**Modul 13**

1. **Tujuan**

1. Mampu menjelaskan definisi dan konsep Rivest-Shamir-Adleman (RSA)

2. Mampu memahami konsep dasar Rivest-Shamir-Adleman (RSA)

3. Mampu mengimplementasikan algoritma RSA dalam mengamankan data pada sistem

1. **Dasar teori**
2. **Rivest-Shamir-Adleman (RSA)**

Algoritma RSA adalah algoritma kriptografi asimetris. Asimetris sebenarnya berarti bekerja pada dua kunci yang berbeda yaitu Kunci Publik dan Kunci Pribadi. Seperti namanya menjelaskan bahwa Kunci Publik diberikan kepada semua orang dan Kunci pribadi disimpan secara pribadi. Ini juga dikenal sebagai kriptografi kunci publik karena salah satu kunci dapat diberikan kepada siapa saja. Perusahaan seperti Acer, Asus, HP, Lenovo, dll, menggunakan teknik enkripsi dalam produk mereka.

Enkripsi asimetris melibatkan mekanisme yang disebut Kunci Publik dan Kunci Pribadi. Semua orang di jaringan dapat mengakses kunci publik tetapi kunci pribadi bersifat anonim. Pengguna membuat kunci pribadi menggunakan fungsi.

* Untuk mengenkripsi pesan, seseorang dapat menggunakan kunci publik.
* Kirim pesan melalui saluran. Kunci pribadi dihasilkan di sisi penerima.
* Kunci pribadi digunakan untuk mendekripsi pesan terenkripsi.

Ini didasarkan pada prinsip bahwa faktorisasi prima dari bilangan komposit besar adalah tangguh. Hanya kunci pribadi penerima yang dapat mendekripsi pesan sandi.

.

1. **Fungsionalitas AES**

Ide RSA didasarkan pada kenyataan bahwa sangat sulit untuk melakukan faktorisasi pada bilangan bulat besar. Kunci publik terdiri dari dua bilangan dimana satu bilangan merupakan hasil perkalian dari dua bilangan prima bernilai besar. Sedngkan kunci privat merupakan koprima dari hasil perkalian dua bilangan tersebut. Oleh karena itu kekuatan enkripsi sepenuhnya terletak pada ukuran kunci dan jika kita menggandakan atau melipatgandakan ukuran kunci, kekuatan enkripsi meningkat secara eksponensial. Kunci RSA biasanya panjangnya 1024 atau 2048 bit, tetapi para ahli percaya bahwa kunci 1024 bit dapat rusak dalam waktu dekat. Contoh ilustrasi nya adalah sebagai berikut :

Seorang klien (browser) mengirimkan kunci publiknya ke server dan meminta beberapa data. Maka :

* Server mengenkripsi data menggunakan kunci publik klien dan mengirimkan data terenkripsi.
* Klien menerima data ini dan mendekripsinya.

Karena ini asimetris, tidak ada orang lain kecuali browser yang dapat mendekripsi data meskipun pihak ketiga memiliki kunci publik browser

1. **Kunci Publik dan Privat**

Besaran-besaran yang digunakan pada algoritma RSA:

* p dan q bilangan prima (rahasia)
* n = p × q (tidak rahasia)
* f(n) = (p – 1)(q – 1) (rahasia)
* e (kunci enkripsi) (tidak rahasia)
* d (kunci dekripsi) (rahasia)
* m (plainteks) (rahasia)
* c (cipherteks) (tidak rahasia)

**Kunci Publik**

* Pilih dua bilangan prima. Misalkan P = 53 dan Q = 59. P tidak sama dengan Q.
* Cari bilangan pertama dari kunci Publik. n = P \* Q = 3127.
* Kemudian cari bilangan eksponen kecil, misalkan e. Nilai e harus memenuhi aturan berikut :
  1. Sebuah bilangan bulat.
  2. Tidak menjadi faktor n.
  3. Nilai e harus relatif prima atau koprima dengan n.
  4. Kita tentukan nilai e adalah 3.
* Kunci Publik kami terbuat dari n dan e, yaitu 3127 dan 3.

**Kunci Privat**

* Hitung nilai Φ(n). Rumusnya Φ(n) = (P-1)(Q-1). Maka hasilnya adalah
* Φ(n) = 3016
* Sekarang hitung Private Key, d :
* d = (k\*Φ(n) + 1) / e, dengan mencoba nilai k = 1, 2, 3, ...., untuk mendapatkan nilai d bilangan bulat. Maka nilai k = 3.
* Maka pasangan kunci privat adalah d dan n, yaitu 2011 dan 3127

**Fungsi enkripsi**

1. Ambil kunci publik penerima pesan, e, dan modulus n.
2. Nyatakan plainteks m menjadi blok-blok m1, m2, …, sedemikian sehingga setiap blok merepresentasikan nilai di dalam selang [0, n – 1].
3. Setiap blok mi dienkripsi menjadi blok ci dengan rumus

Ci = (Mi)e mod n.

**Fungsi dekripsi**

1. Setiap blok cipherteks ci didekripsi kembali menjadi blok mi

Mi= (Ci)d mod n.

1. **Enkripsi dan Dekripsi**

* **Enkripsi**

Contoh kita akan melakukan enkripsi karakter pada teks “HI”.

1. Langkah pertama adalah konversi teks menjadi integer. Bisa menggunakan kode ASCII atau pengisian nilai secara manual, misal A = 0, B = 1, C = 2, dst.
2. Hasil konversi adalah H = 8, I = 9.
3. Tentukan 2 nilai untuk generate publik key, X = 32 Y = 56
4. Hitung nilai e dan n menggunakan algoritma generate public key.
5. Enkripsi = (89)e mod n = 1394
6. Hasil akhir enkripsi teks HI adalah 1394

* **Dekripsi**

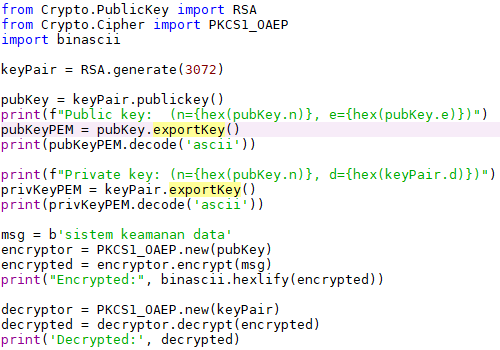
1. Dekrispsi data menggunakan rumus cd mod n, dimana c adalah hasil enkripsi
2. Dekripsi = (1394)d mod n = 89
3. Maka hasil akhir dekripsi karakter 1394 adalah HI
4. **RSA Python Library**

Ada banyak perpustakaan yang tersedia di python untuk enkripsi dan dekripsi pesan, tetapi hari ini kita akan membahas perpustakaan luar biasa yang disebut pycryptodome.

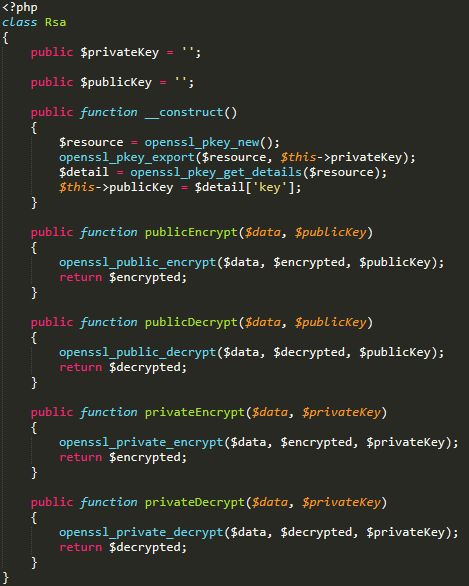
**Apa yang bisa dilakukan Library Algoritma RSA Python :**

* Pembuatan pasangan kunci: menghasilkan kunci pribadi dan kunci publik acak (ukurannya 1024-4096 bit).
* Enkripsi: Ini mengenkripsi pesan rahasia (bilangan bulat dalam rentang [0…key\_length]) menggunakan kunci publik dan mendekripsi kembali menggunakan kunci rahasia.
* Tanda tangan digital: menandatangani pesan (menggunakan kunci pribadi) dan memverifikasi tanda tangan pesan (menggunakan kunci publik).
* Pertukaran kunci: Ini dengan aman mengangkut kunci rahasia yang digunakan untuk komunikasi terenkripsi.

**Contoh Kode Python**



1. **RSA PHP Library**



1. **Faktor Keamanan**

* Kamanan algoritma RSA didasarkan pada sulitnya memfaktorkan bilangan besar menjadi fakto-faktor primanya. Masalah pemfaktoran: Faktorkan n, yang dalam hal ini n adalah hasil kali dari dua atau lebih bilangan prima. Pada RSA, masalah pemfaktoran berbunyi: Faktorkan n menjadi dua faktor primanya, p dan q, sedemikian sehingga n = p × q.
* Penemu algoritma RSA menyarankan nilai p dan q panjangnya lebih dari 100 digit. Dengan demikian hasil kali n = p ´ q akan berukuran lebih dari 200 digit. Menurut Rivest dan kawan-kawan, usaha untuk mencari faktor prima dari bilangan 200 digit membutuhkan waktu komputasi selama 4 milyar tahun, sedangkan untuk bilangan 500 digit membutuhkan waktu 1025 tahun! (dengan asumsi bahwa algoritma pemfaktoran yang digunakan adalah algoritma yang tercepat saat ini dan komputer yang dipakai mempunyai kecepatan 1 milidetik).
* Untunglah algoritma yang paling mangkus untuk memfaktorkan bilangan yang besar belum ditemukan. Selama 300 tahun para matematikawan mencoba mencari faktor bilangan yang besar namun tidak banyak membuahkan hasil. Semua bukti yang diketahui menunjukkan bahwa upaya pemfaktoran itu lura biasa sulit.
* Fakta inilah yang membuat algoritma RSA tetap dipakai hingga saat ini. Selagi belum ditemukan algoritma yang mangkus untuk memfaktorkan bilangan bulat menjadi faktor primanya, maka algoritma RSA tetap direkomendasikan untuk mengenkripsi pesan.

**­­C. TUGAS**

1. Membuat program enkripsi dan dekripsi Algoritma RSA menggunakan PHP atau Python

2. Cari 2 Jurnal berkaitan dengan algoritma RSA, kemudian buatlah review terkait topik yang dibahas pada jurnal tersebut. Tuliskan review masing-masing jurnal di Ms. Word

3. Review harus memuat poin-poin di bawah ini :

a. Judul dan Latar Belakang Masalah

b. Tujuan Penelitian

c. Algoritma yang dipakai beserta alur penelitiannya

d. Ceritakan hasil penelitian pada jurnal tersebut dan kesimpulannya

e. Kelebihan dan kekurangan masing-masing jurnal tersebut

4. Yang dikumpukan :

Kode program, Screenshot hasil program, file review jurnal.

5. Pengumpulan Tugas Praktikum

* **Upload semua file github,** kemudian link dikirimkan ke SPADA
* **Untuk kelas TI E** paling lambat tanggal 29 November 2021 Jam 23.59

**Untuk kelas TI D** paling lambat tanggal 30 November 2021 Jam 23.59

* Format penamaan file SKD\_namakelas\_nim\_nama