UC RIVERSIDE

KU LEUVEN

AirSnitch: Breaking Client Isolation in Wi-Fi Networks

Xin'an Zhou, Juefei Pu, Zhutian Liu, Zhiyun Qian,
Zhaowei Tan, Srikanth Krishnamurthy, **Mathy Vanhoef**

Contributions

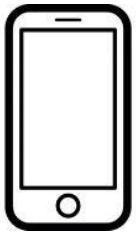
Client isolation should prevent attacks from malicious insiders. We bypass this defense at several layers:

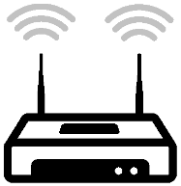
1. Wi-Fi layer → Abusing shared group keys
2. IP layer → Gateway Bouncing
3. Ethernet layer → Port Stealing

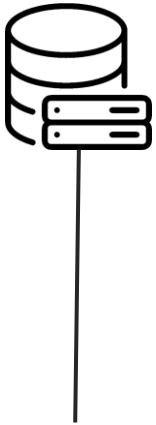
…with combinations & tricks to obtain full MitM!

→ All tested devices/networks had at least one flaw!

Authentication & keys



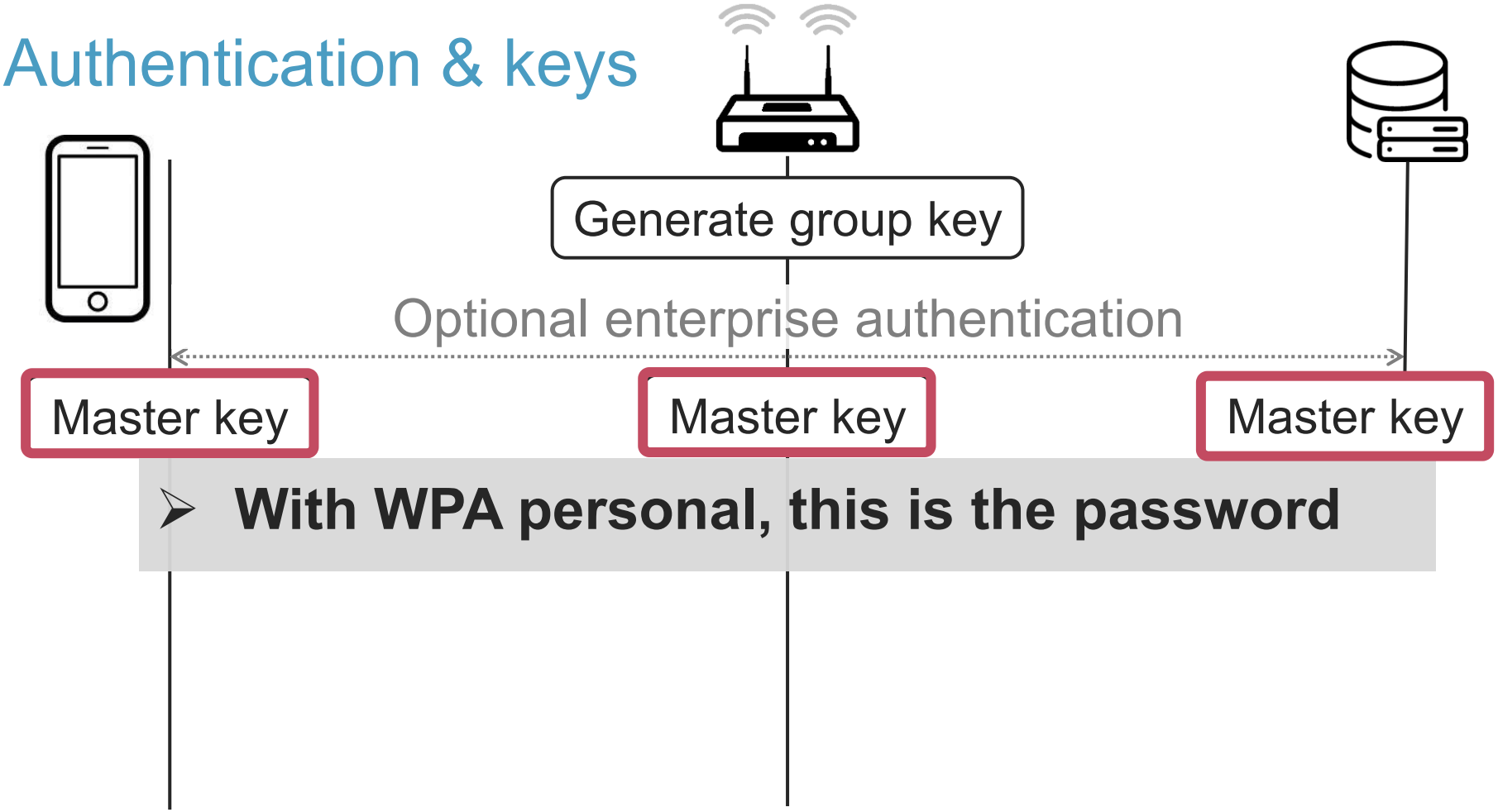




Generate group key

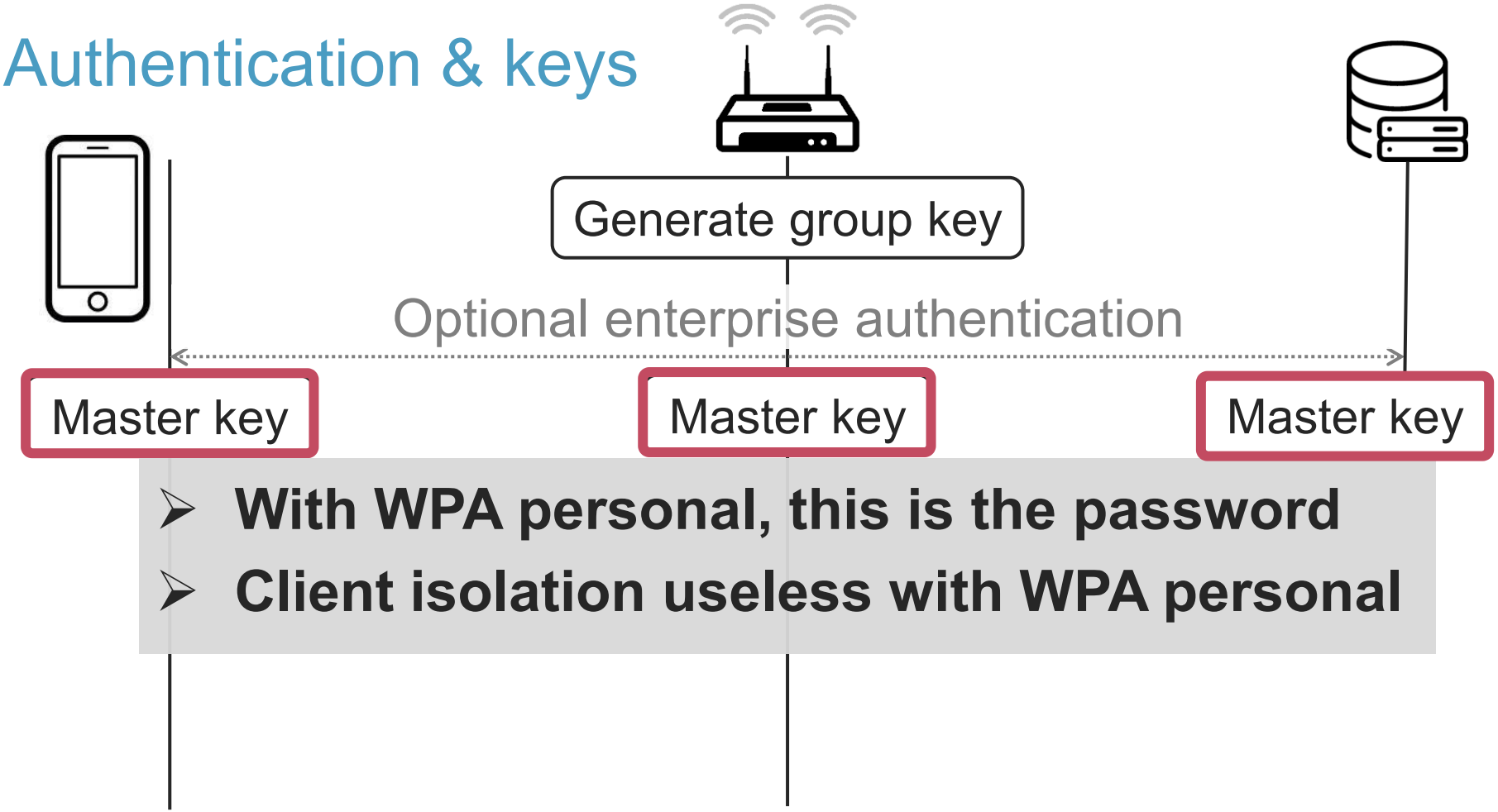
Encrypts broadcast and multicast frames

Authentication & keys



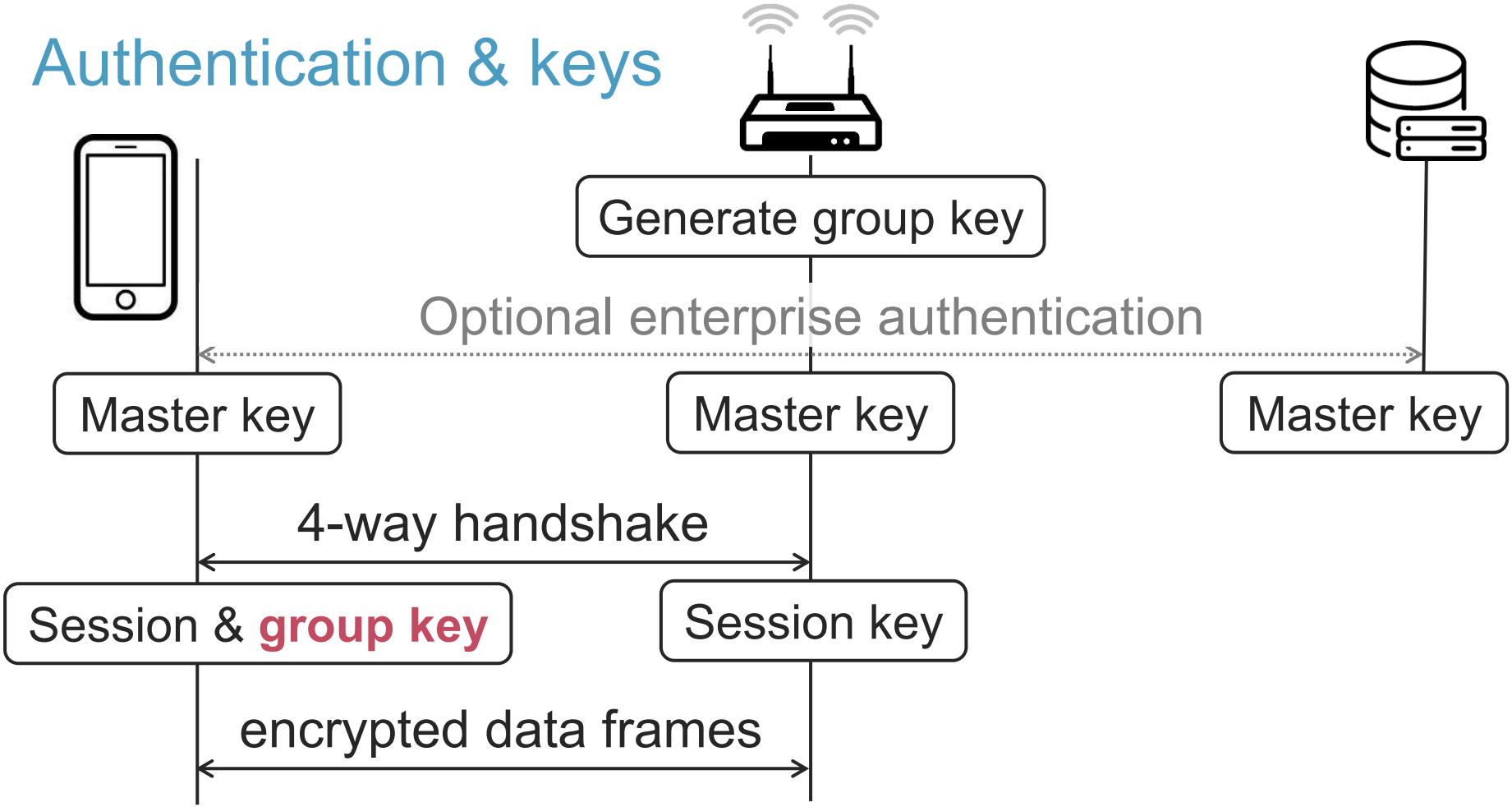
- **With WPA personal, this is the password**

Authentication & keys



- **With WPA personal, this is the password**
- **Client isolation useless with WPA personal**

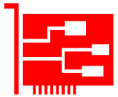
Authentication & keys



Complexity of Wi-Fi

Several Basic Service Sets (BSSs) = “logical APs”

- Each advertises one SSID

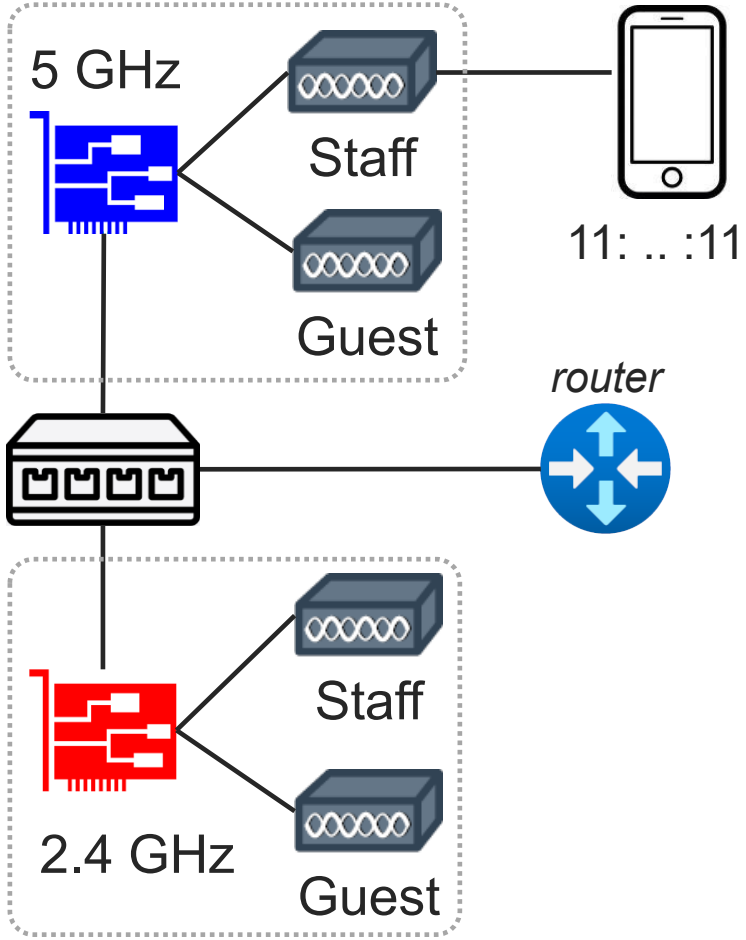


Several NICs

- To cover multiple channels

Several switches

- Interconnect components

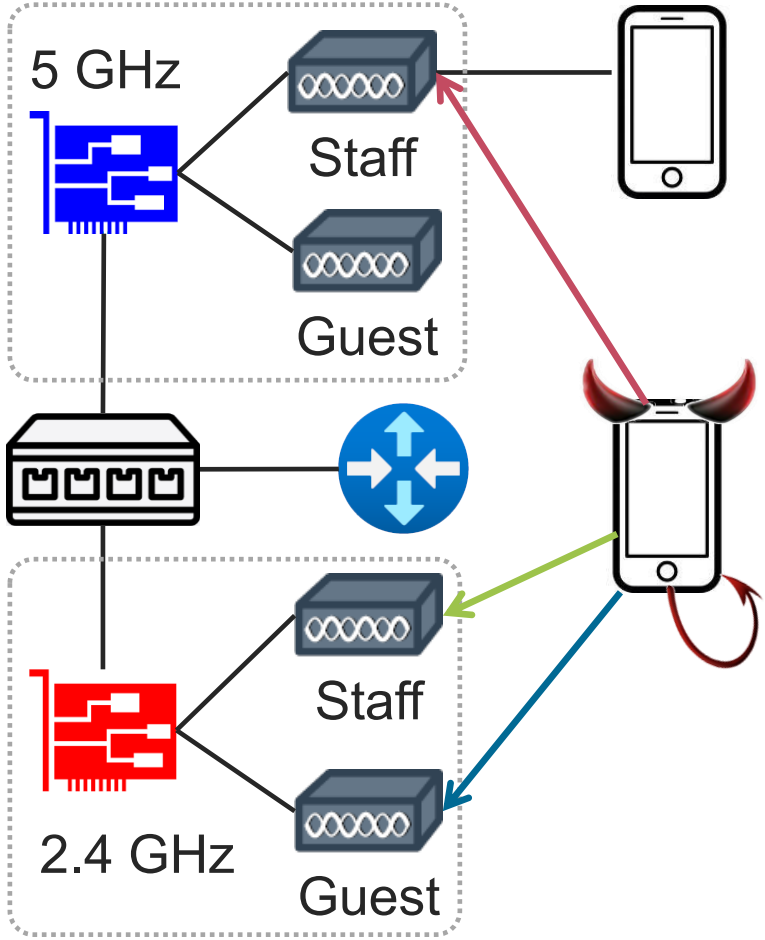


Malicious insiders

We'll consider malicious insiders

- Have legitimate access to network
- Adversary can be in **same BSS**
- Adversary can be in **different BSS**
- Can even be in **another SSID**

→ Malicious insider can directly attack victims, obtain MitM, etc.



Impact of malicious insiders

Why care about Wi-Fi isolation if we have TLS / HTTPS / …?

- Prevents exploitation of IoT devices, old internal servers, etc.

- **Real-world Wi-Fi insider attacks** detected by Volexity [1]!
- Fingerprint HTTPS to identify page being visited [2].

[1] https://www.volexity.com/blog/2024/11/22/the-nearest-neighbor-attack-how-a-russian-apt-weaponized-nearby-wi-fi-networks-for-covert-access/
[2] Shen, M., Liu, Y., Zhu, L., Du, X., & Hu, J. (2020). Fine-grained webpage fingerprinting using only packet length information of encrypted traffic. *IEEE Trans. on Information Forensics and Security*.

Wi-Fi Client Isolation

History: `ap_isolate` on Linux

Kernel config to not bridge traffic between clients in a BSS (2010)

- *“Is this useful?”*
- *“Yes [..] it can **save a lot of airtime** by not forwarding every broadcast message” [1]*

→ *Later **reused as basis for Wi-Fi client isolation***

[1] Source: https://www.spinics.net/lists//linux-wireless/msg50019.html

No standard, created by vendors

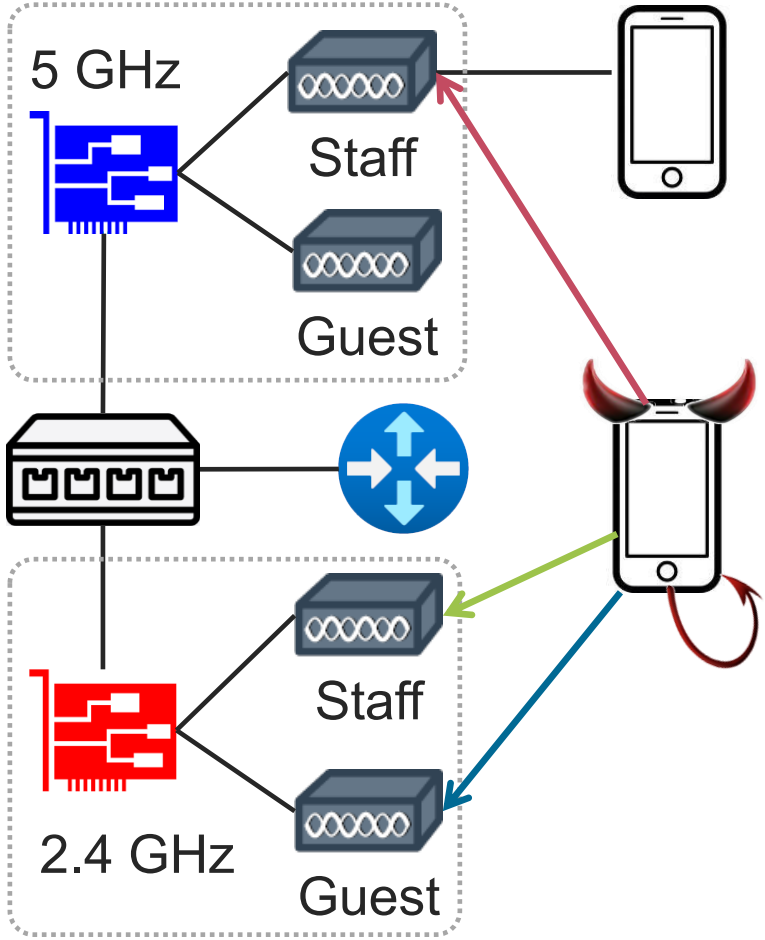
Added by vendors, not part of the IEEE 802.11 standard

- Prohibits clients from communicating with each other
- Many synonyms: AP isolation, Peer-to-peer (P2P) blocking, Layer-2 (L2) isolation, etc.
- Every vendor's implementation may have different security guarantees…

Types of client isolation

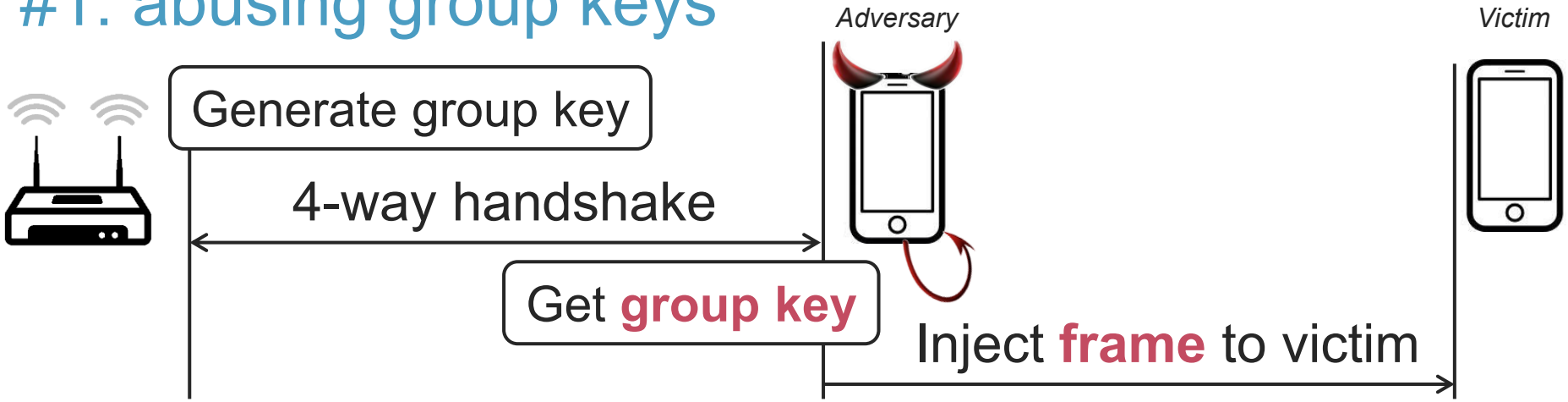
- **Intra-BSSID**: between clients on the same AP, i.e., the **same BSS**
- **Inter-BSSID**: between clients on different APs, but the **same SSID**
- Between **different SSIDs**, e.g., the SSIDs ‘guest’ and ‘staff’

→ Must ensure isolation **<u>at all layers</u>** to guarantee proper security!



LET'S GET HACKING

#1: abusing group keys



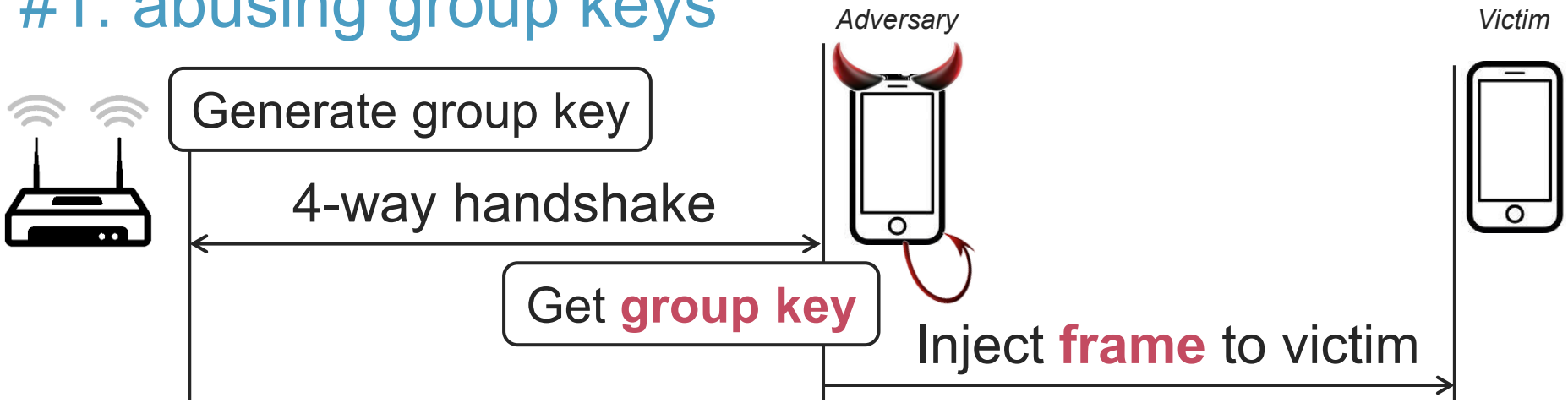
frame = Dot11(dst = ff:ff:ff:ff:ff:ff, src =) / IP(dst = , src =)

Broadcast Wi-Fi frame

Unicast IP packet

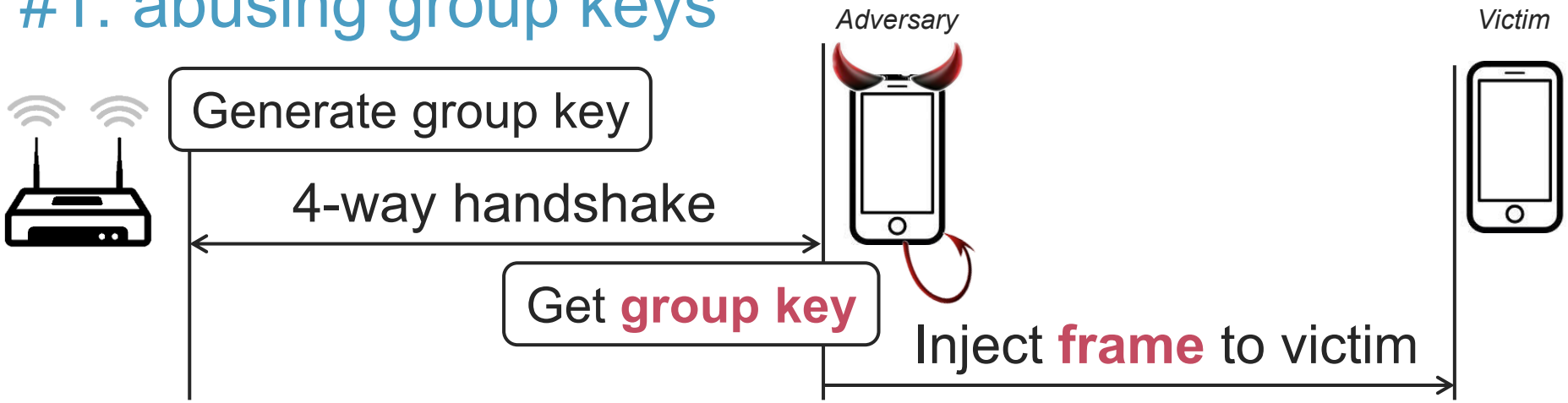
Encrypted using the shared **group key**

#1: abusing group keys



- Group key doesn't rotate often: attack typically remains possible even after user is removed
- Limitation: can only be used to inject frames (not intercept)

#1: abusing group keys



Attack is similar to the Hole-196 (Black Hat ‘10)

- Hole-196 abused group key to bypass IDS & ARP poison
- We use it to bypass client isolation ☺

Defenses

- AP: give each client a **randomized group key**
  - This effectively disables broadcast/multicast traffic
  - Adversary won't know which key the victim has, can't inject frame
  - Combine with Proxy ARP service to ensure network connectivity
- Client: **drop unicast IP packets in broadcast** Wi-Fi frames
  - Mitigates several attacks, but injecting broadcast IP remains possible

Application to Passpoint = Hotspot 2.0

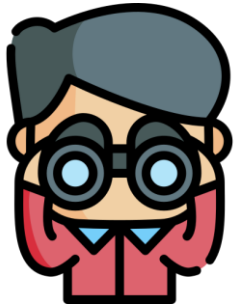
Passpoint is a specification to secure protected hotspots

- Recommended to randomize group keys!

> *"Shall set to a unique **random value the value of the GTK** employed in the 4-Way Handshake [..]"*

- Doesn't randomize Group Key (GTK) in *other* handshakes
- Also forgot to randomize the Integrity Group key (IGTK)
  - Reenables attacks, see paper for details

Attack #2: router/gateway bouncing

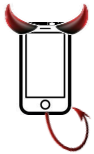
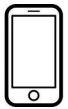


Finding: many vendors only isolate clients at the Wi-Fi / Ethernet layer

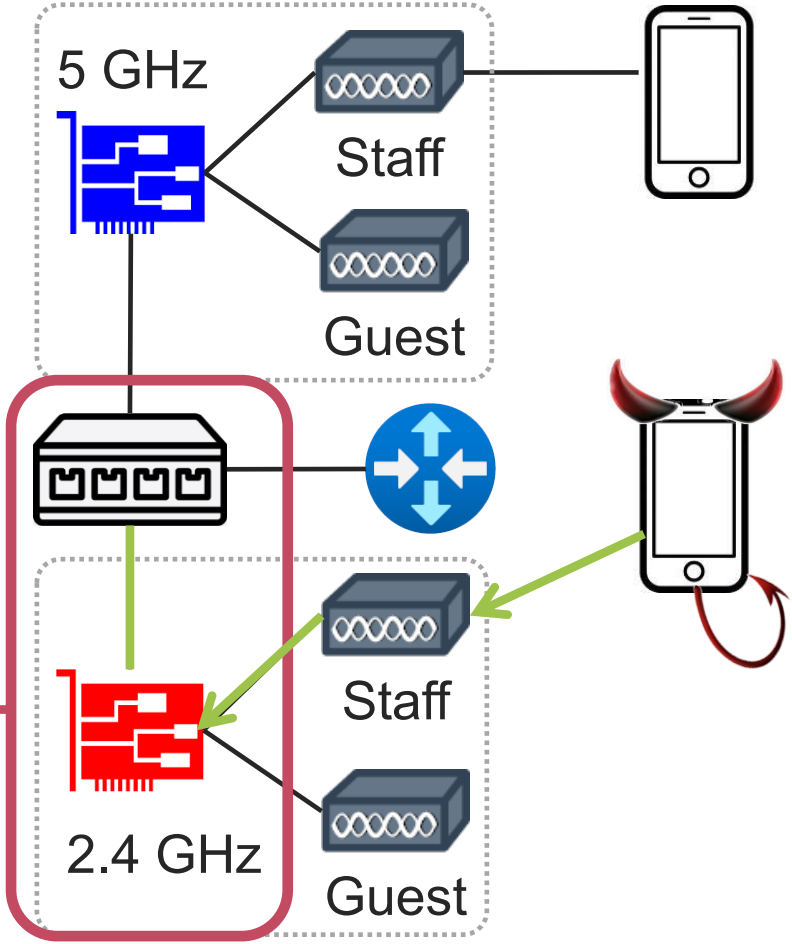
Realization: client isolation can be **bypassed at the IP layer**!

Normal client-to-client frame

Ethernet header has the victim MAC address as destination:

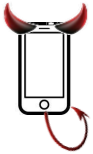
Ether(src =  , dst = ) / IP(...)

This is **blocked by the NIC or switch** due to client isolation.

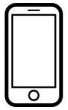


Router/gateway bouncing

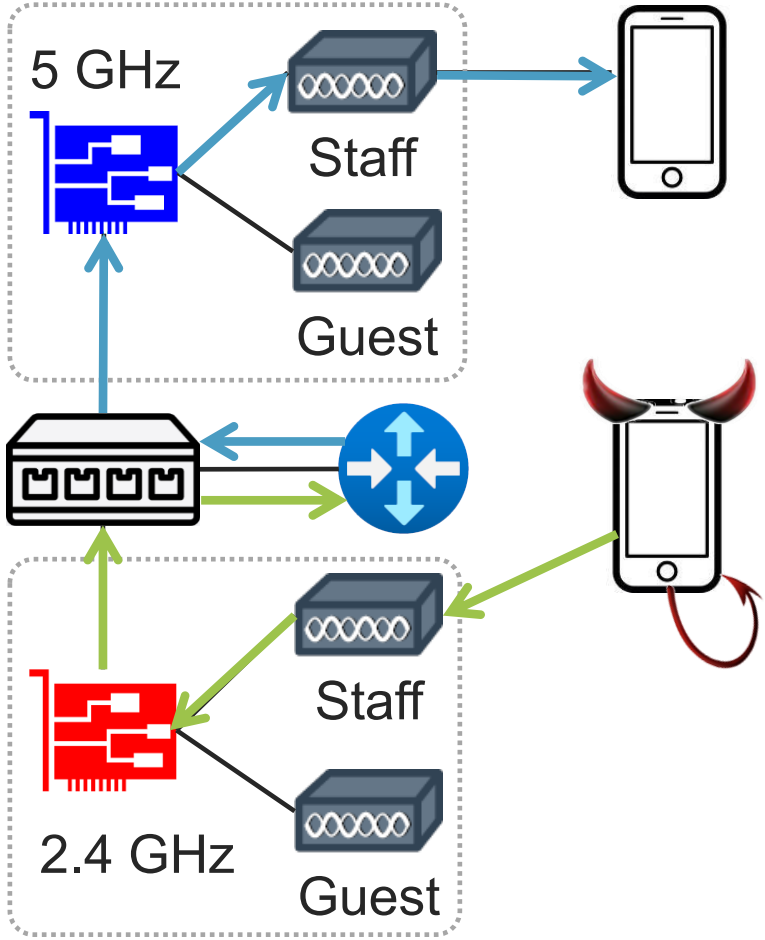
Send Ethernet frame to the **router**:

Ether(src =  , dst = ) / IP(...)

Allowed by NIC and switch. Router routes this at the IP layer and sends:

Ether(src =  , dst = ) / IP(...)

This bypasses client isolation!



Attack #3: Port stealing

Realization: Wi-Fi network internally uses Ethernet switches and tables…

Idea: can we **revive Ethernet port stealing** attacks in Wi-Fi networks?

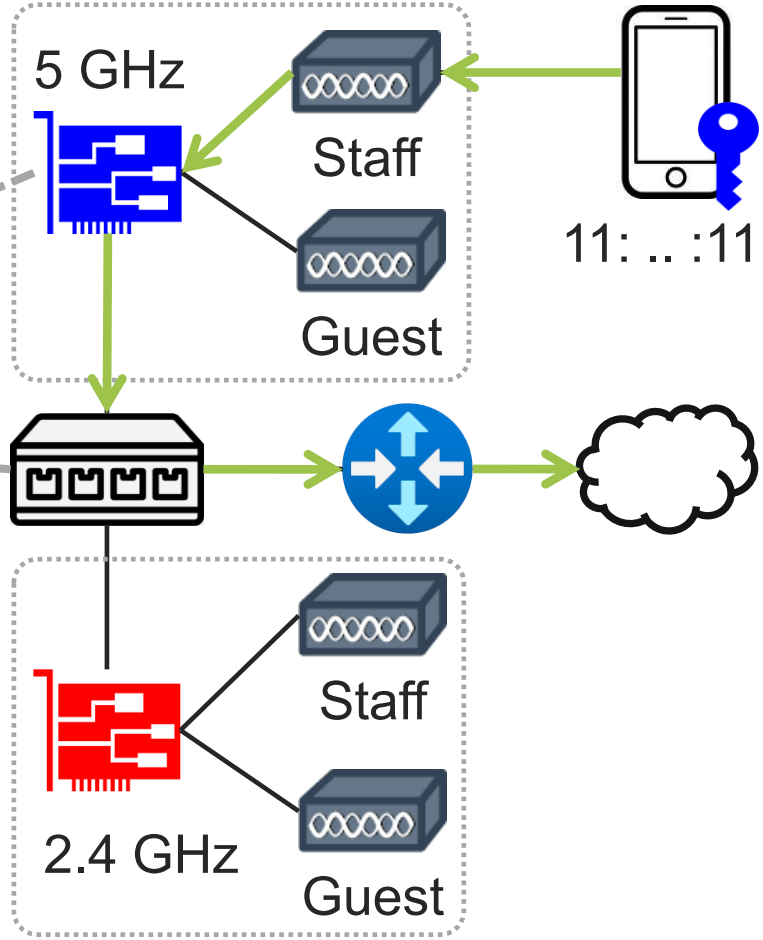
Background

Key table:

MAC address	key
11:11:11:11:11:11	

Switching Table:

MAC address	port
…	



Switching table automatically learns address/port mappings

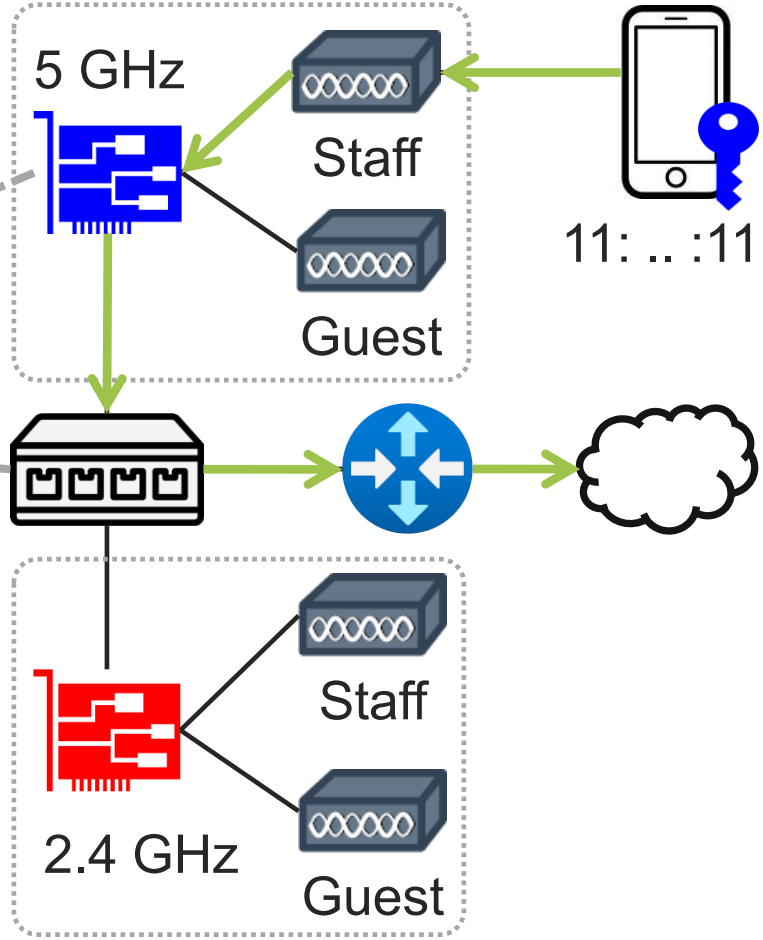
Background

Key table:

MAC address	key
11:11:11:11:11:11	

Switching Table:

MAC address	port
11:11:11:11:11:11	



Switching table automatically learns address/port mappings

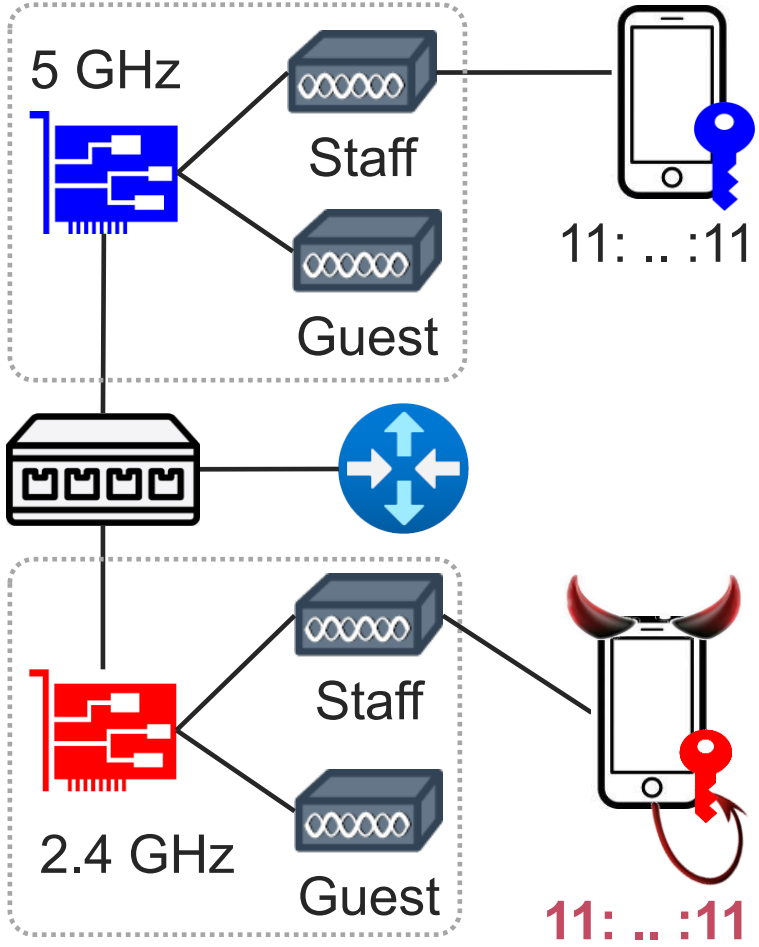
- Can we manipulate these tables?
- Yes, through spoofed addresses

Stealing downlink traffic

Adversary **spoofs victim's MAC**

- Must connect to different BSS to avoid kicking victim off network
- Even possible with WPA1/2/3

What happens to the switching and key tables?

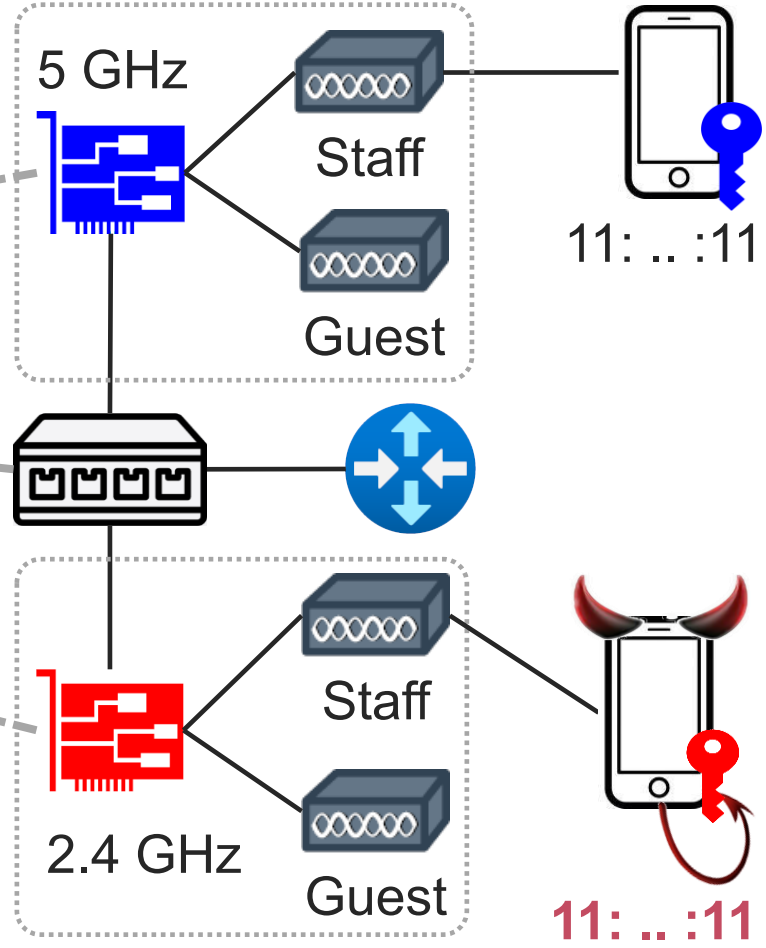


Stealing downlink traffic

MAC address	key
11:11:11:11:11:11	

MAC address	port
11:11:11:11:11:11	

MAC address	key
11:11:11:11:11:11	



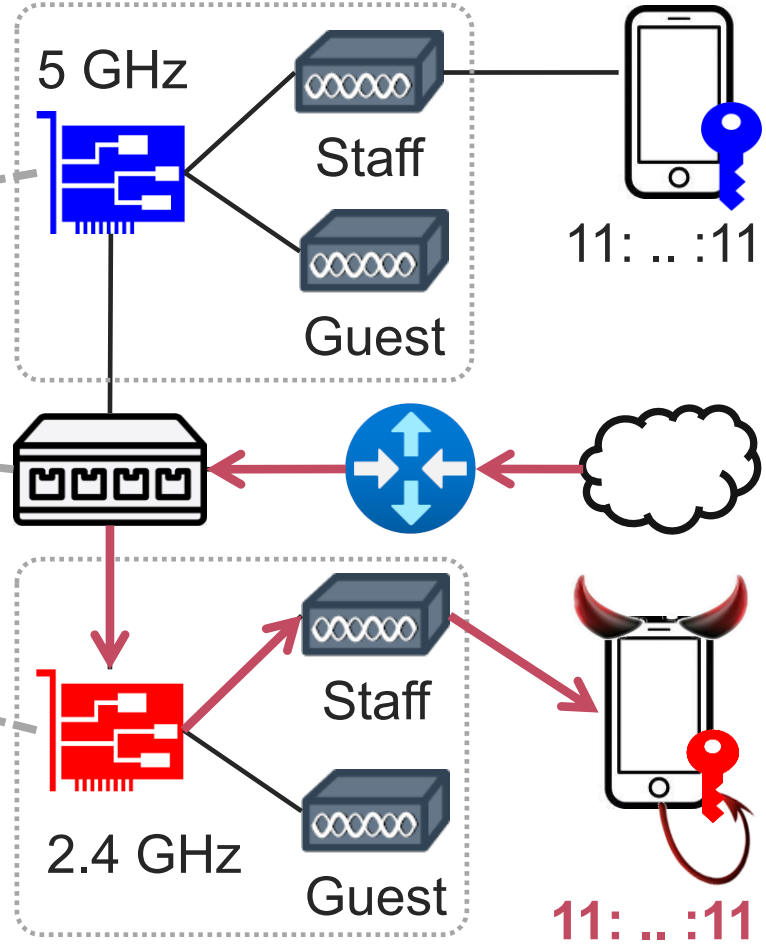
What now if there's an incoming packet for the victim (=downlink)?

Stealing downlink traffic

MAC address	key
11:11:11:11:11:11	

MAC address	port
11:11:11:11:11:11	

MAC address	key
11:11:11:11:11:11	



→ **Downlink traffic is redirected** to the adversary = isolation bypass!

Stealing uplink traffic?

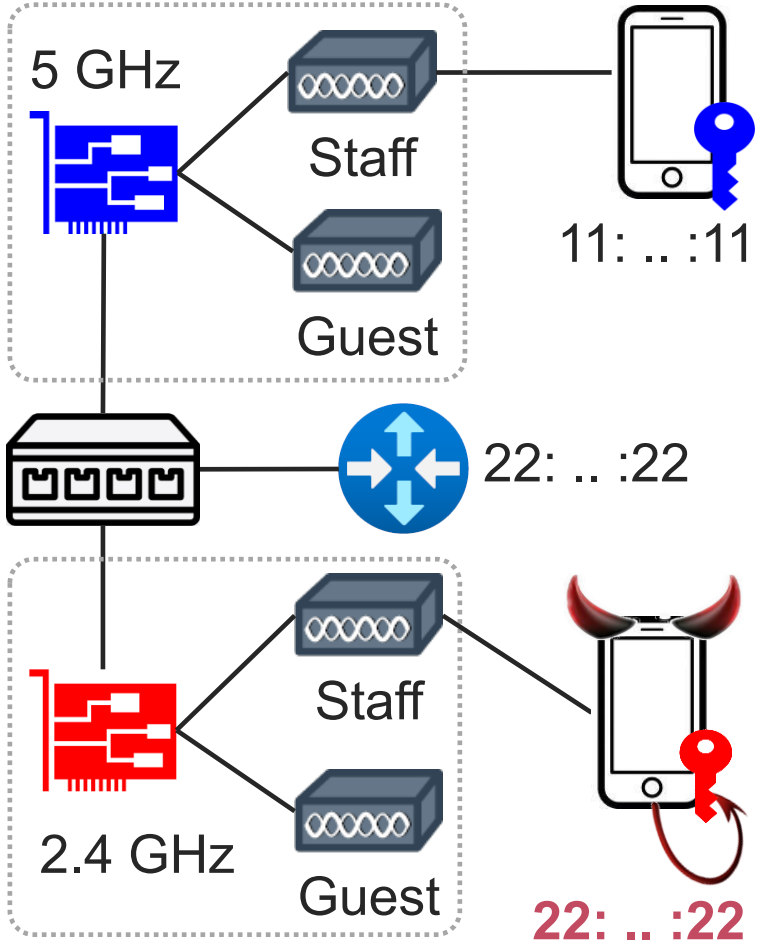
Can we also steal **uplink traffic**?
= Traffic from victim to Internet

Yes: we spoof the router's MAC ☺

Stealing uplink traffic

Adversary **spoofs router's MAC**

- Can connect to any BSS / AP
- Learn the router's MAC address by first connecting normally
- Most networks won't warn about duplicate MAC addresses…

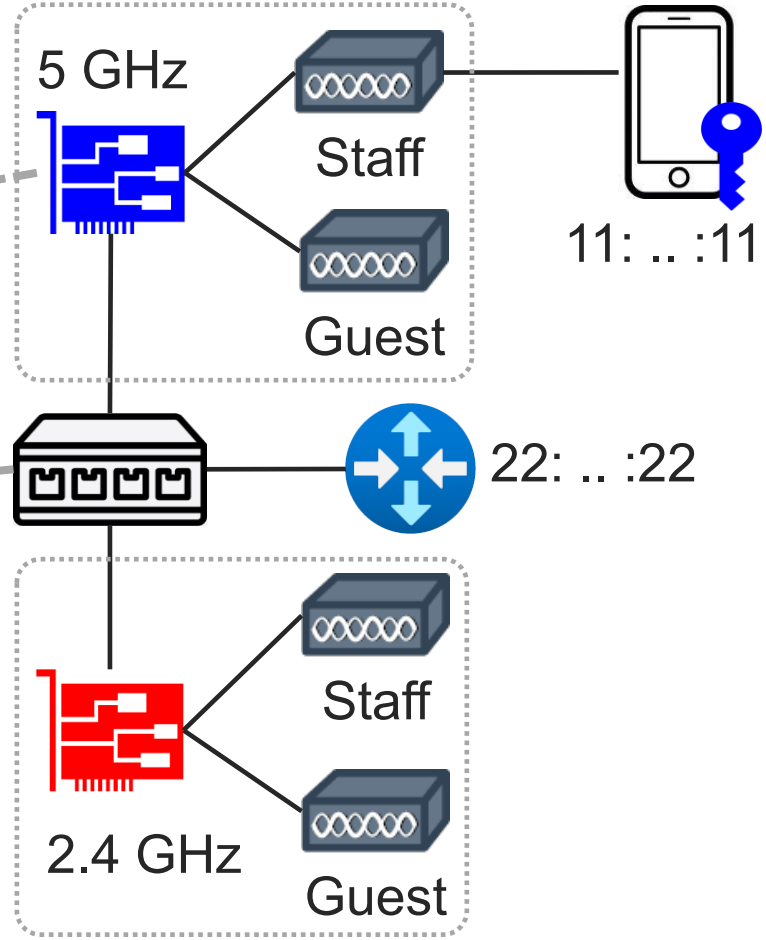


Stealing uplink traffic

MAC address	key
11:11:11:11:11:11	

MAC address	port
11:11:11:11:11:11	
22:22:22:22:22:22	

State before the attack

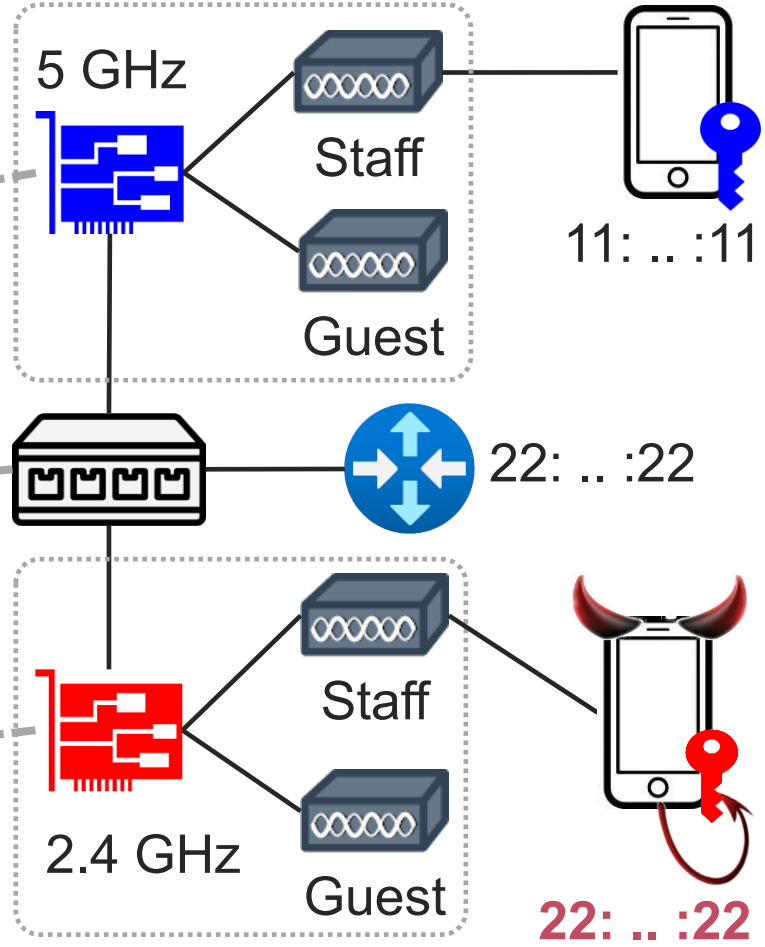


Stealing uplink traffic

MAC address	key
11:11:11:11:11:11	

MAC address	port
11:11:11:11:11:11	
22:22:22:22:22:22	

MAC address	key
22:22:22:22:22:22	



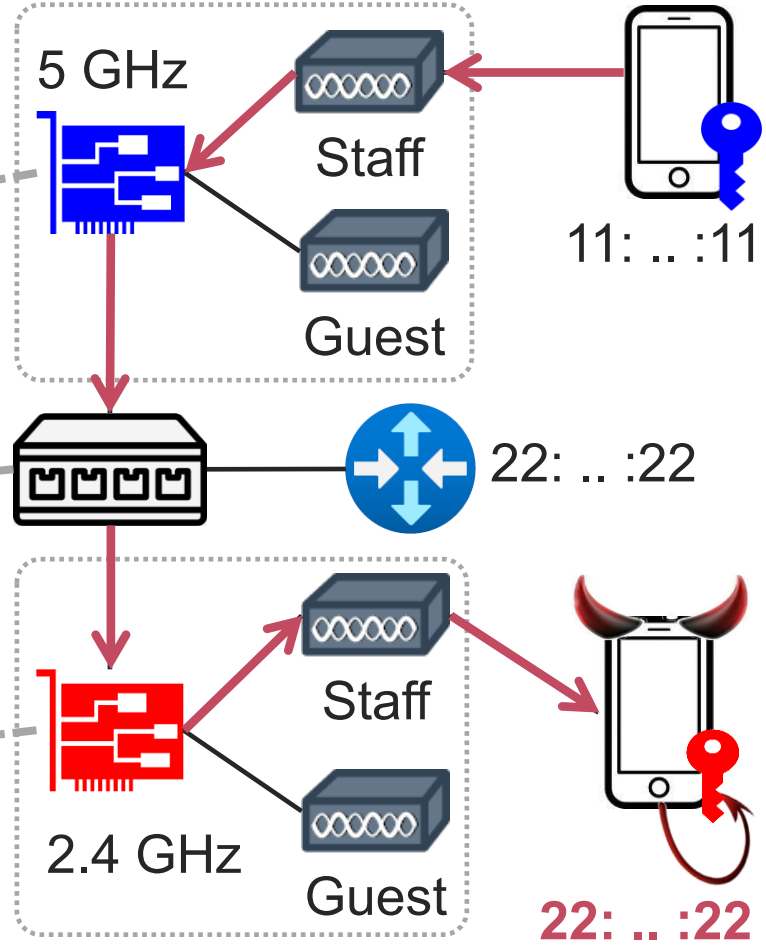
What happens on uplink traffic?

Stealing uplink traffic

MAC address	key
11:11:11:11:11:11	

MAC address	port
11:11:11:11:11:11	
22:22:22:22:22:22	

MAC address	key
22:22:22:22:22:22	



Uplink traffic stolen = **isolation bypass**!

Overview so far

1. Abuse group key → **inject** frames
2. Gateway bouncing → **inject** frames
3. Port stealing → **intercept** frames

Obtain full MitM despite isolation by combining techniques?

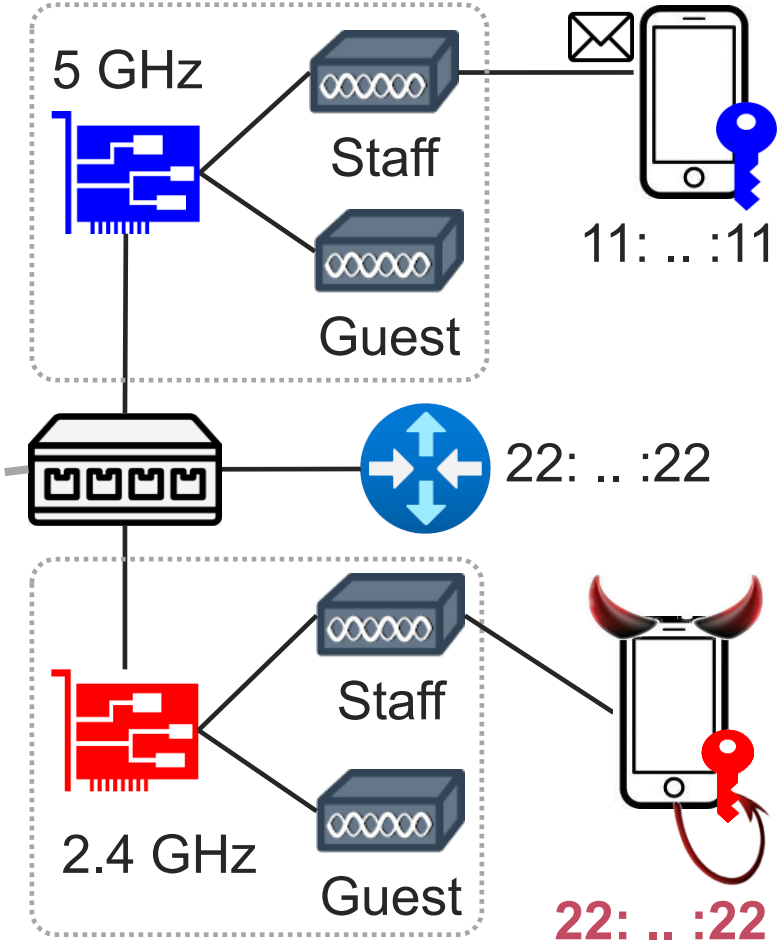
› Yes, but not trivially out-of-the-box…

› …need to handle impact of port stealing

Towards uplink MitM

Assume we used port stealing to intercept uplink packet ✉

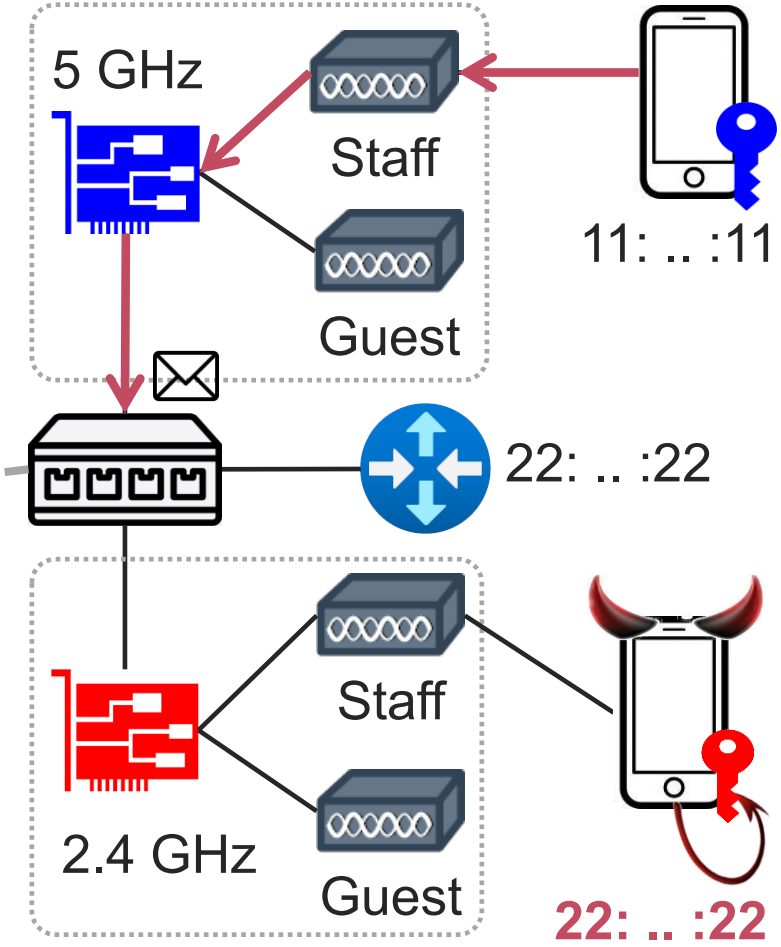
MAC address	port
11:11:11:11:11:11	
22:22:22:22:22:22	



Towards uplink MitM

Assume we used port stealing to intercept uplink packet ✉

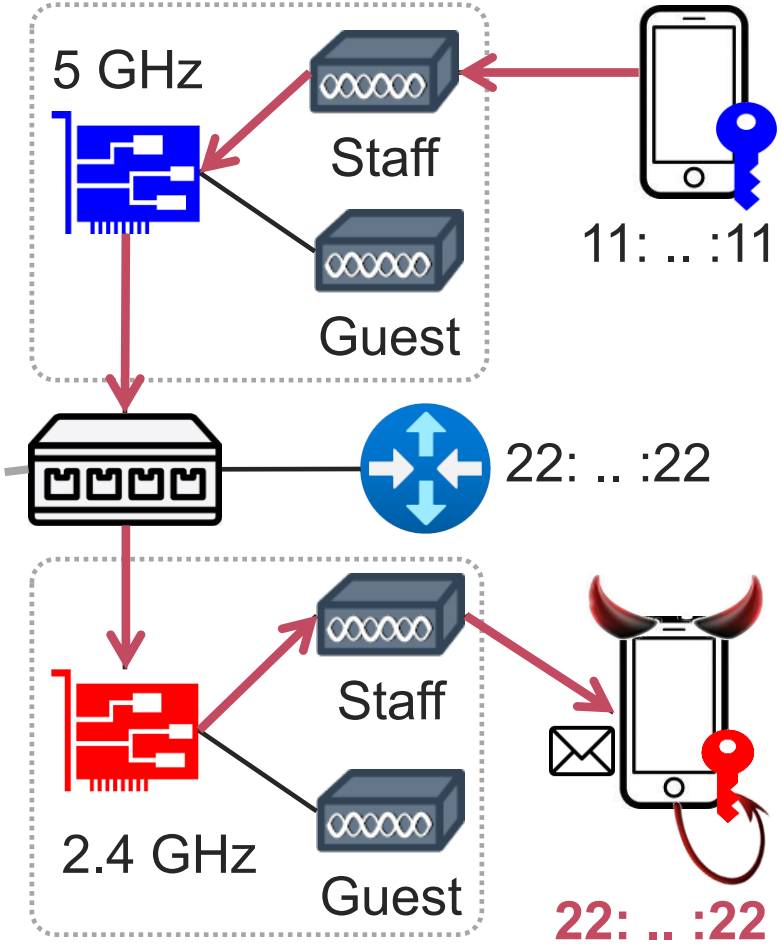
MAC address	port
11:11:11:11:11:11	
22:22:22:22:22:22	



Towards uplink MitM

Assume we used port stealing to intercept uplink packet ✉

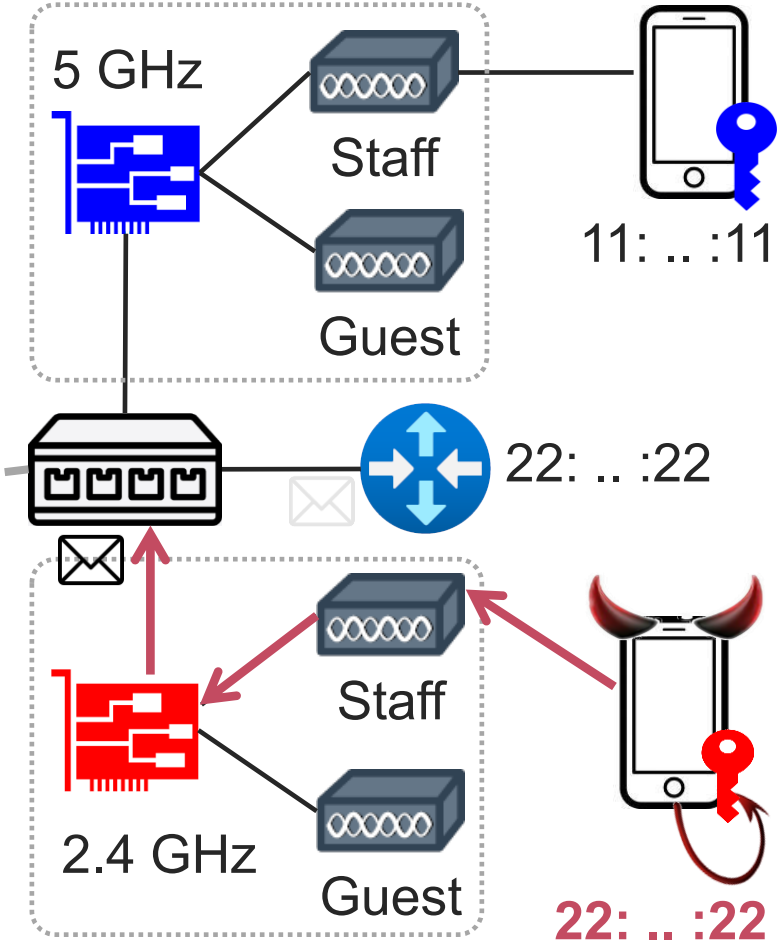
MAC address	port
11:11:11:11:11:11	
22:22:22:22:22:22	



Forwarding packet

Assume we used port stealing to intercept uplink packet ✉

MAC address	port
11:11:11:11:11:11	
22:22:22:22:22:22	

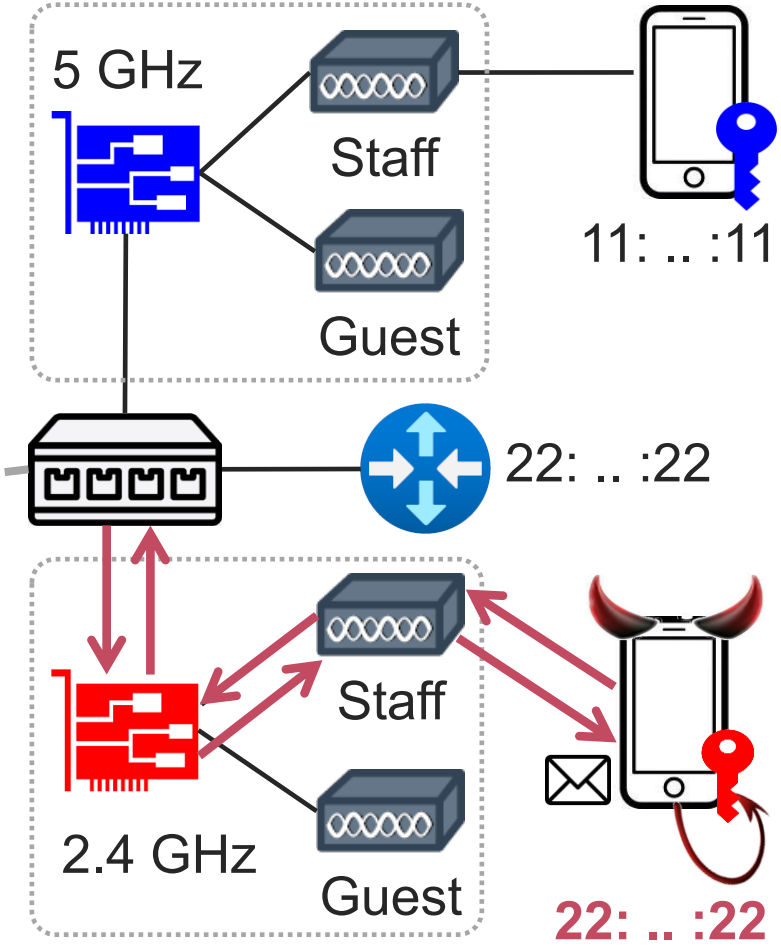


Issue 1: unable to forward packet to its real destination (e.g., the Internet)

Forwarding packet

Assume we used port stealing to intercept uplink packet ✉

MAC address	port
11:11:11:11:11:11	
22:22:22:22:22:22	



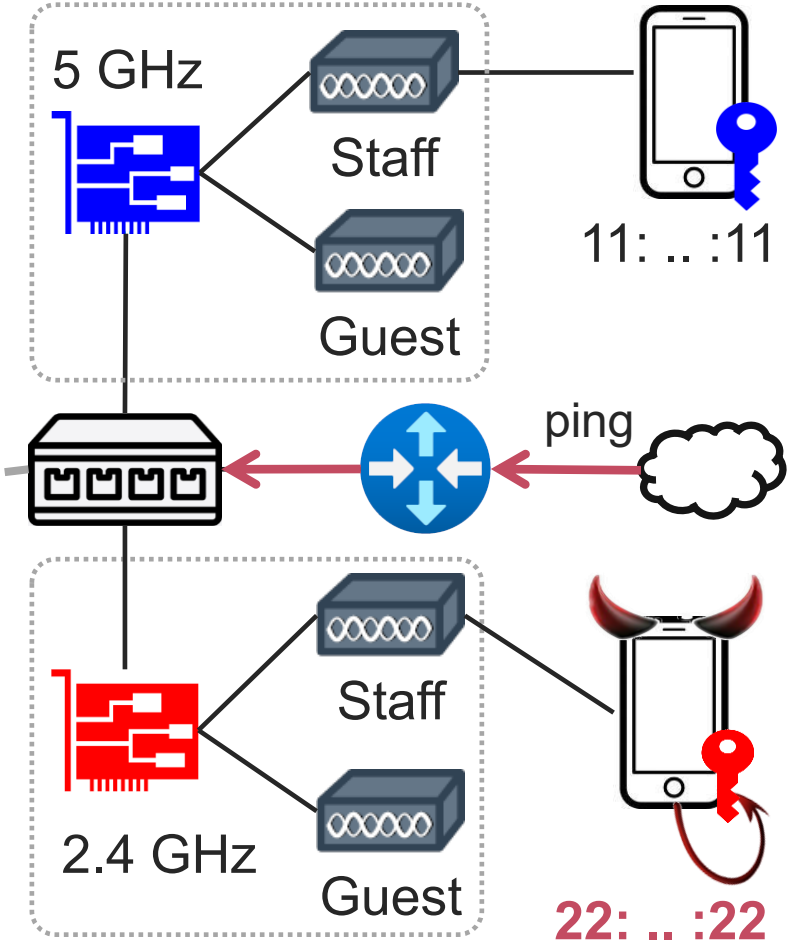
Issue 1: unable to forward packet to its real destination (e.g., the Internet)

› It's sent back to adversary!

Have uplink MitM

Server-triggered **port restoration** by sending ping replies to Wi-Fi network

MAC address	port
11:11:11:11:11:11	
22:22:22:22:22:22	



The attacker now *can* forward uplink traffic to its real destination

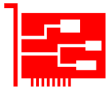
› Port restoration done continuously

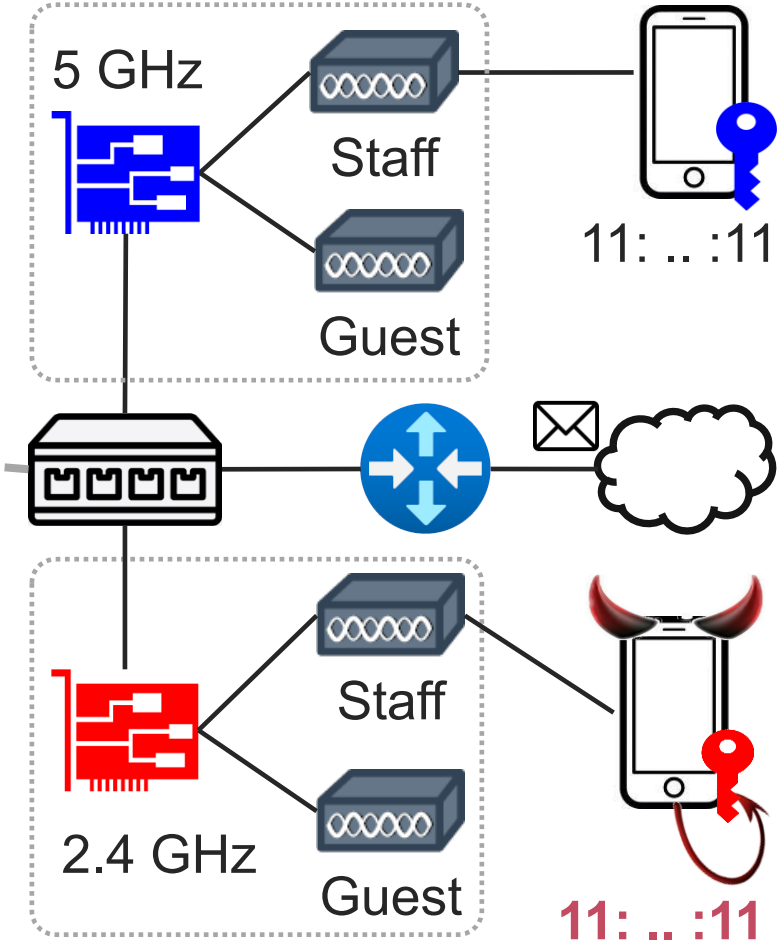
Got MitM in uplink path!

Now MitM for downlink path?

Towards downlink MitM

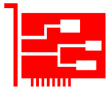
Next, assume we also used port stealing to intercept downlink traffic:

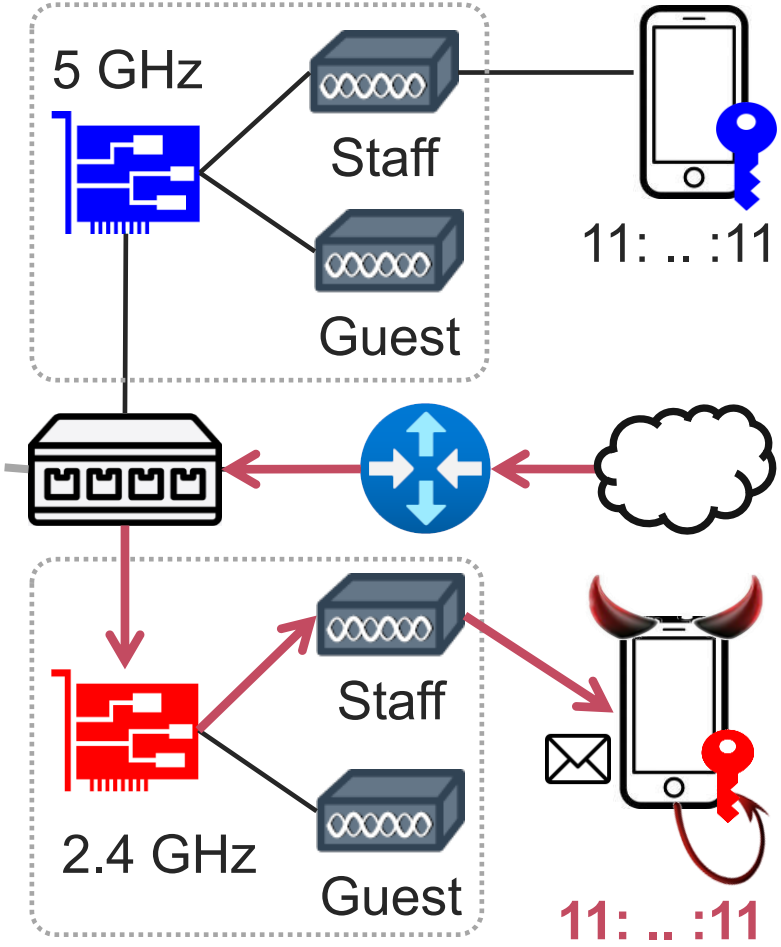
MAC address	port
11:11:11:11:11:11	



Towards downlink MitM

Next, assume we also used port stealing to intercept downlink traffic:

MAC address	port
11:11:11:11:11:11	

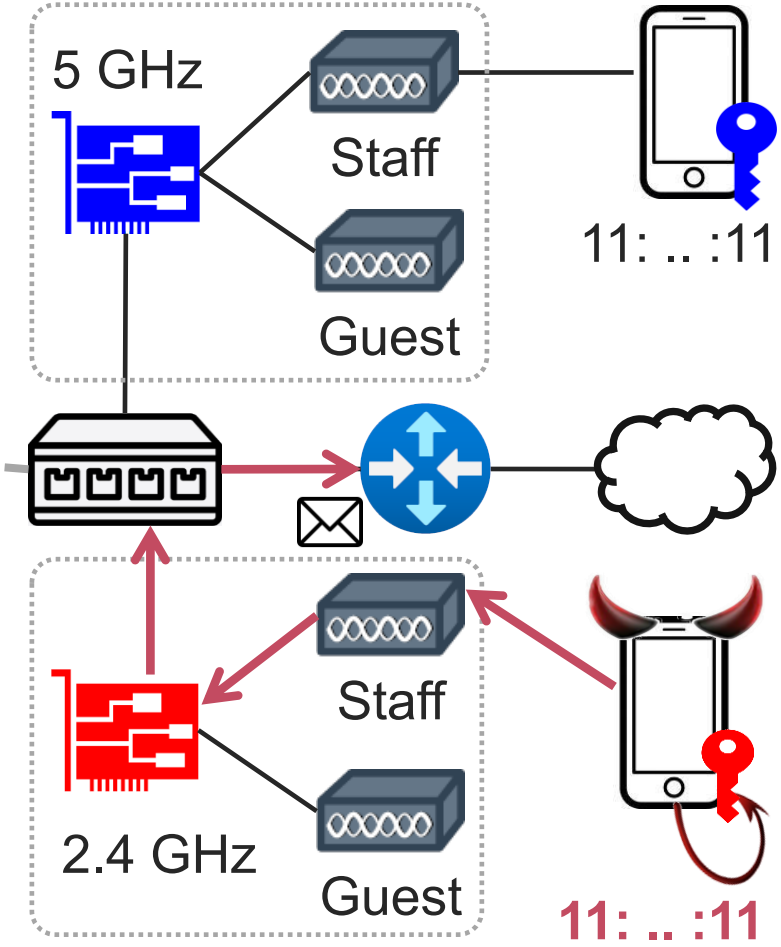


Forwarding packet

Next, assume we also used port stealing to intercept downlink traffic:

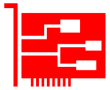
MAC address	port
11:11:11:11:11:11	

Issue 2: can't use gateway bouncing to forward frame to the victim



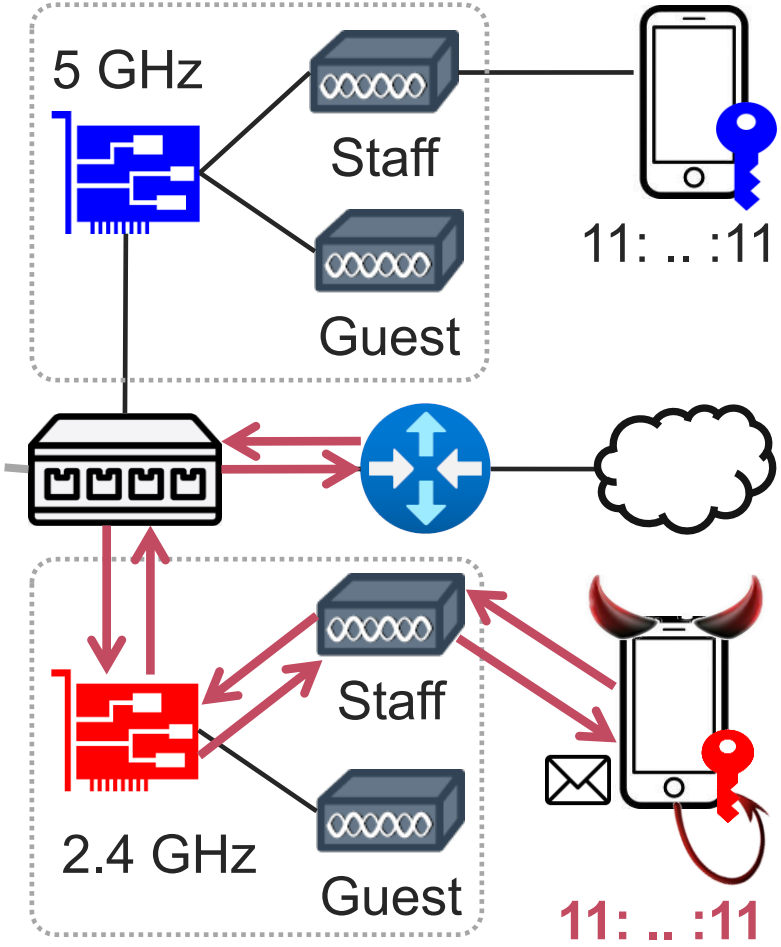
Forwarding packet

Next, assume we also used port stealing to intercept downlink traffic:

MAC address	port
11:11:11:11:11:11	

Issue 2: can't use gateway bouncing to forward frame to the victim

› It's also sent back to adversary…



Have downlink MitM

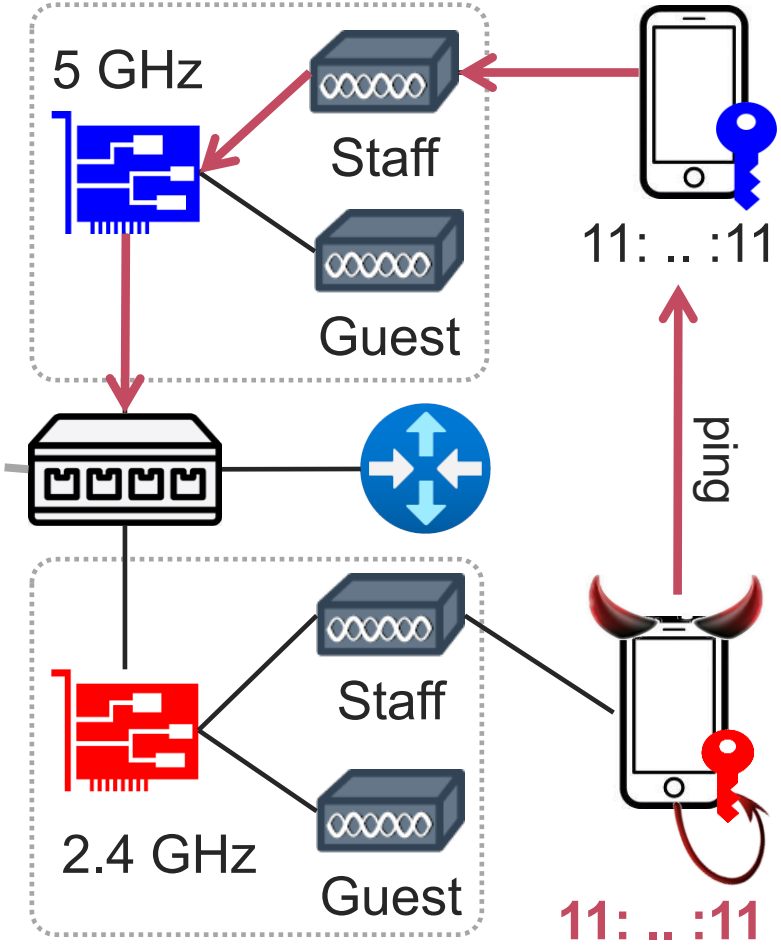
Client-triggered **port restoration** by making the victim send ping replies

MAC address	port
11:11:11:11:11:11	

How to trigger ping replies?

- Abuse shared group key to inject ping requests to the victim

Now have full down/uplink MitM!



Practical Impact: Measurements

Clients: abusing group key (unicast in b-cast Wi-Fi)

OS	group-ping	group-ping6	group-arp-unicast
Windows 11			
Firewall On	×	✓	✓
Firewall Off	✓	✓	✓
Others			
macOS 15.4	✓	✓	✓
iOS 18.3.2	✓	✓	✓
Android 14	✓	✓	✓
Ubuntu 22.04	×*	✓	✓

✓: Test case passed (OS replied to the probe).
×: Test case failed (OS did not reply).

Tested devices

Created AirSnitch, an extension of MacStealer

- Tested 7 home routers and 4 professional APs
- Enabled client isolation in their settings
- Created a guest and main network (sometimes manually)

→ All were vulnerable to at least one attack!

APs: randomized group keys & gateway bouncing

Device Model	Abusing GTK		Gateway Bouncing			
	M→M	G→G	G→M	M→M	G→G	M→G
Netgear Nighthawk X6 R8000	✓	✓				
Tenda RX2 Pro	✓	✓				
D-Link DIR-3040	✓	✓				
TP-Link Archer AXE75	✓	✓				
ASUS RT-AX57	✓	✓				
DD-WRT v3.0-r44715	✓	✓	×	✓	×	×
OpenWrt 24.10	✓	✓	×	✓	×	×
Ubiquiti AmpliFi Alien Router	✓	✓	×	✓	✓	✓
Ubiquiti AmpliFi Router HD	✓	✓	×	✓	✓	✓
Cisco Catalyst 9130	✓	✓	✓	✓	✓	✓
LANCOM LX-6500	✓	✓	✓	✓	✓	✓

No AP randomized the group key

APs: randomized group keys & gateway bouncing

Device Model	Abusing GTK		Gateway Bouncing			
	M→M	G→G	G→M	M→M	G→G	M→G
Netgear Nighthawk X6 R8000	✓	✓	✓	✓	✓	✓
D-Link DIR-3040	✓	✓	×	✓	✓	×
ASUS RT-AX57	✓	✓	✓	✓	✓	✓
OpenWrt 24.10	✓	✓	✓	✓	✓	✓
	✓	✓	×	✓	×	✓
	✓	✓	×	✓	×	×
	✓	✓	×	✓	×	×
	✓	✓	×	✓	✓	✓
Ubiquiti AmpliFi Router HD	✓	✓	×	✓	✓	✓
Cisco Catalyst 9130	✓	✓	✓	✓	✓	✓
LANCOM LX-6500	✓	✓	✓	✓	✓	✓

Inconsistent if gateway bouncing works between main & guest network

→ Need standardization

APs: port stealing attacks

In ~half of cases, guest can intercept traffic of main user

Device Model	Downlink Port Stealing				Uplink Port Stealing			
	G←M	M←M	G←G	M←G	G←M	M←M	G←G	M←G
Netgear Nighthawk X6 R8000	✓	✓	✓	✓	✓	✓	×	×
Tenda RX2 Pro	×	×	×	×	N/A★	✓	N/A★	N/A★
D-Link DIR-3040	✓	✓	✓	✓	✓	✓	✓	✓
TP-Link Archer AXE75	✓	×	✓	✓	✓	✓	×	✓
ASUS RT-AX57	×	✓	×	×	✓	✓	✓	✓
DD-WRT v3.0-r44715	×	×	×	×	×	✓	×	×
OpenWrt 24.10	×	×	×	×	×	×	×	×
Ubiquiti AmpliFi Alien Router	×	✓	✓	×	×	✓	×	×
Ubiquiti AmpliFi Router HD	×	✓	✓	×	×	✓	×	×
Cisco Catalyst 9130	✓	✓	✓	✓	×	×	×	×
LANCOM LX-6500	✓	✓	✓	✓	✓	✓	✓	✓

Enterprise network tests

University #1

- Could inject packets from ‘guest’ to ‘staff’ network

University #2

- Could perform port stealing to intercept packets from clients in the ‘staff’ network
- They were **leaked to the open guest network!**

Defenses

- **Randomizing group keys**
- Drop unicast IP packets inside broadcast Wi-Fi frames

Proper traffic isolation

- Use **VLANs**. If possible, unique VLAN per user.
- Ensure proper firewall rules at Ethernet and IP layer!
- Enable IP spoofing prevention
- See our paper for more…

The Disclosure

“we only cover issues in layer 1 & 2”

“it’s a protocol issue, need input from the Wi-Fi Alliance”

“this is a configuration issue […so the user’s fault]”

The good news

- Apple: patched unicast IP in group Wi-Fi (CVE-2026-20671)
- LANCOM: Adding a setting to randomize group keys.
- Ubiquiti updated documentation: “Guest network users are not isolated from one another”.
- Wi-Fi Alliance is working on WPA3 Client Isolation (??)

Conclusion

AirSnitch GitHub repo*

- Wi-Fi Client isolation was not standardized
- Cross-layer attacks can bypass isolation

- Not always a software flaw, often also a config flaw
- Use AirSnitch to test your network!
- Read our NDSS'26 paper for more details ☺

** Repository is also mirrored by Xin'an and seclab-ucr*

Questions?

AirSnitch GitHub repo*

- Wi-Fi Client isolation was not standardized
- Cross-layer attacks can bypass isolation

- Not always a software flaw, often also a config flaw
- Use AirSnitch to test your network!
- Read our NDSS'26 paper for more details ☺

** Repository is also mirrored by Xin'an and seclab-ucr*