JURNAL ILMIAH DAN KARYA MAHASISWA (JIKMA)
Vol.1, No.2 April 2023
e-ISSN: 2985-7732; p-ISSN: 2985-6329, Hal 41-53

ANALISIS MALWARE ANDROID MENGGUNAKAN METODE REVERSE ENGINEERING

Frenvol De Santonario Magno Moises
Universitas Amikom Yogyakarta
Joko Dwi Santoso, M.Kom.
Universitas Amikom Yogyakarta

email : frenvol.19@students.amikom.ac.id , jds@amikom.ac.id

Abstract: *apid progress has been made in the development of Android-based smartphone technology. Communication, shopping and financial transactions are just a few of the daily tasks that people perform with their smartphones. Android, on the other hand, is an open-source operating system that makes it easy for anyone to create Android apps that can be accessed via a smartphone. Counting applications embedded with malware by application, one of which is malware.The analysis was carried out with samples of Trojan malware in the syssecApp.apk application using the reverse engineering method. in doing Trojan malware reverse engineering using APKTOOL tools, JD-GUI.This research will analyze the syssecApp.apk application that infects Trojan malware using the reverse engineering method. The results of the analysis on the syssecApp.apk application found that there is an ip host receiver contained in the source code in syssecApp.apk*
Keywords **:** *Android, APK, Malware, Reverse engineering*

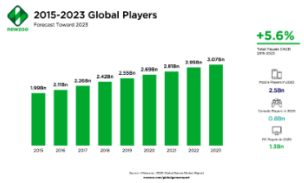
Abstraksi: Kemajuan pesat telah dicapai dalam pengembangan teknologi smartphone berbasis Android. Komunikasi, belanja, dan transaksi keuangan hanyalah beberapa dari tugas sehari-hari yang dilakukan orang dengan smartphone mereka. Android, di sisi lain, adalah sistem operasi open-source yang memudahkan siapa saja untuk membuat aplikasi Android yang dapat diakses melalui smartphone. Menghitung aplikasi yang disematkan malware oleh application, salah satunya adalah malware.Analisis dilakukan dengan sample *malware Trojan* pada aplikasi *syssecApp.apk* dengan menggunakan metode *reverse engineering*. dalam melakukan *reverse engineering malware Trojan* menggunakan *tools APKTOOL*, *JD-GUI*. Penelitian ini akan melakukan analisa terhadap sebauh aplikasi *syssecApp.apk* yang terinfeksi *malware Trojan* dengan menggunakan metode *reverse engineering*. Hasil dari analisis pada aplikasi *syssecApp.apk* ditemukan adanya ip host penerima yang terdapat pada sorce code didalam *syssecApp.apk*
Kata Kunci **:** *Android, APK, Malware, Reverse engineering*

LATAR BELAKANG

Teknologi di balik ponsel cerdas telah berkembang dengan kecepatan yang sangat cepat. Dari anak-anak hingga lansia, jumlah pengguna *smartphone* terus meningkat dari waktu ke waktu. *smartphone* membuat tugas sehari-hari seperti berkomunikasi, berbelanja, dan melakukan transaksi keuangan menjadi lebih mudah dan efektif. Jumlah pemilik *smartphone* Android di seluruh dunia menunjukkan bahwa pengembang jahat atau *malware* terutama

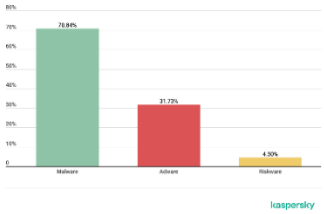
Received Januari 30, 2023; Revised Februari 02, 2023; Maret 22, 2023
*Corresponding author, e-mail address

menargetkan *smartphone* Android. pada tahun ini, sudah ada sekitar 2,5 miliar gamer yang menggunakan perangkat seluler, 0,88 miliar menggunakan *konsol*, dan 1,3 miliar menggunakan PC sebagai *platform* pilihan mereka[1].Seperti yang ditunjukan oleh gambar 1.1.



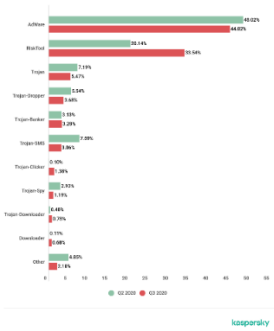
Gambar 1. 1 Data pengguna ponsel cerdas di dunia

Dalam penggunaan bermain game, dibutuhkan aplikasi – aplikasi untuk menunjang performa dari sebuah game. Namun, mengingat banyaknya aplikasi game, hal ini tentunya juga menimbulkan kerentanan keamanan baik untuk aplikasi maupun penggunanya. Menurut data *Kaspersky* dari kuartal ketiga tahun 2020, serangan paling umum pada aplikasi *smartphone* adalah serangan *malware*, dengan *proporsi* 70.84 %[2]. Seperti yang ditunjukan oleh gambar 1.2.



Gambar 1. 2 Data serangan *mobile* aplikasi Q3 2020

Ponsel cerdas menjadi sasaran berbagai jenis *malware*, termasuk *malware Trojan*. *Trojan malware* yang menginstal dirinya sendiri di aplikasi lain dan menunggu koneksi *internet* sebelum terhubung ke *server* untuk melakukan serangan akses awal pada *Command and Control (CnC),* yang digunakan untuk mengirim perintah ke perangkat pengguna dan melakukan kejahatan tindakan. Akibatnya, pengembang jahat memanfaatkan kesempatan ini untuk menyematkan *malware Trojan* di aplikasi Android yang digunakan untuk komunikasi, pendidikan, permainan, dan tujuan lainnya. Dari Q2 ke Q3 tahun 2020, *Trojan malware* meningkat sebesar 1,72 persen, menurut data *Kaspersky*. Seperti yang ditunjukan oleh gambar 1.3.



Atas dasar-dasar masalah diatas maka peneliti memuat sebuah topik penelitian yang berjudul**"Analisis *Malware* Android Menggunakan Metode *Reverse engineering*".** Sebuah metode *analisis statis* yang bertujuan untuk membuka, membaca, dan menemukan kode yang terindikasi *malware* tersebut. dalam analisis *malware* berguna untuk membuka data yang memuat informasi yang ada didalam *malware* [3].

KAJIAN TEORITIS

Malware

Malware yang diartikan dengan bahasa Indonesia adalah perangkat lunak atau program berbahaya, dengan tujuan diciptakan untuk merusak atau menyusup pada sebuah *sistem*, bahkan pada jaringan komputer tanpa diketahui oleh pemilik. Pada saat ini terdapat perkembangan pada *malware*, bukan hanya komputer namun juga *smartphone*.

Virus

Virus merupakan salah satu dari jenis dengan pola penyerangan berupa *file* eksekusi (*.exe*), *Virus* dapat menggandakan diri pada saat *file* yang terjangkit *Virus* dija*LAN*kan. *Virus* dapat digunakan dalam membuat *botnet*, mencuri uang, mencuri informasi hingga merusak jaringan komputer.

Trojan

Trojan biasa dikenal sebagai *Trojan Horse* program yang bias masuk pada *sistem* komputer secara diam diam dan memberikan jalan atau media pada *Virus*, *adware*, *Spyware*, ataupun *malware* lain untuk masuk pada sebuah *sistem*. *Trojan* merupakan salah satu jenis *malware* yang berbahaya karena dapat merusak *sistem*, memungkinkan orang lain untuk mengawasi atau bahkan mengontrol komputer dan mencuri informasi penting.

Backdoor

Backdoor merupakan program yang dirancang untuk memberikan akses berupa koneksi jaringan tanpa diketahui untuk memasukkan *malware* lain seperti *Virus* atau bahkan *spam* yang dikirim untuk *meretas sistem*

Anti-*Malware*

Anti-*malware* yang bisa disebut juga sebagai anti-*Virus* merupakan sebuah program yang diciptakan untuk tujuan untuk mendeteksi, mencegah serta menghapus *malware*

Anomaly-based Detection

Anomaly-based Detection merupakan salah satu dari teknik deteksi dengan berbasis *anomali* yang biasanya terjadi dalam dua *fase*, yaitu *fase training(learning)* dan *fase detection(monitoring*). Selama *fase* training, *detector* akan mencoba mempelajari perilaku *host*

Specification-based Detection

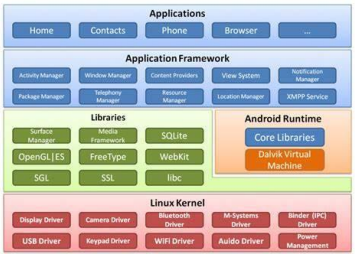
Specification-based Detection adalah deteksi berbasis *anomali* yang mencoba untuk mengatasi *high false* alarm yang sebagian besar terjadi pada deteksi berbasis *anomali*. Dalam deteksi berbasis *spesifikasi*, *fase* training adalah pencapaian beberapa *rule set* atau aturan, yang menentukan semua perilaku *valid* yang dapat ditunjukkan oleh program apa pun yang sedang diperiksa

Signature-based detection

Signature-based detection adalah deteksi dengan berupaya mengambil contoh dari model *malware* dan menggunakannya untuk deteksi *malware*. Kumpulan dari semua model *malware* menggambarkan ciri dari *Signature-based detection*

Android

Android adalah sistem operasi yang dibangun di atas *kernel Linux* dan berbasis *Java. Java* digunakan dalam pengembangan aplikasi Android yang ada untuk memfasilitasi *portabilitas* ke *platform* baru. Agar pengguna dapat menjalankan aplikasi yang diinginkan di ponsel cerdasnya, sistem operasi Android dapat diibaratkan sebagai jembatan. Android adalah sistem operasi seluler yang mencakup sistem operasi, *middleware*, dan berbagai aplikasi dan berbasis *Linux*.



Gambar 2.1 Android *architecture*

Reverse engineering

Reverse engineering atau yang bisa diartikan dalam bahasa Indonesia sebagai rekayasa balik adalah proses dalam menemukan atau mencari prinsip kerja dari sebuah produk teknologi yang ada di sebuah *sistem*, objek atau perangkat, dengan melakukan analisis mendalam pada cara kerja atau fungsi dari *sistem*, objek, atau perangkat yang diteliti.

Assembly

Assembly language adalah salah satu bahasa pemrograman yang ada dalam level rendah dari banyaknya bahasa pemrograman yang ada selama ini. Bahasa *assembly* biasa digunakan dalam membuat mesin dikarenakan mesin tidak bisa membaca bahasa pemrograman tingkat tinggi yang ada saat ini, seperti *basic*, *Java*, *pascal*, dl

Dis*assembly*

Dis*assembly* adalah kebalikan yang ada pada proses *assembly*. Proses dis*assembly* biasa dilakukan pada teknik *reverse engineering* dalam menerjemahkan bahasa mesin ke bahasa yang lebih mudah untuk dipahami atau yang bisa disebut bahasa *assembly*

Debugging

Debugging adalah proses pengujian perangkat lunak. Debugging bertujuan untuk menguji setiap proses inti dalam proses analisis malware

String

String adalah sebuah karakter dalam sebuah *program* dan sebuah nilai dalam prosedur *load* yang akan dilakukan ketika sampel *malware* dijalankan. Karena analisis *string* dapat mengungkap bukti dalam sampel, ini merupakan langkah penting dalam rekayasa terbalik *malware*

Repository malware

Repository malware adalah tempat di mana disimpan sampel sampel *malware* yang tertangkap baik dalam upaya atau yang telah berhasil menyerang *sistem* komputer. *Repository malware* sengaja dibuat dalam upaya membantu analisis *malware* terhadap serangan yang dilakukan.

Apktool

Apktool adalah tool yang sudah terinstal dalam *package Kali Linux* yang berfungsi dalam melakukan *reverse engineering*. *Apktool* dapat digunakan untuk melakukan *reverse engineering* terhadap aplikasi android. Tidak hanya *reverse engineering*, namun *Apktool* juga dapat melakukan *rebuild* atau membangun kembali *file* aplikasi android yang telah dilakukan *reverse engineering*

Kali linux

Kali linux merupakan *sistem* operasi dari salah satu distribusi *Linux* yang termasuk dalam tingkat lanjut untuk audit keamanan atau *penetration testing*. *Kali linux* adalah pengembangan dari *BackTrack linux* yang disempurnakan sepenuhnya dengan mengikuti standar pengembangan dari *Debian*. Infrastruktur yang ada telah diperbaharui, seluruh *tools* dilakukan *review* kembali, serta dalam *Version Control System* (VCS) telah menggunakan *Git*

Genymotion

Genymotion merupakan *virtual enviroment* yang difungsikan sebagai simulasi aplikasi *smartphone* yang dijalankan pada komputer

1. Metodologi

Penulis menggunakan berbagai teknik dalam mewujudkan penelitian ini. Pendekatan yang dilakukan adalah sebagai berikut:

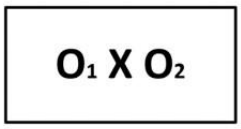
a. Pre-Experimental Design

Karena sama sekali bukan eksperimen, maka desain penelitian yang sebenarnya menggunakan *Pre-Experimental Design*

b. One Group *Pretest Posttest* Design

One Groups Pretest-Posttest Design yang digunakan penelitian ini terdiri dari *pretest* diberikan sebelum *treatment* dan *posttest* diberikan setelah *treatment* [48]. Dari segi karakteristik dan kondisi percobaan, tidak semuanya dapat dikontrol secara ketat. Rancangan penelitian secara rinci ditunjukkan pada Gambar 3.1.

e-ISSN: 2985-7732; p-ISSN: 2985-6329, Hal 01-09



Gambar 3. 1 Rumus One Group *Pretest-Posttest* Design

Keterangan :

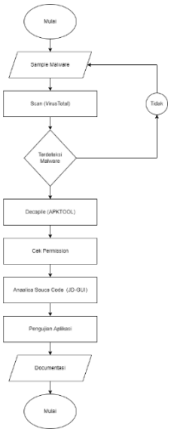
O1 = Nilai *Pretest*, yaitu analisis aplikasi sebelum infeksi *malware*.

X = Treatment (perlakuan), melakukan infeksi *malware* menggunakan Metasploit.

O2 = Nilai *Pretest*, yaitu analisis hasil dari aplikasi setelah infeksi *malware*.

Flowchart Penelitian

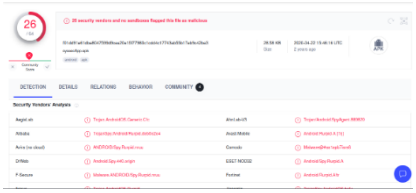
Flowchart atau tahapan penelitian yang dilakukan selama penelitian adalah *flowchart* penelitian. Dari awal hingga akhir penelitian, bagan alir penelitian berfungsi sebagai panduan. Gambar 3.1 menggambarkan *flowchart* penelitian yang digunakan oleh penulis dalam penelitian ini.



Gambar Flowchart Penelitian

2. HASIL DAN PEMBAHASAN

Pra-Analisis

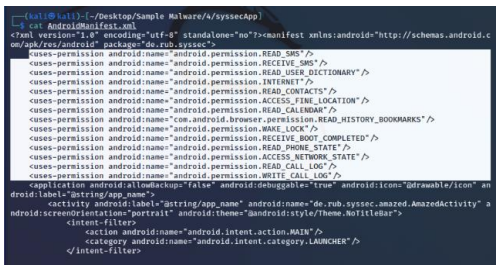


informasi nilai *checksum* dari aplikasi "*syssecApp.apk*" menggunakan *Virustotal* dengan nilai *checksum* *SHA-256*

"f01dd91e61dbe8047599d9cea20e15f77980c1cdd4c17743ab55b17eb9c42be3", nilai *checksum* MD5 "f8a2c86aa30d55c71510252a16b23648 " dan *checksum* SHA-1 "f75f91cf7cf72dfd223ea7a5ee8059648864411a ". File sampel *malware* adalah file Android dengan ukuran 28,58 KB. File Android adalah jenis file yang sering digunakan untuk penginstalan perangkat lunak dan distribusi file di sistem operasi Android.

Permission analysis

Dalam *android* tentunya terdapat sebuah *file* yang bernama *AndroidManifest.xml*. *File* android *Manifest* ini terletak pada *root* sebuah aplikasi. *File manifes* adalah salah satu *file* penting dalam APK paket *kompresi*, yang berisi *informasi perizinan* android dan informasi komponen aplikasi. Mekanisme izinnya adalah dasar dari keamanan *sistem operasi* android, dimana android mengambil *kontrol akses ke API* yang digunakan untuk proses perizinan informasi dari *smartphone.*



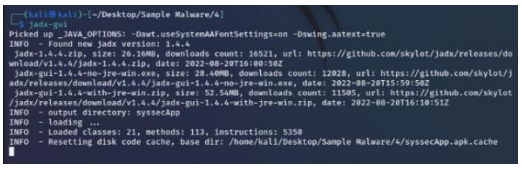
permission yang ada pada aplikasi ini didapatkan 14 buah *permissions* yang dapat dilihat pada gambar 4.31. *Detail* lengkap mengenai *permission* sample *malware* pada aplikasi "*syssecApp.apk*" dapat dilihat pada tabel 4.1.

No	Permission	Status	Info	Deskripsi
1	android.permission.READ_SMS	Dangerous	read SMS or MMS	Allows Application to read SMS messages stored on your phone or SIM card. Malicious Applications may read your confidential messages.
2	android.permission.RECEIVE_SMS	Dangerous	receive SMS	Allows Application to receive and process SMS messages. Malicious Applications may monitor your messages or delete them without showing them to you.
3	android.permission.READ_USER_DICTIONARY	Dangerous	read user-defined dictionary	Allows an Application to read any private words, names and phrases that the user may have stored in the user dictionary.
4	android.permission.INTERNET	Normal	full Internet access	Allows an Application to create network sockets.
5	android.permission.READ_CONTACTS	Dangerous	read contact data	Allows an Application to read all of the contact (address) data stored on your phone. Malicious Applications can use this to send your data to other people.
6	android.permission.ACCESS_FINE_LOCATION	Dangerous	fine (GPS) location	Access fine location sources, such as the Global Positioning System on the phone, where available. Malicious Applications can use this to determine where you are and may consume additional battery power.
7	android.permission.READ_CALENDAR	Dangerous	read calendar events	Allows an Application to read all of the calendar events stored on your phone. Malicious Applications can use this to send your calendar events to other people.
8	com.android.browser.permission.READ_HISTORY_BOOKMARKS	unknown	Unknown permission	Unknown permission from android reference
9	android.permission.WAKE_LOCK	Normal	prevent phone from sleeping	Allows an Application to prevent the phone from going to sleep.
10	android.permission.RECEIVE_BOOT_COMPLETED	Normal	automatically start at boot	Allows an Application to start itself as soon as the System has finished booting. This can make it take longer to start the phone and allow the Application to slow down the overall phone by always running.
11	android.permission.READ_PHONE_STATE	Dangerous	read phone state and identity	Allows the Application to access the phone features of the device. An Application with this permission can determine the phone number and serial number of this phone, whether a call is active, the number that call is connected to and so on.
12	android.permission.ACCESS_NETWORK_STATE	Normal	View network status	view network status Allows an Application to view the status of all networks.
13	android.permission.READ_CALL_LOG	Dangerous		Allows an Application to read the user's call log.
14	android.permission.WRITE_CALL_LOG	Dangerous		Allows an Application to write (but not read) the user's call log data.

Analisis Souce Code

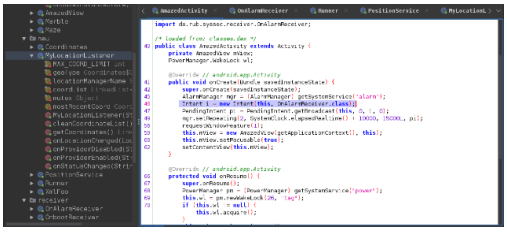
e-ISSN: 2985-7732; p-ISSN: 2985-6329, Hal 01-09

Melakukan analisa *source code* merupakan hal penting dalam analisis *malware* dengan metode *reverse engineering*. Tahapan ini diperlukan dalam mencari kode yang bersifat *malicious* atau berbahaya. Penulis melakukan analisa terhadap *source code* dari aplikasi *syssecApp.apk* menggunakan *tools JD-GUI*, yaitu *tools* yang digunakan membuka *file* jar dengan menampilkan kode sumber *Java* dari *file* "*.class*".



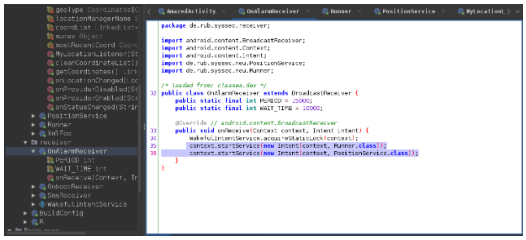
Gambar 4.14 *Decompile*r *file Java* (*JD-GUI)*

Salah satu dari sekian banyak *folder* yang sudah ada memiliki nama yang sulit dipahami dan membuatnya tampak mencurigakan. Saat dibuka, *folder* tersebut berisi sejumlah kelas dengan nama yang dipertanyakan. Seperti yang digambarkan pada Gambar 4.33,



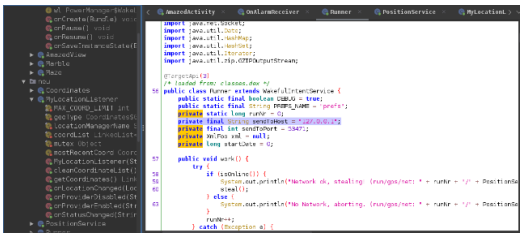
Gambar 4.15 *class* yang pertama dijalankan

Gambar diatas ketika aplikasi berjalan pertama *Kali* menjalankan perintah *OnAlaramReceiver.class*, dari *class OnAlaramReceiver* dia memangil sebuah clas *Runner* dan *PositionService*



Gambar 4.16 *Class OnAlaramReceiver*

Dalam sebuah foler *Runner* kami menemukan sebuh program yang bertujuan untuk mengirim *file* ke *host*, seperti gambar dibawah

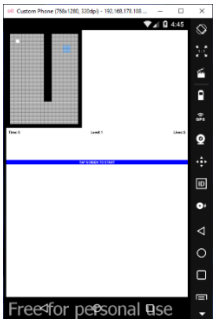


Gambar 4.17 *class* yang berfungsi sebagai *host*

Dalam gambar diatas terdapat *String* yang berfungsi sebagai *host* ip, dan semua data dari korban akan dikirim ke *host* tsb, seperti pada gambar dibawah

Gambar 4.18 fungsi *class* data yang diambil

Dari gambar di atas ada beberapa data yang bias diamabil seperti Imai, sms, bookmarks, contact, location dll.



Gamabr 4.31 Runing aplikasi *syssecApp.apk* di *emulator*

Dari gambar diatas bahwa aplikasi *syssecApp.apk* hanya sebuah game bermain anak-anak.

KESIMPULAN DAN SARAN

Kesimpulan

a. Tahapan untuk melakuakn anaslisa *statis* mengunakan reverse engennering pada aplikasi *syssecApp.apk* yaitu sebahai berikut :
b. Mengidentifikasi sample aplikasi *syssecApp.apk* mengunakan *Virus* total. Ditemukna bahwa aplikasi tersebut ter identifikasi 26 anti*malware* dari 64.

c. Menghitung *checksum* (hasing) aplikasi. Didapatkan hasing SHA 256 aplikasi *syssecApp.apk* yaitu " f01dd91e61dbe8047599d9cea20e15f77980c1cdd4c17743ab55b17eb9c42be3"
d. Melakukan *Decompile*r aplikasi menggunakan *Apktool* dan didapatkan hasil data aplikasi seperti *certificate*, *source*, *Smali* dan *AndroidManifest.xml*
e. Memasukkan analisis *permission* pada aplikasi *syssecApp.apk* dan ditemukan 13 *permission* pada aplikasi tersebut.
f. Melakukan analisis *source code* pada aplikasi *syssecApp.apk* menggunakan *tools JD-GUI* dan ditemukan 1 *class* yang berfungsi untuk mengirim *file* data ke *host (127.0.0.1)* merupakan *localhot* bukan *Ip Publik* milik *attacker*.

Analisis dinamis yang dilakukan pada *emulator Genymotion* dengan diberi hak akses full dan berhasil mendapatkan *class* yang berisi *sentHost (127.0.0.1).*

Saran

Penulis penelitian skripsi ini berharap informasi yang diberikan dapat bermanfaat bagi penulis, pembaca, dan pengguna aplikasi Android dalam berbagai cara pada akhirnya. Penulis sangat menyadari keterbatasan metode rekayasa balik untuk menganalisis malware Trojan. Oleh karena itu, penulis dapat memberikan saran sebagai berikut:

a. Penelitian tentang malware sangatlah luas. Teknik deteksi malware mencakup berbasis perilaku, pembelajaran mendalam, deteksi basis tanda tangan, dan rekayasa balik
b. Untuk penelitian selanjutnya, dapat dititik fokuskan dalam pembuatan *antiVirus* sederhana dalam mencegah *malware Trojan.*
c. Pengguna Android harus mengunduh aplikasi dari situs web terkemuka (Google Playstore), dan sebelum menginstal aplikasi, mereka harus terlebih dahulu memeriksa izin akses aplikasi dengan antivirus untuk melihat apakah aplikasi tersebut mengandung malware.
d. Penulis menyarankan pengguna Android untuk menonaktifkan penginstalan perangkat lunak pihak ketiga di pengaturan perangkat. Dengan mencoba meniru pembaruan sistem dan proses serupa lainnya, ini menghilangkan ancaman yang diunduh secara acak. Hapus centang atau nonaktifkan kotak berlabel "Sumber Tidak Dikenal" (Unknown Sources) di pengaturan Android.

DAFTAR REFERENSI

A. S. Christensen, A. Møller, and M. I. Schwartzbach, "*Precise analysis of String expressions*," in *Static Analysis: 10th International Symposium, SAS 2003 San Diego, CA, USA, June 11–13, 2003 Proceedings*, 2003, pp. 1–18.

C. Steinbeck, C. Hoppe, S. Kuhn, M. Floris, R. Guha, and E. L. *Willighagen, "Recent Developments of the chemistry Development kit (CDK)-an open-source Java library for chemo-and bioinformatics," Curr Pharm Des*, vol. 12, no. 17, pp. 2111–2120, 2006.

D. Android, "Android," *Retrieved February*, vol. 24, p. 2011, 2011.

D. Popa, M. Cremene, M. Borda, and K. Boudaoud, "A *security framework* for *mobile* cloud *Application*s," in *2013 11th RoEduNet International Conference*, 2013, pp. 1–4.

D. X. Johansson *et al.*, *"Human combinatorial libraries yield rare antibodies that broadly neutralize hepatitis C Virus," Proceedings of the National Academy of Sciences*, vol. 104, no. 41, pp. 16269–16274, 2007.

E. Chien, "*Techniques of adware and Spyware," in the Proceedings of the Fifteenth Virus Bulletin Conference, Dublin IreLANd*, 2005, vol. 47.

E. S. Blas, "*Pre-experimental designs in psychology and education: A conceptual review*," *Liberabit*, vol. 19, pp. 133–141, 2013.

F. Reghenzani, G. Massari, and W. Fornaciari, "The real-time *linux Kernel*: A survey on preempt_rt," *ACM Computing Surveys (CSUR)*, vol. 52, no. 1, pp. 1–36, 2019.

G. O'Gorman and G. McDonald, *Ransomware: A growing menace*. *Symantec Corporation Arizona*, AZ, USA, 2012.

H. A. Nugroho and Y. Prayudi, "Penggunaan Teknik *Reverse engineering* Pada *Malware* Analysis Untuk Identifikasi Serangan *Malware*," *Universitas Islam Indonesia, Yogyakarta*, 2014.

I. Vessey, "Expertise in *Debugging* computer programs: A *process* analysis," *Int J Man Mach Stud*, vol. 23, no. 5, pp. 459–494, 1985.

J. Hoffmann, M. Ussath, T. Holz, and M. Spreitzenbarth, "Slicing droids: program *slicing for Smali code," in Proceedings of the 28th Annual ACM Symposium on Applied Computing*, 2013, pp. 1844–1851.

J. Lu and W. Hou, "*Decompile Program Graph Design and Controlling Flow Analysis.," National Air Intelligence Center Wright-Patterson Afb Oh*, 1995.

J. Lu, D. Wu, M. Mao, W. *WAN*g, and G. Zhang, "*Recommender System Application Developments*: a survey," *Decis Support Syst*, vol. 74, pp. 12–32, 2015.

K. A. Talha, D. I. Alper, and C. Aydin, "APK *Auditor: Permission-based Android malware* detection *System*," *DiGit Investig*, vol. 13, pp. 1–14, 2015.

K. *Linux*, "*Kali Linux*." *Obtenido de Official Kali Linux* Documentation: http://docs. *Kali*. org …, 2020.

L. Ďurfina, J. Křoustek, and P. Zemek, "PsybOt *malware*: *A step-by-step decompilation case study*," in *2013 20th Working Conference on (WCRE)*, 2013, pp. 449–456.

L. J. Hoffman, *Rogue Programs: Viruses, Worms and Trojan Horses*. Van Nostrand Reinhold Co., 1990.

L. Zeltser, "*Reverse engineering malware*," *Retrieved June*, vol. 13, p. 2010, 2001.

M. Al-Asli and T. A. Ghaleb, "*Review* of *signature*-based techniques in anti*Virus* products," in *2019 International Conference on Computer and Information Sciences (ICCIS)*, 2019, pp. 1–6.

M. Egele, C. Kruegel, E. Kirda, H. Yin, and D. Song, "Dynamic *Spyware* analysis," 2007.

M. Hardt and A. Negri, *Assembly*. Oxford University Press, 2017.

M. O. F. Rokon, R. Islam, A. Darki, E. E. Papalexakis, and M. Faloutsos, "*Source*Finder: Finding *Malware Source-Code from Publicly Available Repositories in GitHub.,"* in *RAID*, 2020, pp. 149–163.

M. Schoeberl, S. Korsholm, T. *Kali*bera, and A. P. Ravn, "A *Hardware Abstraction layer* in *Java*," *ACM Trans Embed Comput Syst*, vol. 10, no. 4, pp. 1–40, 2011.

N. S. Sibarani, G. Munawar, and B. Wisnuadhi, "Analisis performa aplikasi android pada bahasa pemrograman *Java* dan kotlin," in *Prosiding Industrial Research Workshop and National Seminar*, 2018, vol. 9, pp. 319–324.

P. O. Badyorna, "The Use Of *Genymotion At Android Develppment".*

P. Tuli and P. Sahu, "*System* monitoring and *security* using *Keylogger*," *International Journal of Computer Science and Mobile Computing*, vol. 2, no. 3, pp. 106–111, 2013.

R. P. Goldberg, "Survey of *Virtual Machine* research," *Computer (Long Beach Calif)*, vol. 7, no. 6, pp. 34–45, 1974.

R. Pascanu, J. W. Stokes, H. Sanossian, M. Marinescu, and A. Thomas, "*Malware class*ification with recurrent *network*s," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 1916–1920.

R. S. Chakraborty, F. Wolff, S. Paul, C. Papachristou, and S. Bhunia, "MERO: A *stati*stical approach for *Hardware Trojan* detection," in *Cryptographic Hardware and Embedded Systems-CHES 2009: 11th International Workshop Lausanne, SwitzerLANd, September 6-9, 2009 Proceedings*, 2009, pp. 396–410.

R. Salakhutdinov and G. Hinton, "Semantic *Hash*ing," *International Journal of Approximate Reasoning*, vol. 50, no. 7, pp. 969–978, 2009.

R. Sekar *et al.*, "*Specification-based anomaly detection: a new approach for detecting network intrusions*," in *Proceedings of the 9th ACM conference on Computer and communications security*, 2002, pp. 265–274.

S. D. Sergeevich and T. O. Vladimirovich, "*Virus* detection *Backdoor* in microsoft *security* essentials," *International Information Institute (Tokyo). Information*, vol. 18, no. 6 (A), p. 2513, 2015.

S. Febriani, N. B. A. Karna, and R. Nugraha, "Analisis Performansi Enkripsi Pada *Prosesor* Intel Dengan *Arsitektur* X86," *eProceedings of Engineering*, vol. 7, no. 1, 2020.

S. Kim, J. Park, K. Lee, I. You, and K. Yim, "A Brief Survey on *Rootkit* Techniques in *Malicious* Codes.," *J. Internet Serv. Inf. Secur.*, vol. 2, no. 3/4, pp. 134–147, 2012.

S. Sanders and L. Ziarek, "A comparison and contrast of *Apktool* and Soot for injecting blockchain calls into Android *Application*s," in *Proceedings of the Annual Hawaii International Conference on System Sciences*, 2021.

S. Sen, E. Aydogan, and A. I. Aysan, "Coevolution of *mobile malware* and anti-*malware*," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 10, pp. 2563–2574, 2018.

T. Amperiyanto, *P3K Virus Komputer*. Elex Media Komputindo, 2007.

T. R. Knapp, "*Why is the one-group pretest–posttest design still used?," Clinical Nursing Research*, vol. 25, no. 5. SAGE Publications Sage CA: Los Angeles, CA, pp. 467–472, 2016.

T. Varady, R. R. Martin, and J. Cox, "*Reverse engineering* of geometric models—an introduction," *Computer-aided design*, vol. 29, no. 4, pp. 255–268, 1997.

T. Vidas and N. Christin, "Evading android *Runtime analysis via sandbox detection,"* in *Proceedings of the 9th ACM symposium on Information, computer and communications security*, 2014, pp. 447–458.

Tom Wijman, "*Three Billion Players by 2023: Engagement and Revenues Continue to Thrive Across the Global Games Market," Jun 25 2020*, Jan. 25, 2020.

U. Bayer, A. Moser, C. Kruegel, and E. Kirda, "*Dynamic analysis of malicious code," Journal in Computer Virology*, vol. 2, pp. 67–77, 2006.

V. Jyothsna, R. Prasad, and K. M. Prasad, "*A review of anomaly based intrusion detection Systems," Int J Comput Appl*, vol. 28, no. 7, pp. 26–35, 2011.

VICTOR CHEBYSHEV, "*IT threat evolution Q3 2020 Mobile statis*tics," *2020*, Nov. 20, 2020.

X. Bing, W. Xia, J. Gui, G. Yan, X. *WAN*g, and S. Liu, "Diversity and evolution of the Wolbachia endosymbionts of Bemisia (Hemiptera: Aleyrodidae) whiteflies," *Ecol Evol*, vol. 4, no. 13, pp. 2714–2737, 2014.

Y. Zheng, S. Yang, and H. Cheng, "An *Application framework* of di*Git*al twin and its case study," *J Ambient Intell Humaniz Comput*, vol. 10, pp. 1141–1153, 2019.

Z. Chen, E. Brophy, and T. Ward, "*Malware class*ification using *static* dis*assembly* and *mac*hine learning," *arXiv preprint arXiv:2201.07649*, 2021.