Find All Hidden Commands in The Firmware

Project Website: https://gitlab.utwente.nl/s2694328/CS4150-SystemSecurity/-/tree/main/labOther_Hidden_Commands_in_Firmware

Tools Used

- Python3
- Ghidra with Xtensa
- Mosquitto client and Mosquitto
- esptool
- Wireshark
- WiHotspot

WiFi Hotspot Set Up

Since my computer has no port for ethernet cable, I found another solution to deal with this problem which was using *wihotspot*. The *wihotspot* allows Linux to connect the WiFi and share it using the hotspot at the same time. The installation procedure has been shown below.

1. To clone the latest package,

```
git clone https://github.com/lakinduakash/linux-wifi-hotspot

cd linux-wifi-hotspot
```

2. To install the dependencies,

```
sudo apt install -y libgtk-3-dev build-essential gcc g++ pkg-config
make hostapd
```

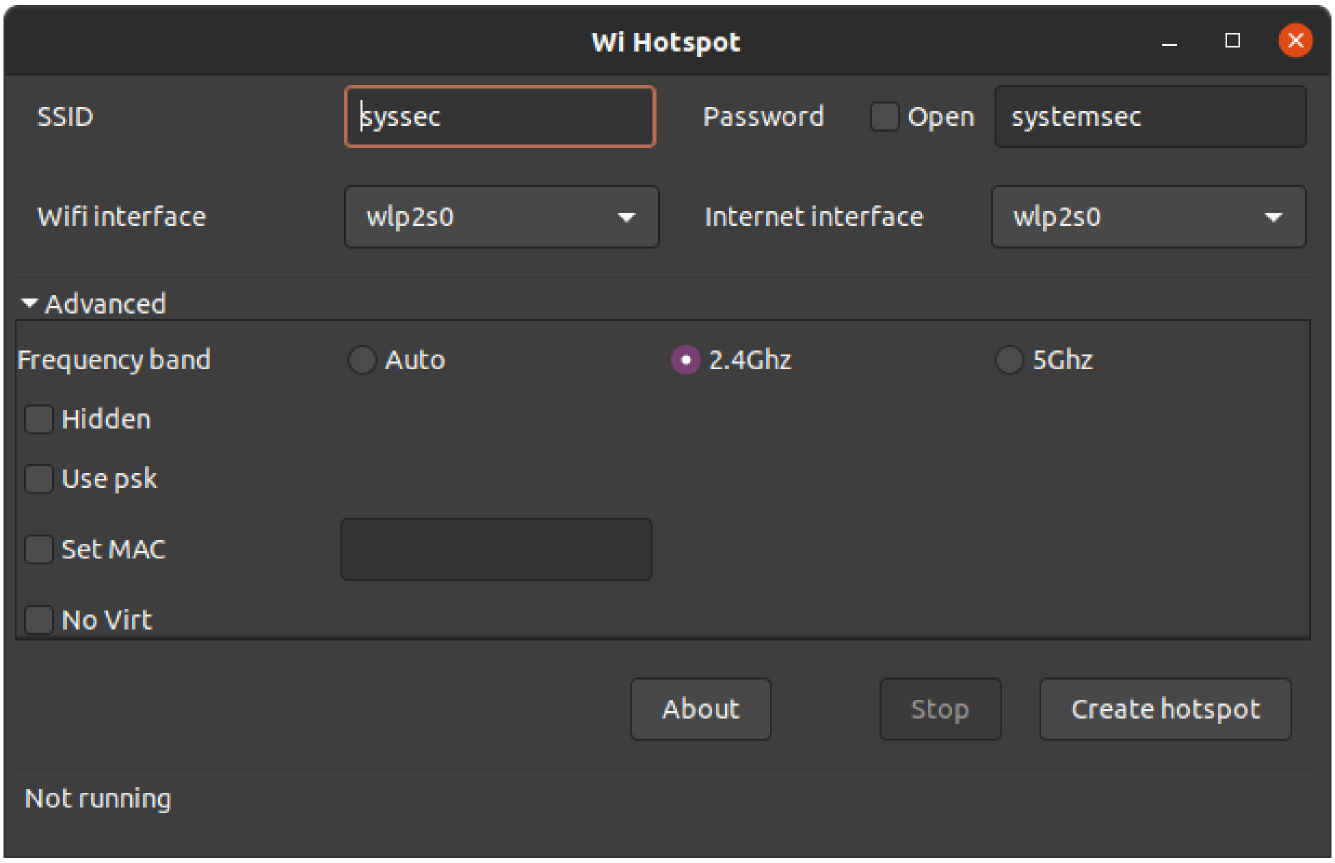
3. To build and install it,

```
make

sudo make install
```

4. To run it,

```
wihotspot
```

After using the commands above, the UI will appear like the figure below.



SSID: syssec
Password: systemsec
Wifi interface: wlp2s0
Internet interface: wlp2s0
Frequency band: Auto [✓ 2.4Ghz] 5Ghz
Not running

For convenience, the SSID and password are set to "syssec" and "systemsec" respectively. The wifi interface and internet interface are both set to `wlp2s0` and it will generate a virtual `ap0` network.

In so far, the hotspot has been built up and can be searched by other devices.

To Redirect & Monitor The Flow

In order to send the mqtt message to the esp board and observe the response, we used local mosquitto to emulate the server and use the serial port to monitor the response.

1. Set up mosquitto server:

```
mosquitto -p 8888
```

2. Redirect mqtt receiving port 1883 - it can be known by using Wireshark - to 8888:

```
sudo iptables -t nat -A PREROUTING -i ap0 -p tcp --dport 1883 -j
REDIRECT --to-port 8888
```

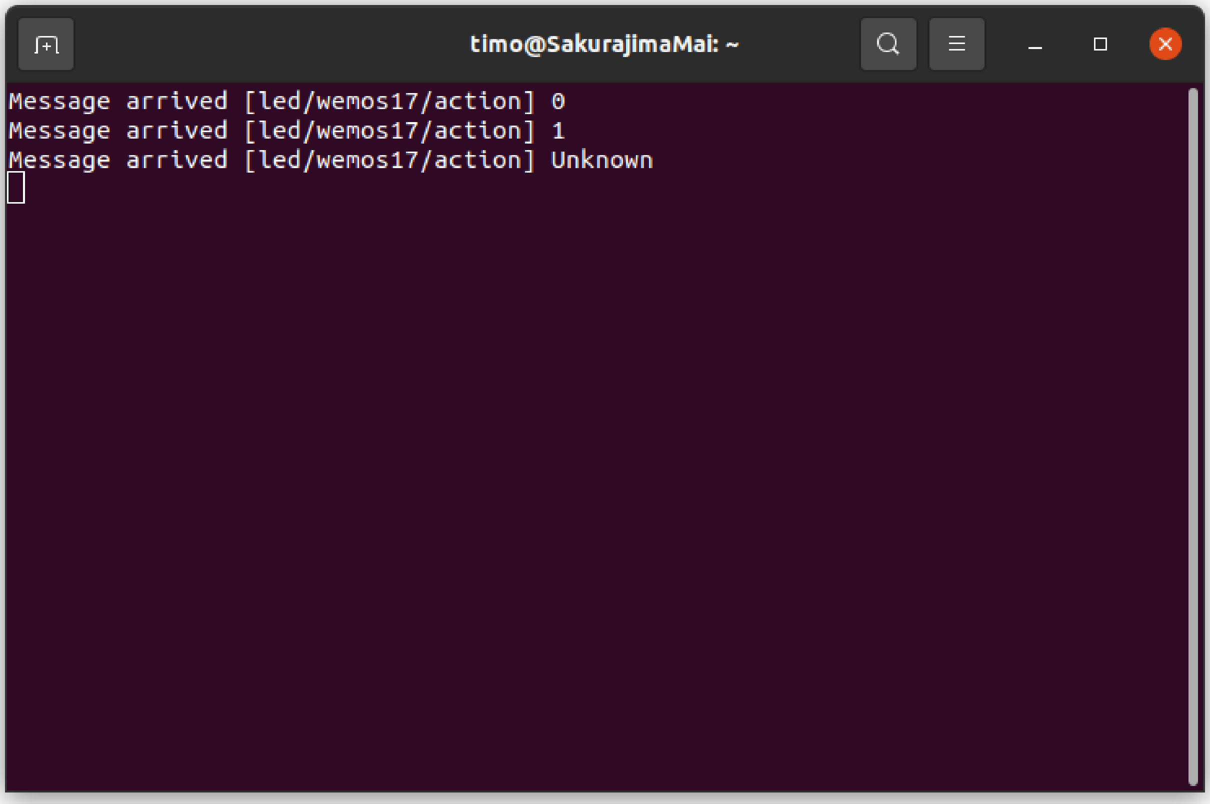
3. Mointer the serial port:

```
# without logging
screen /dev/ttyUSB0 115200

# with logging
screen -L -Logfile ./espOutput.log /dev/ttyUSB0 115200
```

4. Publishing Test:

```
mosquitto_pub -h localhost -p 8888 -t led/wemos17/action -m 0

mosquitto_pub -h localhost -p 8888 -t led/wemos17/action -m 1

mosquitto_pub -h localhost -p 8888 -t led/wemos17/action -m a
```

The serial port prints:



```
Message arrived [led/wemos17/action] 0
Message arrived [led/wemos17/action] 1
Message arrived [led/wemos17/action] Unknown
```

Ghidra for Finding More

According to the previous testing result, we noticed that if the command does not exist, the response will be *Unknown*. We can use Ghidra to see where the program outputs *Message arrived* with *Unkown* and what is the conditional statement.

Read Flash

To extract the raw program from the flash, esptool.py should be used.

```
esptool.py --baud 115200 read_flash 0x0 0x400000 firmware.bin
```

To segment the raw data:

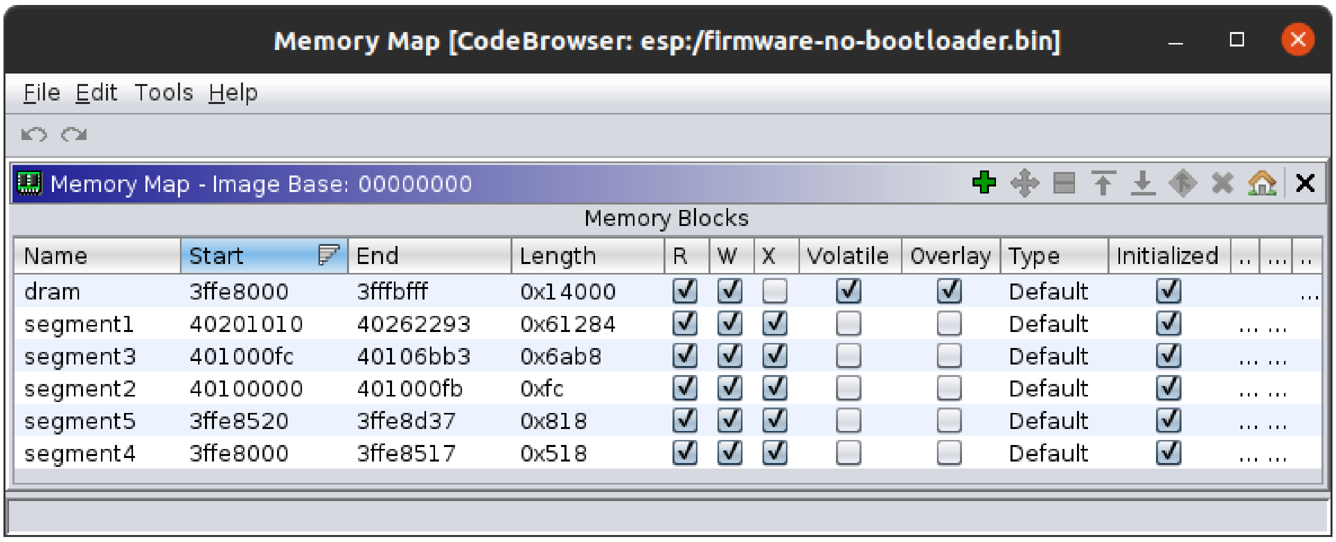
```
dd if=firmware.bin of=firmware-no-bootloader.bin bs=4k skip=1
```

To observe the information of segments:

```
esptool.py image_info firmware-no-bootloader.bin

# output
Segment 1: len 0x62184 load 0x40201010 file_offs 0x00000008 [IROM]

Segment 2: len 0x000fc load 0x40100000 file_offs 0x00062194 [IRAM]

Segment 3: len 0x06ab8 load 0x401000fc file_offs 0x00062298 [IRAM]

Segment 4: len 0x00518 load 0x3ffe8000 file_offs 0x00068d58 [DRAM]

Segment 5: len 0x00818 load 0x3ffe8520 file_offs 0x00069278 [DRAM]
```

Ghidra Auto Analysis

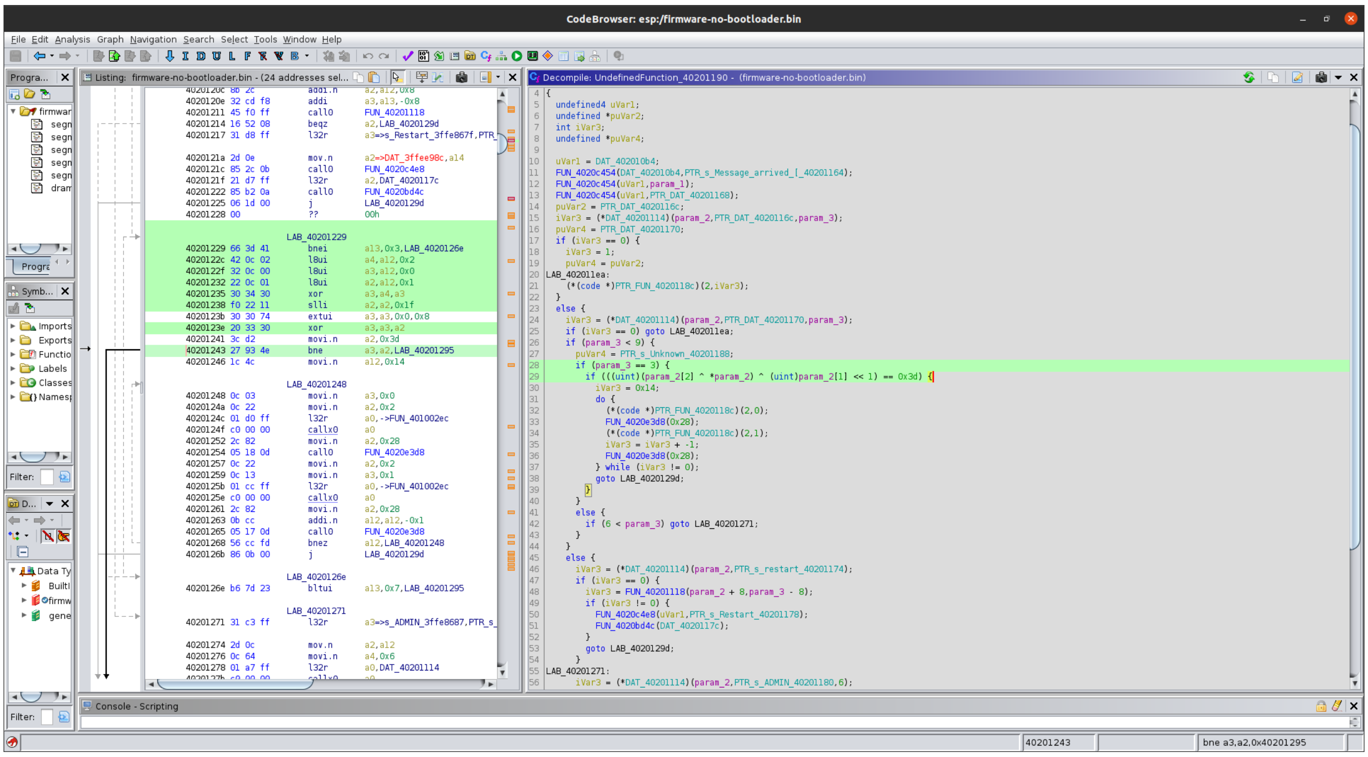
After running Ghidra, the raw program can be loaded into it by using the setting below:


Memory Map [CodeBrowser: esp:/firmware-no-bootloader.bin]
File Edit Tools Help
Memory Map - Image Base: 00000000

Memory Blocks

Name	Start	End	Length	R	W	X	Volatile	Overlay	Type	Initialized	..	...	..
dram	3ffe8000	3fffbfff	0x14000	✓	✓		✓	✓	Default	✓			...
segment1	40201010	40262293	0x61284	✓	✓	✓			Default	✓	...	...	
segment3	401000fc	40106bb3	0x6ab8	✓	✓	✓			Default	✓	...	...	
segment2	40100000	401000fb	0xfc	✓	✓	✓			Default	✓	...	...	
segment5	3ffe8520	3ffe8d37	0x818	✓	✓	✓			Default	✓	...	...	
segment4	3ffe8000	3ffe8517	0x518	✓	✓	✓			Default	✓	...	...	

After auto analysis, using search to locate the `string==Unknown`, we found that both *Message arrived* and *Unknown* are used in `UndefinedFunction_40201190`. We suspected that this the function to process the mqtt message. It has three input arguments - `undefined4 param_1`,`byte *param_2`,`uint param_3`. After reading the body of this function, we conjectured that `param_2` is the pointer to a byte array where ASCII message is stored and `param_3` is the length of the message. When `iVar3` is greater than 0, it stays in the while loop where the led flashes. `FUN_4020e3d8` is the function for time delay. `*(code *)PTR_FUN_4020118c)(2,0)` and `*(code *)PTR_FUN_4020118c)(2,1)` control the off/on of LED.


CodeBrowser: esp:/firmware-no-bootloader.bin

Listing: firmware-no-bootloader.bin - (24 addresses sel...

```
4020120c 8b 2c        addi.n   a2,a12,0x8
4020120e 32 cd f8     addi     a3,a13,-0x8
40201211 45 f0 ff     call0    FUN_40201118
40201214 16 52 08     beqz     a2,LAB_4020129d
40201217 31 d8 ff     l32r     a3=>s_Restart_3ffe867f,PTR_

4020121a 2d 0e        mov.n    a2=>DAT_3ffee98c,a14
4020121c 85 2c 0b     call0    FUN_4020c4e8
4020121f 21 d7 ff     l32r     a2,DAT_4020117c
40201222 85 b2 0a     call0    FUN_4020bd4c
40201225 06 1d 00     j        LAB_4020129d
40201228 00           ??       00h

                  LAB_40201229
40201229 66 3d 41     bnei     a13,0x3,LAB_4020126e
4020122c 42 0c 02     l8ui     a4,a12,0x2
4020122f 32 0c 00     l8ui     a3,a12,0x0
40201232 22 0c 01     l8ui     a2,a12,0x1
40201235 30 34 30     xor      a3,a4,a3
40201238 f0 22 11     slli     a2,a2,0x1f
4020123b 30 30 74     extui    a3,a3,0x0,0x8
4020123e 20 33 30     xor      a3,a3,a2
40201241 3c d2        movi.n   a2,0x3d
40201243 27 93 4e     bne      a3,a2,LAB_40201295
40201246 1c 4c        movi.n   a12,0x14

                  LAB_40201248
40201248 0c 03        movi.n   a3,0x0
4020124a 0c 22        movi.n   a2,0x2
4020124c 01 d0 ff     l32r     a0,->FUN_401002ec
4020124f c0 00 00     callx0   a0
40201252 2c 82        movi.n   a2,0x28
40201254 05 18 0d     call0    FUN_4020e3d8
40201257 0c 22        movi.n   a2,0x2
40201259 0c 13        movi.n   a3,0x1
4020125b 01 cc ff     l32r     a0,->FUN_401002ec
4020125e c0 00 00     callx0   a0
40201261 2c 82        movi.n   a2,0x28
40201263 0b cc        addi.n   a12,a12,-0x1
40201265 05 17 0d     call0    FUN_4020e3d8
40201268 56 cc fd     bnez     a12,LAB_40201248
4020126b 86 0b 00     j        LAB_4020129d

                  LAB_4020126e
4020126e b6 7d 23     bltui    a13,0x7,LAB_40201295

                  LAB_40201271
40201271 31 c3 ff     l32r     a3=>s_ADMIN_3ffe8687,PTR_s_

40201274 2d 0c        mov.n    a2,a12
40201276 0c 64        movi.n   a4,0x6
40201278 01 a7 ff     l32r     a0,DAT_40201114
4020127b c0 00 00     callx0   a0
```

Decompile: UndefinedFunction_40201190 - (firmware-no-bootloader.bin)

```c
{
  undefined4 uVar1;
  undefined *puVar2;
  int iVar3;
  undefined *puVar4;

  uVar1 = DAT_402010b4;
  FUN_4020c454(DAT_402010b4,PTR_s_Message_arrived_[_40201164);
  FUN_4020c454(uVar1,param_1);
  FUN_4020c454(uVar1,PTR_DAT_40201168);
  puVar2 = PTR_DAT_4020116c;
  iVar3 = (*DAT_40201114)(param_2,PTR_DAT_4020116c,param_3);
  puVar4 = PTR_DAT_40201170;
  if (iVar3 == 0) {
    iVar3 = 1;
    puVar4 = puVar2;
LAB_402011ea:
    (*(code *)PTR_FUN_4020118c)(2,iVar3);
  }
  else {
    iVar3 = (*DAT_40201114)(param_2,PTR_DAT_40201170,param_3);
    if (iVar3 == 0) goto LAB_402011ea;
    if (param_3 < 9) {
      puVar4 = PTR_s_Unknown_40201188;
      if (param_3 == 3) {
        if (((uint)(param_2[2] ^ *param_2) ^ (uint)param_2[1] << 1) == 0x3d) {
          iVar3 = 0x14;
          do {
            (*(code *)PTR_FUN_4020118c)(2,0);
            FUN_4020e3d8(0x28);
            (*(code *)PTR_FUN_4020118c)(2,1);
            iVar3 = iVar3 + -1;
            FUN_4020e3d8(0x28);
          } while (iVar3 != 0);
          goto LAB_4020129d;
        }
      }
      else {
        if (6 < param_3) goto LAB_40201271;
      }
    }
    else {
      iVar3 = (*DAT_40201114)(param_2,PTR_s_restart_40201174);
      if (iVar3 == 0) {
        iVar3 = FUN_40201118(param_2 + 8,param_3 - 8);
        if (iVar3 != 0) {
          FUN_4020c4e8(uVar1,PTR_s_Restart_40201178);
          FUN_4020bd4c(DAT_4020117c);
        }
        goto LAB_4020129d;
      }
LAB_40201271:
      iVar3 = (*DAT_40201114)(param_2,PTR_s_ADMIN_40201180,6);
```

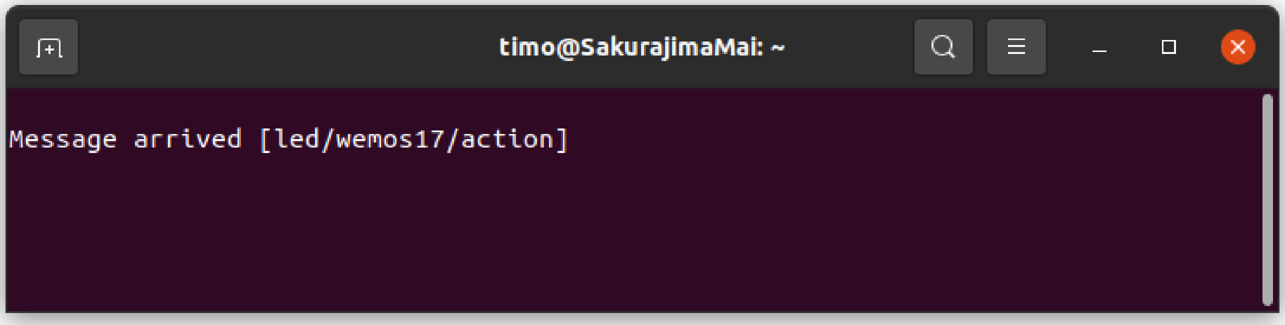
Console - Scripting

40201243 bne a3,a2,0x40201295

It always respond `PTR_s_Unknown_40201188` besides when `param_3==3` and `(uint)(param_2[2] ^ *param_2) ^ (uint)param_2[1] << 1) == 0x3d`. In this situation, we thought that the length of the message should equal to 3 and the three ASCII bytes should follow the above rule. In order to verify our speculation, we used "a28" to do the test because `ord('8')^ord('a')^(ord('2')<<1)==0x3d`.

```
mosquitto_pub -h localhost -p 8888 -t led/wemos17/action -m a28
```

After the command is transmitted, the led is flashing while the serial port does not print *Unkown*. Thus, the speculation is valid.


timo@SakurajimaMai: ~

```
Message arrived [led/wemos17/action]
```

Search Commands & Validation

We made a simple python script that could solve the commands within few seconds.

To run the script, it will generate a file named `commands.txt` where all the commands are included.

```
python3 search.py
```

In addition, in order to validate the commands, we made a script to send commands one by one to the board and recorded responses. Since the speed of processing command is quite slow and mqtt is not quite reliable for the transmission, it needs an extra delay between two consecutive commands, otherwise, some messages might be lost.

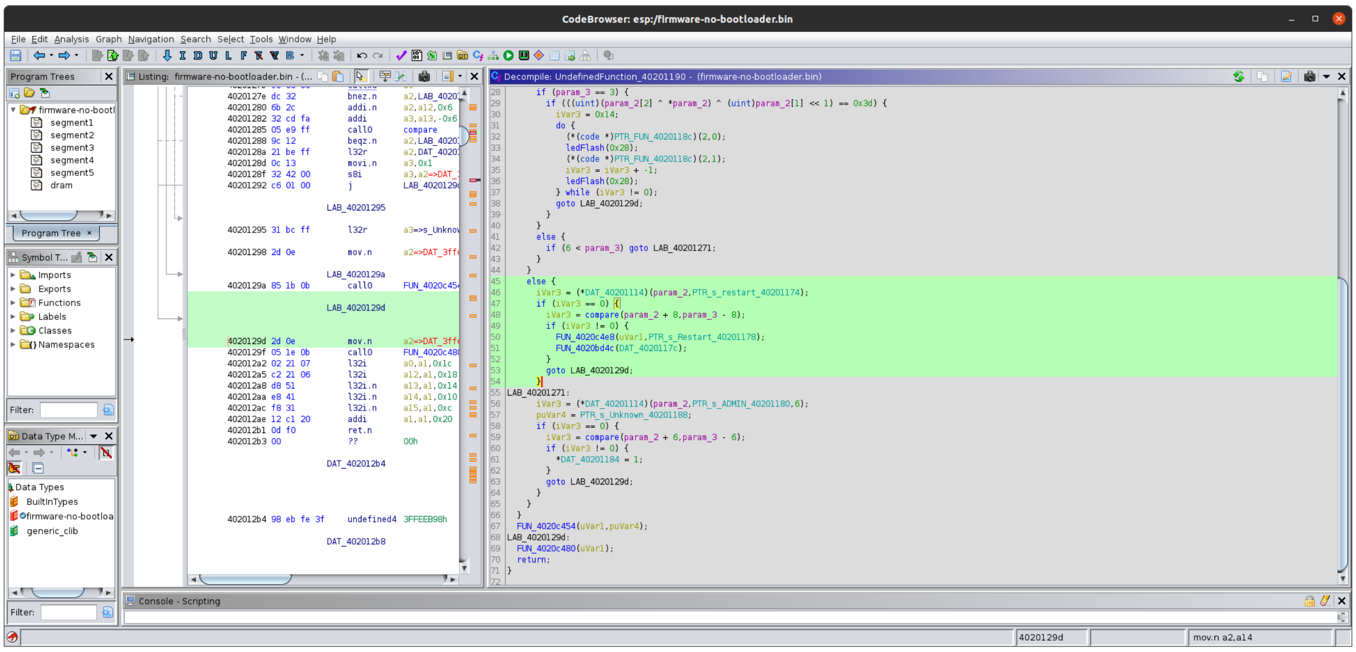
To run the script:

```
# to record reaction
screen -L -Logfile ./espOutput.log /dev/ttyUSB0 115200

# it will take you about an hour to validate all commands.
python3 validation.py
```

The file named `commands` includes all the commands for making the LED flash.

Restart Command

Besides the commands for flashing, we also found a command for restarting the board. In the figure below, it shows that if the first 8 bytes equal to `restart` , it will compare the remaining bytes with something.

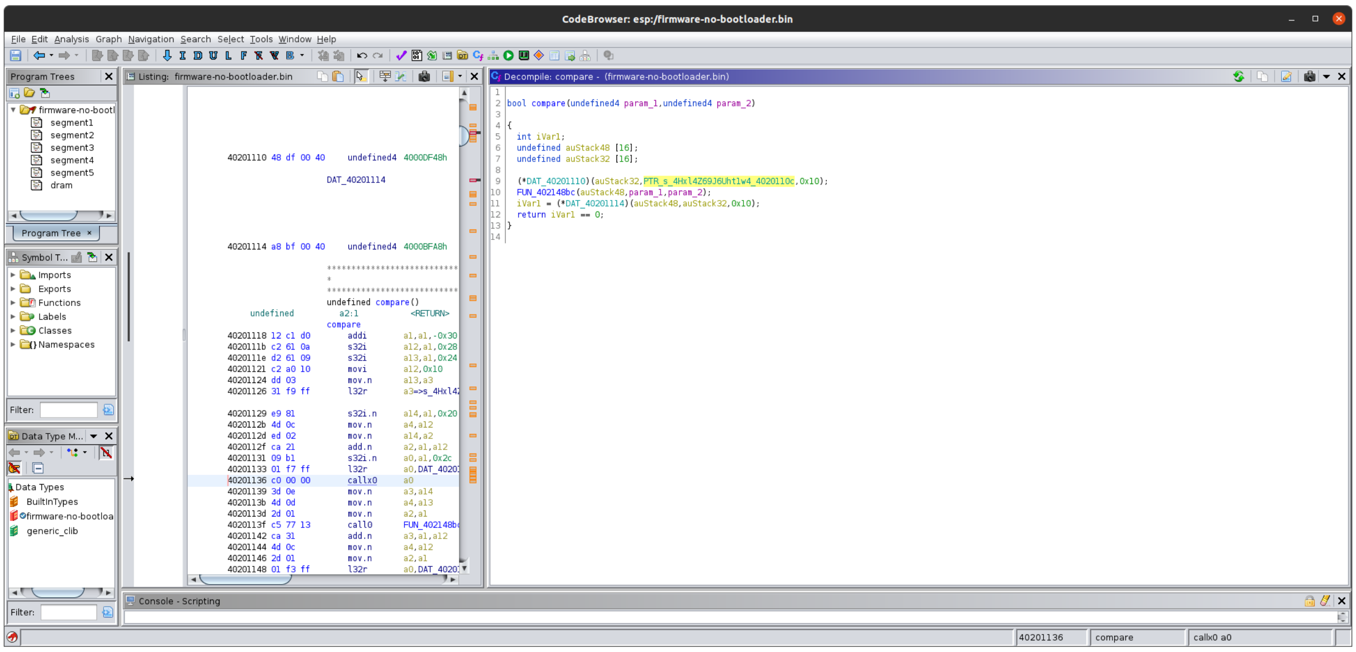

CodeBrowser: esp:/firmware-no-bootloader.bin
Listing: firmware-no-bootloader.bin - (...
Decompile: UndefinedFunction_40201190 - (firmware-no-bootloader.bin)
Console - Scripting

```
4020127e dc 32        bnez.n   a2,LAB_4020
40201280 6b 2c        addi.n   a2,a12,0x6
40201282 32 cd fa     addi     a3,a13,-0x6
40201285 05 e9 ff     call0    compare
40201288 9c 12        beqz.n   a2,LAB_4020
4020128a 21 be ff     l32r     a2,DAT_4020
4020128d 0c 13        movi.n   a3,0x1
4020128f 32 42 00     s8i      a3,a2=>DAT_
40201292 c6 01 00     j        LAB_4020129

                  LAB_40201295
40201295 31 bc ff     l32r     a3=>s_Unknov

40201298 2d 0e        mov.n    a2=>DAT_3ffe

                  LAB_4020129a
4020129a 85 1b 0b     call0    FUN_4020c45

                  LAB_4020129d

4020129d 2d 0e        mov.n    a2=>DAT_3ffe
4020129f 05 1e 0b     call0    FUN_4020c48
402012a2 02 21 07     l32i     a0,a1,0x1c
402012a5 c2 21 06     l32i     a12,a1,0x18
402012a8 d8 51        l32i.n   a13,a1,0x14
402012aa e8 41        l32i.n   a14,a1,0x10
402012ac f8 31        l32i.n   a15,a1,0xc
402012ae 12 c1 20     addi     a1,a1,0x20
402012b1 0d f0        ret.n
402012b3 00           ??       00h

                  DAT_402012b4

402012b4 98 eb fe 3f  undefined4 3FFEEB98h

                  DAT_402012b8
```

```
      if (param_3 == 3) {
        if (((uint)(param_2[2] ^ *param_2) ^ (uint)param_2[1] << 1) == 0x3d) {
          iVar3 = 0x14;
          do {
            (*(code *)PTR_FUN_4020118c)(2,0);
            ledFlash(0x28);
            (*(code *)PTR_FUN_4020118c)(2,1);
            iVar3 = iVar3 + -1;
            ledFlash(0x28);
          } while (iVar3 != 0);
          goto LAB_4020129d;
        }
      }
      else {
        if (6 < param_3) goto LAB_40201271;
      }
    }
    else {
      iVar3 = (*DAT_40201114)(param_2,PTR_s_restart_40201174);
      if (iVar3 == 0) {
        iVar3 = compare(param_2 + 8,param_3 - 8);
        if (iVar3 != 0) {
          FUN_4020c4e8(uVar1,PTR_s_Restart_40201178);
          FUN_4020bd4c(DAT_4020117c);
        }
        goto LAB_4020129d;
      }
LAB_40201271:
      iVar3 = (*DAT_40201114)(param_2,PTR_s_ADMIN_40201180,6);
      puVar4 = PTR_s_Unknown_40201188;
      if (iVar3 == 0) {
        iVar3 = compare(param_2 + 6,param_3 - 6);
        if (iVar3 != 0) {
          *DAT_40201184 = 1;
        }
        goto LAB_4020129d;
      }
    }
  }
  FUN_4020c454(uVar1,puVar4);
LAB_4020129d:
  FUN_4020c480(uVar1);
  return;
}
```

```
4020129d    mov.n a2,a14
```

We entered that function for the comparison and found that it compares the rest of the bytes with `4Hxl4Z69J6Uht1w4`.



```
40201110 48 df 00 40     undefined4  4000DF48h

                    DAT_40201114

40201114 a8 bf 00 40     undefined4  4000BFA8h

                    ****************************
                    *
                    ****************************
                    undefined compare()
     undefined         a2:1          <RETURN>
                    compare
40201118 12 c1 d0         addi       a1,a1,-0x30
4020111b c2 61 0a         s32i       a12,a1,0x28
4020111e d2 61 09         s32i       a13,a1,0x24
40201121 c2 a0 10         movi       a12,0x10
40201124 dd 03            mov.n      a13,a3
40201126 31 f9 ff         l32r       a3=>s_4Hxl4

40201129 e9 81            s32i.n     a14,a1,0x20
4020112b 4d 0c            mov.n      a4,a12
4020112d ed 02            mov.n      a14,a2
4020112f ca 21            add.n      a2,a1,a12
40201131 09 b1            s32i.n     a0,a1,0x2c
40201133 01 f7 ff         l32r       a0,DAT_4020
40201136 c0 00 00         callx0     a0
40201139 3d 0e            mov.n      a3,a14
4020113b 4d 0d            mov.n      a4,a13
4020113d 2d 01            mov.n      a2,a1
4020113f c5 77 13         call0      FUN_402148b
40201142 ca 31            add.n      a3,a1,a12
40201144 4d 0c            mov.n      a4,a12
40201146 2d 01            mov.n      a2,a1
40201148 01 f3 ff         l32r       a0,DAT_4020
```

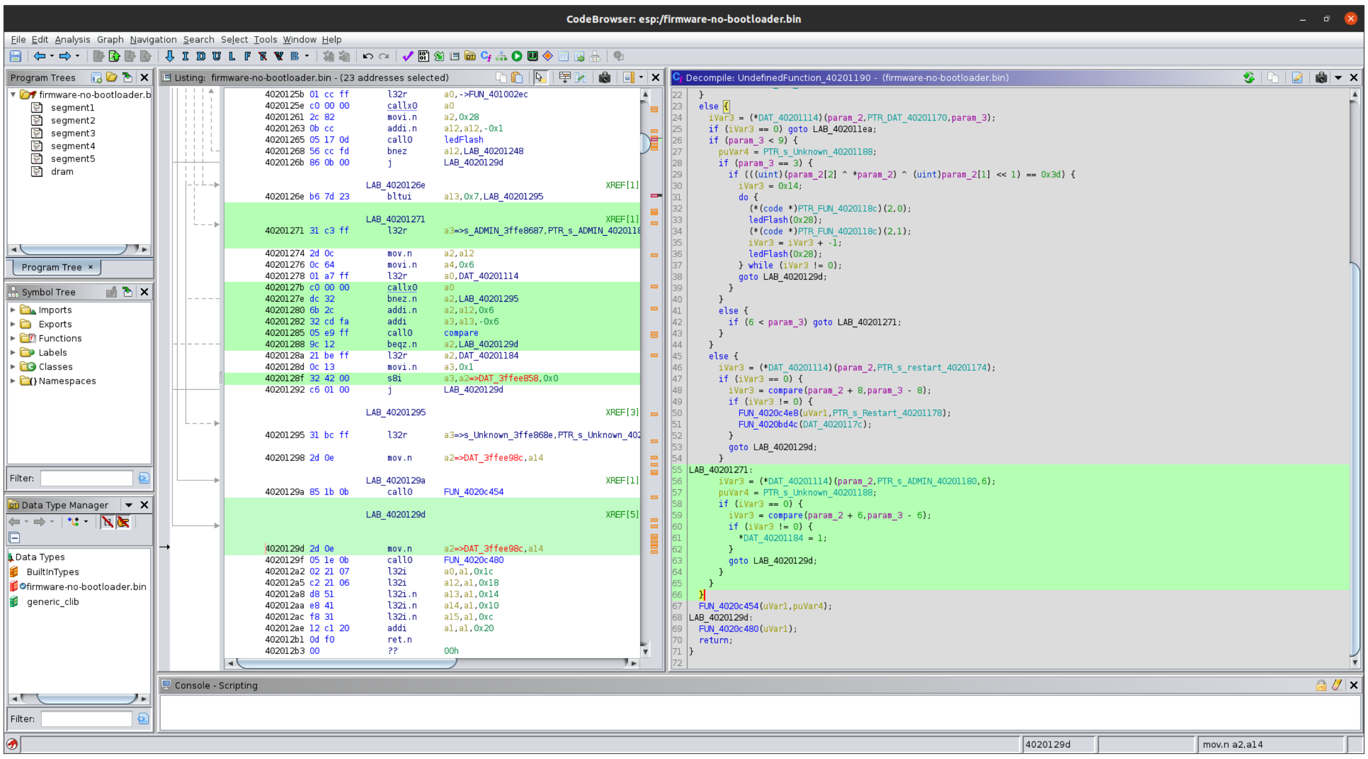
```c
bool compare(undefined4 param_1,undefined4 param_2)

{
  int iVar1;
  undefined auStack48 [16];
  undefined auStack32 [16];

  (*DAT_40201110)(auStack32,PTR_s_4Hxl4Z69J6Uht1w4_4020110c,0x10);
  FUN_402148bc(auStack48,param_1,param_2);
  iVar1 = (*DAT_40201114)(auStack48,auStack32,0x10);
  return iVar1 == 0;
}
```

Then, we passed the message `'restart 4Hxl4Z69J6Uht1w4'` to the board and it would restart after receiving the mqtt package.

```
# input
mosquitto_pub -h localhost -p 8888 -t led/wemos17/action -m 'restart
4Hxl4Z69J6Uht1w4'

# output
Message arrived [led/wemos17/action] Restart

ets Jan  8 2013,rst cause:2, boot mode:(3,7)

load 0x4010f000, len 3584, room 16
tail 0
chksum 0xb0
csum 0xb0
v2843a5ac
~ld
*WM:
*WM: AutoConnect
*WM: Connecting as wifi client...
*WM: Status:
*WM: 6
*WM: Using last saved values, should be faster
*WM: Connection result:
*WM: 3
*WM: IP Address:
*WM: 192.168.12.2
connected...yeey :)
*WM: freeing allocated params!
syssec.ewi.utwente.nl
Attempting MQTT connection...connected
```

ADMIN Command

Similar to the Restart command, in the figure below, it shows that if the first 6 bytes equal to `ADMIN` and the rest bytes equal to the correct password `4Hxl4Z69J6Uht1w4`. If the verification is passed, it will set `*DAT_40201184 = 1`.



```
4020125b 01 cc ff       l32r      a0,->FUN_401002ec
4020125e c0 00 00       callx0    a0
40201261 2c 82          movi.n    a2,0x28
40201263 0b cc          addi.n    a12,a12,-0x1
40201265 05 17 0d       call0     ledFlash
40201268 56 cc fd       bnez      a12,LAB_40201248
4020126b 86 0b 00       j         LAB_4020129d

                     LAB_4020126e                                XREF[1]
4020126e b6 7d 23       bltui     a13,0x7,LAB_40201295

                     LAB_40201271                                XREF[1]
40201271 31 c3 ff       l32r      a3=>s_ADMIN_3ffe8687,PTR_s_ADMIN_402011

40201274 2d 0c          mov.n     a2,a12
40201276 0c 64          movi.n    a4,0x6
40201278 01 a7 ff       l32r      a0,DAT_40201114
4020127b c0 00 00       callx0    a0
4020127e dc 32          bnez.n    a2,LAB_40201295
40201280 6b 2c          addi.n    a2,a12,0x6
40201282 32 cd fa       addi      a3,a13,-0x6
40201285 05 e9 ff       call0     compare
40201288 9c 12          beqz.n    a2,LAB_4020129d
4020128a 21 be ff       l32r      a2,DAT_40201184
4020128d 0c 13          movi.n    a3,0x1
4020128f 32 42 00       s8i       a3,a2=>DAT_3ffee858,0x0
40201292 c6 01 00       j         LAB_4020129d

                     LAB_40201295                                XREF[3]

40201295 31 bc ff       l32r      a3=>s_Unknown_3ffe868e,PTR_s_Unknown_402

40201298 2d 0e          mov.n     a2=>DAT_3ffee98c,a14

                     LAB_4020129a                                XREF[1]
4020129a 85 1b 0b       call0     FUN_4020c454

                     LAB_4020129d                                XREF[5]

4020129d 2d 0e          mov.n     a2=>DAT_3ffee98c,a14
4020129f 05 1e 0b       call0     FUN_4020c480
402012a2 02 21 07       l32i      a0,a1,0x1c
402012a5 c2 21 06       l32i      a12,a1,0x18
402012a8 d8 51          l32i.n    a13,a1,0x14
402012aa e8 41          l32i.n    a14,a1,0x10
402012ac f8 31          l32i.n    a15,a1,0xc
402012ae 12 c1 20       addi      a1,a1,0x20
402012b1 0d f0          ret.n
402012b3 00             ??        00h
```

```c
  }
  else {
    iVar3 = (*DAT_40201114)(param_2,PTR_DAT_40201170,param_3);
    if (iVar3 == 0) goto LAB_402011ea;
    if (param_3 < 9) {
      puVar4 = PTR_s_Unknown_40201188;
      if (param_3 == 3) {
        if (((uint)(param_2[2] ^ *param_2) ^ (uint)param_2[1] << 1) == 0x3d) {
          iVar3 = 0x14;
          do {
            (*(code *)PTR_FUN_4020118c)(2,0);
            ledFlash(0x28);
            (*(code *)PTR_FUN_4020118c)(2,1);
            iVar3 = iVar3 + -1;
            ledFlash(0x28);
          } while (iVar3 != 0);
          goto LAB_4020129d;
        }
      }
      else {
        if (6 < param_3) goto LAB_40201271;
      }
    }
    else {
      iVar3 = (*DAT_40201114)(param_2,PTR_s_restart_40201174);
      if (iVar3 == 0) {
        iVar3 = compare(param_2 + 8,param_3 - 8);
        if (iVar3 != 0) {
          FUN_4020c4e8(uVar1,PTR_s_Restart_40201178);
          FUN_4020bd4c(DAT_4020117c);
        }
        goto LAB_4020129d;
      }
LAB_40201271:
      iVar3 = (*DAT_40201114)(param_2,PTR_s_ADMIN_40201180,6);
      puVar4 = PTR_s_Unknown_40201188;
      if (iVar3 == 0) {
        iVar3 = compare(param_2 + 6,param_3 - 6);
        if (iVar3 != 0) {
          *DAT_40201184 = 1;
        }
        goto LAB_4020129d;
      }
    }
  }
  FUN_4020c454(uVar1,puVar4);
LAB_4020129d:
  FUN_4020c480(uVar1);
  return;
}
```

CodeBrowser: esp:/firmware-no-bootloader.bin
File Edit Analysis Graph Navigation Search Select Tools Window Help
Listing: firmware-no-bootloader.bin - (23 addresses selected)
Decompile: UndefinedFunction_40201190 - (firmware-no-bootloader.bin)
Program Trees
firmware-no-bootloader.b
segment1
segment2
segment3
segment4
segment5
dram
Program Tree
Symbol Tree
Imports
Exports
Functions
Labels
Classes
Namespaces
Filter:
Data Type Manager
Data Types
BuiltInTypes
firmware-no-bootloader.bin
generic_clib
Filter:
Console - Scripting
4020129d
mov.n a2,a14

Appendix

search.py

```python
#!/usr/bin/env python
# coding: utf-8

# In[1]:
ascii_start = 0x20
ascii_end = 0x7E

# In[2]:
f = open("./commands.txt", "a+")

for param2_0 in range(ascii_start, ascii_end + 1):
    for param2_1 in range(ascii_start, ascii_end + 1):
        for param2_2 in range(ascii_start, ascii_end + 1):
            if (((param2_2 ^ param2_0) ^ (param2_1 << 1)) == 0x3D):
                command = chr(param2_0) + chr(param2_1) + chr(param2_2)
                f.write(command + '\n')


f.close()
```

validation.py

```
import os
from time import sleep

f = open("./commands.txt", "r")
command = True
while(command):
    command = f.readline()[:-1]
    command = command.replace("'", "'\"'\"'")
    os.system("mosquitto_pub -h localhost -p 8888 -t led/wemos17/action -m
" + "'" + command + "'")
    sleep(2)
f.close()
```