Cryptography

Five or six weeks later, she asked me if I had deciphered the manuscript… I told her that I had.

"Without the key, sir, excuse me if I believe the thing impossible."

"Do you wish me to name your key, madame?"

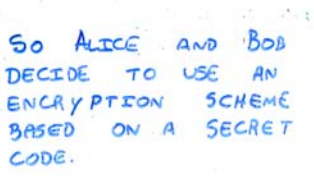
"If you please."

I then told her the key-word which belonged to no language, and I saw her surprise. She told me it was impossible, for she believed herself the only possessor of that word which she kept in her memory and which she had never written down.

I should have told her the truth -- that the same calculation which had served me for deciphering the manuscript had enabled me to learn the word -- but on a caprice it struck me to tell her that a genie had revealed it to me. This false disclosure fettered Madame d'Urfe to me. That day I became the master of her soul, and I abused my power.

From the autobiography
of Casanova (1757)

Most of slides in today's lecture courtesy of
Stephen Rudich and Martin Tompa



THE TROUBLE IS THAT EVE IS THE WORLD'S MOST BRILLIANT CRYPTANALYST! IF THERE IS A CLEVER WAY TO BREAK THE CODE, SHE WILL ALWAYS FIND IT.

① WHAT IS ALICE SAYING?

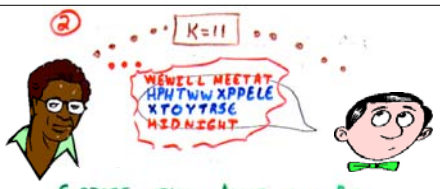
SUBTRACT 3 (MODULO 26) FROM EACH LETTER TO READ THE MESSAGE.

A SHIFT CIPHER TAKES A SECRET $1 \le K \le 25$ AND SHIFTS EVERY LETTER OF THE MESSAGE BY K. (WITH WRAPAROUND)

TO READ THE ENCODED MESSAGE THE RECEIVER SHIFTS THE MESSAGE BY -K.

EX: K=3 IS THE CAESAR CIPHER.

② K=11



SUPPOSE THAT ALICE AND BOB ARE KNOWN TO USE A SHIFT CIPHER.

WHAT IS BOB SAYING?

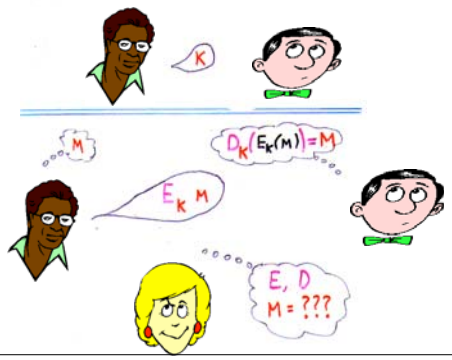
A CRYPTOSYSTEM IS A 5 TUPLE $(\mathcal{P}, \mathcal{C}, \mathcal{K}, E, D)$

1. $\mathcal{P}$ IS A FINITE SET OF POSSIBLE PLAINTEXTS.
2. $\mathcal{C}$ IS A FINITE SET OF POSSIBLE CIPHERTEXTS.
3. $\mathcal{K}$ IS THE KEYSPACE, A FINITE SET OF POSSIBLE KEYS.
4. $E: \mathcal{K} \times \mathcal{P} \to \mathcal{C}$ AND $D: \mathcal{K} \times \mathcal{C} \to \mathcal{P}$ ARE ENCRYPTION AND DECRYPTION FUNCTIONS.

[WE WRITE $E_K(P)$ TO MEAN $E(K,P)$]
[WE WRITE $D_K(C)$ TO MEAN $D(K,C)$]

5. $\forall K \in \mathcal{K}\ \forall P \in \mathcal{P}$
$D_K(E_K(P)) = P$

PICTURE OF CRYPTOSYSTEM (E,D) IN ACTION:



EXAMPLE. SHIFT CIPHERS

$(\mathcal{P}, \mathcal{C}, \mathcal{K}, E, D)$

$\mathcal{P} = \mathcal{C} = [A..Z]^n$

$\mathcal{K} = [1..25]$

$\forall P = P_1 P_2 P_3 \cdots P_n \in \mathcal{P}$

$E_K(P) = (P_1+K)(P_2+K)(P_3+K)\cdots(P_n+K)$

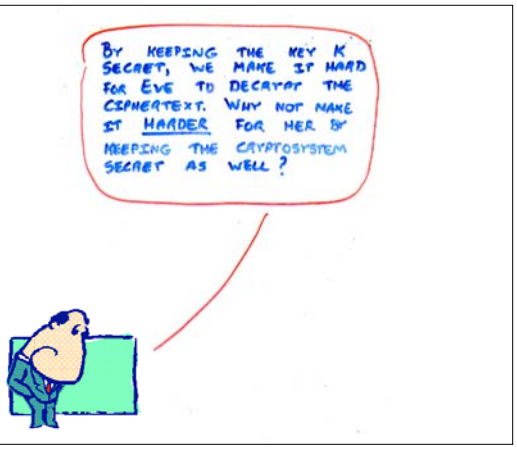
(+ IS MODULO 26)

$\forall C = C_1 C_2 \cdots C_2 \in \mathcal{C}$

$D_K(C) = (C_1-K)(C_2-K)\cdots(C_n-K)$

INDEED:

$D_K(E_K(P)) = P$



NONETHELESS.....

WHEN EVALUATING A PROPOSED CRYPTOSYSTEM WE WILL ALWAYS ASSUME THAT EVE KNOWS ITS COMPLETE DEFINITION BEFORE SHE TRIES TO BREAK IT.

THE ONLY THING SHE DOES NOT KNOW IS WHICH KEY $K \in \mathcal{K}$ IS BEING USED WITH THE ENCRYPTION AND DECRYPTION FUNCTIONS.

WHY DO WE TAKE THIS VIEWPOINT?

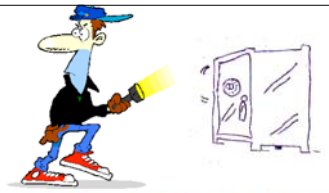
IT IS HARD TO KEEP A CRYPTOSYSTEM SECRET.

EX: GERMAN ENIGMA MACHINE FROM WORLD WAR II.

* THE ENIGMA CRYPTOSYSTEM WAS SUPPOSEDLY "UNBREAKABLE".
* NONETHELESS, THE GERMANS KEPT THE MACHINES UNDER TIGHT SECURITY TO MAKE IT HARDER.
* EARLY IN THE WAR, THE ALLIES CAPTURED AN ENIGMA MACHINE.
* ALAN M. TURING LEAD THE SUCCESSFUL EFFORT TO BREAK THE ENIGMA CRYPTOSYSTEM.

OTHER REASONS IT IS HARD TO KEEP A CRYPTOSYSTEM SECRET.

* SPIES
* TRAITORS
* ANY CRYPTOSYSTEM IMPLEMENTED IN SOFTWARE CAN BE EXAMINED.
* SIMULTANEOUS INVENTION. SOMEBODY ELSE MIGHT HAVE THOUGHT OF IT.



IS THERE A CRYPTOSYSTEM THAT EVE CAN'T BREAK?

IT MAY WELL BE DOUBTED WHETHER HUMAN INGENUITY CAN CONSTRUCT AN ENIGMA OF THIS KIND WHICH HUMAN INGENUITY MAY NOT, BY PROPER APPLICATION, RESOLVE.

EDGAR ALLEN POE

CRYPTOLOGY SEEKS TO FIND:

- AN UNBREAKABLE CRYPTOSYSTEM
- A RIGOROUS THEORY TO JUSTIFY OUR BELIEF IN THE SECURITY OF THE CRYPTOSYSTEM.

NOTE: MODERN CRYPTOLOGY SEEKS A GENERAL THEORY FOR UNDERSTANDING PRIVACY AND SECURITY IN A MUCH WIDER VARIETY OF CONTEXTS.

CRYPTO

IF EDGAR ALLEN POE IS CORRECT, THEN CRYPTOLOGY WILL NEVER FIND WHAT IT IS SEEKING.

QUESTION: IS THE SHIFT CIPHER AN UNBREAKABLE CRYPTOSYSTEM.

NO.

THERE ARE ONLY 25 POSSIBLE KEYS. EVE CAN TRY THEM ALL.

FOR A REASONABLE LENGTH CIPHERTEXT ONLY ONE OF THE 25 POSSIBLE DECRYPTIONS WILL MAKE ANY SENSE.

ANY CRYPTOSYSTEM WITH A SMALL KEYSPACE CAN BE BROKEN BY A BRUTE FORCE SEARCH OF ALL KEYS.

- MAKE LIST OF ALL POSSIBLE DECRYPTIONS OF CIPHERTEXT

  $D_{K_1}(c)$; $D_{K_2}(c)$; $D_{K_3}(c)$; .....

- FOR LONG ENOUGH MESSAGES ONLY ONE DECRYPTION ON THE LIST WILL MAKE SENSE.

THE SUBSTITUTION CIPHER.

(EX: CRYPTOGRAMS IN NEWSPAPERS)

$P = C = [A..Z]^n$

$\mathbb{K} = \{\pi \mid \pi : [A..Z] \to [A..Z]$ IS A PERMUTATION $\}$

$\forall \pi \in \mathbb{K}$

$E_\pi(P_1 P_2 \cdots P_n) = \pi(P_1)\pi(P_2)\cdots\pi(P_n)$

$D_\pi(C_1 \cdots C_n) = \pi^{-1}(C_1)\pi^{-1}(C_2)\cdots\pi^{-1}(C_n)$

SIZE OF KEYSPACE $|\mathbb{K}| = 26! > 4 \times 10^{26}$

IS QUITE LARGE. TOO LARGE FOR BRUTE FORCE.

THE SUBSTITUTION CIPHER CAN BE BROKEN BY LOOKING AT FREQUENCIES OF INDIVIDUAL LETTERS, DIGRAMS, AND TRIGRAMS; FOR THE CIPHERTEXT AND FOR THE ENGLISH LANGUAGE.

ONE-TIME PAD

$P = C = \{A, B, C, \cdots, Z\}^n$

$\mathbb{K} = \{A, B, C, \cdots, Z\}^n$

MESSAGE $M \in P$ IS n LETTERS LONG:

$m_1\ m_2\ m_3 \cdots\cdots m_n$

SECRET-KEY K OR "PAD" OF LENGTH n:

$K_1\ K_2\ K_3 \cdots\cdots K_n$

CIPHERTEXT IS THE "PADDED" MESSAGE:

$m_1 + K_1 \bmod 26 \quad m_2 + K_2 \bmod 26 \cdots$

$\cdots\cdots m_n + K_n \bmod 26$

ONE-TIME PAD (BINARY)

$P = C = \{0,1\}^n$

$K = \{0,1\}^n$

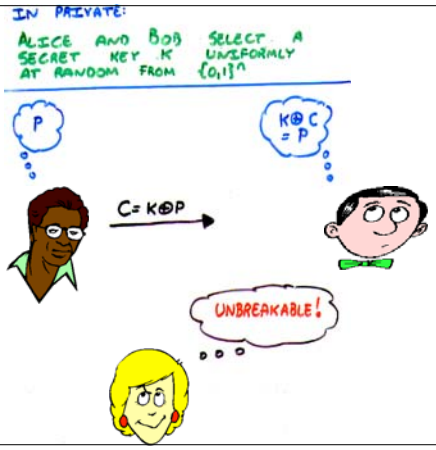
MESSAGE M IS n BITS LONG:
$m_1\ m_2\ m_3 \ldots m_n$

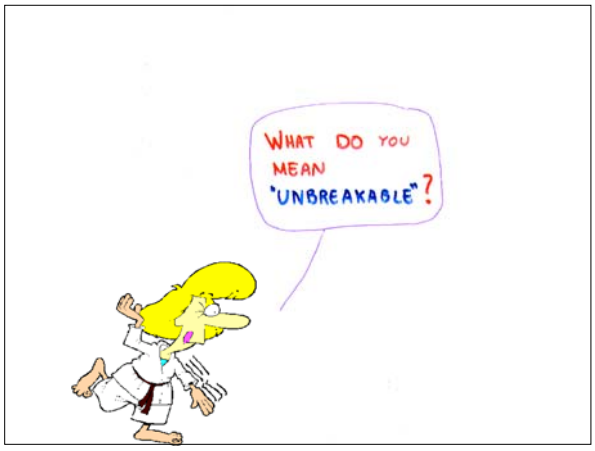
SECRET-KEY K OR "PAD" IS n BITS LONG:
$k_1\ k_2\ k_3 \ldots k_n$

CIPHERTEXT IS THE BITWISE XOR M⊕K:
$m_1 \oplus k_1\ \ m_2 \oplus k_2 \ldots m_n \oplus k_n$

$E_K(M) = M \oplus K \qquad D_K(C) = C \oplus K$

$D_K(E_K(M)) = D_K(M \oplus K) = (M \oplus K) \oplus K$
$= M$





NOTICE:

FIX P. VARY OVER ALL POSSIBLE K.

K⊕P WILL VARY OVER ALL POSSIBLE STRINGS.

EXAMPLE: P = 101

101 ⊕ 000 = 101	101 ⊕ 001 = 100	101 ⊕ 010 = 111	101 ⊕ 011 = 110
101 ⊕ 100 = 001	101 ⊕ 101 = 000	101 ⊕ 110 = 011	101 ⊕ 111 = 010

DEFINITION:

A CRYPTOSYSTEM HAS **PERFECT SECRECY**

IF FOR ALL POSSIBLE MESSAGES $M \in \mathcal{P}$,

$E_K(M)$ IS EQUALLY LIKELY TO BE ANY CIPHERTEXT IN $\mathcal{C}$ WHEN K IS CHOSEN AT RANDOM.

SINCE EACH POSSIBLE CIPHERTEXT C IS EQUALLY LIKELY TO RESULT FROM THE ENCRYPTION OF AN ARBITRARY MESSAGE, EACH MESSAGE IS EQUALLY LIKELY TO HAVE GENERATED C.

THUS C REVEALS NO INFORMATION ABOUT THE MESSAGE.

THEOREM:

A ONE-TIME PAD HAS PERFECT SECRECY.

- EACH MESSAGE IS EQUALLY LIKELY TO GENERATE EACH CIPHERTEXT:

  EACH CIPHERTEXT WILL GET GENERATED ONCE AS WE VARY OVER ALL KEYS.

  FOR ALL M,

  $E_{K_1}(M);\ E_{K_2}(M);\ \cdots\ ;\ E_{K_{2^n}}(M)$

  HAS ALL n-BIT SEQUENCES ONE THE LIST.

QUESTION:

DOES A SUBSTITUTION CIPHER HAVE PERFECT SECRECY?

NO.

DAD AND DOG

WILL NEVER GENERATE THE SAME CIPHERTEXT.

QUESTION:

CAN EVE BREAK A ONE-TIME PAD BY TRYING ALL KEYS AND CHOOSING THE DECRYPTION THAT MAKES SENSE? (LET'S GRANT EVE THE TIME TO MAKE THE LIST.)

NO.

THE LIST OF POSSIBLE DECRYPTIONS WILL CONTAIN ALL 2^n n-BIT STRINGS.

ALL THE SENSICAL STRINGS WILL BE ON THE LIST OF CANDIDATES.

SHANON [1949]

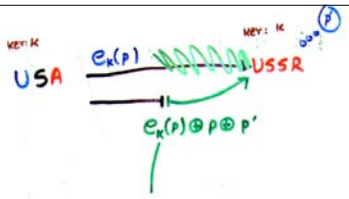
A CRYPTOSYSTEM HAS PERFECT SECRECY IF $\forall x, y \in \Sigma^*$ THE A POSTERIORI PROBABILITY THAT THE PLAINTEXT IS X, GIVEN THAT THE CYPHERTEXT Y IS OBSERVED, IS IDENTICAL TO THE A PRIORI PROBABILITY THAT THE PLAINTEXT IS X.

SHANON:
A ONE-TIME PAD HAS PERFECT SECRECY.

PROBLEM WITH ONE-TIME PAD:

KEY DISTRIBUTION IS EXPENSIVE.

WE NEED MORE MILEAGE OUT OF THE BITS WE TAKE THE TROUBLE TO DISTRIBUTE.



KNOWS p, WANTS TO SUBSTITUTE p'

MORAL:

A SCHEME CAN BE PERFECTLY SECURE AGAINST ONE KIND OF ATTACK WHILE BEING TREMENDOUSLY VULNERABLE TO ANOTHER.

TYPES OF ATTACK THAT ARE COMMONLY STUDIED IN CRYPTOGRAPHY:

"PASSIVE" ATTACKS

- KNOWN CYPHERTEXT
- KNOWN PLAINTEXT
- CHOSEN PLAINTEXT
- CHOSEN CYPHERTEXT

"ACTIVE" ATTACKS:

- DELETING, ADDING, MODIFYING MESSAGES

THEOREM: IF $|P| > |K|$, i.e., THE NUMBER OF POSSIBLE PLAINTEXTS EXCEEDS THE THE NUMBER OF POSSIBLE KEYS, THEN THERE CAN NEVER BE PERFECT SECRECY.

SUPPOSE EVE SEES CYPHERTEXT C.

FOR EACH POSSIBLE KEY K SHE CAN LIST A POSSIBLE PLAINTEXT THAT COULD HAVE ENCRYPTED TO C.

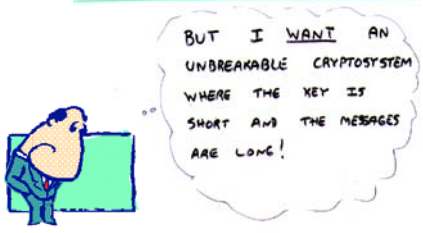
BECAUSE $|P|>|K|$, BY THE PIGEONHOLE PRINCIPLE THERE WILL BE AT ONE PLAINTEXT X THAT CAN'T EVER ENCRYPT TO C.

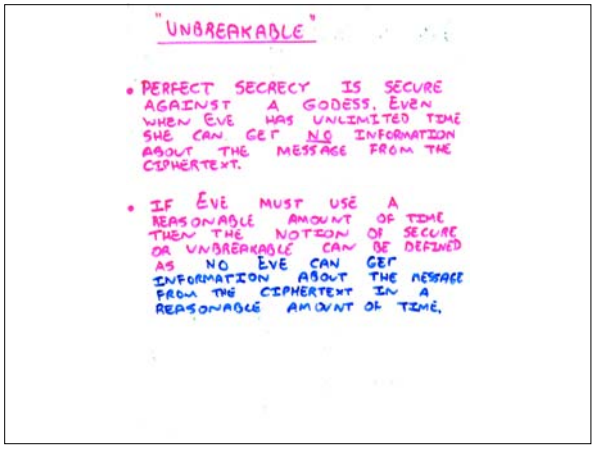
THUS EVE CAN BE SURE THAT X WAS NOT THE MESSAGE.

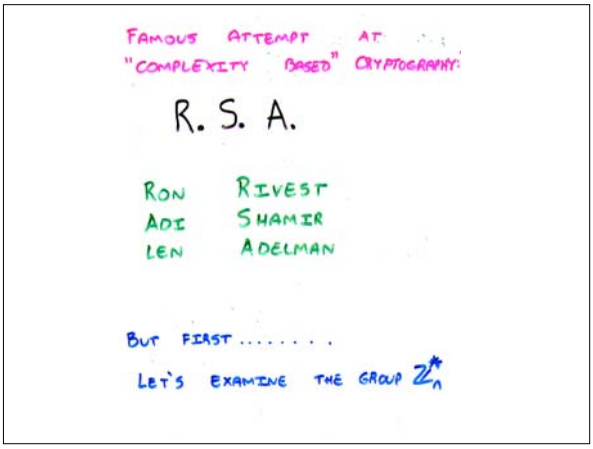
THUS,

WHEN YOU ARE WILLING TO USE AS MANY KEY BITS AS PLAINTEXT BITS, YOU CAN AFFORD A ONE-TIME PAD.

OTHERWISE, $|P|<|K|$ AND SO PERFECT SECRECY IS NOT POSSIBLE.







Receiver's Set-Up

- Choose 500-digit primes p and q (each 2 more than a multiple of 3).

  $p = 5, q = 11$

Receiver's Set-Up

- Choose 500-digit primes p and q (each 2 more than a multiple of 3).

  $p = 5, q = 11$
- Let $n = pq$.

  $n = 55$

Receiver's Set-Up

- Choose 500-digit primes p and q (each 2 more than a multiple of 3).
  $p = 5, q = 11$
- Let $n = pq$.
  $n = 55$
- Let $s = (1/3)\,(2(p - 1)(q - 1) + 1)$.
  $s = (1/3)\,(2 \cdot 4 \cdot 10 + 1) = 27$

Receiver's Set-Up

- Choose 500-digit primes p and q (each 2 more than a multiple of 3).
  $p = 5, q = 11$
- Let $n = pq$.
  $n = 55$
- Let $s = (1/3)\,(2(p - 1)(q - 1) + 1)$.
  $s = (1/3)\,(2 \cdot 4 \cdot 10 + 1) = 27$
- Publish n.
  Keep p, q, and s secret.

Encrypting a Message

- Break the message into chunks.
  H I C H R I S …

Encrypting a Message

- Break the message into chunks.
  H I C H R I S …

Encrypting a Message

- Break the message into chunks.
  H I C H R I S …
- Translate each chunk into an integer M ($0 \le M < n$) by any convenient method.
  8 9 3 8 18 9 19 …

Encrypting a Message

- Break the message into chunks.
  H I C H R I S …
- Translate each chunk into an integer M ($0 \le M < n$) by any convenient method.
  8 9 3 8 18 9 19 …
- Divide M^3 by n. $E(M)$ is the remainder.
  $M = 8$, $n = 55$
  $8^3 = 512 = 9 \times 55 + 17$
  $E(8) = 17$

Decrypting a Cyphertext C

- Divide C^s by n. $D(C)$ is the remainder.
  $C = 17$, $n = 55$, $s = 27$
  $17^{27} =$
  1,667,711,322,168,688,287,513,535,727,415,473
  =
  30,322,024,039,430,696,136,609,740,498,463 × 55 + 8
  $D(17) = 8$

Decrypting a Cyphertext C

- Divide C^s by n. $D(C)$ is the remainder.
  $C = 17$, $n = 55$, $s = 27$
  $17^{27} = 1,667,711,322,168,688,287,513,535,727,415,473$
  $= 30,322,024,039,430,696,136,609,740,498,463 \times 55 + 8$
  $D(17) = 8$
- Translate $D(C)$ into letters.
  H

Why Does It Work?

Euler's Theorem (1736): Suppose

- p and q are distinct primes,
- $n = pq$,
- $0 \leq M < n$, and
- $k > 0$.

If $M^{k(p-1)(q-1)+1}$ is divided by n, the remainder is M.

Why Does It Work?

Euler's Theorem (1736): Suppose

- p and q are distinct primes,
- $n = pq$,
- $0 \leq M < n$, and
- $k > 0$.

If $M^{k(p-1)(q-1)+1}$ is divided by n, the remainder is M.

$$(M^3)^s = (M^3)^{(1/3)(2(p-1)(q-1)+1)}$$
$$= M^{2(p-1)(q-1)+1}$$

Leonhard Euler 1707-1783

… both Gauss and lesser mathematicians may be justified in rejoicing that there is one science [number theory] at any rate… whose very remoteness from ordinary human activities should keep it gentle and clean.

From the autobiography
A Mathematician's Apology
of G.H. Hardy,
number theorist and pacifist,
1940

Why Is It Secure?

- To find $M = D(C)$, you seem to need s.

Why Is It Secure?

- To find $M = D(C)$, you seem to need s.
- To find s, you seem to need p and q.

Why Is It Secure?

- To find $M = D(C)$, you seem to need s.
- To find s, you seem to need p and q.
- All the cryptanalyst has is $n = pq$.

Why Is It Secure?

- To find $M = D(C)$, you seem to need s.
- To find s, you seem to need p and q.
- All the cryptanalyst has is $n = pq$.
- How hard is it to factor a 1000-digit number n?
  With the grade school method,
  doing 1,000,000,000 steps per second
  it would take …

Why Is It Secure?

- To find $M = D(C)$, you seem to need s.
- To find s, you seem to need p and q.
- All the cryptanalyst has is $n = pq$.
- How hard is it to factor a 1000-digit number n?

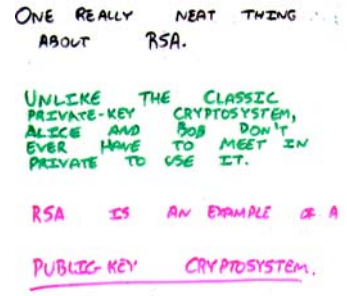
  With the grade school method,
  doing 1,000,000,000 steps per second
  it would take … 10^{483} years.

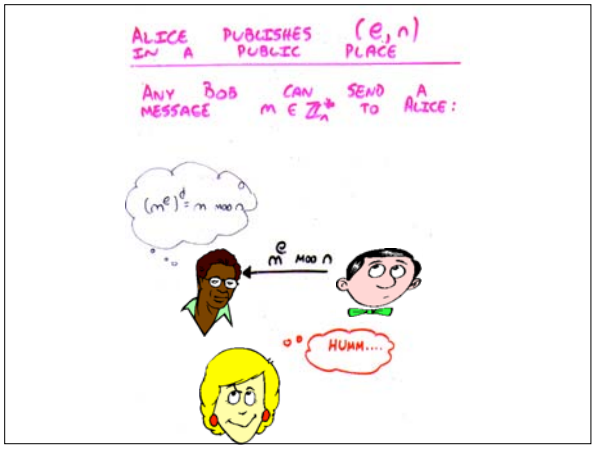
State of the Art in Factoring

- 1977: Inventors encrypt a challenge using "RSA129," a 129-digit number $n = pq$.
- 1981: Pomerance invents Quadratic Sieve factoring method.
- 1994: Using Quadratic Sieve, RSA129 is factored over 8 months using 1000 computers on the Internet around the world.
- (1999: Using a new method, RSA140 is factored.)

State of the Art in Factoring

- 1977: Inventors encrypt a challenge using "RSA129," a 129-digit number $n = pq$.
- 1981: Pomerance invents Quadratic Sieve factoring method.
- 1994: Using Quadratic Sieve, RSA129 is factored over 8 months using 1000 computers on the Internet around the world.
- (1999: Using a new method, RSA140 is factored.)
- Using Quadratic Sieve, a 250-digit number would take 800,000,000 months instead of 8.



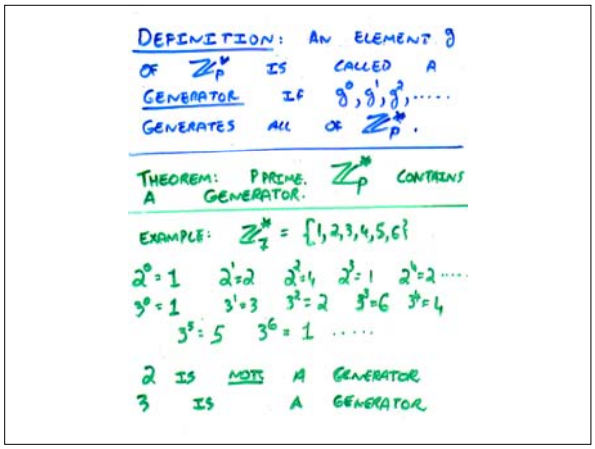


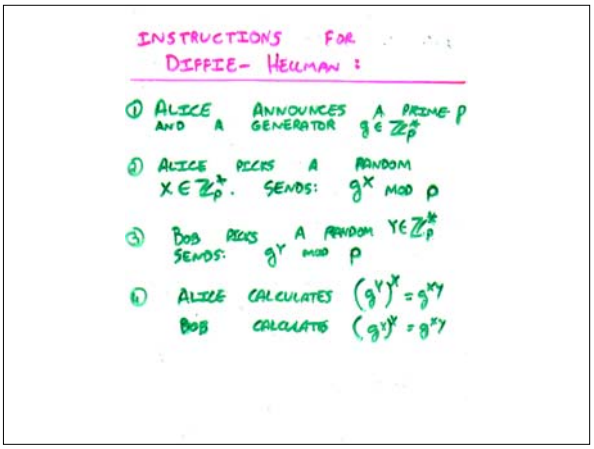
Public Key Cryptography

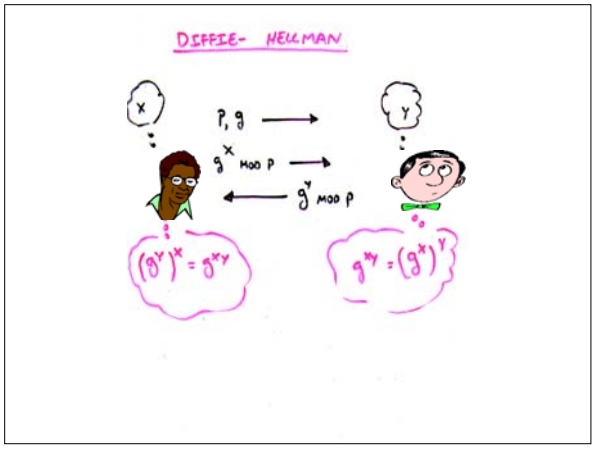
We stand today on the brink of a revolution in cryptography.

Diffie and Hellman, 1976

Another famous example: the first proposed public key cryptosystem.







Cool things we can do with RSA:

Unforgeable Signatures

Signed Messages

- How A sends a secret message to B

A B

$C = E_B(M)$ —C→

$M = D_B(C)$

Signed Messages

- How A sends a secret message to B

A B

$C = E_B(M)$ —C→

$M = D_B(C)$

- How A sends a signed message to B

A B

$C = D_A(M)$ —C→

$M = E_A(C)$

Signed *and* Secret Messages

- How A sends a secret message to B ...

A B

$C = E_B(M)$ —C→

$M = D_B(C)$

- How A sends a signed secret message to B ...

A B

$C = E_B(D_A(M))$ —C→

$M = E_A(D_B(C))$

Flipping a Coin Over the Phone

A B

Choose random x.

$y = E_A(x)$ —y→

Guess if x is even or odd.

"even" / "odd" ←

—x→

Check $y = E_A(x)$.

- B wins if the guess about x was right, or $y \neq E_A(x)$.

More cool things we can do with RSA:

Dating for shy people

Final Application -- Dating

- Alice and Bob want to figure out if they're interested in dating each other, but don't want to reveal it in case other one isn't interested.
- Use RSA if you don't have a trusted mutual friend that won't spill the beans
- First, some remarks:
  - The mere fact that they are carrying out this protocol might be seen as evidence that they are interested. Instead, imagine they are at a singles party where every pair of people has to carry out protocol.
  - If one player is interested and the other one isn't, can't avoid having the interested party learn that other isn't.
  - One player can always pretend to be interested and then say "ha ha".
- Can't ask crypto to solve problem of heartbreak or people being jerks.
- Just ensure that players can't learn if the other is interested without declaring their own interest.

Dating Protocol

- Alice picks 2 large primes such that p-1 and q-1 aren't divisible by 3, then sets N=pq. Sends Bob N, together with X= x^3 mod N and Y= y^3 mod N, where x = 0+ random garbage added, y = "whether she's interested" + random garbage
- Since RSA is secure, X and Y look random to Bob. He picks random r from 0 to N-1. If he's not interested in Alice, he sends her $x^3 r^3$ mod N. If he is interested he sends $y^3 r^3$ mod N
- Alice takes cube root of what Bob sent. Which will be either xr mod N or yr mod N, either way looks random, since she doesn't know r.
- Alice sends result back to Bob.
- Since Bob knows r, he divides it out. If he wasn't interested he gets x, which reveals nothing. If he was interested, he gets y.

Dating Protocol

- Example of "secure multiparty computation".
- Similar ideas can be used for example:
  - To help 2 people figure out who makes more money, without either of them learning anything else about each other's wealth
  - To help a group of people figure out how much money they have in total, without any individual revealing her own amount.

General Comments About Public-Key Cryptosystems

- Slow.
- Vulnerable to exhaustive search, and chosen-ciphertext attacks.

Hybrid Cryptosystems

- In practice, public-key crypto used to secure and distribute **session keys**, which are then used with private-key crypto to secure message traffic.

- Bob sends Alice his public key.
- Alice generates random session key K, encrypts it using Bob's public key, and sends it to Bob.
- Bob decrypts Alice's message using his private key to recover session key.
- Both encrypt their communications using same session key.

Public-key crypto solves important key-management problem.

The Complexity Perspective

- Pseudorandom generators and CPRG
- One-way functions
- Trapdoor one-way functions
- NP-completeness and cryptography
- Zero-knowledge Proofs
- Impagliazzo's Five Worlds

The Complexity Perspective

- The existence of hard problems is usually viewed as a negative
- Bright side: hard problems can be put to work for us. This was the insight of Diffie and Hellman when they suggested complexity-based cryptography.
- RSA is based on intractability of factoring.
- More abstractly, using hard problems seems to require ability to generate lots of hard instances, which are difficult to invert.
- These are **one-way functions**.

Pseudorandomness

- A fresh view of randomness
- "Indistinguishable things are identical" -- Leibniz
- Pseudo-random generator (PRG): efficient deterministic procedure for stretching short random strings into long "random-looking" strings.
- Applications:
  - Cryptography
  - Derandomization e.g. P= BPP

Theories of Randomness

- Shannon: randomness represents lack of information -- modeled as probability distribution on possible values of missing data.
- Kolmogorov-Chaitin: randomness represents lack of structure -- represented by length of most succinct and effective description of object.
- Rooted in computability theory: measures randomness in terms of the shortest program that can generate the object. (not decidable)
- Modern complexity view: views randomness relative to observer's computational view. Objects are equal if they cannot be told apart by any efficient procedure.

Thought Experiment

Alice and Bob play game in one of 4 ways. In each, Alice flips an unbiased coin and Bob is asked to guess its outcome before the coin hits the floor. Alternatives differ by knowledge Bob has before making his guess.

1. Bob has to write down guess before Alice flips coin. Bob wins with probability 1/2.
2. Bob announces his guess while coin is spinning in air. Although outcome determined in principle by motion of coin, Bob doesn't have accurate information on motion. We believe Bob wins with prob 1/2.
3. Like 2, but Bob has sophisticated equipment capable of providing accurate info on coin motion as well as environment. However, Bob cannot process this info in time to improve his guess.
4. Bob's recording equipment is directly connected to a powerful computer programmed to solve motion equations and output prediction. Conceivable that Bob could substantially improve his guess at outcome.

Conclusion

Randomness is an event relative to information and computing resources at our disposal.

Even events fully determined by public information may be perceived as random by an observer lacking relevant info and/or ability to process it.

Our focus in complexity theory is on lack of sufficient processing power

Which may be due to either formidable amount of computation required for analyzing the event in question or to fact that observer is very limited.

Pseudo-random generators

A PRG is a

function: seed --> longer, seemingly random string

- Examples:

In most programming languages: start with x_0

- $x_1 = (a\ x_0 + b) \bmod N$
- $x_2 = (a\ x_1 + b) \bmod N$
- $x_3 = (a\ x_2 + b) \bmod N$

- Good for non-cryptographic applications, but adversary could easily distinguish sequence from random.
- "Anyone who considers arithmetical methods of producing random digits is, of course, in a state of sin." von Neumann
- "The generation of random numbers is too important to be left to chance." R. Coveyou

Cryptographic Pseudo-random generators

A CPRG is a

$f: \{0,1\}^n \rightarrow \{0,1\}^{r(n)}$ s.t.

- f is computable in polynomial time
- For all poly time algorithms A

  $|\Pr(A(y) \text{ accepts when } y \text{ is random } r(n) \text{ bit string}) - \Pr(A(f(x)) \text{ accepts when } x \text{ is random } n \text{ bit string})|$ is negligibly small.
- I.e. output looks random to any poly time algorithm.

Cryptographic Pseudo-random generators

A CPRG is a
f: $\{0,1\}^n$ --> $\{0,1\}^{r(n)}$
S.t.
- f is computable in polynomial time
- For all poly time algorithms A
  |Pr (A(y) accepts when y is random r(n) bits) - Pr (A(f(x)) accepts when x is random n bit string)| is negligibly small.
- I.e. output looks random to any poly time algorithm.

- Can use CPRG to get private-key cryptosystem with a small key.
- Can use CPRG for derandomization (when have right parameters)

One way functions

A **one way function** f is a function
f: $\{0,1\}^n$-> $\{0,1\}^{p(n)}$ such that
- f is computable in polynomial time
- For all poly time algorithms A
  Pr (A can find x' such that f(x') = f(x) when x is random n bit string) is negligible.

I.e. average-case hardness

Conjecture: One way functions exist.
- Theorem: Any CPRG is also a OWF.
- Theorem: If there are OWFs, there are CPRG.

For public key cryptography we need a **trapdoor one-way function**: There is some extra information (the trapdoor), such that with that information, the function f is easy to invert.

NP-completeness and crypto

- We don't know how to base cryptography on an NP-complete problem.
- We need average-case hardness rather than worst-case hardness.
- Another issue is that many problems in crypto belong to NP and coNP. Example: is the first bit of the plaintext 1?
- Problems in NP and coNP can't be NP-complete unless NP= coNP.

Zero-Knowledge Proofs

- Recall the protocol to show that 2 graphs are not isomorphic.
- Observation: Verifier was convinced without gaining any knowledge about two graphs. In particular, she learned nothing that enabled her to prove to anyone else that they're not isomorphic.
- In particular, if verifier trusted the prover, she could have simulated the entire interaction with the prover on her own, without ever involving the prover.
- Zero-knowledge proof system: prover only tells the verifier things she already knew. P is perfect ZK for L if for every V (probabilistic poly time), there is an A (probabilistic poly time) s.t. (P,V)(x) = A(x) for all x in L

Zero-Knowledge Proofs -- General Theorem

- If there are one way functions, then every language in NP has a zero knowledge proof.

ZK Proof Idea for 3 Coloring

- Prover picks 3 coloring and randomly permutes the colors in that 3-coloring.
- Prover writes the color of each vertex on a slip of paper and place it in magic box that's labelled with that vertex's number.
- Give magic boxes to verifier. Magic box = verifier can't open it.
- Verifier picks any 2 neighboring vertices, prover opens up the boxes for those 2 vertices. Verifier rejects if they aren't colored differently.
- Otherwise the verifier accepts.
- Repeat whole protocol.

- Verifier learns nothing, since colors permuted randomly and reshuffled each time.

- Magic boxes = encrypted messages.
- When verifier asks for boxes, prover decrypts the messages.

Impagliazzo's 5 possible worlds

- Algorithmica, Heuristica, Pessiland, Minicrypt and Cryptomania

- Possible worlds = There is an oracle relative to which this world exists = Not currently known to be false.

- Among other things, we'll consider impact on sad story of Professor Grouse, teacher of young Gauss who asked children to add up 1 +2 +… + 100. Professor Grouse became obsessed with getting his revenge by humiliating Gauss in front of the whole class, by inventing problems Gauss could not solve. In real life, this led to Grouse being committed and to Gauss developing a life-long interest in number-theoretic algorithms.
- How would story have ended if Grouse was an expert in complexity?

Algorithmica

- World in which P = NP
- Almost any type of optimization problem would become easy and automatic, e.g.
  - VLSI design would no longer use heuristics -- could produce optimal layouts.
  - Inductive learning systems would work well.
  - Could produce Mr. Spock-like estimates for all sorts of complicated events.
  - "Computer assisted mathematics" would be redundant, since computers could find proofs for any theorem in time roughly the length of the proof.
  - Capacity of computers will become that currently depicted in science fiction.
- No security: no way to allow some people access to information without making it available to everyone. Any means of identification would have to be based on unforgeable physical measurement.
- Grouse would no success at stumping Gauss, since he would need to give Gauss a problem that Grouse could later present an answer to the class for ==> could only present problems with succinct, easily verifiable solutions, I.e. NP.
- But since P=NP, Gauss could solve those problems easily.

Heuristica

- World where NP problems are intractable in worst-case, but tractable on average.
- There exist hard instances of NP problems, but to find such hard instances is itself intractable.
- Grouse might be able to find problems Gauss cannot answer, but it might take Grouse a year to find a problem that Gauss couldn't solve in a day, or 10 years to find a problem that Gauss couldn't solve in a month (Gauss has a polynomial advantage over Grouse since he is a genius.)
- Many practical optimization problems would become easy and automatic.
- Still no security: eavesdroppers would be able to solve problems in about the same amount of time that it would take legitimate users to think up problems to uniquely identify them.

Pessiland

- No one way functions.
- Easy to generate hard instances of NP-complete problems, but no way to efficiently generate hard *solved* instances.
- In Pessiland, Grouse could pose to Gauss problems that he couldn't solve. But Grouse couldn't solve them either, so Gauss's humiliation would not be so great.
- Problems for many domains will have no easy solutions.
- There does not seem to be a way to make use of the hard problems in Pessiland in cryptography.
- Arguably the worst of all possible worlds.

Minicrypt

- There are one-way functions (so we can do private key crypto) but no trapdoor one-way functions (so no public key crypto).
- One way function could be used to generate hard, solved problems: generator would pick x, compute y=f(x) and pose the search problem -- Find any x' with f(x') = y.
- Therefore Grouse can best Gauss in front of the class.
- No positive algorithmic aspects, but can get pseudorandom generators that can be used to derandomize algorithms.

Cryptomania

- Public key cryptography is possible -- two parties can agree on a secret message using only publicly accessible channels.
- Gauss is humiliated -- using conversations in class, Grouse and his pet student could jointly choose a problem that they would both know the answer to, but which Gauss could not solve. In fact, Grouse could arrange that all the students but Gauss would be able to solve all problems in class!
- Almost all cryptographic tasks imaginable can be done.
- This is where we think we live right now.
- But we could be wrong!!!
  - Public key sizes keep growing
  - Number-theoretic problems that are the basis for public key crypto are solvable in poly time on a quantum computer!

Impagliazzo's 5 worlds

- Algorithmica: P=NP or at least fast probabilistic algorithms exist to solve all NP problems.
- Heuristica: P is not = NP, but while NP problems are hard in the worst case, they are easy on average.
- Pessiland: NP-complete problems are hard on average, but one-way functions don't exist, hence no cryptography
- Minicrypt: One way functions exist, but trapdoor one-way functions don't exist. Hence private-key crypto, PRG, etc, but no public-key crypto
- Cryptomania: Public-key crypto exists -- there are trapdoor one-way functions.

- Reigning belief: we live in Cryptomania or at very least Minicrypt.