Global Information Assurance Certification Paper

Practical Guide to Using SUDO

SANS GCUX, Version 2.0
Option 3

Robert W. Powell
May 1, 2004

TABLE OF CONTENTS

Page

EXECUTIVE SUMMARY

The purpose of this paper is to provide a security evaluation of sudo (Superuser Do), version 1.6.7p5, compiled to run on Sun Solaris 9 for use in a multi-user, networked environment. An installation guide will also be provided to assist those users not familiar with deploying sudo in a production environment. Where possible, recommendations will be given to avoid security pitfalls inherent to the use of this software. A section comparing sudo against Roles-Based Access Control (RBAC) will also be included.

The evaluation process consisted of both lab testing and an Internet search for known vulnerabilities. Lab testing revealed that the components of sudo function properly on systems that have the latest software version installed along with the latest vendor recommended patches. An Internet search produced no known software vulnerabilities for the version of sudo used in this testing effort (version 1.6.7p5). However, it should be noted that sudo's functionality poses a serious security risk to any system running the software in a poorly configured manner. It is for this reason that great care and consideration should be taken before deploying sudo in a production environment.

Sudo allows ordinary assigned users to invoke privileged commands as another user, such as root, by requiring each user to supply their password before the command is executed. In this sense, the System Administrator can delegate privileged tasks to users without having to give out the root password. From a security perspective, sudo can be configured to provide the following benefits:

- The ability to restrict what commands a user may run on a per-host basis
- Reduction in the number of users that require root's password
- Require users to enter their password before invoking privileged commands
- Logging of each command invoked by sudo

This product review was conducted using sudo 1.6.7p5 obtained from the Internet website, http://www.courtesan.com/sudo. Several key points should be considered before installing and using sudo. First, the configuration file, */etc/sudoers*, is installed as setuid root and could lead to a root compromise by programs such as vi that allow for shell escapes. Second, a poorly thought-out configuration file can lead to a root compromise through the use of un-intended command execution such as chmod'ing the */etc/passwd* file. Although sudo can be obtained from numerous websites, the software should be downloaded only from a trusted source such as http://www.courtesan.com/sudo. Failure to download software from a trusted source can result in system compromise through the execution of malicious code.

The primary recommendation is that sudo should be considered for use only in environments that have a justifiable functional requirement for allowing ordinary users to execute programs as root or some other user. Although sudo can prove to be a very useful tool from a system's administration perspective, the software poses a very real

threat to system compromise if it has not been configured in a well thought-out manner. For this reason, System Administrators should carefully plan each sudo configuration rule prior to deployment.

1. INTRODUCTION

System Administrators are often faced with the challenge of providing ordinary users with access to perform certain tasks as root or some other privileged user. In such a scenario, it is undesirable to provide the root password to every user needing privileged command access. An acceptable alternative would be to configure privileged command access through the use of sudo (Lukas).

Sudo is a popular freeware package that can be used to designate privileged command access to ordinary users . The software allows a System Administrator to configure users to run commands as either superuser (root) or any other user on the system. Sudo's flexible configuration file allows the System Administrator to strictly limit which users can invoke sudo and what commands authorized users can perform (Russell).

1.1 Files Installed

The sudo package is comprised of the primary configuration file /*etc/sudoers*; executable files for sudo and visudo; and man page files for sudo, visudo, and sudoers. Upon successful installation of the sudo package, the following files will be created with the noted permission sets*.

```
---s--x--x   1 root     root      /usr/local/bin/sudo
---x--x--x   1 root     root      /usr/local/sbin/visudo
-r--r--r--   1 root     root      /usr/local/man/man1m/sudo.1m
-r--r--r--   1 root     root      /usr/local/man/man1m/visudo.1m
-r--r--r--   1 root     root      /usr/local/man/man4/sudoers.4
-r--r-----   1 root     root      /etc/sudoers
```

* ***NOTE:*** *The sudo program itself is installed as a setuid binary and all users on the system have execute permissions for this file. Because the file is setuid and owned by root, your effective userid becomes root when using sudo. If you can get to a shell from sudo, you effectively become root on the system and will have superuser access.*

1.2 Configuration

The primary configuration file associated with sudo is */etc/sudoers* and should only be edited through the use of the **visudo** command. Editing the sudoers file with **visudo** serves two important functions. It locks the configuration file thereby preventing two users from editing the file at the same time and it provides for limited syntax checking. It is imperative that sudoers be free of syntax errors since sudo will not run with a syntactically incorrect configuration file. Should visudo detect a syntax error upon

exiting the sudoers file, a message will be displayed indicating the line number in which the error has been detected.

Although sudo can be configured with very complex and granular rule sets, the basic syntax is quite simple. A sample sudo configuration file has been included in *Appendix B, Example /Etc/Sudoers Configuration*. Each rule entry in the sudoers file consists of three parameters relating to the username, host, and command. The command entry must contain the full path for the command to avoid command spoofing. A fourth parameter, the userid, is optional and is set by default to the root user. The following example depicts a rule set containing the three basic parameters:

```
rpowell    bigchief=/usr/sbin/mount
```

In the example above, user **rpowell** has sudo privilege to execute the command **/usr/sbin/mount** on machine **bigchief**.

In the next example rule set, the optional userid parameter has been defined.

```
rpowell    (bigchief,littlechief)=(bsimpson,jdoe) /bin/kill
```

In the above example, user **rpowell** has been given sudo privilege to kill the processes associated with **bsimpson** and **jdoe** on machines **bigchief** and **littlechief**. This would be accomplished through the use of the command **sudo –u bsimpson** or **sudo –u jdoe**.

Sudo can be configured with a number of aliases to allow for grouping of commands, users, and hosts. Aliases are defined in the */etc/sudoers* file and follow the example outlined below.

```
# Host alias specification
Host_Alias     SUN=tango, bart, bigchief

# User alias specification
User_Alias     SA=jsmith, rjones

# Cmnd alias specification
Cmnd_Alias     USERS=/usr/sbin/useradd, /usr/sbin/userdel
Cmnd_Alias     MOUNT=/usr/sbin/mount -F hsfs -o \
               nosuid\,ro /dev/dsk/c1t0d0s2 /CD
Cmnd_Alias     UMOUNT=/usr/sbin/umount /CD
```

These aliases can then be incorporated into rule sets allowing for a wide range of configuration possibilities. The example below is based on the aliases previously defined.

```
# User privilege specification
SA   SUN=USERS, MOUNT, UMOUNT
```

Sudo can also be configured to specifically restrict certain commands and allow all others. This is accomplished through the use of an exclamation point preceding a given command. While this may seem desirable, there are serious security vulnerabilities inherent to this type of configuration. The best approach is to individually define which privileged commands a user has access to and limit these commands to the absolute minimum necessary*. By default, sudo will deny all other commands that are not specifically defined in the configuration file (Lucas).

***NOTE:** *Faulty rule sets can contain security vulnerabilities that are easily exploited by anyone who understands even the basics of how sudo works. With this in mind, careful planning and thorough consideration should be given to any rule set created.*

1.3 Default Settings

Sudo's source code contains a number of default configuration settings that pose a serious threat to security. These default settings can be overridden by compiling the software with the available configuration options. The following list of default configuration settings pose a security risk.

--with-umask=0022
: By default, the umask is 0022 when sudo is executed (Miller). Administrators should consider changing this setting to a more restrictive umask such as 0027.

--with-passwd-tries=3
: This setting represents the number of tries that a user can attempt to enter his/her password before sudo logs the failure and exits (Miller). By default, this is set to 3. Administrators should consider changing this setting to 1 on systems that require more stringent security settings.

--with-timeout=5
: This setting represents the number of minutes that can elapse before sudo prompts the user for his/her password a second time (Miller). By default, this is set to 5 minutes. This setting essentially enables caching of users' passwords for 5 minutes and allows users to continue executing sudo commands without having to re-authenticate after their first authentication. This setting should be changed to 0, thereby requiring users to always authenticate before sudo commands are executed.

--with-password-timeout=5
: This setting represents the number of minutes that will elapse before the sudo password prompt times out (Miller). By default, this is set to 5 minutes. This setting should be changed to 1 minute on systems that require more stringent security settings.

--enable-root-sudo

By default, sudo allows root to execute sudo commands (Miller). This is undesirable as users can chain sudo commands to get a root shell by executing something similar to **sudo sudo /bin/sh**. This setting should be changed to **--disable-root-sudo**.

To verify what configuration options were invoked during the compile process, execute the following command as root (note that the command is using a capital V):

```
# sudo -V
Sudo version 1.6.7p5

Authentication methods: 'passwd'
Syslog facility if syslog is being used for logging: local2
Syslog priority to use when user authenticates successfully:
notice
Syslog priority to use when user authenticates unsuccessfully:
alert
Send mail if the user is not in sudoers
Lecture user the first time they run sudo
Require users to authenticate by default
Allow some information gathering to give useful error messages
Set the LOGNAME and USER environment variables
Length at which to wrap log file lines (0 for no wrap): 80
Authentication timestamp timeout: 0 minutes
Password prompt timeout: 1 minutes
Number of tries to enter a password: 1
Umask to use or 0777 to use user's: 077
Path to mail program: /usr/lib/sendmail
Flags for mail program: -t
Address to send mail to: root
Subject line for mail messages: *** SECURITY information for %h
***
Incorrect password message: Sorry, try again.
Path to authentication timestamp dir: /var/run/sudo
Default password prompt: Password:
Default user to run commands as: root
Path to the editor for use by visudo: /usr/bin/vi
Environment variables to check for sanity:
     LANGUAGE
     LANG
     LC_*
Environment variables to remove:
     BASH_ENV
     ENV
     TERMCAP
     TERMPATH
     TERMINFO_DIRS
     TERMINFO
     _RLD*
     LD_*
     PATH_LOCALE
```

```
      NLSPATH
      HOSTALIASES
      RES_OPTIONS
      LOCALDOMAIN
      IFS
When to require a password for 'list' pseudocommand: any
When to require a password for 'verify' pseudocommand: all
Local IP address and netmask pairs:
```

1.4 Logging

One of the benefits to using sudo is the tracking and accountability feature that is built into the software (Wreski). By default, sudo logs messages via syslogd to LOCAL2. Each sudo message contains a timestamp, the user's name, the directory where sudo was run, and the command that was executed. The example below is an excerpt from the sudo log file.

```
Apr 15 09:24:06 : rpowell : TTY=pts/4 ; PWD=/home/rpowell ;
      USER=root ;COMMAND=/usr/bin/less /etc/shadow
```

1.5 As Compared to RBAC

Sudo is not the only program that can be used to provide ordinary users with access to privileged commands and utilties. Role-Based Access Control (RBAC) is an effective altenative to sudo that has been widely embraced by military and government organizations alike. The InterNational Committee for Information Technology Standards has approved RBAC as standard ANSI INCIST 359-2004. Sun Microsystems began shipping RBAC with Solaris 8 and it was derived from Sun's Trusted Solaris operating system.

RBAC is an access control program that associates individual users with an assigned role. Roles are associated with profiles that define the administrative tasks allowed. Users execute commands via RBAC by first logging in with their normal user ID and then authenticating to one of the defined roles. Users can only log into one role at a time.

Sudo and RBAC have many similarities. Sudo makes use **User_Alias** for assigned grouping of users, whereas RBAC implements roles for the logical grouping of users. RBAC allows users to be assigned either real or effective IDs, whereas sudo makes use of the **Runas_Alias** to assign UIDs and GIDs. The **Host_Alias** setting is used in sudo to allow for host-specific configurations. RBAC can provide for this same functionality by storing its databases on the local host or through the use of a name service to dispense the information (http://wwws.sun.com/software/whitepapers/wp-rbac/wp-rbac.pdf). Commands can be grouped together in sudo via the **Cmd_Alias** setting, which is similar to the RBAC implementation of rights profiles.

Sun's implementation of RBAC provides a GUI for configuration and allows for more granular controls to be implemented. However, RBAC does have a greater learning

curve than sudo and is available only on a limited number of operating systems. On the other hand, sudo can be configured to run on a wider range of operating systems and is free to the public (http://wwws.sun.com/software/whitepapers/wp-rbac/wp-rbac.pdf).

Because some organizations do not allow for the use of freeware products such as sudo, RBAC may be the only viable option for some administrators. Other organizations may prefer to use RBAC over sudo because RBAC is a vendor-supplied program with a direct support path. Additionally, sudo is a package that must first be compiled from source code before it can be installed on most systems and many organizations do not allow for programs with this requirement.

2. EVALUATION PROCEDURES

A thorough product evaluation was conducted based on the requirements below:

- Reveal any known product vulnerabilities extraneous to configuration settings
- Identify any potential security risks
- Test overall functionality of the software
- Provide recommendations that will enhance the overall security posture of the software

These requirements were satisfied through research and laboratory testing.

2.1 Research

An Internet search was conducted for any known product vulnerabilities. Below is a list of the sites referenced:

- http://www.courtesan.com/sudo/index.html
- http://www.cve.mitre.org
- http://www.cert.org/advisories
- http://icat.nist.gov/icat.cfm
- http://xforce.iss.net/

In addition to the aforementioned sites, the search engines Google and Yahoo were utilized to investigate vulnerabilities.

**** NOTE:*** *Sudo versions prior to 1.6.7p5 contain a number of well-known security vulnerabilities and therefore, should not be installed. Pre-existing installtions should be updated to version 1.6.7p5.*

2.2 Laboratory Testing

A laboratory environment was constructed to validate security risks that might be associated with the installation and use of sudo 1.6.7p5. The following guidelines were used to created the lab environment:

- Obtain source code for sudo 1.6.7p5 for Solaris from a trusted source: http://www.courtesan.com/sudo/
- Install operating system: Sun Solaris 9
- Compile sudo source code and install binary files
- Configure software and test its functionality

2.2.1 Lab Infrastructure

The version of software installed for testing was sudo 1.6.7p5 for Solaris.

Sudo can be installed from a pre-compiled package, available at such sites as http://www.sunfreeware.com, or by obtaining the source code from http://www.courtesan.com/sudo/ and compiling it on a machine that has a C compatible compiler installed. After the software has been compiled, the associated binary and text files can be transferred to the target machines.

Appendix C, Installed Files and Associated Permissions, contains the files and associated permissions that are loaded on a target machine after the sudo package has been successfully installed.

It is important to note that sudo runs as setuid root. Some organizations require such programs to be documented for auditing purposes and sudo should be listed in this documentation.

2.2.2 Test Plan

The test plan focused on configuration of the */etc/sudoers* file with a number of rule sets that could be used to test the software's functionality. After the initial configuration and testing of functionality, common exploits associated with poorly thought-out rule sets will be explored to demonstrate potential security threats.

The following configuration file was used to facilitate the functional and security tests associated with this evaluation:

```
# sudoers file.
#
# This file MUST be edited with the 'visudo' command as root.
#
# See the sudoers man page for the details on how to write a
# sudoers file.
#

# Host alias specification

# User alias specification
User_Alias     SA=jsmith, rjones

# Cmnd alias specification
```

```
Cmnd_Alias     USERS=/usr/sbin/useradd, /usr/sbin/userdel
Cmnd_Alias     MOUNT=/usr/sbin/mount -F hsfs -o nosuid\,ro
/dev/dsk/c1t0d0s2 \
          /CD
Cmnd_Alias     UMOUNT=/usr/sbin/umount /CD

# Defaults specification
Defaults  logfile=/var/log/sudolog

# User privilege specification
rpowell   ALL=(ALL) ALL
SA   ALL=/usr/bin/more /etc/passwd, /usr/sbin/poweroff
SA   littlechief=/usr/bin/more /var/log/authlog
SA   littlechief=USERS, MOUNT, UMOUNT

# Uncomment to allow people in group wheel to run all commands
# %wheel  ALL=(ALL) ALL

# Same thing without a password
# %wheel  ALL=(ALL) NOPASSWD: ALL

# Samples
# %users  ALL=/sbin/mount /cdrom,/sbin/umount /cdrom
# %users  localhost=/sbin/shutdown -h now
```

2.2.3 Functional Tests

The following functional tests were performed to ensure a reasonable level of comfort with the software's performance and functionality:

1. Verify access to privileged commands by authorized users
2. Verify denial of access to privileged commands by unauthorized users
3. Verify logging of sudo messages
4. Verify use of Command Aliases
5. Verify use of User Aliases
6. Verify syntactical checking associated with visudo

It is important to note that the functional testing described herein does not address every possible configuration option for sudo, but it does encompass a major portion of the product's capabilities.

2.2.4 Security Tests

The primary objective of this evaluation was to identify any negative impacts that installation or use of sudo might have on a machine's security posture. With that in mind, the following checks were performed:

1. Verify that password caching can be disabled and that users are required to enter a password prior to sudo command execution
2. Verify that UMASK can be set to 0027
3. Verify that password tries can be set to 1
4. Verify that the password prompt can be set to timeout after one minute

2.2.4.1 Password Caching

The default sudo configuration allows for password caching (Miller). Essentially, users are prompted for their password when the first sudo command is invoked. Subsequent uses, however, will not require users to enter their password for a period of time up to 5 minutes. This is very undesirable from a security perspective as a password should always be required when executing privileged commands. Prompting users for their password at the onset of command execution can prevent mistakes associated with hasty typing because the user is forced to stop momentarily and think about what they are doing. Requiring a password can also prevent malicious activity in situations where an authorized user steps away from the terminal without initiating a screen lock. This default setting can be overridden by compiling the software with the following option:

```
--with-timeout=0
```

2.2.4.2 UMASK Setting

The default sudo configuration umask setting allows a umask of 0022 (Miller). Such a umask can be undesirable as it allows for world read and execute access. Systems that require more stringent security settings should be configured with a default sudo umask of 0027 or perhaps even 0077. Sudo's default umask setting can be overridden by compiling the software with the following option:

```
--with-umask=0027
```

2.2.4.3 Password Tries

Formatted

The default sudo configuration logs failed password attempts after the 3rd consecutive failure (Miller). Systems that require more stringent security settings should be configured to log failed password attempts after the 1st failure. Sudo's default password tries setting can be overridden by compiling the software with the following option:

```
--with-passwd-tries=1
```

2.2.4.4 Password Prompt

Formatted

The default sudo configuration will display the password prompt for 5 minutes (Miller). Systems that require more stringent security settings should be configured to display the password prompt for only 1 minute. The default password prompt setting can be overridden by compiling the sudo with the following option:

```
--with-password-timeout=1
```

2.3 Test Results

Sudo version 1.6.7p5 passed all 6 functional tests.

Functional Test 1:
Authorized users were able to execute privileged commands via sudo.

Functional Test 2:
Unauthorized users were not able to execute privileged commands via sudo per the following example:

```
bash-2.05$ id
uid=1007(bsampson) gid=10(staff)
bash-2.05$ sudo more /etc/passwd
Password:
bsampson is not in the sudoers file.  This incident will be
reported.
bash-2.05$
```

Sudo also logs this failed attempt to the */var/log/sudolog* file per the example below:

```
Apr  21 21:00:21 : bsampson : user NOT in sudoers ;
    TTY=pts/4 ;PWD=/export/home/bsampson ; USER=root ;
    COMMAND=/usr/bin/more /etc/passwd
```

Functional Test 3:
Sudo messages were logged to the default log file defined in the */etc/sudoers* file provided in section *2.2.2 Test Plan*. The excerpt below was taken from */var/log/sudolog*:

```
Apr  21 14:00:48 : jsmith : TTY=pts/5 ; PWD=/home/jsmith ;
    USER=root ;COMMAND=/usr/bin/more /etc/passwd
```

Functional Test 4:
Authorized users were able to execute commands defined in the **Cmnd_Alias** setting.

Functional Test 5:
Commands associated with the **User_Alias** setting could be executed by authorized users.

Functional Test 6:
Syntactical checking worked as expected per the example below:

```
# visudo

User_Alias     SA=jsmith, rjones   -- line 22
User_Alas SA=jdoe, tdavis          -- line 23

>>> sudoers file: syntax error, line 23 <<<
What now?
```

As indicated above, visudo has detected a syntax error on line 23 because the word 'Alias' has been misspelled as 'Alas'. At this point, you can type 'e' to return to the line in question and search for the error, 'x' to quit and revert back to the previously saved file, or 'Q' to force visudo to accept the changes, syntax errors and all. If 'Q' is selected, you are essentially breaking sudo as it will not run with the syntactical errors in place (Lucas).

Sudo version 1.6.7p5 passed all 4 security tests.

Password Caching:
After sudo had been compiled to disable password caching using the option **--with-timeout=0**, users were prompted for their password each time they invoked a command using sudo.

UMASK Setting:
After sudo had been compiled with a umask of 0077 using the option **--with-umask=0077**, the effective umask became 0077 when users invoked a command using sudo. The screen output below verifies the sudo's umask setting of 0077:

```
$ id
uid=1003(rpowell) gid=14(sysadmin)
$ umask
0022
$ sudo view /etc/passwd
Password:
~
~
rtien:HsEdOGGuyr7Nw:12529::::::
jsmith:4jzuFBzqxqbUA:12541::::::
rjones:X6rOkm0/5cnbw:12541::::::
bsampson:mfkj.3l3FrvvQ:12541::::::
~
~
```

```
:! umask
0077
```

As noted on the last line of the above output, the umask is 0077.

Password Tries:
After sudo had been compiled to log failed password attempts after the first try using the option **--with-passwd-tries=1**, the expected logging took place per the example below:

Failed Password Attempt:
```
$ sudo more /etc/shadow
Password:
Sorry, try again.
sudo: 1 incorrect password attempt
```

Associated log file entry:
```
Apr  21 21:56:16 : rpowell : 1 incorrect password attempt ;
      TTY=pts/4 ;PWD=/home/rpowell ; USER=root ;
      COMMAND=/usr/bin/more /etc/shadow
```

Password Prompt:
After sudo had been compiled to display the password prompt for only 1 minute using the option **--with-password-timeout=1**, the password prompt was disabled after 1 minute of idle time. The example below demonstrates the expected behavior after 1 minute of idle time:

```
$ date
Wed Apr  21 22:41:05 EDT 2004
$ sudo more /etc/passwd
Password:
$
$ date
Wed Apr  21 22:42:08 EDT 2004
$
```

As mentioned previously, careful planning and thorough consideration should be given to any rule set created. With this in mind, the following examples depict how easily a root shell can be obtained as a result of poorly thought out rule sets.

Rule set:
```
rpowell   bigchief=/usr/bin/view /etc/shadow
```

Exploit:
```
$ id
uid=1003(rpowell) gid=14(sysadmin)

$ sudo view /etc/shadow
```

```
We trust you have received the usual lecture from the local
System Administrator. It usually boils down to these two things:

        #1) Respect the privacy of others.
        #2) Think before you type.

Password: <user enters password here>
----
~
~
noaccess:NP:6445::::::
nobody4:NP:6445::::::
rpowell:i.85JSwx.Dvxg:12462:1:::::
~
~
:!sh
----
# id
uid=0(root) gid=1(other)
```

In the above example, the user was able to invoke a simple shell escape command to get a root shell. Accordingly, sudo rule sets should not be created for access to applications that allow for shell escapes. Applications such as *view, vi, less,* and *pico* should be avoided.

Rule set:

```
rpowell   bigchief=/usr/bin/chmod
```

Exploit:

```
$ id
uid=1003(rpowell) gid=14(sysadmin)

$ sudo chmod 777 /etc/passwd

We trust you have received the usual lecture from the local
System Administrator. It usually boils down to these two things:

        #1) Respect the privacy of others.
        #2) Think before you type.

Password: <user enters password here>
```

In the above example, the user has changed the permissions of */etc/passwd* to be world writable, world readable, and world executable. At this point, the user can invoke an editor and change their user id to root's id of 0 and root's group of 1. This will effectively give the user a root shell along with superuser permissions. This type of exploit can be avoided by strictly limiting the chmod command to files that do not pose a security risk per the example below.

```
rpowell   bigchief=/usr/bin/chmod  /home/rpowell/report.txt
```

3. CONFIGURATION REQUIREMENTS

In order for sudo to function properly, the software must run as setuid root. Additionally, the two binary files that must be installed are */usr/local/bin/sudo* and */usr/local/sbin/visudo.* The configuration file that must be installed is */etc/sudoers*. Please reference *Appendix C, Installed Files and Associated Permissions,* for file permission settings.

3.1 Installation Guide

The preferred method of installing sudo is to obtain the source code and compile it with a C compatible compiler*. This approach allows for security enhancing configuration options to be compiled into the software. After the software has been compiled, the resulting files can be transferred to any number of machines.

***NOTE:** *Before sudo can be compiled, the system must be loaded with a C compatible compiler such as GCC.*

Obtain the latest version of sudo from http://www.courtesan.com/sudo/ and place it in a directory such as */opt/source*.

Next, unzip the gzip file using the following command:

```
# gunzip sudo-1.6.7p5.tar.gz
```

Then, use the following command to untar the file:

```
# tar –xvf sudo-1.6.7p5.tar
```

Now, cd to the directory that has been created using the following command:

```
# cd sudo-1.6.7p5
```

At this point, sudo is ready to be compiled. The following command will compile sudo with the options necessary to improve the program's overall security posture:

```
# ./configure --with-umask=0077 --with-passwd-tries=1
--with-timeout=0 --with-password-timeout=1 --disable-root-sudo
```

Next, execute the make command:

```
# make
```

Finally, execute the make install command.

```
# make install
```

The following files are created after the software has been compiled.

```
---s--x--x   1 root     root      /usr/local/bin/sudo
---x--x--x   1 root     root      /usr/local/sbin/visudo
-r--r--r--   1 root     root      /usr/local/man/man1m/sudo.1m
-r--r--r--   1 root     root      /usr/local/man/man1m/visudo.1m
-r--r--r--   1 root     root      /usr/local/man/man4/sudoers.4
-r--r-----   1 root     root      /etc/sudoers
```

The security policy of some organizations may not allow for compilers to be installed on production machines. If this is the case, sudo can be compiled on a non-production, standalone machine and the resulting files can be transferred to the target production system.

To create a tar file that can be used to install sudo on production machines, execute the following command after sudo has been compiled:

```
# tar –cvf SUDO.tar /usr/local/bin/sudo /usr/local/sbin/visudo
/usr/local/man/man1m/sudo.1m /usr/local/man/man1m/visudo.1m
/usr/local/man/man4/sudoers.4 /etc/sudoers
```

After the tar file has been copied to a target machine, use the following command to install the sudo files.

```
# tar –xvf SUDO.tar
```

Before logging can take place, an entry must exist in */etc/syslog.conf* similar to the example below.

```
local2.*       /var/log/sudo.log
```

The next step is to create the actual log file per the example below.

```
# touch /var/log/sudo.log
```

Finally, the syslogd process must be restarted before sudo can start logging messages to */var/log/sudo.log*. Use the following SIGHUP command to restart syslogd (Russell).

```
# kill –SIGHUP <syslogd_PID>
```

Alternatively, the */etc/sudoers* file can be configured with a **Defaults** entry to specify where logging is to take place. Be sure to use the **visudo** command if you wish to add the following line to the sudoers file.

```
Defaults  logfile=/var/log/sudo.log
```

Additional information on available configuration options can be obtained from http://www.courtesan.com/sudo/install.html

**** NOTE:*** *Some operating systems, such as Red Hat Linux and Suse Linux, will install sudo as part of the initial operating system install.*

4. RECOMMENDATIONS

As previously mentioned and demonstrated, security vulnerabilities within a poorly thought out sudo configuration can be easily exploited. With this in mind, careful planning and thorough consideration should be given to any rule set created. The primary recommendation is that sudo should be considered for use only in environments that have a justifiable functional requirement for the software's feature set.

Sudo must be edited using the **visudo** command as it locks the file ensuring that only one user at a time can make changes and it also allows for limited syntax checking.

Sudo should be compiled with the following configuration options via the configure command:

Deleted: .

```
# configure --with-umask=0077 --with-passwd-tries=1 --with-
timeout=0 --with-password-timeout=1 --disable-root-sudo
```

The following line should be removed from the default */etc/sudoers* file:

```
root      ALL=(ALL) ALL
```

Sudo rule sets should not be created for access to applications that allow for shell escapes. Users should not be granted access to applications such as **view**, **vi**, **less**, and **pico** as they allow for shell escapes.

Sudo rule sets should not be created to allow ordinary users access to chmod system files.

Sudo rule sets should be configured using the full pathname of commands to avoid command spoofing.

Lastly, rule sets should be configured from the standpoint of least access necessary. In other words, users should be granted access to the absolute minimum amount of privileged commands necessary to perform his or her job function. By default, sudo will deny access to all privileged commands that are not specifically defined in the configuration file.

The preferred method of installing sudo is to obtain the source code and compile the binaries on a standalone machine. This approach allows for security enhancing

configuration options to be compiled into the software. After the software has been compiled, the resulting files can be transferred to the target production machine. *Appendix D, Installation Guide*, contains an installation guide to assist with installing sudo from the source code.

APPENDIX A. SUDO README FILE[1]

```
This is Sudo version 1.6.7

The sudo philosophy
===================
Sudo is a program designed to allow a sysadmin to give limited
root privileges to users and log root activity.  The basic
philosophy is to give as few privileges as possible but still
allow people to get their work done.

Where to find sudo
==================
Before you try and build sudo, *please* make sure you have the
current version.  The latest sudo may always be gotten via
anonymous ftp from ftp.sudo.ws in the directory /pub/sudo/.

The distribution is sudo-M.m.tar.gz where `M' is the major
version number and `m' is the minor version number.

BETA versions of sudo may also be available.  If you join the
`sudo-workers' mailing list you will get the BETA announcements
(see the `Mailing lists' section below).

What's new
==========
For a history of sudo please see the HISTORY file that came with
this release.

For a complete list of changes, see the CHANGES file.  For a
summary, see the web page, http://www.sudo.ws/sudo/.

If you are upgrading from an earlier version of Sudo, please see
the UPGRADE file.

NOTE:  Starting with sudo 1.5.7 the configuration method has
       changed significantly as compared to previous versions.
       All options are now set via the configure script.  See the
       `INSTALL' file for a list of all the configure options.

System requirements
===================
To build sudo from the source distribution you need a machine
running UN*X (most flavors of BSD, SYSV, or POSIX will do), a
working C compiler, and the make utility.

If you wish to modify the parser, then you will need flex
version 2.5.2 or later and either bison or byacc (sudo comes
with a pre-flex'd tokenizer and pre-yacc'd grammar parser).
```

[1] Obtained from http://www.courtesan.com/sudo/man/sudoers.html

You'll also have to uncomment a few lines from the Makefile or run configure with the --with-devel option. You can get flex via anonymous ftp from ftp://ftp.ee.lbl.gov/pub/flex* as well as any GNU mirror. You can get GNU bison from ftp://ftp.gnu.org/pub/gnu/bison/ or any GNU mirror.

Building the release
====================
Please read the installation guide in the `INSTALL' file before trying to build sudo.  The `RUNSON' file contains a list of platforms that this version of sudo is known to work on. If you can add to this list, please send mail to sudo@sudo.ws. If something goes wrong you may want to refer to the `TROUBLESHOOTING' file.

Copyright
=========
Sudo is distributed under a BSD-style license. Please refer to the `LICENSE' file included with the release for details.

Mailing lists
=============
sudo-announce This list receives announcements whenever a new version of sudo is released.
               http://www.sudo.ws/mailman/listinfo/sudo-announce

sudo-users   This list is for questions and general discussion about sudo.
               http://www.sudo.ws/mailman/listinfo/sudo-users

sudo-workers   This list is for people working on and porting sudo.
               http://www.sudo.ws/mailman/listinfo/sudo-workers

To subscribe to a list, visit its url (as listed above) and enter your email address to subscribe. Digest versions are available but these are fairly low traffic lists so the digest versions are not a significant win.

Mailing list archives are also available. See the mailing list web sites for the appropriate links.

Web page
========
There is a sudo `web page' at http://www.sudo.ws/sudo/ that contains an overview of sudo as well as pointers to BETA versions and other useful info.

Bug reports
===========
A list of known bugs may be found in the `BUGS' file. If you have found what you believe to be a bug, you can file a bug report with the sudo bug database, on at web at http://www.sudo.ws/bugs/.

```
Please read over the `TROUBLESHOOTING' file *before* submitting
a bug report.  When reporting bugs, please be sure to include
the version of sudo you are using as well as the platform you
are running it on.
```

APPENDIX B. EXAMPLE /ETC/SUDOERS CONFIGURATION FILE

```
# sudoers file.
#
# This file MUST be edited with the 'visudo' command as root.
#
# See the sudoers man page for the details on how to write a
sudoers file.
#

# Host alias specification

# User alias specification
User_Alias     SA=jsmith, rjones

# Cmnd alias specification
Cmnd_Alias     USERS=/usr/sbin/useradd, /usr/sbin/userdel
Cmnd_Alias     MOUNT=/usr/sbin/mount -F hsfs -o \
               nosuid\,ro /dev/dsk/c1t0d0s2 /CD
Cmnd_Alias     UMOUNT=/usr/sbin/umount /CD

# Defaults specification
Defaults  logfile=/var/log/sudolog

# User privilege specification
rpowell   ALL=(ALL) ALL
SA   ALL=/usr/bin/more /etc/shadow, /usr/sbin/poweroff
SA   littlechief=/usr/bin/more /var/log/authlog
SA   littlechief=USERS, MOUNT, UMOUNT

# Uncomment to allow people in group wheel to run all commands
# %wheel  ALL=(ALL) ALL

# Same thing without a password
# %wheel  ALL=(ALL) NOPASSWD: ALL

# Samples
# %users  ALL=/sbin/mount /cdrom,/sbin/umount /cdrom
# %users  localhost=/sbin/shutdown -h now
```

APPENDIX C. INSTALLED FILES AND ASSOCIATED PERMISSIONS

```
---s--x--x   1 root     root      /usr/local/bin/sudo
---x--x--x   1 root     root      /usr/local/sbin/visudo
-r--r--r--   1 root     root      /usr/local/man/man1m/sudo.1m
-r--r--r--   1 root     root      /usr/local/man/man1m/visudo.1m
-r--r--r--   1 root     root      /usr/local/man/man4/sudoers.4
-r--r-----   1 root     root      /etc/sudoers
```

APPENDIX D. INSTALLATION GUIDE

The preferred method of installing sudo is to obtain the source code and compile the software on a standalone machine. This approach allows for security enhancing configuration options to be compiled into the software. After the software has been compiled, the resulting files can be transferred to the target production machine.

Before sudo can be compiled, the system must be loaded with a C compatible compiler such as GCC. Sudo should be compiled with the following configuration options via the configure command.

```
# ./configure --with-umask=0077 --with-passwd-tries=1
--with-timeout=0 --with-password-timeout=1 --disable-root-sudo
```

Next, execute the make command.

```
# make
```

Finally, execute the make install command.

```
# make install
```

The following files are created after the software has been compiled.

```
---s--x--x   1 root     root      /usr/local/bin/sudo
---x--x--x   1 root     root      /usr/local/sbin/visudo
-r--r--r--   1 root     root      /usr/local/man/man1m/sudo.1m
-r--r--r--   1 root     root      /usr/local/man/man1m/visudo.1m
-r--r--r--   1 root     root      /usr/local/man/man4/sudoers.4
-r--r-----   1 root     root      /etc/sudoers
```

Next, create a tar file that can be used to install sudo on a production machine. The command below will create the necessary tar file.

```
# tar –cvf SUDO.tar /usr/local/bin/sudo /usr/local/sbin/visudo
/usr/local/man/man1m/sudo.1m /usr/local/man/man1m/visudo.1m
/usr/local/man/man4/sudoers.4 /etc/sudoers
```

After the tar file has been copied to a target machine, use the following command to install the sudo files.

```
# tar –xvf SUDO.tar
```

Before logging can take place, an entry must exist in */etc/syslog.conf* similar to the example below.

```
local2.*       /var/log/sudo.log
```

The next step is to create the actual log file per the example below.

```
# touch /var/log/sudo.log
```

Finally, the syslogd process must be restarted before sudo can start logging messages to */var/log/sudo.log*. Use the following SIGHUP command to restart syslogd.

```
# kill –SIGHUP <syslogd_PID>
```

Alternatively, the */etc/sudoers* file can be configured with a **Defaults** entry to specify where logging is to take place. Be sure to use the **visudo** command if you wish to add the following line to the sudoers file.

```
Defaults  logfile=/var/log/sudo.log
```

Additional information on available configuration options can be obtained from http://www.courtesan.com/sudo/install.html

**** NOTE:*** *Some operating systems, such as Red Hat Linux and Suse Linux, will install sudo as part of the initial operating system install.*

APPENDIX E. REFERENCES

"RBAC in the Solaris Operating Environment." Apr 2001.
URL: http://wwws.sun.com/software/whitepapers/wp-rbac/wp-rbac.pdf (29 Apr 2004).

"Role Based Access Contorl." 19 Feb 2004.
URL: http://csrc.nist.gov/rbac/ (29 Apr 2004).

"Using Sudo." URL: http://www.aplawrence.com/Basics/sudo.html (25 Mar 2004).

Galvin, Peter Baer. "Role-Based Access Control." Aug 2001.
URL: http://www.samag.com/documents/s=1147/sam0108p/ (29 Apr 2004).

Harrison, Peter. "Using Sudo." Chapter 12.
URL: http://www.siliconvalleyccie.com/linux-hn/sudo.htm (22 Mar 2004).

Lucas, Michael. "Eliminating Root with Sudo." 29 Aug 2002.
URL: http://www.onlamp.com/pub/a/bsd/2002/08/29/Big_Scary_Daemons.html
(09 Apr 2004).

Lucas, Michael. "Sudo Aliases and Exclusions." 12 Sep 2002.
URL: http://www.onlamp.com/pub/a/bsd/2002/09/12/Big_Scary_Daemons.html
(09 Apr 2004).

Miller, Todd. 08 May 2003. URL: http://www.courtesan.com/sudo/ (20 Mar 2004).

Oliver, Ross. "Using Solaris RBAC." Sys Admin Magazine Archives 2002.
URL: http://www.samag.com/documents/s=7667/sam0213c/0213c.htm (29 Apr 2004).

Russell, Kathy. "Managing Root Access with Sudo."
URL: http://unix.about.com/library/weekly/aa102500a.htm (13 Apr 2004).

Wreski, Dave. "Using Sudo." 07 Aug 2000.
URL: http://linuxsecurity.com/tips/tip-18.html (24 Mar 2004).