Fail2ban Ban brute-force attackers

START HERE

`sudo` **`fail2ban-client status`**
List active jails

`sudo` **`fail2ban-client reload`**
Apply config changes

`sudo` **`fail2ban-client status`** *`<jail>`*
Bans in one jail

`fail2ban-regex` *`<log>`* *`<filter>`*
Test a filter

`sudo` **`fail2ban-client set`** *`<jail>`* **`unbanip`** *`<ip>`*
Unban an address

`sudo` **`fail2ban-client -t`**
Check config syntax

`sudo` **`fail2ban-client set`** *`<jail>`* **`banip`** *`<ip>`*
Ban by hand

`sudo` **`systemctl status fail2ban`**
Is the service up?

Log or journal watched by a jail → `filter regex` → **Failure** one match per IP → `maxretry in findtime` → **Ban** action runs → `after bantime` → **Unban** action undoes it

CONCEPTS

A jail ties a log, a filter and an action together; it bans an IP after maxretry failures within findtime.

Edit jail.local or jail.d/*.local, never jail.conf: .local files override it and survive package upgrades.

Fail2ban only reacts to log lines. It is a complement to a firewall, not a replacement.

SERVICE AND CONFIG CHECK

`sudo` **`systemctl enable --now fail2ban`**
Start now and at boot

`sudo` **`systemctl status fail2ban`**
Check the daemon

`sudo` **`fail2ban-client ping`**
Is the server answering?
Prints 'Server replied: pong' when the daemon is reachable.

`sudo` **`fail2ban-client -t`**
Test the configuration
Run this before every reload: a broken file can stop jails from loading.

`sudo` **`fail2ban-client reload`**
Reload all config

`sudo` **`fail2ban-client reload`** *`<jail>`*
Reload one jail

STATUS AND BANS

`sudo` **`fail2ban-client status`**
List jails

`sudo` **`fail2ban-client status`** *`<jail>`*
Failed and banned IPs of a jail

`sudo` **`fail2ban-client banned`**
All banned IPs, per jail

`sudo` **`fail2ban-client banned`** *`<ip>`*
Which jails ban this IP?

STATUS AND BANS (cont.)

`sudo` **`fail2ban-client set`** *`<jail>`* **`banip`** *`<ip>`*
Ban an IP by hand

`sudo` **`fail2ban-client set`** *`<jail>`* **`unbanip`** *`<ip>`*
Unban an IP in a jail

`sudo` **`fail2ban-client unban`** *`<ip>`*
Unban an IP everywhere

`sudo` **`fail2ban-client unban --all`**
Unban every address

JAILS AT RUNTIME

`sudo` **`fail2ban-client start`** *`<jail>`*
Start a stopped jail

`sudo` **`fail2ban-client stop`** *`<jail>`*
Stop one jail

`sudo` **`fail2ban-client set`** *`<jail>`* **`bantime`** *`<time>`*
Change ban length
Runtime changes are lost on restart; put lasting values in jail.local.

`sudo` **`fail2ban-client set`** *`<jail>`* **`maxretry`** *`<n>`*
Change failure threshold

`sudo` **`fail2ban-client set`** *`<jail>`* **`addignoreip`** *`<ip>`*
Whitelist an IP

CONFIG FILES

`/etc/fail2ban/jail.local`
Your jail overrides
Read order: jail.conf, jail.d/.conf, jail.local, jail.d/*.local. Later files win.*

`/etc/fail2ban/filter.d/`
Filters: failregex per service

`[DEFAULT]`
Section for settings every jail inherits

`enabled = true`
Turn a jail on

CONFIG FILES (cont.)

`bantime = 1h`
Ban length (s, m, h, d, w, mo, y)

`findtime = 10m`
Window for counting failures

`maxretry = 5`
Failures allowed before a ban

`ignoreip = 127.0.0.1/8 ::1` *`<ip>`*
Never ban these hosts or CIDRs

`logpath = /var/log/auth.log`
File(s) to watch, globs allowed

`banaction = nftables`
Firewall backend for bans

`bantime.increment = true`
Lengthen bans for repeat offenders
Option comes from the stock jail.conf; check your version's file for the related bantime. settings.*

FILTERS AND TESTING

`fail2ban-regex` *`<logfile>`* *`<filter-or-regex>`*
Test a filter against a log
The filter may be a name, a file path or a bare regex.

`fail2ban-regex` *`<logfile>`* **`/etc/fail2ban/filter.d/sshd.conf`**
Test a filter file

`fail2ban-regex systemd-journal` *`<filter>`*
Test against the journal

`fail2ban-regex` *`'<log line>'`* *`'<regex>'`*
Test a single line and regex

`fail2ban-regex --print-all-missed` *`<logfile>`* *`<filter>`*
Show lines the filter missed

`<HOST>`
Placeholder in failregex that captures the IP

Fail2ban-regex prints counts of matched, missed and ignored lines. Many missed lines that should fail means the regex is wrong.

LOGS

`sudo` **`tail -f /var/log/fail2ban.log`**
Follow Fail2ban's own log
On journal-only systems use journalctl instead.

`sudo` **`grep Ban /var/log/fail2ban.log`**
Find ban events

`sudo` **`fail2ban-client flushlogs`**
Reopen log files after rotation

Bans are kept in a database, so they are restored after a restart until bantime has passed.

RECIPES

Unban yourself after a lockout

```
# from another route (console, other IP)
sudo fail2ban-client status sshd
sudo fail2ban-client set sshd unbanip <ip>
sudo fail2ban-client set sshd addignoreip <ip>
```

Write and check a custom filter

```
# /etc/fail2ban/filter.d/myapp.conf
[Definition]
failregex = ^.* login failed for .* from <HOST>$

fail2ban-regex /var/log/myapp.log myapp
fail2ban-regex --print-all-missed /var/log/myapp.log myapp
```

Add a jail for a custom filter

```
# /etc/fail2ban/jail.d/myapp.local
[myapp]
enabled = true
filter = myapp
logpath = /var/log/myapp.log
maxretry = 3

sudo fail2ban-client reload
```