Adaptive Rule Blocking - SSH Brute Force

Author - Nic Maurel

Just recently, I had the unpleasant experience to find out that some was actually trying to hack my external sftp server. Now you ask me why would anyone want to have one of these publicly accessible, and I will tell you that it was a business requirement. How did I find out someone was trying to hack me? Well I was doing a routine check through my system logs when I found +- 100 unsuccessful logins at the root password then I definitely knew someone was trying their luck. What does one do when they acquire this information, well straight away I :

- Hardened the root password - uppercase, lowercase, characters and punctuation
- Removed any unused accounts
- Hardened passwords on existing accounts
- Under the *sshd_config* - "*PermitRootLogin no*" and "*Protocol 2*"

But all this stuff still doesn't prevent this guy or guys from hitting my box. If all clients connecting have static ips then it's simple just firewall them. But what if they use dsl and connect from different ips? Aaah, now we need an adaptive rule blocking script. This script will search the log files periodically for the the source ip of the Failed logon attempts and add them to an *iptables* drop rule. Here is my simple script which makes use of *iptables*..

```
## Script Name - blacklistscript
##DEFINING VARIABLES
IPT=/sbin/iptables
SEARCHDIR=/var/log/messages
BLACKLIST=/admin/conf/badips
EXCLUDE="<your internal  IP address range"

## THIS IS THE BIT THAT FINDS THE IP'S
## ADAPT AS NEEDED.
## Note: the grep -v invalid is excluded because we are not concerned with logins to
invalid usernames
BADIP=`cat $SEARCHDIR | grep -e "Failed" | grep -v invalid | grep -v $EXCLUDE | gawk '{
print $11 }' | sort | uniq`

## CLEAR OUT OLD RULEBASE
$IPT -F
$IPT -X
$IPT -N BLACKLIST
$IPT -A INPUT -i <incoming interface> -j BLACKLIST

## CREATE SPECIFIC RULES AND ADD TO HISTORY
for x in `echo $BADIP`
 do
    echo $x >> $BLACKLIST
    $IPT -A BLACKLIST -s $x -i <incoming interface> -p tcp --dport 22 -j DROP
done
$IPT -A BLACKLIST -j RETURN >> $BLACKLIST
```

This script will parse through the */var/log/messages* file looking for bad ssh logins, get the source address and then ban that address. Each time the script is run it will clear out the old banned entries and create new ones. For a history of which addresses have been banned, browse through the */admin/conf/badips* file. It is a simple thing to find out if a certain source has been logged more then once. Once you have the script running well, you can now add it to your *crontab*

```
*/5 * * * * /admin/bin/blacklistscript
```

Conclusion

There are times when checking system logs can be tedious and boring, but never underestimate how exciting it can be when "foreign entries" in logs appear. Your duty as a security administrator is never ending but when you add scripts like this it can give you a bit of peace of mind and the ability to work smarter rather than harder. Simple but effective, that's all for now.