Seven Weeks of Lecture Notes on Core Hard Problems in Cryptography

Contents

1 Course Conventions and Global Preliminaries (use throughout)

1.1 Security parameter, asymptotics, and experiments

Definition 1.1 (Negligible)**.** *$\mu : \mathbb{Z}_{\geq 0} \to \mathbb{R}_{\geq 0}$ is negligible if $\forall c > 0\ \exists \lambda_0\ \forall \lambda \geq \lambda_0 :\ \mu(\lambda) < \lambda^{-c}$. Write $\mu(\lambda) = \mathrm{negl}(\lambda)$.*

Definition 1.2 (PPT)**.** *A probabilistic polynomial-time (PPT) algorithm runs in time* $\mathrm{poly}(\lambda)$ *on inputs of size* $\mathrm{poly}(\lambda)$.

1.2 Search vs. decision vs. distinguishing

We phrase hardness as one of: (i) *search* (find witness), (ii) *decision* (existence), (iii) *distinguishing* (tell apart two distributions).

1.3 L-notation (subexponential)

For $N \to \infty$,

$$L_N[\alpha, c] := \exp\Big((c + o(1))(\log N)^{\alpha}(\log\log N)^{1-\alpha}\Big).$$

We will use $L_N[1/3, \cdot]$ for NFS-type algorithms and $L_N[1/2, \cdot]$ for QS/index-calculus.

1.4 Implementation caveat

All "hard problems" become easy under side-channel leakage, biased randomness, invalid-curve attacks, fault attacks, etc. We separate *mathematical* hardness from *implementation* soundness.

2 Week 1: Integer Factorization (RSA/Rabin) — Theory, Algorithms, and Attacks

2.1 Learning objectives

By the end of Week 1 you should be able to:

- Formally state factoring and closely related computational tasks (order-finding, $\varphi(N)$ recovery).
- Prove equivalences: factoring $\Leftrightarrow$ computing $\varphi(N)$ (for semiprimes), Rabin inversion $\Rightarrow$ factoring.
- Explain the structure of modern classical factoring: (i) ECM, (ii) QS, (iii) GNFS at a black-box level.
- Explain Shor's reduction factoring $\to$ order finding.

2.2 Core definitions and reductions

Definition 2.1 (Factoring (search)). *Given $N \in \mathbb{Z}_{\geq 2}$ composite, output d with $1 < d < N$ and $d \mid N$.*

Definition 2.2 (RSA modulus distribution). *$\mathcal{D}_\lambda$ samples λ-bit primes p, q and outputs $N = pq$.*

Proposition 2.3 (Computing $\varphi(N)$ factors semiprimes). *If $N = pq$ with distinct primes and $\varphi(N)$ is known, then p, q can be recovered in polynomial time.*

Proof sketch. We have $\varphi(N) = (p-1)(q-1) = N - (p+q) + 1$, hence $p + q = N - \varphi(N) + 1$. Solve $X^2 - (p+q)X + N = 0$ over $\mathbb{Z}$. □

Proposition 2.4 (Rabin inversion implies factoring). *Let $N = pq$ with odd primes. An oracle that inverts $x \mapsto x^2 \bmod N$ on a non-negligible fraction of quadratic residues yields a factoring algorithm.*

2.3 Lecture A (Number theory for RSA/Rabin)

- Euler/Carmichael functions, CRT, structure of $(\mathbb{Z}/N\mathbb{Z})^\times$.
- Orders mod p and mod pq, and why order-finding is central.
- Quadratic residues, Jacobi symbol, and why Rabin has 4 square-roots mod N.

2.4 Lecture B (Classical factoring toolkit)

Stage 1: special-purpose methods. Pollard ρ; Pollard $p-1$ (smoothness of $p-1$); ECM (smoothness of group order).

Stage 2: sieve methods.

- Quadratic Sieve (QS): relations $x^2 \equiv y^2 \pmod{N}$ from smooth values of $x^2 - N$; complexity $L_N[1/2, 1]$.
- GNFS: polynomial selection, sieving, sparse linear algebra, square root; complexity $L_N\left[1/3, \sqrt[3]{64/9}\right]$.

2.5 Lecture C (Quantum factoring via order finding)

- Reduction: factoring $\to$ order finding.
- Period finding and QFT intuition (as a Fourier sampling statement).
- Classical post-processing: $\gcd(a^{r/2} \pm 1, N)$.

2.6 Recitation / worked examples

Example 2.5 (Recovering p, q from $\varphi(N)$)**.** *Choose* $N = 221 = 13 \cdot 17$. *Then* $\varphi(N) = 192$ *and* $p + q = 221 - 192 + 1 = 30$, *solve* $X^2 - 30X + 221 = 0$.

2.7 Problem set (Week 1)

1. Prove the proposition "computing $\varphi(N)$ factors semiprimes".
2. Show: if you can compute $\mathrm{ord}_N(a)$ for random a, you can factor N with non-negligible probability.
3. Implement Pollard ρ and ECM preprocessing; report empirical runtimes on random semiprimes with one 30–60 bit factor.
4. (Theory) Explain why QS produces a congruence of squares and why linear algebra over $\mathbb{F}_2$ appears.

3 Week 2: Discrete Logarithms — Generic Groups, Finite Fields, and Elliptic Curves

3.1 Learning objectives

- State DLP, CDH, DDH and relationships among them.
- Prove correctness and complexity of Pohlig–Hellman.
- Explain generic lower bounds (birthday-type) and why Pollard ρ is optimal in generic groups.
- Explain index calculus at a conceptual level; distinguish finite-field DLP vs ECDLP.
- Explain Shor for abelian hidden subgroup as it applies to DLP.

3.2 Definitions

Definition 3.1 (DLP)**.** *Let* $G = \langle g \rangle$ *of order* n. *Given* $g, h \in G$, *find* $x \in \mathbb{Z}_n$ *with* $g^x = h$.

Definition 3.2 (CDH and DDH)**.** *Given* (g, g^a, g^b) *compute* g^{ab} *(CDH). Distinguish* (g, g^a, g^b, g^{ab}) *from* (g, g^a, g^b, g^c) *(DDH).*

3.3 Lecture A (Generic algorithms and lower bounds)

- Baby-step/giant-step: meet-in-the-middle, $\tilde{O}(\sqrt{n})$ time/memory.
- Pollard ρ: random walks, cycle finding, $\tilde{O}(\sqrt{n})$ time, low memory.
- Generic lower bounds (Shoup-type): any generic DLP algorithm needs $\Omega(\sqrt{n})$ operations.

3.4 Lecture B (Pohlig–Hellman and subgroup structure)

- Reduction when $n = \prod p_i^{e_i}$; solve modulo each factor, CRT.
- Practical implication: choose prime-order (or nearly prime-order) groups.

3.5 Lecture C (Index calculus vs ECDLP; pairing pitfalls)

- Index calculus: factor base, relation collection, linear algebra, individual logs.
- NFS-DL for prime fields: $L_p\left[1/3, \sqrt[3]{64/9}\right]$.
- Elliptic curves: why generic $\tilde{O}(\sqrt{n})$ dominates for well-chosen curves.
- MOV/Frey–Rück: reduction to finite-field DLP for special curves (avoid via curve selection).

3.6 Problem set (Week 2)

1. Prove correctness of Pohlig–Hellman and analyze complexity in terms of prime factors of n.
2. Implement Pollard ρ for a cyclic group $\mathbb{Z}_p^\times$ and measure runtime vs $\sqrt{p}$ scaling.
3. Show: if DDH holds in G, then ElGamal is IND-CPA secure (in the standard reduction framework).
4. (Bonus) Explain why pairings break DDH on some pairing-friendly curves.

4 Week 3: Lattices — Geometry of Numbers, SVP/CVP, and Cryptographic Problems (SIS/LWE)

4.1 Learning objectives

- Work fluently with lattice bases, determinant, dual lattice, successive minima.
- State SVP/CVP and approximation variants; connect to Minkowski's theorem.
- Understand LLL and BKZ at the conceptual level; know what a "root Hermite factor" means (informally).
- State SIS and LWE formally; classify attacks (primal/dual/hybrid/BKW).

4.2 Lecture A (Geometry of numbers foundations)

Definition 4.1 (Lattice). $\mathcal{L}(B) = \{Bz : z \in \mathbb{Z}^d\}$ *for invertible* $B \in \mathbb{R}^{d\times d}$.

Definition 4.2 (Dual lattice). $\mathcal{L}^* = \{y \in \mathbb{R}^d : \langle y, x\rangle \in \mathbb{Z}\ \forall x \in \mathcal{L}\}$.

Theorem 4.3 (Minkowski (first theorem)). *Let* $\mathcal{L} \subset \mathbb{R}^d$ *be a full-rank lattice. Then*

$$\lambda_1(\mathcal{L}) \leq \sqrt{d}\ \det(\mathcal{L})^{1/d}.$$

4.3 Lecture B (SVP/CVP and algorithms)

- Exact SVP/CVP definitions; NP-hardness for approximation in various norms (contextual, not proved here).
- LLL: polynomial-time reduction, guarantees, and what "reduced basis" means.
- BKZ: block reduction, why it dominates practical cryptanalysis.
- Enumeration and sieving: exponential-time paradigms; time/memory tradeoffs.

4.4 Lecture C (SIS/LWE and attack taxonomy)

Definition 4.4 (SIS)**.** *Given $A \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$, find $x \in \mathbb{Z}^m \setminus \{0\}$ with $Ax \equiv 0 \pmod q$ and $\|x\| \le \beta$.*

Definition 4.5 (LWE (decision))**.** *Given $(a_i, b_i) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$, distinguish LWE samples $b_i = \langle a_i, s\rangle + e_i \pmod q$ from uniform, where $s \xleftarrow{\$} \mathbb{Z}_q^n$ and $e_i \leftarrow \chi$.*

Attacks. Primal embedding (SVP/CVP), dual distinguishing, hybrid guessing, BKW sample-combining.

4.5 Problem set (Week 3)

1. Compute $\det(\mathcal{L})$ and $\mathcal{L}^*$ for a given 2×2 basis; verify $\det(\mathcal{L}^*) = 1/\det(\mathcal{L})$.
2. Prove Minkowski's bound for λ_1 in dimension 2 using area arguments.
3. Implement LLL and use it to find short vectors in random lattices; report approximation factors empirically.
4. Sketch a primal embedding reduction of LWE to (approximate) CVP; identify where parameters enter.

5 Week 4: Codes — Syndrome Decoding, McEliece, and ISD Cryptanalysis

5.1 Learning objectives

- Manipulate generator/parity-check descriptions; compute syndromes and decode small examples.
- State Syndrome Decoding (SD) and Minimum Distance Problem (MDP) formally.
- Derive Prange ISD success probability; understand Stern/Dumer/BJMM at a structural level.
- Understand "structural attacks" vs "generic ISD".

5.2 Lecture A (Linear codes and decoding)

Definition 5.1 (Linear code)**.** *$C \subseteq \mathbb{F}_q^n$ a k-dimensional subspace. Parity-check H satisfies $C = \{c : Hc^\top = 0\}$.*

Definition 5.2 (Syndrome)**.** *Given received word $r = c+e$, syndrome $s = Hr^\top = He^\top$ depends only on error e.*

5.3 Lecture B (Hard problems and McEliece)

Definition 5.3 (Syndrome Decoding (SD)). *Given (H, s, t), find e with $He^\top = s$ and $w_H(e) \leq t$.*

McEliece (context). Public key hides a structured code (e.g. Goppa) behind random scrambling; security reduces to SD on "random-looking" codes.

5.4 Lecture C (ISD: Prange → Stern/Dumer/BJMM)

Prange analysis. Let I be an "information set" of size k avoiding error positions. Success probability $\approx \binom{n-t}{k}/\binom{n}{k}$; expected trials is its inverse.

Modern ISD. Stern/Dumer/BJMM use meet-in-the-middle and partial collisions to reduce exponent.

5.5 Problem set (Week 4)

1. Derive Prange success probability; compute expected work for small toy parameters.
2. Implement a toy ISD (Prange) on binary linear codes and compare to brute force.
3. Explain how "distinguishers" can break structured code disguises; give an example of what a distinguisher might measure.

6 Week 5: Isogenies — Elliptic Curves, Isogeny Graphs, and Modern Attacks

6.1 Learning objectives

- Work with elliptic curves over finite fields: group law, torsion, endomorphisms (at a high level).
- Define isogenies, degree, kernels; compute small isogenies via Vélu formulas (conceptually).
- Understand supersingular isogeny graphs and why path-finding is hard.
- Distinguish problem classes: supersingular path-finding vs commutative class-group action (CSIDH-style).

6.2 Lecture A (Elliptic curves essentials)

Definition 6.1 (Elliptic curve). *Over $\mathbb{F}_q$, (char $\neq 2, 3$) $E : y^2 = x^3 + ax + b$ with $\Delta \neq 0$; $E(\mathbb{F}_q)$ is finite abelian.*

6.3 Lecture B (Isogenies: kernels, degrees, evaluation)

Definition 6.2 (Isogeny). *A nonconstant morphism $\varphi : E_1 \to E_2$ that is also a group homomorphism.*

Proposition 6.3 (Degree multiplicativity). $\deg(\varphi \circ \psi) = \deg(\varphi)\deg(\psi)$.

Computation. For separable isogenies, kernels determine isogenies; Vélu formulas give explicit maps from kernel generators.

6.4 Lecture C (Hardness and attacks)

- Supersingular ℓ-isogeny graph: regular expander-like graph; problem resembles hidden path.
- Meet-in-the-middle / bidirectional search (Delfs–Galbraith style) and heuristic exponents.
- Protocol-specific breaks: emphasize separating “problem family” from “scheme instantiation”.
- Quantum: hidden-shift style algorithms in commutative settings (Kuperberg-type subexponential).

6.5 Problem set (Week 5)

1. For a small prime field, enumerate $E(\mathbb{F}_p)$ for a toy curve and compute group structure.
2. Prove that finite subgroups correspond to separable isogenies (state precisely; prove a special case).
3. (Conceptual) Explain why expander mixing heuristics suggest meet-in-the-middle complexity for random path problems.

7 Week 6: Multivariate (MQ) — Algebraic Geometry Viewpoint and Cryptanalysis

7.1 Learning objectives

- Formally state MQ and interpret it as solving $V(I)$ for an ideal $I = \langle f_1, \dots, f_m \rangle$.
- Understand Gröbner bases as an elimination engine; connect term orders to solving.
- Understand XL/relinearization and hybrid guessing as complexity trade-offs.
- Recognize structural reductions (MinRank/Kipnis–Shamir-style) in trapdoor designs.

7.2 Lecture A (MQ as a variety problem)

Definition 7.1 (MQ). *Given $f_1, \dots, f_m \in \mathbb{F}_q[x_1, \dots, x_n]$ with $\deg f_i \leq 2$, find $x \in \mathbb{F}_q^n$ such that $f_i(x) = 0$ for all i.*

Geometric lens. Solutions are $\mathbb{F}_q$-rational points of the affine variety $V(I)$.

7.3 Lecture B (Gröbner bases: elimination and degree of regularity)

- Ideals, term orders, leading terms; reduced Gröbner basis as canonical.
- Solving by elimination under lex order; change-of-order strategies.
- Complexity drivers: degree of regularity, sparsity, semi-regular heuristics.

7.4 Lecture C (XL, hybrid, and structural attacks)

- XL: multiply equations by monomials up to degree D, linearize monomials.
- Hybrid: guess k variables, solve remaining by Gröbner/XL.
- MinRank-type reductions: represent quadratic forms via matrices; exploit low rank structures.

7.5 Problem set (Week 6)

1. Solve a small MQ system over $\mathbb{F}_2$ by (i) brute force, (ii) linearization, (iii) a CAS Gröbner basis (if allowed).
2. Show how quadratic polynomials correspond to bilinear forms / symmetric matrices in odd characteristic.
3. Analyze hybrid complexity for guessing k variables: $q^k \cdot T(n-k)$.

8 Week 7: Hash Functions — Security Notions, Generic Bounds, and Structural Attacks

8.1 Learning objectives

- State CR/SPR/OW formally as computational games.
- Prove the birthday bound (collision probability) and expected preimage complexity for random functions.
- Understand Merkle–Damgård iteration and why length extension exists.
- Understand why HMAC fixes length extension and what "random oracle idealization" means.
- Understand quantum impacts (Grover) on preimage security levels.

8.2 Lecture A (Formal games)

Definition 8.1 (Hash family). *$\{H_\lambda\}$ with $H_\lambda : \{0,1\}^* \to \{0,1\}^{n(\lambda)}$ efficiently computable.*

Definition 8.2 (Collision resistance). *For all PPT $\mathcal{A}$,*

$$\mathrm{Prob}\big[(x,x') \leftarrow \mathcal{A}(1^\lambda) : x \neq x' \wedge H_\lambda(x) = H_\lambda(x')\big] = \mathrm{negl}(\lambda).$$

Definition 8.3 (Preimage resistance). *For all PPT $\mathcal{A}$,*

$$\mathrm{Prob}\big[y \xleftarrow{\$} \{0,1\}^{n(\lambda)};\ x \leftarrow \mathcal{A}(1^\lambda, y) : H_\lambda(x) = y\big] = \mathrm{negl}(\lambda).$$

8.3 Lecture B (Generic bounds and proofs)

Birthday bound. For a random function to n bits, collision after about $2^{n/2}$ queries.

Theorem 8.4 (Birthday estimate (standard)). *Let H be a uniformly random function to $\{0,1\}^n$. After q queries,*

$$\mathrm{Prob}[\textit{a collision}] \approx 1 - \exp\left(-\frac{q(q-1)}{2 \cdot 2^n}\right).$$

8.4 Lecture C (Merkle–Damgård, length extension, HMAC, quantum)

- Iterated hash $H(m) = f(\cdots f(IV, m_1), \dots, m_t)$ with padding.
- Length extension: from $H(m)$ and $|m|$ compute $H(m \| pad(m) \| m')$.
- HMAC structure and why it prevents extension attacks.
- Grover: preimage cost $\approx 2^{n/2}$ quantum queries; implication for security levels.

8.5 Problem set (Week 7)

1. Prove the birthday bound formula (using occupancy / union bound / Poisson approximation).

2. Show length extension explicitly for an iterated compression function model.

3. Explain why "hash-then-MAC" with $H(k\|m)$ is insecure but HMAC is secure under standard assumptions.

9 Optional Capstone: Cross-cutting Comparisons (1–2 lectures)

9.1 Comparing hardness landscapes

- Classical vs quantum asymptotics across families (Shor vs Grover vs no-known-poly-time).

- Parameter selection philosophy: "avoid smoothness" (DLP, factoring), "dimension as security" (lattices), "rate/weight tradeoffs" (codes), "path length / graph size" (isogenies), "degree of regularity" (MQ), "output length" (hash).

9.2 Suggested reading (non-exhaustive)

- Boneh–Shoup, *A Graduate Course in Applied Cryptography.*

- Cohen, *A Course in Computational Algebraic Number Theory.*

- Micciancio–Goldwasser (lattices survey/book).

- MacWilliams–Sloane (coding theory).

- Washington (elliptic curves).

- Cox–Little–O'Shea (Gröbner bases).