

CSci 4271W
 Development of Secure Software Systems
 Day 23: Networking and security

Stephen McCamant
 University of Minnesota, Computer Science & Engineering

Outline

- Brief introduction to networking
- Cryptographic protocols
- Announcements intermission
- Some classic network attacks
- Key distribution and PKI

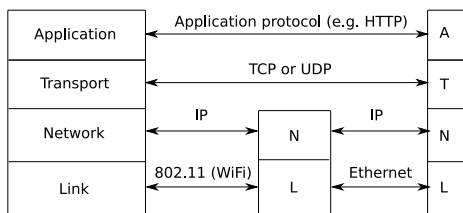
The Internet

- A bunch of computer networks voluntarily interconnected
- Capitalized because there's really only one
- No centralized network-level management
 - But technical collaboration, DNS, etc.

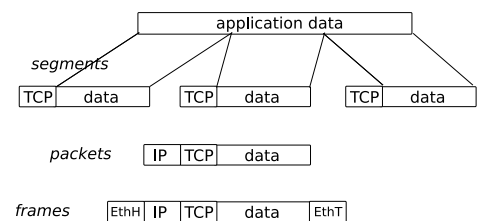
Layered model (OSI)

7. Application (HTTP)
6. Presentation (MIME?)
5. Session (SSL?)
4. Transport (TCP)
3. Network (IP)
2. Data-link (PPP)
1. Physical (IOBASE-T)

Layered model: TCP/IP



Packet wrapping



IP(v4) addressing

- Interfaces (hosts or routers) identified by 32-bit addresses
 - Written as four decimal bytes, e.g. 192.168.10.2
- First k bits identify network, $32 - k$ host within network
 - Can't (anymore) tell k from the bits
- We'll run out any year now

IP and ICMP

- Internet Protocol (IP) forwards individual packets
- Packets have source and destination addresses, other options
- Automatic fragmentation (usually avoided)
- ICMP (I Control Message P) adds errors, ping packets, etc.

UDP

- User Datagram Protocol: thin wrapper around IP
- Adds source and destination port numbers (each 16-bit)
- Still connectionless, unreliable
- OK for some small messages

TCP

- Transmission Control Protocol: provides reliable bidirectional stream abstraction
- Packets have sequence numbers, acknowledged in order
- Missed packets resent later

Flow and congestion control

- Flow control: match speed to slowest link
 - "Window" limits number of packets sent but not ACKed
- Congestion control: avoid traffic jams
 - Lost packets signal congestion
 - Additive increase, multiplicative decrease of rate

Routing

- Where do I send this packet next?
 - Table from address ranges to next hops
- Core Internet routers need big tables
- Maintained by complex, insecure, cooperative protocols
 - Internet-level algorithm: BGP (Border Gateway Protocol)

Below IP: ARP

- Address Resolution Protocol maps IP addresses to lower-level address
 - E.g., 48-bit Ethernet MAC address
- Based on local-network broadcast packets
- Complex Ethernets also need their own routing (but called switches)

DNS

- Domain Name System: map more memorable and stable string names to IP addresses
- Hierarchically administered namespace
 - Like Unix paths, but backwards
- `.edu` server delegates to `.ummn.edu` server, etc.

DNS caching and reverse DNS

- To be practical, DNS requires caching
 - Of positive and negative results
- But, cache lifetime limited for freshness
- Also, reverse IP to name mapping
 - Based on special top-level domain, IP address written backwards

Classic application: remote login

- Killer app of early Internet: access supercomputers at another university
- Telnet: works cross-OS
 - Send character stream, run regular login program
- rlogin: BSD Unix
 - Can authenticate based on trusting computer connection comes from
 - (Also rsh, rcp)

Outline

Brief introduction to networking
Cryptographic protocols
Announcements intermission
Some classic network attacks
Key distribution and PKI

A couple more security goals

- Non-repudiation: principal cannot later deny having made a commitment
 - I.e., consider proving fact to a third party
- Forward secrecy: recovering later information does not reveal past information
 - Motivates using Diffie-Hellman to generate fresh keys for each session

Abstract protocols

- Outline of what information is communicated in messages
 - Omit most details of encoding, naming, sizes, choice of ciphers, etc.
- Describes honest operation
 - But must be secure against adversarial participants
- Seemingly simple, but many subtle problems

Protocol notation

$A \rightarrow B : N_B, \{T_0, B, N_B\}_{K_B}$

- $A \rightarrow B$: message sent from Alice intended for Bob
- B (after $:$): Bob's name
- $\{\dots\}_K$: encryption with key K

Example: simple authentication

$A \rightarrow B : A, \{A, N\}_{K_A}$

- E.g., Alice is key fob, Bob is garage door
- Alice proves she possesses the pre-shared key K_A
 - Without revealing it directly
- Using encryption for authenticity and binding, not secrecy

Nonce

$A \rightarrow B : A, \{A, N\}_{K_A}$

- N is a *nonce*: a value chosen to make a message unique
- Best practice: pseudorandom
- In constrained systems, might be a counter or device-unique serial number

Replay attacks

- A nonce is needed to prevent a verbatim replay of a previous message
- Garage door difficulty: remembering previous nonces
 - Particularly: lunchtime/roommate/valet scenario
- Or, door chooses the nonce: *challenge-response* authentication

Middleperson attacks

- Older name: man-in-the-middle attack, MITM
- Adversary impersonates Alice to Bob and vice-versa, relays messages
- Powerful position for both eavesdropping and modification
- No easy fix if Alice and Bob aren't already related

Chess grandmaster problem

- Variant or dual of middleperson
- Adversary forwards messages to simulate capabilities with his own identity
- How to win at correspondence chess
- Anderson's MiG-in-the-middle

Anti-pattern: "oracle"

- Any way a legitimate protocol service can give a capability to an adversary
- Can exist whenever a party decrypts, signs, etc.
- "Padding oracle" was an instance of this at the implementation level

Outline

Brief introduction to networking
Cryptographic protocols
Announcements intermission
Some classic network attacks
Key distribution and PKI

Note to early readers

- This is the section of the slides most likely to change in the final version
- If class has already happened, make sure you have the latest slides for announcements

Outline

Brief introduction to networking
Cryptographic protocols
Announcements intermission
Some classic network attacks
Key distribution and PKI

Packet sniffing

- Watch other people's traffic as it goes by on network
- Easiest on:
 - Old-style broadcast (thin, "hub") Ethernet
 - Wireless
- Or if you own the router

Forging packet sources

- Source IP address not involved in routing, often not checked
- Change it to something else!
- Might already be enough to fool a naive UDP protocol

TCP spoofing

- Forging source address only lets you talk, not listen
- Old attack: wait until connection established, then DoS one participant and send packets in their place
- Frustrated by making TCP initial sequence numbers unpredictable
 - Fancier attacks modern attacks are "off-path"

ARP spoofing

- Impersonate other hosts on local network level
- Typical ARP implementations stateless, don't mind changes
- Now you get victim's traffic, can read, modify, resend

rlogin and reverse DNS

- rlogin uses reverse DNS to see if originating host is on whitelist
- How can you attack this mechanism with an honest source IP address?

rlogin and reverse DNS

- rlogin uses reverse DNS to see if originating host is on whitelist
- How can you attack this mechanism with an honest source IP address?
- Remember, ownership of reverse-DNS is by IP address

Outline

- Brief introduction to networking
- Cryptographic protocols
- Announcements intermission
- Some classic network attacks
- Key distribution and PKI

Public key authenticity

- Public keys don't need to be secret, but they must be right
- Wrong key → can't stop middleperson
- So we still have a pretty hard distribution problem

Symmetric key servers

- Users share keys with server, server distributes session keys
- Symmetric key-exchange protocols, or channels
- Standard: Kerberos
- Drawback: central point of trust

Certificates

- A name and a public key, signed by someone else
 - $C_A = \text{Sign}_S(A, K_A)$
- Basic unit of transitive trust
- Commonly use a complex standard "X.509"

Certificate authorities

- "CA" for short: entities who sign certificates
- Simplest model: one central CA
- Works for a single organization, not the whole world

Web of trust

- Pioneered in PGP for email encryption
- Everyone is potentially a CA: trust people you know
- Works best with security-motivated users
 - Ever attended a key signing party?

CA hierarchies

- Organize CAs in a tree
- Distributed, but centralized (like DNS)
- Check by follow a path to the root
- Best practice: sub CAs are limited in what they certify

PKI for authorization

- Enterprise PKI can link up with permissions
- One approach: PKI maps key to name, ACL maps name to permissions
- Often better: link key with permissions directly, name is a comment

The revocation problem

- How can we make certs "go away" when needed?
- Impossible without being online somehow
 1. Short expiration times
 2. Certificate revocation lists
 3. Certificate status checking