

Secure Programming

Week 5

Path Traversal Attack

Performing the Attack

We are going to try to gain unauthorised access to the config.env folder because it contains all the username/ passwords for the site. Realistically, such a file should not exist, but if it did, it should have strict access controls set. The config.env file is located in the secrets folder (outside the vagrant folder on your host):

`\django-coffeeshop\coffeeshop\secrets`

When you accessed the secrets.txt file, the application fetched the file from: \coffeeshop\vagrant\coffeeshopsite\coffeeshop\files. The objective is to now traverse out of the folder and up to the config.env file. We do this using the dot-dot-slash notation (`.. /`).

1. What happens when you try to access the config.env file using this URL?

When to try to access the config.env file using the `../` traversal, the application downloads the config.env file (username and password).

```
export DBOWNER=coffeeshopowner
export DBOWNERPWD='Arabica123'
export DBOWNEREMAIL='admin@coffeeshop.com'
export DBADMIN=admin
export DBADMINPWD='Robusto123'
export DBADMINEMAIL='admin@coffeeshop.com'
export DBUSER1=bob
export DBUSER1PWD='bobPass123'
export DBUSER1EMAIL='bob@bob.com'
export DBUSER1FIRSTNAME='Bob'
export DBUSER1LASTNAME='Smith'
export DBUSER2=alice
export DBUSER2PWD='alicePass123'
export DBUSER2EMAIL='alice@alice.com'
export DBUSER2FIRSTNAME='Alice'
export DBUSER2LASTNAME='Adams'
export SECRET_KEY='-l-({@eam)6qhk!_s7r4yyab)apccu8(ut(5pj-vx59xek'o'
export GRAYLOG_ROOT_PASSWORD='grayPass123'
```

2. What was the final URL that successfully carried out the dot-dot-slash attack?

Used 4 dot-dot-slash notation (`../../../../`) to get to the `secrets/config.env`:

http://localhost:8080/download_file/?file=../../../../secrets/config.env

Fixing the Path Traversal Vulnerability

To fix this vulnerability, we need to fix the code so that a user cannot force browse out of the location using the `../` notation. The first security fix we will apply involves normalising the path to the secrets.txt file. We do this first by replacing the line of code:

```
file_path = os.path.join(base_directory, file_name)
```

with:

```
file_path = os.path.normpath(os.path.join(base_directory, file_name))
```

After normalisation, the line:

```
if not file_path.startswith(base_directory):  
    return HttpResponse("Access denied")
```

3. What is the result?

After the fix, when attempting the same dot-dot-slash attack (e.g.

`?file=../../../../secrets/config.env`), the application returns “Access denied” instead of exposing the sensitive file.

The normalization ensures that any traversal attempts using `../` are resolved, keeping the path restricted to the /files/ directory.



The screenshot shows a web browser address bar with the URL `http://localhost:8080/download_file/?file=../../../../secrets/config.env`. Below the address bar, the text "Access denied" is displayed, indicating that the application successfully blocked the attempt to access a file outside the allowed directory.

Fixing the Vulnerability (link to the commit)

[DanyilT/django-coffeeshop](#) repo forked from [stephen-oshaughnessy/django-coffeeshop](#)

Prepare the vulnerability in code for Path Traversal Attack (download_file):

[DanyilT/django-coffeeshop/commit/e0e318dbc928365c8f614c49b7a15614198eb07d](#)

Fixing Path Traversal Attack vulnerability (download_file):

[DanyilT/django-coffeeshop/commit/1d0a1614c56b6c41e949737e8a0ffcbd54e20317](#)