

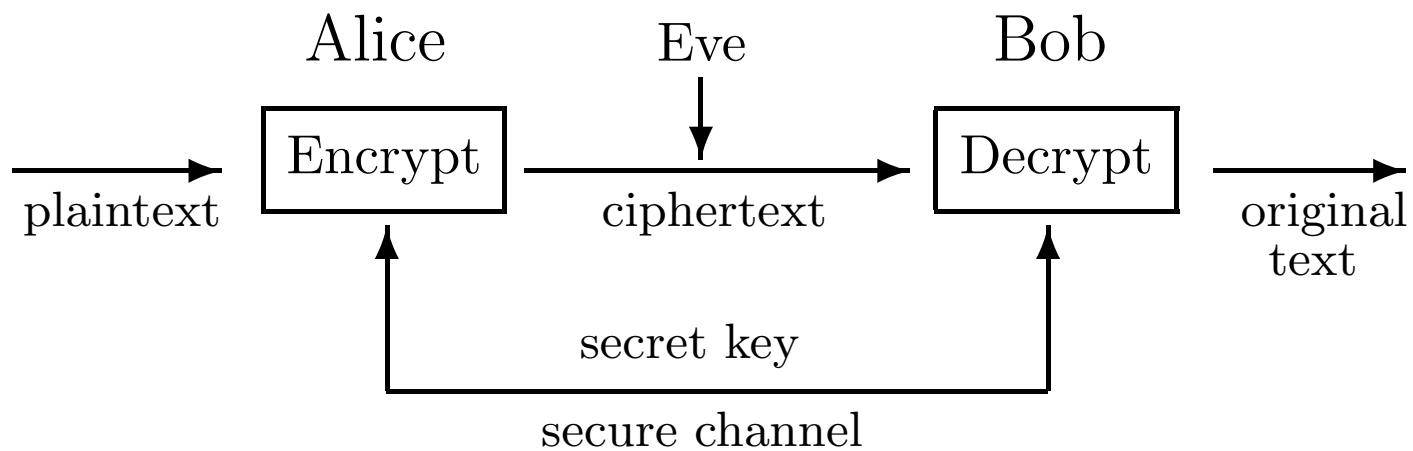
RSA: A public-key cryptosystem

Feryâl Alayont

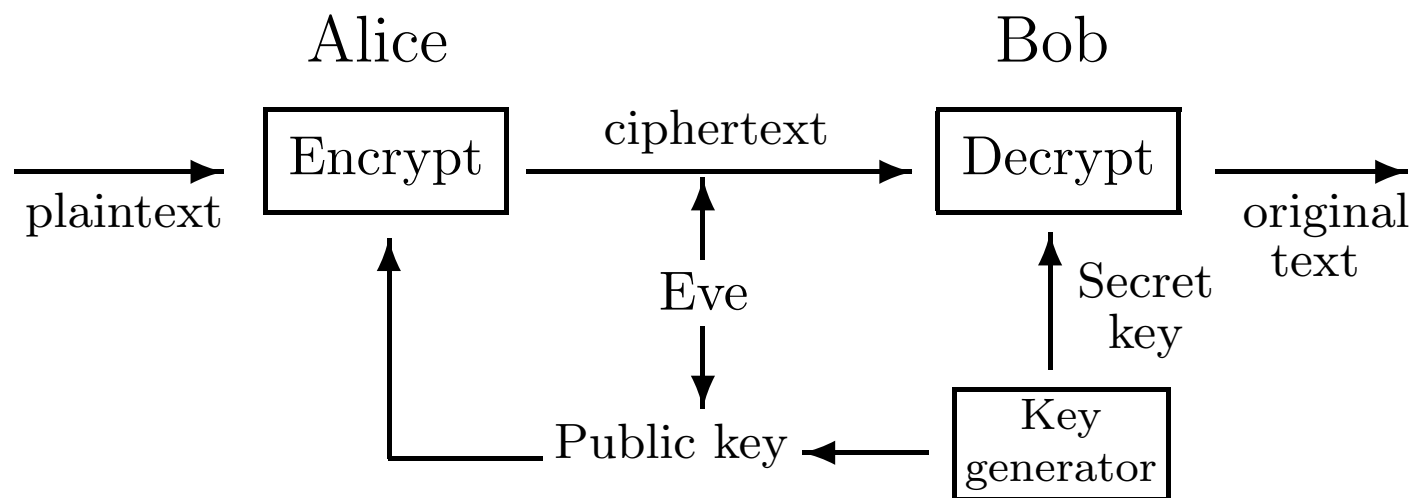
University of Arizona

April 5, 2005

Symmetric-key cryptosystem:



Public-key cryptosystem:



The most common public-key cryptosystem in use: **RSA**

Invented in 1977 by Rivest, Shamir and Adleman from MIT

Usage: to exchange session keys extremely securely

SSH opening lines:

SSH Secure Shell 3.0.0 (Build 203) Copyright (c) 2000-2001 SSH Communications Security Corp - <http://www.ssh.com/>

This copy of SSH Secure Shell is a non-commercial version. This version does not include PKI and PKCS #11 functionality.

This program uses RSA BSAFE©Crypto-C by RSA Security Inc.

Keys and encryption:

Bob: two primes p, q about 155 decimal digits (secret)

$N = pq$ (public)

random e , less than $\varphi(N) = (p - 1)(q - 1)$ (public)

d , the secret key: $ed = 1 \pmod{\varphi(N)}$

Alice: message M , less than N (secret)

ciphertext: $C = M^e \pmod{N}$

Bob: decrypted text: $C^d = M \pmod{N}$

raising to d^{th} ‘undoes’ the e -th power

Example:

$$\text{Bob: } p = 5, q = 11, N = 55$$

$$e = 7 < \varphi(N) = (5 - 1)(11 - 1) = 40$$

$$d = 23, ed = 161 = 1 \pmod{40}$$

$$\text{Alice: } M = 7, C = 7^7 = (7^2)^2 7^2 7 = 28 \pmod{55}$$

$$\text{Bob: } C^{23} = 28^{23} = 7 \pmod{55}$$

Bob recovered the message!

How does RSA work?

Theorem. *For any modulus N and an integer a which does not have a common factor with N , $a^{\varphi(N)} = 1 \pmod{N}$.*

(Euler's Theorem)

RSA decryption: d satisfies

$$ed = 1 + k\varphi(N)$$

for some integer k .

$$\text{mod } N : C^d = (M^e)^d = M^{1+k\varphi(N)} = M (M^{\varphi(N)})^k = M \pmod{N}$$

Conclusion: C^d recovers original message M !

Is RSA computationally feasible?

- Prime number generation

Pick a random large number; test primality using probabilistic tests; repeat many times \rightarrow very likely prime!

PRIMES is in P (2002), but not needed!

- Finding d

Euclidean algorithm can be used efficiently to find the multiplicative inverse of $e \pmod{\varphi(N)}$.

Example:

$$40 - 5 \cdot 7 = 5$$

$$7 - 1 \cdot 5 = 2$$

$$5 - 2 \cdot 2 = 1$$

$$\begin{aligned} \implies 1 &= 5 - 2 \cdot 2 = 5 - 2(7 - 5) \\ &= 3 \cdot 5 - 2 \cdot 7 = \dots = 23 \cdot 7 - 4 \cdot 40 \end{aligned}$$

$$\implies 23 \cdot 7 = 1 + 4 \cdot 40$$

Speed: Takes at most $2 \log_2 e$ steps.

- Exponentiating

Fast exponentiation algorithm.

Example: To find C^{23} :

$$C \xrightarrow{\wedge 2} C^2 \xrightarrow{\wedge 2} C^4 \xrightarrow{\times C} C^5 \xrightarrow{\wedge 2} C^{10} \xrightarrow{\times C} C^{11} \xrightarrow{\wedge 2} C^{22} \xrightarrow{\times C} C^{23}$$

Binary expansion of 23: 10111

$$C^1 \rightarrow C^{10} \rightarrow C^{100} \rightarrow C^{101} \rightarrow C^{1010} \rightarrow C^{1011} \rightarrow C^{10110} \rightarrow C^{10111}$$

Speed: Takes at most $2 \log_2 d$ steps.

Conclusion: Encryption and decryption for authorized users are fast! RSA is computationally feasible.

Attacks on RSA

- Forward search attack

If all possible messages are known, Eve can encrypt all of them and compare with the ciphertext.

Solution: Add random bits at front/back

- Same small e for different users

If same message is sent to three users with same small $e = 3$:

$$C_1 = M^3 \pmod{N_1} \quad C_2 = M^3 \pmod{N_2}$$

$$C_3 = M^3 \pmod{N_3}$$

Using the Chinese Remainder Theorem, find C' such that

$$C' = M^3 \pmod{N_1 N_2 N_3}.$$

$$C' = M^3 \text{ itself since } M^3 < N_1 N_2 N_3.$$

Take cube root of C' in the integers to recover M .

Solution: Avoid small e or add random bits at front/back of messages.

Example: $N_1 = 2 \cdot 13 = 26$, $N_2 = 3 \cdot 11 = 33$,
 $N_3 = 5 \cdot 7 = 35$, $e = 3$.

$$C_1 = M^3 \pmod{26}, \quad C_2 = M^3 \pmod{33}$$
$$C_3 = M^3 \pmod{35}$$

We want $C' \pmod{26 \cdot 33 \cdot 35}$ such that C' satisfies the above three equations.

Find D_1 , D_2 and D_3 such that

$$33 \cdot 35D_1 = 1 \pmod{26}, \quad 26 \cdot 35D_2 = 1 \pmod{33}$$

$$26 \cdot 33D_3 = 1 \pmod{35}$$

$$\begin{aligned} C' &= M^3 \pmod{26} D_1 33 \cdot 35 + M^3 \pmod{33} D_2 26 \cdot 35 \\ &\quad + M^3 \pmod{35} D_3 26 \cdot 33 \pmod{26 \cdot 33 \cdot 35} \end{aligned}$$

Check:

$$C' = M^3 D_3 26 \cdot 33 = M^3 \pmod{35}$$

Similarly $C' = M^3 \pmod{26}$ and $\pmod{33}$. So

$$C' = M^3 \pmod{26 \cdot 33 \cdot 35}$$

- Same N for different users

Suppose e_1 and e_2 are two relatively prime encryption exponents. By using Euclidean algorithm, find f_1, f_2 s.t.

$$1 = f_1 e_1 + f_2 e_2 .$$

Suppose M is sent to both of the users:

$$C_1 = M^{e_1} \quad C_2 = M^{e_2}$$

Eve recovers M :

$$C_1^{f_1} C_2^{f_2} = M^{e_1 f_1 + e_2 f_2} = M \pmod{N}$$

Solution: Each user chooses their own N or adds random bits to messages.

- Factoring N

An easy case: The prime factors p, q are close to each other

$$pq = \left(\frac{p+q}{2}\right)^2 - \left(\frac{p-q}{2}\right)^2$$

If p and q are close, it is easy to express N as a difference of two squares.

Method: Take the smallest integer k greater than \sqrt{N} .

Consider $Q(x) = x^2 - N, x = k, k+1, \dots$

If $Q(x)$ is a square $y^2, N = (x-y)(x+y)$.

Example: $11 \cdot 23 = 253$, $\sqrt{253} \cong 15.9$

$$16^2 - 253 = 3 \text{ not a square}$$

$$17^2 - 253 = 6^2$$

$$253 = (17 - 6)(17 + 6)$$

Easier than trial division with 2, 3, 5, 7, 11.

Example: $13 \cdot 37 = 481, \sqrt{481} \cong 21.9$

$$22^2 - 481 = 3 \quad \text{not a square}$$

$$23^2 - 481 = 3 \cdot 16 \quad \text{not a square}$$

$$24^2 - 481 = 95 \quad \text{not a square}$$

$$25^2 - 481 = 12^2$$

$$481 = (25 - 12)(25 + 12)$$

Not much easier than trial division.

Use a variant:

Basis for most modern factoring algorithms, including Quadratic Sieve and the Number Field Sieve.

Consider $R(x) = x^2 \pmod{N}$, $x = k, k + 1, \dots$

If there are a few x 's for which the product of $R(x)$'s give a square, we are (almost) done!

Example:

$$22^2 = 3 \pmod{481}$$

$$23^2 = 3 \cdot 16 \pmod{481}$$

$$(22 \cdot 23)^2 = (3 \cdot 4)^2 \pmod{481}$$

$$\implies (22 \cdot 23 - 12)(22 \cdot 23 + 12) = 0 \pmod{481}$$

$$\implies (25 - 12)(25 + 12) = 481$$

Other factorization attacks require choosing primes carefully.

Choosing primes:

$p - q$ should not be small.

p and q should be about 512 bits in length.

(Elliptic curve factorization)

$p - 1$ has a large prime factor. (Pollard's $p - 1$ attack)

$p + 1$ has a large prime factor. (Variation of Pollard's attack)

References:

DI Management, *RSA Algorithm* page,
http://www.di-mgt.com.au/rsa_alg.html

Dan Boneh, *Twenty Years of Attacks on the RSA Cryptosystem*,
AMS Notices, 1999.

Whitfield Diffie, *The First Ten Years of Public-Key Cryptography*,
Proceedings of the IEEE, 1988.

James H. Ellis, *The history of Non-Secret Encryption*,
<http://www.cesg.gov.uk/site/publications/media/ellis.pdf>

Paul B. Garrett, *Making, Breaking Codes: An Introduction to
Cryptography*, 2001.

Alfred J. Menezes, Paul C. van Oorschot, Scott A. Vanstone,
Handbook of Applied Cryptography, 2001.

Carl Pomerance, *A tale of Two Sieves*, AMS Notices, 1996.