Protokol Kriptografi dan Komunikasi yang Aman

charmingnaia@gmail.com
http://cryptonia.wordpress.com/

Pendahuluan

Dokumen ini berisi sebuah tutorial yang sederhana dan mudah dipahami oleh pembaca. Adapun isi dari dokumen ini yaitu panduan mengenai bagaimana menerapkan/mengimplementasikan protokol kriptografi untuk sebuah komunikasi antara dua entitas dimana informasi yang terkandung di dalamnya bersifat sensitif sehingga perlu diciptakan sebuah protokol yang aman dengan menerapkan kriptografi.

Pada umumnya, orang hanya mengamankan pesan agar tidak dapat dibaca oleh pihak yang tidak berhak, teknik ini disebut enkripsi. Padahal, itu saja sebenarnya belum cukup. Perlu diperhatikan pengamanan lebih lanjut yaitu bagaimana kunci/*password* untuk enkripsi itu dibangkitkan, bagaimana proses pendistribusiannya, dan bagaimana cara mengetahui bahwa pesan tersebut berasal dari pihak yang benar dan diterima oleh pihak yang benar pula tanpa terjadi perubahan data di dalam proses transimi. Oleh karena itu, penulis akan memberikan gambaran mengenai rancangan protokol kriptografi dan komunikasi yang aman.

Langkah-langkah yang akan dijelaskan meliputi :

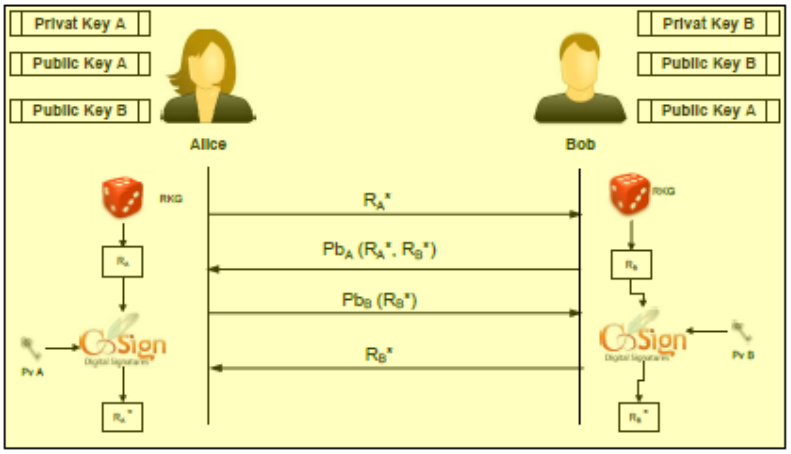
1. Simetrik *Block Cipher* untuk layanan *confidentiality* pesan.
2. Asimetrik *Cipher* untuk distribusi *session key*.
3. *Hash function* untuk layanan *data integrity*.

4. *Random key generator* untuk proses pembangkitan *session key*.

   Otentikasi antar entitas.

Isi

1. Otentikasi antar entitas



Gambar 1. Skema otentikasi

Dua buah entitas yang akan berkomunikasi (Alice dan Bob) membangkitkan pasangan kunci publik dan privat masing-masing. Diasumsikan kedua entitas tersebut saling mengetahui kunci publik satu sama lain. Alice dan Bob membangkitkan sebuah bilangan acak menggunakan *Random Key Generator* (*RKG*) yang mereka miliki. Bilangan acak yang dibangkitkan oleh Alice adalah R_A dan bilangan acak yang dibangkitkan oleh Bob adalah R_B. Kemudian masing-masing bilangan acak tersebut di-*sign* untuk menghasilkan *digital signature* menggunakan kunci privat masing-masing sehingga hasilnya adalah R_A* dan R_B*.

Pada langkah pertama, Alice akan mengirimkan bilangan acak yang sudah di-*sign* tersebut (R_A*) kepada Bob. Kemudian Bob akan mengecek apakah bilangan acak tersebut *valid* milik Alice, dengan cara mengecek *digital signature*-nya menggunakan kunci publik Alice yang sudah dimiliki sebelumnya oleh Bob.

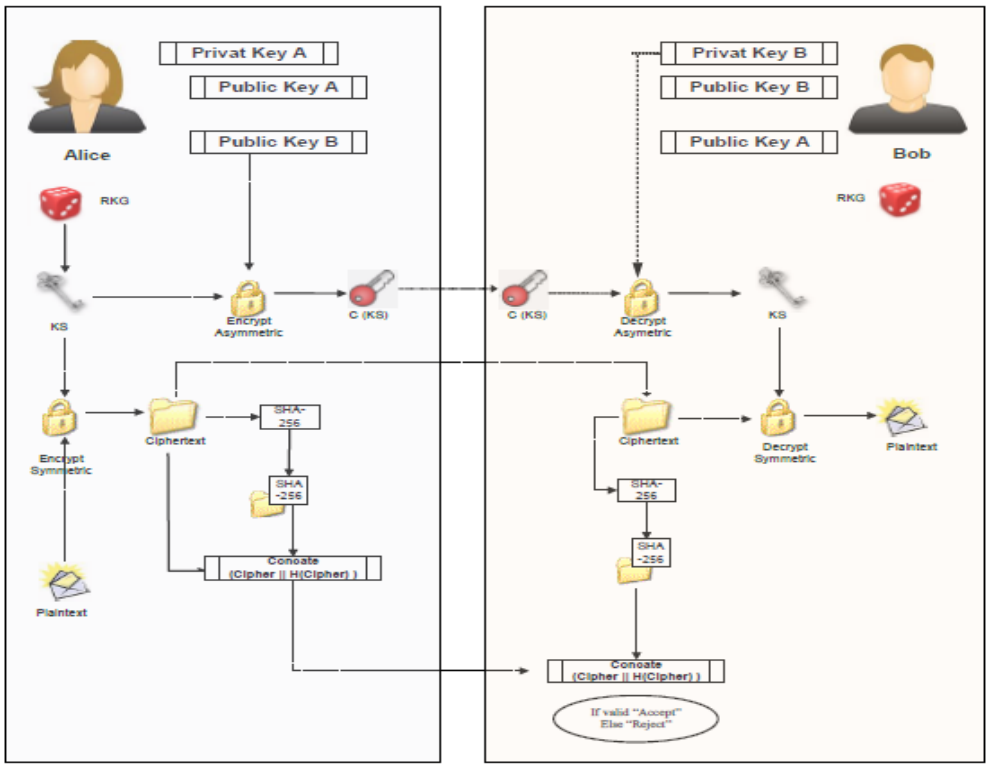
Jika terbukti *valid*, kemudian langkah kedua adalah Bob mengenkripsi kiriman Alice tadi, beserta bilangan acak miliknya yang sudah di-*sign*. Enkripsi dilakukan menggunakan kunci publik milik Alice (P_bA (R_A*, R_B*)) kemudian dikirim kepada Alice. Setelah menerima kiriman dari Bob, Alice akan membuka pesan tersebut menggunakan kunci *private*-nya, sehingga ia akan mendapatkan kembali nilai R_A* yang

ia kirimkan kepada Bob beserta nilai R_B* milik Bob. Kemudian Alice akan membuktikan validitas R_B* menggunakan kunci publik milik Bob yang telah dimiliki Alice.

Langkah ketiga, setelah terbukti *valid*, maka Alice akan mengirimkan kembali R_B* kepada Bob namun dienkripsi terlebih dahulu menggunakan kunci publik Bob ($P_bB(R_B*)$). Setelah diterima oleh Bob, Bob akan membuka pesan dari Alice menggunakan kunci privat miliknya, jika R_B* yang dikirimkan Alice benar maka pada langkah keempat atau yang terakhir pada proses otentikasi ini Bob akan mengirimkan kembali R_B* tersebut kepada Alice untuk meyakinkan bahwa nilai tersebut benar dan Bob tahu nilai yang dikirim Alice tersebut.

Dengan terpenuhinya empat langkah tersebut, maka dapat dipastikan bahwa Alice dan Bob merupakan pihak yang sah yang akan melakukan komunikasi.

2. Skema enkripsi dan dekripsi



Gambar 2. Skema enkripsi dan dekripsi pesan

Langkah awal yang dilakukan adalah salah satu dari Alice atau Bob membangkitkan *Key Session* (*KS*) menggunakan *Random Key Generator* (*RKG*). *KS* ini hanya dibangkitkan sekali dan nantinya akan digunakan untuk enkripsi dan dekripsi pesan. Sebagai contoh, Alice membangkitkan *KS*, kemudian *KS* tersebut di-enkripsi menggunakan algoritma asimetrik menggunakan kunci publik milik Bob dan dikirimkan kepada Bob. Setelah sampai pada Bob, ia akan membuka *KS* terenkripsi menggunakan kunci privat miliknya. *KS* tersebut dijadikan kesepakatan sebagai kunci yang digunakan selama berkomunikasi hingga selesai.

Langkah selanjutnya, pada saat Alice akan mengirim pesan, maka pesan tersebut dienkripsi menggunakan algoritma simetrik dengan kunci *KS* yang sudah dibangkitkan sebelumnya. Kini pesan tersebut telah menjadi sebuah *ciphertext* yang siap dikirim. Untuk menjaga keutuhan pesan (*data integrity*), maka *ciphertext* tersebut dimasukkan ke dalam algoritma *hash function*. Nilai dari *hash function* tersebut kemudian di-*concate* dengan *ciphertext* aslinya. Nilai dari penggabungan tersebut dikirim bersama *ciphertext*-nya kepada Bob. Setelah diterima, Bob akan menguji keutuhan data dengan cara memasukkan *ciphertext* ke algoritma *hashing* yang disepakati untuk digunakan. Kemudian hasilnya di-*concate* seperti yang dilakukan oleh Alice. Jika hasilnya sama, maka dinyatakan *valid* dan pesan tersebut berarti utuh, jika tidak sama maka berarti hal ini telah terjadi perubahan pesan baik penambahan maupun pengurangan isi pesan.

Setelah hasilnya dicek *valid*, langkah selanjutnya Bob membuka *ciphertext* menggunakan algoritma simetrik yang disepakati dengan *KS* yang telah ia terima sebelumnya dari Alice. Maka akan didapatkan hasil berupa *plaintext* yang dikirimkan oleh Alice, sehingga komunikasi pun dapat berjalan lancar.

3. Algoritma yang digunakan

- Untuk pembangkitkan bilangan acak pada saat proses otentikasi dan pembangkitan kunci enkripsi/dekripsi untuk proses komunikasi, menggunakan sebuah algoritma *Random Key Generator* yaitu *Pseudorandom Number Generator* (*PRNG*) yang saat ini masih banyak digunakan karena kekuatannya dalam menghasilkan bilangan acak yang panjang.
- Untuk layanan *confidentiality* (kerahasiaan pesan) maka digunakan algoritma enkripsi simetrik yaitu AES-256. Algoritma ini masih cukup kuat jika digunakan

untuk mengenkripsi pesan yang panjang, enkripsi akan dilakukan dengan cara blok per blok sepanjang 256 bit menggunakan sebuah kunci atau *password*.

- Untuk proses distribusi kunci (*Key Session*) menggunakan algoritma enkripsi asimetrik yaitu RSA-1024, dimana *KS* tersebut dienkripsi menggunakan kunci publik penerima, dan penerima akan membuka *KS* tersebut menggunakan kunci privatnya.
- Untuk layanan *data integrity* (keutuhan pesan) menggunakan algoritma *hash function* yaitu SHA-256. Algoritma ini menghasilkan nilai *hash* dengan panjang mencapai 256 bit. Sebuah pesan yang diberikan algoritma fungsi hash tidak dapat dikembalikan ke nilai semula (*un-inversible*). Fungsi hash berbeda dengan konsep enkripsi yang dapat dibalik (*inversible*) menggunakan algoritma dan kunci yang sama.

4. Rangkuman

Tutorial ini menunjukkan betapa banyak manfaat dari layanan kriptografi yang dapat diimplementasikan untuk mengamankan sebuah proses komunikasi. Pengamanan sebuah komunikasi bukan hanya terletak pada keamanan isi pesan yang tidak dapat dibaca oleh pihak yang tidak berhak, melainkan termasuk juga keutuhan pesan yang disampaikan, keutuhan tidak hanya dilihat dari nilai atau jumlah bit yang sama. Meskipun tidak mengalami penambahan atau pengurangan pesan, tetapi tidak menutup kemungkinan telah terjadi perubahan pesan, oleh karena itu perlu dicek keutuhan pesan dengan algoritma *hash function*. Proses otentikasi sangat diperlukan sebagai langkah awal komunikasi untuk menentukan apakah pihak yang akan berkomunikasi merupakan pihak yang sah atau tidak. Proses komunikasi yang baik selalu menggunakan kunci yang berbeda yang dibangkitkan secara acak, satu kali pada saat akan melakukan komunikasi. Setelah selesai berkomunikasi, maka kunci sebaiknya dihancurkan dan membangkitkan/menggunakan kunci yang baru untuk proses komunikasi lain waktu.

5. Keterangan Gambar

- Alice : Pihak pertama yang akan melakukan komunikasi.
- Bob : Pihak kedua yang akan melakukan

komunikasi.

- RKG : Algoritma *Random Key Generator* (Untuk menghasilkan bilangan acak dan membangkitkan *key session.*)
- R_A : Bilangan acak yang dibangkitkan oleh Alice.
- R_A* : Bilangan acak yang sudah di-*sign* menggunakan kunci privat milik Alice.
- Pb_A : Kunci publik milik Alice.
- P_vA : Kunci privat milik Alice.
- R_B : Bilangan acak yang dibangkitkan oleh Bob.
- R_B* : Bilangan acak yang sudah di-*sign* menggunakan kunci privat milik Bob.
- Pb_B : Kunci publik milik Bob.
- P_vB : Kunci privat milik Bob.
- KS : *Key Session* yang digunakan untuk proses komunikasi.
- *Plaintext* : Pesan terbuka, yang dapat dibaca dengan jelas.
- *Ciphertext* : Pesan yang sudah di-enkripsi, sehingga tidak dapat dibaca dengan jelas.
- *Encrypt/Decrypt Asymmetric* : Algoritma enkripsi/dekripsi asimetrik.
- *Encrypt/Decrypt Symmetric* : Algoritma enkripsi/dekripsi simetrik.
- SHA-256 : Algoritma *hash function* SHA-256 bit.
- *Concate* (*Cipher*||*H*(*Cipher*)) : Penggabungan nilai *ciphertext* dengan nilai *hash*-nya.

Biografi Penulis

Kurnia Wahyu Ningsih, S.T. Menyelesaikan S1 di Universitas Muhammadiyah Prof. Dr. Hamka (UHAMKA) di Jakarta. Penulis mengambil studi S-1 jurusan Teknik Informatika, dengan skripsi berjudul Aplikasi Secure SMS dengan Teknik Kriptografi Menggunakan Algoritma Triple DES. Sempat meniti karir di dunia kerja menjadi *customer service* di PT. Pasific Satelit Nusantara. Saat ini penulis bekerja sebagai Pranata Komputer Pertama di Kementerian Keuangan.