CSc 466/566

Computer Security

18 : Network Security — Introduction

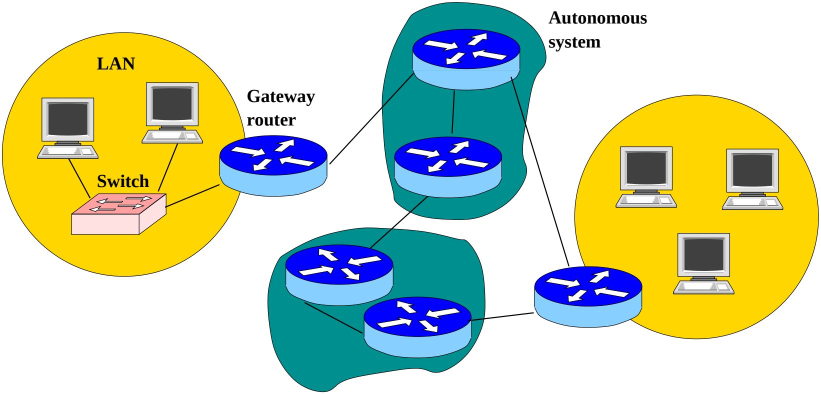
Version: 2013/04/10 11:03:18

Department of Computer Science
University of Arizona

collberg@gmail.com

Christian Collberg

Outline



LAN
Gateway router
Switch
Autonomous system

Network Topology

- Computers are host nodes — they send and receive messages.
- Routers are communication nodes — they pass on messages.
- Local Area Network (LAN) — private network of physically close computers.
- Wide Area Network (WAN) — many physically separated machines/groups of machines.
- Autonomous Systems (AS) — clusters of routers.

Autonomous Systems

- Controlled by a single organizational entity.
- Consist of clusters of routers.
- Routing within an AS is done by shortest route.
- Routing between ASs is by contractual agreements.

Internet Protocol Layers — Physical Layer

- Describes how bitstreams are transferred from one node to another over a physical medium.

Internet Protocol Layers — Physical Layer

- Describes how bitstreams are transferred from one node to another over a physical medium.
- Abstraction:

Internet Protocol Layers — Physical Layer

- Describes how bitstreams are transferred from one node to another over a physical medium.
- Abstraction:
  1. Source/Destination: networking hardware

Internet Protocol Layers — Physical Layer

- Describes how bitstreams are transferred from one node to another over a physical medium.
- Abstraction:
  1. Source/Destination: networking hardware
  2. Data: raw bits

Internet Protocol Layers — Physical Layer

- Describes how bitstreams are transferred from one node to another over a physical medium.
- Abstraction:
  1. Source/Destination: networking hardware
  2. Data: raw bits
  3. Link: copper, coaxial, optical fiber, WiFi...

Internet Protocol Layers — Link Layer

- Describes how collections of bits (frames) are transferred (on top of the physical layer) in a LAN.

Internet Protocol Layers — Link Layer

- Describes how collections of bits (frames) are transferred (on top of the physical layer) in a LAN.
- Abstraction:

Internet Protocol Layers — Link Layer

- Describes how collections of bits (frames) are transferred (on top of the physical layer) in a LAN.
- Abstraction:
  1. Source/Destination: LAN nodes

Internet Protocol Layers — Link Layer

- Describes how collections of bits (frames) are transferred (on top of the physical layer) in a LAN.
- Abstraction:
  1. Source/Destination: LAN nodes
  2. Data: frames

Internet Protocol Layers — Link Layer

- Describes how collections of bits (frames) are transferred (on top of the physical layer) in a LAN.
- Abstraction:
    1. Source/Destination: LAN nodes
    2. Data: frames
    3. Link: Ethernet, Wireless

Internet Protocol Layers — Link Layer

- Describes how collections of bits (frames) are transferred (on top of the physical layer) in a LAN.
- Abstraction:
  1. Source/Destination: LAN nodes
  2. Data: frames
  3. Link: Ethernet, Wireless
  4. Addressing: Media Access Control Addresses (MAC).

Internet Protocol Layers — Link Layer

- Describes how collections of bits (frames) are transferred (on top of the physical layer) in a LAN.
- Abstraction:
  1. Source/Destination: LAN nodes
  2. Data: frames
  3. Link: Ethernet, Wireless
  4. Addressing: Media Access Control Addresses (MAC).
- Detects errors occurring in the physical layer.

Internet Protocol Layers — Link Layer

- Describes how collections of bits (frames) are transferred (on top of the physical layer) in a LAN.
- Abstraction:
  1. Source/Destination: LAN nodes
  2. Data: frames
  3. Link: Ethernet, Wireless
  4. Addressing: Media Access Control Addresses (MAC).
- Detects errors occurring in the physical layer.
- Finds a good routing path in the network.

Internet Protocol Layers — Network (Internet) Layer

- Describes how to move packets between any two hosts on the Internet.

Internet Protocol Layers — Network (Internet) Layer

- Describes how to move packets between any two hosts on the Internet.
- Abstraction:

Internet Protocol Layers — Network (Internet) Layer

- Describes how to move packets between any two hosts on the Internet.
- Abstraction:
  1. Source/Destination: Internet nodes

Internet Protocol Layers — Network (Internet) Layer

- Describes how to move packets between any two hosts on the Internet.
- Abstraction:
  1. Source/Destination: Internet nodes
  2. Data: IP packets

Internet Protocol Layers — Network (Internet) Layer

- Describes how to move packets between any two hosts on the Internet.
- Abstraction:
  1. Source/Destination: Internet nodes
  2. Data: IP packets
  3. Addressing: Internet Protocol (IP) addresses.

Internet Protocol Layers — Network (Internet) Layer

- Describes how to move packets between any two hosts on the Internet.
- Abstraction:
  1. Source/Destination: Internet nodes
  2. Data: IP packets
  3. Addressing: Internet Protocol (IP) addresses.
- IPv4 — 32-bit addresses, IPv6 — 128-bit addresses.

Internet Protocol Layers — Network (Internet) Layer

- Describes how to move packets between any two hosts on the Internet.
- Abstraction:
  1. Source/Destination: Internet nodes
  2. Data: IP packets
  3. Addressing: Internet Protocol (IP) addresses.
- IPv4 — 32-bit addresses, IPv6 — 128-bit addresses.
- Best effort delivery — no guarantees a packet will be delivered.

Internet Protocol Layers — Transport Layer

- Describes how to communicate between two applications (services) running on hosts on the Internet.

Internet Protocol Layers — Transport Layer

- Describes how to communicate between two applications (services) running on hosts on the Internet.
- Abstraction:

Internet Protocol Layers — Transport Layer

- Describes how to communicate between two applications (services) running on hosts on the Internet.
- Abstraction:
  1. Source/Destination: Ports connected to processes

Internet Protocol Layers — Transport Layer

- Describes how to communicate between two applications (services) running on hosts on the Internet.
- Abstraction:
  1. Source/Destination: Ports connected to processes
  2. Data: TCP/UDP packets

Internet Protocol Layers — Transport Layer

- Describes how to communicate between two applications (services) running on hosts on the Internet.
- Abstraction:
  1. Source/Destination: Ports connected to processes
  2. Data: TCP/UDP packets
  3. Addressing: IP address + port number

Internet Protocol Layers — Transport Layer

- Describes how to communicate between two applications (services) running on hosts on the Internet.
- Abstraction:
  1. Source/Destination: Ports connected to processes
  2. Data: TCP/UDP packets
  3. Addressing: IP address + port number
- Transmission Control Protocol (TCP) — connection-based protocol; guaranteed and ordered delivery of packets.

Internet Protocol Layers — Transport Layer

- Describes how to communicate between two applications (services) running on hosts on the Internet.
- Abstraction:
  1. Source/Destination: Ports connected to processes
  2. Data: TCP/UDP packets
  3. Addressing: IP address + port number
- Transmission Control Protocol (TCP) — connection-based protocol; guaranteed and ordered delivery of packets.
- User Datagram Protocol (UDP) — connection-less protocol; quick delivery without guarantees.

Internet Protocol Layers — Application Layer

- Uses the transport layer to provide protocols that support useful functions on the Internet

Internet Protocol Layers — Application Layer

- Uses the transport layer to provide protocols that support useful functions on the Internet
- Examples:

Internet Protocol Layers — Application Layer

- Uses the transport layer to provide protocols that support useful functions on the Internet
- Examples:
    1. HTTP — web browsing over TCP

Internet Protocol Layers — Application Layer

- Uses the transport layer to provide protocols that support useful functions on the Internet
- Examples:
  1. HTTP — web browsing over TCP
  2. DNS — domain name lookup over UDP

Internet Protocol Layers — Application Layer

- Uses the transport layer to provide protocols that support useful functions on the Internet
- Examples:
  1. HTTP — web browsing over TCP
  2. DNS — domain name lookup over UDP
  3. SMTP/IMAP — email over TCP

Internet Protocol Layers — Application Layer

- Uses the transport layer to provide protocols that support useful functions on the Internet
- Examples:
  1. HTTP — web browsing over TCP
  2. DNS — domain name lookup over UDP
  3. SMTP/IMAP — email over TCP
  4. SSL — encrypted connections over TCP

Internet Protocol Layers — Application Layer

- Uses the transport layer to provide protocols that support useful functions on the Internet
- Examples:
  1. HTTP — web browsing over TCP
  2. DNS — domain name lookup over UDP
  3. SMTP/IMAP — email over TCP
  4. SSL — encrypted connections over TCP
  5. VoIP — Internet telephony over UDP.

Network Packets

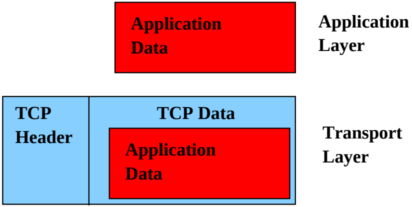
- A packet consists of:
  1. A header (metadata)
  2. Payload (actual data)
  3. A footer (metadata, sometimes)
- Metadata — routing and control information.

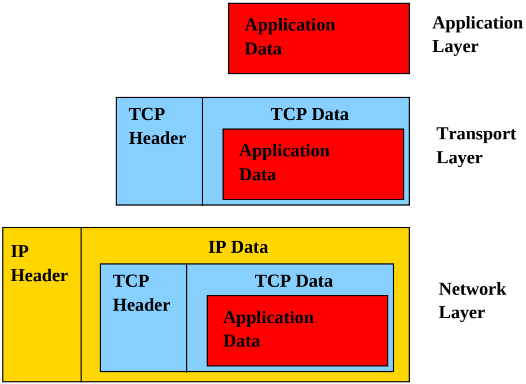
Packet Encapsulation

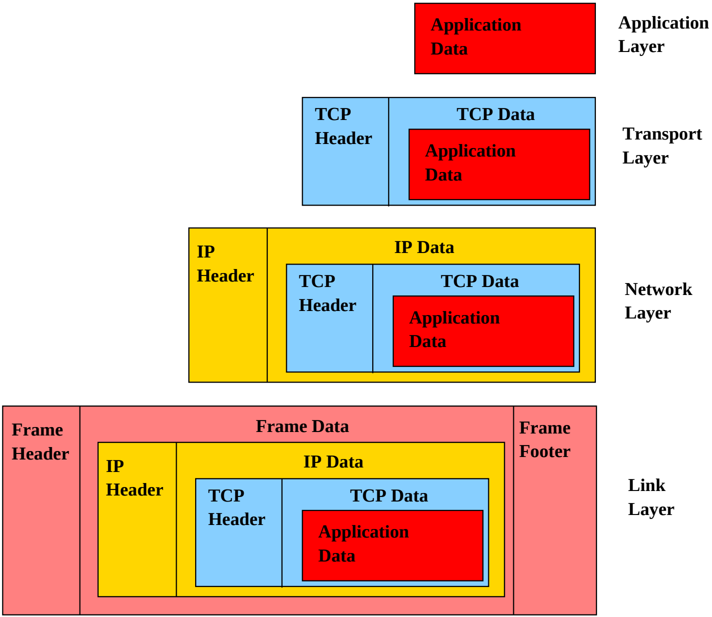
- The payload of each packet encapsulates the packet of a higher layer:
    1. A frame packet encapsulates an IP packet.
    2. An IP packet encapsulates a TCP/UDP packet.
    3. A TCP packet encapsulates application data.

Application Data

Application Layer


Application Data
Application Layer
TCP Header
TCP Data
Application Data
Transport Layer


Application
Data
Application
Layer
TCP
Header
TCP Data
Application
Data
Transport
Layer
IP
Header
IP Data
TCP
Header
TCP Data
Application
Data
Network
Layer


Application Data
Application Layer
TCP Header
TCP Data
Application Data
Transport Layer
IP Header
IP Data
TCP Header
TCP Data
Application Data
Network Layer
Frame Header
Frame Data
Frame Footer
IP Header
IP Data
TCP Header
TCP Data
Application Data
Link Layer

Packet Encapsulation — HTTP

- When Web browsing:
  1. An HTTP packet would be contained in a TCP packet.
  2. The TCP packet would be contained IP packet.
  3. The IP packet would be contained in (for example) an Ethernet frame.

Networking Examples

- OSI model animation: http://www.youtube.com/watch?v=fiMswfo45DQ
- Animation - Networking Tutorial:

  http://www.youtube.com/watch?v=xV-Qq0aHs1o

Network Security Issues — Confidentiality

- Packet data is not kept confidential.
- Two solutions:
  1. Encrypt data at the application level (`https`);
  2. Revise lower level protocol to include encryption (`IPsec`).

Network Security Issues — Integrity

- Packet header/footers include simple checksums:
  - can detect a few communication bit errors;
  - not cryptographically strong.
- Two solutions:
  1. MACs at the application level;
  2. Revise lower level protocol.

Network Security Issues — Availability

- Denial of Service attacks:
  - could be just Christmas rush on `amazon.com`!
  - concerted attacks.
- Two solutions:
  1. Applications need to scale with communication requests;
  2. Block illegitimate requests.

Network Security Issues — Assurance

- Assurance is the way in which trust is provided and managed in a system.
- Packets can travel between any two nodes in a network.
- Solution:
  1. If we want to control packet flow, permissions have to be added on top of the network.
- Example:
  - Firewalls — allows us to block flows of packets we don't trust from entering our system.

Network Security Issues — Authenticity

- Packets have no space for digital signatures!
- IP has no concept of identity.
- Two solutions:
  1. Add signatures at application layer;
  2. Revise lower level layers.

Network Security Issues — Anonymity

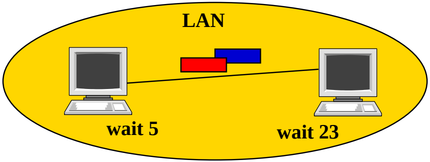
- No concept of identity on the Internet — anonymous by default!
- Good for human rights worker.
- Not good when we can't identify a malicious user.
- Solutions:
    1. Achieve higher level of anonymity by replicating processes in many places on the network.

Outline

The Link Layer

- The Link Layer sits on top of the physical layer.
- Ethernet — IEEE 802.3.
- Ethernet cables connect computers on a LAN.
- Collision: Two computers on the same network segment send a packet at the same time.
- History of Ethernet: http://www.youtube.com/watch?v=g5MezxMcRmk.

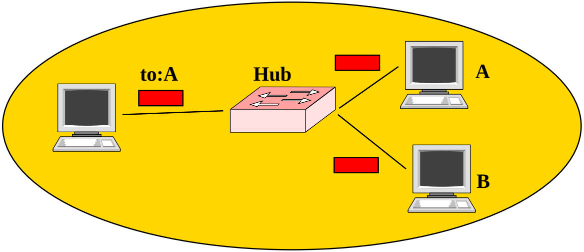
Ethernet Collision

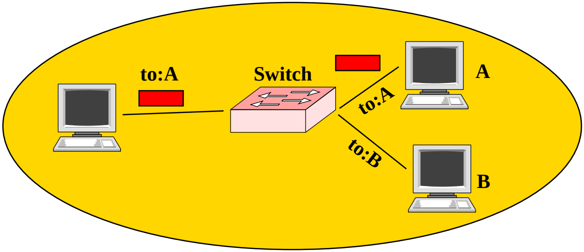


- Collision algorithm:
  1. Each computer waits a random length of time;
  2. Retransmit!
  3. Another collision? Repeat from 1!

Hubs and Switches

- Hubs and Switches connect devices on a LAN.
- Ethernet Hub:
  - Forward all frames to all attached devices.
  - Lots of extra traffic: all frames are duplicated!
  - All devices are on the same network segment, and must do collision avoidance.
- Ethernet Switch:
  - Initially works like a hub.
  - Over time, learns the addresses of attached devices.
  - Eventually, only forwards a frame to the destination device.
  - Fewer collisions.


to:A
Hub
A
B


to:A
Switch
A
to:A
to:B
B

MAC Addresses

- MAC address: 48 bits assigned to network interface.
- MAC structure:

locally assigned (1 bit)	manufacturer (23 bits)	unique number (24 bits)

- Software (Unix: `ifconfig`) can change a device's MAC: *locally assigned*=1.

Ethernet Frame Format

Preamble (7 bytes)
Start-of-Frame delimiter (1 byte)
MAC destination (6 bytes)
MAC source (6 bytes)
Ethertype/length (2 bytes)
Payload (45-1500 bytes)
CRC-32 Checksum (4 bytes)
Interframe Gap (12 bytes)

Ethernet Frame Format...

- The CRC-32 checksum can catch simple transmission errors.
- Switches learn the location of network devices from the MAC addresses.

Address Resolution Protocol

- Address Resolution Protocol (ARP): Find the MAC address given the IP address.
- Algorithm (Bob wants to know the MAC address of IP address A):
  1. Broadcast to all network interfaces: Who has IP address A?.
  2. Wait for a response A is at MAC address M! from the devices with IP address A.
  3. Store $A \leftrightarrow M$ in the ARP cache.
- Problem: no authentication.

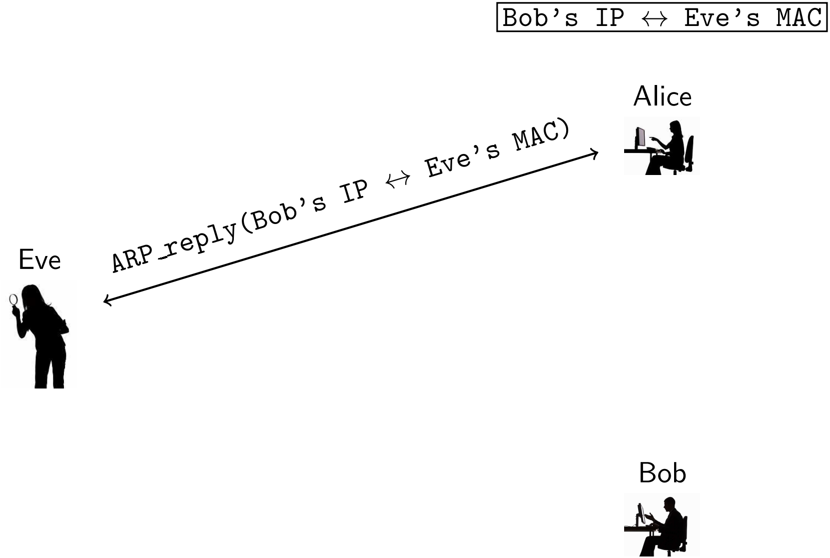
ARP Spoofing

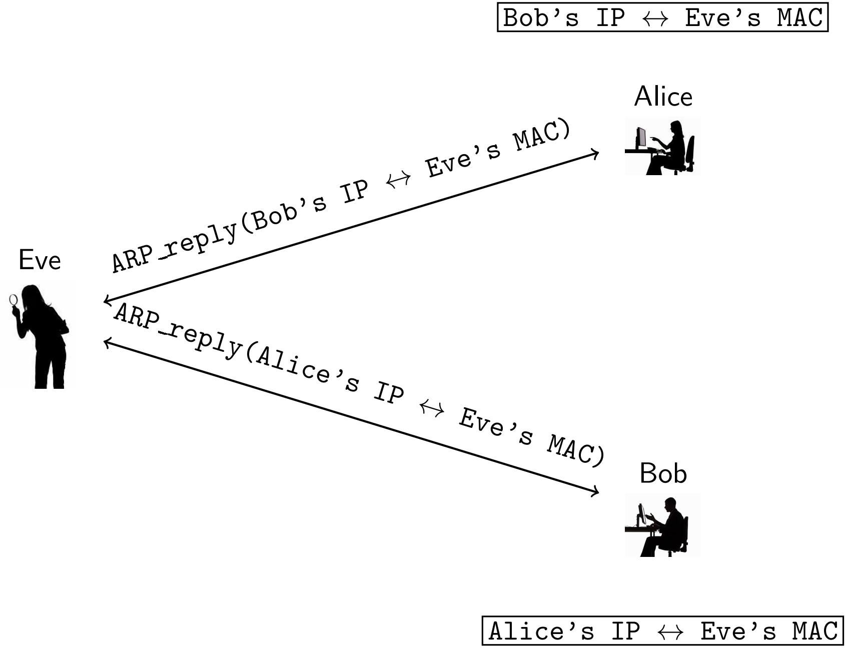
- Any computer on the network could claim to have a particular IP address.
- Machines will update their ARP cache whenever they see an ARP reply — even if there was no corresponding ARP request!
- Attack:
  1. Eve sends `ARP_reply(Bob's IP ↔ Eve's MAC)` to Alice.
  2. Alice puts `Bob's IP ↔ Eve's MAC` in her ARP cache.
  3. Eve sends `ARP_reply(Alice's IP ↔ Eve's MAC)` to Bob.
  4. Bob puts `Alice's IP ↔ Eve's MAC` in his ARP cache.

Alice

Eve

Bob


Bob's IP ↔ Eve's MAC
Alice
ARP_reply(Bob's IP ↔ Eve's MAC)
Eve
Bob


Bob's IP ↔ Eve's MAC
Alice
ARP_reply(Bob's IP ↔ Eve's MAC)
Eve
ARP_reply(Alice's IP ↔ Eve's MAC)
Bob
Alice's IP ↔ Eve's MAC

ARP Spoofing...

- After the ARP cache poisoning all traffic between Alice and Bob is routed through Eve:
  1. MITM attack;
  2. Denial of Service attack.

ARP Spoofing — Countermeasures

1. Restrict LAN access to trusted users.

ARP Spoofing — Countermeasures

1. Restrict LAN access to trusted users.
2. Check for multiple occurrences of the same MAC address on the LAN.

ARP Spoofing — Countermeasures

1. Restrict LAN access to trusted users.
2. Check for multiple occurrences of the same MAC address on the LAN.
3. Static ARP tables: the system adminstrator manually sets up the routers' ARP caches.

ARP Spoofing — Countermeasures

1. Restrict LAN access to trusted users.
2. Check for multiple occurrences of the same MAC address on the LAN.
3. Static ARP tables: the system adminstrator manually sets up the routers' ARP caches.
4. Inspect all ARP packets, detecting attempted spoofing.

Outline

The Network (Internet) Layer

- Best effort routing of packets between any two hosts on the Internet.
- Abstraction:
  1. Source/Destination: Internet nodes
  2. Data: IP packets
  3. Addressing: Internet Protocol (IP) addresses.
- IPv4 — 32-bit addresses, IPv6 — 128-bit addresses.
- No guarantees a packet will be delivered.

Routing Algorithm — From a Host Node

- Sending a packet P from a host node N:
  1. If P's destination is on this LAN:
     - Use the ARP protocol to find the MAC address,
     - deliver directly.
  2. Otherwise:
     - use the ARP protocol to find the MAC address of the gateway,
     - forward.

Routing Algorithm — From a Router

- Router — gateways and other network nodes that handle routing of packages on the Internet.
- A router typically connects two or more LANs.
- Routing tables describe the next router to which a packet should be forwarded.

Router Operations

- For each packet, the router decides whether to
  1. Drop — expired packets (TTL=0) are dropped.
  2. Deliver — if the packet is going to a machine on this LAN, deliver it.
  3. Forward — otherwise, send to neighboring router.
- TTL (time to live): a field in the IP header, decremented by each router, used to prevent packets from living forever.

Routing Table Protocols

- Open Shortest Path First (OSPF) — how should packets be routed *within* an autonomous system?
  - packets should travel along shortest paths.
- Border Gateway Protocol (BGP) — how should packets be routed *between* autonomous systems?
  - packets are routed based on contractual agreements.
- Routing animation: http://www.youtube.com/watch?v=RbY8Hb6abbg

Routing vs. Switch

- Switch:
  - forwards packets on a single LAN.
  - learns routes over time.
- Router:
  - can belong to multiple LANs.
  - uses routing tables to forward packets.

IPv4 Packet Format

Version (4 bits)
Header length (4 bits)
Service type (8 bits)
Total length (16 bits)
Identification (16 bits)
Flags (3 bits)
Fragment offset (13 bits)
Time-to-Live (8 bits)
Protocol (8 bits)
Header Checksum (16 bits)
Source Address (32 bits)
Destination Address (32 bits)
Payload

IP Address Format

- IPv4 address: 32 bits.
- IPv4 address structure:

network portion	host portion

- Network portion: IP prefix for all machines on a network.
- Host portion: identifies a particular device
- Peter Packet & Subnetting:

  http://www.youtube.com/watch?v=x-QC6l9KhQY&feature=related

- Class A — Reserved for government organizations, telcos.
- Class B — Reserved for ISPs, large businesses.
- Class C — Reserved for smaller organizations.

IP Address Classes

Class	Leading bits	Size of network number bit field	Size of rest bit field	Number of networks	Addresses per network
A	0	8	24	2^7	2^{24}
B	10	16	16	2^{14}	2^{16}
C	110	24	8	2^{21}	2^8

Class	Start address	End address
A	0.0.0.0	127.255.255.255
B	128.0.0.0	191.255.255.255
C	192.0.0.0	223.255.255.255

Internet Control Message Protocol

- Internet Control Message Protocol (ICMP) — used for network diagnostics.
- ICMP messages:
  1. Echo request: please acknowledge receipt of packet.

Internet Control Message Protocol

- Internet Control Message Protocol (ICMP) — used for network diagnostics.
- ICMP messages:
  1. Echo request: please acknowledge receipt of packet.
  2. Echo response: packet receipt is acknowledged.

Internet Control Message Protocol

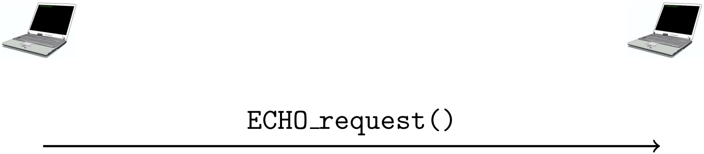
- Internet Control Message Protocol (ICMP) — used for network diagnostics.
- ICMP messages:
  1. Echo request: please acknowledge receipt of packet.
  2. Echo response: packet receipt is acknowledged.
  3. Time exceeded: notify that packet has expired (TTL=0).

Internet Control Message Protocol

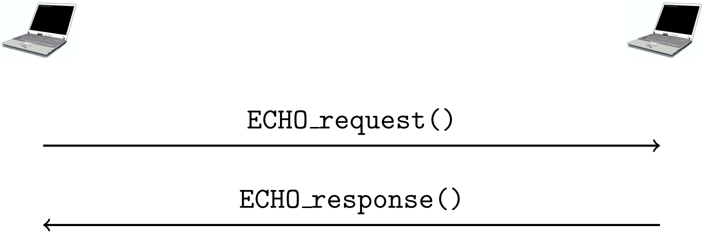
- Internet Control Message Protocol (ICMP) — used for network diagnostics.
- ICMP messages:
  1. Echo request: please acknowledge receipt of packet.
  2. Echo response: packet receipt is acknowledged.
  3. Time exceeded: notify that packet has expired (TTL=0).
  4. Destination unreachable: notify that packet could not be delivered.

Ping Protocol

Ping Protocol



Ping Protocol



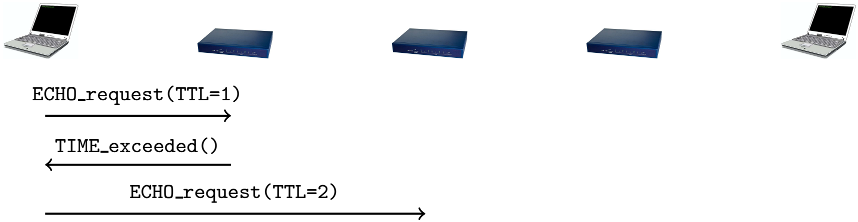
- Diagnostic tool too see if a host is working.

Traceroute Protocol

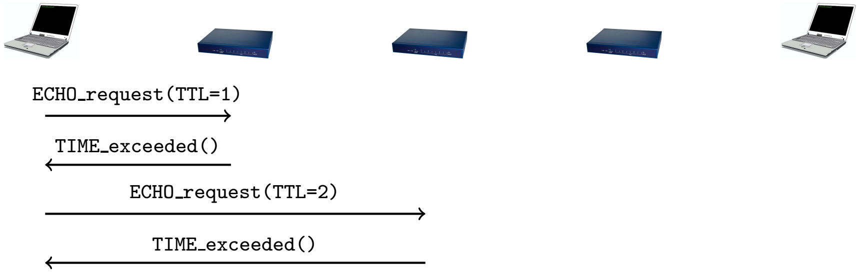
- How do we find the path a packet takes to a node N?
- Algorithm:
  1. Send `ECHO_request(TTL=1)` to N.
  2. A router that receives `ECHO_request(TTL=1)` responds with `TIME_exceeded()`.
  3. Send `ECHO_request(TTL=2)` to N.
  4. Repeat, increasing TTL each time, until N is reached, responding with `ECHO_response()`.


ECHO_request(TTL=1)

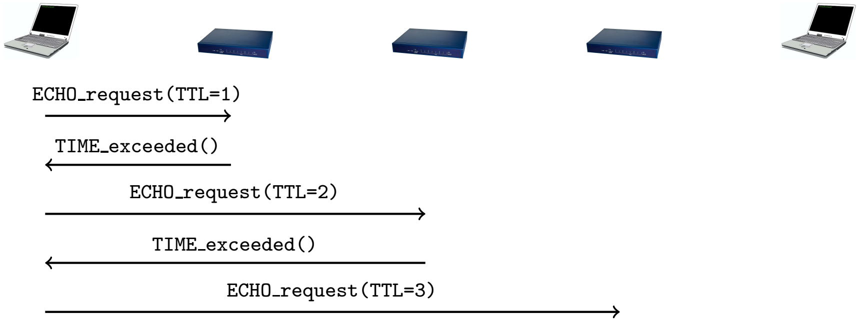

ECHO_request(TTL=1)
TIME_exceeded()



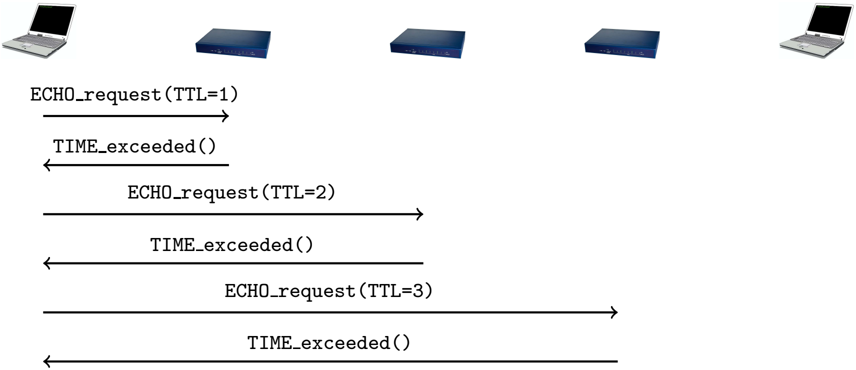
ECHO_request(TTL=1)
TIME_exceeded()
ECHO_request(TTL=2)

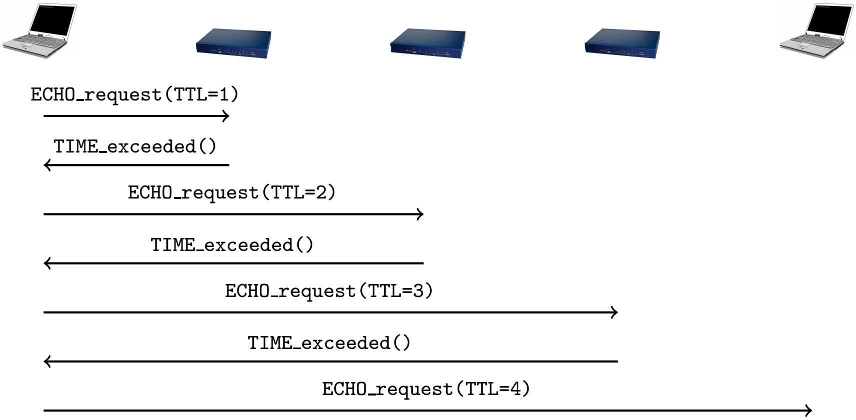


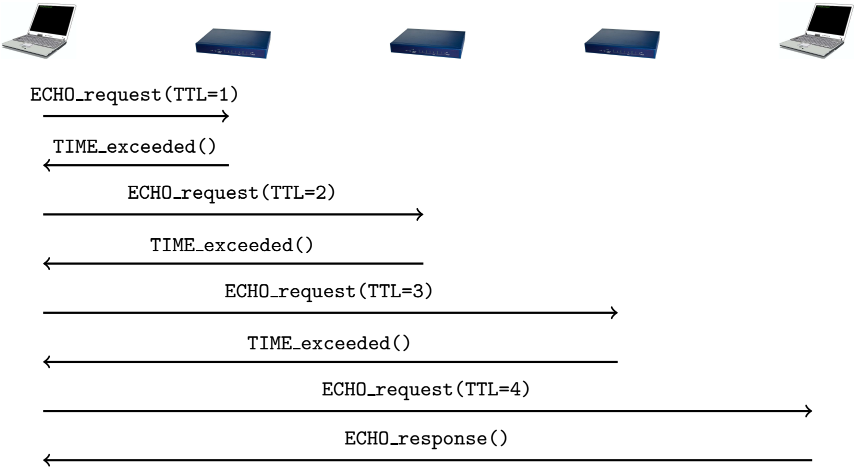
ECHO_request(TTL=1)
TIME_exceeded()
ECHO_request(TTL=2)
TIME_exceeded()



ECHO_request(TTL=1)
TIME_exceeded()
ECHO_request(TTL=2)
TIME_exceeded()
ECHO_request(TTL=3)

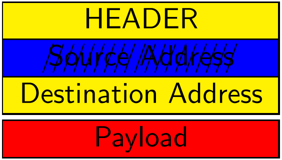

ECHO_request(TTL=1)
TIME_exceeded()
ECHO_request(TTL=2)
TIME_exceeded()
ECHO_request(TTL=3)
TIME_exceeded()


ECHO_request(TTL=1)
TIME_exceeded()
ECHO_request(TTL=2)
TIME_exceeded()
ECHO_request(TTL=3)
TIME_exceeded()
ECHO_request(TTL=4)

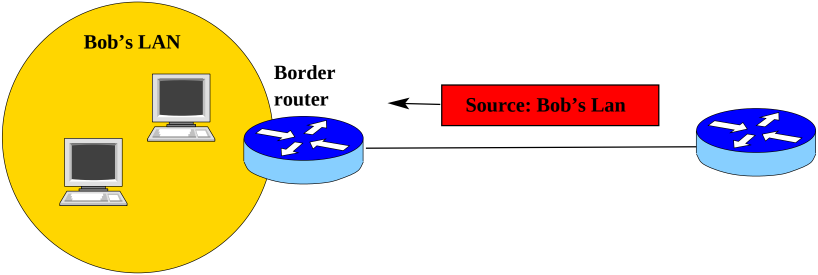

ECHO_request(TTL=1)
TIME_exceeded()
ECHO_request(TTL=2)
TIME_exceeded()
ECHO_request(TTL=3)
TIME_exceeded()
ECHO_request(TTL=4)
ECHO_response()

IP Spoofing

- The source address in an IP packet is never checked: overwrite it!
- The sender will never get a response! So, why? Denial of service attack.

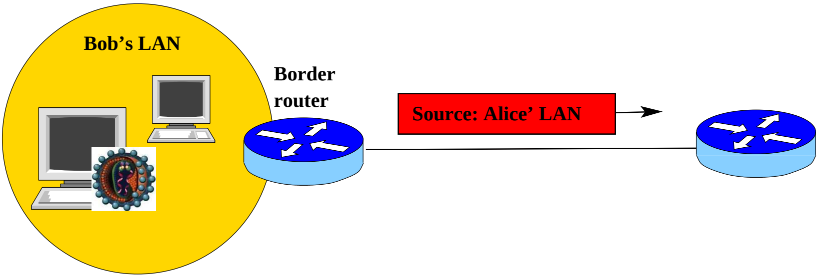


Countermeasures to IP Spoofing



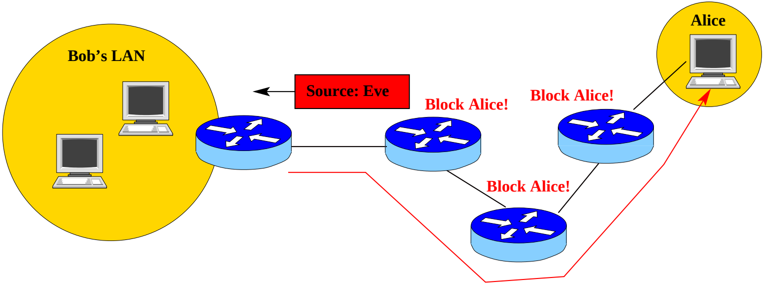
- Border router can block packets whose source address appears to be from inside the subnetwork, although they come from outside the subnetwork.

Countermeasures to IP Spoofing. . .



- Border router can block outgoing packets whose source address appears to be from outside the subnetwork.
- Maybe a node has been compromised by malware?

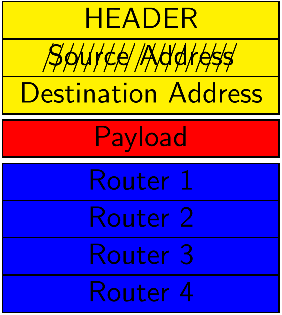
Countermeasures to IP Spoofing...



- IP Traceback — determining the origin of a packet, without using the `source` field.
- Once we know the actual source address, we can ask
  1. the ASs to block packets from this location.
  2. the ISP controlling the source address to block suspicious machines.

IP Traceback Techniques. . .

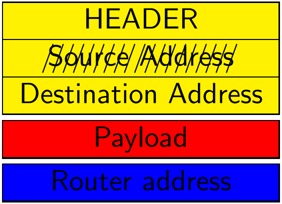
- Packet marking — routers add information to packets, so that their path can be reconstructed.
- Naive approach: each router adds its address to the end of the packet:



- Advantages: Easy to reconstruct path.
- Disadvantages: Router overhead, how to know if there's space in the packet?, packet fragmentation.

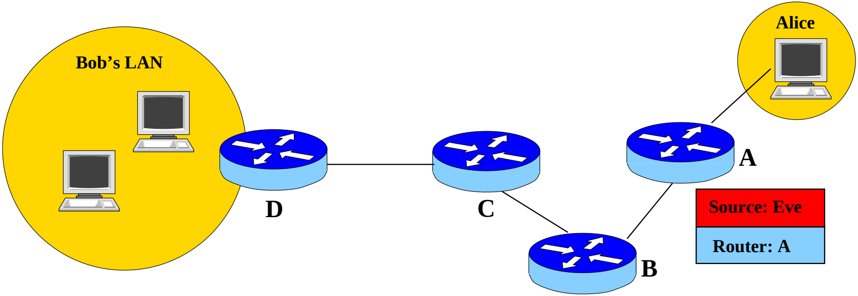
IP Traceback Techniques — Node Sampling

- Node sampling:
  - Only one router address can be stored in the packet.
  - A router writes its address with probability p.



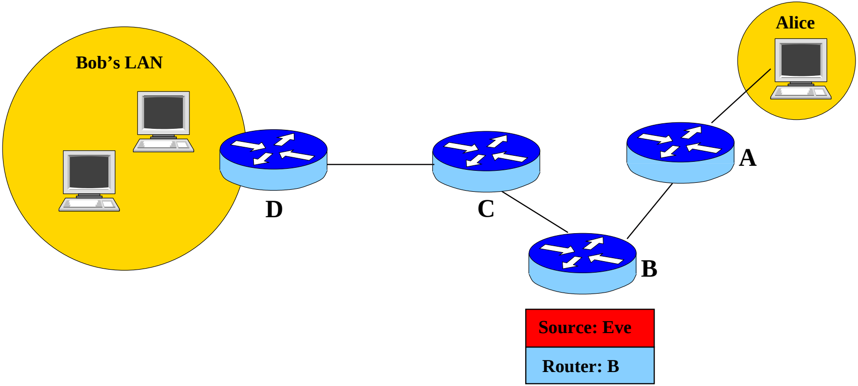
- Given enough packets, the path can be reconstructed.

IP Traceback Technique — Node Sampling...



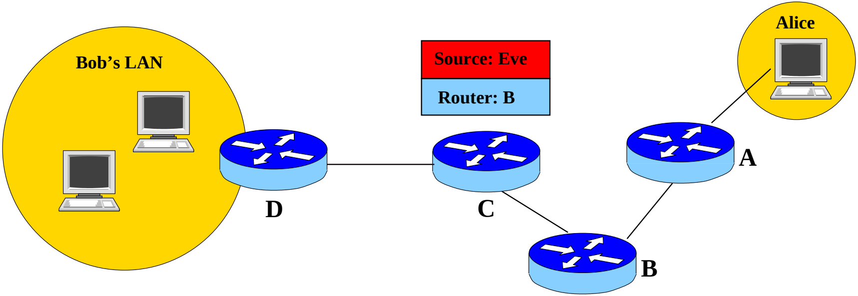
- Probability the packet will be marked by C: p
- Probability the packet will be marked by B: $p \cdot (1-p)$
- Probability the packet will be marked by A: $p \cdot (1-p) \cdot (1-p)$

IP Traceback Technique — Node Sampling. . .



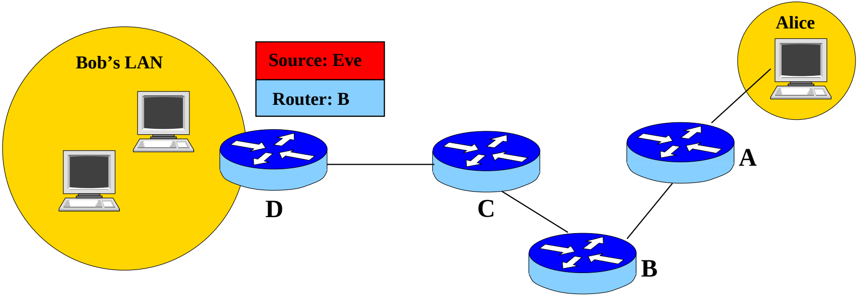
- Probability the packet will be marked by C: p
- Probability the packet will be marked by B: $p \cdot (1-p)$
- Probability the packet will be marked by A: $p \cdot (1-p) \cdot (1-p)$

IP Traceback Technique — Node Sampling...



- Probability the packet will be marked by C: p
- Probability the packet will be marked by B: $p \cdot (1 - p)$
- Probability the packet will be marked by A: $p \cdot (1 - p) \cdot (1 - p)$

IP Traceback Technique — Node Sampling...



- Probability the packet will be marked by C: p
- Probability the packet will be marked by B: $p \cdot (1-p)$
- Probability the packet will be marked by A: $p \cdot (1-p) \cdot (1-p)$

IP Traceback Technique — Other Techniques

- Many other techniques have been proposed.
- Most not implemented — require cooperation from Internet routers.

Outline

The Transport Layer

- Communication between processes connected to ports.

The Transport Layer

- Communication between processes connected to ports.
- Abstraction:

The Transport Layer

- Communication between processes connected to ports.
- Abstraction:
  1. Source/Destination: Ports connected to processes

The Transport Layer

- Communication between processes connected to ports.
- Abstraction:
  1. Source/Destination: Ports connected to processes
  2. Data: TCP/UDP packets

The Transport Layer

- Communication between processes connected to ports.
- Abstraction:
  1. Source/Destination: Ports connected to processes
  2. Data: TCP/UDP packets
  3. Addressing: IP address + port number

The Transport Layer

- Communication between processes connected to ports.
- Abstraction:
  1. Source/Destination: Ports connected to processes
  2. Data: TCP/UDP packets
  3. Addressing: IP address + port number
- Transmission Control Protocol (TCP) — connection-based protocol; guaranteed and ordered delivery of packets.

The Transport Layer

- Communication between processes connected to ports.
- Abstraction:
  1. Source/Destination: Ports connected to processes
  2. Data: TCP/UDP packets
  3. Addressing: IP address + port number
- Transmission Control Protocol (TCP) — connection-based protocol; guaranteed and ordered delivery of packets.
- User Datagram Protocol (UDP) — connection-less protocol; quick delivery without guarantees.

TCP Packet Format

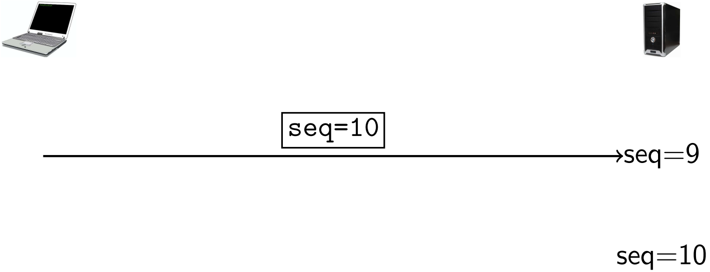
Source Port (16 bits)
Destination Port (16 bits)
Sequence Number (32 bits)
Acknowledgement Number (32 bits)
Offset (4 bits)
Reserved (4 bits)
Flags (8 bits)
Window size (16 bits)
Checksum (16 bits)
Urgent Pointer (16 bits)
Payload

TCP Sequence Number

seq=9

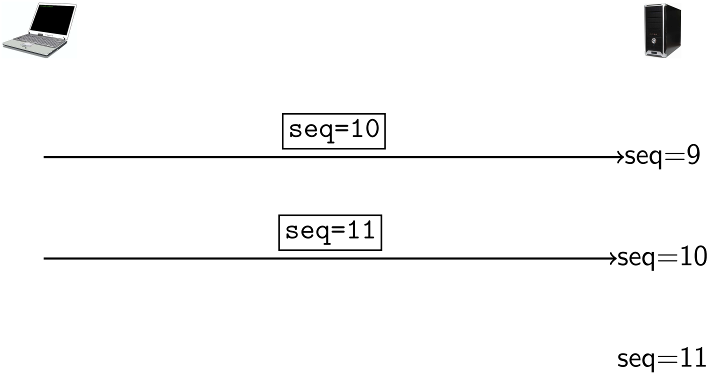
- Incremented for every packet *by payload length*.
- Allows us to determine when packets arrive out of order.
- Allows us to determine when packets don't arrive.

TCP Sequence Number



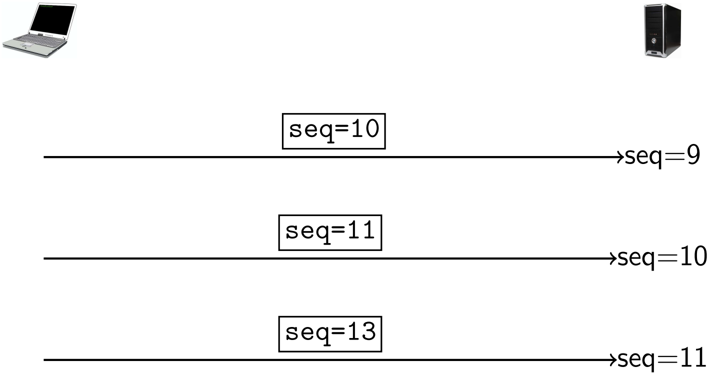
- Incremented for every packet *by payload length*.
- Allows us to determine when packets arrive out of order.
- Allows us to determine when packets don't arrive.

TCP Sequence Number



- Incremented for every packet *by payload length*.
- Allows us to determine when packets arrive out of order.
- Allows us to determine when packets don't arrive.

TCP Sequence Number



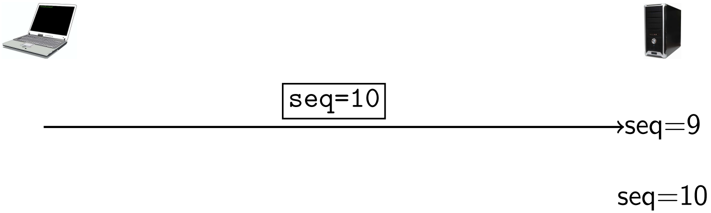
- Incremented for every packet *by payload length*.
- Allows us to determine when packets arrive out of order.
- Allows us to determine when packets don't arrive.

TCP Acknowledgement Number

seq=9

- Receiver sends an acknowledgement package with the sequence number of the next *payload byte* it wants to receive.

TCP Acknowledgement Number



- Receiver sends an acknowledgement package with the sequence number of the next *payload byte* it wants to receive.

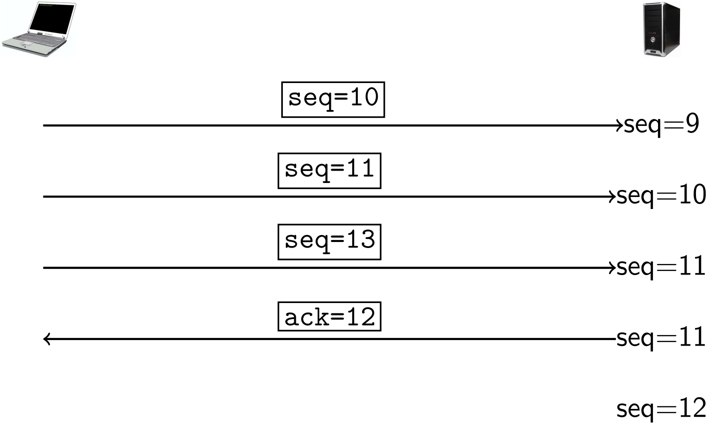
TCP Acknowledgement Number

- Receiver sends an acknowledgement package with the sequence number of the next *payload byte* it wants to receive.

TCP Acknowledgement Number

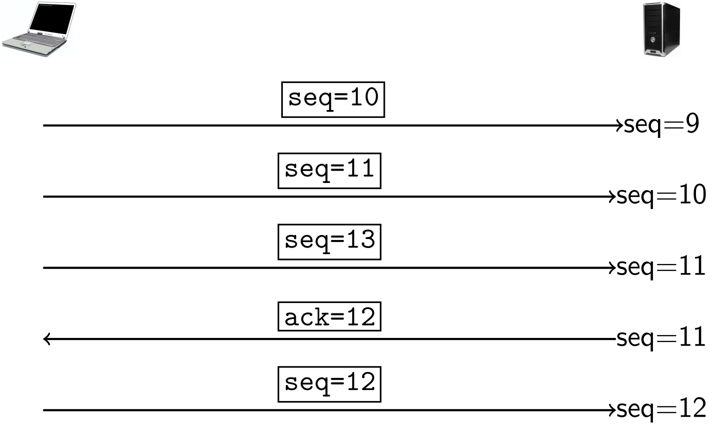
- Receiver sends an acknowledgement package with the sequence number of the next *payload byte* it wants to receive.

TCP Acknowledgement Number



- Receiver sends an acknowledgement package with the sequence number of the next *payload byte* it wants to receive.

TCP Acknowledgement Number

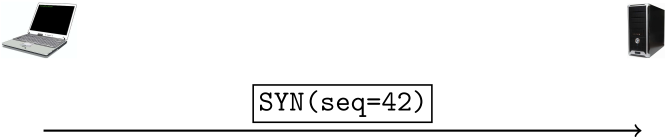


- Receiver sends an acknowledgement package with the sequence number of the next *payload byte* it wants to receive.

TCP Connections

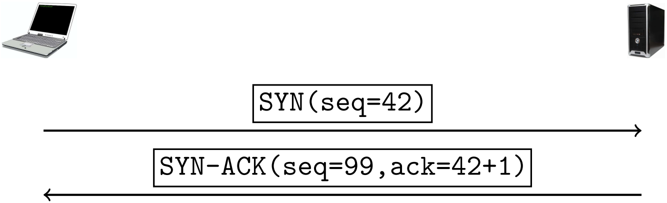
- TCP uses a 3-way handshake to set up a connection.
- The protocol includes a random initialization of the sequence number.

TCP Connections



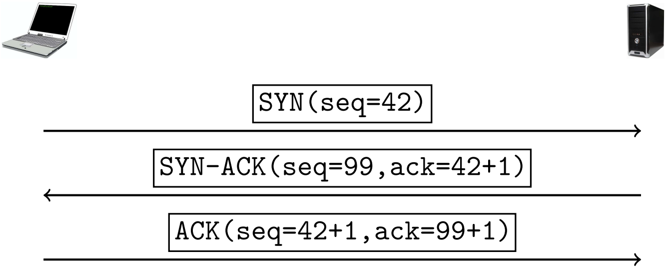
- TCP uses a 3-way handshake to set up a connection.
- The protocol includes a random initialization of the sequence number.

TCP Connections



- TCP uses a 3-way handshake to set up a connection.
- The protocol includes a random initialization of the sequence number.

TCP Connections



- TCP uses a 3-way handshake to set up a connection.
- The protocol includes a random initialization of the sequence number.

TCP Session Hijacking

- TCP Session Hijacking — an attacker
  1. hijacks another user's TCP connection;
  2. alters another user's TCP connection .

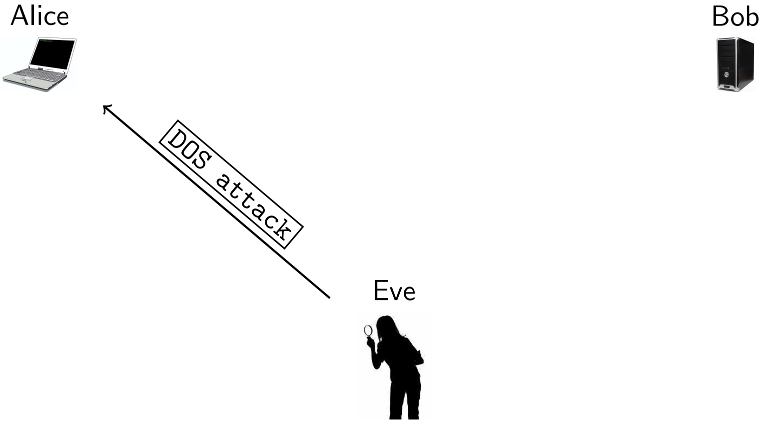
TCP Sequence Prediction Attack

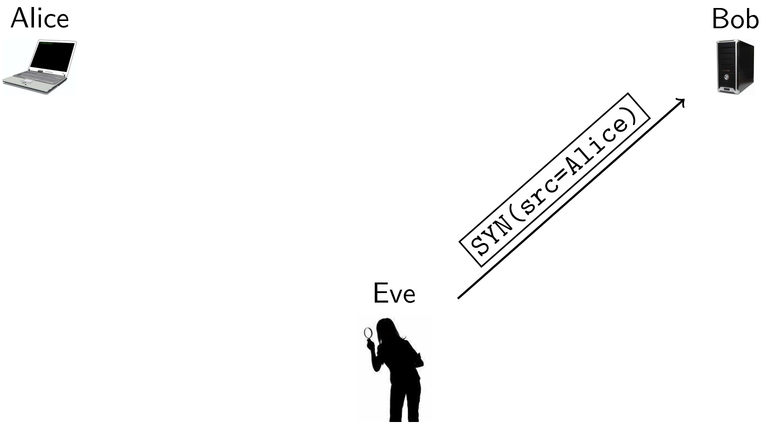
- Session spoofing — The attacker is able to create a TCP session with a server, who thinks it is talking to another client.
- Early TCP implementations had easily guessable sequence numbers.
- Attack:
  1. Eve launches a denial-of-service attack against Alice so she can't interfere with the attack.
  2. Eve sends a `SYN(src=Alice)` to Bob.
  3. Bob responds with a `SYN-ACK` to Alice, who cannot respond since she's under attack.
  4. Eve guesses N, Bob's next sequence number.
  5. Eve sends a `ACK(seq=`N`)` to Bob.
  6. Eve talks to Bob as if she is Alice.
- Blind injection attack: Eve won't receive replies from Bob.

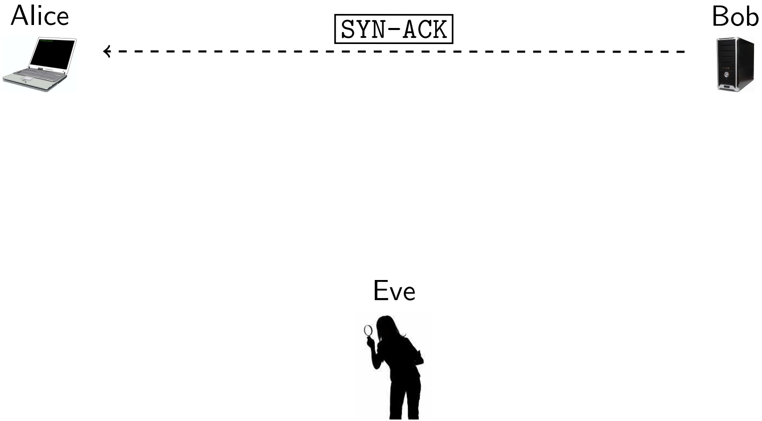
Alice

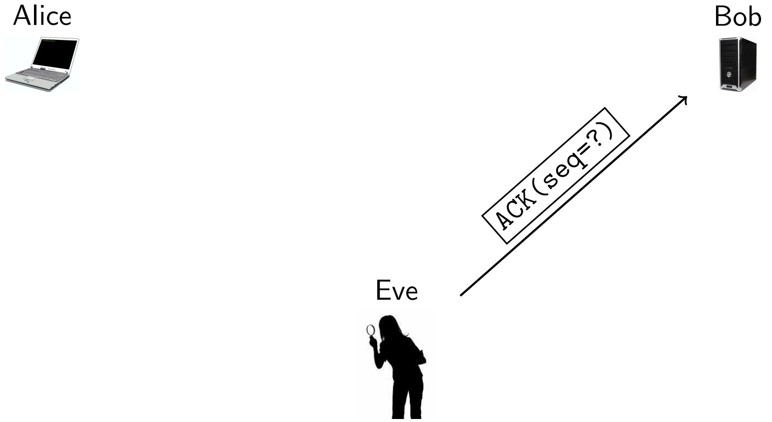
Bob

Eve


Alice
Bob
DOS attack
Eve


Alice
Bob
SYN(src=Alice)
Eve


Alice
SYN-ACK
Bob
Eve

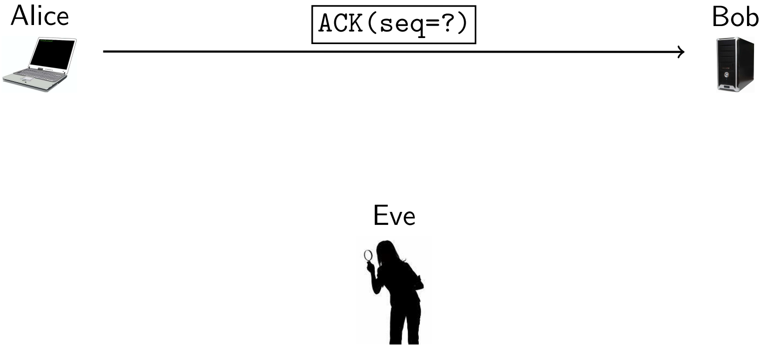


- Eve establishes a TCP connection with Bob, who thinks he's talking to Alice.
- Eve needs to guess the next sequence number Bob will use.

TCP Session Spoofing — ACK Storms

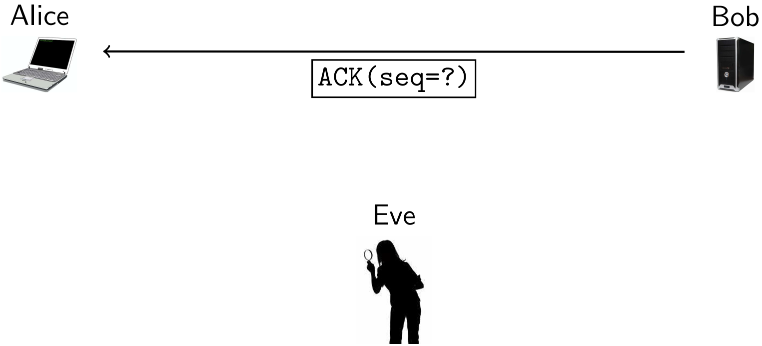
- Blind injection attacks can cause an ACK Storm, when the client and server try to resynchronize their sequence numbers.
- A firewall can, eventually, detect the ACK Storm.

TCP Session Spoofing — ACK Storms



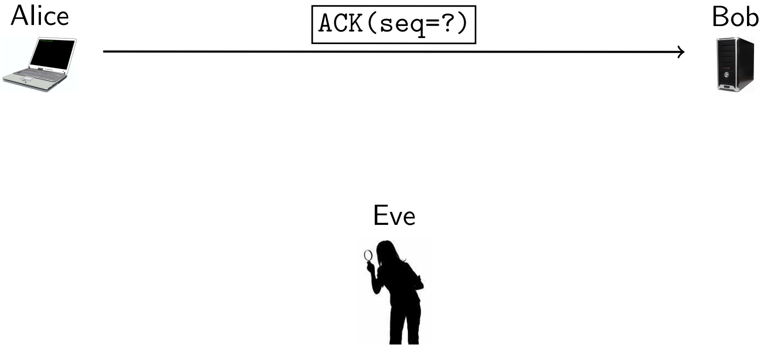
- Blind injection attacks can cause an ACK Storm, when the client and server try to resynchronize their sequence numbers.
- A firewall can, eventually, detect the ACK Storm.

TCP Session Spoofing — ACK Storms



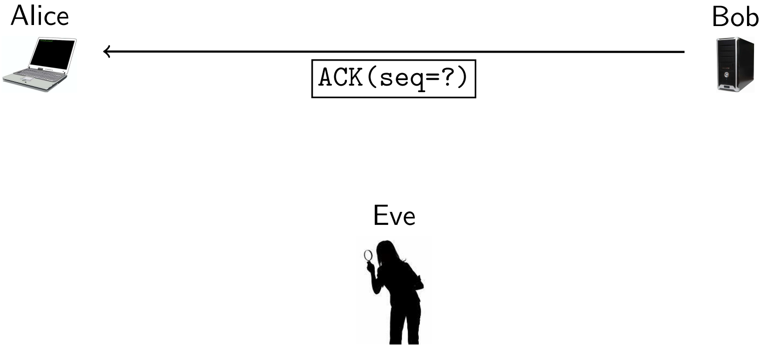
- Blind injection attacks can cause an ACK Storm, when the client and server try to resynchronize their sequence numbers.
- A firewall can, eventually, detect the ACK Storm.

TCP Session Spoofing — ACK Storms



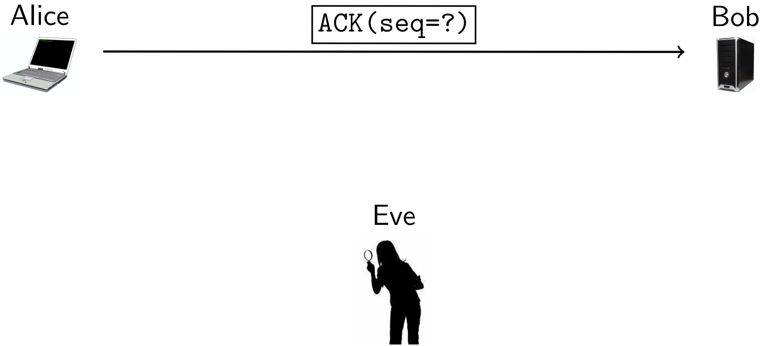
- Blind injection attacks can cause an ACK Storm, when the client and server try to resynchronize their sequence numbers.
- A firewall can, eventually, detect the ACK Storm.

TCP Session Spoofing — ACK Storms



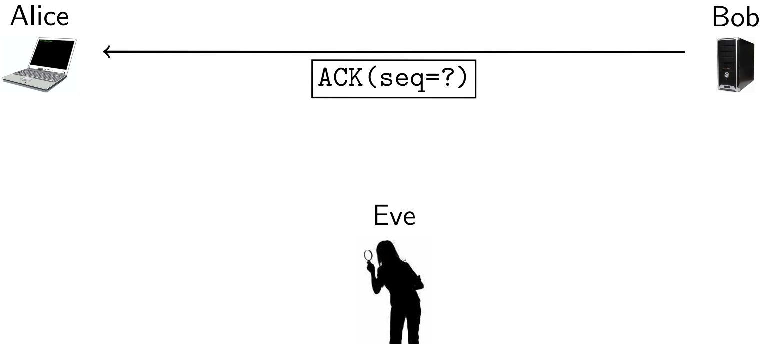
- Blind injection attacks can cause an ACK Storm, when the client and server try to resynchronize their sequence numbers.
- A firewall can, eventually, detect the ACK Storm.

TCP Session Spoofing — ACK Storms



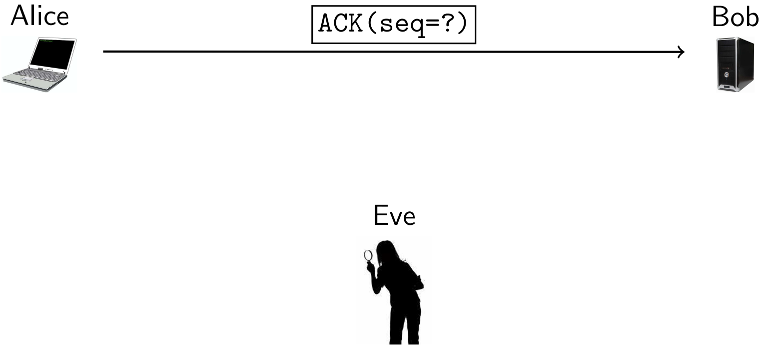
- Blind injection attacks can cause an ACK Storm, when the client and server try to resynchronize their sequence numbers.
- A firewall can, eventually, detect the ACK Storm.

TCP Session Spoofing — ACK Storms



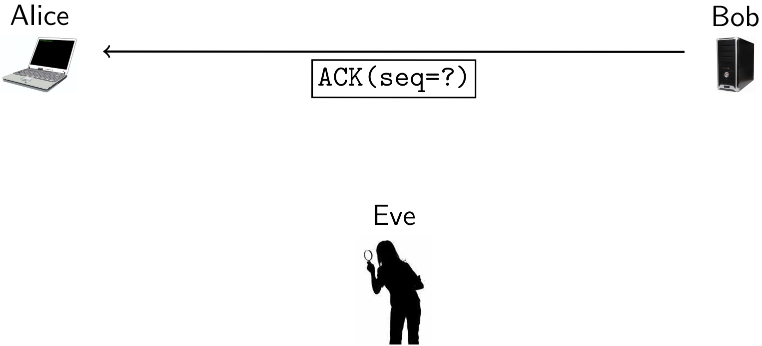
- Blind injection attacks can cause an ACK Storm, when the client and server try to resynchronize their sequence numbers.
- A firewall can, eventually, detect the ACK Storm.

TCP Session Spoofing — ACK Storms



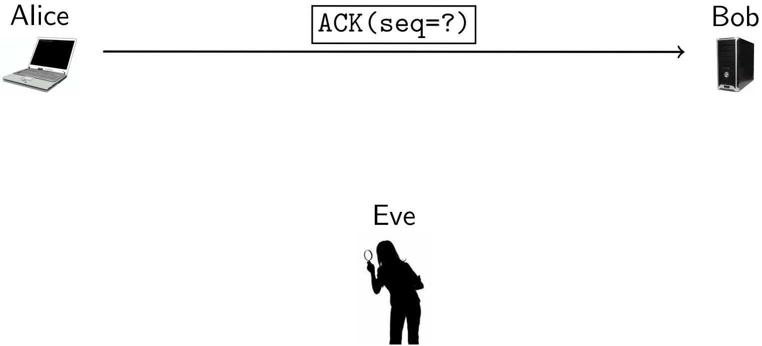
- Blind injection attacks can cause an ACK Storm, when the client and server try to resynchronize their sequence numbers.
- A firewall can, eventually, detect the ACK Storm.

TCP Session Spoofing — ACK Storms



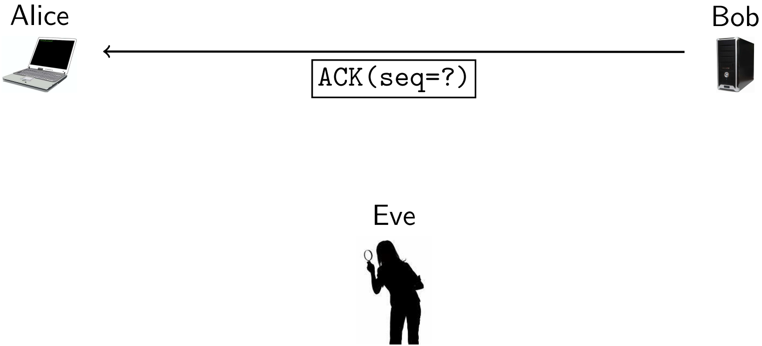
- Blind injection attacks can cause an ACK Storm, when the client and server try to resynchronize their sequence numbers.
- A firewall can, eventually, detect the ACK Storm.

TCP Session Spoofing — ACK Storms



- Blind injection attacks can cause an ACK Storm, when the client and server try to resynchronize their sequence numbers.
- A firewall can, eventually, detect the ACK Storm.

TCP Session Spoofing — ACK Storms



- Blind injection attacks can cause an ACK Storm, when the client and server try to resynchronize their sequence numbers.
- A firewall can, eventually, detect the ACK Storm.

Complete Session Hijacking

- Eve is on the same network segment as Alice and Bob, and packet sniffs on them as they establish their TCP connection.
- Eve guesses the next sequence number and sends a spoofed attack command to Bob, appearing to be Alice.

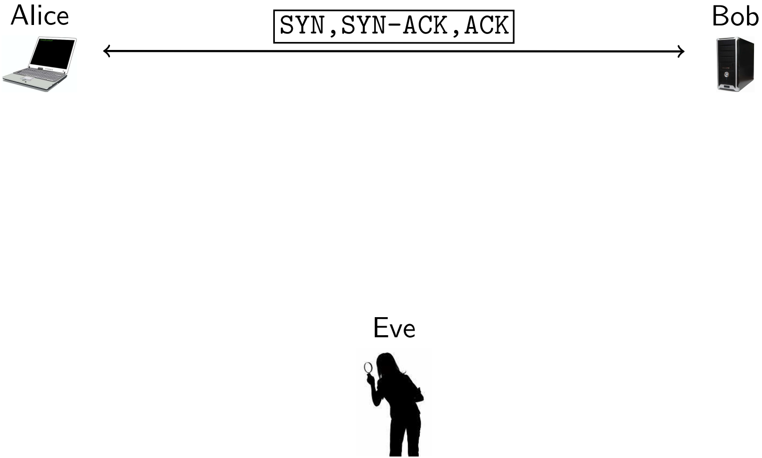
Complete Session Hijacking. . .

Alice

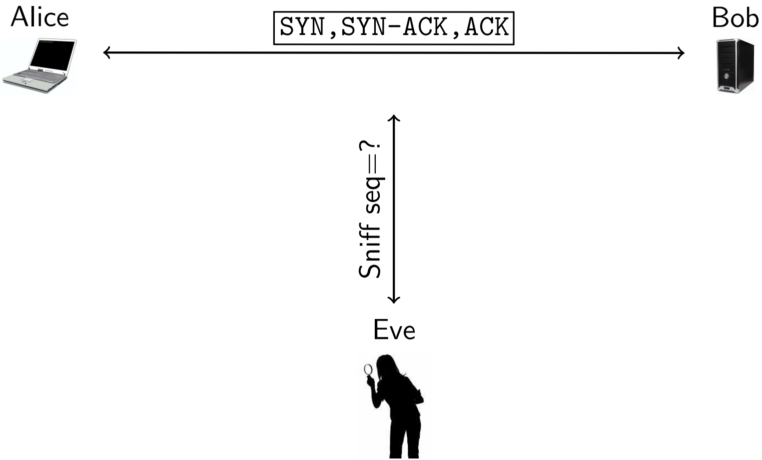
Bob

Eve

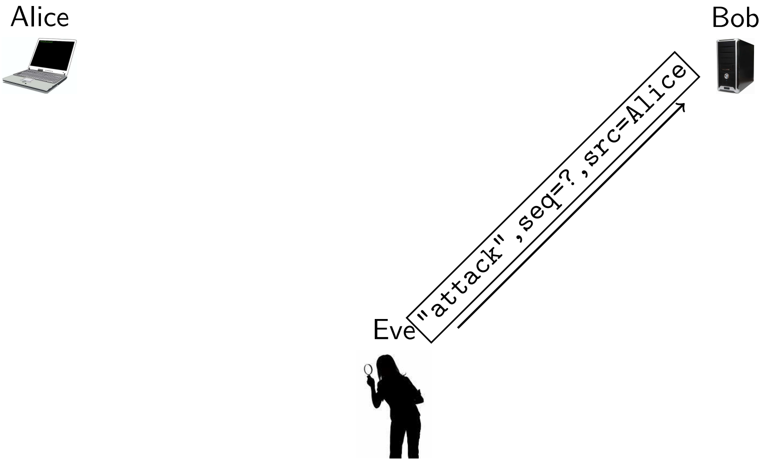
Complete Session Hijacking. . .



Complete Session Hijacking. . .



Complete Session Hijacking. . .



Countermeasures

- Don't use predictable sequence numbers.
- Encrypt at the network layer (IPsec).
- Encrypt at the application layer (https).

Outline

Denial-of-Service Attacks

- Web servers have limited bandwidth.
- Once the server has used up bandwidth/CPU, it starts dropping requests.
- Denial-of-Service Attacks: Any attack that targets a machine/software's availability.
- Source addresses are spoofed to hide the attacker's identity.

Internet Control Message Protocol

- The Internet Control Message Protocol is used for network diagnostics.
- ICMP messages:
  1. Echo request: please acknowledge reciept of packet.

Internet Control Message Protocol

- The Internet Control Message Protocol is used for network diagnostics.
- ICMP messages:
  1. Echo request: please acknowledge reciept of packet.
  2. Echo response: packet receipt is acknowledged.

Internet Control Message Protocol

- The Internet Control Message Protocol is used for network diagnostics.
- ICMP messages:
  1. Echo request: please acknowledge reciept of packet.
  2. Echo response: packet receipt is acknowledged.
  3. Time exceeded: notify that packet has expired (TTL=0).

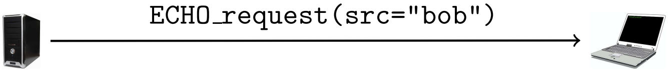
Internet Control Message Protocol

- The Internet Control Message Protocol is used for network diagnostics.
- ICMP messages:
  1. Echo request: please acknowledge reciept of packet.
  2. Echo response: packet receipt is acknowledged.
  3. Time exceeded: notify that packet has expired (TTL=0).
  4. Destination unreachable: notify that packet could not be delivered.

Ping Flood Attack

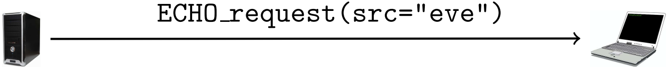
- A powerful machine can attack a less powerful one by sending it a large number of `ECHO_requests`.

Ping Flood Attack



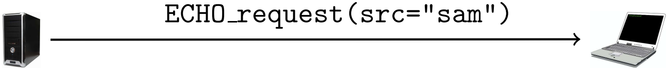
- A powerful machine can attack a less powerful one by sending it a large number of `ECHO_requests`.

Ping Flood Attack



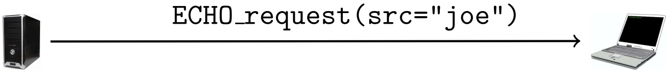
- A powerful machine can attack a less powerful one by sending it a large number of ECHO_requests.

Ping Flood Attack



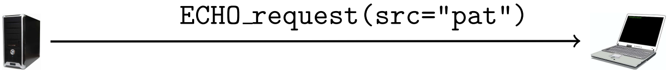
- A powerful machine can attack a less powerful one by sending it a large number of ECHO_requests.

Ping Flood Attack



- A powerful machine can attack a less powerful one by sending it a large number of `ECHO_requests`.

Ping Flood Attack



- A powerful machine can attack a less powerful one by sending it a large number of ECHO_requests.

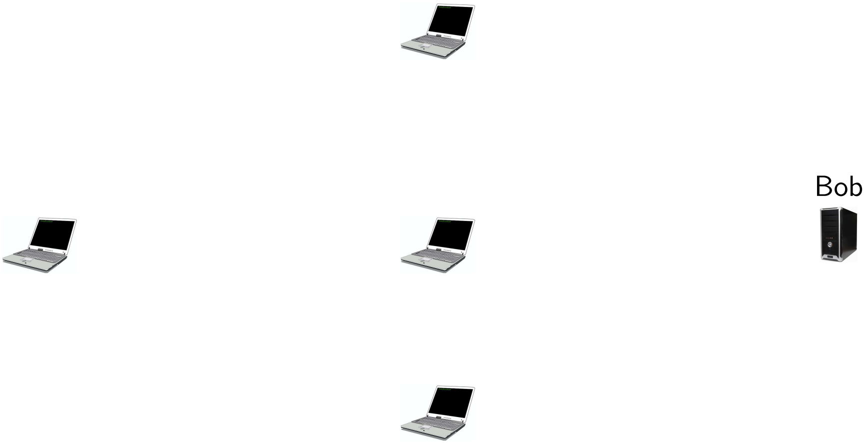
Smurf Attack

- A broadcast address sends to all IP addresses on the network.
- In a smurf attack, we get an amplification effect by creating an `ECHO_request` with a spoofed source address (of the target) and broadcasting this to all nodes on the network.
- Attack:
  1. Broadcast the packet `ECHO_request(src="target",dest="EVERYBODY")` to the nodes on the network.

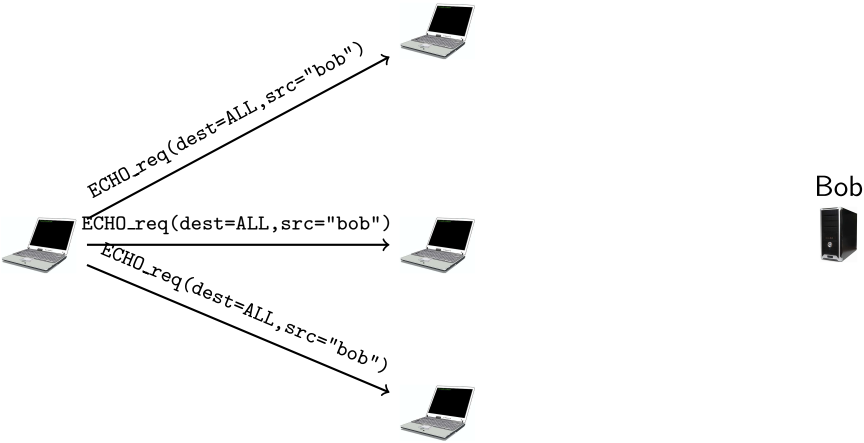
Smurf Attack

- A broadcast address sends to all IP addresses on the network.
- In a smurf attack, we get an amplification effect by creating an ECHO_request with a spoofed source address (of the target) and broadcasting this to all nodes on the network.
- Attack:
  1. Broadcast the packet `ECHO_request(src="target",dest="EVERYBODY")` to the nodes on the network.
  2. Each node N will respond with `ECHO_response(src=`N`,dest="target")`.

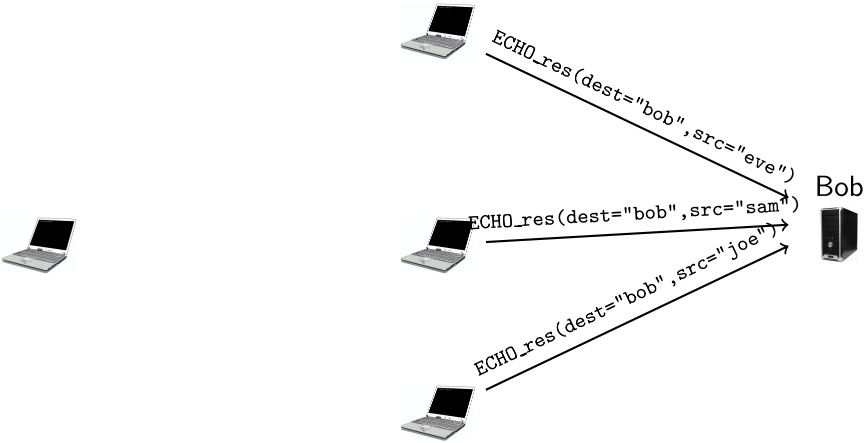
Smurf Attack. . .



Smurf Attack...



Smurf Attack. . .



Smurf Attack...

- Countermeasures:
  1. Make hosts and routers ignore broadcasts.
  2. Make servers ignore all `PING`s.

SYN Flood Attacks

- Idea: Start lots of connections to a server, but never finish the SYN/SYN-ACK/ACK sequence, causing the server's memory to fill up.
- Attack:
    1. Eve sends a `SYN(src="joe")` packet to Alice's server.

SYN Flood Attacks

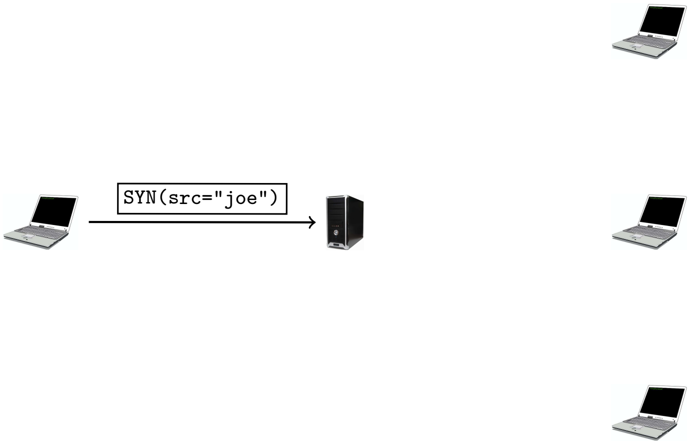
- Idea: Start lots of connections to a server, but never finish the SYN/SYN-ACK/ACK sequence, causing the server's memory to fill up.
- Attack:
  1. Eve sends a `SYN(src="joe")` packet to Alice's server.
  2. Server responds with `SYN-ACK`, sent to joe.

SYN Flood Attacks

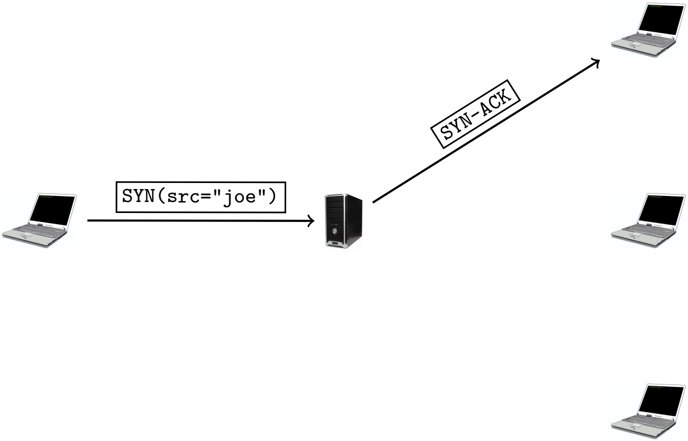
- Idea: Start lots of connections to a server, but never finish the SYN/SYN-ACK/ACK sequence, causing the server's memory to fill up.
- Attack:
  1. Eve sends a `SYN(src="joe")` packet to Alice's server.
  2. Server responds with `SYN-ACK`, sent to joe.
  3. Eve repeats from 1.

SYN Flood Attacks. . .

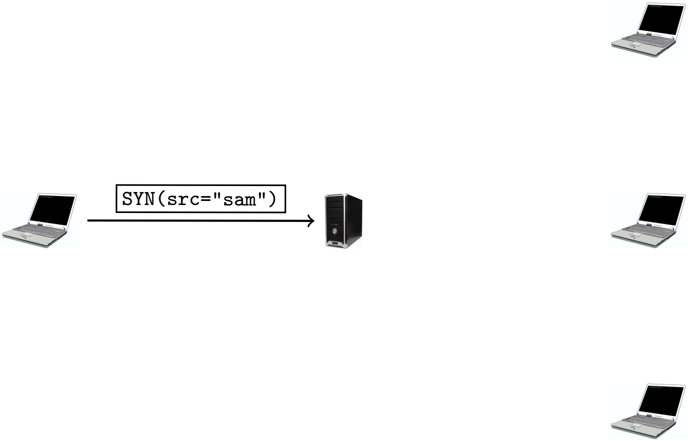
SYN Flood Attacks. . .



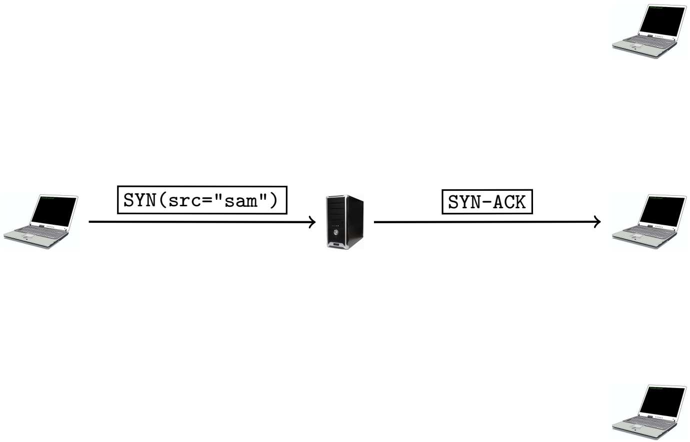
SYN Flood Attacks...



SYN Flood Attacks. . .

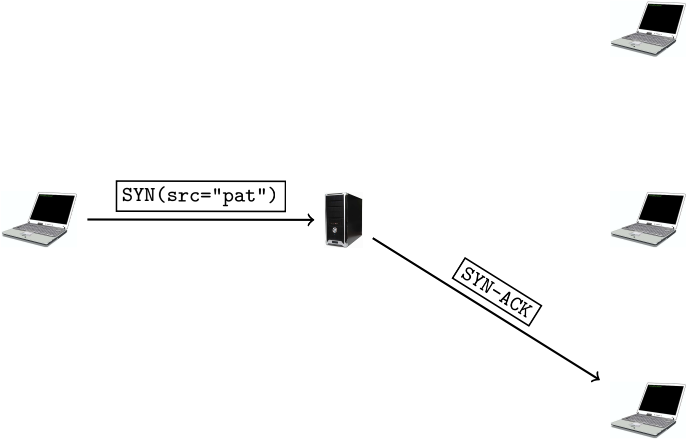


SYN Flood Attacks...



SYN Flood Attacks. . .

SYN Flood Attacks...



SYN Flood Attacks — Countermeasures

- SYN Cookies (see the book).
- Microsoft Windows:
  - A special queue for half-open connections.
  - Don't allocate resources for the TCP connection until the `ACK` has been received.

Outline

Readings and References

- Chapter 5 in *Introduction to Computer Security*, by Goodrich and Tamassia.