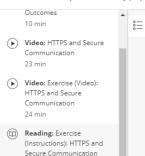




Server-side Development with NodelS. Express and M > Week 4 > Exercise (Instructions): HTTPS and Secure Communica

Prev | Next



Reading: HTTPS and Secure Communication: Additional Resources

#### **Uploading Files**

Cross-Origin Resource Sharing

OAuth and User Authentication

Assignment 4: Backend

# Exercise (Instructions): HTTPS and Secure Communication

#### Objectives and Outcomes

In this exercise you will explore the use of the HTTPS server core node module to create and run a secure server. You will also learn about generating your private key and public certificate and use them to configure your Node HTTPS server. At the end of this exercise, you will be able to:

- · Configure a secure server in Node using the core HTTPS module
- · Generate the private key and public certificate and configure the HTTPS server
- Redirect traffic from the insecure HTTP server to a secure HTTPS server.

#### Generating Private Key and Certificate

• Go to the bin folder and then create the private key and certificate by typing the following at the prompt:

```
openssl genrsa 1024 > private.key
popenssl req -new -key private.key -out cert.csr
popenssl x509 -req -in cert.csr -signkey private.key -out certificate.pem
```

#### Note for Windows Users

. If you are using a Windows machine, you may need to install openssl. You can find some openssl binary distributions here. Also, this article gives the steps for generating the certificates in Windows. Another article provides similar instructions. Here's an online service to generate self-signed certificates.

### Configuring the HTTPS Server

• Open the www file in the bin directory and update its contents as follows:

```
var https = require('https');
var fs = require('fs');
          app.set('secPort',port+443);
   12
13
14
15
16
17
18
19
           * Create HTTPS server.
*/
        var options = {
   key: fs.readFileSync(_dirname+'/private.key'),
   cert: fs.readFileSync(_dirname+'/certificate.pem')
};
  20
21
22
23
24
25
26
27
28
29
         var secureServer = https.createServer(options,app);
         /**  
* Listen on provided port, on all network interfaces.  
*/
         secureServer.listen(app.get('secPort'), () => {
   console.log('Server listening on port ',app.get('secPort'));
         });
secureServer.on('error', onError);
secureServer.on('listening', onListening);
32
33 . . .
```

· Open app.js and add the following code to the file:

```
// Secure traffic only
app.all('*', (req, res, next) => {
  if (req.secure) {
    return next();
}
               res.redirect(307, 'https://' + req.hostname + ':' + app.get('secPort') + req
  10 }
11 });
13 . . .
```

- · Run the server and test.
- . Do a Git commit with the message "HTTPS".

## Conclusions

III ulis exercise, you learnt about configuring a secure server running milits protocol in our express application.

Mark as completed

