Gunicorn Python app server

CONCEPTS

app:app means module:variable. Gunicorn serves one app per process; put nginx in front for TLS and static files.

Client browser
↓ HTTP
nginx TLS, static
↓ proxy_pass
Master gunicorn, no requests
↓ fork
Workers run your app

The master only manages workers; it never handles requests. A worker silent past the timeout is killed and replaced.

Rule of thumb for sync workers: 2 x cores + 1. Threads or async workers help when requests wait on I/O.

RUN

gunicorn *<module>*:*<app>*
Serve a WSGI app

gunicorn '*<module>*:*<factory>*()'
Call an app factory

gunicorn -w *<n>* *<module>*:*<app>*
Set worker count

gunicorn -b *<host>*:*<port>* *<module>*:*<app>*
Bind a host and port

gunicorn -b unix:*<path>* *<module>*:*<app>*
Bind a Unix socket

gunicorn *<project>*.**wsgi**
Run a Django project

gunicorn --reload *<module>*:*<app>*
Auto-restart on code change

gunicorn --check-config *<module>*:*<app>*
Check config, then exit

gunicorn --print-config *<module>*:*<app>*
Show resolved settings

gunicorn --chdir *<dir>* *<module>*:*<app>*
Change directory first

RUN (cont.)

gunicorn -e *<KEY>*=*<value>* *<module>*:*<app>*
Set an environment variable

gunicorn -v
Print the version

GUNICORN_CMD_ARGS="--bind=:8000 -w 4"
Settings from the environment

WORKERS

-k sync
Default: one request per worker

-k gthread --threads *<n>*
Threaded workers

-k gevent
Async workers via gevent

-k asgi
Run an ASGI app

--max-requests *<n>*
Recycle a worker after n requests

--max-requests-jitter *<n>*
Stagger those restarts

-t *<sec>* **/ --timeout** *<sec>*
Kill silent workers after n seconds

--graceful-timeout *<sec>*
Grace period on restart

--preload
Load app before forking

--worker-tmp-dir /dev/shm
Heartbeat files in RAM

-u *<user>* **-g** *<group>*
Drop privileges

-p *<file>*
Write a PID file

Pick a worker

Class	Fits
`sync`	simple CPU-bound apps
`gthread`	mixed or blocking I/O
`gevent`	many slow connections
`asgi`	async frameworks

LOGGING

--access-logfile -
Access log to stdout

--error-logfile -
Error log to stderr

--log-level *<level>*
debug, info, warning, error

LOGGING (cont.)

--access-logformat "*<fmt>***"**
Custom access log format

--capture-output
Send app stdout/stderr to the log

SIGNALS

kill -HUP *<master-pid>*
Reload config, replace workers

kill -TERM *<master-pid>*
Graceful shutdown

kill -TTIN *<master-pid>*
Add one worker

kill -TTOU *<master-pid>*
Remove one worker

kill -USR1 *<master-pid>*
Reopen log files

CONFIG FILE

gunicorn.conf.py

```
bind = "127.0.0.1:8000"
workers = 4
worker_class = "gthread"
threads = 2
timeout = 30
accesslog = "-"
errorlog = "-"
preload_app = True
max_requests = 1000
```

Precedence: command line beats config file beats defaults. ./gunicorn.conf.py is read automatically.

DEPLOY

systemd unit

```
[Service]
User=www-data
WorkingDirectory=/srv/app
ExecStart=/srv/app/.venv/bin/
gunicorn -c gunicorn.conf.py
app:app
ExecReload=/bin/kill -s HUP
$MAINPID
Restart=on-failure
```

nginx in front

```
location / {
    proxy_pass
http://127.0.0.1:8000;
    proxy_set_header Host $host;
    proxy_set_header X-Forwarded-
For $remote_addr;
}
```

--forwarded-allow-ips *<ips>*
Trust proxy headers from these IPs