Network Security

5. Public Key Cryptography

Private-Key Cryptography

- traditional **private/secret/single key** cryptography uses **one** key
- shared by both sender and receiver
- if this key is disclosed communications are compromised
- also is **symmetric**, parties are equal and hence does not protect sender from receiver forging a message & claiming is sent by sender

Public-Key Cryptography

- probably most significant advance in the 3000 year history of cryptography
- uses **two** keys – a public & a private key
- **asymmetric** since parties are **not** equal
- uses clever application of number theoretic concepts to function
- complements **rather than** replaces private key cryptography

Why Public-Key Cryptography?

- Developed to address two key issues:
  - **Key distribution** – how to have secure communications in general without having to trust a KDC with your key
  - **Digital signatures** – how to verify a message comes intact from the claimed sender
- Public invention due to W. Diffie & M. Hellman at Stanford University in 1976

Public-Key Cryptography

- **public-key/two-key/asymmetric** cryptography involves the use of **two** keys:
  - a **public-key**, which may be known by anybody, and can be used to **encrypt messages**, and **verify signatures**
  - a **private-key**, known only to the recipient, used to **decrypt messages**, and **sign** (create) **signatures**

Public-Key Cryptography

- is **asymmetric** because
  - those who encrypt messages or verify signatures **cannot** decrypt messages or create signatures

Public-Key Characteristics

- Public-Key algorithms rely on two keys with the characteristics that it is:
  - computationally infeasible to find decryption key knowing only algorithm & encryption key
  - computationally easy to en/decrypt messages when the relevant (en/decrypt) key is known
  - either of the two related keys can be used for encryption, with the other used for decryption (in some schemes)

Public-Key Applications

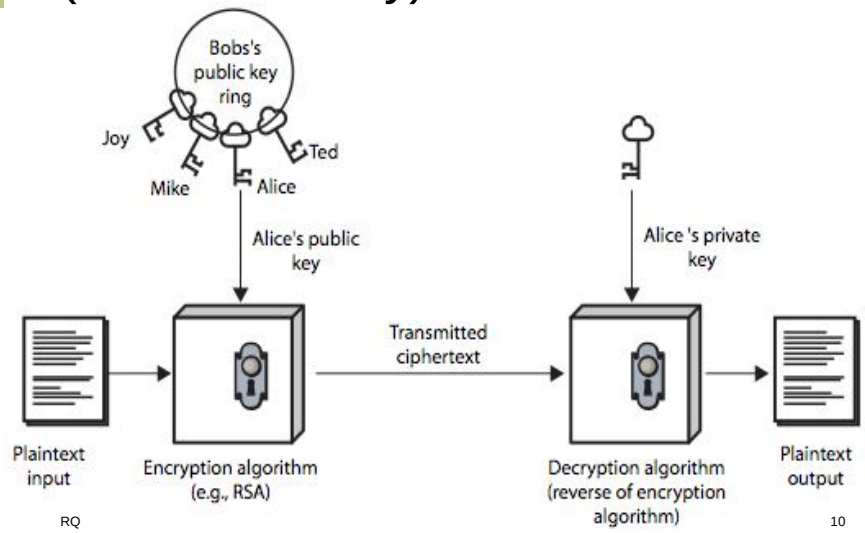
- Can classify uses into 3 categories:
  - Encryption/decryption (provide secrecy)
  - Digital signatures (provide authentication)
  - Key exchange (of session keys)
- Some algorithms are suitable for all uses, others are specific to one

Algorithm	Encryption/Decryption	Digital Signature	Key Exchange
RSA	Yes	Yes	Yes
Elliptic Curve	Yes	Yes	Yes
Diffie-Hellman	No	No	Yes
DSS	No	Yes	No

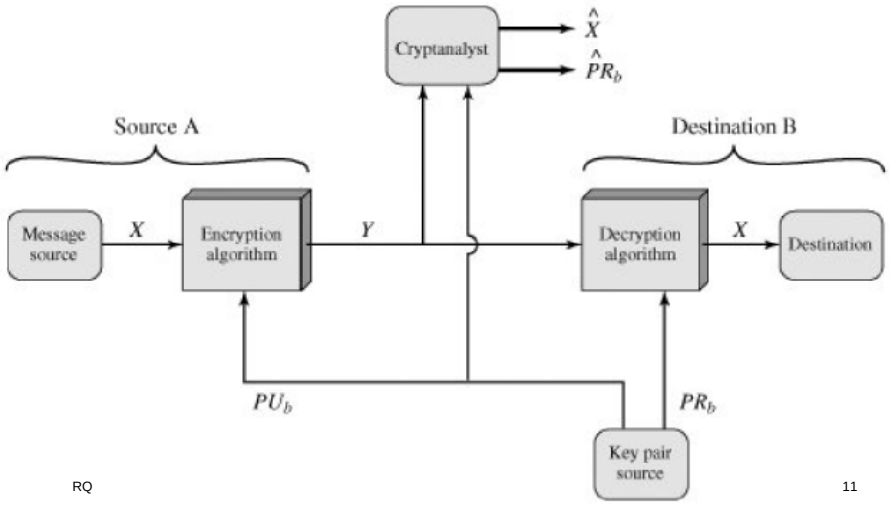
Scenarios

- Sender encrypts using public key of receiver and receiver decrypts using his own private key. Confidentiality is provided since no one else can decrypt the message (Private key is not accessible)
- Sender encrypts using his own private key and receiver decrypts using the senders public key. Authentication is provided since no other user can encrypt the message other than the legitimate sender (His private key is owned by him only)

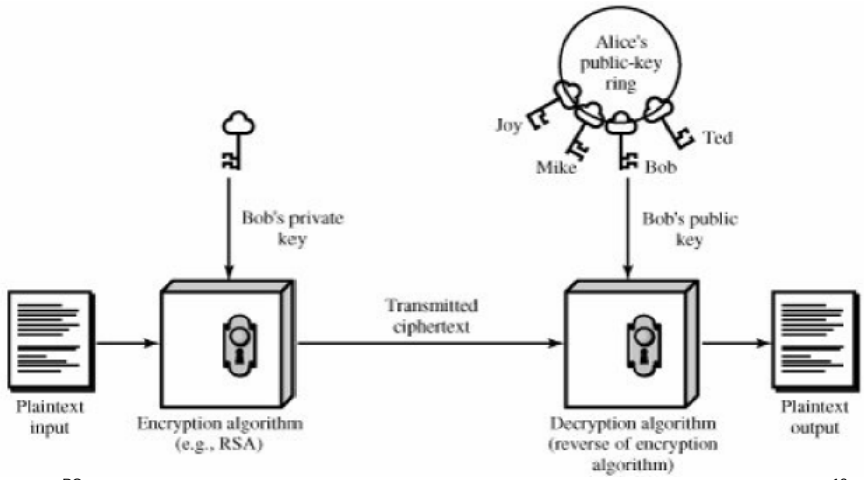
Public-Key Cryptography (Confidentiality)



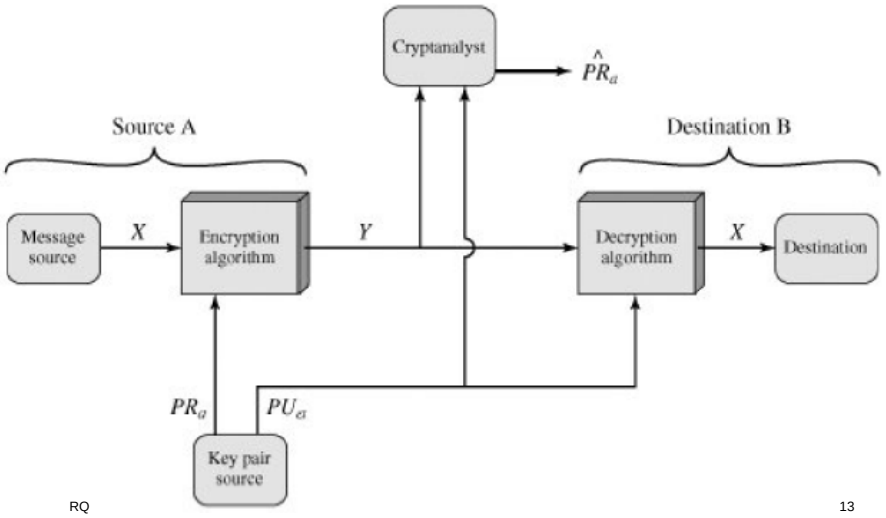
Public-Key Cryptography (Confidentiality)



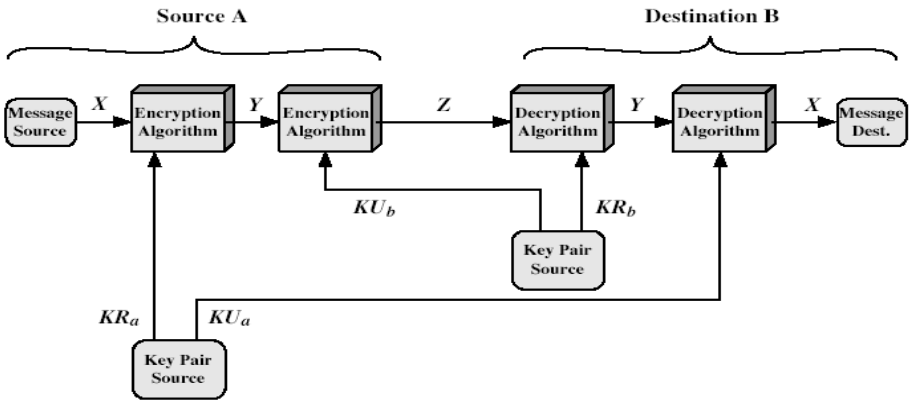
Public-Key Cryptography (Authentication)



Public-Key Cryptography (Authentication)



Public-Key Cryptosystems: Authentication and Secrecy



Security of Public Key Schemes

- like private key schemes brute force **exhaustive search** attack is always theoretically possible
- but keys used are too large (>512bits)
- security relies on a **large enough** difference in difficulty between **easy** (en/decrypt) and **hard** (cryptanalyse) problems
- more generally the **hard** problem is known, its just made too hard to do in practise
- requires the use of **very large numbers**
- hence is **slow** compared to private key schemes

Modular Arithmetic and Prime Numbers

Modular Arithmetic

- The expression $y = x \bmod n$ can be written in the following manners:

$$y \equiv x \bmod n$$
$$y \bmod n = x \bmod n$$
$x \equiv y \iff x$ is congruent to y. Or y is the copy of x in $mod\ n$ arithmetic.

- Hence $5 \equiv 12, 19, 26, \ldots$ in $mod\ 7$ arithmetic.
- The remainder is always the original copy in the congruency set.
- Since $(13 \times 15)\ mod\ 8 = 195\ mod\ 8 = 3$, hence $13\ mod\ 8 = 5$ and $15\ mod\ 8 = 7$. The copy of 13 is 5 and the copy of 15 is 7. Hence, $(5 \times 7)\ mod\ 8 = 3$

Remainder of a Large Number

- For instance, we want to evaluate $7^{34}\ mod\ 9$. First represent 34 in binary. $34 \Rightarrow 100010$. Now we have to do 6 steps (i.e. no. of bits)

1. $7^1\ mod\ 9 = 7$
2. $7^2\ mod\ 9 = 4$
3. $7^4\ mod\ 9 = (7^2 \times 7^2)\ mod\ 9 = (4 \times 4)\ mod\ 9 = 7$
4. $7^8\ mod\ 9 = (7^4 \times 7^4)\ mod\ 9 = (7 \times 7)\ mod\ 9 = 4$
5. $7^{16}\ mod\ 9 = (7^8 \times 7^8)\ mod\ 9 = (4 \times 4)\ mod\ 9 = 7$
6. $7^{32}\ mod\ 9 = (7^{16} \times 7^{16})\ mod\ 9 = (7 \times 7)\ mod\ 9 = 4$

Now looking at the positions of 1, (i.e. 6th and 2nd), we take those values. Hence:

$$(7^{34})\ mod\ 9 = (7^2 \times 7^{32})\ mod\ 9 = (4 \times 4)\ mod\ 9 = 7\ .$$

Prime Numbers

- Prime numbers only have divisors of 1 and self
  - They cannot be written as a product of other numbers
  - Note 1 is a non prime number
- List of prime number less than 200 is:

```
2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59
61 67 71 73 79 83 89 97 101 103 107 109 113
127 131 137 139 149 151 157 163 167 173 179
181 191 193 197 199
```


Relatively Prime Numbers & GCD

- Two numbers ***a*** **and** ***b*** are **relatively prime** if have **no common divisors** apart from 1
  - e.g. 8 & 15 are relatively prime since factors of 8 are 1,2,4,8 and of 15 are 1,3,5,15 and 1 is the only common factor
- Conversely can determine the greatest common divisor by comparing their prime factorizations and using least powers

RSA

RSA

- by Rivest, Shamir & Adleman of MIT in 1977
- best known & widely used public-key scheme
- based on exponentiation in a finite (Galois) field over integers modulo a prime
  - nb. exponentiation takes $O((\log n)^3)$ operations (easy)
- uses large integers (eg. 1024 bits)
- security due to cost of factoring large numbers
  - nb. factorization takes $O(e^{\log n \log \log n})$ operations (hard)

RSA Key Setup

- each user generates a public/private key pair by:
- selecting two large primes at random – **p,q**
- computing their system modulus
  - **n=p.q** and **ø(n)=(p-1)(q-1)**
- selecting at random the encryption key **e**
  - where **1<e<ø(n), gcd(e,ø(n))=1**
- solve following equation to find decryption key **d**
  - **e.d = 1 mod ø(n)** and **0≤d≤N**
- publish their public encryption key: KU={e,n}
- keep secret private decryption key: KR={d,n}
  or KR={d,p,q}

RSA Use

- to encrypt a message M the sender:
  - obtains **public key** of recipient KU={e,n}
  - computes: $C=M^e \mod n$, where $0 \le M < n$
- to decrypt the ciphertext C the owner:
  - uses their private key KR={d,p,q}
  - computes: $M=C^d \mod n$
- note that the message M must be smaller than the modulus n (block if needed)

RSA Example

1. Select primes: $p=17$ & $q=11$
2. Compute $n = pq = 17 \times 11 = 187$
3. Compute $\phi(n)=(p-1)(q-1)=16 \times 10=160$
4. Select e : $\gcd(e,160)=1$; choose $e=7$
5. Determine d: $de=1 \pmod{160}$ and $d < 160$
   Value is d=23 since $23 \times 7=161= 10 \times 160+1$
6. Publish public key $KU=\{7,187\}$
7. Keep secret private key $KR=\{23,17,11\}$

RSA Example cont

- sample RSA encryption/decryption is:
- given message $M = 88$ (nb. $88<187$)
- encryption:
  $C = 88^7 \bmod 187 = 11$
- decryption:
  $M = 11^{23} \bmod 187 = 88$

Exponentiation

- can use the Square and Multiply Algorithm
- a fast, efficient algorithm for exponentiation
- concept is based on repeatedly squaring base
- and multiplying in the ones that are needed to compute the result
- look at binary representation of exponent
- only takes $O(\log_2 n)$ multiples for number n
  - eg. $7^5 = 7^4.7^1 = 3.7 = 10 \bmod 11$
  - eg. $3^{129} = 3^{128}.3^1 = 5.3 = 4 \bmod 11$

RSA Key Generation

- users of RSA must:
  - determine two primes at random - p, q
  - select either e or d and compute the other
- primes p,q must not be easily derived from modulus N=p.q
  - means must be sufficiently large
  - typically guess and use probabilistic test
- exponents e, d are inverses, so use Inverse algorithm to compute the other

RSA Security

- three approaches to attacking RSA:
  - brute force key search (infeasible given size of numbers)
  - mathematical attacks (based on difficulty of computing ø(N), by factoring modulus N)
  - timing attacks (on running of decryption)

Summary

- have considered:
  - principles of public-key cryptography
  - RSA algorithm, implementation, security