django-session-security Documentation

Release 2.5.1

James Pic

Oct 27, 2017

Contents

This app provides a mechanism to logout inactive authenticated users. An inactive browser should be logged out automatically if the user left his workstation, to protect sensitive data that may be displayed in the browser. It may be useful for CRMs, intranets, and such projects.

For example, if the user leaves for a coffee break, this app can force logout after say 5 minutes of inactivity.

CHAPTER 1

Why not just set the session to expire after X minutes ?

Or “Why does this app even exist” ? Here are the reasons:

- if the user session expires before the user is done reading a page: he will have to login again.
- if the user session expires before the user is done filling a form: his work will be lost, and he will have to login again, and probably yell at you, dear django dev ... at least I know I would !

This app allows to short circuit those limitations in session expiry.

CHAPTER 2

How does it work ?

When the user loads a page, SessionSecurity middleware will set the last activity to now. The last activity is stored as datetime in `request.session['_session_security']`. To avoid having the middleware update that last activity datetime for a URL, add the url to `settings.SESSION_SECURITY_PASSIVE_URLS`.

When the user moves mouse, click, scroll or press a key, SessionSecurity will save the DateTime as a JavaScript attribute. It will send the number of seconds since when the last user activity was recorded to PingView, next time it should ping.

First, a warning should be shown after `settings.SESSION_SECURITY_WARN_AFTER` seconds. The warning displays a text like "Your session is about to expire, move the mouse to extend it".

Before displaying this warning, SessionSecurity will upload the time since the last client-side activity was recorded. The middleware will take it if it is shorter than what it already has - ie. another more recent activity was detected in another browser tab. The PingView will respond with the number of seconds since the last activity - all browser tab included.

If there was no other, more recent, activity recorded by the server: it will show the warning. Otherwise it will update the last activity in javascript from the PingView response.

Same goes to expire after `settings.SESSION_SECURITY_EXPIRE_AFTER` seconds. Javascript will first make an ajax request to PingView to ensure that another more recent activity was not detected anywhere else - in any other browser tab.

CHAPTER 3

Requirements

- Python 2.7 or 3.5+
- jQuery 1.7+
- Django 1.8 to 2.0
- django.contrib.staticfiles or #YoYo

CHAPTER 4

Resources

You could subscribe to the mailing list ask questions or just be informed of package updates.

- Git graciously hosted by GitHub,
- Documentation graciously hosted by RTFD,
- Package graciously hosted by PyPi,
- Mailing list graciously hosted by Google
- For **Security** issues, please contact yourlabs-security@googlegroups.com
- Continuous integration graciously hosted by Travis-ci

Contents:

Quick setup

The purpose of this documentation is to get you started as fast as possible, because your time matters and you probably have other things to worry about.

Install the package:

```
pip install django-session-security
# or the development version
pip install -e git+git://github.com/yourlabs/django-session-security.git#egg=django-
↪session-security
```

For static file service, add to `settings.INSTALLED_APPS`:

```
'session_security',
```

Add to `settings.MIDDLEWARE_CLASSES`, **after** django's AuthenticationMiddleware:

```
'session_security.middleware.SessionSecurityMiddleware',
```

Ensure settings.TEMPLATE_CONTEXT_PROCESSORS has:

```
'django.core.context_processors.request'
```

Add to urls:

```
url(r'session_security/', include('session_security.urls')),
```

At this point, we're going to assume that you have django.contrib.staticfiles working. This means that static files are automatically served with runserver, and that you have to run collectstatic when using another server (fastcgi, uwsgi, and whatnot). If you don't use django.contrib.staticfiles, then you're on your own to manage staticfiles.

After jQuery, add to your base template:

```
{% include 'session_security/all.html' %}
```

Full documentation

Settings

Settings for django-session-security.

WARN_AFTER Time (in seconds) before the user should be warned that is session will expire because of inactivity. Default 540. Overridable in `settings.SESSION_SECURITY_WARN_AFTER`.

EXPIRE_AFTER Time (in seconds) before the user should be logged out if inactive. Default is 600. Overridable in `settings.SESSION_SECURITY_EXPIRE_AFTER`.

PASSIVE_URLS List of urls that should be ignored by the middleware. For example the ping ajax request of session_security is made without user intervention, as such it should not be used to update the user's last activity datetime. Overridable in `settings.SESSION_SECURITY_PASSIVE_URLS`.

PASSIVE_URL_NAMES Same as PASSIVE_URLS, but takes Django URL names instead of a path. This is useful in case path names change, or contain parameterized values, and thus cannot be described statically. NOTE: currently namespaces are not handled. Overridable in `settings.SESSION_SECURITY_PASSIVE_URL_NAMES`.

SESSION_SECURITY_INSECURE Set this to True in your settings if you want the project to run without having to set SESSION_EXPIRE_AT_BROWSER_CLOSE=True, which you should because it makes no sense to use this app with `SESSION_EXPIRE_AT_BROWSER_CLOSE` to False.

Middleware

SessionSecurityMiddleware is the heart of the security that this application attemps to provide.

To install this middleware, add to your `settings.MIDDLEWARE_CLASSES`:

```
'session_security.middleware.SessionSecurityMiddleware'
```

Make sure that it is placed **after** authentication middlewares.

class `session_security.middleware.`**`SessionSecurityMiddleware`**(*get_response=None*)
In charge of maintaining the real 'last activity' time, and log out the user if appropriate.

get_expire_seconds(*request*)
Return time (in seconds) before the user should be logged out.

is_passive_request(*request*)
Should we skip activity update on this URL/View.

process_request(*request*)
Update last activity time or logout.

update_last_activity(*request*, *now*)
If `request.GET['idleFor']` is set, check if it refers to a more recent activity than `request.session['_session_security']` and update it in this case.

Utils

Helpers to support json encoding of session data

`session_security.utils.`**get_last_activity**(*session*)
Get the last activity datetime string from the session and return the python datetime object.

`session_security.utils.`**set_last_activity**(*session*, *dt*)
Set the last activity datetime as a string in the session.

Urls

One url meant to be used by JavaScript.

session_security_ping Connects the PingView.

To install this url, include it in `urlpatterns` definition in `urls.py`, ie:

```
urlpatterns = patterns('',
    # ....
    url(r'session_security/', include('session_security.urls')),
    # ....
)
```

Views

One view method for AJAX requests by SessionSecurity objects.

class `session_security.views.`**PingView**(***kwargs*)
This view is just in charge of returning the number of seconds since the 'real last activity' that is maintained in the session by the middleware.

Constructor. Called in the URLconf; can contain helpful extra keyword arguments, and other things.

Templates

session_security/dialog.html

```
{% load i18n %}

<div id="session_security_warning" class="session_security" style="display:none" aria-
↪hidden="true" role="dialog">
```

```
    <div class="session_security_overlay"></div>
    <div class="session_security_modal" role="document" tabindex="-1">
        <h3>{% trans 'Your session is about to expire' %}</h3>
        <p>{% trans 'Click or type to extend your session.' %}</p>
    </div>
</div>
```

session_security/all.html

```
{% comment %}
This demonstrates how to setup session security client side stuff on your own.
It provides sensible defaults so you could start with just::

    {% include 'session_security/all.html' %}

{% endcomment %}

{% load session_security_tags %}
{% load i18n l10n %}
{% load static from staticfiles %}

{# If the user is not authenticated then there is no session to secure ! #}
{% if request.user.is_authenticated %}

    {# The modal dialog stylesheet, it's pretty light so it should be easy to hack #}
    <link rel="stylesheet" type="text/css" href="{% static 'session_security/style.css
↪' %}">

    {# Include the template that actually contains the modal dialog #}
    {% include 'session_security/dialog.html' %}

    {# Load SessionSecurity javascript 'class', jquery should be loaded - by you - at␣
↪this point #}
    <script type="text/javascript" src="{% static 'session_security/script.js' %}"></
↪script>

    {# Bootstrap a SessionSecurity instance as the sessionSecurity global variable #}
    {% localize off %}
        <script type="text/javascript">
            var sessionSecurity = new yourlabs.SessionSecurity({
                pingUrl: '{% url 'session_security_ping' %}',
                warnAfter: {{ request|warn_after|unlocalize }},
                expireAfter: {{ request|expire_after|unlocalize }},
                confirmFormDiscard: "{% trans 'You have unsaved changes in a form of␣
↪this page.' %}"
            });
        </script>
    {% endlocalize %}
{% endif %}
```

Static files

session_security/script.js

Read the script documentation

session_security/style.css

```
/* credit: http://www.csslab.cl/2008/01/30/ventana-modal-solo-con-css/ */
.session_security_overlay {
    position: fixed;
    top: 0;
    left: 0;
    width: 100%;
    height: 100%;
    background: #000;
    z-index:1001;
    opacity:.75;
    -moz-opacity: 0.75;
    filter: alpha(opacity=75);
}

.session_security_modal {
    position: fixed;
    top: 25%;
    left: 25%;
    width: 50%;
    padding: 16px;
    background: #fff;
    color: #333;
    z-index:1002;
    overflow: auto;
    text-align: center;
}
```

How to trigger activity programaticaly in JS ?

Call *sessionSecurity.activity()* <*http://django-session-security.readthedocs.org/en/stable-2.x.x/_static/script.html#section-11*> every time you want to programaticaly trigger an activity.

How to disable the "Are you sure you want to leave this page?" warning ?

Include the JavaScript variable `sessionSecurity.confirmFormDiscard = undefined;` somewhere in your project *after* the plugin's JS. For example:

```
<!-- base.html -->
...
{% include 'session_security/all.html' %}
<script>
    sessionSecurity.confirmFormDiscard = undefined;
</script>
...
```

CHAPTER 5

Indices and tables

- genindex
- modindex
- search

Python Module Index

S

Index

G

I

P

S

U