RSA Algorithm Lecture Note

Private-Key Cryptography

- So far all the cryptosystems discussed have been private/secret/single key (symmetric) systems. All classical, and modern block and stream ciphers are of this form.
- traditional **private/secret/single key** cryptography uses **one** key
- Key is shared by both sender and receiver
- if the key is disclosed communications are compromised

Public-Key Cryptography

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

Public-Key Cryptography

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

- **Public Key Encryption**
  - Public key encryption, commonly known asymmetric encryption, uses two different keys, a public key known by all and a private key known by only the sender and the receiver.
  - Both the sender and the receiver own a pair of keys, one public and the other a closely guarded private one. To encrypt a message from sender A to receiver B, both A and B must create their own pairs of keys.

- Then A and B publicize their public keys – anybody can acquire them. When A is to send a message M to B, A uses B’s public key to encrypt M. On receipt of M, B then uses his or her private key to decrypt the message M. As long as only B, the recipient, has access to the private key, then A, the sender, is assured that only B, the recipient, can decrypt the message.
- This ensures data confidentiality.
- Data integrity is also ensured because for data to be modified by an attacker it requires the attacker to have B’s, the recipient’s private key. Data confidentiality and integrity in public key encryption is also guaranteed.

Why Public-Key Cryptography

- developed to address two key issues:
  - **key distribution** – how to have secure communications in general without having to trust insecure systems
  - **digital signatures** – how to verify a message comes intact from the claimed sender

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

Public-Key Applications

- can classify uses into 3 categories:
  - **encryption/decryption** (provide secrecy)
  - **digital signatures** (provide authentication)
  - **key exchange**

Security of Public Key Schemes

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

Some Facts on Number theory

- Next we discuss two Public-Key Systems that rely more upon number theory. We will need a preliminary result.
- **Theorem (Fermat's Little Theorem)**: If p is a prime and a is not a multiple of p, then $a^{p-1} \cong 1 \pmod{p}$.
- The following result is from elementary number theory.

Some Facts on Number theory

- **Theorem:** *Let p be a prime. There exists a primitive root modulo p.*
- This leads us to the following computational problem.
- **Example**: Compute the remainder of 7^{103} when divided by 17.
- **Solution**
- By fermats theorem, we have . $7^{16} \cong 1$ mod17. Thus 7^{103} = *$7^{116*16+7}$= 7^{7} =12 MOD 17*
-

DISCRETE LOGARITHM (DL):

- Let p be a prime and g and k be nonzero integers in Z_p and suppose k $\cong g^x$ (mod p).
- The problem of finding x is called the discrete logarithm problem.
- We can denote it $x = \log_g k$, where g is a primitive root.

RSA

- Stands for **Rivest, Shamir & Adleman** **.**
- **Proposed arround** 1977
- best known & widely used public-key scheme
- based on exponentiation in a finite (Galois) field over integers modulo a prime
- uses large integers (eg. 1024 bits)
- security due to cost of factoring large numbers

RSA Key Setup

- each user generates a public/private key pair by:
- selecting two large primes at random - p, q
- computing their system modulus $n=p.q$
  - note $\varnothing(n)=(p-1)(q-1)$
- selecting at random the encryption key e
    - where $1<e<\varnothing(N)$, $\gcd(e,\varnothing(N))=1$
- solve following equation to find decryption key d
  - $e.d=1 \bmod \varnothing(n)$ and $0\leq d\leq n$
- publish their public encryption key: {e,n}
- keep secret private decryption key: {d,p,q}

Encryption using RSA

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

Why RSA Works?

- Can show that RSA works as a direct consequence of Euler's Theorem.
- because of Euler's Theorem:
- $a^{\varnothing(n)} \bmod n = 1,$ where $\gcd(a,n)=1$
- in RSA , we have $n=p.q$
  - $\varnothing(n)=(p-1)(q-1)$
  - carefully chosen e & d to be inverses $\bmod \varnothing(n)$
  - hence $e.d=1+k.\varnothing(n)$ for some k
- hence :
  $C^d = (M^e)^d = M^{1+k.\varnothing(N)} = M^1.(M^{\varnothing(n)})^k = M^1.(1)^k = M^1 = M \bmod n$

RSA Example

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

RSA Example cont

- sample RSA encryption/decryption is:
- given message `M = 88` (nb. `88<187`)
- encryption:

  $C = 88^7 \bmod 187 = 11$
- decryption:

  $M = 11^{23} \bmod 187 = 88$

RSA Security

- brute force key search (infeasible given size of numbers)

RSA signature generation and verification

- RSA may be used directly as a digital signature scheme
- Choose large primes p and q
- Choose e such that gcd(e, (p-1)(q-1))=1, 1<e<p-1)(q-1)
- Choose d such that ed=1 mod(p-1)(q-1), 1<d<p-1)(q-1)
- Then RSA scheme with public key (n, e), (d,p,q)}
- to **sign** a message M, compute: $S = M^{d}$(mod pq)
- to **verify** a signature, compute: $M = S^{e}$(mod pq) = $M^{e.d}$(mod pq) = M(modpq)

RSA signature generation and verification

- RSA can be used both for encryption and digital signatures, simply by reversing the order in which the exponents are used: the secret exponent (d) to create the signature, the public exponent (e) for anyone to verify the signature. Everything else is identical.

RSA signature generation and verification

Example: Try to communicate with your friend by exchanging with your signature.

Use primes p and q with p=5, q=7, n = 35 to send the message "the first letter of your name} over the alphabet Z_{26}.

RSA signature generation and verification

- **Example:** let $e = 5$; d: $ed = 5d=1 \bmod 24$ => $d = 5$. Public key: (35,5) Private key: d=5
- Now my name starts at J, which is 9 in Z_{26}.
- Thus signature $s=9^d=9^5 \bmod 35=4$
- Your friend has to varify the signature as follows
- message $m = s^5 \bmod 35 = 4^5 \bmod 35= 9$ [A, Z].