Towards Intuitive Tools for Managing SELinux

Hiding the Details but Retaining the Power

James Athey, Chris Ashworth, Donald Miner, and Frank Mayer
Tresys Technology

Laying the Foundations

- SELinux runs in Enforcing mode by default
  - on RHEL since version 4
  - on Fedora Core since version 3
- Broadly applicable policy
  - To “just work”, the policy has to allow
    - Most network access
    - Common use cases and file system layouts

Current Management Infrastructure

- One policy does not fit all installations
- What if the system doesn't work?
- Booleans
  - Each target can be unconfined ("disable_trans")
  - Some targets have behavior tweaks
- SEManage
  - Label network resources and files with existing types
  - Cannot create new types
  - Hope that nothing breaks ☺
- Or just turn SELinux off ☹

Current Mgmt. Infrastructure, cont.

- What if the policy is too permissive?
  - The policy is not tailored to individual networks
  - Untargeted software runs in unconfined
    - Some custom software will never become part of the policy
- Solution: Write SELinux policy?
  - Buy “SELinux By Example”
  - Attend a Tresys Technology SELinux course
  - Download SLIDE
- Network administrators have more important things to do

Making SELinux Work For You

- Have a computer do the hard work
- Manageable policy features
  - The administrator describes security requirements
  - Configurable
  - Extensible
  - Deployable
- Experience developing Tresys Brickwall Security Suite

TRESYS TECHNOLOGY

Describing Security Requirements

- A GUI is a must!
  - Administrators are not software developers
- Labeled resources are foreign to many administrators
  - Accustomed to firewalls, DAC perms
- Easier to think of access to things, not types
  - The labeling abstraction gets in the way
- Many resources have well-known meanings and properties
  - 10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16 are all private addresses
  - /bin/*, /usr/bin/* are all user-executable binaries

Describing Security Requirements, cont.

- Group concrete resources into definitions
  - Name, description, and contents shown together
    - Reinforce the association between the definition and the resources
- Creating new definitions
  - Only the user knows the details of the system and the environment
- Programs get access to definitions

Configuring and Customizing Policy

- For access to be configurable, it cannot be hardcoded in the policy
  - Remove access to managed resources from base policy
- Software takes care of labeling managed resources
  - Remove labeling statements from base policy
- One solution: generate “allow” rules for each program to each designated resource
  - Breaks encapsulation
  - Lots of rules to generate
  - Base policy must be heavily altered
    - Lose ability to remain in-sync with upstream policy

Configuring and Customizing, cont.

- Better solution: replace access to types with access to attributes
  - Build one module
    - Creates types to apply to resources
    - Puts types into attributes that programs already have access to
    - Updating policy is fast! (~15 seconds instead of 5 minutes)
  - Use SEManage to label resources
    - No reboot required
    - No need to recompile base module

Extending Policy

- End-users have custom software
  - Will never become part of upstream policy
  - Runs in unconfined, so SELinux provides little protection
- Start with a template
  - Common sense default file permissions
    - What lots of programs need, low risk to grant
    - Terminals, /tmp, shared libraries, /proc, /dev/random
  - If access cannot be easily configured, just grant it
    - IPC, dbus, filesystem perms
- Already an improvement over running unconfined
  - SELinux policy is protected
  - Cannot sniff packets on the network unless specifically noted

Extending Policy, cont.

- Define the entrypoint
  - Specify exact path
  - Only files that aren't already labeled *_exec_t
- Provide well-organized, configurable access
  - Networking
  - Files
  - POSIX Capabilities
  - Execution Privileges (execmem, execheap)
  - Access to syslog, RPM, mount

Deploying Policy

- One policy does not fit all computers in an installation
  - Machines used by people with different levels of access
  - Machines serve different purposes
- Many installations are very large
  - Lots of work to configure SELinux on many machines

Deploying Policy, cont.

- Make groups of machines based on security requirements and OS
  - All machines in the group use the same security configuration
- Send security configuration, not policy
  - Each machine runs a daemon that manages policy
  - Generate policy on each machine
  - Central manager does not need to know how to build policies for each kind of client
- Protect machines using public-key crypto
  - Machines only accept connections from manager with matching private key
  - No need to exchange usernames and passwords
  - Secret private key stored in one place – at manager

TRESYS TECHNOLOGY

QUESTIONS?