ECS 235B Module 35 Originator-Controlled Access Control

Originator Controlled Access Control

- Problem: organization creating document wants to control its dissemination
  - Example: Secretary of Agriculture writes a memo for distribution to her immediate subordinates, and she must give permission for it to be disseminated further. This is “originator controlled” (here, the “originator” is a person).

Requirements

- Subject $s \in S$ marks object $o \in O$ as ORCON on behalf of organization X. X allows o to be disclosed to subjects acting on behalf of organization Y with the following restrictions:
  1. o cannot be released to subjects acting on behalf of other organizations without X's permission; and
  2. Any copies of o must have the same restrictions placed on it.

DAC Fails

- Owner can set any desired permissions
  - This makes 2 unenforceable

MAC Fails

- First problem: category explosion
  - Category *C* contains *o*, *X*, *Y*, and nothing else. If a subject $y \in Y$ wants to read *o*, $x \in X$ makes a copy *o′*. Note *o′* has category *C*. If *y* wants to give $z \in Z$ a copy, *z* must be in *Y*—by definition, it's not. If *x* wants to let $w \in W$ see the document, need a new category *C′* containing *o*, *X*, *W*.
- Second problem: abstraction
  - MAC classification, categories centrally controlled, and access controlled by a centralized policy
  - ORCON controlled locally

Combine Them

- The owner of an object cannot change the access controls of the object.
- When an object is copied, the access control restrictions of that source are copied and bound to the target of the copy.
  - These are MAC (owner can't control them)
- The creator (originator) can alter the access control restrictions on a per-subject and per-object basis.
  - This is DAC (owner can control it)

Digital Rights Management (DRM)

- The persistent control of digital content
- Several elements:
  - Content: information being protected
  - License: token describing the uses allowed for the content
  - Grant: part of a license giving specific authorizations to one or more entities, and (possibly) conditions constraining the use of the grant
  - Issuer: entity issuing the license
  - Principal: identification of an entity, used in a license to identify to whom the license applies
  - Device: mechanism used to view the content

Example: Movie Distribution by Downloading

- Content: movie itself
- License: token binding palying the movie to the specific downloaded copy
- Grant: movie can be played on some specific set of equipment provided the equipment is located in a geographical area
- Issuer: movie studio
- Principal: user who downloaded the movie
- Device: set of equipment used to play the movie; it manages the licenses, principle, and any copies of the movie

Relationships

Elements related, and the relationship must satisfy all of:

1. The system must implement controls on the use of the content, constraining what users can do with the content
   - Encrypting the content and providing keys to authorized viewers fails this, as the users can distribute the keys indiscriminently
2. The rules that constrain the users of the content must be associated with the content, not the users
3. The controls and rules must persist throughout the life of the content, regardless of how it is distributed and to whom it is distributed

Conditions

- Stated using a rights expression language
- Example: Microsoft's PlayReady uses a language supporting temporal constraints such as
  - Allowing the content to be viewed over a specific period of time
  - Allowing a validity period for the license
  - Allowing constraints on copying, transferring, converting the content
  - Allowing geographical constraints
  - Allowing availability constraints (for example, content can't be played when being broadcast)

Example: Microsoft PlayReady DRM

Setup

- Content is enciphered using AES
- Key made available to a license server, encrypted content to a distribution server

Play

- Client downloads content, requests license
- License server authenticates client; on success, constructs license and sends it
- Client checks the constraints and, if playback allowed, uses the key in the license to decipher content

Example: Apple's FairPlay DRM

Set up system to play using iTunes

- iTunes generates globally unique number, sends it to Apple's servers
- Servers add it to list of systems authorized to play music for that user
  - At most 5 systems at a time can be authorized

Obtain content using iTunes

- Content enciphers by AES with a master key
- Master key locked with a randomly generated user key from iTunes
- iTunes sends user key to Apple server; stored there and in iTunes, encrypted

Example: Apple's FairPlay DRM

Play content using iTunes

- iTunes decrypts user key
- iTunes uses user key to decrypt master key
- iTunes uses master key to decrypt content
- Note it need not contact Apple servers for authorization

Authorize new system

- Apple server sends that system all user keys stored on server

Example: Apple's FairPlay DRM

Deauthorize system

- System deletes all locally stored user keys
- Notifies Apple servers to delete globally unique number from list of authorized computers

Copying content to another system

- Cannot be decrypted without user key, which is not copied

Oops …

- Sony BMG developed rootkit to implement DRM on a music CDs
  - Only worked on Windows systems; users had to install a proprietary program to play the music
  - Also installed software that altered functions in Windows OS to prevent playing music using other programs
  - This software concealed itself by altering kernel not to list any files or folders beginning with “sys” and storing its software in such a folder
  - On boot, software contacted Sony to get advertisements to display when music was played
  - Once made public, attackers created Trojan horses with names beginning with “sys (like “sysdrv.exe”)
- Result: lawsuits, flood of bad publicity, and recall of all such CDs