Native Host Intrusion Protection with RHEL5 and the Audit Subsystem

Steve Grubb

May 11, 2007

Introduction

- How the audit system works
- How we can layer an IDS/IPS system on top of it

Introduction

- Designed to meet or exceed:
  - CAPP, LSPP, RBAC, NISPOM, FISMA, PCI, DCID 6/3
- Evaluated by NIAP
- Certified to CAPP/EAL4+ on RHEL4
- Under evaluation for LSPP/CAPP/RBAC/EAL4+ RHEL5

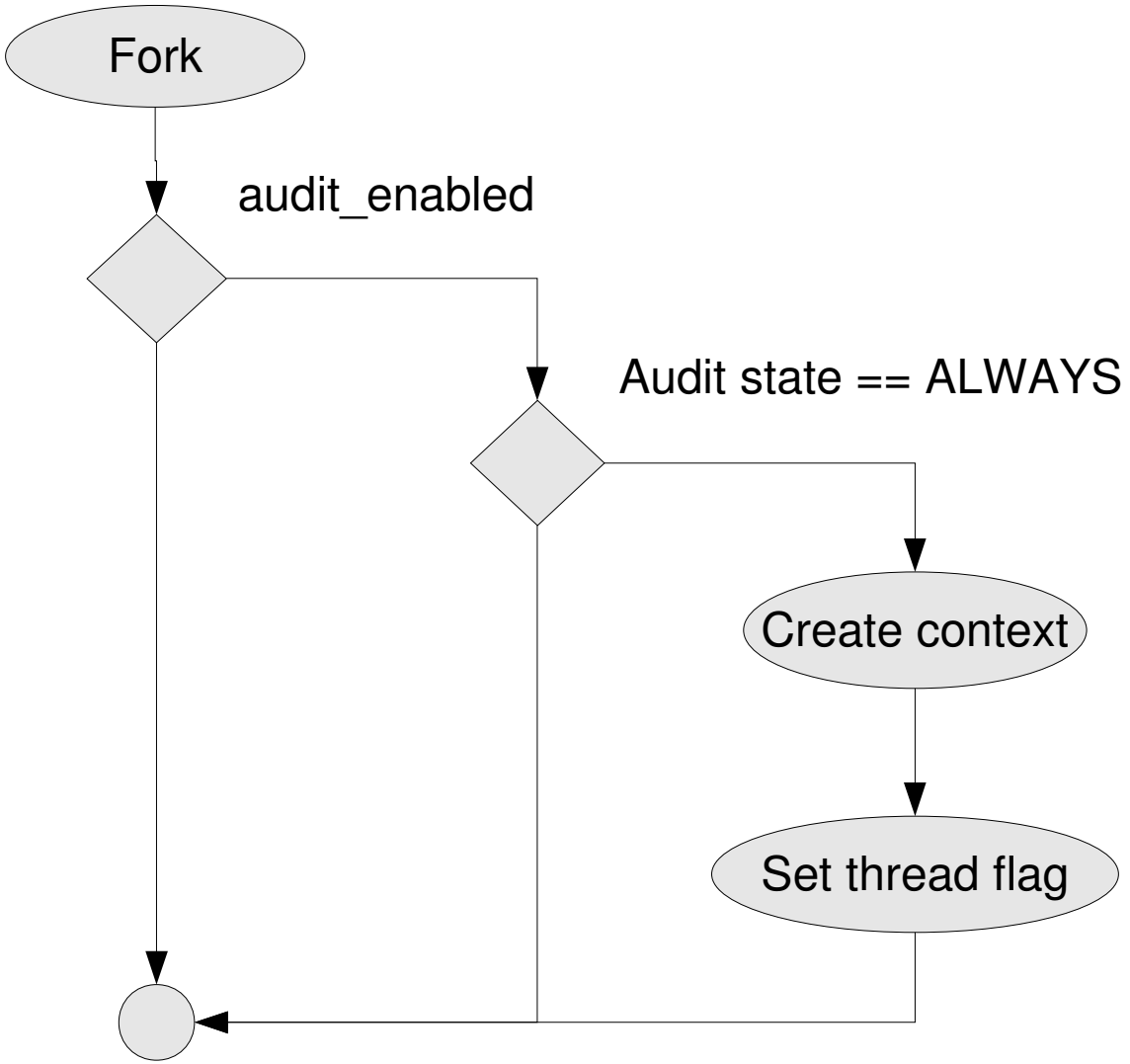
Introduction

- Some of the 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

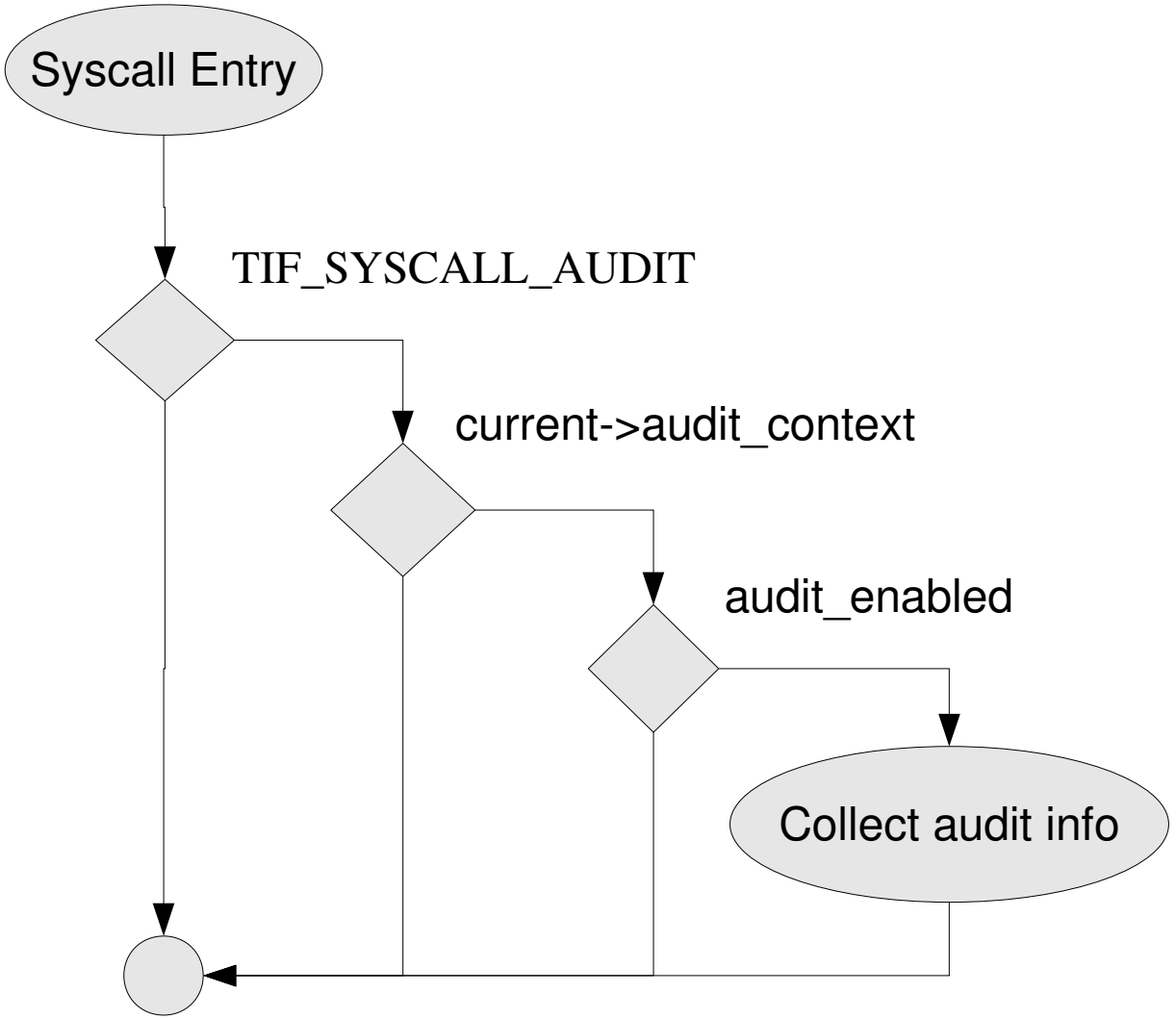
Kernel

- Designed to minimize the performance impact as little as possible
- Relies on a flag, TIF_SYSCALL_AUDIT, which is part of the thread's information flags variable.
- Flag is inheritted at fork when audit_enabled is true
- Flag is reset by "never" audit rule directive
- If you need audit of all processes, you must use audit=1 as a boot parameter.

Kernel – audit flag inheritance

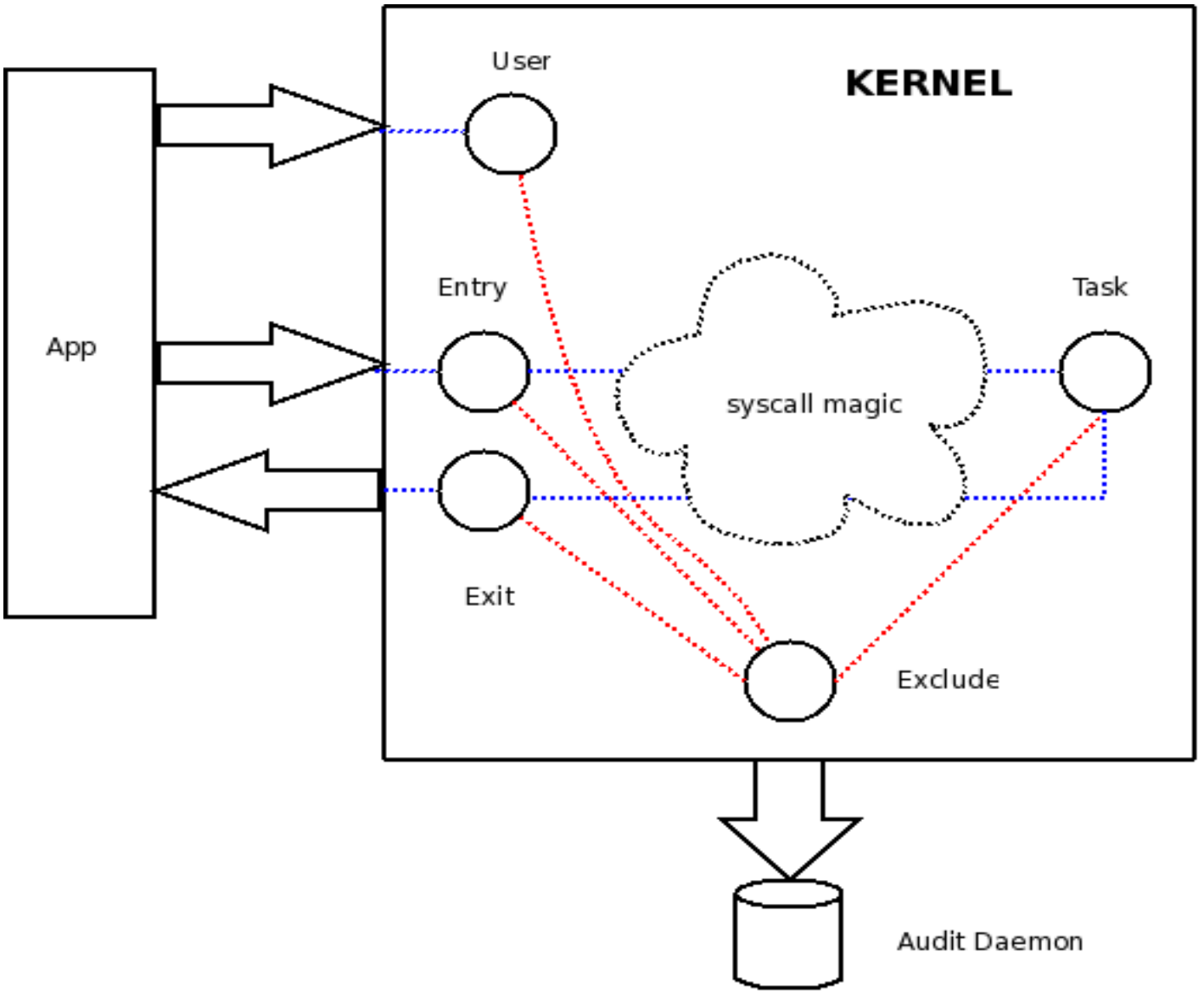


Kernel – syscall entry



Kernel

- Need to decide if the sycall excursion is of interest
- Audit context has a state variable: NEVER and ALWAYS
- Filters
  - Entry
  - Exit
  - Task
  - User
  - Exclude


User
KERNEL
App
Entry
Task
syscall magic
Exit
Exclude
Audit Daemon


RED HAT :: SAN DIEGO :: 2007
SUMMIT

Kernel

- Syscall Exit
  - If context marked auditable emit event
  - Event can be multi-part
    - Ex. Message Queue attributes, IPC attributes, execve args, socket addr, socket call args, file paths, and current working directory.
  - All are tied together with time stamp and serial number
  - Free allocated resources

Audit Event

type=SYSCALL msg=audit(1178552800.984:490): arch=40000003
syscall=10 success=yes exit=0 a0=8ca5460 a1=16 a2=5 a3=8ca547d
items=2 ppid=749 pid=3783 auid=4325 uid=0 gid=0 euid=0 suid=0 fsuid=0
egid=0 sgid=0 fsgid=0 tty=pts2 comm="vim" exe="/usr/bin/vim"
subj=user_u:system_r:unconfined_t:s0key="LOG_audit"

type=CWD msg=audit(1178552800.984:490): cwd="/root"

type=PATH msg=audit(1178552800.984:490): item=1
name="/var/log/audit/.audit.log.swp" inode=295008 dev=08:05
mode=0100600 ouid=0 ogid=0 rdev=00:00
obj=user_u:object_r:auditd_log_t:s0

User Space Controls

- Audit rules are stored at /etc/audit/audit.rules
- Audit rules are loaded by auditctl
- Auditctl can control the kernel settings:
  - -e 0/1/2 disable/enable/enabled and immutable
  - -f 0/1/2 failure mode silent/printk/panic
  - -b 256 backlog
  - -r 0 event rate limit
  - -s get status
  - -l list all rules
  - -D delete all rules

Syscall Rules

Follows the general form:

-a filter,action -S syscall -F field=value

Example to see failed opens for user 500:

-a exit,always -S open -F success!=0 -F auid=500

-F can be one of: a0, a1, a2, a3, arch, auid, devmajor, devminor, user/group ids, inode, msgtype, object/subject context parts, pid, ppid, or success.

“and” created by adding more “-F” name/value pairs. An “or” is created by adding a new rule.

Results are evaluated by the filter to decide if event is auditable

Kernel – File access auditing

- Syscall auditing presents us with a problem when we need to monitor files
- Audit system does collect devmajor/minor information and inode
- But many interesting files are edited as temp copy and then replace original file
- This causes the inode to change

Kernel – File System Auditing

- Audit rules specified as a path and permision
- Kernel translates into inode rule
- When something replaces a watched file, inode rule updated in kernel
- Reconciliation is done by syscall exit filter
- Limitations:
  - No wildcards or recursive auditing
  - If rule specifies directory, audits changes to dir entries

File System audit rules

File system audit rules take the general form of:

-w /full/path-to-file -p wrxa -k “rule note”

Can also be expressed as syscall audit rule:

-a exit,always -F path=/full/path-to-file -F perm=wrxa -k “rule note”

The perm field selects the syscalls that are involved in file writing, reading, execution, or attribute change.

Audit Daemon

- Audit daemon's job is to simply dequeue events from netlink interface as fast and possible and log them to disk
- It does no translation or changing of audit data
- It monitors disk usage for partition where logs are located
- When short on disk space, it can respond in one of several ways
  - ignore, syslog, email, execute program, suspend, single, or halt
- The audit daemon handles its own log rotation since it must always be running or events get dumped to syslog

Ausearch

- The ausearch program is the preferred way to look at audit logs
- Can do simple queries
- Correlates the individual records to 1 event
- Can interpret fields from numeric data to human readable form
- Can be used to extract events from audit logs

Ausearch Examples

- Searching for bad logins:
  - ausearch -m USER_AUTH,USER_ACCT --success no
- Searching for events on shadow file today
  - ausearch -f shadow --start today
- Searching for failed access user acct 500
  - ausearch -m PATH --success no --syscall open --loginuid 500
- Extracting logs for 2 days
  - ausearch --start yesterday --raw > new.log

Audit Event Type Classes

1000 - 1099 are for commanding the audit system
1100 - 1199 user space trusted application messages
1200 - 1299 messages internal to the audit daemon
1300 - 1399 audit event messages
1400 - 1499 kernel SE Linux use
1600 - 1699 kernel crypto events
1700 - 1799 kernel anomaly records
1800 - 1999 future kernel use (maybe integrity labels and related events)
2001 - 2099 unused (kernel)
2100 - 2199 user space anomaly records
2200 - 2299 user space actions taken in response to anomalies
2300 - 2399 user space generated LSPP events
2400 - 2499 user space crypto events
2500 - 2999 future user space (maybe integrity labels and related events)

Audit Event Record Types

ADD_GROUP
ADD_USER
ANOM_ACCESS_FS
ANOM_ADD_ACCT
ANOM_AMTU_FAIL
ANOM_CRYPTO_FAIL
ANOM_DEL_ACCT
ANOM_EXEC
ANOM_LOGIN_ACCT
ANOM_LOGIN_FAILURES
ANOM_LOGIN_LOCATION
ANOM_LOGIN_SESSIONS
ANOM_LOGIN_TIME
ANOM_MAX_DAC
ANOM_MAX_MAC
ANOM_MK_EXEC
ANOM_MOD_ACCT
ANOM_PROMISCUOUS
ANOM_RBAC_FAIL
ANOM_RBAC_INTEGRITY_FAIL
ANOM_SEGFAULT
AVC
AVC_PATH
CHGRP_ID
CONFIG_CHANGE
CRED_ACQ
CRED_DISP
CRED_REFR
CWD
DAC_CHECK
DAEMON_ABORT
DAEMON_CONFIG
DAEMON_END
DAEMON_ROTATE
DAEMON_START
DEL_GROUP
DEL_USER
EXECVE
FD_PAIR
FS_RELABEL
IPC
IPC_SET_PERM
KERNEL
KERNEL_OTHER
LABEL_LEVEL_CHANGE
LABEL_OVERRIDE
LOGIN
MAC_CIPSOV4_ADD
MAC_CIPSOV4_DEL
MAC_CONFIG_CHANGE
MAC_IPSEC_ADDSA
MAC_IPSEC_ADDSPD
MAC_IPSEC_DELSA
MAC_IPSEC_DELSPD
MAC_MAP_ADD
MAC_MAP_DEL
MAC_POLICY_LOAD
MAC_STATUS
MQ_GETSETATTR
MQ_NOTIFY
MQ_OPEN
MQ_SENDRECV
OBJ_PID
PATH
RESP_ACCT_LOCK
RESP_ACCT_LOCK_TIMED
RESP_ACCT_REMOTE
RESP_ACCT_UNLOCK_TIMED
RESP_ALERT
RESP_ANOMALY
RESP_EXEC
RESP_HALT
RESP_KILL_PROC
RESP_SEBOOL
RESP_SINGLE
RESP_TERM_ACCESS
RESP_TERM_LOCK
ROLE_ASSIGN
ROLE_REMOVE
SELINUX_ERR
SOCKADDR
TEST
TRUSTED_APP
USER
USER_ACCT
USER_AUTH
USER_AVC
USER_CHAUTHTOK
USER_CMD
USER_END
USER_ERR
USER_LABELED_EXPORT
USER_LOGIN
USER_LOGOUT
USER_MGMT
USER_ROLE_CHANGE
USER_SELINUX_ERR
USER_START
USER_UNLABELED_EXPORT
USYS_CONFIG

Aureport

- Utility that provides columnar reports on audit data
- Intended to be used for scripting more interesting reports from raw data
- Gives a summary report about what's been happening on your machine
- Each item in summary report leads to a report on that topic where summary or columnar data is given.
- Can read from stdin so that ausearch can pipe data to it

Aureport system summary

Summary Report
==========
Range of time in logs: 07/22/2006 08:29:01.394 - 05/07/2007 16:12:29.832
Selected time for report: 05/01/2007 00:00:01 - 05/07/2007 16:12:29.832
Number of changes in configuration: 85
Number of changes to accounts, groups, or roles: 2
Number of logins: 25
Number of failed logins: 1
Number of authentications: 29
Number of failed authentications: 1
Number of users: 2
Number of terminals: 11
Number of host names: 3
Number of executables: 59
Number of files: 3
Number of AVC denials: 46
Number of MAC events: 21
Number of failed syscalls: 16
Number of anomaly events: 33
Number of responses to anomaly events: 0
Number of crypto events: 0
Number of process IDs: 4087
Number of events: 5885

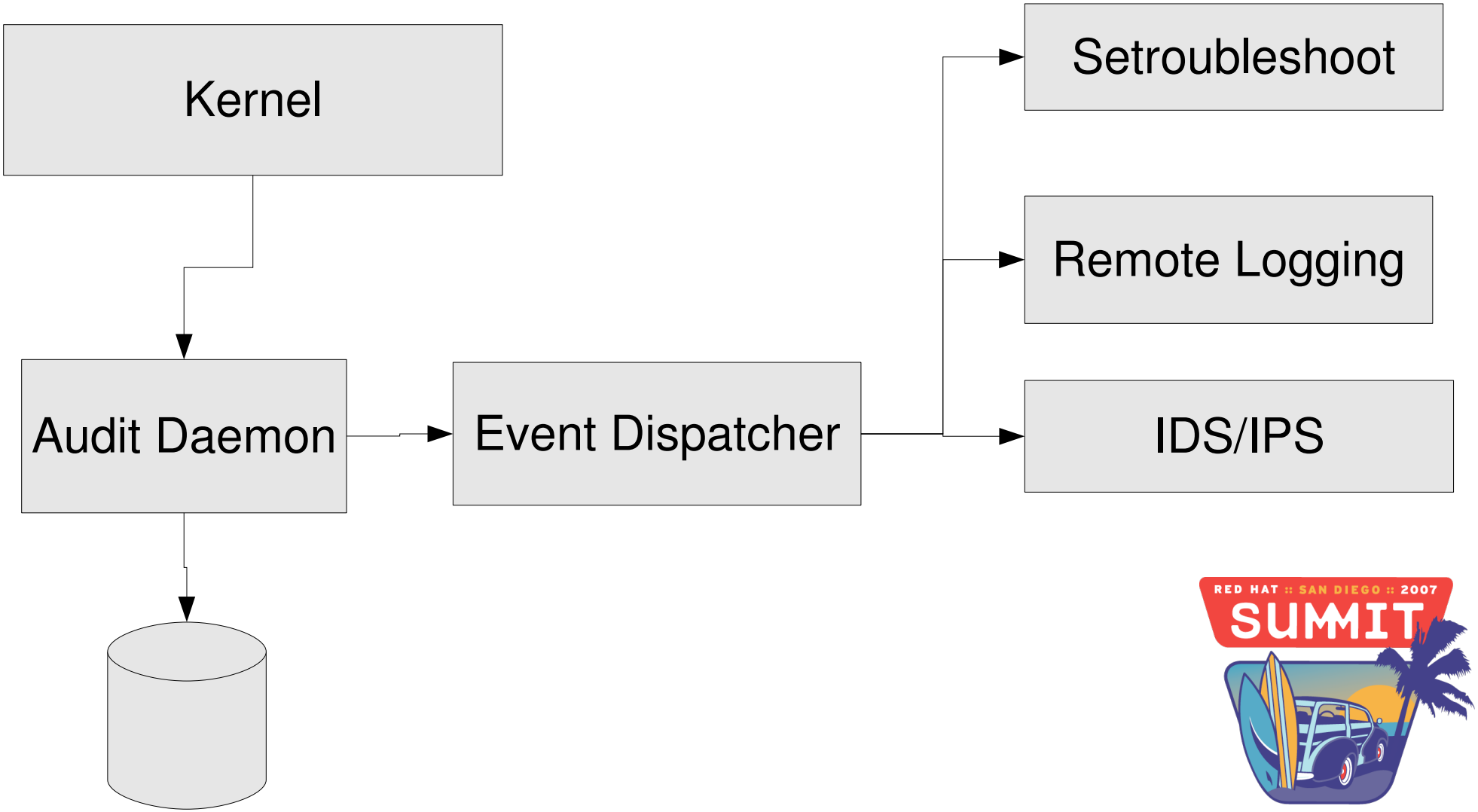
Aureport failed system summary

Failed Summary Report
==========
Range of time in logs: 07/22/2006 08:29:01.394 - 05/07/2007 16:12:29.832
Selected time for report: 05/01/2007 00:00:01 - 05/07/2007 16:12:29.832
Number of changes in configuration: 0
Number of changes to accounts, groups, or roles: 2
Number of logins: 0
Number of failed logins: 1
Number of authentications: 0
Number of failed authentications: 1
Number of users: 1
Number of terminals: 3
Number of host names: 1
Number of executables: 6
Number of files: 2
Number of AVC denials: 46
Number of MAC events: 0
Number of failed syscalls: 16
Number of anomaly events: 0
Number of responses to anomaly events: 0
Number of crypto events: 0
Number of process IDs: 15
Number of events: 54

Audit Event Dispatcher

- There was a desire to create a system where plugins that do different tasks could have access to audit data.
- Audit daemon must be very simple so that its code can be reviewed and fully understood so that it can pass at EAL4+.
- The audit daemon must not be vulnerable to attack by other processes
- Audit daemon has special SE Linux permissions
- This makes it not a good candidate for plugins

Audit Event Dispatcher Data Flow



Audit Event Dispatcher Plugins

- Programming rules
  - Must read from stdin
  - Must obey signals such as SIGHUP, SIGTERM
  - Must read config information from file
- Types of plugins
  - Input
    - Syslog, iptables events
  - Output
    - Remote logging, af_unix, protocol converters
  - Local
    - Event filter, setroubleshooter

Audit Parsing Library

- Design goals
  - Completely hide the log file format so that it can be changed over time
  - Abstract all internal data structures to make friendly to other languages
  - Create iterator approach like database libraries
  - Search API so that only records of interest can be found
  - Ability to translate from numeric values to human readable

Audit Parsing Library Example - C

```
auparse_state_t *au = auparse_init(AUSOURCE_FILE, "./test.log");
do {
    do {
        do {
            printf("%s=%s (%s)\n", auparse_get_field_name(au),
                auparse_get_field_str(au), auparse_interpret_field(au));
        } while (auparse_next_field(au) > 0);
    } while(auparse_next_record(au) > 0);
} while (auparse_next_event(au) > 0);
```

Audit Parsing Library Example - Python

```
au = auparse.AuParser(auparse.AUSOURCE_FILE, "./test.log");
while True:
    while True:
        while True:
            print "%s=%s (%s)" % (au.get_field_name(), au.get_field_str(), au.interpret_field())
            if not au.next_field(): break
        if not au.next_record(): break
if not au.parse_next_event(): break
```

Requirements for IDS/IPS

- The tools shall build upon audit reduction and analysis tools to aid the ISSO or ISSM in the monitoring and detection of suspicious, intrusive, or attack-like behavior patterns.
- The capability of the system to monitor occurrences of, or accumulation of, auditable events that may indicate an imminent violation of security policies.
- The capability of the system to notify the ISSO of suspicious events and taking the least-disruptive action to terminate the suspicious events.
- In real time

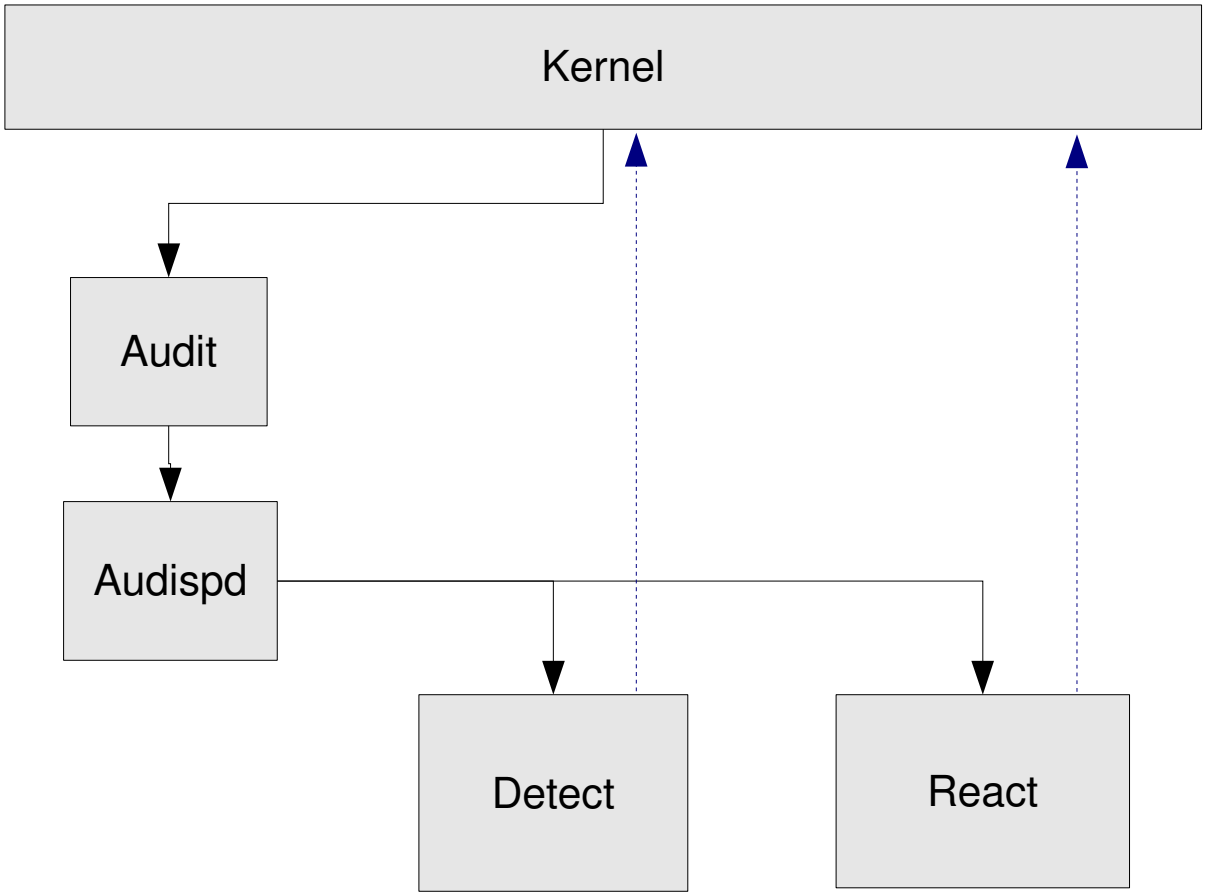
Audit Event Feeds

- Kernel
- Trusted Programs
  - Pam
  - Login, sshd, gdm
  - Shadow-utils, passwd
  - Semanage
- MAC selinux-policy
- Test Apps
  - Amtu
  - Rbac selftest
  - Aide
- (Security Scaning Tool)

Attacks

- Gain Entry to system
  - Login / exploit
    - Normal user
      - Access files or resources
      - Become root
        - Change trusted database
        - Add or modify account and passwords
        - Install programs
        - Start / stop services
        - Watch other users
        - Kill audit system
        - Sniff traffic
        - Gain entry to other systems

IDS/IPS System



Attacks – anomaly record types

- Gain Entry to system
  - Login / exploit
    - AUDIT_ANOM_LOGIN_FAILURES - Failed login limit reached
    - AUDIT_ANOM_LOGIN_TIME - Login attempted at bad time
    - AUDIT_ANOM_LOGIN_SESSIONS - Max concurrent sessions reached
    - AUDIT_ANOM_LOGIN_ACCT - Login attempted to watched acct
    - AUDIT_ANOM_LOGIN_LOCATION - Login from forbidden location
    - AUDIT_ANOM_ABEND - Process ended abnormally
    - AUDIT_ANOM_MAX_MAC - Max MAC failures reached

Attacks – anomaly record types

- Access files or resources
  - AUDIT_ANOM_MAX_DAC - Max DAC failures reached
  - AUDIT_ANOM_MAX_MAC - Max MAC failures reached
  - AUDIT_ANOM_ACCESS_FS - Access of file or dir
  - AUDIT_ANOM_EXEC - Execution of program
- Become root
  - AUDIT_ANOM_ROOT_TRANS
- Change trusted database
  - AUDIT_ANOM_ACCESS_FS - Access of file or dir
  - AUDIT_ANOM_AMTU_FAIL - AMTU failure
  - AUDIT_ANOM_RBAC_FAIL - RBAC self test failure
  - AUDIT_ANOM_RBAC_INTEGRITY_FAIL - RBAC file integrity

Attacks – anomaly record types

- Add or modify account and passwords
  - AUDIT_ANOM_ADD_ACCT - Adding an acct
  - AUDIT_ANOM_DEL_ACCT - Deleting an acct
  - AUDIT_ANOM_MOD_ACCT - Changing an acct
- Install programs
  - AUDIT_ANOM_MK_EXEC - Make an executable
- Start / stop services
  - AUDIT_ANOM_EXEC - Execution of file
- Watch other users
  - AUDIT_ANOM_ACCESS_FS - Access of file or dir
  - AUDIT_ANOM_MK_EXEC - Make an executable
-

Attacks – anomaly record types

- Kill audit system
  - AUDIT_ANOM_RBAC_FAIL - RBAC self test failure
- Sniff traffic
  - AUDIT_ANOM_PROMISCUOUS - Device changed promiscuous mode
- Gain entry to other systems
  - We would have to correlate logging from all machines

Attack reaction types

- AUDIT_RESP_ANOMALY - Anomaly not reacted to
- AUDIT_RESP_ALERT - Alert email was sent
- AUDIT_RESP_KILL_PROC - Kill program
- AUDIT_RESP_TERM_ACCESS - Terminate session
- AUDIT_RESP_ACCT_REMOTE - Acct locked from remote access
- AUDIT_RESP_ACCT_LOCK_TIMED - User acct locked for time
- AUDIT_RESP_ACCT_UNLOCK_TIMED - User acct unlocked from time
- AUDIT_RESP_ACCT_LOCK - User acct was locked
- AUDIT_RESP_TERM_LOCK - Terminal was locked
- AUDIT_RESP_SEBOOL - Set an SE Linux boolean
- AUDIT_RESP_EXEC - Execute a script
- AUDIT_RESP_SINGLE - Go to single user mode
- AUDIT_RESP_HALT - take the system down

Configuring the IDS/IPS system

```
# Failed login limit reached
AUDIT_ANOM_LOGIN_FAILURE_ENABLE = true
AUDIT_ANOM_LOGIN_FAILURE_LIMIT =  5
AUDIT_ANOM_LOGIN_FAILURE_INTERVAL = 10
AUDIT_ANOM_LOGIN_FAILURE_RESPONSE = AUDIT_RESP_ANOMALY

# Login attempted to watched acct
AUDIT_ANOM_LOGIN_ACCT = true
AUDIT_ANOM_LOGIN_ACCT_USER = root ftp daemon
AUDIT_ANOM_LOGIN_ACCT_RESPONSE = AUDIT_RESP_ANOMALY
```

Configuring the IDS/IPS system

```
# Access of file or dir
AUDIT_ANOM_ACCESS_FS = true
AUDIT_ANOM_ACCESS_FS_FILES = /etc/passwd /var/log/*
AUDIT_ANOM_ACCESS_FS_EXCEPTION_USERS = root sgrubb
AUDIT_ANOM_ACCESS_FS_EXCEPTION_GROUPS = wheel root
AUDIT_ANOM_ACCESS_FS_RESPONSE = AUDIT_RESP_ANOMALY

# Execution of file
AUDIT_ANOM_EXEC = true
AUDIT_ANOM_EXEC_BINARIES = /usr/bin/sudo /bin/su /bin/nc
AUDIT_ANOM_EXEC_EXCEPTION_USERS = root sgrubb
AUDIT_ANOM_EXEC_EXCEPTION_GROUPS = wheel root
AUDIT_ANOM_EXEC_RESPONSE = AUDIT_RESP_ANOMALY

# Make an executable
AUDIT_ANOM_MK_EXEC = true
AUDIT_ANOM_MK_EXEC_EXCEPTION_USERS = root sgrubb
AUDIT_ANOM_MK_EXEC_EXCEPTION_GROUPS = root wheel
AUDIT_ANOM_MK_EXEC_RESPONSE = AUDIT_RESP_ANOMALY
```

Configuring Reactions

- Still under design
- Will have several little programs to perform each response
- Will pass command line variables such as pid, user, group, anomaly type, tty, or host name

Questions?

Email: sgrubb@redhat.com

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

Mail list: linux-audit@redhat.com