

(Applied) Cryptography

Tutorial #5

Bernardo Portela (bernardo.portela@fc.up.pt) Rogério Reis (rvreis@fc.up.pt)

MSI/MCC/MERSI – 2025/2026

1 - Use OpenSSL to calculate the SHA256 value of the pdf slides of this week's class. Check if it equals:

4748dfe624d759009e3486fd062cd21dcf1858be435ea51c7ec7b4e53f52e02e

1.1 - What does this tell you about the integrity of the file?

1.2 - Suppose you alter the first 4 bytes of the original pdf file, and recompute the SHA256 value of this altered file. How many bytes do you expect to be affected by this change?

2 - Use python to crack the security of predictable passwords in `crack_hash.py`

- The file has the twenty most common passwords of 2019.
- The code produces hash values of passwords (salted and non-salted), then they are shuffled.
- From the shuffled hashes and the list of most common passwords, retrieve the original passwords!

2.1 - Follow the instructions given as comments in the code. Can you retrieve the passwords?

2.2 - Include a succinct analysis of how long it would take to do the following attacks:

- Cracking this set of passwords without salt
- Cracking this set of passwords with the same 1 byte salt, where all hashes use the same salt
- Cracking this set of passwords with the same 8 byte salt, where all hashes use the same salt
- Cracking this set of passwords with the same 8 byte salt, where all hashes use different salts

3 - Use the tool available [here](#) (or any other tool that works) to construct two PDFs with the same SHA-1 value. Check out the SHAttered paper and explain how the attack works.

4 - A length extension attack works as follows.

- Application generates secret key k , which is kept hidden
- At some point application computes $h = H(k||m)$ for some message m and publishes (m, h) .
- Intuitively it should be impossible for some attacker to compute $H(k||m')$ for $m \neq m'$.
- However, for some hash functions, it is possible to compute such a value using only (M, h) . This technique has been explained in theoretical classes for the SHA-2 family. Demonstrate the attack by constructing:
 - A Python program that generates k , computes $h = SHA2(k||m)$ for some m and saves k, m and h into different files.
 - Another Python program that reads m and h (but not k !) and generates some m' and h' into different files. It must be the case that $SHA2(k||m') = h'$ and that $m \neq m'$.