django-mama-cas Documentation

Release 2.3.0

Jason Bittel

Nov 20, 2017

Contents

MamaCAS is a Django Central Authentication Service (CAS) single sign-on and single logout server. It implements the CAS 1.0, 2.0 and 3.0 protocols, including some of the optional features.

CAS is a single sign-on and single logout web protocol that allows a user to access multiple applications after providing their credentials a single time. It utilizes security tickets, unique text strings generated and validated by the server, allowing applications to authenticate a user without direct access to the user's credentials (typically a user ID and password).

The source code can be found at github.com/jbittel/django-mama-cas, and is the preferred location for contributions, suggestions and bug reports.

CHAPTER 1

Contents

1.1 Installation

1.1.1 Prerequisites

The primary prerequisite of MamaCAS is Django itself. MamaCAS is tested with and supports all supported versions of Django. Other versions of Django may work, but are not officially tested or supported. See the Django downloads page for download and installation details.

If you're installing MamaCAS manually, such as from the GitHub repository, you'll also need to install the Requests library. The optional gevent module may be installed to enable asynchronous single logout requests. The optional defusedxml module may be installed to enable the /samlValidate endpoint.

1.1.2 Installing

Installing the latest release is easiest with pip:

```
$ pip install django-mama-cas
```

To manually install the latest release, download it from PyPI and install with:

```
$ python setup.py install
```

If you need the latest development code, clone the active development repository on GitHub:

```
$ git clone git://github.com/jbittel/django-mama-cas.git
```

1.1.3 Configuring

Add MamaCAS to the `INSTALLED_APPS` setting within your project's `settings.py` (or equivalent) file:

```
INSTALLED_APPS = (
    # ...existing apps...
    'mama_cas',
)
```

Once added, run `migrate` to create the required database tables.

URL paths

Include the required CAS URL endpoints in your project's root `URLconf` with the following:

```
urlpatterns = [
    # ...existing urls...
    url(r'', include('mama_cas.urls')),
]
```

This makes the CAS server available at the top level of your project's URL (e.g. `http://example.com/login`). To add a path to the CAS root (e.g. `http://example.com/cas/login`) include the path in the URL regular expression:

```
urlpatterns = [
    # ...existing urls...
    url(r'^cas/', include('mama_cas.urls')),
]
```

1.1.4 Authenticating

One or more authentication backends must be installed and configured based on your authoritative authentication sources. MamaCAS does not perform authentication itself, but relies on the active authentication backends. The process of installing and configuring authentication backends will change depending on the individual backend.

See also:

- Django user authentication documentation
- Authentication packages for Django

1.2 Settings

None of these settings are required and have sane defaults, but may be used to customize behavior and improve security.

`django.conf.settings.`**`MAMA_CAS_ALLOW_AUTH_WARN`**

Default `False`

If set, allows the user to control transparency of the single sign-on process. When enabled, an additional checkbox will be displayed on the login form.

`django.conf.settings.`**`MAMA_CAS_ATTRIBUTE_CALLBACKS`**

Default `()`

A tuple of dotted paths to callables that each provide a dictionary of name and attribute values. These values are merged together and included with a service or proxy validation success. Each callable is provided the authenticated `User` and the service URL as arguments. For example:

```
# In settings.py
MAMA_CAS_ATTRIBUTE_CALLBACKS = ('path.to.custom_attributes',)

# In a convenient location
def custom_attributes(user, service):
    return {'givenName': user.first_name, 'email': user.email}
```

Two callbacks are provided to cover basic use cases and serve as examples for custom callbacks:

mama_cas.callbacks.user_name_attributes Returns name-related fields using get_username(), get_full_name() and get_short_name().

mama_cas.callbacks.user_model_attributes Returns all fields on the user object, except for `id` and `password`.

Warning: This setting has been deprecated in favor of per-service configuration with MAMA_CAS_SERVICES.

django.conf.settings.**MAMA_CAS_ENABLE_SINGLE_SIGN_OUT**

Default `False`

If set, causes single logout requests to be sent to all accessed services when a user logs out. It is up to each service to handle these requests and terminate the session appropriately.

Note: By default, the single logout requests are sent synchronously. If requests-futures is installed, they are sent asynchronously.

Warning: This setting has been deprecated in favor of per-service configuration with MAMA_CAS_SERVICES.

django.conf.settings.**MAMA_CAS_FOLLOW_LOGOUT_URL**

Default `True`

Controls the client redirection behavior at logout when the `service` parameter is provided. When this setting is `True` and the parameter is present, the client will be redirected to the specified URL. When this setting is `False` or the parameter is not provided, the client is redirected to the login page.

django.conf.settings.**MAMA_CAS_SERVICE_BACKENDS**

Default `['mama_cas.services.backends.SettingsBackend']`

A list of paths to service backends.

django.conf.settings.**MAMA_CAS_SERVICES**

Default `[]`

A list containing all allowed services for the server. Each list item is a dictionary containing the configuration for each service. For example:

```
MAMA_CAS_SERVICES = [
    {
        'SERVICE': '^https://[^\.]+\.example\.com',
        'CALLBACKS': [
```

```
            'mama_cas.callbacks.user_name_attributes',
        ],
        'LOGOUT_ALLOW': True,
        'LOGOUT_URL': 'https://www.example.com/logout',
        'PROXY_ALLOW': True,
        'PROXY_PATTERN': '^https://proxy\.example\.com',
    }
]
```

The following configuration options are available for each service:

SERVICE

A Python regular expression that is tested against to match a given service identifier. This option is required.

CALLBACKS

A list of dotted paths to callables that each provide a dictionary of name and attribute values. These values are merged together and included with a service or proxy validation success. Each callable is provided the authenticated `User` and the service URL as arguments. Defaults to `[]`.

Two callbacks are provided to cover basic use cases and serve as examples for custom callbacks:

`mama_cas.callbacks.user_name_attributes` Returns name-related fields using get_username(), get_full_name() and get_short_name().

`mama_cas.callbacks.user_model_attributes` Returns all fields on the user object, except for `id` and `password`.

LOGOUT_ALLOW

A boolean setting to determine whether single log-out requests are sent for this service. Defaults to `False`.

LOGOUT_URL

A URL that will be used for a single log-out request for the service. If not specified, the service URL will be used instead. Defaults to `''`.

PROXY_ALLOW

A boolean setting to determine whether proxy requests are allowed for this service. Defaults to `True`.

PROXY_PATTERN

A Python regular expression that is tested against to determine if the provided pgtUrl is allowed to make proxy requests. Defaults to `''`.

`django.conf.settings.`**`MAMA_CAS_TICKET_EXPIRE`**

Default `90`

Controls the length of time, in seconds, between when a service or proxy ticket is generated and when it expires. If the ticket is not validated before this time has elapsed, it becomes invalid. This does **not** affect proxy-granting ticket expiration or the duration of a user's single sign-on session.

`django.conf.settings.`**`MAMA_CAS_TICKET_RAND_LEN`**

Default `32`

Sets the number of random characters created as part of the ticket string. It should be long enough that the ticket string cannot be brute forced within a reasonable amount of time. Longer values are more secure, but could cause compatibility problems with some clients.

`django.conf.settings.`**`MAMA_CAS_VALID_SERVICES`**

Default `()`

A list of valid Python regular expressions that a service URL is tested against when a ticket is validated or the client is redirected. If none of the regular expressions match the provided URL, the action fails. If no valid services are configured, any service URL is allowed. For example:

```
MAMA_CAS_VALID_SERVICES = (
    '^https?://www\.example\.edu/secure',
    '^https://[^\.]+\.example\.com',
)
```

Warning: This setting has been deprecated in favor of MAMA_CAS_SERVICES.

`django.conf.settings.`**`MAMA_CAS_LOGIN_TEMPLATE`**

Default `'mama_cas/login.html'`

A path to the login template to use. Make sure Django can find this template using normal Django template discovery rules.

`django.conf.settings.`**`MAMA_CAS_WARN_TEMPLATE`**

Default `'mama_cas/warn.html'`

A path to the warning template to use. Make sure Django can find this template using normal Django template discovery rules.

1.3 Security

This is a high level overview of recommended configuration options and some security best practices. Properly securing a CAS server means understanding your specific security requirements and any unique aspects of your setup. This is not intended to be a comprehensive security guide. It is important to understand each component of your specific stack and ensure it is configured properly.

1.3.1 MamaCAS Configuration

Open vs. Closed

By default, MamaCAS operates in an "open" mode that authenticates or redirects any service URL. It is recommended that a production server be configured as "closed" by specifying approved services with `MAMA_CAS_VALID_SERVICES`. Services not matching one of these patterns will be unable to validate tickets or redirect clients.

1.3.2 Django Configuration

Sessions

MamaCAS relies on standard Django sessions to govern single sign-on sessions. In particular, there are two Django session settings that should be considered:

SESSION_COOKIE_AGE It is recommended this be set shorter than the default of two weeks. This setting controls the duration of single sign-on sessions as well as the duration of proxy-granting tickets.

SESSION_EXPIRE_AT_BROWSER_CLOSE This should be set to `True` to conform to the CAS specification. Note that some browsers can be configured to retain cookies across browser restarts, even cookies set to be removed on browser close.

Additional session settings may need to be configured. For more information, see the Django session documentation.

Best Practices

The Django documentation includes some great security best practices that are useful to review. Some of them do not apply to a dedicated CAS server, but many are both applicable and recommended.

1.3.3 Web Server

Securing a web server is a vast topic completely outside the scope of this guide, and many details depend on the specific server in use. Here are some broadly applicable considerations.

SSL

Obviously, a login server should require SSL. Without it, login credentials and CAS tickets are exposed to anyone with access to the network traffic. Additionally, all services utilizing CAS should communicate with the server via SSL.

HTTP Strict Transport Security

HTTP Strict Transport Security (HSTS) headers tell browsers that the site should only be accessed via HTTPS and not HTTP. When a browser encounters this header, it will automatically use HTTPS for future visits. This prevents some man-in-the-middle attacks caused by browsers initially accessing the page via HTTP, even if they are subsequently redirected.

X-Frame-Options

The X-Frame-Options header indicates whether a page may appear inside a `<frame>`, `<iframe>` or `<object>` element to mitigate clickjacking attacks. If the site should legitimately appear within one of these elements, valid domains may be whitelisted.

1.4 Templates

MamaCAS includes templates implementing standard username and password authentication. Depending on your needs, you can use them as-is, customize portions or replace them entirely.

mama_cas/login.html

Displays the authentication form whenever `LoginView` requires user credentials, as well as authentication success or failure information. When the user logs out, they are redirected to this template with a logout success message if `MAMA_CAS_FOLLOW_LOGOUT_URL` is `False` or no URL is provided.

mama_cas/warn.html

Used by `LoginView` when `MAMA_CAS_ALLOW_AUTH_WARN` is enabled and the user has elected to be notified when authentication occurs. It provides options for the user to continue the authentication process or cancel and log out.

1.4.1 Modifying

To override or extend blocks in the stock templates, include custom templates in the search order by specifying the location with the `DIRS` option to the `TEMPLATES` setting.

The base level stock templates are wrappers to simplify extending the stock templates without circular template inheritance issues. The base template `mama_cas/login.html` has a corresponding `mama_cas/__login.html` and `mama_cas/warn.html` has a corresponding `mama_cas/__warn.html`.

For example, to add a header above the login form with some additional styling create a file named `mama_cas/login.html` that extends `mama_cas/__login.html`:

```
{% extends "mama_cas/__login.html" %}

{% block styles %}
  {{ block.super }}
  <style>.login-title { color: #aaa; font-size: 2em; }</style>
{% endblock styles %}

{% block content_title %}
  If You Can Believe Your Eyes and Ears
{% endblock content_title %}
```

1.4.2 Replacing

If the required changes are substantial, then replace the stock templates entirely. Following the example above, remove the top line that extends the stock template and include the remainder of the page contents. In addition to the login form, some elements custom templates should include are:

Messages The `messages` framework displays information for login, logout or authentication events.

Non-field errors The login form's `non_field_errors` display information regarding authentication failures and other login problems.

1.5 Management Commands

MamaCAS includes custom management commands to aid in some common tasks. You can see which management commands are available by running:

```
$ manage.py
```

The commands specific to MamaCAS will show up underneath the `[mama_cas]` heading. To run a given command:

```
$ manage.py <command name>
```

1.5.1 Commands

checkservice <service> [<pgtUrl>]

Checks the validity and configuration of a given service identifier and optional pgtUrl. For example:

```
$ manage.py checkservice https://www.example.org
Invalid Service: https://www.example.org
```

```
$ manage.py checkservice https://www.example.com
Valid Service: https://www.example.com
Proxy Allowed: False
Logout Allowed: False
Logout URL: None
Callbacks: ['mama_cas.callbacks.user_name_attributes']

$ manage.py checkservice https://www.example.com https://proxy.example.com
Valid Service: https://www.example.com
Proxy Allowed: True
Proxy Callback Allowed: True
Logout Allowed: False
Logout URL: None
Callbacks: ['mama_cas.callbacks.user_name_attributes']
```

cleanupcas

Tickets created by MamaCAS are not removed from the database at the moment of invalidation. Running this command will delete all invalid tickets from the database. Tickets are invalidated either when they expire a configurable number of seconds after creation or by being consumed. Either situation means the ticket is no longer valid for future authentication attempts and can be safely deleted.

It is recommended that this command be run on a regular basis so invalid tickets do not become a performance or storage concern.

1.6 Forms

MamaCAS includes a form class implementing standard username and password authentication. In most cases, this will be the form of authentication required. Trust authentication can be used with CAS, but the requirements will be highly implementation dependent.

1.6.1 Authentication Forms

class `mama_cas.forms.`**`LoginForm`**

This is the base form for handling standard username and password authentication credentials. It contains the following fields:

`username` The username of the client requesting authentication. This field is required.

`password` The password of the client requesting authentication. This field is required.

`warn` A checkbox for configuring transparency of the single sign-on process. If checked, the user will be notified before being authenticated to subsequent services. This field will only be displayed if `MAMA_CAS_ALLOW_AUTH_WARN` is enabled.

The form's `clean()` method attempts authentication against the configured authentication backends and verifies the user account is active. If authentication fails, a `FormValidation` exception is raised with an appropriate error message.

1.6.2 Additional Forms

The following form classes inherit from `LoginForm`, providing additional or alternate behavior during the login process.

class mama_cas.forms.**LoginFormEmail**
A subclass of *LoginForm* which performs additional cleanup on the username field. If an email address is provided for the username, only the username portion of the string is used for authentication.

1.7 CAS Protocol

The official CAS protocol specification can be found at http://jasig.github.io/cas/. Where appropriate, docstrings and other documentation include numbers in parenthesis (e.g. (2.3)) corresponding to the section number within the CAS protocol documentation where that functionality is described. Additionally, views are labeled with a CAS version number in brackets (e.g. [CAS 2.0]) corresponding to the CAS version that defines that particular URI.

CAS 1.0 is a plain text protocol that returns a simple "yes" or "no" response indicating a ticket validation success or failure. CAS 2.0 returns XML fragments for validation responses and allows for proxy authentication. CAS 3.0 expands the protocol with additional request parameters and a SAML response endpoint.

See also:

- CAS Protocol
- CAS User Manual
- CAS 1 Architecture
- CAS 2 Architecture
- Proxy Authentication

1.7.1 Protocol Deviations

In some areas MamaCAS deviates from the official CAS specification to take advantage of built-in Django functionality. These changes do not alter the contract between the client, service and CAS server.

Login ticket (3.5) This ticket string created for the login form is passed along with the username and password to prevent the replaying of credentials. MamaCAS does not implement login tickets and instead relies on the built-in CSRF protection for the login form.

Ticket-granting ticket (3.6) This ticket string is stored on the server and keys to a ticket-granting cookie provided by the client to identify an existing single sign-on session. MamaCAS does not implement ticket-granting tickets, but instead uses Django sessions to determine if a single sign-on session has been established.

1.8 Changelog

Listed are the high-level, notable changes for each MamaCAS release. Backwards incompatible changes or other upgrade issues are also described here. For additional detail, read the complete commit history. From version 0.4.0 and following, version numbers follow the semantic versioning scheme.

django-mama-cas 2.3.0 [2017-05-15]

- Official Django 1.11 support (thanks @pbaehr!)
- Allow multiple attribute values in SAML responses (thanks @richardjs!)
- Replace gevent with requests-futures to eliminate monkeypatching
- Add settings to completely switch out templates (thanks @lorenmh!)

django-mama-cas 2.2.0 [2016-12-20]

- Improve output of `checkservice` management command
- Validate proxy endpoint prior to executing callback
- Handle exceptions in attribute callbacks

django-mama-cas 2.1.0 [2016-09-02]

- Add Django 1.10 compatibility
- Add per-service configuration with `MAMA_CAS_SERVICES`
- Add allowing/disallowing proxying for each service
- Add pgtUrl identifier pattern validation
- Add configurable logout URL for each service
- Add `checkservice` management command for testing service identifiers
- Reverted logout `url` parameter removal for older client compatibility
- Fix direct access of `User` model in test suite

> **Warning:** `MAMA_CAS_VALID_SERVICES`, `MAMA_CAS_ATTRIBUTE_CALLBACKS`, and `MAMA_CAS_ENABLE_SINGLE_SIGN_OUT` have been deprecated in favor of per-service configuration with `MAMA_CAS_SERVICES`.

django-mama-cas 2.0.1 [2016-01-04]

- Fix SLO request encoding

django-mama-cas 2.0.0 [2015-12-17]

- Fix Django 1.9 compatibility
- Drop support for unsupported Django versions
- Remove `url` parameter to `/logout`
- Change service URL comparison to be more strict
- Fix pgtURL validation not checking against valid services

> **Warning:** Support has been dropped for Django 1.4, 1.6 and 1.7, matching Django's supported versions policy.

django-mama-cas 1.2.0 [2015-08-21]

- Add new CAS 3 endpoints as aliases
- Update CAS validation error response codes
- Fix `TARGET` parameter case for /samlValidate
- Require HTTPS URLs for /samlValidate

django-mama-cas 1.1.1 [2015-05-15]

- Fix exception when inserting Unicode parameters

django-mama-cas 1.1.0 [2015-04-01]

- Made defusedxml requirement optional

- Changed stock templates to use Bootstrap
- Enforce CSRF protection on login form
- Fix South migrations for user model foreign keys

django-mama-cas 1.0.0 [2014-12-22]

- Add Django and South database migrations
- Add example user attribute callbacks
- Fix error when supplying non-string attribute values
- Remove `MAMA_CAS_USER_ATTRIBUTES` and `MAMA_CAS_PROFILE_ATTRIBUTES`

> **Warning:** `MAMA_CAS_USER_ATTRIBUTES` and `MAMA_CAS_PROFILE_ATTRIBUTES` have been removed. Use `MAMA_CAS_ATTRIBUTE_CALLBACKS` instead.

django-mama-cas 0.10.0 [2014-10-13]

- Default the asychronous concurrency level to `2`
- Improve test configuration and output

django-mama-cas 0.9.0 [2014-08-07]

- Add support for CAS 3.0 features
- Allow multiple custom attribute callbacks
- Use gevent for asynchronous single sign-out requests, if available
- Fix error when a malformed username was provided
- Fix logout occurring for a renew request
- Fix redirects not checking for a valid URL
- Improve removal of invalid tickets
- Default `MAMA_CAS_FOLLOW_LOGOUT_URL` to `True`
- Deprecate `MAMA_CAS_USER_ATTRIBUTES` and `MAMA_CAS_PROFILE_ATTRIBUTES`

> **Warning:** The `MAMA_CAS_ATTRIBUTES_CALLBACK` setting was renamed to `MAMA_CAS_ATTRIBUTE_CALLBACKS` and now takes a tuple of dotted paths to callables.

django-mama-cas 0.8.1 [2014-05-20]

- Fix validation response not returning PGTIOU

django-mama-cas 0.8.0 [2014-05-09]

- Add single sign-out functionality
- Add callback for returning custom user attributes
- Fix support for custom user models with no `username` field

django-mama-cas 0.7.1 [2014-01-28]

- Fix Python 2.6 compatibility

django-mama-cas 0.7.0 [2014-01-21]

- Generate CAS 2.0 XML responses instead of using templates
- Expire PGTs according to SESSION_COOKIE_AGE
- Change ticket created field to expiry date
- Change ticket expiration duration to seconds
- Fix ticket cleanup cascading to valid tickets

Warning: The `created` field on `ServiceTicket`, `ProxyTicket` and `ProxyGrantingTicket` was renamed to `expires`. If upgrading, you must ensure this field is renamed accordingly.

Warning: The `MAMA_CAS_TICKET_EXPIRE` setting previously specified ticket expiration in minutes and defaulted to *5*. Now the setting is specified in seconds and defaults to *90*.

django-mama-cas 0.6.1 [2013-11-11]

- Django 1.6 compatibility
- Handle exceptions raised by authentication backends

django-mama-cas 0.6.0 [2013-09-04]

- Add Python 3 compatibility
- Add a setting to follow provided logout URLs

django-mama-cas 0.5.0 [2013-04-29]

- Fix login template not validating data properly
- Respect REQUESTS_CA_BUNDLE environment variable
- Fix login failures with case-sensitive authentication backends
- Support for Django 1.5 custom User models

django-mama-cas 0.4.0 [2013-01-31]

- Implement service management setting
- Improve logging levels and specificity
- Fix ticket expiration setting name
- Fix PGTs expiring according to the standard expiration value

django-mama-cas 0.3 [2012-10-26]

- Implement warn parameter for the credential acceptor
- Parse XML in tests to better check validity
- Fix partial logout with the renew parameter
- Implement custom attributes returned with a validation success

django-mama-cas 0.2 [2012-07-12]

- Implement internationalization
- Add proxy ticket validation
- Substantial improvements to the test suite

- Add traversed proxies to proxy validation response
- Add form class to extract usernames from email addresses

Python Module Index

m

Index

L

M