

CS 305 Module Five Coding Assignment Checksum Verification Template

1. Algorithm Cipher

For this assignment, I recommend using the SHA-256 hashing algorithm for generating a checksum.

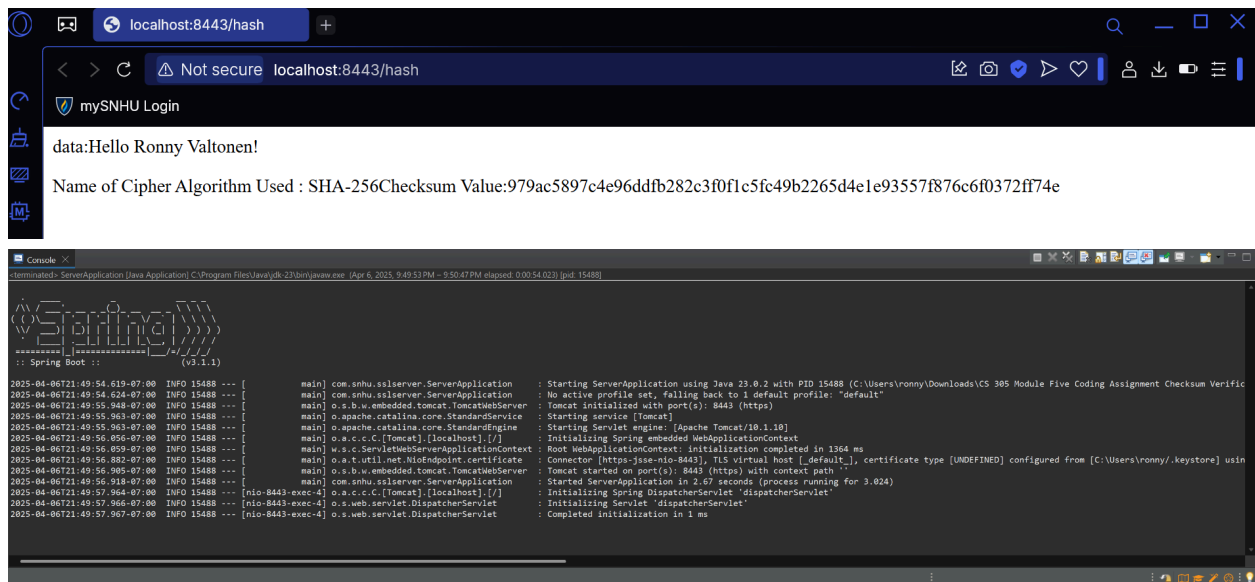
2. Justification

Because this is a widely recognized algorithm with no currently known vulnerabilities, it is quick to implement. No two inputs will ever produce the same hash output. In this scenario, we can verify that the public key hasn't been tampered with. Using MessageDigest (<https://docs.oracle.com/javase/8/docs/api/java/security/MessageDigest.html>) it is easy to implement and reliable. Because it is industry standard, it is compatible with the JDK platform. Avoiding collisions is important because if an attacker is able to create a different file with the same hash, they could replace the public key with a malicious one without anyone knowing. SHA-256 entirely overrides this as even a remote possibility.

3. Generate Checksum

I had to change the versioning of the Java to use Java 21 to support the version of Spring, the latest version of Java 23 did not work and the pre-coded Java version did not work either.

4. Verification



The screenshot shows a web browser window at localhost:8443/hash. The page displays "mySNHU Login" and a message: "data:Hello Ronny Valtonen!". Below this, it shows the "Name of Cipher Algorithm Used : SHA-256" and the "Checksum Value: 979ac5897c4e96ddfb282c3f0f1c5fc49b2265d4e1e93557f876c6f0372ff74e".

The console window below shows the output of a Java application. It includes the Spring Boot logo and a list of log messages. The messages indicate that the application is using Java 23.0.2, has no active profile set, and is starting on port 8443. It also shows the initialization of the Spring embedded WebApplicationContext and the starting of the Tomcat engine. The application is running on a Windows machine with the path C:\Program Files\Java\jdk-23\bin\java.exe.