RSA Algorithm

Number Theory in Cryptography

Terminology: Two parties **Alice** and **Bob** want to communicate securely s.t. a third party **Eve** who intercepts messages cannot learn the content of the messages.

Symmetric Cryptosystems: Alice and Bob share a secret. Only they know a secret key K that is used to encrypt and decrypt messages. Given a message M, Alice encodes it (possibly with padding) into m, and then sends the ciphertext $encrypt(m, K)$ to Bob. Then Bob uses K to decrypt it and obtains $decrypt(encrypt(m, K), K) = m$.

Example: AES.

Public Key Cryptosystems: Alice and Bob do a-priori **not** share a secret. How can they establish a shared secret when others are listening to their messages?

Idea: Have a two-part key, i.e., a key pair. A public key that is used to encrypt messages, and a secret key to decrypt them. Alice uses Bob's public key to encrypt a message (everyone can do that). Only Bob can decrypt the message with his secret key.

Description of RSA: Key generation

- Choose two distinct prime numbers p and q. Numbers p and q should be chosen at random, and be of similar bit-length. Prime integers can be efficiently found using a primality test.
- Let $n = pq$ and $k = (p-1)(q-1)$. (In particular, $k = |Z_n^*|$).
- Choose an integer e such that $1 < e < k$ and $gcd(e, k) = 1$; i.e., e and k are coprime.
  e (for encryption) is released as the public key exponent.
  (e must not be very small.)
- Let d be the multiplicative inverse of e modulo k, i.e., $de \equiv 1(\mod k)$. (Computed using the extended Euclidean algorithm.) d (for decryption) is the private key and kept secret.

The public key is (n, e) and the private key is (n, d).

RSA: Encryption and Decryption

Alice transmits her public key (n, e) to Bob and keeps the private key secret.

Encryption: Bob then wishes to send message M to Alice. He first turns M into an integer m, such that $0 \leq m < n$ by using an agreed-upon reversible protocol known as a padding scheme. He then computes the ciphertext c corresponding to

$$c \equiv m^e \mod n$$

This can be done quickly using the method of exponentiation by squaring. Bob then transmits c to Alice.

Decryption: Alice can recover m from c by using her private key exponent d via computing

$$m \equiv c^d \mod n$$

Given m, she can recover the original message M by reversing the padding scheme.

The RSA Algorithm

To generate a key pair:

– Pick large primes p and q (do not disclose them)

– Let n = p*q

–For the public key, choose e that is relatively prime to ø(n)=(p-1)(q-1).

public key = <e,n>

– For private key, find d that is the multiplicative inverse of e mod ø(n), i.e., e*d

Using RSA

Given pubKey = <e, n> and privKey = <d, n>

If Message = m

Then:

encryption: $c = m^e \bmod n$, $m < n$

decryption: $m = c^d \bmod n$

signature: s = md mod n, m < n

verification: m = se mod n

Example of RSA (1)

Choose p = 7 and q = 17. Compute n = p*q= 119.

Compute f(n)=(p-1)(q-1)=96.
Select e = 5, (a relatively prime to f(n).) Compute d = _77_such that e*d=1 mod f(n).

- Public key: <5,119>
- Private key: <77,119>
- Message = 19
- Encryption: 19^5 mod 119 = 66
- Decryption: 66^{77} mod 119 = 19

Example of RSA (2)

p = 7, q = 11, n = 77

Alice chooses e = 17, making d = 53

Bob wants to send Alice secret message HELLO (07 04 11 11 14)

– $07^{17} \bmod 77 = 28$; $04^{17} \bmod 77 = 16$

– $11^{17} \bmod 77 = 44$; – $11^{17} \bmod 77 = 44$

– $14^{17} \bmod 77 = 42$

- Bob sends **28 16 44 44 42**

Example of RSA (3)

Alice receives **28 16 44 44 42**

Alice uses private key, d = 53, to decrypt message:

– $28^{53} \bmod 77 = 07$; $16^{53} \bmod 77 = 04$

– $44^{53} \bmod 77 = 11$; $44^{53} \bmod 77 = 11$

– $42^{53} \bmod 77 = 14$

- Alice translates **07 04 11 11 14** to ***HELLO***

No one else could read it, as only Alice knows her private key (needed for decryption)