Analisis Algoritma Netralisasi Malware yang Tersembunyi pada Gambar

Kevin Kencana 18219050
Program Studi Sistem dan Teknologi Informasi
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail (gmail): 18219050@std.stei.itb.ac.id

Abstract*—Sejak internet ditemukan dan memfasilitasi komunikasi jarak jauh untuk manusia, perkembangan internet dan jumlah penggunanya diikuti dengan penyebaran *malware* oleh *cyber criminal* untuk tindak kejahatan. Dari awal penyebaran *malware* harus secara fisik lewat *floppy disk* hingga sekarang para *hacker* memiliki banyak celah dan metode untuk menyebar *malware* ke perangkat elektronik kita. Salah satu metode penyebaran yang saat ini sulit untuk dideteksi oleh perangkat lunak antivirus kita adalah menyembunyikan *malware* dalam sebuah file gambar, atau disebut dengan *stegosploit*. Pengguna perangkat dengan hanya membuka file gambar dapat menginfeksi perangkatnya dengan *malware* yang tersembunyi di dalam gambar tersebut. Para ahli sekuriti telah mengajukan algoritma-algoritma untuk mendeteksi dan menangani *stegosploit* pada gambar. Algoritma yang paling efektif dalam menangani *stegosploit* adalah algoritma yang mampu untuk menetralisir kode *malware* yang tersembunyi dalam file gambar sehingga kode tidak dapat berjalan tanpa melihat gambar yang dimodifikasi. Terdapat 3 metode *stegosploit* berdasarkan letak kode *malware* di dalam struktur file gambar dan algoritma terbaik yang dapat menetralisir kode *malware* untuk ketiga metode tersebut adalah algoritma ImageDetox yang diusulkan oleh Jung et al.[2]. Dengan hasil pengujian yang memuaskan, algoritma ini sangat baik untuk mengurangi resiko perangkat kita terinfeksi *malware* dari sebuah gambar.

Keywords: Stegosploit, Gambar, Malware, Algoritma

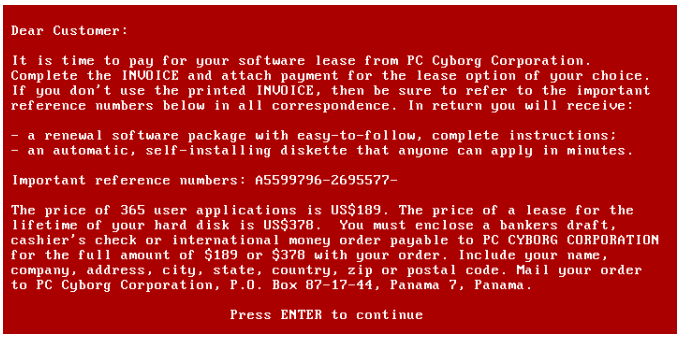
I. Pendahuluan

Sejak internet ditemukan oleh manusia, perkembangan teknologi bertumbuh pesat dalam cara kita berkomunikasi, dari komunikasi tatap muka menjadi komunikasi jarak jauh, surat menyurat kertas menjadi *digital*, dan sebagainya. Dengan internet, kita tidak hanya dapat mengirim pesan berupa teks, namun juga gambar, dokumen, video, dan media lain yang berisikan pesan yang ingin kita sampaikan kepada penerima. Meskipun internet telah membantu manusia, terdapat orang-orang yang menyalahgunakan internet untuk melakukan kejahatan, atau sering disebut dengan istilah *cyber crime*. Salah satu bentuk *cyber crime* yang marak adalah infeksi *malicious software (malware)* di dalam perangkat elektronik. Tujuan *cyber crime* antara lain adalah untuk penyadapan, pencurian informasi rahasia kita, atau merusak file-file dalam perangkat dan meminta uang tebusan untuk mengembalikannya. Karena penjahat tidak memiliki akses fisik pada perangkat elektronik kita, maka mereka perlu mencari cara untuk memasukkan *malware* dari dalam perangkat kita. Salah satu cara penyebaran *malware* yang saat ini terjadi adalah menyembunyikan *malware* di dalam gambar dan mengirimnya ke kita, atau disebut dengan *stegosploit*. Dengan penemuan metode baru ini, maka ahli sekuriti sudah mengajukan solusi-solusi untuk dapat mendeteksi gambar-gambar yang berisi *malware* dan kemudian menetralisir gambar agar *malware* tidak dapat menginfeksi perangkat kita. Pada makalah ini akan dianalisis algoritma yang telah diajukan untuk pendeteksian dan netralisasi gambar hasil *stegosploit*.

II. Malware

Malicious software, atau disingkat *malware*, adalah serangkaian kode program yang diciptakan dengan tujuan untuk mengganggu, merusak, atau mendapatkan informasi rahasia yang terdapat dalam perangkat elektronik. Konsep *malware* telah ada sejak tahun 1971, bahkan sebelum internet dapat digunakan oleh publik. Pada awal-awal perkembangannya, *malware* tidak diciptakan dengan tujuan kejahatan, namun hanya untuk lelucon. Antara lain *malware* pada zaman awal ini adalah "*Elk Cloner"*, sebuah *malware* yang diciptakan oleh seseorang yang berumur 15 tahun dengan tujuan untuk menjahili teman-temannya. *Malware* ini hanya akan menampilkan sebuah puisi apabila perangkat temannya dinyalakan pada ke-50 kalinya.

Malware pertama di dunia yang digunakan untuk kejahatan diciptakan pada tahun 1989, yaitu *AIDS Trojan*. *Malware* ini dikirim melalui surat fisik dalam bentuk *floppy disk*, berpura-pura menjadi kuesioner untuk membantu penelitian penyakit AIDS yang sedang menjadi bincangan populer pada saat itu. Apabila *floppy disk* sudah dimasukkan ke dalam perangkat dan *malware* menginfeksi perangkat tersebut, maka kuesioner akan diberikan sebagaimana seharusnya. Saat perangkat dinyalakan untuk ke-19 kalinya, *malware* akan memblokir akses pengguna dan meminta pengguna untuk mengirim uang ke alamat yang tertera pada layar.



GAMBAR 1. INFEKSI *MALWARE AIDS TROJAN*

Seiring berkembangnya internet dan jumlah penggunanya, meningkat juga penyebaran *malware* dan evolusi *malware*. Metode penyebaran *malware* semakin mudah, seperti mengunduh *file* yang terdapat pada *e-mail*, mengunjungi *website* yang mengandung *malware*, atau melalui kanal seperti *bluetooth* untuk membajak perangkat elektronik kita. Jenis-jenis *malware* juga bertambah, salah satu yang paling berdampak bagi dunia adalah *spyware*. *Malware* jenis ini mulai marak tersebar pada sekitar tahun 2010 dengan tujuan untuk menyadap perangkat elektronik dan memperoleh informasi rahasia. Dengan informasi rahasia setiap negara juga tersimpan dalam internet, *Spyware* menjadi senjata utama dalam perang informasi antar negara.

III. STEGANOGRAFI

A. *Sejarah Steganografi*

Steganografi berasal dari bahasa Yunani yang terdiri dari dua kata, yaitu *steganos* yang berarti tersembunyi dan *graphien* yang berarti menulis. Dengan makna tersebut, maka dapat disimpulkan bahwa steganografi adalah ilmu yang mempelajari teknik untuk menulis pesan tersembunyi di dalam suatu pesan sedemikian rupa agar orang tidak mengetahui bahwa terdapat pesan tersembunyi tersebut.

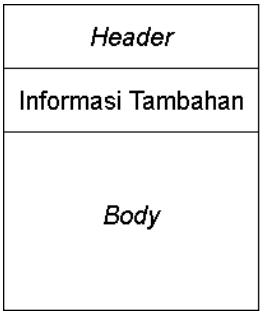
Steganografi sudah digunakan sejak zaman dahulu sebagai teknik komunikasi rahasia dalam perang. Dalam buku "*Histories of Herodotus*" yang ditulis oleh sejarawan Herodotus pada tahun 440 sebelum masehi menceritakan metode steganografi kuno menggunakan kepala budak. Beberapa budak terpilih dicukurkan kepalanya hingga botak, lalu ditatokan pesan rahasia pada kulit kepalanya. Budak dibiarkan rambutnya tumbuh hingga rambut sepenuhnya menutupi kulit rambut yang berisi pesan rahasia. Setelah itu, budak dikirim ke tempat penerima dan kemudian budak tersebut dicukur kembali rambutnya sehingga pesan rahasia tersampaikan ke penerima.

Pada zaman sekarang, steganografi sering digunakan untuk media *digital*. Selain gambar, steganografi juga dapat digunakan untuk menyembunyikan pesan pada teks, video, dan juga audio. Pada makalah ini akan dijelaskan lebih lanjut mengenai teknik steganografi untuk menyembunyikan pesan dalam sebuah gambar.

B. *Steganografi pada Gambar*

1) *Struktur File Gambar*

Sebelum menelusuri teknik steganografi, maka perlu diketahui terlebih dahulu struktur file gambar. Gambar *digital* kita kenali dalam bentuk ekstensi ".png", ".jpg", ".jpeg", ".gif", dan ".svg". Suatu file gambar terdiri dari tiga komponen, yaitu *header,* informasi tambahan, dan *body*.



GAMBAR 2. STRUKTUR FILE GAMBAR

Komponen *header* berisikan *metadata*, yaitu data singkat yang berisikan informasi mengenai *file* tersebut seperti ukuran *file* dan lokasi penyimpanan. Ukuran gambar, resolusi, pembuat file gambar, waktu pembuatan gambar, dan informasi mengenai gambar dalam *file* disimpan pada bagian informasi tambahan. Komponen *body* berisikan gambar yang kita kemudian akan lihat pada layar ketika membuka gambar.

2) *Steganografi pada Header*

Untuk steganografi pada *header* file gambar tidak memiliki teknik khusus. Penyembunyian pesan dilakukan hanya dengan menulis pesan rahasia pada *header file*. Pesan ini sangat mudah ditemukan karena perubahan pada *header* file mudah terdeteksi oleh kita sehingga jarang sekali steganografi dilakukan pada *header* sebuah file.

3) *Steganografi pada Informasi Tambahan*

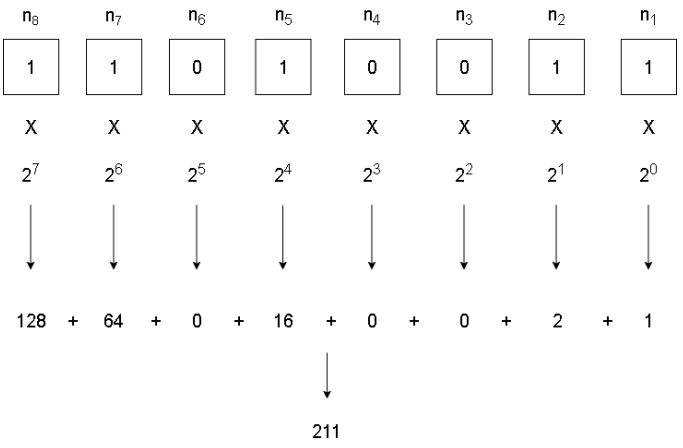
Sama seperti *header*, tidak ada teknik khusus untuk menulis pesan rahasia pada bagian informasi tambahan. Berbeda dengan steganografi pada *header*, pesan rahasia yang dimasukkan pada segmen informasi tambahan tidak mudah ditemukan oleh perangkat kita karena perangkat kita hanya mengecek kondisi *file* berdasarkan *header* sehingga sekilas tidak terlihat bahwa *file* gambar telah dimodifikasi.

4) *Steganografi pada Body*

Metode steganografi paling umum yang digunakan untuk *body* gambar disebut dengan metode *Least Significant Bit* (LSB). Suatu gambar terdiri atas balok-balok kecil yang disebut dengan *pixel*. Setiap *pixel* mengandung suatu warna sehingga kumpulan *pixel* tersebut membentuk gambar utuh yang kita lihat pada layar. Setiap warna *pixel* terbentuk atas tiga spektrum warna, yaitu merah, hijau, dan biru. Dalam dunia desain dan teknologi ketiga spektrum warna ini dinamai *RGB*. Masing-masing spektrum warna dapat bernilai dari 0 hingga 255 dengan nilai *RGB* (0,0,0) menunjukkan warna

hitam dan *RGB* (255,255,255) menunjukkan warna putih. Meskipun kita dapat menilai setiap komponen warna dari angka 0 hingga 255, perangkat kita hanya dapat mengenali angka dalam bentuk *bit* yang bernilai 0 atau 1. Cara perangkat kita dapat menghitung angka 0-255 adalah dengan melihat kombinasi nilai 8 buah bit. Karena setiap *bit* memiliki 2 kemungkinan nilai (0 dan 1), maka kombinasi 8 *bit* akan memiliki 2^8 kemungkinan nilai, atau sebanyak 256 kemungkinan nilai. Setiap komponen warna direpresentasikan dengan 8 bit.

Setiap nilai *bit* dikalikan dengan bilangan 2^{n-1} dengan n adalah bit ke-berapa pada kombinasi 8 bit tersebut. Hitungan n = 1 dimulai dari paling kanan dan berakhir di kiri seperti pada gambar 3. Setelah setiap nilai *bit* telah dikalikan, maka penjumlahan semua hasil perkalian adalah nilai komponen warna yang dicari.



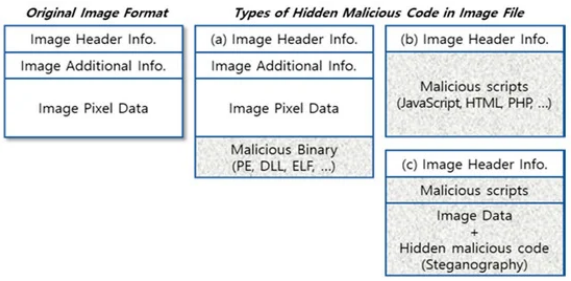
GAMBAR 3. PERHITUNGAN NILAI KOMPONEN WARNA *RGB*

Karena bit paling kanan hanya mempengaruhi nilai komponen warna sebesar 1, maka kita sebut bit tersebut sebagai *least significant bit* (LSB). Apabila nilai LSB diubah untuk pixel dalam sebuah gambar, maka perubahan warna gambar tidak dapat terlihat oleh kasat mata. Maka, steganografi dapat dilakukan dengan mengubah nilai LSB dari pixel-pixel dalam gambar sesuai dengan pesan rahasia yang ingin kita masukkan. Ketika ingin melihat pesan rahasia, maka penerima hanya perlu menyambungkan LSB dari pixel-pixel tersebut untuk mendapatkan pesan rahasia awal. Gambar yang telah dimodifikasi tidak terlihat berbeda dari gambar awal jika dilihat sekilas sehingga metode LSB efektif untuk menyembunyikan pesan dalam *body* file gambar.

IV. STEGOSPLOIT

Stegosploit merupakan kombinasi kata steganography dan exploit yang berarti *stegosploit* adalah penyalahgunaan steganografi untuk tindak kejahatan yang merugikan pihak tertentu. Pada makalah ini diteliti bagaimana steganografi dimanfaatkan oleh para *cyber criminal* untuk melakukan tindak kejahatan.

Terdapat tiga metode *stegosploit* terkait gambar yang dapat dilihat pada gambar 4.



GAMBAR 4. METODE *STEGOSPLOIT* PADA FILE GAMBAR

A. End of Image (EOI)

Metode *stegosploit* ini hanya menambahkan serangkaian kode biner di akhir file gambar, atau setelah *end of image*(EOI). Setelah gambar dibuka, kode tereksekusi, baik itu untuk mengambil informasi rahasia pengguna atau menjalankan kode untuk mengaktifkan *malware* tertentu. Karena metode ini hanya menambahkan kode pada file gambar yang sudah ada, maka terdapat perbedaan ukuran file awal dengan yang telah dimodifikasi.

B. Fake Image File

Pada metode *stegosploit* ini, *header* file tidak diubah sehingga perangkat kita mengidentifikasi file tersebut sebagai sebuah gambar, tetapi isi dari *file* tersebut adalah rangkaian kode yang akan menginfeksi perangkat kita dengan *malware*. Apabila kita mencoba untuk membuka file "gambar" tersebut, maka kode *malware* akan dijalankan sehingga menginfeksi perangkat kita dengan *malware* tersebut.

C. Thoroughly Infected Image File

Malware dapat disembunyikan dalam segmen informasi tambahan file gambar dalam bentuk kode perintah. Apabila file gambar dibuka dan bagian tersebut dibaca, maka kode perintah akan dieksekusi dan *malware* menginfeksi perangkat kita. Selain segment informasi tambahan, *malware* juga dapat disembunyikan dalam gambar itu sendiri dengan metode LSB yang sudah dijelaskan poin III.B.4, atau teknik steganografi gambar lainnya. Metode *stegosploit* ini sangat sulit untuk dideteksi oleh perangkat lunak anti-virus dengan model deteksi *malware* yang digunakan saat ini.

V. STAGANALYZERS

Dengan potensi bahaya *stegosploit* yang sulit untuk dideteksi oleh anti-virus dan metode deteksi *malware* saat ini, beberapa algoritma telah diajukan sebagai solusi untuk menangani kasus-kasus stegosploit. Algoritma-algoritma untuk mendeteksi kegunaan steganografi pada suatu media disebut dengan *staganalyzers*. Pada bab ini akan diteliti jenis-jenis *staganalyzer* yang digunakan untuk masalah *stegosploit* pada gambar.

A. Staganalyzer Deteksi Umum

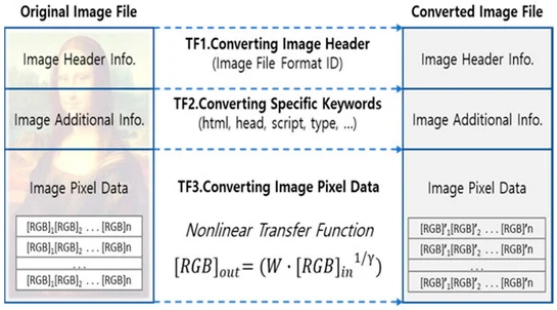
Staganalyzer umum yang digunakan untuk deteksi *malware* pada gambar menggunakan teknik yang menganalisis langsung gambar yang telah dimodifikasi dan memprediksi berdasarkan pola tertentu apakah terdapat pesan rahasia di dalam gambar tersebut atau tidak. Kharazzi et al. [4] telah melakukan eksperimen untuk menganalisis kinerja dari 3 *staganalyzer* deteksi umum, yaitu *binary similarity measures* (BSM), *wavelet-based steganalysis* (WBS), dan *feature-based steganalysis* (FBS). Kharazzi et al. [4] menemukan bahwa kinerja ketiga *staganalyzer* sangat dipengaruhi oleh kualitas gambar. Gambar berkualitas tinggi cenderung untuk lebih sulit dideteksi perubahannya dibandingkan gambar berkualitas rendah. Penelitian Kharazzi et al. [4] juga tidak dapat menjawab hubungan antara ukuran gambar dengan kinerja *staganalyzer*. Menurut hipotesis Kharazzi et al. [4], kinerja *staganalyzer* kemungkinan besar akan menurun jika ukuran gambar semakin kecil akibat berkurangnya jumlah data yang didapatkan dari gambar.

B. Algoritma Netralisasi Stegosploit

Karena pendeteksian *stegosploit* masih memiliki kemungkinan salah dalam prediksinya, maka lebih baik apabila solusi terhadap stegosploit adalah untuk menghapus atau menetralisir rangkaian kode program *malware* di dalam file terinfeksi sebelum dibuka oleh pengguna.

1) ImageDetox

Jung et al. [2] mengajukan sebuah algoritma yang dapat menghapus atau menetralisir rangkaian kode program *malware* pada suatu file gambar yang disebut dengan algoritma ImageDetox. ImageDetox didesain untuk dapat menangani ketiga kasus stegosploit yang dijabarkan pada poin IV.A, IV.B, dan IV.C. ImageDetox bekerja dalam 3 tahap yang dapat dilihat pada gambar 5.



GAMBAR 5. TAHAP-TAHAP ALGORITMA IMAGEDETOX

Tahap pertama (TF1) dimulai dengan mengonversi informasi yang tersimpan pada *header*. Tahap pertama ini bertujuan untuk menghilangkan kode biner *malware* yang ditambahkan ke gambar setelah EOI. Pada konversi *header*, hanya informasi dan data sebelum EOI yang dikonversikan menjadi file gambar yang "dibersihkan".

Tahap kedua (TF2) dilakukan pada segmen yang mengandung informasi tambahan mengenai gambar. Pada segmen ini dicari beberapa kata yang merupakan indikasi bahwa bagian kode itu merupakan sebuah kode *malware*. Apabila ditemukan kata-kata kunci tersebut di dalam bagian informasi tambahan, maka algoritma akan menggantikan kata tersebut dengan nilai lain. Ukuran kata dan nilai pengganti adalah sama sehingga perubahan pada TF2 tidak akan memengaruhi ukuran dari file gambar. Akibat sebagian kode *malware* telah diganti, maka kode tidak dapat berjalan sehingga secara efektif telah ternetralisir.

Tahap ketiga (TF3) dilakukan pada *body* dari file gambar, yaitu gambar itu sendiri yang terdiri dari *pixel-pixel*. Tujuan dari TF3 adalah kita ingin konversi nilai *RGB* dari setiap pixel seminimal mungkin agar kualitas gambar serupa dengan asli, namun kode *malware* ternetralisir. Konversi nilai *RGB* dilakukan dengan sebuah fungsi transformasi non-linear yang tertera di bawah ini.

$$[RGB_{output}] = (W \, x \, [RGB_{Input}]^{1/\gamma})$$

Keterangan:

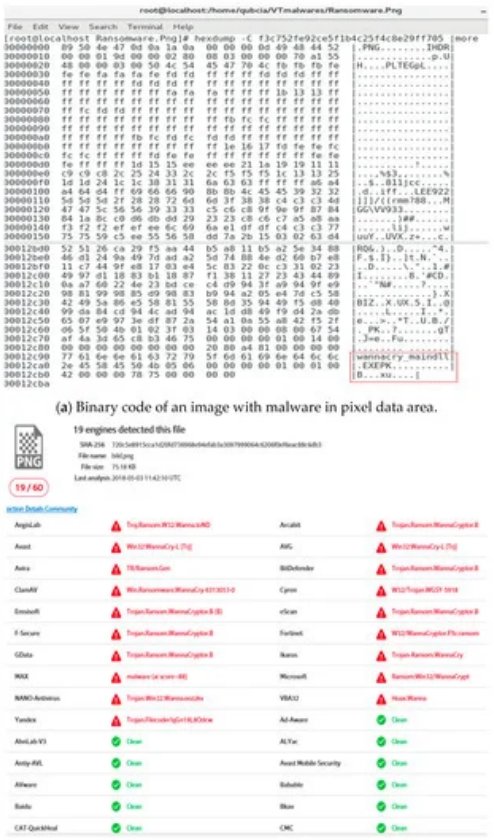
- W adalah *alpha channel* dari pixel tersebut. *Alpha channel* adalah nilai yang menandakan transparansi dari warna pada pixel tersebut.
- Gamma (γ) adalah suatu nilai dalam berada dalam rentang nilai tertentu yang menandakan sebuah fungsi non-linear.

Dalam eksperimen yang dilakukan oleh Jung et al. [2] mengenai rentang nilai gamma (γ) didapatkan bahwa rentang nilai yang terlalu lebar akan sangat merubah gambar sehingga tidak serupa dengan aslinya. Apabila nilai gamma (γ) mendekati angka 1, maka perubahan nilai *RGB* sangat kecil sehingga kode *malware* tetap dapat dijalankan. Didapatkan rentang gamma (γ) yang optimal adalah ($0.950 < \gamma < 0.995$ atau $1.005 < \gamma < 1.050$). Batasan nilai ini menjamin bahwa LSB dari setiap pixel pada gambar hanya akan bergeser sekitar 1-4 bit saja. Perubahan LSB setiap pixel akan merusak kode *malware* sehingga *malware* tidak dapat menjalankan kode.

Proses pengujian algoritma dilakukan oleh Jung et al. [2] dengan menguji coba kinerja setiap tahapan, lalu mengetes kinerja keseluruhan dengan menggunakan perangkat lunak anti-virus VirusTotal.

1) Pengujian Kinerja TF1

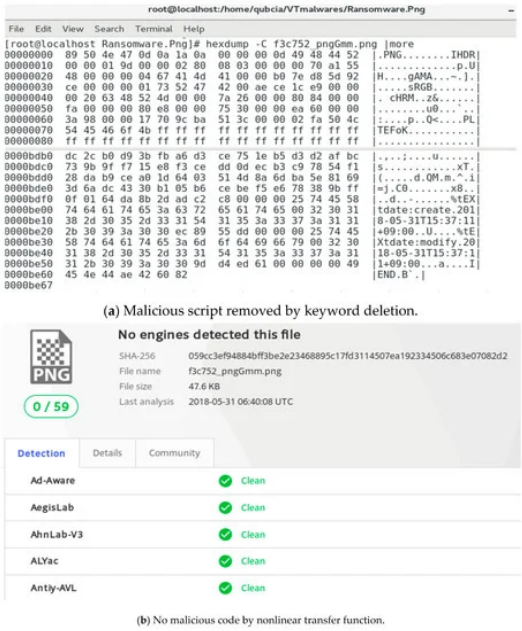
Pada gambar 6 terlihat bahwa di akhir file terdapat sebuah kode *malware*. Ketika dimasukkan ke VirusTotal, terlihat bahwa terdeteksi *malware* di dalam file.



(a) Binary code of an image with malware in pixel data area.

GAMBAR 6. PENGUJIAN KINERJA TF1 (SEBELUM)

Setelah file melewati proses TF1, terlihat pada gambar 7 bahwa VirusTotal sudah tidak mendeteksi adanya *malware* di dalam file gambar.



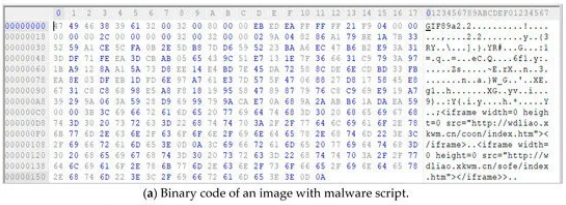
(a) Malicious script removed by keyword deletion.

(b) No malicious code by nonlinear transfer function.

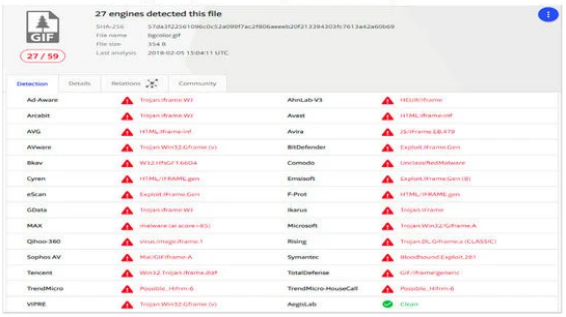
GAMBAR 7. PENGUJIAN KINERJA TF1 (SESUDAH)

2) Pengujian Kinerja TF2

Pengujian TF2 dilakukan dengan menyisipkan serangkaian kode *malware* pada segment informasi tambahan pada file gambar. Terlihat pada gambar 8 bahwa VirusTotal mendeteksi adanya *malware* di dalam file.



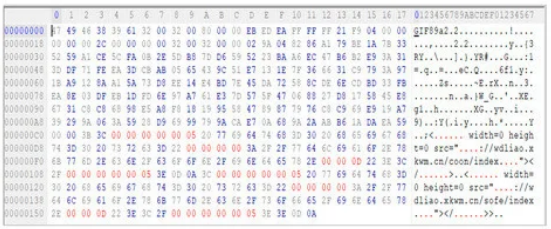
(a) Binary code of an image with malware script.



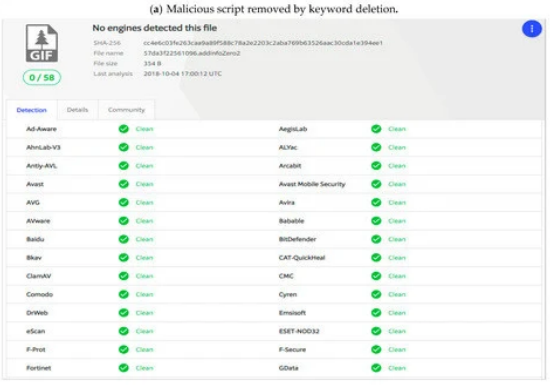
(b) Detection of malware hidden in additional information area.

GAMBAR 8. PENGUJIAN KINERJA TF2 (SEBELUM)

Setelah file gambar melewati proses TF2, terlihat pada gambar 9 bahwa VirusTotal sudah tidak mendeteksi adanya *malware* di file gambar tersebut.



(a) Malicious script removed by keyword deletion.

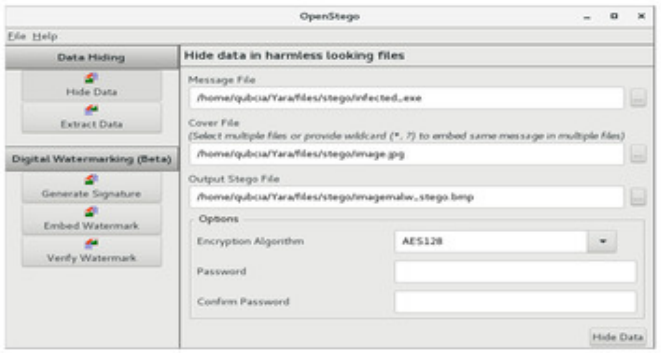


(b) No malicious code by keyword deletion.

GAMBAR 9. PENGUJIAN KINERJA TF2 (SESUDAH)

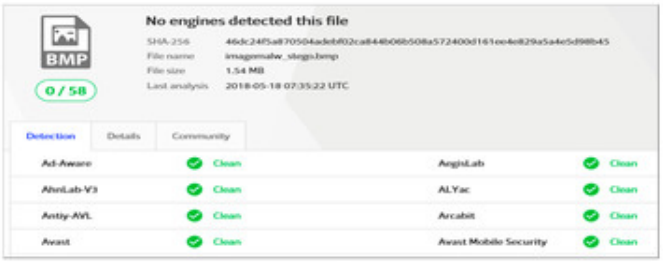
3) Pengujian Kinerja TF3

Pengujian TF3 dilakukan dengan menyembunyikan *malware* di dalam *body* sebuah file gambar menggunakan OpenStego seperti yang terlihat pada gambar 10.



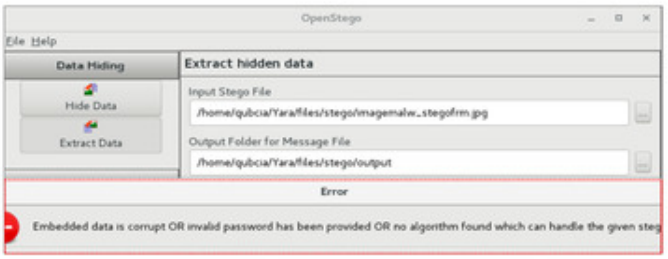
GAMBAR 10. PEMASUKKAN *MALWARE* DALAM FILE GAMBAR MENGGUNAKAN OPENSTEGO

Setelah file gambar melalui proses TF3, maka dicek kembali dengan VirusTotal mengenai keberadaan *malware*. Terlihat pada gambar 11 bahwa *malware* sudah tidak terdeteksi.



GAMBAR 11. PENGUJIAN KINERJA TF3 PERTAMA

Selain menggunakan VirusTotal, pengujian TF3 juga menggunakan fungsi OpenStego untuk menjalankan kode *malware* yang tersembunyi pada gambar. Pada gambar 12 terlihat bahwa kode *malware* tidak dapat dieksekusi yang menandakan bahwa kode *malware* berhasil dinetralisir.



GAMBAR 12. PENGUJIAN KINERJA TF3 KEDUA

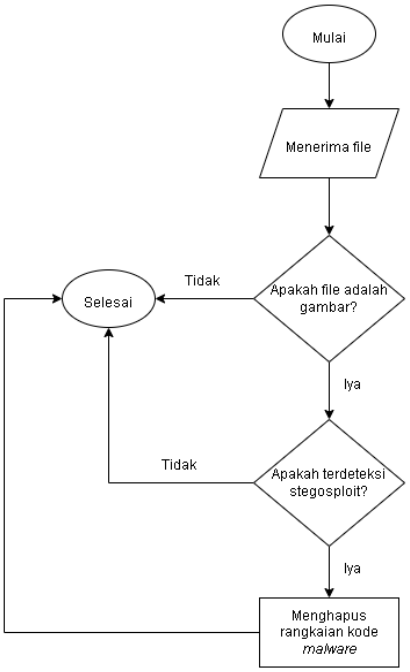
4) Pengujian Kinerja Algoritma

Kinerja algoritma ImageDetox secara keseluruhan diuji dengan mencoba algoritma untuk berbagai jenis *malware*. Pada pengujian yang dilakukan oleh Jung et al.[2], algoritma diuji terhadap 30 file yang terinfeksi berbagai jenis *malware*. Pengujian dilakukan dengan melihat berapa banyak *malware* yang terdeteksi oleh VirusTotal pada awalnya dibandingkan banyaknya *malware* yang terdeteksi oleh VirusTotal setelah algoritma dijalankan. Hasil dari pengujian adalah hampir seluruh pengujian dengan VirusTotal menunjukkan tidak ada *malware* yang terdeteksi untuk file gambar setelah dibersihkan dengan algoritma. Untuk beberapa pengujian VirusTotal dengan 1-3 *malware* terdeteksi, pengujian lebih lanjut dilakukan dengan fungsi OpenStego untuk menguji apakah kode *malware* tetap dapat dijalankan atau tidak. Pengujian lebih lanjut menunjukkan bahwa kode *malware* semua file gambar tersebut tidak dapat dijalankan.

Dari hasil pengujian ini, didapatkan hasil memuaskan bahwa algoritma ImageDetox efektif dan akurat dalam menetralisir kode *malware* yang disembunyikan pada gambar dengan *stegosploit.*

2) Algoritma Vaidya et al.

Dengan konsep menetralisir stegosploit, Vaidya et al. [10] mengajukan algoritma dengan implementasi yang berbeda dari ImageDetox. Algoritma yang dirancang oleh Vaidya et al. [10] hanya mencakup deteksi stegosploit untuk *header* dan segmen informasi tambahan dari file gambar yang dalam bentuk teks. Alur kerja dari algoritma tersebut dapat dilihat pada gambar 13.



GAMBAR 13. ALUR KERJA ALGORITMA VAIDYA ET AL.[10]

Algoritma dapat menerima semua jenis file, namun algoritma hanya akan memproses file bertipe gambar. sesuai dengan yang tertera pada bab III.1. Setelah mengonfirmasi bahwa suatu file adalah file gambar, ditelusuri kode file untuk mengecek apakah terjadi stegosploit. Karena kode *malware* yang disembunyikan dalam segmen informasi tambahan pada file gambar adalah kode yang dapat dijalankan untuk mengeksekusi operasi tertentu, terdapat beberapa kata yang sudah pasti mengindikasi bahwa terdapat kode *malware* di dalam file tersebut. Apabila algoritma mendeteksi kata-kata tersebut, maka rangkaian kode *malware* akan ditelusuri dan kemudian dihapus dari file gambar tersebut.

Hasil pengujian yang dilakukan oleh Vaidya et al. [10] menunjukkan bahwa algoritma bekerja dengan efisien, cepat, dan mampu mendeteksi seluruh stegosploit yang diuji oleh mereka.

VI. Kesimpulan

Stegosploit pada gambar *digital* merupakan ancaman berbahaya bagi pengguna internet akibat kesulitan dalam mendeteksi *stegosploit*. Berdasarkan algoritma-algoritma yang sudah diajukan sebagai solusi terhadap ancaman *stegosploit*, algoritma yang paling efektif dalam menangani *stegosploit* pada gambar adalah untuk menetralisir atau menghapus kode *malware* yang tersembunyi di dalam file gambar. Dari hasil perbandingan dengan algoritma-algoritma lainnya, algoritma ImageDetox merupakan algoritma terbaik untuk melakukan netralisasi terhadap kode *malware* pada file gambar karena algoritma mampu menetralisir kode *malware* untuk ketiga skenario *stegosploit* yang mungkin terjadi.

Video Link at Youtube

-

Acknowledgment

Saya ingin mengucapkan terima kasih kepada Dr. Ir. Rinaldi Munir, M.T. selaku dosen pengampu mata kuliah II4031 Kriptografi dan Koding atas ilmu yang telah disampaikan kepada saya perihal kriptografi.

References

[1] Aslan, Ömer Aslan, and Refik Samet. "A comprehensive review on malware detection approaches." *IEEE Access* 8 (2020): 6249-6271.

[2] Jung, Dong-Seob, Sang-Joon Lee, and Ieck-Chae Euom. 2020. "ImageDetox: Method for the Neutralization of Malicious Code Hidden in Image Files" *Symmetry* 12, no. 10: 1621. https://doi.org/10.3390/sym12101621

[3] Kaspersky. n.d. "Malware & Computer Facts & FAQs". URL: https://www.kaspersky.com/resource-center/threats/computer-viruses-and-malware-facts-and-faqs

[4] Kharrazi, Mehdi, Husrev T. Sencar, and Nasir Memon. 2006. "Performance study of common image steganography and steganalysis techniques". Journal of Electronic Imaging 15(4), 041104.

[5] Park, Byungho, Dukyun Kim, and Daecheol Shin. "A Study on a Method Protecting a Secure Network against a Hidden Malicious Code in the Image." *Indian Journal of Science and Technology* 8 (2015): 26.

[6] Rohayah, Ginanjar Wiro Sasmito, dan Oman Sumantri, Siti (2015). "Aplikasi Steganografi untuk Penyisipan Pesan". *Jurnal Informatika*. **9** (7): 976.

[7] Saengphaibul, Val. 2022. "A Brief History of The Evolution of Malware". URL: https://www.fortinet.com/blog/threat-research/evolution-of-malware

[8] Sentinel One. 2019. "Hiding Code Inside Images: How Malware Uses Steganography". URL: https://www.sentinelone.com/blog/hiding-code-inside-images-malware-steganography/

[9] Toulas, Bill. 2022. "Worok Hackers Hide New Malware In PNGs Using Steganography" URL: https://www.bleepingcomputer.com/news/security/worok-hackers-hide-new-malware-in-pngs-using-steganography/

[10] Vaidya, Neerad and Parag H. Rughani. 2019. "An Efficient Technique to Detect Stegosploit Generated Images on Windows and Linux Subsystem on Windows". International Journal of Computer Sciences and Engineering, Vol. 7, Issue 12. https://doi.org/10.26438/ijcse/v7i12.2126

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, 22 Mei 2023

Kevin Kencana 18219050