Jurnal Sarjana Institut Teknologi Bandung bidang Teknik Elektro dan Informatika
Vol. 1, No. 3 Oktober 2012

PENGEMBANGAN *SECURED VIDEO STREAMING* PADA *SMARTPHONE* DENGAN *PLATFORM* ANDROID

Danang Tri Massandy[#1], Dr. Ir. Rinaldi Munir, M.T.[*2]

Teknik Informatika, Institut Teknologi Bandung
Jl. Ganesha 10, Bandung, Indonesia
[1]danangmassandy@gmail.com
[2]rinaldi@informatika.org

Abstrak— **Pada makalah ini, diajukan sebuah sistem perangkat lunak *video streaming* dari *smartphone* Android menuju ke komputer. Video ditangkap dari kamera *smartphone* dan secara *real-time* dikirimkan ke komputer. Pada komputer, video tersebut diputar dengan menggunakan aplikasi *video player*, contohnya adalah VLC VideoLan. Perkembangan informasi semakin cepat sehingga informasi yang dipertukarkan sangat sensitif dan penting, termasuk juga informasi yang ada pada video. Dengan adanya kriptografi, informasi dalam video dapat diamankan dengan melakukan proses enkripsi selektif terhadap data penting dalam video tersebut. Video yang populer saat ini adalah video dengan format H.264. Dalam pengenkripsian selektif dipilih data *slice* bertipe I sebagai bagian dari *I-Frame* untuk dienkripsi. Pada *video streaming* proses enkripsi selektif dilakukan pada *smartphone* Android sedangkan proses dekripsi selektif dilakukan pada komputer. Berdasarkan hasil pengujian, video dapat dienkripsi secara selektif dengan memilih data *slice* bertipe I untuk mengamankan video sehingga gambar video menjadi rusak dan sukar untuk dilihat. Namun, pada kondisi-kondisi tertentu masih terdapat gambar yang tidak rusak walaupun *slice* bertipe I telah dienkripsi.**

Kata kunci— ***video streaming*, enkripsi selektif, *smartphone* Android**

I. PENDAHULUAN

Video streaming memungkinkan seseorang dapat melihat atau menyaksikan suatu kejadian tanpa harus ada di tempat kejadian tersebut. Teknologi ini dapat dimanfaatkan baik untuk berkomunikasi antar keluarga, teman, keperluan perusahaan, ataupun pemerintahan. *Video streaming* membutuhkan dua hal utama yaitu jaringan dan perangkat. Jaringan diperlukan untuk mengirimkan data (video dan audio) dari pengirim ke penerima yaitu internet.

Sementara perangkat dapat berupa komputer, *laptop*, *tablet*, atau bahkan *smartphone*. Beberapa perangkat ini tentunya harus terhubung dengan jaringan yang memadai. Penggunaan perangkat seperti *smartphone* lebih mudah dan sering dibawa kemana-mana daripada komputer/*laptop*. Salah satu jenis *platform*/sistem operasi untuk *smartphone* adalah Android. *Platform* Android merupakan sistem operasi *mobile* yang pertama kalinya dikembangkan oleh Android Inc., dan kemudian diakuisisi oleh Google[1].

Penggunaan teknologi *video streaming* pada *smartphone* khususnya *platform* Android membuat semakin mudah untuk melaporkan suatu kejadian atau membagikan suatu *event* penting tersebut kepada orang lain. Melaporkan kejadian dengan teknologi *video streaming* dapat dilakukan kapan dan dimana saja karena perangkat mudah untuk dibawa kemana-mana.

Teknologi *video streaming* memungkinkan untuk mengirimkan rekaman video secara *real-time* dari Android ke komputer. Pengiriman data *stream* dapat melalui jaringan internet sehingga dapat dilihat oleh siapapun. Pada komputer, data *stream* tersebut dapat diputar menggunakan *video player*, seperti VLC VideoLan.

Penggunaan teknologi ini dapat dimanfaatkan misalnya untuk mengirimkan video konser yang sedang berlangsung sehingga orang lain dapat melihat konser tersebut tanpa harus datang ke tempat konser. Untuk melihat video yang dikirimkan secara *live* atau *real-time*, seseorang harus membayar biaya tertentu jika konser tersebut berlangsung secara tertutup.

Konten video yang ditransmisikan dapat bermacam-macam, mulai dari kegiatan sehari-hari, kejadian unik, atau yang bersifat pribadi dan rahasia yang menyangkut kepentingan perusahaan, pemerintahan, militer. Konten *video streaming* yang berbayar atau bersifat sangat rahasia ini memerlukan jaminan keamanan saat ditransmisikan pada jaringan internet.

Namun, kemanan data pada jaringan (internet) khususnya dalam hal kerahasiaan merupakan faktor keamanan yang perlu diperhatikan. Hal ini dikarenakan data yang ada pada internet menyebar ke area publik sehingga memungkinkan untuk terjadinya pencurian data. Jika pihak ketiga berhasil mendapat data dari *video streaming*, maka orang tersebut dapat memantau seluruh kegiatan yang terjadi termasuk informasi penting di dalamnya. Oleh karena itu, diperlukan suatu metode pengamanan data yang dikirim ke internet.

Metode pengamanan data dapat menggunakan salah satu algoritma enkripsi yang ada pada Kriptografi yaitu algoritma *Advanced Encryption Standard* (AES). AES merupakan standar enkripsi dengan kunci simetris yang algoritmanya diusulkan oleh J. Daemen dan V. Rijmen dengan nama Rijndael[2].

Penggunaan enkripsi untuk pengiriman data *video streaming* yang efisien harus memperhatikan sumberdaya pada perangkat *smartphone* yang terbatas. Efisiensi dinilai dengan dua kriteria yaitu waktu komputasi yang kecil agar proses pengiriman data *video* tidak terhambat dan penggunaan memori yang sesuai dengan sumberdaya pada *smartphone*.

Pada makalah ini akan diusulkan sebuah aplikasi enkripsi selektif untuk *video streaming* pada *smartphone* dengan *platform* Android. Bagian selanjutnya dari makalah ini sebagai berikut : studi literatur membahas beberapa dasar teori, implementasi membahas arsitektur dan implementasi dari aplikasi, pengujian membahas beberapa pengujian yang dilakukan dan hasil ujinya, analisis hasil pengujian membahas analisis dari hasil uji, dan kesimpulan yang menyimpulkan isi dari makalah ini.

II. STUDI LITERATUR

Pada makalah ini ada beberapa dasar teori yang akan dibahas, yaitu tentang video digital, protokol RTP, serta algoritma AES.

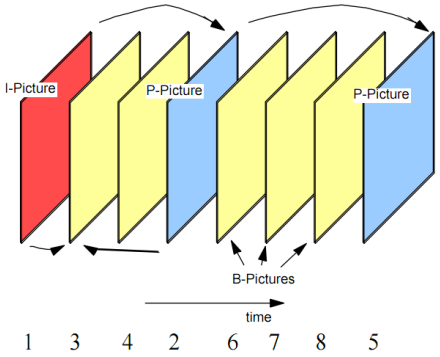
A. *Video Digital*

Video digital merupakan jenis perekaman video dengan menggunakan sinyal video digital. Video digital mulai diperkenalkan pada tahun 1990. Pada tahun tersebut, proses perubahan dari video analog menjadi video digital membutuhkan biaya sangat mahal pada perangkat keras untuk kompresi [3]. Oleh karena itu, dari tahun ke tahun metode kompresi untuk video digital semakin berkembang seperti MPEG-1, MPEG-2, MPEG-4, H.261, H.263, H.264 atau MPEG-4 AVC, serta VP8.

Beberapa karakteristik yang dimiliki oleh video digital adalah *frame dimensions*, *pixel depth*, dan *framerate*. *Frame dimensions* merupakan resolusi dari *frame* video yang diukur dengan satuan piksel. *Pixel depth* merupakan satuan yang digunakan untuk mengukur jumlah warna yang ditampilkan dalam tiap *pixel* gambar. *Framerate* adalah kecepatan *frame* yang ditampilkan dalam satu detik, diukur dengan satuan *fps* (*frame per second*).

MPEG merupakan salah satu format kompresi untuk video digital. Kompresi video digital berguna untuk menghemat ukuran dari suatu video khususnya untuk video yang ditransmisikan dalam jaringan. MPEG sendiri merupakan format kompresi yang paling umum digunakan. Algoritma yang digunakan untuk kompresi pada format ini adalah algoritma JPEG.

Terdapat empat sekuen gambar pada arsitektur MPEG, yaitu *I-Frame*, *P-Frame*, *B-Frame*, serta *D-Frame*. Keempat *frame* ini dibedakan dengan cara kompresinya. *I-Frame* dikompresi secara independen dari gambar lain dengan metode *intraframe compression*. *P-Frame* dikompresi dengan metode *motion compensation* dari *I-Frame* atau *P-Frame* sebelumnya. Sementara, *B-Frame* dikompresi dengan menggunakan *bi-directional prediction* berdasarkan interpolasi dari *I-Frame* atau *P-Frame* sebelumnya [4].

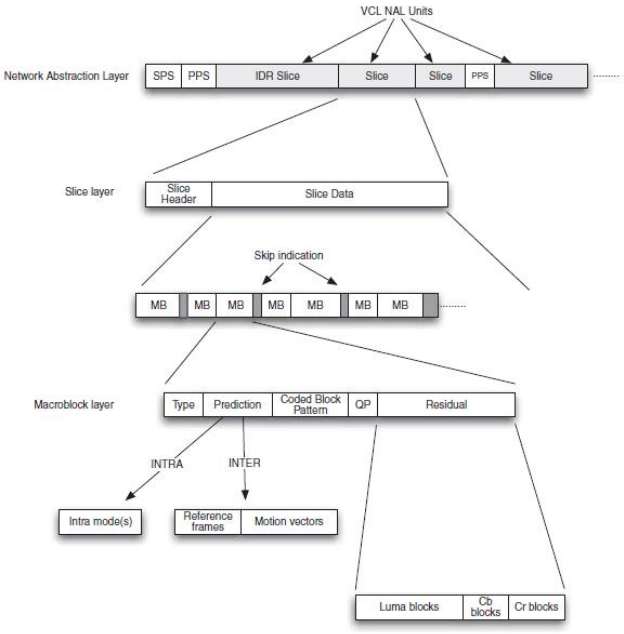


Gambar 1: Group of Pictures (GOP) pada Format MPEG [5]

Kumpulan *frame* pada format MPEG disebut dengan *Group of Pictures* (GOP) yang ditunjukkan pada Gambar 1. Dalam satu GOP biasanya terdapat 8-24 *frame*. GOP menunjukkan pergerakan satu *frame* dengan *frame* lain, misalnya *P-Frame* hanya menunjuk ke *frame* sebelumnya serta *B-Frame* hanya menunjuk ke *frame* sebelum dan sesudahnya.

Salah satu jenis atau versi dari MPEG adalah H.264. H.264 disusun berdasarkan kumpulan NAL Unit sebagai pembungkus layer data di bawahnya. NAL Unit terbagi menjadi dua tipe kelompok berdasarkan *video coding*, yaitu VCL (Video Coding Layer) dan Non-VCL. VCL NAL Unit merupakan NAL unit yang berisi data video, sedangkan Non-VCL NAL Unit berisi informasi-informasi lain tentang video.

Struktur secara keseluruhan ditunjukkan pada Gambar 2. Dari NAL Unit membungkus layer *slice*. Pada layer ini, *slice* terbagi menjadi dua bagian yaitu *slice header* dan *slice data*. Kemudian, di dalam *slice data* terbagi lagi menjadi kumpulan dari *macroblock*. Data *frame* video sebenarnya terdapat pada layer *macroblock* tersebut.



Gambar 2: Struktur Byte Stream H.264 [6]

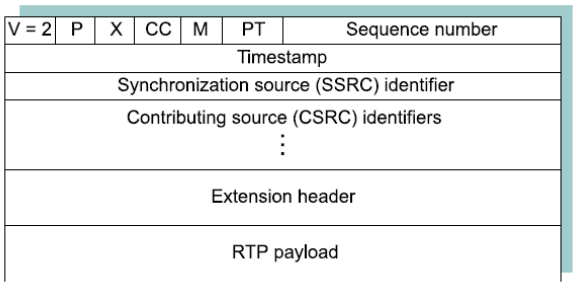
B. *Real-Time Transport Protocol (RTP)*

Real-Time Transport Protocol (RTP) merupakan standar protokol dalam pengiriman paket yang bersifat *real-time*, yaitu audio dan video.RTP berada di *user space* dan berjalan di atas protokol UDP. RTP dibuat untuk melengkapi fitur yang ada pada UDP tetapi berjalan secara *real time*. Aplikasi yang *real time* memerlukan spesifikasi kebutuhan protokol yang khusus karena waktu untuk pengiriman data sangat signifikan berpengaruh pada aplikasi. Protokol ini digunakan dalam berbagai macam aplikasi, seperti radio internet, telefon internet, *video conferencing*, *music-on-demand*, *video-on-demand*[7].

RTP merupakan *transport protocol*, namun RTP diimplementasikan pada *application layer*. Dari desain ini, menggambarkan bahwa sebuah *transport protocol* (RTP) berjalan di atas *transport protocol* lain yaitu UDP. Sumber audio atau video berasal dari aplikasi multimedia yang kemudian dibentuk menjadi paket-paket RTP. Setelah paket RTP terbentuk, paket UDP dibangkitkan dan ditempelkan ke paket IP. Paket-paket IP ini dikirimkan melalui jaringan (*subnet*) yang terhubung.

Format *header* pada protokol RTP dapat dilihat pada Gambar 3. Panjang minimal dari *header* adalah 12 *byte*. Penjelasan dari bagian-bagian *header* sebagai berikut,

1) *Version* (V): menunjukkan versi dari protokol yang digunakan.
2) *Padding* (P): menandakan apakah ada penambahan *byte* di akhir paket RTP.
3) *Extension* (X): menandakan apakah ada *extension header* antara standar *header* dengan RTP *payload*.
4) *CRSCCount* (CC): mengandung jumlah CSRC *identifier*.
5) *Marker* (M): digunakan pada *application-level* dan ditentukan oleh aplikasi yang membuat paket RTP.
6) *Payload Type* (PT): menunjukkan format dari data *payload*.
7) *Sequence Number*: merupakan nomor paket RTP yang dikirimkan.
8) *Timestamp:* digunakan oleh penerima untuk memulai kembali data yang diterima pada interval tertentu.
9) *SSRC identifier*: digunakan untuk mengidentifikasi sumber dari *stream*.
10) *CRSC identifier*: sebagai ID kontributor dari sumber-sumber data jika data dikirim berasal dari sumber yang banyak.
11) *Extension header*: merupakan field opsional.
12) *RTP payload*: berisi data yang akan dikirimkan dan secara spesifik ditentukan oleh aplikasi.



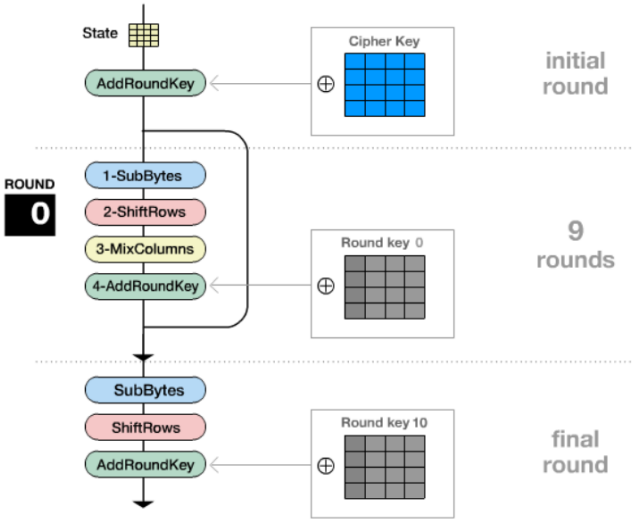
Gambar 3: Format RTP Header[8]

C. *Algoritma Advanced Encryption Standard (AES)*

Standard algoritma enkripsi pada awalnya (sejak 1973) adalah DES (*Data Encryption Standard*). Namun, keamanan DES mulai dipertanyakan oleh pemerintah US sekitar tahun 1993. Hal ini dikarenakan perkembangan teknologi komputer tidak dapat diimbangi oleh desain dari algoritma DES sendiri [9]. Oleh karena itu, *National Institute of Standards and Technology* (NITS) mengadakan sebuah lomba untuk menemukan standar algoritma enkripsi baru pada tahun 1997. Standar algoritma baru ini diberi nama AES (*Advanced Encryption Standard*).

AES merupakan algoritma simetri berbasiskan chipper blok. Terdapat dua varian dari AES yaitu AES-128 (ukuran blok 128-bit dengan panjang kunci 128-bit) dan AES-256 (ukuran blok 128-bit dengan panjang kunci 256-bit). AES juga mendukung kunci dengan 192-bit, namun AES dengan panjang kunci 192-bit tersebut jarang digunakan oleh orang.

Ukuran 16 byte ini menyesuaikan ukuran blok data maupun kunci yang berukuran 128-bit agar dapat dimasukkan ke dalam *array*. AES beroperasi pada matriks byte berukuran 4x4 (untuk blok data 128-bit) yang dinamakan *state*. *Array* dua dimensi *state* diinisialisasi dengan *plaintext* dan dimodifikasi setiap kali langkah yang ada pada komputasi. Secara garis besar, proses pada algoritma Rijndael dapat dilihat pada Gambar 4.

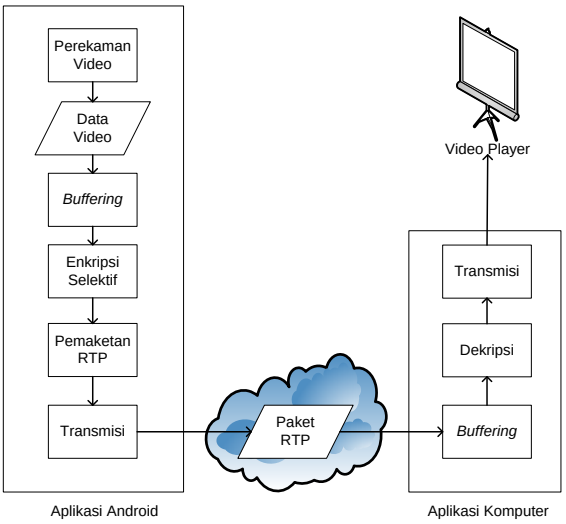


Gambar 4: Alur Proses Enkripsi AES[10]

III. IMPLEMENTASI

A. *Arsitektur Umum Sistem*

Aplikasi video *streaming* dengan enkripsi selektif video pada tugas akhir ini diberi nama *S-Streaming*, yang merupakan kepanjangan dari *Secured Streaming*. *S-Streaming* terdiri dari dua aplikasi yaitu aplikasi pada Android dan aplikasi Java pada komputer.



Gambar 5: Arsitektur Umum *S-Streaming*

Aplikasi *S-Streaming* pada Android merupakan aplikasi yang merekam video serta mengirimkannya ke aplikasi pada komputer. Kemudian dari aplikasi di komputer, video tersebut diteruskan ke *video player* untuk diputar. Arsitektur *S-Streaming* secara keseluruhan dapat dilihat pada Gambar 5. Seperti pada Gambar 5, proses enkripsi dilakukan pada aplikasi Android, sedangkan proses dekripsi dilakukan pada aplikasi di komputer. Pada Gambar 6, ditunjukkan tampilan antarmuka dari aplikasi yang dibangun.

Figure 6: The Interface of Android Application

Beberapa batasan implementasi pada aplikasi yang dibangun adalah sebagai berikut,

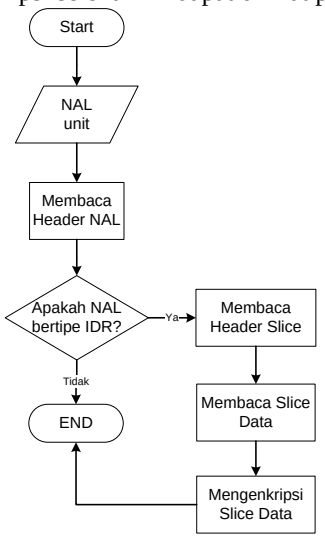
1) *Video streaming* dilakukan dengan menggunakan koneksi Wi-fi dalam jaringan lokal.
2) Target aplikasi pada *platform* Android 4.0.3 dengan minimal API 15.
3) Aplikasi dekripsi pada komputer berjalan bersamaan dengan aplikasi *video player* VLC.
4) Aplikasi dekripsi pada komputer tidak menangani masalah pengurutan paket RTP yang diterima.

B. *Proses Enkripsi Selektif*

Proses enkripsi dan dekripsi selektif memperhatikan struktur dari stream video H.264, yaitu dengan mencari dimana letak *I-frame*. Dalam video H.264, paket-paket data yang menyimpan stream video dinamakan dengan NAL unit (*Network Abstraction Layer unit*). Setiap NAL unit dapat menyimpan bagian dari *slice*. Di dalam *slice* sendiri terdapat bagian dari sebuah *frame* video.

Enkripsi secara selektif untuk video H.264 dilakukan dengan cara mencari NAL unit yang berisi *slice-slice* bertipe I. *Slice* tersebut berada pada NAL unit tertentu yang ditandai dengan tipe NAL IDR (*Instantaneous Decoding Refresh*). NAL bertipe IDR ini ditandai dengan nilai tipe NAL unit 5.

Setelah mendapatkan NAL unit bertipe IDR, langkah selanjutnya adalah membaca *header* dari *slice* yang terdapat pada NAL unit tersebut. Pembacaan *header slice* ini dilakukan secara VLC (*variable length coding*). Kemudian enkripsi dapat dilakukan pada data *slice* yang berada setelah header *slice*. Proses enkripsi selektif ini dapat dilihat pada Gambar 6.



Gambar 6: Proses Enkripsi Selektif

Data s*lice* yang sudah didapat akan dienkripsi dengan menggunakan AES. Mode blok cipher yang dapat digunakan adalah OFB dimana hasil cipherteks sama panjangnya dengan plainteks. Namun, penggunaan AES dengan mode OFB memerlukan IV (*initialization vector*) yang unik sesuai dengan rekomendasi dari NIST 800-38A[11].

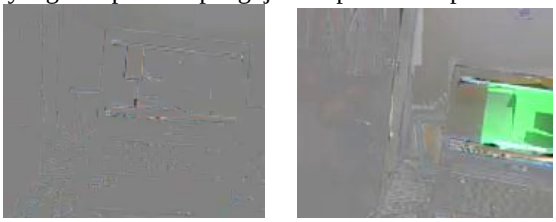
IV yang akan digunakan sepanjang 16 *byte*, yang terdiri dari 8 byte IV random dan 8 byte *counter*. Nilai *sequence* pada *header* paket RTP dapat dijadikan sebagai *counter* karena nilainya selalu bertambah. Sehingga, setiap paket RTP akan menggunakan nilai IV yang berbeda untuk enkripsinya.

IV. PENGUJIAN

Pengujian yang dilakukan pada aplikasi yang dibangun bertujuan sebagai berikut,

1) Mengetahui apakah seluruh kebutuhan fungsional dan non-fungsional telah terpenuhi.
2) Menguji kehandalan/kinerja dari aplikasi yang telah dibuat.
3) Menguji apakah data video yang dikirim telah berhasil terenkripsi secara selektif.

Untuk penggunaan enkripsi selektif, hasil pengujian yang didapatkan seperti yang ditunjukkan pada Gambar 7. Rata-rata *delay* yang didapat dari pengujian dapat dilihat pada Tabel 1.



Gambar 7: Hasil Enkripsi Selektif

Tabel 1: Rata-Rata Delay

Kasus Uji	**Rata-Rata Delay (ms)**
Tanpa Enkripsi	211.7
Dengan Enkripsi	219.4

Pengujian kinerja selain pengukuran *delay* adalah mengukur jumlah total paket yang dibentuk dan dikirimkan. Pengujian dilakukan dengan resolusi video 176x144 dan penggunaan enkripsi serta variabel fps nilainya berubah. Waktu *streaming* yang dilakukan adalah 30 detik. Hasil pengujian dapat dilihat pada Tabel 2.

Tabel 2: Perhitungan Total Pemaketan

Jenis	**10 fps**	**15 fps**	**20 fps**
Total Paket	677	805	863
Total *Slice*	322	476	629
Total *I-Slice*	30	30	30
Total *P-Slice*	292	446	599

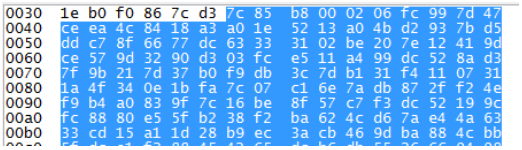
Pengujian kemanan dilakukan untuk mengecek apakah paket RTP yang dikirim sudah terenkripsi dengan benar. Pengujian dilakukan dengan membandingkan data NAL unit sebelum terenkripsi dan data NAL unit setelah terenkripsi. Data NAL unit yang terenkripsi ini diambil dari jaringan menggunakan *wireshark*.

Cuplikan data NAL sebelum terenkripsi ditunjukkan pada Gambar 8, sedangkan cuplikan data NAL yang telah terenkripsi diambil dari *wireshark* ditunjukkan pada Gambar 9.

```
7c85b8000206fc66221806df07400040af16a60ef097d341d406a2
95c000a970403048a50f499a11e2f4b884879b8997587262e7919e
3a110a1a1e649 …
```

Gambar 8: NAL tanpa enkripsi

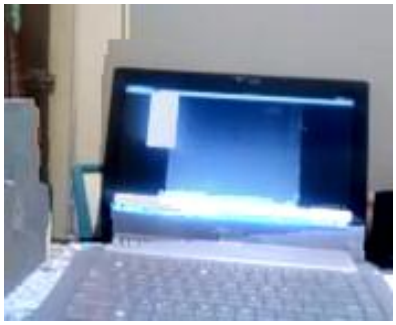
Beberapa byte awal masih sama, yaitu 7c85b8000206fc. Byte tersebut menandakan 2 byte FU-A *header* dan *indicator* serta sisanya merupakan *header* dari *slice*[12]. Setelah byte-byte tersebut, data mulai tidak sama yang menunjukkan bahwa data pada *slice* sudah terenkripsi.



Gambar 9: NAL dengan enkripsi

V. ANALISIS HASIL PENGUJIAN

Pada pengujian enkripsi/dekripsi video, terdapat beberapa gambar yang masih terlihat sebagian. Contohnya, seperti yang ditunjukkan pada Gambar 10. Pada gambar tersebut, masih dapat dilihat bahwa objek didalamnya adalah sebuah komputer jinjing. Dari gambar ini, kualitas enkripsi video masih terlihat sangat kurang aman karena objek di dalamnya masih dapat dilihat.



Gambar 10: Gambar belum terenkripsi secara sempurna

Berdasarkan pengujian keamanan enkripsi yang dilakukan dengan membandingkan NAL unit yang belum terenkripsi dengan NAL unit yang telah dikirim setelah dienkripsi, dapat dipastikan bahwa semua NAL unit dengan tipe IDR telah terenkripsi. Baik NAL unit dengan ukuran kecil (kurang dari MTU) ataupun berukuran besar sehingga perlu dipecah menjadi FU-A NAL unit, telah masuk enkripsi selektif jika tipe NAL tersebut adalah IDR.

Pengujian keamanan tersebut menunjukkan bahwa proses enkripsi telah dilakukan pada data yang benar, yaitu pada data *slice*. Enkripsi dilakukan dengan hati-hati agar dapat menghindari *header* dari *slice*. Hal ini bertujuan agar video yang terenkripsi masih dapat diputar oleh *video player*.

Walaupun semua NAL unit bertipe IDR telah terenkripsi, pada Gambar 10 menunjukkan bahwa enkripsi masih belum sempurna. Sesuai dengan hasil pengujian dalam pengukuran jumlah paket, *slice*, I-*slice*, ataupun P-*slice*, dapat diketahui bahwa jumlah I-*slice* sangat jauh lebih kecil dibandingkan jumlah P-*slice*. I-*slice* memiliki frekuensi kemunculan yang tetap yaitu sebesar 1 *slice*/detik. Sedangkan P-*slice* frekuensinya bergantung kepada jumlah fps.

Kemungkinan penyebab pada Gambar 10 masih terlihat gambar yang cukup jelas adalah gambar tersebut merupakan bagian dari P-*slice* yang tidak ikut terenkripsi. Gambar-gambar yang masih terlihat tidak terenkripsi tersebut sering

muncul ketika terdapat pergerakan objek atau pergerakan kamera sendiri.

Di dalam dokumen standar H.264, disebutkan bahwa setiap P-*slice* memungkinkan untuk berisi *macroblock* dengan tipe P atau I[13]. Jika dalam P-*slice* tersebut terdapat beberapa *macroblock* bertipe I, maka *macroblock* tersebut tidak membutuhkan parameter-parameter tertentu (misalnya *motion vector*) dalam proses *decoding*-nya.

Selain itu, dengan adanya limit bahwa kemunculan I-*slice* hanya dalam 1 kemunculan/detik, maka *encoder* H.264 akan menempatkan *macroblock-macroblock* bertipe I ke dalam P-*slice*. Dengan demikian, P-*slice* tersebut masih dapat di-*decode* menjadi *frame-frame* yang hampir kelihatan utuh tanpa membutuhkan *frame* lain dalam proses tersebut.

Proses pengukuran *delay* pada tahap pengujian dilakukan dengan menghitung selisih waktu antara waktu program dekripsi akan mengirimkan paket RTP ke *video player* dengan waktu program *streaming* mengirimkan paket RTP ke program dekripsi. Perhitungan *delay* dihitung dengan memperhatikan juga sinkronisasi waktu antara *smartphone* Android dengan waktu di komputer.

Jika digunakan enkripsi, maka *delay* dihitung berdasarkan selisih waktu program dekripsi selesai mendekripsi paket RTP dan akan mengirimkannya ke *video player* dengan waktu program *streaming* akan melakukan enkripsi terhadap data NAL unit. Dari hasil pengujian didapat rata-rata *delay* jika tanpa enkripsi sebesar 211.7 ms, sedangkan rata-rata *delay* sebesar 219.4 ms jika digunakan enkripsi.

Delay ini dapat ditoleransi nilainya sesuai standar yang ditentukan oleh ITU-T yaitu kurang dari 10 detik untuk *video streaming*[14]. Dengan maksimal *delay* sebesar 219.4 ms, aplikasi dapat berjalan secara interaktif.

Jika dilihat dari hasil tersebut, penggunaan proses enkripsi atau tanpa proses enkripsi tidak terlalu signifikan *delay* yang dihasilkan. *Delay* yang telah diukur merupakan gabungan dari waktu pembuatan paket RTP, enkripsi, pengiriman melalui jaringan, serta waktu dekripsi. Karena selama pengujian digunakan jaringan Wi-Fi lokal, maka waktu pengiriman dapat diasumsikan sangat kecil.

Namun, waktu lainnya yang perlu dipertimbangkan untuk perhitungan *delay* adalah waktu data video ditampung di dalam *buffer*. Waktu ini dapat ditambah dengan waktu perjalanan data video dari *encoder* kemudian melalui *localserversocket*, dan setelah itu data video baru ditampung di *buffer*. Waktu *delay* lainnya dapat terjadi pada *buffer* yang ada pada VLC sendiri.

VI. KESIMPULAN

Berdasarkan analisis hasil pengujian, sistem perangkat lunak S-Streaming dapat berjalan baik. Perangkat lunak ini dapat melakukan *video streaming* dari video yang ditangkap dengan menggunakan kamera pada *smartphone* dengan *platform* Android dan mengirimkannya ke komputer. Pada komputer, data video/audio yang dikirimkan dapat diputar dengan baik dengan menggunakan *video player* VLC.

Sistem perangkat lunak S-Streaming dapat juga melakukan proses enkripsi/dekripsi secara selektif dari data video yang dikirimkan. Proses pemilihan data video dilakukan dengan memilih *slice-slice* dengan tipe I saja. Kemudian proses enkripsi dilakukan pada data di dalam *slice* tersebut.

Dengan mengenkripsi data di dalam *slice* bertipe I saja, waktu pembacaan struktur H.264 dapat dikurangi. Hasil yang didapat dari enkripsi selektif cukup bagus, walaupun masih terdapat gambar yang terlihat sebagian objek di dalamnya. Hal ini dikarenakan *slice* bertipe P yang tidak terenkripsi mengandung data *frame* video yang dapat di-*decode* tanpa bergantung dengan *slice* bertipe I.

Delay dari proses enkripsi atau dekripsi selektif masih dapat ditoleransi yaitu dengan nilai 219 ms sehingga *video streaming* yang dihasilkan cukup lancar untuk dilihat.

Untuk pengembangan sistem selanjutnya, enkripsi secara selektif dapat dilanjutkan sampai ke layer *macroblock* agar video lebih terjamin keamanannya. Enkripsi selektif ini membutuhkan *parser* H.264 yang efektif dan efisien karena keterbatasan sumberdaya pada Android sehingga waktu komputasi *parser* tersebut dapat ditoleransi.

REFERENSI

[1] Jeyaganesh. (2010), "Introduction to Android Operating System". [Online]. Available: http://devlup.com/mobile/what-is-android/348/.

[2] Daemen, J.,danRijmen, V. (1999), "Rijndael AES Proposal". *AES Algorithm Submission*.

[3] Pctechguide. (2012), "The History of Digital Video". [Online]. Available: http://www.pctechguide.com/uncategorized/the-history-of-digital-video.

[4] Basith, S.A. and Done, S.R. (1996), "Digital Video, MPEG and Associated Artifacts". [Online]. Available: http://www.doc.ic.ac.uk/~nd/surprise_96/journal/vol4/sab/report.html.

[5] Girod, Bernd. (2007), "Video Coding Standards". [Online]. Available: http://www.stanford.edu/class/ee398b/handouts/lectures/04-StandardsMPEG124.pdf.

[6] Richardson, Iain E. G. (2010), "The H.264 advanced video compression standard 2nd Ed". Chichester, West Sussex, United Kingdom: John Wiley & Sons, Ltd.

[7] Tanenbaum, Andrew S. (2003), "Computer Networks". New Jersey : Prentice Hall PTR.

[8] Peterson, Larry L. dan Davie, Bruce S. (2007), "Computer Networks A Systems Approach". San Fransisco, CA : Elsevier, Inc.

[9] Anderson, Benjamin. (2007), "Advanced Encryption Standard". [Online]. Available: http://home.eng.iastate.edu/~hawklan/aes.pdf.

[10] Munir, Rinaldi. (2011), "Advanced Encryption Standard (AES)". [Online]. Available: http://www.informatika.org/~rinaldi/Kriptografi/2010-2011/Advanced%20Encryption%20Standard%20(AES).ppt.

[11] Dworkin, Morris. (2001), "Recommendation for Block Cipher Block Operation, Methods and Techniques". *National Institute of Standards and Technology Special Publication 800-38A*. [Online]. Available: http://csrc.nist.gov/publications/nistpubs/800-38a/sp800-38a.pdf.

[12] Wenger, S., Hannuksela, M.M., Stockhammer, T., Westerlund, M., Singer, D. (2005), "RTP Payload Format For H.264 Video, RFC 3984". [Online]. Available: http://www.ietf.org/rfc/rfc3984.txt.

[13] ITU-T. (2005), "Advanced Video Coding for Generic Audiovisual Services : Recommendation ITU-T for H.264". [Online]. Available: http://www.itu.int/rec/T-REC-H.264-201201-I/en.

[14] ITU-T. (2001), "End-User Multimedia QoS Categories : Recommendation ITU-T for G.1010". [Online]. Available: http://www.itu.int/rec/T-REC-G.1010-200111-I.