName': 'myregion1',
    'name': 'my physical location1',
    'poc': '',
    'phone': '',
    'comments': 'this is physical location1',
    'state': 'SH',
    'email': '',
    'zip': '20000',
    'region_id': '20',
    'city': 'Shanghai',
    'address': '',
    'id': '200'
  }
}
```

## Profiles

### /api/1.1/profiles

#### GET /api/1.1/profiles

Authentication Required: Yes

Role(s) Required: None

##### Response Properties

Parameter	Type	Description
lastUpdated	array	The Time / Date this server entry was last updated
name	string	The name for the profile
id	string	Primary key
description	string	The description for the profile

##### Response Example

```
{
  "response": [
    {
      "lastUpdated": "2012-10-08 19:34:45",
      "name": "CCR_TOP",
      "id": "8",
      "description": "Content Router for top.foobar.net"
    }
  ]
}
```

#### GET /api/1.1/profiles/trimmed

Authentication Required: Yes

Role(s) Required: None

##### Response Properties

Parameter	Type	Description
name	string	The name for the profile

##### Response Example

```
{
  "response": [
    {
      "name": "CCR_TOP"
    }
  ]
}
```

**GET /api/1.1/profiles/:id**

Authentication Required: Yes

Role(s) Required: None

**Request Route Parameters**

Parameter	Required	Description
id	yes	The ID of the profile.

**Response Properties**

Parameter	Type	Description
lastUpdated	array	The Time / Date this server entry was last updated
name	string	The name for the profile
id	string	Primary key
description	string	The description for the profile

**Response Example**

```
{
  "response": [
    {
      "lastUpdated": "2012-10-08 19:34:45",
      "name": "CCR_TOP",
      "id": "8",
      "description": "Content Router for top.foobar.net"
    }
  ]
}
```

**Regions****/api/1.1/regions****GET /api/1.1/regions**

Authentication Required: Yes

Role(s) Required: None

**Response Properties**

Parameter	Type	Description
id	string	Region ID.
name	string	Region name.
division	string	Division ID.
divisionName	string	Division name.

**Response Example**

```
{
  "response": [
    {
      "id": "6",
      "name": "Atlanta",
      "division": "2",
      "divisionName": "West"
    },
    {
      "id": "7",
      "name": "Denver",
      "division": "2",
      "divisionName": "West"
    }
  ]
}
```

**GET /api/1.1/regions/:id**

Authentication Required: Yes

Role(s) Required: None

**Request Route Parameters**

Name	Required	Description
id	yes	Region id.

**Response Properties**

Parameter	Type	Description
id	string	Region ID.
name	string	Region name.
division	string	Division ID.
divisionName	string	Division name.

**Response Example**

```
{
  "response": [
    {
      "id": "6",
      "name": "Atlanta",
      "division": "2",
      "divisionName": "West"
    }
  ]
}
```

**PUT /api/1.1/regions/:id** Update a region

Authentication Required: Yes

Role(s) Required: admin or oper

**Request Route Parameters**

Name	Type	Description
id	int	Region id.

### Request Properties

Parameter	Required	Description
name	yes	The name of the region
division	yes	The division Id

### Request Example

```
{
  "name": "myregion1",
  "division": "4"
}
```

### Response Properties

Parameter	Type	Description
division	string	
divisionName	string	
name	string	
id	string	
lastUpdated	string	

### Response Example

```
{
  "alerts": [
    {
      "level": "success",
      "text": "Region update was successful."
    }
  ],
  "response": {
    "id": "1",
    "lastUpdated": "2014-03-18 08:57:39",
    "name": "myregion1",
    "division": "4",
    "divisionName": "mydivision1"
  }
}
```

### POST /api/1.1/divisions/:division\_name/regions Create Region

Authentication Required: Yes

Role(s) Required: admin or oper

division\_name - The name of division to create new region into.

**\*\* Request Route Parameters\*\***

Name	Required	Description
division_name	yes	The name of division will create new region in

### Request Properties

Parameter	Required	Description
name	yes	The name of the region

### Request Example

```
{  
  "name": "myregion1",  
}
```

### Response Properties

Parameter	Type	Description
name	string	name of region created
id	string	id of region created
divisionName	string	the division name the region belongs to.
divisionId	string	the id of division the region belongs to.

### Response Example

```
{  
  "response": {  
    'divisionName': 'mydivision1',  
    'divisionId': '4',  
    'name': 'myregion1',  
    'id': '19'  
  }  
}
```

## Roles

### /api/1.1/roles

#### GET /api/1.1/roles.json

Authentication Required: Yes

Role(s) Required: None

#### Response Properties

Parameter	Type	Description
name	string	
id	string	
privLevel	string	
description	string	

### Response Example

```
{
  "response": [
    {
      "name": "read-only",
      "id": "2",
      "privLevel": "10",
      "description": "read-only user"
    }
  ],
}
```

## Server

### /api/1.1/servers

#### GET /api/1.1/servers

Retrieves properties of CDN servers.

Authentication Required: Yes

Role(s) Required: None

#### Request Query Parameters

Name	Required	Description
dsId	no	Used to filter servers by delivery service.
status	no	Used to filter servers by status.
type	no	Used to filter servers by type.

#### Response Properties

Parameter	Type	Description
cachegroup	string	The cache group name (see <i>Cache Group</i> ).
cachegroupId	string	The cache group id.
cdnId	string	Id of the CDN to which the server belongs to.
cdnName	string	Name of the CDN to which the server belongs to.
domainName	string	The domain name part of the FQDN of the cache.
guid	string	An identifier used to uniquely identify the server.
hostName	string	The host name part of the cache.
httpsPort	string	The HTTPS port on which the main application listens (443 in most cases).
id	string	The server id (database row number).
iloIpAddress	string	The IPv4 address of the lights-out-management port.
iloIpGateway	string	The IPv4 gateway address of the lights-out-management port.

Continued on next page



Table 6 – continued from previous page

Parameter	Type	Description
iloIpNetmask	string	The IPv4 netmask of the lights-out-management port.
iloPassword	string	The password of the of the lights-out-management user (displays as ** unless you are an ‘admin’ user).
iloUsername	string	The user name for lights-out-management.
interfaceMtu	string	The Maximum Transmission Unit (MTU) to configure for interfaceName.
interfaceName	string	The network interface name used for serving traffic.
ip6Address	string	The IPv6 address/netmask for interfaceName.
ip6Gateway	string	The IPv6 gateway for interfaceName.
ipAddress	string	The IPv4 address for interfaceName.
ipGateway	string	The IPv4 gateway for interfaceName.
ipNetmask	string	The IPv4 netmask for interfaceName.
lastUpdated	string	The Time and Date for the last update for this server.
mgmtIpAddress	string	The IPv4 address of the management port (optional).
mgmtIpGateway	string	The IPv4 gateway of the management port (optional).
mgmtIpNetmask	string	The IPv4 netmask of the management port (optional).
offlineReason	string	A user-entered reason why the server is in ADMIN_DOWN or OFFLINE status.
physLocation	string	The physical location name (see <a href="#">Physical Location</a> ).
physLocationId	string	The physical location id (see <a href="#">Physical Location</a> ).
profile	string	The assigned profile name (see <a href="#">Profiles</a> ).
profileDesc	string	The assigned profile description (see <a href="#">Profiles</a> ).
profileId	string	The assigned profile Id (see <a href="#">Profiles</a> ).
rack	string	A string indicating rack location.
routerHostName	string	The human readable name of the router.
routerPortName	string	The human readable name of the router port.
status	string	The Status string (See <a href="#">Status</a> ).
statusId	string	The Status id (See <a href="#">Status</a> ).
tcpPort	string	The default TCP port on which the main application listens (80 for a cache in most cases).
type	string	The name of the type of this server (see <a href="#">Types</a> ).
typeId	string	The id of the type of this server (see <a href="#">Types</a> ).
updPending	bool	

**Response Example**

```
{
  "response": [
    {
      "cachegroup": "us-il-chicago",
      "cachegroupId": "3",
      "cdnId": "3",
      "cdnName": "CDN-1",
      "domainName": "chi.kabletown.net",
      "guid": null,
      "hostname": "atsec-chi-00",
      "id": "19",
      "iloIpAddress": "172.16.2.6",
      "iloIpGateway": "172.16.2.1",
      "iloIpNetmask": "255.255.255.0",
      "iloPassword": "*****",
      "iloUsername": "",
      "interfaceMtu": "9000",
      "interfaceName": "bond0",
      "ip6Address": "2033:D0D0:3300::2:2/64",
      "ip6Gateway": "2033:D0D0:3300::2:1",

```

(continues on next page)

(continued from previous page)

```

        "ipAddress": "10.10.2.2",
        "ipGateway": "10.10.2.1",
        "ipNetmask": "255.255.255.0",
        "lastUpdated": "2015-03-08 15:57:32",
        "mgmtIpAddress": "",
        "mgmtIpGateway": "",
        "mgmtIpNetmask": "",
        "offlineReason": "N/A",
        "physLocation": "plocation-chi-1",
        "physLocationId": "9",
        "profile": "EDGE1_CDN1_421_SSL",
        "profileDesc": "EDGE1_CDN1_421_SSL profile",
        "profileId": "12",
        "rack": "RR 119.02",
        "routerHostName": "rtr-chi.kabletown.net",
        "routerPortName": "2",
        "status": "ONLINE",
        "statusId": "6",
        "tcpPort": "80",
        "httpsPort": "443",
        "type": "EDGE",
        "typeId": "3",
        "updPending": false
    },
    {
        ... more server data
    }
]
}

```

**GET /api/1.1/servers/:id**

Retrieves properties of a CDN server by server ID.

Authentication Required: Yes

Role(s) Required: None

**Request Route Parameters**

Name	Required	Description
id	yes	Server id.

**Response Properties**

Parameter	Type	Description
cachegroup	string	The cache group name (see <i>Cache Group</i> ).
cachegroupId	string	The cache group id.
cdnId	string	Id of the CDN to which the server belongs to.
cdnName	string	Name of the CDN to which the server belongs to.
domainName	string	The domain name part of the FQDN of the cache.

Continued on next page

Table 7 – continued from previous page

Parameter	Type	Description
guid	string	An identifier used to uniquely identify the server.
hostName	string	The host name part of the cache.
httpsPort	string	The HTTPS port on which the main application listens (443 in most cases).
id	string	The server id (database row number).
iloIpAddress	string	The IPv4 address of the lights-out-management port.
iloIpGateway	string	The IPv4 gateway address of the lights-out-management port.
iloIpNetmask	string	The IPv4 netmask of the lights-out-management port.
iloPassword	string	The password of the of the lights-out-management user (displays as ** unless you are an ‘admin’ user).
iloUsername	string	The user name for lights-out-management.
interfaceMtu	string	The Maximum Transmission Unit (MTU) to configure for interfaceName.
interfaceName	string	The network interface name used for serving traffic.
ip6Address	string	The IPv6 address/netmask for interfaceName.
ip6Gateway	string	The IPv6 gateway for interfaceName.
ipAddress	string	The IPv4 address for interfaceName.
ipGateway	string	The IPv4 gateway for interfaceName.
ipNetmask	string	The IPv4 netmask for interfaceName.
lastUpdated	string	The Time and Date for the last update for this server.
mgmtIpAddress	string	The IPv4 address of the management port (optional).
mgmtIpGateway	string	The IPv4 gateway of the management port (optional).
mgmtIpNetmask	string	The IPv4 netmask of the management port (optional).
offlineReason	string	A user-entered reason why the server is in ADMIN_DOWN or OFFLINE status.
physLocation	string	The physical location name (see <a href="#">Physical Location</a> ).
physLocationId	string	The physical location id (see <a href="#">Physical Location</a> ).
profile	string	The assigned profile name (see <a href="#">Profiles</a> ).
profileDesc	string	The assigned profile description (see <a href="#">Profiles</a> ).
profileId	string	The assigned profile Id (see <a href="#">Profiles</a> ).
rack	string	A string indicating rack location.
routerHostName	string	The human readable name of the router.
routerPortName	string	The human readable name of the router port.
status	string	The Status string (See <a href="#">Status</a> ).
statusId	string	The Status id (See <a href="#">Status</a> ).
tcpPort	string	The default TCP port on which the main application listens (80 for a cache in most cases).
type	string	The name of the type of this server (see <a href="#">Types</a> ).
typeId	string	The id of the type of this server (see <a href="#">Types</a> ).
updPending	bool	

**Response Example**

```
{
  "response": [
    {
      "cachegroup": "us-il-chicago",
      "cachegroupId": "3",
      "cdnId": "3",
      "cdnName": "CDN-1",
      "domainName": "chi.kabletown.net",
      "guid": null,
      "hostName": "atsec-chi-00",
      "id": "19",
      "iloIpAddress": "172.16.2.6",
      "iloIpGateway": "172.16.2.1",
```

(continues on next page)

(continued from previous page)

```

        "iloIpNetmask": "255.255.255.0",
        "iloPassword": "*****",
        "iloUsername": "",
        "interfaceMtu": "9000",
        "interfaceName": "bond0",
        "ip6Address": "2033:D0D0:3300::2:2/64",
        "ip6Gateway": "2033:D0D0:3300::2:1",
        "ipAddress": "10.10.2.2",
        "ipGateway": "10.10.2.1",
        "ipNetmask": "255.255.255.0",
        "lastUpdated": "2015-03-08 15:57:32",
        "mgmtIpAddress": "",
        "mgmtIpGateway": "",
        "mgmtIpNetmask": "",
        "offlineReason": "N/A",
        "physLocation": "plocation-chi-1",
        "physLocationId": "9",
        "profile": "EDGE1_CDN1_421_SSL",
        "profileDesc": "EDGE1_CDN1_421_SSL profile",
        "profileId": "12",
        "rack": "RR 119.02",
        "routerHostName": "rtr-chi.kabletown.net",
        "routerPortName": "2",
        "status": "ONLINE",
        "statusId": "6",
        "tcpPort": "80",
        "httpsPort": "443",
        "type": "EDGE",
        "typeId": "3",
        "updPending": false
    }
  ]
}

```

**GET /api/1.1/servers/summary**

Retrieves a count of CDN servers by type.

Authentication Required: Yes

Role(s) Required: None

**Response Properties**

Parameter	Type	Description
count	int	The number of servers of this type in this instance of Traffic Ops.
type	string	The name of the type of the server count (see <i>Types</i> ).

**Response Example**

```
{
  "response": [
```

(continues on next page)

(continued from previous page)

```

{
  "count": 4,
  "type": "CCR"
},
{
  "count": 55,
  "type": "EDGE"
},
{
  "type": "MID",
  "count": 18
},
{
  "count": 0,
  "type": "INFLUXDB"
},
{
  "count": 4,
  "type": "RASCAL"
}
}

```

**GET /api/1.1/servers/hostname/:name/details**

Retrieves the details of a server.

Authentication Required: Yes

Role(s) Required: None

**Request Route Parameters**

Name	Required	Description
name	yes	The host name part of the cache.

**Response Properties**

Parameter	Type	Description
cachegroup	string	The cache group name (see <a href="#">Cache Group</a> ).
deliveryservices	array	Array of strings with the delivery service ids assigned (see <a href="#">Delivery Service</a> ).
domainName	string	The domain name part of the FQDN of the cache.
hardwareInfo	hash	Hwinfo struct (see <a href="#">Hardware Info</a> ).
hostName	string	The host name part of the cache.
id	string	The server id (database row number).
iloIpAddress	string	The IPv4 address of the lights-out-management port.
iloIpGateway	string	The IPv4 gateway address of the lights-out-management port.
iloIpNetmask	string	The IPv4 netmask of the lights-out-management port.
iloPassword	string	The password of the of the lights-out-management user (displays as ** unless you are an 'admin'.
iloUsername	string	The user name for lights-out-management.
interfaceMtu	string	The Maximum Transmission Unit (MTU) to configure for interfaceName.

Continued on next

Table 8 – continued from previous page

Parameter	Type	Description
interfaceName	string	The network interface name used for serving traffic.
ip6Address	string	The IPv6 address/netmask for interfaceName.
ip6Gateway	string	The IPv6 gateway for interfaceName.
ipAddress	string	The IPv4 address for interfaceName.
ipGateway	string	The IPv4 gateway for interfaceName.
ipNetmask	string	The IPv4 netmask for interfaceName.
lastUpdated	string	The Time/Date of the last update for this server.
mgmtIpAddress	string	The IPv4 address of the management port (optional).
mgmtIpGateway	string	The IPv4 gateway of the management port (optional).
mgmtIpNetmask	string	The IPv4 netmask of the management port (optional).
physLocation	string	The physical location name (see <i>Physical Location</i> ).
profile	string	The assigned profile name (see <i>Profiles</i> ).
rack	string	A string indicating rack location.
routerHostName	string	The human readable name of the router.
routerPortName	string	The human readable name of the router port.
status	string	The Status string (See <i>Status</i> ).
tcpPort	string	The default TCP port on which the main application listens (80 for a cache in most cases).
httpsPort	string	The default HTTPS port on which the main application listens (443 for a cache in most cases).
type	string	The name of the type of this server (see <i>Types</i> ).
xmppId	string	Deprecated.
xmppPasswd	string	Deprecated.

**Response Example**

```
{
  "response": {
    "cachegroup": "us-il-chicago",
    "deliveryservices": [
      "1",
      "2",
      "3",
      "4"
    ],
    "domainName": "chi.kabletown.net",
    "hardwareInfo": {
      "Physical Disk 0:1:3": "D1S2",
      "Physical Disk 0:1:2": "D1S2",
      "Physical Disk 0:1:15": "D1S2",
      "Power Supply.Slot.2": "04.07.15",
      "Physical Disk 0:1:24": "YS08",
      "Physical Disk 0:1:1": "D1S2",
      "Model": "PowerEdge R720xd",
      "Physical Disk 0:1:22": "D1S2",
      "Physical Disk 0:1:18": "D1S2",
      "Enterprise UEFI Diagnostics": "4217A5",
      "Lifecycle Controller": "1.0.8.42",
      "Physical Disk 0:1:8": "D1S2",
      "Manufacturer": "Dell Inc.",
      "Physical Disk 0:1:6": "D1S2",
      "SysMemTotalSize": "196608",
      "PopulatedDIMMSlots": "24",
      "Physical Disk 0:1:20": "D1S2",
      "Intel(R) Ethernet 10G 2P X520 Adapter": "13.5.7",

```

(continues on next page)

(continued from previous page)

```

    "Physical Disk 0:1:14": "D1S2",
    "BACKPLANE FIRMWARE": "1.00",
    "Dell OS Drivers Pack, 7.0.0.29, A00": "7.0.0.29",
    "Integrated Dell Remote Access Controller": "1.57.57",
    "Physical Disk 0:1:5": "D1S2",
    "ServiceTag": "D6XPDV1",
    "PowerState": "2",
    "Physical Disk 0:1:23": "D1S2",
    "Physical Disk 0:1:25": "D903",
    "BIOS": "1.3.6",
    "Physical Disk 0:1:12": "D1S2",
    "System CPLD": "1.0.3",
    "Physical Disk 0:1:4": "D1S2",
    "Physical Disk 0:1:0": "D1S2",
    "Power Supply.Slot.1": "04.07.15",
    "PERC H710P Mini": "21.0.2-0001",
    "PowerCap": "689",
    "Physical Disk 0:1:16": "D1S2",
    "Physical Disk 0:1:10": "D1S2",
    "Physical Disk 0:1:11": "D1S2",
    "Lifecycle Controller 2": "1.0.8.42",
    "BP12G+EXP 0:1": "1.07",
    "Physical Disk 0:1:9": "D1S2",
    "Physical Disk 0:1:17": "D1S2",
    "Broadcom Gigabit Ethernet BCM5720": "7.2.20",
    "Physical Disk 0:1:21": "D1S2",
    "Physical Disk 0:1:13": "D1S2",
    "Physical Disk 0:1:7": "D1S2",
    "Physical Disk 0:1:19": "D1S2"
  },
  "hostName": "atsec-chi-00",
  "id": "19",
  "iloIpAddress": "172.16.2.6",
  "iloIpGateway": "172.16.2.1",
  "iloIpNetmask": "255.255.255.0",
  "iloPassword": "*****",
  "iloUsername": "",
  "interfaceMtu": "9000",
  "interfaceName": "bond0",
  "ip6Address": "2033:D0D0:3300::2:2/64",
  "ip6Gateway": "2033:D0D0:3300::2:1",
  "ipAddress": "10.10.2.2",
  "ipGateway": "10.10.2.1",
  "ipNetmask": "255.255.255.0",
  "mgmtIpAddress": "",
  "mgmtIpGateway": "",
  "mgmtIpNetmask": "",
  "physLocation": "plocation-chi-1",
  "profile": "EDGE1_CDN1_421_SSL",
  "rack": "RR 119.02",
  "routerHostName": "rtr-chi.kabletown.net",
  "routerPortName": "2",
  "status": "ONLINE",
  "tcpPort": "80",
  "httpsPort": "443",
  "type": "EDGE",
  "xmppId": "atsec-chi-00-dummyxmpp",

```

(continues on next page)

(continued from previous page)

```
"xmppPasswd": "X"

}
```

## POST /api/1.1/servercheck

Post a server check result to the serverchecks table.

Authentication Required: Yes

Role(s) Required: None

### Request Route Parameters

Name	Required	Description
id	yes	
host_name	yes	
servercheck_short_name	yes	
value	yes	

### Request Example

```
{
  "id": "",
  "host_name": "",
  "servercheck_short_name": "",
  "value": ""
}
```

### Response Properties

Parameter	Type	Description
alerts	array	A collection of alert messages.
>level	string	Success, info, warning or error.
>text	string	Alert message.
version	string	

### Response Example

Response Example:

```
{
  "alerts":
  [
    {
```

(continues on next page)



(continued from previous page)

```
    "level": "success",
    "text": "Server Check was successfully updated."
  },
],
}
```

## Static DNS Entries

### /api/1.1/staticdnsentries

#### GET /api/1.1/staticdnsentries.json

Authentication Required: Yes

Role(s) Required: None

#### Response Properties

Parameter	Type	Description
deliveryservice	string	
ttl	string	
type	string	
address	string	
cachegroup	string	
host	string	

#### Response Example

```
{
  "response": [
    {
      "deliveryservice": "foo-ds",
      "ttl": "30",
      "type": "CNAME_RECORD",
      "address": "bar.foo.baz.tv.",
      "cachegroup": "us-co-denver",
      "host": "mm"
    }
  ]
}
```

## Status

### /api/1.1/statuses

#### GET /api/1.1/statuses

Retrieves a list of the server status codes available.

Authentication Required: Yes

Role(s) Required: None

## Response Properties

Parameter	Type	Description
id	string	The id with which Traffic Ops stores this status, and references it internally
name	string	The string equivalent of the status
description	string	A short description of the status
lastUpdated	string	The Time / Date this server entry was last updated

## Response Example

```
{
  "response": [
    {
      "id": "4",
      "name": "ADMIN_DOWN",
      "description": "Temporary down. Edge: XMPP client will send status_
↪OFFLINE to CCR, otherwise similar to REPORTED. Mid: Server will not be_
↪included in parent.config files for its edge caches",
      "lastUpdated": "2013-02-13 16:34:29"
    },
    {
      "id": "5",
      "name": "CCR_IGNORE",
      "description": "Edge: 12M will not include caches in this state in CCR_
↪config files. Mid: N\A for now",
      "lastUpdated": "2013-02-13 16:34:29"
    },
    {
      "id": "1",
      "name": "OFFLINE",
      "description": "Edge: Puts server in CCR config file in this state,
↪but CCR will never route traffic to it. Mid: Server will not be included_
↪in parent.config files for its edge caches",
      "lastUpdated": "2013-02-13 16:34:29"
    },
    {
      "id": "2",
      "name": "ONLINE",
      "description": "Edge: Puts server in CCR config file in this state,
↪and CCR will always route traffic to it. Mid: Server will be included in_
↪parent.config files for its edges",
      "lastUpdated": "2013-02-13 16:34:29"
    },
    {
      "id": "3",
      "name": "REPORTED",
      "description": "Edge: Puts server in CCR config file in this state,
↪and CCR will adhere to the health protocol. Mid: N\A for now",
      "lastUpdated": "2013-02-13 16:34:29"
    }
  ]
}
```

## GET /api/1.1/statuses/:id

Retrieves a server status by ID.

Authentication Required: Yes

Role(s) Required: None

### Request Route Parameters

Name	Required	Description
id	yes	Status id.

### Response Properties

Parameter	Type	Description
id	string	The id with which Traffic Ops stores this status, and references it internally
name	string	The string equivalent of the status
description	string	A short description of the status
lastUpdated	string	The Time / Date this server entry was last updated

### Response Example

```
{
  "response": [
    {
      "id": "4",
      "name": "ADMIN_DOWN",
      "description": "Temporary down. Edge: XMPP client will send status_
↪OFFLINE to CCR, otherwise similar to REPORTED. Mid: Server will not be_
↪included in parent.config files for its edge caches",
      "lastUpdated": "2013-02-13 16:34:29"
    }
  ]
}
```

## System

### /api/1.1/system

#### GET /api/1.1/system/info.json

Authentication Required: Yes

Role(s) Required: None

### Response Properties

Key	Type	Description
parameterhash	hash	This is a hash with the parameter names that describe the Traffic Ops installation as keys. These are all the parameters in the GLOBAL profile.
>tm. toolname	string	The name of the Traffic Ops tool. Usually “Traffic Ops”. Used in the About screen and in the comments headers of the files generated (# DO NOT EDIT - Generated for atsec-lax-04 by Traffic Ops (https://traffops.kabletown.net/) on Fri Mar 6 05:15:15 UTC 2015).
>tm. instance_name	string	The name of the Traffic Ops instance. Can be used when multiple instances are active. Visible in the About page.
>trafficmon. fwd_proxy	string	When collecting stats from Traffic Router, Traffic Ops uses this forward proxy to pull the stats through. This can be any of the MID tier caches, or a forward cache specifically deployed for this purpose. Setting this variable can significantly lighten the load on the Traffic Router stats system and it is recommended to set this parameter on a production system.
>tm. url	string	The URL for this Traffic Ops instance. Used in the About screen and in the comments headers of the files generated (# DO NOT EDIT - Generated for atsec-lax-04 by Traffic Ops (https://traffops.kabletown.net/) on Fri Mar 6 05:15:15 UTC 2015).
>trafficrouter. fwd_proxy	string	When collecting stats from Traffic Monitor, Traffic Ops uses this forward proxy to pull the stats through. This can be any of the MID tier caches, or a forward cache specifically deployed for this purpose. Setting this variable can significantly lighten the load on the Traffic Monitor system and it is recommended to set this parameter on a production system.
>tm. logourl	string	This is the URL of the logo for Traffic Ops and can be relative if the logo is under traffic_ops/app/public.
>tm. infourl	string	This is the “for more information go here” URL, which is visible in the About page.

### Response Example

```
{
  "response": {
    "parameters": {
      "tm.toolname": "Traffic Ops",
      "tm.infourl": "http://staging-03.cdnlabs.kabletown.net/tm/info",
      "traffic_mon_fwd_proxy": "http://proxy.kabletown.net:81",
      "traffic_rtr_fwd_proxy": "http://proxy.kabletown.net:81",
      "tm.logourl": "\/images\/tc_logo.png",
      "tm.url": "https://tm.kabletown.net\/",
      "tm.instance_name": "Kabletown CDN"
    }
  },
}
```

## TO Extensions

### /api/1.1/to\_extensions

#### GET /api/1.1/to\_extensions.json

Retrieves the list of extensions.

Authentication Required: Yes

Role(s) Required: None

### Response Properties

Parameter	Type	Description
script_file	string	
version	string	
name	string	
description	string	
info_url	string	
additional_config_json	string	
isactive	string	
id	string	
type	string	
servercheck_short_name	string	

### Response Example

```
{
  "response": [
    {
      script_file: "ping",
      version: "1.0.0",
      name: "ILO_PING",
      description: null,
      info_url: "http://foo.com/bar.html",
      additional_config_json: "{ \"path\": \"/api/1.1/servers.json\",
↪"match": { \"type\": \"EDGE\"}, \"select\": \"ilo_ip_address\", \"cron\": \"9 * * * *\" }",
      isactive: "1",
      id: "1",
      type: "CHECK_EXTENSION_BOOL",
      servercheck_short_name: "ILO"
    },
    {
      script_file: "ping",
      version: "1.0.0",
      name: "10G_PING",
      description: null,
      info_url: "http://foo.com/bar.html",
      additional_config_json: "{ \"path\": \"/api/1.1/servers.json\",
↪"match": { \"type\": \"EDGE\"}, \"select\": \"ip_address\", \"cron\": \"18 * * * *\" }",
      isactive: "1",
      id: "2",
      type: "CHECK_EXTENSION_BOOL",
      servercheck_short_name: "10G"
    }
  ],
}
```

### POST /api/1.1/to\_extensions

Creates a Traffic Ops extension.

Authentication Required: Yes

Role(s) Required: None

### Request Parameters

Parameter	Type	Description
name	string	
version	string	
info_url	string	
script_file	string	
isactive	string	
additional_config_json	string	
description	string	
servercheck_short_name	string	
type	string	

### Request Example

```
{
  "name": "ILO_PING",
  "version": "1.0.0",
  "info_url": "http://foo.com/bar.html",
  "script_file": "ping",
  "isactive": "1",
  "additional_config_json": "{ \"path\": \"/api/1.1/servers.json\", \"match\":
↪ { \"type\": \"EDGE\"} \",
  "description": null,
  "servercheck_short_name": "ILO"
  "type": "CHECK_EXTENSION_BOOL",
}
```

### Response Properties

Parameter	Type	Description
alerts	array	A collection of alert messages.
>level	string	Success, info, warning or error.
>text	string	Alert message.

### Response Example

```
{
  "alerts": [
    {
      "level": "success",
      "text": "Check Extension loaded."
    }
  ],
}
```

**POST /api/1.1/to\_extensions/:id/delete**

Deletes a Traffic Ops extension.

Authentication Required: Yes

Role(s) Required: None

**Request Route Parameters**

Name	Required	Description
id	yes	TO extension id

**Response Properties**

Parameter	Type	Description
alerts	array	A collection of alert messages.
>level	string	Success, info, warning or error.
>text	string	Alert message.

**Response Example**

```
{
  "alerts": [
    {
      "level": "success",
      "text": "Extension deleted."
    }
  ],
}
```

## Types

**/api/1.1/types****GET /api/1.1/types**

Authentication Required: Yes

Role(s) Required: None

**Request Query Parameters**

Name	Required	Description
useInTable	no	Filter types by the table in which they apply

**Response Properties**

Parameter	Type	Description
id	string	
name	string	
description	string	
useInTable	string	
lastUpdated	string	

**Response Example**

```
{
  "response": [
    {
      "id": "22",
      "name": "AAAA_RECORD",
      "description": "Static DNS AAAA entry",
      "useInTable": "staticdnsentry",
      "lastUpdated": "2013-10-23 15:28:31"
    }
  ]
}
```

**GET /api/1.1/types/trimmed**

Authentication Required: Yes

Role(s) Required: None

**Response Properties**

Parameter	Type	Description
name	string	

**Response Example**

```
{
  "response": [
    {
      "name": "AAAA_RECORD"
    },
    {
      "name": "ACTIVE_DIRECTORY"
    },
    {
      "name": "A_RECORD"
    },
    {
      "name": "CCR"
    }
  ],
}
```

**GET /api/1.1/types/:id**



Authentication Required: Yes

Role(s) Required: None

#### Request Route Parameters

Name	Required	Description
id	yes	Type ID.

#### Response Properties

Parameter	Type	Description
id	string	
name	string	
description	string	
useInTable	string	
lastUpdated	string	

#### Response Example

```
{
  "response": [
    {
      "id": "22",
      "name": "AAAA_RECORD",
      "description": "Static DNS AAAA entry",
      "useInTable": "staticdnsentry",
      "lastUpdated": "2013-10-23 15:28:31"
    }
  ]
}
```

## Users

### /api/1.1/users

#### GET /api/1.1/users

Retrieves all users.

Authentication Required: Yes

Role(s) Required: None

#### Response Properties

Parameter	Type	Description
addressLine1	string	
addressLine2	string	
city	string	
company	string	
country	string	
email	string	
fullName	string	
gid	string	
id	hash	
lastUpdated	string	
newUser	string	
phoneNumber	string	
postalCode	string	
publicSshKey	string	
registrationSent	string	
role	string	
roleName	string	
stateOrProvince	string	
uid	string	
username	string	

### Response Example

```
{
  "response": [
    {
      "addressLine1": "",
      "addressLine2": "",
      "city": "",
      "company": "",
      "country": "",
      "email": "email1@email.com",
      "fullName": "Tom Simpson",
      "gid": "0",
      "id": "53",
      "lastUpdated": "2016-01-26 10:22:07",
      "newUser": true,
      "phoneNumber": "",
      "postalCode": "",
      "publicSshKey": "xxx",
      "registrationSent": true,
      "role": "6",
      "rolename": "admin",
      "stateOrProvince": "",
      "uid": "0",
      "username": "tsimpson"
    },
    {
      ... more users
    }
  ]
}
```

**GET /api/1.1/users/:id**

Retrieves user by ID.

Authentication Required: Yes

Role(s) Required: None

**Request Route Parameters**

Name	Required	Description
id	yes	User id.

**Response Properties**

Parameter	Type	Description
addressLine1	string	
addressLine2	string	
city	string	
company	string	
country	string	
email	string	
fullName	string	
gid	string	
id	hash	
lastUpdated	string	
newUser	string	
phoneNumber	string	
postalCode	string	
publicSshKey	string	
registrationSent	string	
role	string	
roleName	string	
stateOrProvince	string	
uid	string	
username	string	

**Response Example**

```
{
  "response": [
    {
      "addressLine1": "",
      "addressLine2": "",
      "city": "",
      "company": "",
      "country": "",
      "email": "email1@email.com",
      "fullName": "Tom Simpson",
      "gid": "0",
      "id": "53",
      "lastUpdated": "2016-01-26 10:22:07",
```

(continues on next page)

(continued from previous page)

```
        "newUser": true,  
        "phoneNumber": "",  
        "postalCode": "",  
        "publicSshKey": "xxx",  
        "registrationSent": true,  
        "role": "6",  
        "rolename": "admin",  
        "stateOrProvince": "",  
        "uid": "0",  
        "username": "tsimpson"  
    }  
]  
}
```

### GET /api/1.1/user/current

Retrieves the profile for the authenticated user.

Authentication Required: Yes

Role(s) Required: None

#### Request Properties

Parameter	Type	Description
email	string	
city	string	
id	string	
phoneNumber	string	
company	string	
country	string	
fullName	string	
localUser	boolean	
uid	string	
stateOrProvince	string	
username	string	
newUser	boolean	
addressLine2	string	
role	string	
addressLine1	string	
gid	string	
postalCode	string	

#### Response Example

```
{  
  "response": {  
    "email": "email@email.com",  
    "city": "",  
    "id": "50",
```

(continues on next page)

(continued from previous page)

```

        "phoneNumber": "",
        "company": "",
        "country": "",
        "fullName": "Tom Callahan",
        "localUser": true,
        "uid": "0",
        "stateOrProvince": "",
        "username": "tommyboy",
        "newUser": false,
        "addressLine2": "",
        "role": "6",
        "addressLine1": "",
        "gid": "0",
        "postalCode": ""
    },
}

```

**POST /api/1.1/user/current/update**

Updates the date for the authenticated user.

Authentication Required: Yes

Role(s) Required: None

**Request Properties**

Parameter	Type	Description
email	string	
city	string	
id	string	
phoneNumber	string	
company	string	
country	string	
fullName	string	
localUser	boolean	
uid	string	
stateOrProvince	string	
username	string	
newUser	boolean	
addressLine2	string	
role	string	
addressLine1	string	
gid	string	
postalCode	string	

**Request Example**

```

{
  "user": {

```

(continues on next page)

(continued from previous page)

```
"email": "",
"city": "",
"id": "",
"phoneNumber": "",
"company": "",
"country": "",
"fullName": "",
"localUser": true,
"uid": "0",
"stateOrProvince": "",
"username": "tommyboy",
"newUser": false,
"addressLine2": "",
"role": "6",
"addressLine1": "",
"gid": "0",
"postalCode": ""
}
}
```

### Response Properties

Parameter	Type	Description
alerts	array	A collection of alert messages.
>level	string	Success, info, warning or error.
>text	string	Alert message.
version	string	

### Response Example

```
{
  "alerts": [
    {
      "level": "success",
      "text": "UserProfile was successfully updated."
    }
  ],
}
```

### GET /api/1.1/user/current/jobs.json

Retrieves the user's list of jobs.

Authentication Required: Yes

Role(s) Required: None

#### Request Query Parameters

Name	Required	Description
keyword	no	PURGE

## Response Properties

Parameter	Type	Description
keyword	string	
objectName	string	
assetUrl	string	
assetType	string	
status	string	
dsId	string	
dsXmlId	string	
username	boolean	
parameters	string	
enteredTime	string	
objectType	string	
agent	string	
id	string	
startTime	string	
version	string	

## Response Example

```
{
  "response": [
    {
      "id": "1",
      "keyword": "PURGE",
      "objectName": null,
      "assetUrl": "",
      "assetType": "file",
      "status": "PENDING",
      "dsId": "9999",
      "dsXmlId": "ds-xml-id",
      "username": "peewee",
      "parameters": "TTL:56h",
      "enteredTime": "2015-01-21 18:00:16",
      "objectType": null,
      "agent": "",
      "startTime": "2015-01-21 10:45:38"
    }
  ],
}
```

### POST/api/1.1/user/current/jobs

Invalidating content on the CDN is sometimes necessary when the origin was mis-configured and something is cached in the CDN that needs to be removed. Given the size of a typical Traffic Control CDN and the amount of content that can be cached in it, removing the content from all the caches may take a long time. To speed up content invalidation, Traffic Ops will not try to remove the content from the caches, but it makes the content inaccessible using the *regex\_revalidate* ATS plugin. This forces a *revalidation* of the content, rather than a new get.

**Note:** This method forces a HTTP *revalidation* of the content, and not a new *GET* - the origin needs to support revalidation according to the HTTP/1.1 specification, and send a 200 OK or 304 Not Modified as applicable.

---

Authentication Required: Yes

Role(s) Required: Yes

### Request Properties

Parameter	Type	Description
dsId	string	Unique Delivery Service ID
regex	string	Path Regex this should be a <a href="#">PCRE</a> compatible regular expression for the path to match for forcing the revalidation. Be careful to only match on the content you need to remove - revalidation is an expensive operation for many origins, and a simple <code>/.*</code> can cause an overload condition of the origin.
startTime	string	Start Time is the time when the revalidation rule will be made active. Populate with the current time to schedule ASAP.
ttl	int	Time To Live is how long the revalidation rule will be active for in hours. It usually makes sense to make this the same as the <code>Cache-Control</code> header from the origin which sets the object time to live in cache (by <code>max-age</code> or <code>Expires</code> ). Entering a longer TTL here will make the caches do unnecessary work.

### Request Example

```
{
  "dsId": "9999",
  "regex": "/path/to/content.jpg",
  "startTime": "2015-01-27 11:08:37",
  "ttl": 54
}
```

### Response Properties

Parameter	Type	Description
alerts	array	A collection of alert messages.
>level	string	Success, info, warning or error.
>text	string	Alert message.
version	string	

### Response Example

```
{
  "alerts": [
    {
      "level": "success",
```

(continues on next page)



(continued from previous page)

```
        "text": "Successfully created purge job for: ."  
      },  
    ],  
  }  
}
```

## POST /api/1.1/user/login

Authentication of a user using username and password. Traffic Ops will send back a session cookie.

Authentication Required: No

Role(s) Required: None

### Request Properties

Parameter	Type	Description
u	string	username
p	string	password

### Request Example

```
{  
  "u": "username",  
  "p": "password"  
}
```

### Response Properties

Parameter	Type	Description
alerts	array	A collection of alert messages.
>level	string	Success, info, warning or error.
>text	string	Alert message.
version	string	

### Response Example

```
{  
  "alerts": [  
    {  
      "level": "success",  
      "text": "Successfully logged in."  
    },  
  ],  
  "version": "2.1.0"  
}
```

**GET /api/1.1/user/:id/deliveryservices/available.json**

Authentication Required: Yes

Role(s) Required: None

**Request Route Parameters**

Name	Required	Description
id	yes	

**Response Properties**

Parameter	Type	Description
xmlId	string	
id	string	

**Response Example**

```
{
  "response": [
    {
      "xmlId": "ns-img",
      "id": "90"
    },
    {
      "xmlId": "ns-img-secure",
      "id": "280"
    }
  ],
}
```

**POST /api/1.1/user/login/token**

Authentication of a user using a token.

Authentication Required: No

Role(s) Required: None

**Request Properties**

Parameter	Type	Description
t	string	token-value

**Request Example**

```
{
  "t": "token-value"
}
```

**Response Properties**

Parameter	Type	Description
alerts	array	
>level	string	
>text	string	
version	string	

**Response Example**

```
{
  "alerts": [
    {
      "level": "error",
      "text": "Unauthorized, please log in."
    }
  ],
}
```

**POST /api/1.1/user/logout**

User logout. Invalidates the session cookie.

Authentication Required: Yes

Role(s) Required: None

**Response Properties**

Parameter	Type	Description
alerts	array	
• level	string	
• text	string	
version	string	

**Response Example**

```
{
  "alerts": [
    {
      "level": "success",
      "text": "You are logged out."
    }
  ],
}
```

**POST /api/1.1/user/reset\_password**

Reset user password.

Authentication Required: No

Role(s) Required: None

**Request Properties**

Parameter	Type	Description
email	string	The email address of the user to initiate password reset.

**Request Example**

```
{
  "email": "email@email.com"
}
```

**Response Properties**

Parameter	Type	Description
alerts	array	A collection of alert messages.
• level	string	Success, info, warning or error.
• text	string	Alert message.
version	string	

**Response Example**

```
{
  "alerts": [
    {
      "level": "success",
      "text": "Successfully sent password reset to email 'email@email.com'"
    }
  ],
  "version": "1.1"
}
```

**API 1.2 Reference****API-Capabilities**

**/api/1.2/api\_capabilities****GET /api/1.2/api\_capabilities**

Get all API-capability mappings.

Authentication Required: Yes

Role(s) Required: None

**Query Parameters**

Name	Required	Type	Description
capability	no	string	Capability name.

**Response Properties**

Parameter	Type	Description
id	int	Mapping id.
httpMethod	enum	One of: 'GET', 'POST', 'PUT', 'PATCH', 'DELETE'.
route	string	API route.
capability	string	Capability name.
lastUpdated	string	

**Response Example**

```
{
  "response": [
    {
      "id": "6",
      "httpMethod": "GET",
      "route": "/api/*/asns",
      "capability": "asn-read",
      "lastUpdated": "2017-04-02 08:22:43"
    },
    {
      "id": "7",
      "httpMethod": "GET",
      "route": "/api/*/asns/*",
      "capability": "asn-read",
      "lastUpdated": "2017-04-02 08:22:43"
    }
  ]
}
```

**GET /api/1.2/api\_capabilities/:id**

Get an API-capability mapping by id.

Authentication Required: Yes

Role(s) Required: None

**Request Route Parameters**

Name	Required	Type	Description
id	yes	int	Mapping id.

### Response Properties

Parameter	Type	Description
id	int	Mapping id.
httpMethod	enum	One of: 'GET', 'POST', 'PUT', 'PATCH', 'DELETE'.
route	string	API route.
capability	string	Capability name.
lastUpdated	string	

### Response Example

```
{
  "response": [
    {
      "id": "6",
      "httpMethod": "GET",
      "route": "/api/*/asns",
      "capability": "asn-read",
      "lastUpdated": "2017-04-02 08:22:43"
    }
  ]
}
```

## POST /api/1.2/api\_capabilities

Create an API-capability mapping.

Authentication Required: Yes

Role(s) Required: admin or oper

### Request Properties

Name	Required	Type	Description
httpMethod	yes	enum	One of: 'GET', 'POST', 'PUT', 'PATCH', 'DELETE'.
route	yes	string	API route.
capability	yes	string	Capability name

### Request Example

```
{
  "httpMethod": "POST",
  "route": "/api/*/cdns",
  "capability": "cdn-write"
}
```

### Response Properties

Parameter	Type	Description
response	hash	The details of the creation, if success.
>id	int	Mapping id.
>httpMethod	enum	One of: 'GET', 'POST', 'PUT', 'PATCH', 'DELETE'.
>route	string	API route.
>capability	string	Capability name
>lastUpdated	string	
alerts	array	A collection of alert messages.
>level	string	Success, info, warning or error.
>text	string	Alert message.

### Response Example

```
{
  "response": {
    "id": "6",
    "httpMethod": "POST",
    "route": "/api/*/cdns",
    "capability": "cdn-write",
    "lastUpdated": "2017-04-02 08:22:43"
  },
  "alerts": [
    {
      "level": "success",
      "text": "API-capability mapping was created."
    }
  ]
}
```

### PUT /api/1.2/api\_capabilities/{:id}

Edit an API-capability mapping.

Authentication Required: Yes

Role(s) Required: admin or oper

#### Request Route Parameters

Name	Required	Type	Description
id	yes	string	Mapping id.

#### Request Properties

Parameter	Type	Description
httpMethod	enum	One of: 'GET', 'POST', 'PUT', 'PATCH', 'DELETE'.
route	string	API route.
capability	string	Capability name

#### Request Example

```
{
  "httpMethod": "GET",
  "route": "/api/*/cdns",
  "capability": "cdn-read"
}
```

### Response Properties

Parameter	Type	Description
response	hash	The details of the creation, if success.
>id	int	Mapping id.
>httpMethod	enum	One of: 'GET', 'POST', 'PUT', 'PATCH', 'DELETE'.
>route	string	API route.
>capability	string	Capability name
>lastUpdated	string	
alerts	array	A collection of alert messages.
>level	string	Success, info, warning or error.
>text	string	Alert message.

### Response Example

```
{
  "response":{
    "id": "6",
    "httpMethod": "GET",
    "route": "/api/*/cdns",
    "capability": "cdn-read",
    "lastUpdated": "2017-04-02 08:22:43"
  },
  "alerts":[
    {
      "level": "success",
      "text": "API-capability mapping was updated."
    }
  ]
}
```

## DELETE /api/1.2/api\_capabilities/{:id}

Delete a capability.

Authentication Required: Yes

Role(s) Required: admin or oper

### Request Route Parameters

Name	Required	Type	Description
id	yes	string	Mapping id.

### Response Properties



Parameter	Type	Description
alerts	array	A collection of alert messages.
>level	string	success, info, warning or error.
>text	string	Alert message.

**Response Example**

```
{
  "alerts": [
    {
      "level": "success",
      "text": "API-capability mapping deleted."
    }
  ],
}
```

**ASN****/api/1.2/asns****GET /api/1.2/asns**

Authentication Required: Yes

Role(s) Required: None

**Request Query Parameters**

Name	Required	Description
cachegroup	no	Filter ASNs by cache group ID

**Response Properties**

Parameter	Type	Description
lastUpdated	string	The Time / Date this server entry was last updated
id	string	Local unique identifier for the ASN
asn	string	Autonomous System Numbers per APNIC for identifying a service provider.
cachegroup	string	Related cachegroup name
cachegroupId	string	Related cachegroup id

**Response Example**

```
{
  "response": [
    {
      "lastUpdated": "2012-09-17 21:41:22",
      "id": "27",
      "asn": "7015",
      "cachegroup": "us-ma-woburn",
      "cachegroupId": "27",
    },
  ],
}
```

(continues on next page)

(continued from previous page)

```
{
  "lastUpdated": "2012-09-17 21:41:22",
  "id": "28",
  "asn": "7016",
  "cachegroup": "us-pa-pittsburgh",
  "cachegroupId": "13"
}
```

### GET /api/1.2/asns/:id

Authentication Required: Yes

Role(s) Required: None

#### Request Route Parameters

Name	Required	Description
id	yes	ASN id.

#### Response Properties

Parameter	Type	Description
lastUpdated	string	The Time / Date this server entry was last updated
id	string	Local unique identifier for the ASN
asn	string	Autonomous System Numbers per APNIC for identifying a service provider.
cachegroup	string	Related cachegroup name
cachegroupId	string	Related cachegroup id

#### Response Example

```
{
  "response": [
    {
      "lastUpdated": "2012-09-17 21:41:22",
      "id": "28",
      "asn": "7016",
      "cachegroup": "us-pa-pittsburgh",
      "cachegroupId": "13"
    }
  ]
}
```

### PUT /api/1.2/asns/{:id}

Allows user to edit an ASN.

Authentication Required: Yes

Role(s) Required: admin or oper

### Request Route Parameters

Name	Type	Description
id	int	ASN id.

### Request Properties

Parameter	Type	Description
asn	string	ASN
cachegroupId	string	The cachegroup the ASN belongs to

### Request Example

```
{
  "asn": "99",
  "cachegroupId": "177"
}
```

### Response Properties

Parameter	Type	Description
response	hash	The details of the update, if success.
>name	string	CDN name.
>id	int	CDN id.
>dnssecEnabled	string	Whether dnssec is enabled.
alerts	array	A collection of alert messages.
>level	string	Success, info, warning or error.
>text	string	Alert message.

### Response Example

```
{
  "response": {
    "lastUpdated": "2012-09-17 21:41:22",
    "id": "28",
    "asn": "99",
    "cachegroup": "us-pa-pittsburgh",
    "cachegroupId": "177"
  },
  "alerts": [
    {
      "level": "success",
      "text": "cdn was updated."
    }
  ]
}
```

## Cache

### /api/1.2/caches/stats

#### GET /api/1.2/caches/stats

Retrieves cache stats from Traffic Monitor. Also includes rows for aggregates.

Authentication Required: Yes

Role(s) Required: None

#### Response Properties

Parameter	Type	Description
profile	string	The profile of the cache.
cachegroup	string	The cache group of the cache.
hostname	string	The hostname of the cache.
ip	string	The IP address of the cache.
status	string	The status of the cache.
healthy	string	Has Traffic Monitor marked the cache as healthy or unhealthy?
connections	string	Cache connections
kbits	string	Cache kbits out

#### Response Example

```
{
  "response": [
    {
      "profile": "ALL",
      "cachegroup": "ALL",
      "hostname": "ALL",
      "ip": null,
      "status": "ALL",
      "healthy": true,
      "connections": 934424,
      "kbits": 618631875
    },
    {
      "profile": "EDGE1_FOO_721-ATS621-45",
      "cachegroup": "us-nm-albuquerque",
      "hostname": "foo-bar-alb-01",
      "ip": "2.2.2.2",
      "status": "REPORTED",
      "healthy": true,
      "connections": 373,
      "kbits": 390136
    }
  ]
}
```

## Cache Group

### /api/1.2/cachegroups

#### GET /api/1.1/cachegroups

Authentication Required: Yes

Role(s) Required: None

#### Request Query Parameters

Name	Required	Description
type	no	Filter cache groups by Type ID.

#### Response Properties

Parameter	Type	Description
id	string	Local unique identifier for the Cache Group
lastUpdated	string	The Time / Date this entry was last updated
latitude	string	Latitude for the Cache Group
longitude	string	Longitude for the Cache Group
name	string	The name of the Cache Group entry
parentCachegroupId	string	Parent cachegroup ID.
parentCachegroupName	string	Parent cachegroup name.
secondaryParentCachegroupId	string	Secondary parent cachegroup ID.
secondaryParentCachegroupName	string	Secondary parent cachegroup name.
shortName	string	Abbreviation of the Cache Group Name
typeId	string	Unique identifier for the 'Type' of Cache Group entry
typeName	string	The name of the type of Cache Group entry

#### Response Example

```
{
  "response": [
    {
      "id": "21",
      "lastUpdated": "2012-09-25 20:27:28",
      "latitude": "0",
      "longitude": "0",
      "name": "dc-chicago",
      "parentCachegroupId": null,
      "parentCachegroupName": null,
      "secondaryParentCachegroupId": null,
      "secondaryParentCachegroupName": null,
      "shortName": "dcchi",
      "typeName": "MID_LOC",
      "typeId": "4"
    },
    {
      "id": "22",
      "lastUpdated": "2012-09-25 20:27:28",
      "latitude": "0",
```

(continues on next page)

(continued from previous page)

```
        "longitude": "0",
        "name": "dc-chicago-1",
        "parentCachegroupId": null,
        "parentCachegroupName": null,
        "secondaryParentCachegroupId": null,
        "secondaryParentCachegroupName": null,
        "shortName": "dcchi",
        "typeName": "MID_LOC",
        "typeId": "4"
    }
],
}
```

### GET /api/1.2/cacheGroups/trimmed

Authentication Required: Yes

Role(s) Required: None

#### Response Properties

Parameter	Type	Description
name	string	

#### Response Example

```
{
  "response": [
    {
      "name": "dc-chicago"
    },
    {
      "name": "dc-cmc"
    }
  ],
}
```

### GET /api/1.2/cacheGroups/:id

Authentication Required: Yes

Role(s) Required: None

#### Response Properties

Parameter	Type	Description
id	string	Local unique identifier for the Cache Group
lastUpdated	string	The Time / Date this entry was last updated
latitude	string	Latitude for the Cache Group
longitude	string	Longitude for the Cache Group
name	string	The name of the Cache Group entry
parentCachegroupId	string	Parent cachegroup ID.
parentCachegroupName	string	Parent cachegroup name.
secondaryParentCachegroupId	string	Secondary parent cachegroup ID.
secondaryParentCachegroupName	string	Secondary parent cachegroup name.
shortName	string	Abbreviation of the Cache Group Name
typeId	string	Unique identifier for the 'Type' of Cache Group entry
typeName	string	The name of the type of Cache Group entry

### Response Example

```
{
  "response": [
    {
      "id": "21",
      "lastUpdated": "2012-09-25 20:27:28",
      "latitude": "0",
      "longitude": "0",
      "name": "dc-chicago",
      "parentCachegroupId": null,
      "parentCachegroupName": null,
      "secondaryParentCachegroupId": null,
      "secondaryParentCachegroupName": null,
      "shortName": "dcchi",
      "typeName": "MID_LOC",
      "typeId": "4"
    }
  ],
}
```

### GET /api/1.2/cachegroups/:id/parameters

Authentication Required: Yes

Role(s) Required: None

#### Response Properties

Parameter	Type	Description
id	int	Local unique identifier for the parameter
name	string	Name of the parameter
value	string	Value of the parameter
configFile	string	Config file associated with the parameter
secure	bool	Is the parameter value only visible to admin users
lastUpdated	string	The Time / Date this entry was last updated

**Response Example**

```
{
  "response": [
    {
      "id": "1100",
      "name": "cgw.originUrl",
      "value": "http://to-short.g.foo.net/data/",
      "configFile": "foo.config",
      "secure": false,
      "lastUpdated": "2015-08-27 15:11:49"
    },
    { ... }
  ]
}
```

**GET /api/1.2/cachegroups/:id/unassigned\_parameters**

Retrieves all parameters NOT assigned to the cache group.

Authentication Required: Yes

Role(s) Required: None

**Request Route Parameters**

Name	Required	Description
id	yes	Cache group id

**Response Properties**

Parameter	Type	Description
last_updated	string	The Time / Date this server entry was last updated
secure	boolean	When true, the parameter is accessible only by admin users. Defaults to false.
value	string	The parameter value, only visible to admin if secure is true
name	string	The parameter name
config_file	string	The parameter config_file

**Response Example**

```
{
  "response": [
    {
      "last_updated": "2012-09-17 21:41:22",
      "secure": false,
      "value": "0,1,2,3,4,5,6",
      "name": "Drive_Letters",
      "config_file": "storage.config"
    },
    {
      "last_updated": "2012-09-17 21:41:22",
```

(continues on next page)



(continued from previous page)

```
"secure": true,  
"value": "STRING __HOSTNAME__",  
"name": "CONFIG proxy.config.proxy_name",  
"config_file": "records.config"  
}  
],  
}
```

### GET /api/1.2/cachegroup/:parameter\_id/parameter

Authentication Required: Yes

Role(s) Required: None

#### Request Route Parameters

Name	Required	Description
parameter_id	yes	

#### Response Properties

Parameter	Type	Description
cachegroups	array	
>name	string	
>id	string	

#### Response Example

```
{  
  "response": {  
    "cachegroups": [  
      {  
        "name": "dc-chicago",  
        "id": "21"  
      },  
      {  
        "name": "dc-cmc",  
        "id": "22"  
      }  
    ]  
  },  
}
```

### GET /api/1.2/cachegroupparameters

Authentication Required: Yes

Role(s) Required: None

### Response Properties

Parameter	Type	Description
cachegroupParameters	array	A collection of cache group parameters.
>parameter	string	
>lastUpdated	string	
>cachegroup	string	

### Response Example

```
{
  "response": {
    "cachegroupParameters": [
      {
        "parameter": "379",
        "lastUpdated": "2013-08-05 18:49:37",
        "cachegroup": "us-ca-sanjose"
      },
      {
        "parameter": "380",
        "lastUpdated": "2013-08-05 18:49:37",
        "cachegroup": "us-ca-sanjose"
      },
      {
        "parameter": "379",
        "lastUpdated": "2013-08-05 18:49:37",
        "cachegroup": "us-ma-woburn"
      }
    ]
  },
}
```

## GET /api/1.2/cachegroups/:parameter\_id/parameter/available

Authentication Required: Yes

Role(s) Required: None

### Request Route Parameters

Name	Required	Description
parameter_id	yes	

### Response Properties

Parameter	Type	Description
name		
id		

**Response Example**

```
{
  "response": [
    {
      "name": "dc-chicago",
      "id": "21"
    },
    {
      "name": "dc-cmc",
      "id": "22"
    }
  ],
}
```

**POST /api/1.2/cacheGroups**

Create cache group.

Authentication Required: Yes

Role(s) Required: admin or oper

**Request Parameters**

Name	Re-quired	Description
name	yes	The name of the Cache Group entry
shortName	yes	Abbreviation of the Cache Group Name
latitude	no	Latitude for the Cache Group
longitude	no	Longitude for the Cache Group
parentCachegroup	no	Name of Parent Cache Group entry.
secondaryParentCachegroup	no	Name of Secondary Parent Cache Group entry.
typeId	yes	The type of Cache Group entry, "EDGE_LOC", "MID_LOC" or "ORG_LOC"

**Request Example**

```
{
  "name": "cache_group_edge",
  "shortName": "cg_edge",
  "latitude": 12,
  "longitude": 45,
  "parentCachegroup": "cache_group_mid",
  "typeId": 6
}
```

**Response Properties**

Parameter	Type	Description
id	string	The id of cache group
name	string	The name of the Cache Group entry
shortName	string	Abbreviation of the Cache Group Name
latitude	string	Latitude for the Cache Group
longitude	string	Longitude for the Cache Group
parentCachegroup	string	Name of Parent Cache Group entry.
parentCachegroupId	string	id of Parent Cache Group entry.
secondaryParentCachegroup	string	Name of Secondary Parent Cache Group entry.
secondaryParentCachegroupId	string	id of Secondary Parent Cache Group entry.
typeName	string	The type of Cache Group entry, "EDGE_LOC", "MID_LOC" or "ORG_LOC"
lastUpdated	string	The Time / Date this entry was last updated
alerts	array	A collection of alert messages.
>level	string	Success, info, warning or error.
>text	string	Alert message.

### Response Example

```
{
  "alerts": [
    {
      "level": "success",
      "text": "Cachegroup successfully created: cache_group_
↪edge"
    }
  ],
  "response": {
    'longitude' : '45',
    'lastUpdated' : '2016-01-25 13:55:30',
    'shortName' : 'cg_edge',
    'name' : 'cache_group_edge',
    'parentCachegroup' : 'cache_group_mid',
    'secondaryParentCachegroup' : null,
    'latitude' : '12',
    'typeName' : 'EDGE_LOC',
    'id' : '104',
    'parentCachegroupId' : '103',
    'secondaryParentCachegroupId' : null
  }
}
```

### PUT /api/1.2/cachegroups/{:id}

Update cache group.

Authentication Required: Yes

Role(s) Required: admin or oper

#### Request Route Parameters

Name	Required	Description
id	yes	The id of the cache group to edit.

### Request Parameters

Name	Required	Description
name	yes	The name of the Cache Group entry
shortName	yes	Abbreviation of the Cache Group Name
latitude	no	Latitude for the Cache Group
longitude	no	Longitude for the Cache Group
parentCachegroup	no	Name of Parent Cache Group entry.
secondaryParentCachegroup	no	Name of Secondary Parent Cache Group entry.
typeName	yes	The type of Cache Group entry, “EDGE_LOC”, “MID_LOC” or “ORG_LOC”

### Request Example

```
{
  "name": "cache_group_edge",
  "shortName": "cg_edge",
  "latitude": 12,
  "longitude": 45,
  "parentCachegroup": "cache_group_mid",
  "typeName": "EDGE_LOC"
}
```

### Response Properties

Parameter	Type	Description
id	string	The id of cache group
name	string	The name of the Cache Group entry
shortName	string	Abbreviation of the Cache Group Name
latitude	string	Latitude for the Cache Group
longitude	string	Longitude for the Cache Group
parentCachegroup	string	Name of Parent Cache Group entry.
parentCachegroupId	string	id of Parent Cache Group entry.
secondaryParentCachegroup	string	Name of Secondary Parent Cache Group entry.
secondaryParentCachegroupId	string	id of Secondary Parent Cache Group entry.
typeName	string	The type of Cache Group entry, “EDGE_LOC”, “MID_LOC” or “ORG_LOC”
lastUpdated	string	The Time / Date this entry was last updated
alerts	array	A collection of alert messages.
>level	string	Success, info, warning or error.
>text	string	Alert message.

### Response Example

```
{
  "alerts": [
    {
```

(continues on next page)

(continued from previous page)

```

        "level": "success",
        "text": "Cachegroup was updated: cache_group_edge"
    },
    ],
    "response": {
        'longitude' : '45',
        'lastUpdated' : '2016-01-25 13:55:30',
        'shortName' : 'cg_edge',
        'name' : 'cache_group_edge',
        'parentCachegroup' : 'cache_group_mid',
        'secondaryParentCachegroup' : null,
        'latitude' : '12',
        'typeName' : 'EDGE_LOC',
        'id' : '104',
        'parentCachegroupId' : '103',
        'secondaryParentCachegroupId' : null
    }
}

```

**DELETE /api/1.2/cachegroups/{:id}**

Delete cache group. The request to delete a cache group, which has servers or child cache group, will be rejected.

Authentication Required: Yes

Role(s) Required: admin or oper

**Request Route Parameters**

Name	Required	Description
id	yes	The id of the cache group to delete.

**Response Properties**

Parameter	Type	Description
alerts	array	A collection of alert messages.
>level	string	Success, info, warning or error.
>text	string	Alert message.

**Response Example**

```

{
    "alerts": [
        {
            "level": "success",
            "text": "Cachegroup was deleted: cache_group_edge"
        }
    ],
}

```

**POST /api/1.2/cachegroups/{:id}/queue\_update**

Queue or dequeue updates for all servers assigned to a cache group limited to a specific CDN.

Authentication Required: Yes

Role(s) Required: admin or oper

**Request Route Parameters**

Name	Required	Description
id	yes	the cachegroup id.

**Request Properties**

Name	Type	Description
action	string	queue or dequeue
cdn	string	cdn name or cdn ID is required
cdnId	string	cdn ID or cdn name is required

**Response Properties**

Name	Type	Description
action	string	The action processed, queue or dequeue.
cachegroupId	integer	cachegroup id
cachegroupName	string	cachegroup name
cdn	string	cdn name
serverNames	array	servers name array in the cachegroup in cdn

**Response Example**

```
{
  "response": {
    "cachegroupName": "us-il-chicago",
    "action": "queue",
    "serverNames": [
      "atsec-chi-00",
      "atsec-chi-01",
      "atsec-chi-02",
      "atsec-chi-03",
    ],
    "cachegroupId": "93",
    "cdn": "cdn_number_1",
  }
}
```

**POST /api/1.2/cachegroups/{:id}/deliveryservices**

Assign deliveryservices for servers in cachegroup

Authentication Required: Yes

Role(s) Required: admin or oper

### Request Route Parameters

Name	Required	Description
id	yes	The cachegroup id.

### Request Properties

Parameter	Type	Description
deliveryServices	array	The Ids of the delivery services to assign to each server in the cachegroup.

### Request Example

```
{
  "deliveryServices": [ 234, 235 ]
}
```

### Response Properties

Parameter	Type	Description
response	hash	The details of the assignment, if success.
>id	int	The cachegroup id.
>serverNames	array	The server name array in the cachegroup.
>deliveryServices	array	The deliveryservice id array.
alerts	array	A collection of alert messages.
>level	string	Success, info, warning or error.
>text	string	Alert message.

### Response Example

```
{
  "response": {
    "id": 3,
    "serverNames": [ "atlanta-edge-01", "atlanta-edge-07" ],
    "deliveryServices": [ 234, 235 ]
  }
  "alerts":
  [
    {
      "level": "success",
      "text": "Delivery services successfully assigned to all the
↪servers of cache group 3."
    }
  ],
}
```



## Cache Group parameters

### /api/1.2/cachegroupparameters

#### POST /api/1.2/cachegroupparameters

Assign parameter(s) to cache group(s).

Authentication Required: Yes

Role(s) Required: Admin or Operations

#### Request Properties

Two formats are acceptable.

Single cachegroup-parameter format:

Parameter	Required	Description
cacheGroupId	yes	cache group id.
parameterId	yes	parameter id.

Profile-parameter array format:

Parameter	Required	Description
	yes	cachegroup-parameter array.
>cacheGroupId	yes	cache group id.
>parameterId	yes	parameter id.

#### Request Example

Single cachegroup-parameter format:

```
{
  "cacheGroupId": 2,
  "parameterId": 6
}
```

Cachegroup-parameter array format:

```
[
  {
    "cacheGroupId": 2,
    "parameterId": 6
  },
  {
    "cacheGroupId": 2,
    "parameterId": 7
  },
  {
    "cacheGroupId": 3,
    "parameterId": 6
  }
]
```

**\*\*Response Properties\*\***

(continues on next page)

(continued from previous page)

Parameter	Type	Description
``response``	array	Cache group-parameter associations.
>cacheGroupId	string	Cache Group id.
>parameterId	string	Parameter id.
alerts	array	A collection of alert messages.
>level	string	success, info, warning or error.
>text	string	Alert message.
version	string	

**Response Example**

```
{
  "response": [
    {
      "cacheGroupId": "2",
      "parameterId": "6"
    },
    {
      "cacheGroupId": "2",
      "parameterId": "7"
    },
    {
      "cacheGroupId": "3",
      "parameterId": "6"
    }
  ]
  "alerts": [
    {
      "level": "success",
      "text": "Cache group parameter associations were created."
    }
  ]
}
```

(continues on next page)

(continued from previous page)

```
]
}
```

**DELETE /api/1.2/cachegroupparameters/{:cachegroup\_id}/{:parameter\_id}**

Delete a cache group parameter association.

Authentication Required: Yes

Role(s) Required: Admin or Operations

**Request Route Parameters**

Name	Required	Description
cachegroup_id	yes	cache group id.
parameter_id	yes	parameter id.

**Response Properties**

Parameter	Type	Description
alerts	array	A collection of alert messages.
>level	string	success, info, warning or error.
>text	string	Alert message.
version	string	

**Response Example**

```
{
  "alerts": [
    {
      "level": "success",
      "text": "Cache group parameter association was deleted."
    }
  ]
}
```

**Cache Statistics****/api/1.2/cache\_stats****GET /api/1.2/cache\_stats.json**

Retrieves statistics about the CDN.

Authentication Required: Yes

Role(s) Required: None

### Request Query Parameters

Name	Re-quired	Description
cdnName	yes	The CDN name to return cache stats for
metricType	yes	The metric type (valid metric types: 'ats.proxy.process.http.current_client_connections', 'bandwidth', 'maxKbps')
startDate	yes	The begin date (Formatted as ISO8601, for example: '2015-08-11T12:30:00-06:00')
endDate	yes	The end date (Formatted as ISO8601, for example: '2015-08-12T12:30:00-06:00')

### Response Properties

Parameter	Type	Description
summary	hash	Summary data
>count	int	
>min	float	
>max	float	
>fifthPercentile	float	
>ninetyEighthPercentile	float	
>ninetyFifthPercentile	float	
>average	float	
series	hash	Series data
>count	int	
>columns	array	
>name	string	
>values	array	
>>time	string	
>>value	float	

### Response Example

```
{
  "response": {
    "series": {
      "columns": [
        "time",
        ""
      ],
      "count": 29,
      "name": "bandwidth",
      "tags": {
        "cdn": "over-the-top"
      },
      "values": [
        [
          "2015-08-10T22:40:00Z",
          229340299720
        ],
        [
          "2015-08-10T22:41:00Z",
```

(continues on next page)

(continued from previous page)

```

        224309221713.334
      ],
      [
        "2015-08-10T22:42:00Z",
        229551834168.334
      ],
      [
        "2015-08-10T22:43:00Z",
        225179658876.667
      ],
      [
        "2015-08-10T22:44:00Z",
        230443968275
      ]
    ]
  },
  "summary": {
    "average": 970410.295,
    "count": 1376041798,
    "fifthPercentile": 202.03,
    "max": 3875441.02,
    "min": 0,
    "ninetyEighthPercentile": 2957940.93,
    "ninetyFifthPercentile": 2366728.63
  }
}

```

## Capabilities

### /api/1.2/capabilities

#### GET /api/1.2/capabilities

Get all capabilities.

Authentication Required: Yes

Role(s) Required: None

#### Response Properties

Parameter	Type	Description
name	string	Capability name.
description	string	Describing the APIs covered by the capability.
lastUpdated	string	

#### Response Example

```

{
  "response": [

```

(continues on next page)

(continued from previous page)

```
{
  {
    "name": "cdn-read",
    "description": "View CDN configuration",
    "lastUpdated": "2017-04-02 08:22:43"
  },
  {
    "name": "cdn-write",
    "description": "Create, edit or delete CDN configuration",
    "lastUpdated": "2017-04-02 08:22:43"
  }
}
```

### GET /api/1.2/capabilities/:name

Get a capability by name.

Authentication Required: Yes

Role(s) Required: None

#### Request Route Parameters

Name	Required	Type	Description
name	yes	string	Capability name.

#### Response Properties

Parameter	Type	Description
name	string	Capability name.
description	string	Describing the APIs covered by the capability.
lastUpdated	string	

#### Response Example

```
{
  "response": [
    {
      "name": "cdn-read",
      "description": "View CDN configuration",
      "lastUpdated": "2017-04-02 08:22:43"
    }
  ]
}
```

### POST /api/1.2/capabilities

Create a capability.

Authentication Required: Yes

Role(s) Required: admin or oper

### Request Parameters

Name	Required	Type	Description
name	yes	string	Capability name.
description	yes	string	Describing the APIs covered by the capability.

### Request Example

```
{
  "name": "cdn-write",
  "description": "Create, edit or delete CDN configuration"
}
```

### Response Properties

Parameter	Type	Description
response	hash	The details of the creation, if success.
>name	string	Capability name.
>description	string	Describing the APIs covered by the capability.
alerts	array	A collection of alert messages.
>level	string	Success, info, warning or error.
>text	string	Alert message.

### Response Example

```
{
  "response": {
    "name": "cdn-write",
    "description": "Create, edit or delete CDN configuration"
  },
  "alerts": [
    {
      "level": "success",
      "text": "Capability was created."
    }
  ]
}
```

## PUT /api/1.2/capabilities/{:name}

Edit a capability.

Authentication Required: Yes

Role(s) Required: admin or oper

### Request Route Parameters

Name	Type	Description
name	int	Capability name.

### Request Properties

Parameter	Type	Description
description	string	Describing the APIs covered by the capability.

### Request Example

```
{
  "description": "View CDN configuration"
}
```

### Response Properties

Parameter	Type	Description
response	hash	The details of the update, if success.
>name	string	Capability name.
>description	int	Describing the APIs covered by the capability.
alerts	array	A collection of alert messages.
>level	string	Success, info, warning or error.
>text	string	Alert message.

### Response Example

```
{
  "response": {
    "name": "cdn-read",
    "description": "View CDN configuration"
  },
  "alerts": [
    {
      "level": "success",
      "text": "Capability was updated."
    }
  ]
}
```

## DELETE /api/1.2/capabilities/{:name}

Delete a capability.

Authentication Required: Yes

Role(s) Required: admin or oper

### Request Route Parameters

Name	Required	Description
name	yes	Capability name.



## Response Properties

Parameter	Type	Description
alerts	array	A collection of alert messages.
>level	string	success, info, warning or error.
>text	string	Alert message.

## Response Example

```
{
  "alerts": [
    {
      "level": "success",
      "text": "Capability deleted."
    }
  ],
}
```

## CDN

### /api/1.2/cdns

#### GET /api/1.2/cdns

Authentication Required: Yes

Role(s) Required: None

## Response Properties

Parameter	Type	Description
id	string	CDN id.
name	string	CDN name.
domainName	string	TLD of the CDN.
dnssecEnabled	bool	DNSSEC enabled.
lastUpdated	string	

## Response Example

```
{
  "response": [
    {
      "id": "1"
      "name": "cdn1",
      "domainName": "cdn1.foo.com",
      "dnssecEnabled": false,
      "lastUpdated": "2014-10-02 08:22:43"
    },
    {
      "id": "2"
      "name": "cdn2",
      "domainName": "cdn2.foo.com",
      "dnssecEnabled": true,
      "lastUpdated": "2014-10-02 08:22:43"
    }
  ]
}
```

(continues on next page)

(continued from previous page)

```
}
  ]
}
```

**GET /api/1.2/cdns/:id**

Authentication Required: Yes

Role(s) Required: None

**Request Route Parameters**

Name	Required	Description
id	yes	CDN id.

**Response Properties**

Parameter	Type	Description
id	string	CDN id.
name	string	CDN name.
domainName	string	TLD of the CDN.
dnssecEnabled	bool	DNSSEC enabled.
lastUpdated	string	

**Response Example**

```
{
  "response": [
    {
      "id": "2"
      "name": "cdn2",
      "domainName": "cdn2.foo.com",
      "dnssecEnabled": false,
      "lastUpdated": "2014-10-02 08:22:43"
    }
  ]
}
```

**GET /api/1.2/cdns/name/:name**

Authentication Required: Yes

Role(s) Required: None

**Request Route Parameters**

Name	Required	Description
name	yes	CDN name.

### Response Properties

Parameter	Type	Description
id	string	CDN id.
name	string	CDN name.
domainName	string	TLD of the CDN.
dnssecEnabled	bool	DNSSEC enabled.
lastUpdated	string	

### Response Example

```
{
  "response": [
    {
      "id": "2"
      "name": "cdn2",
      "domainName": "cdn2.foo.com",
      "dnssecEnabled": false,
      "lastUpdated": "2014-10-02 08:22:43"
    }
  ]
}
```

## POST /api/1.2/cdns

Allows user to create a CDN.

Authentication Required: Yes

Role(s) Required: admin or oper

### Request Parameters

Parameter	Type	Description
name	string	CDN name.
domainName	string	TLD of the CDN.
dnssecEnabled	bool	Whether dnssec is enabled. - false: disabled - true: enabled

### Request Example

```
{
  "name": "cdn_test",
  "domainName": "cdn3.foo.com",
  "dnssecEnabled": true
}
```

### Response Properties

Parameter	Type	Description
response	hash	The details of the creation, if success.
>id	int	CDN id.
>name	string	CDN name.
>dnssecEnabled	string	Whether dnssec is enabled.
>domainName	string	TLD of the CDN.
alerts	array	A collection of alert messages.
>level	string	Success, info, warning or error.
>text	string	Alert message.

### Response Example

```
{
  "response": {
    "id": 3
    "name": "cdn_test",
    "domainName": "cdn3.foo.com",
    "dnssecEnabled": true
  },
  "alerts": [
    {
      "level": "success",
      "text": "cdn was created."
    }
  ]
}
```

### PUT /api/1.2/cdns/{:id}

Allows user to edit a CDN.

Authentication Required: Yes

Role(s) Required: admin or oper

#### Request Route Parameters

Name	Type	Description
id	int	CDN id.

#### Request Properties

Parameter	Type	Description
name	string	CDN name.
domainName	string	TLD of the CDN.
dnssecEnabled	bool	Whether dnssec is enabled. - false: disabled - true: enabled

#### Request Example

```
{
  "name": "cdn_test2",
  "domainName": "cdn3.foo.com",
  "dnssecEnabled": false
}
```

### Response Properties

Parameter	Type	Description
response	hash	The details of the update, if success.
>name	string	CDN name.
>id	int	CDN id.
>domainName	string	TLD of the CDN.
>dnssecEnabled	bool	Whether dnssec is enabled.
alerts	array	A collection of alert messages.
>level	string	Success, info, warning or error.
>text	string	Alert message.

### Response Example

```
{
  "response":{
    "id": 3,
    "name": "cdn_test2",
    "domainName": "cdn3.foo.com",
    "dnssecEnabled": false
  },
  "alerts":[
    {
      "level": "success",
      "text": "cdn was updated."
    }
  ]
}
```

## DELETE /api/1.2/cdns/{:id}

Allows user to delete a CDN.

Authentication Required: Yes

Role(s) Required: admin or oper

### Request Route Parameters

Name	Required	Description
id	yes	CDN id.

### Response Properties

Parameter	Type	Description
alerts	array	A collection of alert messages.
>level	string	success, info, warning or error.
>text	string	Alert message.

### Response Example

```
{
  "alerts": [
    {
      "level": "success",
      "text": "cdn was deleted."
    }
  ],
}
```

### POST /api/1.2/cdns/{:id}/queue\_update

Queue or dequeue updates for all servers assigned to a specific CDN.

Authentication Required: Yes

Role(s) Required: admin or oper

#### Request Route Parameters

Name	Required	Description
id	yes	the cdn id.

#### Request Properties

Name	Type	Description
action	string	queue or dequeue

#### Request Example

```
{
  "action": "queue"
}
```

#### Response Properties

Name	Type	Description
action	string	The action processed, queue or dequeue.
cdnId	integer	cdn id

#### Response Example

```
{
  "response": {
    "action": "queue",
    "cdn": 1
  }
}
```

## Health

### GET /api/1.2/cdns/health

Retrieves the health of all locations (cache groups) for all CDNs.

Authentication Required: Yes

Role(s) Required: None

#### Response Properties

Parameter	Type	Description
totalOnline	int	Total number of online caches across all CDNs.
totalOffline	int	Total number of offline caches across all CDNs.
cachegroups	array	A collection of cache groups.
>online	int	The number of online caches for the cache group
>offline	int	The number of offline caches for the cache group.
>name	string	Cache group name.

#### Response Example

```
{
  "response": {
    "totalOnline": 148,
    "totalOffline": 0,
    "cachegroups": [
      {
        "online": 8,
        "offline": 0,
        "name": "us-co-denver"
      },
      {
        "online": 7,
        "offline": 0,
        "name": "us-de-newcastle"
      }
    ]
  },
}
```

**GET /api/1.2/cdns/:name/health**

Retrieves the health of all locations (cache groups) for a given CDN.

Authentication Required: Yes

Role(s) Required: None

**Request Route Parameters**

Name	Required	Description
name	yes	

**Response Properties**

Parameter	Type	Description
totalOnline	int	Total number of online caches across the specified CDN.
totalOffline	int	Total number of offline caches across the specified CDN.
cachegroups	array	A collection of cache groups.
>online	int	The number of online caches for the cache group
>offline	int	The number of offline caches for the cache group.
>name	string	Cache group name.

**Response Example**

```
{
  "response": {
    "totalOnline": 148,
    "totalOffline": 0,
    "cachegroups": [
      {
        "online": 8,
        "offline": 0,
        "name": "us-co-denver"
      },
      {
        "online": 7,
        "offline": 0,
        "name": "us-de-newcastle"
      }
    ]
  },
}
```

**GET /api/1.2/cdns/usage/overview**

Retrieves the high-level CDN usage metrics.

Authentication Required: Yes

Role(s) Required: None

**Response Properties**



Parameter	Type	Description
currentGbps	number	
tps	int	
maxGbps	int	

**Response Example**

```
{
  "response": {
    "currentGbps": 149.368167,
    "tps": 36805,
    "maxGbps": 3961
  }
}
```

**GET /api/1.2/cdns/capacity**

Retrieves the aggregate capacity percentages of all locations (cache groups) for a given CDN.

Authentication Required: Yes

Role(s) Required: None

**Response Properties**

Parameter	Type	Description
availablePercent	number	
unavailablePercent	number	
utilizedPercent	number	
maintenancePercent	number	

**Response Example**

```
{
  "response": {
    "availablePercent": 89.0939840205533,
    "unavailablePercent": 0,
    "utilizedPercent": 10.9060020300395,
    "maintenancePercent": 0.0000139494071146245
  }
}
```

**Routing****GET /api/1.2/cdns/routing**

Retrieves the aggregate routing percentages of all locations (cache groups) for a given CDN.

Authentication Required: Yes

Role(s) Required: None

#### Response Properties

Parameter	Type	Description
staticRoute	number	Used pre-configured DNS entries.
miss	number	No location available for client IP.
geo	number	Used 3rd party geo-IP mapping.
err	number	Error localizing client IP.
cz	number	Used Coverage Zone geo-IP mapping.
dsrc	number	Overflow traffic sent to secondary CDN.

#### Response Example

```
{
  "response": {
    "staticRoute": 0,
    "miss": 0,
    "geo": 37.8855391018869,
    "err": 0,
    "cz": 62.1144608981131,
    "dsrc": 0
  }
}
```

## Metrics

### GET /api/1.2/cdns/metric\_types/:metric/start\_date/:start/end\_date/:end

Retrieves edge metrics of one or all locations (cache groups).

Authentication Required: Yes

Role(s) Required: None

#### Request Route Parameters

Name	Required	Description
metric_type	yes	ooff, origin_tps
start	yes	UNIX time, yesterday, now
end	yes	UNIX time, yesterday, now

#### Response Properties

Parameter	Type	Description
stats	hash	
>count	string	
>98thPercentile	string	
>min	string	
>max	string	
>5thPercentile	string	
>95thPercentile	string	
>mean	string	
>sum	string	
data	array	
>time	int	
>value	number	
label	string	

### Response Example

```
{
  "response": [
    {
      "stats": {
        "count": 1,
        "98thPercentile": 1668.03,
        "min": 1668.03,
        "max": 1668.03,
        "5thPercentile": 1668.03,
        "95thPercentile": 1668.03,
        "mean": 1668.03,
        "sum": 1668.03
      },
      "data": [
        [
          1425135900000,
          1668.03
        ],
        [
          1425136200000,
          null
        ]
      ],
      "label": "Origin TPS"
    }
  ],
}
```

## Domains

### GET /api/1.2/cdns/domains

Authentication Required: Yes

Role(s) Required: None

### Response Properties

Parameter	Type	Description
profileId	string	
parameterId	string	
profileName	string	
profileDescription	string	
domainName	string	

### Response Example

```
{
  "response": [
    {
      "profileId": "5",
      "parameterId": "404",
      "profileName": "CR_FOO",
      "profileDescription": "Content Router for foo.domain.net",
      "domainName": "foo.domain.net"
    },
    {
      "profileId": "8",
      "parameterId": "405",
      "profileName": "CR_BAR",
      "profileDescription": "Content Router for bar.domain.net",
      "domainName": "bar.domain.net"
    }
  ],
}
```

## Topology

### GET /api/1.2/cdns/:cdn\_name/configs

Retrieves CDN config information.

Authentication Required: Yes

### Request Route Parameters

Name	Required	Description
cdn_name	yes	Your cdn name or, all

### Response Properties

Parameter	Type	Description
id	string	
value	string	
name	string	
config_file	string	

**Response Example**

TBD

**GET /api/1.2/cdns/:name/configs/monitoring**

Retrieves CDN monitoring information.

Authentication Required: Yes

Role(s) Required: None

**Request Route Parameters**

Name	Required	Description
name	yes	CDN name

**Response Properties**

Parameter	Type	Description
trafficServers	array	A collection of Traffic Servers.
>profile	string	
>ip	string	
>status	string	
>cacheGroup	string	
>ip6	string	
>port	int	
>hostName	string	
>fqdn	string	
>interfaceName	string	
>type	string	
>hashId	string	
cacheGroups	array	A collection of cache groups.
>coordinates	hash	
>>longitude	number	
>>latitude	number	
>name	string	
config	hash	
>hack.ttl	int	
>tm.healthParams.polling.url	string	
>tm.dataServer.polling.url	string	
>health.timepad	int	
>tm.polling.interval	int	
>health.threadPool	int	
>health.polling.interval	int	
>health.event-count	int	
>tm.crConfig.polling.url	number	
>CDN_name	number	
trafficMonitors	array	A collection of Traffic Monitors.

Continued on next page

Table 9 – continued from previous page

Parameter	Type	Description
>profile	string	
>location	string	
>ip	string	
>status	string	
>ip6	string	
>port	int	
>hostName	string	
>fqdn	string	
deliveryServices	array	A collection of delivery services.
>xmlId	string	
>totalTpsThreshold	int	
>status	string	
>totalKbpsThreshold	int	
profiles	array	A collection of profiles.
>parameters	hash	
>>health.connection.timeout	int	
>>health.polling.url	string	
>>health.threshold.queryTime	int	
>>history.count	int	
>>health.threshold.availableBandwidthInKbps	string	
>>health.threshold.loadavg	string	
>name	string	
>type	string	

**Response Example**

TBD

**GET /api/1.2/cdns/:name/configs/routing**

Retrieves CDN routing information.

Authentication Required: Yes

Role(s) Required: None

**Request Route Parameters**

Name	Required	Description
name	yes	

**Response Properties**

Parameter	Type	Description
trafficServers	array	A collection of Traffic Servers.
>profile	string	
>ip	string	

Continued on next page

Table 10 – continued from previous page

Parameter	Type	Description
>status	string	
>cacheGroup	string	
>ip6	string	
>port	int	
>deliveryServices	array	
>>xmlId	string	
>>remaps	array	
>>hostName	string	
>fqdn	string	
>interfaceName	string	
>type	string	
>hashId	string	
stats	hash	
>trafficOpsPath	string	
>cdnName	string	
>trafficOpsVersion	string	
>trafficOpsUser	string	
>date	int	
>trafficOpsHost	string	
cacheGroups	array	A collection of cache groups.
>coordinates	hash	
>>longitude	number	
>>latitude	number	
>name	string	
config	hash	
>tld.soa.admin	string	
>tcoveragezone.polling.interval	int	
>geolocation.polling.interval	int	
>tld.soa.expire	int	
>coveragezone.polling.url	string	
>tld.soa.minimum	int	
>geolocation.polling.url	string	
>domain_name	string	
>tld.ttls.AAAA	int	
>tld.soa.refresh	int	
>tld.ttls.NS	int	
>tld.ttls.SOA	int	
>geolocation6.polling.interval	int	
>tld.ttls.A	int	
>tld.soa.retry	int	
>geolocation6.polling.url	string	
trafficMonitors	array	A collection of Traffic Monitors.
>profile	string	
>location	string	
>ip	string	
>status	string	
>ip6	string	
>port	int	
>hostName	string	

Continued on next page

Table 10 – continued from previous page

Parameter	Type	Description
>fqdn	string	
deliveryServices	array	A collection of delivery services.
>xmlId	string	
>ttl	int	
>geoEnabled	string	
>coverageZoneOnly	boolean	
>matchSets	array	
>>protocol	string	
>>matchList	array	
>>>regex	string	
>>>matchType	string	
>bypassDestination	hash	
>>maxDnsIpsForLocation	int	
>>ttl	int	
>>type	string	
>ttls	hash	
>>A	int	
>>SOA	int	
>>NS	int	
>>AAAA	int	
>missCoordinates	hash	
>>longitude	number	
>>latitude	number	
>soa	hash	
>>admin	string	
>>retry	int	
>>minimum	int	
>>refresh	int	
>>expire	int	
trafficRouters	hash	
>profile	int	
>location	string	
>ip	string	
>status	string	
>ip6	string	
>port	int	
>hostName	string	
>fqdn	string	
>apiPort	int	

**Response Example**

TBD
-----



## DNSSEC Keys

### GET /api/1.2/cdns/name/:name/dnsseckeys

Gets a list of dnsseckeys for a CDN and all associated Delivery Services.

Authentication Required: Yes

Role(s) Required: Admin

#### Request Route Parameters

Name	Required	Description
name	yes	

#### Response Properties

Parameter	Type	Description
cdn name/ds xml_id	string	identifier for ds or cdn
>zsk/ksk	array	collection of zsk/ksk data
>>ttl	string	time-to-live for dnssec requests
>>inceptionDate	string	epoch timestamp for when the keys were created
>>expirationDate	string	epoch timestamp representing the expiration of the keys
>>private	string	encoded private key
>>public	string	encoded public key
>>name	string	domain name
version	string	API version
ksk>>dsRecord>>algorithm	string	The algorithm of the referenced DNSKEY-record.
ksk>>dsRecord>>digestType	string	Cryptographic hash algorithm used to create the Digest value.
ksk>>dsRecord>>digest	string	A cryptographic hash value of the referenced DNSKEY-record.

#### Response Example

```
{
  "response": {
    "cdn1": {
      "zsk": {
        "ttl": "60",
        "inceptionDate": "1426196750",
        "private": "zsk private key",
        "public": "zsk public key",
        "expirationDate": "1428788750",
        "name": "foo.kabletown.com."
      },
      "ksk": {
        "name": "foo.kabletown.com.",
        "expirationDate": "1457732750",
        "public": "ksk public key",
        "private": "ksk private key",
        "inceptionDate": "1426196750",
        "ttl": "60",
        dsRecord: {
```

(continues on next page)

(continued from previous page)

```

        "algorithm": "5",
        "digestType": "2",
        "digest": "abc123def456"
    }
},
"ds-01": {
    "zsk": {
        "ttl": "60",
        "inceptionDate": "1426196750",
        "private": "zsk private key",
        "public": "zsk public key",
        "expirationDate": "1428788750",
        "name": "ds-01.foo.kabletown.com."
    },
    "ksk": {
        "name": "ds-01.foo.kabletown.com.",
        "expirationDate": "1457732750",
        "public": "ksk public key",
        "private": "ksk private key",
        "inceptionDate": "1426196750"
    }
},
... repeated for each ds in the cdn
},
}

```

**GET /api/1.2/cdns/name/:name/dnsseckey/delete**

Delete dnssec keys for a cdn and all associated delivery services.

Authentication Required: Yes

Role(s) Required: Admin

**Request Route Parameters**

Name	Required	Description
name	yes	name of the CDN for which you want to delete dnssec keys

**Response Properties**

Parameter	Type	Description
response	string	success response

**Response Example**

```

{
  "response": "Successfully deleted dnssec keys for <cdn>"
}

```

**POST /api/1.2/deliveryservices/dnsseckey/generate**

Generates ZSK and KSK keypairs for a CDN and all associated Delivery Services.

Authentication Required: Yes

Role(s) Required: Admin

**Request Properties**

Parameter	Type	Description
key	string	name of the cdn
name	string	domain name of the cdn
ttl	string	time to live
kskExpirationDays	string	Expiration (in days) for the key signing keys
zskExpirationDays	string	Expiration (in days) for the zone signing keys

**Request Example**

```
{
  "key": "cdn1",
  "name": "ott.kabletown.com",
  "ttl": "60",
  "kskExpirationDays": "365",
  "zskExpirationDays": "90"
}
```

**Response Properties**

Parameter	Type	Description
response	string	response string
version	string	API version

**Response Example**

```
{
  "response": "Successfully created dnssec keys for cdn1"
}
```

**SSL Keys****GET /api/1.2/cdns/name/:name/sslkeys**

Returns ssl certificates for all Delivery Services that are a part of the CDN.

Authentication Required: Yes

Role(s) Required: Admin

**Request Route Parameters**

Name	Required	Description
name	yes	

### Response Properties

Parameter	Type	Description
deliveryservice	string	identifier for deliveryservice xml_id
certificate	array	collection of certificate
>>key	string	base64 encoded private key for ssl certificate
>>crt	string	base64 encoded ssl certificate

### Response Example

```
{
  "response": [
    {
      "deliveryservice": "ds1",
      "certificate": {
        "crt": "base64encodedcrt1",
        "key": "base64encodedkey1"
      }
    },
    {
      "deliveryservice": "ds2",
      "certificate": {
        "crt": "base64encodedcrt2",
        "key": "base64encodedkey2"
      }
    }
  ]
}
```

## Change Logs

### /api/1.2/logs

#### GET /api/1.2/logs

Authentication Required: Yes

Role(s) Required: None

#### Request Query Parameters

Name	Required	Description
days	no	The number of days of change logs to return.
limit	no	The number of rows to limit the response to.

### Response Properties

Parameter	Type	Description
ticketNum	string	Optional field to cross reference with any bug tracking systems
level	string	Log categories for each entry, examples: 'UICHANGE', 'OPER', 'APICHANGE'.
lastUpdated	string	Local unique identifier for the Log
user	string	Current user who made the change that was logged
id	string	Local unique identifier for the Log entry
message	string	Log detail about what occurred

### Response Example

```
{
  "response": [
    {
      "ticketNum": null,
      "level": "OPER",
      "lastUpdated": "2015-02-04 22:59:13",
      "user": "userid852",
      "id": "22661",
      "message": "Snapshot CRConfig created."
    },
    {
      "ticketNum": null,
      "level": "APICHANGE",
      "lastUpdated": "2015-02-03 17:04:20",
      "user": "userid853",
      "id": "22658",
      "message": "Update server odol-atsec-nyc-23.kbaletown.net_
↪status=REPORTED"
    }
  ],
}
```

### GET /api/1.2/logs/:days/days

Authentication Required: Yes

Role(s) Required: None

#### Request Route Parameters

Name	Required	Description
days	yes	Number of days.

#### Response Properties

Parameter	Type	Description
ticketNum	string	
level	string	
lastUpdated	string	
user	string	
id	string	
message	string	

### Response Example

```
{
  "response": [
    {
      "ticketNum": null,
      "level": "OPER",
      "lastUpdated": "2015-02-04 22:59:13",
      "user": "userid852",
      "id": "22661",
      "message": "Snapshot CRConfig created."
    },
    {
      "ticketNum": null,
      "level": "APICHANGE",
      "lastUpdated": "2015-02-03 17:04:20",
      "user": "userid853",
      "id": "22658",
      "message": "Update server odol-atsec-nyc-23.kabletown.net_
↪status=REPORTED"
    }
  ],
}
```

### GET /api/1.2/logs/newcount

Authentication Required: Yes

Role(s) Required: None

#### Response Properties

Parameter	Type	Description
newLogcount	string	

### Response Example

```
{
  "response": {
    "newLogcount": 0
  }
}
```

Config Files and Config File Metadata ===

**/api/1.2/servers/:hostname/configfiles/ats****GET /api/1.2/servers/:hostname/configfiles/ats** **GET /api/1.2/servers/:host\_id/configfiles/ats**

Authentication Required: Yes

Role(s) Required: Operator

**Request Query Parameters****Response Properties**

Info Section			
Parameter	Type	Description	
profileId	int	The ID of the profile assigned to the cache.	
profileName	string	The name of the profile assigned to the cache.	
toRevProxyUrl	string	The configured reverse proxy cache for configfile requests.	
toURL	string	The configured URL for Traffic Ops.	
serverIpv4	string	The configured IP address of the cache.	
serverName	string	The cache's short form hostname.	
serverId	int	The cache's Traffic Ops ID.	
cdnId	int	The ID of the cache's assigned CDN.	
cdnName	string	The name of the cache's assigned CDN.	
serverTcpPort	int	The configured port of the server's used by ATS.	
configFiles Section			
Parameter	Type	Description	
fnameOnDisk	string	The filename of the configuration file as stored on the cache.	
location	string	The directory location of the configuration file as stored on the cache.	
apiUri	string	The path to generate the configuration file from Traffic Ops.	
scope	string	The scope of the configuration file.	

**Response Example**

```
{
  "info": {
    "profileId": 278,
    "toRevProxyUrl": "https://to.example.com:81",
    "toUrl": "https://to.example.com/",
    "serverIpv4": "192.168.1.5",
    "serverTcpPort": 80,
    "serverName": "cache-ats-01",
    "cdnId": 1,
    "cdnName": "CDN_1",
```

(continues on next page)

(continued from previous page)

```

    "serverId": 21,
    "profileName": "EDGE_CDN_1_EXAMPLE"
  },
  "configFiles": [
    {
      "fnameOnDisk": "remap.config",
      "location": "/opt/trafficserver/etc/trafficserver",
      "apiUri": "/api/1.2/profiles/EDGE_CDN_1_EXAMPLE/configfiles/ats/remap.
↪config",
      "scope": "profiles"
    },
    {
      "fnameOnDisk": "ssl_multicert.config",
      "location": "/opt/trafficserver/etc/trafficserver",
      "apiUri": "/api/1.2/cdns/CDN_1/configfiles/ats/ssl_multicert.config",
      "scope": "cdns"
    },
    {
      "fnameOnDisk": "parent.config",
      "location": "/opt/trafficserver/etc/trafficserver",
      "apiUri": "/api/1.2/servers/cache-ats-01/configfiles/ats/parent.config"
    }
  ]
}

```

### **/api/1.2/servers/:hostname/configfiles/ats/:configfile**

**GET /api/1.2/servers/:hostname/configfiles/ats/:configfile** **GET /api/1.2/servers/:host\_id/configfiles/ats/:configfile**

Authentication Required: Yes

Role(s) Required: Operator

#### **Request Query Parameters**

#### **Response Properties**

Returns the requested configuration file for download. If scope used is incorrect for the config file requested, returns a 404 with the correct scope.

#### **Response Example**

```

{
  "alerts": [
    {
      "level": "error",
      "text": "Error - incorrect file scope for route used. Please use the
↪profiles route."
    }
  ]
}

```

### **/api/1.2/profiles/:profile\_name/configfiles/ats/:configfile**

**GET /api/1.2/profiles/:profile\_name/configfiles/ats/:configfile** **GET /api/1.2/profiles/:profile\_id/configfiles/ats/:configfile**



Authentication Required: Yes

Role(s) Required: Operator

#### Request Query Parameters

#### Response Properties

Returns the requested configuration file for download. If scope used is incorrect for the config file requested, returns a 404 with the correct scope.

#### Response Example

```
{
  "alerts": [
    {
      "level": "error",
      "text": "Error - incorrect file scope for route used. Please use the_
↪cdns route."
    }
  ]
}
```

### /api/1.2/cdns/:cdn\_name/configfiles/ats/:configfile

**GET /api/1.2/cdns/:cdn\_name/configfiles/ats/:configfile** **GET /api/1.2/cdns/:cdn\_id/configfiles/ats/:configfile**

Authentication Required: Yes

Role(s) Required: Operator

#### Request Query Parameters

#### Response Properties

Returns the requested configuration file for download. If scope used is incorrect for the config file requested, returns a 404 with the correct scope.

#### Response Example

```
{
  "alerts": [
    {
      "level": "error",
      "text": "Error - incorrect file scope for route used. Please use the_
↪servers route."
    }
  ]
}
```

## Delivery Service

### /api/1.2/deliveryservices

**GET /api/1.2/deliveryservices**

Retrieves all delivery services (if admin or ops) or all delivery services assigned to user. See also [Using Traffic Ops - Delivery Service](#).

Authentication Required: Yes

Role(s) Required: None

### Request Query Parameters

Name	Required	Description
cdn	no	Filter delivery services by CDN ID.
profile	no	Filter delivery services by Profile ID.
tenant	no	Filter delivery services by Tenant ID.
type	no	Filter delivery services by Type ID.
logsEnabled	no	Filter by logs enabled (true/false).

### Response Properties

Parameter	Type	Description
active	bool	true if active, false if inactive.
cacheurl	string	Cache URL rule to apply to this delivery service.
ccrDnsTtl	int	The TTL of the DNS response for A or AAAA queries requesting the IP address of the tr. host.
cdnId	int	Id of the CDN to which the delivery service belongs to.
cdnName	string	Name of the CDN to which the delivery service belongs to.
checkPath	string	The path portion of the URL to check this deliveryservice for health.
deepCachingType	string	When to do Deep Caching for this Delivery Service: <ul style="list-style-type: none"><li>• NEVER (default)</li><li>• ALWAYS</li></ul>
displayName	string	The display name of the delivery service.
dnsBypassCname	string	
dnsBypassIp	string	The IPv4 IP to use for bypass on a DNS deliveryservice - bypass starts when serving more than the globalMaxMbps traffic on this deliveryservice.
dnsBypassIp6	string	The IPv6 IP to use for bypass on a DNS deliveryservice - bypass starts when serving more than the globalMaxMbps traffic on this deliveryservice.
dnsBypassTtl	int	The TTL of the DNS bypass response.
dscp	int	The Differentiated Services Code Point (DSCP) with which to mark downstream (EDGE -> customer) traffic.

Continued on next page

Table 11 – continued from previous page

Parameter	Type	Description
edgeHeaderRewrite	string	The EDGE header rewrite actions to perform.
exampleURLs	array	Entry points into the CDN for this deliveryservice.
geoLimitRedirectUrl	string	
geoLimit	int	<ul style="list-style-type: none"> <li>• 0: None - no limitations</li> <li>• 1: Only route on CZF file hit</li> <li>• 2: Only route on CZF hit or when from USA</li> </ul> <p>Note that this does not prevent access to content or makes content secure; it just prevents routing to the content by Traffic Router.</p>
geoLimitCountries	string	
geoProvider	int	
globalMaxMbps	int	The maximum global bandwidth allowed on this deliveryservice. If exceeded, the traffic routes to the dnsByPassIp* for DNS deliveryservices and to the httpBypassFqdn for HTTP deliveryservices.
globalMaxTps	int	The maximum global transactions per second allowed on this deliveryservice. When this is exceeded traffic will be sent to the dnsByPassIp* for DNS deliveryservices and to the httpBypassFqdn for HTTP deliveryservices
httpBypassFqdn	string	The HTTP destination to use for bypass on an HTTP deliveryservice - bypass starts when serving more than the globalMaxMbps traffic on this deliveryservice.
id	int	The deliveryservice id (database row number).
infoUrl	string	Use this to add a URL that points to more information about that deliveryservice.
initialDispersion	int	
ipv6RoutingEnabled	bool	false: send IPv4 address of Traffic Router to client on HTTP type del.
lastUpdated	string	
logsEnabled	bool	

Continued on next page

Table 11 – continued from previous page

Parameter	Type	Description
longDesc	string	Description field.
longDesc1	string	Description field 1.
longDesc2	string	Description field 2.
maxDnsAnswers	int	The maximum number of IPs to put in a A/AAAA response for a DNS deliveryservice (0 means all available).
midHeaderRewrite	string	The MID header rewrite actions to perform.
missLat	float	The latitude as decimal degrees to use when the client cannot be found in the CZF or the Geo lookup. • e.g. 39.7391500 or null
missLong	float	The longitude as decimal degrees to use when the client cannot be found in the CZF or the Geo lookup. • e.g. -104.9847000 or null
multiSiteOrigin	bool	Is the Multi Site Origin feature enabled for this delivery service (0=false, 1=true). See <i>Multi Site Origin</i>
orgServerFqdn	string	The origin server base URL (FQDN when used in this instance, includes the protocol ( <a href="http://">http://</a> or <a href="https://">https://</a> ) for use in retrieving content from the origin server.
originShield	string	
profileDescription	string	The description of the Traffic Router Profile with which this deliveryservice is associated.
profileId	int	The id of the Traffic Router Profile with which this deliveryservice is associated.
profileName	string	The name of the Traffic Router Profile with which this deliveryservice is associated.
protocol	int	<ul style="list-style-type: none"> <li>• 0: serve with <a href="http://">http://</a> at EDGE</li> <li>• 1: serve with <a href="https://">https://</a> at EDGE</li> <li>• 2: serve with both <a href="http://">http://</a> and <a href="https://">https://</a> at EDGE</li> </ul>

Continued on next page

Table 11 – continued from previous page

Parameter	Type	Description
qstringIgnore	int	<ul style="list-style-type: none"> <li>• 0: no special query string handling; it is for use in the cache-key and pass up to origin.</li> <li>• 1: ignore query string in cache-key, but pass it up to parent and or origin.</li> <li>• 2: drop query string at edge, and do not use it in the cache-key.</li> </ul>
rangeRequestHandling	int	<p>How to treat range requests:</p> <ul style="list-style-type: none"> <li>• 0 Do not cache (ranges requested from files that are already cached due to a non range request will be a HIT)</li> <li>• 1 Use the <code>background_fetch</code> plugin.</li> <li>• 2 Use the <code>cache_range_requests</code> plugin.</li> </ul>
regexRemap	string	Regex Remap rule to apply to this delivery service at the Edge tier.
regionalGeoBlocking	bool	Regex Remap rule to apply to this delivery service at the Edge tier.
remapText	string	Additional raw remap line text.
routingName	string	The routing name of this deliveryservice, e.g. <code>&lt;routing-Name&gt;.&lt;xmlId&gt;.cdn.com</code> .
signed	bool	<ul style="list-style-type: none"> <li>• false: token based auth (see <code>:ref:token-based-auth</code>) is not enabled for this deliveryservice.</li> <li>• true: token based auth is enabled for this delivery-service.</li> </ul>

Continued on next page

Table 11 – continued from previous page

Parameter	Type	Description
signingAlgorithm	string	<ul style="list-style-type: none"> <li>• null: token based auth (see :ref:token-based-auth) is not enabled for this deliveryservice.</li> <li>• “url_sig”: URL Sign token based auth is enabled for this deliveryservice.</li> <li>• “uri_signing”: URI Signing token based auth is enabled for this deliveryservice.</li> </ul>
sslKeyVersion	int	
tenant	string	Owning tenant name
tenantId	int	Owning tenant ID
trRequestHeaders	string	
trResponseHeaders	string	
typeId	int	The type of this deliveryservice (one of :ref:to-api-v11-types use_in_table='deliveryservice').
xmlId	string	Unique string that describes this deliveryservice.

**Response Example**

```
{
  "response": [
    {
      "active": true,
      "cacheurl": null,
      "ccrDnsTtl": "3600",
      "cdnId": "2",
      "cdnName": "over-the-top",
      "checkPath": "",
      "deepCachingType": "NEVER",
      "displayName": "My Cool Delivery Service",
      "dnsBypassCname": "",
      "dnsBypassIp": "",
      "dnsBypassIp6": "",
      "dnsBypassTtl": "30",
      "dscp": "40",
      "edgeHeaderRewrite": null,
      "exampleURLs": [
        "http://foo.foo-ds.foo.bar.net"
      ],
      "geoLimit": "0",
      "geoLimitCountries": null,
      "geoLimitRedirectURL": null,
      "geoProvider": "0",
      "globalMaxMbps": null,
      "globalMaxTps": "0",
      "httpBypassFqdn": ""
    }
  ]
}
```

(continues on next page)

(continued from previous page)

```

    "id": "442",
    "infoUrl": "",
    "initialDispersion": "1",
    "ipv6RoutingEnabled": true,
    "lastUpdated": "2016-01-26 08:49:35",
    "logsEnabled": false,
    "longDesc": "",
    "longDesc1": "",
    "longDesc2": "",
    "maxDnsAnswers": "0",
    "midHeaderRewrite": null,
    "missLat": "39.7391500",
    "missLong": "-104.9847000",
    "multiSiteOrigin": false,
    "orgServerFqdn": "http://baz.boo.net",
    "originShield": null,
    "profileDescription": "Content Router for over-the-top",
    "profileId": "5",
    "profileName": "ROUTER_TOP",
    "protocol": "0",
    "qstringIgnore": "1",
    "rangeRequestHandling": "0",
    "regexRemap": null,
    "regionalGeoBlocking": false,
    "remapText": null,
    "routingName": "foo",
    "signed": false,
    "signingAlgorithm": null,
    "sslKeyVersion": "0",
    "tenant": "root",
    "tenantId": 1,
    "trRequestHeaders": null,
    "trResponseHeaders": "Access-Control-Allow-Origin: *",
    "type": "HTTP",
    "typeId": "8",
    "xmlId": "foo-ds"
  }
  { .. },
  { .. }
]
}

```

**GET /api/1.2/deliveryservices/:id**

Retrieves a specific delivery service. If not admin / ops, delivery service must be assigned to user. See also [Using Traffic Ops - Delivery Service](#).

Authentication Required: Yes

Role(s) Required: None

**Request Route Parameters**

Name	Required	Description
id	yes	Delivery service ID.

### Response Properties

Parameter	Type	Description
active	bool	true if active, false if inactive.
cacheurl	string	Cache URL rule to apply to this delivery service.
ccrDnsTtl	int	The TTL of the DNS response for A or AAAA queries requesting the IP address of the tr. host.
cdnId	int	Id of the CDN to which the delivery service belongs to.
cdnName	string	Name of the CDN to which the delivery service belongs to.
checkPath	string	The path portion of the URL to check this deliveryservice for health.
deepCachingType	string	When to do Deep Caching for this Delivery Service: <ul style="list-style-type: none"><li>• NEVER (default)</li><li>• ALWAYS</li></ul>
displayName	string	The display name of the delivery service.
dnsBypassCname	string	
dnsBypassIp	string	The IPv4 IP to use for bypass on a DNS deliveryservice - bypass starts when serving more than the globalMaxMbps traffic on this deliveryservice.
dnsBypassIp6	string	The IPv6 IP to use for bypass on a DNS deliveryservice - bypass starts when serving more than the globalMaxMbps traffic on this deliveryservice.
dnsBypassTtl	int	The TTL of the DNS bypass response.
dscp	int	The Differentiated Services Code Point (DSCP) with which to mark downstream (EDGE -> customer) traffic.
edgeHeaderRewrite	string	The EDGE header rewrite actions to perform.
exampleURLs	array	Entry points into the CDN for this deliveryservice.
geoLimitRedirectUrl	string	

Continued on next page



Table 12 – continued from previous page

Parameter	Type	Description
geoLimit	int	<ul style="list-style-type: none"> <li>• 0: None - no limitations</li> <li>• 1: Only route on CZF file hit</li> <li>• 2: Only route on CZF hit or when from USA</li> </ul> <p>Note that this does not prevent access to content or makes content secure; it just prevents routing to the content by Traffic Router.</p>
geoLimitCountries	string	
geoProvider	int	
globalMaxMbps	int	The maximum global bandwidth allowed on this deliveryservice. If exceeded, the traffic routes to the dnsByPassIp* for DNS deliveryservices and to the httpBypassFqdn for HTTP deliveryservices.
globalMaxTps	int	The maximum global transactions per second allowed on this deliveryservice. When this is exceeded traffic will be sent to the dnsByPassIp* for DNS deliveryservices and to the httpBypassFqdn for HTTP deliveryservices
httpBypassFqdn	string	The HTTP destination to use for bypass on an HTTP deliveryservice - bypass starts when serving more than the globalMaxMbps traffic on this deliveryservice.
id	int	The deliveryservice id (database row number).
infoUrl	string	Use this to add a URL that points to more information about that deliveryservice.
initialDispersion	int	
ipv6RoutingEnabled	bool	false: send IPv4 address of Traffic Router to client on HTTP type del.
lastUpdated	string	
logsEnabled	bool	
longDesc	string	Description field.
longDesc1	string	Description field 1.
longDesc2	string	Description field 2.
matchList	array	Array of matchList hashes.

Continued on next page

Table 12 – continued from previous page

Parameter	Type	Description
>>type	string	The type of MatchList (one of :ref:to-api-v11-types use_in_table='regex').
>>setNumber	string	The set Number of the match-List.
>>pattern	string	The regexp for the matchList.
maxDnsAnswers	int	The maximum number of IPs to put in a A/AAAA response for a DNS deliveryservice (0 means all available).
midHeaderRewrite	string	The MID header rewrite actions to perform.
missLat	float	The latitude as decimal degrees to use when the client cannot be found in the CZF or the Geo lookup. <ul style="list-style-type: none"> <li>e.g. 39.7391500 or null</li> </ul>
missLong	float	The longitude as decimal degrees to use when the client cannot be found in the CZF or the Geo lookup. <ul style="list-style-type: none"> <li>e.g. -104.9847000 or null</li> </ul>
multiSiteOrigin	bool	Is the Multi Site Origin feature enabled for this delivery service (0=false, 1=true). See <i>Multi Site Origin</i>
orgServerFqdn	string	The origin server base URL (FQDN when used in this instance, includes the protocol ( <a href="http://">http://</a> or <a href="https://">https://</a> ) for use in retrieving content from the origin server.
originShield	string	
profileDescription	string	The description of the Traffic Router Profile with which this deliveryservice is associated.
profileId	int	The id of the Traffic Router Profile with which this deliveryservice is associated.
profileName	string	The name of the Traffic Router Profile with which this deliveryservice is associated.

Continued on next page

Table 12 – continued from previous page

Parameter	Type	Description
protocol	int	<ul style="list-style-type: none"> <li>• 0: serve with <a href="#">http://</a> at EDGE</li> <li>• 1: serve with <a href="#">https://</a> at EDGE</li> <li>• 2: serve with both <a href="#">http://</a> and <a href="#">https://</a> at EDGE</li> </ul>
qstringIgnore	int	<ul style="list-style-type: none"> <li>• 0: no special query string handling; it is for use in the cache-key and pass up to origin.</li> <li>• 1: ignore query string in cache-key, but pass it up to parent and or origin.</li> <li>• 2: drop query string at edge, and do not use it in the cache-key.</li> </ul>
rangeRequestHandling	int	<p>How to treat range requests:</p> <ul style="list-style-type: none"> <li>• 0 Do not cache (ranges requested from files taht are already cached due to a non range request will be a HIT)</li> <li>• 1 Use the <a href="#">background_fetch</a> plugin.</li> <li>• 2 Use the <a href="#">cache_range_requests</a> plugin.</li> </ul>
regexRemap	string	Regex Remap rule to apply to this delivery service at the Edge tier.
regionalGeoBlocking	bool	Regex Remap rule to apply to this delivery service at the Edge tier.
remapText	string	Additional raw remap line text.
routingName	string	The routing name of this deliveryservice, e.g. <routing-Name>.<xmlId>.cdn.com.
signed	bool	<ul style="list-style-type: none"> <li>• false: token based auth (see <a href="#">:ref:token-based-auth</a>) is not enabled for this deliveryservice.</li> <li>• true: token based auth is enabled for this delivery-service.</li> </ul>

Continued on next page

Table 12 – continued from previous page

Parameter	Type	Description
signingAlgorithm	string	<ul style="list-style-type: none"> <li>• null: token based auth (see :ref:token-based-auth) is not enabled for this deliveryservice.</li> <li>• “url_sig”: URL Sign token based auth is enabled for this deliveryservice.</li> <li>• “uri_signing”: URI Signing token based auth is enabled for this deliveryservice.</li> </ul>
sslKeyVersion	int	
tenant	string	Owning tenant name
tenantId	int	Owning tenant ID
trRequestHeaders	string	
trResponseHeaders	string	
typeId	int	The type of this deliveryservice (one of :ref:to-api-v11-types use_in_table='deliveryservice').
xmlId	string	Unique string that describes this deliveryservice.

**Response Example**

```
{
  "response": [
    {
      "active": true,
      "cacheurl": null,
      "ccrDnsTtl": "3600",
      "cdnId": "2",
      "cdnName": "over-the-top",
      "checkPath": "",
      "deepCachingType": "NEVER",
      "displayName": "My Cool Delivery Service",
      "dnsBypassCname": "",
      "dnsBypassIp": "",
      "dnsBypassIp6": "",
      "dnsBypassTtl": "30",
      "dscp": "40",
      "edgeHeaderRewrite": null,
      "exampleURLs": [
        "http://foo.foo-ds.foo.bar.net"
      ],
      "geoLimit": "0",
      "geoLimitCountries": null,
      "geoLimitRedirectURL": null,
      "geoProvider": "0",
      "globalMaxMbps": null,
      "globalMaxTps": "0",
      "httpBypassFqdn": ""
    }
  ]
}
```

(continues on next page)

```

    "id": "442",
    "infoUrl": "",
    "initialDispersion": "1",
    "ipv6RoutingEnabled": true,
    "lastUpdated": "2016-01-26 08:49:35",
    "logsEnabled": false,
    "longDesc": "",
    "longDesc1": "",
    "longDesc2": "",
    "matchList": [
      {
        "pattern": ".*\\.foo-ds\\.\\.\\.\"",
        "setNumber": "0",
        "type": "HOST_REGEX"
      }
    ],
    "maxDnsAnswers": "0",
    "midHeaderRewrite": null,
    "missLat": "39.7391500",
    "missLong": "-104.9847000",
    "multiSiteOrigin": false,
    "orgServerFqdn": "http://baz.boo.net",
    "originShield": null,
    "profileDescription": "Content Router for over-the-top",
    "profileId": "5",
    "profileName": "ROUTER_TOP",
    "protocol": "0",
    "qstringIgnore": "1",
    "rangeRequestHandling": "0",
    "regexRemap": null,
    "regionalGeoBlocking": false,
    "remapText": null,
    "routingName": "foo",
    "signed": false,
    "signingAlgorithm": null,
    "sslKeyVersion": "0",
    "tenant": "root",
    "tenantId": 1,
    "trRequestHeaders": null,
    "trResponseHeaders": "Access-Control-Allow-Origin: *",
    "type": "HTTP",
    "typeId": "8",
    "xmlId": "foo-ds"
  }
}

```

Role(s) Required: Admin or Operations or delivery service must be assigned to user.

### Request Route Parameters

Name	Required	Description
id	yes	Delivery service ID.

### Response Properties

Parameter	Type	Description
cachegroup	string	The cache group name (see <a href="#">Cache Group</a> ).
cachegroupId	string	The cache group id.
cdnId	string	Id of the CDN to which the server belongs to.
cdnName	string	Name of the CDN to which the server belongs to.
domainName	string	The domain name part of the FQDN of the cache.
guid	string	An identifier used to uniquely identify the server.
hostName	string	The host name part of the cache.
httpsPort	string	The HTTPS port on which the main application listens (443 in most cases).
id	string	The server id (database row number).
iloIpAddress	string	The IPv4 address of the lights-out-management port.
iloIpGateway	string	The IPv4 gateway address of the lights-out-management port.
iloIpNetmask	string	The IPv4 netmask of the lights-out-management port.
iloPassword	string	The password of the of the lights-out-management user (displays as ** unless you are an ‘admin’ user).
iloUsername	string	The user name for lights-out-management.
interfaceMtu	string	The Maximum Transmission Unit (MTU) to configure for interfaceName.
interfaceName	string	The network interface name used for serving traffic.
ip6Address	string	The IPv6 address/netmask for interfaceName.
ip6Gateway	string	The IPv6 gateway for interfaceName.
ipAddress	string	The IPv4 address for interfaceName.
ipGateway	string	The IPv4 gateway for interfaceName.
ipNetmask	string	The IPv4 netmask for interfaceName.
lastUpdated	string	The Time and Date for the last update for this server.
mgmtIpAddress	string	The IPv4 address of the management port (optional).
mgmtIpGateway	string	The IPv4 gateway of the management port (optional).
mgmtIpNetmask	string	The IPv4 netmask of the management port (optional).
offlineReason	string	A user-entered reason why the server is in ADMIN_DOWN or OFFLINE status.
physLocation	string	The physical location name (see <a href="#">Physical Location</a> ).
physLocationId	string	The physical location id (see <a href="#">Physical Location</a> ).
profile	string	The assigned profile name (see <a href="#">Profiles</a> ).
profileDesc	string	The assigned profile description (see <a href="#">Profiles</a> ).
profileId	string	The assigned profile Id (see <a href="#">Profiles</a> ).
rack	string	A string indicating rack location.
routerHostName	string	The human readable name of the router.
routerPortName	string	The human readable name of the router port.
status	string	The Status string (See <a href="#">Status</a> ).
statusId	string	The Status id (See <a href="#">Status</a> ).
tcpPort	string	The default TCP port on which the main application listens (80 for a cache in most cases).
type	string	The name of the type of this server (see <a href="#">Types</a> ).
typeId	string	The id of the type of this server (see <a href="#">Types</a> ).
updPending	bool	

### Response Example

```

{
  "response": [
    {
      "cachegroup": "us-il-chicago",
      "cachegroupId": "3",
      "cdnId": "3",
      "cdnName": "CDN-1",
      "domainName": "chi.kabletown.net",
      "guid": null,
      "hostName": "atsec-chi-00",
      "id": "19",
      "iloIpAddress": "172.16.2.6",
      "iloIpGateway": "172.16.2.1",
      "iloIpNetmask": "255.255.255.0",
      "iloPassword": "*****",
      "iloUsername": "",
      "interfaceMtu": "9000",
      "interfaceName": "bond0",
      "ip6Address": "2033:D0D0:3300::2:2/64",
      "ip6Gateway": "2033:D0D0:3300::2:1",
      "ipAddress": "10.10.2.2",
      "ipGateway": "10.10.2.1",
      "ipNetmask": "255.255.255.0",
      "lastUpdated": "2015-03-08 15:57:32",
      "mgmtIpAddress": "",
      "mgmtIpGateway": "",
      "mgmtIpNetmask": "",
      "offlineReason": "N/A",
      "physLocation": "plocation-chi-1",
      "physLocationId": "9",
      "profile": "EDGE1_CDN1_421_SSL",
      "profileDesc": "EDGE1_CDN1_421_SSL profile",
      "profileId": "12",
      "rack": "RR 119.02",
      "routerHostName": "rtr-chi.kabletown.net",
      "routerPortName": "2",
      "status": "ONLINE",
      "statusId": "6",
      "tcpPort": "80",
      "httpsPort": "443",
      "type": "EDGE",
      "typeId": "3",
      "updPending": false
    },
    {
      ... more server data
    }
  ]
}

```

**GET /api/1.2/deliveryservices/:id/servers/unassigned**

Retrieves properties of CDN EDGE or ORG servers not assigned to a delivery service.

Authentication Required: Yes

Role(s) Required: Admin or Operations or delivery service must be assigned to user

### Request Route Parameters

Name	Required	Description
id	yes	Delivery service ID.

### Response Properties

Parameter	Type	Description
cachegroup	string	The cache group name (see <a href="#">Cache Group</a> ).
cachegroupId	string	The cache group id.
cdnId	string	Id of the CDN to which the server belongs to.
cdnName	string	Name of the CDN to which the server belongs to.
domainName	string	The domain name part of the FQDN of the cache.
guid	string	An identifier used to uniquely identify the server.
hostName	string	The host name part of the cache.
httpsPort	string	The HTTPS port on which the main application listens (443 in most cases).
id	string	The server id (database row number).
iloIpAddress	string	The IPv4 address of the lights-out-management port.
iloIpGateway	string	The IPv4 gateway address of the lights-out-management port.
iloIpNetmask	string	The IPv4 netmask of the lights-out-management port.
iloPassword	string	The password of the of the lights-out-management user (displays as ** unless you are an ‘admin’ user).
iloUsername	string	The user name for lights-out-management.
interfaceMtu	string	The Maximum Transmission Unit (MTU) to configure for interfaceName.
interfaceName	string	The network interface name used for serving traffic.
ip6Address	string	The IPv6 address/netmask for interfaceName.
ip6Gateway	string	The IPv6 gateway for interfaceName.
ipAddress	string	The IPv4 address for interfaceName.
ipGateway	string	The IPv4 gateway for interfaceName.
ipNetmask	string	The IPv4 netmask for interfaceName.
lastUpdated	string	The Time and Date for the last update for this server.
mgmtIpAddress	string	The IPv4 address of the management port (optional).
mgmtIpGateway	string	The IPv4 gateway of the management port (optional).
mgmtIpNetmask	string	The IPv4 netmask of the management port (optional).
offlineReason	string	A user-entered reason why the server is in ADMIN_DOWN or OFFLINE status.
physLocation	string	The physical location name (see <a href="#">Physical Location</a> ).
physLocationId	string	The physical location id (see <a href="#">Physical Location</a> ).
profile	string	The assigned profile name (see <a href="#">Profiles</a> ).
profileDesc	string	The assigned profile description (see <a href="#">Profiles</a> ).
profileId	string	The assigned profile Id (see <a href="#">Profiles</a> ).
rack	string	A string indicating rack location.
routerHostName	string	The human readable name of the router.
routerPortName	string	The human readable name of the router port.
status	string	The Status string (See <a href="#">Status</a> ).
statusId	string	The Status id (See <a href="#">Status</a> ).
tcpPort	string	The default TCP port on which the main application listens (80 for a cache in most cases).
type	string	The name of the type of this server (see <a href="#">Types</a> ).
typeId	string	The id of the type of this server (see <a href="#">Types</a> ).
updPending	bool	



**Response Example**

```
{
  "response": [
    {
      "cachegroup": "us-il-chicago",
      "cachegroupId": "3",
      "cdnId": "3",
      "cdnName": "CDN-1",
      "domainName": "chi.kabletown.net",
      "guid": null,
      "hostName": "atsec-chi-00",
      "id": "19",
      "iloIpAddress": "172.16.2.6",
      "iloIpGateway": "172.16.2.1",
      "iloIpNetmask": "255.255.255.0",
      "iloPassword": "*****",
      "iloUsername": "",
      "interfaceMtu": "9000",
      "interfaceName": "bond0",
      "ip6Address": "2033:D0D0:3300::2:2/64",
      "ip6Gateway": "2033:D0D0:3300::2:1",
      "ipAddress": "10.10.2.2",
      "ipGateway": "10.10.2.1",
      "ipNetmask": "255.255.255.0",
      "lastUpdated": "2015-03-08 15:57:32",
      "mgmtIpAddress": "",
      "mgmtIpGateway": "",
      "mgmtIpNetmask": "",
      "offlineReason": "N/A",
      "physLocation": "plocation-chi-1",
      "physLocationId": "9",
      "profile": "EDGE1_CDN1_421_SSL",
      "profileDesc": "EDGE1_CDN1_421_SSL profile",
      "profileId": "12",
      "rack": "RR 119.02",
      "routerHostName": "rtr-chi.kabletown.net",
      "routerPortName": "2",
      "status": "ONLINE",
      "statusId": "6",
      "tcpPort": "80",
      "httpsPort": "443",
      "type": "EDGE",
      "typeId": "3",
      "updPending": false
    },
    {
      ... more server data
    }
  ]
}
```

**GET /api/1.2/deliveryservices/:id/servers/eligible**

Retrieves properties of CDN EDGE or ORG servers not eligible for assignment to a delivery service.

Authentication Required: Yes

Role(s) Required: Admin or Operations or delivery service must be assigned to user

### Request Route Parameters

Name	Required	Description
id	yes	Delivery service ID.

### Response Properties

Parameter	Type	Description
cachegroup	string	The cache group name (see <a href="#">Cache Group</a> ).
cachegroupId	string	The cache group id.
cdnId	string	Id of the CDN to which the server belongs to.
cdnName	string	Name of the CDN to which the server belongs to.
domainName	string	The domain name part of the FQDN of the cache.
guid	string	An identifier used to uniquely identify the server.
hostName	string	The host name part of the cache.
httpsPort	string	The HTTPS port on which the main application listens (443 in most cases).
id	string	The server id (database row number).
iloIpAddress	string	The IPv4 address of the lights-out-management port.
iloIpGateway	string	The IPv4 gateway address of the lights-out-management port.
iloIpNetmask	string	The IPv4 netmask of the lights-out-management port.
iloPassword	string	The password of the of the lights-out-management user (displays as ** unless you are an 'admin' user).
iloUsername	string	The user name for lights-out-management.
interfaceMtu	string	The Maximum Transmission Unit (MTU) to configure for interfaceName.
interfaceName	string	The network interface name used for serving traffic.
ip6Address	string	The IPv6 address/netmask for interfaceName.
ip6Gateway	string	The IPv6 gateway for interfaceName.
ipAddress	string	The IPv4 address for interfaceName.
ipGateway	string	The IPv4 gateway for interfaceName.
ipNetmask	string	The IPv4 netmask for interfaceName.
lastUpdated	string	The Time and Date for the last update for this server.
mgmtIpAddress	string	The IPv4 address of the management port (optional).
mgmtIpGateway	string	The IPv4 gateway of the management port (optional).
mgmtIpNetmask	string	The IPv4 netmask of the management port (optional).
offlineReason	string	A user-entered reason why the server is in ADMIN_DOWN or OFFLINE status.
physLocation	string	The physical location name (see <a href="#">Physical Location</a> ).
physLocationId	string	The physical location id (see <a href="#">Physical Location</a> ).
profile	string	The assigned profile name (see <a href="#">Profiles</a> ).
profileDesc	string	The assigned profile description (see <a href="#">Profiles</a> ).
profileId	string	The assigned profile Id (see <a href="#">Profiles</a> ).
rack	string	A string indicating rack location.
routerHostName	string	The human readable name of the router.
routerPortName	string	The human readable name of the router port.
status	string	The Status string (See <a href="#">Status</a> ).
statusId	string	The Status id (See <a href="#">Status</a> ).
tcpPort	string	The default TCP port on which the main application listens (80 for a cache in most cases).
type	string	The name of the type of this server (see <a href="#">Types</a> ).

Continued on next page

Table 15 – continued from previous page

Parameter	Type	Description
typeId	string	The id of the type of this server (see <i>Types</i> ).
updPending	bool	

**Response Example**

```
{
  "response": [
    {
      "cachegroup": "us-il-chicago",
      "cachegroupId": "3",
      "cdnId": "3",
      "cdnName": "CDN-1",
      "domainName": "chi.kabletown.net",
      "guid": null,
      "hostName": "atsec-chi-00",
      "id": "19",
      "iloIpAddress": "172.16.2.6",
      "iloIpGateway": "172.16.2.1",
      "iloIpNetmask": "255.255.255.0",
      "iloPassword": "*****",
      "iloUsername": "",
      "interfaceMtu": "9000",
      "interfaceName": "bond0",
      "ip6Address": "2033:D0D0:3300::2:2/64",
      "ip6Gateway": "2033:D0D0:3300::2:1",
      "ipAddress": "10.10.2.2",
      "ipGateway": "10.10.2.1",
      "ipNetmask": "255.255.255.0",
      "lastUpdated": "2015-03-08 15:57:32",
      "mgmtIpAddress": "",
      "mgmtIpGateway": "",
      "mgmtIpNetmask": "",
      "offlineReason": "N/A",
      "physLocation": "plocation-chi-1",
      "physLocationId": "9",
      "profile": "EDGE1_CDN1_421_SSL",
      "profileDesc": "EDGE1_CDN1_421_SSL profile",
      "profileId": "12",
      "rack": "RR 119.02",
      "routerHostName": "rtr-chi.kabletown.net",
      "routerPortName": "2",
      "status": "ONLINE",
      "statusId": "6",
      "tcpPort": "80",
      "httpsPort": "443",
      "type": "EDGE",
      "typeId": "3",
      "updPending": false
    },
    {
      ... more server data
    }
  ]
}
```

## Health

### GET /api/1.2/deliveryservices/:id/state

Retrieves the failover state for a delivery service. Delivery service must be assigned to user if user is not admin or operations.

Authentication Required: Yes

Role(s) Required: None

#### Response Properties

Parameter	Type	Description
failover	hash	
>locations	array	
>destination	hash	
>>location	string	
>>type	string	
>configured	boolean	
>enabled	boolean	
enabled	boolean	

#### Response Example

```
{
  "response": {
    "failover": {
      "locations": [ ],
      "destination": {
        "location": null,
        "type": "DNS",
      },
      "configured": false,
      "enabled": false
    },
    "enabled": true
  }
}
```

### GET /api/1.2/deliveryservices/:id/health

Retrieves the health of all locations (cache groups) for a delivery service. Delivery service must be assigned to user if user is not admin or operations.

Authentication Required: Yes

Role(s) Required: None

#### Response Properties

Parameter	Type	Description
totalOnline	int	Total number of online caches across all CDNs.
totalOffline	int	Total number of offline caches across all CDNs.
cachegroups	array	A collection of cache groups.
>online	int	The number of online caches for the cache group
>offline	int	The number of offline caches for the cache group.
>name	string	Cache group name.

### Response Example

```
{
  "response": {
    "totalOnline": 148,
    "totalOffline": 0,
    "cachegroups": [
      {
        "online": 8,
        "offline": 0,
        "name": "us-co-denver"
      },
      {
        "online": 7,
        "offline": 0,
        "name": "us-de-newcastle"
      }
    ]
  }
}
```

### GET /api/1.2/deliveryservices/:id/capacity

Retrieves the capacity percentages of a delivery service. Delivery service must be assigned to user if user is not admin or operations.

Authentication Required: Yes

Role(s) Required: None

#### Request Route Parameters

Name	Required	Description
id	yes	delivery service id.

### Response Properties

Parameter	Type	Description
availablePercent	number	The percentage of server capacity assigned to the delivery service that is available.
unavailablePercent	number	The percentage of server capacity assigned to the delivery service that is unavailable.
utilizedPercent	number	The percentage of server capacity assigned to the delivery service being used.
maintenancePercent	number	The percentage of server capacity assigned to the delivery service that is down for maintenance.

**Response Example**

```
{
  "response": {
    "availablePercent": 89.0939840205533,
    "unavailablePercent": 0,
    "utilizedPercent": 10.9060020300395,
    "maintenancePercent": 0.0000139494071146245
  },
}
```

**GET /api/1.2/deliveryservices/:id/routing**

Retrieves the routing method percentages of a delivery service. Delivery service must be assigned to user if user is not admin or operations.

Authentication Required: Yes

Role(s) Required: None

**Request Route Parameters**

Name	Required	Description
id	yes	delivery service id.

**Response Properties**

Parameter	Type	Description
staticRoute	number	The percentage of Traffic Router responses for this deliveryservice satisfied with pre-configured DNS entries.
miss	number	The percentage of Traffic Router responses for this deliveryservice that were a miss (no location available for client IP).
geo	number	The percentage of Traffic Router responses for this deliveryservice satisfied using 3rd party geo-IP mapping.
err	number	The percentage of Traffic Router requests for this deliveryservice resulting in an error.
cz	number	The percentage of Traffic Router requests for this deliveryservice satisfied by a CZF (coverage zone file) hit.
dsr	number	The percentage of Traffic Router requests for this deliveryservice satisfied by sending the client to the overflow CDN.
fed	number	The percentage of Traffic Router requests for this deliveryservice satisfied by sending the client to a federated CDN.
regionalAlternate	number	The percentage of Traffic Router requests for this deliveryservice satisfied by sending the client to the alternate regional geoblocking URL.
regionalDenied	number	The percent of Traffic Router requests for this deliveryservice denied due to geolocation policy.

### Response Example

```
{
  "response": {
    "staticRoute": 0,
    "miss": 0,
    "geo": 37.8855391018869,
    "err": 0,
    "cz": 62.1144608981131,
    "dsr": 0,
    "fed": 0,
    "regionalAlternate": 0,
    "regionalDenied": 0
  },
}
```

## Delivery Service Server

### GET /api/1.2/deliveryserviceserver

Retrieves delivery service / server assignments.

Authentication Required: Yes

Role(s) Required: None

#### Request Query Parameters

Name	Required	Description
page	no	The page number for use in pagination.
limit	no	For use in limiting the result set.

#### Response Properties

Parameter	Type	Description
lastUpdated	array	
server	string	
deliveryService	string	

### Response Example

```
{
  "page": 2,
  "orderby": "deliveryservice",
  "response": [
    {
      "lastUpdated": "2014-09-26 17:53:43",
      "server": "20",
      "deliveryService": "1"
    },
    {
      "lastUpdated": "2014-09-26 17:53:44",
      "server": "21",
      "deliveryService": "1"
    }
  ],
  "limit": 2
}
```

### POST /api/1.2/deliveryserviceserver

Create one or more delivery service / server assignments.

Authentication Required: Yes

Role(s) Required: Admin or Operations or the delivery service is assigned to the user.

### Request Parameters

Name	Required	Description
dsId	yes	The ID of the delivery service.
servers	yes	An array of server IDs.
replace	no	Replace existing ds/server assignments? (true/false)

### Request Example

```
{
  "dsId": 246,
  "servers": [ 2, 3, 4, 5, 6 ],
  "replace": true
}
```

### Response Properties

Parameter	Type	Description
dsId	int	The ID of the delivery service.
servers	array	An array of server IDs.
replace	array	Existing ds/server assignments replaced? (true/false).

### Response Example



```
{
  "alerts": [
    {
      "level": "success",
      "text": "Server assignments complete."
    }
  ],
  "response": {
    "dsId" : 246,
    "servers" : [ 2, 3, 4, 5, 6 ],
    "replace" : true
  }
}
```

### DELETE /api/1.2/deliveryservice\_server/:dsId/:serverId

Removes a server (cache) from a delivery service.

Authentication Required: Yes

Role(s) Required: Admin or Oper (if delivery service is not assigned to user)

#### Request Route Parameters

Name	Required	Description
dsId	yes	Delivery service ID.
serverId	yes	Server (cache) ID.

#### Response Example

```
{
  "alerts": [
    {
      "level": "success",
      "text": "Server unlinked from delivery_
↪service."
    }
  ],
}
```

## Delivery Service User

### POST /api/1.2/deliveryservice\_user

Create one or more user / delivery service assignments.

Authentication Required: Yes

Role(s) Required: Admin or Operations

### Request Parameters

Name	Required	Description
userId	yes	The ID of the user.
deliveryServices	yes	An array of delivery service IDs.
replace	no	Replace existing user/ds assignments? (true/false).

### Request Example

```
{
  "userId": 50,
  "deliveryServices": [ 23, 34, 45, 56, 67 ],
  "replace": true
}
```

### Response Properties

Parameter	Type	Description
userId	int	The ID of the user.
deliveryServices	array	An array of delivery service IDs.
replace	array	Existing user/ds assignments replaced? (true/false).

### Response Example

```
{
  "alerts": [
    {
      "level": "success",
      "text": "Delivery service assignments complete."
    }
  ],
  "response": {
    "userId": 50,
    "deliveryServices": [ 23, 34, 45, 56, 67 ],
    "replace": true
  }
}
```

## DELETE /api/1.2/deliveryservice\_user/:dsId/:userId

Removes a delivery service from a user.

Authentication Required: Yes

Role(s) Required: Admin or Operations

### Request Route Parameters

Name	Required	Description
dsId	yes	Delivery service ID.
userId	yes	User ID.

**Response Example**

```
{
  "alerts": [
    {
      "level": "success",
      "text": "User and delivery service were↵
↵unlinked."
    }
  ],
}
```

**SSL Keys****GET /api/1.2/deliveryservices/xmlId/:xmlid/sslkeys**

Retrieves ssl keys for a delivery service.

Authentication Required: Yes

Role(s) Required: Admin

**Request Route Parameters**

Name	Required	Description
xmlId	yes	xml_id of the desired delivery service

**Request Query Parameters**

Name	Required	Description
version	no	The version number to retrieve
decode	no	a boolean value to decode the certs or not

**Response Properties**

Parameter	Type	Description
crt	string	base64 encoded (or not if decode=true) crt file for delivery service
csr	string	base64 encoded (or not if decode=true) csr file for delivery service
key	string	base64 encoded (or not if decode=true) private key file for delivery service
cdn	string	The CDN of the delivery service for which the certs were generated.
deliveryservice	string	The XML ID of the delivery service for which the cert was generated.
businessUnit	string	The business unit entered by the user when generating certs. Field is optional and if not provided by the user will not be in response
city	string	The city entered by the user when generating certs. Field is optional and if not provided by the user will not be in response
organization	string	The organization entered by the user when generating certs. Field is optional and if not provided by the user will not be in response
hostname	string	The hostname generated by Traffic Ops that is used as the common name when generating the certificate. This will be a FQDN for DNS delivery services and a wildcard URL for HTTP delivery services.
country	string	The country entered by the user when generating certs. Field is optional and if not provided by the user will not be in response
state	string	The state entered by the user when generating certs. Field is optional and if not provided by the user will not be in response
version	string	The version of the certificate record in Riak

### Response Example

```
{
  "response": {
    "certificate": {
      "crt": "crt",
      "key": "key",
      "csr": "csr"
    },
    "deliveryservice": "my-ds",
    "cdn": "qa",
    "businessUnit": "CDN_Eng",
    "city": "Denver",
    "organization": "KableTown",
    "hostname": "foober.com",
    "country": "US",
    "state": "Colorado",
    "version": "1"
  }
}
```

### GET /api/1.2/deliveryservices/hostname/:hostname/sslkeys

Authentication Required: Yes

Role(s) Required: Admin

#### Request Route Parameters

Name	Required	Description
hostname	yes	pristine hostname of the desired delivery service

### Request Query Parameters

Name	Required	Description
version	no	The version number to retrieve
decode	no	a boolean value to decode the certs or not

### Response Properties

Parameter	Type	Description
crt	string	base64 encoded (or not if decode=true) crt file for delivery service
csr	string	base64 encoded (or not if decode=true) csr file for delivery service
key	string	base64 encoded (or not if decode=true) private key file for delivery service
cdn	string	The CDN of the delivery service for which the certs were generated.
deliveryservice	string	The XML ID of the delivery service for which the cert was generated.
businessUnit	string	The business unit entered by the user when generating certs. Field is optional and if not provided by the user will not be in response
city	string	The city entered by the user when generating certs. Field is optional and if not provided by the user will not be in response
organization	string	The organization entered by the user when generating certs. Field is optional and if not provided by the user will not be in response
hostname	string	The hostname generated by Traffic Ops that is used as the common name when generating the certificate. This will be a FQDN for DNS delivery services and a wildcard URL for HTTP delivery services.
country	string	The country entered by the user when generating certs. Field is optional and if not provided by the user will not be in response
state	string	The state entered by the user when generating certs. Field is optional and if not provided by the user will not be in response
version	string	The version of the certificate record in Riak

### Response Example

```
{
  "response": {
    "certificate": {
      "crt": "crt",
      "key": "key",
      "csr": "csr"
    },
    "deliveryservice": "my-ds",
    "cdn": "qa",
    "businessUnit": "CDN_Eng",
    "city": "Denver",
    "organization": "KableTown",
    "hostname": "foober.com",
    "country": "US",
    "state": "Colorado",
    "version": "1"
  }
}
```

**GET /api/1.2/deliveryservices/xmlId:xmlid/sslkeys/delete**

Authentication Required: Yes

Role Required: Admin

**Request Route Parameters**

Name	Required	Description
xmlId	yes	xml_id of the desired delivery service

**Request Query Parameters**

Name	Required	Description
version	no	The version number to retrieve

**Response Properties**

Parameter	Type	Description
response	string	success response

**Response Example**

```
{
  "response": "Successfully deleted ssl keys for <xml_id>"
}
```

**POST /api/1.2/deliveryservices/sslkeys/generate**

Generates SSL crt, csr, and private key for a delivery service

Authentication Required: Yes

Role(s) Required: Admin

**Request Properties**

Parameter	Type	Description
key	string	xml_id of the delivery service
version	string	version of the keys being generated
hostname	string	the <i>pristine hostname</i> of the delivery service
country	string	Country
state	string	State
city	string	City
org	string	Organization
unit	boolean	Business Unit
deliveryservice	string	The deliveryservice xml-id for which you want to generate certs
cdn	string	The name of the CDN for which the deliveryservice belongs

### Request Example

```
{
  "key": "ds-01",
  "businessUnit": "CDN Engineering",
  "version": "3",
  "hostname": "tr.ds-01.ott.kabletown.com",
  "certificate": {
    "key": "some_key",
    "csr": "some_csr",
    "crt": "some_crt"
  },
  "country": "US",
  "organization": "Kabletown",
  "city": "Denver",
  "state": "Colorado",
  "deliveryservice" : "ds-01",
  "cdn": "cdn1"
}
```

### Response Properties

Parameter	Type	Description
response	string	response string
version	string	API version

### Response Example

```
{
  "response": "Successfully created ssl keys for ds-01"
}
```

## POST /api/1.2/deliveryservices/sslkeys/add

Allows user to add SSL crt, csr, and private key for a delivery service.

Authentication Required: Yes

Role(s) Required: Admin

### Request Properties

Parameter	Type	Description
key	string	xml_id of the delivery service
version	string	version of the keys being generated
csr	string	
crt	string	
key	string	
deliveryservice	string	The deliveryservice xml-id for which you want to generate certs
cdn	string	The name of the CDN for which the deliveryservice belongs
hostname	string	the <i>pristine hostname</i> of the delivery service

### Request Example

```
{
  "key": "ds-01",
  "version": "1",
  "certificate": {
    "key": "some_key",
    "csr": "some_csr",
    "crt": "some_crt"
  }
}
```

### Response Properties

Parameter	Type	Description
response	string	response string
version	string	API version

### Response Example

```
{
  "response": "Successfully added ssl keys for ds-01"
}
```

## URL Sig Keys

### GET /api/1.2/deliveryservices/xmlId/:xmlid/urlkeys

Retrieves URL sig keys for a delivery service.

Authentication Required: Yes

Role(s) Required: None

#### Request Route Parameters

Name	Required	Description
xmlId	yes	xml_id of the desired delivery service

### Response Properties



Parameter	Type	Description
key0	string	base64 encoded key for delivery service
key2	string	base64 encoded key for delivery service
keyn..	string	base64 encoded key for delivery service – repeats to 15 (16 total) and is currently unsorted.

### Response Example

```
{
  "response": {
    "key9": "ZvVQNYpPVQWQV8tjQnU16osm4y7xK4zD",
    "key6": "JhGdpw5X9o8TqHfgezCm0bqb9SQPASWL",
    "key8": "ySXdp1T8IeDEE1OCMftzZb9EIw_20wwq",
    "key0": "D4AYzJ1AE2nYisA9MxMtY03TPDCHji9C",
    "key3": "W90YH1Gc_kY1Yw5_I0Lrkv9JOzSIneI",
    "key12": "ZbtMb3mrKqfS8hnx9_xWBIP_OPWlUpzc",
    "key2": "0qgEoDO7sUsugIQemZbwmMt0tNCwB1sf",
    "key4": "aFJ2Gb7atmxVB8uv7T9S6OaDml3ycpGf",
    "key1": "wnWNR1mCz104C7EFptcqHd0xUMQyNFhA",
    "key11": "k6HMz1BH1x6htKkypRFfWQhAndQqe50e",
    "key10": "zYONfdD7fGYKj4kLvIj4U0918csuZO0d",
    "key15": "3360cGaIip_layZMc_0hI2teJbazxTQh",
    "key5": "SIwv3GOhWN7EE9wSwPFj18qE4M07sFxN",
    "key13": "SqQKBR6LqEOzp8AewZUCVtBcW_8YFclg",
    "key14": "DtXsu8nsw04YhT0kNoKBhu2G3P9WRpQJ",
    "key7": "cmKoIIxXGAxUMdCsWvnGLOIMGmNiuT5I"
  }
}
```

### POST /api/1.2/deliveryservices/xmlId:xmlid/urlkeys/generate

Generates Url sig keys for a delivery service

Authentication Required: Yes

Role(s) Required: Admin OR assigned DS

#### Request Route Parameters

Name	Required	Description
xmlId	yes	xml_id of the desired delivery service

#### Response Properties

Parameter	Type	Description
response	string	response string
version	string	API version

**Response Example**

```
{
  "response": "Successfully generated and stored keys"
}
```

**POST /api/1.2/deliveryservices/xmlId:xmlid/urlkeys/copyFromXmlId/:copyFromXmlId**

Allows user to copy url sig keys from a specified delivery service to a delivery service.

Authentication Required: Yes

Role(s) Required: Admin or assigned DS

**Request Route Parameters**

Name	Required	Description
xmlId	yes	xml_id of the desired delivery service
copyFromXmlId	yes	xml_id of the delivery service to copy url sig keys from

**Response Properties**

Parameter	Type	Description
response	string	response string
version	string	API version

**Response Example**

```
{
  "response": "Successfully copied and stored keys"
}
```

**POST /api/1.2/deliveryservices/request**

Allows a user to send delivery service request details to a specified email address.

Authentication Required: Yes

Role(s) Required: None

**Request Properties**

Parameter	Type	Required	Description
emailTo	string	yes	The email to which the delivery service request will be sent.
details	hash	yes	Parameters for the delivery service request.

Continued on next page

Table 16 – continued from previous page

Parameter	Type	Required	Description
>customer	string	yes	Name of the customer to associated with the delivery service.
>deepCachingType	string	no	When to do Deep Caching for this Delivery Service: <ul style="list-style-type: none"> <li>• NEVER (default)</li> <li>• ALWAYS</li> </ul>
>deliveryProtocol	string	yes	Eg. http or http/https
>routingType	string	yes	Eg. DNS or HTTP Redirect
>routingName	string	no	The routing name for the delivery service, e.g. <routing-Name>.<xmlId>.cdn.com
>serviceDesc	string	yes	A description of the delivery service.
>peakBPSEstimate	string	yes	Used to manage cache efficiency and plan for capacity.
>peakTPSEstimate	string	yes	Used to manage cache efficiency and plan for capacity.
>maxLibrarySizeEstimate	string	yes	Used to manage cache efficiency and plan for capacity.
>originURL	string	yes	The URL path to the origin server.
>hasOriginDynamicRedirect	bool	yes	This is a feature which allows services to use multiple origin URLs for the same service.
>originTestFile	string	yes	A URL path to a test file available on the origin server.
>hasOriginACLWhitelist	bool	yes	Is access to your origin restricted using an access control list (ACL or whitelist) of Ips?
>originHeaders	string	no	Header values that must be passed to requests to your origin.
>otherOriginSecurity	string	no	Other origin security measures that need to be considered for access.
>queryStringHandling	string	yes	How to handle query strings that come with the request.

Continued on next page

Table 16 – continued from previous page

Parameter	Type	Required	Description
>rangeRequestHandle	string	yes	How to handle range requests.
>hasSignedURLs	bool	yes	Are Urls signed?
>hasNegativeCachingCustomization	bool	yes	Any customization required for negative caching?
>negativeCachingCustomizationNote	string	yes	Negative caching customization instructions.
>serviceAliases	array	no	Service aliases which will be used for this service.
>rateLimitingGBPS	int	no	Rate Limiting - Bandwidth (Gbps)
>rateLimitingTPS	int	no	Rate Limiting - Transactions/Second
>overflowService	string	no	An overflow point (URL or IP address) used if rate limits are met.
>headerRewriteEdge	string	no	Headers can be added or altered at each layer of the CDN.
>headerRewriteMid	string	no	Headers can be added or altered at each layer of the CDN.
>headerRewriteRed	string	no	Headers can be added or altered at each layer of the CDN.
>notes	string	no	Additional instructions to provide the delivery service provisioning team.

### Request Example

```
{
  "emailTo": "foo@bar.com",
  "details": {
    "customer": "XYZ Corporation",
    "contentType": "video-on-demand",
    "deepCachingType": "NEVER",
    "deliveryProtocol": "http",
    "routingType": "dns",
    "routingName": "foo",
    "serviceDesc": "service description goes here",
    "peakBPSEstimate": "less-than-5-Gbps",
    "peakTPSEstimate": "less-than-1000-TPS",
    "maxLibrarySizeEstimate": "less-than-200-GB",
    "originURL": "http://myorigin.com",
    "hasOriginDynamicRemap": false,
    "originTestFile": "http://myorigin.com/crossdomain.xml",
  }
}
```

(continues on next page)

(continued from previous page)

```
"hasOriginACLWhitelist": true,
"originHeaders": "",
"otherOriginSecurity": "",
"queryStringHandling": "ignore-in-cache-key-and-pass-up",
"rangeRequestHandling": "range-requests-not-used",
"hasSignedURLs": true,
"hasNegativeCachingCustomization": true,
"negativeCachingCustomizationNote": "negative caching instructions",
"serviceAliases": [
    "http://alias1.com",
    "http://alias2.com"
],
"rateLimitingGBPS": 50,
"rateLimitingTPS": 5000,
"overflowService": "http://overflowcdn.com",
"headerRewriteEdge": "",
"headerRewriteMid": "",
"headerRewriteRedirectRouter": "",
"notes": ""
}
}
```

## Response Properties

Parameter	Type	Description
alerts	array	A collection of alert messages.
>level	string	Success, info, warning or error.
>text	string	Alert message.
version	string	

## Response Example

```
{
  "alerts": [
    {
      "level": "success",
      "text": "Delivery Service request sent to foo@bar.com."
    }
  ]
}
```

## POST /api/1.2/deliveryservices

Allows user to create a delivery service.

Authentication Required: Yes

Role(s) Required: Admin or Operations

**Request Properties**

Parameter	Required	Description
active	yes	true if active, false if inactive.
cacheurl	no	Cache URL rule to apply to this delivery service.
ccrDnsTtl	no	The TTL of the DNS response for A or AAAA queries requesting the IP address of the tr.host.
cdnId	yes	cdn id
checkPath	no	The path portion of the URL to check this deliveryservice for health.
deepCachingType	no	When to do Deep Caching for this Delivery Service: <ul style="list-style-type: none"><li>• NEVER (default)</li><li>• ALWAYS</li></ul>
displayName	yes	Display name
dnsBypassCname	no	Bypass CNAME
dnsBypassIp	no	The IPv4 IP to use for bypass on a DNS deliveryservice - bypass starts when serving more than the globalMaxMbps traffic on this deliveryservice.
dnsBypassIp6	no	The IPv6 IP to use for bypass on a DNS deliveryservice - bypass starts when serving more than the globalMaxMbps traffic on this deliveryservice.
dnsBypassTtl	no	The TTL of the DNS bypass response.
dscp	yes	The Differentiated Services Code Point (DSCP) with which to mark downstream (EDGE -> customer) traffic.
edgeHeaderRewrite	no	The EDGE header rewrite actions to perform.
geoLimitRedirectURL	no	This is the URL Traffic Router will redirect to when Geo Limit Failure.

Continued on next page

Table 17 – continued from previous page

Parameter	Required	Description
geoLimit	yes	<ul style="list-style-type: none"> <li>• 0: None - no limitations</li> <li>• 1: Only route on CZF file hit</li> <li>• 2: Only route on CZF hit or when from geo limit countries</li> </ul> <p>Note that this does not prevent access to content or makes content secure; it just prevents routing to the content by Traffic Router.</p>
geoLimitCountries	no	The geo limit countries.
geoProvider	yes	<ul style="list-style-type: none"> <li>• 0: Maxmind(default)</li> <li>• 1: Neustar</li> </ul>
globalMaxMbps	no	The maximum global bandwidth allowed on this delivery-service. If exceeded, the traffic routes to the dnsByPassIp* for DNS deliveryservices and to the httpBypassFqdn for HTTP deliveryservices.
globalMaxTps	no	The maximum global transactions per second allowed on this deliveryservice. When this is exceeded traffic will be sent to the dnsByPassIp* for DNS deliveryservices and to the httpBypassFqdn for HTTP deliveryservices
httpBypassFqdn	no	The HTTP destination to use for bypass on an HTTP deliveryservice - bypass starts when serving more than the globalMaxMbps traffic on this deliveryservice.
infoUrl	no	Use this to add a URL that points to more information about that deliveryservice.
initialDispersion	yesno	Initial dispersion. Required for HTTP* delivery services.
ipv6RoutingEnabled	yesno	false: send IPv4 address of Traffic Router to client on HTTP type del. Required for DNS*, HTTP* and STEERING* delivery services.

Continued on next page

Table 17 – continued from previous page

Parameter	Required	Description
logsEnabled	yes	<ul style="list-style-type: none"> <li>• false: No</li> <li>• true: Yes</li> </ul>
longDesc	no	Description field.
longDesc1	no	Description field 1.
longDesc2	no	Description field 2.
maxDnsAnswers	no	The maximum number of IPs to put in a A/AAAA response for a DNS deliveryservice (0 means all available).
midHeaderRewrite	no	The MID header rewrite actions to perform.
missLat	yesno	The latitude as decimal degrees to use when the client cannot be found in the CZF or the Geo lookup. e.g. 39.7391500 or null. Required for DNS* and HTTP* delivery services.
missLong	yesno	The longitude as decimal degrees to use when the client cannot be found in the CZF or the Geo lookup. e.g. -104.9847000 or null. Required for DNS* and HTTP* delivery services.
multiSiteOrigin	yesno	true if enabled, false if disabled. Required for DNS* and HTTP* delivery services.
orgServerFqdn	yesno	The origin server base URL (FQDN when used in this instance, includes the protocol ( <a href="http://">http://</a> or <a href="https://">https://</a> ) for use in retrieving content from the origin server. This field is required if type is DNS* or HTTP*.
originShield	no	Origin shield
profileId	no	DS profile ID
protocol	yesno	<ul style="list-style-type: none"> <li>• 0: serve with <a href="http://">http://</a> at EDGE</li> <li>• 1: serve with <a href="https://">https://</a> at EDGE</li> <li>• 2: serve with both <a href="http://">http://</a> and <a href="https://">https://</a> at EDGE</li> </ul> Required for DNS*, HTTP* or <i>STEERING</i> delivery services.

Continued on next page



Table 17 – continued from previous page

Parameter	Required	Description
qstringIgnore	yesno	<ul style="list-style-type: none"> <li>• 0: no special query string handling; it is for use in the cache-key and pass up to origin.</li> <li>• 1: ignore query string in cache-key, but pass it up to parent and or origin.</li> <li>• 2: drop query string at edge, and do not use it in the cache-key.</li> </ul> Required for DNS* and HTTP* delivery services.
rangeRequestHandling	yesno	How to treat range requests (required for DNS* and HTTP* delivery services): <ul style="list-style-type: none"> <li>• 0 Do not cache (ranges requested from files taht are already cached due to a non range request will be a HIT)</li> <li>• 1 Use the background_fetch plugin.</li> <li>• 2 Use the cache_range_requests plugin.</li> </ul>
regexRemap	no	Regex Remap rule to apply to this delivery service at the Edge tier.
regionalGeoBlocking	yes	Is the Regional Geo Blocking feature enabled.
remapText	no	Additional raw remap line text.
routingName	yes	The routing name of this deliveryservice, e.g. <routing-Name>.<xmlId>.cdn.com.
signed	no	<ul style="list-style-type: none"> <li>• false: token based auth (see :ref:token-based-auth) is not enabled for this deliveryservice.</li> <li>• true: token based auth is enabled for this delivery-service.</li> </ul>

Continued on next page

Table 17 – continued from previous page

Parameter	Required	Description
signingAlgorithm	no	<ul style="list-style-type: none"> <li>• null: token based auth (see :ref:token-based-auth) is not enabled for this deliveryservice.</li> <li>• “url_sig”: URL Sign token based auth is enabled for this deliveryservice.</li> <li>• “uri_signing”: URI Signing token based auth is enabled for this deliveryservice.</li> </ul>
sslKeyVersion	no	SSL key version
tenantId	No	Owning tenant ID
trRequestHeaders	no	Traffic router log request headers
trResponseHeaders	no	Traffic router additional response headers
typeId	yes	The type of this deliveryservice (one of :ref:to-api-v12-types use_in_table='deliveryservice').
xmlId	yes	Unique string that describes this deliveryservice.

**Request Example**

```
{
  "xmlId": "my_ds_1",
  "displayName": "my_ds_displayname_1",
  "tenantId": 1,
  "protocol": 1,
  "orgServerFqdn": "http://10.75.168.91",
  "cdnId": 2,
  "typeId": 42,
  "active": false,
  "dscp": 10,
  "geoLimit": 0,
  "geoProvider": 0,
  "initialDispersion": 1,
  "ipv6RoutingEnabled": false,
  "logsEnabled": false,
  "multiSiteOrigin": false,
  "missLat": 39.7391500,
  "missLong": -104.9847000,
  "qstringIgnore": 0,
  "rangeRequestHandling": 0,
  "regionalGeoBlocking": false,
  "signed": false,
  "signingAlgorithm": null
}
```

**Response Properties**

Parameter	Type	Description
active	bool	true if active, false if inactive.
cacheurl	string	Cache URL rule to apply to this delivery service.
ccrDnsTtl	int	The TTL of the DNS response for A or AAAA queries requesting the IP address of the tr. host.
cdnId	int	Id of the CDN to which the delivery service belongs to.
cdnName	string	Name of the CDN to which the delivery service belongs to.
checkPath	string	The path portion of the URL to check this deliveryservice for health.
deepCachingType	string	When to do Deep Caching for this Delivery Service: <ul style="list-style-type: none"> <li>• NEVER (default)</li> <li>• ALWAYS</li> </ul>
displayName	string	The display name of the delivery service.
dnsBypassCname	string	
dnsBypassIp	string	The IPv4 IP to use for bypass on a DNS deliveryservice - bypass starts when serving more than the globalMaxMbps traffic on this deliveryservice.
dnsBypassIp6	string	The IPv6 IP to use for bypass on a DNS deliveryservice - bypass starts when serving more than the globalMaxMbps traffic on this deliveryservice.
dnsBypassTtl	int	The TTL of the DNS bypass response.
dscp	int	The Differentiated Services Code Point (DSCP) with which to mark downstream (EDGE -> customer) traffic.
edgeHeaderRewrite	string	The EDGE header rewrite actions to perform.
exampleURLs	array	Entry points into the CDN for this deliveryservice.
geoLimitRedirectUrl	string	

Continued on next page

Table 18 – continued from previous page

Parameter	Type	Description
geoLimit	int	<ul style="list-style-type: none"> <li>• 0: None - no limitations</li> <li>• 1: Only route on CZF file hit</li> <li>• 2: Only route on CZF hit or when from USA</li> </ul> <p>Note that this does not prevent access to content or makes content secure; it just prevents routing to the content by Traffic Router.</p>
geoLimitCountries	string	
geoProvider	int	
globalMaxMbps	int	The maximum global bandwidth allowed on this deliveryservice. If exceeded, the traffic routes to the dnsByPassIp* for DNS deliveryservices and to the httpBypassFqdn for HTTP deliveryservices.
globalMaxTps	int	The maximum global transactions per second allowed on this deliveryservice. When this is exceeded traffic will be sent to the dnsByPassIp* for DNS deliveryservices and to the httpBypassFqdn for HTTP deliveryservices
httpBypassFqdn	string	The HTTP destination to use for bypass on an HTTP deliveryservice - bypass starts when serving more than the globalMaxMbps traffic on this deliveryservice.
id	int	The deliveryservice id (database row number).
infoUrl	string	Use this to add a URL that points to more information about that deliveryservice.
initialDispersion	int	
ipv6RoutingEnabled	bool	false: send IPv4 address of Traffic Router to client on HTTP type del.
lastUpdated	string	
logsEnabled	bool	
longDesc	string	Description field.
longDesc1	string	Description field 1.
longDesc2	string	Description field 2.
matchList	array	Array of matchList hashes.

Continued on next page

Table 18 – continued from previous page

Parameter	Type	Description
>>type	string	The type of MatchList (one of :ref:to-api-v11-types use_in_table='regex').
>>setNumber	string	The set Number of the match-List.
>>pattern	string	The regexp for the matchList.
maxDnsAnswers	int	The maximum number of IPs to put in a A/AAAA response for a DNS deliveryservice (0 means all available).
midHeaderRewrite	string	The MID header rewrite actions to perform.
missLat	float	The latitude as decimal degrees to use when the client cannot be found in the CZF or the Geo lookup. • e.g. 39.7391500 or null
missLong	float	The longitude as decimal degrees to use when the client cannot be found in the CZF or the Geo lookup. • e.g. -104.9847000 or null
multiSiteOrigin	bool	Is the Multi Site Origin feature enabled for this delivery service (0=false, 1=true). See <i>Multi Site Origin</i>
orgServerFqdn	string	The origin server base URL (FQDN when used in this instance, includes the protocol ( <a href="http://">http://</a> or <a href="https://">https://</a> ) for use in retrieving content from the origin server.
originShield	string	
profileDescription	string	The description of the Traffic Router Profile with which this deliveryservice is associated.
profileId	int	The id of the Traffic Router Profile with which this deliveryservice is associated.
profileName	string	The name of the Traffic Router Profile with which this deliveryservice is associated.

Continued on next page

Table 18 – continued from previous page

Parameter	Type	Description
protocol	int	<ul style="list-style-type: none"> <li>• 0: serve with <a href="#">http://</a> at EDGE</li> <li>• 1: serve with <a href="#">https://</a> at EDGE</li> <li>• 2: serve with both <a href="#">http://</a> and <a href="#">https://</a> at EDGE</li> </ul>
qstringIgnore	int	<ul style="list-style-type: none"> <li>• 0: no special query string handling; it is for use in the cache-key and pass up to origin.</li> <li>• 1: ignore query string in cache-key, but pass it up to parent and or origin.</li> <li>• 2: drop query string at edge, and do not use it in the cache-key.</li> </ul>
rangeRequestHandling	int	<p>How to treat range requests:</p> <ul style="list-style-type: none"> <li>• 0 Do not cache (ranges requested from files taht are already cached due to a non range request will be a HIT)</li> <li>• 1 Use the <a href="#">background_fetch</a> plugin.</li> <li>• 2 Use the <a href="#">cache_range_requests</a> plugin.</li> </ul>
regexRemap	string	Regex Remap rule to apply to this delivery service at the Edge tier.
regionalGeoBlocking	bool	Regex Remap rule to apply to this delivery service at the Edge tier.
remapText	string	Additional raw remap line text.
routingName	string	The routing name of this deliveryservice, e.g. <routing-Name>.<xmlId>.cdn.com.
signed	bool	<ul style="list-style-type: none"> <li>• false: token based auth (see <a href="#">:ref:token-based-auth</a>) is not enabled for this deliveryservice.</li> <li>• true: token based auth is enabled for this delivery-service.</li> </ul>

Continued on next page

Table 18 – continued from previous page

Parameter	Type	Description
signingAlgorithm	string	<ul style="list-style-type: none"> <li>• null: token based auth (see :ref:token-based-auth) is not enabled for this deliveryservice.</li> <li>• “url_sig”: URL Sign token based auth is enabled for this deliveryservice.</li> <li>• “uri_signing”: URI Signing token based auth is enabled for this deliveryservice.</li> </ul>
sslKeyVersion	int	
trRequestHeaders	string	
trResponseHeaders	string	
typeId	int	The type of this deliveryservice (one of :ref:to-api-v11-types use_in_table='deliveryservice').
xmlId	string	Unique string that describes this deliveryservice.

**Response Example**

```
{
  "response": [
    {
      "active": true,
      "cacheurl": null,
      "ccrDnsTtl": "3600",
      "cdnId": "2",
      "cdnName": "over-the-top",
      "checkPath": "",
      "deepCachingType": "NEVER",
      "displayName": "My Cool Delivery Service",
      "dnsBypassCname": "",
      "dnsBypassIp": "",
      "dnsBypassIp6": "",
      "dnsBypassTtl": "30",
      "dscp": "40",
      "edgeHeaderRewrite": null,
      "exampleURLs": [
        "http://foo.foo-ds.foo.bar.net"
      ],
      "geoLimit": "0",
      "geoLimitCountries": null,
      "geoLimitRedirectURL": null,
      "geoProvider": "0",
      "globalMaxMbps": null,
      "globalMaxTps": "0",
      "httpBypassFqdn": "",
      "id": "442",
      "infoUrl": "",

```

(continues on next page)

(continued from previous page)

```

    "initialDispersion": "1",
    "ipv6RoutingEnabled": true,
    "lastUpdated": "2016-01-26 08:49:35",
    "logsEnabled": false,
    "longDesc": "",
    "longDesc1": "",
    "longDesc2": "",
    "matchList": [
        {
            "pattern": ".*\\.foo-ds\\.\\.\\.*",
            "setNumber": "0",
            "type": "HOST_REGEX"
        }
    ],
    "maxDnsAnswers": "0",
    "midHeaderRewrite": null,
    "missLat": "39.7391500",
    "missLong": "-104.9847000",
    "multiSiteOrigin": false,
    "orgServerFqdn": "http://baz.boo.net",
    "originShield": null,
    "profileDescription": "Content Router for over-the-top",
    "profileId": "5",
    "profileName": "ROUTER_TOP",
    "protocol": "0",
    "qstringIgnore": "1",
    "rangeRequestHandling": "0",
    "regexRemap": null,
    "regionalGeoBlocking": false,
    "remapText": null,
    "routingName": "foo",
    "signed": false,
    "signingAlgorithm": null,
    "sslKeyVersion": "0",
    "tenantId": 1,
    "trRequestHeaders": null,
    "trResponseHeaders": "Access-Control-Allow-Origin: *",
    "type": "HTTP",
    "typeId": "8",
    "xmlId": "foo-ds"
}
]
}

```

**PUT /api/1.2/deliveryservices/{:id}**

Allows user to edit a delivery service.

Authentication Required: Yes

Role(s) Required: admin or oper

**Request Route Parameters**



Name	Required	Description
id	yes	delivery service id.

### Request Properties

Parameter	Required	Description
active	yes	true if active, false if inactive.
cacheurl	no	Cache URL rule to apply to this delivery service.
ccrDnsTtl	no	The TTL of the DNS response for A or AAAA queries requesting the IP address of the tr.host.
cdnId	yes	cdn id
checkPath	no	The path portion of the URL to check this deliveryservice for health.
deepCachingType	no	When to do Deep Caching for this Delivery Service: <ul style="list-style-type: none"> <li>• NEVER (default)</li> <li>• ALWAYS</li> </ul>
displayName	yes	Display name
dnsBypassCname	no	Bypass CNAME
dnsBypassIp	no	The IPv4 IP to use for bypass on a DNS deliveryservice - bypass starts when serving more than the globalMaxMbps traffic on this deliveryservice.
dnsBypassIp6	no	The IPv6 IP to use for bypass on a DNS deliveryservice - bypass starts when serving more than the globalMaxMbps traffic on this deliveryservice.
dnsBypassTtl	no	The TTL of the DNS bypass response.
dscp	yes	The Differentiated Services Code Point (DSCP) with which to mark downstream (EDGE -> customer) traffic.
edgeHeaderRewrite	no	The EDGE header rewrite actions to perform.
geoLimitRedirectURL	no	This is the URL Traffic Router will redirect to when Geo Limit Failure.

Continued on next page

Table 19 – continued from previous page

Parameter	Required	Description
geoLimit	yes	<ul style="list-style-type: none"> <li>• 0: None - no limitations</li> <li>• 1: Only route on CZF file hit</li> <li>• 2: Only route on CZF hit or when from geo limit countries</li> </ul> <p>Note that this does not prevent access to content or makes content secure; it just prevents routing to the content by Traffic Router.</p>
geoLimitCountries	no	The geo limit countries.
geoProvider	yes	<ul style="list-style-type: none"> <li>• 0: Maxmind(default)</li> <li>• 1: Neustar</li> </ul>
globalMaxMbps	no	The maximum global bandwidth allowed on this deliveryservice. If exceeded, the traffic routes to the dnsByPassIp* for DNS deliveryservices and to the httpBypassFqdn for HTTP deliveryservices.
globalMaxTps	no	The maximum global transactions per second allowed on this deliveryservice. When this is exceeded traffic will be sent to the dnsByPassIp* for DNS deliveryservices and to the httpBypassFqdn for HTTP deliveryservices
httpBypassFqdn	no	The HTTP destination to use for bypass on an HTTP deliveryservice - bypass starts when serving more than the globalMaxMbps traffic on this deliveryservice.
infoUrl	no	Use this to add a URL that points to more information about that deliveryservice.
initialDispersion	yesno	Initial dispersion. Required for HTTP* delivery services.
ipv6RoutingEnabled	yesno	false: send IPv4 address of Traffic Router to client on HTTP type del. Required for DNS*, HTTP* and STEERING* delivery services.

Continued on next page

Table 19 – continued from previous page

Parameter	Required	Description
logsEnabled	yes	<ul style="list-style-type: none"> <li>• false: No</li> <li>• true: Yes</li> </ul>
longDesc	no	Description field.
longDesc1	no	Description field 1.
longDesc2	no	Description field 2.
maxDnsAnswers	no	The maximum number of IPs to put in a A/AAAA response for a DNS deliveryservice (0 means all available).
midHeaderRewrite	no	The MID header rewrite actions to perform.
missLat	yesno	The latitude as decimal degrees to use when the client cannot be found in the CZF or the Geo lookup. e.g. 39.7391500 or null. Required for DNS* and HTTP* delivery services.
missLong	yesno	The longitude as decimal degrees to use when the client cannot be found in the CZF or the Geo lookup. e.g. -104.9847000 or null. Required for DNS* and HTTP* delivery services.
multiSiteOrigin	yesno	true if enabled, false if disabled. Required for DNS* and HTTP* delivery services.
orgServerFqdn	yesno	The origin server base URL (FQDN when used in this instance, includes the protocol ( <a href="http://">http://</a> or <a href="https://">https://</a> ) for use in retrieving content from the origin server. This field is required if type is DNS* or HTTP*.
originShield	no	Origin shield
profileId	no	DS profile ID
protocol	yesno	<ul style="list-style-type: none"> <li>• 0: serve with <a href="http://">http://</a> at EDGE</li> <li>• 1: serve with <a href="https://">https://</a> at EDGE</li> <li>• 2: serve with both <a href="http://">http://</a> and <a href="https://">https://</a> at EDGE</li> </ul> Required for DNS*, HTTP* or <i>STEERING</i> delivery services.

Continued on next page

Table 19 – continued from previous page

Parameter	Required	Description
qstringIgnore	yesno	<ul style="list-style-type: none"> <li>• 0: no special query string handling; it is for use in the cache-key and pass up to origin.</li> <li>• 1: ignore query string in cache-key, but pass it up to parent and or origin.</li> <li>• 2: drop query string at edge, and do not use it in the cache-key.</li> </ul> Required for DNS* and HTTP* delivery services.
rangeRequestHandling	yesno	How to treat range requests (required for DNS* and HTTP* delivery services): <ul style="list-style-type: none"> <li>• 0 Do not cache (ranges requested from files taht are already cached due to a non range request will be a HIT)</li> <li>• 1 Use the background_fetch plugin.</li> <li>• 2 Use the cache_range_requests plugin.</li> </ul>
regexRemap	no	Regex Remap rule to apply to this delivery service at the Edge tier.
regionalGeoBlocking	yes	Is the Regional Geo Blocking feature enabled.
remapText	no	Additional raw remap line text.
routingName	yes	The routing name of this deliveryservice, e.g. <routing-Name>.<xmlId>.cdn.com.
signed	no	<ul style="list-style-type: none"> <li>• false: token based auth (see :ref:token-based-auth) is not enabled for this deliveryservice.</li> <li>• true: token based auth is enabled for this delivery-service.</li> </ul>

Continued on next page

Table 19 – continued from previous page

Parameter	Required	Description
signingAlgorithm	no	<ul style="list-style-type: none"> <li>• null: token based auth (see :ref:token-based-auth) is not enabled for this deliveryservice.</li> <li>• “url_sig”: URL Sign token based auth is enabled for this deliveryservice.</li> <li>• “uri_signing”: URI Signing token based auth is enabled for this deliveryservice.</li> </ul>
sslKeyVersion	no	SSL key version
tenantId	No	Owning tenant ID
trRequestHeaders	no	Traffic router log request headers
trResponseHeaders	no	Traffic router additional response headers
typeId	yes	The type of this deliveryservice (one of :ref:to-api-v12-types use_in_table='deliveryservice').
xmlId	yes	Unique string that describes this deliveryservice. This value cannot be changed on update.

**Request Example**

```
{
  "xmlId": "my_ds_1",
  "displayName": "my_ds_displayname_1",
  "tenantId": 1,
  "protocol": 1,
  "orgServerFqdn": "http://10.75.168.91",
  "cdnId": 2,
  "typeId": 42,
  "active": false,
  "dscp": 10,
  "geoLimit": 0,
  "geoProvider": 0,
  "initialDispersion": 1,
  "ipv6RoutingEnabled": false,
  "logsEnabled": false,
  "multiSiteOrigin": false,
  "missLat": 39.7391500,
  "missLong": -104.9847000,
  "qstringIgnore": 0,
  "rangeRequestHandling": 0,
  "regionalGeoBlocking": false,
  "signed": false,
  "signingAlgorithm": null
}
```

**Response Properties**

Parameter	Type	Description
active	bool	true if active, false if inactive.
cacheurl	string	Cache URL rule to apply to this delivery service.
ccrDnsTtl	int	The TTL of the DNS response for A or AAAA queries requesting the IP address of the tr. host.
cdnId	int	Id of the CDN to which the delivery service belongs to.
cdnName	string	Name of the CDN to which the delivery service belongs to.
checkPath	string	The path portion of the URL to check this deliveryservice for health.
deepCachingType	string	When to do Deep Caching for this Delivery Service: <ul style="list-style-type: none"><li>• NEVER (default)</li><li>• ALWAYS</li></ul>
displayName	string	The display name of the delivery service.
dnsBypassCname	string	
dnsBypassIp	string	The IPv4 IP to use for bypass on a DNS deliveryservice - bypass starts when serving more than the globalMaxMbps traffic on this deliveryservice.
dnsBypassIp6	string	The IPv6 IP to use for bypass on a DNS deliveryservice - bypass starts when serving more than the globalMaxMbps traffic on this deliveryservice.
dnsBypassTtl	int	The TTL of the DNS bypass response.
dscp	int	The Differentiated Services Code Point (DSCP) with which to mark downstream (EDGE -> customer) traffic.
edgeHeaderRewrite	string	The EDGE header rewrite actions to perform.
exampleURLs	array	Entry points into the CDN for this deliveryservice.
geoLimitRedirectUrl	string	

Continued on next page

Table 20 – continued from previous page

Parameter	Type	Description
geoLimit	int	<ul style="list-style-type: none"> <li>• 0: None - no limitations</li> <li>• 1: Only route on CZF file hit</li> <li>• 2: Only route on CZF hit or when from USA</li> </ul> <p>Note that this does not prevent access to content or makes content secure; it just prevents routing to the content by Traffic Router.</p>
geoLimitCountries	string	
geoProvider	int	
globalMaxMbps	int	The maximum global bandwidth allowed on this deliveryservice. If exceeded, the traffic routes to the dnsByPassIp* for DNS deliveryservices and to the httpBypassFqdn for HTTP deliveryservices.
globalMaxTps	int	The maximum global transactions per second allowed on this deliveryservice. When this is exceeded traffic will be sent to the dnsByPassIp* for DNS deliveryservices and to the httpBypassFqdn for HTTP deliveryservices
httpBypassFqdn	string	The HTTP destination to use for bypass on an HTTP deliveryservice - bypass starts when serving more than the globalMaxMbps traffic on this deliveryservice.
id	int	The deliveryservice id (database row number).
infoUrl	string	Use this to add a URL that points to more information about that deliveryservice.
initialDispersion	int	
ipv6RoutingEnabled	bool	false: send IPv4 address of Traffic Router to client on HTTP type del.
lastUpdated	string	
logsEnabled	bool	
longDesc	string	Description field.
longDesc1	string	Description field 1.
longDesc2	string	Description field 2.
matchList	array	Array of matchList hashes.

Continued on next page

Table 20 – continued from previous page

Parameter	Type	Description
>>type	string	The type of MatchList (one of :ref:to-api-v11-types use_in_table='regex').
>>setNumber	string	The set Number of the match-List.
>>pattern	string	The regexp for the matchList.
maxDnsAnswers	int	The maximum number of IPs to put in a A/AAAA response for a DNS deliveryservice (0 means all available).
midHeaderRewrite	string	The MID header rewrite actions to perform.
missLat	float	The latitude as decimal degrees to use when the client cannot be found in the CZF or the Geo lookup. • e.g. 39.7391500 or null
missLong	float	The longitude as decimal degrees to use when the client cannot be found in the CZF or the Geo lookup. • e.g. -104.9847000 or null
multiSiteOrigin	bool	Is the Multi Site Origin feature enabled for this delivery service (0=false, 1=true). See <i>Multi Site Origin</i>
orgServerFqdn	string	The origin server base URL (FQDN when used in this instance, includes the protocol ( <a href="http://">http://</a> or <a href="https://">https://</a> ) for use in retrieving content from the origin server.
originShield	string	
profileDescription	string	The description of the Traffic Router Profile with which this deliveryservice is associated.
profileId	int	The id of the Traffic Router Profile with which this deliveryservice is associated.
profileName	string	The name of the Traffic Router Profile with which this deliveryservice is associated.

Continued on next page



Table 20 – continued from previous page

Parameter	Type	Description
protocol	int	<ul style="list-style-type: none"> <li>• 0: serve with <a href="#">http://</a> at EDGE</li> <li>• 1: serve with <a href="#">https://</a> at EDGE</li> <li>• 2: serve with both <a href="#">http://</a> and <a href="#">https://</a> at EDGE</li> </ul>
qstringIgnore	int	<ul style="list-style-type: none"> <li>• 0: no special query string handling; it is for use in the cache-key and pass up to origin.</li> <li>• 1: ignore query string in cache-key, but pass it up to parent and or origin.</li> <li>• 2: drop query string at edge, and do not use it in the cache-key.</li> </ul>
rangeRequestHandling	int	<p>How to treat range requests:</p> <ul style="list-style-type: none"> <li>• 0 Do not cache (ranges requested from files taht are already cached due to a non range request will be a HIT)</li> <li>• 1 Use the <a href="#">background_fetch</a> plugin.</li> <li>• 2 Use the <a href="#">cache_range_requests</a> plugin.</li> </ul>
regexRemap	string	Regex Remap rule to apply to this delivery service at the Edge tier.
regionalGeoBlocking	bool	Regex Remap rule to apply to this delivery service at the Edge tier.
remapText	string	Additional raw remap line text.
routingName	string	The routing name of this deliveryservice, e.g. <routing-Name>.<xmlId>.cdn.com.
signed	bool	<ul style="list-style-type: none"> <li>• false: token based auth (see <a href="#">:ref:token-based-auth</a>) is not enabled for this deliveryservice.</li> <li>• true: token based auth is enabled for this delivery-service.</li> </ul>

Continued on next page

Table 20 – continued from previous page

Parameter	Type	Description
signingAlgorithm	string	<ul style="list-style-type: none"> <li>• null: token based auth (see :ref:token-based-auth) is not enabled for this deliveryservice.</li> <li>• “url_sig”: URL Sign token based auth is enabled for this deliveryservice.</li> <li>• “uri_signing”: URI Signing token based auth is enabled for this deliveryservice.</li> </ul>
sslKeyVersion	int	
trRequestHeaders	string	
trResponseHeaders	string	
typeId	int	The type of this deliveryservice (one of :ref:to-api-v11-types use_in_table='deliveryservice').
xmlId	string	Unique string that describes this deliveryservice.

**Response Example**

```
{
  "response": [
    {
      "active": true,
      "cacheurl": null,
      "ccrDnsTtl": "3600",
      "cdnId": "2",
      "cdnName": "over-the-top",
      "checkPath": "",
      "deepCachingType": "NEVER",
      "displayName": "My Cool Delivery Service",
      "dnsBypassCname": "",
      "dnsBypassIp": "",
      "dnsBypassIp6": "",
      "dnsBypassTtl": "30",
      "dscp": "40",
      "edgeHeaderRewrite": null,
      "exampleURLs": [
        "http://foo.foo-ds.foo.bar.net"
      ],
      "geoLimit": "0",
      "geoLimitCountries": null,
      "geoLimitRedirectURL": null,
      "geoProvider": "0",
      "globalMaxMbps": null,
      "globalMaxTps": "0",
      "httpBypassFqdn": "",
      "id": "442",
      "infoUrl": "",

```

(continues on next page)

(continued from previous page)

```

    "initialDispersion": "1",
    "ipv6RoutingEnabled": true,
    "lastUpdated": "2016-01-26 08:49:35",
    "logsEnabled": false,
    "longDesc": "",
    "longDesc1": "",
    "longDesc2": "",
    "matchList": [
      {
        "pattern": ".*\\.foo-ds\\.\\.\\.*",
        "setNumber": "0",
        "type": "HOST_REGEX"
      }
    ],
    "maxDnsAnswers": "0",
    "midHeaderRewrite": null,
    "missLat": "39.7391500",
    "missLong": "-104.9847000",
    "multiSiteOrigin": false,
    "orgServerFqdn": "http://baz.boo.net",
    "originShield": null,
    "profileDescription": "Content Router for over-the-top",
    "profileId": "5",
    "profileName": "ROUTER_TOP",
    "protocol": "0",
    "qstringIgnore": "1",
    "rangeRequestHandling": "0",
    "regexRemap": null,
    "regionalGeoBlocking": false,
    "remapText": null,
    "routingName": "foo",
    "signed": false,
    "signingAlgorithm": null,
    "sslKeyVersion": "0",
    "tenantId": 1,
    "trRequestHeaders": null,
    "trResponseHeaders": "Access-Control-Allow-Origin: *",
    "type": "HTTP",
    "typeId": "8",
    "xmlId": "foo-ds"
  }
]
}

```

**PUT /api/1.2/deliveryservices/{:id}/safe**

Allows a user to edit limited fields of an assigned delivery service.

Authentication Required: Yes

Role(s) Required: users with the delivery service assigned or ops and above

**Request Route Parameters**

Name	Required	Description
id	yes	delivery service id.

### Request Properties

Parameter	Re-quired	Description
display-Name	no	Display name
infoUrl	no	Use this to add a URL that points to more information about that delivery-service.
longDesc	no	Description field.
longDesc1	no	Description field 1.
all other fields	n/a	All other fields will be silently ignored

### Request Example

```
{
  "displayName": "My Cool Delivery Service",
  "infoUrl": "www.info.com",
  "longDesc": "some info about the service",
  "longDesc1": "the customer label"
}
```

### Response Properties

Parameter	Type	Description
active	bool	true if active, false if inactive.
cacheurl	string	Cache URL rule to apply to this delivery service.
ccrDnsTtl	int	The TTL of the DNS response for A or AAAA queries requesting the IP address of the tr. host.
cdnId	int	Id of the CDN to which the delivery service belongs to.
cdnName	string	Name of the CDN to which the delivery service belongs to.
checkPath	string	The path portion of the URL to check this deliveryservice for health.
deepCachingType	string	When to do Deep Caching for this Delivery Service: <ul style="list-style-type: none"><li>• NEVER (default)</li><li>• ALWAYS</li></ul>
displayName	string	The display name of the delivery service.
dnsBypassCname	string	

Continued on next page

Table 21 – continued from previous page

Parameter	Type	Description
<code>dnsBypassIp</code>	string	The IPv4 IP to use for bypass on a DNS deliveryservice - bypass starts when serving more than the <code>globalMaxMbps</code> traffic on this deliveryservice.
<code>dnsBypassIp6</code>	string	The IPv6 IP to use for bypass on a DNS deliveryservice - bypass starts when serving more than the <code>globalMaxMbps</code> traffic on this deliveryservice.
<code>dnsBypassTtl</code>	int	The TTL of the DNS bypass response.
<code>dscp</code>	int	The Differentiated Services Code Point (DSCP) with which to mark downstream (EDGE -> customer) traffic.
<code>edgeHeaderRewrite</code>	string	The EDGE header rewrite actions to perform.
<code>exampleURLs</code>	array	Entry points into the CDN for this deliveryservice.
<code>geoLimitRedirectUrl</code>	string	
<code>geoLimit</code>	int	<ul style="list-style-type: none"> <li>• 0: None - no limitations</li> <li>• 1: Only route on CZF file hit</li> <li>• 2: Only route on CZF hit or when from USA</li> </ul> <p>Note that this does not prevent access to content or makes content secure; it just prevents routing to the content by Traffic Router.</p>
<code>geoLimitCountries</code>	string	
<code>geoProvider</code>	int	
<code>globalMaxMbps</code>	int	The maximum global bandwidth allowed on this deliveryservice. If exceeded, the traffic routes to the <code>dnsByPassIp*</code> for DNS deliveryservices and to the <code>httpBypassFqdn</code> for HTTP deliveryservices.
<code>globalMaxTps</code>	int	The maximum global transactions per second allowed on this deliveryservice. When this is exceeded traffic will be sent to the <code>dnsByPassIp*</code> for DNS deliveryservices and to the <code>httpBypassFqdn</code> for HTTP deliveryservices

Continued on next page

Table 21 – continued from previous page

Parameter	Type	Description
httpBypassFqdn	string	The HTTP destination to use for bypass on an HTTP deliveryservice - bypass starts when serving more than the global-MaxMbps traffic on this deliveryservice.
id	int	The deliveryservice id (database row number).
infoUrl	string	Use this to add a URL that points to more information about that deliveryservice.
initialDispersion	int	
ipv6RoutingEnabled	bool	false: send IPv4 address of Traffic Router to client on HTTP type del.
lastUpdated	string	
logsEnabled	bool	
longDesc	string	Description field.
longDesc1	string	Description field 1.
longDesc2	string	Description field 2.
matchList	array	Array of matchList hashes.
>>type	string	The type of MatchList (one of :ref:to-api-v11-types use_in_table='regex').
>>setNumber	string	The set Number of the match-List.
>>pattern	string	The regexp for the matchList.
maxDnsAnswers	int	The maximum number of IPs to put in a A/AAAA response for a DNS deliveryservice (0 means all available).
midHeaderRewrite	string	The MID header rewrite actions to perform.
missLat	float	The latitude as decimal degrees to use when the client cannot be found in the CZF or the Geo lookup. <ul style="list-style-type: none"> <li>• e.g. 39.7391500 or null</li> </ul>
missLong	float	The longitude as decimal degrees to use when the client cannot be found in the CZF or the Geo lookup. <ul style="list-style-type: none"> <li>• e.g. -104.9847000 or null</li> </ul>
multiSiteOrigin	bool	Is the Multi Site Origin feature enabled for this delivery service (0=false, 1=true). See <a href="#">Multi Site Origin</a>

Continued on next page

Table 21 – continued from previous page

Parameter	Type	Description
orgServerFqdn	string	The origin server base URL (FQDN when used in this instance, includes the protocol ( <a href="http://">http://</a> or <a href="https://">https://</a> ) for use in retrieving content from the origin server.
originShield	string	
profileDescription	string	The description of the Traffic Router Profile with which this deliveryservice is associated.
profileId	int	The id of the Traffic Router Profile with which this deliveryservice is associated.
profileName	string	The name of the Traffic Router Profile with which this deliveryservice is associated.
protocol	int	<ul style="list-style-type: none"> <li>• 0: serve with <a href="http://">http://</a> at EDGE</li> <li>• 1: serve with <a href="https://">https://</a> at EDGE</li> <li>• 2: serve with both <a href="http://">http://</a> and <a href="https://">https://</a> at EDGE</li> </ul>
qstringIgnore	int	<ul style="list-style-type: none"> <li>• 0: no special query string handling; it is for use in the cache-key and pass up to origin.</li> <li>• 1: ignore query string in cache-key, but pass it up to parent and or origin.</li> <li>• 2: drop query string at edge, and do not use it in the cache-key.</li> </ul>
rangeRequestHandling	int	<p>How to treat range requests:</p> <ul style="list-style-type: none"> <li>• 0 Do not cache (ranges requested from files taht are already cached due to a non range request will be a HIT)</li> <li>• 1 Use the <a href="#">background_fetch</a> plugin.</li> <li>• 2 Use the <a href="#">cache_range_requests</a> plugin.</li> </ul>
regexRemap	string	Regex Remap rule to apply to this delivery service at the Edge tier.

Continued on next page

Table 21 – continued from previous page

Parameter	Type	Description
regionalGeoBlocking	bool	Regex Remap rule to apply to this delivery service at the Edge tier.
remapText	string	Additional raw remap line text.
routingName	string	The routing name of this deliveryservice, e.g. <routing-Name>.<xmlId>.cdn.com.
signed	bool	<ul style="list-style-type: none"> <li>• false: token based auth (see :ref:token-based-auth) is not enabled for this deliveryservice.</li> <li>• true: token based auth is enabled for this deliveryservice.</li> </ul>
signingAlgorithm	string	<ul style="list-style-type: none"> <li>• null: token based auth (see :ref:token-based-auth) is not enabled for this deliveryservice.</li> <li>• “url_sig”: URL Sign token based auth is enabled for this deliveryservice.</li> <li>• “uri_signing”: URI Signing token based auth is enabled for this deliveryservice.</li> </ul>
sslKeyVersion	int	
trRequestHeaders	string	
trResponseHeaders	string	
typeId	int	The type of this deliveryservice (one of :ref:to-api-v11-types use_in_table='deliveryservice').
xmlId	string	Unique string that describes this deliveryservice.

**Response Example**

```
{
  "response": [
    {
      "active": true,
      "cacheurl": null,
      "ccrDnsTtl": "3600",
      "cdnId": "2",
      "cdnName": "over-the-top",
      "checkPath": "",
      "deepCachingType": "NEVER",
      "displayName": "My Cool Delivery Service",
      "dnsBypassCname": "",

```

(continues on next page)



(continued from previous page)

```

"dnsBypassIp": "",
"dnsBypassIp6": "",
"dnsBypassTtl": "30",
"dscp": "40",
"edgeHeaderRewrite": null,
"exampleURLs": [
    "http://foo.foo-ds.foo.bar.net"
],
"geoLimit": "0",
"geoLimitCountries": null,
"geoLimitRedirectURL": null,
"geoProvider": "0",
"globalMaxMbps": null,
"globalMaxTps": "0",
"httpBypassFqdn": "",
"id": "442",
"infoUrl": "www.info.com",
"initialDispersion": "1",
"ipv6RoutingEnabled": true,
"lastUpdated": "2016-01-26 08:49:35",
"logsEnabled": false,
"longDesc": "some info about the service",
"longDesc1": "the customer label",
"longDesc2": "",
"matchList": [
    {
        "pattern": ".*\\.foo-ds\\..*",
        "setNumber": "0",
        "type": "HOST_REGEX"
    }
],
"maxDnsAnswers": "0",
"midHeaderRewrite": null,
"missLat": "39.7391500",
"missLong": "-104.9847000",
"multiSiteOrigin": false,
"orgServerFqdn": "http://baz.boo.net",
"originShield": null,
"profileDescription": "Content Router for over-the-top",
"profileId": "5",
"profileName": "ROUTER_TOP",
"protocol": "0",
"qstringIgnore": "1",
"rangeRequestHandling": "0",
"regexRemap": null,
"regionalGeoBlocking": false,
"remapText": null,
"routingName": "foo",
"signed": false,
"signingAlgorithm": null,
"sslKeyVersion": "0",
"tenantId": 1,
"trRequestHeaders": null,
"trResponseHeaders": "Access-Control-Allow-Origin: *",
"type": "HTTP",
"typeId": "8",
"xmlId": "foo-ds"

```

(continues on next page)

(continued from previous page)

```
}
]
}
```

**DELETE /api/1.2/deliveryservices/{:id}**

Allows user to delete a delivery service.

Authentication Required: Yes

Role(s) Required: admin or oper

**Request Route Parameters**

Name	Required	Description
id	yes	delivery service id.

**Response Example**

```
{
  "alerts": [
    {
      "level": "success",
      "text": "Delivery service was deleted."
    }
  ],
}
```

**POST /api/1.2/deliveryservices/:xml\_id/servers**

Assign caches to a delivery service.

Authentication Required: Yes

Role(s) Required: admin or oper

**Request Route Parameters**

Name	Required	Description
xml_id	yes	the xml_id of the deliveryservice

**Request Properties**

Parameter	Required	Description
server-Names	yes	array of hostname of cache servers to assign to this deliveryservice, for example: [ "server1", "server2" ]

**Request Example**

```
{
  "serverNames": [
    "tcl_atl1"
  ]
}
```

**Response Properties**

Parameter	Type	Description
xml_id	string	Unique string that describes this delivery service.
server-Names	string	array of hostname of cache servers to assign to this deliveryservice, for example: [ "server1", "server2" ]

**Response Example**

```
{
  "response": {
    "serverNames": [
      "tcl_atl1"
    ],
    "xmlId": "my_ds_1"
  }
}
```

**URI Signing Keys****DELETE /api/1.2/deliveryservices/:xml\_id/urisignkeys**

Deletes URISigning objects for a delivery service.

Authentication Required: Yes

Role(s) Required: admin

**Request Route Parameters**

Name	Required	Description
xml_id	yes	xml_id of the desired delivery service

**GET /api/1.2/deliveryservices/:xml\_id/urisignkeys**

Retrieves one or more URISigning objects for a delivery service.

Authentication Required: Yes

Role(s) Required: admin

**Request Route Parameters**

Name	Required	Description
xml_id	yes	xml_id of the desired delivery service

### Response Properties

Parameter	Type	Description
Issuer	string	a string describing the issuer of the URI signing object. Multiple URISigning objects may be returned in a response, see example
renewal_kid	string	a string naming the jwt key used for renewals.
keys	string	json array of jwt symmetric keys .
alg	string	this parameter repeats for each jwt key in the array and specifies the jwa encryption algorithm to use with this key, RFC 7518.
kid	string	this parameter repeats for each jwt key in the array and specifies the unique id for the key as defined in RFC 7516.
kty	string	this parameter repeats for each jwt key in the array and specifies the key type as defined in RFC 7516.
k	string	this parameter repeats for each jwt key in the array and specifies the base64 encoded symmetric key see RFC 7516.

### Response Example

```
{
  "Kabletown URI Authority": {
    "renewal_kid": "Second Key",
    "keys": [
      {
        "alg": "HS256",
        "kid": "First Key",
        "kty": "oct",
        "k": "Kh_RkUMj-fzbD37qBnDf_3e_RvQ3RP9PaSmVEpE24AM"
      },
      {
        "alg": "HS256",
        "kid": "Second Key",
        "kty": "oct",
        "k": "fZBpDBNbk2GqhwoB_DGBAsBxqQZVix04rIoLJ7p_RlE"
      }
    ]
  }
}
```

### POST /api/1.2/deliveryservices/:xml\_id/urisignkeys

Assigns URISigning objects to a delivery service.

Authentication Required: Yes

Role(s) Required: admin

#### Request Route Parameters

Name	Required	Description
xml_id	yes	xml_id of the desired delivery service

### Request Properties

Parameter	Type	Description
Issuer	string	a string describing the issuer of the URI signing object. Multiple URISigning objects may be returned in a response, see example
renewal_kid	string	a string naming the jwt key used for renewals.
keys	string	json array of jwt symmetric keys .
alg	string	this parameter repeats for each jwt key in the array and specifies the jwa encryption algorithm to use with this key, RFC 7518.
kid	string	this parameter repeats for each jwt key in the array and specifies the unique id for the key as defined in RFC 7516.
ktypes	string	this parameter repeats for each jwt key in the array and specifies the key type as defined in RFC 7516.
k	string	this parameter repeats for each jwt key in the array and specifies the base64 encoded symmetric key see RFC 7516.

### Request Example

```
{
  "Kabletown URI Authority": {
    "renewal_kid": "Second Key",
    "keys": [
      {
        "alg": "HS256",
        "kid": "First Key",
        "kty": "oct",
        "k": "Kh_RkUMj-fzbD37qBnDf_3e_RvQ3RP9PaSmVEpE24AM"
      },
      {
        "alg": "HS256",
        "kid": "Second Key",
        "kty": "oct",
        "k": "fZBpDBNbK2GqhwoB_DGBAsBxqQZVix04rIoLJ7p_RlE"
      }
    ]
  }
}
```

### PUT /api/1.2/deliveryservices/:xml\_id/urisignkeys

updates URISigning objects on a delivery service.

Authentication Required: Yes

Role(s) Required: admin

### Request Route Parameters

Name	Required	Description
xml_id	yes	xml_id of the desired delivery service

### Request Properties

Parameter	Type	Description
Issuer	string	a string describing the issuer of the URI signing object. Multiple URISigning objects may be returned in a response, see example
renewal_kid	string	a string naming the jwt key used for renewals.
keys	string	json array of jwt symmetric keys .
alg	string	this parameter repeats for each jwt key in the array and specifies the jwa encryption algorithm to use with this key, RFC 7518.
kid	string	this parameter repeats for each jwt key in the array and specifies the unique id for the key as defined in RFC 7516.
kty	string	this parameter repeats for each jwt key in the array and specifies the key type as defined in RFC 7516.
k	string	this parameter repeats for each jwt key in the array and specifies the base64 encoded symmetric key see RFC 7516.

### Request Example

```
{
  "Kabletown URI Authority": {
    "renewal_kid": "Second Key",
    "keys": [
      {
        "alg": "HS256",
        "kid": "First Key",
        "kty": "oct",
        "k": "Kh_RkUMj-fzbD37qBnDf_3e_RvQ3RP9PaSmVEpE24AM"
      },
      {
        "alg": "HS256",
        "kid": "Second Key",
        "kty": "oct",
        "k": "fZBpDBNbk2GqhwoB_DGBAsBxqQZVix04rIoLJ7p_RlE"
      }
    ]
  }
}
```

## Delivery Service Regexes

### GET /api/1.2/deliveryservices\_regexes

Retrieves regexes for all delivery services.

Authentication Required: Yes

Role(s) Required: Admin or Oper

### Response Properties

Parameter	Type	Description
dsName	array	Delivery service name.
regexes	array	An array of regexes for the delivery service.
>type	string	The regex type.
>pattern	string	The regex pattern.
>setNumber	string	The order in which the regex is evaluated.

### Response Example

```
{
  "response": [
    {
      "dsName": "foo-bar",
      "regexes": [
        {
          "type": "HOST_REGEX",
          "pattern": ".*\\.foo-bar\\..*",
          "setNumber": 0
        },
        {
          "type": "HOST_REGEX",
          "pattern": "foo.bar.com",
          "setNumber": 1
        }
      ]
    },
    { ... }
  ]
}
```

### GET /api/1.2/deliveryservices/{:dsId}/regexes

Retrieves regexes for a specific delivery service.

Authentication Required: Yes

Role(s) Required: None

#### Request Route Parameters

Name	Required	Description
dsId	yes	Delivery service id.

#### Response Properties

Parameter	Type	Description
id	string	Delivery service regex ID.
type	string	Delivery service regex type ID.
typeName	string	Delivery service regex type name.
pattern	string	Delivery service regex pattern.
setNumber	string	The order in which the regex is evaluated for the delivery service.

**Response Example**

```
{
  "response": [
    {
      "id": 852,
      "type": 18,
      "typeName": "HOST_REGEX",
      "pattern": ".*\\.foo-bar\\..*",
      "setNumber": 0
    },
    {
      "id": 853,
      "type": 18,
      "typeName": "HOST_REGEX",
      "pattern": "foo.bar.com",
      "setNumber": 1
    }
  ]
}
```

**GET /api/1.2/deliveryservices/{:dsId}/regexes/{:id}**

Retrieves a regex for a specific delivery service.

Authentication Required: Yes

Role(s) Required: None

**Request Route Parameters**

Name	Required	Description
dsId	yes	Delivery service id.
id	yes	Delivery service regex id.

**Response Properties**

Parameter	Type	Description
id	string	Delivery service regex ID.
type	string	Delivery service regex type ID.
typeName	string	Delivery service regex type name.
pattern	string	Delivery service regex pattern.
setNumber	string	The order in which the regex is evaluated for the delivery service.

**Response Example**

```
{
  "response": [
    {
      "id": 852,
      "type": 18,
      "typeName": "HOST_REGEX",
```

(continues on next page)



(continued from previous page)

```
{
  "pattern": ".*\\.foo-bar\\..*",
  "setNumber": 0
}
```

## POST /api/1.2/deliveryservices/{:dsId}/regexes

Create a regex for a delivery service.

Authentication Required: Yes

Role(s) Required: Admin or Oper

### Request Route Parameters

Name	Required	Description
dsId	yes	Delivery service id.

### Request Properties

Parameter	Required	Description
pattern	yes	Regex pattern.
type	yes	Regex type ID.
setNumber	yes	Regex type ID.

### Request Example

```
{
  "pattern": ".*\\.foo-bar\\..*"
  "type": 18
  "setNumber": 0
}
```

### Response Properties

Parameter	Type	Description
id	string	Delivery service regex ID.
type	string	Delivery service regex type ID.
typeName	string	Delivery service regex type name.
pattern	string	Delivery service regex pattern.
setNumber	string	The order in which the regex is evaluated for the delivery service.

### Response Example

```
{
  "response": {
    "id": 852,
    "type": 18,
    "typeName": "HOST_REGEX",
    "pattern": ".*\\.foo-bar\\..*",
    "setNumber": 0
  },
  "alerts": [
    {
      "level": "success",
      "text": "Delivery service regex creation was successful."
    }
  ]
}
```

### PUT /api/1.2/deliveryservices/{:dsId}/regexes/{:id}

Update a regex for a delivery service.

Authentication Required: Yes

Role(s) Required: Admin or Oper

#### Request Route Parameters

Name	Required	Description
dsId	yes	Delivery service id.
id	yes	Delivery service regex id.

#### Request Properties

Parameter	Required	Description
pattern	yes	Regex pattern.
type	yes	Regex type ID.
setNumber	yes	Regex type ID.

#### Request Example

```
{
  "pattern": ".*\\.foo-bar\\..*"
  "type": 18
  "setNumber": 0
}
```

#### Response Properties

Parameter	Type	Description
id	string	Delivery service regex ID.
type	string	Delivery service regex type ID.
typeName	string	Delivery service regex type name.
pattern	string	Delivery service regex pattern.
setNumber	string	The order in which the regex is evaluated for the delivery service.

**Response Example**

```
{
  "response": {
    "id": 852,
    "type": 18,
    "typeName": "HOST_REGEX",
    "pattern": ".*\\.foo-bar\\..*",
    "setNumber": 0
  },
  "alerts": [
    {
      "level": "success",
      "text": "Delivery service regex update was successful."
    }
  ]
}
```

**DELETE /api/1.2/deliveryservices/{:dsId}/regexes/{:id}**

Delete delivery service regex.

Authentication Required: Yes

Role(s) Required: Admin or Oper

**Request Route Parameters**

Name	Required	Description
dsId	yes	Delivery service id.
id	yes	Delivery service regex id.

**Response Properties**

Parameter	Type	Description
alerts	array	A collection of alert messages.
>level	string	Success, info, warning or error.
>text	string	Alert message.

**Response Example**

```
{
  "alerts": [
```

(continues on next page)

(continued from previous page)

```
        {
            "level": "success",
            "text": "Delivery service regex delete was_
↪successful."
        },
    ],
}
```

## Delivery Service Statistics

### /api/1.2/deliveryservice\_stats

#### GET /api/1.2/deliveryservice\_stats.json

Retrieves statistics on the delivery services. See also [Using Traffic Ops - Delivery Service](#).

Authentication Required: Yes

Role(s) Required: None

#### Request Query Parameters

Name	Re-quired	Description
deliveryServiceName	yes	The delivery service with the desired stats
metricType	yes	The metric type (valid metric types: 'kpbs', 'out_bytes', 'status_4xx', 'status_5xx', 'tps_total', 'tps_2xx', 'tps_3xx', 'tps_4xx', 'tps_5xx')
startDate	yes	The begin date (Formatted as ISO8601, for example: '2015-08-11T12:30:00-06:00')
endDate	yes	The end date (Formatted as ISO8601, for example: '2015-08-12T12:30:00-06:00')

#### Response Properties

Parameter	Type	Description
source	string	The source of the data
summary	hash	Summary data
>totalBytes	float	
>count	int	
>min	float	
>max	float	
>fifthPercentile	float	
>ninetyEighthPercentile	float	
>ninetyFifthPercentile	float	
>average	float	
>totalTransactions	int	
series	hash	Series data
>count	int	
>columns	array	
>name	string	
>values	array	
>>time	string	
>>value	float	

### Response Example

```
{
  "response": {
    "source": "TrafficStats",
    "summary": {
      "average": 1081172.785,
      "count": 28,
      "fifthPercentile": 888827.26,
      "max": 1326680.31,
      "min": 888827.26,
      "ninetyEighthPercentile": 1324785.47,
      "ninetyFifthPercentile": 1324785.47,
      "totalBytes": 37841047.475,
      "totalTransactions": 1020202030101
    },
    "series": {
      "columns": [
        "time",
        ""
      ],
      "count": 60,
      "name": "kbps",
      "tags": {
        "cachegroup": "total"
      },
      "values": [
        [
          "2015-08-11T11:36:00Z",
          888827.26
        ],
        [
          "2015-08-11T11:37:00Z",
          980336.563333333
        ]
      ]
    }
  }
}
```

(continues on next page)

(continued from previous page)

```
[
  [
    "2015-08-11T11:38:00Z",
    952111.975
  ],
  [
    "2015-08-11T11:39:00Z",
    null
  ],
  [
    "2015-08-11T11:43:00Z",
    null
  ],
  [
    "2015-08-11T11:44:00Z",
    934682.943333333
  ],
  [
    "2015-08-11T11:45:00Z",
    1251121.28
  ],
  [
    "2015-08-11T11:46:00Z",
    1111012.99
  ]
]
}
```

## Divisions

### /api/1.2/divisions

**GET /api/1.2/divisions** Get all divisions.

Authentication Required: Yes

Role(s) Required: None

#### Response Properties

Parameter	Type	Description
id	string	Division id
lastUpdated	string	
name	string	Division name

#### Response Example

```
{
  "response": [
```

(continues on next page)

(continued from previous page)

```

{
  "id": "1"
  "name": "Central",
  "lastUpdated": "2014-10-02 08:22:43"
},
{
  "id": "2"
  "name": "West",
  "lastUpdated": "2014-10-02 08:22:43"
}
]
}

```

**GET /api/1.2/divisions/:id** Get division by Id.

Authentication Required: Yes

Role(s) Required: None

#### Response Properties

Parameter	Type	Description
id	string	Division id
lastUpdated	string	
name	string	Division name

#### Response Example

```

{
  "response": [
    {
      "id": "1"
      "name": "Central",
      "lastUpdated": "2014-10-02 08:22:43"
    }
  ]
}

```

**PUT /api/1.2/divisions/:id** Update a division

Authentication Required: Yes

Role(s) Required: admin or oper

#### Request Route Parameters

Name	Type	Description
id	int	Division id.

**Request Properties**

Parameter	Required	Description
name	yes	The name of the division

**Request Example**

```
{
  "name": "mydivision1"
}
```

**Response Properties**

Parameter	Type	Description
name	string	
id	string	
lastUpdated	string	

**Response Example**

```
{
  "alerts": [
    {
      "level": "success",
      "text": "Division update was successful."
    }
  ],
  "response": {
    "id": "1",
    "lastUpdated": "2014-03-18 08:57:39",
    "name": "mydivision1"
  }
}
```

**POST /api/1.2/divisions** Create division

Authentication Required: Yes

Role(s) Required: admin or oper

**Request Properties**

Parameter	Required	Description
name	yes	The name of the division

**Request Example**



```
{
  "name": "mydivision1"
}
```

### Response Properties

Parameter	Type	Description
name	string	
id	string	

### Response Example

```
{
  "response": {
    "name": "mydivision1",
    "id": "4"
  }
}
```

## Federation

### /api/1.2/federations

#### GET /api/1.2/federations.json

Retrieves a list of federation mappings (aka federation resolvers) for a the current user.

Authentication Required: Yes

Role(s) Required: Federation

### Response Properties

Parameter	Type	Description
cname	string	
ttl	int	Time to live for the cname.
deliveryService	string	Unique string that describes the deliveryservice.

### Response Example

```
{
  "response": [
    {
      "mappings": [
        "cname": "cname-01.",
        "ttl": 8865,
```

(continues on next page)

(continued from previous page)

```
    ]
    "deliveryService": "ds-01",
  }
]
}
```

### POST /api/1.2/federations.json

Allows a user to add federations for their delivery service(s).

Authentication Required: Yes

Role(s) Required: Federation

#### Request Properties

Parameter	Type	Description
deliveryService	string	Unique string that describes the deliveryservice.
resolve4	array	Array of IPv4 Addresses.
resolve6	array	Array of IPv6 Addresses.

#### Request Example

```
{
  "federations": [
    {
      "deliveryService": "ccp-omg-01",
      "mappings": {
        "resolve4": [
          "255.255.255.255"
        ],
        "resolve6": [
          "FE80::0202:B3FF:FE1E:8329",
        ]
      }
    }
  ]
}
```

### DELETE /api/1.2/federations.json

Deletes **all** federations associated with a user's delivery service(s).

Authentication Required: Yes

Role(s) Required: Federation

**PUT /api/1.2/federations.json**

Deletes **all** federations associated with a user's delivery service(s) then adds the new federations.

Authentication Required: Yes

Role(s) Required: Federation

**Request Properties**

Parameter	Type	Description
deliveryService	string	Unique string that describes the deliveryservice.
resolve4	array	Array of IPv4 Addresses.
resolve6	array	Array of IPv6 Addresses.

**Request Example**

```
{
  "federations": [
    {
      "deliveryService": "ccp-omg-01",
      "mappings": {
        "resolve4": [
          "255.255.255.255"
        ],
        "resolve6": [
          "FE80::0202:B3FF:FE1E:8329",
        ]
      }
    }
  ]
}
```

**GET /api/1.2/cdns/:name/federations**

Retrieves a list of federations for a cdn.

Authentication Required: Yes

Role(s) Required: None

**Response Properties**

Parameter	Type	Description
cname	string	
tTl	int	Time to live for the cname.
deliveryService	hash	
>>id	int	Delivery service ID
>>xmlId	string	Delivery service xml id

**Response Example**

```
{
  "response": [
    {
      "id": 41
      "cname": "booya.com.",
      "ttl": 34,
      "description": "fooya",
      "deliveryService": {
        "id": 61,
        "xmlId": "the-xml-id"
      }
    }
  ]
}
```

### GET /api/1.2/cdns/:name/federations/:id

Retrieves a federation for a cdn.

Authentication Required: Yes

Role(s) Required: None

#### Request Route Parameters

Name	Type	Description
cdn	string	CDN name.
federation	string	Federation ID.

#### Response Properties

Parameter	Type	Description
cname	string	
ttl	int	Time to live for the cname.
deliveryService	hash	
>>id	int	Delivery service ID
>>xmlId	string	Delivery service xml id

#### Response Example

```
{
  "response": [
    {
      "id": 41
      "cname": "booya.com.",
      "ttl": 34,
      "description": "fooya",
      "deliveryService": {
        "id": 61,
        "xmlId": "the-xml-id"
      }
    }
  ]
}
```

(continues on next page)

(continued from previous page)

```

    }
  ]
}
```

**POST /api/1.2/cdns/:name/federations** Create a federation

Authentication Required: Yes

Role(s) Required: Admin

**Request Route Parameters**

Name	Type	Description
cdn	string	CDN name.

**Request Properties**

Parameter	Required	Description
cname	yes	CNAME ending with a dot
ttl	yes	TTL
description	no	Description

**Request Example**

```

{
  "cname": "the.cname.com.",
  "ttl": 48,
  "description": "the description"
}
```

**Response Properties**

Parameter	Type	Description
cname	string	
ttl	string	
description	string	

**Response Example**

```

{
  "alerts": [
    {
      "level": "success",
      "text": "Federation created [ cname = the.cname. ]"
    }
  ]
}
```

↪with id: 26."

(continues on next page)

(continued from previous page)

```
    },
    "response": {
      "id": 26,
      "cname": "the.cname.com.",
      "ttl": 48,
      "description": "the description",
    }
  }
}
```

**PUT /api/1.2/cdns/:name/federations/:id** Update a federation

Authentication Required: Yes

Role(s) Required: Admin

**Request Route Parameters**

Name	Type	Description
cdn	string	CDN name.
federation	string	Federation ID.

**Request Properties**

Parameter	Required	Description
cname	yes	CNAME ending with a dot
ttl	yes	TTL
description	no	Description

**Request Example**

```
{
  "cname": "the.cname.com.",
  "ttl": 48,
  "description": "the description"
}
```

**Response Properties**

Parameter	Type	Description
cname	string	
ttl	string	
description	string	

**Response Example**

```
{
  "alerts": [
    {
      "level": "success",
      "text": "Federation updated [ cname = the.cname. ]"
    }
  ],
  "response": {
    "id": 26,
    "cname": "the.cname.com.",
    "ttl": 48,
    "description": "the description",
  }
}
```

### DELETE /api/1.2/cdns/:name/federations/{:id}

Allow user to delete a federation.

Authentication Required: Yes

Role(s) Required: Admin

#### Request Route Parameters

Name	Type	Description
cdn	string	CDN name.
federation	string	Federation ID.

#### Response Properties

Parameter	Type	Description
alerts	array	A collection of alert messages.
>level	string	Success, info, warning or error.
>text	string	Alert message.
version	string	

#### Response Example

```
{
  "alerts": [
    {
      "level": "success",
      "text": "Federation deleted [ cname = the.cname. ]"
    }
  ],
}
```

## Federation Delivery Service

### /api/1.2/federations/:id/deliveryservices

#### GET /api/1.2/federations/:id/deliveryservices

Retrieves delivery services assigned to a federation.

Authentication Required: Yes

Role(s) Required: None

#### Request Route Parameters

Name	Type	Description
federation	string	Federation ID.

#### Response Properties

Parameter	Type	Description
id	int	
cdn	string	
type	string	
xmlId	string	

#### Response Example

```
{
  "response": [
    {
      "id": 41
      "cdn": "cdn1",
      "type": "DNS",
      "xmlId": "booya-12"
    }
  ]
}
```

#### POST /api/1.2/federations/:id/deliveryservices

Create one or more federation / delivery service assignments.

Authentication Required: Yes

Role(s) Required: Admin

#### Request Parameters

Name	Required	Description
dsIds	yes	An array of delivery service IDs.
replace	no	Replace existing fed/ds assignments? (true/false)



**Request Example**

```
{
  "dsIds": [ 2, 3, 4, 5, 6 ],
  "replace": true
}
```

**Response Properties**

Parameter	Type	Description
dsIds	array	An array of delivery service IDs.
replace	array	Existing fed/ds assignments replaced? (true/false).

**Response Example**

```
{
  "alerts": [
    {
      "level": "success",
      "text": "5 delivery service(s) were assigned to the_
↪cname. federation"
    }
  ],
  "response": {
    "dsIds" : [ 2, 3, 4, 5, 6 ],
    "replace" : true
  }
}
```

**DELETE /api/1.2/federations/:id/deliveryservices/:id**

Removes a delivery service from a federation.

Authentication Required: Yes

Role(s) Required: Admin

**Request Route Parameters**

Name	Required	Description
federation	yes	Federation ID.
ds	yes	Delivery Service ID.

**Response Example**

```
{
  "alerts": [
    {
      "level": "success",
      "text": "Removed delivery service [ booya-
↪12 ] from federation [ cname. ]"
    }
  ]
}
```

(continues on next page)

(continued from previous page)

```
}
  ],
}
```

## Federation Federation Resolver

### `/api/1.2/federations/:id/federation_resolvers`

#### GET `/api/1.2/federations/:id/federation_resolvers`

Retrieves federation resolvers assigned to a federation.

Authentication Required: Yes

Role(s) Required: None

#### Request Route Parameters

Name	Type	Description
<code>federation</code>	string	Federation ID.

#### Response Properties

Parameter	Type	Description
<code>id</code>	int	
<code>ipAddress</code>	string	
<code>type</code>	string	

#### Response Example

```
{
  "response": [
    {
      "id": 41
      "ipAddress": "2.2.2.2/16",
      "type": "RESOLVE4"
    }
  ]
}
```

#### POST `/api/1.2/federations/:id/federation_resolvers`

Create one or more federation / federation resolver assignments.

Authentication Required: Yes

Role(s) Required: Admin

### Request Parameters

Name	Required	Description
fedResolverIds	yes	An array of federation resolver IDs.
replace	no	Replace existing fed/ds assignments? (true/false)

### Request Example

```
{
  "fedResolverIds": [ 2, 3, 4, 5, 6 ],
  "replace": true
}
```

### Response Properties

Parameter	Type	Description
fedResolverIds	array	An array of federation resolver IDs.
replace	array	Existing fed/fed resolver assignments replaced? (true/false).

### Response Example

```
{
  "alerts": [
    {
      "level": "success",
      "text": "5 resolvers(s) were assigned to the cname.↵
↵federation"
    }
  ],
  "response": {
    "fedResolverIds" : [ 2, 3, 4, 5, 6 ],
    "replace" : true
  }
}
```

## Federation Resolver

### /api/1.2/federation\_resolvers

#### POST /api/1.2/federation\_resolvers

Create a federation resolver.

Authentication Required: Yes

Role(s) Required: ADMIN

#### Request Properties

Parameter	Required	Description
ipAddress	yes	IP or CIDR range
typeId	yes	Type Id where useintable=federation

### Request Example

```
{
  "ipAddress": "2.2.2.2/32",
  "typeId": 245
}
```

### Response Properties

Parameter	Type	Description
id	int	
ipAddress	string	
type	int	

### Response Example

```
{
  "alerts": [
    {
      "level": "success",
      "text": "Federation resolver created [ IP = 2.2.2.2/32, ↵
↵] with id: 27"
    }
  ],
  "response": {
    "id" : 27,
    "ipAddress" : "2.2.2.2/32",
    "typeId" : 245,
  }
}
```

## DELETE /api/1.2/federation\_resolvers/:id

Deletes a federation resolver.

Authentication Required: Yes

Role(s) Required: Admin

### Request Route Parameters

Name	Required	Description
resolver	yes	Federation resolver ID.

### Response Example

```
{
  "alerts": [
```

(continues on next page)

(continued from previous page)

```

        {
            "level": "success",
            "text": "Federation resolver deleted [ IP = ↪
↪2.2.2.2/32 ] with id: 27"
        },
    ]
}

```

## Federation User

### /api/1.2/federations/:id/users

#### GET /api/1.2/federations/:id/users

Retrieves users assigned to a federation.

Authentication Required: Yes

Role(s) Required: None

#### Request Route Parameters

Name	Type	Description
federation	string	Federation ID.

#### Response Properties

Parameter	Type	Description
company	string	
id	int	
username	string	
role	string	
email	string	
fullName	string	

#### Response Example

```

{
  "response": [
    {
      "id": 41
      "username": "booya",
      "company": "XYZ Corporation",
      "role": "federation",
      "email": "booya@fooya.com",
      "fullName": "Booya Fooya"
    }
  ]
}

```

**POST /api/1.2/federations/:id/users**

Create one or more federation / user assignments.

Authentication Required: Yes

Role(s) Required: Admin

**Request Parameters**

Name	Required	Description
userIds	yes	An array of user IDs.
replace	no	Replace existing fed/user assignments? (true/false)

**Request Example**

```
{
  "userIds": [ 2, 3, 4, 5, 6 ],
  "replace": true
}
```

**Response Properties**

Parameter	Type	Description
userIds	array	An array of user IDs.
replace	array	Existing fed/user assignments replaced? (true/false).

**Response Example**

```
{
  "alerts": [
    {
      "level": "success",
      "text": "5 user(s) were assigned to the cname.↵
↵ federation"
    }
  ],
  "response": {
    "userIds" : [ 2, 3, 4, 5, 6 ],
    "replace" : true
  }
}
```

**DELETE /api/1.2/federations/:id/users/:id**

Removes a user from a federation.

Authentication Required: Yes

Role(s) Required: Admin

## Request Route Parameters

Name	Required	Description
federation	yes	Federation ID.
user	yes	User ID.

## Response Example

```
{
  "alerts": [
    {
      "level": "success",
      "text": "Removed user [ bobmack ] from_
↪federation [ cnamel. ]"
    }
  ],
}
```

## Hardware Info

### /api/1.2/hwinfo

#### GET /api/1.2/hwinfo.json

Authentication Required: Yes

Role(s) Required: None

#### Response Properties

Parameter	Type	Description
serverId	string	Local unique identifier for this specific server's hardware info
serverHostName	string	Hostname for this specific server's hardware info
lastUpdated	string	The Time and Date for the last update for this server.
val	string	Freeform value used to track anything about a server's hardware info
description	string	Freeform description for this specific server's hardware info

## Response Example

```
{
  "response": [
    {
      "serverId": "odol-atsmid-cen-09",
      "lastUpdated": "2014-05-27 09:06:02",
      "val": "D1S4",
      "description": "Physical Disk 0:1:0"
    },
    {
      "serverId": "odol-atsmid-cen-09",
      "lastUpdated": "2014-05-27 09:06:02",
      "val": "D1S4",

```

(continues on next page)

(continued from previous page)

```
    "description": "Physical Disk 0:1:1"
  }
]
```

## ISO

### GET /api/1.2/osversions

Get all OS versions for ISO generation and the directory where the kickstarter files are found. The values are retrieved from `osversions.cfg` found in either `/var/www/files` or in the location defined by the `kickstart.files.location` parameter (if defined).

Authentication Required: Yes

Role(s) Required: None

#### Response Properties

Parameter	Description
OS version name	OS version name. For example, “CentOS 7.2 vda”.
OS version dir	The directory where the kickstarter ISO files are found. For example, <code>centos72-netinstall</code> .

#### Response Example

```
{
  "response":
  {
    "CentOS 7.2": "centos72-netinstall"
    "CentOS 7.2 vda": "centos72-netinstall-vda"
  }
}
```

### POST /api/1.2/isos

Generate an ISO.

Authentication Required: Yes

Role(s) Required: Operations

#### Request Properties



Parameter	Re-quired	Description
osversionDir	yes	The directory name where the kickstarter ISO files are found.
hostName	yes	
domainName	yes	
rootPass	yes	
dhcp	yes	Valid values are 'yes' or 'no'. If yes, other IP settings will be ignored.
interfaceMtu	yes	1500 or 9000
ipAddress	yes/no	Required if dhcp=no
ipNetmask	yes/no	Required if dhcp=no
ipGateway	yes/no	Required if dhcp=no
ip6Address	no	/64 is assumed if prefix is omitted.
ip6Gateway	no	Ignored if an IPV4 gateway is specified.
interfaceName	no	Typical values are bond0, eth4, etc. If you enter bond0, a LACP bonding config will be written.
disk	no	Typical values are "sda"

### Request Example

```
{
  "osversionDir": "centos72-netinstall-vda",
  "hostName": "foo-bar",
  "domainName": "baz.com",
  "rootPass": "password",
  "dhcp": "no",
  "interfaceMtu": 1500,
  "ipAddress": "10.10.10.10",
  "ipNetmask": "255.255.255.252",
  "ipGateway": "10.10.10.10"
}
```

### Response Properties

Parameter	Type	Description
isoURL	string	The URL location of the ISO. ISO locations can be found in cnd.conf file.

### Response Example

```
{
  "response": {
    "isoURL": "https://traffic_ops.domain.net/iso/fqdn-centos72-
↪netinstall.iso"
  },
  "alerts": [
    {
      "level": "success",
      "text": "Generate ISO was successful."
    }
  ]
}
```

## Jobs

### /api/1.2/jobs

#### GET /api/1.2/jobs

Get all jobs (currently limited to invalidate content (PURGE) jobs) sorted by start time (descending).

Authentication Required: Yes

Role(s) Required: Operations or Admin

#### Request Query Parameters

Name	Required	Description
dsId	no	Filter jobs by Delivery Service ID.
userId	no	Filter jobs by User ID.

#### Response Properties

Parameter	Type	Description
id	int	Job id
assetUrl	string	URL of the asset to invalidate.
deliveryService	string	Unique identifier of the job's DS.
keyword	string	Job keyword (PURGE)
parameters	string	Parameters associated with the job.
startTime	string	Start time of the job.
createdBy	string	Username that initiated the job.

#### Response Example

```
{
  "response": [
    {
      "id": 1
      "assetUrl": "http:\\\\foo-bar.domain.net\\taco.html",
      "deliveryService": "foo-bar",
      "keyword": "PURGE",
      "parameters": "TTL:48h",
      "startTime": "2015-05-14 08:56:36-06",
      "createdBy": "jdog24"
    },
    {
      "id": 2
      "assetUrl": "http:\\\\foo-bar.domain.net\\bell.html",
      "deliveryService": "foo-bar",
      "keyword": "PURGE",
      "parameters": "TTL:72h",
      "startTime": "2015-05-16 08:56:36-06",
      "createdBy": "jdog24"
    }
  ]
}
```

**GET /api/1.2/jobs/:id**

Get a job by ID (currently limited to invalidate content (PURGE) jobs).

Authentication Required: Yes

Role(s) Required: Operations or Admin

**Response Properties**

Parameter	Type	Description
id	int	Job id
assetUrl	string	URL of the asset to invalidate.
deliveryService	string	Unique identifier of the job's DS.
keyword	string	Job keyword (PURGE)
parameters	string	Parameters associated with the job.
startTime	string	Start time of the job.
createdBy	string	Username that initiated the job.

**Response Example**

```
{
  "response": [
    {
      "id": 1
      "assetUrl": "http:\\\\foo-bar.domain.net\\taco.html",
      "deliveryService": "foo-bar",
      "keyword": "PURGE",
      "parameters": "TTL:48h",
      "startTime": "2015-05-14 08:56:36-06",
      "createdBy": "jdog24"
    }
  ]
}
```

**Parameter****/api/1.2/parameters****GET /api/1.2/parameters**

Authentication Required: Yes

Role(s) Required: None

**Request Query Parameters**

Name	Required	Description
name	no	Filter parameters by name.
configFile	no	Filter parameters by config file.

### Response Properties

Parameter	Type	Description
lastUpdated	string	The Time / Date this server entry was last updated
secure	boolean	When true, the parameter is accessible only by admin users. Defaults to false.
value	string	The parameter value, only visible to admin if secure is true
name	string	The parameter name
configFile	string	The parameter config_file
profiles	array	An array of profiles attached to this parameter.

### Response Example

```
{
  "response": [
    {
      "lastUpdated": "2012-09-17 21:41:22",
      "secure": false,
      "value": "foo.bar.net",
      "name": "domain_name",
      "configFile": "FooConfig.xml",
      "profiles": [ "EDGE-FOO, MID-FOO" ]
    },
    {
      "lastUpdated": "2012-09-17 21:41:22",
      "secure": false,
      "value": "0,1,2,3,4,5,6",
      "name": "Drive_Letters",
      "configFile": "storage.config",
      "profiles": [ "EDGE-FOO, MID-FOO" ]
    },
    {
      "lastUpdated": "2012-09-17 21:41:22",
      "secure": true,
      "value": "STRING __HOSTNAME__",
      "name": "CONFIG proxy.config.proxy_name",
      "configFile": "records.config",
      "profiles": [ ]
    }
  ],
}
```

### GET /api/1.2/parameters/:id

Authentication Required: Yes

Role(s) Required: if secure of the parameter fetched is 1, require admin role, or any valid role can access.

### Response Properties

Parameter	Type	Description
id	integer	The parameter index
secure	boolean	When true, the parameter is accessible only by admin users. Defaults to false.
value	string	The parameter value, only visible to admin if secure is true
name	string	The parameter name
config_file	string	The parameter config_file

**Response Example**

```
{
  "response": [
    {
      "last_updated": "2012-09-17 21:41:22",
      "secure": 0,
      "value": "foo.bar.net",
      "name": "domain_name",
      "id": "27",
      "config_file": "FooConfig.xml",
    }
  ]
}
```

**GET /api/1.2/parameters/:id/profiles**

Retrieves all profiles assigned to the parameter.

Authentication Required: Yes

Role(s) Required: None

**Request Route Parameters**

Name	Required	Description
id	yes	Parameter ID.

**Response Properties**

Parameter	Type	Description
lastUpdated	string	The Time / Date this server entry was last updated
name	string	The name for the profile
id	string	Primary key
description	string	The description for the profile
type	string	The type for the profile

**Response Example**

```
{
  "response": [
    {
```

(continues on next page)

(continued from previous page)

```
    "lastUpdated": "2012-10-08 19:34:45",
    "name": "CCR_TOP",
    "id": "8",
    "description": "Content Router for top.foobar.net",
    "type": "ATS_PROFILE"
  }
]
```

### GET /api/1.2/parameters/:id/unassigned\_profiles

Retrieves all profiles NOT assigned to the parameter.

Authentication Required: Yes

Role(s) Required: None

#### Request Route Parameters

Name	Required	Description
id	yes	Parameter ID.

#### Response Properties

Parameter	Type	Description
lastUpdated	array	The Time / Date this server entry was last updated
name	string	The name for the profile
id	string	Primary key
description	string	The description for the profile

#### Response Example

```
{
  "response": [
    {
      "lastUpdated": "2012-10-08 19:34:45",
      "name": "CCR_TOP",
      "id": "8",
      "description": "Content Router for top.foobar.net"
    }
  ]
}
```

### GET /api/1.2/profiles/:id/parameters

Retrieves all parameters assigned to the profile.

Authentication Required: Yes

Role(s) Required: None

### Request Route Parameters

Name	Required	Description
id	yes	Profile id

### Response Properties

Parameter	Type	Description
last_updated	string	The Time / Date this server entry was last updated
secure	boolean	When true, the parameter is accessible only by admin users. Defaults to false.
value	string	The parameter value, only visible to admin if secure is true
name	string	The parameter name
config_file	string	The parameter config_file

### Response Example

```
{
  "response": [
    {
      "last_updated": "2012-09-17 21:41:22",
      "secure": false,
      "value": "foo.bar.net",
      "name": "domain_name",
      "config_file": "FooConfig.xml"
    },
    {
      "last_updated": "2012-09-17 21:41:22",
      "secure": false,
      "value": "0,1,2,3,4,5,6",
      "name": "Drive_Letters",
      "config_file": "storage.config"
    },
    {
      "last_updated": "2012-09-17 21:41:22",
      "secure": true,
      "value": "STRING __HOSTNAME__",
      "name": "CONFIG proxy.config.proxy_name",
      "config_file": "records.config"
    }
  ],
}
```

### GET /api/1.2/profiles/:id/unassigned\_parameters

Retrieves all parameters NOT assigned to the profile.

Authentication Required: Yes

Role(s) Required: None

#### Request Route Parameters

Name	Required	Description
id	yes	Profile id

#### Response Properties

Parameter	Type	Description
last_updated	string	The Time / Date this server entry was last updated
secure	boolean	When true, the parameter is accessible only by admin users. Defaults to false.
value	string	The parameter value, only visible to admin if secure is true
name	string	The parameter name
config_file	string	The parameter config_file

#### Response Example

```
{
  "response": [
    {
      "last_updated": "2012-09-17 21:41:22",
      "secure": false,
      "value": "foo.bar.net",
      "name": "domain_name",
      "config_file": "FooConfig.xml"
    },
    {
      "last_updated": "2012-09-17 21:41:22",
      "secure": false,
      "value": "0,1,2,3,4,5,6",
      "name": "Drive_Letters",
      "config_file": "storage.config"
    },
    {
      "last_updated": "2012-09-17 21:41:22",
      "secure": true,
      "value": "STRING __HOSTNAME__",
      "name": "CONFIG proxy.config.proxy_name",
      "config_file": "records.config"
    }
  ],
}
```

#### GET /api/1.2/profiles/name/:name/parameters

Authentication Required: Yes

Role(s) Required: None

#### Request Route Parameters



Name	Required	Description
name	yes	Profile name

### Response Properties

Parameter	Type	Description
last_updated	string	The Time / Date this server entry was last updated
secure	boolean	When true, the parameter is accessible only by admin users. Defaults to false.
value	string	The parameter value, only visible to admin if secure is true
name	string	The parameter name
config_file	string	The parameter config_file

### Response Example

```
{
  "response": [
    {
      "last_updated": "2012-09-17 21:41:22",
      "secure": false,
      "value": "foo.bar.net",
      "name": "domain_name",
      "config_file": "FooConfig.xml"
    },
    {
      "last_updated": "2012-09-17 21:41:22",
      "secure": false,
      "value": "0,1,2,3,4,5,6",
      "name": "Drive_Letters",
      "config_file": "storage.config"
    },
    {
      "last_updated": "2012-09-17 21:41:22",
      "secure": true,
      "value": "STRING __HOSTNAME__",
      "name": "CONFIG proxy.config.proxy_name",
      "config_file": "records.config"
    }
  ],
}
```

**POST /api/1.2/parameters** Create parameters.

Authentication Required: Yes

Role(s) Required: admin or oper

**Request Route Parameters** The request route parameters accept 2 formats, both single paramter and parameters array formats are acceptable.

single parameter format:

Name	Re-quired	Type	Description
name	yes	string	parameter name
configFile	yes	string	parameter config_file
value	yes	string	parameter value
secure	yes	integer	secure flag, when 1, the parameter is accessible only by admin users. Defaults to 0.

parameters array format:

Name	Re-quired	Type	Description
	yes	array	parameters array
>name	yes	string	parameter name
>configFile	yes	string	parameter config_file
>value	yes	string	parameter value
>secure	yes	integer	secure flag, when 1, the parameter is accessible only by admin users. Defaults to 0.

### Response Properties

Parameter	Type	Description
	array	parameters array
>id	integer	The parameter id
>name	string	parameter name
>configFile	string	parameter config_file
>value	string	parameter value
>secure	integer	secure flag, when 1, the parameter is accessible only by admin users. Defaults to 0.

### Request Example

1. single parameter format example:

```
{
  "name": "param1",
  "configFile": "configFile1",
  "value": "value1",
  "secure": 0
}
```

2. array format example:

```
[
  {
    "name": "param1",
    "configFile": "configFile1",
    "value": "value1",
    "secure": 0
  },
  {
    "name": "param2",
```

(continues on next page)

(continued from previous page)

```

        "configFile": "configFile2",
        "value": "value2",
        "secure": 1
    }
]

```

**Response Example**

```

{
  "response": [
    {
      "value": "value1",
      "secure": 0,
      "name": "param1",
      "id": "1139",
      "configFile": "configFile1"
    },
    {
      "value": "value2",
      "secure": 1,
      "name": "param2",
      "id": "1140",
      "configFile": "configFile2"
    }
  ]
}

```

**PUT /api/1.2/parameters/{:id}** Edit parameter.

Authentication Required: Yes

Role(s) Required: if the parameter's secure equals 1, only admin role can edit the parameter, or admin or oper role can access the API.

**Request Parameters**

Parameter	Type	Description
id	integer	The parameter id

**Request Route Parameters**

Name	Re-quired	Type	Description
name	no	string	parameter name
configFile	no	string	parameter config_file
value	no	string	parameter value
secure	no	integer	secure flag, when 1, the parameter is accessible only by admin users. Defaults to 0.

**Response Properties**

Parameter	Type	Description
id	integer	The parameter id
secure	integer	When 1, the parameter is accessible only by admin users. Defaults to 0.
value	string	The parameter value, only visible to admin if secure is true
name	string	The parameter name
config_file	string	The parameter config_file

**Request Example**

```
{
  "name": "param1",
  "configFile": "configFile1",
  "value": "value1",
  "secure": "0",
}
```

**Response Example**

```
{
  "response": {
    "value": "value1",
    "secure": "0",
    "name": "param1",
    "id": "1134",
    "configFile": "configFile1"
  }
}
```

**DELETE /api/1.2/parameters/{:id}** delete parameter. If the parameter have profile associated, can not be deleted.

Authentication Required: Yes

Role(s) Required: admin or oper role

**Request Parameters**

Parameter	Type	Description
id	integer	The parameter id

**No Request Route Parameters****Response Properties**

Parameter	Type	Description
alerts	array	A collection of alert messages.
>level	string	Success, info, warning or error.
>text	string	Alert message.
version	string	

**Response Example**

```
{
  "alerts":
  [
    {
      "level": "success",
      "text": "Parameter was successfully deleted."
    }
  ]
}
```

**POST /api/1.2/parameters/validate** Validate if the parameter exists.

Authentication Required: Yes

Role(s) Required: None

#### Request Route Parameters

Name	Required	Type	Description
name	yes	string	parameter name
configFile	yes	string	parameter config_file
value	yes	string	parameter value

#### Response Properties

Parameter	Type	Description
id	integer	The parameter id
secure	integer	When 1, the parameter is accessible only by admin users. Defaults to 0.
value	string	The parameter value, only visible to admin if secure is true
name	string	The parameter name
config_file	string	The parameter config_file

#### Request Example

```
{
  "name": "param1",
  "configFile": "configFile1",
  "value": "value1"
}
```

#### Response Example

```
{
  "response": {
    "value": "value1",
    "secure": "0",
    "name": "param1",
    "id": "1134",
    "configFile": "configFile1"
  }
}
```

## Physical Location

### /api/1.2/phys\_locations

#### GET /api/1.2/phys\_locations

Authentication Required: Yes

Role(s) Required: None

#### Request Query Parameters

Name	Required	Description
region	no	Filter by Region ID.

#### Response Properties

Parameter	Type	Description
address	string	
city	string	
comments	string	
email	string	
id	string	
lastUpdated	string	
name	string	
phone	string	
poc	string	
region	string	
regionId	string	
shortName	string	
state	string	
zip	string	

#### Response Example

```
{
  "response": [
    {
      "region": "Mile High",
      "region": "4",
      "poc": "Jane Doe",
      "lastUpdated": "2014-10-02 08:22:43",
      "name": "Albuquerque",
      "comments": "Albuquerque",
      "phone": "(123) 555-1111",
      "state": "NM",
      "email": "jane.doe@email.com",
      "city": "Albuquerque",
      "zip": "87107",
      "id": "2",
      "address": "123 East 3rd St",
```

(continues on next page)

(continued from previous page)

```
    "shortName": "Albuquerque"
  },
  {
    "region": "Mile High",
    "region": "4",
    "poc": "Jane Doe",
    "lastUpdated": "2014-10-02 08:22:43",
    "name": "Albuquerque",
    "comments": "Albuquerque",
    "phone": "(123) 555-1111",
    "state": "NM",
    "email": "jane.doe@email.com",
    "city": "Albuquerque",
    "zip": "87107",
    "id": "2",
    "address": "123 East 3rd St",
    "shortName": "Albuquerque"
  }
]
```

#### GET /api/1.2/phys\_locations/trimmed.json

Authentication Required: Yes

Role(s) Required: None

##### Response Properties

Parameter	Type	Description
name	string	

##### Response Example

```
{
  "response": [
    {
      "name": "Albuquerque"
    },
    {
      "name": "Ashburn"
    }
  ]
}
```

#### GET /api/1.2/phys\_locations/:id

Authentication Required: Yes

Role(s) Required: None

### Request Route Parameters

Name	Required	Description
id	yes	Physical location ID.

### Response Properties

Parameter	Type	Description
address	string	
city	string	
comments	string	
email	string	
id	string	
lastUpdated	string	
name	string	
phone	string	
poc	string	
region	string	
regionId	string	
shortName	string	
state	string	
zip	string	

### Response Example

```
{
  "response": [
    {
      "region": "Mile High",
      "region": "4",
      "poc": "Jane Doe",
      "lastUpdated": "2014-10-02 08:22:43",
      "name": "Albuquerque",
      "comments": "Albuquerque",
      "phone": "(123) 555-1111",
      "state": "NM",
      "email": "jane.doe@email.com",
      "city": "Albuquerque",
      "zip": "87107",
      "id": "2",
      "address": "123 East 3rd St",
      "shortName": "Albuquerque"
    }
  ]
}
```

**PUT /api/1.2/phys\_locations/:id** Update a physical location



Authentication Required: Yes

Role(s) Required: admin or oper

### Request Route Parameters

Name	Type	Description
id	int	Physical location id.

### Request Properties

Parameter	Required	Description
address	yes	Physical location address.
city	yes	Physical location city.
comments	no	Physical location comments.
email	no	Physical location email.
name	yes	Physical location name.
phone	no	Physical location phone.
poc	no	Physical location point of contact.
regionId	no	Physical location region ID.
shortName	yes	Physical location short name.
state	yes	Physical location state.
zip	yes	Physical location zip.

### Request Example

```
{
  "regionId": "1",
  "poc": "Jane Doesssss",
  "name": "Albuquerque",
  "comments": "Albuquerque",
  "phone": "(123) 555-1111",
  "state": "NM",
  "email": "jane.doe@email.com",
  "city": "Albuquerque",
  "zip": "87107",
  "address": "123 East 9rd St",
  "shortName": "Albuquerque"
}
```

### Response Properties

Parameter	Type	Description
address	string	
city	string	
comments	string	
email	string	
id	string	
lastUpdated	string	
name	string	
phone	string	
poc	string	
region	string	
regionId	string	
shortName	string	
state	string	
zip	string	

### Response Example

```
{
  "alerts": [
    {
      "level": "success",
      "text": "Physical location update was successful."
    }
  ],
  "response": [
    {
      "region": "Mile High",
      "region": "4",
      "poc": "Jane Doe",
      "lastUpdated": "2014-10-02 08:22:43",
      "name": "Albuquerque",
      "comments": "Albuquerque",
      "phone": "(123) 555-1111",
      "state": "NM",
      "email": "jane.doe@email.com",
      "city": "Albuquerque",
      "zip": "87107",
      "id": "2",
      "address": "123 East 3rd St",
      "shortName": "Albuquerque"
    }
  ]
}
```

**POST /api/1.2/regions/:region\_name/phys\_locations** Create physical location.

Authentication Required: Yes

Role(s) Required: admin or oper

region\_name: the name of the region to create physical location into.

### Request Route Parameters

Name	Required	Description
region_name	yes	The name of the physical location

### Request Properties

Parameter	Required	Description
name	yes	The name of the location
shortName	yes	The short name of the location
address	yes	
city	yes	
state	yes	
zip	yes	
phone	no	
poc	no	Point of contact
email	no	
comments	no	

### Request Example

```
{
  "name" : "my physical location1",
  "shortName" : "myphylocation1",
  "address" : "",
  "city" : "Shanghai",
  "state": "SH",
  "zip": "200000",
  "comments": "this is physical location1"
}
```

### Response Properties

Parameter	Type	Description
id	string	The id of the physical location created.
name	string	The name of the location
shortName	string	The short name of the location
regionName	string	The region name the physical location belongs to.
regionId	string	
address	string	
city	string	
state	string	
zip	string	
phone	string	
poc	string	Point of contact
email	string	
comments	string	

### Response Example

```
{
  "response": {
    'shortName': 'myphylocati',
    'regionName': 'myregion1',
    'name': 'my physical location1',
    'poc': '',
    'phone': '',
    'comments': 'this is physical location1',
    'state': 'SH',
    'email': '',
    'zip': '20000',
    'region_id': '20',
    'city': 'Shanghai',
    'address': '',
    'id': '200'
  }
}
```

## Profiles

### /api/1.2/profiles

#### GET /api/1.2/profiles

Authentication Required: Yes

Role(s) Required: None

#### Request Query Parameters

Name	Required	Description
param	no	Used to filter profiles by parameter ID.
cdn	no	Used to filter profiles by CDN ID.

#### Response Properties

Parameter	Type	Description
id	string	Primary key
name	string	The name for the profile
description	string	The description for the profile
cdn	int	The CDN ID
cdnName	string	The CDN name
type	string	Profile type
routingDisabled	bool	Traffic router routing disabled - defaults to false.
lastUpdated	array	The Time / Date this server entry was last updated

#### Response Example

```
{
  "response": [
    {
      "id": "8",
      "name": "EDGE_27_PROFILE",
      "description": "A profile with all the Foo parameters"
      "cdn": 1
      "cdnName": "cdn1"
      "type": "ATS_PROFILE"
      "routingDisabled": false
      "lastUpdated": "2012-10-08 19:34:45",
    }
  ]
}
```

### GET /api/1.2/profiles/trimmed

Authentication Required: Yes

Role(s) Required: None

#### Response Properties

Parameter	Type	Description
name	string	The name for the profile

#### Response Example

```
{
  "response": [
    {
      "name": "EDGE_27_PROFILE"
    }
  ]
}
```

### GET /api/1.2/profiles/:id

Authentication Required: Yes

Role(s) Required: None

#### Request Route Parameters

Parameter	Required	Description
id	yes	The ID of the profile.

#### Response Properties

Parameter	Type	Description
id	string	Primary key
name	string	The name for the profile
description	string	The description for the profile
cdn	int	The CDN ID
cdnName	string	The CDN name
type	string	Profile type
routingDisabled	bool	Traffic router routing disabled
lastUpdated	array	The Time / Date this server entry was last updated

### Response Example

```
{
  "response": [
    {
      "id": "8",
      "name": "EDGE_27_PROFILE",
      "description": "A profile with all the Foo parameters"
      "cdn": 1
      "cdnName": "cdn1"
      "type": "ATS_PROFILE"
      "routingDisabled": true
      "lastUpdated": "2012-10-08 19:34:45",
    }
  ]
}
```

### POST /api/1.2/profiles

Create a new empty profile.

Authentication Required: Yes

Role(s) Required: admin or oper

#### Request Properties

Parameter	Type	Re-quired	Description
name	string	yes	Profile name
description	string	yes	Profile description
cdn	int	no	CDN ID
type	string	yes	Profile type
routingDisabled	bool	no	Traffic router routing disabled. Defaults to false.

### Request Example

```
{
  "name": "EDGE_28_PROFILE",
  "description": "EDGE_28_PROFILE description",

```

(continues on next page)

(continued from previous page)

```
"cdn": 1,  
"type": "ATS_PROFILE",  
"routingDisabled": false  
}
```

### Response Properties

Parameter	Type	Description
id	string	Profile ID
name	string	Profile name
description	string	Profile description
cdn	int	CDN ID
type	string	Profile type
routingDisabled	bool	Traffic router routing disabled

### Response Example

```
{  
  "response": [  
    {  
      "id": "66",  
      "name": "EDGE_28_PROFILE",  
      "description": "EDGE_28_PROFILE description",  
      "cdn": 1,  
      "type": "ATS_PROFILE",  
      "routingDisabled": false  
    }  
  ]  
}
```

### POST /api/1.2/profiles/name/:profile\_name/copy/:profile\_copy\_from

Copy profile to a new profile. The new profile name must not exist.

Authentication Required: Yes

Role(s) Required: admin or oper

#### Request Route Parameters

Name	Required	Description
profile_name	yes	The name of profile to copy
profile_copy_from	yes	The name of profile copy from

### Response Properties

Parameter	Type	Description
id	string	Id of the new profile
name	string	The name of the new profile
profileCopyFrom	string	The name of profile to copy
idCopyFrom	string	The id of profile to copy
description	string	new profile's description (copied)

### Response Example

```
{
  "response": [
    {
      "id": "66",
      "name": "CCR_COPY",
      "profileCopyFrom": "CCR1",
      "description": "CCR_COPY description",
      "idCopyFrom": "3"
    }
  ]
}
```

### PUT /api/1.2/profiles/{:id}

Allows user to edit a profile.

Authentication Required: Yes

Role(s) Required: admin or oper

#### Request Route Parameters

Name	Required	Description
id	yes	profile id.

#### Request Properties

Parameter	Type	Re-quired	Description
name	string	yes	Profile name
description	string	yes	Profile description
cdn	int	no	CDN ID - must use the same ID as any servers assigned to the profile.
type	string	yes	Profile type
routingDisabled	bool	no	Traffic router routing disabled. When not present, value defaults to false.

### Request Example



```
{
  "name": "EDGE_28_PROFILE",
  "description": "EDGE_28_PROFILE description",
  "cdn": 1,
  "type": "ATS_PROFILE",
  "routingDisabled": false
}
```

### Response Properties

Parameter	Type	Description
id	string	Profile ID
name	string	Profile name
description	string	Profile description
cdn	int	CDN ID
type	string	Profile type
routingDisabled	bool	Traffic router routing disabled

### Response Example

```
{
  "response":{
    "id": "219",
    "name": "EDGE_28_PROFILE",
    "description": "EDGE_28_PROFILE description"
    "cdn": 1
    "type": "ATS_PROFILE",
    "routingDisabled": false
  }
  "alerts":[
    {
      "level": "success",
      "text": "Profile was updated: 219"
    }
  ]
}
```

### DELETE /api/1.2/profiles/{:id}

Allows user to delete a profile.

Authentication Required: Yes

Role(s) Required: admin or oper

#### Request Route Parameters

Name	Required	Description
id	yes	profile id.

### Response Properties

Parameter	Type	Description
alerts	array	A collection of alert messages.
>level	string	success, info, warning or error.
>text	string	Alert message.
version	string	

### Response Example

```
{
  "alerts": [
    {
      "level": "success",
      "text": "Profile was deleted."
    }
  ]
}
```

## Profile parameters

### /api/1.2/profileparameters

#### POST /api/1.2/profileparameters

Associate parameter to profile.

Authentication Required: Yes

Role(s) Required: admin or oper

**Request Properties** This accept two formats: single profile-parameter, profile-parameter array.

Single profile-parameter format:

Parameter	Required	Description
profileId	yes	profile id.
parameterId	yes	parameter id.

Profile-parameter array format:

Parameter	Required	Description
	yes	profile-parameter array.
>profileId	yes	profile id.
>parameterId	yes	parameter id.

**Request Example**

Single profile-parameter format:

```
{
  "profileId": 2,
  "parameterId": 6
}
```

Profile-parameter array format:

```
[
  {
    "profileId": 2,
    "parameterId": 6
  },
  {
    "profileId": 2,
    "parameterId": 7
  },
  {
    "profileId": 3,
    "parameterId": 6
  }
]
```

**\*\*Response Properties\*\***

+-----+-----+-----+-----+			
+-----+			
Parameter	Type	Description	
+-----+-----+-----+-----+			
``response``	array	Profile-parameter associations.	
+-----+-----+-----+-----+			
>profileId``	string	Profile id.	
+-----+-----+-----+-----+			
>parameterId``	string	Parameter id.	
+-----+-----+-----+-----+			
``alerts``	array	A collection of alert messages.	
+-----+-----+-----+-----+			
>level``	string	success, info, warning or error.	
+-----+-----+-----+-----+			
>text``	string	Alert message.	
+-----+-----+-----+-----+			

(continues on next page)

(continued from previous page)

``version``	string	
↪		↪
+-----+	+-----+	+-----+
↪-----+		

**Response Example**

```
{
  "response": [
    {
      "profileId": "2",
      "parameterId": "6"
    },
    {
      "profileId": "2",
      "parameterId": "7"
    },
    {
      "profileId": "3",
      "parameterId": "6"
    }
  ]
  "alerts": [
    {
      "level": "success",
      "text": "Profile parameter associations were created."
    }
  ]
}
```

**DELETE /api/1.2/profileparameters/{:profile\_id}/{:parameter\_id}**

Delete a profile parameter association.

Authentication Required: Yes

Role(s) Required: admin or oper

**Request Route Parameters**

Name	Required	Description
profile_id	yes	profile id.
parameter_id	yes	parameter id.

**Response Properties**

Parameter	Type	Description
alerts	array	A collection of alert messages.
>level	string	success, info, warning or error.
>text	string	Alert message.
version	string	

**Response Example**

```
{
  "alerts": [
    {
      "level": "success",
      "text": "Profile parameter association was deleted."
    }
  ]
}
```

**POST /api/1.2/profiles/name/{:name}/parameters**

Associate parameters to a profile. If the parameter does not exist, create it and associate to the profile. If the parameter already exists, associate it to the profile. If the parameter already associate the profile, keep the association. If the profile does not exist, the API returns fail.

Authentication Required: Yes

Role(s) Required: admin or oper. If there is parameter's secure equals 1 in the request properties, need admin role.

**Request Route Parameters**

Name	Required	Description
name	yes	profile name.

**Request Properties** The request properties accept 2 formats, both single paramter and parameters array formats are acceptable.

single parameter format:

Name	Re-quired	Type	Description
name	yes	string	parameter name
configFile	yes	string	parameter config_file
value	yes	string	parameter value
secure	yes	integer	secure flag, when 1, the parameter is accessible only by admin users. Defaults to 0.

array parameters format:

Name	Re-quired	Type	Description
	yes	array	parameters array
>name	yes	string	parameter name
>configFile	yes	string	parameter config_file
>value	yes	string	parameter value
>secure	yes	integer	secure flag, when 1, the parameter is accessible only by admin users. Defaults to 0.

### Request Example

```

1. single parameter format example:
{
  "name":"param1",
  "configFile":"configFile1",
  "value":"value1",
  "secure":0,
}

2. array format example:
[
  {
    "name":"param1",
    "configFile":"configFile1",
    "value":"value1",
    "secure":0,
  },
  {
    "name":"param2",
    "configFile":"configFile2",
    "value":"value2",
    "secure":1,
  }
]

```

### Response Properties

Name	Type	Description
``response``		Parameters associated with the profile.
>profileName``	string	profile name
>profileId``	integer	profile index
>parameters``	array	parameters array
>>id``	integer	parameter index
>>name``	string	parameter name
>>configFile``	string	parameter config_file

(continues on next page)

(continued from previous page)

+-----+-----+-----+			
↪	+-----+		
``>>value``	string	parameter value	└
↪			
+-----+-----+-----+			
↪	+-----+		
``>>secure``	integer	secure flag, when 1, the parameter is	└
↪	accessible only by admin users. Defaults to 0.		
+-----+-----+-----+			
↪	+-----+		
``alerts``	array	A collection of alert messages.	└
↪			
+-----+-----+-----+			
↪	+-----+		
``>level``	string	success, info, warning or error.	└
↪			
+-----+-----+-----+			
↪	+-----+		
``>text``	string	Alert message.	└
↪			
+-----+-----+-----+			
↪	+-----+		
``version``	string		└
↪			
+-----+-----+-----+			
↪	+-----+		

### Response Example

```
{
  "response":{
    "profileName": "CCR1",
    "profileId" : "12",
    "parameters":[
      {
        "name":"param1",
        "configFile":"configFile1"
        "value":"value1",
        "secure":"0",
      },
      {
        "name":"param2",
        "configFile":"configFile2"
        "value":"value2",
        "secure":"1",
      }
    ]
  }
  "alerts":[
    {
      "level": "success",
      "text": "Assign parameters successfully to profile CCR1"
    }
  ]
}
```

**POST /api/1.2/profiles/id/{:id}/parameters**

Associate parameters to a profile. If the parameter does not exist, create it and associate to the profile. If the parameter already exists, associate it to the profile. If the parameter already associate the profile, keep the association. If the profile does not exist, the API returns fail.

Authentication Required: Yes

Role(s) Required: admin or oper. If there is parameter's secure equals 1 in the request properties, need admin role.

**Request Route Parameters**

Name	Required	Description
id	yes	profile name.

**Request Properties** The request properties accept 2 formats, both single paramter and parameters array formats are acceptable.

single parameter format:

Name	Re-quired	Type	Description
name	yes	string	parameter name
configFile	yes	string	parameter config_file
value	yes	string	parameter value
secure	yes	integer	secure flag, when 1, the parameter is accessible only by admin users. Defaults to 0.

array parameters format:

Name	Re-quired	Type	Description
	yes	array	parameters array
>name	yes	string	parameter name
>configFile	yes	string	parameter config_file
>value	yes	string	parameter value
>secure	yes	integer	secure flag, when 1, the parameter is accessible only by admin users. Defaults to 0.

**Request Example**

```
1. single parameter format exampe:
{
  "name": "param1",
  "configFile": "configFile1",
  "value": "value1",
  "secure": 0,
}

2. array format example:
```

(continues on next page)



(continued from previous page)

```
[
  {
    "name": "param1",
    "configFile": "configFile1",
    "value": "value1",
    "secure": 0,
  },
  {
    "name": "param2",
    "configFile": "configFile2",
    "value": "value2",
    "secure": 1,
  }
]
```

Response Properties

+-----+-----+-----+			
↪-----+-----+-----+			
Name	Type	Description	└
↪-----+-----+-----+			
``response``		Parameters associated with the profile.	└
↪-----+-----+-----+			
↪-----+-----+-----+			
``>profileName``	string	profile name	└
↪-----+-----+-----+			
↪-----+-----+-----+			
``>profileId``	integer	profile index	└
↪-----+-----+-----+			
↪-----+-----+-----+			
``>parameters``	array	parameters array	└
↪-----+-----+-----+			
↪-----+-----+-----+			
``>>id``	integer	parameter index	└
↪-----+-----+-----+			
↪-----+-----+-----+			
``>>name``	string	parameter name	└
↪-----+-----+-----+			
↪-----+-----+-----+			
``>>configFile``	string	parameter config_file	└
↪-----+-----+-----+			
↪-----+-----+-----+			
``>>value``	string	parameter value	└
↪-----+-----+-----+			
↪-----+-----+-----+			
``>>secure``	integer	secure flag, when 1, the parameter is	└
↪-----+-----+-----+		accessible only by admin users. Defaults to 0.	
↪-----+-----+-----+			
↪-----+-----+-----+			

(continues on next page)

(continued from previous page)

``alerts``	array	A collection of alert messages.	
↪			
+-----+	+-----+	+-----+	+-----+
↪			
``>level``	string	success, info, warning or error.	
↪			
+-----+	+-----+	+-----+	+-----+
↪			
``>text``	string	Alert message.	
↪			
+-----+	+-----+	+-----+	+-----+
↪			
``version``	string		
↪			
+-----+	+-----+	+-----+	+-----+
↪			

**Response Example**

```
{
  "response":{
    "profileName": "CCR1",
    "profileId" : "12",
    "parameters":[
      {
        "name":"param1",
        "configFile":"configFile1"
        "value":"value1",
        "secure":"0",
      },
      {
        "name":"param2",
        "configFile":"configFile2"
        "value":"value2",
        "secure":"1",
      }
    ]
  }
  "alerts":[
    {
      "level": "success",
      "text": "Assign parameters successfully to profile CCR1"
    }
  ]
}
```

**POST /api/1.2/profileparameter**

Create one or more profile / parameter assignments.

Authentication Required: Yes

Role(s) Required: Admin or Operations

### Request Parameters

Name	Required	Description
profileId	yes	The ID of the profile.
paramIds	yes	An array of parameter IDs.
replace	no	Replace existing profile/param assignments? (true/false)

### Request Example

```
{
  "profileId": 22,
  "paramIds": [ 2, 3, 4, 5, 6 ],
  "replace": true
}
```

### Response Properties

Parameter	Type	Description
profileId	int	The ID of the profile.
paramIds	array	An array of parameter IDs.
replace	bool	Existing profile/param assignments replaced? (true/false).

### Response Example

```
{
  "alerts": [
    {
      "level": "success",
      "text": "14 parameters where assigned to the foo_
↪profile."
    }
  ],
  "response": {
    "profileId" : 22,
    "paramIds" : [ 2, 3, 4, 5, 6 ],
    "replace" : true
  }
}
```

## POST /api/1.2/parameterprofile

Create one or more parameter / profile assignments.

Authentication Required: Yes

Role(s) Required: Admin or Operations

### Request Parameters

Name	Required	Description
paramId	yes	The ID of the parameter.
profileIds	yes	An array of profile IDs.
replace	no	Replace existing param/profile assignments? (true/false)

### Request Example

```
{
  "paramId": 22,
  "profileIds": [ 2, 3, 4, 5, 6 ],
  "replace": true
}
```

### Response Properties

Parameter	Type	Description
paramId	int	The ID of the parameter.
profileIds	array	An array of profile IDs.
replace	bool	Existing param/profile assignments replaced? (true/false).

### Response Example

```
{
  "alerts": [
    {
      "level": "success",
      "text": "14 profiles were assigned to the bar_
↪parameter."
    }
  ],
  "response": {
    "paramId" : 22,
    "profileIds" : [ 2, 3, 4, 5, 6 ],
    "replace" : true
  }
}
```

## InfluxDB

---

**Note:** The documentation needs a thorough review!

---

### GET /api/1.2/traffic\_monitor/stats.json

Authentication Required: Yes

Role(s) Required: None

### Response Properties

Parameter	Type	Description
aaData	array	

**Response Example**

```
{
  "aaData": [
    [
      "0",
      "ALL",
      "ALL",
      "ALL",
      "true",
      "ALL",
      "142035",
      "172365661.85"
    ],
    [
      1,
      "EDGE1_TOP_421_PSPP",
      "odol-atsec-atl-03",
      "us-ga-atlanta",
      "1",
      "REPORTED",
      "596",
      "923510.04",
      "69.241.82.126"
    ]
  ],
}
```

**Regions****/api/1.2/regions****GET /api/1.1/regions**

Authentication Required: Yes

Role(s) Required: None

**Request Route Parameters**

Name	Required	Description
division	no	Filter regions by Division ID.

**Response Properties**

Parameter	Type	Description
id	string	Region ID.
name	string	Region name.
division	string	Division ID.
divisionName	string	Division name.

**Response Example**

```
{
  "response": [
    {
      "id": "6",
      "name": "Atlanta",
      "division": "2",
      "divisionName": "West"
    },
    {
      "id": "7",
      "name": "Denver",
      "division": "2",
      "divisionName": "West"
    }
  ]
}
```

**GET /api/1.1/regions/:id**

Authentication Required: Yes

Role(s) Required: None

**Request Route Parameters**

Name	Required	Description
id	yes	Region id.

**Response Properties**

Parameter	Type	Description
id	string	Region ID.
name	string	Region name.
division	string	Division ID.
divisionName	string	Division name.

**Response Example**

```
{
  "response": [
    {
      "id": "6",
      "name": "Atlanta",
      "division": "2",
      "divisionName": "West"
    }
  ]
}
```

**PUT /api/1.2/regions/:id** Update a region

Authentication Required: Yes

Role(s) Required: admin or oper

**Request Route Parameters**

Name	Type	Description
id	int	Region id.

### Request Properties

Parameter	Required	Description
name	yes	The name of the region
division	yes	The division Id

### Request Example

```
{
  "name": "myregion1",
  "division": "4"
}
```

### Response Properties

Parameter	Type	Description
division	string	
divisionName	string	
name	string	
id	string	
lastUpdated	string	

### Response Example

```
{
  "alerts": [
    {
      "level": "success",
      "text": "Region update was successful."
    }
  ],
  "response": {
    "id": "1",
    "lastUpdated": "2014-03-18 08:57:39",
    "name": "myregion1",
    "division": "4",
    "divisionName": "mydivision1"
  }
}
```

**POST /api/1.2/divisions/:division\_name/regions** Create Region

Authentication Required: Yes

Role(s) Required: admin or oper

division\_name - The name of division to create new region into.

**\*\* Request Route Parameters\*\***

Name	Required	Description
division_name	yes	The name of division will create new region in

### Request Properties

Parameter	Required	Description
name	yes	The name of the region

### Request Example

```
{
  "name": "myregion1",
}
```

### Response Properties

Parameter	Type	Description
name	string	name of region created
id	string	id of region created
divisionName	string	the division name the region belongs to.
divisionId	string	the id of division the region belongs to.

### Response Example

```
{
  "response": {
    'divisionName': 'mydivision1',
    'divsionId': '4',
    'name': 'myregion1',
    'id': '19'
  }
}
```

## Roles

**/api/1.2/roles**

**GET /api/1.2/roles.json**



Authentication Required: Yes

Role(s) Required: None

#### Response Properties

Parameter	Type	Description
name	string	
id	string	
privLevel	string	
description	string	

#### Response Example

```
{
  "response": [
    {
      "name": "read-only",
      "id": "2",
      "privLevel": "10",
      "description": "read-only user"
    }
  ]
}
```

## Server

### /api/1.2/servers

#### GET /api/1.2/servers

Retrieves properties of CDN servers.

Authentication Required: Yes

Role(s) Required: None

#### Request Query Parameters

Name	Required	Description
dsId	no	Used to filter servers by delivery service.
status	no	Used to filter servers by status.
type	no	Used to filter servers by type.
profileId	no	Used to filter servers by profile ID.
cdn	no	Used to filter servers by CDN ID.
cachegroup	no	Used to filter servers by cache group ID.
physLocation	no	Used to filter servers by phys location ID.

#### Response Properties

Parameter	Type	Description
cachegroup	string	The cache group name (see <i>Cache Group</i> ).
cachegroupId	string	The cache group id.
cdnId	string	Id of the CDN to which the server belongs to.
cdnName	string	Name of the CDN to which the server belongs to.
domainName	string	The domain name part of the FQDN of the cache.
guid	string	An identifier used to uniquely identify the server.
hostName	string	The host name part of the cache.
httpsPort	string	The HTTPS port on which the main application listens (443 in most cases).
id	string	The server id (database row number).
iloIpAddress	string	The IPv4 address of the lights-out-management port.
iloIpGateway	string	The IPv4 gateway address of the lights-out-management port.
iloIpNetmask	string	The IPv4 netmask of the lights-out-management port.
iloPassword	string	The password of the of the lights-out-management user (displays as ** unless you are an 'admin' user).
iloUsername	string	The user name for lights-out-management.
interfaceMtu	string	The Maximum Transmission Unit (MTU) to configure for interfaceName.
interfaceName	string	The network interface name used for serving traffic.
ip6Address	string	The IPv6 address/netmask for interfaceName.
ip6Gateway	string	The IPv6 gateway for interfaceName.
ipAddress	string	The IPv4 address for interfaceName.
ipGateway	string	The IPv4 gateway for interfaceName.
ipNetmask	string	The IPv4 netmask for interfaceName.
lastUpdated	string	The Time and Date for the last update for this server.
mgmtIpAddress	string	The IPv4 address of the management port (optional).
mgmtIpGateway	string	The IPv4 gateway of the management port (optional).
mgmtIpNetmask	string	The IPv4 netmask of the management port (optional).
offlineReason	string	A user-entered reason why the server is in ADMIN_DOWN or OFFLINE status.
physLocation	string	The physical location name (see <i>Physical Location</i> ).
physLocationId	string	The physical location id (see <i>Physical Location</i> ).
profile	string	The assigned profile name (see <i>Profiles</i> ).
profileDesc	string	The assigned profile description (see <i>Profiles</i> ).
profileId	string	The assigned profile Id (see <i>Profiles</i> ).
rack	string	A string indicating rack location.
routerHostName	string	The human readable name of the router.
routerPortName	string	The human readable name of the router port.
status	string	The Status string (See <i>Status</i> ).
statusId	string	The Status id (See <i>Status</i> ).
tcpPort	string	The default TCP port on which the main application listens (80 for a cache in most cases).
type	string	The name of the type of this server (see <i>Types</i> ).
typeId	string	The id of the type of this server (see <i>Types</i> ).
updPending	bool	

### Response Example

```
{
  "response": [
    {
      "cachegroup": "us-il-chicago",
      "cachegroupId": "3",
      "cdnId": "3",
      "cdnName": "CDN-1",
      "domainName": "chi.kabletown.net",
      "guid": null,
```

(continues on next page)

(continued from previous page)

```

    "hostName": "atsec-chi-00",
    "id": "19",
    "iloIpAddress": "172.16.2.6",
    "iloIpGateway": "172.16.2.1",
    "iloIpNetmask": "255.255.255.0",
    "iloPassword": "*****",
    "iloUsername": "",
    "interfaceMtu": "9000",
    "interfaceName": "bond0",
    "ip6Address": "2033:D0D0:3300::2:2/64",
    "ip6Gateway": "2033:D0D0:3300::2:1",
    "ipAddress": "10.10.2.2",
    "ipGateway": "10.10.2.1",
    "ipNetmask": "255.255.255.0",
    "lastUpdated": "2015-03-08 15:57:32",
    "mgmtIpAddress": "",
    "mgmtIpGateway": "",
    "mgmtIpNetmask": "",
    "offlineReason": "N/A",
    "physLocation": "plocation-chi-1",
    "physLocationId": "9",
    "profile": "EDGE1_CDN1_421_SSL",
    "profileDesc": "EDGE1_CDN1_421_SSL profile",
    "profileId": "12",
    "rack": "RR 119.02",
    "routerHostName": "rtr-chi.kabletown.net",
    "routerPortName": "2",
    "status": "ONLINE",
    "statusId": "6",
    "tcpPort": "80",
    "httpsPort": "443",
    "type": "EDGE",
    "typeId": "3",
    "updPending": false
  },
  {
    ... more server data
  }
]
}

```

**GET /api/1.2/servers/:id**

Retrieves properties of a CDN server by server ID.

Authentication Required: Yes

Role(s) Required: None

**Request Route Parameters**

Name	Required	Description
id	yes	Server id.

### Response Properties

Parameter	Type	Description
cachegroup	string	The cache group name (see <a href="#">Cache Group</a> ).
cachegroupId	string	The cache group id.
cdnId	string	Id of the CDN to which the server belongs to.
cdnName	string	Name of the CDN to which the server belongs to.
domainName	string	The domain name part of the FQDN of the cache.
guid	string	An identifier used to uniquely identify the server.
hostName	string	The host name part of the cache.
httpsPort	string	The HTTPS port on which the main application listens (443 in most cases).
id	string	The server id (database row number).
iloIpAddress	string	The IPv4 address of the lights-out-management port.
iloIpGateway	string	The IPv4 gateway address of the lights-out-management port.
iloIpNetmask	string	The IPv4 netmask of the lights-out-management port.
iloPassword	string	The password of the of the lights-out-management user (displays as ** unless you are an ‘admin’ user).
iloUsername	string	The user name for lights-out-management.
interfaceMtu	string	The Maximum Transmission Unit (MTU) to configure for interfaceName.
interfaceName	string	The network interface name used for serving traffic.
ip6Address	string	The IPv6 address/netmask for interfaceName.
ip6Gateway	string	The IPv6 gateway for interfaceName.
ipAddress	string	The IPv4 address for interfaceName.
ipGateway	string	The IPv4 gateway for interfaceName.
ipNetmask	string	The IPv4 netmask for interfaceName.
lastUpdated	string	The Time and Date for the last update for this server.
mgmtIpAddress	string	The IPv4 address of the management port (optional).
mgmtIpGateway	string	The IPv4 gateway of the management port (optional).
mgmtIpNetmask	string	The IPv4 netmask of the management port (optional).
offlineReason	string	A user-entered reason why the server is in ADMIN_DOWN or OFFLINE status.
physLocation	string	The physical location name (see <a href="#">Physical Location</a> ).
physLocationId	string	The physical location id (see <a href="#">Physical Location</a> ).
profile	string	The assigned profile name (see <a href="#">Profiles</a> ).
profileDesc	string	The assigned profile description (see <a href="#">Profiles</a> ).
profileId	string	The assigned profile Id (see <a href="#">Profiles</a> ).
rack	string	A string indicating rack location.
routerHostName	string	The human readable name of the router.
routerPortName	string	The human readable name of the router port.
status	string	The Status string (See <a href="#">Status</a> ).
statusId	string	The Status id (See <a href="#">Status</a> ).
tcpPort	string	The default TCP port on which the main application listens (80 for a cache in most cases).
type	string	The name of the type of this server (see <a href="#">Types</a> ).
typeId	string	The id of the type of this server (see <a href="#">Types</a> ).
updPending	bool	

### Response Example

```
{
  "response": [
    {
      "cachegroup": "us-il-chicago",
      "cachegroupId": "3",
      "cdnId": "3",
```

(continues on next page)

(continued from previous page)

```

        "cdnName": "CDN-1",
        "domainName": "chi.kabletown.net",
        "guid": null,
        "hostName": "atsec-chi-00",
        "id": "19",
        "iloIpAddress": "172.16.2.6",
        "iloIpGateway": "172.16.2.1",
        "iloIpNetmask": "255.255.255.0",
        "iloPassword": "*****",
        "iloUsername": "",
        "interfaceMtu": "9000",
        "interfaceName": "bond0",
        "ip6Address": "2033:D0D0:3300::2:2/64",
        "ip6Gateway": "2033:D0D0:3300::2:1",
        "ipAddress": "10.10.2.2",
        "ipGateway": "10.10.2.1",
        "ipNetmask": "255.255.255.0",
        "lastUpdated": "2015-03-08 15:57:32",
        "mgmtIpAddress": "",
        "mgmtIpGateway": "",
        "mgmtIpNetmask": "",
        "offlineReason": "N/A",
        "physLocation": "plocation-chi-1",
        "physLocationId": "9",
        "profile": "EDGE1_CDN1_421_SSL",
        "profileDesc": "EDGE1_CDN1_421_SSL profile",
        "profileId": "12",
        "rack": "RR 119.02",
        "routerHostName": "rtr-chi.kabletown.net",
        "routerPortName": "2",
        "status": "ONLINE",
        "statusId": "6",
        "tcpPort": "80",
        "httpsPort": "443",
        "type": "EDGE",
        "typeId": "3",
        "updPending": false
    }
]
}

```

**GET /api/1.2/servers/:id/deliveryservices**

Retrieves all delivery services assigned to the server. See also [Using Traffic Ops - Delivery Service](#).

Authentication Required: Yes

Role(s) Required: None

**Request Route Parameters**

Name	Required	Description
id	yes	Server ID.

**Response Properties**

Parameter	Type	Description
active	bool	true if active, false if inactive.
cacheurl	string	Cache URL rule to apply to this delivery service.
ccrDnsTtl	string	The TTL of the DNS response for A or AAAA queries requesting the IP address of the tr. host.
cdnId	string	Id of the CDN to which the delivery service belongs to.
cdnName	string	Name of the CDN to which the delivery service belongs to.
checkPath	string	The path portion of the URL to check this deliveryservice for health.
deepCachingType	string	When to do Deep Caching for this Delivery Service: <ul style="list-style-type: none"><li>• NEVER (default)</li><li>• ALWAYS</li></ul>
displayName	string	The display name of the delivery service.
dnsBypassIp	string	The IPv4 IP to use for bypass on a DNS deliveryservice - bypass starts when serving more than the globalMaxMbps traffic on this deliveryservice.
dnsBypassIp6	string	The IPv6 IP to use for bypass on a DNS deliveryservice - bypass starts when serving more than the globalMaxMbps traffic on this deliveryservice.
dnsBypassTtl	string	The TTL of the DNS bypass response.
dscp	string	The Differentiated Services Code Point (DSCP) with which to mark downstream (EDGE -> customer) traffic.
edgeHeaderRewrite	string	The EDGE header rewrite actions to perform.
geoLimitRedirectUrl	string	

Continued on next page

Table 24 – continued from previous page

Parameter	Type	Description
geoLimit	string	<ul style="list-style-type: none"> <li>• 0: None - no limitations</li> <li>• 1: Only route on CZF file hit</li> <li>• 2: Only route on CZF hit or when from USA</li> </ul> <p>Note that this does not prevent access to content or makes content secure; it just prevents routing to the content by Traffic Router.</p>
geoLimitCountries	string	
geoProvider	string	
globalMaxMbps	string	The maximum global bandwidth allowed on this deliveryservice. If exceeded, the traffic routes to the dnsByPassIp* for DNS deliveryservices and to the httpBypassFqdn for HTTP deliveryservices.
globalMaxTps	string	The maximum global transactions per second allowed on this deliveryservice. When this is exceeded traffic will be sent to the dnsByPassIp* for DNS deliveryservices and to the httpBypassFqdn for HTTP deliveryservices
httpBypassFqdn	string	The HTTP destination to use for bypass on an HTTP deliveryservice - bypass starts when serving more than the globalMaxMbps traffic on this deliveryservice.
id	string	The deliveryservice id (database row number).
infoUrl	string	Use this to add a URL that points to more information about that deliveryservice.
initialDispersion	string	
ipv6RoutingEnabled	bool	false: send IPv4 address of Traffic Router to client on HTTP type del.
lastUpdated	string	
logsEnabled	bool	
longDesc	string	Description field 1.
longDesc1	string	Description field 2.
longDesc2	string	Description field 2.

Continued on next page

Table 24 – continued from previous page

Parameter	Type	Description
>>type	string	The type of MatchList (one of :ref:to-api-v11-types use_in_table='regex').
>>setNumber	string	The set Number of the match-List.
>>pattern	string	The regexp for the matchList.
maxDnsAnswers	string	The maximum number of IPs to put in a A/AAAA response for a DNS deliveryservice (0 means all available).
midHeaderRewrite	string	The MID header rewrite actions to perform.
missLat	string	The latitude to use when the client cannot be found in the CZF or the Geo lookup.
missLong	string	The longitude to use when the client cannot be found in the CZF or the Geo lookup.
multiSiteOrigin	bool	Is the Multi Site Origin feature enabled for this delivery service (0=false, 1=true). See <a href="#">Multi Site Origin</a>
multiSiteOriginAlgor	bool	Is the Multi Site Origin feature enabled for this delivery service (0=false, 1=true). See <a href="#">Multi Site Origin</a>
orgServerFqdn	string	The origin server base URL (FQDN when used in this instance, includes the protocol ( <a href="#">http://</a> or <a href="#">https://</a> ) for use in retrieving content from the origin server.
originShield	string	
profileDescription	string	The description of the Traffic Router Profile with which this deliveryservice is associated.
profileId	string	The id of the Traffic Router Profile with which this deliveryservice is associated.
profileName	string	The name of the Traffic Router Profile with which this deliveryservice is associated.
protocol	string	<ul style="list-style-type: none"> <li>• 0: serve with <a href="#">http://</a> at EDGE</li> <li>• 1: serve with <a href="#">https://</a> at EDGE</li> <li>• 2: serve with both <a href="#">http://</a> and <a href="#">https://</a> at EDGE</li> </ul>

Continued on next page



Table 24 – continued from previous page

Parameter	Type	Description
qstringIgnore	string	<ul style="list-style-type: none"> <li>• 0: no special query string handling; it is for use in the cache-key and pass up to origin.</li> <li>• 1: ignore query string in cache-key, but pass it up to parent and or origin.</li> <li>• 2: drop query string at edge, and do not use it in the cache-key.</li> </ul>
rangeRequestHandling	string	<p>How to treat range requests:</p> <ul style="list-style-type: none"> <li>• 0 Do not cache (ranges requested from files that are already cached due to a non range request will be a HIT)</li> <li>• 1 Use the <code>background_fetch</code> plugin.</li> <li>• 2 Use the <code>cache_range_requests</code> plugin.</li> </ul>
regexRemap	string	Regex Remap rule to apply to this delivery service at the Edge tier.
regionalGeoBlocking	bool	Regex Remap rule to apply to this delivery service at the Edge tier.
remapText	string	Additional raw remap line text.
routingName	string	The routing name of this deliveryservice.
signed	bool	<ul style="list-style-type: none"> <li>• false: token based auth (see <code>:ref:token-based-auth</code>) is not enabled for this deliveryservice.</li> <li>• true: token based auth is enabled for this deliveryservice.</li> </ul>
sslKeyVersion	string	
tenant	string	Owning tenant name
tenantId	int	Owning tenant ID.
trRequestHeaders	string	
trResponseHeaders	string	
type	string	The type of this deliveryservice (one of <code>:ref:to-api-v11-types</code> use_in_table='deliveryservice').

Continued on next page

Table 24 – continued from previous page

Parameter	Type	Description
typeId	string	The type of this deliveryservice (one of :ref:to-api-v11-types use_in_table='deliveryservice').
xmlId	string	Unique string that describes this deliveryservice.

**Response Example**

```
{
  "response": [
    {
      "active": true,
      "cacheurl": null,
      "ccrDnsTtl": "3600",
      "cdnId": "2",
      "cdnName": "over-the-top",
      "checkPath": "",
      "deepCachingType": "NEVER",
      "displayName": "My Cool Delivery Service",
      "dnsBypassCname": "",
      "dnsBypassIp": "",
      "dnsBypassIp6": "",
      "dnsBypassTtl": "30",
      "dscp": "40",
      "edgeHeaderRewrite": null,
      "exampleURLs": [
        "http://foo.foo-ds.foo.bar.net"
      ],
      "geoLimit": "0",
      "geoLimitCountries": null,
      "geoLimitRedirectURL": null,
      "geoProvider": "0",
      "globalMaxMbps": null,
      "globalMaxTps": "0",
      "httpBypassFqdn": "",
      "id": "442",
      "infoUrl": "",
      "initialDispersion": "1",
      "ipv6RoutingEnabled": true,
      "lastUpdated": "2016-01-26 08:49:35",
      "logsEnabled": false,
      "longDesc": "",
      "longDesc1": "",
      "longDesc2": "",
      "matchList": [
        {
          "pattern": ".*\\.foo-ds\\..*",
          "setNumber": "0",
          "type": "HOST_REGEX"
        }
      ],
      "maxDnsAnswers": "0",
      "midHeaderRewrite": null,
      "missLat": "41.881944",
      "missLong": "-87.627778",
    }
  ]
}
```

(continues on next page)

(continued from previous page)

```

        "multiSiteOrigin": false,
        "multiSiteOriginAlgorithm": null,
        "orgServerFqdn": "http://baz.boo.net",
        "originShield": null,
        "profileDescription": "Content Router for over-the-top",
        "profileId": "5",
        "profileName": "ROUTER_TOP",
        "protocol": "0",
        "qstringIgnore": "1",
        "rangeRequestHandling": "0",
        "regexRemap": null,
        "regionalGeoBlocking": false,
        "remapText": null,
        "routingName": "foo",
        "signed": false,
        "sslKeyVersion": "0",
        "tenant": "root",
        "tenantId": 1,
        "trRequestHeaders": null,
        "trResponseHeaders": "Access-Control-Allow-Origin: *",
        "type": "HTTP",
        "typeId": "8",
        "xmlId": "foo-ds"
    }
    { .. },
    { .. }
]
}

```

**GET /api/1.2/servers/totals**

Retrieves a count of CDN servers by type.

Authentication Required: Yes

Role(s) Required: None

**Response Properties**

Parameter	Type	Description
count	int	The number of servers of this type in this instance of Traffic Ops.
type	string	The name of the type of the server count (see <a href="#">Types</a> ).

**Response Example**

```

{
  "response": [
    {
      "count": 4,
      "type": "CCR"
    },
    {

```

(continues on next page)

(continued from previous page)

```

    "count": 55,
    "type": "EDGE"
  },
  {
    "type": "MID",
    "count": 18
  },
  {
    "count": 0,
    "type": "INFLUXDB"
  },
  {
    "count": 4,
    "type": "RASCAL"
  }
}

```

**GET /api/1.2/servers/status**

Retrieves a count of CDN servers by status.

Authentication Required: Yes

Role(s) Required: None

**Response Properties**

Parameter	Type	Description
ONLINE	int	The number of ONLINE servers. Traffic Monitor will not monitor the state of ONLINE servers. True health is unknown.
REPORTED	int	The number of REPORTED servers. Traffic Monitor monitors the state of REPORTED servers and removes them if unhealthy.
OFFLINE	int	The number of OFFLINE servers. Used for longer-term maintenance. These servers are excluded from CRConfig.json.
ADMIN_DOWN	int	The number of ADMIN_DOWN servers. Used for short-term maintenance. These servers are included in CRConfig.json.
CCR_IGNORE	int	The number of CCR_IGNORE servers. These servers are excluded from CRConfig.json.
PRE_PROD	int	The number of PRE_PROD servers. Used for servers to be deployed. These servers are excluded from CRConfig.json.

**Response Example**

```

{
  "response":
  {
    "ONLINE": 100,
    "OFFLINE": 23,
    "REPORTED": 45,
    "ADMIN_DOWN": 4,

```

(continues on next page)

(continued from previous page)

```

    "CCR_IGNORE": 1,
    "PRE_PROD": 0,
  }
}

```

**GET /api/1.2/servers/hostname/:name/details**

Retrieves the details of a server.

Authentication Required: Yes

Role(s) Required: None

**Request Route Parameters**

Name	Required	Description
name	yes	The host name part of the cache.

**Response Properties**

Parameter	Type	Description
cachegroup	string	The cache group name (see <a href="#">Cache Group</a> ).
deliveryservices	array	Array of strings with the delivery service ids assigned (see <a href="#">Delivery Service</a> ).
domainName	string	The domain name part of the FQDN of the cache.
hardwareInfo	hash	Hwinfo struct (see <a href="#">Hardware Info</a> ).
hostName	string	The host name part of the cache.
id	string	The server id (database row number).
iloIpAddress	string	The IPv4 address of the lights-out-management port.
iloIpGateway	string	The IPv4 gateway address of the lights-out-management port.
iloIpNetmask	string	The IPv4 netmask of the lights-out-management port.
iloPassword	string	The password of the of the lights-out-management user (displays as ** unless you are an 'admin'.
iloUsername	string	The user name for lights-out-management.
interfaceMtu	string	The Maximum Transmission Unit (MTU) to configure for interfaceName.
interfaceName	string	The network interface name used for serving traffic.
ip6Address	string	The IPv6 address/netmask for interfaceName.
ip6Gateway	string	The IPv6 gateway for interfaceName.
ipAddress	string	The IPv4 address for interfaceName.
ipGateway	string	The IPv4 gateway for interfaceName.
ipNetmask	string	The IPv4 netmask for interfaceName.
lastUpdated	string	The Time/Date of the last update for this server.
mgmtIpAddress	string	The IPv4 address of the management port (optional).
mgmtIpGateway	string	The IPv4 gateway of the management port (optional).
mgmtIpNetmask	string	The IPv4 netmask of the management port (optional).
physLocation	string	The physical location name (see <a href="#">Physical Location</a> ).
profile	string	The assigned profile name (see <a href="#">Profiles</a> ).
rack	string	A string indicating rack location.
routerHostName	string	The human readable name of the router.
routerPortName	string	The human readable name of the router port.

Continued on next

Table 25 – continued from previous page

Parameter	Type	Description
status	string	The Status string (See <i>Status</i> ).
tcpPort	string	The default TCP port on which the main application listens (80 for a cache in most cases).
httpsPort	string	The default HTTPS port on which the main application listens (443 for a cache in most cases).
type	string	The name of the type of this server (see <i>Types</i> ).
xmppId	string	Deprecated.
xmppPasswd	string	Deprecated.

**Response Example**

```
{
  "response": {
    "cachegroup": "us-il-chicago",
    "deliveryservices": [
      "1",
      "2",
      "3",
      "4"
    ],
    "domainName": "chi.kabletown.net",
    "hardwareInfo": {
      "Physical Disk 0:1:3": "D1S2",
      "Physical Disk 0:1:2": "D1S2",
      "Physical Disk 0:1:15": "D1S2",
      "Power Supply.Slot.2": "04.07.15",
      "Physical Disk 0:1:24": "YS08",
      "Physical Disk 0:1:1": "D1S2",
      "Model": "PowerEdge R720xd",
      "Physical Disk 0:1:22": "D1S2",
      "Physical Disk 0:1:18": "D1S2",
      "Enterprise UEFI Diagnostics": "4217A5",
      "Lifecycle Controller": "1.0.8.42",
      "Physical Disk 0:1:8": "D1S2",
      "Manufacturer": "Dell Inc.",
      "Physical Disk 0:1:6": "D1S2",
      "SysMemTotalSize": "196608",
      "PopulatedDIMMSlots": "24",
      "Physical Disk 0:1:20": "D1S2",
      "Intel(R) Ethernet 10G 2P X520 Adapter": "13.5.7",
      "Physical Disk 0:1:14": "D1S2",
      "BACKPLANE FIRMWARE": "1.00",
      "Dell OS Drivers Pack, 7.0.0.29, A00": "7.0.0.29",
      "Integrated Dell Remote Access Controller": "1.57.57",
      "Physical Disk 0:1:5": "D1S2",
      "ServiceTag": "D6XPDV1",
      "PowerState": "2",
      "Physical Disk 0:1:23": "D1S2",
      "Physical Disk 0:1:25": "D903",
      "BIOS": "1.3.6",
      "Physical Disk 0:1:12": "D1S2",
      "System CPLD": "1.0.3",
      "Physical Disk 0:1:4": "D1S2",
      "Physical Disk 0:1:0": "D1S2",
      "Power Supply.Slot.1": "04.07.15",
      "PERC H710P Mini": "21.0.2-0001",
      "PowerCap": "689",
    }
  }
}
```

(continues on next page)

(continued from previous page)

```

    "Physical Disk 0:1:16": "D1S2",
    "Physical Disk 0:1:10": "D1S2",
    "Physical Disk 0:1:11": "D1S2",
    "Lifecycle Controller 2": "1.0.8.42",
    "BP12G+EXP 0:1": "1.07",
    "Physical Disk 0:1:9": "D1S2",
    "Physical Disk 0:1:17": "D1S2",
    "Broadcom Gigabit Ethernet BCM5720": "7.2.20",
    "Physical Disk 0:1:21": "D1S2",
    "Physical Disk 0:1:13": "D1S2",
    "Physical Disk 0:1:7": "D1S2",
    "Physical Disk 0:1:19": "D1S2"
  },
  "hostName": "atsec-chi-00",
  "id": "19",
  "iloIpAddress": "172.16.2.6",
  "iloIpGateway": "172.16.2.1",
  "iloIpNetmask": "255.255.255.0",
  "iloPassword": "*****",
  "iloUsername": "",
  "interfaceMtu": "9000",
  "interfaceName": "bond0",
  "ip6Address": "2033:D0D0:3300::2:2/64",
  "ip6Gateway": "2033:D0D0:3300::2:1",
  "ipAddress": "10.10.2.2",
  "ipGateway": "10.10.2.1",
  "ipNetmask": "255.255.255.0",
  "mgmtIpAddress": "",
  "mgmtIpGateway": "",
  "mgmtIpNetmask": "",
  "physLocation": "plocation-chi-1",
  "profile": "EDGE1_CDN1_421_SSL",
  "rack": "RR 119.02",
  "routerHostName": "rtr-chi.kabletown.net",
  "routerPortName": "2",
  "status": "ONLINE",
  "tcpPort": "80",
  "httpsPort": "443",
  "type": "EDGE",
  "xmppId": "atsec-chi-00-dummyxmpp",
  "xmppPasswd": "X"
}

```

**POST /api/1.2/servercheck**

Post a server check result to the serverchecks table.

Authentication Required: Yes

Role(s) Required: None

**Request Route Parameters**

Name	Required	Description
id	yes	
host_name	yes	
servercheck_short_name	yes	
value	yes	

### Request Example

```
{
  "id": "",
  "host_name": "",
  "servercheck_short_name": "",
  "value": ""
}
```

### Response Properties

Parameter	Type	Description
alerts	array	A collection of alert messages.
>level	string	Success, info, warning or error.
>text	string	Alert message.
version	string	

### Response Example

Response Example:

```
{
  "alerts":
  [
    {
      "level": "success",
      "text": "Server Check was successfully updated."
    }
  ],
}
```

## POST /api/1.2/servers

Allow user to create a server.

Authentication Required: Yes

Role(s) Required: admin or oper

### Request Properties



Name	Required	Description
hostName	yes	The host name part of the server.
domainName	yes	The domain name part of the FQDN of the cache.
cachegroupId	yes	Cache Group ID
interfaceName	yes	The interface name (e.g. eth0, p2p1).
ipAddress	yes	Must be unique per server profile.
ipNetmask	yes	The IPv4 Netmask
ipGateway	yes	The IPv4 Gateway.
interfaceMtu	yes	1500 or 9000
physLocationId	yes	The ID of the Physical Location.
typeId	yes	The ID of the Server Type
profileId	yes	Profile ID - Profile's CDN must match server's.
cdnId	yes	CDN ID the server belongs to
updPending	yes	Is there an update pending for this server. (true or false)
statusId	yes	The Status ID of the server.
tcpPort	no	Must be a valid TCP port if specified.
httpsPort	no	Must be a valid TCP port if specified.
xmppId	no	
xmppPasswd	no	
ip6Address	no	IPv6 address and prefix. Must be unique per server profile.
ip6Gateway	no	IPv6 Gateway
rack	no	The rack location in the Data Center.
mgmtIpAddress	no	The IPv4 management address.
mgmtIpNetmask	no	The IPv4 management netmask.
mgmtIpGateway	no	The IPv4 management gateway.
iloIpAddress	no	The IPv4 ILO address.
iloIpNetmask	no	The IPv4 ILO netmask.
iloIpGateway	no	The IPv4 ILO gateway.
iloUsername	no	The ILO username.
iloPassword	no	The ILO password.
routerHostName	no	The hostname of the router the server is connected to.
routerPortName	no	The portname in the router.

### Request Example

```
{
  "hostName": "tc1_ats1",
  "domainName": "cdn1.kabletown.test",
  "cachegroupId": 1,
  "cdnId": 1,
  "interfaceName": "eth0",
  "ipAddress": "10.74.27.188",
  "ipNetmask": "255.255.255.0",
  "ipGateway": "10.74.27.1",
  "interfaceMtu": 1500,
  "physLocationId": 1,
  "typeId": 1,
  "profileId": 1,
  "updPending": true,
  "statusId": 1
}
```

## Response Properties

Name	Type	Description
hostName	string	The host name part of the server.
Name	string	Description
domainName	string	The domain name part of the FQDN of the cache.
cachegroup	string	cache group name
interfaceName	string	
ipAddress	string	
ipNetmask	string	
ipGateway	string	
interfaceMtu	string	1500 or 9000
physLocation	string	
type	string	server type
profile	string	
cdnName	string	cdn name the server belongs to
tcpPort	string	
httpsPort	string	
xmppId	string	
xmppPasswd	string	
ip6Address	string	
ip6Gateway	string	
rack	string	
mgmtIpAddress	string	
mgmtIpNetmask	string	
mgmtIpGateway	string	
iloIpAddress	string	
iloIpNetmask	string	
iloIpGateway	string	
iloUsername	string	
iloPassword	string	
routerHostName	string	
routerPortName	string	

## Response Example

```
{
  'response' : {
    'profileId' : 1,
    'xmppPasswd' : '*****',
    'profile' : 'EDGE1_CDN1_421',
    'iloUsername' : 'username',
    'statusId' : 1,
    'status' : 'REPORTED',
    'ipAddress' : '10.74.27.188',
    'cdnId' : 1,
    'physLocation' : 'plocation-chi-1',
    'cachegroup' : 'cache_group_edge',
    'interfaceName' : 'eth0',
    'ip6Gateway' : null,
    'iloPassword' : null,
    'id' : 1003,
    'routerPortName' : null,
```

(continues on next page)

(continued from previous page)

```

    'lastUpdated' : '2016-01-25 14:16:16',
    'ipNetmask' : '255.255.255.0',
    'ipGateway' : '10.74.27.1',
    'tcpPort' : 80,
    'httpsPort' : 443,
    'mgmtIpAddress' : null,
    'ip6Address' : null,
    'interfaceMtu' : 1500,
    'iloIpGateway' : null,
    'hostName' : 'tcl_ats1',
    'xmppId' : 'tcl_ats1',
    'rack' : null,
    'mgmtIpNetmask' : null,
    'iloIpAddress' : null,
    'mgmtIpGateway' : null,
    'type' : 'EDGE',
    'domainName' : 'cdn1.kabletown.test',
    'iloIpNetmask' : null,
    'routerHostName' : null,
    'updPending' : false,
    'guid' : null,
    'physLocationId' : 1,
    'offlineReason' : 'N/A',
    'cachegroupId' : 1,
    'typeId' : 1,
    'cdnName' : 'cdn1',
    'profileDesc' : 'The profile description'
  }
}

```

**PUT /api/1.2/servers/{:id}**

Allow user to edit server through api.

Authentication Required: Yes

Role(s) Required: admin or oper

**Request Route Parameters**

Name	Required	Description
id	yes	The id of the server to edit.

**Request Properties**

Name	Required	Description
hostName	yes	The host name part of the server.
domainName	yes	The domain name part of the FQDN of the cache.
cachegroup	yes	cache_group name
interfaceName	yes	
ipAddress	yes	Must be unique per server profile.
ipNetmask	yes	
ipGateway	yes	
interfaceMtu	no	1500 or 9000
physLocation	yes	
type	yes	server type
profile	yes	Profile ID - Profile's CDN must match server's.
cdnName	yes	cdn name the server belongs to
tcpPort	no	
httpsPort	no	
xmppId	no	
xmppPasswd	no	
ip6Address	no	Must be unique per server profile.
ip6Gateway	no	
rack	no	
mgmtIpAddress	no	
mgmtIpNetmask	no	
mgmtIpGateway	no	
iloIpAddress	no	
iloIpNetmask	no	
iloIpGateway	no	
iloUsername	no	
iloPassword	no	
routerHostName	no	
routerPortName	no	

### Request Example

```
{
  "hostName": "tcl_ats2",
  "domainName": "my.test.com",
  "cachegroup": "cache_group_edge",
  "cdnName": "cdn_number_1",
  "interfaceName": "eth0",
  "ipAddress": "10.74.27.188",
  "ipNetmask": "255.255.255.0",
  "ipGateway": "10.74.27.1",
  "interfaceMtu": "1500",
  "physLocation": "plocation-chi-1",
  "type": "EDGE",
  "profile": "EDGE1_CDN1_421"
}
```

### Response Properties

Name	Type	Description
hostName	string	The host name part of the server.
Name	string	Description
domainName	string	The domain name part of the FQDN of the cache.
cachegroup	string	cache group name
interfaceName	string	
ipAddress	string	
ipNetmask	string	
ipGateway	string	
interfaceMtu	string	1500 or 9000
physLocation	string	
type	string	server type
profile	string	
cdnName	string	cdn name the server belongs to
tcpPort	string	
httpsPort	string	
xmppId	string	
xmppPasswd	string	
ip6Address	string	
ip6Gateway	string	
rack	string	
mgmtIpAddress	string	
mgmtIpNetmask	string	
mgmtIpGateway	string	
iloIpAddress	string	
iloIpNetmask	string	
iloIpGateway	string	
iloUsername	string	
iloPassword	string	
routerHostName	string	
routerPortName	string	

### Response Example

```
{
  'response' : {
    'xmppPasswd' : '*****',
    'profile' : 'EDGE1_CDN1_421',
    'iloUsername' : null,
    'status' : 'REPORTED',
    'ipAddress' : '10.74.27.188',
    'cdnId' : '1',
    'physLocation' : 'plocation-chi-1',
    'cachegroup' : 'cache_group_edge',
    'interfaceName' : 'eth0',
    'ip6Gateway' : null,
    'iloPassword' : null,
    'id' : '1003',
    'routerPortName' : null,
    'lastUpdated' : '2016-01-25 14:16:16',
    'ipNetmask' : '255.255.255.0',
    'ipGateway' : '10.74.27.1',
    'tcpPort' : '80',
```

(continues on next page)

(continued from previous page)

```
'httpsPort' : '443',
'mgmtIpAddress' : null,
'ip6Address' : null,
'interfaceMtu' : '1500',
'iloIpGateway' : null,
'hostName' : 'tcl_ats2',
'xmppId' : 'tcl_ats1',
'rack' : null,
'mgmtIpNetmask' : null,
'iloIpAddress' : null,
'mgmtIpGateway' : null,
'type' : 'EDGE',
'domainName' : 'my.test.com',
'iloIpNetmask' : null,
'routerHostName' : null
}
}
```

### PUT /api/1.2/servers/{:id}/status

Updates server status and queues updates on all child caches if server type is EDGE or MID. Also, captures offline reason if status is set to ADMIN\_DOWN or OFFLINE and prepends offline reason with the user that initiated the status change.

Authentication Required: Yes

Role(s) Required: Admin or Operations

#### Request Route Parameters

Name	Required	Description
id	yes	The id of the server.

#### Request Properties

Name	Required	Description
status	yes	Status ID or name.
offlineReason	yes/no	Required if status is ADMIN_DOWN or OFFLINE.

#### Request Example

```
{
  "status": "ADMIN_DOWN",
  "offlineReason": "Bad drives"
}
```

#### Response Properties

Parameter	Type	Description
alerts	array	A collection of alert messages.
>level	string	Success, info, warning or error.
>text	string	Alert message.

**Response Example**

```
{
  "alerts": [
    {
      "level": "success",
      "text": "Updated status [ ADMIN_DOWN ] for foo.bar.
↪net [ user23: bad drives ] and queued updates on all child caches"
    }
  ],
}
```

**DELETE /api/1.2/servers/{:id}**

Allow user to delete server through api.

Authentication Required: Yes

Role(s) Required: admin or oper

**Request Route Parameters**

Name	Required	Description
id	yes	The id of the server to delete.

**Response Properties**

Parameter	Type	Description
alerts	array	A collection of alert messages.
>level	string	Success, info, warning or error.
>text	string	Alert message.
version	string	

**Response Example**

```
{
  "alerts": [
    {
      "level": "success",
      "text": "Server was deleted."
    }
  ],
}
```

**POST /api/1.2/servers/{:id}/queue\_update**

Queue or dequeue updates for a specific server.

Authentication Required: Yes

Role(s) Required: admin or oper

**Request Route Parameters**

Name	Required	Description
id	yes	the server id.

**Request Properties**

Name	Type	Description
action	string	queue or dequeue

**Response Properties**

Name	Type	Description
action	string	The action processed, queue or dequeue.
serverId	integer	server id

**Response Example**

```
{
  "response": {
    "serverId": "1",
    "action": "queue"
  }
}
```

**Static DNS Entries****/api/1.2/staticdnsentries****GET /api/1.2/staticdnsentries.json**

Authentication Required: Yes

Role(s) Required: None

**Response Properties**



Parameter	Type	Description
deliveryservice	string	
ttl	string	
type	string	
address	string	
cachegroup	string	
host	string	

**Response Example**

```
{
  "response": [
    {
      "deliveryservice": "foo-ds",
      "ttl": "30",
      "type": "CNAME_RECORD",
      "address": "bar.foo.baz.tv.",
      "cachegroup": "us-co-denver",
      "host": "mm"
    }
  ]
}
```

**Status****/api/1.2/statuses****GET /api/1.2/statuses**

Retrieves a list of the server status codes available.

Authentication Required: Yes

Role(s) Required: None

**Response Properties**

Parameter	Type	Description
id	string	The id with which Traffic Ops stores this status, and references it internally
name	string	The string equivalent of the status
description	string	A short description of the status
lastUpdated	string	The Time / Date this server entry was last updated

**Response Example**

```
{
  "response": [
    {
      "id": "4",
      "name": "ADMIN_DOWN",
```

(continues on next page)

(continued from previous page)

```
    "description": "Temporary down. Edge: XMPP client will send status,
↳OFFLINE to CCR, otherwise similar to REPORTED. Mid: Server will not be
↳included in parent.config files for its edge caches",
    "lastUpdated": "2013-02-13 16:34:29"
  },
  {
    "id": "5",
    "name": "CCR_IGNORE",
    "description": "Edge: 12M will not include caches in this state in CCR,
↳config files. Mid: N/A for now",
    "lastUpdated": "2013-02-13 16:34:29"
  },
  {
    "id": "1",
    "name": "OFFLINE",
    "description": "Edge: Puts server in CCR config file in this state,
↳but CCR will never route traffic to it. Mid: Server will not be included
↳in parent.config files for its edge caches",
    "lastUpdated": "2013-02-13 16:34:29"
  },
  {
    "id": "2",
    "name": "ONLINE",
    "description": "Edge: Puts server in CCR config file in this state,
↳and CCR will always route traffic to it. Mid: Server will be included in
↳parent.config files for its edges",
    "lastUpdated": "2013-02-13 16:34:29"
  },
  {
    "id": "3",
    "name": "REPORTED",
    "description": "Edge: Puts server in CCR config file in this state,
↳and CCR will adhere to the health protocol. Mid: N/A for now",
    "lastUpdated": "2013-02-13 16:34:29"
  }
]
```

**GET /api/1.2/statuses/:id**

Retrieves a server status by ID.

Authentication Required: Yes

Role(s) Required: None

**Request Route Parameters**

Name	Required	Description
id	yes	Status id.

**Response Properties**

Parameter	Type	Description
id	string	The id with which Traffic Ops stores this status, and references it internally
name	string	The string equivalent of the status
description	string	A short description of the status
lastUpdated	string	The Time / Date this server entry was last updated

**Response Example**

```
{
  "response": [
    {
      "id": "4",
      "name": "ADMIN_DOWN",
      "description": "Temporary down. Edge: XMPP client will send status_
↪OFFLINE to CCR, otherwise similar to REPORTED. Mid: Server will not be_
↪included in parent.config files for its edge caches",
      "lastUpdated": "2013-02-13 16:34:29"
    }
  ]
}
```

**Steering Targets****GET /api/1.2/steering/:dsId/targets**

Get all targets for a steering delivery service.

Authentication Required: Yes

Role(s) Required: None

**Request Route Parameters**

Name	Required	Description
dsId	yes	DS ID.

**Response Properties**

Parameter	Type	Description
deliveryServiceId	int	DS ID
deliveryService	string	DS XML ID
targetId	int	Target DS ID
target	string	Target DS XML ID
value	int	Value is weight or order depending on type
typeId	int	Steering target type ID
type	string	Steering target type name

**Response Example**

```
{
  "response": [
    {
      "deliveryServiceId": 1
```

(continues on next page)

(continued from previous page)

```

    "deliveryService": "steering-ds-one",
    "targetId": 2,
    "target": "steering-target-one",
    "value": 1,
    "typeId": 35,
    "type": "STEERING_ORDER"
  },
  {
    "deliveryServiceId": 1
    "deliveryService": "steering-ds-one",
    "targetId": 3,
    "target": "steering-target-two",
    "value": 2,
    "typeId": 35,
    "type": "STEERING_ORDER"
  },
]
}

```

**GET /api/1.2/steering/:dsId/targets/:targetId**

Get a steering target.

Authentication Required: Yes

Role(s) Required: None

**Request Route Parameters**

Name	Required	Description
dsId	yes	DS ID.
targetId	yes	DS Target ID.

**Response Properties**

Parameter	Type	Description
deliveryServiceId	int	DS ID
deliveryService	string	DS XML ID
targetId	int	Target DS ID
target	string	Target DS XML ID
value	int	Value is weight or order depending on type
typeId	int	Steering target type ID
type	string	Steering target type name

**Response Example**

```

{
  "response": [
    {
      "deliveryServiceId": 1

```

(continues on next page)

(continued from previous page)

```

    "deliveryService": "steering-ds-one",
    "targetId": 2,
    "target": "steering-target-one",
    "value": 1,
    "typeId": 35,
    "type": "STEERING_ORDER"
  }
]
}
```

**PUT /api/1.2/steering/:dsId/targets/:targetId**

Update a steering target.

Authentication Required: Yes

Role(s) Required: Admin or Steering

**Request Route Parameters**

Name	Required	Description
dsId	yes	DS ID.
targetId	yes	DS Target ID.

**Request Properties**

Parameter	Required	Description
value	yes	Target value
typeId	yes	Target type ID

**Request Example**

```

{
  "value": 34,
  "typeId": 46,
}
```

**Response Properties**

Parameter	Type	Description
deliveryServiceId	int	Steering DS ID
deliveryService	string	DS XML ID
targetId	int	Target DS ID
target	string	Target DS XML ID
value	string	Target value
typeId	int	Target type ID
type	string	Steering target type name

### Response Example

```
{
  "response": {
    "deliveryServiceId": 1,
    "deliveryService": "steering-ds-one",
    "targetId": 2,
    "target": "steering-target-two",
    "value": "34",
    "typeId": 45,
    "type": "STEERING_ORDER"
  },
  "alerts": [
    {
      "level": "success",
      "text": "Delivery service steering target update was_↵
↵successful."
    }
  ]
}
```

### POST /api/1.2/steering/:dsId/targets

Create a steering target.

Authentication Required: Yes

Role(s) Required: Admin or Steering

#### Request Route Parameters

Name	Required	Description
dsId	yes	DS ID.

#### Request Properties

Parameter	Required	Description
targetId	yes	Target DS ID
value	yes	Target value
typeId	yes	Target type ID

### Request Example

```
{
  "targetId": 6,
  "value": 22,
  "typeId": 47,
}
```

### Response Properties

Parameter	Type	Description
deliveryServiceId	int	Steering DS ID
deliveryService	string	DS XML ID
targetId	int	Target DS ID
target	string	Target DS XML ID
value	string	Target value
typeId	int	Target type ID
type	string	Steering target type name

### Response Example

```
{
  "response": {
    "deliveryServiceId": 1,
    "deliveryService": "steering-ds-one",
    "targetId": 6,
    "target": "steering-target-six",
    "value": "22",
    "typeId": 47,
    "type": "STEERING_ORDER"
  },
  "alerts": [
    {
      "level": "success",
      "text": "Delivery service target creation was_
↪successful."
    }
  ]
}
```

### DELETE /api/1.2/steering/:dsId/targets/:targetId

Delete a steering target.

Authentication Required: Yes

Role(s) Required: Admin or Steering

#### Request Route Parameters

Name	Required	Description
dsId	yes	DS ID.
targetId	yes	DS Target ID.

### Response Example

```
{
  "alerts": [
    {
      "level": "success",
      "text": "Delivery service target delete was ↵
↵successful."
    }
  ],
}
```

## System

### /api/1.1/system

#### GET /api/1.2/system/info.json

Authentication Required: Yes

Role(s) Required: None

#### Response Properties



Key	Type	Description
parameters	hash	This is a hash with the parameter names that describe the Traffic Ops installation as keys. These are all the parameters in the GLOBAL profile.
>tm. toolname	string	The name of the Traffic Ops tool. Usually “Traffic Ops”. Used in the About screen and in the comments headers of the files generated (# DO NOT EDIT – Generated for atsec-lax-04 by Traffic Ops (https://traffops.kabletown.net/) on Fri Mar 6 05:15:15 UTC 2015).
>tm. instance_name	string	The name of the Traffic Ops instance. Can be used when multiple instances are active. Visible in the About page.
>traffic_mon_fwd_proxy	string	When collecting stats from Traffic Router, Traffic Ops uses this forward proxy to pull the stats through. This can be any of the MID tier caches, or a forward cache specifically deployed for this purpose. Setting this variable can significantly lighten the load on the Traffic Router stats system and it is recommended to set this parameter on a production system.
>tm. url	string	The URL for this Traffic Ops instance. Used in the About screen and in the comments headers of the files generated (# DO NOT EDIT – Generated for atsec-lax-04 by Traffic Ops (https://traffops.kabletown.net/) on Fri Mar 6 05:15:15 UTC 2015).
>traffic_rtr_fwd_proxy	string	When collecting stats from Traffic Monitor, Traffic Ops uses this forward proxy to pull the stats through. This can be any of the MID tier caches, or a forward cache specifically deployed for this purpose. Setting this variable can significantly lighten the load on the Traffic Monitor system and it is recommended to set this parameter on a production system.
>tm. logourl	string	This is the URL of the logo for Traffic Ops and can be relative if the logo is under traffic_ops/app/public.
>tm. infourl	string	This is the “for more information go here” URL, which is visible in the About page.

### Response Example

```
{
  "response": {
    "parameters": {
      "tm.toolname": "Traffic Ops",
      "tm.infourl": "http://staging-03.cdnlabs.kabletown.net/tm/info",
      "traffic_mon_fwd_proxy": "http://proxy.kabletown.net:81",
      "traffic_rtr_fwd_proxy": "http://proxy.kabletown.net:81",
      "tm.logourl": "\images\tc_logo.png",
      "tm.url": "https://tm.kabletown.net/",
      "tm.instance_name": "Kabletown CDN"
    }
  },
}
```

## Tenants

### /api/1.2/tenants

#### GET /api/1.2/tenants

Get all tenants.

Authentication Required: Yes

Role(s) Required: None

#### Response Properties

Parameter	Type	Description
id	int	Tenant id
name	string	Tenant name
active	bool	Active or inactive
parentId	int	Parent tenant ID
parentName	string	Parent tenant name

#### Response Example

```
{
  "response": [
    {
      "id": 1
      "name": "root",
      "active": true,
      "parentId": null,
      "parentName": null,
    },
    {
      "id": 2
      "name": "tenant-a",
      "active": true,
      "parentId": 1
      "parentName": "root"
    }
  ]
}
```

#### GET /api/1.2/tenants:id

Get a tenant by ID.

Authentication Required: Yes

Role(s) Required: None

#### Response Properties

Parameter	Type	Description
id	int	Tenant id
name	string	Tenant name
active	bool	Active or inactive
parentId	int	Parent tenant ID
parentName	string	Parent tenant name

#### Response Example

```
{
  "response": [
    {
      "id": 2
      "name": "tenant-a",
      "active": true,
      "parentId": 1,
      "parentName": "root"
    }
  ]
}
```

### **PUT /api/1.2/tenants/:id**

Update a tenant.

Authentication Required: Yes

Role(s) Required: admin or oper

#### **Request Route Parameters**

Name	Type	Description
id	int	Tenant id

#### **Request Properties**

Parameter	Required	Description
name	yes	The name of the tenant
active	yes	True or false
parentId	yes	Parent tenant

#### **Request Example**

```
{
  "name": "my-tenant"
  "active": true
  "parentId": 1
}
```

#### **Response Properties**

Parameter	Type	Description
id	int	Tenant id
name	string	Tenant name
active	bool	Active or inactive
parentId	int	Parent tenant ID
parentName	string	Parent tenant name

### Response Example

```
{
  "response": {
    "id": 2,
    "name": "my-tenant",
    "active": true,
    "parentId": 1,
    "parentName": "root",
    "lastUpdated": "2014-03-18 08:57:39"
  },
  "alerts": [
    {
      "level": "success",
      "text": "Tenant update was successful."
    }
  ]
}
```

### POST /api/1.2/tenants

Create a tenant.

Authentication Required: Yes

Role(s) Required: admin or oper

#### Request Properties

Parameter	Required	Description
name	yes	The name of the tenant
active	no	Defaults to false
parentId	yes	Parent tenant

### Request Example

```
{
  "name": "your-tenant"
  "parentId": 2
}
```

### Response Properties

Parameter	Type	Description
id	int	Tenant id
name	string	Tenant name
active	bool	Active or inactive
parentId	int	Parent tenant ID
parentName	string	Parent tenant name

### Response Example

```
{
  "response": {
    "id": 2,
    "name": "your-tenant",
    "active": false,
    "parentId": 2,
    "parentName": "my-tenant",
    "lastUpdated": "2014-03-18 08:57:39"
  },
  "alerts": [
    {
      "level": "success",
      "text": "Tenant create was successful."
    }
  ]
}
```

## TO Extensions

### `/api/1.2/to_extensions`

#### GET `/api/1.2/to_extensions.json`

Retrieves the list of extensions.

Authentication Required: Yes

Role(s) Required: None

#### Response Properties

Parameter	Type	Description
script_file	string	
version	string	
name	string	
description	string	
info_url	string	
additional_config_json	string	
isactive	string	
id	string	
type	string	
servercheck_short_name	string	

### Response Example

```
{
  "response": [
    {
      script_file: "ping",
      version: "1.0.0",
      name: "ILO_PING",
      description: null,
      info_url: "http://foo.com/bar.html",
      additional_config_json: "{ \"path\": \"/api/1.2/servers.json\",
↪"match": { \"type\": \"EDGE\"}, \"select\": \"ilo_ip_address\", \"cron\": \"9 * * * *\" }",
      isactive: "1",
      id: "1",
      type: "CHECK_EXTENSION_BOOL",
      servercheck_short_name: "ILO"
    },
    {
      script_file: "ping",
      version: "1.0.0",
      name: "10G_PING",
      description: null,
      info_url: "http://foo.com/bar.html",
      additional_config_json: "{ \"path\": \"/api/1.2/servers.json\",
↪"match": { \"type\": \"EDGE\"}, \"select\": \"ip_address\", \"cron\": \"18 * * * *\" }",
      isactive: "1",
      id: "2",
      type: "CHECK_EXTENSION_BOOL",
      servercheck_short_name: "10G"
    }
  ],
}
```

### POST /api/1.2/to\_extensions

Creates a Traffic Ops extension.

Authentication Required: Yes

Role(s) Required: None

## Request Parameters

Parameter	Type	Description
name	string	
version	string	
info_url	string	
script_file	string	
isactive	string	
additional_config_json	string	
description	string	
servercheck_short_name	string	
type	string	

## Request Example

```
{
  "name": "ILO_PING",
  "version": "1.0.0",
  "info_url": "http://foo.com/bar.html",
  "script_file": "ping",
  "isactive": "1",
  "additional_config_json": "{ \"path\": \"/api/1.2/servers.json\", \"match\":
→ { \"type\": \"EDGE\"} }",
  "description": null,
  "servercheck_short_name": "ILO"
  "type": "CHECK_EXTENSION_BOOL",
}
```

## Response Properties

Parameter	Type	Description
alerts	array	A collection of alert messages.
>level	string	Success, info, warning or error.
>text	string	Alert message.

## Response Example

```
{
  "alerts": [
    {
      "level": "success",
      "text": "Check Extension loaded."
    }
  ],
}
```

**POST /api/1.2/to\_extensions/:id/delete**

Deletes a Traffic Ops extension.

Authentication Required: Yes

Role(s) Required: None

**Request Route Parameters**

Name	Required	Description
id	yes	TO extension id

**Response Properties**

Parameter	Type	Description
alerts	array	A collection of alert messages.
>level	string	Success, info, warning or error.
>text	string	Alert message.

**Response Example**

```
{
  "alerts": [
    {
      "level": "success",
      "text": "Extension deleted."
    }
  ],
}
```

**Types****/api/1.2/types****GET /api/1.1/types**

Authentication Required: Yes

Role(s) Required: None

**Request Query Parameters**

Name	Required	Description
useInTable	no	Filter types by the table in which they apply

**Response Properties**



Parameter	Type	Description
id	string	
name	string	
description	string	
useInTable	string	
lastUpdated	string	

**Response Example**

```
{
  "response": [
    {
      "id": "22",
      "name": "AAAA_RECORD",
      "description": "Static DNS AAAA entry",
      "useInTable": "staticdnsentry",
      "lastUpdated": "2013-10-23 15:28:31"
    }
  ]
}
```

**GET /api/1.1/types/trimmed**

Authentication Required: Yes

Role(s) Required: None

**Response Properties**

Parameter	Type	Description
name	string	

**Response Example**

```
{
  "response": [
    {
      "name": "AAAA_RECORD"
    },
    {
      "name": "ACTIVE_DIRECTORY"
    },
    {
      "name": "A_RECORD"
    },
    {
      "name": "CCR"
    }
  ],
}
```

**GET /api/1.1/types/:id**

Authentication Required: Yes

Role(s) Required: None

**Request Route Parameters**

Name	Required	Description
id	yes	Type ID.

**Response Properties**

Parameter	Type	Description
id	string	
name	string	
description	string	
useInTable	string	
lastUpdated	string	

**Response Example**

```
{
  "response": [
    {
      "id": "22",
      "name": "AAAA_RECORD",
      "description": "Static DNS AAAA entry",
      "useInTable": "staticdnsentry",
      "lastUpdated": "2013-10-23 15:28:31"
    }
  ]
}
```

**Users****/api/1.2/users****GET /api/1.2/users**

Retrieves all users.

Authentication Required: Yes

Role(s) Required: None

**Request Query Parameters**

Name	Required	Description
tenant	no	Filter users by tenant ID.

**Response Properties**

Parameter	Type	Description
addressLine1	string	
addressLine2	string	
city	string	
company	string	
country	string	
email	string	
fullName	string	
gid	string	
id	hash	
lastUpdated	string	
newUser	string	
phoneNumber	string	
postalCode	string	
publicSshKey	string	
registrationSent	string	
role	string	
roleName	string	
stateOrProvince	string	
tenant	string	Owning tenant name
tenantId	int	Owning tenant ID
uid	string	
username	string	

### Response Example

```
{
  "response": [
    {
      "addressLine1": "",
      "addressLine2": "",
      "city": "",
      "company": "",
      "country": "",
      "email": "email1@email.com",
      "fullName": "Tom Simpson",
      "gid": "0",
      "id": "53",
      "lastUpdated": "2016-01-26 10:22:07",
      "newUser": true,
      "phoneNumber": "",
      "postalCode": "",
      "publicSshKey": "xxx",
      "registrationSent": true,
      "role": "6",
      "rolename": "admin",
      "stateOrProvince": "",
      "tenant": "root",
      "tenantId": 1,
      "uid": "0",
      "username": "tsimpson"
    },
    {
      ... more users
    }
  ]
}
```

(continues on next page)

(continued from previous page)

```
} ,  
]  
}
```

**GET /api/1.2/users/:id**

Retrieves user by ID.

Authentication Required: Yes

Role(s) Required: None

**Request Route Parameters**

Name	Required	Description
id	yes	User id.

**Response Properties**

Parameter	Type	Description
addressLine1	string	
addressLine2	string	
city	string	
company	string	
country	string	
email	string	
fullName	string	
gid	string	
id	hash	
lastUpdated	string	
newUser	string	
phoneNumber	string	
postalCode	string	
publicSshKey	string	
registrationSent	string	
role	string	
roleName	string	
stateOrProvince	string	
tenant	string	Owning tenant name
tenantId	int	Owning tenant ID
uid	string	
username	string	

**Response Example**

```
{  
  "response": [  
    {
```

(continues on next page)

(continued from previous page)

```

        "addressLine1": "",
        "addressLine2": "",
        "city": "",
        "company": "",
        "country": "",
        "email": "email1@email.com",
        "fullName": "Tom Simpson",
        "gid": "0",
        "id": "53",
        "lastUpdated": "2016-01-26 10:22:07",
        "newUser": true,
        "phoneNumber": "",
        "postalCode": "",
        "publicSshKey": "xxx",
        "registrationSent": true,
        "role": "6",
        "rolename": "admin",
        "stateOrProvince": "",
        "tenant": "root",
        "tenantId": 1,
        "uid": "0",
        "username": "tsimpson"
    },
    {
        ... more users
    },
]
}

```

**POST /api/1.2/users**

Create a user.

Authentication Required: Yes

Role(s) Required: admin or oper

**Request Properties**

Parameter	Type	Required	Description
addressLine1	string	no	
addressLine2	string	no	
city	string	no	
confirmLocalPassword	string	yes	
company	string	no	
country	string	no	
email	string	yes	
fullName	string	yes	
localPassword	string	yes	
newUser	bool	no	
phoneNumber	string	no	
postalCode	string	no	
publicSshKey	string	no	
role	int	yes	
stateOrProvince	string	no	
tenantId	int	no	Owning tenant ID
username	string	yes	

### Request Example

```
{
  "username": "tsimpson"
  "tenantId": 1,
  "fullName": "Tom Simpson"
  "email": "email1@email.com"
  "role": 6
  "localPassword": "password"
  "confirmLocalPassword": "password"
}
```

### Response Properties

Parameter	Type	Description
addressLine1	string	
addressLine2	string	
city	string	
company	string	
country	string	
email	string	
fullName	string	
gid	int	
id	int	
lastUpdated	string	
newUser	string	
phoneNumber	string	
postalCode	string	
publicSshKey	string	
registrationSent	bool	
role	int	
roleName	string	
stateOrProvince	string	
uid	int	
tenantId	int	Owning tenant ID
username	string	

### Response Example

```
{
  "alerts": [
    {
      "level": "success",
      "text": "User creation was successful."
    }
  ],
  "response": {
    "addressLine1": "",
    "addressLine2": "",
    "city": "",
    "company": "",
    "country": "",
    "email": "email1@email.com",
    "fullName": "Tom Simpson",
    "gid": 0,
    "id": 2,
    "lastUpdated": null,
    "newUser": false,
    "phoneNumber": "",
    "postalCode": "",
    "publicSshKey": "",
    "registrationSent": false,
    "role": 6,
    "roleName": "portal",
    "stateOrProvince": "",
    "tenantId": 1,
    "uid": 0,
    "username": "tsimpson",
  }
}
```

**POST /api/1.2/users/register**

Register a user and send registration email.

Authentication Required: Yes

Role(s) Required: Admin or Operations

**Request Properties**

Parameter	Type	Required	Description
email	string	yes	Email address of the new user.
role	int	yes	Role ID of the new user.
tenantId	int	yes	Tenant ID of the new user.

**Request Example**

```
{
  "email": "foo@bar.com"
  "role": 1,
  "tenantId": "1"
}
```

**Response Example**

```
{
  "alerts": [
    {
      "level": "success",
      "text": "Sent user registration to foo@bar.com with the
↪following permissions [ role: admin | tenant: root ]"
    }
  ]
}
```

**GET /api/1.2/users/:id/deliveryservices**

Retrieves all delivery services assigned to the user. See also [Using Traffic Ops - Delivery Service](#).

Authentication Required: Yes

Role(s) Required: None

**Request Route Parameters**

Name	Required	Description
id	yes	User ID.



**Response Properties**

Parameter	Type	Description
active	bool	true if active, false if inactive.
cacheurl	string	Cache URL rule to apply to this delivery service.
ccrDnsTtl	string	The TTL of the DNS response for A or AAAA queries requesting the IP address of the tr. host.
cdnId	string	Id of the CDN to which the delivery service belongs to.
cdnName	string	Name of the CDN to which the delivery service belongs to.
checkPath	string	The path portion of the URL to check this deliveryservice for health.
deepCachingType	string	When to do Deep Caching for this Delivery Service: <ul style="list-style-type: none"> <li>• NEVER (default)</li> <li>• ALWAYS</li> </ul>
displayName	string	The display name of the delivery service.
dnsBypassIp	string	The IPv4 IP to use for bypass on a DNS deliveryservice - bypass starts when serving more than the globalMaxMbps traffic on this deliveryservice.
dnsBypassIp6	string	The IPv6 IP to use for bypass on a DNS deliveryservice - bypass starts when serving more than the globalMaxMbps traffic on this deliveryservice.
dnsBypassTtl	string	The TTL of the DNS bypass response.
dscp	string	The Differentiated Services Code Point (DSCP) with which to mark downstream (EDGE -> customer) traffic.
edgeHeaderRewrite	string	The EDGE header rewrite actions to perform.
geoLimitRedirectUrl	string	

Continued on next page

Table 29 – continued from previous page

Parameter	Type	Description
geoLimit	string	<ul style="list-style-type: none"> <li>• 0: None - no limitations</li> <li>• 1: Only route on CZF file hit</li> <li>• 2: Only route on CZF hit or when from USA</li> </ul> <p>Note that this does not prevent access to content or makes content secure; it just prevents routing to the content by Traffic Router.</p>
geoLimitCountries	string	
geoProvider	string	
globalMaxMbps	string	The maximum global bandwidth allowed on this deliveryservice. If exceeded, the traffic routes to the dnsByPassIp* for DNS deliveryservices and to the httpBypassFqdn for HTTP deliveryservices.
globalMaxTps	string	The maximum global transactions per second allowed on this deliveryservice. When this is exceeded traffic will be sent to the dnsByPassIp* for DNS deliveryservices and to the httpBypassFqdn for HTTP deliveryservices
httpBypassFqdn	string	The HTTP destination to use for bypass on an HTTP deliveryservice - bypass starts when serving more than the globalMaxMbps traffic on this deliveryservice.
id	string	The deliveryservice id (database row number).
infoUrl	string	Use this to add a URL that points to more information about that deliveryservice.
initialDispersion	string	
ipv6RoutingEnabled	bool	false: send IPv4 address of Traffic Router to client on HTTP type del.
lastUpdated	string	
logsEnabled	bool	
longDesc	string	Description field 1.
longDesc1	string	Description field 2.
longDesc2	string	Description field 2.

Continued on next page

Table 29 – continued from previous page

Parameter	Type	Description
>>type	string	The type of MatchList (one of :ref:to-api-v11-types use_in_table='regex').
>>setNumber	string	The set Number of the match-List.
>>pattern	string	The regexp for the matchList.
maxDnsAnswers	string	The maximum number of IPs to put in a A/AAAA response for a DNS deliveryservice (0 means all available).
midHeaderRewrite	string	The MID header rewrite actions to perform.
missLat	string	The latitude to use when the client cannot be found in the CZF or the Geo lookup.
missLong	string	The longitude to use when the client cannot be found in the CZF or the Geo lookup.
multiSiteOrigin	bool	Is the Multi Site Origin feature enabled for this delivery service (0=false, 1=true). See <a href="#">Multi Site Origin</a>
multiSiteOriginAlgor	bool	Is the Multi Site Origin feature enabled for this delivery service (0=false, 1=true). See <a href="#">Multi Site Origin</a>
orgServerFqdn	string	The origin server base URL (FQDN when used in this instance, includes the protocol ( <a href="#">http://</a> or <a href="#">https://</a> ) for use in retrieving content from the origin server.
originShield	string	
profileDescription	string	The description of the Traffic Router Profile with which this deliveryservice is associated.
profileId	string	The id of the Traffic Router Profile with which this deliveryservice is associated.
profileName	string	The name of the Traffic Router Profile with which this deliveryservice is associated.
protocol	string	<ul style="list-style-type: none"> <li>• 0: serve with <a href="#">http://</a> at EDGE</li> <li>• 1: serve with <a href="#">https://</a> at EDGE</li> <li>• 2: serve with both <a href="#">http://</a> and <a href="#">https://</a> at EDGE</li> </ul>

Continued on next page

Table 29 – continued from previous page

Parameter	Type	Description
qstringIgnore	string	<ul style="list-style-type: none"> <li>• 0: no special query string handling; it is for use in the cache-key and pass up to origin.</li> <li>• 1: ignore query string in cache-key, but pass it up to parent and or origin.</li> <li>• 2: drop query string at edge, and do not use it in the cache-key.</li> </ul>
rangeRequestHandling	string	<p>How to treat range requests:</p> <ul style="list-style-type: none"> <li>• 0 Do not cache (ranges requested from files that are already cached due to a non range request will be a HIT)</li> <li>• 1 Use the <code>background_fetch</code> plugin.</li> <li>• 2 Use the <code>cache_range_requests</code> plugin.</li> </ul>
regexRemap	string	Regex Remap rule to apply to this delivery service at the Edge tier.
regionalGeoBlocking	bool	Regex Remap rule to apply to this delivery service at the Edge tier.
remapText	string	Additional raw remap line text.
routingName	string	The routing name of this deliveryservice.
signed	bool	<ul style="list-style-type: none"> <li>• false: token based auth (see <code>:ref:token-based-auth</code>) is not enabled for this deliveryservice.</li> <li>• true: token based auth is enabled for this deliveryservice.</li> </ul>
sslKeyVersion	string	
tenant	string	Owning tenant name
tenantId	int	Owning tenant ID.
trRequestHeaders	string	
trResponseHeaders	string	
type	string	The type of this deliveryservice (one of <code>:ref:to-api-v11-types</code> use_in_table='deliveryservice').

Continued on next page

Table 29 – continued from previous page

Parameter	Type	Description
typeId	string	The type of this deliveryservice (one of :ref:to-api-v11-types use_in_table='deliveryservice').
xmlId	string	Unique string that describes this deliveryservice.

**Response Example**

```
{
  "response": [
    {
      "active": true,
      "cacheurl": null,
      "ccrDnsTtl": "3600",
      "cdnId": "2",
      "cdnName": "over-the-top",
      "checkPath": "",
      "deepCachingType": "NEVER",
      "displayName": "My Cool Delivery Service",
      "dnsBypassCname": "",
      "dnsBypassIp": "",
      "dnsBypassIp6": "",
      "dnsBypassTtl": "30",
      "dscp": "40",
      "edgeHeaderRewrite": null,
      "exampleURLs": [
        "http://foo.foo-ds.foo.bar.net"
      ],
      "geoLimit": "0",
      "geoLimitCountries": null,
      "geoLimitRedirectURL": null,
      "geoProvider": "0",
      "globalMaxMbps": null,
      "globalMaxTps": "0",
      "httpBypassFqdn": "",
      "id": "442",
      "infoUrl": "",
      "initialDispersion": "1",
      "ipv6RoutingEnabled": true,
      "lastUpdated": "2016-01-26 08:49:35",
      "logsEnabled": false,
      "longDesc": "",
      "longDesc1": "",
      "longDesc2": "",
      "matchList": [
        {
          "pattern": ".*\\.foo-ds\\..*",
          "setNumber": "0",
          "type": "HOST_REGEX"
        }
      ],
      "maxDnsAnswers": "0",
      "midHeaderRewrite": null,
      "missLat": "41.881944",
      "missLong": "-87.627778",
    }
  ]
}
```

(continues on next page)

(continued from previous page)

```
"multiSiteOrigin": false,
"multiSiteOriginAlgorithm": null,
"orgServerFqdn": "http://baz.boo.net",
"originShield": null,
"profileDescription": "Content Router for over-the-top",
"profileId": "5",
"profileName": "ROUTER_TOP",
"protocol": "0",
"qstringIgnore": "1",
"rangeRequestHandling": "0",
"regexRemap": null,
"regionalGeoBlocking": false,
"remapText": null,
"routingName": "foo",
"signed": false,
"sslKeyVersion": "0",
"tenant": "root",
"tenantId": 1,
"trRequestHeaders": null,
"trResponseHeaders": "Access-Control-Allow-Origin: *",
"type": "HTTP",
"typeId": "8",
"xmlId": "foo-ds"
}
{ .. },
{ .. }
]
}
```

**GET /api/1.2/user/current**

Retrieves the profile for the authenticated user.

Authentication Required: Yes

Role(s) Required: None

**Response Properties**

Parameter	Type	Description
email	string	
city	string	
id	string	
phoneNumber	string	
company	string	
country	string	
fullName	string	
localUser	boolean	
uid	string	
stateOrProvince	string	
username	string	
newUser	boolean	
addressLine2	string	
role	string	
addressLine1	string	
gid	string	
postalCode	string	
tenant	string	Owning tenant name
tenantId	int	Owning tenant ID

### Response Example

```
{
  "response": {
    "email": "email@email.com",
    "city": "",
    "id": "50",
    "phoneNumber": "",
    "company": "",
    "country": "",
    "fullName": "Tom Callahan",
    "localUser": true,
    "uid": "0",
    "stateOrProvince": "",
    "username": "tommyboy",
    "newUser": false,
    "addressLine2": "",
    "role": "6",
    "addressLine1": "",
    "gid": "0",
    "postalCode": "",
    "tenant": "root",
    "tenantId": 1
  },
}
```

### PUT /api/1.2/user/current

Updates the date for the authenticated user.

Authentication Required: Yes

Role(s) Required: None

### Request Properties

Parameter	Type	Description
email	string	
city	string	
id	string	
phoneNumber	string	
company	string	
country	string	
fullName	string	
localUser	boolean	
uid	string	
stateOrProvince	string	
username	string	
newUser	boolean	
addressLine2	string	
role	string	
addressLine1	string	
gid	string	
postalCode	string	
tenantId	int	Owning tenant ID

### Request Example

```
{
  "user": {
    "email": "",
    "city": "",
    "id": "",
    "phoneNumber": "",
    "company": "",
    "country": "",
    "fullName": "",
    "localUser": true,
    "uid": "0",
    "stateOrProvince": "",
    "username": "tommyboy",
    "newUser": false,
    "addressLine2": "",
    "role": "6",
    "addressLine1": "",
    "gid": "0",
    "postalCode": "",
    "tenant": "root",
    "tenantId": 1,
  }
}
```



### Response Properties

Parameter	Type	Description
alerts	array	A collection of alert messages.
>level	string	Success, info, warning or error.
>text	string	Alert message.
version	string	

### Response Example

```
{
  "alerts": [
    {
      "level": "success",
      "text": "UserProfile was successfully updated."
    }
  ],
}
```

### GET /api/1.2/user/current/jobs.json

Retrieves the user's list of jobs.

Authentication Required: Yes

Role(s) Required: None

#### Request Query Parameters

Name	Required	Description
keyword	no	PURGE

### Response Properties

Parameter	Type	Description
keyword	string	
objectName	string	
assetUrl	string	
assetType	string	
status	string	
dsId	string	
dsXmlId	string	
username	boolean	
parameters	string	
enteredTime	string	
objectType	string	
agent	string	
id	string	
startTime	string	
version	string	

**Response Example**

```
{
  "response": [
    {
      "id": "1",
      "keyword": "PURGE",
      "objectName": null,
      "assetUrl": "",
      "assetType": "file",
      "status": "PENDING",
      "dsId": "9999",
      "dsXmlId": "ds-xml-id",
      "username": "peewee",
      "parameters": "TTL:56h",
      "enteredTime": "2015-01-21 18:00:16",
      "objectType": null,
      "agent": "",
      "startTime": "2015-01-21 10:45:38"
    }
  ],
}
```

**POST/api/1.2/user/current/jobs**

Invalidating content on the CDN is sometimes necessary when the origin was mis-configured and something is cached in the CDN that needs to be removed. Given the size of a typical Traffic Control CDN and the amount of content that can be cached in it, removing the content from all the caches may take a long time. To speed up content invalidation, Traffic Ops will not try to remove the content from the caches, but it makes the content inaccessible using the *regex\_revalidate* ATS plugin. This forces a *revalidation* of the content, rather than a new get.

---

**Note:** This method forces a HTTP *revalidation* of the content, and not a new *GET* - the origin needs to support revalidation according to the HTTP/1.2 specification, and send a 200 OK or 304 Not Modified as applicable.

---

Authentication Required: Yes

Role(s) Required: None

**Request Properties**

Parameter	Type	Description
dsId	string	Unique Delivery Service ID
regex	string	Path Regex this should be a <a href="#">PCRE</a> compatible regular expression for the path to match for forcing the revalidation. Be careful to only match on the content you need to remove - revalidation is an expensive operation for many origins, and a simple <code>/.*</code> can cause an overload condition of the origin.
startTime	string	Start Time is the time when the revalidation rule will be made active. Populate with the current time to schedule ASAP. This value cannot be more than 2 days before now.
ttl	int	Time To Live is how long the revalidation rule will be active for in hours. It usually makes sense to make this the same as the <code>Cache-Control</code> header from the origin which sets the object time to live in cache (by <code>max-age</code> or <code>Expires</code> ). Entering a longer TTL here will make the caches do unnecessary work.

### Request Example

```
{
  "dsId": "9999",
  "regex": "/path/to/content.jpg",
  "startTime": "2015-01-27 11:08:37",
  "ttl": 54
}
```

### Response Properties

Parameter	Type	Description
alerts	array	A collection of alert messages.
>level	string	Success, info, warning or error.
>text	string	Alert message.
version	string	

### Response Example

```
{
  "alerts":
    [
      {
        "level": "success",
        "text": "Successfully created purge job for: ."
      }
    ],
}
```

POST /api/1.2/user/login

Authentication of a user using username and password. Traffic Ops will send back a session cookie.

Authentication Required: No

Role(s) Required: None

**Request Properties**

Parameter	Type	Description
u	string	username
p	string	password

**Request Example**

```
{
  "u": "username",
  "p": "password"
}
```

**Response Properties**

Parameter	Type	Description
alerts	array	A collection of alert messages.
>level	string	Success, info, warning or error.
>text	string	Alert message.
version	string	

**Response Example**

```
{
  "alerts": [
    {
      "level": "success",
      "text": "Successfully logged in."
    }
  ],
}
```

**GET /api/1.2/user/:id/deliveryservices/available**

Authentication Required: Yes

Role(s) Required: None

**Request Route Parameters**

Name	Required	Description
id	yes	

### Response Properties

Parameter	Type	Description
id	string	
displayName	string	
xmlId	string	

### Response Example

```
{
  "response": [
    {
      "id": "90",
      "displayName": "Foo Bar DS",
      "xmlId": "foo-bar"
    },
    {
      "id": "92",
      "displayName": "Foo Baz DS",
      "xmlId": "foo-baz"
    }
  ],
}
```

### POST /api/1.2/user/login/token

Authentication of a user using a token.

Authentication Required: No

Role(s) Required: None

### Request Properties

Parameter	Type	Description
t	string	token-value

### Request Example

```
{
  "t": "token-value"
}
```

### Response Properties

Parameter	Type	Description
alerts	array	
>level	string	
>text	string	
version	string	

**Response Example**

```
{
  "alerts": [
    {
      "level": "error",
      "text": "Unauthorized, please log in."
    }
  ],
}
```

**POST /api/1.2/user/logout**

User logout. Invalidates the session cookie.

Authentication Required: Yes

Role(s) Required: None

**Response Properties**

Parameter	Type	Description
alerts	array	
• level	string	
• text	string	
version	string	

**Response Example**

```
{
  "alerts": [
    {
      "level": "success",
      "text": "You are logged out."
    }
  ],
}
```

**POST /api/1.2/user/reset\_password**

Reset user password.

Authentication Required: No

Role(s) Required: None

**Request Properties**

Parameter	Type	Description
email	string	The email address of the user to initiate password reset.

**Request Example**

```
{
  "email": "email@email.com"
}
```

**Response Properties**

Parameter	Type	Description
alerts	array	A collection of alert messages.
• level	string	Success, info, warning or error.
• text	string	Alert message.
version	string	

**Response Example**

```
{
  "alerts": [
    {
      "level": "success",
      "text": "Successfully sent password reset to email 'email@email.com'"
    }
  ],
  "version": "2.1-dev"
}
```

**Snapshot CRConfig**

/api/1.2/snapshot/{:cdn\_name}

GET /api/1.2/cdns/{:cdn\_name}/snapshot

Retrieves the CURRENT snapshot for a CDN which doesn't necessarily represent the current state of the CDN. The contents of this snapshot are currently used by Traffic Monitor and Traffic Router.

Authentication Required: Yes

Role(s) Required: Admin or Operations

#### Request Route Parameters

Name	Required	Description
cdn_name	yes	CDN name.

#### Response Properties

Parameter	Type	Description
config	hash	General CDN configuration settings.
contentRouters	hash	A list of Traffic Routers.
contentServers	hash	A list of Traffic Servers and the delivery services associated with each.
deliveryServices	hash	A list of delivery services.
edgeLocations	hash	A list of cache groups.
stats	hash	Snapshot properties.

#### Response Example

```
{
  "response": {
    "config": { ... },
    "contentRouters": { ... },
    "contentServers": { ... },
    "deliveryServices": { ... },
    "edgeLocations": { ... },
    "stats": { ... },
  },
}
```

#### GET /api/1.2/cdns/{:cdn\_name}/snapshot/new

Retrieves a PENDING snapshot for a CDN which represents the current state of the CDN. The contents of this snapshot are NOT currently used by Traffic Monitor and Traffic Router. Once a snapshot is performed, this snapshot will become the CURRENT snapshot and will be used by Traffic Monitor and Traffic Router.

Authentication Required: Yes

Role(s) Required: Admin or Operations

#### Request Route Parameters

Name	Required	Description
cdn_name	yes	CDN name.

#### Response Properties



Parameter	Type	Description
config	hash	General CDN configuration settings.
contentRouters	hash	A list of Traffic Routers.
contentServers	hash	A list of Traffic Servers and the delivery services associated with each.
deliveryServices	hash	A list of delivery services.
edgeLocations	hash	A list of cache groups.
stats	hash	Snapshot properties.

**Response Example**

```
{
  "response": {
    "config": { ... },
    "contentRouters": { ... },
    "contentServers": { ... },
    "deliveryServices": { ... },
    "edgeLocations": { ... },
    "stats": { ... },
  },
}
```

**PUT /api/1.2/snapshot/{:cdn\_name}**

Authentication Required: Yes

Role(s) Required: Admin or Operations

**Request Route Parameters**

Name	Required	Description
cdn_name	yes	The name of the cdn to snapshot configure

**Response Properties**

Parameter	Type	Description
response	string	"SUCCESS"

**Response Example**

```
{
  "response": "SUCCESS"
}
```

### 4.1.3 Traffic Portal

#### Introduction

Traffic Portal is an [AngularJS 1.x](#) client served from a [Node.js](#) web server designed to consume the Traffic Ops 1.x API. The Traffic Portal replaces the Traffic Ops UI.

#### Software Requirements

To work on Traffic Portal you need a \*nix (MacOS and Linux are most commonly used) environment that has the following installed:

- [Ruby Devel 2.0.x](#) or above
- [Compass 1.0.x](#) or above
- [Node.js 6.0.x](#) or above
- [Bower 1.7.9](#) or above
- [Grunt CLI 1.2.0](#) or above
- Access to a working instance of Traffic Ops

#### Traffic Portal Project Tree Overview

- **traffic\_control/traffic\_portal/app/src** - contains HTML, JavaScript and Sass source files.

#### Installing The Traffic Portal Developer Environment

- Clone the traffic\_control repository
- Navigate to the traffic\_control/traffic\_portal of your cloned repository.
- Run `npm install` to install application dependencies into traffic\_portal/node\_modules. Only needs to be done the first time unless traffic\_portal/package.json changes.
- Run `bower install` to install client-side dependencies into traffic\_portal/app/bower\_components. Only needs to be done the first time unless traffic\_portal/bower.json changes.
- Run `grunt` to package the application into traffic\_portal/app/dist, start a local https server (Express), and start a file watcher.
- Navigate to <https://localhost:8443>

#### Notes

- The Traffic Portal consumes the Traffic Ops API. By default, Traffic Portal assumes Traffic Ops is running on <https://localhost:8444>. Temporarily modify traffic\_portal/conf/config.js if you need to change the location of Traffic Ops.

## 4.1.4 Traffic Router

### Introduction

Traffic Router is a Java Tomcat application that routes clients to the closest available cache on the CDN using both HTTP and DNS. Cache availability is determined by Traffic Monitor; consequently Traffic Router polls Traffic Monitor for its configuration and cache health state information, and uses this data to make routing decisions. HTTP routing is performed by localizing the client based on the request's source IP address (IPv4 or IPv6), and issues an HTTP 302 redirect to the nearest cache. HTTP routing utilizes consistent hashing on request URLs to optimize cache performance and request distribution. DNS routing is performed by localizing clients, resolvers in most cases, requesting A and AAAA records for a configurable name such as `foo.deliveryservice.somecdn.net`. Traffic Router is comprised of seven separate Maven modules:

- `shared` - A reusable utility JAR for defining Delivery Service Certificates
- `configuration` - A reusable JAR defining the `ConfigurationListener` interface
- `connector` - A JAR that overrides Tomcat's standard `Http11Protocol Connector` class and allows Traffic Router to delay opening listen sockets until it is in a state suitable for routing traffic
- `geolocation` - Submodule for defining geolocation services
- `neustar` - A Jar that provides a bean "neustarGeolocationService" that implements the `GeolocationService` interface defined in the `geolocation` maven submodule, which can optionally be added to the build of Traffic Router
- `core` - Services DNS and HTTP requests, performs localization on routing requests, and is deployed as a WAR to a Service (read: connector/listen port) within Tomcat which is separate from api
- `build` - A simple Maven project which gathers the artifacts from the modules and builds an RPM

### Software Requirements

To work on Traffic Router you need a \*nix (MacOS and Linux are most commonly used) environment that has the following installed:

- Eclipse >= Kepler SR2 (or another Java IDE)
- Maven >= 3.3.1
- JDK >= 8.0

### Traffic Router Project Tree Overview

- `traffic_control/traffic_traffic_router/` - base directory for Traffic Router
  - `connector/` - Source code for Traffic Router Connector;
    - \* `src/main/java` - Java source directory for Traffic Router Connector
  - `core/` - Source code for Traffic Router Core, which is built as its own deployable WAR file and communicates with Traffic Router API using JMX
    - \* `src/main` - Main source directory for Traffic Router Core
      - `etc/init.d` - Init script for Tomcat
      - `conf/` - Minimum required configuration files
      - `java/` - Java source code for Traffic Router Core
      - `opt/tomcat/conf` - Contains Tomcat configuration file(s) pulled in during an RPM build

- resources/ - Resources pulled in during an RPM build
- scripts/ - Scripts used by the RPM build process
- webapp/ - Java webapp resources
- \* src/test - Test source directory for Traffic Router Core
  - conf/ - Minimal Configuration files that make it possible to run junit tests
  - db/ - Files downloaded by unit tests
  - java/ - JUnit based unit tests for Traffic Router Core
  - resources/ - Example data files used by junit tests
  - var/auto-zones - BIND formatted zone files generated by Traffic Router Core during unit testing

## Java Formatting Conventions

None at this time. The codebase will eventually be formatted per Java standards.

## Installing The Developer Environment

To install the Traffic Router Developer environment:

1. Clone the traffic\_control repository using Git.
2. Change directories into traffic\_control/traffic\_router.
3. Follow the instructions in “README.DNSSEC” for DNSSEC support.
4. Set the environment variable TRAFFIC\_MONITOR\_HOSTS to be a semicolon delimited list of Traffic Monitors that can be accessed during integration tests
5. Additional configuration is set using the below files:
  - core/src/test/conf/dns.properties - copy from core/src/main/conf
  - core/src/test/conf/http.properties - copy from core/src/main/conf
  - core/src/test/conf/log4j.properties - copy from core/src/main/conf
  - core/src/test/conf/traffic\_monitor.properties - copy from core/src/main/conf
  - core/src/test/conf/traffic\_ops.properties file holds the credentials for accessing Traffic Ops. - copy from core/src/main/conf
  - Default configuration values now reside in core/src/main/webapp/WEB-INF/applicationContext.xml
  - The above values may be overridden by creating and/or modifying the property files listed in core/src/main/resources/applicationProperties.xml
  - Pre-existing properties files are still honored by Traffic Router. For example traffic\_monitor.properties:

Parameter	Value
traffic_monitor.bootstrap.hosts	FQDN and port of the Traffic Monitor instance(s), separated by semicolons as necessary (do not include <a href="http://">http://</a> ).

6. Import the existing git repo into Eclipse:

- a. File -> Import -> Git -> Projects from Git; Next
  - b. Existing local repository; Next
  - c. Add -> browse to find `traffic_control`; Open
  - d. Select `traffic_control`; Next
  - e. Ensure “Import existing projects” is selected, expand `traffic_control`, select `traffic_router`; Next
  - f. Ensure `traffic_router_api`, `traffic_router_connector`, and `traffic_router_core` are checked; Finish (this step can take several minutes to complete)
  - g. Ensure `traffic_router_api`, `traffic_router_connector`, and `traffic_router_core` have been opened by Eclipse after importing
7. From the terminal, run `mvn clean verify` from the `traffic_router` directory
  8. Start the embedded Tomcat instance for Core from within your IDE by following these steps:
    - a. In the package explorer, expand `traffic_router_core`
    - b. Expand `src/test/java`
    - c. Expand the package `com.comcast.cdn.traffic_control.traffic_router.core`
    - d. Open and run `TrafficRouterStart.java`

---

**Note:** If an error is displayed in the Console, run `mvn clean verify` from the `traffic_router` directory

---

9. Traffic Router Core should now be running; the Traffic Router API is available at <http://localhost:3333>, the HTTP routing interface is available on <http://localhost:8888> and HTTPS is available on <http://localhost:8443>. The DNS server and routing interface is available on `localhost:1053` via TCP and UDP.

## Manual Testing

Look up the URL for your test ‘http’ Delivery Service in Traffic Ops and then:

```
curl -vs -H "Host: <Delivery Service URL>" http://localhost:8888/x
```

## Test Cases

- Unit tests can be executed using Maven by running `mvn test` at the root of the `traffic_router` project.
- Unit and Integration tests can be executed using Maven by running `mvn verify` at the root of the `traffic_router` project.

## RPM Packaging

Running `mvn package` on a Linux based distribution will trigger the build process to create the Traffic Router rpm.

## API

*Traffic Router API*

## Traffic Router API

### **/crs/stats**

General stats.

### **/crs/stats/ip/:ipaddress**

Geolocation information for an IPv4 or IPv6 address.

### **/crs/locations**

A list of configured cache groups.

### **/crs/locations/caches**

A mapping of caches to cache groups and their current health state.

### **/crs/locations/:location/caches**

A list of caches for this cache group only.

## 4.1.5 Traffic Monitor

### Introduction

Traffic Monitor is a Java Tomcat application that monitors caches, provides health state information to Traffic Router, and collects statistics for use in tools such as Traffic Ops and Traffic Stats. The health state provided by Traffic Monitor is used by Traffic Router to control which caches are available on the CDN.

### Software Requirements

To work on Traffic Monitor you need a \*nix (MacOS and Linux are most commonly used) environment that has the following installed:

- Eclipse >= Kepler SR2 (or another Java IDE)
- Maven >= 3.3.1
- JDK >= 6.0

## Traffic Monitor Project Tree Overview

- `traffic_control/traffic_monitor/` - base directory for Traffic Monitor
  - `etc/` - Miscellaneous simulator utilities
  - `src/main` - Main source directory for the Traffic Monitor
    - \* `bin/` - Configuration tools
    - \* `conf/` - Configuration files
    - \* `java/` - Java source code for Traffic Monitor
    - \* `opt/tomcat/conf` - Contains Tomcat configuration file(s) pulled in during an RPM build
    - \* `resources/` - Resources pulled in during an RPM build
    - \* `scripts/` - Scripts used by the RPM build process
    - \* `webapp/` - Java webapp resources
  - `src/test` - Test source directory for Traffic Monitor
    - \* `java/` - JUnit based unit tests for Traffic Monitor
    - \* `resources/conf` - Configuration files used by unit tests
    - \* `resources/db` - Files downloaded by unit tests
    - \* `resources/var` - Files generated by unit tests

## Java Formatting Conventions

None at this time. The codebase will eventually be formatted per Java standards.

## Installing The Developer Environment

To install the Traffic Monitor Developer environment:

1. Clone the `traffic_control` repository using Git.
2. Change directories into `traffic_control/traffic_monitor`.
3. Edit the following parameters in `src/test/resources/conf/traffic_monitor_config.js`:

Parameter	Value
<code>tm.hostname</code>	FQDN of the Traffic Ops instance (do not include <a href="http://">http://</a> ).
<code>tm.username</code>	Admin username for Traffic Ops
<code>tm.password</code>	Password for admin user
<code>cdnName</code>	Name of the CDN this Traffic Monitor will monitor

Note: any change done later in the configuration file requires a `mvn` build in order to be applied.

4. Import the existing git repo into Eclipse:
  - a. File -> Import -> Git -> Projects from Git; Next
  - b. Existing local repository; Next
  - c. Add -> browse to find `traffic_control`; Add
  - d. Select `traffic_control`; Next

- e. Ensure “Import existing projects” is selected, expand `traffic_control`, select `traffic_monitor`; Next
  - f. Ensure `traffic_monitor` is checked; Finish
  - g. Ensure `traffic_monitor` has been opened by Eclipse after importing
5. Run `mvn clean verify` from the `traffic_monitor` directory
6. Start the embedded Jetty instance from within Eclipse
  - a. In the package explorer, expand `traffic_monitor`
  - b. Expand `src/test/java`
  - c. Expand the package `com.comcast.cdn.traffic_control.traffic_monitor`
  - d. Open and run `Start.java`

---

**Note:** If an error is displayed in the Console, run `mvn clean verify` from the `traffic_monitor` directory

---

- e. With a web browser, navigate to <http://localhost:8080>

## Test Cases

Unit tests can be executed using Maven by running `mvn test` at the root of the `traffic_monitor` project.

## API

### *Traffic Monitor APIs*

#### Traffic Monitor APIs

The Traffic Monitor URLs below allow certain query parameters for use in controlling the data returned. The optional query parameters are the *tabbed* in values under each URL, if they exist.

##### **/publish/EventLog**

Log of recent events.

##### **/publish/CacheStats**

Statistics gathered for each cache.

##### **Query Parameters**



Parameter	Type	Description
hc	int	The history count, number of items to display.
stats	string	A comma separated list of stats to display.
wildcard	boolean	Controls whether specified stats should be treated as partial strings.

**/publish/CacheStats/:cache**

Statistics gathered for only this cache.

**Query Parameters**

Parameter	Type	Description
hc	int	The history count, number of items to display.
stats	string	A comma separated list of stats to display.
wildcard	boolean	Controls whether specified stats should be treated as partial strings.

**/publish/DsStats**

Statistics gathered for delivery services.

**Query Parameters**

Parameter	Type	Description
hc	int	The history count, number of items to display.
stats	string	A comma separated list of stats to display.
wildcard	boolean	Controls whether specified stats should be treated as partial strings.

**/publish/DsStats/:deliveryService**

Statistics gathered for this delivery service only.

**Query Parameters**

Parameter	Type	Description
hc	int	The history count, number of items to display.
stats	string	A comma separated list of stats to display.
wildcard	boolean	Controls whether specified stats should be treated as partial strings.

### **/publish/CrStates**

The current state of this CDN per the health protocol.

### **raw**

The current state of this CDN per this Traffic Monitor only.

### **/publish/CrConfig**

The CrConfig served to and consumed by Traffic Router.

### **/publish/PeerStates**

The health state information from all peer Traffic Monitors.

#### **Query Parameters**

Parameter	Type	Description
hc	int	The history count, number of items to display.
stats	string	A comma separated list of stats to display.
wildcard	boolean	Controls whether specified stats should be treated as partial strings.

### **/publish/Stats**

The general statistics about Traffic Monitor.

### **/publish/StatSummary**

The summary of cache statistics.

#### **Query Parameters**

Parameter	Type	Description
startTime	number	Window start. The number of milliseconds since the epoch.
endTime	number	Window end. The number of milliseconds since the epoch.
hc	int	The history count, number of items to display.
stats	string	A comma separated list of stats to display.
wildcard	boolean	Controls whether specified stats should be treated as partial strings.
cache	string	Summary statistics for just this cache.

## **/publish/ConfigDoc**

The overview of configuration options.

## **4.1.6 Traffic Monitor Golang**

### **Introduction**

The next major version of Traffic Monitor has been completely rewritten in Golang. Currently, this version is functionally equivalent, and should be considered “beta.” It is recommended that new CDN deployments continue to use the existing Java version, until the new version is completely moved over in the source and binary distributions. However, developers and administrators are encouraged to test the Golang version, to prepare for operational differences and look for bugs.

Traffic Monitor is an HTTP service application that monitors caches, provides health state information to Traffic Router, and collects statistics for use in tools such as Traffic Ops and Traffic Stats. The health state provided by Traffic Monitor is used by Traffic Router to control which caches are available on the CDN.

### **Software Requirements**

To work on Traffic Monitor you need a \*nix (MacOS and Linux are most commonly used) environment that has the following installed:

- Golang

### **Project Tree Overview**

- `traffic_control/traffic_monitor/` - base directory for Traffic Monitor.
- `cache/` - Handler for processing cache results.
- `config/` - Application configuration; in-memory objects from `traffic_monitor.cfg`.
- `crconfig/` - struct for deserializing the CRConfig from JSON.
- `deliveryservice/` - aggregates delivery service data from cache results.
- `deliveryservicedata/` - deliveryservice structs. This exists separate from `deliveryservice` to avoid circular dependencies.
- `enum/` - enumerations and name alias types.
- `health/` - functions for calculating cache health, and creating health event objects.

- **manager/ - manager goroutines (microthreads).**
  - `health.go` - Health request manager. Processes health results, from the health poller -> fetcher -> manager. The health poll is the “heartbeat” containing a small amount of stats, primarily to determine whether a cache is reachable as quickly as possible. Data is aggregated and inserted into shared threadsafe objects.
  - `manager.go` - Contains `Start` function to start all pollers, handlers, and managers.
  - `monitorconfig.go` - Monitor config manager. Gets data from the monitor config poller, which polls Traffic Ops for changes to which caches are monitored and how.
  - `opsconfig.go` - Ops config manager. Gets data from the ops config poller, which polls Traffic Ops for changes to monitoring settings.
  - `peer.go` - Peer manager. Gets data from the peer poller -> fetcher -> handler and aggregates it into the shared threadsafe objects.
  - `stat.go` - Stat request manager. Processes stat results, from the stat poller -> fetcher -> manager. The stat poll is the large statistics poll, containing all stats (such as HTTP codes, transactions, delivery service statistics, and more). Data is aggregated and inserted into shared threadsafe objects.
  - `statecombiner.go` - Manager for combining local and peer states, into a single combined states threadsafe object, for serving the `CrStates` endpoint.
- `datareq/` - HTTP routing, which has threadsafe health and stat objects populated by stat and health managers.
- `peer/` - Manager for getting and populating peer data from other Traffic Monitors
- `srvhttp/` - HTTP service. Given a map of endpoint functions, which are lambda closures containing aggregated data objects.
- `static/` - Web GUI HTML and javascript files
- `threadsafe/` - Threadsafes objects for storing aggregated data needed by multiple goroutines (typically the aggregator and HTTP server)
- `trafficopsdata/` - Struct for fetching and storing Traffic Ops data needed from the `CRConfig`. This is primarily mappings, such as delivery service servers, and server types.
- `trafficopswrapper/` - Threadsafes wrapper around the Traffic Ops client. The client used to not be threadsafe, however, it mostly (possibly entirely) is now. But, the wrapper also serves to overwrite the Traffic Ops `monitoring.json` values, which are live, with snapshotted `CRConfig` values.

## Architecture

At the highest level, Traffic Monitor polls caches, aggregates their data and availability, and serves it at HTTP JSON endpoints.

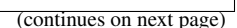
In the code, the data flows thru microthread (goroutine) pipelines. All stages of the pipeline are independent running microthreads<sup>1</sup>. The pipelines are:

- **stat poll** - polls caches for all statistics data. This should be a slower poll, which gets a lot of data.
- **health poll** - polls caches for a tiny amount of data, typically system information. This poll is designed to be a heartbeat, determining quickly whether the cache is reachable. Since it’s a small amount of data, it should poll more frequently.
- **peer poll** - polls Traffic Monitor peers for their availability data, and aggregates it with its own availability results and that of all other peers.

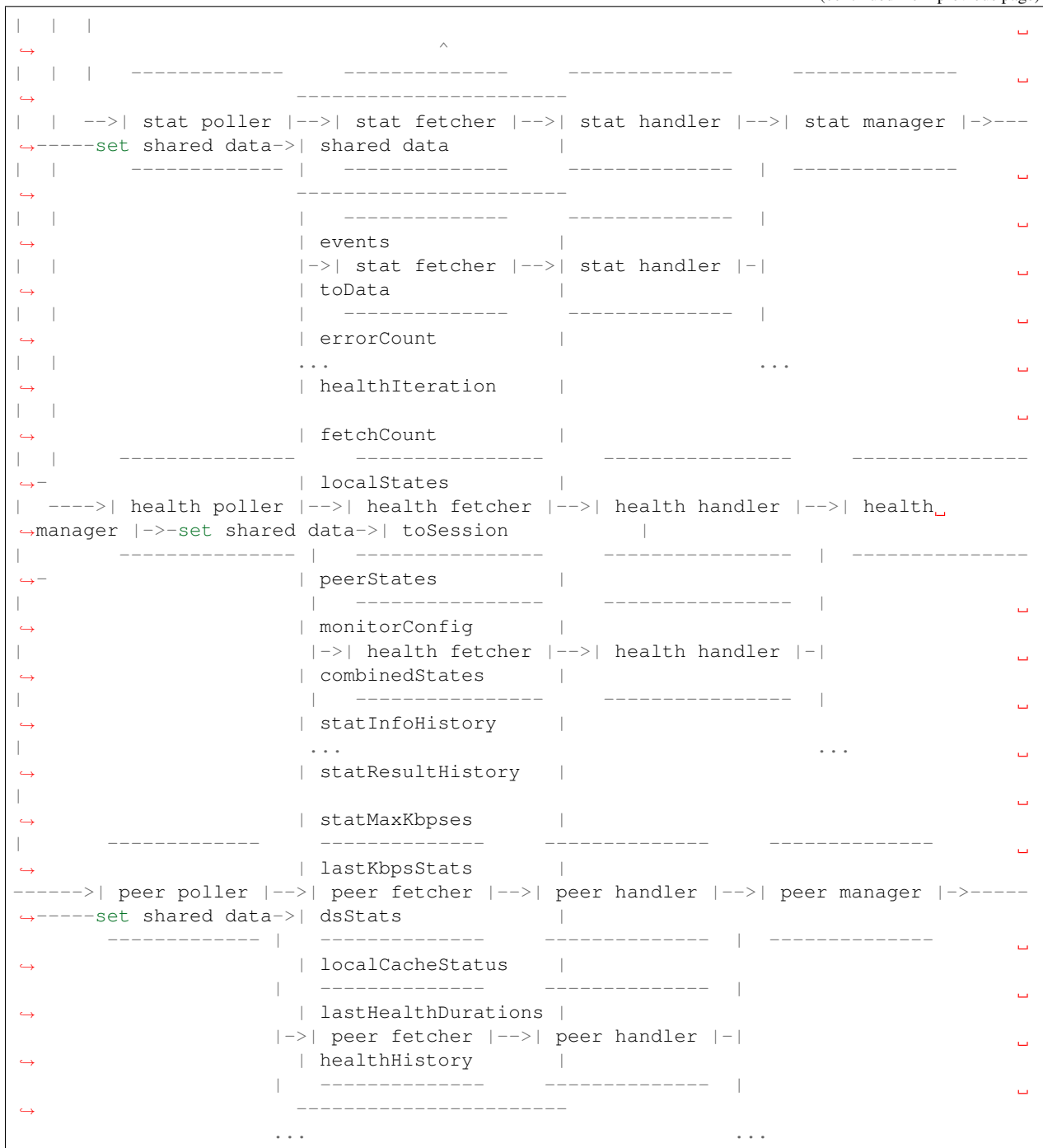
---

<sup>1</sup> Technically, some stages which are one-to-one simply call the next stage as a function. For example, the Fetcher calls the Handler as a function in the same microthread. But this isn’t architecturally significant.

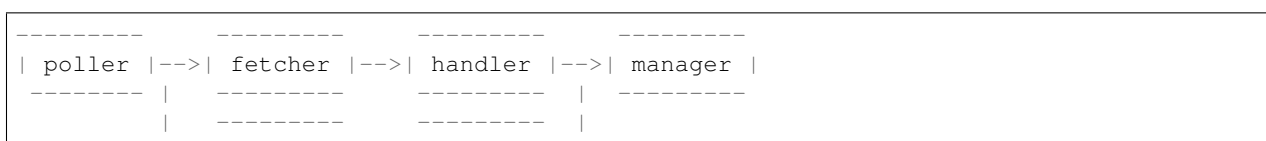
- All microthreads in the pipeline are started by `manager/manager.go:Start()`.



(continued from previous page)



## Stat Pipeline



(continues on next page)

(continued from previous page)

```

|->| fetcher |-->| handler |-|
|   -----   -----   |
...                               ...

```

- **poller** - `common/poller/poller.go:HttpPoller.Poll()`. Listens for config changes (from the ops config manager), and starts its own internal microthreads, one for each cache to poll. These internal microthreads call the Fetcher at each cache's poll interval.
- **fetcher** - `common/fetcher/fetcher.go:HttpFetcher.Fetch()`. Fetches the given URL, and passes the returned data to the Handler, along with any errors.
- **handler** - `traffic_monitor/cache/cache.go:Handler.Handle()`. Takes the given result and does all data computation possible with the single result. Currently, this computation primarily involves processing the denormalized ATS data into Go structs, and processing System data into OutBytes, Kbps, etc. Precomputed data is then passed to its result channel, which is picked up by the Manager.
- **manager** - `traffic_monitor/manager/stat.go:StartStatHistoryManager()`. Takes preprocessed results, and aggregates them. Aggregated results are then placed in shared data structures. The major data aggregated are delivery service statistics, and cache availability data. See [Aggregated Stat Data](#) and [Aggregated Availability Data](#).

## Health Pipeline

```

-----
| poller |-->| fetcher |-->| handler |-->| manager |
-----
|   -----   -----   |
|   -----   -----   |
|->| fetcher |-->| handler |-|
|   -----   -----   |
...                               ...

```

- **poller** - `common/poller/poller.go:HttpPoller.Poll()`. Same poller type as the Stat Poller pipeline, with a different handler object.
- **fetcher** - `common/fetcher/fetcher.go:HttpFetcher.Fetch()`. Same fetcher type as the Stat Poller pipeline, with a different handler object.
- **handler** - `traffic_monitor/cache/cache.go:Handler.Handle()`. Same handler type as the Stat Poller pipeline, but constructed with a flag to not precompute. The health endpoint is of the same form as the stat endpoint, but doesn't return all stat data. So, it doesn't precompute like the Stat Handler, but only processes the system data, and passes the processed result to its result channel, which is picked up by the Manager.
- **manager** - `traffic_monitor/manager/health.go:StartHealthResultManager()`. Takes preprocessed results, and aggregates them. For the Health pipeline, only health availability data is aggregated. Aggregated results are then placed in shared data structures (`lastHealthDurationsThreadsafe`, `lastHealthEndTimes`, etc). See [Aggregated Availability Data](#).

## Peer Pipeline

```

-----
| poller |-->| fetcher |-->| handler |-->| manager |
-----
|   -----   -----   |
|   -----   -----   |
|->| fetcher |-->| handler |-|

```

(continues on next page)

(continued from previous page)

	-----	-----	
...			...

- **poller** - `common/poller/poller.go:HttpPoller.Poll()`. Same poller type as the Stat and Health Poller pipelines, with a different handler object. Its config changes come from the Monitor Config Manager, and it starts an internal microthread for each peer to poll.
- **fetcher** - `common/fetcher/fetcher.go:HttpFetcher.Fetch()`. Same fetcher type as the Stat and Health Poller pipeline, with a different handler object.
- **handler** - `traffic_monitor/cache/peer.go:Handler.Handle()`. Decodes the JSON result into an object, and without further processing passes to its result channel, which is picked up by the Manager.
- **manager** - `traffic_monitor/manager/peer.go:StartPeerManager()`. Takes JSON peer Traffic Monitor results, and aggregates them. The availability of the Peer Traffic Monitor itself, as well as all cache availability from the given peer result, is stored in the shared `peerStates` object. Results are then aggregated via a call to the `combineState()` lambda, which signals the State Combiner microthread (which stores the combined availability in the shared object `combinedStates`; See [State Combiner](#)).

## Monitor Config Pipeline

-----	-----	
poller  -->	manager  -->	stat subscriber (Stat pipeline Poller)
-----	-----	
		-> health subscriber (Health pipeline Poller)
		--> peer subscriber (Peer pipeline Poller)

- **poller** - `common/poller/poller.go:MonitorConfigPoller.Poll()`. The Monitor Config poller, on its interval, polls Traffic Ops for the Monitor configuration, and writes the polled value to its result channel, which is read by the Manager.
- **manager** - `traffic_monitor/manager/monitorconfig.go:StartMonitorConfigManager()`. Listens for results from the poller, and processes them. Cache changes are written to channels read by the Health, Stat, and Peer pollers. In the Shared Data objects, this also sets the list of new delivery services and removes ones which no longer exist, and sets the list of peer Traffic Monitors.

## Ops Config Pipeline

-----	-----	-----
poller  -->	handler  -->	manager  -->  ops config change subscriber (Monitor_
		↪Config Poller)
-----	-----	-----
		--> Traffic ops client change subscriber_
		↪(Monitor Config Poller)

- **poller** - `common/poller/poller.go:FilePoller.Poll()`. Polls for changes to the Traffic Ops config file `traffic_ops.cfg`, and writes the changed config to its result channel, which is read by the Handler.
- **handler** - `common/handler/handler.go:OpsConfigFileHandler.Listen()`. Takes the given raw config, unmarshalls the JSON into an object, and writes the object to its channel, which is read by the Manager, along with any error.



- **manager**-`traffic_monitor/manager/monitorconfig.go:StartMonitorConfigManager()`. Listens for new configs, and processes them. When a new config is received, a new HTTP dispatch map is created via `traffic_monitor/datareq/datareq.go:MakeDispatchMap()`, and the HTTP server is restarted with the new dispatch map. The Traffic Ops client is also recreated, and stored in its shared data object. The Ops Config change subscribers and Traffic Ops Client change subscribers (the Monitor Config poller) are also passed the new ops config and new Traffic Ops client.

## Events

The `events` shared data object is passed to each pipeline microthread which needs to signal events. Most of them do. Events are then logged, and visible in the UI as well as an HTTP JSON endpoint. Most events are caches becoming available or unavailable, but include other things such as peer availability changes.

## State Combiner

The State Combiner is a microthread started in `traffic_monitor/manager/manager.go:Start()` via `traffic_monitor/manager/statecombiner.go:StartStateCombiner()`, which listens for signals to combine states. It should be signaled by any pipeline which updates the local or peer availability shared data objects, `localStates` and `peerStates`. It holds the threadsafe shared data objects for local states and peer states, so no data is passed or returned, only a signal.

When a signal is received, it combines the local and peer states optimistically. That is, if a cache is marked available locally or by any peer, that cache is marked available in the combined states. There exists a variable to combine pessimistically, which may be set at compile time (it's unusual for a CDN to operate well with pessimistic cache availability). Combined data is stored in the threadsafe shared data object `combinedStates`.

## Aggregated Stat Data

The Stat pipeline Manager is responsible for aggregating stats from all caches, into delivery services statistics. This is done via a call to `traffic_monitor/deliveryservice/stat.go:CreateStats()`.

## Aggregated Availability Data

Both the Stat and Health pipelines aggregate availability data received from caches. This is done via a call to `traffic_monitor/deliveryservice/health.go:CalcAvailability()` followed by a call to `combineState()`. The `CalcAvailability` function calculates the availability of each cache from the result of polling it, that is, local availability. The `combineState()` function is a lambda passed to the Manager, which signals the State Combiner microthread, which will combine the local and peer Traffic Monitor availability data, and insert it into the shared data `combinedStates` object.

## HTTP Data Requests

Data is provided to HTTP requests via the threadsafe shared data objects (see [Shared Data](#)). These objects are closed in lambdas created via `traffic_monitor/datareq/datareq.go:MakeDispatchMap()`. This is called by the Ops Config Manager when it recreates the HTTP server.

Each HTTP endpoint is mapped to a function which closes around the shared data objects it needs, and takes the request data it needs (such as query parameters). Each endpoint function resides in its own file in `traffic_monitor/datareq/`. Because each Go HTTP routing function must be a `http.HandlerFunc`, wrapper functions take the

endpoint functions and return `http.HandlerFunc` functions which call them, and which are stored in the dispatch map, to be registered with the HTTP server.

### Shared Data

Processed and aggregated data must be shared between the end of the stat and health processing pipelines, and HTTP requests. The CSP paradigm of idiomatic Go does not work efficiently with storing and sharing state. While not idiomatic Go, shared mutexed data structures are faster and simpler than CSP manager microthreads for each data object.

Traffic Monitor has many threadsafe shared data types and objects. All shared data objects can be seen in `manager/manager.go:Start()`, where they are created and passed to the various pipeline stage microthreads that need them. Their respective types all include the word `Threadsafe`, and can be found in `traffic_monitor/threadsafe/` as well as, for dependency reasons, various appropriate directories.

Currently, all `Threadsafe` shared data types use mutexes. In the future, these could be changed to lock-free or wait-free structures, if the performance needs outweighed the readability and correctness costs. They could also easily be changed to internally be manager microthreads and channels, if being idiomatic were deemed more important than readability or performance.

### Formatting Conventions

Go code should be formatted with `gofmt`. See also `CONTRIBUTING.md`.

### Installing The Developer Environment

To install the Traffic Monitor Developer environment:

1. Install `go` version 1.7 or greater, from <https://golang.org/doc/install> and <https://golang.org/doc/code.html>
2. Clone the `traffic_control` repository using Git, into `$GOPATH/src/github.com/apache/incubator-trafficcontrol`
3. Change directories into `$GOPATH/src/github.com/apache/incubator-trafficcontrol/traffic_monitor_golang/traffic_monitor`
4. Run `./build.sh`

### Test Cases

Tests can be executed by running `go test ./...` at the root of the `traffic_monitor_golang` project.

### API

*Traffic Monitor APIs*

#### 4.1.7 Traffic Stats

##### Introduction

Traffic Stats is a utility written in [Go](#) that is used to acquire and store statistics about CDNs controlled by Traffic Control. Traffic Stats mines metrics from Traffic Monitor's JSON APIs and stores the data in [InfluxDb](#). Data is

typically stored in InfluxDb on a short-term basis (30 days or less) and is used to drive graphs created by [Grafana](#) which are linked from Traffic Ops. Traffic Stats also calculates daily statistics from InfluxDb and stores them in the Traffic Ops database.

## Software Requirements

To work on Traffic Stats you need a \*nix (MacOS and Linux are most commonly used) environment that has the following installed:

- [Go 1.7.x](#) or above
- Access to a working instance of Traffic Ops
- Access to a working instance of Traffic Monitor
- [InfluxDb version 1.0.0](#) or greater

## Traffic Stats Project Tree Overview

- **traffic\_control/traffic\_stats** - contains Go source files and Files used to create the Traffic Stats rpm.
- **traffic\_control/traffic\_stats/grafana/** - contains a javascript file which is installed on the grafana server. This allows Traffic Ops to create custom dashboards for Delivery Services, Caches, etc.
- **traffic\_control/traffic\_stats/influxdb\_tools/** - contains one tool to create the databases and retention policies needed by Traffic Stats as well as continuous queries to downsample data; contains another tool to sync down-sampled data between influxdb instances. This is helpful if you have multiple instances and they get out of sync with data.

## Go Formatting Conventions

In general [Go fmt](#) is the standard for formatting go code. It is also recommended to use [Go lint](#).

## Installing The Developer Environment

To install the Traffic Ops Developer environment:

- Clone the traffic\_control repository using Git into a location accessible by your \$GOPATH
- Navigate to the traffic\_ops/client directory of your cloned repository. (This is the directory containing Traffic Ops client code used by Traffic Stats)
- From the traffic\_ops/client directory run `go test` to test the client code. This will run all unit tests for the client and return the results. If there are missing dependencies you will need to run `go get <dependency name>` to get the dependency
- Once the tests pass, run `go install` to build and install the Traffic Ops client package. This makes it accessible to Traffic Stats.
- Navigate to your cloned repo under Traffic Stats
- Run `go build traffic_stats.go` to build traffic\_stats. You will need to run `go get` for any missing dependencies.

## Test Cases

Currently there are no automated tests for Traffic Stats :( but pull requests are always welcome to fix this problem!

### 4.1.8 Traffic Server

See the [Apache Traffic Server documentation](#).

## 5.1 FAQ

Table of Contents:

### 5.1.1 General

#### Who is using Traffic Control?

**Comcast Cable** Comcast is the original developer of Traffic Control and is using it for all it's IP video delivery, delivering images and software to it's X1 platform, and for [delivering third party content](#) to it's footprint.

**Cox Communications**

**Cisco** Cisco has a product called [Open Media Distribution](#) that is based on Traffic Control.

**Concurrent** Concurrent has a product that uses Traffic Control, see their [github page](#) for more info.

**Augere Pakistan / QUBEE**

#### How do I get help with Traffic Control?

Hop on to our Slack Channel by filling out [this form](#), or send a question to our mailing list [here](#).

#### What is Rascal?

Rascal was the original name for Traffic Monitor. You will sometimes still see this name in the source, or in older documents.

## What is the CCR?

Comcast Content Router was the original name for Traffic Router. You will sometimes still see this name in the source, or in older documents.

## What is Twelve Monkeys?

Twelve Monkeys was the the original internal name for Traffic Ops. You will sometimes still see this name in the source, or in older documents. It's also a good movie.

## What license is Traffic Control released under?

See the [LICENSE](#) file

## 5.1.2 Development

### How can I become involved?

See our [CONTRIBUTING](#) page.

## 5.1.3 Running a Traffic Control CDN

### Why is my CRConfig.json rejected?

Especially in version 1.1.0, there's a number of manual steps that need to be done after the initial install. Make sure that after the initial install, you perform these steps in order:

---

**Note:** Even though Traffic Ops allows you to enter the servers with no IPv6 address information, the CRConfig will not be accepted by Traffic Router without IPv6 address information for at least Traffic Router and Traffic Monitor. Traffic Control assumes in a lot of places that all servers have at least an IPv4 and an IPv6 address. If you are not using IPv6, it is best to enter dummy addresses for all server types, and turn IPv6 off in all delivery services. ([https://github.com/Comcast/traffic\\_control/issues/44](https://github.com/Comcast/traffic_control/issues/44)).

---

- **Add users** Not necessarily needed for getting your CRConfig accepted, but always a good idea.
- **Add Divisions** You will need at least one.
- **Add Regions** You will need at least one.
- **Add Physical Locations** You will need at least one.
- **Add Mid tier Cache Groups** You will need at least one.
- **Add Edge tier Cache Groups** You will need at least one.
- **Add Traffic Monitors** You will need to enter at least one Traffic Monitor - make sure to change the server status to *ONLINE*.
- **Add Traffic Routers** You will need to enter at least one Traffic Router - make sure to change the server status to *ONLINE*.
- **Add Edges** You will need at least one edge cache to make Traffic Router accept the CRConfig.

- **Add Mid** Technically you don't need a mid tier, but if you have one, best to enter the info before continuing.
- **Change the `polling.url` parameters to reflect your CDN** Set where to get the coverage zone map, and the geo IP database.
- **Create at least one delivery service, and assign at least one edge cache in `REPORTED` state to it.** Even if it is a dummy DS, without a single DS, the CRConfig will not be accepted by Traffic Router.
- **Snapshot CRConfig** Tools > Snapshot CRConfig diff, and write.

Now you are ready to install the sw on Traffic Monitor and then Traffic Router.





## 6.1 Glossary

**302 content routing** *HTTP Content Routing*.

**astats (stats\_over\_http)** An ATS plugin that allows you to monitor vitals of the ATS server. See *Cache Monitoring*.

**cache** A caching proxy server. See *Caching Proxies*.

**cachegroup** A group of caches that together create a combined larger cache using consistent hashing. See *Cache Group*.

**consistent hashing** See [the Wikipedia article](#); Traffic Control uses consistent hashing when using *HTTP Content Routing* for the edge tier and when selecting parents in the mid tier.

**content routing** Directing clients (or client systems) to a particular location or device in a location for optimal delivery of content See also *HTTP Content Routing* and *DNS Content Routing*.

**coverage zone map** The coverage zone map (czm) or coverage zone file (zcf) is a file that maps network prefixes to cachegroups. See *Localization*.

**delivery service** A grouping of content in the CDN, usually a determined by the URL hostname. See *Delivery Service*.

**edge (tier or cache)** Closest to the client or end-user. The edge tier is the tier that serves the client, edge caches are caches in the edge tier. In a Traffic Control CDN the basic function of the edge cache is that of a *Reverse Proxy*. See also *Cache Group*.

**(traffic ops) extension** Using *extensions*, Traffic Ops be extended to use proprietary checks or monitoring sources. See *Traffic Ops Extension*.

**forward proxy** A proxy that works that acts like it is the client to the origin. See *Forward Proxy*.

**geo localization or geo routing** Localizing clients to the nearest caches using a geo database like the one from Maxmind.

**health protocol** The protocol to monitor the health of all the caches. See *Health Protocol*.

**localization** Finding location on the network, or on planet earth. See *Localization*.

**mid (tier or cache)** The tier above the edge tier. The mid tier does not directly serves the end-user and is used as an additional layer between the edge and the origin. In a Traffic Control CDN the basic function of the mid cache is that of a *Forward Proxy*. See also *Cache Group*.

**origin** The source of content for the CDN. Usually a redundant HTTP/1.1 webserver.

**parent (cache or cachegroup)** The (group of) cache(s) in the higher tier. See *Cache Group*.

**profile** A group of settings (parameters) that will be applied to a server. See *Profile*.

**reverse proxy** A proxy that acts like it is the origin to the client. See *Reverse Proxy*.

## Symbols

(traffic ops) extension, [509](#)  
302 content routing, [509](#)

## A

astats (*stats\_over\_http*), [509](#)

## B

Bulk Upload Server, [39](#)

## C

cache, [509](#)  
Cache Control Header, [8](#)  
Cache Updates, [56](#), [80](#)  
cachegroup, [509](#)  
CDN, [3](#)  
Change Log, [37](#), [64](#)  
consistent hashing, [509](#)  
Content Delivery Network, [3](#)  
Content Routing, [15](#)  
content routing, [509](#)  
coverage zone map, [509](#)

## D

delivery service, [509](#)  
Delivery Service regexp, [51](#), [77](#)  
Delivery Service Type, [43](#), [69](#)

## E

edge (*tier or cache*), [509](#)  
Edge Health, [37](#)

## F

Forward Proxy, [6](#)  
forward proxy, [509](#)

## G

Generate ISO, [54](#), [78](#)  
geo localization or geo routing, [509](#)

Global Profile, [31](#)

## H

Header Rewrite, [44](#), [70](#)  
HEADER\_REGEXP, [51](#), [77](#)  
Health, [37](#)  
health protocol, [509](#)  
Health Tab, [35](#)  
HOST\_REGEXP, [51](#), [77](#)  
HTTP, [3](#)  
HTTP 304, [8](#)  
http/1.1, [3](#)

## I

Invalidate Content, [57](#), [82](#)  
ISO, [54](#), [78](#)

## L

localization, [509](#)  
Log File Analysis, [3](#)

## M

mid (*tier or cache*), [510](#)  
Monitor, [62](#), [64](#)

## O

origin, [510](#)

## P

parent (*cache or cachegroup*), [510](#)  
PATH\_REGEXP, [51](#), [77](#)  
profile, [510](#)  
Purge, [57](#), [82](#)

## Q

Queue Updates, [56](#), [80](#)

## R

Regex Remap Expression, [50](#), [76](#)

Revalidation, [8](#)  
Reverse Proxy, [4](#)  
reverse proxy, [510](#)

## S

Server Assignments, [51](#), [77](#)  
Signed URLs, [44](#), [71](#)  
Snapshot CRConfig, [56](#), [81](#)  
Static DNS Entries, [51](#), [77](#)

## T

Token Based Authentication, [44](#), [71](#)  
Traffic Monitor - Overview, [16](#)  
Traffic Ops - Default Profiles, [25](#)  
Traffic Ops - Installing, [21](#)  
Traffic Ops - Migrating from Traffic  
Ops 1.x to Traffic Ops 2.x, [26](#)  
Traffic Portal - Overview, [13](#)  
Traffic Router - Overview, [14](#)  
Transparent Proxy, [8](#)