Ozone Platform Configuration Guide

DOD GOSS

DRAFT

Exported on Sep 28, 2018

Table of Contents

DRAFT

1 Introduction

1.1 Objectives

This guide covers topics relevant to installing, configuring, and administering an OZONE Platform. Overview OZONE is an ecosystem of software that enables users from numerous organizations to share data and manipulate it solely within browser memory.

1.2 Document Scope

This guide is intended for administrators or developers who wish to learn how to install and deploy the platform for the purposes of gaining an overall familiarity with product as well as deploying a functional development environment.

1.3 Related Documents

Ozone Platform Quick Start Guide

DRAFT

2 Authentication

2.1 CAS Authentication Setup

2.1.1 Using ozp-docker repository

1. If you are utilizing the ozp-docker repository, enabling CAS support is as simple as setting the `CAS_ENABLED` environment to `"true"` in the docker-compose file.

   Code Block 1 docker-compose.py

   ```
   # IMPORTANT: If you intend to use the test CAS server included
   # Make sure to add an entry for `ozp_cas` to the host system's
   `hosts` file.
   # For example, add `127.0.0.1 ozp_cas` to /etc/hosts or
   C:\Windows\System32\drivers\etc\hosts
   CAS_ENABLED:          "true"
   # URL that points to the demo CAS server
   CAS_SERVER_URL:       "http://ozp_cas/cas/"
   ```

2.1.2 Manual CAS Configuration

1. Add the CAS client library to `requirements.txt`

   Code Block 2 requirements.txt

   ```
   django-cas-ng=3.5.9
   ```

2. Add the CAS app to the list of installed apps in ozp/`settings.py`

   Code Block 3 ozp/settings.py

   ```
   INSTALLED_APPS += ('django_cas_ng',)
   ```

3. Configure the CAS options in `ozp/settings.py` to either read the supplied environment variables, or supply your own predefined settings.
   See the django-cas-ng library documentation at https://github.com/mingchen/django-cas-ng for additional details on available configuration options.

DRAFT

Code Block 4 ozp/settings.py

```
# Add the django_cas_ng CAS library as an installed app
INSTALLED_APPS += (
    'django_cas_ng',
)

# Add the CAS middleware to the middleware chain
MIDDLEWARE += (
    'django_cas_ng.middleware.CASMiddleware',
)

# Set the authentication backends, as required by the CAS
middleware
AUTHENTICATION_BACKENDS = (
    'django.contrib.auth.backends.ModelBackend',
    'django_cas_ng.backends.CASBackend',
)

# Replace the existing authentication with Session-based
authentication
REST_FRAMEWORK['DEFAULT_AUTHENTICATION_CLASSES'] = (
    'rest_framework.authentication.SessionAuthentication',
)

CAS_SERVER_URL = os.getenv('CAS_SERVER_URL',
'https://localhost:8443/cas/')

CAS_CREATE_USER = False
CAS_REDIRECT_URL = "/"
CAS_VERSION = "2"
CAS_USERNAME_ATTRIBUTE = "uid"
```

4. Add the login and logout URLs to `ozp/urls.py`:

   Code Block 5 ozp/urls.py

```
import django_cas_ng.views

...

urlpatterns += [
    url(r'^login/$', django_cas_ng.views.login,
name='cas_ng_login'),
    url(r'^logout/$', django_cas_ng.views.logout,
name='cas_ng_logout'),
]
```

5. *(Optional)* Disable CAS re-directs for AJAX requests and reject with an HTTP 403 Forbidden error status:
   a. An example middleware is provided in `ozp-docker/patch/ozp-backend/ozp/middleware/cas.py`:

DRAFT

Code Block 6 cas.py

```
from django.shortcuts import redirect
from django.http import HttpResponseForbidden

class HandleUnauthenticatedAJAXRequestsMiddleware:
    """
    Optional middleware. Unauthenticated AJAX requests
are rejected as 403 Forbidden.
    Non-AJAX requests are redirected to the login page
(handled in CASbackend Middleware)
    """
    def __init__(self, get_response):
        self.get_response = get_response
    def __call__(self, request):
        response = self.get_response(request)
        if request.user.is_authenticated():
            return response
        if request.is_ajax() or not
request.META.get('HTTP_ORIGIN') is None:
            return HttpResponseForbidden()
        return response
```

b. The middleware can be enabled by adding it to the `MIDDLEWARE` settings in `ozp-backend/ozp/settings.py`:

Code Block 7 settings.py

```
# Add the CAS middleware to the middleware chain
MIDDLEWARE += (

'ozp.middleware.cas.HandleUnauthenticatedAJAXRequestsM
iddleware',  # Must come BEFORE CASMiddleware
    'django_cas_ng.middleware.CASMiddleware',
)
```

DRAFT

3 Classification Banner

3.1 Overview

The `ozp-classification` module has been modified to allow the classification banner to display the standard classification banner background colors.

To enable this option, set the `ozp-color-banners` attribute to `true`.

```
<body ozp-classification="U-FOUO" ozp-color-banners="true">...</body>
```

DRAFT

An option to set a system high classification level has been added to the `OzoneConfig.js` template file in the `ozp-react-commons` module. To override the default `SYSTEM_HIGH_CLASSIFICATION` value of "U" for Unclassified, you may either change the template file or supply the desired value as the `SYSTEM_HIGH_CLASSIFICATION` environment variable during the build process.

3.2 Implementation Details

3.2.1 ozp-center

The `ozp-center` module enables the classification banner plugin in the `/app/js/main.js` source file, enables the classification background colors, and sets the classification level to the value provided by the `ozp-react-commons OzoneConfig.js` configuration file.

Code Block 8 /app/js/main.js

```
var {
  ...
  SYSTEM_HIGH_CLASSIFICATION
} = require('ozp-react-commons/OzoneConfig');


...

$(function() {
    $(document).classification({
        level: SYSTEM_HIGH_CLASSIFICATION,
        colorBanners: true
    });
});
```

3.2.2 ozp-hud

The `ozp-hud` module enables the classification banner plugin in the `/app/index.html` source file, enables the classification background colors, and sets the classification level to the value provided by the `ozp-react-commons` OzoneConfig.js configuration file.

Code Block 9 /app/index.html

```
<script>
    $(document).classification({
        level: window.OzoneConfig.SYSTEM_HIGH_CLASSIFICATION || 'C-NF',
        colorBanners: true
    });
</script>
```

3.2.3 ozp-webtop

The `ozp-webtop` module enables the classification banner plugin in the `/src/index.html` source file and enables the classification background colors, and sets the classification level to the value provided by the application scope variable systemHighClassification. The systemHighClassification scope variable is set in /src/app/app.js from the value provided in the `ozp-webtop` OzoneConfig.js configuration file.

Code Block 10 /src/app/app.js

```
$rootScope.systemHighClassification =
$window.OzoneConfig.SYSTEM_HIGH_CLASSIFICATION;
```

Code Block 11 /src/index.html

```
...
<body ng-app="ozpWebtop" ozp-
classification="{{systemHighClassification}}"" ozp-color-banners="true">
...
```

DRAFT