Analisis Pengamanan Data Menggunakan SSL/TLS dengan Algoritma ECC pada Transaksi *Ecommerce*

Amalia Rahmah [1)]
1) Jurusan Teknik Informatika ITB, Bandung, email: ami_is_amaliarahmah@yahoo.com

Abstract *– Makalah ini membahas mengenai bagaimana pengamanan data yang dipertukarkan pada transaksi ecommerce. Protokol yang digunakan untuk menjaga keamanan data ini adalah SSL/TLS dan menggunakan algoritma ECC. Untuk menjelaskan lebih rinci bagaimana pengamanan data ini, dalam makalah ini akan dibahas mengenai hal-hal sebagai berikut: bagaimana cara kerja SSL/TLS, bagaimana cara kerja algoritma ECC, bagaimana performansi pengamanan data tersebut menggunakan TLS/SSL dengan algoritma ECC melalui analisis keunggulan algoritma ECC dan algoritma alternatif lain yang dapat diterapkan pada TLS/SSL*

Kata Kunci*:* SSL, TLS, ECC, *ecommerce, kunci publik, kunci privat*

1. PENDAHULUAN

Definisi *E-Commerce* (*Electronic Commerce*) : *E-commerce* merupakan suatu cara berbelanja atau berdagang secara online atau direct selling yang memanfaatkan fasilitas Internet dimana terdapat website yang dapat menyediakan layanan "*get and deliver*". *E-commerce* akan merubah semua kegiatan marketing dan juga sekaligus memangkas biaya-biaya operasional untuk kegiatan trading (perdagangan)[6].

Secure Sockets Layer atau SSL adalah suatu protokol yang mendeskripsikan mengenai bagaimana agar aplikasi yang berbasis *client*/*server* bisa berjalan aman dan juga cepat. Protokol keamanan SSL memberikan fasilitas enkripsi data, autentikasi *server*, integritas pesan dan juga pilihan untuk autentikasi *client*[5].

TLS adalah sebuah protokol yang menjamin privasi dan integritas data antara aplikasi *client/server* yang berkomunikasi melalui Internet. TLS mempunyai kapabilitas untuk melayani servis autentikasi antara *client* dan *server* dengan membuat *encrypted connection* antara keduanya sehingga memberikan garansi integritas dan kerahasiaan pengiriman data.TLS merupakan protokol yang independen terhadap protokol aplikasi, sehingga protokol pada tingkat lebih tinggi dapat secara transparan dijalankan di atas protokol TLS. Protokol SSL/TLS memberikan tiga fungsi security di atas transport layer, yaitu data confidentiality, data integrity dan juga autentikasi.

PKI (*Public Key Infrastructure*) merupakan basis dari infrastruktur keamanan dimana data dikirimkan menggunakan konsep dan teknik kriptografi kunci publik. Komponen dari PKI adalah sebagai berikut:

- *Certification authority*
- *Registration authority*
- *Time stamping authority*
- *Certificate repository*

ElGamal ECC merupakan bentuk baru dari kriptografi PKI berdasarkan kurva eliptik, dimana kita menambahkan dua titik pada kurva untuk mencari titik ketiga pada kurva[7].

2. DASAR TEORI

2.1 Transaksi dalam Ecommerce

Proses yang ada dalam *E-commerce* adalah sebagai berikut :

- Presentasi elektronis (Pembuatan *website*) untuk produk dan layanan.
- Pemesanan secara langsung dan tersedianya tagihan.
- Otomasi *account* pelanggan secara aman (baik nomor rekening maupun nomor kartu kredit).
- Pembayaran yang dilakukan secara langsung (*online*) dan penanganan transaksi.

Arsitektur dasar dari aplikasi ecommerce ini adalah arsitektur *client/server*. Artinya pemrosesan aplikasi ini dijalankan melibatkan kedua sisi yakni sisi mesin *server* pusat dan sisi *client*.

Keamanan transaksi *ecommerce* menuntut adanya hal-hal sebagai berikut: autentikasi, integritas, otorisasi, konfidensialitas, dan reliabilitas pembayaran. Selain itu juga harus dijamin anonimitas pembayar, transaksi tidak dapat dilacak, konfidensialitas data pembayaran, non-repudiasi pesan pembayaran, dan kesegaran pesan pembayaran.

2.2 Pengamanan Data dalam *Ecommerce*

Pengamanan data pada aplikasi *ecommerce* menggunakan metode-metode dan protokol sebagai berikut:

1. Skema keamanan kriptografi asimetris

Komputer yang akan berkomunikasi menggunakan data terenkripsi harus memiliki dua buah kunci untuk mengenkripsi data dan mendekripsinya. Pertama, kunci publik tersedia bagi siapa saja yang ingin melakukan komunikasi terhadapnya. Yang kedua adalah kunci privat yang bersifat rahasia. Misalnya

pada pemrosesan kartu kredit dengan sebuah bank, nasabah memiliki kunci publik bank tersebut dimana ia dapat melakukan dekripsi informasi, namun masih diperlukan kunci privat yang disimpan oleh bank tersebut, untuk dapat melakukan dekripsi data.

2. Sertifikat

Keamanan data yang dilihat dari sisi kerahasianan data ditangani dengan keberadaan kedua kunci ini. Permasalahan keamanan data lainnya adalah apakah data yang diperoleh adalah benar dari pihak yang memiliki otorisasi, bukan dari pihak lain yang tidak berkepentingan atau yang menyalahgunakan.

Untuk itu dibutuhkan pihak ketiga untuk memverifikasi pesan yang datang. Pesan terenkripsi yang dikirim dan diterima akan memiliki semacam '*signature*', dan verifikasi selanjutnya dilakukan terhadap 'signature' tersebut. Untuk itu, organisasi yang akan mempergunakan komunikasi melalui web memerlukan kerjasama dengan organisasi lain yang mengeluarkan sertifikat yang memverifikasi pengirim pesan. Organisasi ini pulalah yang memberikan kunci publik dan kunci privat.

2.3 SSL dalam Pengamanan Data

Protokol HTTP secara alamiah bersifat terbuka terhadap penyusupan. Paket-paket data yang melintas melalui *router* Internet dapat disadap dan dibaca. Namun informasi kartu kredit diinginkan agar tidak mudah terbaca. Untuk itu dibutuhkan penggunaan *Secure Socket Layer* atau SSL. SSL hanya digunakan untuk komunikasi yang bersifat *connection oriented*. Selain itu, SSL hanya mengenkripsikan data yang dikirim lewat HTTP.

SSL merupakan protokol tambahan dimana kunci dan sertifikat dari suatu situs *e-commerce* akan ditransfer ke browser atau ke *server* lain. Melalui SSL, browser akan dapat memverifikasi sertifikat dari situs tersebut sehingga dapat mengetahui identitas pengirim sebenarnya. SSL mengambil data untuk dikirimkan, dipecahkan ke dalam blok-blok yang teratur, dikompresi (jika perlu), menerapkan MAC (*Message Authentication Code*), dienkripsi, dan kemudian hasilnya dikirimkan. Di tempat tujuan, data didekripsi, diverifikasi, didekompresi, dan disusun kembali. Setelah itu, hasilnya dikirimkan ke *client* yang berada di lapisan atasnya.

Bagaimana SSL berjalan dapat digambarkan sebagai berikut :

- Pada saat koneksi mulai berjalan, *client* dan *server* membuat dan mempertukarkan kunci rahasia yang akan dipergunakan untuk mengenkripsi data yang akan dikomunikasikan. Meskipun sesi antara *client* dan *server* dapat dilihat oleh pihak lain, namun data yang terlihat sulit untuk dibaca karena sudah dienkripsi.
- SSL mendukung kriptografi *public key*, sehingga *server* dapat melakukan autentikasi dengan metode yang sudah dikenal umum seperti *RSA* dan *Digital Signature Standard (DSS)*.
- SSL dapat melakukan verifikasi integritas sesi yang sedang berjalan dengan menggunakan algoritma *digest* seperti MD5 dan SHA. Hal ini menghindarkan pembajakan suatu sesi.
- Sertifikat pada SSL dapat mengenkripsi informasi sensitif selama transaksi *online.*
- Setiap sertifikat pada SSL terdiri dari informasi autentikasi unik tentang pemilik sertifikat.

2.4 TSL dalam Pengamanan Data

SSL merupakan protocol yang awalnya didesain oleh Netscape. TLS merupakan versi IETF dari SSL. Versi 3 dari SSL digunakan sebagai dasar untuk standar TLS IETF (Internet Engineering Task Force) versi 1.0. TLS merupakan penyempurnaan dari protokol *SSL (Secure Sockets Layer).*

TLS memiliki dua lapisan, yaitu *TLS Record Protocol* dan *TLS Handshake Protocol*. *TLS Record Protokol* menyediakan keamanan koneksi dengan metode enkripsi seperti *Data Encryption Standart (DES)*, Namun walaupun begitu *TLS Record Protokol* dapat juga di gunakan tanpa enkripsi. Sedangkan *TLS Handshake Protokol* mengizinkan autentikasi antara *client* dan *server* yang akan bernegosiasi dengan menggunakan kunci kriptografi sebelum data berpindah.

Protokol TSL mendukung aplikasi untuk dapat berkomunikasi melalui jaringan dan menghindarkannya dari *eyesdropping*, perusakan, dan pemalsuan pesan. TLS menyediakan autentikasi antara dua sisi (*client* dan *server*) dan komunikasi privat melalui internet menggunakan kriptografi. Biasanya, autentikasi hanya dikenakan pada *server* (misal penjaminan identitas), sedangkan *client* tidak diautentikasi. Hal ini berarti pengguna (baik individu maupun aplikasi seperti *web browser*) dapat menjamin dengan siapa mereka berkomunikasi (penjaminan satu sisi). Level keamanan berikutnya adalah kedua sisi/pihak yang melakukan komunikasi dapat menjamin dengan siapa mereka berkomunikasi. Kondisi ini disebut dengan *mutual authentication*. *Mutual authentication* memerlukan infrastruktur kunci publik (PKI) untuk *client*, jika TLS-PSK atau TLS-SRP (yang dapat menjamin *mutual authentication* tanpa PKI) tidak digunakan.

TLS meliputi tiga fase dasar yaitu:

1. *Peer negotiation* untuk pendukung algoritma
2. Pertukaran kunci dan autentikasi
3. Enkripsi sandi simetris dan autentikasi pesan

Selama fase pertama, *client* dan *server* bernegosiasi mengenai cipher suites, yaitu menentukan sandi yang akan digunakan, kunci yang dipertukarkan, algoritma

autentikasi, dan kode autentikasi pesan (MACs). Pertukaran kunci dan algoritma autentikasi biasanya menggunakan algoritma kunci publik atau menggunakan TLS-PSK (TLS-*pre-shared key*). Kode autentikasi pesan dibangkitkan dari fungsi hash menggunakan konstruksi HMAC.

Client dan *server* sebuah TLS bernegosiasi mengenai sambungan *stateful* menggunakan prosedur *handshaking*. Selama proses *handshaking*, *client* dan *server* 'menyepakati' penggunaan beragam parameter yang digunakan untuk menciptakan keamanan sambungan.

- Proses *handshake* dimulai saat *client* yang tersambung dengan *server* yang menggunakan protokol TLS meminta sambungan aman *(request*), dan meminta daftar sandi dan fungsi hash yang didukung.
- Dari daftar tersebut, *server* memilih sandi dan fungsi hash paling kuat yang juga mendukung dan memberikan notifikasi ke *client* hasil pemilihannya.
- *Server* mengirimkan kembali hasil identifikasinya dalam bentuk sertifikat dijital. Sertifikat ini biasanya mengandung informasi berisi nama *server*, CA (*certificate authority*) yang dipercaya, dan kunci publik enkripsi *server*.
- *Client* dapat menghubungi *server* yang menggunakan sertifikat tersebut dan mengkonfirmasikan bahwa sertifikat telah diautentikasi sebelum diproses.
- Untuk membangkitkan kunsi session yang digunakan untuk sambungan aman, *client* mengenkripsi angka acak dengan kunci publik *server*. Hanya *server* yang bersangkutan dapat mendekripsikannya dengan kunci privatnya. Hal ini merupakan salah satu fakta yang membuat kunci ini disimpan dan disembunyikan dari pihak ketiga. Sehingga hanya *client* dan *server* yang dapat mengakses data ini.
- Dari angka acak, keduanya membangkitkan kunci untuk enkripsi dan dekripsi

Setelah itu proses handshake diakhiri dan sambungan aman dibuka yang kemudian dienkripsi dan didekripsi menggunakan kunci yang telah dibangkitkan sampai sambungan ini ditutup.

Pada proses autentikasi, TLS *client* mengirimkan pesan kepada TLS *server*, dan *server* akan merespon dengan menanyakan informasi yang di butuhkan *server* untuk autentikasi, setelah itu *client* dan *server* akan melakukan perubahan pada *session keys* dan autentikasi dialog-pun berakhir. Apabila proses autentikasi gagal maka komunikasi antara *client* dan *server* tidak dapat berlanjut, tetapi apabila autentikasi berhasil, maka komunikasi yang aman menggunakan SSL/TLS dimulai dengan menanfaatkan *symmetric encryption keys* yang telah di tetapkan ketika proses autentikasi.

Beberapa kelebihan TLS dibandingkan dengan SSL adalah:

- Menggunakan algoritma hash autentikasi pesan yang lebih kuat (HMAC) dibandingkan dengan algoritma MAC yang digunakan sebelumnya pada SSL.
- Pembangkitan kunci yang sudah dimodifikasi dengan menggunakan MD5 (Message Digest 5) dan SHA-1 (Secure Hash Algorithm 1) dengan HMAC.
- Menggunakan baik MD5 dan SHA-1 dalam RSA signature.
- Keterangan error yang lebih lengkap.

2.4 Algoritma ECC

ElGamal ECC adalah suatu metode yang efisien dalam memberikan kriptografi asimetris yang kuat melalui sebuah kurva analog eliptik dari suatu masalah log yang diskrit. Ukuran kunci dari Elliptic Curve adalah lebih kecil dibandingkan ukuran kunci dari RSA, tetapi bisa memberikan kekuatan security yang sama. Sebagai contoh, kunci 160-bit dari Elliptic Curve adalah sama dengan kunci 1024-bit dari RSA. ECC memiliki kekuatan per bit yang maksimal dibandingkan sistem kriptografi yang lainnya.

ECC dapat digunakan untuk beberapa keperluan seperti skema enkripsi (contohnya ElGamal ECC), tanda tangan digital (contohnya ECDSA) dan protokol pertukaran kunci (contohnya Diffie Hellman ECC). ECC dapat menggunakan ukuran kunci yang lebih kecil dibandingkan dengan kriptografi lainnya dan memiliki tingkat keamanan yang sama. Kemampuan ini membuat ECC mempunyai keamanan yang terkuat dengan panjang kunci terpendek. Sebagai perbandingan, 160 bit ECC mempunyai tingkat keamanan yang sama dengan 1024 bit RSA atau DSA dan 224 bit ECC memiliki tingkat keamanan yang sama dengan 2048 bit RSA atau DSA.

Kekuatan ElGamal ECC tergantung pada panjang kunci yang digunakan dalam proses enkripsi dan dekripsi serta pemilihan parameter-parameter domainnya. Parameter-parameter tersebut dipilih sehingga diperoleh order *basic point* yang terbesar. Algoritma ElGamal ECC melibatkan komputasi yang rumit meliputi operasi perkalian skalar kurva *elliptic* dan representasi *plaintext* menjadi titik. Tetapi memiliki tingkat keamanan yang tinggi dengan panjang kunci terpendek.

3. PEMBAHASAN

3.1 Protokol SSL/TSL untuk *ecommerce*

Protokol SSL/TLS memberikan tiga fungsi security di atas transport layer, yaitu *data confidentiality*, *data integrity* dan juga autentikasi. Enkripsi SSL memberikan tiga jasa tadi yang akan mampu untuk memproteksi manajemen *traffic* pada web seperti konfigurasi, aktivasi dan juga *billing*.

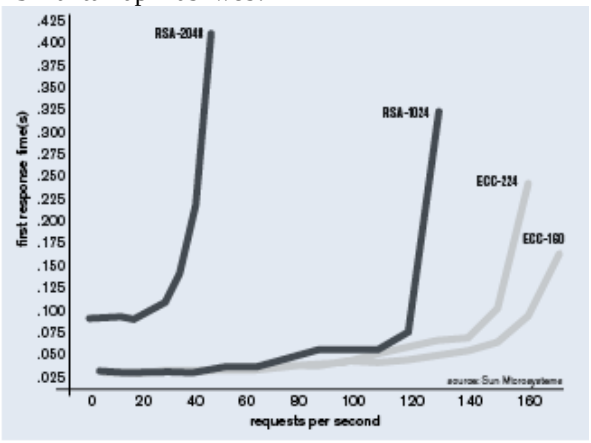
Protokol SSL dan TSL bukan merupakan protokol pembayaran, tetapi saat ini digunakan terutama untuk pembayaran kartu kredit di internet, salah satunya pada *ecommerce*. Komunikasi antara pembeli dan penjual diekripsikan, dan penjual diautentikasikan. Proteksi juga dilakukan terhadap data detail kartu kredit yang melewati jalur komunikasi.

3.2 Analisa Keunggulan Algoritma ECC dibandingkan dengan RSA untuk diterapkan pada TLS/SSL

Perbandingan antara ECC dan RSA dapat diperlihatkan melalui percobaan menggunakan parameter-parameter antara lain handshake crypto latency, server crypto throughput, waktu pembangkitan kunci, serta perhitungan tingkat keamanan masing-masing kunci. Pada percobaan ini dilakukan empat macam skenario yaitu pada RSA-1024, 2048 dan 4096 serta pada ECC-160, 224 dan 384.

Hasil yang didapatkan pada percobaan memperlihatkan bahwa ECC memiliki performansi yang lebih baik dibandingkan RSA yaitu : performansi waktu pembangkitan kunci yang sekitar 73-1300 kali lebih cepat, performansi handshake crypto latency yang lebih baik sekitar 1.7 hingga 4 kali (kecuali untuk ECC-160 lebih lambat 1.7 kali dibanding RSA-1024), serta performansi server crypto throughput sekitar 1.5 – 16 kali lebih besar (pada otetikasi server saja) dan 2-6 kali lebih besar pada otentikasi penuh (kecuali pada ECC-160 lebih kecil 1.5 kali dibadingkan RSA-1024). Disamping itu tingkat keamanan pada ECC memiliki keunggulan ketangguhan hingga 10^3 – 10^{21} kali sehingga memberikan bukti bahwa selain memiliki performansi yang lebih baik juga memberikan keamanan yang jauh lebih tinggi[9].

Grafik berikut menunjukkan dampak pada *server response time* akibat penggunaan algoritma ECC dan RSA untuk aplikasi web:



Gambar 1 Grafik *Server Response Time*[8]

Berdasarkan grafik ini,dapat disimpulkan bahwa untuk menjalankan *web traffic* yang sama, seseorang yang menggunakan algoritma RSA daripada ECC harus mengeluarkan biaya perawatan *server* sebanyak 3,5 kali lebih besar untuk menjalankan jumlah *traffic* yang sama.

Algoritma RSA dan DSA cenderung memiliki kelemahan yaitu jumlah bit (*overhead*) yang terlalu panjang sehingga memerlukan waktu perhitungan yang tinggi, memperlambat pembentukan *digital signatures*. Algoritma ECC lebih baik dibanding algoritma RSA dan DSA karena memiliki kelebihan dalam *digital signatures* yang lebih pendek dan jumlah bit *key* yang lebih pendek

Ukuran kunci dari Elliptic Curve adalah lebih kecil dibandingkan ukuran kunci dari RSA, tetapi bisa memberikan kekuatan security yang sama. Sebagai contoh, kunci 160-bit dari Elliptic Curve adalah sama dengan kunci 1024-bit dari RSA. ECC memiliki kekuatan per bit yang maksimal dibandingkan sistem kriptografi yang lainnya.

3.3 Penggunaan Algoritma ECC pada SSL/TLS

Berdasarkan pemaparan kelebihan algoritma ECC pada subbab sebelumnya, maka dapat dirumuskan alasan penggunaan ECC untuk transaksi pada *ecommerce* adalah sebagai berikut:

a. Menjamin data-data yang terlibat pada transaksi *ecommerce* tetap aman
b. ECC menggunakan *bandwidth* yang lebih kecil daripada algoritma alternatif lainnya untuk SSL/TLS.
c. Transaksi perlu diproses lebih efisien. Tujuannya adalah agar performansi transaksi *ecommerce* lebih baik dan pada akhirnya.
d. Daya komputasi dan memori yang digunakan untuk pemrosesan menggunakan algoritma ECC juga lebih kecil daripada algoritma lainnya.

ECC *Cipher Suites* dapat diimplementasikan pada TLS. ECC Cipher suites ini terdiri dari Elliptic Curve Diffie-Hellman (ECDH) pada kesepakatan penggunaan kunci saat *handshaking* TLS dan Elliptic Curve Digital Signature Algorithm (ECDSA) sebagai mekanisme autentikasi.

4. KESIMPULAN

1. Transaksi *ecommerce* membutuhkan pengamanan data dengan tingkat sekuritas yang tinggi karena melibatkan data-data pribadi dan data-data pembayaran
2. Pengamanan data pada *ecommerce* meliputi enkripsi dan dekripsi menggunakan kriptografi asimetris (kunci publik dan kunci privat) serta sertifikat untuk menjamin dengan siapa komunikasi terjadi (pihak yang benar).
3. SSL merupakan salah satu solusi pengamanan data pada transaksi *ecommerce.*
4. TSL merupakan bentuk pengembangan dari SSL.

5. Algoritma ECC merupakan algoritma yang memiliki tingkat keamanan terkuat dengan kunci terpendek dibandingkan dengan algoritma lainnya dengan waktu komputasi yang digunakan lebih singkat dibandingkan algoritma lainnya sehingga dapat menghasilkan performansi terbaik untuk pengamanan data pada transaksi *ecommerce*.

DAFTAR REFERENSI

[1] http://searchsecurity.techtarget.com/sDefinition/0,,sid14_gci557332,00.html.
[2] http://www.ilmukomputer.com
[3] http://sdn.vlsm.org
[4] [http://research.sun.com/
[5] http://www.sony-ak.com/
[6] http://www.informatika.lipi.go.id/
[7] http://www.digicert.com/
[8] http://www.certicom.com
[9] http://www.stttelkom.ac.id/