Utilizing SELinux in Embedded Devices

Chad Sellers
Lead Software Architect
Product Solutions Group
Tresys Technology
Outline

• Background on SELinux
• Useful things you can do with SELinux
  • examples of what you can do with SELinux
  • flexibility of SELinux
• Useful features
  • Dynamic policy
  • Structured policy
  • Policy management
  • Policy development
What is SELinux?

• Enhancement to Linux Kernel and utilities
  • not a stand-alone distribution
  • available as part of major distributions
• Incorporates flexible mandatory access control
  • primarily through type enforcement
  • cooperates with other security enhancements (execshield)
• Provides a rich and flexible MAC policy language
• Maintains binary compatibility for programs
  • unmodified Linux applications can be controlled
Mandatory Access Controls (MAC)

- Access Control in general
  - subjects and objects have security attributes
  - access determined based on policy rules
- Discretionary Access Control
  - users can change security attributes at request
  - allowing programs running on behalf of a user to affect the results of access rules
- Mandatory Access Control
  - users cannot change security attributes at request
  - user programs must work within the constraints of rules
  - MAC rules are controlled by the organization, not the user
Type Enforcement

- A type is an unambiguous identifier
  - created by the policy writer
  - applied to all subjects and objects and for access decisions
- Types group subjects and objects
  - signifies security equivalence
  - everything with the same type has the same access
  - policies have as few or as many types as needed
Type Enforcement

- Access specified between
  - subject type (e.g., process or domain)
  - and object type (e.g., file, dir, socket, etc.)
- Four elements in defining allowed access
  - source type(s) aka domain(s)
  - target type(s) objects to which access allowed
  - object class(es) classes to which access applies
  - permission(s) type of access allowed
- SELinux prevents access unless explicitly allowed
Where is SELinux being used

- Distributions
  - Fedora 2-7
  - Red Hat Enterprise Linux 4 and 5
  - Hardened Gentoo
  - Debian Etch
  - Montavista
  - others
- Government applications
- Regular users
- Embedded applications
Solving your security problems

- SELinux is useful for solving your security problems
- Examples
  - protecting data from disclosure
  - limiting attack vectors to an application
  - protecting data integrity
  - containing untrusted programs
Protection from disclosure

- Most applications do not need access to important data
  - even those that do typically only need a limited set of files
- SELinux can be used to limit access to important data to only those processes that need it
  - give important files a special type
  - only allow access to those processes that need it
- Reduces trust placed in applications
Protection from disclosure

- Example
  - cell phone application vulnerable to rogue network
  - attacker masquerades as trusted server
  - attacker compromises application to pilfer data
- SELinux can
  - limit access of application to only what it needs
  - this limits the damage of the exploit
- Demo
Limiting attack vectors

• Attackers are constantly finding new attack vectors
  • fixing yesterday’s hole is insufficient
• Developers must reduce attack surface
  • unnecessary access should be blocked
  • unused access vectors are asking for trouble
Limiting attack vectors

- Simple example - firewall
- SELinux network controls
  - firewall at the process level
- Example
  - update application
    - requires significant permission to perform update
    - need to reduce attack surface
  - limit network access to your corporate IP space
    - only limit for the update application
    - other applications can still access what they need
- Policy demo
Protecting data integrity

- `/etc/passwd`
  - pervasive need to read
  - integrity of this file is very important
- Anyone who can modify this file can
  - change their UID to 0
  - lock out another user
  - etc.
Protecting data integrity

- SELinux can
  - limit write access to /etc/passwd
    - /usr/bin/passwd needs access
  - limit who can run /usr/bin/passwd
  - control passwd’s power based on who launched it
    - not really useful in this example though
- Still have to trust application to do its job
  - /usr/bin/passwd can write to /etc/passwd
  - trust it to only change the appropriate user line
Protecting data integrity

login

fork()

bash

execve()

passwd

/etc/passwd

/etc_passwd_t

write(), create(), unlink()

allow passwd_t etc_passwd_t : file { write create unlink };

user_t

passwd_t

passwd_exec_t

/usr/bin/passwd
Containing untrusted programs

- Often need to run programs we don’t trust
  - mobile code downloaded to the device from an untrusted source
  - code written by another company
  - code written by that guy that you don’t trust
- Running these programs without restriction is too risky
- Sandboxing
  - common technique for containing untrusted code
  - most common methods tend to be
    - heavyweight
    - inflexible
  - SELinux can be used as a lightweight flexible sandbox mechanism
Containing untrusted programs

- Example
  - device allows users to download and execute code from arbitrary sources
  - basic common functionality is known a priori
- SELinux can be used to
  - create multiple sandbox buckets
  - allow basic functionality for those buckets
    - access network
    - perhaps access display
- Walls of the sandbox are determined by the policy author
- No additional infrastructure necessary
  - apps utilize the normal system filesystem as permitted by the policy
  - apps run as normal processes, no need for additional supervisor layer
Useful features

- Conditional policy
  - run-time policy change
- Reference Policy
  - structured policy to build on
- Policy management
  - because things always change
- Policy development
  - tools to build your policies
Conditional policy

- Blocks of policy controlled by a boolean expression
- Booleans can be set at run-time
  - using setsebool command, libselinux API, or selinuxfs pseudo-filesystem directly
  - access control is applied to the setting of booleans
- Kernel policy enforcement based on boolean settings
- Useful for operating differently under different modes
  - threat-based
  - configuration-based
Reference Policy

• A new SELinux policy that
  • reduces the complexity of writing, maintaining, and analyzing policy
  • uses modern software engineering principles
  • is well documented, modular, and configurable
  • provides a single source for all the policy variants

• Together this will make a policy that is…
  • maintainable
  • verifiable
  • usable
Layer: admin

Policy modules for administrative functions, such as package management.

<table>
<thead>
<tr>
<th>Module</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>acct</td>
<td>Berkeley process accounting</td>
</tr>
<tr>
<td>alsal</td>
<td>Aint ALSA configuration tool</td>
</tr>
<tr>
<td>amanda</td>
<td>Automated backup program.</td>
</tr>
<tr>
<td>anaconda</td>
<td>Policy for the Anaconda installer.</td>
</tr>
<tr>
<td>apt</td>
<td>APT advanced package toll.</td>
</tr>
<tr>
<td>backup</td>
<td>System backup scripts</td>
</tr>
<tr>
<td>bootloader</td>
<td>Policy for the kernel modules, kernel image, and bootloader.</td>
</tr>
<tr>
<td>certwatch</td>
<td>Digital Certificate Tracking</td>
</tr>
<tr>
<td>consoletype</td>
<td>Determine of the console connected to the controlling terminal.</td>
</tr>
<tr>
<td>ddcprobe</td>
<td>ddcprobe retrieves monitor and graphics card information</td>
</tr>
<tr>
<td>dmesg</td>
<td>Policy for dmesg.</td>
</tr>
<tr>
<td>dmidecode</td>
<td>Decode DMI data for x86/a64 bioses.</td>
</tr>
<tr>
<td>dpkg</td>
<td>Policy for the Debian package manager.</td>
</tr>
</tbody>
</table>
Management tools

- **libsemanage**
  - library for policy management
  - provides ability to make changes to the policy
- **Tools built on libsemanage**
  - semodule (policy modules)
  - semanage
- **Commercial applications**
  - Tresys Brickwall
Policy development tools

- SLIDE
  - open source
  - basic policy editor for use with Reference Policy
- Tresys Bedrock
  - commercial
  - Integrated tool for development and debugging
- others
Conclusion

- SELinux can probably address your security problems
  - flexible enough to address many situations
  - must understand the limits of SELinux and security architecture to fully utilize
  - not just for government high-security needs
- SELinux is getting easier and more useful
  - solid foundation
  - tools developing rapidly to increase usability
Links

- http://www.tresys.com
- http://www.usefulsecurity.com
- http://securityblog.org
- csellers@tresys.com