Advance Software Engineering Lab Report 9(Lab-Id 5\_1)

Name: Harika and Sumanth

Class Id: 15, 20

Team number: 5\_1

ICP team: 5

Approach:

Firstly, we will create middleware functions which can wrap endpoints with.

And the goal is to add JWT Authentication for these middleware functions in order to ensure that only authorized requestors can hit these REST API endpoints.

Code Explanation:

JWT contains a Header, a Payload, and a Signature

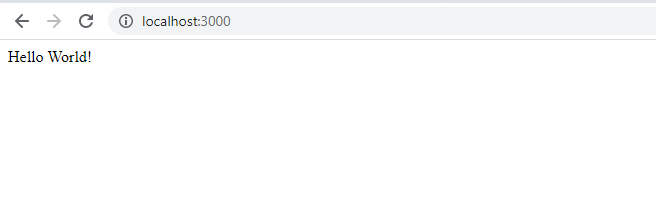
Payload is the set of claims contained within the JWT that contains a series of key/value pairs.

Inorder to start with project. Firstly , we need to install the following express.js node module

>npm install express --save

1noderun.PNG

* Run the above command, and it will display that it is listening on port 3000



* Run the localhost:3000 in the browser, you will be able to see the above output “hello world”
* Now lets add a few endpoints like HTTP Get endpoints

app.get('/secret',isAuthorized ,(req, res) => {

    res.json({ "message" : "THIS IS SUPER SECRET, DO NOT SHARE!" })

})

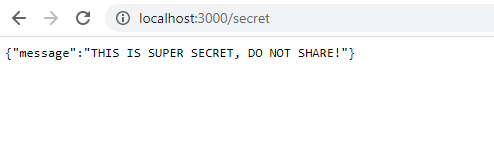
// and a /readme endpoint which will be open for the world to see

app.get('/readme', (req, res) => {

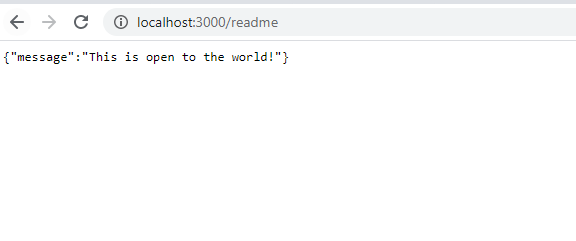
    res.json({ "message" : "This is open to the world!" })

})

* Run the project again, and open “localhost:3000/secret” in the browser and the response is shown the below screenshot



* When we open the readme endpoint, we can see response as shown in below screenshot



* For generating a valid JWT, we need install the jsonwebtoken module first by running below command

>npm install –save jsonwebtoken

* We need to create /jwt endpoint as shown below ,which will create a JWT token for us which features an incredibly simple payload { "body": "stuff" }

And is signed using a private key.

app.get('/jwt', (req, res) => {

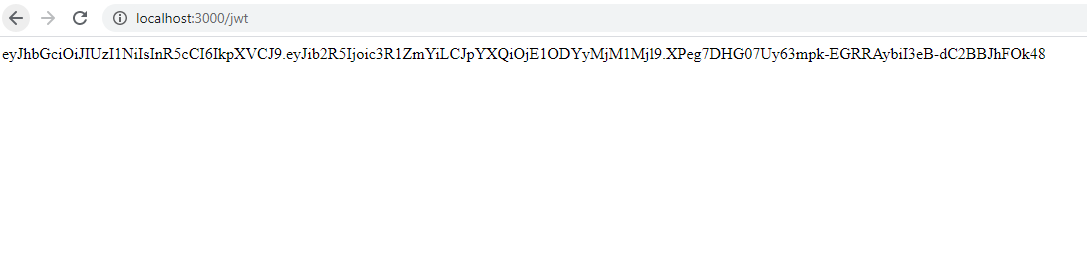
    let privateKey = fs.readFileSync('./private.pem', 'utf8');

    let token = jwt.sign({ "body": "stuff" }, privateKey, { algorithm: 'HS256'});

    res.send(token);

})

* Next, we need to open jwt endpoint in the browser ,and we can the response as shown below



* Now, we need to authenticate the middleware.
* function isAuthorized(req, res, next) {
* if (typeof req.headers.authorization !== "undefined") {
* // retrieve the authorization header and parse out the
* // JWT using the split function
* let token = req.headers.authorization.split(" ")[1];
* let privateKey = fs.readFileSync('./private.pem', 'utf8');
* // Here we validate that the JSON Web Token is valid and has been
* // created using the same private pass phrase
* jwt.verify(token, privateKey, { algorithm: "HS256" }, (err, user) => {
* // if there has been an error...
* if (err) {
* // shut them out!
* res.status(500).json({ error: "Not Authorized" });
* }
* // if the JWT is valid, allow them to hit
* // the intended endpoint
* return next();
* });
* } else {
* // No authorization header exists on the incoming
* // request, return not authorized
* res.status(500).json({ error: "Not Authorized" });
* }
* }

Now we need to update secret end point with isAuthorized() as shown below:

app.get('/secret',isAuthorized ,(req, res) => {

    res.json({ "message" : "THIS IS SUPER SECRET, DO NOT SHARE!" })

})

* Now if you run the secret endpoint in the browser, we will get the response as shown below



* We are not able to access it without a proper authorization header, in order to display it,we need to provide proper authorization header with jwt
* For that,we need to run this secret endpoint with curl command, inorder to get response as shown below

