# How to make WordPress secure with nginx and fail2ban

WPScan with Metasploit can easily hack a WordPress site - unless you automatically block their access to the PHP level.

## Why

I've recently came across a little presentation on how easy is to hack WordPress[^1]. I'd known it's pretty easy, but unfortunately there are tools making this possible for nearly anyone.

To test how problematic my sites are, I've downloaded WPScan[^2] and ran it against my sites. The result was surprising: The website is unreachable.

After the initial panic that (*I've killed my own site!*) I realized that my long forgotten nginx rules in combination with fail2ban banned the testing IP within the first 2 seconds of the scanning attempt.

You may want to introduce something similar to block scanners on your WordPress.
*( Or ask you hosting provider to do so. )*

I've not written all of these rules myself but unfortunately I cannot recall all the sources I've used during the years to collect this, therefore my thanks go to everyone who ever did anything I've read on this.

**Please bear in mind that security is never bullet-proof - this setup will not save you from someone really attacking your site. There are many layers you need to use and always keep an eye out, revisit the logs, supervise what is happening with your sites.**

## nginx[^3]

In nginx you need to add a new log format. This will contain all the blocked requests, including the IP address; **this should go into the http {} block.**

`/etc/nginx/nginx.conf`, `http` section:

```apache
log_format blocked '$time_local: Blocked request from $remote_addr $request';
```

You'll also need the following rules **in the server {} block** for the site you're setting the protection for:

```apache
# note: if you have posts with title matching these, turn them off or fine-tune
# them to exclude those

## Block SQL injections
location ~* union.*select.*\( {
    access_log /var/log/nginx/blocked.log blocked;
    deny all;
}
location ~* union.*all.*select.* {
    access_log /var/log/nginx/blocked.log blocked;
    deny all;
}
location ~* concat.*\( {
    access_log /var/log/nginx/blocked.log blocked;
    deny all;
}

## Block common exploits
location ~* (<|%3C).*script.*(>|%3E) {
    access_log /var/log/nginx/blocked.log blocked;
    deny all;
}
location ~* base64_(en|de)code\(.*\) {
    access_log /var/log/nginx/blocked.log blocked;
    deny all;
```

```
}
location ~* (%24&x) {
    access_log /var/log/nginx/blocked.log blocked;
    deny all;
}
location ~* (%0|%A|%B|%C|%D|%E|%F|127\.0) {
    access_log /var/log/nginx/blocked.log blocked;
    deny all;
}
location ~* \.\.\/  {
    access_log /var/log/nginx/blocked.log blocked;
    deny all;
}
location ~* ~$ {
    access_log /var/log/nginx/blocked.log blocked;
    deny all;
}
location ~* proc/self/environ {
    access_log /var/log/nginx/blocked.log blocked;
    deny all;
}
location ~* /\.(htaccess|htpasswd|svn) { log_not_found off;
    access_log /var/log/nginx/blocked.log blocked;
    deny all;
}

## Block file injections
location ~* [a-zA-Z0-9_]=(\.\.//?)+ {
    access_log /var/log/nginx/blocked.log blocked;
    deny all;
}
location ~* [a-zA-Z0-9_]=/([a-z0-9_.]//?)+ {
    access_log /var/log/nginx/blocked.log blocked;
    deny all;
}

## Block access to internal WordPress assets that isn't
queried under normal
## circumstances
location ~* wp-config.php {
    access_log /var/log/nginx/blocked.log blocked;
```

```
    deny all;
}
location ~* wp-admin/includes {
    access_log /var/log/nginx/blocked.log blocked;
    deny all;
}
location ~* wp-app\.log {
    access_log /var/log/nginx/blocked.log blocked;
    deny all;
}
location ~* (licence|readme|license)\.(html|txt) {
    access_log /var/log/nginx/blocked.log blocked;
    deny all;
}

## In case you have anything using sqlite as database you
probably want to block
## direct access to those as well
location ~* \.sqlite$ {
    access_log /var/log/nginx/blocked.log blocked;
    deny all;
}
location ~* ^/(SQLite|sqlite) {
    access_log /var/log/nginx/blocked.log blocked;
    deny all;
}
location ~* \.sqlite-journal$ {
    access_log /var/log/nginx/blocked.log blocked;
    deny all;
}
```

## fail2ban[^4]

Add this jail to the jails config:

`/etc/fail2ban/jail.conf`

```ini
[nginx-blocked]
enabled = true
port = 80,443
filter = nginx-blocked
logpath = /var/log/nginx/blocked.log
bantime  = 3600
maxretry = 3
backend = auto
findtime = 86400
banaction = iptables-multiport
protocol = tcp
chain = INPUT
```

Note: the logpath accepts * as wildcard, in case you have more blocked logs.

And also add the filter

`/etc/fail2ban/filter.d/nginx-blocked.conf`

```ini
[Definition]
failregex = ^.* Blocked request from <HOST>.*$
ignoreregex =
```

For the `nat banaction`, please see fail2ban for NAT hosts[^5].

### Optional: syslog

In case you have more than one webserver, you probably want to centralize this. The easiest way is to have fail2ban on the loadbalancer, nginx logging into syslog, syslog pushed to the loadbalancer's syslog. **Rsyslog is a deep topic, so please keep in mind that you'll most probably need to read more about it. This is an example for those who are familiar with syslog.** You can replace the `access_log` lines in the collection above with something similar to:

```
access_log syslog:server=unix:/dev/
log,facility=local7,tag=nginx,severity=warn blocked;
```

in the nginx.conf and:

```
local7.warn /var/log/nginx/blocked.log
```

in you rsyslog.conf.

## Troubleshooting

aaronpk[^6] started to use this setup, but had some troubles verifying it's running and happy.

**Note: these all should be run as root.**

### Check if fail2ban is running

`ps aux | grep fail2ban` should show something like `/usr/bin/python /usr/bin/fail2ban-server` (or similar).

### `bantime` and `findtime`

In my example, bantime is set for an hour and findtime for a day. You may need to tune this according to your needs and traffic.

### verify your regex

If you run `fail2ban-regex -v [path-to-log] [path-to-filter-rule]` it should output something like this:

```
Running tests
=============

Use    failregex file : /etc/fail2ban/filter.d/nginx-
blocked.conf
Use         log file : /var/log/nginx.blocked.log


Results
=======

Failregex: 1452 total
|-  #) [# of hits] regular expression
|   1) [1452] ^.*?Blocked request from <HOST> .*$
|      51.255.65.41  Sun Feb 19 06:31:05 2017
```

```
[ ... lots of IPs here ...]

|      157.55.39.20  Thu Feb 23 12:30:23 2017
`-

Ignoreregex: 0 total

Date template hits:
|- [# of hits] date format
|  [2125] ISO 8601
|  [2] Year/Month/Day Hour:Minute:Second
|  [0] WEEKDAY MONTH Day Hour:Minute:Second[.subsecond] Year
|  [0] WEEKDAY MONTH Day Hour:Minute:Second Year
|  [0] WEEKDAY MONTH Day Hour:Minute:Second
|  [0] MONTH Day Hour:Minute:Second
|  [0] Day/Month/Year Hour:Minute:Second
|  [0] Day/Month/Year2 Hour:Minute:Second
|  [0] Day/MONTH/Year:Hour:Minute:Second
|  [0] Month/Day/Year:Hour:Minute:Second
|  [0] Year-Month-Day Hour:Minute:Second[,subsecond]
|  [0] Year-Month-Day Hour:Minute:Second
|  [0] Year.Month.Day Hour:Minute:Second
|  [0] Day-MONTH-Year Hour:Minute:Second[.Millisecond]
|  [0] Day-Month-Year Hour:Minute:Second
|  [0] Month-Day-Year Hour:Minute:Second[.Millisecond]
|  [0] TAI64N
|  [0] Epoch
|  [0] Hour:Minute:Second
|  [0] <Month/Day/Year@Hour:Minute:Second>
|  [0] YearMonthDay Hour:Minute:Second
|  [0] Month-Day-Year Hour:Minute:Second
`-

Lines: 2127 lines, 0 ignored, 1452 matched, 675 missed
```

If not, the regex is not working.

### Query fail2ban

Fail2ban offers a program names `fail2ban-client` to ask for status, so:

```
fail2ban-client status nginx-blocked
```

```
Status for the jail: nginx-blocked
|- filter
|  |- File list:    /var/rlog/mekare.petermolnar.eu/
nginx.blocked.log
|  |- Currently failed: 68
|  `- Total failed: 586
`- action
   |- Currently banned: 6
   |  `- IP list:   122.167.147.40 176.93.112.88
193.154.116.218 203.118.160.75 79.122.16.39 157.55.39.20
   `- Total banned: 139
```

If this is returning some error, it's either the client not being able to talk to the server (check the socket, maybe add `-s [socketpath]` to the client), or the service/jail is not running'.

## Dump iptables

When you're certain an IP should have been or is blocked, you can also check iptables with `iptables-save`.

### Links

1. http://www-personal.umich.edu/~markmont/awp/
2. http://wpscan.org/
3. http://nginx.org/
4. http://www.fail2ban.org/wiki/index.php/Main_Page
5. https://petermolnar.net/linux-tech-coding/fail2ban-nat-hosts/
6. http://aaronparecki.com/

Created by Peter Molnar <mail@petermolnar.net>, published at 2015-04-07 14:40 UTC, last modified at 2021-10-31 15:57 UTC , to canonical URL https://petermolnar.net/article/secure-wordpress-with-nginx-and-fail2ban/ , licensed under CC-BY-4.0 .