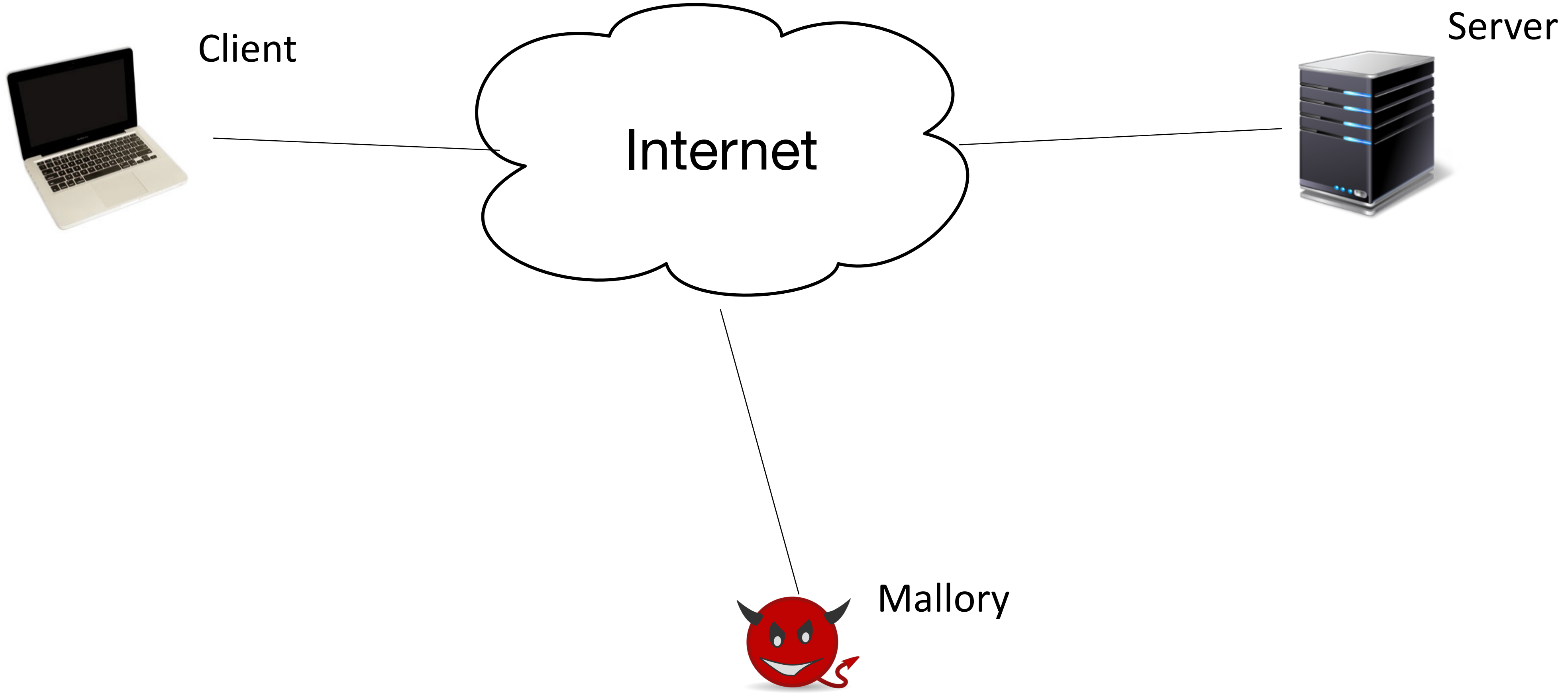
# Off-Path TCP Exploit: How Wireless Routers Can Jeopardize Your Secrets

Weiteng Chen, Zhiyun Qian

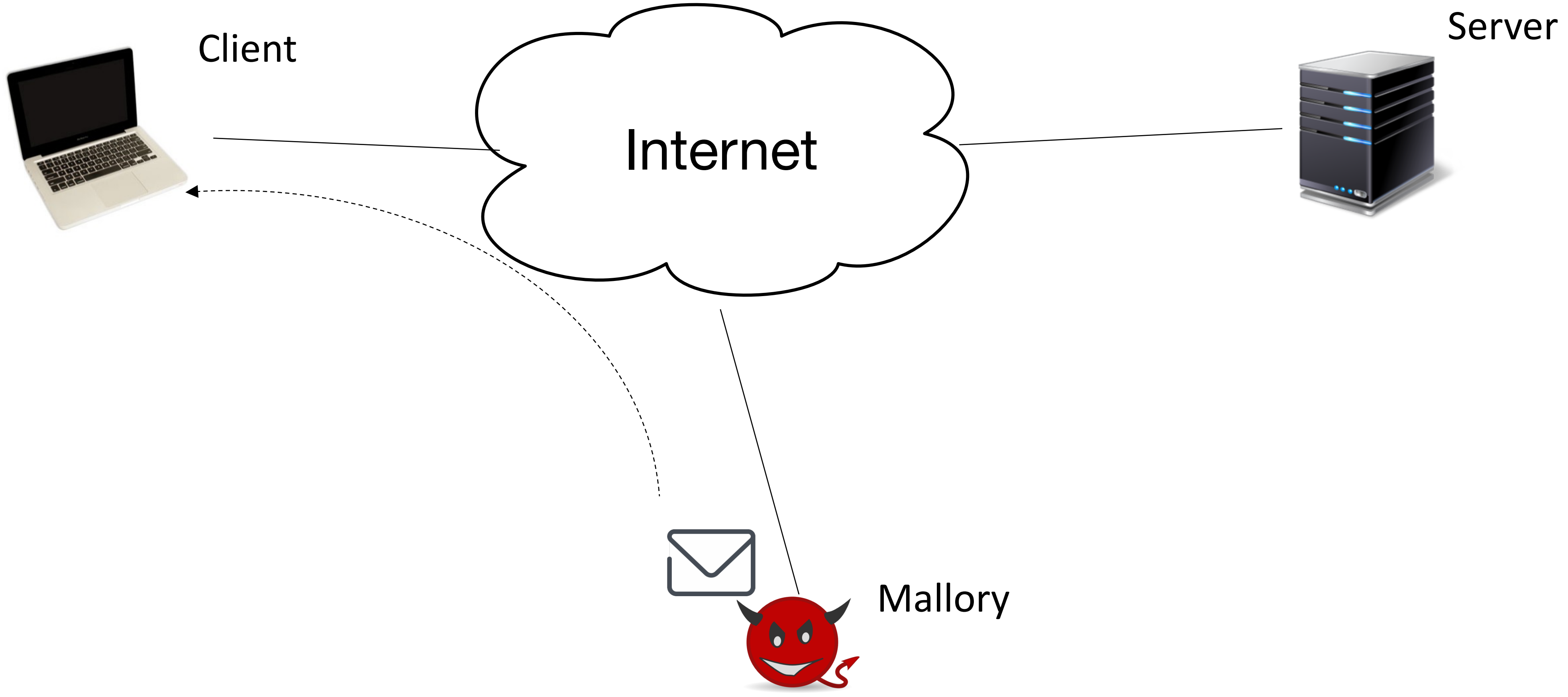
University of California, Riverside

UNIVERSITY OF CALIFORNIA
UCRIVERSIDE

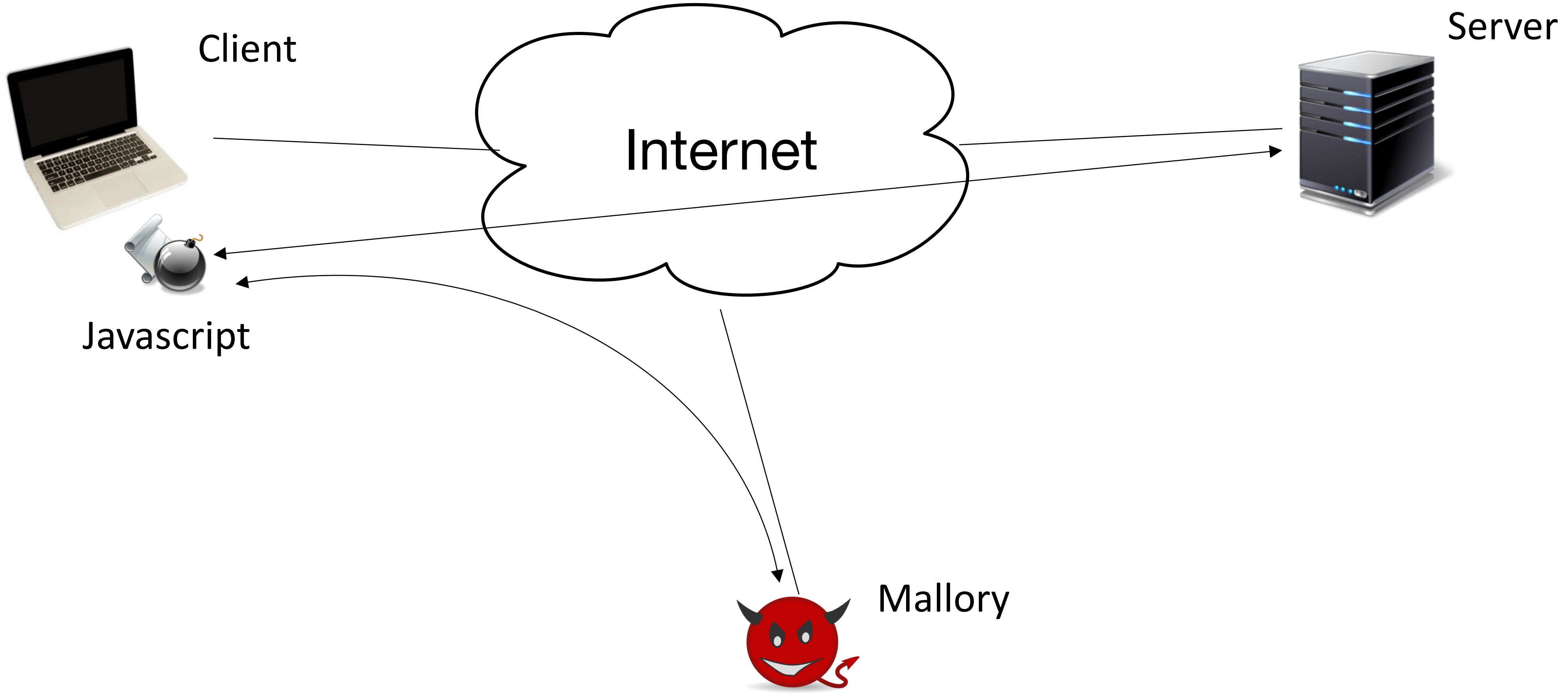
# Threat Model



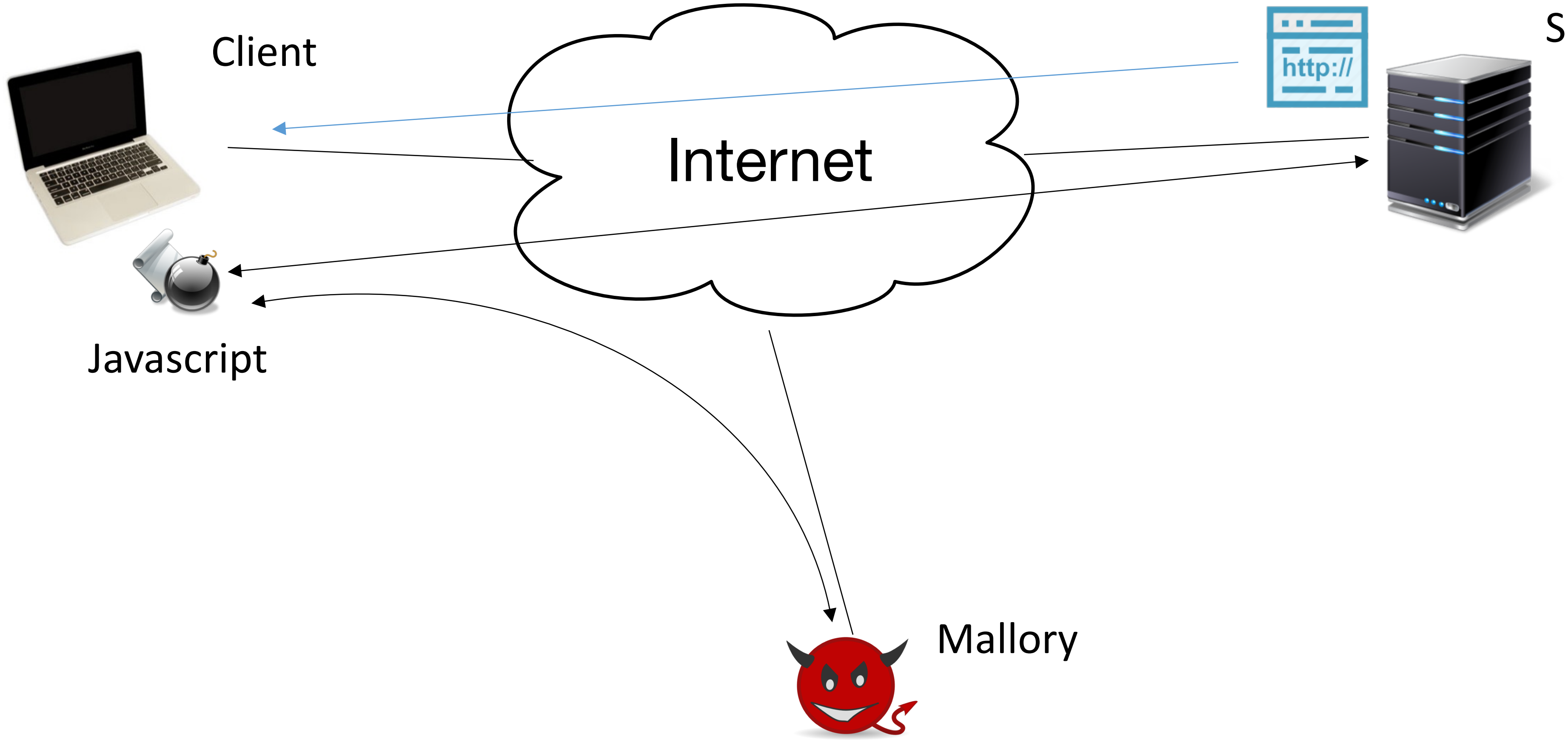
# Threat Model



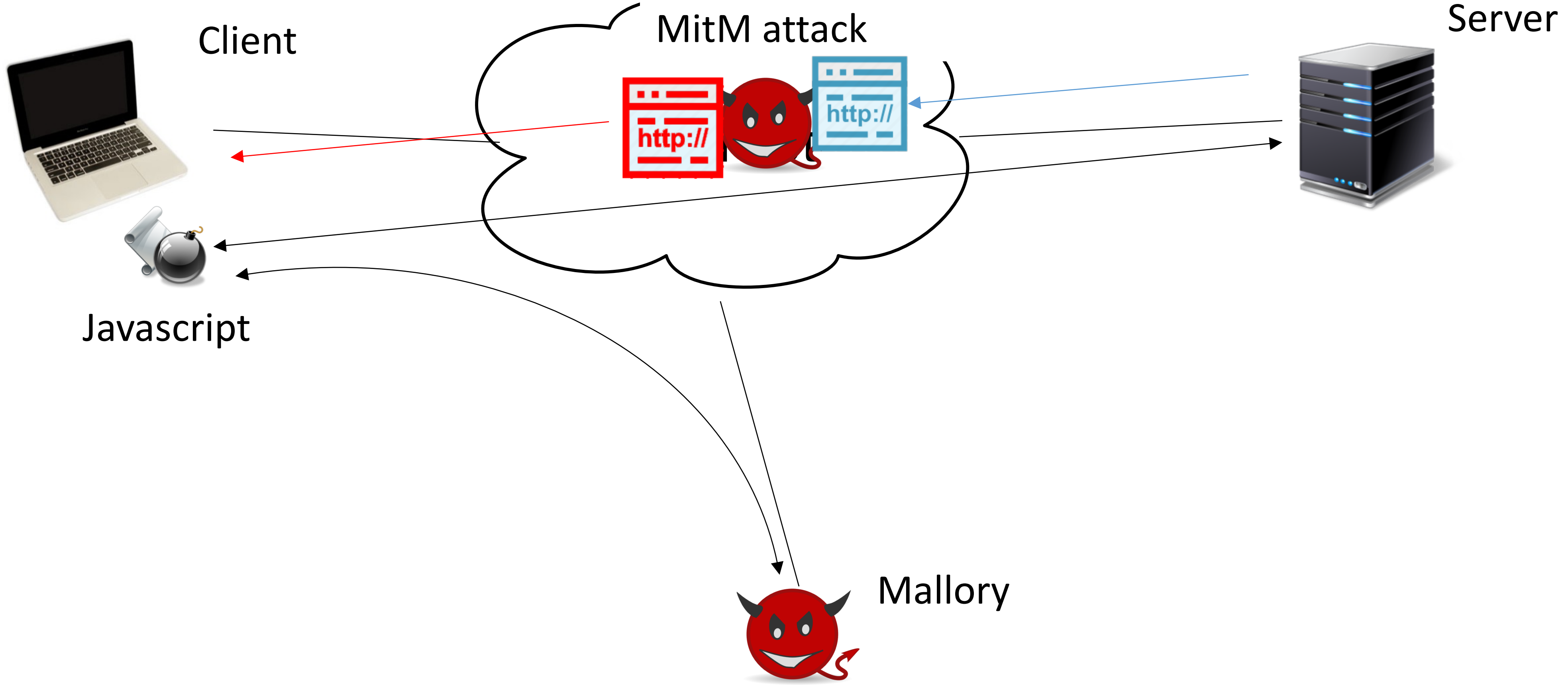
# Threat Model



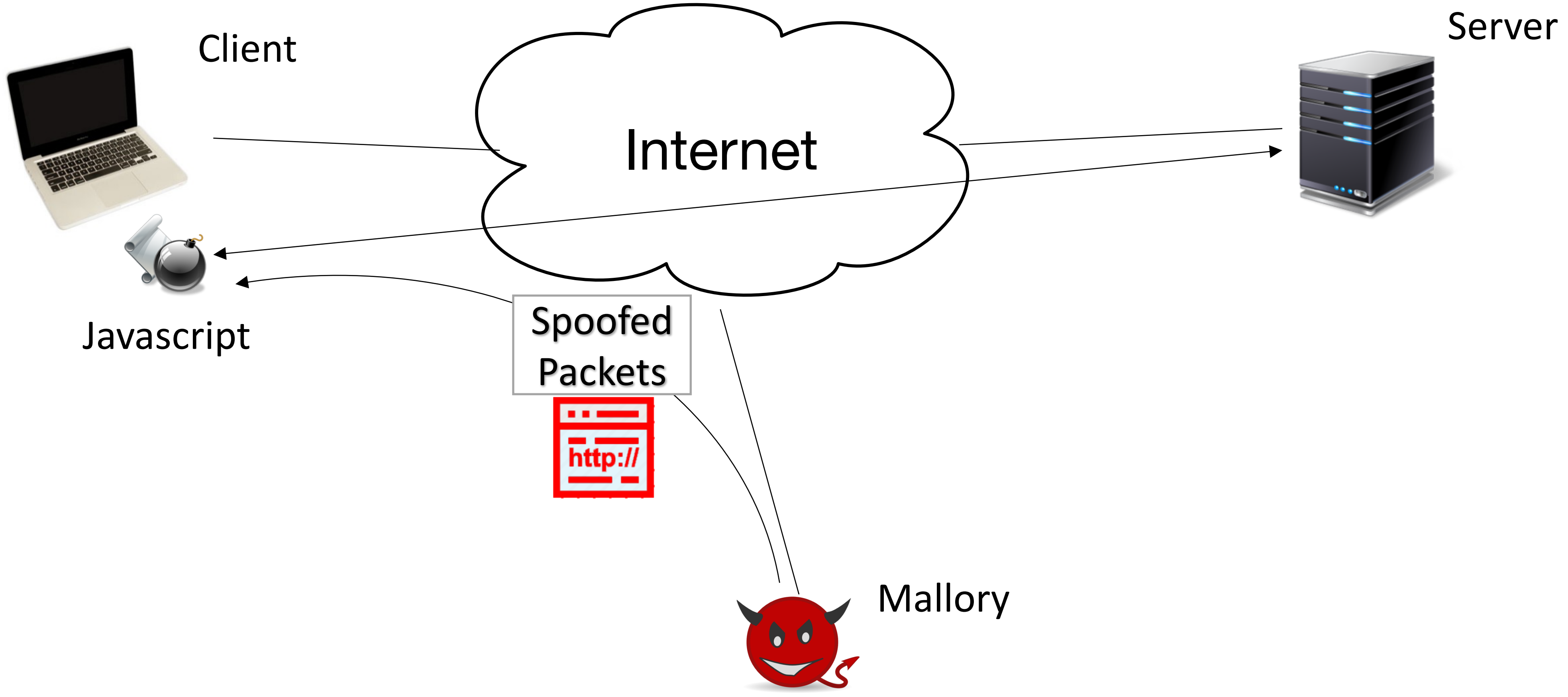
# Threat Model



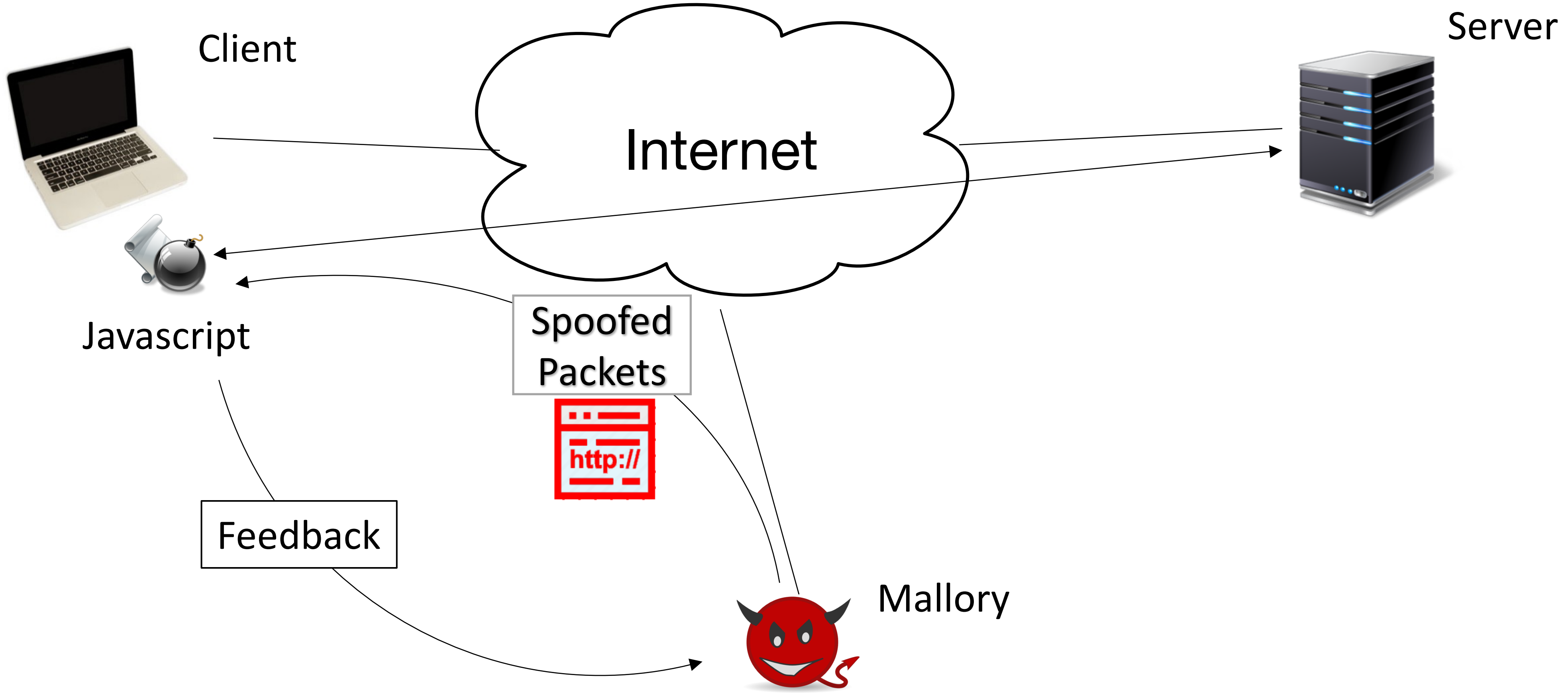
# Threat Model



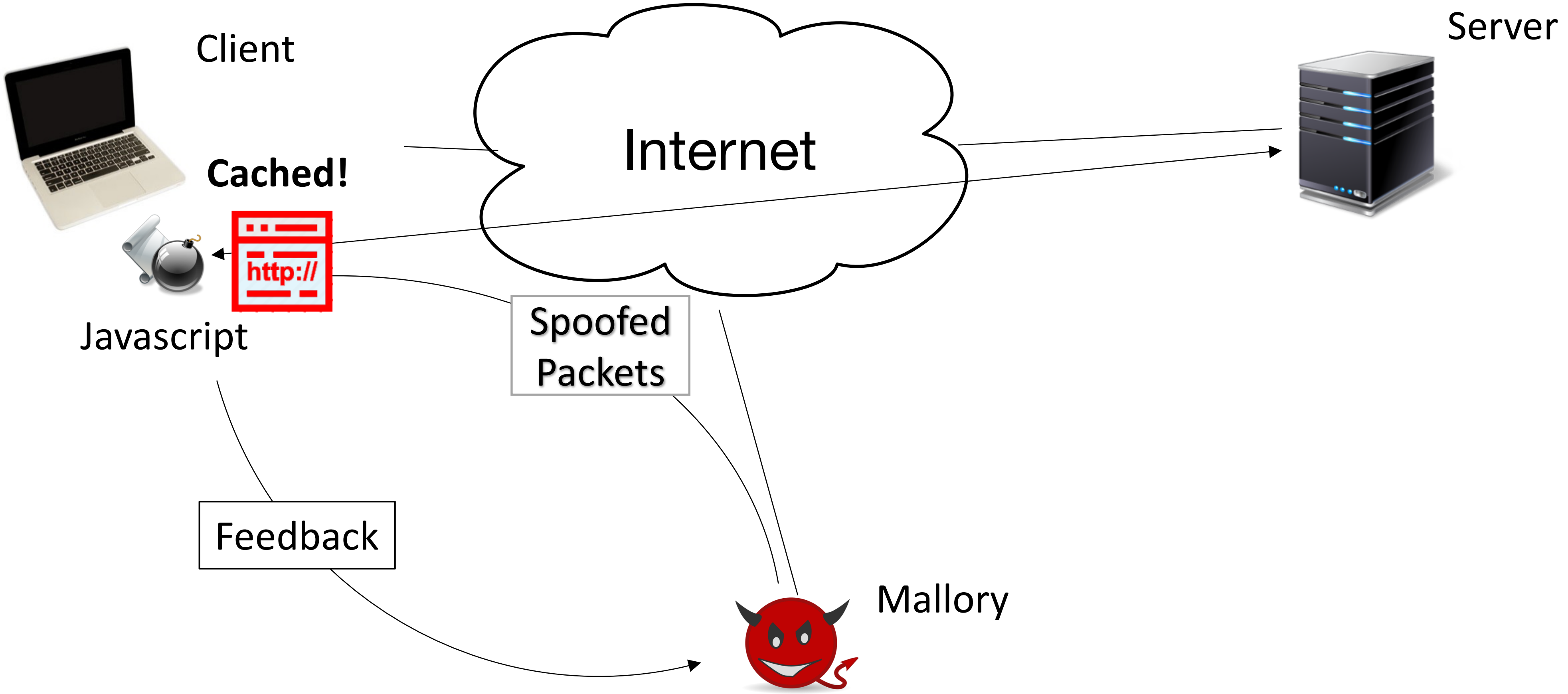
# Threat Model



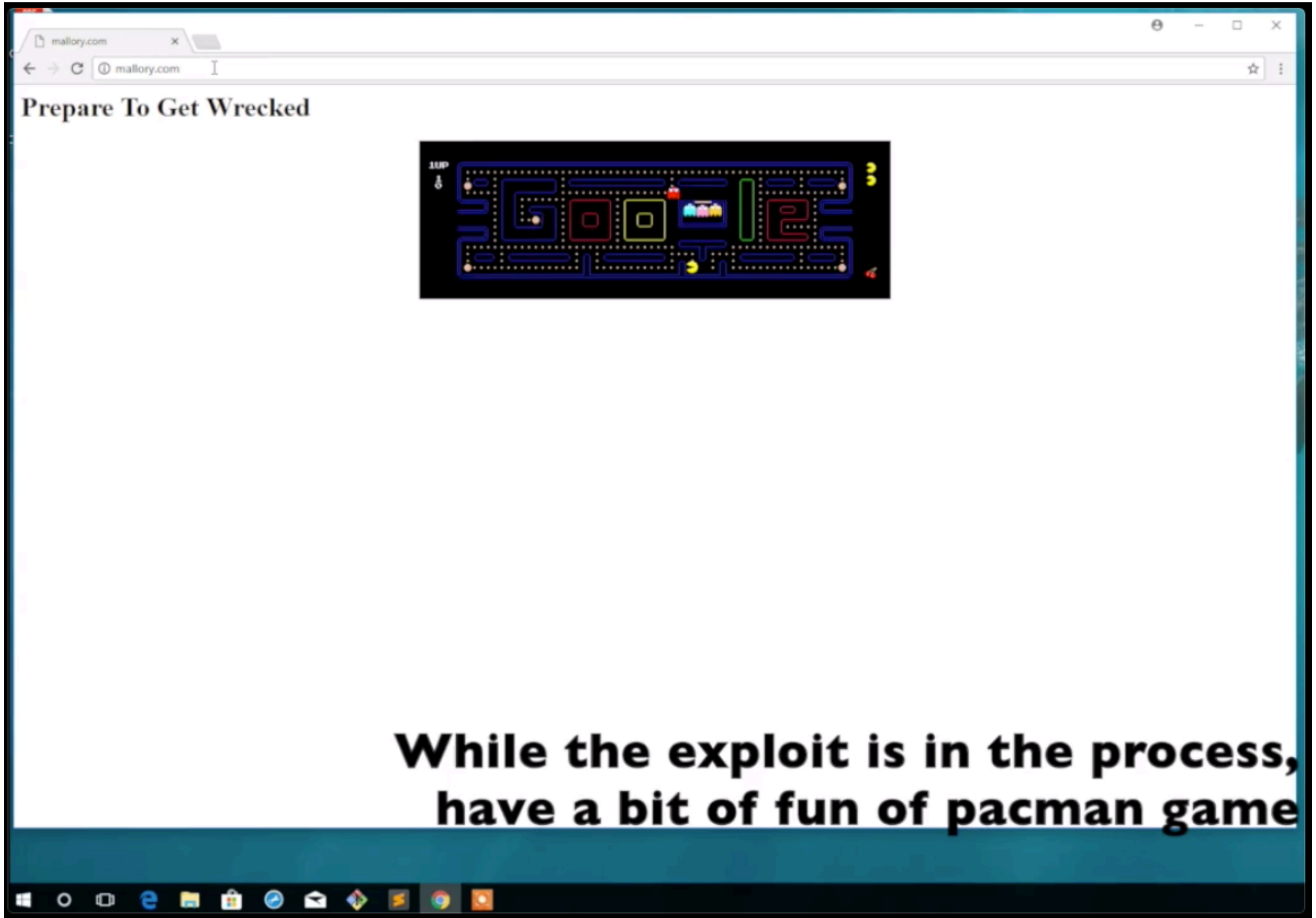
# Threat Model



# Threat Model



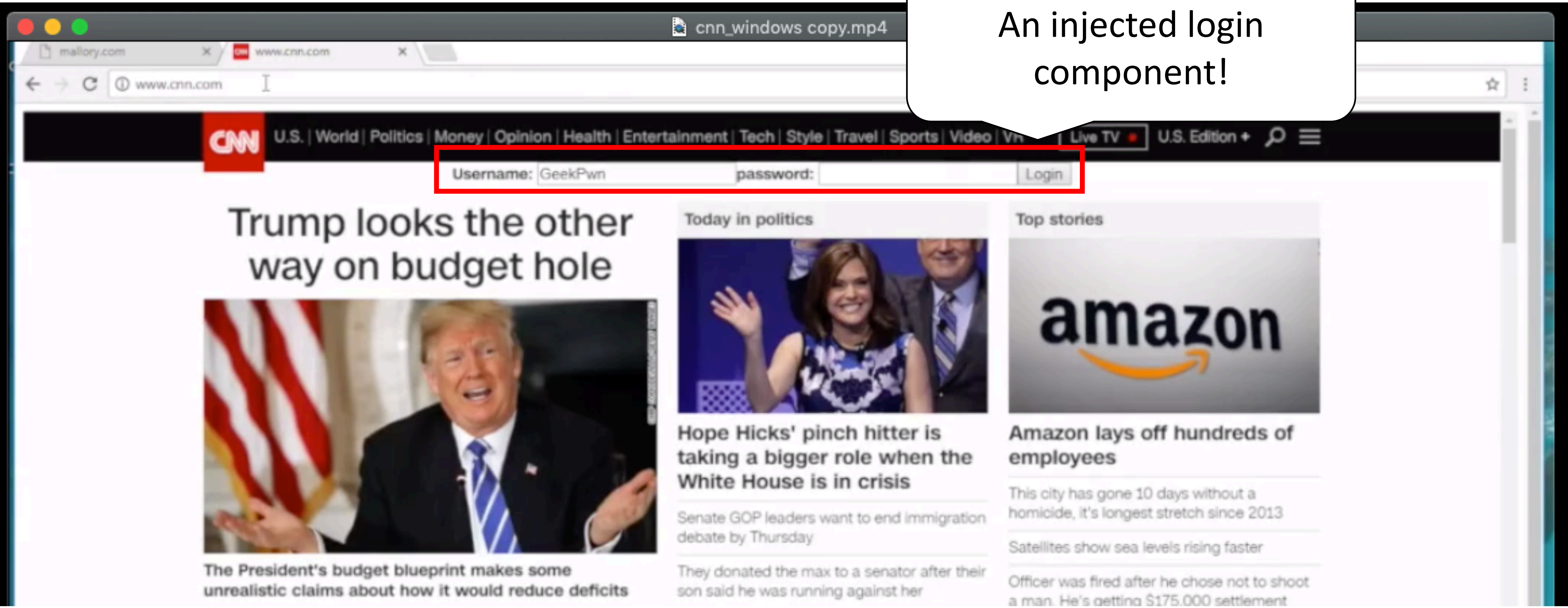
# Demo: Web Cache Poisoning



Prepare To Get Wrecked

While the exploit is in the process,
have a bit of fun of pacman game

# Demo: Web Cache Poisoning



cnn_windows copy.mp4
mallory.com
www.cnn.com
www.cnn.com
CNN
U.S. | World | Politics | Money | Opinion | Health | Entertainment | Tech | Style | Travel | Sports | Video | VR
Live TV
U.S. Edition +

Username: GeekPwn password: Login

An injected login component!

Trump looks the other way on budget hole

The President's budget blueprint makes some unrealistic claims about how it would reduce deficits

Today in politics

Hope Hicks' pinch hitter is taking a bigger role when the White House is in crisis

Senate GOP leaders want to end immigration debate by Thursday

They donated the max to a senator after their son said he was running against her

Top stories

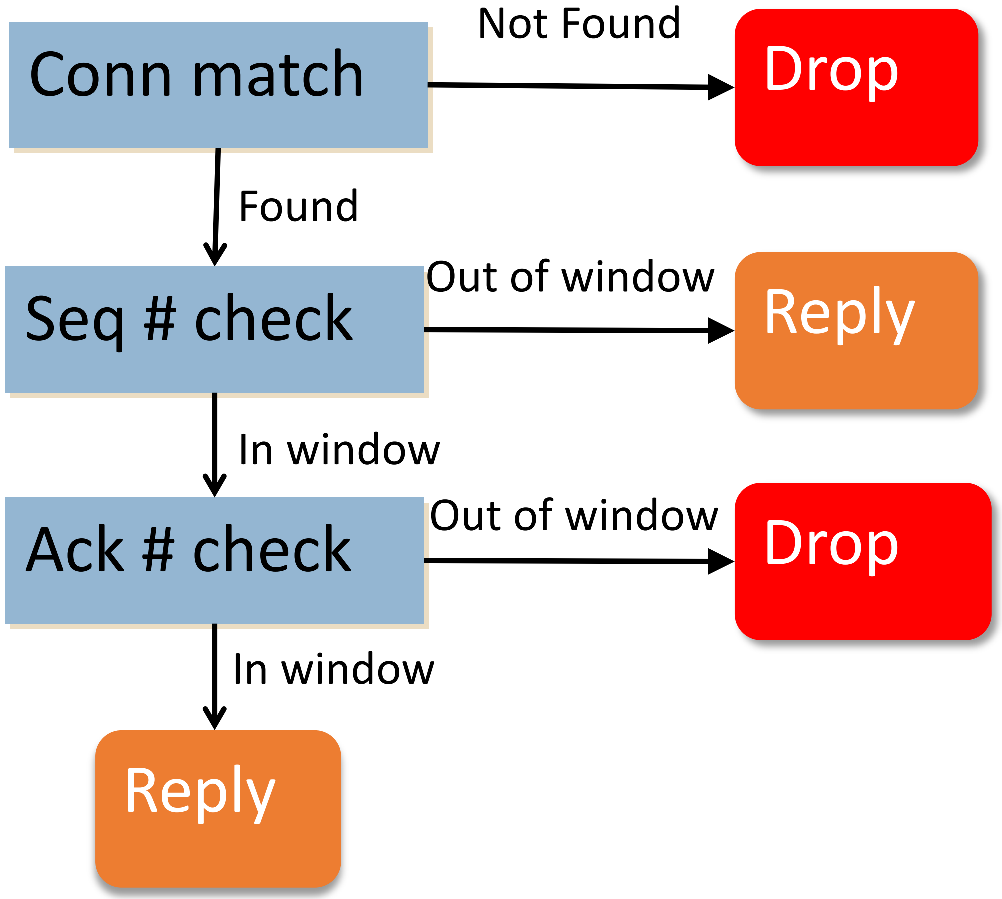
Amazon lays off hundreds of employees

This city has gone 10 days without a homicide, it's longest stretch since 2013

Satellites show sea levels rising faster

Officer was fired after he chose not to shoot a man. He's getting $175,000 settlement

# RFC 793: TCP Packet Receiving Basics



Conn match

Not Found

Drop

Found

Seq # check

Out of window

Reply

In window

Ack # check

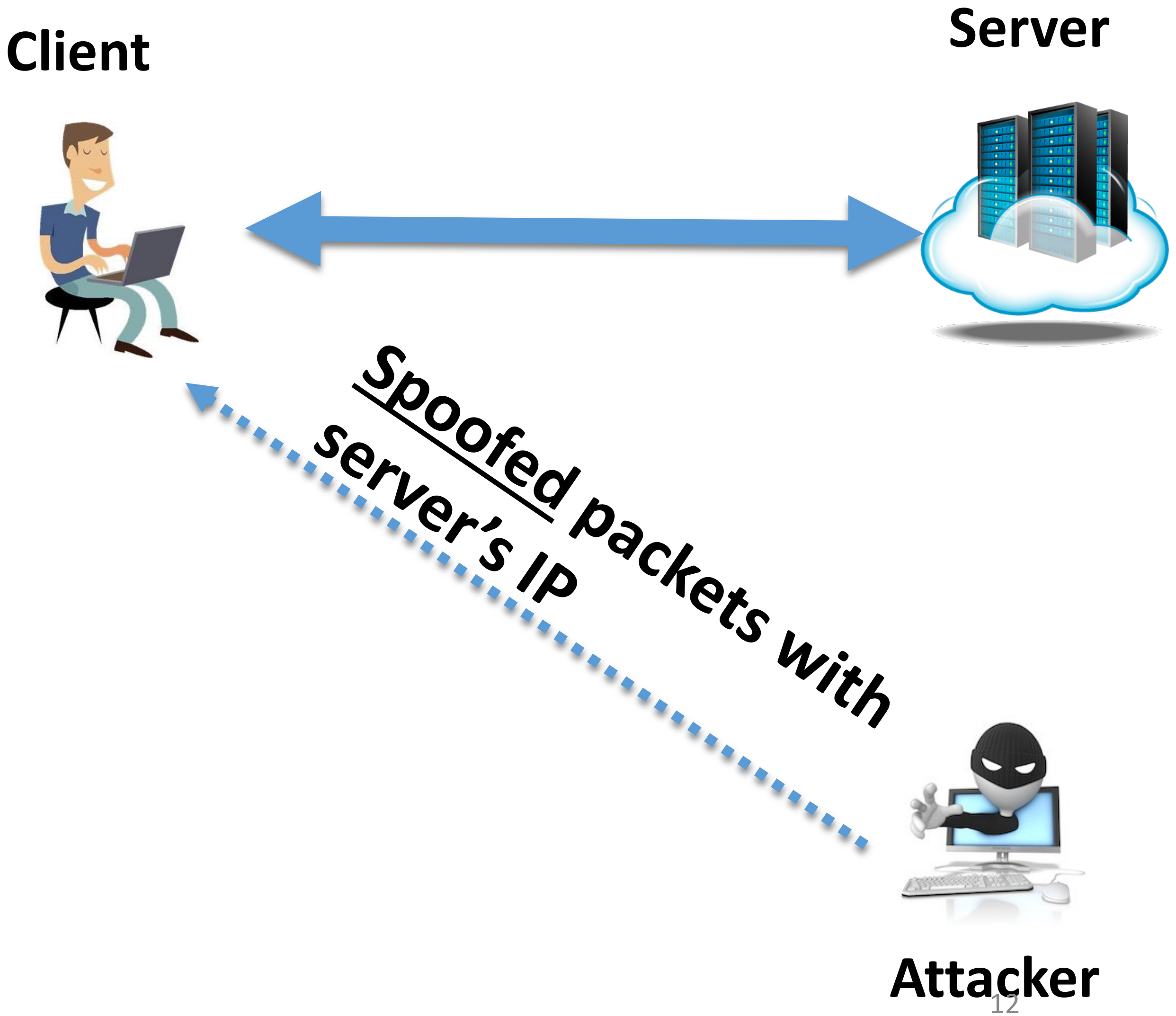
Out of window

Drop

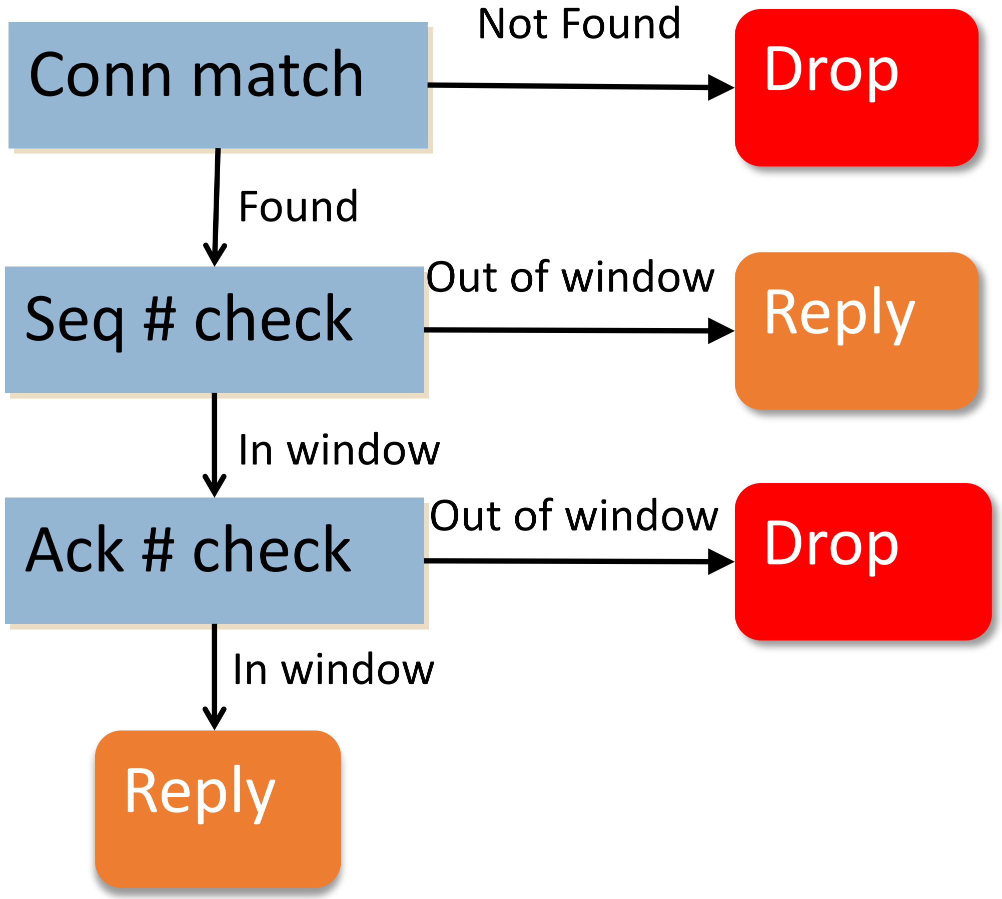
In window

Reply

**Simplified Processing Logic**



# RFC 793: TCP Packet Receiving Basics



Conn match

Not Found

Drop

Found

Seq # check

Out of window

Reply

In window

Ack # check

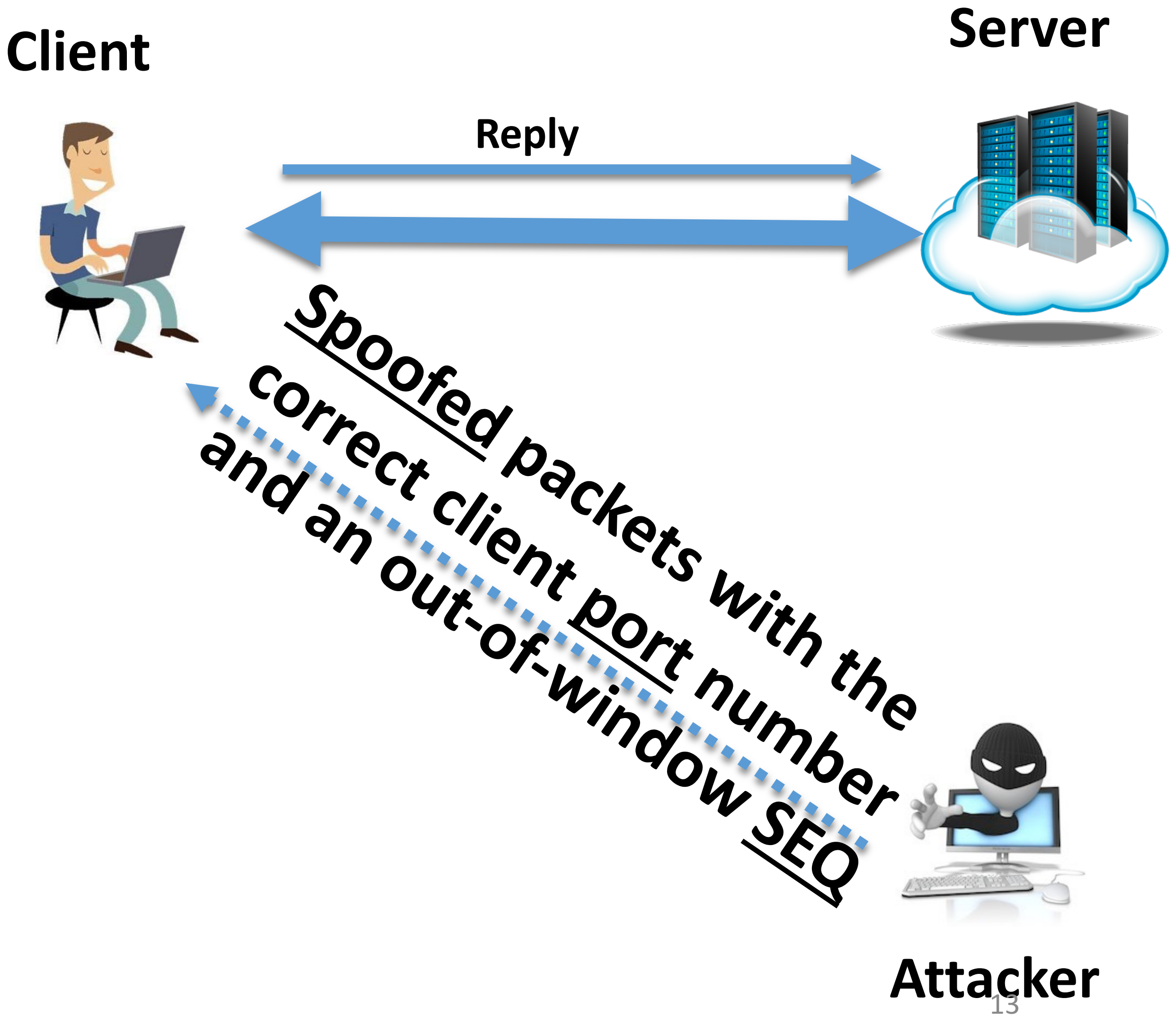
Out of window

Drop

In window

Reply

**Simplified Processing Logic**



Client

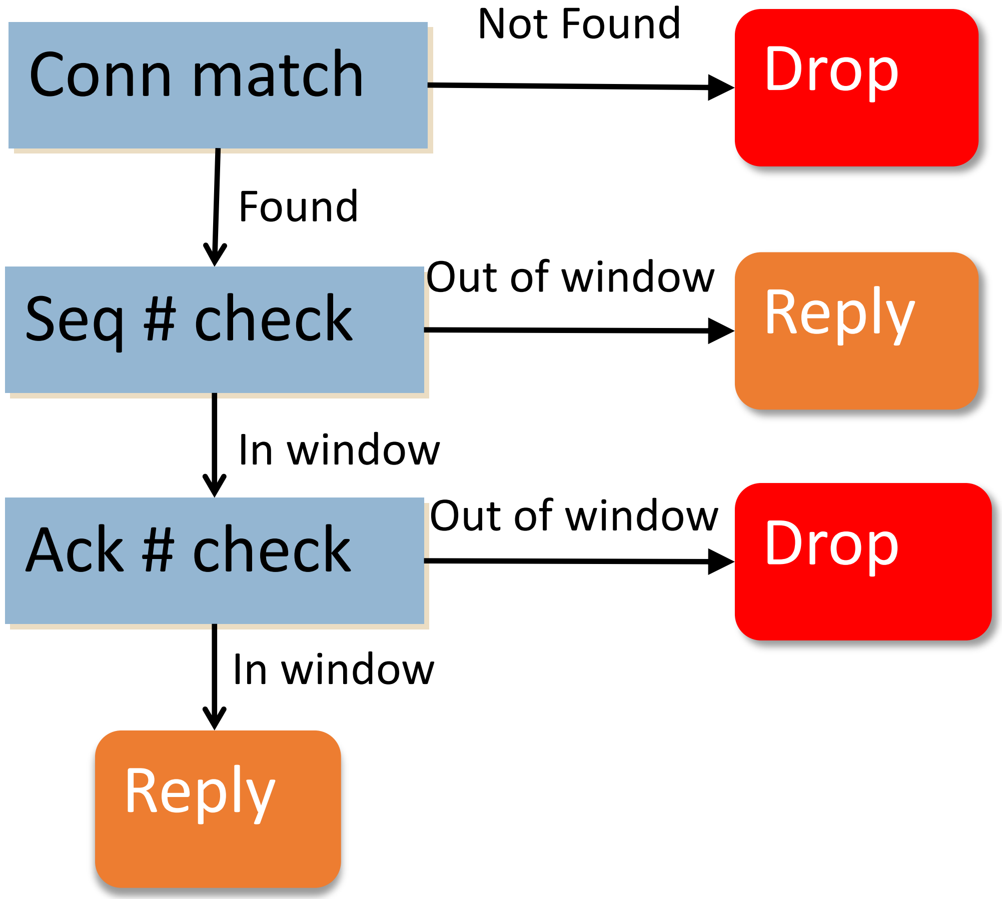
Server

Reply

Spoofed packets with the correct client port number and an out-of-window SEQ

Attacker

# RFC 793: TCP Packet Receiving Basics



Conn match

Not Found

Drop

Found

Seq # check

Out of window

Reply

In window

Ack # check

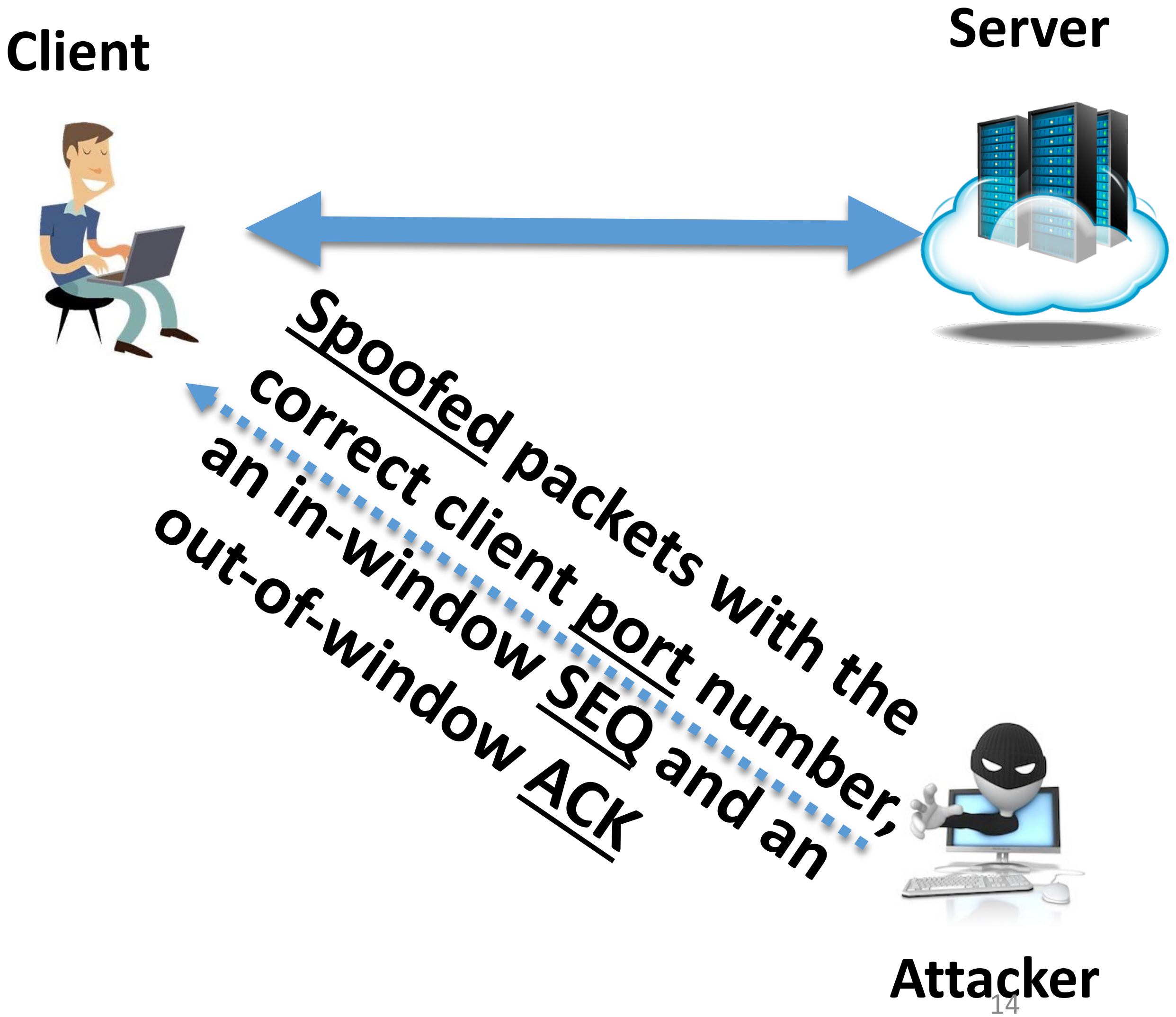
Out of window

Drop

In window

Reply

**Simplified Processing Logic**



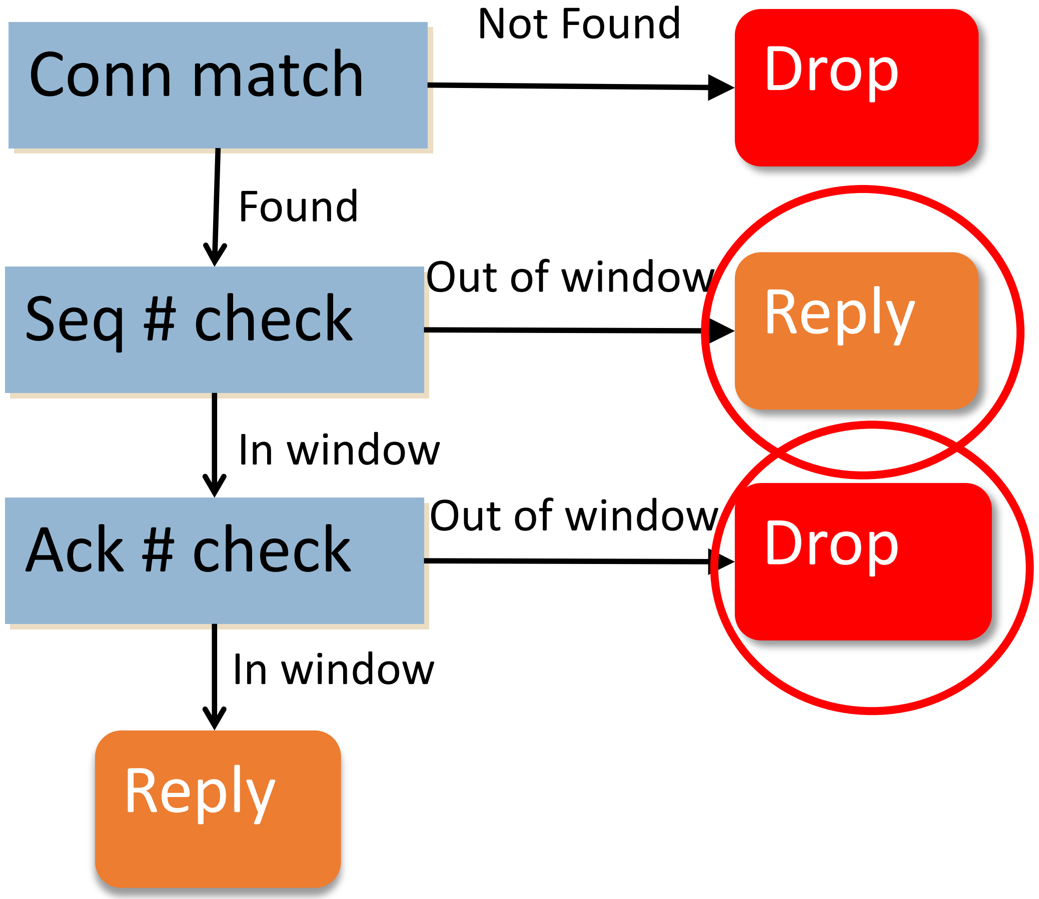
**Client**

**Server**

**Attacker**

**<u>Spoofed</u> packets with the correct client <u>port</u> number, an in-window <u>SEQ</u> and an out-of-window <u>ACK</u>**

# RFC 793: TCP Packet Receiving Basics



Conn match

Not Found

Drop

Found

Seq # check

Out of window

Reply

In window

Ack # check

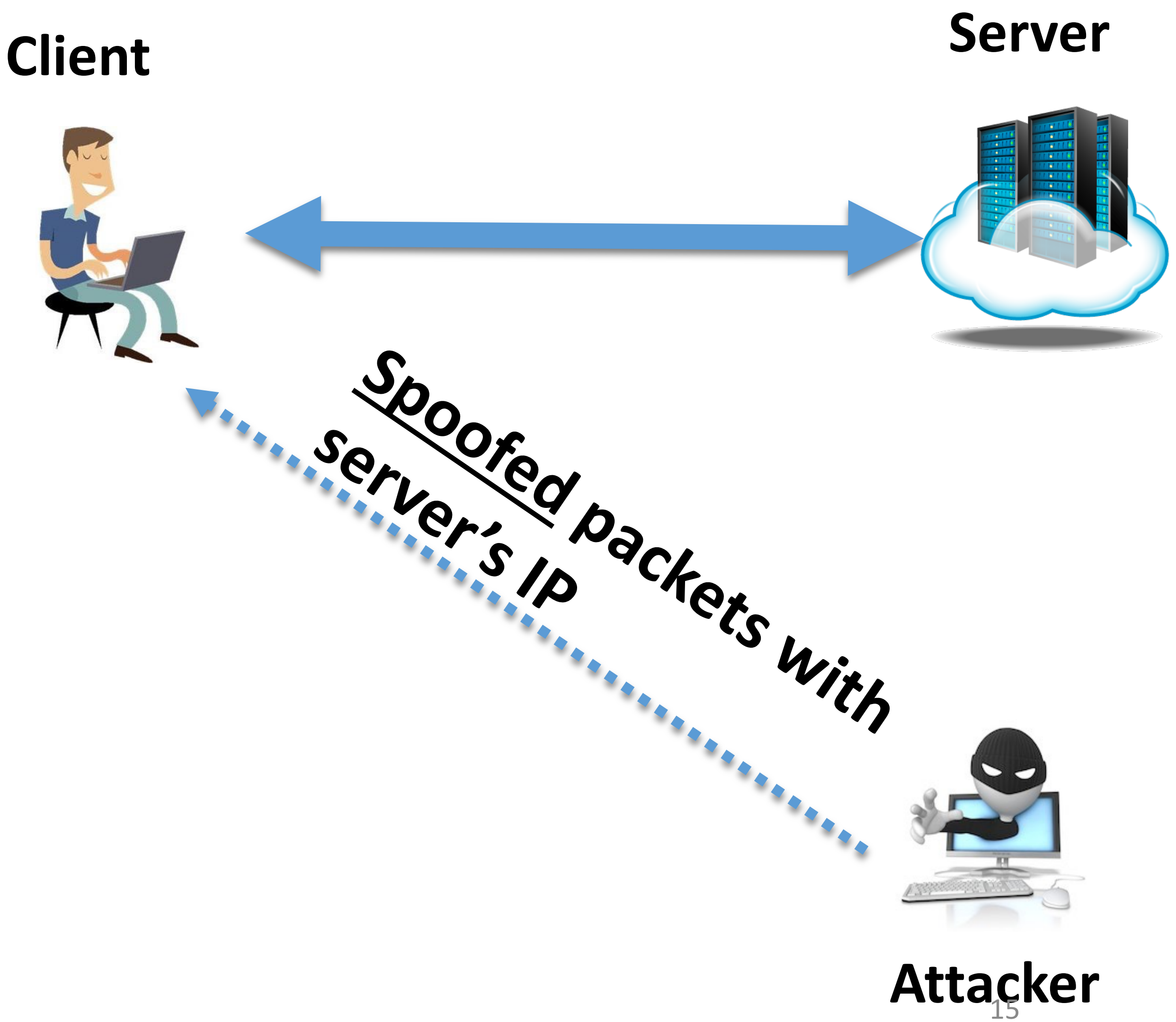
Out of window

Drop

In window

Reply

**Simplified Processing Logic**



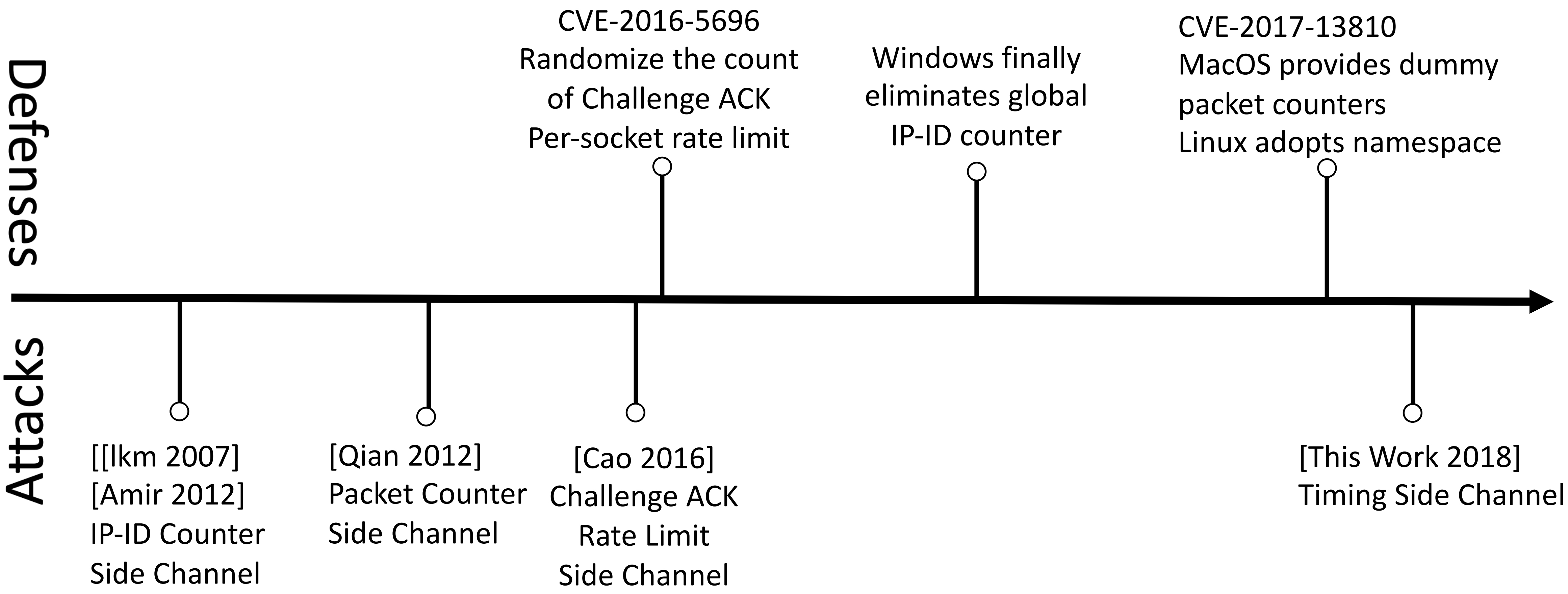
Client

Server

Spoofed packets with server's IP

Attacker

# A Time-Line of TCP Injection Attacks



Defenses

CVE-2016-5696
Randomize the count
of Challenge ACK
Per-socket rate limit

Windows finally
eliminates global
IP-ID counter

CVE-2017-13810
MacOS provides dummy
packet counters
Linux adopts namespace

Attacks

[[lkm 2007]
[Amir 2012]
IP-ID Counter
Side Channel

[Qian 2012]
Packet Counter
Side Channel

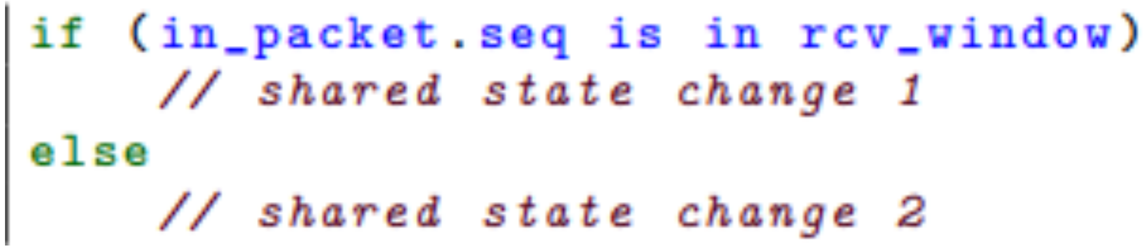
[Cao 2016]
Challenge ACK
Rate Limit
Side Channel

[This Work 2018]
Timing Side Channel

# Off-Path TCP Injection Attacks

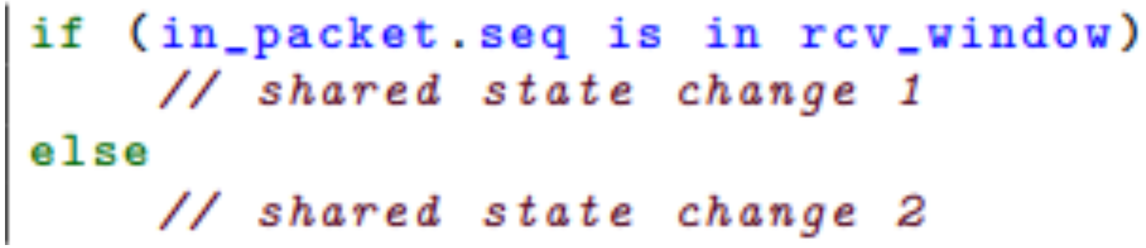
| Side Channel | Requirement | Affected OS | Patch/Mitigation |
|---|---|---|---|
| Global IP-ID counter | N/A | Windows | Global IPID counter eliminated |
| Global challenge ACK rate limit | N/A | Linux | Global rate limit eliminated |
| Packet counter | Malware | Linux, MacOS | Namespace/dummy counter |
| Wireless contention (this work) | Javascript | Any | N/A |

# Building Blocks of Side Channels



```
if (in_packet.seq is in rcv_window)
    // shared state change 1
else
    // shared state change 2
```

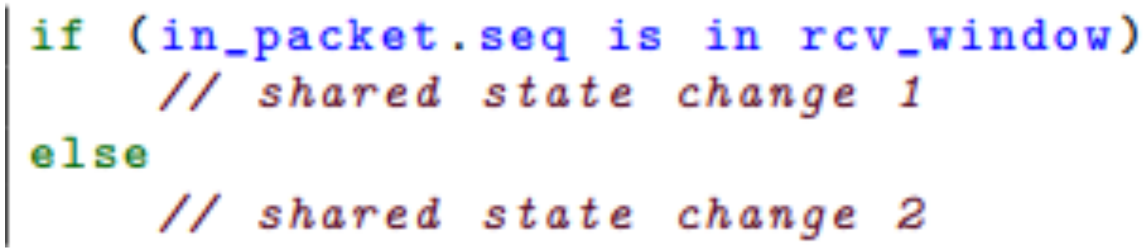
# Building Blocks of Side Channels



```
if (in_packet.seq is in rcv_window)
    // shared state change 1
else
    // shared state change 2
```

- Shared resources
  - e.g., Global IP-ID counter, Packet counter, Global challenge ACK rate limit

# Building Blocks of Side Channels



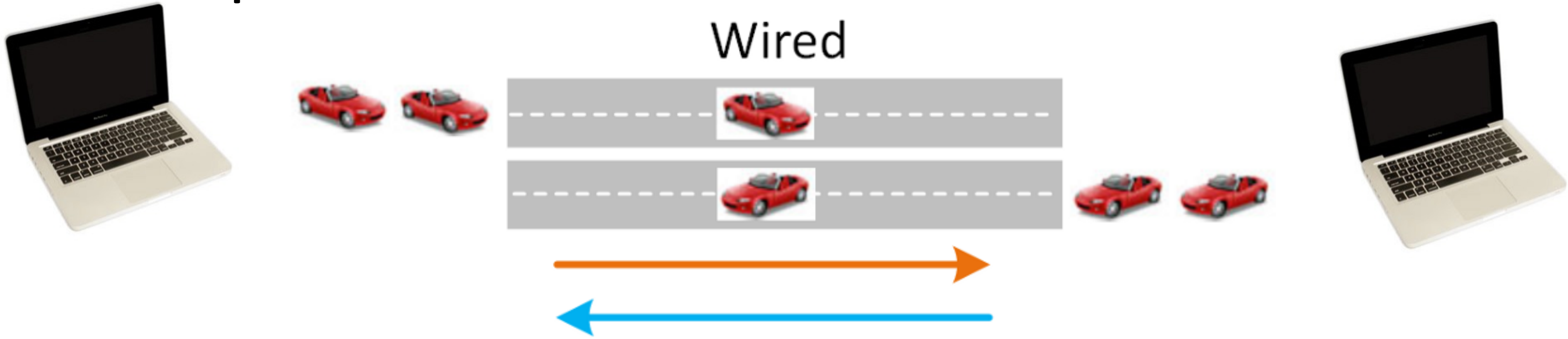
```
if (in_packet.seq is in rcv_window)
    // shared state change 1
else
    // shared state change 2
```

- Shared resources
  - e.g.,  Global IP-ID counter, Packet counter, Global challenge ACK rate limit
- Shared state changes observable to attackers
  - e.g.,  Javascript, Un-priviledged Malware

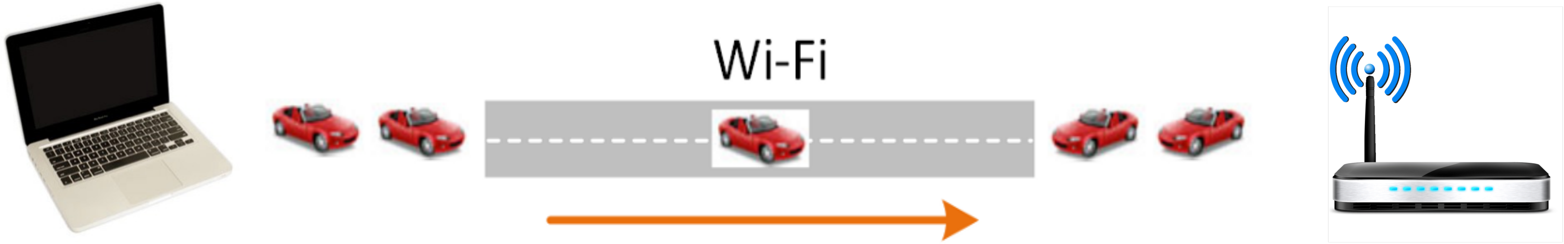
# Wireless Timing Channel

- **Half-duplex: A fundamental design of wireless protocol**
- **Shared Resource: The half-duplex wireless channel**

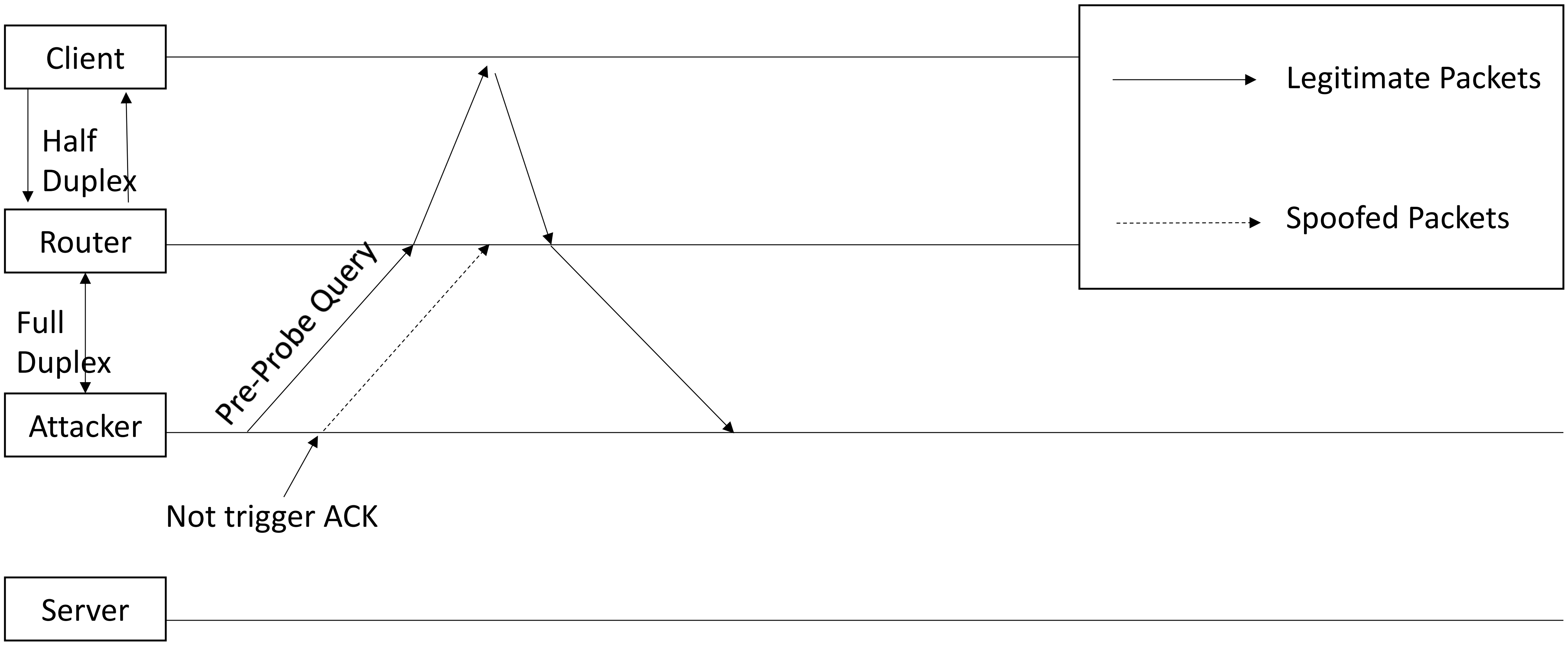
Full-duplex:



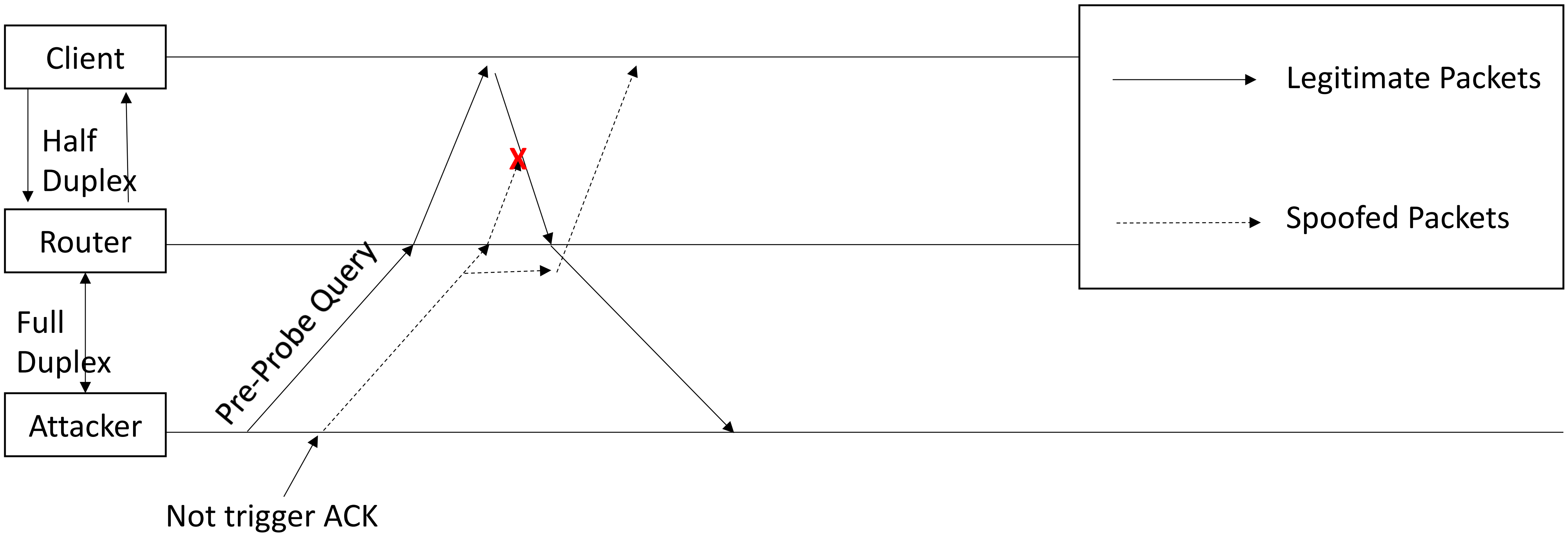
Half-duplex:



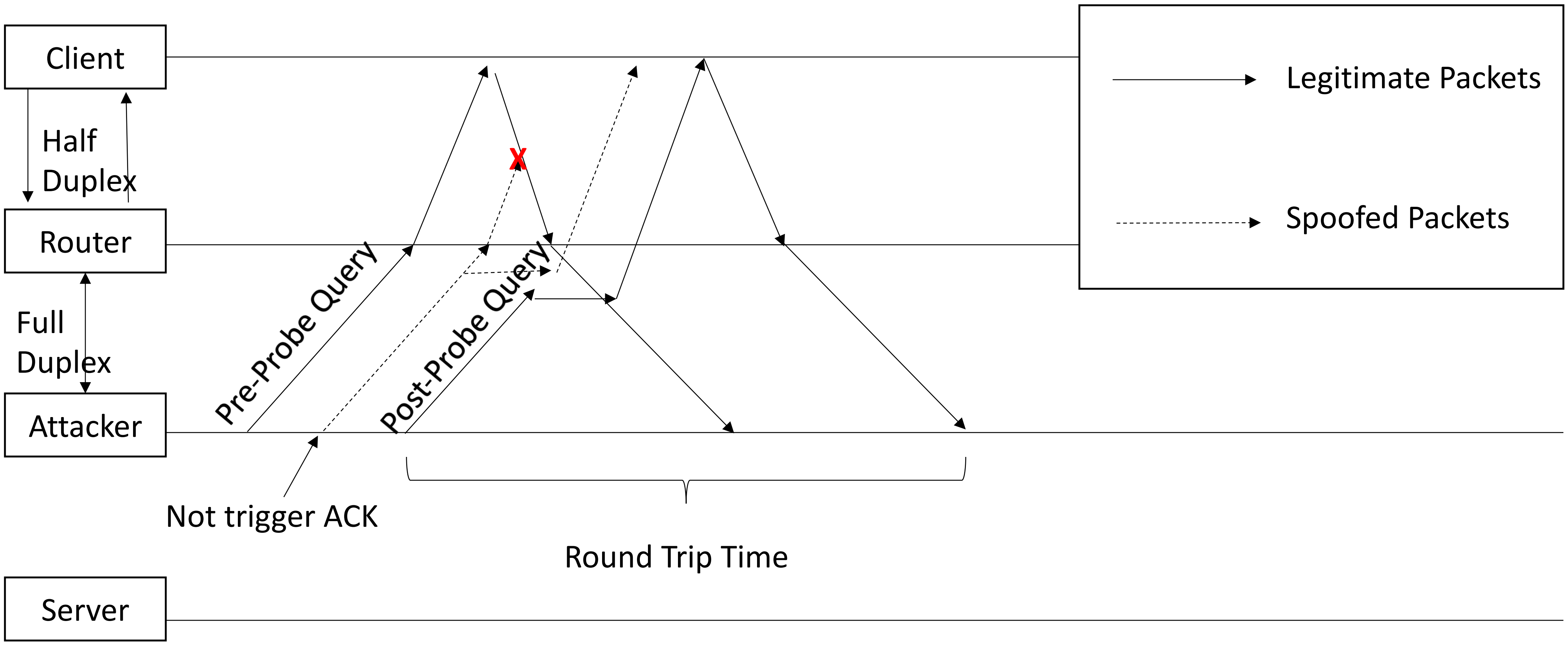
# Probing Strategy



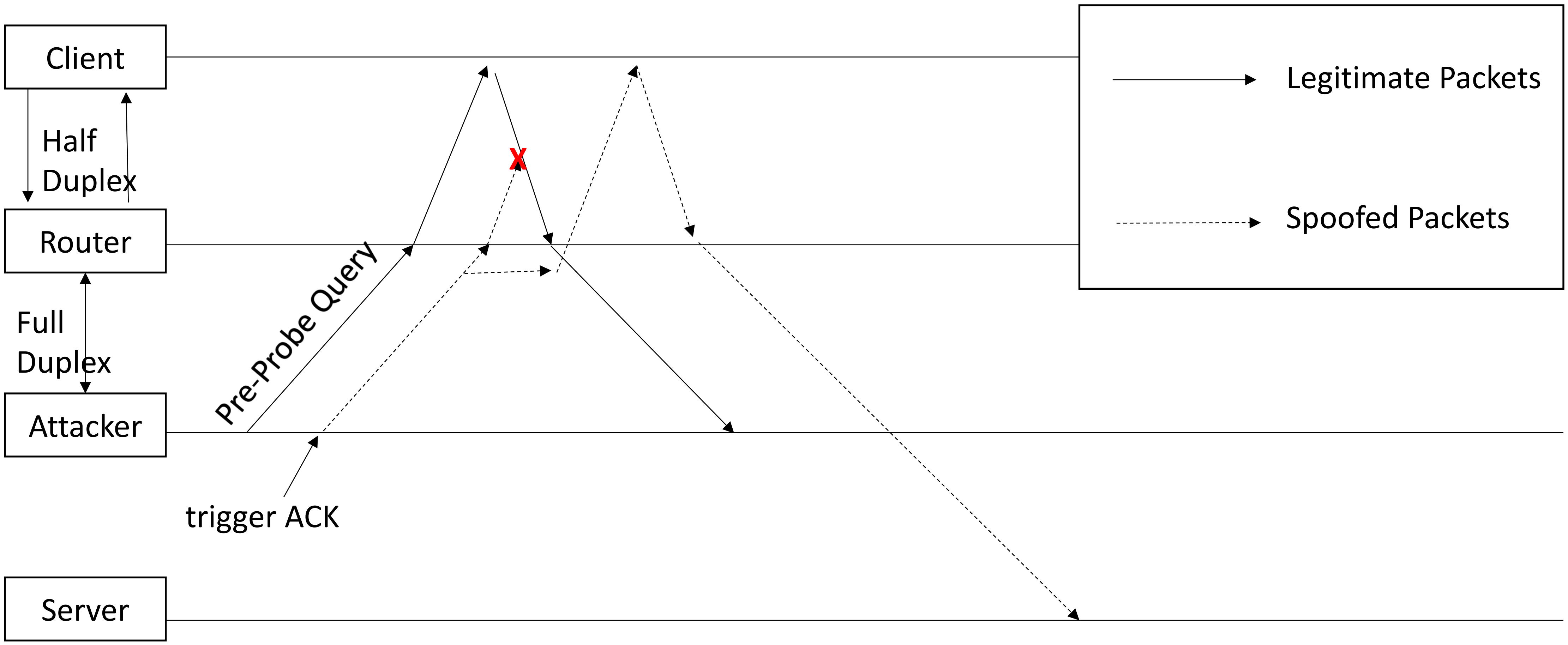
# Probing Strategy



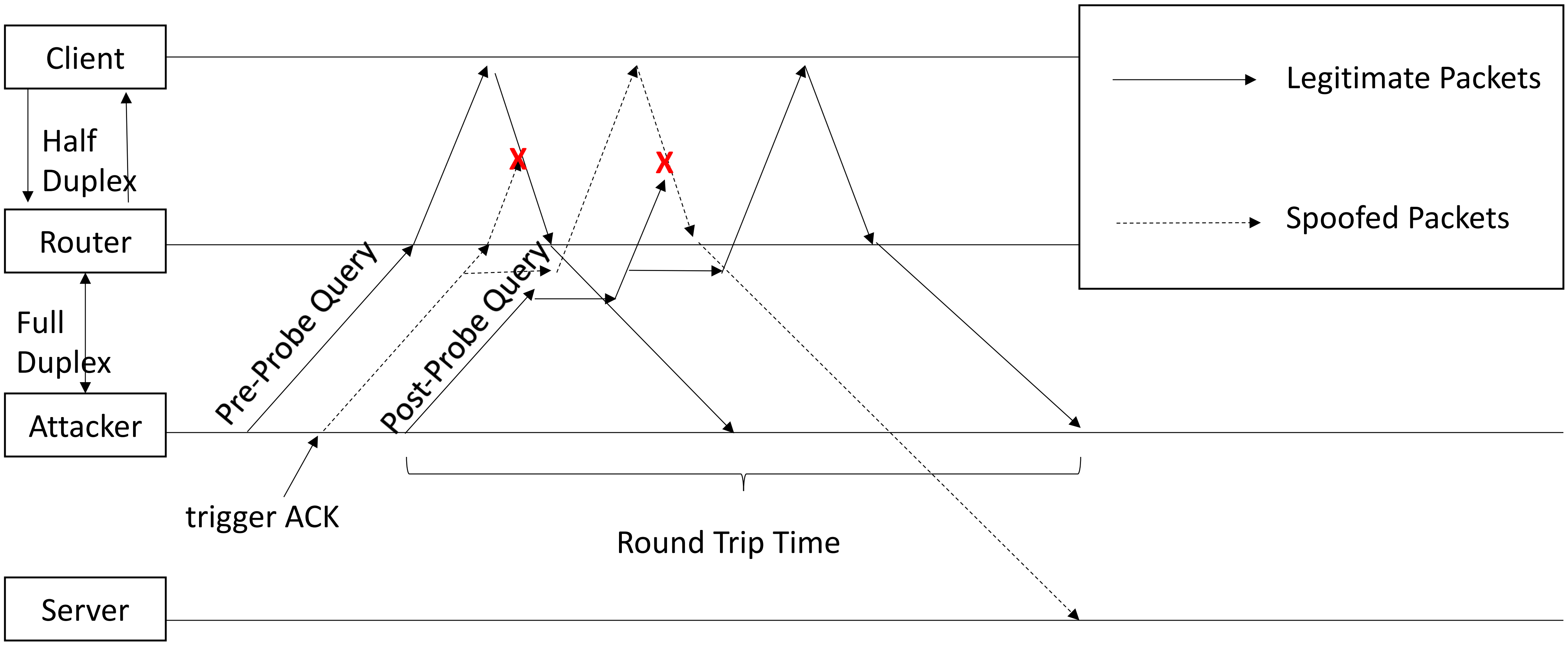
# Probing Strategy



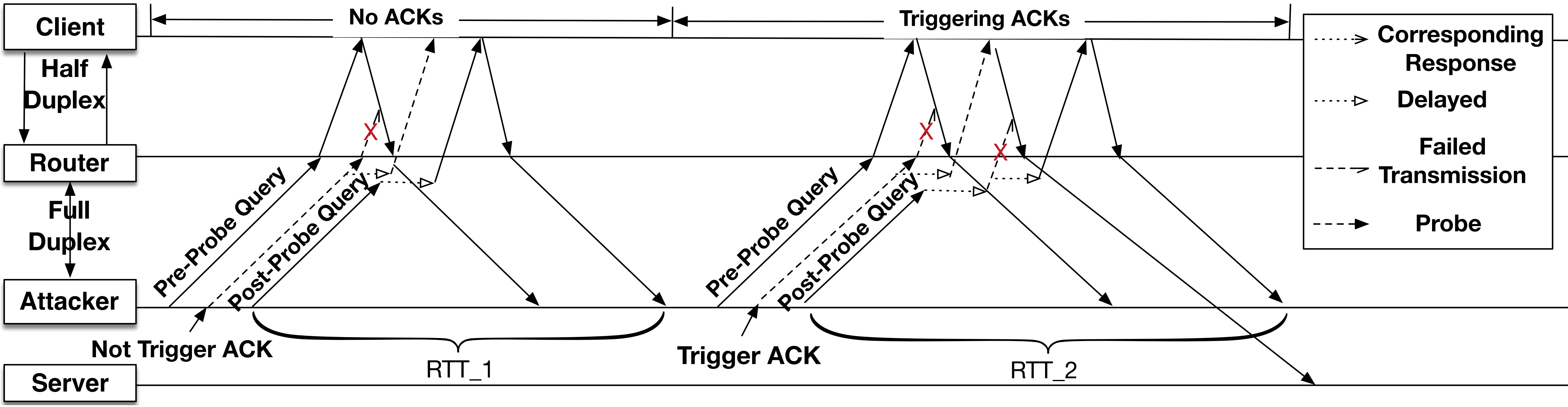
# Probing Strategy (Cont)



# Probing Strategy (Cont)

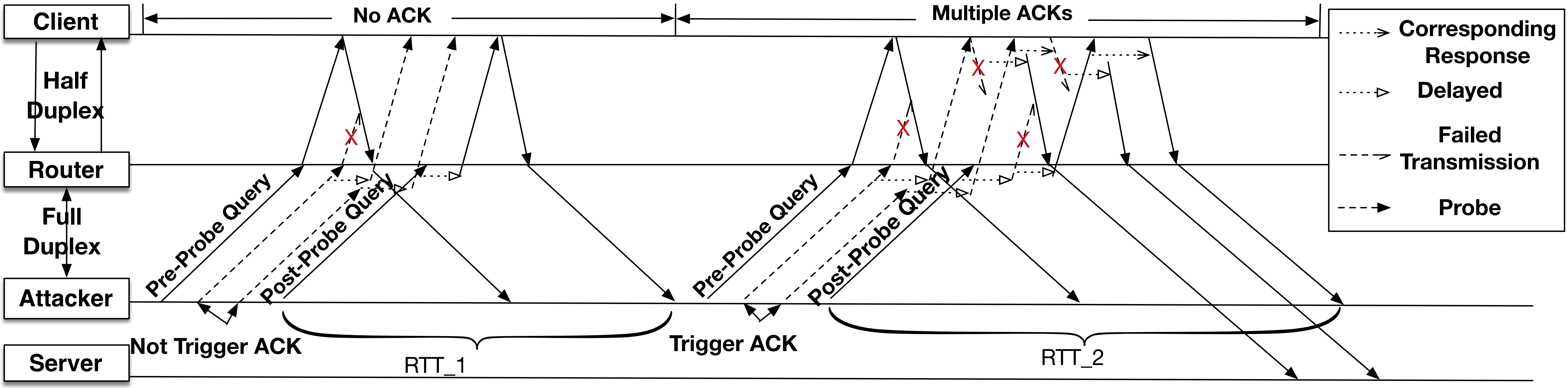


# Timing Difference



- **Larger RTT ➔ Trigger ACK ➔ Correct Sequence Number ?**

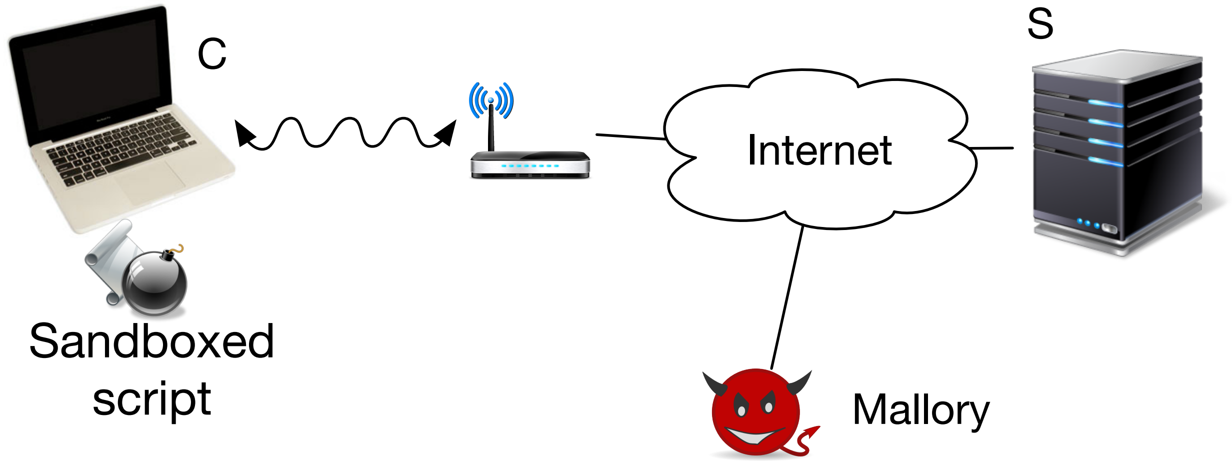
# Timing Difference (Cont)



- **More Probing Packets ➔ More Contention ➔ Larger RTTs**

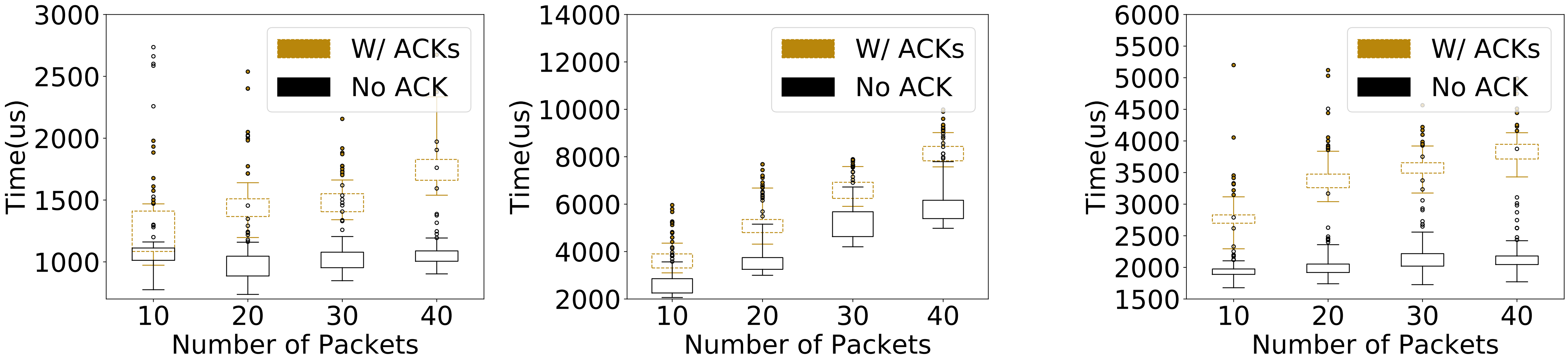
# Empirical Test Results

- Setup:



- 4 wireless routers: from Linksys, Huawei, Xiaomi, and Gee
- 2 machines: 2017 Macbook and 2017 Dell Desktop (Linux)
- 2.4GHz and 5GHz Wi-Fi

# Empirical Test Results (Cont)

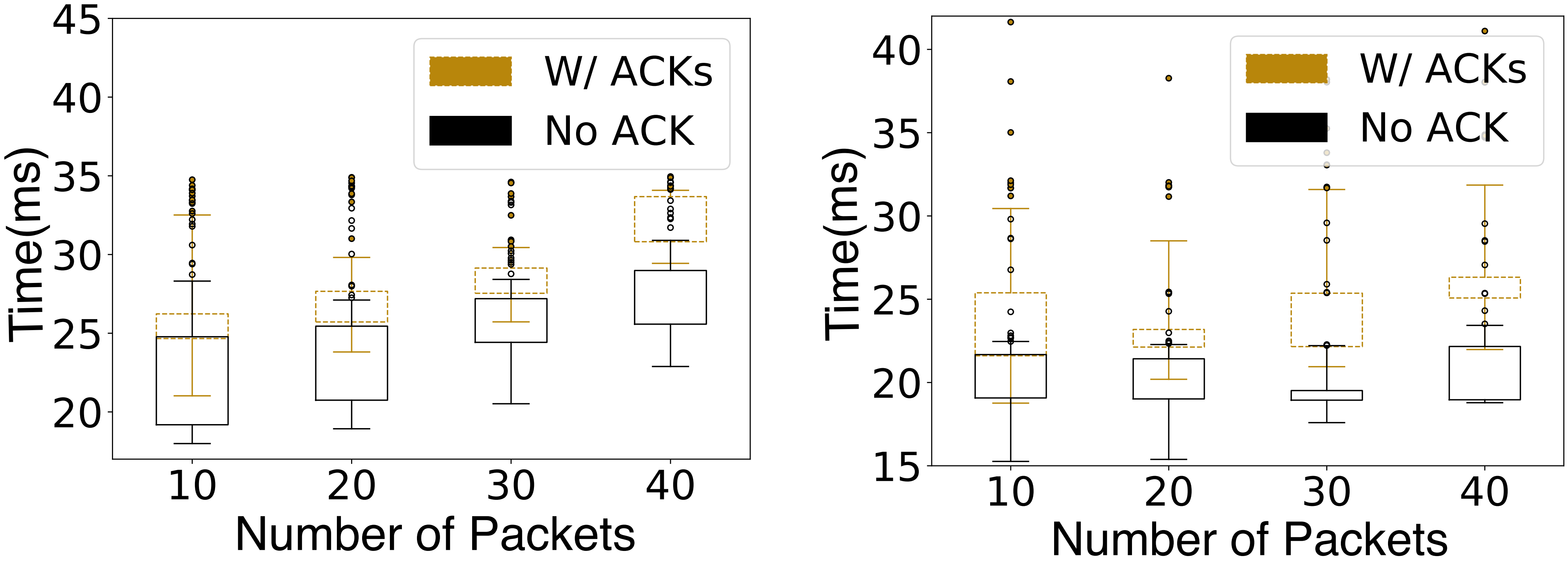


(a) RTT measurement of Linux using 5GHz network of a Linksys router

(b) RTT measurement of macOS using 2.4GHz network of a Xiaomi router

(c) RTT measurement of macOS using 5GHz network of a Huawei router

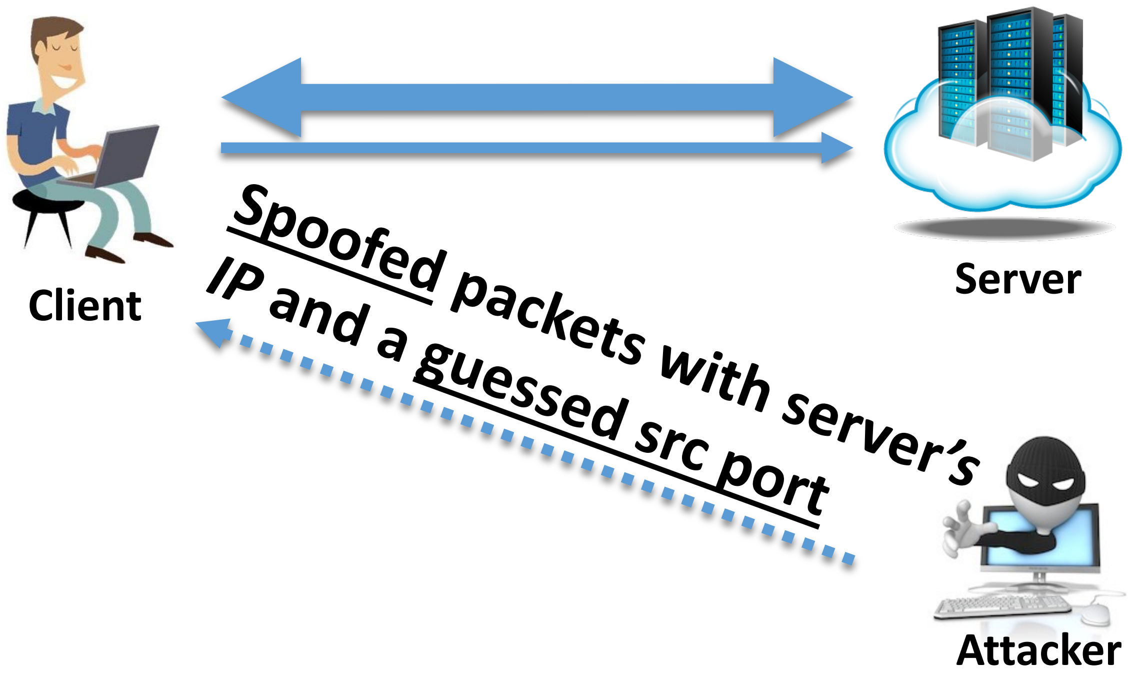
# Empirical Test Results (Cont)



**RTT measurement of macOS using 5GHz network of a Xiaomi router at two different locations with RTTs over 20ms**

# Port Number Inference

Has connection



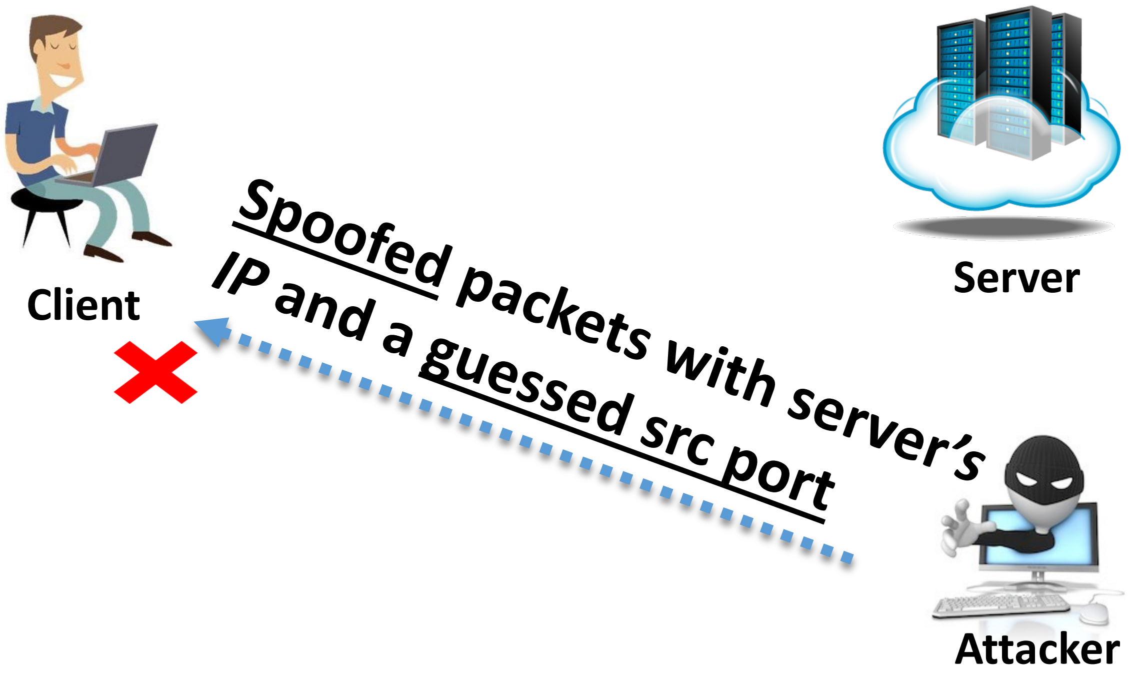
Client

Server

Spoofed packets with server's IP and a guessed src port

Attacker

No connection



Client

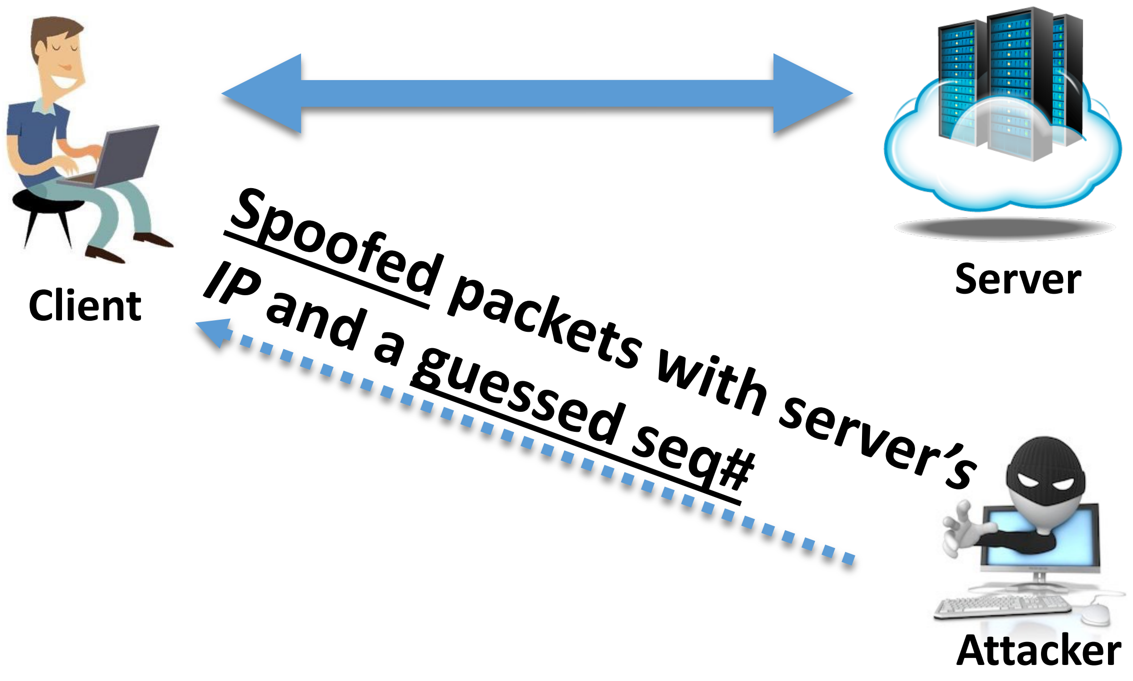
Server

Spoofed packets with server's IP and a guessed src port

Attacker

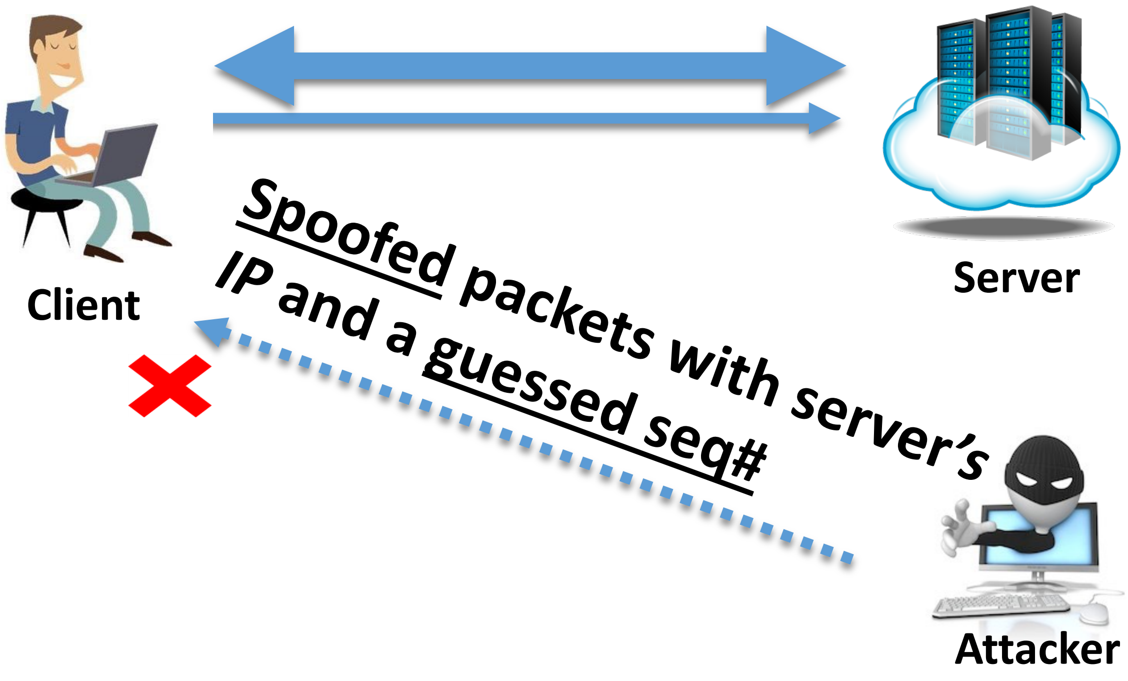
# Sequence Number Inference

Seq in-window



**Spoofed packets with server's IP and a guessed seq#**

Seq out-of-window



**Spoofed packets with server's IP and a guessed seq#**

# TCP Stack Implementations

| No. | OS | FLAG | SEQ | ACK | PAYLOAD | #Responses |
|---|---|---|---|---|---|---|
| 1 | Linux | ACK\|SYN\|RST | Out-of-window | Any | 1 | 10 |
| 3 | Linux | ACK\|SYN\|RST | In-window | > SND.MAX | Any | 0 |
| 10 | MacOS | None\|ACK | Out-of-window | Any | Any | 10 |
| 11 | MacOS | None | In-window | Out-of-window | Any | 0 |
| 17 | Windows | ACK\|FIN\|SYN | Out-of-window | Any | Any | 10 |
| 18 | Windows | ACK\|FIN | In-window | Out-of-window | Any | 0 |

Table. Behaviors on different OSes when processing 10 identical packets*

*:See the complete table in our paper

# ACK Number Inference

- Implementations of ACK number check varies significantly from one OS to another
- Exploit HTTP specifications and behaviors of tolerant browsers
  - Brute-force ACK number
- Only takes a couple of seconds

# Evaluation

| OS | Browser | Success Rate | Avg time cost (s) |
|---|---|---|---|
| Linux | Chrome/Firefox | 10/10 | 188.80 |
| MacOS | Chrome/Firefox | 10/10 | 48.91 |
| Windows | Chrome/Firefox | 10/10 | 43.42 |

Local result

| OS | Browser | Success Rate | Avg time cost (s) |
|---|---|---|---|
| MacOS | Chrome/Firefox | 9/10 | 304.18 |

Remote result (RTT = 20ms)

# How bad?

- Teleconference with IEEE 802.11 working group
- **It’s not possible to be fixed at physical and MAC layers!**

# Defenses/Mitigations

- Wireless Layer: Full-duplex Wi-Fi Technology
  - E.g., Frequency-division duplexing, different frequency sub-bands

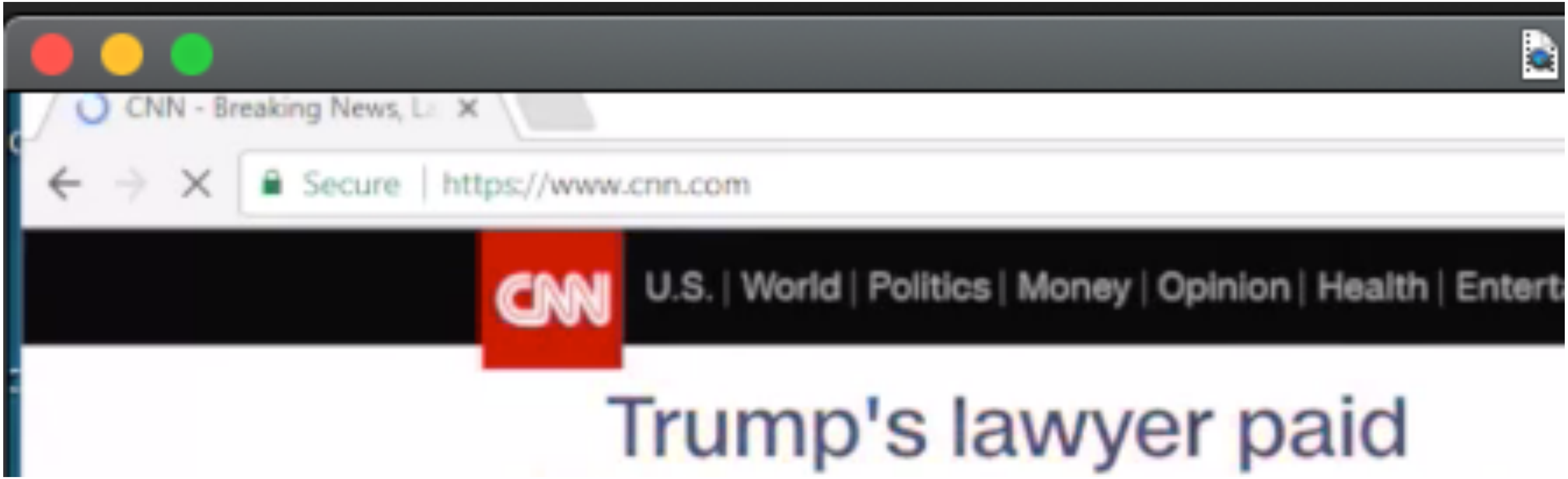
# Defenses/Mitigations

- Wireless Layer: Full-duplex Wi-Fi Technology
  - E.g., Frequency-division duplexing, different frequency sub-bands
- TCP Stack: Revisit TCP Specifications
  - E.g., Rate limit responses for incoming packets with out-of-window SEQ

# Defenses/Mitigations

- Wireless Layer: Full-duplex Wi-Fi Technology
  - E.g., Frequency-division duplexing, different frequency sub-bands
- TCP Stack: Revisit TCP Specifications
  - E.g., Rate limit responses for incoming packets with out-of-window SEQ
- Application Layer: Deploy HSTS (HTTP Strict Transport Security)
  - Preventing access via the insecure HTTP protocol

# Defenses/Mitigations



Trump's lawyer paid

# Conclusion

- A new timing side channel inherent in all generations of IEEE 802.11 or Wi-Fi technology
- Comprehensive analysis of TCP stack implementations in macOS, Windows, and Linux
- Implement practical TCP injection attacks
- Propose possible defenses
- https://github.com/seclab-ucr/tcp_exploit

# Q&A

Thanks for your attention!