Subjects for Bachelor Projects in the Cryptography and Security Group.

The following descriptions of possible differ a bit in how detailed they are. This does not necessarily reflect how hard they will be for the students or how ambitious they are. All projects can be adjusted to fit your ambition and abilities.
There are many proposals and a lot of text, but if this area interests you, try to have a look at all the proposals. The last one may be exactly the one that interests you the most.

Post-Quantum Signatures

Most of the public-key encryption and signature schemes used in practice today are quantum-safe i.e., they can be broken in polynomial time using quantum computers. This is a serious threat: while it is hard to predict when a quantum computer will be built, it is still very important to have post-quantum cryptographic algorithms ready and deployed already today. Most techniques for post-quantum public-key encryption require a mathematical background which you don't have yet. However, there are relatively simple constructions of post-quantum signatures using hash-functions, which you can study and implement in this project.
Here is an indicative list of activities:

1. Read and understand signature schemes submitted to the NIST call. There are many schemes under various hardness assumptions, so there should be something for everyone.
2. Read and understand the parameter choices and implementation aspects for one particular scheme.
3. Implement and benchmark against the signature schemes you have developed during dDistSik.

Post-Quantum Encryption

Most of the public-key encryption and signature schemes used in practice today are not quantum-safe i.e., they can be broken in polynomial time using quantum computers. This is a serious threat: while it is hard to predict when a quantum computer will be built, it is still very important to have post-quantum cryptographic algorithms ready and deployed already today. NIST (a standards organisation in USA) have recently selected to standardize some algorithms for this purpose, after a lengthy competition. In this project, you can pick one of the algorithms they selected and study, implement and compare it with schemes you have studied in dDistSik.
For instance, the encryption scheme Kyber is based on lattices, a family of cryptographic schemes that are believed to resist quantum computers. Kyber requires some background on polynomial rings and FFT algorithms for fast polynomial multiplication, which you will also need to study.
References:

- Algorithms selected by NIST: https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022
- The FFT/NTT in Kyber: https://electricdusk.com/ntt.html

Advanced Consensus Algorithms

Most of you know something about this subject from the DISSY course. In this family of project ideas, you have several possibilities:

1) Implement a prototype consensus layer based on a recent research article that will work in a distributed setting, benchmark the protocol, describe the theory and relate the practical prototype and benchmarking to the theory. Papers can be selected based on interest.

2) Study some recent papers on blockchain consensus mechanisms and describe and relate the theories. Paper can be selected based on interest.

Here are some examples of papers which you could look at:

[Orestis Alpos, Christian Cachin, Simon Holmgaard Kamp, Jesper Buus Nielsen: **Practical Large-Scale Proof-Of-Stake Asynchronous Total-Order Broadcast**. AFT 2023: 31:1-31:22]

[Simon Holmgaard Kamp and Jesper BuusNielsen: Byzantine Agreement Decomposed: **Honest Majority Asynchronous Total-Order Broadcast from Reliable Broadcast and Leader Election**. IACR ePrint 2023.]

[Idit Keidar, Eleftherios Kokoris-Kogias, Oded Naor, and Alexander Spiegelman. **All you need is DAG**. In Avery Miller, Keren Censor-Hillel, and Janne H. Korhonen, editors, PODC ’21: ACM Symposium on Principles of Distributed Computing, Virtual Event, Italy, July 26-30, 2021, pages 165–175. ACM, 2021.]

[Bernardo David, Bernardo Magri, Christian Matt, Jesper Buus Nielsen, Daniel Tschudi: **GearBox: Optimal-size Shard Committees by Leveraging the Safety-Liveness Dichotomy**. CCS 2022: 683-696]

Electronic Voting

In this project, you will build and analyse a number of different solutions for implementing electronic voting in a client-server setting with various types of security guarantees, such as: only the final voting result becomes known, the result correctly reflects the votes cast, even if some participants are malicious. The clients will be the voters, and the servers will be responsible for collecting votes and computing the result.

Throughout the project you will be expected to analyse the security of the solutions you build using the basic concepts of security policy and threat model that you know from the Security course.

You will be given a number of templates for a series of solutions where each one is an extension of the previous one and will handle increasingly powerful adversaries. Your job will be to fill in the details, implement and analyse. At the end, if ambition and time permits you will also be looking for solutions in the literature on your own.

Another possibility is to combine this project with a study of an existing e-voting system such as Helios or ElectionGuard.

Here follows an indicative list of solution concepts you will encounter:

1. 2-server solution with privacy but no security against malicious attacks.
2. Extension to 3 servers and threshold security against 2 break-ins.
3. Fault detection in 3-server solution
4. 4-server solution with error correction.
5. Extension with user identification, study literature.
6. Extension with security against malicious users, study literature.

Secure Multi-Party Computation

Secure multi-party computation (MPC) is an advanced tool in cryptography, which allows several parties to jointly perform computations on their private data, while only revealing the result of the computation and not the private inputs held by each party.

In this project, you will study and implement some solutions for multi-party computation. The main goals could be:

1. Read and understand MPC protocols based on secret-sharing.
2. Implement a basic protocol for secure additions and multiplications.
3. Study an application of MPC, such as secure auctions, and implement it with your protocol.

Various extensions are possible, e.g. different types of auctions, comparing different techniques, protecting against malicious users etc.

Yao's Garbled Circuits: Optimizations

Secure two-party computation (2PC) allows two parties with private data to compute a joint function of that data (e.g. the intersection of their private sets) without learning anything else about one another's data. Yao's Garbled Circuits (YGC) is a tool often used in secure two-party computation. YGC enables one party - the garbler - to 'garble' a circuit C in such a way that another party - the evaluator - can evaluate the garbled circuit on an encoded input X and learn the output C(X), without learning anything else about X. If the evaluator obtains an encoding corresponding to her own and the garbler's data (without learning the garbler’s data in the process), this enables secure two-party computation.

Since the introduction of Yao's garbled circuits (by Yao, in 1982), there have been a number of optimizations to how garbling is done. Some of these focus on minimizing the number of encryption operations that the garbler / evaluator must do; others focus on minimizing the size of the garbled circuit.

The aims of this project are to:

1. Understand Yao's garbled circuits.
2. Understand and implement one or more of the existing optimizations.
3. Write an approachable explanation of these optimizations.

Private Set Intersection

Cryptography allows not only to securely communicate from A to B, but also to securely compute on private data. In this project you will study Private Set Intersection, which allows two parties A and B to compute the intersection of their sets without revealing any other information about each other's set. A simple example of an application where PSI is required is private contact discovery: you install an app and you want to figure out which of your contacts are already using the app. The server can't send you the entire list of their users, and you might not want to upload you entire phonebook to the server (this is what is most commonly done in practice). With PSI you could learn the intersection without disclosing your set. In this project you will:

1. Learn about simple but insecure hash-based PSI protocol.
2. Study advanced cryptographic protocols (such as oblivious pseudorandom functions), which are based on mathematical assumptions similar to the Diffie-Hellman problem (which you have heard about in the Network Security week of Distributed Systems and Security).
3. Implement and benchmark PSI protocols.

Secure Cloud Usage

In this project, you will build and analyse a number of different solutions for using a cloud service to share data with other users in a way that provides various types of security guarantees, such as: the cloud provider does not get the data involved, nor can it modify the data. If more than one cloud provider is involved one may want to guarantee that security holds, even if some of the providers are malicious.

Throughout the project you will be expected to analyse the security of the solutions you build using the basic concepts of security policy and threat model that you know from the Security course.

You will be given a number of templates for a series of solutions where each one is an extension of the previous one and will handle increasingly powerful adversaries. Your job will be to fill in the details, implement and analyse. At the end, if ambition and time permits you will also be looking for solutions in the literature on your own.

Here follows an indicative list of solution concepts you will encounter:

1. Single server solution with keys sent on separate channel.
2. Multiple server solution where some hold the data, some hold the keys.
3. Extend with various types of secret sharing to have security against off-line

attacks on servers.
4. Extend to security against malicious servers.
5. Combine with user identification. Study literature and standards for this.

(In)security of the GSM system
The well-known GSM system for mobile phones was designed to prevent various kinds of attacks, such as spoofing of identities and eavesdropping of phone conversations. In the first version of the system, the encryption and authentication algorithms used were secret. However, phones were soon reverse engineered and the algorithms became known. It was found that the algorithms were not as secure as the designers had hoped, and in fact, some were completely insecure.
In this project, you will study literature on the subject, learn how the attacks worked and learn about the counter measures that were taken in later versions of the system.
Here is an indicative list of activities:

1. Read literature, write summary on what attacks achieve and whether you think this poses realistic threats.
2. Take a closer look and implement one of the ciphers used in GSM and 1 or 2 selected attacks. Test you work.
3. Find material on what was done in GSM in more recent years after the attacks, summarize the updates and comment on what you think about them. Do they solve the problem?
4. If time and ambition permits, find out what was done for security in later generations of phone systems and comment on the solutions.

Attacks and Countermeasures for SSL/TLS encryption
In most communication settings, it is desirable to protect *confidentiality* as well as *integrity* of the data being transmitted. From the security course, you know that *block ciphers* with a good mode of use and *message authentication codes* (MAC) are the right tools for the job. However, there is a
long history of attacks on real world protocols, in particular the SSL/TLS suite. These attacks exploit that combining encryption schemes and MAC schemes is not as simple as one might think. In the recent TLS version 1.3, a lot of work has been done to avoid all these previous attacks
In this project, you will:

- Refresh your knowledge of modes of operations for block ciphers (CBC etc.);
- Understand how padding oracles can be used to break security of encryption schemes;
- Read about how this has been used in practice to break previous versions of widely

deployed cryptographic protocols such as the SSL/TLS protocol suite;

- Perhaps Implement the attacks in a simulated environment to assess their impact;
- Study literature about provably secure countermeasures to these kind of attacks (e.g.,
  authenticated encryption, TLS 1.3);
  By the end of the project you will understand what was the cause of some of serious vulnerabilities discovered in the SSL/TLS protocol suite such as the so-called Lucky13 and Poodle attacks, etc.

High assurance cryptography
Supervisors: Diego Aranha and Bas Spitters(from the Logic and Semantics group)

High Assurance Cryptography
https://github.com/hacspec/hacspec

Cryptography forms the basis of modern secure communication. However, its implementation often contains bugs. That's why modern browsers and the linux kernel use high assurance cryptography: one implements cryptography in a language with a precise semantics and proves that the program meets its specification.

Currently, IETF standards are only human-readable. The hacspec language (a safe subset of rust) makes them machine readable. In this project, you will write a reference implementation of a number of key cryptographic primitives, while at the same time specifying the implementation. You will use either semi-automatic or interactive tools to prove that the program satisfies the implementation.

There are a number of local companies interested in this technology. So, this work will be grounded in practice..

Anonymous Credentials & Selective Disclosure

In many digital-identity settings we want the holder to prove something about their attributes (e.g., “I am over 18”, “I am a student at University X”, “I hold driving-licence category B”) *without* revealing their full identity, and ideally without linking multiple presentations to the same person. Anonymous credentials (also called privacy-preserving or attribute-based credentials) allow exactly that: selective disclosure of attributes, often with zero-knowledge proofs, while still providing security properties such as authenticity of the attribute, binding to the holder, revocation, and unlinkability.

This topic is particularly timely, as the EU’s upcoming Digital Identity Wallet framework includes selective-disclosure mechanisms for privacy-preserving age verification (https://github.com/eu-digital-identity-wallet).

This project invites students to learn about this topic: start from a simple, standardised selective-disclosure protocol, implement and benchmark it, and then optionally extend the work towards more advanced anonymous credential systems (e.g., full zero-knowledge based constructions) and apply to a current real-world use case (such as age-verification or digital identity wallets).

Warning: this is a new project, on a quickly evolving topic. Students going into this project should expect to have to dig for material independently, and help define the exact scope of the project.

Recent blogs on selective disclosure in practice:
https://abhvio.us/posts/mdoc/
https://lowentropy.net/posts/selective-disclosure/
https://developer.chrome.com/blog/digital-credentials-api-origin-trial

Older papers:
https://dl.acm.org/doi/pdf/10.1145/586110.586114
https://www.zurich.ibm.com/pdf/security/idemix/BC2010-Mixing_Identities_with_Ease.pdf

Off-the-Record (OTR) Messaging

In many online-messaging systems, messages are encrypted end-to-end, but the participants cannot later *plausibly deny* having said something, and old messages can often be verified long after the conversation. *Off-the-Record (OTR) Messaging* goes a step further: it guarantees that the communication satisfies confidentiality and authenticity, while also achieving **deniability**, so that transcripts cannot be cryptographically proven to third parties. This can helps e.g., support freedom of expression against oppressive regimens.

This project invites students to learn about this topic: start from the original OTR protocol, study how it achieves its security guarantees, and build a simple prototype secure-messaging system. Students will learn how OTR combines **Diffie–Hellman key exchange**, **message authentication**, etc. to provide privacy and deniability.

Depending on ambition, the project can be extended to study or implement more recent systems derived from OTR, such as **Signal's Double Ratchet.**

Warning: this is a cryptography-heavy project that requires reading and reasoning about protocol specifications. Students should be comfortable with basic cryptographic concepts and have some programming experience.

Suggested reading:

https://otr.cypherpunks.ca/Protocol-v3-4.1.1.html

https://signal.org/docs/specifications/doubleratchet/