Secure SMS : Protokol Komunikasi Aman Diatas Layanan SMS

Fahziar Riesad Wutono - 13512012[1]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]fahziar@gmail.com

***Abstract*—SMS merupakan salah satu media komunikasi yang populer. Isi pesan SMS tidak dienkripsi sehingga memungkinkan SMS untuk disadap dan mengurangi privasi pengirim atau penerima SMS. Protokol *Secure SMS* merupakan protokol yang bertujuan untuk melindungi isi SMS dengan cara mengenkripsinya. Protokol ini menangani pertukaran kunci serta enkripsi dan dekripsi isi SMS. (*Abstract*)**

Keywords—SMS; Pengamanan SMS; Enkripsi SMS

I. PENDAHULUAN

A. Latar Belakang

Saat ini SMS sering digunakan sebagai media komunikasi. Sayangnya, media SMS ini tidak dienkripsi. Hal ini membuat operator atau pihak ketiga lainnya dapat membaca isi SMS yang dikirim.

SMS dikirim melalui gelombang elektromagnetik di udara. Karena dikirim melalui udara, alat-alat di sekitar pengirim atau penerima SMS dapat menangkap SMS yang dikirim.

Saat ini terdapat perangkat lunak yang dapat digunakan untuk menganalisis sinyal GSM. Contohnya adalah GR-GSM. Dengan menggunakan perangkat lunak ini, seseorang dapat menganalisa dan men-*decode* sinyal GSM dan mendapatkan isi SMS.

Agar isi SMS tidak dapat dibaca oleh pihak yang tidak berhak, isi SMS perlu dienkripsi. Dengan kemampuan prosesor telepon genggam saat ini, prosesor telepon genggam mampu menjalankan algoritma kriptografi modern yang umum digunakan di komputer biasa. Meskipun begitu, ada beberapa keterbatasan yang harus diperhatikan :

- Telepon genggam menggunakan baterai sebagai sumber tenaganya. Algoritma kriptografi yang berat akan menghabiskan baterai telepon genggam lebih cepat.
- Kapasitas pengiriman data per SMS adalah 140 bytes. Biaya pengiriman data lewat SMS lebih mahal dibandingkan melalui internet. Untuk menghemat biaya, ukuran cipherteks yang dihasilkan oleh algoritma kriptografi harus sekecil mungkin.

Algoritma kriptografi simetri lebih cocok digunakan untuk mengenkripsi isi SMS dibandingkan dengan algoritma kriptografi asimetri karena algoritma kriptografi simetri lebih ringan dibandingkan dengan algoritma kriptografi asimetri serta ukuran cipherteks yang dihasilkan umumnya sama dengan ukuran plainteks. Namun, algoritma kriptografi simetri mengharuskan pengirim dan penerima pesan mempunyai kunci yang sama. Hal ini menyulitkan jika penerima dan pengirim tidak dapat bertemu langsung dan tidak ada kanal komunikasi yang cepat dan aman untuk bertukar kunci.

Untuk menyelesaikan masalah-masalah tersebut, dibuatlah protokol komunikasi *Secure SMS* yang dapat mengamankan isi SMS. Hal-hal yang ditangani oleh protokol ini antara lain:

- Perlindungan isi pesan dengan menggunakan algoritma kriptografi simetri.
- Pertukaran kunci yang akan digunakan oleh algoritma kriptogafi simetri saat mengenkripsi atau mendekripsi pesan. Pertukaran kunci ini dilakukan lewat SMS sehingga memberikan kemudahan bagi pengirim dan penerima SMS.

B. Pekerjaan Terkait

Protokol *Secure SMS* mempunyai kemiripan dengan protokol TLS yang mengamankan komunikasi melalui internet. Keduanya sama-sama menangani perlindungan komunikasi di kanal yang tidak aman. Keduanya juga menangani pertukaran kunci yang digunakan untuk mengenkripsi data yang dikirim.

Perbedaan dengan protokol TLS adalah protokol *Secure SMS* dioptimasi untuk komunikasi lewat SMS. Untuk bertukar kunci, tahapan yang harus dilakukan lebih sedikit. Ukuran *header* protokol *Secure SMS* juga lebih sederhana dibandingkan dengan *header* protokol TLS.

Untuk pertukaran kunci, protokol *Secure SMS* menggunakan algoritma ECDHE_ECDSA. Algoritma ini merupakan salah satu algoritma pertukaran kunci yang digunakan di protokol TLS.

C. Organisasi Makalah

Makalah ini dibagi menjadi lima bab. Bab pertama merupakan pendahuluan yang membahas latar belakang protokol ini dibuat dan pekerjaan terkait. Bab kedua merupakan teori-teori dasar yang digunakan untuk membuat protokol ini. Bab ketiga menerangkan rancangan protokol *Secure SMS*, spesifikasi protokol *Secure SMS*, struktur dari setiap pesan yang dikirim dan fase-fase dari protokol *Secure SMS*. Bab keempat berisi analisis dari protokol *Secure SMS*. Bab kelima berisi

kesimpulan dan saran untuk implementasi dan pengembangan protokol *Secure SMS*.

II. Dasar Teori

A. SMS

SMS merupakan layanan pengiriman pesan singkat yang populer digunakan. Spesifikasi layanan SMS diatur oleh ETSI. Standar yang digunakan oleh SMS adalah standar GSM 03.40 dan standar GSM 03.38.

SMS dapat dikirimkan dalam mode teks dan mode PDU. Mode PDU memungkinkan pengiriman pesan dalam bentuk biner. PDU memungkinkan pengirim mempunyai kontrol penuh terhadap data yang dikirim. Dengan menggunakan mode PDU, seseorang dapat mengirimkan data biner atau menggunakan *encoding* khusus, tidak seperti mode teks yang hanya dapat mengirimkan teks dengan *encoding* yang sudah ditentukan. Satu pesan SMS dalam mode PDU dapat mengirim maksimal 140 bytes data. [1]

B. Elliptic Curve Cryptography

Elliptic curve cryptography merupakan salah satu bentuk kriptografi asimetri. Kriptografi asimetri memungkinkan penggunaan kunci yang berbeda untuk enkripsi dan dekripsi data.

Operasi matematika pada *elliptic curve cryptography* didefinisikan pada sebuah kurva elliptik dengan persamaan:

$$y^2 = x^3 + ax + b \tag{1}$$

Nilai kurva elliptik merupakan bilangan real. Operasi bilangan real mempunyai kekurangan yaitu tidak secepat operasi bilangan bulat dan kurang presisi karena adanya pembulatan. Salah satu solusinya adalah menggunakan kurva elliptik yang berada di medan bilangan prima. Persamaan kurva elliptik di medan bilangan prima adalah sebagai berikut:

$$y^2 \bmod p = x^3 + ax + b \bmod p \tag{2}$$

Nilai a, b dan p pada persamaan di atas merupakan parameter domain dari ECC di medan bilangan prima. Sejumlah lembaga, seperti SEC, sudah mengeluarkan standar untuk nilai-nilai domain.

Beberapa algoritma yang menggunakan ECC antara lain ECDSA dan ECDH. ECDSA merupakan singkatan dari *Elliptic Curve Digital Signature Algorithm*. Sesuai namanya, ECDSA digunakan untuk membuat dan memverifikasi tanda tangan digital. ECDH merupakan singkatan dari *Elliptic Curve Diffie Hellman*. ECDH digunakan untuk pertukaran kunci di atas kanal yang tidak aman.[2]

C. ECDHE_ECDSA

ECDHE_ECDSA merupakan algoritma pertukaran kunci. Algoritma ini menggunakan perpaduan algoritma ECDH dan ECDSA. Algoritma ini dijadikan salah satu algoritma pertukaran kunci pada protokol TLS. Spesifikasi algoritma ini didefinisikan di RFC4492 *section* 2.2.[3]

Untuk pertukaran kunci, algoritma ini menggunakan algoritma ECDH dengan *ephemeral public key* sebagai kuncinya. *Ephemeral public key* ini dibuat setiap kali pertukaran kunci dilakukan dan pada setiap pertukaran kunci digunakan nilai yang berbeda.

Ephemeral public key tidak dapat digunakan untuk memastikan pengirim kunci karena nilainya selalu berubah. Untuk memastikan *ephemeral public key* datang dari pengirim yang sah, *ephemeral public key* perlu ditandatangani. Tanda tangan digital yang digunakan dibuat dengan menggunakan ECDSA dengan menggunakan *private key* milik pengirim. *Private key* yang digunakan disini merupakan *private key* yang berbeda dengan *ephemeral private key* yang digunakan oleh ECDH. *Private key* di sini bersifat lebih statis sehingga tanda tangan yang dihasilkan dapat diverifikasi keabsahannya.

D. Cipher Feedback

Cipher feedback (CFB) adalah salah satu mode operasi cipher blok. Cipherteks blok sebelumnya menjadi masukan algoritma kriptografi enkripsi. Hasilnya di-XOR-kan dengan plainteks untuk mendapatkan cipherteks. Untuk blok pertama digunakan *initialization vector* (IV) sebagai masukan algoritma kriptografi enkripsi.

Untuk melakukan dekripsi, cipherteks blok sebelumnya dimasukkan ke dalam algoritma kriptografi enkripsi. Hasilnya di-XOR-kan dengan cipherteks untuk mendapatkan plainteks. Untuk blok pertama digunakan IV yang sama dengan IV yang digunakan saat enkripsi. [4]

III. Rancangan Protokol

A. Spesifikasi Protokol

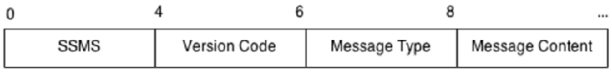
Setiap pesan pada protokol *Secure SMS* mempunyai nomor versi. Dengan begitu, protokol *Secure SMS* dapat mempunyai versi yang berbeda-beda. Setiap versi protokol tidak harus kompatibel dengan versi protokol lainnya. Versi protokol yang dibahas di makalah ini adalah versi 0 yang merupakan versi awal dari protokol *Secure SMS*.

Protokol versi 0 menggunakan spesifikasi algoritma dan kunci sebagai berikut:

- Algoritma kriptografi simetri yang digunakan adalah algoritma Rijndael dengan panjang kunci 128 bit.
- Algoritma kriptografi simetri dioperasikan dengan mode CFB. IV yang digunakan oleh CFB dibangkitkan secara acak setiap kali melakukan enkripsi pesan.
- Algoritma *digital signature* yang digunakan adalah algoritma ECDSA dengan parameter kurva sesuai dengan standar secp128r1 dan fungsi *hash* yang digunakan adalah SHA256.
- Algoritma pertukaran kunci yang digunakan adalah algoritma ECDHE dengan parameter kurva sesuai standar secp192r1.

B. Format Pesan

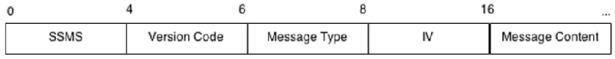
Protokol *Secure SMS* menggunakan berbagai macam jenis pesan. Secara garis besar, format pesan yang dikirim adalah seperti gambar 1. Setiap pesan diawali oleh 4 buah karakter ASCII yaitu 'S', 'S', 'M' dan 'S'. Empat karakter tersebut diikuti oleh versi protokol yang digunakan. Versi protokol ini bertipe *unsigned integer* 16 bit. Setelah versi protokol terdapat informasi tipe pesan. Tipe pesan ini digunakan untuk mengidentifikasi jenis pesan. Setelah informasi tipe pesan terdapat *message content*. Format *message content* berbeda-beda tergantung tipe pesan. Urutan bit yang digunakan pada setiap pesan sesuai dengan aturan *big endian*.



Gambar 1. Struktur umum pesan *Secure SMS*

Berikut ini adalah berbagai tipe pesan yang digunakan oleh protokol *Secure SMS*.

1) Pesan Data

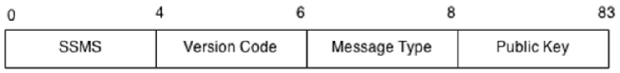


Gambar 2. Struktur pesan data

Pesan ini merupakan pesan SMS yang isinya sudah dienkripsi. Pesan data merupakan pesan yang digunakan oleh kedua pihak untuk berkomunikasi.

Gambar 2 merupakan format pesan data. *Message type* pada *header* bernilai 0. IV merupakan *initialization vector* yang digunakan untuk mengenkripsi pesan.

2) Pesan Hello



Gambar 3. Struktur pesan *hello* dan *hello response*

Pesan *hello* merupakan pesan yang digunakan untuk memulai protokol *Secure SMS*. Pesan ini berisi informasi kunci publik. Kunci publik tersebut di-*encode* dengan format X.509.

Gambar 3 merupakan format pesan *hello*. *Message type* pada *header* bernilai 1.

3) Pesan Hello Response

Pesan *hello response* merupakan pesan yang digunakan untuk membalas pesan *hello*. Format pesan pada sama dengan format pesan *hello*. Perbedaannya terletak pada *message type* yang bernilai 2.

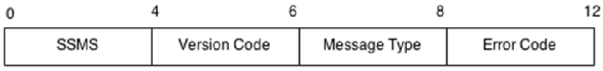
4) Pesan Session Negotiation

Pesan *session negotiation* merupakan pesan yang digunakan untuk menyepakati kunci sesi. Gambar 4 merupakan struktur pesan *session negotiation*. *Message type* berisi nilai 3. *Session lifetime* merupakan waktu kadaluwarsa sebuah sesi, dinyatakan dalam berapa jam setelah 1 Januari tahun 2000 pukul 00.00. *Session ID* merupakan ID dari sesi. *Ephemeral public key* adalah kunci publik yang digunakan oleh ECDHE untuk mendapatkan kunci sesi. *Ephemeral public key* di-*encode* dalam format X.509. *Signature* merupakan tanda tangan digital. Tanda tangan digital ini menandatangani parameter sesi dan *ephemeral public key*.

5) Pesan Session Agreement

Pesan *session agreement* merupakan pesan yang digunakan untuk menyatakan sepakat dengan parameter sesi yang digunakan. Format pesan pada sama dengan format pesan *session negotiation*. Perbedaannya terletak pada *message type* yang bernilai 4.

6) Pesan Error



Gambar 5. Struktur pesan *error*

Pesan *error* dikirimkan jika terjadi kesalahan, ketidaksepakatan saat negosiasi kunci sesi atau kegagalan verifikasi kunci. Pesan *error* berisi kode *error* yang menjelaskan kesalahan apa yang terjadi.

Gambar 5 merupakan format pesan *error*. Kode *error* merupakan *unsigned integer* 32 bit. *Message type* diisi dengan nilai 5.

C. Fase Komunikasi

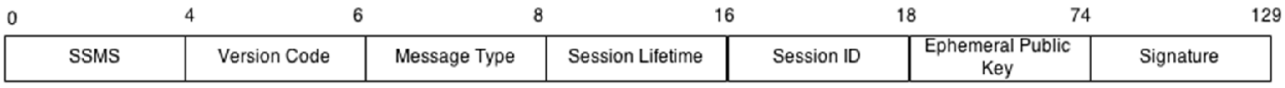
Protokol ini terdiri dari lima fase, yaitu fase *handshaking*, fase *starting session,* fase *established*, fase *session end,* dan fase *end*.

1) Fase Handshaking

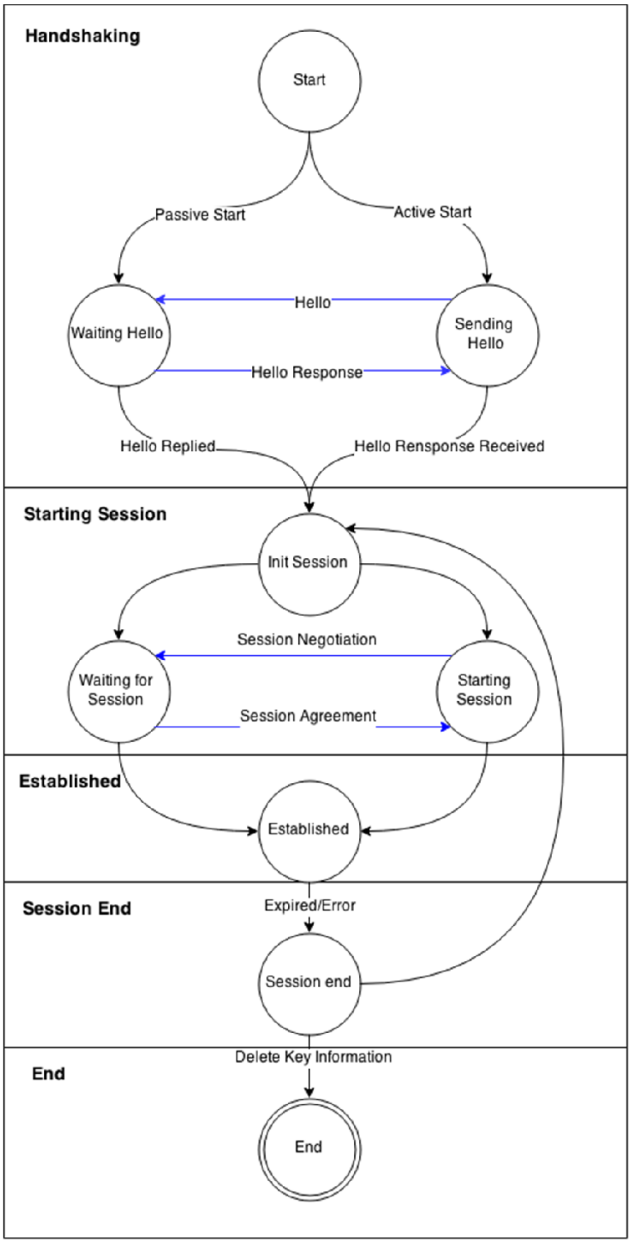
Fase ini merupakan fase di mana kedua pihak saling berkenalan dengan cara saling bertukar kunci publik. Fase ini diawali dengan salah satu pihak mengirimkan pesan *hello* yang berisi kunci publik miliknya. Pelaku pertama yang mengirimkan kunci publik disebut pelaku *active start* sedangkan pihak lainnya disebut pelaku *passive start*.

Pelaku *passive start* menerima kunci publik dan memverifikasi kunci publik tersebut. Jika kunci publik valid, kunci publik tersebut disimpan dan membalas pesan *hello* dengan pesan *hello response*. Pesan ini berisi kunci publik milik pelaku *passive start*.

Setelah pelaku *active start* menerima pesan *hello response* pelaku *active start* memverifikasi kunci publik miliki pelaku



Gambar 4. Struktur pesan *session negotiation* dan *session agreement*



Gambar 6. Diagram fase dan status dari protokol *Secure SMS*. Panah hitam menyatakan perpindahan status. Panah biru menyatakan pengiriman pesan.

passive start. Jika kunci tersebut valid, kunci tersebut disimpan oleh pelaku *active start* dan kedua belah pihak masuk ke fase *starting session*.

Jika salah satu pihak menilai kunci publik yang diterima tidak valid, pihak tersebut dapat mengirimkan pesan *error*. Setelah itu, fase ini harus diulang dari awal.

Kunci publik yang saling dipertukarkan di sini digunakan pada fase *starting session*. Kunci publik ini digunakan untuk memverifikasi tanda tangan digital.

2) Fase Starting Session

Fase ini merupakan fase di mana kunci algoritma kriptografi simetri disepakati. Setiap sesi mempunyai kunci yang berbeda. Kunci ini disebut kunci sesi.

Fase ini diawali dengan salah satu pihak mengirimkan pesan *session negotiation* ke pihak lainnya. Pesan ini berisi parameter sesi dan *ephemeral public key* yang baru dibuatnya. Parameter sesi dan *ephemeral public key* ditandatangani dengan kunci privat milik pengirim.

Kemudian pihak penerima pesan *session negotiation* memverifikasi isi pesan dengan menggunakan *digital signature* yang diterima serta kunci publik yang ia simpan pada fase pertama. Jika valid dan parameter sesi disetujui, pihak ini mengirimkan pesan *session agreement* kepada pengirim pesan *session negotiation*. Pesan tersebut berisi parameter sesi dan *ephemeral public key* yang baru dibuat. Parameter sesi di sini mempunyai nilai yang sama dengan parameter sesi di pesan *session negotiation*.

Jika parameter sesi tidak disetujui, penerima pesan *session negotiation* mengirimkan pesan *error*. Setelah itu, sesi ini harus diulang kembali.

Setelah *session agreement* dikirim, kedua belah pihak menghitung nilai kunci sesi. Kemudian fase berpindah ke fase selanjut.

3) Fase Established

Pada fase ini kedua belah pihak dapat saling berkomunikasi dengan isi pesan SMS dienkripsi. Enkripsi dilakukan dengan algritma kriptografi simetri dan kunci sesi yang dibuat di fase *session start*.

Fase berpindah dari fase *established* ke fase *session end* jika terjadi *error* atau sesi sudah kadaluwarsa.

4) Fase Session End

Pada fase ini kunci sesi dihapus. Kedua pihak harus mengulang dari fase *starting session* untuk kembali berkomunikasi dengan protokol *secure SMS*.

5) Fase End

Pada fase ini kunci publik dihapus. Kedua pihak harus mengulang dari fase *handshaking* untuk kembali berkomunikasi dengan protokol *secure SMS*.

IV. Analisis Keamanan

A. Keamanan Terhadap Padding Oracle Attack

Padding oracle attack merupakan serangan yang mengeksploitasi kelemahan sistem yang mendekripsi pesan. Kelemahan ini berupa pemberitahuan pesan kesalahan saat *padding* hasil dekripsi tidak valid. Pesan kesalahan ini dapat dimanfaatkan oleh penyerang untuk menebak nilai *padding* yang digunakan. Dengan memanfaatkan nilai *padding*, penyerang dapat menemukan plainteks tanpa harus memiliki kunci.[5]

Padding oracle attack menyerang mode kriptografi yang mengharuskan *padding* ditambahkan. Protokol *Secure SMS* menggunakan mode enkripsi CFB yang tidak membutuhkan *padding* sehingga protokol *Secure SMS* aman dari serangan *padding oracle*.

Mode CFB dipilih dengan tujuan agar terlindung dari serangan *padding oracle* tanpa harus menambah ukuran pesan. Mode enkripsi dengan *padding* dapat terlindung dari serangan *padding oracle* dengan menambahkan MAC pada setiap pesan yang dikirim. Karena keterbatasan ukuran SMS, penambahan MAC membuat pengguna hanya dapat mengirimkan sedikit data dalam sebuah SMS.

B. *Perfect Forward Secrecy*

Perfect forward secrecy adalah fitur dari sebuah protokol komunikasi yang menjamin kunci sesi tetap aman walaupun kunci privat salah satu pihak telah dicuri.[6] Penyerang dapat menyimpan seluruh komunikasi dengan harapan di masa depan ia bisa mendapat kunci privat salah satu pihak dan dari kunci privat tersebut didapatkan kunci sesi yang bisa mendekripsi semua komunikasi yang telah ia simpan. *Perfect forward secrecy* menjamin serangan seperti ini tidak dapat dilakukan.

Perfect forward secrecy merupakan salah satu fitur dari protokol *Secure SMS*. Protokol *Secure SMS* menggunakan pasangan *ephemeral public key* dan *ephemeral private key* yang berbeda di tiap sesinya untuk menentukan kunci sesi. Karena tidak menggunakan kunci privat milik salah satu pihak, walaupun kunci privat salah satu pihak dicuri semua komunikasi yang pernah dilakukan di masa lalu tetap aman.

C. *Keamanan Panjang Kunci*

Salah satu cara pengamanan dari serangan *known plaintext attack* dan *known ciphertext attack* adalah menggunakan kunci yang cukup panjang. Beberapa lembaga sudah mengeluarkan standar panjang kunci yang aman untuk digunakan saat ini dan di masa yang akan datang.

Protokol *Secure SMS* menggunakan panjang kunci 128 bit untuk cipher Rijndael. Kunci ini masih cukup panjang untuk digunakan hingga tahun 2030 menurut perhitungan dengan metode ANSSI, NIST, ENCRYPT II, Lenstra Updated dan Lenstra / Verheul.

Panjang kunci yang digunakan *Secure SMS* untuk pembuatan tanda tangan digital adalah 192 bit. Kunci ini masih cukup panjang untuk digunakan hingga tahun 2030 menurut metode penghiutngan NIST, ENCRYPT II, Lenstra Updated dan Lenstra / Verheul.

Panjang kunci yang digunakan protokol *Secure SMS* untuk *ephemeral public key* dan *ephemeral private key* yang digunakan sebagai kunci ECDH adalah 128 bit. Menurut perhitungan metode ANSSI, NIST, ENCRYPT II, Lenstra Updated dan Lenstra Verheul kunci ini tidak cukup panjang. Kunci yang aman untuk digunakan saat ini adalah kunci dengan panjang minimum 160 bit.[7]

D. *Pengujian*

1) *Pengujian Dengan Kunci dan Plaintekst Sama*

Kunci yang digunakan untuk cipher Rijndael selalu sama di setiap sesinya. Pengirim dapat mengirimkan pesan yang sama di dalam satu sesi.

Plainteks: Kunci jawaban UAS ada di laci saya
Kunci (dalam Base64): sq6jlYpKPA38v1jEf0ssQg==

Cipherteks pada pesan pertama:
.{ر U-�┼ Km�• ⁄/⁄{| o⊥ ⁄]7T⁄uu⁄⁄H]S@⁄
Cipherteks pada pesan kedua:
⁄⁄⁄ɪw⁄┼ ü%⁄→ ⁄⁄⁄┬ DX{− ⁄⁄⁄⁄L⁄⁄◀ \g⁄<

Meskipun kunci dan plainteks sama, cipherteks yang dihasilkan berbeda. Penyebab perbedaan ini adalah IV yang dibuat secara acak setiap kali enkripsi pesan dilakukan. Hal ini menyulitkan penyerang untuk mendapatkan plainteks.

2) *Pengujian Perubahan Cipherteks*

Penyerang dapat mengubah cipherteks sesuai dengan keinginannya. Penerima dapat dikelabui dengan cara ini karena seakan-akan pesan yang diterima berasal dari pengirim yang sah, bukan dari penyerang. Algoritma yang baik dapat memberikan peringatan kepada penerima bahwa telah ada isi pesan yang diubah.

Plainteks: Kunci jawaban UAS ada di laci saya
Kunci (dalam Base64): GiF5PBzvB1GXtoMoCxGAlw==

Cipherteks asli (dalam Base64):
mOqF4MTFpjUxM88JVu8ilWvqVMl+5aGMNhiOB9LLWK
bc3Q==
Cipherteks yang telah dimodifikasi (dalam Base64):
mOqFWsTFpjUxM88JVu8ilWvqVMl+5aGMNhiOB9LLWK
bc3Q==

Hasil dekripsi:
Kun⁄i jawaban UA⁄− T⁄Q⁄⁄&− S⁄⁄nWya

Dapat dilihat perubahan kecil (perhatikan karakter ke lima dan ke enam cipherteks) mengakibatkan perubahan besar pada hasil dekripsi.

3) *Pengujian dengan Plainteks yang Berpola*

Pengirim bisa saja mengirim plainteks yang isinya mengandung pengulangan dan menghasilkan sebuah pola. Algoritma yang baik menyembunyikan pengulangan ini sehingga penyerang tidak dapat mengidentifikasi pola hanya dengan melihat cipherteks.

Plainteks:
“halo halo halo halo”

Key (dalam Base64):
8jYxYE1uLziXlHaS6gmFig==

Cipherteks (dalam heksadesimal):
0x0C 0xFB 0xA6 0xB6 0xCA 0x12 0xC6 0xF6 0x56 0x7C
0xBC 0xD8 0x05 0x00 0x86 0xAC 0x91 0x6D 0xE1 0x7B
0x52 0xCB 0x23 0x99

Dapat dilihat bahwa tidak ada pola pada cipherteks. Penggunaan mode CFB membuat cipherteks tidak menghasilkan pola walaupun terdapat pola di plainteks.

V. KESIMPULAN DAN SARAN

A. Kesimpulan

Protokol ini aman digunakan untuk mengamankan isi SMS. Protokol ini memudahkan pengirim dan penerima pesan karena tidak perlu mencari kanal aman untuk pertukaran kunci. Pertukaran kunci ditangani oleh protokol ini dengan algoritma ECDHE_ECDSA.

Protokol ini aman dari berbagai serangan. Penggunaan mode CFB membuat protokol ini aman dari serangan *padding oracle ,* modifikasi isi cipher dan pencarian pola pada cipherteks. Penggunaan ECDHE_ECDSA untuk pertukaran kunci membuat protokol ini mempunyai fitur *perfect forward secrecy*.

B. Saran

Isi pesan sebaiknya menggunakan *encoding* yang hemat tempat, seperti *ecoding* 7 bit yang umum digunakan SMS atau buatlah *encoding* yang lebih hemat tempat lagi. Satu SMS hanya bisa digunakan untuk mengirim 140 bytes dan sebagian sudah digunakan untuk *header Secure SMS*.

Panjang *ephemeral keys* untuk ECDHE sebaiknya dipanjangkan. Hal ini perlu dilakukan agar pertukaran kunci dapat dilakukan dengan aman.

REFERENSI

[1] Advanced Wireless Planet, “SMS PDU mode.” [online]. (http://www.gsm-mode.de/sms-pdu-mode.html diakses pada 8 Mei 2015)

[2] Anoop MS, “Elliptic Curve Cryptography An Implementation Guide”.

[3] Blake-Wilson et al., “Elliptic curve cryptography (ECC) cipher suites for transport layer security (TLS)”, Mei 2006. [Online] (https://tools.ietf.org/html/rfc4492#section-2.2 diakses pada 9 Mei 2015)

[4] Herong Yang, “Cryptography Tutorials - Herong's Tutorial Examples”, 2014. [Online] (http://www.herongyang.com/Cryptography/DES-Mode-CFB-Cipher-FeedBack.html diakses pada 9 Mei 2015)

[5] Robert Heaton, “The Padding Oracle Attack – Why Crypto is Terrifying”, 29 Juli 2013. [Online] (http://robertheaton.com/2013/07/29/padding-oracle-attack/ diakses pada 8 Mei 2015)

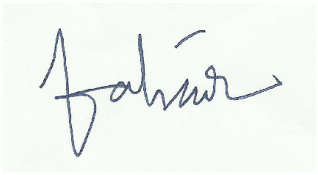
[6] Scott Helme, “Perfect forward secrecy- an introduction” , 10 Mei 2014. [Online] (https://scotthelme.co.uk/perfect-forward-secrecy/ diakses pada 10 Mei 2015)

[7] Damien Giry, “Cryptographic key length recommedation”, 2015. [Online] (http://www.keylength.com/en/compare/ diakses pada 10 Mei 2015)

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 11 Mei 2015



Fahziar Riesad Wutono (13512012)