BTI425/WEB422 - Web Programming for Apps and Services

Lecture Recap:

Week 8 – Introduction to JWT for Securing Web API

Agenda

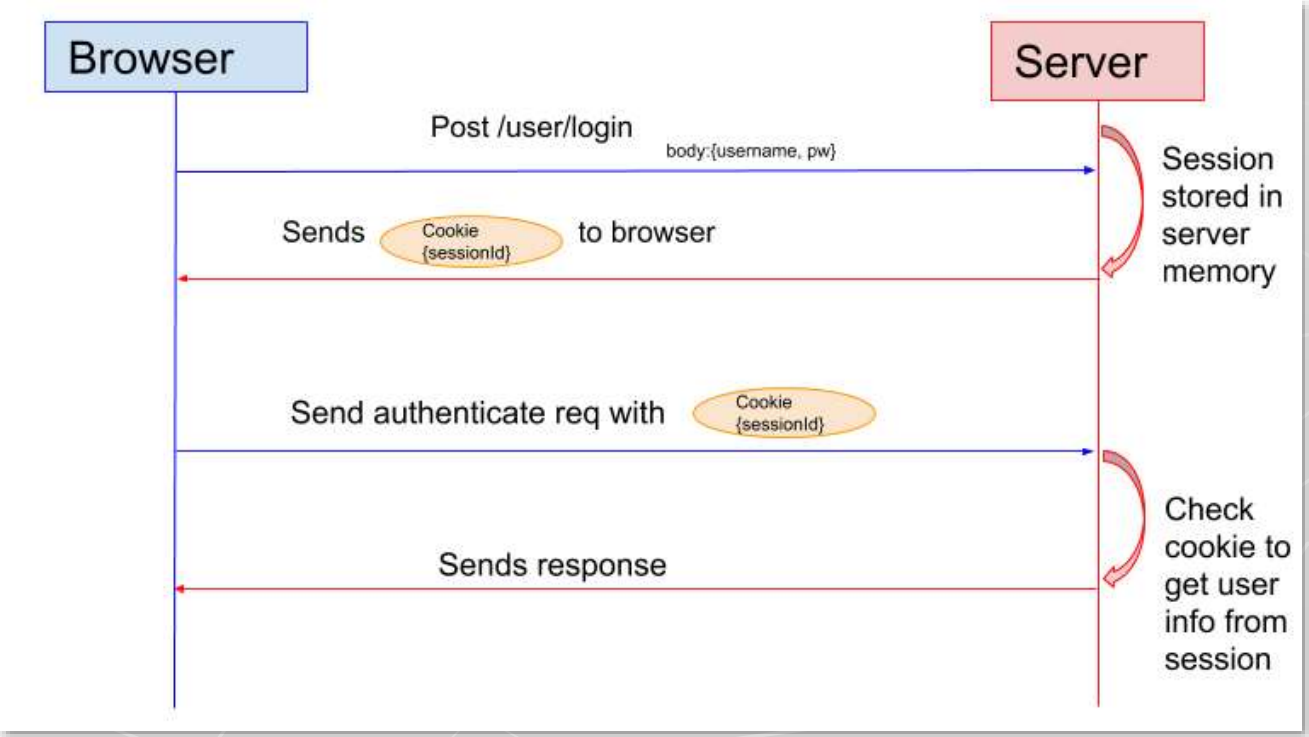
- Review: Web App Security
- Introduction to Securing a Web API with JWT
  - Note: this is a back-end development task

Review – Web App Security

- What is Web App Security?
- AAA
  - Accounting (Identity for MS): users in db table, register()
  - Authentication: Who are you? login()
  - Authorization
    - What can you do? What resources can you access?
      - e.g. securing routes in data-service.js of an app/api
- To ensure AAA
  - http -> https -> certification
  - users in db: password encryption – bcript.js
  - session management
    - Why is it needed?

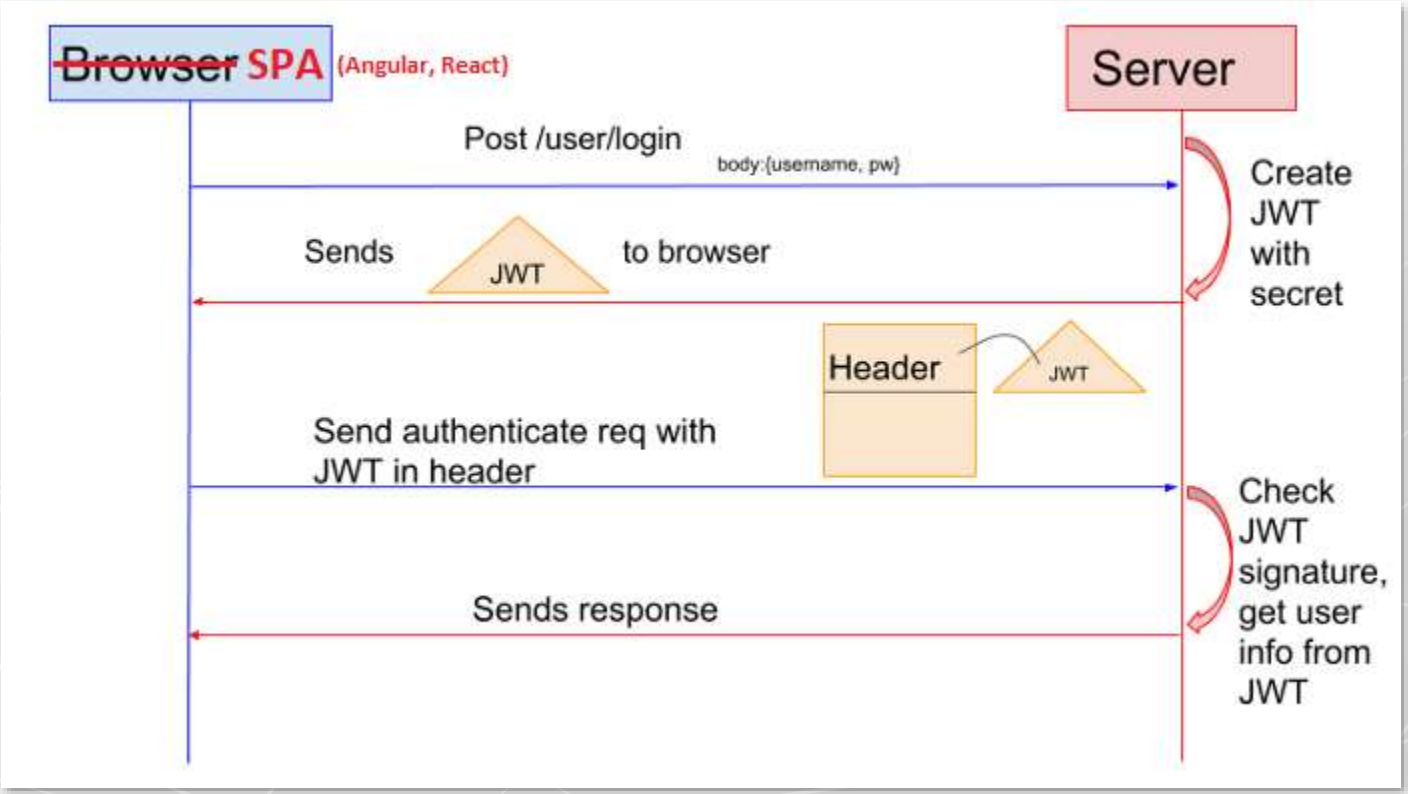
Review – Web App Security

- Session based authentication:
  - Managing state
  - First http request: login > session created on server > encryption > cookie > save in users' browser ← by using client-session.js
  - In all following http requests/responses, the session is attached



Securing a Web API

- WEB API:
  - No UI, no browser, no session
- JWT (JSON Web Token) – Token based authentication:
  - Similar to session based ~, but it's for WEB API



Week 8 Code Example

- Initial Projects:
  - simple-API
    - resource: /api/vehicles ,
    - testing tools; Thunder Client, Postman
- Final Projects:
  - simple-API-complete
    - /api/vehicles (secured with JWT),
    - /api/register, /api/login,
    - hashed password, users in MongoDB
      - MongoDB Atlas "Network Access": "0.0.0.0/0" on IP address whitelist

Securing a Web API with JWT

Introduction to implementing JWT in a WEB API:

- To implement JWT in a WEB API, 3 key modules are needed:
  - jsonwebtoken
    - used primarily to "sign" our json payload with a 'secret' to generate the token

    var token = jwt.sign(payload, jwtOptions.secretOrKey); // where?
  - passport
    - passport is Express-compatible authentication middleware:

    app.get("/api/vehicles", passport.authenticate('jwt', { session: false }), (req, res) => {}
    // this is an Authorization process for WEB API
    - used as application-level middleware: app.use(passport.initialize());
  - passport-jwt
    - used to setup token, configure its options (e.g secretOrKey), setup "strategy" using jwt options, jwt payload (current user's info)
    - the "strategy" is setup for passport: passport.use(strategy);

Securing a Web API with JWT

- Add/read the code to/in server.js

- Testing web API with JWT - Thunder Client/Postman
  - /api/register, /api/login
  - /api/vehicles (secured with JWT)
  - hashed password

The End