Token-based Authentication

Jogesh K. Muppala

THE DEPARTMENT OF
COMPUTER SCIENCE & ENGINEERING
計算機科學及工程學系

香港科技大學
THE HONG KONG UNIVERSITY OF
SCIENCE AND TECHNOLOGY

Cookies + Session Authentication

- Cookies set on the client side by the server
- Cookies used as a storage for session ID that is used as an index into server-side storage of session information

Why Token-Based Authentication?

- Session authentication becomes a problem when we need stateless servers and scalability
- Mobile application platforms have a hard time handling cookies/sessions
- Sharing authentication with other applications not feasible
- Cross-origin resource sharing (CORS) problem
- Cross-site request forgery (CSRF)

Token-based Authentication

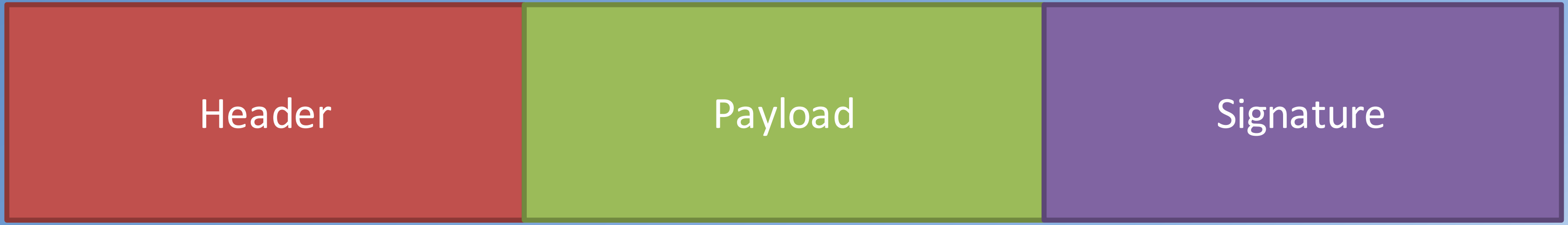
1. User requests access with their username and password
2. Server validates credentials
3. Server creates a signed token and sends it to the client
   - Nothing stored on the server
4. All subsequent requests from the client should include the token
5. Server verifies the token and responds with data if validated

JSON Web Tokens (JWT)

- Standards based:
  - IETF RFC 7519*
- Self-contained
  - carry all the information necessary within itself
- Shareable
  - Can share it with other applications to act on your behalf

Internet Engineering Task Force (IETF)
Request for Comments (RFC)

JSON Web Tokens



Header

Payload

Signature

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

```
{
  “$__”: {

   }
  . . .
 _doc: {
    }
  . . .
}
```

```
HMACSHA256(
 base64UrlEncode(header) + "." +
base64UrlEncode(payload),
secret
 )
```

jsonwebtoken Node Module

- Implementation of JSON web tokens support

  npm install jsonwebtoken --save
- Provides several methods:
  - sign() for signing and issuing token
  - verify() for verifying and decoding token and making it available on the request property in Express