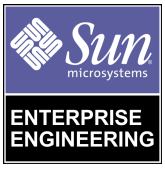
The Solaris™ Fingerprint Database - A Security Tool for Solaris Operating Enironment Files

By Vasanthan Dasan - Support Services Strategy Group, Alex Noordergraaf - Enterprise Engineering, and Lou Ordorica - Global eServices Engineering

Sun BluePrints™ OnLine - May 2001



http://www.sun.com/blueprints

Sun Microsystems, Inc.
901 San Antonio Road
Palo Alto, CA 94303 USA
650 960-1300 fax 650 969-9131

Part No.: 816-1148-10
Revision 01, June 2001

The Solaris™ Fingerprint Database - A Security Tool for Solaris Operating Enironment Files

Overview

Verifying whether system executables, configuration files, and startup scripts have been modified by a user has always been a difficult task. Security tools attempting to address this issue have been around for many years. These tools typically generate cryptographic checksums of files when a system is first installed.

The Solaris Fingerprint Database (sfpDB) is a free SunSolve Online^{SM} service that enables users to verify the integrity of files distributed with the Solaris™ Operating Environment (Solaris OE). Examples of these files include the `/bin/su` executable file, Solaris OE patches, and unbundled products such as Sun Forte™ Developer Tools. The list of checksums, generated for a system, must be updated after the system is modified by patch installation and software installations. The issue with these tools has always been verifying that the files used to generate the baseline checksums are correct and current.

The Solaris Fingerprint Database addresses the issue of validating the base Sun provided files. This includes files distributed with Solaris OE media kits, unbundled software, and patches. The sfpDB provides a mechanism to verify that a true file in an official binary distribution is being used, and not an altered version that compromises system security and causes other problems.

How Does the sfpDB Work?

Through the use of the MD5 hash algorithm, the Solaris Fingerprint Database compares a crytographically secure digital fingerprint with the trusted entry stored online in the sfpDB, and instantly identifies mismatches. The trusted entry is available over the internet at SunSolve OnLine website.

The MD5 digital fingerprint, or hash, generated locally will be used to determine if a file has been modified. It is virtually impossible to modify a file and retain the original MD5 digital fingerprint. The algorithm used by the `cksum(1)` Solaris OE command is much easier to circumvent and re-create the hash of the unmodified file, which is why the MD5 algorithm is used instead.

The sfpDB maps a digital fingerprint to a path name, package version/identifier, and product name. This is a one to many mapping, as some files may occur in several product and patch releases.

Note – Internet connectivity does not have to be available from all systems to use the sfpDB, as the files containing the MD5 hashes can be moved to an internet connected machine and verified either manually, or through the sfpDB Companion described at the end of this article.

---

sfpDB Scope

The goal of the sfpDB is to provide a comprehensive collection of digital fingerprints for Solaris OE software. To this end, the sfpDB is updated daily, and presently contains close to 1 million digital fingerprints for files used in the Solaris OE, Solaris OE patches, and unbundled products.

Limitations

Currently, foreign language versions of the Solaris OE and many encryption products are not supported. To suggest a product be added to sfpDB, please send email to: `fingerprints@sun.com`.

Download and Installation

This section describes how to download and install the MD5 software used to create MD5 digital fingerprints for use with the sfpDB.

To Install the MD5 Program (SPARC™ and Intel Architecture):

1. Download the MD5 binaries from:

`http://sunsolve.Sun.COM/md5/md5.tar.Z`

The MD5 programs are distributed in compressed tar file format.

2. Save the file to a directory (for example: `/usr/local` or `/opt`).

3. Unpack the archive:

```
# zcat md5.tar.Z | tar xvf -
```

The archive contents are extracted into a newly created directory called `md5`. The programs for SPARC and Intel Architecture hardware platforms are placed in this directory.

4. The file permissions on the extracted files must be modified before they can be executed. The following command will permit only root to read, write, and execute the `md5` programs:

```
# chmod 700 /opt/md5/*
# ls -l
total 94
-rwx------   1 21782    320        23892 Apr  5  2000 md5-sparc
-rwx------   1 21782    320         23452 Apr  5  2000 md5-x86
```

5. The owner and group of the extracted files must also be modified to correspond to a system defined user and group ID. Due to the sensitivity of the operations being performed by the `md5` programs, they should be owned by the `root` user and the `root` group. The following demonstrates performing this on the `md5` programs:

```
# chown root:root /opt/md5/*
# ls -l
total 94
-rwx------   1 root     root       23892 Apr  5  2000 md5-sparc
-rwx------   1 root     root       23452 Apr  5  2000 md5-x86
```

Note – The Solaris Fingerprint Database can be used to verify the integrity of the executables included in the package itself.

Creating an MD5 Digital Fingerprint

The following is an example of how to use the `md5` program to create an MD5 digital fingerprint:

```
# /opt/md5/md5-sparc /usr/bin/su
MD5 (/usr/bin/su) = cb2b71c32f4eb00469cbe4fd529e690c
```

The `md5` program can also be used to create multiple MD5 digital fingerprints, as shown in this example:

```
# /opt/md5/md5-sparc /usr/bin/su /usr/bin/ls
MD5 (/usr/bin/su) = cb2b71c32f4eb00469cbe4fd529e690c
MD5 (/usr/bin/ls) = 351f5eab0baa6eddae391f84d0a6c192
```

Note – The two previous examples were performed on a freshly installed Solaris 8 OE Update 3 system. No patches were installed. Do not rely on the output generated above as correct for every system. Use the following procedure, described in the *Testing an MD5 Digital Fingerprint* section.

Use the `md5` program with the `find(1)` command to create MD5 digital fingerprints for files that have changed recently. The next example creates MD5 digital fingerprints for files stored in the `/usr/bin` directory modified in the last day:

```
# find /usr/bin -type f -mtime -1 -print \
| xargs -n100 /opt/md5/md5-sparc > /tmp/md5s.txt
```

The results contained in the `/tmp/md5s.txt` file can be easily reviewed and copied into the Solaris Fingerprint Database web form.

Note – A maximum of 256 entires can be submitted into the web form at one time.

The next example shows how to create MD5 digital fingerprints for all the files stored in the `/usr/bin` directory:

```
# find /usr/bin -type f -print \
| xargs -n100 /opt/md5/md5-sparc > /tmp/md5s.txt
```

Testing an MD5 Digital Fingerprint

To check the digital fingerprint against the trusted entry stored in the sfpDB:

1. Visit the Solaris Fingerprint Database page at:

`http://sunsolve.Sun.COM/pub-cgi/fileFingerprints.pl`

The Solaris Fingerprint web form is displayed.

2. Copy and paste one or more MD5 digital fingerprints into the web form. For example, to verify the output of the md5 checksum of `su` generated in the previous case, the following would be pasted into the web form:

```
MD5 (/usr/bin/su) = cb2b71c32f4eb00469cbe4fd529e690c
```

3. Press `submit` to view the results. The following is an example of the results that are returned:

```
Results of Last Search

cb2b71c32f4eb00469cbe4fd529e690c - (/usr/bin/su) - 1 match(es)
                              canonical-path: /usr/bin/su
                              package: SUNWcsu
                              version:11.8.0,REV=2000.01.08.18.12
                              architecture: sparc
                              source: Solaris 8/SPARC
                              patch: 109005-01
```

Real World Results

The sfpDB provides an excellent mechanism to determine whether system binaries have been replaced by trojaned, or malicously modified, executables. To demonstrate the sfpDB performance when encountering actual trojaned Solaris OE binaries, the following experiment was performed. A freshly installed Solaris 8 OE Update 3 (1/01) system, used in the previous examples, had a Solaris OE rootkit installed. The term ‘rootkit’ is used to describe a set of scripts and executables packaged together which will allow the user to gain root access to a system. The sfpDB was then used to verify that trojaned executables were installed.

The rootkit described in the next few examples was used in a successful attack on one of the HoneyNet projects systems. For more information on this project, refer to the Bibliography.

Note – Many Solaris OE rootkits are available on the internet; most search engines will find several in responding to a simple query for ‘Solaris’ and ‘rootkit.’

This rootkit, called `sun2.rootkit`, replaced several system files, namely:

```
/bin/ls
/usr/bin/ls
/bin/ps
/bin/netstat
/usr/bin/netstat
```

The installation script, `setup.sh`, included in the rootkit, performed the following tasks. First, the rootkit backed up some, but not all, of the files it was going to replace by:

```
cp /bin/ls ./ls-back
cp /bin/ps ./ps-back
cp /bin/netstat ./netstat-back
```

Then, the rootkit installed its own version of these files with the following commands:

```
cp ls /bin/ls
cp ls /usr/bin/ls
cp ps /bin/ps
cp netstat /bin/netstat
cp netstat /usr/bin/netstat
```

The trojaned executables were run through `md5-sparc` with the following command:

```
# /opt/md5/md5-sparc /bin/ls /usr/bin/ls /bin/ps \
/bin/netstat /usr/bin/netstat
```

Which generated the following:

```
MD5 (/bin/ls) = da2ac2fc4645ff9fb737025f2d184aeb
MD5 (/usr/bin/ls) = da2ac2fc4645ff9fb737025f2d184aeb
MD5 (/bin/ps) = abd478c6597b4df1d565b9568f9e91bf
MD5 (/bin/netstat) = 2f4ec308b282c5c362e9fbd052b961f6
MD5 (/usr/bin/netstat) = 2f4ec308b282c5c362e9fbd052b961f6
```

When run through the web interface of the sfpDB, the following output was produced:

```
Results of Last Search

da2ac2fc4645ff9fb737025f2d184aeb - (/bin/ls) - 0 match(es)
                                 Not found in this database.
da2ac2fc4645ff9fb737025f2d184aeb - (/usr/bin/ls) - 0 match(es)
                                 Not found in this database.
abd478c6597b4df1d565b9568f9e91bf - (/bin/ps) - 0 match(es)
                                 Not found in this database.
2f4ec308b282c5c362e9fbd052b961f6 - (/bin/netstat) - 0 match(es)
                                 Not found in this database.
2f4ec308b282c5c362e9fbd052b961f6 - (/usr/bin/netstat) - 0 match(es)
                                 Not found in this database.
```

The Solaris Fingerprint Database correctly identified the trojaned executables as not being part of a Solaris OE distribution.

Additional sfpDB Tools

Recently, several tools making the sfpDB easier to use have been released by Glenn Brunette and Brad Powell. They are called sfpDB Companion and Sidekick. Both of these tools may be downloaded from the Sun BluePrints OnLine Tools site at:

`http://www.sun.com/blueprints/tools`

Solaris FingerPrint Database Companion (sfpC)

The Solaris FingerPrint Database Companion (sfpC) automates the process of collecting and checking MD5 signatures against the sfpDB. The sfpC simplifies this process by accepting as input a file containing a list of MD5 hashes, breaking that list into manageable chunks, and sending it to the sfpDB for processing. Results are then parsed from the returned HTML output. The sfpC makes it much easier to check large lists of Solaris OE files by automating the submission of the files to the sfpDB web site.

Solaris Fingerprint Database Sidekick (sfpS)

The sfpDB `sidekick.sh` is a script that works in conjunction with sfpDB and spfC by simplifying the process of checking a system for rootkits. Sidekick does this by maintaining a list of commonly trojaned Solaris OE executables, such as `/usr/bin/passwd` and `/usr/bin/login`.

sfpDB Frequently Asked Questions

Why do some of the returned entries contain odd path names?

In the process of gathering fingerprint data for the entries, it was discovered that many packages are not properly structured. Some path names may not be decided until installation. For these path names, it is not possible to derive the file name as found installed on the system; some path names are wrong, and some will contain `$SOMEVAR` values to be expanded during installation.

In any case, if a file was positively identified, it was shipped on a CD by Sun. The pathname does not need to match.

Will Sun publish the full content of the database?

Sun is currently studying how best to publish the full content of the database, as for some applications, the web interface to a CGI program is too limiting.

Conclusion

The Solaris Fingerprint Database is a tool for the verification of system files through cryptographic checksums. By being able to verify the integrity of system files, administrators can determine if system binaries have been modified by malicious users or trojaned.This tool provides these capabilities for customers to use when verifying the integrity of their systems.

It should also be noted that in the first six months of sfpDB availability from SunSolve service, several customers reported finding unexpected rootkits through its use.

Bibliography

HoneyNet project - `http://www.honeynet.org`

HoneyNet Project - *Know Your Enemy: Motives* - `http://project.honeynet.org/papers/motives/`

The Solaris Fingerprint Database - An Identification Tool for Solaris Software and Files `http://sunsolve.Sun.COM/pub-cgi/show.pl?target=content/content7`

Author's Biography: Vasanthan Dasan

Vasanthan Dasan is an ES Principal Engineer, one of 5 high-ranked engineers in Sun's Enterprise Services. Vasanthan joined Sun Microsystems in 1992 and is currently a Technology Strategist in the Support Services Global Strategy Business Development group. He is responsible for architecting application availability servies and for providing technical expertise on merger and acquisition activities.

Vasanthan was the Chief Architect for Support Services Engineering, responsible for developing online support servies for Sun's customer support engineers and external customers. Prior to that, he worked on Solaris products such as CacheFS, AutoClient, Solstice PC Products, and Jumpstart as part of the Solaris engineering team. Vasanthan co-authored Hands-On Intranet, published by Prentice Hall, and has written numerous Sun whitepapers. He was largely responsible for Sun's early adoption of the Web in 1994, and holds one of the industry's first Web patents, awarded for the invention of web-based personal newspapers.

Author's Biography: Alex Noordergraaf

Alex Noordergraaf has more than 9 years experience in the area of Computer and Network Security. As a Senior Staff Engineer in the Enterprise Engineering group of Sun Microsystems, he is developing, documenting, and publishing security best practices through the Sun BluePrints™ OnLine program. Articles completed include recommendations on: Solaris Security settings, Solaris Minimization, and Solaris Network settings.

Prior to his role in Enterprise Engineering he was a Senior Security Architect with Sun Professional Services, where he worked with many Fortune 500 companies on projects that included security assessments, architecture development, architectural reviews, and policy/procedure review and development. In addition to providing billable services to customers, he developed and delivered an Enterprise security assessment methodology and training curriculum to be used worldwide by SunPS. His customers have included major telecommunication firms, financial institutions, ISPs, and ASPs.

Author's Biography: Lou Ordorica

Lou Ordorica worked for several years as a system administrator at Sun Microsystems. He went on to teach and write about system administration for Sun's employees and customers, and is currently providing online support to customers using the Web.