RSA example

Key Generation	
Select p, q	p and q both prime, $p \neq q$
Calculate $n = p \times q$	
Calculate $\phi(n) = (p-1)(q-1)$	
Select integer e	$\gcd(\phi(n), e) = 1;\ 1 < e < \phi(n)$
Calculate d	$de \bmod \phi(n) = 1$
Public key	$KU = \{e, n\}$
Private key	$KR = \{d, n\}$

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

RSA use

modular exponentiation

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

Encryption	
Plaintext:	$M < n$
Ciphertext:	$C = M^e \pmod{n}$

Decryption	
Ciphertext:	C
Plaintext:	$M = C^d \pmod{n}$

$$\text{Plaintext} \xrightarrow{C = M^e \bmod n} \text{Ciphertext}$$

`pk={e,n}` `sk={d,p,q}` $n = p \cdot q$

$$M = C^d \bmod n$$

RSA example continue

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

  n

- encryption:

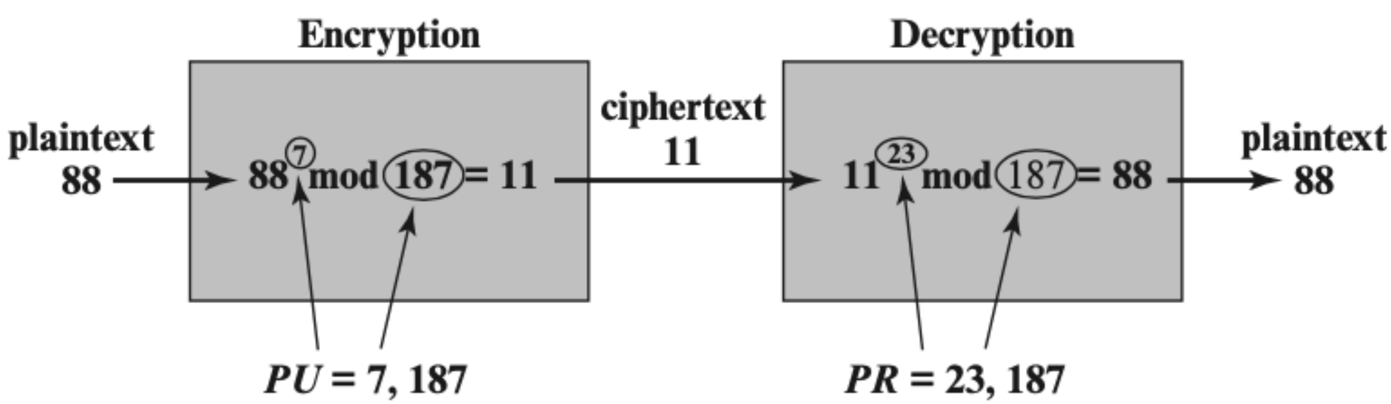
  $C = 88^7 \bmod 187 = 11$

- decryption:

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

  c^d n

Example of RSA algorithm



RSA key generation ← security.

- users of RSA must:
  - determine two primes at random - p, q ↗ large → keep secret
  - select either e or d and compute the other

→ $n \to \phi(n) \to e \to d$

$e \cdot d \equiv 1 \mod \phi(n)$

- primes p, q must not be easily derived from modulus n=p.q ↙ factorization NP hard → prime.
  - means must be sufficiently large
  - typically guess and use probabilistic test

$pk = \{e, n\} \not\Rightarrow \phi(n) = (p-1)\cdot(q-1)$

$= pq - p - q + 1$

n

- exponents e, d are inverses, so use Inverse algorithm to compute the other

$\phi(n)$

Correctness of RSA

coprime,

- Euler's theorem: if gcd (M, n) = 1, then $M^{\phi(n)} = 1$ mod n. Here ϕ(*n*) is Euler's totient function: the number of integers in {1, 2, . . ., *n*-1} which are relatively prime to *n*. When *n* is a prime, this theorem is just Fermat's little theorem

M' = $C^d \bmod n = M^{ed}$ mod n

$= M^{k\phi(n)+1}$ mod n

$= [M^{\phi(n)}]^k \cdot M \bmod n$

= M mod n

Encryption	
Plaintext:	$M < n$
Ciphertext:	$C = M^e \pmod{n}$

$$M \xrightarrow{C = M^e \bmod n} C \qquad M'$$
$$e \qquad\qquad\qquad d$$

To prove $M' = M$

$$M' = C^d \bmod n$$
$$= (M^e)^d \bmod n$$
$$= M^{ed} \bmod n \quad ①$$

$$e \cdot d \equiv 1 \bmod \phi(n), \quad ②$$

By definition of modular arithmetic

$$② \Rightarrow \quad ed = 1 + k \cdot \phi(n) \quad k \in \mathbb{Z} \quad ③$$

$$① = M^{k \cdot \phi(n) + 1} \bmod n$$
$$= M^{k \cdot \phi(n)} \cdot M \bmod n$$
$$= (M^{\phi(n)})^k \cdot M \bmod n \quad ④$$

By Euler's theorem

$$\because \text{ if } \gcd(M, \phi(n)) = 1 \Rightarrow$$
$$M^{\phi(n)} \bmod n = 1$$

if $M << n$, n is large number

$$\Rightarrow M^{\phi(n)} \bmod n = 1 \quad ⑤$$

$$④ = (1)^k \cdot M \bmod n$$
$$M' = M$$

modular exponentiation

Attack approaches

Cl $\xrightarrow{t_1}$ Server $t_1 + t_2$
$\xleftarrow{t_2}$

time variance, 0 1 info leakage,

sound power usage, RF radiation teletype

$n = p \cdot q$

- **Mathematical attacks**: several approaches, all equivalent in effort to factoring the product of two primes. The defense against mathematical attacks is to use a large key size.

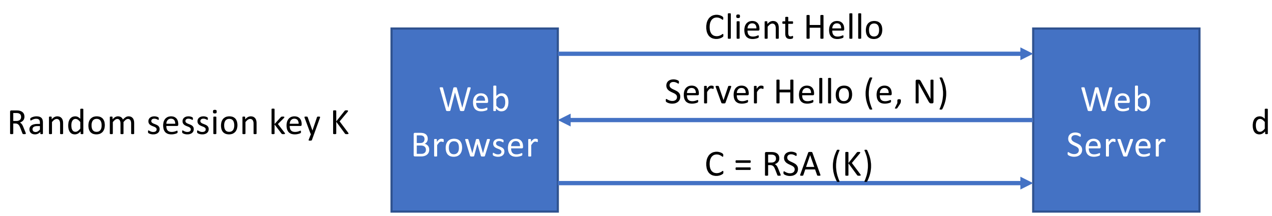
side channel attack, gain ~~from~~ info from implementation

- **Timing attacks**: These depend on the running time of the decryption algorithm

random $\leftarrow r^e \cdot M^e$ TrM/Vi Meta Data Signature

- **Chosen ciphertext attacks**: this type of attacks exploits properties of the RSA algorithm by selecting blocks of data. These attacks can be thwarted by suitable padding of the plaintext, such as PKCS1 V1.5 in SSL

A simple attack on textbook RSA



- Session-key K is 64 bits. View $K \in \{0,\dots,2^{64}\}$
  - Eavesdropper sees: $C = K^e \pmod N$.
- Suppose $\mathbf{K = K_1 \cdot K_2}$ where $K_1, K_2 < 2^{34}$.
  - Then: $\mathbf{C/K_1^{\,e} = K_2^{\,e}} \pmod N$
- Build table: $C/1^e, C/2^e, C/3^e, \dots, C/2^{34e}$. time: 2^{34}

  For $K_2 = 0,\dots, 2^{34}$ test if $K_2^{\,e}$ is in table. time: $2^{34} \cdot 34$
- Attack time: $\approx 2^{40} << 2^{64}$