TechNote – AltitudeCDN™ OmniCache Plugin for Brightcove

Version 1.4

AltitudeCDN OmniCache is a robust proxy cache optimized to deliver enterprise video content with the lowest possible impact on your network infrastructure and security. The AltitudeCDN OmniCache Plugin for Brightcove allows you to integrate OmniCache as a reverse proxying solution for Brightcove Video Cloud, minimizing network bandwidth and enhancing playback quality.

This guide describes how to implement the OmniCache Plugin for Brightcove with Brightcove Video Cloud.

Contents

Introduction

AltitudeCDN™ OmniCache is a robust video proxy cache optimized to deliver enterprise video content with the lowest possible impact on your network infrastructure and security. AltitudeCDN OmniCache improves video playback performance by reducing latency and buffering, and minimizes the amount of bandwidth required for delivering rich media content across the WAN.

When used with Brightcove Video Cloud, OmniCache is easily configurable as a software-defined network (SDN) reverse proxying solution that supports HLS, the primary streaming format used by Brightcove Video Cloud, and MP4 progressive download.

Reverse proxying allows you to rewrite URIs for video content source to become the URI for the OmniCache streaming proxy server. This approach has several advantages over a standard, forward proxying solution:

- Your proxy configurations do not need to change.
- Brightcove-related domains do not need to have an enterprise trusted TLS certificate, if HTTPS support is not required.
- You can manage the solution with Brightcove Video Cloud Studio for each player, as needed.

> ***Note:*** *The Brightcove player is built upon the Brightcove-sponsored Video.js open source player framework. The general techniques described in this guide are adaptable to other Video.js-based players, provided the appropriate changes in the plugin installation steps are made.*

Requirements

To use the OmniCache Plugin for Brightcove, you need the following:

Item	Recommendation
OmniCache	• OmniCache v1.5.0 or later – The OmniCache Plugin for Brightcove needs the viewer protocol matching reverse proxying functionality introduced with this version of OmniCache.
Brightcove Video Cloud Studio	• You must be able to reconfigure Brightcove players in Brightcove Video Cloud Studio.
Brightcove Players	• Brightcove players 5.5.0 through 5.19.1 have been tested with the OmniCache Plugin for Brightcove, but any Brightcove player that supports plugins should work with the solution.

Solution Summary

To use the OmniCache Plugin for Brightcove, you need to:

1. Review the OmniCache Plugin for Brightcove outlined in this TechNote. For more information, see OmniCache Plugin for Brightcove Reference.
   The OmniCache Plugin for Brightcove does the following:
   - Queries OmniCache via XHR to determine its availability.
   - Queues requests to set the player src() until the response from OmniCache is received, or until the timeout is reached.
   - Once the OmniCache response is received or the timeout elapses, the queued src() requests are released:
     - If OmniCache is available, the HLS or MP4 URIs flowing through src() are rewritten to point to the OmniCache reverse proxies.
     - If OmniCache is not available, the src() requests are processed normally, and re-writing is disabled.
2. Configure reverse proxying in OmniCache. For more information, see Configure the Proxy Servers in OmniCache.
3. Configure routing in OmniCache. For more information, see Configure the Routing Engine in OmniCache.
4. Configure caching in OmniCache. For more information, see Configure the Cache Engine in OmniCache.
5. Configure and activate the OmniCache Plugin for Brightcove in Brightcove Video Cloud Studio. For more information, see Configure and Activate the OmniCache Plugin for Brightcove.

OmniCache Plugin for Brightcove Reference

This section describes the OmniCache Plugin for Brightcove that was developed for use with Brightcove Video Cloud.

> ***Note:*** *The OmniCache Plugin for Brightcove is also hosted at the following location: http://livetools.ramp.com/omnicache/plugin/multicastplugin.allinone.js.*

Functionality

The OmniCache Plugin for Brightcove has the following functionality:

1. The OmniCache Plugin for Brightcove is configured within Brightcove Video Cloud when you add the plugin to a player, using simple JSON that provides:
    - The host name for OmniCache.
    - An optional timeout for responses from OmniCache.
    - An optional verbose flag for troubleshooting.
2. The OmniCache Plugin for Brightcove instantiates itself as a player plugin.
3. The OmniCache Plugin for Brightcove front-ends the src() method with a replacement that checks to see if the status of OmniCache is known:
    - If the status is unknown, queues the src() invocations.
    - If the status is known, rewrites URIs to point to OmniCache for HLS and MP4 files as needed.
4. Performs an asynchronous XHR request to OmniCache on its HTTPS port:
    - If OmniCache does not respond correctly, either timing out or sending malformed response JSON, URI rewriting is disabled.
    - Upon response or timeout, the src() method is released (any queued invocations are processed first), then player.ready() is called.
    - Stub code is included to handle checking the browser's IP address against VPN CIDRs. This allows you to turn off rerouting to OmniCache for VPNs, if needed. Using this code requires customizing the OmniCache Plugin for Brightcove reference implementation.
    - The OmniCache Plugin for Brightcove also includes a backup timer, as some browsers do not handle XHR timeouts correctly when there is network blackholing of TCP traffic to the target host. This can cause a delay of approximately 30 seconds until the operating system times out the TCP open. The backup timer avoids this delay by releasing the src() method much earlier.
5. While the code distinguishes between the HTTP and HTTPS reverse proxies, this is for historical reasons only. Current Brightcove players use viewer protocol matching URIs, which can be either HTTP or HTTPS depending on the page embedding context. As a result, both HTTP and HTTPS are required, and are now set by a single "proxy" parameter.

OmniCache Plugin for Brightcove Reference Code

```
var OmniCache4VideoJS =
{
        start: function (player, options)
        {
                this.player = player;
                this.saveSrc = player && player.src;
                this.options = options;
                this.timeout = (typeof options.timeout=="number")?options.timeout:2000;
                this.verbose = options.verbose;
                this.verbose && console.log("OmniCache plugin has started - v1.4");
                this.reverseProxy =
                {
                        http: options.http,
                        https: options.https,
                        match: options.proxy
                };
                var field,prefix;
                if (this.reverseProxy.match)
                {
                        prefix = "";
                        field = "match"
                        this.reverseProxy.http = this.reverseProxy.https = this.reverseProxy.match;
                }
                else { prefix = field = (this.reverseProxy.https?"https":"http"); prefix+=":"; }
                this.statusUri = prefix+"//"+this.reverseProxy[field]+"/";

                var xhr = new XMLHttpRequest();
                var boundListener = this.statusListener.bind(this, xhr);
                xhr.open("get", this.statusUri, true);
                xhr.setRequestHeader("Accept","application/json");
                xhr.addEventListener("readystatechange", boundListener, false);
                xhr.addEventListener("load", boundListener, false);
                xhr.addEventListener("error", boundListener, false);
                xhr.addEventListener("timeout", boundListener, false);
                xhr.timeout = this.timeout; // only allowed on async calls
                this.statusTimer = setTimeout(boundListener, this.timeout+500); // backup timer
                xhr.send();

                // immediate response, active upon receipt
                this.done = false;
                this.queue = [];
                this.player.src = this.src.bind(this);
        }, // end start

        statusListener: function (xhr, event)
        {
                if (!this.done)
                {
                        if (event)
                        { // not timer
                                if (event.type=="readystatechange")
                                {
                                        if (this.statusTimer)
                                        {
                                                clearTimeout(this.statusTimer);
                                                this.statusTimer = null;
                                                this.verbose && console.log("OmniCache external backup
timer cancelled");
                                        }
```

```
                                        this.verbose && console.log("OmniCache readyState
"+xhr.readyState);
                                        if (xhr.readyState==4)
                                        { // done
                                        }
                                        // else wait
                                }
                                else
                                { // assume terminal state
                                        this.done = true;
                                        if (event.type=="load")
                                        { // success
                                                var json;
                                                try { json = JSON.parse(xhr.response); } catch (e) {}
                                                if(json && json['X-Forwarded-For']) {
                                                    console.log("OmniCache detected X-Forwarded-For:
"+json['X-Forwarded-For']);
                                                }
                                                console.log("OmniCache detected client IP:
"+(json?json.Ip:"undefined"));

                                            // placeholder: May want to check the client IP (or X-
Forwarded-For) to see if it is part of a VPN CIDR
                                            //   if so, you may want to NOT use OmniCache. You can do
this by setting
                                            //      this.reverseProxy.match = this.reverseProxy.https =
this.reverseProxy.http = "";
                                        }
                                        else
                                        { // assume failure: error or timeout
                                                console.log("OmniCache failure on "+event.type+",
status:"+xhr.status);
                                                // invalidate reverseProxy (or just skip reassignment of
src?)
                                                this.reverseProxy.match = this.reverseProxy.https =
this.reverseProxy.http = "";
                                        }
                                        /*while*/ if(this.queue.length>0) { this.src(this.queue[0]);
this.queue.shift(); }
                                        this.player.ready();
                                }
                        }
                        else
                        { // timer = failure
                                this.verbose && console.log("OmniCache external backup timer fired");
                                this.timerFired = true;
                                this.statusListener(xhr, {type:"timeout"})
                        }
                }
                else { this.verbose && console.log("OmniCache event blocked: "+(event?event.type:"no
event")); }
                return 0;
        }, // end statusListener

        src: function (sources)
        {
                var i, isVideo, isCached, changed, srcarr;
                if (sources)
                {
                        if (!this.done)
                        {
                                this.queue.push(sources);
```

```
                                return;
                        }
                        // sources could be single object or array of objects
                        if (!(sources instanceof Array)) srcarr = [sources]; else srcarr = sources;
                        var re4cache = new RegExp("/http(?:https|s)?/"); // /http/, /https/, /httphttps/
                        var re4match = new RegExp("^//");
                        for (i=0; i<srcarr.length; i++)
                        {
                                isVideo = (srcarr[i].src && (srcarr[i].src.indexOf('.m3u8')>=0 ||
srcarr[i].src.indexOf('.mp4')>=0));
                                isCached = (srcarr[i].src && srcarr[i].src.match(re4cache));
                                changed = false;
                                if (this.done && isVideo && !isCached)
                                {
                                        this.verbose && console.log("OmniCache src was:
"+srcarr[i].src);
                                        changed = true;
                                        if (this.reverseProxy.http && srcarr[i].src.match(/^http:/))
                                        {
                                                srcarr[i].src =
srcarr[i].src.replace('http://','http://'+this.reverseProxy.http+'/http/');
                                        }
                                        else if (this.reverseProxy.https &&
srcarr[i].src.match(/^https:/))
                                        {
                                                srcarr[i].src =
srcarr[i].src.replace('https://','https://'+this.reverseProxy.https+'/https/');
                                        }
                                        else if (this.reverseProxy.match &&
srcarr[i].src.match(re4match))
                                        {
                                                srcarr[i].src =
srcarr[i].src.replace('//','//'+this.reverseProxy.match+'/httphttps/');
                                        }
                                        else changed = false;
                                }
                                else
                                {
                                        this.verbose &&
console.log("OmniCache:"+(this.done?((isVideo?"":" not")+" video"+(isCached?"":" not")+"
alreadyUpdatedToOmniCache"):"not ready"));
                                }
                                if (this.verbose)
                                {
                                        if (!changed) console.log(".... unchanged: "+(srcarr[i].src ||
"undefined"));
                                        else console.log(".... updated to OmniCache: "+srcarr[i].src);
                                }
                        }
                        // passing an array results in each member coming back in turn, array of 1 will
cause infinite loop
                        this.saveSrc.call(this.player, (srcarr.length==1)?srcarr[0]:srcarr);
                }
        } // end src
};
function OCInterface(options)
{
        OmniCache4VideoJS.start(this, options);
}

videojs.plugin('omnicache', OCInterface); //OmniCache4VideoJS.start.bind(OmniCache4VideoJS));
```

Configure the Proxy Servers in OmniCache

Configure both HTTP and HTTPS reverse proxy servers in OmniCache. As Brightcove uses protocol matching URIs of the form //, which do not permit alternate port numbers, the following ports must be used with the OmniCache reverse proxy server:

- HTTP – 80
- HTTPS – 443

For more information, see the *AltitudeCDN OmniCache Deployment Guide* and *AltitudeCDN OmniCache Reference Guide*.

Configure the Routing Engine in OmniCache

Configure the routing behavior in OmniCache. As Brightcove inserts absolute URIs into its HLS playlists, the Routing Engine must rewrite these URIs to point to OmniCache. Routing requests to multiple OmniCache nodes is supported, but beyond the scope of this document.

For more information, see the *AltitudeCDN OmniCache Deployment Guide* and *AltitudeCDN OmniCache Reference Guide*.

Configure the Cache Engine in OmniCache

Configure the caching behavior in OmniCache:

- For .m3u8 files, use forced caching TTL , as Brightcove prohibits caching these files:
    - For on-demand content, create a rule where the TTL is set to 30 seconds.
    - For live content, create an additional rule where the TTL set to 1 second.
- For .ts segments, you can use a large default TTL. Brightcove does not specify any default, so you can change the OmniCache TTL default from 1 hour to 24 hours.

Cache Engine Rules Example

The example below contains rules that reflect the caching behavior noted above. Also, consider the following:

- If Brightcove changes host names, those also need to be changed in the caching rules.
- If OmniCache is used exclusively with Brightcove, the rules can be broadened to match all host names, as only Brightcove traffic is ever routed to OmniCache (for example, creating a rule that covers all of .*brightcove*).

```
            "rules":[
                {
                    "request":{
                        "matches":[
                            {}
                        ]
                    },
                    "action":{
                        "matches":[
                            {
                                "routing":"standard"
                            }
                        ]
                    }
                },
                {
                    "request":{
                        "matchesMode":"or",
                        "matches":[
                            {
                                "host":".*brightcove.*",
                                "path":".*"
                            }
                        ]
                    },
                    "action":{
                        "matches":[
                            {
                                "contentType":".*video/MP2T.*",
                                "cacheMode": "honorResponseTtl",
                                "defaultTtl":86400
                            },
{
                                "contentType":".*video/mp4.*",
                               "cacheMode": "honorResponseTtl",
                                "defaultTtl":86400
                            }
                        ]
                    }
                },
                {
                    "request":{
                        "matchesMode":"or",
                        "matches":[
                            {
                                "host":".*brightcove.*",
                                "path":".*"
                            }
                        ]
                    },
                    "action":{
                        "matches":[
                            {
                                "contentType":".*application/vnd.apple.mpegurl.*",
                                "routing":"standard",
                                "cacheMode": "honorResponseTtl",
                                "defaultTtl":300    Note: make this 1 if live content expected
                            }
                        ]
                    }
                }
```

Configure and Activate the OmniCache Plugin for Brightcove

To configure and activate the OmniCache Plugin for Brightcove in Brightcove Video Cloud Studio:

> ***Note:*** *You can use the OmniCache Plugin for Brightcove with any Video.js-based players that you wish to use with OmniCache.*

1. Access Brightcove Video Cloud Studio.
2. Click Home, then select the Players option on the page. The Players page appears.
3. Select the name of the Brightcove player you want to use with OmniCache.
4. Scroll to the Plugins section, then click Edit to add a custom plugin.
5. Select JavaScript.
6. Create an entry for the OmniCache Plugin for Brightcove URI (using the // protocol matching format). For example:
   ```
   http://livetools.ramp.com/omnicache/plugin/multicastplugin.allinone.js
   ```
7. Click Name, Options (JSON).
8. Create an entry for the name of the custom plugin. For example:
   ```
   omnicache
   ```
9. Create a JSON configuration entry for the custom plugin (recall that both HTTP at port 80 and HTTPS at port 443 must exist in the OmniCache reverse proxy configuration). For example:
   ```
   {
      "proxy": "omnicachedemo.ramp.com",
      "verbose": true,
      "timeout": 2000
   }
   ```
   Where:
   - proxy – Specifies the host[:port] of the OmniCache proxy, where the host must match the SSL certificate name for the HTTPS reverse proxy.
   - timeout – Specifies the timeout value for the status query made to OmniCache, in milliseconds.
     Default value: 2000
   - verbose – Specifies whether to display most log messages.
     Valid values: true/false
     Default value: false

   > ***Note:*** *A former version supported HTTP and HTTPS. These are still supported, but are deprecated as current versions of the Brightcove player use protocol matching URIs. These URIs require specifying a proxy with reversehttp on port 80 and reversehttps on port 443 to work in all possible embedded use cases.*
10. Publish the player and test.
    If you view the JS console, you should see:
    - Requests for HLS and MP4 assets that are coming to your OmniCache reverse proxy.
    - Update messages when URI rewrites occur.

```
OmniCache plugin has started - v1.4
2017-03-17 08:40:56.353 index.html?videoId=5320160215001:1335 OmniCache external backup timer cancelled
2017-03-17 08:40:56.354 index.html?videoId=5320160215001:1335 OmniCache readyState 2
2017-03-17 08:40:56.354 index.html?videoId=5320160215001:1335 OmniCache readyState 3
2017-03-17 08:40:56.354 index.html?videoId=5320160215001:1335 OmniCache readyState 4
2017-03-17 08:40:56.354 index.html?videoId=5320160215001:1335 OmniCache detected client IP:
192.168.54.215
2017-03-17 08:40:56.532 index.html?videoId=5320160215001:1335 OmniCache src was:
https://secure.brightcove.com/services/mobile/streaming/index/master.m3u8?videoId=5320160215001&pubId=52
70290535001&secure=true
2017-03-17 08:40:56.533 index.html?videoId=5320160215001:1335 .... updated to OmniCache:
https://omnicachedemo.ramp.com/https/secure.brightcove.com/services/mobile/…ng/index/master.m3u8?videoId
=5320160215001&pubId=5270290535001&secure=true
2017-03-17 08:40:56.533 index.html?videoId=5320160215001:1335 OmniCache src was:
https://brightcove.hs.llnwd.net/e1/uds/pd/5270290535001/5270290535001_5320186186001_5320160215001.mp4?pu
bId=5270290535001&videoId=5320160215001
2017-03-17 08:40:56.533 index.html?videoId=5320160215001:1335 .... updated to OmniCache:
https://omnicachedemo.ramp.com/https/brightcove.hs.llnwd.net/e1/uds/pd/5270…_5320186186001_5320160215001
.mp4?pubId=5270290535001&videoId=5320160215001
2017-03-17 08:40:56.533 index.html?videoId=5320160215001:1335 OmniCache src was:
https://brightcove.hs.llnwd.net/e1/uds/pd/5270290535001/5270290535001_5320190050001_5320160215001.mp4?pu
bId=5270290535001&videoId=5320160215001
2017-03-17 08:40:56.533 index.html?videoId=5320160215001:1335 .... updated to OmniCache:
https://omnicachedemo.ramp.com/https/brightcove.hs.llnwd.net/e1/uds/pd/5270…_5320190050001_5320160215001
.mp4?pubId=5270290535001&videoId=5320160215001
2017-03-17 08:40:56.533 index.html?videoId=5320160215001:1335 OmniCache src was:
https://brightcove.hs.llnwd.net/e1/uds/pd/5270290535001/5270290535001_5320188197001_5320160215001.mp4?pu
bId=5270290535001&videoId=5320160215001
2017-03-17 08:40:56.533 index.html?videoId=5320160215001:1335 .... updated to OmniCache:
https://omnicachedemo.ramp.com/https/brightcove.hs.llnwd.net/e1/uds/pd/5270…_5320188197001_5320160215001
.mp4?pubId=5270290535001&videoId=5320160215001
2017-03-17 08:40:56.533 index.html?videoId=5320160215001:1335 OmniCache src was:
http://brightcove.vo.llnwd.net/e1/uds/pd/5270290535001/5270290535001_5320186186001_5320160215001.mp4?pub
Id=5270290535001&videoId=5320160215001
2017-03-17 08:40:56.533 index.html?videoId=5320160215001:1335 .... updated to OmniCache:
http://omnicachedemo.ramp.com/http/brightcove.vo.llnwd.net/e1/uds/pd/527029…_5320186186001_5320160215001
.mp4?pubId=5270290535001&videoId=5320160215001
2017-03-17 08:40:56.533 index.html?videoId=5320160215001:1335 OmniCache src was:
http://brightcove.vo.llnwd.net/e1/uds/pd/5270290535001/5270290535001_5320190050001_5320160215001.mp4?pub
Id=5270290535001&videoId=5320160215001
2017-03-17 08:40:56.533 index.html?videoId=5320160215001:1335 .... updated to OmniCache:
http://omnicachedemo.ramp.com/http/brightcove.vo.llnwd.net/e1/uds/pd/527029…_5320190050001_5320160215001
.mp4?pubId=5270290535001&videoId=5320160215001
2017-03-17 08:40:56.533 index.html?videoId=5320160215001:1335 OmniCache src was:
http://brightcove.vo.llnwd.net/e1/uds/pd/5270290535001/5270290535001_5320188197001_5320160215001.mp4?pub
Id=5270290535001&videoId=5320160215001
2017-03-17 08:40:56.533 index.html?videoId=5320160215001:1335 OmniCache: not video not
alreadyUpdatedToOmniCache
2017-03-17 08:40:56.533 index.html?videoId=5320160215001:1335 .... unchanged: undefined
2017-03-17 08:40:56.533 index.html?videoId=5320160215001:1335 OmniCache: not video not
alreadyUpdatedToOmniCache
2017-03-17 08:40:56.533 index.html?videoId=5320160215001:1335 .... unchanged: undefined
2017-03-17 08:40:56.533 index.html?videoId=5320160215001:1335 OmniCache: not video not
alreadyUpdatedToOmniCache
2017-03-17 08:40:56.533 index.html?videoId=5320160215001:1335 .... unchanged: undefined
2017-03-17 08:40:56.535 index.html?videoId=5320160215001:1335 OmniCache: video alreadyUpdatedToOmniCache
2017-03-17 08:40:56.535 index.html?videoId=5320160215001:1335 .... unchanged:
https://omnicachedemo.ramp.com/https/secure.brightcove.com/services/mobile/…ng/index/master.m3u8?videoId
320160215001&pubId=5270290535001&secure=true
```