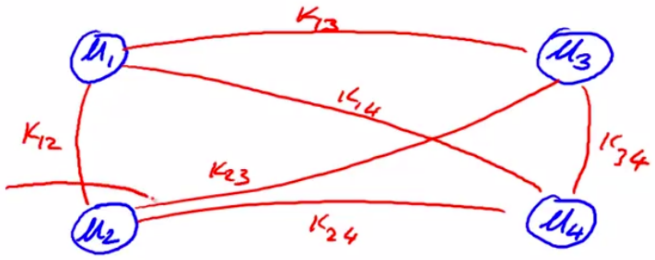
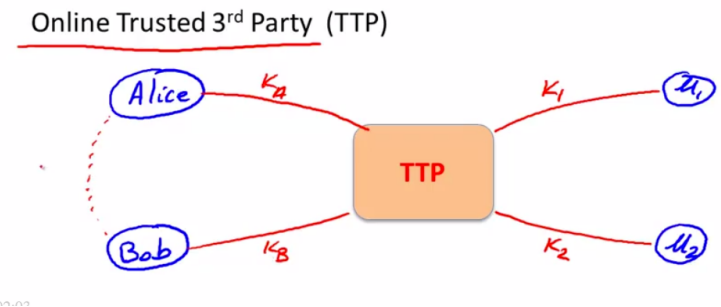
- **Basic Key Exchange 1: problem statement**
- **Trusted 3rd parties**
- Key management

Problem: n users. Storing mutual secret keys is difficult

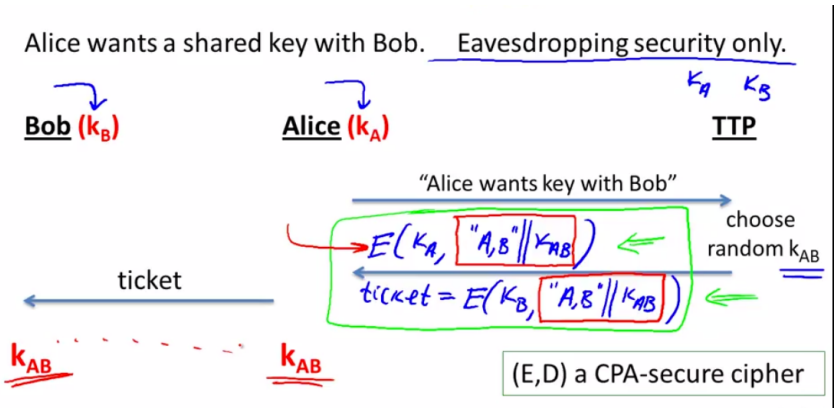


Total: O(n) keys per user

- A better solution



- Generating keys: a toy protocol



- Generating keys: a toy protocol

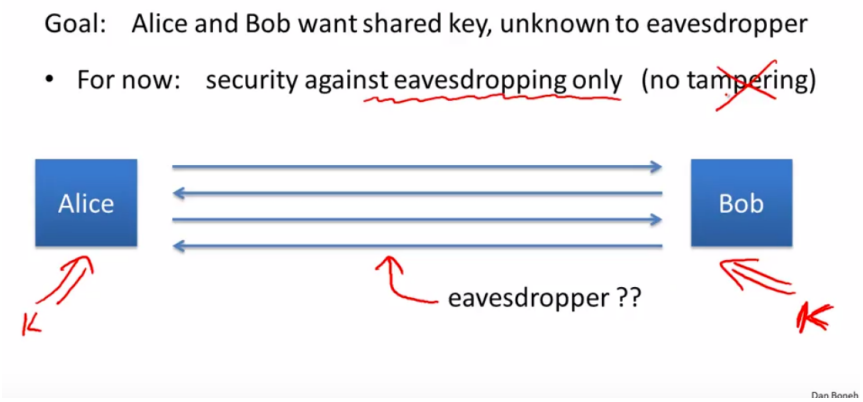
Alice wants a shared key with Bob. Eavesdropping security only.

Eavesdropper sees: $E(k_A, \text{"A, B"} \| k_{AB})$; $E(k_B, \text{"A, B"} \| k_{AB})$

(E,D) is CPA-secure ⇒
eavesdropper learns nothing about k_{AB}

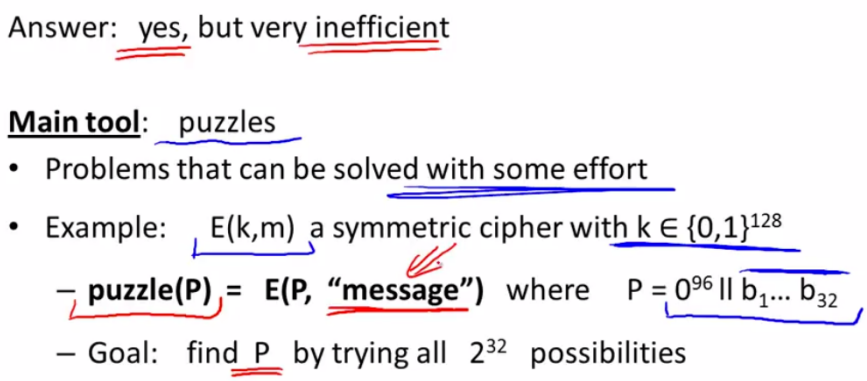
- Note: TTP needed for every key exchange, knows all session keys.
- TTP - trusted third party

- Toy protocol: insecure against active attacker
  - Example: insecure against replay attacks
    - attacker records session between alice and merchant bob
      - for example a book order
    - Attacker replays session to bob
      - bob thinks alice is ordering another copy of book
- Key question?
  - Can we generate shared keys without an online trusted 3rd party?
  - answer: yes!
  - starting point of public key cryptography
- **Merkle Puzzles**
- Key exchange without an online TTP?



- can this be done using generic symmetric crypto?

- Merkle Puzzles



- Merkle puzzles

Alice: prepare 2^{32} puzzles

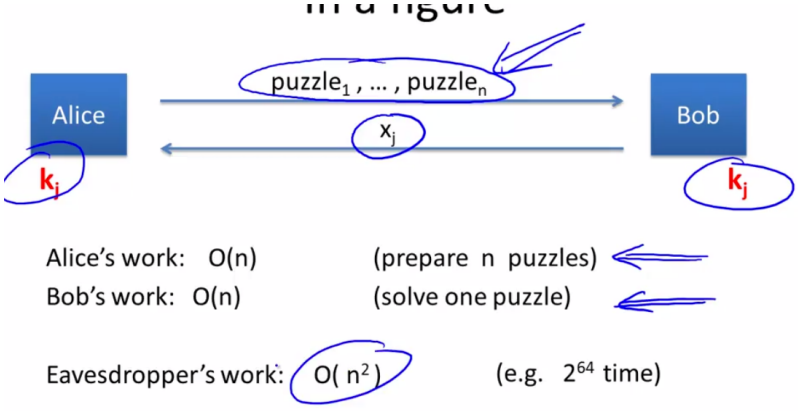
- For $i=1, \ldots, 2^{32}$ choose random $P_i \in \{0,1\}^{32}$ and $x_i, k_i \in \{0,1\}^{128}$
  set $puzzle_i \leftarrow E(0^{96} \| P_i, \text{"Puzzle \# } x_i \text{"} \| k_i)$
- Send $puzzle_1, \ldots, puzzle_{2^{32}}$ to Bob

Bob: choose a random $puzzle_j$ and solve it. Obtain (x_j, k_j).

- Send x_j to Alice

Alice: lookup puzzle with number x_j. Use k_j as shared secret

- In a figure



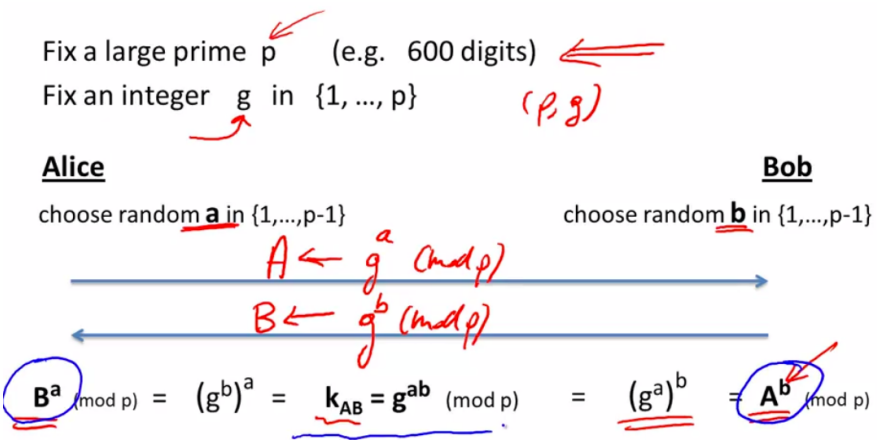
  - participants have to spend linear time attacker has to spend quadratic time
- Impossibility Result
  - Can we achieve a better gap using a general symmetric cipher?
  - answer: unknown
  - But: roughly speaking
    - quadratic gap is best possible if we treat cipher as a block box oracle
- **The Diffie-Hellman Protocol**
- Key exchange without an online TTP?

Goal: Alice and Bob want shared secret, unknown to eavesdropper

- For now: security against eavesdropping only (no tampering)

- can this be done with an exponential gap?

- The Diffie-Hellman protocol (informally)



- Security (much more on this later)

Eavesdropper sees: p, g, $A=g^a \pmod p$, and $B=g^b \pmod p$

Can she compute $g^{ab} \pmod p$??

More generally: define $DH_g(g^a, g^b) = g^{ab} \pmod p$

How hard is the DH function mod p?

- How hard is the DH function mod p?

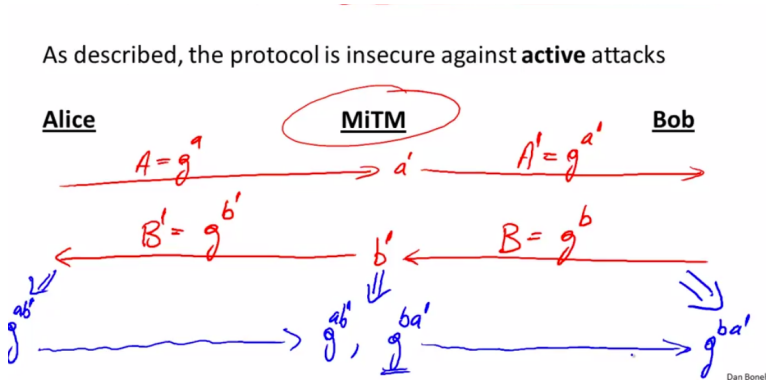
Suppose prime p is n bits long.

Best known algorithm (GNFS): run time $\exp(\tilde{O}(\sqrt[3]{n}))$

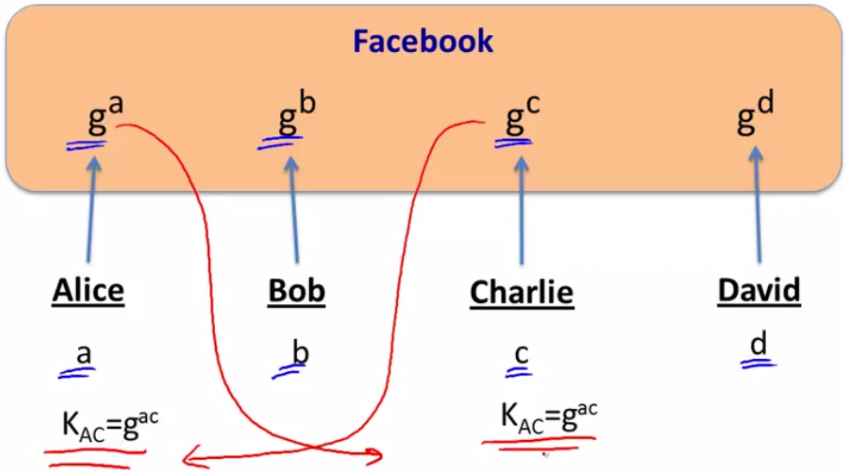
cipher key size	modulus size	Elliptic Curve size
80 bits	1024 bits	160 bits
128 bits	3072 bits	256 bits
256 bits (AES)	**15360** bits	512 bits

As a result: slow transition away from (mod p) to elliptic curves

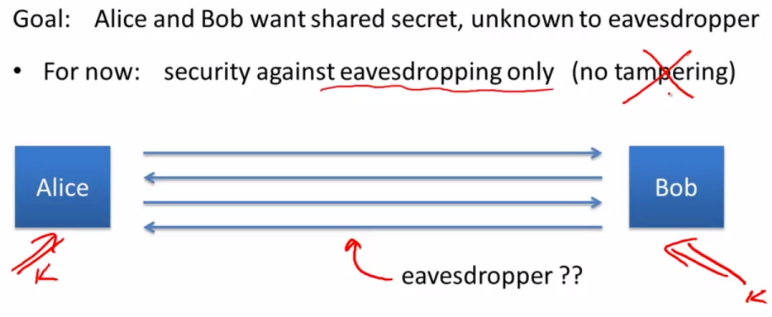
- Insecure against man-in-the-middle



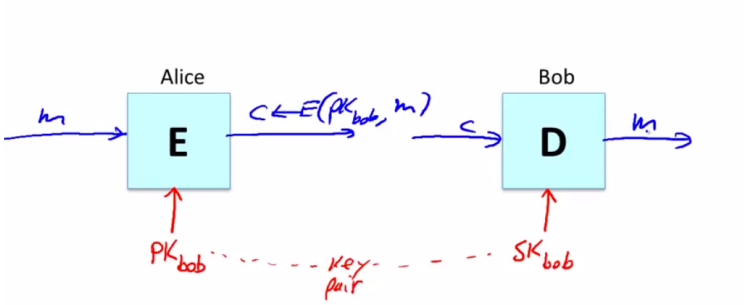
- Another look at DH



  - since there key is posted to there profile they don't need to communicate at all to set up a shared key
  - can communicate securely
- An open problem
  - n = 2 DH
  - n = 3 known (jouis)
  - n = 4 … i (open)
  - open problem joint shared key
- **Public-Key Encryption**
- Establishing a shared secret



- Public key encryption



  - pk = public key
  - sk = secret key
- Public key encryption

Def: a public-key encryption system is a triple of algs. (G, E, D)

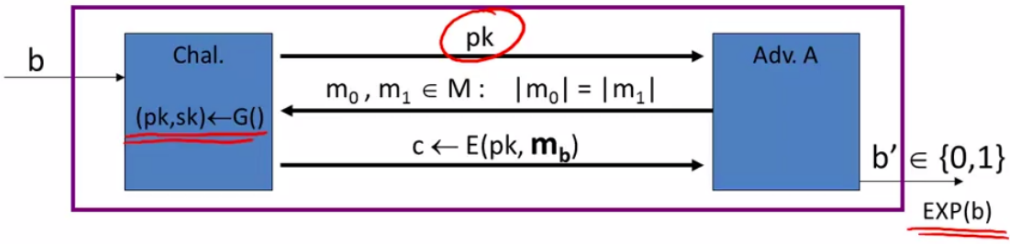
- G(): randomized alg. outputs a key pair (pk, sk)
- E(pk, m): randomized alg. that takes m∈M and outputs c ∈C
- D(sk,c): det. alg. that takes c∈C and outputs m∈M or ⊥

Consistency: ∀(pk, sk) output by G :

∀m∈M: D(sk, E(pk, m)) = m

- Semantic security

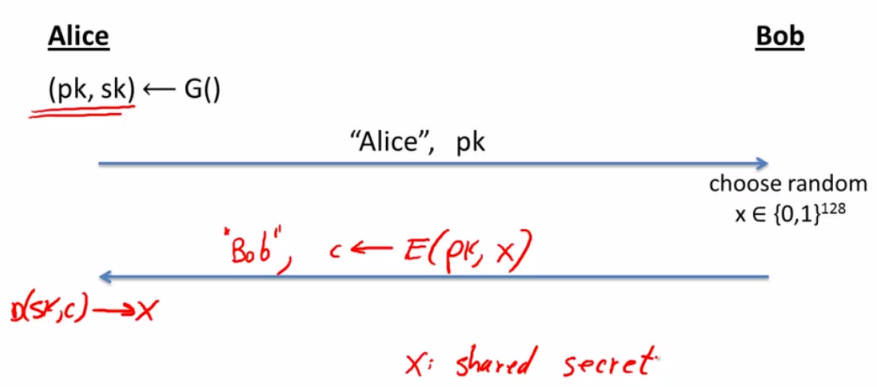
For b=0,1 define experiments EXP(0) and EXP(1) as:



Def: $\mathbb{E}$ =(G,E,D) is sem. secure (a.k.a IND-CPA) if for all efficient A:

$Adv_{SS}[A,\mathbb{E}] = |Pr[EXP(0)=1] - Pr[EXP(1)=1]| < \text{negligible}$

- Establishing a shared secret



- Security

Adversary sees **pk, E(pk, x)** and wants **x ∈M**

Semantic security ⇒
adversary cannot distinguish

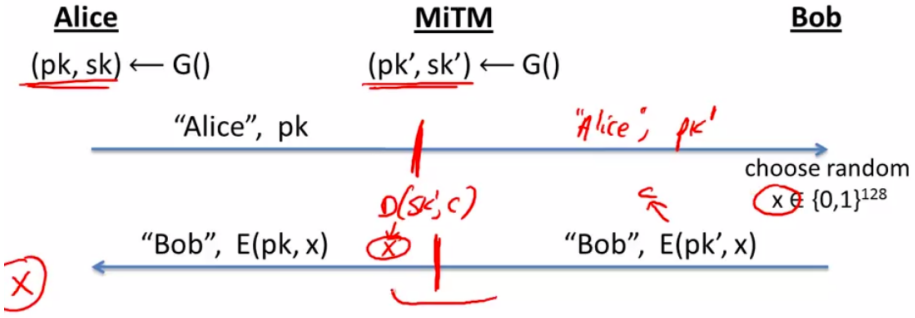
⟹ { pk, E(pk, x), x } from { pk, E(pk, x), rand∈M }

⇒ can derive session key from x.

Note: protocol is vulnerable to man-in-the-middle

- Insecure against man in the middle

As described, the protocol is insecure against **active** attacks



- Public key encryption: constructions
  - Constructions generally rely on hard problems from number theory and algebra

- **Number Theory 1: modular arithmetic**
- **Notation**
- Background
  - We will use a bit of number theory to construct
    - key exchange protocols
    - digital signatures
    - public key encryption
- Notation

From here on:

- N denotes a positive integer.
- p denote a prime.

Notation: $\mathbb{Z}_N = \{0, 1, 2, \ldots, N-1\}$

Can do addition and multiplication modulo N

- Modular arithmetic

Examples: let N = 12

$9 + 8 = 5$ in $\mathbb{Z}_{12}$

$5 \times 7 = 11$ in $\mathbb{Z}_{12}$

$5 - 7 = 10$ in $\mathbb{Z}_{12}$

Arithmetic in $\mathbb{Z}_N$ works as you expect, e.g $x \cdot (y+z) = x \cdot y + x \cdot z$ in $\mathbb{Z}_N$

- Greatest common divisor

Def: For ints. x,y: **gcd(x, y)** is the greatest common divisor of x,y

Example: gcd(12, 18) = 6 $\quad 2 \times 12 - 1 \times 18 = 6$

Fact: for all ints. x,y there exist ints. a,b such that

$a \cdot x + b \cdot y = \gcd(x,y)$

a,b can be found efficiently using the extended Euclid alg.

If gcd(x,y)=1 we say that x and y are **relatively prime**

- Modular inversion

Over the rationals, inverse of 2 is ½. What about $\mathbb{Z}_N$?

Def: The **inverse** of x in $\mathbb{Z}_N$ is an element y in $\mathbb{Z}_N$ s.t. $x \cdot y = 1$ in $\mathbb{Z}_N$

y is denoted x^{-1}.

Example: let N be an odd integer. The inverse of 2 in $\mathbb{Z}_N$ is $\left[\frac{N+1}{2}\right]$

$2 \cdot \left(\frac{N+1}{2}\right) = N+1 = 1$ in $\mathbb{Z}_N$

- Modular inversion

Which elements have an inverse in $\mathbb{Z}_N$?

Lemma: x in $\mathbb{Z}_N$ has an inverse if and only if gcd(x,N) = 1

Proof:

gcd(x,N)=1 ⇒ ∃ a,b: a·x + b·N = 1 ⇒ $a \cdot x = 1$ in $\mathbb{Z}_N$

⇒ $x^{-1} = a$ in $\mathbb{Z}_N$

gcd(x,N) > 1 ⇒ ∀a: gcd(a·x, N) > 1 ⇒ a·x ≠ 1 in $\mathbb{Z}_N$

$gcd(x,N) = 2 \Rightarrow \forall a$: $a \cdot x$ is even $\Rightarrow$ $a \cdot x \neq b \cdot N + 1$ (even ≠ odd)

- More notation

Def: $\mathbb{Z}_N^*$ = (set of invertible elements in $\mathbb{Z}_N$) =

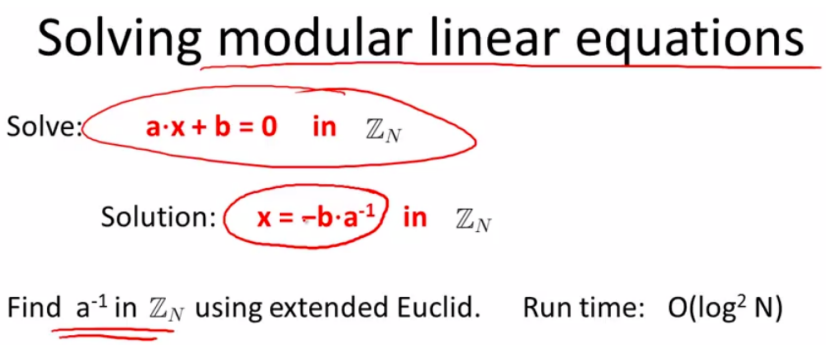
= { x∈ $\mathbb{Z}_N$: gcd(x,N) = 1 }

Examples:

1. for prime p, $\mathbb{Z}_p^* = \mathbb{Z}_p \setminus \{0\} = \{1, 2, \ldots, p-1\}$ $\quad |\mathbb{Z}_p^*| = p-1$
2. $\mathbb{Z}_{12}^*$ = { 1, 5, 7, 11}

For x in $\mathbb{Z}_N^*$, can find x^{-1} using extended Euclid algorithm.

- Solving modular linear equations



- **Fermat and Euler**
- Review

N denotes an n-bit positive integer. p denotes a prime.

- Z_N = { 0, 1, …, N-1 }
- $(Z_N)^*$ = (set of invertible elements in Z_N) = = { x∈Z_N : gcd(x,N) = 1 }

Can find inverses efficiently using Euclid alg.: time = $O(n^2)$

- Fermat's theorem

Thm: Let p be a prime

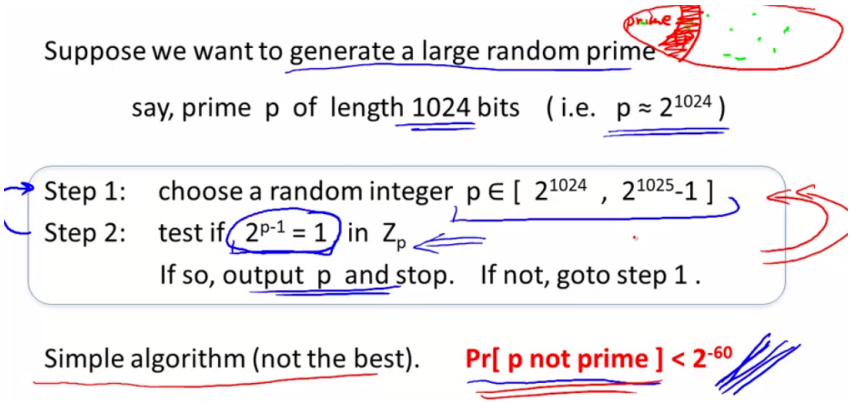
$\forall x \in (Z_p)^*$: $x^{p-1} = 1$ in Z_p

Example: p=5. $3^4 = 81 = 1$ in Z_5

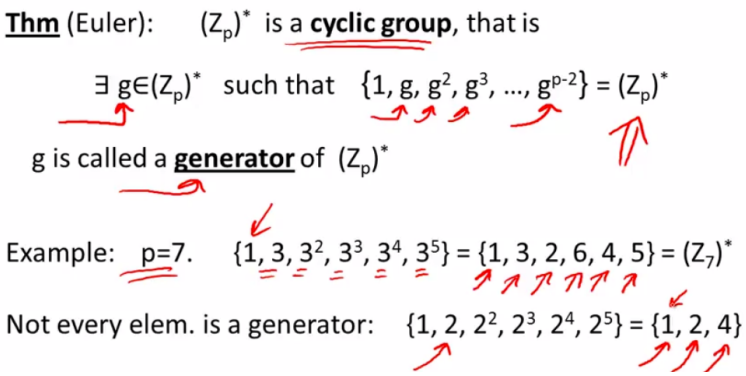
So: $x \in (Z_p)^*$ $\Rightarrow$ $x \cdot x^{p-2} = 1$ $\Rightarrow$ $x^{-1} = x^{p-2}$ in Z_p

another way to compute inverses, but less efficient than Euclid

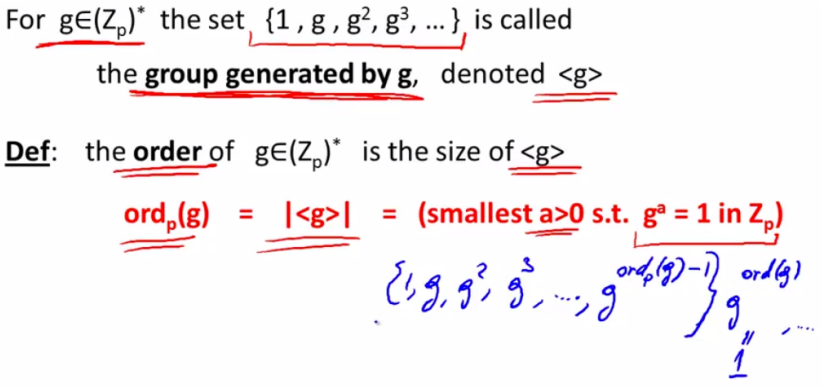
- Application: generating random primes



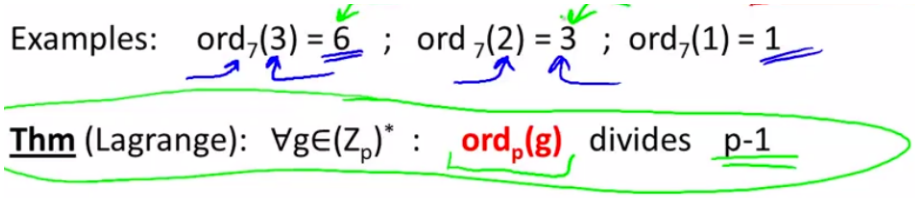
- Structure



- Order



- Examples



- Euler's generalization of Fermat

Def: For an integer N define $\varphi(N) = |(Z_N)^*|$ (Euler's φ func.)

Examples: $\varphi(12) = |\{1,5,7,11\}| = 4$; $\varphi(p) = p-1$

For N=p·q: $\varphi(N) = N-p-q+1 = (p-1)(q-1)$

Thm (Euler): $\forall x \in (Z_N)^*$: $x^{\varphi(N)} = 1$ in Z_N

Example: $5^{\varphi(12)} = 5^4 = 625 = 1$ in Z_{12}

Generalization of Fermat. Basis of the RSA cryptosystem

- **Modular e'th Roots**
- Modular e'th roots

We know how to solve modular **linear** equations:

$a \cdot x + b = 0$ in Z_N Solution: $x = -b \cdot a^{-1}$ in Z_N

What about higher degree polynomials?

Example: let p be a prime and $c \in Z_p$. Can we solve:

$x^2 - c = 0$, $y^3 - c = 0$, $z^{37} - c = 0$ in Z_p

- Modular e'th roots

Let p be a prime and $c \in Z_p$.

Def: $x \in Z_p$ s.t. $x^e = c$ in Z_p is called an **e'th root** of c.

Examples: $7^{1/3} = 6$ in $\mathbb{Z}_{11}$ $6^3 = 216 = 7$ in Z_{11}

$3^{1/2} = 5$ in $\mathbb{Z}_{11}$ $5^2 = 25 = 3$ in Z_{11}

$1^{1/3} = 1$ in $\mathbb{Z}_{11}$ $1^3 = 1$ in Z_{11}

$2^{1/2}$ does not exist in $\mathbb{Z}_{11}$

- The easy case

When does $c^{1/e}$ in Z_p exist? Can we compute it efficiently?

The easy case: suppose $\gcd(e, p-1) = 1$

Then for all c in $(Z_p)^*$: $c^{1/e}$ exists in Z_p and is easy to find.

Proof: let $d = e^{-1}$ in Z_{p-1}. Then $c^{1/e} = c^d$ in Z_p

$d \cdot e = 1$ in Z_{p-1} $\Rightarrow \exists k \in \mathbb{Z}: d \cdot e = k(p-1)+1 \Rightarrow (c^d)^e = c^{de} = c^{k(p-1)+1} = [c^{p-1}]^k \cdot c = c$ in Z_p

- The case e=2: square roots

If p is an odd prime then $\gcd(2, p-1) \neq 1$

x −x

x^2

Fact: in $\mathbb{Z}_p^*$, $x \longrightarrow x^2$ is a 2-to-1 function

Example: in $\mathbb{Z}_{11}^*$:

1	10	2	9	3	8	4	7	5	6
1		4		9		5		3	

Def: x in $\mathbb{Z}_p$ is a **quadratic residue** (Q.R.) if it has a square root in $\mathbb{Z}_p$

p odd prime $\Rightarrow$ the # of Q.R. in $\mathbb{Z}_p$ is $(p-1)/2 + 1$

- Euler's theorem

Thm: x in $(Z_p)^*$ is a Q.R. $\iff$ $x^{(p-1)/2} = 1$ in Z_p (p odd prime)

Example: in $\mathbb{Z}_{11}$: $1^5, 2^5, 3^5, 4^5, 5^5, 6^5, 7^5, 8^5, 9^5, 10^5$

= 1 -1 1 1 1 -1 -1 -1 1 -1

Note: $x \neq 0 \Rightarrow x^{(p-1)/2} = (x^{p-1})^{1/2} = 1^{1/2} \in \{1, -1\}$ in Z_p

Def: $x^{(p-1)/2}$ is called the **Legendre Symbol** of x over p (1798)

- Computing square roots mod p

Suppose $p = 3 \pmod 4$

Lemma: if $c \in Z_p^*$ is Q.R. then $\sqrt{c} = c^{(p+1)/4}$ in Z_p

Proof: $\left[c^{(p+1)/4}\right]^2 = c^{(p+1)/2} = c^{\frac{p-1}{2}} \cdot c = 1 \cdot c = c$ in Z_p

When $p = 1 \pmod 4$, can also be done efficiently, but a bit harder

run time $\approx O(\log^3 p)$

- Solving quadratic equations mod p

Solve: $a \cdot x^2 + b \cdot x + c = 0$ in Z_p

Solution: $x = (-b \pm \sqrt{b^2 - 4 \cdot a \cdot c}) / 2a$ in Z_p

- Find $(2a)^{-1}$ in Z_p using extended Euclid.
- Find square root of $b^2 - 4 \cdot a \cdot c$ in Z_p (if one exists) using a square root algorithm

- Computing e'th roots mod N??

Let N be a composite number and e>1

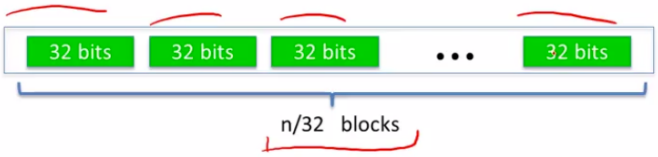
When does $c^{1/e}$ in Z_N exist? Can we compute it efficiently?

Answering these questions requires the factorization of N (as far as we know)

- **More background on number theory**
- **Number Theory 2: easy and hard problems**

- **Arithmetic algorithms**
- Representing bignums

Representing an n-bit integer (e.g. n=2048) on a 64-bit machine



Note: some processors have 128-bit registers (or more) and support multiplication on them

- Arithmetic

Given: two n-bit integers

- **Addition and subtraction**: linear time $O(n)$
- **Multiplication**: naively $O(n^2)$. Karatsuba (1960): $O(n^{1.585})$

  Basic idea: $(2^b x_2 + x_1) \times (2^b y_2 + y_1)$ with 3 mults.

  $\Rightarrow$ Best (asymptotic) algorithm: about $\tilde{O}(n \cdot \log n)$.
- **Division with remainder**: $O(n^2)$.

- Exponentiation
  - repeating square algorithm
  - multiple the appropriate powers to get the answer

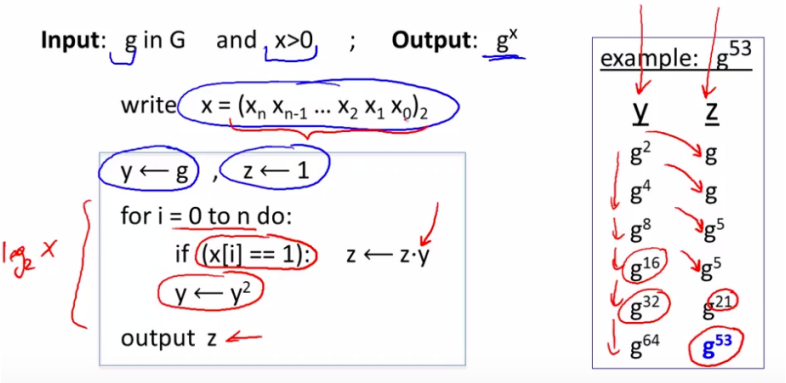
Finite cyclic group G (for example $G = \mathbb{Z}_p^*$)

Goal: given g in G and x compute g^x

Example: suppose $x = 53 = (110101)_2 = 32+16+4+1$

Then: $g^{53} = g^{32+16+4+1} = g^{32} \cdot g^{16} \cdot g^4 \cdot g^1$

- The repeated squaring algorithm



- Running times

Given n-bit int. N:

- **Addition and subtraction in Z_N**: linear time $T_+ = O(n)$
- **Modular multiplication in Z_N**: naively $T_\times = O(n^2)$
- **Modular exponentiation in Z_N (g^x)**:

$$O((\log x) \cdot T_\times) \le O((\log x) \cdot n^2) \le O(n^3)$$

$x < N$

Dan Boneh

- **Intractable Problems**
- Easy problems

- Given composite N and x in Z_N find x^{-1} in Z_N
- Given prime p and polynomial $f(x)$ in $Z_p[x]$

  find x in Z_p s.t. $f(x) = 0$ in Z_p (if one exists)

  Running time is linear in deg(f) .

... but many problems are difficult

- Intractable problems with primes

Fix a prime p>2 and g in $(Z_p)^*$ of order q.

Consider the function: $x \mapsto g^x$ in Z_p

Now, consider the inverse function:

$\text{Dlog}_g(g^x) = x$ where x in {0, …, q-2}

Example:

in $\mathbb{Z}_{11}$:	1,	2,	3,	4,	5,	6,	7,	8,	9,	10
$\text{Dlog}_2(\cdot)$:	0,	1,	8,	2,	4,	9,	7,	3,	6,	5

$2^{(4)} = 16 = 5$ in $\mathbb{Z}_{11}$

Dan Boneh

- DLOG: more generally

Let **G** be a finite cyclic group and **g** a generator of G

$G = \{1, g, g^2, g^3, \ldots, g^{q-1}\}$ (q is called the order of G)

Def: We say that **DLOG is hard in G** if for all efficient alg. A:

$\Pr_{g \leftarrow G,\, x \leftarrow Z_q}[\, A(G, q, g, g^x) = x \,] < \text{negligible}$

Example candidates:

(1) $(Z_p)^*$ for large p, (2) Elliptic curve groups mod p

- Computing DLOG in (Z_p)* (n-bit prime p)

Best known algorithm (GNFS): run time $\exp(\tilde{O}(\sqrt[3]{n}))$

cipher key size	modulus size	Elliptic Curve group size
80 bits	1024 bits	160 bits
128 bits	3072 bits	256 bits
256 bits (AES)	**15360** bits	512 bits

$e^{w/2}$

  - as a result: slow transition away from (mod p) to elliptic curves
- An application: collision resistance

Choose a group G where Dlog is hard (e.g. $(Z_p)^*$ for large p)

Let q = |G| be a prime. Choose generators g, h of G

For $x,y \in \{1,\ldots,q\}$ define $H(x,y) = g^x \cdot h^y$ in G

Lemma: finding collision for H(.,.) is as hard as computing $\text{Dlog}_g(h)$

Proof: Suppose we are given a collision $H(x_0,y_0) = H(x_1,y_1)$

then $g^{x_0} \cdot h^{y_0} = g^{x_1} \cdot h^{y_1} \Rightarrow g^{x_0 - x_1} = h^{y_1 - y_0} \Rightarrow h = g^{(x_0 - x_1)/(y_1 - y_0)}$

- Intractable problems with composites
  - Z notation 2 means that the numbers of two prime factors n means that they are n bit numbers

Consider the set of integers: (e.g. for n=1024)

$\mathbb{Z}_{(2)}(n) := \{ N = p \cdot q \text{ where } p,q \text{ are n-bit primes} \}$

Problem 1: Factor a random N in $\mathbb{Z}_{(2)}(n)$ (e.g. for n=1024)

n=2048

Problem 2: Given a polynomial **f(x)** where degree(f) > 1

and a random N in $\mathbb{Z}_{(2)}(n)$

find x in $\mathbb{Z}_N$ s.t. f(x) = 0 in $\mathbb{Z}_N$

- The factoring problem

Best known alg. (NFS): run time exp($\tilde{O}(\sqrt[3]{n})$) for n-bit integer

Current world record: **RSA-768** (232 digits)

- Work: two years on hundreds of machines
- Factoring a 1024-bit integer: about 1000 times harder

$\Rightarrow$ likely possible this decade

-