



임베디드 시스템 취약점 분석 (홈네트워크 사례 중심)

cybermong@grayhash.com

2015.4.23

NETSEC-KR 2015

목차

- 발표자 소개
- 홈 네트워크 연구 계기
- 홈 네트워크 시스템의 구조
- 펌웨어(소프트웨어) 획득
- 취약점 공격 과정 설명
- 취약점 데모
 - 월패드 쉘 획득
 - 전등 제어
 - 현관/로비 도어락 제어
 - 화상카메라 감시
- 대응방안

발표자 소개

- 정구홍(멍멍, 몽이)
- GrayHash(grayhash.com) 수석 연구원
- 해커스쿨(hackerschool.org) 운영자
- 각종 해킹대회(codegate, secuinside) 운영
- Defcon CTF 본선 다수 진출
- 연락처
 - cybermong@grayhash.com
 - <http://facebook.com/goohong.jung>

홈 네트워크 해킹 계기



홈 네트워크 해킹 계기



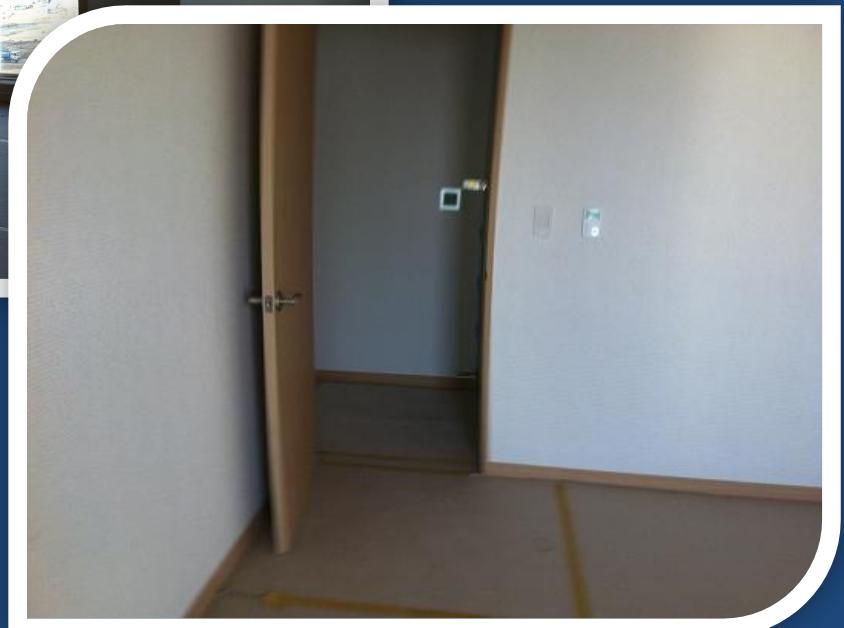
홈 네트워크 해킹 계기



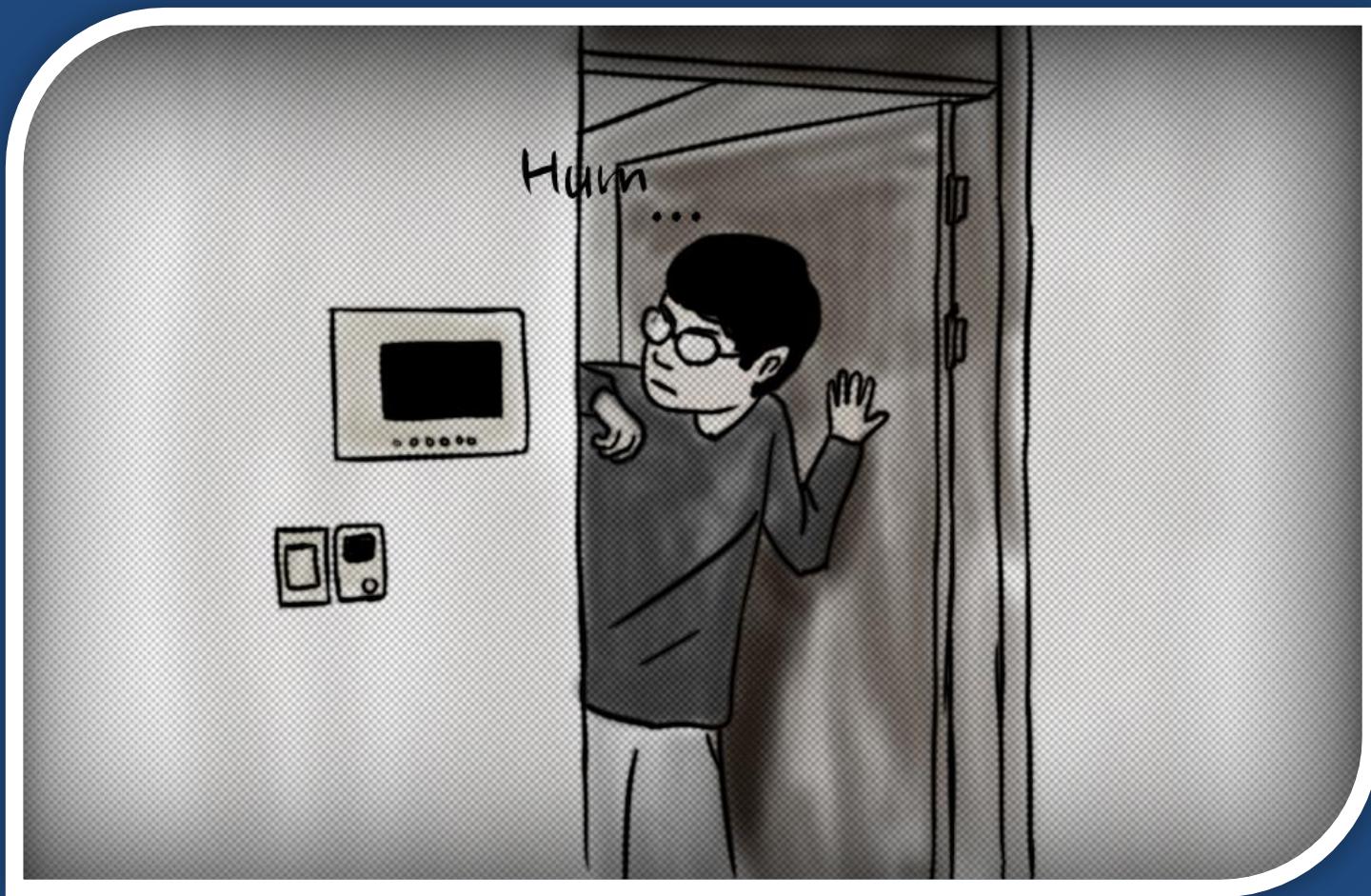
홈 네트워크 해킹 계기



홈 네트워크 해킹 계기



하지만 내 눈에 들어온 것은..



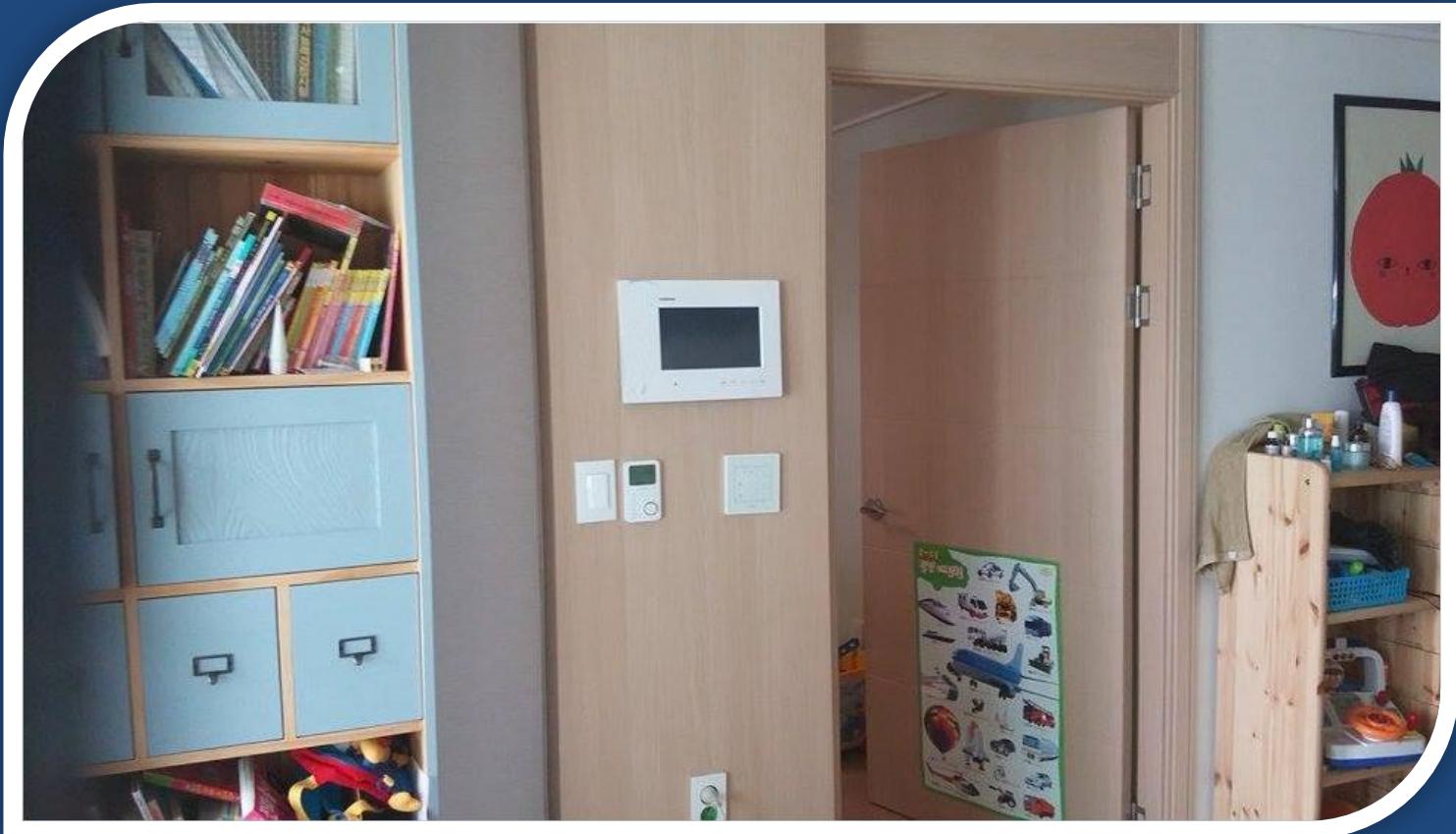
하지만 내 눈에 들어온 것은..



홈 네트워크 해킹 계기



월패드 공략 절차



월패드 공략 절차

- Step1 : 홈 네트워크 시스템의 구조 파악
- Step2 : 공격 대상 선정(wallpad)
- Step3 : wallpad 펌웨어(소프트웨어) 획득
- Step4 : wallpad 분해
- Step5 : UART 연결
- Step6 : 취약점 분석
- Step7 : 공격(Exploitation) 진행

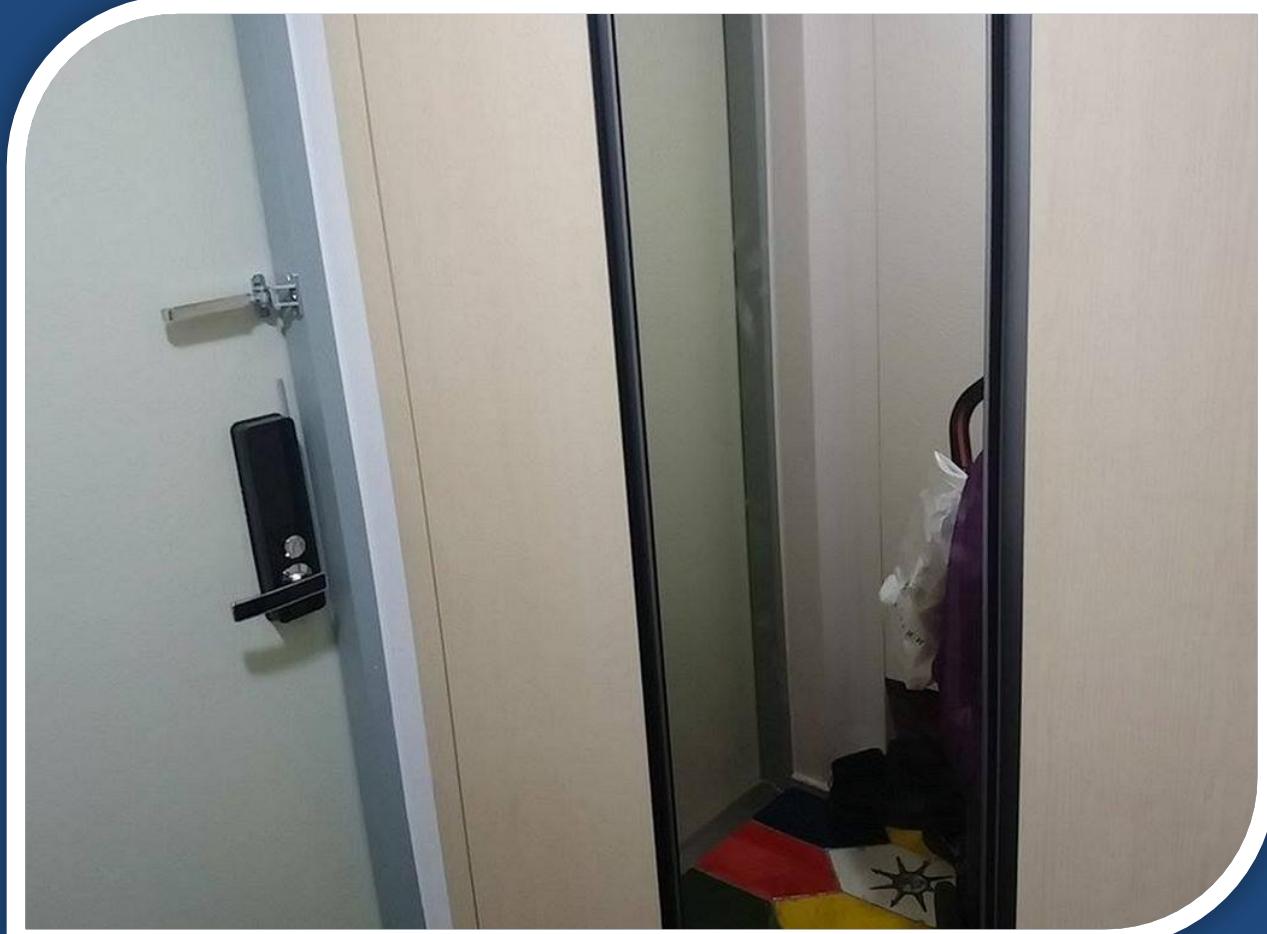
Step1

- 홈 네트워크 시스템의 구조 파악

현관 : 중앙 컨트롤러(gateway)



현관 : 중앙 컨트롤러(gateway)

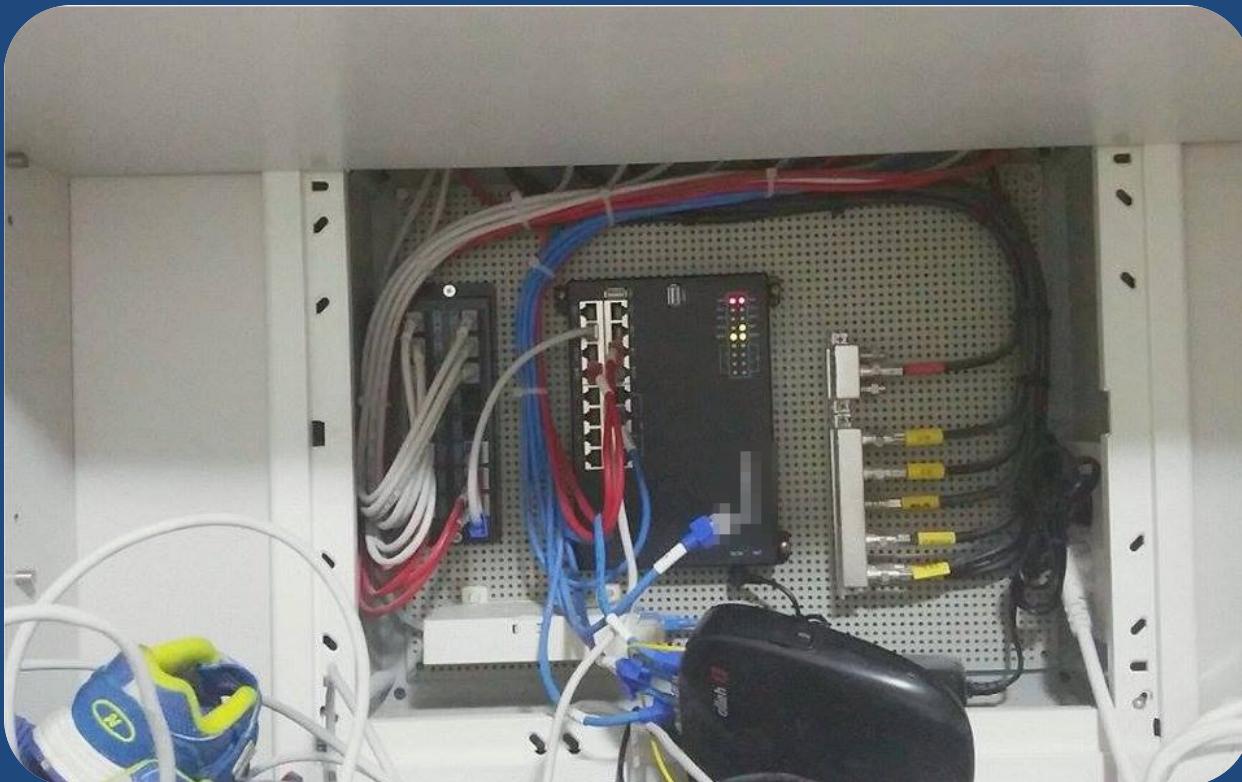


현관 : 중앙 컨트롤러(gateway)



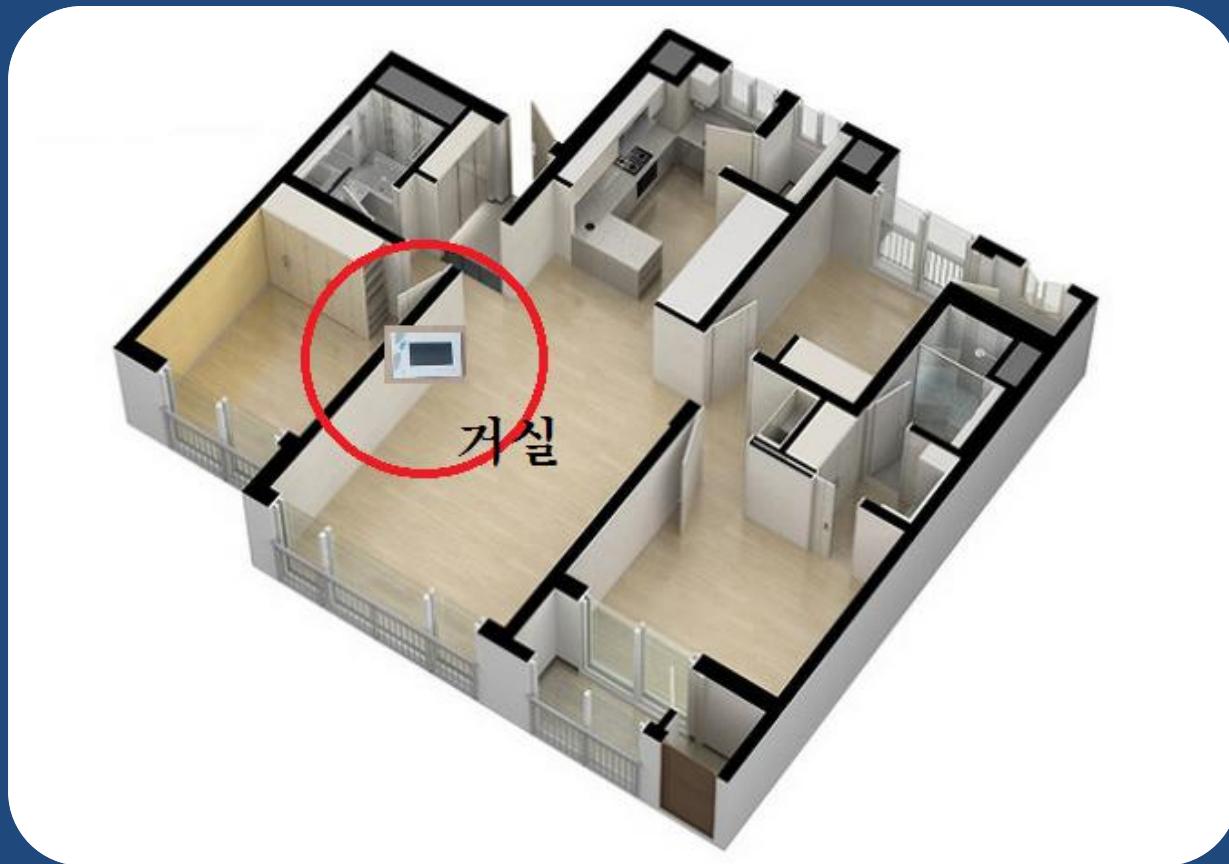
현관 : 중앙 컨트롤러(gateway)

- OS : Embedded Linux



거실 : Wallpad (사용자 인터페이스)

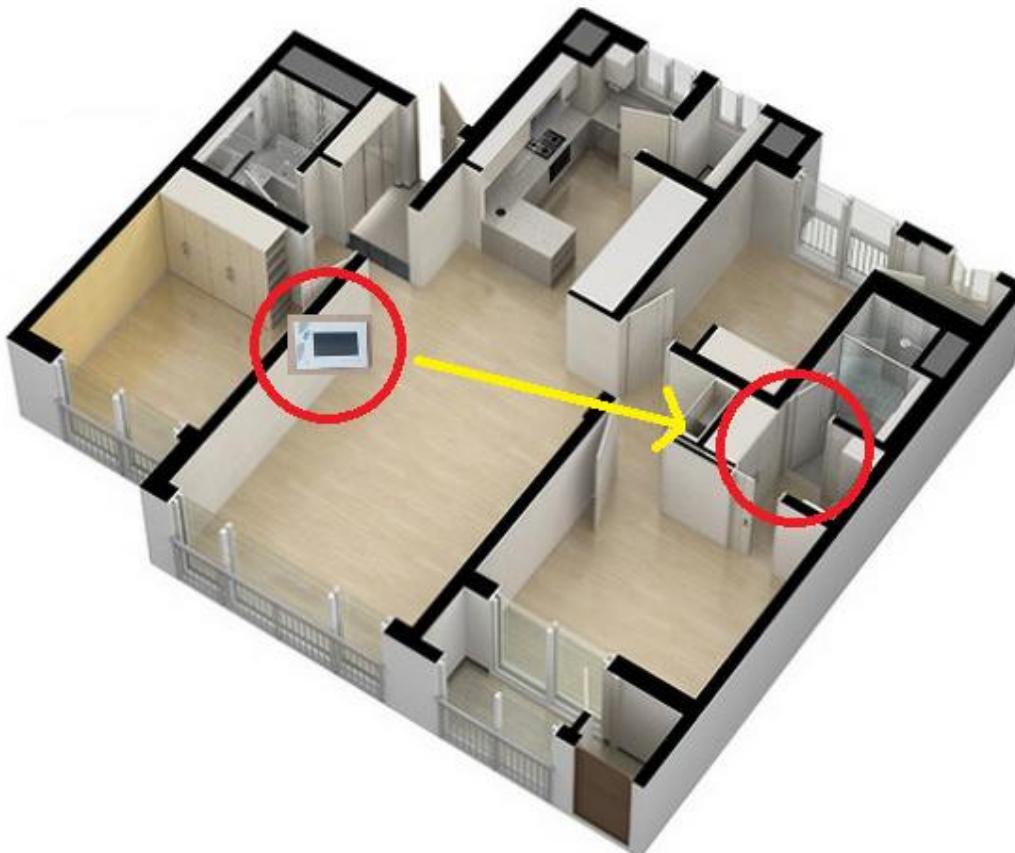
- OS : Linux (개조된 Android 2.3)



거실 : Wallpad (사용자 인터페이스)

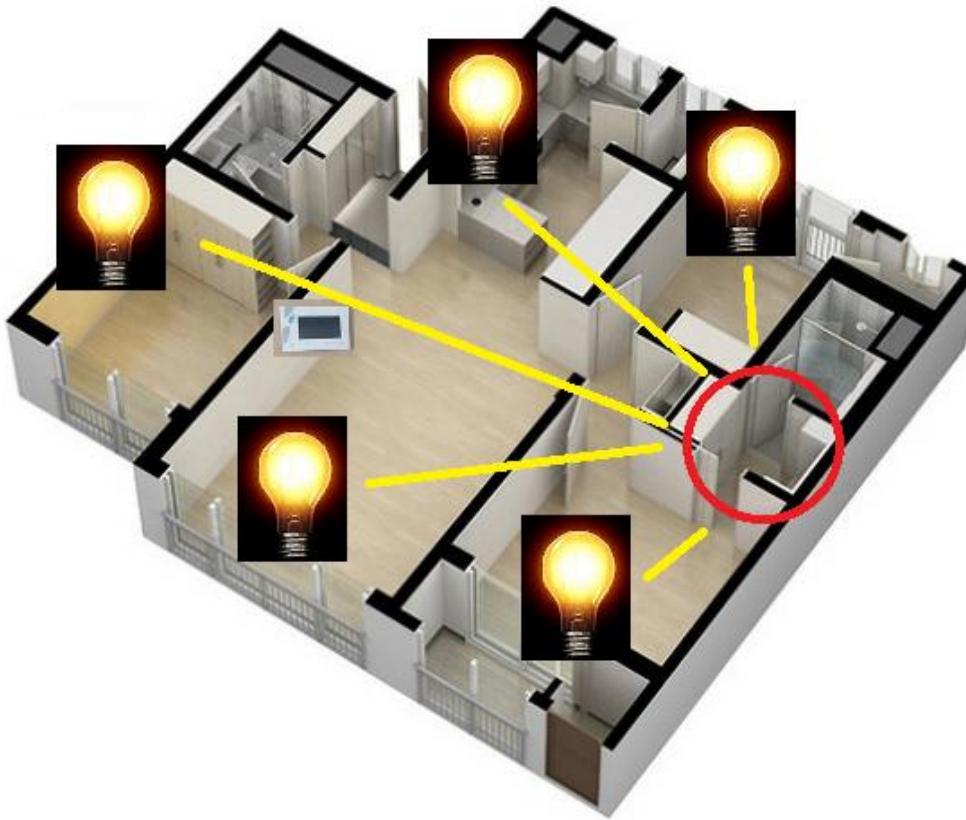


스마트홈 제어 방식



스마트홈 제어 방식

- 전등 제어



스마트홈 제어 방식

- 가스 제어



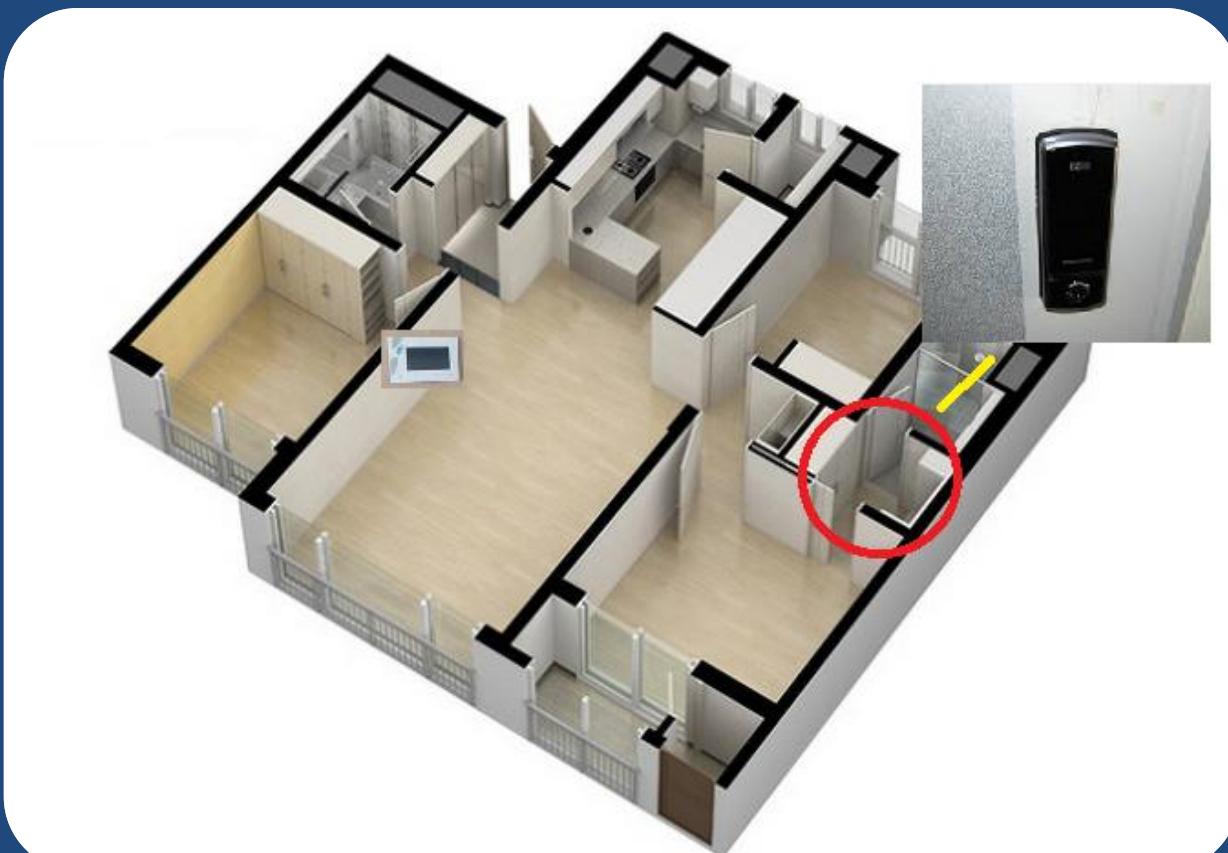
스마트홈 제어 방식

- 난방 제어



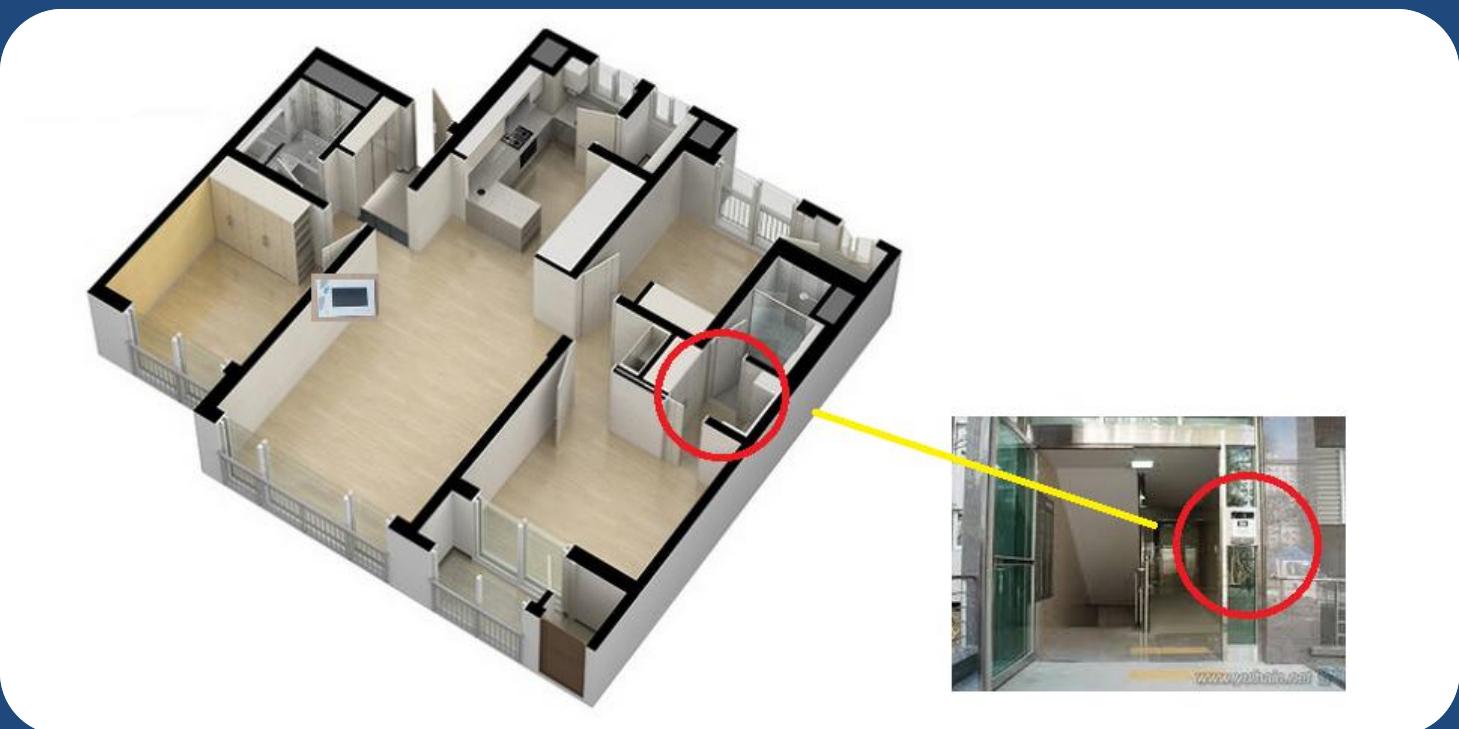
스마트홈 제어 방식

- 현관 도어락 제어



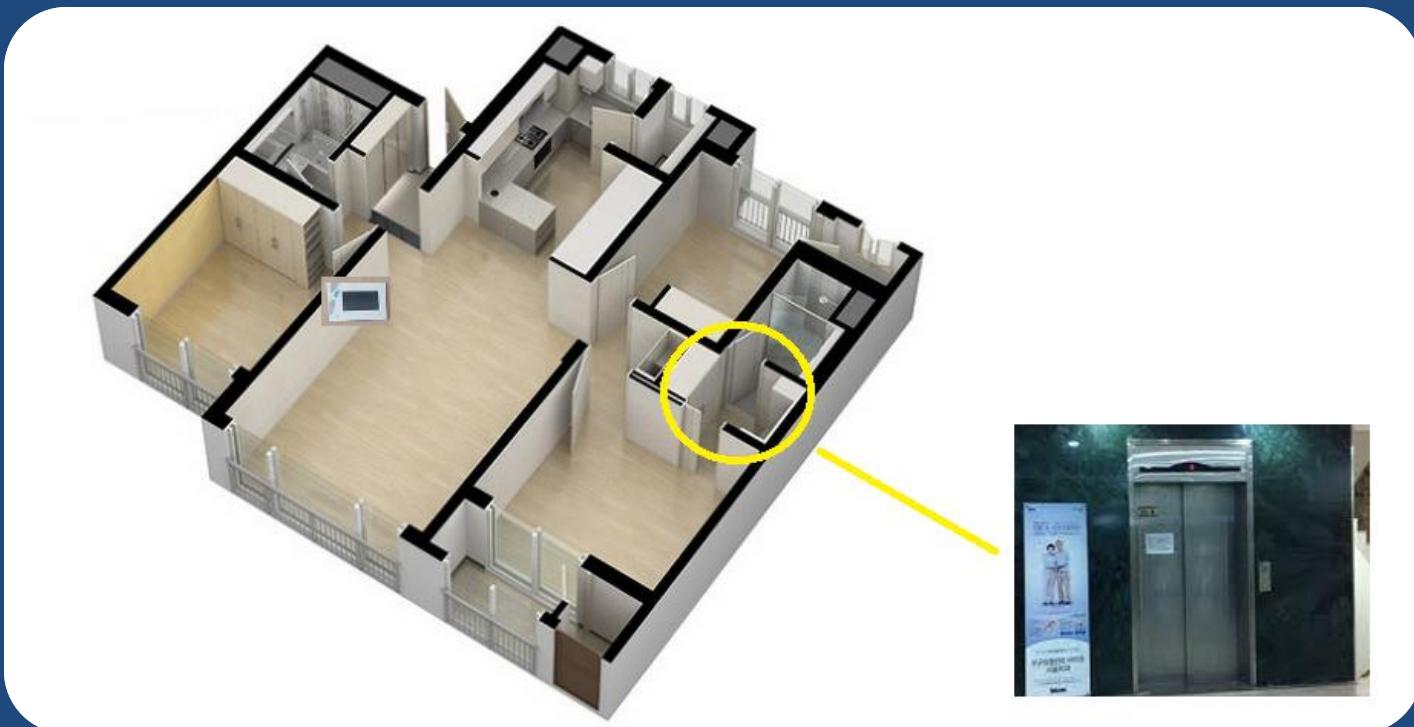
스마트홈 제어 방식

- 로비 출입문 제어



스마트홈 제어 방식

- 엘리베이터 호출



스마트홈 제어 방식

- 타 세대와의 음성/화상 통화 (P2P)



스마트홈 제어 방식

- 단지 내 모든 세대가 서로 연결되어 있음



스마트홈 제어 방식

- 사설망 (인터넷 연결 안 됨)



Step2

- 공격 대상 선정

Wallpad VS Gateway

- Wallpad
 - 사용자 UI 역할
 - 각종 제어 패킷 송신
 - P2P 화상 통화 기능
- Gateway
 - 라우터, 중앙 컨트롤러 역할
 - 각종 패킷 수신 및 주변장치 제어
 - 화상 통화에 관여하지 않음

Wallpad VS Gateway

- 둘 모두 결국 같은 패킷을 송수신하므로 둘 중 어떤 것을 분석해도 상관 없다.
 - 즉, 둘 중 하나를 이용해서 패킷 분석 가능
- 하지만, 화상 카메라는 Gateway와 단절되어 있기 때문에(실제 통화 시의 패킷만 지나감) 화상 카메라를 제어하기 위해선 결국 Wallpad를 노려야 한다.

Step3

- Wallpad 펌웨어(소프트웨어 획득)

펌웨어를 획득하는 여섯 가지 방법

1. 제조사에서 공개하는 펌웨어 다운로드
2. 자동/수동 업데이트가 될 때 패킷 스니핑
3. UART 포트 접속
4. 논리적 취약점을 이용하여 Shell 접근 권한 획득 후 추출
(partition dump, /dev/mtdblock)
5. Flash Memory 덤프
6. JTAG 포트 접속

1. 제조사에서 공개하는 펌웨어 다운로드

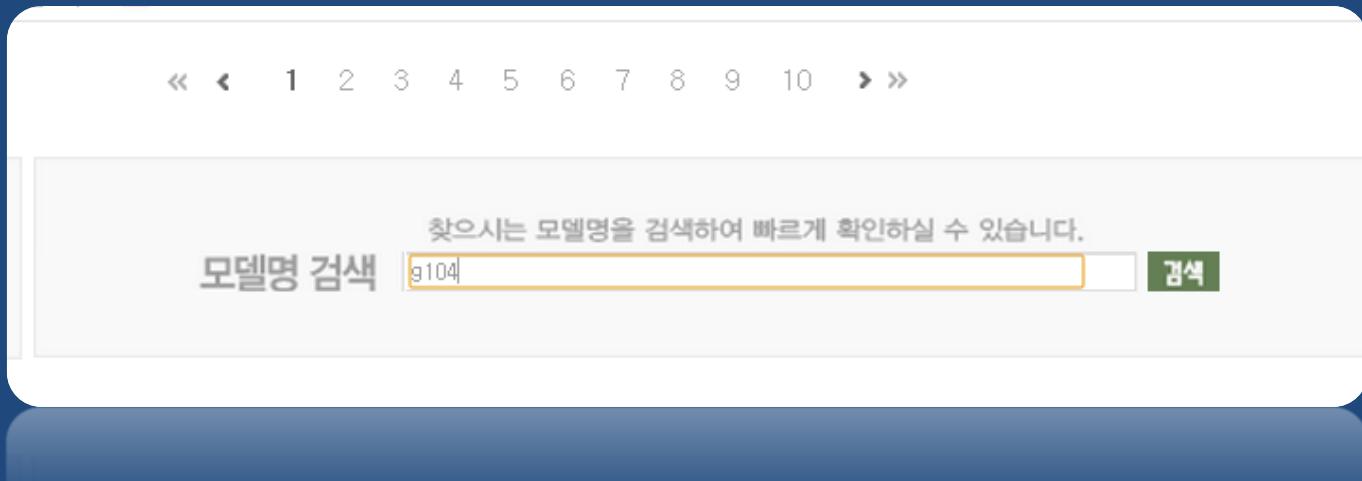
업데이트 파일 다운받기

The screenshot shows the ipTIME download page. At the top, there's a navigation bar with links for HOME, LOGIN, JOIN, and SITEMAP. Below that is a secondary navigation bar with links for 회사소개 (Company), 공지/뉴스 (News), 제품소개 (Product), 고객지원 (Customer Support), and 제품구입 (Product Purchase). The main content area has a title '다운로드' (Download) and three columns: '다운로드 구분' (Download Category) listing '전체보기', '펌웨어', '드라이버/유ти' (Windows), '드라이버/유티 MAC OS', '드라이버/유티 Linux', and '제품설명서'; '제품군' (Product Group) listing '유선공유기', '백업공유기', '11n 무선공유기', '11g 무선공유기', '유선랜카드', and '11n 무선랜카드'; and '모델명' (Model Name) listing 'ipTIME NAS-II', 'ipTIME N500U', 'ipTIME N704S', 'ipTIME N804', 'ipTIME HDD3025', and 'ipTIME N5'. A search bar labeled '검색' is located on the right side of the model list. On the far right, there's a sidebar with icons for '설치 도우미' (Setup Guide), '펌웨어 업그레이드' (Firmware Upgrade), '자주묻는 질문' (FAQ), 'A/S 안내' (Warranty), and 'ipTIME 검색기' (ipTIME Search).

번호	제목	날짜	조회
1671	ipTIME N1 펌웨어 버전 8.28	2012-06-26	64
1670	ipTIME G104A 펌웨어 버전 8.28	2012-06-26	99
1669	ipTIME Smart 펌웨어 버전 8.28	2012-06-26	65
1668	ipTIME N604A 펌웨어 버전 8.28	2012-06-26	381
1667	ipTIME N604S 펌웨어버전 8.28	2012-06-26	1185
1666	ipTIME NAS-II 펌웨어 1.1.30	2012-06-22	328

- http://www.iptime.co.kr/~iptime/bbs/zboard.php?id=sw_download

업데이트 파일 다운받기



번호	제목	날짜	조회
123	ipTIME g104 펌웨어 버전 8.46	2012-11-14	10896
122	ipTIME g104 펌웨어 버전 8.44	2012-11-07	2666
121	ipTIME g104i 펌웨어 버전 8.38	2012-09-06	1993
120	ipTIME g104M 펌웨어 버전 8.38	2012-09-06	6006
119	ipTIME g104BE 펌웨어 버전 8.38	2012-09-06	4293
118	ipTIME g104A 펌웨어 버전 8.38	2012-09-05	2098
117	ipTIME g104A 펌웨어 버전 8.32	2012-07-18	1069
116	ipTIME g104A 펌웨어 버전 8.30 (ipTIME 모바일 앱 지원)	2012-07-05	892
115	ipTIME g104A 펌웨어 버전 8.30 (ipTIME 모바일 앱 지원)	2012-07-02	885
114	ipTIME g104A 펌웨어 버전 8.30	2012-07-02	1098

업데이트 파일 다운받기

다운로드

제 목 : ipTIME G104 펌웨어 버전 8.46

다운로드 #1	g104_kr_8_46.bin (1.87 MB), Download : 8972
---------	---

펌웨어 정보

- 펌웨어 버전: 8.46
- 펌웨어 상태: 정식 버전(자동 업그레이드 적용됨)

문제점 해결

- 8.44 버전에서 VPN서버 접속이 안되는 문제 해결

주의 사항

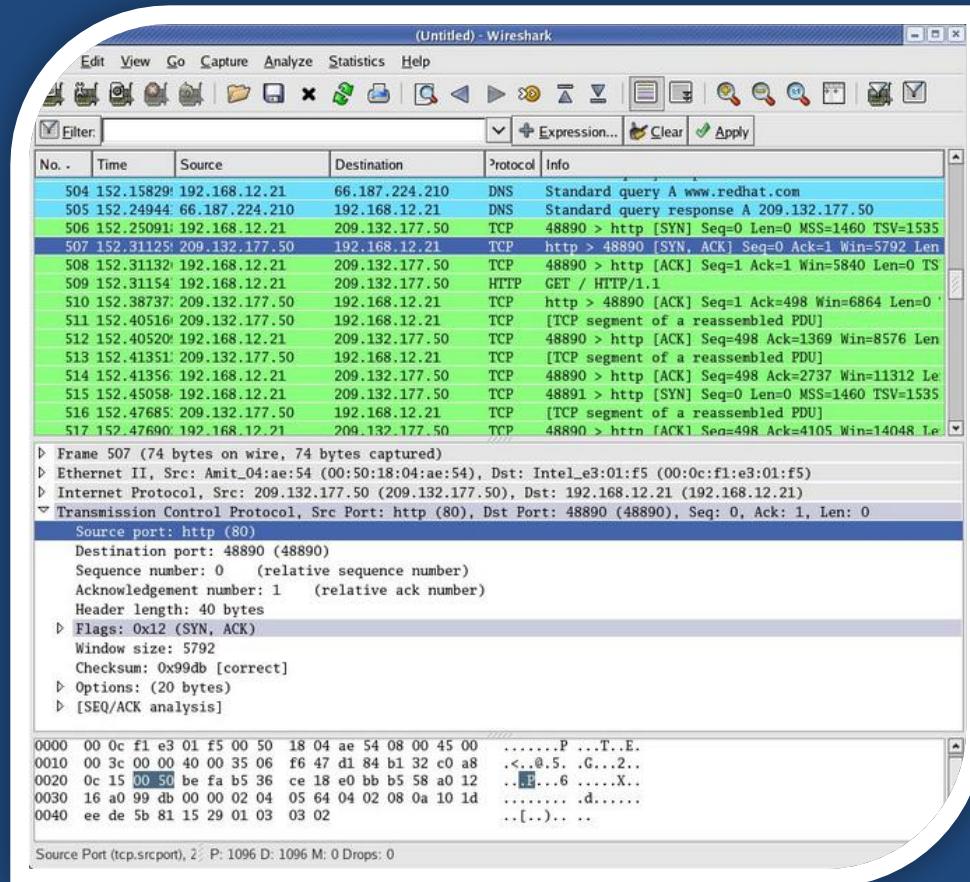
- 예기치 못한 상황으로 인하여 업그레이드가 실패할 경우, 아래의 문서를 참조하여 펌웨어를 복구할 수 있습니다.
참조> [펌웨어 복구 하기]

업데이트 파일의 구성

- Boot-loader
 - Kernel
 - Ram Disk (initrd)
 - Root File System (applications)
-
- 위와 같은 파일들이 하나의 파일로 **덩어리져 있다.**
 - 업데이트 파일의 성격에 따라 구성이 다를 수 있다.

2. 자동/수동 업데이트가 될 때 패킷 스니핑

자동 업데이트가 될 때 패킷 스니핑



자동 업데이트가 될 때 패킷 스니핑

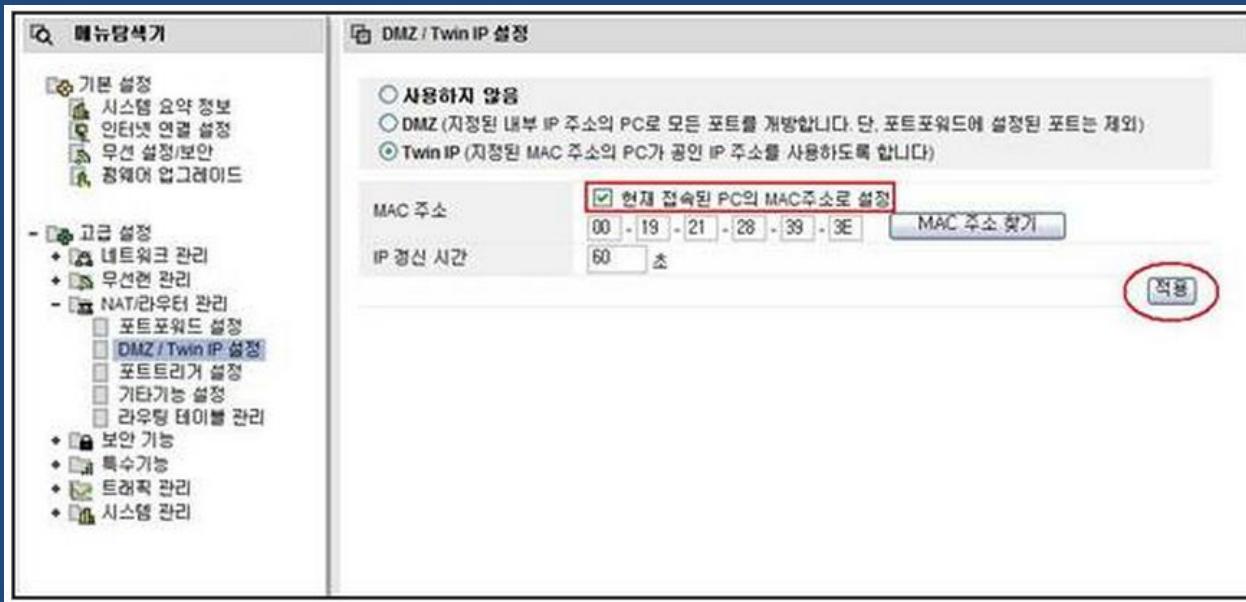
- 언제 업데이트가 되는가?
 - 기기를 재부팅 할 때 => 최신 버전 체크
 - 특정 기간이 지났을 때 ex> 매월 1일
 - 특수한 방법으로 기기를 켰 때
 - Ex> 파워키+리셋키 => 복구 모드
- 패킷 스니핑 방법들
 - 포트 미러링
 - TWIN IP
 - ARP Spoofing

포트 미러링

- 공격 대상 장비를 랜선에서 분리시킨 후,
- 그 랜선에 해커의 공유기를 연결
- 공유기에 대상 장비를 연결
- 공유기에 해커의 노트북을 연결
- “포트미러링” 기능을 이용하여 공유기로 오가는 모든 패킷을 해커의 노트북으로 전달
- 단점
 - 대상 기기가 고정 IP를 사용하는 경우 활용 어려움

TWIN IP

- 공유기의 IP주소를 NAT로 물린 특정 MAC의 내부 기기와 동일시 하는 기능
- 즉, 마치 공유기가 없는 것 같은 효과를 얻게 됨
- 대상 기기가 특정 IP 상에서만 통신이 가능할 경우에 유용



ARP Spoofing

- Ettercap 툴 추천
 - ettercap -T -M arp /192.168.0.1/ /192.168.0.10/
 - 192.168.0.1 : 라우터, 192.168.0.10 : Sniffing 대상

```
root@matriux:~# ettercap -Tq -M arp:remote /192.168.1.118/
ettercap NG-0.7.3 copyright 2001-2004 ALoR & NaGA

Listening on eth1... (Ethernet)

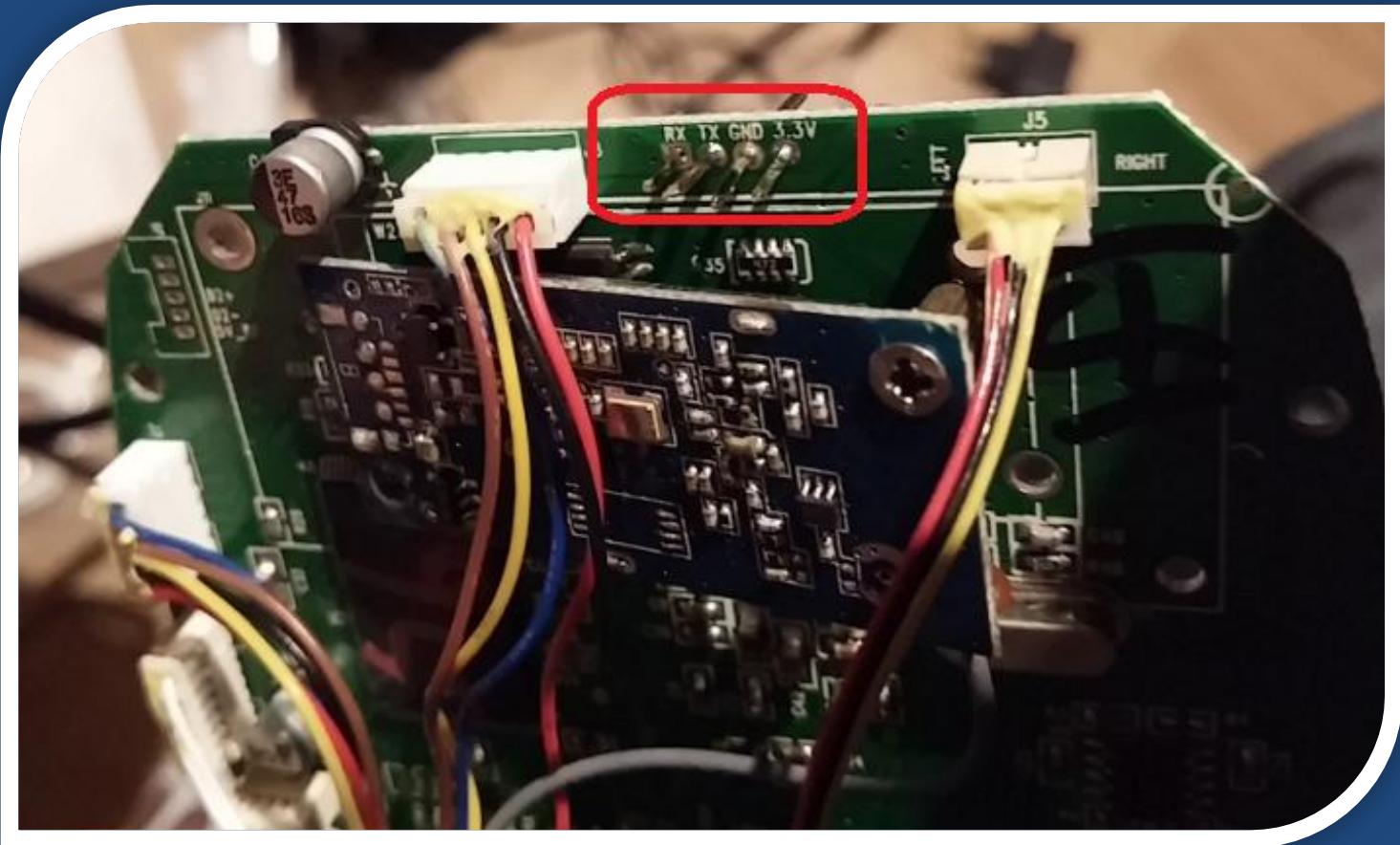
eth1 -> 08:00:27:02:BE:FB      192.168.1.120      255.255.255.0
Privileges dropped to UID 65534 GID 65534...
28 plugins
39 protocol dissectors
53 ports monitored
7587 mac vendor fingerprint
1698 tcp OS fingerprint
2183 known services

Scanning for merged targets (1 hosts)...

* | ======>| 100.00 %
```

3. UART PORT 접속

UART 포트 접속



UART PORT 접속

- Shell 접근이 가능할 시 파티션 덤프
- 부트로더 접근이 가능할 시 memory reading

4. 논리적 취약점 이용

논리적 취약점 이용

- 원격 백도어, Shell command execution 등의 단순한 취약점을 이용하여 Shell 획득
- /dev/에 접근하여 파티션 덤프

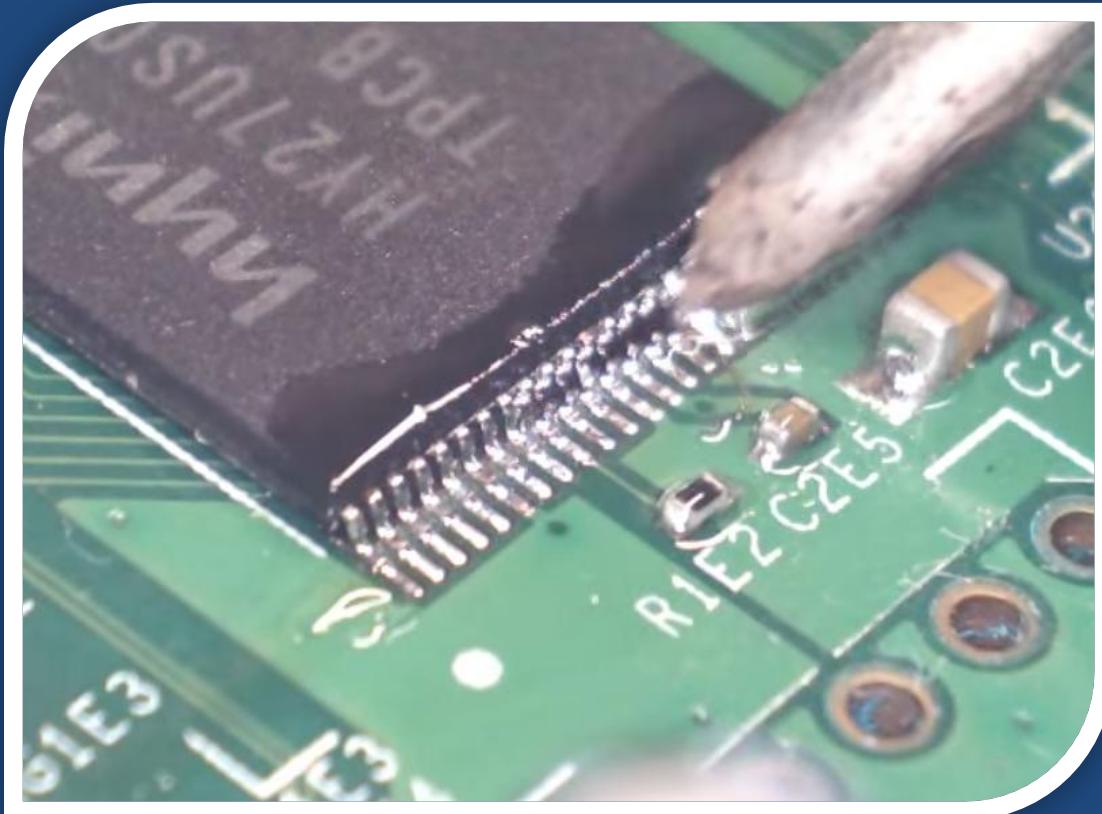
갤럭시S 파티션 덤프 예제

- 부트로더
 - dd if=/dev/block/bml1 of=/sdcard/boot.bin bs=512
- 커널
 - dd if=/dev/block/bml7 of=/sdcard/zImage bs=4096
- 파일시스템
 - dd if=/dev/block/stl9 of=/sdcard/factoryfs.rfs bs=4096

5. Flash Memory Dump

Flash Memory 뎁프

- desoldering



Flash Memory 덤프

- Socket Adaptor

자미전자 소켓 어댑터(JMTSO48-48PB)
48 pin TSOP (12mm x 20mm, Pitch 0.5mm) universal adapter



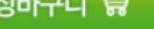
위 상품 이미지는 참조용 대표 이미지이며, 정확한 사양은 데이터시트에서 확인하셔야 합니다.

© 코인마이크로

• 상품코드	22625
• 판매가격	72,000원 (부가세 미포함)
• 제조사	자미전자
• 적립금	0원
• 평균준비기간	2~3일
• 브랜드	자미전자 [브랜드몰 바로가기]
• 최소주문수량	1 개
• 수량	<input type="button" value="1"/> ▲

×반품/취소불가

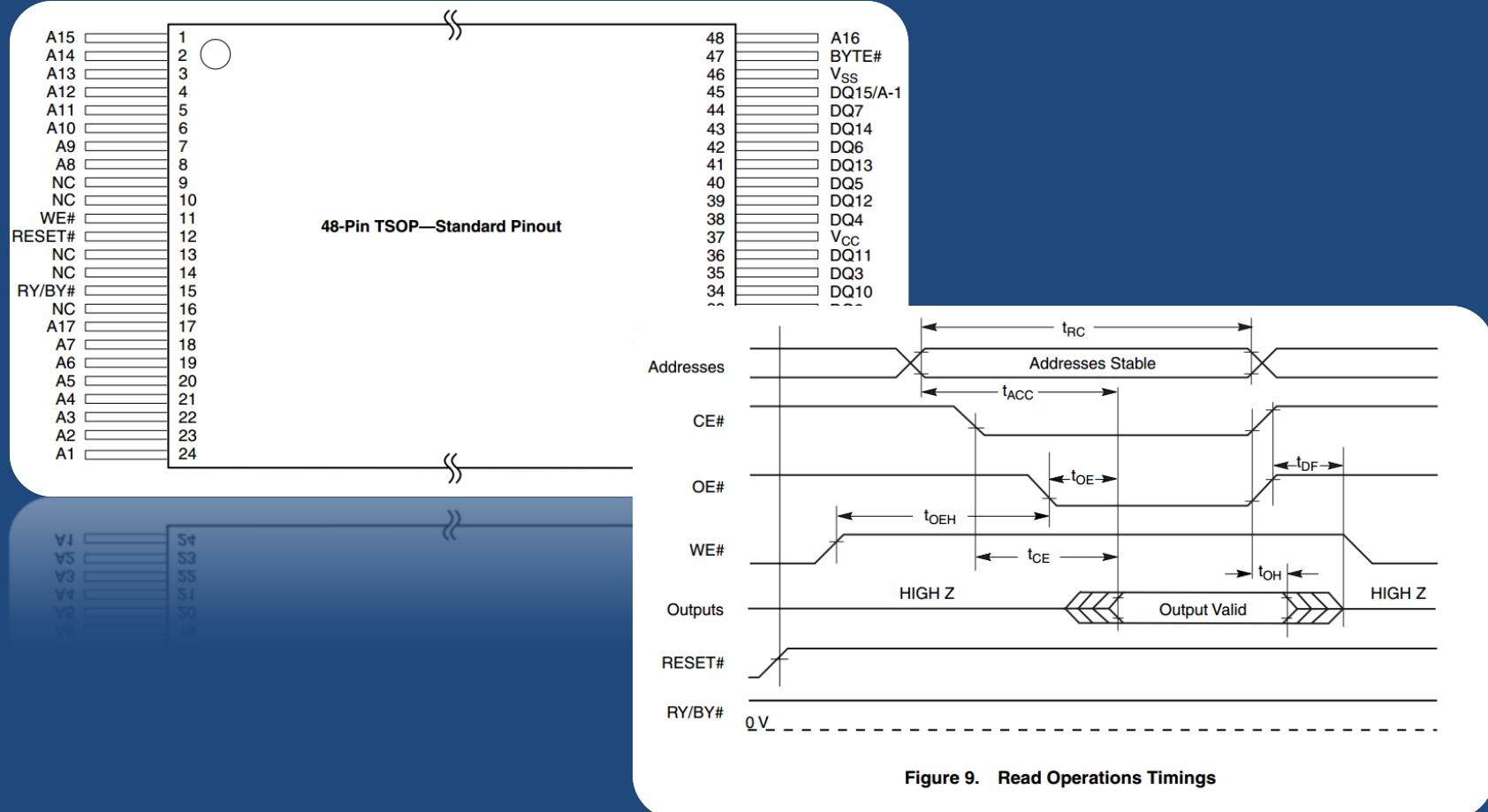
바로구매 

장바구니 

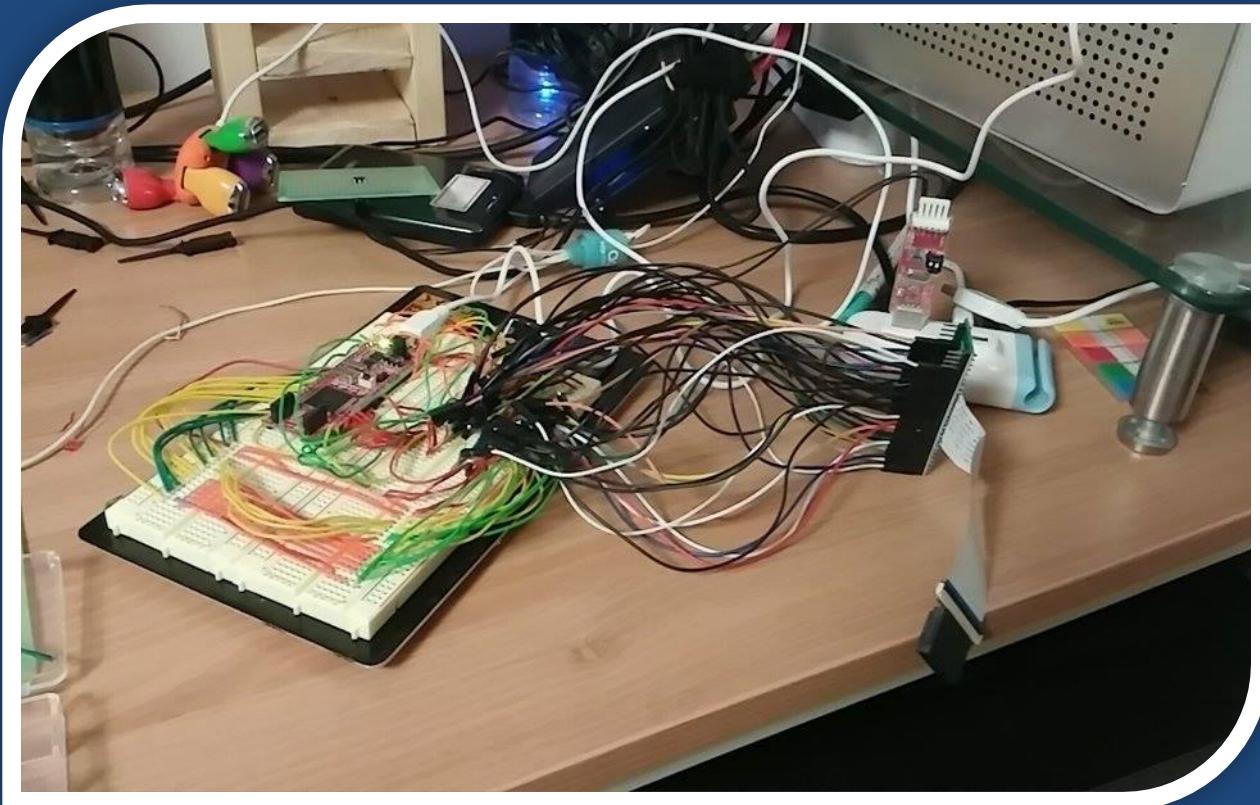
관심상품 

Flash Memory 둠프

- DataSheet 학습

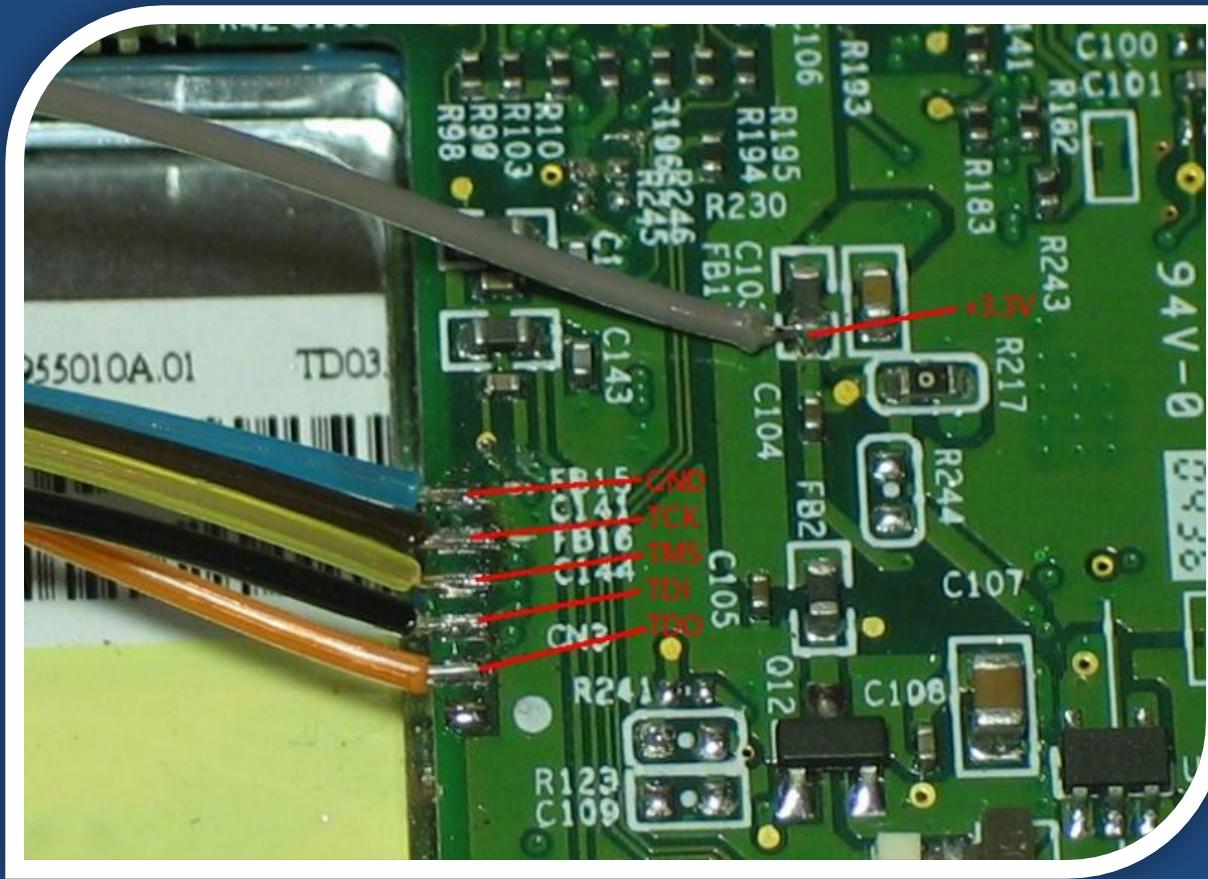


Flash Memory 덤프



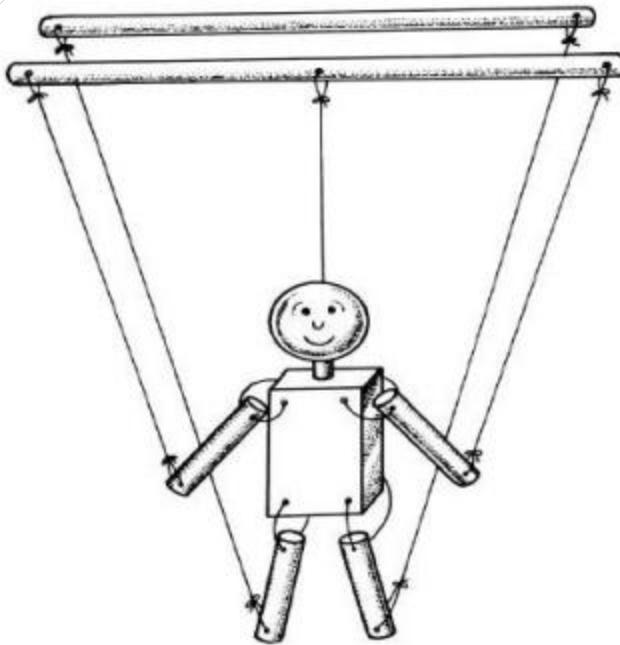
6. JTAG 포트 접속

CPU의 JTAG 포트 접속



JTAG의 비유

- JTAG = marionette(꼭두각시)
 - CPU를 마음대로 조정할 수 있다.



JTAG을 통해 Flash Memory 제어

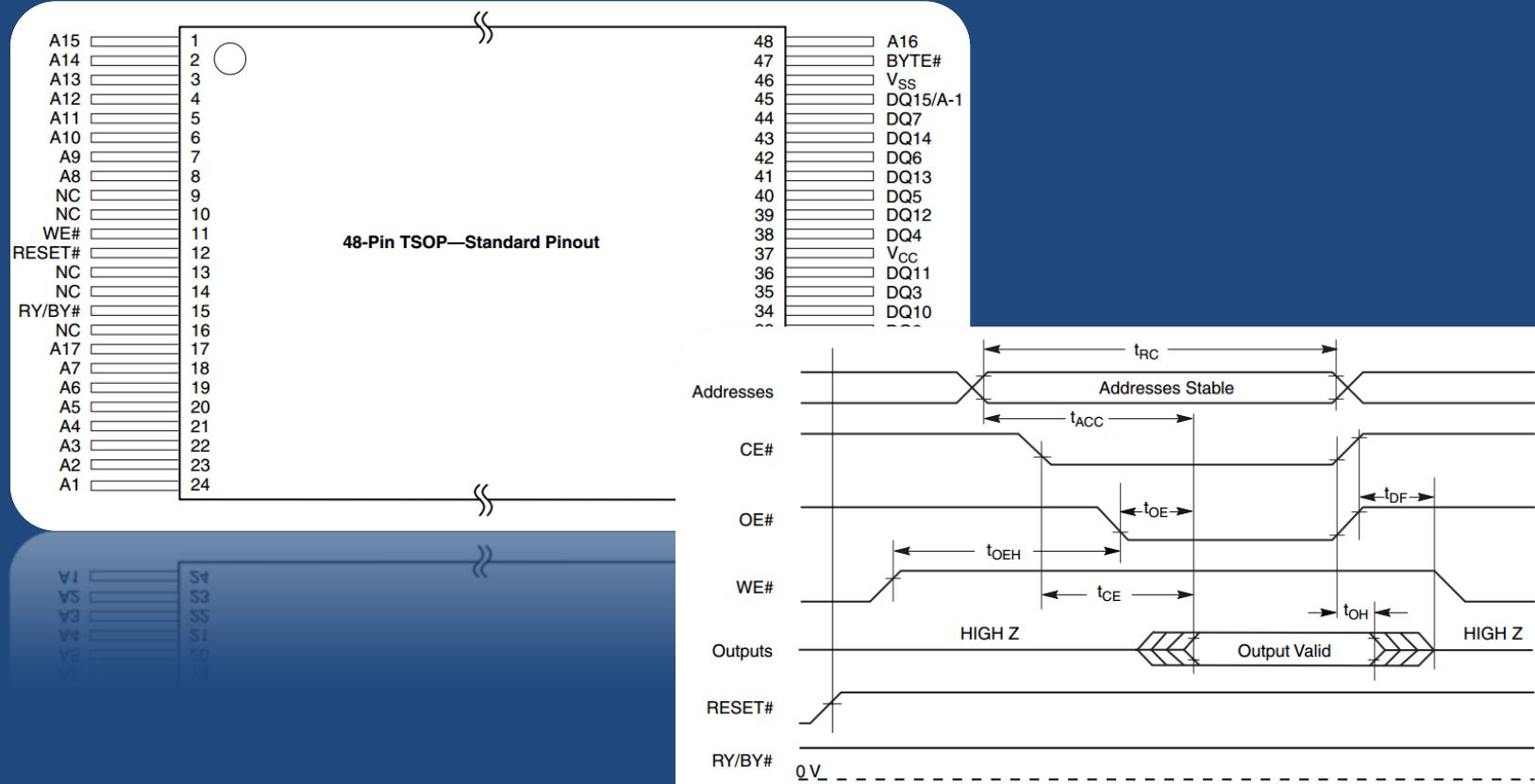


Figure 9. Read Operations Timings

획득한 Firmware 분석

펌웨어 자동 분석 툴

- Binwalk (Firmware Analysis Tool)
 - 펌웨어 파일의 구성 분석
 - 펌웨어 분석의 원리
 - Signature 탐색
 - Ex> squashfs == “hsqs”
 - <http://binwalk.org/>
- FMK (Firmware Mod Kit)
 - 펌웨어 파일 내에서 각종 파일 추출
 - 혹은 수정된 파일을 기반으로 새 펌웨어 빌드
 - <https://code.google.com/p/firmware-mod-kit/>

Firmware Mod Kit 사용 예제

```
[root@hackerschool trunk]# ./extract-ng.sh g104_kr_8_46.bin
Firmware Mod Kit (build-ng) 0.73 beta, (c)2011 Craig Heffner, Jeremy Collake
http://www.bitsum.com
```

Scanning firmware...

DECIMAL	HEX	DESCRIPTION
720896	0xB0000	Squashfs filesystem, little endian, version 3.0, size: 1235415 bytes, 257 inodes, blocksize: 65536 bytes, created: Wed Nov 14 13:28:13 2012

Extracting 720896 bytes of header image at offset 0

Extracting squashfs file system at offset 720896

Extracting 160 byte footer from offset 1957920

Extracting squashfs files...

Firmware extraction successful!

Firmware parts can be found in 'fmk/*'

```
[root@hackerschool trunk]#
```

Firmware Mod Kit 사용 예제

```
[root@hackerschool rootfs]# ls -al
합계 132
drwxr-xr-x 15 root      root      4096 10월 14  2014 .
drwxr-xr-x  5 root      root      4096  4월  3 14:22 ..
drwxr-xr-x  4 553779200 4160815104 4096 10월 14  2014 bin
drwxr-xr-x  6 553779200 4160815104 4096 10월 14  2014 default
drwxr-xr-x  2 553779200 4160815104 4096 10월 14  2014 dev
lrwxrwxrwx  1 root      root      8  4월  3 14:22 etc -> /tmp/etc
drwxr-xr-x  3 553779200 4160815104 4096 10월 14  2014 home
drwxr-xr-x  3 553779200 4160815104 4096 10월 14  2014 lib
lrwxrwxrwx  1 root      root     11  4월  3 14:22 linuxrc -> bin/busybox
drwxr-xr-x  2 553779200 4160815104 4096 10월 14  2014 ndbin
drwxr-xr-x  2 553779200 4160815104 4096 10월 14  2014 plugin
drwxr-xr-x  2 553779200 4160815104 4096 10월 14  2014 proc
drwxr-xr-x  2 553779200 4160815104 4096 10월 14  2014 save
drwxr-xr-x  2 553779200 4160815104 4096 10월 14  2014 sbin
drwxr-xr-x  2 553779200 4160815104 4096 10월 14  2014 tmp
drwxr-xr-x  2 553779200 4160815104 4096 10월 14  2014 upgrade-bin
drwxr-xr-x  5 553779200 4160815104 4096 10월 14  2014 usr
lrwxrwxrwx  1 root      root      8  4월  3 14:22 var -> /tmp/var
[root@hackerschool rootfs]#
```

Binary 분석

The screenshot shows the IDA Pro interface with the following details:

- File menu:** File, Edit, Jump, Search, View, Options, Windows, Help.
- Toolbars:** Functions, Strings, Hex, Structures, Enums, Imports, Exports.
- Windows:** Functions window, Strings window, Hex View-A, Structures, Enums, Imports, Exports.
- Functions window:** Shows a list of functions, with `sub_96A54` selected. A red box highlights the function names in the list.
- Assembly view:** Displays assembly code for several subroutines:
 - `sub_96A54:` moveq #1,d0; bra.w loc_96A84
 - `sub_96A5A:` moveq #2,d0; bra.w loc_96A84
 - `sub_96A60:` moveq #\$10,d0; bra.w loc_96A84
 - `sub_96A66:` moveq #\$20,d0; bra.w loc_96A84A red box highlights the labels and instructions of the first subroutine.
- Output window:** Shows memory allocation and loading logs:

```
3194880      total memory allocated
Loading processor module C:\Program Files (x86)\IDA\procs\mc68.w32 for ColdFire...OK
Loading type libraries...
Autoanalysis subsystem has been initialized.
Database for file 'pronto_modified3' has been loaded.
```
- Status bar:** AU: idle Down Disk: 4GB

월패드 분석 결과

1. 제조사에서 공개하는 펌웨어 다운로드
2. 자동/수동 업데이트가 될 때 패킷 스니핑
3. UART 포트 접속
4. 논리적 취약점을 이용하여 Shell 접근 권한 획득 후 추출
(partition dump, /dev/mtdblock)
5. Flash Memory 덤프
6. JTAG 포트 접속

Step4

- 월패드(wallpad) 분해

월패드 분해



월패드 분해



월패드 분해



월패드 분해



무언가를 분해했다면?

- CPU가 무엇인가?
 - ARM? MIPS?
 - CPU의 종류에 따라서,
 - 어셈블리어 문법이 달라진다.
 - 공격코드(쉘코드)의 구현 방법이 달라진다.
- Flash Memory가 무엇인가?
 - AMD? SAMSUNG?
 - Flash Memory의 종류에 따라서,
 - Reading/Writing 구현 방법이 달라진다.

무언가를 분해했다면?

- UART 포트가 있는가?
 - UART 콘솔 접속 가능
 - 주로 4핀 포트
 - “uart” “debug” “rs232” 라고 적힌 부분 확인
- JTAG 포트가 있는가?
 - CPU 동적 디버깅 가능
 - TDI, TDO, TMS, TCK라고 적힌 핀 확인

월패드 분해 결과

- CPU
 - NXP2120
 - ARM11 기반의 32비트 프로세서
- Flash Memory
 - K9F1G08U0D
 - 삼성 제작, Nand Type, TSOP Package (48p)
- UART 포트 (O)
- JTAG 포트 (X)

Step5

- UART 연결

UART 포트



UART 端口



UART란?

- Universal asynchronous receiver/transmitter
 - 범용 비동기 송/수신기
- 직렬 통신 프로토콜
 - 데이터 송신/수신 시 각각 하나의 LINE만 이용
- 하드웨어 통신 규약의 한 종류
- “프로토콜이 매우 간단함” => 디버깅 용도로 많이 쓰임

하드웨어 디버깅

- 임베디드 개발자들도 개발의 고충이 있다
 - 수 많은 버그들과의 싸움
- 기기의 상태 값을 실시간으로 출력하는 디버깅 방법 필요
 - LED로 출력? => 표현의 한계
 - LCD로 출력? => 구현이 복잡하고 화면 작음
 - 네트워크로? => 배보다 배꼽이 더...
- 그렇다면 개발자들의 선택은?
 - 단순한 **UART!**

UART의 장점들

- 프로토콜이 단순하다.
- 관련 프로그램 구하기가 쉽다.
 - Putty, Xshell, 하이퍼 터미널, ...
- 관련 장비를 구하기가 쉽다.
 - USB-UART, USB-RS232, USB-SERIAL

UART Pin의 구성

- 총 4개의 핀 사용
 - TX : 데이터 송신 핀
 - RX : 데이터 수신 핀
 - GND : 그라운드
 - VCC : 전압
- TX와 RX는 항상 자신의 입장에서 봐야 한다.
 - PC의 TX : PC에서 데이터 송신
 - 기기의 TX : 기기에서 데이터 송신



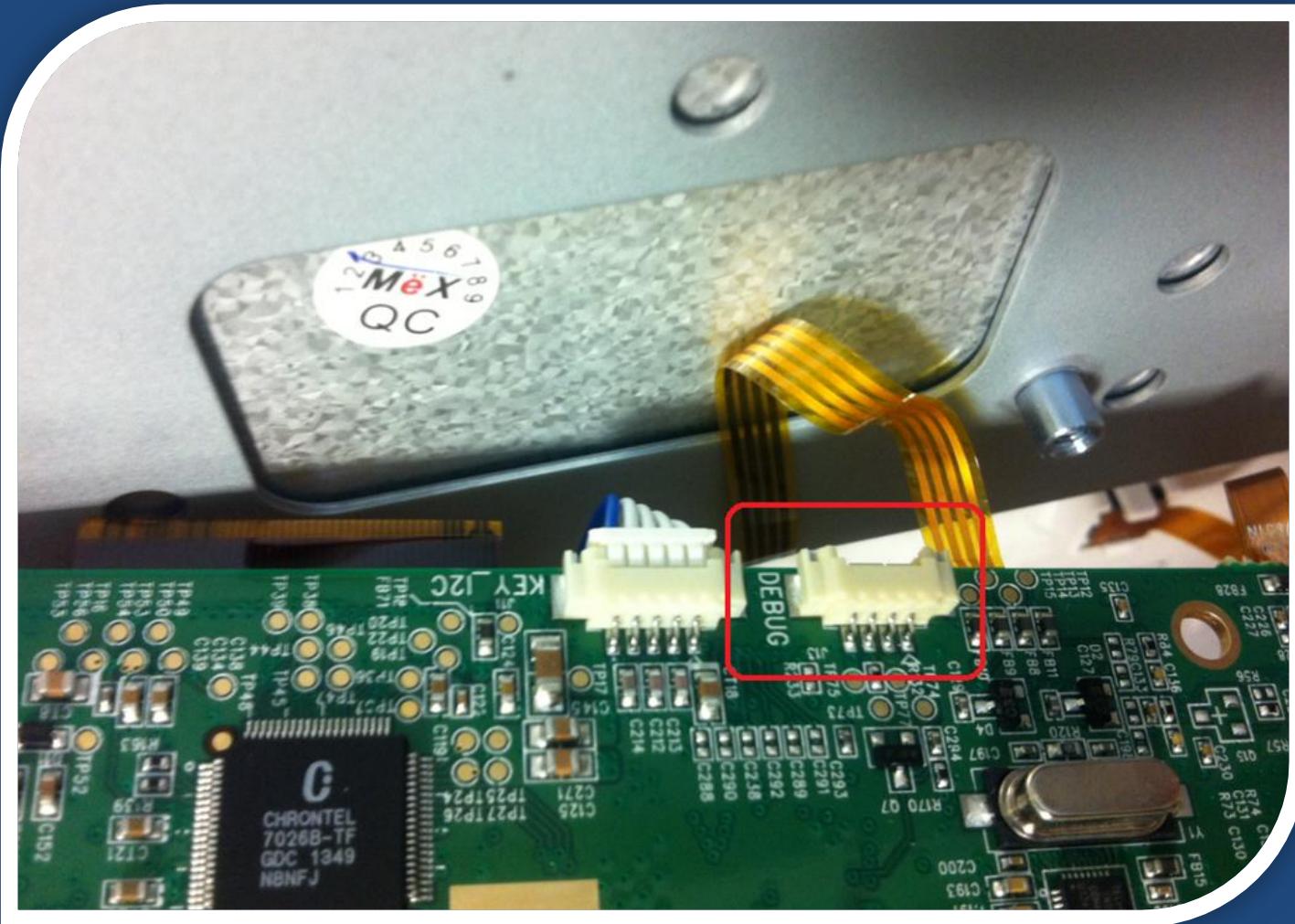
해커가 UART를 통해 얻을 수 있는 것들

- 커널, OS 메시지
 - 취약점 공략에 필요한 각종 정보 획득
- 디버그 메시지
 - Ex> printf("initializing network adaptor ok\n");
- 오류 메시지
 - Ex> Segmentation fault, command not found

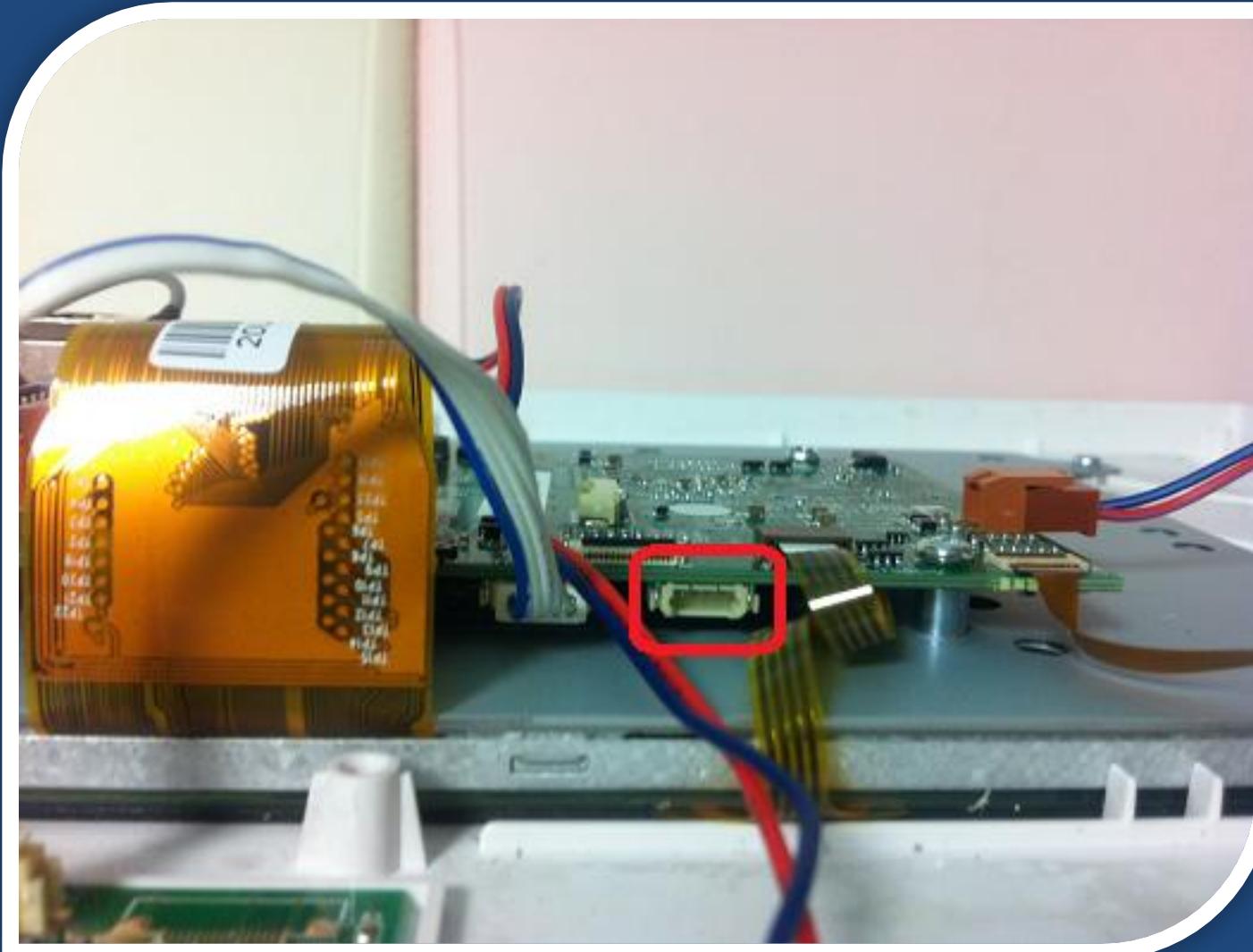
해커가 UART를 통해 얻을 수 있는 것들

- Hidden or Setting Menu
- 부트로더(Bootloader)
 - 펌웨어 획득
 - 새로운 펌웨어 Writing
- 커맨드 쉘(Command Shell)
 - 펌웨어, 바이너리 획득
 - 동적 분석

알파드의 UART 포트



문제점 - 너무 작은 UART 커넥터



UART 케이블 구매



UART 연결을 위한 장비

- USB-UART, USB-RS232, USB-SERIAL
 - USB 기반 UART 통신 장비
 - 장치관리자 -> 포트 -> COM(n)으로 연결 됨



[AD-USBISP-L] AVR용 USB-ISP라이트 [HD추천](#)

(제품번호 : EPX33LYK)

ATmega16, 32, 128등에 적용 (3.5V, 5V 포함)
USB포트를 통해 프로그램 다운로드 가능.

브랜드 : NewTC

제조사 : NewTC

원산지 : 한국

₩ 27,000 (VAT별도)

주문수량 EA

재고 : 있음

JK전자 USB to TTL for Rabbit 개발보드

TTL레벨의 신호를 PC의 USB포트(가상 COM포트)와 RS232레벨로 통신할 수 있도록 변환해 주는 컨버터 모듈입니다. 절퍼를 수 있도록 하여 타겟 보드의 전원에 맞추어서 사용할 수 있도록 하였습니다.



상품코드

32122

판매가격

9,000원 (부가세 미포함가)

제조사

JK전자

적립금

0원

평균준비기간

2~3일

브랜드

JK전자 [브랜드몰 바로가기](#)

최소주문수량

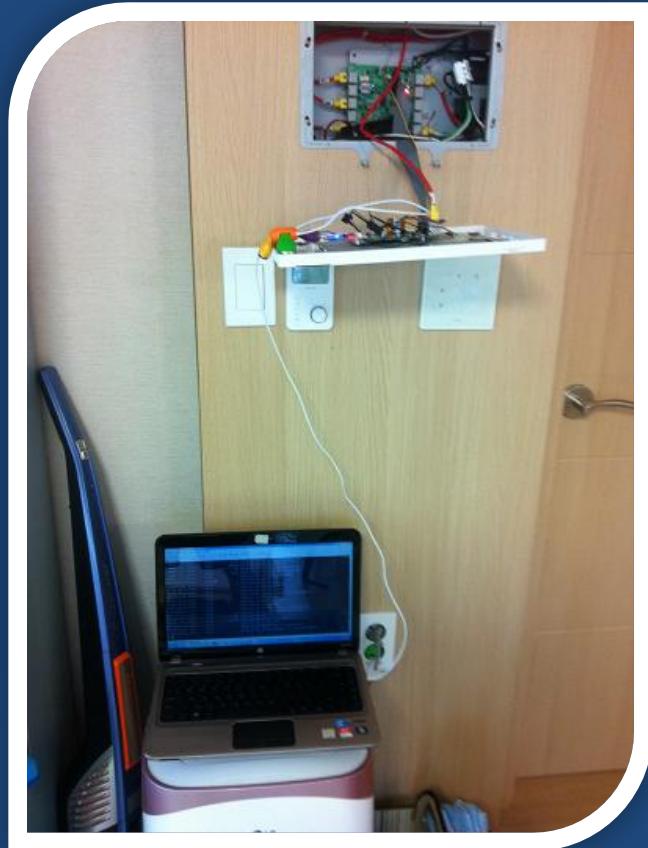
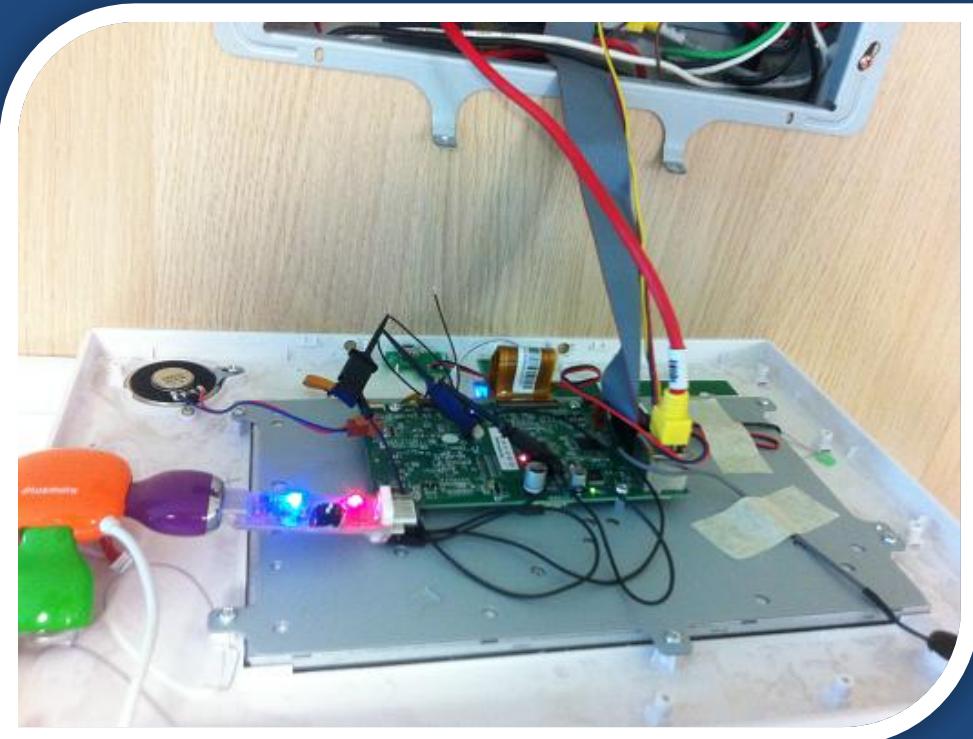
1 개

수량

UART 연결 절차 요약

- 관련 USB 드라이버 설치
 - CP2102, PL2303, FT232 등
- 점퍼 케이블 연결
 - RS, TX, GND, VCC
- 터미널 소프트웨어 설치
 - Putty
 - Xshell
- 연결 정보 설정 및 연결 수행

UART 포트 연결



UART 포트 연결



UART 연결 결과

- 동영상 Demo

Shell 접속 후 확인된 내용

- UART 접속 시 바로 root 쉘 획득
- Android 기반 운영체제
- Telnet(원격 관리) 포트가 열려있음
- 홈 네트워크 작동에 필요한 포트들이 열려있음
- 홈 네트워크 작동에 필요한 프로세스들 실행 중

Step6

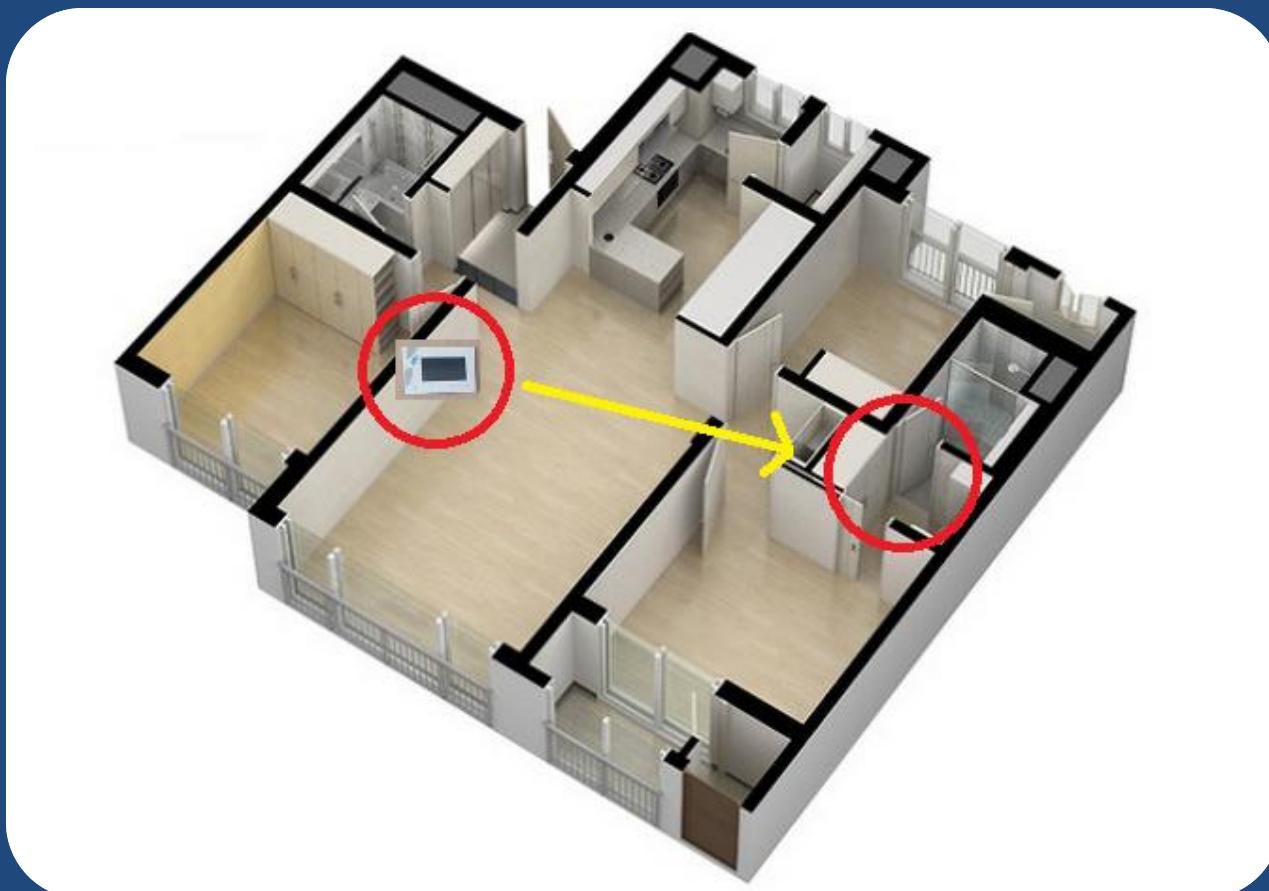
- 취약점 분석

홈 네트워크 해킹 구상

- 기본
 - 모든 제어는 network packet 기반으로 이루어 진다.
 - 그러므로 packet replay attack에 취약할 수 있다.
- 가정 (1)
 - 패킷 송신자의 identity/credential 검사를 하지 않을 것이다.
- 가정 (2)
 - 만약 검사를 한다면 spoofing/bypass가 가능할 것이다.

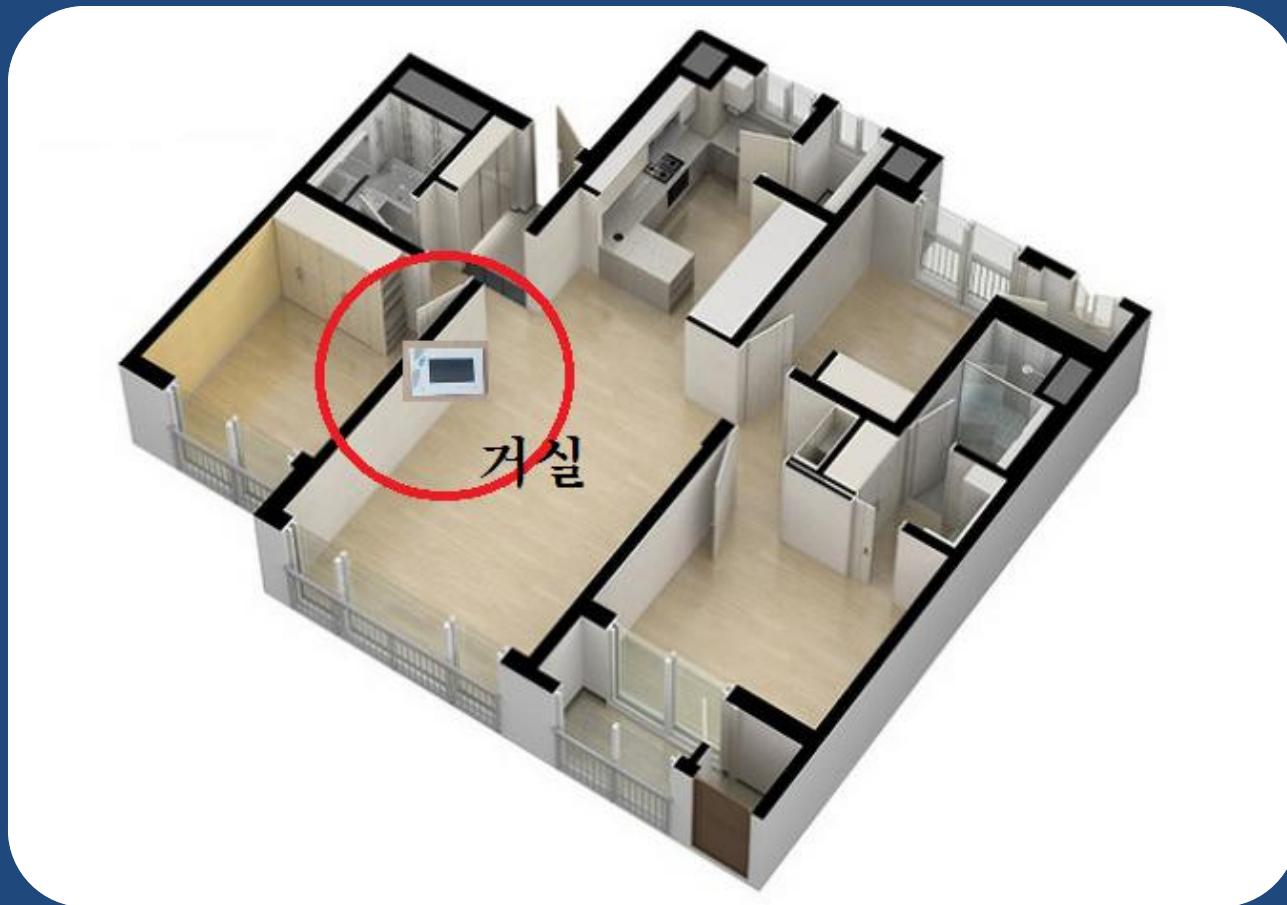
취약점 분석 대상 (1)

- Wallpad와 Gateway 사이의 패킷 분석
- 스마트홈 시스템 제어



취약점 분석 대상 (2)

- Wallpad device 장악
- 카메라/마이크 제어



취약점 분석 진행

- 바이너리 분석

```
Function name
sub_B818
sub_B82C
sub_B86C
sub_B888
sub_B158
sub_B228
sub_BE30
sub_BE68
sub_BEF4
sub_BF2C
sub_BFB0
sub_BFB4
sub_C060
sub_C12C
sub_C180
sub_C1C0
sub_C23C
sub_C41C
sub_C488
sub_C4D0
sub_C4E8
sub_C544
sub_C54C
sub_C59C
sub_C5C0
sub_C888
sub_CC50
sub_CD64
sub_CF84
sub_D03C
sub_D2D8
sub_D354
sub_D370
sub_D400
sub_D510

Line 143 of 469
Output window
9330: using guessed type int _fastcall sub_9330(_DWORD);
9368: using guessed type int _fastcall sub_9368(_DWORD);
9370: using guessed type int _fastcall sub_9370(_DWORD);
9408: using guessed type int _fastcall sub_9408(_DWORD);
9470: using guessed type int _fastcall sub_9470(_DWORD);
950C: using guessed type int _fastcall sub_958C(_DWORD);
9684: using guessed type int sub_9684(void);

9730: using guessed type int _fastcall sub_9730(_DWORD);
9740: using guessed type int _fastcall sub_9740(_DWORD);
9824: using guessed type int _fastcall sub_9824(_DWORD);
9860: using guessed type int _fastcall sub_9860(_DWORD);
974: using guessed type int _fastcall sub_B474(_DWORD, _DWORD);
974: using guessed type int _fastcall sub_E8DC(_DWORD);
974: using guessed type int _fastcall sub_14788(_DWORD);

000041C0 0000C1C0: sub C1C0
    STMD SPT, {R4,R5,LR} ; CODE XREF: .text:00000F4tp
    NOV R4, R8
    BL sub_C12C
    LDR R8, -aVar1_7 ; "Var1"
    NOV R1, R4
    BL sub_D93C
    BL sub_C180
    NOV R6, R4
    BL sub_E444
    NOV R5, R8
    BL sub_D354
    CMP R6, #0
    BEQ loc_C228
    CMP R4, #0
    BEQ loc_C208
    LDR R8, -aSbinInsmodLi_2 ; "/sbin/insmod /lib/modules/smart_qos_mon"...
    BL .system
    B loc_C228
    ; CODE XREF: sub_C1C0+387j
    LDR R8, -aSbinRmmodSmart ; "/sbin/rmmod smart_qos_mon"
    .system
    BL sub_C12C
    NOV R8, #1
    BL sub_CF84
    NOV R8, R4
    BL sub_E4AC
    BL sub_C180
    ; CODE XREF: sub_C1C0+387j
    ; sub_C1C0+447j
    NOV R8, R5
    LDMD SPT, {R4,R5,PC}
    .text:0000C22C ; End of function sub_C1C0
```

취약점 분석 진행

- 네트워크 패킷 분석 (tcpdump + wireshark)

```
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 96 bytes
1:39:00.239963 arp who-has 10-1-119-90.int.sds.uw.edu.pl tell 10-1-232-251.int.sds.uw.edu.pl
  0x0000: ffff ffff ffff 0010 5ae6 d045 0806 0001 .....Z..E....
  0x0010: 0800 0604 0001 0010 5ae6 d045 0a01 e8fb .....Z..E....
  0x0020: 0000 0000 0000 0a01 775a 0000 0000 0000 .....wZ.....
  0x0030: 0000 0000 0000 0000 0000 0000 0000 0000 .....
01:39:00.240803 IP 10-1-225-220.int.sds.uw.edu.pl.32786 > 10-1-254-254.int.sds.uw.edu.pl.domain: 2
680+ PTR? 90.119.1.10.in-addr.arpa. (42)
  0x0000: 0030 4884 5ef6 000f ea39 d0e0 0800 4500 .0H.^....9....E.
  0x0010: 0046 1a89 4000 4011 2b41 0a01 eldc 0a01 .F..@.A.....
  0x0020: fefe 8012 0035 0032 f520 0a78 0100 0001 .....5.2...x....
  0x0030: 0000 0000 0239 3003 3131 3901 3102 .....90.119.1.
  0x0040: 3130 0769 6e2d 6164 6472 0461 7270 6100 10.in-addr.arpa.
  0x0050: 000c 0001 .....
01:39:00.253666 IP 10-1-254-254.int.sds.uw.edu.pl.domain > 10-1-225-220.int.sds.uw.edu.pl.32786: 2
680 1/0# PTR[|domain]
  0x0000: 000f ea39 d0e0 0030 4884 5ef6 0800 4500 ...9...0H.^....E.
  0x0010: 0071 0000 4000 4011 459f 0a01 fefe 0a01 .q..@.E.....
  0x0020: eldc 0035 8012 005d 334c 0a78 8180 0001 ...5...]3L.x.....
  0x0030: 0001 0000 0000 0239 3003 3131 3901 3102 .....90.119.1.
  0x0040: 3130 0769 6e2d 6164 6472 0461 7270 6100 10.in-addr.arpa.
  0x0050: 000c 0001 c00c 000c 0001 0001 4a78 001f .....Jx..
01:39:00.255938 IP 10-1-225-220.int.sds.uw.edu.pl.32786 > 10-1-254-254.int.sds.uw.edu.pl.domain: 6
932+ PTR? 251.232.1.10.in-addr.arpa. (43)
  0x0000: 0030 4884 5ef6 000f ea39 d0e0 0800 4500 .0H.^....9....E.
  0x0010: 0047 1a8d 4000 4011 2b3c 0a01 eldc 0a01 .G..@.A.+<.....
  0x0020: fefe 8012 0035 0033 f521 1b14 0100 0001 .....5.3.!.....
  0x0030: 0000 0000 0332 3531 0332 3332 0131 .....251.232.1.
```

발견된 취약점 정리

1. telnet 서비스(/user/app/bin/telnetd)가 열려 있으며, passwd가 암호화 되어 있지 않고, 기기별로 다르게 설정되어 있지 않음
2. 모든 제어 통신 패킷이 암호화 되어 있지 않아 해커가 쉽게 분석 가능
3. 모든 제어 통신 패킷에 인증 절차 및 ACL 제어가 적용되어 있지 않음
4. 특정 서비스(/user/app/bin/cmxnp)를 통해 원격 임의 명령 실행 가능
5. 많은 서비스들에 secure coding이 되어 있지 않아 Buffer Overflow 및 Format String 취약점에 노출되어 있음

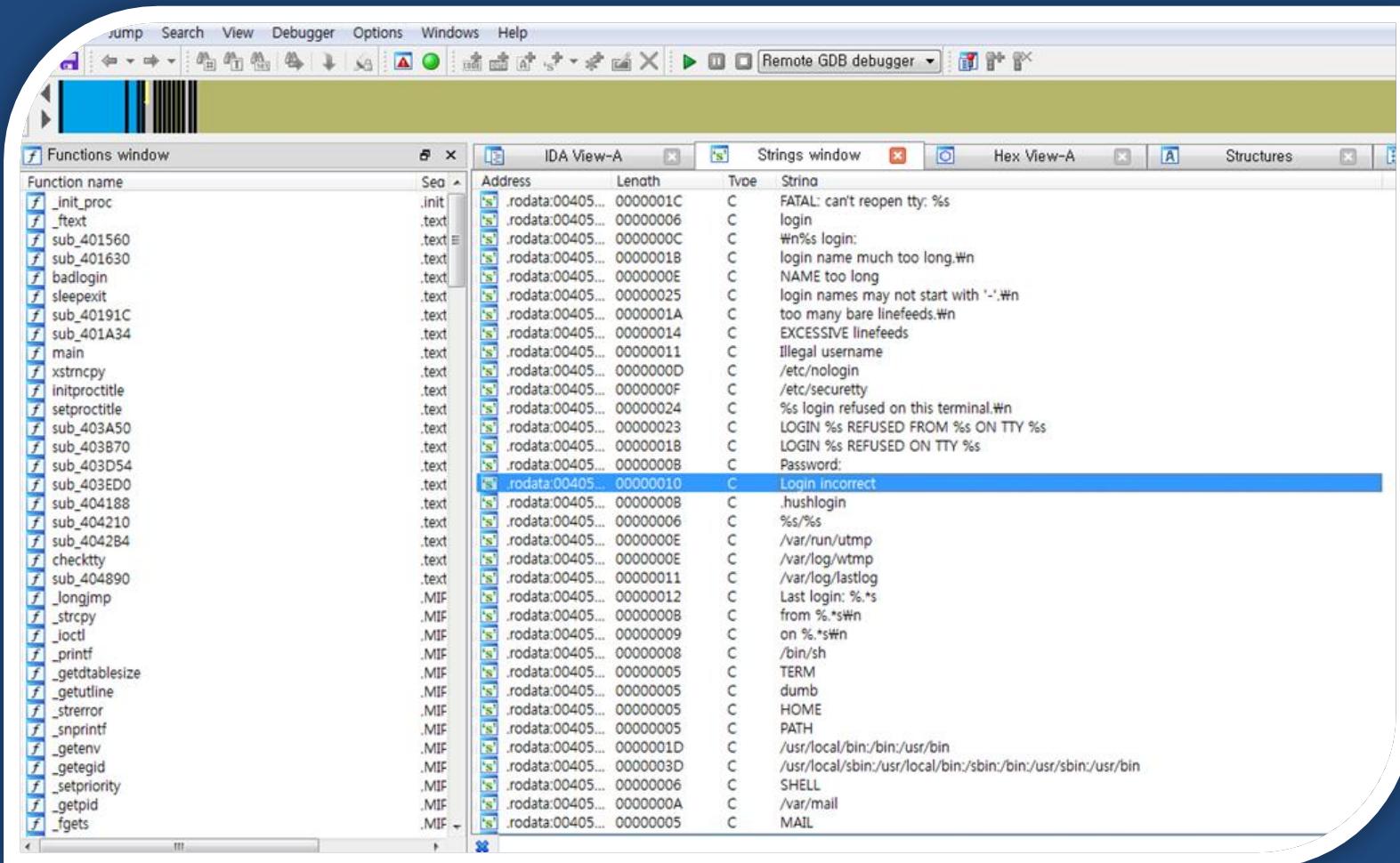
Step7

- 공격(Exploitation) 진행

Telnet 계정 분석

- /etc/passwd, /etc/shadow가 존재하지 않음
- /usr/bin/login
 - 계정 정보를 담고 있는 파일 탐색
=> 계정 정보를 바이너리 안에 가지고 있는 것을 확인
 - 암호가 평문 형태로 존재하는 것을 확인
 - Hash를 사용하지 않음

Telnet 계정 분석



Telnet 계정 분석

The screenshot shows the IDA Pro interface with several windows open:

- Functions window:** Lists various functions including `_init`, `badlogin`, `main`, `xstrncpy`, `initproctitle`, `setproctitle`, `sub_403A50`, `sub_403B70`, `sub_403D54`, `sub_403ED0`, `sub_404188`, `sub_404210`, `sub_404284`, `checkatty`, `sub_404890`, `_longjmp`, `_strcpy`, `_lctl`, `_printf`, `_getdtablesize`, `_getline`, `_sterror`, `_snprintf`, `_getenv`, `_getegid`, `_setpriority`, `_getpid`, and `_fgets`.
- IDB View-A:** Shows the assembly code for the main function.
- Strings window:** Displays strings from the binary, including "Login incorrect".
- Hex View-A:** Provides a hex dump of the assembly code.
- Comments:** Annotations are present in the assembly code to explain the logic:

```
        .text:00402900    lw    $v0, (pwd - 0x100002A8)($v0)
        .text:00402904    nop
        .text:00402908    beqz $v0, loc_40292C
        .text:0040290C    nop
        .text:00402910    la    $t9, strcmp
        .text:00402914    lw    $a1, 4($v0)      # s2
        .text:00402918    jalr $t9 ; strcmp
        .text:0040291C    move $a0, $s1      # s1
        .text:00402920    lw    $gp, 0x44F0+var_44D8($sp)
        .text:00402924    beqz $v0, loc_4029C4
        .text:00402928    nop
        .text:0040292C    .loc_40292C:          # CODE XREF: main+EACTj
        .text:0040292C    la    $a0, 0x400000
        .text:00402930    la    $t9, puts
        .text:00402934    nop
        .text:00402938    jalr $t9 : puts
        .text:0040293C    addiu $a0, (loginIncorrect - 0x400000) ## "Login incorrect"
        .text:00402940    lw    $gp, 0x44F0+var_44D8($sp)
        .text:00402944    nop
        .text:00402948    la    $v0, 0x10000000
        .text:0040294C    la    $t9, badlogin
        .text:00402950    lw    $a0, (dword_100002F4 - 0x10000000)($v0)
        .text:00402954    jalr $t9 : badlogin
        .text:00402958    addiu $s4, 1
        .text:0040295C    lw    $gp, 0x44F0+var_44D8($sp)
        .text:00402960    lw    $a1, 0x44F0+seconds($sp)
        .text:00402964    la    $v1, 0x10000000
        .text:00402968    addiu $a1, 5
        .text:0040296C    lw    $v0, (dword_10000054 - 0x10000000)($v1)
        .text:00402970    slti $a0, $s4, 4
        .text:00402974    addiu $v0, 1
        .text:00402978    sw    $a1, 0x44F0+seconds($sp)
        .text:0040297C    bnez $a0, loc_402274
        .text:00402980    sw    $v0, (dword_10000054 - 0x10000000)($v1)
        .text:00402984    slti $v0, $s4, 0xA
```

IP 체계 분석

- Gateway : 10.7.5.30
- Wallpad : 10.7.5.31
- 10 : 공통
- 7 : 동
- 5 : 층
- 3x : 호수
- 30 : gateway
- 31 : wallpad

스마트홈 강제 제어 취약점

- 전등 제어
- 현관 도어락 제어
- 임의 명령 실행
- 화상 카메라/마이크 제어

스마트홈 제어 패킷 예제

전등 제어 패킷



전등 제어

- 동영상 Demo

스마트홈 제어 패킷 예제

- 현관 도어락 오픈 패킷



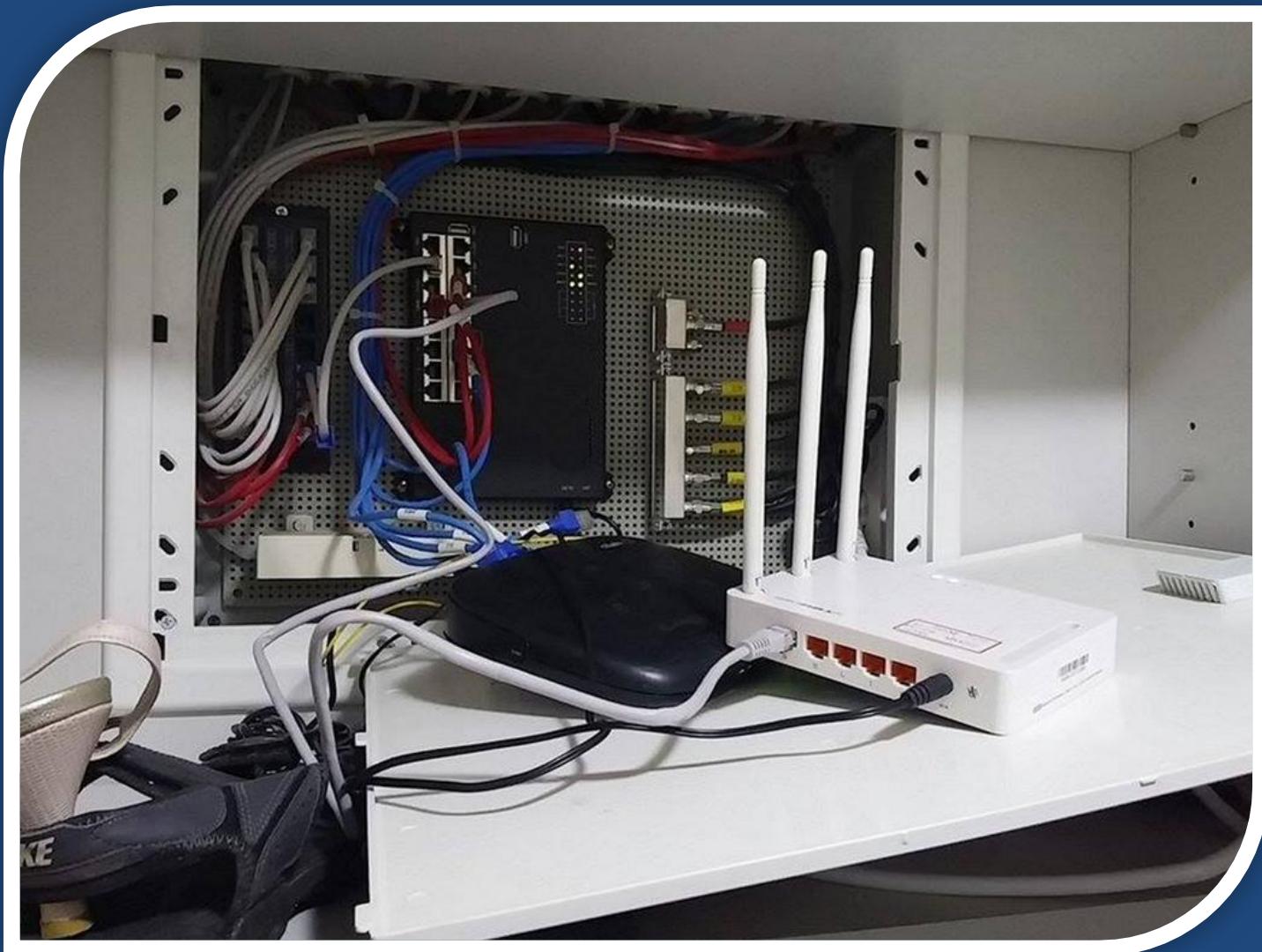
현관 도어락 제어

- 동영상 Demo

외부에서의 접근 방법

- 집 내부에 두 개의 공유기 설치
 - 공유기1 : 인터넷 라인
 - 192.168.0.1/255
 - 공유기2 : 홈네트워크 라인
 - 192.168.1.1/255
 - 공유기 1은 PC의 유선 어댑터로 연결
 - 공유기 2는 PC의 무선 어댑터로 연결
 - 이 라인에 리눅스 연결
 - 공유기 1의 특정 포트로 들어오는 연결을 공유기 2에 연결되어 있는 리눅스의 SSH로 포트 포워딩

외부에서의 접근 방법



스마트홈 제어 패킷 예제

- 임의 명령 실행 가능



스마트홈 제어 패킷 예제

- 화상 카메라/마이크 제어 명령
- Gstreamer Library 이용

월패
/u
ypos:
widt

해커
gs
locat



l=4,
l61

화상 카메라 제어

- 동영상 Demo

대응 방안

- wallpad/gateway의 취약점(**특히 remote**) 제거
 - 계정 정보 및 각종 패스워드 암호화
 - BOF, FS 및 논리적 취약점 패치
- 해커의 리버싱을 방해한다.
 - 난독화(Obfuscation)
 - 안티 리버싱(Anti-Reversing)

대응 방안

- 이상 패킷 모니터링
 - 허용 리스트에 없는 IP나 MAC으로부터 패킷 수신 시 경고
 - 예상되지 않은 패킷 발생 시 경고
 - 디바이스에 Shell 접속이 이루어질 시 경고
- 마이크/카메라 사용 시 하드웨어적으로 표시
- 제어 패킷 송신자의 identity 확인
 - IP, MAC Address
- 제어 패킷 암호화
 - 대칭키
 - 비대칭키

제어 패킷 송신자의 identity 확인

- 송신자의 IP가 설치된 기기의 IP가 맞는가?
 - 해커가 임의로 할당 받은 IP 차단
- 해당 IP의 MAC이 올바른가?
- 단점
 - IP Spoofing 및 ARP Spoofing 가능
- 해결책
 - 제어 패킷 암호화

제어 패킷 암호화 (대칭키)

- 대칭키 암호화
 - 제 3자가 패킷을 해석하거나 변조하지 못한다.
 - 세대별로 서로 다른 KEY를 사용해야 한다.
 - 101호 : wallpad(keyA 사용) <-> gateway(keyA 사용)
 - 102호 : wallpad(keyB 사용) <-> gateway(keyB 사용)
 - Key의 규칙성이 존재하면 안된다.
- 단점
 - Packet replay attack에는 여전히 취약하다.
- 해결책
 - Timestamp
 - Nonce

해결책

- **Timestamp**
 - 제어 패킷 안에 시간 정보를 함께 보낸다.
 - 허용 시간 범위내의 패킷이 아니라면 무시한다.
- **Nonce**
 - 매 요청 시마다 바꾸는 nonce 값을 이용하여 암호화
 - 요청이 끝나면 해당 nonce 값은 폐기
 - 해커가 packet replay attack을 했을 때 nonce가 다르기 때문에 packet이 무시됨
- 보통은 평문+Timestamp|Nonce를 HMAC으로 생성
 - HMAC : keyed-hash message authentication code

제어 패킷 암호화 (비대칭키)

- 제 3자가 패킷을 해석하거나 변조하지 못한다.
- 단점
 - 해커가 자신의 공개키/개인키 사용 가능
 - 본인 장비 분석을 통해 평문의 포맷은 알고 있다고 가정
- 해결책
 - Certificate Pinning
 - Permanent Session

해결책

- Certificate Pinning
 - 특정 기관에서 발급한 인증서만 인정하도록 제한
 - 본 홈 네트워크 환경에서는 해당 안 됨
 - 무조건 정해진 public key만 사용하도록 고정
 - Ex> wallpad A의 public key만 사용 가능
- Permanent Session
 - 홈 네트워크 시스템 최초 초기화 시 random한 Session 생성 후 gateway와 wallpad가 공유
 - 이 값이 맞아야만 정상적인 통신 가능

현재 패치 상황

- UART 콘솔 접속 불가
- telnet 서비스 접속 불가
 - SSH로 대체, shadow 파일 사용
- Packet replay attack에 반응하지 않음
- 원격 명령 실행 취약점 패치됨

결론 - 공격 과정 요약

- Step1 : 홈 네트워크 시스템의 구조 파악
 - gateway + wallpad
- Step2 : 공격 대상 선정(wallpad)
- Step3 : wallpad 펌웨어(소프트웨어) 획득
 - UART, Update, Flash Memory Dump
- Step4 : wallpad 분해
- Step5 : UART 연결
 - Root Shell 획득
- Step6 : 취약점 분석
 - 패킷 스니핑 + 바이너리 분석
- Step7 : 공격(Exploitation) 진행

결론 - 임베디드 장비의 보안

- BOF, FS 등의 철저한 취약점 검증 필요
- 패킷/데이터 암호화 및 Identity 확인 필요
- 리버싱 방지를 위한 난독화, 안티리버싱 적용 필요
- 하드웨어적인 보안 작업 필요 (UART, JTAG 차단)
- 언제든지 해커의 먹이가 될 수 있다는 인식 전환 필요



감사합니다.