Proactive Security in Linux

Lukas Vrabec
About me

- Lukas Vrabec
- Software Engineer
- Member of Security Technologies team at Red Hat
- Fedora Contributor (selinux-policy, xguest, udica, netlabel_tools)
- lvrabec@redhat.com
- https://lukas-vrabec.com
- https://github.com/wrabcak
- https://twitter.com/mynamewrabcak
Agenda

- Proactive Security
- Traditional Linux Security
- SELinux Security Policy
- Updated Userspace with Easier Policy Customization
- SELinux and Containers
- AVC Messages
Proactive Security
WHEN DO PEOPLE CARE ABOUT SECURITY?
This Hack Will Destroy Your Phone!
WHERE DO SECURITY ISSUES COME FROM?
I HAVE NO IDEA WHAT I'M DOING
HOW ARE THEY FIXED?
REACTIVE SECURITY
Vulnerable software released → Vulnerability announcement → Fix backported & updated released → Timeline

Window of vulnerability
YOUR SYSTEM IS NOT PROTECTED DURING THE WINDOW OF VULNERABILITY!
PROACTIVE SECURITY
Vulnerable software released → Vulnerability announcement → Fix backported & updated released → Window of vulnerability is filled by proactive security
PROACTIVE SECURITY HELPS TO **PROTECT** YOUR SYSTEM DURING THE WINDOW OF VULNERABILITY!
SECURITY ENHANCED LINUX IS A SECURITY MECHANISM BRINGING PROACTIVE SECURITY FOR YOUR SYSTEM.
TECHNOLOGY FOR **PROCESS ISOLATION** TO MITIGATE ATTACKS VIA PRIVILEGE ESCALATION
EXPLOIT EXAMPLES WHERE SELINUX HELPED TO PROTECT YOUR SYSTEM
VENOM
VENOM

DOCKER CVE-2016-9962
VENOM
DOCKER CVE-2016-9962
SHELLSHOCK
HACKING TIME!
DEMO TIME!
CONCLUSION?
IF YOU RUN LINUX WITH SELINUX DISABLED
YOUR GONNA HAVE A BAD TIME
Traditional Linux Security
$ ls -dl /var/www/html/

drwx r-x r-x. 2 root root /var/www/html/

USER GROUP ALL
$ ps -ef | grep NetworkManager

root  11781  1  0 Feb27  00:01:24 /usr/sbin/NetworkManager --no-daemon
PROBLEMS
 ROOT BYPASSING THIS SECURITY
 SETUID BIT
SELinux Security Policy
CORE COMPONENT OF SELINUX
CORE COMPONENT OF SELINUX

COLLECTION OF SELINUX POLICY RULES
CORE COMPONENT OF SELINUX
COLLECTION OF SELINUX POLICY RULES
LOADED INTO THE KERNEL BY SELINUX USERSPACE TOOLS
ENFORCED BY THE KERNEL
ENFORCED BY THE KERNEL

USED TO AUTHORIZE ACCESS REQUESTS ON THE SYSTEM
BY DEFAULT EVERYTHING IS DENIED AND YOU DEFINE POLICY RULES TO ALLOW CERTAIN REQUESTS.
DESCRIBE AN INTERACTION BETWEEN PROCESSES AND SYSTEM RESOURCES
SELINUX POLICY RULE IN HUMAN LANGUAGE
"APACHE process can READ its LOGGING FILE"
SELINUX VIEW OF THAT INTERACTION
ALLOW apache_process apache_log:FILE;
READ;
apache_process apache_log

ARE LABELS
LABELS
ASSIGNED TO PROCESSES
ASSIGNED TO PROCESSES

ASSIGNED TO SYSTEM RESOURCES
ASSIGNED TO PROCESSES
ASSIGNED TO SYSTEM RESOURCES
BY SELINUX SECURITY POLICY
ASSIGNED TO PROCESSES
ASSIGNED TO SYSTEM RESOURCES
BY SELINUX SECURITY POLICY

MAP REAL SYSTEM ENTITIES INTO THE SELINUX WORLD
LABELS IN REALITY
# getfattr -n security.selinux /etc/passwd
getfattr: Removing leading '/' from absolute path names
# file: etc/passwd
security.selinux="system_u:object_r:passwd_file_t:s0"

# ls -Z /etc/passwd
system_u:object_r:passwd_file_t:s0 /etc/passwd
SELINUX LABELS CONSIST OF **FOUR PARTS**
<user>:<role>:<type>:<MLS/MCS>

Not the same as Linux users

Several Linux users can be mapped to a single SELinux user

object_u is a placeholder for Linux system resources

system_u is a placeholder for Linux processes

Can be limited to a set of SELinux roles
<user>:<role>:<type>:<MLS/MCS>

<user>:<role>:<type>:<MLS/MCS>
SELinux users can have multiple roles but only one can be active.

object_r is a placeholder for Linux system resources
system_r is a placeholder for system processes
Can be limited to a set of SELinux types
<user>:<role>:<type>:<MLS/MCS>

<user>:<role>:<type>:<MLS/MCS>
Security model known as **TYPE ENFORCEMENT**

In 99% you care only about TYPES policy rules and interactions between types
Multi Level Security

Only the MCS part is used in Targeted Policy with the default s0 level. Allow users to mark resources with compartment tags (MCS1, MCS2). Used for RHEL virtualization and for container security. 

s0:c1 can not access s0:c2
<table>
<thead>
<tr>
<th>User</th>
<th>Role</th>
<th>Domain</th>
<th>X Window System</th>
<th>su or sudo</th>
<th>Execute in home directory and /tmp/ (default)</th>
<th>Networking</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysadm_u</td>
<td>sysadm_r</td>
<td>sysadm_t</td>
<td>yes</td>
<td>su and sudo</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>staff_u</td>
<td>staff_r</td>
<td>staff_t</td>
<td>yes</td>
<td>only sudo</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>user_u</td>
<td>user_r</td>
<td>user_t</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>guest_u</td>
<td>guest_r</td>
<td>guest_t</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>xguest_u</td>
<td>xguest_r</td>
<td>xguest_t</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>Firefox only</td>
</tr>
</tbody>
</table>
IN RHEL7 WE SHIP THE **TARGETED** SELINUX POLICY BY DEFAULT
WE MOSTLY CARE ONLY ABOUT **TYPES**
SELINUX ALLOW RULE SYNTAX WITH TYPES
ALLOW TYPE1 TYPE2:OBJECT_CLASS PERMISSION;
ALLOW APACHE_T APACHE_LOG_T:FILE READ;
DOMAIN TRANSITION RULES
TYPE_TRANSITION TYPE1 TYPE2:PROCESS NEW_DOMAIN;
TYPE_TRANSITION INIT_T
HTTPD_EXEC_T:PROCESS HTTPD_T;
FILE TRANSITION RULES
TYPE_TRANSITION TYPE1 TYPE2: OBJECT_CLASS
NEW_TYPE;
TYPE_TRANSITION HTTPD_T VAR_LOG_T:FILE HTTPD_LOG_T;
ENFORCING
ENFORCING

SELINUX SECURITY POLICY IS ENFORCED BY KERNEL
PERMISSIVE
PERMISSIVE

SELINUX SECURITY POLICY IS NOT ENFORCED BY KERNEL
PERMISSIVE
SELINUX SECURITY POLICY IS NOT ENFORCED BY KERNEL
ACCESSSES ARE LOGGED
UPDATED USERSPACE WITH EASIER POLICY CUSTOMIZATION
NEW COMMON INTERMEDIATE LANGUAGE - CIL
"M4+COMPILATION" VS. CIL
PERFORMANCE IMPROVEMENTS
PERFORMANCE IMPROVEMENTS

NEW POSSIBILITY FOR HLL
PERFORMANCE IMPROVEMENTS
NEW POSSIBILITY FOR HLL
USABILITY
LOCAL POLICY IN TWO STEPS
# cat myapache.cil
(allow httpd_t httpd_log_t (file (open read getattr)))
# semodule -i myapache.cil
HOW DO WE DO IT WITH M4 + COMPILATION?
# cat myapache.te
require {
    type httpd_t;
    type httpd_log_t;
}
allow httpd_t httpd_log_t:file { open read
getattr };
# make -f /usr/share/selinux/devel/Makefile
# semodule -i myapache.pp
SELINUX VS. CONTAINERS
APPLIES MAC TO IMPROVE SECURITY WHEN USING VIRTUAL MACHINES
Granted access:

- container_t:s0:c1,c2
  - container_file_t:s0
  - container_file_t:s0:c1
  - container_file_t:s0:c2
  - container_file_t:s0:c1,c2

- container_t:s0:c2,c3
  - container_file_t:s0
  - container_file_t:s0:c2
  - container_file_t:s0:c3
  - container_file_t:s0:c2,c3
SELinux user:SELinux role:SELinux type:SELinux category
SELinux user:SELinux role:SELinux category
system_u:object_r:svirt_t:c306,c536
SELinux user:SELinux role:SELinux type:SELinux category
system_u:object_r:svirt_t:c306,c536
system_u:object_r:svirt_t:c206,c636
SELINUX KEEPS YOUR CONTAINER IN ITS OWN SPACE
SELinux user:SELinux role:SELinux type:SELinux category
SELinux user:SELinux role:SELinux type:SELinux category
system_u:object_r:container_t:c306,c536
SELinux user:SELinux role:SELinux type:SELinux category
system_u:object_r:container_t:c306,c536
system_u:object_r:container_t:c206,c636
SELinux user:SELinux role:SELinux type:SELinux category

system_u:object_r:container_t:c306,c536
system_u:object_r:container_t:c206,c636
system_u:object_r:container_t:c406,c736
AVC MESSAGES
WHERE CAN WE FIND LOGS?
# cat /var/log/audit/audit.log
# cat /var/log/audit/audit.log

# ausearch -m AVC
type=AVC  msg=audit(1226882925.714:136):  avc: denied { read } for pid=2512  comm="httpd" name="file1"  
  dev=dm-0  ino=284133  
  scontext=unconfined_u:system_r:httpd_t:s0  
  tcontext=unconfined_u:object_r:shadow_t:s0  
  tclass=file
HOW TO PARSE AVC MESSAGES?
# ausearch
# ausearch

# audit2allow
# ausearch -m AVC -ts recent

+ type=AVC msg=audit(1226882925.714:136): avc: denied { read } for
  pid=2512 comm="httpd" name="shadow" dev=dm-0 ino=284133
  scontext=unconfined_u:system_r:httpd_t:s0
  tcontext=unconfined_u:object_r:shadow_t:s0 tclass=file

# ausearch -m AVC -ts recent | audit2allow

#============= httpd_t ==============

allow httpd_t shadow_t:file read;
- # semanage fcontext -> manage SELinux contexts
- # semanage boolean -> manage SELinux booleans
- # semanage port -> manage SELinux ports
- # semanage permissive -> put SELinux domain to permissive mode
- # sesearch -> search for present SELinux rules
- # ausearch -> search for SELinux denials
- # sealert -> SELinux troubleshooter
- # audit2allow -> Parse SELinux denials / create local SELinux module
- # semodule -DB / # semodule -B -> SELinux policy rebuild
ARE YOU USING SELINUX IN ENFORCING?
**BLOGS**

Lukas Vrabec’s blog  [https://lukas-vrabec.com/](https://lukas-vrabec.com/)
Dan Walsh’s blog  [http://danwalsh.livejournal.com/](http://danwalsh.livejournal.com/)
Miroslav Grepl’s blog  [https://mgrepl.wordpress.com/](https://mgrepl.wordpress.com/)
Petr Lautrbach’s blog  [https://plautrba.fedorapeople.org/](https://plautrba.fedorapeople.org/)