Content Delivery User Guide

Table of Contents

Welcome to Edgio Content Delivery

Edgio *Content Delivery* is a Content Delivery Network (CDN) that lets you deliver your digital content to users all over the world. Using Edgio's globally distributed network of thousands of servers - interconnected with the world's leading ISPs - ensures that your content is distributed to your users securely, as fast as possible, and with high availability. You do not need to scale your own infrastructure to deliver your content to your audience, Limelight *Content Delivery* does it for you.

Edgio is not just one of the world's largest networks. It is the best choice for your business: an intelligent network that lets you offload all the complexity of delivering great digital experiences and still stay in control of your content. Even when the public internet is congested, your content gets delivered through our private CDN and into ISPs with over 80 terabits per second of egress capacity.

Content Delivery is designed to accommodate a broad range of use cases and still deliver the same performance and consistency. Scaling content caching to support global or regional software releases or updates, rich media and media library delivery, video streaming for on-demand libraries and live events and supporting online gaming distribution, *Content Delivery* offers the best solution across a wide range of protocols for origin offload at scale, content caching and distribution, and high speed delivery to end users.

Edgio has a rich history of delivering large scale events online including Olympic sports, World Cups, and U.S. presidential elections, as well as ensuring that global software releases and updates are delivered on time and that new video and gaming titles are delivered seamlessly to eager end users all rushing to get them just as they are released.

Our CDN and *Content Delivery* service runs on one of the world's largest private CDNs, by a group of world-class engineers, ensuring that your content gets delivered flawlessly from anywhere, to everywhere.

With *Content Delivery*, you can:

- Deliver content on the world's fastest network
- Dramatically reduce latency and processing overhead
- Easily customize configurations through the online Control Portal and APIs
- Set logical business rules to control where, when, and how your content is delivered
- Make key business decisions based on analytics, revealing object level and geo-specific data
- Protect your content using advanced authentication, verification, SSL delivery, and advanced access controls
- Deliver to multiple devices with today's most popular protocols, including HTTP progressive download, Chunked Streaming, DASH, HLS, HDS, and MSS.
- Integrate with Origin Storage for optimized cloud origin

Getting Started with *Content Delivery*

Working with Edgio

Your primary contact at Limelight will be your Account Manager, who will work closely with you from initial Limelight Account setup to “going live” with your service(s).

Initially, your Account Manager and Solutions Engineer will help you identify and confirm all of the information needed to complete your order. Once your order has been placed, your Account Manager will track the progress of the order and request any additional information needed to ensure a successful implementation.

When your *Content Delivery* service is ready (“provisioned”), you will receive a Welcome Letter from our support team with important information about your Limelight Account. For more information, please see Understanding Your Welcome Letter.

Once your service is configured and available to be used to deliver your content, you can contact our Support team 24 x 7 for technical assistance with your *Content Delivery* service. If you have questions about which *Content Delivery* features you purchased, or about additional services, your Account Manager will be happy to assist.

Finally, you can log in to Edgio Control at any time to submit and track support tickets, access documentation, and view detailed reports on the traffic handled by your *Content Delivery* service.

To set up your Limelight Account, please be prepared to provide your Account Manager the following information:

- **Contacts**
  - An administrative contact for your team
  - A technical contact for your team
- **Content Overview**
  - An overview of the types of content you deliver and the way your audience consumes it
  - The method you use to publish your content today and the frequency with which you update your content
  - Any content security or access restrictions
  - Any requirements for integration with your company’s content-management system
- **Content Statistics**
  - The types of files (file extensions) in your content
  - The average size of your individual content files
  - The approximate disk space required to store all of your content, and the projected annual growth rate
  - The approximate number of files in your library and percentage of those files your audience accesses regularly
- **Traffic Statistics**
  - The monthly traffic volume you expect *Content Delivery* to handle during the first six months and projected annual growth rate
  - The percentage of your traffic that comes from major geographic regions (North America, Europe, Asia, etc.), plus any international markets that require particular attention
- **Configuration**

- **Origin Hostname** - the URL where *Content Delivery* will request content to fill the cache
- **Published Hostname** - the URL prefix you want your audience to see in published links to your content (optional). You can provide this in the form of a DNS CNAME if you want to control the prefix completely or a suggested Limelight Account name (which, if accepted, will be prepended to the default *Content Delivery* form: account_name.vo.llnw.net).

Choosing an Origin

Edgio Origin Storage

Edgio Origin Storageis a purpose-built digital content origin - a secure origin distributed within the Edgio CDN.

Edgio Origin Storage lowers latency (response time) by storing your assets at the CDN edge of Edgio's massive global network. And because Edgio Origin Storage is fully integrated with *Content Delivery*, your content is retrieved in record-setting time from inside the Edgio network on every request - even while *Content Delivery* is retrieving and caching new or updated content.

With *Origin Storage 's* unique policy-based replication, you can ensure your content is replicated to the geographic locations that provide optimal performance. Policy-based replication increases content availability - you achieve global scale and elasticity while improving responsiveness to both planned and unexpected events.

It is easy to select *Origin Storage* as your *Content Delivery* origin - a single configuration setting is all that is required.

To order *Origin Storage* and for answers to additional questions about the service, please contact your Account Manager.

Other Hosting Options

If you are happy with your current origin, you can continue using it without interruption. *Content Delivery* will simply fill its cache from your origin as requests are received.

However, there are a few important requirements your origin must meet:

- Your origin server(s) must be able to respond to HTTP `GET` requests
- Your server(s) must be protected with industry-standard redundancy measures (such as a fault tolerance configuration and some type of load balancing)
- Your server(s) must be able to sustain the load generated when *Content Delivery* fills its cache. If a server fails to respond to a request for content that is either not yet in cache or is uncacheable (such as dynamic content), then content cannot be delivered for that request.

For more information on these requirements, please contact your Account Manager.

To order the service, please contact your Account Manager, who is prepared to help with all pre-sales questions, quotes, and ordering issues.

Preparing Your Content

General Guidelines

Uploading Your Content (Edgio Origin Storage Only)

If you are using Origin Storage as your origin, you need to upload your content to Origin Storage and set up your replication and location policies.

For detailed information on doing so, please see the links to the *Origin Storage Quick Start Guide*, *User Guide* and *Best Practices Guide* in the *Support > Documentation* page in Edgio Control.

Following Content Best Practices

Naming Files

Edgio recommends you incorporate a version number or date in your filenames. For example, the first version of your logo might be titled `logo.gif`. An updated version of the file might then be labeled `logo2.gif`.

Content Delivery will then cache updated objects separately from the objects they replace, which can make it easier for you to understand the data in reports.

Organizing Content

Organizing content into logical groupings and structures helps with configuration management and reporting across all your content. It may be that you have different requirements for the caching and delivery of images and software download or video files, for example, and need to use longer TTLs for the images than you need to for other assets. Organizing your content into different folder structures and URL paths (e.g., content.mycompany.com/images/ and content.mycompany.com/assets/) enables this to be done simply using the configuration options and reports available to you in the Control .

Controlling Freshness Checks

You can use HTTP response headers to control when Content Delivery performs freshness checks on your content.

The two most important response headers for cache control are:

- `Cache-Control: max-age` - specifies a maximum age. Intended for use by the requesting client (browser). If the origin server is running Apache servers, `Cache-Control: max-age` also sets the `Expires` header
- `Cache-Control: s-maxage` - specifically defines the TTL of a file in cache. Intended for use by the CDN.

The Content Delivery services react differently depending on which of these headers is present in an origin response:

- If the response includes the `Cache-Control: s-maxage` header, the `s-maxage` value sets the TTL of the associated object. The `Cache-Control: max-age` header, if any, is ignored.
- If the response includes the `Cache-Control: max-age` header but not `Cache-Control: s-maxage`, the `maxage` value sets the TTL of the associated object
- If `Cache-Control` isn't present, but the `Expires` header is, the `Expires` header value sets the TTL of the associated object
- Finally, if the response contains neither header, TTL is calculated based on the age of the content on the origin. For more information on TTL calculation, please see Time To Live (TTL).

For more information on using headers for cache control and other purposes, please see Header Manipulation.

Migrating from Other CDNs

If you are migrating your content from another CDN, you may want to develop a written migration plan that takes into account any CDN customizations or advanced features you depend on. Your Account Manager and Solutions Engineer can provide general migration advice and guidelines, and if desired, Edgio Advanced Services can help you develop the migration plan and manage the migration itself.

For more information on Edgio Advanced Services, please contact your Account Manager.

Managing Non-Edgio Origins

For the *Content Delivery* service to cache your content, your origin must provide the following two HTTP response headers with every object:

- `Last-Modified` - the date and time the object was last modified on the origin server (typically the same as the file timestamp).
- `Content-Length` - the size of the object in bytes. If the `Content-Length` header is missing, *Content Delivery* will not cache the file.

Also, it is important that you verify the accuracy of file timestamps on your origin server. Correct timestamps ensure *Content Delivery* performs freshness checks at the correct times.

Furthermore, if you are using a server load balancer, it is critical that you verify that file timestamps match each other across all servers. Timestamp inconsistency between servers may cause improper caching behavior.

Protecting Your Content

Content Delivery provides multiple ways to protect your content, including:

- URL protection
- Content and URL authentication
- “Deep link” prevention
- Geographic access restriction
- SSL delivery

For more details, please see Content Protection.

Understanding Content Delivery

Key Concepts

CDN Operation

When using the *Content Delivery* service, incoming content requests are handled directly by the *Content Delivery* service running at the CDN edge of the Edgio CDN, rather than your origin, using a cache hierarchy customized to your content type and the location of your content origin.

Fresh content is retrieved from the origin only as needed, and is quickly cached as close as possible to your audience. The scale needed to service your audience is provided by the CDN edge, delivering content quickly over the last mile of the Internet, removing the issue of scaling your infrastructure to accommodate for peak traffic, enhancing the user experience while retaining your workflow logic and reducing the overall overhead placed on your digital content infrastructure.

Content Cacheability

Content Delivery determines object cacheability (whether an object should be cached) by examining the response headers and status codes received when the object is requested from your origin.

Cache Control Headers

Content Delivery normally requires at least one of the following "cache control" response headers before caching an object:

- `Expires`
- `Last-Modified`
- `Cache-Control`

Objects are generally not cacheable when an origin provides any of the following headers (see `ignore_nocache` below):

- `Cache-Control: private`
- `Cache-Control: no-cache`
  Note: When determining cacheability, anything including and after `=` is stripped. For example. `Cache-Control: no-cache="set-cookie"` becomes `Cache-Control: no-cache`
- `Cache-Control: no-store`
- `Pragma: no-cache`

See also: [Caching Dynamic Content](#).

Response Codes

For the following response codes, responses will generally be cacheable if they contain Date or Last-Modified headers, or expiry information (Cache-Control or Expires headers).

- `200 OK`
- `203 Non-Authoritative Information`
- `300 Multiple Choices`
- `301 Moved Permanently`
- `410 Gone`

The following are cacheable under certain circumstances:

- `206 Partial Content`
  Note: Cacheable only if Partial Caching is enabled
- `302 Moved Temporarily`
  Note: Cacheable only if the origin sends `Cache-Control` headers defining expiry

Normally, *Content Delivery* does not cache responses with the following HTTP status codes:

- `204 No Content`
- `305 Use Proxy`
- `400 Bad Request`
- `403 Forbidden`
- `404 Not found`
- `405 Method Not Allowed`
- `414 Request-URI Too Large`
- `500 Internal Server Error`
- `501 Not Implemented`
- `502 Bad Gateway`
- `503 Service Unavailable`
- `504 Gateway Timeout`

In some cases, this behavior can be modified. For more information, see Content Acquisition, Origin Support and Origin Error Handling.

Authentication, Vary Headers & Cookies

Responses to content requests that require authentication (as defined by the presence of an `Authorization` header) are not cacheable by default; however, options are available to override this and provide cacheability for authenticated responses. If you need this feature, please contact Limelight Customer Service.

Responses with a Vary header are generally not cacheable unless the Accept-Encoding header is present and compression is applied by the configuration. Content Delivery can also be configured to ignore Vary headers. For more information, see Vary Header Optimization.

Even though Content Delivery caches cookie headers, they are stripped when serving objects from cache. Therefore, clients will only see cookie headers when an object they request results in a cache miss. This applies to both `Set-Cookie` and `Set-Cookie2` headers.

Vary Header Optimization

The HTTP `Vary` response header can be used to specify which HTTP request headers are used to determine the uniqueness of a cached object. For each variation in the values of the specified requests headers, a new version of the object must be requested from origin and stored in cache.

For example, `Vary: User-Agent` specifies that the value of the `User-Agent` request header should be used to determine uniqueness. This means that for each variation of the value of `User-Agent`, a different copy of the object will be requested from origin and stored in cache.

Obviously, the use of the `Vary` response header with request headers that have unrestricted values (such as `Cookie`, `Referer` and `User-Agent`) can significantly reduce CDN performance.

Because of this, *Content Delivery* ignores the `Vary` response header by default.

However, if you want to cache unique versions of content based on the `Vary: Accept Encoding` response header, you can configure *Content Delivery* to do so.

Specifically:

- If configured to ignore all Vary headers, the contents of the `Vary` header will not determine cacheability
- If configured to ignore a list of `Vary` headers, those headers will be filtered out of the `Vary` header sent from the origin. With the exception of `Accept-Encoding`, if any headers remain, the content is marked as uncacheable
- If only `Vary: Accept-Encoding` remains, each different encoding of an object is cached

Regardless of this setting, all of the `Vary` headers associated with the object are cached and passed on to the client in the response.

Freshness Checks

Content Delivery automatically performs "freshness checks" on your cached content when stale content is requested and retrieves updates from your origin whenever content becomes "stale". You have complete control over how often *Content Delivery* refreshes cached objects and under what conditions.

Each time it receives a request for an object, *Content Delivery* determines if the object is cacheable, and if so, whether it is currently in cache:

- If the object is not cacheable, *Content Delivery* requests it from your origin and delivers it to the requesting client.
- If the object is cacheable but not in cache, *Content Delivery* requests it from your origin, and then simultaneously caches and delivers it to the requesting client.
- If the object is in cache, *Content Delivery* performs a freshness check based on the cacheability of the object and the details of how it was cached. If the object is fresh, it is delivered directly from cache. Otherwise, it is requested from origin, and then simultaneously cached and delivered.

Time To Live (TTL)

Object freshness is managed using the concept of "Time To Live" (TTL) - how long an object may remain in cache before a freshness check. *Content Delivery* assigns a TTL to every object it caches, and then evaluates the TTL during each freshness check.

If an object's TTL is found to be expired during a freshness check, *Content Delivery* asks your origin if there is a newer version of the object. Specifically, *Content Delivery* makes an HTTP request for the object and includes an `If-Modified-Since` (IMS) header with the date and time the object was last modified.

If your origin determines that the object has not been modified since the last freshness check, it responds with an HTTP `304 Not Modified` status, and *Content Delivery* immediately delivers the cached object.

However, if the object has been modified, your origin responds with an HTTP `200 OK` response and the updated object. *Content Delivery* then simultaneously caches the object, updates its TTL, and delivers it to the requesting client.

If your origin did not provide a TTL-related response header when *Content Delivery* originally cached the object, and no special TTL-related options are configured in *Content Delivery*, TTL is re-calculated during each freshness check based on the HTTP `Last-Modified` header provided by your origin.

Generally speaking, the longer it has been since an object was modified on your origin, the less frequently it will be checked for freshness.

Specifically, *Content Delivery* calculates the TTL by determining the elapsed time since the object was modified and multiplying by a conversion ratio. The default conversion ratio is 20%, and there is also a default maximum freshness check interval of 72 hours.

Examples:

- During a freshness check, object "A" is determined to have been modified 5 hours ago. Multiplying 5 hours by 20% yields a TTL of 1 hour. This means that *Content Delivery* will not check the object for freshness during the next hour.
- During a freshness check, object "B" is determined to have been modified 500 hours ago. Multiplying 500 hours by 20% yields a TTL of 100 hours. However, this exceeds the maximum of 72 hours. This means that *Content Delivery* will not check the object for freshness during the next 72 hours.

> **Note:** By using certain *Content Delivery* features, you can adjust the conversion ratio and also specify the minimum and maximum freshness check intervals. For more information, see TTL Management.

Features

The *Content Delivery* service provides a rich set of features (configuration options) that give you complete control over how your content is requested, cached and delivered.

The most popular configuration options are categorized and described in detail in this section, with references to additional technical documentation when available. If you're unsure whether you need certain options, or which settings to select, please contact your Account Manager or Solutions Engineer.

> **Note:** Some features must be purchased separately or specifically enabled by Edgio before they become available. Please contact your Account Manager or Limelight Customer Service if you have questions about the availability of specific features.

You can change many configuration options yourself, using the Configuration Self Service feature in Control. To access Configuration Self Service, choose the *Configure* tab in *Control* and select the service you want to configure.

> **Note:** Some advanced options can only be configured by Edgio. If such options are configured, they will become visible in "read only" mode in Configuration Self Service.

For more information on individual options and settings, please see the *Configure* section of the *Control User Guide*.

Key Features

Content Acquisition

Customization

Content Protection

Defensive Protection

Acceleration

Delivery Optimization

Content Acquisition

Origin Support and Error Handling

The *Content Delivery* service is designed to provide a caching service for digital files and content, delivering that content to end users requesting it from the Edgio CDN edge, not from the origin server where the content is stored. Distributing the delivery of the content away from the content origin provides the scalability and availability needed to deliver your content at scale, globally.

A content origin can be 1) a Edgio Origin Storage account, 2) a storage service hosted outside the Edgio infrastructure using a third-party storage service, or 3) a dedicated storage infrastructure hosted by you, within your data center facility.

Content Delivery includes a number of configuration options that specify how to acquire content from the content origin and what to do if the origin returns an error (or fails to respond at all).

Published and Origin Hostname URL pairs

Every *Published URL* (a URL you publish to your audience) is mapped to an *Origin URL* (used by *Content Delivery* to request the content from your origin).

For example, suppose that your *Published Hostname* is `www.mydomain.com`, and your *Origin Hostname* is `origin.mydomain.com`. If you publish all the images in the directory `/images`, the URL pair would be

- *Published URL*: `http://www.mydomain.com/images/`
- *Origin URL*: `http://origin.mydomain.com/images/`

When Content Delivery receives a request for a *Published URL*, it "rewrites" the *Published URL* to form the *Origin URL*, and then uses the *Origin URL* to retrieve the associated content from cache, or from your origin if necessary.

Host Header Value (Host Header Override)

Specifies the hostname that will be provided in the HTTP Host Header when *Content Delivery* makes requests to your origin.

- This option is useful where the *Origin Hostname* is not a CNAME target for the *Published Hostname*, but the origin still expects to find the *Published Hostname* in the Host Header.
- Using the Host Header Override option changes only the HTTP Host Header value. *Content Delivery* will continue to use the Origin Hostname when making requests (DNS Lookups and TCP connections) to your origin, and the URL path of the content requested by the client will be unchanged.

Note: Cache Keys are based on the *Origin Host* (not the *Published Host*) of cached objects and contain the following:

- protocol scheme (HTTP or HTTPS)
- domain (*Origin Host*)
- path
- HTTP method that caused the object to be cached
- optional custom values appended to the end of the key

All or part of an URL's query terms can optionally be excluded from the key (see Flexible Cache Key Manipulation).

Backup Origin Support

Specifies a backup origin to connect to, in the event that a request to the content origin specified by the **Origin Hostname** fails, with an HTTP 503 or 504 response code.

301 & 302 Status Code Handling (Edge Redirection)

Content Delivery can redirect requests at the CDN edge, reducing response time by eliminating the need for origin-based redirects. Edge redirects can be triggered by requests for specific URLs and URL patterns and also by requests that contain specific headers.

- **URL-Based Redirection**: If a requested URL matches a specified URL or URL pattern, *Content Delivery* can return a `301 Moved Permanently` or `302 Moved Temporarily` HTTP Status Code in the response, with a `Location` header that contains a URL of your choosing. This URL may redirect the client to an object in cache, another Published URL, or the content origin.
- **Request Header-Based Redirection**: If a request contains a specified header, *Content Delivery* can return a `301 Moved Permanently` or `302 Moved Temporarily` HTTP Status Code in the response, with a `Location` header that contains a rewritten Published URL. When rewriting the *Published URL*, *Content Delivery* replaces the original hostname with a specified hostname.

Enhanced 302 Status Code Handling (Chase Redirect)

Content Delivery has the capability to "chase" (i.e., follow) the Location header returned with a `302 Moved Temporarily` HTTP status code from the origin, instead of returning it to the client. In this case, the status code returned to the client will be whatever is returned by the chase attempt.

400, 403, & 404 Status Code Handling

When Content Delivery has an expired object in cache and receives a `400 Bad Request`, `403 Forbidden`, or `404 Not Found` HTTP status code on an ensuing refresh check, its default behavior is to remove the object from cache. Content Delivery can be configured to leave the object in cache instead. Stale content may be served as a result. This also removes negative caching, which may result in a performance hit.

404 Status Code Handling

- **Request content from an alternate location**: If a response with a `404 Not Found` HTTP status code is received from the content origin, Content Delivery can request the same content from the "backup" host specified in an alternate hostname or from the "backup" origin specified by an alternate origin URL (instead of returning the error code to the client).

- **Object not available URL**: If a response with a `404` HTTP status code is received from the content origin, Content Delivery can return content from a secondary, or “fallback”, URL (instead of returning the error code). This makes it easy to specify a single “content not found” landing page.
- **Serve stale content**: If Content Delivery receives a response with a 404 HTTP status code from the content origin but there is cached content for the requested URL (that is, a refresh check is needed), Content Delivery can return the stale content instead of a response with the `404` status code.
- **Treat responses with 200 HTTP status code without a body as a response with 404 HTTP status code**: If a response with a `200 OK` HTTP status code is received from the content origin, but the response body is “empty” (contains no content), Content Delivery can be configured to return the `404 Not Found` HTTP status code instead of serving the object.

503 & 504 Status Code Handling

- **Request content from an alternate location**: If a response with a `503 Service Unavailable` or `504 Gateway Timeout` HTTP status code is received from the content origin, Content Delivery can request the same content from the “backup” host specified in an alternate hostname or from the "backup" origin specified by an alternate origin URL (instead of returning the error code to the client).
- **Service not available URL**: If a response with a `503 Service Unavailable` or `504 Gateway Timeout` HTTP status code is received from the content origin, Content Delivery can redirect the client to a specific URL (instead of returning the error code to the client). This makes it easy to specify a single “system error” landing page.
- **Don't serve stale content**: If Content Delivery receives a response with a `503 Service Unavailable` or `504 Gateway Timeout` HTTP status code from the content origin when doing a refresh check, and there is cached content for the requested URL, Content Delivery can return the `503` or `504` HTTP status code instead of returning the stale content.

HTTP Requests to Origin

Requests from Edgio CDN to a customer's origin always contain the `Via` and `X-Forwarded-For` headers.

Via Header

The `Via` header provides your origin with the gateways and proxies through which a request has passed. For example:

```
Via: 1.1 cds539.lax.llnw.net:80(EdgePrism/4.0.0.3)
```

X-Forwarded-For

The X-Forwarded-For header provides your origin with the IP address of each node through which a request passes on its way to the origin. For example:

```
X-Forwarded-For: 10.21.18.15
```

The `X-Forwarded-For` header may contain multiple IPs, making it difficult to determine the originating ("client") IP address. Content Delivery provides a separate header that contains the true IP address of the originator. For more information, see Sending True-Client IP to Origin.

Custom Error Landing Pages

If your origin returns one of the following HTTP Status Codes:

- `301 Object Moved`
- `302 Object Moved Permanently`
- `404 Not Found`
- `503 Service Unavailable`
- `504 Gateway Timeout`

During a freshness check for a requested object, *Content Delivery* can respond with a “custom error page”.

Sending Geo IP Info to Origin

When *Content Delivery* makes requests to your origin, it can also include geographic information about your users in special HTTP request headers. This means you can give your origin applications access to detailed user information by making simple configuration changes in *Content Delivery* .

Examples:

- Add a request header, such as `X-IP-Geo-Country,` containing the just the Country name
- Add a request header, such as `X-IP-Geo-All,` containing all geographic information available about the end user

> **Note:** The geographic information *Content Delivery* provides is based on the requesting IP address.

Sending True-Client IP to Origin

If you want to identify the IP addresses of end users during origin requests from *Content Delivery*, you can use the information in the `X-Forwarded-For` HTTP request header, which *Content Delivery* always provides with origin requests.

However, when a content request follows a complex internet path (such as when traversing multiple proxies and load balancers), the IP address of each traversed node is added to the `X-Forwarded-For` header. This can make it difficult to pick out the requesting IP address.

To avoid this problem, you can configure *Content Delivery* to provide just the requesting IP address in a separate HTTP request header, either the default `True-Client-IP` header or in another header name you specify.

Secure Cache Diagnostics (Debug Headers)

By making an HTTP content request with special "Custom Debug Headers," including a shared secret specific to your service, you can retrieve cache-related information about individual content objects and prevent others from accessing the information.

Diagnostic response headers can include the following information:

- Whether or not a response is cacheable
- How the cache responded to a request (hit, miss, etc.)
- The number of seconds before the cached response will be considered stale (TTL)
- The total number of seconds representing the freshness lifetime of the response (age + TTL) and how the value was determined (headers, overrides, adaptive TTL, etc .)

To enable this feature, please contact your Account Manager, Solutions Engineer, or Edgio Support, and specify which of the Header Keys you want to use. Each Header Key must be individually enabled by Edgio before it can be used.

When the feature is activated, you will be provided with a unique shared secret.

Using Secure Cache Diagnostics

To use this feature, include the following custom headers in an HTTP GET request for an object:

- `X-LLNW-Dbg-Secret <sharedsecret>`
- `X-LLNW-Dbg-Hdrs <header_key1>,<header_key2>,...`
- `X-LLNW-Dbg-Cache-Key-Hash`

If the `X-LLNW-Dbg-Secret` header or its value is missing, or if the secret is invalid, the `X-LLNW-Dbg-Hdrs` header will be ignored, and the request will be processed without it.

One or more header keys can be included in the `X-LLNW-Dbg-Hdrs` header. The *Content Delivery* service will reply with a separate response header and value for each.

The table below summarizes the request and response header values for each type of request.

Header Key	Response Header	Possible Values

`is-cacheable`	`X-LLNW-Dbg-Is-Cacheable`	`Yes`
		`No`
		`Negative`

`cache-hit-type`	`X-LLNW-Dbg-Cache-Hit-Type`	`HIT`
		`MISS`
		`REFRESH_HIT`
		`REF_FAIL_HIT`
		`REFRESH_MISS`
		`CLIENT_REFRESH_MISS`
		`IMS_HIT`
		`NEGATIVE_HIT`
		`DENIED`
		`OFFLINE_HIT`
		`REDIRECT`

`ttl`	`X-LLNW-Dbg-TTL`	`n{...} seconds` an integer followed by a space and the string "seconds"
`fresh-life-total`	`X-LLNW-Dbg-Fresh-Life-Total`	`n{...} seconds` an integer followed by a space and the string "seconds"
`cache-key-hash`	`X-LLNW-Dbg-Cache-Key-Hash`	Unique key for a cached object represented as a string of hexadecimal digits

Request & Response Example

Request	Response
`GET http://www.customer.com/object.txt HTTP/1.1...X-LLNW-Dbg-Hdrs: is-cacheable,cache-hit-typeX-LLNW-Dbg-Secret: sharedsecret`	`HTTP/1.1 200 OK...X-LLNW-Dbg-Is-Cacheable: Yes...X-LLNW-Dbg-Cache-Hit-Type: HIT`

Customization

HTTP Methods

HTTP Range Requests

TTL Management

HTTP Methods

The *Content Delivery* service accepts the following types of HTTP methods:

- `GET`
- `HEAD`
- `POST`

If an HTTP request is received for a method that is not configured, *Content Delivery* responds with an HTTP `400 Bad Request` status code.

For HTTP `HEAD` requests, *Content Delivery* sends the `HEAD` request to the origin, caches the response if it is cacheable, and delivers the response to the requestor.

For HTTP `POST` requests, *Content Delivery* provides additional control: you can accept all `POST` requests and their request bodies, accept all `POST` requests but discard the request bodies and process as `GET` requests, or deny `POST` all requests. Accepted `POST` request bodies are passed on to your origin.

HTTP Range Requests

Content Delivery accepts HTTP `GET` requests containing `Range` headers ("range requests"), which can be used to access specific byte ranges (rather than the entire file). Range requests are often used to improve the performance of client-based download and media access operations.

Bytes are numbered from zero, and more than one byte range can be specified in each request.

If a range request is valid, *Content Delivery* responds with a `206 Partial Content` status code, and a `Content-Range` header specifying the range returned.

Content Delivery can be configured to respond with a `416 Requested Range Not Satisfiable` status code if a requested range is invalid.

For more information on range requests, please see RFC 7233.

TTL Management

A cached object's time to live (TTL) determines if a freshness check should be performed on the object, to determine if an updated object should be obtained from the origin server, then cached. Edgio examines headers and performs freshness checks as described in Freshness Checks. If needed, you can override the origin's Cache-Control header and TTL values as described in Cacheability (TTL) Overrides.

Cacheability (TTL) Overrides

Edgio examines headers and performs freshness checks as described in Time to Live (TTL). If desired, you can override the `Cache-Control` header and TTL values using Control.

The override feature is helpful if you cannot easily modify TTLs on your origin, or if you simply prefer to modify them using *Content Delivery* configurations.

You can configure *Content Delivery* to use a fixed TTL value you select , or to use minimum and maximum TTL values you specify (and calculate the most appropriate TTL for each request).

Header Manipulation

The SmartPurge Rest API Guide appendix includes the following:

- Request Endpoint Schema
- Error Response Schema
- Error Response Descriptions
- HTTP Status Codes
- API Client Sample Code

Content Delivery lets you manipulate both HTTP request and response headers. For example, *Content Delivery* can pre-process HTTP request headers before they are passed to your origin. You can configure *Content Delivery* to analyze and manipulate existing headers, to add new "custom" headers, and to remove named headers. In addition, *Content Delivery* can process the response headers received from your origin to determine how to respond to the original request. *Content Delivery* can strip specified response headers, pass specified headers on to the client either "as is" or with modifications, and even add new headers.

Request Header Manipulation

HTTP request headers are used to convey specific information about the requesting user, the content requested and the capabilities of the browser or software making the request. Certain request headers are commonly used by browsers when requesting content, and request headers may also be added by custom clients making content requests.

Content Delivery passes on these request headers when requesting content from your origin (during freshness checks and when requesting non-cacheable content).

When your origin receives HTTP requests, your web applications can analyze the content of specific request headers and apply business logic to determine how to respond. For example, you may want to examine the `Host` header to determine if the request is being made by Edgio, or analyze the requesting IP address to filter out spam.

To make origin-based analysis easier, and also to support more complex business logic, *Content Delivery* can pre-process HTTP request headers before they are passed to your origin. You can configure *Content Delivery* to analyze and manipulate existing headers, to add new "custom" headers, and to remove named headers.

> **Note:** Manipulated request and response headers are not passed to any "fallback" URLs you may have specified. If you need to pass and react to header information after an HTTP error, Limelight recommends using the "backup" URL options instead. For more information, please see the 404 Status Code Handling and 50x Status Code Handling sections.

Response Header Manipulation

HTTP response headers are used by your origin to communicate specific information about the content being requested. Specific response headers may be intended for use by the requesting client, by the CDN, or both.

Content Delivery can process the response headers received from your origin to determine how to respond to the original request. *Content Delivery* can strip specified response headers, pass specified headers on to the client either "as is" or with modifications, and even add new headers.

For example, you can configure *Content Delivery* to add custom response headers to control the behavior of browsers and other clients. In *Control*, you can configure up to three custom response headers to set cookies, manipulate browser cache settings, and so on. You can also configure an unlimited number of additional response headers on request - if you need to do so, please contact Limelight Customer Service. Response headers can also be manipulated by other *Content Delivery* configuration options.

> **Note:** Manipulated request and response headers are not passed to any "fallback" URLs you may have specified. If you need to pass and react to header information after an HTTP error, Limelight recommends using the "backup" URL options instead. For more information, please see the 404 Status Code Handling and 50x Status Code Handling sections.

Vary Header Optimization

The HTTP `Vary` response header can be used to specify which HTTP request headers are used to determine the uniqueness of a cached object. For each variation in the values of the specified requests headers, a new version of the object must be requested from origin and stored in cache.

For example, `Vary: User-Agent` specifies that the value of the `User-Agent` request header should be used to determine uniqueness. This means that for each variation of the value of `User-Agent`, a different copy of the object will be requested from origin and stored in cache.

Obviously, the use of the `Vary` response header with request headers that have unrestricted values (such as `Cookie`, `Referer` and `User-Agent`) can significantly reduce CDN performance.

Because of this, *Content Delivery* ignores the `Vary` response header by default.

However, if you want to cache unique versions of content based on the `Vary: Accept Encoding` response header, you can configure *Content Delivery* to do so.

Specifically:

- If configured to ignore all Vary headers, the contents of the `Vary` header will not determine cacheability
- If configured to ignore a list of `Vary` headers, those headers will be filtered out of the `Vary` header sent from the origin. With the exception of `Accept-Encoding`, if any headers remain, the content is marked as uncacheable
- If only `Vary: Accept-Encoding` remains, each different encoding of an object is cached

Regardless of this setting, all of the `Vary` headers associated with the object are cached and passed on to the client in the response.

CORS Header Manipulation

Content Delivery can be configured to add, set and override CORS ("Cross Origin Resource Sharing") HTTP Headers on a per-request basis.

The following CORS headers are supported:

- Request
  - `Origin`
- Response
  - `Access-Control-Allow-Origin`
  - `Access-Control-Allow-Credentials`
  - `Access-Control-Expose-Headers`

This feature ensures that CORS Request and Response headers are configurable, and simplifies and standardizes the process of adding CORS support to *Content Delivery* configurations.

Using CORS can significantly improve cache efficiency when using a single Origin to syndicate and distribute content over multiple published domains. If you need to use this feature, please contact Limelight Customer Service . For more information on CORS, see the W3C Recommendation.

CORS Configuration Options

Header	Type	Options
`Origin`	Request	The value to send in the header, in the form of an HTTP protocol and domain (such as `http://www.example.com/`). **Note**: This header is added only for requests sent to Origin.
`Access-Control-Allow-Origin` (ACAO)	Response	One of the following: • **Pass Through** (default). Do not add an ACAO header, but if the Origin returns one, pass it through to the client. • **None**, Do not add an ACAO header. If an ACAO header is returned by the Origin, delete it. • **Wildcard**. Add an ACAO header with * (an asterisk) as the value. • **Origin**. Add an ACAO header with the same value as in the `Origin` header.
`Access-Control-Allow-Credentials` (ACAC)	Response	One of the following: • **Pass Through** (default). Do not add an ACAC header, but if the Origin returns one, pass it through to the client. • **None**, Do not add an ACAC header. If an ACAC header is returned by the Origin, delete it. • **True**. Add an ACAC header with `true` as the value. **Note:** If the ACAC header is set to `true` and the ACAO header is set to `*`, the ACAO header will be reset to the value in the `Origin` header. This complies with section 6.1 of the CORS specification, which states that if the ACAC header is `true`, the ACAO header may not be `*`.
`Access-Control-Expose-Headers` (ACEH)	Response	A list of comma-separated header names, without any spaces. Per section 6 of the CORS specification, the list should **not** contain any of the following headers: • `Cache-Control` • `Content-Language` • `Content-Type` • `Expires` • `Last-Modified` • `Pragma`

Note: ACAO, ACAC, and ACEH headers are added only for responses to clients (not to Origin). If the response comes from cache, the headers are added just before the response is sent.

Flexible Cache Key Manipulation

Content Delivery uses unique “Cache Keys” to locate and return cached objects. By default, each Cache Key is based on the full origin URL of the requested object, including any query terms, and the HTTP method that caused the object to be cached.

Query Term Customization

You can configure *Content Delivery* to extensively modify query terms before they are stored in the Cache Key. Modifications may include:

- Stripping (removing) all query terms.
- Excluding specific query terms
- Including only specific query terms

Removing query terms can eliminate duplicate caching of content (and increase CDN efficiency and “Cache Hit Ratios”). Duplicate caching can occur when the origin provides the same content regardless of the presence of one or more query terms. Unless those query terms are removed, a new copy of the content will be cached for each change in the values of the query terms. Removing query terms can also eliminate duplicate caching when multiple URLs map to the same cached content by using different query terms.

Adding Cache Key Suffixes

Cache Keys can also be configured to add custom values to the end of the key.

Adding Suffixes Using Request Headers

Content Delivery caches multiple responses identified by a single URL, based on different request header values

Custom Cache Key suffixes can be provided in an HTTP request header. When a specified header name is present in a request, Content Delivery appends the header value to the cache key of the response.

Contact your Account Manager or Solutions Engineer if you want customized query terms or custom values added to your Cache Keys.

Caching Dynamic Content

Content Delivery normally requires at least one of the following "cache control" response headers before caching an object:

- `Expires`
- `Last-Modified`
- `Cache-Control`

An object length must also be specified in the Content-Length header.

However, for HTTP responses that are generated dynamically (“generated responses”), origins often do not supply these cache control response headers, or when they are provided, they are configured to state that the content is not cacheable.

The default caching behavior of *Content Delivery* is to obey the instructions in the cache control response headers provided by the origin server, ensuring that content your origin specifies as “not cacheable” is not cached, and vice-versa.

However, if most of your dynamically generated content is valid for the same amount of time, and your origin doesn’t supply cache control headers, you can also configure *Content Delivery* to cache it anyway. In these cases, your dynamic content will then be cached regardless of the Expires header, the Last-Modified header, or the `s-maxage` and `max-age` fields of the `Cache-Control` header.

The Last-Modified header is required in addition to a Content-Length to cache content whether or not Cache-Control is present.

Wildcard & Regex Configurations

To provide maximum flexibility during configuration, many features allow the use of wildcard and regular expressions (Regex) to specify the URL pattern that must be matched before a feature is applied.

Cache Purging

Objects are normally updated in or removed from cache during “freshness checks” with your origin. For a given object, a freshness check is initiated when a request has been made for the object, and the object’s TTL (Time To Live) has expired.

In general, setting object TTL is the best and most efficient way to manage cached content. For example, a news site may need to provide rapid updates to a breaking video story. The video can be updated in cache as quickly as desired by assigning it a low TTL value using an HTTP response header. In most cases, there is no need to remove the video from cache directly.

However, there are special cases where content needs to updated on the next user request or even pro-actively removed from cache as soon as possible. This is known as "purging the cache" or just "purging". Examples of when purging might be necessary include:

- Text is misspelled in the caption of a newly-uploaded video, and you need to update the video in cache as quickly as possible.
- You discover that some of your cached content is infringing a copyright and need to delete the content from cache as soon as possible.
- You lose a contract with a content provider and are obligated to delete the provider’s content from your cache as soon as possible.
- During a full website update, when you need to quickly update many related website objects (images, text, video, etc.) at the same time.

Edgio's *SmartPurge* executes purge operations more quickly and reliably than older technologies. The advanced version of *SmartPurge*, *SmartPurge Plus*, provides additional features, including higher purge queue priority, and additional API features such as unlimited callbacks.

You can access *SmartPurge* through either the Control Portal or the SmartPurge REST API. For more on purging, please see the Edgio SmartPurge Data Sheet and Intelligent High-Speed Purging.

Content Delivery lets you purge (remove) cached files. You can do so via the *Control* or the Purge REST API.

Purge Via *Control*

The Purge user interface in the *Control* lets you submit requests to purge one or more cached files. You can also view the status of previously-submitted purge requests, and save frequent requests for reuse.

Purge Via API

The Purge REST API provides programmatic access to the same capabilities as the *Control* . You can use the Purge API to generate Purge requests and view status programmatically.

Content Protection

URL Protection

SSL Protection

Authentication

URL Protection

URL protection is a feature that protects content via allow listing and deny listing of domains and geographic regions. You can use URL protection to guard content from link thieves, sniffers, and problem geographies and IP addresses or ranges.

MediaVault URL Protection

MediaVault is a high-performance URL authentication service. *MediaVault*'s main purpose is to help you secure your content from unauthorized viewing.

Deep Link Prevention

Deep-linking is the practice whereby a user obtains access to a URL that points to some of your content and passes the URL to someone else not authorized to access the content (for example, someone who has not paid for your services). MediaVault 's built-in URL protection ensures deep-link denial. The protection comes from specific parameters in the URL, which are usually generated on the fly using server-side scripting when the URL is published by the customer's server. Parameters include valid start and end time (so the URL can expire), and validation against user IP address, referrer and page URL (where user is coming from).

So, you could create a short-lived URL that gives a user, say, three minutes to access a specific asset on your website, and limits access to requests from that user's originating IP address and referrer. That link would then be useless if published on any other website.

For large files that are accessed via HTTP range requests, MediaVault protection can be limited to range requests for a specified number of bytes at the beginning of a file (the "initial bytes").. Subsequent range requests are not protected. This is useful for preventing deep links to very large files while ensuring the fastest possible response. Users are unable to make use of the files without access to the initial bytes, and only the initial request requires an interaction with MediaVault .

For detailed information on how MediaVault works and how to use the various MediaVault options, see the *MediaVault User Guide* in the *Support > Documentation* page in the *Control*.

Content and URL Authentication

MediaVault maximizes authentication performance by using tokens to avoid three-way handshakes (common to other methods of authentication) that can lead to severe connection time latency.

Please note that MediaVault is *not* a replacement for DRM and should not be associated with user authentication.

MediaVault works like this:

- You enter a shared secret during the configuration process.
- You then generate a token (MD5 hash) for each published URL, based on the shared secret, and append it to the URL in a query term or provide it in a cookie. You can generate the token manually by navigating to the *Content > Secure > MediaVault* page in the *Control*, or in server-side code on your origin.
- When a request is received, MediaVault uses the same hash algorithm to create it's own token, which should be identical to the one you appended.
- If the tokens match, MediaVault then looks for additional MediaVault -specific query terms (such as end date/time and IP address/mask) to determine whether the request is valid. If the tokens don't match, the request is rejected.

For more information, see the latest MediaVault User Guide (navigate to *Support > Documentation* in the *Control*).

Cookie-Based Authentication

URL authentication information is normally passed to MediaVault via URL query terms, which are appended to each protected URL.

For HTTP requests, MediaVault can switch to using a cookie (instead of the URL terms) after the first request from a given user. The cookie is then read on each subsequent request, authentication is performed using the cookie data, and the cookie is updated with new expiration information as needed on each response. This is typically used during request and response conversation involving some form of manifest file and subsequent requests for files, such as media streaming and software downloads. It is most commonly used for Media *Content Delivery* scenarios, but can also be used with *Content Delivery* if needed.

For detailed information on how MediaVault works and how to use the various MediaVault options, see the *MediaVault User Guide* in the *Support > Documentation* page in the *Control* .

Geographic URL Protection

You can use the *Geo Compliance* feature to restrict access to your content by geographic area. Both IPv4 and IPv6 addresses are used when determining geographic location.

This feature is ideal for managing media licenses with geographic restrictions. It's also useful for sites where advertising is a primary driver, as the audience can be constrained to the target geographies specified by the advertisers.

Geographic Blocking

With *Geo Compliance*, you can either allow or deny (allow list or deny list) access for each geography you specify, and also control which parts of your content library are affected by each restriction. Geographies can be specified by country, region or US state.

> **Note:** The geographic origin of each request is determined based on the user's IP address.

Anonymizer Blocking

Geo Compliance also lets you control anonymizer access to your content. You can allow or deny anonymizer access, and specify the level of certainty to apply when identifying anonymizers.

SSL Protection

Securing Your Content with SSL

Edgio SSL Protection

Securing Your Content with SSL

Secure Sockets Layer (SSL) is a security protocol used by web browsers and web servers to help users protect their data during transfer. SSL Certificates are small data files that digitally bind a cryptographic key to an organization's details. In the case of a web browser, SSL activates https and allows secure connections from a Web server to the browser. Edgio supports SSL at both the origin server and the CDN edge.

Customers can use Edgio's certificates but more typically, they opt to use their own. Using your own certificate has advantages: you can use your own domain, and you have full control over certificate attributes. However, note that if you use your own certificates, a custom setup is required, which may delay your implementation.

Using Edgio certificates also has advantages: they are less expensive, you do not have to renew them, and they are faster to implement. However, you cannot use your own domain, you must use a Edgio-owned hostname, and you do not have control over certification attributes.

If you sign up for the Edgio SSL service, we recommend that you use our supplied certificate for SSL connections.

If you sign up for the Edgio SSL service, we recommend that you use our supplied certificate for SSL connections. You can use your own certificate, but that requires a custom setup and may delay the SSL implementation.

> **Note:** You cannot use CNAMEs when using SSL. A CNAME will cause a site name mismatch with the Edgio certificate for the Edgio host.

Contact your Account Manger if you want to include the SSL option with your Orchestrate Performance service.

How do SSL Certificates Work?

When a browser encounters a website with SSL:

1. A browser attempts to connect to a Website secured with SSL.
2. The browser requests that the web server identify itself.
3. The server sends the browser a copy of its SSL Certificate.
4. The browser checks whether it trusts the SSL Certificate. If so, it sends a message to the server.
5. The server sends back a digitally signed acknowledgment to start an SSL encrypted session.
6. Encrypted data is shared between the browser and the server.

HTTPS is HTTP delivered using the SSL protocol. Edgio is equipped to use HTTPS for secure communication between the origin server and Edge Servers. This section provides information about certificates and explains how to obtain and install certificates.

> **Note:** if you want to have your content delivered via HTTPS you must contact your Account Manager to have Edgio perform the necessary configurations. You can have content delivered via HTTP only, HTTPS only, or both HTTP and HTTPS.

Certificate Types

Due to the prevalence of counterfeit Websites on the Internet, one of the key purposes of an SSL Certificate is to help assure consumers that they are actually doing business with the Website they believe they are accessing. An SSL Certificate provided by a trusted third-party authenticates the identity of a Web site based on a validation process performed by the Certificate Authority (CA). However, there are several different levels of validation that back SSL Certificates depending on the certificate and the CA.

The level of identity authentication assured by a CA is a significant differentiator between SSL Certificates. The explosive growth of phishing and other fraudulent websites designed to steal information from consumers has put a spotlight on the authentication strength of various SSL certificates and the authentication processes employed by different CAs. There are three commonly recognized categories of SSL authentication:

- Domain Validation
- Organization Validation
- Extended Validation.

Domain Validation

Domain validation (DV) provides the lowest level of validation available from commercial certificate authorities. DV verifies that the requestor has some kind of control over the domain and triggers most browsers to put a padlock in the address bar.

Most CAs can issue DV certificates in less than a day.

Organization Validation

Organizational Validation (OV) stands as a more advanced and better SSL certificate because it has more validation requirements. OV authenticates domain ownership plus the organization's information included in the certificate (name, city, state, and country). OV triggers some browsers to color the address bar blue.

Most CAs can issue OV certificates in one to two days.

Extended Validation

Extended Validation (EV) represents the best SSL Certificate and is the recommended SSL Certificate type. As the highest level of authentication using validation criteria defined by the CA/browser forum and audited annually by KPMG, EV triggers some web browsers to turn green in the address bar, and displays the organization's name plus the name of the issuing CA. It also validates domain ownership and organizational information, along with the legal existence of the organization, and certifies its awareness and approval of the request. The result of opting for a higher value EV certificate is more security and more online trust, which leads to more transactions conducted online.

Websites that benefit the most from EV are:

- E-commerce Websites that collect credit card information.
- Websites that operate in a competitive environment where customer loyalty and brand protection is key.
- Websites collecting personal data.
- Websites with customer or employee login forms.
- Websites using third-party payment processing (for example PayPal).

Most CAs issue EV certificates in seven to ten days.

Obtaining a Certificate

Customers generally have two SSL certificates options. You can bring your own, or you can use the Edgio shared certificate.

We recommend that you use our supplied certificate for SSL connections. You can use your own certificate, but that requires a custom setup and may delay the SSL implementation.

If you use your own certificates, Edgio has a partner relationship with Symantec in which we can order the certificates on behalf of the customer. For each SSL instance, you must provide Edgio the following:

- X509 server certificate in PEM format (not PKCS7, PKCS10, or DER)
- RSA private key, which will be used to generate the server certificate without a passphrase.

Installing Your Certificate

Installation instructions depend on your platform/operating system. Contact your CA for details, or you can work with your Edgio support team. The support team will check your certificates periodically to ensure they do not expire.

EdgioSSL Content Protection

HTTPS (SSL) support is provided for both IPv4 and IPv6[1] in all Edgio PoPs. You can use SSL to secure any content Edgio delivers.

Note: During SSL negotiation between Edgio caching servers and customer origin, SNI extensions are not sent. This ensures that connection re-use is not impacted by specific hostnames being used during TLS negotiation with the origin.

SSL Delivery With Custom Certificate Option

Edgio makes it easy to procure and manage SSL certificates, including wildcard,Subject Alternative Name and Fully Qualified Domain Name (FQDN) certificates.

Both customer- and Edgio-hosted certificates can be implemented. If you want Edgio to host your existing, in-use certificates, Edgio can do so using 1024-bit encryption or higher intelligence routing. Edgio can even deploy complex custom certificates containing a mix of wildcards and FQDNs, for single or multiple subdomains.

> **Note:** There are two types of multi-use certificates:

- **Wildcard Certificates**: Normally employed to secure multiple subdomains under a single unique FQDN
- **SAN Certificates**: Usually employed to protect multiple domainnames using a single certificate

SSL & Transportation Layer Security (TLS) Offload

All incoming SSL requests are terminated on EdgioEdge Servers, improving response time by lowering the latency of the TLS negotiation between the end user and the content origin.

Support for Chunked Transfer Encoding Over SSL

Chunked Transfer Encoding via SSL improves throughput for secure uploads. Chunked Transfer Encoding enables uploads where the content length is not known in advance, forwarding data as received by the CDN rather than buffering the request.

To enable this feature, please contact Limelight Customer Service for assistance.This feature is enabled on a per-VIP basis.

Authentication

Edgio *Content Delivery* supports the use of Amazon S3 authentication for customers using Amazon S3 as their origin.

When Amazon S3 authorization is enabled, each Edgio request to Amazon S3 includes an `Authorization` header containing an access key identifier and secret access key.

To enable this feature, you must provide Edgio with your Amazon S3 access key identifier and secret access key.

> **Note:** Amazon S3 temporary credentials are not supported.

---

[1]If you need IPv6 support but have not previously requested it be enabled for your Limelight Account, please contact Edgio Support

Defensive Protection

The security features in *Content Delivery* offer a first line of defense from unwanted traffic and requests reaching our customer's web application infrastructure. Built-in defense mechanisms and security features ensure high performance and availability of HTTP and HTTPS web applications.

In a multi-layered security solution, *Content Delivery* provides the best first line of active defense and can easily be augmented with the attack detection, reporting and scrubbing capabilities of the Edgio *DDoS Attack Interceptor* product.

Defense protection features are:

- Deflect network layer attacks:
  - Reflected and brute force attacks
  - HTTP SYN flood
  - ICMP flood
- Maintain HTTP and HTTPS application performance during attack
- Filter unwanted traffic:
  - Check headers, query terms and cookies
  - Assess user location for access rights
- Mask web application infrastructure from the public internet
- Protect access to content with time-limited and individualized URLs
- Real-time status code reporting describes user and application behavior
- Real-time reporting integrates with monitoring systems
- High-performance global SSL infrastructure

Web Site and Application Acceleration

Overview

Web Site and Application Acceleration ensures that personalized, complex websites and applications are delivered to users on any device, anywhere, providing near-instant response times by optimizing and accelerating the delivery of dynamically generated website and application content and data. *Web Site and Application Acceleration* is a set of features integrated with Edgio' global Content Delivery Network and is designed to interoperate with your existing infrastructure and online application solutions.

Web Site and Application Acceleration reduces network latency by providing optimizations in the first, middle, and last miles of the Internet. When combined, optimizations for all three areas are known as "symmetric acceleration" and enable two types of optimization and acceleration to be applied:

1. Dynamic content acceleration (DCA): Optimizes the delivery of website HTML, application frameworks and the dynamic data from the origin infrastructure creating them, over a high speed, congestion-free private network of densely architected CDN Points of Presence (POPs), to the end user.
2. Static object caching and delivery: Objects that are updated infrequently are cached at the Edgio CDN edge and delivered to end users, ensuring low latency delivery and high infrastructure offload. Edgio Network's Cache Hit Ratio for cacheable content is is the highest in the industry.

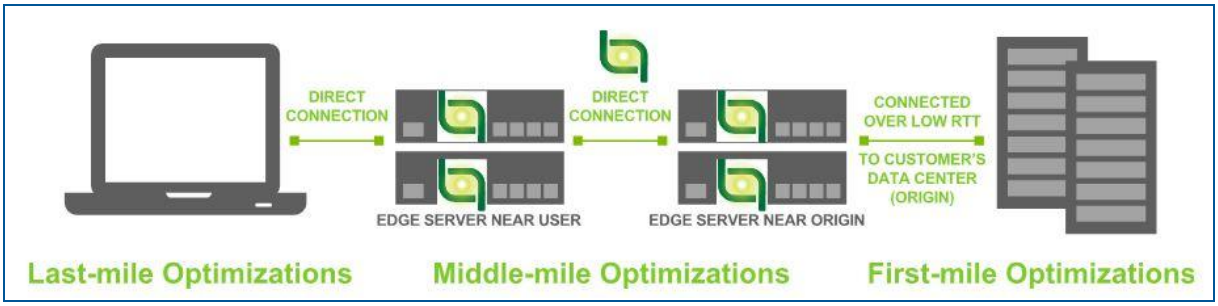

Figure 1. First, Middle, and Last Miles

Additional information is in the following sections:

- Benefits
- Why Accelerate Website Content?
- Architecture

Web Site and Application Acceleration Benefits

Web Site and Application Acceleration helps your content get delivered faster and with better availability–everything from your dynamic web pages, to JavaScript files, to rich media.

With its two levels of optimization, *Web Site and Application Acceleration* provides the following benefits:

- Increases customer conversion rates
- Increases time users spend on web site
- Increases ad revenue
- Increases customer satisfaction
- Reduces support calls
- Improves response time
- Lowers cost of ownership
- Improves infrastructure return on investments
- Provides built-in reliability and redundancy
- Scales to accommodate changing traffic patterns
- Conserves valuable internal resources by offloading to Edgio
- Enforces business rules with an advanced policy engine
- Connects to user access networks through Edgio's global network
- Detects and optimizes for mobile devices
- Shields and protects your application infrastructure from the public internet
- Deflects network layer attacks and filters unwanted traffic

Why Accelerate Website Content?

Without website content acceleration, both mobile and desktop users can have a negative experience with your website.

Studies and Statistics[1]

Studies show that slow web site performance negatively impacts an end-user's perception of a web page and the organization that owns the page. Almost half of all users expect a web page to load in two seconds or less.

A one-second delay in page load time results in 11% fewer page views and a 16% decrease in customer satisfaction. A one-second delay in page load also results in 7% loss in conversion. Moreover, users blame the retailer for this delay, and not the Internet or the service provider, thus eroding brand equity.

The longer a page takes to load, the greater the frequency of page abandonment by the user. About 80% of shoppers who are dissatisfied with web page performance are less likely to buy from the same site again. About half those users will tell their friends about their bad experience.

[1]All statistics in this section are from Sean Work, kissmetrics.com

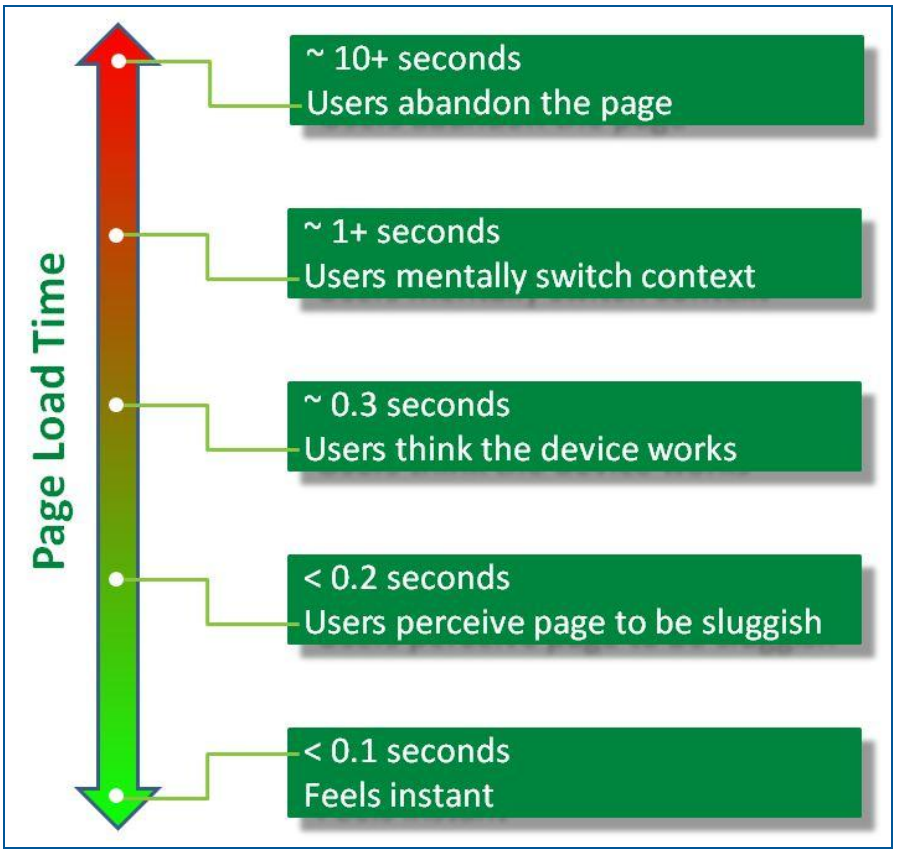

Figure 2. Page Load Time and User Responses

Expectations of Mobile Users

Mobile users expect a desktop-quality experience in terms of the performance of the mobile website or application. Moreover, the mobile experience must be consistently high across all devices. Page loads must be blazingly fast. A CDN must be able to detect a device type and location in order to provide tailored formats and fastest access speeds. *Web Site and Application Acceleration* consistently meets these requirements and expectations.

Types of Content

Website and Application content can typically be divided into two categories:

- Static content: changes infrequently or never. Is cached at the CDN edge in compressed or uncompressed format. Seldom or never requires updates from the origin. Your website's logo is an example of static content.
- Dynamic content: changes frequently and is generally classified as personalized or real-time information. The HTML used in dynamic websites is typically uniquely generated for each user that accesses the website; for example, a user's investment allocation pie chart. Therefore, dynamic pages either need to be fetched from the origin each time a page is used, or they are updated very frequently. Furthermore, a browser can only download resources (images,

JavaScript, stylesheets, etc.) after the HTML framework has been downloaded, because the HTML describes which resources are needed to make up the page. It is therefore imperative that the HTML arrives as quickly as possible, so that the delivery of additional resources is not delayed.

Benefits of Using Dynamic and Static Content Delivery

Without specialized content delivery technologies, all content, both static and dynamic, is delivered slowly thereby increasing the chances of end-users abandoning a page before it fully loads.

Web Site and Application Acceleration provides state of the art dynamic and static content delivery. For static content acceleration, *Web Site and Application Acceleration* uses configurable, high-density edge caching, resulting in an industry-leading Cache Hit Ratio and providing: lower wait times; lower page load errors; lower resource load errors; and lower costs.

Dynamic content delivery is accelerated via symmetric acceleration in the first, middle, and last miles of the content's journey from the origin server to the end user. *Web Site and Application Acceleration* accelerates the delivery of dynamic content across the Edgio global CDN, optimizing its journey across the internet.

Architecture

The following architectural elements help ensure fast, reliable communication and content acceleration with the *Web Site and Application Acceleration* product: densely architected POPs, cache hierarchies, and connection meshing.

Densely Architected POPs

Edgio employs a dense network architecture in which multiple data centers in metro locations (such as Los Angeles, London and Tokyo) are interconnected using high-capacity links and grouped around key transit locations and carrier-neutral interconnection points, using multiple high speed connections to deliver content with low-latency into last mile access networks.

These groupings of metro data centers are Edgio's points of presence (POPs) that operate together as single logical groups in each metro area. They form over 40 POPs globally, housed in over 80 individual Data Centers (DC).

All Edgio POPs are connected together with our own private fiber-optic infrastructure, managed by Edgio for Edgio customers as a media grade, highly efficient private network. Edgio controls and manages this network and its densely architected POPs, to provide industry-leading cache-hit ratios for cacheable content and ensure that cache fill requests and requests for dynamic, personalized content are delivered to the POPs serving the end user requests without suffering from the congestion present on the public Internet. Requests can then be delivered using the Edgio 10+ terabit per second (Tbps) egress capacity Content Delivery Network.

Unlike sparsely architected content delivery service providers who distribute servers within third-party network locations, Edgio's dense architecture of interconnected POPs enables each POP to contain many hundreds of servers, capable of delivering all of Edgio's services from each location. Every server used in the delivery of the *Web Site and Application Acceleration* solution is highly specified and able to deliver multiple gigabits per second directly to the end user access networks available to each POP, and every *Web Site and Application Acceleration* customer's service can be delivered from all of Edgio's POPs.

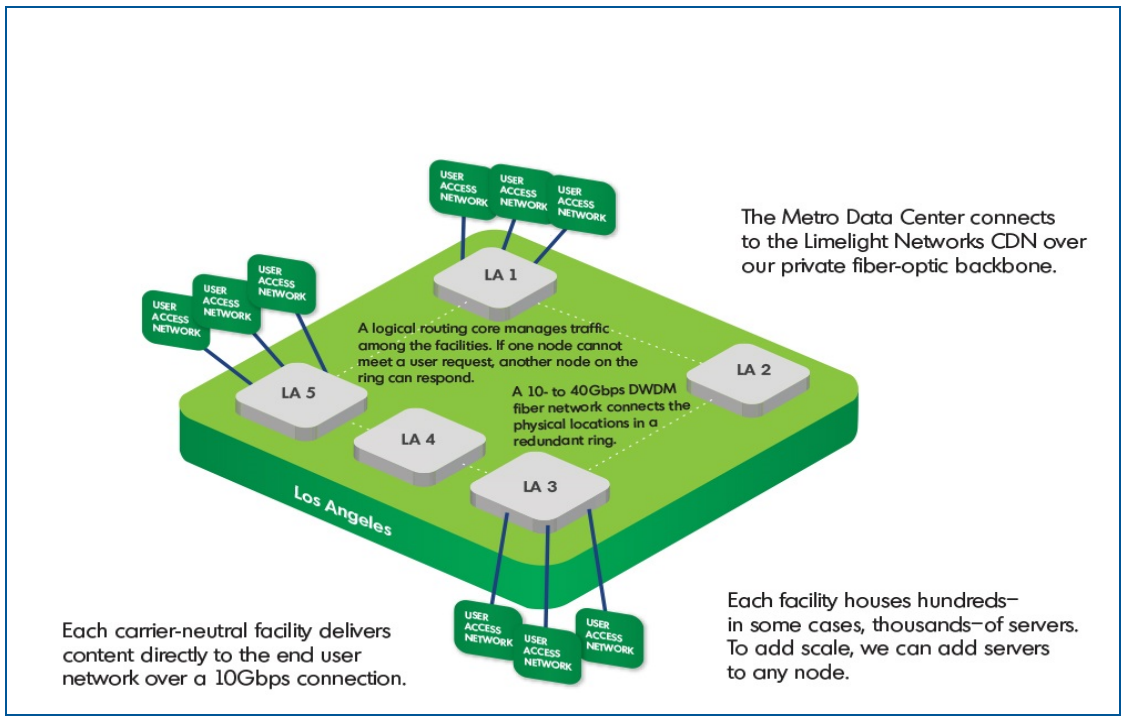

Figure 3. Metro POP Architecture

Cache Hierarchies

As all of the Edgio POPs are connected together into a single physical network, a hierarchy of caching for each website being accelerated with *Web Site and Application Acceleration* is established across the network. A Cache Hierarchy is a logical path that is defined within the Edgio network so that requests which have to be sent to the customer origin, can be directed to use the most efficient path within the Edgio network, to get there.

Hierarchies are defined for *Content Delivery* customers so that requests for dynamic content are accelerated through the Edgio network to the POP closest to the customer's content origin, from every other POP in the network, ensuring the optimization of end user requests for dynamic content.

Requests for *Content Delivery* customer's cacheable content also follow a hierarchy. These can be configured to ensure that content not already in cache is populated across the Edgio POPs as requests for it flow to the origin and back to the end user, as the request is routed through Edgio POPs on its way to the origin and back.

Connection Meshing

As connections for dynamic content flow through the Edgio network between POPs serving requests for end users and the POP closest to the customer's origin infrastructure, a "mesh" of connections is established and maintained to suit the scale and frequency of end user requests. These connections are managed by Edgio's caching and acceleration software so that they can be re-used for many end user requests and the high efficiency of Edgio's accelerated request and response flow can be delivered.

Web Site and Application Acceleration Features

The *Web Site and Application Acceleration* service provides a robust set of features that ensures optimal acceleration of your content to requesting end users.

The remainder of this section provides details about *Web Site and Application Acceleration* features. Features are grouped in the following sections:

First, Middle, and Last Mile Features

Edgio includes content delivery optimization features that speed the delivery of content over the first, last, and middle miles (see About First, Middle, and Last Miles).

See:

- First-mile optimization features: First-Mile Acceleration.
- Middle-mile optimization features: Middle-Mile Acceleration.
- Last-mile optimization features: Last-Mile Acceleration.

Flexibility and Customization Features

Custom Rules on Request

The Rules on Request feature lets you specify custom logic that is executed when requests are received by *Web Site and Application Acceleration* .

Rules typically fall into the following categories and may consist of multiple logical steps:

- Analysis of Request Elements. The URL, query strings, cookies and request headers can be analyzed for specific values (or simply for their presence in the request)
- Modification of Request Elements. Origin location, URL, query strings, cookies and request headers can be modified based on their content.

Rules can be configured to run on any request. For additional information, contact your Account Manager.

Custom Rules on Response

The Rules on Response feature lets you specify custom logic that is executed before the response is sent to the end user.

Rules typically fall into the following categories and may consist of multiple logical steps:

- Analysis of Response Elements. The URL, query strings, cookies and response headers can be analyzed for specific values (or simply for their presence)
- Modification of Response Elements. Cache Keys, response headers, and cookies can be modified based on their content.

Rules can be configured to run on any response. For additional information, contact your Account Manager.

Delivery Optimization Features

HTTP/2 Support

Content Delivery supports HTTP/2, a major revision of the HTTP network protocol that provides significant performance advantages.

- **Speed**. Overhead is reduced by header compression and binary content format
- **Mutliplexing**. Supports multiple requests over a single connection

To enable this feature, please contact Limelight Customer Service for assistance. This feature is enabled on a per-VIP basis.

ALPN (Application-Layer Protocol Negotiation)

ALPN (Application-Layer Protocol Negotiation) helps improves the performance of secure HTTP connections by eliminating multiple client-server exchanges during protocol negotiations.

Content Delivery supports ALPN (Application-Layer Protocol Negotiation) within TLS handshakes over HTTP/2. When multiple application protocols are supported on the same TCP port, ALPN allows the application layer to negotiate which protocol will be used within the TLS connection.

To enable this feature, please contact Limelight Customer Service for assistance.This feature is enabled on a per-VIP basis.

Acceleration Capabilities

Web Site and Application Acceleration has a wide range of capabilities and components that work together to guarantee lightening-fast delivery of your content to end users. These capabilities are explained in the following sections:

- Components
- Adaptive Intelligence
- Dynamic Content Acceleration and Delivery Features In Context
- Static Content Acceleration and Delivery Features In Context

Web Site and Application Acceleration Components

Web Site and Application Acceleration combines a number of technologies and optimizations in order to accelerate and optimize the delivery of complete websites and applications from your content origin to your end users. These components are:

- First-mile acceleration and optimizations
- Middle-mile acceleration and optimizations
- Last-mile acceleration and optimizations

All optimizations are features and are configurable.

About First, Middle, and Last Miles

To understand *Web Site and Application Acceleration* 's accelerations, it is helpful to understand the first, middle, and last miles:

- First mile: The path between the content origin and the Edge Server nearest the origin.
- Middle mile: The path between the Edgio Edge Server nearest the origin, to the Edgio Edge Server nearest the user, within the Edgio network.
- Last mile: The path from the Edgio Edge Server nearest the user to the user's computer or device.

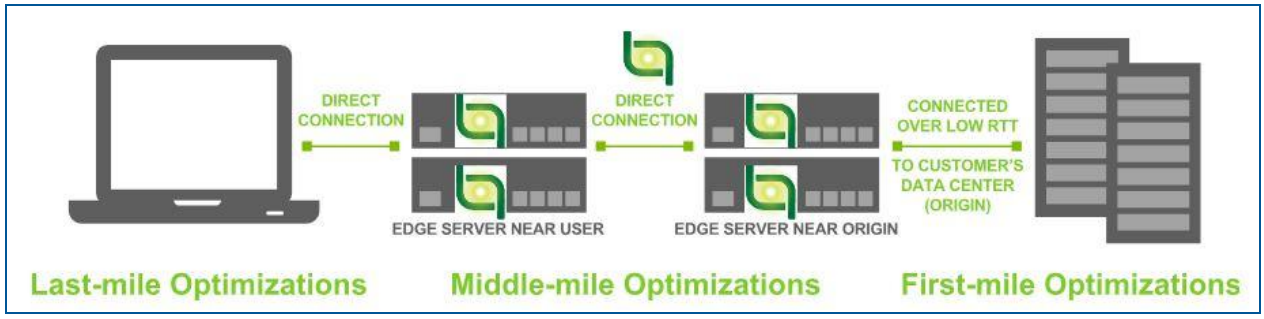

Figure 4. First, Last, and Middle Miles

First-Mile Acceleration

First-mile acceleration occurs between the content origin infrastructure (origin) and the Edgio POP and Edge Server closest to this origin. To reduce the latency and maximize origin efficiency and resource utilization when returning the requested content into the Edgio network, Edge Servers within the Edgio POP closest to the origin are used to concentrate and control connections and communication between the Edgio Network and the origin.

Web Site and Application Acceleration optimizes this connection and data flow while requesting content and scaling to meet the demands of your audience. *Web Site and Application Acceleration* uses a cache-hierarchy (see Cache Hierarchies) to co-ordinate requests across the entire Edgio network and restrict requests to your origin from one Edgio POP to ensure that the parallel load to your infrastructure is controlled.

Middle-Mile Acceleration

Middle-mile acceleration takes place between Edgio Edge Servers in the POP nearest the origin, and the Edge Servers in the POP nearest to the end-user making the request. (See About First, Middle, and Last Miles.) This middle-mile comprises the longest distance that content must travel, on its journey between the origin and the end user. In Edgio's case, this middle mile is within our private network.

The optimizations that are employed by *Web Site and Application Acceleration* in the middle-mile are designed to ensure the fastest delivery of static and personalized, dynamic content from the origin. *Web Site and Application Acceleration* determines the fastest route between the origin and end users and provides a congestion-free connection between them, optimizing the communication protocols and connection management within the Edgio network.

Middle-mile optimizations include:

- Custom cache hierarchies, mapping the best path to the origin across Edgio's private network (see Cache Hierarchies.)
- Connection meshing between POPs, ensuring low latency for all connections across the network (see Connection Meshing)
- Persistent connections for high traffic scalability (see Dynamic Content Acceleration and Delivery Features In Context and Static Content Acceleration and Delivery Features In Context)
- WAN and TCP optimization for lowest content propagation times (see TCP/IP Optimization)

Last-Mile Acceleration

Last-mile acceleration occurs between the Edgio Edge Servers in the POP nearest the user and the user's computer or device, when delivering content over the access networks, internet service provider's (ISP's) and mobile operator networks that connect them. (See About First, Middle, and Last Miles.) Edgio is directly connected to over 900 ISPs, access networks, carriers and mobile networks with connections in multiple locations, ensuring that content being requested is delivered as fast as possible to any device on any network.

Last-mile optimizations include:

- Arc Light (see Dynamic Content Acceleration and Delivery Features In Context and Static Content Acceleration and Delivery Features In Context)
- Cache Control (see Caching Dynamic Content)
- Compression
- Header control (see Header Manipulation)
- *MediaVault* content protection (see URL Protection)
- Device Detection (see Device Detection)
- Accelerated SSL offload
- Persistent connections to browser
- TCP optimizations

Adaptive Intelligence

Web Site and Application Acceleration provides adaptive intelligence capabilities that include:

- Origin offload: As much as is possible, objects are cached on Edge Servers, thereby reducing the number of hits on your origin server and improving performance.
- Connection optimizations: Connections in the first, middle, and last mile segments are optimized via connection pooling and multiplexing to optimize the delivery of uncacheable content from the content origin.
- Custom cache hierarchies
- Round-trip minimization
- Protocol maximization[1]

[1]Protocol maximization dynamically adjusts the parameters of Internet-standard protocols on a per-connection, per-delivery basis for maximum performance.

Dynamic Content Acceleration and Delivery Features In Context

This section illustrates key dynamic content acceleration and delivery features involved in a request and response flow.

Dynamic content is that content which must be generated at the time of request due to its nature, being timely, personalized or unique depending on the parameters used to request it. Dynamic content is always created and served by a content origin or application server, such as a web server interfacing with a database or a content management system. Dynamic content, by definition, cannot be cached by *Web Site and Application Acceleration*, and HTTP response headers should be set by content origins to ensure that content is correctly identified as dynamic, non-cacheable content.

Examples of dynamic content are:

- The list of items in a user's shopping basket
- A list of previously ordered goods in an e-Commerce website user profile
- The results of a search query
- Recommendations to be presented to a user based on a previous activity

Most dynamic content is HTML or similar application framework data, and describes the other resources that are needed to complete the rendering of the information being requested. It is critical that this content is delivered as fast as possible as the browser or app needs this content in order to build, request, and present the other items that make up the complete page.

Web Site and Application Acceleration uses its feature set to optimize the request and response flow of dynamic content to and from the content origin or application infrastructure in the following ways:

Step	*Web Site and Application Acceleration* Features/ Capabilities	Optimization Applied	Description	Implementation Options
DNS resolution & Initial connection	Global Content Delivery Network - Premium service. Traffic Director	CDN Scale Persistent connections to browser	Edgio services are targeted by end users through DNS resolution at request time.	Requests targeted to *Web Site and Application Acceleration* by CNAME are automatically serviced by the CDN configuration in place for that domain. Customer domains (i.e.www.-

Step	*Web Site and Application Acceleration* Features/ Capabilities	Optimization Applied	Description	Implementation Options
		Request targeting	Edgio DNS services target the appropriate POP for the request when the underlying Edgio Hostname is resolved.	mycompany.com) are CNAMED to a Edgio provided hostname (i.e. mycompany.hs.llnwd.net). Traffic Director can be utilized to control content delivery options based on user location (Geo, IP) and to balance content delivery between multiple services.
Secure Connection	SSL Offload	SSL termination	SSL is provided in all Edgio POPs and offers SSL termination at the CDN edge for end users, lowering latency of the TLS negotiation. HTTP and HTTPS protocols can be used to communicate with the customer origin.	Customer certificates or Edgio hosted certificates can be implemented for use. Single domain, Wildcard, SAN and Extended Validation certificates can be used with *Web Site and Application Acceleration*.
Object request	HTTP Methods, Origin support	Origin selection	Options for specifying origin location and header override options	Primary & backup origin servers Host-Header override Cache Hierarchies Error handling Header insertion to identify Edgio to the origin
Object request	HTTP Control, Geographic compliance, GeoIP Information, *MediaVault*	Request control Header manipulation	Requests can be analyzed for their authenticity and denied or allowed to be serviced by the CDN based on combinations of the end users geographic location and information present in the URL.	Allow or deny list countries from being able to access content via the CDN URL tokenization enables URLs to be single or limited use, time and referrer bound. Request Headers can be added to include the users GeoIP information in the request to origin.

Step	*Web Site and Application Acceleration* Features/ Capabilities	Optimization Applied	Description	Implementation Options
Object request	Advanced Rules Engine	Rules on Request	Rules on request offer the opportunity for additional conditional logic to be performed at the point of the request being made to the CDN. Rules typically fall into the following categories and may consist of a combination of logical steps (not comprehensive): -- Analysis of: Headers, URL, Query Strings, or Cookies for their presence or specific values -- Changes to: URLs, request headers, origin location, cookies.	Rules are determined and implemented during configuration.
Object request	Cache hierarchy, Middle Mile Acceleration, First Mile Acceleration	Connection meshing Origin connectivity Connection pooling TCP & WAN acceleration	Requests for dynamic content are accelerated across the Edgio network using a hierarchy of determined paths to the POP closest to the origin location. TCP optimization and WAN acceleration are used to reduce the number of Round	Cache hierarchies are established during configuration. Middle Mile acceleration is enabled by default for *Web Site and Application Acceleration* customers. Best practices are implemented for origin connectivity and can be amended as necessary for specific customer configurations.

Step	*Web Site and Application Acceleration* Features/ Capabilities	Optimization Applied	Description	Implementation Options
			Trips needed between Edgio POPs and maintain connectivity across the network. Requests from across the network are concentrated into one POP to maximize resource reuse and control requests to the origin. Connection management, reuse and TCP optimizations are used to reduce latency and Round Trips for each request to origin.	
Object response	GZIP Passthrough	Compression on-the fly	Content that cannot be compressed or is not compressed by the origin server is selected for compression at the Origin facing Cache Servers on-the-fly. The compressed object then delivered across the Edgio network faster.	Standard compression levels and best-practice sets of file types and sizes will be compressed by default. File types to be compressed can be altered as needed. More aggressive compression and options to compress larger options can be added. Large and small objects can be compressed when using HTTP or HTTPS
Object response	Middle Mile Acceleration	Connection meshing Connection pooling	Responses are accelerated across the Edgio network using the same hierarchy of determined paths that was used to reach the	Cache Hierarchies are established during configuration. Middle Mile acceleration is enabled by default for *Web Site and Application Acceleration* customers.

Step	*Web Site and Application Acceleration* Features/ Capabilities	Optimization Applied	Description	Implementation Options
		TCP & WAN acceleration	POP closest to the origin location. TCP optimization and WAN acceleration is used to reduce the number of Round Trips needed between Edgio POPs and maintain connectivity across the network.	
Object response	Advanced Rules Engine	Rules on Response	Rules on response offer the opportunity for additional, conditional logic to be performed at the point of the response being sent to the end user. Rules typically fall into the following categories and may consist of a combination of logical steps (not comprehensive) Analysis of: Headers, URL, Query Strings, or Cookies for their presence or specific values Changes to:Cache Key, response headers, cookies.	Rules are determined and implemented during configuration. Rules offer a flexible method of applying a required function. They are executed for each requested URL configuration, but being deterministic may not have to be applied each time.
Object response	HTTP Control	Header manipulation	Response headers	Established during configuration and changeable on request.

Step	*Web Site and Application Acceleration* Features/ Capabilities	Optimization Applied	Description	Implementation Options
		Cache Key manipulation	can be inserted with specific values to enable debugging and identification of content delivered by *Web Site and Application Acceleration* in the client. Cache Keys can be manipulated and modified according to specific conditions being present in the request and response details.	These are specific to the configurations that they are applied to, being used for every response for a matched URL that has been requested.
Object response	Last Mile Acceleration	Last mile TCP acceleration	TCP optimizations are applied to control the opening and maintenance of the TCP window size during object transmission, reducing latency and Round Trips needed to deliver each object. Optimizations enable data packets to be consistently kept in flight across both highly latent or congested networks, and highly efficient ones.	Implemented by default and applied on a per-request basis. Options can be selected from based on the profile of the objects being delivered by each URL pattern configured.

Static Content Acceleration and Delivery Features In Context

This section illustrates key static content acceleration and delivery features involved in a request response flow.

Static content includes non-dynamic website content (such as images, CSS, and JavaScript) and file downloads (software releases and updates, downloadable content, and so on).

Web Site and Application Acceleration utilizes its feature set to optimize the request and response flow of static content to and from the content origin or application infrastructure in the following ways:

Step	web *Web Site and Application Acceleration* Features/ Capabilities	Optimization Applied	Description	Implementation Options
DNS Resolution And Initial Connection	Global Content Delivery Network - Premium service. Traffic Director	CDN Scale Persistent Connections to Browser Request Targeting	Edgio services are targeted by end-users through DNS resolution at request time. Edgio DNS services target the appropriate POP for the request when the underlying Edgio Hostname is resolved.	Requests targeted to *Web Site and Application Acceleration* by CNAME are automatically serviced by the CDN configuration in place for that domain. Customer domains (example: www.-mycompany.com)are CNAME'd to a Edgio provided hostname (example: mycompany.hs.llnwd.net. Traffic Director can be utilized to control content delivery options based on user location (Geo, IP) and to balance content delivery between services.
Secure Socket Connection	SSL Offload	SSL Termination	SSL is provided in all Edgio POPs and offers SSL termination at the CDN edge for end users, lowering latency of the TLS negotiation. HTTP and HTTPS protocols can be used to communicate with the customer origin.	Customer certificates or Edgio-hosted certificates can be implemented for use. Single domain, wildcard, SAN and extended validation certificates can be used with *Web Site and Application Acceleration*.
Object request	HTTP Methods, Origin support	Origin Selection	Options for specifying origin location and header override options	Primary and backup origin servers Host-header override Cache hierarchies Error handling Header insertion to identify Edgio to the origin on cache-miss
Object Request	HTTP Control, Geographic Compliance, GeoIP Information, *MediaVault*	Request Control Header Manipulation	Requests can be analyzed for their authenticity and denied or allowed to be serviced by the CDN based on com-	Allow list or deny list countries from being able to access content via the CDN. URL tokenization enables URLs to be single or limited use, time bound and referrer bound.

Step	web *Web Site and Application Acceleration* Features/ Capabilities	Optimization Applied	Description	Implementation Options
			binations of the end users geographic location and information present in the URL.	Request Headers can be added to include the users GeoIP information in the request to origin.
Object Request	Advanced Rules Engine	Rules on Request	Rules on request offer the opportunity for additional conditional logic to be performed at the point of the request being made to the CDN. Rules typically fall into the following categories and may consist of a combination of logical steps (not comprehensive): -- Analysis of: headers, URL, query strings, or cookies for their presence or specific values -- Changes to: URLs, request headers, origin location, cookies.	Rules are determined and implemented during configuration.
Object request	Cache Efficiency, Content Freshness, Cache Hierarchy, Middle-Mile Acceleration, First-Mile Acceleration	Cache Calculations TTL Overrides Connection Meshing Origin Connectivity Connection Pooling TCP & WAN Acceleration	Requests for static content result in the server managing the request checking the local cache for the requested content. The location of the object is determined by a Cache Key, calculated on request and applied when content is initially cached. On cache-hit, the	Cache overrides are configured for each domain and path being accelerated and can be updated as needed. Cache hierarchies are established during configuration. Middle-mile acceleration is enabled by default for *Web Site and Application Acceleration* customers. Best practices are implemented for origin connectivity and can be amended as necessary for specific

Step	web *Web Site and Application Acceleration* Features/ Capabilities	Optimization Applied	Description	Implementation Options
			content is returned to the requesting user directly. Content freshness is determined through analysis of headers supplied by the content origin when content is initially cached. Overrides can be applied to deterministically control when content freshness checks will take place and how long content can remain in cache before the Edge Servers must make a request to check that an object is still fresh. Requests that result in a cache miss or a content freshness check being needed are subsequently accelerated across the Edgio network in the same way as requests for dynamic content are. The responses are then analyzed for cacheability and returned to the end user.	customer configurations.
Object Response	Advanced Rules Engine	Rules on Response.	Rules on response offer the opportunity for additional, conditional logic to be performed at the point of the response being sent to the end user.	Rules are determined and implemented during configuration. Rules offer a flexible method of applying a required function. They are executed for each requested URL configuration, but being deterministic may not have to be applied each time.

Step	web *Web Site and Application Acceleration* Features/ Capabilities	Optimization Applied	Description	Implementation Options
			Rules typically fall into the following categories and may consist of a combination of logical steps (not comprehensive): Analysis of: Headers, URL, query strings, or cookies for their presence or specific values. Changes to:Cache Key, response headers, cookies.	
Object Response	HTTP Control	Header Manipulation Cache Key Manipulation	Response headers can be inserted with specific values to enable debugging and identification of content delivered by *Web Site and Application Acceleration* in the client. Cache Keys can be manipulated and modified according to specific conditions being present in the request and response details.	Established during configuration and changeable on request. These are specific to the configurations that they are applied to, being used for every response for a matched URL that has been requested.
Object Response	Last-Mile Acceleration	Object Compression Last-Mile TCP Acceleration	Static content can be delivered as compressed objects to browsers that are able to de-compress them. This is commonly controlled through use of the VARY header. Compressed and uncompressed versions of the same	Implemented by default and applied on a per-request basis. Standard compression levels and best-practice sets of file types and sizes will be compressed by default. Content to be compressed can be altered as needed. More aggressive compression and options to compress larger options

Step	web *Web Site and Application Acceleration* Features/ Capabilities	Optimization Applied	Description	Implementation Options
			object can be cached and delivered. TCP optimizations are applied to control the opening and maintenance of the TCP window size during object transmission, reducing latency and round trips needed to deliver each object. Optimizations enable data packets to be consistently kept in flight across both highly latent or congested networks, and highly efficient ones.	can be added. Large and small objects can be compressed when using HTTP or HTTPS. Options can be selected from, based on the profile of the objects being delivered by each URL pattern configured.

Delivery Optimization

The SmartPurge Rest API Guide appendix includes the following:

- Request Endpoint Schema
- Error Response Schema
- Error Response Descriptions
- HTTP Status Codes
- API Client Sample Code

Cache Optimization

Cache behavior and performance directly affects your audience's content experience.

The default behavior of the *Content Delivery* cache is tuned to provide the best overall experience in most situations. However, you may want to adjust cache behavior based on your unique content and delivery requirements.

You can use a number of advanced *Content Delivery* configuration options to control cache behavior, including:

- Cacheability (TTL) Overrides
- Flexible Cache Key Manipulation
- TTL Management
- Flexible Cache Hierarchies
- Vary Header Optimization

Unless you are an advanced user, Edgio recommends you keep the default settings for these options. Please consult with Edgio Client Support if you are unsure whether an option is right for you, or which settings are best for your unique needs.

Flexible Cache Hierarchies

During *Content Delivery* setup, Edgio implements a custom cache hierarchy based on your unique requirements, delivering the best possible user experience while minimizing the number of requests to your origin.

Key factors Edgio considers when designing your cache hierarchy include:

- Content library size
- Content type(s) and average file sizes
- Audience distribution (geographic)
- Origin server location(s)
- Content access patterns (by time of day, month and season, event-based or random spikes, and so on.)

The end result is a custom cache hierarchy tuned specifically to your needs. Content is cached as closely as possible to your audience and origin request round trips are minimized.

DNS Services (Failovers & Traffic Direction)

You can use Edgio DNS Services to help balance and manage end-user requests to your origin servers and other IP resources (including requests from more than one CDN).

DNS Services includes two different highly-redundant, highly-scalable DNS-based capabilities with a global footprint:

- Failovers balance traffic loads by dynamically routeing nameserver requests to customer IP resources (such as one or more origin web servers) on an as-needed basis. You can quickly create and manage business policies via the DNS Services management console in Control.
- *Traffic Director* is a global traffic router that directs traffic based on end user IP address, nameserver geographic location, or BGP autonomous system number.

Overview

There are three main steps needed to configure DNS Services:

1. Adding and Configuring Resources: During this step you add, configure, and specify which resource(s) you wish to manage. Ultimately, the resources are the “handout answers” to end user query requests for the hostname.
2. Adding and Configuring Policies: During this step you add and configure one or more policies. The policy is the act of binding one or more resources together using a business rule for the distribution of end user requests to your added and configured resources.
3. Delegating your DNS Resource Record via CNAME: During this step you use the customer-specific hostname specified during the policy-creation process to delegate your DNS resource record via a CNAME.

> **Note:** Changes made in the DNS Services system propagate to the CDN edge in less than 5 minutes. However, policy *Freshness* values (which control the frequency of end-user name server revalidation) may dictate how quickly traffic begins to shift.

Compression On-The-Fly

You can enable the *Compression On-The-Fly* feature to instruct *Content Delivery* to compress and cache your content as needed, based on the capabilities of requesting clients.

When this feature is enabled, *Content Delivery* initially caches and delivers each content object in uncompressed form - until the first request that indicates the client will accept Gzip compression.

Content Delivery then creates a compressed version of the object “on the fly”, simultaneously caching it. Future requests for either compressed or uncompressed versions of that object are fulfilled directly from cache.

> **Note:** The maximum file size for *Compression On-The-Fly* is 2 MB. Larger files are not compressed.

Customers whose workflows require ETags to be preserved can use the GZIP feature for on-the-fly content compression.

In most cases, before compressing content to be cached, it makes sense to strip ETags to improve cache performance. For customers who use ETags as part of their workflow, the ETags can be retained for compressed content.

> **Note:** For most customers, ETag preservation is not recommended for GZIP content. If you need to enable this option, please contact your Account Manager or Solutions Engineer for assistance.

TCP/IP Optimization

The *Content Delivery* service automatically optimizes the speed of TCP/IP traffic between individual *Content Delivery* servers (“middle-mile”) and requesting clients (“last mile”).

Configuration options are available to adjust how TCP/IP optimization is performed over the last mile, with the middle and first mile options being controlled within the Edgio network. Peering connections near the origin and persistent connections can also be used to optimize traffic between Edge Servers and the origin (“first-mile”).

Analytics and Reporting

The SmartPurge Rest API Guide appendix includes the following:

- Request Endpoint Schema
- Error Response Schema
- Error Response Descriptions
- HTTP Status Codes
- API Client Sample Code

View Reports in Control

The reports provided by Control let you track overall traffic and detailed content usage for the *Content Delivery* service.

Reports are grouped by the major types of data they present. Available report titles include:

Traffic Reports	Content Reports	Storage Reports
• Traffic • EdgeFunctions Live Stats • EdgeFunctions Traffic • EdgeFunctions Status Codes	• URL Prefixes • Status Codes • Realtime Live Event Overview	• Origin Storage

Traffic Reports	Content Reports	Storage Reports
• LivePush Streaming • Live Stats • Realtime Streaming		

You can use *Content Delivery* report data for a variety of purposes, including budget planning, marketing analytics and performance analysis and troubleshooting.

Report data can be displayed in hourly, daily, weekly, or monthly increments for both predefined and custom date ranges. All report charts are interactive and let you drag to zoom in on interesting data.

You can also set up any report to be emailed to you automatically at your preferred interval.

For more information, see the *Reports Chapter* of the *Control Portal User Guide* in the secure documentation site (under *Support* > *Documentation* in *Control*).

Access Report Data via the Realtime Reporting API

The Realtime Reporting API lets you programmatically access the data behind the EdgeQuery-powered reports in Control. You can use the Reporting API to integrate this data with other systems.

For more information, see the *Realtime Reporting REST APIs User Guide* in the secure documentation site (under *Support* > *Documentation* in *Control*).

Retrieve Logs

Edgio provides log data generated by worldwide *Content Delivery* servers via the *Live Logs* service. *Live Logs* are updated throughout the day as individual server log data is received and processed.

Retrieve Download Completion Reports

Download Completion Reports and *Download Completion Geo Reports* record the aggregate daily status of HTTP downloads.

For each URL, completion reports include the number of download requests initiated, the number and percentage of downloads completed, and may include optional geographic information. Both reports are available as .csv files through your FTP Account.

> **Note:** These reports are not included in the *Content Delivery* service and must be ordered separately.

Receive Real-Time Download Completion Receipts

Download Completion Receipts are real time notifications of specific download events, and are sent in the form of HTTP `GET` requests to an IP address or URL you specify during setup.

Receipts are triggered by specific stages in the HTTP download process, and each receipt contains detailed information such as the URL of the requested object, the current download status, the IP address of the requesting client, and so on. The information is provided in predefined query terms appended to the `GET` request.

Note that if you are delivering a high volume of HTTP downloads, or expect significant download peaks, the receiving web server(s) should be able to handle the anticipated request load.

Note: *Download Completion Receipts* are not included in the *Content Delivery* service, and must be ordered separately.

Rate Limiting

If Edgio determines it is necessary, customer traffic may be rate limited (bandwidth controlled) on a per-PoP basis. Edgio reserves the right to rate limit traffic as needed to achieve the best overall experience for customers. Specific rate limiting parameters are determined in consultation with affected customers.

Rate limiting provides the following overall customer benefits:

- Service levels are maintained for all customers within individual POPs - even when user demand increases unexpectedly for specific content
- Individual user experience is improved in cases where content (such as video) must be delivered with a specified minimum bitrate, and the client is capable of taking action (such as delaying the end user, or redirecting traffic based) on an HTTP Status Code status code (see below)

For each affected POP, rate limiting parameters include:

- the affected customer Account and configuration(s)
- if needed, the targeted content (as specified by a pattern such as the file path, file name, or Regex)
- the bandwidth limitation (the throughput rate for all clients connected to that POP)
- an optional initial burst length to ignore(the number of bytes from the beginning of a response that should be exempt from rate limiting)

In cases where the maximum bandwidth has been reached, there are two configurable options:

- As new clients connect, overall throughput is adjusted for existing connections to maintain the maximum specified bandwidth, OR
- As new clients attempt to connect, existing connections are unaffected, and the new connections are rejected with a configurable HTTP Status Code (the default is 503).

Options

- **SmartPurge**. *SmartPurge* is Edgio's next generation tool for proactively removing content from cache. *SmartPurge* executes purge operations more quickly and reliably than older technologies. The advanced version of *SmartPurge*, *SmartPurge Plus*, provides additional features, including higher purge queue priority, and additional API features such as unlimited callbacks. For more information, please see Cache Purging.
- **MediaVault**. *MediaVault* is a high-performance URL authentication service. *MediaVault*'s main purpose is to help you secure your content from unauthorized viewing. For more information, please see MediaVault URL Protection.
- **Geo Compliance**. You can use the *Geo Compliance* feature to restrict access to your content by geographic area. Both IPv4 and IPv6 addresses are used when determining geographic location.This feature is ideal for managing media licenses with geographic restrictions. It's also useful for sites where advertising is a primary driver, as the audience can be constrained to the target geographies specified by the advertisers. For more information, please see Geographic URL Protection.
- **Chunked Streaming**. With *Chunked Streaming*, you can deliver chunked video content via HTTP and HTTPS in four different formats: HDS, HLS, MSS and MPEG-DASH. To use *Chunked Streaming*, you first need to chunk your content and generate the associated manifest files (*Chunked Streaming* does not perform these operations). You can host your content on your own origin servers or with *Origin Storage*. For more information, please see Configuring Chunked Streaming in the Control User Guide.

- **DNS Services**. You can use Edgio DNS Services to help balance and manage end-user requests to your origin servers and other IP resources (including requests from more than one CDN). DNS Services includes two different highly-redundant, highly-scalable DNS-based capabilities with a global footprint: Failovers balance traffic loads by dynamically routeing nameserver requests to customer IP resources (such as one or more origin web servers) on an as-needed basis. You can quickly create and manage business policies via the DNS Services management console in Control. *Traffic Director* is a global traffic router that directs traffic based on end user IP address, nameserver geographic location, or BGP autonomous system number.

Technologies

- **EdgePrism**. At the heart of the Content Delivery service is EdgePrism, the advanced caching proxy software that powers your content across the globe. Patented optimization techniques ensure that your live events, software, media, and other files are optimally delivered on every request; both cacheable and uncacheable content are delivered with the lowest latency, best reliability, and highest origin offload.
- **EdgeQuery**. EdgeQuery helps you access critical audience behavioral data with real-time reporting capability via Edgio's powerful, edge-based compute platform. Operating at the speed of digital transactions, EdgeQuery provides the insights you need to make decisions that will generate the best experience for your audience and the best results for your business.

Implementing *Content Delivery*

Understanding Your Welcome Letter

When your *Content Delivery* service is ready for use, each of the Technical Contacts you provide for your Limelight Account will receive an email Welcome Letter from the Limelight NOC (Network Operations Center).

The Welcome Letter will include any information you need to enter to activate and configure *Content Delivery* and will also provide you with contact information for the Edgio Support team, additional information on troubleshooting and escalation, and background information on maintenance notifications.

Specific configuration information to look for in your Welcome Letter includes:

Origin Hostname(also Hostname / Source Host)

The*Origin Hostname* is the hostname of your origin server, as configured in*Content Delivery* during the setup process. For example:

`origin.customer.com`

*Content Delivery*will request content from this server to fill the cache.

You can change your Origin Hostname in the Control.

Account

An *Account*is the name of a unique configuration of the *Content Delivery* service. For example:

`account_name`

You may be provided more than one Limelight Account name depending on the complexity of your requirements; if so, each Limelight Account is separately configurable.

When you log in to the Control, you will see your Limelight Account name(s) in the drop-down menu. If you have more than one Limelight Account, the Limelight Account names will be listed alphabetically in the menu.

Published Hostname(also Published URL / HTTP(s) or RTMP Prepend URL)

The*Published Hostname*is the domain name that your audience will see in published links to your content.*Published Hostname* directs content requests to the *Content Delivery* service - URLs for content you want to cache must be prefixed with the *Published Hostname*. For example:

`http://account_name.vo.llnwd.net/`

For more information on the Published Hostname, see the*Configure* chapter of the *Control User Guide*.

Host Header

Content Delivery includes the Host Header value in the HTTP `Host` header when making requests to your origin. You can use this value to block unauthorized access to content on your origin server.

If you have questions about this configuration information, or anything else in your Welcome Letter, please contact your Account Manager or Limelight Customer Service.

Note: If you are a new customer, you will receive a separate Welcome Letter with access information for the *Control* . You will also receive a separate Welcome Letter any time you order a new service or request major changes to an existing service.

Configuring *Content Delivery*

If you need to change a *Content Delivery* configuration setting provided in your Welcome Letter, or you want to take advantage of a feature described in*Key Content Delivery Features*, you can:

- Contact your Account Manager or Solutions Engineer
- Contact Limelight Customer Service directly
- If Configuration Self Service is enabled for your account(s), and the feature you want to change is customer-configurable, you can manage it directly in the Control under *Configure > Delivery*

Using CNAMEs

If you want Edgio to set up a DNS CNAME (an alias) to use as your Published Hostname, you can order the CNAME in the *Control* - just navigate to *Activate > CNAME* and enter the information requested. When your CNAME is ready, you can configure *Content Delivery* to use the new CNAME as your Published Hostname. You can do this yourself in the configuration wizard in the *Control*, or ask your Account Manager for assistance.

Caching Based On Query Terms

When caching an object, *Content Delivery* normally uses the entire URL, including all query terms, as a unique identifier for the object.

As a result, if your origin serves the same content in response to URLs with different query terms, *Content Delivery* will cache each variation as a separate object.

If you want to change this default behavior, you can configure *Content Delivery* to ignore any query terms you specify.

Testing & Tuning

Updating Links

Before your content can be delivered by the *Content Delivery* service, you need to ensure all links to your content use your Published Hostname.

If you don't know your Published Hostname, you can find it in your *Welcome Letter,* or by viewing your *Content Delivery* configuration in the *Control*. To view your configuration, navigate to *Configure > Deliver* - your Published Hostname will be displayed along with other configuration information in the Deliver Configurations table. If you have multiple configurations and you don't know which one to use, please contact your Account Manager.

If you are already using a DNS CNAME as the domain name in your links, you don't need to change the links - you can just point the CNAME to the Published Hostname when you are ready to go live. Otherwise, you will need to change each link to use the Published Hostname as the domain name.

To ensure a smooth transition, you may want to begin by changing a small number of links and testing them individually. A conservative approach is to start with links on less popular (low traffic) or test pages, then graduate to links in popular (high traffic) locations.

Monitoring Performance

During the transition, you may also want to monitor the before-and-after performance of test object and/or important content using a measurement service, such as Dynatrace (formerly Compuware APM and Gomez).

Going Live

We recommend that any "go-live" activity be carefully planned so that traffic for your content is directed to the CDN in a controlled manner, and can be monitored by your operational teams as this happens.

If you are using your own hostname and CNAMEing this to Edgio in order to go live, we suggest using a low TTL value in your DNS zone when initially making the change to Edgio. This will help you roll-back quickly in the event that you have any issue with the configuration of the *Content Delivery* service. Once you are comfortable with the service, this TTL can be increased to reduce the number of DNS lookups needed to resolve requests for your content to the *Content Delivery* service.

As traffic is directed to the Edgio *Content Delivery* service, cache fill will begin as requests are made for content. For small content libraries, cache fill will be quick and the CDN efficiency you can achieve will be reached quickly. For larger content libraries, the cache-fill can take some time, and will be affected by the rate at which your users request content from across your library. Migrations from other CDN services to Limelight or the addition of Edgio as a secondary CDN to serve content alongside another CDN would be expected to reach optimal cache efficiency over some time, depending on the rate of requests across the entire content library.

In both cases, the efficiency you are achieving can be monitored using the CDN efficiency report available in the Control portal. This report can be used to determine if content is caching as expected. The File Type report will also indicate the cache efficiency being achieved for each type of file being delivered by *Content Delivery* and can be used in combination with Edgio's Customer Facing Troubleshooting Headers, to assess if your configurations are working as expected.

The Content reports available in the Control Portal including the File Errors and Status Code reports can be used to assess how well the service is operating, alongside the CDN Efficiency traffic report.

There are a range of Diagnostic tools available in the Control Portal which can be used to perform troubleshooting and run diagnostic tests during your "go-live" period, as described below.

Edgio Advanced Services are able to provide assistance and monitoring during a go-live process with you. Please contact your Account Manager for more information and to discuss the use of Advanced Services.

Viewing Order Status

For orders you previously submitted using the *Activate* section of the *Control Portal*, you can check order status at any time using the *Support* section of the portal. Just log in and navigate to Support > Order Status. The Order Status Summary table lists each of your orders, both current and historical. For each order, you can see:

- Which service was ordered, and by whom
- When the order was submitted and completed
- The current status of the order
- The order "ticket number" for reference when communicating with the Edgio Support team

Accessing Online Documentation

Online documentation for the *Content Delivery* service can be found in the secure documentation site (under *Support* > *Documentation* in *Control*)

There you'll find links a wealth of technical information, including:

- The most current version of this document
- The latest *Content Delivery* Release Notes
- Best Practices for managing the *Content Delivery* cache

Troubleshooting

You can use the *Control Portal* to perform a variety of troubleshooting and diagnostic tests, including:

- DNS lookups for specific hostnames
- MTRs (traceroutes with pings)
- Geographic lookups
- Cache Hit Ratio analysis for specific URLs

To access these tools, navigate to *Support > Diagnostic Tools* and select the desired test in the *Diagnostic Overview* table header.

Requesting Support

Before you go live, Edgio recommends you direct questions to your Account Manager, who will ensure you the right person is available to provide whatever help you need.

Once you are live, you can contact the Edgio Support team 24 x 7 for technical assistance with your *Content Delivery* service. See the Edgio Customer Supportpage for support email addresses and phone numbers.

If you have questions about which *Content Delivery* features you have purchased, or want to inquire about additional services, your Account Manager will be happy to assist.

Finally, you can log in to the *Control Portal* at any time to check order status, submit and track support tickets, access documentation, and view detailed reports on the traffic handled by your *Content Delivery* service.[1]

1 -
[1]- please refer to your Control Portal Welcome Letter for your login credentials

Managing *Content Delivery*

Following Operational Best Practices

Purging Cached Content

Objects are normally updated in or removed from cache during “freshness checks” with your origin. For a given object, a freshness check is initiated when a request has been made for the object, and the object’s TTL (Time To Live) has expired.

In general, setting object TTL is the best and most efficient way to manage cached content. For example, a news site may need to provide rapid updates to a breaking video story. The video can be updated in cache as quickly as desired by assigning it a low TTL value using an HTTP response header. In most cases, there is no need to remove the video from cache directly.

However, there are special cases where content needs to updated on the next user request or even proactively removed from cache as soon as possible. This is known as "purging the cache" or just "purging". Examples of when purging might be necessary include:

- Text is misspelled in the caption of a newly-uploaded video, and you need to update the video in cache as quickly as possible.
- You discover that some of your cached content is infringing a copyright and need to delete the content from cache as soon as possible.
- You lose a contract with a content provider and are obligated to delete the provider’s content from your cache as soon as possible.
- During a full website update, when you need to quickly update many related website objects (images, text, video, etc.) at the same time.

Edgio's *SmartPurge* executes purge operations more quickly and reliably than older technologies. The advanced version of *SmartPurge*, *SmartPurge Plus*, provides additional features, including higher purge queue priority, and additional API features such as unlimited callbacks.

You can access *SmartPurge* through either the Control Portal or the SmartPurge REST API. For more on purging, please see the Edgio SmartPurge Data Sheet and Intelligent High-Speed Purging.

Legacy Purge Notes:

- Using the legacy *Purge* tool consumes significant resources. Submitting large numbers of purge requests, or purging large numbers of objects, can impact the ability of other customers to use the *Purge* tool.
- When submitting a purge request, Edgio recommends you provide a fully-qualified URL for each object, and avoid using regular expressions wherever possible. Mistakes in regular expressions can impact overall *Purge* performance and may remove unintended content.
- If you plan to purge 500 or more objects in a short period of time, please contact Edgio Support for assistance.

Tracking *Content Delivery* Usage

Content Delivery provides many different ways to access and view your data. The major types and sources of data are covered in detail in in Analytics & Reporting:

- **Browser Access**
  - see Viewing Reports in the *Control*
- **Realtime Reporting API Access**
  - see Accessing Report Data via the Reporting API

- **File Retrieval**
  - **Server Logs** - see Retrieving *Content Delivery* Logs
  - **Download Completion Reports** - see Retrieving Download Completion Reports
- **Notifications**
  - see Receiving Real-Time Download Completion Receipts

Monitoring the User Experience

Using Measurement Services

In addition to tracking content usage, you may want to subscribe to a service specifically designed to monitor and report on the end-user experience. It's not necessary to configure *Content Delivery* to use these services - simply provide the Published URLs for objects you want to track.

Popular monitoring services include those from Catchpoint, Cedexis, Soasta, Dynatrace (Gomez) and Keynote.

Capturing User IP & Geo Information

If you want to capture and analyze user IP address and geographic location yourself, *Content Delivery* can include this information in special request headers when making requests to your origin. For more information on these custom headers, please see Geo IP Info in Headers to Origin and True-Client-IP in Headers to Origin.

Changing *Content Delivery* Configuration

Please see Configuring *Content Delivery* for details.

Accessing *Content Delivery* APIs

The *Content Delivery* service includes APIs that give you programmatic access to the following features:

- **Realtime Reporting API** - gives you access to any realtime data available in reports in the *Control*
- ***SmartPurge* API** - lets you submit purge requests and check their status

Online documentation for these and other APIs is available in the secure documentation site (under *Support* > *Documentation* in *Control*)

Please note that the credentials you need to access an API are unlikely to be the same as your *Control* credentials. To request credentials for a specific API, please contact Edgio Support.

Viewing IP Allow Lists

You can now view the current IP addresses of Edgio Edge Servers to update your firewall, without logging into the Control. The IP allow list returned from the API call is versioned, so you can compare the current version with the prior version to see how the list has changed.

You can use either the REST API or the RSS feed to view the IP allow list.

API Specification

The endpoint returns a versioned list of allowed IP addresses, expressed in JSON.

HTTP Method: GET

URL: `https://control.llnw.com/aportal/api/ipam/getIpAllowList.do`

Required Request Headers

Header	Description
`X-LLNW-Security-Principal`	Caller's user name
`X-LLNW-Security-Timestamp`	Current system UTC time in milliseconds
`X-LLNW-Security-Token`	HMAC-256 digest calculated using the caller's API shared key on: HTTP Method + URL + timestamp

Sample Request

`https://control.llnw.com/aportal/api/ipam/getIpAllowList.do`

Sample Request Headers

`X-LLNW-Security-Principal: sample_user`

`X-LLNW-Security-Timestamp: 1465226821474`

`X-LLNW-Security-Token:b62f93cdde95e94b814ba824430a25cfd31fc13f485f201f6878754caf6f0493`

Sample JSON Response

```
{
       "ipAllowList":["1.9.58.160/27",
       "37.238.255.224/27",
       "41.63.64.0/18"],
       "version":1
}
```

RSS Feed

Subscribe to the following:

`https://control.llnw.com/aportal/support/documentation/iprssfeed`

Each IP address is in a `content:encoded` element within a parent `item` element. Example:

```
<item>
       <content:encoded>69.28.128.0/18</content:encoded>
</item>
```

You can also view the recent IP allow list in the documentation site. Log into the *Control*, then select the Documentation link at the bottom of the page. From there, click the Edge Server IP Allow List link in the green navigation panel on the left.

Receiving Real-Time Download Completion Receipts

Download Completion Receipts are real time notifications of specific download events, and are sent in the form of HTTP `GET` requests to an IP address or URL you specify during setup.

Receipts are triggered by specific stages in the HTTP download process, and each receipt contains detailed information such as the URL of the requested object, the current download status, the IP address of the requesting client, and so on. The information is provided in predefined query terms appended to the `GET` request.

Note that if you are delivering a high volume of HTTP downloads, or expect significant download peaks, the receiving web server(s) should be able to handle the anticipated request load.

> **Note:** *Download Completion Receipts* are not included in the *Content Delivery* service, and must be ordered separately.

Retrieving Download Completion Reports

Download Completion Reports and *Download Completion Geo Reports* record the aggregate daily status of HTTP downloads.

For each URL, completion reports include the number of download requests initiated, the number and percentage of downloads completed, and may include optional geographic information. Both reports are available as .csv files through your FTP Account.

> **Note:** These reports are not included in the *Content Delivery* service and must be ordered separately.

Retrieving Logs

Edgio provides log data generated by worldwide *Content Delivery* servers via the *Live Logs* service. *Live Logs* are updated throughout the day as individual server log data is received and processed.

Viewing Reports in *Control*

The reports provided by Control let you track overall traffic and detailed content usage for the *Content Delivery* service.

Reports are grouped by the major types of data they present. Available report titles include:

Traffic Reports	Content Reports	Storage Reports
• Traffic • EdgeFunctions Live Stats • EdgeFunctions Traffic • EdgeFunctions Status Codes • LivePush Streaming • Live Stats • Realtime Streaming	• URL Prefixes • Status Codes • Realtime Live Event Overview	• Origin Storage

You can use *Content Delivery* report data for a variety of purposes, including budget planning, marketing analytics and performance analysis and troubleshooting.

Report data can be displayed in hourly, daily, weekly, or monthly increments for both predefined and custom date ranges. All report charts are interactive and let you drag to zoom in on interesting data.

You can also set up any report to be emailed to you automatically at your preferred interval.

For more information, see the *Reports Chapter* of the *Control Portal User Guide* in the secure documentation site (under *Support* > *Documentation* in *Control*).