

---

# **secure.py Documentation**

***Release 0.3.0***

**Caleb Kinney**

**Apr 27, 2021**



---

## Contents:

---

<b>1</b>	<b>Supported Python web frameworks:</b>	<b>3</b>
<b>2</b>	<b>Install</b>	<b>5</b>
<b>3</b>	<b>Documentation</b>	<b>7</b>
3.1	Secure Headers . . . . .	7
3.2	Policy Builder . . . . .	9
3.3	Supported Frameworks . . . . .	13
3.4	Resources . . . . .	20
<b>4</b>	<b>Indices and tables</b>	<b>23</b>



`secure.py` is a lightweight package that adds optional security headers for Python web frameworks.



# CHAPTER 1

---

Supported Python web frameworks:

---

aiohttp, Bottle, CherryPy, Django, Falcon, FastAPI, Flask, hug, Masonite, Pyramid, Quart, Responder, Sanic, Starlette, Tornado





## CHAPTER 2

---

### Install

---

**pip:**

```
$ pip install secure
```

**Pipenv:**

```
$ pipenv install secure
```

After installing secure.py:

```
import secure  
  
secure_headers = secure.Secure()
```



### 3.1 Secure Headers

Security Headers are HTTP response headers that, when set, can enhance the security of your web application by enabling browser security policies.

You can assess the security of your HTTP response headers at [securityheaders.com](https://securityheaders.com)

*Recommendations used by `secure.py` and more information regarding security headers can be found at the [OWASP Secure Headers Project](#).*

#### 3.1.1 Server

Contain information about server software

**Default Value:** NULL (*obfuscate server information, not included by default*)

#### 3.1.2 Strict-Transport-Security (HSTS)

Ensure application communication is sent over HTTPS

**Default Value:** `max-age=63072000; includeSubdomains`

#### 3.1.3 X-Frame-Options (XFO)

Disable framing from different origins (clickjacking defense)

**Default Value:** SAMEORIGIN

### 3.1.4 X-XSS-Protection

Enable browser cross-site scripting filters

**Default Value:** 0

### 3.1.5 X-Content-Type-Options

Prevent MIME-sniffing

**Default Value:** nosniff

### 3.1.6 Content-Security-Policy (CSP)

Prevent cross-site injections

**Default Value:** script-src 'self'; object-src 'self' (*not included by default*)\*

### 3.1.7 Referrer-Policy

Enable full referrer if same origin, remove path for cross origin and disable referrer in unsupported browsers

**Default Value:** no-referrer

### 3.1.8 Cache-control

Prevent cacheable HTTPS response

**Default Value:** no-cache

### 3.1.9 Permissions-Policy

Disable browser features and APIs

**Default Value:** accelerometer=(), ambient-light-sensor=(), autoplay=(),camera=(), encrypted-media=(), fullscreen=(),geolocation=(), gyroscope=(), magnetometer=(),microphone=(); midi=(), payment=(), picture-in-picture=(), speaker=(), sync-xhr=(), usb=(), vr=() (*not included by default*)

#### Additional information:

- The Strict-Transport-Security (HSTS) header will tell the browser to **only** utilize secure HTTPS connections for the domain, and in the default configuration, including all subdomains. The HSTS header requires trusted certificates and users will be unable to connect to the site if using self-signed or expired certificates. The browser will honor the HSTS header for the time directed in the max-age attribute (*default = 2 years*), and setting the max-age to 0 will disable an already set HSTS header. Use the hsts=False option to not include the HSTS header in Secure Headers.
- The Content-Security-Policy (CSP) header can break functionality and can (and should) be carefully constructed, use the csp=True option to enable default values.

### 3.1.10 Usage

`secure_headers.framework(response)`

**Default HTTP response headers:**

```
Strict-Transport-Security: max-age=63072000; includeSubdomains
X-Frame-Options: SAMEORIGIN
X-XSS-Protection: 0
X-Content-Type-Options: nosniff
Referrer-Policy: no-referrer, strict-origin-when-cross-origin
Cache-control: no-cache, no-store, must-revalidate, max-age=0
Pragma: no-cache
Expires: 0
```

### 3.1.11 Options

You can toggle the setting of headers with default values by passing `True` or `False` and override default values by passing a string to the following options:

- `server` - set the Server header, e.g. `Server="Secure"` (*string / bool / SecurePolicies, default=False*)
- `hsts` - set the Strict-Transport-Security header (*string / bool / SecurePolicies, default=True*)
- `xfo` - set the X-Frame-Options header (*string / bool / SecurePolicies, default=True*)
- `xxp` - set the X-XSS-Protection header (*string / bool / SecurePolicies, default=True*)
- `content` - set the X-Content-Type-Options header (*string / bool / SecurePolicies, default=True*)
- `csp` - set the Content-Security-Policy (*string / bool / SecurePolicies, default=False*) \*
- `referrer` - set the Referrer-Policy header (*string / bool / SecurePolicies, default=True*)
- `cache` - set the Cache-control and Pragma headers (*string / bool / SecurePolicies, default=True*)
- `feature` - set the Feature-Policy header (*SecurePolicies / string / bool / SecurePolicies, default=False*)

**Example:**

```
from secure import SecureHeaders

secure_headers = SecureHeaders(csp=True, hsts=False, xfo="DENY")

...

secure_headers.framework(response)
```

## 3.2 Policy Builder

### 3.2.1 ContentSecurityPolicy()

**Directives:**

<code>base_uri(sources),</code>	<code>child_src(sources),</code>	<code>connect_src(sources),</code>
<code>default_src(sources),</code>	<code>font_src(sources),</code>	<code>form_action(sources),</code>
<code>frame_ancestors(sources),</code>	<code>frame_src(sources),</code>	<code>img_src(sources),</code>
<code>manifest_src(sources),</code>	<code>media_src(sources),</code>	<code>object_src(sources),</code>
<code>plugin_types(types),</code>	<code>report_to(json_object),</code>	<code>report_uri(uri),</code>

```
require_sri_for(values), sandbox(values), script_src(sources), style_src(sources),
upgrade_insecure_requests(), worker_src(sources)
```

**Example:**

```
csp_policy = (
    secure.ContentSecurityPolicy()
    .default_src('none')
    .base_uri('self')
    .connect_src('self', 'api.spam.com')
    .frame_src('none')
    .img_src('self', 'static.spam.com')
)

secure_headers = secure.Secure(csp=csp_policy)

# default-src 'none'; base-uri 'self'; connect-src 'self' api.spam.com; frame-src
↪ 'none'; img-src 'self' static.spam.com
```

*You can check the effectiveness of your CSP Policy at the [CSP Evaluator](#)*

### 3.2.2 StrictTransportSecurity()

**Directives:** `include_subDomains()`, `max_age(seconds)`, `preload()`

**Example:**

```
hsts_value = (
    secure.StrictTransportSecurity()
    .include_subdomains()
    .preload()
    .max_age(2592000)
)

secure_headers = secure.Secure(hsts=hsts_value)

# includeSubDomains; preload; max-age=2592000
```

### 3.2.3 XFrameOptions()

**Directives:** `allow_from(uri)`, `deny()`, `sameorigin()`

**Example:**

```
xfo_value = secure.XFrameOptions().deny()

secure_headers = secure.Secure(xfo=xfo_value)

# deny
```

### 3.2.4 ReferrerPolicy()

**Directives:** `no_referrer()`, `no_referrer_when_downgrade()`, `origin()`,  
`origin_when_cross_origin()`, `same_origin()`, `strict_origin()`,  
`strict_origin_when_cross_origin()`, `unsafe_url()`

**Example:**

```
referrer = secure.ReferrerPolicy().strict_origin()

secure_headers = secure.Secure(referrer=referrer).headers()

# strict-origin
```

### 3.2.5 PermissionsPolicy()

**Directives:** accelerometer(allowlist), ambient\_light\_sensor(allowlist), autoplay(allowlist), camera(allowlist), document\_domain(allowlist), encrypted\_media(allowlist), fullscreen(allowlist), geolocation(allowlist), gyroscope(allowlist), magnetometer(allowlist), microphone(allowlist), midi(allowlist), payment(allowlist), picture\_in\_picture(allowlist), speaker(allowlist), sync\_xhr(allowlist), usb(allowlist), Values(allowlist), vr(allowlist)

**Example:**

```
permissions = (
    secure.PermissionsPolicy().geolocation("self", "spam.com").vibrate()
)

secure_headers = secure.Secure(permissions=permissions).headers()

# geolocation=(self "spam.com"), vibrate=()
```

### 3.2.6 CacheControl()

**Directives:** immutable(), max\_age(seconds), max\_stale(seconds), min\_fresh(seconds), must\_revalidate(), no\_cache(), no\_store(), no\_transform(), only\_if\_cached(), private(), proxy\_revalidate(), public(), s\_maxage(seconds), stale\_if\_error(seconds), stale\_while\_revalidate(seconds),

**Example:**

```
cache = secure.CacheControl().no_cache()

secure_headers = secure.Secure(cache=cache).headers()

# no-store
```

### 3.2.7 Usage

**Example:**

```
import uvicorn
from fastapi import FastAPI
import secure

app = FastAPI()
```

(continues on next page)

(continued from previous page)

```

server = secure.Server().set("Secure")

csp = (
    secure.ContentSecurityPolicy()
    .default_src("'none'")
    .base_uri("'self'")
    .connect_src("'self' "api.spam.com")
    .frame_src("'none'")
    .img_src("'self'", "static.spam.com")
)

hsts = secure.StrictTransportSecurity().include_subdomains().preload().max_
    ↪age(2592000)

referrer = secure.ReferrerPolicy().no_referrer()

permissions_value = (
    secure.PermissionsPolicy().geolocation("self", "'spam.com'").vibrate()
)

cache_value = secure.CacheControl().must_revalidate()

secure_headers = secure.Secure(
    server=server,
    csp=csp,
    hsts=hsts,
    referrer=referrer,
    permissions=permissions_value,
    cache=cache_value,
)

@app.middleware("http")
async def set_secure_headers(request, call_next):
    response = await call_next(request)
    secure_headers.framework.fastapi(response)
    return response

@app.get("/")
async def root():
    return {"message": "Secure"}

if __name__ == "__main__":
    uvicorn.run(app, port=8081, host="localhost")

. . .

```

**Response Headers:**

```

server: Secure
strict-transport-security: includeSubDomains; preload; max-age=2592000
x-frame-options: SAMEORIGIN
x-xss-protection: 0
x-content-type-options: nosniff
content-security-policy: default-src 'none'; base-uri 'self'; connect-src 'self'api.
    ↪spam.com; frame-src 'none'; img-src 'self' static.spam.com

```

(continues on next page)



(continued from previous page)

```
referrer-policy: no-referrer
cache-control: must-revalidate
permissions-policy: geolocation=(self 'spam.com'), vibrate=()
```

## 3.3 Supported Frameworks

### 3.3.1 Framework Agnostic

Return Dictionary of Headers:

```
secure_headers.headers()
```

**Example:**

```
secure_headers.framework.headers(csp=True, feature=True)
```

**Return Value:**

```
{'Strict-Transport-Security': 'max-age=63072000; includeSubdomains',
'X-Frame-Options': 'SAMEORIGIN', 'X-XSS-Protection': '0',
'X-Content-Type-Options': 'nosniff', 'Content-Security-Policy':
"script-src 'self'; object-src 'self'", 'Referrer-Policy': 'no-referrer,
strict-origin-when-cross-origin', 'Cache-control': 'no-cache, no-store,
must-revalidate', 'Pragma': 'no-cache', 'Feature-Policy': "accelerometer
'none'; ambient-light-sensor 'none'; autoplay 'none'; camera 'none';
encrypted-media 'none'; fullscreen 'none'; geolocation 'none'; gyroscope
'none'; magnetometer 'none'; microphone 'none'; midi 'none'; payment 'none';
picture-in-picture 'none'; speaker 'none'; sync-xhr 'none'; usb 'none'; vr
'none';"}}
```

### 3.3.2 aiohttp

```
secure_headers.framework.aiohttp(resp)
```

**Example:**

```
from aiohttp import web
from aiohttp.web import middleware
import secure

secure_headers = secure.Secure()

. . .

@middleware
async def set_secure_headers(request, handler):
    resp = await handler(request)
    secure_headers.framework.aiohttp(resp)
    return resp

. . .
```

(continues on next page)

(continued from previous page)

```
app = web.Application(middlewares=[set_secure_headers])  
.  
.  
.
```

### 3.3.3 Bottle

```
secure_headers.framework.bottle(response)
```

**Example:**

```
from bottle import route, run, response, hook  
import secure  
  
secure_headers = secure.Secure()  
.  
.  
.  
  
@hook("after_request")  
def set_secure_headers():  
    secure_headers.framework.bottle(response)  
.  
.  
.
```

### 3.3.4 CherryPy

```
"tools.response_headers.headers": secure_headers.framework.cherrypy()
```

**Example:**

CherryPy Application Configuration:

```
import cherrypy  
import secure  
  
secure_headers = secure.Secure()  
.  
.  
.  
  
config = {  
    "/": {  
        "tools.response_headers.on": True,  
        "tools.response_headers.headers": secure_headers.framework.cherrypy(),  
    }  
}  
.  
.  
.
```

### 3.3.5 Django

```
secure_headers.framework.django(response)
```

**Example:**

## Django Middleware Documentation:

```
# securemiddleware.py
import secure

secure_headers = secure.Secure()

...

def set_secure_headers(get_response):
    def middleware(request):
        response = get_response(request)
        secure_headers.framework.django(response)
        return response

    return middleware

...
```

```
# settings.py

...

MIDDLEWARE = [
    'app.securemiddleware.set_secure_headers'
]

...
```

### 3.3.6 FastAPI

```
secure_headers.framework.falcon(resp)
```

**Example:**

```
from fastapi import FastAPI
import secure

secure_headers = secure.Secure()

...

@app.middleware("http")
async def set_secure_headers(request, call_next):
    response = await call_next(request)
    secure_headers.framework.fastapi(response)
    return response

...
```

### 3.3.7 Falcon

```
secure_headers.framework.falcon(resp)
```

**Example:**

```
import falcon
import secure

secure_headers = secure.Secure()

. . .

class SetSecureHeaders(object):
    def process_request(self, req, resp):
        secure_headers.framework.falcon(resp)

. . .

app = api = falcon.API(middleware=[SetSecureHeaders()])

. . .
```

### 3.3.8 Flask

secure\_headers.framework.flask(response)

#### Example:

```
from flask import Flask, Response
import secure

secure_headers = secure.Secure()

app = Flask(__name__)

. . .

@app.after_request
def set_secure_headers(response):
    secure_headers.framework.flask(response)
    return response

. . .
```

### 3.3.9 hug

secure\_headers.framework.hug(response)

#### Example:

```
import hug
import secure

secure_headers = secure.Secure()

. . .

@hug.response_middleware()
def set_secure_headers(request, response, resource):
    secure_headers.framework.hug(response)
```

(continues on next page)

(continued from previous page)

```
. . .
```

### 3.3.10 Masonite

```
secure_headers.framework.masonite(self.request)
```

#### Example:

Masonite [Middleware](#):

```
# SecureMiddleware.py

from masonite.request import Request

import secure

secure_headers = secure.Secure()

class SecureMiddleware:
    def __init__(self, request: Request):

        self.request = request

    def before(self):
        secure_headers.framework.masonite(self.request)

    . . .
```

```
# middleware.py

...

HTTP_MIDDLEWARE = [
    SecureMiddleware,
]

...
```

### 3.3.11 Pyramid

Pyramid [Tween](#):

```
def set_secure_headers(handler, registry):
    def tween(request):
        response = handler(request)
        secure_headers.framework.pyramid(response)
        return response

    return tween
```

#### Example:

```
from pyramid.config import Configurator
from pyramid.response import Response
import secure

secure_headers = secure.Secure()

...

def set_secure_headers(handler, registry):
    def tween(request):
        response = handler(request)
        secure_headers.framework.pyramid(response)
        return response

    return tween

...

config.add_tween(".set_secure_headers")

...
```

### 3.3.12 Quart

secure\_headers.framework.quart(response)

**Example:**

```
from quart import Quart, Response
import secure

secure_headers = secure.Secure()

app = Quart(__name__)

...

@app.after_request
async def set_secure_headers(response):
    secure_headers.framework.quart(response)
    return response

...
```

### 3.3.13 Responder

secure\_headers.framework.responder(resp)

**Example:**

```
import responder
import secure

secure_headers = secure.Secure()
```

(continues on next page)

(continued from previous page)

```

api = responder.API()

. . .

@api.route(before_request=True)
def set_secure_headers(req, resp):
    secure_headers.framework.responder(resp)

. . .

```

You should use Responder's [built in HSTS](#) and pass the `hsts=False` option.

### 3.3.14 Sanic

```
secure_headers.framework.sanic(response)
```

**Example:**

```

from sanic import Sanic
import secure

secure_headers = secure.Secure()

app = Sanic()

. . .

@app.middleware("response")
async def set_secure_headers(request, response):
    secure_headers.framework.sanic(response)

. . .

```

To set Cross Origin Resource Sharing (CORS) headers, please see [sanic-cors](#).

### 3.3.15 Starlette

```
secure_headers.framework.starlette(response)
```

**Example:**

```

from starlette.applications import Starlette
import uvicorn
import secure

secure_headers = secure.Secure()

app = Starlette()

. . .

@app.middleware("http")
async def set_secure_headers(request, call_next):
    response = await call_next(request)
    secure_headers.framework.starlette(response)

```

(continues on next page)

(continued from previous page)

```
    return response  
    . . .
```

### 3.3.16 Tornado

```
secure_headers.framework.tornado(self)
```

**Example:**

```
import tornado.ioloop  
import tornado.web  
import secure  
  
secure_headers = secure.Secure()  
  
    . . .  
  
class BaseHandler(tornado.web.RequestHandler):  
    def set_default_headers(self):  
        secure_headers.framework.tornado(self)  
  
    . . .
```

## 3.4 Resources

### 3.4.1 Frameworks

- [aiohttp](#) - Asynchronous HTTP client/server framework for asyncio and Python
- [Bottle](#) - A fast and simple micro-framework for python web-applications.
- [CherryPy](#) - A pythonic, object-oriented HTTP framework.
- [Django](#) - The Web framework for perfectionists with deadlines.
- [Falcon](#) - A bare-metal Python web API framework for building high-performance microservices, app backends, and higher-level frameworks.
- [Flask](#) - The Python micro framework for building web applications.
- [hug](#) - Embrace the APIs of the future. Hug aims to make developing APIs as simple as possible, but no simpler.
- [Masonite](#) - The Modern And Developer Centric Python Web Framework.
- [Pyramid](#) - A Python web framework
- [Quart](#) - A Python ASGI web microframework.
- [Responder](#) - A familiar HTTP Service Framework
- [Sanic](#) - An Async Python 3.5+ web server that's written to go fast
- [Starlette](#) - The little ASGI framework that shines.
- [Tornado](#) - A Python web framework and asynchronous networking library, originally developed at FriendFeed.



### 3.4.2 General

- [OWASP - Secure Headers Project](#)
- [OWASP - Session Management Cheat Sheet](#)
- [Mozilla Web Security](#)
- [securityheaders.com](#)

### 3.4.3 Policies

- **CSP:** [CSP Cheat Sheet | Scott Helme](#), [Content-Security-Policy | MDN](#), [Content Security Policy Cheat Sheet | OWASP](#), [Content Security Policy CSP Reference & Examples](#)
- **XXP:** [X-XSS-Protection | MDN](#)
- **XFO:** [X-Frame-Options | MDN](#)
- **HSTS:** [Strict-Transport-Security | MDN](#), [HTTP Strict Transport Security Cheat Sheet | OWASP](#)
- **Referrer:** [A new security header: Referrer Policy | Scott Helme](#), [Referrer-Policy | MDN](#)
- **Feature:** [A new security header: Feature Policy | Scott Helme](#), [Feature-Policy | MDN](#), [Introduction to Feature Policy | Google Developers](#)
- **Cache:** [Cache-Control | MDN](#)



## CHAPTER 4

---

### Indices and tables

---

- `genindex`
- `modindex`
- `search`