

CO 485/685 Fall 2022:

Lecture Notes

1	Introduction to Cryptography	2
Lecture 1	(09/07; skipped)	2
Lecture 2	Almost-Public Key Cryptosystems (09/09)	2
Lecture 3	A Public Key Cryptosystem – RSA (09/12)	3
Lecture 4	Security Definitions (09/14)	3
Lecture 5	Actual IND-CPA systems (09/16)	4
2	Quadratic Residues	6
Lecture 6	Number Theory Background (09/19)	6
Lecture 7	Squares Under a Modulus (09/21)	6
Lecture 8	Squares cont'd (09/23)	7
Lecture 9	Applying to DDH (09/26)	8
Lecture 10	Quadratic Characters in the Complex Plane (09/28)	9
Lecture 11	Quadratic Reciprocity (09/30)	10
3	Primality	13
Lecture 12	Primality Testing (10/03)	13
Lecture 13	Strong Primality Testing (10/05)	14
Lecture 14	Malleability (10/07)	15
Lecture 15	Factorization Algorithms (10/17)	16
Lecture 16	Better Sieves (10/19)	17
Lecture 17	(10/21)	18
Lecture 18	Index Calculus (10/26)	18
Lecture 19	Hash Functions (10/28)	19

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
 - $Dec : K_2 \rightarrow C \rightarrow M : (n, d) \mapsto c \mapsto c^d \bmod m$
- 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 λ
 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 $a \equiv g^k$ 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 ζ_{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 β 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, α 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., ± 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 ℓ 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^ℓ 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.¹

¹Where $t = \pi(B)$ 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 $L(1 - \alpha, \frac{1-\alpha}{c})$.² 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 $O(2\sqrt{n})$.

²Where $L_n(\alpha, c) = O(\exp(c(\log n)^\alpha (\log \log n)^{1-\alpha}))$. 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 α 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 α 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 α , 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: $x_1 \equiv \sum e_{i,1} \log_g p_i \pmod{p-1}$.³ 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 α .

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.

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 $H(s) = H(s')$.

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

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 $\text{HashGen} : \mathbb{1}^\lambda \mapsto H_\lambda$. Play the game:

1. Pick a hash function $H_\lambda \xleftarrow{\$} \text{HashGen}(\mathbb{1}^\lambda)$
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 λ .

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_ε .

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_ε because generating pairs of $f(f(\dots))$ and $h(h(\dots))$ cannot create $g(f(\dots))$.