Global Information Assurance Certification Paper

BSM Security Auditing for Solaris Servers

GIAC Security Essentials Certification Practical
John Sun
Version 1.4 (option 1)
Submitted January 3, 2003

Abstract

Although Solaris servers might be inside the firewall and relatively secure, there are still chances for a hacker to break in, or chances for an ordinary user to attempt malicious activities. Therefore, security efforts have to be made to detect intruders and to prevent unauthorized actions.

One of the security utilities for Solaris servers is called BSM (Basic Security Module), which is an auditing tool for system security provided by SUN Microsystems. We can make use of it to increase security on our Solaris systems.

This article discusses the pros and cons of BSM. It describes how to enable, configure, and manage the BSM auditing on Solaris servers to secure the system. Based on the author's experience, this article also gives a few solutions to overcome some problems and disadvantages of BSM.

Overview of BSM

Solaris is a SUN Microsystems's OS product in the UNIX world. Solaris is heavily used and provides excellent networking solutions for both private and government sectors.

BSM is a subsystem under the Solaris Operating Environment and it has been a feature of this Environment since Solaris version 2.5. The full name of the subsystem is SolarisOE SunSHIELD™ Basic Security Module. This auditing tool was added to Solaris to provide the features required by the Trusted Computer System Evaluation Criteria (TCSEC) to a security level referred to as C2.

The TCSEC has been superseded by the newer and more internationally recognized Common Criteria security requirements. The Solaris 8 Operating Environment is certified at Evaluated Assurance Level 4 (EAL4) under the Controlled Access Protection Profile (CAPP) of the Common Criteria IT security evaluation. Basically, this means that the Solaris Operating Environment has been tested and verified to meet security standards set for operating systems that allow user discretionary access control.

The National Security Agency (NSA) National Computer Security Center (NCSC) has defined the computer security levels in the "Orange Book" (http://csrc.nist.gov/secpubs/rainbow/nsaorder.txt). These levels are graded on a scale of A (most secure) through D (least secure). Using default settings, most UNIX systems are rated at the C1 level (security discretionary level). Since SUN Microsystems has provided the new auditing tool BSM, the Solaris system can be upgraded to the security level C2 (added auditing capability and access control):

> The Basic Security Module (BSM) is an Audit Module that meets the C2-level security specifications as defined in the NCSC 'Orange Book' required for government customers.
> The C2-compliant Audit Module is also attractive to commercial customers who want to trace system activities to individual users, providing individual accountability for actions on the system and to detect malicious or security-relevant activity. [1]

BSM auditing enables system administrators to monitor the actions of the users. The auditing mechanism enables an administrator to detect potential security breaches. The auditing can also reveal suspicious or abnormal patterns of system usage and provides the means to trace suspect actions back to a specific user.

BSM audit records are initiated from two distinct places: privileged user land programs (such as login) and the Solaris kernel. All security sensitive kernel system calls will generate an audit record when BSM auditing is enabled. Since BSM does kernel auditing, it has changed as Solaris kernel has changed, but nothing really seen by the user. The advantage of an auditing tool processing at the kernel level is that the possibility of tampering with the auditing data by unauthorized intruders is greatly reduced.

The tradeoff of using BSM is that the system performance could be reduced depending on how many events are logged. In general, BSM is very small at the kernel level, and there is no serious impact on the system performance if we only use certain functions. But Performance can be an issue if we are writing a lot of audit events. Since the speed of Solaris servers is getting faster and faster, the performance impact will become less and less. The other disadvantage is that the size of audit log file is usually big and it requires more disk space. The hard disk partition usage must be regularly scrutinized to avoid filling up disk allocations. The output of BSM is binary which requires more effort to use. Some of these disadvantages can be overcome under more skillful system management, and it will be discussed later in this article.

Enable BSM

[1] SUN Microsystems, "Security - Basic Security Module (BSM)", p.3

BSM has been included in the full release and is part of the release media of SUN Solaris Operating System. We do not need to install BSM separately. All of the BSM software is included in the initial system installation, provided the following packages are installed with Solaris OS:

SUNWcar – Solaris core architecture

SUNWcsr - core SPARC

SUNWcsu - core SPARC

SUNWhea - header files

SUNWman - man pages [2]

BSM auditing is not enabled by default in the Solaris Operating Environment when installed. To enable BSM, bring the system into the single-user mode, change directories to /etc/security, and execute the “bsmconv” script. The script sets up a standard Solaris machine to run BSM after a reboot:

```
#  cd /etc/security
# ./bsmconv
#  init 6
```

This creates /etc/security/audit_startup which causes the audit daemon “auditd” to run. To check look for a running auditd, its pid is the first field of /etc/security/audit_data. [3]

To find out if BSM is enabled, type the command:

```
# /usr/sbin/auditconfig –getcond
```

We should be getting a response like “audit condition = auditing” if it is enabled.

Another command to check the enabled BSM is

```
# /usr/sbin/modinfo | fgrep c2audit
```
[4]

We should find the C2 system call “c2audit” if BSM is enabled.

After enabling BSM, the volume manager can still be used and nothing is affected.

---

[2] Peter H. Gregory, p.260
[3] Darren J Moffat, p.1
[4] Anupam, p.1

Configure BSM

Security-relevant actions can be audited. The system actions that are auditable are defined as audit events in the /etc/security/audit_event file.

Each audit event is also defined as belonging to an audit class or classes. By assigning events to classes, an administrator can more easily deal with large numbers of events. When naming a class, we simultaneously address all of the events in that class. The mapping of audit events to classes is configurable and the classes themselves are configurable. These configuration changes can also be made in the audit_event file.

BSM records data based on audit classes. These classes and their corresponding flags are defined in the file /etc/security/audit_class. One example of a flag and its meanings is:

lo login_logout Login and logout events

There is a special flag 'all' that means 'all events'.

The following are some of the predefined audit classes: [5]

Class Flag	Class Name	Description
fr	file_read	Read of data, open for reading, and so forth
fw	file_write	Write of data, open for writing, and so forth
fm	file_attr_mod	Change of object attributes: chown, flock, etc.
fc	file_creation	Creation of object
fd	file_deletion	Deletion of object
pc	process	Process operations: fork, exec, exit, and so forth
nt	network	Network events: bind, connect, accept, and so forth
ip	ipc	System V IPC operations

[5] SUN Microsystems, **"**Audit Flags - Definitions of Audit Flags", p.1

Class Flag	Class Name	Description
na	non_attrib	Nonattributable events
ad	administrative	Administrative actions
lo	login_logout	Login and logout events
ex	exec	Program execution
all	all	All flags set

Flags can be qualified with:

+ (meaning 'successful')
- (meaning 'failed') and
^ (meaning except).

Thus:

+lo means successful login and logout events
-lo means failed login and logout events
all,^+lo means all events except successful logins and logouts

In order to audit system-wide events, we need to add the flags that represent the events we wish to audit to the audit_control file. Here is the default:

```
dir:/var/audit
flags:
minfree:20
naflags:lo
```

This says that data will be stored in the directory /var/audit, no flags are set, and when 20% percent of the audit space is still available a warning script will be run to notify the administrator to archive or delete data.

So, in order to log all login and logout activity, we would want to change the flags line to:

```
flags:lo
```

The audit_control file on each machine is read by the audit daemon when auditing is enabled. The audit_control file is located in the /etc/security directory. The administrator creates an audit_control file during the configuration process on each machine. After the audit_control file is created during system

configuration, the administrator can edit it. After a change, the administrator runs the “audit –s” command to instruct the audit daemon to reread the audit_control file.

If we want to log events of a specific user, we need to edit the /etc/security/audit_user file. This file takes the form:

username:always-audit-flags:never-audit-flags

where always-audit-flags are the flags specifying event classes we wish to always audit (for username), and never-audit-flags are the flags specifying events that we wish to never audit (for username). Thus:

jsun:lo:+fr

means whatever the system-wide policy is, for user ‘jsun’ always log login and logout events, but never log successful file reads.

Start BSM Auditing

Auditing is started by bringing up the BSM daemon.

The existence of a file with the path name /etc/security/audit_startup causes the audit daemon to be run automatically when the system enters multi-user mode. The file is actually an executable script that is invoked as part of the startup sequence just prior to the execution of the audit daemon. This script is set up during the BSM package enabling.

The BSM audit daemon can also be started manually by executing /usr/sbin/auditd as root.

Query BSM Data

BSM doesn't store its data in easy to read ASCII files like syslog has accustomed us to. BSM uses files that contain binary data, stores them in the directory specified in the audit_control file (/var/audit by default), and gives them cryptic names like:

yyyymmddhhmm.yyyymmddhhmm.hostname

For example,

200205191639.200205191646.dbserver

The first field is the start date and time, the second is the date and time the file was terminated, and the last is the hostname being audited. If a data file has not yet been terminated, the filename would look something like:

200205191639.not_terminated.dbserver

Since these files contain binary data, we must use specific utilities in order to get any useful information out of them. The system provides two utilities for viewing (praudit) and filtering (auditreduce), and the utilities are part of the SUNWcsu package (core solaris user).

To get all the entries out the file “200205191639.200205191646.dbserver”, we can use the command:

praudit 200205191639.200205191646.dbserver

To make the output quite a bit more readable, use the -l flag. This converts the record type and even fields to ASCII and puts one record on one line:

praudit -l 200205191639.200205191646.dbserver

We can also save the ASCII output into a readable file, say, audit_output:

praudit –l 200205191639.200205191646.dbserver > audit_output

Each audit record in the audit file describes the occurrence of a single audited event and includes such information as who did the action, which files were affected, what action was attempted, and where and when it occurred.

After the audit data is collected, the BSM audit-reduction and interpretation tools allow the examination of interesting parts of the audit trail. For example, we can choose to look at audit records for individual users or groups, look at all records for a certain type of event on a specific day, or select records that were generated at a certain time of day.

Usually we want to query all of the logs. To do this, we make use of “auditreduce” utility. To see all logged event:

auditreduce | praudit –l

“auditreduce” will also let us be very specific about our queries. For instance, if we wanted to see only login and logout events we can do:

```
# auditreduce -c lo | praudit –l
```

Other parameters used with auditreduce are following:

-c Event class
-u Real UID
-a Events occurring after the specified time
-b Events occurring before the specified time
-e Effective UID
-g Real GID
-f Effective GID

To see all of the login and logout events by user jsun, we could do:

```
# auditreduce -u jsun -c lo | praudit -l
```

To see all of the login and logout events by user jsun that occurred on or after May 19th 2002:

```
# auditreduce -a 20020519 -u jsun -c lo | praudit -l
```

And finally to see all of the login and logout events by user jsun that occurred during the month of September 2002:

```
# auditreduce -a 20020901 -b +31d -u jsun -c lo | praudit –l
```

The above utilities “auditreduce” and “praudit” are the only tools that SUN Microsystems provides for reading the audit data files. There is a free tool “Basic Security Module GUI” available. The BSM GUI provides an interface enabling the user to configure custom audit queries against the /var/audit log files created by the BSM. The detail information of BSM GUI can be found at URL: http://home.twmi.rr.com/jayd/bsm.html.

Manage BSM data Files

BSM audit data files may become very big in size depending on the audit policies. This could cause a problem that the file system containing the audit data becomes full. After the file system is full, no more data could be stored and no more files could be created in the file system. If the audit data is stored by default under /var and the /var file system becomes full, then, nobody could access /var to edit a file using the vi editor because vi uses /var file system to store the editing file temporarily.

In one of our Solaris system, our audit data files were saved by default in the /var/audit directory. One of the audit data files under /var/audit directory was so

big that it took about 2.5 GB disk space, and it made the /var file system 100% full. The users complained that they could not be able to edit any files there.

When the audit data file system has less space left than the value “minfree” defined in the audit_control file, warning messages will be sent to the system, for example, as following:

```
/etc/security/audit_warn: Soft limit exceeded in file
/var/audit/20021023180315.not_terminated.dbserver.
/etc/security/audit_warn: Soft limit exceeded on all filesystems.
```

To keep the audit files at a manageable size and to prevent the disk from running out of space, a cron job can be set up to periodically purge the old audit files or to move the audit files and save them in a different file system.

The followings are some of the solutions to this disk space problem:

1. Increase the size of the file system for auditing data.

   However, no matter how big the file system is, it will be full someday because new audit data is generated every day and the audit file becomes bigger and bigger and will make the file system full. Therefore, this is a temporary solution.

2. Decrease the value of the “minfree” in the audit_control file.

   For example, we can change it from the default value 20% to 10% so that we can avoid the warning messages or get the warning messages later.

3. Move some of the old audit data files to other file systems.

   It is also a temporary solution because the disk spaces in a Solaris server are also limited.

4. Archive the old audit data files and save them into backup tapes, then, delete the old audit data files.

I have fulfilled the tasks listed in item 2 and 4 for some of our Solaris servers. Instead of manually removing the old audit data files, a script was written to checks the date of the audit data files and to remove them when they were, say, more than 100 days old.

Since the audit data files were saved by default in the /var/audit directory on one of our Solaris servers, I had the following experience: after I moved the /var/audit directory to a different file system to prevent /var file system from being full, I created a symbolic link pointing to the different file system from /var; this freed

part of the disk spaces in the /var file system; but, when I checked the disk space by using the command

```
# df –k /var
```

the output did not show that the disk spaces in the /var file system had been freed. To let the change be recognized by the system, the BSM audit daemon has to be refreshed. This could be done by the following command without shutting down the server:

```
# pkill –HUP auditd
```

Combine BSM with SSH and Cron

SSH, Security Shell, is one of the world's leading Internet-based data security technologies and solutions. It is a program to log into another computer over a network, to execute commands in a remote machine, and to move files from one machine to another. It provides cryptography and strong authentication for Internet and Intranet secure communications over unsecured channels. It is intended as a replacement for rlogin, rsh, and rcp that are vulnerable to different kinds of attacks. [6]

SSH (OpenSSH version 3.4) has been installed, configured, and running on our Solaris servers.

Cron is a UNIX clock daemon for scheduling system jobs. It is executed upon system initialization and remains active while the system is operating in multi-user mode. Cron wakes up every minute and examines all the stored configuration files to check each of them for commands that scheduled to be executed at the current time.[7] Regularly scheduled commands can be specified according to instructions found in crontab files in the directory /var/spool/cron/crontabs.

Cron jobs could be set up to monitor and manage our Solaris systems for the security purpose. It could also be set up to run our script to purge the old BSM audit data files automatically at midnight everyday to make sure our audit data file system is not full.

I have created a cron job that starts a Perl script to remove the very old audit log files and send an email to notice the system administrators and the managers.

But there is a confliction among BSM enabler, SSH, and Cron: by default, they could not be able to work together. Since both BSM and SSH were installed on our server, the Cron job I have created could not work there. To solve the

[6] Thomas Koenig, p.1
[7] Indiana University, “Overview of the cron daemon”, p.1

problem, I worked around by modifying the SSH configuration file /usr/local/etc/sshd_config:

OpenSSH provides for using Solaris login with a "UseLogin" option. In the SSH configuration file “sshd_config”, the default value of "UseLogin" is set to "no", and it can be changed to "yes". By setting UseLogin to "yes", the same policies are applied to SSH as well as to other system accesses, thus, the correct audit characteristics are set and the Cron job works.

Conclusion

Computer security is playing an increasingly important role in Solaris servers as more and more sensitive data is stored, and as computer networks become more and more widespread. By using BSM and the related security utilities described in this article, the Solaris system could be made more secure.

In order to detect intruder and prevent possible security problems, we could configure and manage BSM and its related utilities wisely to reach our specific security goal. Although BSM has a few disadvantages like any other tools, it is still good security facilities for the Solaris servers. As qualified Solaris system administrators, we should have the ability to do research to find the way to resolve the problem or confliction related to BSM.

References

Rogers, Russ. “Solaris Auditing Overview”. 18 Nov. 1999
URL: http://www.securityhorizon.com/whitepapers/archives/bsm.html

Osser, William. “Auditing in the Solaris™ 8 Operating Environment”. Feb. 2001
URL: http://www.sun.com/blueprints/0201/audit_config.pdf

SUN Microsystems. “Security - Basic Security Module (BSM)”
URL: http://wwws.sun.com/software/solaris/2.6/ds-security.html

Gregory, H. Peter, Solaris Security. Prentice-Hall, Upper Saddle River, New Jersey, 2000. Pages 91-94, 259-264

SUN Microsystems. “Administering Auditing”
Chapter 2 in “SunSHIELD Basic Security Module Guide”
URL: http://docs.sun.com/db/doc/806-1789/6jb25l4af?a=view

Moffat, J. Darren. “Solaris BSM Auditing”. 27 Nov. 2000
URL: http://online.securityfocus.com/infocus/1362

SUN Microsystems. “Security Audit - Solaris SunSHIELD Basic Security Module” URL: http://wwws.sun.com/software/security/audit/

Mulligan, P. John. “Enabling the Basic Security Module (BSM)”. 11 July 2001 URL: http://searchsolaris.techtarget.com/tip/1,289483,sid12_gci754085,00.html

Moffat, J. Darren. “Re: Solaris Basic Security Module”. 17 Sep. 2002. URL: http://archives.neohapsis.com/archives/sf/sun/2002-q3/0090.html

Paul, Greene. “Re: Solaris Basic Security Module”. 17 Sep. 2002. URL: http://citadelle.intrinsec.com/mailing/current/HTML/ml_focus_sun/0727.html

Anupam. “RE: Solaris Basic Security Module”. 20 Sep. 2002. URL: http://online.securityfocus.com/archive/92/292684/2002-09-18/2002-09-24/2

SUN Microsystems. **“**Audit Flags - Definitions of Audit Flags” URL: http://docs.sun.com/db/doc/806-1789/6jb25l4ak?a=view

SUN Microsystems. **“**Solaris Security Guide” URL: http://sabernet.home.attbi.com/papers/Solaris.html

Koenig, Thomas. “Ssh Basics - What is ssh”. 6 June 1997. URL: http://www.dreamwvr.com/ssh-faq/ssh-faq-2.html#ss2.1

Indiana University. “Overview of the cron daemon”. URL: http://www.uwsg.iu.edu/usail/automation/cron.html