Django Security Headers

Nov 03, 2019

Contents:

1 Default configuration **3**

2 Optional configuration **5**

3 Development settings **7**

4 Default settings **9**
- 4.1 Django settings . . . 9
- 4.2 Django-CSP settings . . . 11
- 4.3 Middleware settings . . . 12

5 Middleware **15**

6 Models **17**

7 Contributing **19**

Python Module Index **21**

Index **23**

Danger: The default security settings set by this package are aggressive, especially concerning HSTS. Do not deploy in a production environment with default settings unless you are certain your server configuration is compatible.

CHAPTER 1

Default configuration

To apply default security headers to all responses:

1. Installation

   a. From `pip`

   ```
   pip install django-security-headers
   ```

   b. To access the `scan` function from `httpobs`, add the following to your project's dev requirements

   ```
   -e git+https://github.com/jsumnerPhD/http-observatory#egg=httpobs
   ```

2. Add the `csp` and `security_headers` middlewares. For Django 1.11, also add the `samesite` middleware

   ```
   MIDDLEWARES = [
     "django.middleware.security.SecurityMiddleware",
     "csp.middleware.CSPMiddleware",
     "security_headers.middleware.extra_security_headers_middleware",
     "django_cookies_samesite.middleware.CookiesSameSite", # Not needed for Django 2.
   ↪2
       ...
   ]
   ```

3. Add `security_headers` to your `INSTALLED_APPS`.

   ```
   INSTALLED_APPS = [
     ...
     "security_headers",
     ...
   ]
   ```

This will expose a simple admin interface for specifying safe domains.

4. Import default security settings in your project `settings.py`

```
from security_headers.defaults import *  # noqa F403
```

CHAPTER 2

Optional configuration

If you included step 1b, you can add a scan link to `urls.py`. Accessing this link will run a scan against `https://127.0.0.1:8000/<path>` where the path is determined from reversing `url_name`. Note that the sslserver must be running in parallel to the request.

```
from security_headers.views import scan_url

if settings.DEBUG:
    urlpatterns += i18n_patterns(
        url(r"^security/(?P<url_name>[\w-]+)/", scan_url, name="scan")
    )
```

For newer Django syntax

```
urlpatterns += [path("security/<slug:url_name>/", scan_url, name="scan")]
```

To access template tags provided by `django-csp`, add `csp` to `INSTALLED_APPS`

```
INSTALLED_APPS = [
  ...
  "security_headers",
  "csp",
  ...
]
```

To use the sslserver (provided by django-sslserver through `./manage.py runsslserver`)

```
INSTALLED_APPS = [
  ...
  "security_headers",
  "csp",
  "sslserver",
  ...
]
```

CHAPTER 3

Development settings

During development, you will need to overwrite some default settings if not using the ssl server. At the very end of your `settings.py` file, include (this is conveniently done through an imported `local_settings.py`):

```
if "runsslserver" in sys.argv:
    SSL_CONTEXT = True
    SECURE_HSTS_SECONDS = 3600
else:
    SSL_CONTEXT = False
    SECURE_HSTS_SECONDS = 0
    CSRF_COOKIE_NAME = 'csrftoken'

CSRF_COOKIE_SECURE = SSL_CONTEXT
SECURE_SSL_REDIRECT = SSL_CONTEXT
SESSION_COOKIE_SECURE = SSL_CONTEXT
CSP_UPGRADE_INSECURE_REQUESTS = SSL_CONTEXT
```

Reducing `SECURE_HSTS_SECONDS` time is also a good idea during development.

CHAPTER 4

Default settings

4.1 Django settings

4.1.1 CSRF_COOKIE_NAME

Default: `__Host-csrftoken`

Tip: See the Mozilla Http Observatory recommendations regarding cookies.

4.1.2 CSRF_COOKIE_SAMESITE

Default: `Lax`

Note: This setting is available in Django 2.2 or through the `django-cookie-samesite` package in Django 1.11.

4.1.3 CSRF_COOKIE_SECURE

Default: `True`

Note: Requires an HTTPS connection.

4.1.4 SECURE_BROWSER_XSS_FILTER

Default: `True`

4.1.5 SECURE_CONTENT_TYPE_NOSNIFF

Default: `True`

4.1.6 SECURE_HSTS_INCLUDE_SUBDOMAINS

Default: `True`

Warning: Activating HSTS may irreversibly break your site (for `SECURE_HSTS_SECONDS`) if improperly configured. Review Django's advice regarding HSTS first!

4.1.7 SECURE_HSTS_PRELOAD

Default: `True`

Warning: Activating HSTS may irreversibly break your site (for `SECURE_HSTS_SECONDS`) if improperly configured. Review Django's advice regarding HSTS first!

4.1.8 SECURE_HSTS_SECONDS

Default: 183 * 24 * 60 * 60

Warning: Activating HSTS may irreversibly break your site (for `SECURE_HSTS_SECONDS`) if improperly configured. Review Django's advice regarding HSTS first!

4.1.9 SESSION_COOKIE_HTTPONLY

Default: `True`

Hint: This is already true by default in Django 1.11 and 2.2.

4.1.10 SESSION_COOKIE_SAMESITE

Default: `Lax`

Note: This setting is available in Django 2.2 or through `django-cookie-samesite` package in Django 1.11.

4.1.11 SESSION_COOKIE_SECURE

Default: `True`

Note: Requires an HTTPS connection.

4.2 Django-CSP settings

See the django-csp docs for full details.

4.2.1 CSP_BASE_URI

Default: `["'none'"]`

4.2.2 CSP_DEFAULT_SRC

Default: `["'self'"]`

4.2.3 CSP_FONT_SRC

Default: `["'self'"]`

4.2.4 CSP_FORM_ACTION

Default: `["'self'"]`

4.2.5 CSP_FRAME_ANCESTORS

Default: `["'none'"]`

4.2.6 CSP_FRAME_SRC

Default: `["*"]`

4.2.7 CSP_IMG_SRC

Default: `["*", "data:"]`

4.2.8 CSP_MEDIA_SRC

Default: `["*", "data:"]`

4.2.9 CSP_SCRIPT_SRC

Default: `["'self'"]`

4.2.10 CSP_STYLE_SRC

Default: `["'self'"]`

4.2.11 CSP_INCLUDE_NONCE_IN

Default: `["script-src", "style-src"]`

4.2.12 CSP_UPGRADE_INSECURE_REQUESTS

Default: `True`

4.2.13 CSP_BLOCK_ALL_MIXED_CONTENT

Default: `True`

4.2.14 CSP_REPORT_PERCENTAGE

Default: `0.1`

4.3 Middleware settings

4.3.1 REFERRER_POLICY

Default: `same-origin`

Tip: See the Mozilla Http Observatory recommendations regarding the referrer-policy as well as Scott Helme's discussion.

4.3.2 FEATURE_POLICY

Default:

```
[
    "autoplay 'none'",
    "camera 'none'",
    "display-capture 'none'",
    "document-domain 'none'",
    "encrypted-media 'none'",
    "fullscreen *",
    "geolocation 'none'",
```

(continues on next page)

(continued from previous page)

```
    "microphone 'none'",
    "midi 'none'",
    "payment 'none'",
    "vr *",
]
```

Tip: See Scott Helme's discussion on the new feature policy header.

4.3.3 FRAMING_ALLOWED_FROM

Default: `deny`

Safe domains for `X-FRAME-OPTIONS` can be specified two ways:

1. Through the admin interface, or
2. In `settings.py` by assigning a list to `FRAMING_ALLOWED_FROM`. This list supersedes any database entries: if this list is set, domains entered through admin are ignored. To allow all domains, set `FRAMING_ALLOWED_FROM = ["*"]`

CHAPTER 5

Middleware

security_headers.middleware.**extra_security_headers_middleware**(*get_response*)
Sets security headers specified in SETTINGS on all responses.

CHAPTER 6

Models

class security_headers.models.**FramingAllowedFrom**(**args*, ***kwargs*)

Domains from which framing is allowed.

exception DoesNotExist

exception MultipleObjectsReturned

CHAPTER 7

Contributing

1. Install development requirements

```
pip install -r requirements/dev-requirements.txt
```

2. Install pre-commit hooks

```
pre-commit install
```

3. To run localserver

```
python security_headers.py runserver
```

4. To get http-observatory scan report, start a separate secure localhost (at 127.0.0.1:8000) to enable https and then navigate to the /scan/<name of url> from runserver

```
python security_headers.py runsslserver
```

5. To run test suite

```
pytest
```

6. To test build

```
tox
```

7. To make docs locally

```
cd docs
make html
```

Python Module Index

S

security_headers.middleware, 15
security_headers.models, 17

Index

E

extra_security_headers_middleware() (*in module security_headers.middleware*), 15

F

FramingAllowedFrom (*class in security_headers.models*), 17
FramingAllowedFrom.DoesNotExist, 17
FramingAllowedFrom.MultipleObjectsReturned, 17

S

security_headers.middleware (*module*), 15
security_headers.models (*module*), 17