Logging sFTP activity for chrooted users

The goal here is to allow one or more new users to connect to the server using SFTP over SSH. Each user will have a unique username and password and once logged in will be restricted (chrooted) to their own directory (or you can let them access a shared directory).

All the following commands need either to be run by the root user or via sudo.

1. Creating sftp only users

The first step is to create a group `sftponly`:

```
# groupadd sftponly
```

and then create and assign the users to that group:

```
# GROUPID=$(getent group sftponly | cut -d: -f3)
# useradd someuser    -d /path/to/somedirectory    -g sftponly -M -N -o -u $GROUPID -s /bin/false
# useradd anotheruser -d /path/to/anotherdirectory -g sftponly -M -N -o -u $GROUPID -s /bin/false
```

We're assuming here that the directories already exist. If not, you should create them first:

```
# mkdir -m2755 /path/to/somedirectory
# mkdir -m2755 /path/to/anotherdirectory
```

These directories, and all upstream directories, need to be owned by `root:root` and not be writable by any other users. Otherwise you will see the error "bad ownership or modes for chroot directory" in auth.log.

You can check that the users and group have been created properly using:

```
# getent group sftponly
sftponly:x:1004:

# getent passwd someuser
someuser:x:1004:1004::/path/to/somedirectory:/bin/false
```

As you can see our new user or users have been created with both their `uid` and `gid` set to the sftponly group id. Their home directory is the location they will be able to connect to, and only via SFTP, because we've configured their shell as /bin/false which prevents SSH logins.

Finally, don't forget to assign passwords:

```
# passwd someuser
Enter new UNIX password: ********
Retype new UNIX password: ********
passwd: password updated successfully
```

2. SSHD configuration

Now we need to tell the SSH daemon how to recognise and handle our new sftp users by editing `/etc/ssh/sshd_config`.

Starting with the default configuration, we remove the line:

```
Subsystem sftp /usr/lib/openssh/sftp-server
```

replacing it with:

```
Subsystem sftp internal-sftp -l INFO
```

And at the end of the configuration file we append the following:

```
Match Group sftponly
  ChrootDirectory %h
  X11Forwarding no
  AllowTcpForwarding no
  ForceCommand internal-sftp -l INFO
```

Now any connections from users in the `sftponly` group will be handled by the `internal-sftp` in-process SFTP server with a log level of INFO. They will not be able to connect using SSH, and will not have access to files outside of their home (%h) directory.

To apply the new configuration you will need to:

```
# systemctl restart ssh.service
```

3. Blocked SSH gotcha

With the above `Match Group` setting any users who belong to the `sftponly` group will be restricted to sFTP and blocked from SSH - not just those with `sftponly` as their primary group.

So if you have SSH users who also belong to the `sftponly` group, in order to have write permission on uploaded files, you will need to exclude them from the Match criteria, for example:

```
Match Group sftponly User *,!adminuser
```

Otherwise the affected user/s will have no access to SSH, and will also not be able to connect via sFTP if their home director is not write-only to root.

Other options for `Match` are described in the man page:

> The arguments to **Match** are one or more criteria-pattern pairs or the single token **All** which matches all criteria. The available criteria are **User**, **Group**, **Host**, **LocalAddress**, **LocalPort**, and **Address**. -- *man 5 sshd_config*

4. AllowUsers gotcha

If your SSH configuration includes an AllowUsers command then you will need to replace it with AllowGroups because the two aren't compatible. So instead of:

```
AllowUsers adminuser
```

you will have:

```
AllowGroups staff sftponly
```

where `adminuser` is now a member of the `staff` group.

With these settings all members of `staff` as well as our new `sftponly` users can connect to the server. The former will enjoy full SSH access while the latter will be restricted to their own chrooted sftp environment.

To add a user to a group without having to edit the `/etc/group` and related files directly:

```
# usermod -a -G staff adminuser
# groups adminuser
adminuser : adminuser staff
# groups someuser
someuser : sftponly
```

5. Making a connection

At this stage you should already be able to connect and log in using sftp from the command line:

```
sftp someuser@hostname
sftp> help
Available commands:
bye                                Quit sftp
cd path                            Change remote directory to 'path'
...
sftp> bye
```

If not, check your `auth.log`. The problem is likely to do with missing directories or the wrong directory permissions.

6. But where are the logs?

A good question. By default you won't see any logs for sftp actions, only for the usual sshd "session opened" and "session closed" events in the `auth.log`.

The issue is that because each sftp user is chrooted into their own directory, they have no access to write to the usual system logs. The details are pretty complicated and hard to explain, so we'll trying and stick to simple instructions.

First, create a `/dev` directory in the home directory of each chrooted sftp user:

```
# mkdir -m2755 ~someuser/dev
# mkdir -m2755 ~anotheruser/dev
```

Now edit a new file `/etc/rsyslog.d/sftp.conf` to open a socket in each of the new directories, and pipe them to a new `sftp.log` log file:

```
# create additional sockets for the sftp chrooted users
module(load="imuxsock")
input(type="imuxsock" Socket="/path/to/somedirectory/dev/log" CreatePath="on")
input(type="imuxsock" Socket="/path/to/anotherdirectory/dev/log" CreatePath="on")

# log internal-sftp activity to sftp.log
if $programname == 'internal-sftp' then /var/log/sftp.log
& stop
```

Restart the rsyslog daemon, and we're done:

```
# systemctl reload-or-restart rsyslog
```

If you see a syslog message `'imuxsock' already in this config` then you can omit the first line (highlighted above) as `imuxsock` is already available.

Now checking the contents of the new /dev directory you should see that a file `log` has appeared:

```
# ls -l ~someuser/dev/
total 0
srw-rw-rw- 1 root root 0 Jul 20 17:57 log
```

This is actually not a file, but a socket feeding in to rsyslog. The users sFTP actions will not appear here, but in the new `/var/log/sftp.log` file:

```
==> sftp.log <==
Jul 20 18:13:09 hostname internal-sftp[26800]: session opened for local user someuser from [8.8.8.8]
Jul 20 18:13:15 hostname internal-sftp[26800]: opendir "/"
Jul 20 18:13:15 hostname internal-sftp[26800]: closedir "/"
Jul 20 18:13:25 hostname internal-sftp[26800]: opendir "/html"
Jul 20 18:13:26 hostname internal-sftp[26800]: closedir "/html"
Jul 20 18:14:20 hostname internal-sftp[26800]: session closed for local user someuser from [8.8.8.8]
```

And we're done. All that's left is to arrange for log rotation, and if there are already files in the sftp directories their permissions will need to be changed to grant read/write access to the sftp accounts, but those are trivial problems.

For non-Debian or older systems look under References below. And if you have any questions or comments there's a Feedback button below.

7. Log Rotation

The following configuration should work for `logrotate`:

/etc/logrotate.d/sftp

```
/var/log/sftp.log
{
        weekly
        missingok
        rotate 4
        compress
        delaycompress
        notifempty
        create 640 root adm
}
```

8. Showing incorrect timestamp

In the chroot enviroment sftp doesn't have acccess to the system time settings so the displayed timestamps can be off by the difference between your timezone and UTC.

If it bothers you, the simplest fix is to run the following as root:

```
mkdir ~someuser/etc
cp -af /etc/localtime ~someuser/etc/.
```

The copies the server timezone settings (binary) file making it accessible in the chroot'ed environment.

9. Related Articles - Log Files

- SQL Using a PostgreSQL foreign data wrapper to analyze log files
- SYSTEM Controlling what logs where with rsyslog.conf
- SYSTEM **Logging sFTP activity for chrooted users**
- SYSTEM Analyzing Apache Log Files
- SYSTEM Bash script to generate broken links report
- SYSTEM Referer Spam from Microsoft Bing

- SYSTEM Blocking Unwanted Spiders and Scrapers
- SYSTEM Fake Traffic from AVG
- SYSTEM Referer Spam from Live Search

10. References

- Implement a SFTP Service for Ubuntu/Debian With a Chroot'ed, Isolated File Directory [http://devtidbits.com/2011/06/29/implement-a-sftp-service-for-ubuntudebian-with-a-chrooted-isolated-file-directory/]
- OpenSSH/Cookbook/SFTP [https://en.wikibooks.org/wiki/OpenSSH/Cookbook/SFTP]
- OpenSSH/Logging [https://en.wikibooks.org/wiki/OpenSSH/Logging]
- How to log internal-sftp chroot jailed users [https://access.redhat.com/discussions/672633]
- RHEL / CnetOS 7 sftp logging in chroot [http://blog.acsystem.sk/linux/rhel-cn sftp-logging-in-chroot]

11. User Comments

Post your comment or question

Josef 19 January, 2021

Very interesting tutorial – thanks a lot.

Can you please explain the following command to me:
`mkdir -m2755 directory`

I assume that `755` is setting permission.
But what exactly is the `2` about?

Thanks

> If '755' gives you 'drwxr-xr-x', then '2775' gives you 'drwxr-sr-x', where the 's' represents a sticky bit [https://en.wikipedia.org/wiki/Sticky_bit]. 😎

Duncan Wilcox [https://sparkleapp.com] 13 August, 2019

I researched the 50 socket limit Charlie is referring to, and it looks like it's a documentation bug. This 2013 commit has removed the limit and made it "dynamic".

github.com/rsyslog/rsyslog/commit/af5a67eb23ce15a5c226ca4c [https://github.com/rsyslog/rsyslog/commit/af5a67eb23ce15a5c226ca4c] -
10fd774a2ce985b0

Charlie [https://www.rsyslog.com/doc/v8-stable/configuration/modules/imuxsock.html#caveats-known-bugs] 12 April, 2018

Rsyslog has a compile time limit of 50 sockets so this solution does not scale without recompiling the rsyslogd.

Since modern EDI environments can have tens of thousands of SFTP users, even if you recompile this solution still does not scale well.

Daithi 21 October, 2016

Hi,

In my company I am a first year analyst so a complete novice anyway but I am trying to configure sftp for logging.

By following your tutorial I can match the steps we are taking to achieve this.

We are using red-hat gui so its a little different.

Any without getting too technical, i have loosely followed the steps above and the log file was updating as expected.

However after restarting rsyslog and sshd services, the logging stopped.

Any idea why this happened?

Hubert Reeves 5 August, 2016

Hi,

Thanks for your answer.

Yeah I thought about this but it adds a lot of processing if I want to write a script that does transfer statistics (for bw usage per user).

After thinking a bit about it I found another solution:

I've finally created a chrooted dir for each user and "mount -o bind"' ed the exported directories in every user home except the /dev which stays unique for each user.

and added in rsyslog.conf:

```
#
# sftp trick
#
input(type="imuxsock" HostName="user1" Socket="/mnt/export/user1/dev/log" CreatePath="o
input(type="imuxsock" HostName="user2" Socket="/mnt/export/user2/dev/log" CreatePath="o
input(type="imuxsock" HostName="user3" Socket="/mnt/export/user3/dev/log" CreatePath="o
if $fromhost == 'user1' then /var/log/sftp-user1.log
& stop
if $fromhost == 'user2' then /var/log/sftp-user2.log
& stop
if $fromhost == 'user3' then /var/log/sftp-user3.log
& stop
```

Now I have a separate file for each user and the hostname is set as the user name 😃 so I can parse them fingers in the nose.

Thanks 😃

Hubert Reeves 5 August, 2016

Hi, excellent tutorial.

Just a little question:

What if I have multiple users jailed to the same chroot and I want to log the username on every line of the logs ?

Actually the hostname is logged but not the user.

Thx

> A good question.
>
> The username should already be logged at the start and end of the session, and the process id consistent throughout, so some post-processing into a new file might be an option.
>
> ```
> internal-sftp[17902]: session opened for local user sftp-user from [1.2.3.4]
> ...
> internal-sftp[17902]: session closed for local user sftp-user from [1.2.3.4]
> ```