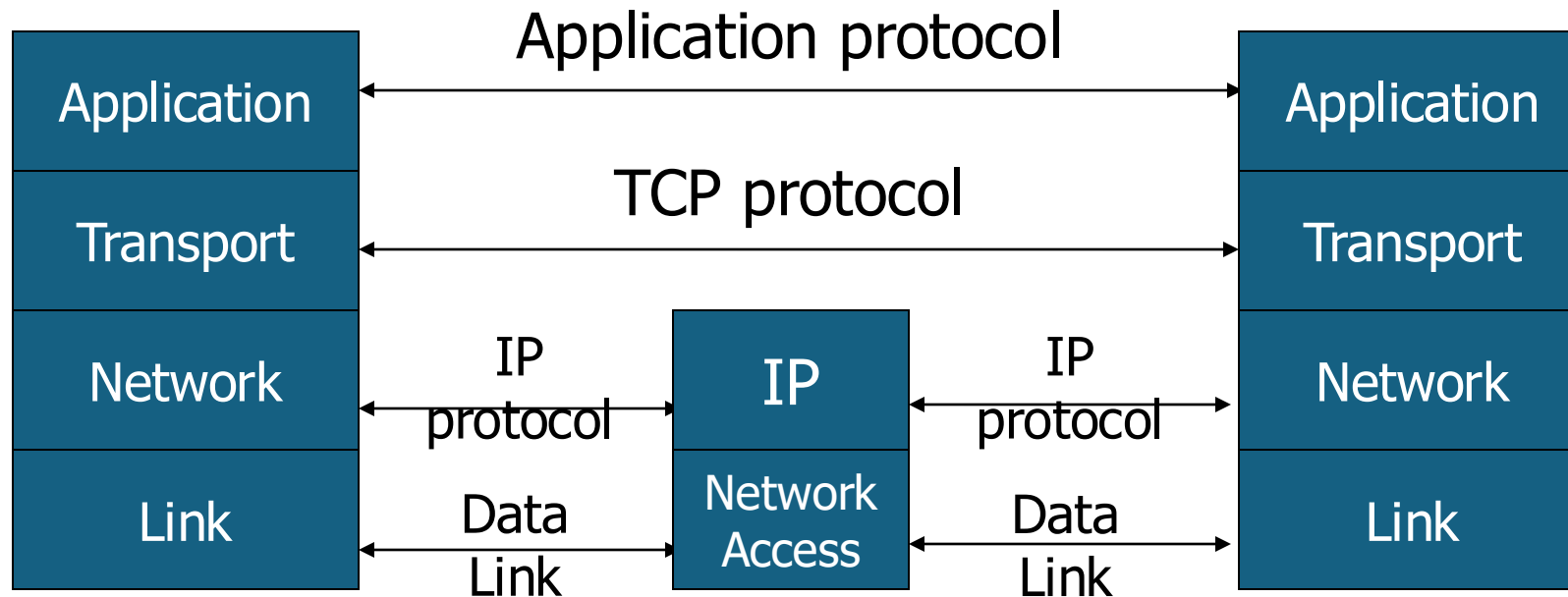


# Network Security

# Network Protocols Stack



# Types of Addresses in Internet

- Media Access Control (MAC) addresses in the network access layer
  - Associated w/ network interface card (NIC)
  - 48 bits or 64 bits
- IP addresses for the network layer
  - 32 bits for IPv4, and 128 bits for IPv6
  - E.g., 128.3.23.3
- IP addresses + ports for the transport layer
  - E.g., 128.3.23.3:80
- Domain names for the application/human layer
  - E.g., www.purdue.edu

# Routing and Translation of Addresses

- Translation between IP addresses and MAC addresses
  - Address Resolution Protocol (ARP) for IPv4
  - Neighbor Discovery Protocol (NDP) for IPv6
- Routing with IP addresses: Destination IP --> Next hop IP
  - TCP, UDP, IP for routing packets, connections
  - Border Gateway Protocol for routing table updates
- Translation between IP addresses and domain names
  - Domain Name System (DNS)

# Threats in Networking

- Confidentiality
  - e.g. Packet sniffing
- Integrity
  - e.g. Session hijacking, MITM, Replay attack
- Availability
  - e.g. Denial of service attacks
- Common
  - e.g. Address translation poisoning attacks
  - e.g. Routing attacks

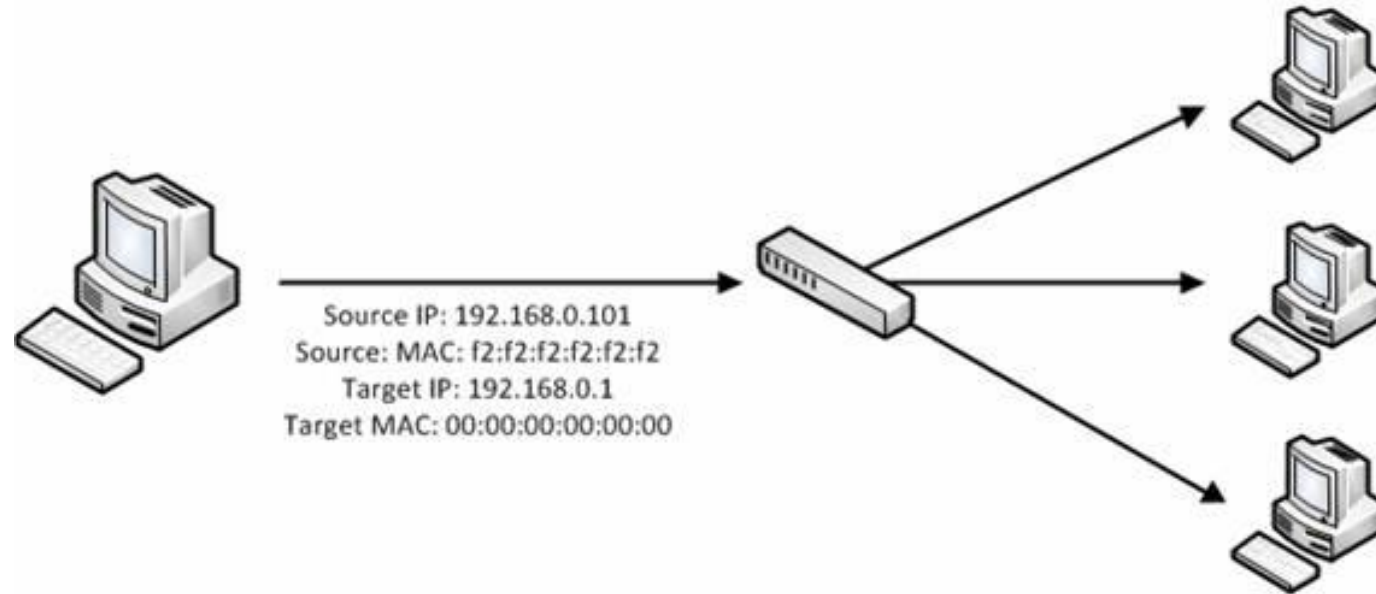
# Concrete Security Problems

- ARP is not authenticated
  - APR spoofing (or ARP poisoning)
- Network packets pass by untrusted hosts
  - Packet sniffing
- TCP state can be easy to guess
  - TCP spoofing attack
- Open access
  - Vulnerable to DoS attacks
- DNS is not authenticated
  - DNS poisoning attacks

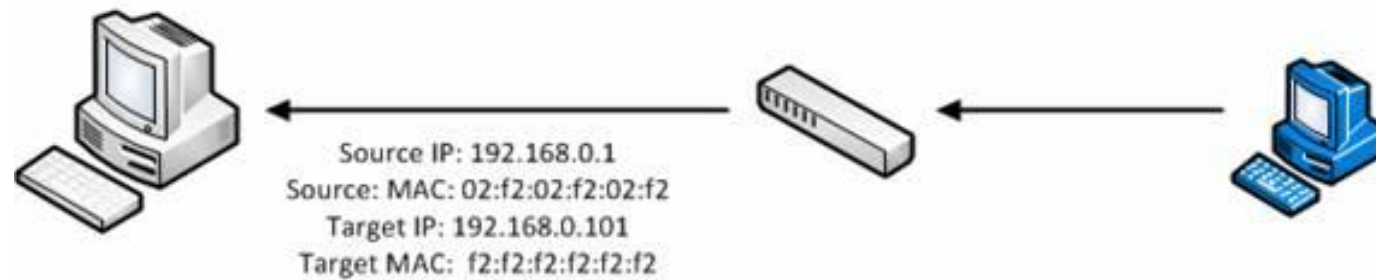
# Address Resolution Protocol (ARP)

- Primarily used to translate IP addresses to Ethernet MAC addresses
  - The device driver for Ethernet NIC needs to do this to send a packet
- Also used for IP over other LAN technologies, e.g. IEEE 802.11
- Each host maintains a table of IP to MAC addresses
- Message types:
  - ARP request
  - ARP reply
  - ARP announcement

## ARP Request



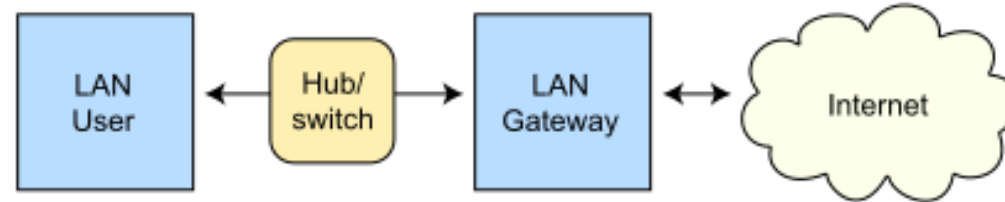
## ARP Response



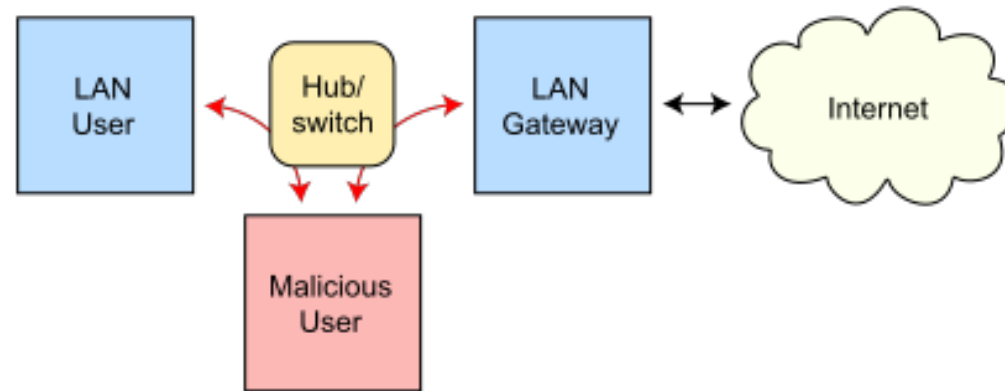
<http://www.windowsecurity.com>

# ARP Spoofing (ARP Poisoning)

Routing under normal operation



Routing subject to ARP cache poisoning



- Send fake or 'spoofed', ARP messages to an Ethernet LAN.
  - To have other machines associate IP addresses with the attacker's MAC
- Legitimate use
  - redirect a user to a registration page before allow usage of the network.
  - Implementing redundancy and fault tolerance

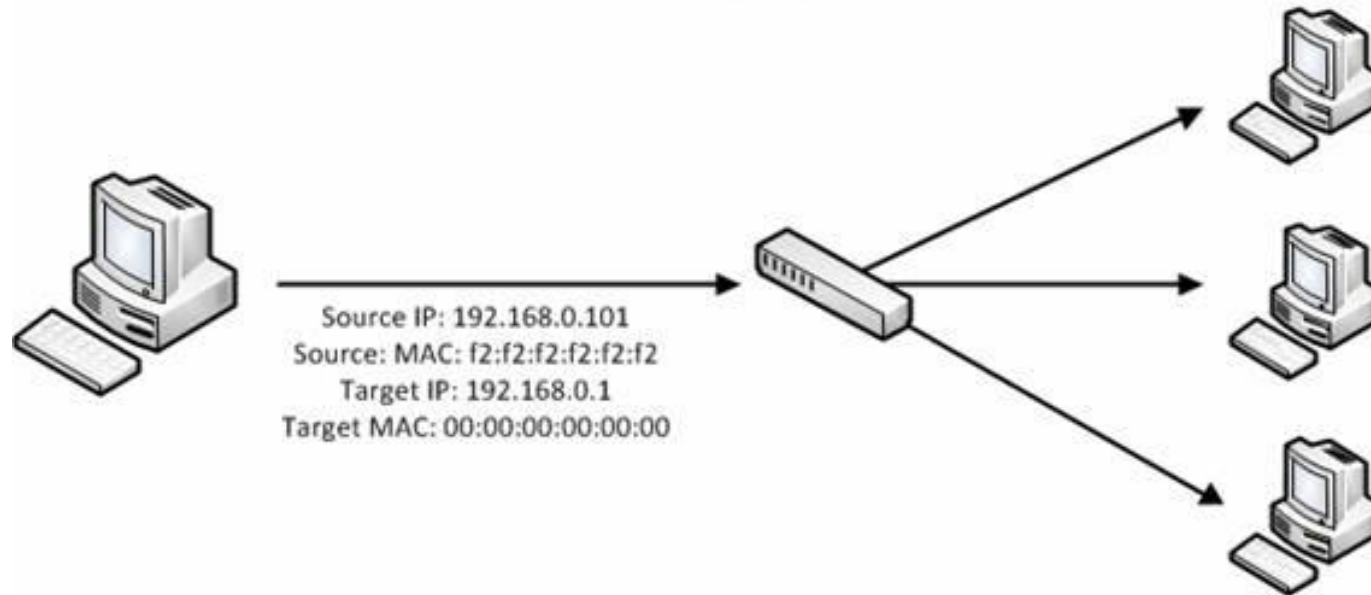
# ARP Spoofing (ARP Poisoning) - 2

- Defenses
  - static ARP table
  - DHCP Certification (use access control to ensure that hosts only use the IP addresses assigned to them, and that only authorized DHCP servers are accessible).
  - detection: Arpwatch (sending email when updates occur),

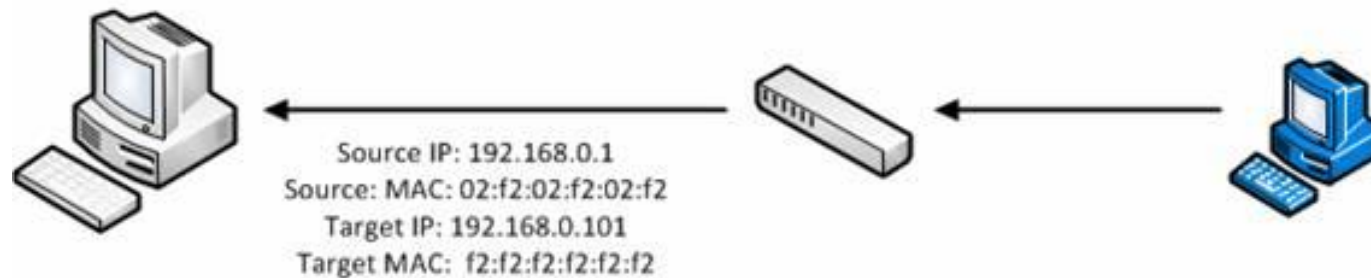
# Address Resolution Protocol (ARP)

- Primarily used to translate IP addresses to Ethernet MAC addresses
  - The device driver for Ethernet NIC needs to do this to send a packet
- Also used for IP over other LAN technologies, e.g. IEEE 802.11
- Each host maintains a table of IP to MAC addresses
- Message types:
  - ARP request
  - ARP reply
  - ARP announcement

## ARP Request



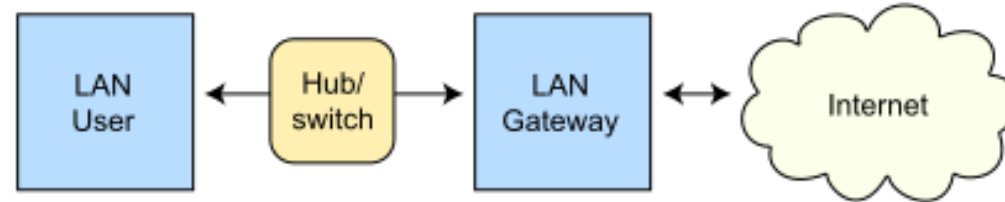
## ARP Response



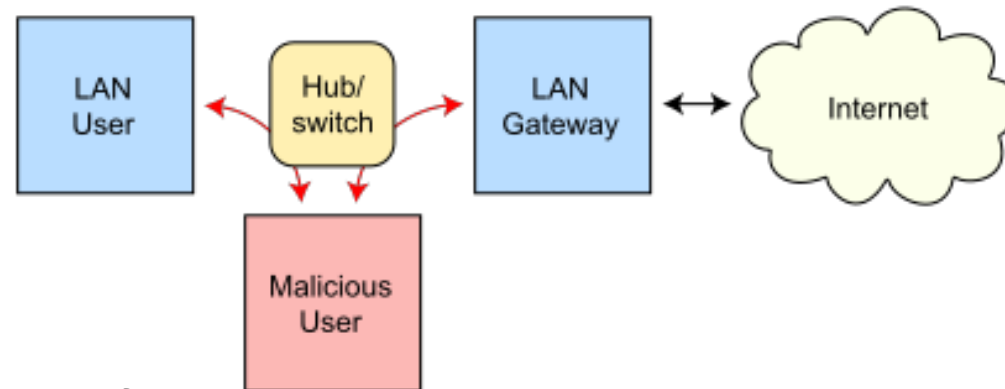
<http://www.windowsecurity.com>

# ARP Spoofing (ARP Poisoning)

Routing under normal operation



Routing subject to ARP cache poisoning

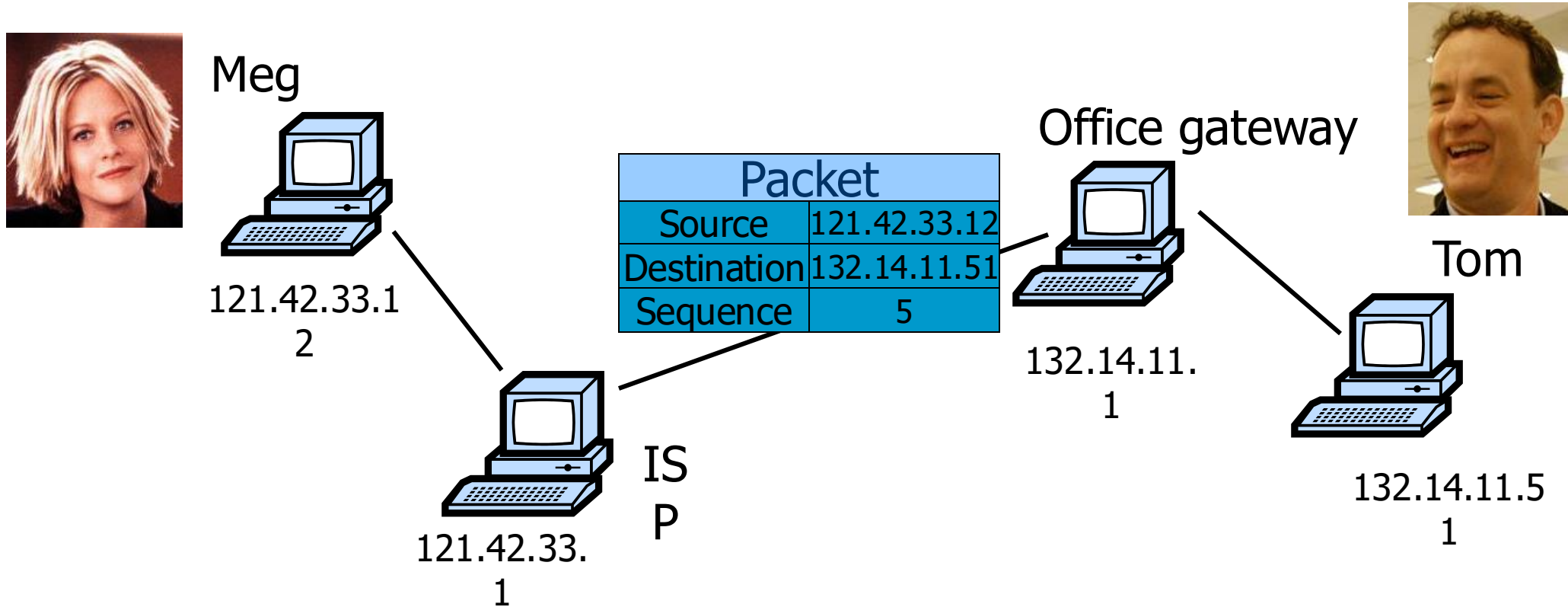


- Send fake or 'spoofed', ARP messages to an Ethernet LAN.
  - To have other machines associate IP addresses with the attacker's MAC
- Legitimate use
  - redirect a user to a registration page before allow usage of the network.
  - Implementing redundancy and fault tolerance

# ARP Spoofing (ARP Poisoning) - 2

- Defenses
  - static ARP table
  - DHCP Certification (use access control to ensure that hosts only use the IP addresses assigned to them, and that only authorized DHCP servers are accessible).
  - detection: Arpwatch (sending email when updates occur),

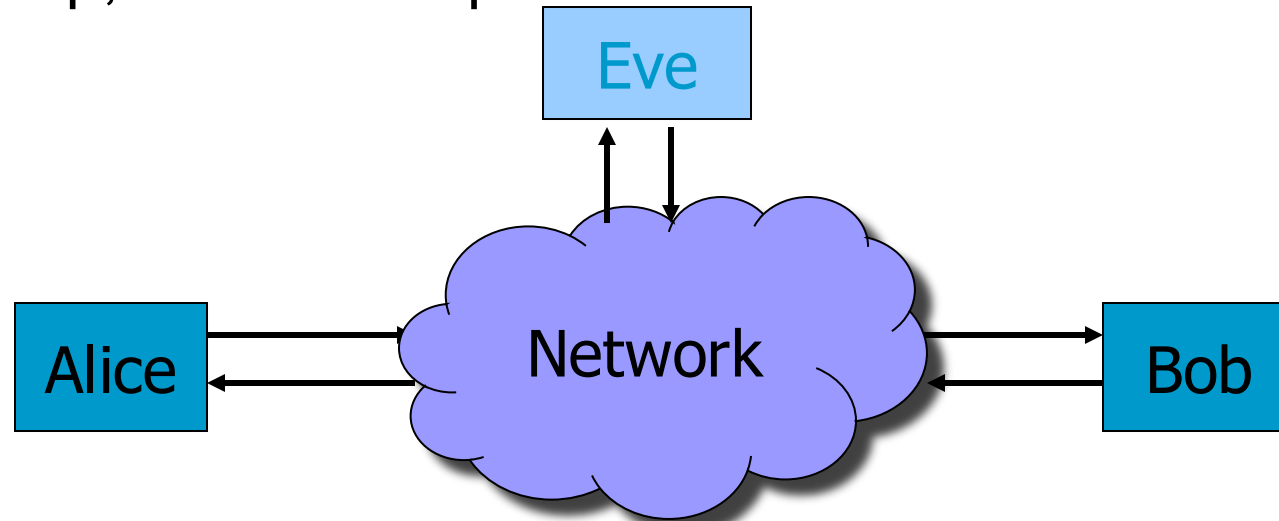
# IP Routing



- Internet routing uses numeric IP address
- Typical route uses several hops

# Packet Sniffing

- Promiscuous Network Interface Card reads all packets
  - Read all unencrypted data (e.g., “ngrep”)
  - ftp, telnet send passwords in clear!



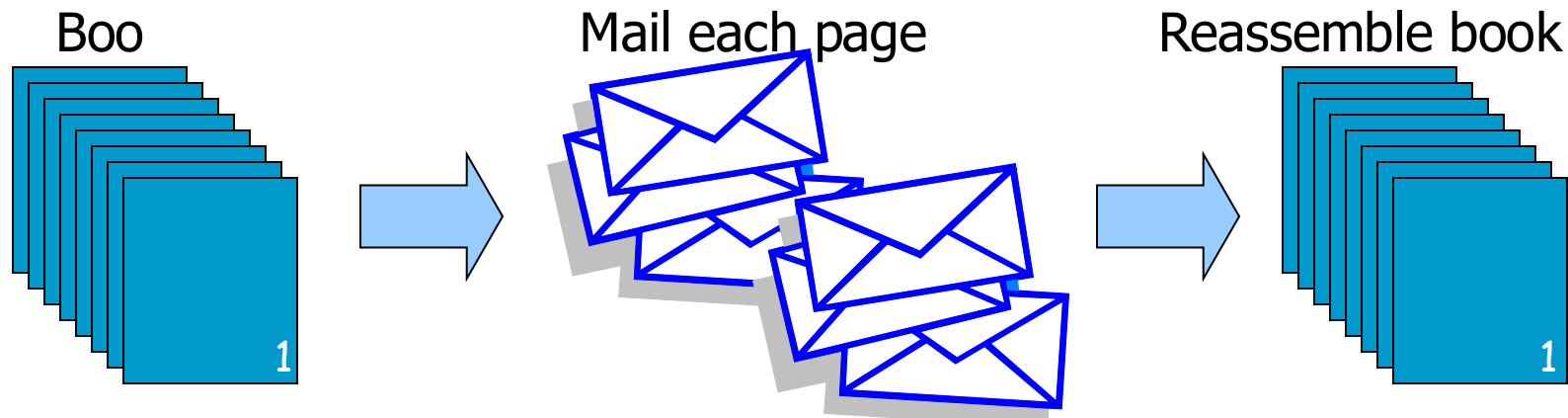
Prevention: Encryption (IPSEC, TLS)

# User Datagram Protocol

- IP provides routing
  - IP address gets datagram to a specific machine
- UDP separates traffic by port (16-bit number)
  - Destination port number gets UDP datagram to particular application process, e.g., 128.3.23.3:53
  - Source port number provides return address
- Minimal guarantees
  - No acknowledgment
  - No flow control
  - No message continuation

# Transmission Control Protocol

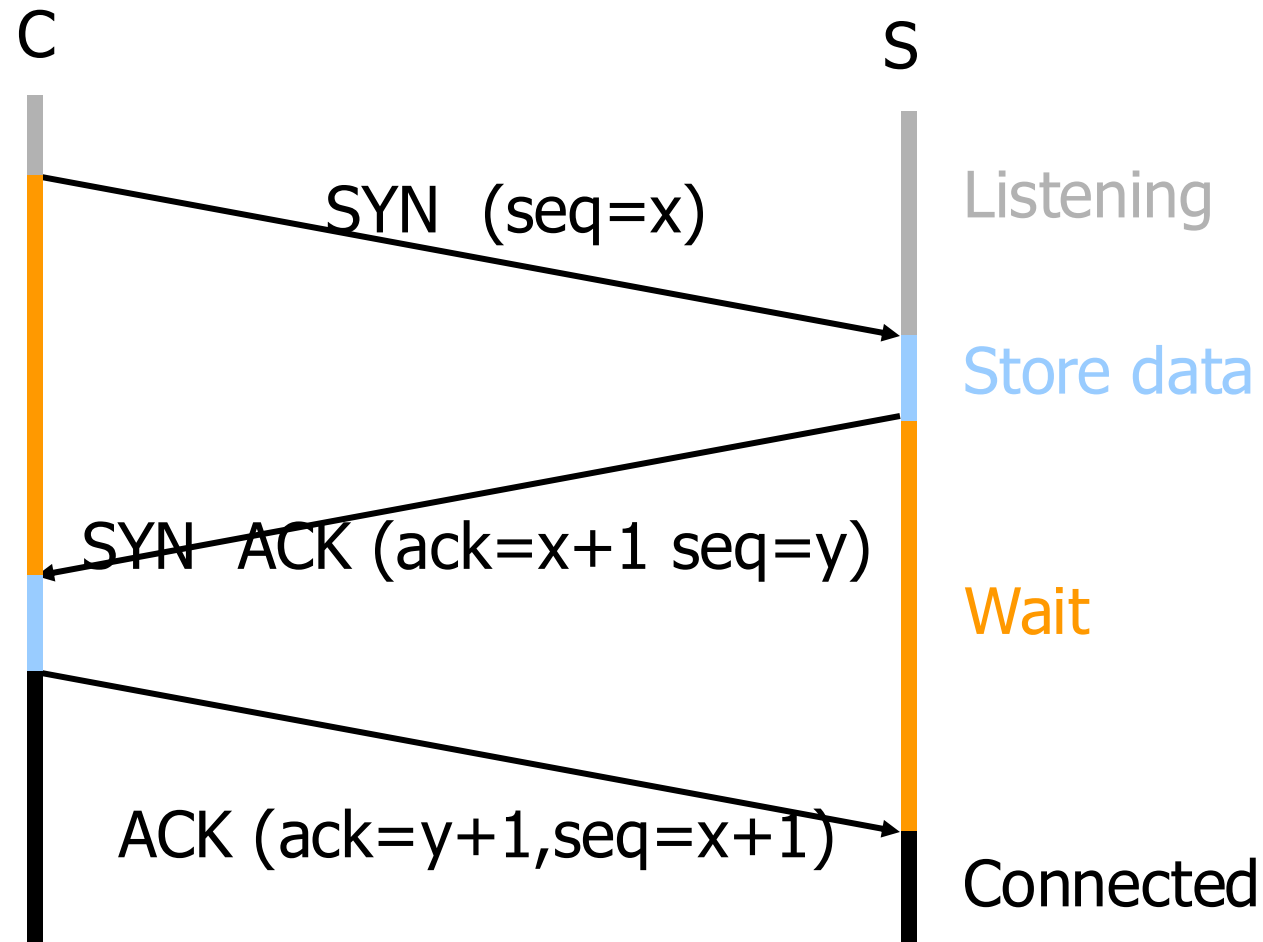
- Connection-oriented, preserves order
  - Sender
    - Break data into packets
    - Attach sequence numbers
  - Receiver
    - Acknowledge receipt; lost packets are resent
    - Reassemble packets in correct order



# TCP Sequence Numbers

- Sequence number (32 bits) – has a dual role:
  - If the SYN flag is set, then this is the initial sequence number. The sequence number of the actual first data byte is this sequence number plus 1.
  - If the SYN flag is clear, then this is the accumulated sequence number of the first data byte of this packet for the current session.
- Acknowledgment number (32 bits) –
  - If the ACK flag is set then this the next sequence number that the receiver is expecting.
  - This acknowledges receipt of all prior bytes (if any).

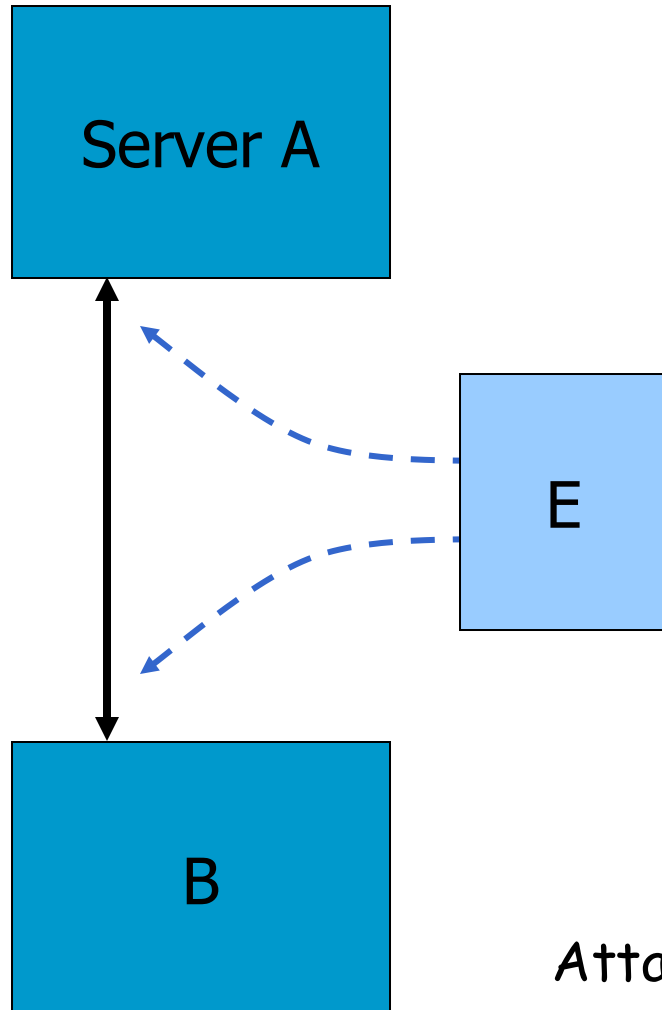
# TCP Handshake



# TCP sequence prediction attack

- Predict the sequence number used to identify the packets in a TCP connection, and then counterfeit packets.
- Adversary: do not have full control over the network, but can inject packets with fake source IP addresses
  - E.g., control a computer on the local network
- TCP sequence numbers are used for authenticating packets
- Initial seq# needs high degree of unpredictability
  - If attacker knows initial seq # and amount of traffic sent, can estimate likely current values
  - Some implementations are vulnerable

# Blind TCP Session Hijacking



- A, B trusted connection
  - Send packets with predictable seq numbers
- E impersonates B to A
  - Opens connection to A to get initial seq number
  - DoS B's queue
  - Sends packets to A that resemble B's transmission
  - E cannot receive, but may execute commands on A

Attack can be blocked if E is outside firewall.

# Risks from Session Hijacking

- Inject data into an unencrypted server-to-server traffic, such as an e-mail exchange, DNS zone transfers, etc.
- Inject data into an unencrypted client-to-server traffic, such as ftp file downloads, http responses.
- Spoof IP addresses, which are often used for preliminary checks on firewalls or at the service level.
- Carry out MITM attacks on weak cryptographic protocols.
  - often result in warnings to users that get ignored
- Denial of service attacks, such as resetting the connection.

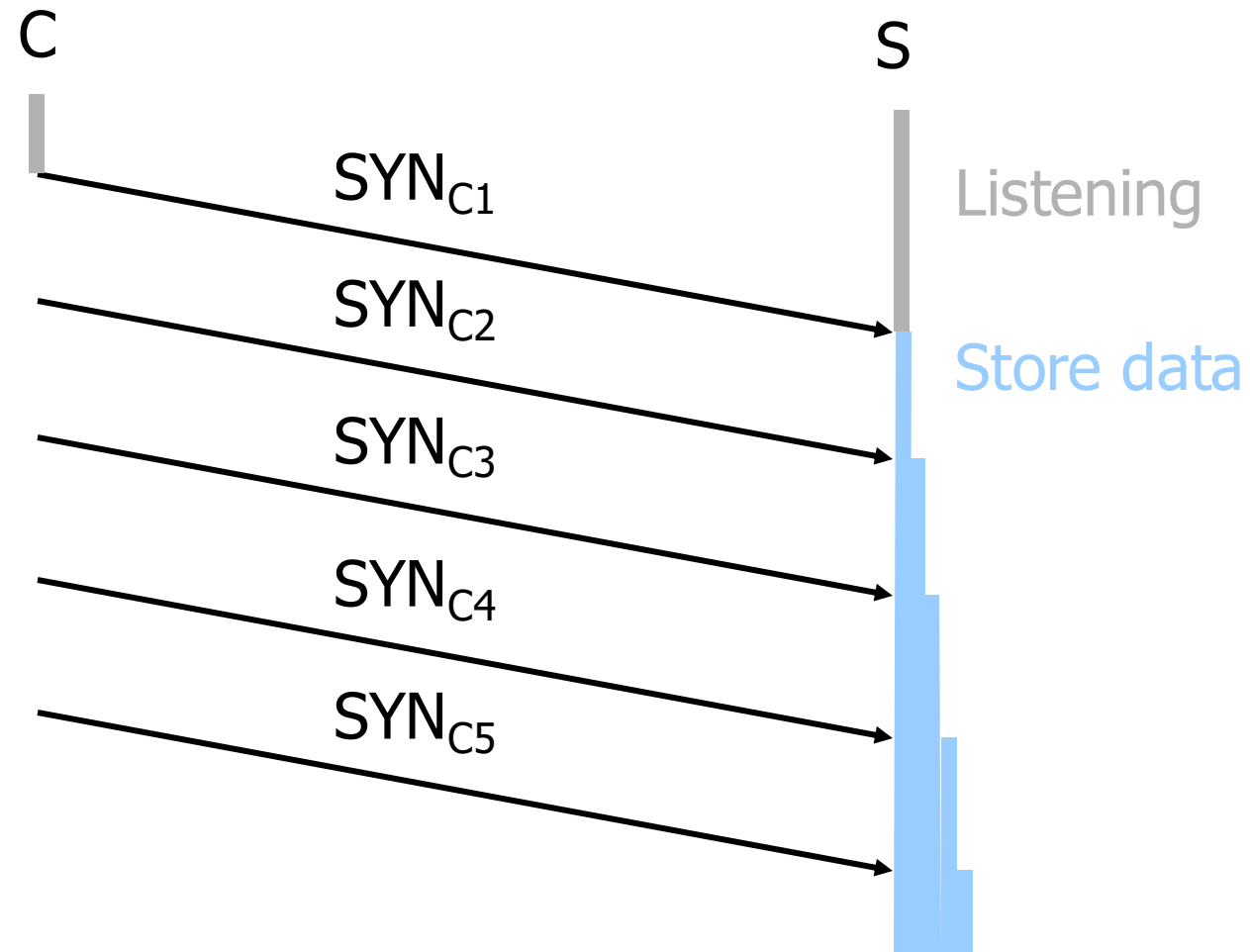
# DoS vulnerability caused by session hijacking

- Suppose attacker can guess seq. number for an existing connection:
  - Attacker can send Reset packet to close connection. Results in DoS.
  - Naively, success prob. is  $1/2^{32}$  (32-bit seq. #'s).
  - Most systems allow for a large window of acceptable seq. #'s
    - Much higher success probability.
- Attack is most effective against long lived connections, e.g. BGP.

# Categories of Denial-of-service Attacks

|          | Stopping services                                                                                                             | Exhausting resources                                                                                                                                                        |
|----------|-------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Locally  | <ul style="list-style-type: none"><li>• Process killing</li><li>• Process crashing</li><li>• System reconfiguration</li></ul> | <ul style="list-style-type: none"><li>• Spawning processes to fill the process table</li><li>• Filling up the whole file system</li><li>• Saturate comm bandwidth</li></ul> |
| Remotely | <ul style="list-style-type: none"><li>• Malformed packets to crash buggy services</li></ul>                                   | <ul style="list-style-type: none"><li>• Packet floods (Smurf, SYN flood, DDoS, etc)</li></ul>                                                                               |

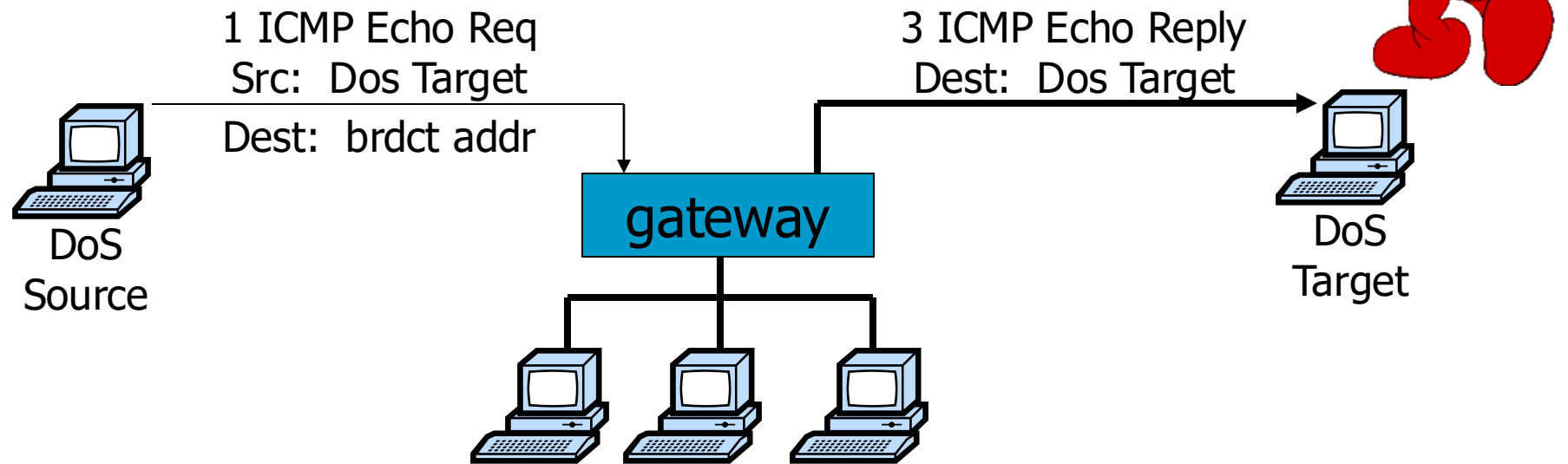
# SYN Flooding



# SYN Flooding

- Attacker sends many connection requests
  - Spoofed source addresses
- Victim allocates resources for each request
  - Connection requests exist until timeout
  - Old implementations have a small and fixed bound on half-open connections
- Resources exhausted  $\Rightarrow$  requests rejected
- No more effective than other channel capacity-based attack today

# Smurf DoS Attack

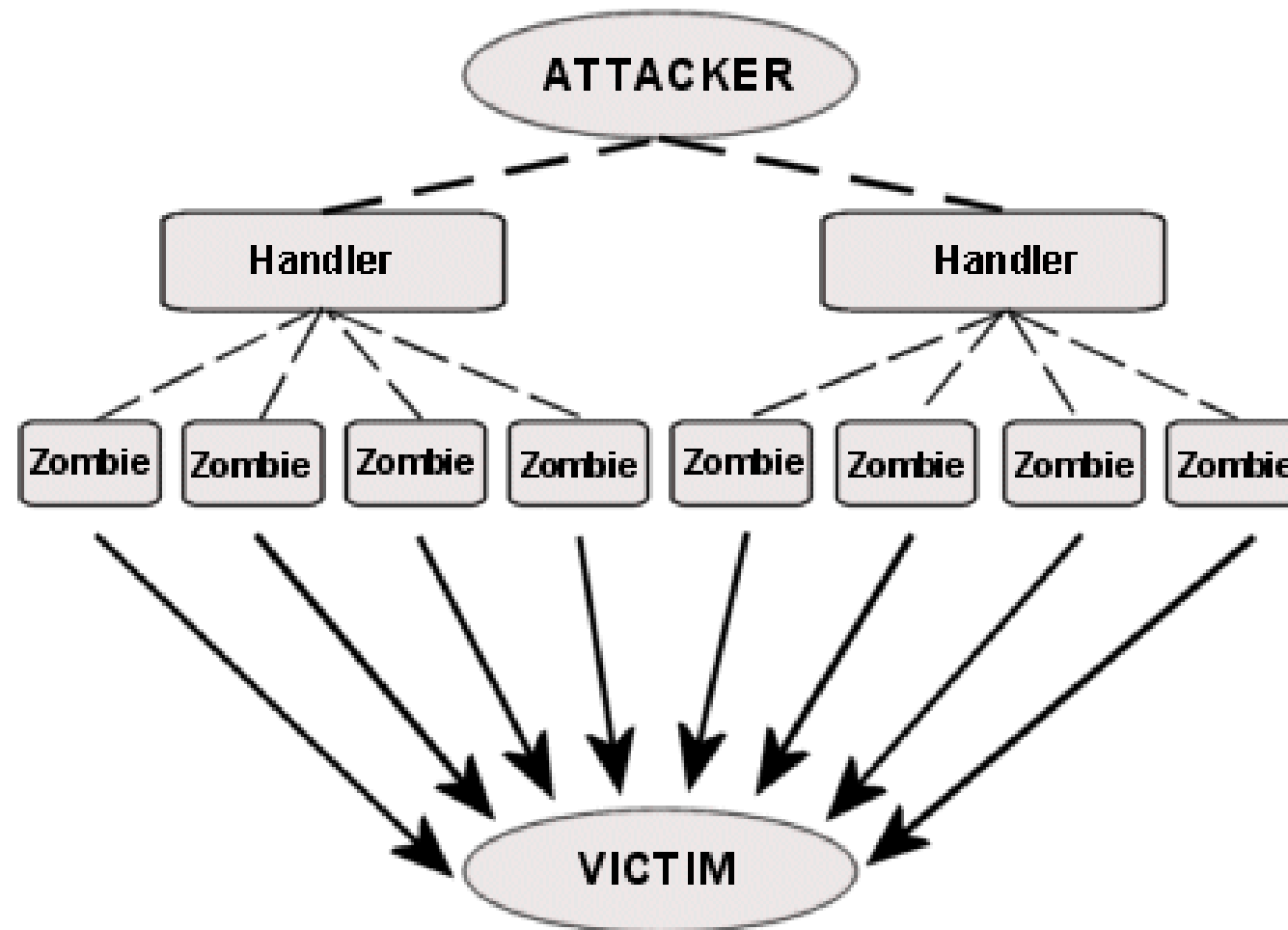


- Send ping request to broadcast addr (ICMP Echo Req)
- Lots of responses:
  - Every host on target network generates a ping reply (ICMP Echo Reply) to victim
  - Ping reply stream can overload victim

Prevention: reject external packets to broadcast address

# Distributed DoS (DDoS)

Architecture of a DDoS Attack



# Hiding DDoS Attacks

- Reflection
  - Find big sites with lots of resources, send packets with spoofed source address, response to victim
    - PING => PING response
    - SYN => SYN-ACK
- Pulsing zombie floods
  - each zombie active briefly, then goes dormant;
  - zombies taking turns attacking
  - making tracing difficult

# Cryptographic network protection

- Solutions above the transport layer
  - Examples: SSL and SSH
  - Protect against session hijacking and injected data
  - Do not protect against denial-of-service attacks caused by spoofed packets
- Solutions at network layer
  - Use cryptographically random ISNs [RFC 1948]
  - More generally: IPsec
  - Can protect against
    - session hijacking and injection of data.
    - denial-of-service attacks using session resets.

# Domain Name System

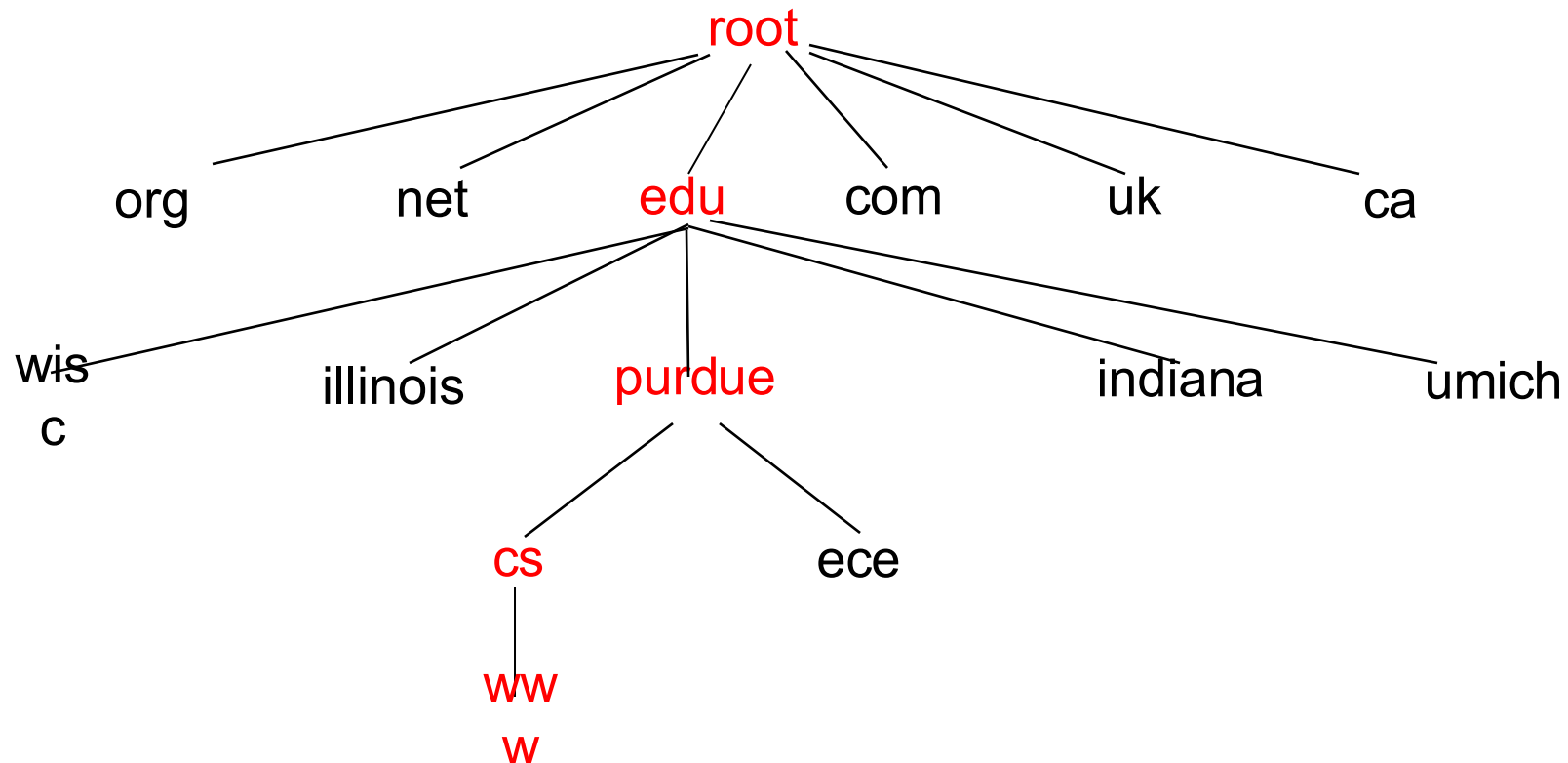
- Translate host names to IP addresses
  - E.g., `www.xyz.com` ➔ `74.125.91.103`
  - Why is needed?
    - E.g. `akami`.
- And back
  - From IP addresses to DNS name

# DNS is a Distributed Database

- Information is stored in a distributed way
- Highly dynamic
- Decentralized authority

# Domain Name System

- Hierarchical Name Space



# Domain Name System



# Domain Name Servers

- Top-level domain (TLD) servers:
  - responsible for com, org, net, edu, etc, and all top-level country domains, e.g. uk, fr, ca, jp.
  - Network Solutions maintains servers for “.com”
- Authoritative DNS servers:
  - organization’s DNS servers, providing authoritative hostname to IP mappings for organization’s servers.
  - can be maintained by organization or service provider.

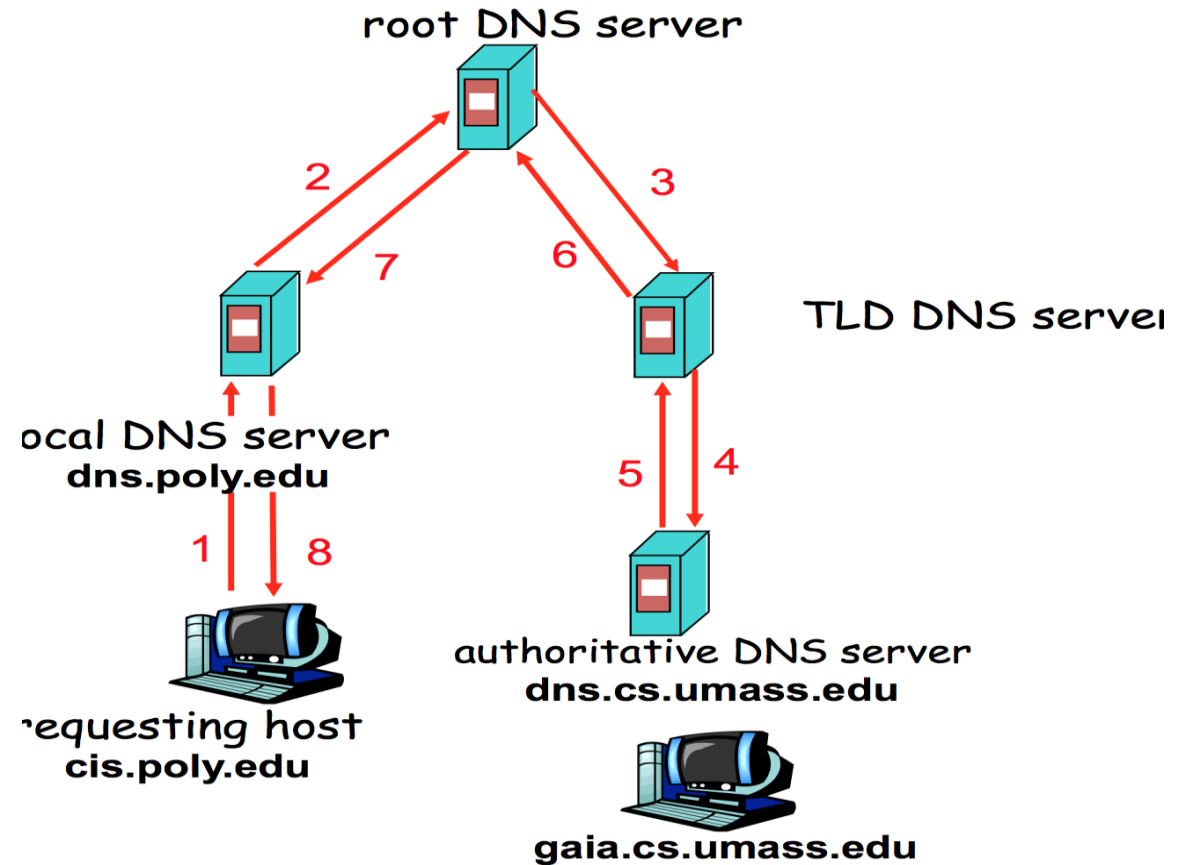
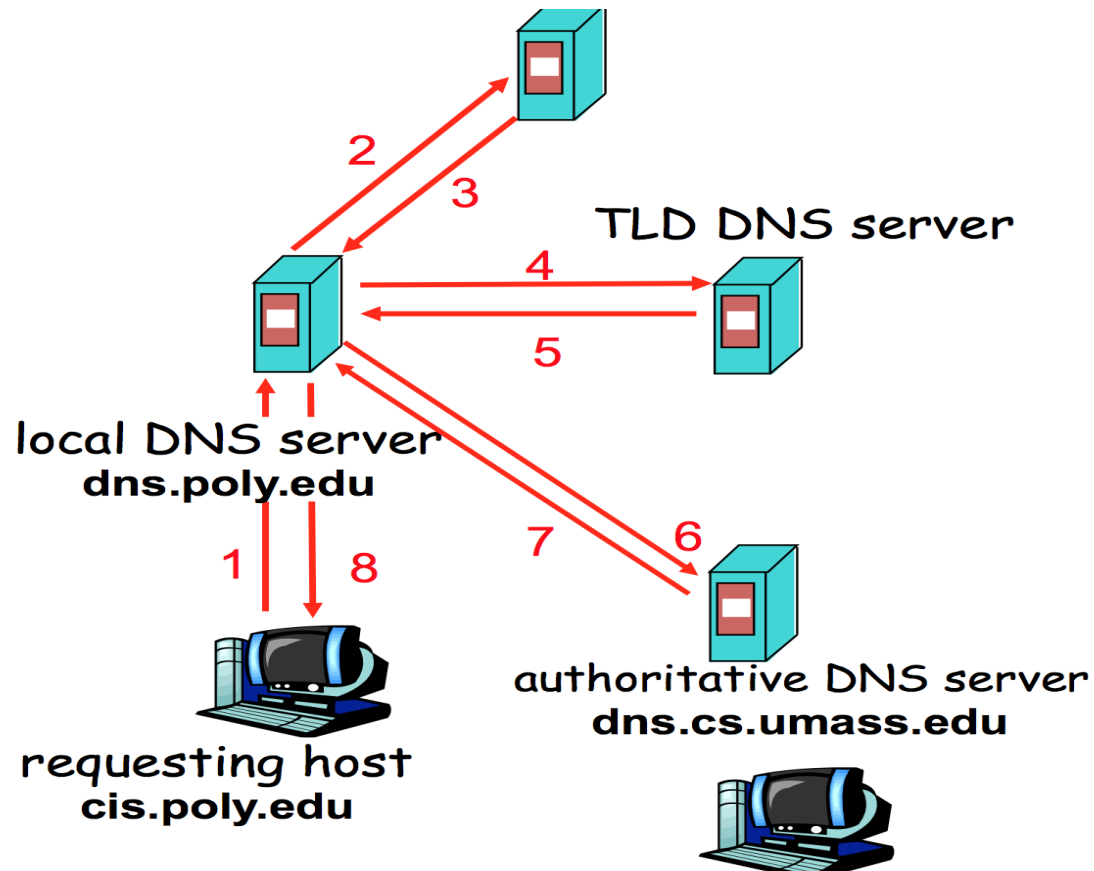
# Domain Name Servers - 2

- Local Name Server
  - does not strictly belong to hierarchy
  - each ISP (residential ISP, company, university) has one.

# DNS Resolving

- When host makes DNS query, query is sent to its local DNS server.
  - acts as proxy, forwards query into hierarchy.
- Two resolving schemes:
  - Iterative, and
  - Recursive.

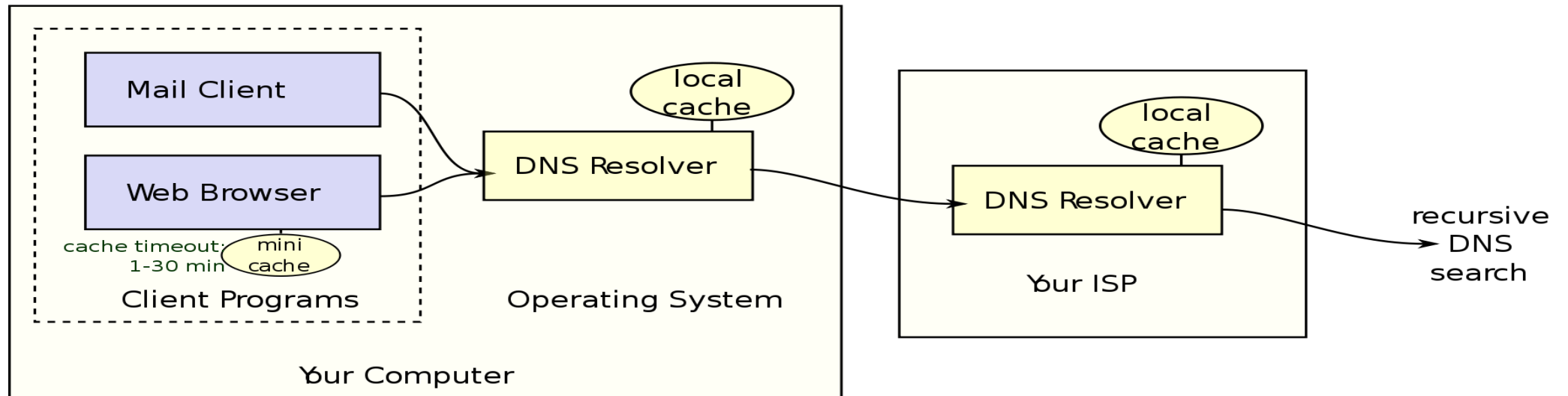
# DNS Resolving - 2



# Caching

- DNS responses are cached
  - Quick response for repeated translations
- Negative results are also cached
  - Save time for nonexistent sites, e.g. misspelling
- Cached data periodically times out
  - Each record has a TTL field

# Caching - 2



# Inherent DNS Vulnerabilities

- Users/hosts typically trust the host-address mapping provided by DNS
  - What bad things can happen with wrong DNS info?
- DNS resolvers trust responses received after sending out queries.
  - How to attack?
- Obvious problem
  - No authentication for DNS responses

# User Side Attack - Pharming

- Exploit DNS poisoning attack
  - Change IP addresses to redirect URLs to fraudulent sites
  - Potentially more dangerous than phishing attacks
    - Why?
- DNS poisoning attacks have occurred:
  - January 2005, the domain name for a large New York ISP, Panix, was hijacked to a site in Australia.
  - In November 2004, Google and Amazon users were sent to Med Network Inc., an online pharmacy

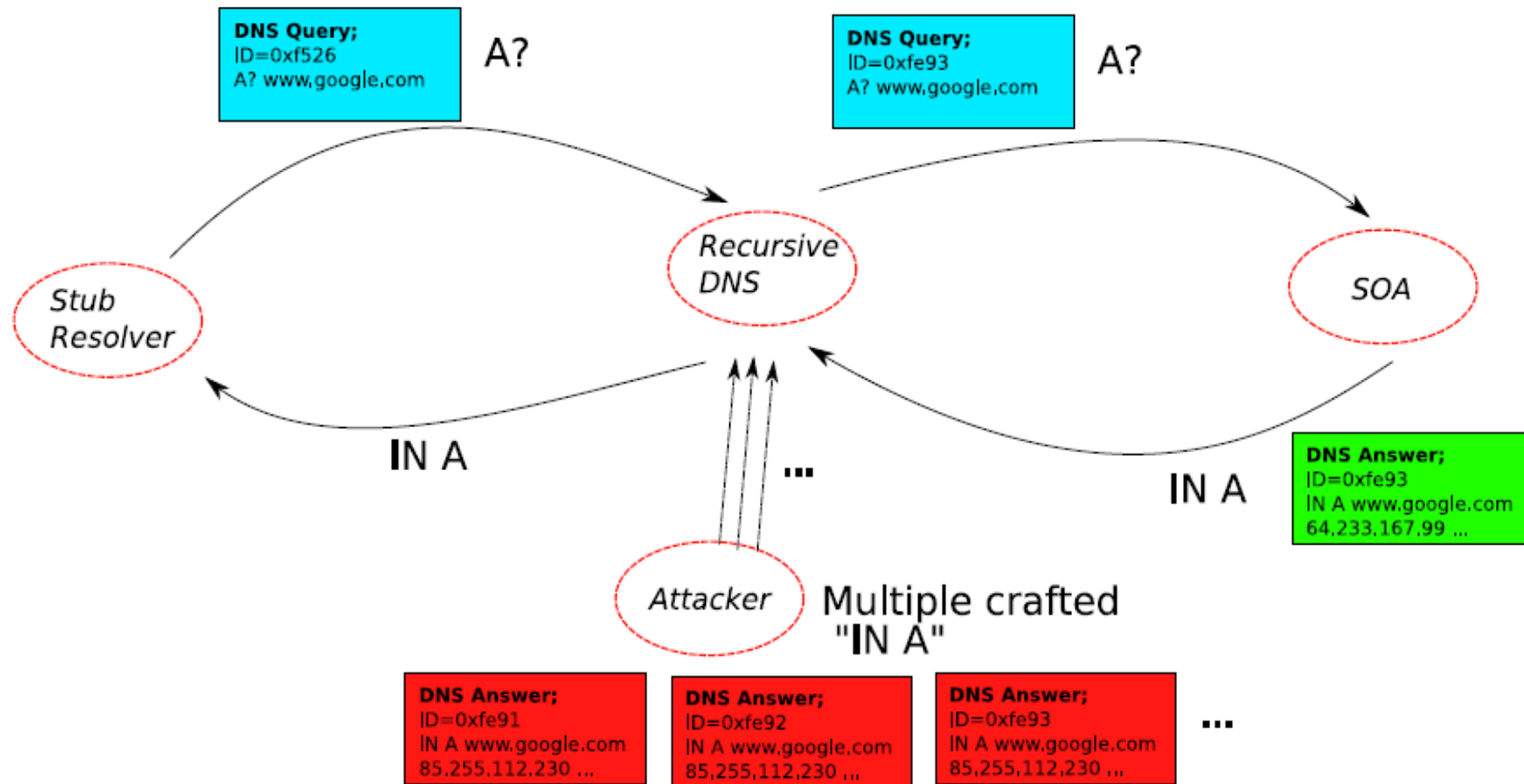
# DNS Cache Poisoning

- Attacker wants his IP address returned for a DNS query
- When the resolver asks ns1.google.com for www.google.com, the attacker could reply first, with his own IP
- What is supposed to prevent this?
- Transaction ID
  - 16-bit random number
  - The real server knows the number, because it was contained in the query
  - The attacker has to guess

# DNS cache poisoning - 2

- Responding before the real nameserver
  - An attacker can guess when a DNS cache entry times out and a query has been sent, and provide a fake response.
  - The fake response will be accepted only when its 16-bit transaction ID matches the query
  - CERT reported in 1997 that BIND uses sequential transaction ID and is easily predicted
    - fixed by using random transaction IDs

# DNS cache poisoning: Racing to Respond First



# Firewall and Network Segregation

# Types of firewalls

- Classification based on production form
  - Personal Firewall (Desktop firewall): Software solution, personal used
  - Network Firewall (Hardware firewall): Hardware dedicated device
- Classification based on feature
  - Packet filtering:
    - Stateless vs. stateful
  - Application Layer Firewall

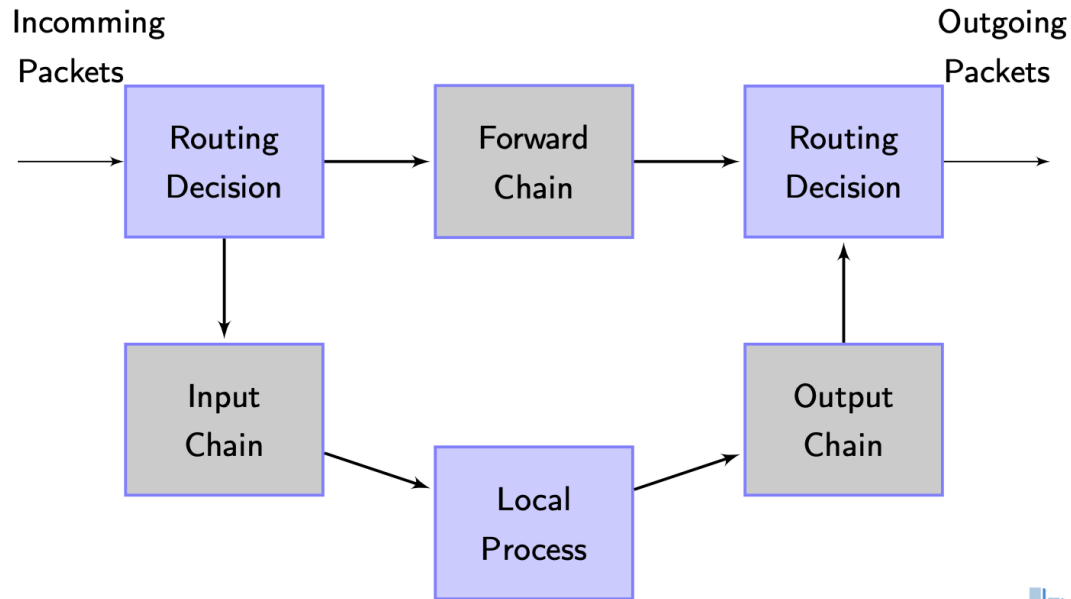
# Packet Filtering Firewall

- Packet Filters have the following characteristics:
  - Checks the IP-Addresses and Ports of incoming and outgoing packets
  - Defines Rules for the incoming, outgoing and forwarded packets
  - Nowadays can be part of Routing devices (e.g. FritzBox)
- Packet filters define Rule Chains and Policies
- Famous and typical : iptables
  - Many software firewalls atop iptables: awall, ufw, firewalld, ...

# Application Layer Firewall

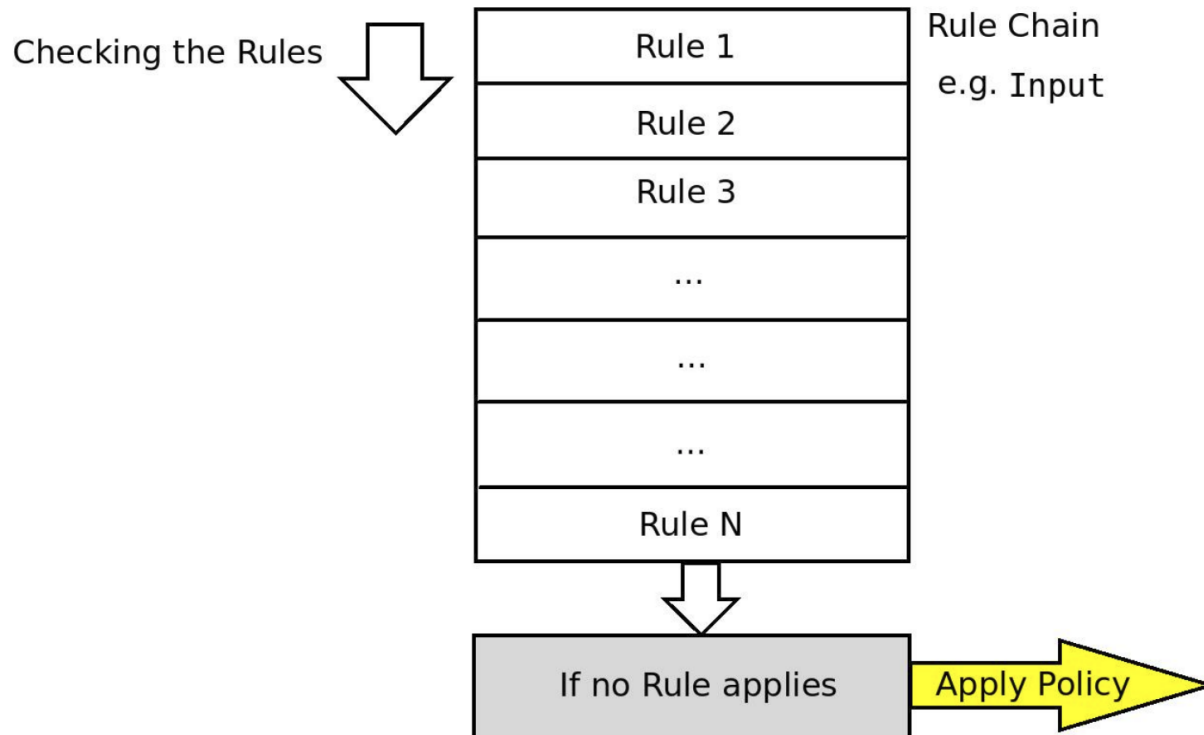
- Application Layer Firewall can be:
  - Proxy Server
  - Interceptor
  - Deep Packet Inspection
  - Accompanied with a set of rules
- Web Application Firewall is a kind of Application Layer Firewall:
  - Typical example:
    - Nginx/Apache with OWASP ModSecurity

# Iptables - Overview



- 03 chains:
  - Chain = list of rules
- INPUT: control inbound packets
- OUTPUT: Control outbound packets
- FORWARD: Control forwarded packets

# Packet processing



- Rule = Matching condition (selectors) + Target Policy
- 4 Policies:
  - DROP
  - ACCEPT
  - REJECT
  - LOG
- Default policy:
  - Policy applied when no rule matched

# Working with iptables

- Cheatsheet:

- <https://andreafortuna.org/2019/05/08/iptables-a-simple-cheatsheet/>

- Some simple tasks:

- List rules: `iptables -nL`

- List rules of a chain: `iptables -nL INPUT`

- List rules with line numbers: `iptables -nL --line-numbers`

- Set default policy for a chain: `iptables -P INPUT ACCEPT`

- Append a rule in a chain: `iptables -A INPUT -s <ip/prefix> -p tcp -j DROP`

- ...

# Testing - netcat

- Server side:
  - `nc -l <port>`
- Client side:
  - `nc <ip> <port>`

# Firewall configuration approaches

- Blacklist:
  - Default accept: `iptables -P INPUT ACCEPT`
  - Selectively block traffics: `iptables -A INPUT <selectors> -j DROP`
- Whitelist:
  - Default DROP: `iptables -P INPUT DROP`
  - Selectively allow traffics: `iptables -A INPUT <selectors> -j ACCEPT`