Developing an application that uses cryptography

Aim

The objective of this practice is for students to know and learn to use cryptographic libraries in order to consolidate the cryptographic concepts studied in theory. Thus, a statement is provided to create a program/ application, whose functionality must be chosen by the students, but which must perform a series of cryptographic operations.

Description

The program that must be implemented in this practice must be carried out in Python or Java and must implement the following cryptographic functions:

1. User authentication 2.
Symmetric (or asymmetric) encryption/decryption
3. Generation/verification of message authentication labels (eg, with functions hashing and HMAC)
4. Digital signature generation/verification
5. Authentication of public keys using certificates (PKI deployment)

At all times it is necessary to use algorithms that are currently used and that have not been compromised. So, for example, DES should not be used, and AES should be used instead.

1. User authentication

Each group can choose the authentication method that it considers most appropriate in each case:

- Based on something we know: passwords; They should be stored conveniently and ideally robust. This will be the option that will be required to be implemented in all cases at a minimum.

- Based on something we have: token, which could be a message to the mobile phone (although SMS is not the best alternative), a type of card, etc. There are multiple alternatives.
- Based on something we are: biometric trait, from the fingerprint, to the facial image or the iris, among others.

2. Symmetric (or asymmetric) encryption and decryption

At some point, in the system to be developed, encryption and decryption of information must occur, and the result of said operations can be seen. Note that if encryption (or any of the following cryptographic operations) is applied, for example, in communications or is transparent to the user, the result must be displayed in a log or in a debug message, along with the type algorithm and the key length used.

Regarding key generation, the following must be considered:

- The keys must have an appropriate length and in relation to the algorithm being used.
  you are using.

3. Generation/verification of message authentication labels

Valuable information that is exchanged or stored in addition to being encrypted must be authenticated. Message authentication code (or tag) algorithms provide this service.

4. Digital signature generation/verification

Digital signatures issued on some type of information will be generated and verified. In the minimum case, the system itself will generate the signatures, but depending on the needs of the system, it could be appropriate for the users themselves to generate them. The verification will at least be carried out automatically (reflecting the result in a log or debugging message) or allowing users to request said verification.

Regarding key generation, the following must be considered:

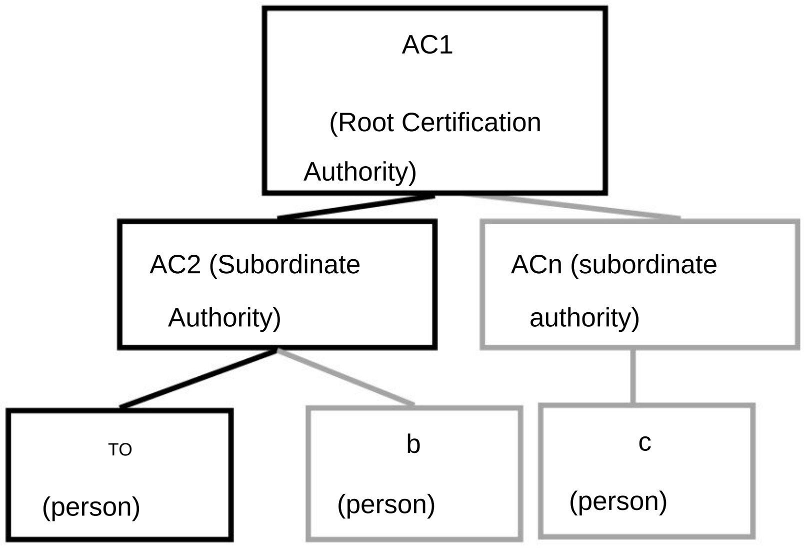
- Asymmetric keys must have an appropriate length and in relation to the
  algorithm being used.

5. Authentication of public keys using certificates (PKI deployment)

Each group becomes a ROOT CERTIFICATION AUTHORITY (as it can be in the real world, the National Mint and Stamp Factory). Said Authority (AC1) will have a self-signed certificate.

For organizational reasons (for example, to have a delegation in each autonomous community) it could be convenient to have several SUBORDINATED CERTIFICATION AUTHORITIES (AC2,..., ACn), which would be dedicated to issuing public key certificates to people (A, B, C), as shown in the following image.

Thus, the public key infrastructures (PKIs) deployed will be composed of at least one root AC (AC1) and, highly recommended, by a subordinate AC (AC2).
Note that this PKI or another PKI can be created with more levels or with a greater number of subordinate authorities.


AC1
(Root Certification Authority)
AC2 (Subordinate Authority)
ACn (subordinate authority)
TO
(person)
b
(person)
c
(person)

Public key certificates will be issued to all users (people) who want or need to use the private keys (that is, at least for the entities that generate digital signatures, and if asymmetric encryption has been included, for those entities that use it).

Key generation and storage

Keys are elements that must be protected against possible attacks, although the differences between symmetric, asymmetric encryption, digital signature and HMAC must be considered:

- Symmetric: since the encryption key is the same as the decryption key, it could be:
  - o Stored in a file/database. Since this key is secret, it could happen that it is stored with some type of protection (for example, encrypted with a password entered by the user).
  - o It is also possible not to store the key:
    - § If the user memorizes it and provides it at moments necessary.
    - § If it is generated from the user's password.
- Message authentication labels: a single key is used in the generation of MACs, so the considerations are those established for symmetric encryption.
- Asymmetric encryption/digital signature: the encryption and decryption keys (generation and verification in the case of the signature) are different and the most common thing is that, given their length, they are not entered by users, but are created and subsequently , the user could use them because they are stored in a file/database and the public or private one is selected, as appropriate. Access to the private key may be protected and may only be accessible through a password. To authenticate the public keys, the PKI will be used as described above.

We must emphasize the importance of keys having the appropriate length and entropy, as well as being stored appropriately.

Improvements

When creating a program or application, there are a large number of improvements that can be made considering the need for cryptography and security. Some of the possible improvements could be the following:

- Database storage.
- Use of different modes of operation (safety).
- Validation of data entered by a user – many attacks begin by not validating user input. Therefore, not assuming that data is entered correctly and validating it is a good security practice.
- Key rotation.
- Other improvements established by the student and that are conveniently justified (they must be accepted as such by teachers).

Assessment

The following table summarizes the qualification that is possible to obtain in this practice.

Criterion	Maximum score	Assessment
Development:		
User authentication	1	Eval. 1
Symmetric/asymmetric encryption	0.75	Eval. 1
Message Authentication Labels	0.75	Eval. 1
Digital signature	1	Eval. 2
Certificates and PKI	1	Eval. 2
Complexity and design of the developed application	0.5	Eval. 2
Enhancements (optional)	1 (additional)	Eval. 2
Total development	5 (potentially +1)	
Documentation and defense:		
Memories	1+1	Eval. 1/Eval. 2
Defenses	1.5 + 1.5	Eval. 1/Eval. 2
Total documentation and defense	5	
Total practice	10	

Deliverables

2 deliverables will be made, one of them associated with the aspects labeled as Eval. 1 in the table and another associated with the aspects labeled as Eval. 2 in the table.

Each of the deliverables must respond to the questions posed here and not exceed the indicated length. In addition, the associated code must be presented, which must be properly documented and using good programming and software design practices. If these criteria are not followed there will be a penalty in the score.

Deliverable 1

Cluster:

Practice Group ID:

Name of all students:

Email from all students:

Code repository (link) if any:

Answer the next questions. Include screenshots that support your arguments.

• What is the purpose of your app? • How is
user authentication performed? What algorithms have you used and why?
Detail how user passwords are managed and whether keys are generated from them.

• What do you use symmetric encryption for? What algorithms have you used and why? How do you manage keys? Explain the same aspects if asymmetric encryption is used for this type of encryption. • What
do you use the Message Authentication Code (MAC) features for? That algorithms have you used and why? How do you manage the key/s?

The page limit is 5, excluding the cover and annexes. The font size must be 11, with single spacing, Calibri/Arial or Times New Roman type.

Deliverable 2

Cluster:

Practice Group ID:

Name of all students:

Email from all students:

Code repository (link) if any:

Answer the next questions. Include screenshots that support your arguments.

• What is the purpose of your app? (repeat what was indicated in report 1)
• What do you use the digital signature for? What algorithms have you used and why? How I know Do they manage and store keys and signatures?
• How are public key certificates generated? What hierarchy of certification authorities has been deployed? Why have you chosen this configuration and not another?
How has it been implemented? When are certificates used and for what? • Discuss the
complexity and design of your application code. • If you have made
improvements, explain which ones and the security implications of each.
them in your program/application.

The page limit is 10, excluding the cover and annexes. The font size must be 11, with single spacing, Calibri/Arial or Times New Roman type.