Man in the NFC

Build a NFC proxy tool from sketch

Haoqi Shan @ UnicornTeam

Agenda

- Who we are
- NFC & ISO14443A
- Competitions
- Yet anther wheel?
- What is UniProxy?
- Master and Slave
- Issues in development
- Thanks, Q&A

Who we are

- Unicorn team
  - Internal security research team of Qihoo 360, founded in 2014
  - Focus on wireless/hardware hacking and defense
  - Security research division and hardware development division
  - Serial wireless researches published in Defcon/BlackHat
    - Low-cost GPS spoofing, Defcon 23
    - LTE redirection attack, Defcon 24
    - Attack on powerline communication, BlackHat USA 2016
    - `Ghost Telephonist', Defcon 25/BlackHat USA 2017
  - Serial hacking tools developed
    - HackID/HackID Pro/SafeRFID/HackNFC, etc
    - https://unicorn.360.com

NFC & ISO14443A

- NFC
  - 13.56MHz
  - Low-cost
  - Not requires power
  - Well developed and deployed
- ISO14443A
  - Widely usage
  - Supports many applications
  - Security/Passport/BankCard

What we aim

- Credit card
  - QuickPass – Unipay (*)
- Starbucks POS machine
- XX: "I thought this question has been solved like a thousand times"
- More like a hacker

The way we used to hack

- Targeting protocols
  - Proxmark III (The Best Hardware way)
- Targeting data
  - NFCProxy
  - NFCGate

Why not?

- Proxmark III
  - Supports many protocols
  - Powerful
  - However, can't hack credit card or we are all rich now
- NFCGate/NFCProxy
  - Based on Android
  - Modified firmware to relay NFC data
  - Monitor transmitted data
  - Rely on Wi-Fi
  - However, too much delay to complete whole payment procedure

Yet another wheel

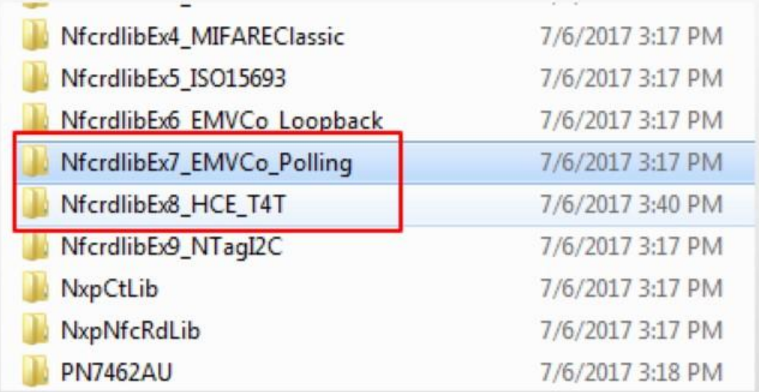
- Inspired by mentioned brilliant hacking tool
- Faster (ms level)
- Lager ranger (50m, even more)
- Pure hardware solution (PN7462AU)
- Highly customization
- Completely self-designed and modify everything we need

What’s UniProxy

- PN7462AU based NFC relay/proxy device
- Support ISO14443A protocol
- Targeting QuickPass(Unipay) credit cards
- Reader emulator, card emulator
- Point to Point wireless data transmission
- Easy to adapt to ISO 14443B/15693

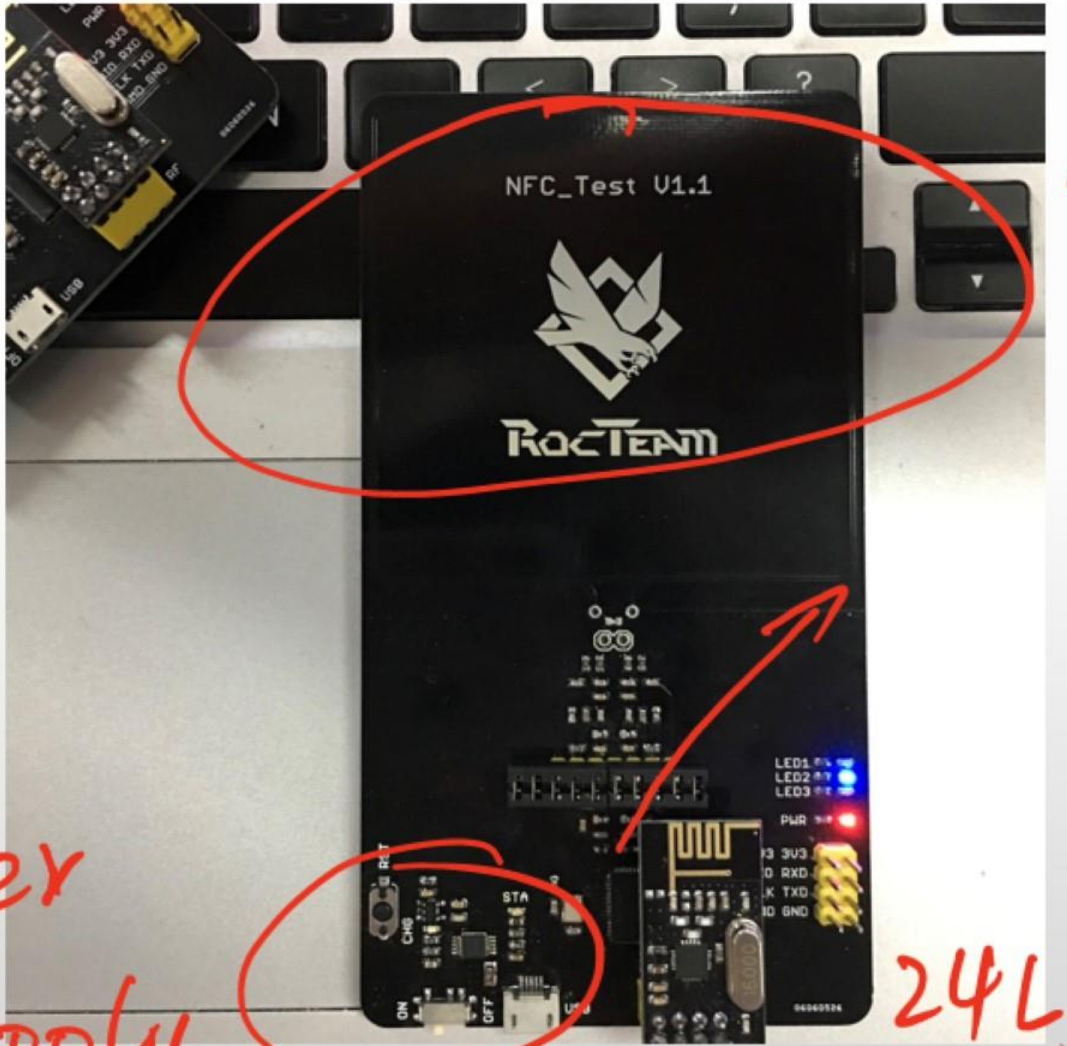
Core of UniProxy

- Why PN7462AU?
  - NXP chip
  - 20 MHz Cortex-M0 core
  - Read/Write, Card Emulation & Peer-to-Peer Modes
  - Transmitter current up to 250 mA
  - Full MIFARE family support
- Architecture
  - Reader/Card Emulator
  - NRF24L01 wireless transmitter
  - Power supply
  - Antenna



NfcrdlibEx4_MIFAREClassic	7/6/2017 3:17 PM
NfcrdlibEx5_ISO15693	7/6/2017 3:17 PM
NfcrdlibEx6_EMVCo_Loopback	7/6/2017 3:17 PM
NfcrdlibEx7_EMVCo_Polling	7/6/2017 3:17 PM
NfcrdlibEx8_HCE_T4T	7/6/2017 3:40 PM
NfcrdlibEx9_NTagI2C	7/6/2017 3:17 PM
NxpCtLib	7/6/2017 3:17 PM
NxpNfcRdLib	7/6/2017 3:17 PM
PN7462AU	7/6/2017 3:18 PM

Master (Front)



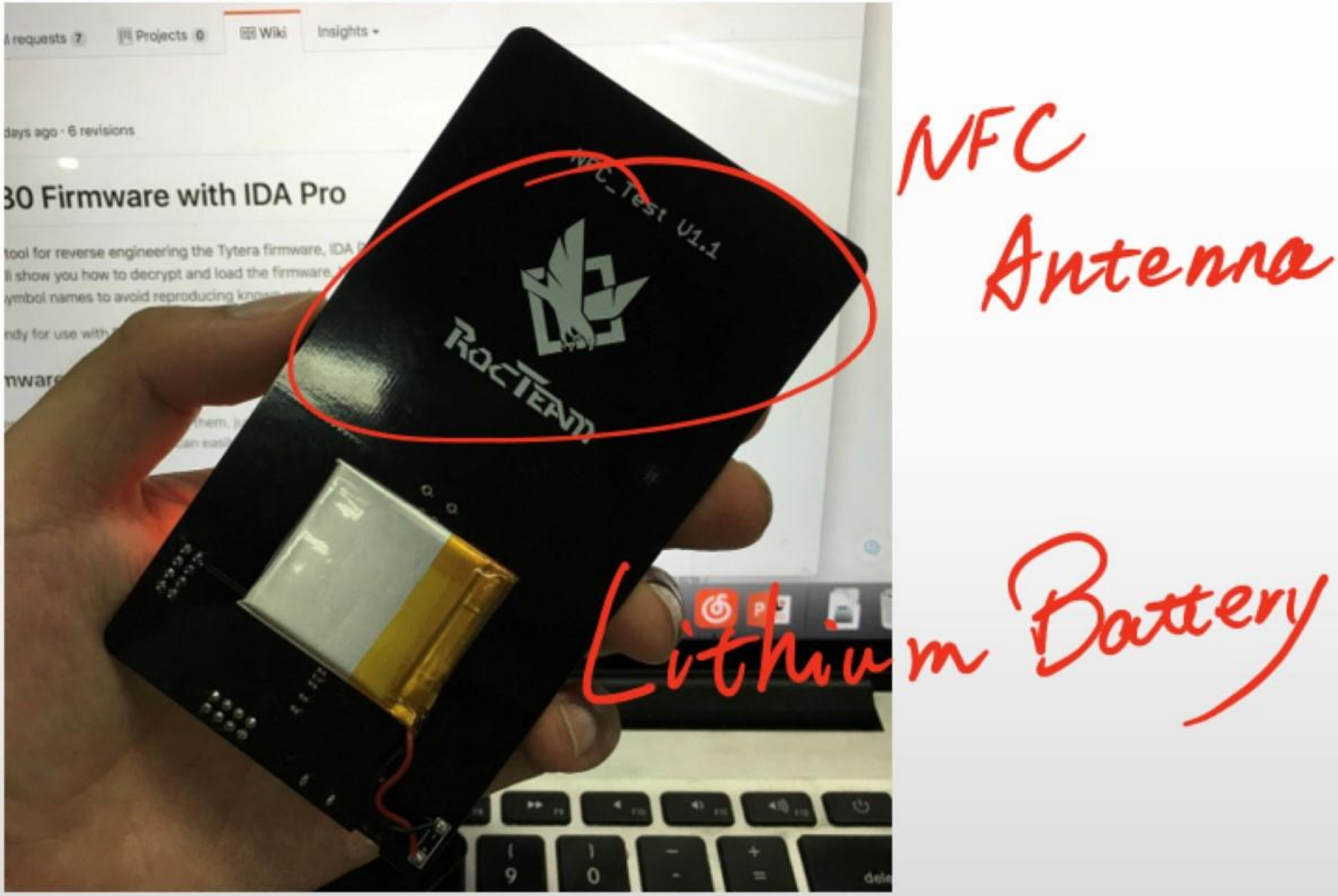
NFC Antenna

chip

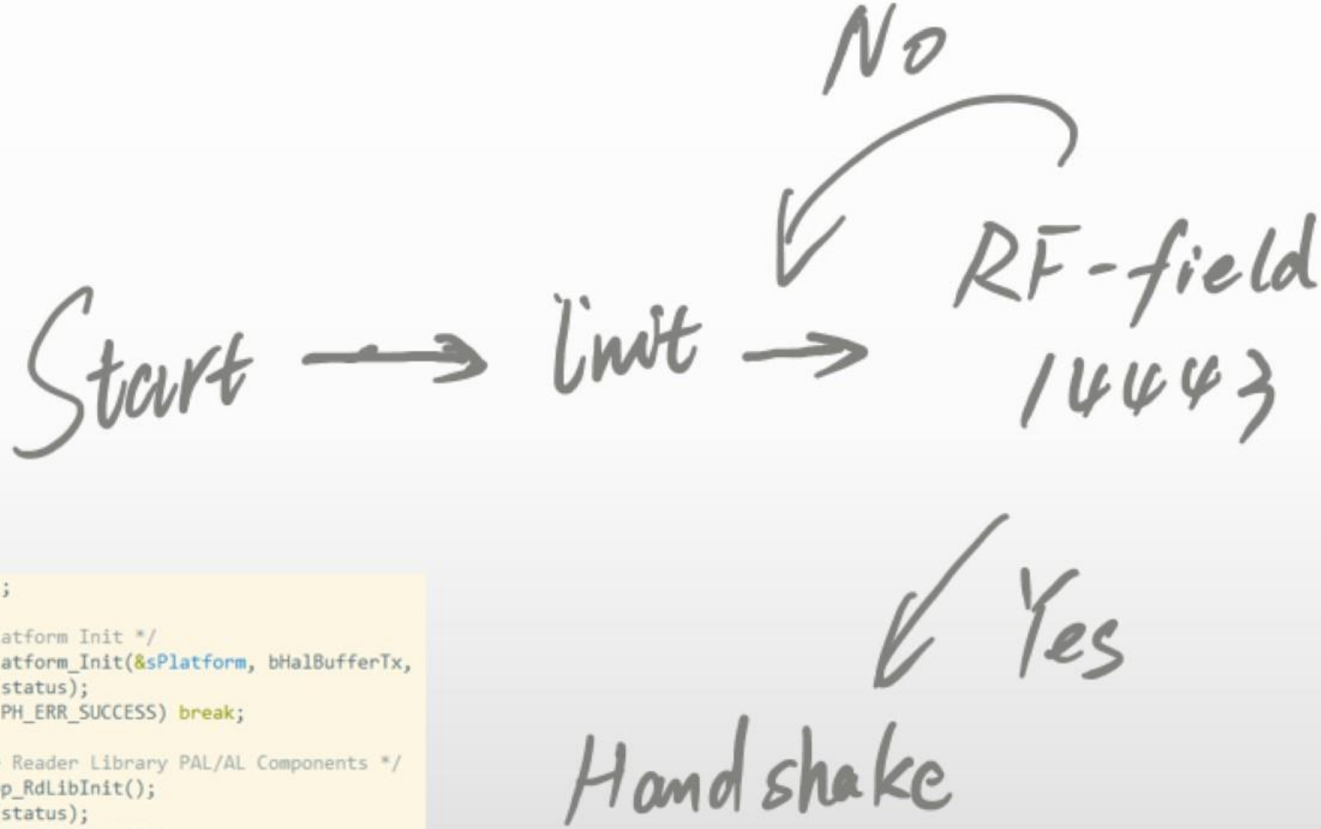
Power Supply

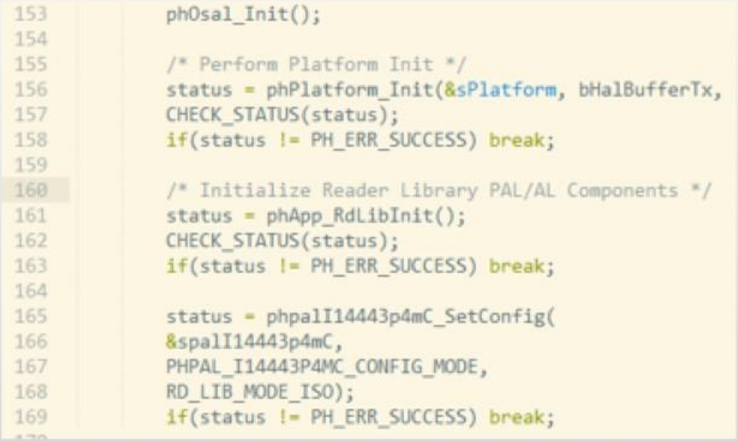
24L01

Master (Back)



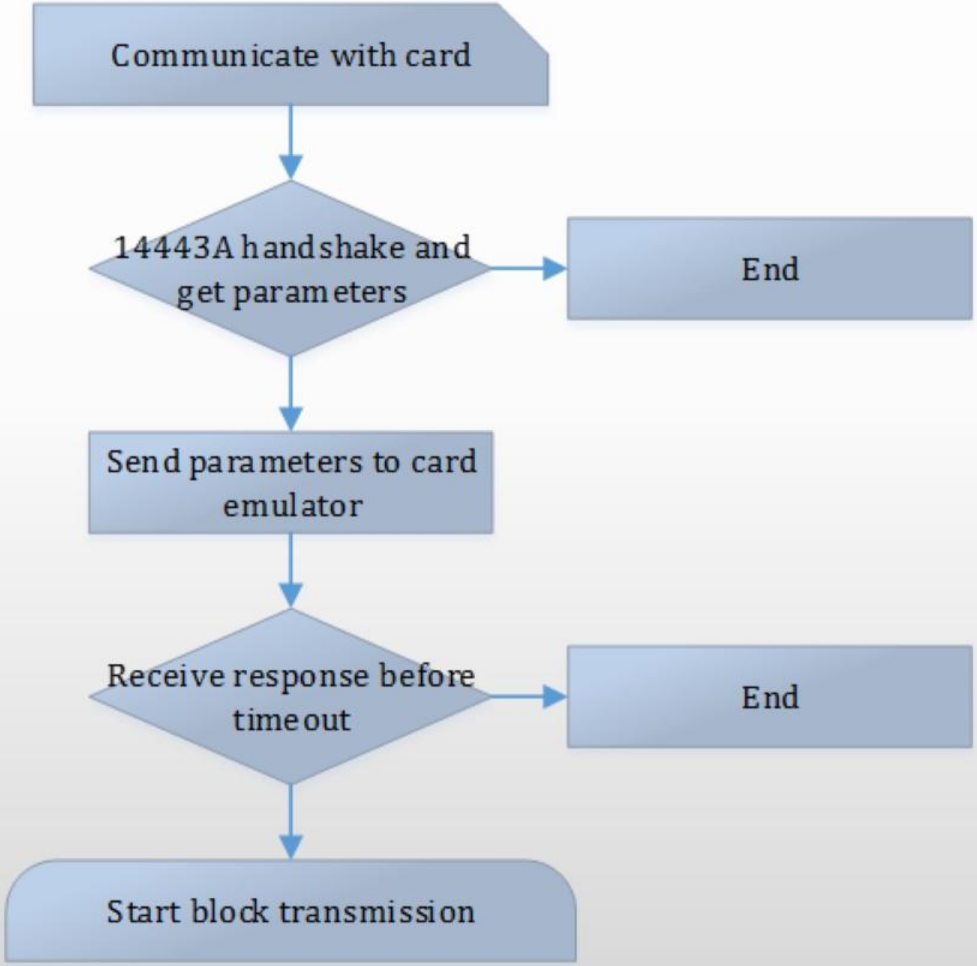
Process of Master (1)

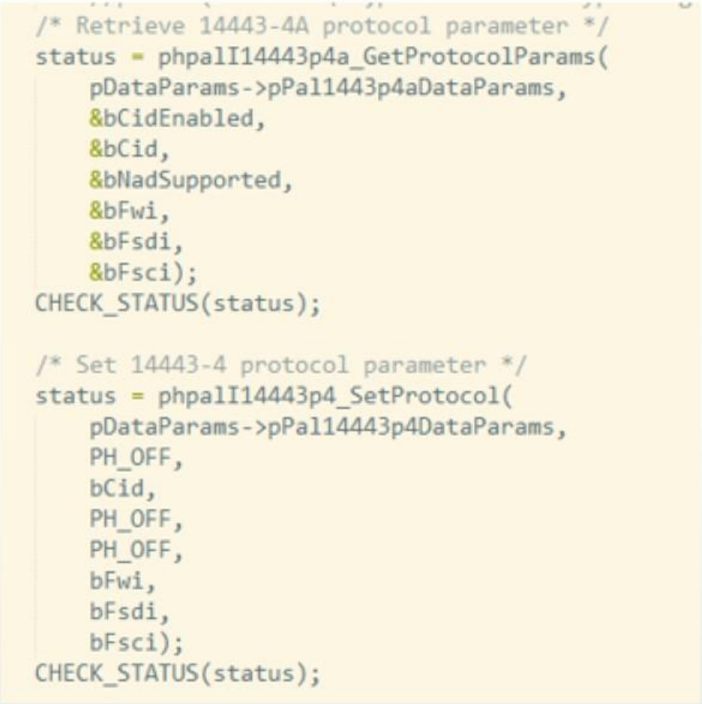




```
        phOsal_Init();

        /* Perform Platform Init */
        status = phPlatform_Init(&sPlatform, bHalBufferTx,
        CHECK_STATUS(status);
        if(status != PH_ERR_SUCCESS) break;

        /* Initialize Reader Library PAL/AL Components */
        status = phApp_RdLibInit();
        CHECK_STATUS(status);
        if(status != PH_ERR_SUCCESS) break;

        status = phpalI14443p4mC_SetConfig(
        &spalI14443p4mC,
        PHPAL_I14443P4MC_CONFIG_MODE,
        RD_LIB_MODE_ISO);
        if(status != PH_ERR_SUCCESS) break;
```

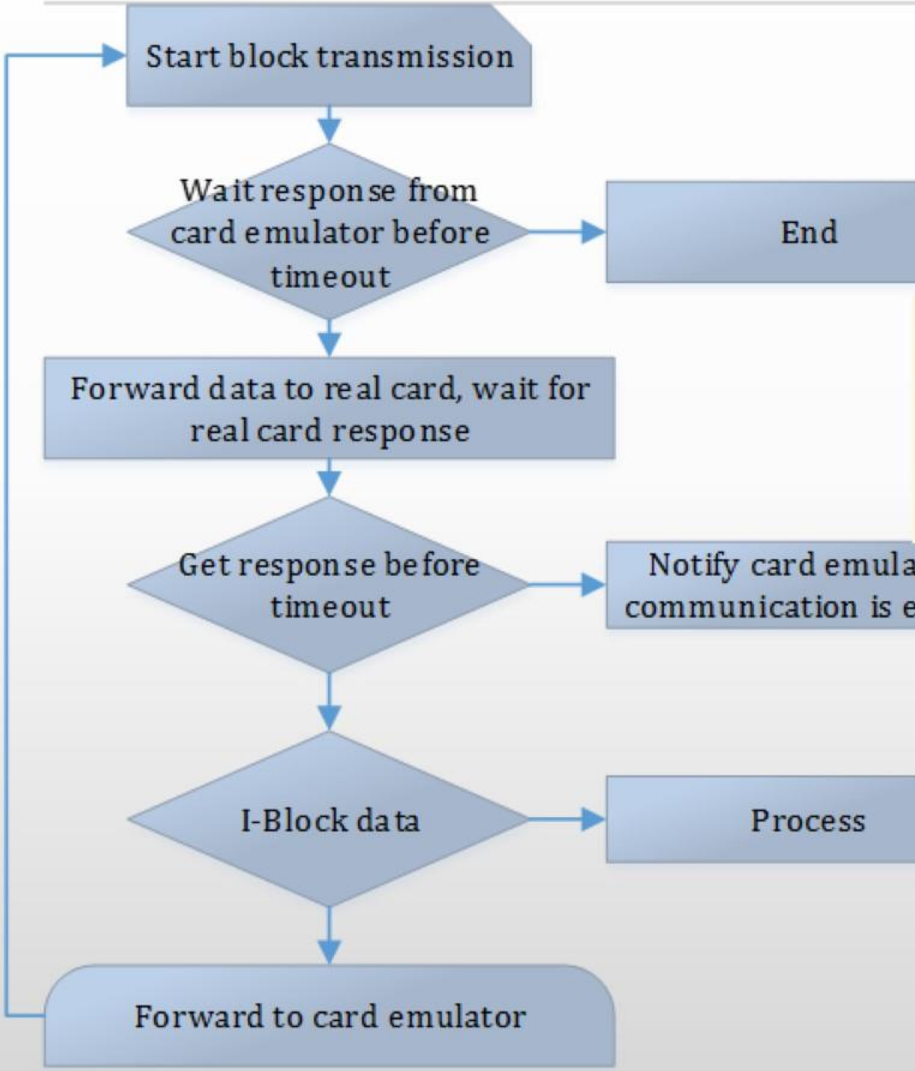
Process of Master (2)


Communicate with card
14443A handshake and get parameters
End
Send parameters to card emulator
Receive response before timeout
End
Start block transmission



```
/* Retrieve 14443-4A protocol parameter */
status = phpalI14443p4a_GetProtocolParams(
    pDataParams->pPal1443p4aDataParams,
    &bCidEnabled,
    &bCid,
    &bNadSupported,
    &bFwi,
    &bFsdi,
    &bFsci);
CHECK_STATUS(status);

/* Set 14443-4 protocol parameter */
status = phpalI14443p4_SetProtocol(
    pDataParams->pPal14443p4DataParams,
    PH_OFF,
    bCid,
    PH_OFF,
    PH_OFF,
    bFwi,
    bFsdi,
    bFsci);
CHECK_STATUS(status);
```

Process of Master (3)



Start block transmission

Wait response from card emulator before timeout

End

Forward data to real card, wait for real card response

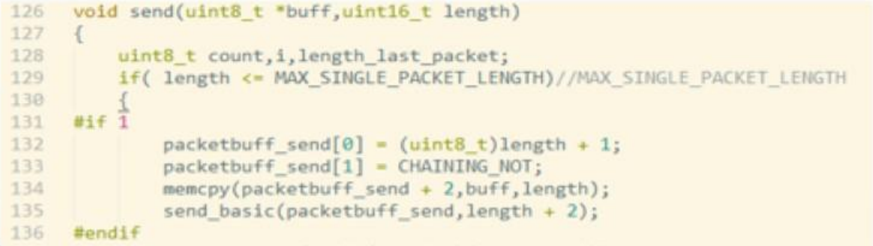
Get response before timeout

Notify card emulator, communication is ended

I-Block data

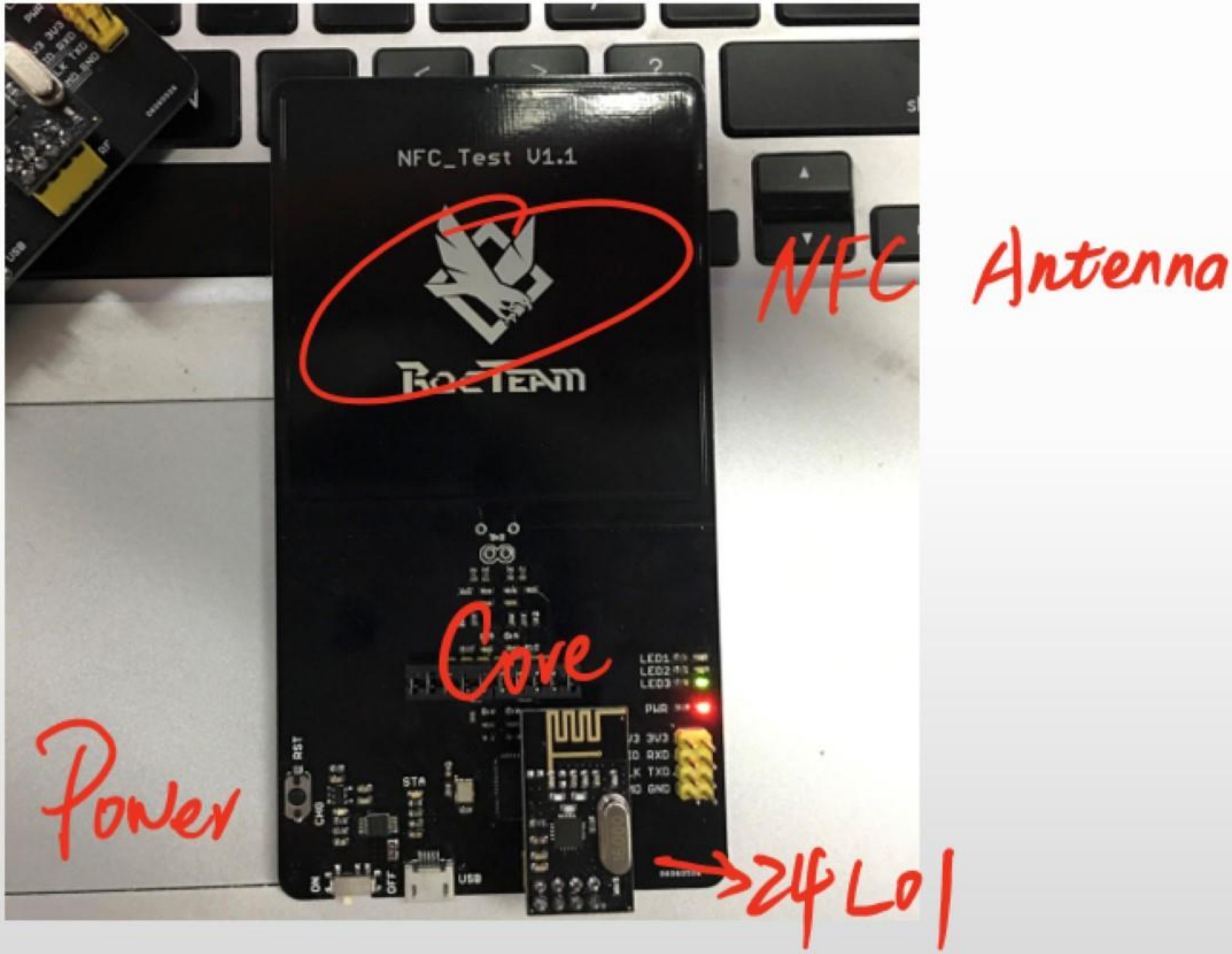
Process

Forward to card emulator

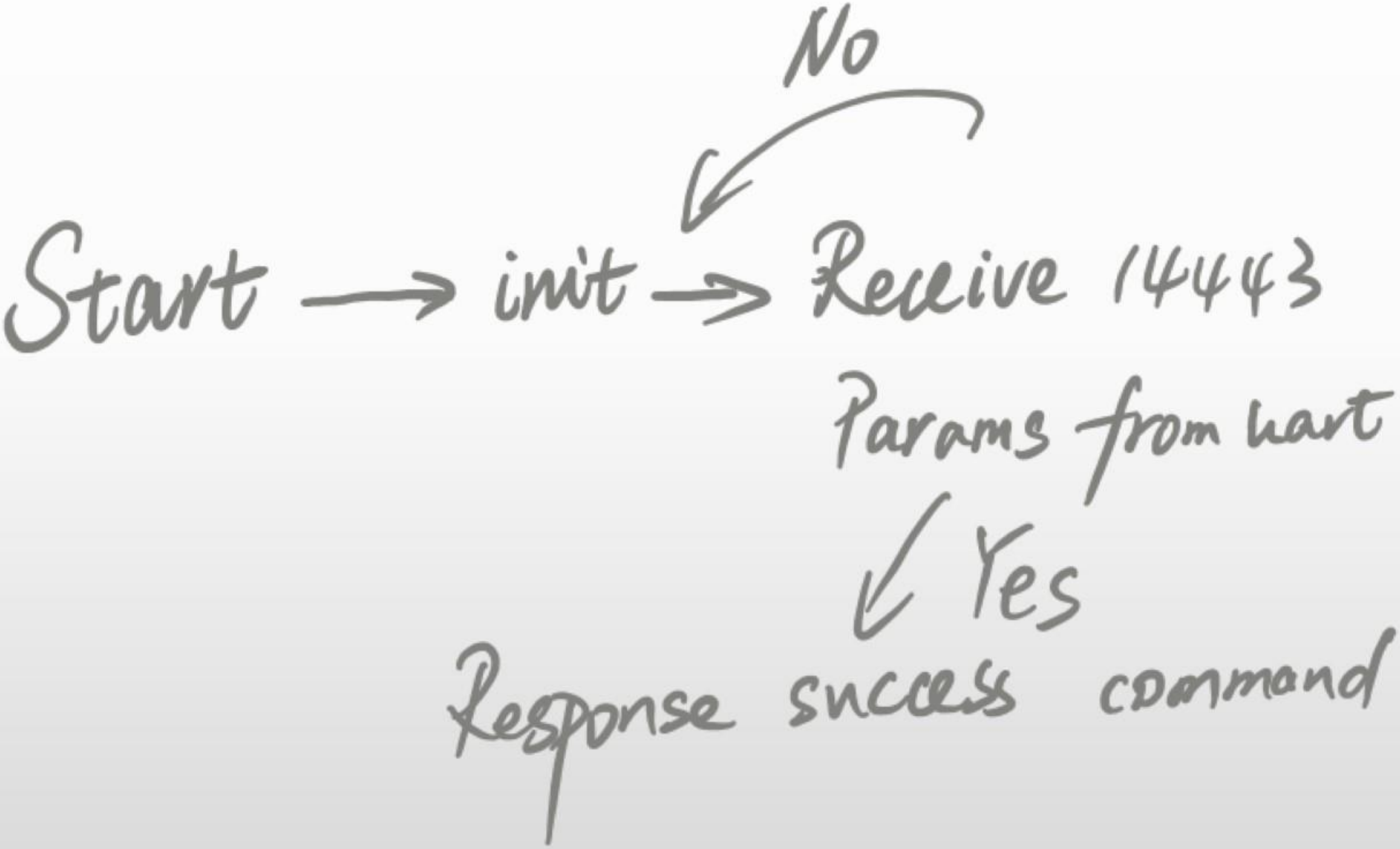


```
void send(uint8_t *buff,uint16_t length)
{
    uint8_t count,i,length_last_packet;
    if( length <= MAX_SINGLE_PACKET_LENGTH)//MAX_SINGLE_PACKET_LENGTH
    {
#if 1
        packetbuff_send[0] = (uint8_t)length + 1;
        packetbuff_send[1] = CHAINING_NOT;
        memcpy(packetbuff_send + 2,buff,length);
        send_basic(packetbuff_send,length + 2);
#endif
```

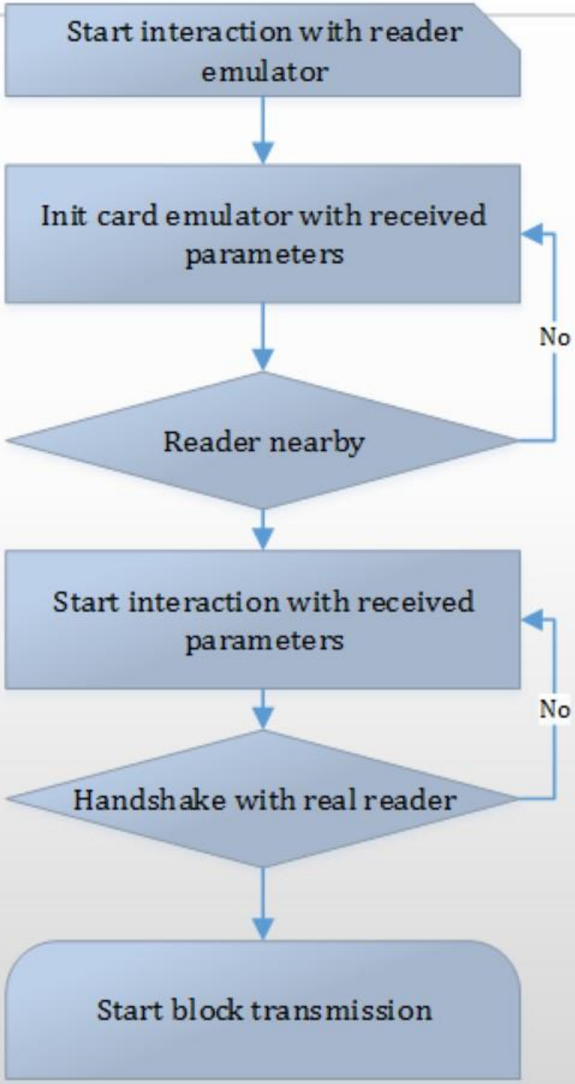
Slave

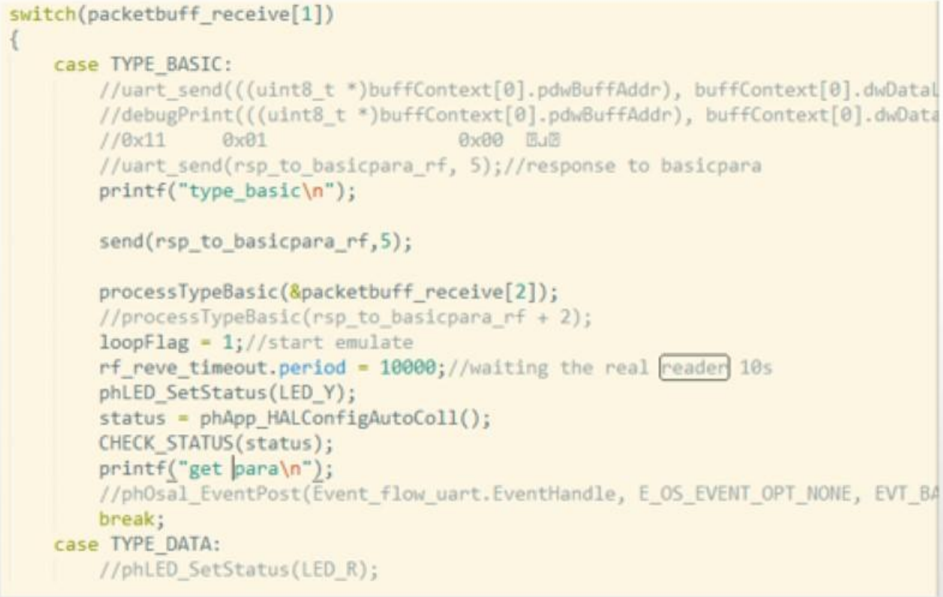


Process of Slave (1)



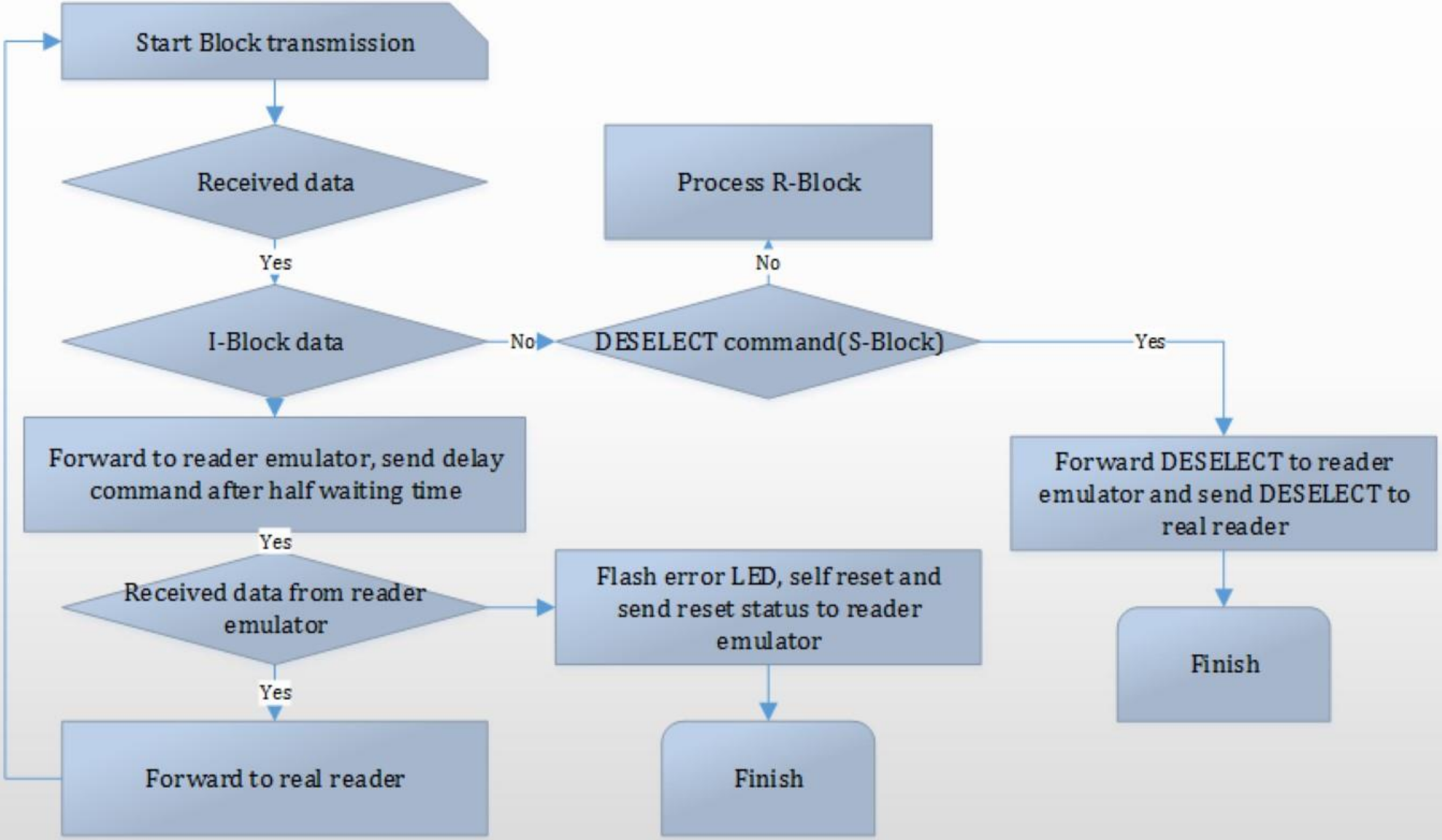
Process of Slave (2)


Start interaction with reader emulator
Init card emulator with received parameters
Reader nearby
No
Start interaction with received parameters
Handshake with real reader
No
Start block transmission



```
switch(packetbuff_receive[1])
{
    case TYPE_BASIC:
        //uart_send(((uint8_t *)buffContext[0].pdwBuffAddr), buffContext[0].dwDataL
        //debugPrint(((uint8_t *)buffContext[0].pdwBuffAddr), buffContext[0].dwData
        //0x11     0x01                    0x00  □」□
        //uart_send(rsp_to_basicpara_rf, 5);//response to basicpara
        printf("type_basic\n");

        send(rsp_to_basicpara_rf,5);

        processTypeBasic(&packetbuff_receive[2]);
        //processTypeBasic(rsp_to_basicpara_rf + 2);
        loopFlag = 1;//start emulate
        rf_reve_timeout.period = 10000;//waiting the real reader 10s
        phLED_SetStatus(LED_Y);
        status = phApp_HALConfigAutoColl();
        CHECK_STATUS(status);
        printf("get para\n");
        //phOsal_EventPost(Event_flow_uart.EventHandle, E_OS_EVENT_OPT_NONE, EVT_BA
        break;
    case TYPE_DATA:
        //phLED_SetStatus(LED_R);
```

Process of Slave (3)


Start Block transmission
Received data
Yes
I-Block data
No
DESELECT command(S-Block)
No
Process R-Block
Yes
Forward to reader emulator, send delay command after half waiting time
Yes
Received data from reader emulator
Flash error LED, self reset and send reset status to reader emulator
Yes
Forward to real reader
Finish
Forward DESELECT to reader emulator and send DESELECT to real reader
Finish

Issues in development

- First byte of UID
- Waiting/Wakeup time
- I/S/R – Block data
- ISO 14443A Part 4
- Power supply
- …

Demo video

Summary

- What we learned
  - Read protocol document well
  - Better not developing without official support
- Further more
  - Improve transmission range up to 100 meters
  - Targeting security ID cards, HID iClass, Chinese ID
  - Self-compatibility
  - How?

References

- [NXP user guide](http://www.nxp.com/docs/en/user-guide/UM10883.pdf)
- [NFC Gate](https://github.com/nfcgate)
- [NFC Proxy](http://sourceforge.net/projects/nfcproxy)
- [ISO14443A](https://www.iso.org/standard/70172.html)

Thanks

- Hardware dev division of Unicorn Team, especially Jian Yuan, Chaoran Wang, and Yunding Jian
- Proxmark III
- NFCProxy
- NFCGate

Q&A

- Mail me: shanhaoqi@360.cn