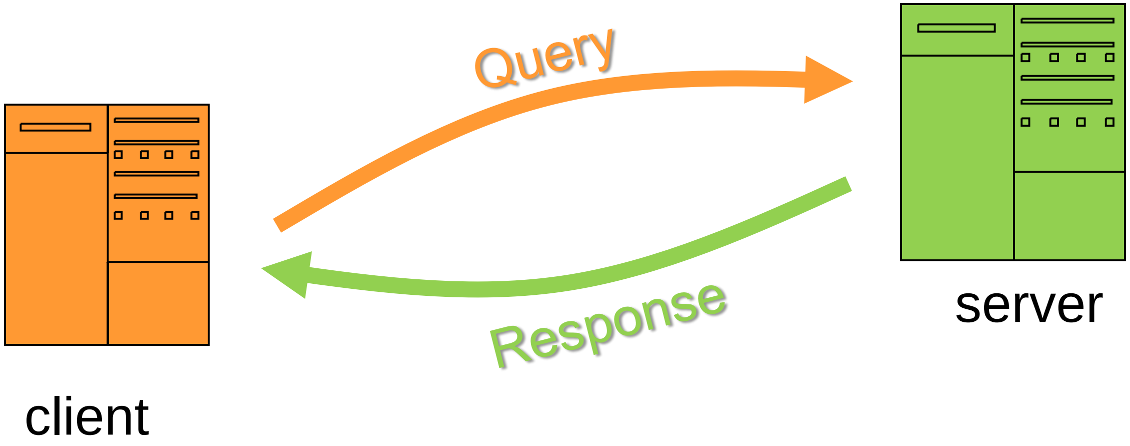
NDSS, 2016

# Attacking the Network Time Protocol (NTP)



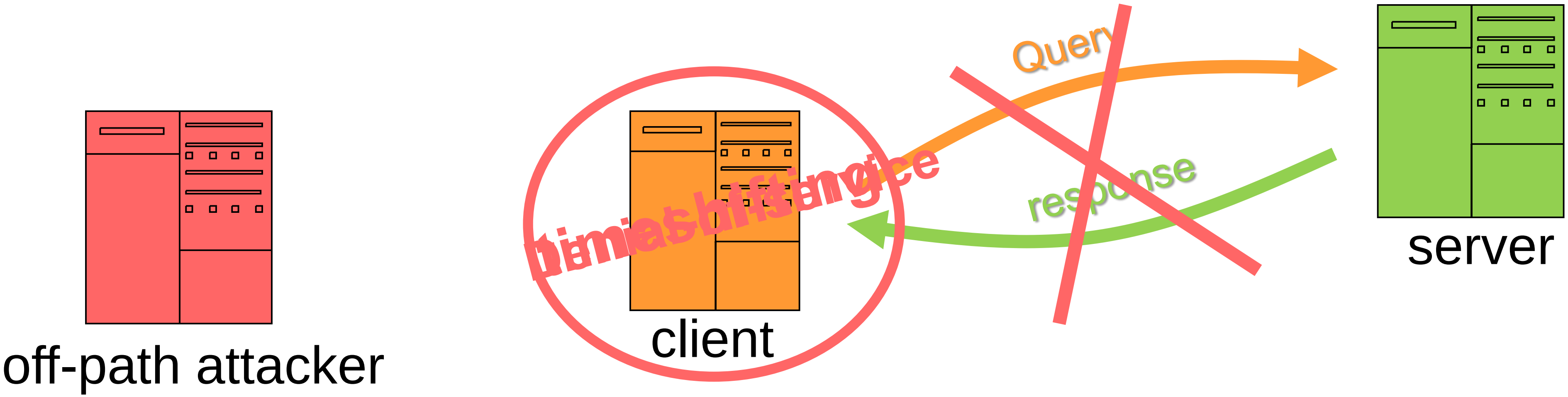
BOSTON UNIVERSITY

Aanchal Malhotra
Isaac E. Cohen, Erik Brakke
Sharon Goldberg

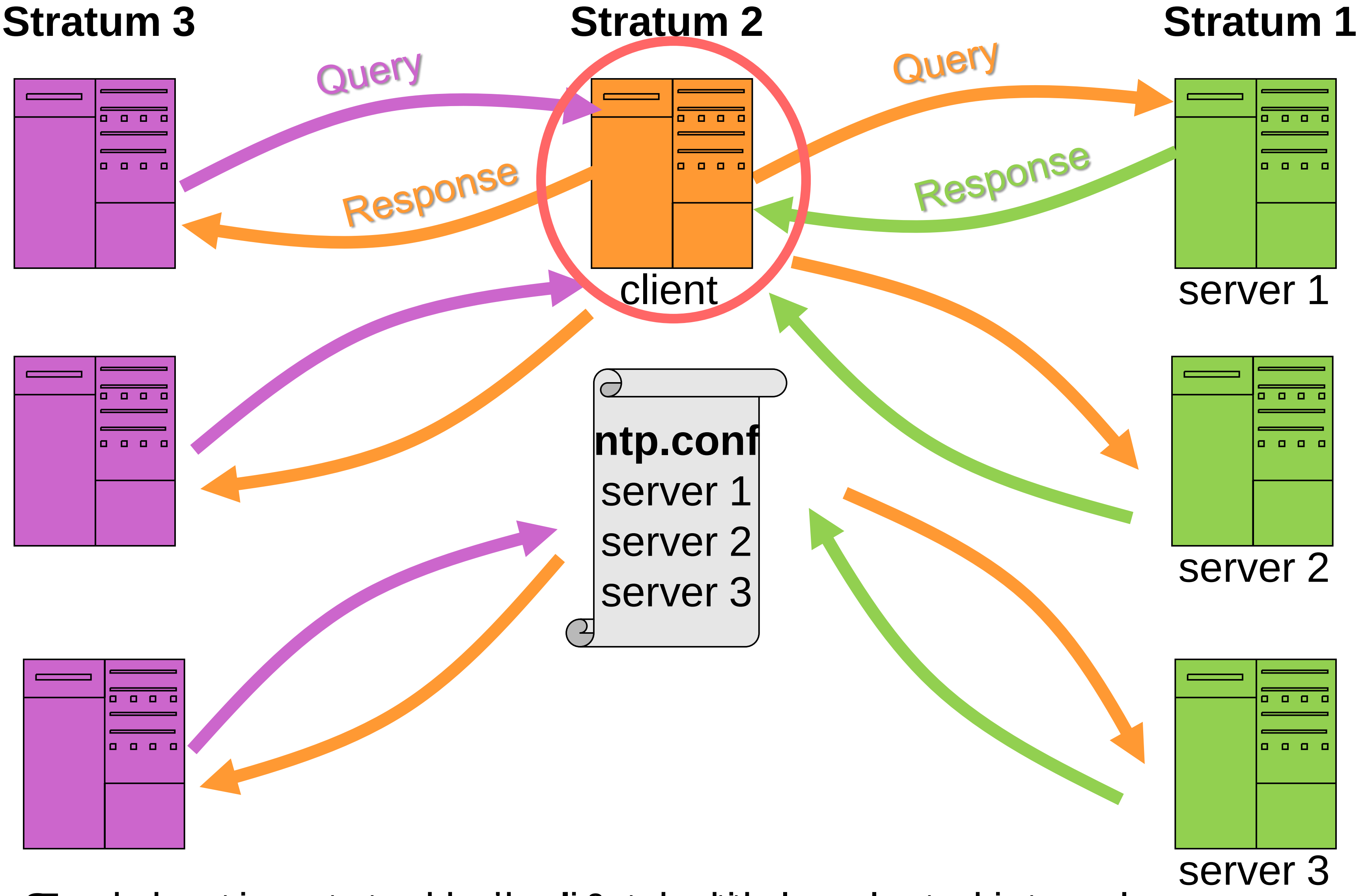
# Outline of the talk

- Background
  - How does NTP work?
  - How does NTP client take time?

- **Our attacks**
  - Denial of Service by Spoofed Kiss-of-Death (off-path)
  - Denial of Service by Priming the Pump (off-path)
  - Timeshifting by IPv4 Packet Fragmentation (off-path)



# Background: How does NTP work?



- Sends queries at randomized & adaptively selected intervals
- Requires a number of self-consistent responses to update its clock

- Every host can act as both client and server
- My laptop will answer queries from public Internet.

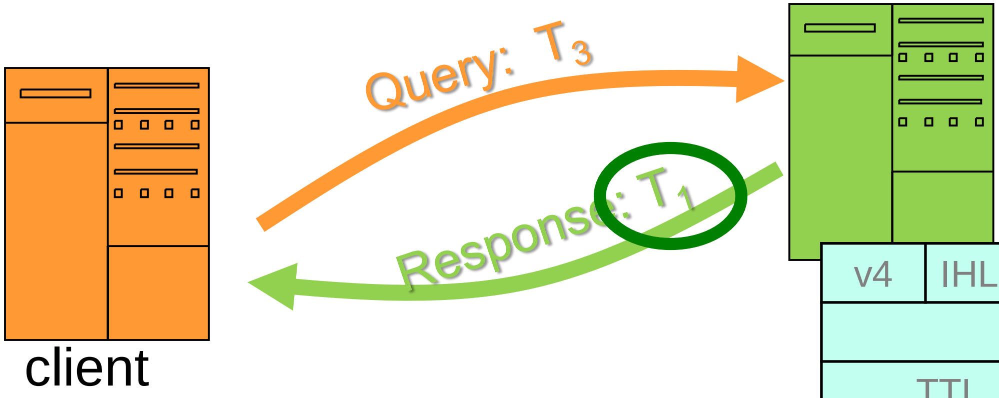
**We assume NTP messages are not cryptographically authenticated.**

(Ask me why after.)

**We attack the NTPv4 spec (RFC5905)**

**and its reference implementation (ntpd v4.2.8p2 & ntpd v4.2.6p5)**

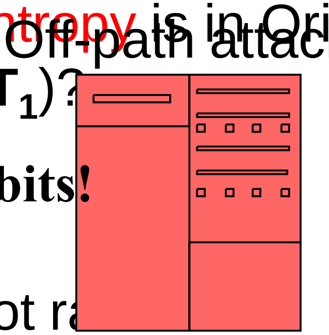
# Non-Crypto Authentication with Origin Timestamp (T$_1$)



Analogous to
- UDP source port randomization
- TCP sequence no randomization

**TEST2**: Match
**T3 in Query** to **T1 in Response**.

How much entropy is in Origin Timestamp (**T$_1$**)?



**≈ 32 bits!**

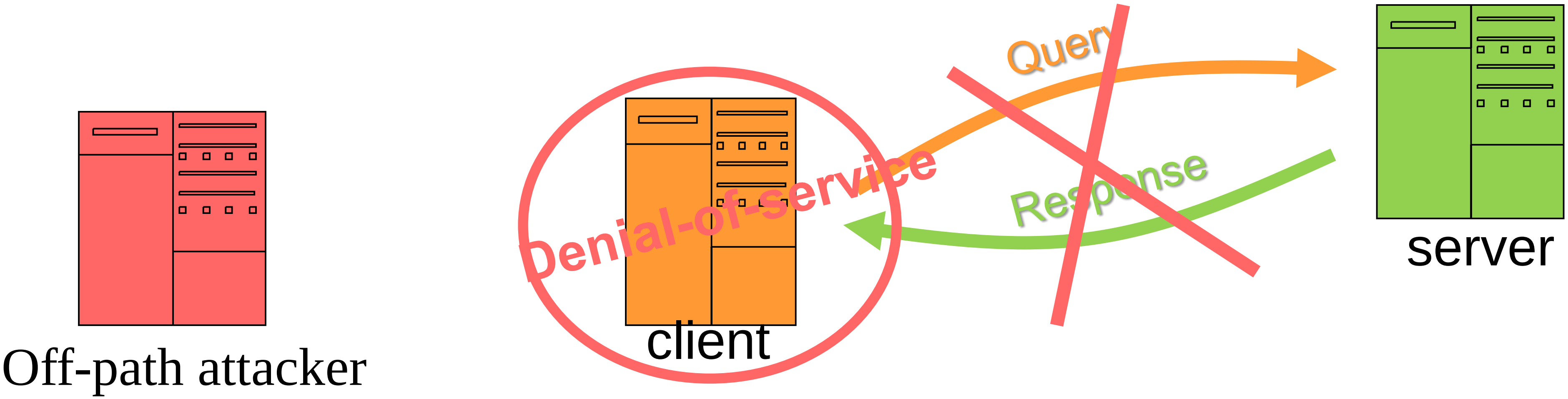
*ntpd does not ra

UDP source port!

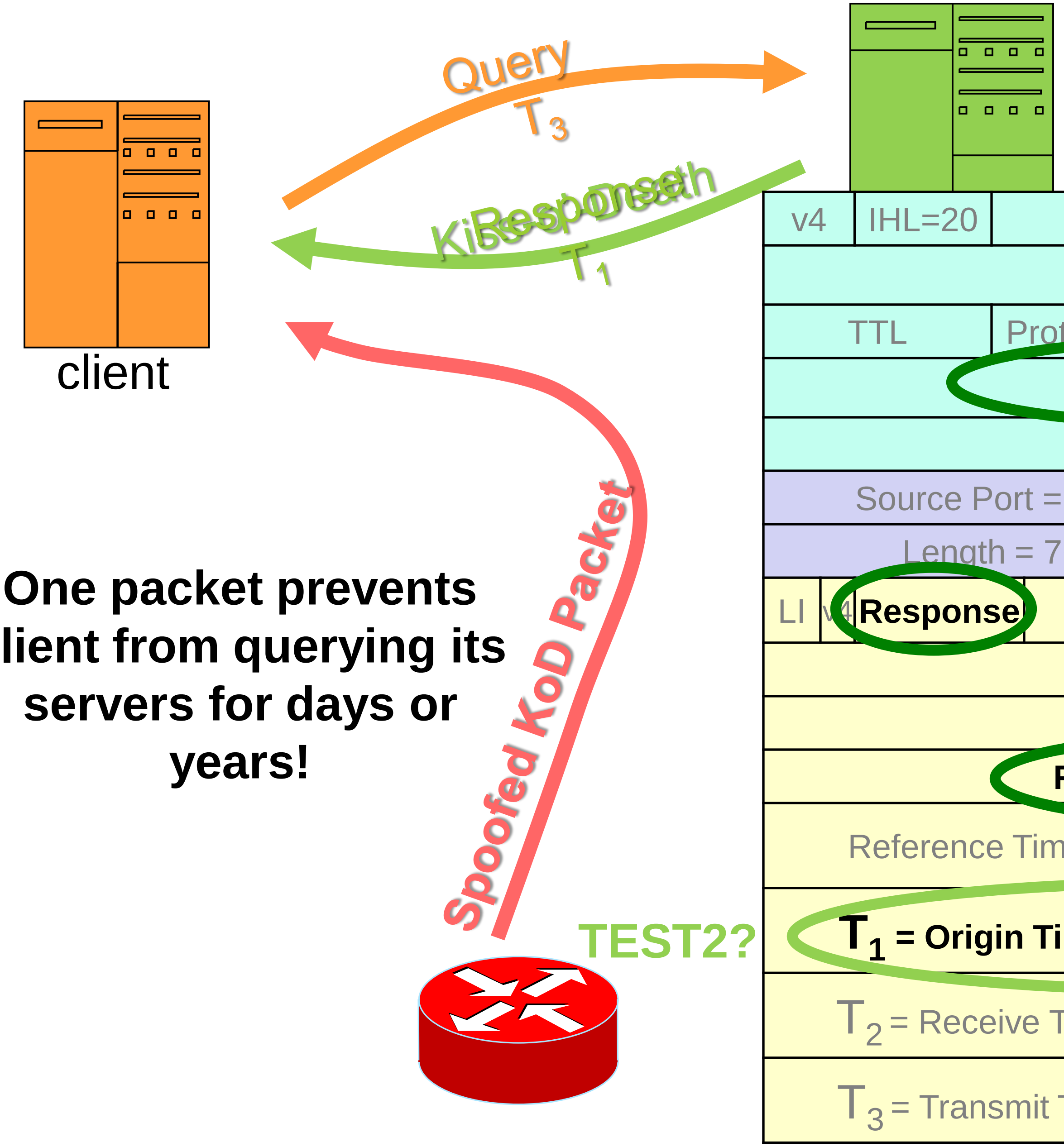
<table>
<tr><td colspan="2">v4</td><td>IHL=20</td><td colspan="2">TOS</td><td colspan="5">Total length = 76</td></tr>
<tr><td colspan="5">IPID</td><td>x</td><td>DF</td><td>MF</td><td colspan="2">Frag Offset</td></tr>
<tr><td colspan="3">TTL</td><td colspan="2">Protocol = 17</td><td colspan="5">IP Header Checksum</td></tr>
<tr><td colspan="10">Source IP</td></tr>
<tr><td colspan="10">Destination IP</td></tr>
<tr><td colspan="5">**Source Port = 123**</td><td colspan="5">Destination Port = 123</td></tr>
<tr><td colspan="5">Length = 76</td><td colspan="5">UDP Checksum</td></tr>
<tr><td>LI</td><td>v4</td><td colspan="2">Response</td><td>Stratum</td><td colspan="4">Poll</td><td>**Precision**</td></tr>
<tr><td colspan="10">Root Delay</td></tr>
<tr><td colspan="10">Root Dispersion</td></tr>
<tr><td colspan="10">Reference ID</td></tr>
<tr><td colspan="10">Reference Timestamp</td></tr>
<tr><td colspan="10">**T$_1$ = Origin Timestamp**</td></tr>
<tr><td colspan="10">T$_2$ = Receive Timestamp</td></tr>
<tr><td colspan="10">T$_3$ = Transmit Timestamp</td></tr>
</table>

# Outline of the talk

- Background
  - How does NTP work?
  - How does NTP client take time?
- **Our attacks**
  - **Denial of Service** by Spoofed Kiss-of-Death (off-path)
  - **Denial of Service** by Priming the Pump (off-path)
  - Timeshifting by IPv4 packet fragmentation (off-path)



# Denial of Service via Spoofed Kiss-o-Death



**Kiss-o'-Death (KoD)**

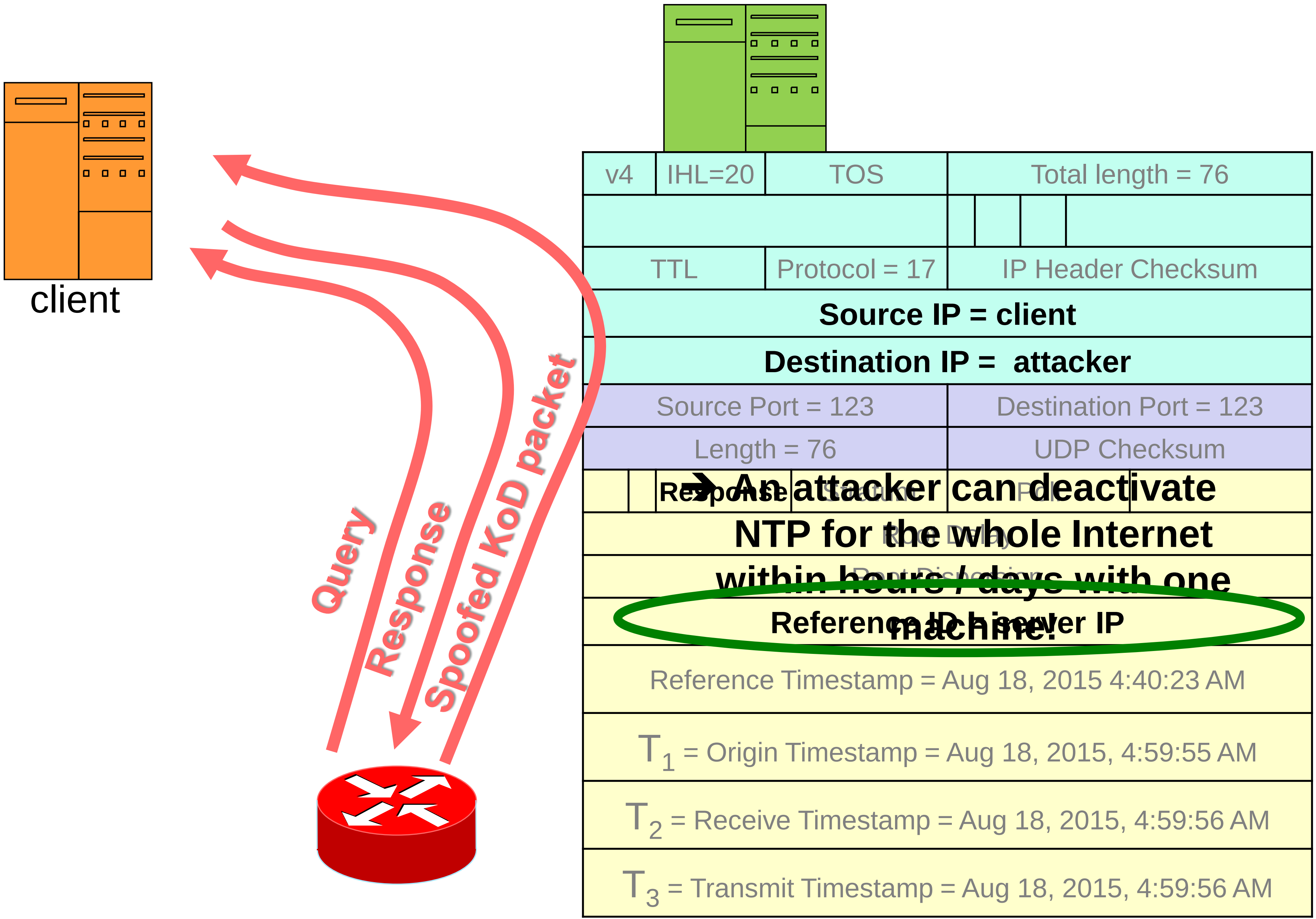
**"Keep quiet for $2^{poll}$ $2^{17}$ sec!"**

**(36 hours!)**

| | | | | | |
|---|---|---|---|---|---|
| v4 | IHL=20 | TOS | Total length = 76 | | |
| | | | | | |
| TTL | Protocol = 17 | | IP Header Checksum | | |
| **Source IP** | | | | | |
| Destination IP | | | | | |
| Source Port = 123 | | | Destination Port = 123 | | |
| Length = 76 | | | UDP Checksum | | |
| LI | v4 | **Response** | Stratum | **Poll** | |
| Root Delay | | | | | |
| Root Dispersion | | | | | |
| **Reference ID = RATE** | | | | | |
| Reference Timestamp = Jan 1, 1970 0:00:00 UTC | | | | | |
| **$T_1$ = Origin Timestamp = July 29, 2015 01:23:45** | | | | | |
| $T_2$ = Receive Timestamp = July 29, 2015 01:23:45 | | | | | |
| $T_3$ = Transmit Timestamp = July 29, 2015 01:23:45 | | | | | |

**One packet prevents client from querying its servers for days or years!**

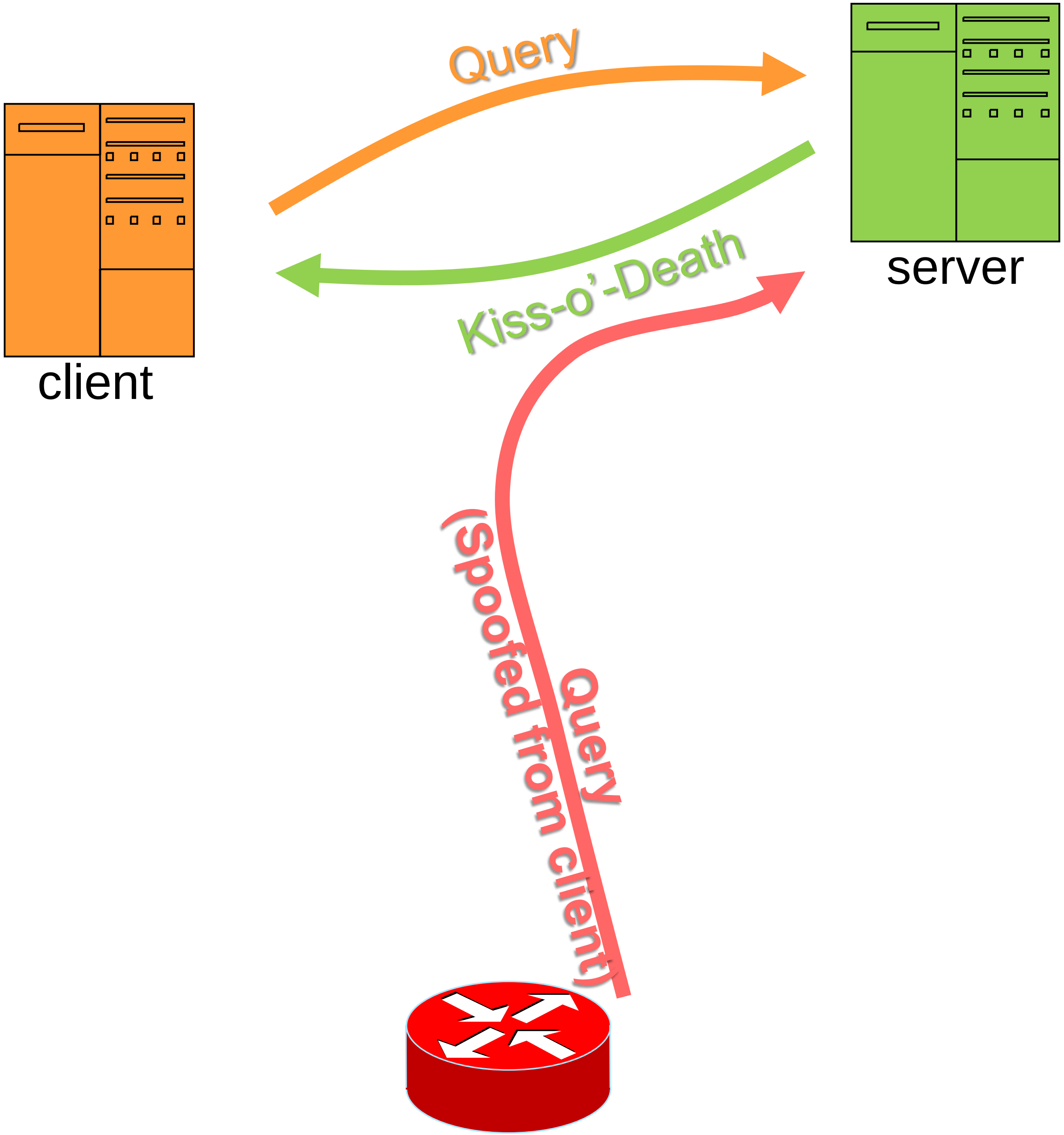
# How to learn the server’s IP for the spoofed KoD?



| | | | |
|---|---|---|---|
| v4 | IHL=20 | TOS | Total length = 76 |
| | | | |
| TTL | | Protocol = 17 | IP Header Checksum |
| **Source IP = client** | | | |
| **Destination IP = attacker** | | | |
| Source Port = 123 | | | Destination Port = 123 |
| Length = 76 | | | UDP Checksum |
| **Response** | | Stratum | Poll |
| Root Delay | | | |
| Root Dispersion | | | |
| **Reference ID = server IP** | | | |
| Reference Timestamp = Aug 18, 2015 4:40:23 AM | | | |
| $T_1$ = Origin Timestamp = Aug 18, 2015, 4:59:55 AM | | | |
| $T_2$ = Receive Timestamp = Aug 18, 2015, 4:59:56 AM | | | |
| $T_3$ = Transmit Timestamp = Aug 18, 2015, 4:59:56 AM | | | |

**➔ An attacker can deactivate NTP for the whole Internet within hours / days with one machine!**

# Denial of Service by Priming-the-Pump



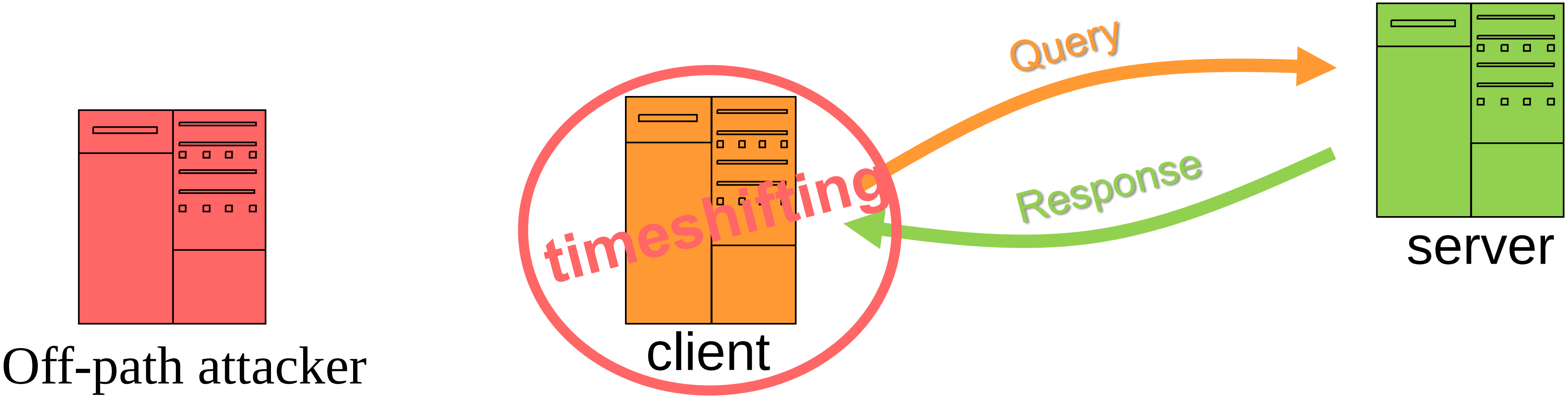
1. **Denial of Service by Spoofed Kiss-of-Death (off-path)**

**Patched!**
**ntpd 4.2.8p4**

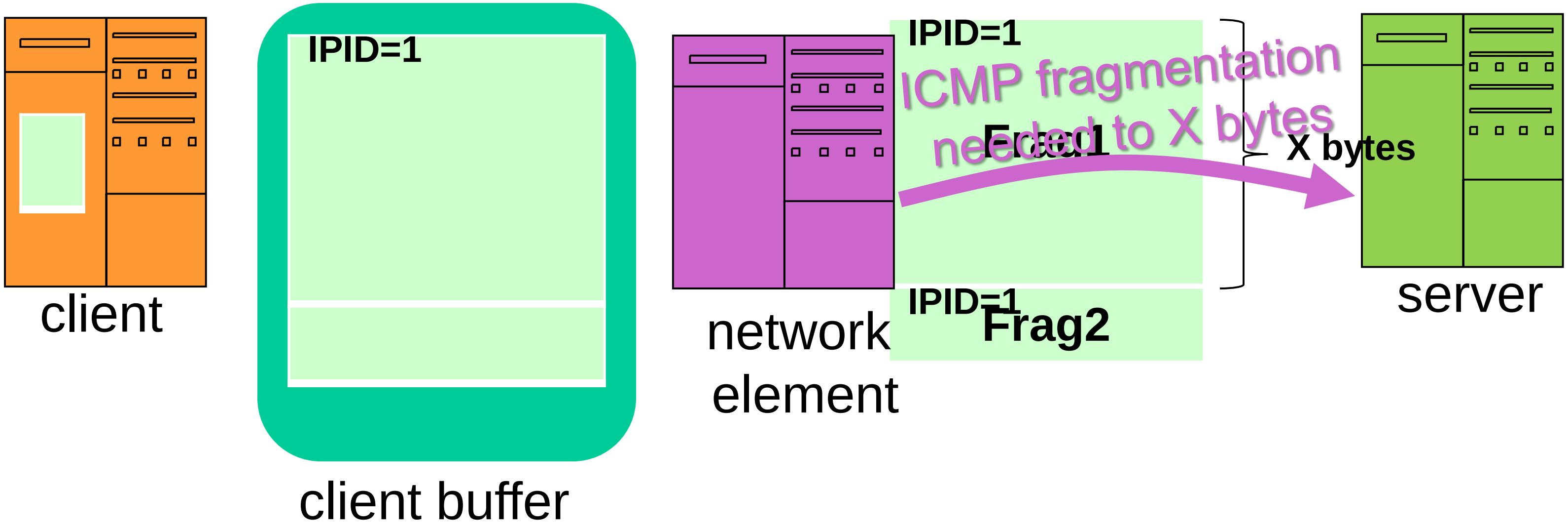
2. **Denial of Service by Priming the Pump (off-path)**

# Outline of the talk

- Background
  - How does NTP work?
  - How does NTP client take time?
- **Our attacks**
  - Denial of Service by Spoofed Kiss-of-Death (off-path)
  - Denial of Service by Priming the Pump (off-path)
  - **Timeshifting** by IPv4 packet fragmentation (off-path)



# Background: IPv4 Packet Fragmentation



client

IPID=1

client buffer

network element

IPID=1

Frag1

ICMP fragmentation needed to X bytes

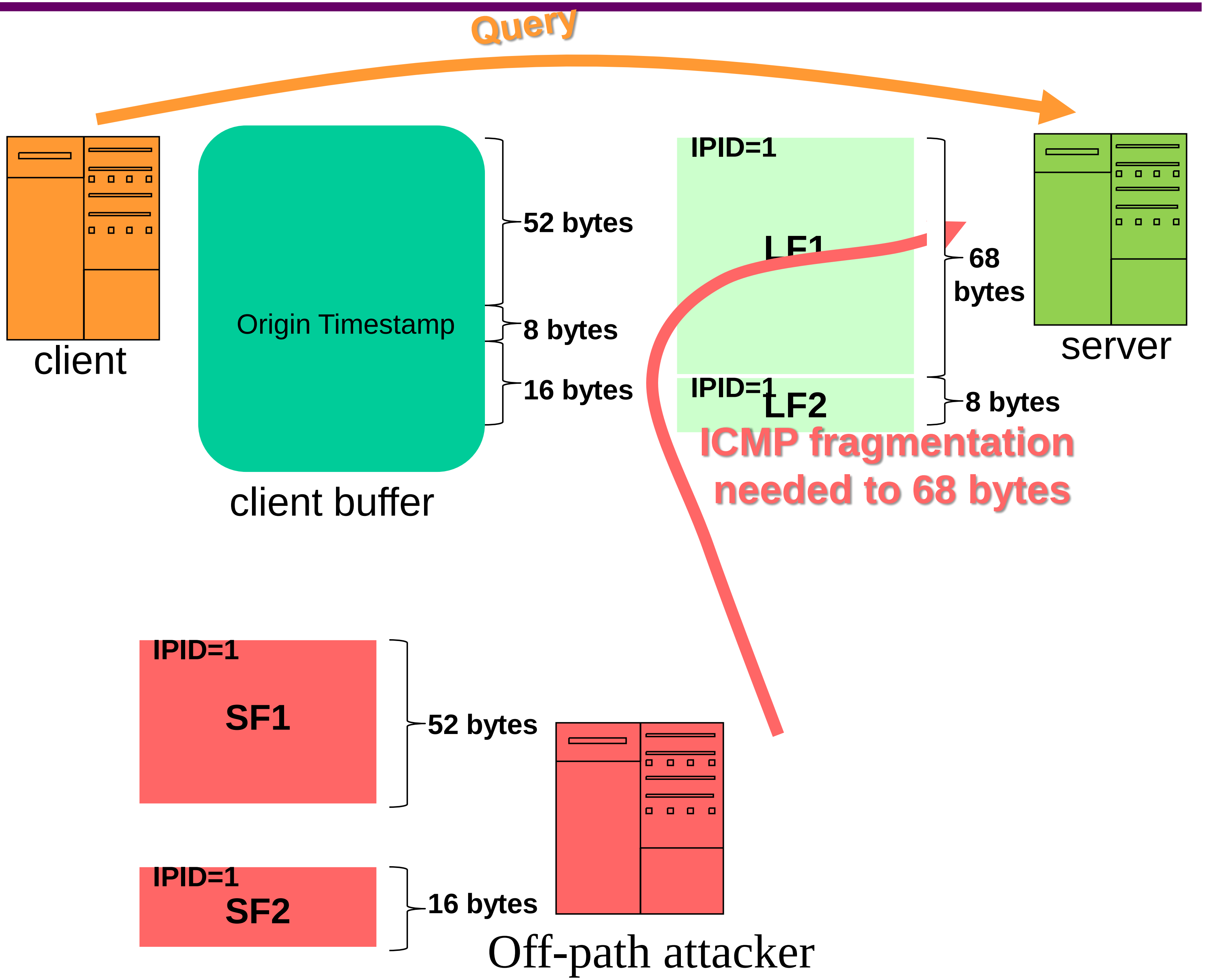
X bytes

IPID=1

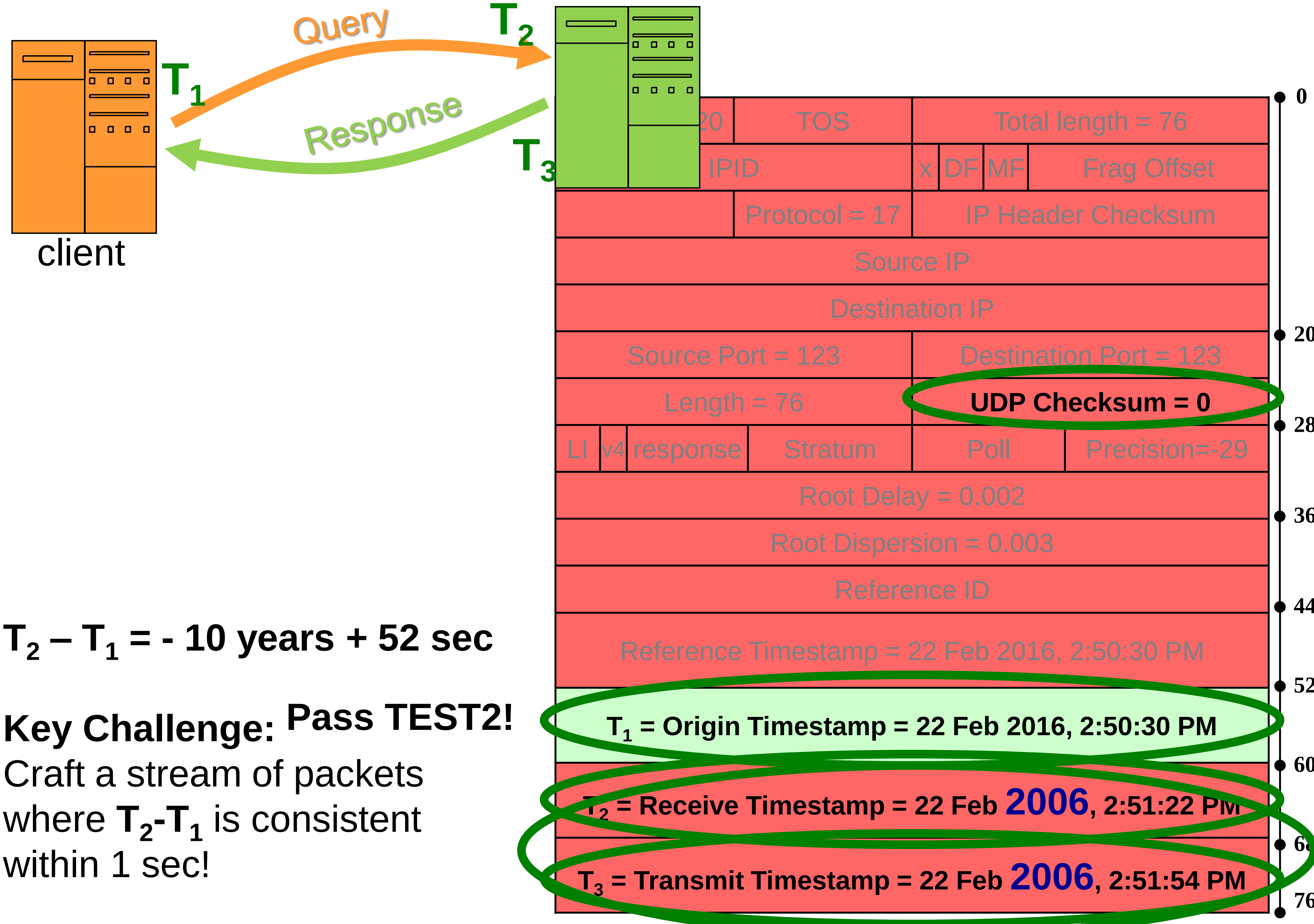
Frag2

server

# How Our Attacker Uses IPv4 Packet Fragmentation?



# Reassembled Packet



20 | TOS | Total length = 76

IPID | x | DF | MF | Frag Offset

Protocol = 17 | IP Header Checksum

Source IP

Destination IP

Source Port = 123 | Destination Port = 123

Length = 76 | **UDP Checksum = 0**

LI | v4 | response | Stratum | Poll | Precision=-29

Root Delay = 0.002

Root Dispersion = 0.003

Reference ID

Reference Timestamp = 22 Feb 2016, 2:50:30 PM

**$T_1$ = Origin Timestamp = 22 Feb 2016, 2:50:30 PM**

**$T_2$ = Receive Timestamp = 22 Feb 2006, 2:51:22 PM**

**$T_3$ = Transmit Timestamp = 22 Feb 2006, 2:51:54 PM**

**$T_2 – T_1$ = - 10 years + 52 sec**

**Key Challenge: Pass TEST2!**

Craft a stream of packets where **$T_2$-$T_1$** is consistent within 1 sec!

# Conditions for the Attack

- Server must fragment NTP packets to 68 bytes
  - Scanned 13M servers
  - About 24K servers were willing to fragment to 68-byte

- Client reassembles overlapping fragments according to First policy
  - The client prefers fragments that arrive earliest
    (We can not safely measure because of teardrop  [CA-1997-28])

- Server uses incrementing IPID
  - attacker can infer IPID using techniques explained in
    [Gilad, Herzberg’2013] and [Knockell, Crandall’2014]

# Summary, Recommendations & Impact

- Attack: DoS by spoofed KoD:
  - Rec: Implement TEST2 (patched in v4.2.8p4 & NTPSec & Cisco & RedHat Linux etc.)

- Attack: DoS by priming the pump:
  - Rec: Authentication in both directions (IETF Network Time Security draft updated)
    - client → server & server →client
  - Rate limit like Response Rate Limiting (RRL) in DNS (under discussion)

- Attack: Time shifting by IPv4 Packet Fragmentation:
  - Rec: Server should not fragment to 68 bytes (Test your server on our site)
  - Clients should drop overlapping fragments

- Other recommendations:
  - Stop my laptop from answering timing queries
  - More work on cryptography for NTP

**Thank You!**

**Questions ?**