

Authentication and Authorization in PHP

FORM-BASED AUTHENTICATION



Matthew Setter

PHP SECURITY ENGINEER

@settermjd matthewsetter.com

Module Overview

- Form-based authentication
- What it is
- How it works
- How to implement it in PHP
- Advantages and disadvantages
- Security considerations

What is Form-based Authentication?

There is no formalized
specification

Form-based Authentication

```
<form method="POST" action="/login">  
  username: <input type="text" name="username" required>  
  password: <input type="password" name="password" required>  
  <input name="__csrf" type="hidden" value="6fGBtLZmVBZ59oy">  
  <button type="submit">Login</button>  
</form>
```

```
<form method="POST" action="/login">  
  username: <input type="text" name="username" required>  
  password: <input type="password" name="password" required>  
  <input name="verification" type="hidden" value="6fGBtLZmVBZ59oy">  
  <button type="submit">Login</button>  
</form>
```

Login Forms Contain

```
<form method="POST" action="/login">  
  username: <input type="text" name="username" required>  
  password: <input type="password" name="password" required>  
  <input name="verification" type="hidden" value="6fGBtLZmVBZ59oy">  
  <button type="submit">Login</button>  
</form>
```

Login Forms Contain

A text field for the username

```
<form method="POST" action="/login">  
  username: <input type="text" name="username" required>  
  password: <input type="password" name="password" required>  
  <input name="verification" type="hidden" value="6fGBtLZmVBZ59oy">  
  <button type="submit">Login</button>  
</form>
```

Login Forms Contain

A text field for the username

A password field for the password


```
<form method="POST" action="/login">  
  username: <input type="text" name="username" required>  
  password: <input type="password" name="password" required>  
  <input name="verification" type="hidden" value="6fGBtLZmVBZ59oy">  
  <button type="submit">Login</button>  
</form>
```

Login Forms Contain

A text field for the username

A password field for the password

A hidden field for a CSRF check

“Cross-Site Request Forgery (CSRF) is an attack that forces an end user to execute unwanted actions on a web application in which they’re currently authenticated. With a little help of social engineering an attacker may trick the users of a web application into executing actions of the attacker’s choosing.”

Cross-Site Request Forgery (CSRF) - OWASP

“If the victim is a normal user, a successful CSRF attack can force the user to perform state changing requests like transferring funds, changing their email address, and so forth. If the victim is an administrative account, CSRF can compromise the entire web application.”

Cross-Site Request Forgery (CSRF) - OWASP

```
<form method="POST" action="/login">  
  username: <input type="text" name="username" required>  
  password: <input type="password" name="password" required>  
  <input name="verification" type="hidden" value="6fGBtLZmVBZ59oy">  
  <button type="submit">Login</button>  
</form>
```

Login Forms Contain

A text field for the username

A password field for the password

A hidden field for a CSRF check

A button to submit the form

```
<form method="POST" action="/login">  
  username: <input type="text" name="username" required>  
  password: <input type="password" name="password" required>  
  <input name="verification" type="hidden" value="6fGBtLZmVBZ59oy">  
  <button type="submit">Login</button>  
</form>
```

Login Forms Contain

A text field for the username

A password field for the password

A hidden field for a CSRF check

A button to submit the form

The form tag containing the form method and form action

```
<form method="POST" action="/login">  
  username: <input type="text" name="username" required>  
  password: <input type="password" name="password" required>  
  <input name="verification" type="hidden" value="6fGBtLZmVBZ59oy">  
  <button type="submit">Login</button>  
</form>
```

Login Forms Contain

A text field for the username

A password field for the password

A hidden field for a CSRF check

A button to submit the form

The form tag containing the form method and form action

```
<form method="POST" action="/login">  
  username: <input type="text" name="username" required>  
  password: <input type="password" name="password" required>  
  <input name="verification" type="hidden" value="6fGBtLZmVBZ59oy">  
  <button type="submit">Login</button>  
</form>
```

Login Forms Contain

A text field for the username

A password field for the password

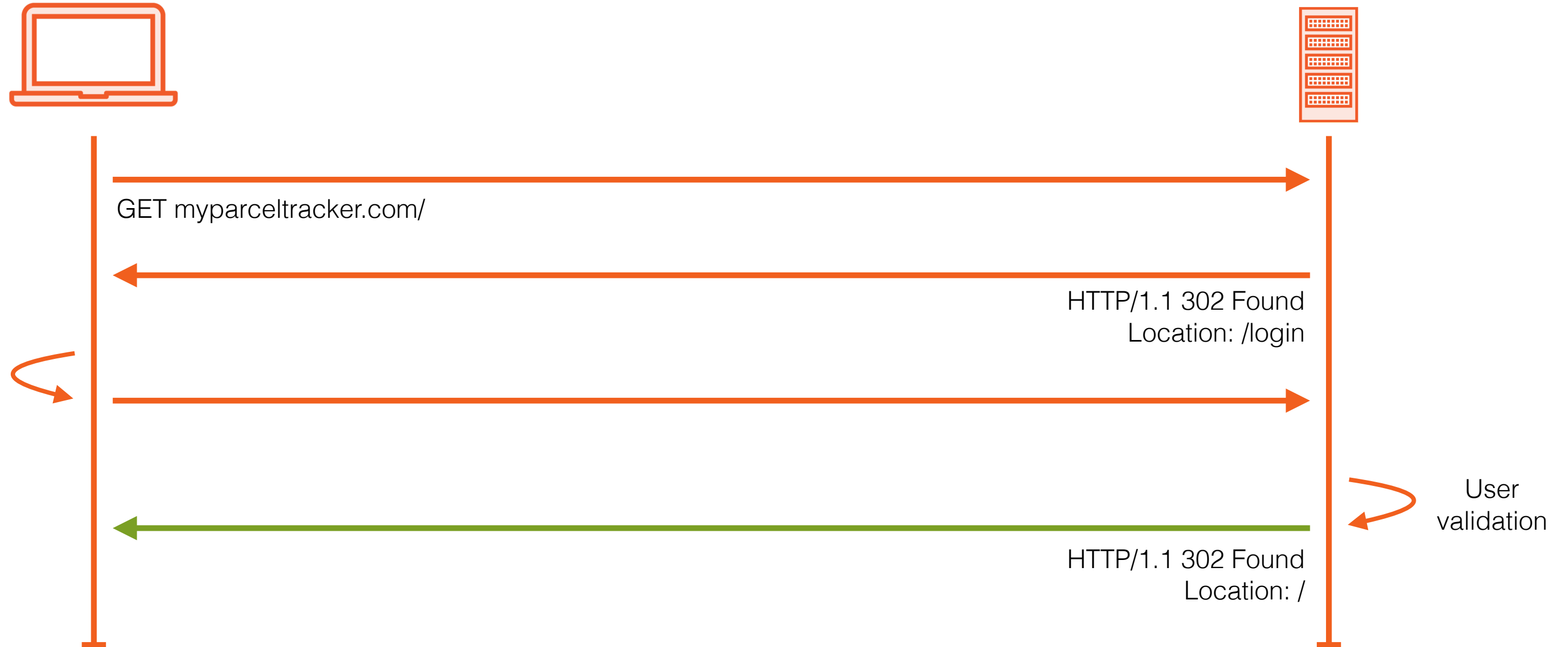
A hidden field for a CSRF check

A button to submit the form

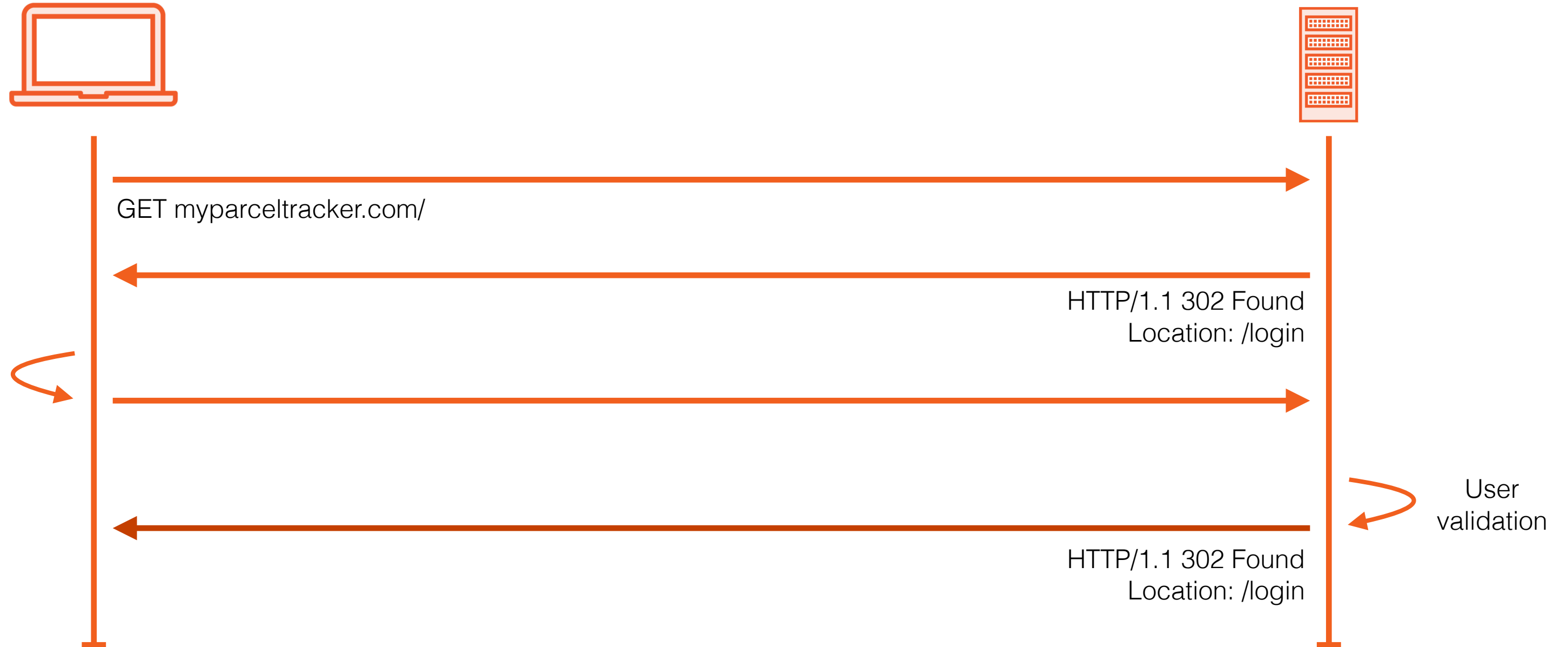
The form tag containing the form method and form action

How Form-based Authentication Works

Form-based Authentication Overview

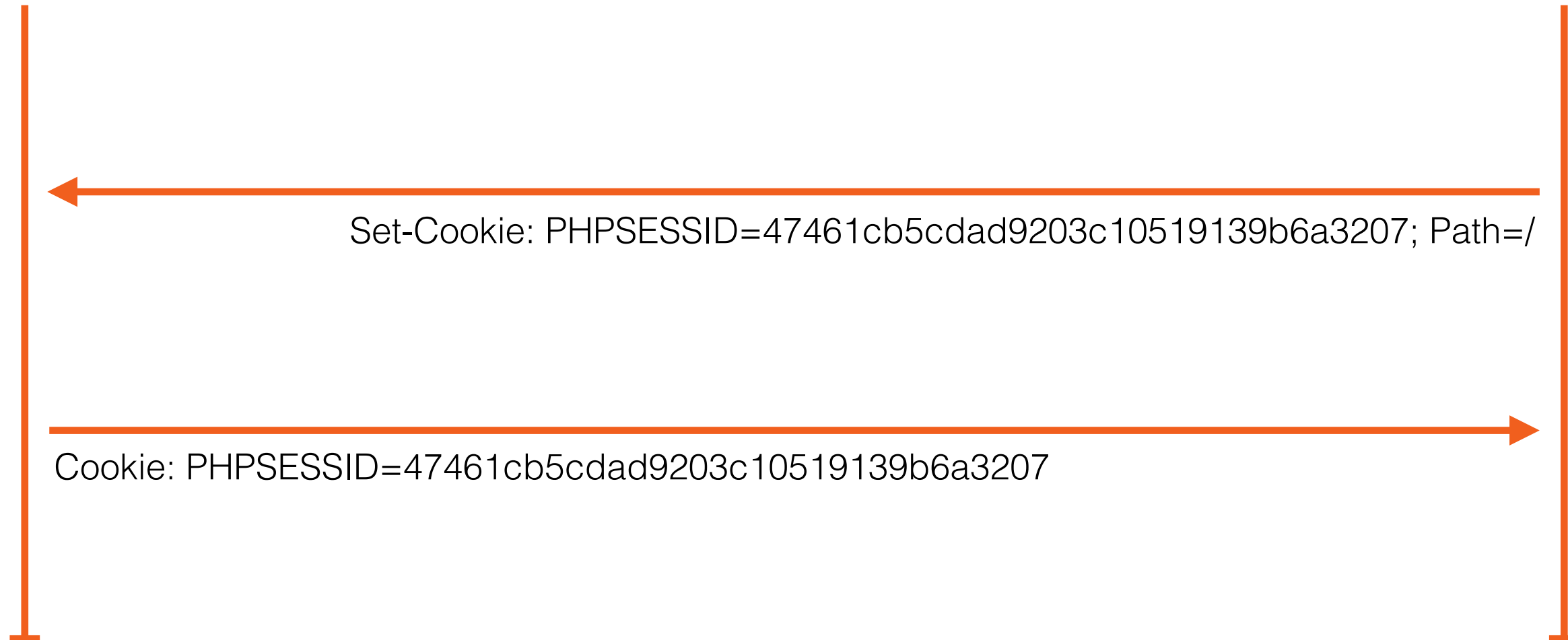


Form-based Authentication Overview



HTTP Is a Stateless Protocol

PHP Sessions



Session Cookies

GET / HTTP/1.1

Host: parceltracker.localhost

Accept-Language: en-GB,en;q=0.5

Accept-Encoding: gzip, deflate

Referer: http://parceltracker.localhost/login

Connection: keep-alive

Cookie: PHPSESSID=0d9660b0520fee6058518b5d516a5df3

Upgrade-Insecure-Requests: 1

Sec-GPC: 1

Sessions

- Perform authentication over HTTPS
- Session id identifies a user's session data
- `$_SESSION` is populated with unserialised session data
- Data in `$_SESSION` is serialized on PHP shutdown or call to `session_write_close`

Sessions

- Session data is saved in flat files by default
- Determined by `session.save_handler`
- File location is set in `session.save_path`
- Session data can be stored in other ways

Quick Recap

- Form-based authentication essentials
- A form's essentials elements
- Saw the form-based authentication flow

Up Next:

Advantages and Disadvantages

Advantages and Disadvantages

Quick Overview

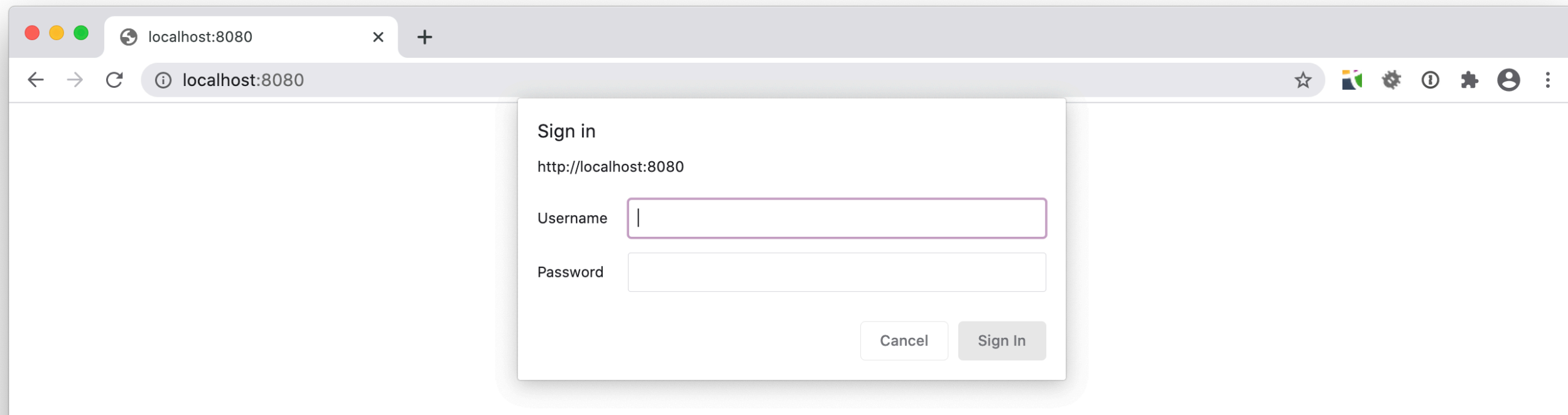
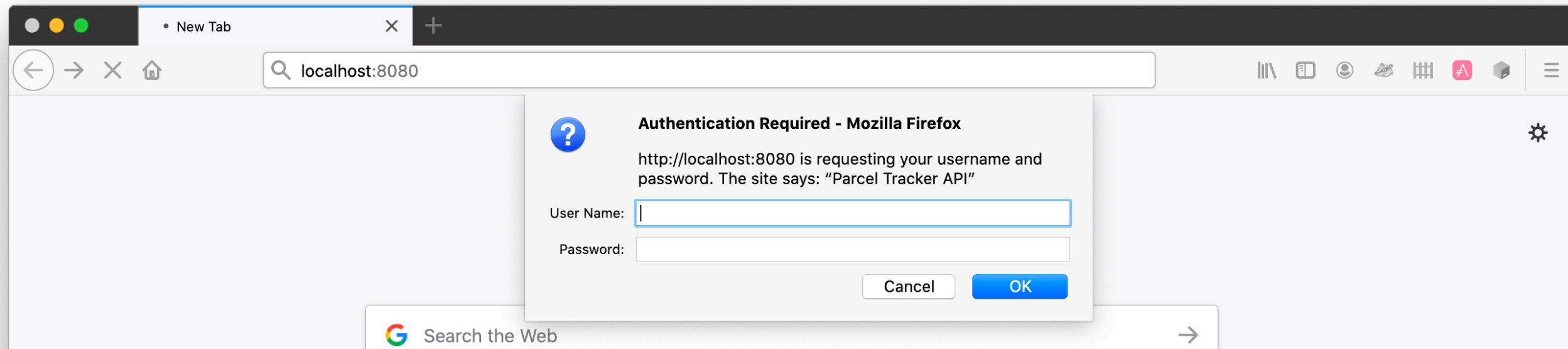
- Advantages and disadvantages of Form-based authentication

Advantages

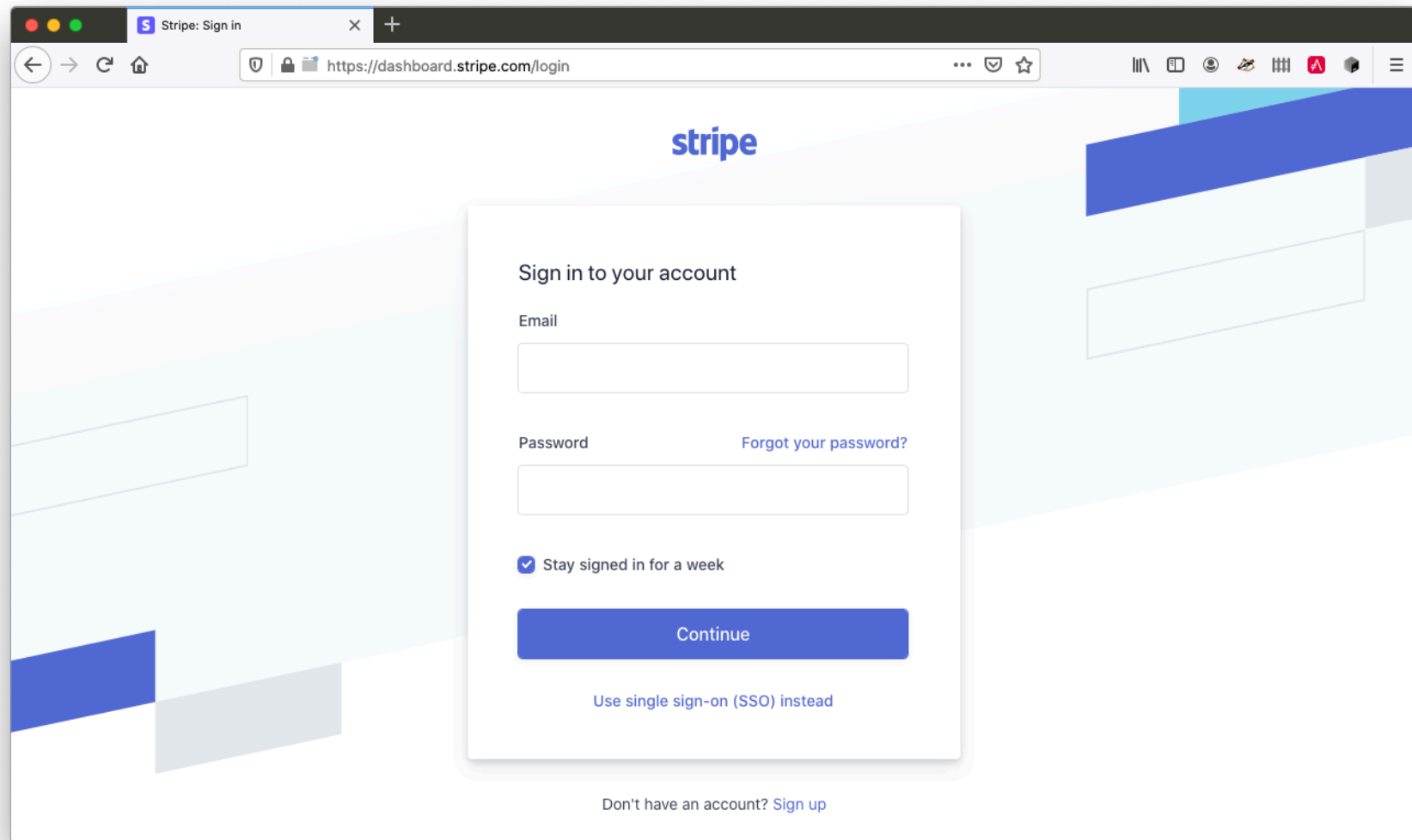


- Extremely common
- Numerous PHP libraries available
- Support available in PHP frameworks
- Very flexible

HTTP Authentication Examples



Form-based Authentication Examples



A screenshot of the Stripe sign-in page. The browser address bar shows 'https://dashboard.stripe.com/login'. The Stripe logo is at the top center. Below it, the heading 'Sign in to your account' is displayed. The form includes an 'Email' input field, a 'Password' input field with a 'Forgot your password?' link, a checked checkbox for 'Stay signed in for a week', and a blue 'Continue' button. At the bottom, there is a link for 'Use single sign-on (SSO) instead' and a link for 'Don't have an account? Sign up'.

Stripe: Sign in

https://dashboard.stripe.com/login

stripe

Sign in to your account

Email

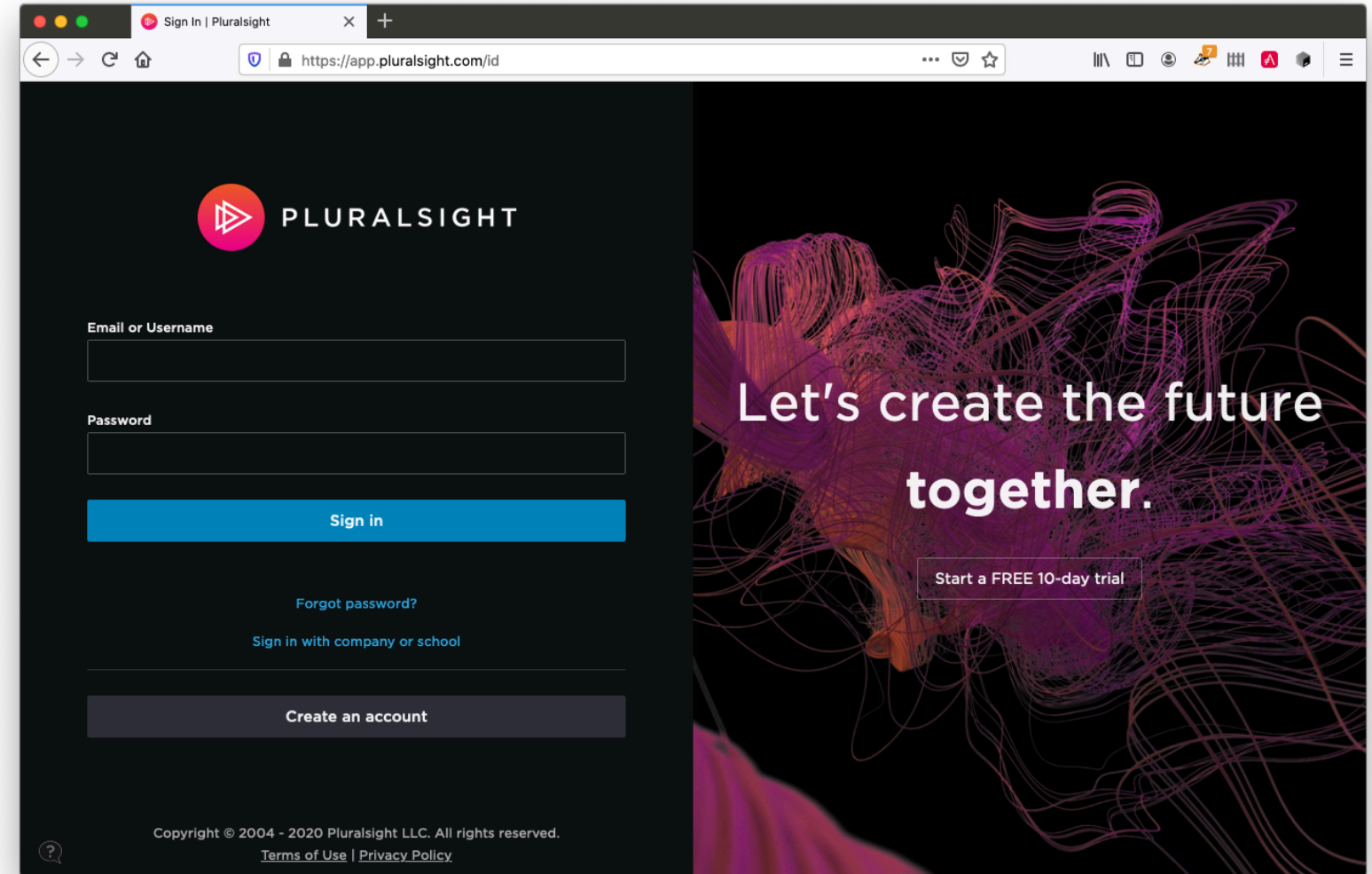
Password [Forgot your password?](#)

☒ Stay signed in for a week

Continue

[Use single sign-on \(SSO\) instead](#)

Don't have an account? [Sign up](#)



A screenshot of the Pluralsight sign-in page. The browser address bar shows 'https://app.pluralsight.com/id'. The Pluralsight logo is at the top center. The form includes an 'Email or Username' input field, a 'Password' input field, a blue 'Sign in' button, a 'Forgot password?' link, a 'Sign in with company or school' link, and a dark grey 'Create an account' button. On the right side, there is a large graphic with the text 'Let's create the future together.' and a button for 'Start a FREE 10-day trial'. The footer contains copyright information and links for 'Terms of Use' and 'Privacy Policy'.

Sign In | Pluralsight

https://app.pluralsight.com/id

PLURALSIGHT

Email or Username

Password

Sign in

[Forgot password?](#)

[Sign in with company or school](#)

Create an account

Let's create the future together.

Start a FREE 10-day trial

Copyright © 2004 - 2020 Pluralsight LLC. All rights reserved.
[Terms of Use](#) | [Privacy Policy](#)

Disadvantages



- Unstandardised look and feel
- Credentials are sent in plain text
- Can be susceptible to Man-in-the-middle attacks

Disadvantages



- Unstandardised look and feel
- Credentials are sent in plain text
- Can be susceptible to Man-in-the-middle attacks
- Susceptible to Phishing attacks

“The internationalized domain name (IDN) homograph attack is a way a malicious party may deceive computer users about what remote system they are communicating with, by exploiting the fact that many different characters look alike.”

IDN Homograph Attack - Wikipedia

A Homograph Attack

<https://matthewsetter.com>

A Homograph Attack

<https://mαttthewsetter.com>

Quick Recap

- Advantages and disadvantages
- Potential attack vectors

Up Next:

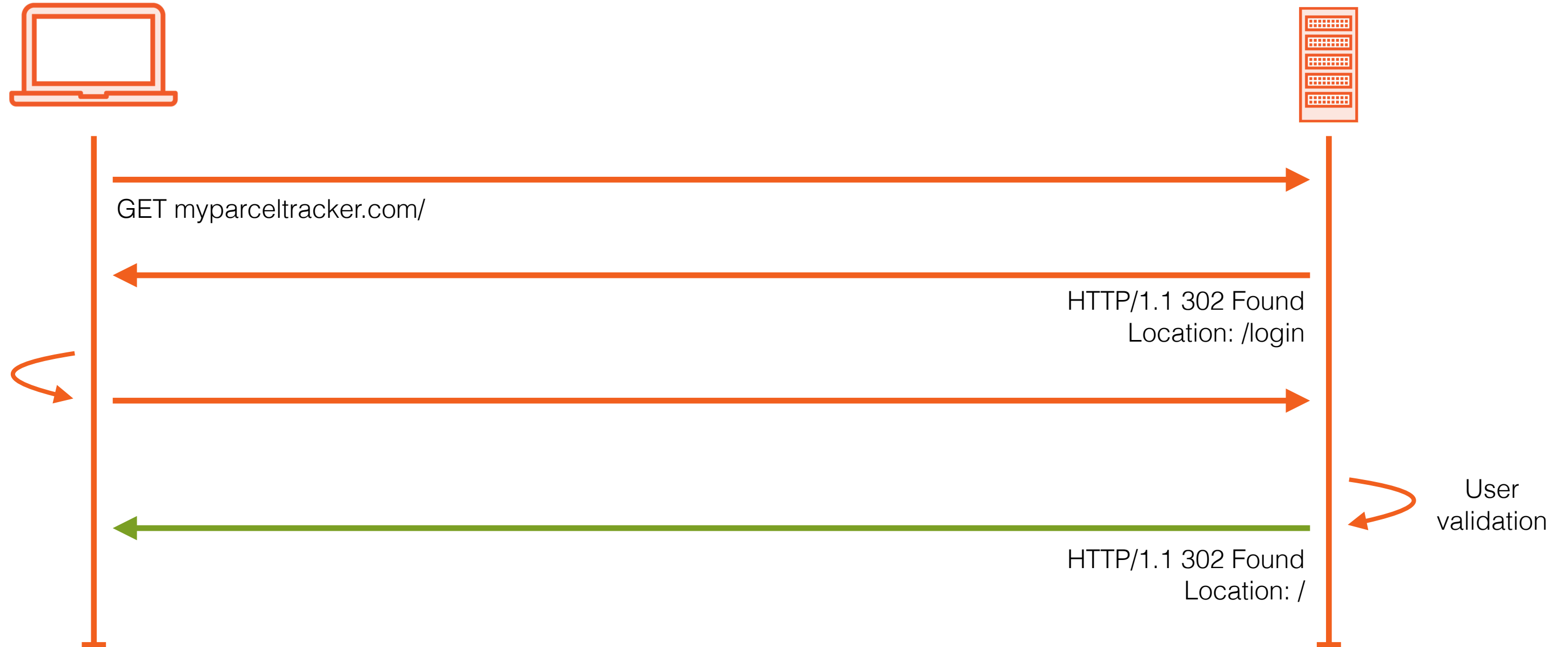
Implement Form-based Authentication in PHP

Implement Form-based Authentication in PHP

Implementing Form-based Authentication in PHP

- Many third-party libraries available
- Can be done without third-party support
- Libraries are likely a better choice

What We're Going To Do



“In cryptography, a timing attack is a side-channel attack in which the attacker attempts to compromise a cryptosystem by analyzing the time taken to execute cryptographic algorithms.”

Timing Attacks - Wikipedia

“Session Fixation is an attack that forces a user’s session ID to a known value, permitting an attacker to hijack user sessions.”

Session Fixation - WhiteHat Security

Key setcookie Parameters

Key setcookie Options

path	Sets the path where the cookie will be used.

Key setcookie Options

path

Sets the path where the cookie will be used.

domain

Sets the domain, or subdomain, that the cookie is used for.

Key setcookie Options

path	Sets the path where the cookie will be used.
domain	Sets the domain, or subdomain, that the cookie is used for.
secure	Sets whether the cookie should only be sent over a secure connection by the client, <i>if</i> a secure connection is available.

Key setcookie Options

path	Sets the path where the cookie will be used.
domain	Sets the domain, or subdomain, that the cookie is used for.
secure	Sets whether the cookie should only be sent over a secure connection by the client, <i>if</i> a secure connection is available.
httponly	Sets whether the cookie is available over HTTP only or not. If set to false the cookie won't be accessible to JavaScript.

Important Session Configuration Directives

Important Session Configuration Directives

session.save_path

By default, this is where PHP stores the session files.

Important Session Configuration Directives

`session.save_path`

By default, this is where PHP stores the session files.

`session.name`

This sets the name of the session.
By default, it's set to "PHPSESSID".

Important Session Configuration Directives

`session.save_path`

By default, this is where PHP stores the session files.

`session.name`

This sets the name of the session.
By default, it's set to "PHPSESSID".

`session.auto_start`

This sets whether sessions are automatically started.

Important Session Configuration Directives

<code>session.save_path</code>	By default, this is where PHP stores the session files.
<code>session.name</code>	This sets the name of the session. By default, it's set to "PHPSESSID".
<code>session.auto_start</code>	This sets whether sessions are automatically started.
<code>session.use_trans_sid</code>	Sets whether transparent sid support is enabled or not, meaning whether the session id is passed in the URL or not.

Important Session Configuration Directives

session.cookie_domain	Sets the domain which the session cookie can be used for.

Important Session Configuration Directives

`session.cookie_domain`

Sets the domain which the session cookie can be used for.

`session.use_strict_mode`

Sets whether accept uninitialized session IDs. If an uninitialized/unknown one is sent from the client, a new session ID will be generate and sent back.

Important Session Configuration Directives

<code>session.cookie_domain</code>	Sets the domain which the session cookie can be used for.
<code>session.use_strict_mode</code>	Sets whether accept uninitialized session IDs. If an uninitialized/unknown one is sent from the client, a new session ID will be generate and sent back.
<code>session.use_cookies</code>	Sets whether the session identifier is sent using cookies.

Important Session Configuration Directives

<code>session.cookie_domain</code>	Sets the domain which the session cookie can be used for.
<code>session.use_strict_mode</code>	Sets whether accept uninitialized session IDs. If an uninitialized/unknown one is sent from the client, a new session ID will be generate and sent back.
<code>session.use_cookies</code>	Sets whether the session identifier is sent using cookies.
<code>session.use_only_cookies</code>	Sets whether PHP will use cookies exclusively to send the session identifier.

Important Session Configuration Directives

session.cookie_secure	Sets whether session cookies should only be sent over secure (HTTPS) connections.

Important Session Configuration Directives

`session.cookie_secure`

Sets whether session cookies should only be sent over secure (HTTPS) connections.

`session.cookie_httponly`

Sets whether session cookies should only be sent over HTTP.

Important Session Configuration Directives

`session.cookie_secure`

Sets whether session cookies should only be sent over secure (HTTPS) connections.

`session.cookie_httponly`

Sets whether session cookies should only be sent over HTTP.

`session.cookie_samesite`

Sets whether cookies should not be sent with cross-site requests. It can be set to one of three options: *None*, *Lax*, and *Strict*.

SameSite Options

None

Lax

Strict

SameSite Options

None

Lax

Strict

SameSite Options

None

Lax

Strict

Quick Recap

- Learned how to implement form-based authentication in PHP
- setcookie's key parameters
- Key session configuration options

Module Recap

Module Recap

- Learned about Form-based authentication
- What it is
- How it works
- How to implement it in PHP
- Advantages and disadvantages
- Security considerations

Up Next:

What is Authorization?
