SELinux Update
Karl MacMillan, Tresys
Dan Walsh, Red Hat
Agenda

- Brief introduction to SELinux
- Example blocked exploits
- SELinux community
- Extending SELinux into applications
- Extending SELinux across the network
- Policy writing usability
- Confining users
SELinux Introduction

- Flexible Mandatory Access Control (MAC)
  - developed by NSA, extended by community
  - confines applications, including root
  - flexible policy language meets many security goals

Goal: provide security in the face of exploits
- engineer systems to be secure even with flaws
- place trust burden on SELinux, not applications

- SELinux is proactive, not reactive security
Example: Dowd’s Flash Exploit

- Cross-platform Flash vulnerability
- SELinux can stop this in two ways:
  - executable memory controls
  - NSPluginWrapper: run plugins in another process
- Separating plugins in a separate process:
  - plugins are given separate access from Firefox
  - access to network, home dir, etc. is removed
  - without loss of functionality
SELinux Community Updates

- SELinux developed by an active community
  - including support from many organizations
- Available in many distributions
  - Fedora, RHEL, Debian, Gentoo
  - and recently Ubuntu
- Also available in other Unixes
  - TrustedBSD, SEDarwin (basis for MacOS Seatbelt)
  - and recently FMAC from OpenSolaris
- Even available in Xen (XSM)
Improved Upstream Resources

- Revamp of selinuxproject.org
  - developer and user documentation (wiki)
  - project hosting for developers
- selinuxnews.org continues to expand
  - community news about selinux
  - SELinux Planet (blogs)
- Upstream code moved to oss.tresys.com
  - bug tracking, scm, releases
  - ideally will encompass resources beyond SELinux
Extending SELinux into Applications

- Many applications maintain private resources
  - e.g., databases, X server, D-Bus
  - resources are not visible to operating system
- Operating system cannot control access
  - typically, applications include access control
  - varying models, poor integration with system
- SELinux can be extended into applications
  - called “userspace object managers”
  - password was the earliest example
SE-Postgres and XACE

- **SE-PostgreSQL**: SELinux enabled PostgreSQL
  - fine-grained control over data and procedures

- **XACE**: access control for X.org
  - [http://people.freedesktop.org/~ewalsh/xace_proposal.html](http://people.freedesktop.org/~ewalsh/xace_proposal.html)
  - access control “hooks” – similar to LSM
  - merged into upstream X.org
  - SELinux hook implementation available
Extending SELinux Across the Network

- Labeled networking
  - labeled-ipsec: transfer process labels
  - secmark: local labeling of network traffic
  - both available in current distributions

- Labeled NFS
  - current NFS support has coarse-grained labeling
  - protocol extension for label transfer
  - currently in active development
Why Are Policy Tools Needed

- Policy tools goal: improve SELinux usability
  - but what is usability?

- Usability is a single metric for a multi-faceted problem
  - often shortened as “make it simple to use”
  - if simplicity were it, software usability would be solved
  - sad results: “clippy” the office assistant

- Inherit tension between expressiveness and simplicity
  - or perhaps expressiveness and initial learning

- Simple applications fail when faced with new situation
  - often caused by “leaky abstractions”
SELinux Usability

- Linux success based on expressiveness and power
  - yields flexibility and simplicity
  - however, the trade-off is not always acceptable
- Microsoft is copying
  - re-introducing the command line for administration
- Alternate goal to simplicity: help users attain goals
  - what users?
  - what goals?
- One possible answer:
  - users: admins disabling SELinux
  - goal: make their systems function correctly
Why Do Admins Disable SELinux?

type=AVC msg=audit(1173876205.535:130): avc: denied { read } for pid=8266 comm="firefox" name="ld.so.cache" dev=dm-0 ino=18874522 scontext=user_u:system_r:firefox_t:s0 tcontext=user_u:object_r:ld_so_cache_t:s0 tclass=file
type=SYSCALL msg=audit(1173876205.535:130): arch=40000003 syscall=5 success=yes exit=3 a0=4a569217 a1=0 a2=0 a3=ffffffff items=0 ppid=7967 pid=8266 auid=500 uid=500 gid=500 euid=500 suid=500 fsuid=500 egid=500 sgid=500 fsgid=500 tty=pts2 comm="firefox" exe="/bin/bash" subj=user_u:system_r:firefox_t:s0 key=(null)
type=AVC msg=audit(1173876205.535:131): avc: denied { getattr } for pid=8266 comm="firefox" name="ld.so.cache" dev=dm-0 ino=18874522 scontext=user_u:system_r:firefox_t:s0 tcontext=user_u:object_r:ld_so_cache_t:s0 tclass=file
type=SYSCALL msg=audit(1173876205.535:131): arch=40000003 syscall=197 success=yes exit=0 a0=3 a1=bfe03330 a2=4a56dfc0 a3=ffffffff items=0 ppid=7967 pid=8266 auid=500 uid=500 gid=500 euid=500 suid=500 fsuid=500 egid=500 sgid=500 fsgid=500 tty=pts2 comm="firefox" exe="/bin/bash" subj=user_u:system_r:firefox_t:s0 key=(null)
type=AVC_PATH msg=audit(1173876205.535:131): path="/etc/ld.so.cache"
type=AVC msg=audit(1173876205.535:132): avc: denied { search } for pid=8266 comm="firefox" name="lib" dev=dm-0 ino=5308417 scontext=user_u:system_r:firefox_t:s0 tcontext=user_u:object_r:lib_t:s0 tclass=dir
type=AVC msg=audit(1173876205.535:132): avc: denied { read } for pid=8266 comm="firefox" name="libtinfo.so.5" dev=dm-0 ino=5308676 scontext=user_u:system_r:firefox_t:s0 tcontext=user_u:object_r:libtinfo.so.5_t:s0 tclass=lnk_file
type=AVC msg=audit(1173876205.535:132): avc: denied { read } for pid=8266 comm="firefox" name="libtinfo.so.5.6" dev=dm-0 ino=5309432 scontext=user_u:system_r:firefox_t:s0 tcontext=user_u:object_r:libtinfo.so.5.6_t:s0 tclass=lnk_file
type=SYSCALL msg=audit(1173876205.535:132): arch=40000003 syscall=5 success=yes exit=3 a0=b7fdf4c7 a1=0 a2=47 a3=b7fdf4c7 items=0 ppid=7967 pid=8266 auid=500 uid=500 gid=500 euid=500 suid=500 fsuid=500 egid=500 sgid=500 fsgid=500 tty=pts2 comm="firefox" exe="/bin/bash" subj=user_u:system_r:firefox_t:s0 key=(null)
Partial Solution: Setroubleshoot

**Summary**

SELinux is preventing `/sbin/dhcdsd` (dhcpcd) "read" access to `/usr/share/X11/locale/iso8859-1/Compose` (usr_t).

**Detailed Description**

SELinux denied access requested by `/sbin/dhcdsd`. It is not expected that this access is required by `/sbin/dhcdsd` and this access may signal an intrusion attempt. It is also possible that the specific version or configuration of the application is causing it to require additional access. Please file a bug report against this package.

**Allowing Access**

Sometimes labeling problems can cause SELinux denials. You could try to restore the default system file context for `/usr/share/X11/locale/iso8859-1/Compose`, `restorecon -v /usr/share/X11/locale/iso8859-1/Compose`. There is currently no automatic way to allow this access. Instead, you can generate a local policy module to allow this access - see FAQ - or you can disable SELinux protection entirely for the application. Disabling SELinux protection is not recommended. Please file a bug report against this package. Changing the "dhcpc_disable_trans" boolean to true will disable SELinux protection this application: "setsebool -P dhcpc_disable_trans=1."

The following command will allow this access:

```
setsebool -P dhcpc_disable_trans=1
```

**Additional Information**

Audit Listener 9/9
Enabling Administrators

- What happens when setroubleshoot has no answer?
  - “It is not expected that this access is required . . . “
- This is one starting point for policy tools
- Improved version of audit messages via audit2allow:
  ```
  # src="firefox_t" tgt="lib_t" class="dir", perms="{ read search getattr }"
  # comm="firefox" exe="" path=""
  allow firefox_t lib_t:dir { read search getattr };
  # src="firefox_t" tgt="lib_t" class="file", perms="{ read getattr execute }"
  # comm="firefox" exe="" path=""
  allow firefox_t lib_t:file { read getattr execute };
  # src="firefox_t" tgt="lib_t" class="lnk_file", perms="read"
  # comm="firefox" exe="" path=""
  allow firefox_t lib_t:lnk_file read;
  ```
Understanding Audit Messages

- What is difficult about translated audit messages?
  - types? object classes and permissions? policy language syntax?
- Type enforcement concepts and syntax are not the problem
  - at least not the most pressing problem
- Evidence:
  - A RH-Firewall-1-INPUT -p udp --dport 5353 -d 224.0.0.251 -j ACCEPT
  - A RH-Firewall-1-INPUT -p udp -m udp --dport 631 -j ACCEPT
  - A RH-Firewall-1-INPUT -p tcp -m tcp --dport 631 -j ACCEPT
  - A RH-Firewall-1-INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
  - A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 22 -j ACCEPT
  - A RH-Firewall-1-INPUT -j REJECT --reject-with icmp-host-prohibited
IPTables is Simple?

- Iptables is accepted and used by administrators
  - at least professional administrators
- What is the difference between Iptables and SELinux?
  - SELinux has simpler policy syntax?
  - Iptables problems are more difficult to diagnose?
- Administrators understand network security
  - have some idea of what network traffic should be allowed
  - understand the access requirements of applications
  - “did you open port 22 for ssh?”
- SELinux access for applications is more challenging . . .
Deciphering Application Access

- Example: allow firefox_t lib_t:file { readgetattr execute };
- Types and object classes are somewhat challenging
  - but not too bad: firefox_t == firefox
  - some object classes and types are worse than others
- Real problem: what does the access mean?
  - and is it dangerous?
- Reference policy helps:
  - libs_exec_lib_files(firefox_t)
- There is even documentation
Policy Writing Tools

- Policy writing usability improving
  - modular policy (available in RHEL 5)
  - Reference Policy adds semantics to access
- Several tools available to help policy writers
  - polgengui: basic module generation
  - sepolgen: policy interface suggestions
  - SLIDE / Bedrock: eclipse-based policy IDE
Polgengui

**Application Type**
- Standard Init Daemon
- Internet Services Daemon (inetd)
- Web Application/Script (CGI)
- User Application

**Incoming Network Port Connections**
- TCP Ports
- UDP Ports: 513
Sepolgen / audit2allow

# grep rwho /var/log/audit/audit.log | audit2allow -R
require {
    type initrc_var_run_t;
    type rwho_t;
    class capability sys_chroot;
    class file { read write getattr lock }; }
#============= rwho_t ==============
allow rwho_t initrc_var_run_t:file { read write getattr lock }; }
#============== rwho_t ===============
allow rwho_t initrc_var_run_t:file { read write getattr lock }; }
allow rwho_t self:capability sys_chroot;
kernél_read_system_state(rwho_t)
policy_module(gpg, 1.0.2)

# Declarations

# Type for gpg or ppg executables.
type gpg_exec_t;

type gpg_helper_exec_t;

type gpg_agent_exec_t;

files_type(gpg_exec_t)
files_type(gpg_helper_exec_t)
files_type(gpg_agent_exec_t)

# Type for the gpg-agent executable.
type gpg_agent_exec_t;

# Type for the pinentry executable

# ifndef ('TODO',
allow sysadm gpg_t { home_root_t user_home_dir_type }:dir search
allow sysadm gpg_t ptyfile:chr_file rw_file_perms;

interface('clock_domtrans',
gen_require(
  type hwclock_t, hwclock_exec_t;
)
Confining the User

Prevent the user from doing things on a computer that you do not want them to do.

Stop malicious software accidentally installed by a user from taking over the machine.

Control either accidental or malicious information flow.

Protect user data.

SELinux in Fedora 2 confined the user

BUT...
So what kind of users can you confine?
Minimal User “guest” account

- Hotel California
  - You can check in anytime you want, but you can never leave!
- Confinement
  - No Setuid
  - No home directory /tmp execution
  - No Networking
  - No X Windows
- System Types
  - Web Site management
  - Git Accounts
  - Terminal Servers
  - Shell Servers
My Bio
If you want to Contact Me

SELinux
Information

My Blog
Fedora
Red Hat Enterprise Linux
National Security Agency (NSA)

Repositories
Presentations
How I confined my wife with SELinux?
How about these users?
And these?
Confining Unknown Users?

Kiosk Mode.

Unknown/Untrusted Users?

Kiosk Mode
Red Hat Non-Technical Staff
Bosses?
CEOS?

CONFINED?
Not Likely...
I even confine this guy!!!