Wowza MediaCache Deployment Architecture

Version 1.4

www.wowza.com

Abstract

Wowza Media Server provides the functionality to serve on-demand content for large-scale deployments. Functioning within Wowza Media Server, MediaCache enables content upload and management through a centralized point, preventing the need for additional hardware, complex content management, bandwidth, network resources, and reducing latency. This document describes a high-level architecture, system considerations, and network layout appropriate for large-scale deployment of video on demand (VOD) streaming that uses the MediaCache AddOn and the HTTP functionality provided by it.

Wowza Media Systems, the leader in media server software, provides reliable, scalable, and secure live and on-demand streaming. Wowza Media Server software enables organizations of all sizes to engage millions of viewers worldwide by delivering the highest quality video and audio to any connected device.

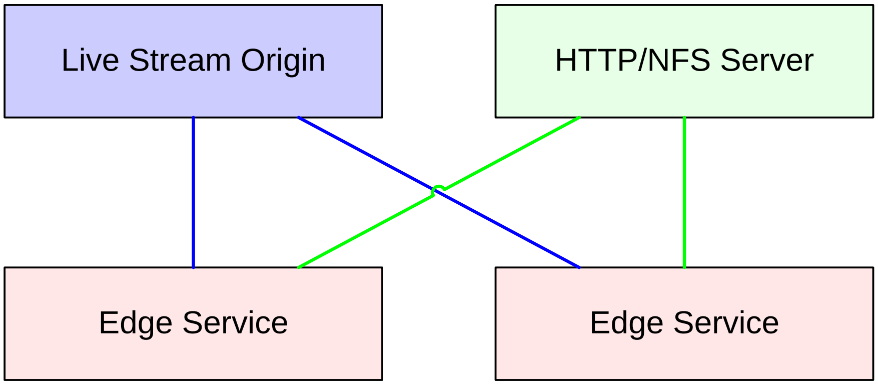
Introduction

Video content increases in resolution and file size, even as more distribution across public and private networks is required. Consequently, making media management as simple as possible becomes increasingly important. MediaCache AddOn for Wowza Media Server provides a simple and effective mechanism to achieve this goal in an extremely cost-effective manner.

This document describes a high-level architecture, system considerations, and a network layout appropriate for large-scale deployment of video on demand streaming using MediaCache AddOn and the HTTP functionality provided. It will outline considerations for network and content management, content access and distribution available within MediaCache, scaling of content stores, and local cache sizing. The final section of this paper will present some high-level deployment options.

High-Level Architecture

The high-level architecture for deploying a large-scale streaming service for both live and video on demand content is simple and consists of three distinct areas: Live Stream Origin, HTTP Server, and Edge servers.

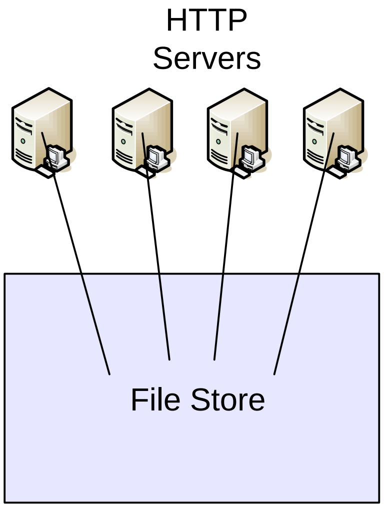


Any one of these functional areas can be expanded to provide resilience, scale, and high performance. When using Wowza Media Server, there are several AddOns that can provide scalability. This document will focus on how to use the Wowza Media Server MediaCache AddOn to scale video on demand streaming.

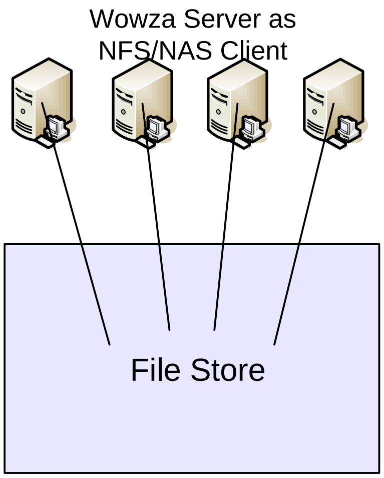
MediaCache AddOn Overview

The Wowza Media Server MediaCache AddOn can be used to retrieve content via HTTP requests from either a compliant HTTP server or from a local file location. The method of use does not alter how MediaCache manages content or how it makes content available to clients that are connecting.

When using HTTP functionality, the Wowza Media Server MediaCache AddOn requires that the HTTP servers support range requests (which are part of the HTTP/1.1 specification) and have sufficient I/O to service requests from a significantly large number of edge servers. Extremely large deployments should provide adequate storage and performance to a cluster of web servers with access to the same file store or to multiple file stores that are mirrored. The high-level diagram below shows how HTTP servers can be used in front of a file store, which then allows MediaCache to access content.



The following high-level diagram shows how MediaCache can be used to access a file store directly.



The method used to access the file store depends on the specific deployment in question and how the file store can be made available to MediaCache.

If a large number of clients will be requesting different content at the same time, the disk storage required on the edge servers running Wowza Media Server with MediaCache AddOn must be extremely fast in order to service all these clients with diverse needs. The disk storage must also be sized correctly to take into account asset sizes, asset churn, and how often assets will be accessed.

A Wowza Media Server can be tuned for specific environments so as to be more efficient with regard to content size. This customization can increase network efficiency and reduce impact on the HTTP store. When large disk arrays are deployed, onboard disk cache and OS-level caching make large reads possible and efficient.

For example:

```
            File size in the file store – 2 GB (2,147,483,648 bytes)
            Default MediaCache Request size – 256000 bytes
            Number of MediaCache access requests from Wowza Media Server – 8389
```

If the default Request size is increased to 2,097,152 bytes (2 MB), then the number of requests for an asset from Wowza Media Server drops from 8,389 to 1,024.

To get the best performance from Wowza Media Server and the file store, it is essential to determine the average asset size when using this approach.

The following is a simple calculation model that can be used to determine asset size:

```
      HTTP Request size = Average asset size / 1000
```

It should be noted that the minimum HTTP Request size should be 256,000 bytes and the maximum should be 2,000,000 bytes to yield the best performance. The Request size can be configured via the `DefaultBlockSize` property parameter.

To calculate edge storage size requirements (I/O speed is a different problem), make the following assumptions:

```
Average asset size is 450 MB (500 Kbps for 2 hours).
Average number of unique assets watched per day is 200.
```

The edge disk requirement to fit all assets for 1 day is 88 GB. Assuming that growth occurs at 5 percent per month over 18 months, then 212 GB of edge storage should be deployed.

To make sure that there is sufficient I/O and seek speed, deploy a set of RAID 1 or RAID 5 disks (depending on the operational requirements). Both of these disk configurations offer similar speeds and redundancy (parity). The number of RAID disks to use depends on the number of Wowza Media Servers that will be deployed in a location.

NOTE

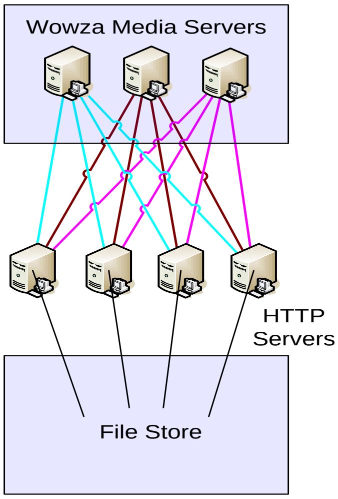
When the assigned cache space in MediaCach is full and no expiration can be performed, the Wowza Media Server will try to get assets from the defined source as they are requested. This will cause performance to degrade. Wowza Media Server will do this on a per-block basis, and this can cause a significant increase in the number of HTTP/File access requests, depending on the number of assets being requested that do not fit.

When using MediaCache, the I/O speed of the cache space is highly important. For content greater than or equal to 212 GB, the use of RAID or solid-state disk (SSD) sets is essential. Wowza Media Server performance metrics show that 2 x 36 GB, serial-attached SCSI (SAS), 15K RPM hard drives with RAID 0 (striping) enable 700 concurrent connections to 400 files. In this configuration, each file will stream at 512 Kbps with no data loss. This metric can be applied to MediaCache because assets that are downloaded from the file store are essentially accessed like any other VOD asset.

When deploying a MediaCache, cache location is important to balance both read and write rates. In normal operations, the number of write operations exceeds the number of read operations; however, it is very important to determine the write rate that will be available and how sustainable this rate will be. A good rule to follow is to assume that 10% of the performance of the cache location will require writes. If the overall throughput of the storage location can be determined—200Mbps, for example—then the write rate should be set to 20Mbps. Setting the write rate too low will cause either a significant bottleneck to the MediaCache system or any retrieved assets will be corrupted when they are written. The write rate can be configured via the `WriteRate` property within the `MediaCacheStore` definition.

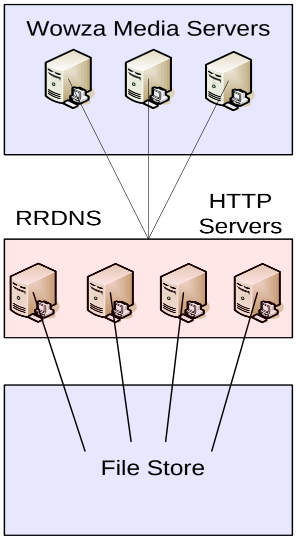
Example Wowza Media Server Deployment

Wowza Media Server can be configured to use multiple HTTP servers for the same content in a load-balancing manner or for different content. The most common deployment method is to load-balance across multiple HTTP servers when multiple requests are received for the same content, as shown in the following high-level diagram.



The MediaCache AddOn can be configured for HTTP timeouts, so if one HTTP server fails, service should continue, though initial requests for content may take a little longer during the timeout period.

An alternative deployment strategy is to use Round-robin DNS (RRDNS) or Geographic DNS for multiple HTTP servers. Operationally, using one of these alternatives has some advantages. If a failure occurs, a change in DNS can resolve the issue without the need to change any Wowza Media Server configurations.

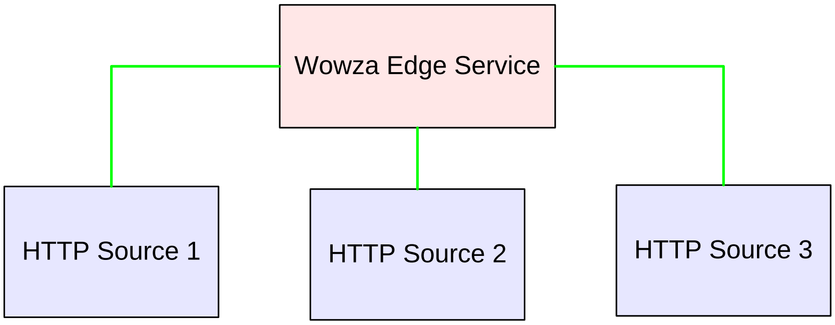


Changes to the MediaCache configuration require that the Wowza Media Server be restarted, so this should be considered when deploying the service.

Multi-Location HTTP Servers

When using MediaCache, it is possible to deploy multiple HTTP source servers and have them referenced independently in the content request that is received by Wowza Media Server. This is an aliasing mechanism that is configurable so that you can partition by different customers or by content type (movies, songs, advertising, and so on).

The following diagram shows a Wowza Media Server that connects to three independent HTTP servers for different content. The content for each of these servers is cached in the same way as in any other MediaCache deployment, so sizing must be considered.



These different HTTP sources would be referenced in any request made to Wowza Media Server. The following examples show how they could be accessed:

```
rtmp://[wowza-ip-address]/vod/mp4:http1/mycustomerfile.mp4
rtmp://[wowza-ip-address]/vod/mp4:http2/mytypefile.mp4
rtmp://[wowza-ip-address]/vod/mp4:http3/myscalefile.mp4
```

The aliases shown above (http1, http2, and http3) would be defined in the MediaCache configuration and could point to a specific HTTP server.

Summary

MediaCache AddOn for Wowza Media Server provides a simple and effective method to scale an on-demand service. It is important to understand the options provided with MediaCache AddOn and how they relate to any content that is being deployed.

Points that should be considered when deploying MediaCache AddOn include:

- Centralized network capacity
- Centralized storage capacity
- Storage growth rate
- Average asset size
- Asset churn over a 30 day period
- Edge cache capacity and I/O speed
    - Specific write rate