Penerapan Kriptografi pada Sistem Otentikasi Terpusat Kerberos v5

Wildan Fakhri, Achmad Rony Fauzan, dan Imam Ahmadi

Departemen Teknik Informatika
Institut Teknologi Bandung
Jalan Ganesha 10 Bandung 40132

E-mail : if12017@students.if.itb.ac.id, if12021@students.if.itb.ac.id, if12036@students.if.itb.ac.id

Abstrak

Pada masa sekarang, perusahaan memakai jaringan didalam melaksanakan proses bisnisnya. Jaringan tersebut digunakan oleh perusahaan untuk berkomunikasi, baik berkomunikasi dengan *client*, cabang, ataupun dengan konsumennya. Jaringan dapat digunakan untuk berhubungan dengan pihak luar perusahaan, misalnya dengan menggunakan internet dan dapat digunakan untuk berhubungan dengan pihak dalam, misalnya dengan menggunakan intranet. Dengan berbagai macam pihak yang dihubungi dan berbagai macam informasi yang mengalir di dalam jaringan tersebut maka keamanan jaringan adalah hal yang sangat penting. Aspek keamanan di dalam jaringan yang perlu diperhatikan adalah otentikasi, *integrity*, *confidentiality*, dan otorisasi. Kerberos adalah protokol jaringan yang menangani masalah otentikasi. Kerberos memungkinkan *client* dan *server* untuk saling mengotentikasi sebelum melakukan koneksi. Tulisan ini bertujuan untuk memberikan pengetahuan mengenai apa yang dapat dilakukan oleh Kerberos dan bagaimana cara kerjanya secara umum.

Kata kunci : Kerberos, otentikasi, jaringan, keamanan

1. Pendahuluan

Aspek-aspek keamanan pada jaringan yang perlu diperhatikasn adalah otentikasi, *integrity*, *confidentiality*, dan otorisasi. Otentikasi adalah verifikasi identitas dari pihak yang meminta atau memberikan suatu data beserta integritas dari data tersebut. *Principal* adalah pihak yang identitasnya akan diverifikasi, sedangkan *verifier* adalah pihak yang memverifikasi *principal*. Integritas data adalah sebuah keadaan dimana data yang diterima adalah sama dengan data saat dihasilkan. Aspek *confidentiality* adalah perlindungan terhadap informasi yang diberikan agar tidak jatuh kepada pihak yang tidak berwenang. Aspek keamanan terakhir pada jaringan adalah otorisasi, proses untuk memberikan kewenangan terhadap *principal* yang telah diotentikasi untuk melakukan operasi yang merupakan haknya.

Kerberos merupakan protokol jaringan yang menangani masalah otentikasi. Nama Kerberos berasal nama dari mitologi Yunani, yaitu Cerberus. Cerberus adalah makhluk berkepala tiga yang menjaga Underworld dari makhluk hidup yang mencoba untuk memasukinya. Protokol Kerberos mulai didesain pada akhir tahun 1980 di Massachusetts Institute of Technology (MIT) sebagai bagian dari proyek Athena. Kerberos adalah mekanisme otentikasi yang ditujukan untuk *distributed server*. Ia memungkinkan server dan client saling mengotentikasi sebelum melakukan koneksi.

Versi pertama yang dirilis secara umum adalah versi 4 yang akhirnya diperbarui menjadi versi 5 pada tahun 1993. Kerberos mengikuti standar proses IETF dan spesifikasinya di internet terdapat di RFC 1510[3)]. Pada awalnya, Kerberos hanya diaplikasikan pada system operasi UNIX, tetapi kemudian tersedia untuk diaplikasikan di dalam berbagai sistem operasi lainnya yang didistribusikan oleh MIT dalam bentuk versi gratis maupun versi komersil.

2. Keuntungan Menggunakan Kerberos

Otentikasi *paswword-based* merupakan proses otentikasi dimana *client* mengirimkan *password* dan kemudian setelah diverifikasi oleh server maka *client* tersebut mendapat otorisasi untuk melakukan operasi-operasi tertentu. Otentikasi *password-based* ini sangat rawan terhadap serangan pihak ketiga, yaitu ketika seorang penyusup berhasil mendapatkan *password* tersebut saat pengiriman *password* berlangsung. Banyak aplikasi yang menggunakan mekanisme otentikasi yang lemah, bahkan mekanisme otentikasi yang menggunakan asersi, misalnya Barkeley R-command dan protokol IDENT. Mekanisme otentikasi dengan menggunakan asersi ini sangatlah tidak aman karena pertukaran informasi dan keamanannya hanya berdasarkan kepercayaan.

Aspek lain dari keamanan pada jaringan adalah bahwa masalah kemanan bukanlah tugas utama aplikasi pada jaringan. Aplikasi *mail server* yang mempunyai tugas utama untuk mengirimkan *email* kepada pihak yang dituju pada jaringan seharusnya tidak bertugas untuk memverifikasi identitas *user*. Oleh karena itu Kerberos dibutuhkan untuk proses otentikasi. Kerberos mempunyai keuntungan untuk melakukan proses otentikasi dari pusat untuk berbagai aplikasi jaringan. Untuk masing–masing aplikasi yang membutuhkan pelayanan Kerberos maka Kerberos sangat *reliable*, *simple* dan mudah digunakan. Terlebih lagi, Kerberos menghindarkan aplikasi jaringan dari tugas untuk melakukan otentikasi.

3. Protokol Kerberos

3.1 Enkripsi

Data yang dikirim melalui jaringan dapat dirusak, dilihat, ataupun dimodifikasi isinya. Kerberos menyediakan otentikasi kriptografi melalui kombinasi penggunaan kunci rahasia dan *strong* enkripsi. Kerberos menjamin integritas dan kerahasiaan data. Kunci rahasia adalah password yang diketahui oleh *client* atau *server*. Enkripsi dilakukan dengan algoritma kunci simetri menggunakan DES atau *triple* DES. Sekarang juga dikembangkan untuk diimplementasikan menggunakan AES[1)].

Kerberos adalah *tools* yang menyediakan otentikasi untuk pelayanan interaktif, seperti telnet, ftp, pop dan sebagainya dimana *user* diminta memasukkan password untuk melakukan login secara *real time*. Kunci simetri mengizinkan otentikasi dapat dilakukan secara *real time* karena karakteristiknya yang cepat. Algoritma kunci simetri menggunakan kunci yang sama untuk melakukan enkripsi dan dekripsi.

3.2 *Key Distribution Center* (KDC)

Protokol Kerberos digunakan untuk mengotentikasi *principal* dimana telah dijelaskan diatas bahwa *principal* adalah pihak yang identitasnya diverifikasi. Sebuah principal dapatlah merupakan *user* biasa,

sebuah aplikasi server atau sebuah entitas jaringan lainnya yang perlu diotentikasi.

Pihak yang terlibat dalam proses otentikasi adalah:

1. *Client* yang biasanya merupakan *principal*
2. *Server* yang biasanya merupakan *verifier*
3. *Server* Kerberos (KDC)

KDC adalah server Kerberos yang bertugas mendistribusikan *session key* kepada *server* dan *client* agar dapat melakukan koneksi, mengotentikasi *server* dan *client*, serta memudahkan *client* untuk melakukan koneksi kepada lebih dari satu *server*. Tugas untuk mengotentikasi *principal* dan memberikan *session key* kepada *principal* oleh KDC dilakukan melalui *Authentication Service* (AS), sedangkan tugas untuk memudahkan *client* melakukan koneksi dengan satu atau lebih server aplikasi dilakukan melalui *Ticket Granting Service* (TGS).

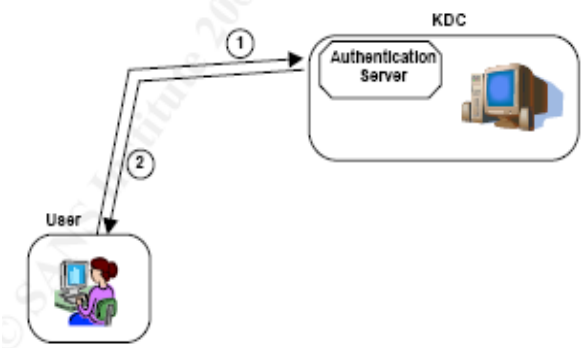
3.2.1 *Authentication Service*

Proses *authentication service* yang dilakukan oleh KDC adalah sebagai berikut:

1. Pada awalnya, *principal (client)* meminta sebuah *ticket* pada KDC dengan mengirimkan namanya, jangka waktu berlakunya permintaan ini, layanan yang diperlukan (tgs), dan beberapa informasi lainnya.

2. KDC kemudian menemukan bahwa *principal* itu ada dalam *database*-nya dan mengirimkan balasan berupa:
   a. Sebuah *client ticket* yang berisi *session key* $S_{A, KDC}$, waktu berlakunya *ticket* ini, dan nama layanan tgs. Semua dienkripsi dengan menggunakan kunci rahasia principal (*password*). Waktu berlakunya tiket ini biasanya selama delapan jam.
   b. Sebuah *granting ticket* yang berisi *session key* $S_{A, KDC}$, waktu berlakunya *ticket* ini, dan nama *client*. Semua dienkripsi dengan kunci rahasia KDC (K_{KDC}). Ini disebut sebagai *Ticket Granting Ticket* (TGT). *Principal* tidak dapat mendekripsi TGT karena dienkripsi dengan menggunakan kunci rahasia KDC. TGT akan digunakan saat meminta *ticket* pada layanan lain.

Dengan melihat proses diatas maka kita dapat menyimpulkan bahwa tidak ada password atau kunci rahasia berupa plainteks yang dikirimkan melalui jaringan. Kunci rahasia *principal* hanya digunakan secara lokal oleh KDC untuk mengenkripsi *ticket* dan digunakan secara local oleh *principal* untuk mendekripsi *ticket. Session key* ($S_{A, KDC}$) digunakan pada saat berkomunikasi untuk menjamin kerahasiaan. *Principal* dapat membuktikan kebenaran identitasnya kepada KDC karena hanya ia yang dapat mendekripsi *client ticket* tersebut.



Gambar 1. *Authentication Service*

Penjelasan pada gambar 1:

1. AS_REQ – {client name, expiration time, tgs service name, …}
2. AS_REP – {$S_{A,\ KDC}$, expiration time, tgs service name, ….}K_A + { $S_{A,\ KDC}$, expiration time, client name, ….}K_{KDC}

3.2.2 Ticket Granting Service

Sekali *client* atau *principal* terotentikasi pada Kerberos maka client tersebut tidak dapat langsung melakukan hubungan dengan layanan (*server*) yang ia butuhkan, tetapi ia harus melakukan permintaan kepada KDC terlebih dahulu. Permintaan (*request*) ini mengandung:

1. Sebuah *authenticator* yang berisi *timestamp* dan *checksum* yang dienkripsi menggunakan session key ($S_{A,\ KDC}$). Pengiriman dengan menggunakan enkripsi ini bertujuan untuk membuktikan identitas *client* karena hanya ia yang mempunyai *session key* ini. *Checksum* berguna untuk membuktikan apakah pesan mengalami perubahan atau tidak pada saat pengiriman berlangsung. *Timestamp* berguna untuk melihat masa berlakunya pesan, yaitu apakah pesan tersebut masih baru untuk menghindari adanya *reply-attack*. Oleh karena itu perlu adanya sinkronisasi waktu.

2. Ticket Granting Ticket (TGT) yang telah diterima pada saat *authentication service*. TGT ini digunakan untuk mengecek nama *client* dan *session key* ($S_{A,\ KDC}$). Apabila *session key*-nya salah maka KDC tidak dapat mendekripsi *authenticator*. TGT juga digunakan untuk mengecek masa berlakunya otentikasi.

3. Nama layanan dari aplikasi yang dibutuhkan oleh *client*.

4. Lama waktu berlakunya TGT

KDC membalas kepada client dengan mengirimkan:

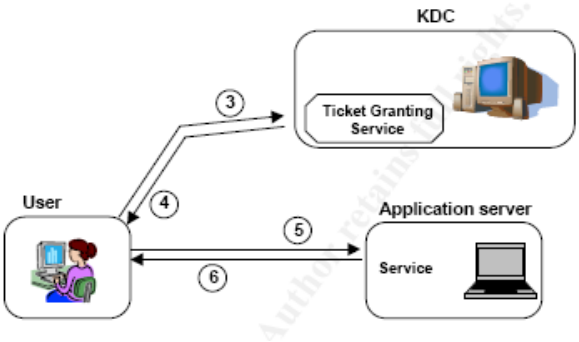
1. *Client Ticket* yang berisi session key ($S_{A,B}$) yang akan digunakan oleh *client* dan *server* untuk berkomunikasi, waktu berlakunya *client ticket* yang baru ini, dan nama layanan aplikasi. Semua ini dienkripsi dengan menggunakan *session key* ($S_{A,\ KDC}$) yang hanya diketahui *client* dan KDC.

2. *Server Ticket* yang berisi session key ($S_{A,B}$), nama *client*, dan waktu berlakunya *ticket* ini. Semua ini dienkripsi dengan menggunakan kunci rahasia server (K_B) yang hanya diketahui oleh server dan KDC.

Lalu menjadi tanggung jawab *client* untuk mengirimkan *server ticket* ini kepada server. Setelah *client* menerima balasan dari KDC tersebut, *client* mendekripsi *client ticket* yang baru dan memperoleh session key ($S_{A,B}$). $S_{A,B}$ diperlukan untuk mengenkripsi *authenticator*. *Authenticator* tersebut berisi *timestamp* dan *checksum*. *Client* lalu mengirim *authenticator* dan *server ticket* tersebut kepada server.

Setelah itu *server* menerima *authenticator* dan *server ticket* dari *client*. *Server* dapat mendekripsi *server ticket* dengan menggunakan kunci rahasianya (K_B) dan memperoleh session key $S_{A,B}$ yang diperlukan untuk mendekripsi authenticator dari *client*. Dengan memperoleh session key $S_{A,B}$ yang benar dari *client* maka berarti juga identitas *client* telah diverifikasi. Lalu sebagai pilihan, *server* akan membalas kepada *client* berupa *timestamp* yang

dienkripsi dengan menggunakan session key $S_{A,B}$. Dengan ini, *client* dapat memverifikasi *server* dan mengetahui bahwa pesan yang diterima adalah baru dengan melihat *timestamp* tersebut.



Gambar 2. *Ticket granting Service*

Penjelasan pada gambar 2 :

3. TGS_REQ – {timestamp, checksum, …} $S_{A,KDC}$ + {$S_{A, KDC}$, expiration name, client name, …} K_{KDC} + application service name + expiration name
4. TGS_REP – {$S_{A,B}$, application service name, expiration time, …}$S_{A, KDC}$ + {$S_{A,B}$, expiration time, client name, …}K_B
5. AP_REQ – {timestamp, checksum, …} $S_{A,B}$ + {$S_{A,B}$, client name, expiration name, …}K_B
6. AP_REP – {timestamp}$S_{A,B}$

3.3 Realms

Kerberos mempunyai kemampuan untuk membagi jaringan didalam grup-grup yang disebut “realms”. Paling sedikit terdapat satu realm dalam KDC. Pembagian ini dilakukan untuk menghindari terlalu banyaknya permintaan terhadap satu KDC. Sebuah nama realm dipetakan khusus terhadap nama domain dari jaringan (DNS) atau terhadap sub-domain, walaupun ini bukan merupakan keharusan. Realm Kerberos yang berasosiasi dengan *principal* mempunyai notasi :

Principal/instance@REALM.COM

Dimana *instance* adalah pilihan masukan yang berasosiasi dengan suatu *principal* pada *database* KDC. Dapat diperhatikan bahwa penulisan realm menggunakan huruf kapital untuk membedakannya dengan nama domain.

Sebuah *principal* yang terdapat pada suatu realm dapat menghubungi sebuah *server* yang berada pada realm yang lain dengan menggunakan kemampuan otentikasi antar realm, yaitu:

1. Pertama-tama *client* meminta TGT dari KDC lokal untuk membuka koneksi dengan *remote* KDC dimana server yang dituju berada.

2. Setelah memperoleh TGT tersebut, *client* mengirimkan TGT tersebut kepada *remote* KDC dan meminta untuk dapat melakukan koneksi kepada server yang dituju

3. Setelah diotentikasi oleh *remote* KDC maka client mengirimkan *authenticator* dan *server ticket* kepada server yang dituju.

Cara ini memungkinkan Kerberos menangani otentikasi ke semua server di jaringan, walaupun dalam realm yang berbeda-beda.

4. Kelemahan Kerberos

Beberapa keterbatasan Kerberos antara lain :
- Walau menyediakan mekanisme enkripsi yang kuat pada suatu jaringan, Kerberos

belum dapat mencegah serangan dengan cara menebak sandi lewat (*password guessing*).
- Otorisasi bukan merupakan bagian dari spesifikasi Kerberos. Oleh karena itu, dibutuhkan mekanisme tambahan yaitu otorisasi yang harus disediakan oleh masing-masing sistem operasi yang menggunakan Kerberos agar tercipta suatu mekanisme otentikasi yang lengkap dan kuat.
- Jika dalam suatu jaringan digunakan sistem otentikasi Kerberos dan sistem yang lain, maka administrator jaringan harus menjamin bahwa *password* Kerberos seorang user tidak digunakan untuk otentikasi pada sistem lain tersebut, untuk menjaga *password* tetap aman.
- Kerberos menggunakan *time-stamp* untuk mengetahui apakah *authenticator* yang dikirimkan masih baru (bukan *replay attack*). Untuk itu dibutuhkan sinkronisasi waktu di seluruh jaringan. Program sinkronisasi waktu yang digunakan umumnya adalah ntpd. Namun, seperti halnya *service* lain di jaringan, *service* ini juga memerlukan otentikasi. Jika tidak, maka dapat terjadi lubang keamanan dalam bentuk *replay attack*.
- Karena Kerberos belum menjadi standar sistem otentikasi, maka aplikasi yang menggunakan Kerberos harus menambahkan suatu modul atau plugin agar dapat menggunakan Kerberos.

5. Kesimpulan

Unsur keamanan dalam suatu jaringan sangatlah penting untuk diperhatikan. Salah satu aspek keamanan jaringan yang penting adalah mekanisme otentikasi. Salah satu protokol yang dapat digunakan untuk mekanisme otentikasi yaitu Kerberos, yang saat ini sudah sampai versi 5. Sistem otentikasi Kerberos merupakan salah satu solusi untuk mengatasi serangan-serangan keamanan di jaringan yang menjadi kelemahan sistem otentikasi konvensional (*password based*).

Mekanisme otentikasi oleh Kerberos dilakukan dengan menggunakan kunci privat yang dibagi (*shared*) antara *client* dengan *server*. Kunci privat tersebut dikeluarkan oleh pihak ketiga yang dipercayai bersama (*trusted third party*). *Username* dan *password* seorang *client* yang tidak dikirimkan melalui jaringan merupakan salah satu keunggulan sistem otentikasi dengan Kerberos ini. Penggunaan *sesssion key* juga mampu meningkatkan keamanan komunikasi dengan protokol Kerberos. Namun, di samping kelebihan tersebut juga terdapat beberapa kekurangan sistem ini, misalnya Kerberos belum dapat mencegah serangan dengan cara menebak sandi lewat (*password guessing*), dan penggunaan *time-stamp* sehingga dibutuhkan sinkronisasi waktu di seluruh jaringan.

[1] Chown, P. *Advanced Encryption Standard (AES)*. June 2002. http://www.ietf.org/rfc/rfc3268.txt. Diakses tanggal 2 Januari 2006.
[2] Fabrice, KAH. *Understanding Kerberos v5 Authentication Protocol*. November 2003.
[3] Kohl, J. Neuman, C. *The Kerberos Network Authentication Service (V5)*. September 1993. http://www.ietf.org/rfc/rfc1510.txt. Diakses tanggal 2 Januari 2006.