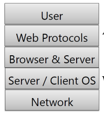
1 Introduction to Computer Security

1.1 Possible Attacks

1. Database/Server/Cloud compromise
2. Insider gains unauthorized access
3. Network sniffing
4. Bugs in web app
5. User impersonation
6. Password breach

1.2 Threat Model



A threat model defines:
- Desired security property / Goal
- Attacker's capability
- Assumption about the setup

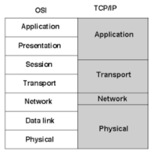
Want to argue from a bottom up perspective

1.3 Basic Principles

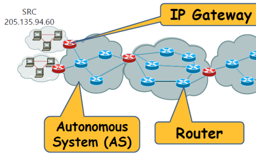
1. Weakest Link Principal
   - Security can be no stronger than its weakest link
   - e.g. Password can be strong and complicated but "recovering" it requires you to answer simple question
2. Kerchkhoff's principal
   - Security by Obscurity is bad

2 Network Attacks

2.1 Network Stack



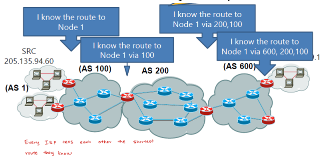
2.2 How the Internet works?



- Built upon a collection of sub-networks
- Each of them sitting on an ISP (**Autonomous System**)
- ISPs can be thought of as "special routers" that runs on **BGP**

2.2.1 BGP (TCP/IP: Physical layer)

- Uses NLRI Update (Network Layer Reachability Information) to propagate information
- Each ISP broadcast to other ISP the shortest path they know to reach a certain node



2.3 IP Protocol

IP information is kept in a IP routing table that looks like this:

Destination	Gateway	Genmask	Flag
0.0.0.0	71.46.14.1	0.0.0.0	UG
10.0.0.0	0.0.0.0	255.0.0.0	U
71.46.14.1	0.0.0.0	255.255.255.255	UH

* Default destination is 0.0.0.0
* Default gateway is the node in a network using the IP protocol that serves as the forwarding host (router) to other networks when no other route matches the destination IP of a packet

2.4 UDP & TCP

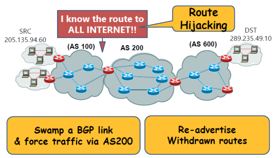
- Unreliable Data delivery over IP: UDP (packets may or may not be received)
- "Reliable" Data Delivery: TCP
  - Connection-oriented, ordered packets

2.5 Network Attacks

2.5.1 Basic terms

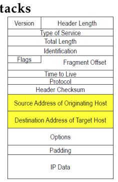
- Eve - passive attacker, can listen to message but cannot modify
- Mallory - malicious attacker, aka MITM, can modify messages, substitute messages, or replay old messages

2.5.2 BGP Attacks



- Attacker impersonate IP gateway and broadcast that it knows the route to the whole internet
- Falsely broadcast that shortest route to all of internet is through a certain ISP to swamp the link
- Flaws of BGP:
  - Lack of security: no built-in mechanisms for authenticating BGP messages or ensuring the integrity of routing information
  - Trust-based model: AS trust information sent by other AS
  - No route validation

2.5.3 IP Attacks



IP Packet Format

- Confidentiality attack:
  - Packet Sniffing
- Integrity attack:
  - IP data pollution
  - Source IP forgery - used for DDoS and anonymous infection (e.g. Slammer worm)

2.5.4 Slammer Worm

- Swamps UDP port 1434 by generating large amounts of packets
- Causes buffer overflow which results in network failure or major slowdown
- Uses randomized SRC IP to conceal attacker's IP as well as to bypass firewalls NIDS

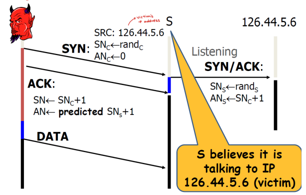
2.5.5 Smurf Attacks

- Attacker send a ICMP (ping) packet with DST field being the broadcast IP and DST being the victims IP (DDoS attack)
- Also called amplify attack
- Takes advantage of other computer in the network to swamp victim with lots of ping requests
- Easy to fix now - just block ICMP channel

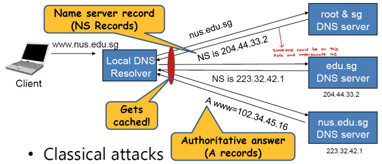
2.5.6 TCP Attacks

Sequence Number Prediction

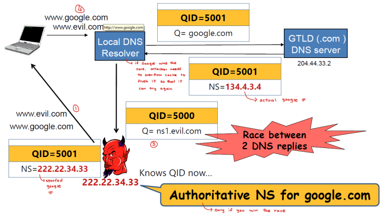
- If TCP uses predictable sequence number for their SYN/ACK e.g. always +k, then an attacker can predict the next sequence number and inject message
- The other party would still assume that it is talking to the correct person since the numbers match up
- Takes advantage of IP authentication which is a weak security protocol
- Solution: randomize the sequence number and also increase the number of bits to make it harder for attacker to guess



2.5.7 DNS Attacks



- Flaw in DNS resolving is that the QID used is predictable
- Possible to carry out DNS Cache poisoning



1. Victim DNS query to resolver for google and some evil website (victim don't even need to click on evil website, could just be an injected JS/img with src of evil website)
2. DNS tries to resolve for evil.com first
3. QID will be known to attacker which he then uses to generate a response for Google.com with QID' = QID + 1
4. If attacker wins the DNS response (could be that the attacker is physically close to the victim) then the DNS will save evil.com NS as Google.com NS
5. Attacker becomes the authoritative NS for Google.com

2.6 Firewalls

Firewalls are tools that control the flow of traffic going between networks.

- Sits at border between networks
- Looks at services, addresses, data etc. of traffic
- Decides whether a packet should be allowed or dropped based on a firewall policy
- Operates at TCP/IP level
- Design principle: Default fail-close policy - Deny on default

2.6.1 Main components of a firewall rule

- Hooks to mount the rule
  - Filtering packets for processes on the firewall computer: INPUT, OUTPUT
  - Filtering packets for other computers connected to the firewall: FORWARD
  - Network address translation: PREROUTING, POSTROUTING
- Conditions
  - IP address, ports, network interface, connection state
- Actions
  - Drop, reject, change packet information

2.6.2 Stateless Packet Filters

- Applies rules to packets in/out of firewall based on SRC/DEST IP address, port, IP protocol, interface
- Does not store any past records and are unable to look at sequence of packets coming in

Action	Src Addr	Dst Addr	Protocol	Src Port	Dst Port	Ctrl-Bit
Allow	1.2.3.*	*	TCP	*	80	*
Allow	*	1.2.3.*	TCP	80	>1023	ACK
Deny	*	*	*	*	*	*

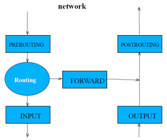
2.6.3 Stateful Packet Filters

- Maintains a state table of all active connections
- Filters packets based on connection states

2.6.4 Proxy-based or Application Firewalls

- Understands application logic
- Acts as a relay of application-level traffic

2.6.5 Netfilter: Linux Firewall



Five hooks to decide the fate of a packet
- Prerouting
- Postrouting
- Forward
- Input
- Output

2.6.6 Threat Model

- Goal:
  - Stop attacker's packet from reaching the end application
- Adversary Capability
  - Adversary can send malicious network packets
  - Adversary is outside the network perimeter
- Assumptions
  - network perimeter is correctly defined
  - firewall is uncompromised (not buggy)
  - firewall sees the same data as end application (need to account for differences in assembly of packets)
  - Defender's policy can tell bad from good traffic by inspecting packet content

Assuming assumptions are true:
- possible to prevent slammer worm by blocking port 1434
- possible to break signature filter by splitting signature across packets
- possible to prevent ICMP attacks by blocking broadcast channel
- impossible to prevent DDoS attack if it is sent by infected host within network
- threats coming from legitimate ports requires additional checks of content and signatures

2.6.7 Weaknesses of Threat Model

- Defender needs to know all possible attacks and how to block them for firewalls to be effective - easy for attackers to evade those attack patterns
- Enforcement is very complex
- Continuous arms race, e.g. evolving signatures
- Easy to violate assumptions
  - Physically Compromise the firewall
  - Difficult to ascertain which service is targeted from inspecting network flows only
  - Firewall network code may differ from host OS / app code
  - Many attack (raw byte) patterns for the same exploit!
- Thwarted completely by encrypted traffic (e.g. HTTPS)
- "Bring your own device" problem: Carrying data on a smartphone from outside the network

3 Secure Channel

A Secure Channel is a data communication protocol established between 2 programs which preserves data in terms of
- Confidentiality
- Integrity
- Authentication/Authenticity (person we are talking to is who they say they are)

* Availability is not a goal!! DDoS attacks permitted by threat model

3.1 Basic Cryptographic Primitives

- Encryption **only provides Confidentiality** promise and does not guarantee Integrity and Authenticity
- MAC & Digital Signature **provides both integrity and authenticity** guarantees but does not guarantee confidentiality
- Example of Secure Channels
  - HTTPS, Encrypted File System, SSH/VPN

3.2 Symmetric Key Encryption

Assume that Alice is communicating with Bob and a malicious attacker A is in between them that is capable of eavesdropping and modifying packets. To prevent attacker, we have 2 options:
- Assume A has poorer network than Alice/Bob and receive less data than Alice/Bob
- Assume Alice and Bob have some pre-shared secret key

3.2.1 How it works

1. Alice and Bob first preshare a secret key K which is randomly generated
2. Encryption: $M \times K \rightarrow C$
3. Decryption: $C \times K \rightarrow M$

Must ensure that it follows the correctness property: $\forall m, k, Dec(Enc(m,k)) = m$

3.2.2 Adversary's Knowledge

- Algorithms Setup, Enc, Dec are public, but internal coin flips are private (makes generation of key probabilistic)
- Adversary knows any distribution over M from background knowledge but defender does not

3.2.3 Chosen Plaintext Attack

- Attacker has access to encryption oracle
- Attacker can repeatedly try sending plaintext over to oracle and check whether the encrypted text matches

3.2.4 Security

Want to achieve perfect secrecy: $Pr[Guess = m|c] = Pr[Guess = m]$, i.e. even if attacker know ciphertext, it is just as good as guessing without any prior knowledge

3.2.5 Caesar Cipher

- Idea is that the key is the number of times to rotate the characters by
- C = (P + X) mod 26 where X is the number of rotations
- Easily broken once we know the (Setup, Enc, Dec) operations
- Deterministic functions, easily subject to frequency analysis

3.2.6 One time pad

- Enc: $c := m \oplus k$, k is the randomly chosen secret key
- Dec: $m = c \oplus k$
- k changes for each plaintext message
- Achieves correctness and perfect secrecy
- $|K| \geq |M|$, requires a huge key space → if can securely transfer K then can also securely transfer M
- Relies on bijection of M → C, if not there will be some c then shows up more frequently and will be susceptible to dictionary attacks
- \# of functions that can be defined from n-bits to m-bits = $2^{m \cdot 2^n}$

3.3 Integrity

In a normal communication setup, Bob would not be able to distinguish data sent by Alice vs Attacker. Checksums are useless too since they are keyless and is only used for error correction

3.3.1 Message Authentication Codes (MAC)

Provides sender Authenticity and integrity of messages sent.

How it works

1. Sender generates a tag: $S(k, m)$ which basically "signs" the message with a secret key that is pre-shared
2. Sender sends message m with the tag
3. Receiver then verifies $V(k, m, tag)$ which detects whether any changes has been made to the message and at the same time verifies that it is the correct key (and hence is the authentic sender)

3.3.2 Formal Security Goal for MAC

- Want to prevent existential forgery under Chosen Message Attack (CMA)
  - Given m, $Pr[V(k,m,t) \rightarrow yes] \leq negl$, where t is a tag guessed by Attacker

3.3.3 Perfectly Secure MAC

$$(a \cdot m + b) mod p$$

where m, p is known publicly and (a, b) is the secret key that is chosen randomly
- Works on universal hash family
- $Pr[S_{a,b}(m) = t \wedge S_{a,b}(m') = t'] = \frac{1}{|T|^2}$
- Key space $|K|$ for perfectly secure MAC is $2n$ for n bit MAC to make it such that attacker has negligible chance of success

3.4 Takeaways

- Symmetric Key Constructions are feasible but having perfect secrecy and perfect MACs take up way too much space and are impractical

3.5 Relaxed Assumption of Attacker

- Assumes that adversary has limited computation power
- Adversary can execute polynomial # of steps
- Has randomized and non-deterministic execution
- Bounds queries in CPA/CMA to polynomial in |K| i.e. if key bits is 128, adversary can't compute all 2^{128} computation

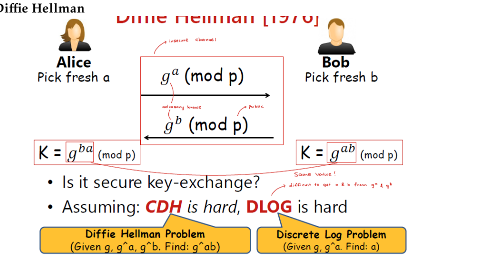
3.6 Public Key Cryptography

Instead of having a pre-shared secret key, both Alice and Bob now have a pair of public and private key
- e.g. Alice wants to send Bob a message, Alive will encrypt the message using Bob's public key and Bob will decrypt using his own private key
- Can also be used to do signature by signing message with private key and receiver can verify it using the public key
- Works under the assumptions:
  - Difficulty of Factoring product of large primes - circumvented by quantum computing
  - Discrete Logarithm in Groups is computationally difficult
  - Problems in Lattices - withstand quantum computing adversary

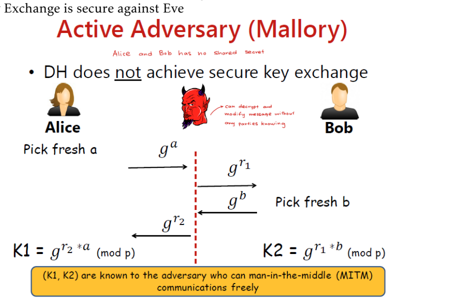
3.6.1 Key Exchange Protocol

- Used to establish fresh shared secrets per session
- Session keys is a means to maintain (Perfect) forward secrecy → protect encrypted information even if long term key is compromised
- Based on computational hardness assumption of Discrete Log (DLOG)
  - For an appropriately chosen group G (e.g. Integer mod prime), where g is the generator. Given $A \in G$, difficult to find any $a \in G$ such that $g^a = A$ (for any "classical" computer)
  - Can be interpreted as how many times do we run the generator to get A

3.6.2 Diffie Hellman



- Eve sees g^a, g^b
- Under Computational Diffie-Hellman (CDH), it is computationally hard to compute g^{ab}
- Under Decisional Diffie-Hellman (DDH), g^{ab} looks like a random element from G
- DH Key Exchange is secure against Eve

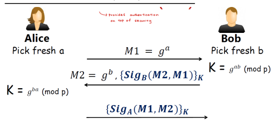


1. MITM can intercept g^a sent by Alice and send g^{r_1} to Bob
2. Bob will then send back g^b which is also intercepted by MITM and sends Alice back g^{r_2}
3. Alice now has $g^{r_2 * a}$ (mod p) and Bob has $g^{r_1 * b}$ (mod p) and MITM has both the r_1 and r_2

3.6.3 Authenticated Key Exchange

- Authentication key exchange protocol guarantees
  - Entity Authentication: Entities are who they claim to be
  - Good Key: Only the intended parties derive shared new secret

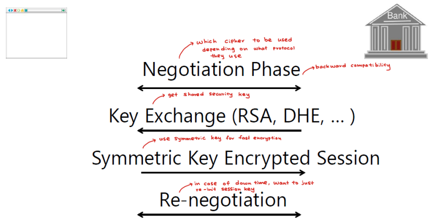
Station-to-station (STS) Protocol



- STS provides both authentication and promises
- Secure against both Eve and Mallory

3.7 HTTPS
- HTTP + Secure Socket Layer (SSL) = HTTPS
- Modern SSL is known as TLS and is continually revised to prevent more threats

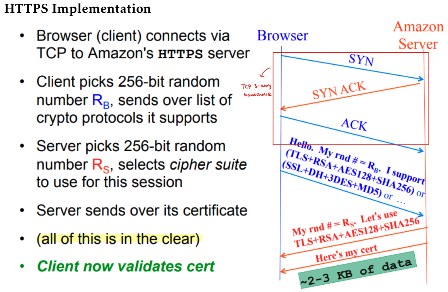
3.7.1 High Level Implementation of SSL/TLS



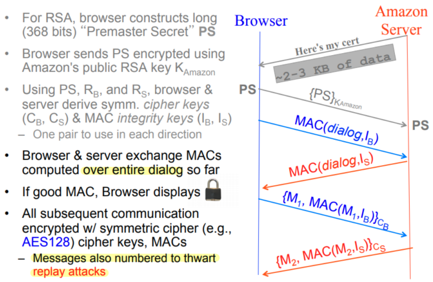
- Negotiation phase: what cipher is used that is compatible to both server and client, provides backward compatibility
- Key Exchange using RSA, DHE etc., session key is generated here
- Symmetric Key is used to encrypt the entire session → Symmetric key used instead of public-private key since it provides much faster encryption, private-public key requires a lot of bits to be safe

3.7.2 Certificates
- Guarantees Integrity and Authenticity
- Similar to Digital Signatures but for Web
- Certificates are signed by Certificate Authorities (CA)
  – Root CA's public keys are hard coded into OS
  – Chain of Trust stops at root CA
- Root CAs can designate intermediate CAs (restricted to signing own subdomain)
- Based on trust system where we completely trust that root CAs are not malicious

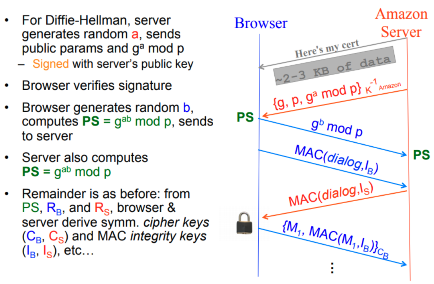
3.7.3 HTTPS Implementation



RSA Implementation
- Does not have forward secrecy guarantee, once attacker gets PS, whole encrypted channel is exposed



Diffie-Hellman Implementation



3.7.4 Assumption of Threat Model
- User is using a secure channel
- Crypto primitives are secure (uses RSA, HMAC etc.)
- TLS protocol design is secure
- TLS protocol implementation is secure
- Certificate issuers are uncompromised
- Users check browser UI correctly
- Alice & Bob's secrets are secure
- Entities are authenticated correctly

3.7.5 Capabilities of HTTPS
- Assuming that we reach https://gmail.com and it shows no cert errors, we can safely assume that it is safe from ALL network attacks
  – Safe from DNS cache poisoning (will not show padlock since they don't have server's certificate)
  – Safe from BGP route hijacking (again certificates won't match)
  – Safe from TCP/IP attacks (data streams are encrypted and attacker can't see seq #)
- Everything of an URL except IPs and ports are encrypted when using HTTPS

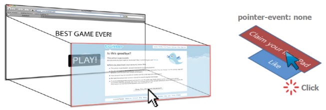
3.7.6 Downfalls of HTTPS
Typical ways attackers can "win" by going outside of the threat model:
- Attack assumptions
- Violate other security properties that are not captured by threat model e.g. attack availability instead

HTTP Downgrade Attack
- Mallory is on the network and downgrades traffic from HTTPS to HTTP
- Possible to downgrade other sub-resources on page to HTTP (e.g. Script tags, img src, iFrames etc.)
- Assume that as long as HTTP then IT IS NOT SAFE
- Can be prevented by using HTTP Strict Transport Security (HSTS) which is a header that states that your website only serves HTTPS sites

Insecure Cookies
- Possible to use an img src to a HTTP site that will leak cookies to HTTP traffic
- Prevented by setting "secure" flag for cookies which tells browser not to send cookies over HTTP
- Cookies are only sent securely through HTTPS and 'Secure' Keyword but can be read by JS via DOM API (if JS is injected then everything will fail)
- Cookies could be overriden by HTTP request (`Set-cookie: SID=bad;secure`)
- Cookies could be overriden /deleted by JS: evil.example.com can set cookies for example.com

UI Confusion
- bankofthewest.com VS bankofthevvest.com
- null byte certificate e.g. gmail.com\0.evil.com will be read as gmail.com from browser perspective
- pаypal.com shows up as accented 'a' in browser which is almost indistinguishable from normal 'a'
- In general note that everything below the address bar is unreliable!

Clickjacking



- Possible to embed iframes into pages that can host any site
- Frames of iframe can overlap and can even be made transparent / makes clicks fall through it by using CSS

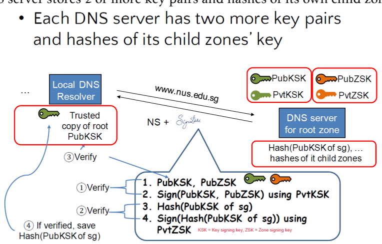
Compromised Cert / CA
- How to detect if being served bad cert
  – Certificate pinning (used in SSH, "trust on first use", cache certificates after first visit)
  – Certificate Revocation (allows CA to remove certs that are found malicious)
  – Certificate Transparency (allows people to check CA's cert and store it in a Public Cert. Log)

Side Channel Attacks
- Size of data, Data access patterns (fixed using deterministic address pattern or randomization), power channel, sound, electromagnetic radiation
- Timing Channel
  – exploit variations in execution time of a program or system based on different inputs or conditions
  – solved by ensuring that computation time does not differ much between branches

Broken Crypto Primitive
- Using MD5 which is prone to collisions
- Using encrypt-and-MAC instead of MAC-then-encrypt (SSL, could be insecure by using padding oracle) or encrypt-then-MAC (IPSec, provably secure)

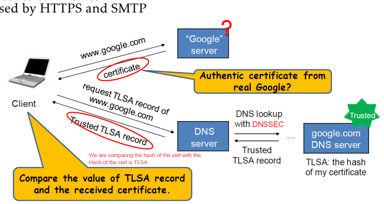
4 Extra Content

4.1 DNSSEC
- DNS is unsafe since we do not know whether the reply is sent from the authentic DNS server and we also do not know whether the final A record is tampered
- DNSSEC prevents this by adding on signatures to each DNS reply
- Each DNS server stores 2 or more key pairs and hashes of its own child zone keys



- Purpose: Ensures that replies from DNS servers are authentic and that the final A records are not tampered with
- Attacks prevented: MITM and DNS cache poisoning and QID prediction attack
- Assumptions: DNS servers are not compromised and that the root DNS KSK must be trusted

4.2 DANE
- Resolves issue of TLS having heavy reliance on the chain of trust → possible that CAs wrongly issues certificates
- Can be used by HTTPS and SMTP



- Uses DNSSEC to check authenticity of certificates
- Prevents impersonation even if CAs wrongly issues certificates

4.3 SPF
- Specifies the servers and domains that are authorized to send email on behalf of your organization
- SPF records are basically DNS TXT records that acts as a whitelist for authorized servers
- Receiver can just verify that the sender is indeed authorized to send by checking the IP address of SPF
- Prevents email spoofing
- Attacker can only bypass SPF by gaining access to IP address of email server which is hard in practice

4.4 DKIM
- Adds a digital signature to every outgoing message, which lets receiving servers verify the message actually came from your organization
- Domain owner add a DKIM key in DKIM TXT record
- Sender is expected to sign the email
- Receiver will add Authentication-Resultsheader (DKIM=pass/OKindicates verified mail)
- Another way to prevent email spoofing attacks by adding a layer of authenticity to the email messages through signatures
- Assumes that Key management and TTLs are in place

4.5 DMARC
- Additional notification protocol for messages that do not pass DKIM or SPF.
- DMARC records will store an email address which will receives reports if any of the above 2 protocols fails

4.6 PGP
- Web of trust —decentralized, accumulate and distribute public keys