Network and Protocol Security

Francesco Mecca
Seminario di Complementi di Reti
2018/2019

Classes of Attacks

We will focus on:

I. Unauthorized access and information gathering

II. Packet capture and analysis

III. Host impersonation

IV. Denial of Service

Structure of the talk

We will study every class of attack by showing historical examples and techniques, following this order:

- Network access layer
- Internet layer
- Transport layer
- Application layer (and/or protocols)

We will deal with...

- Wireless
- Switch
- Hub

- TCP
- UDP
- IP

- ICMP
- ARP
- BGP

- HTTP
- DNS
- FTP

I. Information Gathering

- Many successful cyber attacks are social engineering attacks
- Information Gathering is crucial for social engineering
- Possible solutions:
  - Authentication
  - Authorization
  - Security through obscurity (weak)
  - Hardening (requires lots of knowledge)

Service and port scanning

- Pingscan: map hosts of a network using ICMP echo datagrams
- UDP port scanning:
  - Send 0 length packets to every port
  - If ICMP “port unreachable” error is sent back, service is unavailable
- TCP connect() port scanning:
  - Open a connection to every port
  - If handshake is successful, service is alive

Advanced port scanning

Previous techniques are very noisy and easily detectable

- TCP SYN scanning:
  - Attackers sends a SYN packet
  - If the server responds with SYN/ACK packet, the service is available
  - If the server responds with a RST packet, the service is unavailable
  - The attackers reply with a RST packet instead of ACK so that the connection is not open and the event is not logged

Advanced port scanning #2

Many other flags combinations for TCP:

- TCP FIN scanning:
  - Attackers sends a FIN flagged packet
  - If the server ignores the packed, the port is open
  - If the server responds with a RST packet, the service is unavailable
  - Could be done with PSH, URG and even no flags

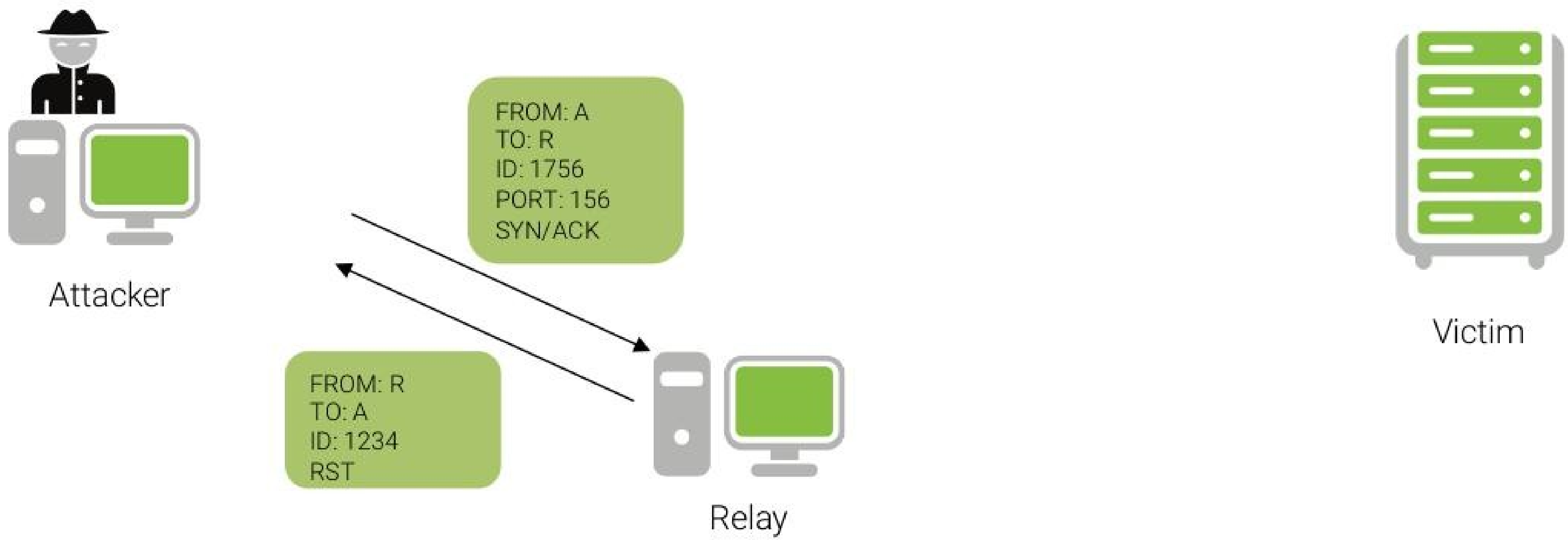
<u>All of these techniques rely on undocumented methods:</u>

<u>the results are not reliable and difficult to reproduce</u>

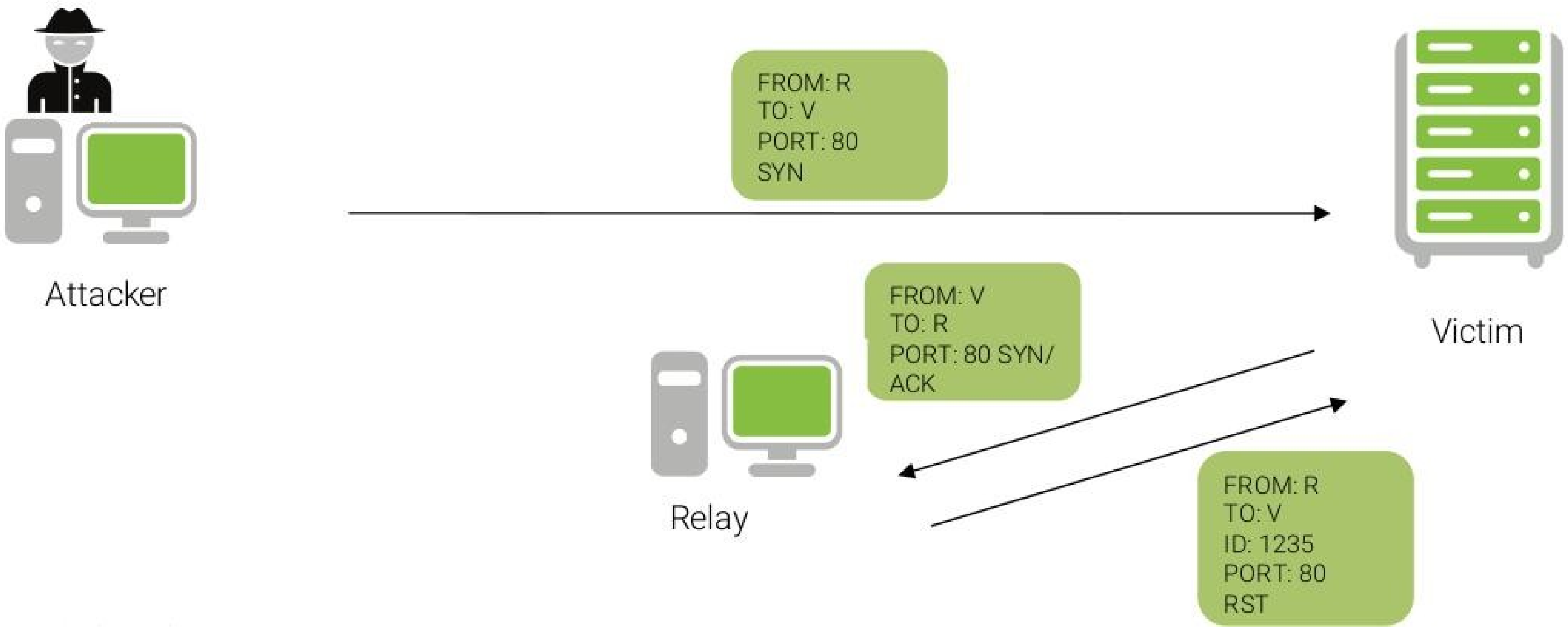
Idle Scanning

- Uses a victim host to “relay” the scan
- The attacker sends spoofed TCP SYN packets to the target
- The packets appear to come from the victim
- The target replies to the victim
  - If the target replies with a SYN+ACK packet (open port) then the victim will send out a RST
  - If the target replies with a RST (closed port) then the victim will not send out any packet
- The attacker checks the IP datagram ID of the victim before and after each port probe
  - If it has increased: port on target was open
  - If it has not increased: port on target was closed

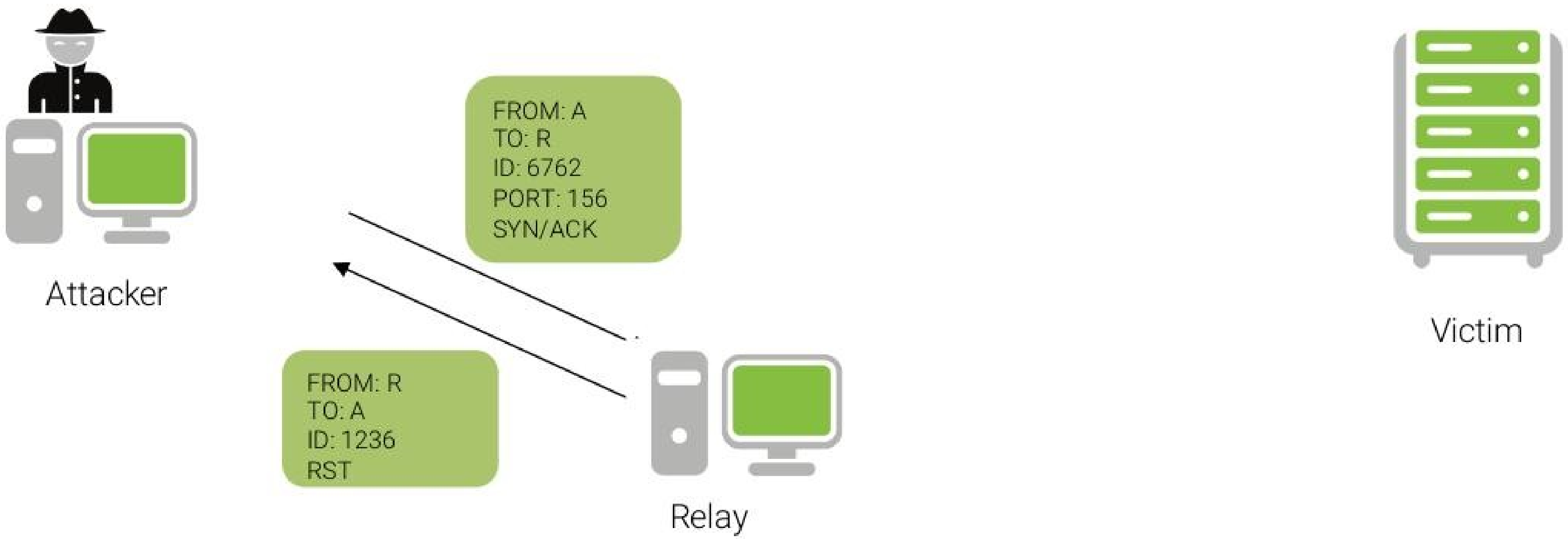
Step 1: determine the relay's initial IP sequence number



Step 2: send a spoofed connection request



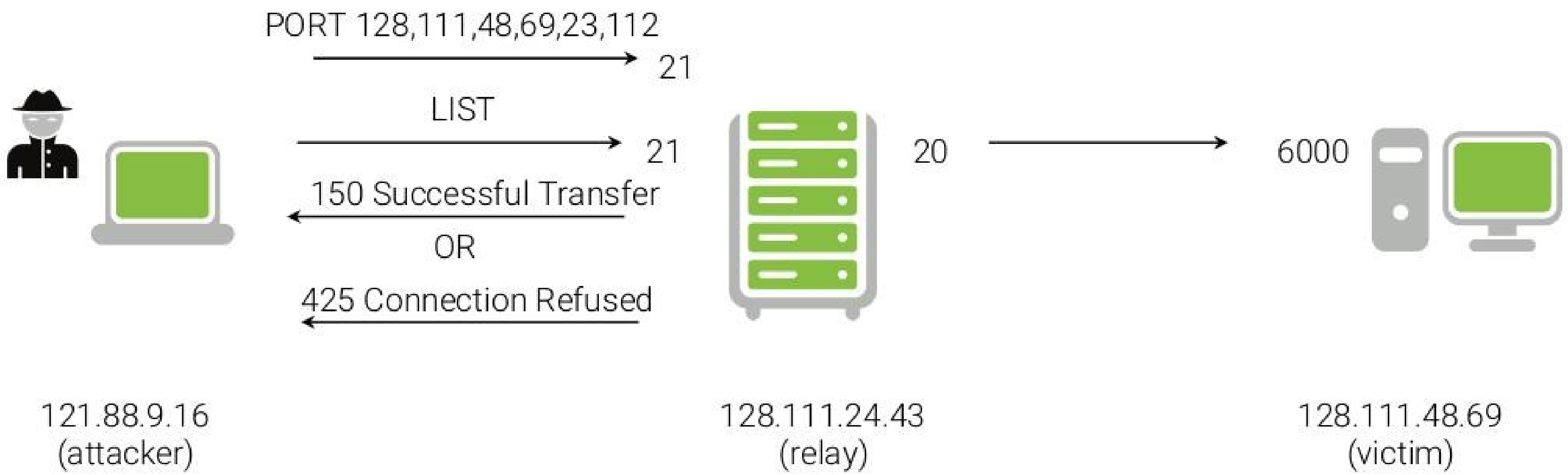
Step 3: determine the relay's final IP sequence number



FTP Bounce Scan

- In the FTP protocol uses two stream for control and data
- The PORT command is used by the client to tell the server the address and port to be used when opening a data connection
- The data port need not be in the same host that initiates the FTP commands via the control connection
- Therefore it is possible to instruct a server to open a connection to a third host

Example



PORT 128,111,48,69,23,112
21
LIST
21
150 Successful Transfer
OR
425 Connection Refused
20
6000
121.88.9.16
(attacker)
128.111.24.43
(relay)
128.111.48.69
(victim)

FTP Bounce Attack

- Can be used to execute a TCP portscan
  - The host that appears to be the source of the scan is the FTP server
  - It is possible to scan a host that is behind a firewall exploiting the trust relationship
- Can be used to bypass restrictions and access control

OS fingerprinting

Determine the operating system of a host by examining the reaction to carefully crafted packets, up to the kernel version, and exploit unpatched vulnerabilities

- Wrong answers to FIN TCP packets
- “Undefined” flags in the TCP header of a request are copied verbatim in the reply
- Weird combinations of flags in the TCP header
- Selection of TCP initial sequence numbers
- Selection of initial TCP window size
- Analysis of the use of ICMP messages
- Error rate
  - Amount of offending datagram included
  - TCP options

- OS fingerprinting also can be performed in a passive way using tools such as p0f, ettercap or by performing the same analysis on different protocols

II. Packet capture and analysis

- The act intercepting and logging traffic over a link, A.K.A. sniffing
- Easier in the case of wireless networks and hubs
- In switched environments, the attacker must convince the switch to send him a copy of the traffic
- Passive form of information gathering
- Depending on network configuration, very hard to detect

Tools

- May protocols sends information in clear:
  - FTP, HTTP, IMAP, XMPP, etc…
- Even if the payload is encrypted, attackers can collect metadata
- Tools:
  - libpcap
  - tcpdump, tcp replay, tcpflow
  - Burp
  - Wireshark

Sniffing in promiscuous networks

- Hubs, wireless networks are susceptible to sniffing
- Network cards can be configured to accept packets sent to different interfaces
  - Promiscuous mode
  - Monitor mode
- Wardriving / wardialing: access points and hosts can be probed without any prior knowledge or physical access

Sniffing in switched Ethernet

Switched Ethernet does not allow direct sniffing

- MAC flooding
  - Switches maintain a table with MAC address/port mappings
  - In some cases, flooding the switch with bogus MAC addresses will overflow the table’s memory and revert the behavior from “switch” to “hub”
- MAC duplicating/cloning
  - Attacker reconfigures his/her host to have the same MAC address as the target machine
  - The switch will record this in its table and send the traffic to the attacker machine (or possibly both)

III. Host impersonation

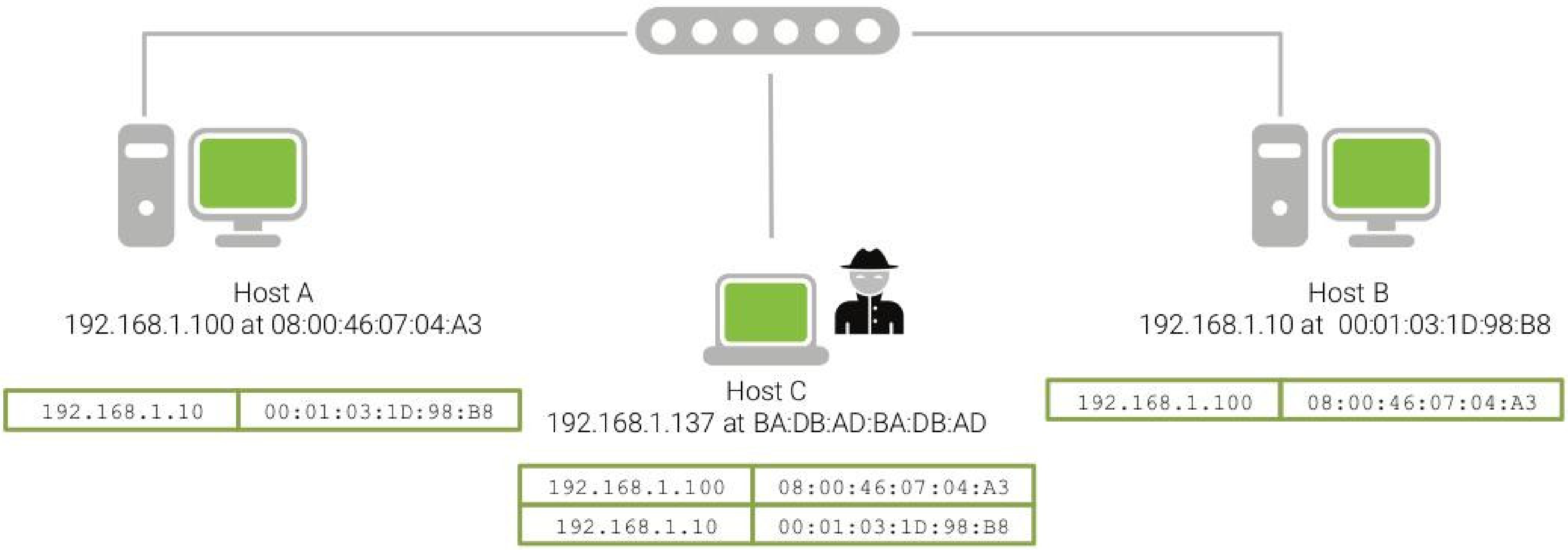
- The attacker disguises himself as the known source or the destination host of the communication
- It manipulates the protocol by forging the data used for routing and access
- Also known as spoofing
- Particularly effective in the absence of authentication and identity verification

ARP spoofing

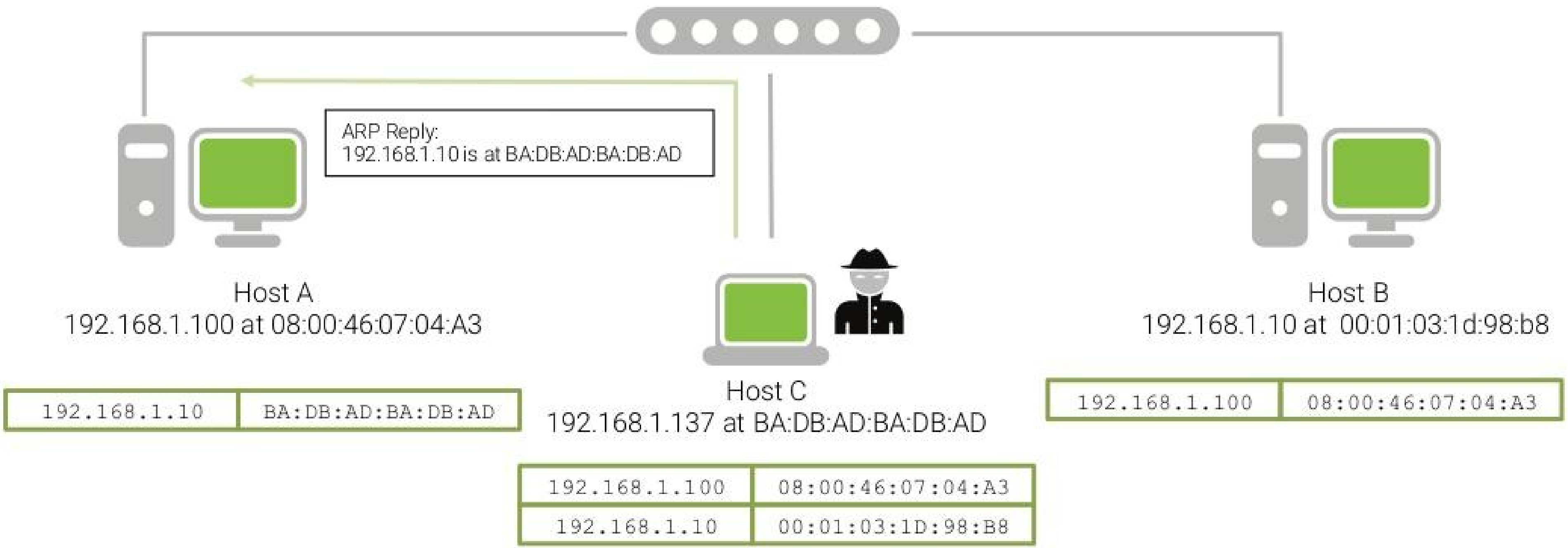
Sniff and manipulates traffic between two hosts in a switched environment

- The attack leverages the stateless nature of the ARP protocol
  - Replies without a request will be accepted
- The attacker host sends spoofed ARP messages to the two victim hosts, poisoning their cache
- The victim host sends their IP packets to the attacker host
  - The attacker host acts has a router
  - Continuously monitor and resend spoofed ARP replies

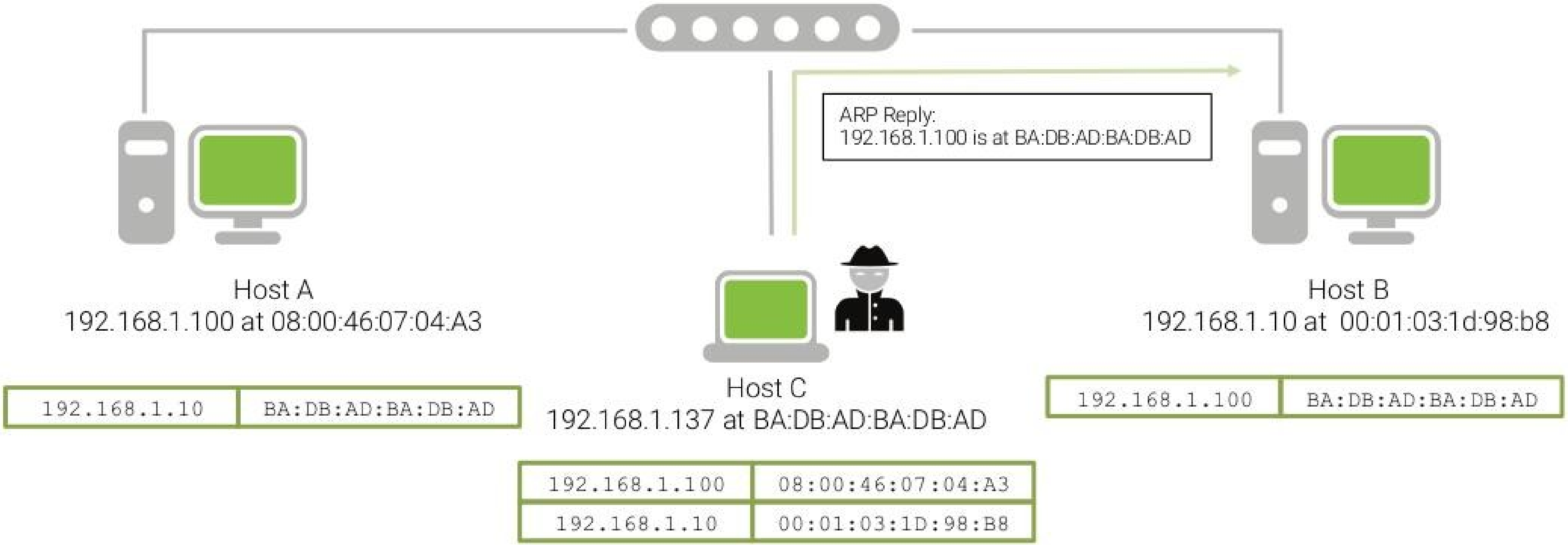
Poisoning the ARP table #1



Poisoning the ARP table #2



Poisoning the ARP table #3



ARP Spoofing Defense

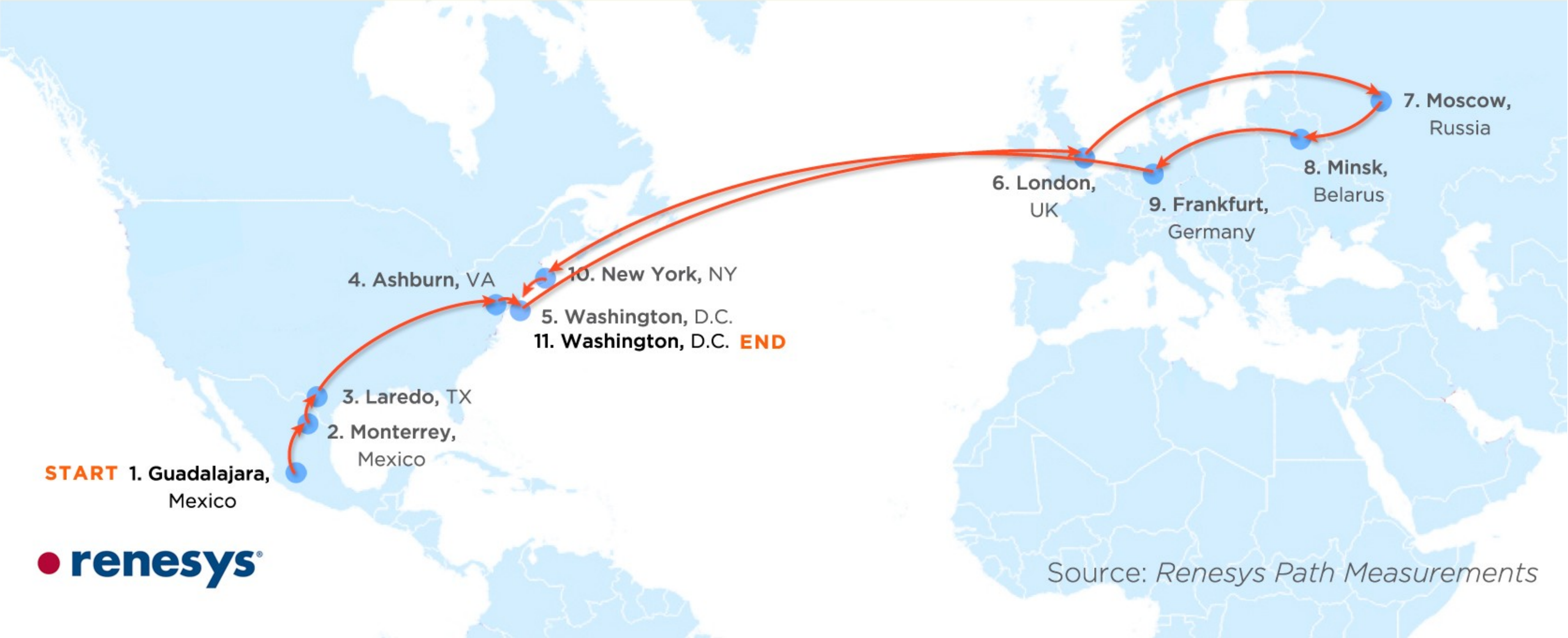
- Static ARP entries
  - The ARP cache can be configured to ignore dynamic updates
  - Difficult to manage in large deployments
  - Could be used for a subset of critical addresses (e.g., DNS servers, gateways)
- Cache poisoning resistance
  - Ignore unsolicited ARP replies (still vulnerable to races)
  - Update on timeout (weak)
- Monitor changes (arpwatch)
  - Listen for ARP packets on a local Ethernet interface
  - Keep track for Ethernet/IP address pairs
  - Report suspicious activity and changes in mapping

BGP Rerouting

- BGP stores many paths for a given destination
- Best path is chosen in relation to a list of attributes: granular control over which AS gets the traffic
- Malicious nodes can advertise false attributes

  - Weight
  - Local Preference
  - Originate
  - AS Path length
  - Origin Code
  - MED
  - eBGP vs IBGP
  - Shortest IGP to next BGP
  - Oldest Path
  - Router ID
  - Neighbor IP Address
  - Others depending on vendor

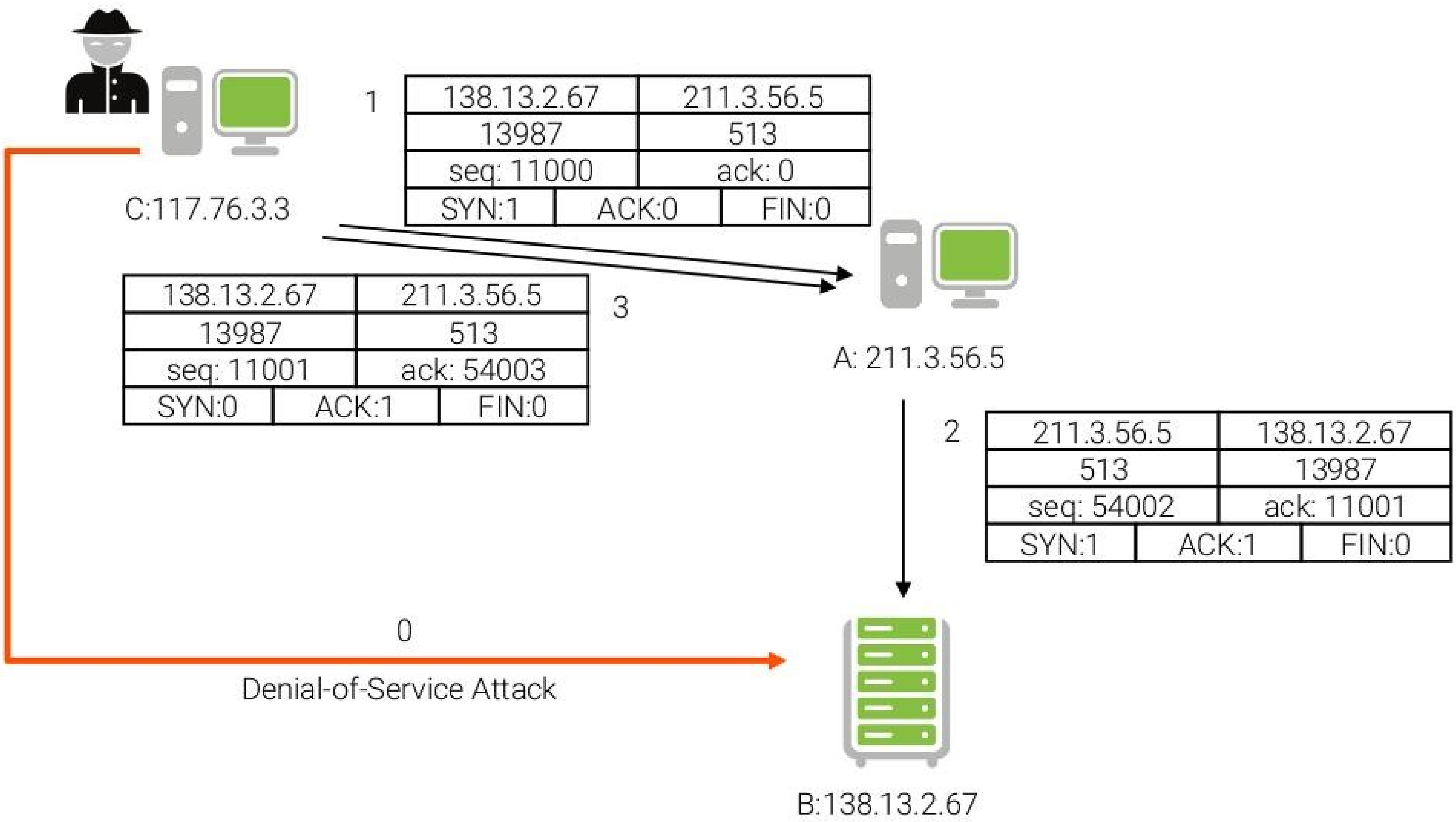
Traceroute Path 1: from Guadalajara, Mexico to Washington, D.C. via *Belarus*



TCP Spoofing

- Alice trusts Bob
- Eve wants to impersonate Bob with respect to Alice in opening a TCP connection
- Eve kills Bob (flooding, crashing, redirecting) so that Bob does not send annoying RST segments
- Eve sends a TCP SYN segment to Alice in a spoofed IP packet with Bob's address and seq num Ss
- Alice replies with a TCP SYN/ACK segment to Bob with seq num Sc. Bob ignores the segment: dead or too busy
- Eve does not receive this segment but to finish the handshake it has to send an ACK segment with Ss + 1 as the ack number
- Eve either eavesdrop the SYN/ACK segment or guesses the correct sequence number Sc

Example



1
138.13.2.67
211.3.56.5
13987
513
seq: 11000
ack: 0
SYN:1
ACK:0
FIN:0
C:117.76.3.3
138.13.2.67
211.3.56.5
13987
513
seq: 11001
ack: 54003
SYN:0
ACK:1
FIN:0
3
A: 211.3.56.5
2
211.3.56.5
138.13.2.67
513
13987
seq: 54002
ack: 11001
SYN:1
ACK:1
FIN:0
0
Denial-of-Service Attack
B:138.13.2.67

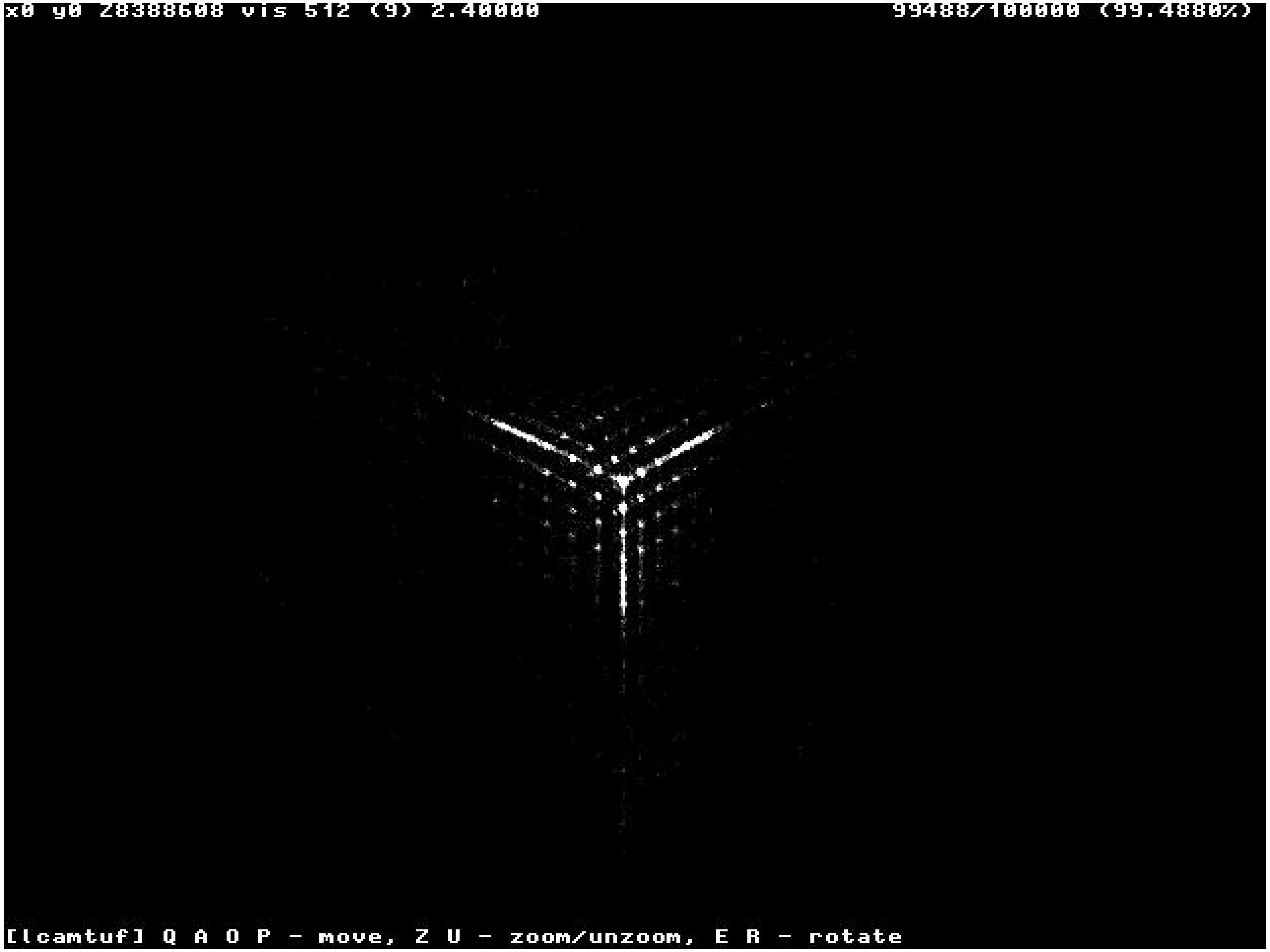
The Kevin Mitnick Attack

- 1992, Kevin Mitnick wanted to access Tsutomu Shimomura's X-Terminal computer
- Shimomura’s terminal was accepting connection only from a trusted IP 125.126.127.128
- Mitnick killed 125.126.127.128 by DOS’ing (we will see later this attack)
- He knew beforehand by “guess and retry” that: $Seq^{th+1} = Seq^{th} + 128\ 000$
- Made a spoofed TCP three way handshake and successfully guessed the correct seq num
- The TCP payload contained: “echo + + >> /.rhost”

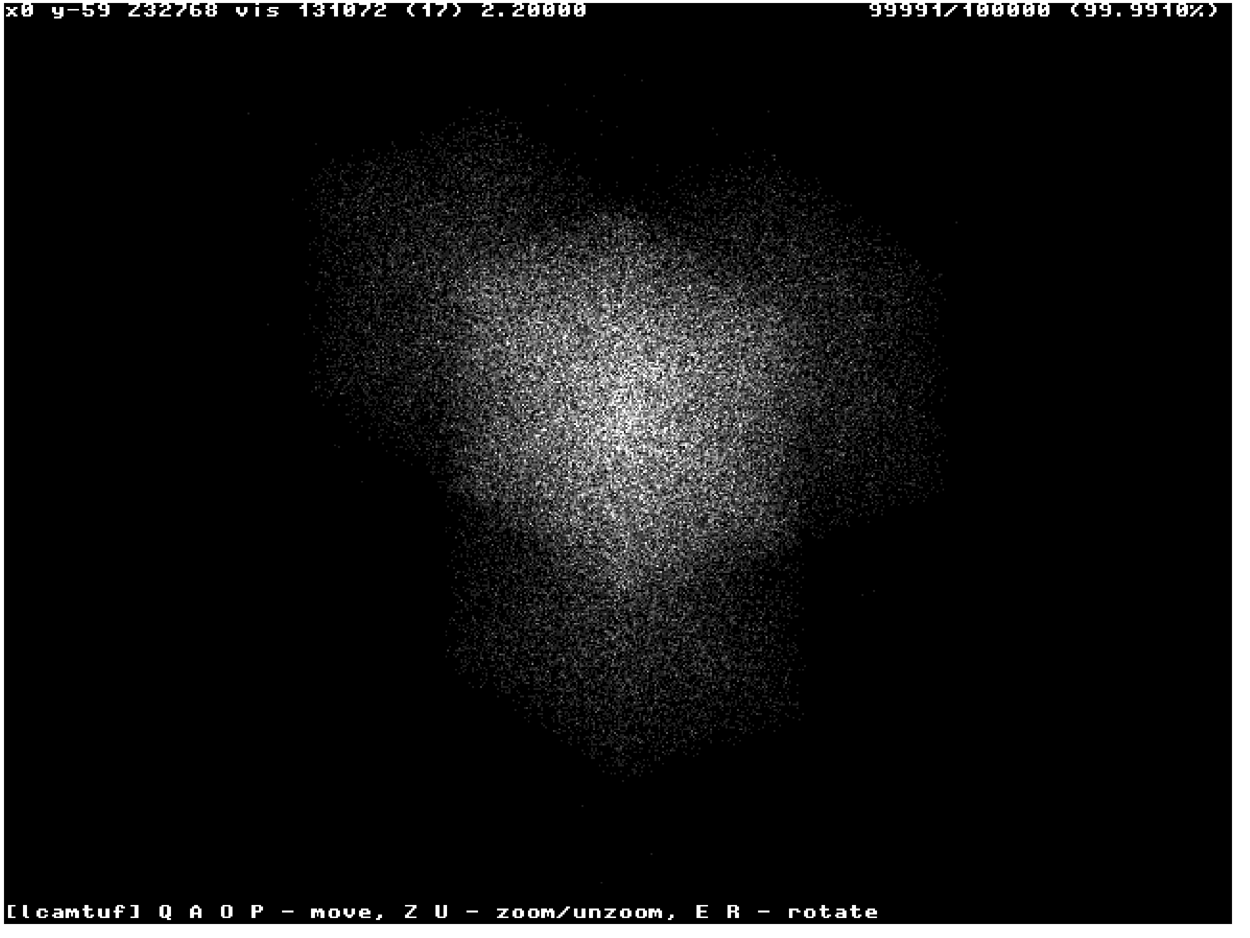
Guess the right Sequence Number

- RFC 1948 defines way to improve sequence number generation
- Some implementations are not compliant / unpredictable
- Michal Zalewski’s paper “Strange Attractors and TCP/IP Sequence Number Analysis” and its update “One Year Later”
- He build a graph using a composition of the values seen recently in a series of sequence numbers:
  - $x[n] = s[n-2] – s[n-3]$
  - $y[n] = s[n-1] – s[n-2]$
  - $z[n] = s[n] - s [n-1]$

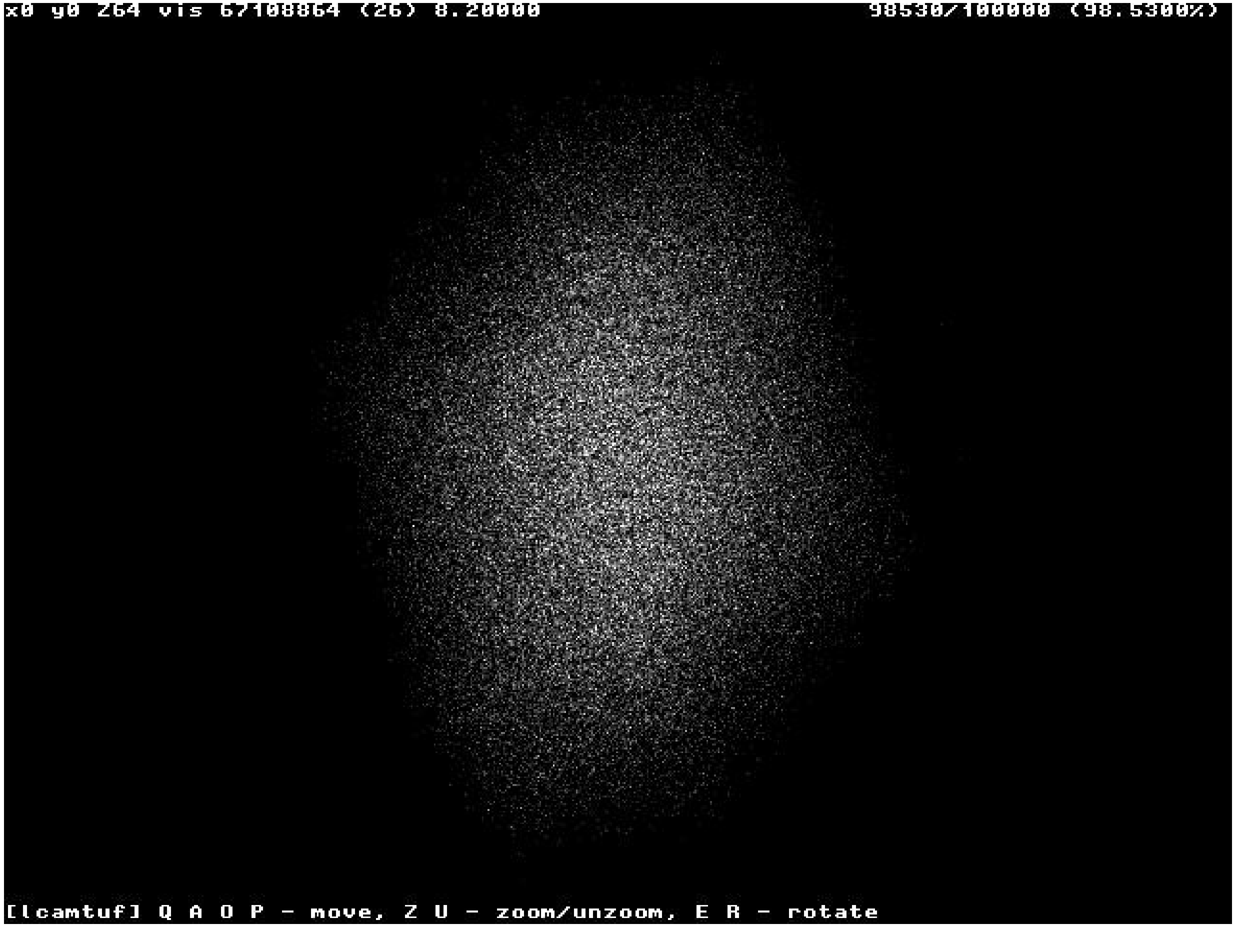
Windows 95


x0 y0 28388608 vis 512 (9) 2.40000
99488/100000 (99.4880%)
[lcamtuf] Q A O P - move, Z U - zoom/unzoom, E R - rotate

Windows 2000 and XP



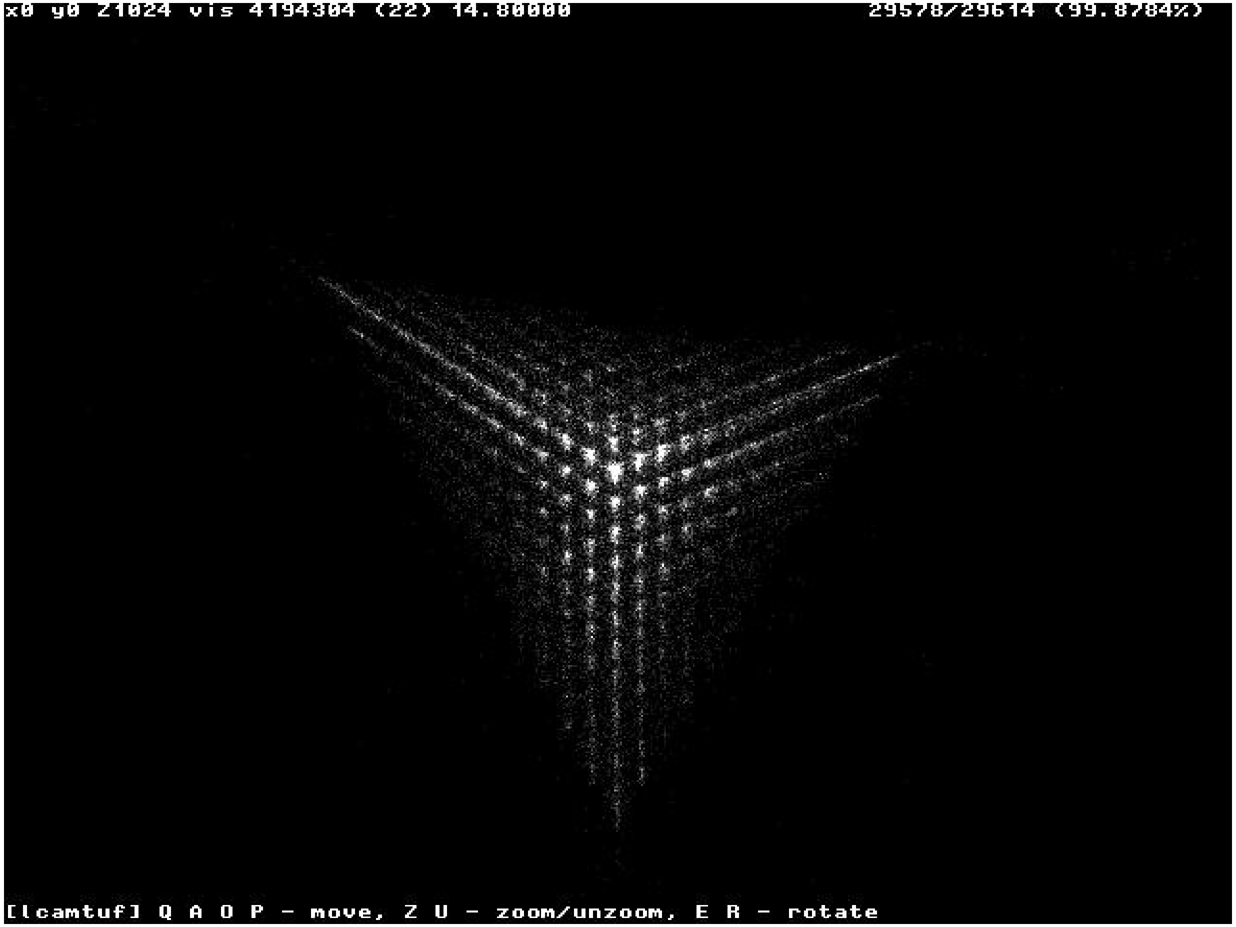
Linux (<Kernel 2.X)



FreeBSD

)0 y0 Z0 vis 2147483648 (33) 0.00000
100000/100000 (100.0000%
[lcamtuf] Q A O P - move, Z U - zoom/unzoom, E R - rotate

Cisco IOS


x0 y0 Z1024 vis 4194304 (22) 14.80000
29578/29614 (99.8784%)
[lcamtuf] Q A O P - move, Z U - zoom/unzoom, E R - rotate

Cisco IOS (one year later)

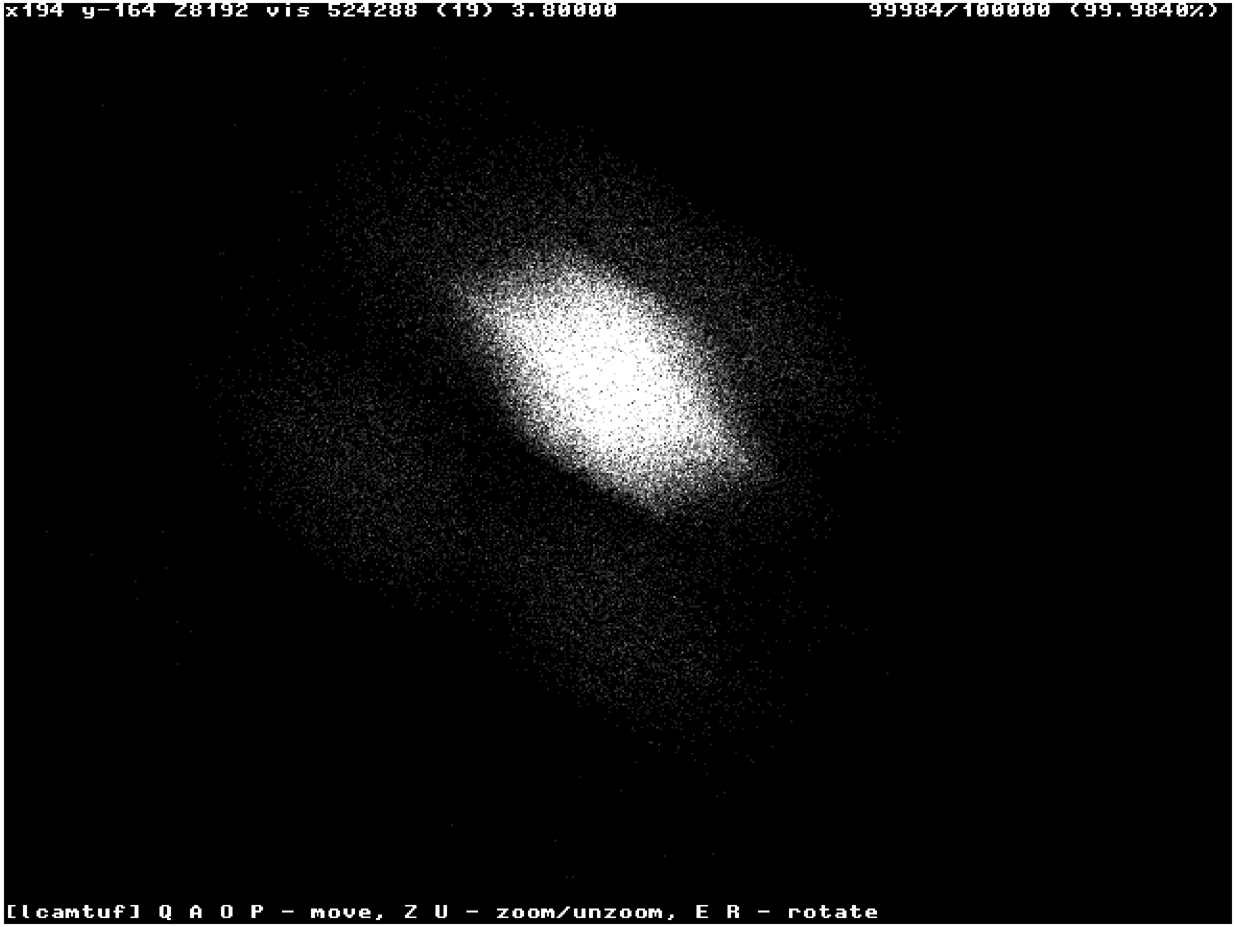
Mac OSX

x0 y0 Z1 vis 2147483648 (32) 0.00000
99996/100000 (99.9960%)
[lcamtuf] Q A O P - move, Z U - zoom/unzoom, E R - rotate

HP-UX


x-119 y-59 24096 vis 1048576 (20) 0.60000
99988/100000 (99.9880%)
[lcamtuf] Q A O P - move, Z U - zoom/unzoom, E R - rotate

HP-UX (one year later)



IRIX


x0 y0 216384 vis 262144 (18) 2.20000
99988/100000 (99.9880%)
[lcamtuf] Q A O P - move, Z U - zoom/unzoom, E R - rotate

IV. Denial of Service

- Making a network resource unavailable to its intended users
- Usually happens by overloading the resources (flooding)
- Could happen by exploiting misconfiguration (crashing)
- Real world example: Protesters crowding Burger King at Palazzo Nuovo

Denial of Service, the easy way

- Wireless networks are particularly vulnerable to DOS attacks
- Manipulation of control frames:
  - Attacker can send a disassociation request to nodes on a wireless network and continue to send disassociation messages whenever they retry
- Frequency interference

Fragmentation Attack

Datagram Fragmentation:

- When a datagram is encapsulated in lower level protocols (e.g., Ethernet) it may be necessary to split the datagram in smaller portions
- This happens when the datagram size is bigger than the data link layer MTU (Maximum Transmission Unit)
- Fragmentation can be performed at the source host or at an intermediate step in datagram delivery
- If the datagram has the “do not fragment” flag set, an ICMP error message is sent back to the originator

Fragmentation Attack #2

- If the datagram can be fragmented:
  - The header is copied in each fragment
    - In particular, the “datagram id” is copied in each fragment
  - The “fragmentation offset” field contains the position of the fragment with respect to the original datagram expressed in 8-byte units
  - The “total length field” is changed to match the size of the fragment
  - Each fragment is then delivered as a separate datagram
  - If one fragment is lost the entire datagram is discarded after a timeout

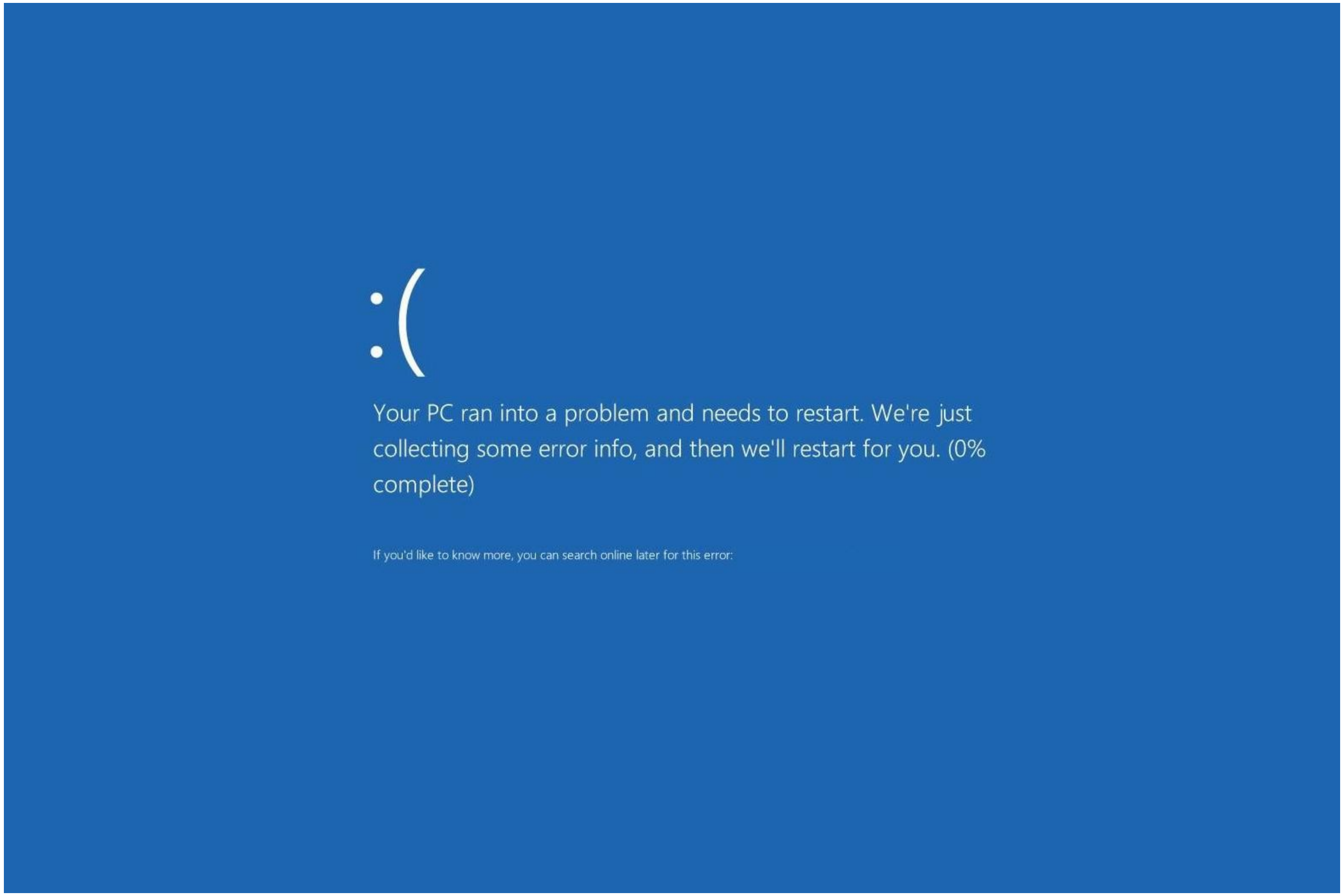
Fragmentation Attack #3

The ping of death:

- The attacker modifies the offset of the last segment such that the total size of the reassembled datagram is bigger than the maximum allowed size
  - A kernel static buffer is overflowed, causing a kernel panic
- In other scenarios fragmentation can be used as a form of evasion because some firewalls don’t reassemble packets

Ping of Death: IPv4 – WinNuke

History repeats itself: IPv6

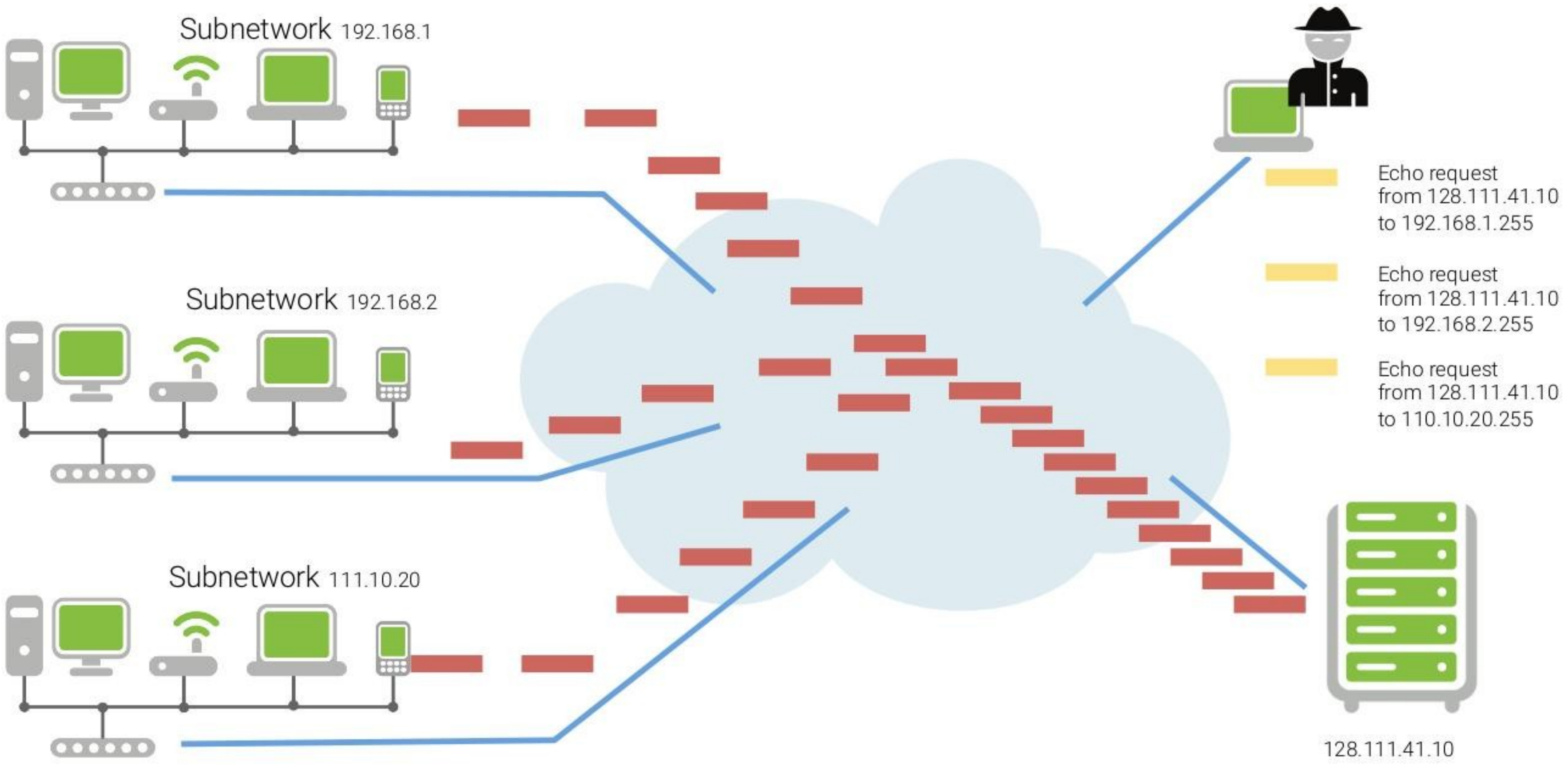


ICMP

Smurf Attack

- 1990, a small attacker versus a crowded network
- Forged ICMP packets with:
  - victim’s spoofed source IP
  - Network broadcast address as destination
- Effective because:
  - Broadcast addresses were in the standards until 1999
  - Routers were accepting packets from the outside even if the IP belonged to an host inside the network
  - A similar attack can be done with UDP

Smurf attack



Exploiting ICMP again

- ICMP defines “destination unreachable” and “redirect” packets
- An attacker forges a ICMP packet that is sent to a router
- The router subsequently reconfigures the routing table
- Traffic gets hijacked and nodes could be cut out from the network

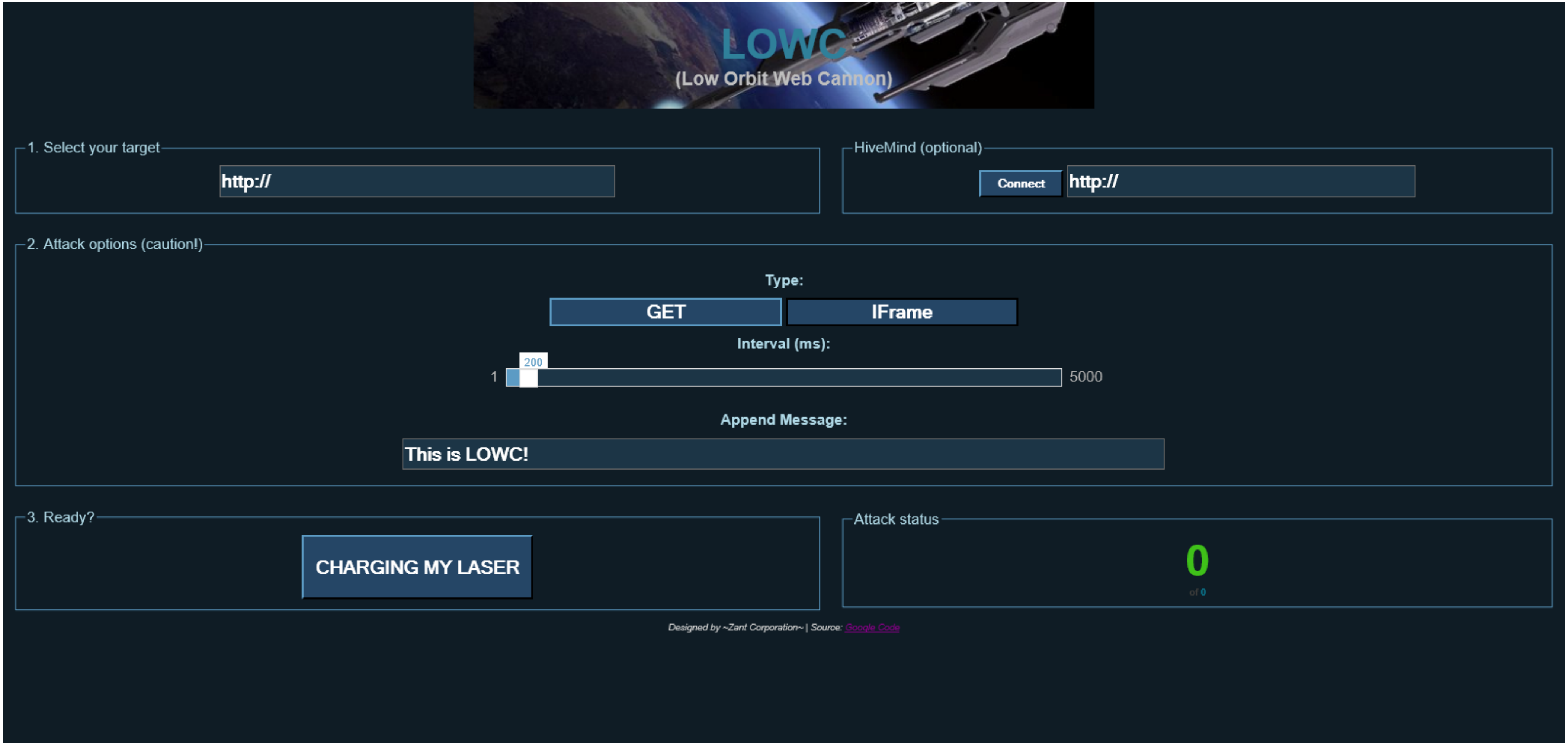
Exploiting the state

- Many protocols are not stateless
- State consumes resources even when the links are idle
  - Memory for the socket descriptor
  - Transactional and pending state
  - Process or thread to manage the connection
  - Memory associated with the data in the TCP stream that has not yet been acknowledged
  - Database and file locking

SYN Flooding

- Attacker starts handshake with SYN-marked segment
- Victim replies with SYN-ACK segment
- Attacker stays silent
  - the source IP of the attacker can be spoofed, since no final ACK is required
  - the attack vector could be a slow link (TOR) because few resources are used
- A host can keep a limited number of TCP connections in half-open state. After that limit, it cannot accept any more connections
- Mitigated by SYN cookies (that requires way less state)

LOIC

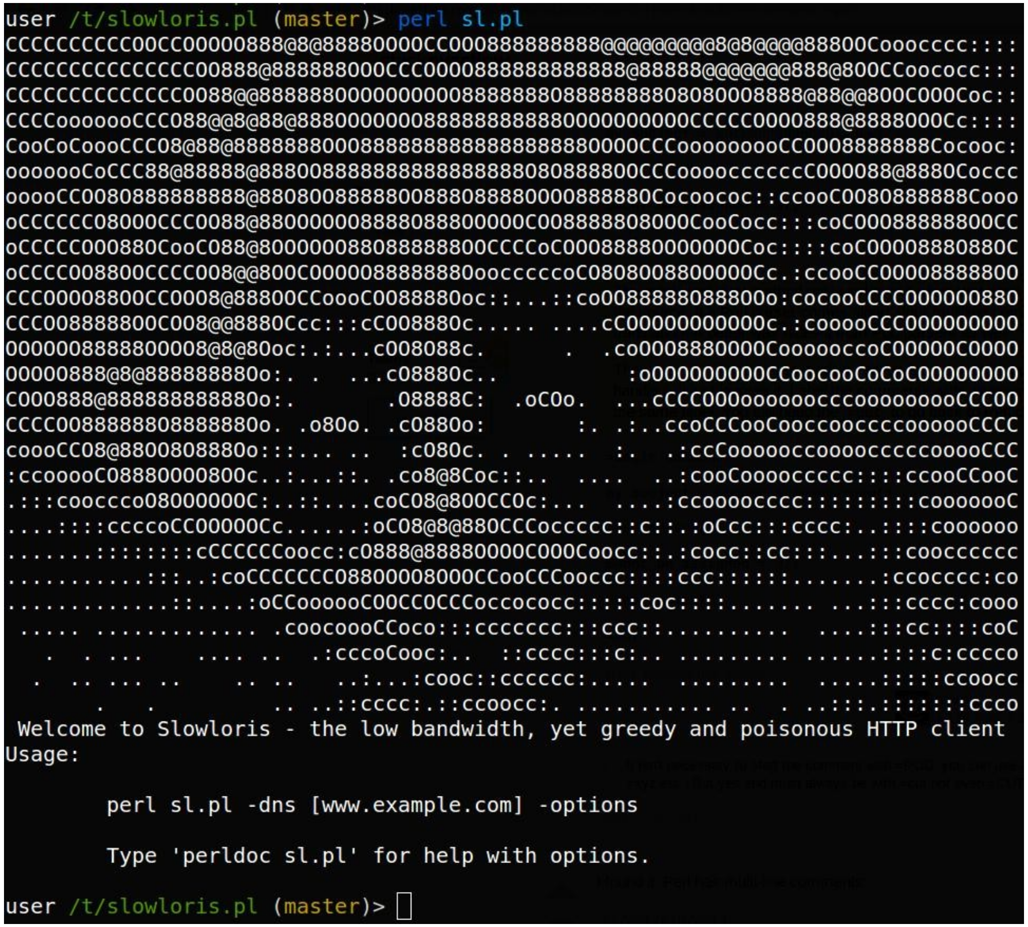


LOWC
(Low Orbit Web Cannon)
1. Select your target
http://
HiveMind (optional)
Connect
http://
2. Attack options (caution!)
Type:
GET
IFrame
Interval (ms):
200
1
5000
Append Message:
This is LOWC!
3. Ready?
CHARGING MY LASER
Attack status
0
of 0
Designed by ~Zant Corporation~ | Source: Google Code

HTTP POST Attack

- Legitimate HTTP POST header
  - “Content-Length” up to 2GB
- The actual message body is transmitted at an extremely slow rate.
- Many of this sessions are opened until logical resources are exhausted
- Difficult to distinguish and filter

SlowLoris


user /t/slowloris.pl (master)> perl sl.pl
Welcome to Slowloris - the low bandwidth, yet greedy and poisonous HTTP client
Usage:
perl sl.pl -dns [www.example.com] -options
Type 'perldoc sl.pl' for help with options.
user /t/slowloris.pl (master)>

Conclusions

Lista di priorità rivista (1988 vs 2008)

1988	2008
1. Sopravvivenza (survivability)	1. Sicurezza
2. Supporto a molteplici tipi di servizi	2. Availability and resilience
3. Possibilità di estensione a grande varietà di reti	3. Convenienza economica
4. Permettere gestione distribuita	4. Migliore gestione
5. Permettere che un host si attacchi alla rete con minimo sforzo	5. Venire incontro ai bisogni della società
6. Efficiente in termini di costi	6. Longevità
7. Permettere monitoraggio e tariffazione delle risorse utilizzate	7. Predisposizione a supportare e sfruttare tecnologie future
	8. Assolvere al suo compito (funziona...)

IP Spoofing

- Used to impersonate sources of security-critical information
- IP spoofing is used to exploit address-based authentication in higher-level protocols
- Many tools available
  - Protocol-specific spoofers (DNS spoofers, NFS spoofers, etc)
  - Generic IP spoofing tools (e.g., hping)
  - Libraries: libnet, scapy

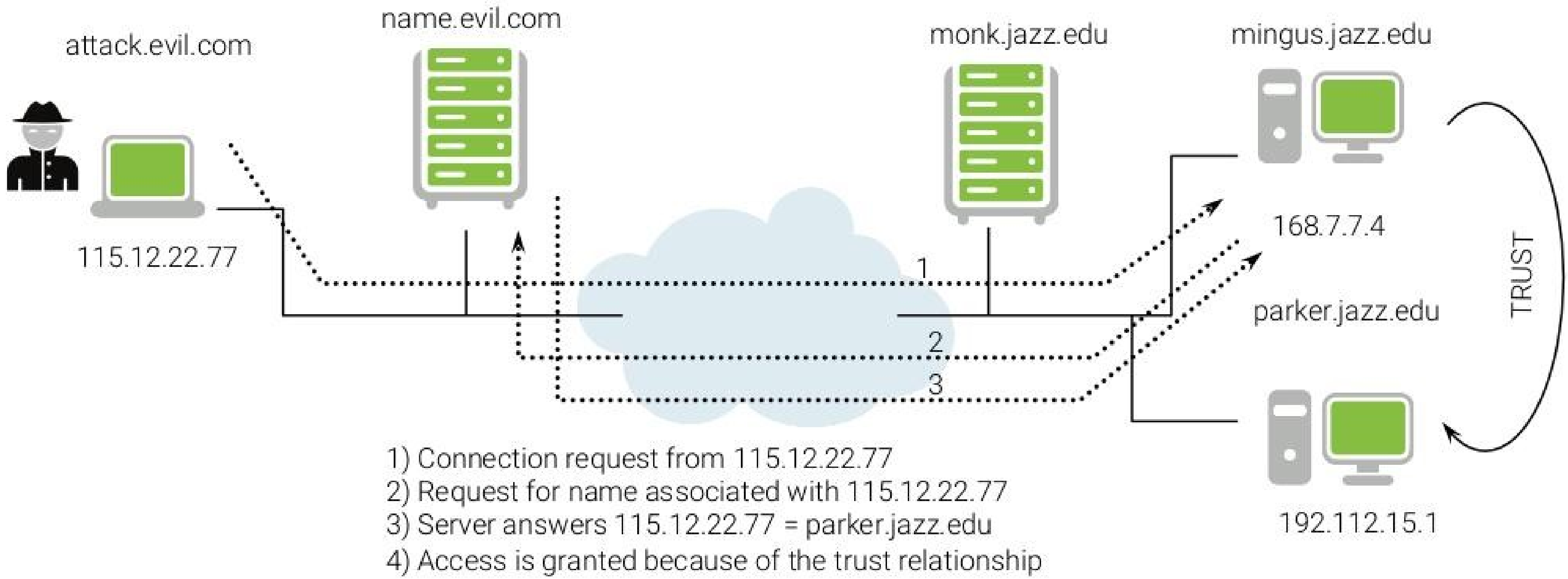
Blind IP Spoofing

- The attacker sends an IP datagram with the address of some other host as the source address
- The attacked host replies to the impersonated host
- Usually the attacker does not have access to the reply traffic
- Can be used to exploit misconfigurations

DNS Spoofing

- Alice and Bob have a trust relationship
- Eve controls a malicious DNS server
- Eve sends a requests to Alice from her IP
- Alice requests the domain name associated to Eve's IP
- Eve's DNS server replies with Bob's domain name
- Access is granted

Example

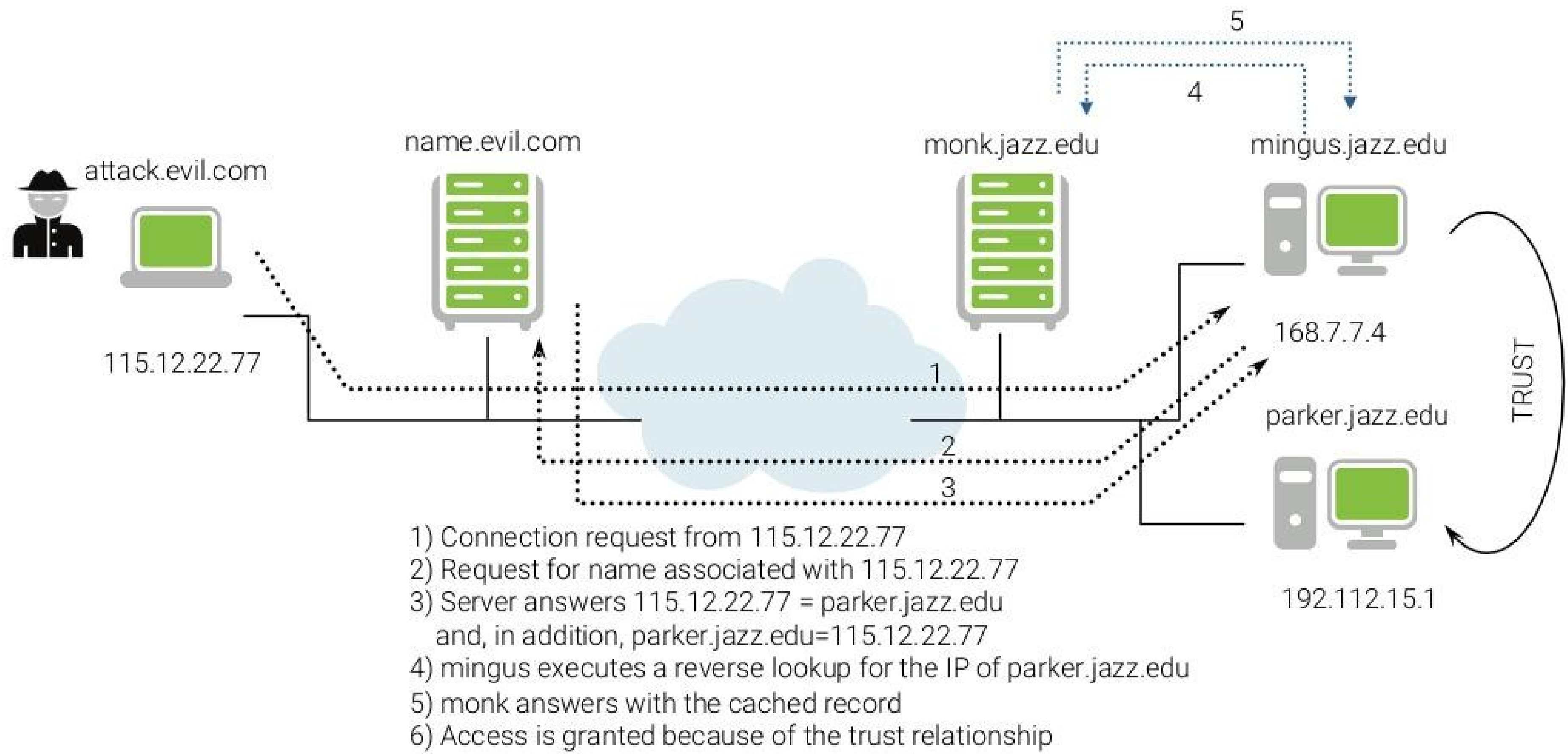


name.evil.com
attack.evil.com
monk.jazz.edu
mingus.jazz.edu
115.12.22.77
168.7.7.4
1
2
3
parker.jazz.edu
TRUST
192.112.15.1
1) Connection request from 115.12.22.77
2) Request for name associated with 115.12.22.77
3) Server answers 115.12.22.77 = parker.jazz.edu
4) Access is granted because of the trust relationship

DNS Spoofing: countermeasure

- Alice could do a double reverse lookup: ask Bob’s authoritative DNS for the real IP and it will get a mismatch with Eve’s IP
- In that scenario Eve could either:
  - poison the DNS cache:
    - Some DNS implementations accept additional commands with a request
  - spoof a UDP packet and race for the reply
    - Techniques for guessing the right ID number

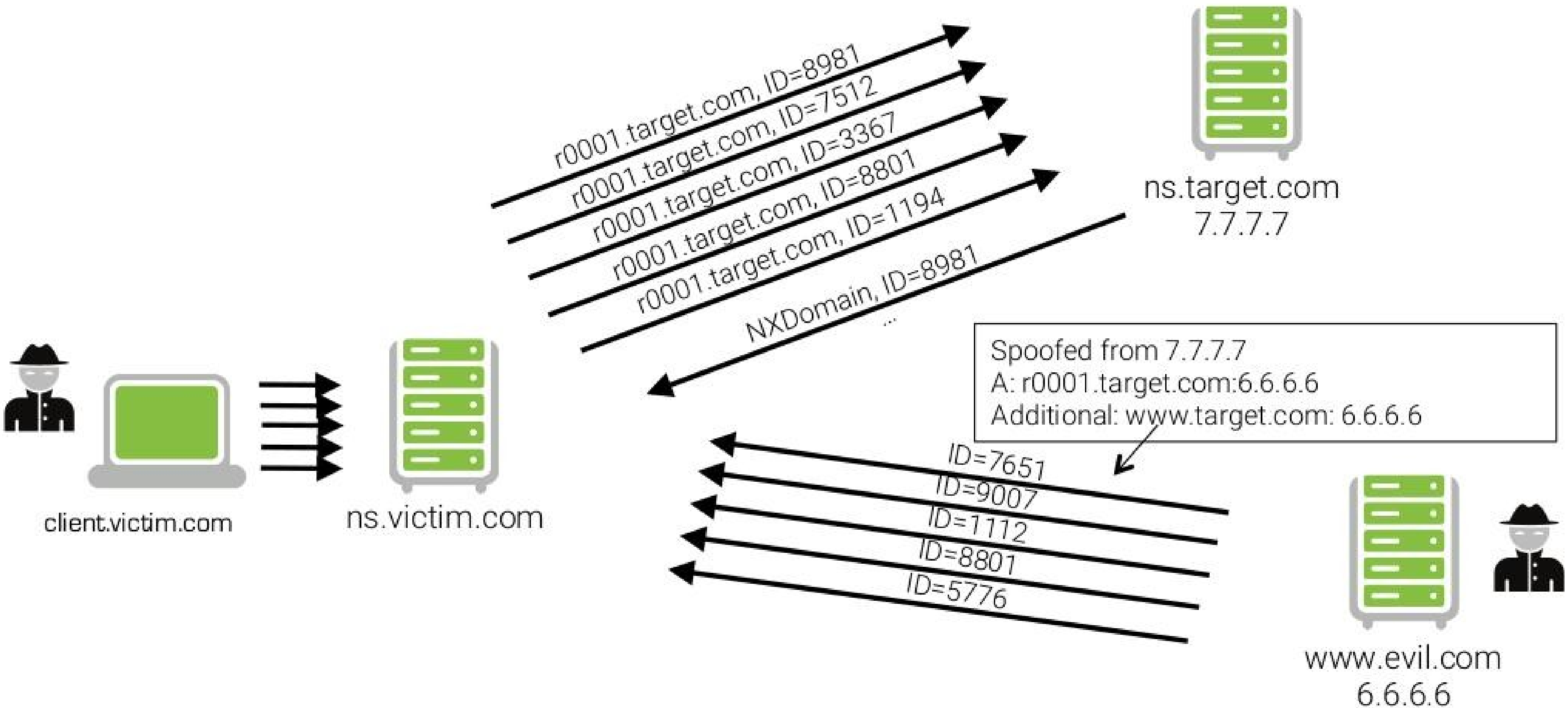
DNS Poisoning



Race for the reply

- Remote DNS cache poisoning through hijacking requires the attacker to guess the 16-bit ID value used to match requests to replies and the source port used in the request
- It can be shown that with ~200 replies, we have 50% possibilities to guess the right ID (Kaminsky attack)
  - ID used to be sequential and it is now random
  - Source port is most of the time not random

Kaminsky Attack



Race and DNS Poisoning

- Remote DNS cache poisoning through hijacking requires the attacker to guess the 16-bit ID value used to match requests to replies and the source port used in the request
- It can be shown that with ~200 replies, we have 50% possibilities to guess the right ID (Kaminsky attack)
  - ID used to be sequential and it is now random
  - Source port is most of the time not random

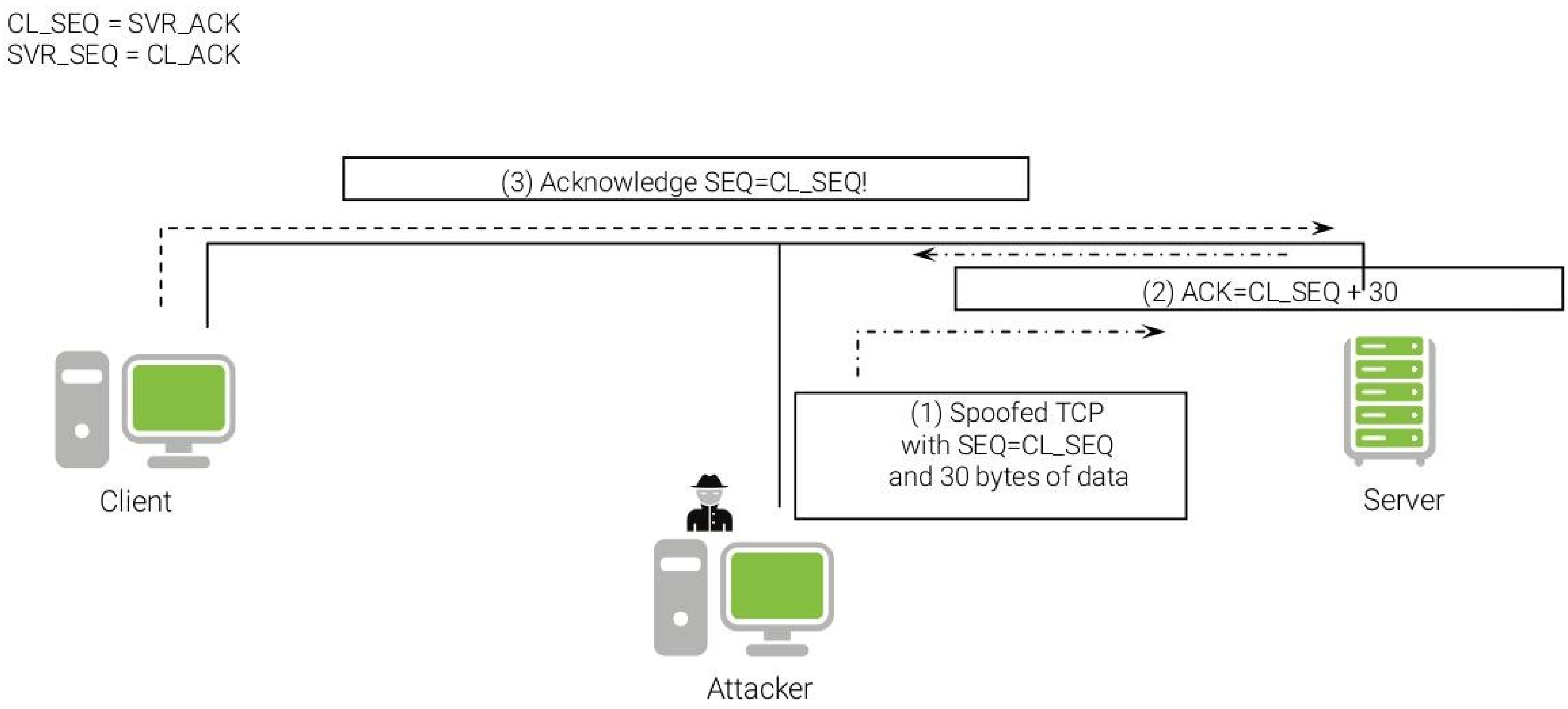
ACK Storm

- The attacker has some knowledge of the state and waits until the connection is “quiet”
  - All the transmitted data have been acknowledged (by both endpoints)
- The attacker injects the data in the stream
  - “Desynchronizes” the connection
- The receiver of the injected data sends an acknowledgment to the apparent sender
- The apparent sender replies with an acknowledgement with the “expected” sequence number
- The receiver considers this as out-of-sync and sends an an acknowledgement with the “expected” sequence number

ACK Storm #2

- ACK messages with no data are not retransmitted in case of loss
- The “ACK storm” continues until one message is lost
- Any subsequent attempt to communicate will generate an ACK storm
- ACK storms can be blocked by the attacker using ACK packets with the right numbers

ACK Storm #3


CL_SEQ = SVR_ACK
SVR_SEQ = CL_ACK
(3) Acknowledge SEQ=CL_SEQ!
(2) ACK=CL_SEQ + 30
(1) Spoofed TCP
with SEQ=CL_SEQ
and 30 bytes of data
Client
Server
Attacker