Painless Authentication

with Access Tokens

Mathieu Santostefano

Web developer at <joliCode/>

Symfony Core Team Member

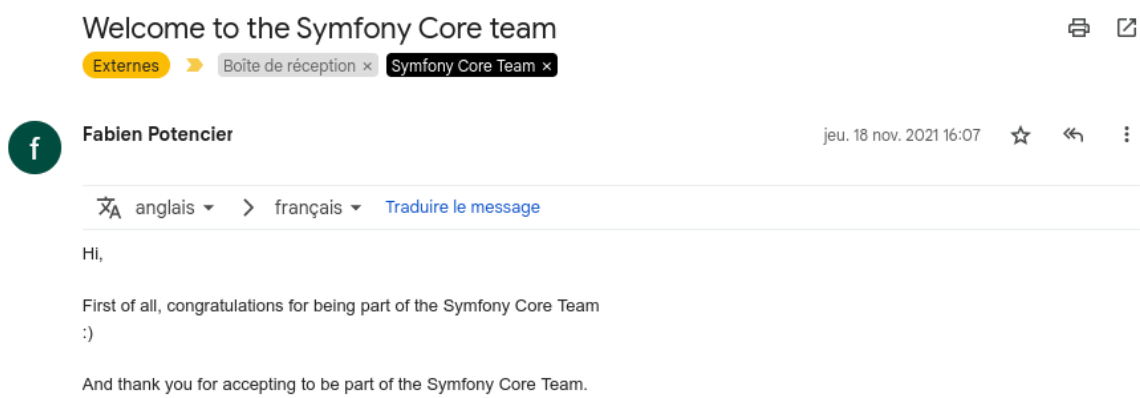
Twitter @welcomattic

GitHub @welcomattic

SymfonyCon
DISNEYLAND PARIS
NOV 17-18 2022

Photo by Kelly Sikkema on Unsplash

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



Welcome to the Symfony Core team

Externes · Boîte de réception × · Symfony Core Team ×

Fabien Potencier jeu. 18 nov. 2021 16:07

anglais › français Traduire le message

Hi,

First of all, congratulations for being part of the Symfony Core Team
:)

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

🍽 On the menu

Photo by Nathan Dumlao on Unsplash

🍽 On the menu

1. Access token, **what is it?**

SymfonyCon
DISNEYLAND PARIS
NOV. 17-18 2022

Photo by Nathan Dumlao on Unsplash

🍽 On the menu

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

Photo by Nathan Dumlao on Unsplash

🍽 On the menu

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

SymfonyCon
DISNEYLAND PARIS
NOV. 17-18 2022

Photo by Nathan Dumlao on Unsplash

🍽 On the menu

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

SymfonyCon
DISNEYLAND PARIS
NOV. 17-18 2022

Photo by Nathan Dumlao on Unsplash

🍽 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

SymfonyCon
DISNEYLAND PARIS
NOV. 17-18 2022

Photo by Nathan Dumlao on Unsplash

🍽 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?

SymfonyCon
DISNEYLAND PARIS
NOV. 17-18 2022

Photo by Nathan Dumlao on Unsplash

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

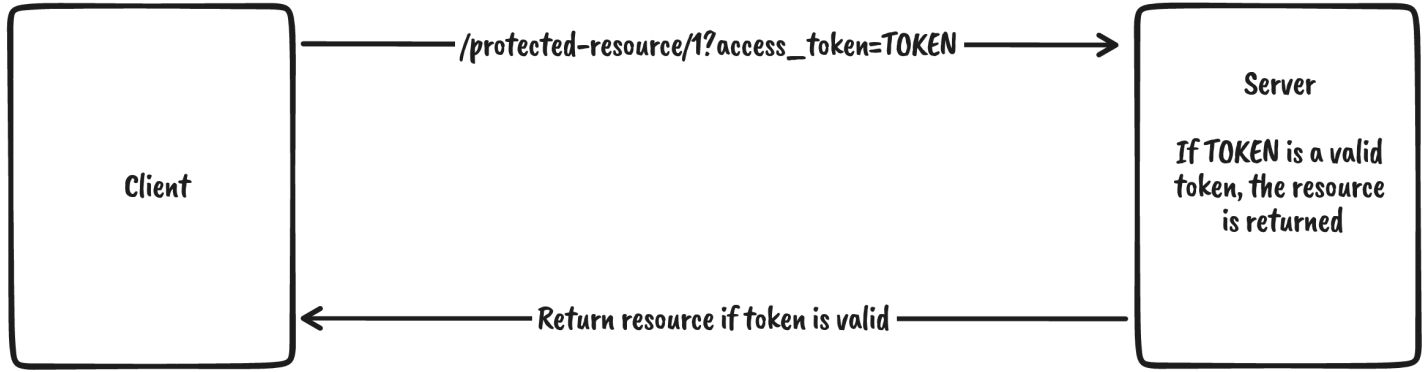
- Stateless user login
- **API Gateway** in front of private APIs
- Application that access to personal data provided by a third party
- **Micro-services** between them
- Client applications of a **SaaS API**
- ...

🤨 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?

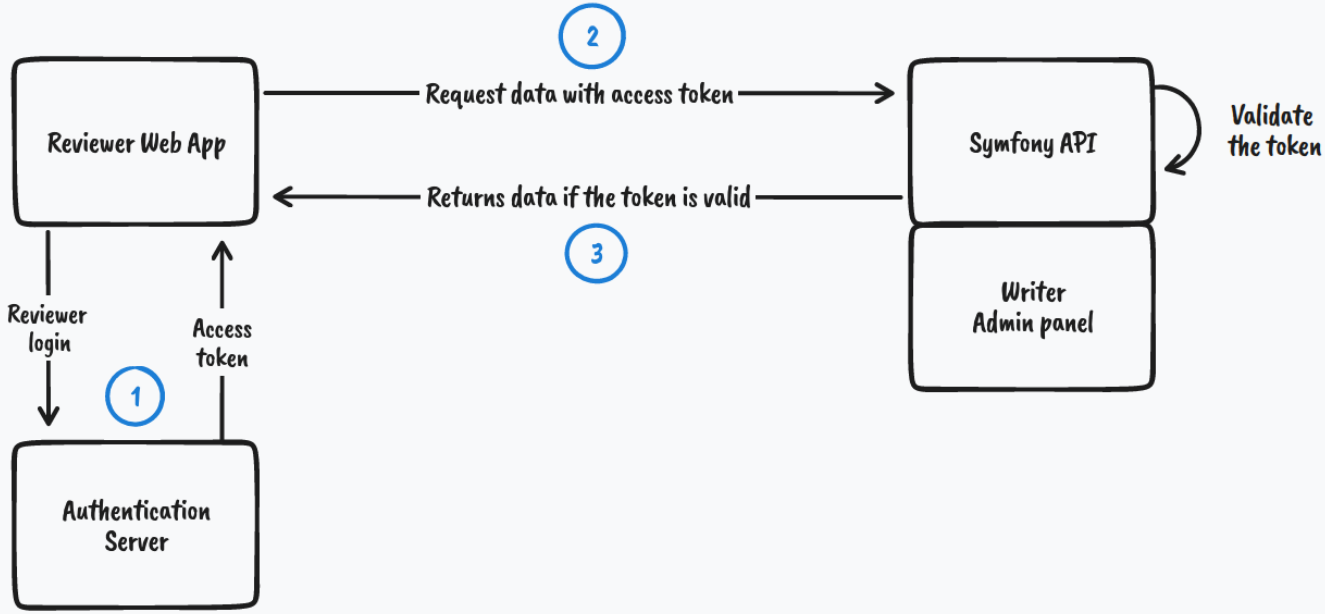
The validation process is up to you

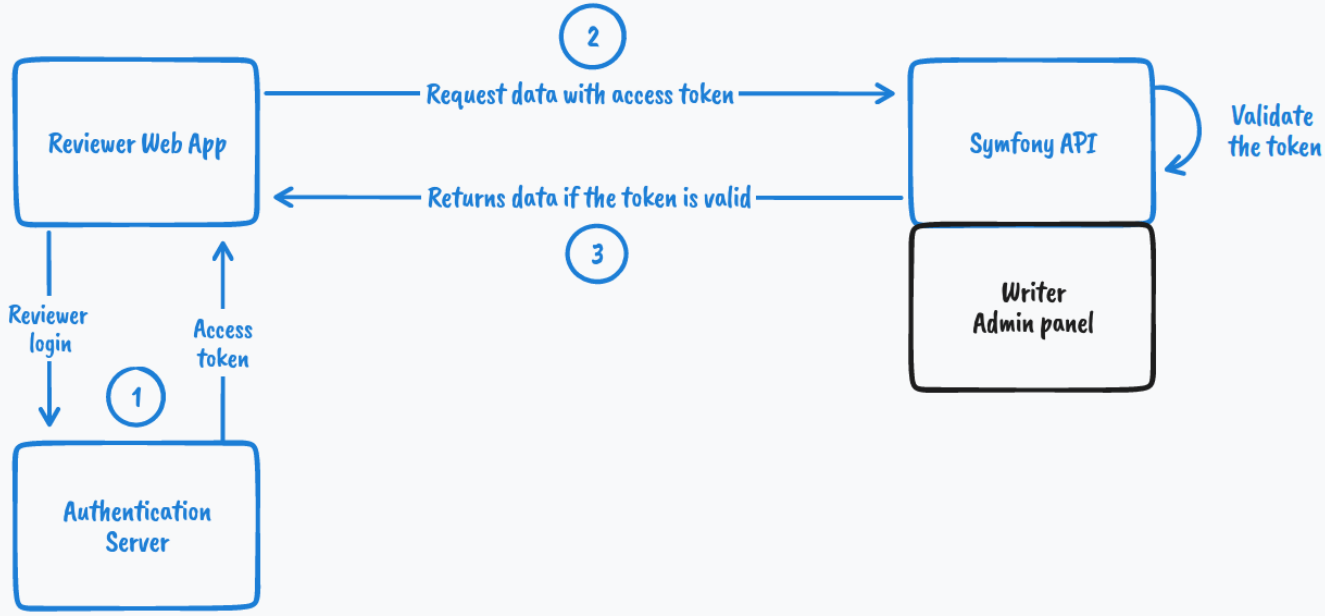
- Check if the string is **present** in a database
- **Compute a hash** and compare it to the expected one
- **Decode the token** (base64, ...) and make **assertions** on decoded values
- Ensure the **expiration date** is not passed if needed
- Check if the token has been **revoked** or not
- Call an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

```php
/** Simplified version */
private function extractToken(Request $req): ?string
{
    return match (true) {
        // Header
        $req->headers->has('Authorization') => str_replace('Bearer ', '', $req->headers->get('Authorization')),

        // Query string
        $req->query->has('access_token') => $req->query->get('access_token'),

        // Request body
        $req->request->has('access_token') => $req->request->get('access_token'),

        default => null,
    };
}
```

👨‍💻 Some code

```php
public function authenticate(Request $req): Passport
{
    if (null === $apiKey = $this->extractToken($req)) {
        throw new AuthenticationException('No API Key provided');
    }

    // Here, it could be some logic to validate the token

    return new SelfValidatingPassport(
        new UserBadge($apiKey,
            function (string $apiKey) {
                return $this->userRepository->findOneByApiKey($apiKey);
            }
        )
    );
}
```

🥱 Boring code

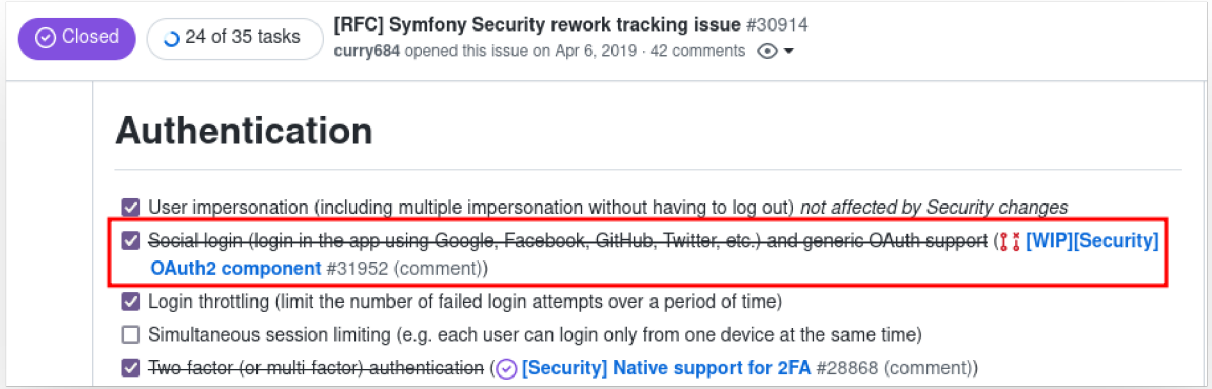
- We have to repeat this code in all our applications, **boring**
- Our responsibility to respect RFC6750, **boring**
- No body likes boring code
- Boring code is code we rewrite in all projects, no business value
- Poor **D**eveloper e**X**perience, Symfony tends to improve DX

This is definitely improvable


SymfonyCon
DISNEYLAND PARIS
NOV 17-18 2022

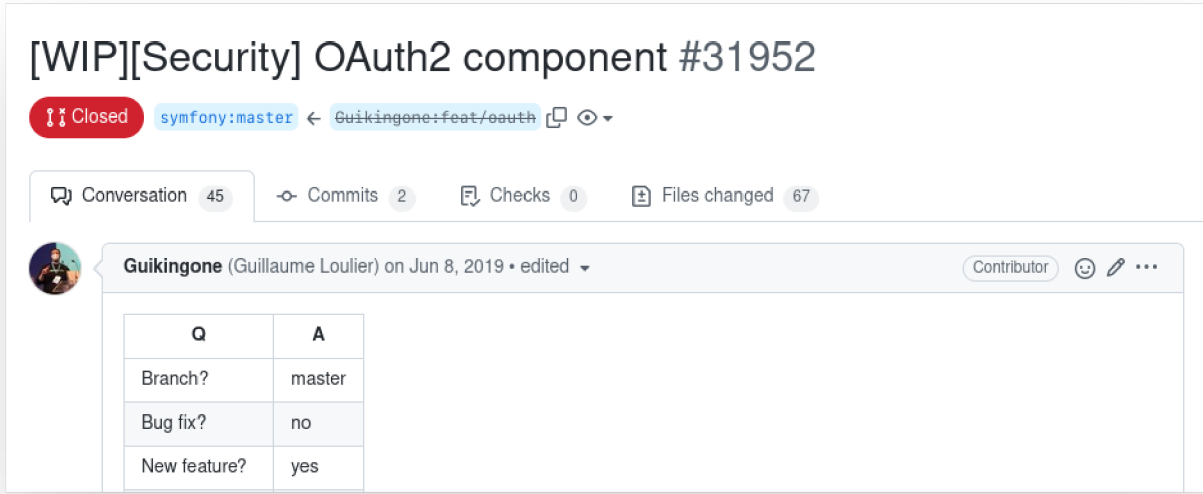
Discussions started long time ago

- **April 2019**, at EU FOSSA Hackathon



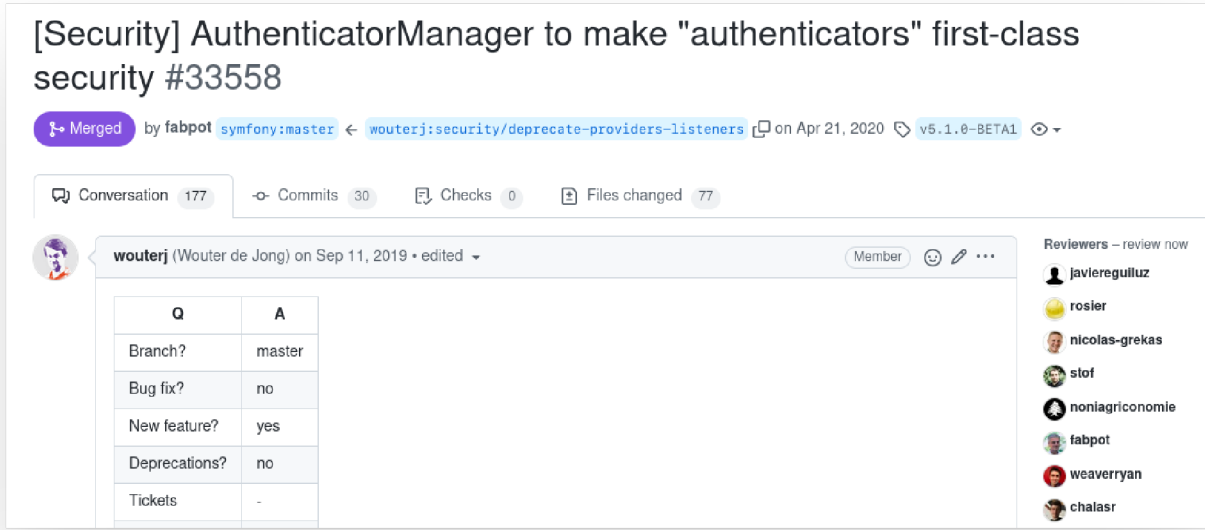
Discussions started long time ago

- **June 2019**, first PR about OAuth2 Component. ❌ Aborted



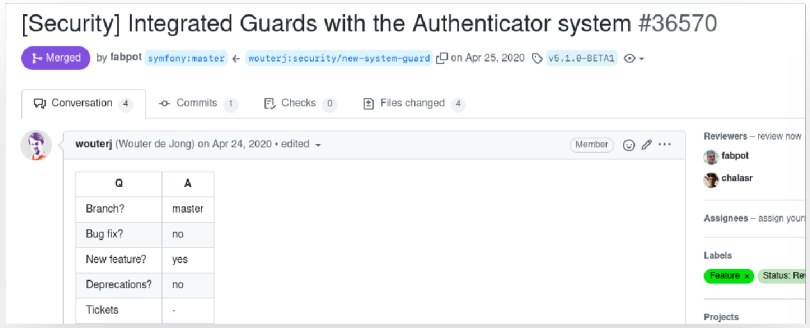
Discussions started long time ago

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



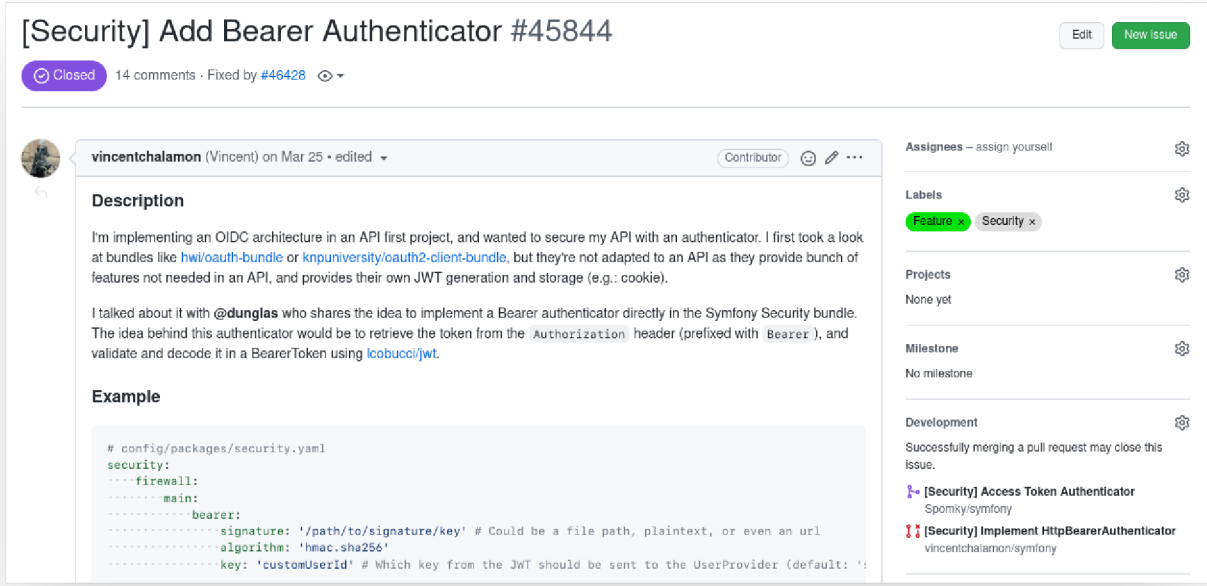
Discussions started long time ago

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



Discussions started long time ago

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



[Security] Add Bearer Authenticator #45844

Edit | New issue

Closed | 14 comments · Fixed by #46428

vincentchalamon (Vincent) on Mar 25 • edited — Contributor

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 knpuniversity/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 lcobucci/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 × Security ×

Projects: None yet

Milestone: No milestone

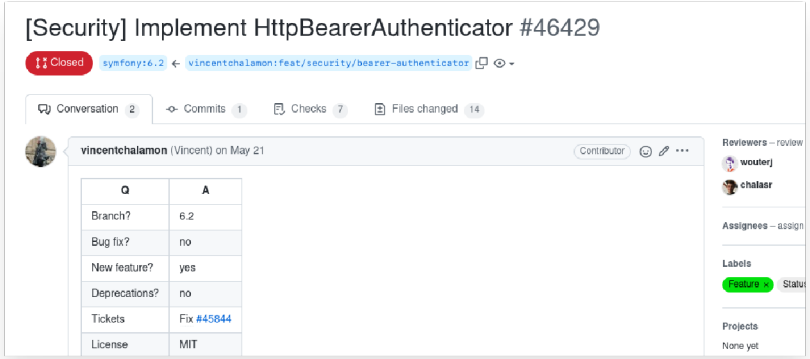
Development: Successfully merging a pull request may close this issue.

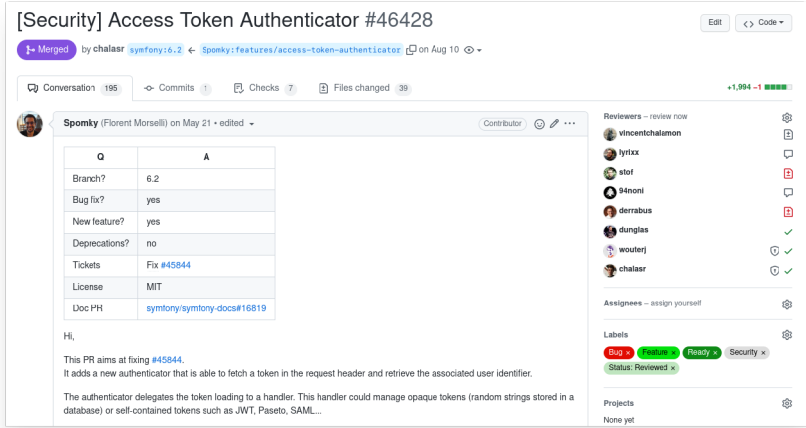
[Security] Access Token Authenticator
Spomky/symfony

[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





SymfonyCon
DISNEYLAND PARIS
NOV. 17-18 2022

🤩 Thanks a lot Florent Morselli  @Spomky

[Security] Access Token Authenticator #46428

Edit | Code

Merged by chalasr `symfony:6.2` ← `Spomky:features/access-token-authenticator` on Aug 10

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

```yaml
security:
  providers:
    user_provider:
      entity:
        class: App\Entity\User
        property: apiKey
  firewalls:
    api:
      pattern: ^/api
      lazy: true
      provider: user_provider
      custom_authenticator: App\Security\ApiKeyAuthenticator
```

6.1 ApiKeyAuthenticator

```php
class ApiKeyAuthenticator extends AbstractAuthenticator
{
    public function __construct(
        private readonly UserRepository $userRepository,
        private readonly string $env,
    ) {}

    public function supports(Request $req): ?bool
    {
        return $req->headers->has('Authorization') || $req->query->has('access_token') || $req->request->has('access_token');
    }

    public function authenticate(Request $req): Passport
    {
        if (null === $apiKey = $this->extractToken($req)) {
            throw new AuthenticationException('No API Key provided');
        }

        // Here, it could be some logic to validate the token

        return new SelfValidatingPassport(
            new UserBadge($apiKey,
                function (string $apiKey) {
                    return $this->userRepository->findOneByApiKey($apiKey);
                }
            )
        );
    }
}
```

6.1 ApiKeyAuthenticator

```php
class ApiKeyAuthenticator extends AbstractAuthenticator
{
    public function __construct(
        private readonly UserRepository $userRepository,
        private readonly string $env,
    ) {}

    public function supports(Request $req): ?bool
    {
        return $req->headers->has('Authorization') || $req->query->has('access_token') || $req->request->has('access_token');
    }

    public function authenticate(Request $req): Passport
    {
        if (null === $apiKey = $this->extractToken($req)) {
            throw new AuthenticationException('No API Key provided');
        }

        // Here, it could be some logic to validate the token

        return new SelfValidatingPassport(
            new UserBadge($apiKey,
                function (string $apiKey) {
                    return $this->userRepository->findOneByApiKey($apiKey);
                }
            )
        );
    }
}
```

6.1 ApiKeyAuthenticator

```php
class ApiKeyAuthenticator extends AbstractAuthenticator
{
    public function __construct(
        private readonly UserRepository $userRepository,
        private readonly string $env,
    ) {}

    public function supports(Request $req): ?bool
    {
        return $req->headers->has('Authorization') || $req->query->has('access_token') || $req->request->has('access_token');
    }

    public function authenticate(Request $req): Passport
    {
        if (null === $apiKey = $this->extractToken($req)) {
            throw new AuthenticationException('No API Key provided');
        }

        // Here, it could be some logic to validate the token

        return new SelfValidatingPassport(
            new UserBadge($apiKey,
                function (string $apiKey) {
                    return $this->userRepository->findOneByApiKey($apiKey);
                }
            )
        );
    }
}
```

6.1 ApiKeyAuthenticator

```php
/** Simplified version */
private function extractToken(Request $req): ?string
{
    return match (true) {
        // Header
        $req->headers->has('Authorization') => str_replace('Bearer ', '', $req->headers->get('Authorization')),

        // Query string
        $req->query->has('access_token') => $req->query->get('access_token'),

        // Request body
        $req->request->has('access_token') => $req->request->get('access_token'),

        default => null,
    };
}
```

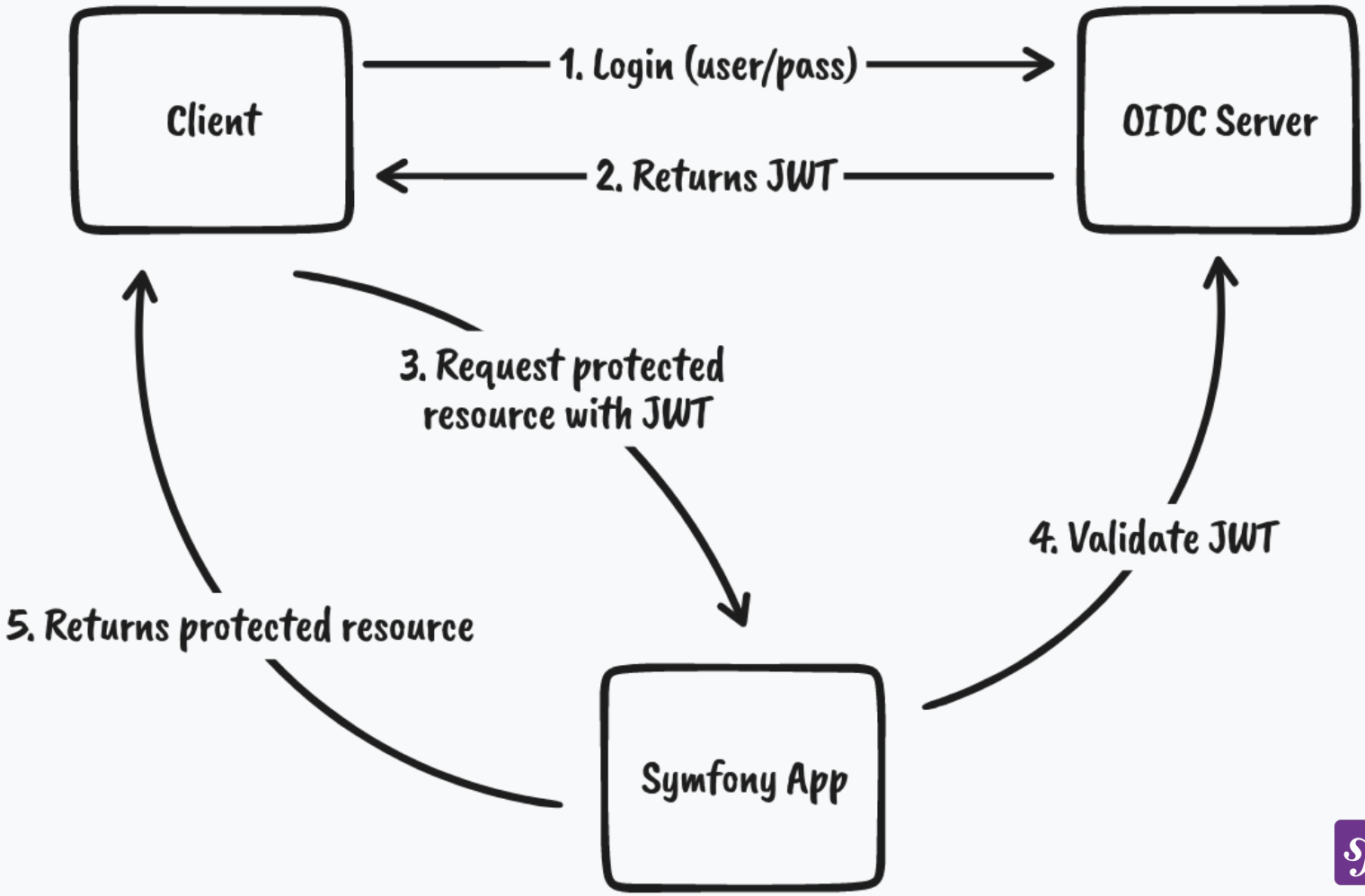
6.2 security.yaml

```yaml
security:
  providers:
    user_provider:
      entity:
        class: App\Entity\User
        property: apiKey
  firewalls:
    api:
      pattern: ^/api
      lazy: true
      provider: user_provider
      access_token:
        token_extractors:
          - header
          - query_string
          - request_body
        token_handler: App\Security\AccessTokenHandler
```

6.2 AccessTokenHandler

```php
class AccessTokenHandler implements AccessTokenHandlerInterface
{
    public function __construct(
        private readonly UserRepository $userRepository,
        private readonly string $env,
    ) {}

    public function getUserIdentifierFrom(string $token): string
    {
        $user = $this->userRepository->findOneByApiKey($token);

        if ($user === null) {
            throw new BadCredentialsException('Invalid credentials.');
        }

        return $user->getUserIdentifier();
    }
}
```

6.2 With a JWT issued by an OIDC server


Client
1. Login (user/pass)
OIDC Server
2. Returns JWT
3. Request protected resource with JWT
4. Validate JWT
5. Returns protected resource
Symfony App

6.2 With a JWT issued by an OIDC server

```php
class JwtHandler implements AccessTokenHandlerInterface
{
    public function __construct(
        private readonly HttpClientInterface $oidcHttpClient,
    ) {}

    public function getUserIdentifierFrom(string $accessToken): string
    {
        try {
            $userInfo = $this->oidcHttpClient->request('GET', 'protocol/openid-connect/userinfo', [
                'auth_bearer' => $accessToken,
            ])->toArray();
        } catch (HttpExceptionInterface $e) {
            throw new BadCredentialsException($e->getMessage());
        }

        return $userInfo['email'];
    }
}
```

 6.2 With a JWT issued by your Symfony app

with lcobucci/jwt

(or web-token/jwt-checker)

6.2 security.yaml

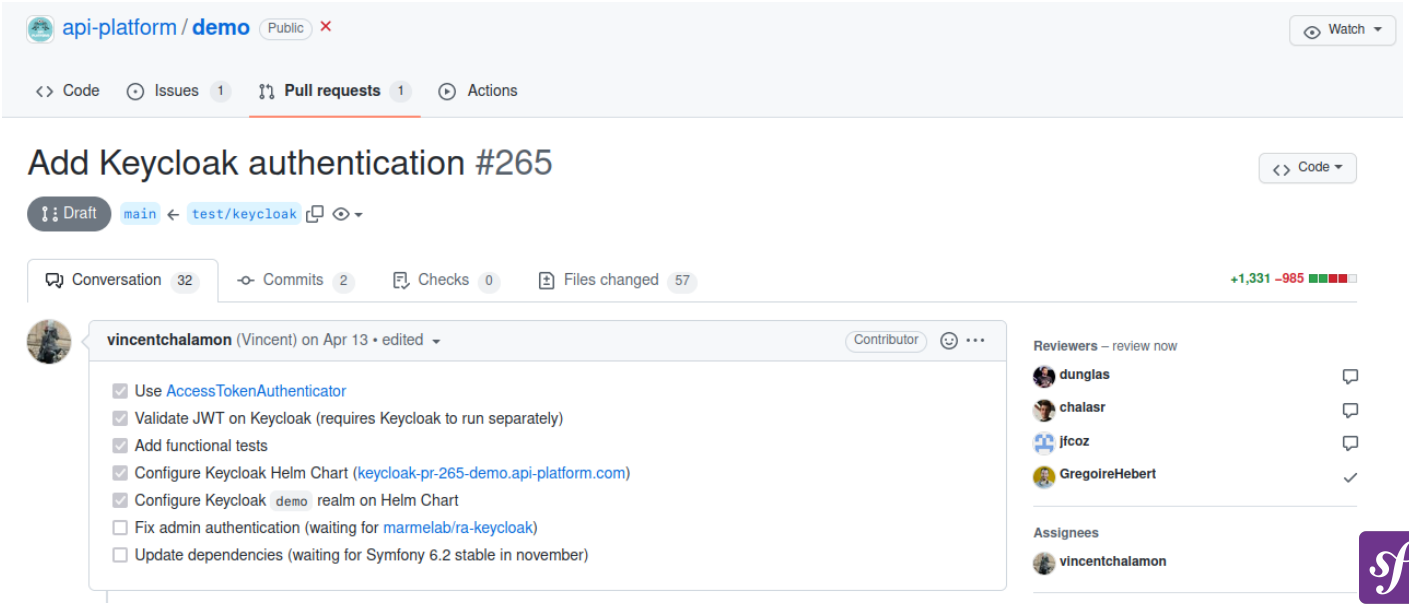
```yaml
security:
  providers:
    user_provider:
      entity:
        class: App\Entity\User
        property: email
  firewalls:
    api:
      pattern: ^/api
      lazy: true
      provider: user_provider
      access_token:
        token_extractors: header
        token_handler: App\Security\JwtHandler
```

6.2 JwtHandler

```php
class JwtHandler implements AccessTokenHandlerInterface
{
    public function __construct(
        private readonly Parser $jwtParser = new Parser(new JoseEncoder()),
        private readonly Validator $jwtValidator = new Validator(),
    ) {}

    public function getUserIdentifierFrom(string $accessToken): string
    {
        $jwt = $this->jwtParser->parse($accessToken);
        $timezone = new \DateTimeZone('Europe/Paris');

        try {
            $this->jwtValidator->assert($jwt, new ValidAt(new SystemClock($timezone)));
            $this->jwtValidator->assert($jwt, new SignedWith(
                new Sha256(),
                InMemory::plainText('PRIVATE-KEY')
            ));
        } catch (RequiredConstraintsViolated $e) {
            throw new BadCredentialsException($e->getMessage());
        }

        return $jwt->claims()->get(RegisteredClaims::SUBJECT);
    }
}
```

API Platform plans to support OpenID Connect authentication

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



💫 In the future?

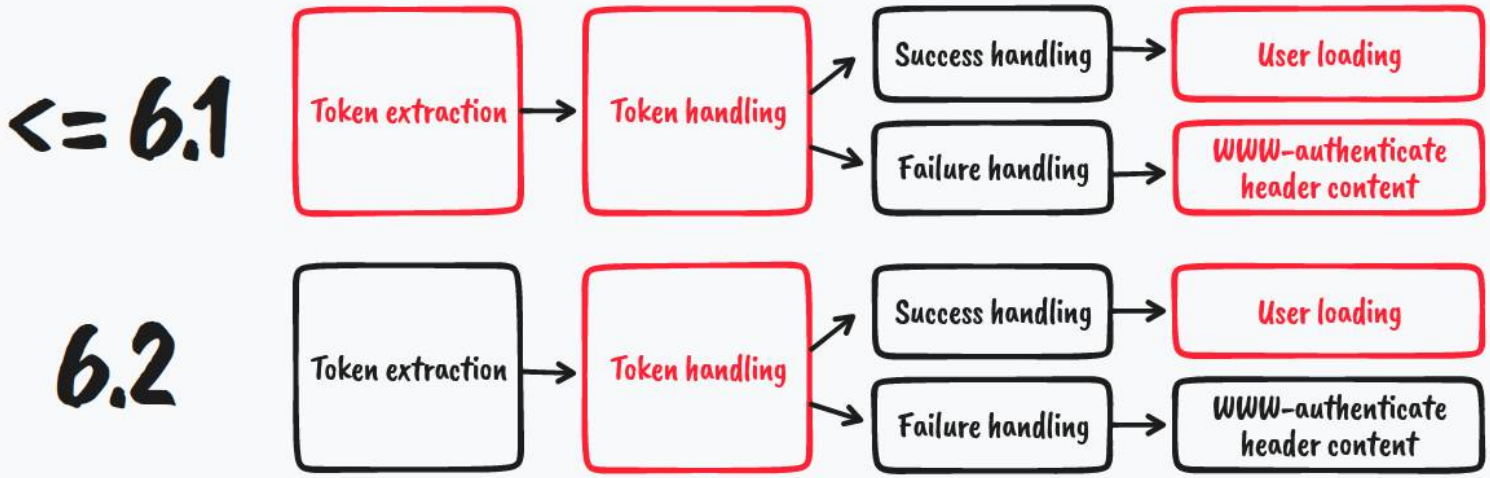
- Add a native JwtHandler to Symfony?
- Add a native SamlHandler to Symfony?
- Add a native BiscuitHandler to Symfony?

> 👉 It's up to the community!

Your responsibility

Symfony's responsibility, but customizable

Access Token Authentication Process

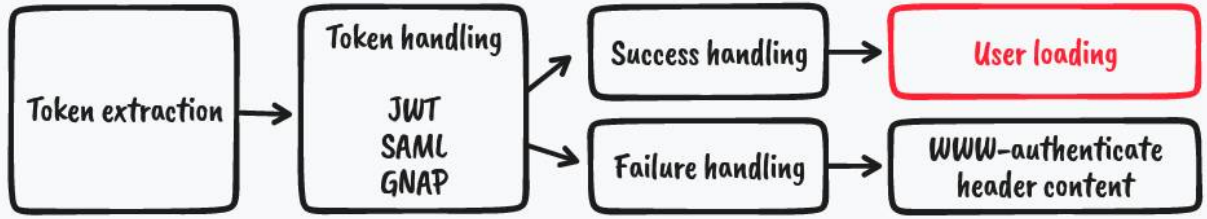


Your responsibility

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 🎉

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