Codes and Lattices in Cryptography Mini-Course **FCT-UNL, Spring 2023**

Notes 1: Basics of provable security

Lecturer: João Ribeiro

Introduction

The goal of this lecture is to introduce some basic cryptographic concepts from a provable security perspective. Due to time constraints we will focus mostly on encryption primitives, and certain less relevant details will be omitted. However, it must be said that cryptography contemplates many other tasks beyond the ones we will see here.

1.1 Perfect encryption schemes

Consider a scenario where Alice wishes to transmit a secret message to Bob through a public (authenticated) channel. This should be done in a way such that an eavesdropper Eve observing the public channel between Alice and Bob will not gain any information about the underlying message. To this end, we assume that Alice and Bob share a secret key sk sampled appropriately from a keyspace $\mathcal{K}$.

This is the setting of *symmetric-key encryption*. In full generality, given a message m from a message space $\mathcal{M}$, Alice will use a (known, and possibly randomized) encryption function $\mathsf{Enc} : \mathcal{M} \times \mathcal{K} \to \mathcal{C}$, where $\mathcal{C}$ is the ciphertext space, to encrypt her message as

$$C = \mathsf{Enc}(m, sk).$$

In turn, Bob will use a decryption function $\mathsf{Dec} : \mathcal{C} \times \mathcal{K} \to \mathcal{M}$ to decrypt C and recover the message m. Of course, we require the basic correctness property that for all messages $m \in \mathcal{M}$ and keys $sk \in \mathcal{K}$ it holds that

$$\mathsf{Dec}(\mathsf{Enc}(m, sk), sk) = m.$$

The eavesdropper Eve learns the ciphertext C.

Let us attempt to formalize what it means for an encryption scheme $(\mathsf{Enc}, \mathsf{Dec})$ to be "secure".

Definition 1.1 (Perfectly secure encryption scheme [Sha49]) *We say that* $(\mathsf{Enc}, \mathsf{Dec})$ *is a* perfect encryption scheme *if the random variables* $\mathsf{Enc}(m, SK)$ *and* $\mathsf{Enc}(m', SK)$ *are identically distributed for any two messages* $m, m' \in \mathcal{M}$ *when the key* SK *is sampled uniformly at random from the keyspace* $\mathcal{K}$.

In a perfect encryption scheme, eavesdroppers learn no information about the message, no matter their computational power.

A well-known (and quite useful!) example of a perfect encryption scheme is the *one-time pad*, proposed by Vernam more than 100 years ago. The one-time pad encryption scheme with message length n is defined by setting $\mathcal{M} = \mathcal{C} = \mathcal{K} = \{0,1\}^n$ and $\mathsf{Enc} : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ as

$$\mathsf{Enc}(m, sk) = m + sk \pmod 2,$$

where the "+" operation is applied bitwise. The associated decryption function Dec is given by $\mathsf{Dec}(c, sk) = c + sk \pmod 2$. The following theorem is a consequence of the fact that $m + SK \pmod 2$ is uniformly distributed over $\{0,1\}^n$ for any fixed message $m \in \{0,1\}^n$ when SK is sampled uniformly at random from $\{0,1\}^n$.

Theorem 1.1 *The one-time pad encryption scheme is perfect.*

1.1.1 Limitations of perfect security

Perfect encryption schemes offer the best possible security. However, they also have some significant drawbacks. For example, the one-time pad scheme we saw above requires the keylength to be equal to the message length. Due to practical constraints, we would like to design encryption schemes which use keys that may be much shorter than the messages being encrypted. A result of Shannon [Sha49] shows that this is impossible – a perfect encryption scheme requires the keys to be at least as long as messages.

Theorem 1.2 *If* $(\mathsf{Enc}, \mathsf{Dec})$ *is a perfect encryption scheme with message space* $\mathcal{M}$ *and keyspace* $\mathcal{K}$*, then it must hold that*

$$|\mathcal{K}| \geq |\mathcal{M}|.$$

Proof: This follows by a simple double counting argument. Note that there are at most $|\mathcal{K}|$ possible ciphertexts for any given message m. Since the scheme is perfect, we claim that there are at most $|\mathcal{K}|$ possible ciphertexts in total across all messages. To see this, suppose not. Then, there exists a message m' and a key sk' such that

$$c' = \mathsf{Enc}(m', sk') \notin \{\mathsf{Enc}(m, sk) : sk \in \mathcal{K}\}.$$

This means that $\mathsf{Enc}(m, SK)$ and $\mathsf{Enc}(m', SK)$ have different supports, violating perfect security.

To conclude the proof, observe that there must also be at least $|\mathcal{M}|$ possible ciphertexts, since the correctness property of the encryption scheme implies that $\mathsf{Enc}(\cdot, sk)$ must be injective for every secret key sk. ■

1.2 Secure encryption with shorter keys?

It is natural to wonder whether we can design "secure" encryption schemes which use keys much shorter than the message. Theorem 1.2 tells us that this is impossible if we are aiming for perfect security, but we may be content with a more relaxed, but still meaningful, notion of security.

In particular, what if we only require security against eavesdroppers with limited computational power? For example, such an adversary should not be able to traverse the whole (huge) keyspace, and so there is hope that we can design protocols with properties that are simply unachievable against computationally-unbounded adversaries.

1.2.1 Computationally-bounded adversaries

Inspired by complexity theory, we consider security against *Probabilistic Polynomial Time* (PPT) adversaries. We say that an algorithm A with input space $\{0,1\}^*$ is PPT if (1) it has access to an unlimited stream of independent uniformly random bits (also called the *coins* of the algorithm), and (2) there exists a polynomial $p(\cdot)$ such that for every input $x \in \{0,1\}^*$ it holds that A terminates within $p(|x|)$ steps, where $|x|$ denotes the length of x.

To be completely formal, we should specify a model of computation for our algorithms. For the knowledgeable reader, one option is to model our algorithms via Turing machines with a read-write tape and another read-only tape filled with independent uniformly random bits. Sometimes a stronger non-uniform computational model is also considered where an algorithm is seen as a family of polynomial-size circuits, or, equivalently, as a Turing machine as above with additional access to a (possibly hard-to-compute) polynomial-length advice string for each input length.

However, for these lectures it is okay if we pick our favorite programming language and think of an algorithm as a program in that language with access to a functionality that generates independent uniformly random bits.

For more on this, see the books by Goldreich [Gol01] or classic books on computational complexity such as that of Arora and Barak [AB09].

Some helpful notation. Throughout these lectures we will be sampling many things, usually either uniformly from a given set or as a result of running some probabilistic algorithm. Therefore, we may write $x \leftarrow \mathcal{X}$ to denote that x was sampled uniformly at random from the set $\mathcal{X}$. Likewise, we may write $x \leftarrow \mathsf{A}(z)$ to denote that x is sampled by running the probabilistic algorithm A on input z.

1.2.2 Encryption secure against computationally-bounded adversaries

Recall from Definition 1.1 that an encryption scheme is perfectly secure if the ciphertext distributions associated to distinct messages m_0 and m_1 are identically distributed. In other words, an adversary cannot distinguish between $\mathsf{Enc}(m_0, SK)$ and $\mathsf{Enc}(m_1, SK)$. Our notion of security against PPT adversaries will follow this intuition – no PPT adversary should be able to distinguish between encryptions of m_0 and m_1. Before we proceed to the formal (asymptotic security) definition, we need to define *negligible functions*.

Definition 1.2 (Negligible function) *A function $f : \mathbb{N} \to \mathbb{R}_{\geq 0}$ is said to be* negligible *if for*

every constant $c > 0$ there exists $n_c \in \mathbb{N}$ such that

$$f(n) \leq n^{-c}$$

for all $n \geq n_c$.

For example, functions such as 2^{-n} and $n^{-\log\log n}$ are negligible. We are now ready to define computational security for encryption schemes.

Definition 1.3 (Computationally-secure encryption scheme) *Let* $(\mathsf{Enc}, \mathsf{Dec})$ *be an encryption scheme. Consider the following game (parameterized by $n \in \mathbb{N}$) between an adversarial algorithm* Adv *and a challenger* Ch*:*

1. *The adversary chooses messages $(m_0, m_1) \leftarrow \mathsf{Adv}(1^n)$ and sends (m_0, m_1) to the challenger* Ch*;*
2. Ch *samples a key $SK \leftarrow \mathcal{K}$ and a challenge bit $b \leftarrow \{0,1\}$. Then,* Ch *computes $c = \mathsf{Enc}(m_b, sk)$ and sends c to the adversary* Adv*.*
3. Adv *computes a guess $b'_{\mathsf{Adv}} \leftarrow \mathsf{Adv}(c, 1^n, m_0, m_1)$. The adversary wins the game if $b' = b$.*

We say that the encryption scheme $(\mathsf{Enc}, \mathsf{Dec})$ *is* computationally-secure *if for any PPT adversary* Adv *in the above game there exists a negligible function* negl *such that*

$$\Pr[b'_{\mathsf{Adv}} = b] \leq \frac{1}{2} + \mathsf{negl}(n)$$

for all $n \in \mathbb{N}$, where the probability is taken over the coins of Adv *and* Ch*.*

Observe that every perfect encryption scheme is also computationally-secure. But can we construct computationally-secure encryption schemes which only need much shorter keys? By Theorem 1.2, such schemes would not be perfect. A natural idea is to start with a short uniformly random seed s and expand it into a longer (but not uniformly random) key $sk = G(s)$. If $G(s)$ looks sufficiently close to uniformly random *to any PPT adversary*, then we could hope to use it this "pseudorandom" key in the one-time pad encryption scheme.

1.2.3 Pseudorandom generators and encryption with short keys

We proceed to formalize our intuition above. First, we define what it means for two distributions to be close to each other from the perspective of PPT adversaries.

Definition 1.4 (Computational indistinguishability) *We say that two ensembles of random variables $(X_n)_{n\in\mathbb{N}}$ and $(Y_n)_{n\in\mathbb{N}}$ are* computationally indistinguishable *if for every PPT algorithm D there exists a negligible function* negl *such that*

$$|\Pr[D(1^n, X_n) = 1] - \Pr[D(1^n, Y_n) = 1]| \leq \mathsf{negl}(n)$$

for every n, where the probability is taken over the coins of D and X_n and Y_n.

We now define pseudorandom generators: Objects that transform short random seeds into longer pseudorandom strings.

Definition 1.5 (Pseudorandom generator) *A function $G : \{0,1\}^n \to \{0,1\}^{\ell(n)}$ is said to be a* pseudorandom generator *(PRG) if the ensembles $(G(U_n))_{n\in\mathbb{N}}$ and $(U_{\ell(n)})_{n\in\mathbb{N}}$ are computationally indistinguishable, where U_m denotes a random variable uniformly distributed over $\{0,1\}^m$.*

Note that PRGs with $\ell(n) \leq n$ are trivial to construct and uninteresting. Therefore, we focus solely on the expanding case where $\ell(n) > n$. The special case where $\ell(n) = n + 1$ is already interesting – the existence of such an efficient PRG implies the existence of efficient PRGs for larger output length $\ell(n) = p(n)$ for any polynomial p.

We remark that the output of an expanding PRG is necessarily far from random, in the sense that a computationally-unbounded algorithm can easily distinguish between $G(U_n)$ and $U_{\ell(n)}$. It suffices to note that $G(U_n)$ is supported on a set of size at most 2^n, while $U_{\ell(n)}$ is supported on a set of size $2^{\ell(n)} \gg 2^n$.

Fix a PRG $G : \{0,1\}^n \to \{0,1\}^{\ell(n)}$. Following the idea above, we consider the encryption scheme $(\mathsf{Enc}_G, \mathsf{Dec}_G)$ for $\ell(\lambda)$-bit messages and λ-bit keys given by

$$\mathsf{Enc}_G(m,s) = G(s) + m \pmod 2 \quad \text{and} \quad \mathsf{Dec}_G(c,s) = G(s) + c \pmod 2. \tag{1.1}$$

The following result establishes the computational security of this scheme, conditioned on G being a PRG. This is proved via a *reduction* – An adversary that breaks the security of the encryption scheme is transformed into an adversary that breaks the guarantees of the underlying PRG. This proof strategy is common throughout cryptography.

Theorem 1.3 *The encryption scheme defined in Equation (1.1) is computationally-secure provided that G is a PRG.*

Proof: The desired result follows if we show that a PPT adversary that breaks the security of the encryption scheme $(\mathsf{Enc}_G, \mathsf{Dec}_G)$ can be efficiently transformed into a PPT distinguisher that breaks the security of the PRG G.

Suppose that there exists a PPT adversary Adv for the encryption security game in Definition 1.3 such that $\Pr[b'_{\mathsf{Adv}} = b] \geq \frac{1}{2} + \frac{1}{p(n)}$ for some polynomial p and all sufficiently large n. Consider the following PPT algorithm D which on input $(1^n, z)$ runs Adv as a subroutine:

1. Run Adv on input 1^n to obtain two messages m_0 and m_1;
2. Sample $b \leftarrow \{0,1\}$. Send $m_b + z \pmod 2$ to Adv;
3. Suppose that Adv outputs b' given this information. Then, D outputs 1 if $b' = b$ and 0 otherwise.

We claim that

$$|\Pr[D(1^{\ell(n)}, G(U_n)) = 1] - \Pr[D(1^{\ell(n)}, U_{\ell(n)}) = 1]| \geq \frac{1}{p(n)}$$

for all sufficiently large n, thus breaking the security of the PRG G. To see this, first note that if $z \leftarrow \{0,1\}^{\ell(n)}$, then $m_b + z \pmod 2$ corresponds exactly to one-time pad encryption, and so $m_b + z \pmod 2$ is independent of b. Therefore, we have that

$$\Pr[D(1^{\ell(n)}, U_{\ell(n)}) = 1] = \frac{1}{2}. \tag{1.2}$$

On the other hand, if $z = G(U_n)$, then $m_b + z \pmod 2 = \mathsf{Enc}_G(m_b, U_n)$, and so Adv is playing the distinguishing game for the encryption scheme $(\mathsf{Enc}_G, \mathsf{Dec}_G)$. By hypothesis, we then have that

$$\Pr[D(1^{\ell(n)}, G(U_n)) = 1] = \Pr[b'_{\mathsf{Adv}} = b] \geq \frac{1}{2} + \frac{1}{p(n)} \tag{1.3}$$

for all sufficiently large n. Combining Equations (1.2) and (1.3) concludes the argument. ■

Stronger notions of security for symmetric-key encryption. The encryption schemes discussed here are *one-time* in the sense that the key must be refreshed every time we wish to encrypt a new message. In fact, if two messages m_0 and m_1 are encrypted using the one-time pad with the same key sk as $c_i = m_i + sk \pmod 2$, then an eavesdropper can learn the non-trivial information $m_0 + m_1 \pmod 2 = c_0 + c_1 \pmod 2$. Stronger notions of security for encryption schemes where security is preserved even when the adversary is allowed to ask for several encryptions and decryptions of messages and ciphertexts of their choice (so-called IND-CPA and IND-CCA security, respectively) have been widely studied.

1.3 One-way functions

It turns out that there exists a particular cryptographic object whose existence is both necessary and sufficient to solve many cryptographic problems (a so-called *minimal assumption* in cryptography): *One-way functions*. Intuitively, a function is one-way if it is both easy-to-compute and hard-to-invert. We make this more precise in the definition below.

Definition 1.6 (One-way function) *A family of functions (f_n) with $f_n : \{0,1\}^n \to \{0,1\}^*$ is said to be* one-way *if:*

- ***Easy to compute:*** *There exists a polynomial-time deterministic algorithm which when given $x \in \{0,1\}^n$ as input outputs $f(x)$ for every input length $n \in \mathbb{N}$;*
- ***Hard to invert:*** *For every PPT adversary Adv there exists a negligible function negl such that for every n the adversary Adv wins the following function inversion game with probability at most $\mathsf{negl}(n)$:*
  1. *The challenger samples $x \leftarrow \{0,1\}^n$ and sends $f(x)$ to Adv;*
  2. *The adversary computes $x' \leftarrow \mathsf{Adv}(1^n, f(x))$ and wins if $f(x') = f(x)$.*

It is not known whether one-way functions exist, although we do have several candidate constructions of such objects based on the conjectured hardness of well-studied computational problems. We will see some examples in the following lectures.

The existence of one-way functions is known to be a sufficient (and necessary) condition for the existence of pseudorandom generators [HILL99] and IND-CCA-secure symmetric-key encryption, and also for the existence of other cryptographic objects we will not discuss in depth, such as commitment schemes, digital signatures, and zero-knowledge proofs. The existence of one-way functions also implies that $\mathsf{P} \neq \mathsf{NP}$.

1.4 Public-key encryption

To conclude, we discuss public-key encryption, a fundamental type of encryption whose existence is *strictly stronger* than the existence of one-way functions.[1] Intuitively, a public-key encryption scheme is a form of asymmetric encryption with a public key and a secret key. Anyone with knowledge of the public key can encrypt messages, but only the individual with access to the secret key should be able to decrypt them. As before, we present a definition based on a ciphertext distinguishing game.

Definition 1.7 (Public-key encryption scheme) *A* public-key encryption scheme *is composed of the following PPT algorithms:*

- *A keypair generation algorithm* Gen *such that* $(sk, pk) \leftarrow \mathsf{Gen}(1^n)$*, where* sk *is the secret key and* pk *is the public key;*
- *An encryption algorithm* Enc *which receives as input a message and the public key* pk*;*
- *A deterministic decryption algorithm* Dec *which receives as input a ciphertext and the secret key* sk*.*

We require the basic correctness property that

$$\Pr[\mathsf{Dec}(\mathsf{Enc}(m, pk), sk) = m] = 1$$

for every message m*, where the probability is taken over the coins of* Gen *and* Enc*. Moreover, we require the security property that for every PPT adversary* Adv *there exists a negligible function* negl *such that the probability of* Adv *winning the following ciphertext distinguishing game is at most* $\frac{1}{2} + \mathsf{negl}(n)$*:*

1. *The challenger* Ch *generates the keypair* $(pk, sk) \leftarrow \mathsf{Gen}(1^n)$ *and sends the public key* pk *to the adversary* Adv*;*
2. *The adversary chooses two messages* $(m_0, m_1) \leftarrow \mathsf{Adv}(1^n, pk)$ *and sends them to the challenger* Ch*;*

[1] The existence of public-key encryption would place us in the "Cryptomania" world, according to Impagliazzo's five-world model [Imp95]. The existence of one-way functions without the existence of public-key encryption corresponds to the "Minicrypt" world. Other options are also possible!

3. Ch *samples* $b \leftarrow \{0,1\}$ *and sends back* $c = \mathsf{Enc}(m_b, pk)$ *to* Adv*;*

4. *The adversary computes* $b' \leftarrow \mathsf{Adv}(1^n, c, pk, m_0, m_1)$ *and wins if* $b' = b$*.*

Remark 1.1 Candidate public-key encryption schemes where Enc is a deterministic function of the message and the public key are not secure with respect to Definition 1.7.

In the next lecture we will see how we can construct public-key encryption schemes from "post-quantum" hardness assumptions.

1.5 Notes and additional reading

Much more in-depth discussions of the material covered here can be found in various textbooks in cryptography, such as the books of Goldreich [Gol01], Katz and Lindell [KL20], and Rosulek [Ros21]. Another useful resource is Boneh's Coursera course on cryptography (`https://www.coursera.org/learn/crypto`).

References

[AB09] Sanjeev Arora and Boaz Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, 2009.

[Gol01] Oded Goldreich. *Foundations of Cryptography – Volume 1 (Basic Techniques)*. 2001.

[HILL99] Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 28(4):1364–1396, 1999.

[Imp95] Russell Impagliazzo. A personal view of average-case complexity. In *10th Annual IEEE Conference on Structure in Complexity Theory*, pages 134–147. IEEE, 1995.

[KL20] Jonathan Katz and Yehuda Lindell. *Introduction to Modern Cryptography*. CRC press, 2020.

[Ros21] Mike Rosulek. *The Joy of Cryptography*. 2021. `https://joyofcryptography.com`.

[Sha49] Claude E. Shannon. Communication theory of secrecy systems. *The Bell System Technical Journal*, 28(4):656–715, 1949.