The Linux Audit Subsystem
Deep Dive

MVMUA
20-Apr 2010

Brent Holden <bholden@redhat.com>
Red Hat, Inc

Session Themes

- Why is Linux Auditing needed? What can it do for me?
- How does it work? How do events get audited?
- How do I make sense of all the data?

Yes, I work for Red Hat. Yes, the demos & examples are on RHEL. Everything is applicable to SuSE as well.

Demos / Examples

Why is Linux Auditing needed? What can it do for me?

- Taken from a customers INFOSEC policy

(GEN002720: CAT II) The SA will configure the auditing system to audit logon (unsuccessful and successful) and logout (successful)

(GEN002760: CAT II) The SA will configure the auditing system to audit unauthorized access attempts to files (unsuccessful)

(GEN002780: CAT II) The SA will configure the auditing system to audit use of privileged commands (unsuccessful and successful)

(GEN002840: CAT II) The SA will configure the auditing system to audit all security personnel actions

(GEN002820: CAT II) The SA will configure the auditing system to audit all system administration actions

Why is Linux Auditing needed? What can it do for me?

- Some of the design requirements for the audit system:
  - Shall be able to record at least the following
    - Date and time of event, type of event, subject identity, outcome
    - Sensitivity labels of subjects and objects
    - Be able to associate event with identity of user causing it
    - All modifications to audit configuration and attempted access to logs
    - All use of authentication mechanisms
    - Changes to any trusted database
    - Attempts to import/export information
    - Be able to include/exclude events based on user identity, subject/object, labels, other attributes

Why is Linux Auditing needed? What can it do for me?

- Linux Audit is a system to
  - Collect information regarding events occurring on the system(s)
    - Kernel events (syscall events)
    - User events (audit-enabled programs)
  - Form a log recording and describing each event (/var/log/audit/audit.log)
  - Components to assist analysing the log

National Information Assurance Partnership

Common Criteria Certificate

is awarded to

IBM Corporation

The IT product identified in this certificate has been evaluated at an accredited testing laboratory using the Common Methodology for IT Security Evaluation (Version 2.3) for conformance to the Common Criteria for IT Security Evaluation (Version 2.3). This certificate applies only to the specific version and release of the product in its evaluated configuration. The product's functional and assurance security specifications are contained in its security target. The evaluation has been conducted in accordance with the provisions of the NIAP Common Criteria Evaluation and Validation Scheme and the conclusions of the testing laboratory in the evaluation technical report are consistent with the evidence adduced. This certificate is not an endorsement of the IT product by any agency of the U.S. Government and no warranty of the IT product is either expressed or implied.

Product Name: Red Hat Enterprise Linux Version 5
Evaluation Platforms: IBM System x3550, Blade Center HS20 and HS21; IBM System x3455 and Blade Center LS21; IBM System p: any POWER5/POWER5+ compliant system or software; IBM System z: any z/Architecture compliant system or software
Assurance Level: EAL 4 Augmented ALC_FLR.3

CCTL: atsec information security corporation
Validation Report Number: CCEVS-VR-07-0037
Date Issued: 7 June 2007
Protection Profile Identifier: Labeled Security Protection Profile, Issue 1.b, 8 October 1999 Controlled Access Protection Profile, V1.d, October 8, 1999;; Role-based Access Control Protection Profile, Version 1.0, July 30, 1998

Original Signed By

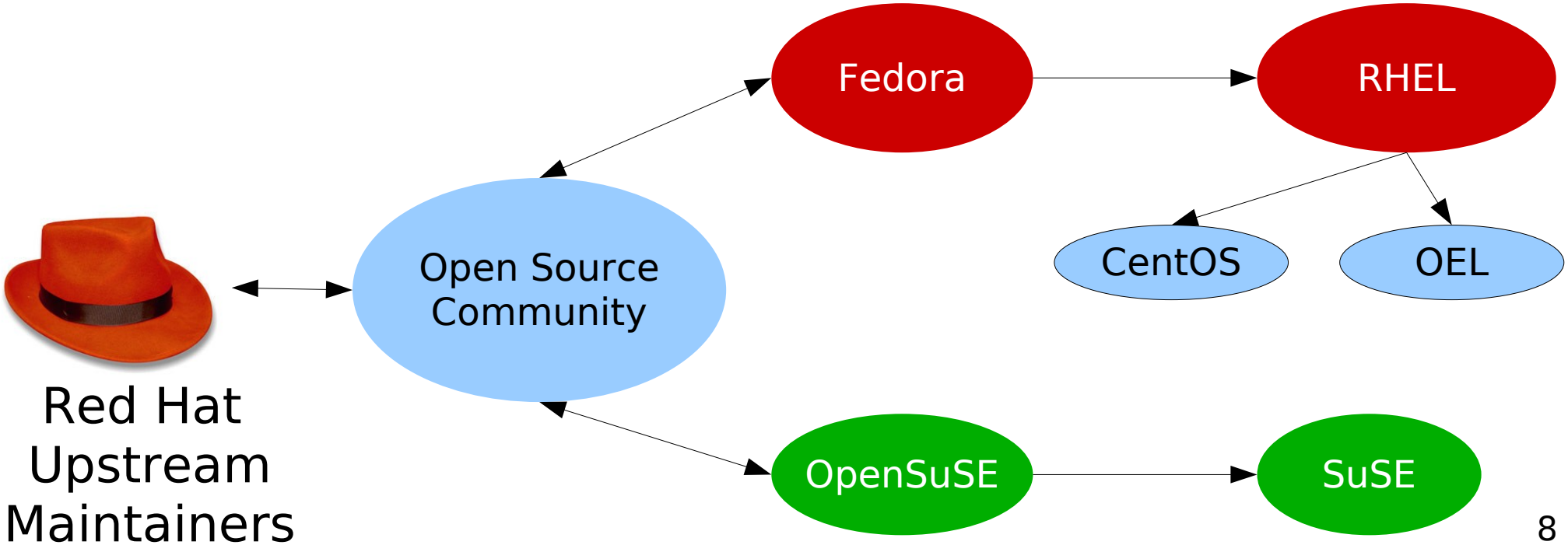
Director, Common Criteria Evaluation and Validation Scheme
National Information Assurance Partnership

Original Signed By

Information Assurance Director
National Security Agency

How Audit is Developed

- The Linux Audit Subsystem is completely open source, and integrated directly into the kernel.
- Red Hat is the creator and upstream maintainer, but that doesn't mean we're the only ones working on it



Audit v. SysLog

- The conventional use of **Syslog** is to record states of the system, such as hardware alerts.

  Applications may also send logging data there, such as the SSH daemon does upon invalide SSH attempts. However, application level syslog use is up to the application, and the application may not record everything.

- Example Syslog notifiers include the following, as defined in linux/kernel.h

```
#define KERN_EMERG      "<0>"   /* system is unusable               */
#define KERN_ALERT      "<1>"   /* action must be taken immediately */
#define KERN_CRIT       "<2>"   /* critical conditions              */
#define KERN_ERR        "<3>"   /* error conditions                 */
#define KERN_WARNING    "<4>"   /* warning conditions               */
#define KERN_NOTICE     "<5>"   /* normal but significant condition */
#define KERN_INFO       "<6>"   /* informational                    */
#define KERN_DEBUG      "<7>"   /* debug-level messages             */
```

Audit v. SysLog

- Since there is no assurance that applications will completely log all user actions, we offloaded that responsibility to the Linux kernel. Thus, the Linux Audit subsystem was born.

- When audit is defined, every single system call will pass through the Audit subsystem, which has rules defined at /etc/audit/audit.rules

- Audit has the ability to log **BEFORE** and **AFTER** an action: this is extremely useful.

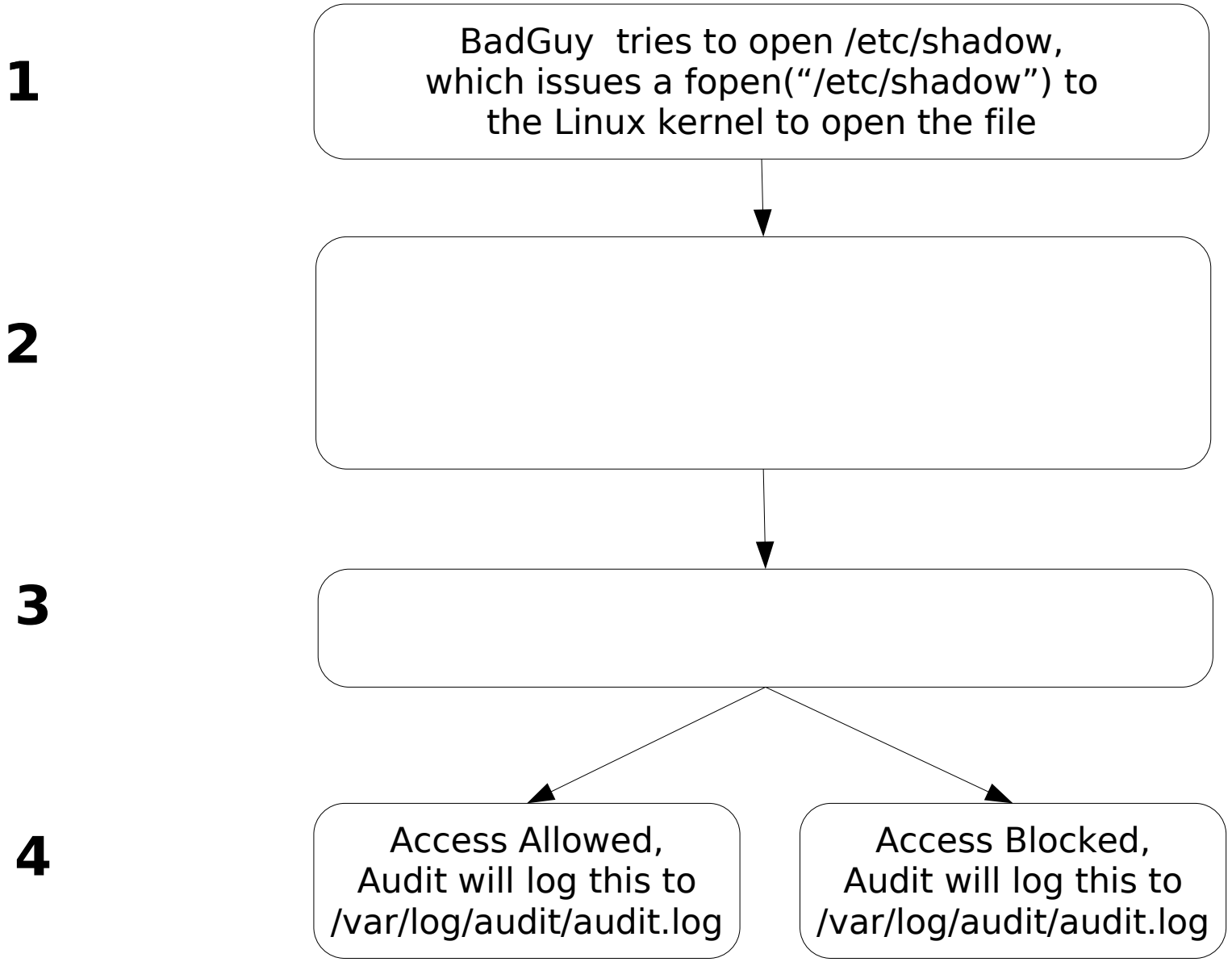
How does it work?

Why is Linux Auditing needed? What can it do for me?

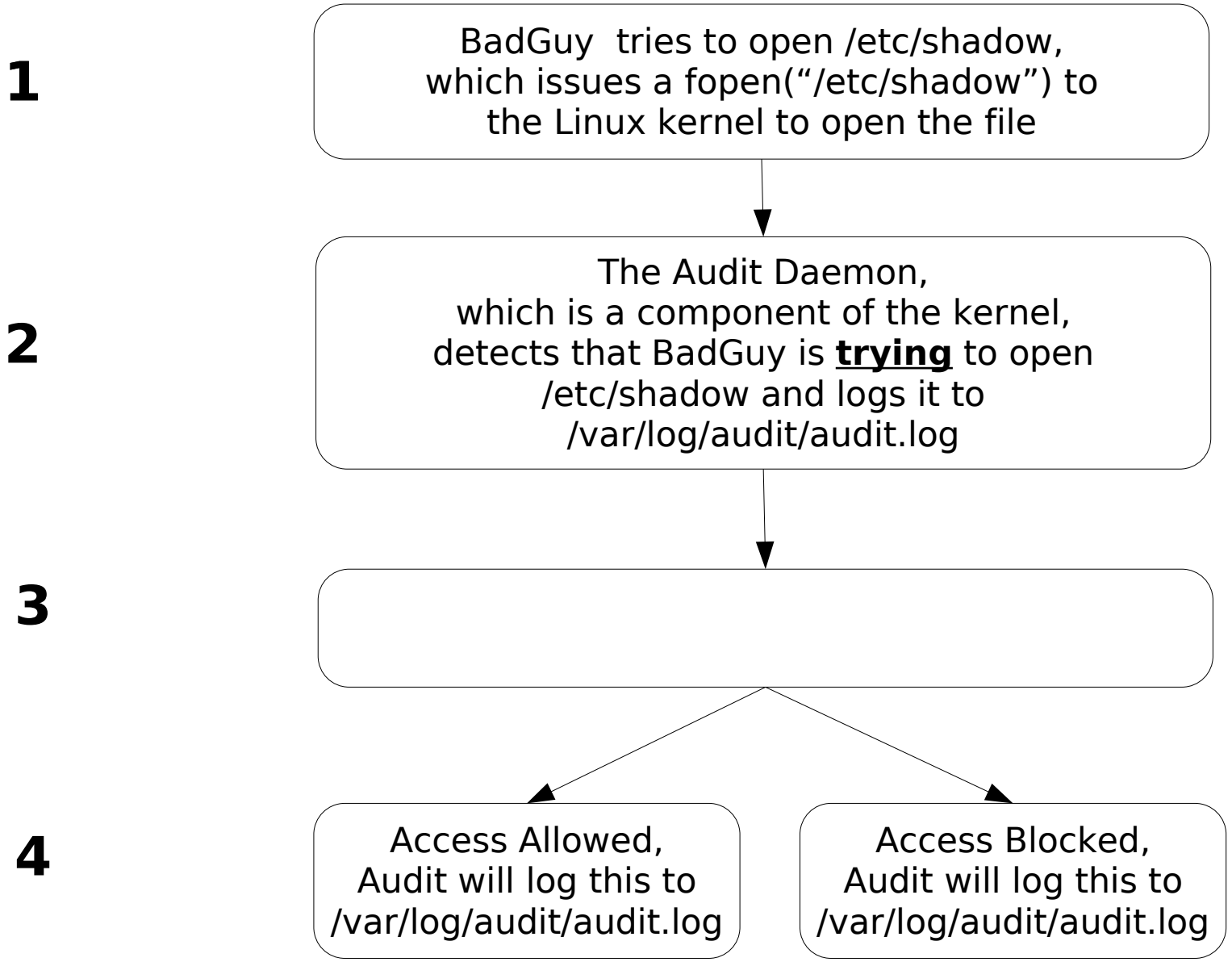
(GEN002760: CAT II) The SA will configure the auditing system to audit unauthorized access attempts to files (unsuccessful)

But How?

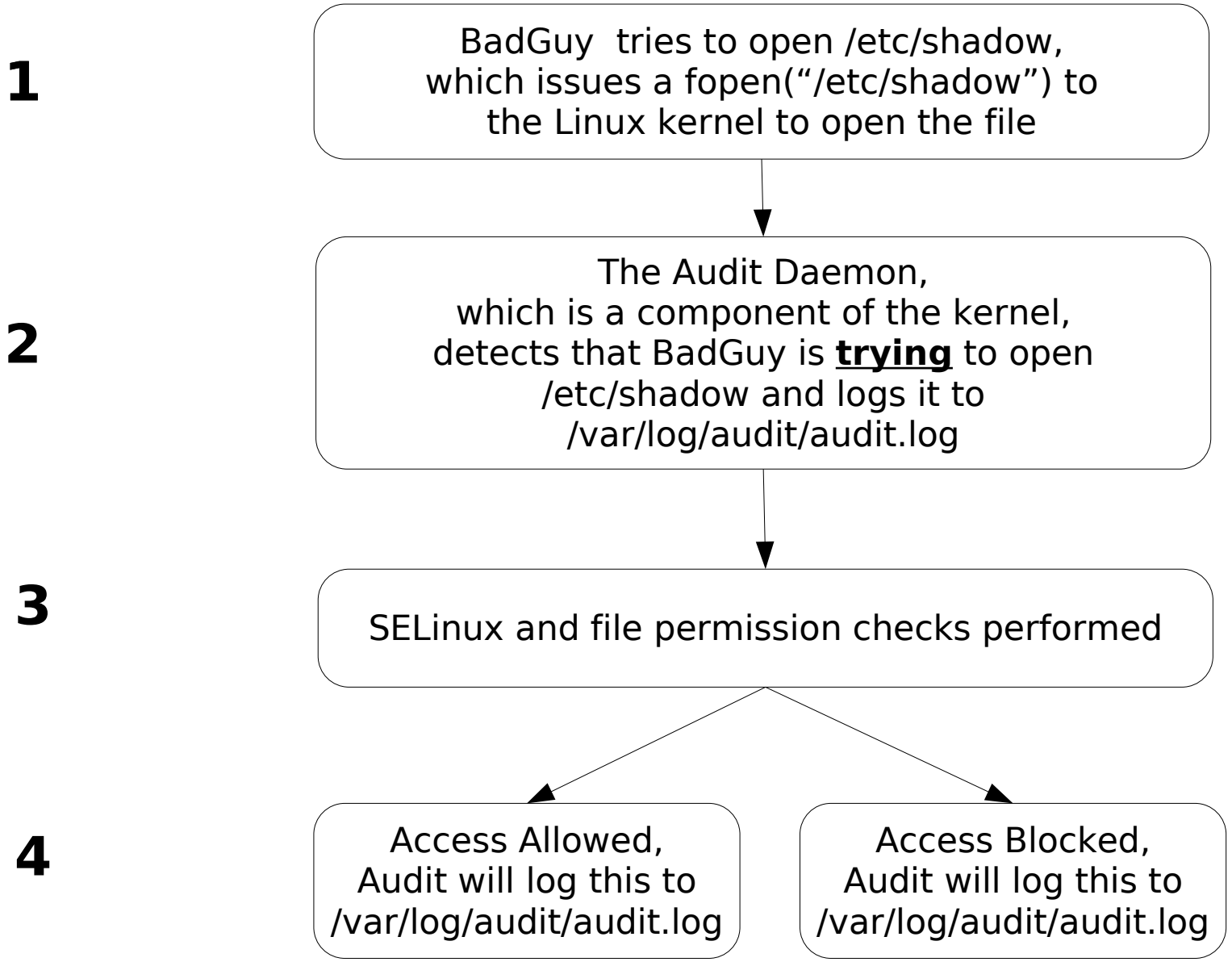
(GEN002760: CAT II) The SA will configure the auditing system to audit unauthorized access attempts to files (unsuccessful)



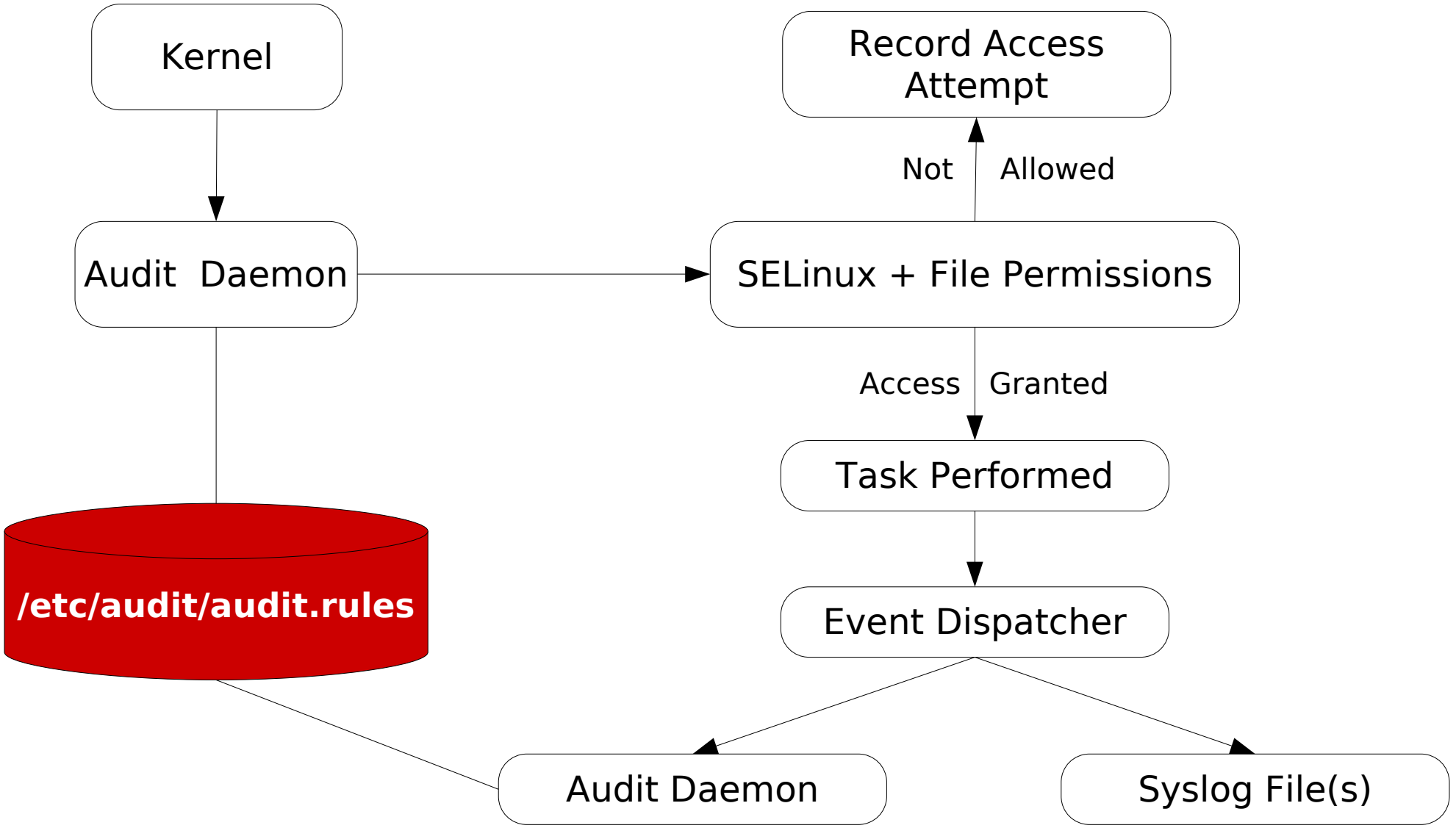
(GEN002760: CAT II) The SA will configure the auditing system to audit unauthorized access attempts to files (unsuccessful)



(GEN002760: CAT II) The SA will configure the auditing system to audit unauthorized access attempts to files (unsuccessful)



(GEN002760: CAT II) The SA will configure the auditing system to audit unauthorized access attempts to files (unsuccessful)



tail -F /var/log/audit/audit.log

$ cat /etc/shadow [as non-root]

Configuration via auditctl

- auditctl is a command line utility to control the behavior, get status, and add or delete rules
  - Useful for kickstarts / system automation
  - Useful for making **non-persistent** changes
    - Reminder: use auditctl, then update audit.rules to be persistent

- Arguments to familiarize yourself with

  -k *{key}* Sets a filter on an audit rule, which you can query against via the ausearch utility

  -F arch=*{b32 , b64}* Selects the architecture libraries to use when logging

  -w *{file}* Watches for edits to a file, such as a user modifying /etc/shadow

- All arguments are in the man page (man auditctl)

LSPP, CAPP, NISPOM example rules
with auditctl

Monitoring /etc/audit/audit.rules

- Always track files by inode number!

```
# auditctl -a exit,always -S open -F inode=`ls
-i /etc/audit/auditd.conf | gawk '{print $1}'`
```

```
# auditctl -l
. . . . . .
AUDIT_LIST: exit,always inode=1637178
(0x18fb3a) syscall=open
```

When someone accesses the file, you'll receive a log similar to

```
type=PATH
msg=audit(1251123553.303:206): item=0
name="/etc/audit/audit.rules"
inode=77546 dev=fd:01 mode=0100640
ouid=0 ogid=0 rdev=00:00
obj=system_u:object_r:auditd_etc_t:s0
```

Configuration via system-config-audit

How do I make sense of the data?

Usage of ausearch

- Ausearch is a command-line utility to query your audit logs

  ausearch -f <file>

  ausearch -ui <user>

ausearch Example (1/3)

- auditctl -l | grep shadow

  LIST_RULES: exit,always
  watch=/etc/gshadow perm=wa key=auth

  LIST_RULES: exit,always
  watch=/etc/shadow perm=wa key=auth

- sudo -u badguy cat /etc/shadow
  cat: /etc/shadow: Permission denied

ausearch Example (2/3)

- aureport -r

  Summary Report

  ======================

  Range of time in logs: 06/21/2009 14:43:44.362 - 08/24/2009 10:29:29.498

  Selected time for report: 06/21/2009 14:43:44 - 08/24/2009 10:29:29.498

  Number of changes in configuration: 164

  Number of changes to accounts, groups, or roles: 32

  Number of logins: 2

  Number of failed logins: 6

  Number of failed syscalls: 290

ausearch Example (3/3)

- ausearch -sc open -sv no

time->Mon Aug 24 10:27:40 2009

type=PATH msg=audit(1251124060.832:231): item=0 **name="/etc/shadow" inode=137125** dev=fd:01 mode=0100400 ouid=0 ogid=0 rdev=00:00 obj=system_u:object_r:shadow_t:s0

type=CWD msg=audit(1251124060.832:231): cwd="/root"

type=SYSCALL msg=audit(1251124060.832:**231**): arch=40000003 syscall=5 **success=no** exit=-13 a0=bfeec9e8 a1=8000 a2=0 a3=bfeeab6c items=1 ppid=3934 **pid=7964 auid=500 uid=500 gid=500** euid=500 suid=500 fsuid=500 egid=500 sgid=500 fsgid=500 tty=pts2 ses=1 **comm="cat" exe="/bin/cat"** subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 key="open"

Usage of autrace

- autrace is similar to strace
- This command deletes all audit rules prior to executing the target program and after executing it.
- As a safety precaution, it will not run unless all rules are deleted with auditctl prior to use.

autrace Example (1/2)

- autrace /bin/ls /etc/audit/auditd.conf

  autrace cannot be run with rules loaded.

  Please delete all rules using 'auditctl -D' if you really wanted to run this command

- auditctl -D

  No rules

- autrace /bin/ls /etc/audit/auditd.conf

  Waiting to execute: /bin/ls

  /etc/audit/auditd.conf

  Cleaning up...

  Trace complete. You can locate the records with **'ausearch -i -p 8031'**

autrace Example (2/2)

- autrace -i -p 8031

```
----
type=SYSCALL msg=audit(08/24/2009
10:38:09.251:382) : arch=i386 syscall=write
success=yes exit=23 a0=1 a1=b7fc0000 a2=17 a3=17
items=0 ppid=8029 pid=8031 auid=sdw uid=root
gid=root euid=root suid=root fsuid=root egid=root
sgid=root fsgid=root tty=pts2 ses=1 comm=ls
exe=/bin/ls
subj=unconfined_u:unconfined_r:unconfined_t:s0-
s0:c0.c1023 key=(null)
----
type=SYSCALL msg=audit(08/24/2009
10:38:09.251:383) : arch=i386 syscall=close
success=yes exit=0 a0=1 a1=0 a2=930ff4 a3=9314c0
items=0 ppid=8029 pid=8031 auid=sdw uid=root
gid=root euid=root suid=root fsuid=root egid=root
sgid=root fsgid=root tty=pts2 ses=1 comm=ls
exe=/bin/ls
subj=unconfined_u:unconfined_r:unconfined_t:s0-
s0:c0.c1023 key=(null)
```

Data Visualization

Audit Data Visualization

- http://people.redhat.com/sgrubb/audit/visualize/index.html

  Steve Grub, the maintainer of the Linux Audit subsystem, has written an excellent paper + utilities for visualizing audit data. There are two utilities, mkbar and mkgraph, which perform functions as indicated in the name

- The mkgraph script can be downloaded from:

  http://people.redhat.com/sgrubb/audit/visualize/mkgraph

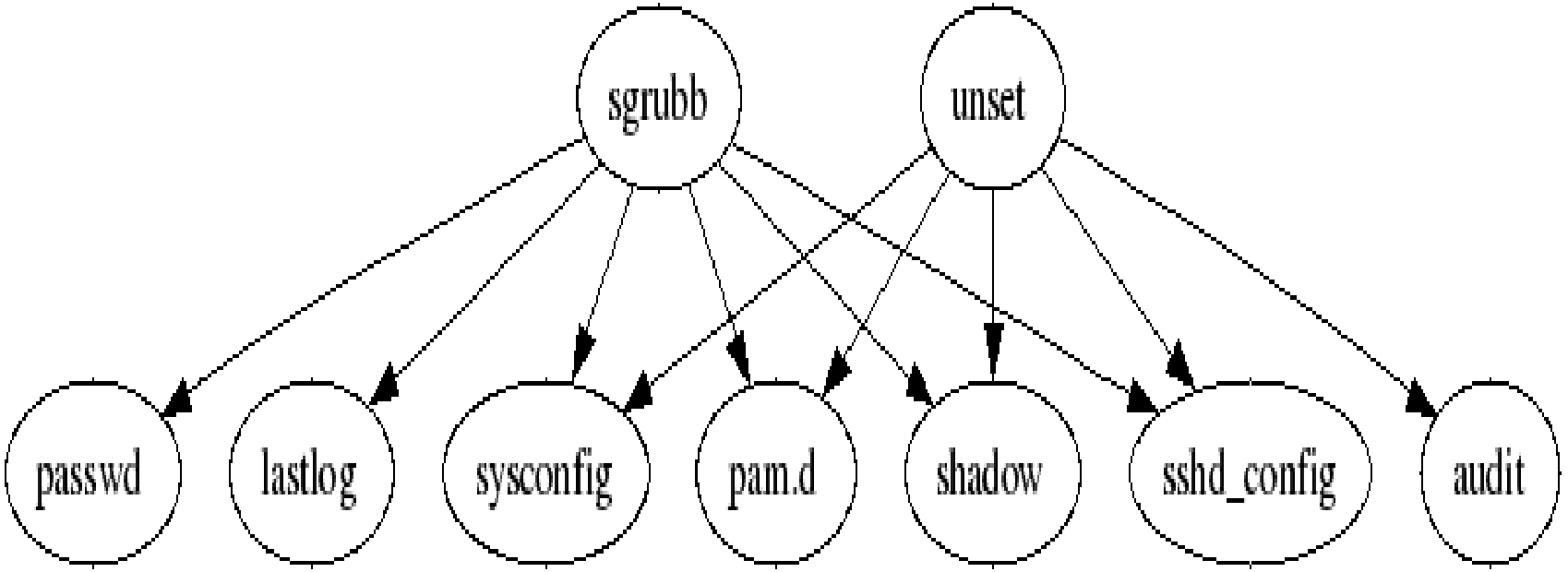
- The mkbar script can be downloaded from:
  http://people.redhat.com/sgrubb/audit/visualize/mkbar

- Note that these scripts have absolutely no vendor support, they're simply utilities we thought others would get use from

Audit Data Visualization

- See who is accessing files:

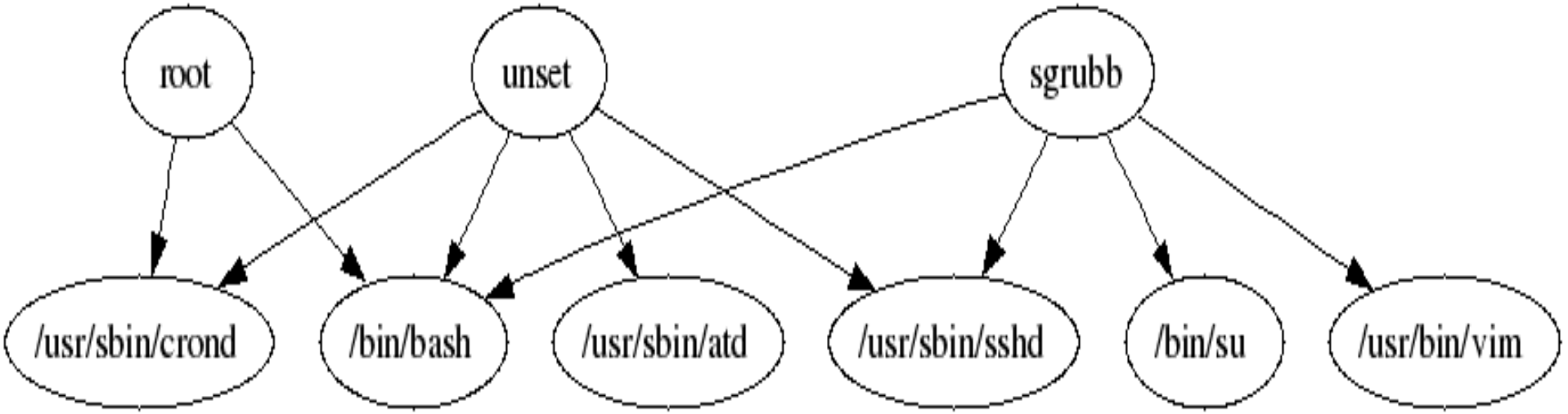
  aureport -f -i | awk '/^[0-9]/ { printf "%s %s\n", $8, $4 }' |
  sort | uniq | ./mkgraph



Audit Data Visualization

- See what account is running which exes:

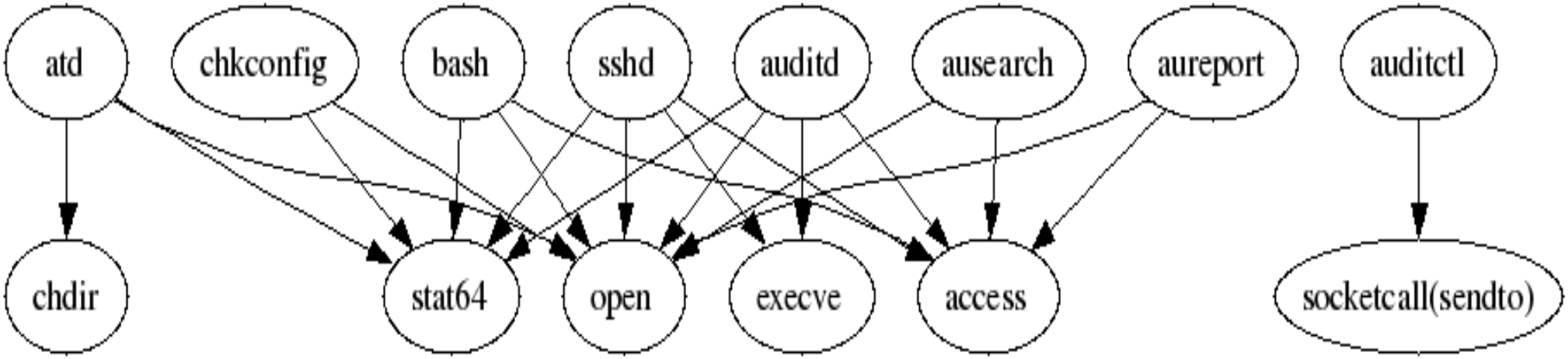
  aureport -u -i | awk '/^[0-9]/ { printf "%s %s\n", $4, $7 }' |
  sort | uniq | ./mkgraph



Audit Data Visualization

- See what syscalls a program makes:

  aureport -s -i | awk '/^[0-9]/ { printf "%s %s\n", $6, $4 }' |
  sort | uniq | ./mkgraph



Appendix

Appendix

- The Linux audit subsystem is generally available in the vanilla Linux kernel at http://www.kernel.org

- Audit userspace tools and daemon are available at:

  http://people.redhat.com/sgrubb/audit

- linux-audit mailing list (for all Audit, not just RHEL implementations)
  http://www.redhat.com/mailman/listinfo/linux-audit