4.4 Simulation 4: Something they don't know, don't have, or aren't

Topics: Two-Factor Authentication

Regular authentication with only a username and password is insufficient to protect top secret information within DCS. We must extend the new departmental website with two-factor authentication (2FA). Our intern, "Joe", has made a start on this and you can clone his code off GitHub. Don't expect too much, though:

```
$ git clone https://github.com/dcs-cs263/lab4
```

The code comes in two parts: the web `server` and the `authenticator`, found in their respective folders.

Server If you `cd server`, you can compile and run the server with the following:

```
$ gradle run
```

The `gradle run` command will recompile the code if necessary before running it. This will start up a web server which hosts the new DCS website on a random port. The chosen port will be shown in the standard output. Open a web browser and navigate to the address shown in the standard output. The website is using the Spark framework for Java[6] as before.

In order to avoid annoying issues related to total data loss with VolatileDB, a default account is now created when the server is started:

Username: `intern@wondoughbank.com`
Password: `password`

If you log in with these credentials, you should normally be taken to the 2FA page which will display a key and has an input for the one-time password. However, that is currently not implemented.

Authenticator If you `cd authenticator`, you can compile and run the authenticator application with the following:

```
$ gradle run
```

[6] `http://sparkjava.com/`

This will start a terminal program that prompts you for the key. Once you complete the exercises below, it should calculate a suitable one-time password for the 2FA page.

Ex10 In the `authenticator/src/main/java/Program.java` file, implement the `intervals` method, which should calculate the number of intervals that have elapsed since the date and time represented by `startDateTime` where the duration of an interval is represented by `interval`. You can find the documentation for the `java.time` package on the Oracle website:

`https://docs.oracle.com/javase/8/docs/api/java/time/package-summary.html`

Ex11 Also in `authenticator/src/main/java/Program.java`, implement the `otp` method, which should calculate a time-based one-time password (TOTP) based on the `key` that is supplied via the standard input to the program and the number of intervals `c` that have elapsed since `startDateTime`. The exact process you use to calculate the TOTP is up to you, but you are encouraged to use some form of HMAC. You don't need to implement this from scratch and can use one of the available implementations in the Java standard library (available in the `java.crypto` package).

Ex12 Once you have completed the implementation of the authenticator application, you can move on to the DCS web application in `server`. You will want to copy your `intervals` and `otp` methods to the server code. You may wish to implement a different `startDateTime` and `interval` value on a per-user basis.

Ex13 Implement the `generate2FAKey` method in `LoginController.java` which should generate a random key to be used for the 2FA process.

Ex14 Complete the `handleLoginPost` method so that the `generate2FAKey` method is used to generate a key for the 2FA process. You should also stored the key in *e.g.* the current session. See the Spark documentation or other methods in the skeleton code for details on how to do that.

Ex15 Complete the `serve2FAPage` method. You should check that the user has completed the first step in the authentication process and that a 2FA key is available where you stored it. You then want to put the key in the model for the view so that it will be rendered to the user.

Ex16 Finally, you want to complete the `handle2FA` method which handles the submitted 2FA form. You should retrieve the key from where you have stored it. Assuming the OTP you calculate matches the one supplied by the user, you should then store the user's name in the session as `"currentUser"`.