AIS3

教育部先進資通安全實務人才培育計畫

112年度新型態資安實務暑期課程

軟體安全 S6

Fuzzing N-days in MiniWeb http server

林映辰、陳彥凱、黃晴威、李玟毅

目錄

- 研究動機
- 目標
- 背景知識
- CVE-2020-29596

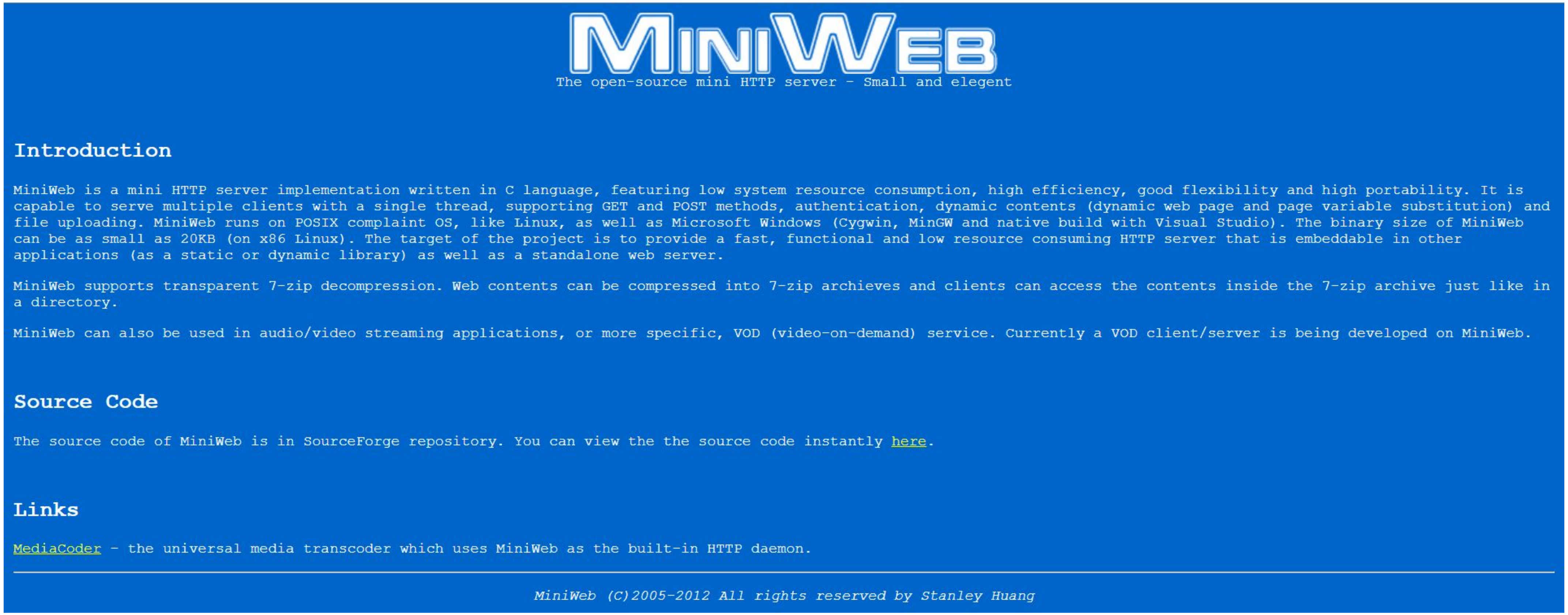
動機

研究動機

- 聽完 NiNi 講師講完 fuzzing 的主題後, 想親自操作 fuzzer
- 使用 fuzzing 找(新)漏洞
- 重現漏洞, 分析漏洞成因
- 修補漏洞

目標

miniweb



The open-source mini HTTP server - Small and elegent

Introduction

MiniWeb is a mini HTTP server implementation written in C language, featuring low system resource consumption, high efficiency, good flexibility and high portability. It is capable to serve multiple clients with a single thread, supporting GET and POST methods, authentication, dynamic contents (dynamic web page and page variable substitution) and file uploading. MiniWeb runs on POSIX complaint OS, like Linux, as well as Microsoft Windows (Cygwin, MinGW and native build with Visual Studio). The binary size of MiniWeb can be as small as 20KB (on x86 Linux). The target of the project is to provide a fast, functional and low resource consuming HTTP server that is embeddable in other applications (as a static or dynamic library) as well as a standalone web server.

MiniWeb supports transparent 7-zip decompression. Web contents can be compressed into 7-zip archieves and clients can access the contents inside the 7-zip archive just like in a directory.

MiniWeb can also be used in audio/video streaming applications, or more specific, VOD (video-on-demand) service. Currently a VOD client/server is being developed on MiniWeb.

Source Code

The source code of MiniWeb is in SourceForge repository. You can view the the source code instantly here.

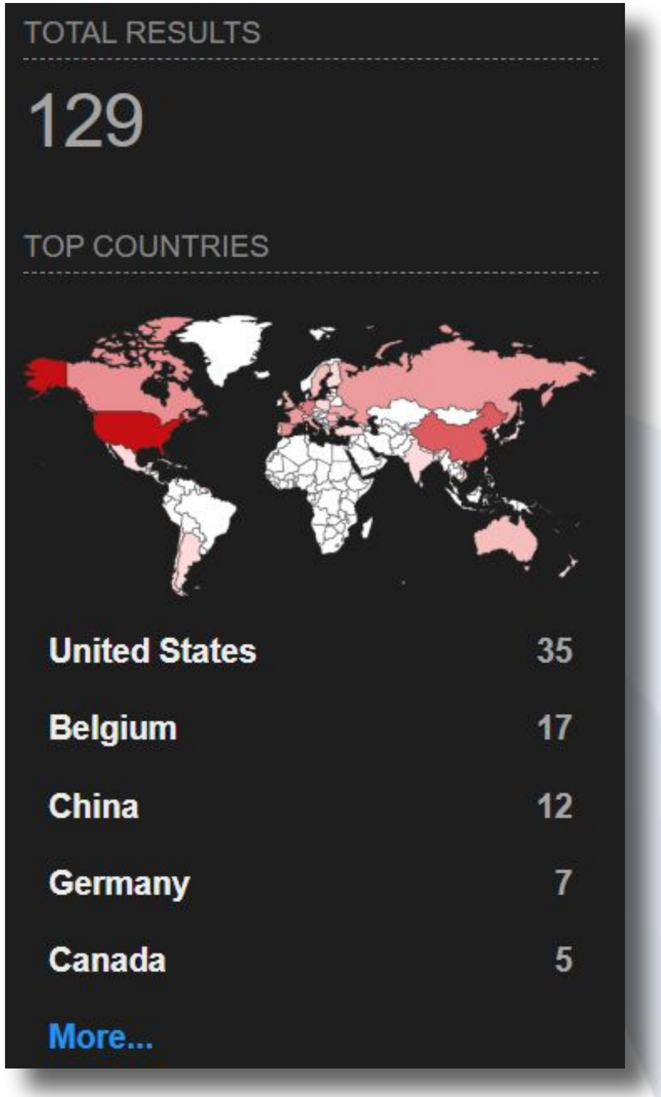
Links

MediaCoder - the universal media transcoder which uses MiniWeb as the built-in HTTP daemon.

MiniWeb (C)2005-2012 All rights reserved by Stanley Huang

miniweb 簡介

- http server
- C 語言搭建的
- 有開源原始碼
- 有部分網站使用



TOTAL RESULTS

129

TOP COUNTRIES

United States	35
Belgium	17
China	12
Germany	7
Canada	5

More...

miniweb 歷年漏洞

- CVE-2020-29596
  - allow remote attackers to cause a denial of service (daemon crash) via a long name for the first parameter in a POST request.
- CVE-2008-0338
  - Directory traversal vulnerability allows remote attackers to read arbitrary files and list arbitrary directories
- CVE-2008-0337
  - Heap-based buffer overflow allows remote attackers to execute arbitrary code via a long URI.

miniweb 歷年漏洞

- CVE-2007-3159
  - allow remote attackers to cause a denial of service (application crash) via a negative value in the Content-Length HTTP header.
- CVE-2002-0298
  - allow remote attackers to cause a denial of service (crash) via certain HTTP GET requests containing (1) a %2e%2e (encoded dot-dot), (2) several /../ (dot dot) sequences, (3) a missing URI, or (4) several ../ in a URI that does not begin with a / (slash) character.
- CVE-2002-0297
  - Buffer overflow allows remote attackers to cause a denial of service (crash) and possibly execute arbitrary code via a long URL in an HTTP request.

Credit

- 幫忙修漏洞

  → 發 pr + 聯繫維護者/ 使用者

- shodan 發現使用 miniweb 的網站

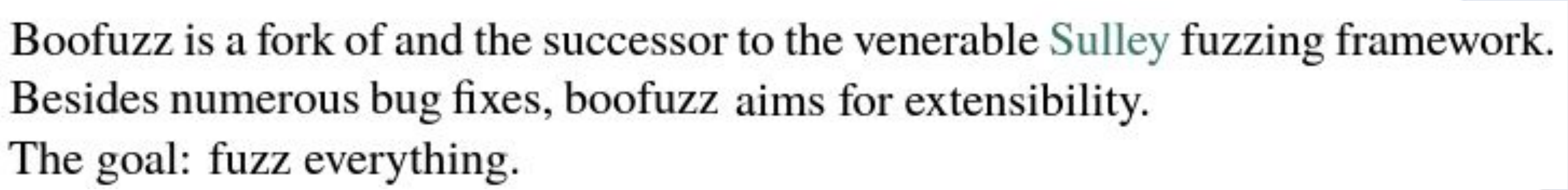
背景知識

Boofuzz 簡介

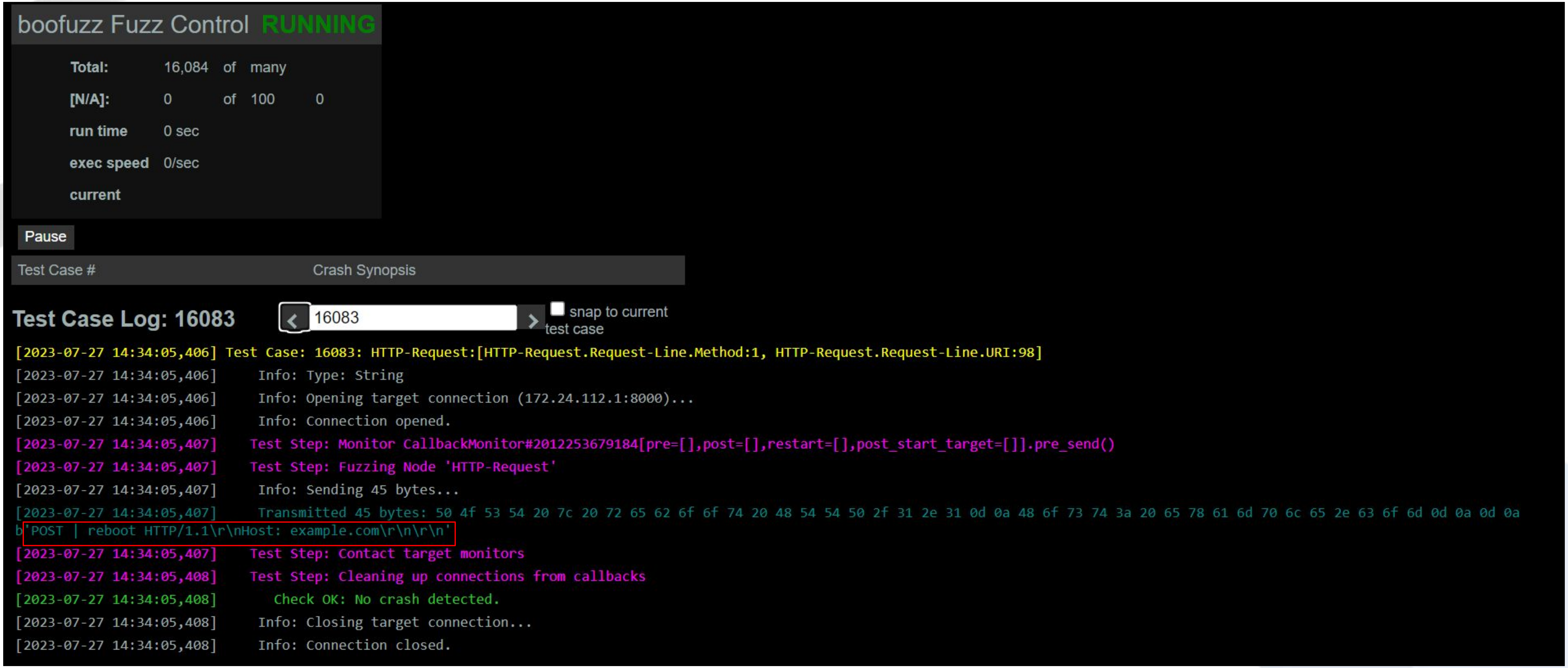
- 架構繼承自 Sulley (已停止維護)
- 改善 Sulley 功能並除錯，並增加支持任何通訊媒介、內建支持 serial fuzzing, ethernet- and IP-layer, UDP broadcast 等

Boofuzz/ Sulley

- generation-based (with specified format)
  - pros: more efficient
  - cons: smaller input space
- mutate each field one at a time
- fuzz everything!



Boofuzz is a fork of and the successor to the venerable Sulley fuzzing framework.
Besides numerous bug fixes, boofuzz aims for extensibility.
The goal: fuzz everything.



boofuzz Fuzz Control RUNNING

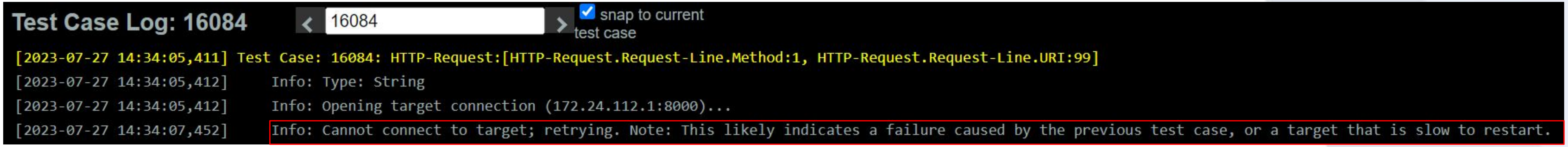
```
Total:       16,084   of   many
[N/A]:       0        of   100     0
run time     0 sec
exec speed   0/sec
current
```

Pause

Test Case #	Crash Synopsis

Test Case Log: 16083 ‹ 16083 › □ snap to current test case

```
[2023-07-27 14:34:05,406] Test Case: 16083: HTTP-Request:[HTTP-Request.Request-Line.Method:1, HTTP-Request.Request-Line.URI:98]
[2023-07-27 14:34:05,406]     Info: Type: String
[2023-07-27 14:34:05,406]     Info: Opening target connection (172.24.112.1:8000)...
[2023-07-27 14:34:05,406]     Info: Connection opened.
[2023-07-27 14:34:05,407]    Test Step: Monitor CallbackMonitor#2012253679184[pre=[],post=[],restart=[],post_start_target=[]].pre_send()
[2023-07-27 14:34:05,407]    Test Step: Fuzzing Node 'HTTP-Request'
[2023-07-27 14:34:05,407]     Info: Sending 45 bytes...
[2023-07-27 14:34:05,407]     Transmitted 45 bytes: 50 4f 53 54 20 7c 20 72 65 62 6f 6f 74 20 48 54 54 50 2f 31 2e 31 0d 0a 48 6f 73 74 3a 20 65 78 61 6d 70 6c 65 2e 63 6f 6d 0d 0a 0d 0a
b'POST | reboot HTTP/1.1\r\nHost: example.com\r\n\r\n'
[2023-07-27 14:34:05,407]    Test Step: Contact target monitors
[2023-07-27 14:34:05,408]    Test Step: Cleaning up connections from callbacks
[2023-07-27 14:34:05,408]       Check OK: No crash detected.
[2023-07-27 14:34:05,408]     Info: Closing target connection...
[2023-07-27 14:34:05,408]     Info: Connection closed.
```



Test Case Log: 16084 ‹ 16084 › ☑ snap to current test case

```
[2023-07-27 14:34:05,411] Test Case: 16084: HTTP-Request:[HTTP-Request.Request-Line.Method:1, HTTP-Request.Request-Line.URI:99]
[2023-07-27 14:34:05,412]     Info: Type: String
[2023-07-27 14:34:05,412]     Info: Opening target connection (172.24.112.1:8000)...
[2023-07-27 14:34:07,452]     Info: Cannot connect to target; retrying. Note: This likely indicates a failure caused by the previous test case, or a target that is slow to restart.
```

漏洞:CVE-2020-29596

漏洞說明

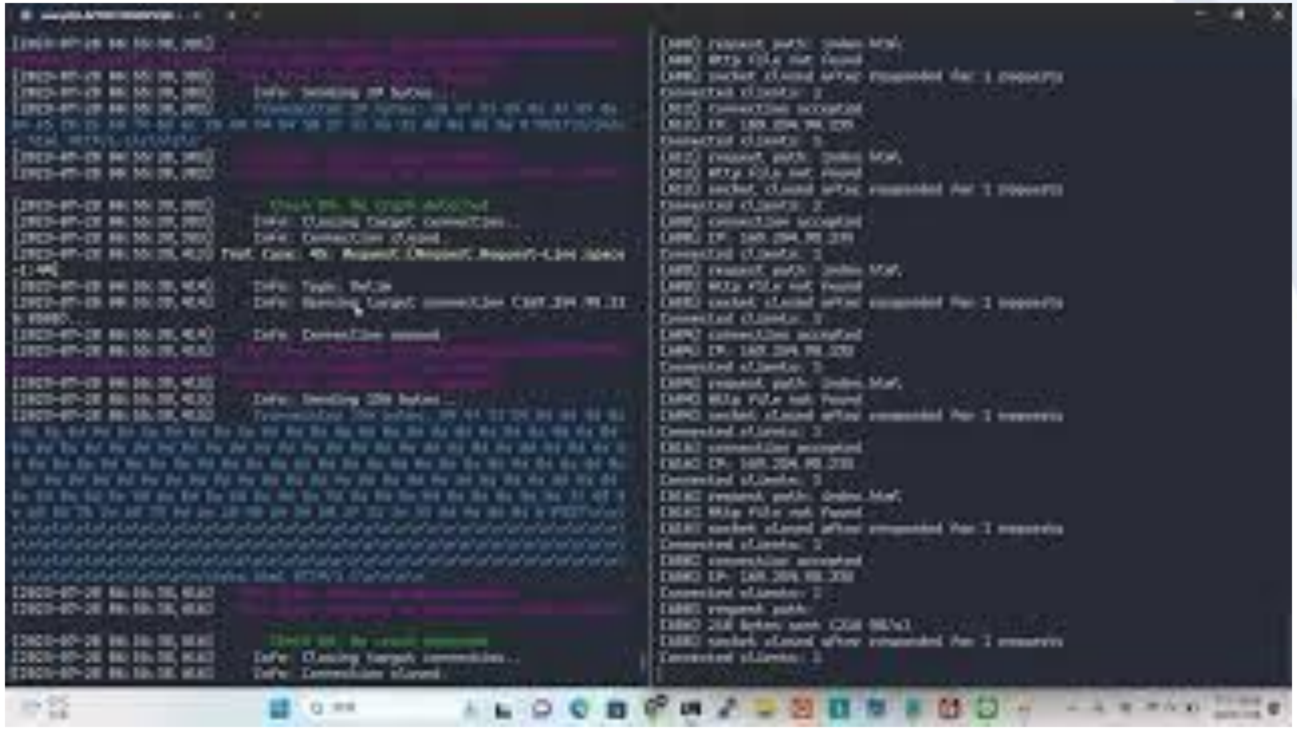
MiniWeb HTTP server 0.8.19 allows remote attackers to cause a denial of service (daemon crash) via a long name for the first parameter in a POST request.

漏洞重現

嘗試透過 fuzzing 重新找到這個漏洞

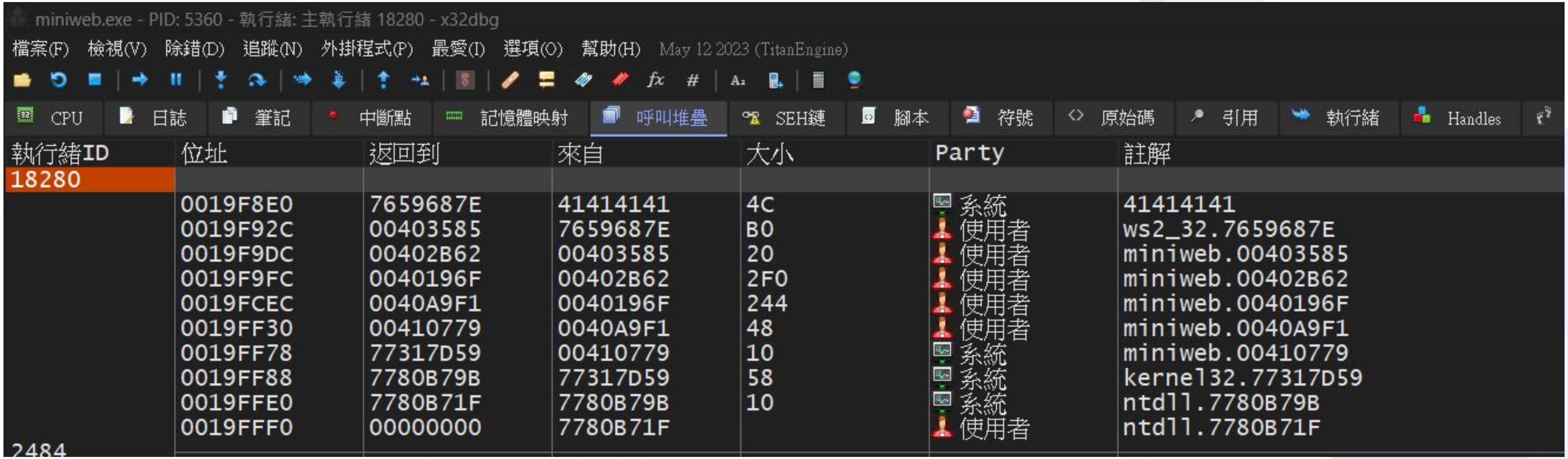
fuzzing 過程

使用 github 的範例腳本, 針對 POST request 去 fuzz



漏洞分析

使用 x32dbg 查看 crash 時的 call stack

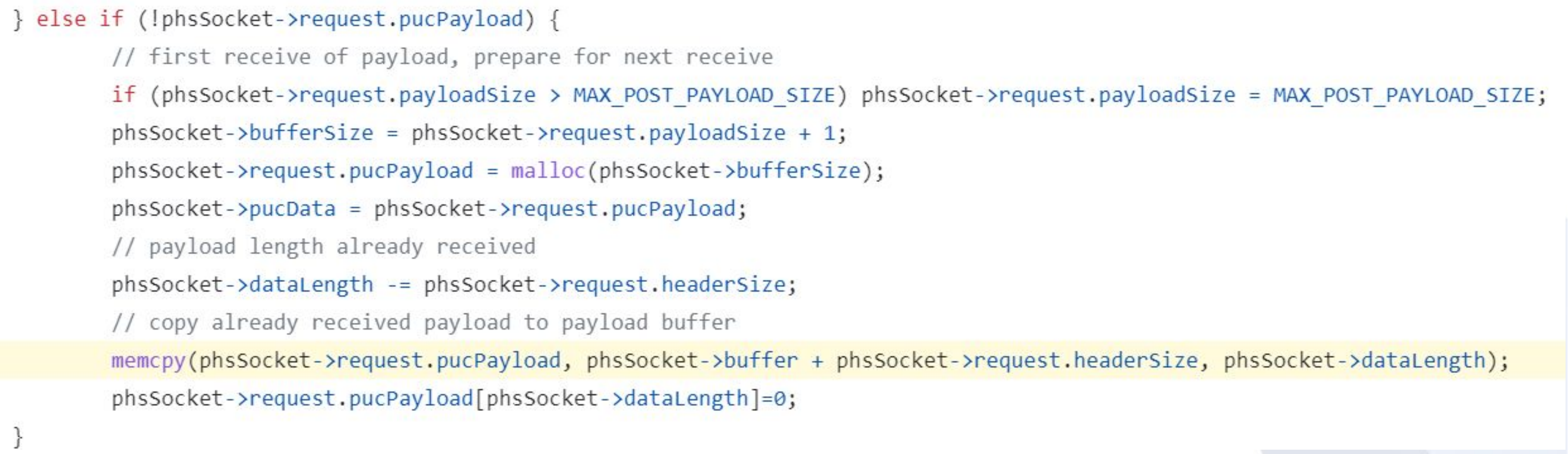

miniweb.exe - PID: 5360 - 執行緒: 主執行緒 18280 - x32dbg
檔案(F) 檢視(V) 除錯(D) 追蹤(N) 外掛程式(P) 最愛(I) 選項(O) 幫助(H) May 12 2023 (TitanEngine)
CPU
日誌
筆記
中斷點
記憶體映射
呼叫堆疊
SEH鏈
腳本
符號
原始碼
引用
執行緒
Handles

執行緒ID	位址	返回到	來自	大小	Party	註解
18280						
	0019F8E0	7659687E	41414141	4C	系統	41414141
	0019F92C	00403585	7659687E	B0	使用者	ws2_32.7659687E
	0019F9DC	00402B62	00403585	20	使用者	miniweb.00403585
	0019F9FC	0040196F	00402B62	2F0	使用者	miniweb.00402B62
	0019FCEC	0040A9F1	0040196F	244	使用者	miniweb.0040196F
	0019FF30	00410779	0040A9F1	48	使用者	miniweb.0040A9F1
	0019FF78	77317D59	00410779	10	系統	miniweb.00410779
	0019FF88	7780B79B	77317D59	58	系統	kernel32.77317D59
	0019FFE0	7780B71F	7780B79B	10	系統	ntdll.7780B79B
	0019FFF0	00000000	7780B71F		使用者	ntdll.7780B71F
2484						

漏洞分析

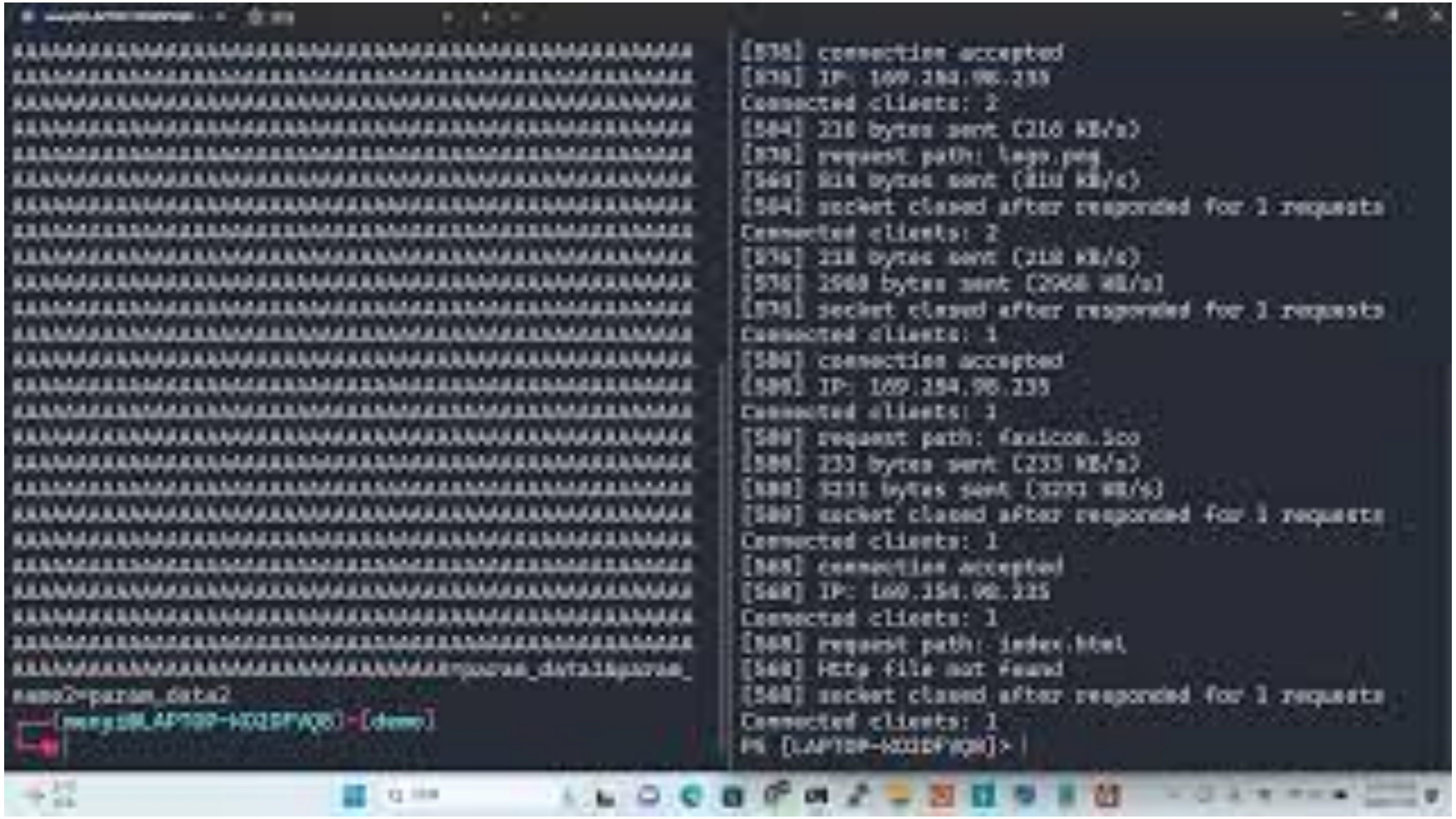
- 查看 call stack 發現呼叫路徑：main() -> _mwHttpLoop() -> _mwProcessReadSocket() -> _mwStartSendFile()
- 問題：_mwProcessReadSocket() 中未對 phsSocket->dataLength 做檢查，導致 heap overflow

漏洞分析



```
} else if (!phsSocket->request.pucPayload) {
        // first receive of payload, prepare for next receive
        if (phsSocket->request.payloadSize > MAX_POST_PAYLOAD_SIZE) phsSocket->request.payloadSize = MAX_POST_PAYLOAD_SIZE;
        phsSocket->bufferSize = phsSocket->request.payloadSize + 1;
        phsSocket->request.pucPayload = malloc(phsSocket->bufferSize);
        phsSocket->pucData = phsSocket->request.pucPayload;
        // payload length already received
        phsSocket->dataLength -= phsSocket->request.headerSize;
        // copy already received payload to payload buffer
        memcpy(phsSocket->request.pucPayload, phsSocket->buffer + phsSocket->request.headerSize, phsSocket->dataLength);
        phsSocket->request.pucPayload[phsSocket->dataLength]=0;
}
```

POC demo

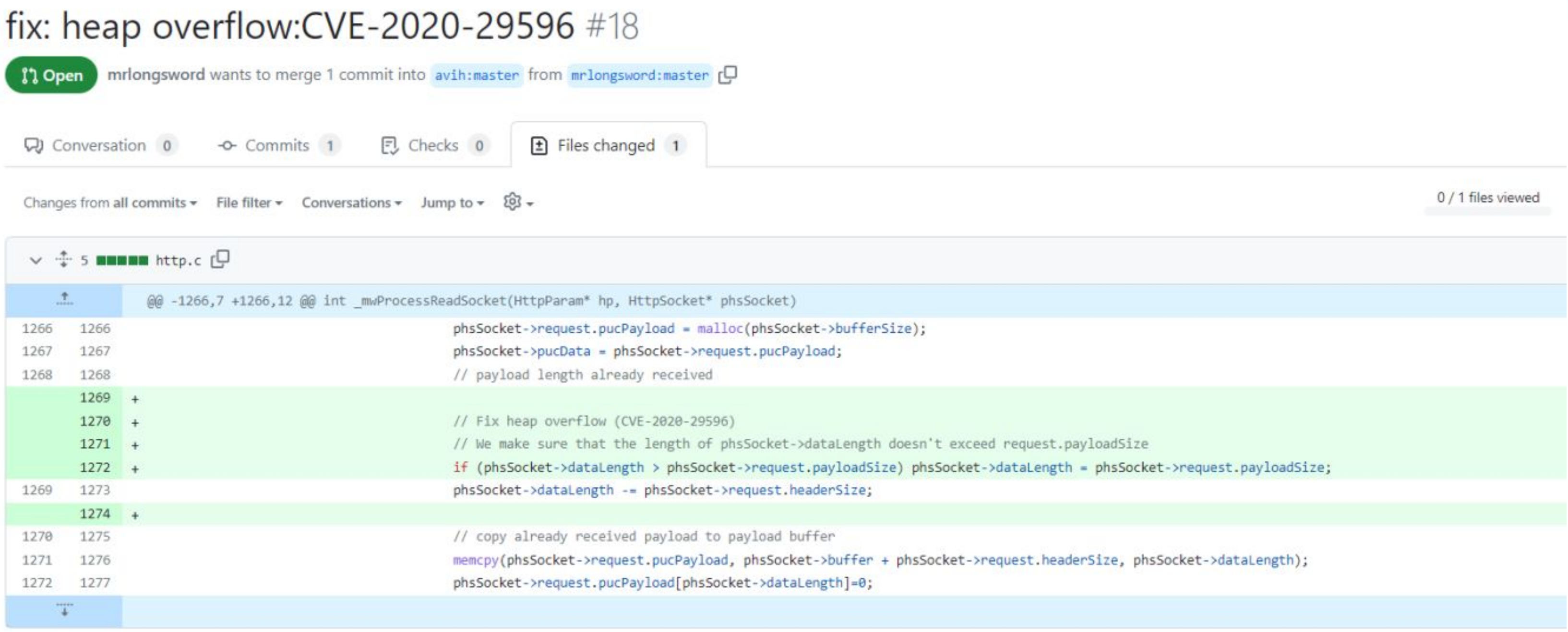


```
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA=param_data1&param_
name2=param_data2
┌──(mengyi@LAPTOP-KO2DFVQ8)-[demo]
└─$
```

```
[576] connection accepted
[576] IP: 169.254.98.235
Connected clients: 2
[564] 216 bytes sent (216 KB/s)
[576] request path: logo.png
[564] 814 bytes sent (814 KB/s)
[564] socket closed after responded for 1 requests
Connected clients: 2
[576] 218 bytes sent (218 KB/s)
[576] 2968 bytes sent (2968 KB/s)
[576] socket closed after responded for 1 requests
Connected clients: 1
[580] connection accepted
[580] IP: 169.254.98.235
Connected clients: 1
[580] request path: favicon.ico
[580] 233 bytes sent (233 KB/s)
[580] 3231 bytes sent (3231 KB/s)
[580] socket closed after responded for 1 requests
Connected clients: 1
[568] connection accepted
[568] IP: 169.254.98.235
Connected clients: 1
[568] request path: index.html
[568] Http file not found
[568] socket closed after responded for 1 requests
Connected clients: 1
PS [LAPTOP-KO2DFVQ8]> 
```

漏洞修補

將漏洞修補並發 Pull Request

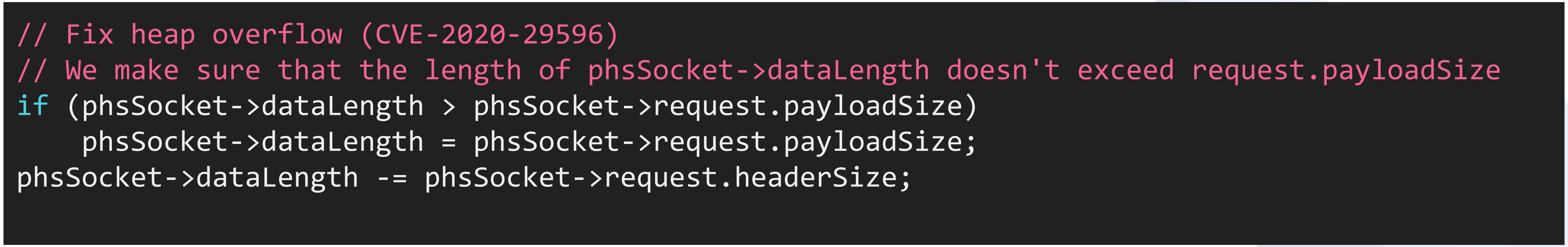


fix: heap overflow:CVE-2020-29596 #18

mrlongsword wants to merge 1 commit into avih:master from mrlongsword:master

```
@@ -1266,7 +1266,12 @@ int _mwProcessReadSocket(HttpParam* hp, HttpSocket* phsSocket)
                                phsSocket->request.pucPayload = malloc(phsSocket->bufferSize);
                                phsSocket->pucData = phsSocket->request.pucPayload;
                                // payload length already received
+
+                               // Fix heap overflow (CVE-2020-29596)
+                               // We make sure that the length of phsSocket->dataLength doesn't exceed request.payloadSize
+                               if (phsSocket->dataLength > phsSocket->request.payloadSize) phsSocket->dataLength = phsSocket->request.payloadSize;
                                phsSocket->dataLength -= phsSocket->request.headerSize;
+
                                // copy already received payload to payload buffer
                                memcpy(phsSocket->request.pucPayload, phsSocket->buffer + phsSocket->request.headerSize, phsSocket->dataLength);
                                phsSocket->request.pucPayload[phsSocket->dataLength]=0;
```

漏洞修補



```
// Fix heap overflow (CVE-2020-29596)
// We make sure that the length of phsSocket->dataLength doesn't exceed request.payloadSize
if (phsSocket->dataLength > phsSocket->request.payloadSize)
    phsSocket->dataLength = phsSocket->request.payloadSize;
phsSocket->dataLength -= phsSocket->request.headerSize;
```

結論

結論

- 透過 boofuzz 實作 fuzzing 的過程
- 透過 fuzzing 發現 N-day
- 分析漏洞並修補

收穫與心得

- 熟悉 fuzzing 的工具
- 熟悉漏洞分析、debug 的技巧
- 修補漏洞、回饋社群

End

Any Questions?