4/20/2009 – Upinder Malhi(cs162-at) and Edward Lin (cs162-eu)
Protection and Security - Jingtaow Wang

- Protection: To prevent accidental or intentional misuse of a system while permitting controlled sharing.
- Security: Use of protection mechanisms to prevent misuse of resources
    - Misuse defined with respect to policy
- Accidental and Intentional Misuse
    - Example: Program mistakenly overwrites the file used by the system shell. Nobody else can log in.
- Malicious Abuse
    - Example: Some high school brat who can't get a a date, so instead he transfers $3 billion from B to A.
- Three types of effects we are concerned with
    - Unauthorized information modification
    - Unauthorized denial of use
    - Unauthorized information release
- Other Protection Problems hard to address from O.S.'s perspective
    - Fake timesheets for paychecks.
    - Repeat button printer to print extra paychecks.
- Functional Levels of Information Protection
    - Unprotected System
    - All or nothing system
    - Controlled sharing
    - User programmed sharing controllers
        - Users want to put complex restrictions on use, such as time of day, or concurrence of another user.
- Design principles for protection mechanisms
    - Keep the design as simple and small as possible
    - Fail safe defaults
    - Complete mediation
    - Open design
    - Separation of privilege
    - Least privilege
    - Least common mechanism
    - Psychologically acceptability
- Three pieces to Security
    - Authentication – Who the user actually is
    - Authorization – Who is allowed to do what
    - Enforcement – Make sure people do only what they are supposed to do
- Authentication: Identifying Users
    - Q. How to identify users to system?
        - Passwords : Shared secret between two parties
            - Don't store the password directly. Store the hash of the password. Use salt to prevent dictionary attacks. Salts add

random strings to the end of the password and hash that, instead of the original password.
        - Ways of Compromising Passwords: Password Guessing(Often people use obvious information like birthday, favorite color, girlfriends's name, etc), Dictionary Attack (Work way through dictionary and compare encrypted version of dictionary words with entries in /etc/passwd) and Dumpters Diving (Find pieces of paper with passwords written on them)
        - Use changing passwords on keyboard logging attacks on public computer.
    - Smart Cards: Electronics embedded in card capable of providing long passwords or satisfying challenges.
    - Biometrics: use of one or more intrinsic physical or behavioral traits to identify someone
- Counter-actions
    - Passwords should not be stored in a directly-readable form
        - One-way function like a hash
    - Password testing should be slow
        - Limit the number of tests
    - Passwords should be relatively long and obscure
        - Paradox: short passwords are easy to crack, long passwords are easily forgotten and usually written down
    - Must protect the authorizer
        - The program that checks whether the password matches must be incorruptible
- Password Attacks
    - Recover 80% of the password just by hearing the sound of the keystrokes
    - You need to also protect your environment
- Using Badge or Key for Authentication
    - smart card is not usually a good design choice
        - pain to carry
        - key paradox: must be cheap to make, hard to duplicate
- Authorization:
- Access Control Matrix
    - each row represents a User, process, etc
    - each column is a file, devices
    - problem of the Access Control Matrix
        - if there are too many resources or files or users/processes, the matrix would become too big
        - it is not practical to implement this matrix in your operating system
- access control list
    - vertically split the matrix into several lists
    - most modern operating systems use this design for file system
    - really easy to implement
    - very easy to answer who has this file
    - very easy to revoke access

    - it is hard to answer the question: which files can be accessed by user A
    - how can you reduce number of entries in access control list?
        - you assign users to groups
    - overhead of checking an access list is “high”
        - if access list is in memory, takes 50-100 instructions
        - if stored on disk, it takes considerable time
- Capabillities
    - horizontal strips of the matrix
    - very easy to answer which files this user has access to
    - used by
        - Intel 423
        - cambridge CAP system
        - IBM system/38
    - Example: page tables
- Multics
    - 8 levels (rings of protection)
    - For each file and segment, there is a level associated with read, write, execute, and call
    - Protection levels are associated with segments and can be found in the segment tables
        - Accessed with little/no extra overhead
    - Intel x86 architecture supports the same thing, but with 4 levels
- psychological acceptability
    - random strings are good passwords but not good for humans
- access enforcement
    - hackers can control the system with a stack overflow to exploit a bug in a complex security system
    - you need to the system to be flexible
        - to follow a lot of control rules
    - you need the system to be reliable
        - simple systems are more reliable