Weighing in on Issues with “Cloud Scale”

Michael Coppola
Summercon 2013
Who am I?

- #
- Student at Northeastern University
- CTF every now and then
- http://poppopret.org/
In this presentation

- A bit of this
- A bit of that
- Successes, failures
- Tool development
- Mad t3chn1quez
Cloud scaling

OMG CLOUD
Wireless Scale WS-30

Step up for instant weighing and BMI.
Attack surface

- WiFi / Bluetooth driver and application
- Network communications
- Application input parsing
- No network services (no open ports)
Can the Wireless Scale be kept up-to-date with new software?

Yes, Wireless Scale is a smart and updateable device. It can be updated with new software to add new features, make it compatible with new apps or devices, or fix issues that our users have reported. If you have a Wi-Fi network, software update will occur automatically at night as soon as an update is available. If you don’t have a Wi-Fi network, the Withings app will advise you when an update is available and provide update instructions.
Sniffing network traffic

- Associate to WiFi using config from Bluetooth
- DNS lookup for scalews.withings.net
- JSON-based protocol over plaintext HTTP
Sniffing network traffic

- Challenge-handshake authentication
- Send device info (MAC, fw version, battery)

POST /cgi-bin/once HTTP/1.1
action=get

HTTP/1.1 200 OK
{
  "status": 0,
  "body": {
    "once": "00d016bf-242e0bb1"
  }
}

POST /cgi-bin/session HTTP/1.1
action=new
&auth=00:24:e4:06:59:dc
&hash=25fd29132cf66a5cdf1a7efdc673be26
&mfgid=262151&currentfw=211
&batterylvl=69&duration=30&zreboot=1
MITM'ing network traffic

- We want the firmware image
  - Maybe sending a lesser fw version will initiate an update

- We don't know how to complete the handshake, so we still need the device

- DNS spoof the device, interpose ourselves in the session
DNS spoofing the handshake

1. Device initiates connection
2. Server responds with nonce
3. Device sends calculated hash with diagnostic info
3'. Hacker modifies the fw version and sends to the server
Firmware header

- No results from binwalk
- Lots of strings → likely no encryption or compression
- Multiple null padded sections → likely multiple objects packaged together

```
[mncoppola@dysthyemia firmware]$ hexdump -C wbs03_211.bin | head
00000000  7c 47 0a 00 01 00 00 00 d3 00 00 00 28 00 00 00 | G.............|
00000010  f0 61 06 00 65 f6 3b 0c 18 62 06 00 a4 11 03 00 | .a..e.;b.....|
00000020  bc 73 09 00 ba d3 00 00 00 00 00 01 20 e9 91 02 00 | .s............|
00000030  cd 91 02 00 19 96 02 00 cd 91 02 00 cd 91 02 00 | ................|
00000040  cd 91 02 00 cd 91 02 00 cd 91 02 00 cd 91 02 00 | ................|
00000050  cd 91 02 00 05 58 04 00 cd 91 02 00 cd 91 02 00 | ................|
00000060  d5 58 04 00 11 59 04 00 45 8b 02 00 6d 8b 02 00 | X...Y...E...m...|
00000070  ed 3a 01 00 cd 91 02 00 cd 91 02 00 cd 91 02 00 | ................|
00000080  cd 91 02 00 cd 91 02 00 cd 91 02 00 cd 91 02 00 | ................|
```
Identifying the MCU

- MK20DN512ZVLL10
- Freescale Kinetis K20 family
- ARM Cortex-M4 (ARMv7)
- Memory-mapped peripheral registers
Locating code blocks

- Find a dense area of bytes and start disassembling

- Common bytes:
  - ARM: 0xe*
  - Padding: 0xbf00 (nop)

- Byte search the addresses of strings and disassemble backwards
Things aren't lining up...
basefind.py

• Every dword in file is treated as a pointer

• Does base + dword point to the beginning of a string?

• Repeat for all possible base addresses

• Highest score is likely the correct base address

sh-4.2$ ./basefind.py noheader2.bin
Scanning binary for strings...
Total strings found: 2945
Scanning binary for pointers...
Total pointers found: 115282
Trying base address 0x0
New highest score, 0x0: 71
New highest score, 0x1000: 72
New highest score, 0x2000: 88
New highest score, 0x3000: 92
New highest score, 0x4000: 2170
Trying base address 0x10000
Trying base address 0x20000
Trying base address 0x30000
Trying base address 0x40000
Trying base address 0x50000
Trying base address 0x60000
Trying base address 0x70000
Trying base address 0x80000
Trying base address 0x90000
Trying base address 0xa0000
Trying base address 0xb0000
Trying base address 0xc0000
Trying base address 0xd0000
Trying base address 0xe0000
Trying base address 0xf0000
Trying base address 0x100000
Trying base address 0x110000
Trying base address 0x120000
Trying base address 0x130000
Trying base address 0x140000
Trying base address 0x150000
Trying base address 0x160000
Trying base address 0x170000
Trying base address 0x180000
Base address 0x4000
One room available in Huaihai xi lu - Panyu Lu

¥2,700

Hi everybody!
I'm leaving shanghai and my room will be available on 21th of October. It's a nice room in a 4 bedroom shared appartment located in Huai Hai xi road and panyu Road. You'll be sharing this apartement with one HongKongnese girl, one latvian guy and one french guy.
The flat has a spacious living room, a balcony with a nice view on a small park, a full kitchen, and all needed services: Washing machine, TV, DVD player, wireless internet, Air conditioner for every room and for the living room. An aji is passing by twice a week to clean the flat.
The appartement is very convenient in term of location:
- 5 min from Hongqiao Lu subway station, line 3/4/10
- 15 min walk from Xujiuhui subway station, line 1
- 1 min walk from bus 911, 926 which lead you directly to people square
- 15 min from Grand Gateway, Xujiuhui
Send me a message to my personal mail fdusanter@gmail.com if you are interested!
Firmware subsystems

- LibCURL – HTTP library
- lwIP – TCP/IP stack
- DbLib – Key=value store
- WsLib – Interact with Withings web services
- CnLib – Network connection management
- UsLib – User management
- LibPairing – Bluetooth pairing
### DbLib_GetElement()

<table>
<thead>
<tr>
<th>Index</th>
<th>Name</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>mac_address</td>
<td>0x12</td>
</tr>
<tr>
<td>2</td>
<td>hostname</td>
<td>0x40</td>
</tr>
<tr>
<td>3</td>
<td>secret</td>
<td>0x11</td>
</tr>
<tr>
<td>4</td>
<td>time_constant</td>
<td>0xc</td>
</tr>
<tr>
<td>5</td>
<td>users</td>
<td>0x280</td>
</tr>
<tr>
<td>6</td>
<td>uid</td>
<td>0x26</td>
</tr>
<tr>
<td>7</td>
<td>pref_lang</td>
<td>0x6</td>
</tr>
<tr>
<td>8</td>
<td>scale_constant</td>
<td>0x10</td>
</tr>
<tr>
<td>9</td>
<td>ssid</td>
<td>0x6c</td>
</tr>
<tr>
<td>10</td>
<td>IP config</td>
<td>0x16</td>
</tr>
<tr>
<td>11</td>
<td>mfg_id</td>
<td>0x4</td>
</tr>
<tr>
<td>12</td>
<td>calibration</td>
<td>0x14e</td>
</tr>
<tr>
<td>14</td>
<td>last connection</td>
<td>0x20</td>
</tr>
<tr>
<td>16</td>
<td></td>
<td>0x2c</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Index</th>
<th>Name</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>17</td>
<td>factory_mode</td>
<td>0x4</td>
</tr>
<tr>
<td>19</td>
<td>debug traces</td>
<td>0x0</td>
</tr>
<tr>
<td>20</td>
<td>factory weight verif</td>
<td>0x4c</td>
</tr>
<tr>
<td>22</td>
<td>wifi_country</td>
<td>0x4</td>
</tr>
<tr>
<td>24</td>
<td>wpa_key</td>
<td>0x50</td>
</tr>
<tr>
<td>25</td>
<td>battery level</td>
<td>0x8</td>
</tr>
<tr>
<td>27</td>
<td>wifi_delay</td>
<td>0x4</td>
</tr>
<tr>
<td>28</td>
<td>Bluetooth config</td>
<td>0x16</td>
</tr>
<tr>
<td>32</td>
<td></td>
<td>0x4</td>
</tr>
<tr>
<td>33</td>
<td>calibration parameters</td>
<td>0x4e</td>
</tr>
<tr>
<td>34</td>
<td></td>
<td>0x10</td>
</tr>
<tr>
<td>37</td>
<td></td>
<td>0x4</td>
</tr>
<tr>
<td>38</td>
<td></td>
<td>0x82</td>
</tr>
</tbody>
</table>
Device association request

```c
printf("[WSLIB] In StartSession, once <\%s\n", &once);
DBLib_GetElement(1, mac_addr, 0x12);
DBLib_GetElement(3, secret, 0x11);
sprintf(resbuf, "%s:%s:%s", mac_addr, secret, &once);
printf2("resbuf = %s\n", resbuf);
hash = get_hash(resbuf);
mfgid = get_mfg_id();
currentfw = get_firmware_version();
batterylvl = get_battery_level();
zreboot = get_zreboot();
res = snprintf(
    POST_fields,
    220,
    "action=new&auth=%s&hash=%s&mfgid=%d&currentfw=%d&batterylvl=%d&duration=30&zreboot=%d",
    mac_addr,
    hash,
    mfgid,
    currentfw,
    batterylvl,
    zreboot);
if ( res > 219 )
    bof_detected = ((unsigned int)(res + 1) <= 0) | 1;
else
    bof_detected = (unsigned int)(res + 1) <= 0;
if ( bof_detected )
{
    v18 = "[WSLIB] StartSession (new) buffer overflow\n";
    LABEL_29:
    printf(v18);
    goto LABEL_30;
}
if ( do_HTTP_POST_request(hostname, "session", POST_fields, resbuf, 0x767) )
{
    err msg 0 = "[WSLIB] StartSession (new) HTTP error"
    goto LABEL_26;
}
sub_168EC();
v19 = new_JSON_parser(&config);
if ( !v19 )
{
    v18 = "[WSLIB] StartSession JSON mem error"
    goto LABEL_29;
}
```
Cracking the firmware header

[mncoppola@dysthymia firmware]$ hexdump -C wbs03_211.bin | head -n3
00000000 7c 47 0a 00 01 00 00 00  d3 00 00 00 28 00 00 00 |...G.............|
00000010  f0 61 06 00 65 f6 3b 0c 18 62 06 00 a4 11 03 00  |a..e.;...b......|
00000020  bc 73 09 00 ba d3 00 00 00 00 01 20 e9 91 02 00  |s.............|
[mncoppola@dysthymia firmware]$ hexdump -C wbs03_211.bin | tail -n3
000a4760  d1 1c 60 01 e0 13 68 0b 60 14 60 10 bd 00 00 4e  |...`...h.`...N|
000a4770  fc 04 ff ff ff ff ff ff ff 65 68 f3 4f  |.........eh.0|
000a477c

if ( open_spi_flash(&SPI0_regs_base) )
{
    printf("Fail to open flash\n");
} else
{
    memset(&fw, 0, 0x128u);
    read_from_SPI(&SPI0_regs_base, (char *)&fw, bank_addr, 0x28);
    sub_235D00(&SPI0_regs_base);
    snprintf(str, 0x20, "blk%d_tbl_", op);
    printf("%s\n", str, fw.total_size);
    printf("%s\n", str, fw0.gold);
    printf("%s\n", str, fw0.version);
    printf("%s\n", str, fw0.kinetis_address);
    printf("%s\n", str, fw0.kinetis_size);
    printf("%s\n", str, fw0.wifi_address);
    printf("%s\n", str, fw0.bluetooth_address);
    printf("%s\n", str, fw0.bluetooth_size);
    ret = check_firmware_CRC32(crc_op, &crc, &tmp);
    printf("blk%d_computed_crc=0x%08X\n", op, crc);
    *(DWORD *)&valid = "no";
    if (!ret )
    {
        *(DWORD *)&valid = "yes";
        printf("blk% valid=\n", op, *(DWORD *)&valid);
    }
}

LABEL_10:
JUMPOUT(__CS__, fw0.kinetis_crc);

Total size: 673660 bytes
Gold status: 0x1
Firmware version: 211
Kinetis address: 0x28
Kinetis size: 418288 bytes
Kinetis CRC: 0x0c3bf665
WiFi address: 0x66218
WiFi size: 201124 bytes
Bluetooth address: 0x973bc
Bluetooth size: 54202 bytes
Firmware CRC: 0x4ff36865
Reversing the CRC validation

```c
unsigned int __fastcall init_CRC32(unsigned int poly, unsigned int seed)
{
    SIM_SGC6 |= 0x400000u; // TCRC = 1, 32-bit CRC mode
    CRC_CTRL = 0x1000000u; // FXOR = 1, complement data
    CRC_CTRL = 0x4000000u; // TOTR = 10, bits and bytes are transposed for read
    CRC_CTRL = 0x8000000u; // TOT = 10, bits and bytes are transposed for write
    CRC_P0LY = poly;
    CRC_CTRL = 0x2000000u; // WAS = 1, writes to CRC_CRC (data register) are seed values
    CRC_CRC = seed;
    CRC_CTRL &= 0xFDFFF000u; // WAS = 0, writes to CRC_CRC (data register) are data values
    return poly;
}
```

```c
read_from_SPI(&SPI0_regs_base, firmware, bank_addr, 160);
offset = *(DWORD *)&firmware - 4; // first dword is firmware size
if ( (unsigned int)(*(DWORD *)&firmware - 4) <= 0xDFFFC ) // max firmware size 917,500 bytes
{
    read_from_SPI(&SPI0_regs_base, (char *)&firmware_CRC, bank_addr + offset, 4); // get firmware checksum
    init_CRC32(CRC32_P0LY, 0xFFFFFFFF);
    for ( i = 0; ; i = next_i )
    {
        next_i = i + 512;
        if ( i + 512 > (unsigned int)offset )
            break;
        read_from_SPI(&SPI0_regs_base, fw_data, bank_addr + i, 512);
        sub_311F00();
        write_to_CRC(fw_data, 128);
    }
    leftover = offset - i;
    if ( leftover > 0 )
    {
        read_from_SPI(&SPI0_regs_base, fw_data, bank_addr + i, (unsigned __int16)leftover);
        write_to_CRC(fw_data, leftover >> 2);
    }
    calculated_CRC = CRC_CRC;
```
Crafting arbitrary images

[mncoppola@dysthymia scale2]$ ./image_chksum wbs03_211_mod.bin summercon.bin
Image is 673660 bytes
Found WS-30 firmware image:
  Total size: 673660
  Gold: 0x00000001
  Kinetis address: 0x28
  Kinetis size: 418288
  Kinetis CRC: 0x00c3bf665
  WiFi address: 0x66218
  WiFi size: 201124
  Bluetooth address: 0x973bc
  Bluetooth size: 54202
  Image CRC: 0x4ff36865

Calculated image CRC: 0x7ac5b543
Calculated Kinetis CRC: 0x476422e1

Patched Kinetis CRC, recalculating image CRC...
Calculated image CRC: 0x5afcaadd
Patched image CRC

Wrote 673660 bytes to summercon.bin
Let's fuzz this thing

- We want some delicious Odayz
- Send it invalid JSON / garbage values
- We need introspection
- Debug console?
Debug console!
8.1 K20 Signal Multiplexing and Pin Assignments

The following table shows the signals available on each pin and the locations of these pins on the devices supported by this document. The Port Control Module is responsible for selecting which ALT functionality is available on each pin.
Spot the UART!
Not a great signal...
Introspection-less

- Code execution on the device would solve all the problems

- Hook the ARM hard fault handler to send crash dumps over the wire

- Re-purpose WsLib to easily send HTTP requests with register context info
Pushing updates

- We'll probably brick it at some point
- Need a recovery plan
- No bootloader we can break into
- Dump (and reprogram) the flash memory!
Identify the flash chip
Desolder the flash chip

- Heat gun + tweezers
- Soldering iron blade tip
- Solder wick
- Rework station
This means you did it wrong
This means you fixed it wrong
Fuck it, new scale
Replace chip with pins
Yup, it still works
Connect to USB programmer
MAC Address: 00:24:e4:06:59:dc
Secret: 649a16bf977d3b3e
Nonce: 00d016bf-242e0bb1

HASH: 25fd29132cf66a5cdf1a7efdc673be26
DEMO TIME
Lessons learned

- ARM compilers are aggressive
  - Reference middle of strings
  - Inline everything possible, even data

- Strings are your friends
  - Find base address
  - Find code blocks
  - Determine symbol names, branch purposes, debugging info

- Lots of help from hardware data sheets and reference manuals

- Embedded system security sucks
Thanks

- Albert Cahalan
- Rob Jerrell
- Jordan Wiens
- Andrew Watts
- Paul Furtado
Greetz

• #busticati

• Marauders

• bliss, thing2
Questions?

@mncoppola
poppopret.org