

Bahan kuliah IF4020 Kriptografi

Algoritma RSA

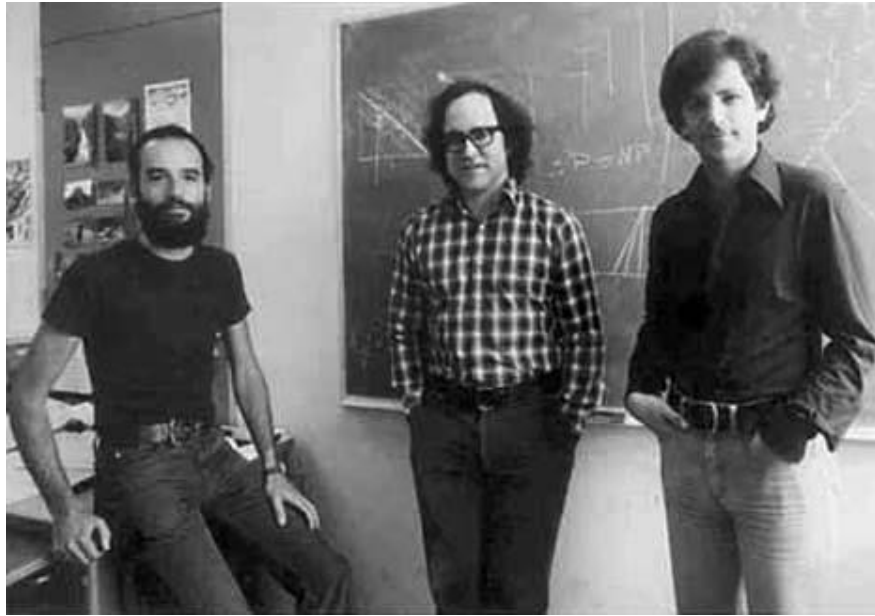
Oleh: Rinaldi Munir

Program Studi Informatika
Sekolah Teknik Elektro dan Informatika
ITB

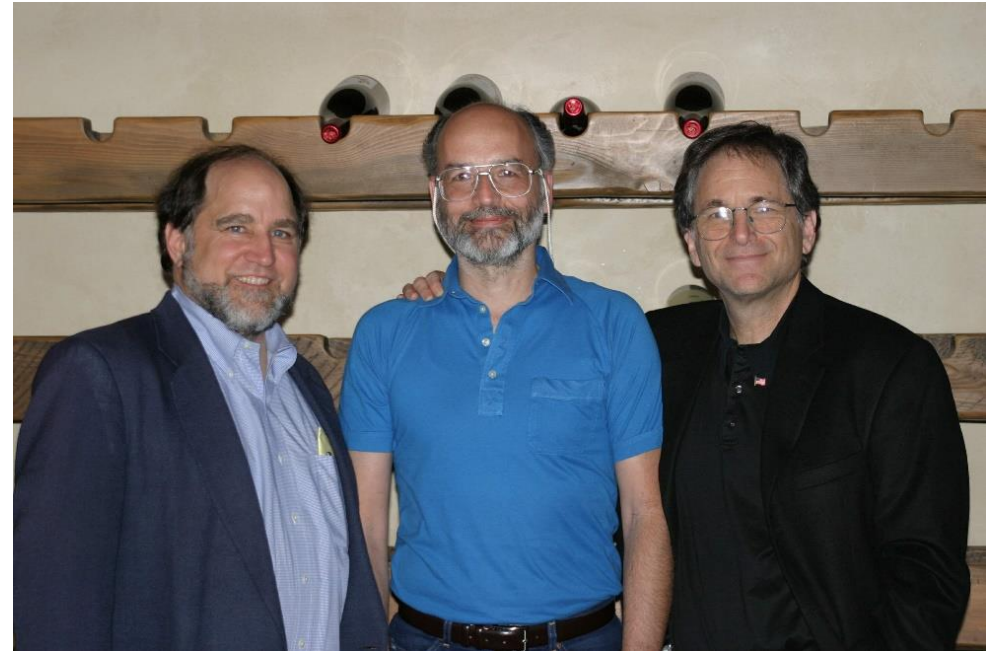
Pendahuluan

- RSA merupakan algoritma kunci-publik yang paling terkenal dan paling banyak aplikasinya.
- Ditemukan oleh tiga peneliti dari *MIT (Massachusetts Institute of Technology)*, yaitu Ronald **Rivest**, Adi **Shamir**, dan Len **Adleman**, pada tahun 1976.
- RSA = Rivest-Shamir-Adleman
- Keamanan algoritma RSA terletak pada sulitnya memfaktorkan bilangan bulat yang besar menjadi faktor-faktor prima.

The authors of RSA: [Rivest](#), [Shamir](#) and [Adleman](#)



dahulu



sekarang

Properti Algoritma RSA

1. p dan q bilangan prima (rahasia)
2. $n = p \cdot q$ (tidak rahasia)
3. $\phi(n) = (p - 1)(q - 1)$ (rahasia)
4. e (kunci enkripsi) (tidak rahasia)

Syarat: $\text{PBB}(e, \phi(n)) = 1$, PBB = pembagi bersama terbesar = gcd

5. d (kunci dekripsi) (rahasia)
 d dihitung dari $d \equiv e^{-1} \pmod{\phi(n)}$
6. m (plainteks) (rahasia)
7. c (cipherteks) (tidak rahasia)

Penurunan Rumus RSA

- Prinsip: Teorema Euler $a^{\phi(n)} \equiv 1 \pmod{n}$
- Syarat:
 1. a harus relatif prima terhadap n
 2. $\phi(n)$ = Totient Euler = fungsi yang menentukan berapa banyak dari bilangan-bilangan $1, 2, 3, \dots, n$ yang relatif prima terhadap n .

Contoh: $\phi(20) = 8$, sebab terdapat 8 buah yang relatif prima dengan 20, yaitu 1, 3, 7, 9, 11, 13, 17, 19.

Jika $n = pq$ adalah bilangan komposit dengan p dan q prima, maka

$$\phi(n) = \phi(p) \phi(q) = (p - 1)(q - 1).$$

$$a^{\phi(n)} \equiv 1 \pmod{n}$$

↓ (pangkatkan kedua ruas dengan k)

$$a^{k\phi(n)} \equiv 1^k \pmod{n}$$

↓

$$a^{k\phi(n)} \equiv 1 \pmod{n}$$

↓ (ganti a dengan m)

$$m^{k\phi(n)} \equiv 1 \pmod{n}$$

↓ (kalikan kedua ruas dengan m)

$$m^{k\phi(n) + 1} \equiv m \pmod{n}$$

- Misalkan e dan d dipilih sedemikian sehingga
$$e \cdot d \equiv 1 \pmod{\phi(n)}$$

atau

$$e \cdot d = k\phi(n) + 1$$

Maka

$$m^{k\phi(n) + 1} \equiv m \pmod{n}$$

↓

$$m^{e \cdot d} \equiv m \pmod{n} \rightarrow (m^e)^d \equiv m \pmod{n}$$

- Enkripsi: $E_e(m) = c = m^e \pmod{n}$

- Dekripsi: $D_d(c) = m = c^d \pmod{n}$

Prosedur Pembangkitan Sepasang Kunci

1. Pilih dua bilangan prima, p dan q
2. Hitung $n = pq$.
3. Hitung $\phi(n) = (p - 1)(q - 1)$.
4. Pilih sebuah bilangan bulat e sebagai kunci publik, e harus relatif prima terhadap $\phi(n)$.
5. Hitung kunci dekripsi, d , dengan persamaan
$$ed \equiv 1 \pmod{\phi(n)} \text{ atau } d \equiv e^{-1} \pmod{\phi(n)}$$

Hasil dari algoritma di atas:

- Kunci publik adalah pasangan (e, n)
- Kunci privat adalah pasangan (d, n)

Enkripsi

1. Nyatakan pesan menjadi blok-blok plainteks: m_1, m_2, m_3, \dots
(syarat: $0 \leq m_i < n - 1$)
2. Hitung blok cipherteks c_i untuk blok plainteks m_i menggunakan kunci publik e dengan persamaan

$$c_i = m_i^e \mathbf{mod} n$$

Dekripsi

1. Misalkan blok-blok cipherteks adalah c_1, c_2, c_3, \dots
2. Hitung kembali blok plainteks m_i dari blok cipherteks c_i menggunakan kunci privat d dengan persamaan

$$m_i = c_i^d \bmod n,$$

Contoh pembangkitan kunci oleh Alice

- Misalkan Alice memilih $p = 47$ dan $q = 71$ (keduanya prima), maka dapat dihitung:

$$n = p \times q = 3337$$

$$\phi(n) = (p - 1) \times (q - 1) = 3220.$$

- Alice memilih kunci publik $e = 79$ (yang relatif prima dengan 3220 karena pembagi bersama terbesarnya adalah 1).
- Nilai e dan n dapat dipublikasikan ke umum.

- Selanjutnya Alice menghitung kunci privat d dengan kekongruenan:

$$ed \equiv 1 \pmod{\phi(n)}$$

d adalah balikan e dalam modulus $\phi(n)$

d dapat dihitung dengan algoritma Euclidean atau dengan rumus:

$$d = \frac{1 + k\phi(n)}{e}$$

Dengan mencoba nilai-nilai $k = 1, 2, 3, \dots$, diperoleh nilai d yang bulat adalah 1019. Ini adalah kunci privat (untuk dekripsi).

- Misalkan Bob akan mengirim plainteks $M = \text{'HELLO ALICE'}$ kepada Alice
- Dengan memisalkan $A = 00, B = 01, \dots, Z = 25$, maka pesan m dikodekan ke dalam *integer* (spasi diabaikan) menjadi

$$M = 07041111140011080204$$

Pecah M menjadi blok yang 4 digit:

$$\begin{array}{ll} m_1 = 0704 & m_4 = 1108 \\ m_2 = 1111 & m_5 = 0204 \\ m_3 = 1400 & \end{array}$$

(Perhatikan, m_i masih terletak di dalam selang $[0, 3337 - 1]$)

- Bob mengenkripsi setiap blok dengan menggunakan kunci public Alice ($e = 79$):

$$c_1 = 704^{79} \bmod 3337 = 328;$$

$$c_2 = 1111^{79} \bmod 3337 = 301;$$

$$c_3 = 1400^{79} \bmod 3337 = 2653;$$

$$c_4 = 1108^{79} \bmod 3337 = 2986;$$

$$c_5 = 204^{79} \bmod 3337 = 1164;$$

Cipherteks: $C = 0328\ 0301\ 2653\ 2986\ 1164$

-
- Bob mengirim cipherteks C kepada Alice

- Alice mendekripsi cipherteks dengan menggunakan kunci privatnya, yaitu $d = 1019$

$$m_1 = 328^{1019} \bmod 3337 = 704 = 0704$$

$$m_2 = 301^{1019} \bmod 3337 = 1111$$

$$m_3 = 2653^{1019} \bmod 3337 = 1400$$

$$m_4 = 2986^{1019} \bmod 3337 = 1108$$

$$m_5 = 1164^{1019} \bmod 3337 = 204$$

- Alice memperoleh kembali plainteks dari Bob

$$M = 07041111140011080204$$

yang dikodekan kembali menjadi

$$M = \text{HELLO ALICE}$$

Keamanan RSA

- Keamanan algoritma *RSA* terletak pada tingkat kesulitan dalam memfaktorkan bilangan bulat n faktor-faktor prima (p dan q), yang dalam hal ini $n = p \times q$.

- Sekali n berhasil difaktorkan menjadi p dan q , maka

$$\phi(n) = (p - 1) \times (q - 1) \text{ dapat dihitung.}$$

Selanjutnya, karena kunci enkripsi e diumumkan (tidak rahasia), maka kunci dekripsi d dapat dihitung dari kekongruenen

$$ed \equiv 1 \pmod{\phi(n)}.$$

- Penemu algoritma *RSA* menyarankan nilai p dan q panjangnya lebih dari 100 digit. Dengan demikian hasil kali $n = p \times q$ akan berukuran lebih dari 200 digit.
- Usaha untuk mencari faktor bilangan 200 digit membutuhkan waktu komputasi selama 4 milyar tahun, sedangkan untuk bilangan 500 digit membutuhkan waktu 10^{25} tahun

(dengan asumsi bahwa algoritma pemfaktoran yang digunakan adalah algoritma yang tercepat saat ini dan komputer yang dipakai mempunyai kecepatan 1 milidetik).

- Algoritma pemfaktoran yang tercepat saat ini memiliki kompleksitas

$$O(\exp(\sqrt[3]{\frac{64}{9} b(\log(b))^2}))$$

untuk bilangan bulat n sepanjang b -bit.

- Hingga saat ini belum ditemukan algoritma pemfaktoran bilangan bulat besar dalam waktu polinomial.
- Fakta inilah yang membuat algoritma *RSA* dianggap masih aman untuk saat ini. Semakin panjang bilangan bulatnya, maka semakin lama waktu yang dibutuhkan untuk memfaktorkannya.

Contoh parameter RSA

- Modulus n sepanjang 1024 bit (setara 300 angka decimal)
- Bilangan prima p dan q masing-masing panjangnya sekitar 154 angka decimal
- Sumber: https://www.di-mgt.com.au/rsa_alg.html

- $n =$
1192941348401695090555272113312556496446065696615276380120674819549430568
5115033380631595703771562029730500011862877084668996911289221224545711806
0574995989517080042105263427376322274266393116193517839570773505632231596
6811219273374739732203125125990612313222509455062600665575382385175753906
21262940383913963

- $p =$
1093376618363257581761151703473066828715579998463222345413874567112127345
6287670008290843302875521274970245314593222946129064538358581018615539828
479146469

- $q =$
1091061696734911023172373407861492264533706088214174896820983422513897601
1179993394299810159736904468554021708289824396553412180514827996444845438
176099727

- Secara umum dapat disimpulkan bahwa RSA hanya aman jika n cukup besar.
- Jika panjang n hanya 256 bit atau kurang, ia dapat difaktorkan dalam beberapa jam saja dengan sebuah komputer *PC* dan program yang tersedia secara bebas.
- Jika panjang n adalah 512 bit atau kurang, ia dapat difaktorkan dengan beberapa ratus komputer

- Tahun 1977, 3 orang penemu *RSA* membuat sayembara untuk memecahkan cipherteks dengan menggunakan *RSA* di majalah *Scientific American*.
- Hadiahnya: \$100
- Tahun 1994, kelompok yang bekerja dengan kolaborasi internet berhasil memecahkan cipherteks hanya dalam waktu 8 bulan.

Kelemahan RSA

- *RSA* lebih lambat daripada algoritma kriptografi kunci-simetri seperti *DES* dan *AES*
- Dalam praktek, *RSA* tidak digunakan untuk mengenkripsi pesan, tetapi mengenkripsi kunci simetri (kunci sesi) dengan kunci publik penerima pesan.
- Pesan dienkripsi dengan algoritma simetri seperti *DES* atau *AES*.
- Pesan dan kunci simetri dikirim bersamaan.
- Penerima mendekripsi kunci simetri dengan kunci privatnya, lalu mendekripsi pesan dengan kunci simetri tersebut.