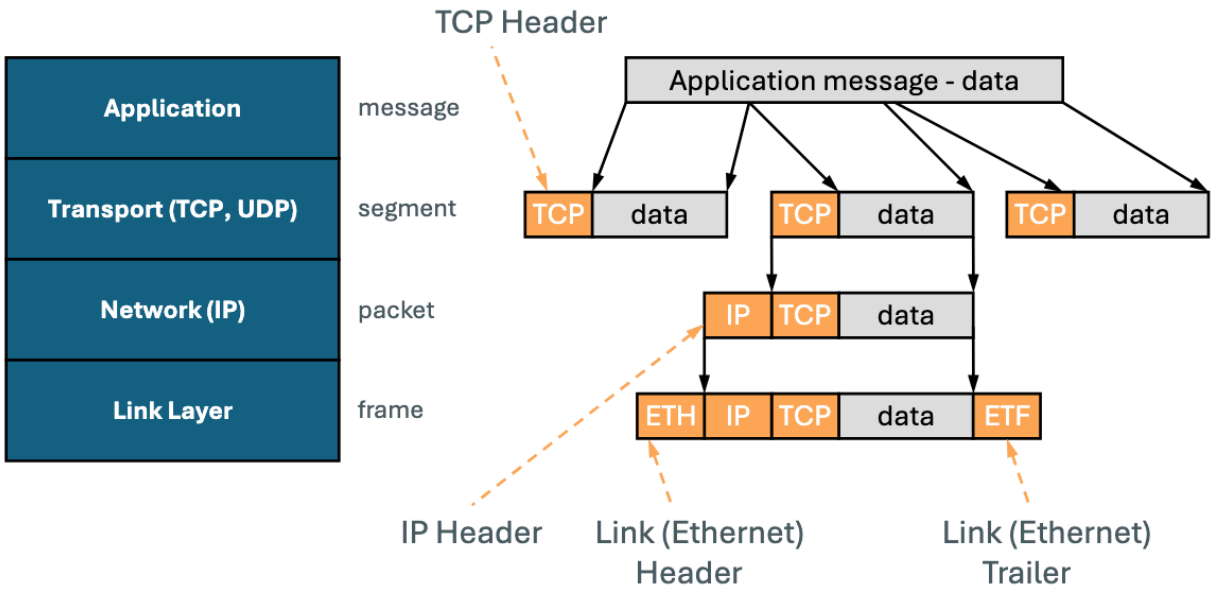
Internet Protocols

- **ARP**
  - MAC discovery
- **ICMP** used by IP layer, sends error messages; TTL exceeded etc
- **TCP, UDP, IP** routes packets
  - OS creates IP packets, filling SRC & DST addresses
  - *data in packet header can be easily changed*
- **BGP** network route discovery
- **DNS** IP address discovery (hostname → IP address)



Attacks

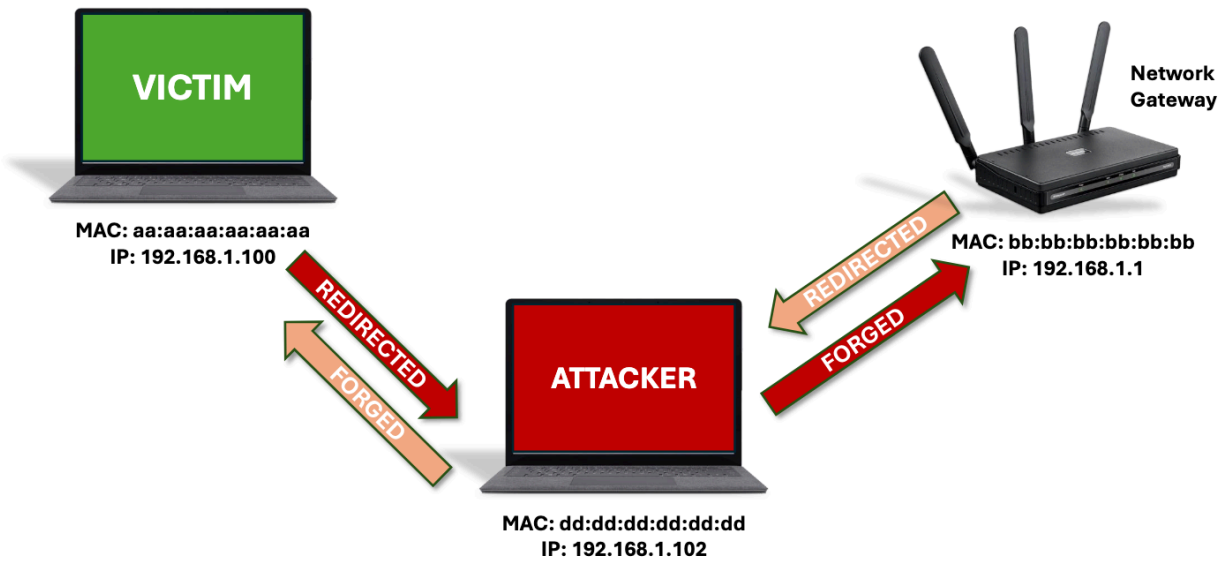
SMTP

- integrity
  - anyone can forge email address (alter content during transit, no TLS)
- confidentiality
  - anyone on a server between sender & recipient can read the email (in PT)
- availability
  - if an email is dropped, neither sender/receiver will know

ARP Spoofing

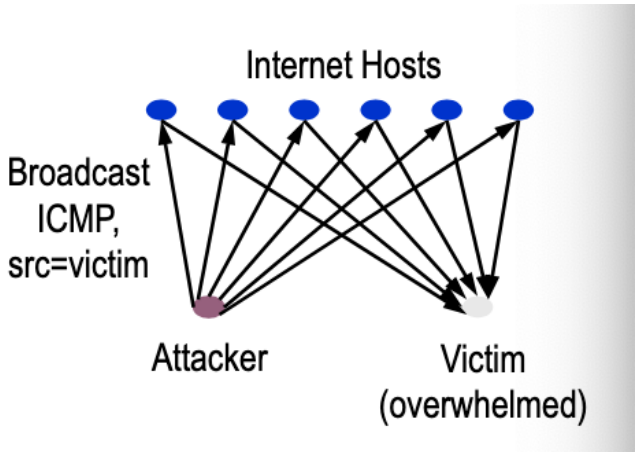
- ARP broadcast to all devices asking which device owns that IP address. All hosts ignore the broadcast except the host that has IP address A, who will respond with its MAC address
- ARP broadcasts are never forwarded outside of its local network subnet, so the attacker must control a machine on a subnet
- **Needs Full Access to a Host**
  - Reply to every ARP broadcast with **fake data**
  - All LAN traffic is redirected to hacked machine; it can then start faking services, stealing passwords, etc.

ARP Spoofing



ICMP Smurf/Amplification Attack

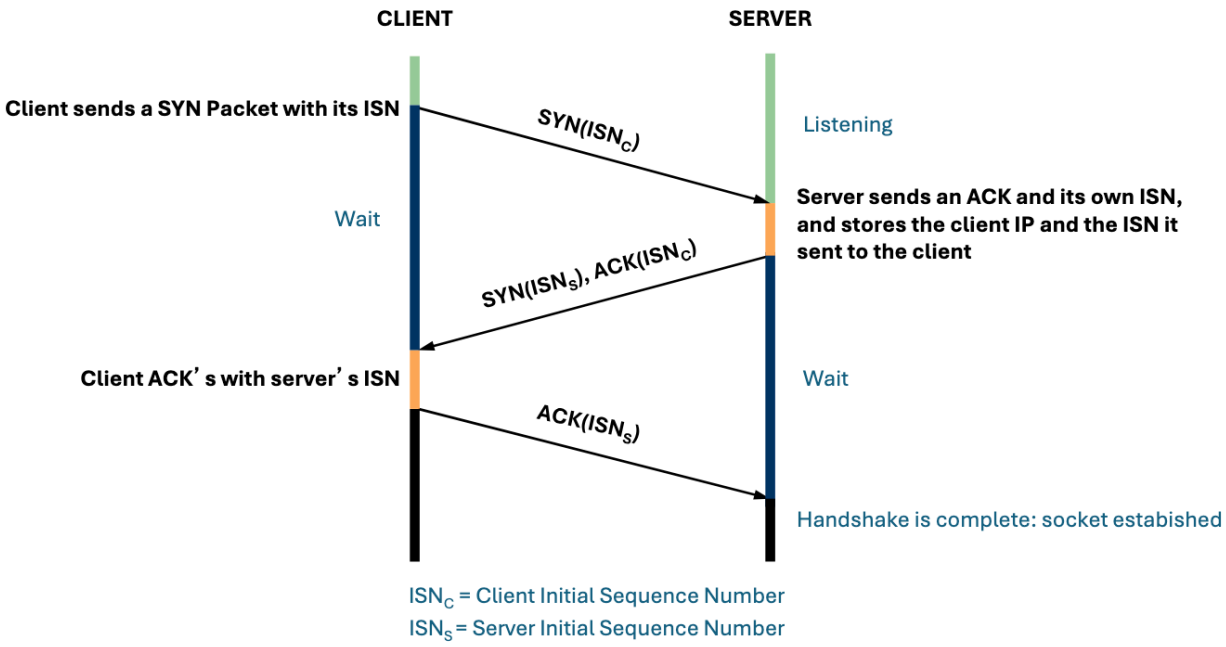
- generates **lots of network traffic** by flooding a victim machine with ICMP packets from many different machines
  - ping (ICMP echo requests) IP broadcast address with spoofed SRC IP, **overloading victim**



Defense

- disable response to ping broadcast (host)
- at router/switch: disable broadcast forwarding

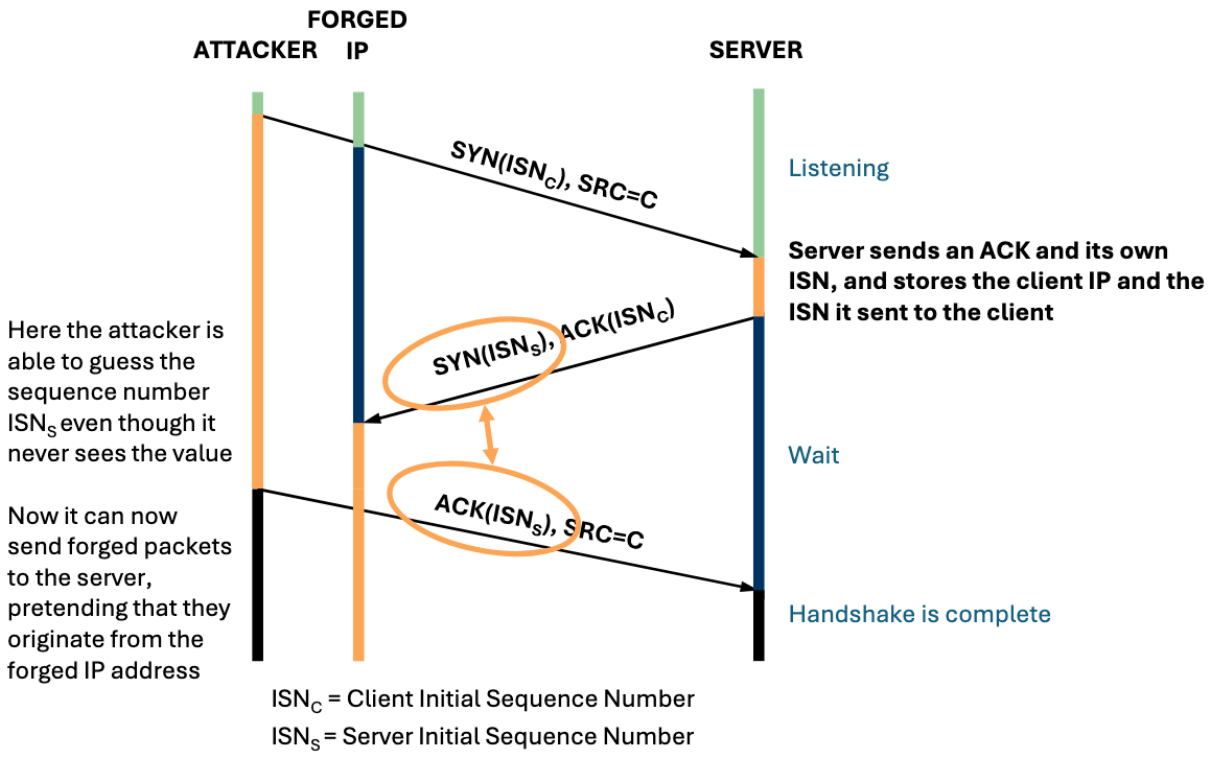
TCP Handshake



Connection Spoofing

- TCP handshake relies on initial SeqNum
  - wrong numbers = connection not set up
- attack with forged IP address (doesn't allow attacker to receive packets)
- if server's initial SeqNum can be guessed, attacker can connect and send packets **to the server** with forged IP address
  - DDoS attacks, bypassing firewalls, spamming, hiding the source of an attack

TCP Connection Spoofing



TCP Reset Attacks

- attacker falsely **terminate TCP connection** resulting in DoS
- PreReq = IP spoofing
  - attacker spoofs sender's connection and sends **RST** (TCP reset)

Defence

- ignoring bogus RST packets

SYN Flooding

- Victim allocates resources for each request until some **timeout**
- Typically, OSs have a fixed bound on these **half-open connections**
- Eventually, the half-open connection **queue resource is exhausted**
- Then, no more requests are accepted, leading to **Denial of Service**

Defense

- reduce half-open connection timeout
- drop half-open connections randomly

SYN-ACK cookies

- client sends SYN
- server responds to client with SYN-ACK cookie
  - ISN_S = HMAC(timer, src addr, src port, des addr, dest port)
  - normal response; **server does not save state**
  - honest client responds with ACK(ISN_S)
    - no changes required at client
  - server regenerates ISN_S and checks that it matches the client's message. If it does, server accepts.

This is a stateless defense because the server doesn't need to store connection information during the initial handshake, making it effective against SYN flood attacks where attackers send many SYN packets without completing the handshake.

BGP Route Announcements

- **Autonomous System** is a connected group of one or more Internet Protocol prefixes using a single routing policy (aka domain
- gateway router between AS communicate using **BGP** to update routing info as *services change*
  - if router down/service relocated, BGP detects and updates the routing info with alternate routes to destinations
  - routes are updated in peer to peer manner, asking neighbours if they have a route to certain destination

Attack

- similar to ARP
- routers don't authenticate peers, BGP packets, or routes advertised by peers
- By routing traffic through their systems, attackers can inspect, record, or modify traffic before passing it along (man-in-the-middle attack)

In your example:

- AS1 = YouTube
- AS2 = Attacker
- AS3 = Client network (ISP, etc.)

Normally, the legitimate path might be:
AS3 → [some transit providers] → AS1 (YouTube)

But the attacker (AS2) exploits BGP by advertising either:

1. A more specific route: BGP prefers more specific routes (smaller IP blocks), so if YouTube advertises a /24 block, an attacker advertising a /25 block within it would be preferred.
2. A shorter path: The attacker claims they have a more direct route to YouTube than what actually exists.

When the attacker's BGP advertisements are accepted:
AS3 → AS2 (Attacker) → AS1 (YouTube)
or possibly just:
AS3 → AS2 (Attacker) [traffic never reaches YouTube]

The client network (AS3) believes this is the best path based on BGP's routing decision logic. The attacker can then intercept all that traffic, potentially performing surveillance, modification, or simply dropping it.

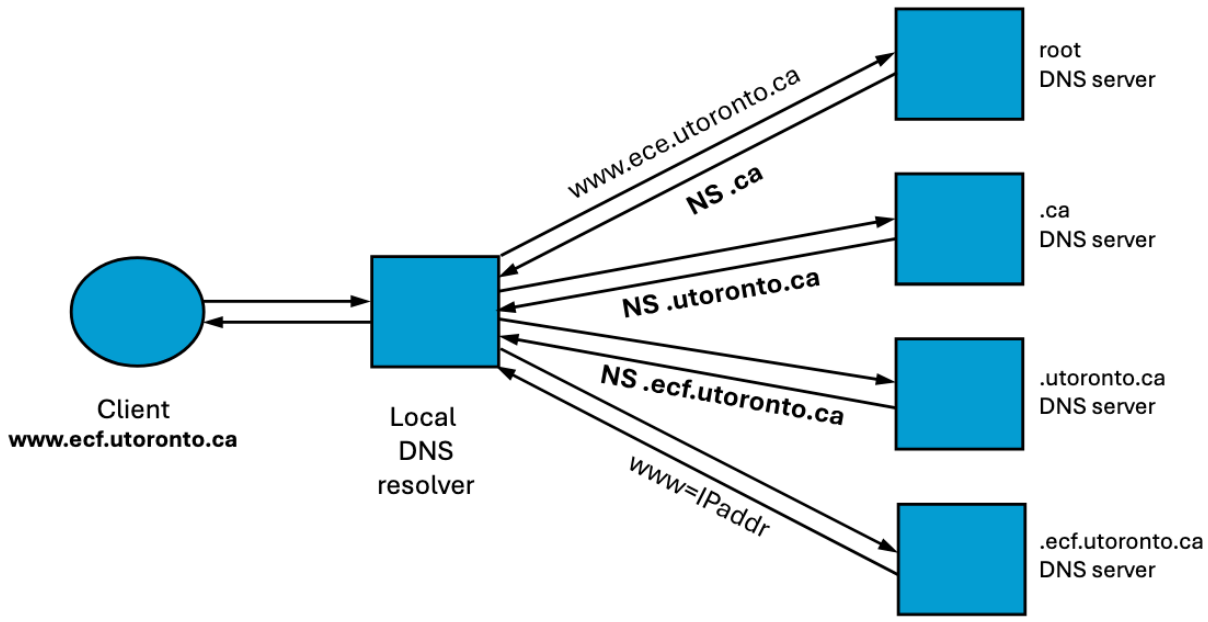
DNS

- **hierarchical naming system**
- maps hostnames to IP addresses using a set of **authoritative name servers**
  - each domain has 1+ NS for DNS mapping for its domain
  - it can assign other NS for sub-domains
- DNS is distributed
  - avoid single register continually consulted/updated

    - single point of failure, extreme bottleneck (DNS queries)

NS for **utoronto.ca** is **ns1.d-zone.ca** and **ns2.d-zone.ca**
NS for **ecf.utoronto.ca** is **ns2.utoronto.ca** and **ns7.utoronto.ca**

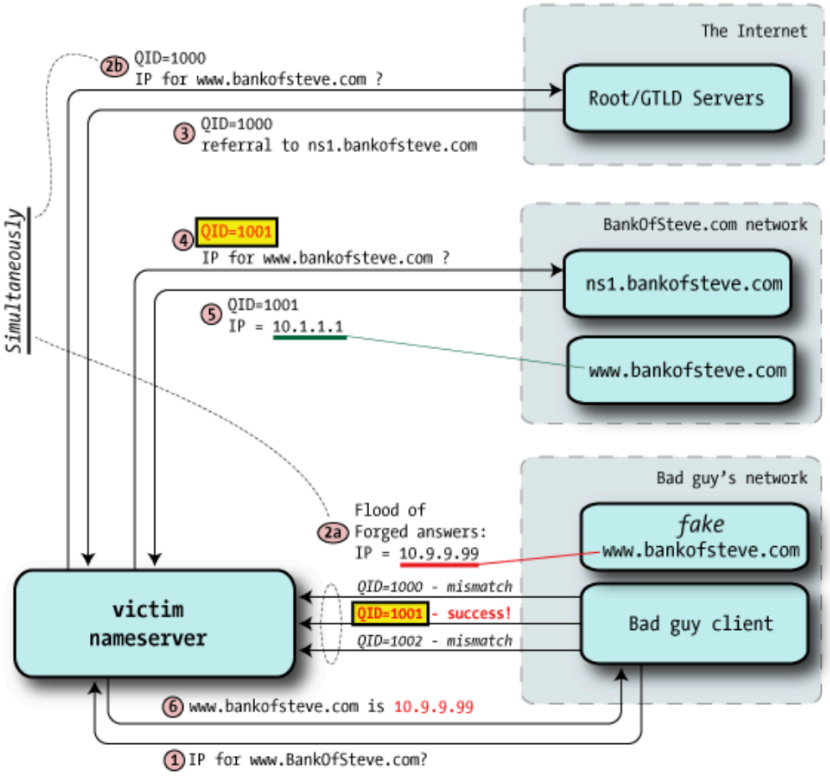
- client DNS query local DNS software (**resolver**)
    - resolver queries NS at top level, replies with info about authoriative server (name OR IP) one level down
    - resolver repeats, until IP addr is returned
    - each query has **unique query id**
        - used in all DNS messages throughout the entire resolution process



- DNS responses are cached at **name server**
    - quick response for repeated queries (negative / failures are cached also)
    - cached until TTL expires
    - shorter TL allows faster recovery during failure/changes

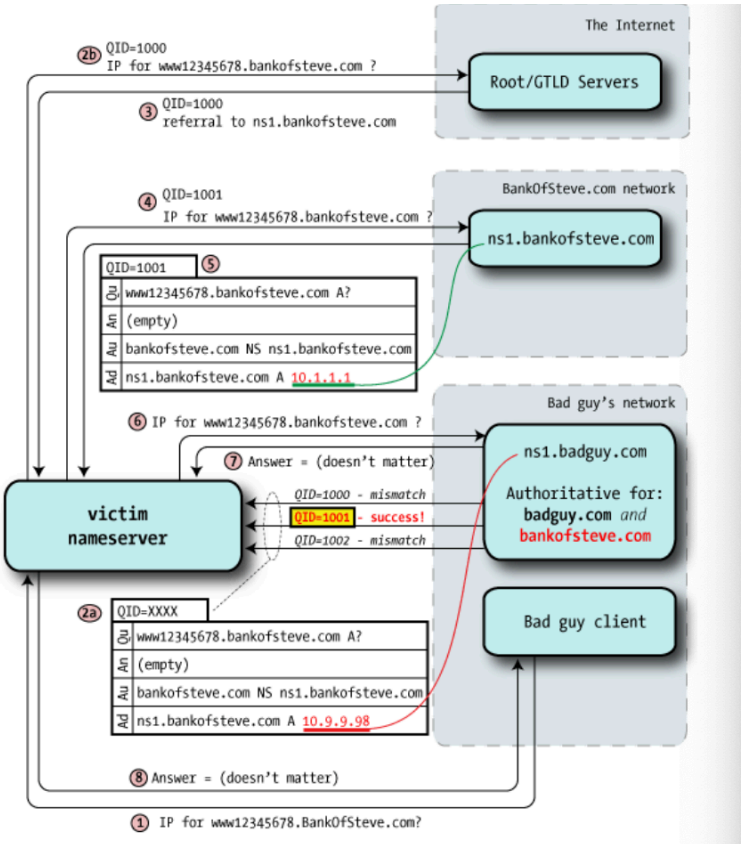
DNS Cache Poisoning

1. Attacker initiates DNS request for **www.bankofsteve.com** to a victim nameserver
2. Then it sends several DNS responses with **spoofed IP addresses** and various guessed **Query IDs**
3. If a spoofed response has a correct query id, the victim name server then has a poisoned cache entry for **www.bankofsteve.com**



Previous attack has short window of vulnerability due to caching (TTL). In this attack:

1. The attacker performs several DNS requests for **wwwN.bankofsteve.com** to a victim nameserver where **N** changes with each request
2. Then they send several DNS responses with a **spoofed authoritative name server**
3. If attack succeeds, attacker will be able to provide forged replies for any future local lookups for the entire **bankofsteve.com** domain



- The first valid response wins. If the legitimate answer arrives first, the resolver accepts it and ignores later responses.
- Victim domain → Attacker's IP address

DNS Rebinding

- Attacker domain → Victim's IP address

    - bypassing the browser's same origin policy
    - an attacker's IP and victim IP appear to belong to same domain!
- **Browser** attacker gets client to visit attacker site
- **A** attacker controls DNS of site, returns DNS response with short TTL. Attacker site serves web page with malicious script
- **B** script makes a request to attacker's site. Browser makes another DNS query, reaching attacker's DNS (cached entry has short TTL)
- **A** attacker's DNS returns IP of victim web server
- **B** victim's web server and attacker's web server appear to be in same origin
    - if browser located within an intranet, rebinding attack allows internal access to company machines

> In web security, the "same-origin policy" is enforced by browsers to prevent websites from different origins from accessing each other's data. An origin is defined by the combination of protocol (HTTP/HTTPS), domain name, and port number.
>
> In this DNS rebinding attack:
>
> 1. Initially, the browser loads content from the attacker's domain (e.g., evil.com)
> 2. Later, when the browser makes another request to what it thinks is still evil.com, the DNS has been rebound to point to a victim's internal IP address (e.g., 192.168.1.100)
> 3. From the browser's perspective, both requests are to the same origin (evil.com), even though the second request actually goes to a completely different server
>
> The browser considers these to be the same origin because the domain name remains unchanged (evil.com), despite the fact that the IP address has changed. This is the key security bypass - the browser's same-origin policy is based on the domain name, not the IP address.
>
> This allows the attacker's JavaScript (which was loaded from the initial evil.com page) to make requests to and read responses from the internal victim server, which would normally be blocked by same-origin policy.

Defenses

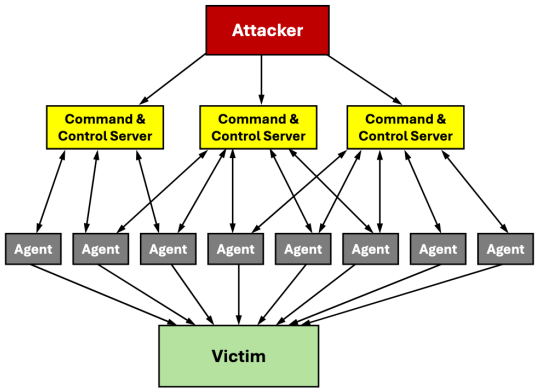
- browsers use domain names instead of IP to enforce **same origin policy**
    - domain names change IP for load balancing
- browsers can pin DNS/IP mapping to the value in the first DNS response
- block resolution of external names into local IP addresses at a local nameserver

DoS

- denial of service - attack making a system inaccessible to legitimate users
    - consume as much resource as possible
    - BW/memory
- **bandwidth attack** - flooding the server with packets, so the attacker needs more bandwidth than victim

Distributed DoS

- build up large number of compromised hosts, use to simultaneously attack single target
    - image bombs
    - lengthy queries
    - email spamming



Common Defences

- cryptographic protocols
  - protect against spoofing attacks and injected data
  - fail to protect DoS

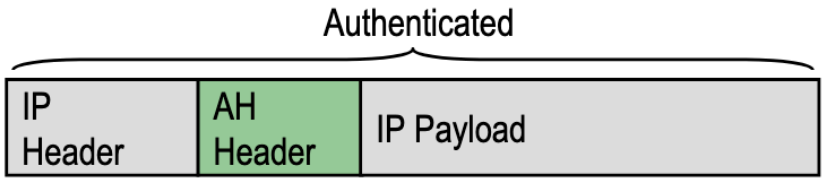
Application Layer SSL, SSH
Transport Layer cryptographically random ISNs for TCP/IP
Network Layer IPSec

IPSec

- at IP layer, provide...
  - message confidentiality
  - message integrity
  - source authentication
  - protection against replay
- 2 protocols
  - **Authenticated Headers**(AH)
    - provides 3 except confidentiality
  - **Ecapsulating Security Payload**(ESP)
    - provides all

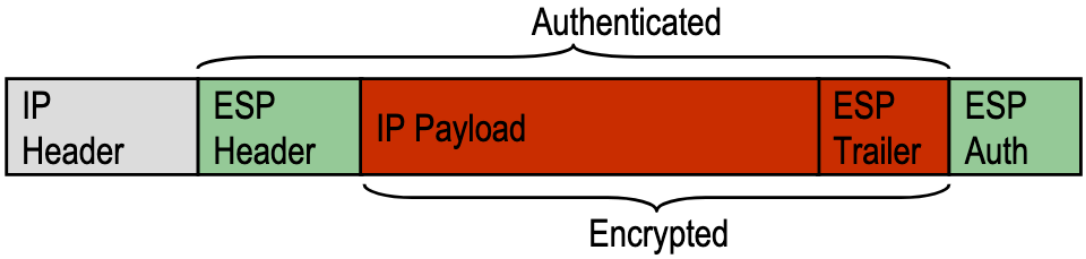
Authenticated Headers

- Protects the IP packet (except IP header fields that are altered during transit) using a MAC stored in AH Header



Encapsulating Security Payload

- Payload is encrypted to protect contents



IPSec Modes

- doesn't require routes between endpoints speak IPSec
  - **Transport Mode**
    - 2 endpoints support IPSec, intermediary routers do not
    - encrypts/authenticates **packet payload**
  - **Tunnel Mode**
    - endpoints do not support IPSec, endpoint gateways do
    - encrypts/authenticates packet **header & payload** and encapsulates it in another regular IP packet

IPSec vs. SSL/TLS

Choosing IPSec or SSL depends on security needs

- If a specific service is required and is supported by SSL it is better to select SSL
- If access to an entire network is required, VPN software/device using IPSec is a good choice

Security

- SSL can provide more security because if one connection is compromised, others are not
- SSL can provide better access control since with IPSec allows access to entire network

IPSec vs. SSL/TLS

Compatibility and deployment

- SSL does not require special client software (already in browser) and configuration
- SSL is more compatible with firewalls, unless IPSec and firewall are on same device
- IPSec often doesn't interoperate well (poorly standardized implementations), so both sides of the connection are required to have the same vendor's devices
- IPSec supports pre-shared keys so PKI is not needed

Overhead

- IPSec has lower overhead because many users can use the same secure channel while SSL requires connection establishment for each channel

For more info see Bruce Schneier's "Criticism/Evaluation of IPSec" on course web site.

Defenses

PKI

- public key (~100 bytes)
  - will not work from efficiency or math standpoint