Practical Security Event Auditing with FreeBSD

Christian Brüffer

brueffer@FreeBSD.org

NYCBSDCon – New York City, USA

November 14, 2010

Outline

1. What / What for / Why
2. Format
3. Configuration and Operation
4. Processing Tools and Audit Pipes
5. Demonstration
6. Considerations and Caveats
7. Summary

What is Security Event Auditing

- Logging of security-relevant system events
  - Secure
  - Reliable
  - Fine-grained
  - Configurable
- Resulting logs are called trail files

Straightforward Uses

- Post-mortem analysis
- Intrusion Detection
- Live system monitoring
- Debugging

Creative Uses

- Audit and rsync as a poor man's cluster file system
- From a German blog entry:

  http://blog.elitecoderz.net/de/cluster-filesystem-for-freebsd-gfs-ocfs2/2010/06/
- Watching audit trail for file system activity
- Syncing changes via rsync

Why?

- “*Detection is much more important than prevention*” - Bruce Schneier
- No system is 100% secure
- Accountability
- Mandatory for some security evaluations

Common Criteria and CAPP

- CAPP = Controlled Access Protection Profile
- Defines functional requirements
- Security evaluation needed for certain uses and users (e.g. US DoD)
- Audit implementation required for CAPP compliance

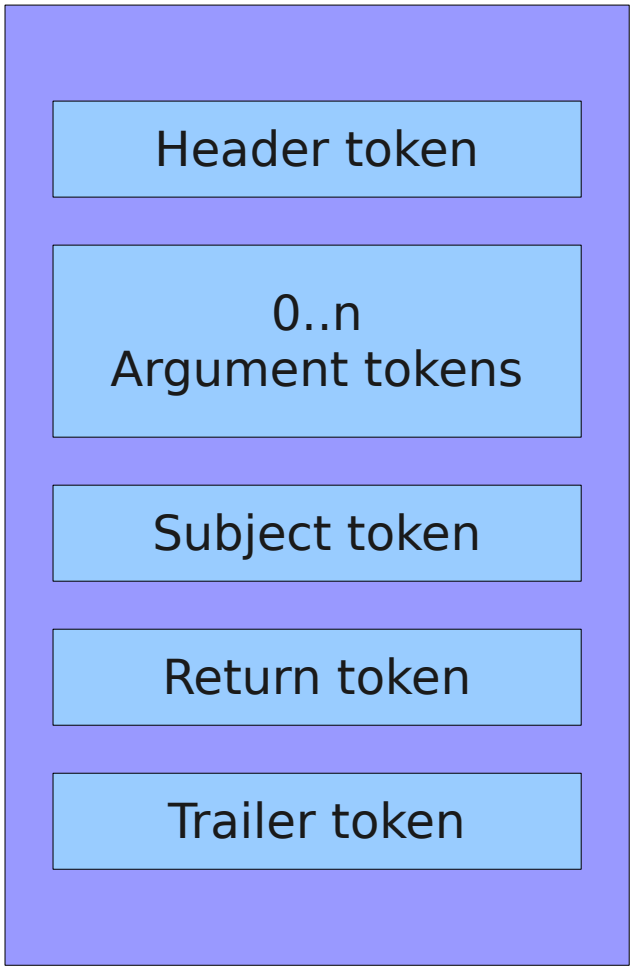
FreeBSD Audit History

- Originally by McAfee for Apple's Mac OS X
- Relicensed to 3-clause BSD license
- Ported to FreeBSD by TrustedBSD project
- Experimental feature in FreeBSD 6.2
- Production feature in FreeBSD 6.3

BSM – Basic Security Module

- Defined by Sun Microsystems
- Standard audit file format:
  - Solaris / OpenSolaris
  - Mac OS X
  - FreeBSD
- Binary format
- Token-based record stream

BSM Record Format




```
header,98,11,OpenSSH login,0,Sat Nov 06
09:50:52 2010, + 559 msec
subject_ex,chris,chris,chris,chris,chris,1
083,1083,53331,192.168.56.1
test,successful login chris
return,success,0
trailer,98
```

```
header,121,11,execve(2),0,Sat Nov 06
06:50:52 2010, + 588 msec
exec arg,-tcsh
path,/bin/tcsh
attribute,555,root,wheel,61,30026,122312
subject,chris,chris,chris,chris,chris,1087
,1083,53331,192.168.56.1
return,success,0
trailer,121
```

→ see audit.log(5) for token documentation

Configuration Files

- /etc/security/audit_event
  - Event definitions and mapping to classes
- /etc/security/audit_class
  - Class definitions
- /etc/security/audit_control
  - Global configuration
- /etc/security/audit_user
  - Per-user configuration
- /etc/security/audit_warn
  - Script with handlers for audit warnings

Config Files Demonstration

Handling the Daemon

- No kernel re-compile necessary since 7.0
  - # echo 'auditd_enable=”YES”' >> /etc/rc.conf
  - # /etc/rc.d/auditd start
- Shutdown
  - # /etc/rc.d/auditd stop

    or
  - # audit -t
- Config changes:
  - # audit -s

Audit Trail Files

- Standard location /var/audit
- Group *audit* allows trail access delegation
- File name format:
  - starttime.endtime (time in UTC)
  - 20101013181949.20101013183539
- In some cases *starttime.not_terminated*
  - File active, or auditd shut down improperly
- *current* → symlink to currently active trail

OpenBSM

- User space part of FreeBSD audit
- Open source implementation of BSM
  - API
  - file format
  - basic tools
- BSD license
- Portable
- http://www.openbsm.org

OpenBSM Tools

- audit(8)
  - Control utility for auditd
- auditd(8)
  - Manages trail files
  - Communicates with kernel by /dev/audit
- auditreduce(1)
  - Returns selected records from a trail file
- praudit(1)
  - Prints a trail file in human-readable form

Audit Pipes

- tee(1) for audit stream
- */dev/auditpipe* → clonable special device
- Live system monitoring

```
# praudit /dev/auditpipe
```

- Reliability not guaranteed
  - As opposed the trail files
- Applications can select which events they're interested in

Demonstration

Considerations

- Performance overhead
  - more auditing → higher penalty
  - essentially no overhead when disabled
- Space requirements
  - dependent on configuration
  - system activity
  - can be terabytes on a busy system

Caveats

- Some subsystems do not generate audit records yet (NFS, Packet Filters, ...)
- No per-Jail audit trails
  - Audit records go straight to the host system trail file
  - “zonename” token not utilized yet
- No open source distributed log daemon

Documentation

- Handbook

  http://www.freebsd.org/doc/en_US.ISO885

  9-1/books/handbook/audit.html

- Manpages:
  - auditreduce(1), praudit(1), libbsm(3), audit(4), auditpipe(4), audit_class(5), audit_control(5), audit_event(5), audit_user(5), audit.log(5), audit(8), auditd(8), ...

3rd Party Software

- bsmGUI (http://bsmgui.sf.net/)
- BSM Analyzer (http://bsma.sf.net)
- BSMTrace (ports/security/bsmtrace)
- Splunk + BSM Audit log loader
- Snare + Snare Agent for Solaris
- all other BSM compatible software

Google Summer of Code (1)

- Projects in 2007 and 2008:
  - Distributed Audit Daemon (2007)
    by Alexey Mikhailov, mentor: bz@
  - Audit Log Analysis Tool (2007)
    by Dongmei Lui, mentor: rwatson@
  - Auditing System Testing (2008)
    by Vincenzo Iozzo, mentor: attilio@
  - Audit Firewall Events from Kernel (2008)
    by Diego Giagio, mentor csjp@

Google Summer of Code (2)

- Projects in 2009 and 2010
  - Application-Specific Audit Trails (2009)
    by Ilias Marinos, mentor rwatson@
  - Log Conversion Tools to/from Linux (2009)
    by Satish Srinivasan, mentor sson@
  - Audit Kernel Events (2010)
    by Efstratios Karatzas, mentor rwatson
  - Distributed Audit Daemon (2010)
    by Sergio Ligregni, mentor sson@

Summary

- Audit can be an important piece in a security infrastructure
- API / file format compatible with Solaris
- BSM format → compatibility with existing tools for free
- Tradeoff log detail ↔ performance and space requirements

Questions?

Thanks...

...for your attention

...and to Robert Watson for lots of material for this talk