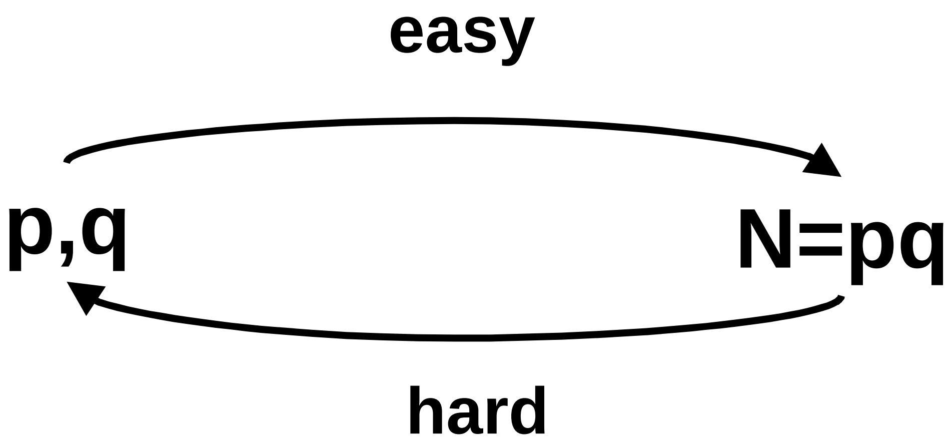
Introduction to Cryptography Lecture

RSA encryption, Rabin encryption, digital signatures

Benny Pinkas

Integer Multiplication & Factoring as a One Way Function.



Can a public key system be based on this observation ?????

Excerpts from RSA paper (CACM, 1978)

The era of "electronic mail" may soon be upon us; we must ensure that two important properties of the current "paper mail" system are preserved: (a) messages are *private*, and (b) messages can be *signed*. We demonstrate in this paper how to build these capabilities into an electronic mail system.

At the heart of our proposal is a new encryption method. This method provides an implementation of a "public-key cryptosystem," an elegant concept invented by Diffie and Hellman. Their article motivated our research, since they presented the concept but not any practical implementation of such system.

The Multiplicative Group Z_{pq}^*

- *p* and *q* denote two large primes (e.g. 512 bits long).
- Denote their product as $N = pq$.
- The multiplicative group $Z_N^* = Z_{pq}^*$ contains all integers in the range *[1,pq-1]* that are relatively prime to both *p* and *q*.

- The size of the group is
  - $\phi(n) = \phi(pq) = (p-1)(q-1) = N - (p+q) + 1$

- For every $x \in Z_N^*$, $x^{\phi(N)} = x^{(p-1)(q-1)} = 1 \bmod N$.

Exponentiation in Z_N^*

- Motivation: use exponentiation for encryption.

- Let e be an integer, $1 < e < \phi(N) = (p-1)(q-1)$.
  - Question: When is exponentiation to the e^{th} power, ($x \to x^e$), a one-to-one operation in Z_N^* ?

- Claim: If e is relatively prime to $(p-1)(q-1)$ (namely $gcd(e, (p-1)(q-1))=1$) then $x \to x^e$ is a one-to-one operation in Z_N^*.
- Constructive proof:
  - Since gcd$(e, (p-1)(q-1))=1$, e has a multiplicative inverse modulo $(p-1)(q-1)$.
  - Denote it by d, then $ed=1+c(p-1)(q-1)=1+c\phi(N)$.
  - Let $y=x^e$, then $y^d = (x^e)^d = x^{1+c\phi(N)} = x$.
  - I.e., $y \to y^d$ is the inverse of $x \to x^e$.

The RSA Public Key Cryptosystem

- Public key:
  - $N=pq$ the product of two primes (we assume that factoring N is hard)
  - e such that $\gcd(e,\phi(N))=1$ *(are these hard to find?)*
- Private key:
  - d such that $de \equiv 1 \bmod \phi(N)$
- Encryption of $M \in Z_N{}^*$
  - $C=E(M)=M^e \bmod N$
- Decryption of $C \in Z_N{}^*$
  - $M=D(C)=C^d \bmod N$ *(why does it work?)*

Constructing an instance of the RSA PKC

- Alice
  - picks at random two large primes, *p* and *q*.
  - picks (uniformly at random) a (large) *d* that is relatively prime to (p-1)(q-1) (namely, gcd$(d,\phi(N))=1$).
  - Alice computes *e* such that $de\equiv 1 \bmod \phi(N)$

- Let $N=pq$ be the product of *p* and *q*.
- Alice publishes the public key (N,e).
- Alice keeps the private key *d*, as well as the primes *p, q* and the number $\phi(N)$, in a safe place.

Efficiency

- The public exponent e may be small.
  - It is common to choose its value to be either 3 or $2^{16}+1$. The private key d must be long.
  - Each encryption involves only a few modular multiplications. Decryption requires a full exponentiation.

- Usage of a small $e \Rightarrow$ Encryption is more efficient than a full blown exponentiation.

- Decryption requires a full exponentiation ($M=C^d \bmod N$)
- Can this be improved?

The Chinese Remainder Theorem (CRT)

- Thm:
  - Let $N=pq$ with gcd(p,q)$=1$.
  - Then for every pair $(y,z) \in Z_p \times Z_q$ there exists a *unique* $x \in Z_n$, s.t.
    - $x=y \bmod p$
    - $x=z \bmod q$
- Proof:
  - The extended Euclidian algorithm finds a,b *s.t.* $ap+bq=1$.
  - Define $c=bq$. Therefore $c=1 \bmod p$. $c=0 \bmod q$.
  - Define $d=ap$. Therefore $d=0 \bmod p$. $d=1 \bmod q$.
  - *Let* $x=cy+dz \bmod N$.
    - $cy+dz = 1y + 0 = y \mod p$.
    - $cy+dz = 0 + 1z = z \bmod q$.
  - (How efficient is this?)
  - (The inverse operation, finding (y,z) from x, is easy.)

More efficient RSA decryption

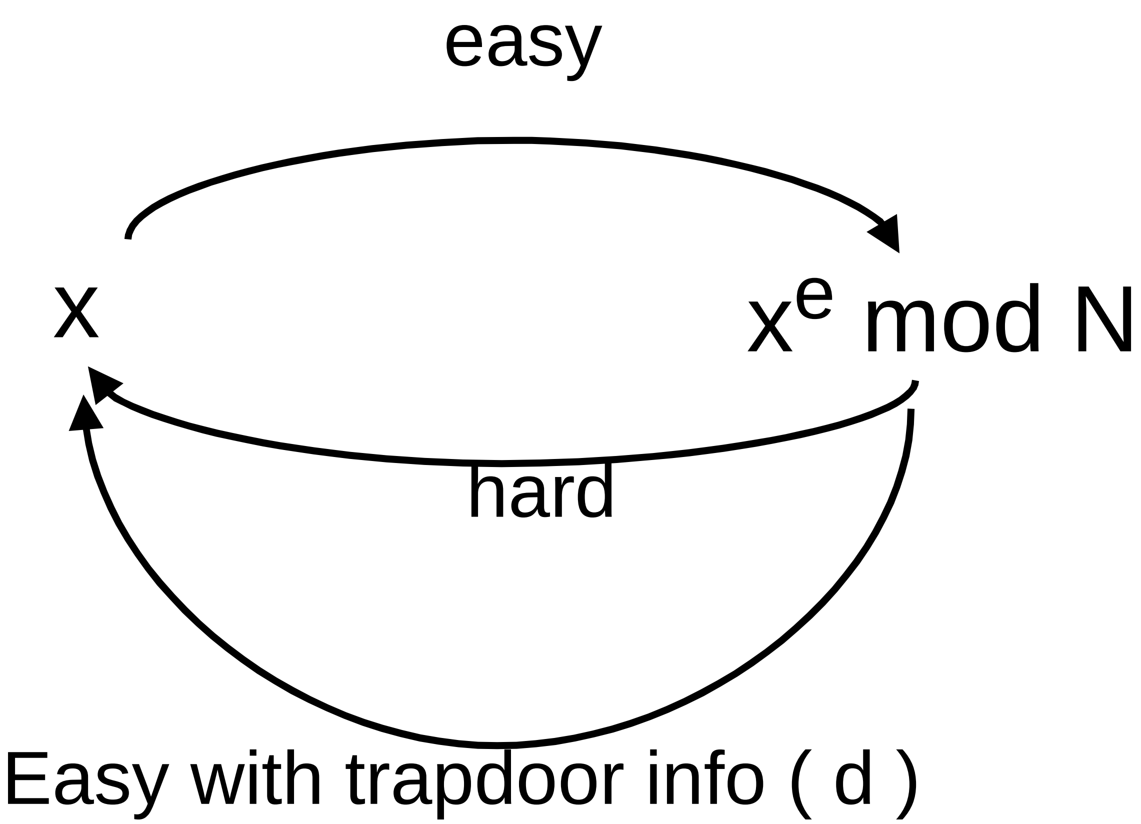
- CRT:
  - Given p,q compute a,b s.t. $ap+bq=1.$
  - $c=bq;\ \ d=ap$

Once for all messages

- Decryption, given C:
  - Compute $y'=C^d \bmod p$. (instead of d can use $d'=d \bmod p\text{-}1$)
  - Compute $z'=C^d \bmod q$. (instead of d can use $d''=d \bmod q\text{-}1$)
  - Compute $M=cy'+dz' \bmod N$.

- Overhead:
  - Two exponentiations modulo $p,q,$ instead of one exponentiation modulo N.
  - Overhead of exponentiation is *cubic* in length of modulus.
  - I.e., save a factor of $2^3/2.$

RSA as a One Way Trapdoor Permutation



Security reductions

- Security by reduction
  - Define what it means for the system to be “secure” (chosen plaintext/ciphertext attacks, etc.)
  - State a “hardness assumption” (e.g., that it is hard to extract discrete logarithms in a certain group).
  - Show that if the hardness assumption holds then the cryptosystem is secure.

- Benefits:
  - To examine the security of the system it is sufficient to check whether the assumption holds
  - Similarly, for setting parameters (e.g. group size).

RSA Security

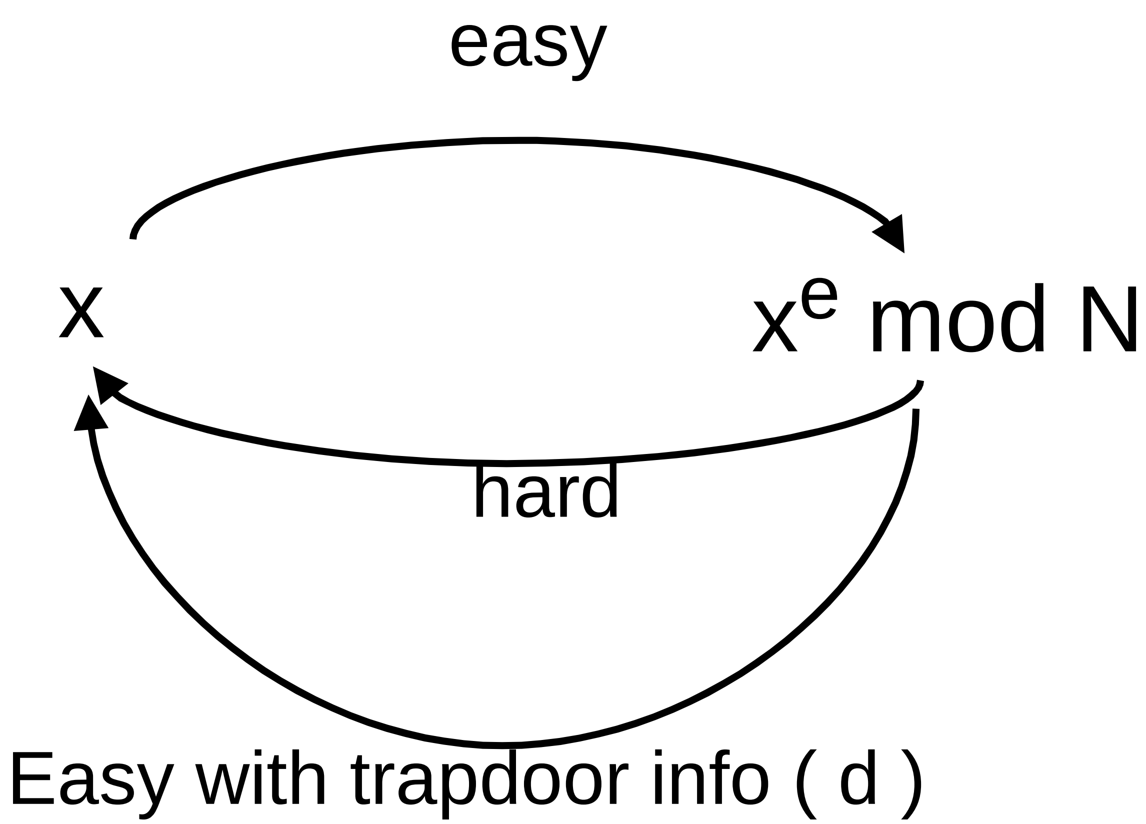
- (For ElGamal encryption, we showed that if the DDH assumption holds then El Gamal encryption has semantic security.)
- If factoring *N* is easy then RSA is insecure
  - (factor $N \Rightarrow$ find $p,q \Rightarrow$ find $(p-1)(q-1) \Rightarrow$ find d from e)
- Factoring assumption:
  - For a randomly chosen prime numbers p,q of appropriate length, it is infeasible to factor $N=pq$.
- This assumption might be too weak (might not ensure secure RSA encryption)
  - Maybe it is possible to break RSA without factoring *N* ?
  - We don't know how to reduce RSA security to the hardness of factoring.

- Fact: finding *d* is equivalent to factoring.
  - I.e., if it is possible to find *d* given (N,e) , then it is easy to factor *N.*
- Therefore, "hardness of finding *d* assumption" no stronger than hardness of factoring.

The RSA assumption: Trap-Door One-Way Function (OWF)

- (what is the minimal assumption required to show that RSA encryption is secure?)
- (Informal) definition: $f : D \rightarrow R$ is a *trapdoor one way function* if there is a trap-door d such that:
  - Without knowledge of d, the function f is a one way. I.e., for a randomly chosen x, it is hard to invert $f(x)$.
  - Given d, inverting f is easy
- Example: $f_{g,p}(x) = g^x \bmod p$ is *not* a trapdoor one way function.
- Example: the assumption that RSA is a trapdoor OWF
  - $f_{N,e}(x) = x^e \bmod N$. (assumption: for a random N,e,x, inverting is hard.)
  - The trapdoor is d s.t. $ed = 1 \bmod \phi(N)$
  - $[f_{N,e}(x)]^d = x \bmod N$

RSA as a One Way Trapdoor Permutation



RSA assumption: cautions

- The RSA assumption is quite well established:
  - RSA is actually a Trapdoor One-Way *Permutation*
  - Hard to invert on random input (if you don't know the secret key)
- But is it a secure cryptosystem?
  - Given the assumption it is hard to reconstruct the input, but is it hard to learn *anything* about the input?
- Theorem [G]: RSA hides the log(log(N)) least *and* most significant bits of a uniformly-distributed random input
  - But some (other) information about pre-image may leak
  - And… adversary can detect a repeating message
- And, of course, as a deterministic cipher RSA does not provide semantic security.

Security of RSA

- Chosen ciphertext attack: (homomorphic property)
  - Textbook RSA is also susceptible to chosen ciphertext attacks:
    - We are given a ciphertext $C=M^e$
    - We can choose a random R and generate $C'=CR^e$ (an encryption of $M\cdot R$).
    - Suppose we can receive the decryption of C'. It is equal to $M\cdot R$.
    - We divide it by R and reveal M.

Padded RSA

- In order to make textbook RSA semantically secure we must change it to be a probabilistic encryption
- For example, we could pad the message with random bits.
  - Suppose that messages are of length $|N|-L$ bits
  - To encrypt a message M, choose a random string r of length L, and compute $(r \mid M)^e$ mod N.
  - When decrypting, output only the last $|N|-L$ bits of C^d mod N

- Any message has 2^L possible encryptions. L must be long enough so that a search of all 2^L pads is inefficient.
- There is no known proof that this secure.
- Similar schemes are known to be secure under certain assumptions

Is it safe to use a common modulus ?

- Consider the following environment:
  - There is a global modulus N. No one knows its factoring.
  - Each party has a pair (e_i,d_i), such that $e_i{,}d_i = 1 \bmod \phi(N)$.
    - Used as a public/private key pair.

- The system is insecure.

- Party 1, knowing (e_1,d_1)
  - can factor N
  - Find d_i for any other party i.

RSA with a small exponent

- Setting $e=3$ enables efficient encryption
- Might be insecure if not used properly
  - Assume three users with public keys N_1, N_2, N_3.
  - Alice encrypts the same message to all of them
    - $C_1 = m^3 \bmod N_1$
    - $C_2 = m^3 \bmod N_2$
    - $C_3 = m^3 \bmod N_3$
- Can an adversary which sees C_1,C_2,C_3 find m?
  - $m^3 < N_1N_2N_3$
  - N_1, N_2 and N_3 are most likely relatively prime (otherwise we can factor them).
  - Chinese remainder theorem -> can find $m^3 \bmod N$ (and therefore m^3 over the integers)
  - Easy to extract 3rd root over the integers.

Rabin's encryption systems

- Key generation:
  - Private key: random primes p,q (e.g. 512 bits long).
  - Public key: $N=pq$.

- Encryption:
  - Plaintext $m \in Z_N^*$.
  - Ciphertext: $c = m^2 \bmod N$. *(very efficient)*

- Decryption: Compute $c^{1/2} \bmod N$.

Square roots modulo N

- $\Rightarrow$ Let x be a quadratic residue (QR) modulo $N=pq$, then
  - x mod p is a QR mod p. x mod q is a QR mod q
  - x mod p has *two* roots mod p: y and $p - y$
  - x mod q has *two* roots mod q: z and $q - z$

- $\Leftarrow$ If x is a QR mod p and mod q, it is also a QR mod N. (Follows from the Chinese remainder theorem.)

Square roots modulo N

- If x has a square root modulo N then it has 4 different square roots modulo N.
  - Let A be s.t. $A^2=x$ mod N.
  - Let c be s.t. $c=1$ mod p, $c=-1$ mod q.
  - Then $A, -A, cA, -cA$ are all square roots of x modulo N.

- Each combination of roots modulo p and q results in a root modulo N.
  - x therefore has four roots modulo pq:
  - (y,z) -> A, $(p - y, q - z)$ -> $pq – A$
  - $(y, q - z)$ -> B, $(p – y, z)$ -> $pq – B$

$= (y,z) \cdot (1,-1)$

Square roots modulo *N*

- Exactly ¼ of the elements are QR mod *N.*
  - $QR_N = QR_p \times QR_q$. $|QR_N| = (p-1)(q-1)/4$

- Assume that *p=q=3 mod 4.* (Blum integers.)
  - Then it is easy to see that exactly one of the four roots is a QR mod *N.*
    - Note that -1 is an NQR mod *p* and mod *q* (Euler's thm).
    - Let one of the square roots of *x* modulo *N* be *A.*
    - Then the other square roots are *-A, cA, -cA,* where *c=1* mod *p*, *c=-1* mod *q*.
    - Assume that *A* is a QR mod *N*, and therefore it is a QR *p* and a QR mod *q.* Then none of the other roots is a QR mod *p* and a QR mod *q.*

Finding square roots modulo N

- Need to compute $y=x^{1/2}$ mod N.
- Suppose we know (the private key) p, q.
  - Compute the roots of x modulo p, q. Use Chinese remainder theorem to find x.

- Computing square roots in Z_p^*,
  - Recall, $x \in QR_p$ iff $x^{(p-1)/2}=1$ mod p.
  - Assume $p=3$ mod 4. (p is a Blum integer).
  - Compute the root as $y=x^{(p+1)/4}$ mod p.
    - $(p+1)/4$ is an integer
    - $y^2 = (x^{(p+1)/4})^2 = x^{(p+1)/2} = x^{(p-1)/2}x = x$
  - If $p=1$ mod 4 the computation is more complicated (no deterministic algorithm is known)

Decryption of Rabin cryptosystem

- Input: $c, p, q.$ ($p=q=3$ mod 4)
- Decryption:
  - Compute $m_p = c^{(p+1)/4}$ mod p.
  - Compute $m_q = c^{(q+1)/4}$ mod q.
  - Use CRT to compute the four roots mod N, i.e. four values mod N corresponding to (m_p, m_q), $(p-m_p, m_q)$, $(m_p, q-m_q)$, $(p-m_p, q-m_q)$.
- There are four possible options for the plaintext!
  - The receiver must select the correct plaintext
  - This can be solved by requiring the sender to embed some redundancy in m
    - E.g., a string of bits of specific form
    - Make sure that m is always a QR

Security of the Rabin cryptosystem

- Good news:
  - The Rabin cryptosystem is secure against passive attacks iff factoring is hard. ☺

- Bad news:
  - The Rabin cryptosystem is completely insecure against chosen-ciphertext attacks ☹

Security of the Rabin cryptosystem

- Security against chosen plaintext attacks
- Suppose there is an adversary that completely breaks the system
  - Adversary's input: N, c
  - Adversary's output: m s.t. $m^2 = c \bmod N$.

- We show a reduction showing that given this adversary we can break the factoring assumption.
- I.e., we build an algorithm:
  - Input: N
  - Operation: can ask queries to the Rabin decryption oracle
  - Output: the factoring of N.
- Therefore, if one can break Rabin's cryptosystem it can also solve factoring.
- Therefore, if factoring is hard the Rabin cryptosystem is "secure" in the sense defined here.

The reduction

- Input: N
- Operation:
  - Choose random x.
  - Send N and $c=x^2 \bmod N$, to adversary.
  - Adversary answers with y s.t. $c=y^2 \bmod N$.
  - If $y=x$ or $y=N-x$, go back to step *1.*
  - Otherwise
    - $x^2 - y^2 = 0 \bmod N$.
    - $0 \neq (x-y)(x+y) = cN = cpq$.
    - Compute gcd$(x+y,N)$, gcd$(x-y,N)$ and obtain p or q.
    - (The gcd is not N since $0<x,y<N$, and therefore $-N < x+y,x-y < 2N$, and it is known that $x+y,x-y \neq 0,N$).

happens with prob 1/2

Insecurity against chosen-ciphertext attacks

- A chosen-ciphertext attack reveals the factorization of N.
- The attacker's challenge is to decrypt a ciphertext c.
- It can ask the receiver to decrypt any ciphertext except c.
- The attacker can use the receiver as the "adversary" in the reduction, namely
  - Chooses a random x and send $c=x^2$ mod N to the receiver
  - The receiver returns a square root y of c
  - With probability ½, $x \neq y$ and $x \neq -y$. In this case the attacker can factor N by computing gcd$(x-y,N)$.

  - (The attack does not depend on homomorphic properties of the ciphertext. Namely, it is not required that E(x)E(y)=E(xy).)

Comparing RSA and Rabin encryption

- RSA encryption is infinitely more popular than Rabin encryption (also more popular than El Gamal)
- Advantage of Rabin encryption: it seems more secure, security of Rabin is equivalent to factoring and we don't know to show that for RSA.
- Advantages of RSA
  - RSA is a permutation, whereas decryption in Rabin is more complex
  - Security of Rabin is only proven for encryption as $C=M^2 \bmod N$, and this mode
    - does not enable to identify the plaintext
    - is susceptible to chosen ciphertext attack.