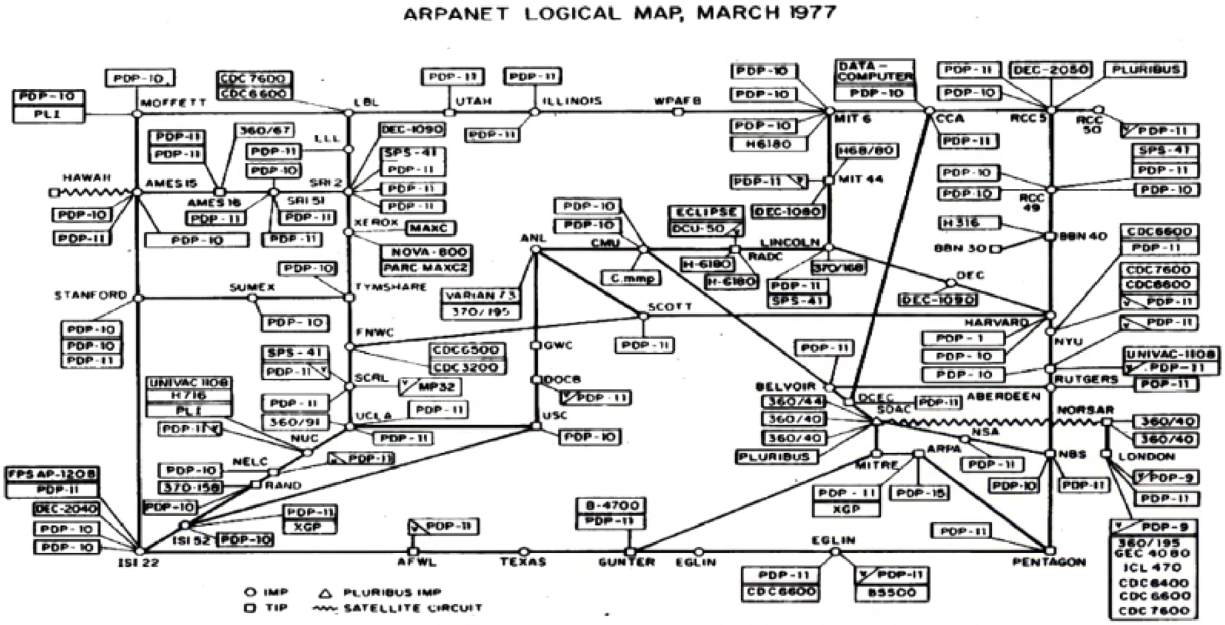
Information Security

Principles of Information Security

1. Principles of Information Security Introduction
    - Introduction to terminologies pertaining to information security
    - Cryptographic techniques used to provide distributed authentication
        - Private key infrastructure
2. Firsts from Computing Pioneers
    - First vision for a network of computers: 1963
    - First computer-computer communication: 1969
    - First email: 1971
    - Main frame: Contained CPU, memory, and I/O with CRT terminals to connect to the mainframe in 1975 (timesharing system)



All of the Computers in the World (1977)

3. Terminologies
    - When to release information?
        - Privacy: When do individuals expect information to be protected or released?
        - Security: Concerned with protection and authentication (system)
    - Comprehensive set of security concerns
        - Unauthorized information release
        - Unauthorized information modification
        - Unauthorized denial of use (DOS attacks)
    - Goal of a secure system
        - Prevent all violations of the concerns above
        - Negative statement, impossible to achieve absolutely
        - At best, negative statement provides false sense of security
4. Levels of Protection
    - Unprotected
        - MSDOS: Hooks for mistake prevention, not the same as security
    - All or nothing
        - IBM VM-370: Only way to interact with one another is explicit I/O across virtual machines

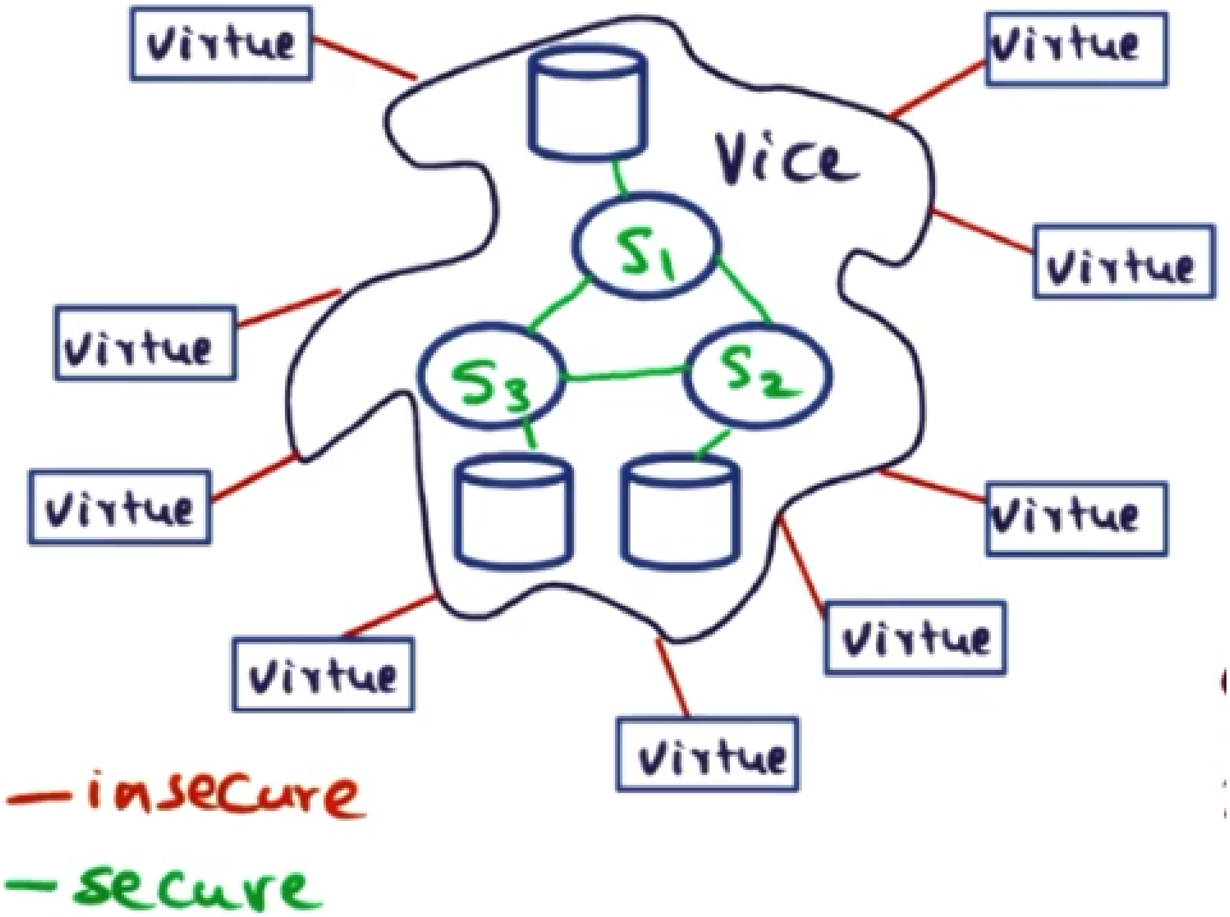
- Controlled sharing
  - Access lists associated with files
- User programmed sharing controls
  - Unix-like semantics for files (owner, group, everyone else)
- Strings on information
  - "TOP SECRET" need to deal with dynamics of use

5. Design Principles
   - Economy of mechanisms: Easy to verify whether it works or not
   - Fail safe defaults: Explicitly allow access -> default should not be no access (no way to guarantee information is protected)
   - Complete mediation: Security mechanism shouldn't take any shortcuts
     - Caching passwords is a bad idea
   - Open design: Publish the design but protect the keys
     - Must present keys to get information and breaking the keys should be computationally infeasible
     - Detection is easier than prevention (don't know what attacks are possible, so impossible to prevent all of them)
   - Separation of privelege: Two keys to open a vault
   - Least privilege: "Need to know" based controls
     - Require administrative priveleges for certain things
     - Origin for the idea of firewalls
   - Least common mechanism: Should mechanism be implemented in the kernel or as a library running on top?
   - Psychological acceptability: Good UI
   - Important takeaways
     - All of the principles are positive statements
     - All of the principles are still relevant to today's systems, despite being theorized at a time when computers weren't networked
     - Difficult to crack protection boundary (computationally infeasible)
     - Detection rather than prevention
6. Principles of Information Security Conclusion
   - Paper by SALSA is a classic
   - Visionary ideas, thought of at a time when computers were independent

Security in Andrew

1. Security in Andrew Introduction
   - Enable students at CMU to walk up to any workstation and use it (1988)
   - Access files from central server
   - Assume network was untrusted
   - Focus on security using a private key infrastructure
2. State of Computing Circa 1988
   - Local disks served as efficient caches
   - Vision of Andrew file system and Coda (both from CMU)
     - Walk up to any workstation and log in
     - Your content magically appears
     - Similar to cloud and mobile devices today
3. Andrew Architecture
   - Client workstation (virtues)
     - Unix
     - Connected to LAN through insecure network links
     - Servers within a "vice" can communicate securely
   - Venus
     - Process on each virtue for user authentication and client caching

    - Users use RPC to transfer files from vice to virtue
    - RPC must be secure for passing parameters and receiving results
  - Client communication with servers must be encrypted
  - Server communication with each other does not



Andrew Architecture

4. Encryption Primer
   - Private key system
     - Symmetric keys to encrypt and decrypt (e.g. passwords to login)
     - Publish the design but protect the key (SALSA)
     - Key distribution is difficult as the organization grows
     - Sender: data -> encrypt(data, key) -> cyphertext
     - Receiver: cyphertext -> decrypt(cyphertext, key) -> data
   - Public key system
     - Asymmetric keys (pair of keys)
     - Public key published -> Encrypt
     - Private key -> Decrypt
     - Mathematical basis is a one-way function
     - Sender: Data -> encrypt(data, public key) -> cyphertext
     - Receiver: Cyphertext -> decrypt(data, private key) -> data
5. Private Key Encryption System in Action
   - With private key encryption

    - A and B have exchanged keys
    - B needs to know identity of send to know which key to use
    - The identity of the sender is sent in cleartext
    - KeyA can be the same as KeyB

6. Challenges for Andrew System
   - Authenticate user
     - How can you verify the identity of the person logging in?
   - Authenticate server
     - How can you be sure the message is from the actual server?
   - Prevent replay attacks
     - A person sniffing packets from the wire shouldn't be able to resend a packet and fool the sender or receiver?
   - Isolate users
     - Community is shielded from the actions of other users
   - Andrew used a private key crypto system to protect RPC calls
     - Key distribution is not as big of a challenge
     - Identity of sender in cleartext
   - Traditional Unix: Username, password
     - Overuse of usernames and passwords results in a security hole
     - Violates the principle of protecting the keys
   - Dilemma: What to use as identity and private key?
7. Andrew Solution
   - Username and password only for login
   - Use ephemeral ID and keys for subsequent Venus-vice communication
   - Three classes of client-server interaction
     - Login: Username, password
     - RPC session establishment: Open communication with file system
     - File system access during session: Download files and work locally
   - 1 uses username and password
   - 2 and 3 uses ephemeral IDs and keys
8. Login Process
   - User logs in with username and password to login server
     - Authentication server is separate from authentication server
     - Login process returns cleartoken and secrettoken
     - This communication is secure
   - Cleartoken: Data structure
     - Extract handshake key client (HKC)
     - Use HKC as private key for establishing a new RPC session
   - Secrettoken: Cleartoken encrypted with key known only to vice
     - Encryption is different from the HKC
     - Unique for this login session (just a bitstream)
     - Use as ephemeral client-id for this login session
     - Only vice knows how to decrypt
   - Venus throws away clear and secret tokens at the end of the session
   - Bind mechanism is at the core of the Andrew file system
     - Used for establishing secure communication
9. RPC Session Establishment
   - After a user logs in, Venus establishes an RPC session on behalf of the client (bind client-server)
   - Client sends:
     - clientID, E[Xr,HKC]
     - Xr is a random number for each RPC
     - HKC is extracted from the cleartoken
   - Server sends:
     - E[(Xr+1,Yr),HKS] (HKC = HKS by design)

    - Xr+1 establishes that the server is genuine
    - Xr is encrpyted, so this authenticates the server and prevents replay attacks
    - Yr is another random number generated by the server
  - Client sends:
    - E[Yr+1,HKC]
    - Server checks if the value == Yr+1 to establish that the client is genuine and prevents replay attacks
  - Server only uses HKC for establishing a login session
    - Within an RPC session, a client may make many calls (open, read, write, close, etc)
    - After the server validates the ID of the client, it creates a session key (SK) and sends it to the client
  - Server sends:
    - E[(SK,num),HKS]
    - Client can extract the SK using its own HKC and use it for the duration of the session
    - num is the starting sequence number for RPC session (safeguard against replay attacks)
    - Use SK as handshake key for the rest of RPC session with server

10. Sequence Establishment
    - The sequence client(Xr) -> server(Xr+1) -> client establishes that the server is genuine
    - The sequence server(Yr) -> client(Yr+1) -> server establishes that the client is genuine
11. Login is a Special Case of Bind
    - Client/server validation is identical for login and bind
    - Password used as HKC
    - Username used for clientID
    - Get back two tokens
      - Secrettoken (encrypted with password)
      - Cleartoken (encrypted with password)
    - Tokens kept by Venus for this login session
12. Putting it All Together
    - Login using username and password
    - Vice sends secret and clear tokens (1)
    - Venus establishes an RPC session on behalf of the client using secret token and HKC
    - Vice sends session key for this particular RPC session (2)
    - Venus uses secret token and session key to make file system calls (3)
    - Upshots:
      - username, password are only exposed once per login session
      - HKC used only for new RPC session establishment
      - SK used for all RPC calls to file system (valid for duration of RPC session)
13. AFS Security Report Card
    - Mutual suspicion: Yes
      - Protection from fellow users
    - Protection from system for users: No
      - Users must trust system, so no protection
    - Confinement of resource usage: No
      - User can make many calls on server and consume bandwidth (DoS)
    - Authentication: Yes
      - Validation of client-server
    - Server integrity: No
      - Servers are within a firewall, so if somebody penetrated this, they could wreak havoc
      - Must physically and socially protect servers
14. Security in Andrew Conclusion
    - Main takeaway: How do OS designers make the best decisions for information security to make a secure and usable distributed system?
      - Able to compare against the principles outlined by SALSA
    - Introduces notion of audit chain for system administrators modifying the system

- Access lists for files with positive and negative rights