CSci 5271
Introduction to Computer Security
Day 14: Network, etc., security overview

Stephen McCamant
University of Minnesota, Computer Science & Engineering

Outline

Brief introduction to networking

Midterm debrief, etc.

Some classic network attacks

Second half of course

More Unix access control

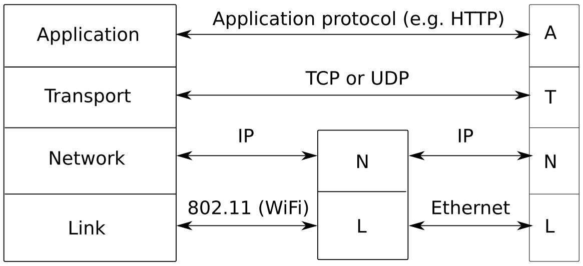
The Internet

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

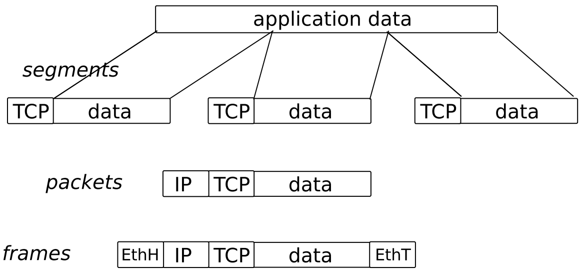
Layered model (OSI)

7. Application (HTTP)
6. Presentation (MIME?)
5. Session (SSL?)
4. Transport (TCP)
3. Network (IP)
2. Data-link (PPP)
1. Physical (10BASE-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 (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 `.umn.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

Midterm debrief, etc.

Some classic network attacks

Second half of course

More Unix access control

Midterm results schedule

- Graded yesterday, posted on Moodle last night
- Paper copies here today (available after)
- Some discussion now
- Full solution set posted later this week

Midterm result: high-order bit

- Failed to make test easier than last year's
- Results on both easier and harder questions disappointing to me
- Final grade will reflect a $+16$ point adjustment
- Final exam similar, but less time pressure

(Non-) race condition 1

```
int fd = open("file", O_WRONLY);
int res = fstat(fd, &st_buf);
if ((st_buf.st_mode & 0222) != 0222)
  abort();
write(fd, data, data_size);
```

(Non-) race condition 2

```
int res =
  stat("/etc/hostname", &st_buf);
char *buf = malloc(st_buf.st_size);
int fd =
  open("/etc/hostname", O_RDONLY);
read(fd, buf, st_buf.st_size);
write(1, buf, st_buf.st_size);
```

(Non-) race condition 3

```
int res = stat("file", &st_buf);
if ((st_buf.st_mode & 0222) != 0222)
  abort();
int fd = open("file", O_WRONLY);
write(fd, data, data_size);
```

(Non-) race condition 4

```
int fd = open("file",
  O_CREAT|O_WRONLY|O_TRUNC, 0666);
int res = fchmod(fd, 0600);
write(fd, secret, 1024);
```

Q5 attack

- Interpret user-controlled point (I:) as list node (D:)
- `list_delete` provides attacker-controlled write
- GOT overwrite replaces `puts` with `system`
- "Print" user-controlled message `/bin/sh` (F:)

Project meetings schedule

- Mostly this week, invited over the weekend
- Some will spill into next week
- Third meetings tentatively 11/17-11/21

Optional reading

- Ch. 2-3 of *Firewalls and Internet Security*, 2nd Ed.
- Security-oriented overview of network protocols
- Not posted until last night (sorry), optional
- Crypto readings start for Thursday

Outline

Brief introduction to networking

Midterm debrief, etc.

Some classic network attacks

Second half of course

More Unix access control

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
  - But see Oakland'12, WOOT'12 for fancier attacks, keyword "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

Midterm debrief, etc.

Some classic network attacks

Second half of course

More Unix access control

Cryptographic primitives

- Core mathematical tools
- Symmetric: block cipher, hash function, MAC
- Public-key: encryption, signature
- Some insights on how they work, but concentrating on how to use them correctly

Cryptographic protocols

- Sequence of messages and crypto privileges for, e.g., key exchange
- A lot can go wrong here, too
- Also other ways security can fail even with a good crypto primitive

Crypto in Internet protocols

- How can we use crypto to secure network protocols
- E.g., rsh → ssh
- Challenges of getting the right public keys
- Fitting into existing usage ecosystems

Web security: server side

- Web software is privileged and processes untrusted data: what could go wrong?
- Shell script injection (Ex. 1)
- SQL injection
- Cross-site scripting (XSS) and related problems

Web security: client side

- JavaScript security environment even more tricky, complex
- More kinds of cross-site scripting
- Possibilities for sandboxing

Security middleboxes

- Firewall: block traffic according to security policy
- NAT box: different original purpose, now de-facto firewall
- IDS (Intrusion Detection System): recognize possible attacks

Malware and network DoS

- Attacks made possible by the network
- Viruses, trojans, bot nets
  - Detection?
  - Mitigation?
- Distributed denial of service (DDoS)

Usability of security

- Prevent people from being the weakest link
- Usability of authentication
- “Secure” web sites, phishing
- Making decisions about mobile apps

Electronic money (Bitcoin)

- Current payment systems have strong centralized trust
  - US Federal Reserve and mint
  - Banks, PayPal
- Could they be replaced by a peer-to-peer distributed system?
- Maybe

Outline

Brief introduction to networking

Midterm debrief, etc.

Some classic network attacks

Second half of course

More Unix access control

“POSIX” “capabilities”

- Divide root privilege into smaller (~35) pieces
- Note: not real capabilities
- First runtime only, then added to FS similar to setuid
- Motivating example: `ping`
- Also allows permanent disabling

Privilege escalation dangers

- Many pieces of the root privilege are enough to regain the whole thing
  - Access to files as UID 0
  - `CAP_DAC_OVERRIDE`
  - `CAP_FOWNER`
  - `CAP_SYS_MODULE`
  - `CAP_MKNOD`
  - `CAP_PTRACE`
  - `CAP_SYS_ADMIN` (`mount`)

Legacy interaction dangers

- Former bug: take away capability to drop privileges
- Use of temporary files by no-longer setuid programs
- For more details: “Exploiting capabilities”, Emeric Nasi

Next time

- Symmetric crypto primitives