

## **IPTABLES Explained Part 4: IPTables and Portsentry, the dynamic duo (how to dynamically block IPs and unblock automagically)**

**IMPORTANT:** This tutorial is oriented for Fedora, but shouldn't be hard to adapt to Debian/Ubuntu. If someone knows the startup scripts well, mail me or please leave a comment with the changes needed.

Ok, as we saw in the previous chapters of this tutorial series, iptables is an amazingly flexible and powerful tool. By now you should have a good grasp of what iptables can do. It is time to bring in the sidekick: Portsentry. Due to the fact that portsentry has no pre-built packages (to my knowledge) for any distributions, you will have to download it from here:[\[Sourceforge\]](#) ; and build it. The process is very easy BUT (yes there is always a but) it requires you to edit one file. More on that a bit down the road. First, what does Portsentry do? Well, you see Portsentry is a nifty little program that listens on your Linux (or any Unix, but for now Linux) box on a set of defined (by you) ports. That way, when someone port scans you to find open ports, say for example ssh or web or any other open port that they could attack, Portsentry will notice this and take an appropriate action (also defined by you!).

You can do very cool things like immediately block the IP, so even if they find port 22 (SSH) open, by the time they try to connect to it, they are explicitly blocked from accessing port 22 or ANY port, if you so choose. The only problem with this scenario is, that, for example, if you host a big website (like <http://blog.2blocksaway.com> : ) ), and you run Portsentry, you will get scanned a lot. The internet is a bad place and port scanning seems to be a hobby for any script kiddie nowadays. This in itself wouldn't be a problem, but static IPs are rare and most port scanners are run from dynamic IPs which get re-assigned after a certain time period to other people (computers). I think you see where the problem comes in. Unless you constantly maintain your iptables blocked list, you will have soon a lot of IPs banned and even legit users won't get access to your site anymore.

So, you need to periodically "clean" your iptables. In order to do that we will create a loop into your iptables script, make it redundant, in case you have to reboot, AND make "old IPs" get cleaned automatically every 5 days. Sounds good? Great, only a few steps and you have this setup running.

### **Step 1: GET AND INSTALL Portsentry**

(version 1.2 at the time of this writing) from the link above and extract it. You will have a directory called portsentry\_beta. Enter it and open the file portsentry.c, specifically line 1584 with your favorite editor. If you want to use vi try this: `vi +1584 portsentry.c`. You will see in that line, that it is word wrapped wrongly. The line looks like this:

```
printf ("Copyright 1997-2003 Craig H. Rowland  
sourceforget dot net>\n");
```

fix it to make it one line:

```
printf ("Copyright 1997-2003 Craig H. Rowland \n");
```

save the file and exit. Well done, this was the hard part. Now come the easy ones. Type at your command line:

```
make linux && make install
```

and hit enter.

*Notice you need gcc installed but most distros come with it by default.*

Once done the program will inform you that it is now installed in: `/usr/local/psionic/portsentry/`  
Let's memorize that and keep going.

## Step 2: EDIT your current iptables script.

We know by now that the iptables script is actually nothing more than a simple shell script which is run. This means we can use any shell script action within the iptables script. <- A GOOD THING! First create a directory, if you have not done so, where your scripts will all be in. In my case I created `/opt/flosse`. This is where most of your magic will happen. Copy your current iptables shell script there or work in the directory where it is at now, `/root` is NOT an option!

When you have the directory and the script ready, create an empty file called `blocked.list` (*touch blocked.list*) in that directory. In this file Portsentry will add all the blocked IPs and other information. Now we come to the edit part of your iptables script. You need to create a new table and put it at the top of your INPUT/OUTPUT/FORWARD chains (if you don't use FORWARDING, then don't add it there :). You need to create a chain that drops packets from a certain source immediately, but this source will be added dynamically AND, in case of a re-run of the script, you do not want to lose all the collected bad hosts. So, we just add an actual script part into our iptables script. It should look something like this:

```
iptables -N blocked-list
for i in `cat blocked.list`
do iptables -A -i eth0 -p tcp,udp -s $1 -d 0.0.0.0/0 -j DROP
done
iptables -A INPUT -j blocked-list
iptables -A OUTPUT -j blocked-list
```

What this does is, it creates a new chain called `blocked-list`, and then for each line in the file `blocked.list`, it adds the first value as an IP to the `blocked-list` table with the action to drop the traffic for TCP and UDP traffic in the network card `eth0`. You might have to change the network interface for your setup, and of course you can modify the iptables line as you want it (for example log and then drop).

Great, so now we have an iptables script that will take all the IPs out of our blocked list and add them as we need to. EXCELLENT! We are about 50% done, but please bear with me, it will be worth it.

## Step 3: CONFIGURE Portsentry to take certain actions.

First off, there are 2 config files for port sentry. Yes, I know, editing config files blah blah blah. Look on the bright side, you will have a dynamic firewall later one that blocks automatically and unblocks after a certain time frame. It's extremely LOW maintenance. Heck you only need to check the logs, the rest will be done for you! Anyway, the 2 config files are:

`/usr/local/psionic/portsentry/portsentry.ignore` where you configure your own networks. And `/usr/local/psionic/portsentry/portsentry.conf` where you configure your actions. First change the lines:

```
BLOCK_UDP="1"
BLOCK_TCP="1"
```

to:

```
BLOCK_UDP="2"
BLOCK_TCP="2"
```

Then set the line `SCAN_TRIGGER="0"` to `SCAN_TRIGGER="2"`.

This way Portsentry will run an external command only and it won't block immediately everyone. You can only uncomment one "action", so, we want to uncomment the line `KILL_RUN_CMD` and change it to this:

```
KILL_RUN_CMD="/opt/flosse/block.sh $TARGET$ $PORT$"
```

Using this option, and changing the directory to your own script directory, will execute `block.sh` and passing it 2 variables. Save the file and exit.

#### Step 4: CREATE block.sh.

Yes, you are right and I hope you noticed it, the script block.sh has not been created yet, but no worries, here it is:

```
#!/bin/sh
#
#
# block script , a response action for Portsentry
#
# by Flosse for 2blocksaway.com (c) 2007
#
today=`date +%Y%m%d`
dir=/opt/flosse
#
#First we find out if we have it blocked already:
#
for i in `cat blocked.list`
do

    if [ ! "grep '$1'" == "0" ]
    exit;

#
#If there is more then 0 matches, we exit. Otherwise we continue.
#
    else
        #
        # first we add that IP and the current date to the blocked.list file
        #
        echo $1 $today >> $dir/blocked.list
        #
        # Then we add that block to the iptables that are currently in effect
        #
        iptables -A blocked-list -i eth0 -p tcp,udp -s $1 -d 0.0.0.0/0 -j DROP
        #
        #And finally we save the current setup so that at a reboot
        # all those settings and blocked IPs will be in effect again
        #
        /sbin/iptables-save
        #
        #Then we log, just for kicks the IP and what ports it scanned.
        # you can comment this out if you want to.
        echo $1 scanned $2 >> $dir/scan.log
    fi
done
```

It is a very simple script and just copy it as block.sh into your scripts directory (or download it from the files page). Notice this contains Fedora specific options (such as the iptables-save command). It is self explanatory I think and there are the comments as to what each thing does. Please remember to modify in all my scripts the \$dir variable to your script directory. You can periodically check the scan.log file to see what has been going on, it might surprise you. :)

#### Step 5: CLEANING the old Ips.

Ok so we have so far accomplished the following: Portsentry, when started, will listen on specific ports, and based on those ports execute a script, which in turn will add a block rule and an entry to blocked.list. The file blocked.list will be read in case you make changes to your iptables script and have to run it manually again. All your IPs would be lost otherwise, but this way you have them constantly there. It will also make a log entry in scan.log. Pretty good I would say, only thing missing what I promised is getting rid of those old IPs that might be actually occupied by "normal" users by now. Well we create another script which will be listed in a second, and copy it to one of the following directories: /etc/cron.daily, cron.hourly or cron.monthly I would recommend cron.daily and since our script will be adjusted to remove any entries older then 5 days, that is teh way to go for this tutorial. Copy the following script there and make it executable (*chmod a+x*) and root owned (*chown root.root block.cron*).

```

#!/bin/sh
#
#Cronscript to find old dates in files basedon patterns
#
# by Flosse for 2blocksaway.com (c) 2007
#
dir=/opt/flosse
dateago=(date --date='5 days ago' +%Y%m%d)

cat $dir/blocked.list | sed -e '/$dateago/d' > $dir/blocked.list.tmp
rm $dir/blocked.list && mv $dir/blocked.list.tmp $dir/blocked.list

```

What this simple thing does is, it scans the file blocked.list for a specific pattern every day (since you put it into cron.daily) and removes the lines that are 5 days old. There are no fail saves yet for even older entries, since this is supposed to run every day. If anyone wants to contribute please comments or mails are welcome. I chose the "temporary file" way with sed simply so that it is more "debuggable" and user friendly. This way EVERYONE understands it.

### Step 6: CLEANING up.

We are pretty much done, All thats missing is what? Correct, making Portsentry startup automatically every time to start your machine AND usable as a "service" command within Fedora. This is a simple implementation that gives you 3 options: start the service, which will start Portsentry in TCP and UDP listen mode, STOP which will stop the process and EXTREME-START which will start Portsentry with ALL options. Edit your portsentry.conf files before you use this and READ up on it.

```

#!/bin/bash
#
#chkconfig: 3456 9 91
#
#description: start the portsentry service
#
#Simple Portsentry service file
#
# By Flosse for 2blocksaway.com (c) 2007
#
#This file is made available under the GNU Public License v.2
#
. /etc/rc.d/init.d/functions
[ -e /etc/sysconfig/network ] && . /etc/sysconfig/network
[ "${NETWORKING}" = "no" ] && exit 0
binary=portsentry
dir=/usr/local/psionic/portsentry

start() {
    echo -n $"Starting $binary with TCP and UDP monitoring:"
    daemon $dir/portsentry -tcp -udp
    RETVAL=$?
    echo
    [ $RETVAL -eq 0 ] && touch /var/lock/portsentry
    return $RETVAL;
}

stop () {
    echo -n $"Stopping $prog:"
    if [ -e /var/lock/portsentry ] ; then
        killproc portsentry -TERM
        RETVAL=$?
    fi;
    rm -f /var/lock/portsentry
    if [ $RETVAL -eq 0 ]; then
        success;
    echo;
}

```

```

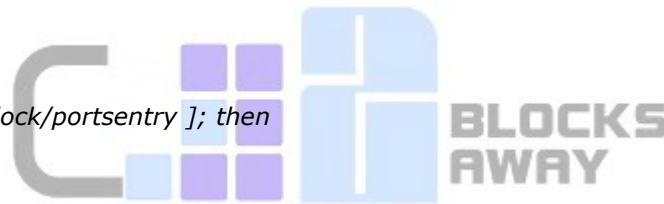
else
    failure;
    echo;
fi;
}

extreme-start() {
    echo -n $"Starting $binary with EXTREME monitoring be careful! :)"
    daemon $dir/portsentry -tcp -udp -atcp -audp -stcp -sudp
    RETVAL=$?
    echo
    [ $RETVAL -eq 0 ] && touch /var/lock/portsentry
    return $RETVAL;
}

case "$1" in
    start)
        start
        ;;
    stop)
        stop
        ;;
    status)
        status
        ;;
    reload)
        reload
        ;;
    restart)
        stop
        start
        ;;
    condrestart)
        if [ -f /var/lock/portsentry ]; then
            stop
            start
        fi
        ;;
    *)
        echo "Usage: $0 {start|stop|restart|reload|condrestart|status}"
        RETVAL=1
esac

exit $RETVAL

```



Copy that file called portsentry into your /etc/rc.d/init.d directory and use the commands:

```

chkconfig --add portsentry
chkconfig --level 23456 on

```

in order to start it at boot. To start it then immediately type service portsentry start and it should tell you that it started OK.

All files are zipped and can be downloaded from the website.

Hope you enjoyed this.