Hacking Cell Phone Embedded Systems

Keegan Ryan – RECON 2017


nccgroup
The Target

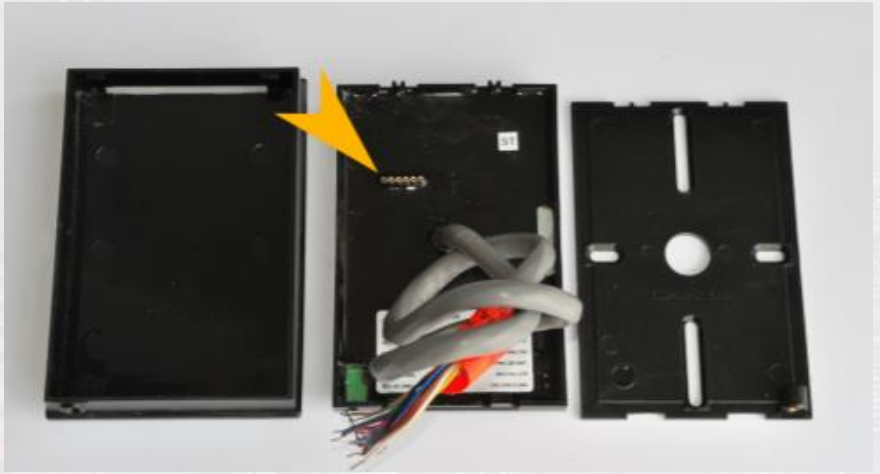
The Target

The Target

Meriac (2010), Churchill

Legacy ICLASS

- Introduced in 2007
- Broken in 2010
  - Master key on every reader
  - Security of card reader broken
  - Protocol reverse engineered
- New version of iCLASS released, but many still use Legacy iCLASS
- Uses ISO15693



CONTACTLESS COMMUNICATION

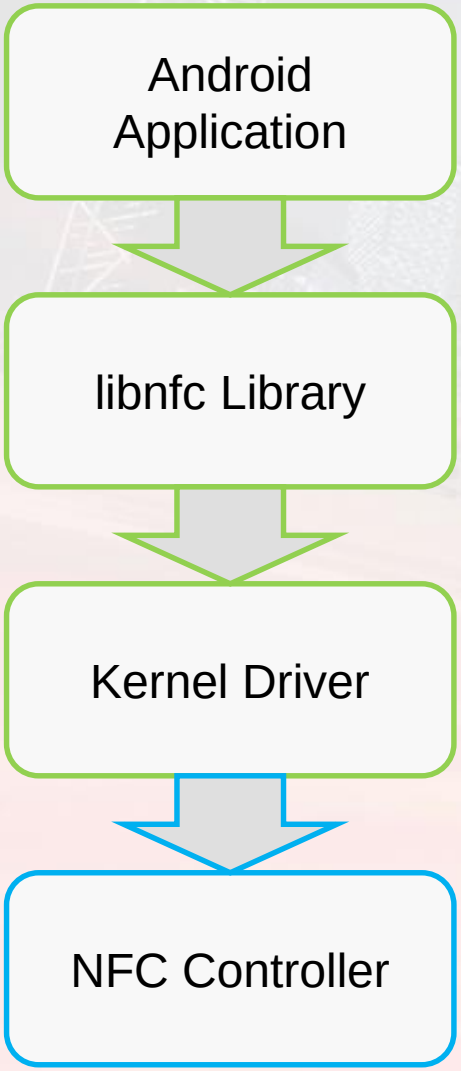
- Carrier frequency: **13.56 MHz ±7 KHz.**
- Data rate : 26 kBit/s (ISO 15 693), 106 or
- Data coding : User can configure :

Meriac (2010), Inside Contactless (2004)

Nexus S

- Introduced in 2010
- One of earliest to support NFC, including ISO15693
- Android source code available
- Cheap

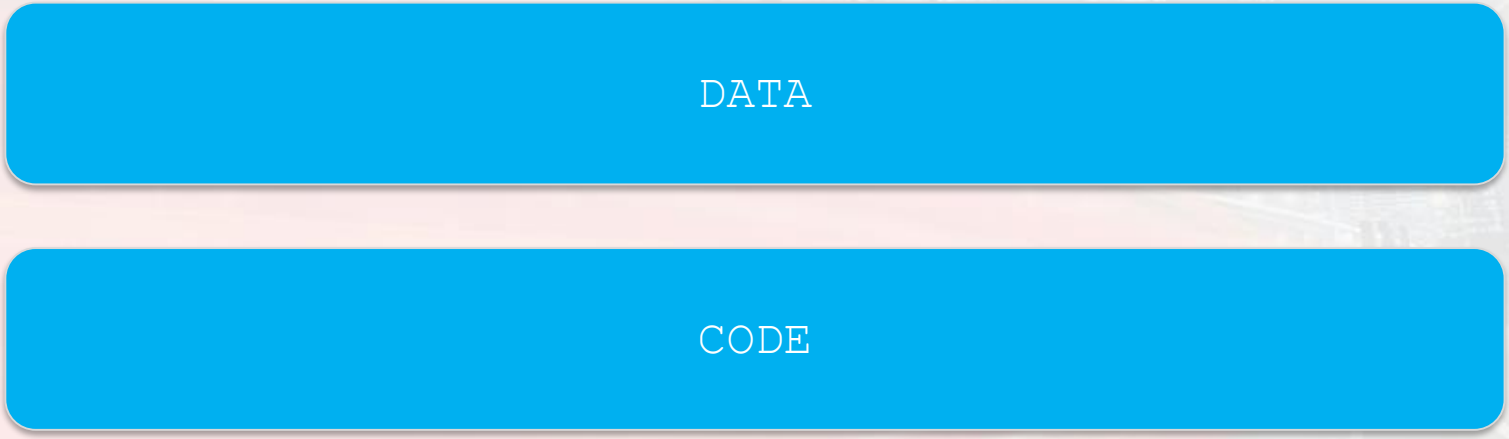
Nexus S



- Try Android app first
- Transceive raw bytes
- CRC added automatically, but we don’t want a CRC
- Not added by libraries
- Not added by kernel
- Must be added by NFC controller chip

PN544

- Separate from Nexus S CPU
- Powered by host or external field
- Supports ISO 15693, Mifare, FeliCa
- Supports firmware upgrades
- Uses 80C51MX Processor



NXP (2010), Wharton (1980)

Investigating the PN544

Firmware Recovery

- `PHDNLD_CMD_READ`
- Pull from update file
- Code signing
  - Protected with SHA1 and RSA-1024
  - Introduced after first devices shipped

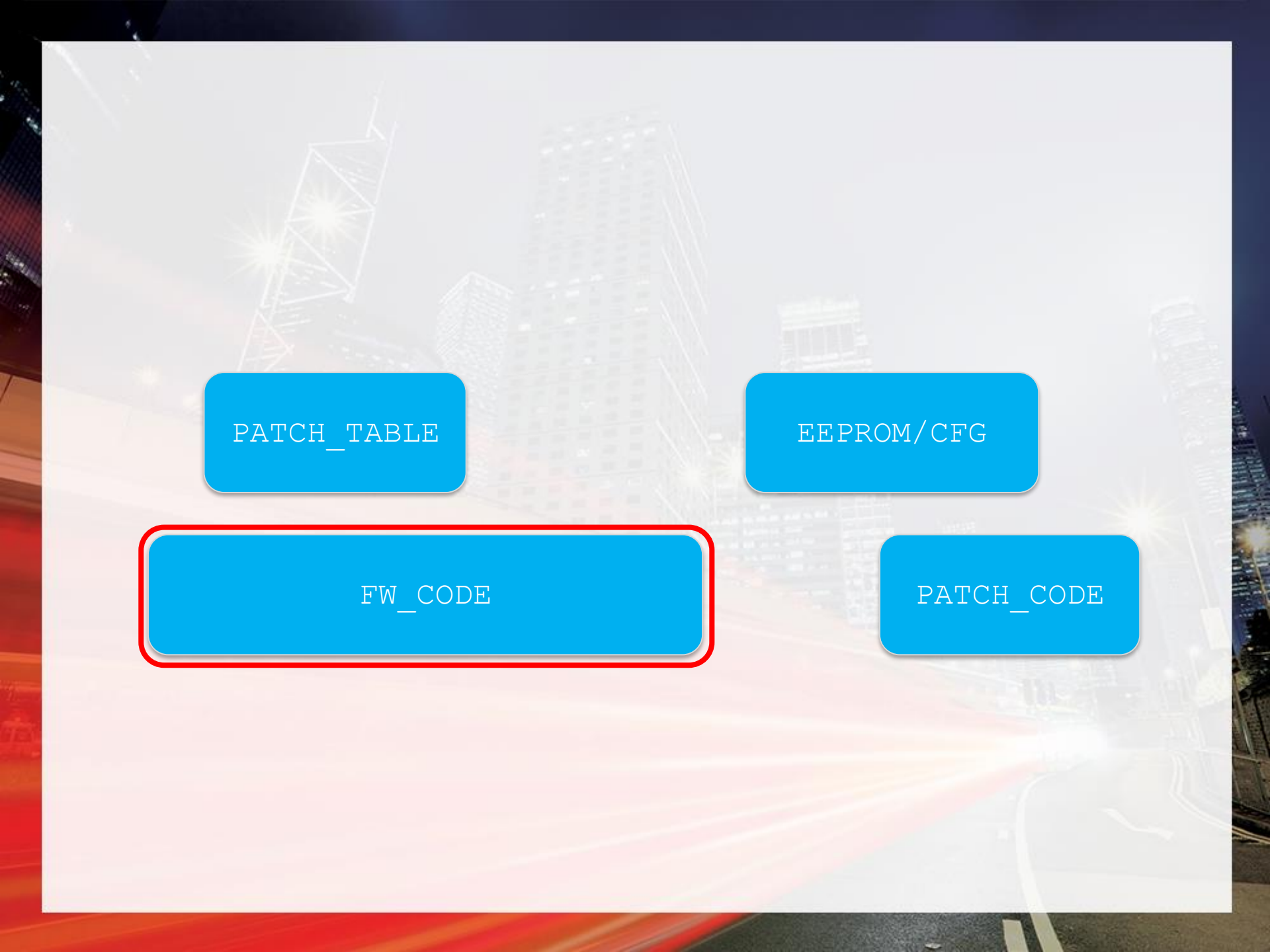
tree 08510355fb6f70462288c28e03fafc99ae9ee7e9
parent df82c4dd7c6d5ad232b5628edf73aa9ea3f8c2c0 [diff]

Patch to add Secure Download Mechanism in the libnfc

This patch permit to support secure download update and
also to avoid locking states in case of download failures

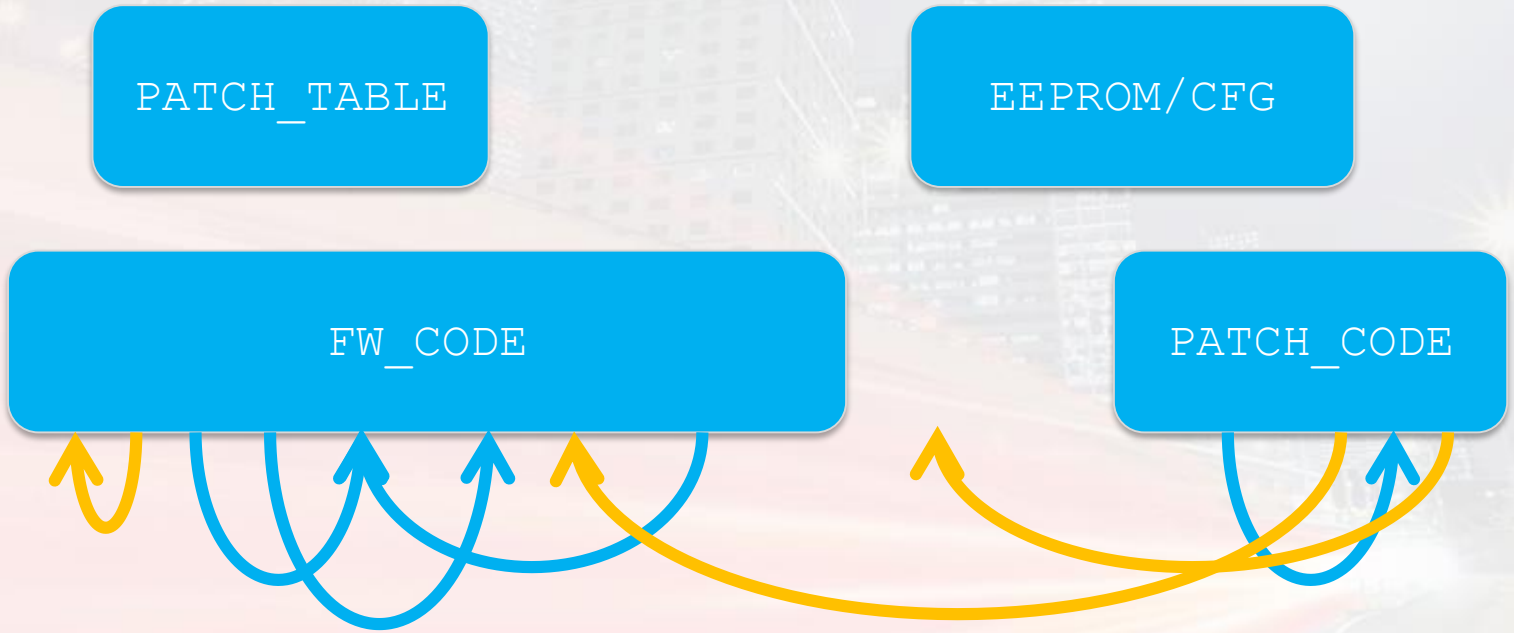
Change-Id: I98aa80976a67b18562ddcff4d085ed415dac4933

  - Need a device never updated past Gingerbread


PATCH_TABLE
EEPROM/CFG
FW_CODE
PATCH_CODE

Reverse Engineering

- Look for strings. There aren’t any.
- Look for CRC constants. They don’t exist.
- Look for usage of the XOR instruction. No help.
- Just start reversing until we find something useful.


PATCH_TABLE
EEPROM/CFG
FW_CODE
PATCH_CODE

Reverse Engineering

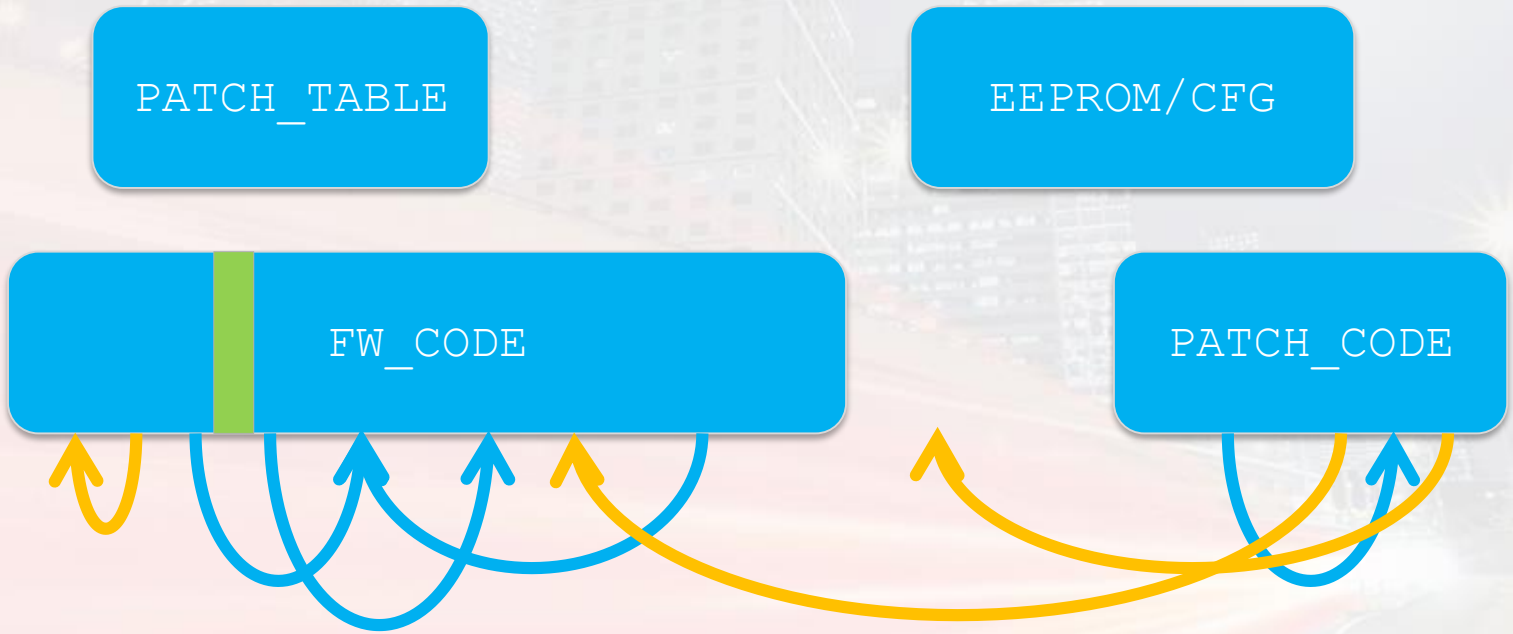
- Reverse commonly called functions
- Find switch function
- Find command switching
- Trace known command IDs through code

Reverse Engineering

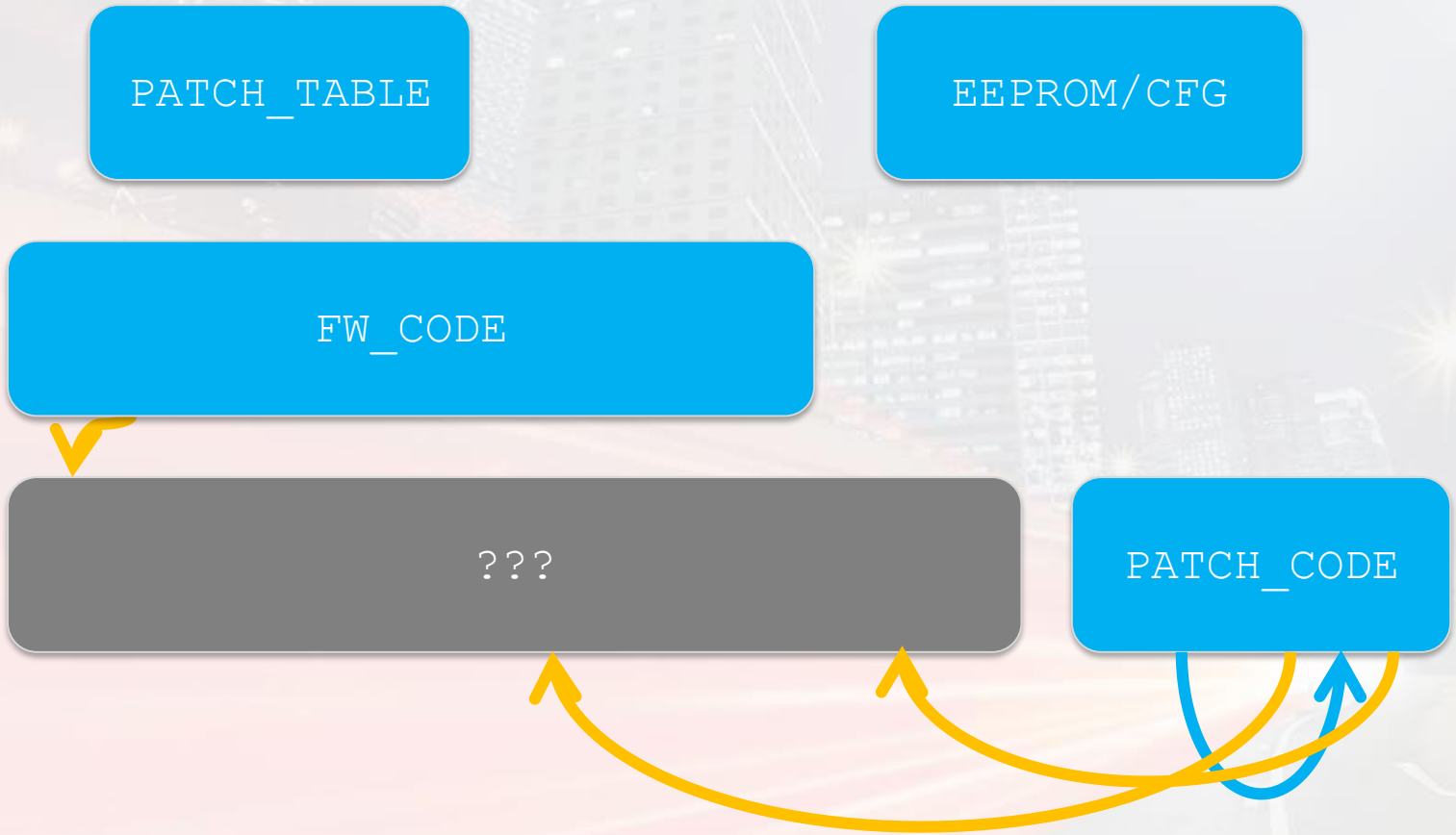
```
352     /* ETSI HCI Specific RF Reader Gates */
353     phHciNfc_RFReaderAGate                  = 0x13,
354     phHciNfc_RFReaderBGate                  = 0x11,
355
356     /* Proprietary Reader Gate */
357     phHciNfc_ISO15693Gate                   = 0x12,
358     phHciNfc_RFReaderFGate                  = 0x14,
359     phHciNfc_JewelReaderGate                = 0x15,
360
361     /* ETSI HCI Card RF Gates */
362     phHciNfc_CETypeBGate                    = 0x21,
363     phHciNfc_CETypeBPrimeGate               = 0x22,
364     phHciNfc_CETypeAGate                    = 0x23,
365     phHciNfc_CETypeFGate                    = 0x24,
366
367     /* NFC-IP1 Gates */
368     phHciNfc_NFCIP1InitRFGate               = 0x30,
369     phHciNfc_NFCIP1TargetRFGate             = 0x31,
370
```

```
seg002_805B0F:                          ; CODE XREF: switch_by_gate+6↑j
                acall   get_53h_GateID
                lcall   switch
; ---------------------------------------------------------------------------
                .word 0x5B68
                .byte 4
                .word 0x5B84
                .byte 5
                .word 0x5B4E
                .byte 0x11
                .word 0x5B56
                .byte 0x12
                .word 0x5B4B
                .byte 0x13
                .word 0x5B51
                .byte 0x14
                .word 0x5B53
                .byte 0x15
                .word 0x5B48
                .byte 0x21
                .word 0x5B48
                .byte 0x22
                .word 0x5B48
                .byte 0x23
                .word 0x5B59
                .byte 0x30
                .word 0x5B5B
                .byte 0x31
                .word 0x5B5D
                .byte 0x90
                .word 0x5B66
                .byte 0x94
                .word 0x5B63
                .byte 0xA0
                .word 0x5B60
                .byte 0xA1
                .word 0
                .word 0x5B87
; End of function switch_by_gate
```

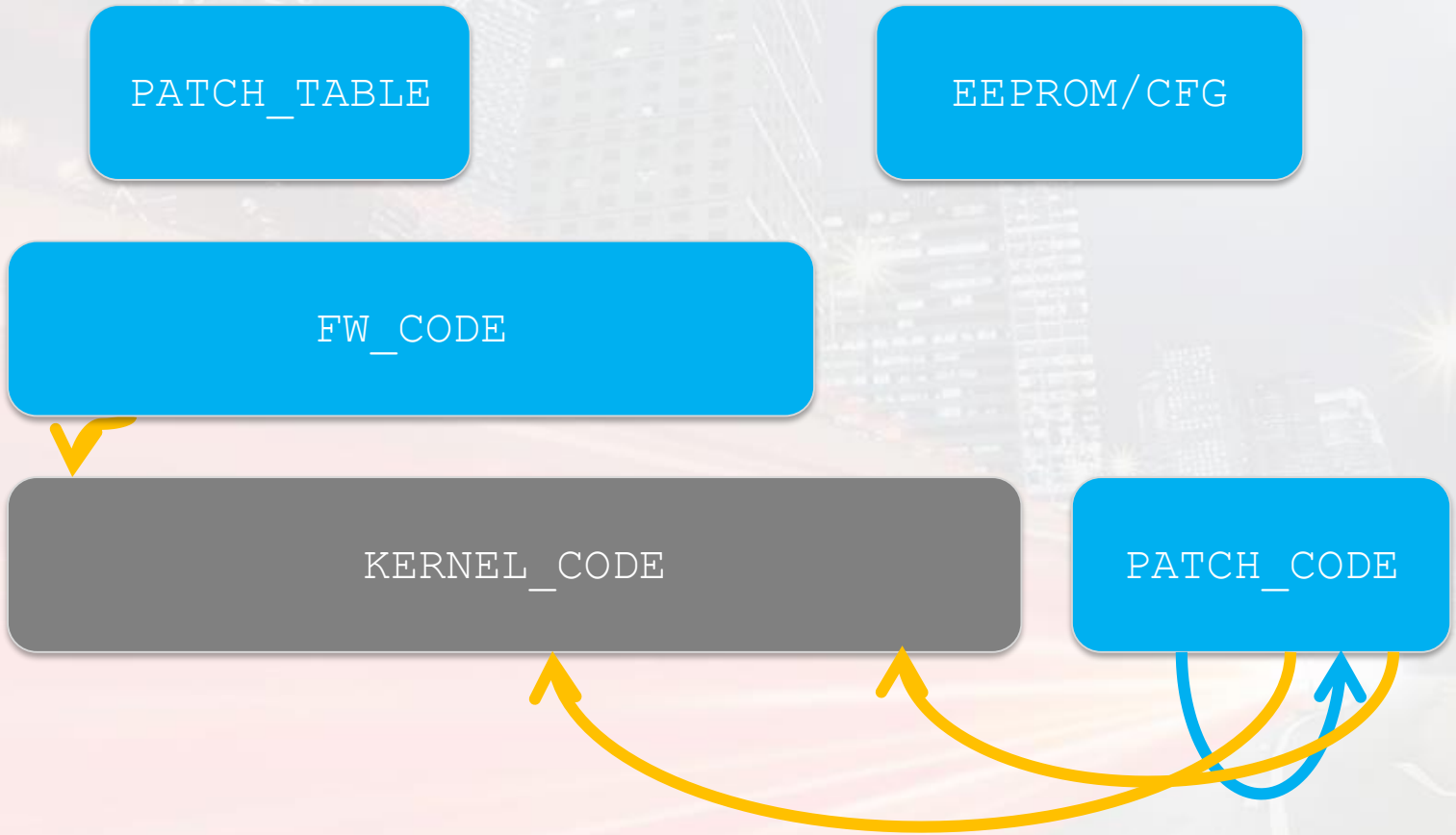
Problem:



Problem: Missing Code



Problem: Missing Code

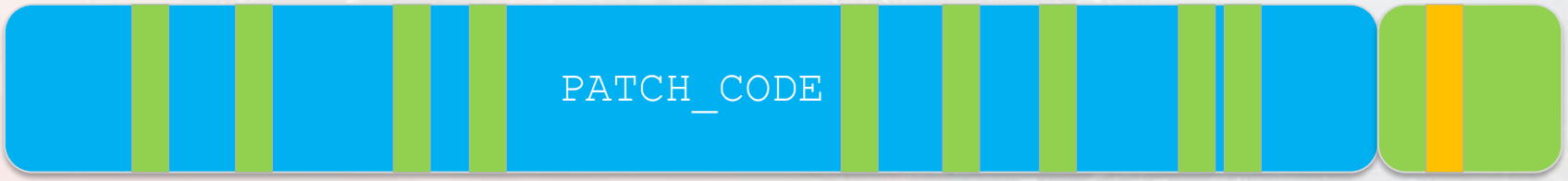


Kernel Recovery

- We understand and can modify `FW_CODE`
- `FW_CODE` doesn't have access to kernel
- We can modify `PATCH_CODE`
- Don't know how to trigger `PATCH_CODE`
- Want to maximize chances of executing our code

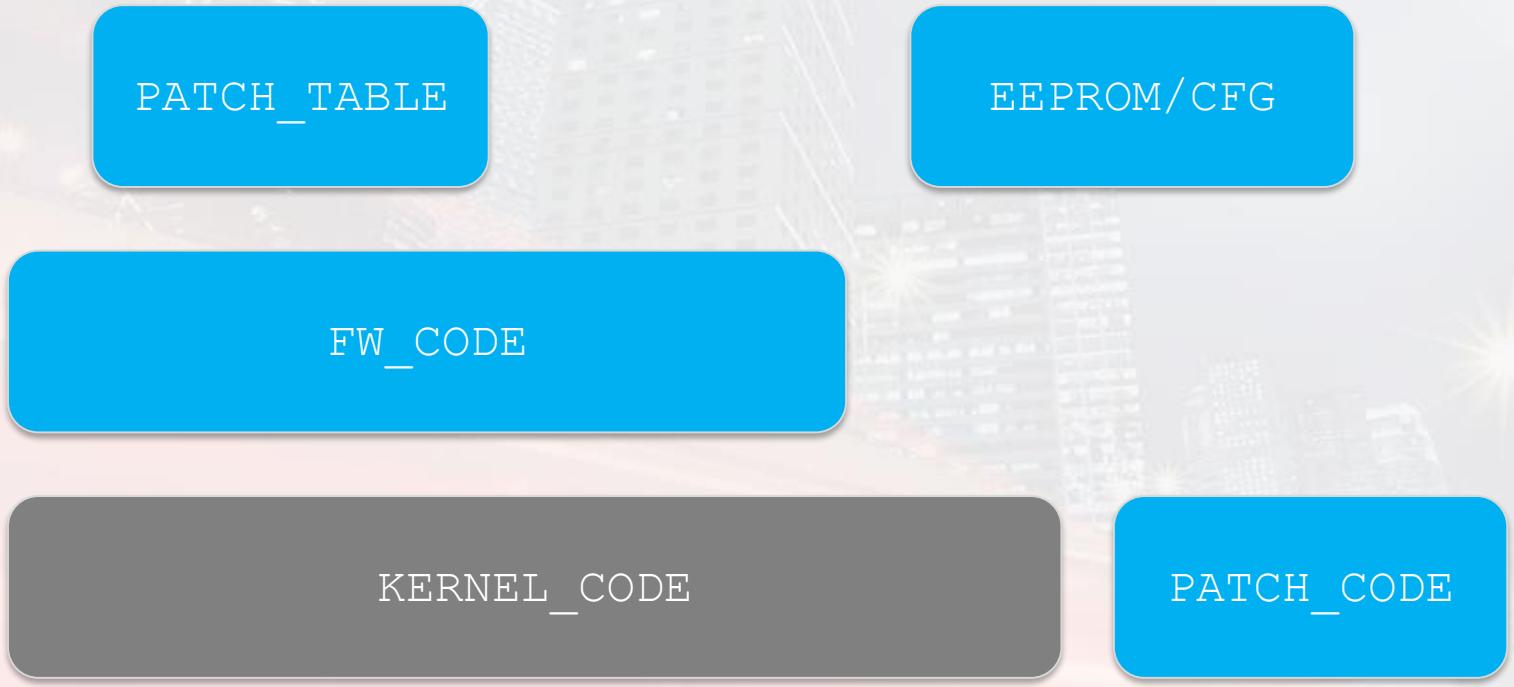
Kernel Recovery

Kernel Recovery

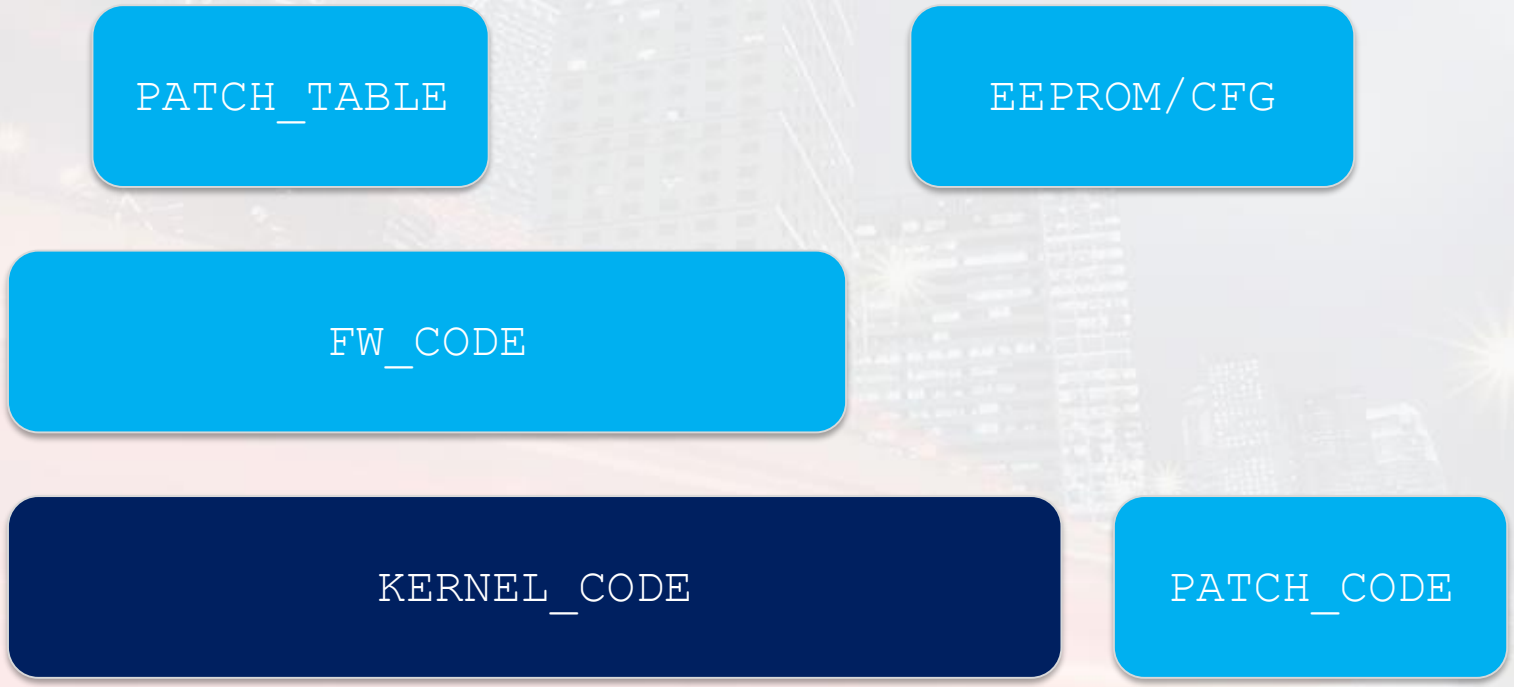


Kernel Recovery

Problem: Missing Code



Problem: Missing Code

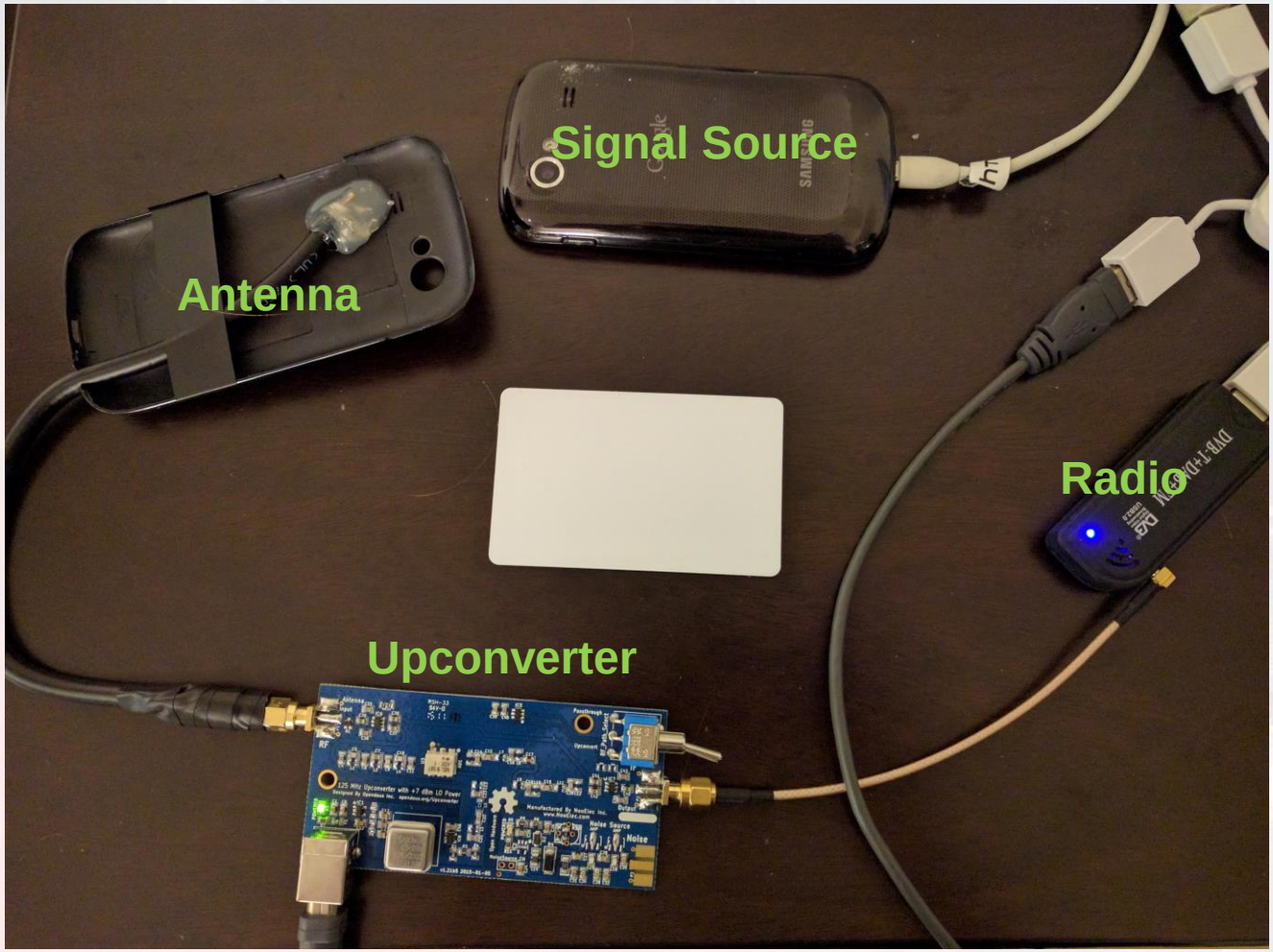


Reverse Engineering Kernel

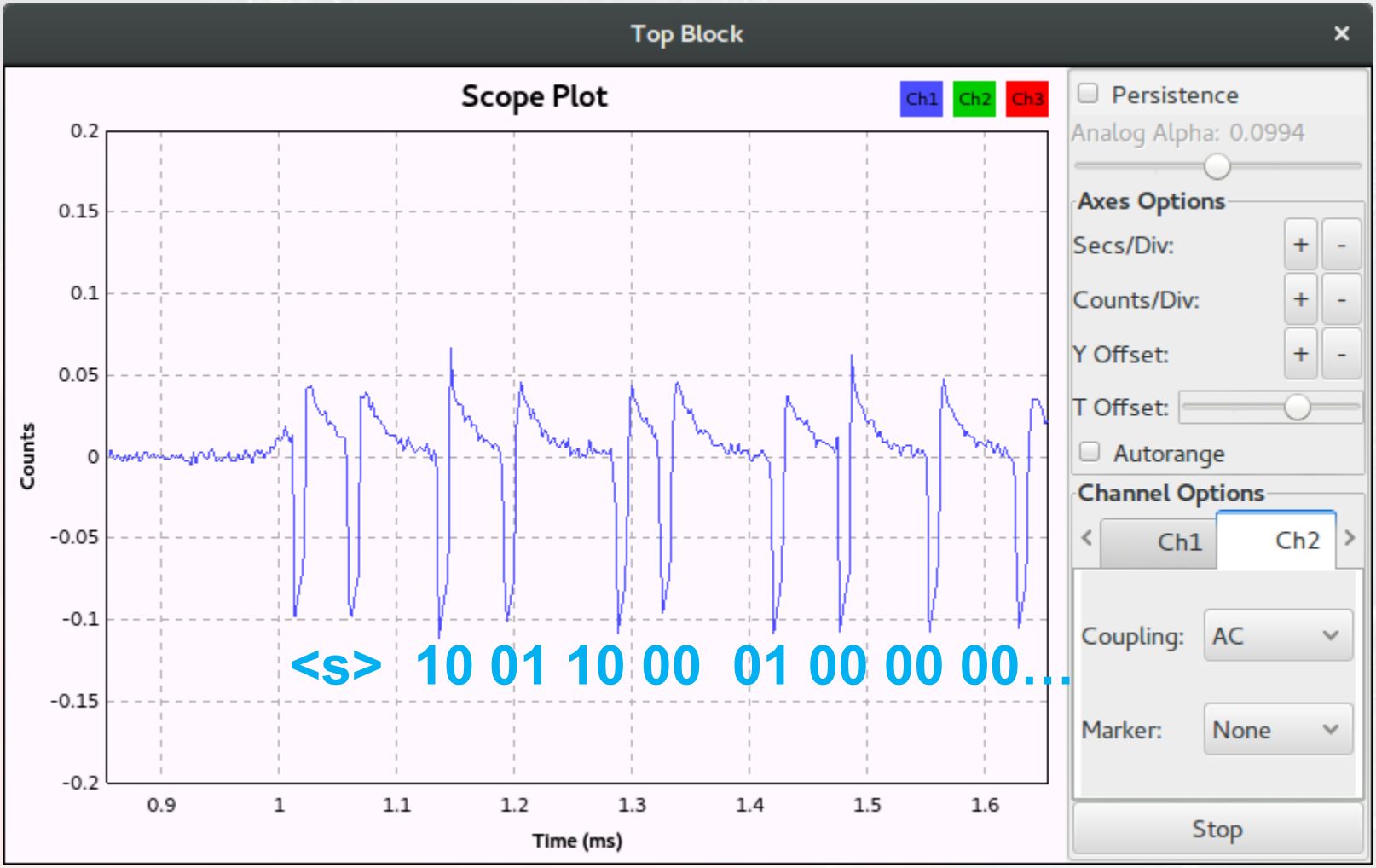
- Look for strings. Still aren’t any.
- Look for CRC constants. Still don’t exist.
- Look for usage of the XOR instruction. No help.
- CRC creation is done by hardware
- Still not impossible, but we need a new approach

Wireless Protocols

SDR Setup

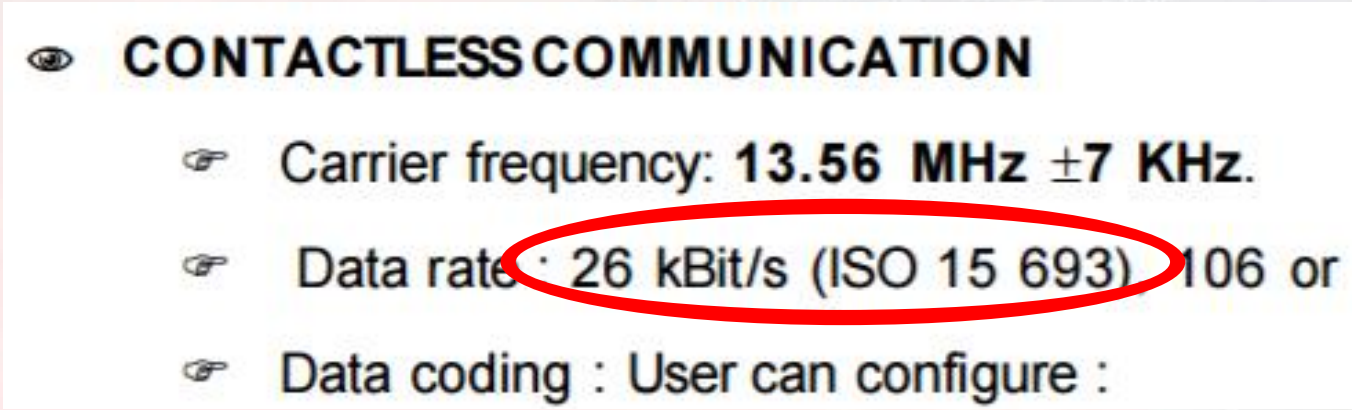


SDR Setup


Top Block
Scope Plot
Ch1
Ch2
Ch3
Counts
Time (ms)
0.2
0.15
0.1
0.05
0
-0.05
-0.1
-0.15
-0.2
0.9
1
1.1
1.2
1.3
1.4
1.5
1.6
<s> 10 01 10 00 01 00 00 00...
Persistence
Analog Alpha: 0.0994
Axes Options
Secs/Div:
Counts/Div:
Y Offset:
T Offset:
Autorange
Channel Options
Ch1
Ch2
Coupling:
AC
Marker:
None
Stop

Transfer Speed

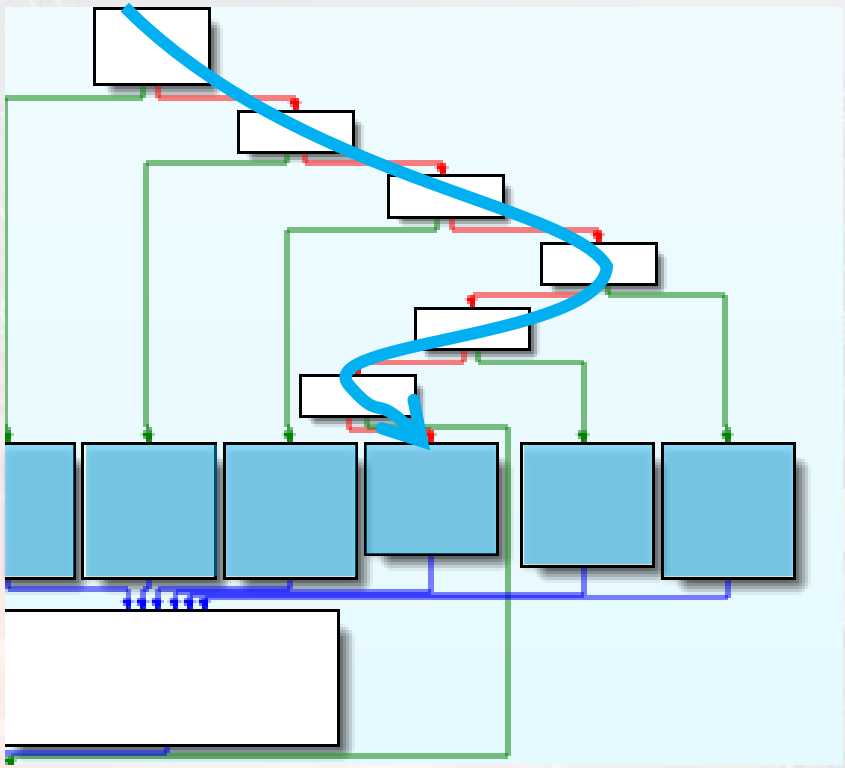
- ISO15693 has two modes:
  - Slow (1.65 kbps)
  - Fast (26.48 kbps)
- Nexus S uses slow mode
- ICLASS only uses fast mode



Inside Contactless (2004)

Problem: Transfer Speed

- Capability probably exists, but is unused.
- Find transmission code
- Loads settings from `EEPROM/CFG`
- Only uses one set of values
- Swap around values in `EEPROM/CFG`
- Fast mode!

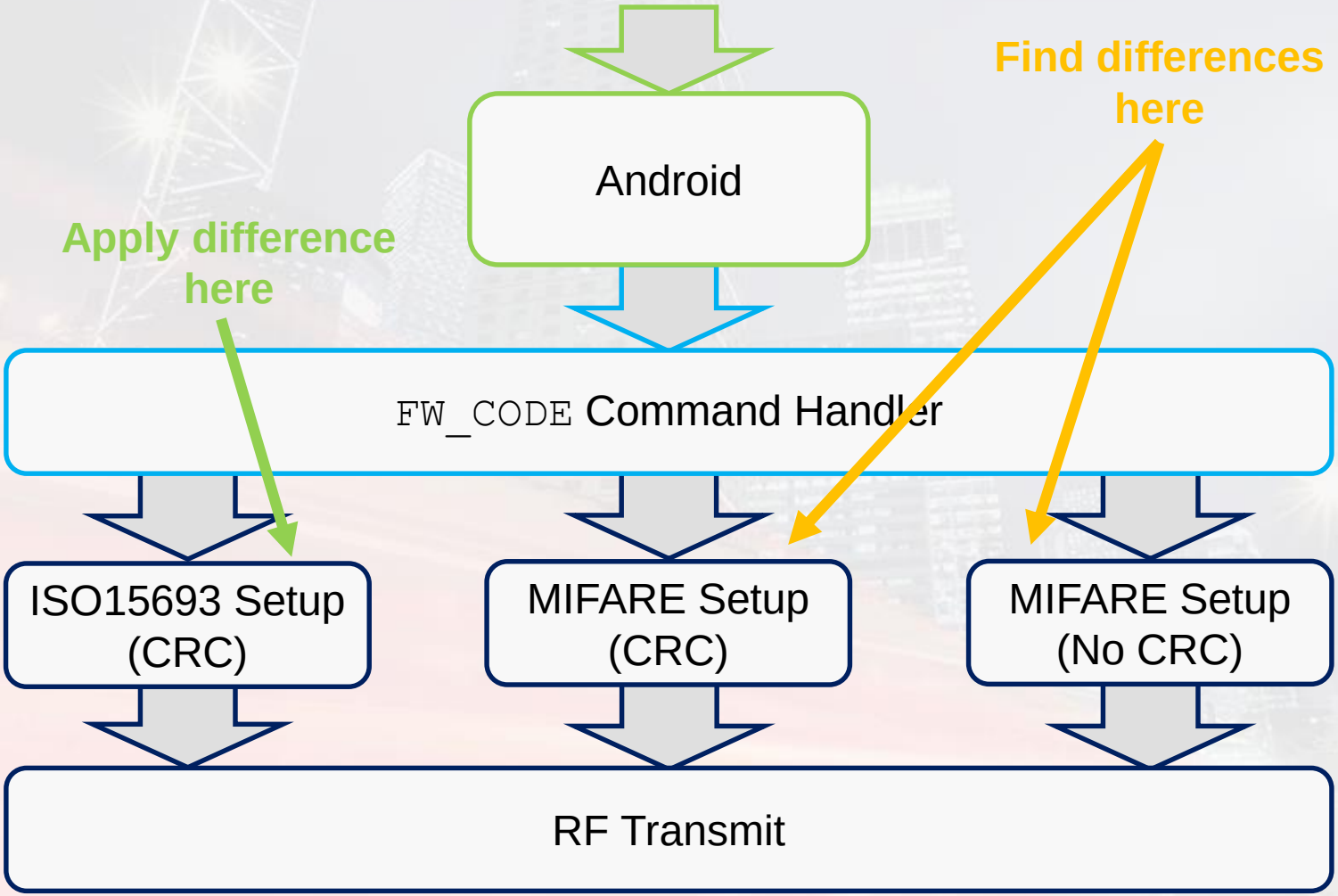


Mifare

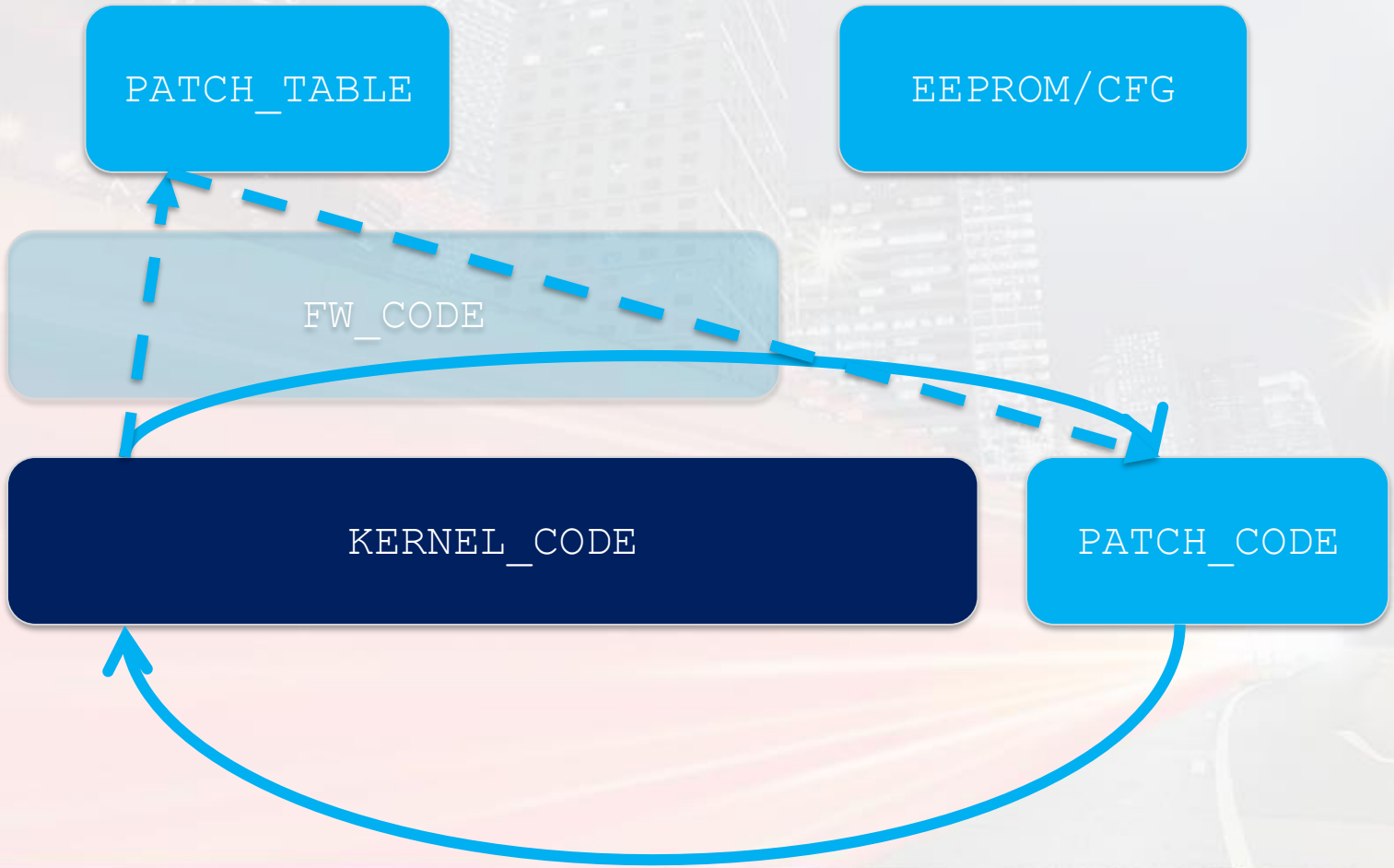
```
case NXP_MIFARE_RAW:
{
    if (p_pipe_info->param_length < RDR_A_MIFARE_RAW_LENGTH)
    {
        status = PHNFCSTVAL(CID_NFC_HCI,
                            NFCSTATUS_INVALID_PARAMETER);
    }
    else
    {
        /*
            Buffer shall be updated with
            TO -              Time out (1 byte)
            Status -          b0 to b2 indicate valid bits (1 byte)
            Data (with CRC) - params received from this function
        */
```

```
case NXP_MIFARE_CMD:
{
    /*
        Buffer shall be updated with
        Cmd -               Authentication A/B, read/write
                            (1 byte)
        Addr -              Address associated with Mifare cmd
                            (1 byte)
        Data  -             params received from this function
    */
```

Problem: Checksum Generation

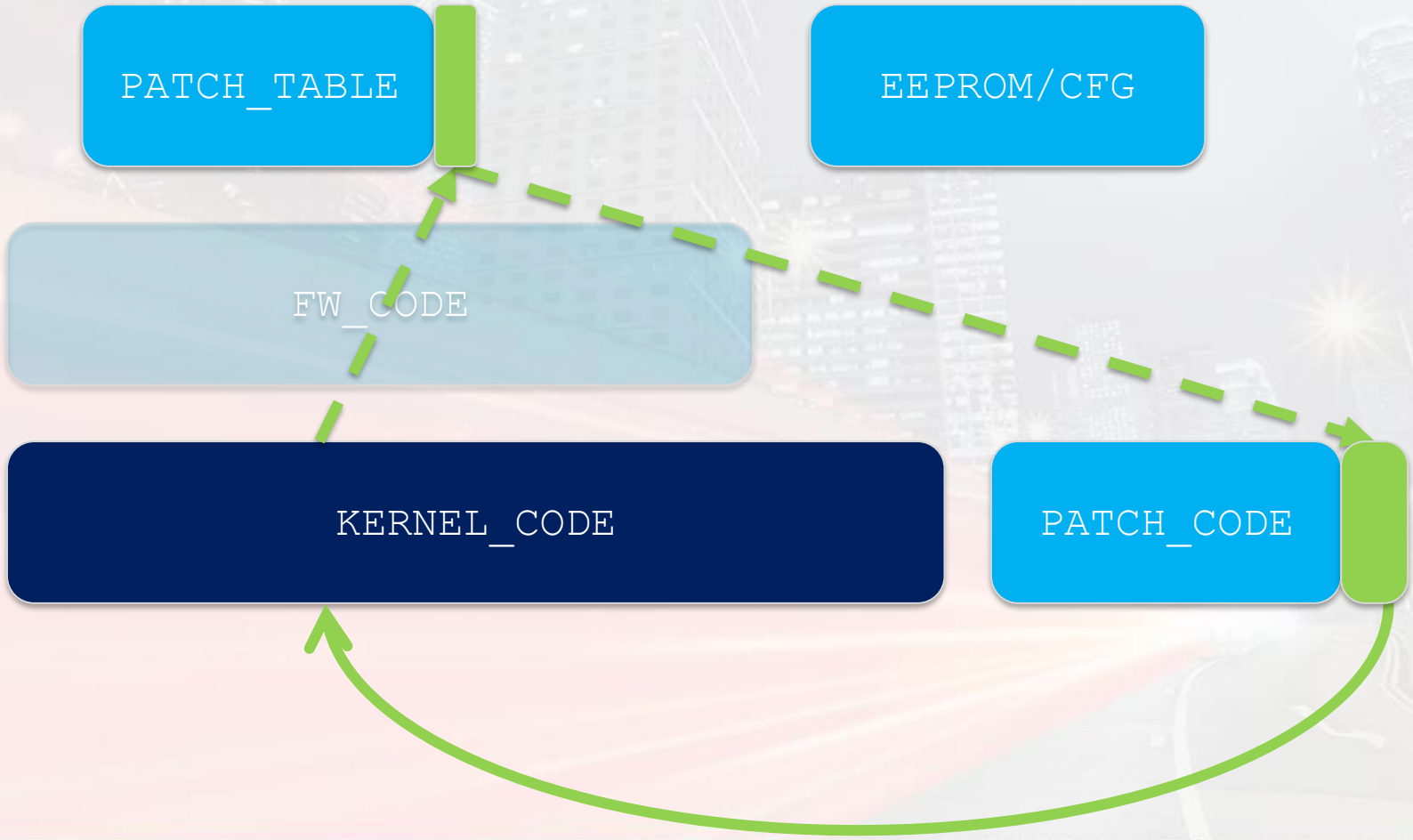


Patching the Kernel



Exploitation

Patching Checksum Generation



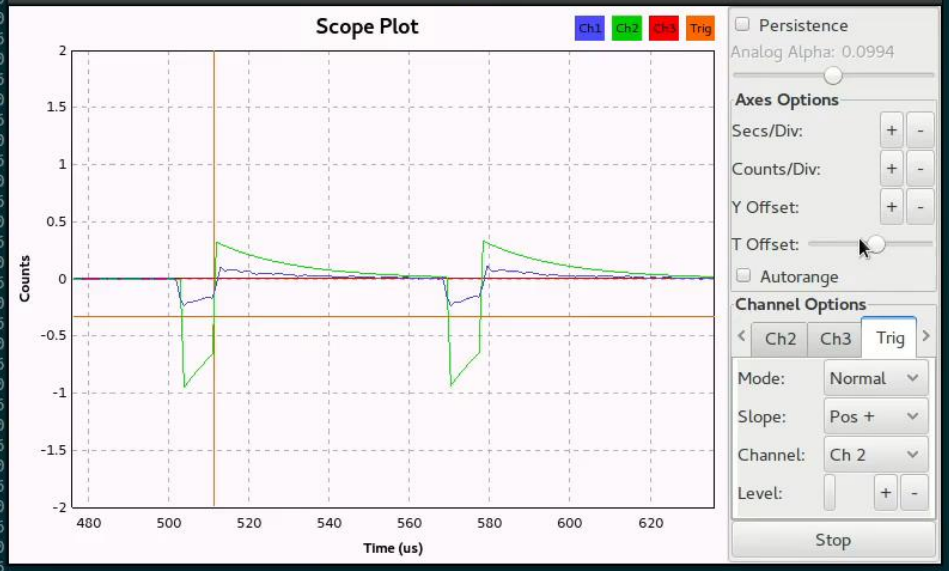
Activities Reader_sdr.py Tue 14:42

keegan@keegan-VirtualBox: ~/magic/decoder

File Edit View Search Terminal Help

```
1 in 256
26 01 00
1 in 256
26 01 00
1 in 256
26 01 00
1 in 256
26 01 00
1 in 256
26 01 00
1 in 256
26 01 00
1 in 256
26 01 00
1 in 256
26 01 00
1 in 256
26 01 00
1 in 256
26 01 00
1 in 256
26 01 00
1 in 256
26 01 00
1 in 256
26 01 00
1 in 256
26 01 00
1 in 256
26 01 00
1 in 256
26 01 00
1 in 256
26 01 00
1 in 256
26 01 00 f6 0a
1 in 256
26 01 00 f6 0a
1 in 256
26 01 00 f6 0a
1 in 256
26 01 00 f6 0a
1 in 256
26 01 00 f6 0a
1 in 256
26 01 00 f6 0a
```

Top Block



Persistence

Analog Alpha: 0.0994

Axes Options

Secs/Div: + -

Counts/Div: + -

Y Offset: + -

T Offset:

Autorange

Channel Options

Ch2 Ch3 Trig

Mode: Normal

Slope: Pos +

Channel: Ch 2

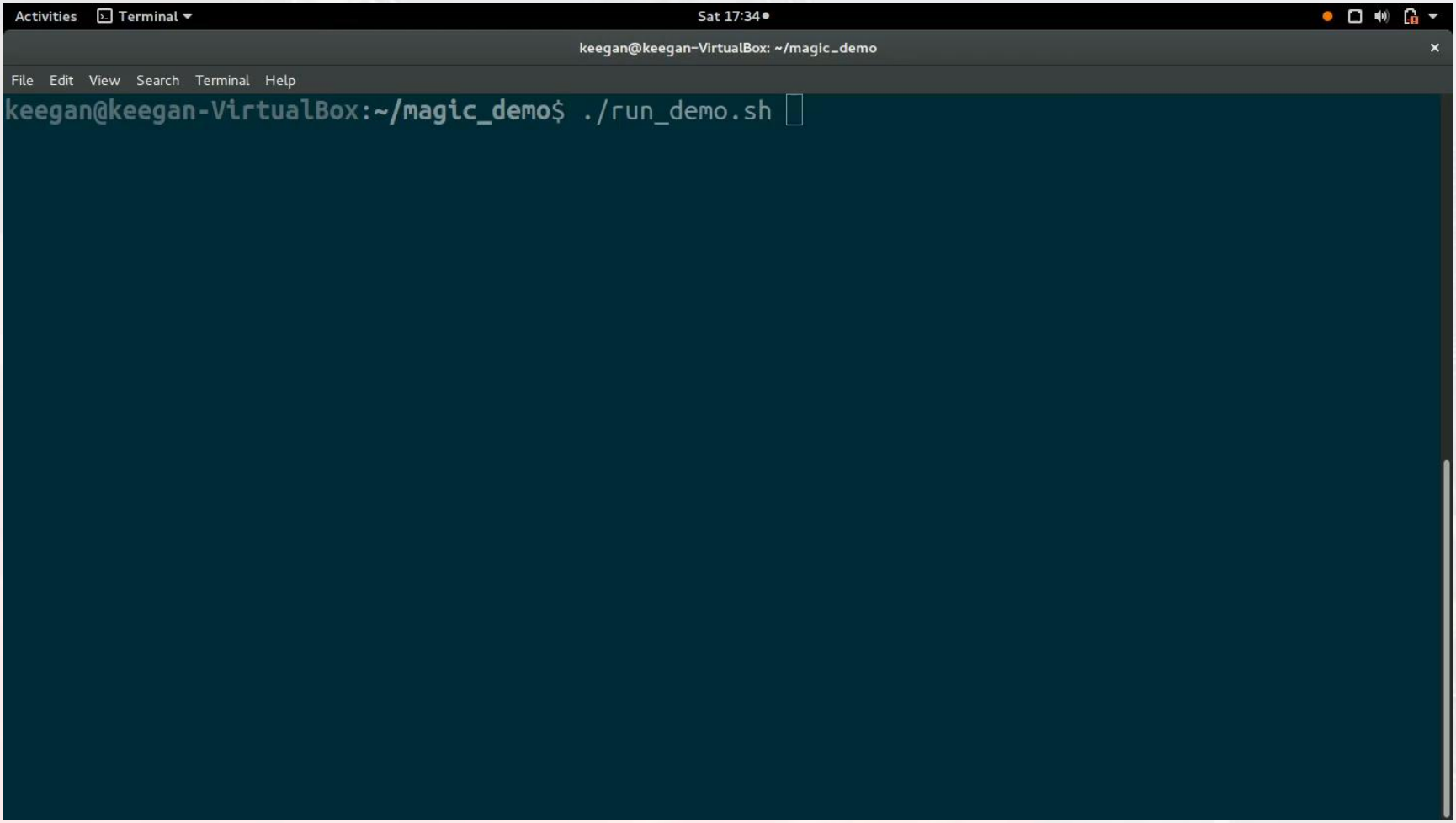
Level: + -

Stop

Putting It All Together


nccgroup
Demo

Demo


Activities
Terminal
Sat 17:34
keegan@keegan-VirtualBox: ~/magic_demo
File Edit View Search Terminal Help
keegan@keegan-VirtualBox:~/magic_demo$ ./run_demo.sh

Future Research

What can be done with a hacked NFC controller?

- Surreptitiously read a badge
- Information storage
- Information exfiltration

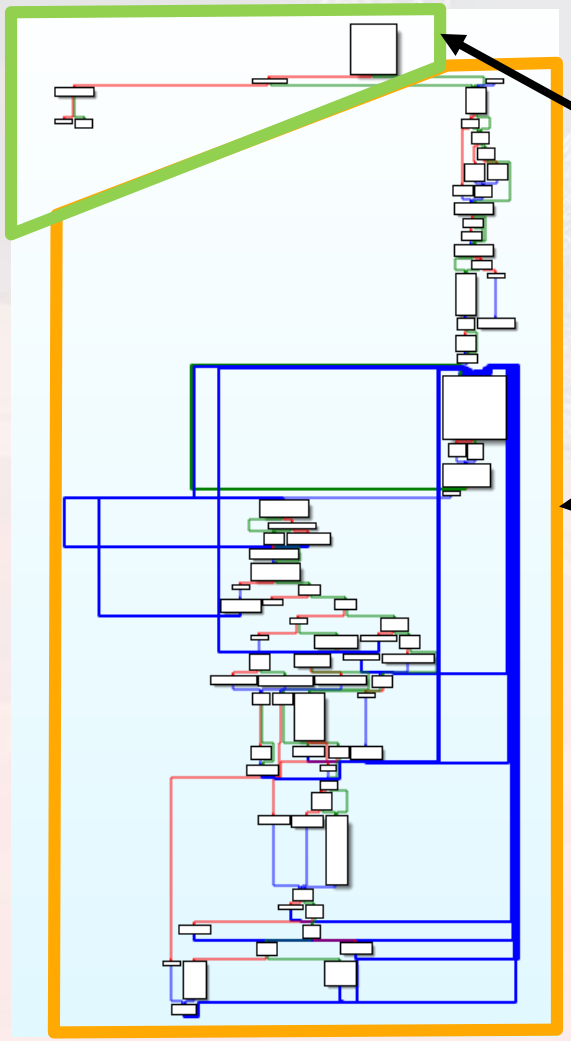
Future Research

- What other embedded systems do we carry everywhere?
  - Bluetooth
  - USB controller
  - Baseband radio
  - Camera
  - Fingerprint reader
- What could you make these systems do?

The End

Keegan Ryan
Keegan.Ryan@nccgroup.trust
@inf_0_

Bypassing Firmware Signing?



```
if (*flag == 0xa55a)
    doInsecureDownload();
else
    doSecureDownload();
```

Bibliography

Brendangates. “Badge reader.” Licensed under a Creative Commons Attribution 2.0 Generic (CC BY-NC-ND 2.0). Accessed 11 June 2017. https://www.flickr.com/photos/brendangates/2384518688.

Churchill, Sam. “nfc.phone.” Licensed under a Creative Commons Attribution 2.0 Generic (CC BY 2.0). Accessed 11 June 2017. https://www.flickr.com/photos/samchurchill/5181496553

Inside Contactless. "Datasheet PicoPass 2KS." *Rapport technique* (2004).

Libnfc-nxp Library. Accessed June 11, 2017. https://android.googlesource.com/platform/external/libnfc-nxp.

Meriac, Milosch. "Heart of darkness-exploring the uncharted backwaters of hid iclass (TM) security." In *27th Chaos Communication Congress*. 2010.

NXP. “NXP NFC controller PN544 for mobile phones and portable equipment." *On Line: http://www.nxp.com/documents/leaflet/75016890.pdf* (2010).

Wharton, John. "An Introduction to the Intel-MCS-51 Single-Chip Microcomputer Family." *Intel Corporation* (1980).