Add Token Authentication to Our Fastify App with fastify-jwt





Photo by Harley-Davidson on Unsplash

With the fastify-jwt library, we can add basic authentication to our Fastify app quickly.

In this article, we'll look at how to use the library to add authentication to our Fastify app.

Installation

We can install the package by running:

Issuing Tokens

We can issue tokens by using thee fastify.jwt.sign method.

For example, we can write:

```
const fastify = require('fastify')({
   logger: true
})
fastify.register(require('fastify-jwt'), {
   secret: 'secret'
})

fastify.post('/signup', (req, reply) => {
   const token = fastify.jwt.sign({ foo: 'bar' })
   reply.send({ token })
})

fastify.listen(3000, '0.0.0.0', function (err, address) {
   if (err) {
     fastify.log.error(err)
     process.exit(1)
   }
   fastify.log.info(`server listening on ${address}`)
})
```

We just call the method with the object we want as the payload to return a token.

Verify Token

We can verify the token with the jwtVerify method.

For example, we can write:

```
const fastify = require('fastify')({
  logger: true
fastify.register(require('fastify-jwt'), {
  secret: 'secret'
})
fastify.decorate("authenticate", async (request, reply) => {
    try {
      await request.jwtVerify()
    } catch (err) {
      reply.send(err)
  })
  .after(() => {
    fastify.route({
      method: 'GET'
      url: '/secret',
      preHandler: [fastify.authenticate],
      handler: (req, reply) => {
```

```
reply.send('secret')
}
})
})

fastify.post('/signup', (req, reply) => {
  const token = fastify.jwt.sign({
    foo: 'bar'
  })
  reply.send({
    token
  })
})

fastify.listen(3000, '0.0.0.0', function(err, address) {
  if (err) {
    fastify.log.error(err)
    process.exit(1)
  }
  fastify.log.info(`server listening on ${address}`)
})
```

We call the request.jwtVerify method, which is available after we call the fastify.register method.

If there's an error, the catch block will be run.

To add some routes that require the auth token, we add the after callback with the routes defined by fastify.route.

preHandler runs before the route is called so that we can run the auth token check and call the route handler only if the token is valid.

Cookies

We can put the auth token in the cookies.

For example, we can write:

```
const fastify = require('fastify')()
const jwt = require('fastify-jwt')

fastify.register(jwt, {
    secret: 'key',
    cookie: {
       cookieName: 'token'
    }
})

fastify
    .register(require('fastify-cookie'))

fastify.get('/cookies', async (request, reply) => {
    const token = await reply.jwtSign({
       name: 'foo',
    }
}
```

```
role: ['admin', 'spy']
 })
reply
    .setCookie('token', token, {
      domain: '*',
      path: '/',
      secure: true,
      httpOnly: true,
      sameSite: true
   })
    .code(200)
    .send('Cookie sent')
})
fastify.get('/verifycookie', async (request, reply) => {
   await request.jwtVerify()
    reply.send({ code: 'OK', message: 'it works!' })
 catch(error){
   reply.send(error);
})
fastify.listen(3000, '0.0.0.0', function (err, address) {
 if (err) {
   fastify.log.error(err)
   process.exit(1)
 fastify.log.info(`server listening on ${address}`)
})
```

We add the /cookies route to issue the cookie.

The setCookie is available after we register the fastify-cookie plugin.

We pass the token into the setCookie method to pass the cookie in the response header.

secure makes sure it's sent over HTTPS.

httpOnly makes sure it's only used with HTTP requests.

sameSite checks if the cookie is issued from the same site it's used in.

In the verifycookie route, we call request.jwtVerify to verify the cookie before doing anything else.

Conclusion

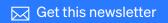
We can use fastify-jwt to issue JSON web tokens and verify it.

Enjoyed this article? If so, get more similar content by <u>subscribing to our YouTube</u> <u>channel!</u>

Sign up for In Plain English

By JavaScript in Plain English

Updates from the world of programming, and In Plain English. Always written by our Founder, Sunil Sandhu. <u>Take a look.</u>



You'll need to sign in or create an account to receive this newsletter.

Programming

Web Development

Technology

Software Development

JavaScript



About Help Legal

Get the Medium app



