Brief Overview

- Main focuses:
  - Learning cryptography through practice
  - Securely implementing cryptographic algorithms
  - Applying cryptographic algorithms to real-world problems

- This module assumes you have some basic background about cryptography
  - E.g., Formal Definitions of (a)symmetric encryption, digital signature,…
  - Provable security are **not** included in this module

Time

- Lecture dates:
  - October: 21, 29
  - November: 05, ~~12~~, 19, 26
  - December: 03, ~~10, 17~~
  - January 2027: 14, 21, 28
  - February 2027: 04, 11

- No lectures on November 12, December 10, or December 17
  - The first two hours on November 5 will be devoted to lectures
  - All three hours on December 3 will be devoted to lectures.

Contact Information

- Lecturer & TA: Runzhi Zeng
- Email:
  - runzhi.zeng@uni-kassel.de
- Office hours
  - Office: Room 2628, FB16
  - 3:30 pm – 4:00 pm, **Thursday**
  - (Please send an email in advance)
- Slides will be updated after lectures

Format

- Lecturing (about an hour, 12:00 – 13:00, but we remain flexible)
  - All lectures and slides are given in English

- Coding (about two hours, 13:00 – 15:00, including discussion, Q&A, etc.)

- **Coding-oriented ----- Please bring your laptop!**

- **Please also bring your pen and paper**

Programming Language

- **Rust**
  - Efficient and memory-safe
  - Rich ecosystem and active community



- **We require all implementations to be written in Rust.**

Assessment and Grading

- The final grade is based on two components:
  - **Written exam**
  - **Final project with a report**

- Written exam:
  - Students must score at least 60 points on the written exam to pass the course, **regardless of their project performance.**

- Final project:
  - For students who pass the written exam, the final grade is calculated as the weighted average of the written exam score and the final project/report score.

Written Exam

- Goal: Assess your **understanding of the cryptographic building blocks** covered in this course and your ability to apply them correctly in practice.

- In particular, you should understand:
  - What each cryptographic primitive is designed to achieve.
  - When and where a particular primitive should be used.
  - When using a particular primitive would be inappropriate or insecure.

- When: TBD

Sample Written Exam

- **Sample Written Exam**: A sample written exam will be released around **early December**.
  - You are encouraged to try it to assess your understanding of the course material.
  - You can complete the sample exam and email me your answers for evaluation.
  - The sample exam will **not** count towards your final grade.

Final Project

- One or two options
- What to submit: **Code and a simple report**
- The simple report should contain:
  - Choose 3-6 functions that you think are the best in your program and present them in your report, including *What it does*, *How it works*, and *Why it works correctly*
  - **2-4 pages**

- Submission deadline for the final project: TBD (after the written exam)

- How to submit:
  - Send me an E-Mail (with all files of your code or your github link, etc) before the deadline

Short Summary about Final Grade

$$\text{Final Score} = \begin{cases} Fail. & \text{if } W < 60 \\ \frac{1}{2}(W + P). & \text{if } W \geq 60 \end{cases}$$

W: Written exam score P: Final project and report score

- **The final score (out of 100) will be converted to the German grading scale (1.0–5.0).**

AI Usage

- **AI as a Tool, Not a Replacement**
  - Using AI is encouraged
  - Let AI assist you, not replace you

- **Learn to solve real-world cryptographic problems with AI**
  - **Model** real-world security problems
  - **Identify** which cryptographic algorithms are needed and where they should be applied
  - **Guide** AI to implement your designs correctly and securely