Known-Plaintext Attack Terhadap Data Terenkripsi WhatsApp

Muhammad Ismu Hadi
Lembaga Sandi Negara
Jakarta, Indonesia
cryptonezia@yahoo.com

***Abstrak*—WhatsApp merupakan salah satu fenomena digital yang muncul dalam beberapa tahun belakangan. Bagaimana tidak, pada tahun 2016 jumlah penggunanya sudah mencapai angka 1 milyar, dengan rata-rata pesan yang lalu-lalang setiap harinya sebesar 42 milyar. Dukungan fitur komunikasi yang super lengkap (*chat, group chat, voice, image, video, teleconference,* dan *document*), dikemas di dalam sebuah aplikasi adalah faktor penyebab mengapa WhatsApp begitu populer di berbagai kalangan masyarakat dunia. Pada April 2016 WhatsApp meningkatkan keamanan pada aplikasinya dengan mengusung konsep *end-to-end encryption* (E2EE), yang berarti proses enkripsi terjadi pada *application layer*. Tulisan ini akan berupaya meninjau terhadap seberapa efektif penerapan E2EE dalam mengamankan data pengguna. Adapun pendekatan yang digunakan berupa *known-plaintext attack*, dengan inputan berupa teks terang dan teks sandi yang saling bersesuaian. Simulasi dilakukan dengan menggunakan perangkat komputasi yang terdiri dari 256 buah FPGA (*Field-Programmable Gate Array*).**

Kata kunci— WhatsApp; dokumen terenkripsi; known-plaintext attack

I. PENDAHULUAN

Hampir seluruh sistem komunikasi yang ada saat ini telah mengimplementasikan kriptografi. Jika dahulu dianggap sebagai sebuah kemewahan, lain halnya dengan sekarang dimana kriptografi melalui layanannya, berupa confidentiality, authentication, data integrity dan non-repudiation telah bertransformasi menjadi faktor penting di dunia telematika.

Tentu masih hangat di telinga para pengguna aplikasi WhatsApp, kabar tentang peningkatan sistem keamanan pada aplikasi tersebut menjadi *end-to-end encryption* (E2EE). Hal itu membuktikan bahwa pengelola WhatsApp menaruh perhatian yang cukup besar terkait isu privasi dan keamanan bagi para penggunanya. Selain itu, juga menyiratkan hal penting dimana pelaku usaha sudah mulai sadar bahwa penerapan layanan kriptografi adalah bagian dari strategi bisnis yang sangat menjanjikan atau bahkan menentukan pada saat sekarang dan masa yang akan datang. Sebab, pada masa itu kriptografi menjelma sebagai komponen penting dalam menciptakan komoditas unggulan. Meskipun yang baru tampak saat ini hanyalah komoditas di bidang layanan telekomunikasi. Namun, tidak menutup kemungkinan akan merambah ke berbagai sektor kehidupan.

Konsekuensi dari penerapan E2EE adalah proses enkripsidekripsi berlangsung pada application layer (OSI layer). Dengan demikan tidak ada satupun yang dapat mengakses atau melihat isi pesan yang ditransmisikan, kecuali hanya pihak pengirim dan penerimanya saja. Sehubungan dengan hal itu, tulisan ini akan mengulas tentang tinjauan tentang penerapan *known-plaintext attack* terhadap dokumen terenkripsi dari WhatsApp.

II. LANDASAN TEORI

A. End-to-End Encryption pada WhatsApp

Aplikasi WhatsApp yang ada saat ini memiliki kemiripan dengan Signal Protocol (populer dengan nama TextSecure) hasil rancangan Open Whisper System. Signal protocol sendiri merupakan perpaduan antara algoritma Double Ratchet, Prekeys, dan sebuah Triple Diffie–Hellman (3-DH) Handshake, yang menggunakan Curve25519, AES256 dan HMACSHA256 sebagai primitifnya. Adapun kemampuan yang ditawarkan dari Signal Protocol, berupa: *confidentiality*, *integrity*, *authentication*, *participant consistency*, *destination validation*, *forward secrecy*, *backward secrecy*, *causality preservation*, *message unlinkability*, *message repudiation*, *participation repudiation*, dan *asynchronicity*. Ruang lingkup penerapan E2EE pada WhatsApp tidak hanya mengakomodir aspek keamanan pada chat saja, melainkan juga termasuk datadata lainnya seperti gambar, video, dan dokumen.

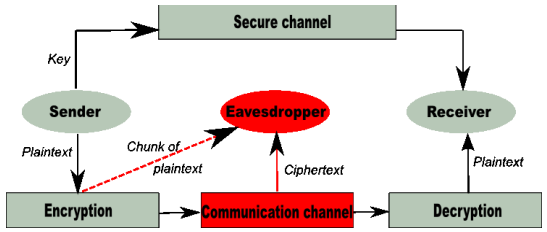
B. Known-Plaintext Attack

Tidak jarang proses kriptanalis hanya bertumpu pada petunjuk sederhana berupa potongan plaintext. Potongan tersebut bisa jadi sebuah perwujudan dari kata duga, yang justru akan menjadi acuan dalam upaya pencarian terhadap kunci atau potongan-potongan plaintext lainnya. Pada tataran praktis, kata duga dapat digali dari hal-hal sederhana, seperti:

- Kata-kata yang sering atau umum digunakan;
- Frasa stereotip, seperti kata pembuka atau penutup pada surat;
- *File signature* yang umumnya dimiliki oleh setiap file sebagai identitas;
- Konteks informasi, sebagai contoh dokumen berbentuk *spreadsheet* umumnya digunakan untuk membuat informasi tentang laporan laba-rugi atau neraca, sementara angka-angka yang tercantum di dalamnya adalah data yang telah diberi konteks sehingga bermakna dan bermanfaat;

- Kecerobohan operator, sebagai contoh mengirimkan pesan yang sama secara berulang-ulang, dan lain-lain.

Terdapat 3 (tiga) hal yang harus dipenuhi untuk melakukan known-plaintext attack, yaitu informasi tentang algoritma enkripsi, ciphertext yang akan dipecahkan, sepasang atau lebih ciphertext-plaintext yang bersesuaian.



Gambar 1. Skema *known-plaintext attack*

III. METODOLOGI

Secara umum, sistematika pada penelitan ini terdiri dari 4 tahap, yaitu: 1) Identifikasi data sampel; 2) Perekaman data sampel; 3) Investigasi Kriptogram dan Identifikasi algoritma kriptografi; dan 4) Penerapan known-plaintext attack.

A. *Identifikasi Data Sampel*

WhatsApp memiliki kemampuan mengakomodir proses kirim-terima data dalam bentuk yang bervariasi. Akan tetapi, tidak semua jenis data bisa digunakan sebagai sampel dalam penelitian ini. Oleh karenanya, perlu dilakukan proses identifikasi terhadap jenis data terlebih dahulu, agar diperoleh sampel yang tepat. Pendekatan yang digunakan, yaitu dengan cara menganalisis baik dari sisi penerapan port data maupun prosedur atau mekanisme kompresi. Berikut adalah skenario yang akan digunakan dalam melakukan analisis terhadap port data yang digunakan WhatsApp, yaitu:

- Laptop, berfungsi sebagai *gateway* yang didalamnya berjalan aplikasi Wireshark;
- Koneksi Laptop ke internet menggunakan *interface* LAN; dan
- Koneksi Laptop dengan Tab menggunakan *interface* WLAN.

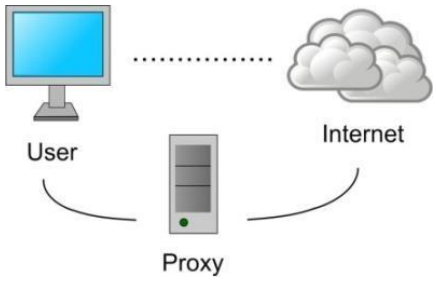
Gambar 2. Skema analisis *port*

Selanjutnya adalah skenario yang akan digunakan dalam melakukan analisis terhadap pemberlakukan prosedur kompresi, dengan memperbandingkan file asli yang terdapat di sisi pengirim dengan file yang sampai di tujuan

B. *Pengumpulan Data Sampel Terenkripsi*

Tahap berikutnya adalah melakukan pengumpulan data sampel. Data sampel yang dimaksud merupakan data terenkripsi yang dikirim-terimakan menggunakan WhatsApp. Adapun skenario yang akan dilakukan, sebagai berikut:

- Proxy, berfungsi untuk menjalankan aplikasi Web Proxy;
- User, berfungsi untuk menjalankan aplikasi WhatsAppWeb;
- Sertifikat digital yang dibangkitkan secara independen. Kemudian dipasang ke Proxy (*private*) dan User (*public*). Hal ini berfungsi untuk melancarkan modus serangan *Man-in-the-Middle* (MITM), guna menegasikan fitur keamanan WhatsApp yang terjadi pada *presentation layer* (OSI); dan
- Koneksi internet.



Gambar 3. Skema pengumpulan sampel

C. *Investigasi Kriptogram dan Identifikasi Algoritma Kriptografi*

Proses investigasi pada hakikatnya bertujuan untuk memastikan posisi kriptogram di dalam struktur data sampel. Sedangkan proses kelanjutannya adalah identifikasi yang bertujuan untuk menemukenali jenis algoritma kriptografi yang digunakan saat mengenkripsi kriptogram tersebut.

D. *Analisis Known-Plaintext Attack*

Tahap ini dilakukan analisis terhadap penerapan *known-plaintext attack*. Masukan yang akan digunakan berupa pasangan blok *plaintext* dan *ciphertext* yang bersesuaian Kemudian mengukur kadar kompleksitasnya dengan menggunakan perangkat komputasi. Adapun spesifikasi perangkat yang digunakan adalah sebuah server yang di lengkapi dengan komponen pemroses data berupa *Field Programmable Gate Array* (FPGA) sebanyak 256 buah yang akan bekerja secara paralel.

IV. PEMBAHASAN

A. *Identifikasi Data Sampel*

Pada tahap awal identifikasi, data yang digunakan merepresentasikan setiap bentuk data sesuai dengan format yang didukung WhatsApp. Berikut adalah daftar dari data-data tersebut.

TABEL I. DAFTAR DATA SAMPEL

Nama	Ekstensi (*.)	Kategori	Ukuran (Kb)
fileuji01	jpg	Gambar	1,050
fileuji02	png	Gambar	1,050
fileuji03	tiff	Gambar	1,050
fileuji04	bmp	Gambar	1,050
fileuji05	aac	audio	3,372
fileuji06	m4a	audio	2,367
fileuji07	mov	video	13,222
fileuji08	mp4	video	13,220
fileuji09	txt	dokumen	1,050
fileuji10	doc	dokumen	1,050
fileuji11	ppt	dokumen	1,050
fileuji12	xls	dokumen	1,050
fileuji13	pdf	dokumen	1,050

Data-data tersebut kemudian dijadikan sebagai objek dalam proses kirim-terima antara 2 pengguna WhatsApp. Selama proses kirim-terima berlangsung, terdapat 2 hal yang dilakukan, yaitu analisis port dan prosedur kompresi. Berikut adalah hasil yang didapat dari kegiatan analisis terhadap penggunaan port oleh WhatsApp.

TABEL II. PORT YANG DIGUNAKAN WHATSAPP

Kategori	Port
Chat atau teks	5222/tcp
File dengan format gambar	443/tcp
File dengan format audio	443/tcp
File dengan format video	443/tcp
File dengan format dokumen	443/tcp

Berdasarkan data di atas, diketahui bahwa WhatsApp hanya menggunakan 2 jenis port, yaitu 5222/tcp dan 443/tcp. Port 5222/tcp digunakan untuk komunikasi chat, yang secara umum juga digunakan oleh protokol xmpp. Sedangkan, port 443/tcp digunakan untuk komunikasi data dengan format selain teks, dan umumnya digunakan oleh protokol HTTP *over* SSL / TLS (HTTPS).

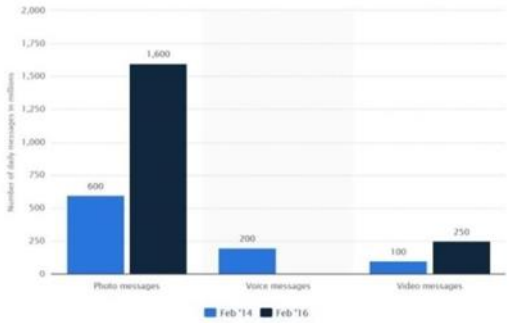
Langkah selanjutnya adalah mengidentifikasi penerapan prosedur kompresi pada WhatsApp. Caranya dengan mengirimkan file-file berukuran besar dengan ekstensi sesuai tabel 1. Selanjutnya membandingkan ukuran file baik sebelum dan sesudah dikirim ke tujuan. File yang ukurannya menyusut, maka diindikasikan terkena prosedur kompresi.

TABEL III. IDENTIFIKASI JENIS FILE YANG DIKOMPRESI PADA WHATSAPP

Nama	Ukuran Awal (Kb)	Ukuran Akhir (Kb)	Kompresi (%)
fileuji01	1,050	233	77.81
fileuji02	1,050	233	77.81
fileuji03	1,050	233	77.81
fileuji04	1,050	233	77.81
fileuji05	3,372	3,372	0.00
fileuji06	2,367	2,367	0.00
fileuji07	13,222	13,222	0.00
fileuji08	13,220	13,220	0.00
fileuji09	1,050	1,050	0.00
fileuji10	1,050	1,050	0.00
fileuji11	1,050	1,050	0.00
fileuji12	1,050	1,050	0.00
fileuji13	1,050	1,050	0.00

Berdasarkan tabel di atas, diketahui bahwa prosedur kompresi pada WhatsApp hanya berlaku pada file-file dengan kategori gambar dengan ketentuan ukurannya ≥ 1 MB. Berikut analisis yang mendasari penerapan kompresi bersifat parsial, antara lain:

- Murni bisnis. Setiap harinya WhatsApp mengelola *traffic* data dalam jumlah yang sangat besar. Berdasarkan data statistik yang dipublikasikan oleh Statista[1] pada salah satu halamannya, didapati bahwa volume *traffic* terbesar disumbangkan oleh data yang berbentuk gambar, video, dan audio. Konsekuensi dari hal tersebut adalah kebutuhan akan media penyimpanan (*storage*) dalam jumlah yang banyak, khususnya untuk menangani persoalan file gambar. Dalam konteks bisnis, pengelolaan sumber daya akan berbanding lurus dengan *cost* atau biaya. Oleh sebab itu, segenap daya dan upaya tentu akan dilakukan oleh manajemen agar pemanfaatan *storage* dapat lebih efisien. Salah satu cara adalah dengan mereduksi ukuran file gambar melalui penerapan teknik kompresi.

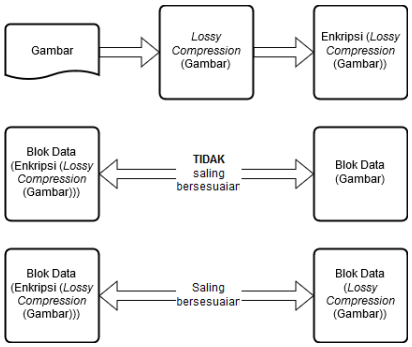


Gambar 4. Grafik perbandingan volume traffic pesan yang dikelola WhatsApp setiap harinya pada Tahun 2014 dan 2016

- Proses kompresi yang dilakukan tergolong *lossy*, sehingga data yang telah terkompresi tidak dapat dikembalikan ke kondisi awal. Itulah mengapa WhatsApp melakukan diskriminasi dalam hal kompresi. Analogi sederhananya adalah karena konsep *lossy* pada file gambar tidak terlalu menjadi persoalan selama masih dapat diterima secara visual oleh mata. Sama halnya dengan audio oleh telinga dan video oleh mata dan telinga. Lain halnya dengan file dokumen, yang justru bermasalah jika kehilangan bit meski dalam jumlah sedikit saja. Sebab, dapat menyebabkan kesalahan (error) pada file ketika dibaca.
- Performa. Tenik kompresi menyebabkan kemampuan transfer data menjadi lebih baik atau efisien.
- Prosedur kompresi pada gambar juga bisa dimaknai sebagai bentuk preventif terhadap ancaman yang memanfaatkan WhatsApp untuk kirim-terima *stegofile* dengan file carrier berupa gambar. Seperti halnya modus teroris pada aksi 911 di Amerika Serikat.

Sebagaimana diketahui bahwa *lossy compression* akan menyebabkan hilangnya informasi bit pada file secara permanen. Apabila dihubungkan dengan *known-plaintext attack*, yang notabenenya membutuhkan perpadanan antara dua

buah blok, yaitu *plaintext* dan *ciphertext*. Maka, file yang terkena *lossy compression* tidak dapat digunakan sebagai sampel. Karena sudah pasti tidak bisa menghasilkan padanan blok. Secara sederhana kondisi tersebut dapat digambarkan dalam bentuk skema di bawah ini.



Gambar 5. Skema perpadanan blok data

B. Pengumpulan Data Sampel Terenkripsi

1) Port 443/tcp

WhatsApp menggunakan *port* 443/tcp untuk kepentingan *media sharing*, meliputi gambar, video, dan dokumen. Berdasarkan hasil analisis yang dilakukan terhadap data sampel, diketahui bahwa pesan yang akan ditransmisikan tersusun atas 2 (dua) buah komponen, yaitu: *Entity-Header Fields* dan *Entity-Body*.

a. Entity-Header Fields

Entity-header fields dapat juga disebut sebagai metadata dari *entity-body*. Dapat dimaknai sebagai bagian dari prosedur *request* & *response* di dalam *Hypertext Transfer Protocol* (HTTP). Fungsinya adalah untuk mendefinisikan parameterparameter yang digunakan saat komunikasi berlangsung. Adapun parameter yang dimaksud, meliputi: *http request method*, *host*, *user-agent*, *accept*, *content-language*, *contentencoding*, *referrer*, *content-length*, *content-type*, *origin*, DNT, dan *connection*. Berikut adalah salah satu cuplikan informasi *entity-header fields* yang berasal dari file sampel dengan nama "fileuji09".

POST */u/f/Zhto8ebKl4nmjdm52cPXJFjNL2MABUsA4Ffiiw/AshNU4X oaRJrcReCIxoPJEjkYQf105dSZpNSGDxCvFaG?f=j HTTP/1.1*
Host: mmg.whatsapp.net
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64;rv:52.0) Gecko/20100101 Firefox/52.0
Accept*: */**

Accept-Language: en-US,en;q=0.5

Referer: *https://web.whatsapp.com/*

Content-Length: *914*

Content-Type: *multipart/form-data; boundary=----------23769175275774*
Origin: https://web.whatsapp.com
DNT: 1
Connection: close

Dari cuplikan informasi tentang *entity-header fields* di atas, terdapat beberapa hal yang dapat dijelaskan, yaitu:

- POST adalah jenis *http request method*. Penggunaannya bertujuan untuk mengirimkan file sesuai dengan URI (*Uniform Resource Indicator*) yang telah ditetapkan. Dalam kasus ini berupa *mmg.whatsapp.net* dengan subordinat /f/AshNU4XoaRJrcReCIxoP *JEjkYQf105dSZpNSGDxCvFaG.enc HTTP/1.1*. Akan tetapi, pada sisi penerima metode yang digunakan berupa GET. Fungsinya untuk mengambil pesan dari server yang diperuntukkan baginya pada URI tertentu. Metode inilah yang nantinya digunakan untuk mengakses dokumen terenkripsi dengan ciri khas berupa ekstensi *.enc.
- *Accept* merupakan *request fields* yang memuat informasi tentang *content-types*. Dalam hal ini jenis *content-types* yang diizinkan adalah */*;
- *Referer* yaitu alamat dari halaman web asal yang bertautan dengan halaman yang sedang diakses, dalam hal ini adalah *https://web.whatsapp.com/*;
- *Content-Length* merupakan *request fields* yang memuat informasi tentang panjang daripada *body* file yang dikirim. Format yang digunakan adalah *octet* (8bit). Berdasarkan contoh di atas diketahui *contentlength* sebesar 914 *byte*;
- *Content-types* adalah informasi yang menjelaskan tentang jenis *Multipurpose Internet Mail Extension* (MIME) dari *body* pada file yang dikirim. Berdasarkan contoh di atas terdapat keterangan berupa *multipart/form-data; boundary=----------23769175275774*. Adapun *multipart/formdata* adalah jenis *content-type* yang diperuntukkan bagi file berbentuk gambar, video, dokumen, dan sebagainya. Sedangkan boundary berfungsi sebagai pembatas tiap komponen data di dalam struktur *body*. Berdasarkan contoh di atas diketahui nilainya adalah "----------6766941413187"; Secara teknis, susunan karakter penyusun nilai *boundary* bersifat bebas, asalkan jumlahnya tidak boleh lebih dari 70. Selain itu, penerapannya adalah konsekuensi dari penggunaan *content-type* berupa *multipart/**.

b. Entity-Body

Entity-*body* adalah bagian dari http *request* yang memuat informasi tentang data atau file yang akan ditransmisikan bersamaan dengan *header fields*. Berdasarkan analisis terhadap file sampel, diketahui bahwa penyusun *entitiy-body* terdiri atas 3(tiga) bagian. Untuk lebih jelas, berikut cuplikan *entity-body* dari file sampel dengan nama file09.

----------23769175275774

Content-Disposition: form-data; name="hash"

DgOx+d8EoZ6v/m0MHVEPWO8JKI4SWJ9S/9pj3RIa
F00= -----------------------------23769175275774
Content-Disposition: form-data; name="refs"

Jl8jJu8Ai6Xzv2sTFyqp24d2U2U=

lYV-2zwdtIv1Y1jymP5kVMfsX_Q=

-----------------------------23769175275774

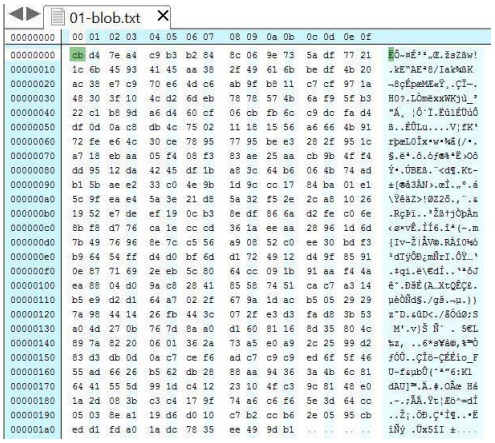
Content-Disposition: form-data; name="file";

filename="blob"

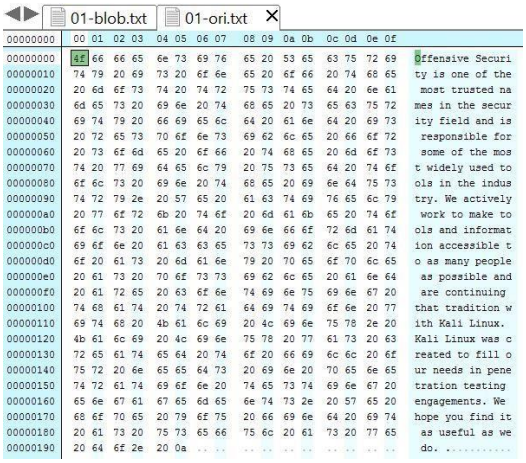
***Content-Type:** application/octet-stream*

Sebagaimana dijelaskan dalam dokumen RFC nomor 2046, dimana setiap *content-type* mengandung 2 buah informasi penting yang akan menggambarkan struktur data dari *body*. Informasi tersebut berupa *media type* dan *media subtype*. Secara sederhana contoh informasi dari *media type*, adalah *text*, *image*, *audio*, *video*, dan *application*. Selanjutnya, informasi dari tiap-tiap *media type* tersebut akan diinci oleh informasi *subtype*. Sebagai contoh *plain* pada *text*, jpeg pada *image*, *basic* pada *audio*, mpeg pada *video*, dan seterusnya. Adapun tata cara penulisannya sebagai berikut *media type*/*subtype*, contoh: *text/plain*, *image/jpeg*, dan seterusnya.

Kembali ke *entitiy-body* dari file sampel, dimana informasi yang tercantum dalam *content-type* berupa *application/octetstream*. Jika diurai 1 per 1, maka dapat diketahui bahwa *media type* yang digunakan adalah *application* dengan *subtype* berupa *octet-stream*. Penggunaan *application* pada hakikatnya menunjukkan bahwa data yang terdapat di dalam *body* bukan tergolong sebagai *text*, *image*, *audio*, ataupun *video*. Mekanisme ini hanya diperuntukkan bagi data-data yang identik dengan aplikasi khusus dengan format data tersendiri. Termasuk salah satunya data hasil enkripsi. Agar lebih jelas, berikut adalah cuplikan informasi yang terdapat di dalam *body* dari file bernama *AshNU4XoaRJrcReCIxoPJEjkYQf105dSZpNSGDxCvFaG.enc* yang bersesuaian dengan file sampel dengan nama fileuji09.txt.



Gambar 6. Informasi *body* dari AshNU4XoaRJrcReCIxoPJEjkYQf105dSZpNSGDxCvFaG.enc.



Gambar 7. Informasi *body* dari fileuji09.txt

2) Port 5222/tcp

Sebagaimana telah dijelaskan pada tahap identifikasi data sampel terenkripsi sebelumnya, bahwa WhatsApp menggunakan port 5222/tcp untuk memfasilitasi komunikasi berbasis teks atau *chat*. Nomor tersebut merupakan *port* yang memang secara resmi diperuntukkan bagi protokol XMPP. Secara *default*, XMPP yang berjalan pada *port* tersebut tidak memiliki pengamanan kriptografi. Lain halnya jika *port* yang digunakan adalah 5223/tcp, yang memang diperuntukkan bagi XMPP *client connection over* SSL. Secara sederhana, XMPP yang berjalan pada port 5222/tcp seharusnya dalam kondisi terbuka atau *plain*. Namun, berdasarkan analisis terhadap *chat traffic* WhatsApp yang berhasil diperoleh, didapati hasil yang berbeda dari yang dibayangkan sebelumnya. Kondisi tersebut mulai tampak setelah dilakukan rekonstruksi terhadap *chat traffic*, dimana informasi yang terbentuk sama sekali tidak terbaca. Berikut adalah cuplikan hasil rekonstruksi *chat traffic* WhatsApp tersebut.

Gambar 8. Hasil rekonstruksi terhadap paket data chat WhatsApp

C. Investigasi Kriptogram dan Identifikasi Algoritma Kriptografi

Berikut ini adalah prosedur yang diterapkan WhatsApp saat melakukan proses kirim-terima file berjenis gambar, audio,

video, dan dokumen, yaitu (WhatsApp Encryption Overview: Technical White Paper, 2016):

1) Pertama-tama pihak pengirim membangkitan rangkaian bit sebanyak 32-*byte* yang akan dipakai sebagai kunci temporer dalam proses enkripsi menggunakan AES256. Selain itu, juga membangkitkan data dengan panjang sama, yang diperuntukkan sebagai kunci temporer dalam pembuatan *Message Authentication Code* (MAC) menggunakan algoritma HMAC-SHA256.

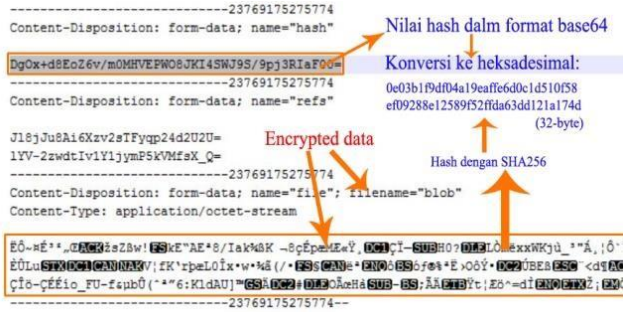
2) Selanjutnya pihak pengirim mengenkripsi *attachment file* tersebut dengan AES256 dan kunci temporer yang telah dibangkitkan sebelumnya, beserta sebuah inisial vektor (IV) dengan nilai acak. Adapun mode penyandian yang digunakan berupa *Cipher Block Chaining* (CBC). Hasil keluaran dari proses ini adalah data terenkripsi. Selanjutnya, pada data terenkripsi tersebut ditambahkan informasi MAC, yang didapat dari proses *hash* terhadap data terenkripsi itu sendiri dengan algoritma HMAC-SHA256.

3) Setelah itu, pengirim mengunggah data tersandi tersebut ke dalam sebuah *blob store*.

4) Informasi tentang data tersandi tersebut, selanjutnya ditransmisikan pihak pengirim kepada pihak penerima. Saat pengiriman dilakukan, terdapat parameter-parameter yang menyertai, meliputi: *encryption key*, HMAC *key*, sebuah nilai hash yang berasal dari pemrosesan *encrypted blob* menggunakan algoritma fungsi hash SHA256, dan sebuah *pointer* dari *blob* di dalam *blob store*.

5) Langkah terakhir adalah pihak penerima mendekripsi pesan, lalu mengunduh data terenkripsi dari dalam *bob store*, memverifikasi MAC, dan kemudian mendekripsi data.

Berdasarkan di atas, dan dikaitkan dengan hasil analisis terhadap data sampel, maka dapat diketahui bahwa algoritma kriptografi yang digunakan adalah AES256. Adapun posisi kriptogram, secara teknis terdapat pada blob dengan *contenttype : application / octet-stream*. Agar lebih jelas, berikut penjelasan dari posisi kriptogram di dalam *blob store*.



Gambar 9. Posisi data terenkripsi dan parameter yang menyertai

Pada gambar diketahui bahwa nilai hash dari data terenkripsi dalam format base64 adalah:

DgOx+d8EoZ6v/m0MHVEPWO8JKI4SWJ9S/9pj3RIaF00=

jika dikonversi ke dalam bentuk heksadesimal menjadi

0e03b1f9df04a19eaffe6d0c1d510f58ef09288e12589f52ffda63dd121a174d

Apabila data *blob* di atas dimasukkan ke dalam fungsi hash dengan algoritma SHA256, maka akan menghasilkan dua nilai hash yang identik.

D. Penerapan Known-Plaintext Attack

Pada tahap ini, terdapat 2 jenis *input* yang digunakan, yaitu:

- *Ciphertext* dan *plaintext* yang berpadanan, masingmasing 128-bit dalam bentuk heksadesimal. Berikut adalah padanan *cipher* dan *plaintext* yang digunakan, yaitu:

  ciphertext = cb d4 7e a4 c9 b3 b2 84 8c 06 9e 73 5a df 77 21

  plaintext = 66 66 65 6e 73 69 76 65 20 53 65 63 75 72 69

- *Start-key*, 256-bit dalam bentuk heksadesimal, yang akan berubah secara *incremental*.

Selanjutnya, teknik serangan yang diterapkan dengan menggunakan seluruh kemungkinan percobaan (*pure bruteforce*). Sehingga, kemungkinan terburuk dari seluruh percobaan yang akan dilakukan adalah 2^{256}.

$$O(2^N) \quad (1)$$

Perangkat komputasi yang digunakan berupa sebuah server yang didukung oleh 256 buah FPGA dan bekerja secara paralel. Kemampuan perangkat dalam melakukan percobaan serangan *bruteforce* terhadap AES256 adalah $153x10^9 \sim 2^{37}$ percobaan/detik. Berdasarkan hal tersebut, maka dapat diperkirakan lama waktu yang diperlukan untuk melakukan *known-plaintext attack* terhadap dokumen terenkripsi WhatsApp, sebagai berikut:

$$t = 2^{256} \text{ percobaan} \div 2^{37} \text{ percobaan/detik}$$
$$= 2^{219} \text{ detik} \sim 2{,}71 \times 10^{58} \text{ tahun} \quad (2)$$

Kondisi di atas secara terang menjelaskan bahwa proses komputasi memerlukan waktu yang sangat lama. Meski demikian, terdapat cara untuk mempersingkatnya, yaitu dengan menambah jumlah sumber daya komputasi. Akan tetapi, akan mucul persoalan baru berupa kebutuhan peningkatan sumber tenaga listrik yang sangat besar.

V. KESIMPULAN

Semenjak April 2016 yang lalu, WhatsApp telah meningkatkan mekanisme pengamanan datanya menjadi E2EE. Bukan hanya *chat*, melainkan seluruh layanan komunikasinya, seperti: gambar, audio, video, dokumen, dan panggilan suara. Hal ini tentu menjadikan keamanan data para pengguna WhatsApp menjadi lebih terjaga. Tulisan ini telah menunjukkan secara langsung tentang bukti penerapan kriptografi dalam pengamanan WhatsApp dengan menyajikan hasil-hasil analisis paket data terhadap keluaran dari WhatsApp. Jenis data sampel yang digunakan pada penelitian ini adalah file dokumen. Hal ini disebabkan, WhatsApp tidak memberlakukan prosedur kompresi pada file-file tersebut. Dari hasil investigasi diketahui bahwa hasil enkripsi data terdapat pada bagian *blob* di dalam struktur *body*. Kemudian, dari hasil identifikasi juga diketahui bahwa, algoritma kriptografi yang digunakan untuk enkripsi-

dekripsi data tersebut adalah AES256. Untuk melakukan *known-plaintext attack*, maka dilakukan ekstraksi dari data-data yang berpadanan sebanyak 128-bit, yang masing-masing berasal dari *ciphertext* dan *plaintext*. Dengan menggunakan pendekatan *bruteforce* dan didukung oleh perangkat paralel komputer berbasis FPGA (256 *board*), maka waktu yang diperlukan adalah 2,71 x 10^{58} tahun. Untuk dapat mempersingkatnya, diperlukan penambahan perangkat komputasi dalam jumlah yang masif. Meski demikian, perlu dipertimbangkan dari aspek pemenuhan sumber daya listrik.

DAFTAR PUSTAKA

[1] Gordon, Katriel Kohn, Cremers, Cas, dkk. Oktober 2016. *A Formal Security Analysis of the Signal Messaging Protocol*.

[2] Li, Calvin, Sanchez, Daniel, dan Hua, Sean. 2016. *WhatsApp Security Paper Analysis*.

[3] Freed, Ned, Kucherawy, Murray et al. 2017. *Media Types*. http://www.iana.org/assignments/media-types/media-types.xhtml (diakses 18 Maret 2017).

[4] Freed, Ned, Borenstein et al. 1996. *Multipurpose Internet Mail Extensions (MIME) Part One : Format of Internet Message Bodies*. https://tools.ietf.org/html/rfc2045 (diakses 18 Maret 2017).

[5] Freed, Ned, Borenstein et al. 1996. *Multipurpose Internet Mail Extensions (MIME) Part Two : Media Types*. https://tools.ietf.org/html/rfc2046 (diakses 18 Maret 2017).

[6] Fielding, R., Irvine, UC et al. 1999. *Hypertext Transfer Protocol -- HTTP / 1.1*. https://www.ietf.org/rfc/rfc2616.txt (diakses 18 Maret 2017). WhatsApp. 2016. WhatsApp Encryption Overview: Technical White Paper.