Министерство образования и науки Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего образования «Уральский федеральный

университет имени первого президента России Б. Н. Ельцина»

Институт фундаментального образования

Кафедра интеллектуальных информационных технологий

**Фреймворк Pyramid. Безопасность**

Пояснительная записка

|  |  |
| --- | --- |
| Руководитель: | Свинцов Д.М. |
| Студентка гр. ФО-350005 | Богомолова Е.В. |

Екатеринбург 2017

Оглавление

[ВВЕДЕНИЕ 3](#_Toc501745033)

[Аутентификация и авторизации 4](#_Toc501745034)

[Что тебе дозволено? 5](#_Toc501745035)

[Добавление авторизации в проект 5](#_Toc501745036)

[Права доступа по умолчанию 6](#_Toc501745037)

[Access листы (ACL) 7](#_Toc501745038)

[Практическое применение 9](#_Toc501745039)

[ЗАКЛЮЧЕНИЕ 11](#_Toc501745040)

# ВВЕДЕНИЕ

В Pyramid система безопасности поделена на 2 части. Это аутентификация и авторизация. Pyramid имеет возможность обрабатывать сложные шаблоны аутентификации и авторизации.

# Аутентификация и авторизации

Аутентификация производит идентификацию пользователя, его проверку (например, что он есть в базе данных и он не заблокирован) и определяет какими правами он наделен. Авторизация, система которая проверяет, имеет ли этот пользователь доступ к запрошенному ресурсу.

Для аутентификации используется расширение repoze.who, которое отвечает за идентификацию пользователя.

*Who?* т.е. *Кто? ты.*

Для авторизации используется расширение repoze.what, которое проверяет какие ресурсы тебе доступны.

*What?* т.е. *Что? доступно тебе.*

Определение текущего пользователя при поступлении HTTP запроса, это задача аутентификации. Производится она в 3 этапа:

1. Идентифицируем пользователя путем проверки токенов/заголовков/и т.д. в HTTP запросе.

Например: ищем auth\_token в cookie запроса, проверяем, что токен правильно подписан, и возвращаем id пользователя.

1. Подтверждаем статус идентифицированного пользователя. Например: проверяем что id этого пользователя все еще в базе данных и пользователь еще активен. Пользователя могли удалить из БД, но при этом в cookie браузера хранится валидный токен auth\_token.
2. Ищем группы, которые принадлежат пользователю и добавляем их в список.

Например: берем из БД группы пользователя и добавляем в список. Для текущего идентифицированного пользователя это может быть: «vasya», «user\_admin», «editor».

## Что тебе дозволено?

Каждый ресурс Pyramid может быть защищен правами доступа. Задача авторизации определение того, какие пользователи имеют доступ к ресурсам.

После аутентификации создается список групп пользователя. Политика авторизации запрещает или разрешает доступ к ресурсу на основании этого списка групп, сверяя его с правами ресурса.

## Добавление авторизации в проект

В Pyramid по умолчанию авторизация отключена. Все представления (views) полностью доступны анонимным пользователям. Для того чтобы их защитить нужно добавить в настройки политику безопасности.

Для включения политики авторизации используется метод конфигуратора pyramid.config.Configurator.set\_authorization\_policy(). Для аутентификации pyramid.config.Configurator.set\_authentication\_policy() соответственно. Так как авторизация не может существовать без аутентификации, необходимо указывать обе политики в проекте.

Пример:

from pyramid.config import Configurator

from pyramid.authentication import AuthTktAuthenticationPolicy

from pyramid.authorization import ACLAuthorizationPolicy

authn\_policy = AuthTktAuthenticationPolicy('seekrit', hashalg='sha512')

authz\_policy = ACLAuthorizationPolicy()

config = Configurator()

config.set\_authentication\_policy(authn\_policy)

config.set\_authorization\_policy(authz\_policy)

Здесь pyramid.authentication.AuthTktAuthenticationPolicy это механизм аутентификации пользователя, который ищет его из «auth ticket» cookie. pyramid.authorization.ACLAuthorizationPolicy механизм авторизации по access листам.

# Права доступа по умолчанию

Если ресурсу не присвоены права доступа, то используются права по умолчанию. В пирамиде права по умолчанию (pyramid.security.NO\_PERMISSION\_REQUIRED) подразумевают, что ресурсы доступны всем, даже анонимным пользователям.

Это поведение возможно изменить при помощи метода pyramid.config.Configurator.set\_default\_permission().

Пример:

config.set\_default\_permission('my\_default\_permission')

# Access листы (ACL)

Access Control List (ACL) — список контроля доступа, который определяет, кто или что может получать доступ к конкретному объекту, и какие именно операции разрешено или запрещено этому субъекту проводить над объектом.

В Pyramid access лист это список содержащий записи, определяющие права индивидуального пользователя или группы на ресурсы проекта. Элементы ACL также еще называют Access Control Entry или ACE.

Пример:

from pyramid.security import Allow, Deny

from pyramid.security import Everyone

\_\_acl\_\_ = [

(Deny, 'vasya', 'move'),

(Deny, 'group:blacklist', ('add', 'delete', 'edit')),

(Allow, Everyone, 'view'),

(Allow, 'group:editors', ('add', 'edit')),

(Allow, 'group:editors', 'move'),

(Allow, 'group:deleter', 'delete'),

],

где \_\_acl\_\_ это список контроля доступа (ACL), (Allow, Everyone, 'delete') это ACE, т.е. запись в ACL.

Первый элемент в списке ACE это действие, т.е. «что делать?» разрешить или запретить. Действия представляются константами pyramid.security.Allow и pyramid.security.Deny.

Второй элемент списка это группы, к которым принадлежит пользователь.

Последний элемент это права или список прав.

Также существую специальные группы:

* pyramid.security.Everyone - для всех.
* pyramid.security.Authenticated - для аутентифицированнных пользователей.

Если хотим запретить всем, кроме тех ACE которые в списке, можем написать это так:

from pyramid.security import Allow

from pyramid.security import ALL\_PERMISSIONS

\_\_acl\_\_ = [(Allow, 'fred', 'view'),

(Deny, Everyone, ALL\_PERMISSIONS)]

или воспользоваться встроенным в Pyramid ACE:

from pyramid.security import Allow

from pyramid.security import DENY\_ALL

\_\_acl\_\_ = [(Allow, 'fred', 'view'),

DENY\_ALL]

# Практическое применение

Реализация входа и выхода в ресурс под своей учетной записью:

from wsgiref.simple\_server import make\_server

from pyramid.authentication import AuthTktAuthenticationPolicy

from pyramid.authorization import ACLAuthorizationPolicy

from pyramid.config import Configurator

from pyramid.httpexceptions import HTTPFound

from pyramid.response import Response

from pyramid.security import Allow, forget, remember

class HelloFactory(object):

def \_\_init\_\_(self, request):

self.\_\_acl\_\_ = [

(Allow, 'vasya', 'view'),

(Allow, 'group:editors', 'add'),

(Allow, 'group:editors', 'edit'),

]

def hello\_world(request):

return Response('Hello %(name)s!' % request.matchdict)

def login(request):

headers = remember(request, 'vasya')

return HTTPFound(location=request.route\_url('hello', name='vasya'),

headers=headers)

def logout(request):

headers = forget(request)

return HTTPFound(location=request.route\_url('hello', name='log out!!!'),

headers=headers)

if \_\_name\_\_ == '\_\_main\_\_':

authn\_policy = AuthTktAuthenticationPolicy('seekrit', hashalg='sha512')

authz\_policy = ACLAuthorizationPolicy()

config = Configurator(root\_factory=HelloFactory)

config.set\_authentication\_policy(authn\_policy)

config.set\_authorization\_policy(authz\_policy)

config.add\_route('hello', '/hello/{name}')

config.add\_view(hello\_world,

route\_name='hello',

permission='view')

# login form

config.add\_route('login', '/login')

config.add\_route('logout', '/logout')

config.add\_view(login, route\_name='login')

config.add\_view(logout, route\_name='logout')

app = config.make\_wsgi\_app()

server = make\_server('0.0.0.0', 8080, app)

server.serve\_forever()

Добавляем окно входа, которое отображает форму входа и обрабатывает сообщение из формы входа, проверяя учетные данные.

Также добавляем окно выхода, которое можно вызывать в приложении и предоставить ссылку на него. Это представление очистит учетные данные зарегистрированного пользователя и перенаправит обратно на главную страницу.

# ЗАКЛЮЧЕНИЕ

Главными в системе безопасности Pyramid являются авторизация и аутентификация, которые определяют, кто и какие действия может производить.

Почти все примеры авторизации зависят от ACLAuthorizationPolicy, т к в ACL прописывается основная информация о доступности.

Для безопасности своих данных, необходимо добавить авторизацию в проект и изменять права доступа по умолчанию.