Final Project
Cryptography
2021-2

In this first programming project, you and your team will implement a program in which you compare the efficiency of the algorithms shown below. Remember that, on this occasion, you do not need to implement the algorithms, you only use the implementations that the programming language you choose has. This implies the selection of a library that already implements the standard algorithms. To do this, you need to use the testing vectors provided on the following table. These will allow you to measure the time your computer takes executing each algorithm.

Algorithm	**Size**	**Testing Vectors**
AES-EBC	Key Size 256 bits	https://csrc.nist.gov/CSRC/media/Projects/Cryptographic-Algorithm-Validation-Program/documents/aes/AESAVS.pdf
AES-CBC	Key size 256 bits	https://csrc.nist.gov/CSRC/media/Projects/Cryptographic-Algorithm-Validation-Program/documents/aes/AESAVS.pdf
SHA-2	Hash size 384 bits	https://www.cosic.esat.kuleuven.be/nessie/testvectors/hash/sha/Sha-2-384.unverified.test-vectors
SHA-2	Hash size 512 bits	https://www.cosic.esat.kuleuven.be/nessie/testvectors/hash/sha/Sha-2-512.unverified.test-vectors
SHA-3	Hash size 384 bits	https://csrc.nist.gov/projects/cryptographic-algorithm-validation-program/secure-hashing
SHA-3	Hash size 512 bits	https://csrc.nist.gov/projects/cryptographic-algorithm-validation-program/secure-hashing
RSA-OAEP	1024 bits	https://www.cosic.esat.kuleuven.be/nessie/testvectors/asymenc/rsa-oaep/RSA-OAEP-1024.unverified.test-vectors
RSA-PSS	1024 bits	https://www.cosic.esat.kuleuven.be/nessie/testvectors/sign/rsa-pss/RSA-PSS-1024.unverified.test-vectors
DSA	1024 bits	https://csrc.nist.gov/Projects/cryptographic-algorithm-validation-program/digital-signatures
ECDSA Prime Field	ECDSA, 521 Bits (Prime Field)	https://www.rfc-editor.org/rfc/rfc6979.txt
ECDSA Binary Field	ECDSA, 571 Bits (Binary Field, Koblitz Curve)	https://www.rfc-editor.org/rfc/rfc6979.txt

Each algorithm is used for some goal; therefore, you need to compare only the ones that share such a goal. For example, if you want to compare hashing algorithms you compare the efficiency of SHA-2 and SHA-3 by using the same input testing vector.
Following this idea, you need to create a table comparing the efficiency of these algorithms for the following operations:

- Encryption
- Decryption
- Hashing
- Signing
- Verifying

After the execution of the program, you should show a table showing the results for each operation. Coming back to the hashing example, after the execution of all hashing algorithms with all the hashing vectors, you should show a table similar to the following:

Hashing Operations

	SHA-2 384	SHA-2 512	SHA-3 384	SHA-3 512
Vector 1	<time taken to hash vector 1 with SHA-2>	<time taken to hash vector 1 with SHA-2>	<time taken to hash vector 1 with SHA-3>	<time taken to hash vector 1 with SHA-3>
Vector 2	<time taken to hash vector 2 with SHA-2>	<time taken to hash vector 2 with SHA-2>	<time taken to hash vector 2 with SHA-3>	<time taken to hash vector 2 with SHA-3>
...				
Vector n	<time taken to hash vector n with SHA-2>	<time taken to hash vector n with SHA-2>	<time taken to hash vector n with SHA-3>	<time taken to hash vector n with SHA-3>

Finally, you need to create a report of this work in which you discuss:

- How many test vectors you use for each algorithm.
- Why you decided to use that quantity.
- How you calculate the average time of execution
- What does that time says about each algorithm
- For each classification, which algorithm is the best? Why?

Check the specific instructions for the report on the corresponding space on Schoology.

References

- NIST Official Site for testing Vectors
  http://csrc.nist.gov/groups/STM/cavp/

- European Project for Security Testing
  https://www.cosic.esat.kuleuven.be/nessie/testvectors/