Authentication

Authentication using Symfony Security component

Native and universal `form_login` is used, in conjunction with an extended `DaoAuthenticationProvider` (DAO stands for *Data Access Object*), the `RepositoryAuthenticationProvider`. Native behavior of `DaoAuthenticationProvider` has been preserved, making it possible to still use it for pure Symfony applications.

Security controller

A `SecurityController` is used to manage all security-related actions and is thus used to display login form. It is pretty straightforward and follows all standards explained in Symfony security documentation.

Base template used is `EzPublishCoreBundle:Security:login.html.twig` and stands as follows:

```
{% extends layout %}

{% block content %}
    {% block login_content %}
        {% if error %}
            <div>{{ error.message|trans }}</div>
        {% endif %}

        <form action="{{ path( 'login_check' ) }}" method="post">
        {% block login_fields %}
            <label for="username">{{ 'Username:'|trans }}</label>
            <input type="text" id="username" name="_username" value="{{ last_username
}}" />

            <label for="password">{{ 'Password:'|trans }}</label>
            <input type="password" id="password" name="_password" />

            <input type="hidden" name="_csrf_token" value="{{
csrf_token("authenticate") }}" />

            {#
                If you want to control the URL the user
                is redirected to on success (more details below)
                <input type="hidden" name="_target_path" value="/account" />
            #}

            <button type="submit">{{ 'Login'|trans }}</button>
        {% endblock %}
        </form>
    {% endblock %}
{% endblock %}
```

The layout used by default is `%ezpublish.content_view.viewbase_layout%` (empty layout) but can be configured easily together with the login template:

ezplatform.yml

```
ezpublish:
    system:
        my_siteaccess:
            user:
                layout: "AcmeTestBundle::layout.html.twig"
                login_template: "AcmeTestBundle:User:login.html.twig"
```

Redirection after login

By default, Symfony redirects to the URI configured in `security.yml` as `default_target_path`. If not set, it will default to `/`.

This setting can be set by siteaccess, via `default_page` setting.

Configuration

To use Symfony authentication with eZ Platform, the configuration goes as follows:

app/config/security.yml

```
security:
    firewalls:
        ezpublish_front:
            pattern: ^/
            anonymous: ~
            form_login:
                require_previous_session: false
            logout: ~
```

app/config/routing.yml

```
login:
    path:   /login
    defaults:  { _controller: ezpublish.security.controller:loginAction }
login_check:
    path:   /login_check
logout:
    path:   /logout
```

Note
You can fully customize the routes and/or the controller used for login. However, remember to match `login_path`, `check_path` and `logout.path` from `security.yml`.

See security configuration reference and standard login form documentation.

Access control

See the documentation on access control.

Remember me

It is possible to use the `remember_me` functionality. For this you can refer to the Symfony cookbook on this topic.

If you want to use this feature, you must at least extend the login template in order to add the required checkbox:

```
{# your_login_template.html.twig #}
{% extends "EzPublishCoreBundle:Security:login.html.twig" %}

{% block login_fields %}
    {{ parent() }}
    <input type="checkbox" id="remember_me" name="_remember_me" checked />
    <label for="remember_me">Keep me logged in</label>
{% endblock %}
```

Login handlers / SSO

Symfony provides native support for multiple user providers. This makes it easy to integrate any kind of login handlers, including SSO and existing third-party bundles (e.g. FR3DLdapBundle, HWIOauthBundle, FOSUserBundle, BeSimpleSsoAuthBundle, etc.).

Further explanation can be found in the multiple user providers cookbook entry.

Integration with Legacy

- When **not** in legacy mode, legacy `user/login` and `user/logout` views are deactivated.
- Authenticated user is injected in legacy kernel.

Authentication with Legacy SSO Handlers

To be able to use your legacy SSO (Single Sign-on) handlers, use the following config in your `ezpublish/config/security.yml`:

Use your legacy SSO handlers

```
security:
    firewalls:
        ezpublish_front:
            pattern: ^/
            anonymous: ~
            # Adding the following entry will activate the use of old SSO handlers.
            ezpublish_legacy_sso: ~
```

If you need to create your legacy SSO Handler, please read this entry