PENGAMANAN KOMUNIKASI SUARA MELALUI INTERNET PADA TELEPON SELULER DENGAN ALGORITMA TEA PADA *PLATFORM* ANDROID

Denver[#1], Dr. Ir. Rinaldi Munir, M. T.[#2]
[#]*School of Electrical Engineering and Informatics, Institute Technology of Bandung, 10th Ganeca Street, Bandung, Indonesia.*
[1]*deng37@gmail.com*
[2]*rinaldi@informatika.org*

***Abstrak*— Komunikasi suara pada telepon seluler melalui internet sudah mulai berkembang. Keuntungan yang diperoleh ialah rendahnya biaya yang dibutuhkan. Tetapi keuntungan ini perlu dibayar dengan lemahnya proteksi. Oleh karena itu, dibuatlah suatu pengaman untuk mengamankan komunikasi suara tersebut.**

Dalam tugas akhir ini, dibuat sebuah aplikasi pengamanan komunikasi suara dengan menggunakan *Tiny Encryption Algorithm* (TEA). Aplikasi yang dibangun ditujukan untuk telepon seluler dengan *platform* Android. Sebelum aplikasi dibuat, akan dilakukan analisis terkait masalah yang muncul dan perancangan solusinya. Kemudian dilakukan analisis terkait kebutuhan sistem dan perancangan perangkat lunak. Pengujian dilakukan pada dua perangkat Android. Pengujian ini meliputi tiga aspek yaitu pengujian *delay* telepon dengan enkripsi dekripsi, pengujian *delay* telepon tanpa enkripsi dekripsi, dan pengujian hasil enkripsi.

Hasil pengujian memperlihatkan bahwa *delay* lebih besar dari batas yang ditentukan. Tetapi hal ini dapat ditoleransi karena nilai lebih yang dimilikinya. Nilai lebih ini adalah pengamanan yang dilakukan pada komunikasi yang berjalan.

Kata Kunci — komunikasi suara, telepon seluler, internet, TEA, enkripsi

I. LATAR BELAKANG

Pada jaman sekarang ini, orang-orang semakin mudah melakukan komunikasi suara. Dengan adanya telepon seluler, komunikasi suara dapat dibangun dengan *mobile*. Komunikasi suara ini sudah berkembang memasuki dunia internet. Adapun keuntungan komunikasi suara pada jaringan internet adalah biaya yang murah ketimbang komunikasi suara melalui *Public Switched Telephone Network* (PSTN).

Karena komunikasi ini berjalan dalam jaringan internet, tingkat keamanan menjadi rendah. Penyadapan lebih mudah dilakukan. Dalam melakukan penyadapan ini biasanya memiliki tujuan untuk mendapatkan informasi yang tidak seharusnya didapatkan oleh penyadap ini. Dengan didapatnya informasi ini, maka pihak yang melakukan komunikasi ini akan merasa dirugikan. Oleh karena itu, perlu dilakukan pengamanan terhadap komunikasi suara tersebut.

Pada tugas akhir ini akan dibuat sebuah aplikasi yang dapat melakukan komunikasi suara dengan paket-paket suara yang dikirimkan, diamankan terlebih dahulu. Aplikasi ini akan dibangun pada telepon seluler dengan *platform* Android. Android merupakan salah satu sistem operasi pada *smartphone*. Berbagai perusahaan telepon seluler menggunakan Android sebagai sistem operasinya.

Pengamanan yang dilakukan pada komunikasi suara dapat bermacam-macam. Metode jaman dahulu yakni dengan mengacaukan sinyal masukan seperti pada penelitian [ISL09]. Adapun metode pengamanan yang diterapkan dengan metode kriptografi modern. Pengamanan akan dilakukan pada masing-masing bit masukan, melewati serangkaian pengamanan, kemudian menghasilkan keluaran yang sama sekali berbeda dengan masukan.

Algoritma yang dipakai untuk pengamanan komunikasi suara di dalam Tugas Akhir ini adalah algoritma *Tiny Encryption Algorithm* (TEA). Algoritma TEA merupakan algoritma *cipher* block dengan ciri khas berupa kode yang tidak panjang tetapi kuat, dan cepat. Kekuatan algoritma ini sekompleks *Data Encryption Standard* (DES) dan kesederhanaannya membuat algoritma ini dapat ditranslasikan pada berbagai bahasa serta digunakan pada berbagai macam alat komputasi. [WHE94].

II. DASAR TEORI

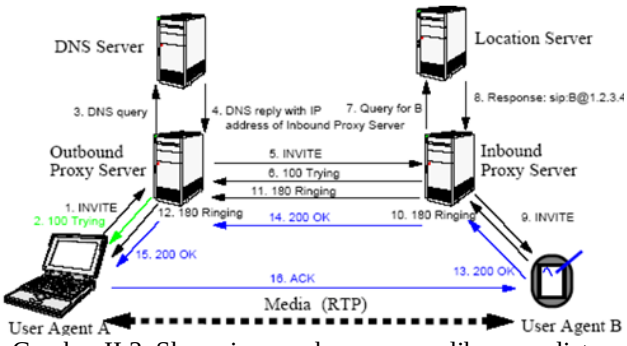
A. Komunikasi Suara pada Internet

Komunikasi suara yang dibangun pada internet membutuhkan dua tahapan. Tahapan yang pertama berupa *call setup* yakni pengaturan telepon yang mencakup semua detil terkait komunikasi agar komunikasi dapat berjalan. Tahapan kedua yang berupa pengiriman data melalui internet, menggunakan protokol SIP yang beroperasi pada layer aplikasi. Arsitektur dari SIP adalah *user agent* dan *server*. *User agent* merupakan aktor dari

sistem dan memiliki dua subsistem yaitu *user agent client* (UAC) dan *user agent server* (UAS). UAC ini berfungsi untuk membangkitkan *request* dan UAS berfungsi untuk merespon *request*. Sedangkan *server* ini sendiri memiliki empat bagian menurut [CAR04] yakni

1. *Proxy server*: merupakan *host* dari jaringan berfungsi meminta *request* atas nama *client* yang lain. *Proxy server* ini harus dapat bertindak sebagai *client* maupun *server*, serta dapat mengarahkan *request* pada UAC dan UAS. *Proxy server* harus dapat melakukan *routing*, memastikan setiap *request* yang ada sampai pada penerima yang memiliki hak tersebut.
2. *Redirect server*: merupakan *server* yang berfungsi untuk mengalihkan *request-request* yang ada pada perangkat pengganti dari *Uniform Resource Indicators* (URIs).
3. *Registrar server*: merupakan *server* yang mampu untuk menerima dan memproses pesan pendaftaran dari suatu titik akhir yang dapat diketahui lokasinya. *Registrar server* ini bekerja sama dengan *location server*.
4. *Location server*: merupakan *server* yang mampu menterjemahkan alamat pada *database* yang ada pada *domain* jaringan.

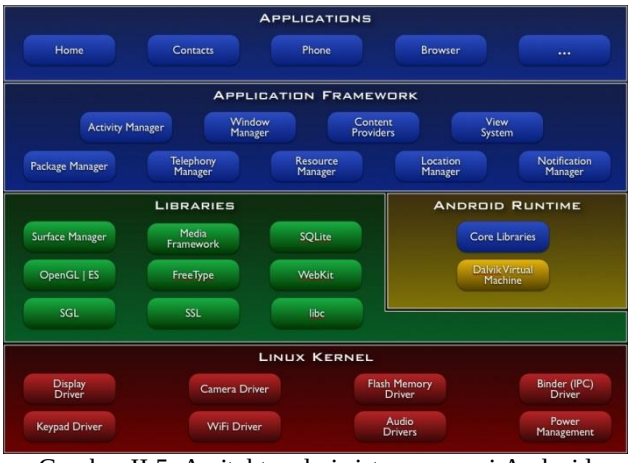
Pada Gambar II.3 [TON05] digambarkan percakapan menggunakan protokol SIP antara *user agent* A dan B. Masing-masing *user agent* memiliki *proxy server* dan DNS *server*. Kedua *proxy server* meminta alamat dari DNS *server*. Alamat ini akan dipertukarkan untuk kemudian dipakai masing-masing *user agent* untuk mencapai *user agent* yang lain. Komunikasi selanjutnya akan berjalan pada protokol RTP.



Gambar II.3. Skenario percakapan yang dibangun diatas SIP

B. Android

Android memiliki arsitektur seperti pada Gambar II.5 [AND12]. Pada gambar tersebut dapat dilihat ada lima bagian yakni *applications, application framework, libraries, android runtime*, dan Linux *kernel*. Bagian *applications* terdiri dari aplikasi-aplikasi dimana setiap aplikasi menjalankan fungsinya tertentu.

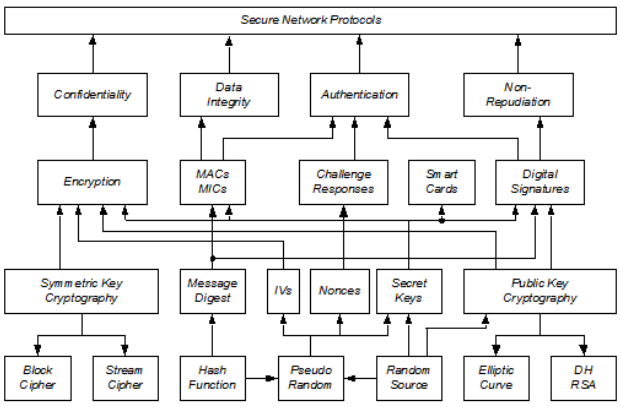


Gambar II.5. Arsitektur dari sistem operasi Android

C. Algoritma Kriptografi *Modern*

Ide dasar algoritma kriptografi modern ialah berasal dari algoritma klasik. Algoritma kriptografi klasik memiliki 2 ide dasar yakni *cipher* substitusi dan *cipher* tranposisi/permutasi. *Cipher* substitusi melakukan pertukaran masing-masing karakter dengan karakter-karakter lain yang berbeda. *Cipher* transposisi melakukan pengubahan posisi yang terdapat pada karakter-karakter. Dengan menggabungkan substitusi dan tranposisi pada bit-bit karakter maka muncullah algoritma kriptografi modern ini [BIS03].

Algoritma kriptografi modern memproses baik *plaintext, ciphertext*, maupun kunci dalam rangkaian bit. Operasi yang seringkali digunakan adalah operasi bit XOR(*exclusive* OR). Operasi bit ini memiliki maksud semua data masukan akan diproses dalam bentuk bit *biner* (nol dan satu). Dokumen seperti makalah atau laporan, dalam media digital, telah mempunyai representasi *biner*. Hanya saja yang tertampil dalam representasi karakter, yang dapat dibaca. Setelah melakukan enkripsi dalam representasi *biner* maka hasil enkripsi/*ciphertext* tidak dapat terbaca lagi, menjadi kacau. Gambar II.6 memperlihatkan diagram blok pada kriptografi modern [MUN05]

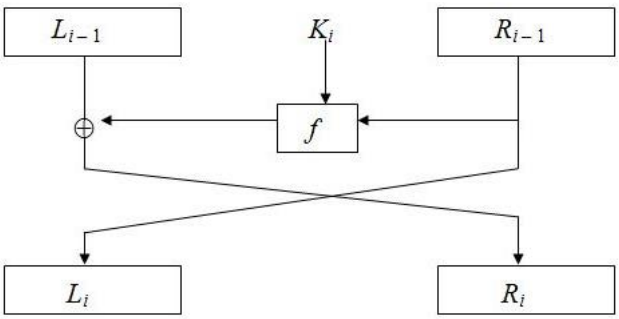


Gambar II.6. Diagram Blok Kriptografi Modern

Cipher blok merupakan algoritma modern yang beroperasi pada sejumlah blok bit yang telah ditentukan. Masing-masing blok bit tersebut akan diproses sendiri-

sendiri. *Cipher* blok ini menyajikan pengamanan lebih baik ketimbang *cipher* aliran, tetapi membutuhkan waktu lebih banyak untuk memroses ketimbang dengan *cipher* aliran. Hal ini disebabkan *cipher* blok harus menunggu satu blok terpenuhi untuk dapat diproses. Skema *cipher* blok dapat dilihat pada gambar II.7 [MUN05].

Kunci dari *cipher* blok merupakan kunci yang tetap, *cipher blok* ini tidak seperti *cipher* aliran memiliki *keystream generator*, penghasil kunci untuk setiap masukan. Kunci yang dihasilkanpun berbeda-beda. Kunci pada *cipher* blok memiliki panjang yang sama dengan panjang blok itu sendiri.



Gambar II.7. Konsep *cipher* blok dengan fungsi *round*

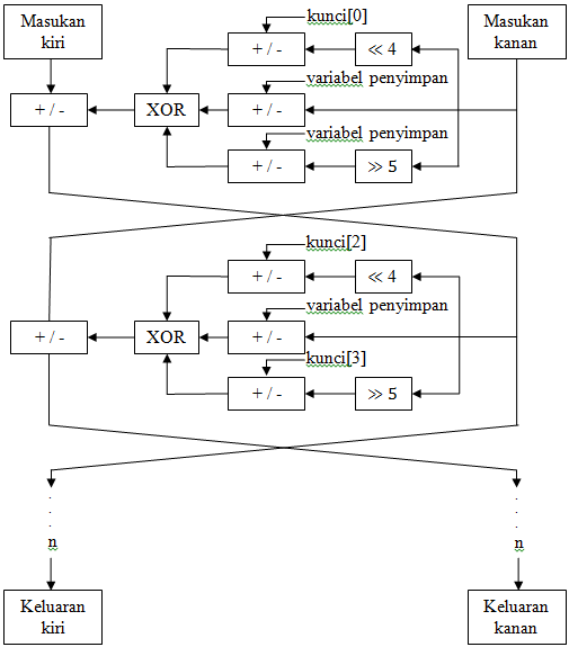
Dalam perancangannya, terdapat lima prinsip yang digunakan yakni *confusion – diffusion*, *cipher* berulang, *substitution box*, kunci lemah, dan jaringan *Feistel*. Jaringan *Feistel* dipakai pada berbagai macam algoritma karena bersifat *reversible*. *Reversible* disini mengandung makna proses enkripsi dan dekripsi hampir serupa, hanya berkebalikan. Hal ini mengakibatkan jaringan *Feistel* ini banyak digunakan pada berbagai algoritma. Jaringan *Feistel* merupakan perulangan *cipher* dimana didalamnya terdapat fungsi yang dinamakan fungsi *round*. Perulangan sebanyak satu kali ditampilkan pada gambar II.7.

D. Algoritma TEA

Algoritma TEA merupakan salah satu algoritma *cipher* blok. Algoritma ini diajukan pada *Fast Software Encryption*, dan telah mengalami beberapa perubahan. Mulai dari TEA versi asli, kemudian dikembangkan menjadi *extended* TEA (XTEA), dan perubahan terakhir menjadi *corrected block* TEA (XXTEA). Algoritma ini dibuat dengan tujuan menghasilkan suatu algoritma dengan penggunaan memori sekecil-kecilnya untuk mendapat tingkat peformansi dan pengamanan yang baik. Dengan kesederhanaannya, diharapkan algoritma TEA ini mampu diimplementasikan pada sebagian besar alat komputasi [WHE94].

Algoritma TEA memiliki masukan sebesar 64 bit integer dan menggunakan kunci 128 bit. Struktur algoritma TEA adalah jaringan *Feistel* dengan menggunakan operasi penambahan dan pengurangan. Operasi ini berjalan pada fungsi *round*. Fungsi ini dipanggil setiap kali melakukan iterasi. Iterasi ini terbagi menjadi dua bagian yang berjalan secara bersamaan. Masukan sebesar 64 bit terbagi menjadi dua bagian, masing-masing sebesar 32 bit. Iterasi ini akan berjalan sebanyak 32 kali. Konstanta *magic* yang digunakan adalah 2654435769, digunakan untuk mencegah serangan yang berdasarkan simetri putaran. Keluaran algoritma TEA sama besarnya dengan masukan yakni 64 bit yang terdiri dari dua bagian masing-masing 32 bit.

Skema pada gambar II.9 merupakan gambaran dari fungsi *round* dari iterasi pertama sampa iterasi terakhir. Fungsi *round* ini dapat berupa enkripsi atau dekripsi. Hanya berbeda pada operasi penambahan dan pengurangan saja. Operasi penambahan bekerja pada enkripsi, sedangkan operasi pengurangan bekerja pada dekripsi.



Gambar II.9. Skema fungsi *round* pada iterasi pertama sampai iterasi terakhir

Berikut dibawah merupakan tahapan yang dilakukan untuk proses enkripsi:

1. Tahap pertama: Menambahkan variabel penyimpan dengan masukan konstanta *magic* dan menyimpannya ke dalam dirinya sendiri.
2. Tahap kedua: Hasil keluaran bagian kiri terdiri dari hasil penambahan masukan kiri dengan operasi XOR pada tiga bagian. Bagian pertama menambahkan kunci bagian pertama dengan masukan kanan yang digeser ke kiri sebanyak empat kali. Bagian kedua menambahkan variabel penyimpan dengan masukan kanan. Bagian ketiga menambahkan kunci bagian kedua (kunci dibagi menjadi empat bagian) dengan masukan kanan yang digeser ke kanan sebanyak lima kali.
3. Tahap ketiga: Hasil keluaran bagian kanan terdiri dari hasil penambahan masukan kanan dengan operasi XOR pada tiga bagian. Bagian pertama menambahkan kunci bagian ketiga dengan masukan kiri yang digeser ke kiri sebanyak empat kali. Bagian kedua menambahkan variabel penyimpan dengan masukan kiri. Bagian ketiga menambahkan kunci

bagian keempat dengan masukan kiri yang digeser ke kanan sebanyak lima kali.

Untuk proses dekripsi sendiri hanya dengan mengganti operasi penambahan dengan operasi pengurangan.

E. Algoritma SNOW dan KASUMI

Algoritma SNOW merupakan salah satu algoritma *cipher* aliran. Algoritma ini diajukan pada *New European Schemes for Signatures, Integrity, and Encryption* (NESSIE) dan algoritma ini telah mengalami beberapa perubahan. Mulai dari SNOW 1.0, SNOW 2.0, sampai dengan SNOW 3G. Algoritma SNOW yang dipakai disini adalah SNOW 3G.

Algoritma SNOW 3G terdiri dari dua bagian yakni *Linear Feedback Shift Register* (LFSR) dan *Finite State Machine* (FSM). LFSR merupakan *shift register* yang masukannya adalah hasil keluaran dari fungsi dari keadaan sebelumnya. Terkadang keluaran LFSR ini dikombinasikan dengan XOR. Proses enkripsi dan dekripsi yang dilakukan pada algoritma ini serupa [ORH10].

Algoritma KASUMI merupakan algoritma *cipher* blok. Algoritma ini dibuat oleh 3rd *Generation Partnership Project* (3GPP) untuk mengamankan komunikasi pada telepon seluler. Struktur algoritma KASUMI adalah jaringan Feistel dengan masukan sebesar 64 bit dan keluaran sebesar 64 bit. Kunci yang digunakan sebesar 128 bit. Algoritma KASUMI terdiri dari beberapa sub fungsi-fungsi yang digunakan bersama-sama dengan beberapa sub kunci-kunci [ETS99].

III. PEMBAHASAN

A. Analisis Masalah dan Perancangan Solusi

Berikut merupakan berbagai permasalahan yang perlu diperhatikan:

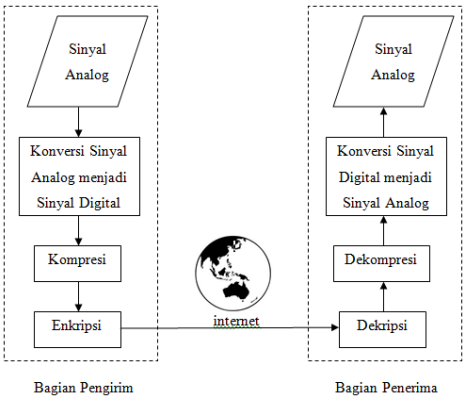
1. Penggunaan API dari Android memberikan kemudahan dalam membangun komunikasi melalui protokol internet. Dalam Tugas Akhir akan dilakukan pengamanan pada bit-bit paket suara yang akan dikirim. Tetapi diperlukan suatu pengaksesan pada paket-paket yang akan dikirim. API Android tidak dapat diakses sehingga dapat dipilih aplikasi Sipdroid [GOO09]. Aplikasi ini membangun sendiri komunikasi suara melalui internet. Sehingga penyisipan enkripsi dan dekripsi dapat dilakukan.
2. Penyedia layanan SIP dibagi menjadi berbayar dan tidak berbayar. Untuk mencapai tujuan nilai ekonomis yang rendah, dipilih layanan yang tidak berbayar. Tetapi layanan SIP yang tidak berbayar ini ada berbagai kendala yang bisa terjadi seperti adanya batasan durasi telepon, *server* yang suka mati, hingga proses pendaftaran yang menyulitkan pengguna. SIP Linphone dipilih karena layanan SIP tidak berbayar ini tidak memiliki kendala yang disebutkan.
3. Dalam membangun komunikasi melalui internet, terdapat dua jenis sambungan yakni *Transmission Control Protocol* (TCP) dan *User Datagram Protocol* (UDP). TCP cenderung lebih lama dalam membangun koneksi, tetapi koneksi lebih terjamin pada saat komunikasi sudah berjalan. TCP dipilih karena jaminan komunikasinya.
4. Berdasarkan implementasi algoritma TEA, SNOW, dan KASUMI, didapatkan bahwa algoritma SNOW dan KASUMI tidak dapat dipakai. Kedua algoritma ini menghasilkan lama waktu enkripsi dan dekripsi yang terlalu besar. Lama waktu yang dihasilkan dari proses enkripsi dan dekripsi tidak boleh terlalu besar, bila terlalu besar maka paket-paket suara yang ada menjadi pecah, tidak dalam satu kesatuan dan komunikasi tidak dapat dilakukan. Dengan algoritma TEA lama waktu enkripsi dekripsi jauh lebih kecil dan komunikasi suara dapat berjalan.
5. Kompresi paket dapat dilakukan secara *lossy* maupun *lossless*. Paket-paket yang hilang dapat menyulitkan proses enkripsi dan dekripsi karean urutannya menjadi tidak sesuai. Penggunaan kompresi *lossless* dirasa lebih tepat.
6. Sekalipun metode pengompresian *lossless*, tidak dapat menjamin semua paket sampai. Hal utama yang dapat mempengaruhi ialah masalah konektifitas internet yang lambat. Karena dimungkinkan adanya paket hilang, diperlukan penanganan pada saat dekripsi dilakukan.
7. Kunci yang menjadi masukan pada saat enkripsi dan dekripsi dilakukan, perlu ditangani apabila kunci masukan pengguna tidak sesuai ukuran yang diperlukan. Fungsi MD5 menghasilkan keluaran yang sesuai sebagai masukan kunci enkripsi dan dekripsi.

B. Analisis Perangkat Lunak

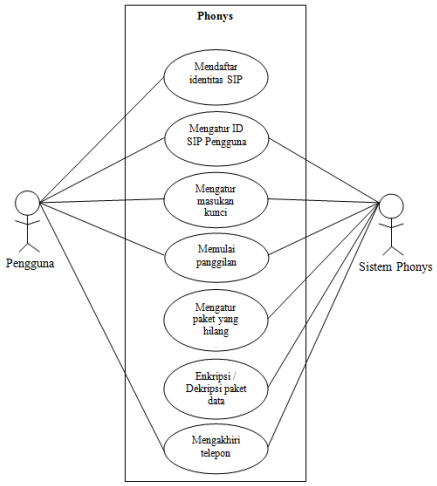
Perangkat lunak yang dibangun diberi nama Phonys. Perangkat lunak ini terdapat dua bagian yakni bagian pengirim dan bagian penerima, ditunjukan pada gambar III.1.

Dari hasil analisis yang dilakukan, dapat dibuat diagram *use case*. Diagram *use case* yang ditunjukkan pada gambar IV.1, memiliki dua aktor yakni pengguna dan sistem Phonys. Pengguna memiliki lima *use case* dan sistem Phonys memiliki enam *use case*.

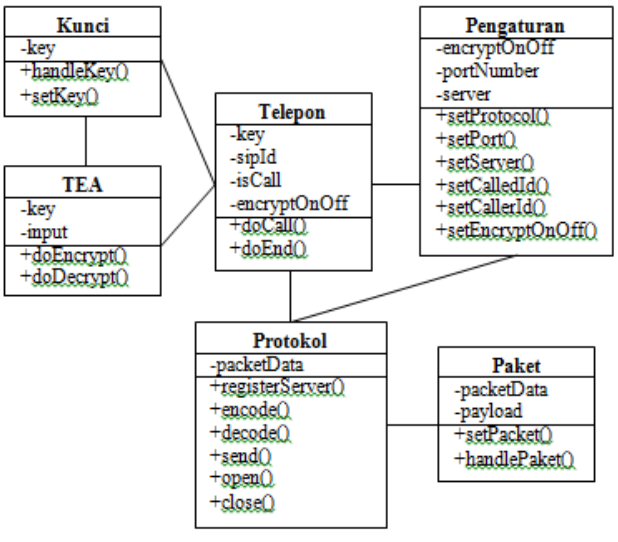
Adapun kelas-kelas yang dibangun memiliki rancangan seperti ditunjukkan pada gambar IV.2. Terdapat enam kelas, masing-masing kelas saling terkait dengan kelas yang lain.



Gambar III.1. Arsitektur Phonys



Gambar IV.1. Diagram *Use Case*



Gambar IV.2. Diagram Rancangan Kelas Phonys

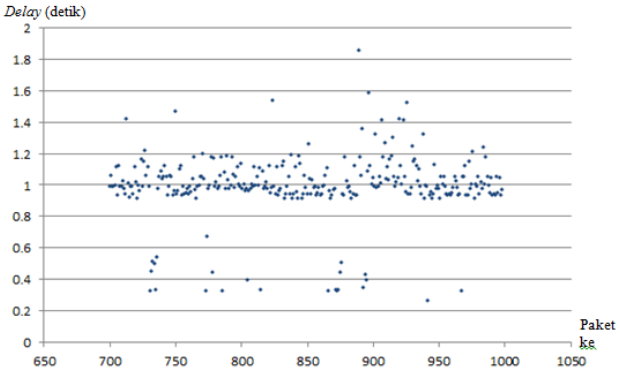
IV. PENGUJIAN

Pengujian dibagi menjadi tiga bagian yakni pengujian *delay* dengan enkripsi dekripsi, pengujian *delay* tanpa enkripsi dekripsi, dan pengujian hasil enkripsi. Pengujian ini dilakukan pada dua telepon seluler Android yakni Samsung Galaxy Tab P3100 dan SNexian Mi320. Komputer jinjing Asus U36S digunakan untuk menangkap paket-paket yang dikirim.

A. Pengujian *Delay* dengan Enkripsi dan Dekripsi

Pengujian *delay* dengan enkripsi dan dekripsi dilakukan pada kedua perangkat dengan menyalakan Wi-Fi (untuk SNexian Mi320) dan menyalakan *packet data* (untuk Samsung P3100), menjalankan aplikasi Phonys, melakukan pengaturan ID SIP, menunggu sampai aplikasi Phonys terhubung ke *server* (ditandai dengan berubahnya bulatan kuning pada jendela notifikasi menjadi berwarna hijau), dan memasukan kunci. Setelah itu pada Samsung P3100 (sip:byupayoy@sip.linphone.org) diisi alamat SIP SNexian (sip:deng37@sip.linphone.org) dan panggilan dilakukan. Untuk Samsung P3100 perlu mencentang bagian "Use 3G" karena jaringan yang digunakan adalah internet dari kartu Telkomsel Halo.

Komputer jinjing ini menjalankan program Eclipse untuk mencatat lama waktu yang dibutuhkan dari proses enkripsi, pengiriman, sampai dekripsi. Sampel data yang digunakan sebanyak tiga ratus paket, dari paket ke tujuh ratus sampai paket ke seribut. Persebaran data yang digunakan dapat dilihat pada gambar V.4. Dari ketiga ratus paket tersebut, waktu *delay* tersebut dirata-ratakan dan didapatkan hasil tertera pada tabel V.2.



Gambar V.4: Grafik Persebaran Data *Delay* dengan Enkripsi dan Dekripsi

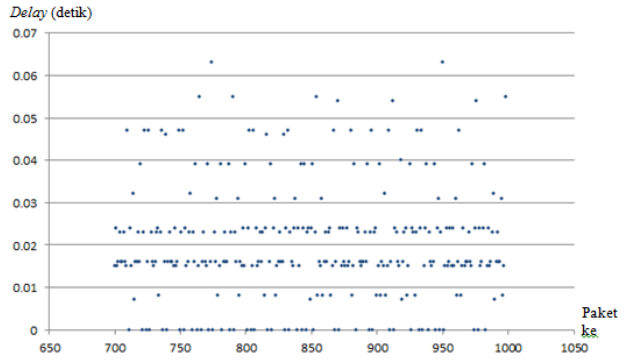
Table V.2: Tabel Hasil Pengujian *Delay* dengan Enkripsi dan Dekripsi

Kasus Uji	Rata-rata *Delay* (mili detik)
Pengukuran *delay* dengan enkripsi dan dekripsi	989,686

B. Pengujian *Delay* tanpa Enkripsi dan Dekripsi

Pengujian kedua ini mengukur *delay* yang dihasilkan tanpa adanya proses enkripsi dan dekripsi. Skema pengujian yang digunakan sama dengan pengujian pada V.2.3.1, hanya saja terdapat langkah yang berbeda. Kunci tidak perlu dimasukan dan tombol "Encryption" ditekan sehingga status enkripsi menjadi "off".

Komputer jinjing menjalankan program Eclipse untuk mencatat *delay*. Sampel data yang digunakan sebanyak tiga ratus, dari paket ke tujuh ratus sampai paket ke seribu. Persebaran data dapat dilihat pada gambar V.5. Dari ketiga ratus paket tersebut, *delay* dirata-ratakan dan didapatkan hasil tertera pada tabel V.3.



Gambar V.5: Grafik Persebaran Data *Delay* tanpa Enkripsi dan Dekripsi

Table V.3: Tabel Hasil Pengujian *Delay* tanpa Enkripsi dan Dekripsi

Kasus Uji	Rata-rata *Delay* (mili detik)
Pengukuran *delay* tanpa enkripsi dan dekripsi	20,013

C. Pengujian Hasil Enkripsi

Pengujian terakhir, menentukan apakah *ciphertext* dengan plainteks merupakan data yang berbeda. Pengujian ini dilakukan dengan menggunakan program Wireshark, digunakan untuk menangkap paket yang dikirim. Paket yang ditangkap Wireshark ini dibandingkan dengan paket belum dienkripsi dan belum dikirim, dicatat pada program Eclipse. Hasil penangkapan paket untuk nomor paket ke dua ratus ditampilkan pada tabel V.4.

Table V.4: Tabel Perbandingan Paket Setelah Dikirim dan Sebelum Enkripsi

Paket Setelah Dikirim	ce 38 42 8f 0f 01 55 01 ce 38 42 8f 0f 01 55 01 18 f3 02 d1 5d d3 24 77 0c fa 3e 36 d3 e7 75 6e ec 29 8d 0e 13 7b 7b 19 65 e6 14 23 d3 17 a3 7a c0 12 4d 86 d1 33 65 f5 28 c4 3c ba cd 49 b8 b1 ec 11 3c f7 58 f0 26 3e 73 3e e6 11 a3 f0 d2 5c 60 f7 43 63 9c 4f 2d 8a d3 97 11 62 16 35 21 4e ce a3 5d 1c 39 d5 d7 06 fc 49 71 3b da 7c 10 e0 a7 c6 49 cd 6e dc f0 b6 d8 b2 39 6d c0 f9 0b 75 59 4b 4b 7d cb 23 fe 6b ba 65 f4 ea 3a da 6f f9 13 e4 6b 8b a5 ff fd ae b2 14 c3 07 39 d0 3a d9
Paket Sebelum Enkripsi	d5 55 d5 d5 54 55 55 54 d5 55 55 55 55 d5 d5 d5 d5 d5 55 55 d5 d5 d5 55 d5 d4 d4 55 55 55 55 55 55 55 55 55 55 55 55 55 55 d5 d5 55 55 d5 d5 d5 55 d5 55 55 55 55 55 54 54 55 55 55 54 55 55 d5 d5 d5 d4 d5 d5 d5 d5 d5 d5 55 55 d5 55 54 55 55 d5 55 55 55 55 d5 55 d5 55 d5 55 54 d5 d5 55 d5 d5 d5 d5 d4 d5 d5 d5 55 55 55 55 55 54 54 54 55 55 55 55 55 55 d5 d4 d4 d4 d4 d5 d5 d5 55 55 55 54 55 55 55 54 54 55 55 55 54 55 55 d5 55 55 55 55 d5 d5 d5 55 d5 55 54 55 d5 d5 d5 55 55 d5 d5 55 55 55

D. Analisis Hasil Pengujian

Menurut ITU tertera pada spesifikasi G.114 (International Telecommunication Union, 1993), direkomendasikan *delay* sebesar 150 mili detik untuk kualitas tinggi yang *real time*. Adapun untuk panggilan internasional memiliki rekomendasi *delay* sebesar 300 mili detik. Diambil nilai rokemendasi 300 mili detik karena *server* yang digunakan berada di luar negeri, Perancis.

Hasil pengujian didapatkan bahwa untuk menyampaikan satu paket dari mulai dari enkripsi, pengiriman paket, dan dekripsi dibutuhkan waktu 989,686 mili detik. *Delay* yang dihasikan kurang lebih tiga kali lebih lambat ketimbang nilai rekomendasi.

Akan tetapi dalam standar tersebut (International Telecommunication Union, 1993) disebutkan bahwa bila *delay* melebihi angka rekomendasi, masih dapat diterima jika memiliki suatu nilai lebih. Nilai lebih tersebut dapat berupa kualitas yang baik ataupun jarangnya komunikasi terputus. Aplikasi yang dibangun memberikan nilai lebih berupa pengamanan pada komunikasinya. Adapun pada saat pengujian belum pernah mengalami terputusnya komunikasi, walaupun koneksi internet pada saat tersebut relatif buruk.

Delay yang dihasilkan dengan dan tanpa enkripsi dekripsi memiliki selisih waktu sebesar 969,672 mili detik. Hal ini dapat dilihat bahwa peran enkripsi dan dekripsi dalam menghasilkan *delay* sangat besar. *Delay* yang berasal dari enkripsi dekripsi ini dapat berkurang nilainya bila menggunakan perangkat Android dengan kemampuan prosesor lebih baik.

Delay yang dihasilkan tidak boleh terlalu besar. Bila *delay* memiliki nilai yang sangat besar maka suara tidak dapat didengar lagi. Suara yang tidak terdengar ini diakibatkan paket-paket yang diterima menjadi pecah, bukan lagi merupakan satu kesatuan. Suara dapat terdengar bila jarak antar paket cukup berdekatan, bila jarak antar paket jauh, maka yang terdengar hanyalah potongan suara. Potongan-potongan suara ini bila didengar akan menyerupai suara terenkripsi yang tidak didekripsi, bising.

Dari pengujian dengan telepon terenkripsi, dapat dilihat bahwa aplikasi ini dapat melakukan komunikasi antara kedua perangkat dimana salah satu perangkatnya memiliki spesifikasi rendah. Telepon seluler SNexian Mi320 merupakan salah satu telepon seluler Android Gingerbread dengan spesifikasi rendah. Tapi aplikasi Phonys dapat berjalan diatas telepon seluler ini. Hal ini dikarenakan pemilihan algoritma TEA yang tepat. Algoritma ini ringan sehingga *processor* SNexian Mi320 yang tidak hebat sekalipun dapat berjalan dengan cepat.

Pengujian delay dengan menggunakan kartu Telkomsel Halo dapat dilakukan dengan syarat sinyal yang didapat minimal pada rentang 3G keatas. Hal ini dapat dilakukan dengan melakukan telepon diluar bangunan berbeton. Bangunan bertingkat sekarang memiliki beton-beton yang dapat menghalangi sinyal masuk. Adapun bangunan bertingkat yang modern memiliki *transmitter* sehingga sinyal yang didapat tetap baik.

Koneksi *Asymmetric Digital Subsciber Line* (ADSL) Telkom Speedy pada pengujian baik untuk digunakan. Koneksi ini menggunakan sambungan telepon PSTN. Sambungan ini lebih stabil ketimbang menggunakan jaringan telepon seluler. Telepon PSTN tidak mengenal adanya derajat sinyal yang diterima seperti halnya telepon seluler.

Pengujian langsung untuk mengetahui bahwa paket benar terenkripsi dilakukan dengan mengatur salah satu perangkat pada mode terenkripsi, dan perangkat satu lagi dalam mode tidak terenkripsi. Hasil keluaran yang didengar berupa suara bising. Suara bising ini merupakan suara asli yang terenkripsi tapi tidak didekripsi, karena pada penerima mode enkripsi dimatikan. Suara bising juga akan dihasilkan bila kedua belah pihak memasukan kunci yang berbeda.

.

V. KESIMPULAN

Pada pengerjaan Tugas Akhir ini, dapat diperoleh beberapa kesimpulan sebagai berikut:

1. Pengamanan dapat dilakukan pada komunikasi suara yang dibangun diatas jaringan internet, algoritma TEA sebagai pengamanannya, dan telepon seluler Android dengan versi minimal 2.3 (Android Gingerbread).
2. Dengan algoritma TEA, pengamanan yang dilakukan tidak merusak jalannya komunikasi suara.
3. *Delay* yang dihasilkan dari komunikasi suara dengan enkripsi dan dekripsi memiliki yakni sebesar 989,686 mili detik. *Delay* ini melebihi batas rekomendasi (300 mili detik), tetapi masih dapat ditoleransi karena nilai lebih yang diberikan berupa pengamanan yang dilakukan pada komunikasi suaranya.
4. Perbedaan *delay* antara komunikasi suara dengan dan tanpa enkripsi dekripsi adalah 969,672 mili detik. Angka ini mendekati *delay* komunikasi suara dengan enkripsi dan dekripsi yang berarti proses enkripsi dan dekripsi mempengaruhi hampir semua *delay* yang dihasilkan.
5. Data suara yang dienkripsi benar aman karena antara data yang dikirim (data yang telah dienkripsi) dengan data sebelum dienkripsi adalah berbeda.

DAFTAR PUSTAKA

[BIS03] Bishop, David (2003). *Introduction to Cryptography with Javatm Applets*. Massachusetts, Jones and Bartlett Publishers.

[CAR04] Carlson, Ian, Charles Avila (2004). *Voice over IP (VoIP)/SIP Infrastructure Considerations for the Interaction Center Platform*. Interactive Intelligence.

[ETS99] ETSI/SAGE (1999). KASUMI Specification: *Specification of the 3GPP Confidentiality and Integrity Algorithms Document 2*. Sophia Antipolis, France.

[GOO09] Google Code (2009): Free SIP/VoIP Client for Android, https://code.google.com/p/sipdroid/, *download*(diturunkan/diunduh) pada 27 April 2013.

[ISL09] Islam, Saad, Fatima Ajmal, Salman Ali, Jawad Zahid, Adnan Rashdi (2009). *Secure End-to-End Communication over GSM and PSTN Networks*. IEEE International Conference, National University of Sciences and Technology, Pakistan.

[ORH10] Orhanau, Ghizlane, Said El Hajji, Youssef Bentaleb (2010). *SNOW 3G Stream Cipher Operation and Complexity Study*. Contemporary Engineering Sciences, Vol. 3, 2010, 97-111, Universite Mohammed V Agdal, Maroc.

[TON05] Tong, Hoang Anh (2005). *SIP-Based VoIP Service – Architecture and Comparison*. University of Stuttgart.

[WHE94] Wheeler, David J, Roger Needham (1994). *TEA, a tiny encryption algorithm*. Fast Software Encryption, Leuven, Belgium.

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, 1 Agustus 2013

ttd

Denver
13509056