The following environment variables are available for CA configuration:

- `DOCKER_STEPCA_INIT_NAME` (**required**) the name of your CA—this will be the issuer of your CA certificates
- `DOCKER_STEPCA_INIT_DNS_NAMES` (**required**) the hostname(s) or IPs that the CA will accept requests on
- `DOCKER_STEPCA_INIT_PROVISIONER_NAME` a label for the initial admin (JWK) provisioner. Default: "admin"
- `DOCKER_STEPCA_INIT_SSH` set this to a non-empty value to create an SSH CA
- `DOCKER_STEPCA_INIT_PASSWORD` specify a password for the encrypted CA keys and the default CA provisioner. A password is generated by default. Note: In a production environment, a more secure option for specifying a password is to use the manual installation process, below.

Once `step-ca` is running, the CA's URL and SHA256 fingerprint are all clients need to bootstrap with the CA.

Let's bootstrap the `step` client. Run:

```
{
  CA_FINGERPRINT=$(docker run -v step:/home/step smallstep/step-ca step certificate fingerprint certs/root_
  step ca bootstrap --ca-url https://localhost:9000 --fingerprint $CA_FINGERPRINT
}
```

Output:

```
The root certificate has been saved in /Users/alice/.step/certs/root_ca.crt.
Your configuration has been saved in /Users/alice/.step/config/defaults.json.
```

Your CA is ready for use. You can view your CA password via:

container again and write that file:

```
docker run -it -v step:/home/step smallstep/step-ca sh
```

**Inside your container**, write the file into the expected location:

```
echo "<your password here>" > /home/step/secrets/password
```

Your CA is configured and ready to run.

### 4. START `STEP-CA`

The CA runs an HTTPS API on port 9000 inside the container. Expose the server address locally and run the `step-ca` with:

```
docker run -d -p 9000:9000 -v step:/home/step smallstep/step-ca
```

Now, on your Docker host, bootstrap your `step` client configuration:

```
{
  CA_FINGERPRINT=$(docker run  -v step:/home/step smallstep/step-ca step certificate fingerprint /home/step
  step ca bootstrap --ca-url https://localhost:9000 --fingerprint $CA_FINGERPRINT
}
```

Output:

```
step ca root root_ca.crt
```

Output:

```
The root certificate has been saved in root_ca.crt.
```

Next, let's launch a web server secured by HTTPS:

```
{
cat <<EOF > server.py
import BaseHTTPServer, ssl

class HelloHandler(BaseHTTPServer.BaseHTTPRequestHandler):
    def do_GET(self):
        self.send_response(200);
        self.send_header('content-type', 'text/html; charset=utf-8');
        self.end_headers()
        self.wfile.write(b'\\n\\xf0\\x9f\\x91\\x8b Hello! Welcome to TLS \\xf0\\x9f\\x94\\x92\\xe2\\x9c\\x8

httpd = BaseHTTPServer.HTTPServer(('', 8443), HelloHandler)
httpd.socket = ssl.wrap_socket(httpd.socket,
                   server_side=True,
                   keyfile="localhost.key",
                   certfile="localhost.crt",
                   ca_certs="root_ca.crt")
httpd.serve_forever()
EOF
```