Authentication and Authorization in PHP

JSON WEB TOKENS

Matthew Setter
PHP SECURITY ENGINEER

@settermjd matthewsetter.com

Module Overview

- Essentials of JSON Web Tokens
- What they are
- How they work
- Advantages and disadvantages
- How to implement them in PHP
- Some security considerations

What are JSON Web Tokens?

“A **compact** and **self-contained** way for **securely transmitting information** between parties as a JSON object. This information can be **verified** and **trusted** because it is **digitally signed** (and can be encrypted). JWTs can be signed using a secret (with the HMAC algorithm) or a public/private key pair using RSA or ECDSA.”

JSON Web Tokens - jwt.io

“Signed tokens can verify the integrity of the claims contained within it, while encrypted tokens hide those claims from other parties. When tokens are signed using public/private key pairs, the signature also certifies that only the party holding the private key is the one that signed it.”

JSON Web Tokens - jwt.io

JWTs

Composed of three parts:

- A header
- A payload
- A signature

JWT Example

eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpc3MiOiJodHRwczpcL1wvbG9jYWxkb21haW4uZGV2IiwiYXVkIjoiaHR0cHM6XC9cL2xvY2FsZG9tYWluLmRldiIsImlhdCI6MTYwMTU4MDcwOSwibmJmIjoxNjAxNTgwNzY5LCJzdWIiOiJhZG1pbiJ9.qvVcdbKmmXJxEqz4xbrAcOT2KbAtBi3GQbfexL713eM

```
{
  "alg": "HS256",
  "typ": "JWT"
}
```

The JWT Header

Consists of two parts:

- **The signing algorithm**
- **The token’s type**

```
{
  "iss": "settermjd",
  "sub": "login",
  "nbf": 1601296828,
  "exp": 1602592985,
  "name": "Tammy Green",
  "nickname": "Tam",
  "address": "210 Queen St, Brisbane, Queensland, Australia, 4006",
  "updated_at": 1600951534
}
```

The JWT Payload

- A JSON string containing “claims”, which are:
  - Statements about an entity
  - Additional data
- Claims can be registered, public, and private

Registered Claims

Claim	Description
aud	This determines who the JWT is intended for, or its audience.

Registered Claims

Claim	Description
aud	This determines who the JWT is intended for
exp	This determines the JWT’s expiration time

Registered Claims

Claim	Description
aud	This determines who the JWT is intended for
exp	This determines when the JWT expires
iat	This determines when the JWT was issued

Registered Claims

Claim	Description
aud	This determines who the JWT is intended for
exp	This determines when the JWT expires
iat	This determines when the JWT was issued
iss	This determines who issued a JWT

Registered Claims

Claim	Description
aud	This determines who the JWT is intended for
exp	This determines when the JWT expires
iat	This determines when the JWT was issued
iss	This determines who issued a JWT
jti	This is the JWT's unique id

Registered Claims

Claim	Description
aud	This determines who the JWT is intended for
exp	This determines when the JWT expires
iat	This determines when the JWT was issued
iss	This determines who issued a JWT
jti	This is the JWT's unique id
nbf	This determines the date/time before which the JWT cannot be accepted or is not valid.

Registered Claims

Claim	Description
aud	This determines who the JWT is intended for
exp	This determines when the JWT expires
iat	This determines when the JWT was issued
iss	This determines who issued a JWT
jti	This is the JWT's unique id
nbf	This determines the date/time before which the JWT cannot be accepted or is not valid.
sub	This determines the subject of a JWT

Public Claims

- Similar to registered claims
- Can be defined at will
- Be careful when creating public claims

Private Claims

- Agreed upon between a JWT producer and a JWT consumer
- Neither registered nor public

```
{
  "iss": "settermjd",
  "sub": "login",
  "nbf": 1601296828,
  "exp": 1602592985,
  "name": "Tammy Green",
  "nickname": "Tam",
  "address": "210 Queen St, Brisbane, Queensland, Australia, 4006",
  "updated_at": 1600951534
}
```

Example Payload

- Agreed upon between a JWT producer and a JWT consumer
- Neither registered nor public

```
{
  "iss": "settermjd",
  "sub": "login",
  "nbf": 1601296828,
  "exp": 1602592985,
  "name": "Tammy Green",
  "nickname": "Tam",
  "address": "210 Queen St, Brisbane, Queensland, Australia, 4006",
  "updated_at": 1600951534
}
```

Example Payload

- Agreed upon between a JWT producer and a JWT consumer
- Neither registered nor public

```
{
  "iss": "settermjd",
  "sub": "login",
  "nbf": 1601296828,
  "exp": 1602592985,
  "name": "Tammy Green",
  "nickname": "Tam",
  "address": "210 Queen St, Brisbane, Queensland, Australia, 4006",
  "updated_at": 1600951534
}
```

Example Payload

- Agreed upon between a JWT producer and a JWT consumer
- Neither registered nor public

```
{
  "iss": "settermjd",
  "sub": "login",
  "nbf": 1601296828,
  "exp": 1602592985,
  "name": "Tammy Green",
  "nickname": "Tam",
  "address": "210 Queen St, Brisbane, Queensland, Australia, 4006",
  "updated_at": 1600951534
}
```

Example Payload

- Agreed upon between a JWT producer and a JWT consumer
- Neither registered nor public

```
{
  "iss": "settermjd",
  "sub": "login",
  "nbf": 1601296828,
  "exp": 1602592985,
  "name": "Tammy Green",
  "nickname": "Tam",
  "address": "210 Queen St, Brisbane, Queensland, Australia, 4006",
  "updated_at": 1600951534
}
```

Example Payload

- Agreed upon between a JWT producer and a JWT consumer
- Neither registered nor public

The JWT Signature

A combination of:

- The JWT header
- The JWT payload
- A secret value

The JWT Signature

- Digitally signed
- The signature ensures:
  - The JWTs integrity
  - That it has not been tampered with
- Algorithm can be symmetric or asymmetric

Symmetric Algorithms

- Uses a single, shared key
- They protect against manipulation
- Must only be known by producer and the consumer

Asymmetric Algorithms

- Private keys sign the token
- Public keys verify the token
- Used when:
  - Shared secrets are impractical
  - Others need to verify the token

Quick Recap

- JWT essentials
- Composed of three parts
- Can be used with symmetric and asymmetric algorithms
- Quite complex

Up Next:
Advantages and Disadvantages

Advantages and Disadvantages

Advantages

- Lightweight authorization method
- Single-use authorization token

Disadvantages

- Can add weight to requests
- Can be complicated to understand
- Can be a challenge to revoke them

Quick Recap

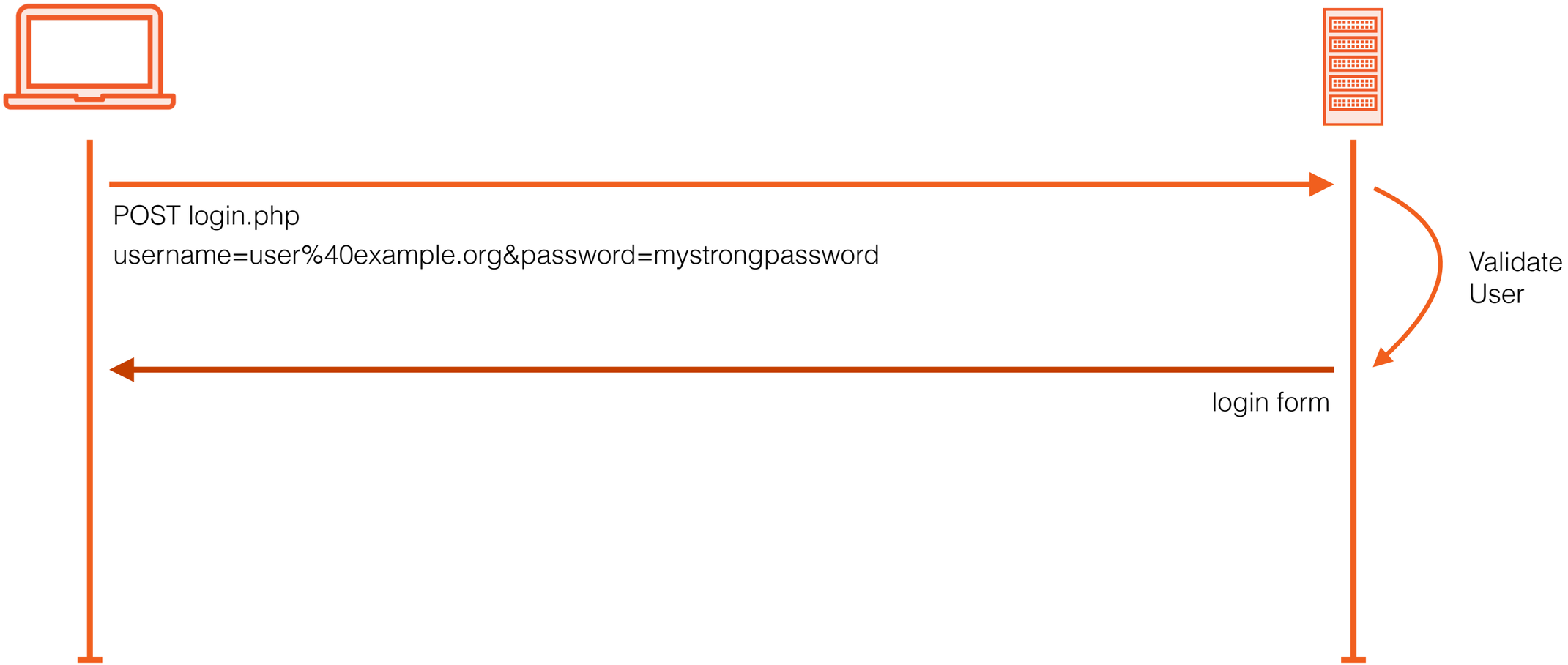
- Advantages and disadvantages

Up Next:
Implement JSON Web Tokens in PHP

Implement JSON Web Tokens in PHP

Demo Application Overview

Demo Application Overview



Demo Application Overview

Demo Application Overview

Module Recap

Module Recap

- Learned about JSON Web Tokens
- What they are
- How they work
- Advantages and disadvantages
- How to implement them in PHP

Up Next:
Introduction to Privacy Legislation