Course Project

Due dates: **Friday, 09/08/2023** at midnight (project proposal),

Friday, 10/20/2023 Office hour (midterm project discussion),

Sunday, 12/10/2023 at midnight (final report)

Preliminaries

Implementation projects involve the implementation and experimentation with a topic of interest. Typical outcomes are building a useful tool, testing a hypothesis, or automating cryptanalysis attacks, to name a few. Implementation projects must provide an extensive description of the experimentation setup, the experiments performed and the validity of the results. For example, you may build an automated tool for intelligently finding collisions of well-known hash functions and describe any optimizations you have opted for.

You can work on teams of three people of your own choosing, in order to expand the scope of your project. You may also work on your own if you wish to. Regardless of project type, you are expected to generate a final report. All projects will be invited to do a final in-class presentation to showcase your findings.

Projects that combine research and experimentation components are always welcome.

The project Timeline and Important Dates

- *Project Proposal*. Please start thinking about your project topic immediately and try to fix that within a week or so. The following aspects should be included:

  - The title of your project, and the category and topic you choose;
  - Your motivation of choosing the topic and the challenges in that topic;
  - Any hardware required (Radio testbed, test WiFi router, WiFi receiver, etc.)
  - Outline the things you will do and the results you plan to submit for the project.
  - Name of group members.

Some sample topics for both type of projects are listed in the project description. A written one-page project proposal will be submitted in SoC Webhandin before the deadline of **Friday, 09/08/2023** at midnight. Please do not hesitate to discuss with me whenever there are doubts on choosing your topic. Project proposal accounts for 10% of the project grade. Grade will be judged on the clarity of the motivation and outline of planned outcome.

- *Midterm Report Discussion.* We will have in-person discussions on the advancement on the project on during office hour **Friday, 10/20/2023**.

- *Final Project Report*: Please start doing your project once you have determined the topic (and don't wait until after 10/18).

Final Presentations and Slides (20 points) Slides Due: **Sunday, 11/26/2023** 11:59PM, Presentation 11/27/2023 and 12/01/2023 9.30 – 10.20 AM.

At least 15 min presentation for each group. All members should participate in the presentation. Slides due on the SoC webhandin. Suggested topics to be covered: Introduction, Problem Definition and Objectives, Demo, Steps taken to achieve the result, explanation of the result.

Final Report (45 Points) Due: **Sunday, 12/10/2023** 11:59PM Grade assignment: Abstract: 5; Introduction: 10; Problem Definition and Objectives: 10; Related Work: 5; Methods: 10; Result: 5; Conclusion and Future work: Optional.
Each team should provide a self-contained, readable final report. The following format is suggested but you do not have to follow it exactly.

1. Title and abstract (5 points).
2. Introduction -- Include background material and motivation (10 points).
3. Statement of the research problem(s): outline what is the common goals to achieve, such as system model, security requirements, threat models (types of adversaries existing work assume and what are the usual goals of the adversary), and major technical challenges in this domain. (10 points).
4. Main Body – survey of the state-of-the-art (summarizing the ideas and results of major existing works/solutions, from classical ones up to the most recent results), and critically analyze/compare the advantages/disadvantages of current approaches (e.g., in terms of security, performance and/or usability). (15 points)
5. Conclusions – what you have learnt from this survey so far. (2.5 points)
6. Future work and open problems: analyze the open problems/challenges (with your own opinion) and suggest future research directions (discuss how you may solve them). (2.5 points)
7. References.

Project Demo Review (20 points) Due: 12/10/2023 11:59PM.

All the project presentation/demo will be in class and will not be recorded. The pdfs of the slides are available under the resources in piazza. Please review all except your own. If you are not present during the presentations and submit the report, you will lose 10 points automatically. For each of them complete the following:

Group members:
Title:
What did they accomplish:
What was their procedure:
What are their future ideas:
Your critique:
Your rating (out of 5):
You can complete this in a single document file and upload it to the SoC web Handin.

Note that: You can work on and write your own project independently or a team of up to 3 students.

Implementation Guidelines

You are expected to implement the core functionalities by yourself. In case that the functionalities require low-level functions that existing libraries exist, you can use the existing libraries, however you should not reuse existing implementations for high level functions. For an example, suppose you are implementing the Kerberos network authentication protocol, existing library for encryption primitives (such as OpenSSL) can be used so that you don't need to re-implement the encryption schemes, but everything else should be implemented by yourself. However, if you project is to implement RSA itself, then you cannot directly use existing library which contains RSA. Rather, you may use some library for low-level mathematical operations, such as GNU GMP library. Or, suppose you want to implement a pairing-based cryptosystem such as identity-based encryption, you can use the PBC library which is a free C library that performs the underlying mathematical operations (bilinear pairing). If you use any libraries, you should clearly identify it in your report and cite the sources.

The goal of this project is NOT to directly implement the cryptographic primitive themselves that are learnt in the class or are already provided by an existing online library (e.g., AES, DES, Hash functions). Instead, you are encouraged to implement something based on the existing cryptographic primitives, or to gain deeper insights on the security and applications of these schemes. In general, it should be beneficial to your learning and understanding of information and network security. Of course, it should require an effort level commensurate to the project period (about 5 weeks). For the programming language, you can choose the one you are most familiar with, such as C/C++, Java, Matlab, etc. However, Matlab may only be suitable for a few projects.

You will need to submit both a report and codes used for your implementation and experimentation. The report should include: the title, abstract, introduction (Include background material, motivation, project scope and overview of what you have done in your project), description of the scheme(s) you implemented, your approach of implementation (high level flow as well as detailed steps), experimental results in terms of figures or tables (screenshots may be included if necessary), analysis and/or comparison of the results, and conclusion. Also, in your appendix, please include a description of the organization of your code, platform, and how to compile & run your code. You can also append your code to the report.

Grading: Your grade will be judged on the completeness and correctness of the functionalities (30%), the quality of presented results (30%), and the overall quality (e.g., organization, and readability) of your final report (30%).

A non-exhaustive list of suggested implementation topics (you may propose your own):

1. Implement the five modes (ECB, CBC, CFB, OFB, CNT) to encrypt large messages using any block cipher of your choice (DES, AES, etc). Benchmark and compare their performance. Try to use repeated message blocks in different input messages and see the resulting ciphertext. Try also to swap the ciphertext blocks and modify some of the blocks and see the decrypted messages.
2. Cryptanalysis. Write programs to implement brute force attacks against a public key cryptosystem (a shorter key version). For example, factorization of large numbers to break RSA, solving the discrete-logarithm problem such as Pollard-Rho algorithm in order to break Diffie-Hellman or ElGamal encryption.
3. Primality testing and factoring (implementation of several algorithms for primality testing, complexity comparison, deterministic vs. probabilistic tests, etc.)

4. Hashing -- Find collisions, preimages, and second preimages in hashes with small ranges and domains. First familiarize yourself with efficient ways of finding collisions.
5. Side-channel attacks on cryptographic systems. Finding cryptographic keys via power analysis, or timing attacks, Fault Based Cryptanalysis (good project for people with background in hardware).
6. Set up an IPsec tunnel between two computers, or an IPsec VPN between two routers (if you have the resource), and record the entire transaction (e.g., using a packet sniffer software such as WireShark). In the recorded transaction, mark the message(s) that performed a certain step in the protocol (e.g., Phase 1 IKE crypto negotiation, session key establishment, proof of identity, Phase 2 IKE).
7. A basic, intermediate, advanced or esoteric cryptographic protocol, such as authentication services, key distribution, secret splitting, secret sharing (threshold schemes), etc.
8. A password generation/authentication scheme from the textbook/reference books or that is currently used in practice (or any scheme that you like), and thoroughly evaluating its usability and security (e.g., vulnerability to dictionary attacks). Try brute force attacks to break different password hashing schemes and record the average time needed.
9. The Kerberos network authentication protocol (V4, V5).
10. Implement the PGP protocol, and analyze its strengths and vulnerabilities.
11. The generation of cryptographically-secure (pseudo)random numbers used for keys, IVs, and other cryptographic parameters.
12. Implement the blockchain used in bitcoins. Include essential features, such as the record of transactions, hash chain, digital signatures, and proof-of-work generation.
13. In cybersecurity, the strength of a password is one of the essential elements for a strong password. This password must include alphabets, digits & different symbols. For that, you can create a tool to check the password strength and informs you if it is secure to utilize or not. This project can be done by using Python.
14. A free tool like Pi-Hole is mainly designed to block unnecessary advertisements from your home n/w completely. These advertisements can be injected into scams & malicious software by hackers and scammers to begin phone scams. This Pi-Hole could help you from above just annoyance. Pi-hole installation is a very simple method to block the ads for the protection of data.
15. Adequate security of information and information systems is a fundamental management responsibility. Nearly all applications that deal with financial, privacy, safety, or defense include some form of access (authorization) control. Access control is concerned with determining the allowed activities of legitimate users, mediating every attempt by a user to access a resource in the system. In some systems, complete access is granted after s successful authentication of the user.
16. Secret-free access control.