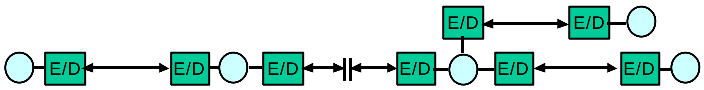
Operation of E/D device

(1) link-by-link



Ex : M/W Link, Satellite Link etc

(2) end-to-end

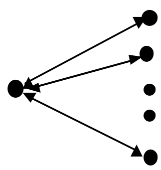
Ex : Telephone, Fax, Data Terminal etc

(3) Hybrid operation: (1) + (2)

Problem of Symmetric Cryptosystems

❑ **Key management**

- ✓ Keep secret key in secret
- ✓ Over complete graph with n nodes, ${}_nC_2$ = n(n-1)/2 pairs secret keys are required.
- ✓(Ex) n=100, 99 x 50 = 4,950 keys

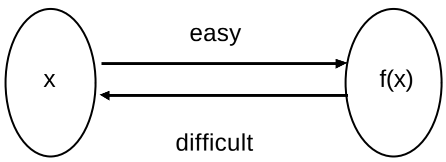


Merkle's Puzzle

- Merkle registered Fall 1974 for Lance Hoffman's course in computer security at UC, Berkeley.
- Hoffman wanted term papers & proposal.
- Merkle addressed "Secure Communication over Insecure Channels" CACM, pp.294-299,1978.
- Hoffman didn't understand Merkle's proposal and asked him to write precisely 2 times.
- Merkle dropped the course, but continued working.
- Key idea : Hiding a key in a large collection of puzzles. (Later he proposed knapsack PKC)

Concepts of PKC(I)

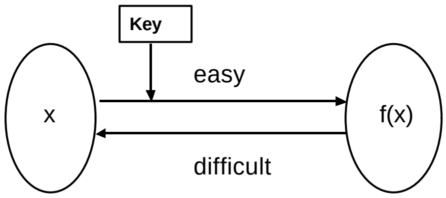
- **1-way ft.**
  - ✓ Given x, easy to compute f(x).
  - ✓ Difficult to compute $f^{-1}(x)$ for given f(x).



Ex) $f(x)= x^5 + x^3 + x^2+1$

Concepts of PKC(II)

- **Keyed 1-way ft : 1-way ft with a key**

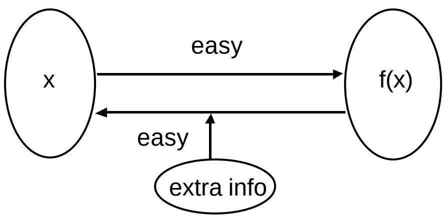


5

©ICU Kwangjo Kim

Concepts of PKC(III)

- **1-way trapdoor ft.**
  - ✓ **Given x, easy to compute f(x)**
  - ✓ **Easy to compute $f^{-1}(x)$ for given f(x) and some information -> trapdoor information**

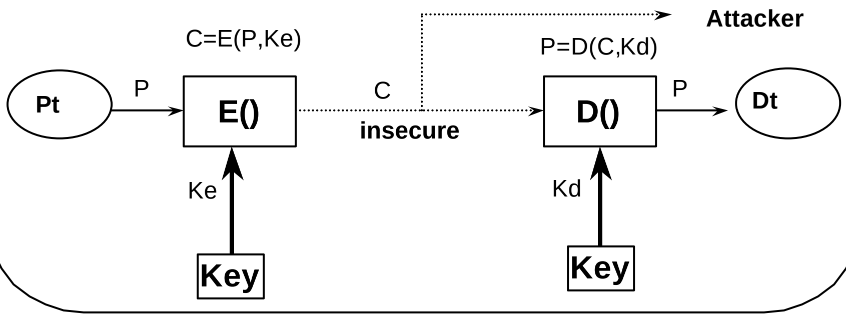


6

©ICU Kwangjo Kim

Concepts of PKC(IV)

- Use 2 keys
  - ✓ Given public key, easy to compute -> anyone can lock.
  - ✓ Only those has secret key, compute inverse -> only who has it can unlock, vice versa.



Concepts of PKC(V)

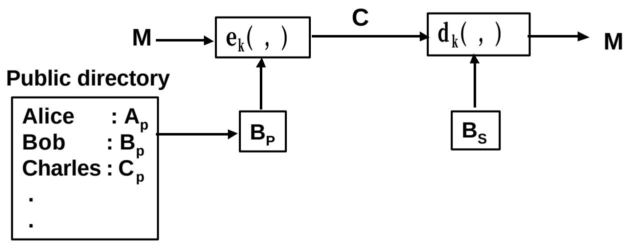
- Diffie & Hellman, “New directions in Cryptography”, IEEE Tr. on IT. ,Vol. 22, pp. 644-654, Nov., 1976.
- 2-key or Asymmetric Cryptosystem
- PKC (Public-Key Cryptosystem)
  - private(secret) key, public key
- Need Public key directory
- Slow operation relative to symmetric cryptosystem

What service PKC provides ?(I)

❑ For Privacy

- Encrypt M with Bob' s public key : $C = e_K(B_p, M)$
- Decrypt C with Bob' s private key : $D = d_K(B_s, C)$

*Anybody can generate C, but only B can recover C.

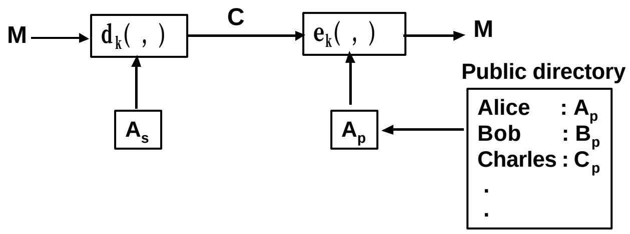


What service PKC provides ?(II)

❑ For authentication(Digital Signature)

- Encrypt M with Alice' s private key : $C = d_K(A_s, M)$
- Decrypt C with Alice' s public key : $D = e_K(A_p, C)$

* Only Alice can generate C, but anybody can recover C.



What service PKC provides ?(III)

- Identification
- Non-Repudiation
- Applicable to various cryptographic protocols
- Hybrid use with symmetric cryptosystem

11

PKC Schemes

- RSA scheme ('78) : R.L.Rivest, A.Shamir, L.Adleman, "A Method for Obtaining Digital Signatures and Public Key Cryptosystems",CACM, Vol.21, No.2, pp.120-126,Feb,1978
- McEliece scheme ('78)
- Rabin scheme ('79)
- Knapsack scheme ('79-)
- Williams scheme ('80)
- ElGamal scheme ('85)
- Elliptic Curve based scheme('85)
- Braid group Cryptosystem(2000)

12

Security of PKC

- Discrete Logarithm Problem (DLP)
- Integer Factorization Problem (FP)
- Quadratic Residue
- Linear Code Decoding
- CLP (Closest Lattice Problem)
- DLP over Elliptic Curve

* subexp. problem : O(exp c sqrt{log(n)log(log(n))}

Comparision

Cryptosystem Item	Symmetric	Asymmetric (O : merit X : demerit)
Key relation	Enc. key = Dec. key	Enc. Key ¹ Dec. key
Enc. Key	Secret	Public,{private}
Dec. key	Secret	Private,{public}
Algorithm	Secret Public	Public
Typical ex.	Skipjack DES	RSA
Key Distribution	Req'd (X)	Not req'd (O)
Number of keys	Many(X),keep many partners' secret key	Low(O), keep his pri. Key only
Secure authentication	Hard(X)	Easy(O)
E/D Speed	Fast(O)	Slow(X)

Def. of Provable Security

- **OW (Onewayness) : given a challenge ciphertext y, adversary's inability to decrypt y and get the whole plaintext x.**
- **IND (Indistinguishability) : given a challenge ciphertext y, adversary's inability to learn any information about the plaintext x.**
- **NM (Non-malleability) : given a challenge ciphertext y, adversary's inability to get a different ciphertext y' s.t. the corresponding plaintexts, x and x' are meaningfully related. e.g., meaningful relation x = x' +1.**

RSA Scheme(I)

- **For large 2 primes *p,q***
- ***n=pq , $\phi(n)=(p-1)(q-1)$: Euler phi ft.***
- **Select random *e* s.t. gcd($\phi(n)$, *e*) = 1**
- **Compute *ed* = 1 mod $\phi(n)$ -> $ed = k\phi(n) +1$**
- **Public key = {e, n}, secret key = {d, {n}}**
- **For given M in [0, n-1],**
- **Encryption, $C = M^e$ mod n**
- **Decryption, $D = C^d$ mod n**

(Proof) $C^d = (M^e)^d = M^{ed} = M^{k\phi(n)+1} = M \{M^{\phi(n)}\}^k = M$

RSA Scheme(II)

- ***p***=3, ***q***=11
- ***n = pq*** = 33, ***ƒ(n)*** =(***p***-1)(***q***-1)=2 x10 = 20
- ***e*** = 3 ***s.t***. gcd(***e***, ***ƒ(n)***)=(3,20)=1
- Choose d ***s.t. ed*** =1 mod***ƒ(n),*** 3d=1mod 20, d=7
- Public key ={e,n}={3,33}, private key ={d}={7}

- M =5
- C = M^e mod n = 5^3 mod 33 =26
- M =C^d mod n = 26^7 mod 33= 5

RSA Scheme(III)

- ***p***=2357, ***q***=2551
- ***n = pq*** = 6012707
- ***ƒ(n)*** = (***p***-1)(***q***-1) = 6007800
- ***e*** = 3674911 ***s.t.*** gcd(***e***, ***ƒ(n)***)=1
- Choose d ***s.t. ed*** =1 mod***ƒ(n), d***= 422191
- M =5234673
- C = M^e mod n = $5234673^{3674911}$ mod 6012707
  = 3650502
- M =C^d mod n = 3650502^{422191} mod 6012707
  = 5234673

Fast Exp. Algorithm(I)

❑ **Repeated Square-and-multiply**

INPUT : g, and pos. int $e=(e_t e_{t-1} ..e_1 e_0)_2$

OUPTUT : g^e mod n

1. **A =1**
2. **For i from t down to 0 do the following**
   - **2.1 A = A · A**
   - **2.2 If e_i=1, then A = A · g mod n**
3. **Return(A)**

19

Fast Exp. Algorithm(II)

❑ **(Ex) $g^{283,}$ t=8, 283=$(100011011)_2$**

i	8	7	6	5	4	3	2	1	0
e_i	1	0	0	0	1	1	0	1	1
A	g	g^2	g^4	g^8	g^{17}	g^{35}	g^{70}	g^{141}	g^{283}

❑ **Workload**

t+1 : bit length of e

wt(e) : e' s Hamming weight

t+1 times : squaring, wt(e)-1 times : mul. by g

20

Fast RSA Computation

(1) $M=C^d$ mod n where n=pq

(2) (Def)c_1=C mod p, c_2 = C mod q

d_1=d mod (p-1),d_2= d mod(q-1)

(3) If m_1=M mod p, m_2 = M mod q

then $m_1=c_1^{d1}$ mod p, $m_2=c_2^{d2}$mod q

(4) Solve 2 Eqs. in (3)

M=m_1 mod p, M=m_2 mod q

(5) Using CRT in (4), get M as (I)

<Effect> Faster 4 ~ 8 times than direct computation as (1) while keeping p and q secret

Security of RSA Scheme(I)

- **When using Common Modulus**
  - use m pairs of (e_i, d_i) given n=pq
- **(Cryptanalysis)**
  - User m_1 : $C_1 = M^{e_1}$ mod n
  - User m_2 : $C_2 = M^{e_2}$ mod n
  - if gcd(e_1,e_2)=1, there are a and b s.t. $ae_1 + be_2 = 1$.

  Then, $(C_1)^a(C_2^{-1})^{|b|}$ mod n=$(M^{e_1})^a((M^{e_2})^{-1})^{|b|}$ mod n = $M^{ae1+be2}$ mod n = M mod n

Security of RSA Scheme(II)

- Bit Security (LSB, parity)
- Special Attack
  - Periodic attack $f^m(C)=C$ where $f(x) = x^e \bmod n$
  - Special form
    - ✓ Pr{C=k× p or m× q} = 1/p + 1/q -1/pq
    - ✓ Pr{C=M} = 9/pq
  - Exhaustive search of n
  - Low exponent(e=3) attack

RSA Chips

Company	Clock Speed (MHz)	Buad rate per 512 bits	Clock cycles per 512 bits encryption	Technology	Bits/Chip	# of Trs.
Alpha Tech.	25	13K	.98M	2 m	1024	180,000
AT&T	15	19K	.4M	1.5 m	298	100,000
BT	10	5.1K	1M	2.5 m	256	------
Business Sim.	5	3.8K	.67M	GA	32	------
Calmos Sys.	20	28K	.36M	2 m	593	95,000
CNET	25	5.3K	2.3M	1 m	1024	100,000
Cryptech	14	17K	.4M	GA	120	33,000
Cylink	30	6.8K	1.2M	1.5 m	1024	150,000
GEC Marconi	25	10.2K	.67M	1.4 m	512	160,000
Pijnenburg	25	50K	.256M	1 m	1024	400,000
Sandia	8	10K	.4M	2 m	272	86,000
Siemens	5	8.5K	.3M	1 m	512	60,000

RSA speed with 8-bit public key (on SPARC II)

	512bits (sec)	768bits (sec)	1024bits (sec)
Encrypt	0.03	0.05	0.08
Decrypt	0.16	0.48	0.93
Sign	0.16	0.52	0.97
Verify	0.02	0.07	0.08

Distribution of prime

- p(x) : # of primes in [2,x] ~ x / ln(x)
- Probabilistic Prime Generation
  (1) Generate candidate random #
  (2) Test for primality
  (3) If composite, goto (1)
- Pseudo Prime (composites passing Fermat test)

Ex) 341=11x31, 2^{341-1} = 1mod 341

Prime generation(I)

Fermat Test(n,t)

Input : odd int. $n \geq 3$, security parameter : t

Output : prime or composite

1. For i=1 to t

1.1 Choose random a, $2 \leq a \leq n-2$.

1.2 Compute $r=a^{n-1}$ mod

1.3 If $r \neq 1$ then return("composite")

2. Return("prime")

Prime generation(II)

Solovay-Strassen Test(n,t)

Input : odd int. $a \geq 3$, security parameter : t

Output : "prime" or "composite"

1. For i=1 to t

1.1 Choose random a, $2 \leq a \leq n-2$

1.2 Compute $r=a^{(n-1)/2}$ mod n

1.3 If $r \neq 1$ and $r \neq n-1$ then return("composite")

1.4 Compute Jacobi symbol s =(a/n)

1.5 If $r \neq s$ mod n then return("composite")

2. Return("prime)

Prime Generation(III)

Miller-Rabin Test(n,t):
Input : odd int. $a \geq 3$, security parameter : t
Output : “prime” or “composite”

1. Write $n-1 = 2^s r$ such that r is odd.
2. For i =1 to t
   - 2.1 Choose random int. a, $2 \leq a \leq n-2$
   - 2.2 Compute $y=a^r \bmod n$
   - 2.3 If $y \neq 1$ and $y \neq n-1$ then
     - j=1
     - while $j \leq s-1$ and $y \neq n-1$ do
       - compute $y = y^2 \bmod n$
       - If y=1 then return(“composite”)
       - j=j+1
     - If $y \neq n-1$ then return(“composite”)
3. Return(“prime”)

Factorization(I)

- **Trial Division**
  - For given n, divide n by every odd integer upto $\sqrt{n}$
  - if $n < 10^{12}$, reasonable. Otherwise need to use sophisticated tech.

Factorization(II)

Pollard's p-1 method

Input : composite int. n that is not a prime power.

Output : Non-trivial factor d of n

1. Select smoothness bound B
2. Select random int. a, 2£ a £ n-1, compute d = gcd(a,n). If d ³ 2 then return(d)
3. For each prime q £ B do
   3.1 Compute l=ë ln n / ln qû
   3.2 Compute a = a^{p^l} mod n
4. Compute d = gcd(a-1,n)
5. If d=1 or d=n, then terminate with failure. Otherwise, return(d)

Factorization(III)

- William's p+1 method
- Quadratic Sieve : O(exp(1+o(1))sqrt{ln n ln ln n})
- Elliptic Curve : O(exp(1+o(1))sqrt{2 ln n ln ln n})
- Number Field Sieve: O(exp(1.92 +o(1))(ln n)$^{1/3}$ (ln ln n)$^{2/3}$)
- Continued Fraction etc

o(1) : ft of n that approach 0 as n ->¥

RSA Challenge

Digits	Year	MIPS-year	Algorithm
RSA-100	'91.4.	7	Q.S.
RSA-110	'92.4.	75	Q.S
RSA-120	'93.6.	830	Q.S.
RSA-129	'94.4.(AC94)	5000	Q.S.*3
RSA-130	'96.4.(AC96)	?	NFS
RSA-140	'99.2 (AC99)	?	NFS

*** MIPS : 1 Million Instruction Per Second for 1 yr = 3.1 x 10^{13} instruction.**
*** 2: The Magic Words are Squeamish Ossifrage.**

33

Choosing p and q for RSA Scheme

(1) |p-q| must be small.

(2) (p ± 1) and (q ± 1) have large prime factors p'_+, p'_- and q'_+, q'_-

(3) (p'_+ ± 1), (p'_- ±1), (q'_+ ± 1) and (q'_- ±1) have large prime factor

(4) gcd(p-1, q-1) has large value

34