Protocol Security and DoS Attacks

CS155 Computer and Network Security

Stanford University

Ethernet

Provides connectivity between hosts on a *Local Area Network*

Frames are addressed to a device's physical (MAC) address

Switches forward frames based on *learning* where different MACs are located. *No guarantees not sent to other hosts!*

No security (confidentiality, authentication, or integrity)

ARP (Address Resolution Protocol)

ARP allows hosts to find each others' MAC addresses on the local network

Client: To Broadcast (all MACs): *Which MAC address has IP 192.168.1.1?*

No built-in security. Attacker can impersonate a host by faking its identity
and responding to ARP requests or sending gratuitous ARP announcement

IP (Internet Protocol)

Provides routing between hosts on the Internet. Unreliable. Best Effort.

Routers simply route IP packets based on their destination address.

No inherent security. Packets have a checksum, but it’s non-cryptographic. Attackers can change any packet.

Source address is set by sender—can be faked by an attacker

BGP (Border Gateway Protocol)

Internet Service Providers announce their presence on the Internet via BGP

No authentication—possible to announce someone else’s network

- Commonly occurs (often due to operator error)

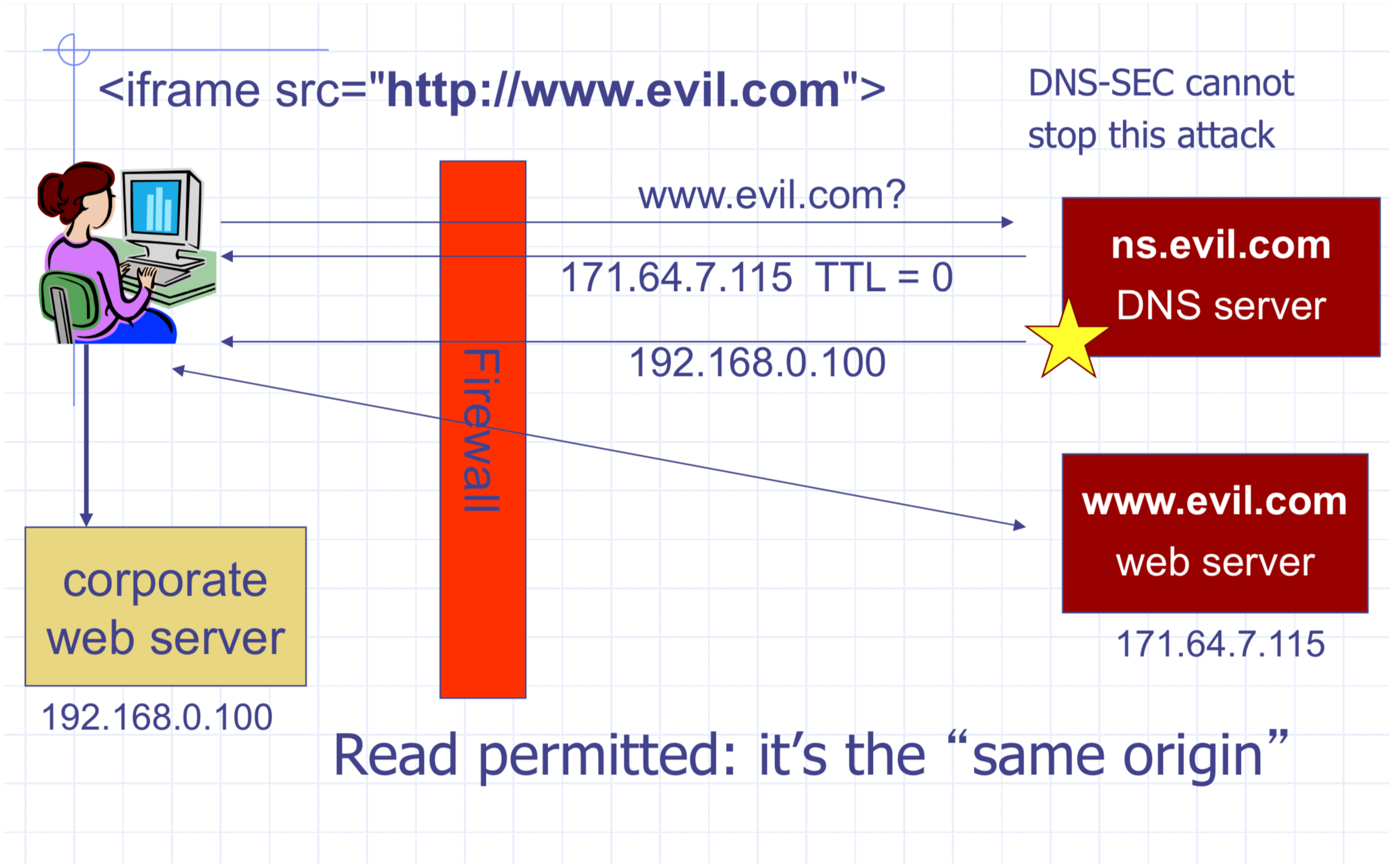
TCP (Transmission Control Protocol)

TCP provides reliable stream of data on top of unreliable IP and Ethernet

Data is split into segments and sender/receiver acknowledge received byte of data, sender retransmit dropped packets

Every TCP connection starts with a three-way handshake

DNS Rebinding



Rebinding Defenses

Browser Mitigations:

- Refuse to switch IPs mid session
- Interacts poorly with proxies, VPNs, CDNs, etc
- Not consistently implemented in any browser

Server Defenses

- Check Host header for unrecognized domains
- Authenticate users with something else beyond IP address

DNSSEC

Adds authentication and integrity to DNS responses

Authoritative DNS servers sign DNS responses using cryptographic key

Clients can verify that a response is legitimate by checking signature through PKI similar to HTTPS

Most people don’t use DNSSEC and never will. Use TLS instead.

Network Security Takeaway

Assume the network is out to get you.

If you want any guarantee of any security, use TLS.

Denial of Service Attacks

Goal: take large site offline by overwhelming it with network traffic such that they can’t process real requests

How: find mechanism where attacker doesn’t have to spend a lot of effort, but requests are difficult/expensive for victim to process

Types of Attacks

DoS Bug: design flaw that allows one machine to disrupt a service. Generally a protocol asymmetry, e.g., easy to send request, difficult to create response. Or requires server state.

DoS Flood: control a large number of requests from a botnet of machines you control

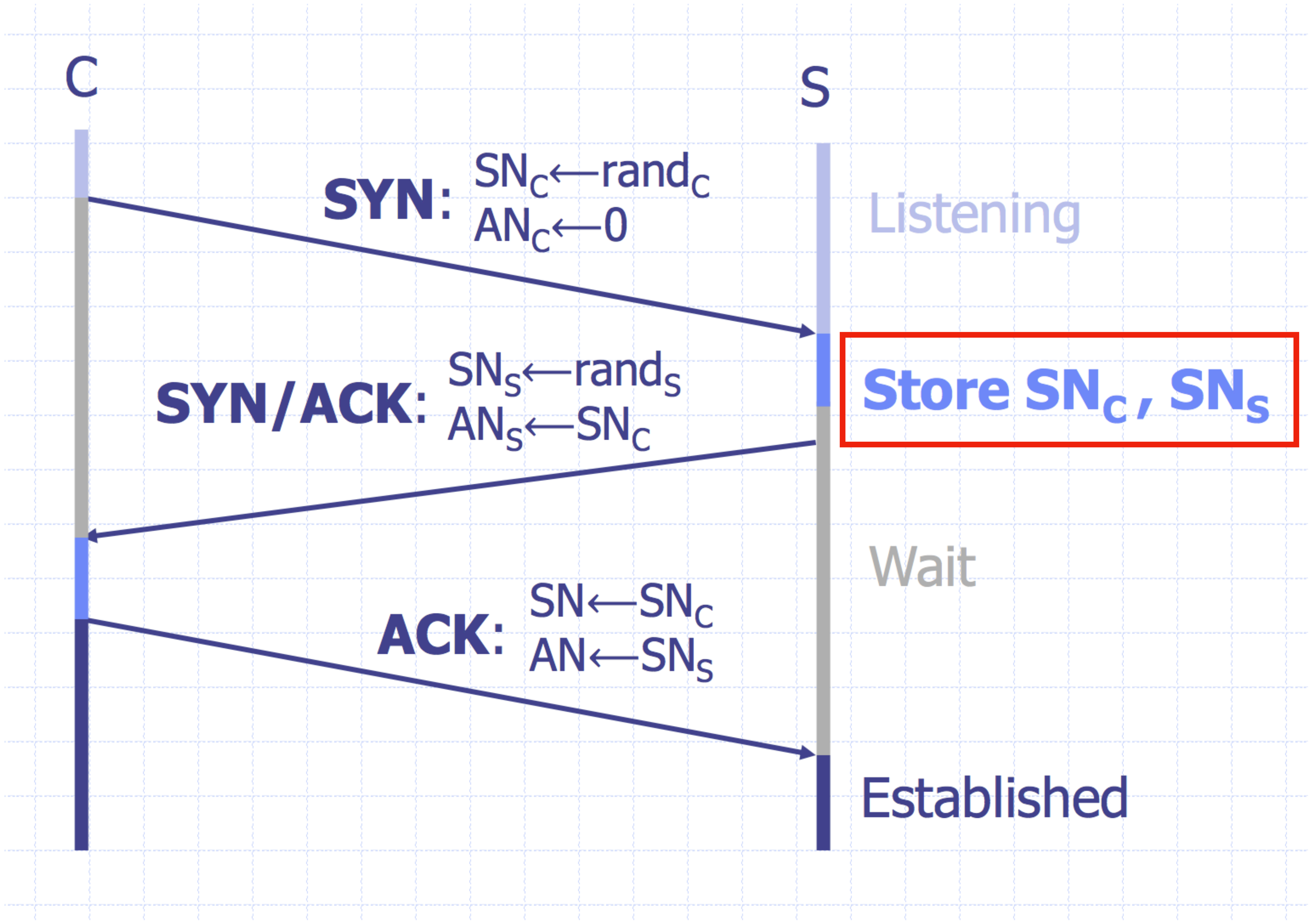
Possible at Every Layer

Link Layer: send too much traffic for switches/routers to handle

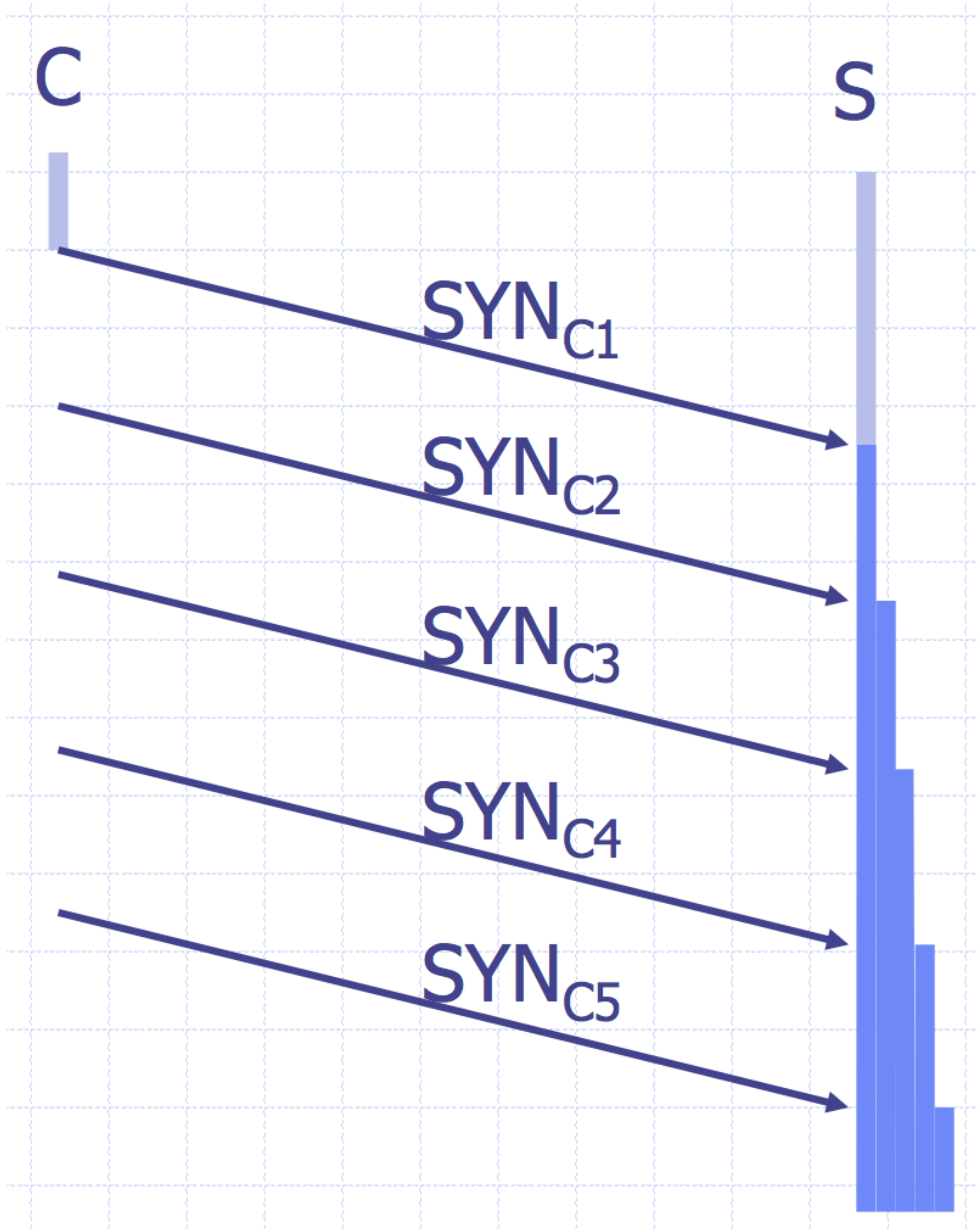
TCP/UDP: require servers to maintain large number of concurrent connections or state

Application Layer: require servers to perform expensive queries or cryptographic operations

TCP Handshake


C
S
SYN: $SN_C \leftarrow rand_C$ $AN_C \leftarrow 0$
Listening
Store SN_C, SN_S
SYN/ACK: $SN_S \leftarrow rand_S$ $AN_S \leftarrow SN_C$
Wait
ACK: $SN \leftarrow SN_C$ $AN \leftarrow SN_S$
Established

SYN Floods



Single machine:

- SYN Packets with **random source IP addresses**
- Fills up backlog queue on server
- No further connections possible

Core Problem

Problem: server commits resources (memory) before confirming identify of client (when client responds)

Bad Solution:

- Increase backlog queue size
- Decrease timeout

Real Solution: Avoid state until 3-way handshake completes

SYN Cookies

Idea: Instead of storing SNc and SNs…
send a cookie back to the client.

L = MAC_{key} (SAddr, SPort, DAddr, DPort, SN_C, T)

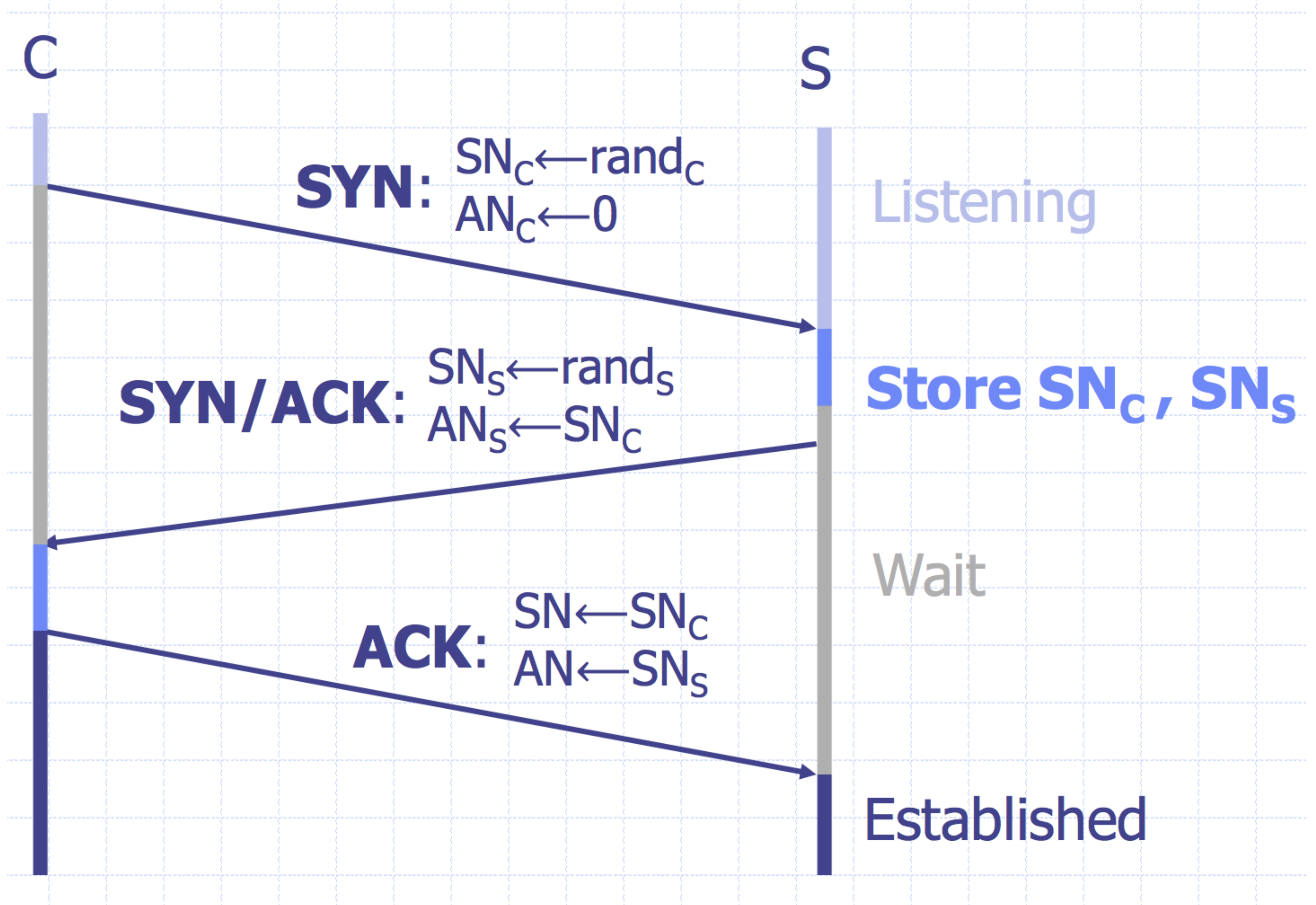
key: picked at random during boot

T = 5-bit counter incremented every 64 secs.

SN_s = (T || mss || L)

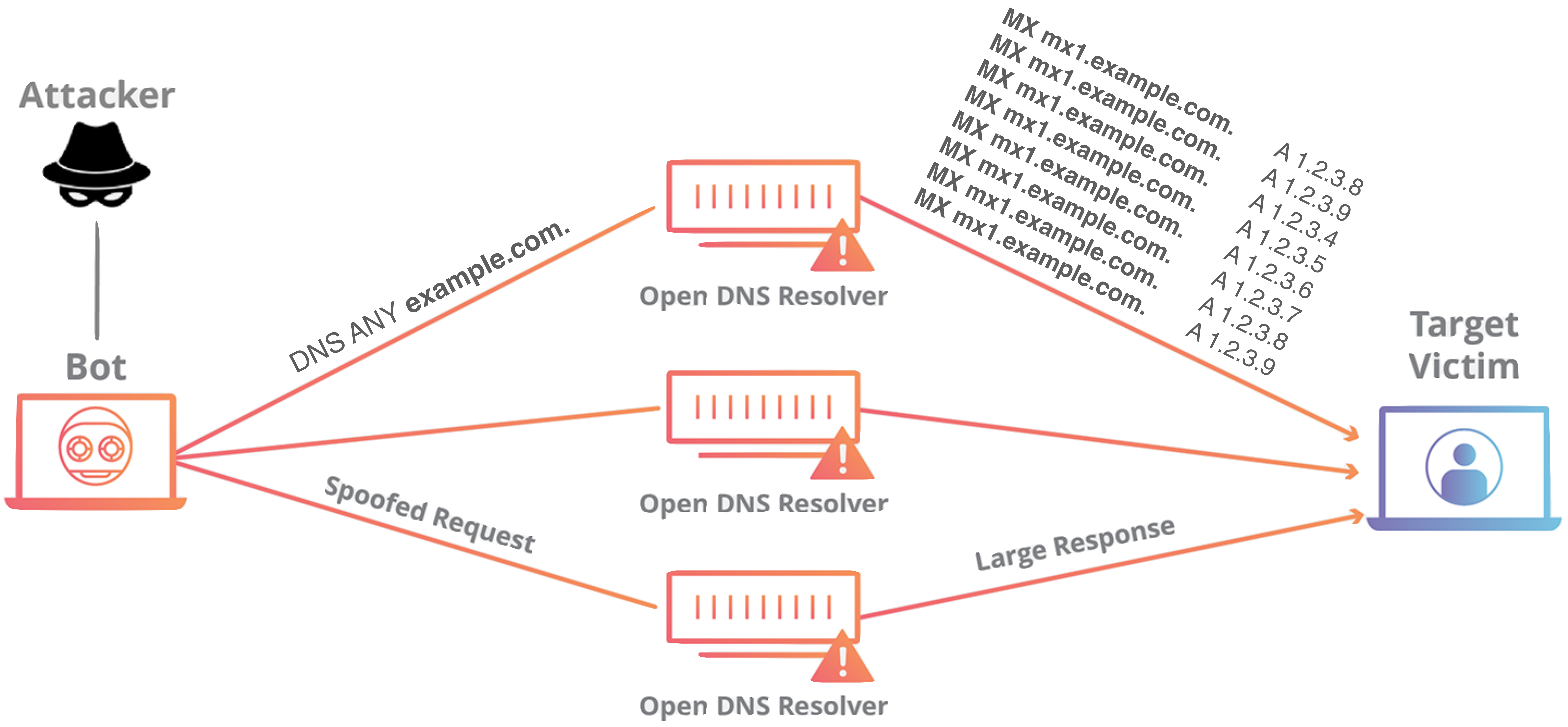
Honest client sends ACK (AN=SN_s , SN=SN_C+1)

Server allocates space for socket only if valid SNs


C
S
SYN: $SN_C \leftarrow rand_C$ $AN_C \leftarrow 0$
Listening
SYN/ACK: $SN_S \leftarrow rand_S$ $AN_S \leftarrow SN_C$
Store SN_C, SN_S
Wait
ACK: $SN \leftarrow SN_C$ $AN \leftarrow SN_S$
Established

Server does not save state
(loses TCP options)

Amplification Attacks



60-70x Increase in Size

Image: Cloudflare

Common UDP Amplifiers

DNS: ANY query returns *all* records server has about a domain

NTP: MONLIST returns list of last 600 clients who asked for the time recently

Only works if you can receive a big response by sending a single packet — otherwise spoofing doesn't help you.

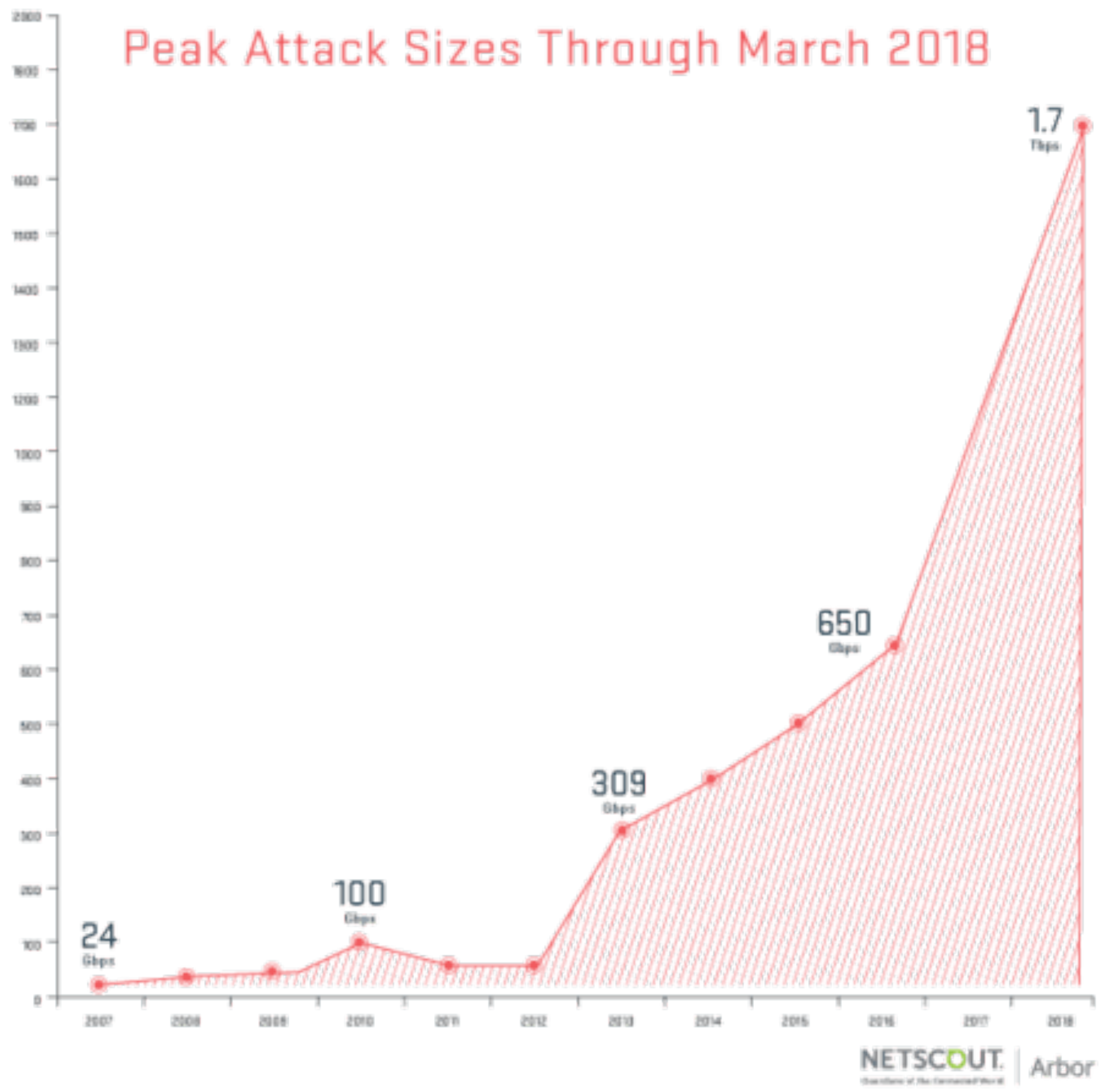
Amplification Attacks

2013: DDoS attack generated 300 Gbps (DNS)

- 31,000 misconfigured open resolvers, each at 10 Mbps
- Source: 3 networks that allowed IP spoofing

2014: 400 Gbps DDoS attacked used 4500 NTP servers

Memcache



Memcache: retrieve large record

The server responds by firing back as much as 50,000 times the data it received.

THE WALL STREET JOURNAL.

October 21, 2016

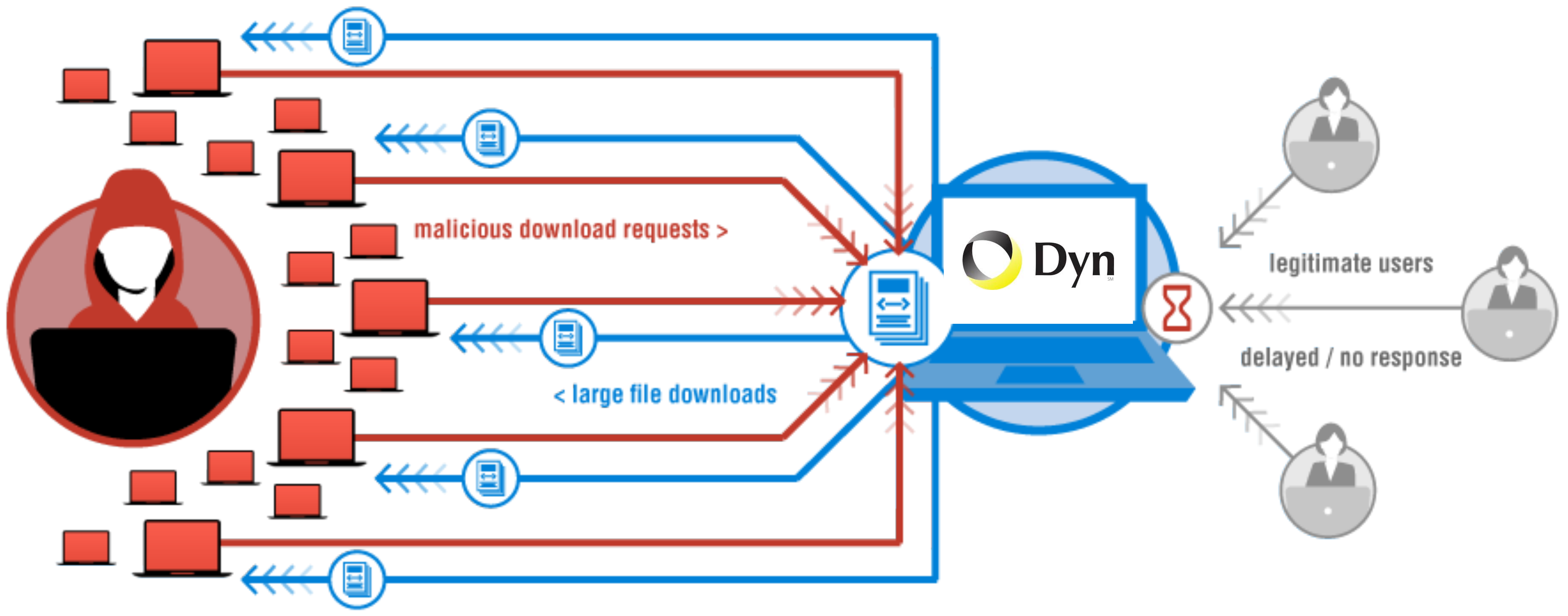
Cyberattack Knocks Out Access to Websites

Popular sites such as Twitter, Netflix and PayPal were unreachable for part of the day

United States
of America

twitter amazon web services PayPal NETFLIX SOUNDCLOUD

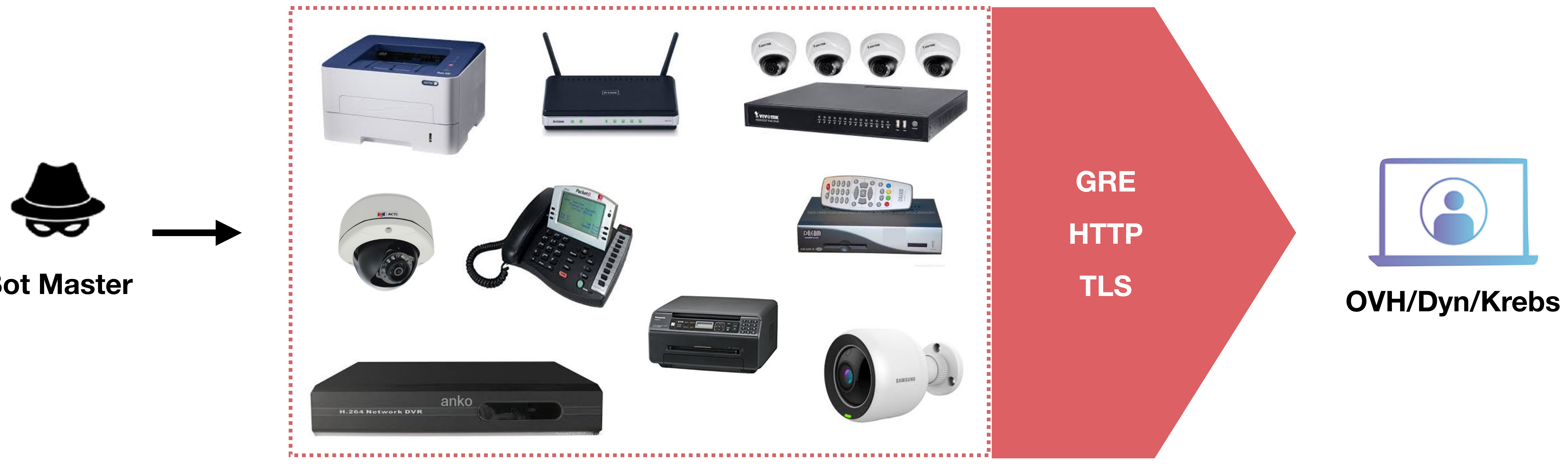
Spotify GitHub reddit New York Times



> “We are still working on analyzing the data but the estimate at the time of this report is up to 100,000 malicious endpoints. […] There have been some reports of a magnitude in the 1.2 Tbps range; at this time we are unable to verify that claim.”

Image: Verisign

A Botnet of IoT Devices



≈ ~~200K Hosts~~
200K IoT devices

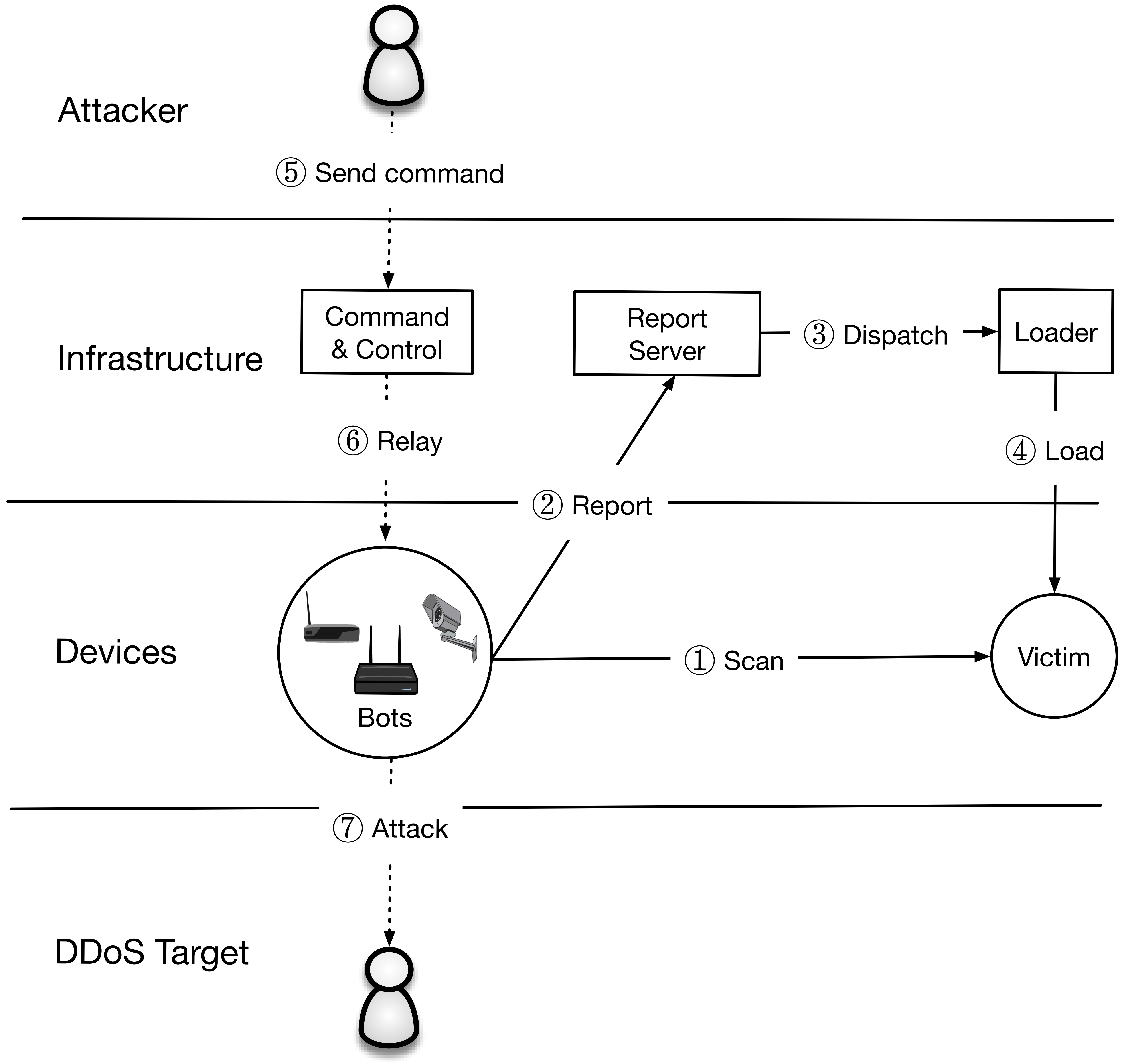
Not Amplification.
Flood with SYN, ACK, UDP, and GRE packets

The Mirai Malware

5-7. Later, the **bot master** will issue commands to pause scanning and to start an attack

Attack Command:

- Action (e.g., START, STOP)
- Target IP(s)
- Attack Type (e.g., GRE, DNS, TCP)
- Attack Duration



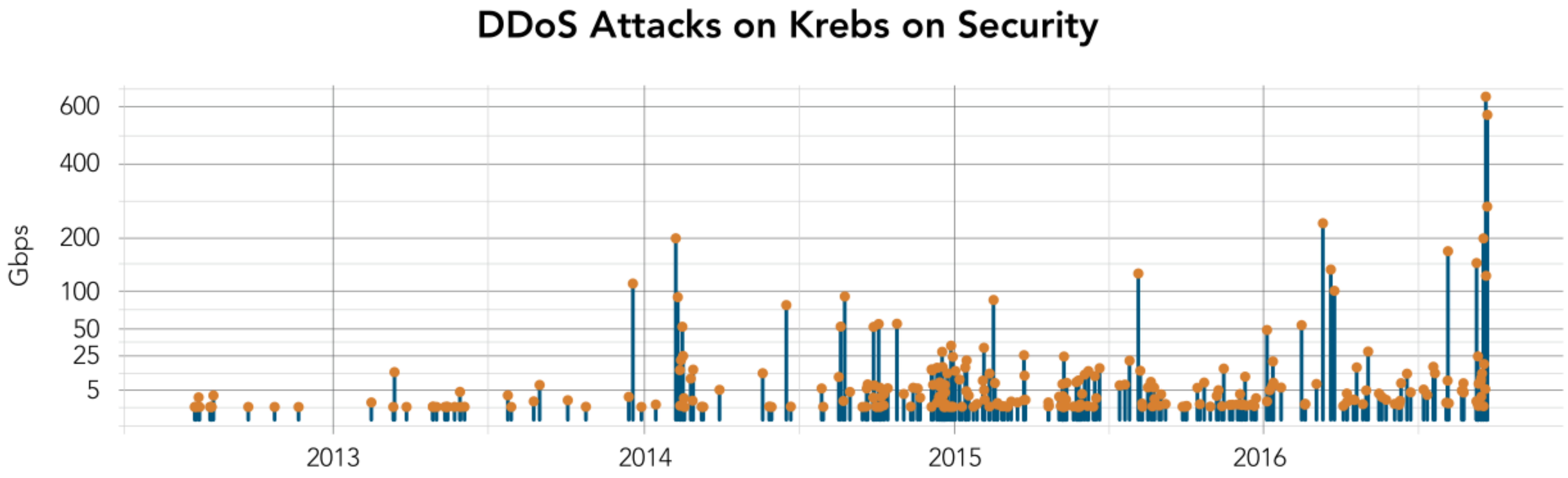
Password Guessing

Password	Device Type	Password	Device Type	Password	Device Type
123456	ACTi IP Camera	klv1234	HiSilicon IP Camera	1111	Xerox Printer
anko	ANKO Products DVR	jvbzd	HiSilicon IP Camera	Zte521	ZTE Router
pass	Axis IP Camera	admin	IPX-DDK Network Camera	1234	Unknown
888888	Dahua DVR	system	IQinVision Cameras	12345	Unknown
666666	Dahua DVR	meinsm	Mobotix Network Camera	admin1234	Unknown
vizxv	Dahua IP Camera	54321	Packet8 VOIP Phone	default	Unknown
7ujMko0vizxv	Dahua IP Camera	00000000	Panasonic Printer	fucker	Unknown
7ujMko0admin	Dahua IP Camera	realtek	RealTek Routers	guest	Unknown
666666	Dahua IP Camera	1111111	Samsung IP Camera	password	Unknown
dreambox	Dreambox TV Receiver	xmhdipc	Shenzhen Anran Camera	root	Unknown
juantech	Guangzhou Juan Optical	smcadmin	SMC Routers	service	Unknown
xc3511	H.264 Chinese DVR	ikwb	Toshiba Network Camera	support	Unknown
OxhlwSG8	HiSilicon IP Camera	ubnt	Ubiquiti AirOS Router	tech	Unknown
cat1029	HiSilicon IP Camera	supervisor	VideoIQ	user	Unknown
hi3518	HiSilicon IP Camera	<none>	Vivotek IP Camera	zlxx.	Unknown
klv123	HiSilicon IP Camera				

Mirai Population

700,000
600,000
500,000
400,000
300,000
200,000
100,000
0
# network telescope scans
08/01/16
09/01/16
10/01/16
11/01/16
12/01/16
01/01/17
02/01/17
Date
Total Mirai Scans
TCP/23231
TCP/22
TCP/2222
TCP/37777
TCP/443
TCP/5555
TCP/6789
TCP/8080
TCP/80
TCP/23
TCP/2323
TCP/7547

~600K devices compromised



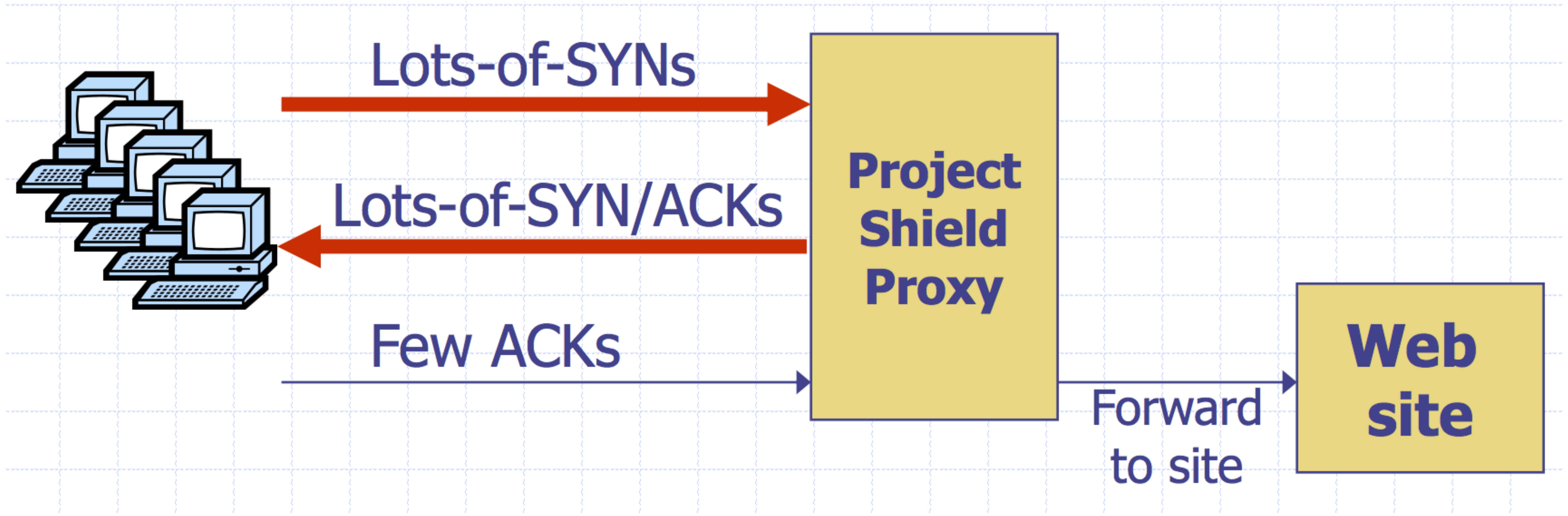
> “The magnitude of the attacks seen during the final week were significantly larger than the majority of attacks Akamai sees on a regular basis. […] In fact, while the attack on September 20 was the largest attack ever mitigated by Akamai, the attack on September 22 would have qualified for the record at any other time, peaking at 555 Gbps.”

Source: 2017 Akamai State of the Internet

Google Project Shield

DDoS Attacks are often used to censor content. In the case of Mirai, Brian Kreb's blog was under attack.

Google Project shield uses Google bandwidth to shield vulnerable websites (e.g., news, blogs, human rights orgs)



Moving Up Stack: GET Floods

Command bot army to:
  * Complete real TCP connection
  * Complete TLS Handshake
  * GET large image or other content

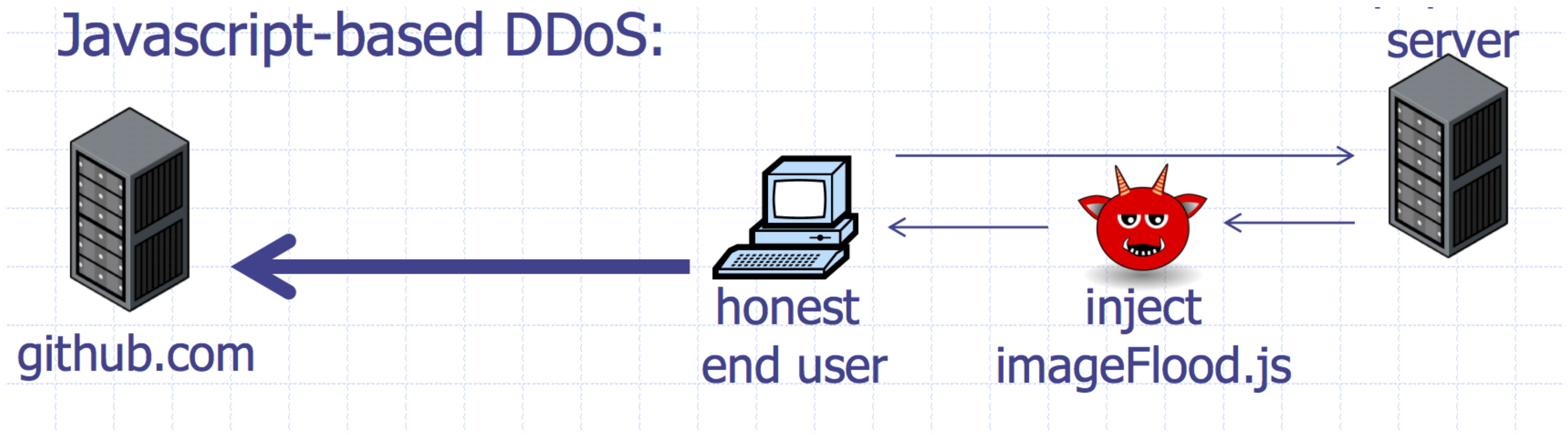
Will bypass flood protections…. but attacker can no longer use random source IPs

Victim site can block or rate limit bots

Github Attacks

1.35 Tbps attack against Github caused by javascript injected into HTTP web requests

The Chinese government was widely suspected to be behind the attack



Client Puzzles

Idea: What if we force every client to do moderate amount of work for every connection they make?

Example:

1) Server Sends: C

2) Client: find X s.t. $LSB_n(SHA\text{-}1(C||X)) = 0^n$

Assumption:

Puzzle takes 2^n for the client to compute (0.3 s on 1Ghz core)

Solution is trivial for server to check (single SHA-1)

Client Puzzles

Not frequently used in the real world

Benefits:

* Can change *n* based on amount of attack traffic

Limitations:

* Requires changes to both protocols, clients, and servers
* Hurts low power legitimate clients during attack (e.g., phones)