

Painless Authentication

with Access Tokens



Mathieu Santostefano

Web developer at [joliCode](https://jolicode.com)

Symfony Core Team Member

Twitter @welcomattic

GitHub @welcomattic







On nov. 18 2021, I received an email to join the Core Team

Welcome to the Symfony Core team

Fabien Potencier

jeu. 18 nov. 2021 16:07

 anglais ▾ > français ▾ Traduire le message

Hi,

And thank you for accepting to be part of the Symfony Core Team.





On the menu





On the menu

1. Access token, what is it?





On the menu

1. Access token, **what is it?**
2. Implementation with **Symfony 6.1**





On the menu

1. Access token, **what is it?**
2. Implementation with **Symfony 6.1**
3. Time travel





On the menu

1. Access token, **what is it?**
2. Implementation with **Symfony 6.1**
3. Time travel
4. Implementation with **Symfony 6.2**





On the menu

1. Access token, **what is it?**
2. Implementation with **Symfony 6.1**
3. Time travel
4. Implementation with **Symfony 6.2**
5. Code examples





On the menu

1. Access token, **what is it?**
2. Implementation with **Symfony 6.1**
3. Time travel
4. Implementation with **Symfony 6.2**
5. Code examples
6. In the future?





Who has Access Token authentication in their app?



Who is working with the new Symfony Security?



Who is working with the new Symfony Security?

Thank you for it, Wouter 🙏

✨ The New Security System

- Removed everything but Guards
- Moved to an event-based system
- Authenticator based: instantiate a Passport with Badges

Your job is to use Authenticator or implement your own

The New Security System

Event-based system

- You can interact on different levels
 - CheckPassportEvent
 - AuthenticationTokenCreatedEvent
 - AuthenticationSuccessEvent
 - LoginSuccessEvent
 - LoginFailureEvent
 - LogoutEvent
 - TokenDeauthenticatedEvent
 - SwitchUserEvent

✨ The New Security System

As before, you can still handle what happens in case of authentication success or failure

👑 Like many things since the last years in Symfony, it improves the DX



What is an Access Token?



What is an Access Token?

- `i-am-an-4cc3ss-t0k3n` could be an Access Token



What is an Access Token?

- `i-am-an-4cc3ss-t0k3n` could be an Access Token
- `mF_9.B5f-4.1JqM` could be an Access Token



What is an Access Token?

- `i-am-an-4cc3ss-t0k3n` could be an Access Token
- `mF_9.B5f-4.1JqM` could be an Access Token
- `eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4...`
could be an Access Token



What is an Access Token?

- `i-am-an-4cc3ss-t0k3n` could be an Access Token
- `mF_9.B5f-4.1JqM` could be an Access Token
- `eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4...` could be an Access Token
- `🌮` could be an Access Token



What is an Access Token?

- `i-am-an-4cc3ss-t0k3n` could be an Access Token
- `mF_9.B5f-4.1JqM` could be an Access Token
- `eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4...` could be an Access Token
- `🌮` could be an Access Token



An Access Token is represented by a string.



Use cases

Authentication with **Access Token** is useful in some contexts, like

- Stateless user login
- **API Gateway** in front of private APIs
- Application that access to personal data provided by a third party
- **Micro-services** between them
- Client applications of a **SaaS API**
- ...



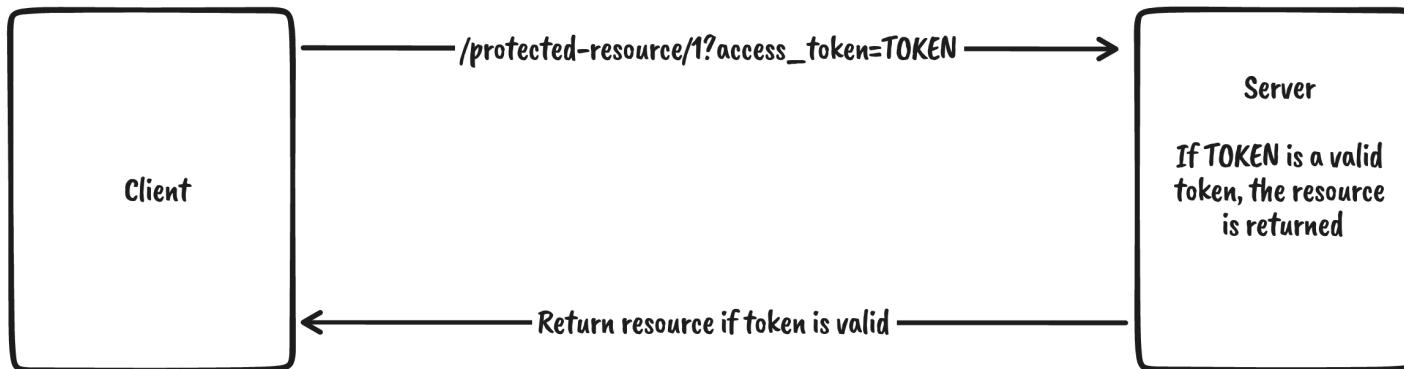
Ok, so?

- Must be sent in a **HTTP request** to fetch a protected resource
- The application expects to find a token, **👉 validate** **👉** it and decide to allow access or not



Ok, so?

- Must be sent in a **HTTP request** to fetch a protected resource
- The application expects to find a token, **👉 validate** **👉** it and decide to allow access or not





Token issuer

**Let's assume our tokens come from an external authentication server,
the user identity has been verified by this server**



Validate the token?

The validation process is up to you

- Check if the string is **present** in a database
- **Compute a hash** and compare it to the expected one
- **Decode the token** (base64, ...) and make **assertions** on decoded values
- Ensure the **expiration date** is not passed if needed
- Check if the token has been **revoked** or not
- Call an **OpenID Connect** server to validate the token
- ...

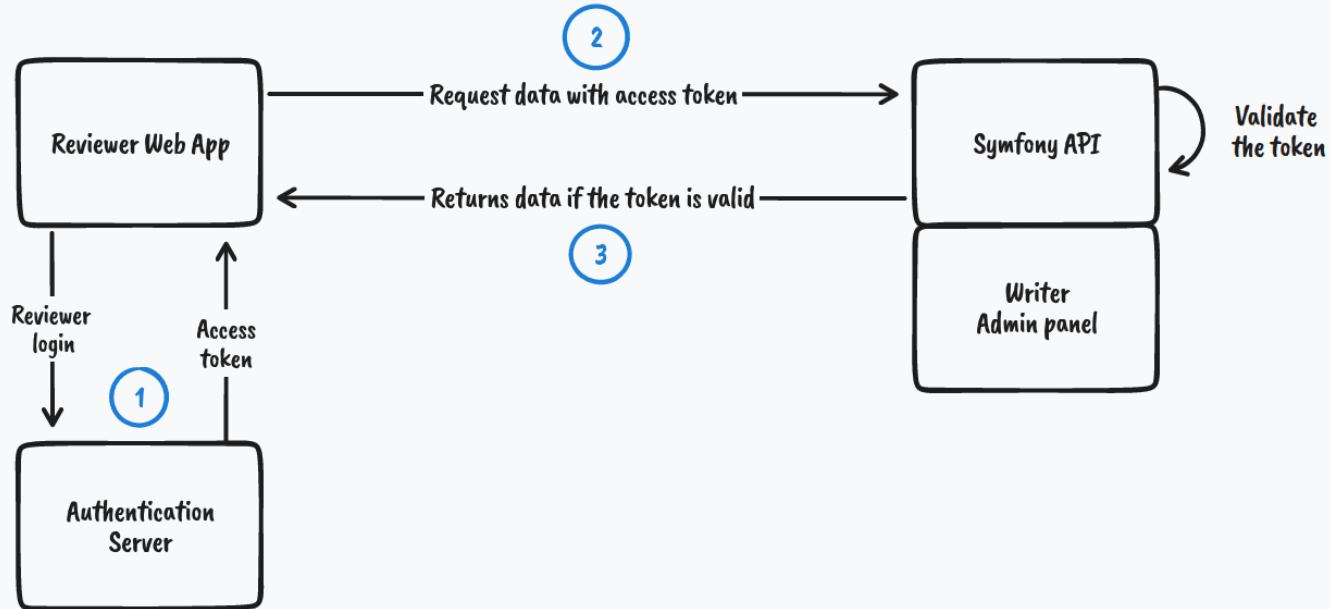


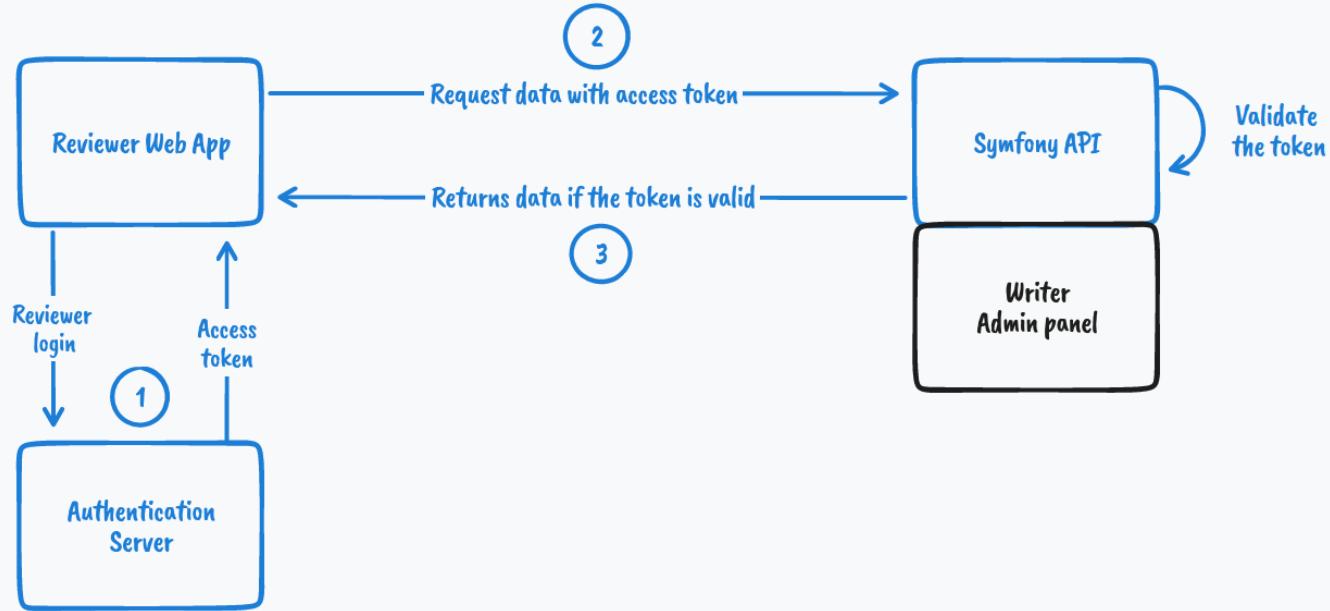
How to set up an Access Token auth with Symfony?



Marvel Scenarios Manager, a web app for writers and reviewers

- Each user (Writers and Reviewers) has an API key
- Admin for Writers
 - Form login authentication
 - Writers can create Scenarios
- Web app for Reviewers
 - Call the private API using API Key obtained after Reviewer login
- Admins can revoke API key at any time





👉 With Symfony <= 6.1

Create a custom Authenticator

1. Extract **token**
2. Decode **token** if needed (JWT, SAML, ...)
3. Check **token** validity
4. Retrieve user identifier from the **token**
5. Then load User object
6. Handle authentication failure cases



Have you heard about RFC 6750?

- Token transportation
 - In **request header**? → `Authorization` header
 - In **query string**? → parameter `access_token`
 - In **request body**? → parameter `access_token`



Have you heard about RFC 6750?

- Token transportation
 - In **request header**? → `Authorization` header
 - In **query string**? → parameter `access_token`
 - In **request body**? → parameter `access_token`
- `WWW-Authenticate` response header in case of failure



Have you heard about RFC 6750?

- Token transportation
 - In **request header**? → `Authorization` header
 - In **query string**? → parameter `access_token`
 - In **request body**? → parameter `access_token`
- `WWW-Authenticate` response header in case of failure
- **HTTPS** protocol mandatory

Your responsibility

Symfony's responsibility,
but customizable

Access Token Authentication Process

≤ 6.1





Some code

```
1  /** Simplified version */
2  private function extractToken(Request $req): ?string
3  {
4      return match (true) {
5          // Header
6          $req->headers->has('Authorization') => str_replace('Bearer ', '', $req->headers->get('Authorization')),
7
8          // Query string
9          $req->query->has('access_token') => $req->query->get('access_token'),
10
11         // Request body
12         $req->request->has('access_token') => $req->request->get('access_token'),
13
14         default => null,
15     };
16 }
```



Some code

```
1  public function authenticate(Request $req): Passport
2  {
3      if (null === $apiKey = $this->extractToken($req)) {
4          throw new AuthenticationException('No API Key provided');
5      }
6
7      // Here, it could be some logic to validate the token
8
9      return new SelfValidatingPassport(
10         new UserBadge($apiKey,
11             function (string $apiKey) {
12                 return $this->userRepository->findOneByApiKey($apiKey);
13             }
14         )
15     );
16 }
```



Boring code

- We have to repeat this code in all our applications, **boring**
- Our responsibility to respect RFC6750, **boring**
- No body likes boring code
- Boring code is code we rewrite in all projects, no business value
- Poor Developer eXperience, Symfony tends to improve DX

This is definitely improvable



 SymphonyCon
DISNEYLAND PARIS
NOV 17-18 2024



Discussions started long time ago

- April 2019, at EU FOSSA Hackathon



[RFC] Symfony Security rework tracking issue #30914
curry684 opened this issue on Apr 6, 2019 · 42 comments

Authentication

- User impersonation (including multiple impersonation without having to log out) *not affected by Security changes*
- Social login (login in the app using Google, Facebook, GitHub, Twitter, etc.) and generic OAuth support (  [WIP][Security] OAuth2 component #31952 (comment))
- Login throttling (limit the number of failed login attempts over a period of time)
- Simultaneous session limiting (e.g. each user can login only from one device at the same time)
- Two-factor (or multi-factor) authentication ( [Security] Native support for 2FA #28868 (comment))



Discussions started long time ago

- June 2019, first PR about OAuth2 Component. ✘ Aborted

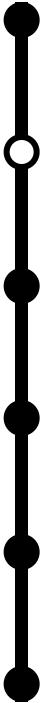
[WIP][Security] OAuth2 component #31952

Closed · symfony:master ← Guikingone:feat/oauth · ⚙

Conversation 45 · Commits 2 · Checks 0 · Files changed 67

Guikingone (Guillaume Loulier) on Jun 8, 2019 • edited

Q	A
Branch?	master
Bug fix?	no
New feature?	yes





Discussions started long time ago

- September 2019, Wouter's 1st PR about redesign of Security

[Security] AuthenticatorManager to make "authenticators" first-class security #33558

Merged by fabpot symfony:master ← wouterj:security/deprecate-providers-listeners on Apr 21, 2020 v5.1.0-BETA1

Conversation 177 Commits 30 Checks 0 Files changed 77

wouterj (Wouter de Jong) on Sep 11, 2019 • edited Member

Q	A
Branch?	master
Bug fix?	no
New feature?	yes
Deprecations?	no
Tickets	-

Reviewers – review now

- javiereguiluz
- rosier
- nicolas-grekas
- stof
- nonlagronomie
- fabpot
- weaverryan
- chaliasr





Discussions started long time ago

- April 2020, continuation of Wouter's work on new Security system

[Security] Integrated Guards with the Authenticator system #36570

Merged by fabpot symfony:master ← wouterj:security/new-system-guard on Apr 25, 2020 v6.1.0-BETA1

Conversation 4 Commits 1 Checks 0 Files changed 4

wouterj (Wouter de Jong) on Apr 24, 2020 • edited

Q	A
Branch?	master
Bug fix?	no
New feature?	yes
Deprecations?	no
Tickets	-

Reviewers – review now
fabpot
chalaar

Assignees – assign yourself

Labels
Feature Status Review

Projects

[Security] Removed anonymous in the new security system #36574

Merged by fabpot symfony:master ← wouterj:security/remove-anonymous on May 3, 2020 v6.1.0-BETA1

Conversation 47 Commits 1 Checks 0 Files changed 14

wouterj (Wouter de Jong) on Apr 24, 2020 • edited

Q	A
Branch?	master
Bug fix?	no
New feature?	yes
Deprecations?	no
Tickets	-

Reviewers – review now
javiergulluz
weaverryan
simonberger
fabpot

Assignees – assign yourself

Labels
Feature Security





Discussions started long time ago

- March 2022, Vincent Chalamon opens an issue about "Bearer Authenticator"

[Security] Add Bearer Authenticator #45844

(Closed) 14 comments · Fixed by #46428

vincentchalamon (Vincent) on Mar 25 · edited

Description

I'm implementing an OIDC architecture in an API first project, and wanted to secure my API with an authenticator. I first took a look at bundles like [hwi/oauth-bundle](#) or [knpu/university/oauth2-client-bundle](#), but they're not adapted to an API as they provide bunch of features not needed in an API, and provides their own JWT generation and storage (e.g.: cookie).

I talked about it with [@dunglas](#) who shares the idea to implement a Bearer authenticator directly in the Symfony Security bundle. The idea behind this authenticator would be to retrieve the token from the `Authorization` header (prefixed with `Bearer`), and validate and decode it in a `BearerToken` using [locabucl/jwt](#).

Example

```
# config/packages/security.yaml
security:
    firewall:
        main:
            bearer:
                signature: '/path/to/signature/key' # Could be a file path, plaintext, or even an url
                algorithm: 'hmac.sha256'
                key: 'customUserId' # Which key from the JWT should be sent to the UserProvider (default: '')
```

Assignees – assign yourself

Labels

Feature x Security x

Projects

None yet

Milestone

No milestone

Development

Successfully merging a pull request may close this issue.

↳ [Security] Access Token Authenticator
Spomky/symfony

↳ [Security] Implement HttpBearerAuthenticator
vincentchalamon/symfony





Discussions started long time ago

- Finally, on May 21st, 2022, Vincent Chalamon & Florent Morselli each open a pull request to implement the Authenticator

[Security] Implement HttpBearerAuthenticator #46429

Closed [symfony/symfony:6.2](#) ← [vincentchalamon:feat/security/bearer-authenticator](#) ⚡ ④ · 14

Conversation 2 · Commits 1 · Checks 7 · Files changed 14

vincentchalamon (Vincent) on May 21

Q	A
Branch?	6.2
Bug fix?	no
New feature?	yes
Deprecations?	no
Tickets	Fix #45844
License	MIT

Reviewers — review now
wouterj, chalar

Assignees — assign

Labels [reviewed](#), Status [ready](#)

Projects None yet

[Security] Access Token Authenticator #46428

Merged by chalar [symfony:6.2](#) ← [Spomky:features/access-token-authenticator](#) ⚡ Aug 10 ⚡ +1,994 -1

Conversation 195 · Commits 1 · Checks 7 · Files changed 39

Spomky (Florent Morselli) on May 21 · edited

Q	A
Branch?	6.2
Bug fix?	yes
New feature?	yes
Deprecations?	no
Tickets	Fix #45844
License	MIT
Doc PH	symfony/symfony-docs#16819

Reviewers — review now
vincentchalamon, tyrix, stof, mmonni, derrabus, dunglas, wouterj, chalar

Assignees — assign yourself

Labels [Bug](#), [Feature](#), [Ready](#), [Security](#)
[Status: Reviewed](#)

Projects None yet

Hi,
This PR aims at fixing #45844.
It adds a new authenticator that is able to fetch a token in the request header and retrieve the associated user identifier.
The authenticator delegates the token loading to a handler. This handler could manage opaque tokens (random strings stored in a database) or self-contained tokens such as JWT, Paseto, SAML...





Thanks a lot Florent Morselli @Spomky



[Security] Access Token Authenticator #46428

Merged

by chalasr symfony:6.2

↳ Spomky:features/access-token-authenticator

on Aug 10

Edit

Code

Conversation 195

Commits 1

Checks 7

Files changed 39

+1,994 -1



Spomky (Florent Morselli) on May 21 · edited

Contributor

Q	A
Branch?	6.2
Bug fix?	yes
New feature?	yes
Deprecations?	no
Tickets	Fix #45844
License	MIT
Doc PR	symfony/symfony-docs#16819

Hi,

This PR aims at fixing #45844.

It adds a new authenticator that is able to fetch a token in the request header and retrieve the associated user identifier.

The authenticator delegates the token loading to a handler. This handler could manage opaque tokens (random strings stored in a database) or self-contained tokens such as JWT, Paseto, SAML...

Reviewers – review now

vincentchalamon



lyrixx



stof



94noni



derrabus



dunglas



wouterj



chalasr



Assignees – assign yourself

Labels

Bug × Feature × Ready × Security ×

Status: Reviewed ×

Projects

None yet



SymfonyCon
DISNEYLAND PARIS
NOV.17-18.2022

Adding a feature to Symfony could take years



Let's meet AccessTokenAuthenticator in Symfony 6.2

AccessTokenAuthenticator

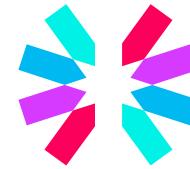
- Takes care of token extraction
 - Header
 - Query string
 - Request body
 - And/or your **Custom extractors**
- Calls **your Token Handler** to check the token (revocation, expiration, signature, ...)
- Custom success / failure handlers if needed



All this via configuration!

What kind of tokens could be used?

- JWT
- SAML2
- Biscuit
- Macaroons
- Homemade tokens (with chocolate chips and nuts 😊)
- ...
- Any kind of token, as it's up to you to handle them



Internally, in Symfony

- Extraction with `ChainTokenExtractor` (configurable order)
 - Default and custom extractors can be used at the same time
- Handle the token with **your** `TokenHandlerInterface` implementation
- `AccessTokenAuthenticator` will
 - create a `PostAuthenticationToken` object
 - set `WWW-Authenticate` Response header content in case of failure
 - use the configured User Provider in security.yaml



How much easier is it with 6.2?

6.1 security.yaml

```
1  security:
2    providers:
3      user_provider:
4        entity:
5          class: App\Entity\User
6          property: apiKey
7    firewalls:
8      api:
9        pattern: ^/api
10       lazy: true
11       provider: user_provider
12       custom_authenticator: App\Security\ApiKeyAuthenticator
```

6.1 ApiKeyAuthenticator

```
1 class ApiKeyAuthenticator extends AbstractAuthenticator
2 {
3     public function __construct(
4         private readonly UserRepository $userRepository,
5         private readonly string $env,
6     ) {}
7
8     public function supports(Request $req): ?bool
9     {
10        return $req->headers->has('Authorization') || $req->query->has('access_token') || $req->request->has('access_token');
11    }
12
13    public function authenticate(Request $req): Passport
14    {
15        if (null === $apiKey = $this->extractToken($req)) {
16            throw new AuthenticationException('No API Key provided');
17        }
18
19        // Here, it could be some logic to validate the token
20
21        return new SelfValidatingPassport(
22            new UserBadge($apiKey,
23                function (string $apiKey) {
24                    return $this->userRepository->findOneByApiKey($apiKey);
25                }
26            )
27        );
28    }
29 }
```

6.1 ApiKeyAuthenticator

```
1 class ApiKeyAuthenticator extends AbstractAuthenticator
2 {
3     public function __construct(
4         private readonly UserRepository $userRepository,
5         private readonly string $env,
6     ) {}
7
8     public function supports(Request $req): ?bool
9     {
10        return $req->headers->has('Authorization') || $req->query->has('access_token') || $req->request->has('access_token');
11    }
12
13    public function authenticate(Request $req): Passport
14    {
15        if (null === $apiKey = $this->extractToken($req)) {
16            throw new AuthenticationException('No API Key provided');
17        }
18
19        // Here, it could be some logic to validate the token
20
21        return new SelfValidatingPassport(
22            new UserBadge($apiKey,
23                function (string $apiKey) {
24                    return $this->userRepository->findOneByApiKey($apiKey);
25                }
26            )
27        );
28    }
29 }
```

6.1 ApiKeyAuthenticator

```
1 class ApiKeyAuthenticator extends AbstractAuthenticator
2 {
3     public function __construct(
4         private readonly UserRepository $userRepository,
5         private readonly string $env,
6     ) {}
7
8     public function supports(Request $req): ?bool
9     {
10         return $req->headers->has('Authorization') || $req->query->has('access_token') || $req->request->has('access_token');
11     }
12
13    public function authenticate(Request $req): Passport
14    {
15        if (null === $apiKey = $this->extractToken($req)) {
16            throw new AuthenticationException('No API Key provided');
17        }
18
19        // Here, it could be some logic to validate the token
20
21        return new SelfValidatingPassport(
22            new UserBadge($apiKey,
23                function (string $apiKey) {
24                    return $this->userRepository->findOneByApiKey($apiKey);
25                }
26            )
27        );
28    }
29 }
```

6.1 ApiKeyAuthenticator

```
1  /** Simplified version */
2  private function extractToken(Request $req): ?string
3  {
4      return match (true) {
5          // Header
6          $req->headers->has('Authorization') => str_replace('Bearer ', '', $req->headers->get('Authorization')),
7
8          // Query string
9          $req->query->has('access_token') => $req->query->get('access_token'),
10
11         // Request body
12         $req->request->has('access_token') => $req->request->get('access_token'),
13
14         default => null,
15     };
16 }
```

6.2 security.yaml

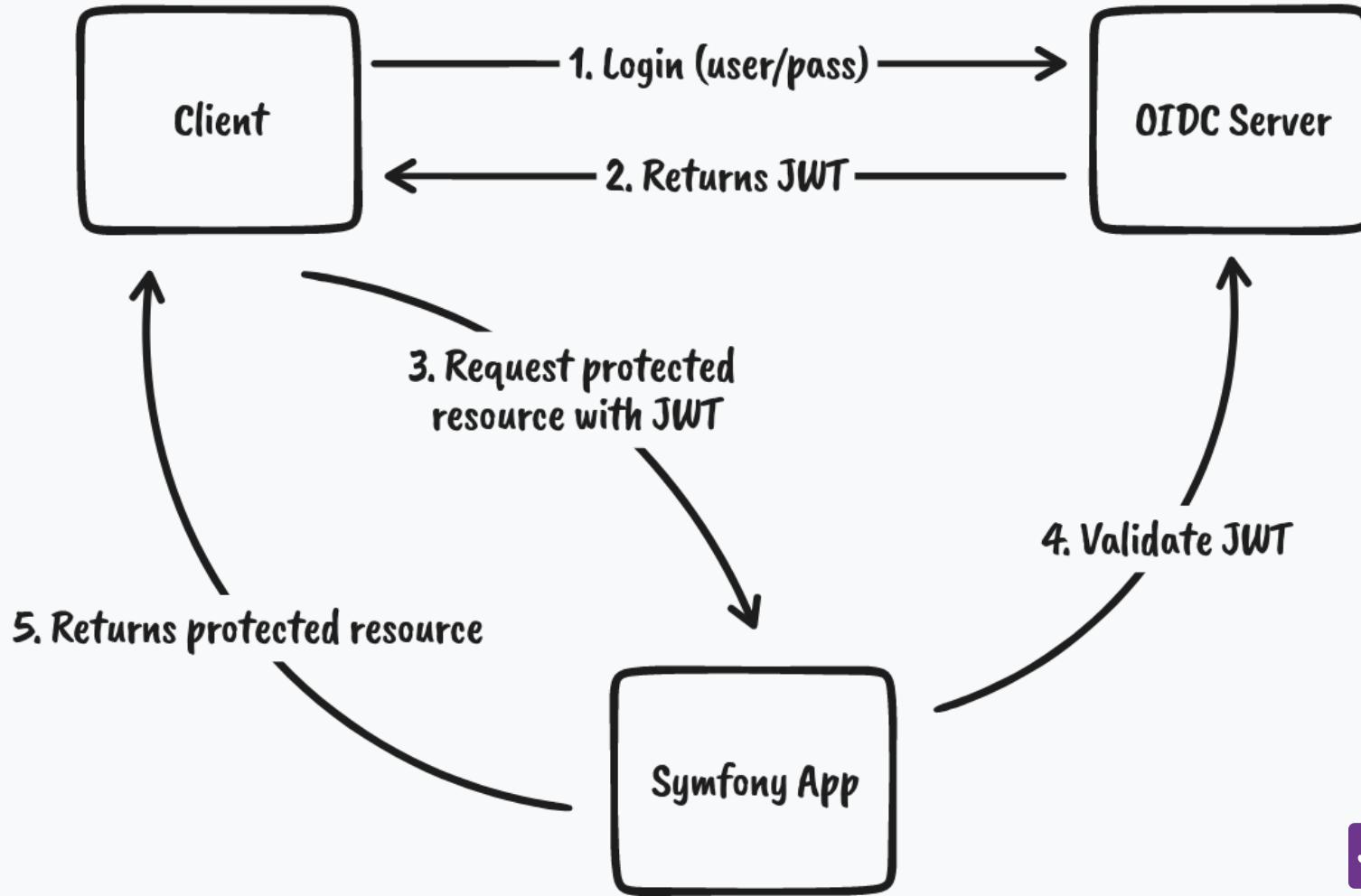
```
1  security:
2      providers:
3          user_provider:
4              entity:
5                  class: App\Entity\User
6                  property: apiKey
7      firewalls:
8          api:
9              pattern: ^/api
10             lazy: true
11             provider: user_provider
12             access_token:
13                 token_extractors:
14                     - header
15                     - query_string
16                     - request_body
17             token_handler: App\Security\AccessTokenHandler
```

6.2 AccessTokenHandler

```
1  class AccessTokenHandler implements AccessTokenHandlerInterface
2  {
3      public function __construct(
4          private readonly UserRepository $userRepository,
5          private readonly string $env,
6      ) {}
7
8      public function getUserIdentifierFrom(string $token): string
9      {
10         $user = $this->userRepository->findOneByApiKey($token);
11
12         if ($user === null) {
13             throw new BadCredentialsException('Invalid credentials.');
14         }
15
16         return $user->getUserIdentifier();
17     }
18 }
```



6.2 With a JWT issued by an OIDC server





6.2 With a JWT issued by an OIDC server

```
1  class JwtHandler implements AccessTokenHandlerInterface
2  {
3      public function __construct(
4          private readonly HttpClientInterface $oidcHttpClient,
5      ) {}
6
7      public function getUserIdentifierFrom(string $accessToken): string
8      {
9          try {
10              $userInfo = $this->oidcHttpClient->request('GET', 'protocol/openid-connect/userinfo', [
11                  'auth_bearer' => $accessToken,
12              ])->toArray();
13          } catch (HttpExceptionInterface $e) {
14              throw new BadCredentialsException($e->getMessage());
15          }
16
17          return $userInfo['email'];
18      }
19  }
```



6.2 With a JWT issued by your Symfony app

with [lcobucci/jwt](#)

(or [web-token/jwt-checker](#))



6.2 security.yaml

```
1  security:
2    providers:
3      user_provider:
4        entity:
5          class: App\Entity\User
6          property: email
7    firewalls:
8      api:
9        pattern: ^/api
10       lazy: true
11       provider: user_provider
12       access_token:
13         token_extractors: header
14         token_handler: App\Security\JwtHandler
```

6.2 JwtHandler

```
1 class JwtHandler implements AccessTokenHandlerInterface
2 {
3     public function __construct(
4         private readonly Parser $jwtParser = new Parser(new JoseEncoder()),
5         private readonly Validator $jwtValidator = new Validator(),
6     ) {}
7
8     public function getUserIdentifierFrom(string $accessToken): string
9     {
10         $jwt = $this->jwtParser->parse($accessToken);
11         $timezone = new \DateTimeZone('Europe/Paris');
12
13         try {
14             $this->jwtValidator->assert($jwt, new ValidAt(new SystemClock($timezone)));
15             $this->jwtValidator->assert($jwt, new SignedWith(
16                 new Sha256(),
17                 InMemory::plainText('PRIVATE-KEY')
18             ));
19         } catch (RequiredConstraintsViolated $e) {
20             throw new BadCredentialsException($e->getMessage());
21         }
22
23         return $jwt->claims()->get(RegisteredClaims::SUBJECT);
24     }
25 }
```

API Platform plans to support OpenID Connect authentication



<https://github.com/api-platform/demo/pull/265>

api-platform / demo Public

Watch

Code Issues 1 Pull requests 1 Actions

Add Keycloak authentication #265

Draft main ← test/keycloak

Conversation 32 Commits 2 Checks 0 Files changed 57 +1,331 -985

vincentchalamon (Vincent) on Apr 13 • edited

Contributor

Use AccessTokenAuthenticator

Validate JWT on Keycloak (requires Keycloak to run separately)

Add functional tests

Configure Keycloak Helm Chart (keycloak-pr-265-demo.api-platform.com)

Configure Keycloak `demo` realm on Helm Chart

Fix admin authentication (waiting for [marmelab/ra-keycloak](#))

Update dependencies (waiting for Symfony 6.2 stable in november)

Reviewers – review now

dunglas chalasr jfcoz GregoireHebert

Assignees

vincentchalamon

SymfonyCon DISNEYLAND PARIS NOV.17-18.2022



In the future?

- Add a native JwtHandler to Symfony?
- Add a native SamlHandler to Symfony?
- Add a native BiscuitHandler to Symfony?

👉 It's up to the community!

Your responsibility

Symfony's responsibility,
but customizable

Access Token Authentication Process

<= 6.1



6.2

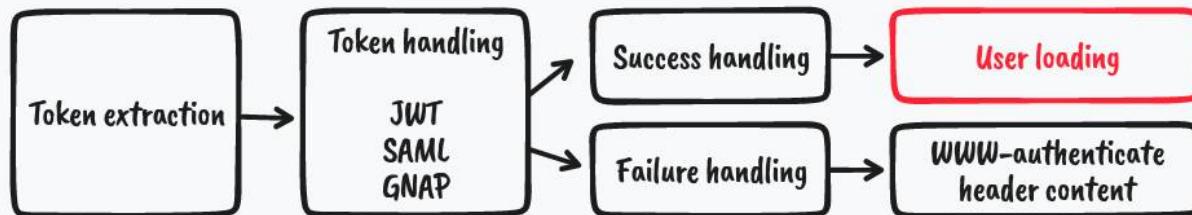


Your responsibility

Symfony's responsibility,
but customizable

Access Token Authentication Process

6.3 ?



Less responsibility, less code

- **Configure** the way the extraction should be done
- **Focus** on the token processing
 - Decoding
 - Checking signature, expiration, revocation
 - Retrieve user identifier



Leverage all Symfony power to fine tune configuration to your needs



Thanks a lot

**Wouter
Guillaume
Vincent
Florent**

And all reviewers, commenters A small emoji of a colorful party hat with streamers.

Thank you



Any questions?

Slides and demo apps  welcomattic.github.io/painless-authentication-with-access-token

Sources

- [JWT RFC](#)
- [Bearer Token Usage RFC](#)
- [In-depth article about token authentication](#)