Preventing brute force attacks with Fail2ban, Denyhosts and BlockHosts / Ban ips that cause multiple authentication errors

Author : admin

Do you have a lot of authentication errors in your **/var/log/messages** file that look like:

Jul 10 16:01:00 pcfreak sshd[3381]: error: PAM: authentication error for illegal user kadilack from 219.143.202.186
Jul 10 16:03:57 pcfreak sshd[3384]: error: PAM: authentication error for illegal user porsche from 116.55.226.131
Jul 10 16:08:52 pcfreak sshd[3418]: error: PAM: authentication error for illegal user windows from 212.254.218.162
Jul 10 16:13:09 pcfreak sshd[3469]: error: PAM: authentication error for illegal user xp from 210.21.208.238

Are you attacked often by **Script Kiddies (pseudo hackers)** trying to brute forcely get access to your **SSH** , **FTP** or **E-mail POP3 Account**?

Does viewing your **/var/log/auth.log** and **/var/log/messages** are filled in with few hundreds of failed logins originating from different hosts on the SSH, FTP and Mail services?
Itâ€™s almost impossible that you havenâ€™t!
Almost everybody who owns a home router running some kind of **home crafted unix bsd or linux based unix distribution** has probably encountered brute force attacks in his most essential services (SSH, FTP, POP3).

Most of this authentication attacks are using the so called Brute, Force Method attack and are albeit trying to break into your system probing with various dictionary based common login names and passwords.

The common user names so often probed for are: **root , toor , admin , john , mike** and so on and so on ..

To deal with this brute force authorization break attacks, many methodologists were invented. Most of which implemented by a number of software programs written to adequately deal with the problem.

The most popular programs I found that resolve issues with authentication break in attempts are:

1. fail2ban
- Fail2Ban is an intrusion prevention framework written in the Python
Fail2ban is able to protect against intursion attempts over the protocols FTP, SSH, SFTP and POP3, HTTP (Apache), Kerberos, MTAs. Mailservers, VPN and probably even more.
Likewise fail2ban seems to be becoming the de-facto standard proram against brute force attacks

2. DenyHosts
- DenyHosts is a script intended to be run by Linux system administrators to help thwart SSH server attacks (also known as dictionary based attacks and brute force attacks).

Though Denyhosts is a superb piece of software according to the user feedbacks Iâ€™ve red online, it has currently only support for the SSH protocol.

A good thorough article On how to use DenyHosts to prevent SSH Dictionary Login attacks is available here

3. Blockhosts
- Blockhosts is a script that checks how many failure attempts are available in your Linux Server log files and whenever a certain amount of failed attempts are reached, blockhosts is also able to: use /etc/hosts.allow file, set-up null-routing for the intruder source host address or by setting up **iptables deny rules to drop packets from the intruder host**
Blockhosts is written in Python so if you intend to use it make sure you have an installed and workable python interpreter version 2.3 or higher.
Blockhosts is also able to both block up brute force attacks to the SSH as well the FTP network services.
A good article on Brute Force Protection with BlockHosts on Debian Linux is found here

After I have reviewed all of the up three mentioned brute force authentication failure brute force prevention applications, I decided to go with **fail2ban** since it looked like the most promising and the most supported one.

A good quick and dirty article on How to setup fail2ban to refuse brute-force attacks is here â€“ the article is described on how to set up fail2ban to ban brute force probers through **/etc/hosts.deny**

Another good worthy to take a look at article about Preventing Brute Force Attacks with Fail2ban on Debian Etch can be red here

Here is **how to install fail2ban on Debian Lenny Linux**, I issued:

debian-server:~# apt-get install fail2ban

Before you proceed further in order to configure fail2ban to properly work with your services of choice please edit:

/etc/fail2ban/jail.conf

Next I started the fail2ban daemon.

debian-server:~# /etc/init.d/fail2ban start

I also tried to include in fail2ban support for vpopmail as described in official fail2banâ€™s page wiki
Well it didnâ€™t work out though I spend some time trying to figure out to figure out why itâ€™s not working I eventually couldnâ€™t, it would be my pleasure if some of my readers suggests a good article on how to enable vpopmail pop3 email logins to be checked for mass login failures with fail2ban.

In some cases the support for proftpd which is included by default with fail2ban also refused to work correctly and often times a completely legit FTP logged in users were banned by fail2ban which was

pretty annoying. I didnâ€™t really spend much time to look for what was causing the problem but anyways if somebody has stuck on the same issue, please share about the solution.

Since I couldnâ€™t make fail2ban to work with proftpd because of the improper ip filtering of a completely regular FTP logged in users during some FTP file list operations, I decided to **completely disable fail2banâ€™s proftpd support** to do so I changed in **/etc/fail2ban/jail.conf**

[proftpd]

enabled = true
port = ftp,ftp-data,ftps,ftps-data
filter = proftpd
logpath = /var/log/proftpd/proftpd.log
maxretry = 6

To:

[proftpd]

enabled = false
port = ftp,ftp-data,ftps,ftps-data
filter = proftpd
logpath = /var/log/proftpd/proftpd.log
maxretry = 6

Regardless the installation path taken, if you choose to install fail2ban, I suggest you remove the fail2ban **POSSIBLE BREAK-IN ATTEMPT preventing**
doing so would prevent a legitimate hosts which are lacking a correct **PTR record (also called in sys-admin jargon back resolving)** from being erroneously (wrongly) denied by fail2ban.

To disable the possible break in checks in fail2ban on Linux hosts open all files located in **/etc/fail2ban/filter.d/** directory one by one and comment out anywhere you see the line:

^%(__prefix_line)sAddress .* POSSIBLE BREAK-IN ATTEMPTs*$

To look like:

^%(__prefix_line)sAddress .* POSSIBLE BREAK-IN ATTEMPTs*$

On most of Fedora and CentOS (Redhat RPM based Linux distributions), fail2ban is also really easy to install and should be directly available through **yum package manager**

In order to **install fail2ban on CentOS release 5.5 (Final)** all you need to execute is:

[root@centos-server: fail2ban]# yum install fail2ban

To install and configure fail2ban on FreeBSD 7.2 with pf (packet filter firewall):

freebsd# uname -a;
FreeBSD pcfreak 7.2-RELEASE-p4 FreeBSD 7.2-RELEASE-p4 #0: Fri Oct 2 12:21:39 UTC 2009 root@i386-builder.daemonology.net:/usr/obj/usr/src/sys/GENERIC i386

First you need to install the fail2ban FreeBSD port:

freebsd# cd /usr/ports/security/py-fail2ban
freebsd# make install clean

â€¦.

creating /var/run/fail2ban
running install_egg_info
Writing /usr/local/lib/python2.6/site-packages/fail2ban-0.8.4-py2.6.egg-info

Please do not forget to update your configuration files.
They are in **/usr/local/etc/fail2ban/**.

===> Installing rc.d startup script(s)
===> Registering installation for py26-fail2ban-0.8.4
===> Cleaning for py26-fail2ban-0.8.4

Furthermore it will be necessery to configure within your firewall filter rules a table where fail2ban would automatically include new rules if some of the failure logins events configured within fail2ban configuration files occurs.
For a users which prefer to use **freebsd pf (packet filter)** you will have to need few custom lines to **fail2ban** and to **/etc/pf.conf** files in order to have fail2ban up and running on your BSD.
In **/usr/local/etc/fail2ban/jail.conf** you will need to include the following fail2ban configuration options:

PF jail

[ssh-pf]

enabled = true
filter = sshd
action = pf
sendmail-whois[name=SSH, dest=email at domain.com]
logpath = /var/log/auth.log

Likewise in **/usr/local/etc/fail2ban/action.d/pf.conf** it's necessary to include:

[Definition]

actionstart =

```
actionstop =
actioncheck =
actionban = pfctl -t fail2ban -T add
actionunban = pfctl -t fail2ban -T delete `pfctl -t fail2ban -T show 2>/dev/null | grep `

[Init]

port = ssh
localhost = 127.0.0.1
```

Also you will have to include in your **/etc/pf.conf** the following rules which will create a new fail2ban table in firewall where fail2ban will insert it's deny rules.

```
table persistblock in on $ext_if from
```

Thereafter to configure fail2ban on FreeBSD hosts, you should include in **/usr/local/etc/fail2ban/jail.conf** at least the following code:

```
[ssh-pf]
# this checks if fail2ban jail is switched on and it combines the filter.d/sshd.conf with action.d/pf.conf
enabled = true
filter = sshd
action = pf
logpath = /var/log/auth.log
maxretry = 5

[ssh-ddos]
# this check if fail2ban-jail is switched on and it combines the filter.d/sshd-ddos.conf with
action.d/pf.conf
enabled = true
filter = sshd-ddos
action = pf
logpath = /var/log/auth.log
maxretry = 3
```

Inside **/etc/pf.conf** in the appropriate place you should add:

```
## FILTER RULES
table persist
block in on $EXT_NIC from
```

Where EXT_NIC or $ext_if should be your external interface variable defined in **pf.conf**
To test your new pf.conf definitions with included support for fail2ban issue the command:

```
freebsd# pfctl -nvf /etc/pf.conf
```

If you get a message like:

/etc/pf.conf:20: Rules must be in order: options, normalization, queueing, translation, filtering

while testing your integrity of pf rules. This is a sure sign that you have misplaced the **filter rules shown a bit upwards**
Furthermore you will have to flush and reload the pf firewall rules (nat, filter, queue, state, info, table), before fail2ban is ready to go on BSD
To do so use the command:

pfctl -Fa -f /etc/pf.conf

Let us also not forget to set it to run automatically on system login via the **/etc/rc.conf bsd boot system**

freebsd# echo 'fail2ban_enable="YES"' >> /etc/rc.conf

Lastly we will also have to manually start up fail2ban

freebsd# /usr/local/etc/fail2ban start

Also I suggest you take a look on the down further nice articles on how to install and configure fail2ban on FreeBSD:

FreeBSD SSH port Security 1 wifh fail2ban
FreeBSD SSH port Security 2 with fail2ban
FreeBSD SSH port Security 3 with fail2ban

Until recently there are no publicly known security threats with fail2ban, however bear in mind that using **fail2ban** could also be a security hole if there are some errors in the program log parser.