Lecture Notes for Cryptographic Computing
1. Introduction

Lecturers: Claudio Orlandi, Peter Scholl, Aarhus University

October 23, 2023

1 Introduction to Cryptographic Computing

The first lecture will be used to give a high-level overview of the course. There are no notes for this lecture, but you should read (preferably before our first meeting) one or both of the following introductory texts to the course content: [HL10, Chapter 1, Introduction], [Orl11].

2 Mandatory Assignments

Exercise 1. (Mandatory Assignment)

During the course we will be using the following example of a secure two-party computation task: Alice wants to learn if she can receive blood from Bob but both Alice and Bob care about the privacy of their blood type. Therefore Alice should learn only the result (a single bit), while Bob should learn nothing. In this assignment you have to:

- Find information about compatibility between the 8 different bloodtypes

$$\{0-, 0+, A-, A+, B-, B+, AB-, AB+\}$$

  (for instance on Wikipedia);

- Write two functions that take as input two blood types x and y and output whether x can receive from y:

  1. The first function should compute the output by performing a simple lookup in a "truth table";
  2. The second function should compute the output using a Boolean formula (i.e., a formula that computes the result using the Boolean operators AND, OR, XOR, NOT, etc.); (Hint: since there are 8 blood types you need 3 bits to describe each input. Choosing the "right" encoding of a blood type into the three bits helps in finding nicer formulas!)

3 Things you should already know

The following exercises ask you to recall some of the topics from earlier courses which will be relevant for Cryptographic Computing. If you have not followed *Cryptography* and *Cryptologic Protocol Theory* at Aarhus University, talk to the instructors.

Exercise 2. (One-Time Pad)

Recall how the cryptosystem One-Time Pad works. By now you should have heard that OTP is perfectly secure meaning that:

$$H(M|C) = H(M)$$

Here is an attempt to define the security of any cryptosystem as a game between an adversary A and a challenger B.

1. The adversary chooses a message $m \in \{0,1\}^n$ and sends it to B;
2. B flips a bit b: if $b = 0$ we say that we are in the *Ideal World* and B outputs a random value c from $\{0,1\}^n$; if $b = 1$ we say that we are in the *Real World* and B samples a random value k in $\{0,1\}^n$, and then outputs $c = k \oplus m$.
3. A outputs a guess g at b.

The (in)security of the scheme is measured by the ability of the adversary to guess correctly i.e., the ability of distinguishing between the *real* and *ideal world*.
Recall the notion of statistical distance of distributions from CPT. Let U be the distribution of c when $b = 0$ and V the distribution of c when $b = 1$. Prove that U, V are perfectly indistinguishable (and therefore the adversary A cannot guess correctly with probability greater than $1/2$).

Exercise 3. (Integer One-Time Pad)

Define two family of distributions, parametrized by an integer m in the interval $[0, 2^n)$.

1. IDEAL$_m$: output a random number c in the interval $[0, 2^{n+s})$ where s is a security parameter;
2. REAL$_m$; sample a random number k in the interval $[0, 2^{n+s})$ and then output $c = m + k$ ($+$ is integer addition).

Recall the notion of statistical distance of distributions from CPT. What is the statistical difference between the two distributions (as a function of m)? What can you conclude about the security of the scheme?

Exercise 4. (Impossibility of Two-Party Secure Computation)

In CPT you learned that secure two-party computation is impossible. Recall the main idea of the impossibility proof which shows that there exist no secure protocol for securely computing the AND of two bits in the presence of passively corrupted adversaries.

Exercise 5. (The RSA Cryptosystem)

Recall how the "vanilla" RSA encryption works (that is, the RSA trapdoor permutation which is *not* a secure encryption scheme). If you multiply two ciphertexts encrypted under the same public key together and then decrypt, what happens? (*Optional:* verify your hypothesis by implementing the cryptosystem. Using the class `BigInteger` in Java (or similar libraries for large number arithmetic in Python or Go) is probably the easiest way to get this done).

Exercise 6. (The ElGamal Cryptosystem)

Recall how ElGamal cryptosystem works, when implemented using the multiplicative subgroup of order q of $\mathbb{Z}_p^*$ with $p = 2q + 1$ and p, q are prime numbers. What happens if you perform a pointwise multiplication of two ciphertexts? – i.e.,

$$(a, b) \cdot (c, d) = (a \cdot c, b \cdot d)$$

Define a modified version of ElGamal, called Exp-ElGamal where:

- The plaintext m can only be chosen in the interval $\{0, 1\}$;
- Let g be a generator of the multiplicative subgroup of order q in $\mathbb{Z}_p^*$. To encrypt with Exp-ElGamal first compute $M = g^m$ and then encrypt using standard ElGamal;
- To decrypt 1) decrypt using the standard ElGamal to recover M and 2) Find the value of m from M;

What happens if you perform a pointwise multiplication of two ExpElGamal ciphertexts and then decrypt the result? (*Optional:* verify your hypothesis by implementing the cryptosystem.)

Exercise 7.

(A reduction – computational problem) Recall the discrete logarithm problem (DL). Consider a game where an adversary is given two uniformly random generators g, h of a group of prime order q and outputs a, b, c, d in $\mathbb{Z}_q$, with $(a, b) \neq (c, d)$. The adversary wins if $g^a h^b = g^c h^d$. Prove that if the DL problem is hard, the adversary cannot win the game.

Exercise 8.

(A reduction – decisional problem) Recall the decisional Diffie-Hellman assumption (DDH). Prove that if the DDH assumption holds then ElGamal satisfies IND-CPA security (indistinguishability against against chosen-plaintext attacks).

Exercise 9.

(Defining knowledge) Find in CPT notes the definition of "zero-knowledge". How do we formally define that the verifier "does not learn anything"? Is this a good definition? Discuss.

Exercise 10.

(Defining knowledge) Find in the CPT notes the definition of a "proof of knowledge". How do we formally define that the prover "knows the witness"? Is this a good definition? Discuss.

Exercise 11.

(Concrete parameters and efficiency) Find online the current recommendations for key sizes for RSA, elliptic curves and symmetric crypto if you want your data to be secure for 20 years. A good place to start is `http://www.keylength.com/`. Run `openssl speed` on your machine to evaluate the performances of those algorithms with those parameters (approximately). How does the speed of public-key cryptography (RSA, elliptic curve, etc.) compare with the speed of symmetric-key cryptography (AES, SHA, etc.)?

References

[HL10] Carmit Hazay, Yehuda Lindell
Efficient Secure Two-Party Protocols
`http://lib.myilibrary.com.ez.statsbiblioteket.dk:2048/Open.aspx?id=300348`

[Orl11] Claudio Orlandi
http://cs.au.dk/ orlandi/icassp-draft.pdf
ICASSP 2011.
Available at `http://cs.au.dk/~orlandi/icassp-draft.pdf`