

Lecture 3, April 3, 2026

Identity on the Web

- Host identity
 - Static identifiers: do not change over time
 - Dynamic identifiers: changes as a result of an event or the passing of time
- State and Cookies
- Anonymity
 - Anonymous email
 - Anonymity: good or bad?

Host Identity

- Bound up to networking
 - Not connected: pick any name
 - Connected: one or more names depending on interfaces, network structure, context
- *Name* identifies principal
- *Address* identifies location of principal
 - May be virtual location (network segment) as opposed to physical location (room 222)

Example

- Layered network
 - MAC layer
 - Ethernet address: 00:05:02:6B:A8:21
 - AppleTalk address: network 51, node 235
 - Network layer
 - IP address: 192.168.35.89
 - Transport layer
 - Host name: cherry.orchard.chekhov.ru

Danger!

- Attacker spoofs identity of another host
 - Protocols at, above the identity being spoofed will fail
 - They rely on spoofed, and hence faulty, information
- Example: spoof IP address, mapping between host names and IP addresses

Domain Name Server

- Maps transport identifiers (host names) to network identifiers (host addresses)
 - Forward records: host names → IP addresses
 - Reverse records: IP addresses → host names
- Weak authentication
 - Not cryptographically based
 - Various techniques used, such as reverse domain name lookup
- DNSSEC (security-enhanced DNS) uses strong authentication
 - Based on cryptography

Reverse Domain Name Lookup

- Validate identity of peer (host) name
 - Get IP address of peer
 - Get associated host name via DNS
 - Get IP addresses associated with host name from DNS
 - If first IP address in this set, accept name as correct; otherwise, reject as spoofed
- If DNS corrupted, this won't work

Floating (Dynamic) Identifiers

- Assigned to principals for a limited time
 - Server maintains pool of identifiers
 - Client contacts server using *local identifier*
 - Only client, server need to know this identifier
 - Server sends client *global identifier*
 - Client uses global identifier in other contexts, for example to talk to other hosts
 - Server notifies intermediate hosts of new client, global identifier association

Example: DHCP

- DHCP server has pool of IP addresses
- Laptop sends DHCP server its MAC address, requests IP address
 - MAC address is local identifier
 - IP address is global identifier
- DHCP server sends unused IP address
 - Also notifies infrastructure systems of the association between laptop and IP address
- Laptop accepts IP address, uses that to communicate with hosts other than server

Example: Gateways

- Laptop wants to access host on another network
 - Laptop's address is 10.1.3.241
- Gateway assigns legitimate address to internal address
 - Say IP address is 156.43.21.241
 - Gateway rewrites all outgoing, incoming packets appropriately
 - Invisible to both laptop, remote peer
- Internet protocol NAT works this way
 - NAT stands for Network Address Translation

Weak Authentication

- Static: host/name binding fixed over time
- Dynamic: host/name binding varies over time
 - Must update reverse records in DNS
 - Otherwise, the reverse lookup technique fails
 - Cannot rely on binding remaining fixed unless you know the period of time over which the binding persists

DNS Security Issues

- Trust is that name/IP address binding is correct
- Goal of attacker: associate incorrectly an IP address with a host name
 - Assume attacker controls name server, or can intercept queries and send responses

Attacks

- Change records on server
- Add extra record to response, giving incorrect name/IP address association
 - Called “cache poisoning”
- Attacker sends victim request that must be resolved by asking attacker
 - Attacker responds with answer plus two records for address spoofing (1 forward, 1 reverse)
 - Called “ask me”

Cookies

- Token containing information about state of transaction on network
 - Usual use: refers to state of interaction between web browser, client
 - Idea is to minimize storage requirements of servers, and put information on clients
- Client sends cookies to server

Some Fields in Cookies

- *name, value*: *name* has given *value*
- *expires*: how long cookie valid
 - Expired cookies discarded, not sent to server
 - If omitted, cookie deleted at end of session
- *domain*: domain for which cookie intended
 - Consists of last n fields of domain name of server
 - *Must* have at least one “.” in it
- *secure*: send only over secured (TLS, HTTPS) connection

Example: Web Cookies

- Caroline puts 2 books in shopping cart at books.com
 - Cookie: *name* bought, *value* BK=234&BK=8753, *domain* .books.com
- Caroline looks at other books, but decides to buy only those
 - She goes to the purchase page to order them
- Server requests cookie, gets above
 - From cookie, determines books in shopping cart

Who Can Get the Cookies?

- Web browser can send *any* cookie to a web server
 - Even if the cookie's domain does not match that of the web server
 - Usually controlled by browser settings
- Web server can *only* request cookies for its domain
 - Cookies need not have been sent by that browser

Where Did the Visitor Go?

- Server books.com sends Caroline 2 cookies
 - First described earlier
 - Second has *name* “id”, *value* “books.com”, *domain* “adv.com”
- Advertisements at books.com include some from site adv.com
 - When drawing page, Caroline’s browser requests content for ads from server “adv.com”
 - Server requests cookies from Caroline’s browser
 - By looking at *value*, server can tell Caroline visited “books.com”

Principles of Secure Design

- Simplicity, restriction
- Principles
 - Least Privilege
 - Fail-Safe Defaults
 - Economy of Mechanism
 - Complete Mediation
 - Open Design
 - Separation of Privilege
 - Least Common Mechanism
 - Least Astonishment

Overview

- **Simplicity**
 - Less to go wrong
 - Fewer possible inconsistencies
 - Easy to understand
- **Restriction**
 - Minimize access
 - Inhibit communication

Least Privilege

- A subject should be given only those privileges necessary to complete its task
 - Function, not identity, controls
 - Rights added as needed, discarded after use
 - Minimal protection domain

Examples

- The UNIX/Linux user *root*: no access controls applied
- Mail server running as an ordinary user
 - May need to have *root* privileges to open port 25
 - Needs to be able to create files in spool directory

Related: Least Authority

- Principle of Least Authority (POLA)
 - Often considered the same as Principle of Least Privilege
 - Some make distinction:
 - *Permissions* control what subject can do to an object directly
 - *Authority* controls what influence a subject has over an object (directly or indirectly, through other subjects)

Fail-Safe Defaults

- Default action is to deny access
- If action fails, system as secure as when action began

Example: Mail Spool Directory Full

- What to do
 - Notify client email is rejected due to full disk, and close connection
 - SMTP error code is 431
 - Notify administrator that spool directory cannot be written to as it is full
- What not to do
 - Increase privileges so it can store message elsewhere
 - Begin deleting old spooled mail messages

Economy of Mechanism

- Keep it as simple as possible
 - KISS Principle
- Simpler means less can go wrong
 - And when errors occur, they are easier to understand and fix
- Interfaces and interactions

Complete Mediation

- Check every access
- Usually done once, on first action
 - UNIX: access checked on open, not checked thereafter
- If permissions change after, may get unauthorized access

Examples: Complete Mediation

- When UNIX/Linux checks permissions to read, write a file
 - At open *only*
- DNS cache poisoning
 - Attacker inserts bogus DNS record in a reply
 - Victim contacts host with poisoned IP address
 - IP address is *not* revalidated so this goes to the wrong host

Open Design

- Security should not depend on secrecy of design or implementation
 - Popularly misunderstood to mean that source code should be public
 - “Security through obscurity”
 - Does not apply to information such as passwords or cryptographic keys
 - Plan for compromise of anything kept secret