



EE209AS Course Project: Home Occupancy Detection Via Encrypted WiFi Traffic Sniffing

Cong(Colin) Jin
ZhuoQi(George) Li

University of California, Los Angeles



Overview

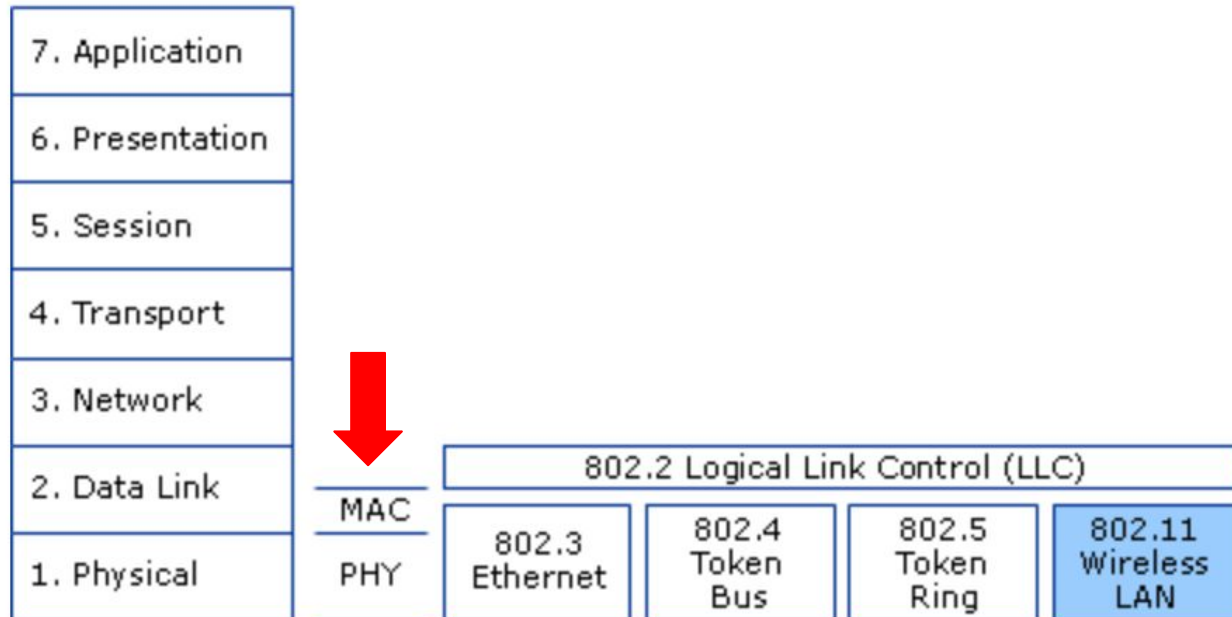
- Introduction
- Proposed Method
- Implementation
- Result
- Future Work



Introduction -- WiFi (802.11) Traffic Packets

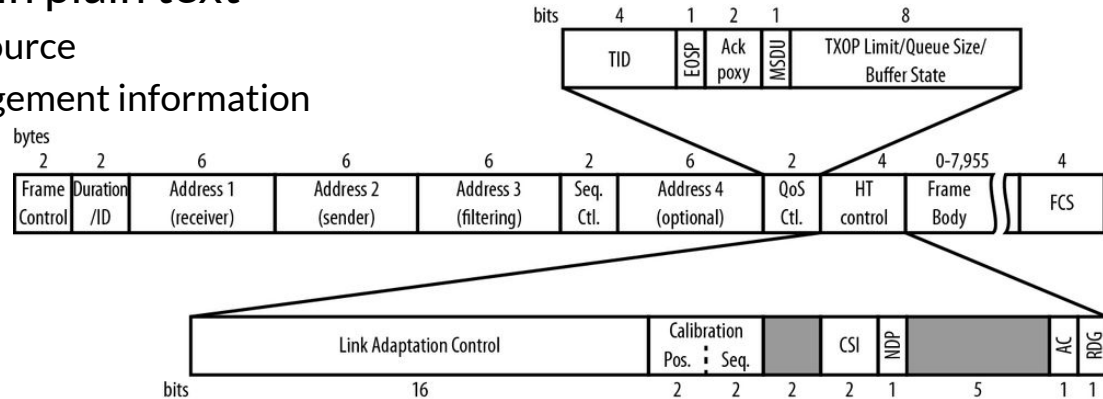
- **Smart Devices collect & exchange datas with network constantly**
 - Smart Phones, Laptop, IoT devices
- **Data usage**
 - Need to perform algorithm computation on cloud
 - Social Media Updates
 - Streaming
- **Security issues**
- **Objective: Room Occupancy Detection**

Introduction -- 802.11 & OSI model



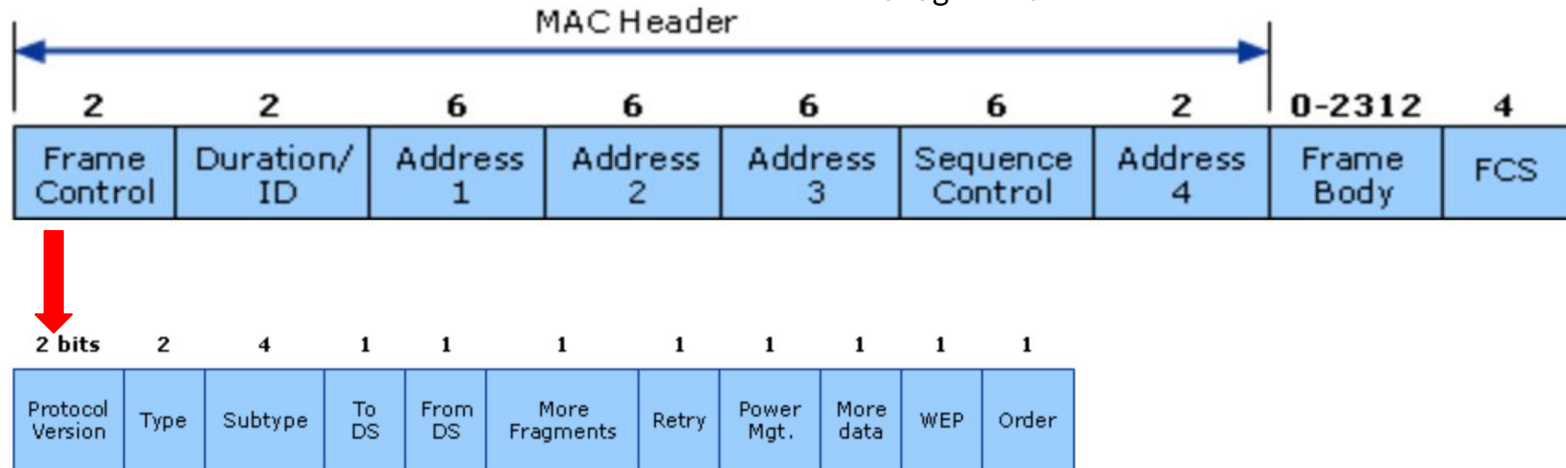
Introduction -- Information Retrieval From Encrypted Network Traffic

- Payload is encrypted (WPA2 protected)
- 802.11 frame is in plain text
 - Destination/Source
 - Control/management information
 - Packet Size
 - Time stamp



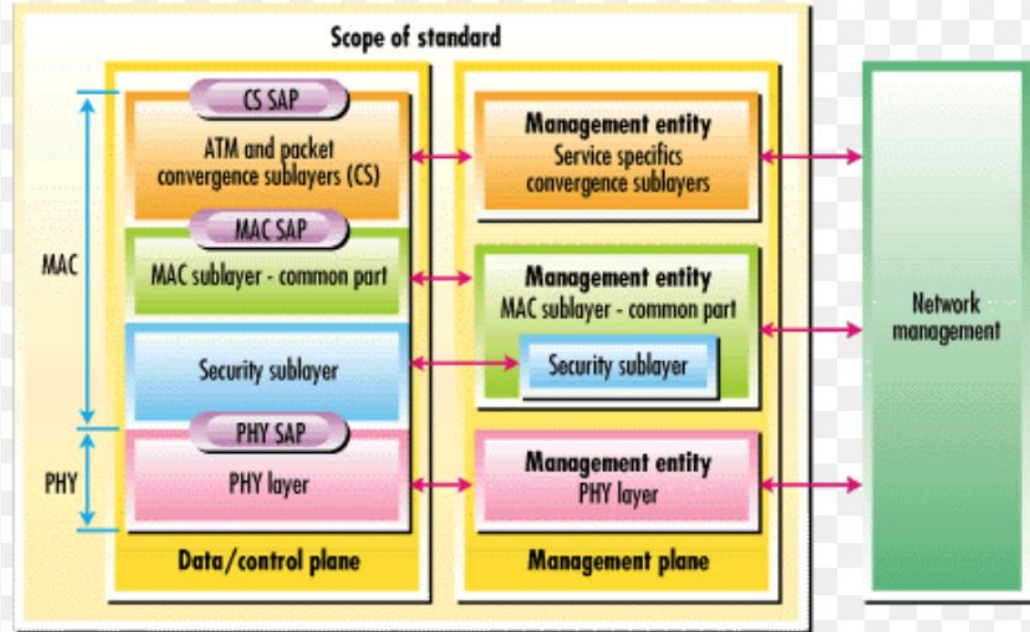
Introduction -- 802.11 MAC Frame

- MAC header
- Frame Body
- Frame Check Sequence(FCK)
- 802.11 Frame types
 - Data
 - Control
 - Management



Introduction -- Mac Layer Management Entity (MLME) Frames

- Fixed parameters
- Vendor-specific tagged parameters





Mac Layer Management Entity(MLME) Frames

- **Probe Frames**

- Sent by clients searching for Access Point(AP)
- Indicate capabilities of clients (supporting rates, authentication caps etc.)

- **Association Frames**

- Sent by clients to be added to the network by AP
- Also include client capability

- **Deauthentication Frames**

- Clients and Ap can send this frame when all communication is terminated
- Disconnect from the WiFi network



Proposed Method I - Traffic Pattern Classification

- **Motivation**

- [Deep Packet: A Novel Approach For Encrypted Traffic Classification Using Deep Learning](#)

- **Traffic Pattern Analysis**

- Traffic collection/filtering
- Feature selection/extraction
- K-means clustering
- Support Vector Machine(SVM)
- CNN



Proposed Method I - Traffic Pattern Classification

- **Why fail?**
 - In ability to capture all data packets (no Greenfield cap)
 - Input Feature Dependency
 - Network Activity Insufficient to infer house occupancy
- **Other Approach?**



Proposed Method II - Smartphone Fingerprinting

- **SmartPhone Usage Statistics**

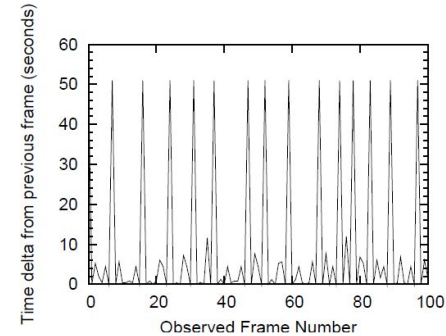
- 60% American Adults own smartphones with Wifi capability
- 90% of American adults always carry smartphone with them

- **SmartPhone usage & House Occupancy**

- Smartphones of Occupants connect with house AP automatically
- Occupants take cellphone with them when they leave the house
- Occupants do not turn off their cellphones at night and so remain in the network

Proposed Method II - Smartphone Fingerprinting

- **Active Scanning**
 - [Identifying unique devices through wireless fingerprinting](#)
 - Probe request timing analysis
- **Mac address randomization**
 - [A Study of MAC Address Randomization in Mobile Devices and When it Fails](#)



Proposed Method II - Smartphone Fingerprinting

The image displays a Wireshark packet capture window titled "rpi-cappcap [Wireshark 1.6.7 (SVN Rev 41973 from /trunk/1.6)]". The interface shows a list of captured packets in the upper pane, with packet 22 selected. The lower pane shows the details of the selected packet, which is an IEEE 802.11 Probe Request. The packet data is displayed in the bottom pane as a hex dump and ASCII representation.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	Sonos_48:85:03	Broadcast	802.11	109	Probe Request, SN=1002, FN=0, Flags=....., SSID=HHID_d8Ww4DQKca1418b5Trkbnctfca
2	0.250179000	Sonos_48:85:03	Broadcast	802.11	109	Probe Request, SN=1003, FN=0, Flags=....., SSID=HHID_d8Ww4DQKca1418b5Trkbnctfca
3	0.266192000	Sonos_5c:3b:71	Broadcast	802.11	109	Probe Request, SN=3513, FN=0, Flags=....., SSID=HHID_d8Ww4DQKca1418b5Trkbnctfca
4	0.500000000	Sonos_48:85:03	Broadcast	802.11	109	Probe Request, SN=1004, FN=0, Flags=....., SSID=HHID_d8Ww4DQKca1418b5Trkbnctfca
5	0.750094000	Sonos_48:85:03	Broadcast	802.11	109	Probe Request, SN=1005, FN=0, Flags=....., SSID=HHID_d8Ww4DQKca1418b5Trkbnctfca
6	0.765476000	Sonos_5c:3b:71	Broadcast	802.11	109	Probe Request, SN=3515, FN=0, Flags=....., SSID=HHID_d8Ww4DQKca1418b5Trkbnctfca
7	1.000224000	Sonos_48:85:03	Broadcast	802.11	109	Probe Request, SN=1007, FN=0, Flags=....., SSID=HHID_d8Ww4DQKca1418b5Trkbnctfca
8	1.250045000	Sonos_48:85:03	Broadcast	802.11	109	Probe Request, SN=1008, FN=0, Flags=....., SSID=HHID_d8Ww4DQKca1418b5Trkbnctfca
9	1.265495000	Sonos_5c:3b:71	Broadcast	802.11	109	Probe Request, SN=3518, FN=0, Flags=....., SSID=HHID_d8Ww4DQKca1418b5Trkbnctfca
10	1.500017000	Sonos_48:85:03	Broadcast	802.11	109	Probe Request, SN=1009, FN=0, Flags=....., SSID=HHID_d8Ww4DQKca1418b5Trkbnctfca
11	1.750110000	Sonos_48:85:03	Broadcast	802.11	109	Probe Request, SN=1010, FN=0, Flags=....., SSID=HHID_d8Ww4DQKca1418b5Trkbnctfca
12	1.765333000	Sonos_5c:3b:71	Broadcast	802.11	109	Probe Request, SN=3520, FN=0, Flags=....., SSID=HHID_d8Ww4DQKca1418b5Trkbnctfca
13	2.000068000	Sonos_48:85:03	Broadcast	802.11	109	Probe Request, SN=1012, FN=0, Flags=....., SSID=HHID_d8Ww4DQKca1418b5Trkbnctfca
14	2.252449000	Sonos_5c:3b:71	Broadcast	802.11	109	Probe Request, SN=1013, FN=0, Flags=....., SSID=HHID_d8Ww4DQKca1418b5Trkbnctfca
15	2.265318000	Sonos_5c:3b:71	Broadcast	802.11	109	Probe Request, SN=3522, FN=0, Flags=....., SSID=HHID_d8Ww4DQKca1418b5Trkbnctfca
16	2.500030000	Sonos_48:85:03	Broadcast	802.11	109	Probe Request, SN=1014, FN=0, Flags=....., SSID=HHID_d8Ww4DQKca1418b5Trkbnctfca
17	2.750111000	Sonos_48:85:03	Broadcast	802.11	109	Probe Request, SN=1015, FN=0, Flags=....., SSID=HHID_d8Ww4DQKca1418b5Trkbnctfca
18	2.765525000	Sonos_5c:3b:71	Broadcast	802.11	109	Probe Request, SN=3524, FN=0, Flags=....., SSID=HHID_d8Ww4DQKca1418b5Trkbnctfca
19	3.000077000	Sonos_48:85:03	Broadcast	802.11	109	Probe Request, SN=1017, FN=0, Flags=....., SSID=HHID_d8Ww4DQKca1418b5Trkbnctfca
20	3.260182000	Sonos_48:85:03	Broadcast	802.11	109	Probe Request, SN=1018, FN=0, Flags=....., SSID=HHID_d8Ww4DQKca1418b5Trkbnctfca
21	3.265347000	Sonos_5c:3b:71	Broadcast	802.11	109	Probe Request, SN=3526, FN=0, Flags=....., SSID=HHID_d8Ww4DQKca1418b5Trkbnctfca
22	3.490673000	Nintendo_9a:1f:9e	Broadcast	802.11	78	Probe Request, SN=10, FN=0, Flags=....., SSID=NETGEAR
23	3.500088000	Sonos_48:85:03	Broadcast	802.11	109	Probe Request, SN=1019, FN=0, Flags=....., SSID=HHID_d8Ww4DQKca1418b5Trkbnctfca
24	3.514713000	Nintendo_9a:1f:9e	Broadcast	802.11	78	Probe Request, SN=11, FN=0, Flags=....., SSID=NETGEAR
25	3.534994000	Nintendo_9a:1f:9e	Broadcast	802.11	78	Probe Request, SN=12, FN=0, Flags=....., SSID=NETGEAR
26	3.750172000	Sonos_48:85:03	Broadcast	802.11	109	Probe Request, SN=1020, FN=0, Flags=....., SSID=HHID_d8Ww4DQKca1418b5Trkbnctfca

Frame 22: 78 bytes on wire (624 bits), 78 bytes captured (624 bits) on interface v0, length 18
IEEE 802.11 Probe Request, Flags:

0000 00 00 12 00 2e 48 00 00 00 02 85 09 a0 00 bf 01H.....
0010 00 00 40 00 00 00 ff ff ff ff ff ff 00 21 47 9a ..@.....Ig.
0020 1f 9e ff ff ff ff ff ff a0 00 00 07 4e 45 54 47NETG
0030 45 41 52 01 08 02 04 0b 16 24 30 48 6c 32 04 0c EAB.....S0M12..
0040 12 18 60 dd 09 00 10 18 02 00 10 00 00 00
File: rpi-capp... Packets: 39 Displayed: 0 Load time: 0:00.612 Profile: Default



Proposed Method II - Smartphone Fingerprinting

- **Wifi signatures from MLME frames Tagged parameters**
 - [Passive Taxonomy of Wifi Clients using MLME Frame Contents](#)
 - Generated from 802.11 Probe & Assoc request frames from traffic
- **Distinctiveness of WiFi Signature**
 - Wifi chip specific
 - Device driver specific
 - WPA supplicant specific
 - PCB layout specific

Proposed Method II - Device Fingerprinting

```

> Frame 1033: 259 bytes on wire (2072 bits), 259 bytes captured (2072 bits)
> Radiotap Header v0, Length 25
> IEEE 802.11 Probe Response, Flags: .....C
▼ IEEE 802.11 wireless LAN management frame
  > Fixed parameters (12 bytes)
  ▼ Tagged parameters (104 bytes)
    > Tag: SSID parameter set: PSK
    > Tag: Supported Rates 1(B), 2(B), 5.5(B), 6, 9, 11(B), 12, 18, [Mbit/sec]
    > Tag: DS Parameter set : Current Channel: 11
    > Tag: Country Information: Country Code BE, Environment Any
    > Tag: QBSS Load Element 802.11e CCA Version
    > Tag: ERP Information
    > Tag: Extended Supported Rates 24, 36, 48, 54, [Mbit/sec]
    > Tag: Cisco CCX1 CKIP + Device Name
    > Tag: Cisco Unknown 96: Tag 150 Len 6
    ▼ Tag: Vendor Specific: Microsoft: WPA Information Element
      Tag Number: Vendor Specific (221)
      Tag length: 24
      OUI: 00-50-f2 (Microsoft)
      Vendor Specific OUI Type: 1
      Type: WPA Information Element (0x01)
      WPA Version: 1
      > Multicast Cipher Suite: 00-50-f2 (Microsoft) TKIP
      Unicast Cipher Suite Count: 1
      > Unicast Cipher Suite List 00-50-f2 (Microsoft) PSK
      Auth Key Management (AKM) Suite Count: 1
      > Auth Key Management (AKM) List 00-50-f2 (Microsoft) PSK
    > Tag: Vendor Specific: Microsoft: WMM/WME: Parameter Element
    > Tag: Vendor Specific: Aironet: Aironet Unknown
    > Tag: Vendor Specific: Aironet: Aironet CCX version = 5

```



Proposed Method II - Device Fingerprinting

LG G4

```
wifi4|probe:0,1,3,45,127,107,191,221(506f9a,16),                221(0
01018,2),221(00904c,51),221(00904c,4),221(0050f2,8),htcap:016f,htag
g:17,htmcs:000000ff,vhtcap:0f805932,vhtrxmcs:0000fffe,vhttxmcs:0000
fffe,extcap:0000088001400040|assoc:0,1,33,36,48,    45,127,191,221(0
01018,2),221(00904c,4),221(0050f2,2),htcap:016f,htagg:17,htmcs:0000
00ff,vhtcap:0f805932,vhtrxmcs:0000fffe,vhttxmcs:0000fffe,txpow:1d01
,extcap:0000008001400040
```

iPhone 6

```
wifi4|probe:0,1,    45,127,107,191,                221(0050f2,8),221(0
01018,2),                htcap:0063,htag
g:17,htmcs:000000ff,vhtcap:0f805032,vhtrxmcs:0000fffe,vhttxmcs:0000
fffe,extcap:0400088400000040|assoc:0,1,33,36,48,70,45,127,191,221(0
01018,2),                221(0050f2,2),htcap:0063,htagg:17,htmcs:0000
00ff,vhtcap:0f805032,vhtrxmcs:0000fffe,vhttxmcs:0000fffe,txpow:e002
,extcap:0400000000000040
```




Proposed Method II - Device Fingerprinting

iPhone 5

wifi4|probe:0,1,45,127,107,221(001018,2),221(00904c,51),221(0050f2,8),htcap:0062,htag:1a,htmcs:000000ff,extcap:00000004|assoc:0,1,33,36,48,45,221(001018,2),221(00904c,51),221(0050f2,2),htcap:0062,htag:1a,htmcs:000000ff,txpow:1504

iPhone 5s

wifi4|probe:0,1,45,127,107,221(001018,2),221(00904c,51),221(0050f2,8),htcap:0062,htag:1a,htmcs:000000ff,extcap:00000804|assoc:0,1,33,36,48,45,221(001018,2),221(00904c,51),221(0050f2,2),htcap:0062,htag:1a,htmcs:000000ff,txpow:1603

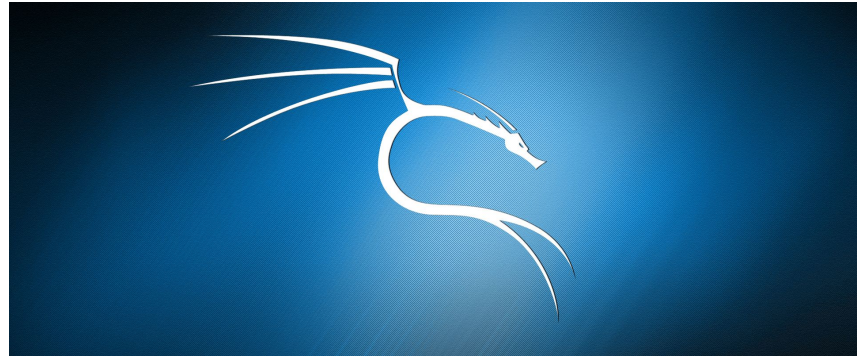
Implementation - Hardware Configuration

- Rpi + Wifi Adaptor with monitor mode and packets injection capability
- Monitor mode
- Packet Injection

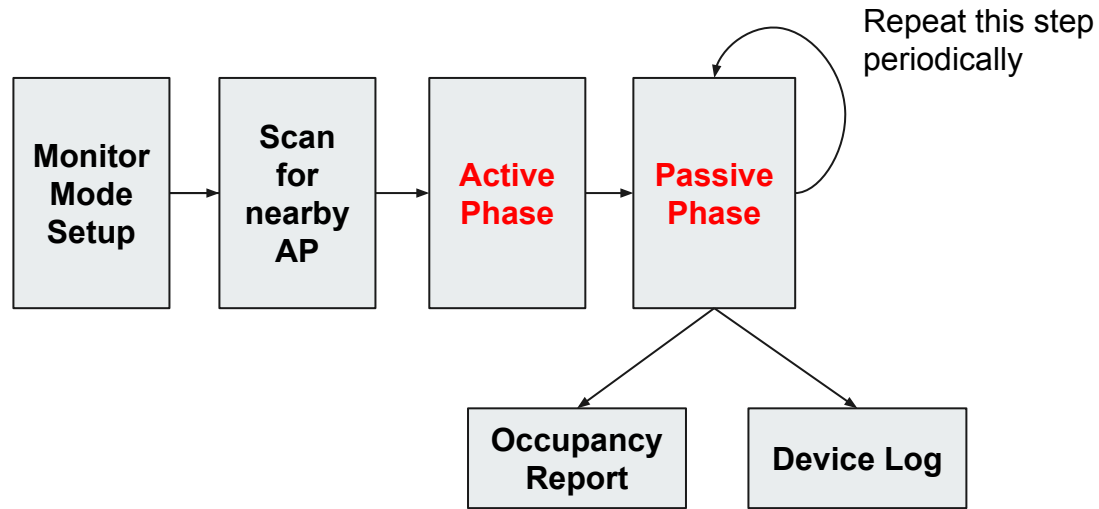


Implementation - Software Support

- **Operating System**
 - Kali Linux
- **Packet Dump & Inspection**
 - Wireshark
 - Tcpdump
- **Packet Injection and AP scanning**
 - airodump-ng
 - aireplay-ng
- **Project development**
 - Pycharm
 - Git



Implementation - Overview





Implementation - Setup

- **Monitor mode setup**
- **Scan for nearby AP**
 - Airodump-ng
 - Select target using SNR strength
 - Record AP mac address and listening channel

- What happens upon connection
- Deauthentication Attack
- Scan for devices - Null data frames

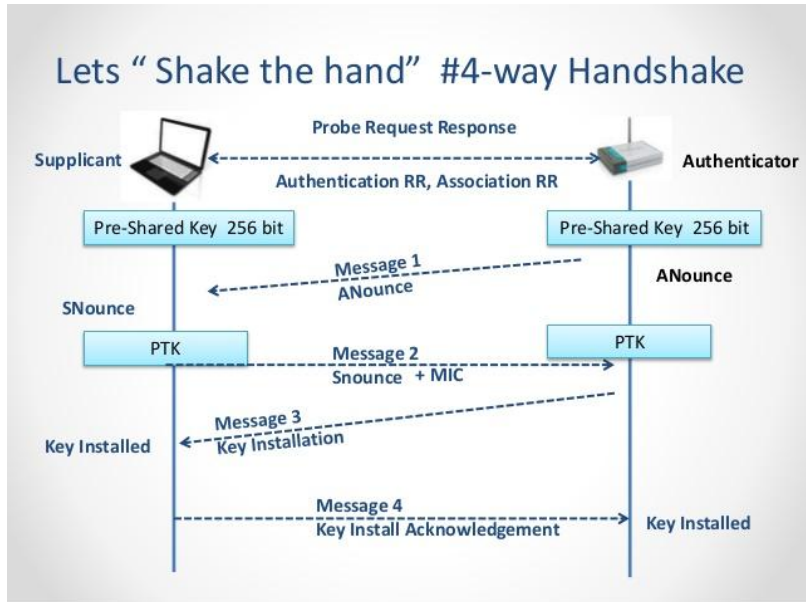
74	12/5/20/9/42/893901	3.650514	4	10	-51	24		Intel 11:A8:02		802.11 acknowledgement
75	12/5/20/9/42/893941	3.676254	4	-26	-23	5	Intel 11:A8:02	00:13:7F:43:25:60	00:13:7F:43:25:60	802.11 QoS data null function
76	12/5/20/9/42/893954	3.676467	4	10	-28	5	Intel 11:A8:02			802.11 acknowledgement
77	12/5/20/9/42/893956	3.750268	8	177	-28	1	00:13:7F:43:25:60	FF:FF:FF:FF:FF:FF	00:13:7F:43:25:60	802.11 beacon
78	12/5/20/9/42/893949	3.750562	8	26	-20	5	Intel 11:A8:02	00:13:7F:43:25:60	00:13:7F:43:25:60	802.11 QoS data null function
79	12/5/20/9/42/894172	3.750785	8	10	-29	5	Intel 11:A8:02			802.11 acknowledgement
80	12/5/20/9/42/894327	3.750940	8	94	-35	54	00:13:7F:43:25:60	Intel 11:A8:02	00:13:7F:43:25:60	ICMP data
81	12/5/20/9/42/894361	3.750974	8	10	-44	24	00:13:7F:43:25:60	00:13:7F:43:25:60	802.11 acknowledgement	
82	12/5/20/9/42/894018	3.776794	8	26	-20	5	Intel 11:A8:02	00:13:7F:43:25:60	00:13:7F:43:25:60	802.11 QoS data null function
83	12/5/20/9/42/894034	3.777007	8	10	-30	5	Intel 11:A8:02			802.11 acknowledgement
84	12/5/20/9/42/894057	3.852670	8	177	-29	1	00:13:7F:43:25:60	FF:FF:FF:FF:FF:FF	00:13:7F:43:25:60	802.11 beacon
85	12/5/20/9/42/894101	3.949078	8	177	-39	1	00:13:7F:43:25:60	FF:FF:FF:FF:FF:FF	00:13:7F:43:25:60	802.11 beacon

Network media information

802.11 MAC header:

- frame control
 - protocol version: 0
 - frame type: 0 Data
 - subtype: 1200 QoS data null function
- frame control
 - to DS: Yes
 - from DS: No
 - more frag: No
 - retry: Not Awaiting
 - power management: PS mode
 - more data: No more data
 - WEP/Protected Frame: Non-Protected Frame
 - order: Non-Strict Order

802.11 QoS data null function



Implementation - Active Phase

- Build wifi signature
- Identify device
 - Signature hamming distance
 - Find match in database

```
# first Perform device scanning
device_list = device_tracking(Ap_addr, channel, duration='180')
```

```
# for each device, launch deauth attack and capture 4-way
handshake moment to build wifi signature
```

```
for dev in device_list:
    deauth(mon_card, target=dev, AP=Ap_addr)
    passive_tracking(sig_stats, ap_addr=Ap_addr,
duration='120', mode=0)
```

Run in parallel

```
def passive_tracking(...):
```

```
    packets = collect_packets(duration)
```

```
    signature = build_sig(packets)
```

```
    device = database_match(db_file, signature)
```

```
    ...
```

Sniff packets for a fixed duration

Build signature from captured packets

Find matched device in database

Implementation - Passive Phase

- Passively listen for any new connection
- Log any new device in the network
- Generate a status report every 30 minutes period (default)
- Monitor active devices in the network

```
# update_fre: how many monitor cycle between adjacent updates
# running time: (passive_dur/60)*update_fre*period minutes
def passive_phase(ap, sig_stats, passive_dur='300', period=10, update_fre=6):
    # loop for passive monitoring
    for i in range(period):
        for j in range(update_fre-1):
            # perform passive tracking every passive_dur/60 minute
            passive_tracking(sig_stats, ap, duration=passive_dur, mode=0)
        # Also scan for current devices in network
        dev_list = passive_tracking(sig_stats, ap, duration=passive_dur,
mode=1)

    # Update active devices accordingly
    check_devices(dev_list, sig_stats)
    ...
```

Loop for passive monitoring

Passive listening subperiod, stop listen and analyze packets every 5 minutes (default)



Implementation - Database Match

- **Built a small signature database with our own devices**

- Smartphone: Iphone7, iphone7 plus, iphone 6, Huawei Android
- Laptop: Macbook Pro, Huawei
- IoT Device: Google Home

iPhone 5

```
wifi4|probe:0,1,45,127,107,221(001018,2),221(00904c,51),221(0050f2,8),htcap:0062,htag:1a,htmcs:000000ff,extcap:00000004|assoc:0,1,33,36,48,45,221(001018,2),221(00904c,51),221(0050f2,2),htcap:0062,htag:1a,htmcs:000000ff,txpow:1504
```

- **Signature Comparison**

- Signature Hamming Distance Calculation

iPhone 5s

```
wifi4|probe:0,1,45,127,107,221(001018,2),221(00904c,51),221(0050f2,8),htcap:0062,htag:1a,htmcs:000000ff,extcap:000000804|assoc:0,1,33,36,48,45,221(001018,2),221(00904c,51),221(0050f2,2),htcap:0062,htag:1a,htmcs:000000ff,txpow:1603
```



Result

- [Occupancy 3/18/2018](#)
- [Device log 3/18/2018](#)
- [Occupancy 3/19/2018](#)
- [Device log 3/19/2018](#)
- [Occupancy 3/20/2018](#)
- [Device log 3/20/2018](#)



Future Work

- **Packet Collection Improvement**
 - Get packets directly from kernel
 - Process packets right after capture
 - No need to have subperiods to collect packets for analysis
 - Need to write customized C code instead of using Tcpdump
- **Test with a much larger database**
 - How fast does it find a match
 - Match Accuracy



Reference

- Shield: vulnerability-driven network filters for preventing known vulnerability exploits
- Network Traffic Classification using Support Vector Machine and Artificial Neural Network
- Deep Packet: A Novel Approach For Encrypted Traffic Classification Using Deep Learning
- Spying on the Smart Home: Privacy Attacks and Defenses on Encrypted IoT Traffic
- Is Anybody Home? Inferring Activity From Smart Home Network Traffic
- A Study of MAC Address Randomization in Mobile Devices and when it Fails
- On Security Vulnerabilities of Null Data Frames in IEEE 802.11 based WLANs
- Passive Taxonomy of Wifi Clients using MLME Frame Contents
- Passive Data Link Layer 802.11 Wireless Device Driver FingerPrinting
- Identify Unique Devices through Wireless FingerPrinting