Name: Pelumi Johnson
Date: January 19, 2026
Course: CMIT 436 | Cloud Security
Institution: University of Maryland Global Campus (UMGC)
Lab Title: Observing an SHA256-Generated Hash Value

Objective

The objective of this lab was to observe and verify an SHA-256 generated hash value for a downloaded file. The lab focused on generating an SHA-256 hash in a Linux environment and comparing it to an expected hash value to confirm file integrity.

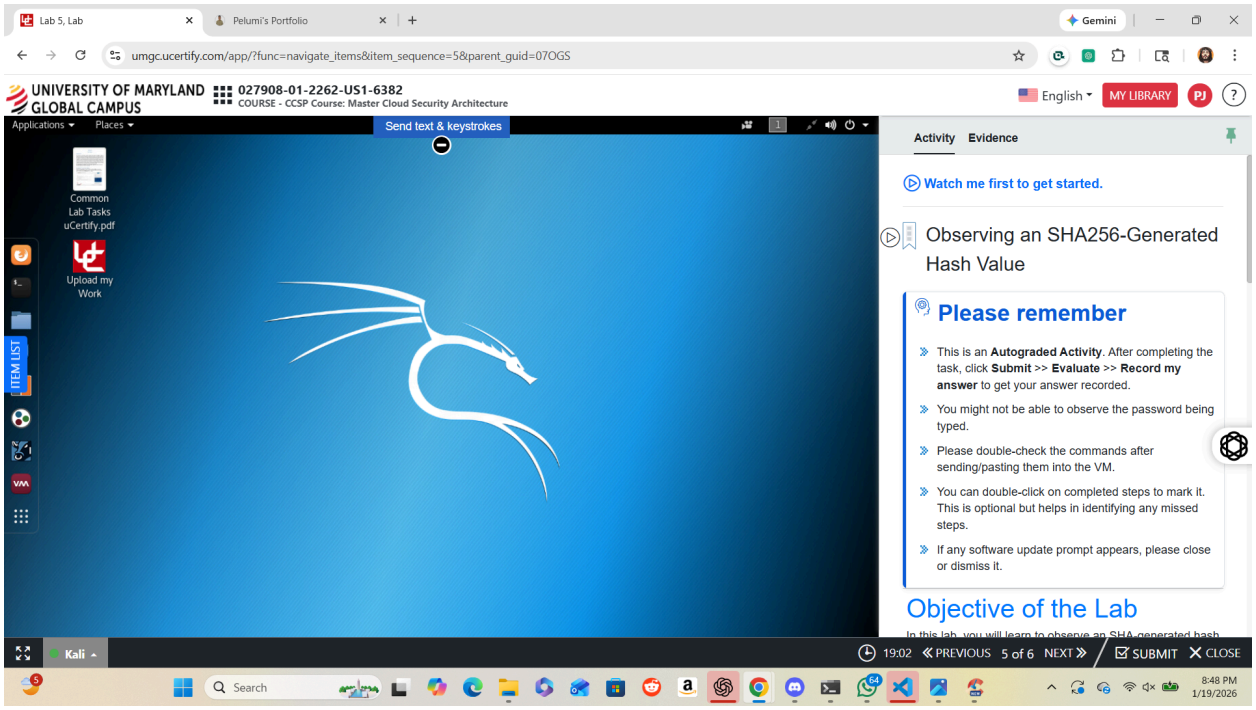
Tools & Environment Used

- uCertify Virtual Lab
- Kali Linux
- Terminal (Command Line Interface)
- sha256sum command
- diff command
- Downloaded ISO file (ucertify.iso)

Lab Overview

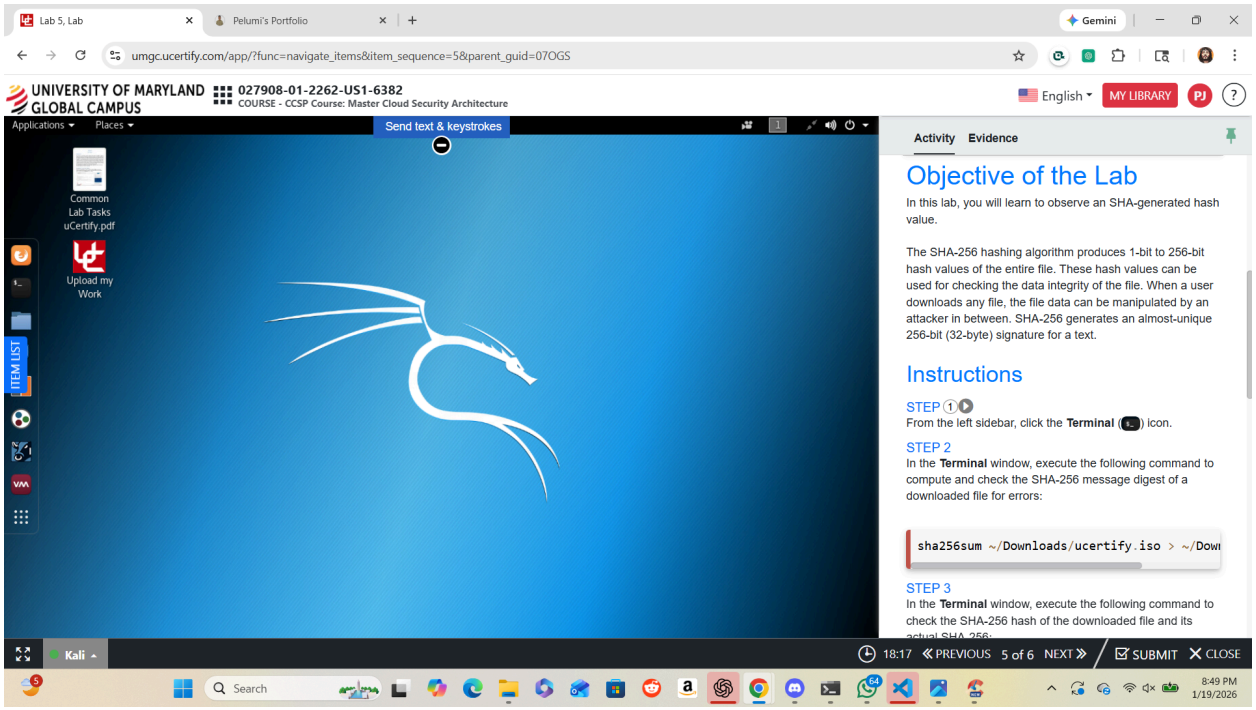
SHA-256 is part of the Secure Hash Algorithm (SHA-2) family and produces a 256-bit (32-byte) hash value. It generates an almost-unique digital fingerprint for a file. Hashes are used to verify that files have not been altered during download or transfer. If even one bit changes in the file, the resulting hash will be completely different.

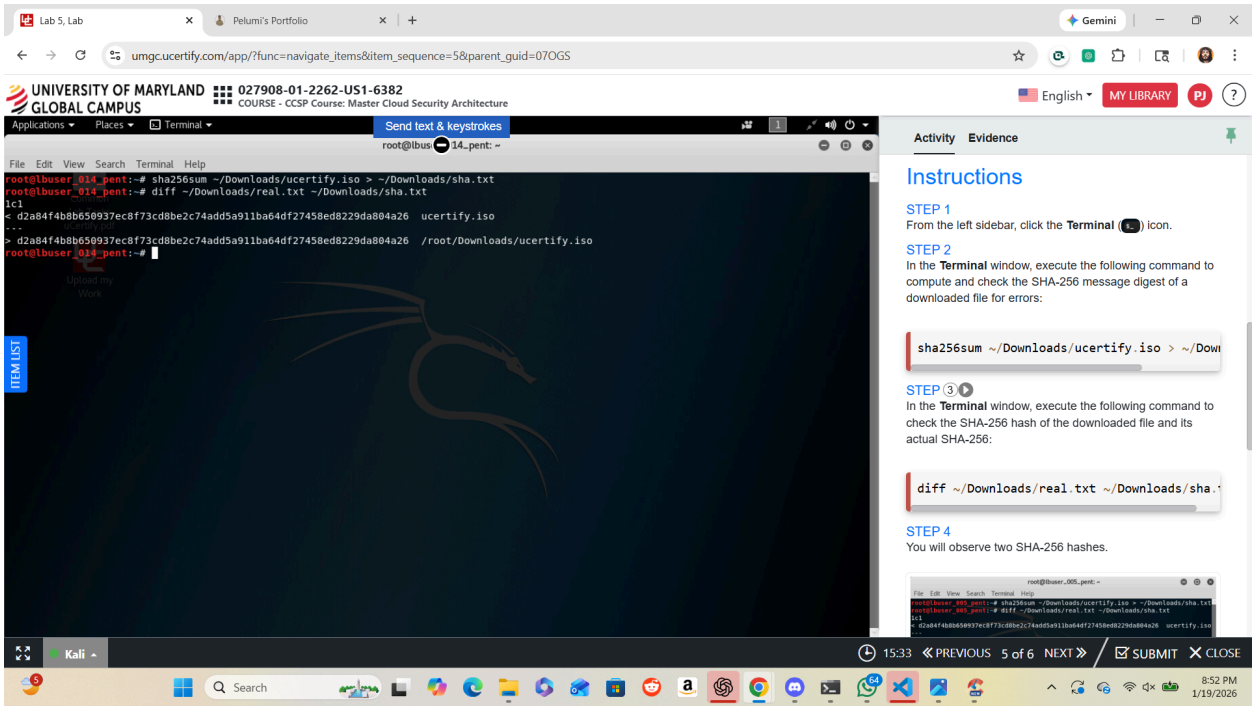
In this lab, the SHA-256 hash of a downloaded ISO file was generated and compared with the expected hash value to verify file integrity.



Step 1: Opening the Terminal

The Terminal icon was selected from the left sidebar in Kali Linux to open the command-line interface.





Step 2: Generating the SHA-256 Hash

The following command was executed to generate the SHA-256 hash of the downloaded ISO file and store the output in a text file:

sha256sum ~/Downloads/ucertify.iso > ~/Downloads/sha.txt

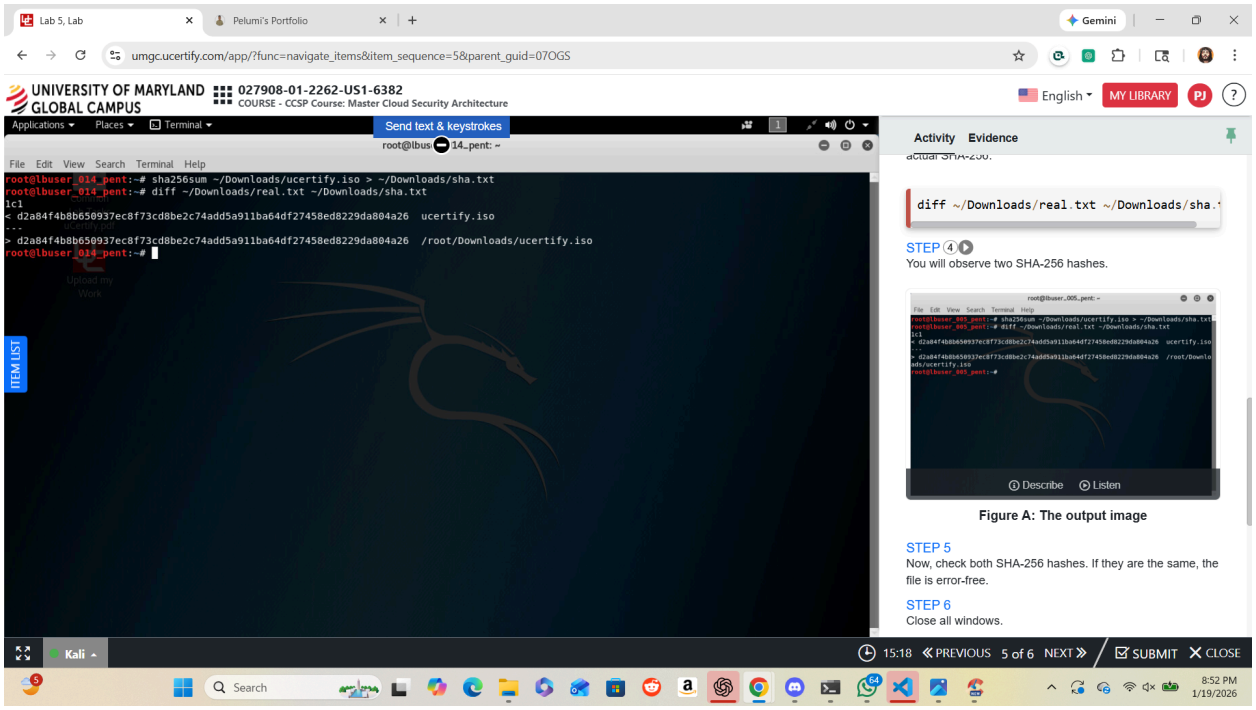
This command computed the SHA-256 hash and redirected the output to a file for comparison.

Step 3: Comparing the Hash Values

To compare the generated SHA-256 hash with the expected hash value, the following command was executed:

diff ~/Downloads/real.txt ~/Downloads/sha.txt

The diff command compared both files line by line to identify any differences.

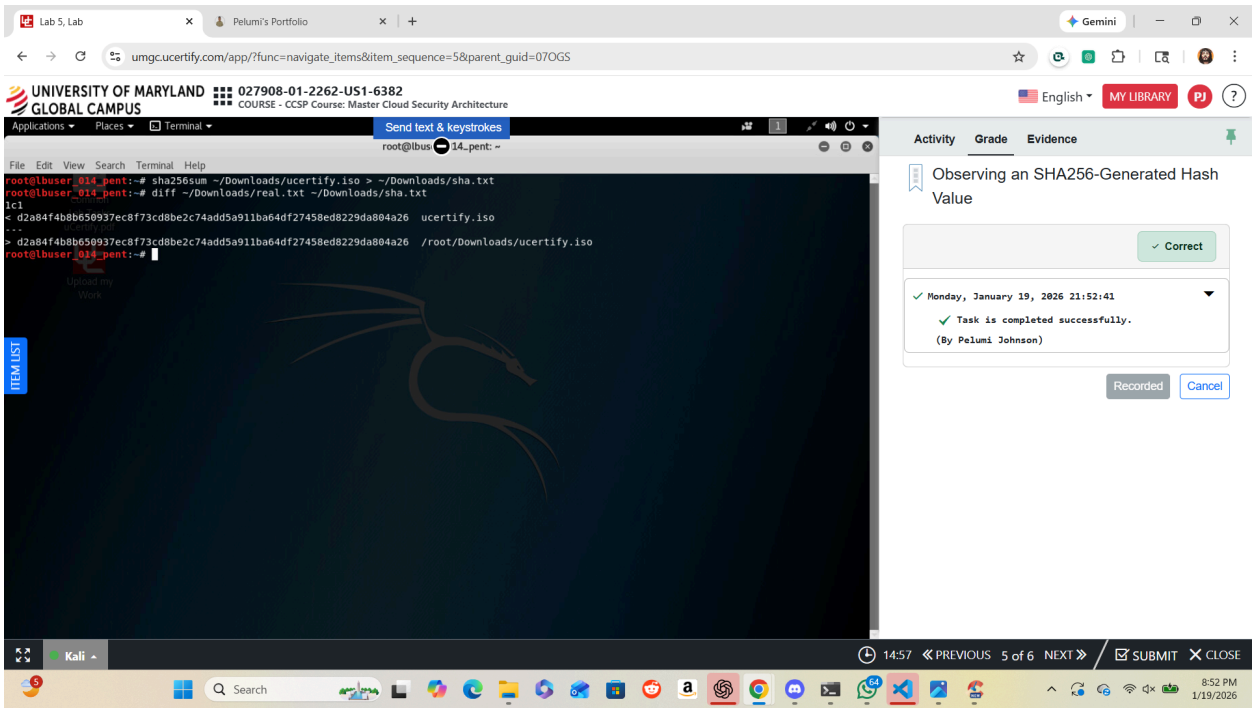


Step 4: Observing the Output

The Terminal displayed two identical SHA-256 hash values for the ucertify.iso file. No differences were reported by the diff command.

Step 5: Verifying File Integrity

Since both SHA-256 hash values matched, it was confirmed that the downloaded file was error-free and had not been altered.



Step 6: Closing All Windows

All open Terminal and lab windows were closed to complete the lab.

Results

- SHA-256 hash was successfully generated.
- The generated hash was compared with the expected value.
- Both SHA-256 hash values matched.
- File integrity was successfully verified.
- The lab task was completed successfully.

Conclusion

This lab demonstrated how SHA-256 hashing can be used to verify the integrity of a file in a Linux environment. Compared to MD5, SHA-256 provides stronger cryptographic security and is more resistant to collisions. Hash verification is a critical security practice in cloud computing, secure software distribution, and digital forensics.

Key Takeaways

- SHA-256 produces a 256-bit hash value.
- Even a small change in a file produces a completely different hash.
- The sha256sum command is used in Linux to generate SHA-256 hashes.
- Hash comparison ensures file authenticity and integrity.
- SHA-256 is more secure than MD5 for modern security applications.