Network and Web Security

Prepared By

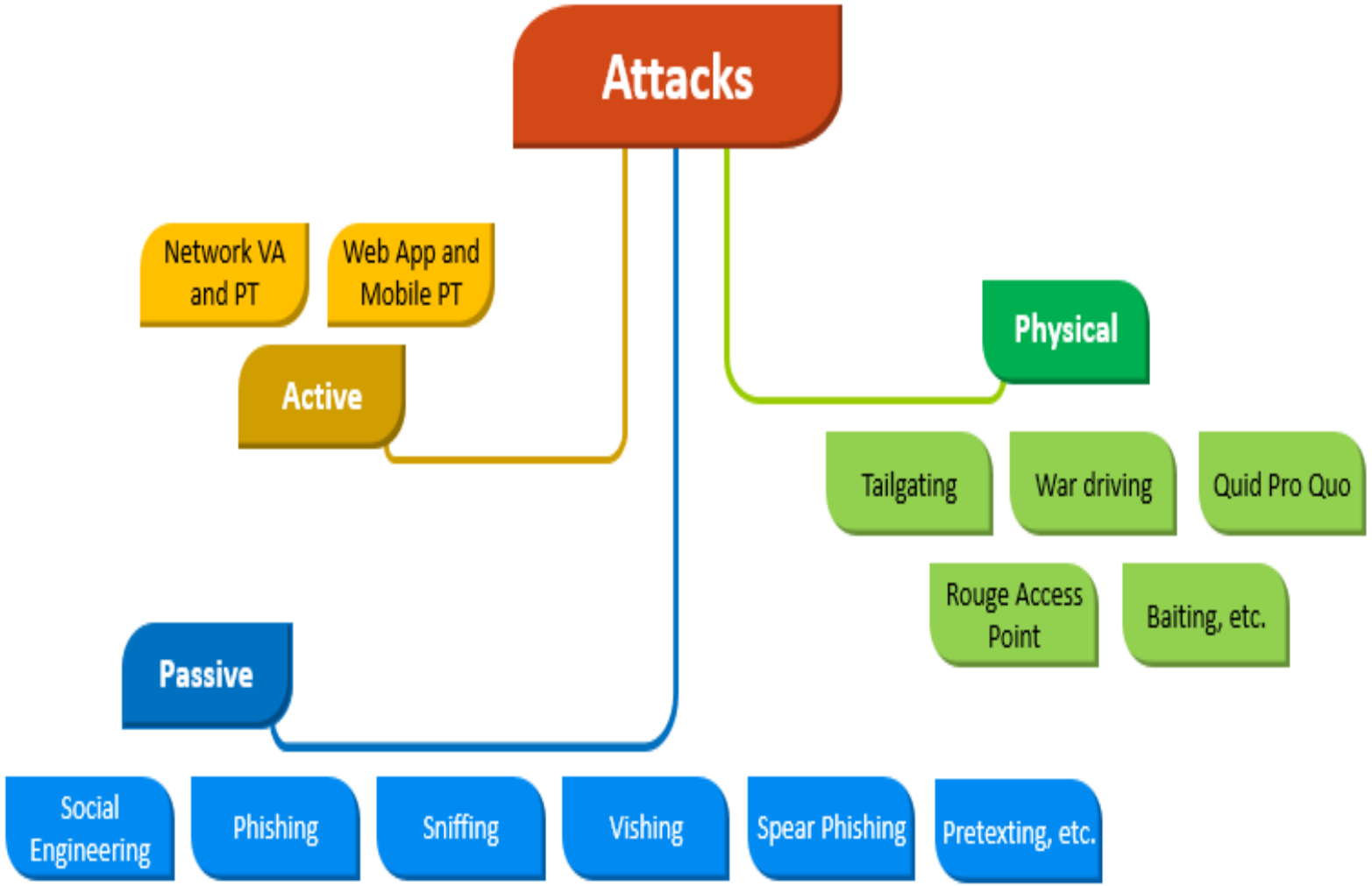
-Anooja Joy

Reconnaissance of network- the Eagle's Eye of Cyber Security

- ***Network reconnaissance*** or simply **recon** *is a term for testing for potential vulnerabilities in a computer network by acquiring information about network using* **Information Gathering.**
- It may be a **legitimate activity** by the network owner/operator, seeking to protect it or to enforce its acceptable use policy and studying how attackers perpetrate their attacks.
- It also may be a **precursor to external attacks** on the network. An attacker uses reconnaissance to discover and analyze the targets of his attack.
- **Example:** Port Scanning(Nessus, NMAP), Ping Sweeps, Sniffing
- It plays a key role in **penetration testing**.

Motives of Network Reconnaissance

- To discover **IP Address of Hosts**.
- To identify **accessible UDP** and **TCP ports**.
- To identify **OS type**.
- To discover **active hosts in network.**
- To identify **network structure.**
- To identify **applications** and **services** are running on devices on the network
- To **discover vulnerabilities** that exist that can be exploited
- To discover **Trust relationships, File permissions** and **User account information**



TYPES OF RECONNAISSANCE ATTACKS

- Social Engineering;
- Site (Physical) Reconnaissance;
- Internet Reconnaissance;
- IP/Network Reconnaissance;
- DNS Reconnaissance.

Reconnaisance Procedure

1. **Information Gathering:** Getting information on network environment like website organization, owner,location, contact person, pjhone number
   - **Tools: whois lookup,domain tools,geek tools,DNSStuff, search tools(google,yahoo), nslookup**
2. **Network Mapping:** Creating blueprint of organization network
   - **Tools:** packet sniffing tools like wireshark, usage of ping, traceroute,tracert,cheops network mapping tool
3. **Port Scanning:** Finding open ports that are accessible and underlying applications.
   - **Tools: namap, nessus and perform** TCP SYN scan, XMAS TREE SCAN,NULL SCAN, FIN SCAN, TCP Connect, UDP Scan, Ping sweep
4. **OS detcetion/TCP stack fingerprinting:** To identify target OS and system responses. It take advantage of ambiguity of how to handle illegal combination of TCP code bits that is found in RFC. Each OS responds to illegal combinations in different ways.
   - **Tools: namap, P0F,XProbe2**
5. **Identifying Active Elements**
   - **Tools:** ping, traceroute latency calculation

Packet Sniffing

- **Packet sniffing** is the act of capturing IP data packets of data flowing across a computer network. Its a passive attack on an ongoing conversation.
- The software or device used to do this is called a **packet sniffer.**
- **Example:** Wireshark, ngrep, ettercap, tcpdump, dSniff
- Data from upper layer **encapsulated** into **IP packet**. **De-encapsulation** of IP packets result in access to sensitive data if protocols like SNMP,SMTP,POP3 are used.
- There are two types of sniffing attacks: **active sniffing** and **passive sniffing**.

Types of Sniffing

- **Active sniffing** – this is sniffing that is conducted on a **switch**. A **switch** learns a **CAM table** that has the **mac addresses of the destinations**. Based on this table the switch is able to decide what network packet is to be sent where. In **active sniffing**, **the sniffer will flood the switch with bogus ARP requests so that the CAM table gets full.** CAM keeps track of which host is connected to which port. Once the CAM is full it will send the network traffic to all ports. Now, this is legitimate traffic that gets distributed to all the ports. This way the attacker can sniff the traffic from the switch. A **switch** is a device that connects two network devices together. Switches use the **media access control (MAC) address** to forward information to their intended destination ports. Attackers take advantage of this by injecting traffic into the LAN to enable sniffing.

- **Passive sniffing** – passive sniffing uses **hubs** instead of switches. **Hubs** that exist in physical layer that receives network traffic on one port and then retransmits that traffic on all other ports withot knowing source and destination. All an attacker needs to do is to simply connect to LAN and place sniifer at hub and sniff data traffic in that network.

Packet Sniffing techniques

1. ARP Spoofing
2. IP Spoofing
3. Session Hijacking
4. MAC Fooding
5. ICMP redirection
6. DHCP spoofing
7. Port stealing

COUNTERMEASURES

- Use a VPN all the time
- Always use HTTPS when available
- Never send form data using HTTP
- use encrypted connections. Eg: PGP, SSH
- Use switch instead of Hubs

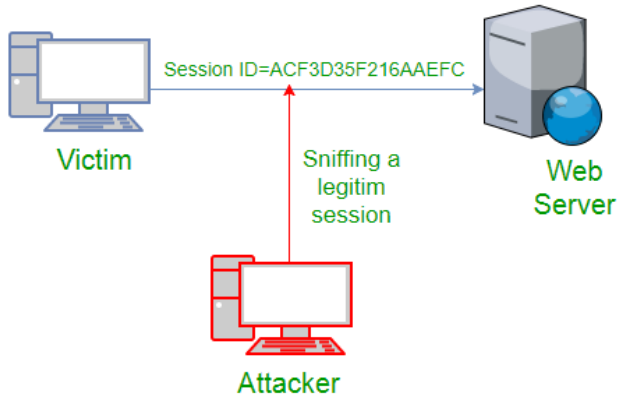
Session Hijacking

Session

- http communication uses many different TCP connections, so the web server needs a method to recognize every user's connections
- Web Server sends a session token(composed of a string of variable width) to the client browser after a successful client authentication.
- The session ID is normally stored within a cookie or URL.
- By using this cookie, only your web server is able to identify who the user is and it will provide content accordingly. No sensitive information in the cookie, just the random ID (non-guessable)
- The Session Hijacking attack compromises the session token by stealing or predicting a valid session token to gain unauthorized access to the Web Server.
- It is also known as **cookie stealing/hijacking/ man in the middle attack**

How Session hijacking can be done?

1. **Session Sniffing: T**he attacker uses a sniffer to capture a valid token session called “Session ID”, then he uses the valid token session to gain unauthorized access to the Web Server.
2. **Cross-site script attack:** The attacker can compromise the session token by using malicious code or programs running at the client-side. Eg:`<SCRIPT> alert(document.cookie);</SCRIPT>`
3. **Session fixation**, where the attacker sets a user's session id to one known to him, for example by sending the user an email with a link that contains a particular session id. The attacker now only has to wait until the user logs in.
4. **IP Spoofing:** gain unauthorized access to the computer with an IP address of a trusted host. Attcker has to obtain the IP address of the client and inject his own packets spoofed with the IP address of client into the TCP session, so as to fool the server that it is communicating with the victim i.e. the original host
5. **Blind Attack: If attacker is not able to sniff packets and guess the correct sequence number** expected by server, brute force combinations of sequence number can be tried.




RBS
The Royal Bank of Scotland
Make it happen

Digital Banking

Dear Valued Customer,

Due to the recent amount of fraudlent messages that targeted our customers recently. Royal bank of scotland has decided to upgrade the security of our online banking digital system. To do this we need the proper verification of each and everyone of our customers. We were unable to reach you by phone thereby sending this email as an alternative to please verify your account information by clicking on the Login button below

LOG IN

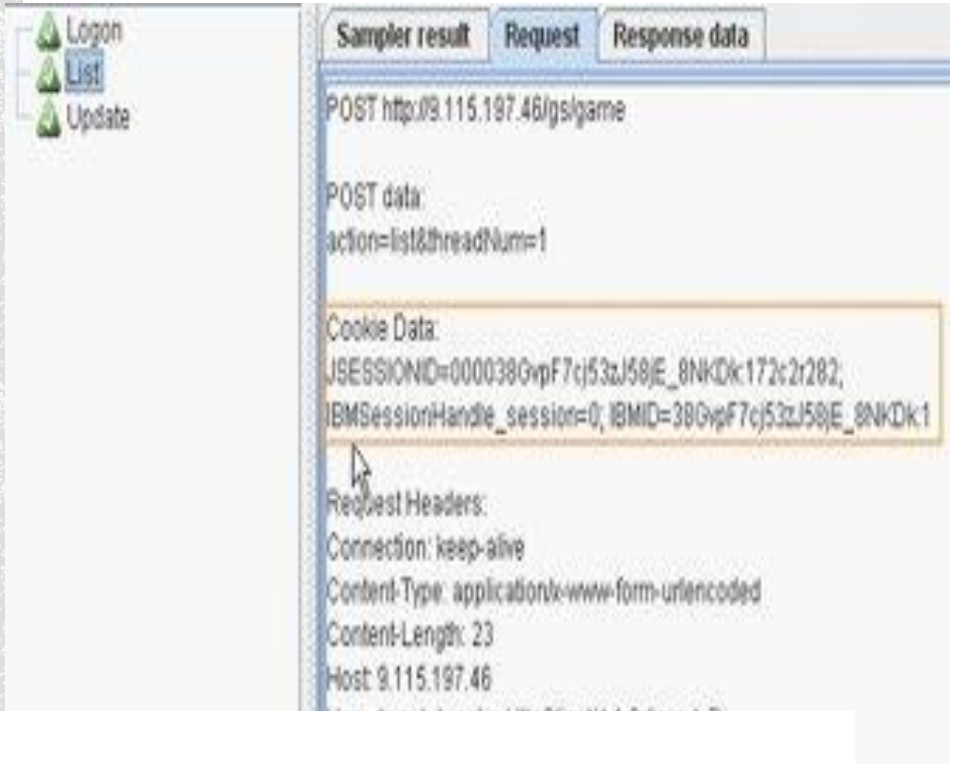
Failure to do this within 24hrs will lead to restricted access to your account for security reasons

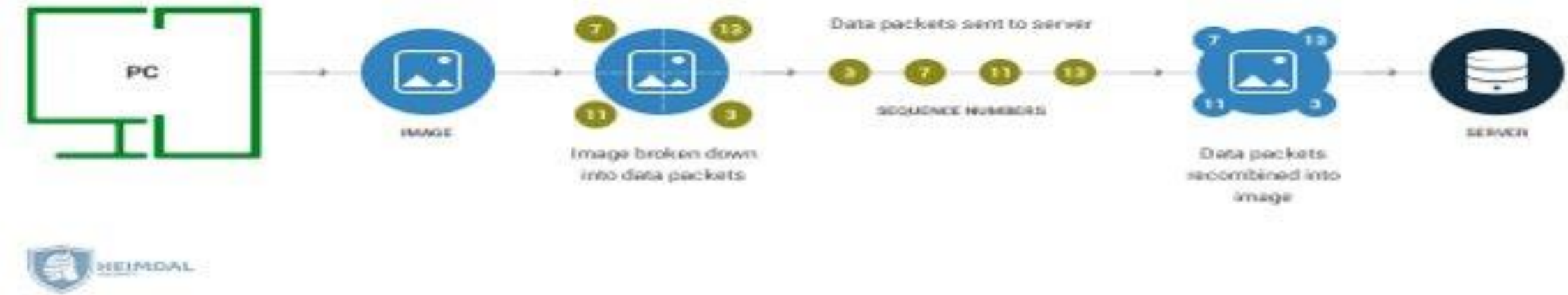
Sorry fo the inconvience

Regards
Royal Bank Of Scotland

Session id and sequence Number

- SessionID is basically the “name” of a particular session.
- Session sequence number is a number assigned to each data packet so the receiving device knows the order used to reassemble the data.





Mitigation

- Network level hijacks can be prevented by **Ciphering the packets** so that the hijacker cannot decipher the packet headers, to obtain any information which will aid in spoofing. This encryption can be provided by using protocols such as IPSEC, SSL, SSH etc.
- Eg: Internet security protocol (IPSEC)

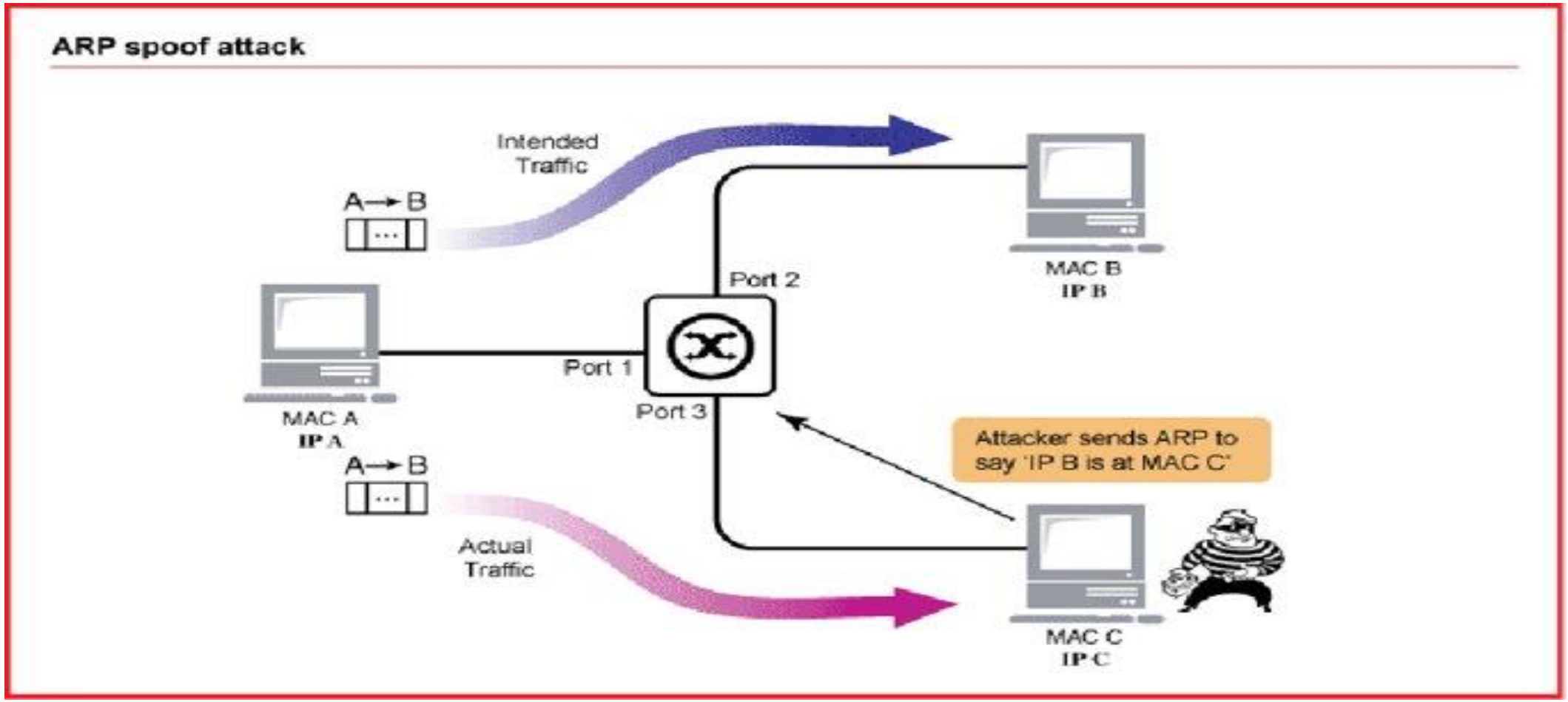
ARP Attacks

- Address Resolution Protocol (ARP) is a **stateless protocol** used for finding **MAC addresses** that corresponds to given **IP addresses of host machine** in a LAN**.**
- An **internet protocol address(IPV4 or IPV6)** is used to uniquely identify a **computer** or device such as printers, storage disks on a computer network.
- **MAC addresses** are used to uniquely identify **network interfaces** for communication at the physical layer of the network. MAC addresses are usually embedded into the **network card**.
- All network devices that need to communicate on the network broadcast ARP queries in the system to find out other machines' MAC addresses. Each machine has its own ARP table/ARP Cache containing mapping between IP address and MAC addres. It is generated automatically and expires after a period of time so that it should be refreshed periodically.
- ARP-stateless protocol, a node does not have a record of ARP requests that it has sent.
- Here is how ARP works −
    1. When one machine needs to communicate with another, it looks up its ARP table.
    2. If the MAC address is not found in the table, the **ARP_request** is broadcasted over the network.
    3. All machines on the network will compare this IP address to MAC address.
    4. If one of the machines in the network identifies this address, then it will respond to the **ARP_request** with its IP and MAC address. A node will accept any ARP reply that it receives (even if it has made no corresponding ARP request)
    5. The requesting computer will store the address pair in its ARP table and communication will take place.

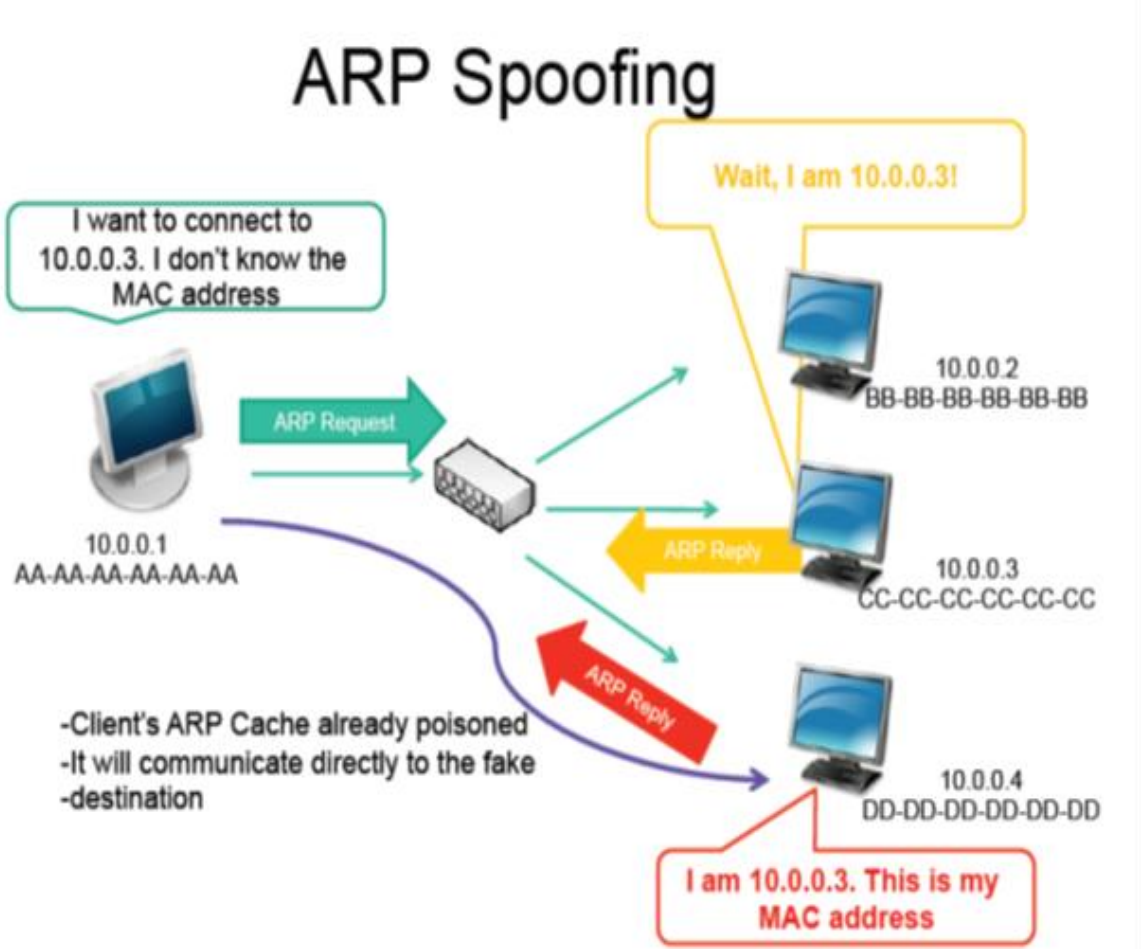
ARP Spoofing

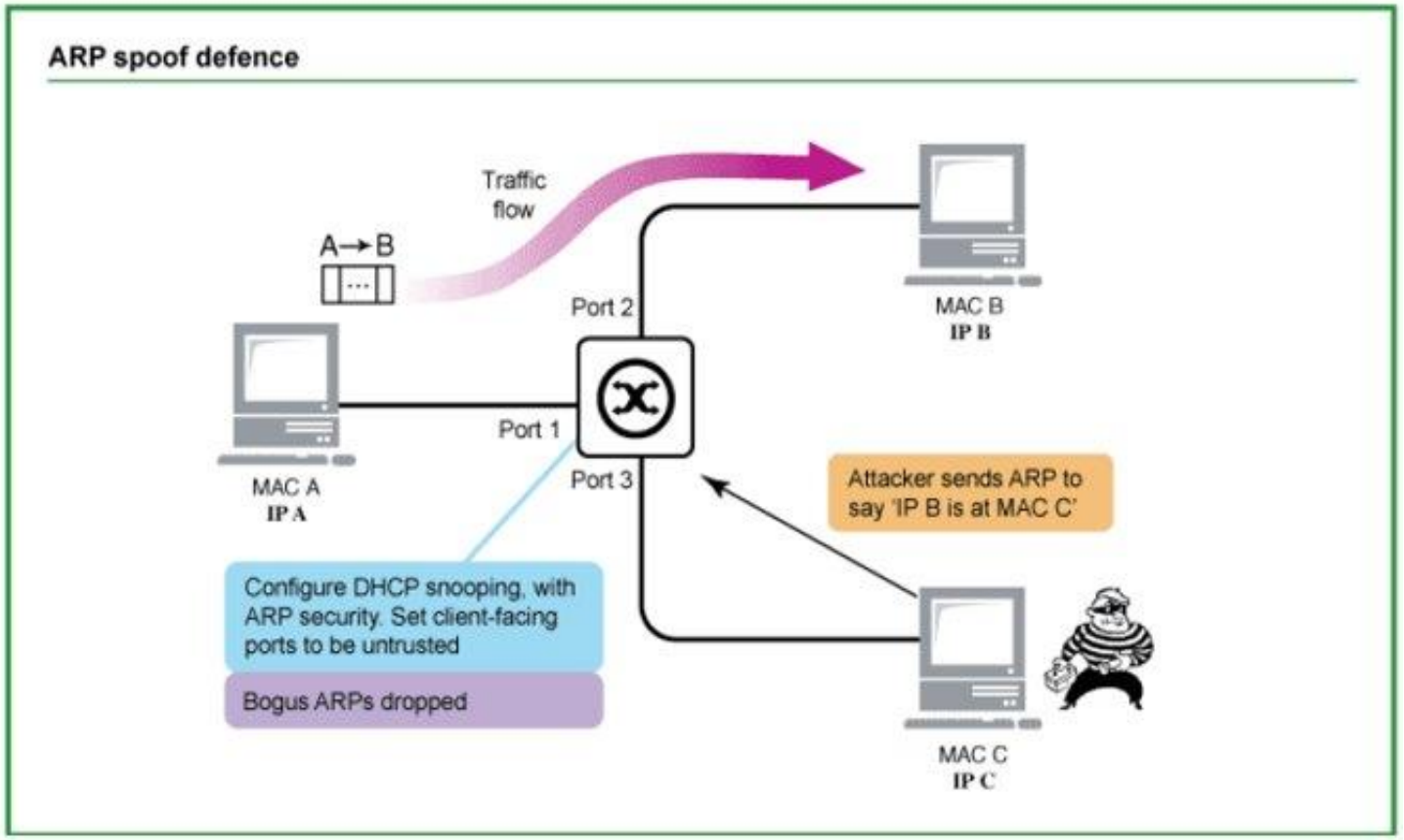
- **Spoofing** means **hide one's true identity** in a network.
- **ARP spoofing** is a type of attack in which a **attacker sends forged ARP** (Address Resolution Protocol) messages to the client inside a LAN.
- A **network attacker** can **respond to ARP request**, posing that it has the requested IPv4 address. Once the attacker's MAC address is mapped to a authentic legitimate IPv4 address, the **attacker will begin receiving any data that is intended for that legitimate IPv4 address.** Now the attacker can launch a man-in-the-middle attack can start capturing the network traffic for any sensitive user data.
- ARP spoofing constructs a large number of forged ARP request and reply packets to overload the switch.
- It opens the door for
  - **Denial-of-service attacks**
  - **Session hijacking**
  - **Man-in-the-middle** attacks

ARP Spoofing



ARP Spoofing





ARP SPOOFING COUNTERMEASURES

- **Get a Detection Tool**
- **Set Up Packet Filtering**
- **Static ARP entries**: these can be defined in the local ARP cache and the switch configured to ignore all auto ARP reply packets.

IP spoofing

- Attackers use this method when they wish to send packets with malicious content to a target machine and **do not want to get identified.**
- Attackers gain unauthorized access to machines by sending messages that appears to be from trusted source.
- Victim is unaware that the packet is not from a trusted host, and hence it accepts the packet and sends a response back to the source computer.
- **Challenge**- attacker must guess proper sequence no to send final ACK packet. Also attacker must use variety of techniques to find address of trusted source and modify packet headers to make it looking like coming from trusted source.
- If successful, attacker may have a connection to the victim's machine as long as that machine is active.
- **Countermeasures**
  - **N/W ingress filtering :** Packet filtering technique that tries to prevent the source address spoofing of the Internet traffic and help prevent IP spoofing.

DoS & DDoS

- **Denial-of-service attack** (**DoS attack**) or **Distributed denial-of-service attack** (**DDoS attack**) is an attempt to make a machine or network resource unavailable to its intended users by temporarily or indefinitely disrupting services of a host .
- In a DoS attack, the attacker usually sends excessive messages asking the network or server to authenticate requests that have invalid return addresses. The network or server will not be able to find the return address of the attacker when sending the authentication approval, causing the server to wait before closing the connection. When the server closes the connection, the attacker sends more authentication messages with invalid return addresses. Hence, the process of authentication and server wait will begin again, keeping the network or server busy.
- Effects
  - Resource Exhaustion
  - Altering or destructing configuration Information

Causes & Types of DoS attack

- Network architecture weakness such as bandwidth limitation
- Server Architecture weakness
- OS or software vulnerabilities
- Flaws in System Security
- The basic types of DoS attack include:
  - Flooding the network to prevent legitimate network traffic
  - Disrupting the connections between two machines, thus preventing access to a service
  - Preventing a particular individual from accessing a service.
  - Disrupting a service to a specific system or individual
  - Disrupting the state of information, such resetting of TCP sessions

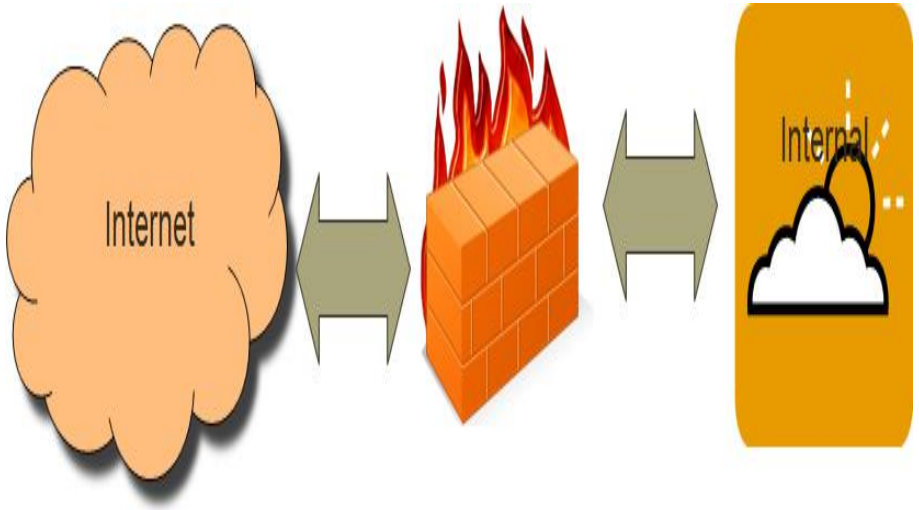
Firewall

Firewall

- A **firewall** is a network security device that **monitors incoming and outgoing network traffic** and **decides whether to allow or block specific traffic based on a defined set of security rules**.
- Firewalls installed specifically to protect information systems can provide another layer of protection and dedicated rules. It is recommended that, if possible, two different hardware vendors are used to provide security against vulnerabilities in the firewall code.
- Firewall match the network traffic against the rule set defined in its table. Once the rule is matched, associate action is applied to the network traffic. Based on a defined set of security rules it accepts, rejects or drops that specific traffic.
  - **Accept :** allow the traffic
  - **Reject :** block the traffic but reply with an “unreachable error”
  - **Drop :** block the traffic with no reply
- Firewall rules list:
  - **Traffic source**
  - **Traffic destination**
  - **Service**
  - **Whether the traffic is allowed or denied**

Firewall as Network Access Control

- Firewall is an interface between networks ie external (internet) and internal and allows traffic flow in both directions
- Firewall provides following Access Control
  - Authentication
  - Authorization
  - Single Sign On

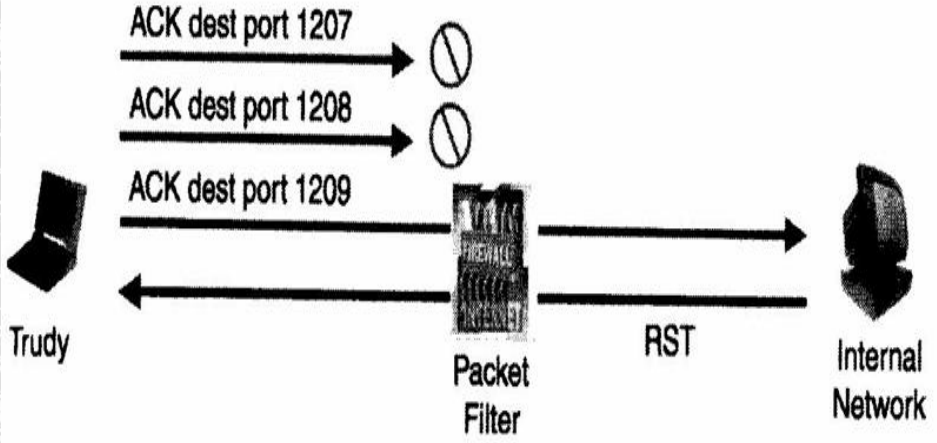


Firewall Types

1. **Packet Filter**
   - Operates at network Layer.
   - Filters packets based on information available at network layer like source IP address, destination IP address, source port, destination port and TCP flag bits(SYN, ACK,RST etc)
   - Different filtering rules for incoming(ingress) and outgoing packets(outgress).
   - Configured using ACL
   - **Advanteages:**
     - Packet filtering generally is inexpensive to implement.
     - It is very efficient since it does NOT read the packet payload and process only header information
   - **Disadvantages:**
     - No maintenance of state. Trudy can use it and identify open ports which are free(Eg: **TCP "ACK" Scan**)
     - Vulnerable to IP spoofin

TCP ACK SCAN ATTACK

- Attacker sends packet with ACK set (without prior handshake) using port p. It's a violation of TCP/IP protocol
- Packet filter firewall passes packet. Firewall considers it part of an ongoing connection
- Receiver sends RST that indicates to the sender that the connection should be terminated. Receiving RST indicates that port p is open!!
- RST confirms that port 1209 is open
- Problem: packet filtering is stateless; the firewall should track the entire connection exchange



Firewall Types

1. Stateful packet filter

- Operates at **transport Layer**
- In addition to **packet inspection** it **validate attributes of multi-packet flows** ie state.
- It can **keep track of connection state** (e.g. TCP streams, active connections, etc…) This means it can keep track of TCP connections and remember UDP connections.
- **Advanteages:**
  - In addition to features of packet filter it keeps track of ongoing connection ie it maintains state and prevents attacks like TCP ACK SCAN.
- **Disadvantages:**
  - It cannot examine application data
  - Slower than packet filtering since more processing is required.

Firewall Types

2. Application Proxy:

- Operates at **Application Layer** and **function as proxy**.
- Allows data into/out of a process based on wat process' type.

e.g. allowing only HTTP traffic to a website

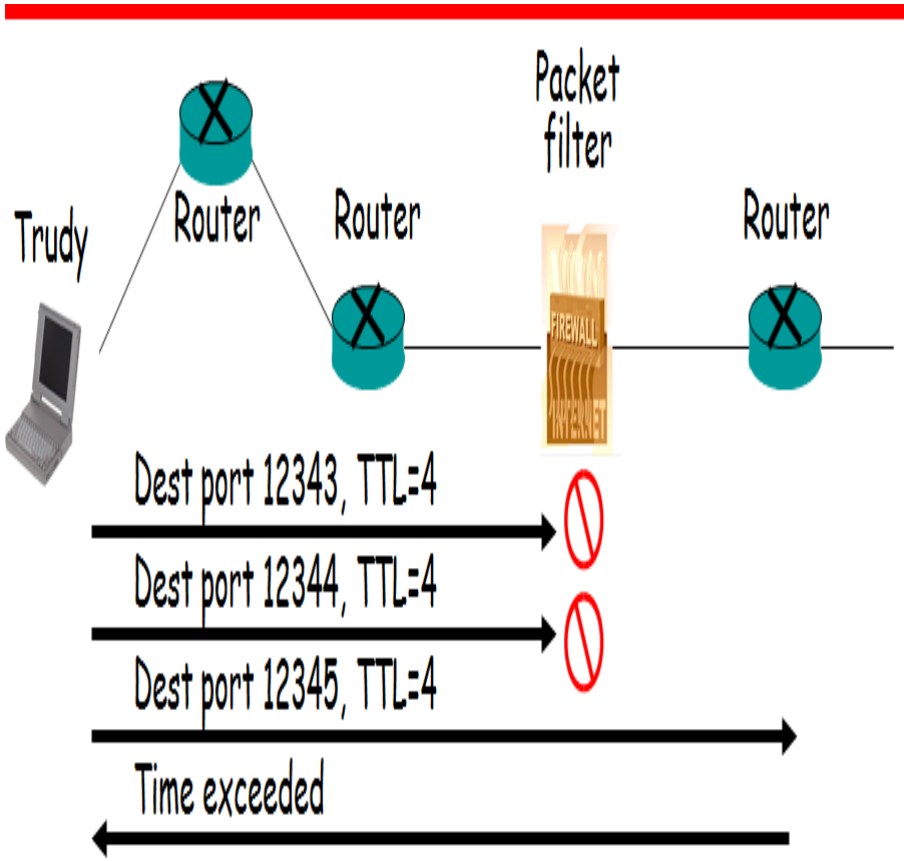
- It c**ompletely destroys incoming packet** and a **new packet** is created.

What is a Proxy server?

- A proxy ia an intermediate connection between servers on internet and internal servers.
- For incoming data, Proxy is server to internal network clients and for outgoing data Proxy is client sending out data to the internet
- **Advanteages:**
  - It has a complete view of connections and application data. So able to filter bad data at application layer and transport layer.
- **Disadvantages:**
  - Speed of filtering is les3. **Personal Firewall:** To protect single host or home network. Uses any of the above 3 methods

How application proxy prevents portscanning attack using newly creted packets?

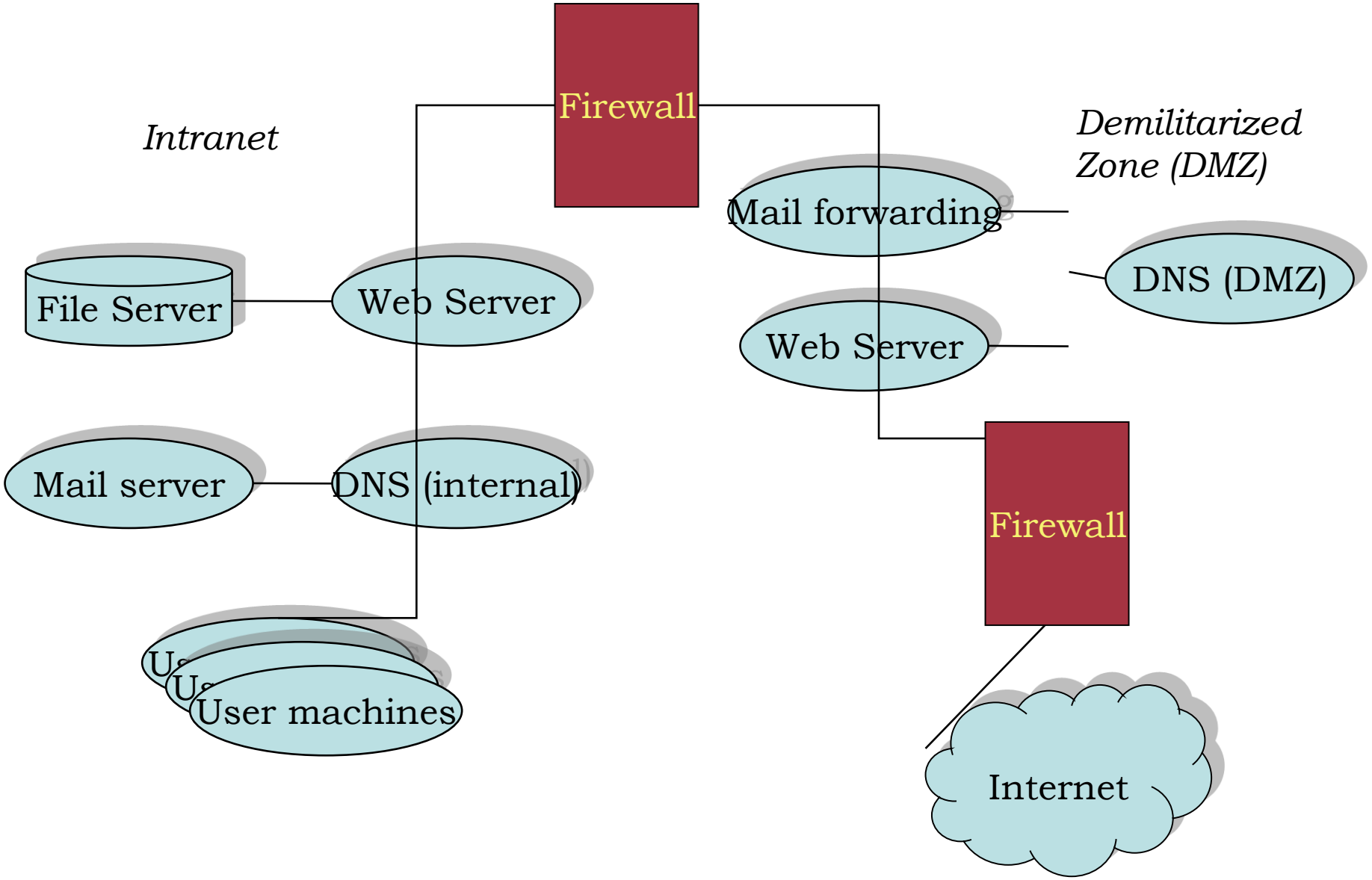
- Firewalk – A Port Scanning tool.
- Scan ports through firewalls
- It requires knowledge of
  - IP address of firewall
  - IP address of one system in internal network
  - Number of hops to the firewall
- Set TTL (time to live) = Hops to firewall +1
- Set destination port to be p
- If firewall does not pass data for port p, then no response
- If data passes thru firewall on port p, then time exceeded error message
- Attack stopped by proxy firewall
  - Incoming packet destroyed (old TTL value also destroyed)
  - New outgoing packet will not exceed TTL.



Firewall Types

- Firewalls can be **host-based** (**running on the host they are protecting and identifying intrusions by analyzing system calls, application logs, file-system modifications (binaries, password files, capability databases, Access control lists, etc.) and other host activities and state**.) or **network-based** (**protecting the local network from external untrusted traffic and identify intrusions by examining network traffic and monitors multiple hosts**.). Host-based firewalls are usually **software implementation**, while network-based ones tend to be **hardware (or a hybrid) implementation**. Network-based girewall gain access to network traffic by connecting to a network devices for port mirroring, or a network tap.
- There are two types of firewalls: ***Stateful*** *or* ***Stateless(packet filtering firewall)***.
- **Stateful firewalls** keep track of the *state* or *type of connection* that is made and can remember specific traits of that connection.
- **Stateless firewalls** block or allow Internet traffic to a server based on a firewall ruleset or, the origin and destination web addresses requested by the server. Stateless firewalls **do not** inspect the packets of information sent to or from the server

"Typical" corporate network



Demilitarised Zone

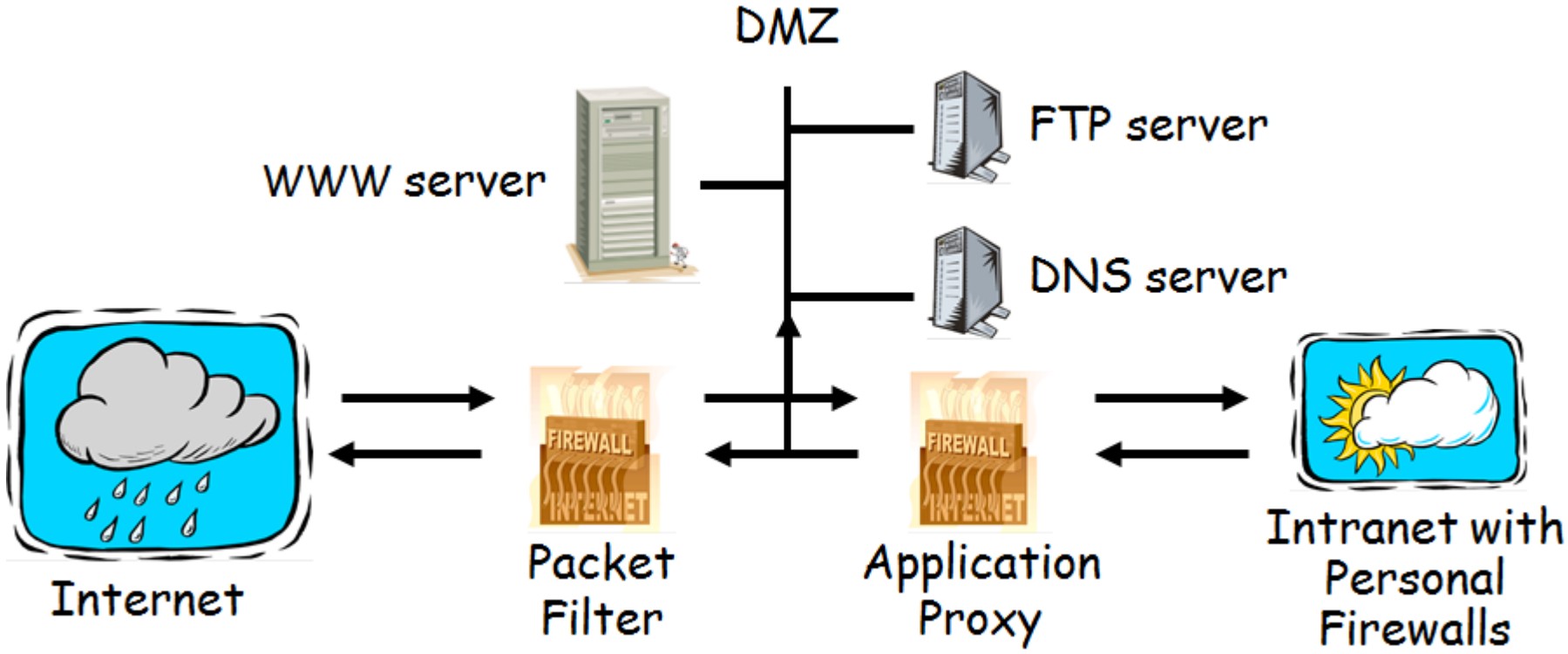
- A DMZ describes a network in which the host servers are located. Limited connections from the Internet are allowed into the DMZ to provide services like web (HTTP) access and e-mail (SMTP et al). Connections from the DMZ to the internal network are not usually allowed by default, which protects the computers inside from compromised hosts in the DMZ.
- Hosts in the DMZ are frequently additionally protected using NAT or PAT to further obfuscate the networking configuration.
- A DMZ is often implemented using a third physical interface on the firewall, but an alternative is to use two firewalls in series with the DMZ. This provides an additional level of protection for the internal network.
- Ideally:

⑩ Internal hosts can access DMZ and Internet

⑩ External hosts can access DMZ only, not Intranet

⑩ DMZ hosts can access Internet only

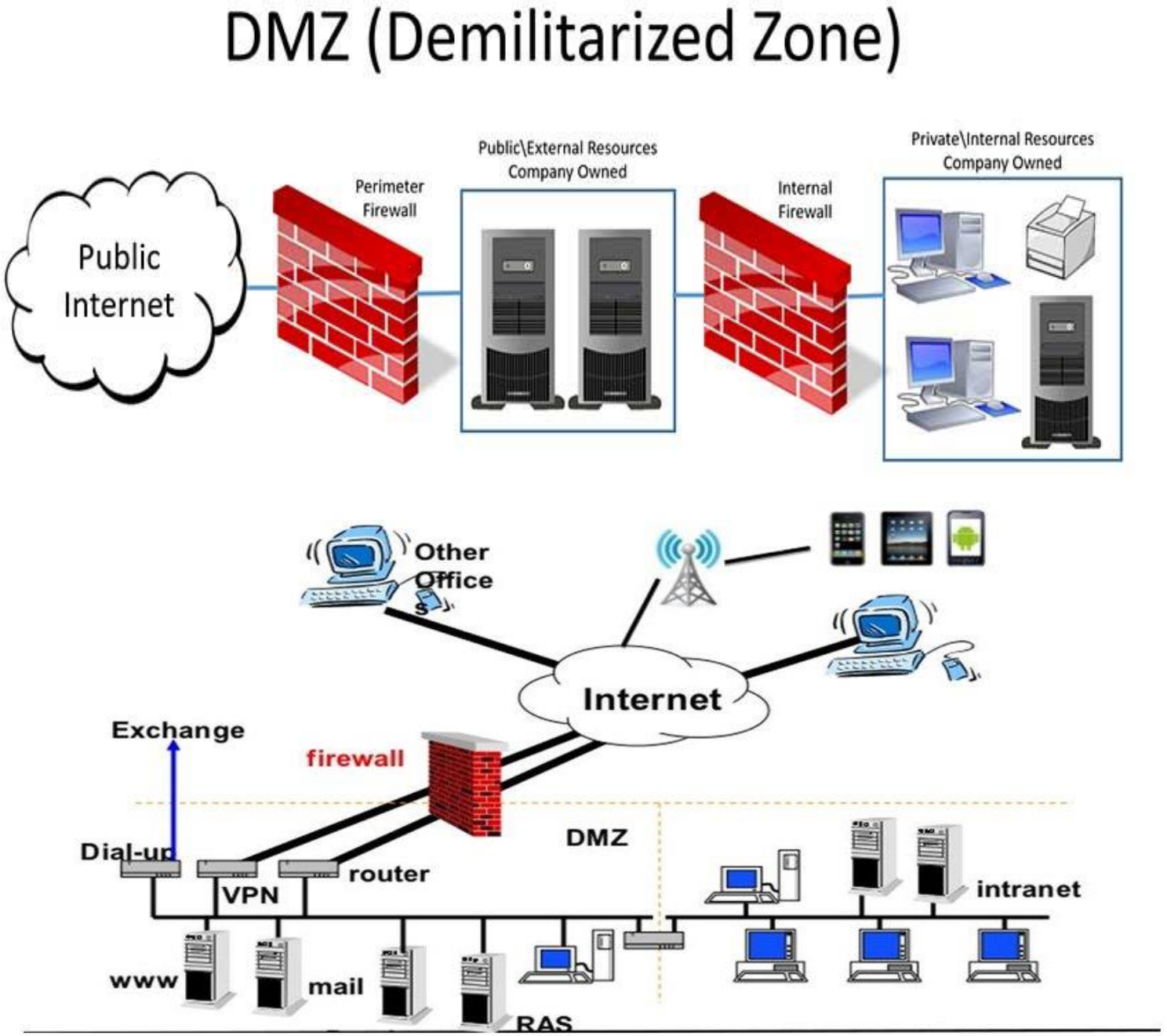
Firewalls and Defense in Depth



- The systems in DMZ are those that must be exposed to the outside world. So this systems should be maintained carefully by administrator.
- Amount of traffic into internal network is relatively small, hence application proxy firewall is employed
- Here there are 3 different layers of security added with different firewalls.

Firewall rules in the demilitarized zone

- **DMZ (Demilitarized Zone)** is an independent network that acts as a buffer zone **between an external network and the internal network.**
- The buffer network contains, for example, **web** servers or mail servers, the communication of which is monitored by **firewalls**.
- Users from the **Internet** allow only to access servers in the demilitarized zone and not to resources of the internal network.
- As a rule, users from the internal system do not communicate directly with support from the Internet.
- For example, they access external resources via a **proxy server**, which, as a **proxy**, handles the Internet communication for them.
- The firewall rejects packets from the DMZ for which there are no corresponding input packets in the direction of the Internet and the internal network.

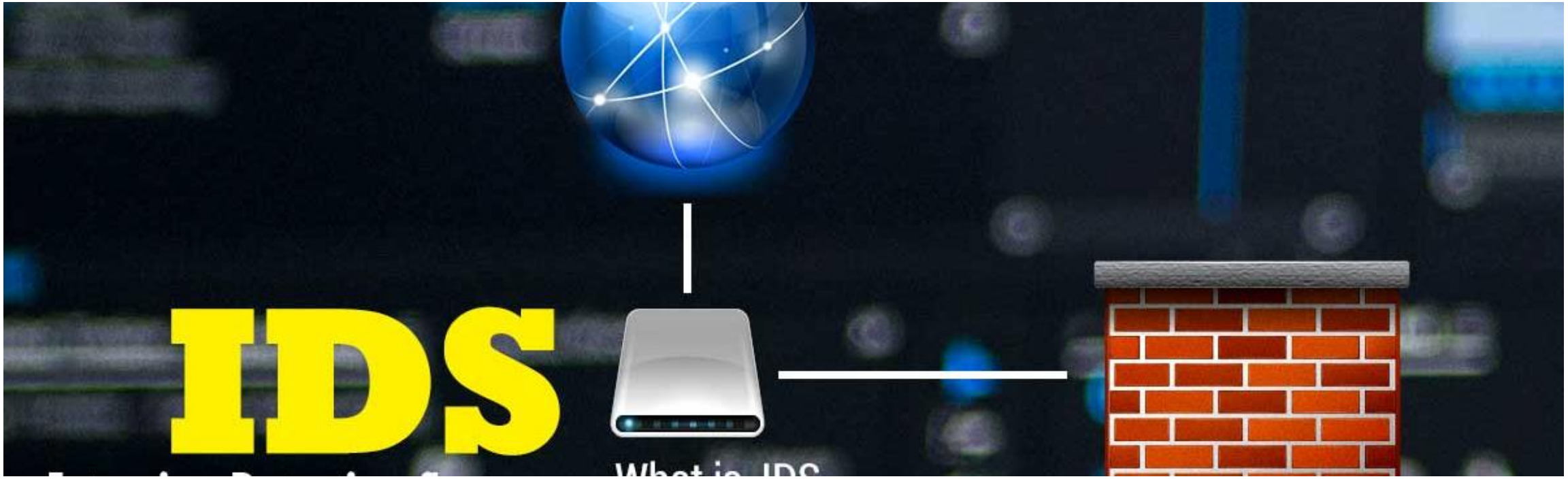


How DMZ can be implemented?

- A demilitarized zone can implement with one or two firewalls. Ideally, these are firewalls from different manufacturers.
- If two firewalls used, there is one between the DMZ and the internal network (inner firewall) and between the DMZ and the external network (outer firewall).
- It prevents security gaps from allowing both firewalls to overcome at the same time.
- A more cost-effective solution is to implement the demilitarized zone with only one firewall.

Firewall Disadvantages

- Firewalls allow traffic only to legitimate hosts and services
- Traffic to the legitimate hosts/services can have attacks
- Solution?
  - Intrusion Detection Systems
  - Monitor data and behavior
  - Report when identify attacks



IDS

Intrusion Detection System

- An **intrusion detection system** (**IDS**) is a device or software application that monitors a network or systems for malicious activity or policy violations. It looks for unusual activity or monitor for “suspicious activity” on a network.
- IDS can protect against known software exploits, like buffer overflows.
- Example: Open Source IDS: Snort, www.snort.org
- IDS uses “intrusion signatures” ie, well known patterns of behavior like Ping sweeps, port scanning, web server indexing, OS fingerprinting, DoS attempts, etc.
- However, IDS is only useful if contingency plans are in place to curb attacks as they are occurring

Methods of Intrusion Detection

1. Signature-based intrusion detection systems/ Misuse Detection monitor all the packets traversing the network and compares them against a database of signatures or patterns of attributes of known malicious threats, much like antivirus software.

- Must tune the IDS to match the characteristics of your network. E.g., what might be unusual in a network of Unix systems might be normal in a network of Windows Systems (or visa versa)
- **Example:** Signature for port sweep
  - A set of TCP packets attempting to connect to a sequence of ports on the same device in a fixed amount of time

ADVANTAGES:

- Widely available
- Fairly fast
- Easy to implement
- Easy to update

DISADVANTAGES:

- Cannot detect attacks for which it has no signature
- The static signature mechanism has obvious problems in that a dedicated attacker can adjust his behaviour to avoid matching the signature.
- The volume of signatures can result in many false positives

Methods of Intrusion Detection

2. Anomaly-based intrusion detection systems/Statistical Detection monitor network traffic and compare it against an established baseline, to determine what is considered normal for the network with respect to bandwidth, protocols, ports and other devices. This type of IDS alerts administrators to potentially malicious activity.

- using statistics will result in a more adaptable and self-tuning system, hence uses neural networks, data mining and create training data from observing "good" runs. Gradual changes can be easily addressed. Gradually adjust expected changes over time

ADVANTAGES:

- Can detect attempts to exploit new and unforeseen vulnerabilities
- Can recognize authorized usage that falls outside the normal pattern

DISADVANTAGES:

- Generally slower, more resource intensive compared to signature-based IDS
- Greater complexity, difficult to configure
- Higher percentages of false alerts

Architectures of IDS

1. **Network based intrusion detection system (NIDS)** is deployed at a strategic point or points within the network, where it can monitor inbound and outbound traffic to and from all the devices on the network.

- **Advantages**
  - Easy deployment
  - Unobtrusive
  - Difficult to evade if done at low level of network operation
- **Disadvantages**
  - Fail Open
  - Different hosts process packets differently
  - NIDS needs to create traffic seen at the end host
  - Need to have the complete network topology and complete host behavior

Architectures of IDS

1. **Host based intrusion detection systems (HIDS)** run on all computers or devices in the network with direct access to both the internet and the enterprise internal network. HIDS have an advantage over NIDS in that they may be able to detect anomalous network packets that originate from inside the organization or malicious traffic that a NIDS has failed to detect. HIDS may also be able to identify malicious traffic that originates from the host itself, as when the host has been infected with malware and is attempting to spread to other systems.

- **Advantages**
  - More accurate than NIDS
  - **Less volume of traffic so less overhead**
- **Disadvantages**
  - Deployment is expensive
  - What happens when host get compromised?

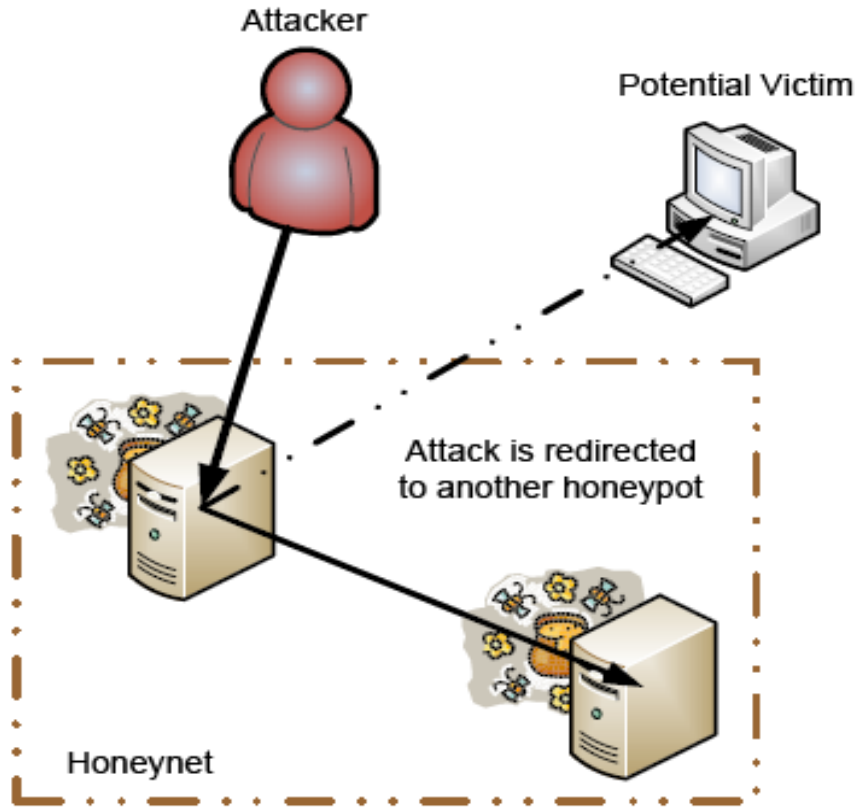
Honeypots

Honeypots

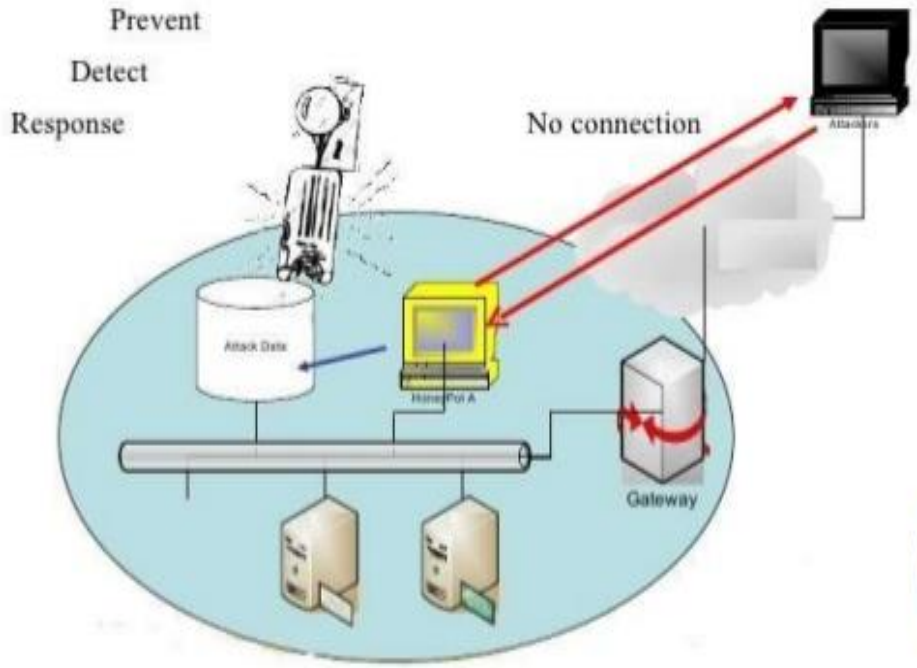
- A honey pot is a trap that attracts potential attackers.
- Honeypots are filled with fabricated information.
- Any access to honeypot triggers monitors, sensors and event loggers with alarms.
- A honeypot isn't set up to address a specific problem, like a firewall or anti-virus. Instead, it's an information tool that can help you understand existing threats to your business and spot the emergence of new threats.

Aim

1. It can be used to detect attacks or Drive attention of a potential attacker and to deflect them from critical systems
2. It can also be used to gain information about how cybercriminals operate and intruders action.
3. Encouraging intruders to stay for some time so administrators can detect and swiftly act against them.



➢HOW HONEYPOT WORKS :



Honeypot Working

- The honeypot looks like a real computer system, with applications and data, fooling cybercriminals into thinking it's a legitimate target.
- Honeypots are made attractive to attackers by building in deliberate security vulnerabilities. For instance, a honeypot might have ports that respond to a port scan or weak passwords. Vulnerable ports might be left open to entice attackers into the honeypot environment, rather than the more secure live network.
- Honeypots can be placed:
  - In front of the firewall(Internet)
  - Demilitarized Zone
  - Behind the firewall(Intranet)

Honeypot Types

- **Honeypots** can be classified based on their deployment (use/action) and based on their level of involvement. Based on deployment, honeypots may be classified as
  - **production honeypots:** are low interaction honeypots which are easy to use, capture only limited information, and are used primarily by corporations.
  - **research honeypots:** are complicated honeypots that are run to gather information about the motives and tactics of the black hat community targeting different networks. These honeypots do not add direct value to a specific organization; instead, they are used to research the threats that organizations face and to learn how to better protect against those threats.

1. A **decoy database** can be set up to monitor software vulnerabilities and spot attacks exploiting insecure system architecture or using SQL injection, SQL services exploitation, or privilege abuse.
2. A **malware honeypot** mimics software apps and APIs to invite malware attacks. The characteristics of the malware can then be analyzed to develop anti-malware software or to close vulnerabilities in the API.
3. A **spider honeypot** is intended to trap webcrawlers ('spiders') by creating web pages and links only accessible to crawlers
4. **Email traps** or spam traps place a fake email address in a hidden location where only an automated address harvester will be able to find it.

Web security basics

- A **web server** is a computer host configured and connected to Internet for serving web pages on request and is publicly accessible over Internet. It rns on different platforms like UNIX,Windows and Macintosh.
- **Web Server Vulnerability:** Weakness in custom Web Application services, architecture, design, configuration, langages, servers, interpreters or code can be exploited by hackers to compromise web servers.

OWASP TOP 10

- The Open Web Application Security Project (OWASP) is a worldwide not-for-profit charitable organization focused on improving the security of software
- OWASP Top 10 is the list of the 10 most common application vulnerabilities. It also shows their risks, impacts, and countermeasures. Updated every three to four years, the latest OWASP vulnerabilities list was released in 2018.

OWASP top 10 vulnerabilities

- **Injection**. Injection flaws, such as SQL, NoSQL, OS, and LDAP injection, occur when untrusted data is sent to an interpreter as part of a command or query. The attacker's hostile data can trick the interpreter into executing unintended commands or accessing data without proper authorization.
- **Broken Authentication**. Application functions related to authentication and session management are often implemented incorrectly, allowing attackers to compromise passwords, keys, or session tokens, or to exploit other implementation flaws to assume other users' identities temporarily or permanently.
- **Sensitive Data Exposure**. Many web applications and APIs do not properly protect sensitive data, such as financial, healthcare, and PII. Attackers may steal or modify such weakly protected data to conduct credit card fraud, identity theft, or other crimes. Sensitive data may be compromised without extra protection, such as encryption at rest or in transit, and requires special precautions when exchanged with the browser.
- **XML External Entities (XXE)**. Many older or poorly configured XML processors evaluate external entity references within XML documents. External entities can be used to disclose internal files using the file URI handler, internal file shares, internal port scanning, remote code execution, and denial of service attacks.
- **Broken Access Control**. Restrictions on what authenticated users are allowed to do are often not properly enforced. Attackers can exploit these flaws to access unauthorized functionality and/or data, such as access other users' accounts, view sensitive files, modify other users' data, change access rights, etc.

OWASP top 10 vulnerabilities

- **Security Misconfiguration**. Security misconfiguration is the most commonly seen issue. This is commonly a result of insecure default configurations, incomplete or ad hoc configurations, open cloud storage, misconfigured HTTP headers, and verbose error messages containing sensitive information. Not only must all operating systems, frameworks, libraries, and applications be securely configured, but they must be patched/upgraded in a timely fashion.
- **Cross-Site Scripting XSS**. XSS flaws occur whenever an application includes untrusted data in a new web page without proper validation or escaping, or updates an existing web page with user-supplied data using a browser API that can create HTML or JavaScript. XSS allows attackers to execute scripts in the victim's browser which can hijack user sessions, deface web sites, or redirect the user to malicious sites.
- **Insecure Deserialization**. Insecure deserialization often leads to remote code execution. Even if deserialization flaws do not result in remote code execution, they can be used to perform attacks, including replay attacks, injection attacks, and privilege escalation attacks.
- **Using Components with Known Vulnerabilities**. Components, such as libraries, frameworks, and other software modules, run with the same privileges as the application. If a vulnerable component is exploited, such an attack can facilitate serious data loss or server takeover. Applications and APIs using components with known vulnerabilities may undermine application defenses and enable various attacks and impacts.
- **Insufficient Logging & Monitoring**. Insufficient logging and monitoring, coupled with missing or ineffective integration with incident response, allows attackers to further attack systems, maintain persistence, pivot to more systems, and tamper, extract, or destroy data. Most breach studies show time to detect a breach is over 200 days, typically detected by external parties rather than internal processes or monitoring.

SQL INJECTION

- User-controlled input that enables the attacker to interact with the application's back-end database (DB) in non-intended ways.
- This could lead to user account compromise, extraction of sensitive data or denial of service.
- Common causes:
  - Lack of validation and sanitization
  - No prepared statements (bind queries) used
  - Principle of least privilege not applied
- Attacker submits HTTP request with a malicious parameter value that modifies an existing SQL query, or adds new queries

HTTP Request
POST /login?u=foo&p=bar

SQL Query
SELECT user, pwdFROM users WHERE u = 'foo'

SQL INJECTION

HTTP Request

POST /login?u=**‘+OR+1<2#**&p=bar

SQL Query

SELECT user, pwd FROM users WHERE u = ‘’ OR 1<2#

Attacker submits HTTP request with a malicious parameter value that modifies an existing SQL query, or adds new queries

- Error messages
  You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near """" at line 1 SELECT * FROM authors WHERE name = """"
- Special queries
  - " union select null,null,null,null,null -- ” gives SQL error message
  - " union select null,null,null,null,null,null – ” gives invalid credential message

EXAMPLES

- $username = $_POST['username']; $query = “ SELECT * FROM Users WHERE username = ‘$username’;“;
- $username=“Bob’ AND DoB=‘11111918”;$query=“ SELECT * FROM Users WHERE username=‘Bob’ AND DoB=‘11111918’;”;

SQL INJECTION EXAMPLES

SQL Injection –Authentication Bypass

$username=$_POST[‘username’];

$password=$_POST[‘password’];

$query=“SELECT * FROM users WHERE username=‘$username’
AND password=‘$password’;”;

$username=“Bob’ OR username=‘Alice’--”

$query=“SELECT * FROM users WHERE username=‘Bob’ OR username=‘Alice’--’;”

SQL Injection –Data Theft

$company=$_POST[‘company’];

$query=“SELECT name,lastname,DoBFROM users WHERE company=‘$company’;”;

company=Accenture’ UNION SELECT null,null,passwordFROM users WHERE
‘1’=‘1”

$query=“ SELECT name,lastname,DoBFROM users WHERE company=‘Accenture’
UNION SELECT null,null,passwordFROM users WHERE ‘1’=‘1’;”;

Secure web programming

- Secure Web development is an iterative process that comprises application design, implementation, vulnerability testing, and monitoring.

1. Maintain security during web app development
2. Be paranoid: require injection & input validation (user input is not your friend)
   - data type validation (ensures that parameters are of the correct type: numeric, text, et cetera).
   - data format validation (ensures data meets the proper format guidelines for schemas such as json or xml).
   - data value validation (ensures parameters meet expectations for accepted value ranges or lengths).
3. Encrypt your data
4. Use exception management
5. Apply authentication, role management & access control
6. Avoid security misconfigurations
7. Implement https (and redirect all http traffic to https)
8. Include auditing & logging
9. Use rigorous quality assurance and testing
10. Minimize the Window of Exposure with Virtual Patching
11. Monitor Web Applications for Attacks

Real World Internet Security Protocols

Protocols for Secure Communications

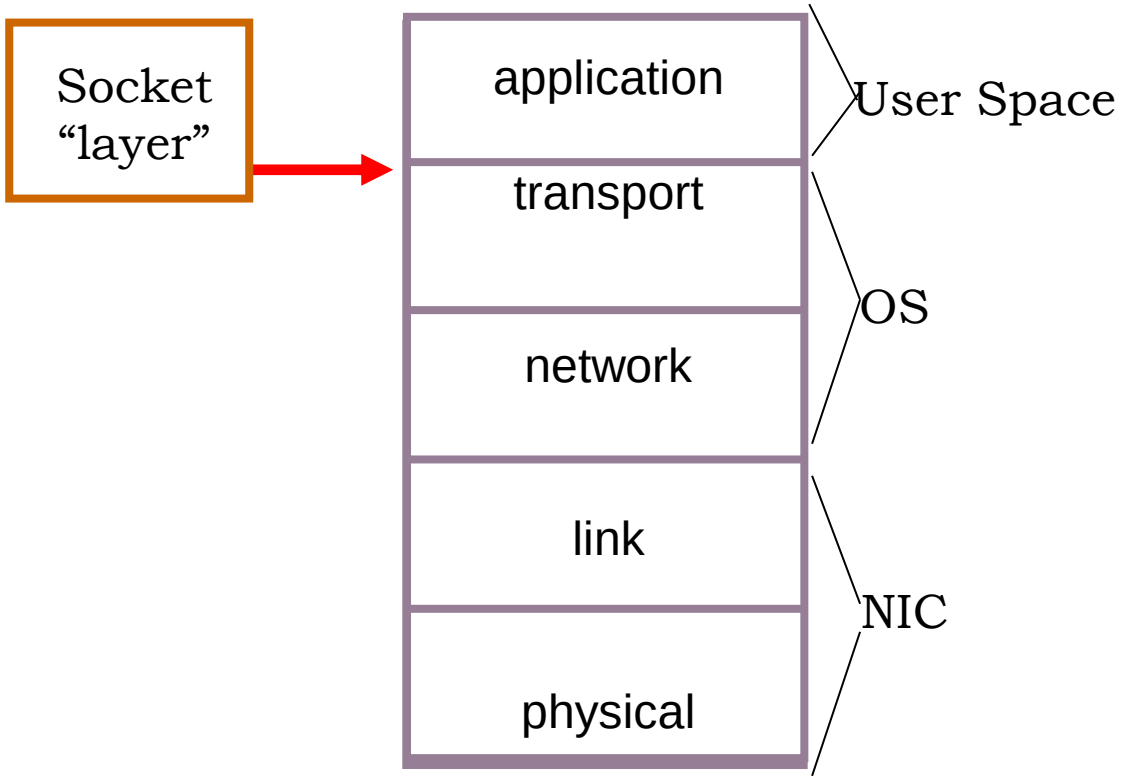
- **Secure Socket Layer (SSL) protocol:** uses public key encryption to secure channel over public Internet
- **Securing TCP/IP with IPSec**
  - **Internet Protocol Security (IPSec):** open source protocol to secure communications across any IP-based network
  - IPSec designed to protect data integrity, user confidentiality, and authenticity at IP packet level
  - IPSec combines several different cryptosystems: Diffie-Hellman; public key cryptography; bulk encryption algorithms; digital certificates
  - In IPSec, IP layer security obtained by use of application header (AH) protocol or encapsulating security payload (ESP) protocol

SSL/TLS

- **Socket layer** lies between **transport layer(tcp)** and **application layer(http)** which deals with web browsing.
- **SSL** was created by **Netscape corporation** in **1994** which **encrypts application layer data.**
- **Secure Socket Layer (SSL)** is a **cryptographic protocol** in transport layer which defines how 2 entities (client and server) communicate with each other securely.(using https)
- SSL provides **authentication** and **confidentiality**.
- SSL has 3 sub protocols: **Handshake protocol, Record Protocol & Alert Protocol.**
- Transport Layer Security (TLS) is the successor of SSL.
  - SSL 2.0
  - SSL 3.0
  - TLS 1.0 (SSL 3.1)
  - TLS 1.1 (SSL 3.2)
  - TLS 1.2 (SSL 3.3)

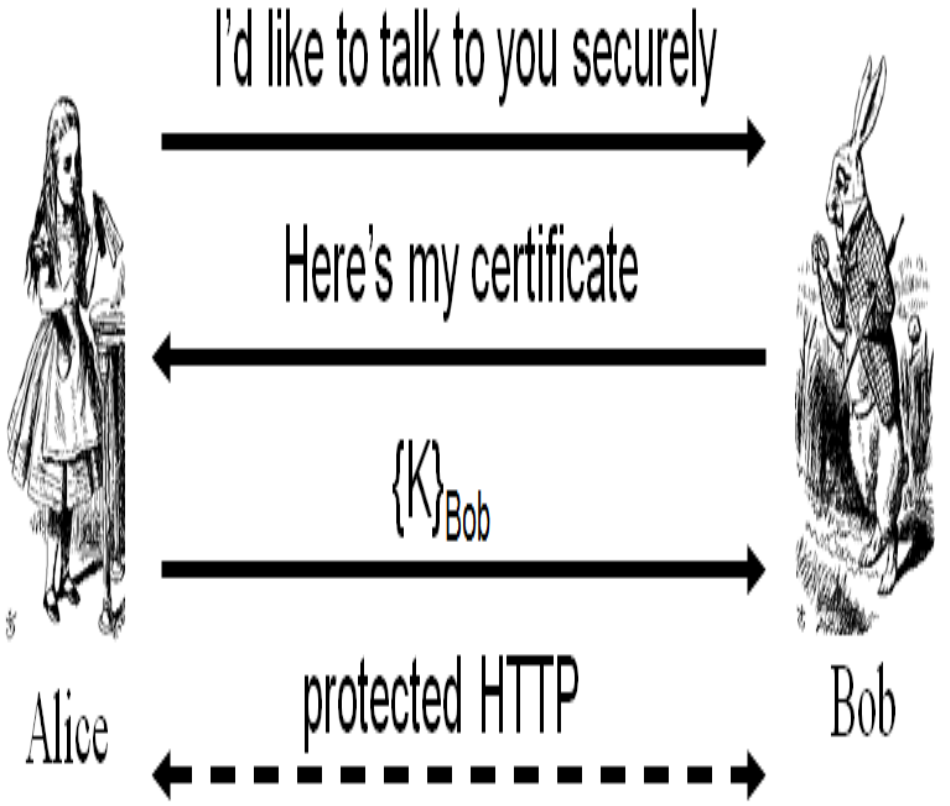
Socket Layer

- "Socket layer" lives between application and transport layers in Internet Protocol Stack.
- SSL deals with web browsing, which uses protocols of application layer which is HTTP and transport layer protocol TCP.
- SSL is used for secure transaction over Internet.
- For example, if you want to buy a book at amazon.com…
  - You want to be sure you are dealing with Amazon (**authentication**)
  - Your credit card information must be protected in transit (**confidentiality** and/or **integrity**)
  - As long as you have money, Amazon does not care who you are
  - So, no need for mutual authentication

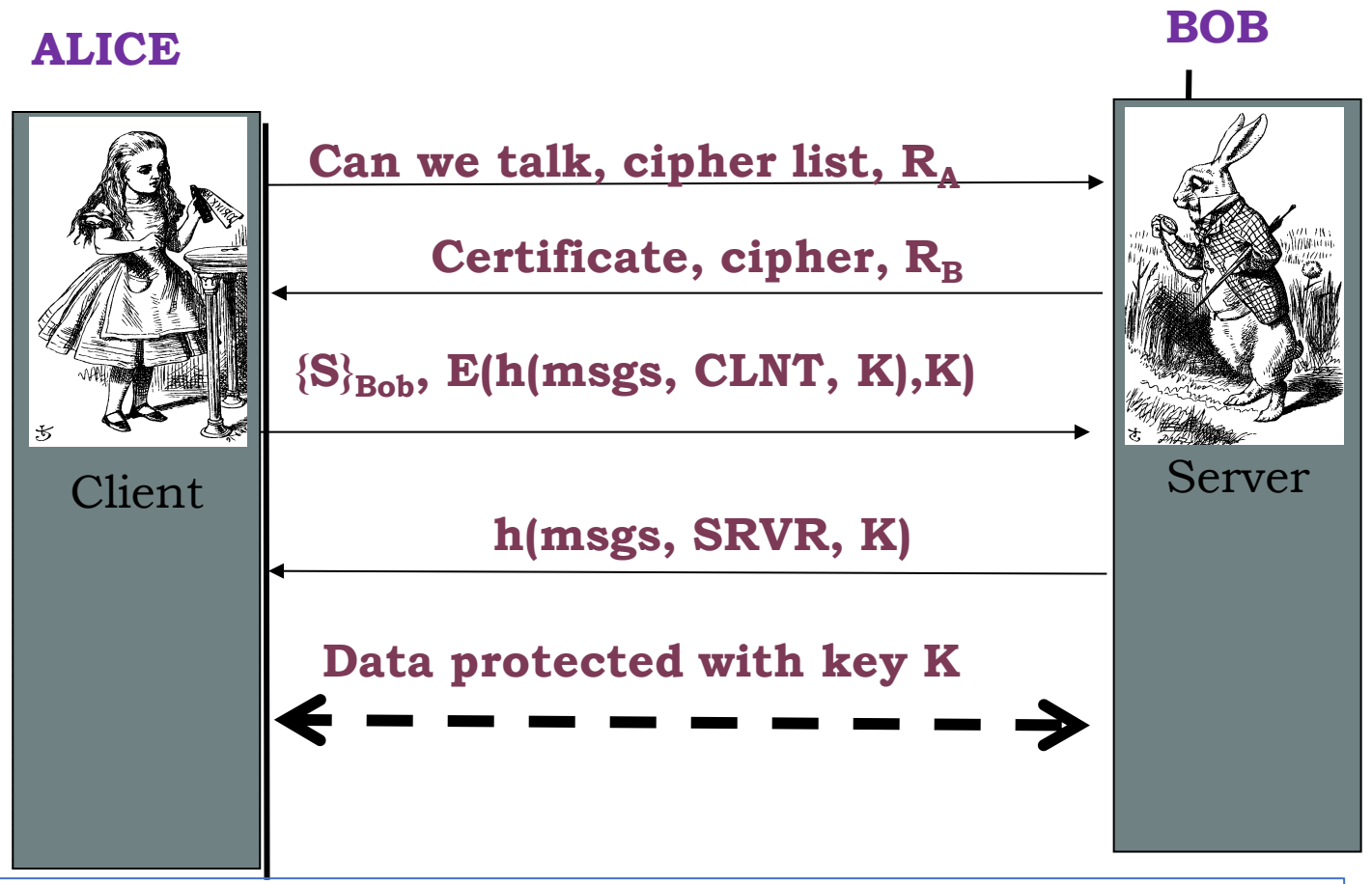


SSL-General Idea

- Encrypting/decrypting with public/private keys are used during the handshake and the session key is used after that.
- 1.**Browser** connects to a web server (website) secured with SSL (https). Browser requests that the server identify itself.
- 2.**Server** sends its SSL Certificate, including the server's public key.
- 3.**Browser** checks the certificate a list of trusted CAs and sends back a symmetric session key using the server's public key.
- 4.**Server** decrypts the symmetric session key using its private key and sends back an acknowledgement encrypted with the session key to start the encrypted session.
- 5.**Server** and **Browser** now encrypt all transmitted data with the session key.



SIMPLIFIED SSL HANDSHAKE



S= Premaster Secret
K = h(S,R_A,R_B)
CLNT - literal string
SRVR-literal String

SSL Keys
- 6 "keys" derived from K = h(S,RA,RB)
  - 2 encryption keys(sending and receiving): client and server
  - 2 integrity keys(sending and receiving): client and server
  - 2 Initialization Vectors(Ivs) (sending and receiving): client and server

Handshake Steps(Simplified)

1. **Client send** hello message including **a random message, its protocol version, session ID, list of cipher suite, a random nonce R_A and compression method**
2. **Server replies** with a hello message with **its own cipher suite selcted from list of cipher suite that Alice has send, random message, its certificate, random nonce R_B and requests for client certificate**
3. **Client** authenticates server and verifies Bob's certificate, then creates **a pre-master secret S** for the session and encrypts it with the **servers public key** $\{S\}_{BOB}$ and also sends a hash **h(msgs,CLNT,K)** that is encrypted with key **K i**e **E(h(msgs, CLNT, K),K)**. These hash contains previous messages and **CLNT** a literal string. Hash is used to verify the previous messages has been received correctly.
4. **Server** authenticates the client, and uses its **private key** to decode the **pre-master secret S** and uses pre-master secret to create a **master secret key** $K = h(S,R_A,R_B)$ for the session and tells the client that it will use the master key for the session. Also it responds with similar hash **h(msgs, SRVR, K)**, so Alice can verify bob by computing Hash herself and assure if messages are received correctly.
5. Client decodes the master key and tells the server that it will use the key to encode the session also.
6. Handshake is done

SSL Authentication & MITM Attack

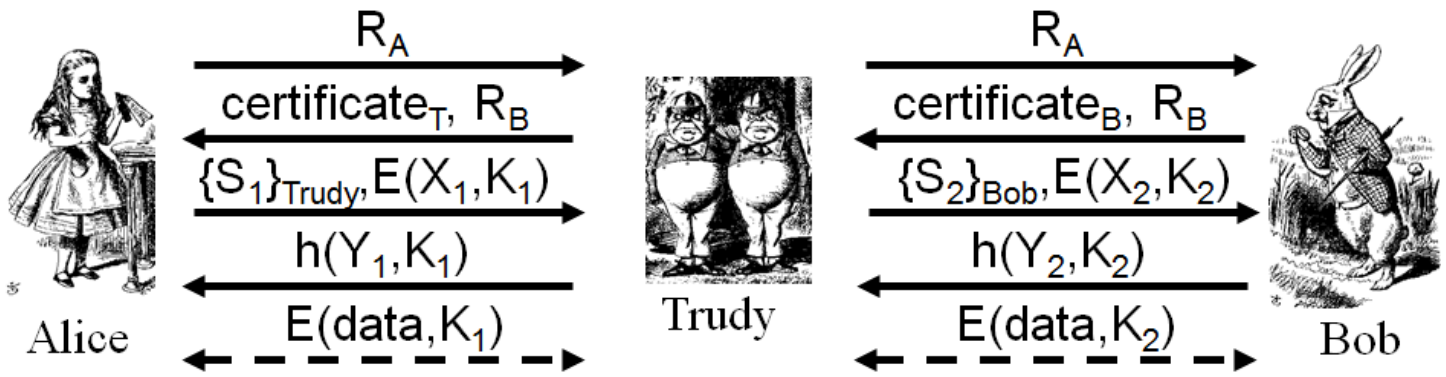
SSL Athentication

- Alice authenticates Bob, not vice-versa
  - How does client authenticate server?
  - Why would server not authenticate client?
- Mutual authentication is possible: Bob sends **certificate request** in message 2
  - Then client must have a valid certificate
  - But, if server wants to authenticate client, server could instead require password

Man in the middle Attack

- Different keys in each direction prevents Replay Attacks.
- SSL is designed to protect against MITM Attack
- What prevents this MiTM “attack”?
  - Bob’s certificate is signed by a certifying authority(like Verisign)
  - **Scenario 1:** Trudy sending Bob’s certificate to Alice. **Defense:** Trudy couldn’t authenticate herself as Bob.
  - **Scenario 2:** Trudy sending her own certificate instead of Bob’s certificate to Alice. **Defense:** Attack fails as Alice attempts to verify the signature. Alice’s web browser provide a warning; but if user ignore warning then Trudy’s MITM attack succeed.

SSL MiM Attack



Q: What prevents this MiM "attack"?

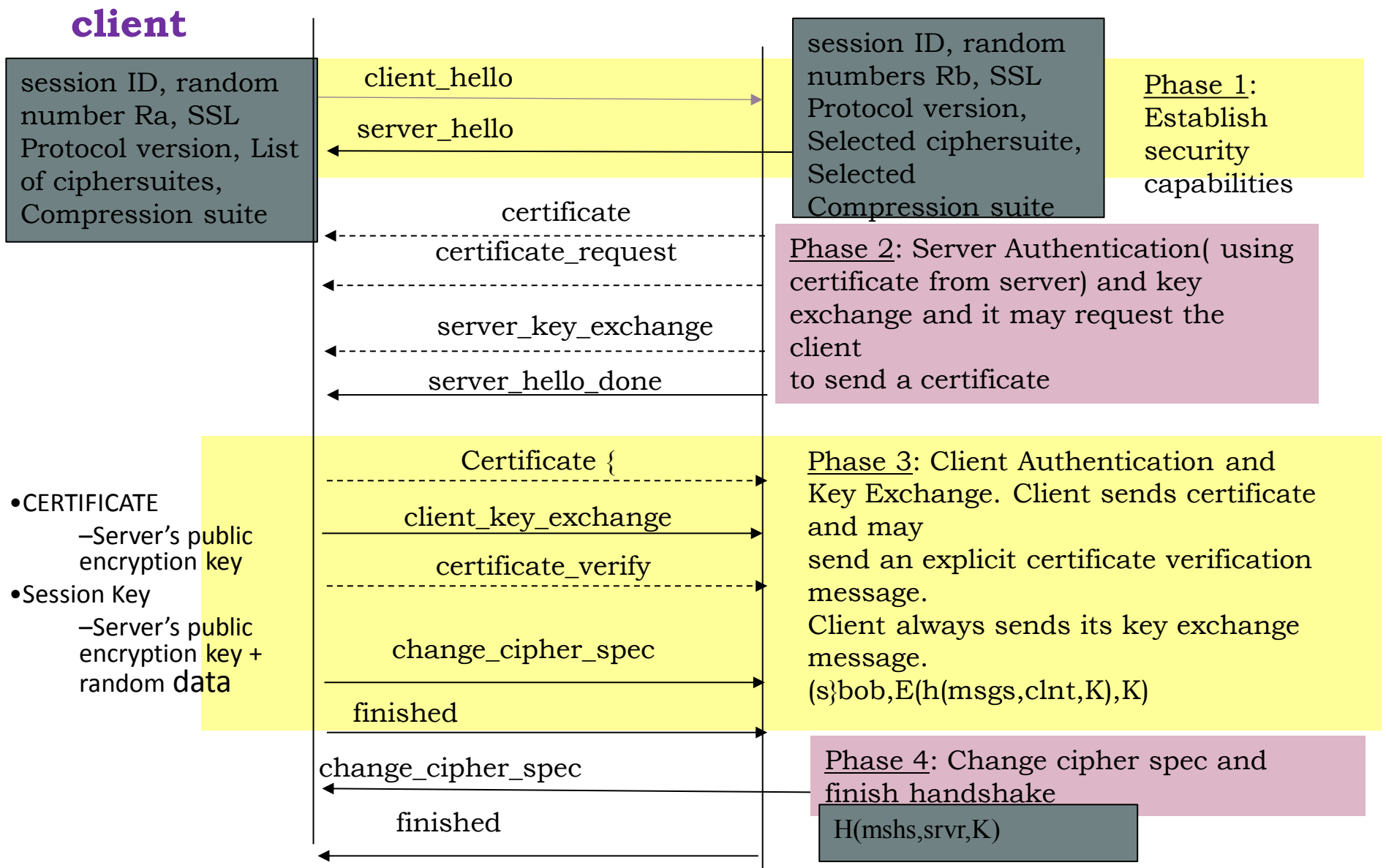
A: Bob's certificate must be signed by a certificate authority (CA)

- What does browser do if signature not valid?
- What does user do when browser complains?

Secure Sockets Layer (SSL) Protocols

- **SSL Handshake Protocol**
  - negotiation of security algorithms and parameters
  - key exchange
  - server authentication and optionally client authentication
- **SSL Record Protocol**
  - fragmentation
  - compression
  - message authentication and integrity protection
  - encryption
- **SSL Alert Protocol**
  - error messages (fatal alerts and warnings)
- Supports many algorithms
  - **Public Key:** RSA, DH, DSA
  - **Symmetric Key:** RC2, RC4, IDEA, DES, 3DES, AES
  - **Hashes:** MD5, SHA
- **Public keys** are signed by **CA (Certificate Authority)** using **X.509 certificates**.

SSL Handshake Protocol – overview



How SSL Works: the Handshake in Detail

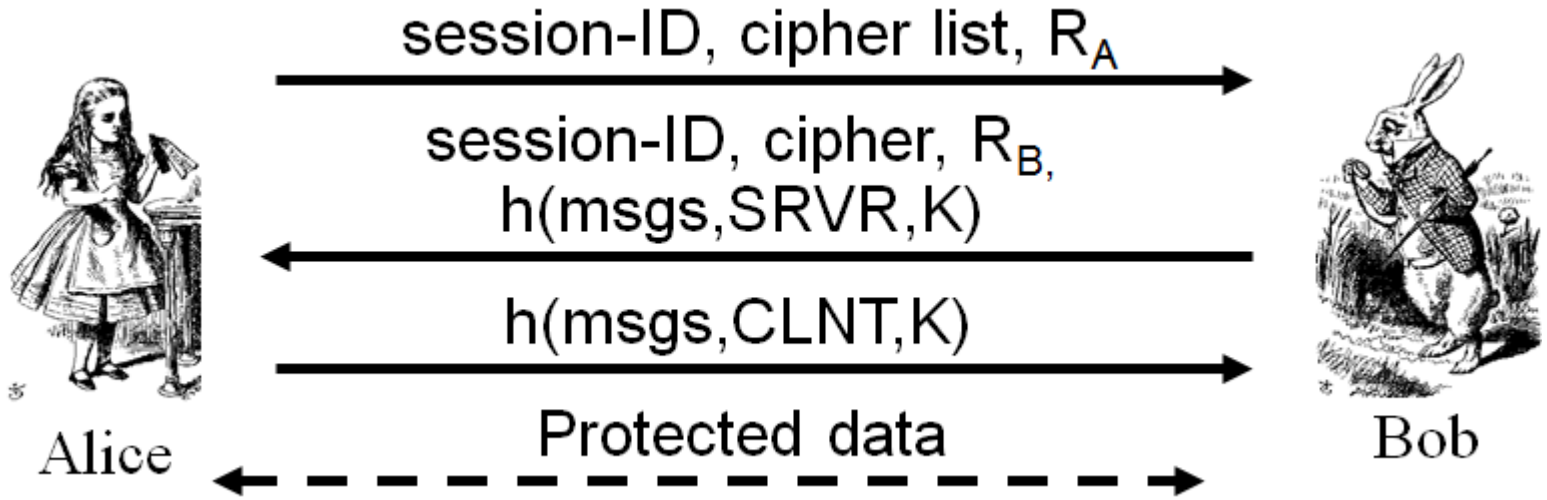
1. **Client hello** - The client sends the server information including the highest version of SSL it supports and a list of the cipher suites it supports.
2. **Server hello** - The server chooses the highest version of SSL and the best cipher suite that both the client and server support and sends this information to the client.
3. **Certificate** - If server authentication is required then the server sends the client a certificate or a certificate chain.
4. **Certificate request** - If the server needs to authenticate the client, it sends the client a certificate request.
5. **Server key exchange** - The server sends the client a server key exchange message when the public key information sent in 3) above is not sufficient for key exchange.
6. **Server hello done** - The server tells the client it is finished with its initial negotiation messages.

How SSL Works: the Handshake in Detail

1. **Certificate** - If the server requests a certificate from the client in Message 4, the client sends its certificate chain, like the server did in Message 3.
2. **Client key exchange** - The client generates information used to create a key to use for symmetric encryption. For RSA, the client then encrypts this key information with the server's public key and sends it to the server.
3. **Certificate verify** – If the server is authenticating the client, the client sends a random number that it digitally signs. When the server decrypts number with the client's public key, the server authenticates the client.
4. **Change cipher spec** - The client tells the server to change to encrypted mode.
5. **Finished** - The client sends the server a hash of the handshake messages.
6. **Change cipher spec** - The server tells the client to change to encrypted mode.
7. **Finished** - The server sends the client a hash of the handshake messages.

- **Encrypted data** - The client and the server communicate using the symmetric encryption algorithm and the cryptographic hash function negotiated in Messages 1 and 2, using the secret key that the client sent to the server in Message 8.

SSL Session/Connections



- An **SSL session** is an association between a client and a server
- Sessions are stateful; the session state includes security algorithms and parameters . Sessionn establishment is expensive, since public key operations are required.
- A session may include multiple secure connections between the same client and server
- Connections of the same session share the session state
- Sessions are used to avoid expensive negotiation of new security parameters for each connection

SSL Session/Connections

- A **session state** includes
  - **session identifier**
    - arbitrary byte sequence chosen by the server to identify the session
  - **certificate**
    - X509 certificate of the peer
    - may be null
  - **compression method**
  - **cipher spec**
    - bulk data encryption algorithm (e.g., null, DES, 3DES, …)
    - MAC algorithm (e.g., MD5, SHA-1)
    - cryptographic attributes (e.g., hash size, IV size, …)
  - **master secret**
    - 48-byte secret shared between the client and the server

IPSEC PROTOCOL

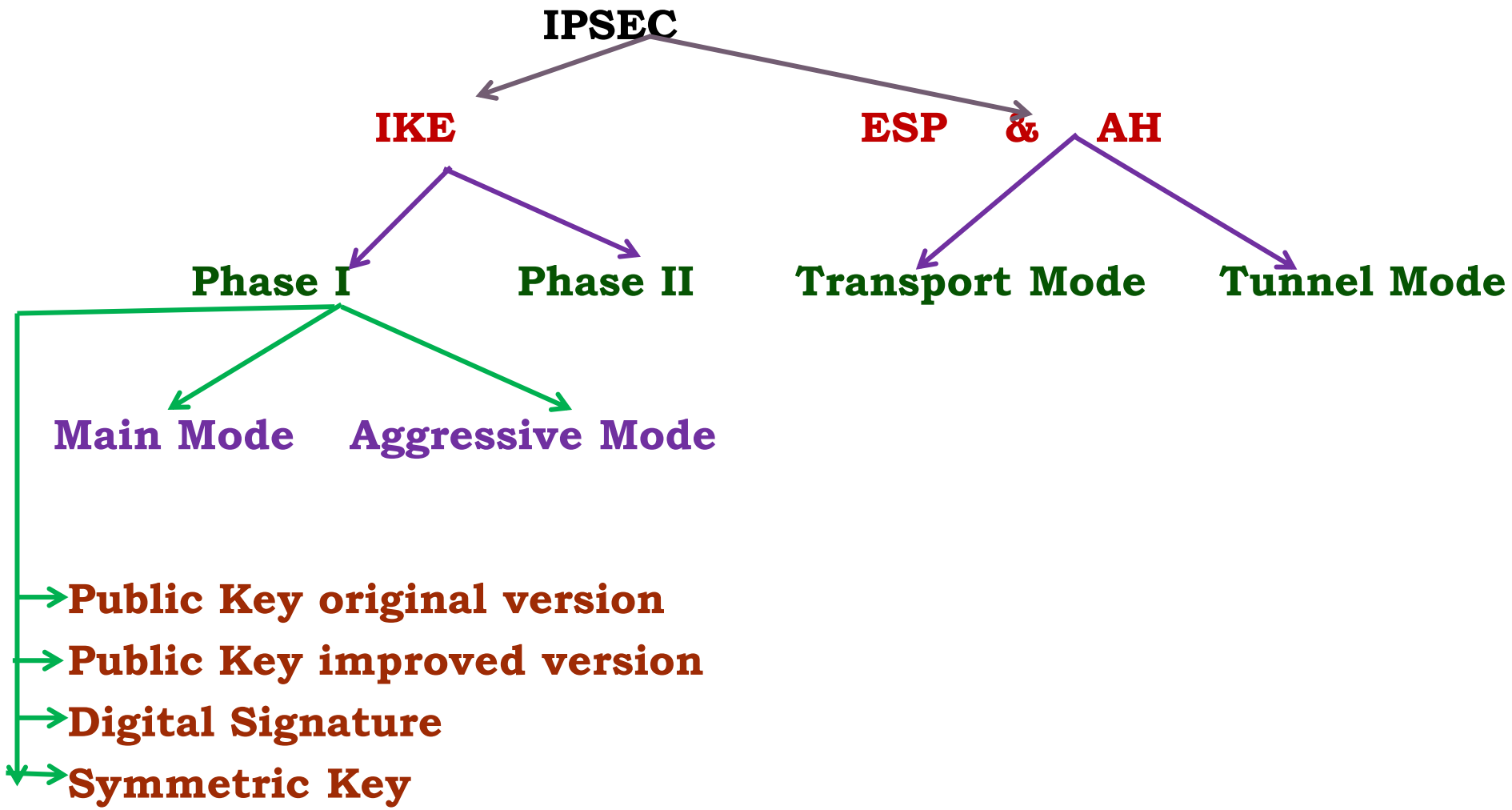
IPSec protocol

- IPSec is a suite of protocols that interact with one another to provide secure private communications across IP networks

> "IPsec is designed to provide interoperable, high quality, cryptographically based security for IPv4 and IPv6" - (RFC 2401)

- An IPsec protected connection is called a **security association,** which may be either end-to-end or link-to-link.
- IPsec is situated on **network layer**(layer-3) below TCP/UDP and is **transparent to applications**
  - IPsec provides security to all traffic passing through the IP layer
- IPSec provides
  - **authentication**
  - **confidentiality**
  - **key management**
  - **integrity**

IPSec Components



IPSec in a Glance

IPsec = AH + ESP + IKE

Protection for IP traffic

Sets up keys and algorithms for AH and ESP

I. **Internet Key Exchange (IKE)Authenticated Keying**

- **Goal:** provides **mutual authentication** and **establishment of a shared symmetric key.** Establish security association for AH and ESP – If IKE is broken, AH and ESP provide no protection. Sets up keys and algorithms for ESP and AH. IKE Phase 1 exchange uses UDP Port 500.
- Consist of **2 phases:**

1. **IKE Pase1(IKE security assocaiation establishment) .** Equivalent to **SSL session establishment. Mutual authentication** accomplished. It has **4 different options:**
    1. **Public key encryption (original version)**
    2. **Public key encryption (improved version)**
    3. **Public key signature**
    4. **Symmetric key**
    - Each option has two modes:
        1. **Aggressive mode (3 messages)**
        2. **Main mode (6 messages)**
    - **There are 8 versions of IKE Phase 1!**
2. **IKE Pahse2(IPSec security association establishment). Equivalent to SSL connection establishment. Sharing of symmetric key happens here and re-authentication.**

- **Data Encapsulation . IPSec uses either ESP or AH.** Independent of ESP or AH IPSec can use either **transport mode** or **tunnel mode.**
    1. **IP Encapsulating Security Payload:** Encapsulating Security Payload is for confidentiality or integrity of IP packets. It provides encryption. ESP uses IP port 50.
    2. **IP Authentication Header :** provides integrity and authentication. Authentication is performed by computing a cryptographic hash-based message authentication code over nearly all the fields of the IP packet and stores this in a IPSec AH header. This AH header is injected between the original IP header and the payload.AH uses IP port 51.

IKE PHASE 1

MOTIVES

1. Establishes a secure, authenticated channel between the two computer
2. Authenticates and protects the identities of the peers
3. Negotiates what SA policy to use
4. Performs an authenticated shared secret keys exchange
5. Sets up a secure tunnel for phase 2

- Each of phase 1uses ephemeral Diffie-Hellman to establish session key to ensure Provides perfect forward secrecy (PFS)

Main Mode IKE

- Negotiate algorithms & hashes.
- Users identity is hidden
- Generate shared secret keys using a Diffie-Hillman exchange.
- Verification of Identities.
- Main mode **MUST** be implemented

Aggressive Mode IKE

- Squeezes all negotiation, key exchange, etc. into less packets.
- Users identity is revealed
- In aggressive mode, Alice chooses some Elgamal context *(p, g)*. Bob may not support it, and reject the connection. If that happens, Alice should try and connect to Bob using main mode.
- Aggressive mode provides mutual authentication, and a shared secret $g^{ab} \bmod p$, which can be used to derive keys for the symmetric crypto protocols.
- Aggressive mode **SHOULD** be implemented. So, if aggressive mode is not implemented, "you should feel guilty about it".
- Might create interoperability issues
  - **Advantage:** Less network traffic & faster than main mode
  - **Disadvantage:** Information exchanged before a secure channel is created. Vulnerable to sniffing

IKE Phase 1: Digital Signature (Main Mode)

Alice

IC, CP

IC,RC, CS

IC,RC, g^a mod p, R_A

IC,RC, g^b mod p, R_B

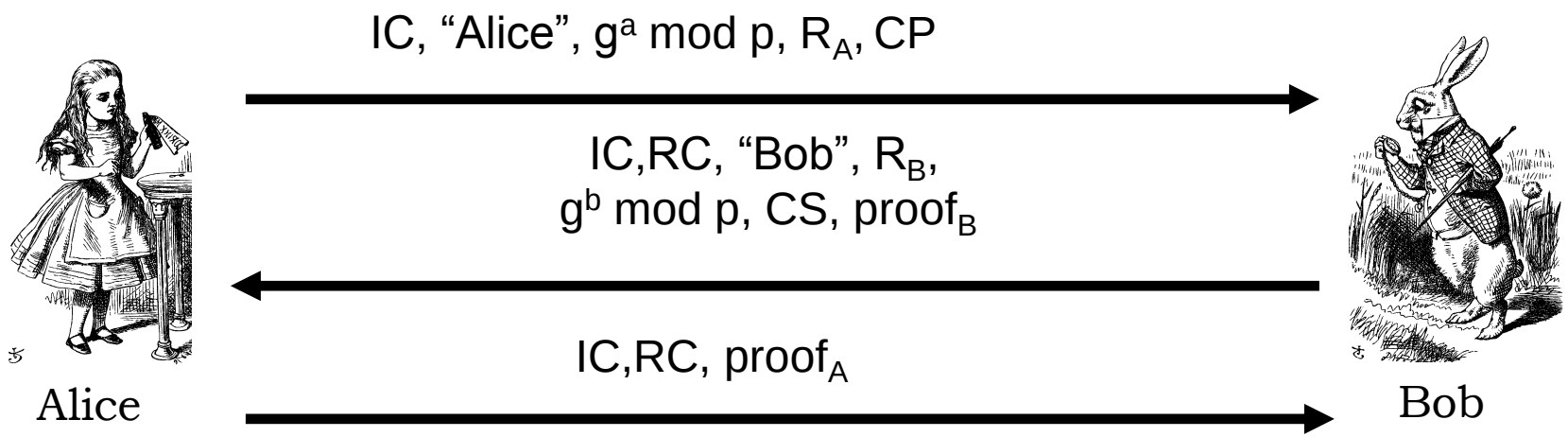
IC,RC, E("Alice", $proof_A$, K)

IC,RC, E("Bob", $proof_B$, K)

Bob

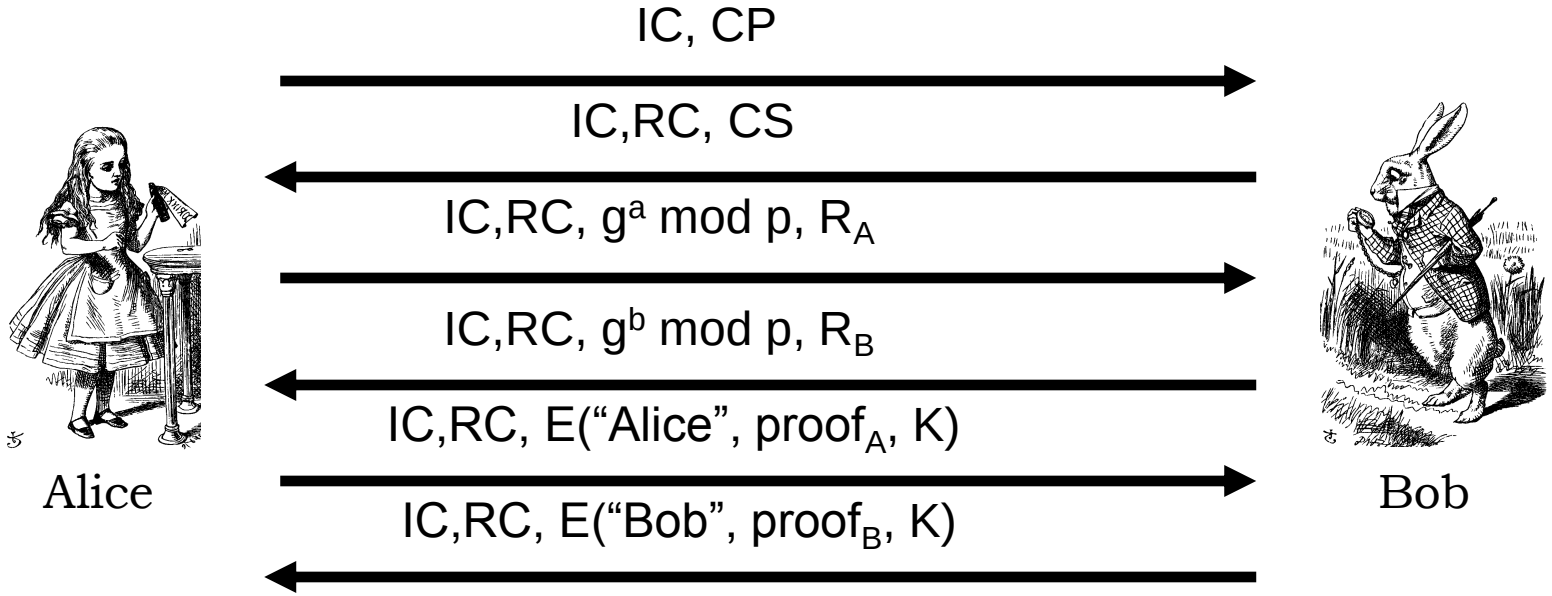
- CP = crypto proposed, CS = crypto selected
- IC = initiator "cookie", RC = responder "cookie"
- K = h(IC,RC,g^{ab} mod p,R_A,R_B)
- SKEYID = h(R_A, R_B, g^{ab} mod p)
- $proof_A$ = [h(SKEYID,g^a mod p,g^b mod p,IC,RC,CP,"Alice")]$_{Alice}$
- Let a be Alice's Diffie-Hellman exponent, b be Bob's Diffie-Hellman exponent, g be generator and p prime where p and g are public

IKE Phase 1: Public Key Signature (Aggressive Mode)



- Main differences from main mode and aggressive mode
  - Main mode can negotiate the values of g and p as part of crypto proposed and crypto accepted but in aggressive mode cannot negotiate g or p
  - A passive attacker knows identities of Alice and Bob in aggressive mode, but not in main mode.
- In aggressive mode, Alice chooses some DH context (p, g) and sends that in the first message exchange. Bob may not support it, and reject the connection. If that happens, Alice should try and connect to Bob using main mode.

IKE Phase 1: Symmetric Key Encryption (Main Mode)

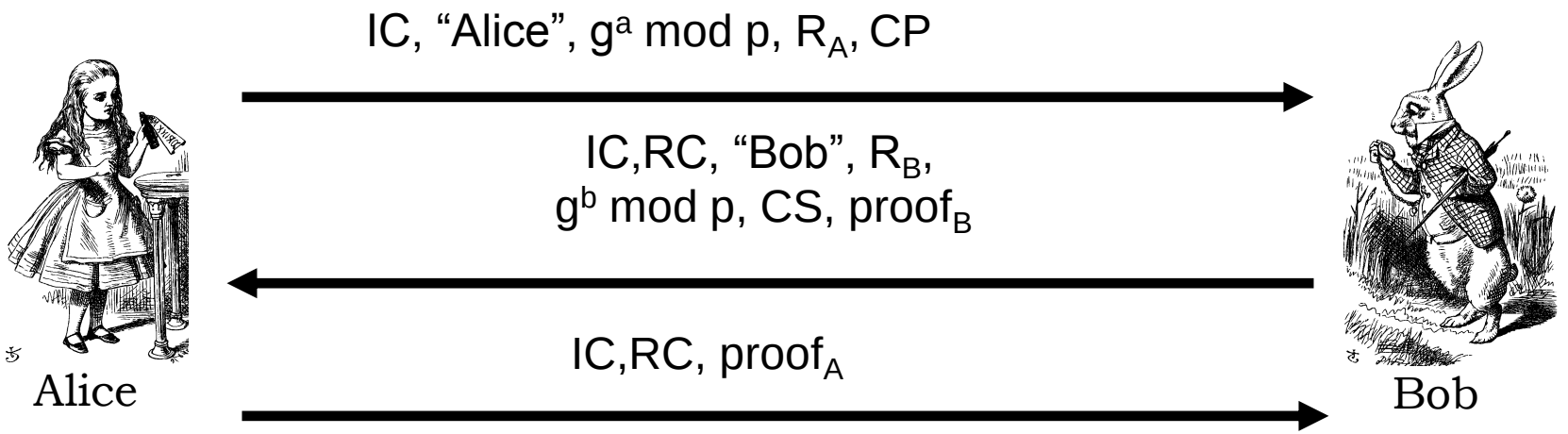


- K_{AB}= shared symmetric key in advance
- CP = crypto proposed, CS = crypto selected
- IC = initiator "cookie", RC = responder "cookie"
- K = h(IC,RC,g^{ab} mod p,R_A,R_B, K_{AB})
- SKEYID = h(K, g^{ab} mod p)
- $proof_A$ = h(SKEYID, g^a mod p, g^b mod p, IC, RC, CP, "Alice")

Problems with Symmetric Key (Main Mode)

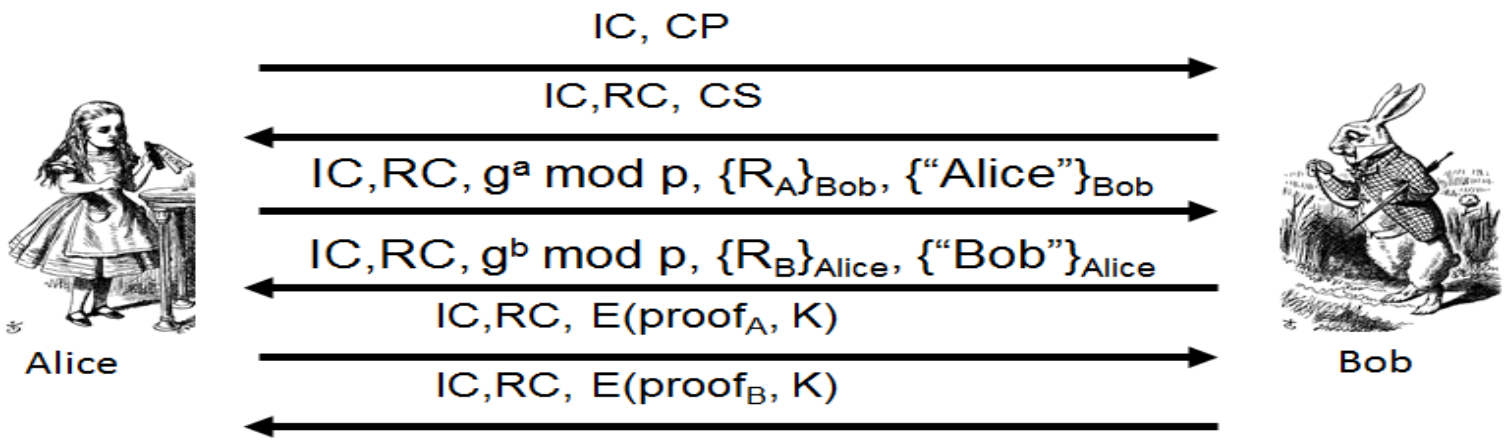
- Catch-22
  - Alice sends her Identity encrypted with K in message 5
  - To find K Bob must know K_{AB}
  - To get K_{AB} Bob must know he's talking to Alice!
- Result: **Alice's IP address used as ID!**
- Useless mode for the "road warrior"
- Why go to all of the trouble of trying to hide identities in 6 message protocol?
- Hence symmetric key main mode fails to hide Alice's identity.

IKE Phase 1: Symmetric Key (Aggressive Mode)



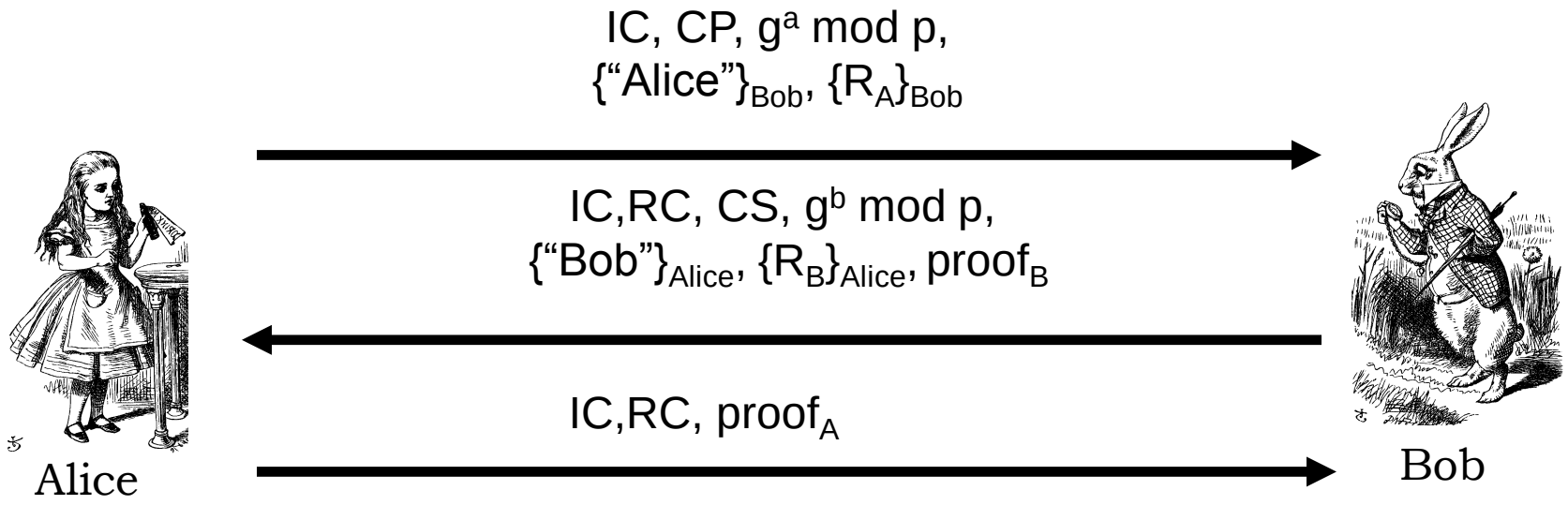
- Same format as digital signature aggressive mode
- Not trying to hide identities…
- As a result, does **not** have problems of main mode
- But does not (pretend to) hide identities

IKE Phase 1: Public Key Encryption (Main Mode)



- CP = crypto proposed, CS = crypto selected
- IC = initiator “cookie”, RC = responder “cookie”
- K = h(IC,RC,g^{ab} mod p,R_A,R_B)
- SKEYID = h(R_A, R_B, g^{ab} mod p)
- $proof_A$ = h(SKEYID,g^a mod p,g^b mod p,IC,RC,CP,“Alice”)

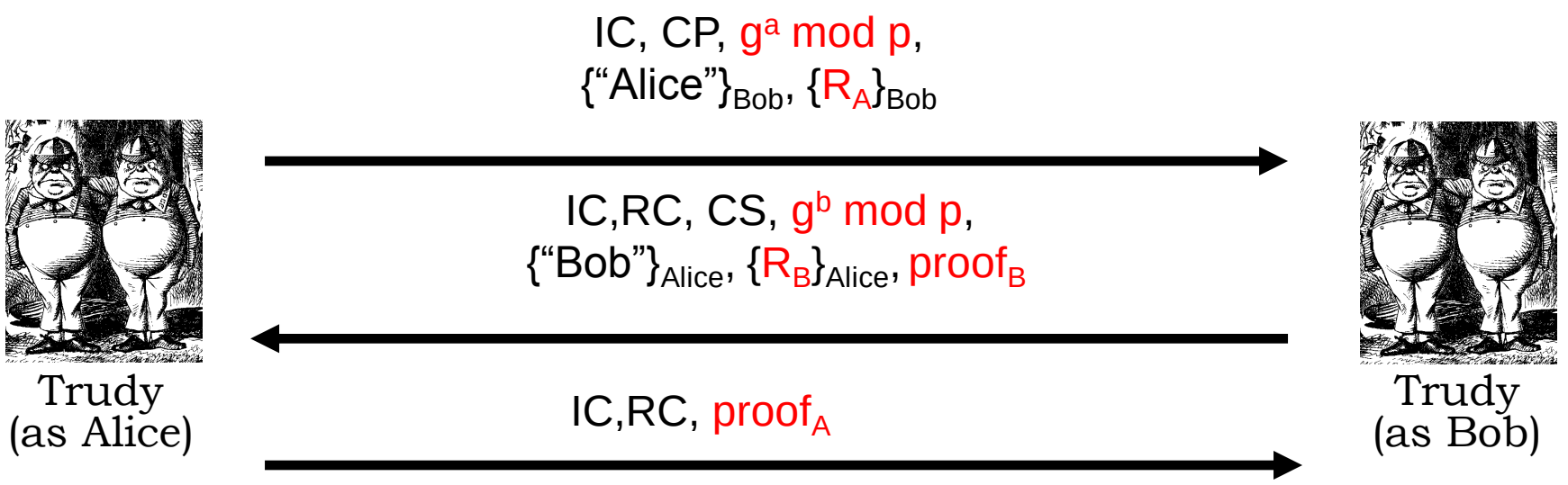
IKE Phase 1: Public Key Encryption (Aggressive Mode)



- K, $proof_A$, $proof_B$ computed as in main mode
- Note that identities are hidden
  - The only aggressive mode to hide identities
  - So, why have a main mode?

Public Key Encryption Issue?

- In public key encryption, aggressive mode…
- Suppose **Trudy** generates
  - Exponents **a** and **b**
  - Nonces **R_A** and **R_B**
- Trudy can compute "valid" keys and proofs: **g^{ab} mod p**, **K**, **SKEYID**, **$proof_A$** and **$proof_B$**. Since public keys of Alice and Bob are public.
- All of this also works in main mode



Plausible Deniability

- Trudy can create messages that appears to be between Alice and Bob
- Appears valid to any observer, **including Alice and Bob!**
- Trudy can create fake "conversation" that appears to be between Alice and Bob
  - Appears valid, even to Alice and Bob!
- A security ***failure***?
- In IPSec public key option, it is a ***security feature…***
  - **Plausible deniability:** A protocol that includes plausible deniability allows Alice and Bob to deny that any conversation took place!
- In some cases it might create a problem
  - E.g., if Alice makes a purchase from Bob, she could later repudiate it (unless she had signed)

IKE Phase 1 Summary

- Result of IKE phase 1 is
  - Mutual authentication
  - Shared symmetric key
  - IKE **Security Association (SA)**
- But phase 1 is expensive
  - Especially in public key and/or main mode
- Developers of IKE thought it would be used for lots of things not just IPSec
  - Partly explains the over-engineering…

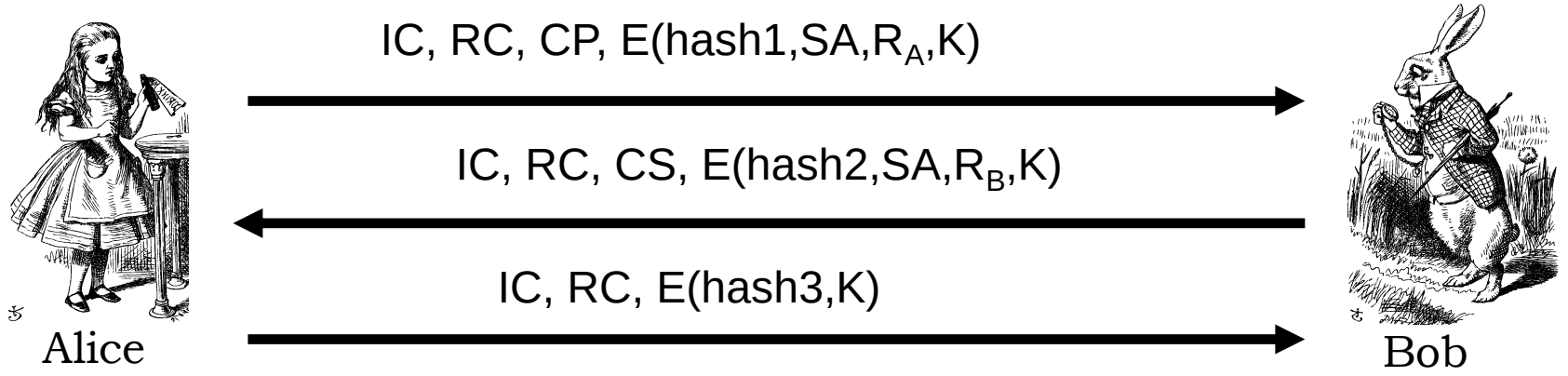
IKE Phase 1 Cookies

- IC and RC cookies (or "anti-clogging tokens") supposed to prevent DoS attacks
  - No relation to Web cookies
- To reduce DoS threats, Bob wants to remain **stateless** as long as possible and IPSec cookies helps in accomplishing this.
- But Bob must remember CP from message 1 (required for proof of identity in message 6) in main mode
- Bob must keep state from 1st message on to calculate $Proof_B$ in message 6. So, these "cookies" offer little DoS protection

IKE Phase 2

- IKE Phase 1 establishes IKE SA. SSL session is comparable to IKE Phase 1.
- IKE Phase 2 establishes **IPSec SA.** SSL connections are like IKE Phase 2.
- IKE Phase 1 establishes shared symmetric key K, IP Sec Cookies-IC, RC and IKE Security Association.
- CP includes ESP or AH or both.
- SA identifies IKE SA from phase 1.
- An AH or ESP packet is then sent using the agreed upon “main” SA during the IKE phase 1.
- IKE Phase 2
  - Negotiates IPSec SA parameters
  - Establishes IPSec security associations for specific connections (like FTP, telnet, etc)
  - Renegotiates IPSec SAs periodically
  - Optionally performs an additional Diffie-Hellman exchange
- Once Phase 2 has established an SA for a particular connection, all traffic on that connection is communicated using the SA.

IKE Phase 2



- Key K, IC, RC and SA known from Phase 1
- Proposal CP includes ESP and/or AH
- Hashes 1,2,3 depend on SKEYID, SA, R_A, R_B(R_A, R_B are not same as those of phase 1) and IKE SA (from phase 1).
- Keys derived from KEYMAT = h(SKEYID,R_A,R_B,junk), where junk is known to all.
- Recall SKEYID depends on phase 1 key method
- Optional PFS (via ephemeral Diffie-Hellman exchange)
- After IKE Phase 1 and IKE Phase 2, we have we have an IKE SA and IPSec SA with Authentication completed and have a shared symmetric key (session key)
- Now what? We want to protect **IP datagrams**

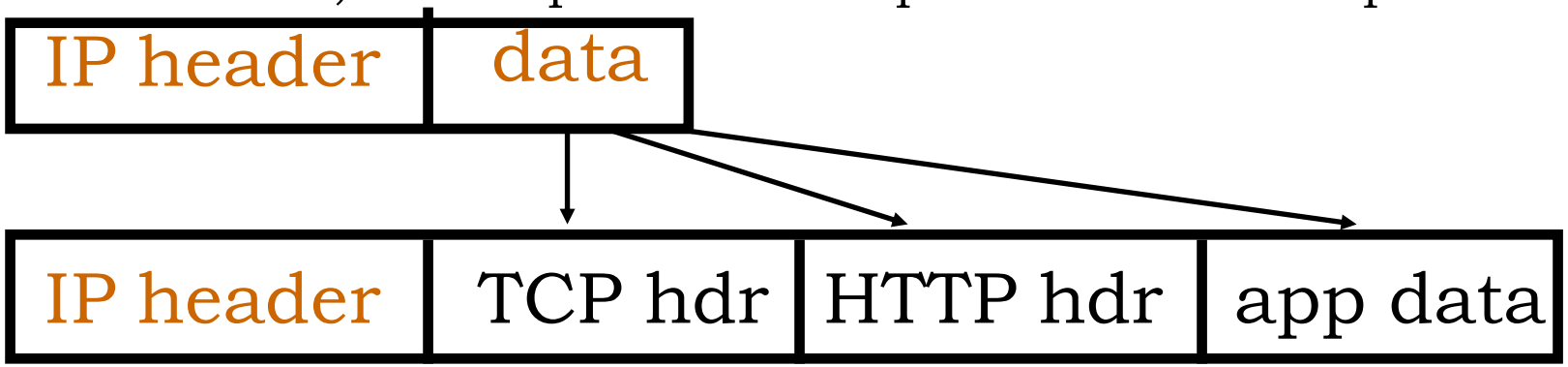
IPSEC protecting IP datagrams

- IP datagram is of the form.

- Routers must see IP header to route the packet and hence shouldn't be encrypted.
- Some fields of of IP header changes as the packet is forwarded. Eg: TTL

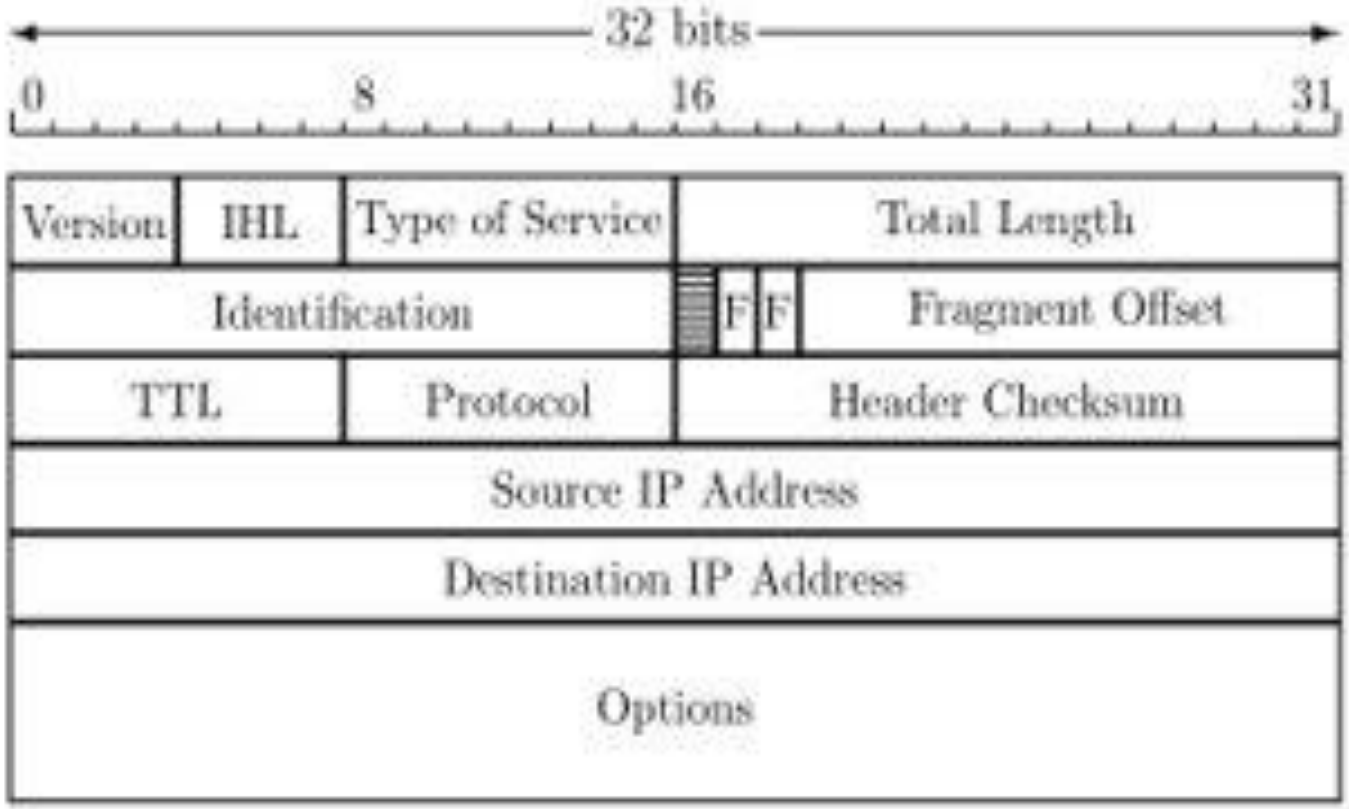
IP and TCP

- Consider Web traffic, IP encapsulates TCP packet which encapsulates HTTP packet.



- IP data includes TCP header, HTTP header

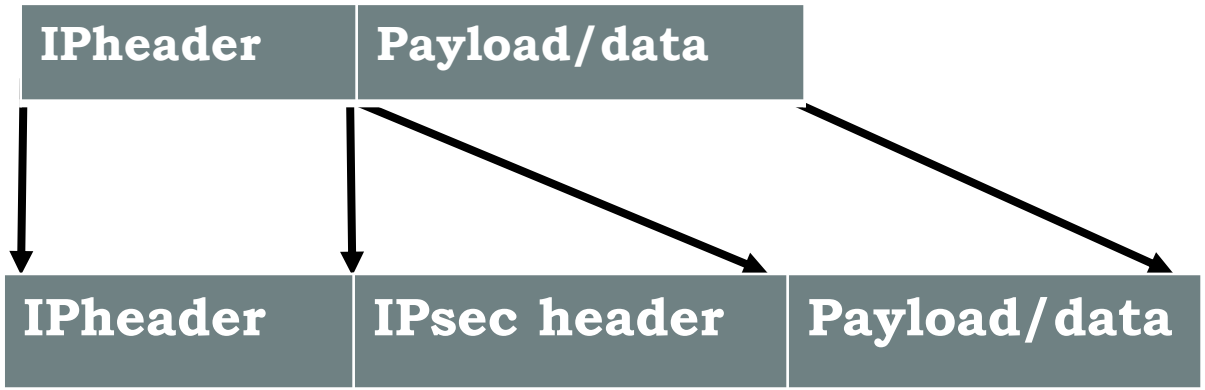
IP Header


32 bits
0
8
16
31
Version
IHL
Type of Service
Total Length
Identification
F
F
Fragment Offset
TTL
Protocol
Header Checksum
Source IP Address
Destination IP Address
Options

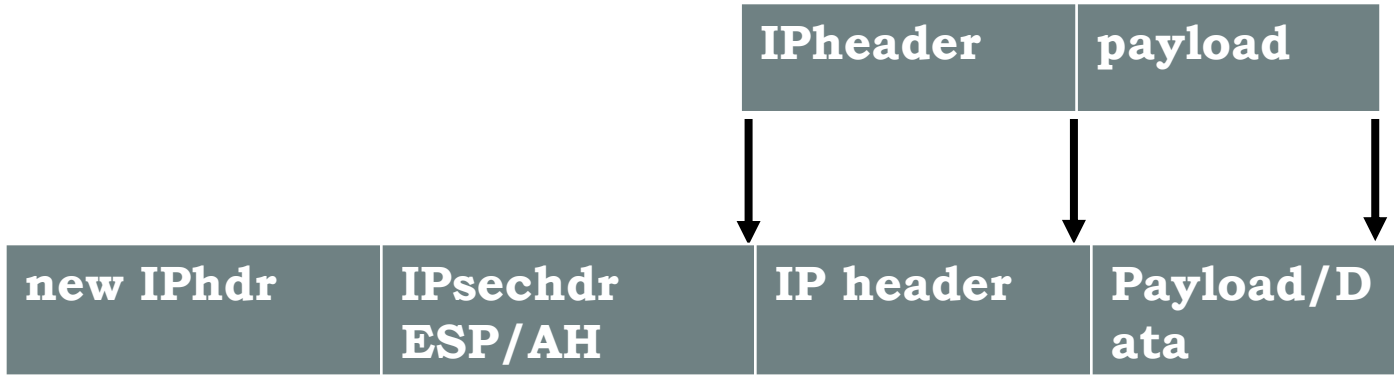
IPSec Modes

- Two modes of encapsulating **IPsec data into an IP packet ie, Transport mode and tunnel mode** independent of whether ESP or AH is used

1. **Transport Mode**
   - IPsec header(ESP/AH) is inserted into the IP packet **sandwiched between IP header and data.** Hence its **efficient** since it adds minimal amount of additional header data.
   - Only **IP payload is encrypted and authenticated**. IP headers are left in tact and may or may not be authenticated. Passive attacker can see who is talking.
   - Save bandwidth in end-to-end associations.
   - Used to deliver services from **host to host** or from **host to gateway,** usually within the same network, but can also be end-to-end across networks
   - Protect the upper layer protocols.



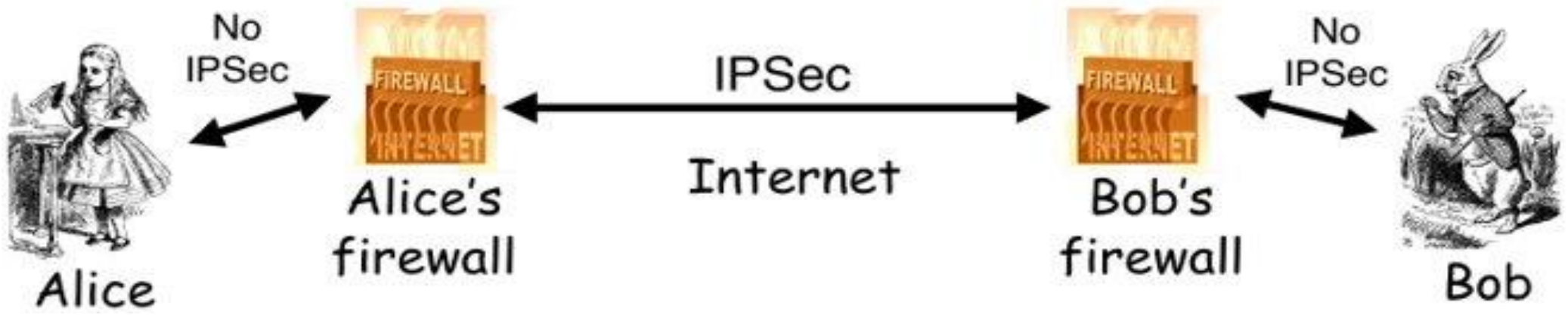
Tunnel Mode



- Tunnel mode protects both the payload and IP header of the original packet by encapsulating entire IP packet into a new packet.
- Since entire IP packet is encrypted new IP headers are generated for this packet. Hence IP header is no longer visible to an attacker. Attacker does not know which hosts are talking
- If encryption is used between gateways in tunnel mode, then it reduces information for traffic analysis.
- Used Tunnel mode is used for *firewall-to-firewall* traffic to deliver services from host to network(gateways) or network to network or usually gateways owned by the same organization. New IP header from firewall to firewall.
- Disadvantage is overhead of an additional IP header.

IPSec: Firewall-to-Firewall

- IPSec tunnel mode used for firewall to firewall protected traffic, as we must preserve the original IP header to correctly route the packet from destination firewall to destination host.



- Note: Local networks not protected. Is there any advantage here for transport mode?
- Transport mode is more efficient and preferable in traffic protected from host to host. Tunnel ode would not protect source and destination address in host to host, since these addresses would be repeated in new header.

IPSec Security

- What kind of protection? Confidentiality? Integrity? Or Both?
- What to protect? Data? Header? Both?
- ESP/AH allow some combinations of these.

AH (Authentication Header)

- Provides **Integrity only** (no confidentiality)
- With AH No encryption
- Integrity-protect **everything beyond IP header** and **some fields of header** (why not all fields?) ie **immutable fields**

ESP (Encapsulating Security Payload)

- Provides both **Integrity and confidentiality**
- Protects everything beyond IP header
- Integrity-only by using NULL encryption

Authentication Header

- This information is added to the header to provide the following services:
  - Access control, connectionless integrity, data origin authentication, rejection of replayed packets
  - Information added are:
    - **Sequence number (**32-bit)
    - **Integrity check value** (variable, multiple of 32-bits)
- **Sequence number**
  - Prevents Anti-replay attacks
  - Range of sequence numbers for session is 2^{32}-1
  - Sequence numbers are not reused
- **Integrity Check Value (ICV)**
  - Keyed MAC algorithms used: AES, MD5, SHA-1
  - MAC is calculated over immutable fields in transit (source/dest. addr, IP version, header length, packet length)

ESP

- Authentication Header (AH) will protect data from tampering, but it will not stop people from seeing it.
- it can provide authentication, replay-proofing and integrity checking.
  It acomplishes this by adding 3 separate components:
  - 1) An ESP header
  - 2) An ESP trailer and
  - 3) An ESP authentication block.

ESP NULL Encryption

- According to RFC 2410
  - NULL encryption “is a block cipher the origins of which appear to be lost in antiquity”
  - “Despite rumors”, there is no evidence that NSA “suppressed publication of this algorithm”
  - Evidence suggests it was developed in Roman times as exportable version of Caesar’s cipher
  - Can make use of keys of varying length
  - No IV is required
  - Null(P,K) = P for any P and any key K
- Is ESP with NULL encryption same as AH ?

Why Does AH Exist?

- Cannot encrypt IP header and AH provides integrity for AH header whereas ESP doesn't provide.
  - Routers must look at the IP header
  - IP addresses, TTL, etc.
  - IP header exists to route packets!
- AH protects **immutable fields** in IP header
  - Cannot integrity protect all header fields
  - TTL, for example, will change
- ESP does not protect IP header at all
- Firewalls won't be able to look into TCP header if its encrypted.

- ESP encrypts everything beyond the IP header (if non-null encryption)
- If ESP-encrypted, firewall cannot look at TCP header in host-to-host case
- Why not use ESP with NULL encryption?
  - Firewall sees ESP header, but does not know whether null encryption is used
  - End systems know, but **not** the firewalls

Security Association(SA)

- A **security association (SA)** is a set of security information that describes a particular kind of secure connection between one device and another.
- A device's security associations are contained in its **security association database (SAD).**
- SA determines how packets are processed
  - Cryptographic algorithms, keys, AH/ESP, lifetimes, sequence numbers, mode (transport or tunnel) SA is uniquely identified by {SPI, dst IP addr, flag}
  - SPI: Security Parameter Index
- Chosen be destination (unless traffic is multicast…)
  - Flag: ESP or AH
  - Each IPsec implementation keeps a database of SAs
  - SPI is sent with packet, tells recipient which SA to use

Security Architecture of Message Transfer

- When Alice is sending to Bob:
  - Consult “security policy database” (SPD) to check if packet should protected with IPsec or not (“selector” fields)
  - SPD provides pointer to the associated SA entry in the security association database (SAD)
  - SA provides SPI, algorithm, key, sequence number, etc.
  - Include the SPI in the message
- When Bob receives a message:
  - Lookup the SA based on the destination address and SPI (In a multicast message the address is not Bob's own)
  - Find algorithm, key, sequence number, etc.
  - After decrypting message, verify that packet matches “selector” in the policy database (SPD)

Benefits of IPSec

- in a firewall/router provides strong security to all traffic crossing the perimeter
- in a firewall/router is resistant to bypass
- is below transport layer, hence transparent to applications
- can be transparent to end users
- can provide security for individual users
- secures routing architecture
- Offers Confidentiality (encrypting data), Integrity , and Authentication
- Data integrity and source authentication
  - Data “signed” by sender and “signature” is verified by the recipient –
  - Modification of data can be detected by signature “verification” – Because “signature” is based on a shared secret, it gives source authentication
- Anti-replay protection
  - Optional; the sender must provide it but the recipient may ignore
- Key management
  - IKE – session negotiation and establishment
  - Sessions are rekeyed or deleted automatically
  - Secret keys are securely established and authenticated
  - Remote peer is authenticated through varying options

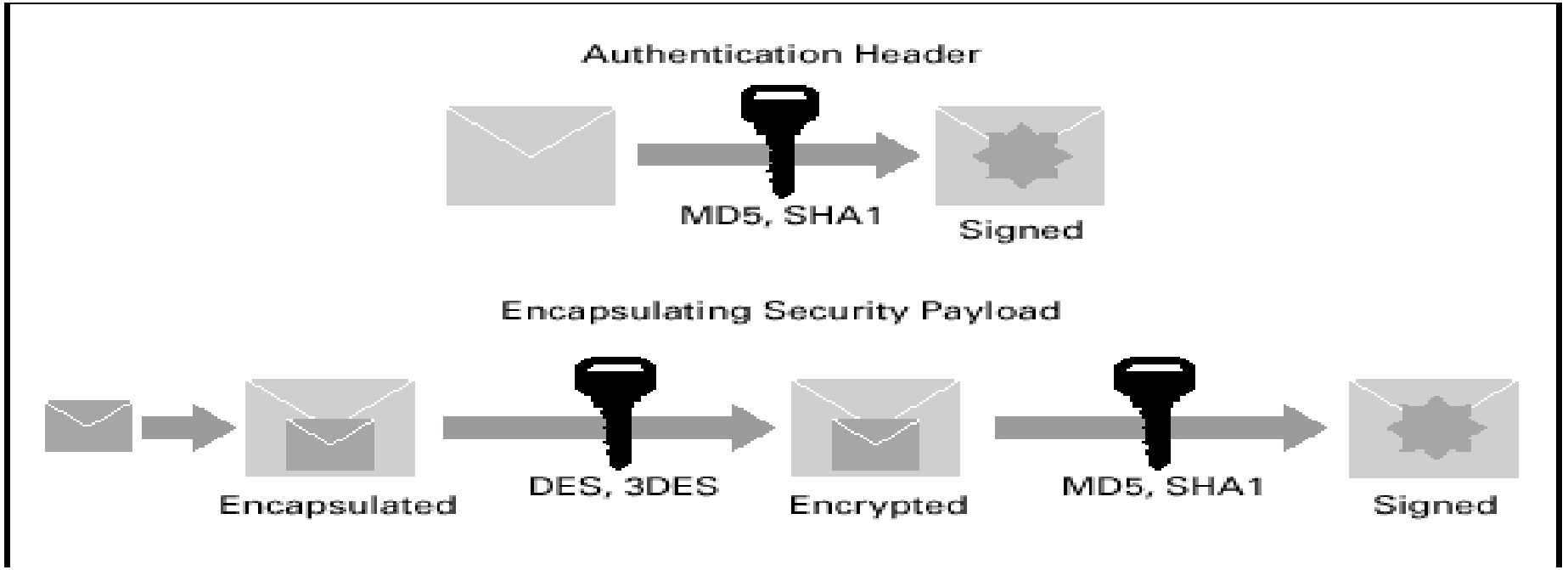
Why not use IPSec?

- Processor overhead to encrypt & verify each packet can be great.
- Added complexity in network design.
- **Over-engineered**
  - Lots of (generally useless) features
- Interoperability is serious challenge
- IPSec is a complex protocol
- Flawed Some significant security issues

SSL vs IPSec

- **IPSec**
  - Lives at the **network layer (part of the OS)**
  - Provides **Encryption, integrity, authentication**, etc.
  - Is **complex,** has some security "issues"
  - Secure virtual private network also for IPV6
- SSL (known as TLS)
  - Lives at **socket layer (part of user space)**
  - Provides **Encryption, integrity, authentication,** etc.
  - Relatively simple and elegant specification
  - Secure web transactions

Data Integrity and Confidentiality



Basic difference between AH and ESP

Thank You

anooja@somaiya.edu