django-agent-trust Documentation

Release 0.2.1

Peter Sagerson

Jul 19, 2017

Contents

This project has tools for managing trusted user agents. For example, you might allow the user to indicate whether they are using a public or private computer and implement different policies for each. Or you might be using a two-factor authentication scheme, allowing the users to bypass the second factor on machines that they designate as trusted. This uses Django 1.4's signed cookie facility and operates independently of sessions.

Short list of features:

- `request.agent.is_trusted` tells you whether the request came from a trusted agent.
- APIs to trust or revoke the agent that made a given request.
- Global, per-user, and per-agent settings can set the duration of agent trust as well as an inactivity timeout.
- Supports session-scoped agent trust for consistency of authorization policies.
- Revoke all of a user's previously trusted agents at any time.

The mechanisms by which a user is allowed to designate trusted agents is left entirely to clients of this library. For an application of this API using one-time passwords, see django-otp-agents, part of the django-otp suite.

CHAPTER 1

Installation

django-agent-trust supports Python 2.6, 2.7, and 3.3+. It requires at least Django 1.4 with `django.contrib.auth`.

1. Add `'django_agent_trust'` to `INSTALLED_APPS`.
2. Add `'django_agent_trust.middleware.AgentMiddleware'` to `MIDDLEWARE_CLASSES`. It must come after `AuthenticationMiddleware`.
3. If you want to access agent information in templates, add `'django_agent_trust.context_processors.agent'` to `TEMPLATE_CONTEXT_PROCESSORS`.

Warning: Version 0.2.0 added Django migrations and 0.2.1 added the corresponding South migrations. If you're upgrading from a previous version, you may need to `migrate --fake-initial django_agent_trust` (Django 1.8+) or `migrate --fake django_agent_trust 0001` (South).

CHAPTER 2

Assessing Trust

A view can determine whether it's being requested from a trusted agent by checking `request.agent.is_trusted`. Agent trust is tied to authenticated users; each user gets a different cookie, so multiple users can maintain separate trust settings on your site using the same machine/browser. Anonymous users always have untrusted agents.

`AgentMiddleware` installs an object on requests that will tell you whether the requesting user agent has been marked trusted and at what time:

You may optionally install an included context processor to propagate these objects to template contexts:

`django_agent_trust.context_processors.`**`agent`**(*request*)

A context processor that sets the template context variable `agent` to the value of `request.agent`.

CHAPTER 3

Managing Trust

Agent trust may be persistent or scoped to a session. Of course, the point of the library is the former, but the latter is included to enable more consistent authorization polices. For example, if you ask the user whether they are on a public or shared device, you might set session-scoped trust for public agents and persistent trust for private agents. Your authorization policy can then refer solely to agent trust: for public agents, this will be synonymous with authentication; for private agents, trust will persist across login sessions. Persistent trust is typically implemented with two-factor authentication, where the second factor is used to establish the trusted agent.

You can update the status of the current agent with the following APIs:

django_agent_trust.**trust_agent**(*request*, *trust_days=None*)

Mark the requesting agent as trusted for the currently logged-in user. This does nothing for anonymous users.

Parameters

- **request** (HttpRequest) – The current request.
- **trust_days** (*float*) – The number of days to trust this agent. None for no agent-specific limit.

django_agent_trust.**trust_session**(*request*)

Mark the requesting agent as trusted in the context of the current session; when the session ends, the agent's trust will be revoked. This replaces any agent trust that already exists. All expiration settings and future revocations still apply. This does nothing for anonymous users.

Parameters **request** (HttpRequest) – The current request.

django_agent_trust.**revoke_agent**(*request*)

Revoke trust in the requesting agent for the currently logged-in user.

Parameters **request** (HttpRequest) – The current request.

django_agent_trust.**revoke_other_agents**(*request*)

Revoke trust in all of the logged-in user's agents other than the current one. This does nothing for anonymous users.

Parameters **request** (HttpRequest) – The current request.

CHAPTER 4

Limiting Access

django_agent_trust.decorators.**trusted_agent_required**(*view=None*, *redirect_field_name='next'*, *login_url=None*)

Similar to `login_required()`, but requires `request.agent.is_trusted` to be true. This will frequently be used in conjunction with login_required, unless you're allowing trusted agents to bypass authentication.

The default value for `login_url` is *AGENT_LOGIN_URL*.

CHAPTER 5

Expiration

django-agent-trust supports two types of trust expiration: simple expiration based on the original trust date and expiration from inactivity. Simple expiration can be managed on three levels: a global setting, a per-user setting, and a setting on the agent itself. Inactivity timeouts can be managed globally and per-user. If any time expirations are specified at multiple levels, the most restrictive takes precedence. All expiration settings are measured in days, although fractional days are permitted.

Global configuration takes the form of two settings: *AGENT_TRUST_DAYS* and *AGENT_INACTIVITY_DAYS*. Per-user configuration is done through a model object:

A custom duration can be set on an individual agent at the time that it is trusted by *trust_agent()*.

CHAPTER 6

Settings

AGENT_COOKIE_DOMAIN

Default: `None`

The domain to use for agent cookies or `None` to use a standard domain. **AGENT_COOKIE_HTTPONLY**

Default: `True`

Whether to use HTTPOnly flag on agent cookies. If this is set to True, client-side JavaScript will not to be able to access the session cookie in many browsers. **AGENT_COOKIE_NAME**

Default: `'agent-trust'`

A prefix for agent cookies. This can be anything. **AGENT_COOKIE_PATH**

Default: `'/'`

The path set on the agent cookies. This should either match the URL path of your Django installation or be a parent of that path. **AGENT_COOKIE_SECURE**

Default: `False`

Whether to use a secure cookie for the agent cookies. If this is set to `True`, the cookie will be marked as "secure," which means browsers may ensure that the cookie is only sent under an HTTPS connection. **AGENT_LOGIN_URL**

Default: `LOGIN_URL`

The URL where requests are redirected for login when using the *`trusted_agent_required()`* decorator. **AGENT_TRUST_DAYS**

Default: `None`

The number of days an agent will remain trusted. This can be `None` to impose no limit. **AGENT_INACTIVITY_DAYS**

Default: `365`

The number of days allowed between requests before an agent's trust is revoked. This can not be `None` (all cookies expire eventually), but you can always set it to a very large number.

CHAPTER 7

Changes

changes

CHAPTER 8

License

Python Module Index

d

Index