

# CO 485/685 Fall 2022:

## Lecture Notes

<b>1</b>	<b>Introduction to Cryptography</b>	<b>3</b>
Lecture 1	(09/07; skipped) . . . . .	3
Lecture 2	Almost-Public Key Cryptosystems (09/09) . . . . .	3
Lecture 3	A Public Key Cryptosystem – RSA (09/12) . . . . .	4
Lecture 4	Security Definitions (09/14) . . . . .	4
Lecture 5	Actual IND-CPA systems (09/16) . . . . .	5
<b>2</b>	<b>Quadratic Residues</b>	<b>7</b>
Lecture 6	Number Theory Background (09/19) . . . . .	7
Lecture 7	Squares Under a Modulus (09/21) . . . . .	7
Lecture 8	Squares cont'd (09/23) . . . . .	8
Lecture 9	Applying to DDH (09/26) . . . . .	9
Lecture 10	Quadratic Characters in the Complex Plane (09/28) . . . . .	10
Lecture 11	Quadratic Reciprocity (09/30) . . . . .	11
<b>3</b>	<b>Primality</b>	<b>14</b>
Lecture 12	Primality Testing (10/03) . . . . .	14
Lecture 13	Strong Primality Testing (10/05) . . . . .	15
Lecture 14	Malleability (10/07) . . . . .	16
Lecture 15	Factorization Algorithms (10/17) . . . . .	17
Lecture 16	Better Sieves (10/19) . . . . .	18
Lecture 17	(10/21) . . . . .	19
Lecture 18	Index Calculus (10/26) . . . . .	19
<b>4</b>	<b>Signatures</b>	<b>21</b>
Lecture 19	Hash Functions (10/28) . . . . .	21
Lecture 20	Signature Schemes (10/31) . . . . .	22
Lecture 21	Hashed RSA (11/02) . . . . .	23
Lecture 22	Zero-Knowledge Proofs (11/04) . . . . .	24
Lecture 23	ZKP Signatures (11/07) . . . . .	25
Lecture 24	CCA2-Secure Signature Schemes (11/09) . . . . .	27
Lecture 25	Proving Fujisaki–Okamoto Security (11/11) . . . . .	27
<b>5</b>	<b>Elliptic Curve Cryptography</b>	<b>30</b>
Lecture 26	Elliptic Curves (11/14) . . . . .	30
Lecture 27	(11/16) . . . . .	31
Lecture 28	Attacks on ECDH (11/18) . . . . .	31
Lecture 29	Pairing-Based Cryptography (11/21) . . . . .	32

Lecture 30	Divisors (11/23)	33
Lecture 31	(11/25)	35

Lecture notes taken, unless otherwise specified, by myself during the Fall 2022 offering of CO 485/685, taught by David Jao.

Chapter/lecture titles are made-up nonsense and do not follow the textbook or any other published resource. Actually, scratch that, this entire document is nonsense because I am literally auditing this course two nested prerequisites behind.

# Chapter 1

## Introduction to Cryptography

Lecture 1 (09/07; skipped)

Lecture 2 Almost-Public Key Cryptosystems (09/09)

- For a symmetric key cryptosystem, require sets of key space  $K$ , message space  $M$ , and ciphertext space  $C$ 
  - Define encryption function  $Enc : K \rightarrow M \rightarrow C$  and decryption  $Dec : K \rightarrow C \rightarrow M$
  - Correctness property: for all  $k$ ,  $Dec(k)$  is a left inverse of  $Enc(k)$
  - Symmetric means that both decryption and encryption use shared secret  $k$ , which we assume is drawn randomly from  $K$
- Public key encryption scheme (Diffie, Hellman, Merkle, c. 1976)
  - Setup similar: message space  $M$  and ciphertext space  $C$  but with two key spaces  $K_1$  of public keys and  $K_2$  of private keys
  - Define  $Enc : K_1 \rightarrow M \rightarrow C$  and  $Dec : K_2 \rightarrow C \rightarrow M$
  - Define  $KeyGen : \mathbb{1}^\ell \rightarrow R \subset K_1 \times K_2$ 
    - \* For some reason, let  $\mathbb{1}^n$  be the unary representation of  $n$ ??
  - Correctness: for all  $(k_1, k_2) \in R$  related,  $Dec(k_2)$  is a left inverse of  $Enc(k_1)$
- Merkle puzzle (1974)
  - Each party creates “puzzle” which is hard to solve but not too hard
  - Alice generates 1,000,000 puzzles and sends them to Bob
  - Bob solves one of the puzzles arbitrarily and sends half of the answer to Alice
  - Alice knows the answer, so Alice knows the second half of the answer, which becomes the shared secret
  - Eve cannot (realistically) solve 500,000 puzzles in time to intercept
- Diffie–Hellman key exchange
  - Consider the multiplicative group  $G = (\mathbb{Z}/p\mathbb{Z})^* = 1, \dots, p-1$  and some arbitrary element  $g \in G$  with sufficiently large order
  - Alice privately picks some  $x \in \mathbb{Z}$ , computes  $g^x$ , and sends it to Bob
  - Bob privately picks some  $y \in \mathbb{Z}$ , computes  $g^y$ , and sends it to Alice
  - Both can now calculate a shared secret  $k = g^{xy} = (g^x)^y = (g^y)^x$
  - Eve would have to solve the Diffie–Hellman problem: given  $p, g, g^x, g^y$ , find  $g^{xy}$  which is known to be hard
- Clifford Cocks privately discovered RSA 1973, DH 1974 for GCHQ (if you believe the intelligence community)

### Lecture 3 A Public Key Cryptosystem – RSA (09/12)

- RSA (Rivest, Shamir, Adleman 1977): first cryptosystem and remains secure
- Theoretically secure, but implementations are ass (cf. “Fuck RSA”)
- MATH 135 review of the algorithm:
  - This “textbook RSA” has practical flaws and is insecure
  - $KeyGen : \mathbb{1}^\ell \rightarrow (pk, sk) \in R$ 
    1. Choose random primes  $p, q \approx 2^\ell$  where  $p$  and  $q$  are odd and distinct
    2. Compute  $n = pq$
    3. Choose  $e \in (\mathbb{Z}/\phi(n)\mathbb{Z})^\times$  where  $\phi(n) = (p-1)(q-1)$
    4. Compute  $d = e^{-1} \bmod \phi(n)$
    5. Disclose public key  $(n, e)$  and keep secret key  $(n, d)$
  - $Enc : K_1 \rightarrow M \rightarrow C : (n, e) \mapsto m \mapsto m^e \bmod n$  where  $M = (\mathbb{Z}/n\mathbb{Z})^\times = x : \mathbb{Z}/n\mathbb{Z} : \gcd(x, n) = 1 = C$ 
    - \* Weird that  $M$  depends on  $n$  (part of the key). In practice, it doesn’t matter because the only messages that divide  $n$  are the primes, which breaks RSA anyways
  - $Dec : K_2 \rightarrow C \rightarrow M : (n, d) \mapsto c \mapsto c^d \bmod n$
- Correctness: Must show that  $(m^e \bmod n)^d \bmod n = m$  *Proof.*  $(m^e \bmod n)^d \bmod n = m^{ed} \bmod n$  (exponentiation under mod). Then, since  $d = e^{-1} \bmod \phi(n)$ , there exists  $k$  such that  $de - 1 = k\phi(n)$ , we have  $m^{\phi(n)k+1} \equiv (m^{\phi(n)})^k m \equiv m \pmod{n}$ . This holds by Euler’s theorem ( $\forall m \in (\mathbb{Z}/n\mathbb{Z})^\times, m^{\phi(n)} \equiv 1 \pmod{n}$ ) or Fermat’s Little Theorem + Chinese Remainder Theorem (MATH 135)
- Security: Trivial that factoring  $n = pq$  breaks RSA by computing  $\phi(n)$ 
  - Conversely, if you know  $\phi(n) = (p-1)(q-1)$  you can take  $q\phi(n) = (n-1)(q-1)$  and solve for  $q$ 
    - \* To avoid this, use the Carmichael exponent  $\lambda(n) = \text{lcm}(p-1, q-1)$  instead of  $\phi(n)$  which works. Of course, this doesn’t work in practice because it’s not actually that much different
  - For any non-trivial case, knowing one pair  $(e, d)$  also allows factoring  $n$
  - Must make an assumption about hardness to prove security:
    - \* Factoring assumption: factoring random integers is hard
    - \* RSA factoring assumption: factoring  $n = pq$  is hard (see, e.g., elliptical curve algorithm which depends on size of smallest prime in the factorization)
      - Of course, quantum computing fucks all of this to hell (see troll PQRSA which uses many small primes to make terabyte-sized moduli)
    - \* RSA assumption: given  $n, e, m^e \bmod n$ , it is hard to find  $m$
  - Can prove RSA assumption  $\implies$  RSA works (cannot prove without assumption without better results from complexity theory)

### Lecture 4 Security Definitions (09/14)

- Security definitions, e.g., OW-CPA, IND-CPA, IND-CCA (Boneh, Shoup)
- How secure is a cryptosystem? Specify:
  - Allowable interactions between adversaries and parties
    - \* Second part of abbreviation
  - Computational limits of adversary
    - \* Not usually specified, usually probabilistic polynomial time

- Goal of the adversary to “break” the cryptosystem
  - \* First part of abbreviation
- OW-CPA: “one-way chosen-plaintext attack”
  - Adversary, given public key  $pk$  and encryption  $c$  of message  $m$  under  $pk$ , wants to determine  $m$
  - Formally, given a random  $pk$  and  $c$  such that  $c = Enc(pk, m)$  for some random  $m$ , it is infeasible for any probabilistic polynomial time algorithm  $\mathcal{A}$  to determine  $m$  with non-negligible probability. That is,  $\Pr[\mathcal{A}(pk, c) = m] = O(\frac{1}{\lambda^c})$  for all  $c > 0$ .
- Easier way to formalize (“Sequences of Games”, Shoup 2004)
  - Two players: challenger  $\mathcal{C}$  and adversary  $\mathcal{A}$
  - Then, OW-CPA is
    1.  $\mathcal{C}$  runs  $KeyGen : \mathbb{1}^\lambda \xrightarrow{\$} (pk, sk)$
    2.  $\mathcal{C}$  chooses  $m \xleftarrow{\$} M$
    3.  $\mathcal{C}$  computes  $c \leftarrow Enc(pk, m)$
    4.  $m' \xleftarrow{\$} \mathcal{A}(pk, c)$ 
      - \* with the win condition that  $m' = m$ , and we say that a cryptosystem is OW-CPA if a probabilistic polynomial time adversary  $\mathcal{A}$  cannot win this game with non-negligible probability
  - IND-CPA (Goldmeier, Micoli 1984): indistinguishability
    1.  $\mathcal{C}$  runs  $(pk, sk) \xleftarrow{\$} KeyGen(\mathbb{1}^\lambda)$
    2.  $(m_0, m_1) \xleftarrow{\$} \mathcal{A}(\mathbb{1}^\lambda, pk)$
    3.  $\mathcal{C}$  picks  $b \xleftarrow{\$} 0, 1$
    4.  $\mathcal{C}$  computes  $c \xleftarrow{\$} Enc(pk, m_b)$
    5.  $b' \xleftarrow{\$} \mathcal{A}(\mathbb{1}^\lambda, pk, c)$ 
      - \* with the win condition  $b = b'$ , and a cryptosystem is IND-CPA if for all prob. poly. time  $\mathcal{A}$ ,  $|\frac{1}{2} - \Pr[\text{win}]| = O(\frac{1}{\lambda^\epsilon})$  for all  $\epsilon > 0$
      - \* Encryption function must be random, otherwise  $\mathcal{A}$  can re-encrypt

## Lecture 5 Actual IND-CPA systems (09/16)

- IND-CPA is the standard security definition for symmetric security
  - Ciphertext contains no information about plaintext (except length)
- Design a slightly different equivalent IND-CPA game:
  1.  $\mathcal{C}$  runs  $(pk, sk) \xleftarrow{\$} KeyGen(\mathbb{1}^\lambda)$
  2.  $(m_0, m_1) \xleftarrow{\$} \mathcal{A}(\mathbb{1}^\lambda, pk)$
  3.  $\mathcal{C}$  picks  $b \xleftarrow{\$} 0, 1$
  4.  $\mathcal{C}$  computes  $c_1 \xleftarrow{\$} Enc(pk, m_b)$  and  $c_2 \xleftarrow{\$} Enc(pk, m_{b-1})$
  5.  $b' \xleftarrow{\$} \mathcal{A}(\mathbb{1}^\lambda, pk, c_1, c_2)$
- Consider textbook RSA:  $\mathcal{A}$  can choose  $m_0 \neq m_1$  and compute  $Enc(pk, m_0)$  and  $Enc(pk, m_1)$  which allows it to win
  - In general, this applies to any scheme with deterministic encryption
- Goldwasser-Micali (“Probabilistic Encryption” 1982)
  1. Pick  $n = pq$  (useful to have  $p \equiv q \equiv 3 \pmod{4}$ )
  2. Pick  $r \in (\mathbb{Z}/n\mathbb{Z})^\times$  such that  $r \not\equiv x^2 \pmod{p}$  and  $r \not\equiv x^2 \pmod{q}$
  3. Define  $pk = (n, r)$  and  $sk = (p, q)$
  4. Select a message bit  $b$  from  $M = 0, 1$

5. Encrypt  $Enc(b) = r^b y^2$  for some  $y \xleftarrow{\$} (\mathbb{Z}/n\mathbb{Z})^\times$ 
  - Then, decrypt by determining ciphertext's squareness mod  $n$ 
    - \* This is easy with the factorization  $n = pq$  by Euler's criterion ( $a$  is square mod prime  $p$  if and only if  $a^{(p-1)/2} \equiv 1 \pmod{p}$ )
    - \* Determining squareness without factorization of  $n$  is hard, apparently
  - Since plaintexts are one bit, OW  $\iff$  IND and this is provable under the circular- $y$  assumption that determining squareness is hard
  - Also one bit messages are literally useless so who cares
- Elgamal (1984) (sometimes IND-CPA)
  - Publickeycryptosystemified Diffie-Hellman
  - 1. Setup is the same as DH, take some element  $g \in G$  of a group
  - 2. Define  $pk = g^x$  and  $sk = x$
  - 3. Encrypt  $Enc(m) = (g^y, g^{xy} \cdot m)$  for  $y \xleftarrow{\$} \mathbb{Z}$ 
    - Then, decrypt  $Dec(c_1, c_2) = \frac{c_2}{c_1^x} = \frac{g^{xy} \cdot m}{(g^y)^x} = m$
    - In general, key sharing schemes can be cryptosystemified like this
    - In an IND-CPA game, given  $(g^y, g^{xy} m_b)$ 
      - \* Divide out  $m_0$  to get either  $g^{xy}$  (if  $m_b = m_0$ ) or garbage
      - \* Real challenge is distinguishing  $g^{xy}$  from garbage
    - Decisional Diffie-Hellman assumption: in the following game,  $|\Pr[\mathcal{A} \text{ wins}] - \frac{1}{2}|$  is negligible in  $\lambda$ 
      1.  $\mathcal{C}$  chooses  $p \xleftarrow{\$} \mathbb{Z}$  prime,  $p \approx 2^\lambda$
      2.  $\mathcal{C}$  chooses  $g \in (\mathbb{Z}/p\mathbb{Z})^\times$
      3.  $\mathcal{C}$  chooses  $x, y \xleftarrow{\$} \mathbb{Z}$  and  $h \xleftarrow{\$} (\mathbb{Z}/p\mathbb{Z})^\times$ , computes  $g_1 = g^x, g_2 = g^y, g_3 = g^{xy}$
      4.  $\mathcal{C}$  chooses  $b \xleftarrow{\$} 0, 1$  and  $g_4 = g_3$  if  $b = 0$  and  $h$  if  $b = 1$
      5.  $b' \leftarrow \mathcal{A}(1^\lambda, p, g, g_1, g_2, g_4)$
    - Can prove: if DDH assumption holds, Elgamal is IND-CPA
  - Layers of assumptions here:
    - DLOG: given  $g$  and  $g^x$ , it is hard to find  $x$
    - CDH: given  $g, g^x$ , and  $g^y$ , it is hard to find  $g^{xy}$  (equivalent to Elgamal being OW-CPA)
    - DDH: given  $g^{xy}$  and garbage, is hard to distinguish the garbage
  - How to piss off mathematicians: solving DLOG in  $\mathbb{Z}/n\mathbb{Z}$  is easy but in  $(\mathbb{Z}/p\mathbb{Z})^\times$  is hard
    - But  $(\mathbb{Z}/p\mathbb{Z})^\times$  is isomorphic to  $\mathbb{Z}/(p-1)\mathbb{Z}$  so DLOG difficulty must not be preserved over isomorphism
    - Specifically, DLOG is as exactly hard as computing the isomorphism (notice that we send  $x \mapsto g^x$ )
  - DDH is actually easy in  $(\mathbb{Z}/p\mathbb{Z})^\times$ , need a subgroup  $G \subset (\mathbb{Z}/p\mathbb{Z})^\times$  with  $|G|$  prime

## Chapter 2

# Quadratic Residues

### Lecture 6 Number Theory Background (09/19)

- Recall: RSA primes are gigantic so it takes time to do operations
  - e.g. picking  $e \in (\mathbb{Z}/\phi(n)\mathbb{Z})^\times$  or finding  $d = e^{-1} \pmod{\phi(n)}$  using EEA which runs in a logarithmic number of steps
  - e.g. running  $Enc(m) = m^e \pmod{n}$  or  $Dec(c) = c^d \pmod{n}$  using square-and-multiply which runs in a logarithmic number of steps
- Hard: picking non-squares in integers modulo  $p$ 
  - Set of primes  $|\{(\mathbb{Z}/p\mathbb{Z})^\times\}^2| = \frac{p-1}{2}$  for odd  $p > 2$
  - This is because  $f(x) = x^2$  is a 2-to-1 function on  $(\mathbb{Z}/p\mathbb{Z})^\times$ 
    - \* To prove, show  $f(a) = f(b) \iff a = \pm b$
    - \* Apply Euclid's Lemma:  $p \mid (x-y)(x+y)$  implies  $p \mid x-y$  or  $p \mid x+y$ , equivalently,  $x = y \pmod{p}$  or  $x = -y \pmod{p}$
    - \* Also another theorem: for  $R$  integral domain, every polynomial of degree  $n$  over  $R$  has at most  $n$  roots

### Lecture 7 Squares Under a Modulus (09/21)

The big problem: Given  $(\mathbb{Z}/n\mathbb{Z})^\times$  and  $x \in (\mathbb{Z}/n\mathbb{Z})^\times$ , when is  $x \equiv \square \pmod{n}$ ?

For example, for  $\mathbb{Z}/15\mathbb{Z}$ , 1 and 4 are squares; for 8: just 1; for 7: 1, 2, and 4; and for 13: 1, 3, 4, 9, 10, and 12.

This breaks down into cases:  $n$  composite,  $n$  prime power,  $n$  prime

#### Theorem

Suppose  $n = \prod p_i^{e_i}$ . Then,  $x \equiv \square \pmod{n}$  if and only if for all  $i$ ,  $x \equiv \square \pmod{p_i^{e_i}}$ .

*Proof.* Suppose  $x = y^2 \pmod{n}$  for a unit  $y$ . Then,  $n \mid (x - y^2)$  and  $p_i^{e_i} \mid (x - y^2)$  by transitivity. That is,  $x \equiv y^2 \pmod{p_i^{e_i}}$ . In the reverse direction, if  $p_i^{e_i} \mid (x - y^2)$  for all  $i$ , then by UPF (with some omitted detail),  $n \mid (x - y^2)$ .  $\square$

The prime power case reduces to the prime case under conditions discovered in the homework problems lol.

**Theorem**

The number of squares in  $(\mathbb{Z}/p\mathbb{Z})^\times$  is  $\frac{p-1}{2}$  for primes  $p \geq 3$ .

*Proof.* This is because  $x = y^2 = (-y)^2$  and the size of the set is  $p - 1$ .

Build a table  $(x, g^x)$  instead of  $(x, x^2)$ :

For  $p = 13$  and  $g = 2$ , we get  $(1, 2, 4, 8, 3, 6, 12 = -1, -2, -4, -8, -3, -6, -12 = 1)$  and the squares are the even-indexed values  $(1, 4, 3, 12, 9, 10, 1)$ .

This works for tables starting with non-squares: in fact, if  $g \neq \square$ , then  $g^3 \neq \square$  (by the contrapositive, if  $g^3 = \square$ , then  $g = \frac{g^3}{g^2} = \frac{\square}{\square} = \square$ ).

This gives us the result that  $g^x = g^y$  when  $x \equiv y \pmod{p-1}$  (note that this is equivalent to Fermat's Little Theorem, the reverse direction requires  $g$  coprime to  $p-1$ ).  $\square$

**Definition (order)**

$\text{ord}(a)$  is the period of  $x \mapsto a^x$  for  $a \in (\mathbb{Z}/p\mathbb{Z})^\times$ .  
Equivalently,  $\text{ord}(a) = \min\{t \in \mathbb{Z} : a^t = 1, t > 0\}$ .

**Lemma**

Given elements  $a$  and  $b$ , numbers  $x$  and  $y$ :

- $a^x = 1$  if and only if  $\text{ord}(a) \mid x$
- $a^x = a^y$  if and only if  $x \equiv y \pmod{\text{ord}(a)}$
- $\text{ord}(a^x) = \frac{\text{ord}(a)}{\gcd(x, \text{ord}(a))}$
- If  $\text{ord}(a)$  and  $\text{ord}(b)$  are coprime, then  $\text{ord}(ab) = \text{ord}(a) \text{ord}(b)$ .

*Proof.* Only prove the last one:

Let  $t = \text{ord}(a)$ ,  $u = \text{ord}(b)$ ,  $v = \text{ord}(ab)$ . Then,  $(ab)^{tu} = a^{tu}b^{tu} = 1^u 1^t = 1$  so we have  $v \mid tu$ . Now, WLOG,  $(ab)^{vu} = 1^u = 1 \implies a^{vu}b^{vu} = a^{vu}1 = a^{vu} = 1$ . This gives  $t \mid vu$  and  $t \mid v$  since  $\gcd(t, u) = 1$ . Likewise,  $u \mid v$  and we can conclude  $tu \mid v$  because  $\gcd(t, u) = 1$ . That is,  $tu = v$ .  $\square$

**Lecture 8 Squares cont'd (09/23)****Definition (primitive element)**

$g \in G$  where  $\{g^n : n \in \mathbb{N}\} = G$ . Also called a generator.

Recall: if there exists primitive  $g \in (\mathbb{Z}/p\mathbb{Z})^\times$ , then for all  $h \in (\mathbb{Z}/p\mathbb{Z})^\times$  where  $h = g^k$ ,  $h \equiv \square \iff k$  even. We can determine squareness using this fact, but finding  $k$  such that  $h = g^k$  is doing a discrete log, which is hard.

Whether or not a primitive element exists is a non-trivial observation:

**Theorem (Gauss' primitive root)**

For all primes  $p$ ,  $(\mathbb{Z}/p\mathbb{Z})^\times$  has a primitive element.

*Proof.* Observe that for all polynomials  $f(x) \neq 0$  over  $\mathbb{Z}/p\mathbb{Z}$ , the number of roots of  $f(x)$  is at most  $\deg f$ . Note that factorization fails in  $\mathbb{Z}/n\mathbb{Z}$  in general: e.g.  $x^2 - 1 = (x-1)(x+1) = (x-3)(x-5) \pmod{8}$  or something weird like  $x = (3x+2)(2x+3) \pmod{6}$ . We have this observation because  $\mathbb{Z}/p\mathbb{Z}$  is an integral domain (and indeed, a field).



Consider  $a \in (\mathbb{Z}/p\mathbb{Z})^\times$ .

Claim  $t = \text{ord}(a) \mid p-1$ . Write  $p-1 = tq + r$ . If  $r = 0$ , done. If  $r > 0$ ,  $\text{ord}(a) = r < t$ , contradiction and indeed  $r = 0$ .

For each divisor  $d$  of  $p-1$ , consider  $S_d = \{x \in (\mathbb{Z}/p\mathbb{Z})^\times : \text{ord}(x) = d\}$ . Then,  $\bigcup_{d \mid p-1} S_d = (\mathbb{Z}/p\mathbb{Z})^\times$  and this is a disjoint union. To prove Gauss' theorem, we just need  $|S_{p-1}| > 0$ .

Proceed in general for arbitrary  $|S_d| > 0$  for all  $d \mid p-1$ .

If  $S_d = \emptyset$ , then  $|S_d| = 0$ . Otherwise, claim that  $|S_d| = \phi(d) = |(\mathbb{Z}/d\mathbb{Z})^\times|$ .

If  $S_d$  is not empty, then  $\exists a \in S_d$  where  $\text{ord}(a) = d$ . Consider  $x^d - 1$ . The roots of this polynomial will include all elements of  $S_d$  (and others). We can write the set of roots as exactly  $\{a^0, \dots, a^{d-1}\}$ . So for all  $b \in S_d$ ,  $b = a^k$  since  $b$  is a root and we need only count those powers with order  $d$ . But that is exactly  $\text{ord}(a^i) = \frac{\text{ord}(a)}{\gcd(i, d)} = \frac{d}{\gcd(i, d)}$ . So we are counting the  $i$  such that  $\gcd(i, d) = 1$ , which is exactly  $\phi(d)$ .

Now,  $p-1 = |(\mathbb{Z}/p\mathbb{Z})^\times| = \left| \bigcup_{d \mid p-1} S_d \right| = \sum |S_d| \leq \sum \phi(d)$  which is equal to  $p-1$  by Möbius inversion. That last inequality being an equality implies that  $|S_d| \neq 0$  for any  $d \mid p-1$ , and in particular  $p-1 \mid p-1$ .

Quick combinatorial proof of this fact: write out all the  $p-1$  fractions over  $p-1$ , then each of  $\phi(d)$  is the number of fractions where the denominator reduces to  $d$ . The sum must be  $p-1$ .  $\square$

## Lecture 9 Applying to DDH (09/26)

Recall the Decisional Diffie-Hellman problem: Given  $g, g^x, g^y, g^z$ , determine if  $z = xy$ . Formally, as a game:

- $\mathcal{C}$  chooses a bit  $b \in \{0, 1\}$  and  $x, y \xleftarrow{\$} \mathbb{Z}$
- $b' \leftarrow \mathcal{A}(g, g^x, g^y, g^z)$  where  $z \leftarrow \begin{cases} xy & b = 0 \\ \xleftarrow{\$} \mathbb{Z} & b = 1 \end{cases}$
- Win condition:  $b = b'$  with non-negligible probability

Notice that if  $g$  is a primitive root, then  $|\{g^x : x \in \mathbb{Z}\}| = p-1$ . But brute force DLOG takes  $\frac{p-1}{2}$  steps on average. Then, Elgamal is IND-CPA  $\iff$  DDH holds.

### Proposition

The Decisional Diffie-Hellman assumption in  $(\mathbb{Z}/p\mathbb{Z})^\times$  with a primitive base  $g$  does not hold.

*Proof.* We tell squares and non-squares apart.

Recall from last lecture's theorem we have that if  $g$  is a primitive root,  $g^x \equiv \square \pmod{p} \iff x \equiv 0 \pmod{2}$ . Then, by Euler's criterion,  $a \equiv \square \pmod{p} \iff a^{(p-1)/2} \equiv 1 \pmod{p}$ . Therefore, it is possible to tell the parity of  $x, y$ , and  $z$  in reasonable time using Euler's criterion (since raising to a power is easy).

If  $xy$  is odd only when  $x$  and  $y$  are odd, so if you know the parity of  $z$  you can distinguish if  $z = xy$  or random with non-negligible advantage.  $\square$

**Proposition** (*Euler's criterion*)

$$a \equiv \square \pmod{p} \iff a^{(p-1)/2} \equiv 1 \pmod{p}$$

*Proof.* Suppose  $a \equiv \square$  iff  $a = g^k$  for even  $k = 2\ell$  iff  $a^{(p-1)/2} = (g^k)^{(p-1)/2} = g^{k(p-1)/2} = (g^{p-1})^\ell = 1^\ell = 1$  by FLT.

Otherwise,  $a \not\equiv \square$  iff  $a = g^k$  for  $k = 2\ell + 1$  iff  $a^{(p-1)/2} = (g^k)^{(p-1)/2} = g^{k(p-1)/2} = g^{(p-1)/2 \cdot 2\ell + 1} = g^{(p-1)/2 \cdot 2\ell} \cdot g^{(p-1)/2} = g^{(p-1)/2} \neq 1$ . But in fact  $g^{(p-1)/2} = \sqrt{g^{p-1}} = \sqrt{1} = -1$  since it is not positive 1.  $\square$

*Corollary.* For  $p > 2$ ,  $-1$  is a square mod  $p$  if and only if  $p \equiv 1 \pmod{4}$ .

*Proof.* For  $-1$  to be a square, we need  $(-1)^{(p-1)/2} \equiv 1 \pmod{p}$ . That is,  $\frac{p-1}{2}$  is even and we have  $p \equiv 1 \pmod{4}$ .  $\square$

This quantity  $g^{(p-1)/2}$  is useful and we give it a name:

**Definition** (*Legendre symbol*)

For  $p > 2$  and  $a \in \mathbb{Z}/p\mathbb{Z}$ , the quadratic character of  $a$ , written  $\left(\frac{a}{p}\right) = a^{(p-1)/2}$ , is 1 if  $a \equiv \square$ , 0 if  $a \equiv 0$ , and -1 if  $a \not\equiv \square$ .

Equivalently, define  $\chi_p : (\mathbb{Z}/p\mathbb{Z})^\times \rightarrow \{\pm 1\} : a \mapsto \left(\frac{a}{p}\right)$  and notice that this is a multiplicative homomorphism that preserves  $\chi_p(ab) = \chi_p(a)\chi_p(b)$ .

**Theorem** (*multiplicativity*)

$$\left(\frac{ab}{p}\right) = \left(\frac{a}{p}\right)\left(\frac{b}{p}\right)$$

*Proof.*  $\left(\frac{ab}{p}\right) = (ab)^{(p-1)/2} = a^{(p-1)/2}b^{(p-1)/2} = \left(\frac{a}{p}\right)\left(\frac{b}{p}\right)$   $\square$

**Lecture 10 Quadratic Characters in the Complex Plane (09/28)**

Recall: we have that for odd primes,  $\left(\frac{-1}{p}\right) = 1 \iff p \equiv 1 \pmod{4}$  which we proved by applying Euler's criterion. We have the similar lemma:

**Lemma**

$$\left(\frac{2}{p}\right) = 1 \iff p \equiv 1, 7 \pmod{8}.$$

*Proof.* This is harder because  $2^{(p-1)/2}$  is not easy to analyze, i.e., the order of 2 is not easy to derive.

What numbers, in general, have finite/known order? Complex roots of unity  $\zeta_n = e^{2\pi i/n}$ .

We can write  $\sqrt{2} = \zeta_8 + \zeta_8^7$ , so  $2^{(p-1)/2} = (\zeta_8 + \zeta_8^7)^{p-1} = \frac{(\zeta_8 + \zeta_8^7)^p}{\zeta_8 + \zeta_8^7}$ . The last transformation is helpful since  $p$  powers behave well mod  $p$ .

Now, notice that  $(x+y)^p \equiv x^p + y^p \pmod{p}$  because all the other terms will have a factor of  $p \mid \binom{p}{i}$ .

Therefore,  $\left(\frac{2}{p}\right) \equiv 2^{(p-1)/2} \equiv \frac{\zeta_8^p + \zeta_8^{7p}}{\zeta_8 + \zeta_8^7} \pmod{p}$ .

There are four cases for  $p \pmod{8}$  because we assume  $p > 2$ :

1.  $\zeta_8^p = \zeta_8^1$  and  $\zeta_8^{7p} = \zeta_8^7$

3.  $\zeta_8^p = \zeta_8^3$  and  $\zeta_8^{7p} = \zeta_8^5$
5.  $\zeta_8^p = \zeta_8^5$  and  $\zeta_8^{7p} = \zeta_8^3$
7.  $\zeta_8^p = \zeta_8^7$  and  $\zeta_8^{7p} = \zeta_8^1$

Clearly, for  $p \equiv 1, 7 \pmod{8}$ , we have  $\frac{\zeta_8^p + \zeta_8^{7p}}{\zeta_8 + \zeta_8^7} = \frac{\zeta_8 + \zeta_8^7}{\zeta_8 + \zeta_8^7} = 1$ . Slightly less intuitively, for  $p \equiv 3, 5 \pmod{8}$ , notice that  $\zeta_8^3 + \zeta_8^5 = -\sqrt{2}$ , so the fractions go to  $-1$ .  $\square$

Note: We can algebraically extend  $\mathbb{Z}/p\mathbb{Z}$  with the necessary complex numbers to make the proof valid (or simply assert that the necessary roots of unity exist).

The pattern sort of extends:

- $\left(\frac{3}{p}\right) = 1$  if  $p \equiv 1, 11 \pmod{12}$  and  $-1$  if  $p \equiv 5, 7 \pmod{12}$ .
- $\left(\frac{5}{p}\right) = 1$  if  $p \equiv \pm 1, \pm 9 \pmod{20}$  and  $-1$  if  $p \equiv \pm 3, \pm 7 \pmod{20}$ .
- $\left(\frac{7}{p}\right) = 1$  if  $p \equiv \pm 1, \pm 3, \pm 9 \pmod{28}$  and  $-1$  if  $p \equiv \pm 5, \pm 11, \pm 13 \pmod{28}$ .

In fact, we have  $\left(\frac{7}{p}\right) = 1$  if  $p \equiv \pm 1, \pm 9, \pm 25 \pmod{28}$ . This flips the question from is 7 a square mod  $p$  to asking if  $p$  is a square mod 28.

#### Lemma

$$\left(\frac{-3}{p}\right) = \begin{cases} 1 & p \equiv 1 \pmod{3} \\ -1 & p \equiv 2 \pmod{3} \end{cases}$$

*Proof.* Consider again  $(-3)^{(p-1)/2} = (\sqrt{-3})^{p-1}$ . We can notice  $\sqrt{-3} = \sqrt{3}i = \zeta_6 + \zeta_3$ .

This gives us  $(\sqrt{-3})^{p-1} = \frac{\zeta_3^p - \zeta_3^{2p}}{\zeta_3 - \zeta_3^2}$  because  $\zeta_6 = -\zeta_3^2$ .

If  $p \equiv 1 \pmod{3}$ , then  $\frac{\zeta_3^p - \zeta_3^{2p}}{\zeta_3 - \zeta_3^2} = \frac{\zeta_3 - \zeta_3^2}{\zeta_3 - \zeta_3^2} = 1$  and if  $p \equiv 2 \pmod{3}$ ,  $\frac{\zeta_3^p - \zeta_3^{2p}}{\zeta_3 - \zeta_3^2} = \frac{\zeta_3^2 - \zeta_3^1}{\zeta_3 - \zeta_3^2} = -1$ .  $\square$

Notice that to get to  $\sqrt{3}$  on the complex plane, we need  $\zeta_{12}$ , which explains why we see mod 12 in the rule. To get  $\sqrt{5}$ , we can either use the fact that  $\cos \frac{2\pi}{5} = \frac{1}{4}(\sqrt{5} - 1)$  or notice that  $(\zeta_5 - \zeta_5^2 - \zeta_5^3 + \zeta_5^4)^2 = (4 - \zeta_5 - \zeta_5^2 - \zeta_5^3 - \zeta_5^4) = 5 - (1 + \zeta_5 + \zeta_5^2 + \zeta_5^3 + \zeta_5^4) = 5$ . We can then execute the same fraction-by-cases technique, getting our result mod 5.

Aside: This is the Gauss sum for  $\sqrt{5} = \sum \left(\frac{i}{5}\right) \zeta_5^i$ .

## Lecture 11 Quadratic Reciprocity (09/30)

Recall the pattern from last lecture, where we noticed that asking if  $q$  is a square mod  $p$  seems to be like asking if  $p$  is a square mod  $4q$ . This is almost true, but in fact

#### Theorem (Quadratic Reciprocity)

$\left(\frac{q}{p}\right) = \left(\frac{p}{q}\right)$  for odd primes  $p \neq \pm q$  where at least one is congruent to 1 mod 4 and at least one is positive.

Equivalently, for all distinct positive odd primes  $p$  and  $q$ ,  $\left(\frac{p}{q}\right)\left(\frac{q}{p}\right) = (-1)^{\frac{p-1}{2} \frac{q-1}{2}}$

The proof follows by Gauss sums and the vague ideas from the last lecture.

This means we can evaluate any Legendre symbol using a modulus as either one of

$$\begin{aligned} \left(\frac{-1}{p}\right) &= (-1)^{\frac{p-1}{2}} = \begin{cases} 1 & p \equiv 1 \pmod{4} \\ -1 & p \equiv 3 \pmod{4} \end{cases} \\ \left(\frac{2}{p}\right) &= (-1)^{\frac{p^2-1}{8}} = \begin{cases} 1 & p \equiv \pm 1 \pmod{8} \\ -1 & p \equiv \pm 3 \pmod{8} \end{cases} \\ \left(\frac{q}{p}\right) &= \left(\frac{p}{q}\right)(-1)^{\frac{p-1}{2} \frac{q-1}{2}} = \begin{cases} \left(\frac{p}{q}\right) & p \equiv \pm 1 \pmod{4} \text{ or } q \equiv \pm 1 \pmod{4} \\ -\left(\frac{p}{q}\right) & p \equiv q \equiv 3 \pmod{4} \end{cases} \end{aligned}$$

which is nicer than using Euler's criterion.

**Example 11.1.** Is 71 a square mod 101?

*Solution.* Write  $\left(\frac{71}{101}\right) = \left(\frac{101}{71}\right) = \left(\frac{30}{71}\right) = \left(\frac{2}{71}\right)\left(\frac{3}{71}\right)\left(\frac{5}{71}\right)$  by quadratic reciprocity and multiplicativity.

Then,  $\left(\frac{2}{71}\right) = 1$  since  $71 \equiv 7 \pmod{8}$ .

Also,  $\left(\frac{3}{71}\right) = -\left(\frac{71}{3}\right) = -\left(\frac{2}{3}\right) = 1$  since  $71 \equiv 3 \pmod{4}$ .

Finally,  $\left(\frac{5}{71}\right) = \left(\frac{71}{5}\right) = \left(\frac{1}{5}\right) = 1$  since 1 is always a square.

This gives  $\left(\frac{71}{101}\right) = 1 \cdot 1 \cdot 1 = 1$  so 71 is a square mod 101.  $\square$

Asymptotically, this is not faster than Euler's criterion because we require factoring. However, it is prettier.

To deal with a random large number, we must consider what to do after factoring out all the 2s (since we can deal with those quickly).

**Definition** (*Jacobi symbol*)

For all  $m, n \in \mathbb{N}_{>0}$  with  $n$  odd,  $\left(\frac{m}{n}\right) = \prod_{i=1}^k \left(\frac{m}{p_i}\right)$  where  $\prod_{i=1}^k p_i = n$  is the prime factorization of  $n$

**Theorem** (*Jacobi*)

For all positive and odd  $m$  and  $n$ ,

$$\begin{aligned} \left(\frac{-1}{n}\right) &= (-1)^{\frac{n-1}{2}} = \begin{cases} 1 & n \equiv 1 \pmod{4} \\ -1 & n \equiv 3 \pmod{4} \end{cases} \\ \left(\frac{2}{n}\right) &= (-1)^{\frac{n^2-1}{8}} = \begin{cases} 1 & n \equiv \pm 1 \pmod{8} \\ -1 & n \equiv \pm 3 \pmod{8} \end{cases} \\ \left(\frac{m}{n}\right) &= \left(\frac{n}{m}\right)(-1)^{\frac{n-1}{2} \frac{m-1}{2}} = \begin{cases} \left(\frac{n}{m}\right) & n \equiv \pm 1 \pmod{4} \text{ or } m \equiv \pm 1 \pmod{4} \\ 0 & \gcd(m, n) \neq 1 \\ -\left(\frac{n}{m}\right) & n \equiv m \equiv 3 \pmod{4} \end{cases} \end{aligned}$$

Note: For Legendre symbols,  $\left(\frac{a}{p}\right) = 1 \iff a \equiv \square \pmod{p}$ . However, for Jacobi symbols, we only have the one-way implication  $\left(\frac{m}{n}\right) = -1 \implies m \not\equiv \square \pmod{n}$ .

Return now to the application to cryptography, specifically to Goldwasser-Micali.

**Goldwasser–Micali cryptosystem**

**Key Generation:** Choose random primes  $p, q$ . Set  $n = pq$ .

Choose  $x \in (\mathbb{Z}/n\mathbb{Z})^\times$  such that  $\left(\frac{x}{p}\right) = \left(\frac{x}{q}\right) = -1$  (then  $\left(\frac{x}{n}\right) = 1$ ). Publish  $x$ .

**Encrypt:**  $m \in \{0, 1\}$

Choose some  $r \xleftarrow{\$} (\mathbb{Z}/n\mathbb{Z})^\times$ . Then,  $Enc(m) = x^m r^2 = c$ .

**Decrypt:** Determine whether  $c$  is a “fake” square using the factorization.

The underlying assumption is that it is not easy to distinguish actual squares mod  $n$  and “fake” squares mod  $n$ .

## Chapter 3

# Primality

### Lecture 12 Primality Testing (10/03)

Given  $n \in \mathbb{Z}$ , how can we tell if  $n$  is prime?

**Lemma** (*Fermat test*)

Recall FLT: for a prime  $p$ ,  $a \in (\mathbb{Z}/p\mathbb{Z})^\times \implies a^{p-1} = 1$ . Therefore, if  $a \in (\mathbb{Z}/n\mathbb{Z})^\times$  and  $a^{n-1} \neq 1$ , then  $n$  is not prime.

**Definition** (*Fermat witness*)

Let  $n \in \mathbb{N}$ ,  $\alpha \in (\mathbb{Z}/n\mathbb{Z})^\times$  where  $\alpha^{n-1} \neq 1$ .

When  $n$  is prime, no Fermat witness can exist. When  $n$  is not prime, only some elements are Fermat witnesses. The other elements are *Fermat liars*. How many liars are in  $(\mathbb{Z}/n\mathbb{Z})^\times$ ?

**Theorem**

For  $n > 2$ , if there exists one Fermat witness in  $(\mathbb{Z}/n\mathbb{Z})^\times$ , then there exist at least  $\frac{\phi(n)}{2}$  Fermat witnesses.

*Proof.* Consider the set  $H = \{\alpha \in (\mathbb{Z}/n\mathbb{Z})^\times : \alpha^{n-1} = 1\}$ .

$H$  is a subgroup:  $1 \in H$ ,  $ab \in H$ ,  $a^{-1} \in H$  (trivial by exponentiation properties).

So by Lagrange's theorem,  $|H| \mid |(\mathbb{Z}/n\mathbb{Z})^\times|$ .

Either (1)  $|H| = \phi(n)$ , so there are no witnesses, or (2)  $|H| < \phi(n)$ , so  $|H| \leq \frac{\phi(n)}{2}$ .  $\square$

**Definition** (*Carmichael number*)

$n \in \mathbb{N}$ ,  $n > 2$  such that  $n$  is composite and  $n$  has no Fermat witnesses.

Examples:  $n = 561 = 3 \times 11 \times 17$ . By FLT, we have  $\alpha^{n-1} = \alpha^{560}$  is 1 mod 3, 1 mod 11, and 1 mod 17.

Recall that for  $n$  prime:  $a^{\frac{n-1}{2}} \equiv \left(\frac{a}{n}\right) \pmod{n}$  when  $n > 2$ , odd, and  $a \in (\mathbb{Z}/n\mathbb{Z})^\times$ . This gives us the following test:

**Lemma** (*Solovay–Strassen test*)

If  $a^{\frac{n-1}{2}} \not\equiv \left(\frac{a}{n}\right) \pmod{n}$ , then  $n$  is not prime.

We can calculate  $a^{\frac{n-1}{2}}$  by repeated squaring and  $\left(\frac{a}{n}\right)$  by Jacobi reciprocity and factoring out 2's. We can now define witnesses as in the Fermat test.

**Definition** (*Euler (Solovay–Strassen) witness*)

An element  $\alpha \in (\mathbb{Z}/n\mathbb{Z})^\times$  where  $\left(\frac{\alpha}{n}\right) \not\equiv \alpha^{\frac{n-1}{2}} \pmod{n}$ . If an element is not an Euler witness, it is an Euler liar.

Notice that all Euler witnesses must also be Fermat witnesses, meaning that hopefully we have a more refined test here.

**Theorem**

If  $n > 2$  is composite and odd, then there exists at least one Euler witness.

*Proof.* Suppose  $n$  is composite and  $n = p \times k$ .

If  $p \nmid k$ , then solve  $\alpha \equiv \beta \pmod{p}$  and  $\alpha \equiv 1 \pmod{k}$  where  $\beta$  is a quadratic non-residue mod  $p$ . Now, calculate

$$\left(\frac{\alpha}{n}\right) = \left(\frac{\alpha}{p}\right)\left(\frac{\alpha}{k}\right) = \left(\frac{\beta}{p}\right)\left(\frac{1}{k}\right) = (-1)(1) = -1$$

Suppose  $\alpha^{\frac{n-1}{2}}$  is  $-1$ . Then,  $\alpha^{\frac{n-1}{2}} \equiv -1 \pmod{n}$  and that means  $\alpha^{\frac{n-1}{2}} \equiv -1 \pmod{k}$ . But we know  $\alpha \equiv 1 \pmod{k}$ , so this is a contradiction.

Otherwise,  $p \mid k$ . Let  $\alpha = 1 + k$ . Calculate

$$\left(\frac{\alpha}{n}\right) = \left(\frac{1+k}{n}\right) = \left(\frac{1+k}{p}\right)\left(\frac{1+k}{k}\right) = \left(\frac{1}{p}\right)\left(\frac{1}{k}\right) = (1)(1) = 1$$

Suppose  $\alpha^{\frac{n-1}{2}} = 1$ . This implies that  $\text{ord}(\alpha) \mid \frac{n-1}{2}$ . Calculate  $\alpha^p = (1+k)^p = 1^p + \underbrace{pk^1 + \dots + \binom{p}{p}k^p}_{0 \pmod{n}} = 1$  which implies  $\text{ord}(\alpha) = p$ . But  $p \mid n \implies p \nmid n-1 \implies p \nmid \frac{n-1}{2}$ .

Therefore,  $\alpha$  is an Euler witness. □

This theorem combined with the at-least- $\frac{\phi(n)}{2}$  theorem means that we have for every odd, composite  $n > 2$  there are  $\frac{\phi(n)}{2}$  Euler witnesses.

**Lecture 13 Strong Primality Testing (10/05)**

Recall: for the Fermat test, evaluate  $a^{n-1}$  a bunch of times. If it is equal to 1, prime or liar; otherwise, composite. For the Solovay–Strassen test, evaluate  $a^{\frac{n-1}{2}} = \left(\frac{a}{n}\right)$ . If yes, prime or Euler liar; otherwise, composite. Also, there are an infinite number of Carmichael numbers that screw with this but otherwise you have around a 50% chance of getting a witness.

We can refine this further beyond considering  $n-1$  and  $\frac{n-1}{2}$ .

Write  $n-1 = 2^t \cdot s$  so that  $s$  is odd. Then,  $a^{n-1}$  is  $a^s$  squared  $t$  times. So instead of asking if  $a^{2^t s} = 1$ , consider if  $a^{2^{t-1}s}$  is an “expected” square root of 1, i.e.,  $\pm 1$ . If it is not,

it is composite. If it is and it is  $-1$ , we have a prime or liar. If it is and it is  $1$ , keep going back. If we reach  $a^s = 1$ , we get no information.

**Lemma** (*Miller–Rabin test*)

Let  $x \leftarrow a^s$ . Do:

- If  $x = 1$ , stop. Probably prime.
- If  $x = -1$ , stop. Probably prime.
- Otherwise,  $x \leftarrow x^2$

while  $x \neq a^{2^t s}$ . If we reach the end, it is composite.

**Definition** (*Miller–Rabin (strong) liar*)

$a \in (\mathbb{Z}/n\mathbb{Z})^\times$  if either  $a^s = 1$  or  $a^{2^k s} = -1$  for  $0 \leq k < t$ .

We call this a “strong liar” because every strong liar is an Euler liar, and every Euler liar is a Fermat liar.

**Theorem**

Suppose  $n$  has at least two distinct prime factors. Then, the number of Miller–Rabin liars is at most  $\frac{\phi(n)}{4}$  and in general, if  $n$  has  $\ell$  distinct prime factors, there are at most  $\frac{\phi(n)}{2^\ell}$  Miller–Rabin liars.

We can make these primality tests deterministic by iterating  $a = 1, \dots, n$ . We do not need to go to  $a = n$  and instead we can establish an upper bound on the smallest witness. The bound (by Bach) is  $O(\log^2 n)$ , specifically,  $2 \log^2 n$ . But this requires the Generalized Riemann Hypothesis which everyone believes anyways, so we just check  $a = 1, \dots, 2 \log^2 n$ .

To analyze complexity, notice that we have  $\log n$  multiplications at each step, i.e.,  $\log^{1+\epsilon} n$  bit operations using fast multiplication. So the complexity is  $O(\log^{2+(1+\epsilon)+1} n)$ .

Further reading:

- AKS (Agrawal–Kayal–Saxena; 2004) primality test in  $O(\log^6 n)$  which does not rely on GRH and was an undegrad project(!!)
- ECPP (elliptic curve prime proving) notable for not having liars, also does not require GRH and runs non-deterministically (Monte Carlo) in  $O(\log^5 n)$
- Cyclotomic primality test in  $O((\log n)^{\log \log n})$ , best until AKS proved that primality is in P.

Since there are  $\frac{n}{\log n} + O(\sqrt{n})$  primes less than  $n$ , we can pick random numbers of size  $e^\ell$  to get an approximate  $\frac{1}{\ell}$  probability of a prime.

## Lecture 14 Malleability (10/07)

Recall the Goldwasser–Micali cryptosystem. It satisfies IND-CPA provided that the quadratic reciprocity problem is hard. That is, determining whether an  $x = pq$  with  $\left(\frac{x}{n}\right) = 1$  is actually a square or not (i.e.  $\left(\frac{x}{p}\right) = 1$ ).

However, an adversary can still alter the message without needing to decrypt. This also applies, for example, to XOR one-time pads (since if  $c = k \oplus m$  and we intercept  $c \mapsto c \oplus n$ , recipient will get  $m' = m \oplus n$ ). Using MACs can get around this problem (e.g. AES with GCM or Chacha20 with Poly1305).



**Definition** (*non-malleability*)

Given the game NM-CPA:

1.  $\mathcal{C}$  generates  $(pk, sk)$
2.  $(m_0, m_1, m'_0, m'_1) \xleftarrow{\$} \mathcal{A}(\lambda, pk)$  where  $m'_0 \neq m'_1$
3.  $\mathcal{C}$  chooses  $b \xleftarrow{\$} \{0, 1\}$
4.  $\mathcal{C}$  computes  $c = E(m_b)$
5.  $c' \leftarrow \mathcal{A}(\lambda, pk, c)$

with win condition  $D(c') = m'_b$  with non-negligible probability above 50%.

Instead of CPA games, consider CCA2 (chosen-ciphertext attack 2) games. Here, the adversary has a decryption oracle that takes anything except  $c$ . In CCA1, the oracle can only be accessed prior to receiving  $c$ .

**Theorem**

IND-CCA2 is equivalent to NM-CCA2

Note that IND-CPA is not equivalent to NM-CPA, which is instead equal to IND-PCA (parallel ciphertext attack, where all oracle queries must occur at once).

**Lecture 15 Factorization Algorithms (10/17)**

Naive approach: trial division by  $1, \dots, \sqrt{n}$  which is  $O(\sqrt{n}) = O(\exp(\frac{1}{2} \log n))$ . Note that we call this “exponential” because we measure with respect to the size of the input, i.e.,  $\lg n \approx \log n$ .

**Proposition**

If  $x, y \in (\mathbb{Z}/n\mathbb{Z})^\times$  satisfy  $x^2 \equiv y^2 \pmod{n}$  and  $x \not\equiv \pm y \pmod{n}$ , then  $\gcd(n, x - y)$  is a non-trivial factor of  $n$ .

*Proof.* Since  $x^2 - y^2 \equiv 0$ , we have  $(x - y)(x + y) \equiv 0$ . But we know that  $x - y \not\equiv 0$  and  $x + y \not\equiv 0$  so there must be some weird hidden factor.

If  $\gcd(n, x - y) = n$ , then  $n \mid (x - y) \implies x \equiv y \pmod{n}$  and if  $\gcd(n, x - y) = 1$ , then  $n \mid (x - y)(x + y)$  which implies  $n \mid (x + y)$  by Gauss’ Lemma which gives the same contradiction. Therefore, since the GCD must divide  $n$ , it is non-trivial.  $\square$

Using this, we can find non-trivial factors of  $n$  by finding  $x$  and  $y$  and then applying the EEA. How to find  $x$  and  $y$ ?

**Random squares (Dixon)****Definition** (*B-smoothness*)

$n \in \mathbb{N}$  where the largest prime factor is less than  $B$

Chose  $x_i \xleftarrow{\$} (\mathbb{Z}/n\mathbb{Z})^\times$ . For each  $x_i$ , compute  $x_i^2 \pmod{n}$  and keep the  $B$ -smooth squares. We can tell if a number is  $B$ -smooth by trial division (since  $B$  is small).

We need at least  $t + 1$  squares that are  $B$ -smooth.<sup>1</sup>

<sup>1</sup>Where  $t = \pi(B)$  is the prime-counting function.

This gives us squares  $x_1^2 \bmod n = p_1^{e_{1,1}} \cdots p_t^{e_{t,1}}$  up to  $x_{t+1}^2 \bmod n = p_1^{e_{1,t+1}} \cdots p_t^{e_{t,t+1}}$ .

Take the subset product  $\prod x_i^2 \bmod n = p_1^{\sum e_{1,i}} \cdots p_t^{\sum e_{t,i}}$ .

We can define  $b_i = \begin{cases} 0 & i \in S \\ 1 & i \notin S \end{cases}$  so that  $\sum_{i \in S} e_{j,i} = \sum_{i=1}^{t+1} e_{j,i} b_i = 0 \bmod 2$  to find squares.

Solve this homogeneous linear system over  $\mathbb{Z}/2\mathbb{Z}$  (where the  $b_i$  are variables). We know there exists a non-trivial solution because there are more variables (at least  $t+1$ ) than equations (exactly  $t$ ).

That gives a square subset product  $x^2 = \prod_{i=1}^{t+1} (x_i^2)^{b_i} \bmod n = \prod_{j=1}^t p_j^{\sum_{i=1}^{t+1} e_{j,i} b_i} \bmod n = y^2$ .

The LHS and RHS are unrelated except for the fact that they are equal mod  $n$ . In fact, with about 50% probability,  $x \not\equiv \pm y \bmod n$ . The probability can be improved by increasing  $t+1$  to like  $t+10$ . Since  $t \approx B$  is large, this is negligible.

Picking  $B$ : large  $B$  makes it more likely to find  $B$ -smooth squares, however, the amount of work  $t+1$  is proportional to  $B$ .

We want to pick  $B$  such that the probability of squares being  $B$ -smooth is  $\frac{1}{B}$ . This depends on  $n$ .

From analytic number theory, the probability that a random  $y \xleftarrow{\$} \mathbb{Z}/n\mathbb{Z}$  is  $L(\alpha, c)$ -smooth is  $L(1 - \alpha, \frac{1-\alpha}{c})$ .<sup>2</sup> So we set a bound on  $B$  of  $L_n(\frac{1}{2}, \frac{\sqrt{2}}{2})$ . Since  $(\log n)^k \ll B \ll \sqrt{n}$ , we call this subexponential.

## Lecture 16 Better Sieves (10/19)

What is the probability that a particular  $x^2 \bmod n$  is  $B$ -smooth? Vanishingly small for large  $n$  (in the hundreds of digits) and small-ish  $B$  (around  $10^9$ ). However, we can prove that the runtime for random squares is  $L_n(\frac{1}{2}, 2\sqrt{2})$  using results from analytic number theory, i.e., probabilistic subexponential time.

How can we improve? Pick  $x$  such that  $x^2 \bmod n$  is small (and more likely to be  $B$ -smooth). Naively: small numbers stay small (but are useless). Instead, pick  $x \approx \sqrt{n}$  so that  $x^2 \bmod n = x^2 - n$ .

Then, if  $x = \sqrt{n} + k$ ,  $x^2 = (\sqrt{n} + k)^2 - n = 2k\sqrt{n} + k^2 = O(k\sqrt{n})$ , i.e., around half the size of  $n$  and much smaller than  $n$ .

This is the quadratic sieve. We can bound  $B < L_n(\frac{1}{2}, 1)$  and prove runtime  $L_n(\frac{1}{2}, \sqrt{2})$ .

Suppose we write  $\sqrt{n} = a_0 + \frac{1}{a_1 + \frac{1}{a_2 + \frac{1}{\ddots}}} = [a_0, a_1, \dots]$  as a continued fraction.

Define  $\frac{P_i}{Q_i} = [a_0, \dots, a_i]$ . These fractions rapidly approach  $\sqrt{n}$  (and are in fact the best rational approximations). That is,  $P_i^2 - nQ_i^2$  rapidly approaches 0. We can prove that  $0 < P_i - nQ_i^2 < 2\sqrt{n} + 1$ . Then, we can take  $P_i^2 \bmod n$  and sieve guaranteed that the squares are  $O(2\sqrt{n})$ .

<sup>2</sup>Where  $L_n(\alpha, c) = O(\exp(c(\log n)^\alpha (\log \log n)^{1-\alpha}))$ . Notably,  $L_n(1, 1) = n$  and  $L_n(1, c) = n^c$ . Then,  $\sqrt{n} = L_n(1, \frac{1}{2})$ . Also,  $L_n(0, c) = (\log n)^c$ . That is, we interpolate between  $L_n(0, c)$  polynomial time and  $L_n(1, c)$  exponential time

Comparing the continued fraction sieve and quadratic sieve,  $O(2\sqrt{n})$  appears better than  $O(k\sqrt{n})$ . However, if the quadratic sieve considers consecutive numbers to square, we can do a sieve of Eratosthenes-like search to find good  $B$ -smooth candidates. This is faster.

One more improvement step: number field sieve.

Choose  $d \approx 6 \in \mathbb{Z}$  and  $m \approx n^{1/d}$ . Write  $n$  in base  $m$ :  $n = a_0 + a_1m + \dots + a_5m^5$  and consider the polynomials  $f(x) = a_0 + a_1x + \dots + a_5x^5$  and  $g(x) = x - m$ .

We know that  $f(m) = n \equiv 0 \pmod{n}$  and  $g(m) = 0$ . That is,  $m$  is a root of both  $f$  and  $g \pmod{n}$ . The coefficients are also all around  $m = O(n^{1/d})$  in size.

Consider  $\alpha$  a complex root of  $f$  (but consider it as part of a number field  $\alpha \in \mathbb{Z}[\alpha]$ ). We pick  $a_i, b_i \in \mathbb{Z}$  such that  $a_i + b_i\alpha = \prod \beta_i$  is smooth in  $\mathbb{Z}[\alpha]$  and  $a_i + b_im = \prod q_i$  is smooth in  $\mathbb{Z}$ .

Pick a subset  $S$  such that  $\prod_{i \in S} (a_i + b_i\alpha) = \square$  in  $\mathbb{Z}[\alpha]$  and  $\prod_{i \in S} (a_i + b_im) = \square \pmod{n}$  in  $\mathbb{Z}$ . We can expand the first sum, then replace  $\alpha$  with  $m \pmod{n}$  to get congruent squares for a sieve. In fact,  $\alpha \mapsto m \pmod{n}$  is a ring homomorphism from  $\mathbb{Z}[\alpha] \rightarrow \mathbb{Z}/n\mathbb{Z}$ .

Since the numbers are smaller, we have complexity  $L_n(\frac{1}{3}, \sqrt[3]{\frac{64}{9}})$ .

## Lecture 17 (10/21)

## Lecture 18 Index Calculus (10/26)

There is a connection between runtimes of factoring algorithms and DLOG algorithms:

Factoring $n$	DLOG in $\mathbb{Z}_p^*$ or $\mathbb{F}_q^*$ where $q = p^k$
Trial ( $O(n)$ )	Trial ( $O(p)$ )
$O(\sqrt{n})$	Pollard's rho ( $O(\sqrt{p})$ )
Random squares ( $L_p(\frac{1}{2}, \sqrt{2})$ )	Index calculus ( $L_p(\frac{1}{2}, \sqrt{2})$ )
NFS ( $L_n(\frac{1}{3}, \sqrt[3]{\frac{64}{9}})$ )	NFS for DLOG ( $L_p(\frac{1}{3}, \sqrt[3]{\frac{64}{9}})$ )
Special NFS ( $L_n(\frac{1}{3}, \sqrt[3]{\frac{32}{9}})$ )	Tower NFS ( $L_q(\frac{1}{3}, \sqrt[3]{\frac{32}{9}})$ if $k < 50$ )
???	$L_q(\varepsilon, c)$ for $p < 10$

### Theorem (Shoup)

For a generic group, classical probabilistic DLOG algorithms require  $\Omega(\sqrt{p})$  group operations.

What we mean by generic here is that the group “interface” is exposed (multiplication, inversion, equality) but we don’t know anything about the elements/structure.

### Index calculus

Consider  $\mathbb{Z}_p^* = \{1, 2, \dots, p-1\}$ ,  $g, h = g^\alpha$ . We want to find  $\alpha$ , the “index”. We construct “random index calculus” from the random squares algorithm. Pick random  $x_i$  and

calculate:

$$\begin{aligned} g^{x_1} \bmod p &= p_1^{e_{1,1}} \dots p_t^{e_{t,1}} \\ &\vdots \\ g^{x_{t+1}} \bmod p &= p_1^{e_{1,t+1}} \dots p_t^{e_{t,t+1}} \end{aligned}$$

where we keep  $B$ -smooth  $g^{x_i} \bmod p \approx O(p)$  until we get more equations than primes  $p_i$ .

If we take log base  $g$  on both sides:  $x_1 \equiv \sum e_{i,1} \log_g p_i \pmod{p-1}$ .<sup>3</sup> Since we know the  $x_i$  and  $e_{i,j}$ , we can solve the system of linear equations for the discrete logs  $\log_g p_i$  (since there are at least  $t+1$  equations and  $t$  variables).

Now, take random  $y$  find an  $h^y = p_1^{f_1} \dots p_t^{f_t}$  that is  $B$ -smooth. Taking logs as above,  $y\alpha = \sum f_i \log_g p_i$  and we can solve for  $\alpha$ .

Since this is basically the same process as random squares, it is no surprise it has similar time complexity  $L_p(\frac{1}{2}, \sqrt{2})$ . Practically, it's slightly harder than factoring.

---

<sup>3</sup> $\log_g p_i$  always exists. If  $g$  is not a generator since *some* generator  $h$  exists and we have  $\frac{\log_h p_i \bmod p-1}{\log_h g \bmod p-1}$ .

# Chapter 4

## Signatures

### Lecture 19 Hash Functions (10/28)

To establish something that is NM-CCA2 secure, we need to somehow “sign” the ciphertext to distinguish “authenticated” ciphertexts. We can do this with MACs (e.g., AES-GCM or ChaCha20-Poly1305) but we will do something different.

Consider a hybrid encryption scheme: use public-key encryption to send a symmetric key that encrypts the message. This is CO 487 content.

#### Hash functions

Most common hash functions are the SHA family: SHA0 (broken 2005), SHA1 (broken 2017), SHA2 (actually used), SHA3 (not really used, made in anticipation of SHA2 breaking). Again, CO 487 content beyond the scope of this course.

#### **Definition** (*hash function*)

Function  $H : S \rightarrow T$  (typically,  $S = \{0, 1\}^*$  and  $T = \{0, 1\}^\lambda$ )

Ideally, a hash function is a random oracle, i.e.,  $H \stackrel{\$}{\leftarrow} \{f : (f : S \rightarrow T)\}$ . This is useful, e.g., for making hashed RSA signatures existentially unforgeable under chosen message attack.

There is no way to easily construct a random oracle because (1) we can’t construct the set of all functions and (2) we run into measure theory issues with defining a probability distribution on that set. Instead we construct with desired properties:

1. Preimage resistant: Given  $t \in T$ , it is infeasible to find  $s \in H^{-1}(t)$ .
2. Second preimage resistant: Given  $s \in S$ , it is infeasible to find  $s' \in S$  such that  $s \neq s'$  and  $H(s) = H(s')$ .
3. Collision resistant: It is infeasible to find  $s \neq s'$  such that  $H(s) = H(s')$ .

**Example 19.1.** Are all preimage resistant functions second preimage resistant?

*Solution.* Consider  $f(x) = x^2 \bmod n$ . To find a preimage, take  $x = \sqrt{y}$  (hard). To find a second preimage, take  $x' = -x \neq x$  so  $(-x)^2 = x^2$  (easy).  $\square$

To be formal, use games. For example, with collision resistance: Suppose we have a family of hash functions  $HashGen : \mathbb{1}^\lambda \mapsto H_\lambda$ . Play the game:

1. Pick a hash function  $H_\lambda \xleftarrow{\$} HashGen(\mathbb{1}^\lambda)$
2.  $(s, s') \leftarrow \mathcal{A}(\mathbb{1}^\lambda, H_\lambda)$

with win condition  $H_\lambda(s) = H_\lambda(s')$  and  $s \neq s'$ . We define  $\{H_\lambda : \lambda \in \mathbb{N}\}$  to be collision-resistant if no probabilistic polynomial time adversary  $\mathcal{A}$  can win this game with non-negligible probability in  $\lambda$ .

We can construct collision-resistant hash functions from claw-free permutations by Damgård.

**Definition** (*claw-free permutation*)

Given a set  $X$ , the pair of permutations  $(f, g)$  is claw-free if it is infeasible to find  $x_1, x_2 \in X$  such that  $f(x_1) = g(x_2)$ .

The wrong way: Given claw-free permutations  $f : X \rightarrow X$  and  $g : X \rightarrow X$ , we define  $H : \{0, 1\}^* \rightarrow X$  with  $H(\varepsilon) = x_\varepsilon$ . Inductively,  $H(b_1 b_2 \dots b_n) = h(H(b_1 \dots b_{n-1}))$  where  $h = f$  if  $b_n = 0$  and  $g$  if  $b_n = 1$ . Claim this is collision-resistant because if there is a collision  $H(m) = H(m')$  and  $m \neq m'$ , we have a claw at some point, which is a contradiction. Unfortunately, we could run into a loop back to  $x_\varepsilon$ .

Instead, pick  $x_0 \in X$  and define  $x_\varepsilon = g(f(x_0))$  and define  $H(b_0 \dots b_n) = h(h(H(b_0 \dots b_{n-1})))$  as above. Then, we cannot arrive at  $x_\varepsilon$  because generating pairs of  $f(f(\dots))$  and  $h(h(\dots))$  cannot create  $g(f(\dots))$ .

## Lecture 20 Signature Schemes (10/31)

Consider some RSA modulus  $n = pq$ ,  $p > 2$ ,  $q > 2$ ,  $p \neq q$  where  $p \equiv 3 \pmod{4}$  and  $q \equiv 3 \pmod{4}$  (Blum integers, notable for use in the Blum–Blum–Shub generator).

Let  $y_p$  and  $-y_p$  be square roots of  $y \pmod{p}$  (and for  $q$ ). Notice that  $\left(\frac{-1}{p}\right) = -1$  and  $\left(\frac{-1}{q}\right) = -1$ . Then, exactly one of  $\{y_p, -y_p\}$  is a square mod  $p$  (and for  $q$ ).

Finally, combining gives exactly one of the square roots of  $y \pmod{n}$  is a square mod  $n$ . This means that  $f(x) = x^2$  is a permutation on  $((\mathbb{Z}/n\mathbb{Z})^\times)^2$ .

Choose  $a \in (\mathbb{Z}/n\mathbb{Z})^\times$  such that  $\left(\frac{a}{n}\right) = -1$ . Then, define  $g(y) = a^2 y^2$  which is also a permutation.

Note: suppose  $f(x) = g(y)$ . Then,  $x^2 = (ay)^2$  and  $x \neq \pm ay$  because if  $x = \pm ay$  then  $\left(\frac{x}{n}\right) = \left(\frac{\pm 1}{n}\right)\left(\frac{a}{n}\right)\left(\frac{y}{n}\right)$  but this is  $1 = (1)(-1)(1)$ , contradiction. From this, we can factor  $n$  (by Fermat).

Overall: claw-free permutations  $\rightarrow$  collision-resistant hash functions  $\rightarrow$  {secure digital signatures, CCA2-secure encryption, etc.}

How do we generate secure digital signatures?

Suppose we have RSA  $pk = (n, e)$  and  $sk = (n, d)$ . Then, define signing and verification as  $Sign : (\mathbb{Z}/n\mathbb{Z})^\times \rightarrow (\mathbb{Z}/n\mathbb{Z})^\times, m \mapsto \sigma := m^d \pmod{n}$  and  $Verify : (m, \sigma) \mapsto \sigma^e \pmod{n} \stackrel{?}{=} m$ .

**Definition** (*signature schemes*)

A signature (scheme) is a tuple  $(KeyGen, Sign, Verify)$  where

- $KeyGen : \mathbb{1}^\lambda \mapsto (pk, sk)$
- $Sign : (sk, m) \mapsto \sigma$
- $Verify : (pk, m, \sigma) \mapsto \{0, 1\}$

and we have that if  $(pk, sk) \xleftarrow{\$} KG(\mathbb{1}^\lambda)$  and  $\sigma \xleftarrow{\$} S(sk, m)$ , then  $V(pk, m, \sigma) = 1$ .

Under Textbook RSA, it is trivial to forge junk (but valid) signatures, i.e., given random signature  $\sigma$ , it signs some calculable message.

Example security definition game: EUF-CMA

Existential unforgeability (EUF): adversary produces a valid signature

Chosen-message attack (CMA): adversary can always use a signing oracle

1.  $(pk, sk) \xleftarrow{\$} KeyGen(\mathbb{1}^\lambda)$
2. **for**  $i = 1 \dots q$  **do**:
  - $m_i \xleftarrow{\$} \mathcal{A}(\mathbb{1}^\lambda, pk, (m_1, \sigma_1), \dots, (m_{i-1}, \sigma_{i-1}))$
  - $\sigma_i \xleftarrow{\$} Sign(sk, m_i)$
- end**
3.  $(m, \sigma) \xleftarrow{\$} \mathcal{A}(\mathbb{1}^\lambda, pk, (m_1, \sigma_1), \dots, (m_q, \sigma_q))$

with win condition  $Verify(pk, m, \sigma) = 1$  and for all  $i$ ,  $m \neq m_i$ .

**Definition** (*EUF-CMA*)

A signature scheme is EUF-CMA if there does not exist a probabilistic polynomial time adversary  $\mathcal{A}$  which wins the EUF-CMA game with non-negligible probability.

**Hashed RSA**  $KeyGen : \mathbb{1}^\lambda \mapsto ((n, e), (n, d))$

$Sign : m \mapsto H(m)^d \bmod n$  for hash  $H : \{0, 1\}^* \rightarrow (\mathbb{Z}/n\mathbb{Z})^\times$  (i.e., a claw-free permutation)

$Verify : (m, \sigma) \mapsto H(m) \stackrel{?}{=} \sigma^e \bmod n$

We can prove that if the RSA assumption holds<sup>1</sup> and the hash function  $H$  is a random oracle, then Hashed RSA is EUF-CMA.

**Lecture 21 Hashed RSA (11/02)**

Recall EUF-CMA and Hashed RSA. We want to prove

**Theorem**

Hashed RSA is EUF-CMA assuming:

- The RSA assumption holds
- The hash functions  $H$  are random oracles

*Proof.* For a contradiction, let  $\mathcal{A}$  be an adversary that wins the EUF-CMA game, generating a forged signature  $(m_*, \sigma_*)$ . Note that we must expose the hash function  $H$  to the adversary.

Consider when  $H$  has the property that for some  $\sigma \in (\mathbb{Z}/n\mathbb{Z})^\times$ ,  $H(m_*) = m\sigma^e$ . Then,  $\sigma_* = H(m_*)^d = (m\sigma^e)^d = m^d\sigma$  so  $\sigma_*\sigma^{-1} = m^d$ . We could return  $H(m_*) = m\sigma^e$  but we

<sup>1</sup>Given  $n$ ,  $e$ ,  $m^e$ , it is infeasible to find  $m$ .

still have to respond to signing queries of  $\mathcal{A}$  somehow.

To respond to a query for  $m_i$ , pick a random  $\sigma_i \xleftarrow{\$} (\mathbb{Z}/n\mathbb{Z})^\times$  and set  $H(m_i) = \sigma_i^e$  and respond with  $\sigma_i$ . Note that the challenger must maintain a table of  $H(m_i)$  to respond to duplicates.

To make this work somehow, we define  $H(m) = \begin{cases} m\sigma^e & \text{with probability } \frac{1}{q+1} \\ \sigma^e & \text{with probability } \frac{q}{q+1} \end{cases}$ .

Then, notice that the adversary will make at most  $q + 1$  relevant hash function requests ( $q$  for the signing queries, 1 for  $m_*$ ). Now, the probability that we get what we want, i.e., calculate  $m^d$ , is  $\left(\frac{q}{q+1}\right)^q \frac{1}{q+1} \text{Adv}(\mathcal{A}) > \frac{1}{(q+1)\exp(1)} \text{Adv}(\mathcal{A})$  which is non-negligible since  $q$  is polynomial and  $\text{Adv}(\mathcal{A})$  is non-negligible.

That is, we can break RSA in probabilistic polynomial time with non-negligible probability, violating the RSA assumption. Therefore,  $\mathcal{A}$  cannot exist.  $\square$

Note: non-negligible means that there exists an  $n$  such that  $\Pr[\mathcal{A} \text{ wins}] = f(\lambda) \in \Omega(\frac{1}{\lambda^n})$ .

Further reading: EdDSA (Schnorr), “Short signatures without random oracles” (Boneh–Boyen)

## Lecture 22 Zero-Knowledge Proofs (11/04)

Suppose that  $x \in (\mathbb{Z}/n\mathbb{Z})^\times$  where  $n = pq$ .

Claim: there exists a  $y$  such that  $x = y^2 \pmod n$ .

If Alice knows that  $x = y^2$  and sends  $y$  to Bob, that is a full-knowledge proof. A zero-knowledge proof would not send  $y$ .

Instead, Alice chooses a random  $r \xleftarrow{\$} (\mathbb{Z}/n\mathbb{Z})^\times$  and computes  $xr^2 = y^2r^2 = (yr)^2$ . If she sends  $\beta = xr^2$  and  $\alpha = yr$ , Bob can verify that  $\alpha^2 = \beta$ . However, Bob cannot trust that  $\alpha$  is in fact  $yr$  and cannot prove that  $\frac{\beta}{x} = r^2$  without sending  $r$ .

**Protocol** For Alice to prove that she knows  $y$  such that  $y^2 = x$ ,

1. Alice picks  $r \xleftarrow{\$} (\mathbb{Z}/n\mathbb{Z})^\times$  and sends  $xr^2$
2. Bob picks  $b \xleftarrow{\$} \{0, 1\}$  and sends  $b$
3. Alice sends  $\rho = y^b r$  and sends  $\rho$
4. Bob verifies that  $\rho^2 = \beta x^{b-1}$

Then, if  $b = 0$ , Bob can catch a forged  $y$  and if  $b = 1$ , Bob is more certain that  $y$  exists. The chance that Alice is cheating and avoids being caught in  $\lambda$  iterations is  $2^{-\lambda}$ .

Suppose Alice does not know a square root  $y = \sqrt{x}$ . She could:

- Choose  $r$  randomly, send  $\beta = xr^2$ , and hope that  $b = 0$  to send  $r$
- Choose  $\alpha$  randomly, send  $\beta = \alpha^2$ , and hope that  $b = 1$  to send  $\alpha$

meaning that Alice can forge with success probability 50%, and indeed Bob could fool himself half the time by doing this himself. That is, a zero-knowledge proof does not introduce any new information that Bob could not have produced on his own.



Then, a security definition for a ZKP protocol requires

1. Correctness: With an honest prover and honest verifier, the proof succeeds with probability 100%.
2. Soundness: With a dishonest prover and honest verifier, then there is a non-negligible probability that they get caught.
3. Zero-knowledge: With an honest prover and dishonest verifier, then the verifier can simulate correct proofs with non-negligible probability. This means that the verifier cannot actually use any information for anything else (e.g., cannot factor a number even if the ZKP proves that the prover knows the factors).

where non-negligible means any useful number (e.g., 50%, 25%, 30%, etc.).

From a ZKP, we can construct a signature scheme. Generate a key  $(x, y)$  where  $y^2 = x$ . Signing is done by:

1. Alice picks random  $r$  and sends  $x$  and  $\beta = xr^2$ .
2. Bob picks random  $b$  and sends  $b$
3. Alice calculates  $\rho = ry^b$  and sends  $\rho$

To verify, ensure that  $\rho^2 = \beta x^{-1}$ .

Alternatively, if we want to use DLOG (i.e., with  $g$  and  $g^x$ , prove that you know  $x$ ):

1. Alice picks random  $r$  and sends  $g$ ,  $g^x$ , and  $\beta = g^r$
2. Bob picks random bit  $b$  and sends  $b$
3. Alice calculates  $\rho = r + bx$  and sends  $\rho$

To verify, ensure that  $g^\rho = \beta(g^x)^b$ .

We call these  $\Sigma$  protocols because the back-and-forth looks like a  $\Sigma$ .

**Definition** (*Fiat-Shamir transformation*)

To transform a ZKP protocol to a signing scheme, set  $b = H(\beta, m)$  where  $H$  is a random oracle. Then, the signature of  $m$  is  $(\beta, \rho)$ . To verify, assert  $\rho$  satisfies the ZKP protocol given  $\beta$ .

Notice that there is only one bit of entropy, so it is forgeable 50% of the time. If we try increasing entropy in the DLOG scheme by making  $b \in \mathbb{Z}$ , correctness and soundness still hold but zero-knowledge might not.

## Lecture 23 ZKP Signatures (11/07)

Recall the Fiat-Shamir transform:

Given a  $\Sigma$  protocol, Peggy sends Victor the problem  $\pi$  and a commitment  $c$  (usually some sort of randomized value). Victor returns a challenge  $b$ . Peggy's response  $r$  depends on  $b$ .

From the perspective of a signature scheme, we generate a private key (statement to be proved) and public key  $\pi$ .

To sign, choose a commitment at random  $c \xleftarrow{\$} C$ . Set the challenge to be a deterministic but random function  $b = H(m, c)$ . Calculate a response for  $b$ . Then, let  $\sigma = (c, r)$ .

To verify, recalculate the challenge and verify with the response.

Notice that if  $b \in \{0, 1\}$ , then signature forgery is permitted half the time. To actually use this, the challenge space must be large.

Generically, the signer repeats the protocol  $\lambda$  times and initially commit to a commitment vector  $\mathbf{c} = (c_1, \dots, c_\lambda)$ . They also make a challenge vector  $\mathbf{b} = (b_1, \dots, b_\lambda) = H(m, \mathbf{c})$ . Then, to successfully fake the proof (and forge a signature), the signer would need to very luckily get a  $\mathbf{b}$  that matches perfectly with a malicious  $\mathbf{c}$ . Finally, generate a response vector  $\mathbf{r}$  and return  $\sigma = (\mathbf{c}, \mathbf{r})$ .

This cannot be proved to be ZK because the proof cannot be simulated. We assume that the heuristic (that ZKP's produce ZKP's) holds.

**Example 23.1.** Why do we need to generate the vectors at once?

WLOG, suppose we want to prove knowledge of  $x$  (in the DLOG problem).

The public key is  $\pi = g^x$ . The commitment is  $c = g^y$  for random  $y$ . The challenge is a bit  $b$ . The response is  $r = y + bx$ .

If Peggy predicts  $b = 0$ , pick  $y$  randomly, set  $c = g^y$ , and Victor verifies  $r = y$ . Otherwise, if she predicts  $b = 1$ , pick  $r_0$  randomly, set  $c = \frac{g^{r_0}}{g^x}$ , and Victor verifies  $r = r_0$ .

If Peggy continuously generates random  $y$ , she only has to try twice until getting desired  $b = 0$ . Then, she only needs to do  $2\lambda$  work instead of  $2^\lambda$  work.

What we're describing is the Schnorr signature.

**Schnorr scheme** Given a group  $G$  and  $g \in G$ , generate keys  $(pk, sk) = (g^x, x)$ .

A signature of  $m$  is a proof of knowledge of  $x$ . First, generate a commitment  $r \xleftarrow{\$} \mathbb{Z}$ ,  $c = g^r$ . Then, calculate a challenge  $b \leftarrow H(m, c) = H(m, g^r)$ . The response is  $r + bx$ , so return  $(c, \sigma) = (g^r, r + bx)$ .

To verify a signature  $(c, \sigma)$ , check if  $g^\sigma \stackrel{?}{=} c(g^x)^{H(m, c)}$ . That is, compute  $b' \leftarrow H(m, c) = H(m, g^r)$  and check if  $g^{r+bx} = g^r(g^x)^b \stackrel{?}{=} c(pk)^{b'}$ .

Alternatively, send  $(b, \sigma)$ . Then, calculate  $c' \leftarrow \frac{g^\sigma}{(g^x)^b}$  and check if  $H(m, c') \stackrel{?}{=} b$ . This is better since  $b$  is an integer, which is easier to serialize and send than a group element  $c$ .

**Theorem (Schnorr)**

Assuming that  $H$  is a random oracle and that DLOG is hard in  $G$ , the Schnorr signature scheme is EUF-CMA.

*Proof.* Suppose we are an adversary  $\mathcal{A}_1$  trying to solve DLOG. Let  $g, g^\alpha$  be a challenge from  $\mathcal{C}_1$ . Suppose also that we are a challenger  $\mathcal{C}_2$  with access to an adversary  $\mathcal{A}_2$  that breaks Schnorr.

Give  $\mathcal{A}_2$  the parameter  $g^\alpha$ . Then, the adversary forges a signature  $(b, r + b\alpha)$ . We want to isolate  $\alpha$ , so we need two signatures with the same public key, same commitment, but with different hashes  $b$  and  $b'$ . Using the *forking lemma*, we stop execution before the hashing and swap out the hash function  $H$ .

Then, we have  $(b, r + b\alpha)$  and  $(b', r + b'\alpha)$ . We can now solve for  $\alpha$  and return it to  $\mathcal{C}_1$ .

Since  $\mathcal{A}_2$  runs in poly. time, we  $(\mathcal{A}_1)$  ran in poly. time, meaning that DLOG is easy.  $\square$

## Lecture 24 CCA2-Secure Signature Schemes (11/09)

Recall: in IND-CCA2,  $\mathcal{A}$  can use a decryption oracle, then produce two messages.  $\mathcal{C}$  picks a random one of the two and encrypts it. Then,  $\mathcal{A}$  gets access to the decryption oracle and wins if they can distinguish which message was encrypted.

In Textbook RSA,  $E(m) = m^e \bmod n$ , so an attacker can pick garbage  $k$  and ask for the decryption of  $m^e k^e$ . The core issue here is that  $E$  is a group homomorphism, i.e.,  $E(m_1 m_2) = E(m_1) E(m_2)$ .

*Remark.* Any homomorphic cryptosystem is not CCA2-secure.

For example, Rabin encryption  $E(m) = m^2 \bmod n$  is homomorphic and Elgamal  $E(m) = (g^y, g^{xy}m)$  is also homomorphic in each entry.

**Symmetric + asymmetric hybrid** Let  $KeyGen, Enc : M \rightarrow C$ , and  $Dec : C \rightarrow M$  be a public key cryptosystem. Also let  $\mathcal{E}nc$  and  $\mathcal{D}ec$  be a symmetric key cryptosystem.

Suppose Alice wants to send to Bob. Bob generates  $(pk, sk) \xleftarrow{\$} KeyGen$  and publishes  $pk$ .

Alice picks random  $\sigma \xleftarrow{\$} M$ , encrypts both  $c = Enc(pk, \sigma)$  and  $d = \mathcal{E}nc(\sigma, m)$ , and sends  $(c, d)$ . Notice that we can reinterpret  $\sigma$  as a key for the SKC by just treating it as an appropriately-sized bistring.

Bob can now decrypt  $(c, d)$  by first decrypting  $\sigma = Dec(sk, c)$  and then  $m = \mathcal{D}ec(\sigma, d)$ .

**Fujisaki–Okamoto** (1999) is a CCA2-secure one-time pad (OTP) hybrid. Let  $KeyGen, Enc, Dec$  be a PKC. Then, make a pseudo-OTP  $\mathcal{E}nc(k, m) = m \oplus H_1(k)$  and add a MAC  $H_2(k, m)$ .

Generate  $(pk, sk) \xleftarrow{\$} KeyGen(1^\ell)$  and pick  $\sigma \xleftarrow{\$} M$ .

Then,  $E(m) = (Enc(pk, \sigma), m \oplus H_1(\sigma), H_2(\sigma, m))$ .

To invert,  $D(c, d, e) = d \oplus H_1(Dec(sk, c)) = m$  and check  $H_2(\sigma, m) = e$ . If the MAC does not check out, either explicitly error or implicitly output random garbage  $H_3(s, (c, d, e))$  with a secret seed  $s$ .

Then, the CCA2 oracle is sabotaged.

## Lecture 25 Proving Fujisaki–Okamoto Security (11/11)

Recall the Fujisaki–Okamoto inputs:  $KGen : 1^\ell \mapsto (pk, sk)$ ,  $Enc : M \rightarrow C$ ,  $Dec : C \rightarrow M$ ,  $H_1 : M \rightarrow \{0, 1\}^n$ , and  $H_2 : \{0, 1\}^n \times M \rightarrow T$ .

Then,  $\mathcal{E}nc(pk, m) = (Enc(pk, r), m \oplus H_1(\sigma), H_2(m, \sigma))$  where  $m \in \{0, 1\}^n$  and  $\sigma \xleftarrow{\$} M$ .

### Theorem

If the original PKC is OW-CPA and  $H_1, H_2$  are reandom oracles, then this basic Fujisaki–Okamoto is IND-CPA.

*Proof.* Let  $\mathcal{A}$  be an adversary that can win IND-CPA for FO. Recall IND-CPA: let  $m_0, m_1 \leftarrow \mathcal{A}(1^\ell, pk)$  and  $b' \leftarrow \mathcal{A}(1^\ell, pk, \mathcal{E}nc(pk, m_b))$ . Then,  $\mathcal{A}$  can find  $b = b'$  with

non-negligible probability.

Notice that the second term  $m \oplus H_1(\sigma)$  is garbage since  $\sigma$  is random so  $m$  is randomly scrambled. Therefore, it is information-theoretically indistinguishable from random garbage. So the only way to get any information about  $m$  is to find  $\sigma$ .

Therefore,  $\Pr[\mathcal{A} \text{ wins}] \leq \Pr[\mathcal{A} \text{ finds } \sigma]$ .

Suppose we are challenged to break the PKC in the OW-CPA game and are given  $(pk, \sigma)$ .

Then, we can challenge  $\mathcal{A}$  with  $(Enc(pk, \sigma), \tau, \mu)$  with random garbage  $\tau, \mu$ . Then, at some point  $\mathcal{A}$  must call  $H_1(\sigma)$ . We intercept all the calls to  $H_1$  (since we control  $H_1$ ) and respond with  $\sigma$  with non-negligible probability. Therefore, if FO is not IND-CPA, then the PKC is not OW-CPA.  $\square$

This reduction is not *tight* because we randomly pick potential  $\sigma$  candidates. If the original PKC is deterministic, then we can re-encrypt all potential  $\sigma$  to find the right one.

### Theorem

If  $Enc$  is deterministic, the PKS is OW-CPA, and  $H_1, H_2$  are random oracles, then FO is IND-CCA2.

*Proof.* First, notice that the IND-CCA2 game without the decryption oracle is the IND-CPA game.

However, in FO, we claim the decryption oracle is “useless” because there is no information-theoretic use of it. Therefore, since FO is IND-CPA, it is also IND-CCA2.

To prove the claim, consider  $\mathcal{E}nc(pk, m) = (Enc(pk, \sigma), m \oplus H_1(\sigma), H_2(m, \sigma))$ .

$$\text{Then, } Dec(sk, (c_1, c_2, c_3)) = \begin{cases} \underbrace{H_1(Dec(sk, c_1))}_{m'} \oplus c_2 & \text{otherwise} \\ \perp & c_3 \neq H_2(m', \sigma') \end{cases}$$

Since encryption is deterministic, the only way to construct a valid ciphertext that gets a return value is to know both  $m$  and  $\sigma$  to calculate  $Enc(pk, m)$  and  $H_2(m, \sigma)$ .

We can simulate this for the adversary by intercepting calls to  $H_2$  and checking if  $\sigma$  matches the encryption of  $m$  (i.e.  $c_1$ ). Therefore, there is no difference between IND-CPA and IND-CCA2.  $\square$

**Full Fujisaki–Okamoto** Instead of using  $Enc(pk, \sigma)$ , randomize to  $Enc(pk, \sigma; r)$ . For example, in Elgamal,  $Enc(g^x, \sigma; r) = (g^r, g^{xr}\sigma)$ .

Then,  $\mathcal{E}nc(pk, m) = (Enc(pk, \sigma; H_2(m, \sigma)), m \oplus H_1(\sigma))$  for  $\sigma \xleftarrow{\$} M$ . That is, we use the tag as the randomness.

$$\text{Finally, } Dec(sk, (c_1, c_2)) = \begin{cases} \underbrace{H_1(Dec(sk, c_1))}_{m'} \oplus c_2 & \text{otherwise} \\ \perp & c_1 \neq Enc(pk, \sigma'; H_2(m', s')) \end{cases}$$

### Theorem

If the PKC is OW-CPA and  $H_1, H_2$  are random oracles, then full FO is IND-CCA2.

What if we don't have random oracles? Cramer–Shoup (1998) gets IND-CCA2 using DDH

and a collision-resistant hash function. It is also stupid complicated.

Given a group  $|G| = q$  with two generators  $g_1, g_2$  where  $\langle g_1 \rangle = \langle g_2 \rangle = G$ .

The  $sk = (x_1, x_2, y_1, y_2, z) \in (\mathbb{Z}/q\mathbb{Z})^5$  and  $pk = (c, d, h) = (g_1^{x_1} g_2^{x_2}, g_1^{y_1} g_2^{y_2}, g_1^z)$ .

Encryption is  $Enc(pk, m) = (g_1^r, g_2^r, h^r m, c^r d^{rH(g_1^m, g_2^m, h^r m)})$  for  $m \in G$  and  $r \xleftarrow{\$} \mathbb{Z}/q\mathbb{Z}$ .

Then, the last part  $c^r d^{r\alpha}$  acts as a checksum. To generate a valid ciphertext and use the CCA2 oracle, an adversary must generate this, which breaks DDH.

## Chapter 5

# Elliptic Curve Cryptography

### Lecture 26 Elliptic Curves (11/14)

Recall the conic sections:  $y^2 = 1 - x^2$  (circles),  $y^2 = x^2 - 1$  (hyperbola), etc. If we replace the quadratic in  $x$  with a cubic, we get an elliptic curve. For our purposes,

**Definition** (*curve*)

The set of points satisfying  $f(x, y) = 0$  where  $f \in K[x, y]$  for a field  $K$ .

where  $K$  is a (usually finite) field, e.g.,  $\mathbb{Z}/p\mathbb{Z}$  or something funny like  $\mathbb{Z}/3\mathbb{Z}[i] = \mathbb{F}_9$ .

Note that we can rewrite any cubic  $ax^3 + bx^2 + cx + d$  by first dividing through by  $a$  to get  $x^3 + b'x^2 + c'x + d'$ . Then, send  $x \mapsto x - \frac{b'}{3}$  to get  $x^3 + c''x + d''$ . This only works if  $3 \neq 0$  so we can divide by 3, i.e., the characteristic of  $K$  is not 3.

To simplify the quadratic in  $y$ , we can complete the square as long as  $2 \neq 0$ , i.e., the characteristic of  $K$  is not 3.

**Definition** (*elliptic curve*)

A solution set to an equation of the form  $y^2 = x^3 + ax + b$  where  $a, b \in K$  and  $\text{char}(K) \neq 2, 3$ .

Consider an ellipse centered at the origin with semimajor axes  $a$  and  $b$ . Then, the arc length is  $\int_0^{\pi/2} \sqrt{a^2 \cos^2 t + b^2 \sin^2 t} dt$ . Make the substitution  $u = \sin t$ ,  $du = \cos t dt$  to get  $\int \sqrt{a^2 - \frac{(a^2 - b^2)u^2}{1 - u^2}} du$ . Then,  $k^2 = 1 - \frac{b^2}{a^2}$  gives  $\int a \sqrt{\frac{1 - k^2 u^2}{1 - u^2}} du$  and finally  $x = 1 - k^2 u^2$  for  $\frac{1}{2} \int_{1-k^2}^1 \frac{x dx}{\sqrt{x(x-1)(x-(1-k^2))}}$ . This is our *elliptic integral*.

Generally, elliptic integrals of the first kind  $\int \frac{dx}{\sqrt{x^3 + \dots}}$  and of the second kind  $\int \frac{x dx}{\sqrt{x^3 + \dots}}$ .

Just as  $\int \frac{dx}{\sqrt{x^2 + \dots}}$  gives  $\sin^{-1}(x)$ , the inverse of a periodic function, complex analysis shows that elliptic integrals of the first kind gives the inverse of the doubly periodic Weierstrass function  $\wp^{-1}(x)$ .

By analogy to circles which can be defined by  $f^2 + f'^2 = 1$ , elliptic integrals of the first kind satisfy  $\wp'^2 = 4\wp^3 + c_1\wp + c_2$ .

Elliptic curves have a somewhat natural group law.

**Lemma**

Every line intersects an elliptic curve in exactly three places (up to multiplicity).

*Proof.* Let  $y^2 = x^3 + ax + b$  be an elliptic curve.

Consider a line  $L$  through  $P = (x_P, y_P)$  and  $Q = (x_Q, y_Q)$ . Then, the slope of  $L$  is  $\frac{y_Q - y_P}{x_Q - x_P} = m$ . Substitute  $y = mx + c$  into the elliptic curve to get a cubic in  $x$ . The cubic has three roots.

Then, it is  $(x - x_P)(x - x_Q)(x - x_R)$ . Taking the coefficient on  $x^2$ , we have  $x_R = m^2 - x_P - x_Q$  and  $y_R = m((m^2 - x_P - x_Q) - x_P) + y_P$ .  $\square$

Define the group law as  $P + Q = (x_R, -y_R)$ .

What is  $P + P$ ? Take the tangent line to  $P$ , i.e., “ $\lim_{Q \rightarrow P}(P + Q)$ ”.

What if there is no tangent line (self-intersection or cusps)? To ensure this cannot happen, assert that the discriminant  $4a^3 + 27b^2 \neq 0$ .

What is the identity? Not really a point, denote  $\infty$  or  $[0 : 1 : 0]$  in Sage.

**Lecture 27 (11/16)****Lecture 28 Attacks on ECDH (11/18)**

Recall: we have an elliptic curve  $y^2 = x^3 + ax + b$  over a field  $K$  and a group law  $x_{P+Q} = m^2 - x_P - x_Q$  and  $y_{P+Q} = y_P - m(x_{P+Q} - x_P)$  where  $m = \frac{y_Q - y_P}{x_Q - x_P}$  if  $x_P \neq x_Q$  and  $m = \frac{3x_P^2 + a}{2y_P}$  if  $P = Q$ .

We denote this as  $E(K) = \{\overbrace{(x, y) \in K^2 : y^2 = x^3 + ax + b}^{\text{affine (finite) points}}\} \cup \{\infty\}$ , so  $E(K) \subset K^2 \cup \{\infty\}$ .

We can show by Hasse–Weil that for any elliptic curve over  $\mathbb{F}_q$  with prime power  $q$ ,  $|E(\mathbb{F}_q)| = q + 1 - t$  for a trace of Frobenius  $|t| \leq 2\sqrt{q}$ .

Consider the size of  $E(\mathbb{F}_p)$ . We want to find square roots of  $x^3 + ax + b$  for all  $x$  to find  $y$ . There are two square roots  $(x, \pm y)$ , one square root  $(x, 0)$ , or potentially none. We can show that

$$|\{P \in E : x_P = x_0\}| = \left(\frac{x_0^3 + ax_0 + b}{p}\right) + 1$$

This gives us

$$|E(\mathbb{F}_p)| = 1 + \sum_{x \in \mathbb{F}_p} \left(1 + \left(\frac{x^3 + ax + b}{p}\right)\right) = p + 1 + \sum_{x \in \mathbb{F}_p} \left(\frac{x^3 + ax + b}{p}\right)$$

and we can say that  $t = -\sum \left(\frac{x^3 + ax + b}{p}\right)$ .<sup>1</sup>

Then,  $|E(\mathbb{F}_p)| \approx O(p)$ , so a generic DLOG algorithm over  $E$  should take around  $O(\sqrt{p})$  steps. For some elliptic curves, this is the best we can do.

This is very attractive. To get time approximately  $2^{128}$ ,  $p$  only needs to be  $2^{256}$  whereas the NFS would require  $n \geq 2^{3072}$ . This means faster computation with smaller keys.

<sup>1</sup>Consider that  $\sum \left(\frac{x}{p}\right) \leq \sqrt{p} \ln p$

There are some issues.

**CRT attack** Suppose that  $|E(\mathbb{F}_p)| = p + 1 - t = p_1 p_2$  for small primes (or in general, that it is  $q$ -smooth). Then, Pohlig–Hellman allows us to find DLOG by the CRT in about  $O(\sqrt{p_1} + \sqrt{p_2})$  time.

This is because  $E \simeq \mathbb{Z}/p_1 p_2 \mathbb{Z} \simeq \mathbb{Z}/p_1 \mathbb{Z} \times \mathbb{Z}/p_2 \mathbb{Z}$ . We can compute these isomorphisms. Let  $P \in E$  with order  $p_1 p_2$  and  $Q = \alpha P$ . Then,  $p_1 Q$  has order  $p_2$  and  $p_2 Q$  has order  $p_1$ .

That is,  $p_1 \alpha P = p_1 Q$  is in  $\mathbb{Z}/p_2 \mathbb{Z}$  and we can solve DLOG here to obtain  $\alpha \bmod p_2$ . Likewise with  $p_2$  to find  $\alpha \bmod p_1$ . Then, by CRT, we can find  $\alpha$ .

**Invalid curve attack** Let an otherwise secure curve  $y^2 = x^3 + ax + b$  over  $|E(\mathbb{F}_p)| = q$ . Notice that the equations to calculate  $P + Q$  do not use  $B$ .

Suppose Alice generates  $A = \alpha P$  and Bob  $B = \beta P$  so that they calculate  $\alpha B$  and  $\beta A$ , respectively. If Bob instead sends  $B \in E' : y^2 = x^3 + ax + b'$  where  $|E'(\mathbb{F}_p)| = 3 \dots$  (or smooth or otherwise insecure). Then, Alice instead computes  $\alpha B \in E'$  and Bob can find  $\alpha \bmod 3$ . Repeating, Bob can recover Alice's key by CRT.

To avoid this, just check that  $B \in E$ . Alternatively, express  $P + Q$  using  $b$ . When we are doubling  $P + P$ ,  $m^2 = \frac{(3x_P^2 + a)^2}{4y_P^2} = \frac{(3x_P^2 + a)^2}{4(x_P^3 + ax_P + b)}$  so then  $x_{2P} = \frac{(3x_P^2 + a)^2}{4(x_P^3 + ax_P + b)} - 2x_P$ .

Since this relies only on  $x_P$ , we can only send the  $x$ -coordinate in ECDH. That is, Alice sends  $x_{\alpha P}$  (i.e.,  $\pm \alpha P$ ) and Bob sends  $x_{\beta P}$  (i.e.,  $\pm \beta P$ ). They both calculate  $\pm \alpha \beta P = \pm \beta \alpha P$ , so  $x_{\alpha \beta P}$  is the shared secret.

## Lecture 29 Pairing-Based Cryptography (11/21)

Recall: starting at an elliptic curve  $E$ , we get a group  $E(\mathbb{F}_p)$  and from there get ECDLOG and ECDH. Applying the idea of Schnorr signatures gives us EdDSA.<sup>2</sup> CCA2 security can be achieved with Cramer–Shoup.

After basic ECC developed, pairing-based and post-quantum isogeny<sup>3</sup>-based cryptosystems developed.

**Definition** (*cryptographic pairing*)

Bilinear<sup>4</sup> non-degenerate<sup>5</sup> map  $e : G \times G \rightarrow G_T$  where (usually)  $|G| = |G_T| = p$ .

**MOV attack** (Menezes–Okamoto–Vanstone) Suppose  $E$  is an elliptic curve admitting a cryptographic pairing. Consider a ECDLOG problem  $P, \alpha P \rightarrow \alpha$ .

Let  $e(P, P) = g$  so that  $e(P, \alpha P) = g^\alpha$  by bilinearity. Then, we can consider the DLOG for  $g$  and  $g^\alpha$  in the new group  $G_T$  which is some finite field  $\mathbb{F}_q^*$ . But the whole point of ECDLOG is that it is harder than DLOG on a similarly sized finite field. Transferring from  $E$  to  $\mathbb{F}_q^*$  made it easy again.

<sup>2</sup>There is a slight difference, where instead of hashing  $H(m, g^r)$ , we hash  $H(m, g^r, g^\alpha)$

<sup>3</sup>SIDH and SIKE broken but CSIDH and SQIsign still unbroken.

<sup>4</sup>i.e.,  $e(g^\alpha, g^\beta) = e(g, g)^{\alpha\beta} = \text{orange}^{\alpha\beta}$  (or in additive notation,  $e(\alpha P, \beta P) = e(P, P)^{\alpha\beta}$ )

<sup>5</sup>i.e., for all  $g$ ,  $(\forall h, e(g, h) = 1_T) \implies g = 1_G$  and for all  $h$ ,  $(\forall g, e(g, h) = 1_T) \implies h = 1_G$



Over time, people found enough use in pairings to make it worth using large enough curves admitting pairings.

**Joux** (2000) 3-party Diffie-Hellman setup. Suppose Alice, Bob, and Carol have keys  $g^a$ ,  $g^b$ , and  $g^c$ . Alice and Bob can generate a shared secret  $g^{ab}$  by normal DH but can't easily add Carol.

But with pairings, each one calculates  $e(g, g)^{abc} = \underbrace{e(g^a, g^b)^c}_{\text{Carol}} = \underbrace{e(g^b, g^c)^a}_{\text{Alice}} = \underbrace{e(g^a, g^c)^b}_{\text{Bob}}$ .

For this to work, we must assume the bilinear Diffie-Hellman assumption: given  $g^a$ ,  $g^b$ ,  $g^c$ , it is infeasible to compute  $e(g, g)^{abc}$ .

Likewise, define bilinear DDH as given  $g^a, g^b, g^c \in G$  and  $h \in G_T$ , it is infeasible to compute  $h \stackrel{?}{=} e(g, g)^{abc}$ .

Note that normal DDH does not hold in a pairing, i.e., given  $g^a, g^b, g^z \in G$ , is  $z = ab$ ? Simply take  $e(g, g^z) = e(g, g)^z \stackrel{?}{=} e(g, g)^{ab} = e(g^a, g^b)$ .

### Proposition

CDH  $\geq_P$  BDH

*Proof.* Suppose CDH is broken, i.e., we can find  $g^{ab}$  from  $g^a$  and  $g^b$ . Then, we can take  $e(g^{ab}, g^c) = e(g, g)^{abc}$ .  $\square$

### Proposition

CDH<sub>T</sub>  $\geq_P$  BDH

*Proof.* Suppose CDH<sub>T</sub> is broken. Then, we can find  $e(g^a, g^b) = e(g, g)^{ab}$  and  $e(g, g^c) = e(g, g)^c$  normally but use CDH<sub>T</sub> to get  $e(g, g)^{abc}$ .  $\square$

## Lecture 30 Divisors (11/23)

### Definition (*divisor*)

Formal sum  $\sum_{P \in E} a_P(P)$  of points  $P \in E$  with integer coefficients  $a_P \in \mathbb{Z}$  where only finitely many  $a_P$  are non-zero.

**Example 30.1.** Given points  $P$  and  $Q$  in  $E$ ,  $(P)$ ,  $-(P)$ ,  $2(P)$ , and  $3(P) - (Q)$  are divisors.

The set of all divisors  $\text{Div}(E)$  is a free  $\mathbb{Z}$ -module with basis  $E(K)$ .<sup>6</sup>

**Example 30.2.** Let  $E : y^2 = x^3 - x$  over some finite field  $\mathbb{F}_p$ . Then, there are roots  $P = (-1, 0)$ ,  $Q = (0, 0)$ , and  $R = (1, 0)$ . We can make divisors  $(P) + (Q) + (R)$  or  $(P) - (R)$ . Notice that  $(P) - (R) \neq (P) + (-(R))$ . Likewise,  $(P + Q) \neq (P) + (Q)$ .

If we define the empty divisor  $\emptyset = \sum_{P \in E} 0(P)$ , notice that we get a group.

We can treat divisors as prime factorizations in disguise. Consider that if  $n = p_1^{\alpha_1} \cdots p_k^{\alpha_k}$ , then  $\log n = \alpha_1 \log p_1 + \cdots + \alpha_k \log p_k$ .

<sup>6</sup>Recall: a vector space  $V$  over a field  $K$  with basis  $B$  is  $V = \{\sum_{b \in B} k_b b : k_b \in K, \text{finite } k_b \text{ are non-zero}\}$ . By analogy, a  $\mathbb{Z}$ -module is a “vector space” over  $\mathbb{Z}$  (because  $\mathbb{Z}$  is not a field).

Then, let  $(n) = \log n$  and we get  $(n) = \alpha_1(p_1) + \dots + \alpha_k(p_k)$ . We have  $(1) = \log 1 = 0$ , the “empty divisor”.

**Definition** (*degree*)

If  $D = \sum a_P(P)$ , then  $\deg(D) = \sum a_P$ .

**Example 30.3.**  $\deg((P) - (\infty)) = 0$  and  $\deg(2(P) - 4(Q)) = -2$ .

**Proposition**

$\deg : \text{Div}(E) \rightarrow \mathbb{Z}$  is a homomorphism, i.e.,  $\deg(D_1 + D_2) = \deg(D_1) + \deg(D_2)$ .

Since this is a homomorphism, we get a kernel  $\ker \deg = \{D \in \text{Div}(E) : \deg(D) = 0\}$  which is the degree zero divisors  $\text{Div}^0(E)$ .

**Definition** (*rational function*)

A quotient of polynomials.

Note: On  $E : y^2 = x^3 + ax + b$ , a rational function  $\frac{f_1(x,y)}{f_2(x,y)}$  is written in two variables.

**Example 30.4.** Consider the polynomial  $2x + 3y + 5y^2 + 7x^2y + y^3$  over the elliptic curve  $y^2 = x^3 + ax + b$ .

Substituting gives  $2x + 5(x^3 + ax + b) + (3 + 7x^2 + x^3 + ax + b)y$ , i.e.,  $f_0(x) + f_1(x)y$  for some polynomials in only  $x$ .

That is, we can always split into a “real” part  $f_0(x)$  and “imaginary” part  $f_1(x)$  and only consider them the way we would consider only  $a + bi$  in  $\mathbb{Z}[i]$ .

Therefore, the set of polynomials  $K[E] = \{f_0 + f_1y : f_0, f_1 \in K[x]\}$ . Then, the set of rational functions  $K(E) = \{\frac{f}{g} : f, g \in K[E]\}$  where  $\frac{f}{g} = \frac{f_0 + f_1y}{g_0 + g_1y} \cdot \frac{g_0 - g_1y}{g_0 - g_1y} = \frac{f_0g_0 + 2f_1g_1 + f_1g_1y^2}{g_0^2 - g_1^2y^2} = \frac{F_0}{G} + \frac{F_1}{G}y$  because we can again substitute out the  $y^2$  terms.

(since these are commutative rings, PMATH 446 says we can get this just by localizing)

**Definition** (*divisor of a polynomial*)

Consider the set of polynomials  $K[x]$  over an algebraically closed field  $K$ . Then,  $\text{div}(f) = \sum_{i=1}^n e_i(r_i)$  where  $r_i$  are the roots of  $f$ ,  $n$  is the number of roots of  $f$ , and  $e_i$  is the multiplicity of the corresponding root.

**Example 30.5.**  $\text{div}(x^3 + 2x^2) = \text{div}(x^2(x + 2)) = 2(0) + 1(-2)$ .

By analogy,  $\log(x^3 + 2x^2) = \log x + \log x + \log(x + 2)$ . Then, if  $(r) = \log(x - r)$ , we have  $(0) + (0) + (-2) = 2(0) + 1(-2)$ .

**Key Observation.** A prime factor of multiplicity  $r$  corresponds exactly with a root of multiplicity  $r$ .

**Definition** (*divisor of a rational function*)

$\text{div}(\frac{f}{g}) = \text{div } f - \text{div } g$ .

**Definition** (*order of vanishing*)

For  $f \in K(E)$  and  $P \in E$ ,  $\text{ord}_P(f)$  is the multiplicity of the “prime”  $P$  in the factorization of  $f$ , i.e., the coefficient in the divisor.

**Theorem**

$$\text{ord}_P(f \cdot g) = \text{ord}_P(f) + \text{ord}_P(g)$$

Suppose we have polynomials  $f = f_0 + f_1y$  and its “conjugate”  $g = f_0 - f_1y$ . Then,  $\text{ord}_P(f \cdot g) = \text{ord}_P(f_0^2 - f_1^2y^2)$  which is a polynomial in only  $x$ , so we can find the multiplicity normally. Using a symmetry argument, we can then derive other orders.

**Lecture 31 (11/25)**

Let  $e_n(P, Q) = \frac{f_P}{f_Q}$  where  $f_P = \text{div}(P)$

Recall the degree of a single-variable polynomial  $\deg(a + a_1x + \dots + a_dx^d) = d$ . This definition breaks for multi-variable polynomials, so instead define the degree as the number of roots.<sup>7</sup>

For a polynomial  $f \in K[E]$ , we can WLOG write  $f(x, y) = f_0(x) + f_1(x)y$  and say that  $\deg f$  is the number of roots of  $f$ .

**Example 31.1.**  $\deg x = \deg(x - \alpha) = 2$ ,  $\deg y = \deg(y - \alpha) = 3$

We have properties of normal degrees:  $\deg(f \cdot g) = \deg f + \deg g$ ,  $\deg(f + g) = \max\{\deg f, \deg g\}$ .

**Definition (conjugation)**

$$\text{If } f = f_0 + f_1y, \text{ then } \bar{f} = f_0 - f_1y.$$

Since conjugation is an automorphism,  $\deg f = \deg \bar{f}$ . Then,  $\deg(f \cdot \bar{f}) = 2 \deg f$  but we have that  $f \cdot \bar{f} = f_0^2 - f_1^2y^2 = f_0^2 - f_1^2(x^3 + ax + b) \in K[x]$ . We can factor the usual way to get  $n$  linear factors, which each have degree 2, so  $\deg f = \deg(f_0^2 - f_1^2(x^3 + ax + b))$ .

There are three points where  $y = 0$ , i.e.,  $P_i = (r_i, 0)$  where  $r_i$  are the roots of the cubic. Claim that  $\text{div } y = (P_1) + (P_2) + (P_3)$ .

Relate every point  $P = (\alpha, \beta)$  and relate it to a maximal prime ideal generated by  $(x - \alpha, y - \beta)$ , i.e.,  $K[x, y]/(x - \alpha, y - \beta) = K$ . Considering the principal ideal generated by  $(y) = (x - r_1, y - 0)(x - r_2, y - 0)(x - r_3, y - 0)$  which factors into prime ideals. This is why we have  $\text{div } y = (P_1) + (P_2) + (P_3)$  since divisors correspond with prime factorizations.

But we need to find  $\text{ord}_\infty(y)$ . With projective coordinates  $x = \frac{X}{Z}$  and  $y = \frac{Y}{Z}$ , we get  $Y^2Z = X^3 + aXZ^2 + bZ^3$ . Finally, if  $\tilde{x} = \frac{X}{Y}$  and  $\tilde{z} = \frac{Z}{Y}$ , then we have  $\tilde{z} = \tilde{x}^3 + a\tilde{x}\tilde{z}^2 + b\tilde{z}^3$ . Then, we have  $\text{ord}_\infty(y) = \text{ord}_\infty(\frac{1}{\tilde{z}}) = -\text{ord}_\infty(\tilde{z}) = -\text{ord}_{(0,0)}(\tilde{z}) = -3$ .

So  $\text{div } y = (P_1) + (P_2) + (P_3) - 3(\infty)$  and  $\deg(\text{div } y) = 0$ . In fact,  $\deg(\text{div } f) = 0$  for all  $f$ .

**Example 31.2.** Calculate  $\text{div } x$ .

*Solution.* Likewise,  $\text{div } x = (Q_1) + (Q_2) - 2(\infty)$  because we have two points on the line  $x = 0$  and  $\text{ord}_\infty(x) = \text{ord}_{(0,0)}(\frac{x}{\tilde{z}}) = \text{ord}_{(0,0)}(\tilde{x}) - \text{ord}_{(0,0)}(\tilde{z}) = 1 - 3 = -2$ .  $\square$

**Example 31.3.** Calculate  $\text{div}(\frac{x}{x^2 + y})$ .

*Solution.* First, we have  $\text{div } x - \text{div}(x^2 + y)$ .

Using the conjugate trick,  $\text{div}(x^2 + y) + \text{div}(x^2 - y) = \text{div}(x^4 - (x^3 + ax + b))$  which factors

<sup>7</sup>Assuming the polynomial is separable and the field is algebraically closed, counting multiplicities.

$\operatorname{div} \prod (x - e_i) = \sum \operatorname{div}(x - e_i)$  with roots  $e_i$ . Then, we have  $\sum((P_i) + (-P_i)) - 8(\infty)$ . By symmetry,  $\operatorname{div}(x^2 + y) = \sum(\pm(P_i)) - 4(\infty)$ .  $\square$