

Re: HEADS UP: Audit integration into CVS in progress, some tree disruption (fwd)

## Re: HEADS UP: Audit integration into CVS in progress, some tree disruption (fwd)

---

*Source:* <http://www.derkeiler.com/Mailing-Lists/FreeBSD-Security/2006-02/msg00001.html>

---

- *From:* Robert Watson <[rwatson@xxxxxxxxxxxxx](mailto:rwatson@xxxxxxxxxxxxx)>
  - *Date:* Thu, 2 Feb 2006 09:58:55 +0000 (GMT)
- 

FYI, since this is probably of interest to subscribers of this mailing list also.

Robert N M Watson

----- Forwarded message -----

Date: Wed, 1 Feb 2006 22:55:40 +0000 (GMT)  
From: Robert Watson <[rwatson@xxxxxxxxxxxxx](mailto:rwatson@xxxxxxxxxxxxx)>  
To: Julian Elischer <[julian@xxxxxxxxxxxxx](mailto:julian@xxxxxxxxxxxxx)>  
Cc: trustedbsd-audit@xxxxxxxxxxxxxxx,  
Kövesdán Gábor <[gabor.kovesdan@xxxxxxxxxxxxxxx](mailto:gabor.kovesdan@xxxxxxxxxxxxxxx)>, current@xxxxxxxxxxxxxxx  
Subject: Re: HEADS UP: Audit integration into CVS in progress,  
some tree disruption

On Wed, 1 Feb 2006, Julian Elischer wrote:

I'll send out follow-up e-mail once the worst is past, along with information on what it all means, and how to try it out (for those not already on trustedbsd-audit, who have been hearing about this for a while).

Do you plan to merge it to RELENG\_6? If so, when? Maybe for the upcoming 6.1? Or only for 6.2 or later?

is there a website about all this stuff? "What's it for?"

Re: HEADS UP: Audit integration into CVS in progress, some tree disruption (fwd)

I'm sure I promised to answer exactly that question in my followup e-mail once the integration is done. :-)

The quick answer is that this is an implementation of security event auditing, as required by the Orange Book C2 and later Common Criteria CAPP security evaluation/standard. These documents provide specifications for a set of functional requirements (and assurance requirements) regarding the behavior of operating systems with respect to security. One of the requirements is the fine-grained and configurable logging of security-relevant events. Security-relevant turns out to be pretty all-inclusive, as CAPP requires the ability to log the results of access control decisions associated with discretionary access control, which means basically all file I/O, including path lookups. So what is present in our implementation is:

- The introduction of a centralized kernel audit event engine, `src/sys/security/audit`, which includes various system calls, an event queue, kernel worker thread to process the queue, interfaces to capture system call information, a system call for user applications to submit audit records, pre-selection mechanism, etc.
  
- OpenBSM, an implementation of the Solaris/OpenSolaris Basic Security Module API and file format for audit trails. This is derived from the BSM audit support found in the Apple Mac OS X and Darwin operating systems, although substantially reworked, cleaned up, and synchronized to recent BSM changes in Solaris, such as 64-bit records.
  
- `auditd`, a daemon for managing audit event logs and the audit subsystem.
  
- Modifications throughout the kernel and in many places in user space to generate audit records.

Unlike existing logging and tracing mechanisms, audit has to meet a number of reliability, security, and functional requirements that basically drove the implementation of a new logging system rather than adaptation of an existing one:

- Only authorized processes can read and write to the audit log.
  
- Detailed subject and object information, including file paths, full credential information for processes, etc.

Re: HEADS UP: Audit integration into CVS in progress, some tree disruption (fwd)

- Configurable log granularity by user, subsystem, operation, including the ability to control the logging of non-attributable events.
- Audit log reduction tools and pre-selection mechanism.
- Reliability requirements relating to maximum record loss in the event of power loss, configurable ability to fail-stop the system when the audit store is filled.
- Portable log format based on the de facto industry standard BSM format (used by Solaris, Mac OS X, and a moderate number of intrusion detection tools, post-mortem tools, etc).

The implementation is not yet fully complete, but it's now at the point where more broad exposure and testing would be very helpful. The hope is to have much of the current implementation merged in the next couple of days, and the remainder over the next couple of weeks.

Since I did the intro for this, I should take this opportunity to thank Apple Computer for sponsoring the original development work as part of their Common Criteria CAPP evaluation for Mac OS X, and then releasing the results under a BSD license (announcement on this to follow), SPARTA for releasing extensions and additional work on the system, not to mention the team of people who have been involved in porting over, adapting, and substantially enhancing the Darwin audit support, including Wayne Salamon (part of the original audit development team), who has done extensive development work on it, and Tom Rhodes, who has written a lot of the new documentation including a new handbook chapter on configuring audit support.

Robert N M Watson

---

freebsd-current@xxxxxxxxxxx mailing list  
<http://lists.freebsd.org/mailman/listinfo/freebsd-current>  
To unsubscribe, send any mail to "freebsd-current-unsubscribe@xxxxxxxxxxx"

---

freebsd-security@xxxxxxxxxxx mailing list  
<http://lists.freebsd.org/mailman/listinfo/freebsd-security>  
To unsubscribe, send any mail to "freebsd-security-unsubscribe@xxxxxxxxxxx"