Recent security features and issues in embedded systems

NEC OSS Promotion Center
KaiGai Kohei <kaigai@ak.jp.nec.com>
Asset and Vulnerability should be considered as a pair.

Threat intend to attack vulnerabilities.

Risk means possibilities the threat to be actualized.
- more worthwhile asset, weaker vulnerability has its risk grow.
- risk depends on environmental factors, organization policy.

Security is a way to reduce risk.
Evolutions of Cellular-Phone in the last 20 years

The first generation cellular-phones (1987)

Value of Assets ↑
Vulnerability ↑

electronic money, Credit card

Internet connectivity (Web, mail, ...)

commodity operating system

Java application

Mass storage & Data exchange

stored privacy, individual info
Evolutions of Video/Television in the last 20 years

Analog Video/Television
20 years ago

Value of Assets ↑
Vulnerability ↑

20 years later

Internet connectable home server
Privacy

Online firmware update
Mass storages to store movie contents

commodity operating system
Charging Information
Evolution of OS security in the last 20 years

- What kind of security features are currently available within the operating system?
  - SELinux, Cryptograph, ACL, ...
- Can we apply them on embedded systems?
- Is it really comprehensive?

Traditional UNIX permission

[kaigai@mailsv ~]$ ls -l /etc
-rw-r--r-- 1 root root 23954 Apr  1 18:21 /etc/aliases
-rw-r--r-- 1 root root 17 Jul 24 2000 /etc/host.conf
-rw-r--r-- 2 root root 147 Nov 30 2005 /etc/hosts
-rw-r--r-- 1 root root 161 Jan 13 2000 /etc/hosts.allow
-rw-r--r-- 1 root root 347 Jan 13 2000 /etc/hosts.deny
-rw-r--r-- 1 root root 1666 Aug 29 2005 /etc/inittab
-rw-r--r-- 1 root root 5553 Mar 19 08:46 /etc/passwd
drwxr-xr-x 10 root root 4096 Nov 13 14:10 /etc/rc.d/
-r-------- 1 root root 5563 Mar 19 08:46 /etc/shadow
          :          :          :          :          :          :          :
Today's Topics

- Sorting out security requirements
  - ISO15408 framework makes it clearly categorized.
  - Recent security futures can be mapped on the categories.

- Recent security feature and its issue
  - Introductions/Overviews of recent security features
  - Possible difficulty in applying security features to embedded systems
  - Why? differences in CPU, Filesystem, Memory size, ...

- This session shows the issues to be resolved when we apply recent security features on embedded systems.
ISO/IEC15408

The purpose of ISO/IEC15408
- Common criteria to evaluate security functionalities of IT products, not only software

Essentials
- more than 20 years history since TCSEC, ITSEC
- 11 functional categories, 8 assurance ones
  - Developers can select functional ones suitable for them.
- We can use it as a comprehensive catalog of security functionalities.
Protection Profile

What is Protection Profile?
- A set of requirements for specific categories
  - OS, RDBME, Firewall, SmartCard, Digital-Copier, etc...
- Example:
  - CAPP, LSPP, MLOSPP for OS, DBMSPP for RDBMS

ISO/IEC15408 Functional Requirements
- Security Audit
- Cryptographic Support
- Identification and Authentication
- Security Management
- Protection of the TSF
- TOE Access
- Communication
- User Data Protection
- Privacy
- Resource Utilization
- Trusted Path/Channels

Labeled Security Protection Profile
- Security Audit
- User Data Protection

DBMS Protection Profile
- Security Audit
- User Data Protection
- Security Management
Technology map of Recent Security Features

- Re-organized major categories of security functionalities required by protection profiles for operating system.
Security Audit

What is the purpose?
- To confirm what has happened later, when we get security incidents
  - Unified event logging for both operating system and applications
  - Well formalized audit logs, In-kernel event filter
- To alert system administrators

Related components
- linux-audit
linux-audit features (1/2)

- Designed for ISO15408 requirements
  - System call audit
  - In-kernel event filters
  - Selective audit review
  - Event notification for administrators
- Utilities
  - auditd
  - auditctl
  - ausearch, aureport
linux-audit features (2/2)

```
[root@saba ~]# auditctl -a exit,always -S open  
      -F path=/etc/shadow -F exit!=0
[root@saba ~]#

[kaigai@saba ~]$ less /etc/shadow
/etc/shadow: Permission denied
[kaigai@saba ~]$

[root@saba ~]# ausearch --file /etc/shadow
----
time->Mon Mar 24 19:19:05 2008
  type=PATH  msg=audit(1206353945.330:3541): item=0 name="/etc/shadow"
      inode=6111049 dev=08:06 mode=0100400 uid=0 ogid=0 rdev=00:00
      obj=system_u:object_r:shadow_t:s0
  type=CWD   msg=audit(1206353945.330:3541):  cwd="/home/kaigai"
  type=SYSCALL msg=audit(1206353945.330:3541): arch=40000003 syscall=5
      success=no exit=-13 a0=806d010 a1=8000 a2=0 a3=8000 items=1
      ppid=3638 pid=18234 auid=1001 uid=1001 gid=1001 euid=1001
      suid=1001 fsuid=1001 egid=100 sgid=100 fsgid=100 tty=pts6 ses=52
      comm="less" exe="/usr/bin/less"
      subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
      key=(null)
```
**linux-audit issues in embedded systems**

- **Architecture Limitation**
  - It hooks the entry-point of system call
    - Implemented in assembler code
  - **Unsupported architectures**
    - Only x86, ppc, ppc64, s390, ia64, UML, sparc64 are supported now.
    - Super-H coming soon (2.6.25)
    - Where is ARM, MIPS?

- **Storage size limitation**
  - smaller storage than server/desktop
Cryptographic Support

What is the purpose?
- Crypto-key operations based on standard algorithms.
- Encryption/Decryption based on standard algorithms.

Related components
- dm-crypt
- eCryptFS
  - based on in-kernel crypt API
dm-crypt/ eCryptFS features

- **dm-crypt**
  - One of the device-mapper modules
  - Works in Block Layer
  - Per-device encryption

- **eCryptFS**
  - One of the pseudo filesystem works as NFS doing
  - Per-file encryption
    - Metadata as contains of files
    - Encryption is done before compression in jffs2
dm-crypt/ eCryptFS issues in embedded systems

- dm-crypt and MTD devices
  - I/O traffic on MTD devices don't go through block layer.
  - Cryptographic Support on JFFS2, LogFS and so on?

- Feasible ideas
  - Utilization of JFFS2 compress handlers
  - Pseudo cryptographer device on MTD
Data Protection

- Essentials
  - Access Control
  - Data Flow Control
  - Data Integrity

- What is the purpose?
  - To protect data (including metadata) from leaking, manipulation and destruction.

- Related components
  - POSIX ACL
  - SELinux

[Diagram with various components related to security and data protection]
POSIX ACL

- The limitation in traditional permission model
  - 'rwx' permission for owner, a group and others

- How POSIX ACL works
  - It stores ACL within xattrs, used in permission checks.

- Default ACL supports

- POSIX ACL Issues in embedded systems
  - XATTR supports in filesystems are needed.
  - Busybox supports are needed.
SELinux (1/3)

- What is the purpose of SELinux?
  - Mandatory Access Controls (MAC)
  - Data Flow Controls (DFC)

Methods of inter-processes communication

Security policy checks
SELinux (2/3)

- **Features**
  - It associates a security attribute for each subject/object.
    - e.g) "root:object_r:var_log_t:Unclassified"
  - SELinux hooks any system-call invocation to apply its decision based on its security policy.

- **Why we need SELinux?**
  - root can be a single-failure point
  - Fine-grained access control
  - A single unified security policy
  - Generally, fewer checks are better
  - If SELinux is disabled?
    - Massive checks in userspace
    - or, Quality degrading
SELinux (3/3)

- SELinux issues on embedded systems
  - Filesystem XATTR support
    - Security context of files are stored within xattr field.
    - cramfs, LogFS, yaffs, etc...
  - Userland utilities support
    - Now busybox has 12 applets, 12 extensions for SELinux
      - load_policy, setenforce, restorecon, ...
      - '-Z' option support, preserving security context, ...
    - Libselinux provides fundamental facilities to userland.
    - PAM_Selinux.so associate a user with its security context.
- Security Policy
  - Different application, environment from server/desktop
  - Distributors should provide its suitable base security policy
Identification and Authentication

What is the purpose?
- To ensure a process works with correct identifier.
- To associate a user with correct authorities.
- These features are foundation for other security facilities to work correctly.
  - Security-Audit, Access Controls, ...

Related components
- POSIX Capabilities
- PAM
POSIX Capabilities (1/4)

The purpose of POSIX Capabilities
- Least privileged set

Features
- It enables to associate a part of 'root privileges'
  - E.g) Using network port < 1024, Ignoring DAC permission
- Linux kernel has this feature from 2.4.x series, however, it has been hard to utilize.

Recent updates
- Filesystem POSIX Capability
- Per-process Capability Bounding Set
POSIX Capabilities (2/4)

The calculation rule of capabilities on execve()

\[
P'(\text{permitted}) = (F(\text{permitted}) \& \text{cap\_bset}) \\
\quad \big| (P(\text{inheritable}) \& F(\text{inheritable}))
\]

\[
P'(\text{effective}) = F(\text{effective}) \, ? \, P'(\text{permitted}) : 0
\]

\[
P'(\text{inheritable}) = P(\text{inheritable})
\]

- Pseudo File POSIX capability bits
  - If euid = 0 → F(*) is set to All-1 (0xfff...fff)
  - If euid != 0 → F(*) is set to All-0 (0x00...00)
  - We had no way to represent F(*) bitmasks.

- Filesystem POSIX Capability stores F(*) information within xattr of executable files.
  - We can run privileged programs with more restricted power.
  - E.g) /bin/ping with only CAP_NET_RAW
POSIX Capabilities (3/4)

The calculation rule of capabilities on execve():

\[
\begin{align*}
P'(\text{permitted}) & = (F(\text{permitted}) \& \text{cap_bset}) \\
& \quad | (P(\text{inheritable}) \& F(\text{inheritable})) \\
P'(\text{effective}) & = F(\text{effective}) ? P'(\text{permitted}) : 0 \\
P'(\text{inheritable}) & = P(\text{inheritable})
\end{align*}
\]

Example

- /bin/ping with CAP_NET_RAW on F(permitted), not SetUID'ed
  - \( P'(\text{permitted}) = \text{CAP_NET_RAW} \& 0xff\text{fff...fff} \mid 0 \& 0 \)
  - \( P'(\text{effective}) = (\text{true}) ? \text{CAP_NET_RAW} : 0 \)

Features

- It stores F(*) bits within filesystem XATTR region
- It enables to replace SetUID programs.
- Available on 2.6.24 or later
Features

- Capability bounding-set can mask root privileges
- \( F(\text{permitted}) \) is 0xffff...fff when euid = 0
- In 2.6.24 or former, \( \text{cap\_bset} \) is system wide variable
  - In the next kernel, we can set per-process capability bounding set, as follows.

\[
\begin{align*}
P'(\text{permitted}) &= (F(\text{permitted}) \& \text{cap\_bset}) \\
&\quad \mid (P(\text{inheritable}) \& F(\text{inheritable})) \\
P'(\text{effective}) &= F(\text{effective}) \ ? P'(\text{permitted}) : 0 \\
P'(\text{inheritable}) &= P(\text{inheritable})
\end{align*}
\]
PAM

PAM (Pluggable Authentication Module)
- A framework of authentication modules

PAM Issues in Embedded systems
- Widely used to set up initial users security attribute
  - security context of SELinux
  - per-process capability bounding set
- It means these 'secure initial state' depends on PAM
  - How tinylogin handle it?
Communication & Others

What is the purpose?
- To provide secret, trusted and separated communication channel
- Resource utilization, trusted timestamp, ...

Related components
- SSH/SSL/PGP
- IPsec
  - Sensitivity and Integrity on communication channels
- Cgroups/-rt kernel
  - Resource availabilities are also required to security aspect

Give us issues in this region, if you have anything.
Embedded Linux Conference 2008

Summary (1/2)

- Asset
- Vulnerability
- Risk
- Security
- Threat

Vulnerability ↓
Risk ↓

- Security Audit
- Data Protection
- Identification & Authentication
- Cryptographic
- Others
- Communication
Summary (2/2)

- These issues should be solved to provide 'secure' embedded systems.

- Filesystem XATTR support
  - Common requirement for selinux, ACL, capabilities
  - It is now available on most of regular filesystems, but ...

- Utilities support
  - busybox
    - Now, about 70% of SELinux utilities are merged into upstream
    - Features of ACLs and capabilities are also necessary

- Cryptograph support on MTD devices
- Linux-audit support for embedded architecture
Any Question?
Thank you!