Monash University FIT5163

SEMESTER 2, 2024

Programming Assignment (30%)

Group Assignment

Due Date: 20 – 09 – 2024 (11:55pm)

− This is a group assignment. **Form a group of 3 students within your applied session ONLY. (2 students per group is only allowed for leftovers.)**

− You will present this work as a group in your **Week 11 Applied session.**

− **NO MARK WILL BE GIVEN TO THE WHOLE ASSIGNMENT if you have not come to the presentation** (without any acceptable reason, e.g. a medical certificate), even if you have submitted your report or your groupmate has presented the result. In that case, you would not pass the hurdle and will fail the unit!!!

− **FULL MARK: 30 (Report: 10 Marks. Presentation: 10 Marks. Source Code: 10 Marks.)**

– **You don't need to start from scratch!** You could use ChatGPT or other AI tools to generate the first version of the code for each of the following tasks or google a close or similar system/project from others. Check the issues, bugs, functionalities, and performances. Address the issues and bugs, and improve the functionalities and performance. (Note that starting with an AI tool or others' code is not necessarily more efficient. You might spend more time on debugging.)

– **Each group generates a git account or uses an existing one to commit all your changes,** the commit history should be demonstrated in your presentation and report**.**

– **If your code is not started from scratch, the differences between the first version (obtained from AI tools or others' code) and the final version should be highlighted in your report.** Identify all the issues of the first version and explain how you address the issues in your final version in the report and presentation.

Overview

This assignment is a programming project. You have the option to select one specific task from the list provided below (**Each task can only be selected by a maximum of two groups**) or propose your own project. However, if you choose to propose your own project, you **MUST submit a proposal to your tutor for quality auditing** before proceeding with the assignment. Please notify your tutor of the chosen task or the proposal via email. Once you have made your selection, you can implement the task using ANY programming language of your preference, such as Python, Java, C, C++, etc.

1. **Secure Chat Application:** create a secure chat application that allows two users to communicate securely over an untrusted network. You need to use the following techniques to ensure security of the application:
   a. Using end-to-end symmetric key encryption to encrypt messages sent between users.
   b. Implement a message integrity mechanism such that the receiver can detect any tampering or modification of received messages.
   c. Implement secure authentication mechanisms (e.g., username/password authentication) to verify user identities before allowing access to the secure chat application.
   d. The networking part is encouraged to be implemented and show the authentication and communication with two ends or vms. For instance, you can use socket or other tools. (optional, will give 2 bonus marks)

2. **Secure file transfer system:** Developing a secure file transfer system that allows users to upload and download files securely over an untrusted network and server:
   a. Implement end-to-end encryption to encrypt files before transmission (you can choose either symmetric key encryption or public key encryption, but the choice should be justified in the report).
   b. Include features for file integrity verification (e.g., hash checks).
   c. Define user roles (e.g., admin, regular user) and assign permissions (e.g., read-only, read-write) to each role. Implement access controls and user

permissions to restrict access to uploaded files based on user roles and permissions.

d. The networking part is encouraged to be implemented and show the authentication and communication with two ends or vms. For instance, you can use socket or other tools. (optional, will give 2 bonus marks)

3. **Implement linkable ring signature scheme.**
    a. Paper name: Linkable Spontaneous Anonymous Group Signature for Ad Hoc Groups (Liu et al. 2004).
    b. Paper link: https://eprint.iacr.org/2004/027.pdf
    c. Implement the following algorithms described in Section 4. (A LSAG Signature Scheme):
        i. Key generation
        ii. Signature Generation
        iii. Signature Verification
        iv. Link
    d. The code should consider using secure setup parameters.

4. **RSA attacks with a shared common factor**:
    a. Implement a program that generates multiple RSA key pairs with varying sizes (e.g., 1024, 2048, 3072 bits). Ensure that some of these generated key pairs have a shared common factor in their moduli.
    b. Develop an algorithm to discover a shared common factor among two or more RSA moduli.
    c. Implement and demonstrate the algorithm's effectiveness by finding shared factors in the generated RSA key pairs.
    d. Use the discovered shared factor to recover the private keys of RSA key pairs that share this factor.
    e. Implement the necessary mathematical operations to reconstruct the private keys and decrypt a sample encrypted message.

5. **Secure email application.** Create a secure email application that allows users to send and receive encrypted emails over an untrusted network. The application must use cryptographic techniques to ensure the confidentiality, integrity, and authenticity of email communications:
    a. Use end-to-end symmetric key encryption to encrypt emails.
    b. Implement a mechanism to detect any tampering or modification of received emails.
    c. Use digital signatures (e.g., RSA) to sign emails. The signature is required only for first time communication between email addresses (once verified, the email address will be added to a trust list).
    d. You can implement the system based on Postfix or other mail servers (note that the server is considered as untrusted. The confidentiality and integrity of emails should be protected from the server. Postfix's TLS feature cannot achieve that.)

6. **Implement the Boneh-Franklin IBE scheme**.
    a. Paper name: Identity-Based Encryption from the Weil Pairing (Boneh and Franklin, 2001)
    b. Paper link: https://eprint.iacr.org/2001/090.pdf
    c. Implement the following algorithms in Section 4.1 (BasicIndent):
        i. Setup
        ii. Extract
        iii. Encrypt
        iv. Decrypt
    d. The code should consider using secure setup parameters.

7. **Diffie-Hellman key exchange:** Design and implement a secure Diffie-Hellman Key Exchange protocol that is resistant to man-in-the-middle attacks, and extend it to to Password Authenticated Key Exchange (PAKE) that resists off-line password dictionary attacks:
    a. Implement the Diffie-Hellman key exchange protocol.

   b. Allow two parties to establish a shared secret key securely over an insecure channel.
   c. Extend your Diffie-Hellman protocol implementation to PAKE protocol, such as the OPAQUE protocol described in Sec. 6 of this research paper.

8. **Timing side channel attack:** Extract RSA secret key by attacking the left-to-right square-and-multiply algorithm with the timing side-channel attack:
   a. Implement a text-book RSA encryption with small modulus (e.g., 64 bits).
   b. Implement the left-to-right square-and-multiply algorithm for modular exponentiation, which is commonly used in RSA encryption and decryption. Ensure the algorithm handles both encryption and decryption operations.
   c. Implement a timing side-channel attack mechanism to extract the secret key of the RSA scheme (use variations in execution time caused by conditional branches or loop iterations within the square-and-multiply algorithm to infer information about the secret key).

9. **Public key certificate system**: Design a public key certificate system that involves one root CA, two sub-CAs and three clients:
   a. Implement functionalities for the root CA to generate its own private/public key pair.
   b. Develop functions for issuing certificates for clients, including necessary attributes such as client ID, public key, and validity period.
   c. Implement client registration functionality, allowing clients to provide their identity and public key.
   d. Develop a mechanism for clients to submit certificate requests to the CA. The request should be encrypted.
   e. Develop methods for validating certificates using the corresponding public keys and CA signatures.
   f. Implement mechanisms for certificate revocation in case of compromise or expiration.

10. **Elgamal encryption to encrypt audio files**: Implement Elgamal encryption algorithm (can be found from Section 8.4 in “Handbook of Applied Cryptography”. Open access link is provided: https://cacr.uwaterloo.ca/hac/about/chap8.pdf)

    a. **Audio File Format**: The input audio file should be in WAV format for simplicity.

    b. **Key Generation**:

       - Generate a public and private key pair for ElGamal encryption.
       - Students should choose an appropriate large prime number p and generator g.

    c. **Encryption**: encrypt the audio file with ElGamal encryption algorithm

    d. **Decryption**: decrypt the encrypted audio file into its original form

    e. **Validation**:

       - Compare the decrypted audio file to the original audio file.
       - Ensure that the decrypted audio file matches the original in terms of integrity and quality.

11. **Secure IoT device communication:**

    a. Implement a lightweight encryption protocol (e.g., TLS/DTLS) for secure communication between IoT devices and a central server.

    b. Ensure mutual authentication between IoT devices and the server using certificates or pre-shared keys.

    c. We can use simulated IoT devices and servers. The important thing is we need to use lightweight primitives on the IoT sides as they have constrained resources.

12. **EMV card payment protocol.** Implement a demo of the Eurocard / Mastercard / Visa (EMV) credit chip card payment protocol.

    a. You can find a description of the EMV protocol in Sec. III of the following research paper:

David A. Basin, Ralf Sasse, Jorge Toro-Pozo. *The EMV Standard: Break, Fix, Verify*. IEEE Security & Privacy Symposium 2021, pp. 1766-1781.

b. Your EMV demo implementation can run on a single PC, but should implement and output the functionality of, and the protocol messages exchanged between, the three parties involved in the protocol: (1) the credit chip card, (2) the merchant's card reader terminal, and (3) the customer's bank that issued the card.

13. **Secure voting system.** Develop a secure online voting system that ensures the integrity, confidentiality, and verifiability of votes.
    a. Implement a secure voting protocol that uses cryptographic techniques to ensure vote confidentiality and integrity.
    b. Develop a user interface for voter registration, voting, and result viewing.
    c. Ensure voter authentication and authorization, preventing double voting and unauthorised access.
    d. Implement a mechanism for voters to verify that their vote was counted without revealing their choice.(optional, will give 1 bonus mark)
    e. Use blockchain technology to provide an immutable audit trail of votes (optional, will give 2 bonus marks).

14. **Secure electronic health record (EHR) system.** Develop a secure electronic health record system that ensures patient data privacy and integrity.
    a. Implement a secure EHR backend service that encrypts patient data using AES-256.
    b. Use role-based access control (RBAC, explore this on your own) to ensure that only authorized personnel can access specific records.
    c. Alternatively, you could use attribute-based encryption (ABE) to achieve confidentiality and access control.
    d. Develop a user interface for patients and healthcare providers to access and manage health records securely.

e. Implement audit logging to track access and modifications to health records. (optional, will give 1 bonus marks)

f. Integrate with a secure messaging system for healthcare providers to communicate sensitive patient information. (optional, will give 2 bonus marks)

15. Searchable encryption protocol.

a. Implement a Searchable Encryption protocol from any research paper you choose (however, you must understand the protocol) - the OXT protocol introduced in this paper is suggested, as it can be understood using cryptographic techniques covered in this unit.

b. Your protocol implementation should include both

   i. Encrypted Database Setup Algorithm, allowing a client to encrypt a database to be uploaded to the server

   ii. Search protocol allowing the client to issue an encrypted search query, obtain an encrypted search query result from the encrypted database server, and then decrypt the search results.

16. A security system of your choice: Implement the system proposed in a research paper (whose source code is not publicly available). **You need to consult your tutor about the selected paper before starting to implement the system.**

Additionally, you will need to write a report discussing the design, implementation, and security analysis of your application. The report should be **at least one page long but not exceeding five pages,** formatted with 11-point font, single column, normal margins, and default line spacing.

You will submit your source code and the report separately (see the specification below). **Each group only needs to submit files by only one of the group members.** You will also present your code during your **Week 11 Applied class** slot (**Maximum 15 minutes + 5 minutes Q & A**).

Submission Requirements:

All the following files should be uploaded to Moodle and **only one group member needs to upload the files. There is a mark deduction for any missing document.**

1. A PDF report
2. A zip file containing
    a. Assignment Cover Sheet for the group
    b. The source code

Note: Hand-written report is NOT accepted. No mark will be given to any hand-written report.

Late Submission:

Late Assignments or extensions will not be accepted unless you submit a special consideration form and provide valid documentation such as a medical certificate **prior** to the submission deadline (NOT after). Otherwise, there will be a **5% penalty per day** including weekends.

PLEASE NOTE.

Before submitting your assignment, please make sure that you haven't breached the University plagiarism and cheating policy. It is the student's responsibility to make themselves familiar with the contents of these documents.

Please also note the following from the Plagiarism Procedures of Monash, available at https://www.monash.edu/policy-bank/policies-and-procedures/academic/education/student-academic-integrity-procedure