Advanced Cryptography
MFOCS, Oxford

Christophe Petit

January 16, 2019

Contents

1 Introduction

1.1 Course Content and Prerequisites

Cryptography is the science and art of ensuring private and authenticated communications. How does modern cryptography proceed to achieve that?

- We provide rigorous definitions of what security means in a given context, for example IND-CCA security or existential unforgeability definitions.
- We make some hardness assumption on some computational problem, for example the discrete logarithm problem or the integer factorization problem.
- We build some protocol so that we can prove that breaking the protocol (in the sense of our security definition) would imply solving the hard computational problem. For example, ElGamal encryption is IND-CPA secure if the decisional Diffie-Hellman problem is hard.

In this course I will assume you have the basic knowledge on cryptography that is normally provided in an introduction course, and certainly the one offered in MFOCS. I also assume that finite fields have no secret for you, as well as basic algebra concepts such as rings and groups.

In addition to this, I assume that you have some experience with Sage and know some basic computer algebra algorithms. (Have a look at Appendix A if you feel that a further introduction is needed.)

Of course, I assume that you are motivated, eager to learn and hard-working. I hope you will feel comfortable to ask questions when we meet or by email anytime.

Compared to the introduction course offered in MFOCS, this one is likely to focus more on the mathematical constructs and less on the security definitions. This does not mean that I do not consider those as important. I assume you master the ones covered in the introduction course, and we will also study some new definitions.

Based on the preferences you have expressed, we will cover the following content this year:

1. Elliptic curve cryptography, including isogeny-based cryptography.
2. Lattice-based cryptography.

I can provide references on other topics of your interest upon request, but these won't be considered part of the assessed material. I am also open to discuss potential dissertations topics.

1.2 Organisation

This course will be organized as a reading course.

You are expected to learn by yourselves or in groups from the reference material. The questions listed in Section 2 should help you identify the basic knowledge to acquire on each of the topics listed above.

The questions listed in Section 3 cover advanced material or they aim at developing your critical understanding. You are required to prepare a 60 min group presentation answering them. The presentation will be interrupted by discussions to check your understanding of the material covered, or to provide additional information. You are welcome to split the work between you, but everybody must have a good understanding of the whole content. Your understanding of this material may be assessed in the mini-project.

Here is a tentative schedule **(to be agreed)**, per week

1. Week 1: introduction (Wednesday 5pm)
2. Weeks 2,4,6,7 **(Wednesday 1pm)**: 1h for discussion on reading material (if requested)
3. Weeks 3,5,8 **(Wednesdays noon-2pm)**: 2h for presentations
4. Weeks 3,5,8 **(Wednesday 5pm)**: 1h for discussion on reading material (if requested)

By "requested" I mean that at least one of you has explicitly asked me to meet. All meetings will be open to everyone. We may occasionally have a meeting over Skype (definitely in Week 4, and not when you will be presenting).

1.3 Feedback welcome

I have been teaching Advanced Cryptography for MFOCS in 2015 and 2016 as a regular course, and as a reading course since 2017. I am still very keen to improve it, and your constructive feedback can help me with that. Feel free to provide feedback any time and be ensured it will be very welcome!

2 Guided Reading

2.1 Elliptic Curve Cryptography

2.1.1 Elliptic Curves (Week 1)

The main references for this part are my lecture slides from last year [16] and Silverman's book [19], Chapter III.

- What are the advantages of elliptic curve cryptography compared to cryptography based on the discrete logarithm over finite fields? Look at `www.keylength.com`
- Define an elliptic curve.

- Define Weierstrass equations. Can any elliptic curve be represented by a Weierstrass equation?
- Define reduced Weierstrass equations. Why are the characteristic 2 and 3 cases treated separately?
- Define the discriminant of an elliptic curve. How is this related to the smoothness condition?
- Define the j-invariant of an elliptic curve. In what sense is this an invariant?
- Suppose $\varphi : E_1 \to E_2$ is an isomorphism mapping one curve in Weierstrass coordinates to another curve in Weierstrass coordinates. What is the most general form of φ?
- Give the equation of one curve with j-invariant j.
- Define a group law on the points of an elliptic curve. Assuming the curve is in (reduced) Weierstrass coordinates, how do you add two points together? What is the neutral element? How do you define the inverse of a point? Understand the group operation both from geometric and algebraic points of view.
- Prove that this operation indeed defines a group. The only difficult property to prove is associativity. I personally like Sutherland's proof in his MIT lecture notes (`http://math.mit.edu/classes/18.783/2017/LectureNotes2.pdf`)
- Define scalar multiplication. Express scalar multiplication as a rational map.
- Define torsion points and division polynomials. How are these related?
- What is the group structure of elliptic curves over finite fields? How does the torsion look like over their algebraic closure?
- Understand Hasse's theorem. For any finite field K, why do we expect to have roughly K points on any curve defined over K? How accurate is this prediction in general?
- Understand Weil-Deligne's theorem. Assume you know the number of points of a curve over some finite field. How can you use this theorem to deduce the number of points over any extension field?
- Define an isogeny. Show that its kernel is a subgroup of the curve. Define the dual isogeny.
- Define an endomorphism. Define the Frobenius endomorphism. What is its characteristic equation?

- Describe the structure of endomorphisms.
- Use Sage: define a finite field, an elliptic curve, add points on the curve, perform scalar multiplications, compute the number of rational points.

2.1.2 Elliptic Curve Discrete Logarithm Problem (Weeks 1-2)

The main references for this part are my lecture slides from last year [16], and Blake-Serousi-Smart [1].

- Define the elliptic curve discrete logarithm problem (ECDLP)
- Define Computational/ decisional elliptic curve Diffie-Hellman problems
- Define the EC Diffie-Hellman protocol. What security guarantees does this protocol offer?
- Define EC ElGamal. Show that ElGamal is IND-CPA secure if ECDDH is hard. Show that ElGamal is not IND-CCA secure.
- Define ECDSA. What security guarantees does this protocol offer?
- Discuss the importance of using good randomness in ECDSA (attack on Sony's signatures, bitcoin theft due to Android's RNG weaknesses)
- Compare existing attack on DLP and ECDLP.
- What are NIST curves? Are these curves perfectly safe to use?
- Check other popular curves on `ttp://safecurves.cr.yp.to/`
- There are two main methods to generate suitable curves: the first one is to generate random coefficients and use point counting algorithms; the second one is the complex multiplication method. Discuss the advantages of each approach.
- Describe Schoof's point counting algorithm, and discuss its complexity.
- What is the quadratic twist of a curve? How are the number of points on one curve and its quadratic twist related?
- Use Sage to implement EC ElGamal encryption algorithm and Pollard's rho algorithm on ECDLP.

2.1.3 Algorithmic and implementation aspects (Week 2)

The main references for this part are my lecture slides from last year [16], and Blake-Serousi-Smart [1].

- Describe the basic double-and-add algorithm. What is its complexity?
- Study various methods to accelerate scalar multiplication [1, Chapter 4].

- Define Edwards curves. Are they as generic as Weierstrass curves? What are the advantages of using Edwards curves formulae over Weirstrass curves formulae?
- Use Sage to implement a scalar multiplication method of your choice for Edwards curves.

2.1.4 Pairing-based cryptography (Week 3)

The main references for this part are my lecture slides from last year [16], the survey [6], and references therein.

- Define pairings. What are their main properties? What are Type I, Type II, Type III pairings?
- How can pairings help to build a 3-partite Diffie-Hellman protocol? Can this protocol be built with any type of pairing?
- What computational assumptions are required for the security of the 3-partite Diffie-Hellman protocol? How can these assumptions be related to other computational assumptions?
- Can we extend this protocol to a 4-partite Diffie-Hellman protocol?
- What is the main idea behind identity-based cryptography? What problem does it aim to solve and what new problems does it create?
- Study Boneh-Franklin ID-based encryption protocol. Check that decryption of a valid ciphertext gives back the corresponding plaintext.
- What are the security requirements for ID-based encryption? Do the requirements posed by Boneh-Franklin look sufficient to you?
- What are the security requirements for the hash function in Boneh-Franklin protocol?
- Study Boneh-Lynn-Sacham signatures. What security arguments/proofs can be provided for this protocol?
- Use Sage to implement one pairing-based protocol of your choice.

2.1.5 Elliptic curve pairings (Week 3)

The main references for this part are my lecture slides from last year [16], and Galbraith's survey in Blake-Serousi-Smart [1][Vol 2, Chapter IX].

- Define a divisor, its degree, its support.
- Define the divisor of a function, a principal divisor

- Show that the map sending an elliptic curve point P to the divisor $(P)-(0)$ is a group homomorphism up to principal divisors
- Let $D = \sum_P n_P(P)$ be a degree 0 divisor on E. Then $D \sim 0$ if and only if $\sum_P [n_P]P = O$.
- State the Weil reciprocity.
- Define the Tate pairing. State and prove its main properties.
- Define the reduced Tate pairing. Why is this definition useful?
- Define the embedding degree. What is its importance for cryptographic applications? What is expected about the embedding degree of random curves?
- Define the Weil pairing. State and prove its main properties.
- Show how the Tate and Weil pairings reduce ECDLP to DLP over a finite field. Does this reduction necessarily give a subexponential algorithm for ECDLP?
- Can we use the Tate and Weil pairings for protocols that require a symmetric pairing?
- What is a distorsion map and how can it help buidling a symmetric pairing?
- Study how the Tate and Weil pairing can be efficiently computed using Miller's algorithm
- What are the requirements for elliptic curves to have suitable cryptographic pairings? Give some examples of suitable families.

2.1.6 Isogeny-based cryptography (Week 4)

The main references for this part are my lecture slides from 2016 [18] and the recent survey [7].

- Define an isogeny; give a standard representation as a rational map. Define the kernel, the degree. Define the dual isogeny.
- What are Vélu's formulae? In general, what is the computational cost of these formulae?
- Define the endomorphism ring computation problem. How should the answer to this problem be returned? Is this representation efficient in general?
- Define isogeny graphs and state their main properties.

- Sketch Kohel's algorithm to compute endomorphisms of supersingular elliptic curves. What is its cost?
- What is the isogeny computation problem?
- Why are isogeny problems appealing for cryptography?
- Describe Charles-Goren-Lauter hash function based on isogenies. What are the security arguments for this function? What is known about its security?
- Describe the supersingular isogeny key exchange protocol (SIDH). Why are Alice and Bob's secret kernels of order coprime to each other? Could we complete the protocol without exchanging extra points (i.e. exchanging only j-invariants)? What are the security guarantees provided by the protocol?
- Show how to derive an encryption scheme from this key exchange protocol.
- Study the results in [17]. What is the impact of extra points revealed in SIDH protocol with respect to security?
- Implement CGL hash function in Sage.
- How does CSIDH algorithm [3] compare to SIDH?

2.2 Lattice-based cryptography

Lattices were first used in cryptography for cryptanalysis purposes. Constructions based on lattice problems followed , some of which were quickly broken. Considerable progress has now been made, and lattice-based cryptography is emerging as a front-runner to replace currently used cryptography before quantum computers become a reality. Many challenges remain, including understanding how the numerous parameters involved in lattice-based cryptography protocols impact their security.

2.2.1 Lattices and hard lattice problems (Week 5)

The main reference for the following questions are my lecture notes from 2016 (particularly the first section). Other useful references are Micciancio and Goldwasser's book [13] and Oded Regev's 2014 lecture notes at Tel Aviv University.

- Define a lattice, its dimension and its rank.
- A lattice is usually given by a basis, which itself is often represented by a matrix. Explain how two such matrices for the same lattice are related to each other.
- Define a fundamental parallelepiped corresponding to a basis, and show that its volume is independent of the basis choice. Define the determinant of the lattice.

- Define scalar product, Euclidean norm.
- Define shortest vector problem (SVP). How does its complexity depend on the choice of basis?
- State and prove the convex body theorem.
- State and prove Minkowski's first theorem. How is it generalized by Minkowski's second theorem?
- What is the Gaussian heuristic? Compare it with the bound provided by Minkowski's theorem. Can you produce any counter-example? Verify the Gaussian heuristic experimentally on "random lattices" of dimension 2.
- Define the closest vector problem (CVP). How does its complexity depend on the choice of basis?
- Define decisional versions of SVP and CVP. What are the known relationships between these problems? What is know about their hardness?
- Define approximate versions of SVP and CVP. For which parameters are those problems known or believed to be NP-hard? What about the parameters used in cryptography?
- What are "worst-case hardness" and "average-case hardness". Reflect on the guarantees offered by both.
- Define a modular lattice. Give one example of a modular lattice and one example of a non modular lattice.
- Define the short integer solution problem (SIS). What do we know about its hardness? How is it connected to other lattice problems?
- Define the learning with error problem (LWE). What do we know about its hardness? How is it connected to other lattice problems? What is the impact of the noise and its distribution? Define its decisional version.
- Define ideal lattices. Give an example of an ideal lattice in dimension 3. Show that this lattice is modular.

The main reference for the following questions are my lecture notes from 2016 (particularly the fourth section). Other useful references are Micciancio and Goldwasser's book [13] and Oded Regev's 2014 lecture notes at Tel Aviv University.

- What are the guarantees provided by Babai's nearest plane algorithm? (see third section in my slides)
- Show that Decisional CVP and Search CVP reduce to each other.

- How can we solve any subset-sum problem given a Decisional CVP algorithm? Conclude that Search and Decisional CVP are NP complete problems.
- How can we solve any Decisional SVP given a Decisional CVP algorithm? To what extent is the converse true as well?

2.2.2 Lattice-based cryptographic constructions (Week 6)

The main reference for this section are my lecture notes from 2016 (particularly the second section). Other useful references are Peikert's and Micciancio-Regev's surveys [15, 14].

- Recall the main security properties for a cryptographic hash function.
- Spend some time trying to build a one-way function from one of the lattice problems studied in Week 5. What security guarantees can you obtain for your construction(s)?
- Study Ajtai's hash function. How is collision resistance derived from the hardness of SIS?
- Study Ajtai's worst case to average case reduction. What guarantees does it provide on the hash function?
- Describe an attack on Ajtai's hash function when using ideal lattices. How does this attack not contradict the above guarantees? How can the construction be (hopefully) repaired?
- Recall the main security properties of a public key encryption scheme.
- Spend some time trying to build a public key encryption scheme with lattices. What will be the secret key and the public key? What form will messages and ciphertexts have?
- Describe GGH cryptosystem. Verify correctness of the scheme. What is known about its security?
- What is the Fujisaki-Okamoto transform?
- Describe the NTRU cryptosystem. Verify its correctness. How can it be seen as a lattice-based cryptosystem? What is known about its security?
- Study Regev (LWE-based) cryptosystem. Verify its correctness. What is known about its security? How should one choose the parameters?
- Compare GGH, NTRU and Regev cryptosystems.
- Recall the main security properties of a signature scheme.

- Spend some time trying to build a signature scheme with lattices. What will be the secret key and the public key? What form will messages and signatures have?
- Describe GGH signatures.
- Understand how GGH and NTRU signatures could be broken by Nguyen and Regev.
- Define "rejection sampling". What is the purpose of this technique?
- Recall the RSA encryption scheme and describe its homomorphic properties.
- What is a fully homomorphic encryption (FHE) scheme? Reflect on potential applications of such a scheme.
- What is a somewhat homomorphic encryption? Reflect on potential applications of such a scheme.
- What is the purpose of the bootstrapping procedure invented by Gentry? How is it used?
- Give a simple example of an FHE scheme.
- Are existing FHE schemes good enough for applications?

2.2.3 Solving hard lattice problems (Week 7)

The main reference for this section are my lecture notes from 2016 (particularly the third section). Another useful reference is Joux's book [10].

- What is the difference between lattice reduction algorithms and exact SVP solvers? How do they complement each other?
- What is the purpose of LLL algorithm?
- Recall Gram-Schmidt orthogonalization process.
- Study Gauss reduction process for lattices in dimension 2. Apply the algorithm on the example of slide 68 and on other examples of your choice. Understand why the algorithm terminates, and why it returns a shortest vector.
- When do we say that a basis is reduced in dimension 2? Understand that condition geometrically.
- Define δ-LLL-reduced basis. Define Lovasz condition. Are LLL-reduced basis elements ordered from shortest to largest? What is the impact of the parameter δ? Are reduced bases in dimension 2 also LLL-reduced bases? When is a δ-LLL reduced basis also a δ'-LLL reduced basis?

- What guarantees do LLL-reduced bases provide in terms of containing short elements? Give two bounds on the length of the first basis vector, with respect to λ_1 and the lattice determinant. Compare the later guarantee with Minkowski's bound and the Gaussian heuristic.
- Study the LLL algorithm. Apply it by hand to a small dimensional example.
- Implement your own version of LLL in Sage and observe its behaviour on small dimensional examples.
- Understand why the LLL algorithm terminates, and why the output has the desired form.
- "LLL performs much better in practice than in theory." What is/are the efficiency metric(s) considered here?
- One method to find the shortest vector of a lattice is to consider any lattice vector within a box. Explain how to conduct this enumeration process with absolute guarantees that the shortest vector will be found. How many vectors will the enumeration process need to consider? Explain methods to reduce this number.
- What are sieving algorithms? Compare them with enumeration algorithms from an efficiency point of view.
- What are birthday attacks in cryptography?
- Study Wagner's generalized birthday attack algorithm [20].
- How can this algorithm be used to compute short vectors in a modular lattice?
- Study Babai's "nearest plane" algorithm. What is the purpose of the algorithm? How good is the result provided by the algorithm? Explain how this depends on existing lattice reduction algorithms.

2.2.4 Cryptanalysis with lattices (Week 8)

The main reference for this section are my lecture notes from 2016 (particularly the fifth section). Another useful reference is Joux's book [10].

- Define the subset sum problem. What do we know about its hardness?
- Describe Merkle-Hellman cryptosystem. What is the secret key and what is the public key? What forms have messages and ciphertexts? Verify correctness. Try to attack the scheme on your own.
- Study the algorithm in slide 123: an algorithm to compute short relations, namely small λ_i such that $\sum_i \lambda_i v_i = 0$ for given integers or vectors v_i. What should the parameter K be chosen? How much does that answer depend on the actual lattice reduction algorithm used?

- Is the algorithm guaranteed to succeed? May it succeed for smaller and larger values of K as well?
- Describe the knapsack-based hash function. Try to attack the hash function on your own.
- Study the attack provided in slide 126. What should the parameter K be chosen? How much does that answer depend on the actual lattice reduction algorithm used?
- Is the algorithm guaranteed to succeed? May it succeed for smaller and larger values of K as well? How does it perform in practice?
- Study Coppersmith's small root attacks. What is the problem considered?
- Start with univariate polynomials. What are the key ideas in the algorithm?
- Describe the lattice constructed in the algorithm. Can you think of other ways to construct a lattice and achieve similar results?
- Is the algorithm guaranteed to succeed? Study its analysis.
- What is the complexity of this algorithm? How does the quality of the bound we have on the root affect the efficiency of the algorithm? How does the degree of the polynomial affect it?
- Sketch how the approach can be extended to multivariate polynomials as well. How will the complexity be changed? Is the resulting algorithm guaranteed to succeed?
- Describe some applications of Coppersmith's small root attacks to attacks on the RSA cryptosystem.
- Recall DSA, ECDSA and ElGamal signature protocols.
- DSA, ECDSA and ElGamal all use ephemeral secret keys. Study the attacks in [9] when parts of these ephemeral keys are leaked. Describe the attack model and the main consequences of the attack.
- What are the key ideas in the algorithm?
- Describe the lattice constructed in the algorithm. Can you think of other ways to construct a lattice and achieve similar results?
- Is the algorithm guaranteed to succeed? Study its analysis.
- What is the complexity of this algorithm? How does the amount of information we have on the ephemeral keys affect the efficiency?

3 Group Presentations

3.1 Elliptic Curve Cryptography (I) (Week 2)

1. Read the survey on recent elliptic curve discrete logarithm results [8]. Summarize the current state of the art and main open problems.

2. Describe the Elliptic Curve Factorization Method. What is its complexity? Is it used today to factor big numbers? Implement the method using Sage. Compare the efficiency of your implementation with the internal routine.

3. Study Golwasser-Killian primality proofs. Implement the method in Sage. Study the efficiency of your implementation.

4. Describe Ciet-Joye fault's attack [4]. What is the attack model? What can be achieved with this attack?

Prepare a presentation to show your results in Week 2.

3.2 Elliptic Curve Cryptography (II) (Week 5)

1. Pairings have resulted in a tremendous amount of cryptographic schemes. Browse the web to find some applications and list them into categories. Choose one application that has not been covered in the lecture slides and describe it.

2. Read Koblitz-Menezes' opinion on the proliferation of pairing assumptions [12] and summarize their main arguments.

3. Based on the papers [5, 6], provide constructions of elliptic curve pairings that are suitable for the protocol chosen in Question 1. Explain the rationales behind your choices, and discuss the resulting security and efficiency both asymptotically and at the 128-bit security level. In your parameter estimation, be sure to take into account recent progress in discrete logarithm computation [11]. How is the discussion in Q2 relevant here?

4. Implement Miller's algorithm in Sage. Compare the cost of a pairing and a scalar multiplication for your implementation. Does this fit with the a priori theoretical estimations?

Prepare a presentation to show your results in Week 5.

3.3 Lattice-based Cryptography (Week 8)

1. Browse `http://www.latticechallenge.org/`. What are the largest size of lattice challenges that can be solved? Are ideal lattices easier to solve? What is the gap between solving exact SVP and approximate SVP? What sort of algorithms have been used for the records?

2. In Sage, lattices are stored as matrices, where the row vectors of a matrix generate the lattice. You can generate random matrices with the function `random_matrix` and apply LLL on them with the function `LLL`.

   Apply the LLL algorithm on random matrices, varying the sizes of the random matrices and the δ parameter in LLL. Study the impact of these parameters on the shortest vector found and on the running time of LLL. Can you explain your observations from the theory?

3. The following Sage code implements a knapsack-based hash function. The first program `Parameters(n,mu)` generates `params` used for hashing values from $\{0,1\}^n$. Larger values of `mu` increase the output length of the hash function. The second program `KnapsackHash(A,x)` computes the hash of x.

```
def Parameters(n,mu):
    A = list();
    for i in range(0,n):
        A.append(randint(0,2**mu-1))
    return [vector(A),n,mu]

def KnapsackHash(params,x):
    A = params[0]
    n = params[1]
    mu = params[2]
    k = ceil(log(n,2))+mu
    if n != len(x):
        return "The input vectors are not the same length!"
    z = 0;
    for i in range(0,len(x)):
        z = z + A[i]*x[i];
    z = z.bits()
    while len(z) < k:
        z.append(0)
    return z
```

   (a) Write a program which takes the output of `Parameters` and finds a collision on the resulting hash function using the LLL algorithm.

   (b) Test your program on `params = Parameters(n,mu)` for $(n, mu) = (10, 10), (20, 20), (40, 40)$. What are the largest values of (n, mu) for which your program finds a collision?

4. Study Boneh-Durfee's paper [2] and describe one of their attacks. Implement the attack in Sage. Study the performances of your implementation experimentally and compare them with the theoretical predictions.

Prepare a presentation to show your results in Week 8.

References

[1] Ian F. Blake, G. Seroussi, and N. P. Smart. *Elliptic curves in cryptography.* Cambridge University Press, New York, NY, USA, 1999.

[2] Dan Boneh and Glenn Durfee. Cryptanalysis of RSA with private key d less than $N^{0.292}$. In Jacques Stern, editor, *Advances in Cryptology - EUROCRYPT '99, International Conference on the Theory and Application of Cryptographic Techniques, Prague, Czech Republic, May 2-6, 1999, Proceeding*, volume 1592 of *Lecture Notes in Computer Science*, pages 1–11. Springer, 1999.

[3] Wouter Castryck, Tanja Lange, Chloe Martindale, Lorenz Panny, and Joost Renes. CSIDH: an efficient post-quantum commutative group action. In Thomas Peyrin and Steven D. Galbraith, editors, *Advances in Cryptology - ASIACRYPT 2018 - 24th International Conference on the Theory and Application of Cryptology and Information Security, Brisbane, QLD, Australia, December 2-6, 2018, Proceedings, Part III*, volume 11274 of *Lecture Notes in Computer Science*, pages 395–427. Springer, 2018.

[4] Mathieu Ciet and Marc Joye. Elliptic curve cryptosystems in the presence of permanent and transient faults. *IACR Cryptology ePrint Archive*, 2003:28, 2003.

[5] David Freeman, Michael Scott, and Edlyn Teske. A taxonomy of pairing-friendly elliptic curves, 2006.

[6] Steven D. Galbraith, Kenneth G. Paterson, and Nigel P. Smart. Pairings for cryptographers. *Discrete Applied Mathematics*, 156(16):3113–3121, 2008.

[7] Steven D. Galbraith, Christophe Petit, and Javier Silva. Identification protocols and signature schemes based on supersingular isogeny problems. In Tsuyoshi Takagi and Thomas Peyrin, editors, *Advances in Cryptology - ASIACRYPT 2017 - 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part I*, volume 10624 of *Lecture Notes in Computer Science*, pages 3–33. Springer, 2017.

[8] Pierrick Gaudry and Steven D. Galbraith. Recent progress on the elliptic curve discrete logarithm problem. To appear in Design, Codes and Cryptography, 2015.

[9] Nick Howgrave-Graham and Nigel P. Smart. Lattice attacks on digital signature schemes. *Des. Codes Cryptography*, 23(3):283–290, 2001.

[10] Antoine Joux. *Algorithmic Cryptanalysis.* Chapman & Hall/CRC, 2009.

[11] Taechan Kim and Razvan Barbulescu. Extended tower number field sieve: A new complexity for the medium prime case. In *Advances in Cryptology -*

CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part I, pages 543–571, 2016.

[12] Neal Koblitz and Alfred J. Menezes. The brave new world of bodacious assumptions in cryptography. *Notices of the American Mathematical Society*, 57:357–365, 2010.

[13] Daniele Micciancio and Shafi Goldwasser. *Complexity of Lattice Problems: A Cryptographic Perspective*. Springer, 2002.

[14] Daniele Micciancio and Oded Regev. Lattice-based cryptography. In D.J. Bernstein, J. Buchmann, and E. Dahmen, editors, *Post Quantum Cryptography*, pages 147–191. Springer, 2009.

[15] Chris Peikert. A decade of lattice cryptography. *Foundations and Trends in Theoretical Computer Science*, 10(4):283–424, 2016.

[16] Christophe Petit. Elliptic curve cryptography. Lecture slides for MFOCS, oxford, 2016.

[17] Christophe Petit. Faster algorithms for isogeny problems using torsion point images. In *Advances in Cryptology - ASIACRYPT 2017 - 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part II*, pages 330–353, 2017.

[18] Christophe Petit, Michiel Kosters, and Ange Messeng. Algebraic approaches for the elliptic curve discrete logarithm problem over prime fields. In Chen-Mou Cheng, Kai-Min Chung, Giuseppe Persiano, and Bo-Yin Yang, editors, *Public-Key Cryptography - PKC 2016 - 19th IACR International Conference on Practice and Theory in Public-Key Cryptography, Taipei, Taiwan, March 6-9, 2016, Proceedings, Part II*, volume 9615 of *Lecture Notes in Computer Science*, pages 3–18. Springer, 2016.

[19] Joseph Silverman. *The Arithmetic of Elliptic Curves*. Springer Verlag, 1986.

[20] David Wagner. A generalized birthday problem. In Moti Yung, editor, *CRYPTO*, volume 2442 of *Lecture Notes in Computer Science*, pages 288–303. Springer, 2002.

A Introduction to Sage

Sage is an open-source computer algebra system that is very popular in research, an in particular among cryptography researchers. For an introduction, have a look at the slides written by Benjamin Pring (University of Bath and Florida Atlantic University) for MFOCS students in 2015. Sage is installed on the Mathematical Institute's computing servers, which should be accessible to you.

There are of course other computer algebra systems, and when some programming task is requested you are welcome to use any system you prefer. (Personally I can read Sage code but I am more familiar with Magma, which is also available on the Mathematical Institute servers.)