Part1

Friday, May 31, 2019 11:59 AM

Security environment: Threats

- Operating systems have goals
  - Confidentiality
  - Integrity
  - Availability
- Someone attempts to subvert the goals
  - Fun
  - Commercial gain

Goal	Threat
Data confidentiality	Exposure of data
Data integrity	Tampering with data
System availability	Denial of Service

What kinds of intruders are there?

- Casual prying by nontechnical users
  - Curiosity
- Snooping by insiders
  - Often motivated by curiosity or money
- Determined attempt to make money
  - May not even be an insider
- Determined attempt to make mischief
  - Money typically…

Accidents cause problems too

- Acts of God
  - Fires
  - Earthquakes
  - Wars
- Hardware or software error
  - CPU malfunction
  - Disk crash
  - Program bugs
- Human errors
  - Data entry
  - Wrong tape mounted
  - Rm * .o

Protection

- Security is mostly about mechanism
  - How to enforce policies
  - Policies largely independent of mechanism
- Protection is about specifying policies
  - How to decide who can access what?
- Specifications must be

        - Correct
        - Efficient
        - Easy to use

Protection domains
- Three protection domains
    - Each lists objects with permitted operations
- Domains can share objects …
- …

Protection matrix
- Each domain has a row in the matrix
- Each object has a column in the matrix
- Entry for object has the permissions
- Who's allowed to modify the protection matrix?
    - What changes can they make
- How is this implemented efficiently?

Domains: objects in the protection matrix

Access control Lists
- Each object has a list attached to it
- List has
    - Protection domain
        - User range
        - Group of users
        - Other
    - Access rights
        - Read
        - Write
        - Execute (?)
        - Others?
- No entry for domain => no rights for that domain
- Operating system checks permissions when access is needed

Access control lists in the real world
- Unix file system
    - Access list for each file has exactly three domains on it
        - User (owner)
        - Group: set of users
        - Others
    - Rights include read, write, execute: interpreted for directories and files
    - Users may be in more than one group
- AFS (unix, ic)
    - Access lists only …
    - …

Capabilities
- Each process has a capability list
- List has one entry per object the process can access
    - Object name
    - Object permissions
- Objects not listed are not accessible

- How are these secured?
  - Kept in kernel
  - Cryptographically secured

Cryptographically protected capability
- Rights include generic rights (read, write, execute) and
  - Copy capability
  - Copy object
  - Remove capability
  - Destroy object
- Server has a secret (Check) and uses it to verify capabilities presented to it
  - Alternatively, use public-key signature technique

Protecting the access matrix: summary
- OS must ensure that the access matrix isn't modified (or even accessed) in an unauthorized way
- Access control lists
  - Reading or modifying the ACL is a system call
  - OS makes sure the desired operation is allowed
- Capability lists
  - Can be handled the same way as ACLs: reading and modification done by OS
  - Can be handed to processes and verified cryptographically later on
  - May be better for widely distributed systems where capabilities can't be centrally checked

Covert channels
- Circumvent security model by using more subtle ways of passing information
- Can't directly send data against system's wishes
- Send data using "side effects"
  - Allocating resources
  - Using the CPU
  - Locking a file
  - Making small changes in legal data exchange
- Very difficult to plug leaks in covert channels!

Covert channel using file locking
- Exchange information using file locking
- Assume n+1 files accessible to both A and B
- A sends information by
  - Locking files 0 … n-1 according to an n-bit quantity to be conveyed to B
  - Locking fie n to indicate that information is Available
- …
- …

Steganography
- Hide information in other data
- Picture on right has text of 5 Shakespeare plays
  - Encrypted, inserted into low order bits of color values
- …
- …

Cryptography
- Goal: keep information from those who aren't supposed to see it
  - Do this by "scrambling" the data
- Use a well-known algorithm to scramble data

    - Algorithm has two inputs: data and key
    - Key is known only to "authorized" users
    - Relying upon the secrecy of the algorithm is a very bad idea such as the WW2 Enigma
- Cracking codes is very difficult, Sneakers and Swordfish and other movies notwithstanding

Cryptography basics
- Algorithms (E, D) are widely known
- Keys (KE, KD) should be less widely distributed
- For this to be effective, the ciphertext should be the only information that's available to the world
- Plaintext is known only to the people with the keys (in an ideal world…)

Secret key encryption
- Also called symmetric key encryption
- Monoalphabetic substitution
    - Each letter replaced by different letter
- Vignere cipher
    - Use a multi character key
      THEMESSAGE
      ELMELMELME
      XSQQPEWLSI
- Both are easy to break!
- Given the encryption key, easy to generate the decryption key
- Alternatively, use different…

Modern encryption algorithms
- Data Encryption Standard
    - Uses 56-bit keys
    - Same key is used to encrypt and decrypt
    - Keys used to be difficult to guess
        - Needed to try 2^55 different keys, on average
        - Modern computers can try millions of keys per second with special hardware
        - For $250k, EFF built a machine that broke DES quickly
- Current algorithms (AES, Blowfish) use at least 128 bit keys
    - Adding one bit to the key makes it twice as hard to guess
    - Must try 2^127 keys on average to find the right one
    - At 1-^15 keys per second, this would require over 10^21 seconds, or 1000 billions years
    - ….

Unbreakable codes
- There is such a thing as an unbreakable code: one-time pad
    - Use a truly random key as long as the message to be encoded
    - XOR the message with the key a bit at a time
- Code is unbreakable because
    - Key could be anything
    - Without knowing key, message could be anything with the correct number of bits in it
- Difficulty: distributing key is as hard as distributing message
    - May be easier because of timing
- Difficulty: generating truly random bits