4.2 Simulation 2: Authentication

Topics: Implementing authentication mechanisms

The Warwick Megalomaniacs Groups' Hackermen have become more aggressive in their attempts to break into our students' accounts in order to access module resources on the Department of Computer Science website. In order to make the life of the Warwick Megalomaniacs Group's Hackermen more difficult, you are to develop a new website for the department which features an authentication system which uses PBKDF2 to hash passwords.

One of our "interns", "Joe", has already started work on such a system over the "summer". You can obtain the code he wrote by running the following in a terminal:

```
$ git clone https://github.com/dcs-cs263/lab2
```

"Joe" had certain views on security guided by our first year security modules. Needless to say, the new website is incomplete and in dire need of improvements.

Once you have obtained the code from GitHub, you can compile the code with:

```
$ gradle build
```

This may take a little while to complete the first time you run it since it will initialise Gradle[4] and download some dependencies. Once the code has compiled successfully, you can run the program with:

```
$ gradle run
```

This will start up a web server which hosts the new DCS website. Open a web browser and navigate to the URL shown in the standard output to see it (the port is randomised on each start-up). The website is using the Spark framework for Java[5]. Most of the work that needs to be done will have to be implemented in the `LoginController` class which can be found in the following file: `src/main/java/LoginController.java`. You should also familiarise yourself with the other `.java` files in the `src/main/java/` directory.

Ex4 Our students will need to register accounts on our new authentication system. However, for some reason, the `register` method in the `LoginController` class does not

[4] If you have no prior exposure to Gradle, you can find more information on their website at `https://gradle.org/`.

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

seem to do this right. If a user tries to register, they are told that "WE WERE UNABLE TO CREATE YOUR IDENTITY". Additionally, no accounts are ever added to the database `LoginController.database`. Fix this. Remember that passwords should be hashed and salted using PBKDF2. The `SecurityConfiguration.pbkdf2` method implements the PBKDF2 algorithm for us. The `SecurityConfiguration` class also has static fields for the configuration to use: `SecurityConfiguration.ITERATIONS` for the number of iterations to use and `SecurityConfiguration.KEY_SIZE` for the key size.

Ex5 Our students will need to sign in to the new system. Currently, the system seems to just accept any username and password as valid. Fix this by correctly modifying the `authenticate` method in the `LoginController` class.

Ex6 The Warwick Megalomaniacs Group will not stop trying to break into our systems. We need to ensure that we future-proof our system and allow password hashes to be rehashed using different security settings if we feel that the number of iterations is too low or the key size too small. Complete the `rehashPassword` method in the `LoginController` class and update your `authenticate` method to use the `rehashPassword` method if necessary.

Ex7 Your web browser communicates with the web server via TCP. However, the data sent across the TCP connection is not encrypted. Therefore, it is possible to intercept messages between your browser and the web server. Write a program in a language of your choice (or use an existing tool) to perform a man-in-the-middle attack on messages sent between the browser and web server. For example, you could take the following approaches:

- Since you have full control over the system in this simulation, you could write a program which acts as a proxy between the browser and the web server (by listening on some TCP port and passing on all data to the web server). You can then point your web browser at that port rather than port 4567. You can monitor the data as it passes through your program.

- Use a program such as Wireshark or tcpflow to monitor the data that is sent across the connection. (These programs are not available on the machines in the DCS labs at the time of writing.)

Try to intercept a password as it is sent from the browser to the server.