To appear in Recent Trends in Cryptography, I. Luengo (Ed.), Contemporary Mathematics series, AMS-RSME, 2008.

# **Public-Key Cryptanalysis**

Phong Q. Nguyen

ABSTRACT. In 1976, Diffie and Hellman introduced the revolutionary concept of public-key cryptography, also known as asymmetric cryptography. Today, asymmetric cryptography is routinely used to secure the Internet. The most famous and most widely used asymmetric cryptosystem is RSA, invented by Rivest, Shamir and Adleman. Surprisingly, there are very few alternatives known, and most of them are also based on number theory. How secure are those asymmetric cryptosystems? Can we attack them in certain settings? Should we implement RSA the way it was originally described thirty years ago? Those are typical questions that cryptanalysts have tried to answer since the appearance of public-key cryptography. In these notes, we present the main techniques and principles used in public-key cryptanalysis, with a special emphasis on attacks based on lattice basis reduction, and more generally, on algorithmic geometry of numbers. To simplify our exposition, we focus on the two most famous asymmetric cryptosystems: RSA and Elgamal. Cryptanalysis has played a crucial rôle in the way cryptosystems are now implemented, and in the development of modern security notions. Interestingly, it also introduced in cryptology several mathematical objects which have since proved very useful in cryptographic design. This is for instance the case of Euclidean lattices, elliptic curves and pairings.

# 1. Introduction

Public-key cryptography, also called asymmetric cryptography, was invented by Diffie and Hellman [**DH76**] more than thirty years ago. In public-key cryptography, a user U has a pair of related keys (pk, sk): the key pk is public and should be available to everyone, while the key sk must be kept secret by U. The fact that sk is kept secret by a single entity creates an asymmetry, hence the name *asymmetric cryptography*, to avoid confusion with symmetric cryptography where a secret key is always shared by at least two parties, whose roles are therefore symmetric. The alternative (and perhaps more common) name *public-key cryptography* comes from the very existence of a public key: in conventional cryptography, all keys are secret.

<sup>1991</sup> Mathematics Subject Classification. Primary 94A60, 11T71; Secondary 11H06, 14G50, 68P25.

Key words and phrases. Cryptanalysis, Security, Public-Key Cryptography, Asymmetric Cryptography, Euclidean Lattices, Geometry of Numbers.

### PHONG Q. NGUYEN

Today, public-key cryptography offers incredibly many features ranging from zero-knowledge to electronic voting (see the handbook [MOV97]), but we will restrict to its main goals defined in [DH76], which are the following two:

- Asymmetric encryption (also called public-key encryption): anyone can encrypt a message to U, using U's public key pk. But only U should be able to decrypt, using his secret key sk.
- Digital signatures: U can sign any message m, using his secret key sk. Anyone can check whether or not a given signature corresponds to a given message and a given public key.

Such basic functionalities are routinely used to secure the Internet. For instance, digital signatures are prevalent under the form of certificates (which are used everyday by Internet browsers), and asymmetric encryption is used to exchange session keys for fast symmetric encryption, such as in the TLS (Transport Layer Security) protocol.

**1.1. Hard problems.** Both keys pk and sk are related to each other, but it should be computationally hard to recover the secret key sk from the public key pk, for otherwise there would be no secret key. As a result, public-key cryptography requires the existence of hard computational problems. But is there any provably hard computational problem? This is a very hard question underlying the famous  $P \neq NP$  conjecture from complexity theory. Instead of trying to settle this major open question, cryptographers have adopted a more down-to-earth approach by trying various candidates over the years: if a computational problem resists the repeated assaults of the research community, then maybe it should be considered hard, although no proof of its hardness is known or sometimes, even expected. Furthermore, it is perhaps worth noting that the  $P \neq NP$  conjecture refers to worst-case hardness, while cryptography typically requires average-case hardness. The (potentially) hard problems currently in consideration within public-key cryptography can be roughly classified into two families.

The first family of hard problems involves problems for which there are very few unknowns, but the size of the unknowns must be rather large to guarantee hardness, which makes the operations rather slow compared to symmetric cryptography. The main members of this family are:

• Integer factorization, popularized by RSA [**RSA78**]. The current factorization record for an RSA number (*i.e.* a product of two large primes) is the following factorization [**BBFK05**] of a 200-digit number (663 bits), obtained with the number field sieve (see the book [**CP01**]):

2799783391	1221327870	8294676387	2260162107	0446786955	4285375600
0992932612	8400107609	3456710529	5536085606	1822351910	9513657886
3710595448	2006576775	0985805576	1357909873	4950144178	8631789462
9518723786	9221823983	=	3532461934	4027701212	7260497819
8464368671	1974001976	2502364930	3468776121	2536794232	0005854795
6528088349	×	7925869954	4783330333	4708584148	0059687737
9758573642	1996073433	0341455767	8728181521	3538140930	4740185467

A related (and not harder) problem is the so-called *e*-th root problem, which we will discuss when presenting RSA.

• The discrete logarithm problem in appropriate groups, such as:

- Multiplicative groups of finite fields, especially prime fields, like in the DSA signature algorithm [Nat94]. The current record for a discrete logarithm computation in a general prime field is 160 digits [Kle07], obtained with the number field sieve.
- Additive groups of elliptic curves over finite fields. There are in fact two kinds of elliptic curves in consideration nowadays:
  - \* Random elliptic curves for which the best discrete logarithm algorithm is the generic square root algorithm. It is therefore no surprise that the current discrete logarithm record for those curves is 109 bits [**HDdL00**].
  - \* Special elliptic curves (*e.g.* supersingular curves) for which an efficient pairing is available. On the one hand, this decreases the hardness of the discrete logarithm to the case of finite fields (namely, a low-degree extension of the base field of the curve), which implies bigger sizes for the curves, but on the other hand, it creates exciting cryptographic applications such as identity-based cryptography (see [**Men08**] and the book [**BSS04**]).

Interestingly, these problems would theoretically not resist to large-scale quantum computers (as was famously shown by Shor [Sho99]), but the feasibility of such devices is still open.

The second family of hard problems involves problems for which there are many small unknowns, but this number of small unknowns must be rather large to guarantee hardness. Such problems are usually related to NP-hard combinatorial problems for which no efficient quantum algorithm is known. The main examples of this family are:

- Knapsacks and lattice problems. In the knapsack problem, the unknowns are bits. The Merkle-Hellman cryptosystem [MH78], an early alternative to RSA, was based on the knapsack (or subset sum) problem. Although knapsack cryptosystems have not been very successful (see the survey [Odl90]) due to lattice attacks, they have in some sense enjoyed a second coming under the disguise of lattice-based cryptosystems (see the survey [NS01]). Of particular interest is the very efficient NTRU cryptosystem [HPS98], which offers much smaller keys than other lattice-based or knapsack-based schemes. Knapsacks and lattice problems are tightly connected.
- Coding problems. The McEliece cryptosystem [McE78] is a natural cryptosystem based on the hardness of decoding, which has several variants depending on the type of code used. The lattice-based Goldreich-Goldwasser-Halevi cryptosystem [GGH97, Ngu99] can be viewed as a lattice-based analogue of the McEliece cryptosystem.
- Systems of multivariate polynomial equations over small finite fields. The Matsumoto-Imai cryptosystem [MI88] is the ancestor of what is now known as multivariate cryptography (see the book [Kob98]). In order to prevent general attacks based on Gröbner bases, the security parameter must be rather large. All constructions known use a system of equations with a very particular structure, which they try to hide. Like knapsack cryptography, many multivariate schemes have been broken due to their

exceptional structure. The latest example is the spectacular cryptanalysis [**DFSS07**] of the SFLASH signature scheme.

The main drawback with this second family of problems is the overall size of the parameters. Indeed, apart from NTRU [**HPS98**], the size of the parameters for such problems grows at least quadratically with the security parameter. NTRU offers a smaller keysize than the other members of this family because it uses a compact representation, which saves an order of magnitude.

**1.2.** Cryptanalysis. Roughly speaking, cryptanalysis is the science of codebreaking. We emphasized earlier that asymmetric cryptography required hard computational problems: if there is no hard problem, there cannot be any asymmetric cryptography either. If any of the computational problems mentioned above turns out to be easy to solve, then the corresponding cryptosystems can be broken, as the public key would actually disclose the secret key. This means that one obvious way to cryptanalyze is to solve the underlying algorithmic problems, such as integer factorization, discrete logarithm, lattice reduction, Gröbner bases, *etc.* Here, we mean a study of the computational problem in its full generality.

Alternatively, one may try to exploit the special properties of the cryptographic instances of the computational problem. This is especially true for the second family of hard problems: even though the underlying general problem is NP-hard, its cryptographic instances may be much easier, because the cryptographic functionalities typically require an unusual structure. In particular, this means that maybe there could be an attack which can only be used to break the scheme, but not to solve the underlying problem in general. This happened many times in knapsack cryptography and multivariate cryptography. Interestingly, generic tools to solve the general problem perform sometimes even much better on cryptographic instances (see [FJ03] for Gröbner bases and [GN08b, NS01] for lattice reduction).

However, if the underlying computational problem turns out to be really hard both in general and for instances of cryptographic interest, this will not necessarily imply that the cryptosystem is secure. First of all, it is not even clear what is meant exactly by the term *secure* or *insecure*. Should an encryption scheme which leaks the first bit of the plaintext be considered secure? Is the secret key really necessary to decrypt ciphertexts or to sign messages? If a cryptosystem is theoretically secure, could there be potential security flaws for its implementation? For instance, if some of the temporary variables (such as pseudo-random numbers) used during the cryptographic operations are partially leaked, could it have an impact on the security of the cryptosystem? This means that there is much more to cryptanalysis than just trying to solve the main algorithmic problems. In particular, cryptanalysts are interested in defining and studying realistic environments for attacks (adaptive chosen-ciphertext attacks, side-channel attacks, etc.), as well as the goals of attacks (key recovery, partial information, existential forgery, distinguishability, etc.). This is very much related to the development of *provable security*, a very popular field of cryptography. Overall, cryptanalysis usually relies on three types of failures:

- Algorithmic failures: The underlying hard problem is not as hard as expected. This could be due to the computational problem itself, or to special properties of cryptographic instances.
- **Design failures:** Breaking the cryptosystem is not as hard as solving the underlying hard problem.

**Implementation failures:** Exploiting additional information due to implementation mistakes or side-channel attacks. This is particularly relevant to the world of smartcards, and is not well covered by provable security.

Thirty years after the introduction of public-key cryptography, we have a much better understanding of what security means, thanks to the advances of public-key cryptanalysis. It is perhaps worth noting that cryptanalysis also proved to be a good incentive for the introduction of new techniques in cryptology. Indeed several mathematical objects now invaluable in cryptographic design were first introduced in cryptology as cryptanalytic tools, including:

- Euclidean lattices, whose first cryptologic use was the cryptanalysis [Adl83, Sha82] of the Merkle-Hellman cryptosystem [MH78]. Besides crypt-analysis, they are now used in lattice-based cryptosystems (see the survey [NS01]), as well as in a few security proofs [Sh001, FOPS01, CNS02].
- Elliptic curves. One might argue that the first cryptologic usage of elliptic curves was Lenstra's ECM factoring algorithm [Len87], before the proposal of cryptography based on elliptic curves [Kob87, Mil87]: both articles [Kob87, Mil87] mention a draft of [Len87] in their introduction.
- Pairings, whose first cryptologic use was cryptanalytic [**MOV93**], to prove that the discrete logarithm problem in certain elliptic curves could be reduced efficiently to the discrete logarithm problem in finite fields. See [**Men08**] and the book [**BSS04**] for positive applications of pairings.

1.3. Road Map. In these notes, we intend to survey the main principles and the main techniques used in public-key cryptanalysis. As a result, we will focus on the two most famous (and perhaps simplest) asymmetric cryptosystems: RSA [RSA78] and Elgamal in prime fields [El 85], which we will recall in Section 2. Unfortunately, this means that we will ignore the rich cryptanalytic literature related to the second family of hard problems mentioned in Section 1.1, as well as that of elliptic-curve cryptography. Another important topic of cryptanalysis which we will not cover is side-channel cryptanalysis (as popularized by [Koc96, BDL97]).

In Section 3, we review the main security notions, which we will illustrate by simple attacks in Section 4. In Section 5, we present a class of rather elementary attacks known as square-root attacks. In Section 6, we introduce the theory of lattices, both from a mathematical and a computational point of view, which is arguably the most popular technique in public-key cryptanalysis. This will be needed for Section 7 where we present the vast class of lattice attacks.

# 2. Textbooks Cryptosystems

In order to explain what is public-key cryptanalysis, it would be very helpful to give examples of attacks. Although plenty of interesting cryptanalyses have been published in the research literature (see the collections of proceedings [MZ98, IAC04]), many require a good understanding of the underlying cryptosystem, which may not be very well-known and may be based on unusual techniques. To simplify our exposition, we only present attacks on the two most famous cryptosystems: RSA [RSA78] and Elgamal over prime fields [El 85]. Both cryptosystems have the additional advantage of being very easy to describe. We refer to these cryptosystems as textbook cryptosystems, because we will consider the original

### PHONG Q. NGUYEN

description of those schemes, the one that can be found in most cryptography textbooks, but not the one which is actually implemented in practice nowadays. Cryptanalysis has played a crucial role in the way cryptosystems are now implemented. We now recall briefly how RSA and Elgamal work.

**2.1. RSA.** The RSA cryptosystem [**RSA78**] is the most widely used asymmetric cryptosystem. It is based on the hardness of factoring large integers.

2.1.1. Key generation. The user selects two large primes p and q (of the same bit-length) uniformly at random, so that N = pq is believed to be hard to factor. As previously mentioned, the factoring record for such numbers is currently a 663-bit N. In electronic commerce, the root certificates used by Internet browsers typically use a N of either 1024 or 2048 bits.

Next, the user selects a pair of integers (e, d) such that:

(2.1) 
$$ed \equiv 1 \pmod{\phi(N)},$$

where  $\phi(N) = (p-1)(q-1)$  is Euler's function:  $\phi(N)$  is the number of integers in  $\{1, \ldots, N-1\}$  which are coprime with N. The integers e and d are called the RSA exponents: e is the public exponent, while d is the secret exponent. The RSA public key is the pair (N, e), and the RSA secret key is d. The primes p and q do not need to be kept.

There are essentially three ways to select the RSA exponents:

- **Random exponents:** The user selects an integer  $d \in \{2, ..., \phi(N) 1\}$  uniformly at random among those which are coprime with  $\phi(N)$ . The public exponent e is chosen as the inverse of d modulo  $\phi(N)$ .
- **Low Public Exponent:** To speed up public exponentiation, the user selects a very small e, possibly with low Hamming weight. If e is not invertible modulo  $\phi(N)$ , then the user selects a new pair (p,q) of primes, otherwise, the secret exponent d is chosen as the inverse of e modulo  $\phi(N)$ . The most popular choices are e = 3 and  $e = 2^{16} + 1 = 65537$ . Note that e must be odd to have a chance of being invertible modulo  $\phi(N)$ .
- Short Secret Exponent: To speed up private exponentiation, the user selects this time a short d, with a sufficiently long bit-length so that it cannot be exhaustively searched. If d is not invertible modulo  $\phi(N)$ , a new d is picked. Otherwise, the public exponent e is chosen as the inverse of dmodulo  $\phi(N)$ . This choice of d is however not recommended: it is known that it is provably insecure [**Wie90**] if  $d \leq N^{1/4}$ , and it is heuristically insecure [**BD99**] if  $d \leq N^{1-1/\sqrt{2}} \approx N^{0.292...}$ . In such attacks (which we will describe in later sections), one may recover the factorization of N, given only the public key (N, e).

If one knows the factorization of N, then one can obviously derive the secret exponent d from the public exponent e. In fact, it is well-known that the knowledge of the secret exponent d is equivalent to factoring N. More precisely, it was noticed as early as in [**RSA78**] that if one knows the secret key d, then one can recover the factorization of N in probabilistic polynomial time. It was recently proved in [**CM04**] that this can actually be done in deterministic polynomial time. Hence, recovering the RSA secret key is as hard as factoring the RSA public modulus, but this does not necessarily mean that breaking RSA is as hard as factoring.

2.1.2. Trapdoor permutation. We denote by  $\mathbb{Z}_N$  the ring  $\mathbb{Z}/N\mathbb{Z}$ , which we represent by  $\{0, 1, \ldots, N-1\}$ . The main property of the RSA key generation is the congruence (2.1) which implies, thanks to Fermat's little theorem and the Chinese remainder theorem, that the modular exponentiation function  $x \mapsto x^e$  is a permutation over  $\mathbb{Z}_N$ . This function is called the RSA permutation. It is well-known that its inverse is the modular exponentiation function  $x \mapsto x^d$ , hence the name trapdoor permutation: if one knows the trapdoor d, one can efficiently invert the RSA permutation. Without the trapdoor, the inversion problem is believed to be hard, and is known as the e-th root problem (also called the RSA problem): given an integer  $y \in \mathbb{Z}_N$  chosen uniformly at random, find  $x \in \mathbb{Z}_N$  such that  $y \equiv x^e \mod N$ . The RSA assumption states that no probabilistic polynomial-time algorithm can solve the RSA problem with non-negligible probability.

It is however unknown if the knowledge of d is necessary to solve the e-th root problem. Maybe there could be an alternative way to invert the RSA permutation, other than raising to the power d. In fact, the work [**BV98**] suggests that the e-th root problem with a small e might actually be easier than factoring.

An important property of the RSA permutation is its multiplicativity. More precisely, for all x and y in  $\mathbb{Z}_N$ :

(2.2) 
$$(xy)^e \equiv x^e y^e \pmod{N}.$$

This homomorphic property will be very useful for certain attacks.

2.1.3. Asymmetric encryption. Textbook-RSA encryption is a simple application of the RSA trapdoor permutation, in which encryption is achieved by applying the RSA permutation. More precisely, the set of messages is  $\mathbb{Z}_N = \{0, 1, \dots, N-1\}$ . To encrypt a message m, one simply raises it to the power e modulo N, which means that the ciphertext is:

$$(2.3) c = m^e \bmod N.$$

To decrypt the ciphertext c, one simply inverts the RSA permutation:

$$(2.4) m = c^d \bmod N.$$

This is the way the RSA public-key encryption scheme was originally described in [**RSA78**], and is still described in many textbooks, but this is not the way RSA is now implemented in various products or standards due to security problems, even though the basic principle remains the same. It is now widely accepted that a trapdoor permutation should not be directly used as a public-key encryption scheme: a preprocessing of the messages is required, *e.g.* OAEP (optimal asymmetric encryption) [**BR95**, **Poi05**]. The attacks we will present in these notes explain why.

It is worth noting that Textbook-RSA encryption is multiplicative like the RSA permutation. If  $m_1$  are  $m_2$  are two messages in  $\mathbb{Z}_N$  encrypted as  $c_1$  and  $c_2$  using (2.3), then their product  $m_3 = (m_1m_2) \mod N$  is encrypted as  $c_3 = (c_1c_2) \mod N$ . In other words, the ciphertext of a product is the product of the ciphertexts.

2.1.4. *Digital signature*. The magical property of RSA is its trapdoor permutation: most public-key cryptosystems known involve a trapdoor one-way function instead (see [**MOV97**]). Fortunately, it is very easy to derive a digital signature scheme from a trapdoor permutation.

In the original description [**RSA78**], the set of messages to sign is  $\mathbb{Z}_N = \{0, 1, \ldots, N-1\}$ . The signature of a message  $m \in \mathbb{Z}_N$  is simply its preimage through the RSA permutation:

$$(2.5) s = m^d \bmod N.$$

To verify that s is the signature of m with the public key (N, e), one checks that  $s \in \mathbb{Z}_N$  and that the following congruence holds:

$$(2.6) m \equiv s^e \pmod{N}.$$

Similarly to the asymmetric encryption case, this is not the way RSA signatures are now implemented in various products or standards due to security problems, even though the basic principle remains the same. Again, we will present attacks which explain why. A trapdoor permutation should not be directly used as a digital signature scheme: a hashing-based preprocessing of the messages is required, *e.g.* FDH (full-domain hash) [**BR96**, **Poi05**] or PSS (probabilistic signature scheme) [**BR96**, **Poi05**].

It is worth noting that the preprocessing now in use in asymmetric encryption or digital signatures involves a cryptographic hash function. However, when **[RSA78]** was published, no cryptographic hash function was available! This is why many *ad hoc* solutions were developed (and sometimes deployed) in the eighties, with various degrees of success. We will describe attacks on some of those. The RSA standards **[Lab]** currently advocated by the RSA Security company are: RSA-OAEP for asymmetric encryption and RSA-PSS for signatures.

2.2. Elgamal. While there is essentially only one RSA cryptosystem, there is much more flexibility with the Elgamal cryptosystem [El 85] based on the hardness of the discrete logarithm problem: it has many variants depending on the group or subgroup used, as well as the encoding of messages and ciphertexts. Here, we only consider the so-called Textbook Elgamal, that is, the basic Elgamal cryptosystem over a prime field  $\mathbb{Z}_p$ , as originally described in [El 85]. Another significant difference with RSA is the gap between the Elgamal asymmetric encryption scheme and the Elgamal digital signature scheme. In RSA, asymmetric encryption and signatures are the two facets of the RSA trapdoor permutation. Because the Elgamal asymmetric encryption scheme involves a trapdoor one-way function based on the Diffie-Hellman key exchange [DH76], rather than a trapdoor permutation, it does not naturally lead to an efficient digital signature scheme. The Elgamal signature scheme is quite different from its asymmetric encryption counterpart: it is the ancestor of most discrete-log based signature schemes, such as DSA, ECDSA or Schnorr's signature (see [MOV97]).

2.2.1. Key generation. The user selects a large random prime p, in such a way that p-1 has at least one large prime factor and has known factorization. It is then believed that the discrete logarithm problem in  $\mathbb{Z}_p^{\times}$  is hard. Thanks to the factorization of p-1, the user can compute a generator g of the multiplicative group  $\mathbb{Z}_p^{\times}$ . There are essentially two ways to select the generator g:

- **Random generators:** This is the recommended option: the generator g is selected uniformly at random among all generators of  $\mathbb{Z}_p^{\times}$ .
- **Small generators:** One tries small values for g, such as g = 2, to speed up exponentiation with base g. If none works, one picks another prime p.

We will later see that the choice g = 2 has dramatic consequences on the security of the Elgamal signature scheme [**Ble96**].

The parameters g and p are public. They can be considered as central parameters, since they can be shared among several users, but if that is the case, it is important that all users are convinced that the parameters have been generated in a random way so that they have no special property.

The user's secret key is an integer x chosen uniformly at random over  $\mathbb{Z}_{p-1} = \{0, 1, \dots, p-2\}$ . The corresponding public key is the integer  $y \in \mathbb{Z}_p^{\times}$  defined as:

$$(2.7) y = g^x \pmod{p}$$

Many variants of Elgamal alternatively use a prime order subgroup, rather than the whole group  $\mathbb{Z}_p^{\times}$ . More precisely, they select an element  $g \in \mathbb{Z}_p^{\times}$  of large prime order  $q \ll p$ : the secret key x is then chosen in  $\mathbb{Z}_q$ .

2.2.2. Asymmetric encryption. The Elgamal asymmetric encryption scheme can be viewed as an application of the Diffie-Hellman key exchange protocol [**DH76**]. In the well-known basic Diffie-Hellman protocol, Alice and Bob do the following to establish a shared secret key:

- Alice selects an integer  $a \in \mathbb{Z}_{p-1}$  uniformly at random, and sends  $A = g^a \mod p$  to Bob.
- Bob selects an integer  $b \in \mathbb{Z}_{p-1}$  uniformly at random, and sends  $B = g^b \mod p$  to Alice.
- The secret key shared by Alice and Bob is  $s = g^{ab} \mod p$ . Alice may compute s as  $s = B^a \mod p$ , while Bob may alternatively compute s as  $s = A^b \mod p$ .

To transform this key exchange protocol into a probabilistic asymmetric encryption scheme, let us view Alice as the user who possesses the pair of keys (x, y) defined in (2.7), so that (a, A) = (x, y), and let us view Bob as the person who wishes to encrypt messages to the user. Bob knows the public key  $y = g^x \mod p$ . The set of plaintexts is  $\mathbb{Z}_p$ . To encrypt a message  $m \in \mathbb{Z}_p$ :

- Bob selects an integer  $k \in \mathbb{Z}_{p-1}$  uniformly at random.
- The ciphertext is the pair  $(c, d) \in \mathbb{Z}_p^{\times} \times \mathbb{Z}_p$  defined as

$$(2.8) c = g^k \pmod{p}$$

$$(2.9) d = my^k \pmod{p}$$

To see how decryption works, notice that thanks to the Diffie-Hellman trick, Alice may compute the (virtual) secret  $s = g^{xk} = y^k \mod p$  from her secret key x and the first half c of the ciphertext. This is because  $s = c^x \mod p$ , as if Bob's pair (b, B) in the Diffie-Hellman protocol was (k, c). Once  $y^k \mod p$  is known, Alice may recover the message m from the second half d of the ciphertext, by division.

In other words, the first half (2.8) of the ciphertext sets up a one-time Diffie-Hellman secret key  $y^k = g^{kx}$ . The second half (2.9) of the ciphertext can be viewed as a one-time pad (using modular multiplication rather than a xor) between the the message and the one-time key. Decryption works by recovering this one-time key using the user's secret key, thanks to the Diffie-Hellman trick.

Since Elgamal encryption [El 85] is very much related to the Diffie-Hellman key exchange [DH76], one may wonder why it did not already appear in [DH76]. Perhaps one explanation is that, strictly speaking, public-key encryption as defined in [DH76] was associated to a trapdoor permutation, so that it would be easy to

derive both encryption and signature: it was assumed implicitly that the set of ciphertexts had to be identical to the set of plaintexts. But Elgamal encryption does not use nor define a trapdoor permutation. The closest thing to a permutation in Elgamal encryption is the following bijection between  $\mathbb{Z}_p \times \mathbb{Z}_{p-1}$  and  $\mathbb{Z}_p^{\times} \times \mathbb{Z}_p$ :

$$(m,k) \mapsto (c,d) = (g^k, my^k)$$

But the secret key x only helps to partially invert this bijection: given an image (c, d), one knows how to efficiently recover the corresponding m, but not the second half k, which is a discrete logarithm problem. Thus, it cannot be considered as a trapdoor permutation. In some sense, it could be viewed as a partial trapdoor permutation.

We saw two significant differences between Textbook-Elgamal encryption and Textbook-RSA encryption: Elgamal is probabilistic rather than deterministic, and it is not based on a trapdoor permutation. Nevertheless, there is one noticeable thing in common: Elgamal is multiplicative too. Indeed, assume that two plaintexts  $m_1$  and  $m_2$  are encrypted into  $(c_1, d_1)$  and  $(c_2, d_2)$  (following (2.8) and (2.9)) using respectively the one-time keys  $k_1$  and  $k_2$ . In a natural way, one could define the product of ciphertexts as  $(c_3, d_3)$  where:

$$c_3 = c_1 c_2 \in \mathbb{Z}_p^\times$$
$$d_3 = d_1 d_2 \in \mathbb{Z}_p$$

Then it can be easily checked that  $(c_3, d_3) \in \mathbb{Z}_p^{\times} \times \mathbb{Z}_p$  would be decrypted as  $m_3 = (m_1m_2) \mod p$  because it is the ciphertext of  $m_3$  with the one-time key  $k_3 = (k_1 + k_2) \mod p$ . Thus, in Textbook-Elgamal as well as Textbook-RSA, the product of ciphertexts is a ciphertext of the product.

2.2.3. Digital signature. Surprisingly, the Elgamal signature scheme [El 85] has nothing to do with the Elgamal asymmetric encryption scheme [El 85]. The only thing in common is the key generation process and the fact that the scheme is probabilistic.

The set of messages is  $\mathbb{Z}_p$ . To sign a message  $m \in \mathbb{Z}_p$ :

- The user selects uniformly at random a one-time key  $k \in \mathbb{Z}_{p-1}^{\times}$ , that is an integer in  $\{0, \ldots, p-2\}$  coprime with p-1.
- The signature of m is the pair  $(a, b) \in \mathbb{Z}_p^{\times} \times \mathbb{Z}_{p-1}$  defined as:

$$(2.10) a = g^k \pmod{p}$$

(2.11) 
$$b = (m - ax)k^{-1} \pmod{p-1}.$$

To verify a given signature (a, b) of a given message m, one checks that  $(a, b) \in \mathbb{Z}_p^{\times} \times \mathbb{Z}_{p-1}$  and that the following congruence holds:

$$(2.12) g^m \equiv y^a a^b \pmod{p}$$

The previous congruence can be equivalently rewritten as:

(2.13) 
$$m \equiv ax + b \log a \pmod{p-1},$$

where log denotes the discrete log in  $\mathbb{Z}_p^{\times}$  with respect to the base g. This rewriting will prove particularly useful when presenting attacks. Note that if the pair (a, b) has been generated according to (2.10) and (2.11), then  $k = \log a$ , so that (2.13) follows easily from (2.11).

#### **3.** Security Notions

Perhaps one of the biggest achievements of public-key cryptography is the introduction of rigorous and meaningful security notions for both encryption and signatures. Rigorous, because these notions can be formally defined using the language of complexity theory. Meaningful, because the relatively young history of public-key cryptography seems to indicate that they indeed capture the "right" notion of security, as various attacks have shown that (even slightly) weaker notions of security would be insufficient. However, it should be noted that security notions do not take into account implementation issues: in particular, side-channel attacks are not currently covered by provable security.

Since our focus is on cryptanalysis, rather than provable security, we will not properly define all the security notions: we will content ourselves with informal definitions, to convey intuitions more easily, and to keep our presentation light. We refer the interested reader to the lecture notes [**Poi05**] for a more technical treatment.

We would like to insist on the following point. Some of the security notions widely accepted today may look a bit artificial and perhaps too demanding at first sight. In fact, it could be argued that it is the discovery of certain realistic attacks which have convinced the community of the importance of such strong notions of security. In other words, public-key cryptanalysis has helped to find the right notion of security, but it has also helped in the acceptance of strong security notions. For instance, it is arguably Bleichenbacher's practical attack [**Ble98**] which triggered the switch to OAEP for RSA encryption in the PKCS standards [**Lab**], even though chosen-ciphertext attacks on RSA had appeared long before.

Roughly speaking, it is now customary to define security notions using games (see the survey [Sho04]): a cryptographic scheme is said to be secure with respect to a certain security notion if a specific game between a challenger and an attacker cannot be won by the attacker with non-negligible probability, where the attacker is modeled as a probabilistic polynomial-time Turing machine with possibly access to oracles: the security notion defines exacly which oracles the attacker has access to. Informally, a security notion consists of two definitions:

- The goal of the attacker. This defines the rules of the game: what is the purpose of the attacker (that is, when is the game won or lost), and how the game is run.
- The means of the attacker. This is where the access to oracles is defined. For instance, in chosen-ciphertext security, the attacker has access to a decryption oracle, which may decrypt any ciphertext apart from the challenge ciphertext.

The oracles may also depend on the security model. For instance, in the well-known random oracle model, a hash function is modeled as an oracle which behaves like a random function.

**3.1. Digital Signatures.** We start with digital signatures because the "right" security notion is fairly natural here. Of all the possible goals of the attacker, the most important are the following ones:

**Key recovery:** The attacker wants to recover the secret key sk of the signer. **Universal forgery:** the attacker wants to be able to sign any message. This is also called a *selective forgery*.

#### PHONG Q. NGUYEN

**Existential forgery:** The attacker wants to exhibit a new signature. By a new signature, one usually means a signature of a new message, but it may also mean a new signature of a message for which a signature was already known, which is meaningful for a probabilistic signature.

Attacks on signature schemes are also classified based on the means available to the attacker:

- No-message attacks: the attacker only knows the public key pk of the signer.
- Known-message attacks: the attacker knows a list of valid random pairs (message,signature).
- **Chosen-message attacks:** the attacker may ask for signatures of messages of his/her choice. If the requests are not independent, the chosen-message attack is said to be *adaptive*. Of course, depending on the goal of the attacker, there is a natural restriction over the requests allowed: for instance, in a universal forgery, the attacker cannot ask for the signature of the challenge message he has to sign.

We will see that the original description of the main signature schemes only satisfy very weak notions of security. To achieve the strongest notions of security under appropriate assumptions, a preprocessing of the message is required, using hash functions, but it is not mandatory to have a probabilistic signature scheme, which is a noteworthy difference with the situation of asymmetric encryption.

**3.2.** Asymmetric Encryption. It took cryptographers significantly longer to define the strongest security notions for asymmetric encryption than for digital signatures, which is a sign that things are arguably more complex with encryption. Of all the possible goals of the attacker, the most important are the following ones:

- **Key recovery:** The attacker wants to recover the secret key sk of the user. **Decryption:** the attacker wants to be able to decrypt any ciphertext. The encryption scheme is said to be *one-way* if no efficient attacker is able to decrypt a random ciphertext with non-negligible probability. By a random ciphertext, we mean the ciphertext of a plaintext chosen uniformly at random over the plaintext space.
- Malleability: Given a list of ciphertexts, the attacker wants to build a new ciphertext whose plaintext is related to the plaintexts of the input ciphertexts.
- **Distinguisher:** The attacker wants to output two distinct messages  $m_0$  and  $m_1$  such that if a challenger encrypts either  $m_0$  or  $m_1$  into c, the attacker would be able to tell which message was encrypted, just by looking at the challenge ciphertext c.

Clearly, if the encryption scheme is deterministic, there is always a trivial distinguisher: one could select any pair of distinct messages  $m_0$  and  $m_1$ , and by encrypting both  $m_0$  and  $m_1$ , one could tell which one corresponds to the challenge ciphertext. This implies that probabilistic encryption is necessary to satisfy strong security notions.

Attacks on encryption schemes are also classified based on the means available to the attacker:

**Chosen-plaintext attacks:** the attacker only knows the public key pk of the user, which implies that he may encrypt any plaintext of his choice.

- Valid-ciphertext attacks: the attacker can check whether a given ciphertext is valid, that is, that there exists a plaintext which may be encrypted into such a ciphertext. This makes sense when the set of ciphertexts is bigger than the set of plaintexts.
- **Plaintext-checking attacks:** the attacker can check whether a given ciphertext would be decrypted as a given plaintext.
- **Chosen-ciphertext attacks:** the attacker may ask for decryption of ciphertexts of its choice: if the ciphertext is not valid, the attacker will know. If the requests are not independent, the chosen-message attack is said to be *adaptive*. Of course, depending on the goal of the attacker, there is a natural restriction over the requests allowed: for instance, in a chosen-ciphertext distinguisher, the attacker cannot ask for the decryption of the challenge ciphertext.

#### 4. Elementary Attacks

The goal of this section is to illustrate the security notions described in Section 3 by presenting very simple attacks on textbook cryptosystems.

**4.1. Digital Signatures.** We first start with elementary attacks on textbook digital signatures.

4.1.1. Textbook-RSA. We first consider Textbook-RSA. Like any trapdoor permutation used directly as a signature scheme, Textbook-RSA is vulnerable to a no-message existential forgery. Indeed, anyone can select uniformly at random a number  $s \in \mathbb{Z}_N$ , and compute:

$$(4.1) m = s^e \bmod N.$$

Then s is a valid signature of the message  $m \in \mathbb{Z}_N$ . But this existential forgery is far from being a universal forgery, since there is very limited freedom over the choice of m.

However, in the particular case of Textbook-RSA, it is easy to obtain an adaptive chosen-message universal forgery, thanks to the multiplicativity of the RSA permutation. Indeed, assume that we would like to sign a message  $m \in \mathbb{Z}_N$ . Select  $m_1 \in \mathbb{Z}_N$  uniformly at random. If  $m_1$  is not invertible mod N (which is unlikely), then we have found a non-trivial factor of N, which allows us to sign m. Otherwise, we may compute:

$$m_2 = mm_1^{-1} \pmod{N}.$$

We ask the oracle the signatures  $s_1$  and  $s_2$  of respectively  $m_1$  and  $m_2$ . Then it is clear by multiplicativity that  $s = (s_1 s_2) \mod N$  is a valid signature of m.

A well-known countermeasure to avoid the previous attacks is to hash the message before signing it, that is, we assume the existence of a cryptographic hash function h from  $\{0,1\}^*$  to  $\mathbb{Z}_N$ . Instead of signing a message  $m \in \mathbb{Z}_N$ , we sign an arbitrary binary message  $m \in \{0,1\}^*$  and replace m by h(m) in both the signing process (2.5) and the verification process (2.6). The resulting RSA signature scheme is known as FDH-RSA for full-domain hash RSA [**BR96**], and it is provably secure in the random oracle model (roughly speaking, this assumes that the hash function is perfect: behaving like a random function), under the RSA assumption. To make sure that the hash function does not create obvious security failures, the hash function is required to be at least collision-free, that is, it should be "computationally hard" to output two distinct messages  $m_0$  and  $m_1$  such that  $h(m_0) = h(m_1)$ . In the case of Textbook-RSA, the use of a hash function prevented elementary forgeries and even provided a security proof in the random oracle model, but hash functions do not necessarily solve all the security problems by magic, as we will now see with Textbook-Elgamal.

4.1.2. Textbook-Elgamal. First, let us see an elementary existential forgery on Textbook-Elgamal. To forge a signature, it suffices to find a triplet  $(m, a, b) \in \mathbb{Z}_p \times \mathbb{Z}_p^{\times} \times \mathbb{Z}_{p-1}$  satisfying (2.13):

$$m \equiv ax + b \log a \pmod{p-1}.$$

Given an arbitrary m, the signer finds a valid pair (a, b) because he/she selects an a for which he/she already knows  $\log a$  (this logarithm is the one-time key k) and makes sure it is invertible modulo p - 1. Then because the signer knows the secret exponent x, he/she can solve (2.13) for b. But the attacker does not know the secret exponent x in (2.13), so he/she cannot do the same. One way to solve that problem would be to select a in such a way that ax cancels out with  $b \log a$ . For instance, if we select an a of the form:

$$a = g^B y^C \pmod{p},$$

where B and C are integers, then

$$ax + b \log a \equiv x(a + bC) + bB \pmod{p-1}$$

So if we select a C coprime with p-1, we can choose b such that:

 $a + bC \equiv 0 \pmod{p-1}$ .

Finally, we select the message m as:

$$m \equiv bB \pmod{p-1}$$
.

Our choice of (m, a, b) then satisfies (2.13). We thus have obtained a no-message existential forgery on Textbook-Elgamal. But this forgery, which was first described in [El 85], has almost no flexibility over m: we can obtain many forgeries thanks to different choices of (B, C), but each choice of (B, C) gives rise to a unique m. This means that this forgery will be prevented if we hash the message before hashing, like in FDH-RSA.

We now describe another existential forgery on Textbook-Elgamal, which can also be prevented by hashing. However, as opposed to the previous existential forgery, we will later see that this existential forgery can be transformed into a clever universal forgery found by Bleichenbacher [**Ble96**], which cannot therefore be prevented by hashing.

This alternative existential forgery finds a triplet  $(m, a, b) \in \mathbb{Z}_p \times \mathbb{Z}_p^{\times} \times \mathbb{Z}_{p-1}$ satisfying (2.13) by solving the congruence by Chinese remainders separately. Thus, we decompose the modulus p-1 as p-1 = qs where s is smooth (that is, it has no large prime factor, see [Sho05]). The reason why we choose s to be smooth is that it is easy to extract discrete logarithm in a group of smooth order, using Pohlig-Hellman's algorithm (see [MOV97, Sho05]). In particular, we do not know how to compute efficiently the discrete-log function log over  $\mathbb{Z}_p^{\times}$ , but for any  $z \in \mathbb{Z}_p^{\times}$ , we can efficiently compute (log z) mod s. We do not know the secret key x, but because we know the public key  $y = g^x \mod p$ , we may compute the smooth part x mod s. Since p-1 is always even, the smooth part s is at least 2. Because p - 1 = qs, the congruence (2.13) would imply the following two congruences:

- (4.2)  $m \equiv ax + b \log a \pmod{q}$
- (4.3)  $m \equiv ax + b \log a \pmod{s}$

Reciprocally, if we could find a triplet (m, a, b) satisfying both (4.2) and (4.3), would it necessarily satisfy (2.13)? The answer would be positive if q and s were coprime, by the Chinese remainder theorem. So let us assume that we put all the smooth part of p-1 into s, so that the smooth number s is indeed coprime with q = (p-1)/s.

We do not know  $x \mod q$ , so the mod q-congruence (4.2) looks hard to satisfy. However, note that the triplet (m, a, b) = (m, q, 0) is a trivial solution of (4.2) whenever  $m \equiv 0 \pmod{q}$ . So let us consider any message m such that  $m \equiv 0 \pmod{q}$ , and set a = q. It remains to satisfy the second congruence (4.3). We can compute  $\log a \mod s$ , and if we are lucky, it will be invertible mod s, so that we can solve (4.3). Thus, we have obtained a probabilistic existential forgery, which is weakly universal in the sense that if  $\log q$  is coprime with s, then we can forge the signature of any message m divisible by q. Like the previous existential forgery, this attack could easily be avoided using a cryptographic hash function, but Bleichenbacher [**Ble96**] found a trick to remove this limitation over m. We now describe Bleichenbacher's attack, with a presentation slightly different from that of [**Ble96**].

We restrict to the simplest form of Bleichenbacher's forgery, which requires that the generator g is smooth and divides p-1: a natural choice would be g = 2. Thus, we let s = g where p-1 = qs and we assume that s is smooth as before. However, we will no longer assume that q and s are coprime, so it will not suffice to work with (4.2) and (4.3) only. Instead, we will work with the congruence (2.13) mod p-1 directly. We can compute  $x_0 = x \mod s$ , so that  $x = x_0 + sx_1$  where  $x_1$ is unknown. If we let a = q, then (2.13) becomes:

(4.4) 
$$m \equiv ax_0 + b \log a \pmod{p-1}.$$

This congruence looks hard to solve for b since we know  $\log a \mod s$  but not mod p-1. The trick is that the particular choice a = q enables us to compute  $\log a$ . We claim that  $\log a = \log q$  is equal to the integer k = (p-3)/2 = (p-1)/2 - 1. To see this:

$$g^{k} \equiv g^{(p-1)/2}g^{-1} \pmod{p}$$
  

$$\equiv (-1)g^{-1} \text{ because } g \text{ is generator, so its Legendre symbol is -1.}$$
  

$$\equiv qsg^{-1} \text{ because } p-1 = qs.$$
  

$$\equiv q \text{ because } g = s.$$

It follows that (4.4) can be rewritten as:

(4.5) 
$$m \equiv ax_0 + bk \pmod{p-1}.$$

It is an elementary fact of number theory that this linear congruence can be solved for b if and only if gcd(k, p - 1) divides  $m - ax_0$ . To evaluate gcd(k, p - 1), note that:

$$k^{2} = ((p-1)/2 - 1)^{2} = ((p-1)/2)^{2} - (p-1) + 1 \equiv 1 + ((p-1)/2)^{2} \pmod{p-1}.$$

#### PHONG Q. NGUYEN

We distinguish two cases:

- If  $p \equiv 1 \pmod{4}$ , then gcd(k, p-1) = 1 because the previous congruence becomes  $k^2 \equiv 1 \pmod{p-1}$  as  $((p-1)/2)^2$  is a multiple of p-1. It follows that whatever the value of m, we can always solve (4.5) for b.
- Otherwise,  $p \equiv 3 \pmod{4}$ , and we claim that gcd(k, p-1) = 2. Indeed, this time, we have that  $((p-1)/2)^2 \equiv 1 \pmod{p-1}$  rather than 0, which implies that

$$k^2 \equiv 2 \pmod{p-1}.$$

It follows that gcd(k, p - 1) = 2 because we already know that it is  $\geq 2$ . Hence, if we assume that m is uniformly distributed modulo p - 1, then the probability that gcd(k, p - 1) divides  $m - ax_0$  is exactly 1/2. This means that we can solve (4.5) half of the time.

Hence, if the generator is smooth and divides p-1, we can either forge a signature on every message if  $p \equiv 1 \pmod{4}$ , or on half of the messages if  $p \equiv 3 \pmod{4}$ . Bleichenbacher describes other attacks on other specific generators in [**Ble96**].

Surprisingly, on the other hand, Pointcheval and Stern [**PS96**] showed at the same conference as [**Ble96**] that a slight modification of the Elgamal signature scheme is provably secure in the random oracle model. Furthermore, Bleichenbacher's attack applied to that modification as well, but there is fortunately no contradiction because the Pointcheval-Stern security proof assumed that the generator g was chosen uniformly at random among all generators of  $\mathbb{Z}_p^*$ , in which case it is very unlikely that g will be smooth and dividing p - 1. This suggests the following lesson: one should always carefully look at all the assumptions made by a security proof.

### 4.2. Asymmetric Encryption.

4.2.1. Textbook-RSA. We first consider Textbook-RSA. Like any trapdoor permutation used directly as a public-key encryption scheme, Textbook-RSA is vulnerable to brute-force attacks over the plaintext. More precisely, an attacker has access to a plaintext-checking oracle: the attacker can check whether a given ciphertext cwould be decrypted as a given plaintext m, by checking if:

$$(4.6) c \equiv m^e \bmod N.$$

In particular, if the set of plaintexts  $\mathcal{M}$  (where  $m \in \mathcal{M}$ ) is small, one can decrypt by brute-force: one would simply enumerate all  $m' \in \mathcal{M}$  and check whether the ciphertext c corresponds to the plaintext m', in which case m = m'. This would be for instance the case if we were encrypting English plaintexts letter by letter. In other words, when the distribution of plaintexts is very different from the uniform distribution over  $\mathbb{Z}_N$ , (such as when the set of plaintexts  $\mathcal{M}$  is a very small subset of  $\mathbb{Z}_N$ ), attacks may arise. Another famous example is the short-message attack. Assume that the plaintexts are in fact very small: for instance, assume that the plaintext m satisfies  $0 \leq m \leq N^{1/e}$ , (e.g. m is a 128-bit AES key, N a 1024-bit modulus, and e = 3). Then the integer m satisfies:  $0 \leq m^e \leq N$ , which means that the congruence (4.6) is in fact an equality over  $\mathbb{Z}$ ,

 $c = m^e$ .

But it is well-known that solving univariate polynomial equations over  $\mathbb{Z}$  can be done in polynomial time: extracting *e*-th roots over  $\mathbb{Z}$  is simply a particular case. In other words, if  $0 \le m \le N^{1/e}$ , then one can recover the plaintext *m* from (c, N, e)

16

in polynomial time. To summarize, if the distribution of the plaintext m is the uniform distribution over  $\mathbb{Z}_N$ , no one currently knows how to recover efficiently the plaintext m from its ciphertext  $c = m^e \mod N$ : this is exactly the RSA assumption. But if the distribution of the plaintext m is very different, there are examples for which there exist very efficient attacks.

Another elementary remark is that the RSA permutation provably leaks information. Given  $c = m^e \mod N$  where m has uniform distribution over  $\mathbb{Z}_N$ , one does not know how to recover m efficiently, but it is easy to recover efficiently one bit of information on the plaintext m. More precisely, because e must be odd (since it is coprime with  $\phi(N)$  which is even), the congruence (4.6) implies the following equality of Jacobi symbols:

$$\left(\frac{c}{N}\right) = \left(\frac{m}{N}\right)^e = \left(\frac{m}{N}\right).$$

In other words, one can derive efficiently the Jacobi symbol  $\binom{m}{N}$ , which provides one bit of information on the plaintext m.

We earlier saw an adaptive chosen-message universal forgery on Textbook-RSA signatures based on the multiplicativity of the RSA permutation. This elementary attack has an encryption analogue: it can be transformed into an adaptative chosenciphertext attack. Indeed, assume that we would like to decrypt a ciphertext  $c = m^e \mod N \in \mathbb{Z}_N$ : in other words, we would like to recover the plaintext  $m \in \mathbb{Z}_N$ . Select  $m_1 \in \mathbb{Z}_N$  uniformly at random. If  $m_1$  is not invertible mod N (which is unlikely), then we have found a non-trivial factor of N, which of course allows us to decrypt c. Otherwise, we may compute:

$$c_2 = cm_1^{-e} \pmod{N}.$$

We ask the decryption oracle to decrypt the ciphertext  $c_2$ : this gives the plaintext  $m_2 \in \mathbb{Z}_N$  defined by  $c_2 = m_2^e \mod N$ . Then it is clear by multiplicativity that  $m = (m_1 m_2) \mod N$ , which allows us to recover the initial plaintext m.

4.2.2. Textbook-Elgamal. Textbook-Elgamal is a probabilistic encryption scheme, unlike Textbook-RSA. In particular, there is no access to a plaintext-checking or-acle. However, Textbook-Elgamal provably leaks one bit of information on the plaintext, just like Textbook-RSA. Indeed, if g is a generator of  $\mathbb{Z}_p^*$ , then its Legendre symbol  $\left(\frac{g}{p}\right)$  must be equal to -1. In particular, the congruence (2.8) implies that the ciphertext (c, d) of a message m satisfies:

$$\left(\frac{c}{p}\right) = (-1)^k,$$

which discloses the parity of the one-time key k. Furthermore, the congruence (2.9) implies that:

$$\left(\frac{d}{p}\right) = \left(\frac{m}{p}\right) \left(\frac{y}{p}\right)^k.$$

Because d, y and p are public, and since the parity of k is now known, one can compute the Legendre symbol  $\left(\frac{m}{p}\right)$ , which discloses one bit of information on the plaintext m.

### PHONG Q. NGUYEN

We saw in Section 4.2.1 an adaptive chosen-ciphertext attack on Textbook-RSA encryption based on the multiplicativity of the RSA permutation. Since Textbook-Elgamal is multiplicative as well (see Section 2.2.2), this adaptive chosen-ciphertext attack can trivially be adapted to the Elgamal setting.

The fact that Textbook-RSA encryption is deterministic makes it vulnerable to several elementary attacks, but transforming it into a probabilistic encryption scheme will not prevent all the security problems by magic, as the example of Textbook-Elgamal encryption shows.

## 5. Square-Root Attacks

Whenever an exhaustive search over a secret key or a plaintext (or any other secret value) is possible, cryptographers often look for improved attacks based on time/memory trade-offs (see [MOV97, Hel80, Oec03, BBS06]). Usually, exhaustive search requires negligible memory M and exponential time T. A time/memory trade-off tries to balance those two costs. It is often achieved by splitting the secret value in values of half-size, in which case the new time and space complexity become roughly the square root of the cost of exhaustive search: that is, if T is the running time of exhaustive search, then both the time and space complexities become roughly  $\sqrt{T}$ . Sometimes, it is possible to further improve the space complexity of such square-root attacks to negligible memory, which is of considerable interest in practice. But among the three square-root attacks we will present, such a memory improvement is only known for the first one, which deals with the discrete logarithm problem.

5.1. The Discrete Logarithm Problem. As an illustration, consider the discrete logarithm problem used in Textbook-Elgamal. Let p be a prime and g be a generator of  $\mathbb{Z}_p^*$ . Assume that one is given an integer y satisfying:

$$(5.1) y = g^x \bmod p,$$

where the integer x is secret. The discrete logarithm problem asks to recover x modulo p - 1. Assume that the secret exponent x satisfies  $0 \le x \le X$ , where the public bound X is much smaller than p: does that make the discrete logarithm easier? Obviously, the simplest method would be to exhaustive search all exponents x such that  $0 \le x \le X$ , and find out which one satisfies (5.1). This costs X group operations with negligible space. A simple time/memory trade-off is obtained by splitting the secret exponent x in two parts. More precisely, the integer x can be written as:

$$x = x_1 + \lfloor \sqrt{X} \rfloor x_2$$

where  $x_1$  are  $x_2$  are two integers satisfying  $0 \le x_1 \le \lfloor \sqrt{X} \rfloor \le \sqrt{X}$  and  $0 \le x_2 \le X/\lfloor \sqrt{X} \rfloor = O(\sqrt{X})$ . This enables to rewrite (5.1) as:

$$y \equiv g^{x_1 + \lfloor \sqrt{X} \rfloor x_2} \pmod{p},$$

that is:

(5.2) 
$$y/g^{\lfloor \sqrt{X} \rfloor x_2} \equiv g^{x_1} \pmod{p}.$$

Reciprocally, any pair  $(x_1, x_2)$  satisfying (5.2) gives rise to a solution x of (5.1). This suggests the following time/memory trade-off:

- Precompute the list L of all  $g^{x_1} \mod p$  where  $0 \le x_1 \le |\sqrt{X}|$ , and sort the list L to allow binary search. This will cost essentially  $O(\sqrt{X} \ln X)$ polynomial-time operations.
- For all integers  $x_2$  such that  $0 \le x_2 \le X/\lfloor \sqrt{X} \rfloor$ , compute  $y/g^{\lfloor \sqrt{X} \rfloor x_2}$  mod p and find out if it belongs to the list L. If it belongs to L, output the corresponding solution x to (5.1). This will also cost essentially  $O(\sqrt{X} \ln X)$ polynomial-time operations.

In other words, we have obtained a time/memory trade-off to solve (5.2) (and therefore (5.1)), which has time and space complexity roughly  $O(\sqrt{X \ln X})$ , if we ignore polynomial costs. The method we have just described is known as the babystep/giant-step method in the literature (see [**MOV97**]). For the discrete logarithm problem, there are improvements to this basic square-root attack which allow to decrease the space requirement to negligible memory: see for instance Pollard's  $\rho$  and kangaroo methods in [CP01, MOV97], which are based on cycle-finding algorithms such as Floyd's.

5.2. RSA encryption of short messages. Another simple example of squareroot attacks is given by Textbook-RSA encryption of short messages with an arbitrary public exponent e, as explained in [BJN00]. Let  $0 \le m \le B$  be a plaintext encrypted as  $c = m^e \mod N$ . We assume that the plaintext is small, that is,  $B \ll N$ . For instance, m could be a 56-bit DES, N a 1024-bit RSA modulus, and  $e = 2^{16} + 1$ . It might happen that m can be split as  $m = m_1 m_2$  where  $m_1$  and  $m_2$  are between 0 and roughly  $\sqrt{B}$ . Splitting probabilities (as well as theoretical results) are listed in [BJN00]:

- For example, if 1 ≤ m ≤ 2<sup>64</sup> has uniform distribution then m can be split as a product m<sub>1</sub>m<sub>2</sub> where 1 ≤ m<sub>i</sub> < 2<sup>32</sup> with probability ≈ 0.18.
  Extending to 1 ≤ m<sub>i</sub> ≤ 2<sup>33</sup> increases the probability to ≈ 0.29, while extending to 1 ≤ m<sub>i</sub> ≤ 2<sup>34</sup> increases the probability to ≈ 0.35.

This suggests the following attack [BJN00]:

- Compute all the values  $m_1^e \mod N$  where  $1 \le m_1 \le A\sqrt{B}$  for some small constant A. These values (together with the corresponding  $m_1$ ) should be stored in a structure which is easily searched.
- For all values  $m_2$  such that  $1 \le m_2 \le A'\sqrt{B}$ , compute  $c/m_2^e \mod N$  and, for each value, see if this number appears in the earlier structure.
- If a match is found then we have  $c/m_2^e \equiv m_1^e \pmod{N}$  in which case  $c \equiv (m_1 m_2)^e \pmod{N}$  and therefore, the secret plaintext is  $m = m_1 m_2$ .

The cost of the attack is essentially  $O((A+A')\sqrt{B}\ln B)$  polynomial-time operations.

5.3. RSA with small CRT secret exponents. The square-root attacks we have described are very elementary, but sometimes, square-root attacks can be tricky. A less elementary example is given by Coppersmith's square-root attack on the discrete logarithm problem with sparse exponents: this is a particular case of the discrete logarithm problem when the secret exponent has low Hamming weight. The motivation is that such exponents allow faster exponentiation, and are therefore tempting for certain cryptographic schemes. For more details, Coppersmith's attack is described in [Sti02]: it was originally presented in the eighties as a remark on the message security of the Chor-Rivest public-key encryption scheme. Its time and

space complexities are roughly the square root of the running time of exhaustive search over all sparse exponents.

A more sophisticated square-root attack applies to RSA with small CRT secret exponent: the attack is vaguely described in [**QL00**] and is attributed to Richard Pinch. The motivation is the following. To speed up RSA decryption or signature generation, one could select a small secret exponent d. But we will see later (in Section 7.1.1) an attack (due to Wiener [**Wie90**]), which recovers the factorization of the RSA modulus N for usual parameters whenever  $d = O(N^{1/4})$ . And Wiener's attack was improved by Boneh and Durfee [**BD99**] to  $d = O(N^{1-1\sqrt{2}/2}) =$  $O(N^{0.292...})$  using lattice-based techniques which we will describe in Section 7.3. A better way to speed up RSA decryption or signature generation is to choose N = pqand e so that the integers  $d_p$  and  $d_q$  satisfying

$$ed_p \equiv 1 \pmod{p-1}$$
 and  $ed_q \equiv 1 \pmod{q-1}$ 

are small. If  $d_p$  and  $d_q$  are both O(B), there is a simple brute-force attack which costs O(B). Namely, assume without loss of generality that  $1 < d_p, d_q < B$  with  $d_p \neq d_q$ , and consider the following:

- Choose a random 1 < m < N and set  $c = m^e \mod N$ . Recall that  $c^{d_p} = m^{ed_p} \equiv m \mod p$ .
- For each 1 < i < B one can compute

$$gcd(c^i - m \mod N, N)$$

- and see if we have factored N.
- When  $i = d_p \neq d_q$  we have  $c^i \equiv m \mod p$  and  $c^i \not\equiv m \mod q$ . Hence the algorithm will succeed.
- The complexity is O(B).

It is natural to seek a square-root attack in this case. Consider what happens if one tries the obvious approach:

- Write  $M = |\sqrt{B}|$  and  $d_p = d_1 + Md_2$  with  $0 \le d_1 < M, 0 \le d_2 \le M + 1$ .
- One would expect to compute and store a table of 'baby steps'  $c^i \mod N$  for  $0 \le i < M$ .
- Then one would expect to compute the giant steps  $(c^M)^j \mod N$  for  $0 \le j \le M + 1$ .
- For each new giant step we must test whether there is a match, i.e., a value for *i* such that  $gcd(c^i(c^M)^j m, N) \neq 1$ .

The problem is that it seems the only way to check this is to run over the entire table of the baby steps and try each one. If this is done then the final complexity is still  $\tilde{O}(B)$  rather than the square root.

The following attack reaches the square-root goal:

• Compute the polynomial

$$G(x) = \prod_{j=0}^{M+1} ((c^M)^j x - m) \mod N.$$

- This computation takes time  $\tilde{O}(M)$  and storing G(x) requires space  $\tilde{O}(M)$ .
- Note that  $G(c^{d_1}) \equiv 0 \mod p$  since

$$(c^M)^{d_2} c^{d_1} \equiv c^d \equiv m \mod p$$

• Evaluate G(x) modulo N at  $c^i$  for all  $0 \le i < M$ .

• This gives a list of *M* numbers, one of which has a non-trivial gcd with *N*. One therefore factors *N*.

However, this method requires the evaluation of G(x) at M points. Since G(x) is a polynomial of degree M + 1, one might think that this is too expensive, that it would cost  $\tilde{O}(M^2)$ . Fortunately, there is an algorithm due to Strassen (see the textbook [**JvzGG03**]), which uses the Fast Fourier Transform (FFT) to evaluate a polynomial of degree M at M points in time  $\tilde{O}(M)$ . Using this algorithm, we eventually obtain a square-root complexity  $\tilde{O}(\sqrt{B})$  as announced. Recently, a lattice attack on this problem appeared in [**JM07**], based on techniques which we will describe in Section 7.3.

### 6. An Introduction to Lattices

**6.1. Background.** We will consider  $\mathbb{R}^n$  with its usual topology of an Euclidean vector space. We will use bold letters to denote vectors, usually in row notation. The Euclidean inner product of two vectors  $\mathbf{x} = (x_i)_{i=1}^n$  and  $\mathbf{y} = (y_i)_{i=1}^n$  is denoted by:

$$\langle \mathbf{x}, \mathbf{y} \rangle = \sum_{i=1}^{n} x_i y_i.$$

The corresponding Euclidean norm is denoted by:

$$\|\mathbf{x}\| = \sqrt{x_1^2 + \dots + x_n^2}$$

Denote by  $B(\mathbf{x}, r)$  the open ball of radius r centered at  $\mathbf{x}$ :

$$B(\mathbf{x}, r) = \{ \mathbf{y} \in \mathbb{R}^n : \|\mathbf{x} - \mathbf{y}\| < r \}.$$

A subset D of  $\mathbb{R}^n$  is called *discrete* when it has no limit point, that is: for all  $x \in D$ , there exists  $\rho > 0$  such that  $B(x, \rho) \cap D = \{x\}$ . As an example,  $\mathbb{Z}^n$ is discrete (because  $\rho = 1/2$  clearly works), while  $\mathbb{Q}^n$  and  $\mathbb{R}^n$  are not. The set  $\{1/n : n \in \mathbb{N}^*\}$  is discrete, but the set  $\{0\} \cup \{1/n : n \in \mathbb{N}^*\}$  is not. Any subset of a discrete set is discrete.

For any ring R, we denote by  $\mathcal{M}_{n,m}(R)$  (resp.  $\mathcal{M}_n(R)$ ) the set of  $n \times m$  (resp.  $n \times n$ ) matrices with coefficients in R.  $GL_n(R)$  denotes the group of invertible matrices in the ring  $\mathcal{M}_n(R)$ .

For any subset S of  $\mathbb{R}^n$ , we define the linear span of S, denoted by span(S), as the minimal vector subspace (of  $\mathbb{R}^n$ ) containing S.

Let  $\mathbf{b}_1, \ldots, \mathbf{b}_m$  be in  $\mathbb{R}^n$ . The vectors  $\mathbf{b}_i$ 's are said to be *linearly dependent* if there exist  $x_1, \ldots, x_m \in \mathbb{R}$  which are not all zero and such that:

$$\sum_{i=1}^{m} x_i \mathbf{b}_i = 0$$

Otherwise, they are said to be *linearly independent*.

The Gram determinant of  $\mathbf{b}_1, \ldots, \mathbf{b}_m \in \mathbb{R}^n$ , denoted by  $\Delta(\mathbf{b}_1, \ldots, \mathbf{b}_m)$ , is by definition the determinant of the Gram matrix  $(\langle \mathbf{b}_i, \mathbf{b}_j \rangle)_{1 \leq i,j \leq m}$ . This real number  $\Delta(\mathbf{b}_1, \ldots, \mathbf{b}_m)$  is always  $\geq 0$ , and it turns out to be zero if and only if the  $\mathbf{b}_i$ 's are linearly dependent. The Gram determinant is invariant by any permutation of the *m* vectors, and by any integral linear transformation of determinant  $\pm 1$  such as adding to one of the vectors a linear combination of the others. The Gram determinant has a very useful geometric interpretation: when the  $\mathbf{b}_i$ 's are linearly

independent,  $\sqrt{\Delta(\mathbf{b}_1, \ldots, \mathbf{b}_m)}$  is the *m*-dimensional volume of the parallelepiped spanned by the  $\mathbf{b}_i$ 's.

**6.2.** Lattices. We call *lattice* of  $\mathbb{R}^n$  any discrete subgroup of  $(\mathbb{R}^n, +)$ ; that is any subgroup of  $(\mathbb{R}^n, +)$  which has the discreteness property. Notice that an additive group is discrete if and only if 0 is not a limit point, which implies that a lattice is any non-empty set  $L \subseteq \mathbb{R}^n$  stable by subtraction (in other words: for all **x** and **y** in L,  $\mathbf{x} - \mathbf{y}$  belongs to L), and such that  $L \cap B(0, \rho) = \{0\}$  for some  $\rho > 0$ .

With this definition, the first examples of lattices which come to mind are the zero lattice  $\{0\}$  and the *lattice of integers*  $\mathbb{Z}^n$ . Our definition implies that any subgroup of a lattice is a lattice, and therefore, any subgroup of  $(\mathbb{Z}^n, +)$  is a lattice. Such lattices are called *integral lattices*. As an example, consider two integers a and  $b \in \mathbb{Z}$ : the set  $a\mathbb{Z} + b\mathbb{Z}$  of all integral linear combinations of a and b is a subgroup of  $\mathbb{Z}$ , and therefore a lattice; it is actually the set  $gcd(a, b)\mathbb{Z}$  of all multiples of the gcd of a and b. For another example, consider n integers  $a_1, \ldots, a_n$ , together with a modulus M. Then the set of all  $(x_1, \ldots, x_n) \in \mathbb{Z}^n$  such that  $\sum_{i=1}^n a_i x_i \equiv 0 \pmod{M}$  is a lattice in  $\mathbb{Z}^n$  because it is clearly a subgroup of  $\mathbb{Z}^n$ .

We give a few basic properties of lattices:

PROPOSITION 6.1. Let L be a lattice in  $\mathbb{R}^n$ .

(1) There exists  $\rho > 0$  such that for all  $\mathbf{x} \in L$ :

$$L \cap B(\mathbf{x}, \rho) = \{\mathbf{x}\}\$$

- (2) L is closed.
- (3) For all bounded subsets S of  $\mathbb{R}^n$ ,  $L \cap S$  is finite.
- (4) L is countable.

PROOF. We know that  $L \cap B(0, \rho) = \{0\}$  for some  $\rho > 0$ . Since L is an additive group, we obtain property 1. It follows that any convergent sequence of L is stationary, which proves property 2. If S is a bounded subset, it must be included in some closed ball B. The set  $L \cap B$  is closed and bounded, thus compact. Since it is also discrete, it must be finite (by the Borel-Lebesgue theorem), which gives property 3. Since  $\mathbb{R}^n$  is the union of all B(0, r) for  $r \in \mathbb{N}$ , we obtain property 4.  $\Box$ 

Notice that a set which satisfies either property 1 or 3 is necessarily discrete, but an arbitrary discrete subset of  $\mathbb{R}^n$  does not necessarily satisfy property 1 nor 3. It is the group structure of lattices which allows such additional properties.

**6.3. Lattice Bases.** Let  $\mathbf{b}_1, \ldots, \mathbf{b}_m$  be arbitrary vectors in  $\mathbb{R}^n$ . Denote by  $L(\mathbf{b}_1, \ldots, \mathbf{b}_m)$  the set of all integral linear combinations of the  $\mathbf{b}_i$ 's:

$$L(\mathbf{b}_1,\ldots,\mathbf{b}_m) = \left\{\sum_{i=1}^m n_i \mathbf{b}_i : n_1,\ldots,n_m \in \mathbb{Z}\right\}$$

This set is a subgroup of  $\mathbb{R}^n$ , but it is not necessarily discrete. For instance, one can show that  $L((1), (\sqrt{2}))$  is not discrete because  $\sqrt{2} \notin \mathbb{Q}$ . However, notice that if the  $\mathbf{b}_i$ 's are in  $\mathbb{Q}^n$ , then  $L(\mathbf{b}_1, \ldots, \mathbf{b}_m)$  is discrete, and so is a lattice. When  $L = L(\mathbf{b}_1, \ldots, \mathbf{b}_m)$  is a lattice, we say that L is spanned by the  $\mathbf{b}_i$ 's, and that the  $\mathbf{b}_i$ 's are generators. When the  $\mathbf{b}_i$ 's are further linearly independent, we say that  $(\mathbf{b}_1, \ldots, \mathbf{b}_m)$  is a basis of the lattice L, in which case each lattice vector decomposes itself uniquely as an integral linear combination of the  $\mathbf{b}_i$ 's. Bases and sets of generators are useful to represent lattices, and to perform computations. One will

typically represent a lattice on a computer by some lattice basis, which can itself be represented by a matrix with real coefficients. In practice, one will usually restrict to integral lattices, so that the underlying matrices are integral matrices.

We define the dimension or rank of a lattice L, denoted by dim(L), as the dimension d of its linear span denoted by span(L). The dimension is the maximal number of linearly independent lattice vectors. Any lattice basis of L must have exactly d elements. There always exist d linearly independent lattice vectors, however such vectors do not necessarily form a basis, as opposed to the case of vectors spaces. But the following theorem shows that one can always derive a lattice basis from such vectors:

THEOREM 6.2. Let L be a lattice of  $\mathbb{R}^n$ , with dimension d. Let  $\mathbf{c}_1, \ldots, \mathbf{c}_d$  be linearly independent vectors of L. There exists a lower triangular matrix  $(u_{i,j}) \in \mathcal{M}_d(\mathbb{R})$  such that the vectors  $\mathbf{b}_1, \ldots, \mathbf{b}_d$  defined as  $\mathbf{b}_i = \sum_{j=1}^i u_{i,j} \mathbf{c}_j$  form a basis of L.

PROOF. We reproduce the proof of [Sie89, Theorem 18, p. 45]. Let  $1 \le i \le d$ . Consider the following set:

$$S_i = \left\{ x_i \in ]0,1] : \exists x_1, \dots, x_{i-1} \in \mathbb{R} \text{ such that } \sum_{j=1}^i x_j \mathbf{c}_j \in L \right\}.$$

This set is actually finite because  $x_i \in S_i$  implies that  $x_i \mathbf{c}_i + \sum_{j=1}^{i-1} (x_j - \lfloor x_j \rfloor) \mathbf{c}_j$ belongs to  $L \cap B(0, \sum_{j=1}^{i} \|\mathbf{c}_j\|)$  which is finite. And  $S_i$  is not empty since it contains 1, therefore it has a smallest element which is strictly positive, and which we denote by  $u_{i,i} > 0$ . By definition, there exist  $u_{i,1}, \ldots, u_{i,i-1} \in \mathbb{R}$  such that  $\mathbf{b}_i = \sum_{j=1}^{i} u_{i,j} \mathbf{c}_j \in L$ .

It remains to prove that the  $\mathbf{b}_i$ 's form a basis. Since  $u_{i,i} > 0$ , the  $\mathbf{b}_i$ 's are linearly independent. Now, let  $\mathbf{y} \in L$ . Since the  $\mathbf{b}_i$ 's are linearly independent, there exist  $y_1, \ldots, y_n \in \mathbb{R}$  such that  $\mathbf{y} = \sum_{i=1}^d y_i \mathbf{b}_i$ . Define  $\mathbf{x} = \sum_{i=1}^d x_i \mathbf{b}_i$  where  $x_i = y_i - \lfloor y_i \rfloor$ . We have  $\mathbf{x} \in L$  and  $0 \leq x_i < 1$ . Suppose *ad absurdum* that not all the  $y_i$ 's are integral: let k be the largest index such that  $y_k \notin \mathbb{Z}$ . Then  $x_k > 0$  and  $x_i = 0$  if i > k. Thus:

$$\mathbf{x} = u_{k,k} x_k \mathbf{c}_k + \sum_{j=1}^{k-1} u_{k,j} x_k \mathbf{c}_j + \sum_{i=1}^{k-1} x_i \sum_{j=1}^i u_{i,j} \mathbf{c}_j.$$

Since  $0 < x_k < 1$ ,  $0 < u_{k,k}x_k < u_{k,k}$  which contradicts the fact that  $u_{k,k}$  is the smallest element of  $S_k$ .

This gives the unconditional existence of lattice bases:

COROLLARY 6.3. Any lattice of  $\mathbb{R}^n$  has at least one basis.

Thus, even if sets of the form  $L(\mathbf{b}_1, \ldots, \mathbf{b}_m)$  may or may not be lattices, all lattices can be written as  $L(\mathbf{b}_1, \ldots, \mathbf{b}_m)$  for some linearly independent  $\mathbf{b}_i$ 's. The converse is easy to prove:

THEOREM 6.4. Let  $\mathbf{b}_1, \ldots, \mathbf{b}_d \in \mathbb{R}^n$  be linearly independent. Then the set  $L(\mathbf{b}_1, \ldots, \mathbf{b}_d)$  is a lattice of dimension d.

PROOF. Let  $L = L(\mathbf{b}_1, \dots, \mathbf{b}_d)$ . It suffices to show that 0 is not a limit point of L. Consider the parallelepiped P defined by:

$$P = \left\{ \sum_{i=1}^d x_i \mathbf{b}_i : |x_i| < 1 \right\}.$$

Since the  $\mathbf{b}_i$ 's are linearly independent,  $L \cap P = \{0\}$ . Besides, there exists  $\rho > 0$  such that  $B(0, \rho) \subseteq P$ , which shows that 0 cannot be a limit point of L.  $\Box$ 

Corollary 6.3 together with Theorem 6.4 give an alternative definition of a lattice: a non-empty subset L of  $\mathbb{R}^n$  is a lattice if only if there exist linearly independent vectors  $\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_d$  in  $\mathbb{R}^n$  such that:

$$L = L(\mathbf{b}_1, \ldots, \mathbf{b}_d).$$

This characterization suggests that lattices are discrete analogues of vector spaces.

Lattice bases are characterized by the following elementary result, whose proof is omitted:

THEOREM 6.5. Let  $(\mathbf{b}_1, \ldots, \mathbf{b}_d)$  be a basis of a lattice L in  $\mathbb{R}^n$ . Let  $\mathbf{c}_1, \ldots, \mathbf{c}_d$ be vectors of L: there exists a  $d \times d$  integral matrix  $U = (u_{i,j})_{1 \leq i,j \leq d} \in \mathcal{M}_d(\mathbb{Z})$  such that  $\mathbf{c}_i = \sum_{j=1}^d u_{i,j} \mathbf{b}_j$  for all  $1 \leq i \leq d$ . Then  $(\mathbf{c}_1, \ldots, \mathbf{c}_d)$  is a basis of L if and only if the matrix U has determinant  $\pm 1$ .

As a result, as soon as the lattice dimension is  $\geq 2$ , there are infinitely many lattice bases.

**6.4. Lattice Volume.** Let  $(\mathbf{b}_1, \ldots, \mathbf{b}_d)$  and  $(\mathbf{c}_1, \ldots, \mathbf{c}_d)$  be two bases of a lattice L in  $\mathbb{R}^n$ . By Theorem 6.5, there exists a  $d \times d$  integral matrix  $U = (u_{i,j})_{1 \le i,j \le d} \in \mathcal{M}_d(\mathbb{Z})$  of determinant  $\pm 1$  such that  $\mathbf{c}_i = \sum_{j=1}^d u_{i,j} \mathbf{b}_j$  for all  $1 \le i \le d$ . It follows that the Gram determinant of those two bases are equal:

$$\Delta(\mathbf{b}_1,\ldots,\mathbf{b}_d)=\Delta(\mathbf{c}_1,\ldots,\mathbf{c}_d)>0.$$

The volume (or determinant) of the lattice L is defined as:

$$\operatorname{vol}(L) = \Delta(\mathbf{b}_1, \dots, \mathbf{b}_d)^{1/2},$$

which is independent of the choice of lattice basis  $(\mathbf{b}_1, \ldots, \mathbf{b}_d)$ . We prefer the name *volume* to the name *determinant* because of its geometric interpretation: it corresponds to the *d*-dimensional volume of the parallelepiped spanned by any basis. In the mathematical literature, the lattice volume we have just defined is sometimes alternatively called co-volume, because it is also the volume of the torus  $\operatorname{span}(L)/L$ . In the important case of full-dimensional lattices where  $\dim(L) = n = \dim(\mathbb{R}^n)$ , the volume is equal to the absolute value of the determinant of any lattice basis (hence the alternative name determinant).

Given a lattice L, how does one compute the volume of L? If an explicit basis of L is known, this amounts to computing a determinant: for instance, the volume of the hypercubic lattice  $\mathbb{Z}^n$  is clearly equal to one. But if no explicit basis is known, there is sometimes another way, due to the following elementary result: if  $L_1$  and  $L_2$  are two lattices of  $\mathbb{R}^n$  with the same dimension such that  $L_1 \subseteq L_2$ , then  $L_2/L_1$  is a finite group of order denoted by  $[L_2:L_1]$  which satisfies

$$\operatorname{vol}(L_1) = \operatorname{vol}(L_2) \times [L_2 : L_1].$$

24

As an illustration, consider n integers  $a_1, \ldots, a_n$ , together with a modulus M. We have seen in Section 6.2 that the set L of all  $(x_1, \ldots, x_n) \in \mathbb{Z}^n$  such that  $\sum_{i=1}^n a_i x_i \equiv 0 \pmod{M}$  is a lattice in  $\mathbb{Z}^n$  because it is a subgroup of  $\mathbb{Z}^n$ . But there seems to be no trivial basis of L. However, note that  $L \subseteq \mathbb{Z}^n$  and that the dimension of L is n because L contains all the vectors of the canonical basis of  $\mathbb{R}^n$ multiplied by M. It follows that:

$$\operatorname{vol}(L) = [\mathbb{Z}^n : L].$$

Furthermore, the definition of L clearly implies that:

$$[\mathbb{Z}^n:L] = M/\operatorname{gcd}(M, a_1, a_2, \dots, a_n).$$

Hence:

$$\operatorname{vol}(L) = \frac{M}{\gcd(M, a_1, a_2, \dots, a_n)}.$$

**6.5.** Lattice Reduction. A fundamental result of linear algebra states that any finite-dimensional vector space has a basis. We earlier established the analogue result for lattices: any lattice has a basis. In the same vein, a fundamental result of bilinear algebra states that any finite-dimensional Euclidean space has an orthonormal basis, that is, a basis consisting of unit vectors which are pairwise orthogonal. A natural question is to ask whether lattices also have orthonormal bases, or at least, orthogonal bases. Unfortunately, it is not difficult to see that even in dimension two, a lattice may not have an orthogonal basis. Informally, the goal of lattice reduction is to circumvent this problem: more precisely, the theory of lattice reduction shows that in any lattice, there is always a basis which is not that far from being orthogonal. Defining precisely what is meant exactly by not being far from being orthogonal is tricky, so for now, let us just say that such a basis should consist of reasonably short lattice vectors, which implies that geometrically, such vectors are not far from being orthogonal to each other.

6.5.1. Minkowski's successive minima. In order to explain what is a reduced basis, we need to define what is meant by short lattice vectors. Let L be a lattice of dimension  $\geq 1$  in  $\mathbb{R}^n$ . There exists a non-zero vector  $\mathbf{u} \in L$ . Consider the closed hyperball B of radius  $\|\mathbf{u}\|$ , centered at zero. Then  $L \cap B$  is finite and contains  $\mathbf{u}$ , so it must have a shortest non-zero vector. The Euclidean norm of that shortest non-zero vector is called the first minimum of L, and is denoted by  $\lambda_1(L) > 0$  or  $\|L\|$ . By definition, any non-zero vector  $\mathbf{v}$  of L satisfies:  $\|\mathbf{v}\| \geq \lambda_1(L)$ . And there exists  $\mathbf{w} \in L$  such that  $\|\mathbf{w}\| = \lambda_1(L)$ : any such  $\mathbf{w}$  is called a shortest vector of L, and it is not unique since  $-\mathbf{w}$  would also be a shortest vector. The kissing number of L is the number of shortest vectors in L: it is upper bounded by some exponential function of the lattice dimension (see [CS98]).

If **w** is a shortest vector of L, then so is  $-\mathbf{w}$ . Thus, one must be careful when defining the *second-to-shortest* vector of a lattice. To circumvent this problem, Minkowski [**Min96**] defined the other minima as follows. For all  $1 \le i \le \dim(L)$ , the *i*-th minimum  $\lambda_i(L)$  is defined as the minimum of  $\max_{1\le j\le i} \|\mathbf{v}_j\|$  over all *i* linearly independent lattice vectors  $\mathbf{v}_1, \ldots, \mathbf{v}_i \in L$ . Clearly, the minima are increasing:  $\lambda_1(L) \le \lambda_2(L) \le \cdots \le \lambda_d(L)$ . And it is not difficult to see that there always exist linearly independent lattice vectors  $\mathbf{v}_1, \ldots, \mathbf{v}_d$  reaching simultaneously the minima, that is  $\|\mathbf{v}_i\| = \lambda_i(L)$  for all *i*. However, surprisingly, as soon as  $\dim(L) \ge 4$ , such vectors do not necessarily form a lattice basis. The canonical example is the 4-dimensional lattice L defined as the set of all  $(x_1, x_2, x_3, x_4) \in \mathbb{Z}^4$  such that  $\sum_{i=1}^{4} x_i$  is even. It is not difficult to see that  $\dim(L) = 4$  and that all the minima of L are equal to  $\sqrt{2}$ . Furthermore, it can be checked that the following row vectors form a basis of L:

$$\begin{pmatrix} 1 & -1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{pmatrix}.$$

The basis proves in particular that vol(L) = 2. However, the following row vectors are linearly independent lattice vectors which also reach all the minima:

$$\begin{pmatrix} 1 & -1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & -1 \end{pmatrix}.$$

But they do not form a basis, since their determinant is equal to 4: another reason is that for all such vectors, the sum of the first two coordinates is even, and that property also holds for any integral linear combination of those vectors, but clearly not for all vectors of the lattice L. More precisely, the sublattice spanned by those four row vectors has index two in the lattice L.

Nevertheless, in the lattice L, there still exists at least one basis which reaches all the minima simultaneously, and we already gave one such basis. This also holds for any lattice of dimension  $\leq 4$ , but it is no longer true in dimension  $\geq 5$ , as was first noticed by Korkine and Zolotarev in the 19th century, in the language of quadratic forms. More precisely, it can easily be checked that the lattice spanned by the rows of the following matrix

$$\begin{pmatrix} 2 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

has no basis reaching all the minima (which are all equal to two).

6.5.2. Hermite's constant and Minkowski's theorems. Now that successive minima have been defined, it is natural to ask how large those minima can be. Hermite [Her50] was the first to prove that the quantity  $\lambda_1(L)/\operatorname{vol}(L)^{1/d}$  could be upper bounded over all *d*-rank lattices *L*. The supremum of  $\lambda_1(L)^2/\operatorname{vol}(L)^{2/d}$  over all *d*-rank lattices *L* is denoted by  $\gamma_d$ , and called Hermite's constant of dimension *d*, because Hermite was the first to establish its existence in the language of quadratic forms. The use of quadratic forms explains why Hermite's constant refers to  $\max_L \lambda_1(L)^2/\operatorname{vol}(L)^{2/d}$  and not to  $\max_L \lambda_1(L)/\operatorname{vol}(L)^{1/d}$ . Clearly,  $\gamma_d$  could also be defined as the supremum of  $\lambda_1(L)^2$  over all *d*-rank lattices *L* of unit volume.

be defined as the supremum of  $\lambda_1(L)^2$  over all *d*-rank lattices *L* of unit volume. It is known that  $\gamma_d$  is reached, that is: for all  $d \ge 1$ , there is a *d*-rank lattice *L* such that  $\gamma_d = \lambda_1(L)^2/\operatorname{vol}(L)^{2/d}$ , and any such lattice is called *critical*. But finding the exact value of  $\gamma_d$  is a very difficult problem, which has been central in Minkowski's geometry of numbers. The exact value of  $\gamma_d$  is known only for  $1 \le d \le 8$  (see the book [**Mar03**] for proofs) and very recently also for d = 24 (see [**CK04**]): the values are summarized in the following table.

d	2	3	4	5	6	7	8	24
$\gamma_d$	$2/\sqrt{3}$	$2^{1/3}$	$\sqrt{2}$	$8^{1/5}$	$(64/3)^{1/6}$	$64^{1/7}$	2	4
Approximation	1.1547	1.2599	1.4142	1.5157	1.6654	1.8114	2	4

Furthermore, the list of all critical lattices (up to scaling and isometry) is known for each of those dimensions.

However, rather tight asymptotical bounds are known for Hermite's constant. More precisely, we have:

$$\frac{d}{2\pi e} + \frac{\log(\pi d)}{2\pi e} + o(1) \le \gamma_d \le \frac{1.744d}{2\pi e} (1 + o(1)).$$

For more information on the proof of those bounds: see [MH73, Chapter II] for the lower bound (which comes from the Minkowski-Hlawka theorem), and [CS98, Chapter 9] for the upper bound. Thus,  $\gamma_d$  is essentially linear in d. It is known that  $\gamma_d^d \in \mathbb{Q}$  (because there is always an integral critical lattice), but it is unknown if  $\gamma_d$  is an increasing sequence.

Hermite's historical upper bound [Her50] on his constant was exponential in the dimension:

$$\gamma_d < (4/3)^{(d-1)/2}.$$

The first linear upper bound on Hermite's constant is due to Minkowski, who viewed it as a consequence of his Convex Body Theorem:

THEOREM 6.6 (Minkowski's Convex Body Theorem). Let L be a full-rank lattice of  $\mathbb{R}^n$ . Let C be a measurable subset of  $\mathbb{R}^n$ , convex, symmetric with respect to 0, and of measure >  $2^n \operatorname{vol}(L)$ . Then C contains at least a non-zero point of L.

This theorem is a direct application of the following elementary lemma (see [Sie89]), which can be viewed as a generalization of the pigeon-hole principle:

LEMMA 6.7 (Blichfeldt). Let L be a full-rank lattice in  $\mathbb{R}^n$ , and F be a measurable subset of  $\mathbb{R}^n$  with measure  $> \operatorname{vol}(L)$ . Then F contains at least two distinct vectors whose difference is in L.

Indeed, we may consider  $F = \frac{1}{2}C$ , and the assumption in Theorem 6.6. implies that the measure of F is  $> \operatorname{vol}(L)$ . From Blichfeldt's lemma, it follows that there exist  $\mathbf{x}$  and  $\mathbf{y}$  in F such that  $\mathbf{x} - \mathbf{y} \in L \setminus \{0\}$ . But

$$\mathbf{x} - \mathbf{y} = \frac{1}{2}(2\mathbf{x} - 2\mathbf{y})$$

which belongs to C by convexity, and symmetry with respect to 0. Hence:  $\mathbf{x} - \mathbf{y} \in C \cap (L \setminus \{0\})$ , which completes the proof of Theorem 6.6.

One notices that the bound on the volumes in Theorem 6.6 is the best possible, by considering

$$C = \left\{ \sum_{i=1}^n x_i \mathbf{b}_i : |x_i| < 1 \right\},\$$

where the  $\mathbf{b}_i$ 's form an arbitrary basis of the lattice. Indeed, in this case, the measure of C is exactly  $2^n \operatorname{vol}(L)$ , but by definition of C, no non-zero vector of L belongs to C.

In Theorem 6.6, the condition on the measure of C is a strict inequality, but it is not difficult to show that the strict inequality can be relaxed to an inequality  $\geq 2^n \operatorname{vol}(L)$  if C is further assumed to be compact. By choosing for C a closed hyperball of sufficiently large radius (so that the volume inequality is satisfied), one obtains that any d-dimensional lattice L of  $\mathbb{R}^n$  contains a non-zero  $\mathbf{x}$  such that

$$\|\mathbf{x}\| \le 2\left(\frac{\operatorname{vol}(L)}{v_d}\right)^{\frac{1}{d}},$$

where  $v_d$  denotes the volume of the closed unitary hyperball of  $\mathbb{R}^d$ . Using well-known formulas for  $v_d$ , one can derive a linear bound on Hermite's constant, for instance:

$$\forall d, \ \gamma_d \le 1 + \frac{d}{4}.$$

One can obtain an analogous result for the max-norm:

THEOREM 6.8. Let L be a d-dimensional lattice. Then there exists a non-zero  $\mathbf{x}$  in L such that:

$$\|\mathbf{x}\|_{\infty} \le \operatorname{vol}(L)^{1/d}.$$

Notice that this bound is reached by  $L = \mathbb{Z}^d$ .

Now that we know how to bound the first minimum, it is natural to ask if a similar bound can be obtained for the other minima. Unfortunately, one cannot hope to upper bound separately the other minima, because the successive minima could be unbalanced. For instance, consider the rectangular 2-rank lattice L spanned by the following row matrix:

$$\begin{pmatrix} \varepsilon & 0 \\ 0 & 1/\varepsilon \end{pmatrix},$$

where  $\varepsilon > 0$  is small. The volume of L is one, and by definition of L, it is clear that  $\lambda_1(L) = \varepsilon$  and  $\lambda_2(L) = 1/\varepsilon$  if  $\varepsilon \leq 1$ . Here,  $\lambda_2(L)$  can be arbitrarily large compared to the lattice volume, while  $\lambda_1(L)$  can be arbitrarily small compared to the upper bound given by Hermite's constant.

However, it is always possible to upper bound the geometric mean of the first consecutive minima, as summarized by the following theorem (for an elementary proof, see [Sie89, MG02]):

THEOREM 6.9 (Minkowski's Second Theorem). Let L be a d-rank lattice of  $\mathbb{R}^n$ . Then for any integer r such that  $1 \leq r \leq d$ :

$$\left(\prod_{i=1}^r \lambda_i(L)\right)^{1/r} \le \sqrt{\gamma_d} \operatorname{vol}(L)^{1/d}.$$

6.5.3. Random Lattices. The upper bound on the first minimum derived from Hermite's constant is only tight for critical lattices, which are very special lattices. One might wonder what happens for more general lattices, say random lattices. But what is a random lattice? Surprisingly, from a mathematical point of view, there is a natural (albeit technically involved) notion of random lattice, which follows from a measure on full-rank lattices with determinant 1 introduced by Siegel [Sie45] back in 1945, to provide an alternative proof of the Minkowski-Hlawka theorem; this measure is derived from Haar measures of classical groups. In these lecture notes, no formal definition of random lattices will be needed: we refer the interested reader to the recent articles [Ajt02, GM03] which propose efficient ways to generate lattices which are provably random in this sense: see also [NS06] for practical considerations. We now list a few important properties of random lattices, to give more intuition on random lattices. We saw in Section 6.5.2 that an *n*-rank lattice L satisfies:

(6.1) 
$$\lambda_1(L) \le \sqrt{\gamma_n} \operatorname{vol}(L)^{1/n} \le \sqrt{1 + n/4} \operatorname{vol}(L)^{1/n}.$$

28

Interestingly, a random *n*-rank lattice L satisfies asymptotically with overwhelming probability (see [Ajt06] for a proof):

$$\forall 1 \le i \le n, \lambda_i(L) \approx \sqrt{\frac{n}{2\pi e}} \operatorname{vol}(L)^{1/n}.$$

In particular, the bound on the first minimum derived from Hermite's constant is not that far from being tight in the random case: the ratio between the two upper bounds is bounded independently of the dimension. Thus, even though it is easy to construct lattices for which the first minimum is arbitrarily small compared to Hermite's bound, such lattices are far from being random: the first minimum of random lattices is almost as large as the one of critical lattices. Furthermore, [Ajt06] also shows that asymptotically, in a random *n*-rank lattice *L*, there exists with overwhelming probability a lattice basis  $(\mathbf{b}_1, \ldots, \mathbf{b}_n)$  such that:

$$\forall \ 1 \le i \le n, \|\mathbf{b}_i\| \approx \sqrt{\frac{n}{2\pi e}} \operatorname{vol}(L)^{1/n}.$$

Such a basis consists of very short vectors, since their norms are close to the successive minima. Thus, there are always nice bases in random lattices.

The previous properties are useful to distinguish specific lattices from random lattices. For instance, in cryptography, one often encounters lattices for which the first minimum is provably much smaller than Hermite's bound (6.1), so such lattices cannot be random, and they might have exceptional properties which can be exploited. And when a lattice is very far from being random, certain computational problems which are hard in the general case may become easy.

6.5.4. Reduction Notions. In Section 6.5.3, we saw that random lattices always have nice bases: naturally, one might wonder what happens in the general case. The goal of lattice (basis) reduction is to prove the existence of nice lattice bases in every lattice, and not just random lattices. Such nice bases are called *reduced*, but it is important to stress that there are many notions of reduction. Usually, one first defines a notion of reduction, then shows that there exist bases which are reduced in this sense, and finally proves that bases which are reduced in this sense have interesting properties. In terms of interesting properties, we are interested in both mathematical and computational properties: does it have nice mathematical properties, and is it easy to compute such reduced bases? Computational aspects will only be discussed in the next subsection.

In low dimension  $\leq 4$ , there is one notion of reduction which is arguably better than all the others: the so-called Minkowski reduction. Minkowski defined a natural notion of reduction, for which it is easy to prove that there are Minkowski-reduced bases in all lattices. And Minkowski proved that when the lattice dimension d is  $\leq 4$ , a Minkowski-reduced basis  $(\mathbf{b}_1, \ldots, \mathbf{b}_d)$  must satisfy:  $\forall i, \|\mathbf{b}_i\| = \lambda_i(L)$ , which is arguably the best one can hope for a basis. Furthermore, there is a very natural algorithm to compute such bases, and which is very efficient up to dimension  $\leq 4$ (see [**NS04**]). However, when the dimension is  $\geq 5$ , among all the reduction notions which are known, none is clearly better than the others. But the best notions of reduction known all provide guarantees on the norm of the basis vectors  $\mathbf{b}_1, \ldots, \mathbf{b}_d$ , such as upper bounds on the d ratios  $\|\mathbf{b}_i\|/\lambda_i(L)$ .

Enumerating all the reduction notions known is beyond the scope of these notes. In fact, we will not even define precisely any reduction notion. Instead, we will only present properties of two important notions of reduction (see [**MG02**]): the Lenstra-Lenstra-Lovász reduction [LLL82] (called LLL for short), and the Hermite-Korkine-Zolotarev reduction [KZ73] (called HKZ for short). The HKZ reduction is a very strong notion of reduction which is computationally expensive, while the LLL reduction is a weaker notion of reduction which is computationally inexpensive. An HKZ-reduced basis  $(\mathbf{b}_1, \ldots, \mathbf{b}_d)$  of a lattice L satisfies for all  $1 \le i \le d$ :

$$\frac{4}{i+3} \le \left(\frac{\|\mathbf{b}_i\|}{\lambda_i(L)}\right)^2 \le \frac{i+3}{4}.$$

In particular,  $\|\mathbf{b}_1\| = \lambda_1(L)$ , and one can see that all the other vectors are very close to the minima. The LLL reduction is a relaxed variant of a notion of reduction proposed by Hermite [Her50]: it depends on a factor  $\delta$  satisfying  $\frac{1}{4} < \delta \leq 1$ . Historically, the factor chosen in [LLL82] was  $\delta = 3/4$  for ease of notation, but the closer  $\delta$  is to 1, the stronger the LLL reduction. An LLL-reduced basis  $(\mathbf{b}_1, \ldots, \mathbf{b}_d)$ with factor  $\delta$  of a lattice L satisfies (see [LLL82]):

- (1)  $\|\mathbf{b}_1\| \le \alpha^{(d-1)/4} (\operatorname{vol} L)^{1/d}$ , where  $\alpha = 1/(\delta \frac{1}{4})$ . (2) For all  $1 \le i \le d$ ,  $\|\mathbf{b}_i\| \le \alpha^{(d-1)/2} \lambda_i(L)$ . (3)  $\|\mathbf{b}_1\| \times \cdots \times \|\mathbf{b}_d\| \le \alpha^{d(d-1)/4} \operatorname{vol} L$ .

Note that for the historical choice  $\delta = 3/4$  of **[LLL82**], we have  $\alpha = 2$ . Interestingly, for the optimal choice  $\delta = 1$ , we obtain  $\alpha = 4/3$ , in which case the previous inequality (1) matches Hermite's exponential bound on Hermite's constant. The vectors of an LLL-reduced basis are thus at most exponentially far from the minima.

We stress that we have not even defined the LLL or HKZ reductions: we only listed a few properties of those reductions. We have not even proved that any lattice must have LLL-reduced bases and HKZ-reduced bases (which holds): typically, the existence of reduced bases is established by means of an algorithm (efficient or not). We will discuss computational aspects of lattice reduction in the next subsection.

**6.6.** Computational Aspects. In this section, we discuss computational aspects of lattices. More information can be found in [MG02, GLS93]. We would like to emphasize that there is a well-known gap between theory and practice: one should be very careful when interpreting theoretical or practical results.

6.6.1. Computational Model. When dealing with complexity aspects, we assume implicitly that the lattices under consideration are rational lattices given explicitly by a basis, that is a matrix with rational coefficients: the cost of the algorithm will be measured with respect to the size of this matrix, that is, the maximal bit-length of the numerator and denominator of the coefficients, as well as the numbers of rows and columns of the matrix. From a practical point of view, all the parameters of the matrix are important. Naturally, a lattice algorithm will be said to be polynomial-time if its running time is polynomial in the size of the matrix representing the lattice. Since any rational lattice can easily be transformed into an integral lattice by an appropriate scaling, we can assume without loss of generality that the input lattices are integral lattices.

One may wonder about alternative representations of lattices. For instance, if one is only given a set of generators of an integral lattice, a classical result states that one can compute in polynomial time a basis of the lattice (see [GLS93]). More generally, a lattice may be given only implicitly, and the first task is to efficiently find a basis. For instance, the set of integral solutions to a system of linear equations over the integers is a lattice: a classical result states that one can

compute in polynomial time a basis of that lattice from the system of equations (see [**GLS93**]).

Complexity results assume that there is a main parameter, and the hardness refers to when that parameter grows to infinity. In the case of lattices, the main parameter is the lattice dimension: the other parameters (bit-length of the matrix entries and the dimension of the space) are then assumed to be polynomial in the lattice dimension. In fixed lattice dimension, all lattice problems become easy.

6.6.2. *Lattice Problems*. There are many computational problems related to lattices, which can be roughly classified in two categories: those which are easy, and those which are believed to be hard.

Among the easy lattice problems which can be solved in polynomial time (see [**GLS93**]), one can find:

**Membership:** Given a basis of a lattice L in  $\mathbb{Q}^n$  and a target vector  $\mathbf{t} \in \mathbb{Q}^n$ , decide if  $\mathbf{t} \in L$  or not.

**Equality:** Given bases of two lattices  $L_1$  and  $L_2$  in  $\mathbb{Q}^n$ , decide if  $L_1 = L_2$ . **Inclusion:** Given bases of two lattices  $L_1$  and  $L_2$  in  $\mathbb{Q}^n$ , decide if  $L_1 \subseteq L_2$ . **Intersection:** Given bases of two lattices  $L_1$  and  $L_2$  in  $\mathbb{Q}^n$ , find a basis of the lattice  $L_1 \cap L_2$ .

Interestingly, there are lattice problems which seem to be very hard, due to the existence of NP-hardness results (see [MG02]). The most famous lattice problem is the *shortest vector problem* (SVP for short): given a basis of a rational lattice L, find  $\mathbf{v} \in L$  such that  $\|\mathbf{v}\| = \lambda_1(L)$ . Because Ajtai [Ajt98] proved that SVP is NP-hard under randomized reductions, the existence of efficient algorithms to solve SVP seems unlikely. In fact, the best deterministic SVP algorithm is Kannan's super-exponential algorithm [Kan83] which requires  $O(d^{d/(2e)+o(d)})$  polynomial operations (and negligible memory) [HS07, HS08], where d is the lattice dimension. The probabilistic SVP algorithm of [AKS01] improves the running time to  $2^{O(d)}$  polynomial operations, but its space requirements become exponential  $2^{O(d)}$  (see [**NV08**] for an assessment of the O() constant). In low dimension  $d \leq 4$ , there is an elegant and very efficient algorithm to solve SVP, which generalizes Lagrange's algorithm (see [**NS04**]). Since SVP seems to be a very hard problem, one often considers approximate versions of SVP. For instance the  $\gamma$ approximate SVP with  $\gamma \in \mathbb{R}$  is: given a basis of a rational lattice L, find a nonzero  $\mathbf{v} \in L$  such that  $\|\mathbf{v}\| \leq \gamma \lambda_1(L)$ . Approximating SVP within a factor  $\gamma$  means solving  $\gamma$ -approximate SVP. Naturally, the bigger  $\gamma$ , the easier  $\gamma$ -approximate SVP. The LLL algorithm [LLL82] solves  $(4/3)^{d/2}$ -approximate SVP in polynomial time (see [MG02]). The best polynomial-time deterministic algorithm for approximate-SVP is Gama and Nguyen's algorithm [GN08a] (an improvement of [Sch87, **GHGKN06**]) with an appropriate blocksize k, which can solve  $\gamma$ -approximate SVP for  $\gamma = \gamma_k^{(d-k)/(k-1)} = 2^{O(d(\log \log d)^2/\log d)}$  for  $k = O(\log d/\log \log d)$ . The best polynomial-time randomized algorithm for approximate-SVP is Gama and Nguyen's algorithm [GN08a] (an improvement of [Sch87, GHGKN06]) using the randomized AKS algorithm [**AKS01**] within blocks of size k, which can solve  $\gamma$ -approximate SVP for  $\gamma = \gamma_k^{(d-k)/(k-1)} = 2^{O(d(\log \log d)/\log d)}$  for  $k = O(\log d)$ .

Another famous lattice problem is the closest vector problem (CVP for short), also called the *nearest lattice point problem*: given a basis of a rational lattice  $L \in \mathbb{Q}^n$  and a target vector  $\mathbf{t} \in \mathbb{Q}^n$ , find  $\mathbf{v} \in L$  minimizing  $\|\mathbf{t} - \mathbf{v}\|$ , that is, such that  $\|\mathbf{t}-\mathbf{v}\| \leq \|\mathbf{t}-\mathbf{w}\|$  for all  $\mathbf{w} \in L$ . Similarly as for SVP, one defines  $\gamma$ -approximate CVP as follows: given a basis of a rational lattice  $L \in \mathbb{Q}^n$  and a target vector  $\mathbf{t} \in \mathbb{Q}^n$ , find  $\mathbf{v} \in L$  such that  $\|\mathbf{t} - \mathbf{v}\| \leq \gamma \|\mathbf{t} - \mathbf{w}\|$  for all  $\mathbf{w} \in L$ . Note that if one knows a orthogonal basis for the lattice L (such as is the case for  $\mathbb{Z}^n$ ), CVP becomes trivial, but in general, one only knows a weakly-reduced basis, which makes the problem very difficult. CVP was shown to be NP-hard as early as in 1981 [**Emd81**] (for a much simpler "one-line" proof using the knapsack problem, see [**Mic01**]). Babai's nearest plane algorithm [**Bab86**] uses LLL to solve  $2(4/3)^{d/2}$ -approximate CVP in polynomial time (see [**MG02**]). Using any of [**Sch87**, **GHGKN06**, **GN08a**], this can be improved to  $2^{O(d(\log \log d)^2/\log d)}$  in polynomial time, and even further to  $2^{O(d \log \log d/ \log d)}$  in randomized polynomial time using [**AKS01**], due to Kannan's link between CVP and SVP (see further). For exact CVP, the best algorithm is Kannan's super-exponential algorithm [**Kan83**, **Kan87b**], with running time  $2^{O(d \log d)}$  (see also [**Hel85**, **HS07**] for an improved constant).

Interestingly, NP-hardness results for SVP and CVP are known to have limits. Goldreich and Goldwasser [**GG98**] showed that approximating SVP or CVP to within  $\sqrt{d/\log d}$  cannot be NP-hard, unless the polynomial-time hierarchy collapses.

There are relationships between SVP and CVP. Goldreich *et al.* [GMSS99] showed that CVP cannot be easier than SVP: given an oracle that solves f(d)-approximate CVP, one can solve f(d)-approximate SVP in polynomial time. Reciprocally, Kannan proved in [Kan87a, Section 7] that any algorithm solving f(d)-approximate SVP where f is a non-decreasing function can be used to solve  $d^{3/2}f(d)^2$ -approximate CVP in polynomial time.

In practice, a popular strategy to try to solve CVP when the target vector is very close to the lattice is Kannan's *embedding method* (see [Kan87b, GGH97, Ngu99, MG02]), which uses the previous algorithms for SVP and a simple heuristic reduction from CVP to SVP. Namely, given a lattice basis  $(\mathbf{b}_1, \ldots, \mathbf{b}_d)$  and a vector  $\mathbf{v} \in \mathbb{R}^n$ , the embedding method builds the (d + 1)-dimensional lattice (in  $\mathbb{R}^{n+1}$ ) spanned by the row vectors  $(\mathbf{b}_i, 0)$  and  $(\mathbf{v}, 1)$ . Depending on the lattice, one should choose a coefficient different from 1 in  $(\mathbf{v}, 1)$ . It is hoped that a shortest vector of that lattice is of the form  $(\mathbf{v} - \mathbf{u}, 1)$  where  $\mathbf{u}$  is a closest vector (in the original lattice) to  $\mathbf{v}$ , whenever the distance to the lattice is smaller than the lattice first minimum. This heuristic may fail (see for instance [Mic98] for some simple counterexamples), but it can also sometimes be proved, notably in the case of lattices arising from low-density knapsacks (see [NS05b]).

Approximating SVP or CVP is often achieved by solving a more general problem: lattice reduction, which is roughly speaking finding a basis close to all the minima.

6.6.3. *Cost of HKZ and LLL Reductions*. The classical results regarding HKZ and LLL reductions are the following:

- It is possible to compute an HKZ-reduced basis of a *d*-dimensional lattice in  $O(d^{O(d)})$  polynomial operations (see [Kan83, Sch87, HS07]): note that this running time is super-exponential in *d*.
- If the reduction factor  $\delta$  is a rational number such that  $1/4 < \delta < 1$ , the LLL algorithm [LLL82] computes an LLL-reduced basis of factor  $\delta$  in polynomial time (see also [NS05a] for optimized variants). Note that we need  $\delta < 1$ , in which case  $\alpha > 4/3$ . In practice, one often uses  $\delta = 0.99$  so that  $\alpha \approx 4/3$ : in [MG02], it is even shown how to select  $\delta$  converging

to 1 while keeping polynomial-time complexity. However, the constant  $\alpha$  is typically a worst-case constant: on the average, in practice, it seems that  $\alpha$  should be replaced by a smaller constant close to 1.08 for moderate dimension (see [**NS06, GN08b**]).

6.6.4. *Experimental Facts.* For those who are interested in performing experiments, the NTL library [**Sho**] provides an easy-to-use lattice package, which includes efficient implementations of the main lattice reduction algorithms. In low dimension, one can also play with GP/PARI [**BBB**<sup>+</sup>].

In this section, we discuss what can be expected in practice regarding the solvability of lattice problems: more information can be found in [GN08b]. We stress that there is unfortunately no easy rule-of-thumb to predict what one can do or cannot do in practice. In low dimension, say  $\leq 60$ , the most important lattice problems become easy: for instance, exact SVP and CVP can be quickly solved using existing tools. The main reason is that lattice reduction algorithms behave better than their worst-case bounds: see for instance [NS06] for the case of LLL, and [GN08b] for the case of BKZ. However, as soon as the lattice dimension becomes very high, it is difficult to predict experimental results in advance. Several factors seem to influence the result: the lattice dimension, the input basis, the structure of the lattice, and in the case of CVP, the distance of the target vector to the lattice. What is always true is that one can quickly approximate SVP and CVP up to exponential factors with an exponentiation base very close to 1 (see [GN08b] for concrete values of the exponentiation bases), but in high dimension, such exponential factors may not be enough for cryptanalytic purposes, depending on the application. If better approximation factors are required, one should perform experiments to see if existing algorithms are sufficient. If the lattice and the input basis are not exceptional, there is no reason to believe that exact SVP can be solved in very high dimension (say > 300), although one can always give it a try. Furthermore, if the target vector is not unusually close to the lattice, there is also no reason to believe that exact CVP could be solved in very high dimension (say  $\geq 300$ ).

One example of unusual lattice structure is when one knows the existence of a non-zero lattice vector much smaller than Hermite's bound: one should compare the norm of that lattice vector with  $\sqrt{d} \operatorname{vol}(L)^{1/d}$ . In this case, one is advised to try existing algorithms in practice, since there is hope: for instance, [**Ngu99**] reported successes for such SVP instances (and CVP instances for which the target vector is unusually close to the lattice, *i.e.* when the distance is much smaller than  $\sqrt{d} \operatorname{vol}(L)^{1/d}$ ) in very high dimension; and the experiments of [**GN08b**] suggest that SVP can be solved for lattices L such that  $\lambda_2(L)/\lambda_1(L)$  is a not too small fraction of  $1.012^d$ .

### 7. Lattice Attacks

In this section, we survey the main lattice attacks:

- Section 7.1 presents natural attacks which use lattices of low dimension.
- Section 7.2 presents natural attacks which use lattices of high dimension.
- Section 7.3 presents attacks based on unusually small roots of polynomial equations (or congruences): finding such roots is done using lattices of moderate dimension.

**7.1. Low-Dimensional Attacks.** The attacks we will present in this section are fairly representative of attacks based on low-dimensional lattices. Here, the underlying problem which will be tackled by the use of lattices is as follows: assume that we have a linear congruence of the form

(7.1) 
$$\sum_{i=1}^{n} a_i x_i \equiv b \pmod{M},$$

where only the  $x_i$ 's are unknown integers, whereas the integer  $a_i \in \mathbb{Z}$ , the integer  $b \in \mathbb{Z}$  and the modulus M are known. Obviously, if there is no constraint on the size of the  $x_i$ 's, it is easy to find a solution  $(x_1, \ldots, x_n) \in \mathbb{Z}^n$  to (7.1), so we are interested in solutions satisfying special properties, say the size of the  $x_i$ 's is small. When n is small (say, less than 10), the following holds:

- Lattice reduction can efficiently find a solution  $(x_1, \ldots, x_n) \in \mathbb{Z}^n$  such that  $x_i = O(M^{1/n})$ . Note that this is trivial if n = 1. If  $b \equiv 0 \pmod{M}$ , the problem can be reduced to finding a very short vector in a lattice. If  $b \not\equiv 0 \pmod{M}$ , the problem can be reduced to finding a very close lattice vector.
- If there is an exceptional solution  $(x_1, \ldots, x_n) \in \mathbb{Z}^n$  such that  $\prod_{i=1}^n x_i$  is much smaller than M, then it can probably be recovered in practice, and perhaps also in theory. More precisely, if  $b \equiv 0 \pmod{M}$ , it means that there exists an exceptionally short vector in a certain lattice. And if  $b \not\equiv 0 \pmod{M}$ , it means that there exists a vector in a certain lattice which is unusually close to a certain target vector.

Such results have been applied many times in cryptanalysis.

7.1.1. RSA with small secret exponent. Consider the usual RSA key generation:

- The public modulus is N = pq where p and q are large primes of about the same bit-length, that of  $N^{1/2}$ .
- The pair (e, d) of public and secret exponents satisfy the congruence (2.1), and we have  $0 \le e, d \le N$ .

Wiener [Wie90] showed that if the secret exponent d is such that  $0 \le d \le N^{1/4}$ , then one can recover p and q in polynomial time from N and e. Wiener's attack was historically presented using continued fractions. Here, we will present a lattice version of this attack, based on a two-dimensional shortest vector problem. Note that this lattice version will only be heuristic, while Wiener's attack is provable: however, in practice, both work as well. Furthermore, this lattice attack is fairly representative of the numerous heuristic cryptanalyses based on low-dimensional lattices.

Because p and q are balanced, we have:

$$\phi(N) = N + O(\sqrt{N}).$$

The congruence (2.1) implies the existence of some k = O(d) such that  $e \cdot d = 1 + k(N + O(\sqrt{N}))$ , thus:

$$\ell = e \cdot d - kN = O(d\sqrt{N}).$$

Now consider the 2-rank lattice L spanned by the rows of:

$$\begin{pmatrix} e & \sqrt{N} \\ N & 0 \end{pmatrix}$$

Then L contains  $\mathbf{t} = d \times \text{first row} - k \times \text{second row} = (\ell, d\sqrt{N})$ , whose norm is  $\approx d\sqrt{N}$ , while  $\text{vol}(L)^{1/2} = N^{3/4}$ . Thus,  $\mathbf{t}$  is heuristically expected to be the shortest vector of L if  $d\sqrt{N} < N^{3/4}$ , that is,  $d \leq N^{1/4}$ . Note however that we do not claim to have proved that  $\mathbf{t}$  is the shortest vector: it is only a very reasonable guess. By solving SVP in the 2-rank lattice L, we can hope to find  $\mathbf{t}$ , and therefore the secret exponent d.

Let us a give a baby example for concreteness. Assume that Alice had selected the primes p = 6011673201679823947 and q = 6987193563793194751, so that her RSA modulus is:

#### N = 42004724302405294297751453898364502197.

The bit-length of N is 125, so let us assume that Alice selected a 30-bit prime at random as her secret exponent, such as d = 814510573, so that Wiener's bound  $d < N^{1/4}$  is satisfied. Then Alice's public exponent is:

e = 17924546723775007116522646995236610637.

From the public key (e, N), the attacker computes  $\sqrt{N} \approx 6481105176002414967$ , and derives the following  $2 \times 2$  integer matrix:

```
\begin{pmatrix} 17924546723775007116522646995236610637 & 6481105176002414967 \\ 42004724302405294297751453898364502197 & 0 \end{pmatrix}.
```

After running Lagrange's algorithm, the attacker obtains the following reduced basis:

```
\begin{pmatrix} 4518062787607145156653412229 & -5278928690578992864154946091 \\ 28630395383776734081193510984 & 26803350500352508931781506895 \end{pmatrix}.
```

Notice that the first row vector of the reduced basis is substantially shorter than the second row vector, which proves that the lattice is not random. From the first row vector, the attacker guesses that Alice's secret exponent is:

```
d = 5278928690578992864154946091/6481105176002414967 = 814510573,
```

which is correct!

7.1.2. RSA signatures with constant-based padding. We saw in Section 4.1.1 an adaptive chosen-message universal forgery on Textbook-RSA, thanks to the multiplicativity of the RSA permutation. This forgery shows that one should preprocess the message before signing it, and check the preprocessing when verifying the signature. One early candidate of preprocessing is constant-based padding, which means that we pad the messages by a fixed (public) series of bits before signing, and check that redundancy when verifying a signature. In other words:

- There is a constant *P* defining the padding.
- A message m to sign is assumed to be small (say,  $|m| \leq M$  where M is much smaller than N), and its signature is:

$$s = (P+m)^d \pmod{N}.$$

• A signature s of a message m is checked using the congruence:

$$s^e \equiv P + m \pmod{N}.$$

One further checks that m is sufficiently small, that is,  $|m| \leq M$ .

#### PHONG Q. NGUYEN

The smaller M is, the harder it should be for an attacker to forge signatures.

Assume that we have three messages  $m_1$ ,  $m_2$  and  $m_3$ . Each  $m_i$  is signed as:

$$s_i = (P + m_i)^d \pmod{N}$$

Then  $s_1 \equiv s_2 s_3 \pmod{N}$  if and only if  $(P+m_1) \equiv (P+m_2)(P+m_3) \pmod{N}$ . We claim that given  $m_3$ , we can find suitable  $m_1$  and  $m_2$  less than roughly  $\sqrt{N}$  using lattice reduction, namely approximating the closest vector problem in dimension two. This leads to a chosen-message universal forgery, provided that the message size is at least half that of N.

We want to solve  $(P + m_1) \equiv (P + m_2)(P + m_3) \pmod{N}$ , which is of the form  $m_1 - m_2 \alpha \equiv \beta \pmod{N}$ . Consider the 2-rank lattice L of all  $(x, y) \in \mathbb{Z}^2$ such that  $x - y\alpha \equiv 0 \pmod{N}$ . Notice that  $\operatorname{vol}(L) = N$ . We can hope to find a lattice vector  $\mathbf{u} = (u_1, u_2)$  whose distance to  $\mathbf{t} = (\beta, 0)$  is  $\approx \operatorname{vol}(L)^{1/2} \approx N^{1/2}$ . Then  $m_1 = \beta - u_1$  and  $m_2 = -u_2$  are both  $O(N^{1/2})$ . And  $m_1 - m_2\alpha \equiv \beta \pmod{N}$ which leads to a heuristic forgery which works very well in practice. Hence, we have obtained a chosen-message universal forgery up to the bound  $M \approx N^{1/2}$ . Interestingly, the bound  $N^{1/2}$  for the message bound M has been improved to  $N^{1/3}$  using four messages by Brier *et al.* [BCCN01] using different lattice-based techniques: Lenstra and Shparlinski [LS02] improved the existential forgery of [BCCN01] into a universal forgery.

Again, let us give a baby example for concreteness. We take the same RSA modulus N as the example of Section 7.1.1. The constant P is chosen as the first decimal digits of  $\pi$  multiplied by a suitable power of 10:

We would like to sign the message  $m_3 = 2718281828459045235$ . This implies that:

 $\beta = P(P+m_3) - P \equiv 28532925287943337534233793526174219074 \pmod{N},$  and

 $\alpha = P + m_3 \equiv 31415926535897932302718281828459045235 \pmod{N}.$ 

So we consider the lattice L of all  $(x, y) \in \mathbb{Z}^2$  such that  $x - y\alpha \equiv 0 \pmod{N}$ . The following  $2 \times 2$  matrix is clearly a basis of L:

$$\begin{pmatrix} \alpha & 1 \\ N & 0 \end{pmatrix} = \begin{pmatrix} 31415926535897932302718281828459045235 & 1 \\ 42004724302405294297751453898364502197 & 0 \end{pmatrix}$$

After running the LLL algorithm, we obtain the following reduced basis:

 $\begin{pmatrix} 2840910670399556715 & 3383974095730158874 \\ -8143041377019128593 & 5086004066464213681 \end{pmatrix}$ 

After running Babai's nearest plane algorithm [**Bab86**] on the target vector  $\mathbf{t} = (\beta, 0)$ , we obtain the following lattice vector

 $\mathbf{u} = (28532925287943337532025597115229231563 \quad 1667092550642276495),$ 

which leads to  $m_1 = 2208196410944987511$  and  $m_2 = -1667092550642276495$ . It can be checked that:

 $(P+m_1) \equiv (P+m_2)(P+m_3) \pmod{N}.$ 

Note that both  $|m_1|$  and  $|m_2|$  are less than  $2^{61}$ , whereas the bit-length of  $m_3$  is 62. By comparison, the bit-length of N is 125, so  $m_1$  and  $m_2$  are indeed close to  $\sqrt{N}$ .

36

7.1.3. Elgamal signature in GnuPG. GnuPG [**GPG**] is a widely deployed software to secure emails: it is present in most distributions of the Linux operating system, and is more or less an open source version of the famous PGP software. Prior to the publication of [**Ngu04**], GnuPG included an implementation of the Elgamal signature, which turned out to be extremely insecure. Namely, Nguyen showed in [**Ngu04**] that after one signature had been released, an attacker could recover the signer's secret key in less than a second on a personal computer. The attack is based on low-dimensional lattices.

First, let us describe the Elgamal signature scheme as implemented in GnuPG, which slightly differs from Textbook Elgamal. The parameters are a large prime p such that (p-1)/2 has large factors, and a generator g of  $\mathbb{Z}_p^*$ . The secret key is a small exponent x less than  $p^{3/8}$ , and the public key is  $y = g^x \pmod{p}$ . Messages are preprocessed before being signed, using a usual padding which we omit here. To sign a padded message  $m \in \mathbb{Z}_p$ :

- Select a small "random" number k coprime with p-1, which turns out to be less than  $p^{3/8}$ .
- The signature is (a, b) where  $a = g^k \mod p$  and  $b = (m ax)k^{-1} \pmod{p-1}$ .

The attack [Ngu04] works as follows. Assume that a signature (a, b) of a message m is known. We focus on the congruence  $b \equiv (m - ax)k^{-1} \pmod{p-1}$  satisfied by the signature (a, b), that is:

(7.2) 
$$bk + ax \equiv m \pmod{p-1},$$

where both k and x are  $\leq p^{3/8}$ . Consider the 2-rank lattice L of all  $(\alpha, \beta) \in \mathbb{Z}^2$  such that:

$$b\alpha + a\beta \equiv 0 \pmod{p-1}$$
.

The volume of L is  $\operatorname{vol}(L) = (p-1)/\operatorname{gcd}(a, b, p-1) \approx p$  because a and b are unlikely to have a large gcd. We can easily find integers  $u_1, u_2 \in \mathbb{Z}$  such that

$$bu_1 + au_2 \equiv m \pmod{p-1}$$

Then the lattice vector  $\mathbf{t} = (u_1 - k, u_2 - x) \in L$  is unusually close to  $\mathbf{u} = (u_1, u_2)$ : the distance between  $\mathbf{t}$  and  $\mathbf{u}$  is less than  $p^{3/8}$ , which is itself much less than  $\operatorname{vol}(L)^{1/2} \approx p^{1/2}$ . If  $\mathbf{t}$  is indeed the closest lattice vector, then the secret key x is recovered.

The attack is very efficient in practice, and can be made provable if a and b are assumed to be uniformly distributed: for more details, see [Ngu04]. Namely, if a and b are assumed to be uniformly distributed, one can prove that L has no unusually short vectors, which implies that when a lattice vector is unusually close to a target vector, any other lattice vector must be sustantially farther away from that target vector.

**7.2. High-Dimensional Attacks.** As an illustration of attacks based on high-dimensional lattices, we present an important attack which was historically not presented in terms of lattices, but which can interestingly be viewed in terms of lattices in a simple manner. The attack is Bleichenbacher's celebrated chosen-ciphertext attack [Ble98] on RSA-PKCS#1 encryption version 1.5, which arguably motivated the use of chosen-ciphertext security in cryptography standards. Bleichenbacher did not use lattices because he wanted to optimize the attack, but the lattice version of the attack is perhaps easier to understand.

### PHONG Q. NGUYEN

In Section 4.2.1, we saw elementary attacks on Textbook RSA encryption which show that messages must be preprocessed prior to raw RSA encryption (raising to the power e modulo N). A natural question arises: which preprocessing should one use? In the nineties, a very popular solution was to use the PKCS#1 v1.5 standard [Lab] advocated by the RSA company: for instance, the standard was used in SSL v3.0, which is widely deployed in Internet browsers. The standard specified how to transform a message M, prior to raw RSA encryption (that is, exponentiation to the power e):

- The message *m* to encrypt is assumed to be much smaller than the RSA modulus *N*: it will be at least a few bytes less than *N*.
- This value is then padded as described in PKCS#1 v1.5 block type 02 (see Figure 1): a zero byte is added to the left, as well as as many nonzero random bytes as necessary in such a way that the first two bytes of the final value are 00 02 followed by as many nonzero random bytes as necessary to match the size of the modulus. In other words, the whole value m described in Figure 1 must fit the size of the modulus N.

00   02   Non-zero random bytes $  00  $ message	00	-zero random by	es $00$ Message $\Lambda$
--	----	-----------------	---------------------------



When decrypting an RSA ciphertext encrypted by the PKCS#1 v1.5 block type 02 standard, one recovers a value m of the form given in Figure 1 and must proceed as follows to recover the message M:

- One first checks that the first two most significant bytes are 00 and 02.
- Next, one removes all the non-zero bytes until one finds a 00 byte.
- The rest must be the message M.

But what if the decryption process failed? For instance, what if the first two most significant bytes of  $c^d \pmod{N}$  are not 00 and 02? Such a situation may arise since anybody can submit ciphertexts, and therefore, ciphertexts are not necessarily valid ciphertexts. Bleichenbacher [**Ble98**] noticed that in several implementations of SSL v3.0, the person who decrypts – in real life, a server which receives many messages encrypted with its RSA public key – actually returns an error message when there is a problem during the decryption process. In other words, in this case, an adversary has access to a 0002-oracle: given any  $c \in \mathbb{Z}_N$ , the 0002-oracle answers whether or not the first two most significant bytes of  $c^d \pmod{N}$  are 00 and 02. In [**Ble98**], Bleichenbacher showed how such an oracle enables an adversary to decrypt any RSA ciphertext  $c = m^e \pmod{N}$ , including those m of the form described in Figure 1. The attack presented in [**Ble98**] is rather technical, so as to minimize the number of queries to the oracle. In these notes, we will present an alternative lattice-based attack, which is simpler to present, but is not intended to be optimal: the main ideas are nevertheless identical.

Assume that there is an RSA public key (N, e), and that a message  $m \in \{0, \ldots, N-1\}$  has been encrypted as  $c = m^e \pmod{N}$ : the ciphertext c is public, but the message m is of course secret. Assume also that one has access to a 0002-oracle  $\mathcal{O}$ : given any  $c' \in \mathbb{Z}_N$ , the oracle  $\mathcal{O}$  answers whether or not the first two most significant bytes of  $c'^d \pmod{N}$  are 00 and 02. We will see how one can recover m using a reasonable number of oracle queries.

First of all, we note that if we select a  $c \in \mathbb{Z}_N$  uniformly at random, the probability that the oracle  $\mathcal{O}$  answers yes is very close to  $1/256^2 = 1/65536$ : here, we only say "very close" because N is not exactly a power of two. This suggests to do the following many times:

- Select uniformly at random  $r \in \mathbb{Z}_N$ .
- Compute  $c' = r^e c \pmod{N}$  and send c' to the oracle  $\mathcal{O}$ .
- If the answer is yes (which should happen with probability 1/65536), store r: it means that  $rm \pmod{N}$  starts with 00 02, because  $c' \equiv (rm)^e \pmod{N}$  by multiplicativity of the RSA permutation. Otherwise, start again.

We now know many random integers  $r_1, \ldots, r_n$  such that each  $r_im \mod N$  starts with 00 02, and we would like to recover the message m. If we knew one of the  $r_im \mod N$  exactly, it would be easy to recover m by dividing by  $r_i$ , whose value is known. But here, we only know an approximation of each of the  $r_im \mod N$ : more precisely, if we let  $a = 2^{\ell-15}$  where  $\ell$  is the bit-length of N, then a represents the number 0002 shifted to the left  $0 \leq (r_im \mod N) - a < N/2^{16}$ ; we can even have a better approximation if we use  $a' = a + N/2^{17}$ , which implies that:

$$|(r_i m \mod N) - a'| \le N/2^{17}.$$

This kind of problem has been dubbed hidden number problem (HNP) by Boneh and Venkatesan  $[\mathbf{BV96}]$ : here, *m* is the "hidden number". Boneh and Venkatesan  $[\mathbf{BV96}]$  studied the HNP to obtain bit-security results on the Diffie-Hellman key exchange in prime fields. Later, the HNP and variants were applied to cryptanalysis (see  $[\mathbf{NS02, NS03}]$ ), namely to attack classical signature schemes based on the discrete logarithm problem when partial information on the one-time keys used during signature generation is leaked.

We will now solve the HNP by viewing it as a lattice closest vector problem. Consider the (n + 1)-rank lattice L spanned by the following rows:

$\binom{1/65536}{0}$	$r_1$ N	$r_2 \\ 0$	 	$\begin{pmatrix} r_n \\ 0 \end{pmatrix}$
•	0	N	·	÷
:	:	·	·	0
\ 0	- 0		- 0	N/

By multiplying the first row by m, and subtracting appropriate multiples of the other rows, one sees that the lattice L contains the vector

 $\mathbf{m} = (m/65536, mr_1 \mod N, \dots, mr_n \mod N).$ 

If we could recover the vector  $\mathbf{m} \in L$ , we would derive the message m. Since each  $mr_i \mod N$  starts with the 00 02 bytes, we have seen that if  $\ell$  denotes the bit-length of N then the constant  $a' = 2^{\ell-15} + N/2^{17}$  satisfies:

$$|(r_i m \mod N) - a'| \le N/2^{17}.$$

This suggests to define the target vector  $\mathbf{t} = (N/2^{17}, a', a', \dots, a')$ . We note that the lattice vector  $\mathbf{m}$  is very close to  $\mathbf{t}$ . Indeed, each coordinate of  $\mathbf{m} - \mathbf{t}$  is less than  $N/2^{17}$  in absolute value. Hence:

$$\|\mathbf{m} - \mathbf{t}\| \le N\sqrt{n+1}/2^{17}.$$

Is that distance exceptional? Since  $vol(L) = N^n/65536$  and L has rank n + 1, a typical lattice distance should be:

$$\sqrt{n+1}(N^n/65536)^{1/(n+1)}$$

which is roughly  $\sqrt{n+1}N^{n/(n+1)}$ . Thus, one would expect  $\mathbf{m} \in L$  to be heuristically the closest vector to  $\mathbf{t}$  if:

(7.3) 
$$N/2^{17} \ll N^{n/(n+1)}.$$

If  $\mathbf{m} \in L$  was indeed the closest lattice vector, any CVP oracle would disclose  $\mathbf{m}$ . However, in general, the closest vector problem can only be solved in practice when the dimension is not too big. So we shouldn't take too large values of n.

We performed experiments on 512-bit and 1024-bit modulus N using the NTL library [**Sho**]. To try to solve the closest vector problem, we applied Babai's nearest plane algorithm [**Bab86**] on an LLL-reduced basis and BKZ-reduced bases of blocksize 10 and 20. If N is a 512-bit number, then (7.3) is satisfied as soon as  $n + 1 \gg 30$ . In practice, we were able to recover m within a few seconds when n is roughly greater than 40. This means that the total number of oracle queries is  $\approx 40 \times 65536 = 2,621,440$ . If N is a 1024-bit number, then (7.3) is satisfied as soon as  $n + 1 \gg 60$ . In practice, we were able to recover m within a few minutes when n is roughly greater than 80. This means that the total number of oracle queries is  $\approx 80 \times 65536 = 5,242,880$ . Interestingly, the numbers of oracle queries required by the lattice attack are not that much bigger than in the initial (non-lattice) method of Bleichenbacher [**Ble98**].

**7.3.** Polynomial Attacks. We now survey an important application of lattice reduction found in 1996 by Coppersmith [Cop97, Cop01], and its developments. These results illustrate the power of linearization combined with lattice reduction.

7.3.1. Univariate modular equations. Consider Textbook-RSA encryption with a small public exponent e, such as e = 3. Recall that a message  $m \in \mathbb{Z}_N$  is encrypted as:

$$c = m^e \mod N.$$

We saw in Section 4.2.1 the short-message attack: if  $0 \le m \le N^{1/e}$ , then the short message m can be recovered from c. Can this short-message attack be extended?

For instance, if the message m is the shift of a short message (less than  $N^{1/e}$ ), then the same attack applies, after division (modulo N) by a suitable power of two. More generally, what if only a few consecutive bits of the message m were unknown? Such messages are called stereotyped: many parts of the message are known. This is the case of emails, which include known fields such as the name of the sender, the name of the recipient, *etc.* Formally speaking, we assume that the secret message  $m \in \{0, \ldots, N-1\}$  is of the form:

$$m = m_0 + 2^k s,$$

where  $m_0, s, k$  are all non-negative integers, but only s is secret: the integers  $m_0$  and k are known. This corresponds to the situation where m = 'known bits'  $\parallel$  'unknown bits'  $\parallel$ 'known bits', where the  $\parallel$  symbol denotes concatenation. Then the ciphertext c satisfies:

$$c = (m_0 + 2^k s)^e \pmod{N},$$

40

which, after division by a suitable power of two, can be rewritten as

$$P(s) \equiv 0 \pmod{N},$$

where  $P(x) \in \mathbb{Z}[x]$  is a monic polynomial of degree *e* whose coefficients can all be derived from  $c, k, m_0$  and *N*.

At first sight, this is just an instance of the general problem of solving univariate polynomial equations modulo some integer N of unknown factorization, which is considered to be hard. Indeed, for some polynomials, the problem is equivalent to the knowledge of the factorization of N. And the particular case of extracting *e*-th roots modulo N is the problem of decrypting ciphertexts in the RSA cryptosystem, for an eavesdropper. Surprisingly, Coppersmith [Cop97] showed using the LLL algorithm that the special problem of finding small roots was easy:

THEOREM 7.1 (Coppersmith). Let  $P(x) \in \mathbb{Z}[x]$  be a monic polynomial of degree  $\delta$  in one variable, and let N be an integer of unknown factorization. Then one can find in time polynomial in  $(\log N, \delta)$  all integers  $x_0$  such that  $P(x_0) \equiv 0 \pmod{N}$  and  $|x_0| \leq N^{1/\delta}$ .

Before proving Theorem 7.1, let us make a few remarks. Related (but weaker) results appeared in the eighties [Hås88, VGT88]. More precisely, Håstad [Hås88] presented his result in terms of a system of low-degree modular equations, but he actually studies the same problem, and his approach proves a weaker version of Theorem 7.1 with the smaller bound  $N^{2/(\delta(\delta+1))}$  instead of  $N^{1/\delta}$ . Incidentally, Theorem 7.1 implies that the number of roots less than  $N^{1/\delta}$  is polynomial, which was also proved in [KS94] (using elementary techniques).

Theorem 7.1 is easy to prove if P(x) is of the form  $P(x) = x^{\delta} + c$ : this is what we used in the short-message attack against Textbook-RSA. Can we hope to improve the bound  $N^{1/\delta}$  to say  $C \times N^{1/\delta}$ ? It is not difficult to see that we can do so if we multiply the polynomial running-time of Theorem 7.1 by C, but the new running-time is then exponential in  $\log C$ : namely, one splits the roots interval of length  $2C \times N^{1/\delta}$  into roughly C intervals of length  $2N^{1/\delta}$ . Unfortunately, if one would like to keep a polynomial running-time, one cannot hope to improve the (natural) bound  $N^{1/\delta}$  for all polynomials and all moduli N. Indeed, for the polynomial  $P(x) = x^{\delta}$  and  $N = p^{\delta}$  where p is prime, the roots of P mod N are the multiples of p. Thus, one cannot hope to find all the small roots (slightly) beyond  $N^{1/\delta} = p$ , because there are simply too many of them. This suggests that even an SVP-oracle (instead of an approximate-SVP algorithm like LLL) should not improve Theorem 7.1 in general, as evidenced by the proof of Theorem 7.1: the approximation factor provided by LLL does not play a significant role, because the lattices considered by the proof have a huge volume (compared to their dimension). It was noticed in [BN00] that if one only looks for the smallest root mod N, an SVPoracle can improve the bound  $N^{1/\delta}$  for very particular moduli (namely, squarefree N of known factorization, without too small factors). Note that in such cases, finding modular roots can still be difficult, because the number of modular roots can be exponential in the number of prime factors of N. Coppersmith discusses potential improvements in [Cop01]. For instance, the condition P(X) being monic can replaced by the gcd of the coefficients of P(X) and N being equal to 1.

Theorem 7.1 has many applications. The historical application was to attack RSA encryption when a very small public exponent is used (see [Bon99] for a survey). Later applications include Chinese remaindering in the presence of noise **[BN00**], and surprisingly, a few security proofs of factoring-based cryptographic schemes (see [Sho01, Bon01]). We already saw the cryptanalytic application to stereotyped messages, which generalized the short-message attack: if there are less than  $(\log N)/e$  unknown consecutive bits in the message m, then the whole message m can be recovered in polynomial time from its ciphertext  $c = m^e$ (mod N) and the public key (N, e). A less direct application is the random pad: to prevent elementary attacks on RSA, rather than applying the PKCS#1 v1.5 padding, one could simply append random bytes to a message m, before raising it to the power e modulo N. More precisely the ciphertext of a message  $m \ll N$  is  $c = (m \| r)^e \pmod{N}$ , where r is a sequence of bits chosen uniformly at random for each encryption. If the same message  $m \ll N$  is encrypted twice, an adversary may collect the ciphertexts  $c_1 = (m \| r_1)^e \pmod{N}$  and  $c_2 = (m \| r_2)^e \pmod{N}$ . Coppersmith [Cop97] noticed that by computing the resultant of those two polynomials in  $(m, r_1, r_2)$ , one obtains a univariate polynomial congruence modulo N of degree  $e^2$ , satisfied by  $r_1 - r_2$ . Thus, if the random sequence r has less than  $\log(N)/(e^2)$ bits, then one can recover  $r_1 - r_2$ , which eventually leads to the recovery of the message m using other techniques (see [**CFPR96**]).

We now sketch a proof of Theorem 7.1, in the spirit of Howgrave-Graham [HG97], who simplified Coppersmith's original proof by working in the dual lattice of the lattice originally considered by Coppersmith. More details can be found in the survey [Cop01]. Coppersmith's method reduces the problem of finding small modular roots to the (easy) problem of solving polynomial equations over  $\mathbb{Z}$ . More precisely, it applies lattice reduction to find an integral polynomial equation satisfied by all small modular roots of P. The intuition is to linearize all the equations of the form  $x^i P(x)^j \equiv 0 \pmod{N^j}$  for appropriate integral values of i and j. Such equations are satisfied by any solution of  $P(x) \equiv 0 \pmod{N}$ . Small solutions  $x_0$  will give rise to unusually short solutions to the resulting linear system. To transform modular equations into integer equations, we will use the elementary fact that any sufficiently small integer must be zero. More precisely, we will use the following elementary lemma<sup>1</sup>, with the notation  $||r(x)|| = \sqrt{\sum a_i^2}$  for any polynomial  $r(x) = \sum a_i x^i \in \mathbb{Q}[x]$ :

LEMMA 7.2. Let  $r(x) \in \mathbb{Q}[x]$  be a polynomial of degree  $\langle n \rangle$  and let X be a positive integer. Suppose  $||r(xX)|| \langle 1/\sqrt{n}$ . If  $r(x_0) \in \mathbb{Z}$  with  $|x_0| \leq X$ , then  $r(x_0) = 0$  holds over the integers.

**PROOF.** If  $|x_0| \leq X$ , then the Cauchy-Schwarz inequality ensures that:

$$r(x_0)^2 = \left(\sum_{i=0}^{n-1} r_i x_0^i\right)^2 = \left(\sum_{i=0}^{n-1} r_i X^i (x_0/X)^i\right)^2$$
$$\leq \left(\sum_{i=0}^{n-1} (r_i X^i)^2\right) \times \left(\sum_{i=0}^{n-1} (x_0/X)^{2i}\right)$$
$$\leq \|r(xX)\|^2 \times n$$

Thus, if we further have  $||r(xX)|| < 1/\sqrt{n}$ , then  $|r(x_0)| < 1$ . Hence, if  $r(x_0) \in \mathbb{Z}$ , it must be zero.

<sup>&</sup>lt;sup>1</sup>A similar lemma is used in [Hås88]. Note also the resemblance with [LLL82, Prop. 2.7].

We would like to apply Lemma 7.2 to a suitable polynomial  $r(x) \in \mathbb{Q}[x]$ , that is, a polynomial satisfying:

- Property 1:  $||r(xX)|| < 1/\sqrt{1 + \deg r}$ . In other words, the vector corresponding to the polynomial r(xX) must be short.
- Property 2:  $r(x_0) \in \mathbb{Z}$  whenever  $P(x_0) \equiv 0 \pmod{N}$  and  $x_0 \in \mathbb{Z}$ .

If we find such a polynomial  $r(x) \in \mathbb{Q}[x]$ , then by solving the equation  $r(x_0) = 0$ over  $\mathbb{Z}$ , we will find in polynomial time all the integers  $x_0 \in \mathbb{Z}$  such that  $P(x_0) \equiv 0$ (mod N) and  $|x_0| \leq X$ . And if X is sufficiently close to  $N^{1/\delta}$ , say  $N^{1/\delta} = O(X)$ , then Theorem 7.1 would follow.

But how can we find such a polynomial  $r(x) \in \mathbb{Q}[x]$ ? An obvious candidate is  $q(x) = P(x)/N \in \mathbb{Q}[x]$ , which clearly satisfies Property 2. But it is unclear whether q(x) would satisfy Property 1. Other natural choices would be all the polynomials of the form  $q_{u,v}(x) = x^u(P(x)/N)^v \in \mathbb{Q}[x]$  where u and v are nonnegative integers. Such polynomials satisfy Property 2, just like q(x), but they are also unlikely to satisfy Property 1. There are however many other candidates: notice that any integral linear combination of the  $q_{u,v}(x)$ 's satisfies Property 2, and maybe one such combination could satisfy Property 1. This suggests to find  $r(x) \in \mathbb{Q}[x]$  satisfying Lemma 7.2 among all integral linear combinations of the  $q_{u,v}(x)$ 's: this is reminiscent of finding short vectors in a lattice.

Since there is an infinite number of  $q_{u,v}(x)$ 's, it might be useful to restrict to polynomials of bounded degree, where the bound is a parameter which we will select in an appropriate manner. More precisely, let us consider a non-negative integer h, as well as the  $n = (h + 1)\delta$  polynomials  $q_{u,v}(x) = x^u(P(x)/N)^v \in \mathbb{Q}[x]$ , where  $0 \le u \le \delta - 1$  and  $0 \le v \le h$ . Now, we would like to find a short vector in the lattice corresponding to the  $q_{u,v}(xX)$ 's. More precisely, define the  $n \times n$ matrix M whose *i*-th row consists of the coefficients of  $q_{u,v}(xX)$ , starting by the low-degree terms, where  $v = \lfloor (i-1)/\delta \rfloor$  and  $u = (i-1) - \delta v$ . Notice that the *i*-th row represents a polynomial  $q_{u,v}(xX)$  of degree i - 1, whose leading coefficient is  $X^u(X^{\delta}/N)^v = X^{i-1}/N^v$ . Hence, the matrix M is lower triangular, and a simple calculation leads to:

$$\det(M) = X^{n(n-1)/2} N^{-nh/2}.$$

Let us now apply an LLL-reduction to the full-dimensional lattice spanned by the rows of M. The first vector of the reduced basis corresponds to a non-zero polynomial of the form r(xX), and has Euclidean norm ||r(xX)||. The theoretical bounds of the LLL algorithm ensure that:

$$||r(xX)|| \le 2^{(n-1)/4} \det(M)^{1/n} = 2^{(n-1)/4} X^{(n-1)/2} N^{-h/2}.$$

Recall that we need  $||r(xX)|| \leq 1/\sqrt{n}$  to apply Lemma 7.2. Hence, for a given choice of h, the method is guaranteed to find modular roots  $x_0$  up to the bound X if the bound satisfies:

$$X \le \frac{1}{\sqrt{2}} N^{h/(n-1)} n^{-1/(n-1)}.$$

The limit of the upper bound, when h grows to  $\infty$ , is  $\frac{1}{\sqrt{2}}N^{1/\delta}$ . Thus, we would like to select a sufficiently large h so that the bound X satisfies  $N^{1/\delta} = O(X)$ , but on the other hand, we need to keep the running time of the algorithm polynomial by restricting to sufficiently small values of h, with respect to log N and  $\delta$ . Fortunately, both requirements are compatible: Theorem 7.1 follows from an appropriate choice of h as a function of log N and  $\delta$ .

### PHONG Q. NGUYEN

The algorithm of Theorem 7.1 is practical: see [CNS99, HG98] for experimental results. In practice, the optimal choice of parameters depends very much on the implementation of the lattice reduction algorithm: rather than fix the bound X, and choose h and n accordingly, one should select the lattice rank n, and compute the theoretical bound X which is guaranteed. In order to find the value of n which offers the best trade-off between the running time and the size of the bound X, one should perform a series of experiments with existing implementations of lattice reduction algorithms.

7.3.2. The gcd generalization. Interestingly, Theorem 7.1 can be viewed as a particular case of the following gcd result:

THEOREM 7.3. Let  $P(x) \in \mathbb{Z}[x]$  be a monic polynomial of degree  $\delta$  in one variable, and let N be an integer of unknown factorization. Let  $\alpha \in \mathbb{Q}$  such that  $0 \leq \alpha \leq 1$ . Then one can find in time polynomial in  $(\log N, \delta)$  and the bit-size of  $\alpha$  all integers  $x_0$  such that  $gcd(P(x_0), N) \geq N^{\alpha}$  and  $|x_0| \leq N^{\alpha^2/\delta}$ .

Strictly speaking, Theorem 7.3 only appeared explicitly in [May03, May04] where it was attributed to Coppersmith. However, it was earlier presented in several workshop/summer school talks (such as SAC 2001), and could be considered as a folklore theorem: the result was implicit in [BDHG99, Bon00]; the particular case P(x) of degree 1 was stated and proved in [HG01], and the proof of [HG01] also works for the general case. Blömer and May [BM05, Cor. 14] proved a slightly different result where there are two modifications in the statement of Theorem 7.3: one replaces the assumption P(x) monic by the weaker assumption that the gcd of the coefficients of P(x) is coprime with N, and one replaces the property  $gcd(P(x_0), N) \ge N^{\alpha}$  by the stronger property:  $P(x_0) \ge N^{\alpha}$  and  $P(x_0)$  divides N.

Note that Theorem 7.1 is simply the case  $\alpha = 1$  in Theorem 7.3. By choosing different values of  $\alpha$ , one obtains interesting applications [**BDHG99**, **Bon00**, **HG01**, **CM04**]. To give a flavour of the applications, let us present two examples:

**Factoring with a hint:** Consider an RSA modulus N = pq where p and q have the same size. Assume that we know half of the most significant bits of p: for instance, one could imagine that half of the bits are given by the identity of the user, so that it would not be necessary to store them. Thus, we know an integer  $p_0$  such that  $p = p_0 + \varepsilon$  where  $\varepsilon$  is an unknown integer such that  $0 \le \varepsilon \lesssim N^{1/4}$ . Consider the polynomial  $P(x) = p_0 + x$ . Then  $gcd(P(\varepsilon), N) = p \gtrsim N^{1/2}$  with  $\varepsilon \le N^{1/4}$ . By applying Theorem 7.3 with  $\alpha = 1/2$ , we obtain  $\varepsilon$  and therefore factor N in polynomial time. Such a result was first proved by Coppersmith [Cop97]. but not using Theorem 7.3. Rather, Coppersmith [Cop97] applied an analogue of Theorem 7.1 to bivariate equations over the integers, which we discuss in Section 7.3.4. We assumed that the (half) unknown bits of p were the least significant bits of p: by tweaking the polynomial P(x), one can apply Theorem 7.3 to easily prove the more general result where the unknown bits of p are located at an arbitrary position (such as most significant bits, or middle bits), as while as they are all consecutive (not split among several blocks). More precisely, we may write  $p = p_0 + \varepsilon 2^k$ where  $p_0$  and k are known, which leads us to consider  $P(x) = cp_0 + x$ where c is chosen as the inverse of  $2^k$  modulo N.

**Factoring of**  $N = p^r q$ : Assume that  $N = p^r q$  where r is large, and p and q need not be prime. Assume that we know an approximation  $p_0$  of  $p : p = p_0 + \varepsilon$ . Consider the polynomial  $P(x) = (p_0 + x)^r$ . Then  $gcd(P(\varepsilon), N) = p^r$  is very large. By a careful application of Theorem 7.3, Boneh, Durfee and Howgrave-Graham [**BDHG99**] proved that all numbers  $N = p^r q$  where  $r > \log p$  and  $\log q = O(\log p)$  can be factored in time polynomial in  $\log N$ . In such a case, a sufficiently good approximation  $p_0$  of p can be found in polynomial time by brute force: because r is large, we do not need a very good approximation.

We will not give a complete proof of Theorem 7.3: see [HG01, May03] for more details. Rather, we will give the main argument, compared to Theorem 7.1. In Theorem 7.1, we used the fact that every sufficiently small integer was zero, in order to transform a polynomial congruence modulo N into a polynomial equation over  $\mathbb{Z}$ . Theorem 7.3 relies on the fact that every sufficiently small rational with bounded denominator must be zero. More precisely, the proof considers again an integral linear combination  $r(x) \in \mathbb{Q}[x]$  of the polynomials  $q_{u,v}(x) = x^u(P(x)/N)^v$ with the constraint  $0 \le v \le h$ . If the gcd of  $P(x_0)$  with N is  $\ge N^{\alpha}$ , then  $Q(x_0)$  is not necessarily an integer like in the proof of Theorem 7.1: However, the rational number  $Q(x_0)$  then has denominator  $\le N^{h(1-\alpha)}$ . Thus, this rational number is therefore zero if it is  $< 1/N^{h(1-\alpha)}$ . This still reduces the problem to finding short lattice vectors, but the proof is more technical: namely, because the bound on the short vector depends here on the parameter h, we need to find the right balance between all the parameters used by the algorithm.

7.3.3. Multivariate modular equations. Interestingly, Coppersmith [Cop97] noticed that Theorem 7.1 can be heuristically extended to multivariate polynomial modular equations. Assume for instance that one would like to find all small roots of  $P(x, y) \equiv 0 \pmod{N}$ , where P(x, y) has total degree  $\delta$  and has at least one monic monomial  $x^{\alpha}y^{\delta-\alpha}$  of maximal total degree. If one could obtain two algebraically independent integral polynomial equations satisfied by all sufficiently small modular roots (x, y), then one could compute (as resultant) a univariate integral polynomial equations, one can use an analogue of Lemma 7.2 to bivariate polynomials, with the (natural) notation  $||r(x, y)|| = \sqrt{\sum_{i,j} a_{i,j}^2}$  for  $r(x, y) = \sum_{i,j} a_{i,j} x^i y^j$ :

LEMMA 7.4. Let  $r(x,y) \in \mathbb{Q}[x,y]$  be a sum of at most w monomials. Assume  $||r(xX, yY)|| < 1/\sqrt{w}$  for some  $X, Y \ge 0$ . If  $r(x_0, y_0) \in \mathbb{Z}$  with  $|x_0| < X$  and  $|y_0| < Y$ , then  $r(x_0, y_0) = 0$  holds over the integers.

By analogy, one chooses a parameter h and select r(x, y) as a linear combination of the polynomials  $q_{u_1,u_2,v}(x,y) = x^{u_1}y^{u_2}(P(x,y)/N)^v$ , where  $u_1 + u_2 + \delta v \leq h\delta$ and  $u_1, u_2, v \geq 0$  with  $u_1 < \alpha$  or  $u_2 < \delta - \alpha$ . Such polynomials have total degree less than  $h\delta$ , and therefore are linear combinations of the  $n = (h\delta + 1)(h\delta + 2)/2$ monic monomials of total degree  $\leq \delta h$ . Due to the condition  $u_1 < \alpha$  or  $u_2 < \delta - \alpha$ , such polynomials are in bijective correspondence with the n monic monomials (associate to  $q_{u_1,u_2,v}(x,y)$  the monomial  $x^{u_1+v\alpha}y^{u_2+v(\delta-\alpha)}$ ). One can represent the polynomials as n-dimensional vectors in such a way that the  $n \times n$  matrix consisting of the  $q_{u_1,u_2,v}(xX,yY)$ 's (for some ordering) is lower triangular with coefficients  $N^{-v}X^{u_1+v\delta}y^{u_2+v(\delta-\alpha)}$  on the diagonal. Now consider the first two vectors  $r_1(xX, yY)$  and  $r_2(xX, yY)$  of an LLLreduced basis of the lattice spanned by the rows of that matrix. Since the rational  $q_{u_1,u_2,v}(x_0, y_0)$  is actually an integer for any root  $(x_0, y_0)$  of P(x, y) modulo N, we need  $||r_1(xX, yY)||$  and  $||r_2(xX, yY)||$  to be less than  $1/\sqrt{n}$  to apply Lemma 7.4. A (tedious) computation of the triangular matrix determinant enables to prove that  $r_1(x, y)$  and  $r_2(x, y)$  satisfy that bound when  $XY < N^{1/\delta-\varepsilon}$  and h is sufficiently large. Thus, one obtains two integer polynomial bivariate equations satisfied by all small modular roots of P(x, y).

The problem is that, although such polynomial equations are linearly independent as vectors, they might be algebraically dependent, making the method heuristic. This heuristic assumption is unusual: many lattice-based attacks are heuristic in the sense that they require traditional lattice reduction algorithms to behave like SVP-oracles. An important open problem is to find sufficient conditions to make Coppersmith's method provable for bivariate (or multivariate) equations: see [**BJ07**] for recent progress on this question. Note that the method cannot work all the time. For instance, the polynomial x - y has clearly too many roots over  $\mathbb{Z}^2$  and hence too many roots modulo any N (see [**Cop97**] for more general counterexamples).

Such a result may enable to prove several attacks which are for now, only heuristic. Indeed, there are applications to the security of the RSA encryption scheme when a very low public exponent or a low private exponent is used (see [Bon99] for a survey), and related schemes such as the KMOV cryptosystem (see [Ble97]). In particular, the experimental evidence of [BD99, Ble97, DN00] shows that the method is very effective in practice for certain polynomials.

Let us make a few remarks. In the case of univariate polynomials, there was basically no choice over the polynomials  $q_{u,v}(x) = x^u (P(x)/N)^v$  used to generate the appropriate univariate integer polynomial equation satisfied by all small modular roots. There is much more freedom with bivariate modular equations. Indeed, in the description above, we selected the indices of the polynomials  $q_{u_1,u_2,v}(x,y)$  in such a way that they corresponded to all the monomials of total degree  $\leq h\delta$ , which form a triangle in  $\mathbb{Z}^2$  when a monomial  $x^i y^j$  is represented by the point (i, j). This corresponds to the general case where a polynomial may have several monomials of maximal total degree. However, depending on the shape of the polynomial P(x, y)and the bounds X and Y, other regions of  $(u_1, u_2, v)$  might lead to better bounds.

and the bounds X and Y, other regions of  $(u_1, u_2, v)$  might lead to better bounds. Assume for instance P(x, y) is of the form  $x^{\delta_x}y^{\delta_y}$  plus a linear combination of  $x^i y^j$ 's where  $i \leq \delta_x$ ,  $j \leq \delta_y$  and  $i + j < \delta_x + \delta_y$ . Intuitively, it is better to select the  $(u_1, u_2, v)$ 's to cover the rectangle of sides  $h\delta_x$  and  $h\delta_y$  instead of the previous triangle, by picking all  $q_{u_1,u_2,v}(x, y)$  such that  $u_1 + v\delta_x \leq h\delta_x$  and  $u_2 + v\delta_y \leq h\delta_y$ , with  $u_1 < \delta_x$  or  $u_2 < \delta_y$ . One can show that the polynomials  $r_1(x, y)$  and  $r_2(x, y)$  obtained from the first two vectors of an LLL-reduced basis of the appropriate lattice satisfy Lemma 7.4, provided that h is sufficiently large, and the bounds satisfy  $X^{\delta_x}Y^{\delta_y} \leq N^{2/3-\varepsilon}$ . Boneh and Durfee [**BD99**] applied similar and other tricks to a polynomial of the form P(x, y) = xy + ax + b. This allowed better bounds than the generic bound, leading to improve attacks on RSA with low secret exponent (see also [**DN00**] for an extension to the trivariate case, useful when the RSA primes are unbalanced). More precisely, recall that the RSA exponents d and e are such that  $e \cdot d \equiv 1 \mod \phi(N)$ . Since  $\phi(N) = (p-1)(q-1) = N + 1 - p - q = N + 1 - s$ , where  $s = p + q \approx N^{1/2}$  there exists k such that  $e \cdot d + k(N + 1 - s) = 1$ . We obtain a bivariate polynomial congruence with unknowns k and s:

$$k(N+1-s) \equiv 1 \pmod{e},$$

which is of the form P(x, y) = xy + ax + b as mentioned previously. Here,  $s \approx N^{1/2}$  is relatively small. If *d* is small, then so will be the unknown integer *k*. By optimizing Coppersmith's technique to this polynomial, Boneh and Durfee [**BD99**] showed that one can heuristically factor the RSA modulus N = pq from the public key (N, e) if  $d \leq N^{1-1/\sqrt{2}} \approx N^{0.292}$ , which improved the bound  $N^{0.25}$  of Wiener [**Wie90**] (see Section 7.1.1). The bound can be improved if *p* and *q* are unbalanced [**DN00**].

7.3.4. Multivariate integer equations. The general problem of solving multivariate polynomial equations over  $\mathbb{Z}$  is also hard, as integer factorization is a special case. Coppersmith [**Cop97**] showed that a similar (albeit more technical) latticebased approach can be used to find small roots of bivariate polynomial equations over  $\mathbb{Z}$ :

THEOREM 7.5 (Coppersmith). Let P(x, y) be a polynomial in two variables over  $\mathbb{Z}$ , of maximum degree  $\delta$  in each variable separately, and assume the coefficients of P are relatively prime as a set. Let X, Y be bounds on the desired solutions  $x_0, y_0$ . Define  $\hat{P}(x, y) = P(Xx, Yy)$  and let D be the absolute value of the largest coefficient of  $\hat{P}$ . If  $XY < D^{2/(3\delta)}2^{-14\delta/3}$ , then in time polynomial in  $(\log D, \delta)$ , we can find all integer pairs  $(x_0, y_0)$  such that  $P(x_0, y_0) = 0, |x_0| < X$  and  $|y_0| < Y$ .

Again, the method extends heuristically to more than two variables, and there can be improved bounds depending on the shape<sup>2</sup> of the polynomial (see [**Cop97**]). Theorem 7.5 was introduced to factor in polynomial time an RSA-modulus N = pq provided that half of the (either least or most significant) bits of either p or q are known (see [**Cop97**, **Bon00**, **BDF98**]). This was sufficient to break an ID-based RSA encryption scheme proposed by Vanstone and Zuccherato [**VZ95**]. Boneh *et al.* [**BDF98**] provide another application, for recovering the RSA secret key when a large fraction of the bits of the secret exponent is known. However, none of the applications cited above happen to be "true" applications of Theorem 7.5: it was later realized in [**HG98**, **BDHG99**] that those results could alternatively be obtained from Theorem 7.3, which is the gcd generalization of Theorem 7.1.

The main idea of the proof of Theorem 7.5 is to find another bivariate integer polynomial equation satisfied by the small roots. Surprisingly, it is possible to do so using lattice reduction while making sure that this new equation is algebraically independent from the first equation. Then, by computing a resultant, and solving univariate polynomial equations over  $\mathbb{Z}$ , one can deduce all the small roots.

The original proof by Coppersmith can be found in [Cop97]. Coron [Cor07] found an alternative method inspired by Theorem 7.1. Blömer and May [BM05] showed a general method to adapt the bounds of Theorem 7.5 depending on the shape of the polynomial P(x, y). In particular, they showed that Theorem 7.1 can actually follow from Theorem 7.5. Surprisingly, Theorem 7.3 does not seem to follow from Theorem 7.5, though Blömer and May [BM05] are able to show that a result close to Theorem 7.3 can be viewed as a corollary of Theorem 7.5.

<sup>&</sup>lt;sup>2</sup>The coefficient 2/3 is natural from the remarks at the end of the previous section for the bivariate modular case. If we had assumed P to have total degree  $\delta$ , the bound would be  $XY < D^{1/\delta}$ .

# Acknowledgements

We would like to thank Steven Galbraith, Alexander May, Igor Shparlinski, Damien Stehlé and Frederik Vercauteren for helpful discussions and comments on drafts of these notes.

# References

[Adl83]	L. M. Adleman, On breaking generalized knapsack publick key cryptosystems, Proc. of 15th STOC, ACM, 1983, pp. 402–412.
[Ajt98]	M. Ajtai, The shortest vector problem in $L_2$ is NP-hard for randomized reductions, Proc. of 30th STOC, ACM, 1998, Available at [ECC] as TR97-047.
[Ajt02]	M. Ajtai, Random lattices and a conjectured 0-1 law about their polynomial time computable properties, Proc. of FOCS 2002, IEEE, 2002, pp. 13–39.
[Ajt06]	M. Ajtai, Generating random lattices according to the invariant distribution, Draft of March, 2006.
[AKS01]	M. Ajtai, R. Kumar, and D. Sivakumar, A sieve algorithm for the shortest lattice vector problem, Proc. 33rd STOC, ACM, 2001, pp. 601–610.
[Bab86]	L. Babai, On Lovász lattice reduction and the nearest lattice point problem, Combi- natorica <b>6</b> (1986), 1–13.
$[BBB^+]$	C. Batut, K. Belabas, D. Bernardi, H. Cohen, and M. Olivier, <i>PARI/GP computer package version 2</i> , Université de Bordeaux I.
[BBFK05]	F. Bahr, M. Boehm, J. Franke, and T. Kleinjung, <i>Factorization of RSA-200</i> , Public announcement on May 9th., 2005.
[BBS06]	Elad Barkan, Eli Biham, and Adi Shamir, <i>Rigorous bounds on cryptanalytic time/memory tradeoffs</i> , Advances in Cryptology – Proc. CRYPTO '06, Lecture Notes in Computer Science, vol. 4117, Springer, 2006, pp. 1–21.
[BCCN01]	E. Brier, C. Clavier, JS. Coron, and D. Naccache, <i>Cryptanalysis of RSA signatures with fixed-pattern padding.</i> , Proc. CRYPTO '01, LNCS, vol. 2139, IACR, Springer-Verlag, 2001, pp. 433–439
[BD99]	D. Boneh and G. Durfee, Cryptanalysis of RSA with private key d less than $N^{0.292}$ , Proc. of Eurocrypt '99, LNCS, vol. 1592, IACR, Springer-Verlag, 1999, pp. 1–11.
[BDF98]	D. Boneh, G. Durfee, and Y. Frankel, An attack on RSA given a small fraction of the private key bits, Proc. of Asiacrypt '98, LNCS, vol. 1514, Springer-Verlag, 1998, pp. 25–34.
[BDHG99]	D. Boneh, G. Durfee, and N. A. Howgrave-Graham, Factoring $n = p^r q$ for large r, Proc. of Crypto '99, LNCS, vol. 1666, IACR, Springer-Verlag, 1999.
[BDL97]	D. Boneh, R. A. DeMillo, and R. J. Lipton, On the importance of checking crypto- graphic protocols for faults, Proc. of Eurocrypt '97, LNCS, vol. 1233, IACR, Springer- Verlag, 1997, pp. 37–51.
[BJ07]	Aurélie Bauer and Antoine Joux, <i>Toward a rigorous variation of Coppersmith's algorithm on three variables</i> , Advances in Cryptology - Proc. EUROCRYPT '07, Lecture Notes in Computer Science, vol. 4515, Springer, 2007, pp. 361–378.
[BJN00]	D. Boneh, A. Joux, and P. Q. Nguyen, Why textbook ElGamal and RSA encryption are insecure, Proc. of Asiacrypt '00, LNCS, vol. 1976, IACR, Springer-Verlag, 2000.
[Ble96]	D. Bleichenbacher, Generating ElGamal signatures without knowing the secret key, Proc. of Eurocrypt '96, LNCS, vol. 1070, IACR, Springer-Verlag, 1996, Corrected version available from the author, pp. 10–18.
[Ble97]	, On the security of the KMOV public key cryptosystem, Proc. of Crypto '97, LNCS, vol. 1294, IACR, Springer-Verlag, 1997, pp. 235–248.
[Ble98]	, Chosen ciphertext attacks against protocols based on the RSA encryption standard PKCS #1, Proc. of Crypto '98, LNCS, vol. 1462, IACR, Springer-Verlag, 1998, pp. 1–12.
[BM05]	J. Blömer and A. May, A tool kit for finding small roots of bivariate polynomials over the integers., Advances in Cryptology - Proc. of EUROCRYPT 2005, Lecture Notes in Computer Science, vol. 3494, Springer, 2005, pp. 251–267.

- [BN00] D. Bleichenbacher and P. Q. Nguyen, Noisy polynomial interpolation and noisy Chinese remaindering, Proc. of Eurocrypt '00, LNCS, vol. 1807, IACR, Springer-Verlag, 2000.
- [Bon99] D. Boneh, Twenty years of attacks on the RSA cryptosystem, Notices of the AMS 46 (1999), no. 2, 203–213.
- [Bon00] \_\_\_\_\_, Finding smooth integers in short intervals using CRT decoding, Proc. of 32nd STOC, ACM, 2000.
- [Bon01] \_\_\_\_\_, Simplified OAEP for the RSA and Rabin functions, Proc. of Crypto '01, LNCS, IACR, Springer-Verlag, 2001.
- [BR95] M. Bellare and P. Rogaway, Optimal asymmetric encryption, Proc. of Eurocrypt '94, LNCS, vol. 950, IACR, Springer-Verlag, 1995, pp. 92–111.
- [BR96] \_\_\_\_\_\_, The exact security of digital signatures how to sign with RSA and Rabin, Proc. of Eurocrypt '96, LNCS, vol. 1070, IACR, Springer-Verlag, 1996, pp. 399–416.
   [BSS04] I. F. Blake, G. Seroussi, and N. Smart, Advances in elliptic curve cryptography, London Mathematical Society Lecture Note Series, vol. 317, Cambridge University Press, 2004.
- [BV96] D. Boneh and R. Venkatesan, Hardness of computing the most significant bits of secret keys in Diffie-Hellman and related schemes, Proc. of Crypto '96, LNCS, IACR, Springer-Verlag, 1996.
- [BV98] \_\_\_\_\_, Breaking RSA may not be equivalent to factoring, Proc. of Eurocrypt '98, LNCS, vol. 1233, Springer-Verlag, 1998, pp. 59–71.
- [CFPR96] D. Coppersmith, M. K. Franklin, J. Patarin, and M. K. Reiter, Low-exponent RSA with related messages, Proc. of Eurocrypt '96, LNCS, vol. 1070, IACR, Springer-Verlag, 1996, pp. 1–9.
- [CK04] H. Cohn and A. Kumar, The densest lattice in twenty-four dimensions, Electron. Res. Announc. Amer. Math. Soc. 10 (2004), 58–67 (electronic).
- [CM04] J.-S. Coron and A. May, Deterministic polynomial time equivalence of computing the RSA secret key and factoring, Cryptology ePrint Archive: Report 2004/208. Appeared in J. Cryptology, 2007., 2004.
- [CNS99] C. Coupé, P. Q. Nguyen, and J. Stern, The effectiveness of lattice attacks against low-exponent RSA, Proc. of PKC'99, LNCS, vol. 1431, Springer-Verlag, 1999.
- [CNS02] D. Catalano, P. Q. Nguyen, and J. Stern, The hardness of Hensel lifting: the case of RSA and discrete logarithm, Proc. of Asiacrypt '02, LNCS, vol. 2501, IACR, Springer-Verlag, 2002, pp. 299–310.
- [Cop97] D. Coppersmith, Small solutions to polynomial equations, and low exponent RSA vulnerabilities, J. of Cryptology 10 (1997), no. 4, 233–260, Revised version of two articles from Eurocrypt '96.
- [Cop01] \_\_\_\_\_, Finding small solutions to small degree polynomials, Proc. of CALC '01, LNCS, Springer-Verlag, 2001.
- [Cor07] J.-S. Coron, Finding small roots of bivariate integer polynomial equations: A direct approach, Advances in Cryptology - Proc. of CRYPTO 2007, Lecture Notes in Computer Science, vol. 4622, Springer, 2007, pp. 379–394.
- [CP01] R. Crandall and C. Pomerance, Prime numbers a computational perspective, Springer-Verlag, 2001.
- [CS98] J.H. Conway and N.J.A. Sloane, Sphere packings, lattices and groups, Springer-Verlag, 1998, Third edition.
- [DFSS07] V. Dubois, P.-A. Fouque, A. Shamir, and J. Stern, *Practical cryptanalysis of SFLASH*, Advances in Cryptology Proceedings of CRYPTO '07, LNCS, vol. 4622, Springer-Verlag, 2007.
- [DH76] W. Diffie and M. E. Hellman, New directions in cryptography, IEEE Trans. Inform. Theory IT-22 (1976), 644–654.
- [DN00] G. Durfee and P. Q. Nguyen, Cryptanalysis of the RSA schemes with short secret exponent from Asiacrypt '99, Proc. of Asiacrypt '00, LNCS, vol. 1976, IACR, Springer-Verlag, 2000.
- [ECC] ECCC, http://www.eccc.uni-trier.de/eccc/, The Electronic Colloquium on Computational Complexity.
- [El 85] T. El Gamal, A public key cryptosystem and a signature scheme based on discrete logarithms, IEEE Trans. Inform. Theory 31 (1985), 469–472.

PHONG	Q.	NGUYEN
-------	----	--------

[Emd81] [FJ03]	P. van Emde Boas, Another NP-complete problem and the complexity of computing short vectors in a lattice, Tech. report, Mathematische Instituut, University of Am- sterdam, 1981, Report 81-04. Available at http://turing.wins.uva.nl/~peter/. J.C. Faugère and A. Joux, Algebraic cryptanalysis of hidden field equation (HFE)
	cryptosystems using Gröbner bases, Proc. of Crypto '03, LNCS, vol. 2729, Springer-Verlag, 2003, pp. 44–60.
[FOPS01]	E. Fujisaki, T. Okamoto, D. Pointcheval, and J. Stern, <i>RSA–OAEP is secure under the RSA assumption</i> , Proc. of Crypto '01, LNCS, IACR, Springer-Verlag, 2001.
[GG98]	O. Goldreich and S. Goldwasser, On the limits of non-approximability of lattice problems, Proc. of 30th STOC, ACM, 1998, Available at [ECC] as TR97-031.
[GGH97]	O. Goldreich, S. Goldwasser, and S. Halevi, <i>Public-key cryptosystems from lattice reduction problems</i> , Proc. of Crypto '97, LNCS, vol. 1294, IACR, Springer-Verlag, 1997, Available at [ <b>ECC</b> ] as TR96-056., pp. 112–131.
[GHGKN06]	N. Gama, N. Howgrave-Graham, H. Koy, and P. Q. Nguyen, <i>Rankin's constant and blockwise lattice reduction</i> , Proc. of Crypto '06, LNCS, vol. 4117, Springer-Verlag, 2006, pp. 112–130.
[GLS93]	M. Grötschel, L. Lovász, and A. Schrijver, <i>Geometric algorithms and combinatorial optimization</i> , Springer-Verlag, 1993.
[GM03]	D. Goldstein and A. Mayer, On the equidistribution of Hecke points, Forum Mathematicum <b>15</b> (2003), 165–189.
[GMSS99]	O. Goldreich, D. Micciancio, S. Safra, and JP. Seifert, <i>Approximating shortest lattice vectors is not harder than approximating closest lattice vectors</i> , Information Processing Letters <b>71</b> (1999), 55–61, Available at [ <b>ECC</b> ] as TR99-002.
[GN08a]	N. Gama and P. Q. Nguyen, <i>Finding short lattice vectors within Mordell's inequality</i> , STOC '08 – Proc. 40th ACM Symposium on the Theory of Computing, ACM, 2008.
[GN08b]	, <i>Predicting lattice reduction</i> , Advances in Cryptology – Proc. EUROCRYPT '08, Lecture Notes in Computer Science, Springer, 2008.
[GPG]	GPG The GNU privacy guard http://www.gnung.org
[Hås88]	I Histad Solving simultaneous modular equations of low degree SIAM I Comput
[IIIdb000]	17 (1088) no 2 336 341 Preliminary version in Proc. of Crupto '85
[HDdL00]	<ul> <li>R. Harley, D. Doligez, D. de Rauglaudre, and X. Leroy, <i>Ecc2k-108 challenge solved</i>,</li> <li>Public announcement on April 4th 2000</li> </ul>
[Hel80]	M. E. Hellman, A cryptanalytic time-memory tradeoff, IEEE Trans. Inform. Theory 26 (1980), 401–406.
[Hel85]	B. Helfrich, Algorithms to construct Minkowski reduced and Hermite reduced bases, Theoretical Computer Science <b>41</b> (1985), 125–139.
[Her50]	C. Hermite, Extraits de lettres de M. Hermite à M. Jacobi sur différents objets de la théorie des nombres, deuxième lettre, J. Reine Angew. Math. <b>40</b> (1850), 279–290, Also available in the first volume of Hermite's complete works, published by Gauthier-Villars.
[HG97]	N. A. Howgrave-Graham, Finding small roots of univariate modular equations re- visited, Cryptography and Coding, LNCS, vol. 1355, Springer-Verlag, 1997, pp. 131– 142.
[HG98]	, Computational mathematics inspired by RSA, Ph.D. thesis, University of Bath, 1998.
[HG01]	, Approximate integer common divisors, Proc. of CALC '01, LNCS, Springer-Verlag, 2001.
[HPS98]	J. Hoffstein, J. Pipher, and J.H. Silverman, <i>NTRU: a ring based public key cryp-</i> tosystem, Proc. of ANTS III, LNCS, vol. 1423, Springer-Verlag, 1998, Additional information at http://www.ntru.com.pp. 267-288
[HS07]	Guillaume Hanrot and Damien Stehle, <i>Improved analysis of kannan's shortest lattice vector algorithm</i> , Advances in Cryptology - Proc. CRYPTO 2007, Lecture Notes in Computer Science, vol. 4622, Springer, 2007, pp. 170–186.
[HS08]	Worst-case hermite-korkine-zolotarev reduced lattice bases, CoRR abs/0801.3331 (2008).
[IAC04]	IACR, Advances in cryptology 1998–2003, Springer-Verlag, 2004, Electronic Proceedings of the Eurocrypt, Crypto, Asiacrypt, FSE and PKC Conferences.

[JM07]	Ellen Jochemsz and Alexander May, A polynomial time attack on RSA with private $CRT$ -exponents smaller than $0.073$ , Advances in Cryptology – Proc. CRYPTO '07,
[JvzGG03]	Lecture Notes in Computer Science, vol. 4622, Springer, 2007, pp. 395–411. Joachim <sup>4</sup> J. von zur Gathen and J. Gerhard, <i>Modern computer algebra</i> , second ed., Cambridge University Press, Cambridge, 2003.
[Kan83]	R. Kannan, Improved algorithms for integer programming and related lattice prob- lems, Proc. of 15th STOC, ACM, 1983, pp. 193–206.
[Kan87a]	, Algorithmic geometry of numbers, Annual review of computer science <b>2</b> (1987), 231–267.
[Kan87b]	, Minkowski's convex body theorem and integer programming, Math. Oper. Res. <b>12</b> (1987), no. 3, 415–440.
[Kle07]	T. Kleinjung, Discrete logarithms in $GF(p) - 160$ digits, Public announcement on Feb. 5th., 2007.
[Kob87]	N. Koblitz, Elliptic curve cryptosystems, Math. Comp. 48 (1987), 203–209.
[Kob98]	, Algebraic aspects of cryptography, Springer-Verlag, 1998.
[Koc96]	P. C. Kocher, <i>Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems</i> , Proc. of Crypto '96, LNCS, vol. 1109, IACR, Springer-Verlag, 1996, pp. 104–113.
[KS94]	S. V. Konyagin and T. Seger, <i>On polynomial congruences</i> , Mathematical Notes <b>55</b> (1994), no. 6, 596–600.
[KZ73]	A. Korkine and G. Zolotareff, Sur les formes quadratiques, Math. Ann. 6 (1873), 336–389.
[Lab]	RSA Laboratories, <i>The public-key cryptography standards (PKCS)</i> , Available at
[Len87]	H. W. Lenstra, Jr., Factoring integers with elliptic curves, Ann. of Math. <b>126</b> (1987), 649–673.
[LLL82]	A. K. Lenstra, H. W. Lenstra, Jr., and L. Lovász, <i>Factoring polynomials with rational coefficients</i> , Mathematische Ann. <b>261</b> (1982), 513–534.
[LS02]	A. K. Lenstra and I. Shparlinski, Selective forgery of RSA signatures with fixed-
	pattern padding., Proc. of PKC '02, Lecture Notes in Computer Science, vol. 2274, Springer, 2002, pp. 228–236.
[Mar03]	J. Martinet, <i>Perfect lattices in Euclidean spaces</i> , Grundlehren der Mathematischen Wissenschaften, vol. 327, Springer-Verlag, Berlin, 2003.
[May03]	A. May, New RSA vulnerabilities using lattice reduction methods, Ph.D. thesis, University of Paderborn, 2003.
[May04]	, Secret exponent attacks on RSA-type schemes with moduli $n = p^r q$ , Public
	Key Cryptography – Proc. of PKC 2004, Lecture Notes in Computer Science, vol. 2947, Springer, 2004, pp. 218–230.
[McE78]	R. J. McEliece, A public-key cryptosystem based on algebraic number theory, Tech. report, Jet Propulsion Laboratory, 1978, DSN Progress Report 42-44.
[Men08]	A. Menezes, An introduction to pairing-based cryptography, AMS, 2008, In this book.
[MG02]	D. Micciancio and S. Goldwasser, <i>Complexity of lattice problems: A cryptographic perspective</i> , Kluwer Academic Publishers, 2002.
[MH73]	J. Milnor and D. Husemoller, Symmetric bilinear forms, Springer-Verlag, 1973.
[MH78]	R. Merkle and M. Hellman, <i>Hiding information and signatures in trapdoor knapsacks</i> , IEEE Trans. Inform. Theory <b>IT-24</b> (1978), 525–530
[MI88]	T. Matsumoto and H. Imai, Public quadratic polynominal-tuples for efficient
	signature-verification and message-encryption, Proc. of Eurocrypt '88, LNCS, vol.
	330, Springer-Verlag, 1988, pp. 419–453.
[Mic98]	D. Micciancio, On the hardness of the shortest vector problem, Ph.D. thesis, Mas-
[Mic01]	The hardness of the closest sector problem with proprocessing IEEE Trops
[1011001]	Inform. Theory 47 (2001), no. 3, 1212–1215
[Mil87]	V. Miller, Use of elliptic curves in cryptography, Proc. of Crypto '85, LNCS, vol.
	218, IACR, Springer-Verlag, 1987, pp. 417–426.
[Min96]	H. Minkowski, Geometrie der Zahlen, Teubner-Verlag, Leipzig, 1896.
[MOV93]	A. Menezes, T. Okamoto, and S. A. Vanstone, <i>Reducing elliptic curve logarithms to logarithms in a finite field</i> , IEEE Trans. Inform. Theory <b>39</b> (1993), no. 5, 1639–1646.

#### PHONG Q. NGUYEN

[MOV97] A. Menezes, P. Van Oorschot, and S. Vanstone, Handbook of applied cryptography, CRC Press, 1997, Freely available on the Internet. [MZ98] K. S. McCurley and C. D. Ziegler, Advances in cryptology 1981-1997, Springer-Verlag, 1998, Electronic Proceedings of the Eurocrypt and Crypto Conferences. [Nat94] National Institute of Standards and Technology (NIST), FIPS publication 186: Digital signature standard, May 1994. [Ngu99] P. Q. Nguyen, Cryptanalysis of the Goldreich-Goldwasser-Halevi cryptosystem from Crypto '97, Proc. of Crypto '99, LNCS, vol. 1666, IACR, Springer-Verlag, 1999, pp. 288–304. [Ngu04] , Can we trust cryptographic software? Cryptographic flaws in GNU Privacy Guard v1.2.3., Advances in Cryptology - Proc. EUROCRYPT 2004, Lecture Notes in Computer Science, vol. 3027, Springer, 2004, pp. 555-570. [NS01] P. Q. Nguyen and J. Stern, The two faces of lattices in cryptology, Cryptography and Lattices - Proc. CALC '01, LNCS, vol. 2146, Springer-Verlag, 2001, pp. 146-180. [NS02] P. Q. Nguyen and I. E. Shparlinski, The insecurity of the digital signature algorithm with partially known nonces., J. Cryptology 15 (2002), no. 3, 151-176. [NS03] ., The insecurity of the elliptic curve digital signature algorithm with partially known nonces., Des. Codes Cryptography 30 (2003), no. 2, 201-217. [NS04] P. Q. Nguyen and D. Stehlé, Low-dimensional lattice basis reduction revisited (extended abstract), Proc. of the 6th Algorithmic Number Theory Symposium (ANTS VI), LNCS, vol. 3076, Springer-Verlag, 2004, pp. 338-357. [NS05a], Floating-point LLL revisited, Proc. of Eurocrypt 2005, LNCS, vol. 3494, Springer-Verlag, 2005, pp. 215-233. [NS05b] Phong Q. Nguyen and Jacques Stern, Adapting density attacks to low-weight knapsacks, Advances in Cryptology – Proceedings of ASIACRYPT '05, LNCS, vol. 3788, Springer-Verlag, 2005, pp. 41–58. [NS06] P. Q. Nguyen and D. Stehlé, LLL on the average, Proc. of ANTS-VII, LNCS, vol. 4076, Springer-Verlag, 2006. [NV08] P. Q. Nguyen and T. Vidick, Sieve algorithms for the shortest vector problem are practical, J. of Mathematical Cryptology (2008), To appear. [Odl90] A. M. Odlyzko, The rise and fall of knapsack cryptosystems, Cryptology and Computational Number Theory, Proc. of Symposia in Applied Mathematics, vol. 42, A.M.S., 1990, pp. 75-88. [Oec03] Philippe Oechslin, Making a faster cryptanalytic time-memory trade-off, Advances in Cryptology – Proc. CRYPTO '03, Lecture Notes in Computer Science, vol. 2729, Springer, 2003, pp. 617-630. [Poi05] D. Pointcheval, Provable security for public key schemes, pp. 133-185, Birkhäuser Verlag. 2005. [PS96] D. Pointcheval and J. Stern, Security proofs for signature schemes, Proc. of Eurocrypt '96, LNCS, vol. 1070, IACR, Springer-Verlag, 1996, pp. 387-398. [QL00] G. Qiao and K.-Y. Lam, RSA signature algorithm for microcontroller implementation., Proc. of CARDIS '98, Lecture Notes in Computer Science, vol. 1820, Springer-Verlag, 2000, pp. 353-356. [RSA78] R. L. Rivest, A. Shamir, and L. M. Adleman, A method for obtaining digital signatures and public-key cryptosystems, Communications of the ACM 21 (1978), no. 2, 120 - 126.[Sch87] C. P. Schnorr, A hierarchy of polynomial lattice basis reduction algorithms, Theoretical Computer Science 53 (1987), 201-224. [Sha82]A. Shamir, A polynomial time algorithm for breaking the basic Merkle-Hellman cryptosystem, Proc. of 23rd FOCS, IEEE, 1982, pp. 145-152. [Sho] Library (NTL), V Shoup, NumberTheoryC++Available at http://www.shoup.net/ntl/. [Sho99] Peter W. Shor, Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer, SIAM Rev. 41 (1999), no. 2, 303-332 (electronic). MR MR1684546 (2000e:11159) [Sho01] V. Shoup, OAEP reconsidered, Proc. of Crypto '01, LNCS, IACR, Springer-Verlag, 2001.

52

# PUBLIC-KEY CRYPTANALYSIS

[Sho04]	, Sequences of games: a tool for taming complexity in security proofs, Cryp-
	tology ePrint Archive: Report 2004/332, 2004.
[Sho05]	, A computational introduction to number theory and algebra, Cambridge
	University Press, 2005, Also available on the Internet.
[Sie 45]	C. L. Siegel, A mean value theorem in geometry of numbers, Annals of Mathematics
	<b>46</b> (1945), no. 2, 340–347.
[Sie89]	C. L. Siegel, Lectures on the geometry of numbers, Springer-Verlag, 1989.
[Sti02]	D. R. Stinson, Some baby-step giant-step algorithms for the low hamming weight
	discrete logarithm problem., Math. Comput. 71 (2002), no. 237, 379–391.
[VGT88]	B. Vallée, M. Girault, and P. Toffin, How to guess $\ell$ -th roots modulo n by reducing
	lattice bases, Proc. of AAEEC-6, LNCS, vol. 357, Springer-Verlag, 1988, pp. 427–442.
[VZ95]	S. A. Vanstone and R. J. Zuccherato, Short RSA keys and their generation, J. of
	Cryptology 8 (1995), no. 2, 101–114.

[Wie90] M. Wiener, Cryptanalysis of short RSA secret exponents, IEEE Trans. Inform. Theory **36** (1990), no. 3, 553–558.

ENS & CNRS. École normale supérieure, département d'informatique; 45, rue d'Ulm, F–75005 Paris.

E-mail address: http://www.di.ens.fr/~pnguyen/