
django-mama-cas Documentation

Release 1.0.0

Jason Bittel

April 02, 2015

1	Contents	3
1.1	Installation	3
1.2	Settings	4
1.3	Security	6
1.4	Templates	7
1.5	Management Commands	8
1.6	Forms	8
1.7	CAS Protocol	9
1.8	Changelog	10
	Python Module Index	13

MamaCAS is a Django [Central Authentication Service \(CAS\)](#) single sign-on 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 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.

Contents

1.1 Installation

1.1.1 Prerequisites

The primary prerequisite of MamaCAS is [Django](#) itself. MamaCAS supports the last two major release versions of Django and the current LTS release, generally following Django's [support policy](#). Other versions of Django may work, but are not officially tested or supported. See the [Django downloads](#) page for information on downloading and installing Django.

If you're installing MamaCAS manually, such as from the [GitHub](#) repository, you'll need to install the [Requests](#) and [defusedxml](#) libraries. The optional [gevent](#) module may also be installed to enable asynchronous single sign-out requests.

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

First, 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 `manage.py 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 = patterns('',
    # ...existing urls...
    (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 subpath to the CAS root (e.g. `http://example.com/cas/login`) add the path to the URL regular expression:

```
urlpatterns = patterns('',
    # ...existing urls...
    (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. Note that some of these settings alter stock CAS behavior.

`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_ASYNC_CONCURRENCY`

Default `2`

If single sign-out is enabled and [gevent](#) is installed, this setting limits the concurrency of requests sent for a logout event. If the number of requests reaches this limit, additional requests block until there is room. Setting this value to zero disables this limiting.

`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 available 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`.

django.conf.settings.**MAMA_CAS_ENABLE_SINGLE_SIGN_OUT**

Default False

If set, causes single sign-out 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 sign-out requests are sent synchronously. If `gevent` is installed, they are sent asynchronously.

django.conf.settings.**MAMA_CAS_FOLLOW_LOGOUT_URL**

Default True

Controls the client redirection behavior at logout when the `url` (CAS 2.0) or `service` (CAS 3.0) parameter is provided. When this setting is `True` and one of these parameters is present, the client will be redirected to the specified URL. When this setting is `False`, the client will be redirected to the login page. When `url` is present, the login page will then display the provided URL as a recommended link to follow.

If neither parameter is specified or is not a valid service URL, the client will be redirected to the login page.

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 will become 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',  
)
```

The `url` and `service` parameters are checked against this list of services at logout. If the provided URL does not match one of these regular expressions, it is ignored.

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 configuration options that ought to be considered.

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

mama_cas/login.html

This template 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

This template is 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, first make sure the custom templates come first in the search order. If the custom templates exist in a directory specified by `TEMPLATE_DIRS`, then `TEMPLATE_LOADERS` should look something like this:

```
TEMPLATE_LOADERS = (
    'django.template.loaders.filesystem.Loader',
    'django.template.loaders.app_directories.Loader',
)
```

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` in one of the `TEMPLATE_DIRS` that extends `mama_cas/__login.html`:

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

{% block styles %}
    {{ block.super }}
    <style>#header { font-size:3em; text-align:center; color:#aaa; }</style>
{% endblock styles %}

{% block header %}
    <h1>If You Can Believe Your Eyes and Ears</h1>
{% endblock header %}
```

The stock templates have a variety of blocks defined to make many modifications straightforward.

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 to the user for a login, logout or authentication notification event.

Non-field errors The login form's `non_field_errors` inform the user of authentication failures and other login problems.

1.5 Management Commands

MamaCAS ships with 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

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 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 asynchronous 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 PGTIOUT

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

m

`mama_cas.forms`, [8](#)

L

LoginForm (class in mama_cas.forms), [9](#)

LoginFormEmail (class in mama_cas.forms), [9](#)

M

mama_cas.forms (module), [8](#)

MAMA_CAS_ALLOW_AUTH_WARN (in module
django.conf.settings), [4](#)

MAMA_CAS_ASYNC_CONCURRENCY (in module
django.conf.settings), [4](#)

MAMA_CAS_ATTRIBUTE_CALLBACKS (in module
django.conf.settings), [4](#)

MAMA_CAS_ENABLE_SINGLE_SIGN_OUT (in
module django.conf.settings), [5](#)

MAMA_CAS_FOLLOW_LOGOUT_URL (in module
django.conf.settings), [5](#)

MAMA_CAS_TICKET_EXPIRE (in module
django.conf.settings), [5](#)

MAMA_CAS_TICKET_RAND_LEN (in module
django.conf.settings), [5](#)

MAMA_CAS_VALID_SERVICES (in module
django.conf.settings), [5](#)