IST-2002-507932

ECRYPT

European Network of Excellence in Cryptology

Network of Excellence

Information Society Technologies

D.AZTEC.6
Hardness of the Main Computational Problems Used in Cryptography

Actual submission date: 14. March 2007

Start date of project: 1. February 2004 Duration: 4 years

Lead contractor: Katholieke Universiteit Leuven (KUL)

Revision 1.0

Project co-funded by the European Commission within the 6th Framework Programme		
Dissemination Level		
PU	Public	X
PP	Restricted to other programme participants (including the Commission services)	
RE	Restricted to a group specified by the consortium (including the Commission services)	
CO	Confidential, only for members of the consortium (including the Commission services)	

Hardness of the Main Computational Problems Used in Cryptography

Editors: Jean-Sébastien Coron (Gemplus), Benne de Weger (TUE)

Contributors: Daniel Augot (INRIA), Andreas Enge (LIX),
Marc Girault (FT-R&D), Louis Goubin (Axalto), Ellen Jochemsz (TUE),
Tanja Lange (RUB), Phong Nguyen (ENS), Christine Priplata (EDIZONE),
Nicolas Sendrier (INRIA), Colin Stahlke (EDIZONE), Christopher Wolf (KUL).

14. March 2007
Revision 1.0

The work described in this report has in part been supported by the Commission of the European Communities through the IST program under contract IST-2002-507932. The information in this document is provided as is, and no warranty is given or implied that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability.

Contents

Abstract

Public-key cryptography is based on mathematical problems believed to be hard to solve. In this document, we evaluate the hardness of the main computational problems used in cryptography. For each identified problem, we provide the best algorithm known for solving it, and the corresponding performance records. This in turn enables us to recommend key-lengths for which solving the problem is infeasible using today's technology.

The newest results on Discrete Logarithms in finite fields have been incorporated.

Chapter 1

Introduction

By definition, there cannot be unconditional security for asymmetric cryptography: one needs to make computational assumptions, which are more or less well-defined and more or less plausible. The aim of this document is to study the hardness of computational problems which arise in asymmetric cryptography. This includes:

- Integer factorization, on which RSA is based.
- The RSA problem.
- Discrete logarithm in finite fields. Many cryptosystems have a security based on the discrete logarithm problem in finite fields, including DSA, torus-based cryptosystems and XTR.
- Discrete logarithm in elliptic curves.
- Lattice basis reduction, on which the security of the NTRU cryptosystem is based.
- Systems of multivariate polynomial equations over finite fields.
- Error-decoding codes.

Estimating the required key-size for public-key cryptosystems is harder than for good secret-key cryptosystems that can only be broken by a brute-force attack. First, the required security level must be determined. Then one considers the best algorithm known that breaks the cryptosystem and estimate its computational cost. This in turn enables one to determine the required key-size.

We consider various security levels: 64-bit security is broken. 80 to 90-bit security is on the edge of what is not doable today. 128-bit is safe today, and 256-bit should remain secure for a long time.

For each computational problem on which a given public-key cryptosystem is based, we recall the best existing algorithms that solve such problems, and study the relationships between those problems. For each identified problem, we use the following framework :

1. Problem statement.
2. Parameters of the problem.

3. Complexity class (with optionally the quantum complexity class).
4. Best algorithm known
5. Performance records.
6. Recommended key-length
7. Related open-problems.

Chapter 2

Integer factorization

2.1 Problem statement

Let $N > 1$ be an integer which is not a prime. Then there is an integer $1 < a < N$ such that a divides N. The problem is to find such a factor a.

Usually it is easy to show that N is composite by compositeness tests like Miller-Rabin (see [Co]). If this is not easy, then N might be prime which can be proved by primality tests like the Jacobi Sum Test or the Elliptic Curve Test (see [Co]).

Depending on the specific properties of the problem, there are different useful algorithms. The choice of an algorithm depends on many circumstances: the size of N, whether or not one expects N to have many factors or a small factor, if one needs just one factor or all of them, etc. Also if the main purpose is factoring RSA moduli, several algorithms are worth looking at, since some of them are intensely used for smoothness testing within other algorithms.

The hardest case of the problem is to factor a number N which is the product of two large primes of similar size. The hardness of that problem is crucial for some important asymmetric primitives (RSA) and protocols. We will focus on this problem here.

2.2 Parameters of the problem

The only parameter of the problem is the number N. If N has a small factor, one might use trial division, the Pollard-ρ-Method, the Pollard-$(p-1)$-Method resp. the Pollard-$(p+1)$-Method and the Elliptic Curve Method (ECM, see [Co]) together with varying optimizations. For these methods the relevant parameter is (the size of) the smallest factor. ECM has been used to find factors up to the size of 219 bit (see [ECM]). To find the larger factors of N, one should use other methods. The relevant parameter is then the size of N.

The Multiple Polynomial Quadratic Sieve (MPQS) is good for factoring integers up to 80 or 100 digits, larger numbers should better be factored with the General Number Field Sieve (GNFS). For performance reasons, implementations of GNFS for different sizes of N should be different.

2.3 Complexity class

An integer N can be factored in subexponential time with respect to its size. More precisely let

$$L_N(\alpha,\beta) := e^{(\beta+o(1))(\log N)^\alpha(\log\log N)^{1-\alpha}}.$$

Then for a suitable choice of parameters in GNFS, the complexity is heuristically

$$L_N\left(\frac{1}{3},\sqrt[3]{\frac{64}{9}}\right).$$

If N has a special form, e.g. $N = a^b + c$ with small numbers a, b and c, then two certain polynomials used in GNFS have small coefficients and therefore small values, which speeds up everything. The sieve is then called Special Number Field Sieve (SNFS). For a suitable choice of parameters in SNFS, the complexity is heuristically

$$L_N\left(\frac{1}{3},\sqrt[3]{\frac{32}{9}}\right).$$

The current record for SNFS is the factorization of $2^{821} + 2^{411} + 1$, which is a number with 248 digits (see [SNFS248]).

Using a quantum computer of size $O(\log N)$ qubits to factor N, one needs only $O((\log N)^3)$ operations. The quantum computer is used to find the order of some random x modulo N. It is still necessary to do some calculations classically.

2.4 Best algorithm known

Roughly speaking, the best algorithm known to factor integers of more than 100 digits is GNFS. The algorithm heavily depends on the choice of parameters. The hard parts of GNFS are the sieving step and the matrix step. Talking about more than 700 bit, it would probably be best to support general purpose computers with special hardware in these two steps.

We briefly report on the state of the art in special purpose hardware for factoring. A good summary of the latest results and proposals on special-purpose hardware for cryptanalytic attacks can be found in the contributions to SHARCS 2005 (see [SHW]) and in the ECRYPT deliverable on hardware crackers (see [ECR2]). For a list of suggested cryptanalytic devices with a strong focus on GNFS see also [Tro].

Before the sieving starts, one chooses two polynomials with certain properties and small coefficients. If N is of a special form, these polynomials can have exceptionally small coefficients; we then talk about SNFS. Otherwise one needs to extensively search for good polynomials.

The next step is the sieving which is the hardest part. Here it is important to find a good balance between the pure sieving and the cofactorization which can be done with ECM in special hardware (see [FKPPPSS]). Asymptotically it is best to extensively use ECM and hardly any sieving in the sieving step. For 1024-bit-numbers extensive sieving and less ECM is better. To do the sieving of 1024-bit-numbers there are proposals like TWIRL (see [ST]) which rely on many giant monolytic wafers. Cost estimates lie in the millions of US dollars, but they seem not to be realizable with today's technology. The proposal SHARK (see [FKPPPS]) focuses on realizable technology (based on a modular architecture using many small ASICs) and claims costs of 200 million US dollars. It uses lattice sieving which gives better results than line sieving (see [FK2]).

Similar problems arise for the matrix step which consists of solving a large matrix over $\mathbb{F}_2$. This can be done with the Block-Wiedemann algorithm. The expensive part consists of many matrix-vector multiplications. One needs lots of memory and efficient sorting. The latest proposal is "Scalable Hardware for Sparse Systems of Linear Equations" (see [GSST]). More sieving simplifies the matrix step.

Already now it seems possible to factor 1024-bit-numbers, only it is very expensive. Using general purpose computers without the support of special hardware, factoring 1024-bit-numbers costs thousands of millions of US dollars.

2.5 Performance records

The world record in factoring of Franke, Kleinjung et al. is RSA-200 (a 663-bit-number, see [FK3]). It used a software implementation of GNFS with lattice sieving and was done with several linux computers working in parallel and a cluster for solving the matrix. The time for the sieving corresponds to about 55 years on a single 2.2 GHz Opteron CPU (actual time was less, of course, since many computers worked in parallel). After simplifications, the matrix had a size of around 64 million times 64 million with around 11000 million non-zero entries. The matrix step was performed on a cluster of 80 2.2 GHz Opterons connected via a Gigabit network and took about 3 months.

Their previous record was reached one and a half years earlier (a 576-bit-number, see [FK1]). The amount of work for searching two good polynomials corresponds to less than a year on a 1 GHz PIII CPU. The time for the sieving corresponds to about 13 years on a 1 GHz PIII CPU. After simplifications, the matrix had a size of more than 14 million times 14 million with more than 3000 million non-zero entries. It was solved on an Alpha cluster with 64 processors running at 616 MHz. Each processor needed less than 300 MB RAM. Solving the matrix took 12 days.

2.6 Recommended key-length

Most cryptographic products currently use a key-length of 1024 bit for cryptosystems based on the difficulty to factor integers. With enough money, i.e. enough computing power and storage, these cryptosystems seem to be breakable.

It might be questionable to compare RSA key-lengths with symmetric key-lengths. But it is a reasonable way to give at least some feeling about security levels. Unfortunately such comparisons always have to be given before a machine has been built to crack a certain key-size. Therefore the basis for comparisons, namely estimates about the total costs for cracking, leaves a vast margin for opinions. If precise numbers are given, these should be interpreted as rough approximations. One might compare the security of 1024 bit RSA to (130 or) 160 bit ECC which is comparable to (70 or) 80 bit of a symmetric algorithm. Numbers in this range have been proposed e.g. in [LV]. In [ECR1] ECRYPT suggests the following numbers (security level

n means that $O(2^n)$ operations are needed by the best algorithm to break the system, usually this is the corresponding key-length of symmetric systems):

security level	48	56	64	80	112	128	160	192	256
RSA	480	640	816	1248	2432	3248	5312	7936	15424
ECC	96	112	128	160	224	256	320	384	512

Several countries suggest to use 2048 bit after the year 2010 (and 256 bit for symmetric algorithms). Up to now there seems to be no way to factor a 2048 bit RSA modulus (maybe corresponding to roughly 100 bit symmetric security).

Of course the key-length to be recommended depends on the importance of the data to be secured. Since recent estimates show that RSA-1024 is within reach, if the information is interesting enough, one should convert to 2048 or even 4096 bit RSA.

2.7 Related open problems

GNFS is also the most efficient algorithm to solve the discrete logarithm problem in $\mathbb{F}_p^*$ where p is prime (see Section 5). After adapting the parameters, a sieving machine built for factoring can also perform the sieving step for the discrete logarithm problem. The resulting matrix has to be solved modulo $p-1$ instead of modulo 2 which makes the matrix step harder. Therefore one should sieve more in order to get a smaller matrix for the matrix step.

Acknowledgements:

The eCrypt contributor for this section Christine Priplata (EDIZONE), Colin Stahlke (EDIZONE) and Tanja Lange (RUB).

Bibliography

[Ber1] D. J. Bernstein, *Circuits for Integer Factorization: A Proposal*, Manuscript, November 2001.
`http://cr.yp.to/papers.html#nfscircuit`

[Ber2] D. J. Bernstein, *Integer factorization: a progress report*, Talk at FoCM 2005.
`http://cr.yp.to/talks.html`

[Bre] R. P. Brent, *Recent Progress and Prospects for Integer Factorisation Algorithms*, COCOON 2000, LNCS **1858**, Springer, 2000, 3-22.

[Co] H. Cohen, *A Course in Computational Algebraic Number Field Theory*, Graduate Texts in Math. **138**, Springer, 1993.

[Cop] D. Coppersmith, *Solving Homogeneous Linear Equations over GF(2) via Block-Wiedemann Algorithm*, Mathematics of Computation 62(205), 1994, 333-350.

[ECM] P. Zimmermann, ECMNET, Website, 2005.
`http://loria.fr/~zimmerma/records/ecmnet.html`

[ECR1] ECRYPT NoE, *ECRYPT Yearly Report on Algorithms and Keysizes (2004)*, Document D.SPA.10, March 2005.

[ECR2] ECRYPT NoE, *ECYRPT Report on Hardware Crackers*, Document D.VAM.3, 2005.

[FK1] J. Franke, T. Kleinjung et al., *RSA*-576, E-mail announcement, 2003.
`http://www.crypto-world.com/announcements/rsa576.txt`

[FK2] J. Franke, T. Kleinjung, *Continued Fractions and Lattice Sieving*, in: Special-Purpose Hardware for Attacking Cryptographic Systems – SHARCS 2005, Paris, 2005.

[FK3] J. Franke, T. Kleinjung et al., *RSA*-200, E-mail announcement, 2005.
`http://www.crypto-world.com/announcements/rsa200.txt`

[FKPPPS] J. Franke, T. Kleinjung, C. Paar, J. Pelzl, C. Priplata, C. Stahlke, *SHARK — A Realizable Special Hardware Sieving Device for Factoring* 1024-*bit Integers*, in: Special-Purpose Hardware for Attacking Cryptographic Systems – SHARCS 2005, Paris, 2005, also to appear in Proc. CHES 2005.

[FKPPPSS] J. Franke, T. Kleinjung, C. Paar, J. Pelzl, C. Priplata, M. Šimka, C. Stahlke, *An Efficient Hardware Architecture for Factoring Integers with the Elliptic Curve Method*, in: Special-Purpose Hardware for Attacking Cryptographic Systems – SHARCS 2005, Paris, 2005.

[GLM] R. A. Golliver, A. K. Lenstra, K. S. McCurley, *Lattice sieving and trial division*, in: Algorithmic Number Theory (ed. by L. M. Adleman, M.-D. Huang), LNCS **877**, Springer, 1994, 18-27.

[Gor] D. M. Gordon, *Discrete logarithms in GF(p) using the number field sieve*, SIAM Journal on Discrete Mathematics 6(1), 1993, 124-138.

[GS] W. Geiselmann, R. Steinwandt, *Yet another sieving device*, CT-RSA 2004, LNCS **2964**, Springer, 2004, 278-291.

[GSST] W. Geiselmann, A. Shamir, R. Steinwandt, E. Tromer, *A systolic design for supporting Wiedemann's algorithm*, in: Special-Purpose Hardware for Attacking Cryptographic Systems – SHARCS 2005, Paris, 2005.

[Kle] T. Kleinjung, *On Polynomial Selection for the General Number Field Sieve*, 2004, to appear in Mathematics of Computation.

[LL] A.K. Lenstra and H.W. Lenstra, Jr. (eds.), *The Development of the Number Field Sieve*, Lecture Notes in Math. **1554**, Springer, 1993.

[LV] A.K. Lenstra and E.R. Verheul, *Selecting Cryptographic Key Sizes*, Journal of Cryptology **14** No. 4, 2001, 255-293.

[LTS+] A. K. Lenstra, E. Tromer, A. Shamir, W. Kortsmit, B. Dodson, J. Hughes, P. Leyland, *Factoring Estimates for a* 1024*-bit RSA Modulus*, in: Proc. ASIACRYPT 2003, LNCS **2894**, Springer, 2003, 55-74.

[SHW] *SHARCS 2005 – Workshop on Special-Purpose Hardware for Attacking Cryptographic Systems*, ENSTA Paris, Booklet, Februar 2005.
`http://www.sharcs.org`

[SNFS248] K. Aoki, Y.Kida, T. Shimoyama, Y.Sonoda, H. Ueda, *SNFS*-248, E-mail announcement, 2004.
`http://www.crypto-world.com/announcements/SNFS248.txt`

[ST] A. Shamir and E. Tromer, *Factoring Large Numbers with the TWIRL Device*, in: Proc. Crypto 2003, LNCS **2729**, Springer, 2003, 1–26.
`http://www.wisdom.weizmann.ac.il/~tromer/papers/twirl.ps.gz`

[Tro] E. Tromer, *Special-Purpose Cryptanalytic Devices*, Website on Special-Purpose Hardware for Cryptanalysis, 2005.
`http://www.wisdom.weizmann.ac.il/~tromer/cryptodev/`

[Wied] D. H. Wiedemann, *Solving Sparse Linear Equations over Finite Fields*, IEEE Transactions on Information Theory 32(1), 1986, 54-62.

Chapter 3

The RSA problem

3.1 Introduction

The RSA problem is the following: given a modulus N, product of two primes p and q, a public-exponent e, and an element $y \in \mathbb{Z}_N^*$, find x such that $y = x^e \mod N$.

The problem parameters are the following: the bit-size k of N, and the size of the public exponent e. In practice, one uses small public-exponent in order to speed up RSA encryption or RSA signature verification. The RSA with small public exponent in studied in more details in the next sub-section.

The best algorithm known for solving the RSA problem consists in factoring the modulus. Therefore, the recommended key-size are the same as for the factorization problem.

3.2 Low public exponent

3.2.1 Problem statement:

RSA key pair generation methods have both security and performance requirements. A public key operation (notably encryption or signature verification) consists of a modular exponentiation with the public variable e as exponent. Modular exponentiation is computationally intensive, and its complexity is proportional to both the bitsize and the Hamming weight of the exponent e. For this reason $e = 3, 17$ and 65537 are popular, as they are small and have Hamming weight only 2 (and moreover they are prime, hence with reasonable probability coprime to $\phi(N)$, which is a soundness requirement for RSA).

The question is what this means for the security of the key pair: does this special form of e help in breaking RSA, i.e. in factoring the modulus (key recovery) or in solving the eth root problem (message recovery)?

No attacks are known that take advantage of small public exponent in factoring the modulus. Indeed, such attacks seem unlikely, because an attacker who is given only a modulus can pick a public exponent at will (without knowing the corresponding private exponent or any property of it), and produce as many plaintext-ciphertext pairs as he wants.

The only known result relating the size (more precisely: the smoothness) of the public exponent e to factoring the modulus is that of Boneh and Venkatesan [BV]. They show that any algebraic polynomial time oracle algorithm that, given $N = pq$ and a cube root oracle modulo N, factors N, can be converted into a non-oracle factorization algorithm. This shows that a cube root (or other smooth eth root) algorithm does not help in factoring, and thus that message recovery with such a small public exponent might be easier than factoring. However, efficient cube root algorithms are not known.

There exist message recovery attacks that are effective for small public exponents, such as Håstad's broadcast attacks on linearly related messages [H], and Coppersmith's attacks on messages with partially known plaintext or small random padding [C]. These attacks can usually be thwarted by adding sufficient redundancy (random padding) to the plaintext.

We can formulate the following hardness problems:

- The hardness of factoring the modulus N does not depend on properties of the public exponent e, as long as no knowledge on the private exponent is available.
- The hardness of message recovery from knowledge of a number of known ciphertexts.
- The hardness of extracting eth roots modulo N, for small $e > 2$, is as difficult as for arbitrary e.

3.2.2 Parameters of the problem:

The parameters of the problem are the bitsize k of the modulus N, the bitsize of the public exponent e, the Hamming weight h of the public exponent e, and the number of known ciphertexts corresponding to related plaintexts.

3.2.3 Complexity class:

See factoring.

3.2.4 Best algorithm known:

See factoring.

3.2.5 Performance record:

See factoring.

3.2.6 Recommended key-length:

See factoring.

3.2.7 Related open problems:

See factoring.

3.3 Low private exponent

3.3.1 Problem statement:

As described in the section above on low public exponent RSA, there may be performance reasons for allowing a private key with a private exponent d that is substantially smaller than the modulus n, and / or with small Hamming weight.

The first type of attack to worry about is brute force search. This shows that the bitsize of d should be at least 80. When d has bitsize > 256 then a lower bound for its Hamming weight is 18 only, and when d has bitsize ≥ 1024 then a lower bound for its Hamming weight is 13 only. The reason is that $\binom{n-2}{16} > 2^{80}$ when $n > 256$, and $\binom{n-2}{11} > 2^{80}$ when $n \geq 1024$ (we subtract 2 from the bitsize and the Hamming weight, because the most and least significant bits of d will be 1 typically).

We are not aware of other attacks on low Hamming weight private exponents. But there exist more clever attacks on low private exponents. The best one to date is that of Boneh and Durfee [BD]. It shows that there is a polynomial time attack on low private exponent RSA when $d = O(N^\alpha)$ for $\alpha = 1 - \frac{1}{2}\sqrt{2} = 0.292\ldots$. Boneh and Durfee give as a heuristic that RSA might be vulnerable for α up to $\frac{1}{2}$.

Often the CRT-variant of RSA is used to speed up private key operations. This means that next to the private exponent d the prime factors p, q of the modulus are kept as part of the private key; the modular exponentiation is done separately modulo p and modulo q, and the results are then combined into the final answer modulo N by Chinese remaindering. Asymptotically this gives a speedup factor 4 for full sized d. It is possible to achieve further speedups by choosing small sized $d_p = d \bmod p - 1$ and $d_q = d \bmod q - 1$, or low Hamming weight d_p, d_q. There exist attacks on low private CRT-exponents (see [M]), but they work only in the case of unbalanced modulus factors.

We formulate the following hardness problems:

- The hardness of factoring a k-bit modulus N in the case of a private exponent d that is in bitsize between $\frac{1}{2}k$ and k.
- The hardness of factoring the modulus N in the case of a low Hamming weight private exponent d (say Hamming weight between 20 and half the bitsize of d).
- The hardness of factoring the modulus N in the case of low private CRT exponents and balanced prime factors po, q.

3.3.2 Parameters of the problem:

The bitsize of the modulus N, and the bitsize and Hamming weight of the private exponent d.

3.3.3 Complexity class:

Polynomial when $d = O(N^{0.292})$, otherwise see factoring.

3.3.4 Best algorithm known:

Coppersmith's method for finding small roots of multivariate polynomials, as employed by Boneh and Durfee [BD].

3.3.5 Performance record:

For private keys up to $N^{0.28}$ for 1000-bit RSA the Boneh-Durfee method has been demonstrated in practice.

3.3.6 Recommended key-length:

Take the private exponent $d \gg N^{1/2}$, and with Hamming weight > 20.

3.3.7 Related open problems:

-

3.4 Partial Key Exposure

3.4.1 Problem statement:

When an attacker has enough partial information about the private key, he may be able to recover the full private key. Several attacks exist, depending on the bitsizes of d and e relative to the modulus bitsize, and on the amount and position of the known bits. Generally speaking, the attacks only work when the unknown part of d consists of consecutive bits, i.e. the known part of d is a combination of least and most significant bits.

The best partial key exposure attacks so far are those of Boneh, Durfee, and Frankel [BDF] (who describe several partial key exposure attacks for $e < N^{\frac{1}{2}}$), Blömer and May [BM] (who extend the attacks of [BDF] to methods that work up to $e < N^{0.725}$), and Ernst, Jochemsz, May, and de Weger (who show partial key exposure attacks exist if either e or d is chosen significantly smaller than N).

The case where both d and e are full size seems not to be vulnerable.

3.4.2 Parameters of the problem:

Size of the modulus N, size of the public exponent e, size of the private exponent d, the amount of known most significant bits of d, the amount of known least significant bits of d.

3.4.3 Complexity class:

Polynomial time within the attack boundaries, otherwise see factoring.

3.4.4 Best algorithm known:

Most partial key exposure attacks use Coppersmith's method for finding small modular roots or small integer roots. The attack that is best, depends on the choice of the size of e and d, as can be seen in the overview of the currently known attacks in [EJMW].

3.4.5 Performance record:

Since Coppersmith-like attacks result in asymptotic bounds, it is important to see what one can obtain in practice. Both [BDF], [BM], and [EJMW] provide experimental results. In [BDF], it is mentioned that for performance, it is sometimes better to implement an extension of the Lattice Factoring Method by Boneh, Durfee, and Howgrave-Graham [BDH].

3.4.6 Recommended key-length:

Not relevant. To prevent from this kind of attacks, it is recommended that e and d are chosen at random (and are thus full size), besides taking sufficient countermeasures against side-channel attacks.

3.4.7 Related open problems:

-

3.5 RSA assumptions

In the following, we study in more detail the relation between two RSA assumptions: the RSA one-wayness assumption, already defined above, which consists, given $y \in \mathbb{Z}_N$, in finding x such that $y = x^e \mod N$, and the RSA partial-domain one-wayness, which consists, given $y \in \mathbb{Z}_N$, in recovering some part of x. The latter assumption arises when proving the security of the OAEP public-key encryption scheme [FOPS].

3.5.1 Problem statement:

We denote by f the RSA permutation. The security of most RSA-based cryptosystem is based on the one-wayness of f, which is defined as follows:

- (τ, ε)**-one-wayness of** f, means that for any adversary $\mathcal{A}$ who wishes to recover the pre-image x of $f(x)$ in time lesser than τ, $\mathcal{A}$'s success probability $\mathsf{Succ}^{\mathsf{ow}}(\mathcal{A})$ is upper-bounded by ε:

$$\mathsf{Succ}^{\mathsf{ow}}(\mathcal{A}) = \Pr_x[\mathcal{A}(f(x)) = x] < \varepsilon$$

The partial-domain one-wayness and the set partial-domain one-wayness of a permutation f was defined in [FOPS] :

- (τ, ε)**-partial-domain one-wayness of** f, means that for any adversary $\mathcal{A}$ who wishes to recover the partial pre-image ω of $f(\omega, s)$ in time lesser than τ, $\mathcal{A}$'s success probability $\mathsf{Succ}^{\mathsf{pd-ow}}(\mathcal{A})$ is upper-bounded by ε:

$$\mathsf{Succ}^{\mathsf{pd-ow}}(\mathcal{A}) = \Pr_{\omega,s}[\mathcal{A}(f(\omega, s)) = \omega] < \varepsilon$$

- $(\ell, \tau, \varepsilon)$**-set partial-domain one-wayness of** f, means that for any adversary $\mathcal{A}$ who wishes to output a set of ℓ elements which contains the partial pre-image ω of $f(\omega, s)$, in time lesser than τ, $\mathcal{A}$'s success probability $\mathsf{Succ}^{\mathsf{s-pd-ow}}(\mathcal{A})$ is upper-bounded by ε:

$$\mathsf{Succ}^{\mathsf{s-pd-ow}}(\mathcal{A}) = \Pr_{\omega,s}[\omega \in \mathcal{A}(f(\omega, s))] < \varepsilon$$

As in [FOPS], we denote by $\mathsf{Succ}^{\mathsf{ow}}(\tau)$, (resp. $\mathsf{Succ}^{\mathsf{pd-ow}}(\tau)$ and $\mathsf{Succ}^{\mathsf{s-pd-ow}}(\ell,\tau)$) the maximal success probability $\mathsf{Succ}^{\mathsf{ow}}(\mathcal{A})$, (resp. $\mathsf{Succ}^{\mathsf{pd-ow}}(\mathcal{A})$ and $\mathsf{Succ}^{\mathsf{s-pd-ow}}(\mathcal{A})$), over all adversaries whose running times are lesser than τ. For any τ and $\ell \geq 1$, we have:

$$\mathsf{Succ}^{\mathsf{s-pd-ow}}(\ell,\tau) \geq \mathsf{Succ}^{\mathsf{pd-ow}}(\tau) \geq \mathsf{Succ}^{\mathsf{ow}}(\tau)$$

Moreover, by randomly selecting any element in the set returned by the adversary against the set partial-domain one-wayness, one can break the partial-domain one-wayness with probability $1/\ell$, which gives:

$$\mathsf{Succ}^{\mathsf{pd-ow}}(\tau) \geq \mathsf{Succ}^{\mathsf{s-pd-ow}}(\ell,\tau)/\ell \tag{3.1}$$

3.5.2 Parameters of the problem:

The parameters of the problem are : the modulus size k, the size $|e|$ of the public-exponent, and the size k_1 of the partial pre-image.

3.5.3 Complexity class:

For RSA, the three problems are polynomially equivalent. This has been shown in [FOPS] for the case $k_1 > k/2$ (the size of the partial pre-image is greater than half the size of the modulus). [FOPS] relies upon lattice reduction techniques for lattices of dimension 2. The extension to smaller k_1 is shown in [CJNP] and is based on using lattices of higher dimension.

Moreover, the work of Boneh and Venkatesan [BV] shows that breaking low-exponent RSA may not be equivalent to factoring integers. Namely, they show that an algebraic reduction from factoring to breaking low-exponent RSA can be converted into an efficient factoring algorithm.

3.5.4 Best algorithm known

The best algorithm known for inverting RSA, *i.e.* for breaking the one-wayness of RSA, is to factor the modulus.

The best algorithm known to break the one-wayness of RSA given access to an oracle which breaks the partial one-wayness is based on lattice reduction techniques, and is described in [FOPS, CJNP].

3.5.5 Performance record

The performance record for inverting RSA are the same as for factoring an RSA modulus.

3.5.6 Recommended key-kength

See the factoring integer section.

3.5.7 Related open problems

The strong-RSA assumption is defined as follows: given an RSA modulus N and given $y \in \mathbb{Z}_N$, find e, x with $e \geq 2$ such that $y = x^e \mod N$. A related open-problem is to show that the strong-RSA assumption is polynomially equivalent to the RSA assumption.

Acknowledgements:

The eCrypt contributors for this section are J.S. Coron (Gemplus), Marc Girault (FT), Ellen Jochemsz (TUE) and Benne de Weger (TUE).

Bibliography

[BM] Johannes Blömer, and Alexander May, New Partial Key Exposure Attacks on RSA, D. Boneh (Ed.): CRYPTO 2003, LNCS 2729, pp. 27-43, 2003.

[BD] Dan Boneh and Glenn Durfee, Cryptanalysis of RSA with Private Key d Less Than $N^{0.292}$, IEEE Transactions on Information Theory 46 [2000], 1339-1349.

[BDF] Dan Boneh, Glenn Durfee and Yair Frankel, An Attack on RSA given a Small Fraction of the Private Key Bits, Proceedings of ASIACRYPT 1998, LNCS 1514 [1998], 25-34.

[BDH] Dan Boneh, Glenn Durfee, and Nicholas Howgrave-Graham, Factoring $N = p^r q$ for Large r, Michael Wiener (Ed.): CRYPTO'99, LNCS 1666, pp. 326–337, 1999.

[BV] D. Boneh, and R. Venkatesan, Breaking RSA may not be equivalent to factoring, In Proceedings Eurocrypt '98, Lecture Notes in Computer Science, Vol. 1233, Springer-Verlag, pp. 59–71, 1998.

[C] Don Coppersmith, Small Solutions to Polynomial Equations and Low Exponent RSA Vulnerabilities, Journal of Cryptology 10 [1997], 233-260.

[CJNP] J.S. Coron, Marc Joye, David Naccache, Pascal Paillier. Universal Padding Schemes for RSA Lecture Notes In Computer Science; Vol. 2442 archive Proceedings of the 22nd Annual International Cryptology Conference on Advances in Cryptology. Pages: 226 - 241

[EJMW] Matthias Ernst, Ellen Jochemsz, Alexander May and Benne de Weger, Partial Key Exposure Attacks on RSA Up to Full Size Exponents, R. Cramer (Ed.): EUROCRYPT 2005, LNCS 3494, pp. 371-386, 2005.

[FOPS] Eiichiro Fujisaki, Tatsuaki Okamoto, David Pointcheval and Jacques Stern. RSA-OAEP is Secure under the RSA Assumption In Advances in Cryptology - Proceedings of CRYPTO '01 (august 19 - 23, 2001, Santa Barbara, California, USA) J. Kilian Ed., Pages 260-274, LNCS 2139, Springer-Verlag, 2001.

[H] J. Håstad, Solving simultaneous modular equations of low degree, SIAM Journal on computing, 17, pp. 336–341, 1988.

[M] Alexander May, Cryptanalysis of Unbalanced RSA with Small CRT-Exponent, Proceedings of CRYPTO 2002, LNCS 2442 [2002], 242-256.

Chapter 4

The general discrete logarithm problem

4.1 Problem statement and parameters

Given a finite cyclic group $(G, +)$ of order n, a generator P and a second element Q of the group, the discrete logarithm problem consists of finding an integer x such that $Q = xP$. This integer is in fact unique modulo n. In most cases relevant for cryptology, one may assume that the group order n is known. The algorithms in this section, however, are essentially based on exhaustive search, and can easily be modified to work even when the group order is not known, and may in fact be used to compute it.

More generally, one may consider the discrete logarithm problem in a non-cyclic group G, and given two elements P, Q, one may ask whether there exists an integer x such that $Q = xP$. This question, however, is not relevant to cryptology, and shall not be considered in the following. Suffice it to say that the following deterministic algorithms, due to their resemblance to exhaustive search, may be used to solve the discrete logarithm problem in this more general setting.

4.2 Parameters of the problem

In the most general and abstract setting, we do not make any assumptions on the particular representation of the group elements; so we assume that only the group operations are available to solve the problem:

Input: A finite cyclic group $(G, +)$, its order n and two bit strings representing an element P of order n of G and another element Q of G. The group operations are realised via calls to oracles that

- upon input of two bit strings representing group elements return a bit string representing their sum (addition); or
- upon input of a bit string representing a group element, returns a bit string representing its negative (inverse); or

- upon input of two bit strings representing group elements, returns the information whether these two group elements are actually the same (comparison).

In a slightly more restrictive model, satisfied by all groups suggested for cryptography, one may assume that each group element is represented by a unique bit string, and thus may dispense with the comparison oracle.

Output: The integer $x \in \{0, \dots, n-1\}$ such that $Q = xP$.

4.3 Complexity class

Letting p denote the largest prime factor of n, Nechaev [34] and Shoup [43] have shown that the discrete logarithm problem requires $\Omega(\sqrt{p})$ oracle calls even if group elements are represented by unique bit strings. Matching algorithms are known (see the following section), so that this bound is actually tight. I think that a simple modification of Shoup's argument shows that the optimal algorithm in the case that a comparison oracle is required is given by exhaustive search with a complexity of $\Theta(p)$ oracle calls.

4.4 Best algorithm known

A simple argument [35] shows that if the prime factorisation of $n = \prod p_i^{e_i}$ is known, then the discrete logarithm problem in G may be reduced to a series of discrete logarithm problems in its subgroups of order p_i: Chinese remaindering performs the reduction to the subgroups of coprime order $p_i^{e_i}$, and a Hensel lifting argument shows that the discrete logarithm in the subgroup of order p^k may be obtained from discrete logarithms in the subgroups of orders p^{k-1} and p. Induction concludes the argument.

We assume from now on that the group elements are represented by unique bit strings and that no comparison oracle is needed; n may or may not be prime. There are essentially two algorithms achieving a complexity of $\Theta(\sqrt{n})$ group operations and $\Theta(\sqrt{n} \log n)$ comparisons of bit strings.

Shanks's algorithm [42] is deterministic and has a space complexity of $\Theta(\sqrt{n})$ group elements. It proceeds in two phases. Let $r = \lceil \sqrt{n-1} \rceil$. In the *baby step* phase, the multiples iP for $i = 0, \dots, r$ are computed and sorted (this is where the uniqueness of the representation of the group elements is crucial). In the *giant step* phase, $R = rP$ and the $Q + jR$ for $j \geq 0$ are computed. If an element $Q + jR$ is found in the list of baby steps, then $(x + jr)P = Q + jR = iP$, and $x = i - jr \bmod n$. Such a match occurs after at most r giant steps.

Another kind of algorithms, suggested in [36] and known under the names of “Pollard ρ”, “Pollard λ” or “tame and wild kangaroos”, is probabilistic. In its simplest version, random linear combinations $a_i P + b_i Q$ with a_i, $b_i \in \{0, \dots, n-1\}$ are formed until the same group element is encountered twice; in that case, $a_i P + b_i Q = a_j P + b_j Q$ implies $a_i - a_j = -(b_i - b_j)x \bmod n$. If $b_i - b_j$ is coprime with n, which happens with high probability, then $x = -(b_i - b_j)^{-1}(a_i - a_j) \bmod n$. The birthday paradox ensures that such a collision happens on average after $O(\sqrt{n})$ steps.

The interest of the randomised algorithms lies in the fact that properly implemented (replacing random combination by pseudorandom walks, for instance), they dispense with the need for storage and work essentially with constant memory. The running time analysis becomes heuristic, but appears to be verified in practice. Moreover, the algorithms are arbitrarily parallelisable, with essentially no communication overhead.

4.5 Performance records

The generic algorithms have been implemented in the elliptic curve setting, see Section 6.4 for details.

4.6 Recommended key lengths

For a security level of 2^n, a key length of $2n$ bits is required.

Chapter 5

Discrete Logarithms in finite fields

5.1 Parameters of the problem

The discrete logarithm problem in finite fields is the specialisation of the general discrete logarithm problem of Section 4 to the multiplicative groups $\mathbb{F}_q^*$. These groups are always cyclic, of cardinality $n = q - 1$.

Input: A finite field $\mathbb{F}_q$ and two elements g, $h \in \mathbb{F}_p^*$

Output: The integer $x \in \{0, \ldots, q-2\}$ such that $h = g^x$

5.2 Complexity class

Of course, the generic algorithms of Section 4 may be used also in the special case of finite fields, so that if the factorisation $q - 1 = \prod p_i^{e_i}$ is known, a trivial upper bound of the problem complexity is given by $O\left(\sum e_i \sqrt{p_i}\right)$ finite field operations and $O\left(\log q \sum e_i \sqrt{p_i}\right)$ comparisons of field elements. In particular, if $q - 1$ is smooth (has only small prime factors), then the discrete logarithm problem in $\mathbb{F}_q$ becomes easy to solve.

Otherwise, there is a close parallel between the factorisation problem and the discrete logarithm problem in finite fields. In both cases, there are probabilistic algorithms relying on finding smooth relations (identities involving small primes) and linear algebra that have a subexponential complexity.

Precisely, let for $\alpha \in (0, 1)$ and $c > 0$

$$L_q(\alpha, c) = e^{(c+o(1))\,(\log q)^\alpha\,(\log\log q)^{1-\alpha}}$$

denote the subexponential function with respect to the input size $\log q$.

Then for $q = p$ prime [37] or $q = 2^m$ [37, 6], there is an algorithm with complexity $L_q(1/2, \sqrt{2})$ for the discrete logarithm problem in $\mathbb{F}_q$. For general finite fields, an algorithm with heuristic complexity $L_q(1/2)$ has been given by Adleman and DeMarrais [2] (here and in the following, we omit the precise value of the constant c).

Again for the special cases $q = p$ or $q = 2^m$, special algorithms (the number field sieve [21, 40] resp. the function field sieve [1] or Coppersmith's algorithm [8]) obtain a heuristic complexity of $L_q(1/3)$. The function field sieve applies as well to other fields of small characteristic. In [30], the authors close the gap between these two extreme cases by suitably modifying the number field sieve. As a result, there is an algorithm of heuristic complexity $L(1/3)$ for all finite fields.

No lower bounds are known.

5.3 Best algorithm known

Practice matches theory in this context, since the asymptotically fastest algorithms of the previous section behave well in practice and have indeed been used to obtain the records mentioned in the next section.

5.4 Performance records

The current record for discrete logarithms in finite prime fields is hold by Kleinjung, who has computed them in a prime field of 530 bits [31]. For fields of characteristic 2, the record of $\mathbb{F}_{2^{613}}$ has been obtained by Joux and Lercier [28]. An implementation for fields of characteristic 3 is described in [19]. Joux and Lercier also hold the record for the intermediate case of a medium degree extension of a medium characteristic prime field [29].

5.5 Recommended key lengths

A precise running time analysis and prediction for input sizes beyond those already tackled by the number and function field sieves is rather difficult. Too many parameters would have to be taken into account, and their optimal choice is unknown. So the only thing that can be said with certainty is that keys have to be longer than the records mentioned in the previous section, but of course, this is not sufficient. It appears that the discrete logarithm problem is harder than the factorisation problem for the same key length. This is due to the fact that the linear algebra step has to be carried out not modulo 2, but modulo $q - 1$. So taking the same key lengths as for cryptosystems based on the factorisation problem seems to be a good choice.

5.6 Related open problems

It is unknown whether an algorithm of (heuristic) complexity $L_q(1/3)$ exists that works for all finite fields.

Chapter 6

Elliptic Curve and Algebraic Curve Cryptography

6.1 Parameters of the problem

Let $\mathbb{F}_q$ be a finite field, and C a non-singular, absolutely irreducible projective curve over $\mathbb{F}_q$ with a unique point at infinity, that is furthermore defined over $\mathbb{F}_q$. Then the $\mathbb{F}_q$-rational part J of the Jacobian of C is a finite abelian group, in which the discrete logarithm problem is defined as usual.

In the particular case of an elliptic curve defined over a finite field $\mathbb{F}_q$, the group is given by the points on the curve with coordinates in $\mathbb{F}_q$, and the tangent-and-chord law for the addition.

6.2 Complexity class

Again, the generic algorithms of Section 4 can be applied, so if the cardinality n of the Jacobian is smooth (it has only small prime factors), then the discrete logarithm problem in J is easy. For an optimal security level with respect to a given parameter size, one should thus choose n prime or almost prime with a very small cofactor.

For arbitrary elliptic curves, no other than the generic algorithms of exponential complexity are known. Some special classes of negligible density in the set of all elliptic curves admit faster algorithms for solving the discrete logarithm problem:

- If J has a subgroup of order p, where p is the characteristic of the base field, then the discrete logarithm problem in this subgroup can be embedded into the additive group of $\mathbb{F}_p$, where it is solved in polynomial time by the extended Euclidian algorithm [39, 41, 44]. An analogous algorithm exists for more general curves [38].
- If J has a subgroup of prime order r not dividing $q - 1$ and k is such that $r | q^k - 1$, then the full group of r-torsion points on J is defined over $\mathbb{F}_{q^k}$ [5]. The Tate or Weil pairing allow to embed the discrete logarithm problem on J into the multiplicative group of $\mathbb{F}_{q^k}$ [33, 16]. If k is bounded, then the discrete logarithm problem in the finite field is subexponential (see Section 5.2). In particular, this is true for supersingular elliptic curves, that have $k \geq 2$

over a field of characteristic different from 2 and 3, $k \geq 4$ over $\mathbb{F}_{2^m}$ and $k \leq 6$ over $\mathbb{F}_{3^m}$. Again, the same ideas apply also to more general curves [16].

For hyperelliptic curves over $\mathbb{F}_q$ whose genus g tends sufficiently fast to infinity, the complexity of the discrete logarithm problem is bounded above by the subexponential function $L_{q^g}(1/2)$. As in the case of finite fields (see Section 5.2), the corresponding algorithms form a matrix of "smooth relations" between small prime elements and proceed by linear algebra. The first algorithm with a conjectured subexponential running time is given in [3]. [14] describes the first algorithm with a proven complexity, provided that g grows at least as fast as $\log q$. See also [13], where a general framework for subexponential discrete logarithm algorithm is developed.

A more general subexponential algorithm (with complexity $L(1/2)$) for discrete logarithms in high genus curves is given in [9]. The main conditions that have to be fulfilled are that the curve genus g tend to infinity, that there is a rational point on each curve, and that the cardinalities of the J are bounded above by $q^{g+O(\sqrt{g})}$. In particular, these conditions hold for superelliptic curves $Y^a = f(X)$ for any fixed a (and the degree of f tending to infinity) or more generally $C_{a,b}$ curves for any fixed a (and b tending to infinity).

The most general result to date is given in [25], where the essential restrictions are that the genus g tends to infinity and that the ratio $\log q/(g \log g)$ tends to zero. No limits are placed on the size of the Jacobian.

For curves of fixed genus, the smooth relation approach yields algorithms with an exponential complexity, that may however be faster than the generic ones (see the following section).

The discrete logarithm in some curves (probably of negligible density) is subject to the Weil descent algorithm [15]. Precisely, the discrete logarithm problem of a curve C defined over $\mathbb{F}_{p^m}$ with $m > 1$ may be embedded into the Weil restriction A of the curve, which is an abelian variety of dimension m over $\mathbb{F}_p$. Sometimes, it is possible to identify a curve on A of comparatively low genus (optimally of genus m) [22, 4, 20, 26, 11]; in this case, the relation generating algorithms apply, and a subexponential complexity may be derived for such special classes of curves. The way in which hyperelliptic curves may be found on the Weil restriction is not canonic, and for a given curve C, there is no algorithm for determining the minimal genus of a curve on A.

No lower bound is known.

6.3 Best algorithm known

We assume that the discrete logarithm problem on J is not embeddable into the additive or multiplicative group of a low degree extension of its field of definition $\mathbb{F}_q$ as described in the previous section; otherwise said, the cardinality of J is neither q, nor does it (or its largest prime factor) divide $q^k - 1$ for some small k.

For the time being, let us also assume that the discrete logarithm problem is not embedded via Weil descent into a higher dimensional abelian variety over a subfield, so that it is solved directly in J.

For elliptic curves and curves of genus 2, the best algorithms known are the generic ones of Section 4 of square root complexity. If the curves have an automorphism of order m whose action is easily identified by inspection (for instance, the Frobenius endomorphism $(x, y) \mapsto (x^p, y^p)$ on an elliptic curve over $\mathbb{F}_{p^m}$, multiplication by i on an elliptic curve with complex multiplication by $\mathbb{Z}+\mathbb{Z}i$ or multiplication by ζ_3 on an elliptic curve with complex multiplication bz $\mathbb{Z}+\mathbb{Z}\frac{1+\sqrt{3}}{2}$), then the algorithms may be sped up by a factor of $\sqrt{m}$ by identifying orbits of the endomorphism [10].

In [17], it is noted that in hyperelliptic curves of fixed and sufficiently large genus, the relation collecting algorithms, while still exponential, have in fact a better complexity than the generic ones. The algorithm extends trivially to the setting of more general curves, and later refinements [45, 23] arrive at a complexity of $O(q^{2-2/g})$ for a curve of genus g defined over $\mathbb{F}_q$. This is better than generic square root algorithms, of complexity $O(q^{g/2})$, as soon as $g \geq 3$.

In [12], this method is improved in the case where the curve has an equation of low degree. In particular, for non-hyperelliptic curves of genus 3, the complexity drops to $O(q)$.

Furthermore, [18] presents an algorithm for elliptic curves over extension fields $\mathbb{F}_{q^m}$ with $m > 1$ that does not rely on Weil descent and in theory works for all such curves. The use of Gröbner basis computations seems to restrict the algorithm in practice to $m \leq 4$. It attains the same complexity as the algorithms of the previous paragraph, so that the discrete logarithm problem in elliptic curves defined over fields $\mathbb{F}_{p^3}$ and $\mathbb{F}_{p^4}$ becomes easier than over prime fields or quadratic extensions, for a comparable group size. Claus Diem has noticed (presentation at the 8th Workshop on Elliptic Curve Cryptography, ECC 2004) that letting tend p and m to infinity suitably at the same time, one obtains a family of curves for which the discrete logarithm problem can be solved in subexponential time $L(3/4)$.

6.4 Performance records

All records obtained for computing elliptic curve discrete logarithms have used a generic, parallelised birthday paradox algorithm as explained in Section 4. The problem instances are those provided by the Certicom challenge [7].

The record for a general discrete logarithm computation on an elliptic curve over a finite prime field was set in 2002, and is for a curve of 109 bits. The same team successfully computed a discrete logarithm on a curve over a field of characteristic 2 of the same bit size in 2004.

Concerning relation collecting algorithms, the current record is set in [17] for a hyperelliptic curve of genus 6 over $\mathbb{F}_{2^{23}}$ corresponding to about 138 bits. The more advanced algorithms of [45, 23] have not been used to set records; their main merit is to yield precise recommendations for key sizes.

6.5 Recommended key lengths

For elliptic curves over fields of the form $\mathbb{F}_p$ or $\mathbb{F}_{p^2}$ or hyperelliptic curves of genus 2 over $\mathbb{F}_p$, the recommendations of Section 4 apply: a security of 2^n group operations is obtained with a key of $2n$ bits. In the case of a Koblitz curve, that is, a curve over F_{2^m} that is defined already over $\mathbb{F}_2$, one roughly needs to add an additional $\log_2 m$ bits to compensate the Frobenius automorphism.

Taking [23, 18] into account, the key size has to be multiplied by a factor of 9/8 for elliptic curves over $\mathbb{F}_{p^3}$ or hyperelliptic curves of genus 3, and by a factor of 4/3 for elliptic curves over $\mathbb{F}_{p^4}$ or curves of genus 4. For non-hyperelliptic curves of genus 3, due to [12] the key size should be multiplied by a factor of 3/2.

6.6 Related open problems

As for general finite fields, it remains an open problem to find an algorithm for the discrete logarithm in some class of curves with a complexity of $L(1/3)$. Another question is whether the discrete logarithm problem for elliptic curves with complex multiplication by a small discriminant is easier to solve than in the general case.

Acknowledgements:

The eCrypt contributor for the discrete-log sections is Andreas Enge (INRIA).

Bibliography

[1] Leonard M. Adleman. The function field sieve. In Leonard M. Adleman and Ming-Deh Huang, editors, *Algorithmic Number Theory*, volume 877 of *Lecture Notes in Computer Science*, pages 108–121, Berlin, 1994. Springer-Verlag.

[2] Leonard M. Adleman and Jonathan DeMarrais. A subexponential algorithm for discrete logarithms over all finite fields. *Mathematics of Computation*, 61(203):1–15, 1993.

[3] Leonard M. Adleman, Jonathan DeMarrais, and Ming-Deh Huang. A subexponential algorithm for discrete logarithms over the rational subgroup of the Jacobians of large genus hyperelliptic curves over finite fields. In Leonard M. Adleman and Ming-Deh Huang, editors, *Algorithmic Number Theory*, volume 877 of *Lecture Notes in Computer Science*, pages 28–40, Berlin, 1994. Springer-Verlag.

[4] Seigo Arita. Weil descent of elliptic curves over finite fields of characteristic three. In Tatsuaki Okamoto, editor, *Advances in Cryptology — ASIACRYPT 2000*, volume 1976 of *Lecture Notes in Computer Science*, pages 248–258, Berlin, 2000. Springer-Verlag.

[5] R. Balasubramanian and N. Koblitz. The improbability that an elliptic curve has subexponential discrete log problem under the Menezes–Okamoto–Vanstone algorithm. *Journal of Cryptology*, 11:141–145, 1998.

[6] Renet Lovorn Bender and Carl Pomerance. Rigorous discrete logarithm computations in finite fields via smooth polynomials. In D. A. Buell and J. T. Teitelbaum, editors. *Computational Perspectives on Number Theory: Proceedings of a Conference in Honor of A.O.L. Atkin*, volume 7 of *Studies in Advanced Mathematics*. American Mathematical Society, 1998. Pages 221–232.

[7] Certicom. ECC challenge. http://www.certicom.com/research/ecc_challenge.html.

[8] Don Coppersmith. Fast evaluation of logarithms in fields of characteristic two. *IEEE Transactions on Information Theory*, 30(4):587–594, July 1984.

[9] Jean-Marc Couveignes. Algebraic groups and discrete logarithm. In K. Alster, J. Urbanowicz, and H. C. Williams, editors, *Public-Key Cryptography and Computational Number Theory*, pages 17–27, Berlin, 2001. De Gruyter.

[10] I. Duursma, P. Gaudry, and F. Morain. Speeding up the discrete log computation on curves with automorphisms. In Kwok Yan Lam, Eiji Okamato, and Chaoping Xing, editors.

Advances in Cryptology — ASIACRYPT '99, volume 1716 of *Lecture Notes in Computer Science*, Berlin, 1999. Springer-Verlag. Pages 103–121.

[11] Claus Diem. The GHS attack in odd characteristic. *J. Ramanujan Math. Soc.*, 18(1):1–32, 2003.

[12] Claus Diem. Index calculus in class groups of plane curves of small degree. Preprint, 2005.

[13] Andreas Enge and Pierrick Gaudry. A general framework for subexponential discrete logarithm algorithms. *Acta Arithmetica*, 102(1):83–103, 2002.

[14] Andreas Enge. Computing discrete logarithms in high-genus hyperelliptic Jacobians in provably subexponential time. *Mathematics of Computation*, 71(238):729–742, 2002.

[15] Gerhard Frey. Applications of arithmetical geometry to cryptographic constructions. In Dieter Jungnickel and Harald Niederreiter, editors, *Finite Fields and Applications — Proceedings of The Fifth International Conference on Finite Fields and Applications F_q5, held at the University of Augsburg, Germany, August 2–6, 1999*, pages 128–161, Berlin, 2001. Springer-Verlag.

[16] Gerhard Frey and Hans-Georg Rück. A remark concerning m-divisibility and the discrete logarithm in the divisor class group of curves. *Mathematics of Computation*, 62(206):865–874, April 1994.

[17] Pierrick Gaudry. An algorithm for solving the discrete log problem on hyperelliptic curves. In Bart Preneel, editor, *Advances in Cryptology — EUROCRYPT 2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 19–34, Berlin, 2000. Springer-Verlag.

[18] Pierrick Gaudry. Index calculus for abelian varieties and the elliptic curve discrete logarithm problem. Preprint, 2004.

[19] R. Granger, A. J. Holt, D. Page, N. P. Smart, and F. Vercauteren. Function field sieve in characteristic three. In Duncan Buell, editor, *Algorithmic Number Theory — ANTS-VI*, volume 3076 of *Lecture Notes in Computer Science*, pages 223–234, Berlin, 2004. Springer-Verlag.

[20] P. Gaudry, F. Hess, and N. P. Smart. Constructive and destructive facets of Weil descent on elliptic curves. *Journal of Cryptology*, 15:19–46, 2002.

[21] Daniel M. Gordon. Discrete logarithms in $GF(p)$ using the number field sieve. *SIAM Journal on Discrete Mathematics*, 6(1):124–138, 1993.

[22] Steven D. Galbraith and Nigel P. Smart. A cryptographic application of Weil descent. In Michael Walker, editor, *Cryptography and Coding*, volume 1746 of *Lecture Notes in Computer Science*, Berlin, 1999. Springer-Verlag.

[23] Pierrick Gaudry and and Nicolas Thériault and Emmanuel Thomé. A double large prime variation for small genus hyperelliptic index calculus. Preprint, available at http://-www.lix.polytechnique.fr/Labo/Pierrick.Gaudry/publis/dbleLP.ps.gz 2004.

[24] Robert Granger and Frederik Vercauteren. On the discrete logarithm problem on algebraic tori. To appear in *Crypto 2005*.

[25] Florian Hess. Computing relations in divisor class groups of algebraic curves over finite fields. Preprint, available at http://www.math.tu-berlin.de/ hess/dlog.ps.gz.

[26] Florian Hess. The GHS attack revisited. In Eli Biham, editor, *Advances in Cryptology — EUROCRYPT 2003*, volume 2656 of *Lecture Notes in Computer Science*, pages 374–387, Berlin, 2003. Springer-Verlag.

[27] Antoine Joux and Reynald Lercier. Discrete logarithms in GF(p) — 130 digits. Posting to the Number Theory List, available at http://listserv.nodak.edu/cgi-bin/wa.exe?A2=ind0506&L=nmbrthry&F=&S=&P=2037, 2005.

[28] Antoine Joux and Reynald Lercier. Discrete Logarithms in $GF(2^{607})$ and $GF(2^{613})$. Posting to the Number Theory List, available at http://listserv.nodak.edu/cgi-bin/wa.exe?A2=ind0509&L=nmbrthry&T=0&P=3690, September 2005.

[29] Antoine Joux and Reynald Lercier. Discrete Logarithms in $GF(65537^{25})$ — 120 digits — 400 bits. Posting to the Number Theory List, available at http://listserv.nodak.edu/cgi-bin/wa.exe?A2=ind0510&L=nmbrthry&T=0&P=1640, October 2005.

[30] Antoine Joux and Reynald Lercier and Nigel Smart and Frederik Vercauteren. The Number Field Sieve in the Medium Prime Case. In: Advances in Cryptology — CRYPTO 2006, Cynthia Dwork (ed.), Lecture Notes in Computer Science Vol. 4117, Springer-Verlag, Berlin, 2006, pp. 326–344.

[31] Thorsten Kleinjung. Discrete Logarithms in $GF(p)$ — 160 digits. Posting to the Number Theory List, available at http://listserv.nodak.edu/cgi-bin/wa.exe?A2=ind0702&L=nmbrthry&T=0&P=194, February 2007.

[32] Reynald Lercier and Frederik Vercauteren. Discrete logarithms in GF(p^{18}) - 101 digits Posting to the Number Theory List, available at http://listserv.nodak.edu/cgi-bin/wa.exe?A2=ind0506&L=nmbrthry&F=&S=&P=2803, 2005.

[33] Alfred J. Menezes, Tatsuaki Okamoto, and Scott A. Vanstone. Reducing elliptic curve logarithms to logarithms in a finite field. *IEEE Transactions on Information Theory*, 39(5):1639–1646, September 1993.

[34] V. I. Nechaev. Complexity of a determinate algorithm for the discrete logarithm. *Mathematical Notes*, 55(2):165–172, 1994.

[35] Stephen C. Pohlig and Martin E. Hellman. An improved algorithm for computing logarithms over $GF(p)$ and its cryptographic significance. *IEEE Transactions on Information Theory*, 24(1):106–110, January 1978.

[36] J. M. Pollard. Theorems on factorization and primality testing. *Proc. Camb. Phil. Soc.*, 76:521–528, 1974.

[37] Carl Pomerance. Fast, rigorous factorization and discrete logarithm algorithms. In David S. Johnson, Takao Nishizeki, Akihiro Nozaki, and Herbert S. Wolf, editors. *Discrete Algorithms and Complexity, Proceedings of the Japan–US Joint Seminar, June 4–6, 1986, Kyoto, Japan*, volume 15 of *Perspectives in Computing*, Orlando, 1987. Academic Press. Pages 119–143.

[38] Hans-Georg Rück. On the discrete logarithm in the divisor class group of curves. *Mathematics of Computation*, 68(226):805–806, April 1999.

[39] Takakazu Satoh and Kiyomichi Araki. Fermat quotients and the polynomial time discrete log algorithm for anomalous elliptic curves. *Commentarii Mathematici Universitatis Sancti Pauli*, 47(1):81–92, 1998. Errata in vol. 48 (2):211–213, 1999.

[40] Oliver Schirokauer. Discrete logarithms and local units. *Philosophical Transactions Royal Society London A*, 345:409–423, 1993.

[41] I. A. Semaev. Evaluation of discrete logarithms in a group of p-torsion points of an elliptic curve in characteristic p. *Mathematics of Computation*, 67(221):353–356, 1998.

[42] D. Shanks. Class number, a theory of factorization and genera. Donald Lewis, editor. In *Proceedings of Symposia in Pure Mathematics*, volume 10, Providence (Rhode Island), 1971. American Mathematical Society. Pages 415–440.

[43] Victor Shoup. Lower bounds for discrete logarithms and related problems. In Walter Fumy, editor. *Advances in Cryptology — EUROCRYPT '97*, volume 1233 of *Lecture Notes in Computer Science*, Berlin, 1997. Springer-Verlag, 1997, pages 256–266.

[44] Nigel P. Smart. The discrete logarithm problem on elliptic curves of trace one. *Journal of Cryptology*, 12(3):193–196, 1999.

[45] Nicolas Thériault. Index calculus attack for hyperelliptic curves of small genus. In Chi Sung Laih, editor, *Advances in Cryptology — ASIACRYPT 2003*, volume 2894 of *Lecture Notes in Computer Science*, pages 75–92, Berlin, 2003. Springer-Verlag.

[46] Emmanuel Thomé. Computation of discrete logarithms in $\mathbb{F}_{2^{607}}$. In Colin Boyd, editor, *Advances in Cryptology — ASIACRYPT 2001*, volume 2248 of *Lecture Notes in Computer Science*, pages 107–124, Berlin, 2001. Springer-Verlag.

Chapter 7

Lattice reduction

7.1 Problem statement

Let L be a full-rank lattice in $\mathbb{Z}^n$ (see [16, 14] for a bibliography on lattices). Denote by $\|.\|$ the Euclidean norm in $\mathbb{R}^n$ The two most famous lattice problems are the shortest vector problem (SVP) and the closest vector problem (CVP):

- SVP: Given an arbitrary basis of L, find a shortest vector of L, that is, find a non-zero $\vec{x} \in L$ such that $\|\vec{x}\|$ is minimal. Approximating SVP to within a factor k means finding a non-zero $\vec{y} \in L$ such that $\|\vec{y}\| \leq k\|\vec{x}\|$, where $\vec{x}$ is a shortest vector.

- CVP: Given an arbitrary basis of L and a target vector $\vec{t} \in \mathbb{Q}^n$, find a lattice vector closest to $\vec{t}$, that is, find $\vec{x} \in L$ minimizing $\|\vec{x} - \vec{t}\|$ among all lattice vectors.

7.2 Parameters of the problem

There are two lattice parameters: the lattice dimension n and the size $\log B$ of the lattice basis defining the lattice (that is, the number of bits required to store the matrix corresponding to the lattice basis). The parameter defining the hardness of lattice problems is the lattice dimension n.

7.3 Complexity class

Complexity results are asymptotic results when the lattice dimension increases. CVP is NP-hard. Approximating CVP to within a quasi-polynomial factor $2^{\log^{1-\varepsilon} n}$ is NP-hard [3, 4]. SVP is NP-hard under randomized reductions [1]. Approximating SVP to within a constant is also NP-hard under randomized reductions [13].

CVP seems to be a more difficult problem than SVP. CVP cannot be easier than SVP: given an oracle that approximates CVP to within a factor $f(n)$, one can approximate SVP in polynomial time to within the same factor $f(n)$. Reciprocally, Kannan proved in [12, Section 7] that any algorithm approximating SVP to within a non-decreasing function $f(d)$ can be used to approximate CVP to within $n^{3/2}f(n)^2$.

However, NP-hardness results for SVP and CVP have limits. Goldreich and Goldwasser [5] showed that approximating SVP or CVP to within $\sqrt{n/\log n}$ is not NP-hard, unless the polynomial-time hierarchy collapses, which is considered unlikely.

7.4 Best algorithm known

The best provable algorithm known for solving SVP exactly is the AKS sieving algorithm [2], which is a probabilistic algorithm running in time $2^{O(n)}$. For exact CVP, the best algorithm remains Kannan's super-exponential algorithm [10, 11], with running time $2^{O(n \log n)}$ (see also [8] for an improved constant).

However, in practice, one rather uses more efficient approximation algorithms, and hope that the output is the shortest lattice vector. The best provable polynomial-time approximation algorithm is Schnorr's block Korkine-Zolotorev (BKZ) algorithm [17] coupled with the AKS algorithm [2]: its approximation factor is $2^{O(n(\log\log n)^2/\log n)}$, which is subexponential. Experimentally, the best approximation algorithm seems to be the heuristic Schnorr-Hrner algorithm [18], which is a variant of BKZ, and which is implemented in NTL [19]. No good prediction on the behaviour of the algorithm is known.

7.5 Performance records

The largest lattice problem ever solved seem to be: the 350-dimensional GGH [7] lattice [15] and the 214-dimensional NTRU lattice [9] (corresponding to NTRU-107 parameters). Those records used very limited computational power: a single computer for a few days at most.

7.6 Recommended key lengths

It is very hard to make any recommendation, because the behaviour of lattice basis reduction algorithms is not very well-understood at the moment. The authors of the NTRU cryptosystem [9] now recommend to use a lattice of dimension ≥ 500, based on their own experiments with NTL [19]. The lattices used in NTRU have a special form: it is unknown if such a form can be exploited by lattice basis reduction algorithms.

7.7 Related open problems

Lattice problems are connected with coding problems.

Bibliography

[1] M. Ajtai. The shortest vector problem in L_2 is NP-hard for randomized reductions. In *Proc. of 30th STOC*. ACM, 1998.

[2] M. Ajtai, R. Kumar, and D. Sivakumar. A sieve algorithm for the shortest lattice vector problem. In *Proc. 33rd STOC*, pages 601–610. ACM, 2001.

[3] S. Arora, L. Babai, J. Stern, and Z. Sweedyk. The hardness of approximate optima in lattices, codes, and systems of linear equations. *Journal of Computer and System Sciences*, 54(2):317–331, 1997.

[4] I. Dinur, G. Kindler, and S. Safra. Approximating CVP to within almost-polynomial factors is NP-hard. In *Proc. of 39th FOCS*, pages 99–109. IEEE, 1998.

[5] O. Goldreich and S. Goldwasser. On the limits of non-approximability of lattice problems. In *Proc. of 30th STOC*. ACM, 1998.

[6] O. Goldreich, S. Goldwasser, and S. Halevi. Challenges for the GGH cryptosystem. Available at `http://theory.lcs.mit.edu/~shaih/challenge.html`.

[7] O. Goldreich, S. Goldwasser, and S. Halevi. Public-key cryptosystems from lattice reduction problems. In *Proc. of Crypto '97*, volume 1294 of *LNCS*, pages 112–131. IACR, Springer-Verlag, 1997.

[8] B. Helfrich. Algorithms to construct Minkowski reduced and Hermite reduced bases. *Theoretical Computer Science*, 41:125–139, 1985.

[9] J. Hoffstein, J. Pipher, and J.H. Silverman. NTRU: a ring based public key cryptosystem. In *Proc. of ANTS III*, volume 1423 of *LNCS*, pages 267–288. Springer-Verlag, 1998.

[10] R. Kannan. Improved algorithms for integer programming and related lattice problems. In *Proc. of 15th STOC*, pages 193–206. ACM, 1983.

[11] R. Kannan. Minkowski's convex body theorem and integer programming. *Math. Oper. Res.*, 12(3):415–440, 1987.

[12] R. Kannan. Algorithmic geometry of numbers. *Annual review of computer science*, 2:231–267, 1987.

[13] S. Khot. The hardness of . In *Proc. of FOCS*, IEEE, 2004.

[14] D. Micciancio and S. Goldwasser. *Complexity of lattice problems: A cryptographic perspective.* Kluwer Academic Publishers, Boston, 2002.

[15] P. Q. Nguyen. Cryptanalysis of the Goldreich-Goldwasser-Halevi cryptosystem from Crypto '97. In *Proc. of Crypto '99*, volume 1666 of *LNCS*, pages 288–304. IACR, Springer-Verlag, 1999.

[16] P. Q. Nguyễn and J. Stern. The two faces of lattices in cryptology. In *Cryptography and Lattices – Proc. CALC '01*, volume 2146 of *Lecture Notes in Computer Science*, pages 146–180. Springer-Verlag, 2001.

[17] C. P. Schnorr. A hierarchy of polynomial lattice basis reduction algorithms. *Theoretical Computer Science*, 53:201–224, 1987.

[18] C. P. Schnorr and H. H. Hörner. Attacking the Chor-Rivest cryptosystem by improved lattice reduction. In *Proc. of Eurocrypt '95*, volume 921 of *LNCS*, pages 1–12. IACR, Springer-Verlag, 1995.

[19] V. Shoup. Number Theory C++ Library (NTL). Available at `http://www.shoup.net/ntl/`.

Chapter 8

Systems of polynomial equations in finite field

8.1 Problem statement

Let $\mathbb{F}$ be a finite field of prime characteristic with $q := |\mathbb{F}|$ elements. In particular, q is a prime-power. Let $n \in \mathbb{N}$ be the number of variables, $m \in \mathbb{N}$ the number of equations, and $d \in \mathbb{N}$ the degree of the system. Moreover, let $x_1, \ldots, x_n$ be variables over $\mathbb{F}$. By convention, we set $x_0 := 1$, *i.e.* the multiplicative neutral in $\mathbb{F}$. Furthermore, we define

$$\mathcal{V}_n^d := \begin{cases} \{0\} & \text{for } d = 0 \\ \{v \in \{0, \ldots, n\}^d : i \leq j \Rightarrow v_i \leq v_j\} & \text{otherwise} \end{cases}$$

where we denote components of the vector v by $v_1, \ldots, v_d \in \{0, \ldots, n\}$. We are now able to state our problem. Let $\mathcal{P}$ be a system of m polynomials in n variables with maximum degree $d \in \mathbb{N}$ each, *i.e.* we have $\mathcal{P} := (p_1, \ldots, p_m)$ where all p_i have the form

$$p_i(x_1, \ldots, x_n) := \sum_{v \in \mathcal{V}_n^d} \gamma_{i,v} \prod_{j=1}^{d} x_{v_j} \text{ for } 1 \leq i \leq m$$

with the coefficients $\gamma_{i,v} \in \mathbb{F}$ and vectors $v \in \mathcal{V}_n^d$.

For any q and $d = 2$, we speak about the problem of $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic equations and denote the class of corresponding polynomial vectors with $\mathcal{MQ}_m(\mathbb{F}^n)$. As we will see below, this class plays an important role for the construction of public key schemes based on the problem of polynomial equations in finite fields. Therefore, we give the polynomials p_i explicitly for this case:

$$p_i(x_1, \ldots, x_n) := \sum_{1 \leq j \leq k \leq n} \gamma_{i,j,k} x_j x_k + \sum_{j=1}^{n} \beta_{i,j} x_j + \alpha_i$$

for $1 \leq i \leq m, 1 \leq j \leq k \leq n$, and the coefficients $\gamma_{i,j,k}, \beta_{i,j}, \alpha_i \in \mathbb{F}$. In the case of $d = 2$, we call them quadratic ($\gamma_{i,j,k}$), linear ($\beta_{i,j}$), and constant (α_i) coefficients, respectively.

8.2 Parameters of the problem

The problem has the parameters degree $d \in \mathbb{N}$, field-size $q = p^k$ for $k \in \mathbb{Z}^+$ and p a prime number. Moreover, we have the number of variables $n \in \mathbb{N}$ and the number of equations $m \in \mathbb{N}$.

8.3 Complexity class

Degree 0

For degree 0, the problem becomes trivial as there are no variables anymore. Hence, it is not possible to find a satisfying assignment for a given formula.

Degree 1

In the case of degree 1, the problem coincides with the well-studied linear and affine equations over finite fields. They can be solved in $\mathcal{O}(n^3)$ (assuming $n = m$) using Gaussian elimination and $\mathcal{O}(n^\omega)$, with $2 \le \omega \le 3$, in the case of sparse equations or using Strassen's algorithm.

Degree ≥ 2

With $d \geq 2$ and for any admissible q, the problem becomes $\mathcal{NP}$-complete (cf see [GJ79, p. 251]). The case of $d \geq 3$ equations has been treated in [AY79] with reference to an unpublished manuscript for the degree 2 case. To the knowledge of the authors, the first published proof of the case $d = 2$ appeared in [PG97, Appendix]; the paper also includes $d \geq 2$. A more detailed proof of the general case can be found in [Wol02, Sect. 3.2].

Quantum complexity class

At present, there are no results known for the solvability of this problem in degree 2 or higher using quantum algorithms.

8.4 Best algorithms known

For the case $n \leq m$, the algorithms F_5 and $\mathrm{F}_5/2$ are most powerful [Fau02b, FJ03]. They are based on Gröbner basis computation, cf [ALW95] for an introduction to Gröbner bases.

In the underdetermined case, *i.e.* for $n > m$, the algorithms described in [CGMT02] outperform Gröbner basis computations.

8.5 Performance records

In 2002, Faugère reported in [Fau02a] that he broke HFE Challenge 1. The parameters where $d = 2$, $n = 80$, $m = 80$. He used the algorithm $\mathrm{F}_5/2$ for this purpose (cf Sect. 8.4). According to [Fau02a], it took one AlphaServer DS20E (EV68 833 Mhz) and 4 GB of RAM a total of 96 hours of computing time, to break HFE Challenge 1.

In 2004, Steel showed that the Magma [MAG] implementation of the F_4-algorithm outperforms Faugère's own $\mathrm{F}_5/2$ implementation, cf [Ste]. He reports the use of a Sun V20Z with one

Opteron 248 processor (2.2GHz, 1MB L2 cache) and 8GB memory. The computations took 6.93 hours and needed 8.07GB of memory.

Timings for random systems, *i.e.*, systems without a hidden trapdoor, are given in [Fau03, p. 9]. We quote the corresponding results. They have been achieved on a Pentium III 700 MHz (the memory-requirement is not given).

Algorithm \ n	14	16	18	19	21	33
F_4	2.4 [s]	12.3 [s]	70.5 [s]	133.2 [s]	436.9 [s]	
$F_5/2$			0.9 [s]	1.5 [s]	4.25 [s]	442.7 [s]

Hence, breaking HFE Challenge 1 was only possible as the trapdoor imposed a special structure on the system of $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic equations, which were hence easy to solve.

Finding related messages

Let $d = 2$, *i.e.*, we are now in the case of multivariate quadratic equations. Assume we know some $\delta \in \mathbb{F}^n$ such that $x' = x + \delta$ for two unknown $x, x' \in \mathbb{F}^n$. This case has been described in [Pat96, Sect. 3, "Attack with related messages"]. It is shown that it can be solved in polynomial time — in particular $\mathcal{O}(n^3)$ if normal Gaussian elimination is used or $\mathcal{O}(n^{\log_2 7})$ with Strassen's algorithm.

8.6 Recommended key-length

The key-length in a system based on the intractability of the simultaneous solving of multivariate, non-linear equations can be computed using the following formulas. First, we define

$$\tau^d(\mathbb{F}^n) := \sum_{i=0}^{d} \tau_{(i)}(\mathbb{F}^n)$$

for the number of terms in a single polynomial equation over $\mathbb{F}$, degree d and in n variables. Here, we have

$$\tau_{(d)}(\mathbb{F}^n) := \begin{cases} \sum_i^{\min(|\mathbb{F}|-1,d)} \binom{n}{i} & \text{for } d > 0 \\ 1 & \text{for } d = 0 \end{cases}$$

for the number of terms for all degrees. For the correctness of the above formula, we want to point out that we have $x^q = x$ with $q := |\mathbb{F}|$ in all finite fields.

As previously stated, the case of $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic equations, *i.e.*, $d = 2$ is particularly important for practical applications. Hence, we give the corresponding formula explicitly for this case:

$$\tau(n) := \begin{cases} 1 + n + \frac{n(n-1)}{2} = 1 + \frac{n(n+1)}{2} \text{ if } \mathbb{F} = GF(2) \\ \\ 1 + n + \frac{n(n+1)}{2} = 1 + \frac{n(n+3)}{2} \quad \text{otherwise} . \end{cases}$$

We are now able to give some recommended key-sizes as function

$$\textit{size}(\mathbb{F}, n, m, d) := m\tau^d(\mathbb{F}^n) \log_2 q \ .$$

In general, we obtain a key-length of $\mathcal{O}(mn^d)$ for the public key. However, we want to stress that the correct choice of parameters — and hence the sizes achieved — depend heavily on the intractability of the corresponding trapdoor-problem. Hence, we have to specify the recommended key-length in terms of the trapdoor-function used. For this subsection, we concentrated on trapdoors which are known since at least 5 years, so we may assume that their security is well understood by the cryptographic research community.

No Trapdoor

Strictly speaking, this cannot be used to construct a public key scheme. However — keeping the limitations of Sect. 8.5 in mind — we can use this problem to "benchmark" other problems in terms of minimal key sizes possible and also to construct a one-way-function, *i.e.*, the problem to find $x \in \mathbb{F}^n$ for given $y \in \mathbb{F}^m$.

q	n	m	Key Size (kByte)
16	42	42	12.5
128	37	37	23

The values given above are rather conservative as we are not aware of a systematic study on the intractability of random systems of equations, employing different algorithms. Hence, we expect these values to drop considerably as soon as such a study has been carried out.

$\mathbf{C^{*-}}$

We suggest the parameter from Sflash^{v3} here [CGP02]:

q	n	m	Key Size (kByte)
128	67	56	112.3

HFE-

We use the parameter from a tweaked version of Quartz [CGP01, WP04]:

q	n	m	Key Size (kByte)
2	107	100	71

UOV

We use the parameter from [KPG03], taking the attacks from [CGMT02, BWP05] into account:

q	n	m	Key Size (kByte)
16	32	16	9
16	48	16	16

8.7 Related open-problems

8.7.1 Isomorphism of Polynomials

For the construction of secure public key systems based on polynomial equations over finite fields, the security of the IP-problem [Pat96], *i.e.*, the difficulty to find affine transformations $S \in \mathrm{AGL}_n(\mathbb{F})$ and $T \in \mathrm{AGL}_m(\mathbb{F})$ such that $\mathcal{P} = T \circ \mathcal{P}' \circ S$ for given polynomial vectors $\mathcal{P}, \mathcal{P}'$ is also important. In particular, the private key in such systems is usually the triple $(S, \mathcal{P}', T)$ and the public key the polynomial vector $\mathcal{P}$. Hence, if the IP-problem were easy, the security of these schemes would be seriously jeopardized. Hence, these constructions have to make the (often not explicitly stated) assumption that the corresponding IP-problem is difficult. However, if $\mathcal{P}'$ has a special structure — as it is the case for all systems based on the difficulty of solving a system of polynomial equations over a finite field — it is possible that the corresponding IP-problem becomes easy and the system can be broken that way, cf [KS98, GC00, WBP04].

A discussion of the security of the general IP-problem can be found in [Pat96, PGC98, GMS02, LP03]. In particular, [LP03] shows that the IP-problem with one secret, *i.e.*, T is given or the identity transformation, can be solved if $m \geq n$, *i.e.*, the number of variables does not exceed the number of variables.

8.7.2 MinRank

Let $(M_1, \ldots, M_k)$ be a sequence of $k \in \mathbb{N}$ matrices over $\mathbb{F}^{n \times n}$ each. Moreover, let $r \in \mathbb{N}$. For the MinRank-problem, we are interested in finding a linear combination of the above matrices, *i.e.*, a vector $\lambda \in \mathbb{F}^k$ such that

$$\mathrm{Rank}(\sum_{i=1}^{k} \lambda_i M_i) \leq r\,.$$

The above problem has been shown to be $\mathcal{NP}$-complete when stated over finite fields [BFS96].

In special cases, namely when the rank r is extremely small or the maximal rank $R \in \mathbb{N}$ of the matrices $M_1, \ldots, M_k$ is very close to n, the problem becomes tractable. In particular, [GC00] gives two algorithms with complexity $\mathcal{O}(q^r)$ and $\mathcal{O}(q^{n-R})$, respectively, for these two special cases. The question if a more efficient algorithm for these cases or even the general MinRank-problem exists remains open. A positive answer would have serious consequences for the security of several schemes based on the $\mathcal{MQ}$-problem as the MinRank-problem has been used in the cryptanalysis of several systems, *e.g.*, in [CSV93, CSV97, KS98, KS99, GC00, WBP04].

Acknowledgements:

The eCrypt contributors for this section are Louis Goubin and Christopher Wolf.

Bibliography

[ALW95] Iyad A. Ajwa, Zhuojun Liu, and Paul S. Wang. Gröbner bases algorithm. Technical report, Institute for Computational Mathematics, Kent State University, February 1995. `http://icm.mcs.kent.edu/reports/1995/gb.pdf`, 15 pages.

[AY79] A.S.Fraenkel and Y.Yesha. Complexity of problems in games, graphs and algebraic equations. *Discrete Applied Mathematics*, 1(1-2):15–30, September 1979.

[BFS96] Jonathan F. Buss, Gudmund Skovbjerg Frandsen, and Jeffrey Outlaw Shallit. The computational complexity of some problems of linear algebra. Research Series RS-96-33, BRICS, Department of Computer Science, University of Aarhus, September 1996. `http://www.brics.dk/RS/96/33/`, 39 pages.

[BWP05] An Braeken, Christopher Wolf, and Bart Preneel. A study of the security of unbalanced oil and vinegar signature schemes. In *The Cryptographer's Track at RSA Conference 2005*, Lecture Notes in Computer Science. Alfred J. Menezes, editor, Springer, 2005. 13 pages, cf `http://eprint.iacr.org/2004/222/`.

[CGMT02] Nicolas Courtois, Louis Goubin, Willi Meier, and Jean-Daniel Tacier. Solving underdefined systems of multivariate quadratic equations. In *Public Key Cryptography — PKC 2002*, volume 2274 of *Lecture Notes in Computer Science*, pages 211–227. David Naccache and Pascal Paillier, editors, Springer, 2002.

[CGP01] Nicolas Courtois, Louis Goubin, and Jacques Patarin. *Quartz: Primitive specification (second revised version)*, October 2001. `https://www.cosic.esat.kuleuven.ac.be/nessie/workshop/submissions/quartzv21-b.zip`, 18 pages.

[CGP02] Nicolas Courtois, Louis Goubin, and Jacques Patarin. *SFlash: Primitive specification (second revised version)*, 2002. `https://www.cosic.esat.kuleuven.ac.be/nessie/workshop/submissions/sflashv2.zip`, 11 pages.

[CSV93] Don Coppersmith, Jacques Stern, and Serge Vaudenay. Attacks on the birational permutation signature schemes. In *Advances in Cryptology — CRYPTO 1993*, volume 773 of *Lecture Notes in Computer Science*, pages 435–443. Douglas R. Stinson, editor, Springer, 1993.

[CSV97] Don Coppersmith, Jacques Stern, and Serge Vaudenay. The security of the birational permutation signature schemes. *Jounal of Cryptology*, 10:207–221, 1997.

[Fau02a] Jean-Charles Faugère. HFE challenge 1 broken in 96 hours. Announcement that appeared in `news://sci.crypt`, 19th of April 2002.

[Fau02b] Jean-Charles Faugère. A new efficient algorithm for computing Gröbner bases without reduction to zero (F_5). In *International Symposium on Symbolic and Algebraic Computation — ISSAC 2002*, pages 75–83. ACM Press, July 2002.

[Fau03] Jean-Charles Faugère. Algebraic cryptanalysis of (HFE) using Gröbner bases. Technical report, Institut National de Recherche en Informatique et en Automatique, February 2003. `http://www.inria.fr/rrrt/rr-4738.html`, 19 pages.

[FJ03] Jean-Charles Faugère and Antoine Joux. Algebraic cryptanalysis of Hidden Field Equations (HFE) using gröbner bases. In *Advances in Cryptology — CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 44–60. Dan Boneh, editor, Springer, 2003.

[GC00] Louis Goubin and Nicolas T. Courtois. Cryptanalysis of the TTM cryptosystem. In *Advances in Cryptology — ASIACRYPT 2000*, volume 1976 of *Lecture Notes in Computer Science*, pages 44–57. Tatsuaki Okamoto, editor, Springer, 2000.

[GJ79] Michael R. Garay and David S. Johnson. *Computers and Intractability — A Guide to the Theory of NP-Completeness.* W.H. Freeman and Company, 1979. ISBN 0-7167-1044-7 or 0-7167-1045-5.

[GMS02] Willi Geiselmann, Willi Meier, and Rainer Steinwandt. An attack on the Isomorphisms of Polynomials problem with one secret. Cryptology ePrint Archive, Report 2002/143, 2002. `http://eprint.iacr.org/2002/143`, version from 2002-09-20, 12 pages.

[KPG03] Aviad Kipnis, Jacques Patarin, and Louis Goubin. Unbalanced oil and vinegar signature schemes — extended version, 2003. 17 pages, `citeseer/231623.html`, 2003-06-11.

[KS98] Aviad Kipnis and Adi Shamir. Cryptanalysis of the oil and vinegar signature scheme. In *Advances in Cryptology — CRYPTO 1998*, volume 1462 of *Lecture Notes in Computer Science*, pages 257–266. Hugo Krawczyk, editor, Springer, 1998.

[KS99] Aviad Kipnis and Adi Shamir. Cryptanalysis of the HFE public key cryptosystem. In *Advances in Cryptology — CRYPTO 1999*, volume 1666 of *Lecture Notes in Computer Science*, pages 19–30. Michael Wiener, editor, Springer, 1999. `http://www.minrank.org/hfesubreg.ps` or `http://citeseer.nj.nec.com/kipnis99cryptanalysis.html`.

[LP03] Françoise Levy-dit-Vehel and Ludovic Perret. Polynomial equivalence problems and applications to multivariate cryptosystems. In *Progress in Cryptology — INDOCRYPT 2003*, volume 2904 of *Lecture Notes in Computer Science*, pages 235–251. Thomas Johansson and Subhamoy Maitra, editors, Springer, 2003.

[MAG] Computational Algebra Group, University of Sydney. *The MAGMA Computational Algebra System for Algebra, Number Theory and Geometry.* `http://magma.maths.usyd.edu.au/magma/`.

[Pat96] Jacques Patarin. Hidden Field Equations (HFE) and Isomorphisms of Polynomials (IP): two new families of asymmetric algorithms. In *Advances in Cryptology — EUROCRYPT 1996*, volume 1070 of *Lecture Notes in Computer Science*, pages 33–48. Ueli Maurer, editor, Springer, 1996. Extended Version: `http://www.minrank.org/hfe.pdf`.

[PG97] Jacques Patarin and Louis Goubin. Trapdoor one-way permutations and multivariate polynomials. In *International Conference on Information Security and Cryptology 1997*, volume 1334 of *Lecture Notes in Computer Science*, pages 356–368. International Communications and Information Security Association, Springer, 1997. Extended Version: `http://citeseer.nj.nec.com/patarin97trapdoor.html`.

[PGC98] Jacques Patarin, Louis Goubin, and Nicolas Courtois. Improved algorithms for Isomorphisms of Polynomials. In *Advances in Cryptology — EUROCRYPT 1998*, volume 1403 of *Lecture Notes in Computer Science*, pages 184–200. Kaisa Nyberg, editor, Springer, 1998. Extended Version: `http://www.minrank.org/ip6long.ps`.

[Ste] Allan Steel. Allan steel's grbner basis timings page. `http://magma.maths.usyd.edu.au/users/allan/gb/`.

[WBP04] Christopher Wolf, An Braeken, and Bart Preneel. Efficient cryptanalysis of RSE(2)PKC and RSSE(2)PKC. In *Conference on Security in Communication Networks — SCN 2004*, Lecture Notes in Computer Science, pages 145–151, September 8–10 2004. extended version: `http://eprint.iacr.org/2004/237`.

[Wol02] Christopher Wolf. "Hidden Field Equations" (HFE) - variations and attacks. Diplomarbeit, Universität Ulm, December 2002. `http://www.christopher-wolf.de/dpl`, 87 pages.

[WP04] Christopher Wolf and Bart Preneel. Asymmetric cryptography: Hidden field equations. In *European Congress on Computational Methods in Applied Sciences and Engineering 2004*. P. Neittaanmäki, T. Rossi, S. Korotov, E. Oñate, J. Périaux, and D. Knörzer, editors, Jyväskylä University, 2004. 20 pages, extended version: `http://eprint.iacr.org/2004/072/`.

Chapter 9

Codes

9.1 Decoding in an arbitrary linear code

9.1.1 Problem statement:

We are given a linear code $\mathcal{C}$ of length n and dimension k over $\mathbb{F}_q$, for instance by means of a genarator matrix G. Given y in $\mathbb{F}_q^n$ at distance t or less from $\mathcal{C}$, it is difficult to find x in $\mathcal{C}$ such that $d_H(x, y) \leq t$ (where $d_H(x, y)$ denotes the Hamming distance between x and y).

9.1.2 Parameters of the problem:

Parameters are a linear code of length n and dimension k (given by its generator or parity check matrix) and a number of errors t to be corrected. In order to keep the problem difficult t must not be too large or too small. Most code-based cryptosystems use binary Goppa codes where $t = (n - k)/log_2 n$. More generally $t = f(n, k)$ must be "reasonably" increasing with $n-k$. Determining exactly which functions $f()$ lead to a difficult problem is not a fully resolved problem [5].

9.1.3 Complexity class:

The problem is closely related to *Syndrome decoding problem* [2] which was proven NP-complete. The *Goppa bounded decoding problem* is also NP-complete (see [5], the same proof as [2] applies):

> *Goppa Bounded Decoding*
> *Instance:* An $r \times n$ binary matrix H and a word s of $\mathbb{F}_2^r$.
> *Question:* Is there a word e in $\mathbb{F}_2^n$ of weight $\leq r/\log_2 n$ such that $He^T = s$?

(the matrix H above is a parity check matrix and $r = n - k$).

9.1.4 Best algorithm known:

Decoding in an arbritrary linear code is an old problem in algorithmic coding theory, a complete review can be found in [1]. The best known algorithm to address cryptographic sizes is due to

Canteaut and Chabaud [3]. Its complexity is exponential (more details in [15])

$$\exp\left(\left(\ln\frac{n}{n-k}\right)\frac{n-k}{\log_2 n}(1+o(1))\right).$$

9.1.5 Performance records:

The parameters proposed by McEliece [12] for his public key encryption scheme were $n = 1024$ and $t = 50$ (thus $k = n - t\log_2 n = 524$). The binary work factor of Canteaut-Chabaud's algorithm can be accurately estimated for a given set of parameters, for the original parameters we get $\approx 2^{64}$ binary operations. With an actual implementation, this would not require more than a few centuries of CPU time.

9.1.6 Recommended key-length:

To resist to the above attack, the length must be raised to $n = 2048$ and t must be at least 30 (see [16]).

9.1.7 Related open-problems:

The worst case complexity is known (the problem is NP-hard). Though in practice all instances are difficult, nothing is known about the average-case complexity of bounded decoding problems. Proving average case completeness [10, 7, 6] would considerably improve the formal security reduction of code-based cryptosystems.

9.2 Pseudo-randomness of binary Goppa codes

9.2.1 Problem statement:

Binary Goppa codes are used in code-based cryptosystems. Obviously, decoding in an arbitrary binary Goppa code is not the same thing as decoding in an arbitrary linear code. To reduce the security of those systems to the difficulty of decoding in a linear code, one must somehow argue that the public key of the system (a generator matrix for instance) "looks random", even though it comes from a known class and has just been shuffled to remove any apparent structure.

This leads to the following statement: given a $k \times n$ binary matrix G (with $k = n - t\log_2 n$ for some t), it is difficult to decide whether or not G is a generator matrix of some binary Goppa code.

9.2.2 Parameters of the problem:

Two positive integers t, m and a binary $k \times n$ matrix where $n \leq 2^m$ and $k = n - tm$. Usually n is chosen to be equal to 2^m but smaller values are possible to tune the matrix size.

9.2.3 Complexity class:

It is possible to state an NP problem:

Goppa Code Distinguishing
Instance: An $r \times n$ binary matrix H.
Question: Is H the parity check matrix of some binary Goppa code?

but nothing is known about it.

9.2.4 Best algorithm known:

The best known technique to solve the problem (for length $n = 2^m$) is to enumerate all Goppa codes of given support and then to test equivalence with the support splitting algorithm [11, 14]. The algorithmic complexity is $O(2^{tm}/tm)$. In practice, as far as code-based cryptosystems are concerned, this attack is always less effective than decoding attacks.

9.2.5 Recommended key-length:

As mentionned above, this attack is not the most threatening for code-based systems. For a binary workfactor of 2^{86}, a system that would depend on this problem alone would require to have $tm > 93$. For instance $t = 9$ and $n = 2^m = 2^{11} = 2048$ or $t = 6$ and $n = 2^m = 2^{16}$.

9.2.6 Related open-problems:

The problem is related with the existence of code invariants computable with low (polynomial) complexity. An invariant is a property of a code which remains the same when the code coordinates are permuted. Very few invariants can be computed in polynomial time. Actually, if we except trivial invariants (length, dimension, even weight), the only known polynomial time invariant is the weight distribution of the hull [13, 14]. All others are at least NP-hard (minimum distance, weight enumerator, ...). Finding new invariants computable in polynomial time would be a remarquable achievement in coding theory with potential consequences on the security of code-based systems. Negative results, like proving the absence of easy invariants (see Vertigan's work for instance [18]), would also be of great interest.

9.3 Decoding in a Reed-Solomon code

9.3.1 Problem statement:

We are given a Reed-Solomon code, defined as an evaluation code. Given a support $\mathcal{D} = \{\alpha_1, \dots, \alpha_n\} \subset \mathbb{F}_q$, it has a length n, a dimension k and a minimum distance $n - k + 1$, and is defined over the field $\mathbb{F}_q$. We denote this code by $RS_k(\mathcal{D})$. Let $y \in \mathbb{F}_q^n$ be given, and w an integer be given, the problem is to find x in the Reed-Solomon code, such that $d_H(x, y) \leq w$, where $d_H(x, y)$ denotes the Hamming distance betwenn x and y. It is simply the decoding problem in the case of Reed-Solomon codes.

9.3.2 Parameters of the problem:

Parameters are the support (which defines the length and the alphabet $\mathbb{F}_q$), k the dimension, and t the decoding radius. It can be decided that the support is a given mathematical set, such as, for instance, the set of n-th roots of unity over a given finite field. In such a case, the sole paramaters q and k are enough to specify the code, and there is no need for a generating matrix.

9.3.3 Complexity class:

Maximum likelihood decoding of Reed-Solomon has recently been shown to be a NP-complete problem [9]. It is the only sub instance of *Maximum likelihood decoding* known to be hard. The problem is:

> *Problem:* Maximum Likelihood decoding of Reed-Solomon codes
> *Instance:* An integer $m > 0$, a set $\mathcal{D} = \{x_1, \ldots, x_n\}$ consisting of n distinct elements of $\mathbb{F}_{2^m}$, an integer $k > 0$, a target vector $y \in \mathbb{F}_{2^m}^n$, and an integer $w > 0$.
> *Question:* Is there a codeword $c \in RS_k(\mathcal{D})$, such that $d(c, y) \leq w$.

Note that the support is part of the problem, and it remains to show the same hardness result in the case of mathematically specified subset, like, as above, the set of n-th root of unity.

9.3.4 Best algorithm known:

The main difference with respect to algorithms for solving the syndrom decoding problem, is that the size of the alphabet is growing with the input size. There has not been as much research for the q-ary case as for the binary case, for which many improvements are known. We think that the best algorithms are Information Set based decoding algorithms, which are of exponential nature. Yet, generalisation of Sudan-like algorithms [17, 8], to large decoding radius, remain to be investigated.

9.3.5 Performance records:

None reported, since the problem has not been seriously tackled.

9.3.6 Related open-problems:

Surpringly, a connection to the problem of solving discrete logarithms in a finite field has been made in [4]. Precisely the following Theorem holds

> **Theorem.** *If there exists an algorithm solving the list-decoding problem of a Reed-Solomon code of parameters $[n, k]_q$, for a decoding radius $n - \hat{g}(n, k, q)$, in time $q^{O(1)}$, then there exists an algorithm solving the discrete logarithm problem in time $q^{O(1)}$ in the field $\mathbb{F}_{q^{\hat{g}(n,k,q)-k}}$. where $\hat{g}(n, k, q)$ is a parameter, known to be smaller than $\sqrt{nk}$ (i.e. Sudan-like algorithms do not apply for this range of parameters).*

Acknowledgements:

The eCrypt contributors for this section are Nicolas Sendrier and Daniel Augot (INRIA).

Bibliography

[1] A. Barg. Complexity issues in coding theory. In V. S. Pless and W. C. Huffman, editors, *Handbook of Coding theory*, volume I, chapter 7, pages 649–754. North-Holland, 1998.

[2] E. R. Berlekamp, R. J. McEliece, and H. C. van Tilborg. On the inherent intractability of certain coding problems. *IEEE Transactions on Information Theory*, 24(3), May 1978.

[3] A. Canteaut and F. Chabaud. A new algorithm for finding minimum-weight words in a linear code: Application to McEliece's cryptosystem and to narrow-sense BCH codes of length 511. *IEEE Transactions on Information Theory*, 44(1):367–378, January 1998.

[4] Qi Cheng and Daqing Wan. On the list and bounded distance decodibility of Reed-Solomon cdoes. In *FOCS 2004 - 45th Annual IEEE Symposium on Foundations of Computer Science*. IEEE Computer Society, 2004.

[5] Matthieu Finiasz. *Nouvelles constructions utilisant des codes correcteurs d'erreurs en cryptographie clef publique*. Th'ese de doctorat, École Polytechnique, October 2004.

[6] O. Goldreich. Notes on Levin's theory of average-case complexity. In *Electronic Colloquium on Computational Complexity*, number TR97-058, November 1997.

[7] Y. Gurevich. Average case completeness. *Journal of Computer and System Sciences*, 42(3):346–398, 1991.

[8] V. Guruswami and M. Sudan. Improved decoding of Reed-Solomon and Algebraic-Geometric codes. *IEEE Transactions on Information Theory*, 45:1757–1767, 1999.

[9] Venkatesan Guruswami and Alexander Vardy. Maximum-likelihood decoding of reed-solomon codes is np-hard. Technical Report TR04-040, Electronic Colloquium on Computational Complexity, `http://www.eccc.uni-trier.de/eccc/`, 2004.

[10] L. Levin. Average case complete problems. *SIAM Journal on Computing*, 15(1):285–286, 1986.

[11] P. Loidreau and N. Sendrier. Weak keys in McEliece public-key cryptosystem. *IEEE Transactions on Information Theory*, 47(3):1207–1212, April 2001.

[12] R. J. McEliece. A public-key cryptosystem based on algebraic coding theory. *DSN Prog. Rep.,* Jet Prop. Lab., California Inst. Technol., Pasadena, CA, pages 114–116, January 1978. Jet Prop. Lab., California Inst. Technol., Pasadena, CA.

[13] N. Sendrier. On the dimension of the hull. *SIAM Journal on Discrete Mathematics*, 10(2):282–293, May 1997.

[14] N. Sendrier. Finding the permutation between equivalent codes: the support splitting algorithm. *IEEE Transactions on Information Theory*, 46(4):1193–1203, July 2000.

[15] N. Sendrier. *Cryptosystmes cl publique bass sur les codes correcteurs d'erreurs*. Mmoire d'habilitation diriger des recherches, Universit Paris 6, 2002. `http://www-rocq.inria.fr/codes/Nicolas.Sendrier/`.

[16] N. Sendrier. On the security of the McEliece public-key cryptosystem. In M. Blaum, P.G. Farrell, and H. van Tilborg, editors, *Information, Coding and Mathematics*, pages 141–163. Kluwer, 2002. Proceedings of Workshop honoring Prof. Bob McEliece on his 60th birthday.

[17] Madhu Sudan. Decoding of Reed-Solomon codes beyond the error-correction boud. *Journal of Complexity*, 13, 1997.

[18] D. Vertigan. Bicycle dimension and special points of the Tutte polynomial. *Journal of Combinatorial Theory, Series B*, 74(2):378–396, November 1998.