FINAL REVIEW-ADMIN

Final Review-Admin

1) Final is Saturday Dec 14 1:30-3:30 in IRB 0318.

Final Review-Admin

1) Final is Saturday Dec 14 1:30-3:30 in IRB 0318.

2) Can bring one sheet of notes.
Can: use both sides, type it, put what you want on it.
Can: copy a classmates, cram entire course–Bad Ideas
Can: cram THIS talk on it-Bad Idea

Final Review-Admin

1) Final is Saturday Dec 14 1:30-3:30 in IRB 0318.

2) Can bring one sheet of notes.
Can: use both sides, type it, put what you want on it.
Can: copy a classmates, cram entire course–Bad Ideas
Can: cram THIS talk on it-Bad Idea

3) No calculators allowed.

Final Review-Admin

1) Final is Saturday Dec 14 1:30-3:30 in IRB 0318.

2) Can bring one sheet of notes.
Can: use both sides, type it, put what you want on it.
Can: copy a classmates, cram entire course–Bad Ideas
Can: cram THIS talk on it-Bad Idea

3) No calculators allowed.

4) Coverage: Slides/HW. Comprehensive.
5) Not on Exam: LWE, Bridge Cheating, My Book Talk, NSA talk.

Final Review-Admin

1) Final is Saturday Dec 14 1:30-3:30 in IRB 0318.

2) Can bring one sheet of notes.
Can: use both sides, type it, put what you want on it.
Can: copy a classmates, cram entire course–Bad Ideas
Can: cram THIS talk on it-Bad Idea

3) No calculators allowed.

4) Coverage: Slides/HW. Comprehensive.
5) Not on Exam: LWE, Bridge Cheating, My Book Talk, NSA talk.

6) We hope to grade it and post it Saturday Afternoon.

Final Review-Admin

1) Final is Saturday Dec 14 1:30-3:30 in IRB 0318.

2) Can bring one sheet of notes.
Can: use both sides, type it, put what you want on it.
Can: copy a classmates, cram entire course–Bad Ideas
Can: cram THIS talk on it-Bad Idea

3) No calculators allowed.

4) Coverage: Slides/HW. Comprehensive.
5) Not on Exam: LWE, Bridge Cheating, My Book Talk, NSA talk.

6) We hope to grade it and post it Saturday Afternoon.

7) If can't take the exam tell me ASAP.

Final Review-Admin

1) Final is Saturday Dec 14 1:30-3:30 in IRB 0318.

2) Can bring one sheet of notes.
Can: use both sides, type it, put what you want on it.
Can: copy a classmates, cram entire course–Bad Ideas
Can: cram THIS talk on it-Bad Idea

3) No calculators allowed.

4) Coverage: Slides/HW. Comprehensive.
5) Not on Exam: LWE, Bridge Cheating, My Book Talk, NSA talk.

6) We hope to grade it and post it Saturday Afternoon.

7) If can't take the exam tell me ASAP.

8) Advice: Understand rather than memorize.

FINAL REVIEW-CONTENT

Alice, Bob, and Eve

- Alice sends a message to Bob in code.
- Eve overhears it.
- We want Eve to not get any information.

There are many aspects to this:

- Information-Theoretic Security.
- Comp-Theoretic Security (Hardness Assumption)
- NY,NY problem.
- Private Key or Public key
- Kerckhoff's principle: Eve knows cryptosystem.

Private Key Ciphers

Single Letter Sub Ciphers

1. Shift cipher: $f(x) = x + s$. $s \in \{0, \ldots, 25\}$.
2. Affine cipher: $f(x) = ax + b$. $a, b \in \{0, \ldots, 25\}$. a rel prime 26.
3. Keyword Shift: From keyword and shift create random-looking perm of $\{a, \ldots, z\}$.
4. Keyword Mixed: From keyword create random-looking perm of $\{a, \ldots, z\}$.
5. Gen Sub Cipher: Take random perm of $\{a, \ldots, z\}$.

All Single Letter Sub Ciphers Crackable

Important: Algorithm Is-English.

1. Input(T) a text
2. Find f_T, the freq vector of T
3. Find $x = f_T \cdot f_E$ where f_E is freq vector for English
4. If $x \geq 0.06$ then output YES. If $x \leq 0.04$ then output NO. If $0.04 < x < 0.06$ then something is wrong.

How to Use:

1. Shift , Affine have small key space: can try all keys and see when Is-English says YES.
2. For others use freq analysis.
3. If message Credit Cards or ASCII there are patterns; use freq analysis.

Randomized Shift

How to avoid NY,NY Problem:

Randomized shift: Key is a function $f : S \rightarrow S$.

1. To send message $(m_1, \ldots, m_L)$ (each m_i is a character)
   1.1 Pick random $r_1, \ldots, r_L \in S$. For $1 \le i \le L$ compute $s_i = f(r_i)$.
   1.2 Send $((r_1; m_1 + s_1), \ldots, (r_L; m_L + s_L))$
2. To decode message $((r_1; c_1), \ldots, (r_L; c_L))$
   2.1 For $1 \le i \le L$ $s_i = f(r_i)$.
   2.2 Find $(c_1 - s_1, \ldots, c_L - s_L)$

   Note: Can be cracked.

More Advanced Ciphers

1. Vigenère cipher (Can get more out of the phrase using LCM)
2. Book Cipher
3. Matrix Cipher
4. Playfair, Railfence, Autokey
5. General 2-letter sub.

All have their PROS and CONS but all are, in the real world, crackable (today).

One-time pad

1. Let $\mathcal{M} = \{0,1\}^n$
2. *Gen*: choose a uniform key $k \in \{0,1\}^n$
3. $Enc_k(m) = k \oplus m$
4. $Dec_k(c) = k \oplus c$
5. Proof of Correctness:

$$\begin{aligned} Dec_k(Enc_k(m)) &= k \oplus (k \oplus m) \\ &= (k \oplus k) \oplus m \\ &= m \end{aligned}$$

PROS AND CONS Of One-time pad

1. If Key is N bits long can only send N bits.
2. $\oplus$ is FAST!
3. The one-time pad is uncrackable. YEAH!
4. Generating truly random bits is hard. BOO!
5. Pseudo-random can be insecure – I did example of cracking linear Congruential generators.

Public Key Ciphers
Eve can go . . .

Public Key Cryptography

Alice and Bob never have to meet!

Number Theory Algs needed for Public Key

The following can be done quickly.

Number Theory Algs needed for Public Key

The following can be done quickly.

1. Given (a, n, p) compute $a^n \pmod{p}$. Repeated Squaring. (1) $\le 2 \lg n$ always, (2) $\le \lg n + O(1)$ if n close to 2^{2^m}.

Number Theory Algs needed for Public Key

The following can be done quickly.

1. Given (a, n, p) compute $a^n \pmod{p}$. Repeated Squaring. (1) $\le 2\lg n$ always, (2) $\le \lg n + O(1)$ if n close to 2^{2^m}.
2. Given n, find a safe prime of length n and a generator g.

Number Theory Algs needed for Public Key

The following can be done quickly.

1. Given (a, n, p) compute $a^n \pmod{p}$. Repeated Squaring. (1) $\le 2\lg n$ always, (2) $\le \lg n + O(1)$ if n close to 2^{2^m}.
2. Given n, find a safe prime of length n and a generator g.
3. Given a, b rel prime find inverse of a mod b: Euclidean alg.

Number Theory Algs needed for Public Key

The following can be done quickly.

1. Given (a, n, p) compute $a^n \pmod{p}$. Repeated Squaring. (1) $\le 2 \lg n$ always, (2) $\le \lg n + O(1)$ if n close to 2^{2^m}.
2. Given n, find a safe prime of length n and a generator g.
3. Given a, b rel prime find inverse of a mod b: Euclidean alg.
4. Given $a_1, \ldots, a_L$ and $b_1, \ldots, b_L$ with b_i's rel prime, find $x \equiv a_i \pmod{b_i}$.

Number Theory Algs needed for Public Key

The following can be done quickly.

1. Given (a, n, p) compute $a^n \pmod{p}$. Repeated Squaring. (1) $\le 2 \lg n$ always, (2) $\le \lg n + O(1)$ if n close to 2^{2^m}.
2. Given n, find a safe prime of length n and a generator g.
3. Given a, b rel prime find inverse of a mod b: Euclidean alg.
4. Given $a_1, \ldots, a_L$ and $b_1, \ldots, b_L$ with b_i's rel prime, find $x \equiv a_i \pmod{b_i}$.
5. Given (a, p) find $\sqrt{a}$'s. We did $p \equiv 3 \pmod{4}$ case.

Number Theory Algs needed for Public Key

The following can be done quickly.

1. Given (a, n, p) compute $a^n \pmod{p}$. Repeated Squaring. (1) $\le 2 \lg n$ always, (2) $\le \lg n + O(1)$ if n close to 2^{2^m}.
2. Given n, find a safe prime of length n and a generator g.
3. Given a, b rel prime find inverse of a mod b: Euclidean alg.
4. Given $a_1, \ldots, a_L$ and $b_1, \ldots, b_L$ with b_i's rel prime, find $x \equiv a_i \pmod{b_i}$.
5. Given (a, p) find $\sqrt{a}$'s. We did $p \equiv 3 \pmod{4}$ case.
6. Given (a, N) and p, q such that $N = pq$, find $\sqrt{a} \pmod{p}$ (there will probably be two of them and you an find both).

Number Theory Assumptions

1. Discrete Log is hard.
2. Factoring is hard.
3. Given (a, N), find $\sqrt{a}$ without being given factors of N is hard. (This is equiv to factoring.)

Note: We usually don't assume these but instead assume close cousins.

The Diffie-Helman Key Exchange

Alice and Bob will share a secret s. Security parameter L.

The Diffie-Helman Key Exchange

Alice and Bob will share a secret s. Security parameter L.

1. Alice finds a (p, g), p of length L, g gen for $\mathbb{Z}_p$.

The Diffie-Helman Key Exchange

Alice and Bob will share a secret s. Security parameter L.

1. Alice finds a (p, g), p of length L, g gen for $\mathbb{Z}_p$.
2. Alice sends (p, g) to Bob in the clear (Eve sees (p, g)).

The Diffie-Helman Key Exchange

Alice and Bob will share a secret s. Security parameter L.

1. Alice finds a (p, g), p of length L, g gen for $\mathbb{Z}_p$.
2. Alice sends (p, g) to Bob in the clear (Eve sees (p, g)).
3. Alice picks random $a \in \{1, \ldots, p-1\}$, computes g^a and sends it to Bob in the clear (Eve sees g^a).

The Diffie-Helman Key Exchange

Alice and Bob will share a secret s. Security parameter L.

1. Alice finds a (p, g), p of length L, g gen for $\mathbb{Z}_p$.
2. Alice sends (p, g) to Bob in the clear (Eve sees (p, g)).
3. Alice picks random $a \in \{1, \ldots, p-1\}$, computes g^a and sends it to Bob in the clear (Eve sees g^a).
4. Bob picks random $b \in \{1, \ldots, p-1\}$, computes g^b and sends it to Alice in the clear (Eve sees g^b).

The Diffie-Helman Key Exchange

Alice and Bob will share a secret s. Security parameter L.

1. Alice finds a (p, g), p of length L, g gen for $\mathbb{Z}_p$.
2. Alice sends (p, g) to Bob in the clear (Eve sees (p, g)).
3. Alice picks random $a \in \{1, \ldots, p-1\}$, computes g^a and sends it to Bob in the clear (Eve sees g^a).
4. Bob picks random $b \in \{1, \ldots, p-1\}$, computes g^b and sends it to Alice in the clear (Eve sees g^b).
5. Alice computes $(g^b)^a = g^{ab}$.

The Diffie-Helman Key Exchange

Alice and Bob will share a secret s. Security parameter L.

1. Alice finds a (p, g), p of length L, g gen for $\mathbb{Z}_p$.
2. Alice sends (p, g) to Bob in the clear (Eve sees (p, g)).
3. Alice picks random $a \in \{1, \dots, p-1\}$, computes g^a and sends it to Bob in the clear (Eve sees g^a).
4. Bob picks random $b \in \{1, \dots, p-1\}$, computes g^b and sends it to Alice in the clear (Eve sees g^b).
5. Alice computes $(g^b)^a = g^{ab}$.
6. Bob computes $(g^a)^b = g^{ab}$.

The Diffie-Helman Key Exchange

Alice and Bob will share a secret s. Security parameter L.

1. Alice finds a (p, g), p of length L, g gen for $\mathbb{Z}_p$.
2. Alice sends (p, g) to Bob in the clear (Eve sees (p, g)).
3. Alice picks random $a \in \{1, \ldots, p-1\}$, computes g^a and sends it to Bob in the clear (Eve sees g^a).
4. Bob picks random $b \in \{1, \ldots, p-1\}$, computes g^b and sends it to Alice in the clear (Eve sees g^b).
5. Alice computes $(g^b)^a = g^{ab}$.
6. Bob computes $(g^a)^b = g^{ab}$.
7. g^{ab} is the shared secret.

The Diffie-Helman Key Exchange

Alice and Bob will share a secret s. Security parameter L.

1. Alice finds a (p, g), p of length L, g gen for $\mathbb{Z}_p$.
2. Alice sends (p, g) to Bob in the clear (Eve sees (p, g)).
3. Alice picks random $a \in \{1, \ldots, p-1\}$, computes g^a and sends it to Bob in the clear (Eve sees g^a).
4. Bob picks random $b \in \{1, \ldots, p-1\}$, computes g^b and sends it to Alice in the clear (Eve sees g^b).
5. Alice computes $(g^b)^a = g^{ab}$.
6. Bob computes $(g^a)^b = g^{ab}$.
7. g^{ab} is the shared secret.

Definition

Let f be $f(p, g, g^a, g^b) = g^{ab}$.

Hardness assumption: f is hard to compute.

ElGamal Uses DH So Can Control Message

1. Alice and Bob do Diffie Helman.
2. Alice and Bob share secret $s = g^{ab}$.
3. Alice and Bob compute $(g^{ab})^{-1} \pmod{p}$.
4. To send m, Alice sends $c = mg^{ab}$
5. To decrypt, Bob computes
   $c(g^{ab})^{-1} \equiv mg^{ab}(g^{ab})^{-1} \equiv m$

We omit discussion of Hardness assumption (HW)

RSA

Let L be a security parameter

RSA

Let L be a security parameter

1. Alice picks two primes p, q of length L and computes $N = pq$.

RSA

Let L be a security parameter

1. Alice picks two primes p, q of length L and computes $N = pq$.
2. Alice computes $\phi(N) = \phi(pq) = (p-1)(q-1)$. Denote by R

RSA

Let L be a security parameter

1. Alice picks two primes p, q of length L and computes $N = pq$.
2. Alice computes $\phi(N) = \phi(pq) = (p-1)(q-1)$. Denote by R
3. Alice picks an $e \in \{\frac{R}{3}, \ldots, \frac{2R}{3}\}$ that is relatively prime to R. Alice finds d such that $ed \equiv 1 \pmod{R}$.

RSA

Let L be a security parameter

1. Alice picks two primes p, q of length L and computes $N = pq$.
2. Alice computes $\phi(N) = \phi(pq) = (p-1)(q-1)$. Denote by R
3. Alice picks an $e \in \{\frac{R}{3}, \ldots, \frac{2R}{3}\}$ that is relatively prime to R. Alice finds d such that $ed \equiv 1 \pmod{R}$.
4. Alice broadcasts (N, e). (Bob and Eve both see it.)

RSA

Let L be a security parameter

1. Alice picks two primes p, q of length L and computes $N = pq$.
2. Alice computes $\phi(N) = \phi(pq) = (p-1)(q-1)$. Denote by R
3. Alice picks an $e \in \{\frac{R}{3}, \ldots, \frac{2R}{3}\}$ that is relatively prime to R. Alice finds d such that $ed \equiv 1 \pmod{R}$.
4. Alice broadcasts (N, e). (Bob and Eve both see it.)
5. Bob: To send $m \in \{1, \ldots, N-1\}$, send $m^e \pmod{N}$.

RSA

Let L be a security parameter

1. Alice picks two primes p, q of length L and computes $N = pq$.
2. Alice computes $\phi(N) = \phi(pq) = (p-1)(q-1)$. Denote by R
3. Alice picks an $e \in \{\frac{R}{3}, \ldots, \frac{2R}{3}\}$ that is relatively prime to R. Alice finds d such that $ed \equiv 1 \pmod{R}$.
4. Alice broadcasts (N, e). (Bob and Eve both see it.)
5. Bob: To send $m \in \{1, \ldots, N-1\}$, send $m^e \pmod{N}$.
6. If Alice gets $m^e \pmod{N}$ she computes

$$(m^e)^d \equiv m^{ed} \equiv m^{ed \pmod{R}} \equiv m^{1 \pmod{R}} \equiv m$$

Hardness Assumption for RSA

Recall If Alice and Bob do RSA and Eve observes:

Hardness Assumption for RSA

Recall If Alice and Bob do RSA and Eve observes:

1. Eve sees (N, e, m^e). The message is m.

Hardness Assumption for RSA

Recall If Alice and Bob do RSA and Eve observes:

1. Eve sees (N, e, m^e). The message is m.
2. Eve knows that there exists primes p, q such that $N = pq$, but she does not know what p, q are.

Hardness Assumption for RSA

Recall If Alice and Bob do RSA and Eve observes:

1. Eve sees (N, e, m^e). The message is m.
2. Eve knows that there exists primes p, q such that $N = pq$, but she does not know what p, q are.
3. Eve knows that e is relatively prime to $(p-1)(q-1)$.

Hardness Assumption for RSA

Recall If Alice and Bob do RSA and Eve observes:

1. Eve sees (N, e, m^e). The message is m.
2. Eve knows that there exists primes p, q such that $N = pq$, but she does not know what p, q are.
3. Eve knows that e is relatively prime to $(p-1)(q-1)$.

Definition: Let f be $f(N, e, m^e) = m$, where $N = pq$ and e has an inverse mod $(p-1)(q-1)$.
Hardness assumption (HA): f is hard to compute.

Plain RSA Bytes!

The RSA given above is referred to as Plain RSA.
Insecure! m is always coded as $m^e \pmod{N}$.

Make secure by padding: $m \in \{0,1\}^{L_1}$, $r \in \{0,1\}^{L_2}$.

To send $m \in \{0,1\}^{L_1}$, pick rand $r \in \{0,1\}^{L_2}$, send $(rm)^e$.
(NOTE- rm means r CONCAT with m here and elsewhere.)
DEC: Alice finds rm and takes rightmost L_1 bits.
Caveat: RSA still has issues when used in real world.
They have been fixed. Maybe.

Attacks on RSA

Attacks on RSA

1. Factoring Algorithms. Pollard $p-1$, Pollard ρ, QS.
   Response: Pick larger p, q

Attacks on RSA

1. Factoring Algorithms. Pollard $p-1$, Pollard ρ, QS.
   Response: Pick larger p, q
2. If Zelda give A_i (N_i, e):

Attacks on RSA

1. Factoring Algorithms. Pollard $p - 1$, Pollard ρ, QS.
   Response: Pick larger p, q
2. If Zelda give A_i (N_i, e):

Attacks on RSA

1. Factoring Algorithms. Pollard $p-1$, Pollard ρ, QS.
   Response: Pick larger p, q
2. If Zelda give A_i (N_i, e):
   2.1 Low-e attack: Response: High e. Duh.

Attacks on RSA

1. Factoring Algorithms. Pollard $p-1$, Pollard ρ, QS.
   Response: Pick larger p, q
2. If Zelda give A_i (N_i, e):
   2.1 Low-e attack: Response: High e. Duh.
   2.2 $m^e < N_1 \cdots N_L$: Response: Pad m.

Attacks on RSA

1. Factoring Algorithms. Pollard $p-1$, Pollard ρ, QS.
   Response: Pick larger p, q
2. If Zelda give A_i (N_i, e):
   2.1 Low-e attack: Response: High e. Duh.
   2.2 $m^e < N_1 \cdots N_L$: Response: Pad m.
3. If Zelda give A_i (N, e_i) and two of the e_i's are rel prime, then Euclidean Alg Attack: Response: Give everyone diff N's. Duh.

Attacks on RSA

1. Factoring Algorithms. Pollard $p-1$, Pollard ρ, QS. Response: Pick larger p, q
2. If Zelda give A_i (N_i, e):
   2.1 Low-e attack: Response: High e. Duh.
   2.2 $m^e < N_1 \cdots N_L$: Response: Pad m.
3. If Zelda give A_i (N, e_i) and two of the e_i's are rel prime, then Euclidean Alg Attack: Response: Give everyone diff N's. Duh.
4. Timing Attacks: Response: Pad time used.

Caveat: Theory says use different e's. Practice says use $e = 2^{16} + 1$ for speed.

Other Public Key Schemes

Other Public Key Schemes

1. Rabin PRO- equiv to factoring, CON- Alice cannot decode uniquely. CAVEAT-Blum-Williams Variant enables unique decoding.

Other Public Key Schemes

1. Rabin PRO- equiv to factoring, CON- Alice cannot decode uniquely. CAVEAT-Blum-Williams Variant enables unique decoding.
2. GM PRO- equiv to hardness of sqrt mod pq. CON-Can only send one bit.

Other Public Key Schemes

1. Rabin PRO- equiv to factoring, CON- Alice cannot decode uniquely. CAVEAT-Blum-Williams Variant enables unique decoding.
2. GM PRO- equiv to hardness of sqrt mod pq. CON-Can only send one bit.
3. BG PRO- equiv to factoring. No real CON. Might have caught on if history was different.

Factoring Algorithms: Pollard $p-1$

Pollard $p-1$ algorithm

Parameter B and hence also

$$M = \prod_{q \leq B, q \text{ prime}} q^{\lceil \log_q(B) \rceil}.$$

```
FOUND = FALSE
while NOT FOUND
     a=RAND(1,N-1)
     d=GCD(a^M-1,N)
     if d=1 then increase B
     if d=N then decrease B
     if (d NE 1,N) then FOUND=TRUE
output(d)
```

Pollard $p-1$ algorithm

Parameter B and hence also

$$M = \prod_{q \le B, q \text{ prime}} q^{\lceil \log_q(B) \rceil}.$$

```
FOUND = FALSE
while NOT FOUND
     a=RAND(1,N-1)
     d=GCD(a^M-1,N)
     if d=1 then increase B
     if d=N then decrease B
     if (d NE 1,N) then FOUND=TRUE
output(d)
```

KEY If $p-1$ divides M then $a^M - 1 \equiv 0 \pmod{N}$ so $GCD(a^M - 1, N)$ will yield factor.
NOTE Works well if $p-1$ only has small factors so more likely $p-1$ divides M.

Factoring Algorithms: Pollard rho

Birthday Paradox

Concrete Scenario If you have 23 people in a room than the prob that there are two with the same birthday is $\geq \frac{1}{2}$. Note that there are 365 birthdays. View this as putting 23 people into 365 buckets.

Birthday Paradox

Concrete Scenario If you have 23 people in a room than the prob that there are two with the same birthday is $\geq \frac{1}{2}$. Note that there are 365 birthdays. View this as putting 23 people into 365 buckets.

General Scenario If you put $2\sqrt{n}$ balls into n buckets the prob that there are 2 balls in the same bucket is $\geq \frac{1}{2}$.

Pollard ρ Algorithm

Define $f_c(x) \leftarrow x * x + c$. Looks random.

$x \leftarrow RAND(0, N-1)$, $c \leftarrow RAND(0, N-1)$, $y \leftarrow f_c(x)$
while TRUE
    $x \leftarrow f_c(x)$
    $y \leftarrow f_c(f_c(y))$
    $d \leftarrow GCD(x-y, N)$
    if $d \neq 1$ and $d \neq N$ then break
output(d)

Pollard ϕ Algorithm: Though Exp

Let p be the least prime that div N. We do not know p.

Pollard ϕ Algorithm: Though Exp

Let p be the least prime that div N. We do not know p.

The sequence $x, f_c(x), f(f_c(x)), \ldots$ is random-looking.

Pollard ϕ Algorithm: Though Exp

Let p be the least prime that div N. We do not know p.

The sequence $x, f_c(x), f(f_c(x)), \ldots$ is random-looking.

Put each element of the seq into its $\equiv$ class mod p.

Pollard ϕ Algorithm: Though Exp

Let p be the least prime that div N. We do not know p.

The sequence $x, f_c(x), f(f_c(x)), \ldots$ is random-looking.

Put each element of the seq into its $\equiv$ class mod p.

View the $\equiv$-classes as buckets at the sequence as balls.

Pollard ϕ Algorithm: Though Exp

Let p be the least prime that div N. We do not know p.

The sequence $x, f_c(x), f(f_c(x)), \ldots$ is random-looking.

Put each element of the seq into its $\equiv$ class mod p.

View the $\equiv$-classes as buckets at the sequence as balls.

By Bday Paradox there will 2 elements of the seq in same bucket within the first $2\sqrt{p} \le 2N^{1/4}$ with high prob.

Pollard ϕ Algorithm: Though Exp

Let p be the least prime that div N. We do not know p.

The sequence x, $f_c(x)$, $f(f_c(x))$, $\ldots$ is random-looking.

Put each element of the seq into its $\equiv$ class mod p.

View the $\equiv$-classes as buckets at the sequence as balls.

By Bday Paradox there will 2 elements of the seq in same bucket within the first $2\sqrt{p} \le 2N^{1/4}$ with high prob.

By Thm there is an i such that the ith element in same bucket as $2i$th element, some $i \le 3\sqrt{N^{1/4}}$), with high prob.

Pollard ϕ Algorithm: Though Exp

Let p be the least prime that div N. We do not know p.

The sequence x, $f_c(x)$, $f(f_c(x))$, $\ldots$ is random-looking.

Put each element of the seq into its $\equiv$ class mod p.

View the $\equiv$-classes as buckets at the sequence as balls.

By Bday Paradox there will 2 elements of the seq in same bucket within the first $2\sqrt{p} \le 2N^{1/4}$ with high prob.

By Thm there is an i such that the ith element in same bucket as $2i$th element, some $i \le 3\sqrt{N^{1/4}}$), with high prob.

Hence $(\exists x, y)[x \equiv y \pmod{p}]$ so $GCD(x-y, N) \ne 1$.

Pollard ϕ Algorithm: Though Exp

Let p be the least prime that div N. We do not know p.

The sequence x, $f_c(x)$, $f(f_c(x))$, $\ldots$ is random-looking.

Put each element of the seq into its $\equiv$ class mod p.

View the $\equiv$-classes as buckets at the sequence as balls.

By Bday Paradox there will 2 elements of the seq in same bucket within the first $2\sqrt{p} \le 2N^{1/4}$ with high prob.

By Thm there is an i such that the ith element in same bucket as $2i$th element, some $i \le 3\sqrt{N^{1/4}}$), with high prob.

Hence $(\exists x, y)[x \equiv y \pmod p)]$ so $GCD(x - y, N) \ne 1$.

Caveat Need the sequence to be truly random to prove it works. Don't have that, but it works in practice.

Factoring Algorithms: Quad Sieve

Quad Sieve Alg

Given N let $x = \lceil \sqrt{N} \rceil$. All $\equiv$ are mod N. B, M are params.

Quad Sieve Alg

Given N let $x = \lceil \sqrt{N} \rceil$. All $\equiv$ are mod N. B, M are params.

$$
\begin{array}{ll}
(x+0)^2 \equiv y_0 & \text{Try to } B\text{-Factor } y_0 \text{ to get parity } \vec{v}_0 \\
\vdots & \vdots \\
(x+M)^2 \equiv y_M & \text{Try to } B\text{-Factor } y_M \text{ to get parity } \vec{v}_M
\end{array}
$$

Quad Sieve Alg

Given N let $x = \lceil \sqrt{N} \rceil$. All $\equiv$ are mod N. B, M are params.

$$\begin{array}{rl} (x+0)^2 \equiv y_0 & \text{Try to } B\text{-Factor } y_0 \text{ to get parity } \vec{v}_0 \\ \vdots & \vdots \\ (x+M)^2 \equiv y_M & \text{Try to } B\text{-Factor } y_M \text{ to get parity } \vec{v}_M \end{array}$$

Some of the y_i were B-factored, but some were not.
Let I be the set of all i such that y_i was B-factored.

Quad Sieve Alg

Given N let $x = \lceil \sqrt{N} \rceil$. All $\equiv$ are mod N. B, M are params.

$$
\begin{array}{rl}
(x+0)^2 \equiv y_0 & \text{Try to } B\text{-Factor } y_0 \text{ to get parity } \vec{v}_0 \\
\vdots & \vdots \\
(x+M)^2 \equiv y_M & \text{Try to } B\text{-Factor } y_M \text{ to get parity } \vec{v}_M
\end{array}
$$

Some of the y_i were B-factored, but some were not.
Let I be the set of all i such that y_i was B-factored.

Find $J \subseteq I$ such that $\sum_{i \in J} \vec{v}_i \equiv \vec{0} \pmod{2}$.

Quad Sieve Alg

Given N let $x = \lceil \sqrt{N} \rceil$. All $\equiv$ are mod N. B, M are params.

$$
\begin{array}{rl}
(x+0)^2 \equiv y_0 & \text{Try to } B\text{-Factor } y_0 \text{ to get parity } \vec{v}_0 \\
\vdots & \vdots \\
(x+M)^2 \equiv y_M & \text{Try to } B\text{-Factor } y_M \text{ to get parity } \vec{v}_M
\end{array}
$$

Some of the y_i were B-factored, but some were not.
Let I be the set of all i such that y_i was B-factored.

Find $J \subseteq I$ such that $\sum_{i \in J} \vec{v}_i \equiv \vec{0} \pmod 2$.

Hence $\prod_{i \in J} y_i$ has all even exponents.
Important! Since $\prod_{i \in J} y_i$ has all even exponents, there exists Y such that
$\prod_{i \in J} y_i = Y^2$. From this can get $X^2 \equiv Y^2 \pmod N$. DONE!

IDEA: Do the Factoring in Bulk

New Problem Given N, B, M, x, want to B-factor

$$
\begin{aligned}
&(x+0)^2 \pmod{N}\\
&(x+1)^2 \pmod{N}\\
&\quad\vdots \qquad \vdots\\
&(x+M)^2 \pmod{N}
\end{aligned}
$$

We do an example on the next slide.

QS Example: $N = 1147$, $M = 10$, $\lceil \sqrt{N} \rceil = 34$

For which $0 \leq i \leq 10$ is $((34 + i)^2 \bmod N) \equiv 0 \pmod{2}$?

QS Example: $N = 1147$, $M = 10$, $\lceil \sqrt{N} \rceil = 34$

For which $0 \le i \le 10$ is $((34+i)^2 \bmod N) \equiv 0 \pmod 2$?

Need to know what $(34+i)^2 \pmod N$ is.
Key We show $1147 < (34+i)^2 \le 2 \times 1147$ and hence

$$(34+i)^2 \pmod N = (34+i)^2 - 1147$$

QS Example: $N = 1147$, $M = 10$, $\lceil \sqrt{N} \rceil = 34$

For which $0 \le i \le 10$ is $((34+i)^2 \bmod N) \equiv 0 \pmod 2$?

Need to know what $(34+i)^2 \pmod N$ is.
Key We show $1147 < (34+i)^2 \le 2 \times 1147$ and hence

$$(34+i)^2 \pmod N = (34+i)^2 - 1147$$

$(34+i)^2$ is min: $i = 0$. Its $34^2 = 1156 > 1147$.

QS Example: $N = 1147$, $M = 10$, $\lceil \sqrt{N} \rceil = 34$

For which $0 \le i \le 10$ is $((34+i)^2 \bmod N) \equiv 0 \pmod 2$?

Need to know what $(34+i)^2 \pmod N$ is.
Key We show $1147 < (34+i)^2 \le 2 \times 1147$ and hence

$$(34+i)^2 \pmod N = (34+i)^2 - 1147$$

$(34+i)^2$ is min: $i = 0$. Its $34^2 = 1156 > 1147$.
$(34+i)^2$ is max:$i = 10$. Its $44^2 = 1936 < 2 \times 1147$.

QS Example: $N = 1147$, $M = 10$, $\lceil \sqrt{N} \rceil = 34$

For which $0 \le i \le 10$ is $((34+i)^2 \bmod N) \equiv 0 \pmod{2}$?

Need to know what $(34+i)^2 \pmod{N}$ is.
Key We show $1147 < (34+i)^2 \le 2 \times 1147$ and hence

$$(34+i)^2 \pmod{N} = (34+i)^2 - 1147$$

$(34+i)^2$ is min: $i = 0$. Its $34^2 = 1156 > 1147$.
$(34+i)^2$ is max:$i = 10$. Its $44^2 = 1936 < 2 \times 1147$.

$(34+i)^2 \bmod 1147 = (34+i)^2 - 1147 \equiv i^2 - 1 \pmod{2}$.
$i^2 - 1 \equiv 0 \pmod{2}$ if $i \equiv 1 \pmod{2}$.

QS Example: $N = 1147$, $M = 10$, $\lceil \sqrt{N} \rceil = 34$

For which $0 \le i \le 10$ is $((34+i)^2 \bmod N) \equiv 0 \pmod 2$?

Need to know what $(34+i)^2 \pmod N$ is.
Key We show $1147 < (34+i)^2 \le 2 \times 1147$ and hence

$$(34+i)^2 \pmod N = (34+i)^2 - 1147$$

$(34+i)^2$ is min: $i = 0$. Its $34^2 = 1156 > 1147$.
$(34+i)^2$ is max:$i = 10$. Its $44^2 = 1936 < 2 \times 1147$.

$(34+i)^2 \bmod 1147 = (34+i)^2 - 1147 \equiv i^2 - 1 \pmod 2$.

$i^2 - 1 \equiv 0 \pmod 2$ if $i \equiv 1 \pmod 2$.

Can do similar for any prime p.

Quad Sieve Alg

Given N let $x = \lceil \sqrt{N} \rceil$. All $\equiv$ are mod N. B, M are params.

B-factor $(x+0)^2 \pmod{N}, \ldots, (x+M)^2 \pmod{N}$ by Quad S.

Let $I \subseteq \{0, \ldots, M\}$ so that $(\forall i \in I)$, y_i is B-factored. Find $J \subseteq I$ such that $\sum_{i \in J} \vec{v}_i = \vec{0}$. Hence $\prod_{i \in J} y_i$ has all even exponents, so there exists Y

$$\prod_{i \in J} y_i = Y^2$$

$$(\prod_{i \in J} (x+i))^2 \equiv \prod_{i \in J} y_i = Y^2 \pmod{N}$$

Let $X = \prod_{i \in J}(x+i) \pmod{N}$ and $Y = \prod_{i \in J} q_i^{e_i} \pmod{N}$.

$$X^2 - Y^2 \equiv 0 \pmod{N}.$$

$GCD(X - Y, N)$, $GCD(X + Y, N)$ should yield factors.

Secret Sharing

Threshold Secret Sharing

Zelda has a secret $s \in \{0,1\}^n$.

Def: Let $1 \le t \le m$. (t,L)-secret sharing is a way for Zelda to give strings to $A_1,\ldots,A_L$ such that:

1. If any t get together than they can learn the secret.
2. If any $t-1$ get together they cannot learn the secret.

Threshold Secret Sharing Caveats

Cannot learn the secret. Two flavors:

1. info-theoretic
2. computational.

Note: Access Structure is a set of sets of students closed under superset. Can also look at Secret Sharing with other access structures.

Methods For Secret Sharing

Assume $|s| = n$.

1. Random String Method.
   PRO: Can be used for ANY access structure.
   CON: For Threshold Zelda may have to give Alice LOTS of strings
2. Poly Method. Uses: t points det poly of deg $t-1$.
   PRO: Zelda gives Alice a share of exactly n. Simple.
   CON: Only used for threshold secret sharing
   CAVEAT: For exactly n need fields. Get $n+1$ with mod p.

Short Shares

If demand Info-theoretic security then shares have to be $\geq |s|$.

We did that in class.

So we go to comp theoretic, next slide.

Short Shares

Thm: Assume there exists an α-SES. Assume that for message of length n, it is secure. Then, for all $1 \leq t \leq L$ there is a (t, L)-scheme for $|s| = n$ where each share is of size $\frac{n}{t} + \alpha n$.

Short Shares

Thm: Assume there exists an α-SES. Assume that for message of length n, it is secure. Then, for all $1 \leq t \leq L$ there is a (t, L)-scheme for $|s| = n$ where each share is of size $\frac{n}{t} + \alpha n$.

1. Zelda does $k \leftarrow GEN(n)$. Note $|k| = \alpha n$.

Short Shares

Thm: Assume there exists an α-SES. Assume that for message of length n, it is secure. Then, for all $1 \le t \le L$ there is a (t, L)-scheme for $|s| = n$ where each share is of size $\frac{n}{t} + \alpha n$.

1. Zelda does $k \leftarrow GEN(n)$. Note $|k| = \alpha n$.
2. $u = ENC_k(s)$. Let $u = u_0 \cdots u_{t-1}$, $|u_i| \sim \frac{n}{t}$.

Short Shares

Thm: Assume there exists an α-SES. Assume that for message of length n, it is secure. Then, for all $1 \le t \le L$ there is a (t, L)-scheme for $|s| = n$ where each share is of size $\frac{n}{t} + \alpha n$.

1. Zelda does $k \leftarrow GEN(n)$. Note $|k| = \alpha n$.
2. $u = ENC_k(s)$. Let $u = u_0 \cdots u_{t-1}$, $|u_i| \sim \frac{n}{t}$.
3. Let $p \sim 2^{n/t}$. Zelda forms poly over $\mathbb{Z}_p$:

$$f(x) = u_{t-1}x^{t-1} + \cdots + u_1 x + u_0$$

Short Shares

Thm: Assume there exists an α-SES. Assume that for message of length n, it is secure. Then, for all $1 \le t \le L$ there is a (t, L)-scheme for $|s| = n$ where each share is of size $\frac{n}{t} + \alpha n$.

1. Zelda does $k \leftarrow GEN(n)$. Note $|k| = \alpha n$.
2. $u = ENC_k(s)$. Let $u = u_0 \cdots u_{t-1}$, $|u_i| \sim \frac{n}{t}$.
3. Let $p \sim 2^{n/t}$. Zelda forms poly over $\mathbb{Z}_p$:

$$f(x) = u_{t-1}x^{t-1} + \cdots + u_1 x + u_0$$

4. Let $q \sim 2^{\alpha n}$. Zelda forms poly over $\mathbb{Z}_q$ by choosing $r_{t-1}, \ldots, r_1 \in \{0, \ldots, q-1\}$ at random and then:

$$g(x) = r_{t-1}x^{t-1} + \cdots + r_1 x + k$$

Short Shares

Thm: Assume there exists an α-SES. Assume that for message of length n, it is secure. Then, for all $1 \le t \le L$ there is a (t, L)-scheme for $|s| = n$ where each share is of size $\frac{n}{t} + \alpha n$.

1. Zelda does $k \leftarrow GEN(n)$. Note $|k| = \alpha n$.
2. $u = ENC_k(s)$. Let $u = u_0 \cdots u_{t-1}$, $|u_i| \sim \frac{n}{t}$.
3. Let $p \sim 2^{n/t}$. Zelda forms poly over $\mathbb{Z}_p$:

$$f(x) = u_{t-1}x^{t-1} + \cdots + u_1 x + u_0$$

4. Let $q \sim 2^{\alpha n}$. Zelda forms poly over $\mathbb{Z}_q$ by choosing $r_{t-1}, \ldots, r_1 \in \{0, \ldots, q-1\}$ at random and then:

$$g(x) = r_{t-1}x^{t-1} + \cdots + r_1 x + k$$

5. Zelda gives A_i, $(f(i), g(i))$. Length: $\sim \frac{n}{t} + \alpha n$.

Verifiable Secret Sharing VSS

Cannot do it if demand info-theoretic security.
That was a HW.
So we go to comp theoretic, next slide.

Verifiable Secret Sharing

1. Secret is s, $|s| = n$. Zelda finds $p \sim n$.
2. Zelda finds a generator g for $\mathbb{Z}_p$.
3. Zelda picks rand $r_{t-1}, \ldots, r_1$,
   $f(x) = r_{t-1}x^{t-1} + \cdots + r_1 x + s$.
4. For $1 \le i \le L$ Zelda gives A_i $f(i)$.
5. Zelda gives to EVERYONE the values $g^{r_1}, \ldots,$
   $g^{r_{t-1}}, g^s, g$.
   (We think discrete log is HARD so r_i not revealed.)

Recover: The usual – any group of t can blah blah.

Verify: A_i reveals $f(i) = 17$. Group computes:

1) g^{17}.

2) $(g^{r_{t-1}})^{i^{t-1}} \times (g^{r_{t-2}})^{i^{t-2}} \times \cdots (g^{r_1})^{i^1} \times (g^s)^{i^0} = g^{f(i)}$

If this is g^{17} then A_i is truthful. If not then A_i is dirty stinking liar.

Alice and Bob and Love

The Problem

1. Alice has bit a, Bob has bit b, and they want to compute $a \wedge b$. They have a many decks of cards. At the end of the protocol:
    1.1 They both know $a \wedge b$.
    1.2 If $a = 0$ then A does not know b.
    1.3 If $b = 0$ then B does not know a.
    1.4 If $a = 1$ then since A knows a and $a \wedge b$, A knows b.
    1.5 If $b = 1$ then since B knows b and $a \wedge b$, B knows a.
2. Alice, Bob, Cards, and Love is Fair Game for the final. For example, I could ask you to extend to $a \wedge b \wedge c$.

The 3-Card Solution by Karun Singh

All cards are face down.

1. The cards ♣♣♥ are on the table.

The 3-Card Solution by Karun Singh

All cards are face down.

1. The cards ♣♣♥ are on the table.
2. Bob is not in the room.
   A-YES: Switch cards 2&3. A-NO: No switch.

The 3-Card Solution by Karun Singh

All cards are face down.

1. The cards ♣♣♥ are on the table.
2. Bob is not in the room.
   A-YES: Switch cards 2&3. A-NO: No switch.
3. Alice is not in the room.
   B-YES: Switch cards 1 and 2. B-NO: No switch.

The 3-Card Solution by Karun Singh

All cards are face down.

1. The cards ♣♣♥ are on the table.
2. Bob is not in the room.
   A-YES: Switch cards 2&3. A-NO: No switch.
3. Alice is not in the room.
   B-YES: Switch cards 1 and 2. B-NO: No switch.
4. Not done yet, but let's see what we got.

A	B	After A	After B
Y	Y	♣♥♣	♥♣♣
Y	N	♣♥♣	♣♥♣
N	Y	♣♣♥	♣♣♥
N	N	♣♣♥	♣♣♥

The 3-Card Solution by Singh, cont

The cards are face down.

A	B	After A	After B
Y	Y	♣♥♣	♥♣♣
Y	N	♣♥♣	♣♥♣
N	Y	♣♣♥	♣♣♥
N	N	♣♣♥	♣♣♥

Just reveal the first card:

- If it's ♥ then 2nd date!
- If not then no 2nd date!

Good Luck on the Exam

Good Luck on the Exam!