Authentication with Angular and JSON Web Tokens

Token Based Authentication

Check for Understanding

1. What is **authentication**?
2. What is **authorization**?
3. How have you previously implemented **authentication**?

Objectives

- Explain what a **token** is?
- Describe why we use **tokens** instead of cookies?
- Explain what **JSON Web Tokens** are?
- Describe an **interceptor**?
- Explain what a **resolve** is?

Cookies and Sessions

With cookies and sessions we authenticate the user on every request.

There is another way, we can use tokens.

What is a token?

Basically it is a signed (and possibly encrypted) string sent to the server with every request.

Why use tokens instead of cookies?

Before the emergence of single page applications, we usually had a single client and server and used cookies/sessions to maintain state and handle authentication.

The way we structure our applications has changed greatly over the past couple years.

We now have many different technologies and tools and our Single Page Applications consume multiple APIs.

We can easily have an application that uses a **Node API**, a **Rails API** as well as other **Web/Mobile APIs**.

This makes it a nightmare and almost impossible to try to share cookie/session data between these APIs.

It would be really nice if we could have one single "secret" (a key we store on a server) on all of our servers and share the token between each one!

Other advantages

- Cross-domain / **CORS**
  - **Cookies** + **CORS** don't play well together across different domains.
- **Stateless**
  - There is no need for a **session** store, the **token** is a self-contained entity that conveys all the user information

Other advantages

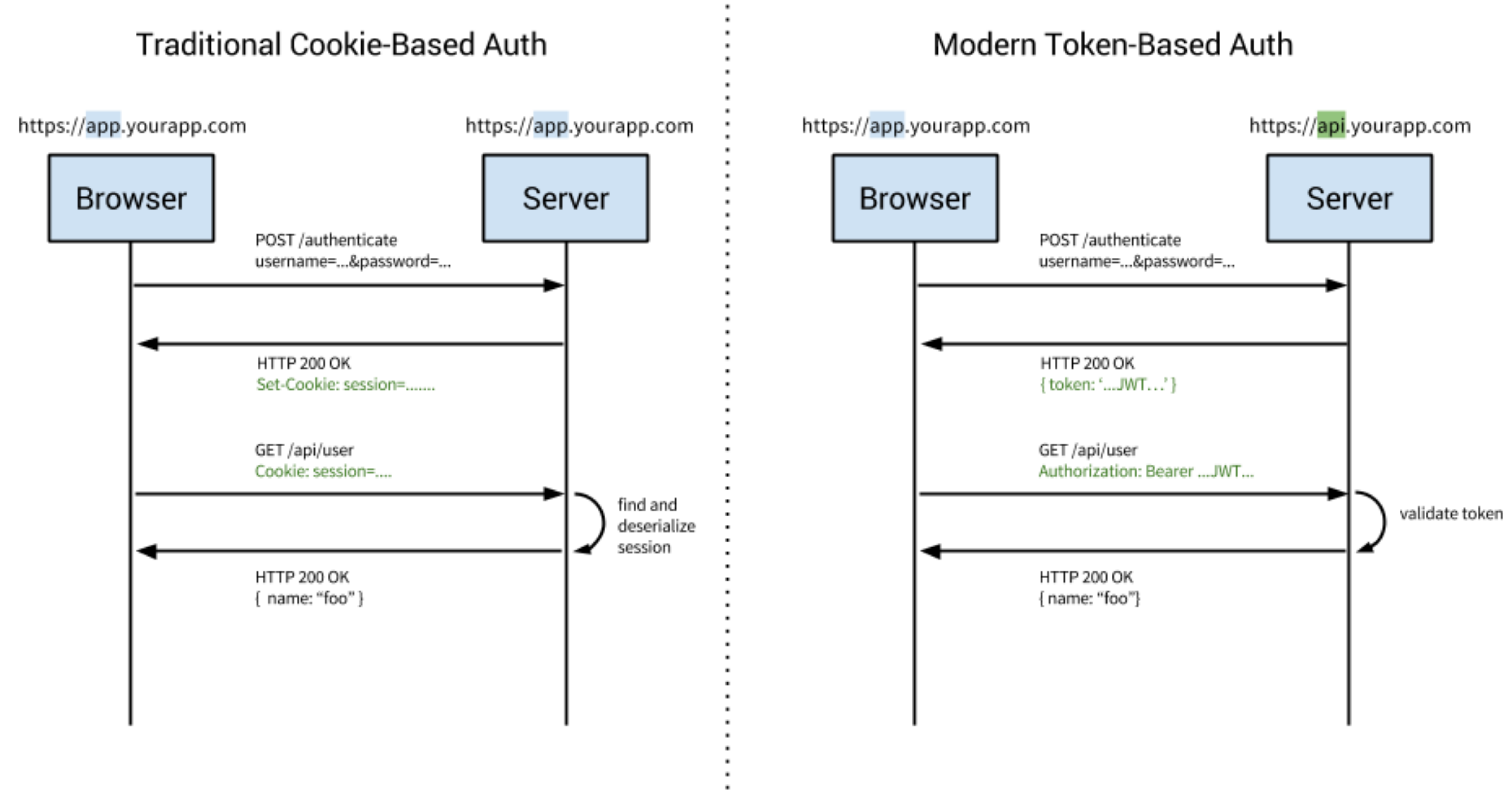
- **CDN**
  - You can serve all the assets from a **CDN** and your server side is just the **API**.
- Decoupling
  - You are not tied to a particular **authentication** scheme.

Other Advantages

- Mobile
  - Cookies are not ideal for native app development. **Token**-based approach simplifies this a lot.
- **CSRF**
  - Since you are not relying on **cookies**, you don't need to protect against cross site **requests**.

Other advantages

- Performance
  - A database lookup is much more costly than the calculation to validate a **token.**
- Standards based


Traditional Cookie-Based Auth
https://app.yourapp.com
https://app.yourapp.com
Browser
Server
POST /authenticate
username=...&password=...
HTTP 200 OK
Set-Cookie: session=.......
GET /api/user
Cookie: session=....
find and deserialize session
HTTP 200 OK
{ name: "foo" }
Modern Token-Based Auth
https://app.yourapp.com
https://api.yourapp.com
Browser
Server
POST /authenticate
username=...&password=...
HTTP 200 OK
{ token: '....JWT...' }
GET /api/user
Authorization: Bearer ....JWT...
validate token
HTTP 200 OK
{ name: "foo"}

Since **tokens** are a better option, what kind of **token** should we use?

JSON Web Tokens (JWT)

pronounced "Jot"

Very Popular

Open Standard (RFC 7519)

Interceptors

Interceptors are services that allow us to modify requests and responses before they are sent and after they return.

We can intercept our HTTP requests to attach the token to the header!

In an interceptor, this looks something like:

```
request: function(config){
  var token = localStorage.getItem("token");
  if(token)
    config.headers.Authorization = "Bearer " + token;
  return config;
}
```

Resolve

We want to make sure that promises are resolved before we render a page.

To do this, we use the resolve property which is accessible in each one of our routes.

Here is an example of a **resolve**. In this **route** we are **injecting two dependencies** into our **controller**, *currentUser* and ***users***.

```
.when('/home',{
  templateUrl: "templates/home.html",
  controller: "HomeController",
  resolve: {
    currentUser : function(UserService) {
      return UserService.getCurrentUser();
    },
    users: function(UserService){
      return UserService.getAllUsers()
    }
  }
})
```

A resolve contains one or more promises that must resolve successfully before the route will change.

Objectives

- Explain what a **token** is?
- Describe why we use **tokens** instead of cookies?
- Explain what **JSON Web Tokens** are?
- Describe an **interceptor**?
- Explain what a **resolve** is?

?