Subjects for Bachelor Projects in the Cryptography and Security Group.

The following possibilities 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.

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 one-time signatures based on hash functions.
2. Read and understand how this can be extended to signature schemes that can be used for multiple messages.
3. Implement and benchmark against the signature schemes you have developed during dDistSik.

Blockchain technology and Consensus algorithms

Most of you know something about this subject from the dDistSec course. In this family of project ideas, you have several possibilities:
1) Implement a fully functional consensus layer that will work in a distributed setting, with finalization and all the bells and whistles.

2) Like 1), but focus on sharding and its effect on smart contracts.
3) Study the rich body literature on consensus algorithms – and compare and/or implement solutions.

Byzantine broadcast protocols

In the dDistSec course, you saw that *n* servers can reliably create a broadcast channel, even if up to *n-1* of them are faulty, given a public-key infrastructure as setup. This protocol, based on digital signatures, is known as the Dolev-Strong protocol.
In this project, you will implement and study the efficiency of the Dolev-Strong protocol and more recent broadcast protocols which claim better efficiency for long messages.

Here is an indicative list of activities:
1. Implement reliable broadcast using Dolev-Strong.
2. Read, understand and analyze one or more alternative methods with lower complexity [1, 2].
3. Implement one or more of the alternative methods.
4. Benchmark your implementations in a variety of configurations and analyze the results.

References:
[1] Multi-Valued Byzantine Broadcast: the $t < n$ Case – *Martin Hirt and Pavel Raykov (Asiacrypt, 2014)*
[2] Optimal Extension Protocols for Byzantine Broadcast and Agreement – *Chaya Ganesh and Arpita Patra (PODC, 2018)*

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.

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.
5.

Attacks and Counter measures 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.

Security of COVID contact-tracing apps

Digital contact tracing using mobile applications has been adopted by many governments in the strategy to find the spread of COVID-19. However, but there are still unanswered questions about their security and privacy, both from the point of view of the cryptographic security of the underlying protocol and implementation security of the concrete applications.

Different projects within this topic can be defined, depending on whether you are more interested in a more theoretical or more applied project. A project in this area will include one or more of the following elements:
- Studying the main protocols for decentralized digital contact tracing, such as DP-3T, and their security/privacy claims
- Reverse engineering of a subset of relevant contact-tracing apps to understand their implementation and integration with national healthcare systems
- Security analysis of their privacy features, both from theoretical and practical perspectives

References for further reading:
- The DP-3T protocol: https://github.com/DP-3T/documents
- Analysis of DP-3T: https://eprint.iacr.org/2020/399.pdf
- Security analysis of the Exposure Notification API used in the Danish Smitte|stop app: https://eprint.iacr.org/2020/428.pdf