Low Cost Video on Demand Platform Based on Open Source Software

Jernej Rožac, Klemen Pečnik, Matevž Pogačnik

University of Ljubljana, Faculty of electrical engineering, Tržaška 25, 1000 Ljubljana
E-mail:jernej.rozac@ltfe.org, klemen.pecnik@ltfe.org, matevz.pogacnik@ltfe.org

Abstract

Video on demand servers are complex, expensive and inappropriate to be used in low-budget environments. This work proposes an approach to use an open source web server and an open source content management system as a video on demand platform for public web purposes. Pseudo-streaming over the hyper text transfer protocol is introduced with the supporting web server features. A practical implementation of the solution is provided with the extensions made to Wordpress content management system.

1 Introduction

Video on demand (VOD) is a service enabling the end-users online access to video files. There are numerous VOD scenarios, online services and software platforms supporting VOD, such as Wowza Media Server [1] or Adobe Flash Media Server [2]. In web environments VOD streams are delivered by Real Time Messaging Protocol (RTMP). Common VOD servers have the embedded support for data rights management (DRM) and bandwidth control.

For general purposes of web environments, it is possible to use alternative solutions based on web servers. This paper focuses on a pseudo-streaming concept using the Hyper Text Transfer Protocol (HTTP) [3]. It presents an idea of using a regular web server and a Content Management System (CMS) used as a web VOD platform, where all entities of the solution are open source or free. The rest of the paper is structured as follows.

Next section explains the pseudo-streaming concept and introduces the architecture solution for the pseudo-streaming environment. The main entities of the architecture are introduced and explained.

Section three describes the configuration of the web server for pseudo-streaming of video files. It explains the issues that need to be solved to achieve the end-user real VOD experience. Further the appropriate solution based on Apache Web Server [6] is given.

Fourth section explains the use of Wordpress CMS [17] as a DRM based on a practical implementation. It explains the modifications that need to be made to the selected CMS.

Last section presents opportunities and drawbacks of such VOD solution. In addition it exposes some future issues that need to be solved.

2 System Architecture

The solution proposed in this paper is based on open source or freely available software and on pseudo-streaming concept. Pseudo-streaming concept is a concept of progressive download of video files over the HTTP protocol with the end-user experience of a real video streaming. To support such experience, regular VOD requirements must be fulfilled:

1. **Direct access to any part of video files**. It enables the handling of time shift requests of the client player.
2. **Access control**. Video files have to be protected according to single user permissions.
3. **Minimalisation of network traffic.** There must not be generated more network traffic than the minimal traffic needed to download the selected video contents.

The proposed VOD solution is based on three main entities – web server extensions for pseudo-streaming, a CMS and a client player. The architecture matching the requirements described above is shown on Figure 1, where the arrow shows the user request flow starting from the client player.

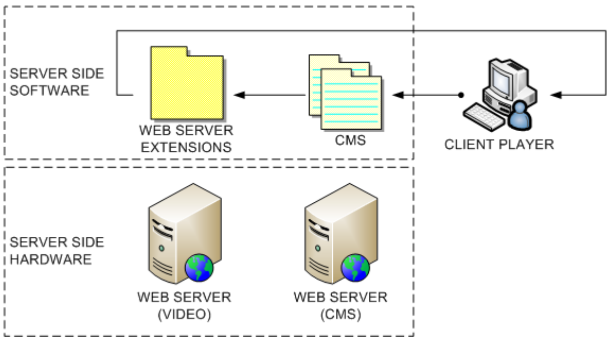


Figure 1. The solution architecture.

The architecture hardware is in general based on two separate web servers. However a single web server can also be used.

Web server extensions to support pseudo-streaming represent then main entity of the solution. There are three main extensions needed to fulfill the pseudo-streaming requirements. The extensions enable the web server to handle the response of progressive download of parts of video files from custom bytes offset which maps to download of video files from a custom time onwards. This fulfills the end-user perception of video files being streamed no matter that files are progressively downloaded. In addition the web server has to support the access control to the files with bandwidth control. Related details are explained in next section.

The second main entity is a CMS to handle the management and access control of video files. A custom CMS can be used with some extensions. A CMS has to be integrated with a web server handling the pseudo-streaming of video files to support the access control.

A client player is the third entity. As a major part of client players supports the pseudo-streaming, client player does not represent an issue which would need special attention. There are existing web clients players that can be used for free, such as JW Player [4] or Flowplayer [5].

The proposed architecture solution was implemented and tested for Flash Video (FLV) and MP4 video files. Other video files types are not supported. However FLV and MP4 are compatible with most of the available end-user devices. Next section describes the details regarding the configuration of a web server to act as a pseudo-streaming server.

3 Video streaming server

Streaming video files over the HTTP protocol equals to a progressive download of parts of video files. To make pseudo-streaming equal to real video streaming, basic VOD requirements presented in the previous section have to be fulfilled:

1. **Access of parts of video files**. Web server has to support the progressive download of custom parts of files. This is the main feature that already makes possible the pseudo-streaming.
2. **Access control**. Access to video files has to be limited according to access control policy. Web server has to implement a mechanism to verify the access to video files over the CMS.
3. **Bandwidth control**. Download bandwidth when accessing web pages is regularly the maximum bandwidth available by the network. In case of downloading video files, which represent large data, there has to be set a bandwidth limit per user per file for files being downloaded. The bandwidth has to be equal to the maximum bitrate of the video currently being downloaded. Limiting the bandwidth avoids exceeded network traffic without limiting the live access to video files.

A web server has to be extended with appropriate modules to handle the requirements presented above. To fulfill the requirements several third-party modules for different web servers exist. Pseudo streaming is possible on all popular web servers, such as Apache [6], Lighttpd [7], Nginx [8] and IIS [9]. There are existing extensions for all mentioned web servers. However the focus in this paper will be on the configuration of the Apache web server which was selected due to its significant market share [15] .

3.1 Apache web server

Apache web server is a robust and flexible web server with a robust core and the support for third-party modules. Figure 2 shows the high level architecture of the Apache web server modules. Each module can be deployed to the modules container and enabled in the web server configuration. Apache core is responsible for HTTP client interaction, while special handling is done through modules. A module accepts the forwarded core request and returns a response. More modules can be activated when processing a user request.

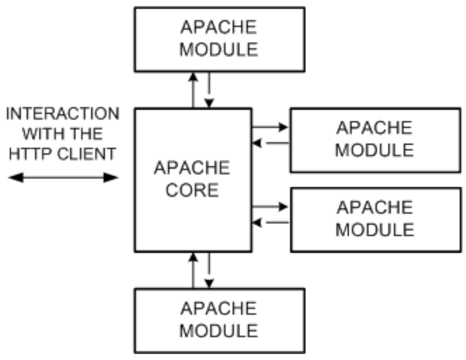


Figure 2. Apache modules high level architecture.

To fully support the pseudo-streaming compliant with all the previously exposed issues, the Apache web server has to be installed on a custom operating system and has to implement selected third-party modules:

1. **Modules "mod_flvx" and "mod_h264"**. Modules enable the pseudo-streaming of FLV and MP4 files. When requesting a video file with the URL parameter "start" containing the information of the requested position of video file, the server responds with the part of the file from the position requested onwards. Modules are easy to install and configure and do need special attention.
2. **Module "mod_auth_token".** Enables the access control of video files over the CMS, where CMS generates URL access links with time limited access validity [14]. This module can be used for all files in general.
3. **Module "mod_bw".** This module enables the bandwidth control. It supports custom configurations for separate file types and containing folders [13].

3.2 Video files limitations

Both, MP4 and FLV need to contain key frames that make possible for client player to request appropriate part of the file when an end-user requests a video file from certain time on. Time shifting is possible only for locations containing a complete key frame. This is a drawback of pseudo-streaming. As more key frames are present, the more accurate time shifting is possible, where a video file containing more key frames has a higher size and consequently a higher bitrate.

3.3 Limiting the access to video files

Access to video files has to be controlled by the CMS. In this case the CMS has the role of video DRM. The mechanism that supports the integration of the web server access and CMS is implemented in the authentication token module. Figure 3 shows the authentication mechanism.

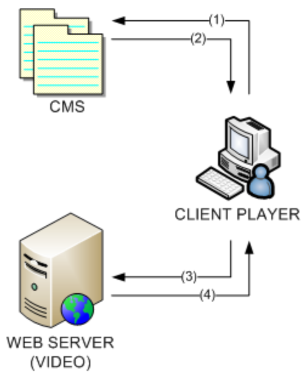


Figure 3. Token based authentication mechanism.

Initial video request is sent to the CMS, which checks the user credential to access the requested file (1). In case of invalid credentials, CMS returns an error message with the HTTP error code 403. Otherwise the CMS responses with a direct video access link with a valid token (2). A client player makes a direct request to a web video server to access the video (3). Authentication module of the web server checks the token validity and, if the token is valid, it replies with the progressive video download (4).

The time limited access token is calculated based on an access secret string and Message-Digest Algorithm (MD5), where the access secret string is stored in the CMS and in the web server configuration and can be modified at any time. The access token is calculated as a MD5 hash of a concatenated string of the access secret string, filename and current time in hexadecimal format. The direct video access address contains the current time, filename and the hash token.

3.4 Bandwidth control

Bandwidth control is done via the bandwidth control module. To support the video quality selection, for each video file there need to exist separate files for each quality category. Quality categories can be set arbitrarily. However there need to exist separate root folders for each quality category, where each folder has different bandwidth limitation according to the minimum bandwidth needed to transport the file without interruptions.

3.5 Video quality adjustments

Depending on the device type from which users access the video content, video quality can be adjusted. Web server checks the data present in the HTTP request header and determinates the device type and the video quality to be delivered. The solution proposed to handle the video quality adjustment is to store multiple copies of the same content encoded with different coding profiles. On the contrary simultaneous live transcoding of multiple files is not appropriate for multi user environment with large number of simultaneous accesses.

Offline video files transcoding can be done with VideoLAN (VLC) [20], which is open source software for mostly all available operating systems. In compare with professional transcoders, such as EPISODE [21], the video quality of the video files transcoded with professional encoders is slightly better. However VLC is a good choice for low-budget environments.

4 Management system and the solution in practice

According to the proposed solution, a custom CMS can be used to manage the data rights on video files. As the solution is based on dynamic tokens redirects, CMS can be installed and configured on the same server which serves video files or on a custom third-party web server. To make the solution work it is needed to ensure the end-user network access to both servers.

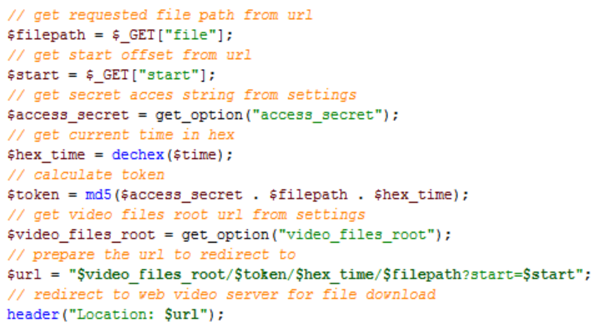
For the practical implementation an Apache web server was installed and configured according to the preferences and requirements explained in the previous section. To support the DRM, the Wordpress CMS was used. However any other CMS system that supports third-party modules can be used. Wordpress was selected as it is released under the General Public Licence (GPL), version 2 [19]. In addition it offers application programming interfaces (API) to develop third-party modules (called Wordpress plugins) to interact with the main system [18] and already supports users and contents management and content access control.

To support the VOD DRM a purpose-built plugin was developed. The plugin enables the CMS editors to enter video files locations for video content and to grant the access to selected users. The plugin extends the CMS with three additional screens:

1. **Plugin settings screen**. It enables the storage of plugin settings.
2. **Video search screen**. It shows the list of video files using the system search filters.
3. **Video data screen**. It shows the video details. If the web server on which Wordpress is installed serves also video files, the plugin handles also the video files upload. Otherwise it shows the input box to enter the video file path on web server serving video data. In addition it enables the management of users' rights to access the video.

4.1 Video files access control

As described in previous section, direct access to video files is not possible. In addition the access to video files is possible only by using valid time-dependent access tokens. The developed plugin handles the video files access request, calculates the access tokens and redirects the request to the video file storage address with the appropriate token. Figure 4 shows the token redirect procedure written in PHP programming language.



```php
// get requested file path from url
$filepath = $_GET["file"];
// get start offset from url
$start = $_GET["start"];
// get secret acces string from settings
$access_secret = get_option("access_secret");
// get current time in hex
$hex_time = dechex($time);
// calculate token
$token = md5($access_secret . $filepath . $hex_time);
// get video files root url from settings
$video_files_root = get_option("video_files_root");
// prepare the url to redirect to
$url = "$video_files_root/$token/$hex_time/$filepath?start=$start";
// redirect to web video server for file download
header("Location: $url");
```

Figure 4. Wordpress token redirect procedure in PHP.

5 Conclusion

The integration of the Apache web server and Wordpress CMS into a working VOD solution shows that it is possible to avoid regular VOD server complexity for streaming video over the internet and to replace the complex VOD solutions with two correctly integrated open source entities. In addition it is confirmed that the end-user experience is the same with the HTTP protocol as it is when using the RTSP protocol. Use of HTTP protocol enables the solution to work also in the environments where RTSP traffic is blocked for any reason.

The solution proposed is robust and stable as it is based on the web server and CMS with the largest market share. As the solution is stable and flexible, there are two main drawbacks – the requirement to supply the video files with key frames and the restriction to use client players supporting the HTTP streaming. However the key frames can be automatically injected into files in the process of video conversion. Regarding the client players, a major part of video players does not have problems with playing HTTP pseudo-streams.

Future work should focus on automatic conversion of video files. Furthermore additional research is needed in field of the automatic switching to lower video quality in case of insufficient bandwidth. In addition there has to be done research in field of protecting video files of being downloaded and saved. This is an issue for RTSP streams also. However tools supporting the download and saving of HTTP pseudo-streams are easy to use in compare with tools for saving RTSP streams.

Acknowledgement

The operation that led to this paper is partially financed by the Slovenian research Agency, grant No. P2-0246.

References

[1] “Wowza Media Server”, Internet: http://www.wowza.com/media-server [Aug. 27, 2012]

[2] “Adobe Flash Media Server”, Internet: http://www.adobe.com/products/flash-media-server-family.html [Aug. 27, 2012]

[3] “Video Delivery: HTTP Pseudo-Streaming”, Internet: http://www.longtailvideo.com/support/jw-player/jw-player-for-flash-v5/12534/video-delivery-http-pseudo-streaming [Aug. 27, 2012]

[4] “JW Player”, Internet: http://www.longtailvideo.com/players/jw-flv-player [Aug. 27, 2012]

[5] “Flowplayer”, Internet: http://flowplayer.org [Aug. 27, 2012]

[6] “Apache Web Server”, Internet: http://httpd.apache.org [Aug. 27, 2012]

[7] “Lighttpd Web Server”, Internet: http://www.lighttpd.net [Aug. 27, 2012]

[8] “Nginx Web Server”, Internet: http://wiki.nginx.org/Main [Aug. 27, 2012]

[9] “IIS Web Server”, Internet: http://www.iis.net [Aug. 27, 2012]

[10] O. Dragoi: “The Conceptual Architecture of the Apache Web Server”, Internet: http://www.voneicken.com/courses/ucsb-cs290i-wi02/papers/Concept_Apache_Arch.htm [Aug. 27, 2012]

[11] “H264 Streaming module”, Internet: http://h264.code-shop.com/trac [Aug. 27, 2012]

[12] “Flash streaming with mod flvx”, Internet: http://www.mosalov.com/wiki/Flash_streaming_with_mod_flvx [Aug. 27, 2012]

[13] “Bandwidth Limiter For Apache”, Internet: http://bwmod.sourceforge.net [Aug. 27, 2012]

[14] “Token based URI access”, Internet: http://code.google.com/p/mod-auth-token [Aug. 27, 2012]

[15] “Usage of web servers for websites”, Internet: http://w3techs.com/technologies/overview/web_server/all [Aug. 27, 2012]

[16] “Stream your videos with standard HTTP servers”, Internet: http://flowplayer.org/plugins/streaming/pseudostreaming.html [Aug. 27, 2012]

[17] “Wordpress”, Internet: http://wordpress.org [Aug. 27, 2012]

[18] “Wordpress pugins”, Internet: http://wordpress.org/extend/plugins [Aug. 27, 2012]

[19] “GNU General Public Licence, version 2”, Internet: http://www.gnu.org/licenses/gpl-2.0.html [Aug. 27, 2012]

[20] “VideoLAN”, Internet: http://www.videolan.org [Aug. 27, 2012]

[21] “EPISODE”, Internet: http://www.telestream.net/episode/overview.htm [Aug. 27, 2012]