

## **CMS-Controlled Fail2Ban Jail**

### **Configuration and Details**

#### **Objective**

To have the CMS handle user login banning in an attempt to deter brute force attacks.

#### **CMS Configuration Page**

The following settings are accessible from the CMS administrative backend.

These options are accessible by going to the User Menu Members and Member Options.

#### **User Password Reset Limit**

The number of password resets per time period is limited. If the user attempts to reset their password at a threshold deemed by the HUB administrator as excessive, the following message is displayed.

## **User Failed Login Limit**

The number of failed logins per time period is limited. If the user attempts to reset their password at a threshold deemed by the HUB administrator as excessive, the following message is displayed. This means that an individual's account is temporarily blocked until the time period expires.

## **IPbased Blocked User Limit**

When the threshold of blocked user accounts per IP network is met, the CMS will trigger a Fail2Ban rule which will block incoming requests from an IP address for a period of time. This is the last line of defense as blocking an IP address may have unintended consequences such as

blocking a NATed IP address which several valid users are using to access the hub.

### Assumptions

This approach assumes that the system administrator has configured a jail and a system user account to execute (with sudo) Fail2Ban via the fail2banclient utility.

On Debian Hosts, Fail2Ban should be version 0.9.5.

0.9.51~nd70+1 from <http://neuro.debian.net/debian/> wheezy/main amd64 Packages

### Setup & Configuration

There are a number of subsystems which need to be configured for this scheme to work properly.

#### sudo Configuration

A privileged user which can execute:

```
(root) NOPASSWD: /usr/bin/fail2banclient set hublogin banip [09.]* (root) NOPASSWD:  
/usr/bin/fail2banclient set hublogin unbanip [09.]*
```

This can be accomplished by adding a sudoers rule in /etc/sudoers.d/ that looks like:

```
wwwdata ALL=(root)NOPASSWD: /usr/bin/fail2banclient set hublogin banip [09.]*
```

### Fail2Ban Configuration

The system administrator should configure Fail2Ban to create jails which the CMS can add offending IP address into. The amount of time that the ban is valid is configured in Fail2Ban.

The CMS will simply add IP addresses to the Fail2Ban jail which will trigger the Ban Action as specified in the rule set.

The following configuration are more of an example than anything. A seasoned system administrator will have crafted better rules.

#### *[Sample] Jail Configuration*

```
#  
  
# JAILS  
  
# /etc/fail2ban/jail.local  
  
#  
  
[hublogin] enabled = true  
  
port    = http,https filter    = hublogin  
  
logpath = /var/log/messages banaction = hubloginfailure bantime = 600  
  
findtime = 1  
  
maxretry = 1
```

#### *[Sample] Filter Configuration*

## CMS-CONTROLLED FAIL2BAN JAIL

---

/etc/fail2ban/filter.d/hublogin.conf

# Fail2Ban configuration file

#

[Definition]

# Option: failregex

# Notes.: Regexp to catch known spambots and software alike. Please verify

# that it is your intent to block IPs which were driven by

# abovementioned bots.

# Values: TEXT

#

#We choose something that will never happen

# Since the CMS will control IP's placed in the jails failregex =  
^<HOST>thisfilterwillneverbefound

# Option: ignoreregex

# Notes.: regex to ignore. If this regex matches, the line is ignored.

# Values: TEXT

#

ignoreregex =

### *[Sample] Action Configuration*

# Fail2Ban configuration file

# cat /etc/fail2ban/action.d/hubloginfailure.conf [INCLUDES]

before = iptablescommon.conf

### [Definition]

# Option: actionstart

# Notes.: command executed once at the start of Fail2Ban.

# Values: CMD

#

actionstart = iptables N fail2banhublogin

iptables A INPUT j DROP

iptables I INPUT p tcp j fail2banhublogin

# Option: actionstop

# Notes.: command executed once at the end of Fail2Ban

# Values: CMD

#

actionstop = iptables D fail2banhublogin p tcp j fail2banhublogin iptables F fail2banhublogin

### iptables X fail2banhublogin

# Option: actioncheck

# Notes.: command executed once before each actionban command

# Values: CMD

#

actioncheck = iptables n L fail2banhublogin | grep q fail2banhublogin

# Option: actionban

# Notes.: command executed when banning an IP. Take care that the

# command is executed with Fail2Ban user rights.

# Tags: <ip> IP address

# <failures> number of failures

# <time> unix timestamp of the ban time

# Values: CMD

#

actionban = iptables I fail2banhublogin p tcp dport 443 s <ip> j DROP

iptables I fail2banhublogin p tcp dport

80 s <ip> j DROP

# Option: actionunban

# Notes.: command executed when unbanning an IP. Take care that the

# command is executed with Fail2Ban user rights.

## CMS-CONTROLLED FAIL2BAN JAIL

---

# Tags: <ip> IP address

# <failures> number of failures

# <time> unix timestamp of the ban time

# Values: CMD

#

actionunban = iptables D fail2banhublogin p tcp dport 443 s <ip> j DROP iptables D fail2banhublogin p tcp dport 80 s <ip> j DROP

[Init]

# Defaut name of the chain

#

name = DEFAULT

# Option: protocol

# Notes.: internally used by config reader for interpolations.

# Values: [ tcp | udp | icmp | all ] Default: tcp

#

protocol = tcp

# Option: chain

# Notes specifies the iptables chain to which the fail2ban rules should be



# CMS-CONTROLLED FAIL2BAN JAIL

---

# added

# Values: STRING Default: INPUT chain = INPUT

[Sample] Banned IP address results root@example:/var/www/example# fail2banclient status  
hublogin Status for the jail: hublogin

| Filter

| | Currently failed: 0

| | Total failed: 4

| ` File list: /var/log/messages

` Actions

| Currently banned: 1

| Total banned: 4

` Banned IP list: 192.168.226.1

root@example:/var/www/example# iptables L Chain INPUT (policy DROP)

```
target prot opt source destination fail2banhublogin tcp anywhere  
anywhere ACCEPT all anywhere anywhere
```

```
ACCEPT all anywhere anywhere state RELATED,ESTABLISHED
```

ACCEPT	tcp		anywhere	anywhere		tcpdpt:ssh
ACCEPT	tcp		anywhere	anywhere		tcpdpt:smtp
ACCEPT	tcp		anywhere	anywhere		tcpdpt:mysql
ACCEPT	tcp		anywhere	anywhere		tcpdpt:ldap
ACCEPT	tcp		anywhere	anywhere		tcpdpt:http
ACCEPT	tcp		anywhere	anywhere		tcpdpt:https
ACCEPT	tcp		anywhere	anywhere		tcpdpt:httpalt
ACCEPT	tcp		anywhere	anywhere		tcpdpts:830:831
ACCEPT	tcp		anywhere	anywhere		tcpdpt:http
ACCEPT	tcp		anywhere	anywhere		tcpdpt:https
ACCEPT	tcp		anywhere	anywhere		tcpdpt:httpalt
ACCEPT	tcp		anywhere	anywhere		tcpdpt:1170

## CMS-CONTROLLED FAIL2BAN JAIL

ACCEPT	icmp		anywhere	anywhere		
DROP	all		anywhere	anywhere		

Chain FORWARD (policy DROP)

target	prot	all	opt	source	destination	anywhere	
ACCEPT	all			10.0.0.0/8	anywhere		
ACCEPT				anywhere			ctstate
RELATED,ESTABLISHED,DNAT							
ACCEPT	tcp			anywhere	anywhere	tcp	dpts:830:831
ACCEPT	tcp			anywhere	anywhere	tcp	dpt:http
ACCEPT	tcp			anywhere	anywhere	tcp	dpt:https
ACCEPT	udp			anywhere	anywhere	udp	dpt:domain

Chain OUTPUT (policy ACCEPT)

target    prot opt source                    destination

Chain fail2banhublogin (1 references)							
target	prot	opt	source	destination			
DROP	tcp		container.localhost	anywhere	tcp	dpt:http	
DROP	tcp		container.localhost	anywhere	tcp	dpt:https	

root@example:/var/www/example#

## User Impact

## CMS-CONTROLLED FAIL2BAN JAIL

---

When a large number of people intend on using the CMS, it may be wise to temporarily disable this feature (e.g. conference, class activity, etc). In the past, many conference goers have mistyped their password in a short period of time creating a false positive for normal Fail2Ban operation. This risk is mitigated by the fact that the number of blocked users is observed before triggering Fail2Ban.