CSCI 466: Networks

Operational Security (Firewalls, Protocols, Etc)

Reese Pearsall

Fall 2023

Announcements

PA4 due **tonight**

Wireshark Lab 4 due next Wednesday

Gradebook

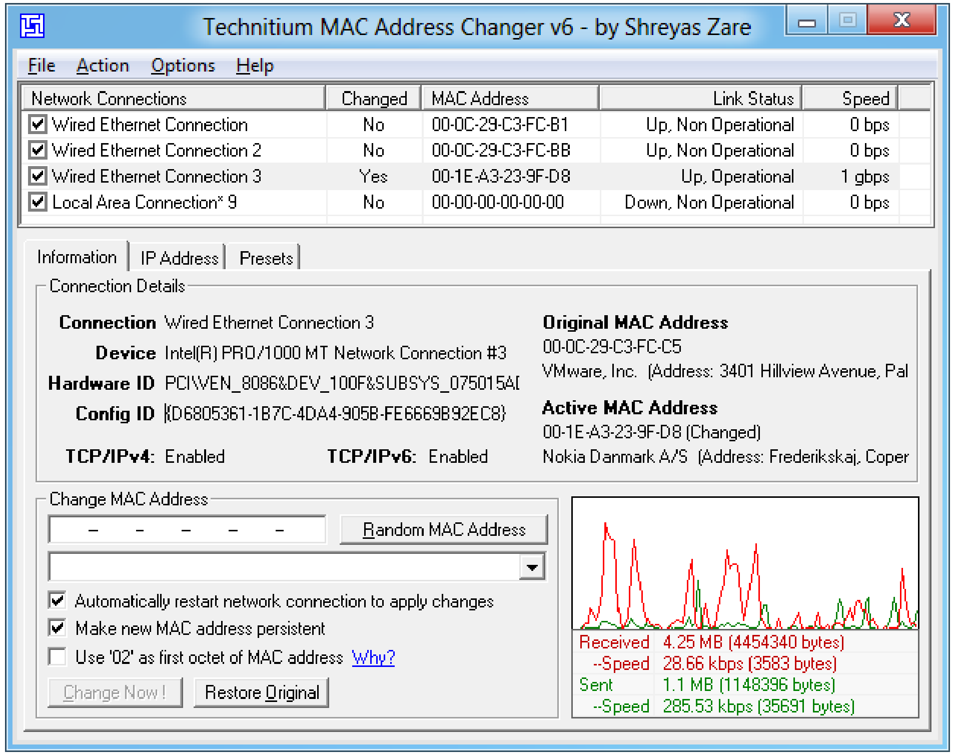
Final Exam is next Friday (12/8) (**In-person**)

Quiz on Friday

Wireshark Lab 4

Spoofing MAC Addresses

We used `scapy` to spoof network layer and transport layer information, but we can also use it to spoof MAC addresses or things like ARP packets



Network Connections	Changed	MAC Address	Link Status	Speed
☑ Wired Ethernet Connection	No	00-0C-29-C3-FC-B1	Up, Non Operational	0 bps
☑ Wired Ethernet Connection 2	No	00-0C-29-C3-FC-BB	Up, Non Operational	0 bps
☑ Wired Ethernet Connection 3	Yes	00-1E-A3-23-9F-D8	Up, Operational	1 gbps
☑ Local Area Connection* 9	No	00-00-00-00-00-00	Down, Non Operational	0 bps

Connection Details

Connection Wired Ethernet Connection 3
Device Intel(R) PRO/1000 MT Network Connection #3
Hardware ID PCI\VEN_8086&DEV_100F&SUBSYS_075015A[
Config ID {D6805361-1B7C-4DA4-905B-FE6669B92EC8}
TCP/IPv4: Enabled **TCP/IPv6:** Enabled

Original MAC Address
00-0C-29-C3-FC-C5
VMware, Inc. (Address: 3401 Hillview Avenue, Pal

Active MAC Address
00-1E-A3-23-9F-D8 (Changed)
Nokia Danmark A/S (Address: Frederikskaj, Coper

Change MAC Address

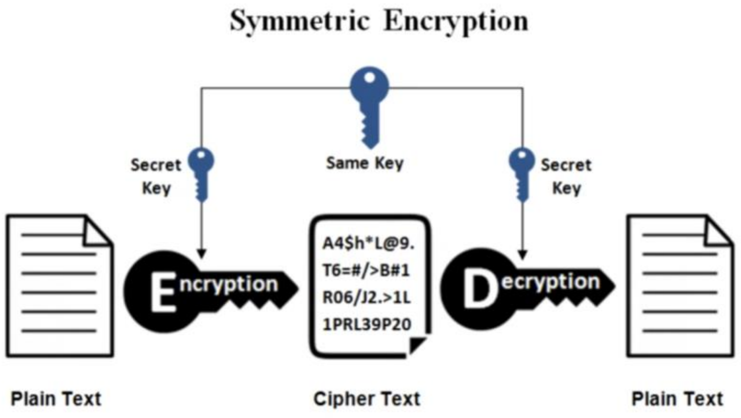
☑ Automatically restart network connection to apply changes
☑ Make new MAC address persistent
☐ Use '02' as first octet of MAC address Why?

Received 4.25 MB (4454340 bytes)
--Speed 28.66 kbps (3583 bytes)
Sent 1.1 MB (1148396 bytes)
--Speed 285.53 kbps (35691 bytes)

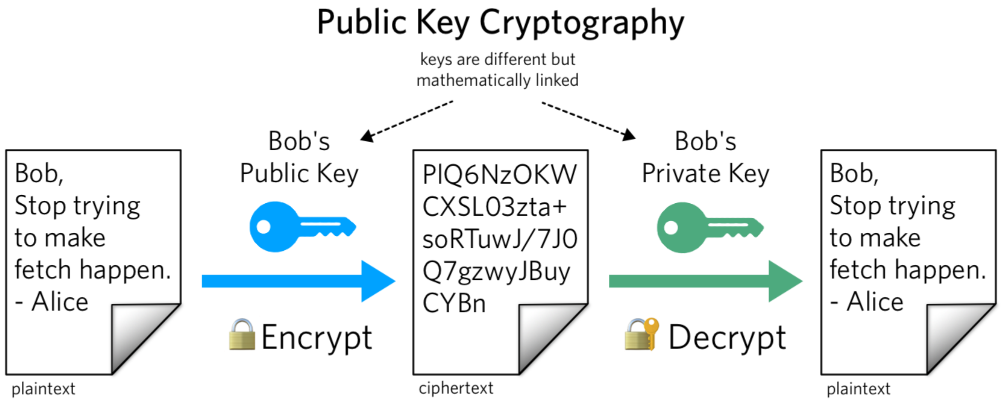
There is a variety of tools out there that allow you to override the MAC addresses of your NIC

Technitium is one program that overrides Windows settings to spoof MAC addresses

Review



- Same key used for encrypting and decrypting
- Using block ciphers (AES), we can encrypt an arbitrary size of data
- Issue: How to securely share secret keys with each other?



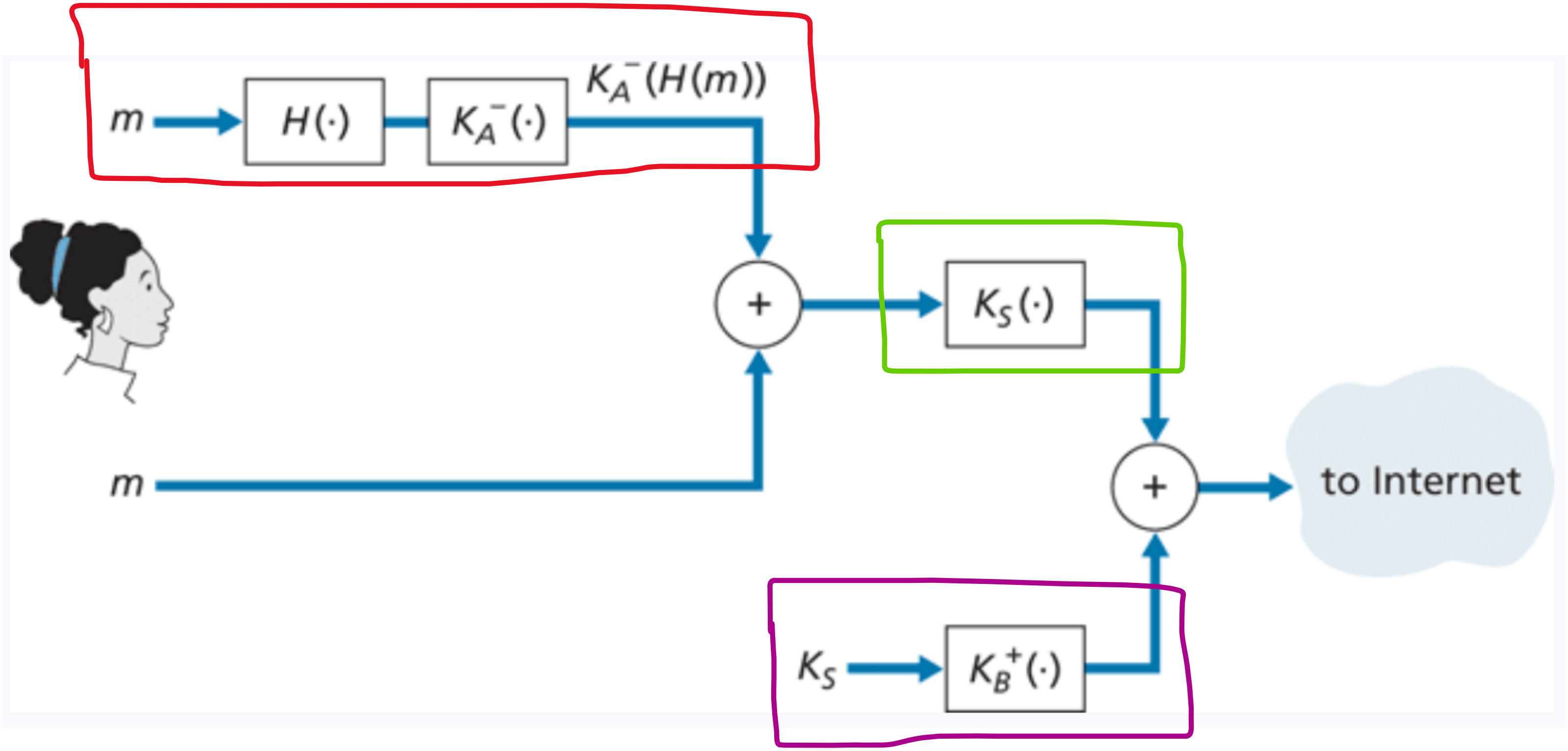
- Two keys: Public Key (a lock), and a price key (the key)
- Public key is used to encrypt. Private key used to decrypt message
- Using math, we can securely send messages over an unsecure channel without sharing any sensitive information
- Issue: We can not encrypt stuff bigger than our key (2048 bits)

- Symmetric and asymmetric cryptography are used **together**

(use RSA to send the key for symmetric crypto!)

Review

Symmetric Crypto, Asymmetric Crypto, and Hashing all work together to send secure, authentic messages

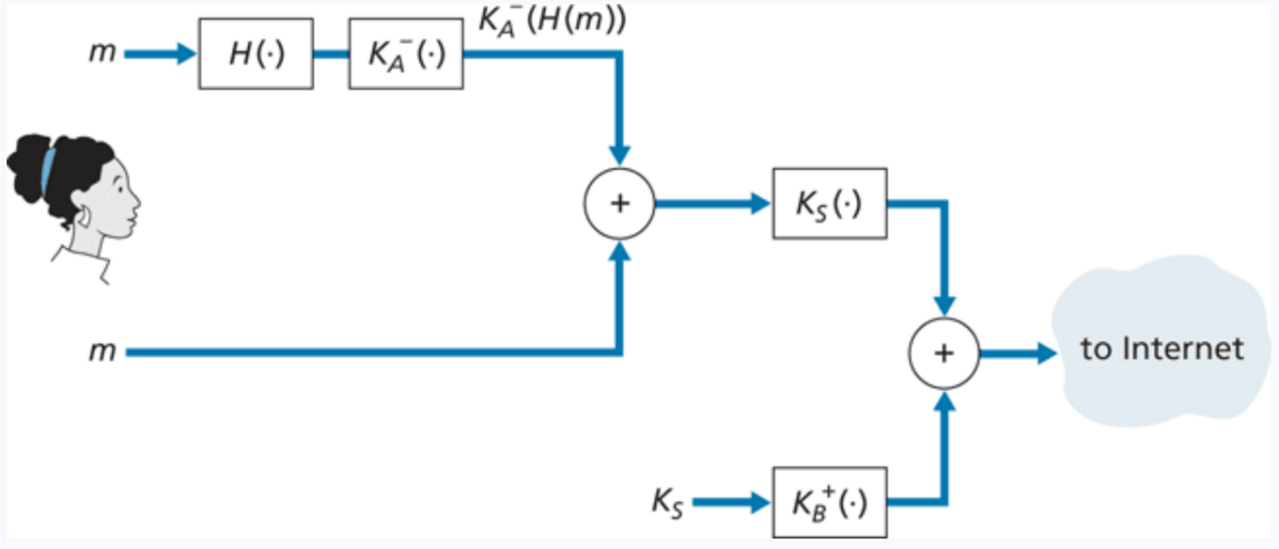


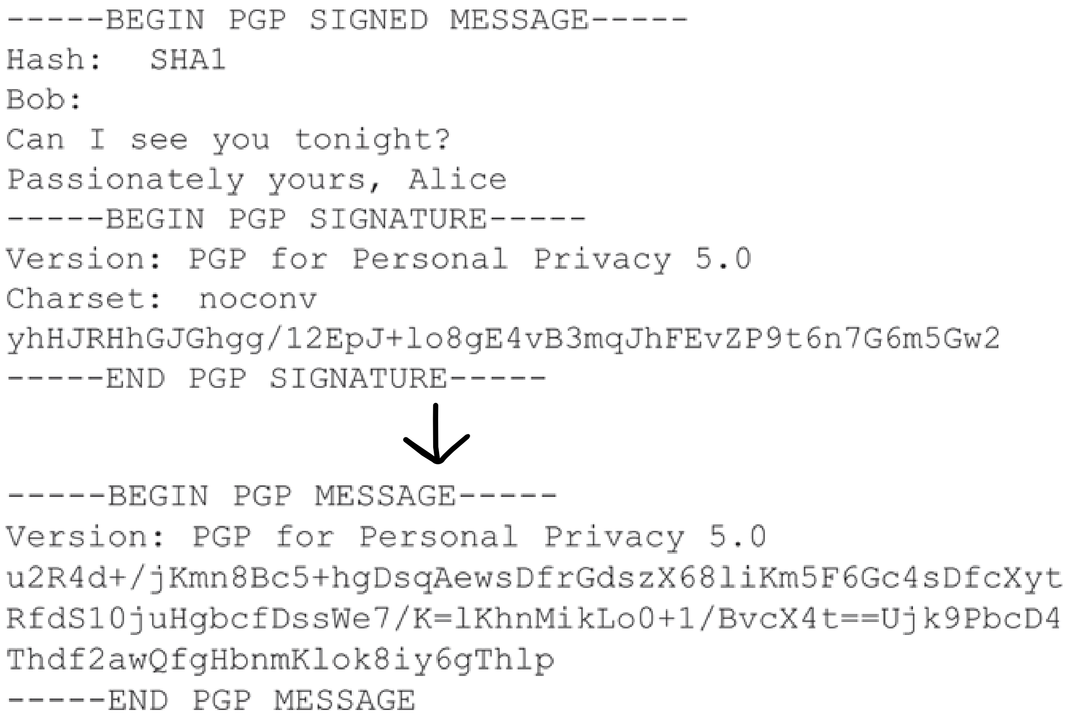
PGP (Pretty Good Privacy)

Email encryption scheme that involves sending a signed hash and encrypted message, and receiver decrypts with public key

PGP software is installed, and public key is generated for user

Private key is generated and protected by a user password





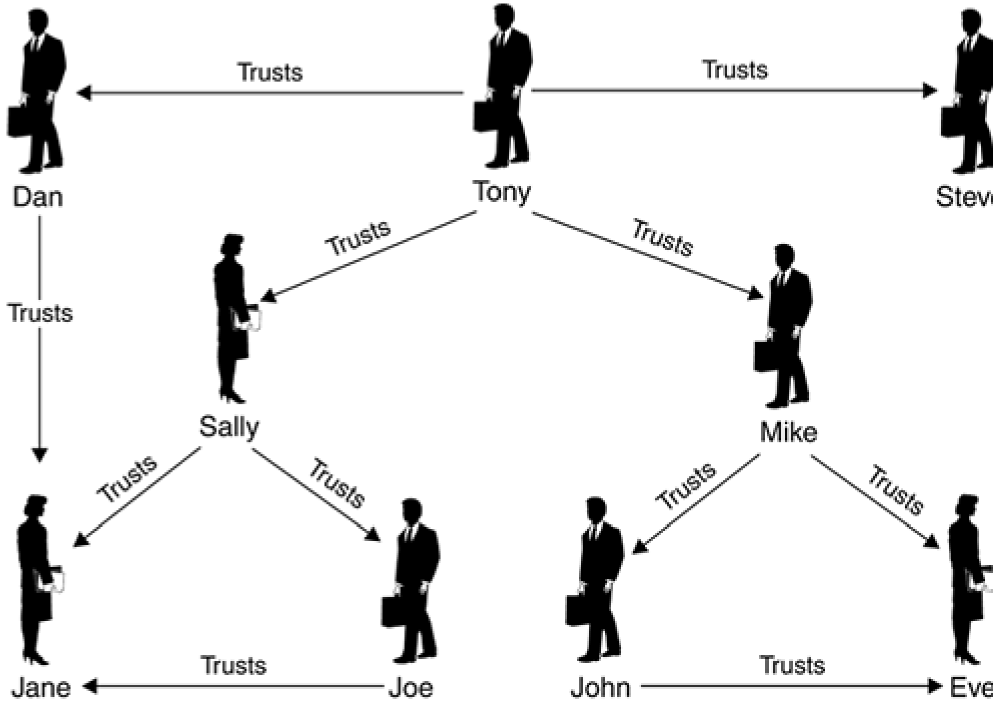
```
-----BEGIN PGP SIGNED MESSAGE-----
Hash:  SHA1
Bob:
Can I see you tonight?
Passionately yours, Alice
-----BEGIN PGP SIGNATURE-----
Version: PGP for Personal Privacy 5.0
Charset:  noconv
yhHJRHhGJGhgg/12EpJ+lo8gE4vB3mqJhFEvZP9t6n7G6m5Gw2
-----END PGP SIGNATURE-----
```

```
-----BEGIN PGP MESSAGE-----
Version: PGP for Personal Privacy 5.0
u2R4d+/jKmn8Bc5+hgDsqAewsDfrGdszX68liKm5F6Gc4sDfcXyt
RfdS10juHgbcfDssWe7/K=lKhnMikLo0+1/BvcX4t==Ujk9PbcD4
Thdf2awQfgHbnmKlok8iy6gThlp
-----END PGP MESSAGE
```

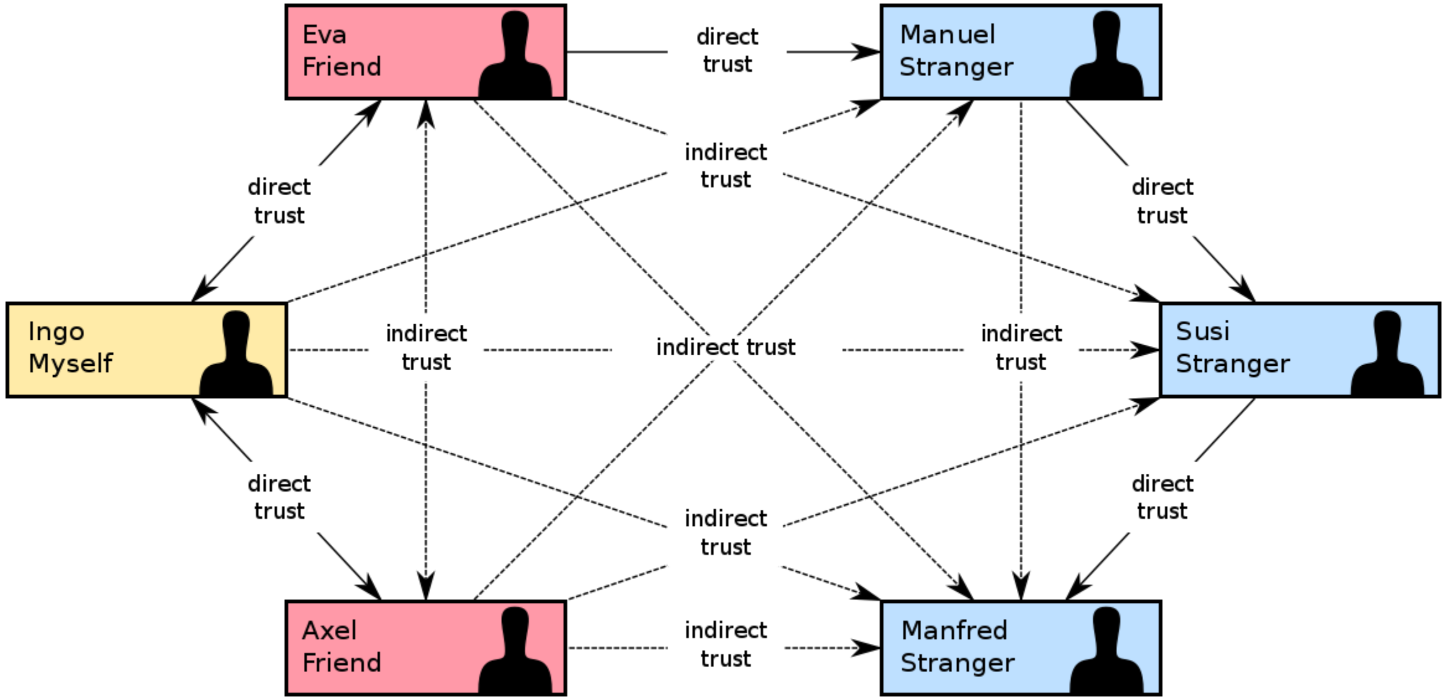
PGP provides a mechanism for public key certification.

Public keys are certified by a **web of trust** (other users will vouch for other users)

Web of Trust



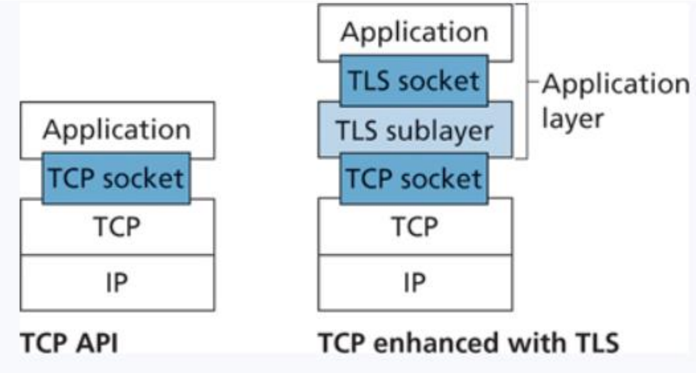
Trusts
Trusts
Dan
Tony
Stev
Trusts
Trusts
Trusts
Sally
Mike
Trusts
Trusts
Trusts
Trusts
Jane
Trusts
Joe
John
Trusts
Eve



Eva
Friend
direct
trust
Manuel
Stranger
indirect
trust
direct
trust
direct
trust
Ingo
Myself
indirect
trust
indirect trust
indirect
trust
Susi
Stranger
direct
trust
indirect
trust
direct
trust
Axel
Friend
indirect
trust
Manfred
Stranger

TLS

- **Transport Layer Security (TLS)** is a protocol used to provide communication security over a TCP connection
  - This exists somewhere between the application layer and transport layer

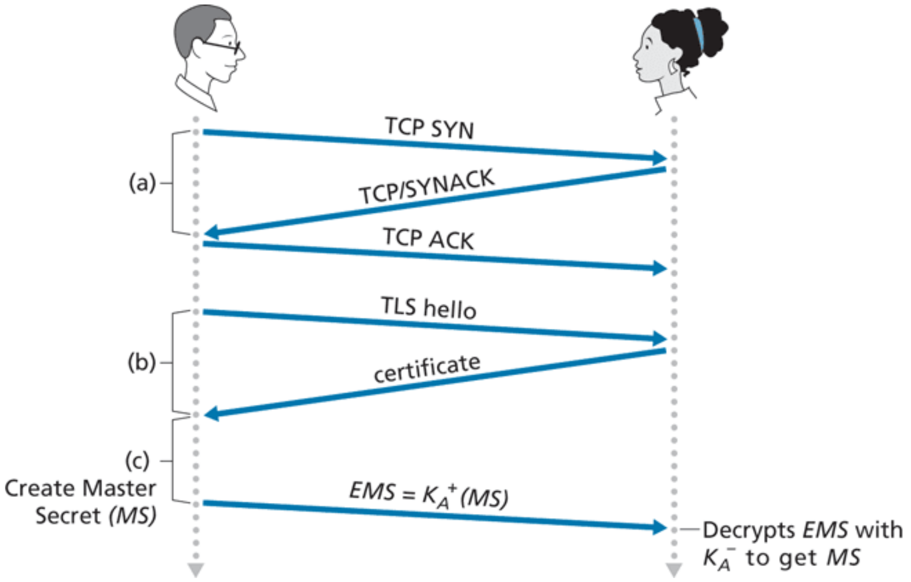


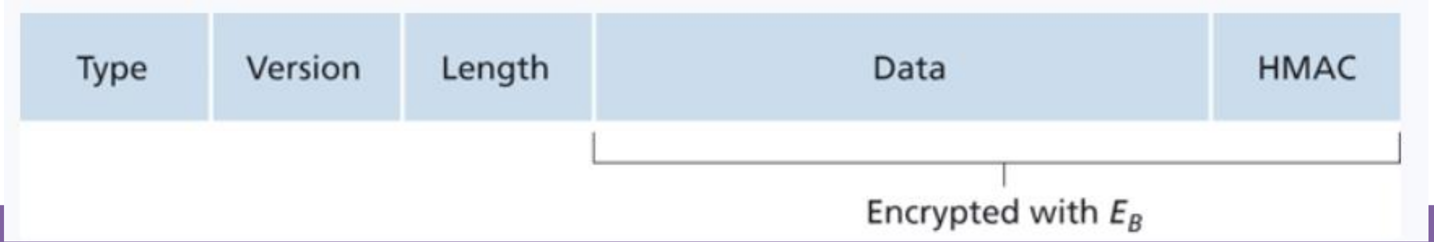
TLS will always be running if you are doing web communication with https

https = Hypertext Transfer Protocol ***Secure***

Port 443

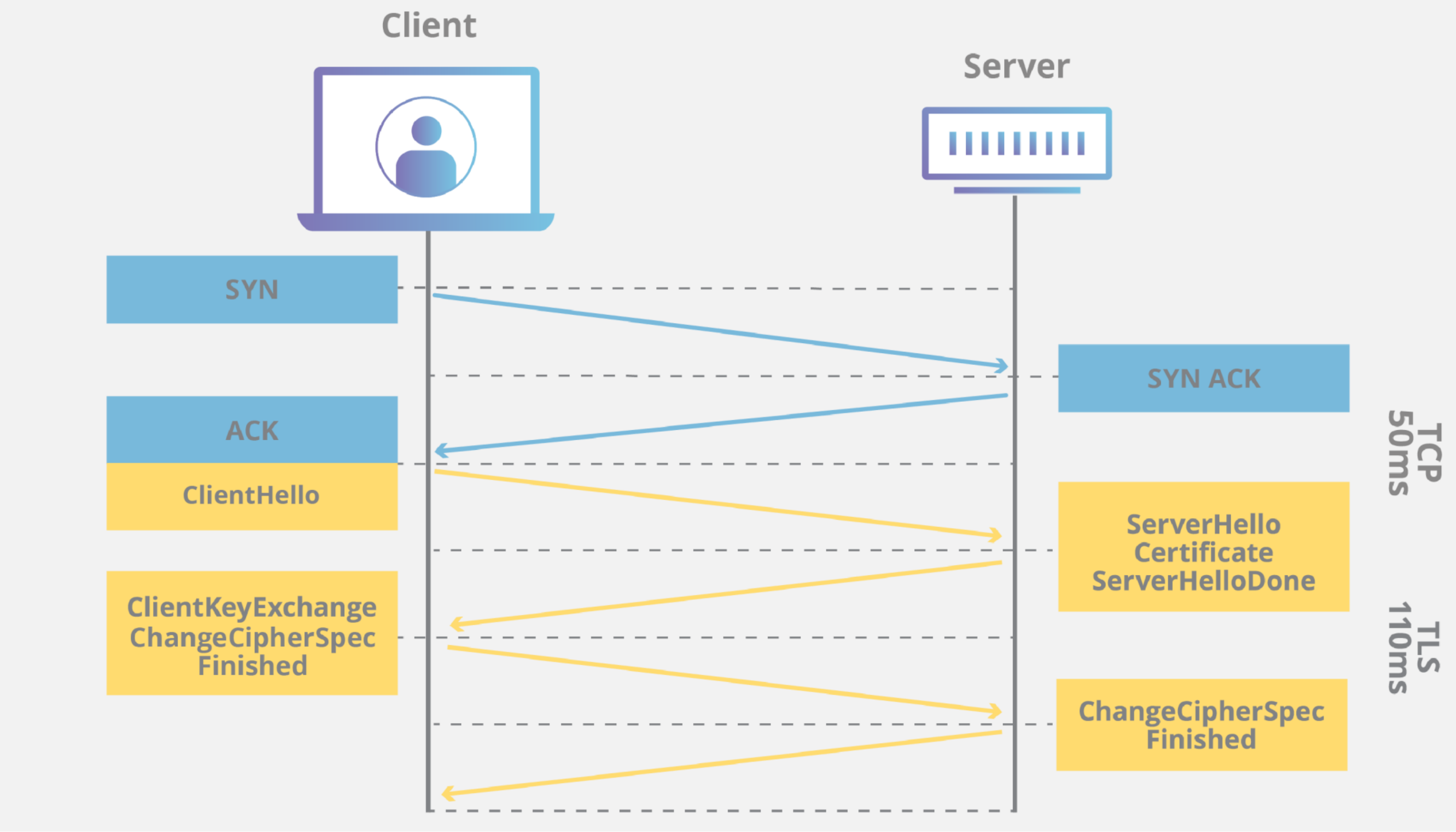
HTTPS/TLS will handle all the encryption, key generation, certificate checking, authentication for you!





UDP does not use TLS

TLS



TLS/SSL does not mandate that two users use a specific symmetric key algorithm

Server will select encryption and hashing algorithm to use

TLS connection can be closed using a TCP FIN

TLS

1. The client sends a list of cryptographic algorithms it supports, along with a client nonce.
2. From the list, the server chooses a symmetric algorithm (for example, AES) and a public key algorithm (for example, RSA with a specific key length), and HMAC algorithm (MD5 or SHA-1) along with the HMAC keys. It sends back to the client its choices, as well as a certificate and a server nonce.

TLS

1. The client sends a list of cryptographic algorithms it supports, along with a client nonce.
2. From the list, the server chooses a symmetric algorithm (for example, AES) and a public key algorithm (for example, RSA with a specific key length), and HMAC algorithm (MD5 or SHA-1) along with the HMAC keys. It sends back to the client its choices, as well as a certificate and a server nonce.
3. The client verifies the certificate, extracts the server’s public key, generates a Pre-Master Secret (PMS), encrypts the PMS with the server’s public key, and sends the encrypted PMS to the server.

TLS

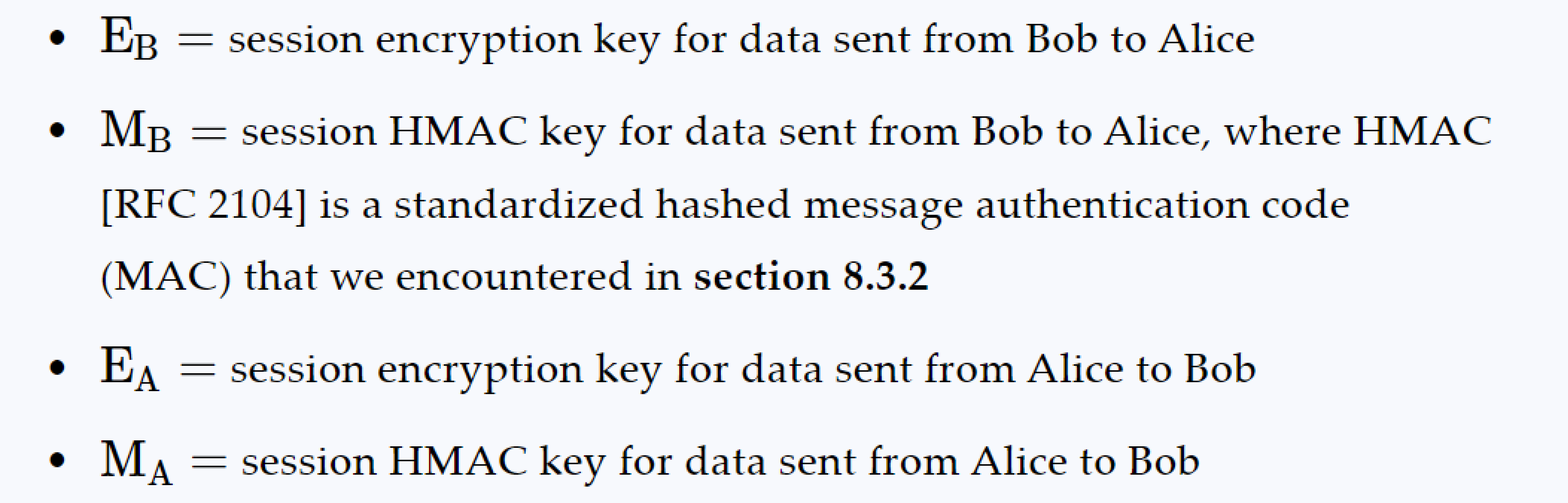
1. The client sends a list of cryptographic algorithms it supports, along with a client nonce.
2. From the list, the server chooses a symmetric algorithm (for example, AES) and a public key algorithm (for example, RSA with a specific key length), and HMAC algorithm (MD5 or SHA-1) along with the HMAC keys. It sends back to the client its choices, as well as a certificate and a server nonce.
3. The client verifies the certificate, extracts the server’s public key, generates a Pre-Master Secret (PMS), encrypts the PMS with the server’s public key, and sends the encrypted PMS to the server.
4. Using the same key derivation function (as specified by the TLS standard), the client and server independently compute the Master Secret (MS) from the PMS and nonces. The MS is then sliced up to generate the two encryption and two HMAC keys. Furthermore, when the chosen symmetric cipher employs CBC (such as 3DES or AES), then two Initialization Vectors (IVs)—one for each side of the connection—are also obtained from the MS. Henceforth, all messages sent between client and server are encrypted and authenticated (with the HMAC).

TLS

1. The client sends a list of cryptographic algorithms it supports, along with a client nonce.
2. From the list, the server chooses a symmetric algorithm (for example, AES) and a public key algorithm (for example, RSA with a specific key length), and HMAC algorithm (MD5 or SHA-1) along with the HMAC keys. It sends back to the client its choices, as well as a certificate and a server nonce.
3. The client verifies the certificate, extracts the server’s public key, generates a Pre-Master Secret (PMS), encrypts the PMS with the server’s public key, and sends the encrypted PMS to the server.
4. Using the same key derivation function (as specified by the TLS standard), the client and server independently compute the Master Secret (MS) from the PMS and nonces. The MS is then sliced up to generate the two encryption and two HMAC keys. Furthermore, when the chosen symmetric cipher employs CBC (such as 3DES or AES), then two Initialization Vectors (IVs)—one for each side of the connection—are also obtained from the MS. Henceforth, all messages sent between client and server are encrypted and authenticated (with the HMAC).
5. The client sends the HMAC of all the handshake messages.
6. The server sends the HMAC of all the handshake messages.

TLS

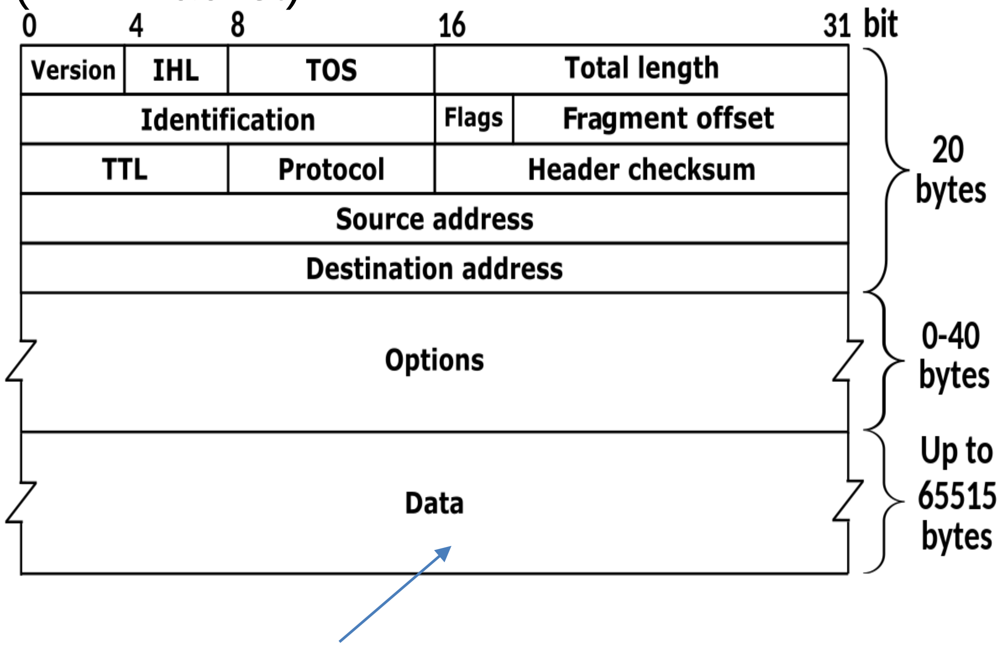
The TLS Master secret, or “session key” consists of four keys



- E_B = session encryption key for data sent from Bob to Alice
- M_B = session HMAC key for data sent from Bob to Alice, where HMAC [RFC 2104] is a standardized hashed message authentication code (MAC) that we encountered in **section 8.3.2**
- E_A = session encryption key for data sent from Alice to Bob
- M_A = session HMAC key for data sent from Alice to Bob

Network-Layer Security

(IPv4 Packet)


0 4 8 16 31 bit

Version | IHL | TOS | Total length

Identification | Flags | Fragment offset

TTL | Protocol | Header checksum

Source address

Destination address

Options

Data

20 bytes

0-40 bytes

Up to 65515 bytes

This could be a **TCP** segment (unencrypted), **TLS** segment, **UDP** segment, **ICMP** packet etc

We have security at the transport layer, but we might also desire security at a network-layer level

The IP security protocol (IPsec) provides data integrity, origin authentication, attack prevention, and confidentiality at the network-layer

IPsec is most commonly seen when using a **Virtual Private Network (VPN)**

Public Network- Anyone can access/communicate with the devices on the network
Private Network- Completely isolated from public internet, typically reserved for a particular institution (this can be costly)

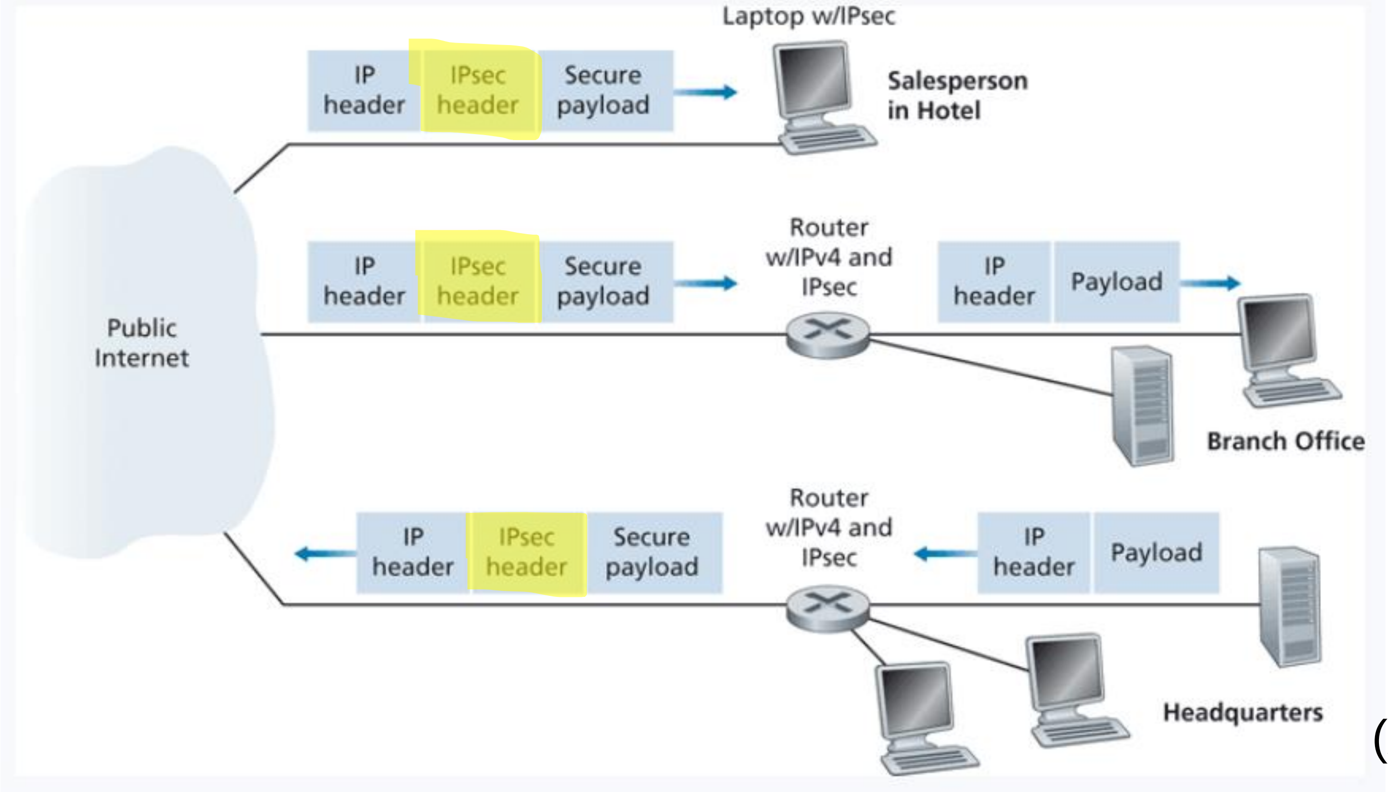
VPNs

IPsec is most commonly seen when using a **Virtual Private Network (VPN)**

Public Network- Anyone can access/communicate with the devices on the network
Private Network- Completely isolated from public internet, typically reserved for a particular institution (this can be costly)

VPNs extend a **private network** over a **public network**

- All messages get encrypted prior to entering any public network (using IPsec), and rerouted through a secure network



Converts vanilla IPv4 datagrams to IPsec datagrams

When a source IPsec entity (host or router) sends secure datagrams to a destination entity, it does so with either the **Authentication Header (AH)** protocol, or the **Encapsulation Security Payload (ESP)** protocol

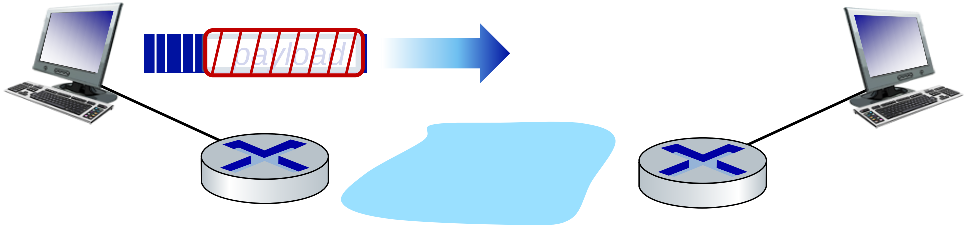
(AH = no confidentiality– not used as much as ESP)

IP Sec

provides datagram-level encryption, authentication, integrity

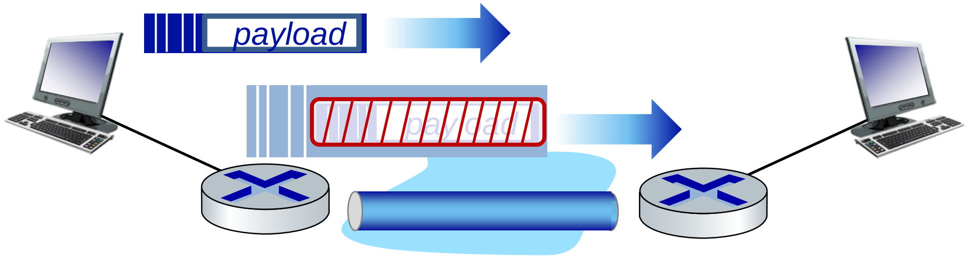
for both user traffic and control traffic (e.g., BGP, DNS messages)

two “modes”:



transport mode:

- *only* datagram *payload* is encrypted, authenticated



tunnel mode:

- entire datagram is encrypted, authenticated
- encrypted datagram encapsulated in new datagram with new IP header, tunneled to destination

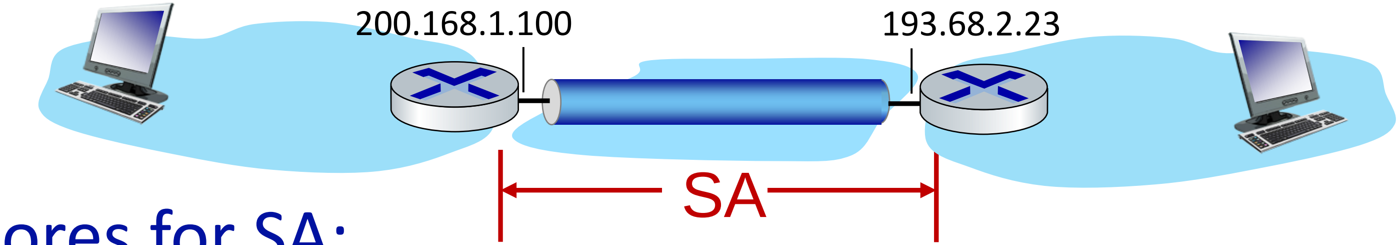
Security associations (SAs)

before sending data, security association (SA) established from sending to receiving entity (directional, simplex)

Sending, receiving entitles maintain *state information* about SA

recall: TCP endpoints also maintain state info

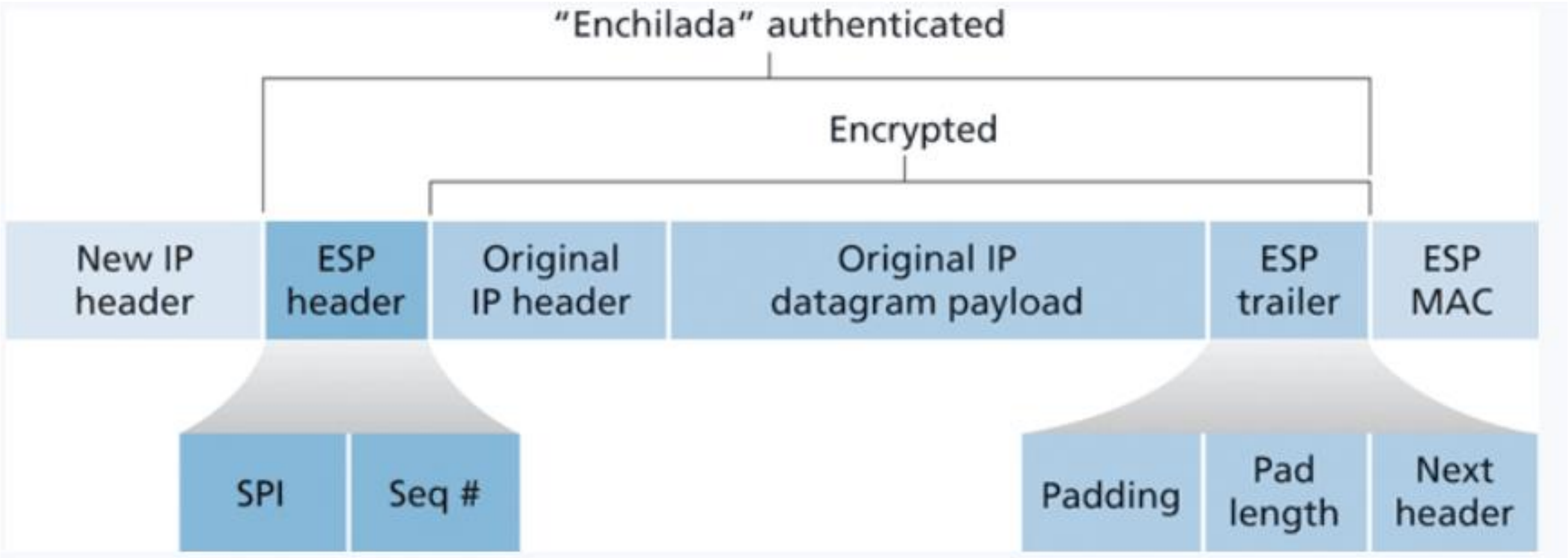
IP is connectionless; IPsec is connection-oriented!



R1 stores for SA:

- 32-bit identifier: *Security Parameter Index (SPI)*
- origin SA interface (200.168.1.100)
- destination SA interface (193.68.2.23)
- type of encryption used
- encryption key
- type of integrity check used
- authentication key

Ipsec Datagram

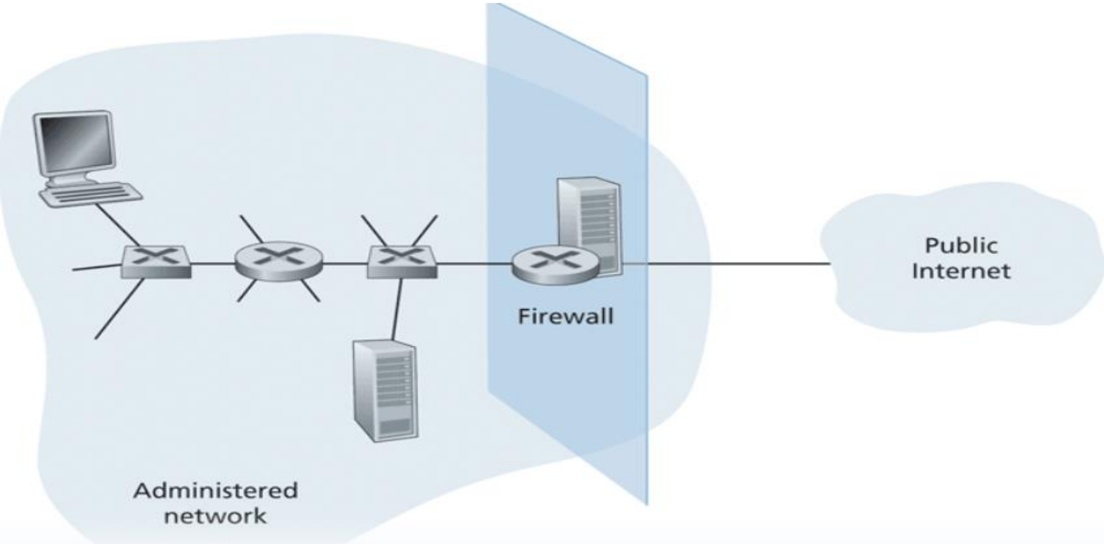


Endpoint Security

A **firewall** is a combination of hardware and software that isolates an organizations internal network form the internet at large, allowing some packets to pass, and blocking others

Three goals

- All traffic from outside to inside, inside to outside, passes through the firewall
- Only authorized traffic (defined by firewall's policy) will be allowed to pass
- The firewall itself is immune to penetration



Types of Firewalls

Packet Filter
→ Analyze packet’s details, and make decision.
→ Look at IPs, Ports, Protocol, TCP Flags, ICMP Type, and more

Example Polices

Policy	Rules
No outside web access.	

Types of Firewalls

Packet Filter
→ Analyze packet’s details, and make decision.
→ Look at IPs, Ports, Protocol, TCP Flags, ICMP Type, and more

Example Polices

Policy	Rules
No outside web access.	Drop all outgoing packets to any IP address, port 80 or 443

Types of Firewalls

Packet Filter
→ Analyze packet’s details, and make decision.
→ Look at IPs, Ports, Protocol, TCP Flags, ICMP Type, and more

Example Polices

Policy	Rules
No outside web access.	Drop all outgoing packets to any IP address, port 80 or 443
No incoming TCP connections	

Types of Firewalls

Packet Filter
→ Analyze packet’s details, and make decision.
→ Look at IPs, Ports, Protocol, TCP Flags, ICMP Type, and more

Example Polices

Policy	Rules
No outside web access.	Drop all outgoing packets to any IP address, port 80 or 443
No incoming TCP connections	Drop all incoming TCP SYN packets

Types of Firewalls

Packet Filter
→ Analyze packet’s details, and make decision.
→ Look at IPs, Ports, Protocol, TCP Flags, ICMP Type, and more

Example Polices

Policy	Rules
No outside web access.	Drop all outgoing packets to any IP address, port 80 or 443
No incoming TCP connections	Drop all incoming TCP SYN packets

(Would this help with SYN flooding?)

Types of Firewalls

Packet Filter
→ Analyze packet's details, and make decision.
→ Look at IPs, Ports, Protocol, TCP Flags, ICMP Type, and more

Example Polices

Policy	Rules
No outside web access.	Drop all outgoing packets to any IP address, port 80 or 443
No incoming TCP connections	Drop all incoming TCP SYN packets
Prevent your network from being tracerouted	

Types of Firewalls

Packet Filter
→ Analyze packet’s details, and make decision.
→ Look at IPs, Ports, Protocol, TCP Flags, ICMP Type, and more

Example Polices

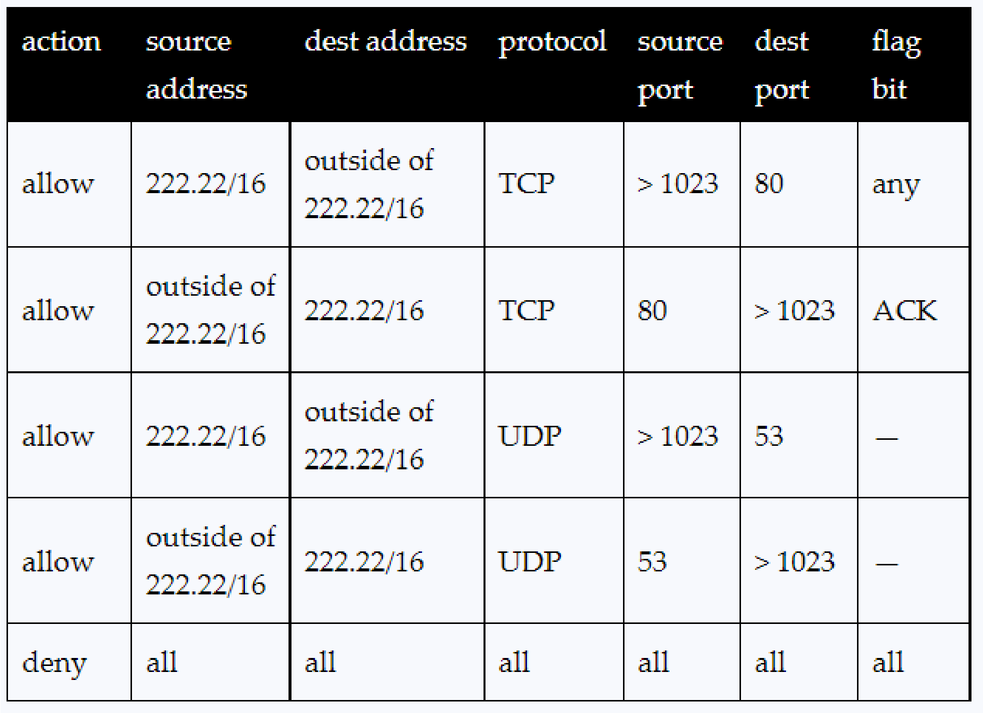
Policy	Rules
No outside web access.	Drop all outgoing packets to any IP address, port 80 or 443
No incoming TCP connections	Drop all incoming TCP SYN packets
Prevent your network from being tracerouted	Drop all incoming/outgoing ICMP traffic

Types of Firewalls

Packet Filter

→ Analyze packet’s details, and make decision.

→ Look at IPs, Ports, Protocol, TCP Flags, ICMP Type, and more



action	source address	dest address	protocol	source port	dest port	flag bit
allow	222.22/16	outside of 222.22/16	TCP	> 1023	80	any
allow	outside of 222.22/16	222.22/16	TCP	80	> 1023	ACK
allow	222.22/16	outside of 222.22/16	UDP	> 1023	53	—
allow	outside of 222.22/16	222.22/16	UDP	53	> 1023	—
deny	all	all	all	all	all	all

Access Control List

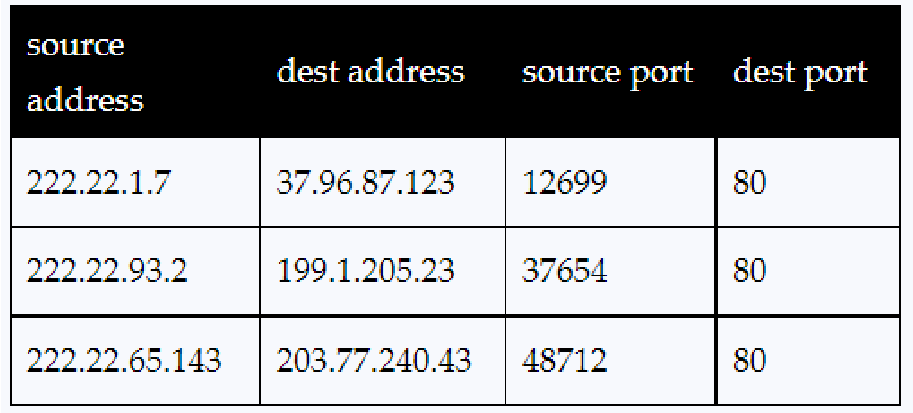
Allow/Deny Traffic

In Linux world, these tables are set using the `iptables` program

Types of Firewalls

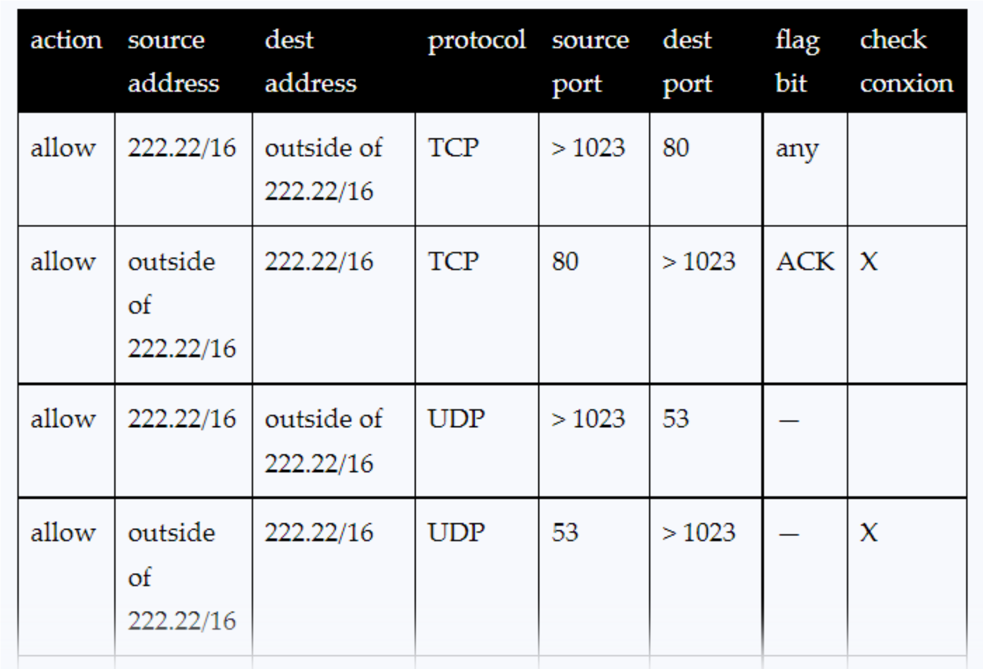
Stateful Filter
→ Make decisions based on **connection information**

Firewall may keep an internal table



source address	dest address	source port	dest port
222.22.1.7	37.96.87.123	12699	80
222.22.93.2	199.1.205.23	37654	80
222.22.65.143	203.77.240.43	48712	80

Access Control List for Stateful Filter



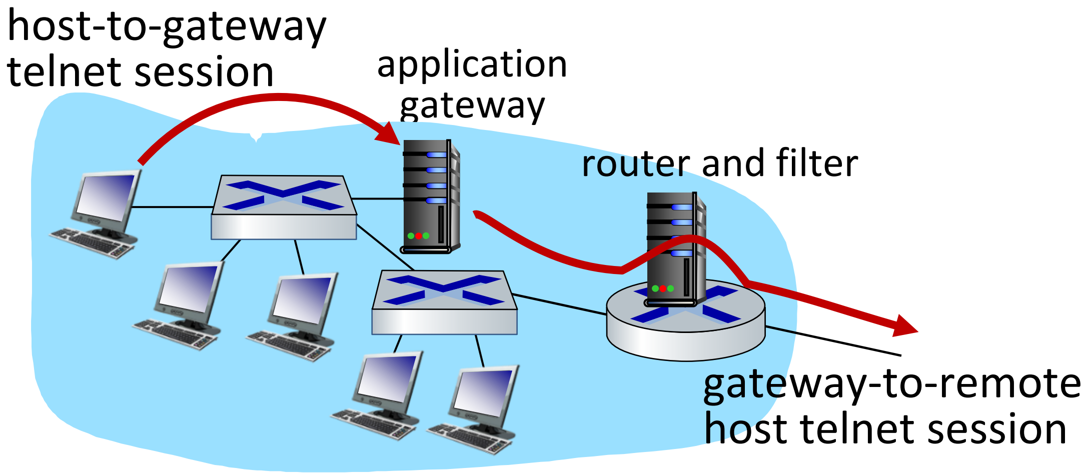
action	source address	dest address	protocol	source port	dest port	flag bit	check conxion
allow	222.22/16	outside of 222.22/16	TCP	> 1023	80	any	
allow	outside of 222.22/16	222.22/16	TCP	80	> 1023	ACK	X
allow	222.22/16	outside of 222.22/16	UDP	> 1023	53	—	
allow	outside of 222.22/16	222.22/16	UDP	53	> 1023	—	X

Example: Random TCP packet with the ACK bit set, even if no TCP connection established

Types of Firewalls

Application gateways

- filter packets on application data as well as on IP/TCP/UDP fields.
- *example:* allow select internal users to telnet outside


host-to-gateway telnet session
application gateway
router and filter
gateway-to-remote host telnet session

Intrusion Detection System

Stateless and Stateful firewalls are examples of **Intrusion Prevention Systems (IPS)**

There will be times where malicious packets will bypass the firewall, so we need another system *within* the network to detect potentially malicious traffic

To detect attacks, we need to perform **deeper** inspection
→ A series of packets
→ The context of a packet
→ Application Data of a Packet

An **instruction detection system (IDS)** inspects packets will generate an alert when potentially malicious traffic is observed

Intrusion Detection System

Stateless and Stateful firewalls are examples of **Intrusion Prevention Systems (IPS)**

There will be times where malicious packets will bypass the firewall, so we need another system *within* the network to detect potentially malicious traffic

To detect attacks, we need to perform **deeper** inspection
→ A series of packets
→ The context of a packet
→ Application Data of a Packet

An **instruction detection system (IDS)** inspects packets will generate an alert when potentially malicious traffic is observed

What might suspicious traffic look like?

Intrusion Detection System

Stateless and Stateful firewalls are examples of **Intrusion Prevention Systems (IPS)**

There will be times where malicious packets will bypass the firewall, so we need another system *within* the network to detect potentially malicious traffic

To detect attacks, we need to perform **deeper** inspection
→ A series of packets
→ The context of a packet
→ Application Data of a Packet

An **instruction detection system (IDS)** inspects packets will generate an alert when potentially malicious traffic is observed

What might suspicious traffic look like?
→ Communication with foreign, unknown IP address
→ Unauthorized web traffic
→ A large spike in traffic

Intrusion Detection System

Two types –

1. Signature-Based Detection Systems

Maintain a large database of **known** “signatures” for malicious packets

-- Malicious IP addresses or URLs
-- Specific String of Bits
-- Protocol Specific (nmap)

-- Email Addresses
-- File/Message Hashes

When would signature-based detection **not work ?**

Intrusion Detection System

Two types –

1. Signature-Based Detection Systems

Maintain a large database of **known** “signatures” for malicious packets

-- Malicious IP addresses or URLs	-- Email Addresses
-- Specific String of Bits	-- File/Message Hashes
-- Protocol Specific (nmap)	

When would signature-based detection **not work ?**

Signature-based detection will never work for **new threats,** so we need a way to dynamically analyze threats

2. Anomaly-based Detection System

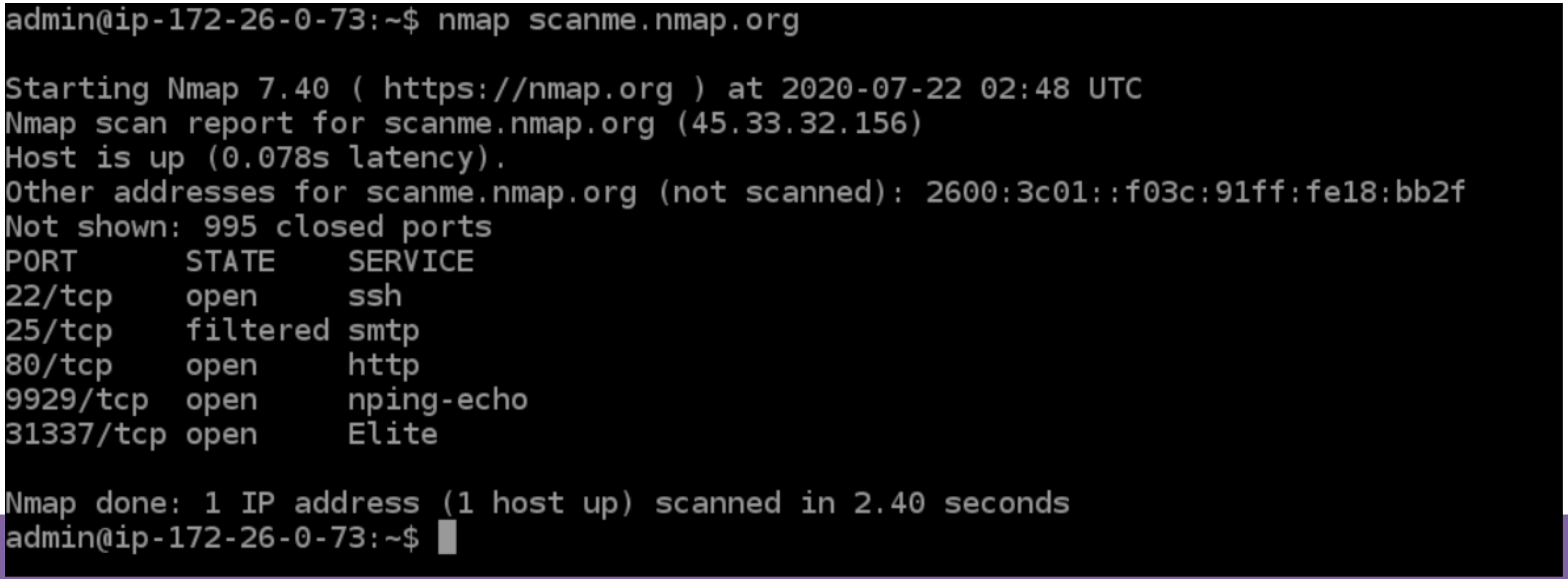
If you know what “normal” traffic looks like, you can identify unusual, potentially malicious traffic
You get a large spike in ICMP packets? Someone might be trying to NMAP you

Nmap

Short for network mapper. It is an open-source Linux command-line tools that is used to scan IP addresses or networks to see which **hosts** are running on their network, discover **open ports** and **services**, and **detect vulnerabilities**

Command issues a bunch of ICMP packets at the target host

Has a lot of great uses to network administrators, and for malicious actors (ie hackers)

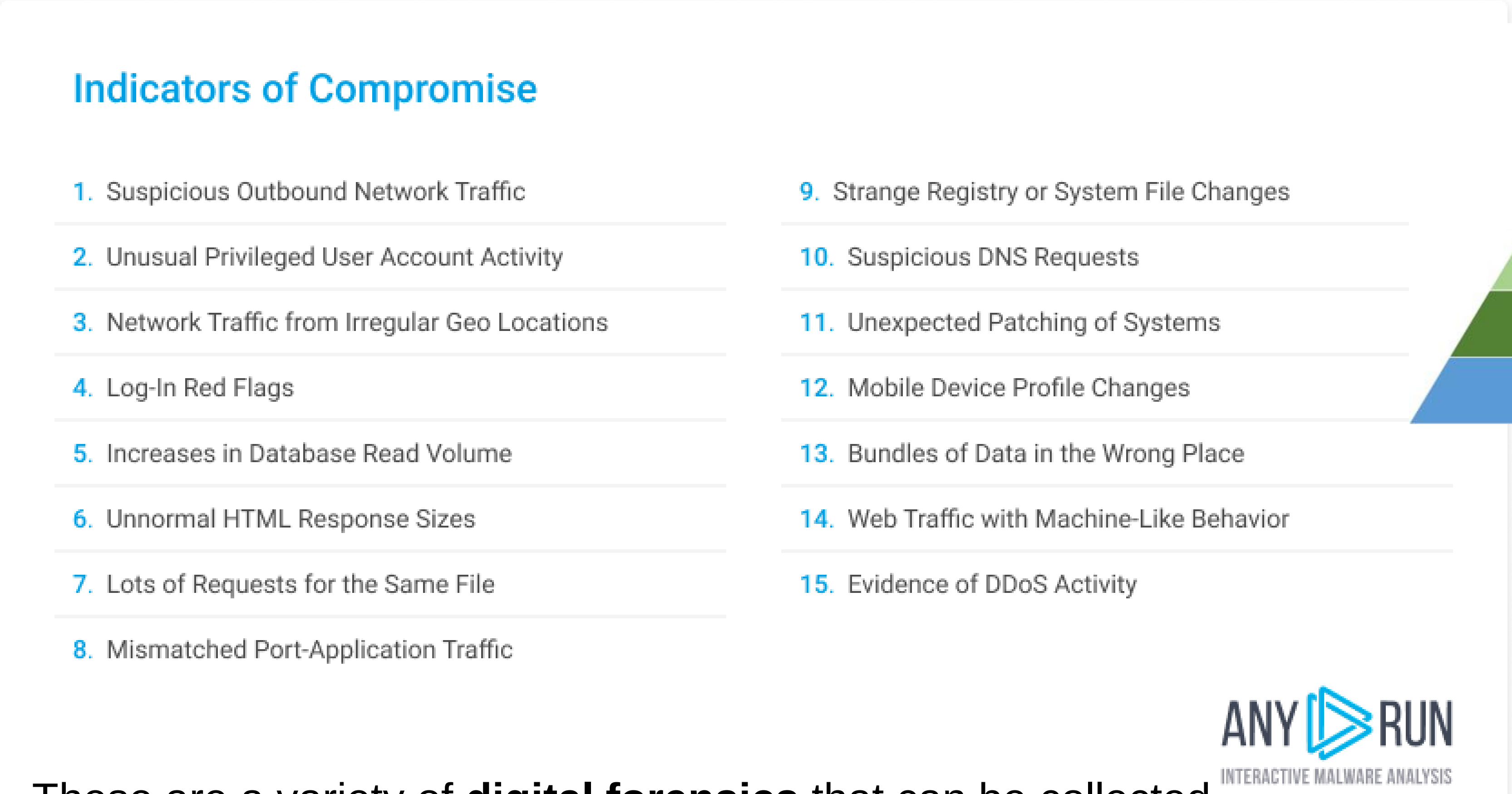


```
admin@ip-172-26-0-73:~$ nmap scanme.nmap.org

Starting Nmap 7.40 ( https://nmap.org ) at 2020-07-22 02:48 UTC
Nmap scan report for scanme.nmap.org (45.33.32.156)
Host is up (0.078s latency).
Other addresses for scanme.nmap.org (not scanned): 2600:3c01::f03c:91ff:fe18:bb2f
Not shown: 995 closed ports
PORT      STATE    SERVICE
22/tcp    open     ssh
25/tcp    filtered smtp
80/tcp    open     http
9929/tcp  open     nping-echo
31337/tcp open     Elite

Nmap done: 1 IP address (1 host up) scanned in 2.40 seconds
admin@ip-172-26-0-73:~$ 
```

Can also return things such as OS versions and application versions

Intrusion Detection System

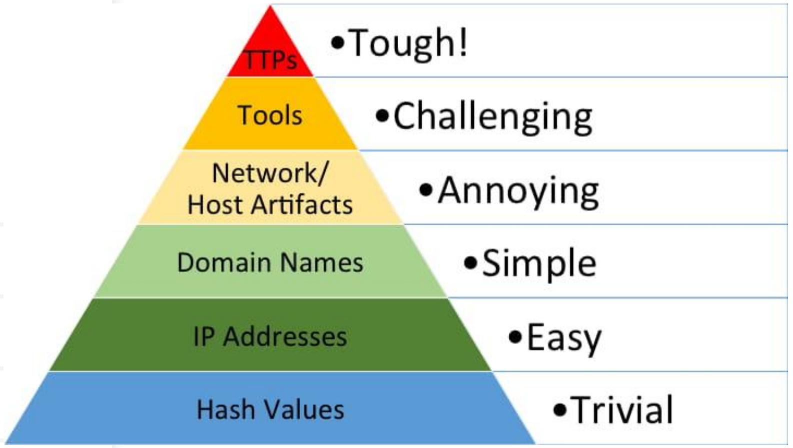
Security/Network Engineers will look for **Indicators of Compromise (IOCs)** to determine if a network may have been breached



Indicators of Compromise

1. Suspicious Outbound Network Traffic
2. Unusual Privileged User Account Activity
3. Network Traffic from Irregular Geo Locations
4. Log-In Red Flags
5. Increases in Database Read Volume
6. Unnormal HTML Response Sizes
7. Lots of Requests for the Same File
8. Mismatched Port-Application Traffic
9. Strange Registry or System File Changes
10. Suspicious DNS Requests
11. Unexpected Patching of Systems
12. Mobile Device Profile Changes
13. Bundles of Data in the Wrong Place
14. Web Traffic with Machine-Like Behavior
15. Evidence of DDoS Activity

ANY RUN
INTERACTIVE MALWARE ANALYSIS



“Pyramid of Pain”

These are a variety of **digital forensics** that can be collected