Homework 1 for 02231, 2025 (10 points)

Due 29.09.2025, **08:00**

Notation. We denote vectors as $\vec{x} \in \{0,1\}^\lambda$. By $\vec{x}[i]$ we denote the ith index of $\vec{x}$, where $i \in \{1,\dots,\lambda\}$. As in the lecture, we write $k \leftarrow K$ if k is sampled uniformly from the finite set K, i.e. such that each element from K is chosen with equal probability $1/|K|$. For two strings x, y we use $x|y$ to denote the string obtained from concatenating x with y. For a prime p and $a \in \{1,\dots,p-1\}$ we write a^{-1} to denote the modular inverse of $a \bmod p$. For example, if $p = 7$ and $a = 3$ then $a^{-1} = 5$, because $a \cdot a^{-1} = 3 \cdot 5 = 15 = 1 \bmod 7$.

Exercise 1.1. (3+1 points)
In this exercise, we will consider the One-time Pad for messages of length λ with point-wise multiplication modulo a prime $p > 2$ instead of addition. For two vectors $\vec{a}, \vec{b}$ we will write $\vec{a} \odot \vec{b}$ to denote the vector which is $\vec{a}[i] \cdot \vec{b}[i]$ at index i. We write $\vec{a}^{-1}$ to denote a vector that is $\vec{a}[i]^{-1}$ at index i.

We define the following algorithms:

KEYGEN() generates a key $\vec{k} \leftarrow \{1,\dots,p-1\}^\lambda$.

ENC$(\vec{k}, \vec{m})$ on input a key $\vec{k} \in \{1,\dots,p-1\}^\lambda$ and message $\vec{m} \in \{0,\dots,p-1\}^\lambda$ outputs $\vec{c} = \vec{m} \odot \vec{k}$.

DEC$(\vec{k}, \vec{c})$ on input a key $\vec{k} \in \{1,\dots,p-1\}^\lambda$ and ciphertext $\vec{c} \in \{0,\dots,p-1\}^\lambda$ outputs $\vec{m} = \vec{c} \odot \vec{k}^{-1}$.

Clearly, we have to exclude keys that contain 0 as the encryption scheme will otherwise not be correct. We therefore have the key space $\mathcal{K} = \{1,\dots,p-1\}^\lambda$ and message and ciphertext space $\mathcal{C} = \mathcal{M} = \{0,\dots,p-1\}^\lambda$.

1. Show that the resulting symmetric-key encryption scheme (KEYGEN, ENC, DEC) is not Real-or-Random secure. For this, write down an attack that makes one query to $CTXT$ with a specially chosen message. For your attack, you can choose λ, p as you wish. Argue what the success probability of your attack is.

2. Consider that we change the encryption scheme such that $\mathcal{M} = \{1,\dots,p-1\}^\lambda$ as well, i.e. that $\vec{m} \in \{1,\dots,p-1\}^\lambda$. What is the ciphertext space $\mathcal{C}$ for this SKE for $p = 5$? Show that the SKE is now Real-or-Random secure.

Bonus Can you show that your security argument holds for all $p > 2$?

Exercise 1.2. (3 points)
For this exercise, we denote with & the AND-function, which is defined by the following truth table:

x	y	$x\&y$
0	0	0
0	1	0
1	0	0
1	1	1

For two vectors $\vec{x}, \vec{y}$ we can define the coordinate-wise AND of both vectors, $\vec{x}\&\vec{y}$, by saying that $(\vec{x}\&\vec{y})[i] = \vec{x}[i]\&\vec{y}[i]$. This means that $\vec{x}\&\vec{y}$ is computed by applying & to the individual coordinates of each $\vec{x}, \vec{y}$ and concatenating the outcomes.

We now attempt to give a different version of the Feistel construction, which uses & instead of $\oplus$. Our construction is as follows: Let $F : \{0,1\}^\lambda \times \{0,1\}^\lambda \to \{0,1\}^\lambda$ be a PRF. For an input vector $\vec{x} \in \{0,1\}^{2\lambda}$ where $\vec{x} = (\underbrace{\vec{x}_0}_{\lambda \text{ bits}} \| \underbrace{\vec{x}_1}_{\lambda \text{ bits}})$ we define one round of the modified Feistel construction as

$$MF(\vec{k}, \vec{x}) = (\vec{x}_1 \| F(\vec{k}, \vec{x}_1) \& \vec{x}_0)$$

1. Show that applying this version of Feistel for 3 or more rounds, i.e. computing

   $$\vec{y} = P(\vec{k}_1 \| \vec{k}_2 \| \vec{k}_3, \vec{x}) = MF(\vec{k}_3, MF(\vec{k}_2, MF(\vec{k}_1, \vec{x})))$$

   does not yield a pseudorandom permutation. For this, construct an attacker that can distinguish $L^P_{PRP-real}$ from $L^P_{PRP-rand}$ with overwhelming probability in λ. Your attacker only has to do one query to the library.

Exercise 1.3. (4 points)
In this exercise, we ask you to build an implementation of the mode-of-operation CTR based on AES-128 and attack it. You should implement a library that offers three functions:

1. *Key Generation*, which should output a random key of length 128 bits.
2. *Encryption*, which should encrypt a message of ℓ blocks of length 128 bits each using a key k using AES-128 in CTR mode.

3. *Decryption*, which should decrypt a ciphertext of $\ell+1$ blocks of length 128 each using a key k and AES-128 in CTR mode.

Your code should generate the nonce, to be used in CTR, itself (i.e. it should not be a direct input to the encryption function), and it should choose the nonce uniformly at random from the appropriate space. You may use any library that implements encryption and decryption of blocks using AES-128, such as the *aes* crate in Rust. You can generate the randomness for the nonce using standard libraries in the programming language, or appropriate cryptographic libraries.

The exercise consists of the following two tasks:

1. Implement the 3 functions mentioned above, allowing you to encrypt and decrypt messages of length $128 \cdot \ell$ using AES-CTR. Write test code that checks that your implementation is correct. Describe in the handin what general approach your code follows.

2. CTR-mode is not IND-CCA secure, i.e. it is possible to modify ciphertexts so that the plaintexts get modified in a predictable way. Write a test that performs this attack, and shows that it works on your AES-CTR implementation. In addition, also describe the attack on a high level in your handin.

You should upload a .zip-file containing the code which you wrote for this exercise (in addition to the handin), together with a README file. Your code must compile on a machine running Ubuntu 24.04.2 LTS. Please include the following two sections in the README:

Compilation and Installation: What are the prerequisites (installed libraries and versions etc.) to compile and run your code, which commands should be used to compile the code, how is the code installed?

Running the tests: Describe which commands should be used to run tests that show that your functions work. How should the outputs of the tests be interpreted by a reader?

Please test if the instructions in the README work in the described environment before handing in. Non-functioning code will impact how many points you obtain.

If you found a library that directly implements AES-CTR and your code only calls that library, then this is not a viable solution. The point of this exercise is to implement CTR mode, using pure AES-128, yourself.

What you should do

- Enroll into one of the homework submission groups. You are encouraged to work in groups of (up to) 3, so the groups have capacity 3.
- Write the solutions to the exercises in one document.
- Upload your document on Learn, together with the .zip file containing the code.
- You may work in groups of at most three students.
- The format of your document should be PDF.
- Please do not forget to describe any coding-based exercises to the depth required by the exercise.