Version of November 27, 2023

Proofs of Security

Tentative lecture notes

Léo Ackermann, Pierre-Alain Fouque, Olivier Sanders

Introduction

The evolution of cryptography. Cryptography, the art and science of secure communication, has an illustrious history dating back to ancient civilizations. Early cryptographic techniques, often referred to as historical ciphers, were used to encode messages and protect sensitive information from prying eyes. These methods included the Caesar cipher, the Vigenère cipher, and transposition ciphers. While these early systems provided a degree of security, they were far from invulnerable, as their encryption mechanisms were often based on simple and easily breakable algorithms.

In the modern era, cryptography has evolved into a complex and sophisticated field that plays a pivotal role in securing digital communication, financial transactions, and data protection. Let focus on encryption for a moment. Contemporary cryptographic methods can be broadly categorized into two paradigms: symmetric and asymmetric cryptography. Symmetric cryptography, akin historical ciphers, involves the use of a shared secret key for both encryption and decryption. This shared key must be kept confidential between the communicating parties. Notable symmetric encryption algorithms include the Data Encryption Standard (DES), the Advanced Encryption Standard (AES), and the Rivest Cipher (RC4). These algorithms are highly efficient and widely used for bulk data encryption, ensuring the privacy and integrity of information. On the other hand, asymmetric cryptography, also known as public-key cryptography, utilizes a pair of keys: a public key and a private key. The public key is openly distributed and can be used for encryption, while only the holder of the private key can decrypt the data. But cryptography is way more than that and is useful securing digital signatures, key exchange protocols, and other applications where trust and authenticity are paramount.

A little bit of math can accomplish what all the guns and barbed wire can't: a little bit of math can keep a secret.

— Edward Snowden

One of the most significant developments in modern cryptography is the pursuit of provable security. The Kirchhoff principle posists that the security of a cryptographic system should depend solely on the secrecy of the cryptographic key, not on the obscurity of the algorithms or methods used. This principle guides the development of robust and trustworthy encryption techniques, emphasizing the need for strong, well-protected keys as the cornerstone of secure communication. Consequently, cryptographers aim to design cryptographic schemes that are based on hard mathematical assumptions and have rigorously proven security properties. This involves the use of reductions and game-based proofs to demonstrate that the security of a cryptographic construction can be reduced to the hardness of a well-defined mathematical problem, eg. the discrete logarithm or factorization problem. This rigorous approach provides a high level of confidence in the security of cryptographic systems and has become a cornerstone of cryptographic research. ***Those approaches are the main focus of this unit***.

Limits of modern cryptography. While modern cryptography has made significant strides in enhancing the security of digital communication, it is essential to acknowledge the inherent limitations and challenges that persist in this ever-evolving field. These limitations can be grouped into three critical aspects.

Perhaps one of the most pressing concerns in modern cryptography is the impending era of quantum computing. Quantum computers have the potential to efficiently solve certain mathematical problems that underpin the security of widely used encryption schemes, such as integer factorization and discrete logarithm problems. This development threatens the security of many existing cryptographic systems, emphasizing the urgency of post-quantum cryptography. Researchers are actively exploring new encryption methods that can withstand the computational power of quantum computers, and this challenge remains a focal point for the field. A second concern lies within the fact that cryptography often grapples with a divide between provable and concrete security. While provable security relies on rigorous mathematical proofs and reductions to demonstrate the resistance of cryptographic schemes to attacks, it sometimes falls short in practical applicability. In contrast, concrete security is established through heuristics, cryptanalysis, and empirical evidence. Many widely-used cryptographic systems, such as those based on the RSA or ECC algorithms, rely on concrete security due to the lack of efficient, provably secure alternatives. Bridging this gap is an ongoing effort, aiming to ensure both theoretical rigor and real-world practicality in cryptographic systems. A last concern to be mentioned is that cryptography traditionally concentrates on isolated cryptographic primitives, such as encryption, digital signatures, and key exchange. However, as the complexity of digital systems and networks grows, cryptographic protocols involve multiple interacting primitives, leading to intricate security challenges. To address these complexities, the development of formal methods becomes essential. Formal methods allow for the systematic verification of entire cryptographic protocols, ensuring that they remain secure and robust in the face of ever-evolving threats. These formal methods offer a holistic approach to security, mitigating potential vulnerabilities in the design and implementation of cryptographic systems.

Defining security Defining security in a cryptographic context is a multi-faceted process, typically approached in a three-step journey that involves rigorous and precise considerations.

First and foremost, the cryptographic scheme itself must be precisely defined, with clarity regarding its interface and the properties it is intended to provide. This step involves establishing the rules and mechanics of the scheme, outlining how it operates, and specifying what it aims to achieve in terms of features. This typically includes a specification of the primitive behavior in an even world, often refered as correctness property. The scheme's structure should be clearly articulated to serve as the foundation for security analysis. The second crucial step revolves around defining what it means to "break the scheme". This is often done through the formulation of cryptographic games or experiments. These games serve as a means to quantitatively and qualitatively assess the security of the scheme. For example, in a security game, participants (i.e., the cryptographic scheme and an adversary) engage in a competition, with the adversary's objective being to exploit the scheme's vulnerabilities and compromise its security. The definition of "breaking the scheme" becomes the adversary's successful achievement of specified objectives, such as decrypting an encrypted message without the secret key. These well-defined games or experiments provide a structured framework for evaluating the scheme's robustness against potential threats. The third and final step involves explicitly defining the capabilities and constraints of the attacker. This encompasses setting boundaries on the attacker's computational resources, such as their running time, and access to oracles, which are external tools that might aid the attacker in their efforts. The precise delineation of attacker capabilities is essential for assessing the scheme's security under realistic threat scenarios. It also ensures that security definitions remain relevant and aligned with the evolving landscape of computational power and adversarial tactics.

1 Asymmetric cryptography

Notation. In this section, we highlight *secret* and *public* cryptographic materials, typically keys, by changing their color.

Public-key cryptography, a revolutionary concept in modern cryptography, enables secure communication, digital signatures and much more, without the need for a shared secret key. Its foundational idea involves the use of a key pair: a public key that can be openly distributed and a corresponding private key that remains exclusively known to the key's owner.

1.1 Confidential communication through public-key encryption

Public-key encryption stands as a cornerstone of securing communication in the digital age. It allows for the confidential exchange of information between parties by leveraging a pair of keys: a public key for encryption and a private key for decryption. This approach ensures that sensitive data can be securely transmitted over potentially insecure channels, with only the intended recipient possessing the means to decipher the message. Another possible approach to achieve such a goal, we do not focus on here, is as follows: public-key cryptography can be used for key-encapuslation mechanisms, enabling parties to securely negotiate and establish shared secret keys for subsequent symmetric cryptography.

1.1.1 Public-key encryption and related properties

Let's follow our motto to study the security of public-key encryption scheme by precisely defining the primitive and stating the security property we consider.

Definition 1 (Public-key encryption scheme)**.** *A public-key encryption scheme is a tuple of three probabilistic polynomial time (PPT) algorithms* $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *such that:*

- *The key generation algorithm* KeyGen *takes as input the security parameter* λ *written in unary*[*] *and returns a key-pair* $(pk, sk) \leftarrow \mathsf{KeyGen}(1^\lambda)$, *respectively called the public-key and the secret-key.*
- *The encryption algorithm* Enc *takes as input the public key* pk, *a message* m *and returns a ciphertext* $c = \mathsf{Enc}(pk, m)$.
- *The decryption algorithm* Dec *takes as input the secret key* sk, *a ciphertext* c *and returns a message* $m \leftarrow \mathsf{Dec}(sk, c)$.

The previous definition only describes the interface of a PKE scheme, hence does not capture any of its expected behavior of security properties Exo 1. Let discuss this further.

Correctness of PKE The *correctness* property reflects the intented behavior of a primitive without considering an adversary in town. In the case of encryption scheme, we expect that using evenly generated keys, the Dec algorithm invert the Enc algorithm. More precisely, such a scheme is said *correct* whenever for all possible message m it holds that:

$$\Pr\left[\begin{array}{l}(pk, sk) \leftarrow \mathsf{KeyGen}(1^\lambda) : \\ \mathsf{Dec}(sk, \mathsf{Enc}(pk, m)) = m\end{array}\right] = 1 - \mathrm{negl}(\lambda),$$

where $\mathrm{negl}(\lambda)$ is a function negligible in λ, that is a function that decrease faster that any inverse of polynomial in λ.[†] The later probability is written on an experiment: the part preceeding the

[*]The reason why is that one wants running time polynomial in the security parameter in cryptography, and that the standard notion of *efficiency* in complexity theory is polynomial time in the input size. By handing the security parameter as unary, the notions coincide.

[†]In particular, this implies that correctness, as many cryptographic notion, is an *asymptotical* notion.

colon describes the experiment and the part after is the event the probability focuses on. This could have been written $\Pr[\mathsf{Dec}(sk, \mathsf{Enc}(pk, m)) = m \mid (pk, sk) \leftarrow \mathsf{KeyGen}(1^\lambda)]$ using conditional probabilities.

Impossiblity of perfect secrecy. From the correctness property, we are ensured that the PKE scheme behaves weel in an even environment... which is clearly unsufficient for real world purposes. Let discuss informally what a *good* security property should capture. The first notion that comes to mind is the one of perfect secrecy: it should be unfeasible for an adversary to recover *any* information about m. This idea was first captured by Shannon in the context of information theory.

Definition 2 (Public-key perfect secrecy). *An public-key encryption scheme over message space $\mathcal{M}$ and ciphertext space $\mathcal{C}$ is* perfectly secret *if for any random variable M over $\mathcal{M}$, C the encryption of M under a public-key generated by* KeyGen*, any message $m \in \mathcal{M}$ and any ciphertext $c \in \mathcal{C}$, it holds:*

$$\Pr(M = m) = \Pr(M = m \mid C = c).$$

Unfortunately, this appealing security notion cannot be achieved in public-key cryptography. Intuitively, this is because the cryptographic keys are related to each other in some ways – so that the cryptographic algorithms work as expected – and one of them is public. This can be made precise by expliciting an unbounded adverary attacking the perfect secrecy Exo 2.

Lemma 3. *No public-key encryption scheme can be perfectly secure.*[*]

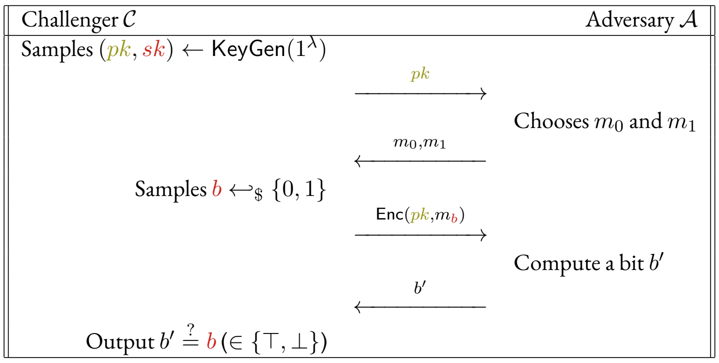
Nevertheless, the adversary we described is arguably inefficient. This encourage to finetune the notion of security by limiting the power an adversary, and deteriorate the concept.

Security of PKE. Let continue the later discussion with more reasonable notion of security. A first natural notion could be that an adversary is not capable of recovering the message underlying a specific ciphertext. This gives us the notion of *one-way security*, written OW. A second notion that comes to mind deals with indistinguishability: given a ciphertext that may be an encryption of x or y, an adversary cannot guess in which case we are. Typically, one can think of an encrypted referendum: if an adversary is able to distinguish an encryption of "YES" from an encryption of "NO", this is already a threat. In the case the adversary can choose both plaintexts, this gives us the notion of *indistinguishability security*, indicated with IND. In the case the adversary can only choose one plaintext and the other possible plaintext is sampled at random in the message space, this gives us the notion of *real-or-random security*, denoted RoR. Another interesting notion is that of *non-malleability*, written NM. Informally, it states that an adversary cannot "deform" a ciphertext meaningfully, that is producing a new ciphertext from an old one, for which it can predict a relation on the underlying messages. A corresponding attack scenario could be as follows: in a encrypted referendum where choices are encoded by bit, an adversary could potentially flip the vote of someone – while letting it encrypted. Many other flavours of security could be imagined, eg. where the attacker is only interested in parts of the plaintext, as its most significant bit.

A crucial point we did not discussed yet is the behavior and computational power of the adversary. It is commonplace to assume it polynomially bounded (ie. we restrict the security study to efficient adversaries), but this is not the only variable. A first scenario considers *passive* adversaries, that will follow the protocol but will try to extract as much information as possible. We denote this scenario by PASS. In the case of PKE, such an adversary knows the public key and can thus encrypt plaintexts of its choice. We talk about *chosen-plaintext attacks*, or CPA in short. In other scenarii, closer to some real world applications, the adversary can access a *decryption oracle*. Called on input c the latter returns the underlying plaintext or $\perp$ if the decryption fails, in constant time $O(1)$. In this case, we talk of *chosen-ciphertext attacks*, or CCA[†]. As before, many others variants could be considered.

[*]Note that this reasoning holds for any public key scheme, and not solely for encryption.

[†]In the case where the oracle is accessible only *before* the attacker commits on its challenge, CCA1 is used instead.



Challenger $\mathcal{C}$
Adversary $\mathcal{A}$
Samples $(pk, sk) \leftarrow \mathsf{KeyGen}(1^\lambda)$
pk
Chooses m_0 and m_1
m_0, m_1
Samples $b \hookleftarrow_\$ \{0,1\}$
$\mathsf{Enc}(pk, m_b)$
Compute a bit b'
b'
Output $b' \overset{?}{=} b \, (\in \{\top, \bot\})$

Figure 1: IND-CPA security game

Combining both aspects, let's now try to formalize one common security property of PKE: the *indistinguishability-under-chosen-plaintext-attacks* property, IND-CPA for short. Before writing down this property within an experiment formalism, see it as an interactive game between an adversary $\mathcal{A}$ and a challenger $\mathcal{C}$, as drawn in Figure 1. The game starts with the challenger generating keys and handling the public key to the adversary. Then, the adversary chooses two messages m_0 and m_1 of its choices and gives them to the challenger. The latter samples a bit $b \hookleftarrow_\$ \{0,1\}$ and hands an encryption of m_b under the public key to the adversary. Finally, $\mathcal{A}$ guess a bit b' and sends it back to $\mathcal{C}$. The adversary wins the experiment whenever $b = b'$. As the naive adversary, that hands b' as a tossed coin, wins this experiment with probability $1/2$, we focus on the *advantage* of an adversary rather than its success probabilty. Basically, it captures "how better" this adversary is compared to the very naive one.

Implicitly, we considered here an adaptative adversary: the latter propose its challenge (m_0, m_1) after learning the public key of the scheme and possibly performing some encryptions. There are easier variants of security properties, known as *selective*, where the adversary must commit to its challenge beforehand.

With all of this in mind, we can precisely define the IND-CPA within the experiment formalism.

Definition 4 (IND-CPA PKE). *A PKE is said IND-CPA secure if for any polytime adversary $\mathcal{A}$ its advantage $\mathrm{Adv}_{\mathcal{A}}(\mathcal{G}^{\textsf{IND-CPA}}) := |\Pr(\mathcal{G}^{\textsf{Ind-CPA}}(\mathcal{A}, \lambda) \to \top) - 1/2|$ is negligible (in the security parameter). The security game is defined as follows.*

$\mathcal{G}^{\textsf{IND-CPA}}$

```
1: $(pk, sk) \leftarrow \mathsf{KeyGen}(1^\lambda)$
2: $(m_0, m_1) \leftarrow \mathcal{A}(pk)$
3: $b \hookleftarrow_\$ \{0,1\}$
4: $b' \leftarrow \mathcal{A}(pk, \mathsf{Enc}(pk, m_b))$
5: If $b = b'$, then return $\top$, else return $\bot$
```

Naturally, any combination of aforementioned properties can be formalized similarly. This is a good warm-up exercise Exo 3 . Some security properties are stronger than others, and some are incompatible with some extra features one may want for a PKE scheme. We will explore those incompatibilities and hierarchy of security notions during tutorials.

• 1.1.2 An example of IND-CPA scheme: El-Gamal

The El-Gamal public-key encryption scheme was coined by Taher Elgamal in 1985, and is still the ciphering system behing GPG, the Gnu Privacy Guard that allows transmission of signed and encrypted electronic messages. It relies on the Decicisional Diffie-Hellmann (DDH) problem, which is semantically close to the Discrete logarithm (DL) problem.

Underlying mathematical assumptions. Before reducing the IND-CPA security from the hardness of the DDH problem, we first define things properly. Recall that a cyclic group is entirely generated by the orbit of a generator, and that for a finite group its order denotes the cardinal of the group.

Definition 5 (DL, CDH, DDH). *Let G be a cyclic group of order q, and g a generator of it. The Discrete Logarithm, Computation and Decisional Difffie-Hellmann problems are defined as follows.*

- ***DL.*** *Given (G, q, g, g^a) for $a \hookleftarrow_\$ [q]$, the goal is to recover a.*
- ***CDH.*** *Given (G, q, g, g^a, g^b) for $(a, b) \hookleftarrow_\$ [q]^2$, the goal is to compute g^{ab}.*
- ***DDH.*** *Given (G, q, g) and access to an oracle $\mathcal{O}^{\mathsf{samples}}$, decide whether $\mathcal{O}^{\mathsf{samples}}$ is returning samples of the form (g^a, g^b, g^{ab}) or (g^a, g^b, g^c) for $(a, b, c) \hookleftarrow_\$ [q]^3$.*

Those problems can be hierarchized by reducing them to each other Exo 4. It seems logical that the hardness of the problem strongly depend on G Exo 5. Historical instantiation consider $(\mathbb{Z}_p^*, *)$ with p prime, while subgroup of the latter are now considered (quadratic residues)[*]. Elliptic curves over finite fields are another widely spread choice.

The scheme. Defining the scheme amounts to precise the algorithms behind the PKE interface. Here are the algorithms considered in the El-Gamal scheme.

Definition 6 (El-Gamal PKE). *The El-Gamal PKE consists in the following three algorithms:*

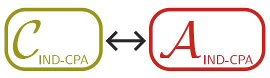
- *KeyGen(1^λ) produces (G, q, g) – a cyclic group, its order and a generator of it – then samples $x \hookleftarrow_\$ \mathbb{Z}_q$ and computes $h = g^x$. The secret key is x, the public key is (G, q, g, h).*
- *Enc$(pk, m \in G)$ samples $y \hookleftarrow_\$ \mathbb{Z}_q$ and returns $c = (g^y, h^y \cdot m)$.*
- *Dec$(sk, c = (c_1, c_2))$ returns c_2/c_1^x.*

It can readily be checked that all those algorithms run in polynomial time and that the scheme is correct. The main work to be achieved concerns the security proof.

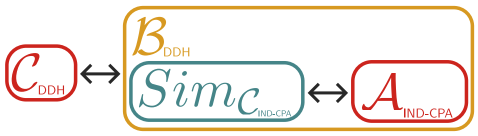
Finally, a proof of security. In order to prove security, we rely on contraposition. We first make a hardness assumption H and formalize a security property S. Then, we prove $\neg S \Rightarrow \neg H$, which is equivalent to $H \Rightarrow S$. Finally, we obtain S by *modus ponens*.

Proposition 7. *Under the assumption the DDH is hard, the ElGamal encryption schme is IND-CPA secure.*

Proof. Let assume that there exists an adversary $\mathcal{A}$, ie. a polytime algorithm, that wins the IND-CPA game with non-negligible probability. We build upon an adversary $\mathcal{B}$ against the DDH problem. The latter will internaly call $\mathcal{A}$ and simulate the behavior of a challenger $\mathcal{C}$ through $Sim_{\mathcal{C}}$. From the adversary point of view, the behavior of $Sim_{\mathcal{C}}$ and $\mathcal{C}$ have to be the same (otherwise, we have no guarantee that $\mathcal{A}$ will succeed in breaking the IND-CPA security of the scheme). Nevertheless, $\mathcal{B}$ can access any internal state of $Sim_{\mathcal{C}}$ and make use of tricks.


$\mathcal{C}_{\text{IND-CPA}} \leftrightarrow \mathcal{A}_{\text{IND-CPA}}$

$\mathcal{A}$'s world within IND-CPA game


$\mathcal{B}_{\text{DDH}}$
$\mathcal{C}_{\text{DDH}} \leftrightarrow Sim_{\mathcal{C}_{\text{IND-CPA}}} \leftrightarrow \mathcal{A}_{\text{IND-CPA}}$

$\mathcal{A}$'s world within our reduction

[*]The non-prime order and the easyness of the decisional variant of the DL problem was limiting.

This reduction is pretty straightforward. First, the challenger generate a DDH instance, that is a tuple (G, q, g, g^x, g^y, g^z), and hands it to $\mathcal{B}$. $\mathcal{B}$ then simulate the key generation step of $Sim_{\mathcal{C}}$ by forwarding (G, q, g, g^x) to $\mathcal{A}$, which is run internaly. At some point, she receives (m_0, m_1) from $\mathcal{A}$ and compute the response of $Sim_{\mathcal{C}}$ as follows: she samples evenly $b \hookleftarrow_{\$} \{0,1\}$ and send $(c_1, c_2) = (g^y, g^z \cdot m_b)$. Finally, after receiving $\mathcal{A}$'s guess, she output $\top$ to $\mathcal{C}$ if the guess is right, and $\perp$ otherwise.

Let's now computes the advantage of $\mathcal{B}$, $\mathrm{Adv}(\mathcal{B}) = |\Pr(\mathcal{B} \xrightarrow{\mathsf{RAND}} \top) - \Pr(\mathcal{B} \xrightarrow{\mathsf{DDH}} \top)|$.

- In the RAND case, the quantity z of the DDH instance is sampled uniformly at random and independantly of other variables. Hence, cipherpart c_2 generated by $Sim_{\mathcal{C}}$ is also random and independant of other quantities. Hence, $\mathcal{A}$ cannot distinguish between the ciphertexts: its advantage is zero, its success probability 1/2. Finally, $\mathcal{B}$ returns $\top$ to $\mathcal{C}$ with probability exactly 1/2.

- In the DDH case, the quantity z is equal to xy. It follows that (c_1, c_2) is a well-formed Elgamal ciphertext. Hence, the probability that $\mathcal{B}$ returns $\top$ is exactly the winning probability of $\mathcal{A}$ within the IND-CPA game, which is at non-negligible distance from 1/2 by assumption.

Finally, it follows that one can reduce an adversary $\mathcal{A}$ against the IND-CPA game into an adversary against the DDH problem with constant time overhead and achieving the same probability of success. By contraposition, this concludes the proof. ■

• 1.1.3 The RSA cryptosystem, an history of refinements

Let review another common public-key encryption scheme introduced by Rivest, Shamir and Adleman in 1977: the RSA[*] cryptosystem. While widely known PKE – the RSA acronym often rings a bell for non-specialists – had progressively been replaced by elliptic-curves based schemes, it is still extensively used in the industry (most public key infrastructures products rely on RSA) and on the web (PGP).

The RSA trapdoor permutation Informally, a trapdoor permutation is a bijection from a set to itself that is easy to compute but hard to invert *unless* one knows some *trapdoor* information. More formally,

Definition 8 (Trapdoor permutation). *A family of trapdoor permutations (TP) is tuple three polytime algorithms* (Gen, Eval, Invert) *such that*

- *The generation algorithm* Gen *takes as input the security paremeter* λ *and returns a pair* (i, τ) *made of a function index and a trapdoor.*
- *The evaluation algorithm* Eval *is deterministic and for any index* i*, the function* $f_i : x \mapsto \mathsf{Eval}(i, x)$ *is a bijection over* D_i*.*
- *The invertion algorithm* Invert *is deterministic and for any pair* $i, \tau) \leftarrow \mathsf{Gen}(1^\lambda)$ *it holds for any* x *that* $\mathsf{Invert}(\tau, \mathsf{Eval}(i, x)) = x$*, ie.* $x \mapsto \mathsf{Invert}(\tau, x)$ *is precisely* f_i^{-1}.

It is said secure if for any PPT adversary $\mathcal{A}$*,*

$$\Pr\left[\begin{array}{c}(i,\tau) \leftarrow \mathsf{Gen}(1^\lambda), x \hookleftarrow_{\$} D_i, \\ y \leftarrow \mathsf{Eval}(i,x) : x = \mathcal{A}(i,y)\end{array}\right] \leq \mathrm{negl}(\lambda).$$

Before introducing the RSA trapdoor permutation, recall that *Euler's totient function*, usually denoted , is the function associating to any positive integer n the cardinal of the set made of numbers no greater than n that are relatively primes to n. Amongs remarkable properties of this function,

[*]Which is readily the acronym of its authors.

$\phi(n)$ is the order of the multiplicative group of integers modulo n, written $\mathbb{Z}_n$. In particular, from Euler's theorem it follows that for any inversible $x \in (\mathbb{Z}_N)^*$, $x^{\phi(N)} \equiv 1 \mod N$.

So, as mentionned in introduction, Rivest, Shamir and Adleman conjectured the following function family to be a trapdoor permutation.

Definition 9 (RSA-TP). *The RSA trapdoor permutations are defined by the following algorithms.*

- *Gen*(1^λ) *samples* $N := pq$, *with* p *and* q *primes being of similar bitsize (depending on* λ*). It samples* e *relatively prime with* $\phi(n)$*. It computes* d *such that* $ed \equiv 1 \mod \phi(n)$*. Finally, it returns* $(N, e), d$*.*
- *Eval*$((N, e), x)$ *returns* $(m^e \mod N)$*, hence describes a permutation over* $\mathbb{Z}_N^*$*.*
- *Invert*(d, y) *returns* $(y^d \mod N)$*.*

We call this conjecture RSA in these notes, and this is a problem no harder than the factorisation problem Exo 6. Whether those problems are equivalent or not is still an open problem, while it seems that there is no better (on reasonable set of parameters) way to attack RSA than attacking FACT.

Definition 10 (FACT, RSA). *Let* $N = pq$ *be a semiprime. The Factorisation and the RSA problems are defined as follows:*

- ***FACT.*** *Given* N*, recover* p *and* q*.*
- ***RSA.*** *Given* e *such that* $\gcd(e, \phi(N)) = 1$*, where* ϕ *denotes Euler's totient function, and* $c \in \mathbb{Z}_N^*$*, find* m *such that* $m^e = c \mod N$

Textbook RSA One may remark how close the definition of trapdoor permutation seems to be from public-key encryption, and that's exactly the idea behind the scheme known as *textbook-RSA*. Morally, the trapdoor will play the role of the secret key: in order to encrypt a message, simply compute its image through the permutation – the security might comes from the non-inversibility of the permutation without knowing the trapdoor. More formally, the scheme is as follows.

Definition 11 (Textbook RSA). *The textbook RSA scheme is defined by the three algorithms:*

- *KeyGen*(1^λ) *samples* (N, e, d) *mimicing Gen's algorithm of the RSA trapdoor permutations. It hands* (N, e) *as the public key, and* d *as the secret key.*
- *Enc*(pk, m) *returns* $c = m^e \mod N$*.*
- *Dec*(sk, m) *returns* $m = c^d \mod N$*.*

This scheme is broken in many ways, and here are some of the simplest ones.

The first attack against textbook RSA is simply that the scheme is not IND-CPA, which is the basic security notion of a scheme.[*] Indeed, as a *deterministic* PKE scheme, it is necessarily vulnerable to such attacks Exo 7. In particular, it allows an attacker to search for a specific plaintext – which is not as costly as it seems in particular contexts: birthday, pincode, ballot, etc. And even without recovering the corresponding plaintext, deterministic encryption made it easy to decide whether the same message is encrypted twice. Against another security model, remarking that the scheme is homomorphic prevents it from being OW-CCA: one could call the decryption oracle on $2c$, get m and finally return $m/2$.

The second attack takes advantage of the following fact: whenever m is smaller than $N^{1/e}$, no modulus reduction is applied to the ciphertext. Hence, computing the e-th root over integers suffices to recover the corresponding plaintext. In practical instantiations of RSA, e is taken as small as 3 for efficiency reasons and would lead a non-negligible part of the message space vulnerable to such

[*]The OW-CPA property is often considered too weak.

an attack. A third attack scenario is the one of broadcast encryption, where a message m is sent to many recipient why different key materials; exception made of e we supposed fixed by the protocol. Here is how to exploit a collection of cipertexts $\{c_i = m^e \mod N_i)\}_{0\leq i<e}$ where the N_i's are pairwise coprime by exploiting the Chinese Remainder Theorem we recall here for completeness.

Theorem 12 (Chinese remainder theorem)**.** *For $(n_1, \cdots, n_k)$ pairwise coprime, and $N = \prod_i n_i$, the map $x \mod N \mapsto (x \mod n_1, \cdots, x \mod n_k)$ defines a ring isomorphism between the ring of integers modulo N and the direct product of the rings of integers modulo the n_i's.*

Denoting by N^* the product of the N_i's, it follows that one can efficiently compute the unique c^* such that $c^* = m^e \mod N^*$. Now observe that m is smaller than the smallest N_i as we supposed well-formed the ciphertexts, causing m^e to be smaller than N^*. As no modular reduction occurs, taking the e-root suffices (again) to recover m.

Paddings and ROM to the rescue For all of those reasons, RSA encryption concretly consists in two phases: a preprocessing phase during which a message is padded and transformed, and then an encryption phase following aforementioned algorithms.

A first (depreciated) standard, introduced in 1993 by RSA laboratories, we discuss here is `PKCS1-v1.5`. It suggests the use of padding to introduce non-determinism during the ciphering process and circumvent direct attacks on IND-CPA security. Denoting by k the length of N in bytes, assuming m to be encoded as an up-to-$(k-11)$-bytes word, a message m is padded into

$$\tilde{m} = 00000000 \| 00000010 \| r \| 00000000 \| m,$$

where r is an at-least-eight-bytes-long randomly generated (with none of its bytes being 0^8), before being ciphered into $c = \tilde{m}^e \mod N$. The first 0 block ensures that the encryption block, converted into an integer, is less than the modulus. Without security proof, it was (and unfortunately still is) widely deployed on web servers and browsers. In 1998, Bleinchenbacher published an padding oracle attack against `PKCS1-v1.5`. In this security model, the attacker has access to a padding oracle $\mathcal{O}^{\mathsf{padding}}$ that takes ciphertexts as inputs and returns $\top$ if the padding of the decrypted ciphertext is correct[*], $\perp$ otherwise. It is important to note that in real-life the server not necessarily hand $\top$ or $\perp$, but it suffices that its behavior depends on the validity of the padding (response time, error throwing, etc.) to turn it into a padding oracle.

Lemma 13 (Bleichenbacher)**.** *The `PKCS1-v1.5` standard is vulnerable to padding oracle attacks.*

Proof. The key observation is that for c a ciphertext, the fact that $\mathcal{O}^{\mathsf{padding}}(c)$ returns $\top$ carries the information that the corresponding plaintext starts with `0x00`$\|$`0x02`.

For a target ciphertext $c = \mathsf{Enc}(pk, \tilde{m})$, the attacker produce a collection of re-randomized ciphetexts $\{c_i = cr_i^e = (\tilde{m}r_i)^e \mod N\}_i$ and submits them to the padding oracle. Letting $B = 2^{8(k-2)}$, the sets of c_i's upon which the oracle returned $\top$ are such that $2B \leq \tilde{m}r_i \leq 3B$. This set of equations suffices to recover $\tilde{m}$ efficiently. ■

As a consequence, the proposed standard is not IND-CCA[†] Exo 8.

Another padding scheme was proposed in 1994 by Bellare and Rogaway, and was called OAEP for Optimal Asymmetric Encryption Padding. It is build upon two hash functions G and H, that respectively compress and extend randomness of inputs (ie. with range respectively smaller/wider than their definition domain). For a message m, the corresponding ciphertext is roughly Exo 9 :

$$\tilde{m}^e \mod N, \text{where } \tilde{m} = ((m\|\mathbf{0}) \oplus G(r) \| r \oplus H((m\|\mathbf{0}) \oplus G(r))),$$

where $\mathbf{0}$ denotes null bytes, r is a random bit string, and $\oplus$ denote the (bitwise) XOR operator. This padding can efficiently be removed Exo 10 , which make it suitable for cryptography. In the

[*]That is the plaintext starts with the null and "2" bytes, followed by at least 8 non-null bytes and remaining bytes starting with a null byte.

[†]In fact, it is not IND-CPA neither, but this is beyond the scope of those lecture notes.

case where hash functions are idealized[*], the padding scheme turns the textbook RSA scheme into a CCA secure scheme under the RSA hypothesis[†].

Lemma 14 (Fujisaki-Okamoto-Pointcheval-Stern)**.** *In the random oracle model,* `RSA-OAEP` *is IND-CCA secure if the RSA problem is hard.*

This ends our (small) RSA journey.

• 1.1.4 Generically increasing security: the Fujisaki-Okamoto transform

Ongoing work

• 1.1.5 A word about key-encapuslation mechanisms

Ongoing work

This blank space will be removed as soon as 1.1.4 and 1.1.5 subsubsections are complete.

[*]This will be precised in the next subsection.

[†]Initially, the OAEP transformation was claimed secure for any underlying trapdoor permutation. Unfortunately, a flaw in the original proof causes the later statement to be incorrect. The refined transformation OAEP+, of Shoup, achieves what OAEP aimed to.

Going further

Here are a few bibliographic references upon which the PRS is partially based. You may explore them for further insight. You can also reach us by email.

- Katz, J., & Lindell, Y. (2021). *Introduction to modern cryptography (3rd ed.).* CRC Press.
- Vaudenay, S. (2006). *A Classical Introduction to Cryptography: Applications for Communication Security*. Springer.
- Smart, N. P. (2016). *Cryptography Made Simple*. Springer.
- Boneh, D., Shoup, V. (2023). *A Graduate Course in Applied Cryptography (draft 0.6).* Available online.

Exercises left along the way

Exo 1 Give an example of non-correct PKE scheme. Give an example of a correct PKE scheme that won't verify any reasonable security property.

Exo 2 Prove Lemma 3 formaly.

Exo 3 Give formal security games for the following properties: OW-CPA, IND-CCA, NM-CPA.

Exo 4 Produce as many (simple) reductions as you can between those three problems. Write down at least one of them formally.

Exo 5 Give an example of a group where the DL problem is easy.

Exo 6 Show that RSA reduces to FACT.

Exo 7 Show that any IND-CPA PKE scheme has a non-deterministic encryption function.

Exo 8 Prove that any IND-CCA secure scheme is secure against padding oracle attacks.

Exo 9 Assuming that hash functions operate on bitstrings, precise their range/domain and the length of m.

Exo 10 Given a padded message $\tilde{m}$, show how one can recover the underlying message m back.