Newbies

Project	Capacity
• House of classical ciphers **(GUI)** ○ Implement 6 classical ciphers and their cracking algorithms. ○ Cracking can be brute force or other analysis methods ○ You must explain each cipher and how to break it.	2 teams
• Secure data communication and key exchange ○ A client-server application to establish a secure connection between a client and a server thorough a key-exchange protocol ○ After exchanging a secret key, the users create a shared symmetric key to encrypt the data using a symmetric encryption algorithm	2 teams

Average

Project	Capacity
• Parallel Blake3 hash function ○ Implement Blake3 hash function ○ Parallelize the computation ○ Explain the algorithm and how it works ○ Demonstrate the performance on large files	2 teams
• Password manager with cryptographic Security **(GUI)** ○ Implement a tool that reads plaintext password from a user, encrypting it, and storing it. ○ You must use secure cryptographic algorithms to maintain the confidentiality of the passwords ○ You must ensure the integrity of the encrypted passwords. ○ The main application must be protected by a username and a master password	2 teams
• Hash cracking tool ○ Implement a basic tool (similar to JohnTheRipper and HashCat) to crack 3 types of hashes. ○ The program should be multithreaded ○ Your program must maintain a wordlist of common passwords and their hashes to recover the hash from it ○ (optional) can you implement a hash type detection tool?	3 teams
• End2End chat application **(GUI)** ○ A secure instant message application that encrypts the messages between two users ○ The application allows each user to have a public key and a private key. ○ When two users start a conversation, they exchange a shared secret key. ○ The shared secret key is passed to a KDF to generate a symmetric secret key to be used with a **stream cipher** ○ The exchanged messages are encrypted with the stream cipher ○ **Your program must do all the encryption/decryption operations automatically.**	2 teams

Crypto affection

Project	Capacity

• (Educational) Public-key crypto is fun **(GUI)**
  ○ Choose 3 algorithms of set below to implement.
  ○ Your program should be used to teach students the nuts and bolts of public key cryptographic algorithms
  ○ You program must demonstrate the math and the steps of key generation, encryption, and decryption
Suggested algorithms: RSA, DH/ElGamal, Elliptic curves, Rabin Cryptosystem
	3 teams

• Secure Multiparty Computation (MPC)
  ○ Solve at least two of the open issues related to secure multiparty computations using the Nada framework, available here: here https://github.com/NillionNetwork/nada-by-example/issues
  ○ You should fork the repo to your profile and work on it.
  ○ You must explain what MPC is
	2 teams

• Simulate MQV (Menezes–Qu–Vanstone) key exchange protocol **(GUI)** ○ A program that simulates a client-server application	2 team

• (Educational) Side channels simulation-RSA
  ○ A program that simulates a side-channel attack on the RSA algorithm
  ○ Your program should be used to teach students how side-channel attack can exploit the RSA algorithm to recover the private key
	2 teams

• ECDSA
  ○ Implement the Elliptic Curve Digital Signature Algorithm to sign and verify PDF documents and images.
  ○ Implement a function that breaks the ECDSA algorithm given two signatures as explained in the class
	1 team

Math maniac

Project	Capacity

• PQS – NTRU cryptosystem
  ○ **This project mainly aims to exploring Post Quantum Cryptography**
  ○ The focus is on the NTRU cryptosystem
  ○ Implement (basic implementation) the key generation, encryption, and decryption of the NTRU cryptosystem
  ○ **(optional) you can use existing libraries to demonstrate the NTRU cryptosystem in secure chat application**
• Helpful resources: https://github.com/pointedsphere/NTRU_python, https://medium.com/@vihren.stoev/the-essence-of-ntru-key-generation-encryption-decryption-7c0540ef8441
	3 teams

• PQS – Digital Signatures
  ○ **This project mainly aims to exploring Post Quantum Cryptography.**
  ○ The focus is on the ML-DSA standard for digital signatures
  ○ The goal is to explain how key generation, signature generation, and verification work (without from scratch implementation)
  ○ **Use existing libraries to demonstrate it in secure chat application**
• Helpful resources: https://github.com/itzmeanjan/ml-dsa, https://nvlpubs.nist.gov/nistpubs/fips/nist.fips.204.pdf#page=2.09
	3 teams

Other

- If you have any other ideas, discuss it with me.

General instructions

- Teams should consist of 3-5 people
- Choose your own programming language
- It's **very recommended** to develop GUI applications as a web application and, if possible, deploy it on GitHub (http://pages.github.com/)
- Account for 10-min quick presentation – no slides needed
- Due date: TBD