Dialup firewalling with FreeBSD

Marc Silver

marcs@draenor.org

$FreeBSD: doc/en_US.ISO8859-1/articles/dialup-firewall/article.sgml,v 1.43 2008/09/01 22:37:57 keramida Exp $

This article documents how to set up a firewall using a PPP dialup with FreeBSD and IPFW, and specifically with firewalling over a dialup with a dynamically assigned IP address. This document does not include information on setting up an initial PPP connection. For more information on setting up a PPP connection, consult the ppp(8) manual page.

1 Preface

This document outlines the steps required to set up firewalling with FreeBSD when an IP address is assigned dynamically by your ISP. While every effort has been made to make this document as informative and correct as possible, you are welcome to mail any corrections, comments or suggestions to the author at <marcs@draenor.org>.

2 Kernel Options

In order to use IPFW, support for it must be compiled into the kernel. For more information on how to recompile the kernel, please see the kernel configuration section in the Handbook (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/kernelconfig.html). The following options must be added into your kernel configuration file for IPFW support:

```
options IPFIREWALL
```

Enables the kernel firewall code.

> **Note:** This document assumes that you are running FreeBSD 5.X. Users running FreeBSD 4.X will need to recompile their kernels with *IPFW2* support. FreeBSD 4.X users should consult the ipfw(8) manual page for more information on using IPFW2 on their systems, and should pay particular attention to the *USING IPFW2 IN FreeBSD-STABLE* section.

`options IPFIREWALL_VERBOSE`

Sends logged packets to the system logger.

`options IPFIREWALL_VERBOSE_LIMIT=`*`500`*

Limits the number of times a matching entry may be logged. This allows you to log firewall activity without the risk of syslog flooding in the event of a denial of service attack. *`500`* is a reasonable number to use, but may be adjusted based on your requirements.

Warning: Once the kernel recompile has been completed, *do not reboot* your system. Doing so may result in you being locked out of your own system. You must only reboot once the ruleset is in place and all the relevant configuration files have been updated.

3 Changing `/etc/rc.conf` to load the firewall

`/etc/rc.conf` needs to be slightly modified in order to tell the system about the firewall and to specify the location for our rules file. Add the following lines to `/etc/rc.conf`:

```
firewall_enable="YES"
firewall_script="/etc/firewall/fwrules"
```

For more information on the functions of these statements take a look at `/etc/defaults/rc.conf` and read rc.conf(5)

4 Enable PPP's network address translation

In order to allow clients on your network to connect via your gateway, you will need to enable PPP's network address translation (NAT). In order to use PPP's NAT functions, add the following lines to `/etc/rc.conf`:

```
ppp_enable="YES"
ppp_mode="auto"
ppp_nat="YES"
ppp_profile="your_profile"
```

Note: Take care to change `your_profile` to the name of your own dialup profile. The profile name should match the name of the dialup connection in your `/etc/ppp/ppp.conf` file.

5 The rule set for the firewall

This is the point where we define the firewall rules for your system. The ruleset that we are about to describe is a generic template for most dialup users. While it will not suit the exact needs of every user, it provides you with a basic idea of how IPFW works and should be fairly easy to customize.

First, let's start with the basics of closed firewalling. Closed firewalling is based on the idea that everything is denied by default. The system administrator may then explicitly add rules for traffic that he or she would like to allow. Rules should be in the order of allow first, and then deny. The premise is that you add the rules for everything you would like to allow, and then everything else is automatically denied.

Following that, let's create the directory where we will store our firewall rules. In this example, we will use `/etc/firewall`. Change into the directory and edit the file `fwrules` as we specified in `rc.conf`. Please note that you can change this filename to anything you wish. This guide merely gives an example of a filename you may want to use.

Now, let's look at a nicely commented sample firewall file.

```
# Define the firewall command (as in /etc/rc.firewall) for easy
# reference.  Helps to make it easier to read.
fwcmd="/sbin/ipfw"

# Define our outside interface.  With userland-ppp this
# defaults to tun0.
oif="tun0"

# Define our inside interface.  This is usually your network
# card.  Be sure to change this to match your own network
# interface.
iif="fxp0"

# Force a flushing of the current rules before we reload.
$fwcmd -f flush

# Check the state of all packets.
$fwcmd add check-state

# Stop spoofing on the outside interface.
$fwcmd add deny ip from any to any in via $oif not verrevpath

# Allow all connections that we initiate, and keep their state.
# but deny established connections that don't have a dynamic rule.
$fwcmd add allow ip from me to any out via $oif keep-state
$fwcmd add deny tcp from any to any established in via $oif

# Allow all connections within our network.
$fwcmd add allow ip from any to any via $iif

# Allow all local traffic.
$fwcmd add allow all from any to any via lo0
$fwcmd add deny all from any to 127.0.0.0/8
$fwcmd add deny ip from 127.0.0.0/8 to any

# Allow internet users to connect to the port 22 and 80.
```

```
# This example specifically allows connections to the sshd and a
# webserver.
$fwcmd add allow tcp from any to me dst-port 22,80 in via $oif setup keep-state

# Allow ICMP packets: remove type 8 if you don't want your host
# to be pingable.
$fwcmd add allow icmp from any to any via $oif icmptypes 0,3,8,11,12

# Deny and log all the rest.
$fwcmd add deny log ip from any to any
```

You now have a fully functional firewall that only allows connections to ports 22 and 80 and will log any other connection attempts. You may now safely reboot and the firewall should be automatically started and the ruleset loaded. If you find this incorrect in any way or experience any problems, or have any suggestions to improve this page, please email me.

6 Questions

1. I get messages like `limit 500 reached on entry 2800` and after that my machine stops logging denied packets that match that rule number. Is my firewall still working?

This merely means that the maximum logging count for the rule has been reached. The rule itself is still working, but it will no longer log until such time as you reset the logging counters. An example of how to clear your counters can be found below:

```
# ipfw resetlog
```

Alternatively, you may increase the log limit in your kernel configuration with the `IPFIREWALL_VERBOSE_LIMIT` option as described above. You may also change this limit (without recompiling your kernel and having to reboot) by using the net.inet.ip.fw.verbose_limit sysctl(8) value.

2. There must be something wrong. I followed your instructions to the letter and now I am locked out.

This tutorial assumes that you are running *userland-ppp*, therefore the supplied rule set operates on the `tun0` interface, which corresponds to the first connection made with ppp(8) (a.k.a. *user-ppp*). Additional connections would use `tun1`, `tun2` and so on.

You should also note that pppd(8) uses the `ppp0` interface instead, so if you start the connection with pppd(8) you must substitute `tun0` for `ppp0`. A quick way to edit the firewall rules to reflect this change is shown below. The original rule set is backed up as `fwrules_tun0`.

```
% cd /etc/firewall
/etc/firewall% su
Password:
/etc/firewall# mv fwrules fwrules_tun0
/etc/firewall# cat fwrules_tun0 | sed s/tun0/ppp0/g > fwrules
```

To know whether you are currently using ppp(8) or pppd(8) you can examine the output of ifconfig(8) once the connection is up. E.g., for a connection made with pppd(8) you would see something like this (showing only the relevant lines):

```
% ifconfig
(skipped...)
ppp0: flags=8051<UP,POINTOPOINT,RUNNING,MULTICAST> mtu 1524
                inet xxx.xxx.xxx.xxx --> xxx.xxx.xxx.xxx netmask 0xff000000
(skipped...)
```

On the other hand, for a connection made with ppp(8) (*user-ppp*) you should see something similar to this:

```
% ifconfig
(skipped...)
ppp0: flags=8010<POINTOPOINT,MULTICAST> mtu 1500
(skipped...)
tun0: flags=8051<UP,POINTOPOINT,RUNNING,MULTICAST> mtu 1524
        (IPv6 stuff skipped...)
                inet xxx.xxx.xxx.xxx --> xxx.xxx.xxx.xxx netmask 0xffffff00
                Opened by PID xxxxx
        (skipped...)
```