Debian Package Installation

Install *nftfw* from the Debian Package

nftfw can be installed from a Debian binary package, there is a zip file called *nftfw_current.zip* in the package directory containing the most recent version. For safety, *nftfw* needs some configuration after installation. See the installation document Install *nftfw* from Debian package for a how-to guide.

Following Debian practice, the system will be installed in the root of the file system, so the control files will be in */etc/nftfw* with the library files in */var/lib/nftfw*.

Getting started

This section presents the bare bones of installing the *nftfw* package on a vanilla system. To cope with some special circumstances, links in the document jump to sets of instructions which start after the main installation documentation.

Iptables check

First check that you can upgrade your system to run *nftables*:

```
$ sudo iptables -V
iptables v1.8.2 (nf_tables)
```

If the output is not as above, then you need to swap your *iptables* version. See Switching iptables below, then come back here when you've done that.

Download the package

Download the zipfile containing the most recent debian binary package from nftfw github site. This will download a file (*nftfw_current_deb.zip*) used to hide the version number and running *unzip* on the file will yield the package. The filename of the package contains a version number and ends in *.deb*, for example *nftfw_1.0.0-1_all.deb*.

What to do if you are running a manually installed *nftfw* version

See Manually installed nftfw below, and return here when done.

Install the package

```
$ sudo dpkg -i nftfw_XXXXX_all.deb
```

where XXXXX is the version number of the file you downloaded. *dpkg* doesn't install dependencies and may complain and stop. If this happens run:

```
$ sudo apt-get --fix-broken install
```

which will install the dependencies and then install *nftfw*.

The `dkpg -i` command can also be used to update a previously installed package to a new version.

When installing *nftfw*, you will be asked if you want to change the ownership of the */etc/nftfw* directory to allow configuration by a non-root user. When*nftfw* writes files under the directory it will take the ownership from the owner of */etc/nftfw*. Debian's *debconf* is used to remember this setting for later updates, and you can change ownership after installation using:

```
$ sudo dpkg-reconfigure nftfw
```

What is installed?

The package will install:

- the Python commands in */usr/bin*: *nftfw*, *nftfwls*, *nftfwedit* and *nftfwadm*.
- Control files in */etc/nftfw*, unless they exist. The*rule .d* directory will be updated. The firewall is populated to permit access to commonly used services.
- Basic directory structure in */usr/var/lib/nftfw*.
- Manual pages for the commands above, and section 5 manual pages for *nftfw_config* and *nftfw_files*.
- Documentation and examples in */usr/share/doc/nftfw*.
- A cron file in */etc/cron.d/nftfw*, this will need editing to make active.
- *systemd* path file to enable monitoring of the directories in */etc/nftfw*.

Many directories have *README* files explaining what is there and why.

Check *nftfw* is running

Check that it's running:

```
$ sudo nftfw -x -v load
nftfw[15264]: Loading data from /etc/nftfw
nftfw[15264]: Creating reference files in /var/lib/nftfw/test.d
nftfw[15264]: Test files using nft command
nftfw[15264]: Testing nft rulesets from nftfw_init.nft
nftfw[15264]: Determine required installation
nftfw[15264]: No install needed
```

The number in the log is the process id, so will be different for you.

On first installation

See Taking precautions if you have a live firewall if your system is running a live *iptables* or *nftables* firewall, and you want to keep that active until *nftfw* is live and configured.

If you are running *nftfw* on a Sympl or Symbiosis system then you might want to migrate your current firewall settings into *nftfw* - see Migrating a Sympl or Symbiosis firewall below. It's a good idea to do this now, before starting systems that run *nftfw* automatically.

Loading the rules

Load the rules into the kernel:

```
$ sudo nftfw -f -v load
```

nftfw will tell you what it's done.

Look at the nftables rules

```
$ sudo nft list ruleset ip | less
```

for ipv4 and

```
$ sudo nft list ruleset ip6 | less
```

for ipv6. Hint: this is a lot to type and you may want to use the commands again, so create and store shell aliases in your shell's .rc file for them.

```
alias nfl='sudo nft list ruleset ip|less'
alias nfl6='sudo nft list ruleset ip6|less'
```

In extremis, you can clear the rules with

```
$ sudo nft flush ruleset
```

Changing *config.ini*

The *nftables.conf* file is the input file for the *nftables* system and is what *nftfw* creates. For safety, the distributed version writes the file in */etc/nftfw/nftables.conf*. The file here can be deleted. You need to tell *nftfw* to write the file in the correct place - in */etc*.

Edit */etc/nftfw/config.ini* to correctly site the *nftables.conf* file:

```
#  Location of system nftables.conf
#  more comments...
#  Usually /etc/nftables.conf
nftables_conf = /etc/nftables.conf
```

run *nftfw* to write the file, and also to load the kernel's *nftables*:

```
$ sudo nftfw -f -v load
```

Start the *nftables* service

Check that *nftables.service* is running:

```
$ sudo systemctl status nftables
```

and if not:

```
$ sudo systemctl enable nftables
$ sudo systemctl start nftables
```

Changing */etc/cron.d/nftfw*

Edit the */etc/cron.d/nftfw* file to make the working lines active, removing the '#' from the start of the lines containing cron commands.

Start the active control directories

```
$ sudo systemctl enable nftfw.path
$ sudo systemctl start nftfw.path
```

making *nftfw* run when anything changes in the *incoming.d*, *outgoing.d*, *blacklist.d*, *whitelist.d* and *blacknets.d* directories in */etc*.

You are done

If you are new to *nftfw*, look at the How do I... document which has sections on how to add or remove firewall controls. It should get you going on how to configure the firewall. As distributed, *nftfw* allows access to most of the usual services supplied by a LAMP system.

You now have an active *nftfw* system and should look in */etc/nftfw* to configure the various control directories to your system needs.

More complex scenarios

This section contains extra command sequences and information, that are referenced above for special circumstances.

Switching iptables

Here is what to do if `iptables -V` says 'legacy' and not 'nf_tables':

```
$ sudo iptables-save > ipsaved
$ sudo ip6tables-save > ip6saved
$ sudo update-alternatives --config iptables
# select selection 0, /usr/sbin/iptables-nft, auto mode
$ sudo update-alternatives --config ip6tables
```

```
# select selection 0, /usr/sbin/iptables-nft, auto mode
```

Run the `sudo iptables -V` again, to check things have switched, and

```
$ sudo iptables-restore < ipsaved
$ sudo ip6tables-restore < ip6saved
$ sudo iptables-legacy -F
$ sudo ip6tables-legacy -F
```

The last two commands are very important to clear out the old tables.

Back to Install the package

Precautions for a live firewall

If have a running *nftables* or *iptables* firewall, then it's a good idea to save its rules in *nftfw*'s internal backup system so that the system will revert to your working firewall on a problem.

If you have a running firewall, save its rules first, and then load the *nftfw* rules:

```
$ sudo nftfwadm save
$ sudo nftfw -f -v load
```

Output should end with 'Install rules in ...' - wherever the *config.ini* file tells *nftfw* to store the *nftables.conf* file. The new rules will be installed in the kernel tables:

```
$ sudo nft list ruleset
```

will list the ruleset which will have been changed by *nftfw*.

If you have a problem, revert to old rules:

```
$ sudo nftfwadm restore
```

if not

```
$ sudo nftfwadm clean
```

What's happening here? The first `nftfwadm save` saves the current settings into *nftfw's* backup file. In the event of *nftfw* failing, it will revert to the saved information. You can make this happen by using `restore.` When testing is over, it's also important to run the `clean` command, because *nftfw* won't create a safety backup file if one exists.

Back to Loading the rules

Migrating a Sympl or Symbiosis firewall

If you are installing *nftfw* on a Sympl or Symbiosis system then read this section.

The Debian package is supplied with a python script in */usr/share/doc/nftfw/import_tool*. It can import all the firewall settings from *incoming.d*, *outgoing.d*, *blacklist.d* and *whitelist.d* into *nftfw*. The script contains a lot of built-in information and sample commands. The script is also available in the *import_tool* directory in the *nftfw* source release.

```
$ cd /usr/share/doc/nftfw/import_tool
$ ./import_to_nftfw.py | less
```

will give you the basic information. Running the output through*less* will help with seeing the output. When run with action arguments, the script will tell you what it intends to do. Arguments are needed to force it to write files. The idea is look and check, then write files by adding an argument. You'll need to use *sudo* to update things.

Try:

```
$ ./import_to_nftfw --rules
```

to see what rules will be used by the new firewall files. The script understands about the *local.d* directory and will flag up any local scripts that will need porting into the *nftfw* system.

Once you've updated the firewall, run *nftfw* to load the new settings:

```
$ sudo nftfw -f -v load
```

you can check the rules using the *nft* commands

If are are here from the text above, return to Loading the rules. Otherwise, if you are upgrading a manually installed firewall, complete the end of Section 3 below.

Manually installed *nftfw*

There have been some small changes in the way that *nftfw* works that have been developed to make things simpler for users, and also to remove some of the lesser used features. Mostly, the package installs and expects its control files in */etc/nftfw* and will use working files in */var/lib/nftfw*.

There are a small number of steps that are needed to switch to the package version, the idea here is to retain a working firewall while you are upgrading.

1. Stop cron and systemd

The first thing to do is to stop the background processes that will fire up *nftfw* in the background.

First cron:

```
$ sudo rm /etc/cron.d/nftfw
```

and then if you've loaded the *systemd* files as per the installation instructions:

```
$ sudo rm /etc/systemd/system/nftfw.path /etc/systemd/system/nftfw.service
$ sudo systemctl daemon-reload
```

2. Update your source distribution

pp You are going to need some scripts to help you to migrate and also later to remove the installed source distribution. You don't need (and shouldn't) install or update anything.

3. Are you using part of the Sympl/Symbiosis firewall?

The latest version of *nftfw* does not support *nftfw_base* in *config.ini* that used to point to */etc/{sympl,symbiosis}/firewall*. If you are not using this feature, then skip to section 4.

Otherwise you need to unwind the linkage and ensure that all the *nftfw* information is derived from files in */usr/local/etc/nftfw*. This can be done with the*import_to_nftfw .py* tool. The command will work to move the current settings from *firewall* into the directories in */usr/local/etc/nftfw*. The help information in the tool talks about moving files into */etc/nftfw*, but the tool will work to install files in */usr/local/etc/nftfw* as long as */etc/nftfw* doesn't exist. When using the tool, you won't need to update the database. See Migrating a Sympl or Symbiosis firewall above, then return to complete the para below.

Copy the new version of *nftfw_init.nft* from *etc_nftfw* in the source directory to */usr/local/etc/nftfw*. There are some recent changes in this file. Having updated */usr/local/etc/nftfw*, you can edit *config.ini* to remove or comment out the definition for *nftfw_base* and your current version of *nftfw* can be used to update the firewall. If you want to check that it's all working as expected: n

```
$ sudo nftfw -x -f load
```

can be used to test loading from the source files without affecting the firewall.

4. Delete the nftfw_ installation

Return to the source distribution and run:

```
$ cd _your source_
$ sudo ./Uninstall.sh
```

It will search for what's installed and where on your system, and ask if you want to delete it.

- On the first run, say 'y' to the dry-run question, it will print the commands that it intends to run.
- To retain the control directories, answer 'no' to 'Remove nftfw controls'. Cautious people may like to backup the two control directories to say */tmp* before running the script.
- Say 'yes' to all the other questions.

The script will ask you to confirm your selection before actually doing the deletion deed.

5. Move your directories

```
$ sudo mv /usr/local/etc/nftfw /etc
$ sudo mv /usr/local/var/lib/nftfw /var/lib
```

nftfw will find the files the next time it's run.

6. Ready for package installation now

The package will install several new and amended rules in */etc/nftfw/rule.d*. It's also a good idea to remove */etc/nftfw/config.ini* and */etc/nftfw/nftfw_init.ini* before installing the package. They will be reinstalled from up-to-date versions.

The new versions ensure that the rules match the *nftfw_init.nft* template. Also, importantly, the installed *config.ini* will not make *nftfw* write into */etc/nftables.conf* until you edit the value. The installation will write its versions in */etc/nftfw/nftables.conf*, which can be deleted later. The*config.ini* file will need editing as part on the commissioning process to make *nftfw* install the file in */etc*.

If things go wrong, you can always load the firewall settings from */etc/nftables.conf* using:

```
$ sudo nft -f /etc/nftfables.conf
```

Return to Install the package.