University of Cape Town

Department of Computer Science

CSC4026Z Network and Internetwork Security

April - May 2024

Introduction:

This tutorial serves as the practical component for CSC4026Z and is intended to be completed in groups of **three**. The primary objective of this practical is to provide hands-on experience with cryptographic functions and enhance your understanding of protocols. Specifically, it involves the secure exchange of encrypted images along with captions between two parties. This will be achieved by implementing a compact Pretty Good Privacy (PGP) cryptosystem that combines various cryptographic techniques, including shared key encryption, public-key encryption, and certificate-based authentication. The primary focus is on ensuring key authenticity validation, and replicating the message confidentiality and authentication aspects of PGP.

Task Description:

Develop a network client application designed to establish a secure communication between "Alice" and "Bob". This involves the exchange and validation of public keys issued by a trusted Certification Authority (CA). Subsequently, messages will be secured using shared keys, private keys, public keys, hashing functions, and compression techniques, similar to the principles of PGP.

Specific Requirements:

Key Exchange: Both "Alice" and "Bob" must each possess:

- A private and public key pair.
- The public key of the trusted Certification Authority.
- A certificate containing their own public key, signed by a trusted Certification Authority.
- Alice and Bob jointly generate and share a secret key, e.g. using the Diffie-Hellman algorithm

Message Composition: Before applying cryptographic algorithms, the message structure should include:

- A text caption for an image.

- The image is encoded as a string.

Communication: Establish a communication system between the two clients, potentially using TCP. Both "Alice" and "Bob" can act as senders and receivers interchangeably.

Security Measures: Implement the following cryptographic functions:
- For asymmetric encryption and decryption, use RSA (algorithm: "RSA/ECB/PKCS1Padding").
- For symmetric encryption and decryption, use AES (algorithm: "AES/CBC/PKCS5Padding").
- Message hashing.

Message Exchange: Both "Alice" and "Bob" should be able to:
- Set up a connection for communication.
- Exchange certificates to establish trust.
- Load, encode, and decode image files and captions.
- Save decoded strings as files and display captions.
- Encrypt, compress, hash messages, and reverse these processes.
- Exchange encrypted messages.

Testing and Debugging: Include debugging statements to display essential information, such as encrypted messages, session keys, hashed messages, etc., to the console for documenting system runs.

Cryptographic and Implementation Details:

- The prescribed programming language is **Python**.
- Encourage the use of encryption libraries such as; you do not need to create your encryption libraries.
- Use RSA for public-key encryption and generate shared keys for symmetric encryption (e.g., DES, AES).

Documentation:

Prepare a concise write-up (up to 5 pages) explaining and documenting your implementation, including:
- Cryptosystem design.
- Communication connectivity model.
- Key management.

- Choice of cryptographic algorithms.
- Testing procedures and assumptions.
- Instructions on how to execute/run the submitted program(s) (can be written as a separate document, e.g., README).

The write-up must clearly indicate the team members.

Assessment Criteria:

This practical component constitutes 40% of the module assessment. The assessment will be based on the following aspects:

- Communications implementation (20 points).
- Security implementation (40 points).
- Overall system design, functionality and creativity to achieve the stated goal (20 points).
- Evidence of testing (20 points).

This assignment aims to *develop secure communication applications*, adhering to *cryptographic best practices*, and providing *clear documentation* for understanding and evaluation purposes.

To facilitate group coordination, please provide the names of your group members on a spreadsheet that will be shared by Lucas Carr CRRLUC003@myuct.ac.za, who is the TA for this course.

Submission Deadline:

The system, along with the write-up, is due on **Friday, 17 May 2024 at 23:55**, and must be submitted via Vula (one submission per group).