04832250 – Computer Networks (Honor Track)

A Data Communication and Device Networking Perspective

Module 6: Network Security

Prof. Chenren Xu（许辰人）

Center for Energy-efficient Computing and Applications

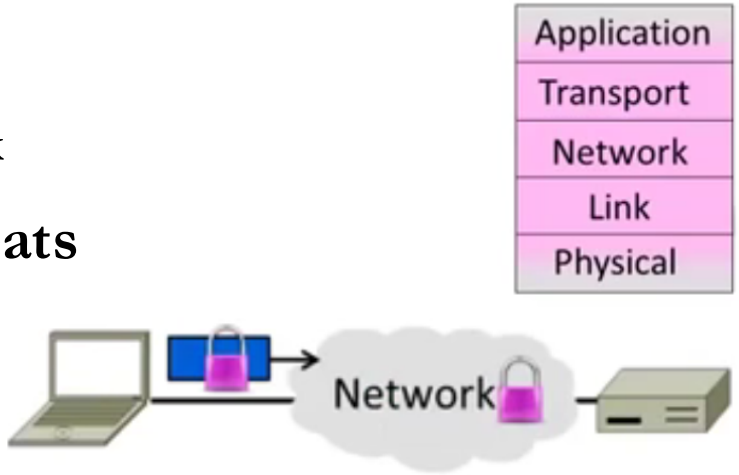
Computer Science, Peking University

chenren@pku.edu.cn

http://soar.pku.edu.cn/

Overview

- Revisiting the layers
  - Network security affects all layers because each layer may pose a risk
- Network security designs to protest against a variety of threats
  - Often build on cryptography
  - Just a brief overview. Take a course!
    - MIT 6.857 Computer and Network Security
      - http://courses.csail.mit.edu/6.857/
    - MIT 6.858 Computer Systems Security
      - http://css.csail.mit.edu/6.858/
    - CMU 14829 Mobile Embedded and Wireless Security
      - http://mews.sv.cmu.edu/teaching/14829/



Topics

- Threat models
- Crypto
  - Confidentiality
  - Authentication
- Applied crypto
  - Wireless security (802.11)
  - Web security
  - DNS security
- Connectivity
  - Firewalls
  - Distributed denial-of-service

Security Threats

- "Security" is like "performance"
  - Means many things to many people
  - Must define the properties we want
- Key part of network security is clearly stating the threat model
  - The dangers and attacker's abilities
  - Can't assess risk otherwise
- Some example threats
  - It's not all about encrypting messages

Attacker	Ability	Threat
Eavesdropper	Intercept messages	Read contents of message
Intruder	Compromised host	Tamper with contents of message
Impersonator	Remote social engineering	Trick party into giving information
Extortionist	Remote / botnet	Disrupt network services

Risk Management

- Security is hard as a negative goal
  - Try to ensure security properties that don't let anything bad happen!
- Only as secure as the weakest link
  - Could be design flaw or bug in code
  - But often the weak link is elsewhere

Design flaws
Implementation bug
Weak password, etc.

- 802.11 security … early on, WEP (Wired Equivalent Privacy):
  - Cryptography was flawed: session key is too short; can run cracking software to read WiFi traffic in a few minutes
    - Borisov, Nikita, et al., "Intercepting mobile communications: the insecurity of 802.11." ACM MobiCom, 2001
- Today, WPA2/802.11i security:
  - Computationally infeasible to break!
- So that means 802.11 is secure against eavesdropping?
  - Many possible threats

Thread Model	Old WiFi (WEP)	New WiFi (WPA2)
Break encryption from outside	Very easy	Very diffucult
Guess WiFi password	Often possible	Often possible
Get password from computer	May be possible	May be possible

  - 802.11 is more secure against eavesdropping in that the risk of successful attack is lower. But it is not "secure".

Cryptology

- Rich history, especially spies / military
  - From the Greek "hidden writing"
- Cryptography
  - Focus is encrypting information
- Cryptanalysis
  - Focus is how to break codes
  - Modern emphasis is on codes that are "computationally infeasible" to break
- Uses of Cryptography
  - Encrypting information is useful for more than deterring eavesdroppers
    - Prove message came from real sender
    - Prove remote party is who they say
    - Prove message hasn't been altered
  - Designing a secure cryptographic scheme is full of pitfalls!
    - Use approved design in approved way

Internet Reality

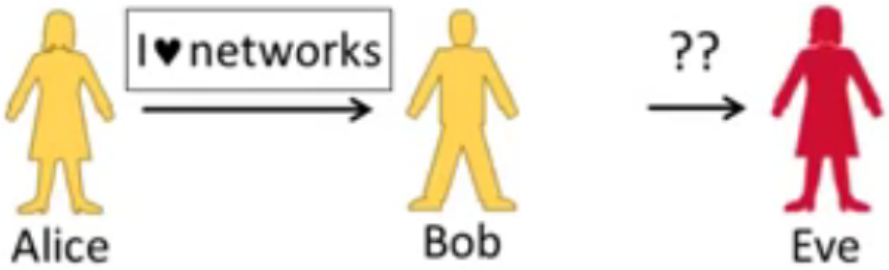
- Most of the protocols were developed before the Internet grew popular
  - It was a smaller, more trusted world
  - So protocols lacked security …
- We have strong security needs today
  - Clients talk with unverified servers
  - Servers talk with anonymous clients
  - Security has been retrofitted
  - This is far from ideal!

Topics

- Threat models
- Crypto
  - Confidentiality
  - Authentication
- Applied crypto
  - Wireless security (802.11)
  - Web security
  - DNS security
- Connectivity
  - Firewalls
  - Distributed denial-of-service (DDoS)

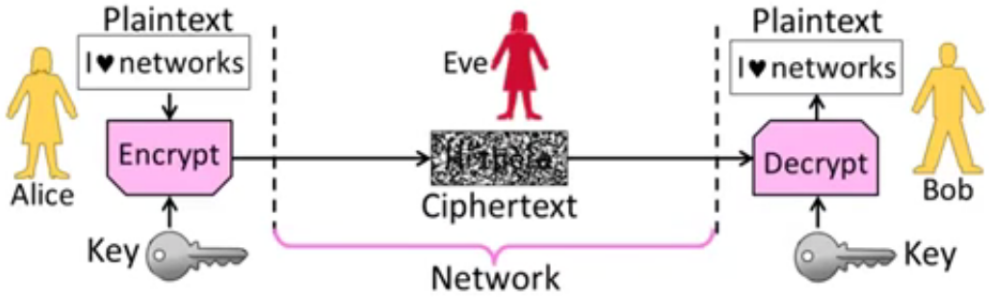
Confidentiality

- Encrypting information to provide confidentiality
  - Symmetric and public key encryption
  - Treat crypto functions as black boxes
- Goal and Threat Model
  - Goal is to send a private message from Alice to Bob
    - This is called confidentiality
  - Threat is Eve will read the message
    - Eve is a passive adversary (observes)



Encryption/Decryption Model

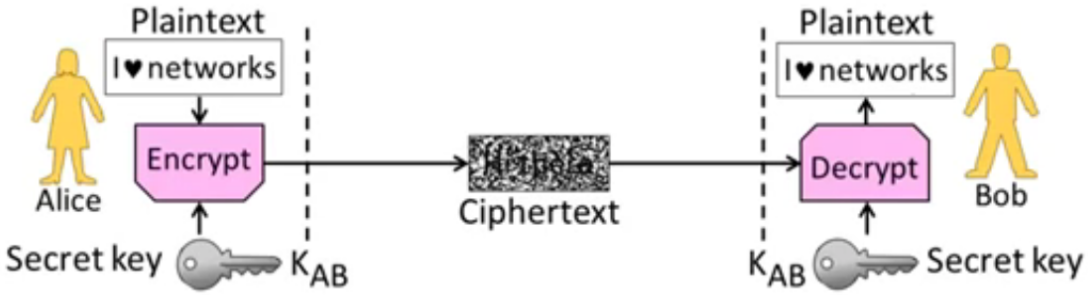
- Alice encrypts private message (plaintext, or P) using key
- Eve sees ciphertext C but can't relate it to private message
  - $C = E_K(P)$
- Bob decrypts using key to obtain the private message
  - $D_K(C) = D_K(E_K(P)) = P$
- Encryption is a reversible mapping
  - Ciphertext is confused plaintext
- Assume attacker knows algorithm
  - Security does not rely on its secrecy
  - Also known as "Kerckhoff's principle"
    - All algorithms must be public: only the keys are secret
- Algorithm is parameterized by keys
  - Security does rely on key secrecy
  - Must be distributed (Achilles' heel)



- Two main kinds of encryption
  - Symmetric key encryption, e.g., AES
    - Alice and Bob share secret key
    - Encryption is a bit mangling box
  - Public/asymmetric key encryption, e.g., RSA
    - Alice and Bob each have a key in two parts: a public part (widely known), and a private part (only owner knows)
    - Encryption is bases on mathematics (e.g., RSA is based on difficulty of factoring)

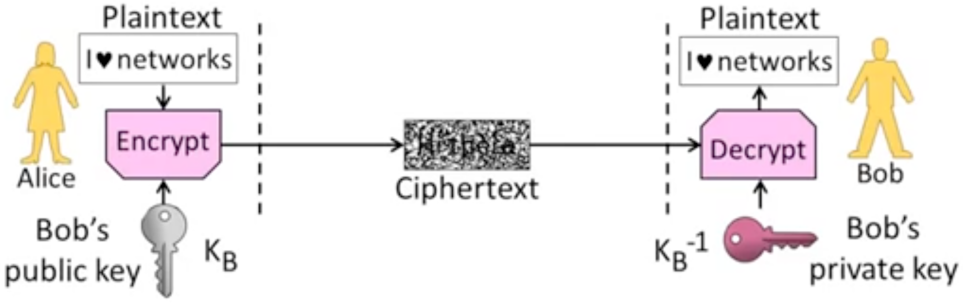
Symmetric (Secret Key) Encryption

- Alice and Bob have the same secret key, K_{AB}
  - Anyone with the secret key can encrypt/decrypt
- Example: AES (Advanced Encryption Standard)
  - Bake-off rules:
    - The algorithm must be a symmetric block cipher
    - The full design must be public
    - Key lengths of 128, 192, 256 bits must be supported
    - Both software and hardware implementations must be possible.
    - The algorithm must be public or licensed on nondiscriminatory terms.
  - Rijndael has become the world's dominant cryptographic cipher



Public Key (Asymmetric) Encryption

- Alice and Bob each have public/private key pair (K_B / K_B^{-1})
  - Public keys are well-known, private keys are secret to owner
- Alice encrypts with Bob's public key K_B; anyone can send
- Bob decrypts with his private key K_B^{-1}; only he can do so
- Example: RSA (Rivest, Shamir, and Adleman)



The RSA method is based on some principles from number theory. We will now summarize how to use the method; for details, consult the paper.

1. Choose two large primes, p and q (typically 1024 bits).
2. Compute $n = p \times q$ and $z = (p - 1) \times (q - 1)$.
3. Choose a number relatively prime to z and call it d.
4. Find e such that $e \times d = 1 \ mod \ z$.

With these parameters computed in advance, we are ready to begin encryption. Divide the plaintext (regarded as a bit string) into blocks, so that each plaintext message, P, falls in the interval $0 \le P < n$. Do that by grouping the plaintext into blocks of k bits, where k is the largest integer for which $2^k < n$ is true.

To encrypt a message, P, compute $C = P^e \pmod{n}$. To decrypt C, compute $P = C^d \pmod{n}$. It can be proven that for all P in the specified range, the encryption and decryption functions are inverses. To perform the encryption, you need e and n. To perform the decryption, you need d and n. Therefore, the public key consists of the pair (e, n) and the private key consists of (d, n).

- Factoring large numbers is hard!
- It takes 1016 years to factor a 500-digit number with a million chips running in parallel, each with an instruction time of 1 ns

Discussion

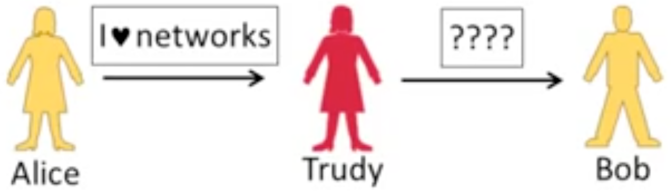
- Key Distribution
  - This is a big problem on a network!
    - Often want to talk to new parties
  - Symmetric encryption problematic
    - Have to first set up shared secret
  - Public key idea has own difficulties
    - Need trusted directory service
    - We'll look at certificates later
- Symmetric vs. Public Key
  - Have complementary properties
    - Want the best of both!

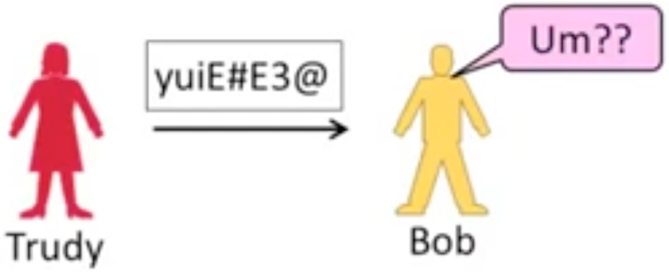
Property	Symmetric	Public key
Key Distribution	Hard – share serect per pair of users	Easier – publish public key per user
Runtime Performance	Fast – good for high data rate	Slow – few, small messages

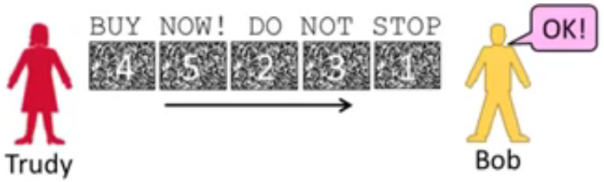
- Winning Combination
  - Alice uses public key encryption to send Bob a small private message
    - It's a key! (Say 256 bits.)
  - Alice and Bob send large messages with symmetric encryption
    - Using the key they now share
  - The key is called a session key
    - Generated for short-term use

Authentication

- Encrypting information to provide authenticity (=correct sender) and integrity (=unaltered)
  - Confidentiality isn't enough
- Goal and Threat Model
  - Goal is to let Bob verify the message came from Alice and is unchanged
    - This is called integrity/authenticity
  - Threat is Trudy will tamper with messages
    - Trudy is an active adversary (interferes)
- Why encryption is not enough?
  - What will happen if Trudy flips some of Alice's message bits?
    - Bob will decrypt it, and will receive an altered message
  - Typically encrypt blocks of data
  - What if Trudy reorders message?
    - Bob will receive altered message
    - "Stop, don't buy it now" -> …



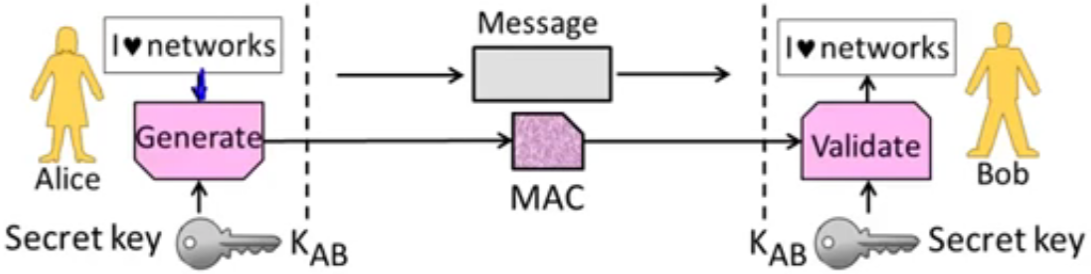




MAC (Message Authentication Code)

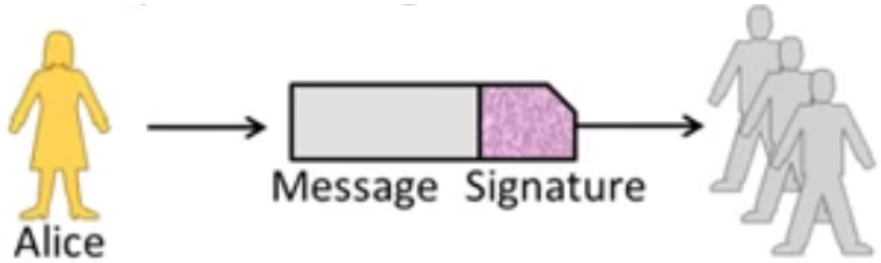
- MAC is a small token to validate the integrity/authenticity of a message
  - Send the MAC along with message
  - Validate MAC, process the message
  - Example: HMAC (Hash-based MAC) scheme

- MAC of symmetric encryption operation – key is shared
  - Lets Bob validate unaltered message came from Alice
  - Doesn't let Bob convince Charlie that Alice sent the message

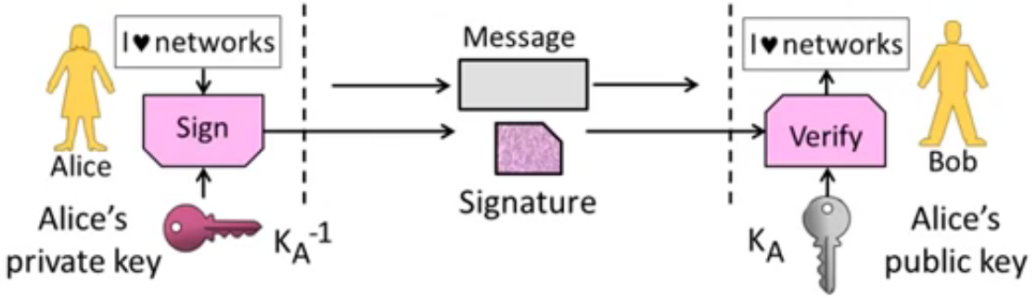


Digital Signature

- Signature validates the integrity/authenticity of a message
  - Send it along with the message
  - Lets all parties validate
  - Example: RSA signatures

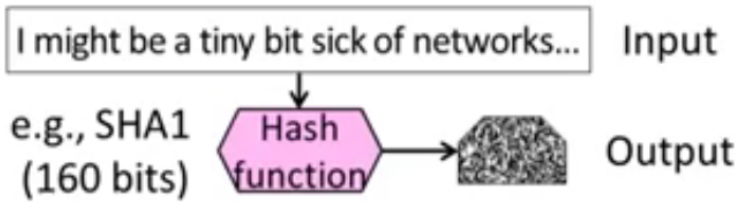


- Kind of public key operation – public/private key parts
  - Alice signs with private key, K_A^{-1}, Bob verifies with public key, K_A
  - Does let Bob convince Charlie that Alice sent the message

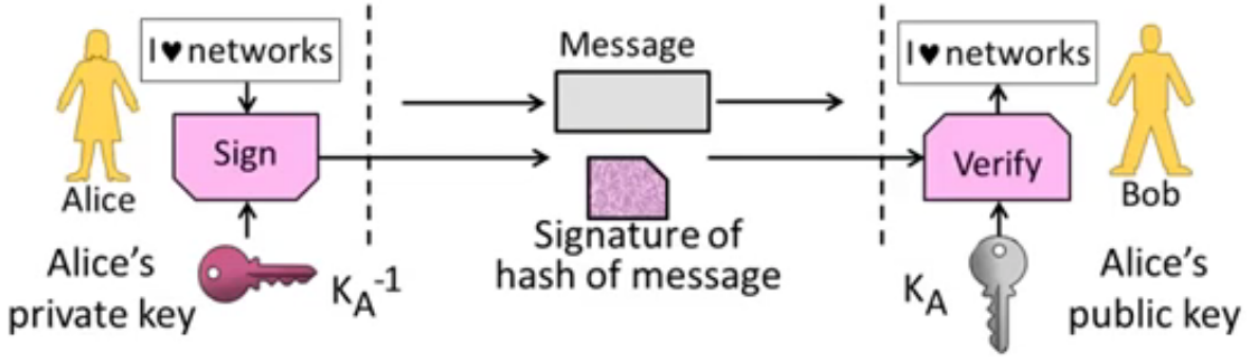


Speeding up Signatures

- Same tension as for confidentiality
  - Public key has keying advantages
  - But it has slow performance!
- Use a technique to speed it up
  - Message digest stands for message
  - Sign the digest instead of full message
- Message Digest or Cryptographic Hash is a secure checksum
  - Deterministically mangles bits to pseudo-random output (like CRC)
  - Can't find messages with same hash
  - Acts as a fixed-length descriptor of message



- Conceptually as before except sign the hash of message
  - Hash is fast to compute, so it speeds up overall operation
  - Hash stands for message as can't find another with same hash



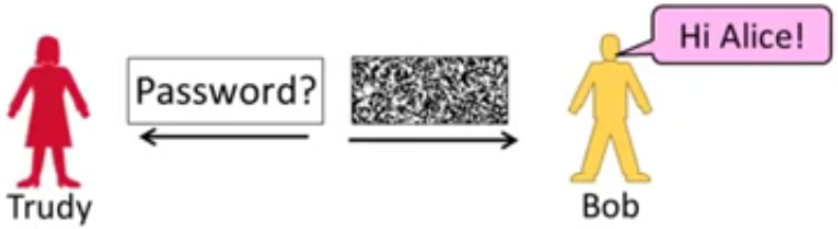
Preventing Replays

- We normally want more than confidentiality, integrity, and authenticity for secure messages!
  - Want to be sure message is fresh

- Don't want to mistake old message for a new one – a replay
  - Acting on it again may cause trouble

- Replay attack:
  - Trudy records Alice's messages to Bob
  - Trudy later replays them (unread) to Bob; she pretends to be Alice

- To prevent replays, include proof of freshness in messages
  - Use a timestamp, or nonce (number once)



Takeaway

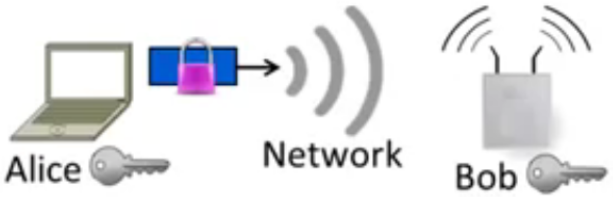
- Cryptographic designs can give us integrity, authenticity and freshness as well as confidentiality.
- Real protocol designs combine the properties in different ways
  - We'll see some examples
  - Note many pitfalls in how to combine, as well as in the primitives themselves

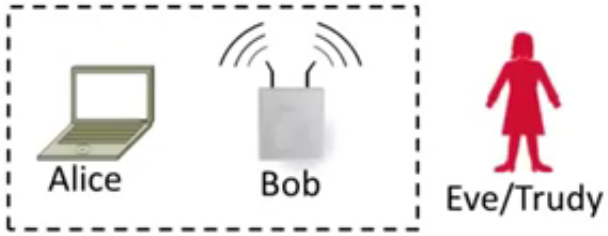
Topics

- Threat models
- Crypto
  - Confidentiality
  - Authentication
- Applied crypto
  - Wireless security (802.11)
  - Web security
  - DNS security
- Connectivity
  - Firewalls
  - Distributed denial-of-service

Wireless Security

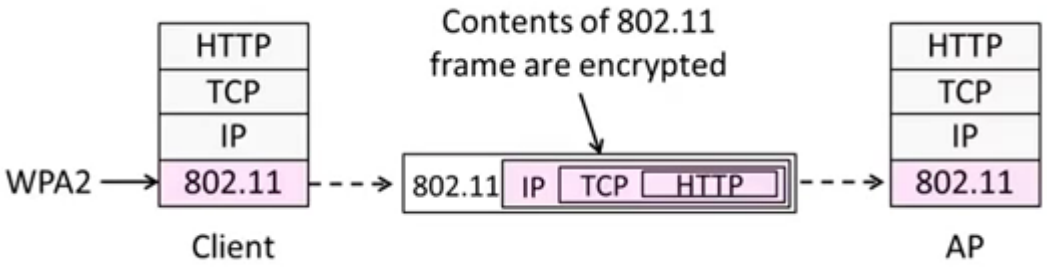
- Securing wireless networks
  - Focus on 802.11
- Goal and Threat Model
  - Unlike wired, wireless messages are broadcast to all nearby receivers
    - Don't need physical network access
    - Heightens security problems
  - Two main threats:
    1. Eavesdropping on conversations
    2. Unauthorized access to network
  - We'll consider 802.11 setting
    - Assume external attacker can send/receive wireless messages





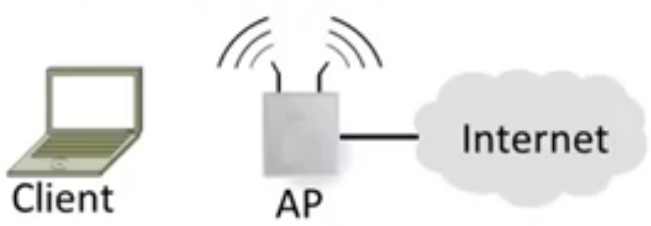
802.11 Security

- Security is based on passwords
  - For access control and confidentiality and integrity/authenticity
- 802.11 standard (1999) used WEP
  - For "wired Equivalent Privacy"
  - Badly flawed, easily broken
- 802.11i standard in 2004
  - WiFi Protected Access or WPA (2)
  - This is what you should use
- Security is part of 802.11 protocol
  - Encrypted message between client and AP; removed after AP



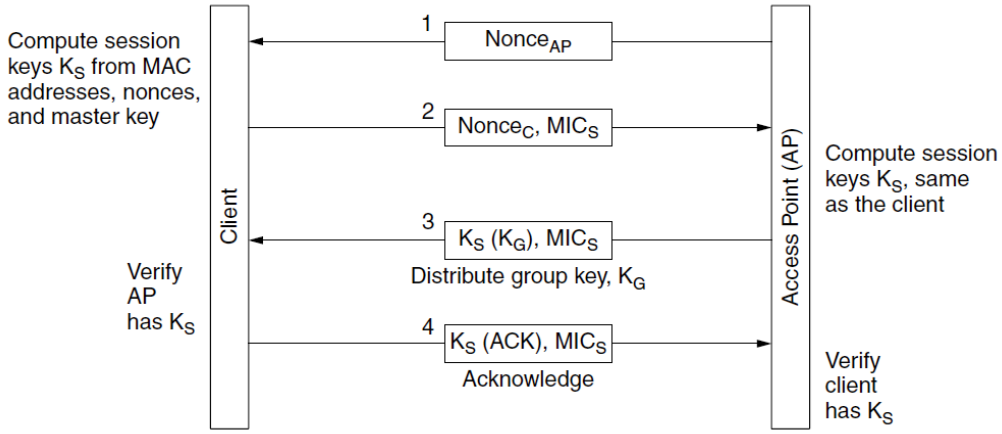
Home Network

- AP is set up with network password
- Each client also knows password
- Client proves it knows password
  - AP grants network access if successful



- For access, client authenticates to AP
  - Different keys need to be derived from a single shared password
  - Both compute a shared session key based on the password
- For usage, client/AP encrypt messages
  - For confidentiality, integrity/authenticity
  - No access without the session key
  - Also group key for AP to reach all clients

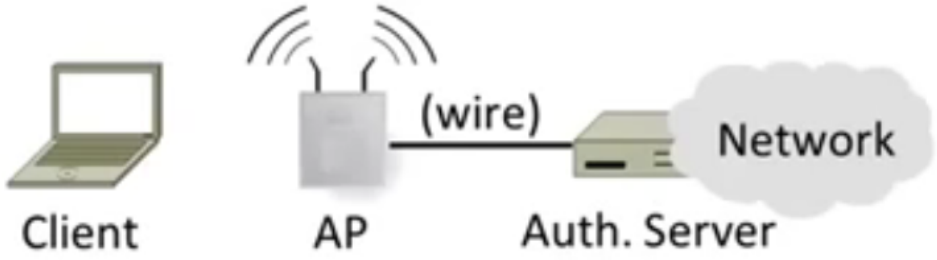
- Goal: compute session key K_S for encrypting traffic
  - Master key is derived from password; nouce for freshness
    - K_S lets client talk to AP
    - K_G lets AP talk to all clients, needs to be updated as clients leave and join the network
    - MIC (Message Integrity Check), another name for MAC



The 802.11i key setup handshake

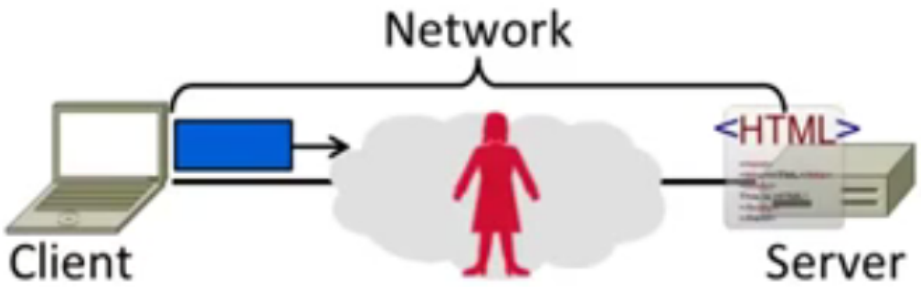
Enterprise Network

- Network has authentication server
- Each client has own credentials
- AP lets client talk to auth. Server
  - Grants network access if successful
- More information refer to 802.1X in the link layer lecture



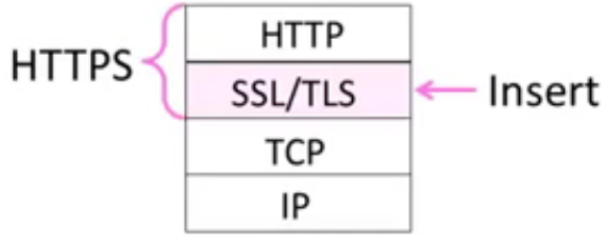
Web Security

- Securing the web
  - Focus on SSL/TLS for HTTPS
  - Including certificates
- Goal and Threat Model
  - Much can go wrong on the web
    - Clients encounter malicious content
    - Web servers are target of break-ins
    - Fake content/servers trick users
    - Data sent over network is stolen …
  - Goal of HTTPS is to secure HTTP
  - We focus on network threats:
    1. Eavesdropping client/server traffic
    2. Tampering with client/server traffic
    3. Impersonating web servers



HTTPS Context

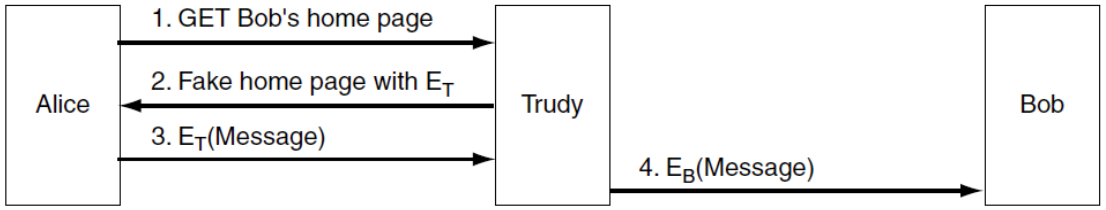
- HTTPS (HTTP Secure) is an add-on
  - Means HTTP over SSL/TLS
    - SSL (Secure Sockets Layer) precedes TLS (Transport Layer Security)
  - Motivated by secure web commerce
    - Can be used by any app, not just HTTP
- SSL came out of Netscape
  - SSL2 (flawed) made public in '95
  - SSL3 fixed flaws in '96
- SSL builds a secure connection between sockets:
  - Parameter negotiation between client and server
  - Authentication of the server by the client
  - Secret communication
  - Data integrity protection



- TLS is the open standard
  - TLS 1.0 in '99, 1.1 in '06, 1.2 in '08
  - Builds on top of SSL3, fall back to SSL
- SSL/TLS Authentication
  - Must allow clients to securely connect to servers not used before
    - Client must authenticate server
    - Server typically doesn't identify client
  - Uses public key authentication
    - But how does client get server's key?
      - With certificates

Certificates

- A certificate binds public key to an identity, e.g., domain, individual, company, etc.
  - Issued by CA (Certification Authority)
  - Distributes public keys when signed by a party you trust
  - Commonly in a format called X.509
- Without certificate, Trudy can:
  - Intercept the GET and replies with a fake home page with the replacement of Bob's public key with her public key
  - Read Alice's message, re-encrypts it with Bob's public key and reads the message from Bob to Alice
  - Modify the message before re-encryping them for Bob



I hereby certify that the public key
19836A8B03030CF83737E3837837FC3s87092827262643FFA82710382828282A
belongs to
Robert John Smith
12345 University Avenue
Berkeley, CA 94702
Birthday: July 4, 1958
Email: bob@superdupernet.com

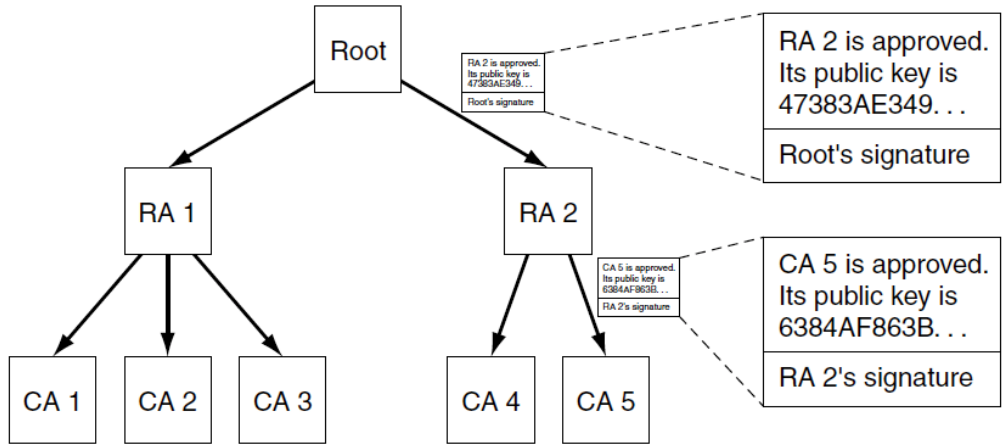
SHA-1 hash of the above certificate signed with the CA's private key

- Now:
  - If Trudy replaces Bob's public key with her own, Alice will get a hash that does not agree with the one she gets when she applied the CA's well-known public key

PKI (Public Key Infrastructure)

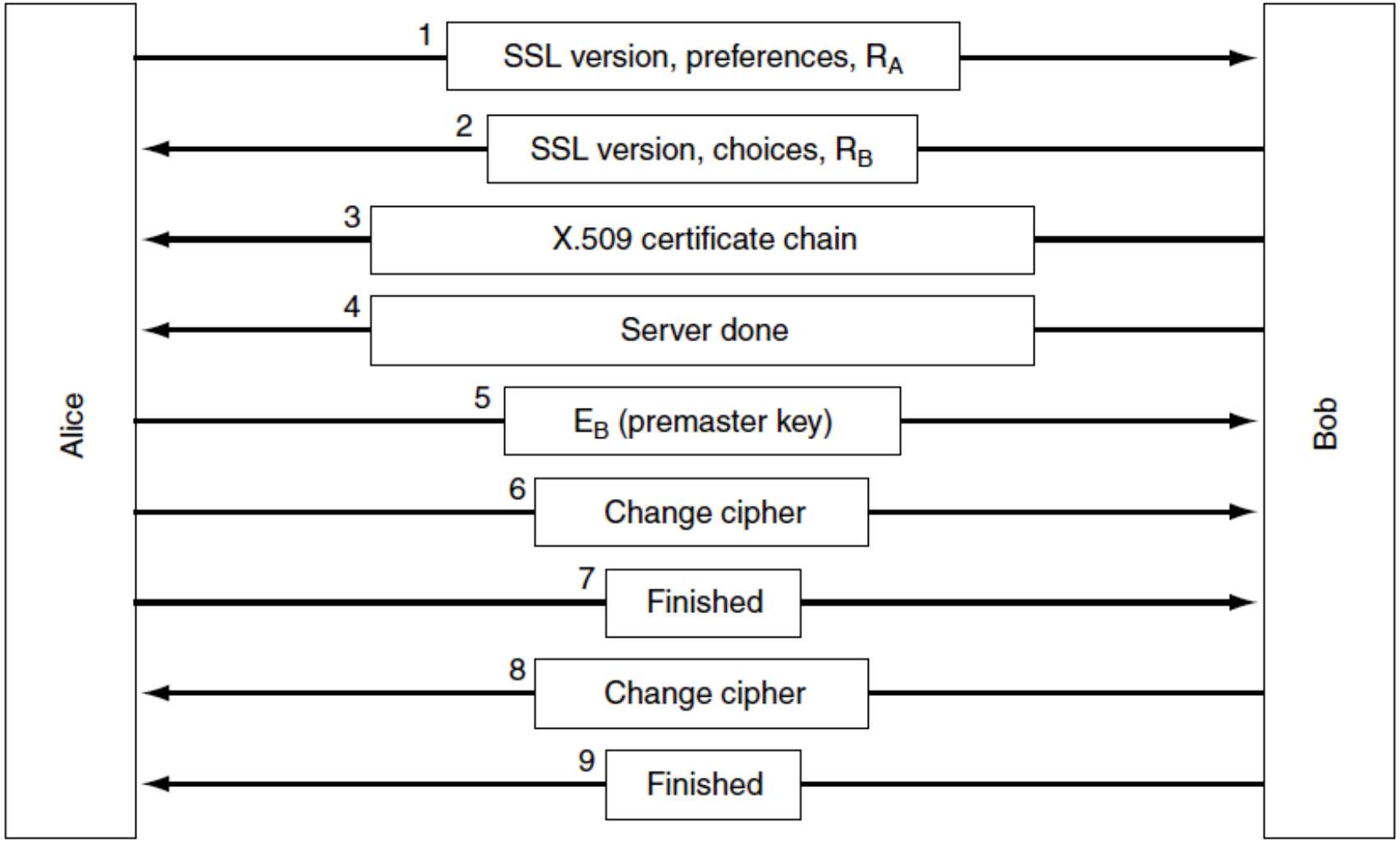
- Adds hierarchy to certificates to let many parties issue
  - Issuing parties are called CAs (Certificate Authorities)
- Need public key of PKI root and trust in servers on path to verify a public key of website ABC
  - Browser has Root's public key
  - {RA 1's key is X} signed Root
  - {CA 1's key is Y} signed RA 1
  - {ABC's key Z} signed CA 1
- Browser/OS has public keys of the trusted roots of PKI
  - >100 root certificates!
  - That's a problem …
  - Inspect your web browser

I certified the ABC website



- Real-world complication:
  - Private keys may be compromised
  - Certificates must then be revoked
- PKI includes a CRL (Certificate Revocation List)
  - Browsers use to weed out bad keys

SSL3 Authentication



Negotiate ciphers, send certificate, …

Certificate lets Alice check Bob

Switch to Alice's session key

Real Bob can compute session key

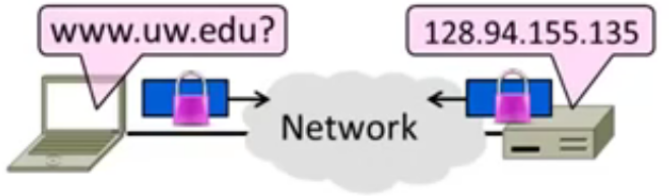
Encrypted data

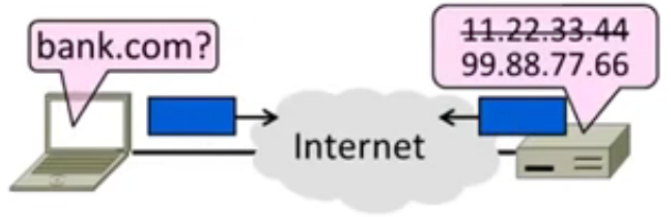
Takeaways

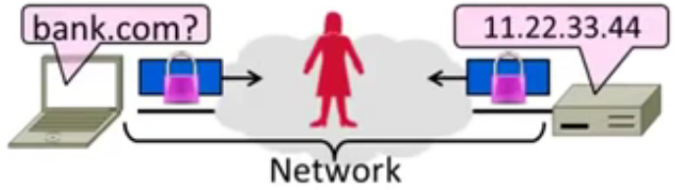
- SSL/TLS is a secure transport
  - For HTTPS and more, with the usual confidentiality, integrity / authenticity
  - Very widely used today
- Client authenticates web server
  - Done with a PKI and certificates
  - Major area of complexity and risk

DNS Security

- Securing Internet naming
  - DNS security extensions (DNSSEC)
- Goal and Threat Model
  - Naming is a crucial Internet service
    - Binds host name to IP address
    - Wrong binging can be disastrous
  - Goal is to secure the DNS so that the returned binding is correct
    - Integrity/authenticity vs confidentiality
  - Attacker can intercept/tamper with messages on the network

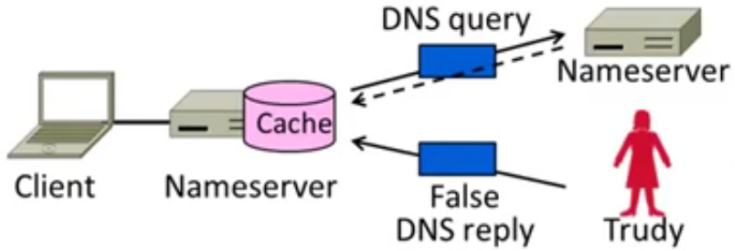






DNS Spoofing

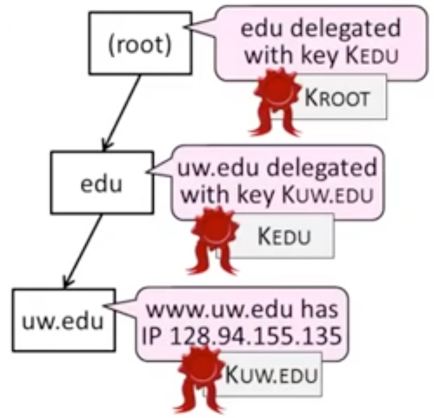
- Hang on – how can a network attacker corrupt the DNS?
- Trudy can trick a nameserver into caching the wrong binding by using the DNS protocol itself
  - Sends lookup request to victim's ISP asking for the IP address of bob.com
  - Immediately sends a fake DNS response to the cache server
    - Assumes cache server has no entry for bob.com and the query response from top level server comes later
  - Fake response contains bad binding, causes DNS cache pollution/poisoning
- Lots of questions!
  - How can Trudy supply a fake DNS reply that appears to be real?
    - Put IP of authoritative nameserver as the source IP address
      - Reply ID that matches the request
  - What happens when the real DNS reply shows up?
    - There is no outstanding query after fake reply is accepted, so real reply will be discarded



DNSSEC (DNS Security Extensions)

- Extends DNS with new record types
  - RRSIG for digital signatures of records
  - DNSKEY for public keys for validation
  - DS for public keys for delegation
  - First version in '97, revised by '05
- Deployment requires software upgrade at both client and server
  - Root servers upgraded in 2010
  - Followed by uptick in deployment
- New records
  - As well as the usual A, NS records
  - RRSIG: Digital signatures of domain records
  - DNSKEY: Public key used for domain RRSIGs
  - DS: Public key used for delegated domain

- Validating Replies
  - DNS clients query DNS as usual, then validate replies to check that content is authentic
  - Trust anchor is root public keys and proceeds down DNS hierarchy
    - Part of DNS client configuration
  - Client queries www.uw.edu as usual
    - Replies include signatures/keys
  - Client validates answer:
    1. K_{ROOT} is a trust anchor
    2. Use K_{ROOT} to check K_{EDU}
    3. Use K_{EDU} to check $K_{UW.EDU}$
    4. Use $K_{UW.EDU}$ to check IP



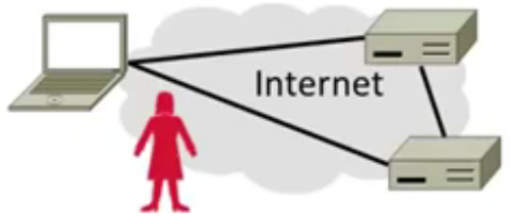
Signature ensures the authenticity of the reply

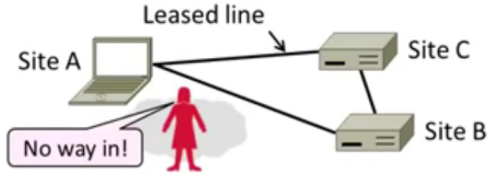
Takeaways

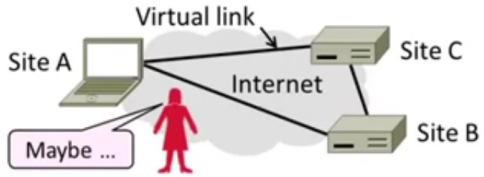
- DNS spoofing is possible without added security measures
  - Large problem in practice!
- DNSSEC adds authentication (only) of replies to the DNS
  - Using a hierarchy of public keys

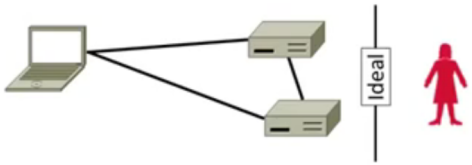
Virtual Private Networks (VPNs)

- Run as closed networks on Internet
- Use IPSEC to secure messages
- Motivation
  - The best part of IP connectivity: You can send to any other host
  - The worst part of IP connectivity: Any host can send packets to you!
    - There's nasty stuff out there …
  - Often desirable to separate network from the Internet, e.g., a company
    - Private network with leased lines
    - Physically separated from Internet
  - Idea: use the public Internet instead of leased lines – cheaper!
    - Logically separated from Internet …
    - This is a Virtual Private Network (VPN)
- Goal and Threat Model
  - Goal is to keep a logical network (VPN) separate from the Internet while using it for connectivity
    - Threat is Trudy may access VPN and intercept or tamper with messages



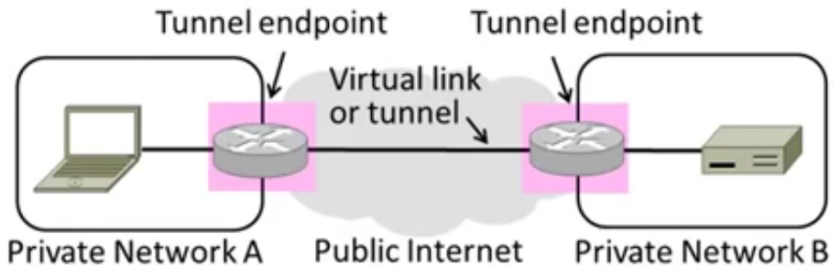


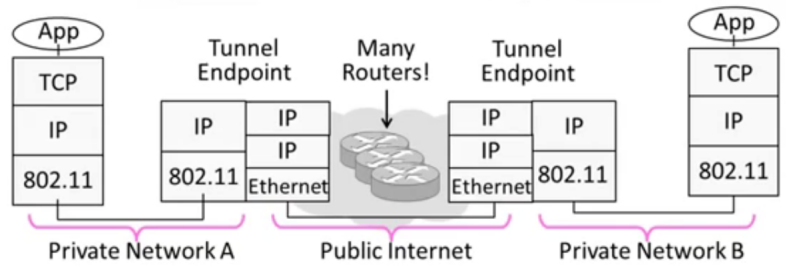


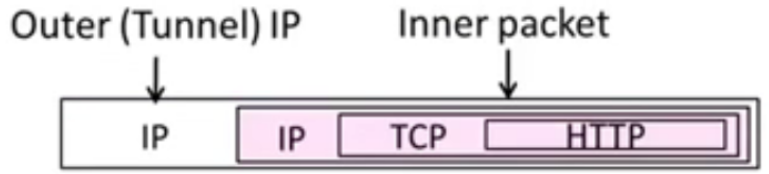


Tunneling

- How can we build a virtual link? With tunneling!
  - Hosts in private network send to each other normally
  - To cross virtual link (tunnel), endpoints encapsulate packet
- Tunnel endpoints encapsulate IP packets ("IP in IP")
  - Add/modify outer IP header for delivery to remote endpoint
- Simplest encapsulation wraps packet with another IP header
  - Outer (tunnel) IP header has tunnel endpoints as source/destination
  - Inner packet has private network IP addresses as source/destination
- Tunneling alone is not secure …
  - No confidentiality, integrity/authenticity
  - Trudy can read, inject her own messages
  - We require cryptographic protections!
- IPSEC (IP Security) is often used to secure VPN tunnels

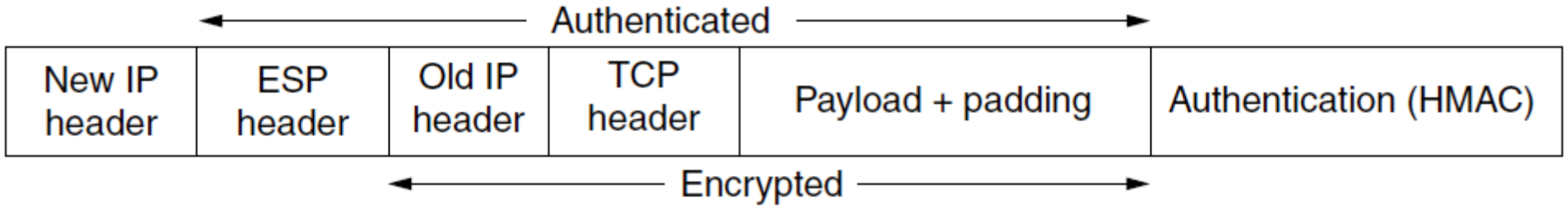






IPSEC (IP Security)

- Longstanding effort to secure the IP layer
  - Adds confidentiality, integrity/authenticity
- IPSEC operation:
  - KEYs are set up for communicating host pairs (tunnel endpoints)
  - Communication becomes more connection-oriented
  - Header and trailer added to protect IP packets



Takeaways

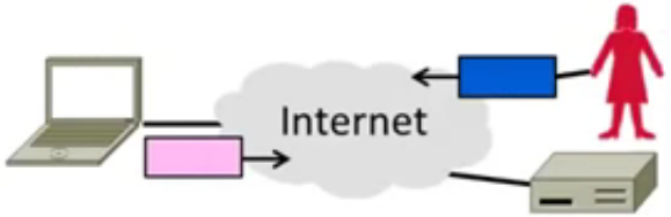
- VPNs are useful for building networks on top of the Internet
  - Virtual links encapsulate packets
  - Alters IP connectivity for hosts
- VPNs need crypto to secure messages
  - Typically IPSEC is used for confidentiality, integrity/authenticity

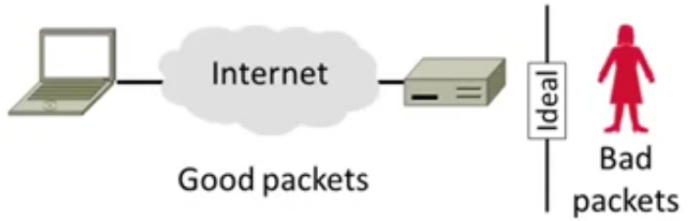
Topics

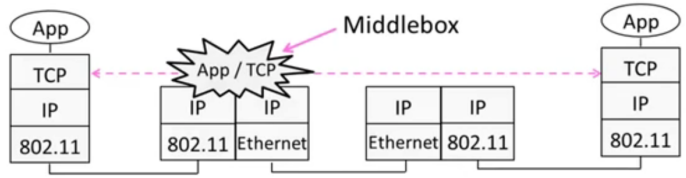
- Threat models
- Crypto
  - Confidentiality
  - Authentication
- Applied crypto
  - Wireless security (802.11)
  - Web security
  - DNS security
- Connectivity
  - Firewalls
  - Distributed denial-of-service

Firewalls

- Protecting hosts by restricting network connectivity
- Motivation
  - The best part of IP connectivity
    - You can send to any other host
  - The worst part of IP connectivity
    - Any host can send packets to you!
    - There's nasty stuff out there …
- Goal and Threat Model
  - Goal of firewall is to implement a boundary to restrict IP connectivity:
    - You can talk to hosts as intended
    - Trudy can't talk to you over network
- Recall Middleboxes
  - Sit "inside the network" but perform "more than IP" processing on packets to add new functionality
    - NAT box, Firewall / Intrusion Detection System

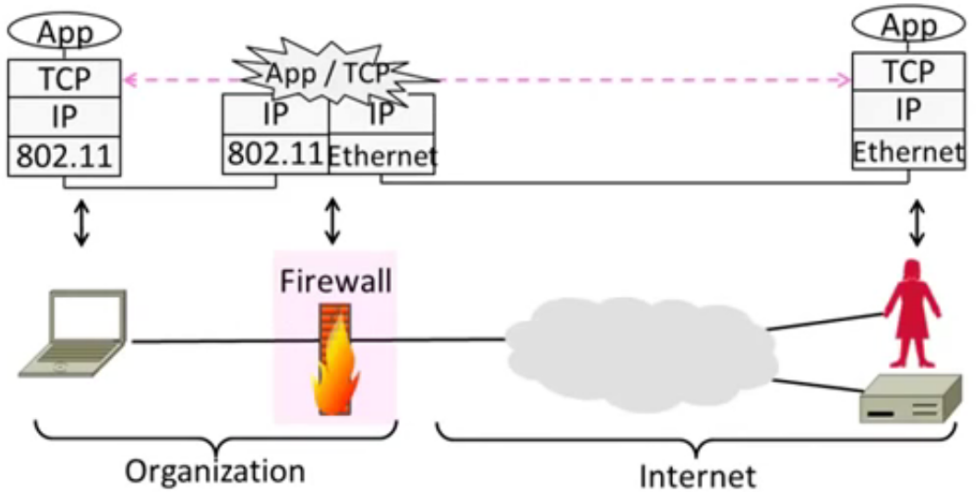






Firewall as Middlebox

- Operation
  - Firewall has two sides:
    - Internal (organization) and external (internet)
  - For each packet that tries to cross, decide whether to:
    - ACCEPT = pass unaltered; or DENY = discard silently
    - Decision is a local policy; firewall centralizes IT job

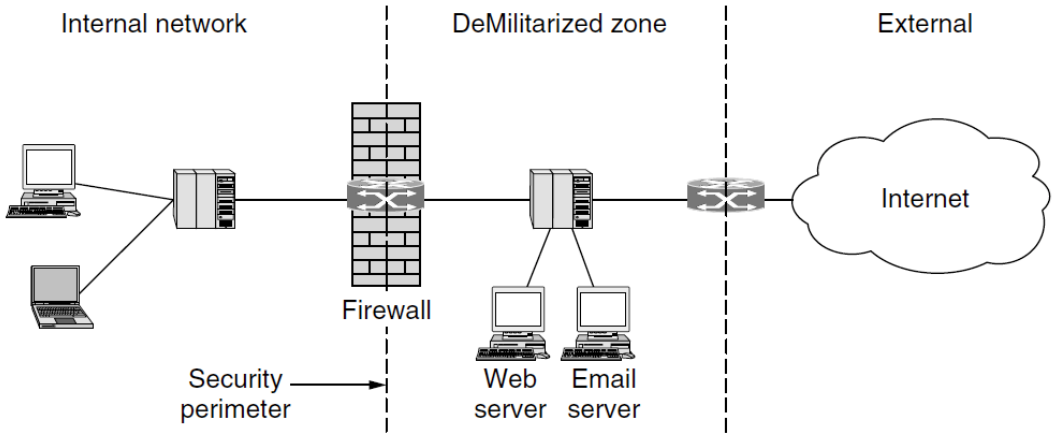


Design

- Key tension:
  - Translate desired policies into packet filtering rules
- Policies are high-level statements
  - Relate to usage of apps, content
- Packet filtering is low-level
  - Limited viewpoint in the network, e.g., no app messages, encryption
- Stateless firewall
  - Simplest kind of firewall
  - Implements static packet filter rules
  - Typically using TCP/UDP ports
  - E.g., deny TCP port 23 (telnet)
  - Can allow/disallow many types of services and destinations
- Stateful firewall
  - A step up from stateless
  - Implements stateful packet filter rules that track packet exchanges
  - NAT example: accept incoming TCP packets after internal host connects
    - Reject outsider's initialtives
- Application layer firewall:
  - Another step up
  - Implements rules based on app usage and content
  - E.g., inspect content for viruses
  - Tries to look beyond packets by emulating higher layers, e.g., by reassembling app messages

Deployment

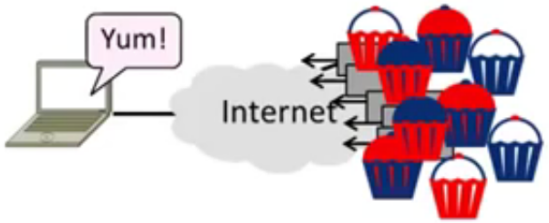
- Firewall is placed around internal/external boundary
  - Classic setup includes DMZ (DeMilitarized Zone) to put busy Internet hosts on the outside for better separation
- Various device options:
  - Specialized network firewall
  - Firewall in boundary device, e.g., AP
  - Firewall as part of host, e.g., in OS
- Tradeoff:
  - Centralizing simplifies IT job
  - Distributing improves protection, visibility into apps, and performance

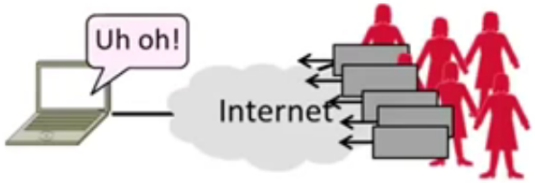


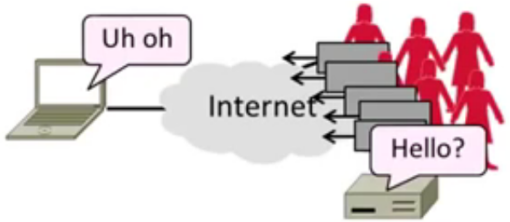
In case Web/Email server got compromised …

Distributed Denial-of-Service (DDOS)

- An attack on network availability
- Motivation
  - The best part of IP connectivity
    - You can send to any other host
  - The worst part of IP connectivity
    - Any host can send packets to you!
  - Flooding a host with many packets can interfere with its IP connectivity
    - Host may become unresponsive
    - This is a form of denial-of-service
- Goal and Threat Model
  - Goal is for host to keep network connectivity for desired services
  - Threat is Trudy may overwhelm host with undesired traffic





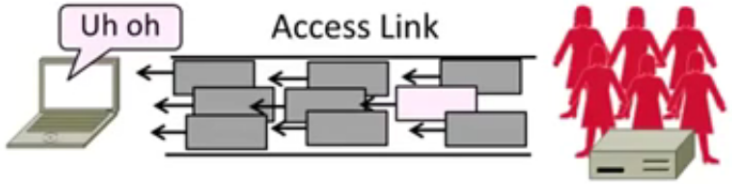


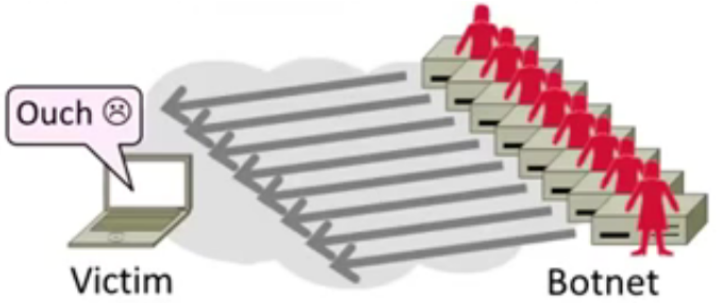
Internet Reality

- Distributed Denial-of-Service is a huge problem today!
  - Akamai Q3-12 reports DDOS against US banks peaking at 65Gbps …
- There are no great solutions
  - CDNs, network traffic filtering, and best practices all help

Denial-of-Service

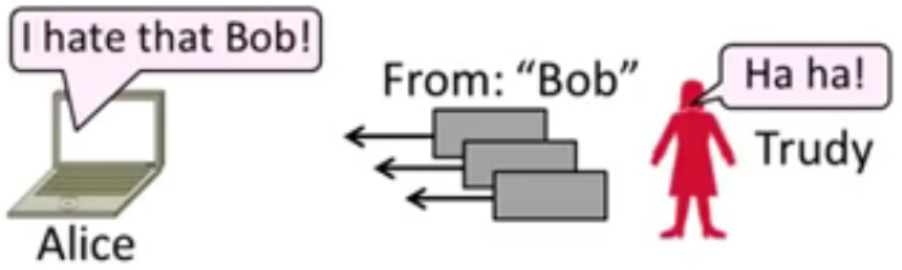
- Denial-of-Service means a system (server) is made unavailable to intended users
  - Typically because its resources (network bandwidth, host CPU or memory) are consumed by attackers instead
- Host Denial-of-Service
  - Strange packets can sap host resources!
    - "Ping of death" malformed packet – bug driven
    - "SYN flood" sends many TCP connect requests and never follows up
  - Patches exist for these vulnerabilities
    - Read about "SYN cookies" for interest
- Network Denial-of-Service
  - Network DOS needs many packets
    - To saturate network links and cause high congestion/loss
  - Helpful to have many attackers … or Distributed Denial-of-Service (DDoS)
    - Botnet provides many attackers in the form of compromised hosts
      - Hosts send traffic flood to victim
      - Network saturates near victim



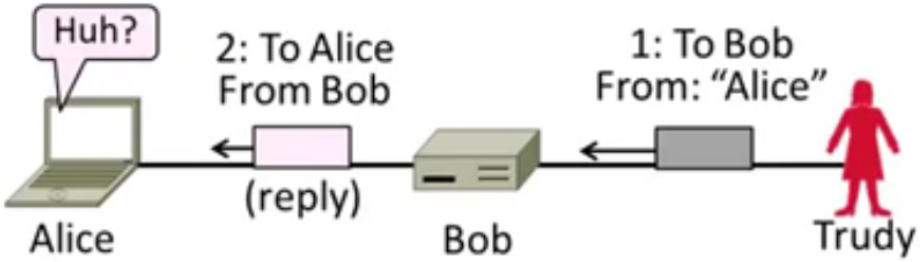


Complication: Spoofing

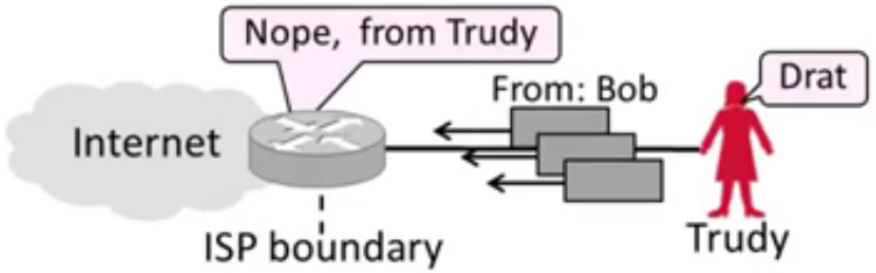
- Attackers can falsify their IP address
  - Put fake source address on packets
  - Historically network doesn't check
  - Hides location of the attackers
  - Called IP address spoofing



- Actually, it's worse than that
  - Trudy can trick Bob into really sending packets to Alice
  - To do so, Trudy spoofs Alice to Bob



- Best Practice: Ingress Filtering
  - Idea: validate the IP source address of packets at ISP boundary
    - Ingress filtering is a best practice, but deployment has been slow



Flooding Defenses

- Increase network capacity around the server; harder to cause loss
  - Use a CDN for high peak capacity
- Filter out attack traffic within the network (at routers)
  - The earlier the filtering, the better
  - Ultimately what is needed, but ad hoc measures by ISPs today

Presentation Schedule (12:30 – 14:50, 15 min for each)

12/19

- 黎才华 DARPA
- 唐毅 E2Earg
- 刘汉彻 Internet@C
- 任泓宇 MIMO
- 姚思羽 DCNets
- 胡子牛 DCTCP
- 田得雨 CAN
- 吴文俊 Whitespace
- 郭天魁 CongestionManager

12/26

- 沈洋 Chord
- 史桀绮 P2P Video
- 陈淙靓 QoE
- 姜宛彤 WirelessTCP
- 刘鑫远 MPTCP
- 朱近 SDN
- 杨垒 OpenFlow
- 周子凯 TOR
- 郑潇龙 AIP