02

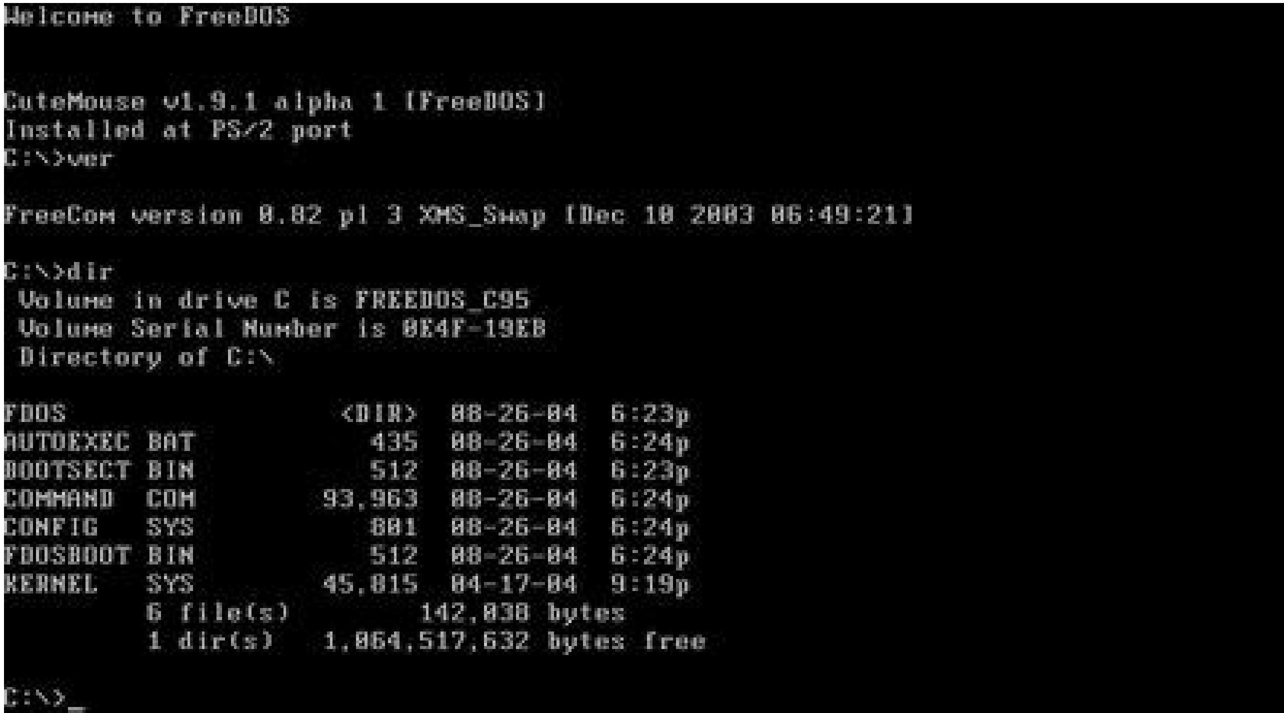
DDoS Attack & Defense

Kai Bu
kaibu@zju.edu.cn
http://list.zju.edu.cn/kaibu/netsec2022

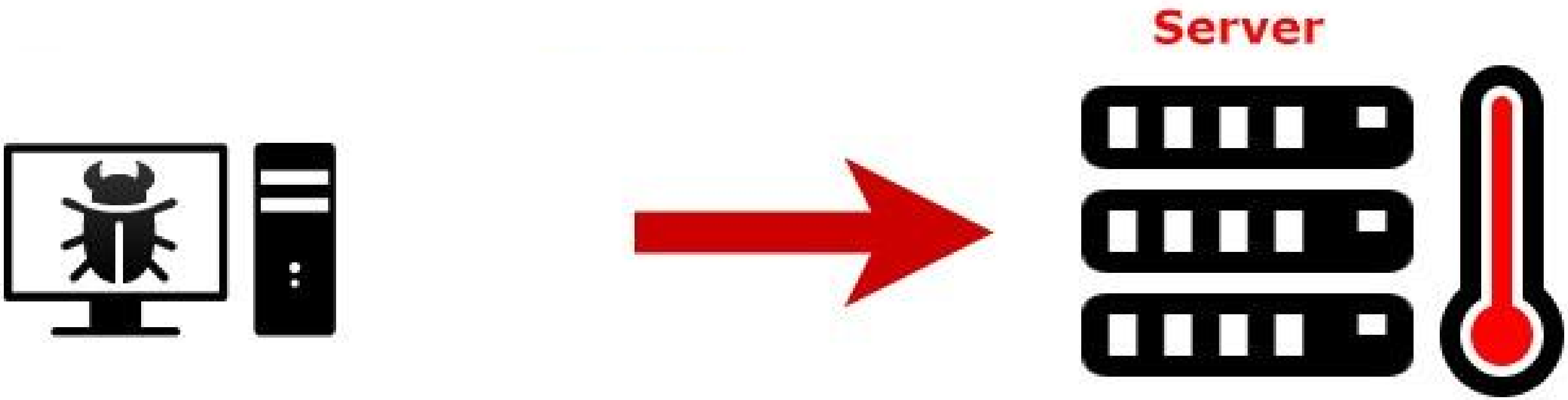
DDoS?

DoS?

DoS?



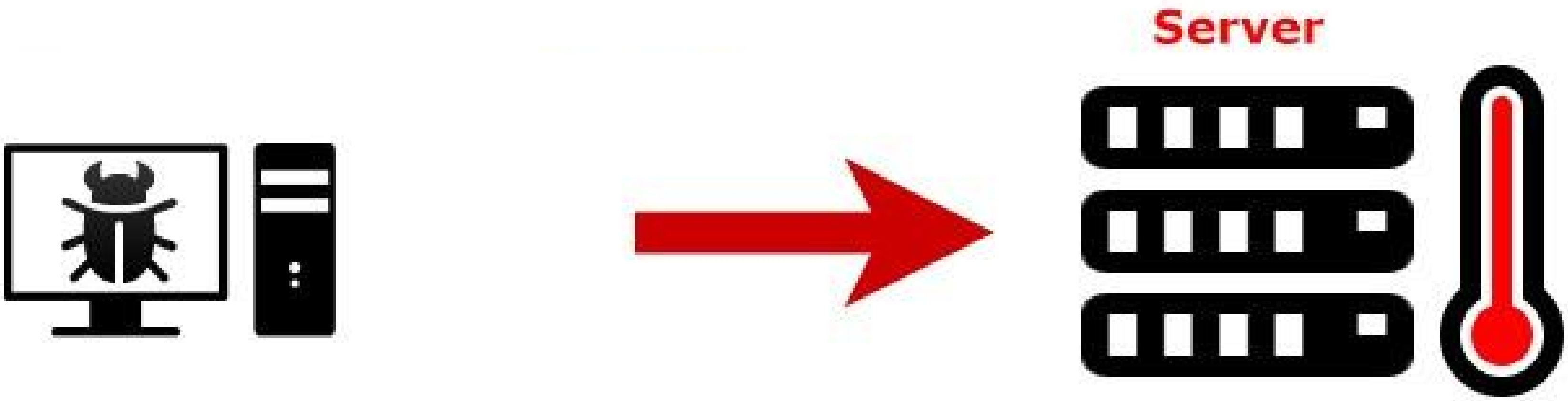
```
Welcome to FreeDOS


CuteMouse v1.9.1 alpha 1 [FreeDOS]
Installed at PS/2 port
C:\>ver

FreeCom version 0.82 pl 3 XMS_Swap [Dec 10 2003 06:49:21]

C:\>dir
 Volume in drive C is FREEDOS_C95
 Volume Serial Number is 0E4F-19EB
 Directory of C:\

FDOS                 <DIR>  08-26-04  6:23p
AUTOEXEC BAT           435  08-26-04  6:24p
BOOTSECT BIN           512  08-26-04  6:23p
COMMAND  COM        93,963  08-26-04  6:24p
CONFIG   SYS           801  08-26-04  6:24p
FDOSBOOT BIN           512  08-26-04  6:24p
KERNEL   SYS        45,815  04-17-04  9:19p
         6 file(s)        142,038 bytes
         1 dir(s)   1,864,517,632 bytes free

C:\>_
```



DoS!

Denial-of-Service Attack:

control an attacking computer/device;
flood victim with superfluous requests;
overload victim and prevent it from fulfilling some legitimate requests;

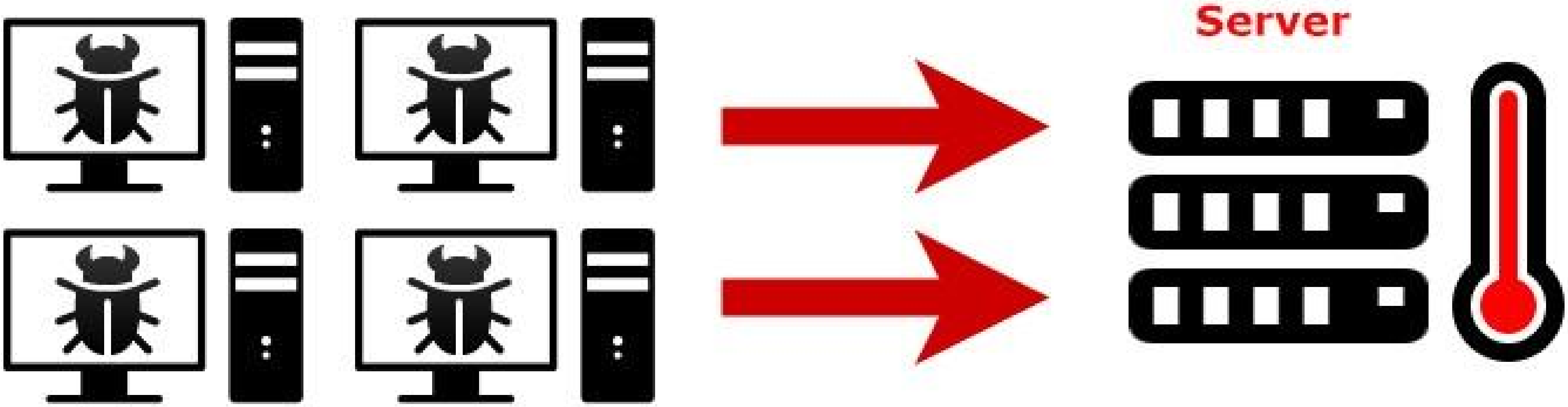


DoS!?

Denial-of-Service Attack:

control an attacking computer/device;
flood victim with superfluous requests;
overload victim and prevent it from
fulfilling some legitimate requests;

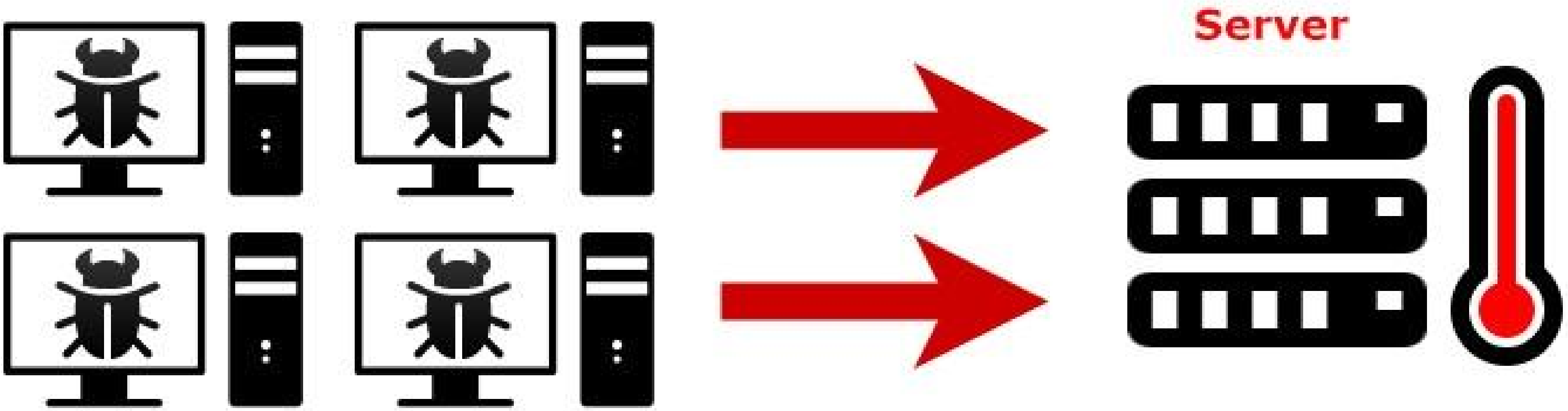
defense: block the attacking source



DDoS!!

Distributed Denial-of-Service Attack:

control many different attacking sources;
make it harder to stop the attack simply by blocking a single source;



DDoS!!

Distributed Denial-of-Service Attack:

how to attack?

how to defend?

DDoS!!

Distributed Denial-of-Service Attack:

how to attack a network service?

Ping Flood

- Exploit Internet Control Messge Protocol (ICMP)

  an internet layer protocol used by network devices to communicate;

  also used by network diagnostic tools such as traceroute and ping;
- ICMP Echo Request: sender to receiver
- ICMP Echo Reply: receiver to sender

Ping Flood

- **Attack principle**

  both incoming ICMP Echo Request and outgoing ICMP Echo Reply consume bandwidth;

  overwhelm the target device’s ability to respond to a high number of requests

  and/or overload the network connection with bogus traffic

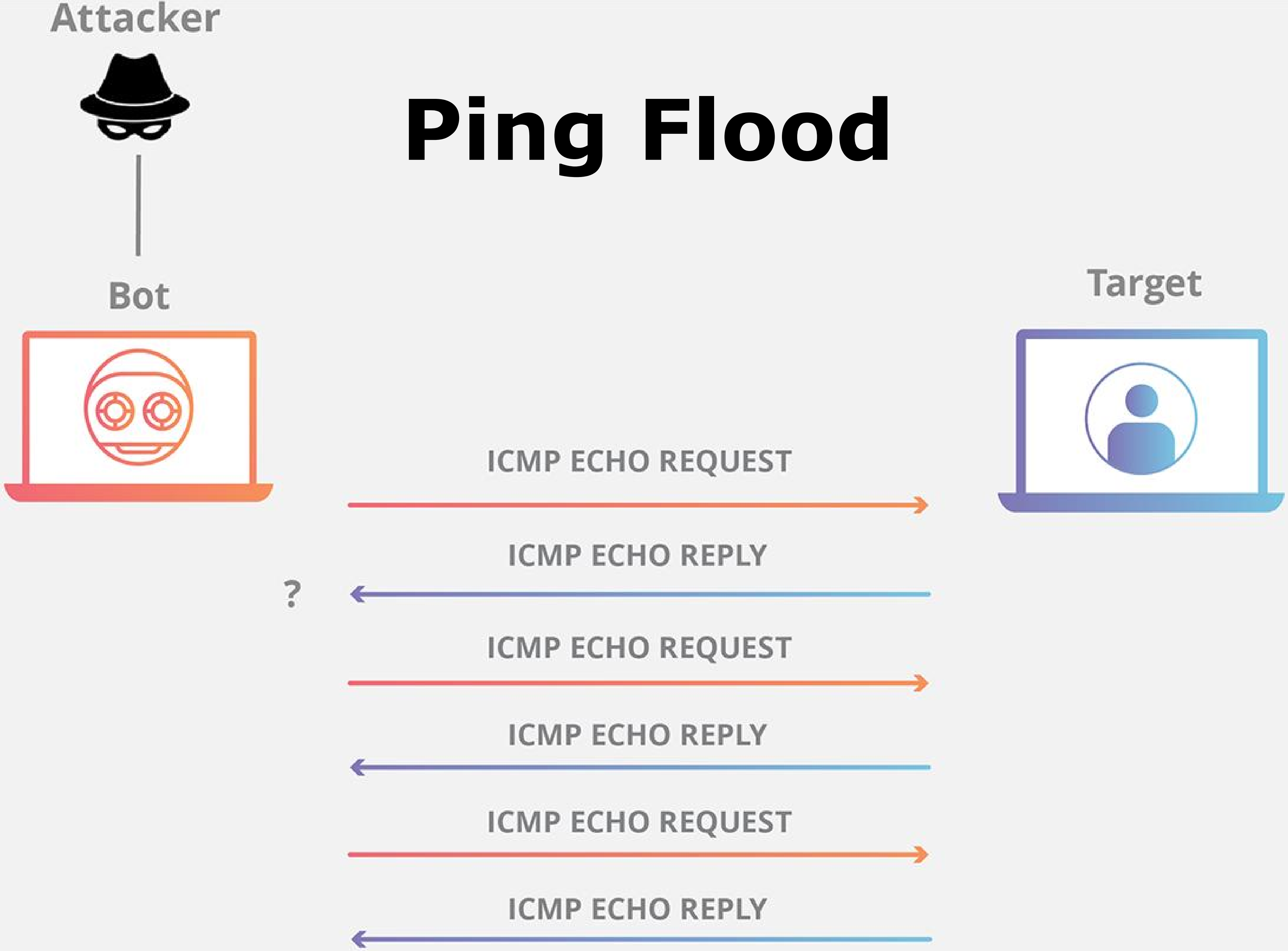
Ping Flood

- **Attack principle**

  The attacker sends many ICMP echo request packets to the targeted server using multiple devices;

  The targeted server then sends an ICMP echo reply packet to each requesting device’s IP address as a response.

Ping Flood



https://www.cloudflare.com/learning/ddos/ping-icmp-flood-ddos-attack/

Ping Flood

- **Attack principle**

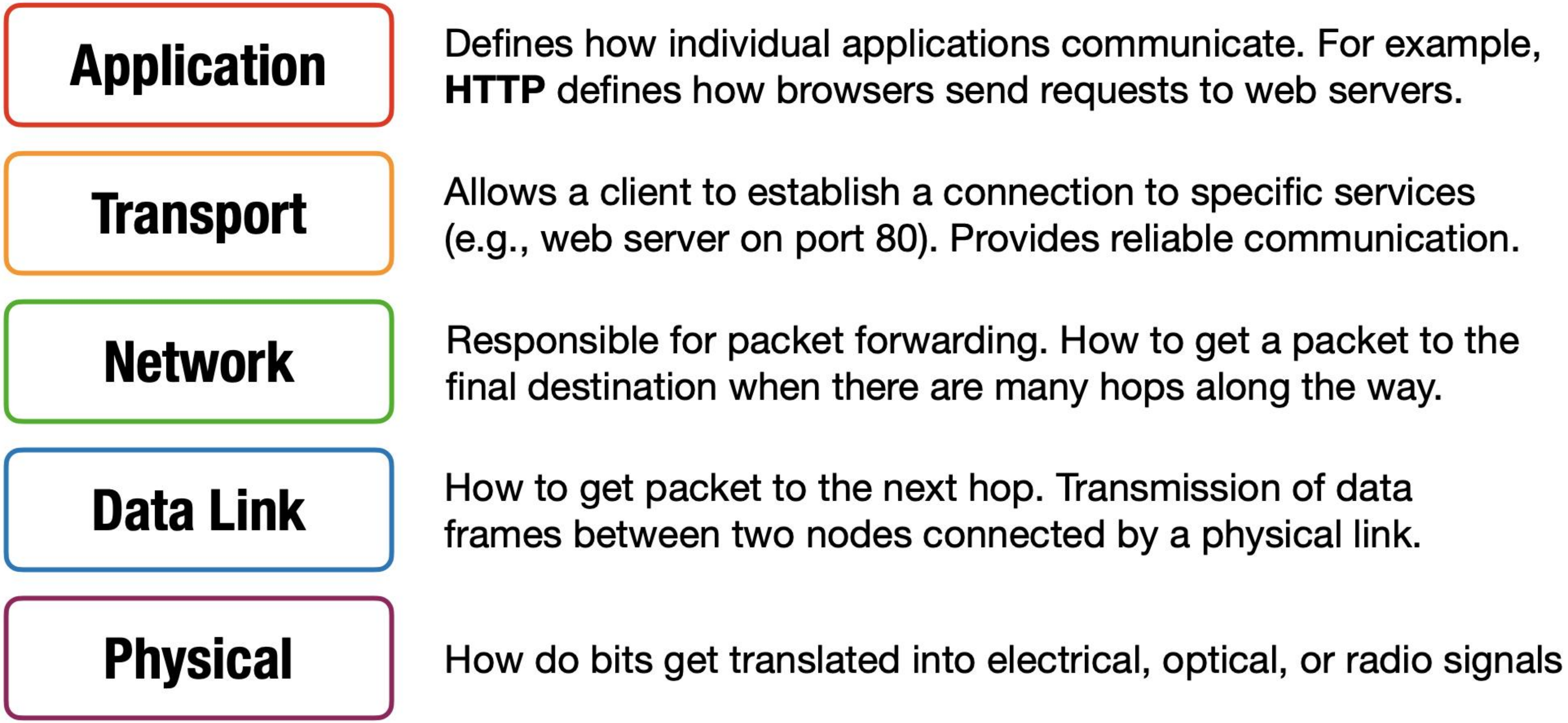
  saturate the target device’s capacity by sending many requests;

- **Solution**

  disable the ICMP functionality of the target device;

  (make the device unresponsive to ping requests and traceroute requests)

OSI 5 Layer Model



Application

Defines how individual applications communicate. For example, **HTTP** defines how browsers send requests to web servers.

Transport

Allows a client to establish a connection to specific services (e.g., web server on port 80). Provides reliable communication.

Network

Responsible for packet forwarding. How to get a packet to the final destination when there are many hops along the way.

Data Link

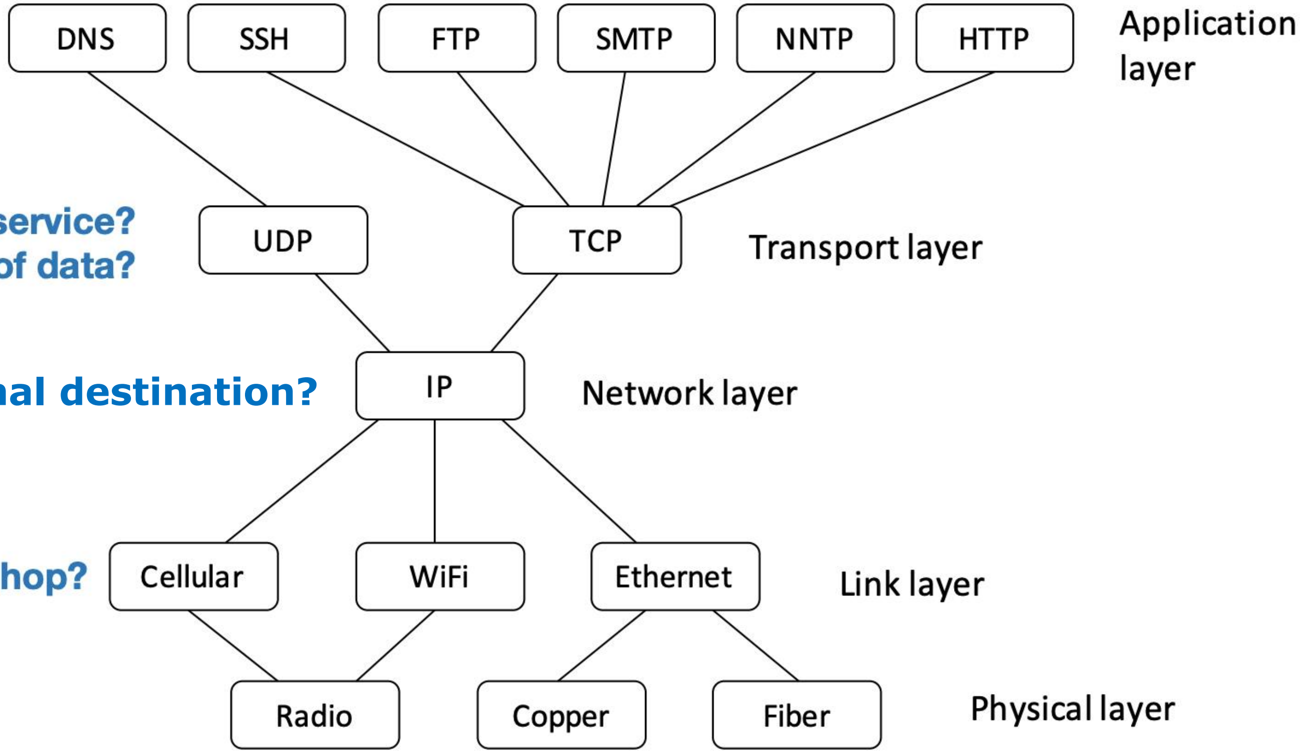
How to get packet to the next hop. Transmission of data frames between two nodes connected by a physical link.

Physical

How do bits get translated into electrical, optical, or radio signals

examples from https://cs155.stanford.edu/lectures/13-internet-protocols.pdf

OSI 5 Layer Model



How does Application structure data?

DNS SSH FTP SMTP NNTP HTTP

Application layer

How do I get to the right service?
How do I have a reliable “stream” of data?

UDP TCP

Transport layer

How do I get to final destination?

IP

Network layer

How do I get to next hop?

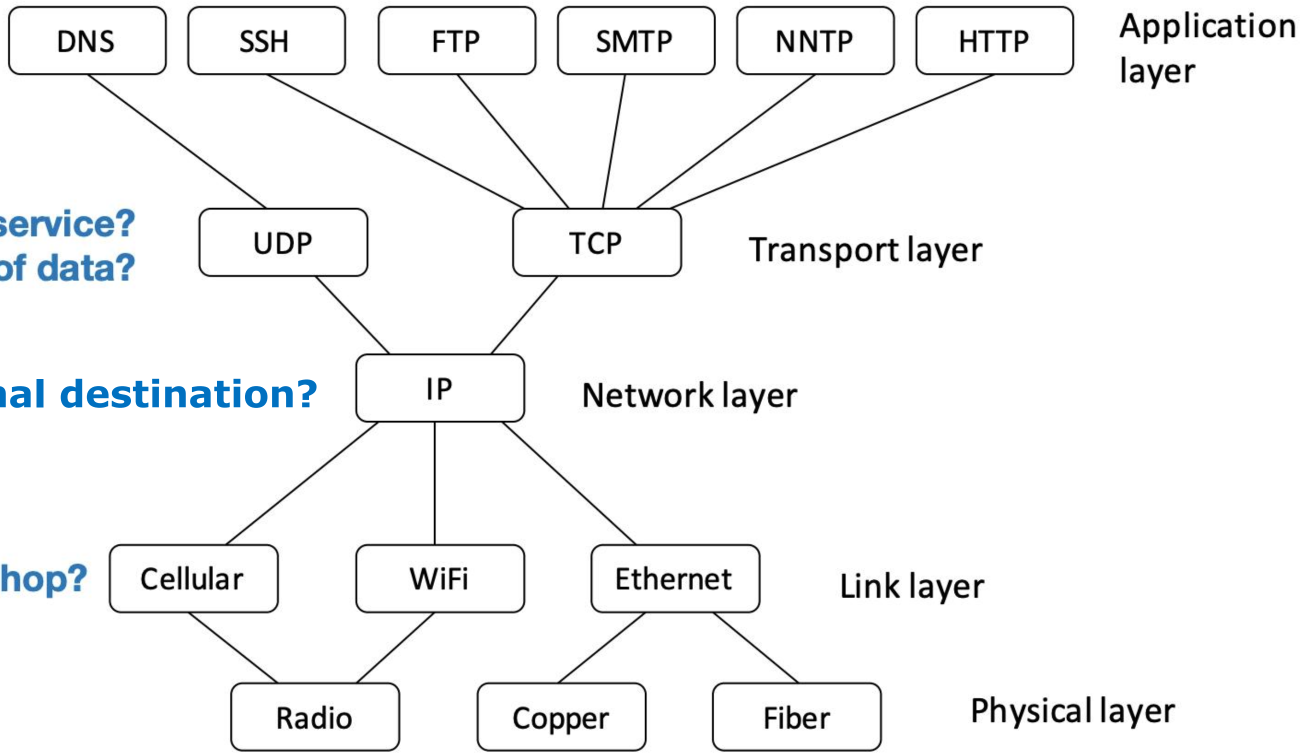
Cellular WiFi Ethernet

Link layer

Radio Copper Fiber

Physical layer

OSI 5 Layer Model



How does Application structure data?

DNS SSH FTP SMTP NNTP HTTP

Application layer

How do I get to the right service?
How do I have a reliable “stream” of data?

UDP TCP

Transport layer

How do I get to final destination?

IP

Network layer

How do I get to next hop?

Cellular WiFi Ethernet

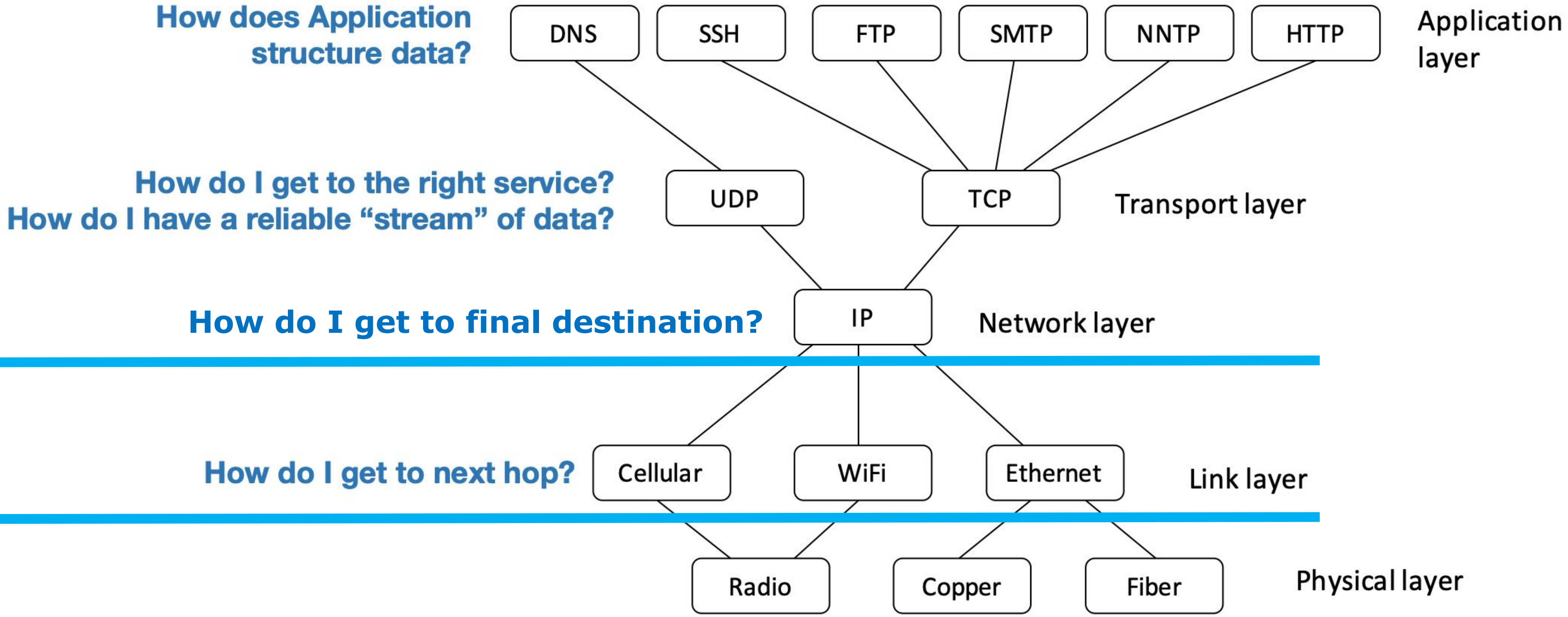
Link layer

Radio Copper Fiber

Physical layer

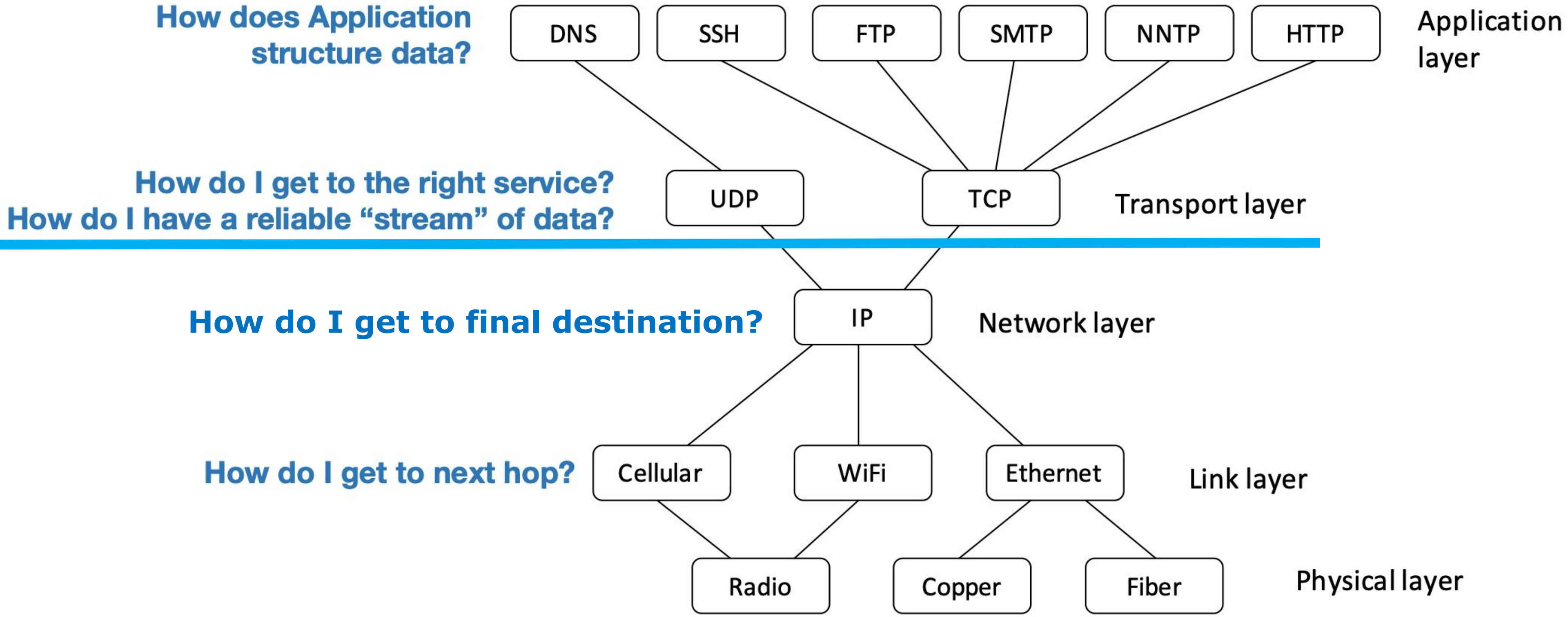
how to ddos at each layer?

OSI 5 Layer Model


How does Application structure data?
DNS
SSH
FTP
SMTP
NNTP
HTTP
Application layer
How do I get to the right service?
How do I have a reliable "stream" of data?
UDP
TCP
Transport layer
How do I get to final destination?
IP
Network layer
How do I get to next hop?
Cellular
WiFi
Ethernet
Link layer
Radio
Copper
Fiber
Physical layer

link/IP layer:
send too much traffic for switches/routers to handle

OSI 5 Layer Model



How does Application structure data?

DNS SSH FTP SMTP NNTP HTTP

Application layer

How do I get to the right service?

How do I have a reliable “stream” of data?

UDP TCP

Transport layer

How do I get to final destination?

IP

Network layer

How do I get to next hop?

Cellular WiFi Ethernet

Link layer

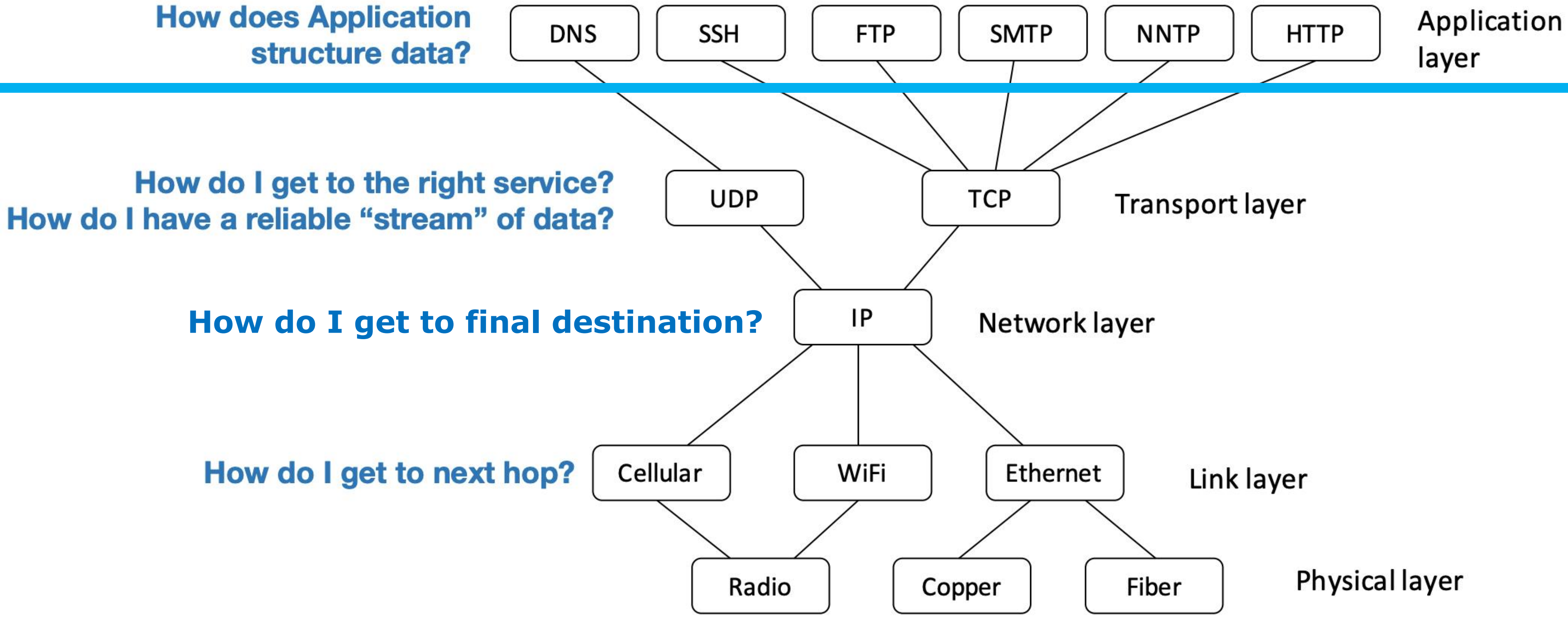
Radio Copper Fiber

Physical layer

transport layer:

require servers to maintain large number of concurrent connections or state

OSI 5 Layer Model



How does Application structure data?

DNS SSH FTP SMTP NNTP HTTP

Application layer

How do I get to the right service?
How do I have a reliable “stream” of data?

UDP TCP

Transport layer

How do I get to final destination?

IP

Network layer

How do I get to next hop?

Cellular WiFi Ethernet

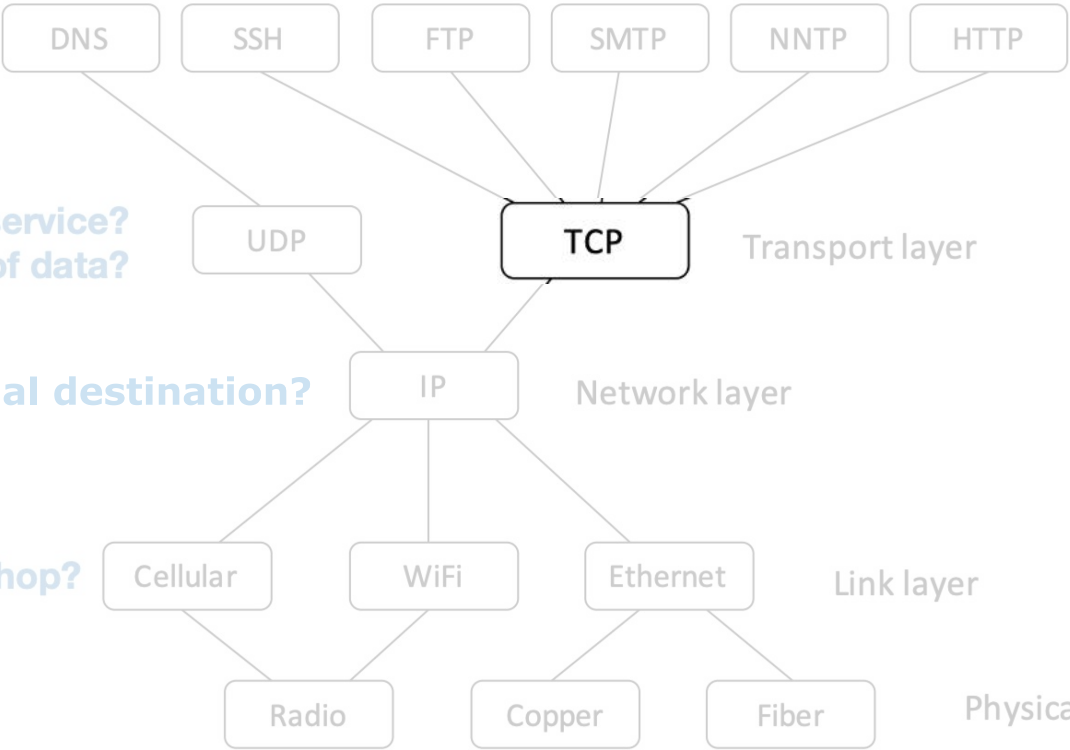
Link layer

Radio Copper Fiber

Physical layer

application layer:
require servers to perform expensive queries or cryptographic operations

OSI 5 Layer Model



How does Application structure data?

DNS SSH FTP SMTP NNTP HTTP

Application layer

How do I get to the right service?
How do I have a reliable "stream" of data?

UDP **TCP**

Transport layer

How do I get to final destination?

IP

Network layer

How do I get to next hop?

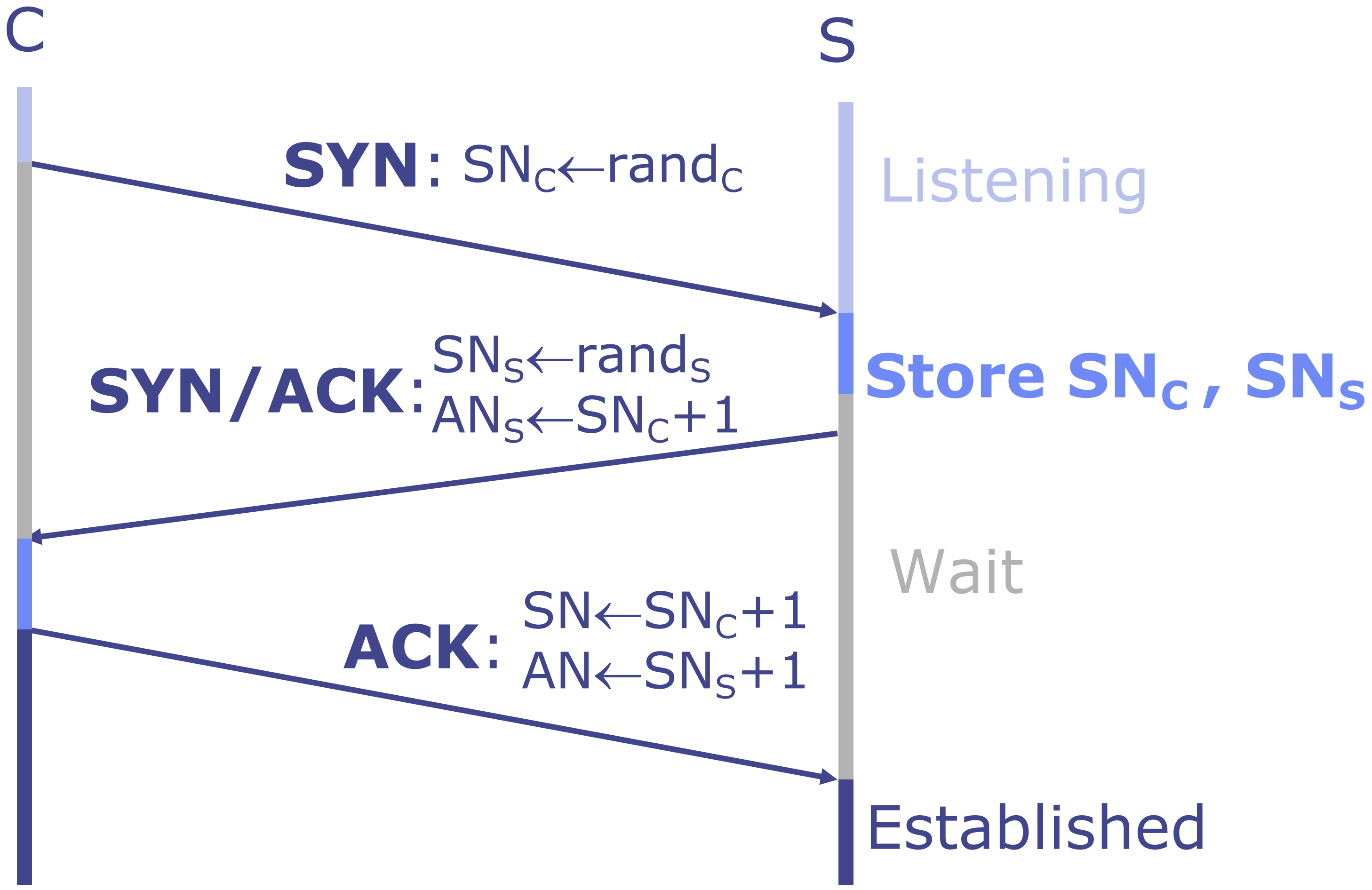
Cellular WiFi Ethernet

Link layer

Radio Copper Fiber

Physical layer

TCP Handshake



C

S

SYN: $SN_C \leftarrow rand_C$

Listening

$SN_S \leftarrow rand_S$

SYN/ACK: $AN_S \leftarrow SN_C+1$

Store SN_C , SN_S

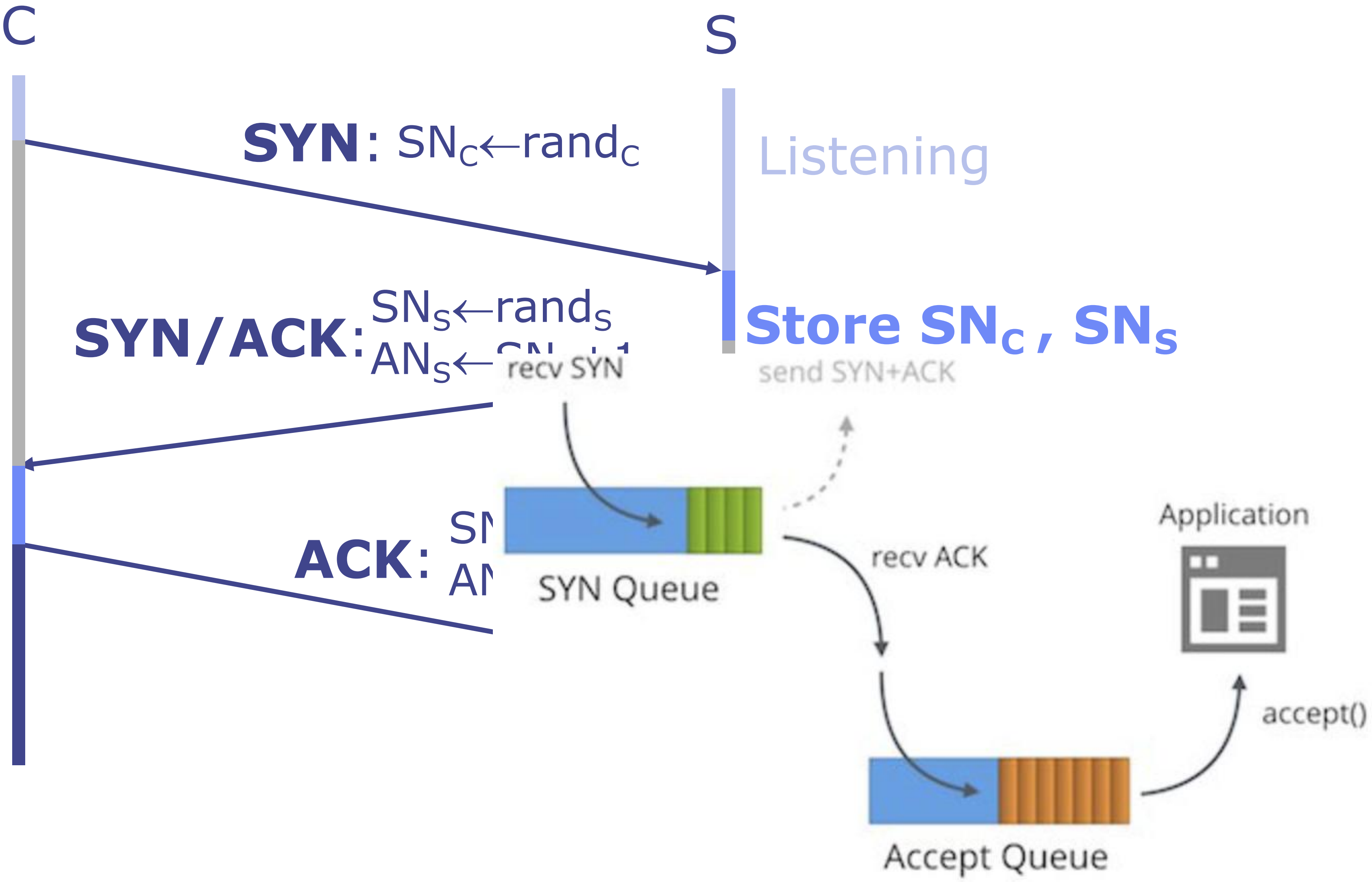
Wait

$SN \leftarrow SN_C+1$

ACK: $AN \leftarrow SN_S+1$

Established

TCP Handshake



C

S

SYN: $SN_C \leftarrow rand_C$

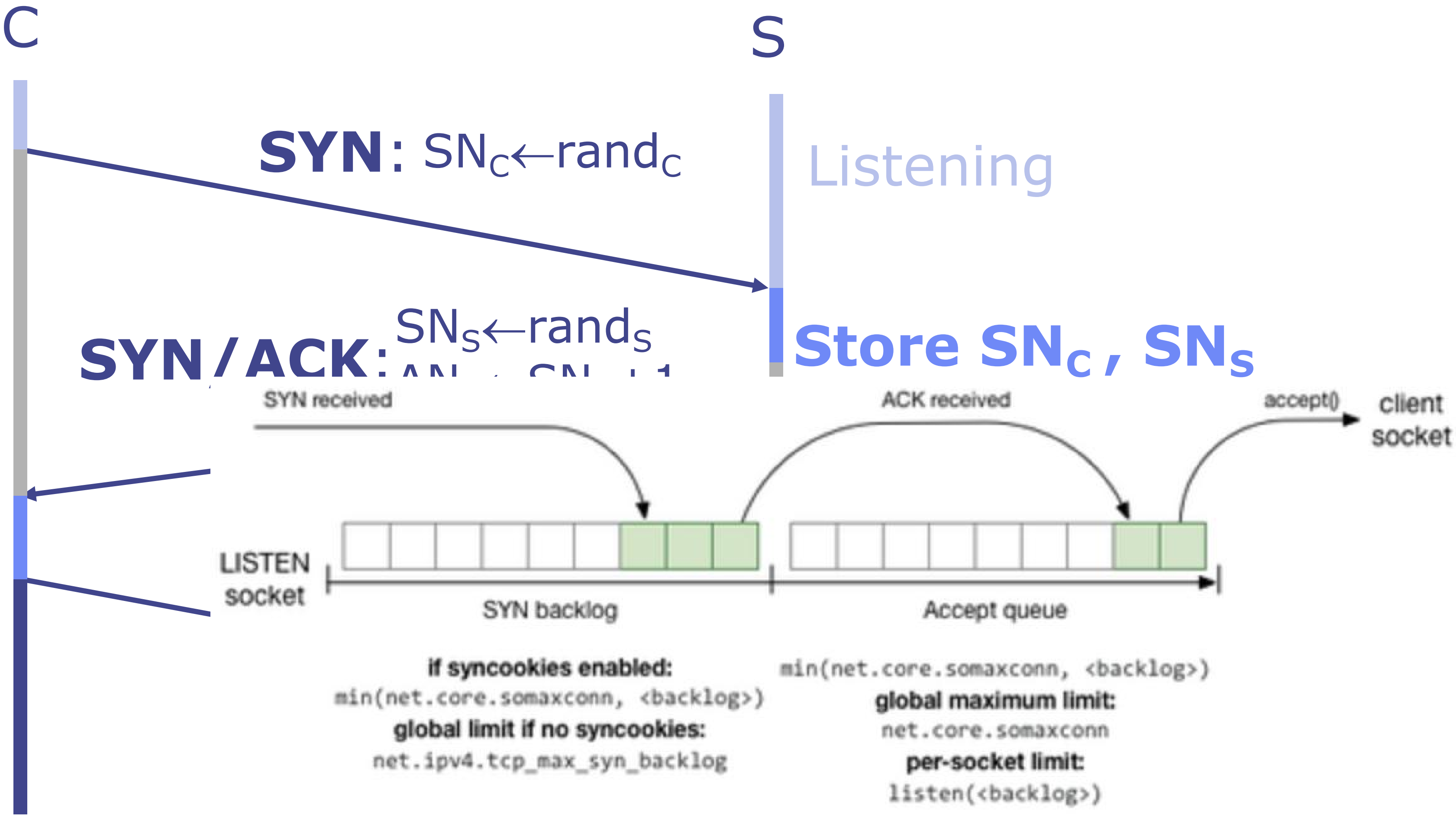
Listening

SYN/ACK: $SN_S \leftarrow rand_S$ $AN_S \leftarrow$

Store SN_C , SN_S

ACK: SN AN

TCP Handshake



C

S

SYN: $SN_C \leftarrow rand_C$

Listening

$SN_S \leftarrow rand_S$

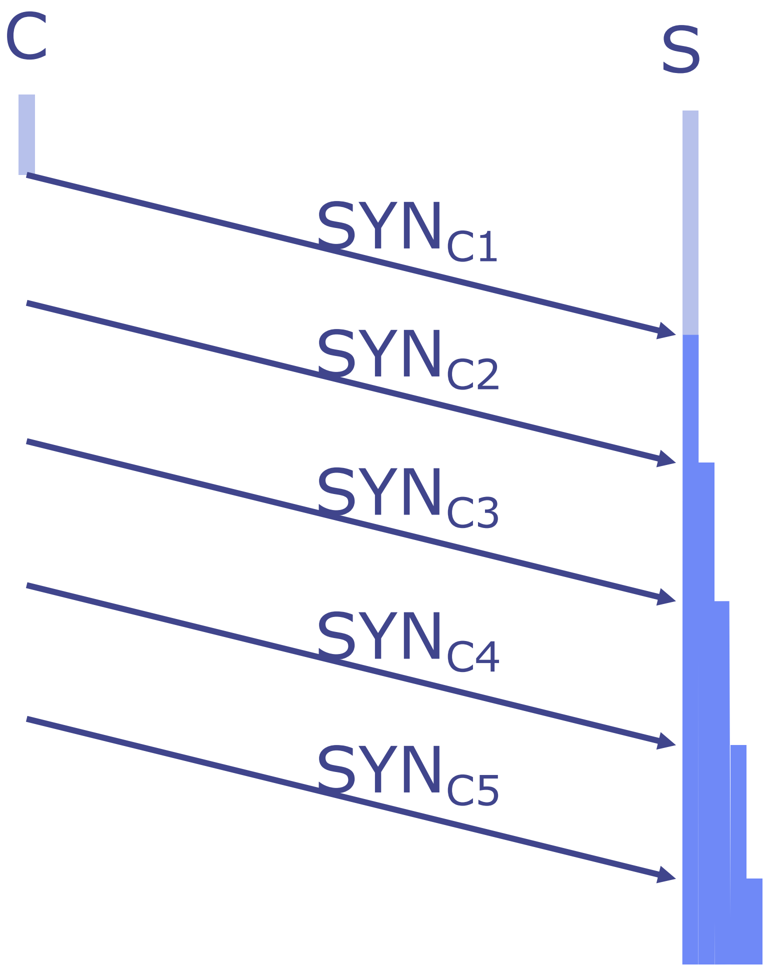
SYN/ACK: $AN \leftarrow SN + 1$

Store SN_C, SN_S

if syncookies enabled:
`min(net.core.somaxconn, <backlog>)`
global limit if no syncookies:
`net.ipv4.tcp_max_syn_backlog`

`min(net.core.somaxconn, <backlog>)`
global maximum limit:
`net.core.somaxconn`
per-socket limit:
`listen(<backlog>)`

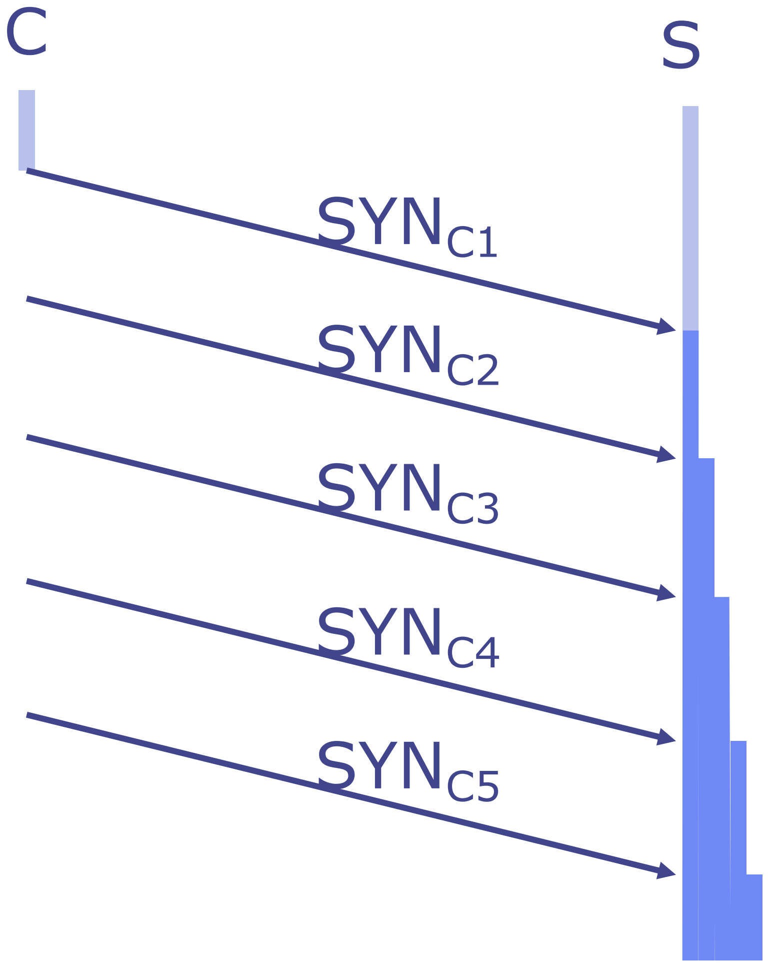
TCP SYN Flood



Single machine:

- SYN packets with random source IP addresses
- Fill up backlog queue on server
- No further connections possible

TCP SYN Flood



IP Spoofing:

- SYN packets with random source IP addresses
- Fill up backlog queue on server
- No further connections possible

TCP SYN Flood

- **Queue size**

  commonly set as 128 by default on some Linux systems;

- **Timeout**

  evict a backlog entry if no ack is received until timeout, e.g., 3 mins

- **Attack example:**

  attacker sends 128 SYN every 3 mins without responding with ACK pkts

TCP SYN Flood

- **Attack principle**

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

- **Solution?**

TCP SYN Flood

- **Attack principle**

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

- **Solution?**

  increase backlog queue size

TCP SYN Flood

- **Attack principle**

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

- **Solution?**

  increase backlog queue size

  attacker sends more SYN packets!

TCP SYN Flood

- **Attack principle**

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

- **Solution?**

  decrease timeout

TCP SYN Flood

- **Attack principle**

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

- **Solution?**

  decrease timeout

  interrupt normal service requests!

SYN Cookies

- **Goal**

  avoid state storage on server

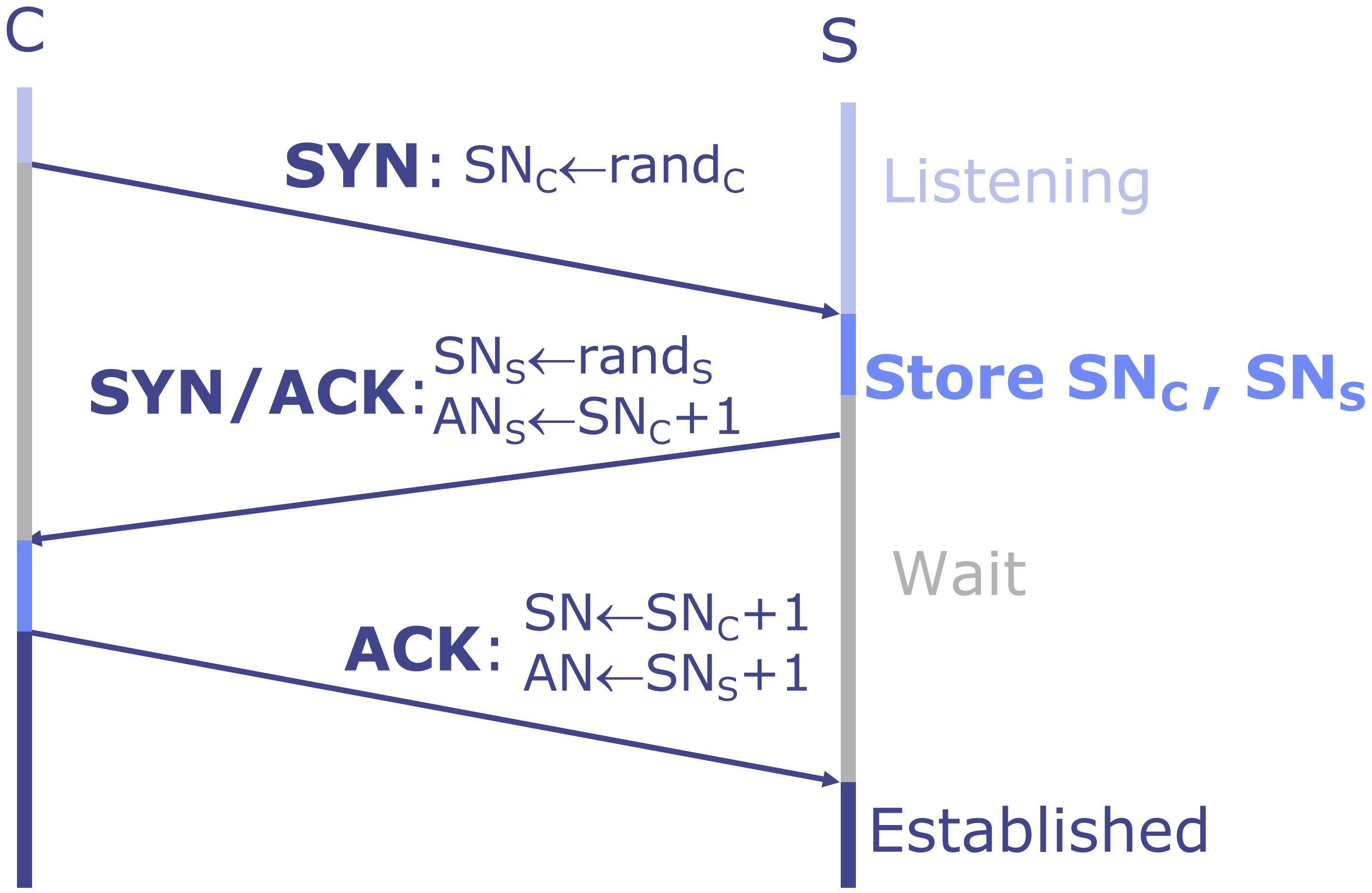
  until 3-way handshake completes

- **Idea**

  server sends necessary states to client along with SYN-ACK;

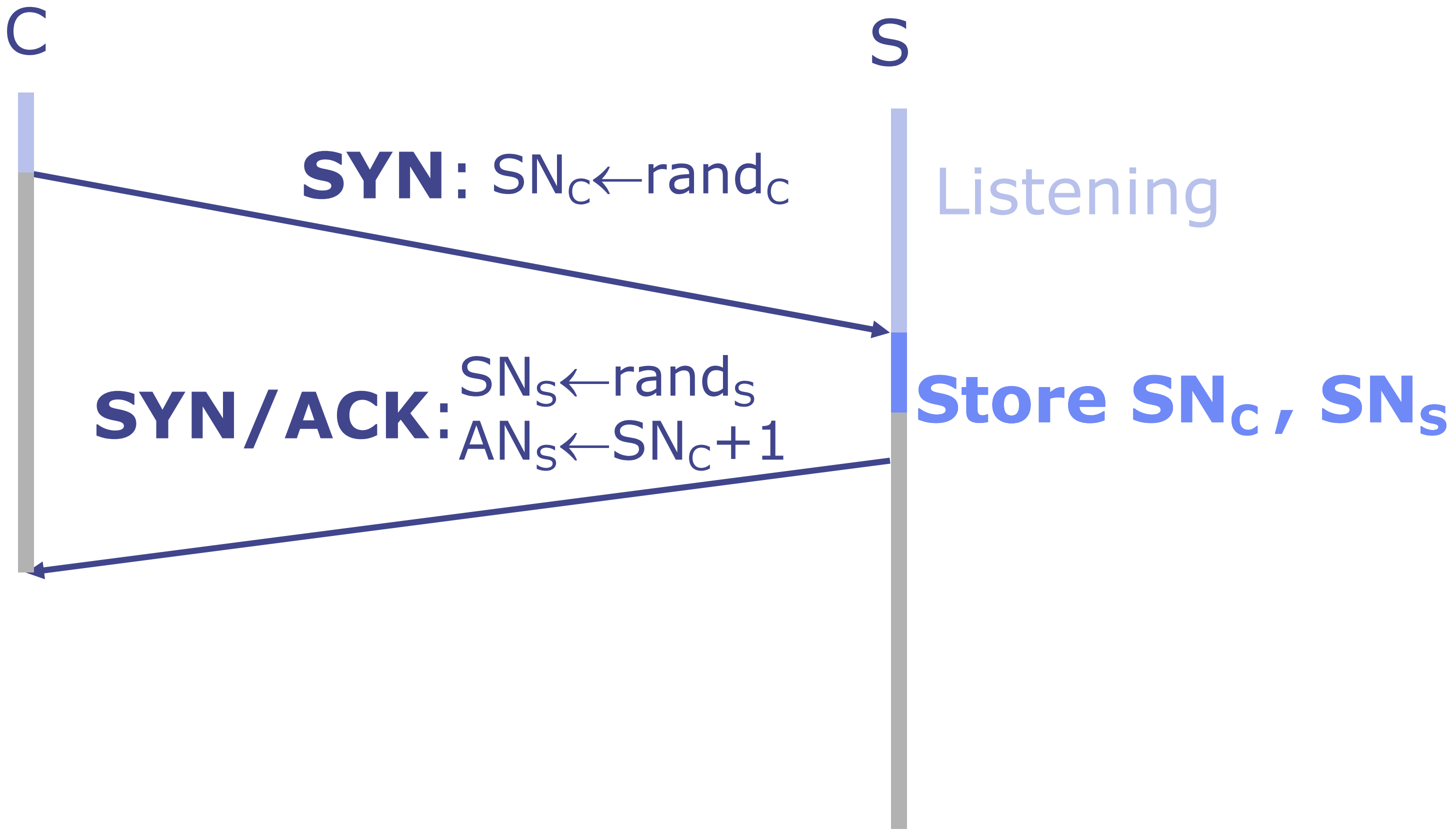
  client sends these states back to server along with ACK;

SYN Cookies



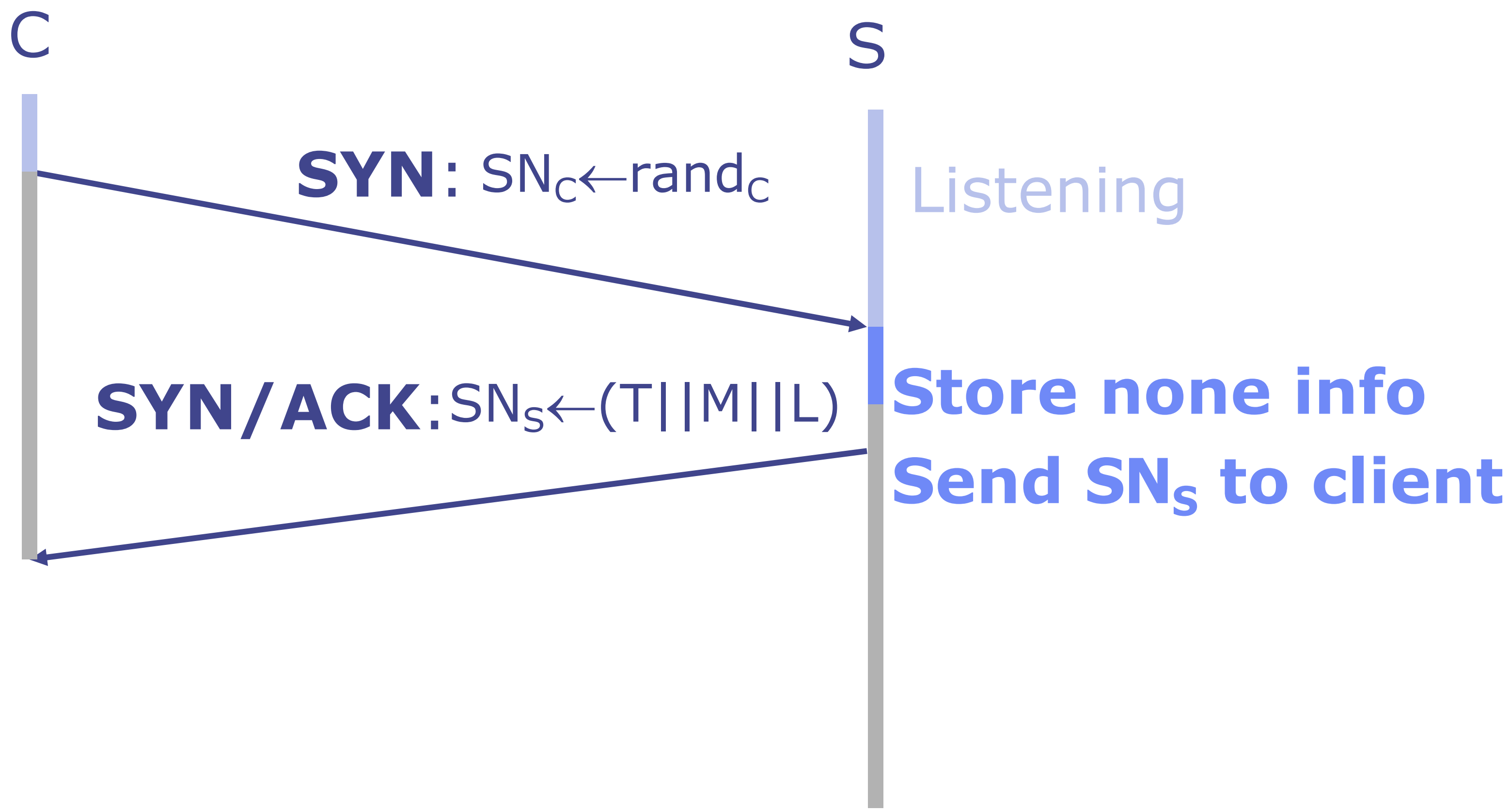
traditional TCP handshake

SYN Cookies


C
S
SYN: $SN_C \leftarrow rand_C$
Listening
SYN/ACK: $SN_S \leftarrow rand_S$ $AN_S \leftarrow SN_C+1$
Store SN_C , SN_S

traditional TCP handshake

SYN Cookies



C

S

SYN: $SN_C \leftarrow rand_C$

Listening

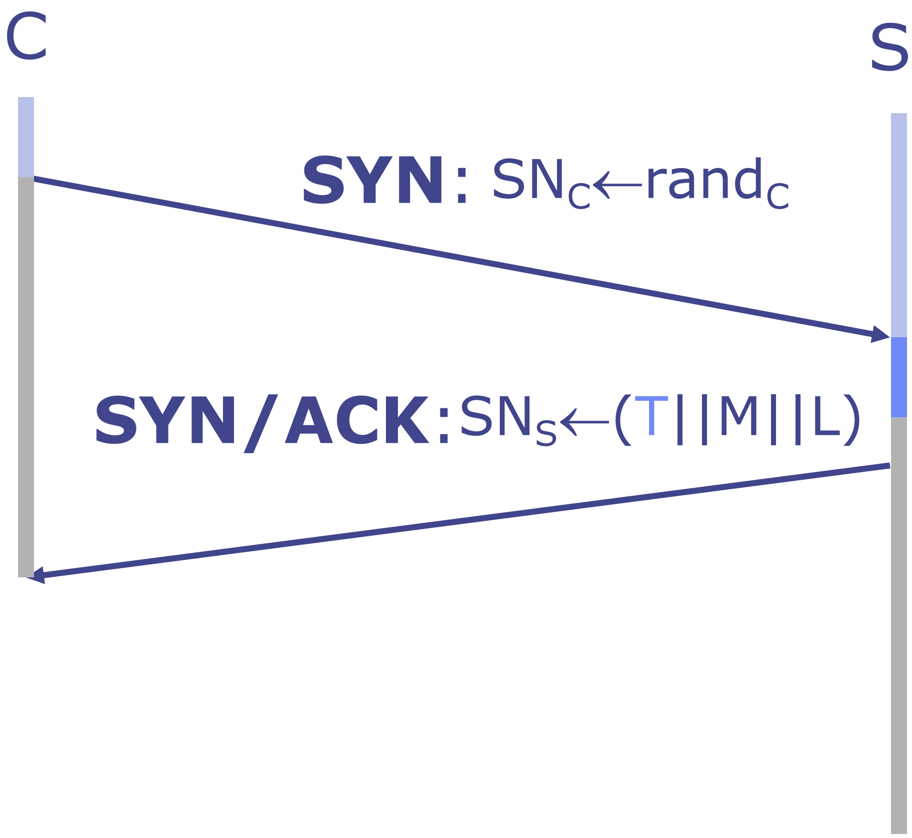
SYN/ACK: $SN_S \leftarrow (T||M||L)$

Store none info

Send SN_S to client

SYN cookies

SYN Cookies



C

S

SYN: $SN_C \leftarrow rand_C$

Listening

SYN/ACK: $SN_S \leftarrow (T||M||L)$

Store none info

Send SN_S to client

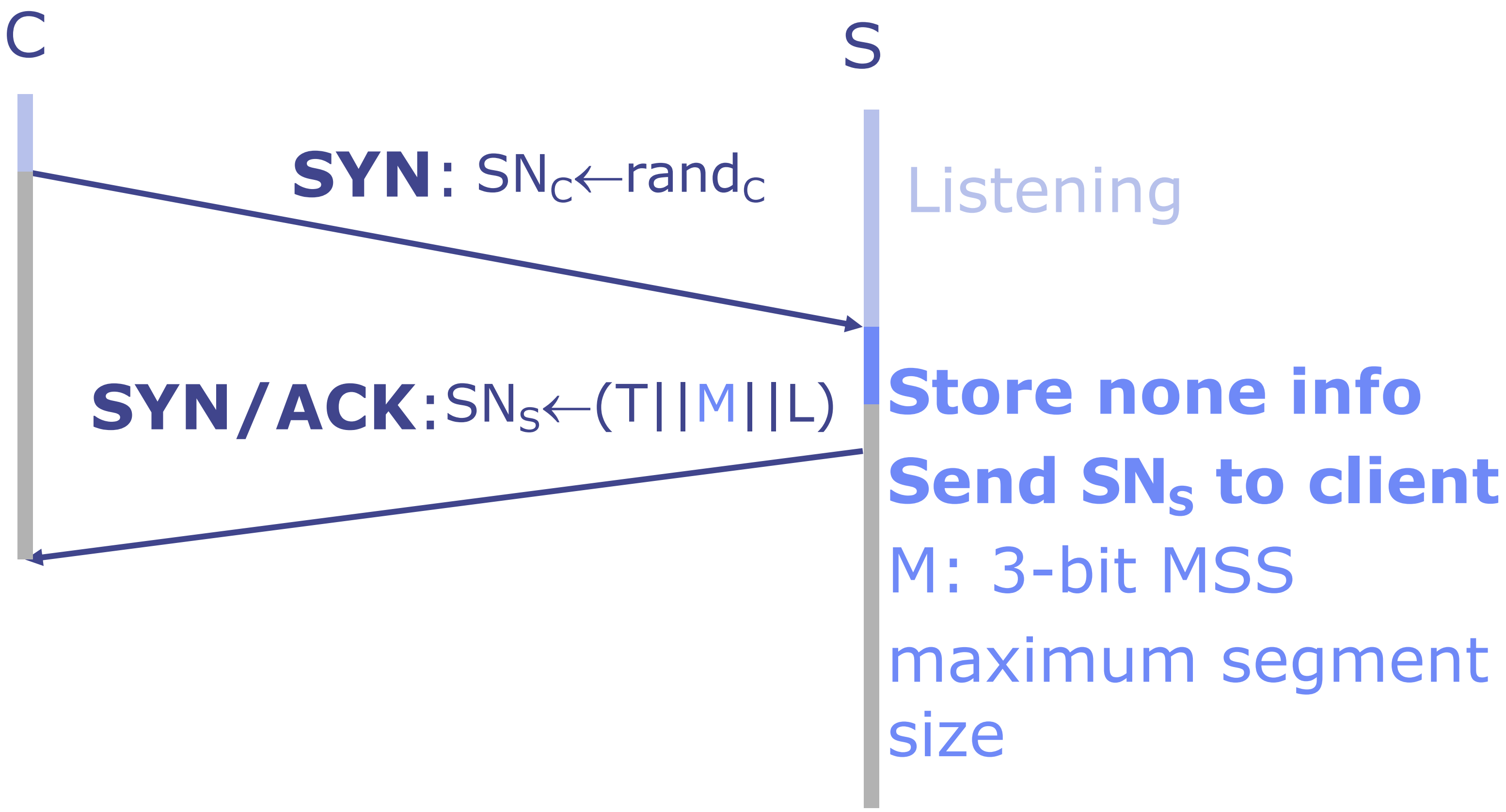
T: 5-bit timestamp

time() logically right-shifted 6 positions;

64-second resolution;

SYN cookies

SYN Cookies



C

S

SYN: $SN_C \leftarrow rand_C$

Listening

SYN/ACK: $SN_S \leftarrow (T||M||L)$

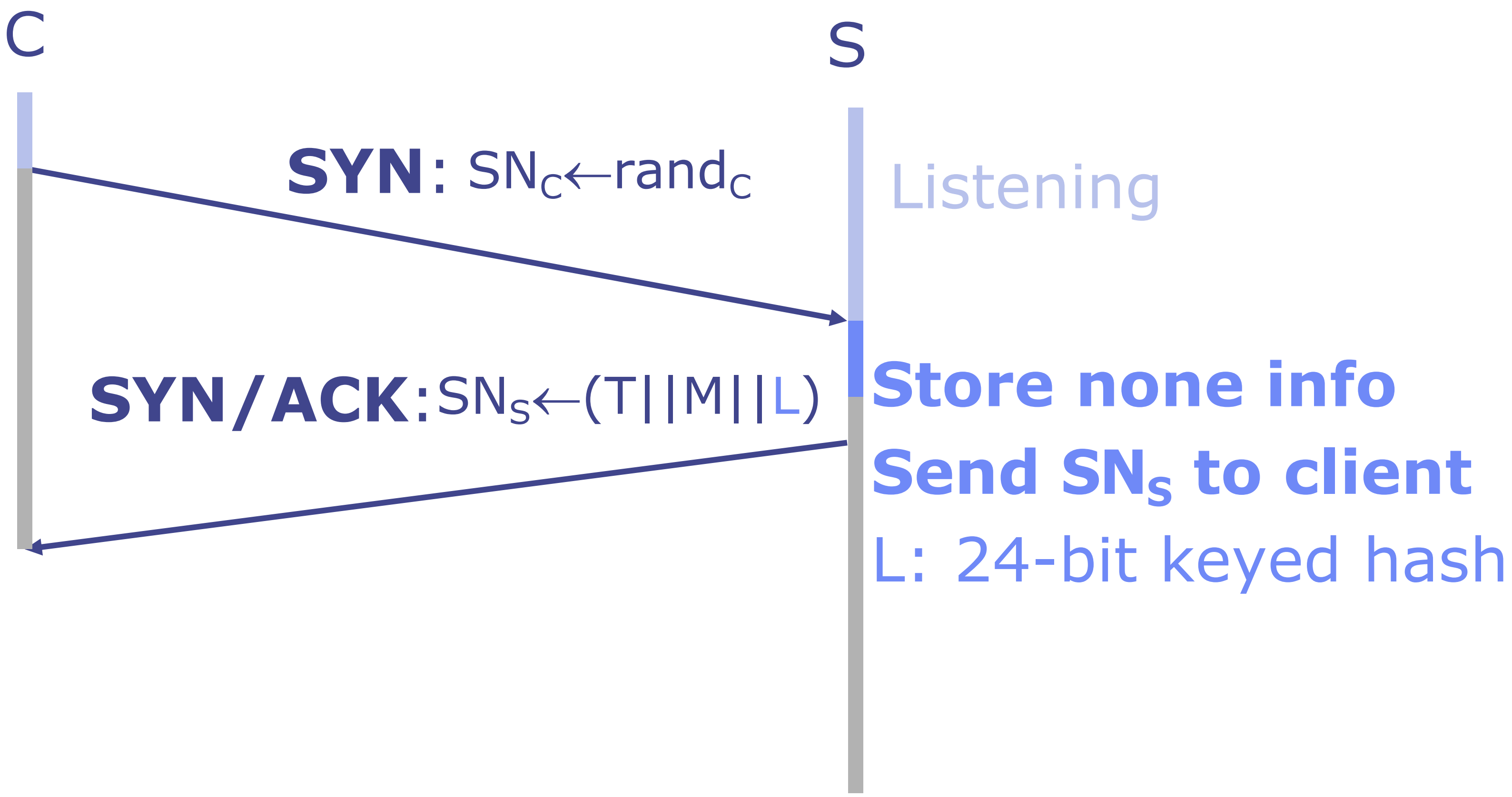
Store none info

Send SN_S to client

M: 3-bit MSS

maximum segment size

SYN Cookies



C

S

SYN: $SN_C \leftarrow rand_C$

Listening

SYN/ACK: $SN_S \leftarrow (T||M||L)$

Store none info

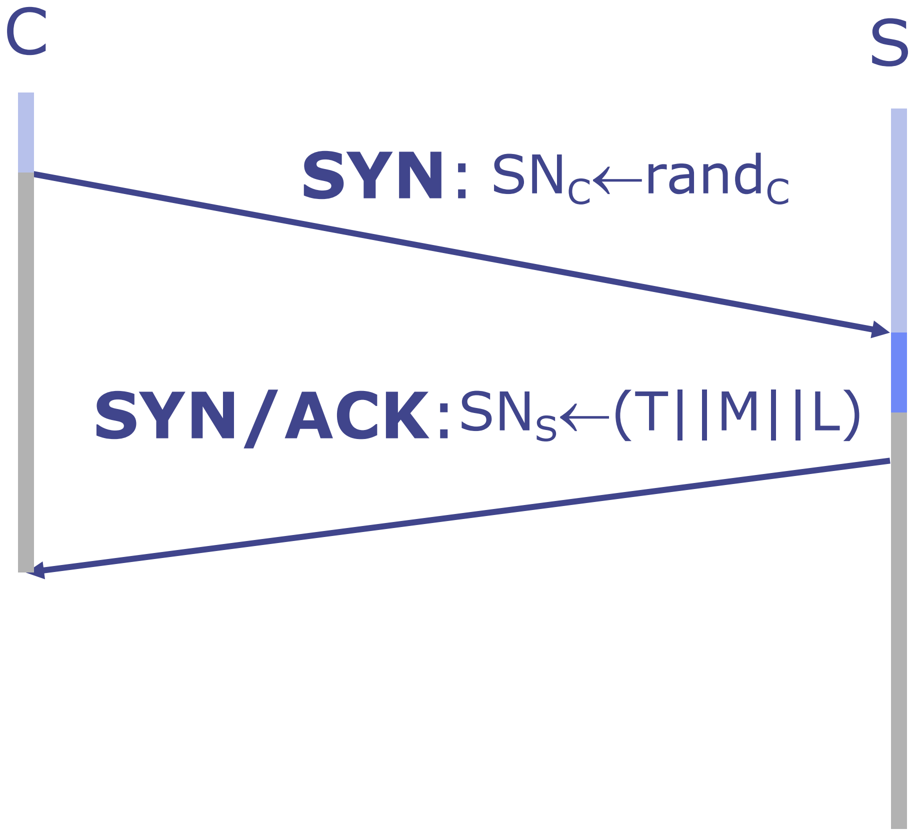
Send SN_S to client

L: 24-bit keyed hash

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

SYN cookies

SYN Cookies



C

S

SYN: $SN_C \leftarrow rand_C$

Listening

SYN/ACK:$SN_S \leftarrow (T||M||L)$

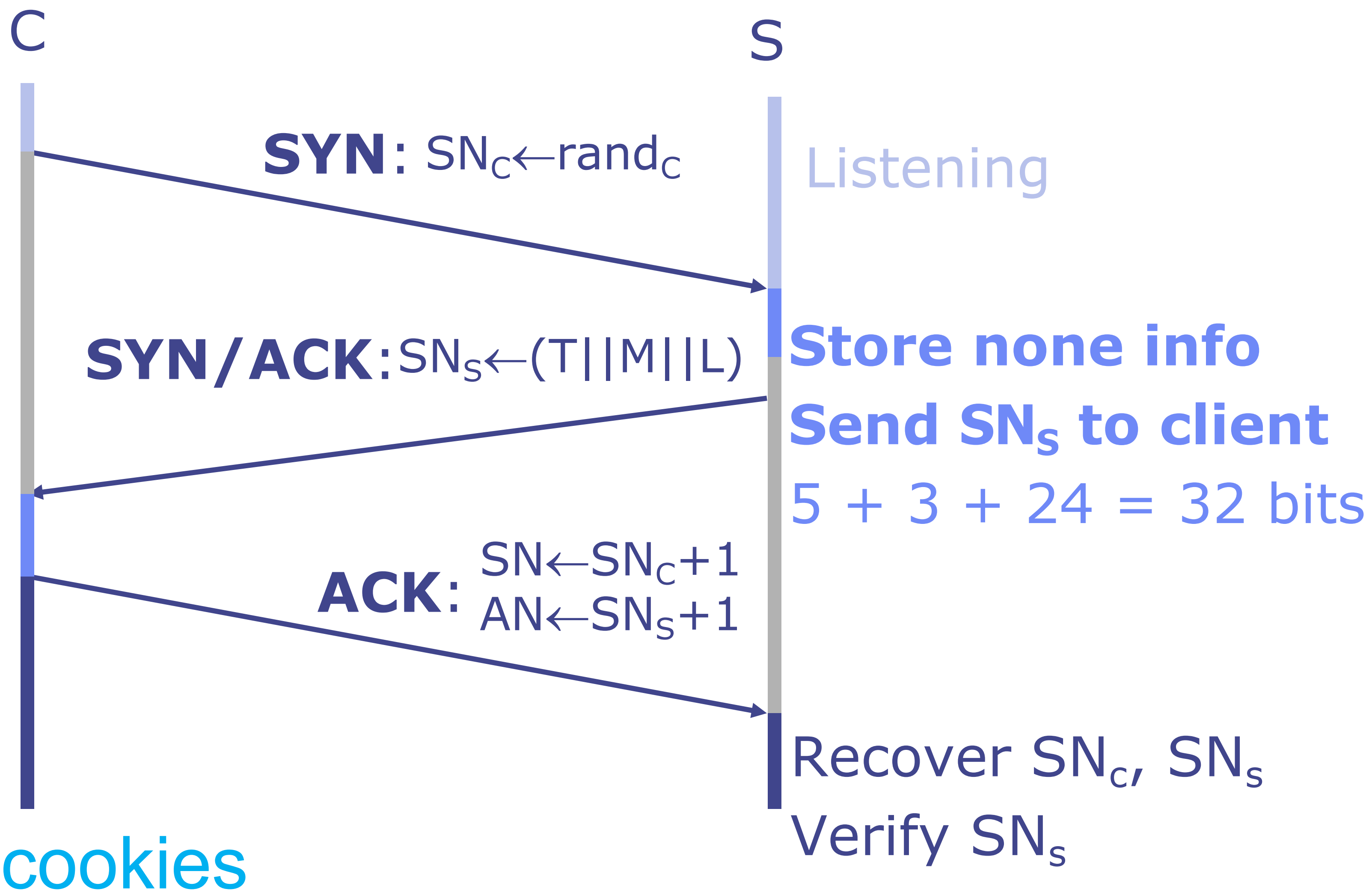
Store none info

Send SN_S to client

5 + 3 + 24 = 32 bits

SYN cookies

SYN Cookies



C

S

SYN: $SN_C \leftarrow rand_C$

Listening

SYN/ACK: $SN_S \leftarrow (T||M||L)$

Store none info

Send SN_S to client

5 + 3 + 24 = 32 bits

ACK: $SN \leftarrow SN_C+1$ $AN \leftarrow SN_S+1$

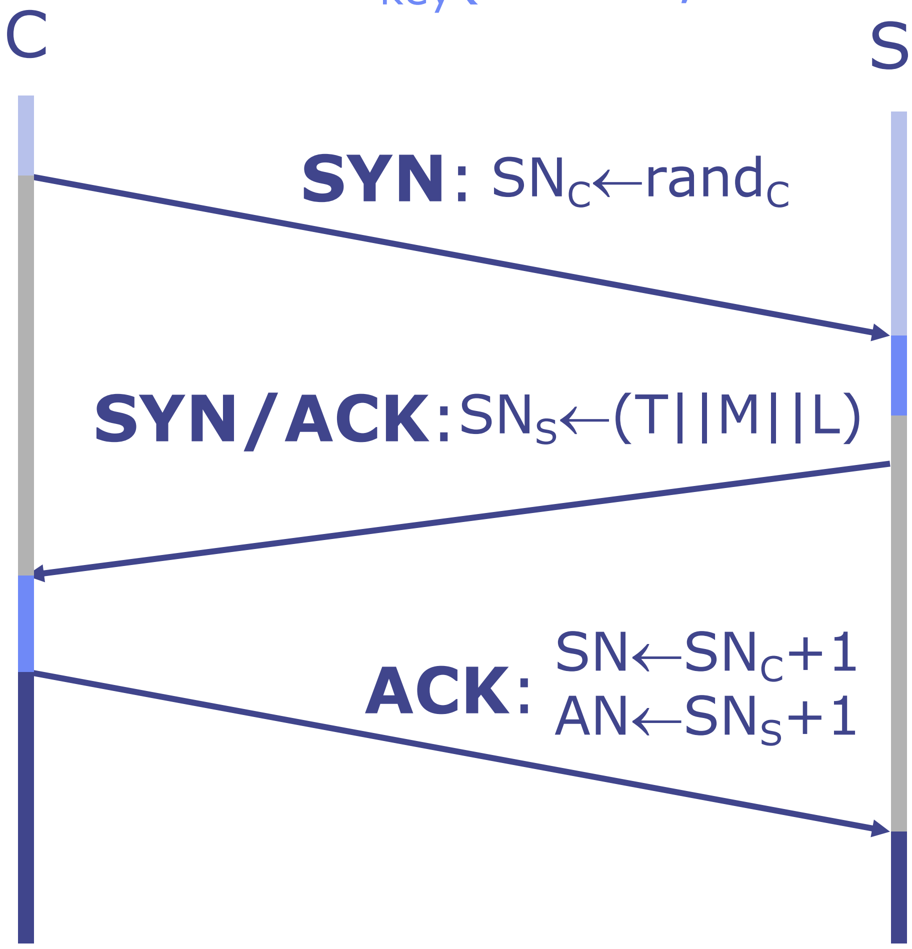
Recover SN_c, SN_s

Verify SN_s

SYN cookies

SYN Cookies

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



C

S

SYN: $SN_C \leftarrow rand_C$

Listening

SYN/ACK: $SN_S \leftarrow (T||M||L)$

Store none info

Send SN_S to client

5 + 3 + 24 = 32 bits

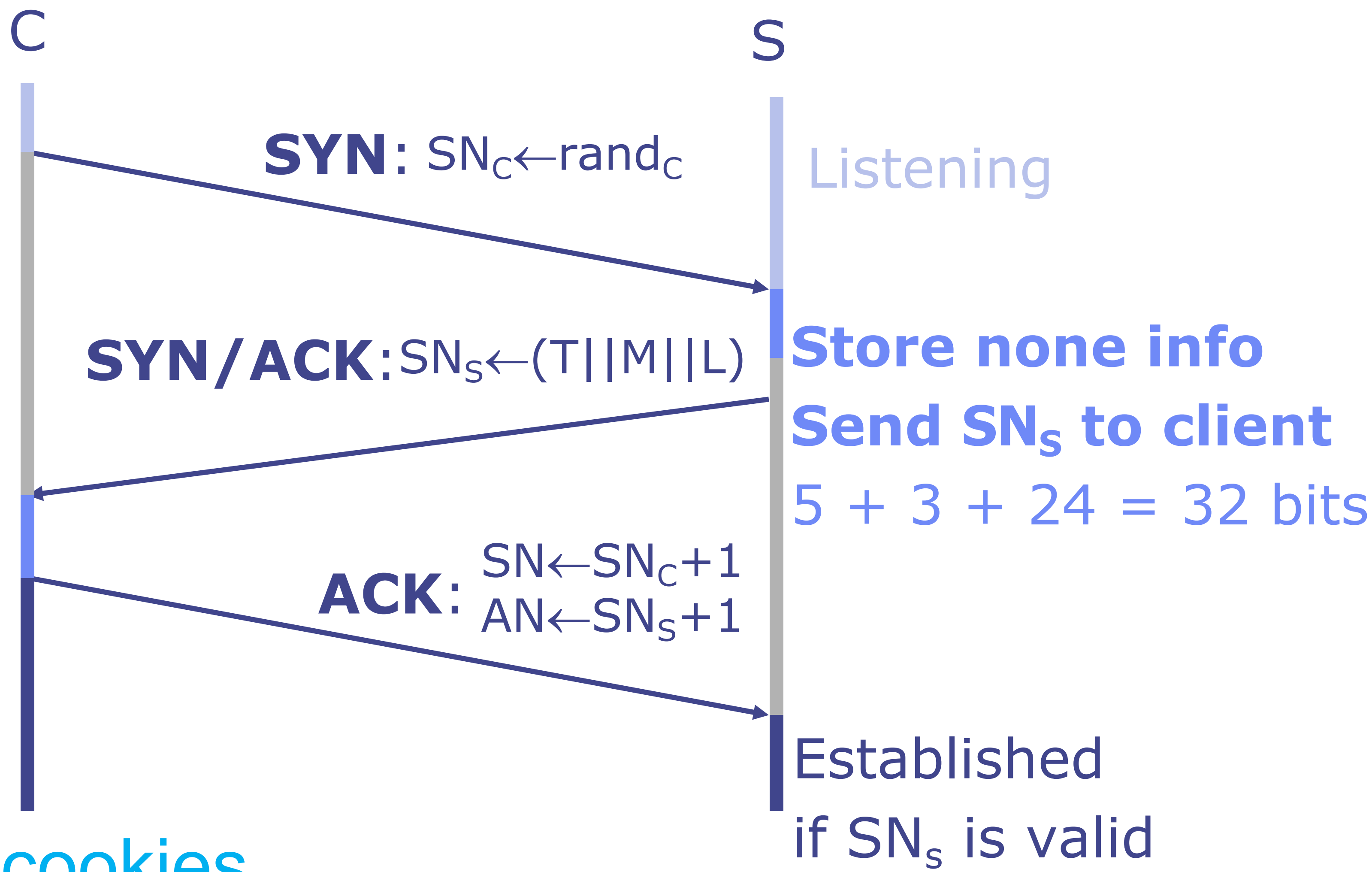
ACK: $SN \leftarrow SN_C+1$
$AN \leftarrow SN_S+1$

Recover SN_c, SN_s

Recompute SN_s

SYN cookies

SYN Cookies



C

S

SYN: $SN_C \leftarrow rand_C$

Listening

SYN/ACK: $SN_S \leftarrow (T||M||L)$

Store none info

Send SN_S to client

5 + 3 + 24 = 32 bits

ACK: $SN \leftarrow SN_C+1$
$AN \leftarrow SN_S+1$

Established

if SN_s is valid

SYN cookies

TCP SYN Flood Backscatter

- SYN with forged source IP $\Rightarrow$ SYN-ACK to random host

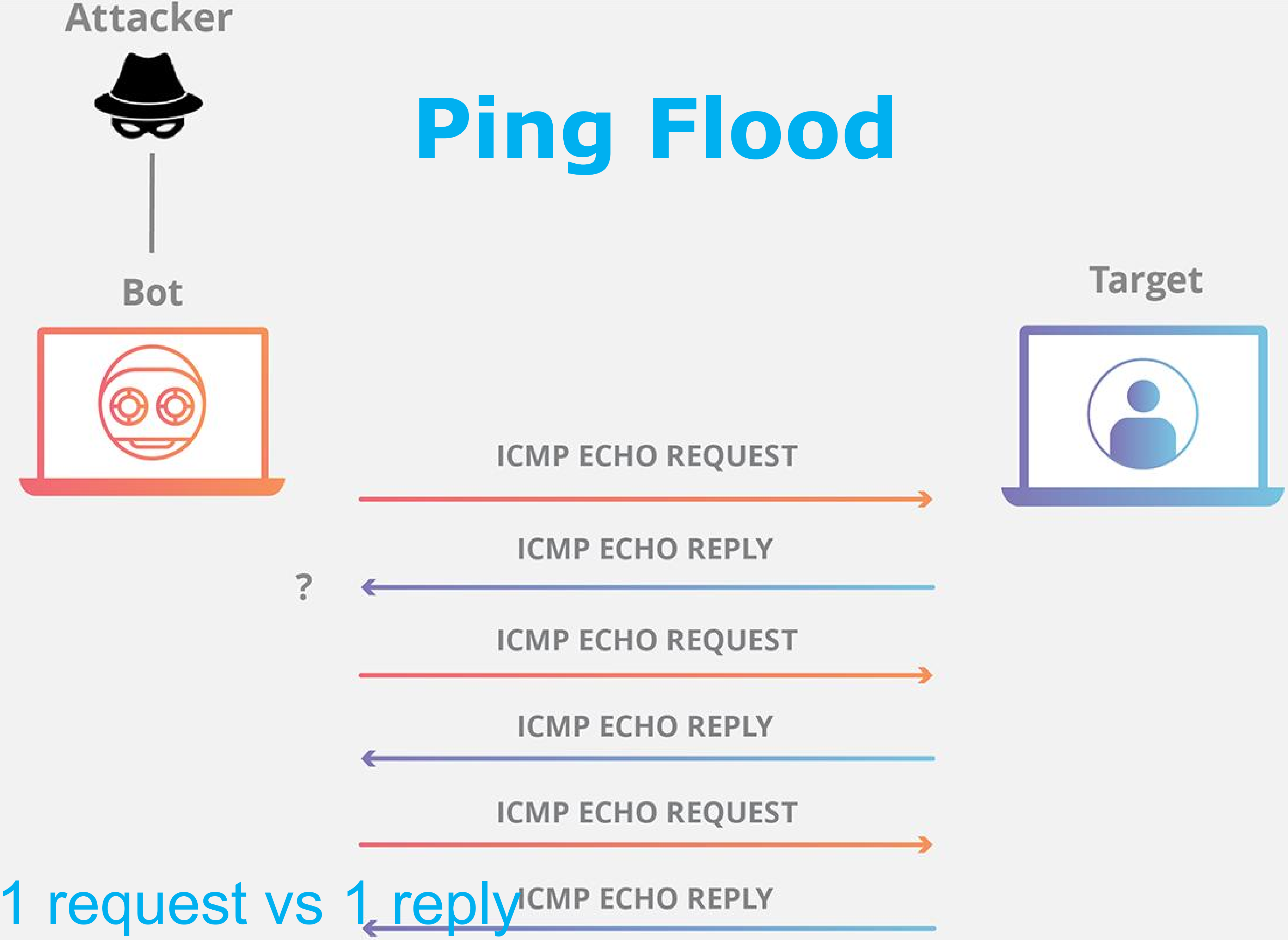
TCP SYN Flood Backscatter

- SYN with forged source IP ⇒ SYN-ACK to random host

backscatter packets can be used for detecting DDoS

DDoS attacks so far

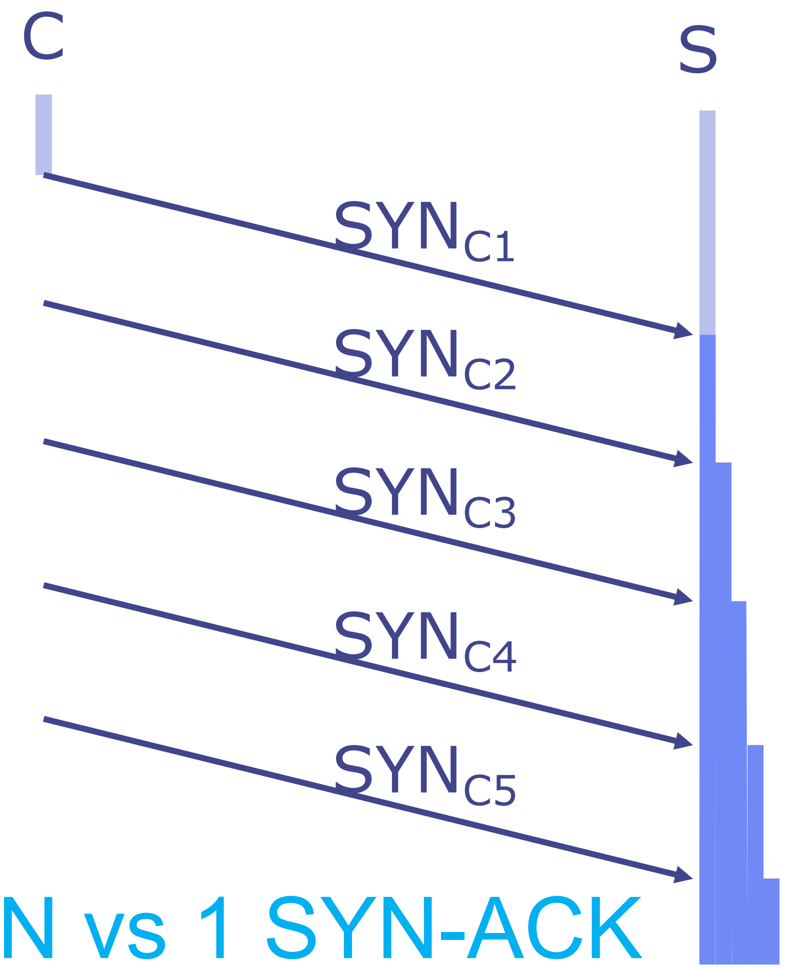
Ping Flood



1 request vs 1 reply

https://www.cloudflare.com/learning/ddos/ping-icmp-flood-ddos-attack/

TCP SYN Flood



1 SYN vs 1 SYN-ACK

Single machine:

- SYN packets with random source IP addresses
- Fill up backlog queue on server
- No further connections possible

symmetric DDoS attack

the amount of bandwidth the targeted device consumes is simply the sum of the total traffic sent from each attacker/bot;

symmetric DDoS attack

the attacker requires a substantial amount of traffic to succeed;

how to attack with less effort?

asymmetric DDoS attack

a relatively small number or low levels of resources are required by an attacker to cause a significantly greater number or higher level of target resources to malfunction or fail

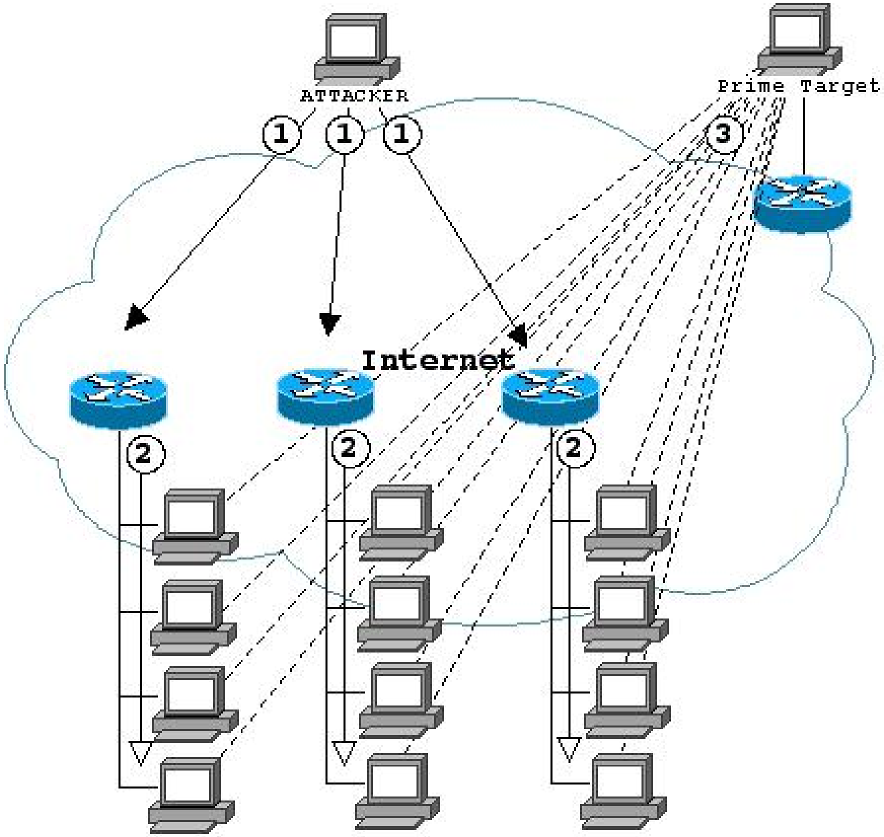
Smurf Attack

- Amplify the effect of ping flood
- Exploit IP broadcast address
- Forward the single ICMP Echo Request to any other hosts in the same network
- Each host responds with an ICMP Echo Reply

- 1 request vs many replies

Smurf Attack

- Amplify the effect of ping flood
- 1 request vs many replies

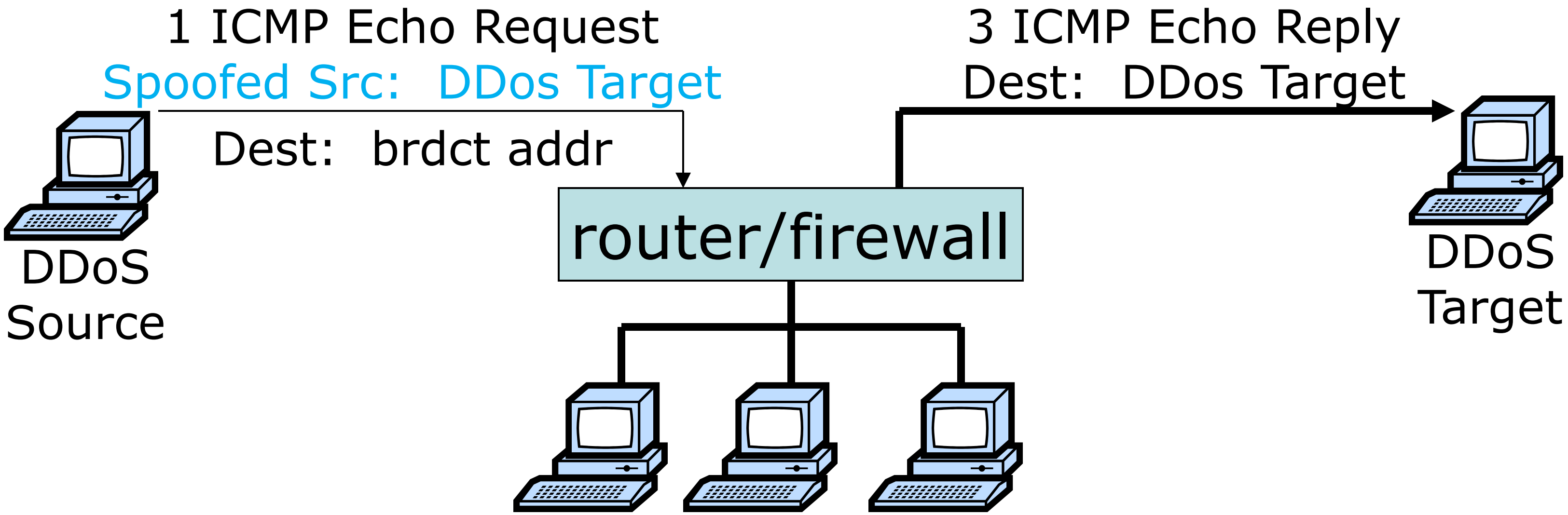


router and firewall
as amplifier

shorturl.at/iBDQR

Smurf Attack

- Attack with an ICMP Echo Request with spoofed source IP address of the targeted server and destination IP address of an IP broadcast address

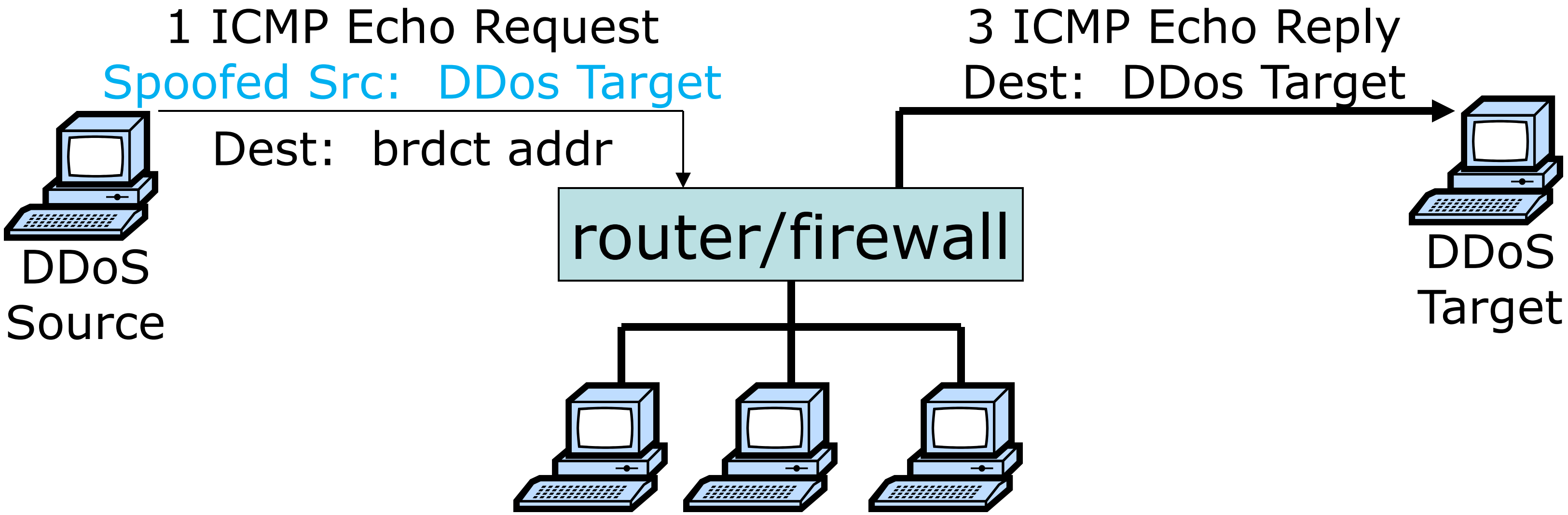


Smurf Attack

- **Solution**

  disable IP broadcast addresses on router and firewall, or

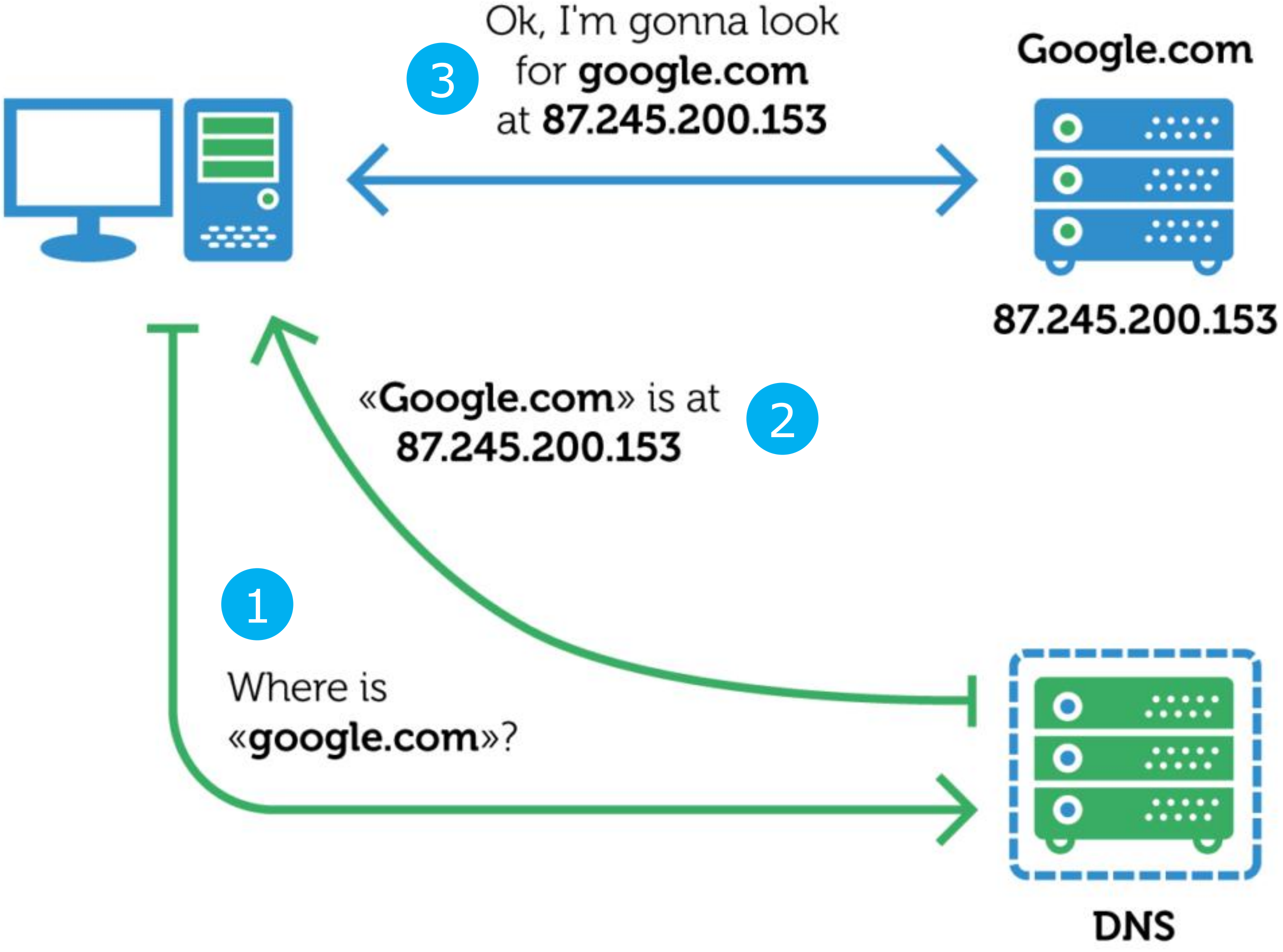
  reject external packets to brdct addr



asymmetric DDoS attack

any other amplifiers?

DNS Resolver

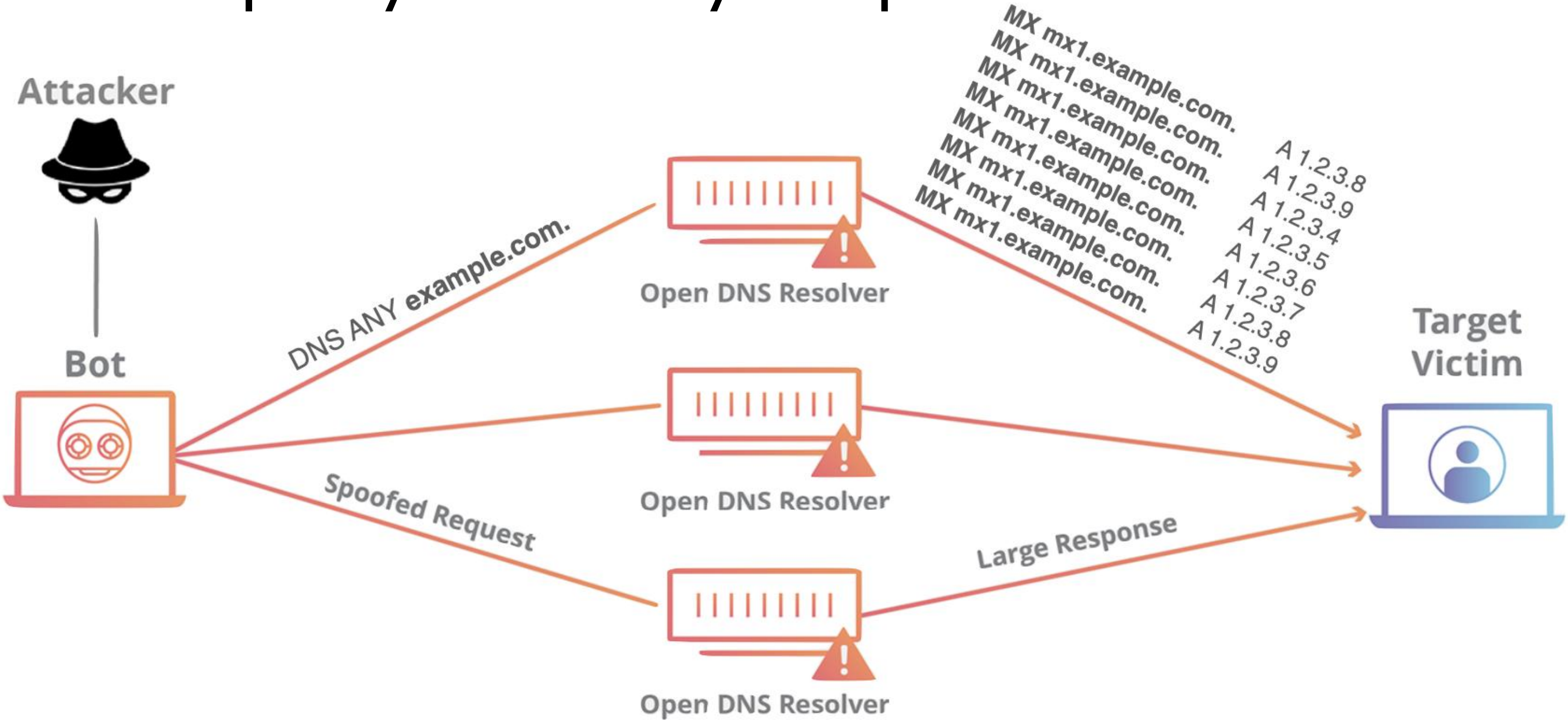


DNS Amplification Attack

- Leverage open DNS resolvers
- Exploit DNS query of type ANY that retrieves all the available types for a given name

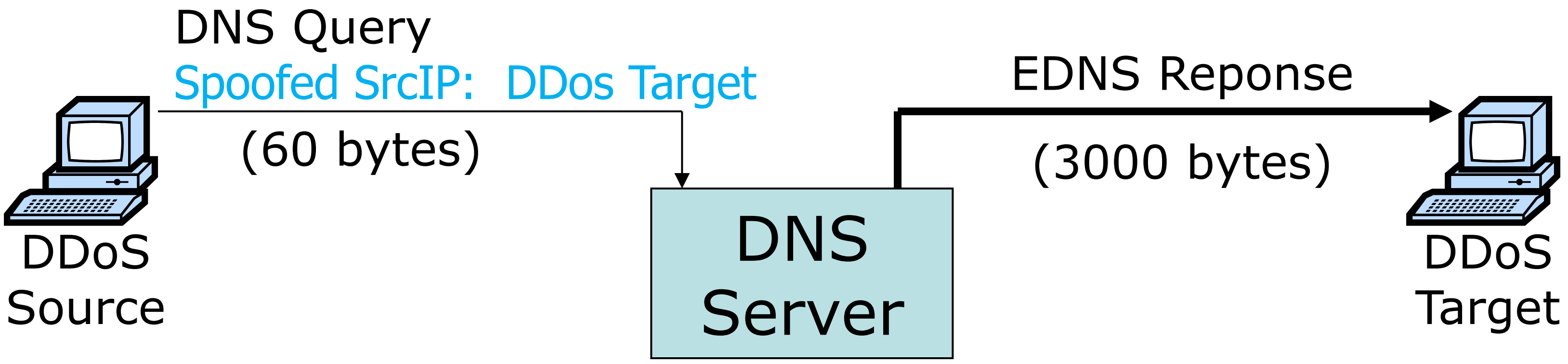
DNS Amplification Attack

- Amplify the effect of DNS query
- 1 query vs many responses



DNS Amplification Attack

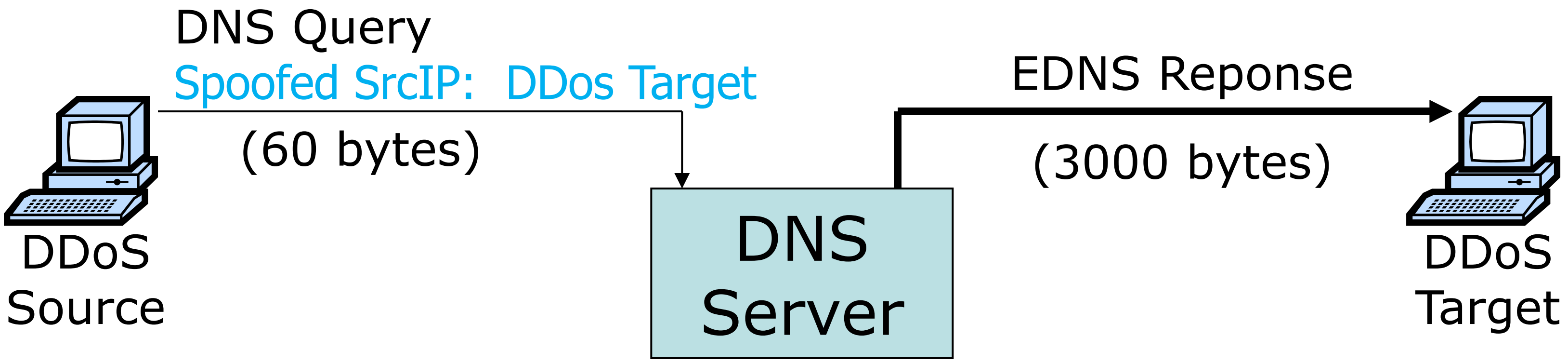
- Attack with an ANY-type DNS query with spoofed source IP address of the targeted server



DNS Amplification Attack

- Attack with an ANY-type DNS query with spoofed source IP address of the targeted server

  EDNS: Extension Mechanisms for DNS sends DNS data in larger UDP packets

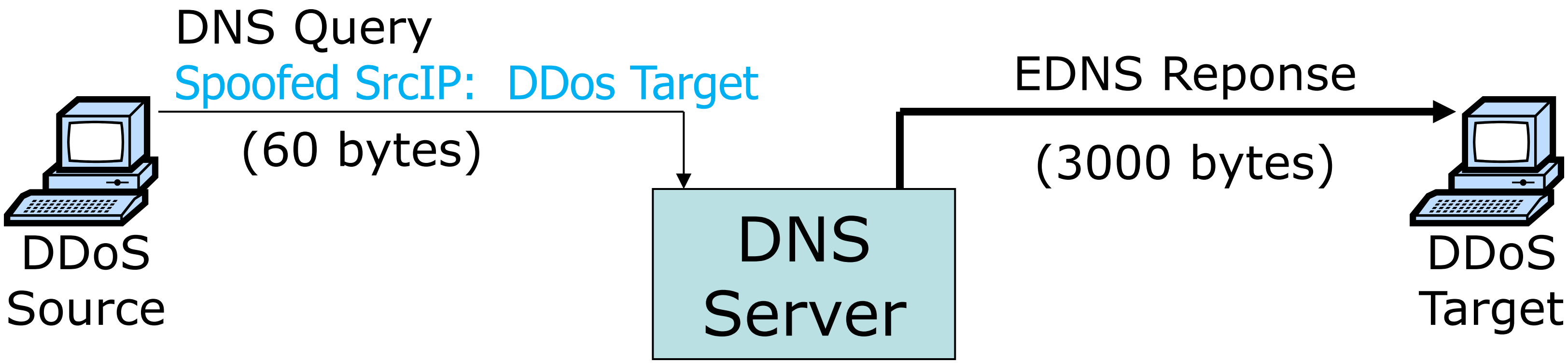


DNS Amplification Attack

- **Solution**

  reduce the number of open resolvers;

  source IP verification – stop spoofed packets leaving network;

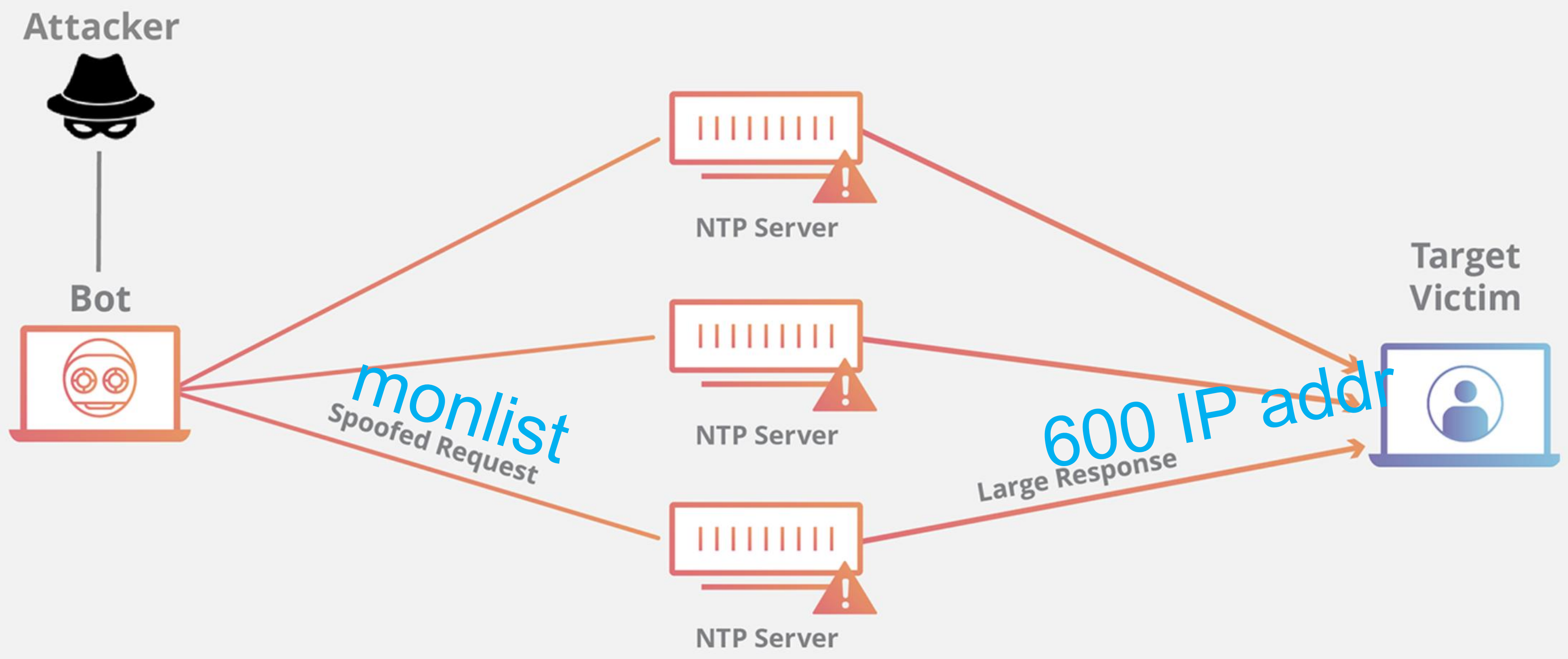


NTP Amplification Attack

- Leverage Network Time Protocol (NTP) servers
- Exploit monlist command that triggers a response with the last 600 source IP addresses of requests made to the NTP server

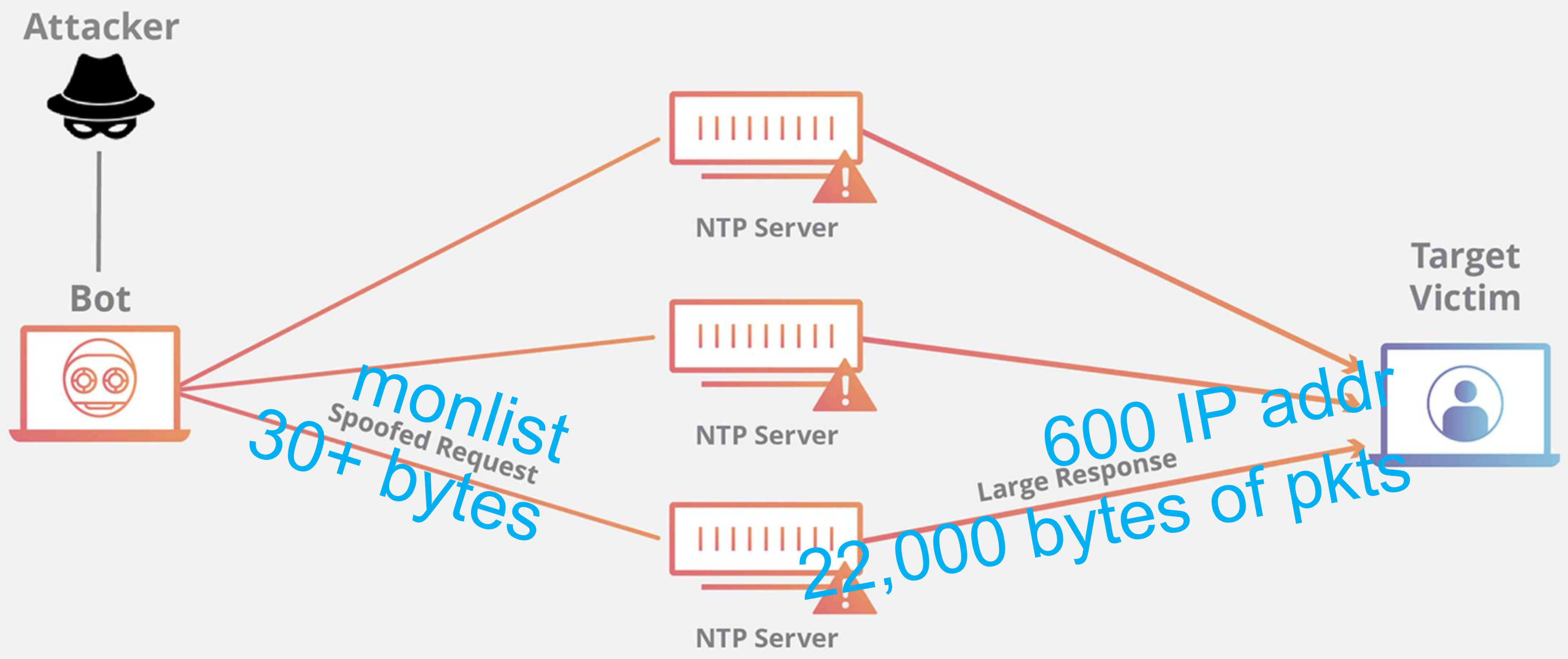
NTP Amplification Attack

- Amplify the effect of NTP query
- 1 query vs a large response



NTP Amplification Attack

- Amplify the effect of NTP query
- 1 query vs a large response

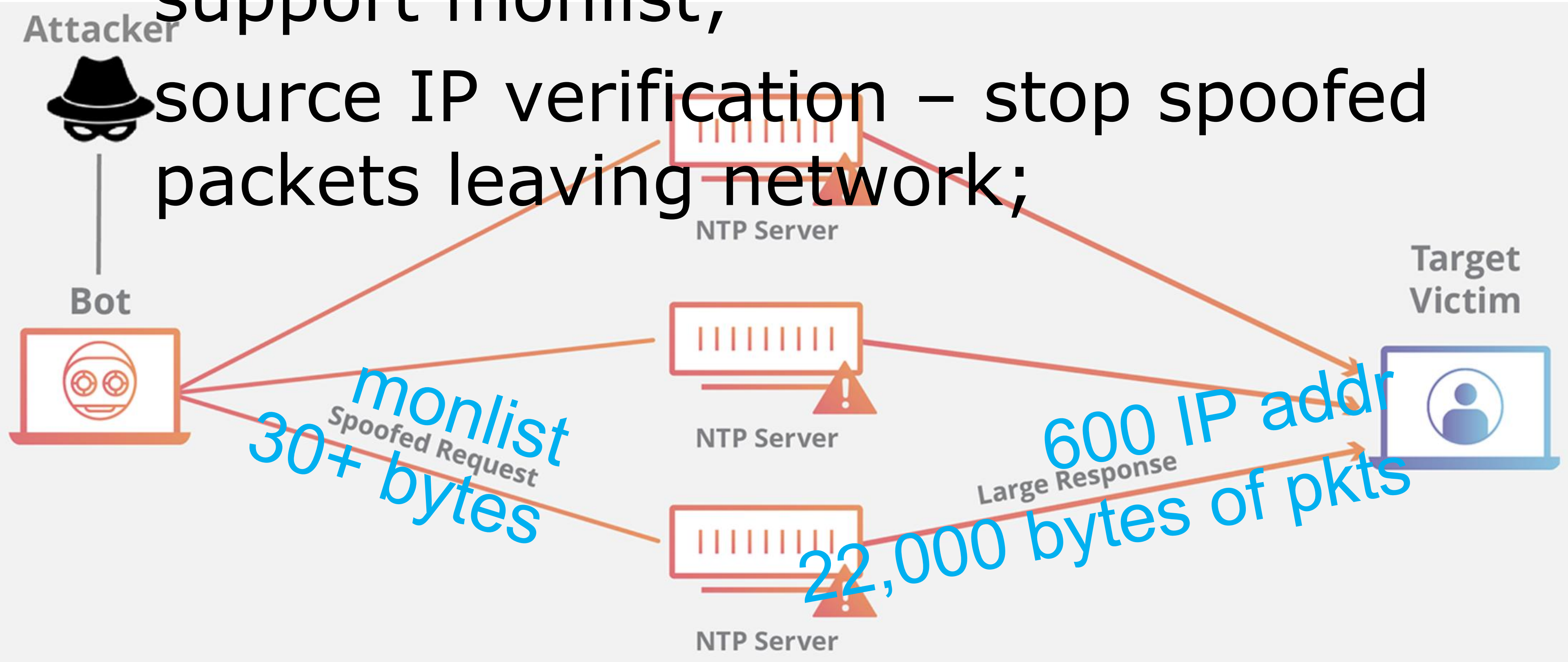


NTP Amplification Attack

- **Solution**

  reduce the number of NTP servers that support monlist;

  source IP verification – stop spoofed packets leaving network;

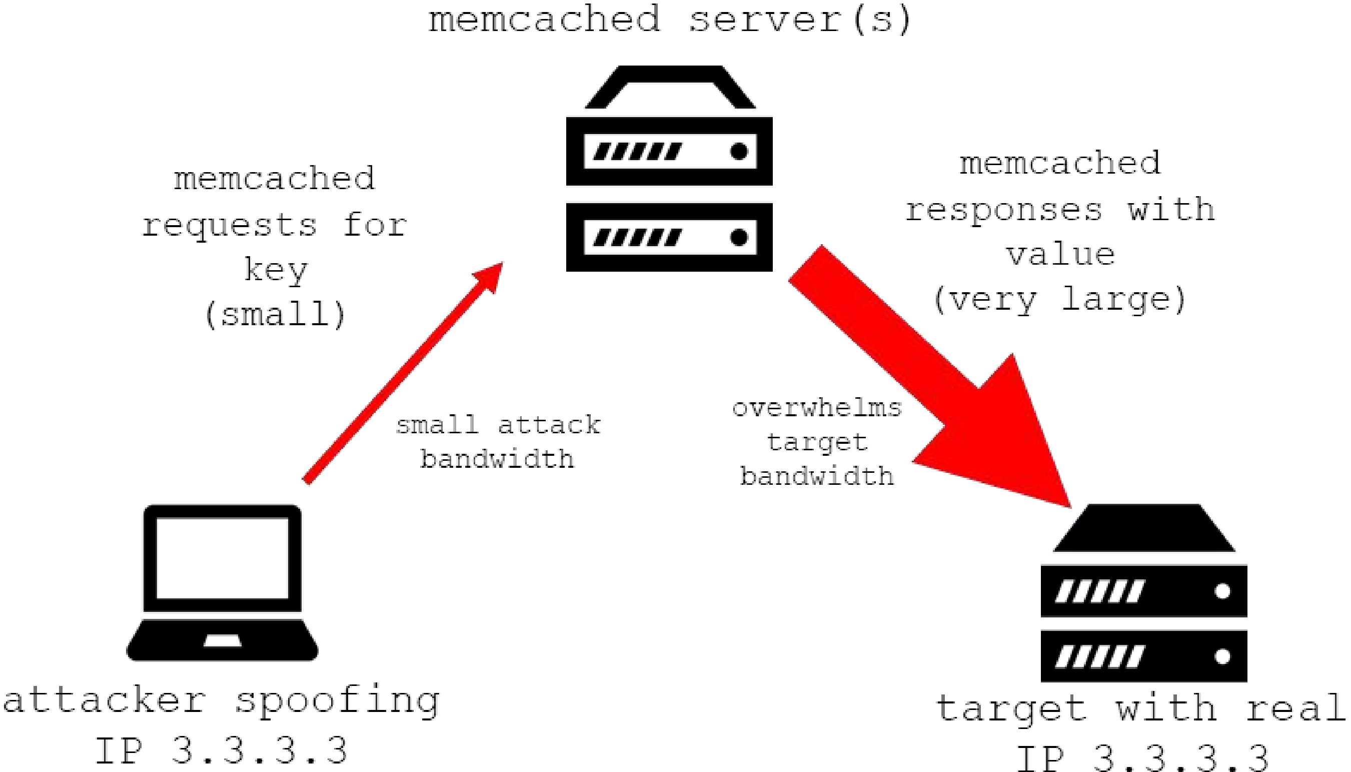


Memcached Attack

- Leverage Memcached servers
  a general-purpose distributed memory-caching system for speeding up websites and networks
- Exploit memcached request that triggers a response with a large volume of data to target

Memcached Attack

- Exploit memcached request that triggers a response with a large volume of data to target


memcached server(s)
memcached requests for key (small)
small attack bandwidth
memcached responses with value (very large)
overwhelms target bandwidth
attacker spoofing IP 3.3.3.3
target with real IP 3.3.3.3

Memcached Attack

- **Attack principle**

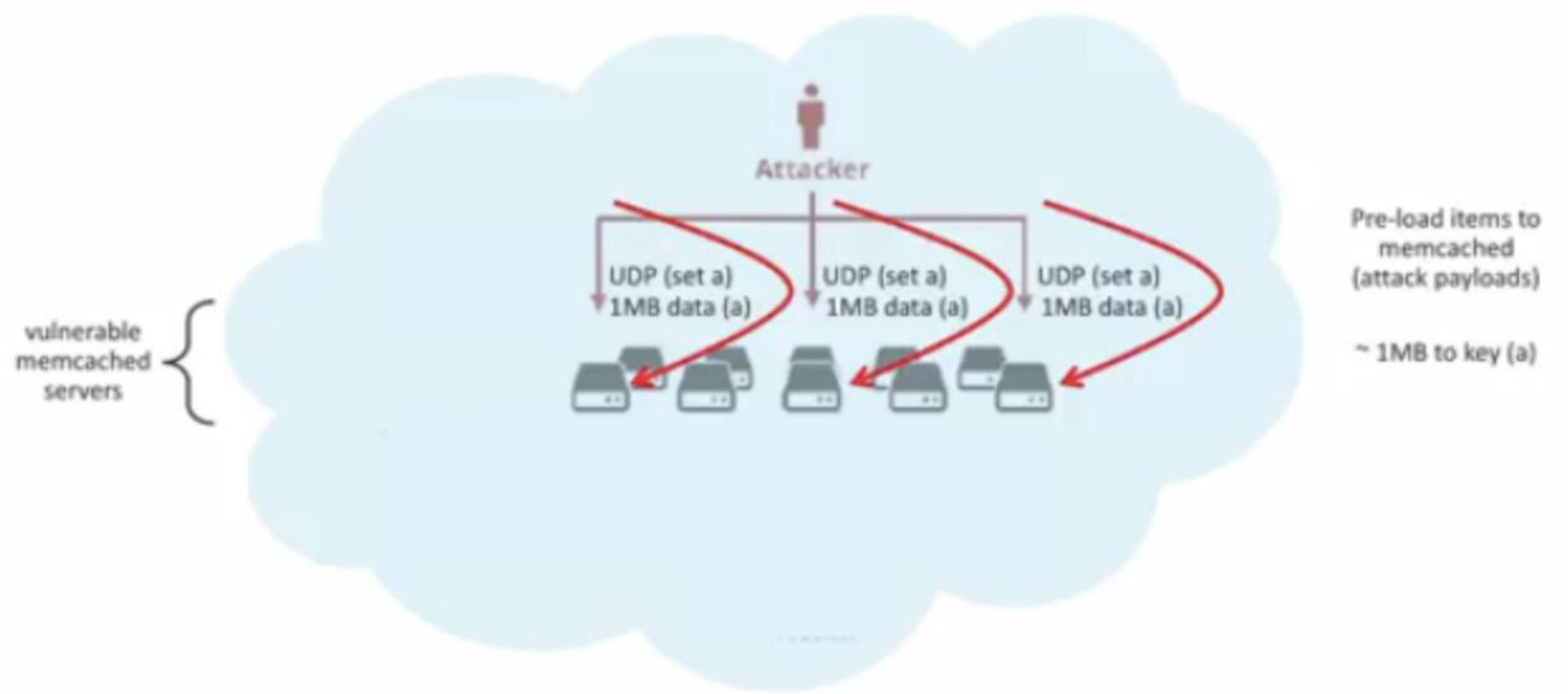
  preload large data to Memcached server;

  spoof request to preloaded data from target;

Memcached Attack

- **Attack principle**

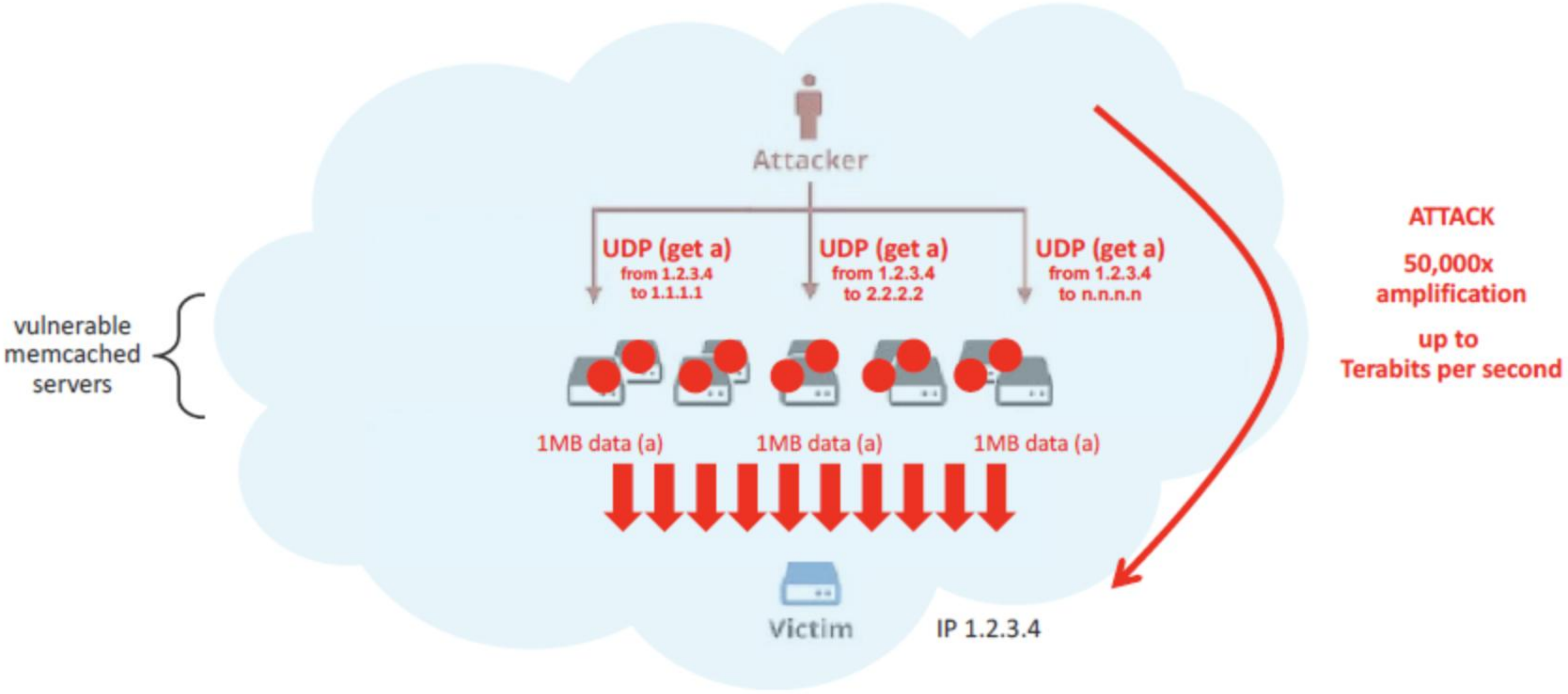
  preload large data to Memcached server;



Memcached Attack

- **Attack principle**

  spoof request to preloaded data from target;



Memcached Attack

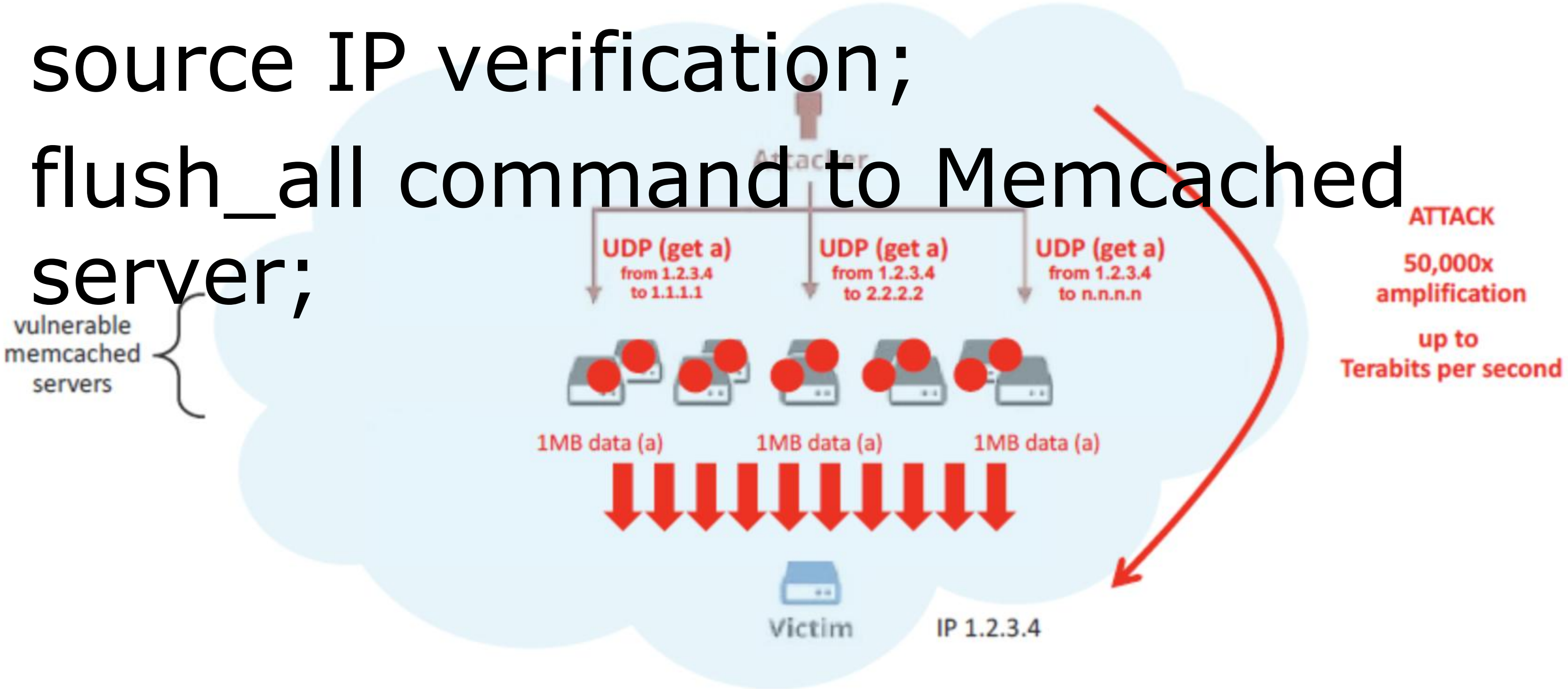
- **Solution**

  disable UDP on Memcached server;

  firewall Memcached server;

  source IP verification;

  flush_all command to Memcached server;

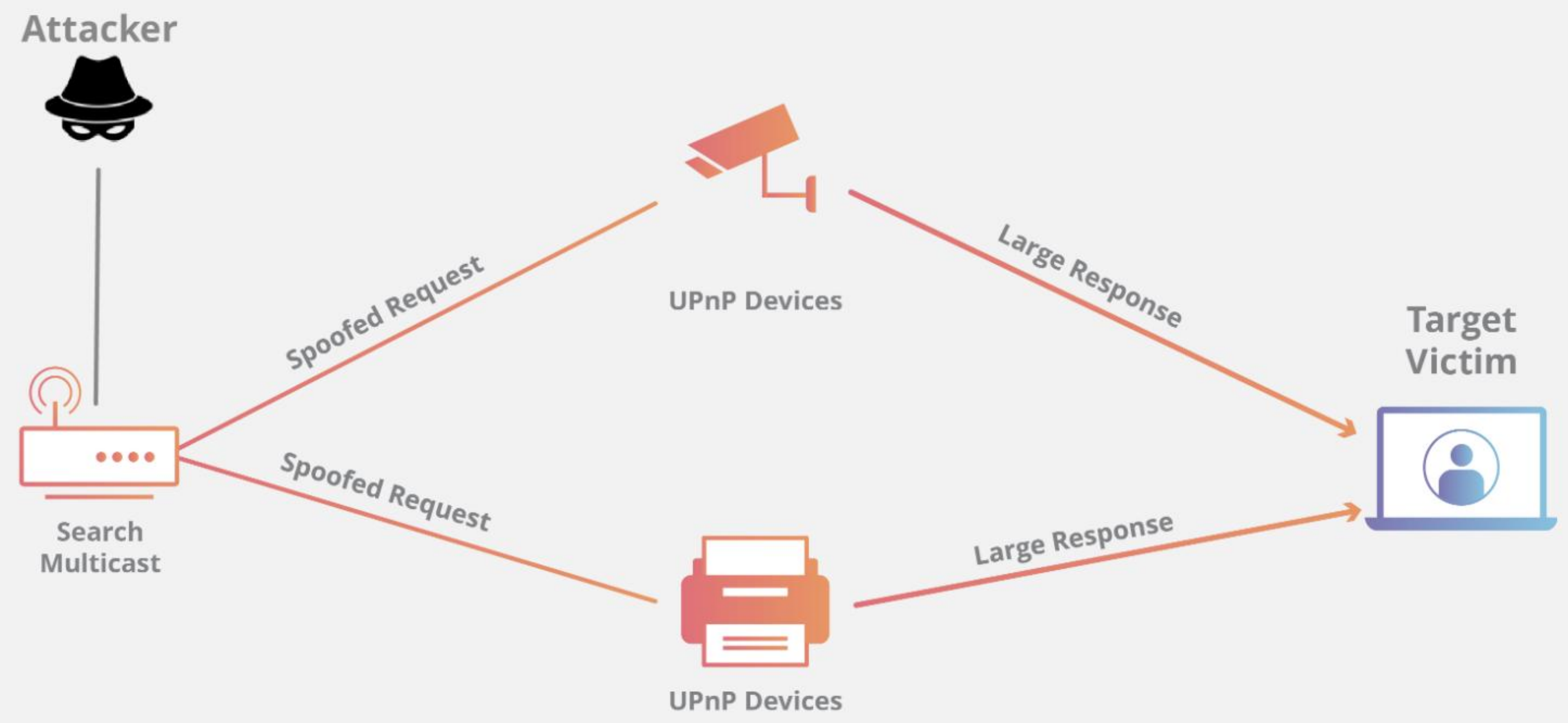


SSDP Attack

- Leverage Simple Service Discovery Protocol (SSDP)
- Exploit Universal Plug and Play (UPnP) networking protocols that trigger UPnP devices to respond with a complete list of all services it has to offer

SSDP Attack

- Exploit SSDP and UpnP



SSDP Attack

- **Attack principle**

  1. the attacker conducts a scan looking for plug-and-play devices that can be utilized as amplification factors;

SSDP Attack

- **Attack principle**

  1. the attacker conducts a scan looking for plug-and-play devices that can be utilized as amplification factors;

  2. as the attacker discovers networked devices, they create a list of all the devices that respond;

SSDP Attack

- **Attack principle**

  1. the attacker conducts a scan looking for plug-and-play devices that can be utilized as amplification factors;

  2. as the attacker discovers networked devices, they create a list of all the devices that respond;

  3. the attacker creates a UDP packet with the spoofed IP address of the targeted victim.

SSDP Attack

- **Attack principle**

  4. the attacker then uses a botnet to send a spoofed discovery packet to each plug-and-play device with a request for as much data as possible by setting certain flags, specifically ssdp:rootdevice or ssdp:all;

SSDP Attack

- **Attack principle**

  5. as a result, each device will send a reply to the targeted victim with an amount of data up to about 30 times larger than the attacker’s request;

SSDP Attack

- **Attack principle**

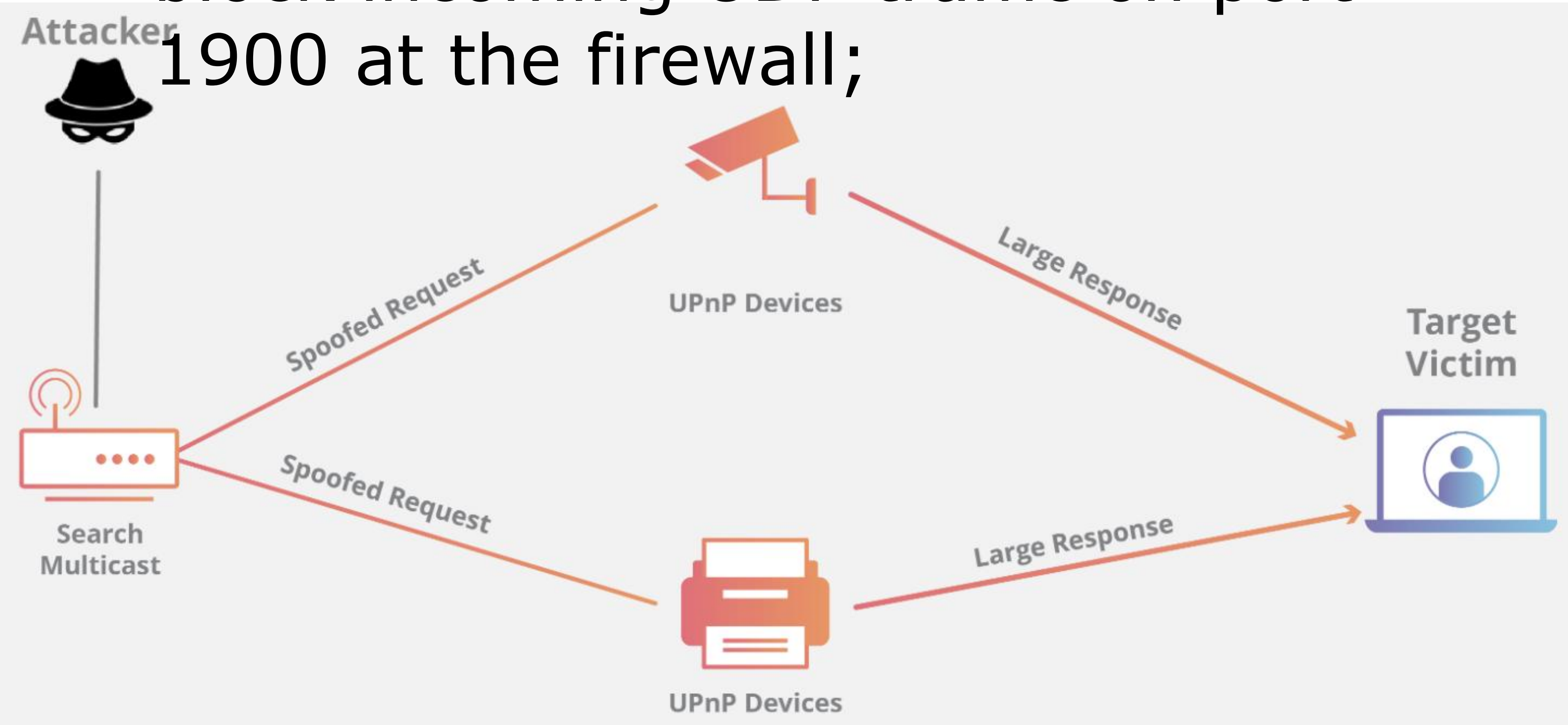
  5. as a result, each device will send a reply to the targeted victim with an amount of data up to about 30 times larger than the attacker’s request;

  6. the target then receives a large volume of traffic from all the devices and becomes overwhelmed, potentially resulting in denial-of-service to legitimate traffic;

SSDP Attack

- **Solution**

  block incoming UDP traffic on port 1900 at the firewall;

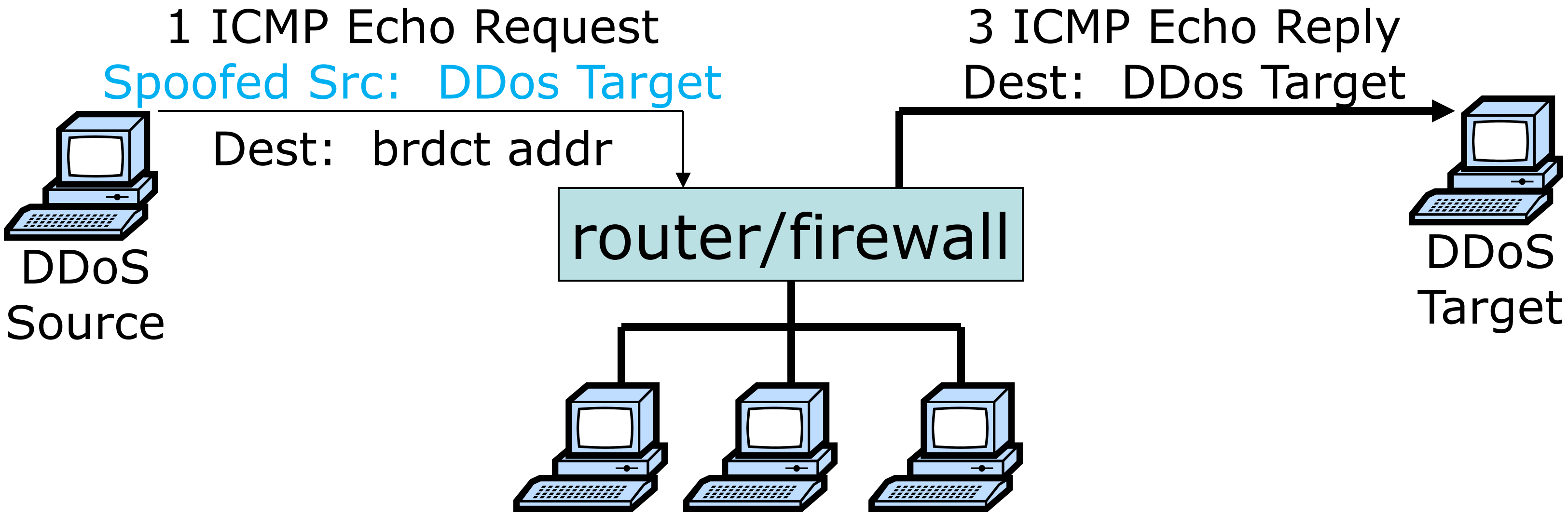


asymmetric DDoS attack

exploit traffic amplifier so far

Smurf Attack

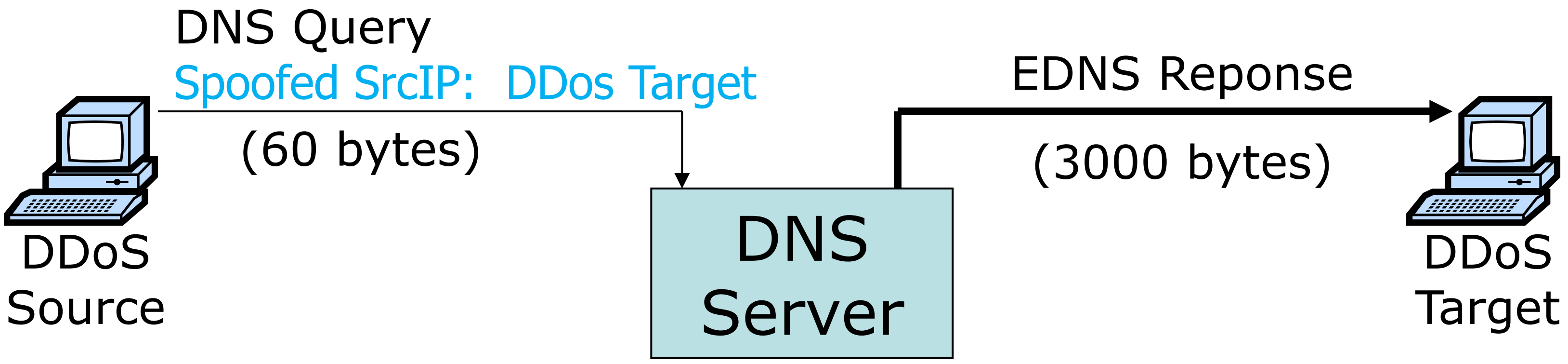
- Attack with an ICMP Echo Request with spoofed source IP address of the targeted server and destination IP address of an IP broadcast address



DNS Amplification Attack

- Attack with an ANY-type DNS query with spoofed source IP address of the targeted server

  EDNS: Extension Mechanisms for DNS sends DNS data in larger UDP packets



DNS Query

Spoofed SrcIP: DDos Target

(60 bytes)

EDNS Reponse

(3000 bytes)

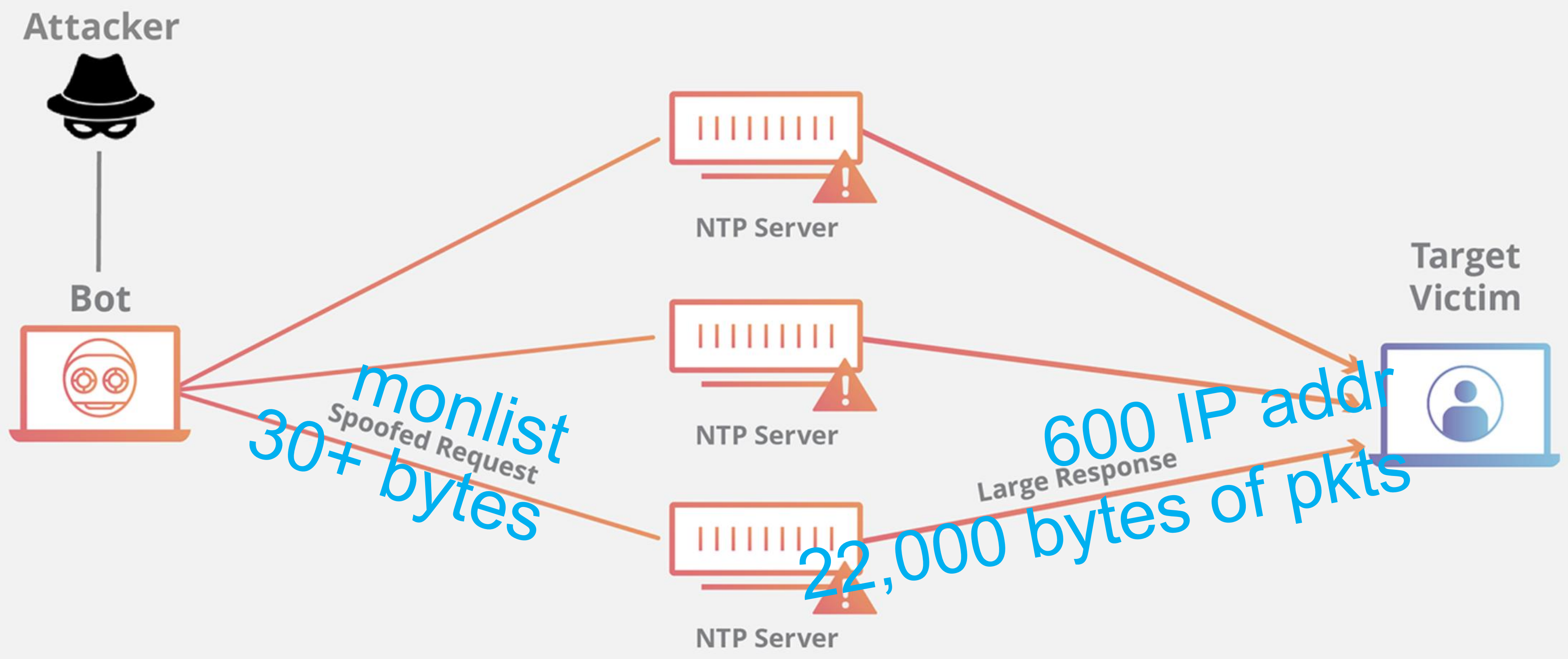
DDoS Source

DNS Server

DDoS Target

NTP Amplification Attack

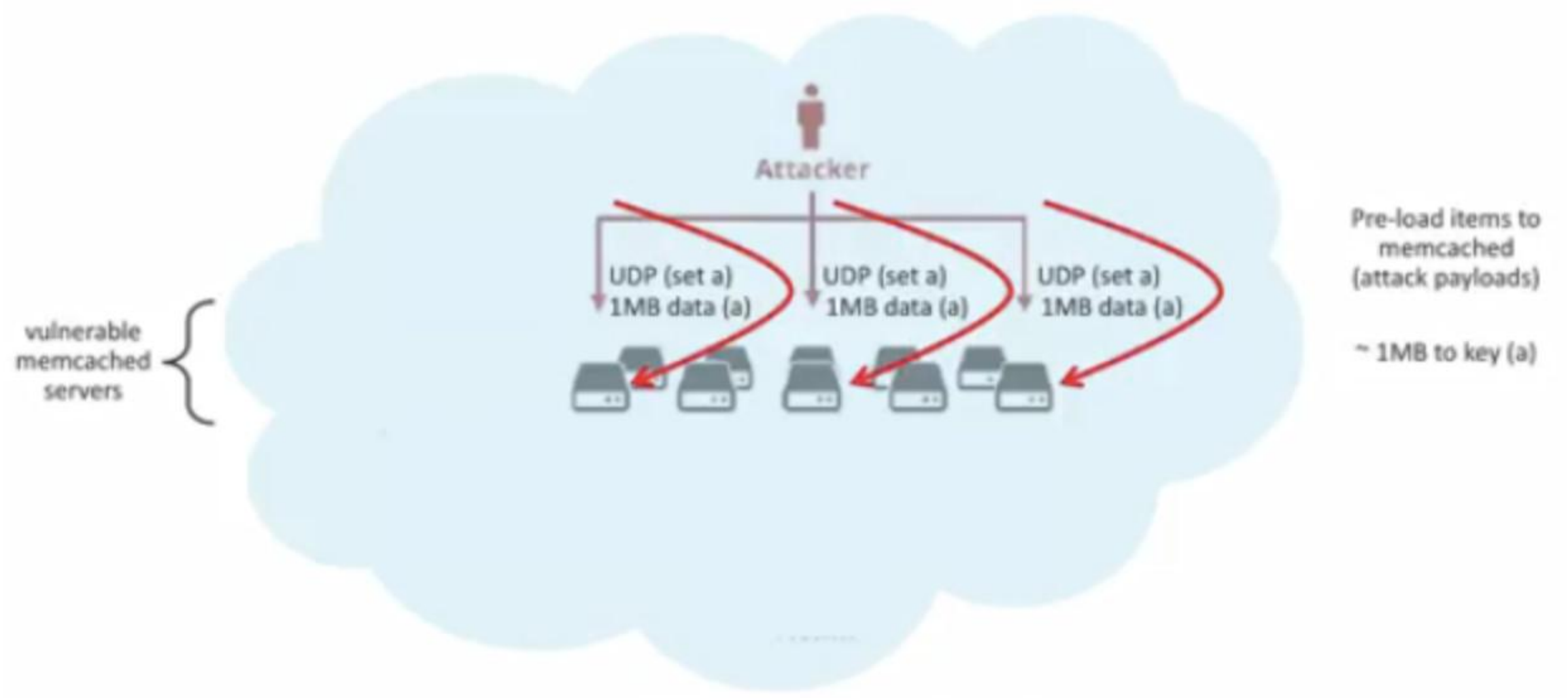
- Amplify the effect of NTP query
- 1 query vs a large response



Memcached Attack

- **Attack principle**

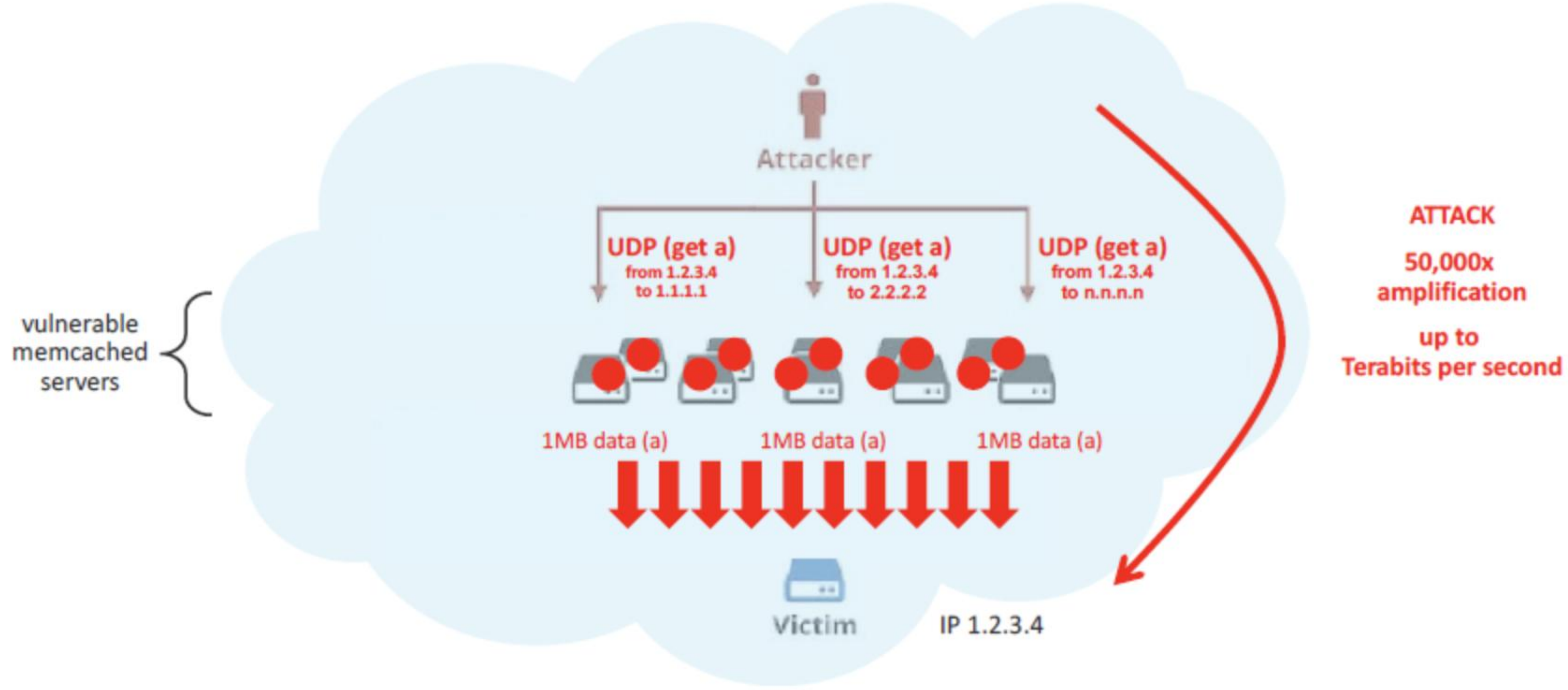
  preload large data to Memcached server;



Memcached Attack

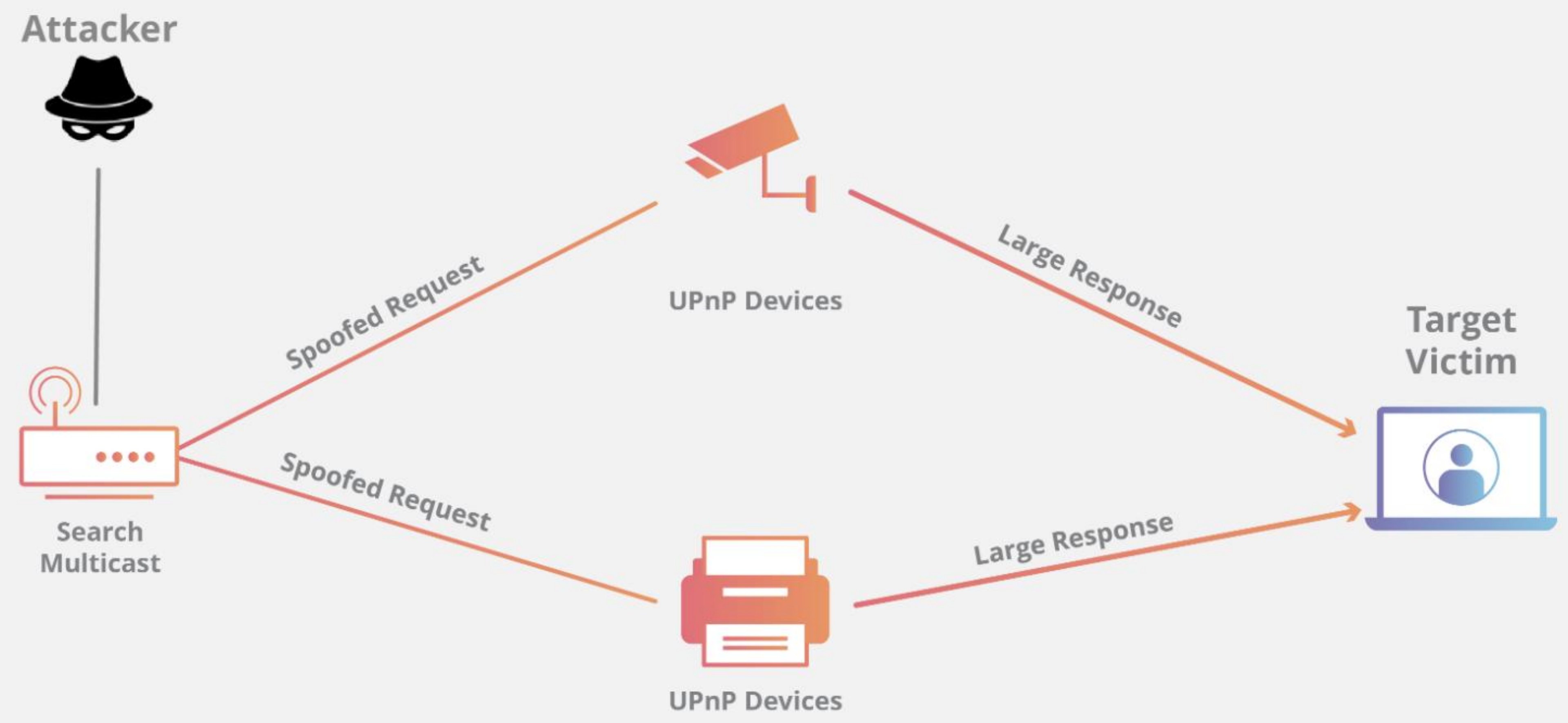
- **Attack principle**

  spoof request to preloaded data from target;



SSDP Attack

- Exploit SSDP and UpnP

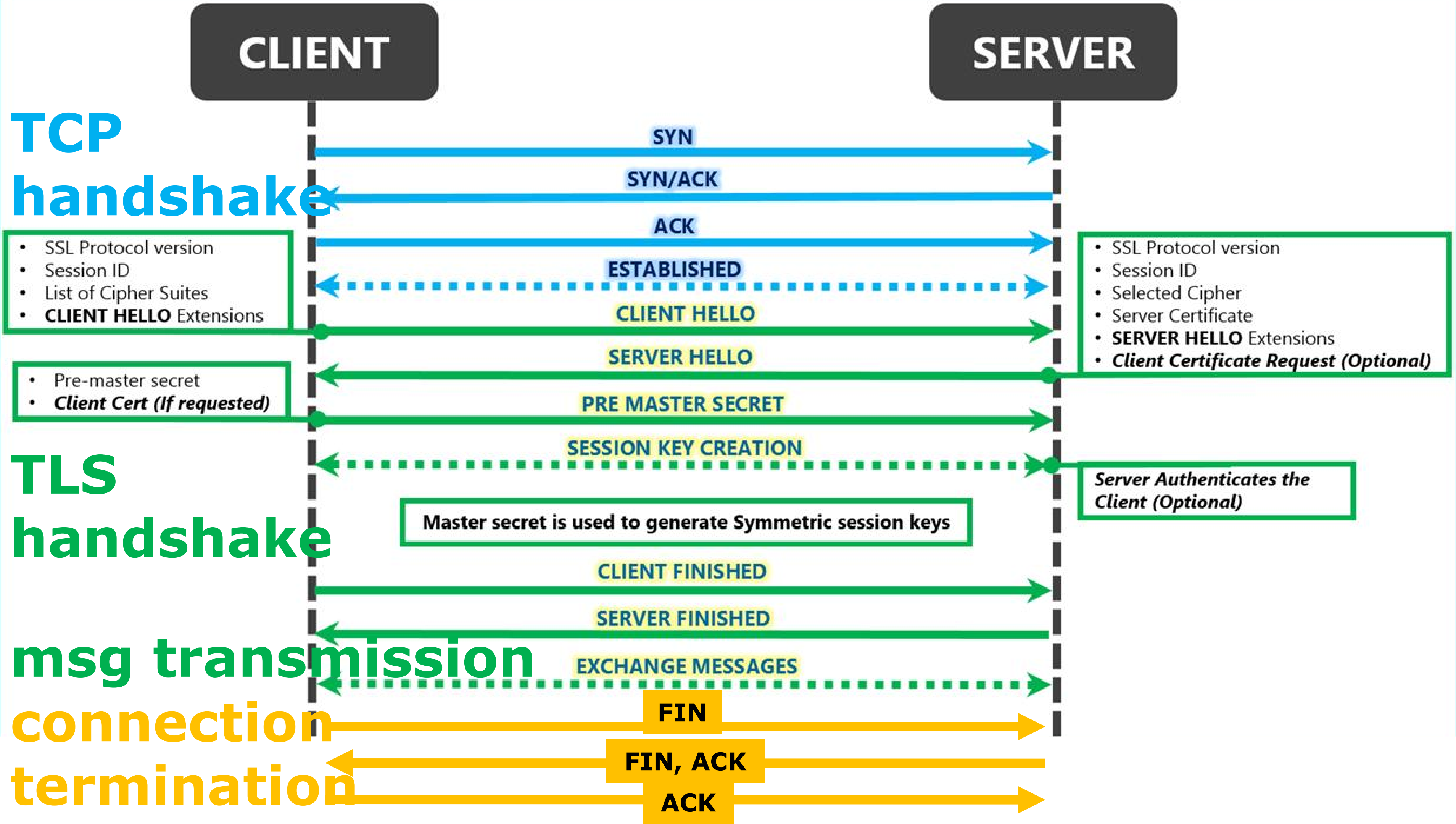


asymmetric DDoS attack

computation asymmetry:

server costs more computation resources than attacker for a service request

Secure Connection



CLIENT

SERVER

TCP handshake

SYN

SYN/ACK

ACK

ESTABLISHED

- SSL Protocol version
- Session ID
- List of Cipher Suites
- **CLIENT HELLO** Extensions

CLIENT HELLO

- SSL Protocol version
- Session ID
- Selected Cipher
- Server Certificate
- **SERVER HELLO** Extensions
- ***Client Certificate Request (Optional)***

SERVER HELLO

- Pre-master secret
- ***Client Cert (If requested)***

PRE MASTER SECRET

SESSION KEY CREATION

Server Authenticates the Client (Optional)

TLS handshake

Master secret is used to generate Symmetric session keys

CLIENT FINISHED

SERVER FINISHED

msg transmission

EXCHANGE MESSAGES

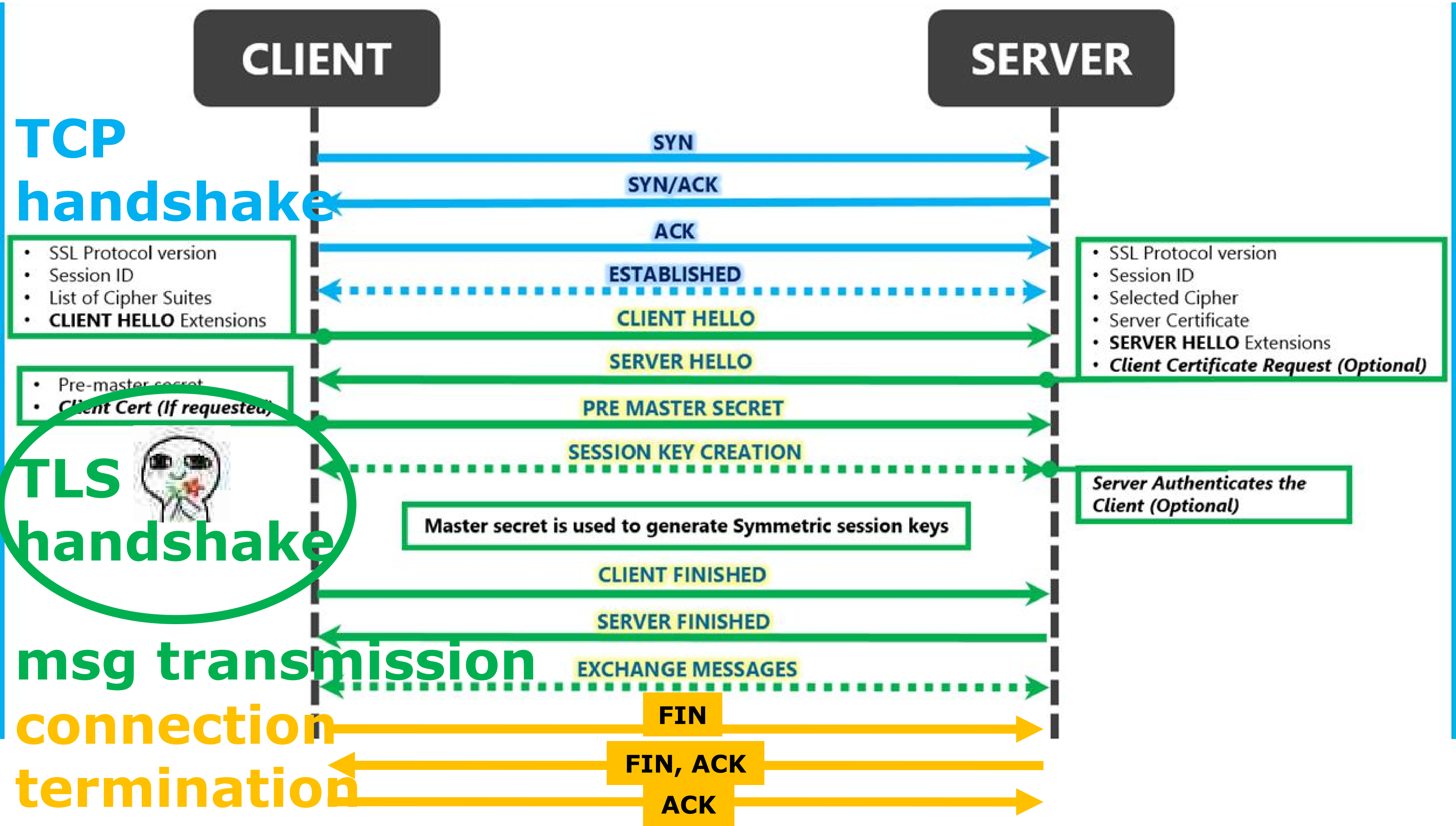
connection termination

FIN

FIN, ACK

ACK

SSL/TLS Handshake



CLIENT

SERVER

TCP handshake

SYN

SYN/ACK

ACK

ESTABLISHED

- SSL Protocol version
- Session ID
- List of Cipher Suites
- **CLIENT HELLO** Extensions

CLIENT HELLO

SERVER HELLO

- SSL Protocol version
- Session ID
- Selected Cipher
- Server Certificate
- **SERVER HELLO** Extensions
- ***Client Certificate Request (Optional)***

- Pre-master secret
- ***Client Cert (If requested)***

PRE MASTER SECRET

TLS handshake

SESSION KEY CREATION

Server Authenticates the Client (Optional)

Master secret is used to generate Symmetric session keys

CLIENT FINISHED

SERVER FINISHED

msg transmission

EXCHANGE MESSAGES

connection termination

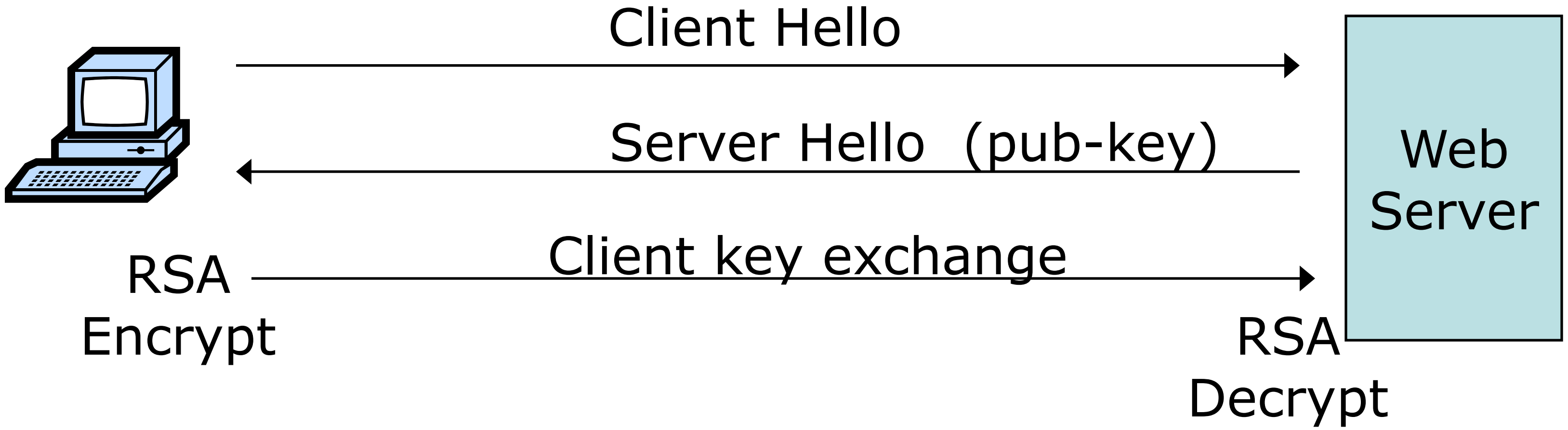
FIN

FIN, ACK

ACK

SSL/TLS Flood

- Exploit SSL/TLS handshake requests to drain server resources



Client Hello

Server Hello (pub-key)

Client key exchange

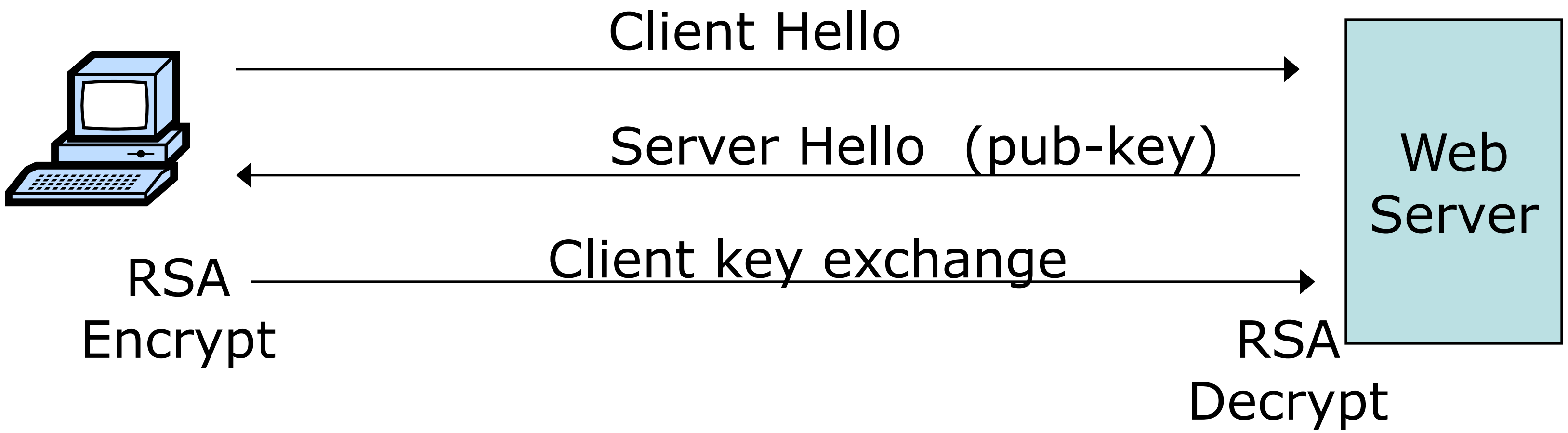
RSA Encrypt

Web Server

RSA Decrypt

SSL/TLS Flood

- Exploit SSL/TLS handshake requests to drain server resources



Client Hello

Server Hello (pub-key)

Client key exchange

RSA Encrypt

Web Server

RSA Decrypt

- RSA-enc speed ≈ 10x RSA-dec speed
- Single machine can bring down ten web servers

HTTP Flood

- Command attackers to:

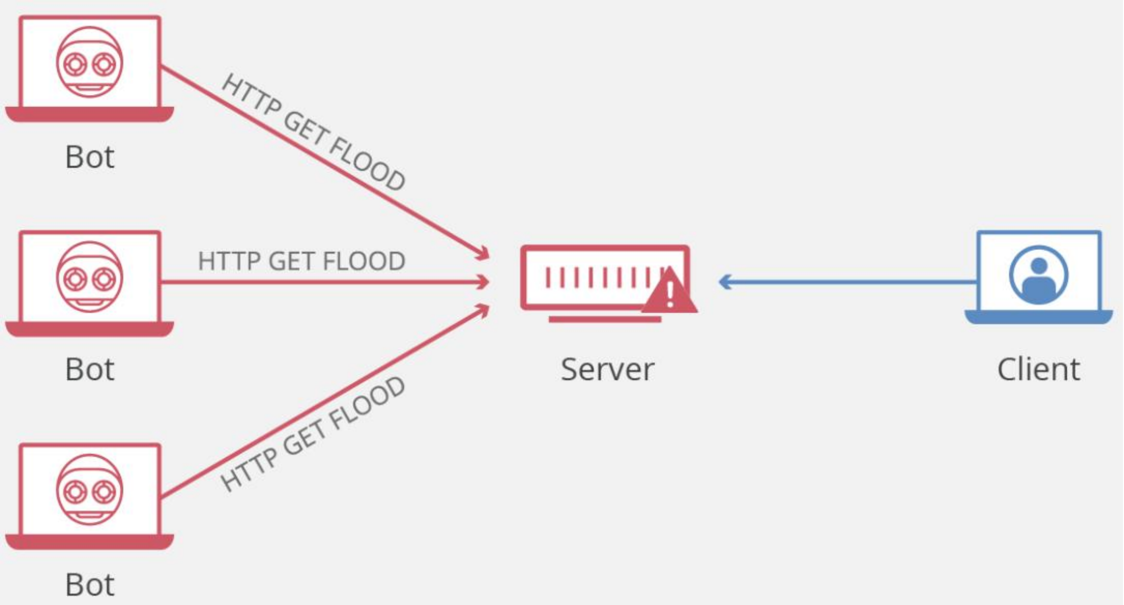
  Complete real TCP connection

  Complete TLS Handshake

  GET large image or other content

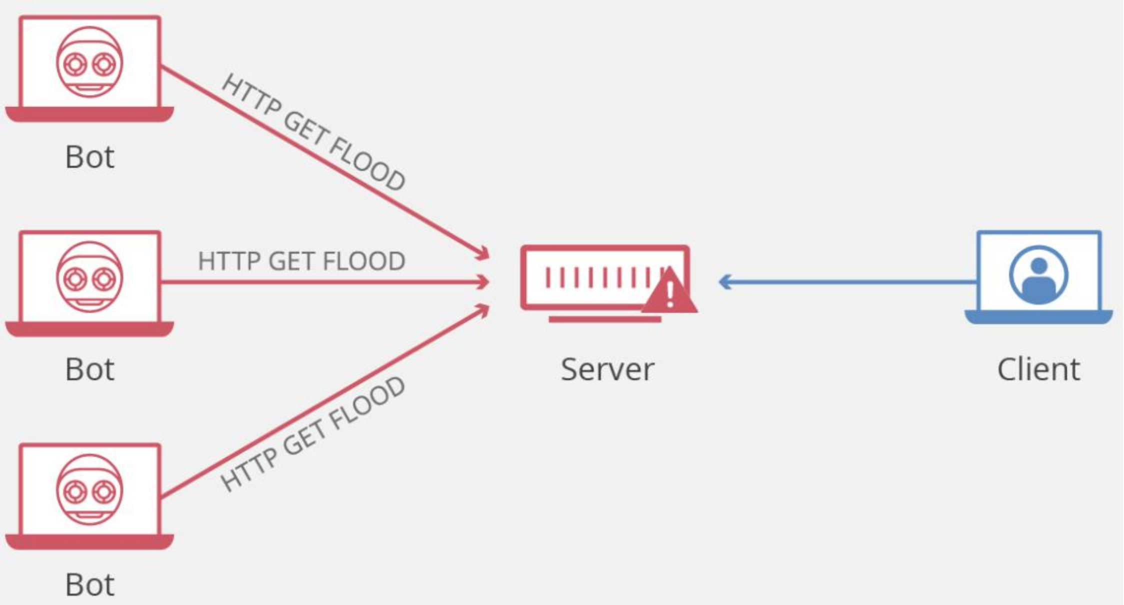
HTTP Flood

- Command attackers to:
  Complete real TCP connection
  Complete TLS Handshake
  GET large image or other content



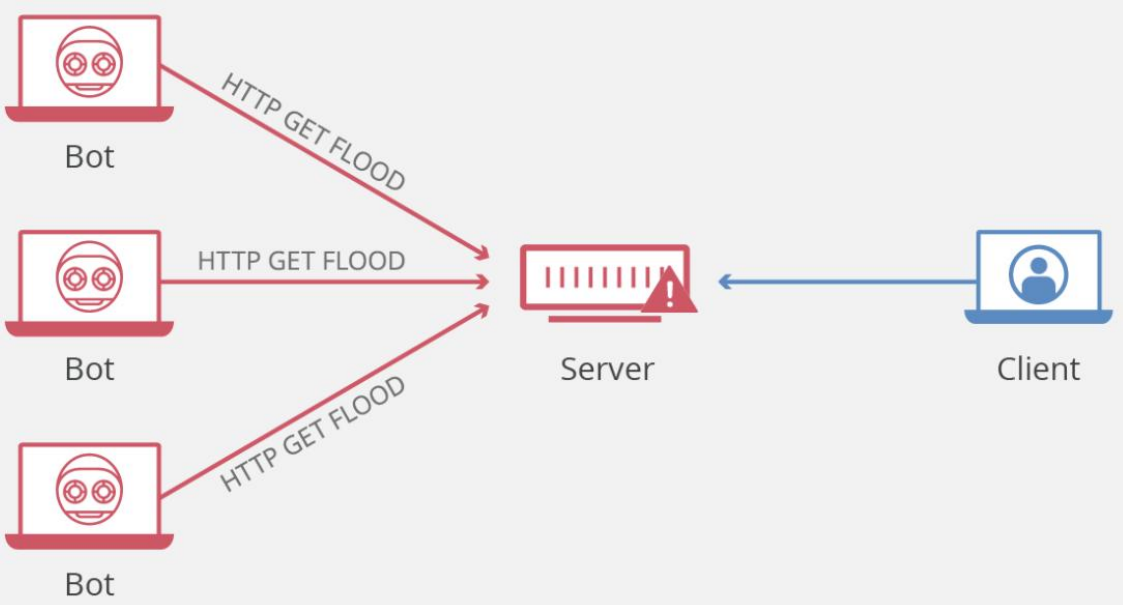
HTTP Flood

- Command attackers to:
  Complete real TCP connection
  Complete TLS Handshake
  GET large image or other content



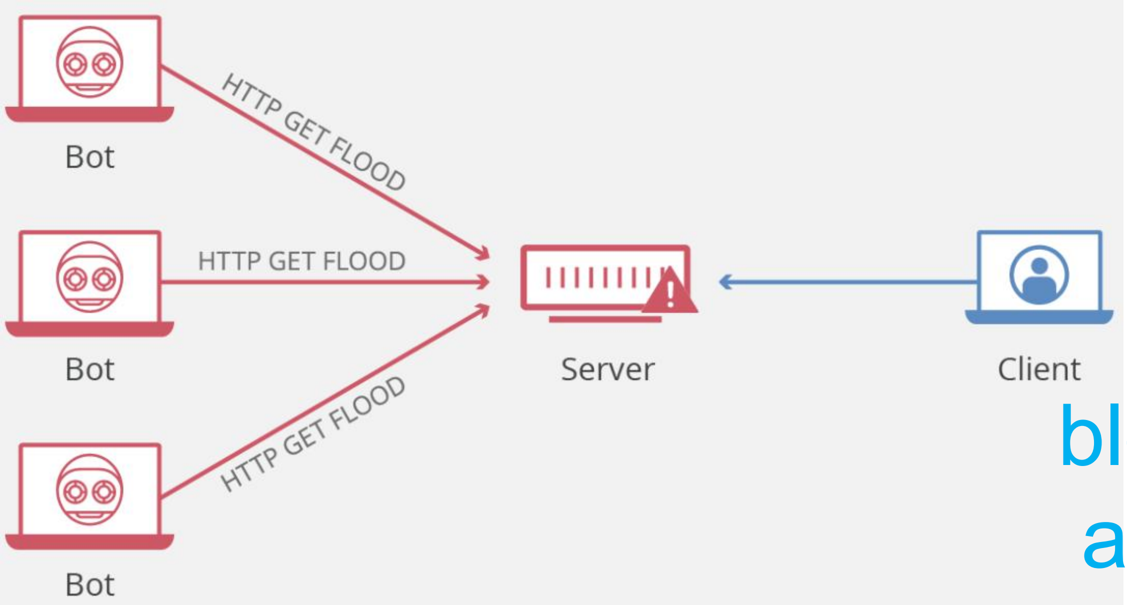
HTTP Flood

- Command attackers to:
  Complete real TCP connection
  Complete TLS Handshake
  POST large image or other content



HTTP Flood

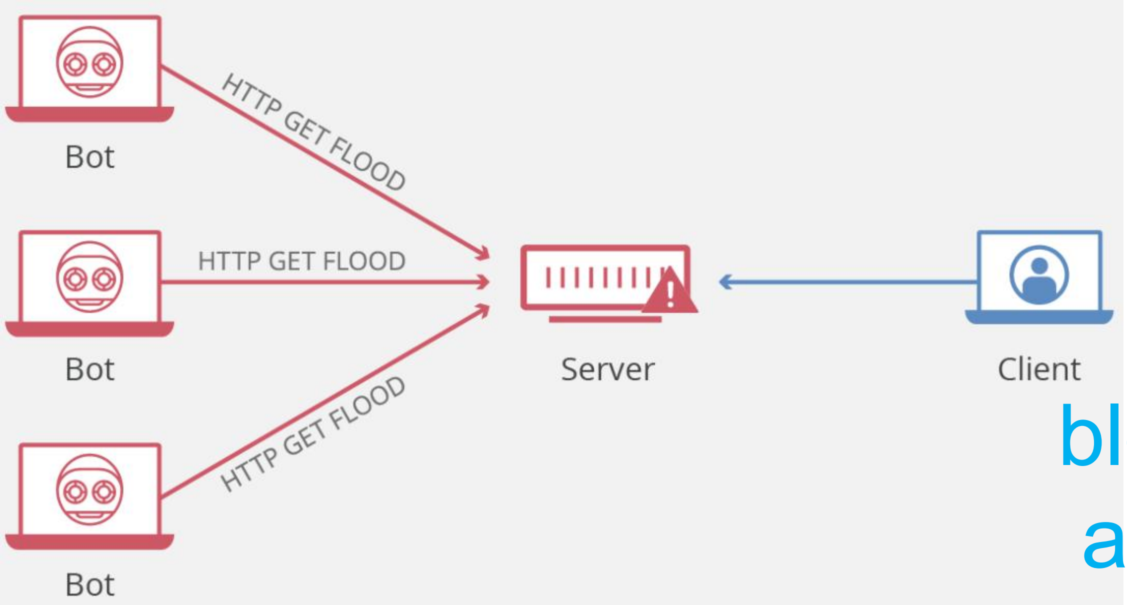
- Command attackers to:
  Complete real TCP connection
  Complete TLS Handshake
  GET/POST large image or other content



Solution:
block or rate limit
attacking source

HTTP Flood

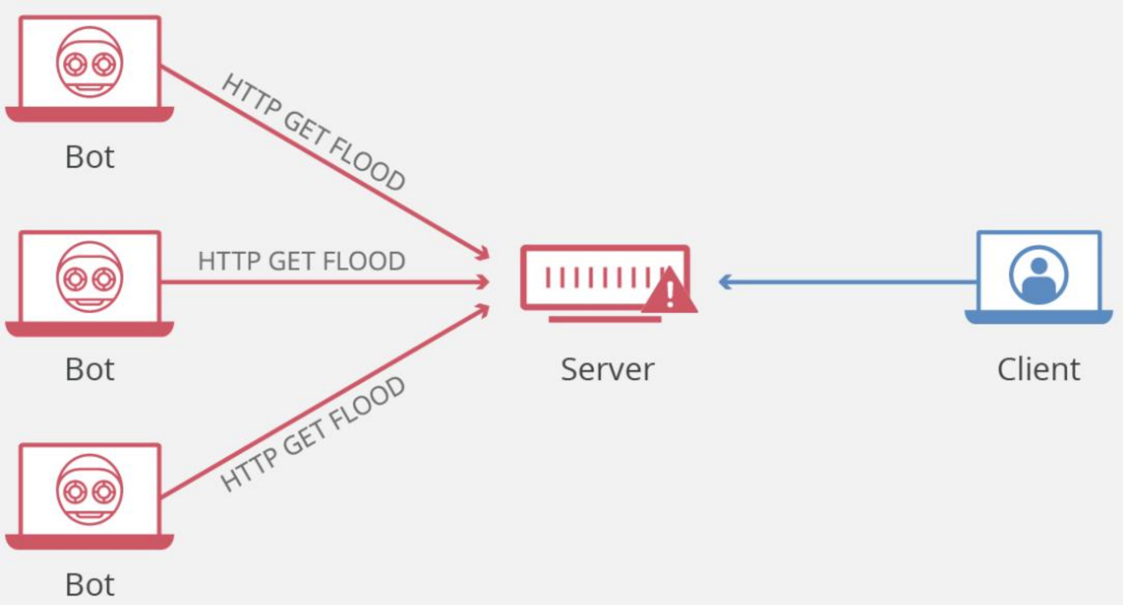
- Command attackers to:
  Complete real TCP connection
  Complete TLS Handshake
  GET/POST large image or other content



Solution:
block or rate limit
attacking source

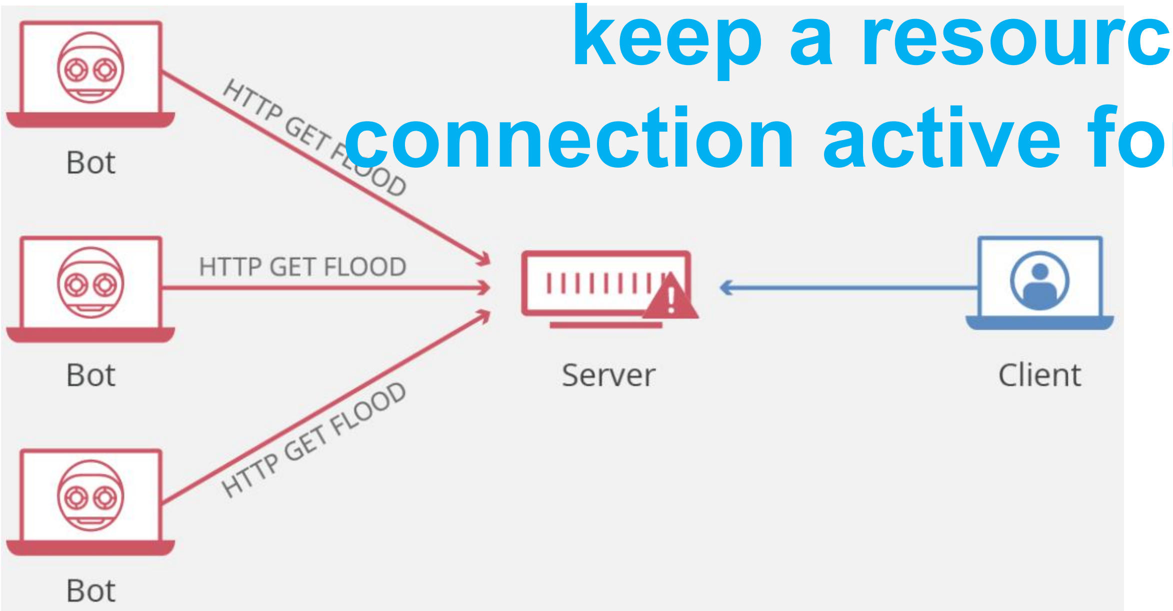
Fragmented HTTP Flood

- Establish a valid HTTP connection
- Split HTTP packets into tiny fragments
- Send fragments to the target as slowly as it allows before it times out



Fragmented HTTP Flood

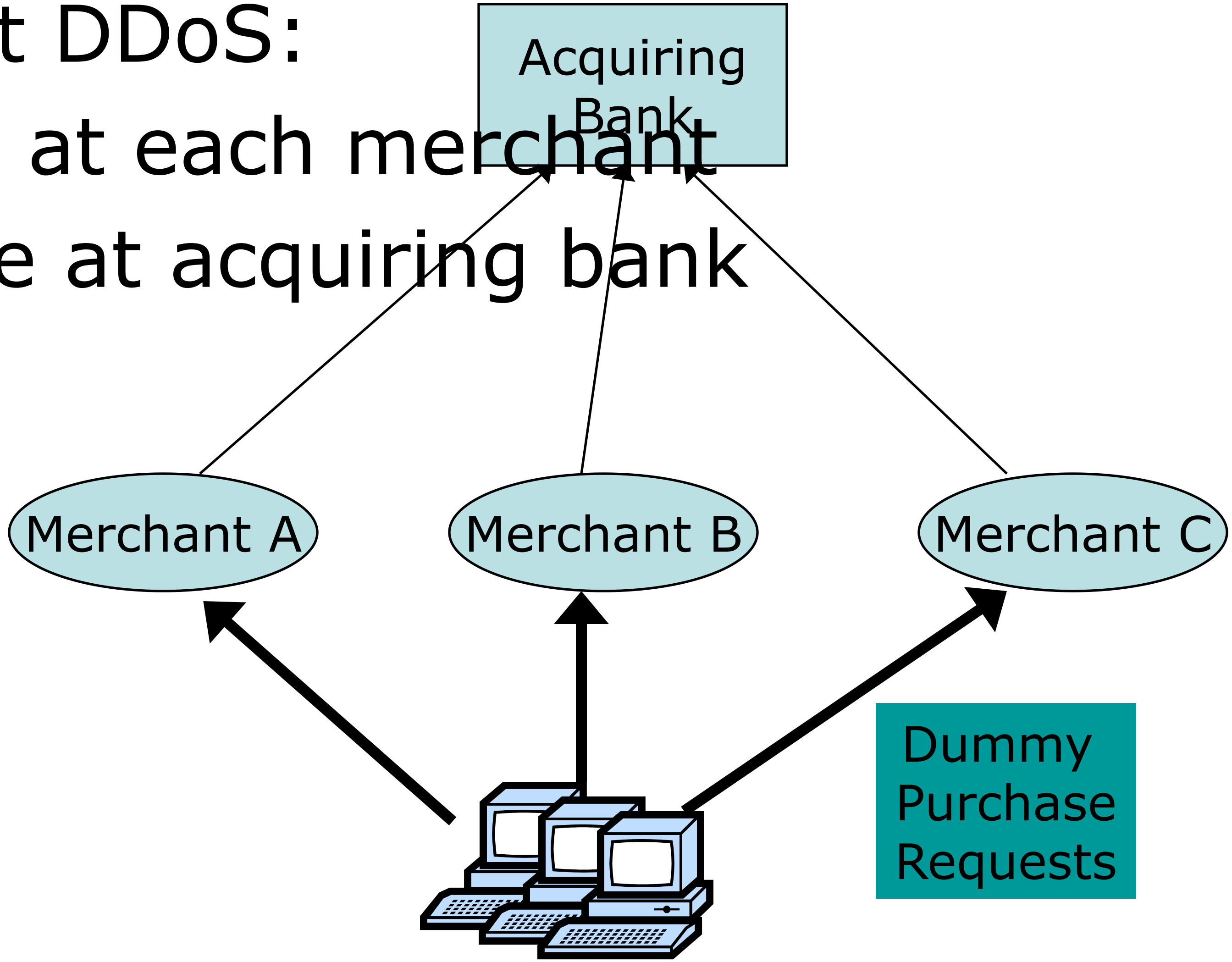
- Establish a valid HTTP connection
- Split HTTP packets into tiny fragments
- Send fragments to the target as slowly as it allows before it times out



keep a resource-consuming connection active for a long time;

Payment DDoS

- Payment DDoS:
  low rate at each merchant
  high rate at acquiring bank



Acquiring Bank

Merchant A

Merchant B

Merchant C

Dummy Purchase Requests

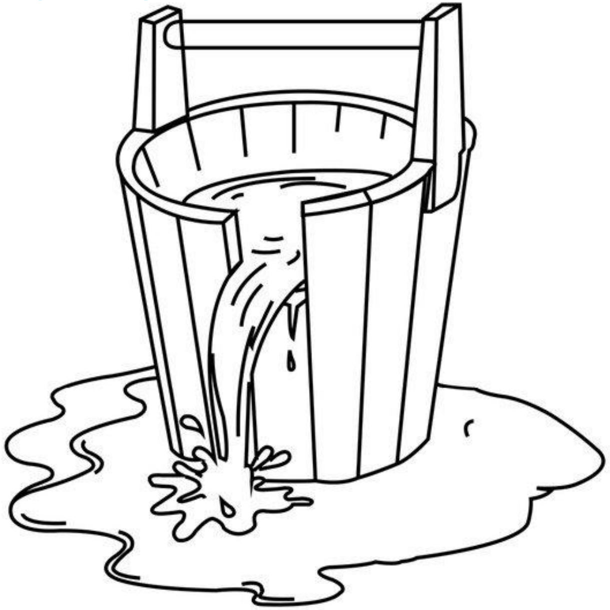
asymmetric DDoS attack

bring down the entire server so far

asymmetric DDoS attack

bring down the entire server so far

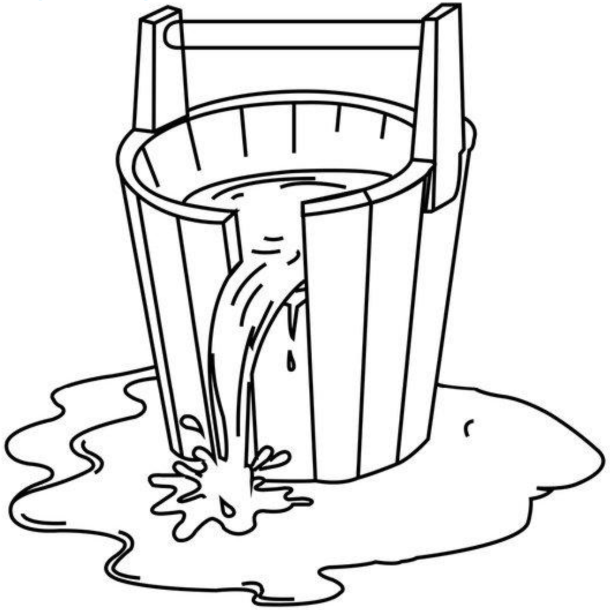
weakest link?!



asymmetric DDoS attack

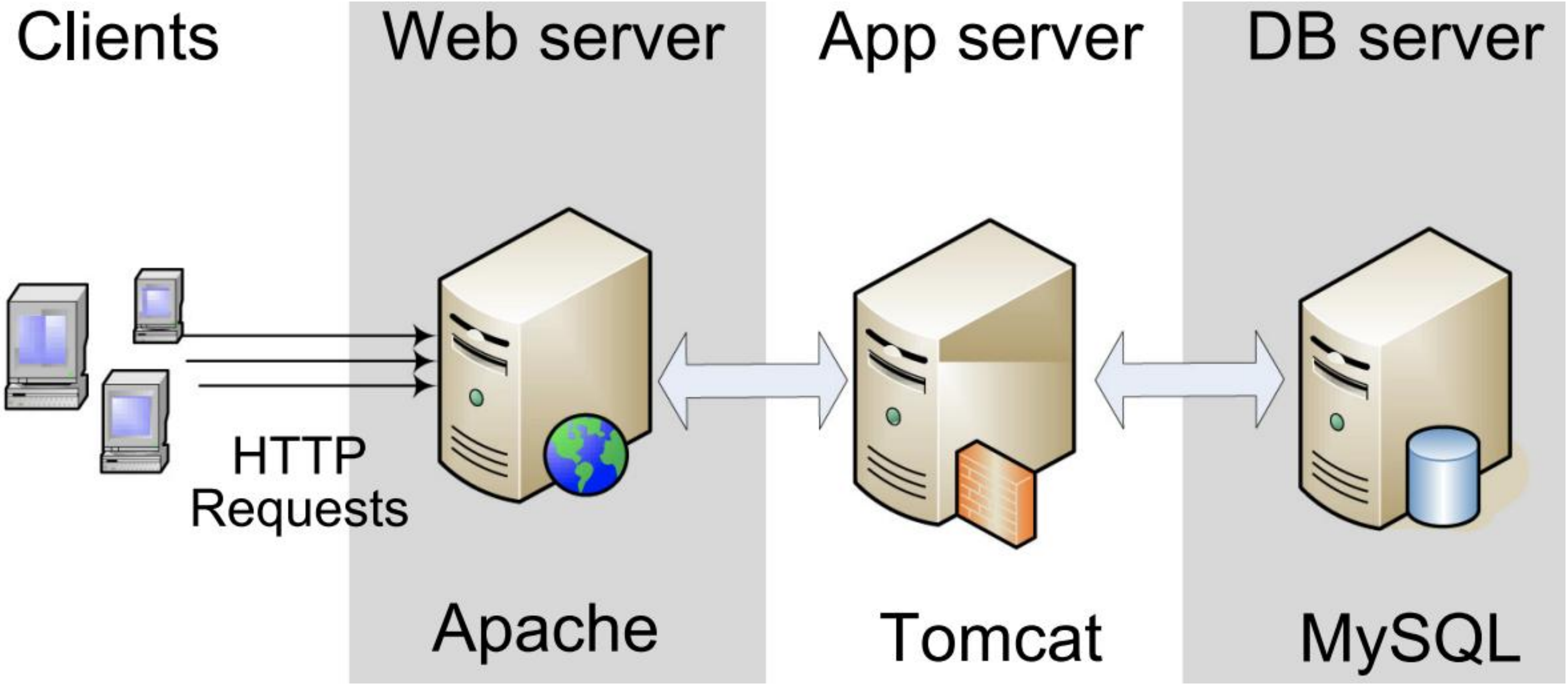
bring down the entire server so far

weakest link from inside



Tail Attack

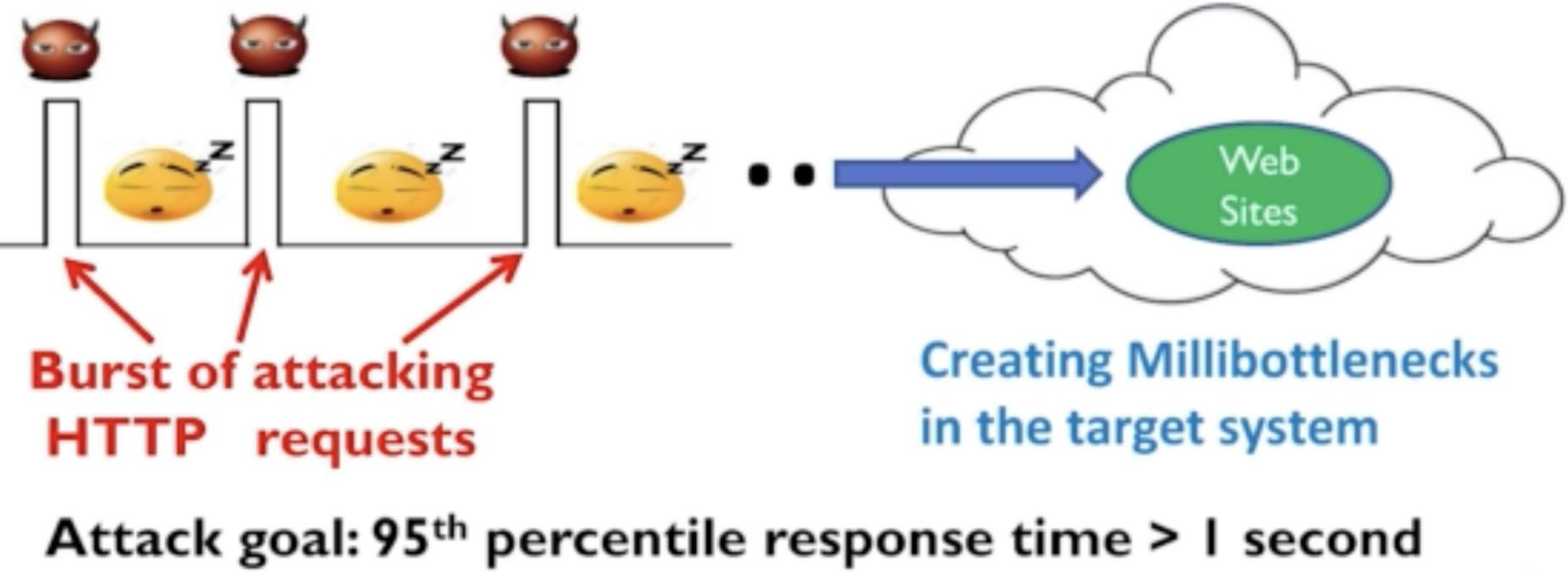
- Tail attacks on n-tier web applications
- Identify weakest link across tiers



http://csc.lsu.edu/~qywang/papers/TOIT-Wang-2019.pdf

Tail Attack

- Tail attacks on n-tier web applications
- Saturate weakest link w/ low-rate traffic



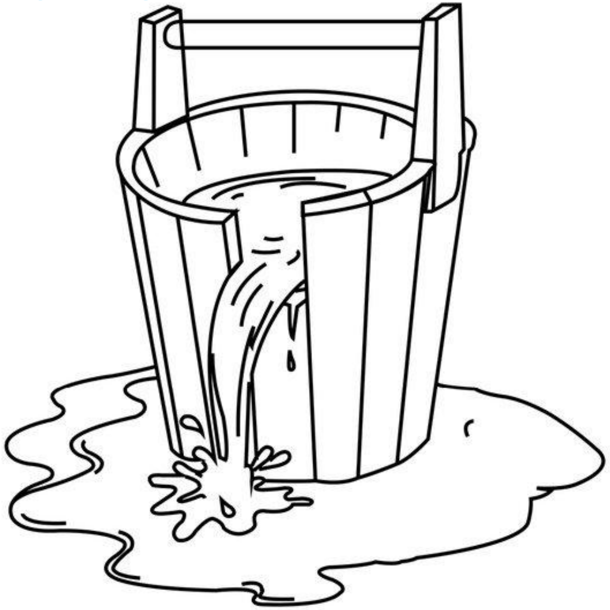
Attack goal: 95th percentile response time > 1 second

https://dl.acm.org/doi/10.1145/3133956.3133968

asymmetric DDoS attack

bring down the entire server so far

weakest link from outside



SDN CrossPath Attack

- Disrupt SDN control channel via shared links
- Do not directly attack SDN controller
- Instead, block control messages with attacking traffic

https://www.usenix.org/sites/default/files/conference/protected-files/sec19_slides_cao.pdf

SDN CrossPath Attack

- SDN: Software-Defined Networking

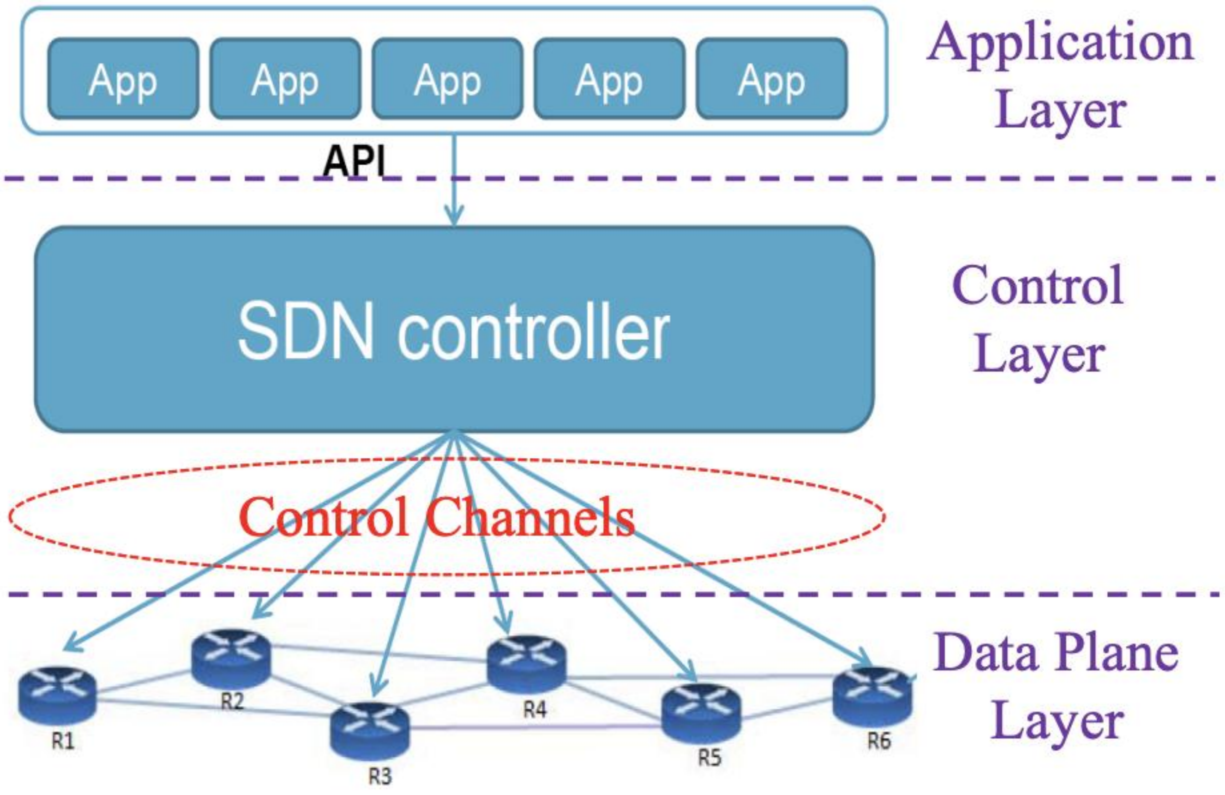
  separate control and data planes

  take centralized network control

  enable network programmability

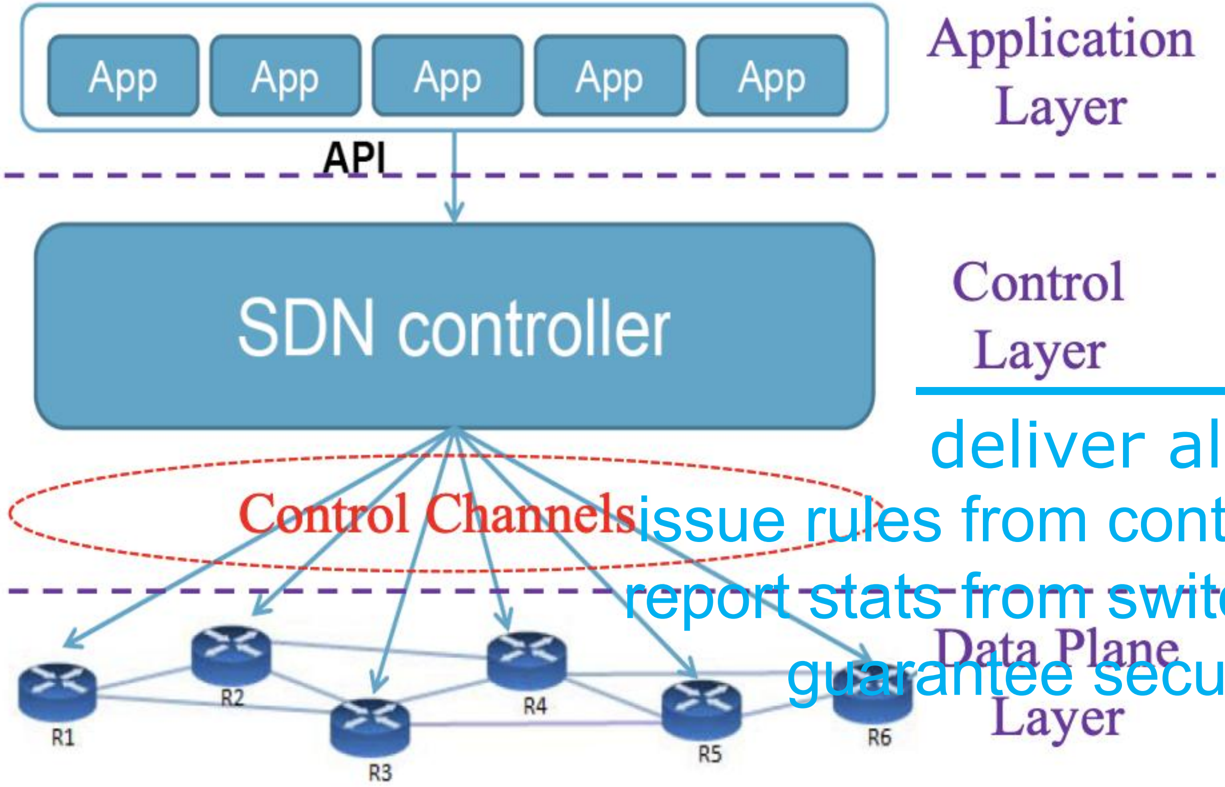
SDN CrossPath Attack

- Three-layer architecture



SDN CrossPath Attack

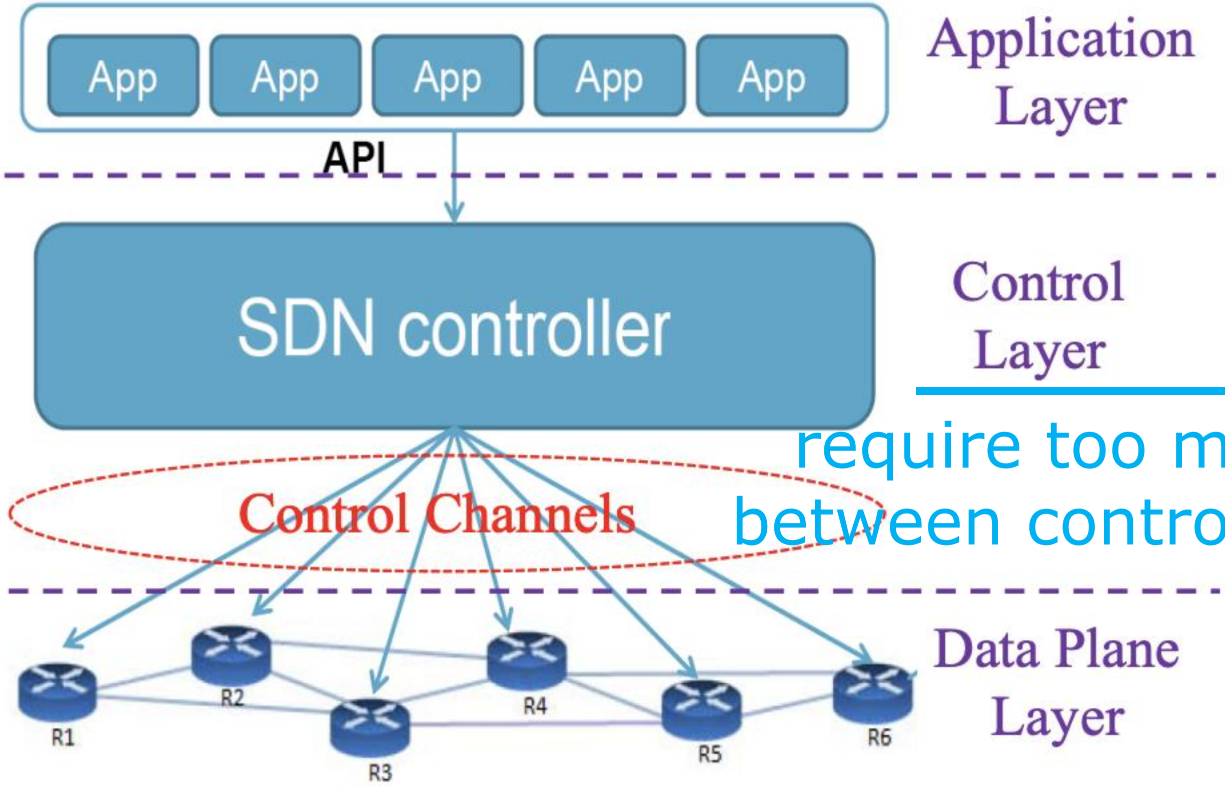
- Three-layer architecture



deliver all control traffic;
issue rules from controller to switches;
report stats from switches to controller;
guarantee security and reliability;

SDN CrossPath Attack

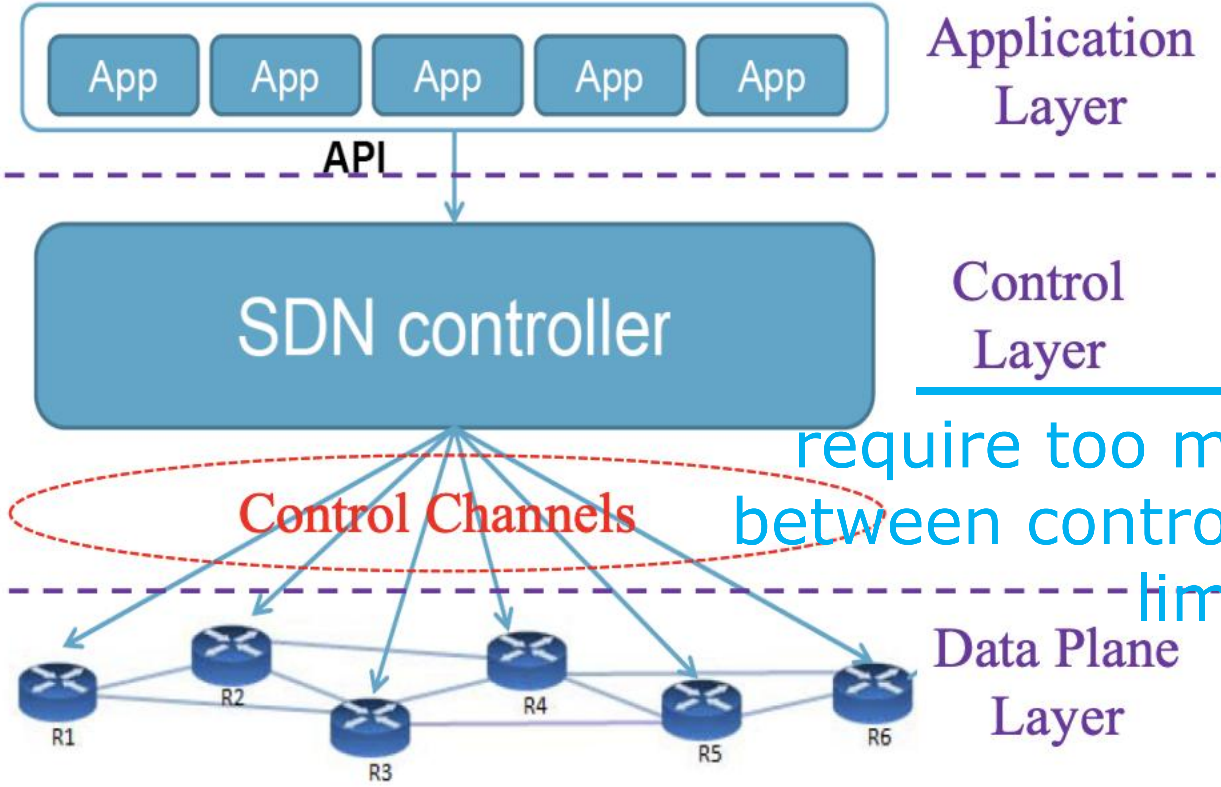
- Three-layer architecture



require too many direct links between controller and switch!

SDN CrossPath Attack

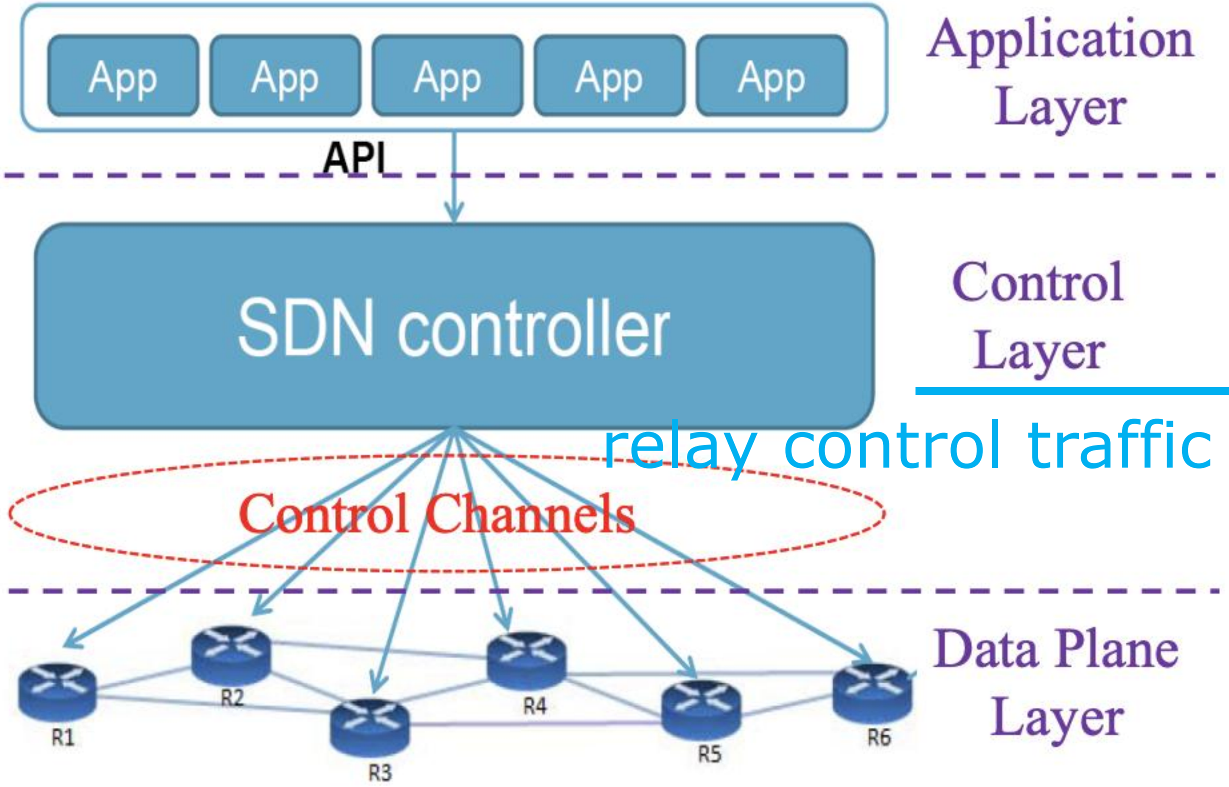
- Three-layer architecture



require too many direct links between controller and switch! limited scalability;

SDN CrossPath Attack

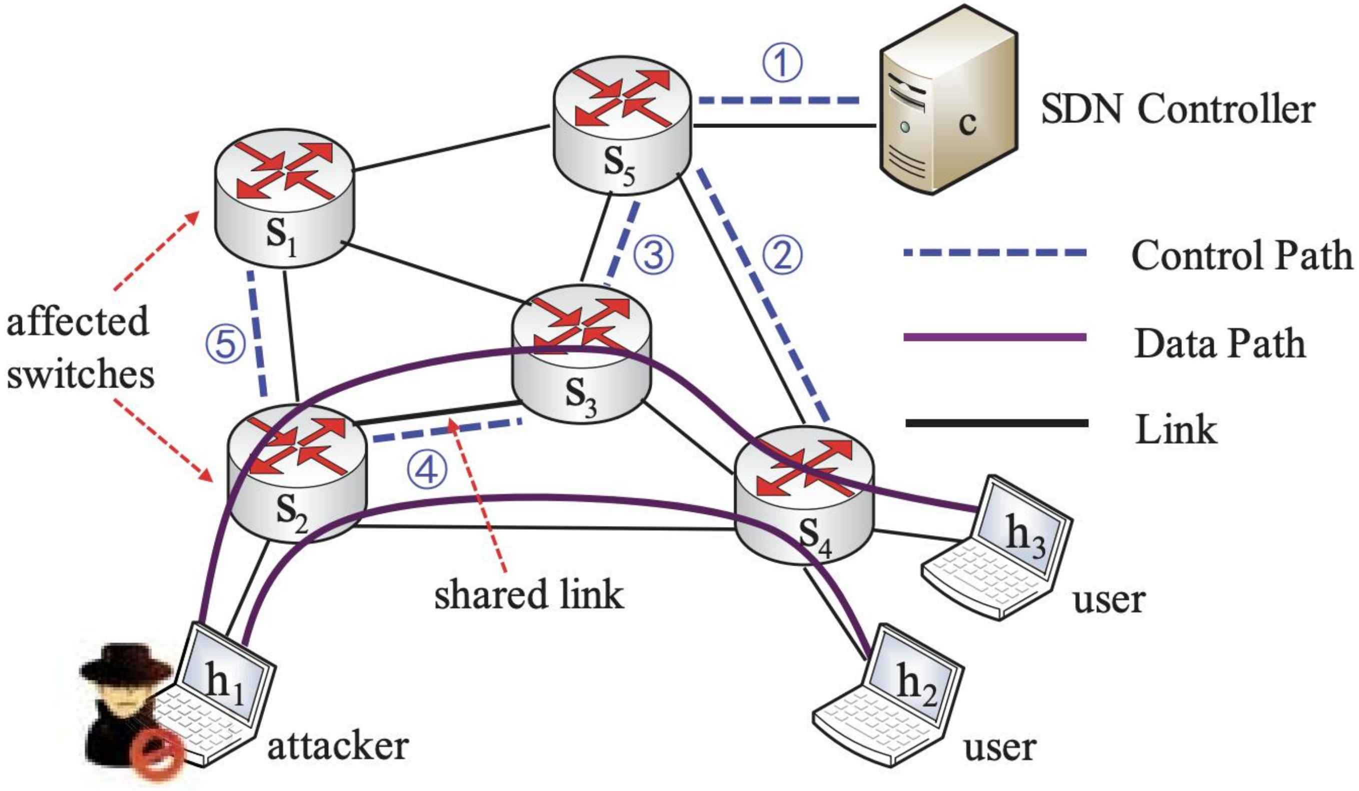
- Introduce shared links



relay control traffic via data paths;

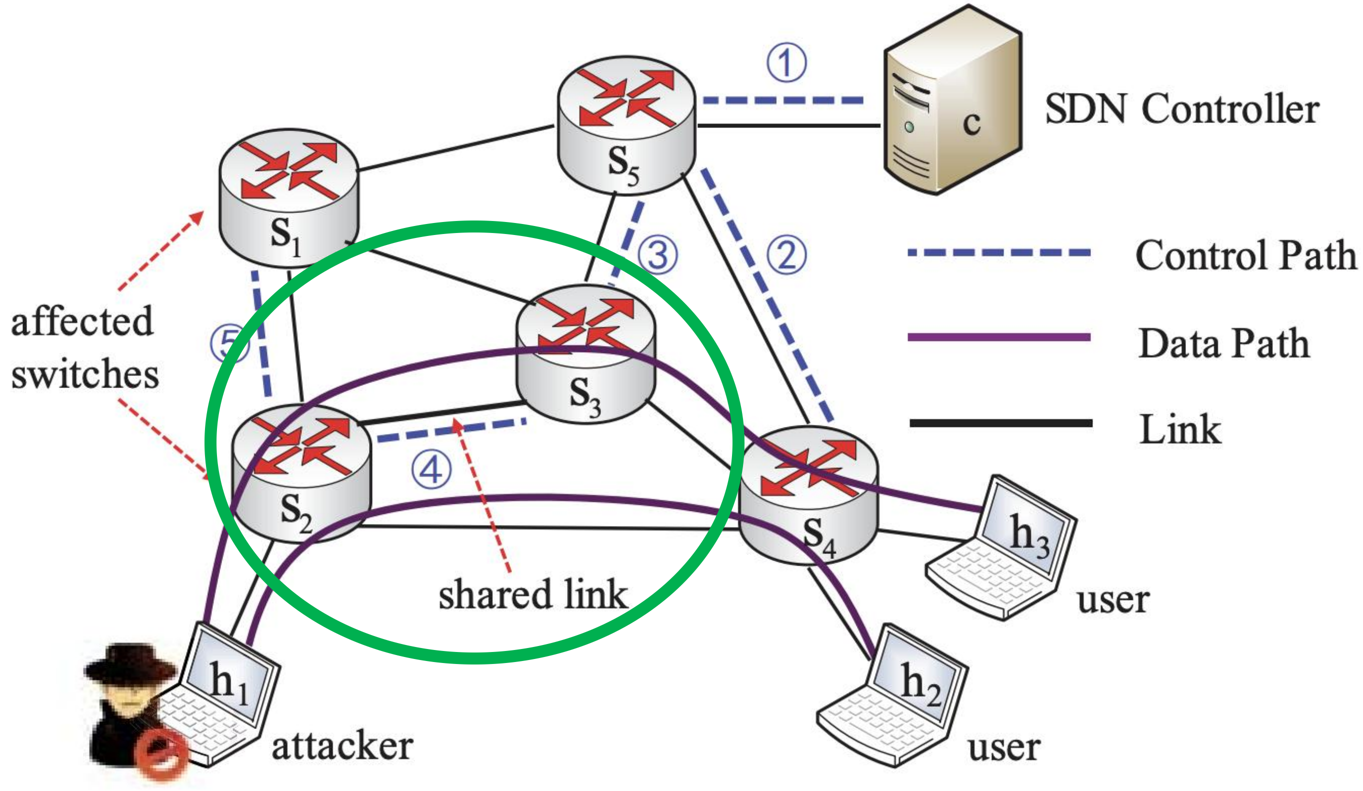
SDN CrossPath Attack

- Introduce shared links



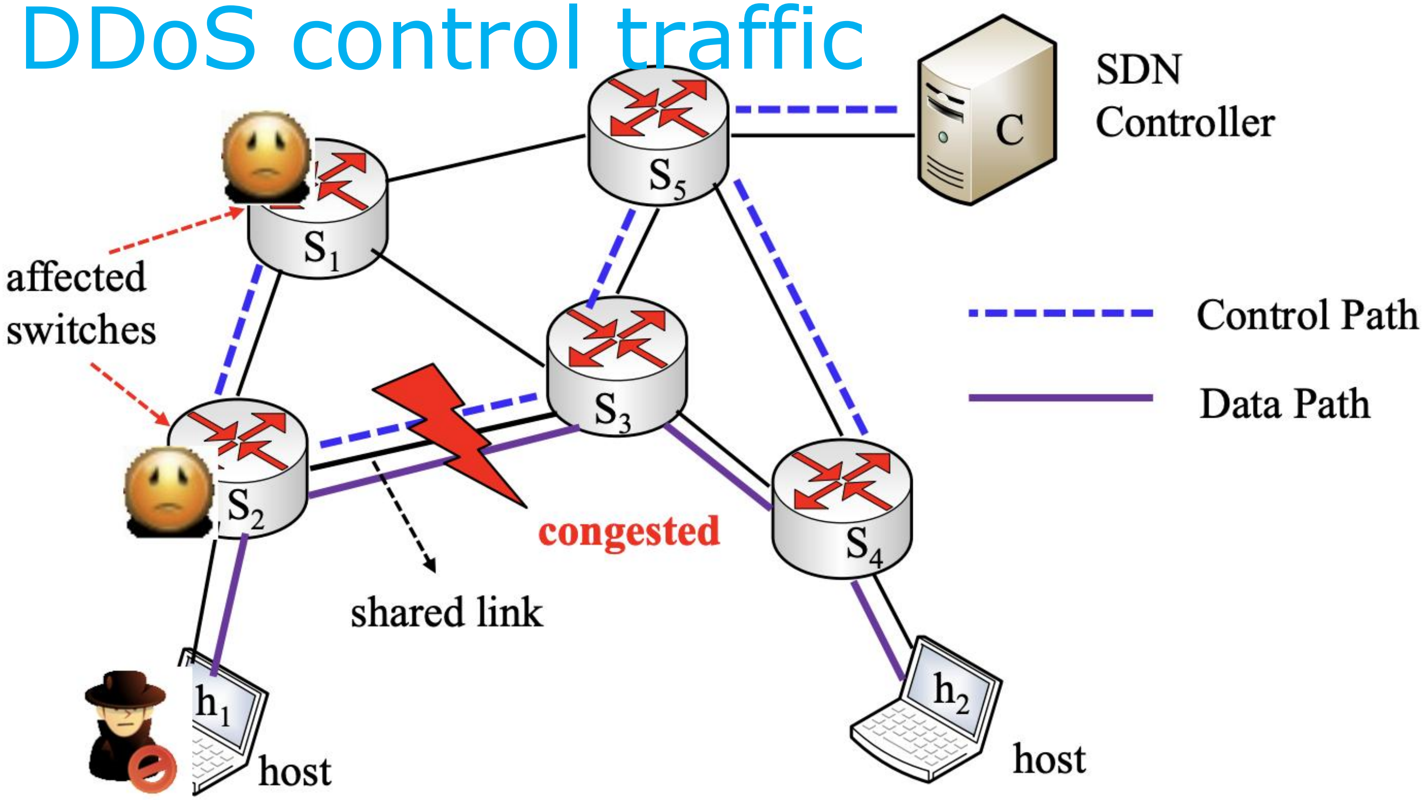
SDN CrossPath Attack

- Introduce shared links: control&data



SDN CrossPath Attack

- Send data traffic to congest shared links
- DDoS control traffic



DDoS attacks so far

DDoS defenses

DDoS defenses

make server harder to be attacked

DDoS defenses

make server harder to be attacked:

enrich server with more resources;

DDoS defenses

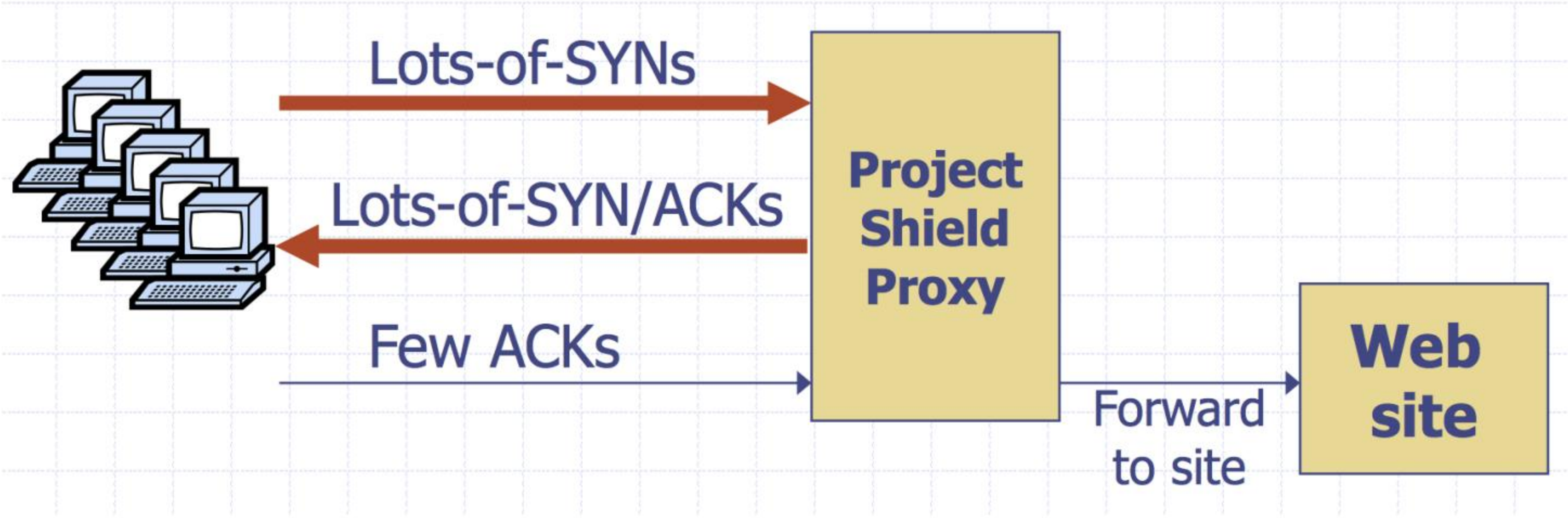
make server harder to be attacked:

enrich server with more resources;

leverage the sources of others;

Google Project Shield

- Use Google bandwidth to shield vulnerable websites



DDoS defenses

make server harder to be attacked:

detect and filter attack traffic

DDoS defenses

make server harder to be attacked:

detect and filter attack traffic

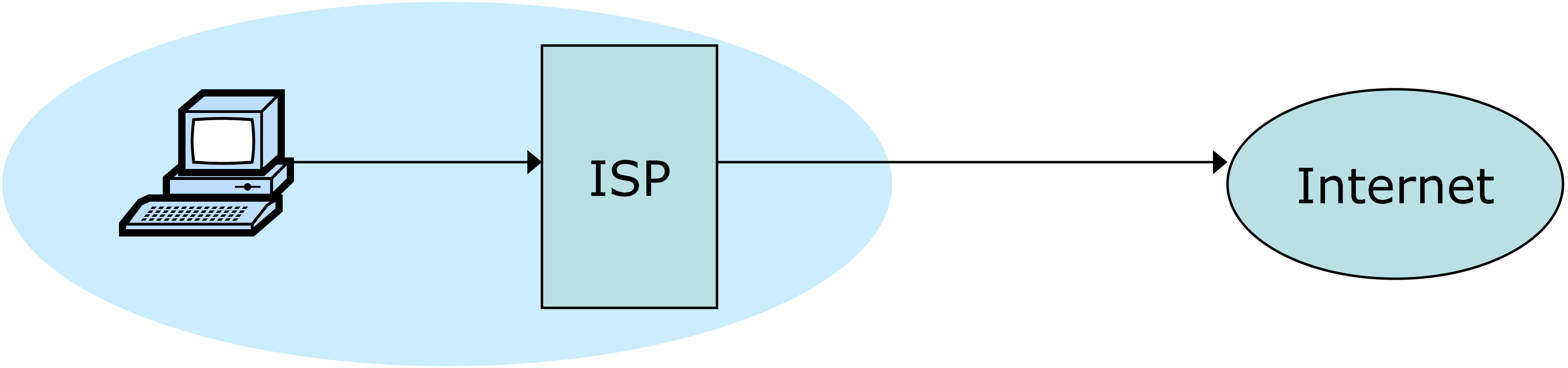
with spoofed IP addresses

Ingress Filtering

- How to find packet origin?

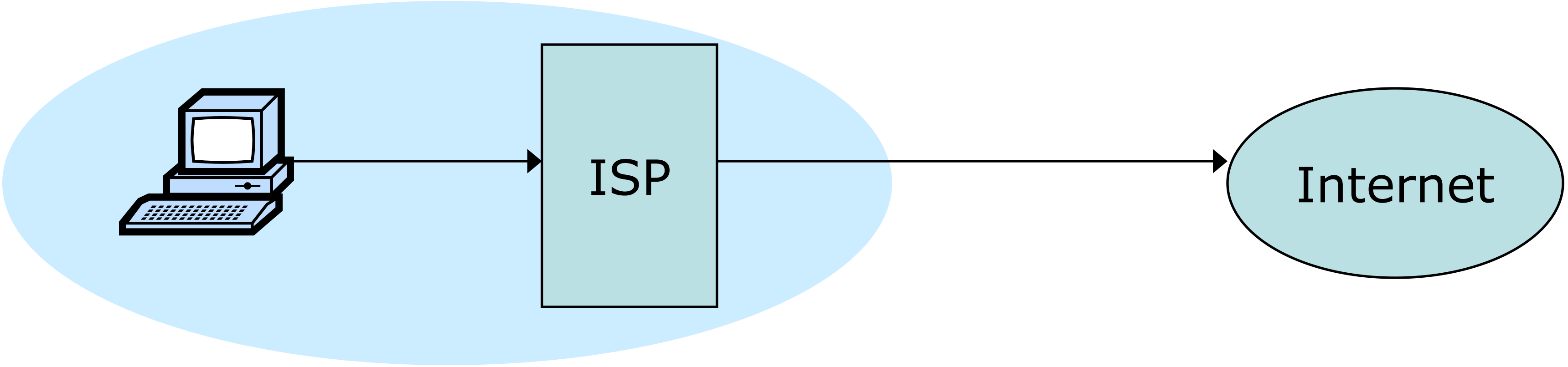
Ingress Filtering

- How to find packet origin?



Ingress Filtering

- How to find packet origin?



- Ingress filtering policy:

  ISP only forwards packets with legitimate source IP

Ingress Filtering

Implementation challenges:

- All ISPs need to do this — requires global coordination:

  If 10% of networks don’t implement, there’s no defense;

  No incentive for an ISP to implement — doesn’t affect them;

Ingress Filtering

Implementation challenges:

- As of 2017 (from CAIDA):

  33% of autonomous systems allow spoofing;

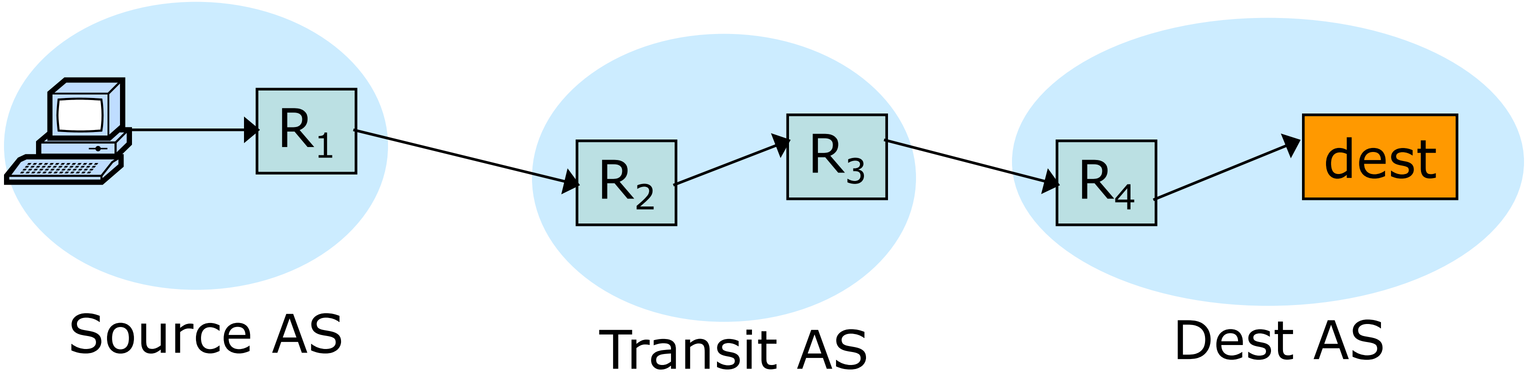
  23% of announced IP address space allow spoofing;

Ingress Filtering

- Can transit AS verify packet origin?

Ingress Filtering

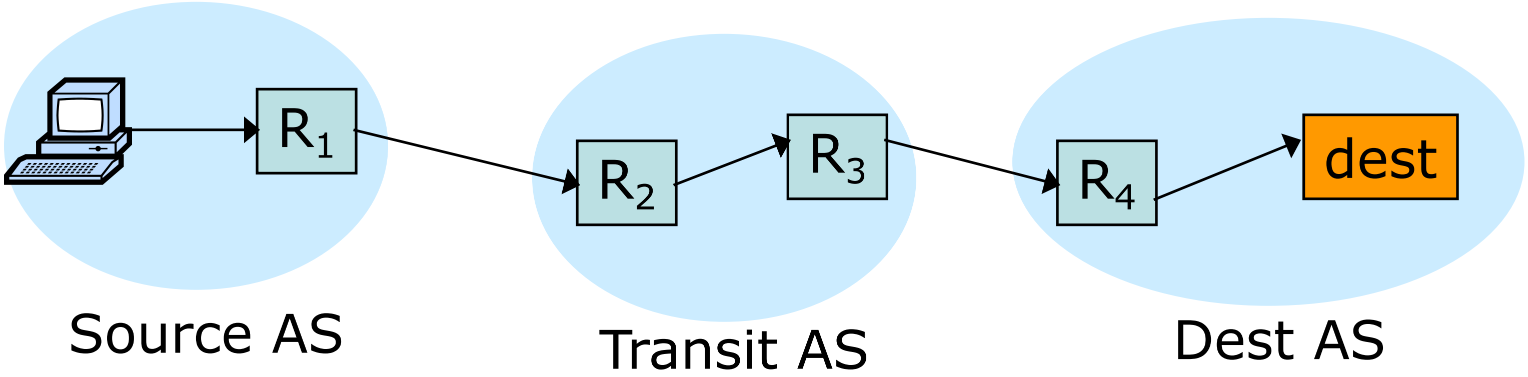
- Can transit AS verify packet origin? No



- Routing protocols care about only destination IP addresses

Ingress Filtering

- Can transit AS verify packet origin? Yes



- Routing protocols care about only destination IP addresses
- Were routing protocols modified…

Traceback

- Goal

  given set of attack packets

  determine path to source

- How

  change routers to record info in packets

Traceback

- Goal

  given set of attack packets

  determine path to source

- How

  change routers to record info in packets

- Assumptions

  trusted routers

  sufficient packets to track

  stable route from attacker to victim

Traceback

- Write path into packets

  router adds its own IP address to packet

  victim reads path from packet

Traceback

- Write path into packets

  router adds its own IP address to packet

  victim reads path from packet

- Limitations

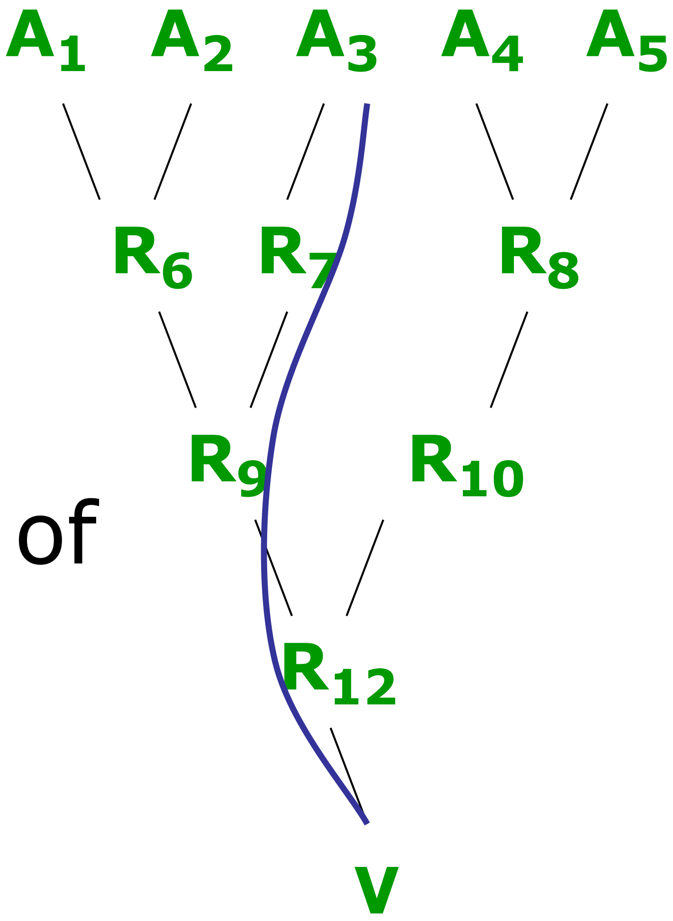
  requires space in packet

  path can be long

  no extra fields in current IP format (changes to packet format too much to expect)

Traceback

- Sample and Merge
  store one link in each packet;
  router probabilistically stores own address;
  fixed space regardless of path length;

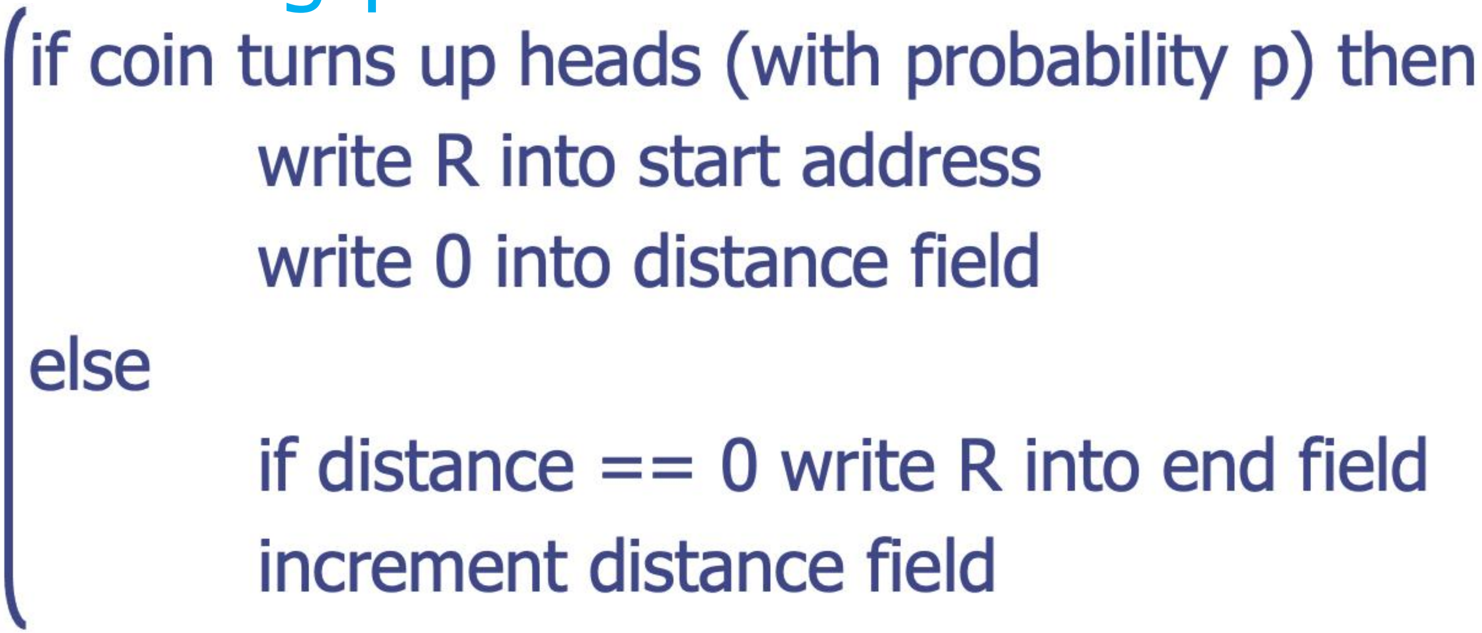


Traceback

- Edge Sampling: fields into packet

  edge: *start* and *end* IP addresses

  distance: no. of hops since edge stored
- Marking procedure of router R



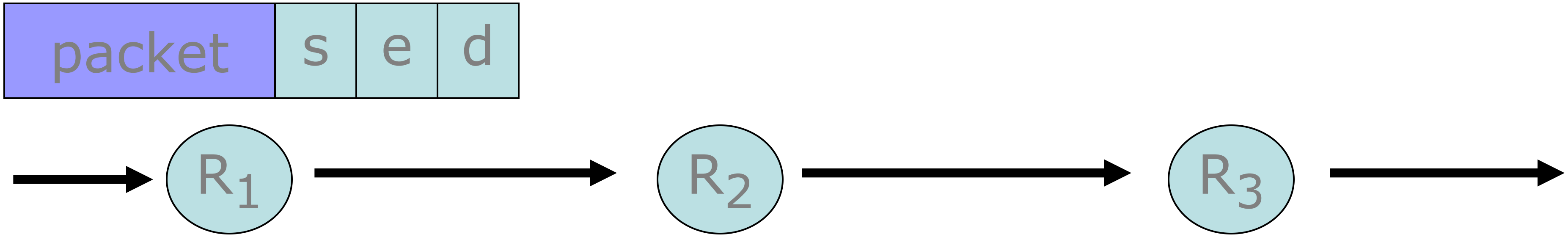
```
if coin turns up heads (with probability p) then
        write R into start address
        write 0 into distance field
else
        if distance == 0 write R into end field
        increment distance field
```

Traceback

- Packet received

  R_1 receives packet from source or another router;

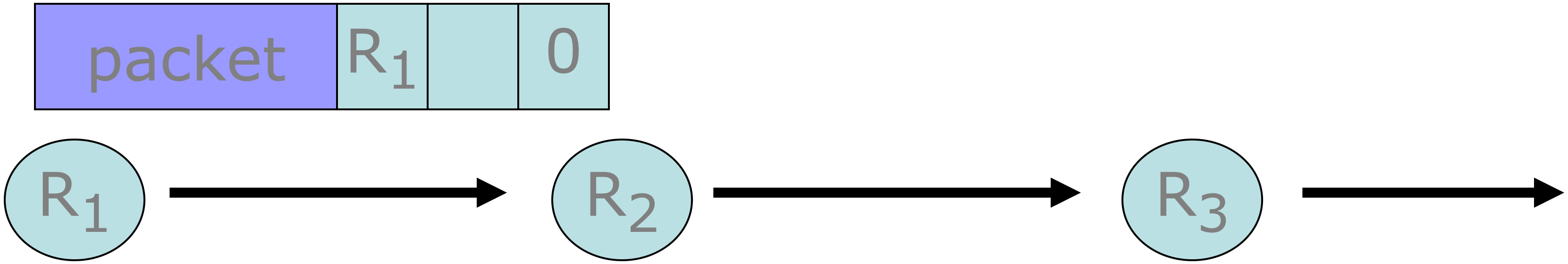
  packet contains space for start, end, distance;



Traceback

- Begin writing edge

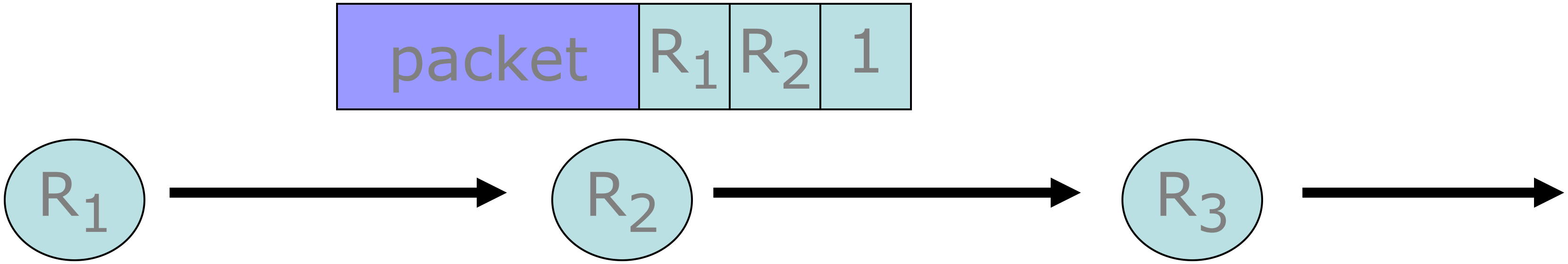
  R_1 chooses to write start of edge; sets distance to 0;



Traceback

- Finish writing edge

  R_2 chooses not to overwrite edge; distance is 0: write end of edge, increment distance to 1;

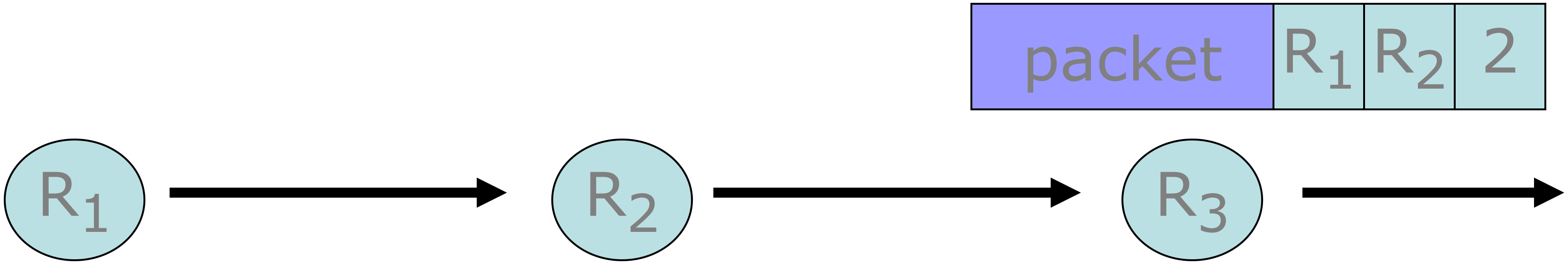


Traceback

- Increment distance

  R_3 chooses not to overwrite edge;

  distance>0: increment distance to 2;

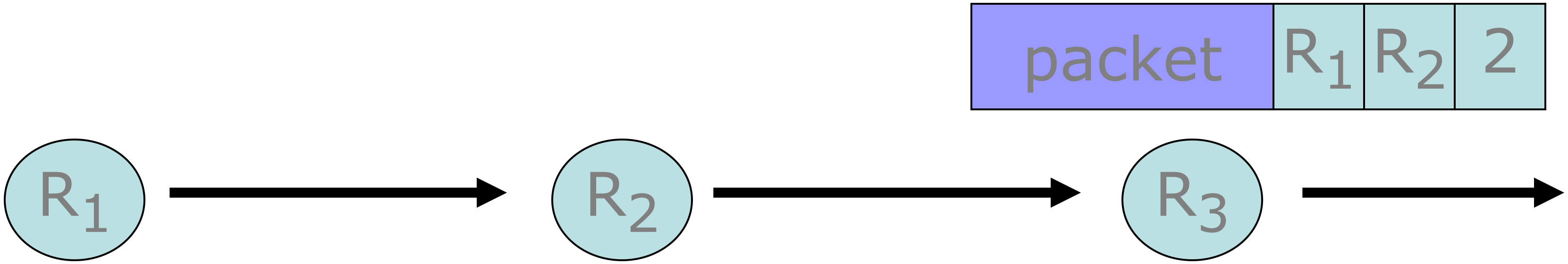


Traceback

- Increment distance

  R_3 chooses not to overwrite edge;

  distance>0: increment distance to 2;



- **Were routing protocols modified…**

Path Validation

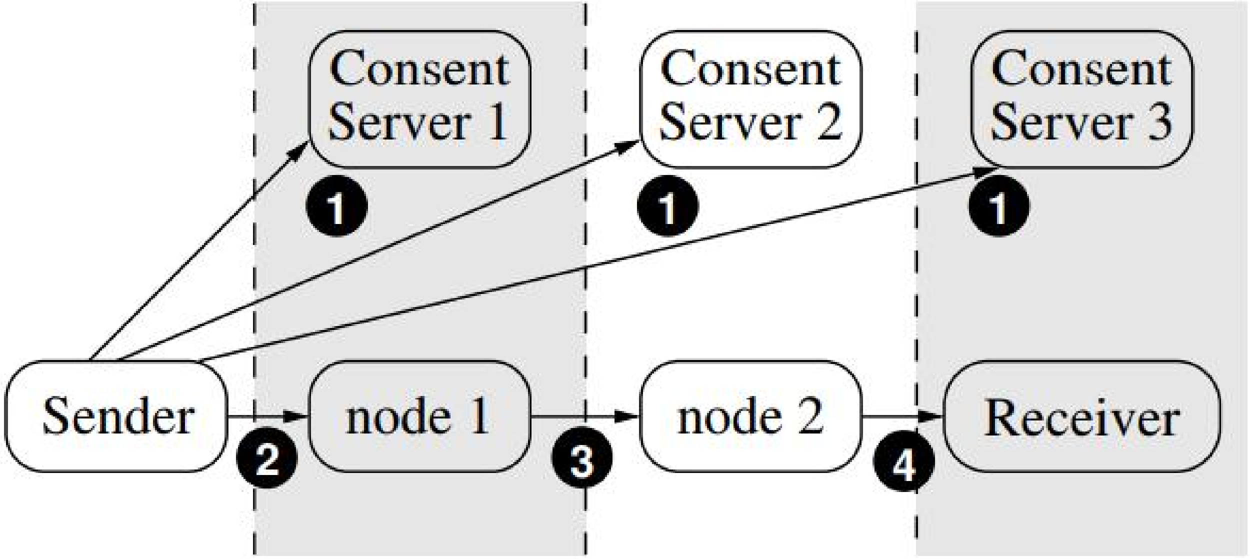
- PoC: Proof of Consent

  certify the provider’s consent to carry traffic along the path

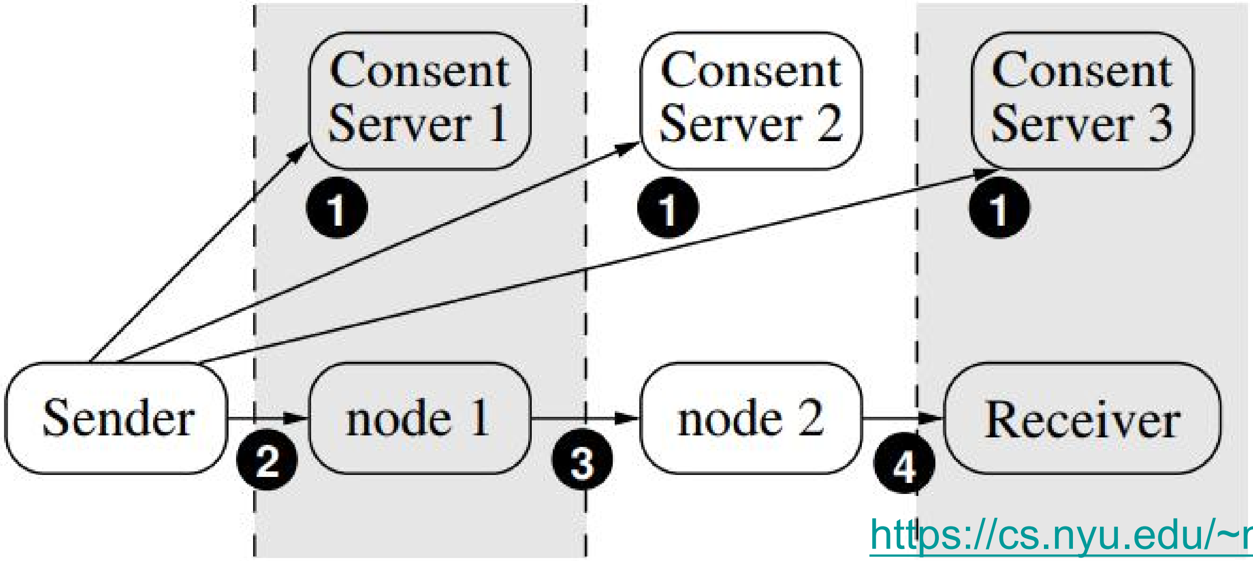
- PoP: Proof of Provenance

  allow upstream nodes to prove to downstream nodes that they carried the packet

Path Validation



Path Validation



https://cs.nyu.edu/~mwalfish/papers/icing-conext11.pdf

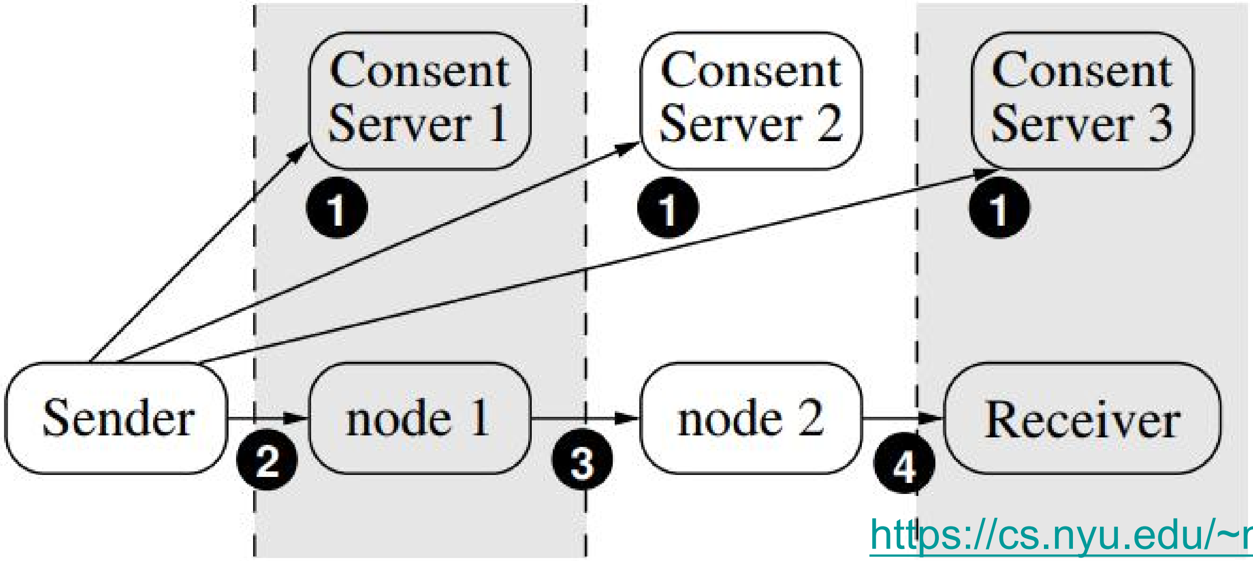
P	N_0	N_1	N_2	N_3
V_1	$A_1 \oplus$ PoP$_{0,1}$			
V_2	$A_2 \oplus$ PoP$_{0,2}$			
V_3	$A_3 \oplus$ PoP$_{0,3}$			
	Payload			

2

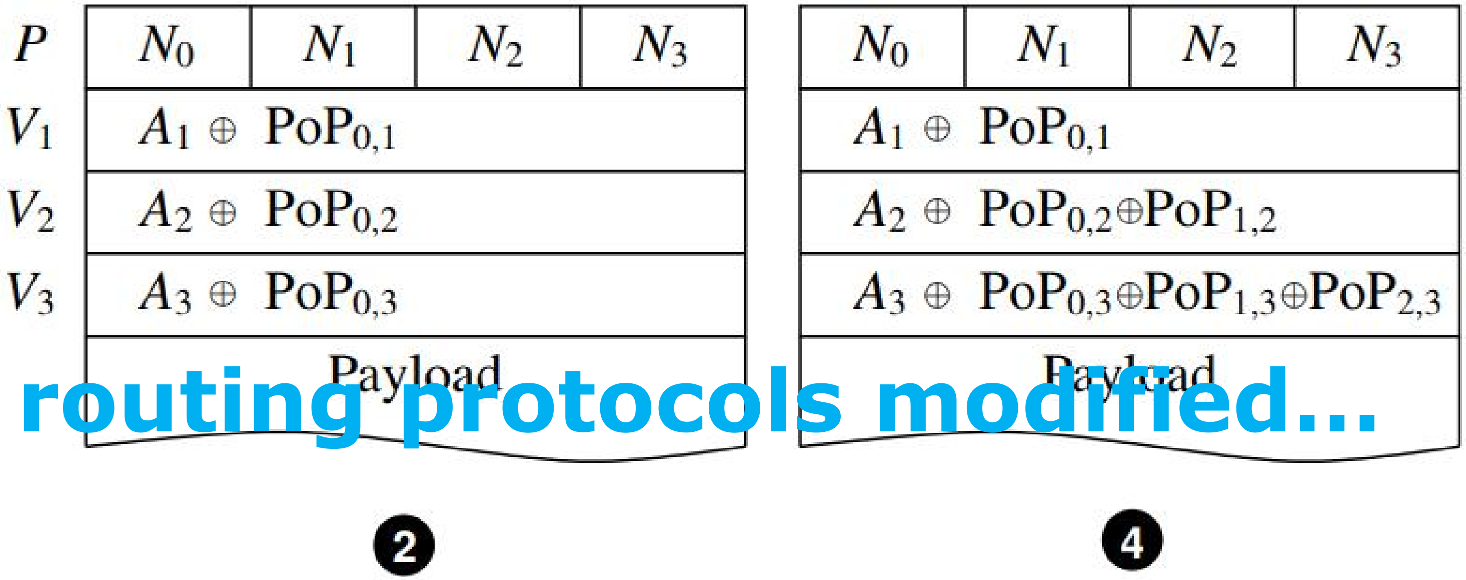
N_0	N_1	N_2	N_3
$A_1 \oplus$ PoP$_{0,1}$			
$A_2 \oplus$ PoP$_{0,2} \oplus$PoP$_{1,2}$			
$A_3 \oplus$ PoP$_{0,3} \oplus$PoP$_{1,3} \oplus$PoP$_{2,3}$			
Payload			

4

Path Validation



https://cs.nyu.edu/~mwalfish/papers/icing-conext11.pdf



P	N_0	N_1	N_2	N_3
V_1	$A_1 \oplus \mathrm{PoP}_{0,1}$			
V_2	$A_2 \oplus \mathrm{PoP}_{0,2}$			
V_3	$A_3 \oplus \mathrm{PoP}_{0,3}$			
	Payload			

2

N_0	N_1	N_2	N_3
$A_1 \oplus \mathrm{PoP}_{0,1}$			
$A_2 \oplus \mathrm{PoP}_{0,2} \oplus \mathrm{PoP}_{1,2}$			
$A_3 \oplus \mathrm{PoP}_{0,3} \oplus \mathrm{PoP}_{1,3} \oplus \mathrm{PoP}_{2,3}$			
Payload			

4

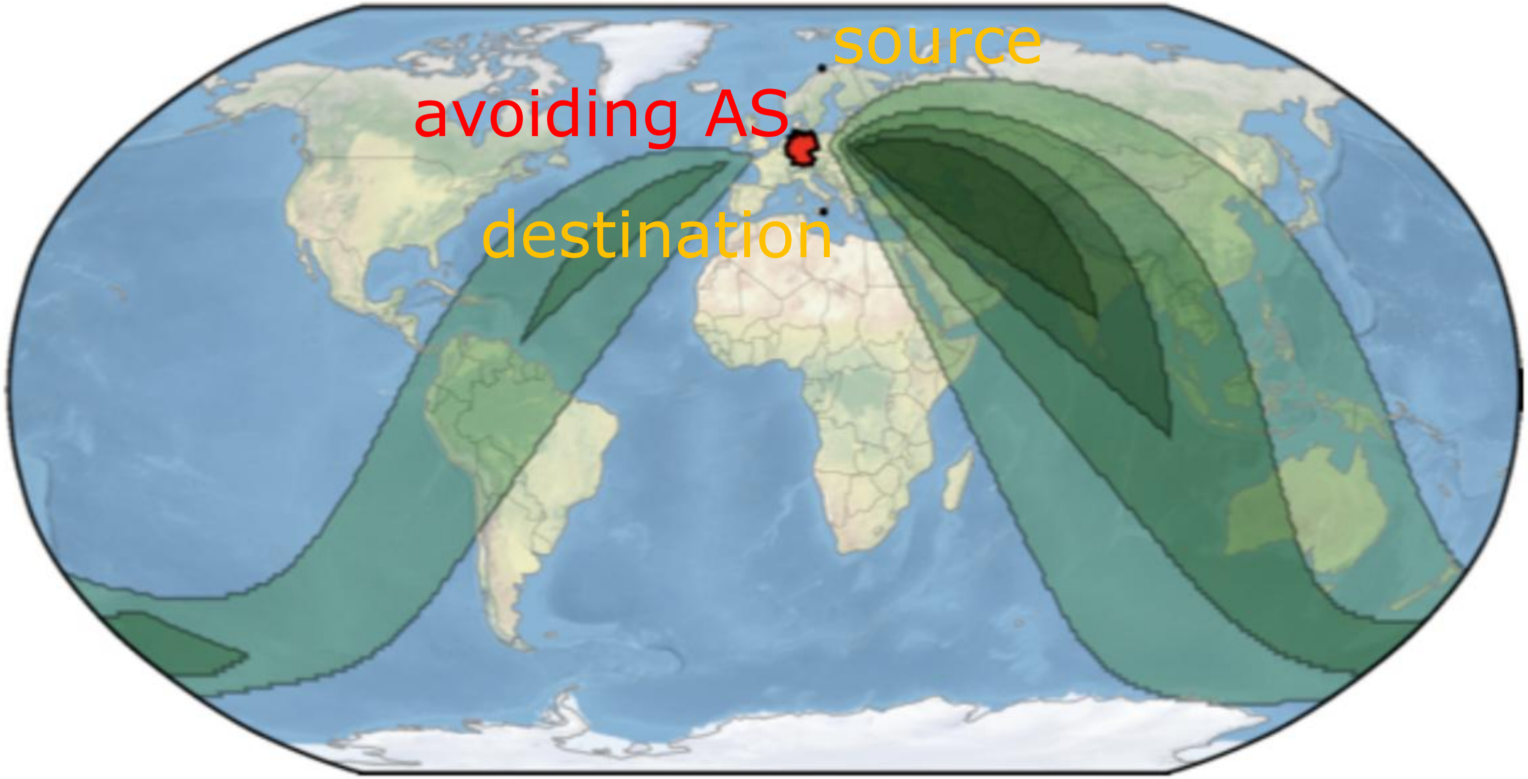
- Were routing protocols modified…

Alibi Routing

- How to verify that a packet DO NOT transmit via a specific AS?

Alibi Routing

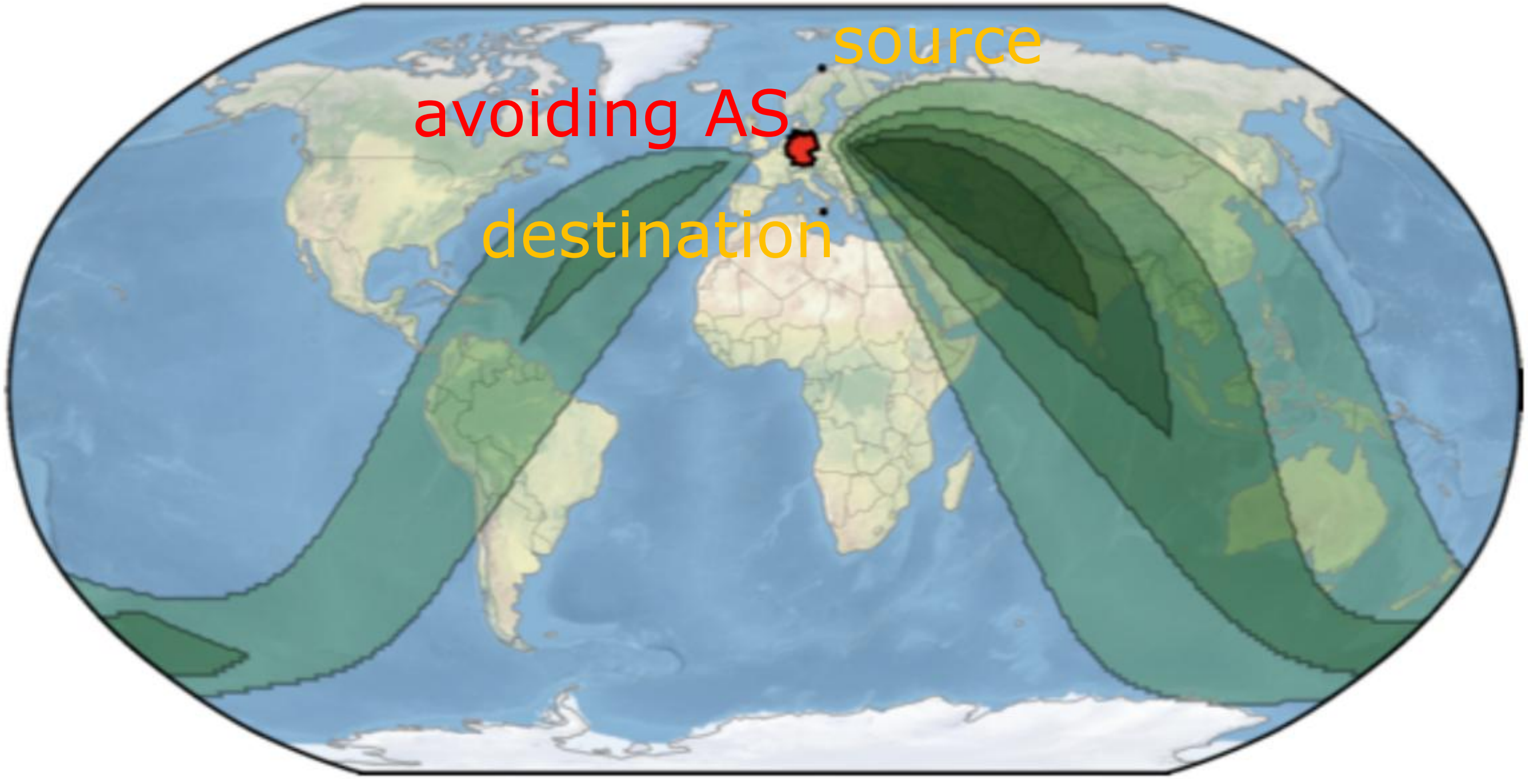
- How to verify that a packet DO NOT transmit via a specific AS?



https://conferences.sigcomm.org/sigcomm/2015/pdf/papers/p611.pdf

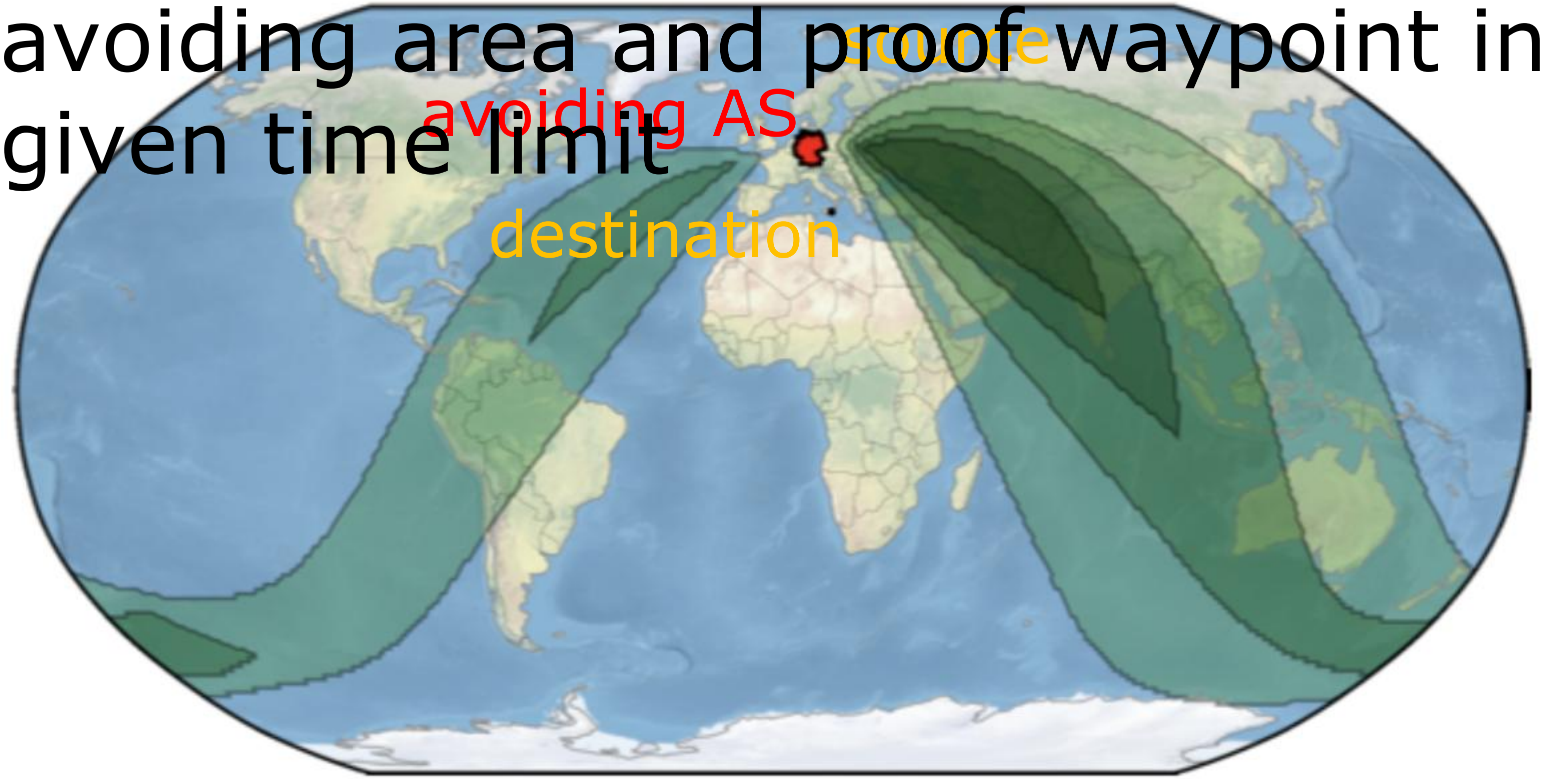
Alibi Routing

- Introduce proof waypoint



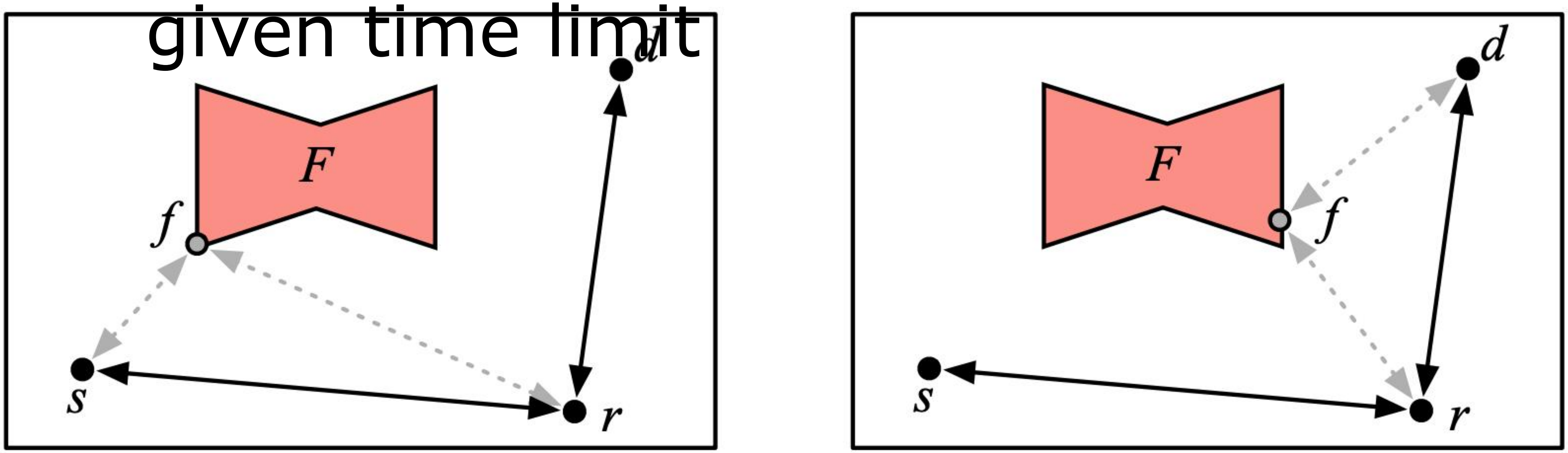
Alibi Routing

- Introduce proof waypoint such that packets cannot transmit via both avoiding area and proof waypoint in a given time limit



Alibi Routing

- Introduce proof waypoint such that packets cannot transmit via both avoiding area and proof waypoint in a given time limit

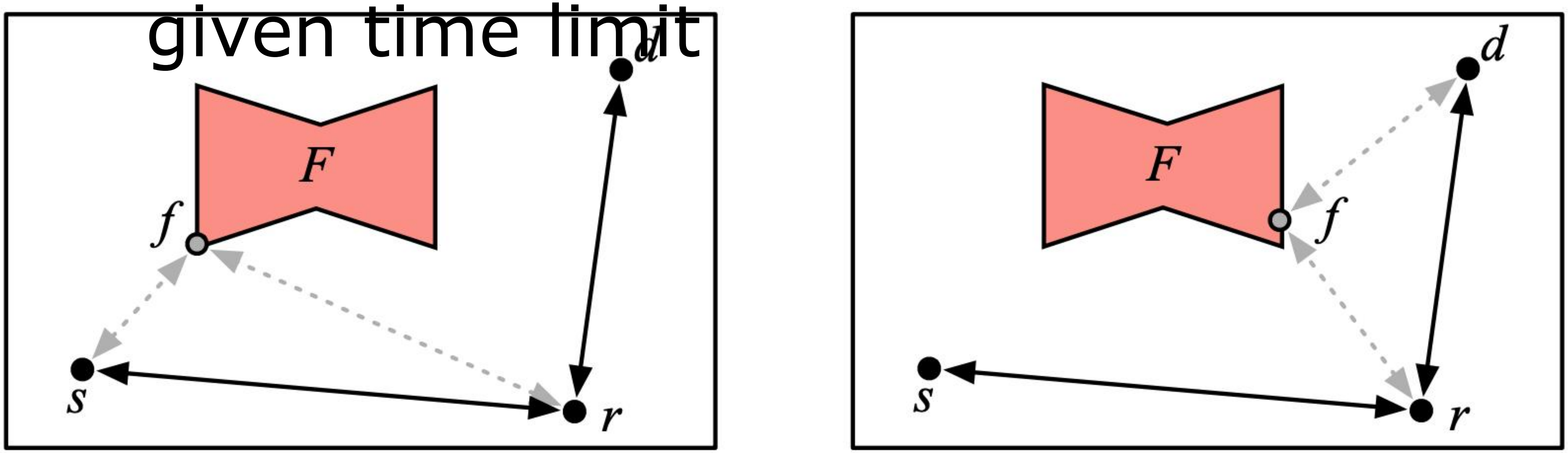


(a) $R(s,r) + R(r,d) \ll \min_f \{R(s,f) + R(f,r)\} + R(r,d)$

(b) $R(s,r) + R(r,d) \ll R(s,r) + \min_f \{R(r,f) + R(f,d)\}$

Alibi Routing

- Introduce proof waypoint such that packets cannot transmit via both avoiding area and proof waypoint in a given time limit



(a) $R(s,r) + R(r,d) \ll \min_f \{R(s,f) + R(f,r)\} + R(r,d)$

(b) $R(s,r) + R(r,d) \ll R(s,r) + \min_f \{R(r,f) + R(f,d)\}$

- Were routing protocols modified…

DDoS defenses

make attacker harder to attack

DDoS defenses

make attacker harder to attack

cost more resources from attacker

Client Puzzles

- Idea

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

- Example

  server sends: C

  client: given challenge C find X s.t.

  $LSB_n(SHA\text{-}1(C||X)) = 0^n$

Client Puzzles

- Benefits

  invoked upon attack detection;

  can tune n in reactive to amount of attack traffic;

- Limitations

  require changes to protocols, clients, and servers;

  during attack, hurts low-power legitimate clients (e.g., phones);

CAPTCHA

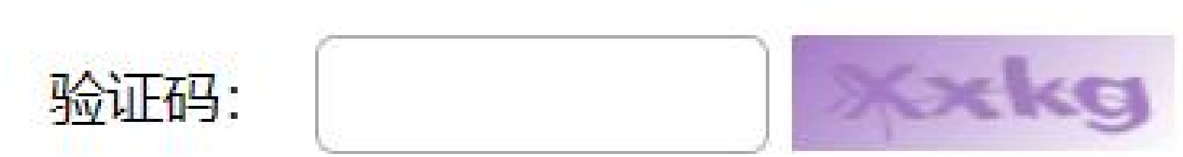
- Completely Automated Public Turing test to tell Computers and Humans Apart

CAPTCHA

- Completely Automated Public Turing test to tell Computers and Humans Apart
- challenge–response test used in computing to determine whether or not the user is human

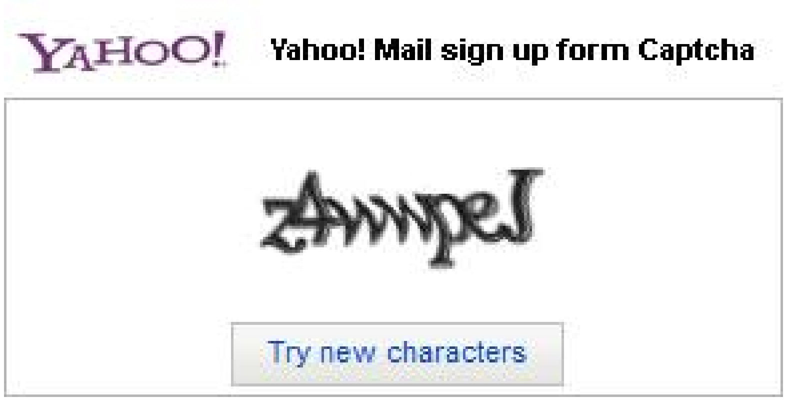
CAPTCHA

- Completely Automated Public Turing test to tell Computers and Humans Apart
- challenge–response test used in computing to determine whether or not the user is human

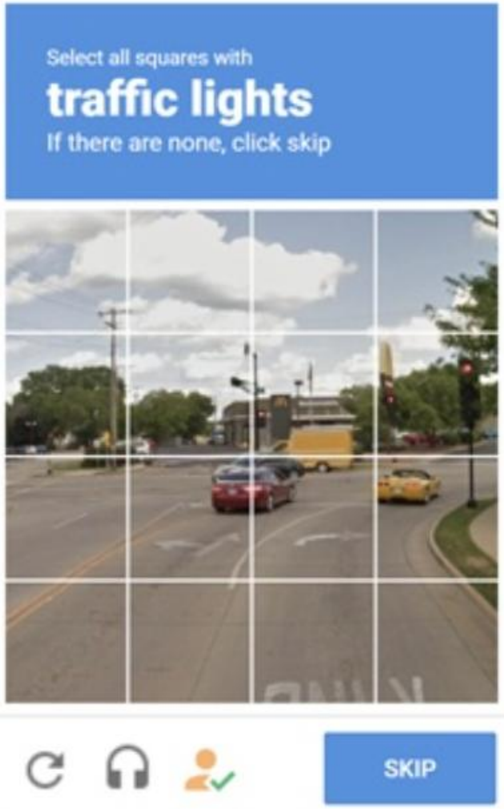


CAPTCHA

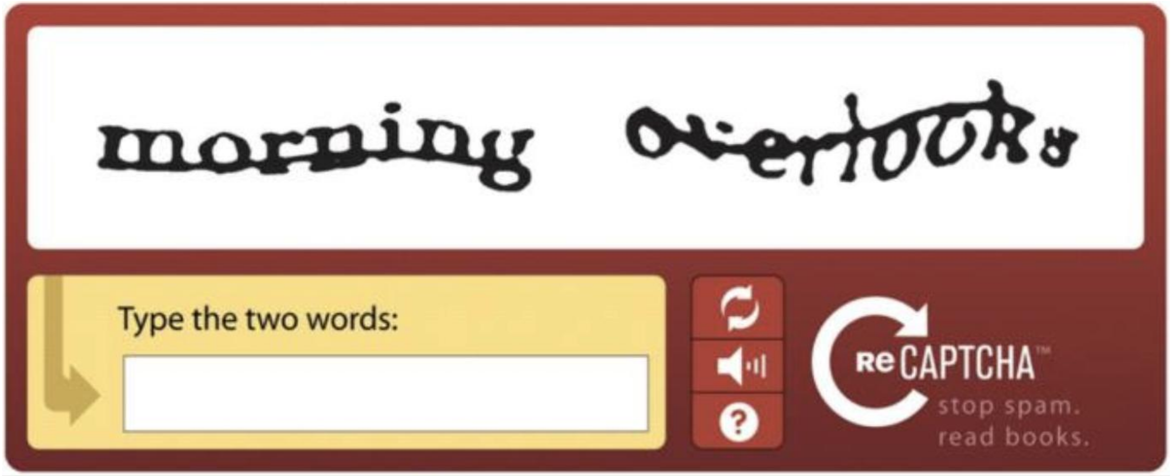
- Text, image, audio…



Yahoo! Mail sign up form Captcha



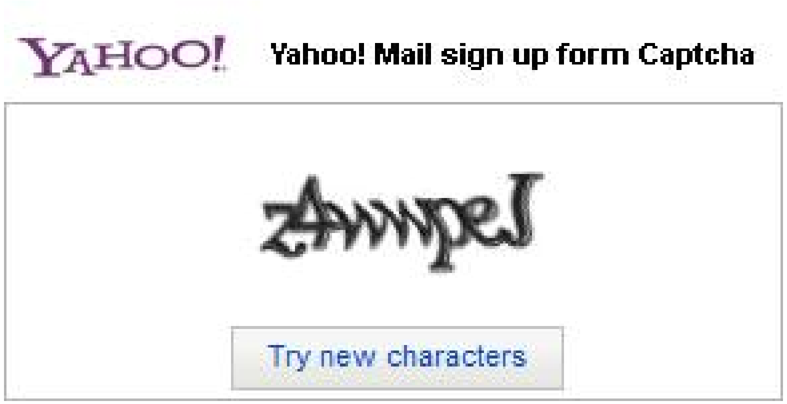
Select all squares with
traffic lights
If there are none, click skip

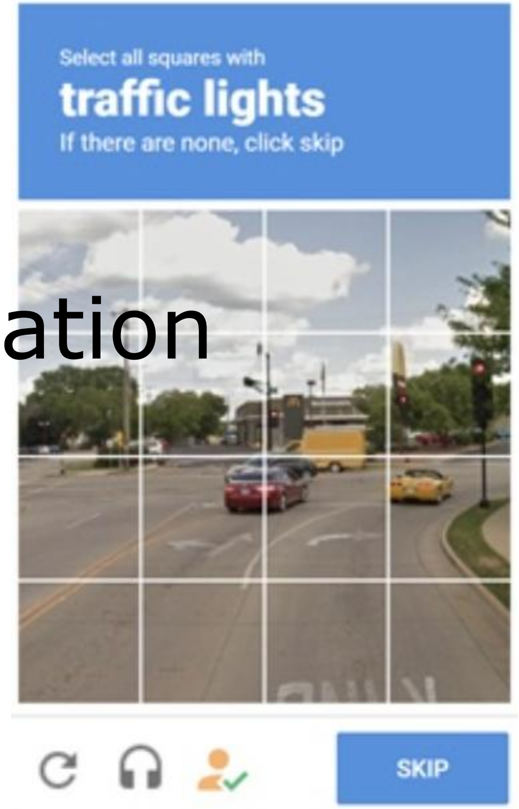


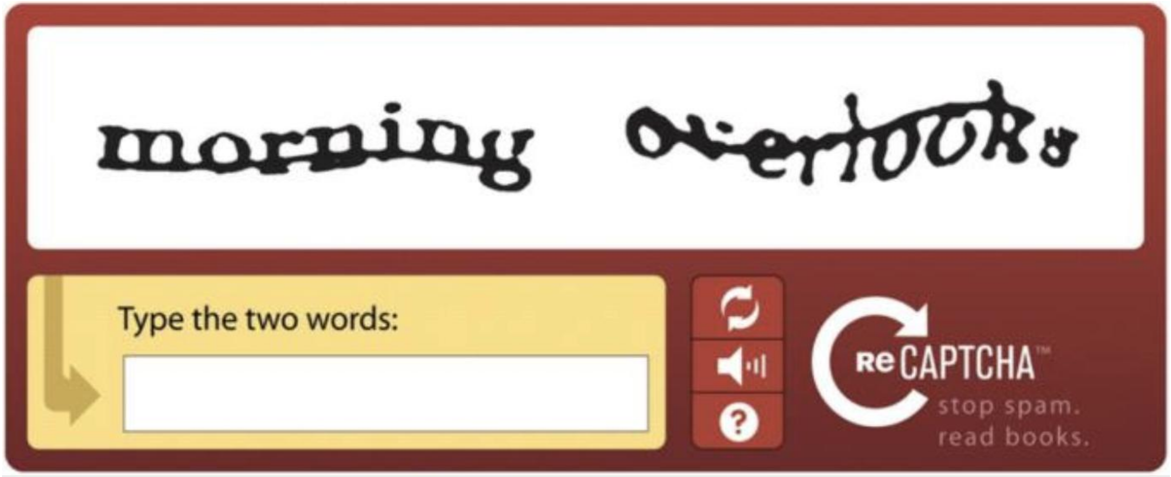
Type the two words:

CAPTCHA

- Vulnerable to auto-identification

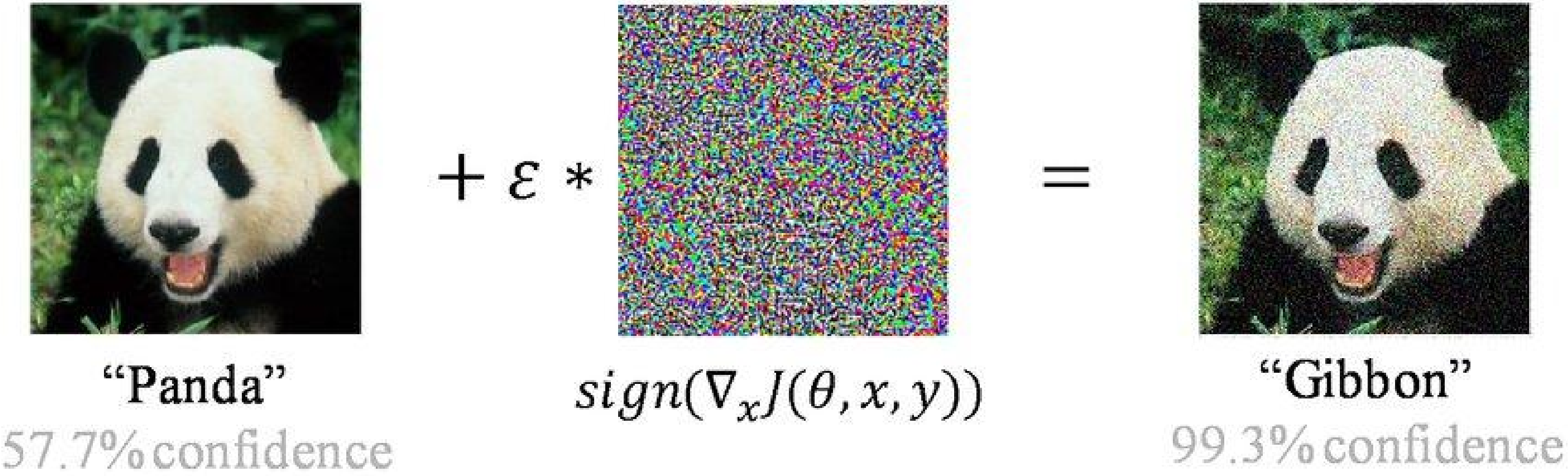






CAPTCHA

- Adversarial CAPTCHA



shorturl.at/bBVY0

dndyjmzxixi

qydszsyd

?

Readings

- What is a DDoS Attack?
  by CLOUDFLARE
- Protocol Security and DoS Attacks
  by Dan Boneh and Zakir Durumeric
- Denial of Service Attacks
  by Zulfikar Ramzan

Thank You