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

Cong(Colin) Jin
ZhuoQi(George) Li

University of California, Los Angeles

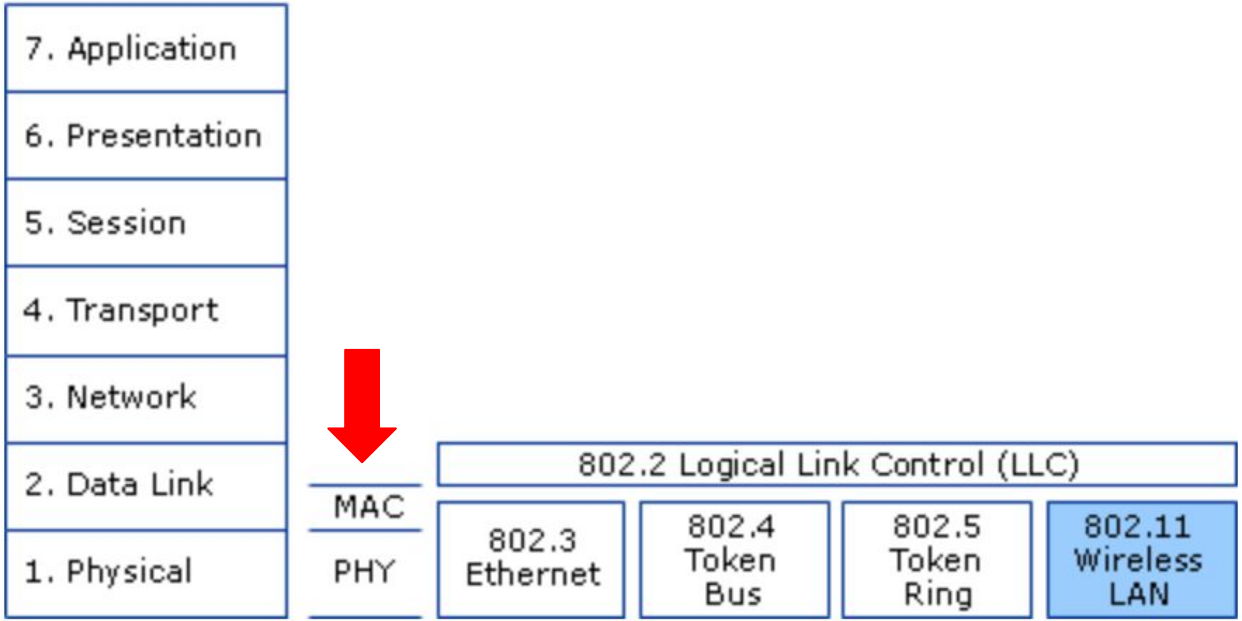
Overview

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

Introduction -- WiFi (802.11) Traffic Packets

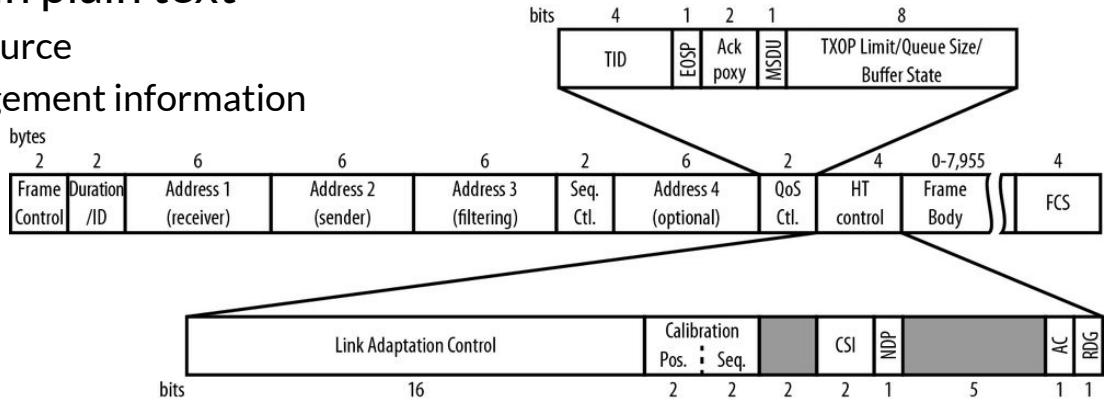
- **Smart Devices collect & exchange datas with network constantly**
  - Smart Phones, Laptop, IoT devices
- **Data usage**
  - Need to perform algorithm computation on cloud
  - Social Media Updates
  - Streaming
- **Security issues**
- **Objective: Room Occupancy Detection**

Introduction -- 802.11 & OSI model



Introduction -- Information Retrieval From Encrypted Network Traffic

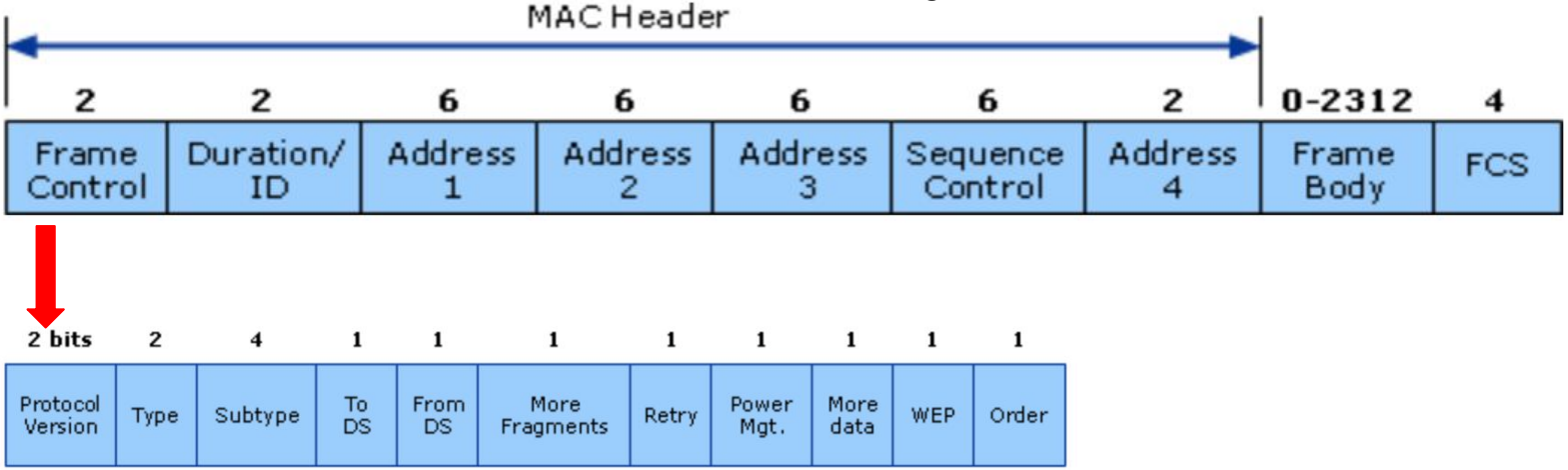
- Payload is encrypted (WPA2 protected)
- 802.11 frame is in plain text
  - Destination/Source
  - Control/management information
  - Packet Size
  - Time stamp



Introduction -- 802.11 MAC Frame

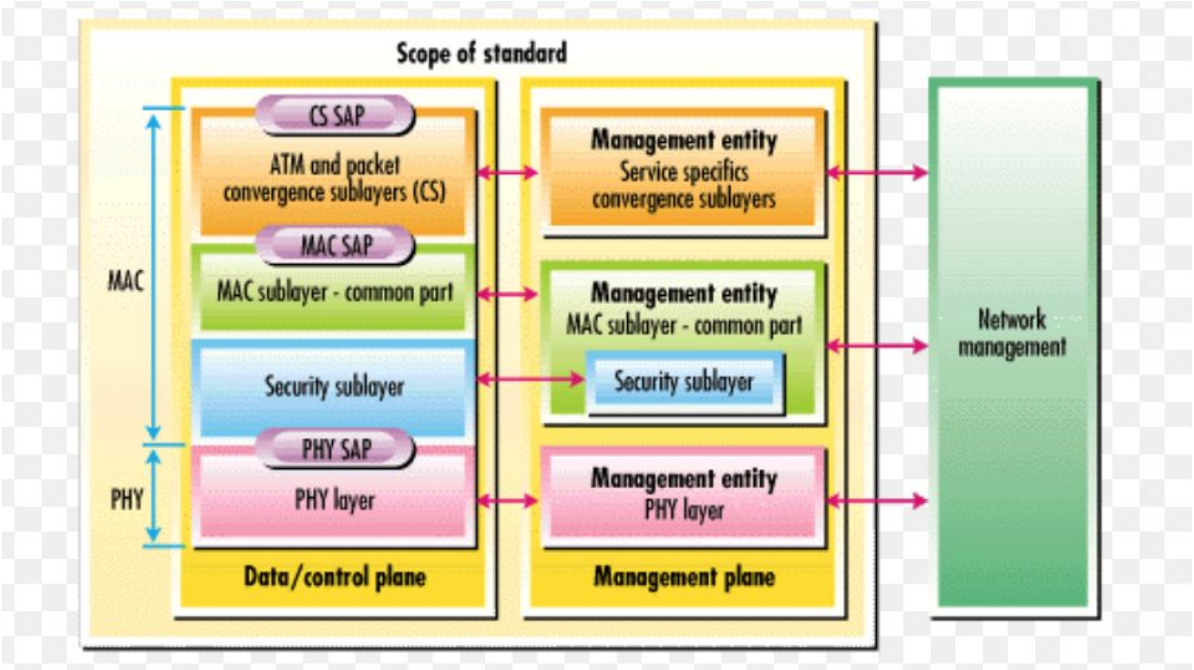
- MAC header
- Frame Body
- Frame Check Sequence(FCK)

- 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

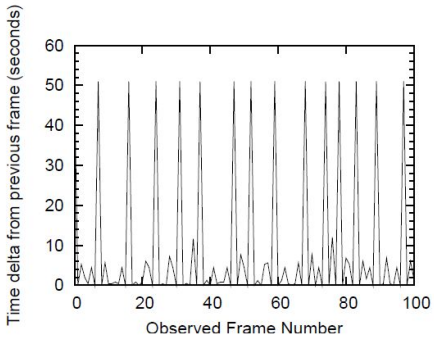
- **Why fail?**
  - In ability to capture all data packets (no Greenfield cap)
  - Input Feature Dependency
  - Network Activity Insufficient to infer house occupancy
- **Other Approach?**

Proposed Method II - Smartphone Fingerprinting

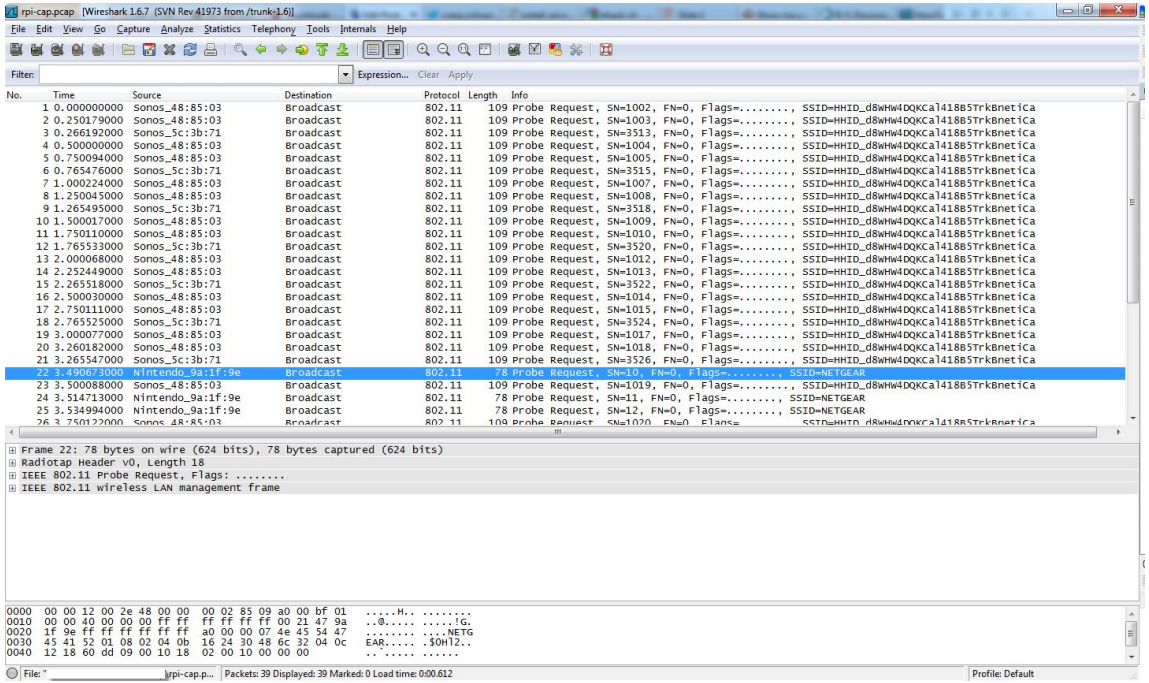
- **SmartPhone Usage Statistics**
  - 60% American Adults own smartphones with Wifi capability
  - 90% of American adults always carry smartphone with them
- **SmartPhone usage & House Occupancy**
  - Smartphones of Occupants connect with house AP automatically
  - Occupants take cellphone with them 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



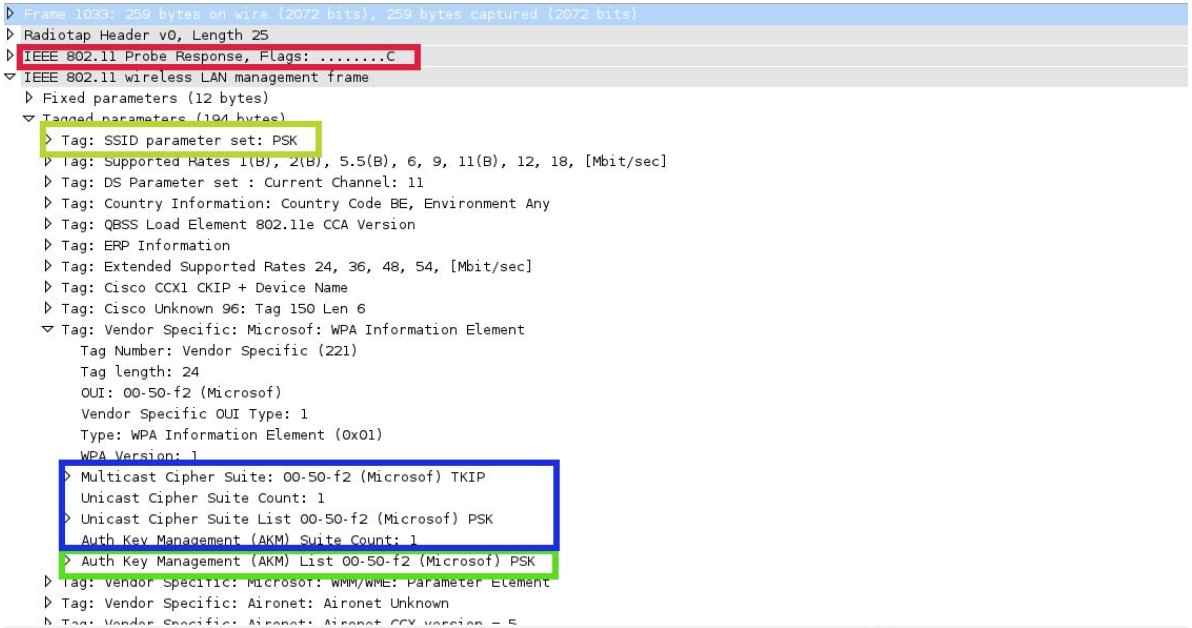
Proposed Method II - Smartphone Fingerprinting



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



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:0**1**6**f**,htag
g:17,htmcs:000000ff,vhtcap:0f805**9**32,vhtrxmcs:0000fffe,vhttxmcs:0000
fffe,extcap:0**0**00088**0**0**14**00040|assoc:0,1,33,36,48, 45,127,191,221(0
01018,2),**221(00904c,4),**221(0050f2,2),htcap:0**1**6**f**,htagg:17,htmcs:0000
00ff,vhtcap:0f805**9**32,vhtrxmcs:0000fffe,vhttxmcs:0000fffe,txpow:**1d**0**1**
,extcap:0**0**0000**8**00**14**00040

iPhone 6

wifi4|probe:0,1, 45,127,107,191, **221(0050f2,8),**221(0
01018,2), htcap:0**0**6**3**,htag
g:17,htmcs:000000ff,vhtcap:0f805**0**32,vhtrxmcs:0000fffe,vhttxmcs:0000
fffe,extcap:0**4**00088**4**0**00**00040|assoc:0,1,33,36,48,**70,**45,127,191,221(0
01018,2), 221(0050f2,2),htcap:0**0**6**3**,htagg:17,htmcs:0000
00ff,vhtcap:0f805**0**32,vhtrxmcs:0000fffe,vhttxmcs:0000fffe,txpow:**e0**0**2**
,extcap:0**4**0000**0**00**00**00040

Proposed Method II - Device Fingerprinting

iPhone 5

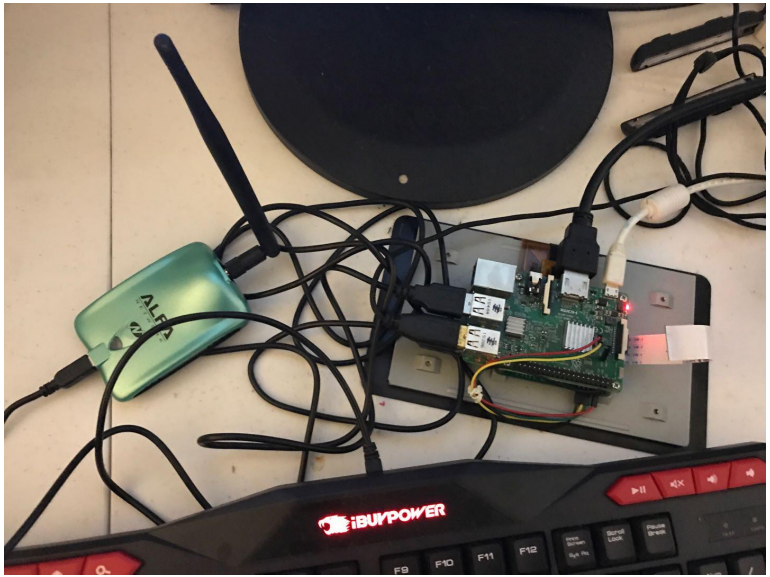
```
wifi4|probe:0,1,45,127,107,221(001018,2),221(00904c,51),221(0050f2,
8),htcap:0062,htagg:1a,htmcs:000000ff,extcap:00000004|assoc:0,1,33,
36,48,45,221(001018,2),221(00904c,51),221(0050f2,2),htcap:0062,htag
g:1a,htmcs:000000ff,txpow:1504
```

iPhone 5s

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

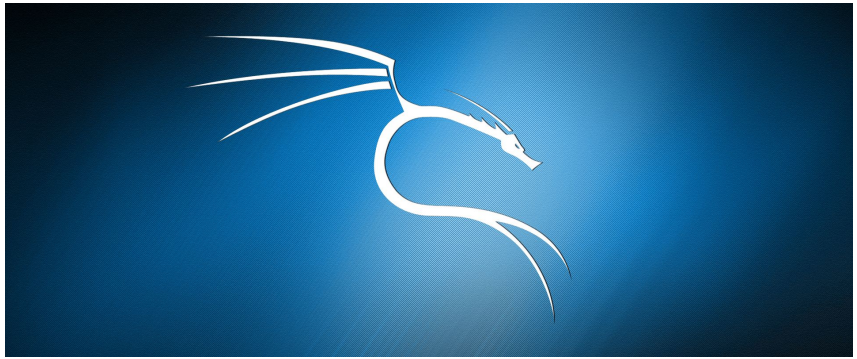
Implementation - Hardware Configuration

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

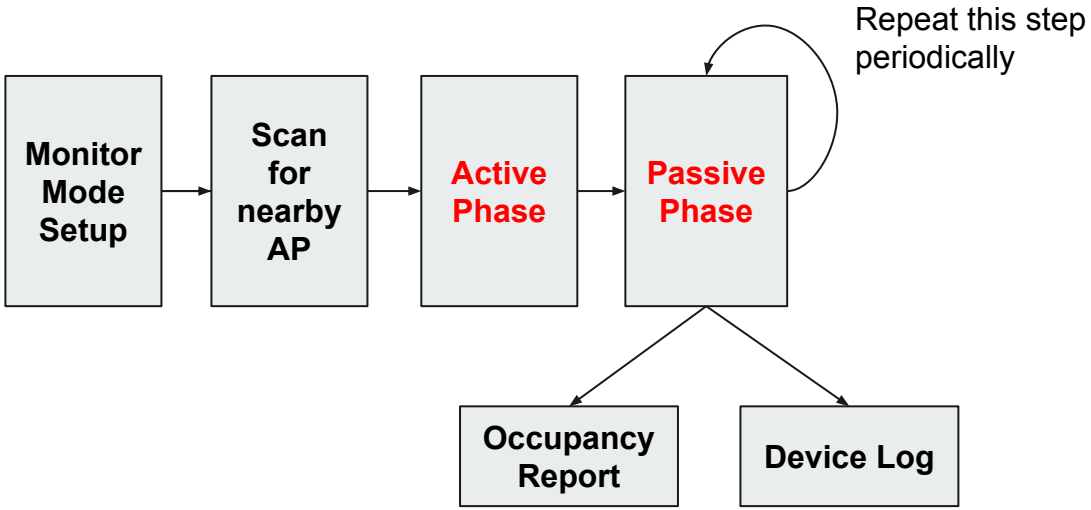


Implementation - Software Support

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



Implementation - Overview

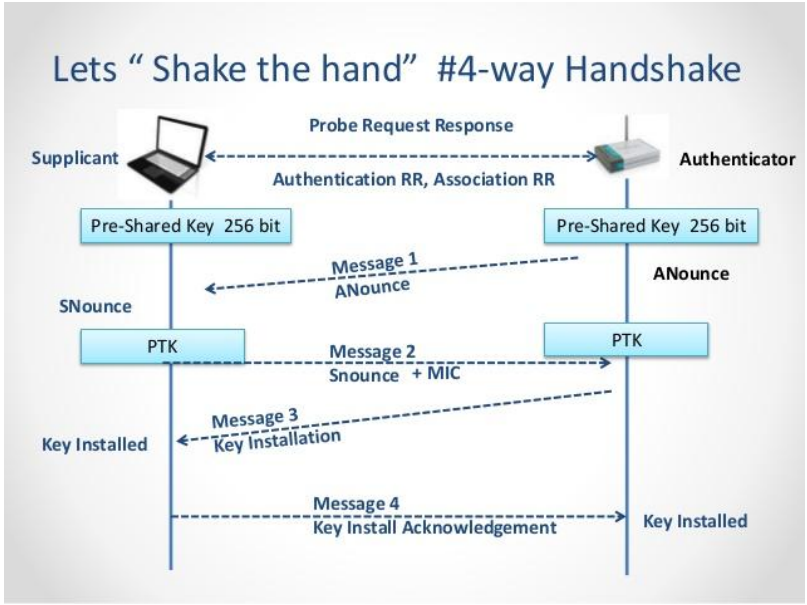


Implementation - Setup

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

Implementation - Active Phase

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



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)

def passive_tracking(...):
        packets = collect_packets(duration)
        signature  = build_sig(packets)
        device = database_match(db_file, signature)
        ...
```

Run in parallel

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

- **Signature Comparison**
  - Signature Hamming Distance Calculation

iPhone 5

```
wifi4|probe:0,1,45,127,107,221(001018,2),221(00904c,51),221(0050f2,
8),htcap:0062,htagg:1a,htmcs:000000ff,extcap:00000004|assoc:0,1,33,
36,48,45,221(001018,2),221(00904c,51),221(0050f2,2),htcap:0062,htag
g: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,htagg:1a,htmcs:000000ff,extcap:00000804|assoc:0,1,33,
36,48,45,221(001018,2),221(00904c,51),221(0050f2,2),htcap:0062,htag
g: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