Building of Informatics, Technology and Science (BITS)
Volume 4, No 1, Juni 2022 Page: 69−79
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v4i1.1515

Analisis Malicious Software Trojan Downloader Pada Android Menggunakan Teknik Reverse Engineering (Studi Kasus: Kamus Kesehatan v2.apk)

Andriyan Dwi Putra[1,*], Joko Dwi Santoso[2], Ipung Ardiansyah[2]

[1] Fakultas Ilmu Komputer, Sistem Informasi, Universitas AMIKOM, Yogyakarta, Indonesia
[2] Fakultas Ilmu Komputer, Teknik Komputer, Universitas AMIKOM, Yogyakarta, Indonesia
Email: [1,*]andriyan.putra@amikom.ac.id, [2]jds@amikom.ac.id, [3]ipung.19@students.amikom.ac.id
Email Penulis Korespondensi: andriyan.putra@amikom.ac.id
Submitted: **18/04/2022**; Accepted: **11/05/2022**; Published**: 30/06/2022**

Abstrak−Perkembangan teknologi ponsel cerdas berplatform android telah mencapai kemajuan yang sangat pesat. Ponsel cerdas membantu dan memudahkan pekerjaan manusia sehari – hari seperti berkomunikasi, berbelanja, hingga transaksi keuangan. Namun karena android merupakan sistem *open-source*, siapapun dapat dengan mudah mengembangkan aplikasi android yang dapat diunduh di android *app market*. Termasuk aplikasi yang disisipkan *malware* oleh pengembang aplikasi, salah satunya adalah *malware trojan downloader*. Analisis dilakukan dengan mengimplementasikan penginfeksian *malware trojan downloader* pada aplikasi kamus kesehatan menggunakan metode reverse engineering. Penginfeksian *malware trojan downloader* menggunakan *tools metasploit framework.* Aplikasi akan terinfeksi *payload* yang diciptakan dari *metasploit framework*. Penelitian ini akan melakukan analisa terhadap aplikasi kamus kesehatan sebelum dan sesudah terinfeksi *malware trojan downloader* dengan menggunakan teknik *reverse engineering*. Hasil dari analisis pada aplikasi kamus kesehatan ditemukan perbedaan ukuran aplikasi menjadi 10.17 MB yang sebelumnya adalah 10.05 MB. Tentunya dengan perubahan ukuran file, maka secara otomatis juga perubahan pada hashing *SHA256*. Pada bagian *permissions*, ditemukan perbedaan yang sebelumnya hanya ada *9 permissions*, namun setelah terinfeksi ditemukan *18 permissions* tambahan sehingga total keseluruhan menjadi *27 permissions*.

Kata Kunci: Malware; Trojan Downloader; Reverse Engineering; Metasploit Framework; Android

Abstract−The development of smartphone technology with the Android platform has made very rapid progress. Smartphones help and facilitate daily human work such as communicating, shopping, and financial transactions. However, because android is an open- source system, anyone can easily develop android applications that can be downloaded on the android app market. Including applications that have been inserted by malware by application developers, one of which is the Trojan downloader malware. The analysis was carried out by implementing the Trojan downloader malware infection in the health dictionary application using the reverse engineering method. Trojan downloader malware infection uses metasploit framework tools. The application will be infected with the payload created from the metasploit framework. This study will analyze the health dictionary application before and after being infected with the Trojan downloader malware using the reverse engineering method. The results of the analysis on the health dictionary application found that the difference in the size of the application was 10.17 MB, which previously was 10.05 MB. Of course, by changing the file size, the SHA256 hashing changes automatically. In the permissions section, it was found that there were only 9 permissions before being infected, but after being infected, we found 18 additional permissions, bringing the total to 27 permissions.
Keywords: Malware; Trojan Downloader; Reverse Engineering; Metasploit Framework; Android

1. PENDAHULUAN

Perkembangan teknologi ponsel cerdas telah mencapai kemajuan yang sangat pesat. Jumlah pengguna ponsel cerdas pun terus meningkat seiring berjalannya waktu, baik dari kalangan anak – anak hingga orang tua. Ponsel cerdas membantu dan memudahkan pekerjaan manusia sehari – hari seperti berkomunikasi, berbelanja, hingga transaksi keuangan. Ponsel cerdas dengan sistem operasi android menjadi sasaran utama bagi pengembang jahat atau malware, hal tersebut dapat dilihat dari jumlah pengguna ponsel cerdas android di dunia mencapai 72,48% dan menyisakan 26,91% untuk iOS, 0,23% untuk Samsung, 0,14% untuk Unknown, 0,13% untuk kaiOS, dan 0,02% untuk Windows pada bulan desember 2020 [1].

Dalam penggunaan ponsel cerdas, dibutuhkan aplikasi – aplikasi untuk menunjang kegiatan pengguna dalam kehidupan sehari – hari. Namun seiring banyaknya aplikasi ponsel cerdas, tentunya juga membuka celah keamanan baik melalui aplikasi itu sendiri maupun pengguna aplikasi tersebut. Berdasarkan data Q3 (quarter ketiga) 2020 dari Kaspersky, serangan yang paling banyak menyerang aplikasi pada ponsel cerdas yaitu serangan *malware* dengan persentase 70.84% [2]. Banyak jenis – jenis *malware* yang menyerang ponsel cerdas, salah satunya yaitu *malware trojan downloader*. Sebuah *malware trojan* yang memasang diri sendirinya ke dalam aplikasi lain dan menunggu hingga sambungan internet tersedia untuk tersambung ke server guna dapat melakukan serangan *initial access* kepada *CnC-nya* yang digunakan untuk mengirim perintah ke perangkat pengguna dan melakukan tindakan berbahaya. Dengan demikian, pengembang jahat memanfaatkan peluang ini untuk menyisipkan *malware trojan downloader* pada aplikasi android baik dalam bidang kesehatan, pendidikan, komunikasi ataupun bidang lainnya. Berdasarkan data dari Kaspersky, *Malware trojan downloader* mengalami peningkatan 0,38 % dari rentang Q2 ke Q3 pada tahun 2020 [2]. Oleh karena itu, melakukan analisis terhadap *malware* adalah suatu tindakan untuk menentukan karakteristik dan perilaku *malware*. Sehingga ketika data karakteristik dan perilaku *malware* telah dikenal, maka memudahkan dalam menentukan langkah-langkah pencegahan terhadap serangan malware tersebut.

Building of Informatics, Technology and Science (BITS)
Volume 4, No 1, Juni 2022 Page: 69−79
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v4i1.1515

Beberapa penelitian sebelumnya tercatat melakukan *research* tentang malware dilakukan oleh Anandika Nur Imam yang berjudul *"Analisis Malware Pada Sistem Operasi Android Menggunakan Permission Based"* yang dilakukan pada tahun 2019. Penelitian ini melakukan pengecekan hak akses perizinan *10 sample malware* mulai dari *permissions internet, access network state, write external storage, read phone state, dan access wifi state* [3]. Penelitian lain yang dilakukan oleh Aldy Putra Aldya yang berjudul "*Reverse Engineering* untuk Analisis *Malware Remote Access Trojan*" dilakukan pada tahun 2019. Penelitian ini mengidentifikasi bahwa *malware Flawed Ammyy RAT* melakukan koneksi dengan jaringan eksternal dengan *ip address 172.217.16.174* [4]. "*Malware Analysis and Detection Using Reverse Engineering Technique*" yang dilakukan pada tahun 2018 oleh Selvy Megira menunjukan bahwa *malware best.exe* merupakan *Malware Virus Gen: Variant.Razy* dengan ukuran file 626 KB yang dapat menyembunyikan jejak setelah download, mengetahui nama komputer yang aktif, membuat waktu *mode sleep* menjadi lama, serta merekam *history browsing* [5]. Penelitian dengan judul *"Analysis Malware Flawed Ammyy RAT Dengan Metode Reverse Engineering"* yang dilakukan pada tahun 2018 oleh Tesa Pajar Setia menunjukan bahwa *Malware Flawed Ammyy RAT* bekerja dengan bersembunyi pada aplikasi *Ammyy Admin* kemudian melakukan koneksi dengan *attacker* dengan *ip address* 103.208.86.69. *netname ip address* 103.208.86.69 adalah *zappie host.* Perubahan 50 *registry* yang dilakukan *malware* pada sistem yang terinfeksi [6].

Teknik *reverse engineering* akan penulis gunakan untuk menganalisa *malicious software trojan downloader* pada studi kasus kamus kesehatan v2.apk. Teknik dari penelitian yang akan dilakukan harapannya berkontribusi dalam dunia keamanan, khususnya analisis malicious software. Hasil dari penelitian yang dilakukan juga membantu pengguna smartphone android untuk lebih *aware* dalam menginstal aplikasi- aplikasi yang tersedia di android.

2. METODOLOGI PENELITIAN

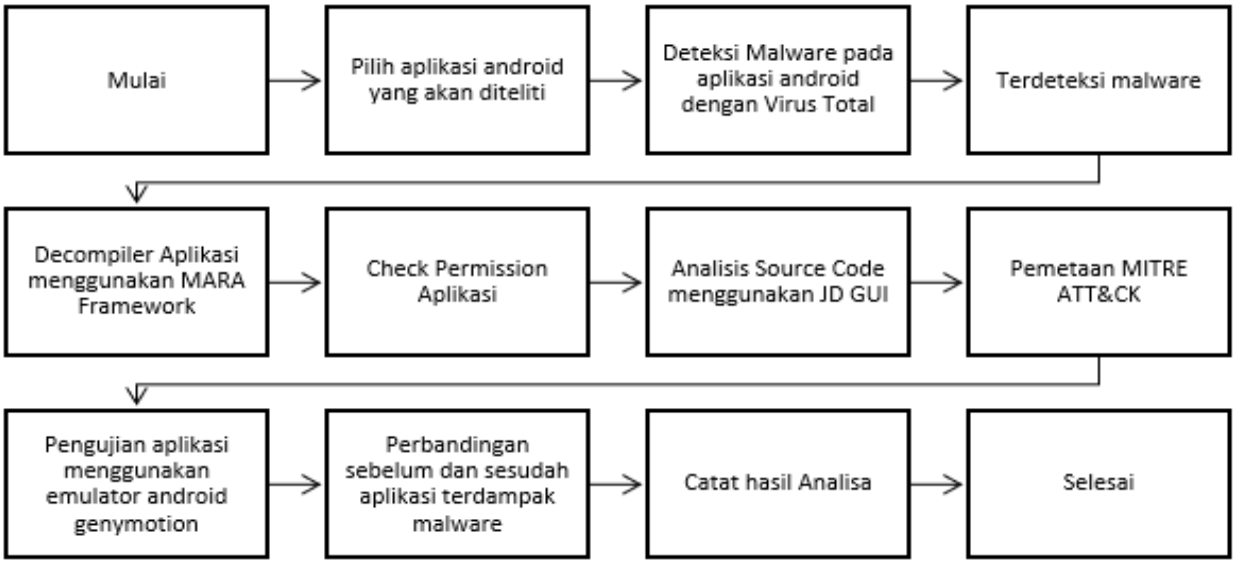
2.1 Tahapan Penelitian

Metode yang digunakan pada penelitian ini adalah sebagai berikut :

a. Pre-Experimental Design
   Desain penelitian yang dilakukan merupakan desain penelitian *Pre-Experimental Design* [10], dikarenakan tidak menjadi eksperimen secara sungguh-sungguh. Penelitian yang dilakukan ini diharapkan untuk dapat mengetahui perbedaan terhadap aplikasi android kamus kesehatan sebelum dan sesudah terinfeksi infeksi *malware trojan downloader.*
b. One Group Pretest Posttest Design
   Desain penelitian yang digunakan dalam penelitian ini adalah *"One Groups Pretest- Posttest Design",* yaitu desain penelitian yang terdapat *pretest* (sebelum) diberi perlakuan dan *posttest* (setelah) diberi treatment [10] Dengan alasan kondisi eksperimen dan karakteristik tidak semuanya dapat diatur serta dikontrol dengan ketat. Rumus One Groups Pretest- Posttest Design sebagai berikut:

$$O1 \; x \; O2 \qquad (1)$$

Keterangan:
O1 = Nilai *Pretest*, yaitu analisis aplikasi sebelum dilakukan infeksi *malware.*
X = *Treatment* (perlakuan), melakukan infeksi *malware* dengan menggunakan *metasploit framework.*
O2 = Nilai *Posttest*, yaitu analisis hasil dari aplikasi setelah dilakukan infeksi *malware*.



Gambar 1. Alur Penelitian

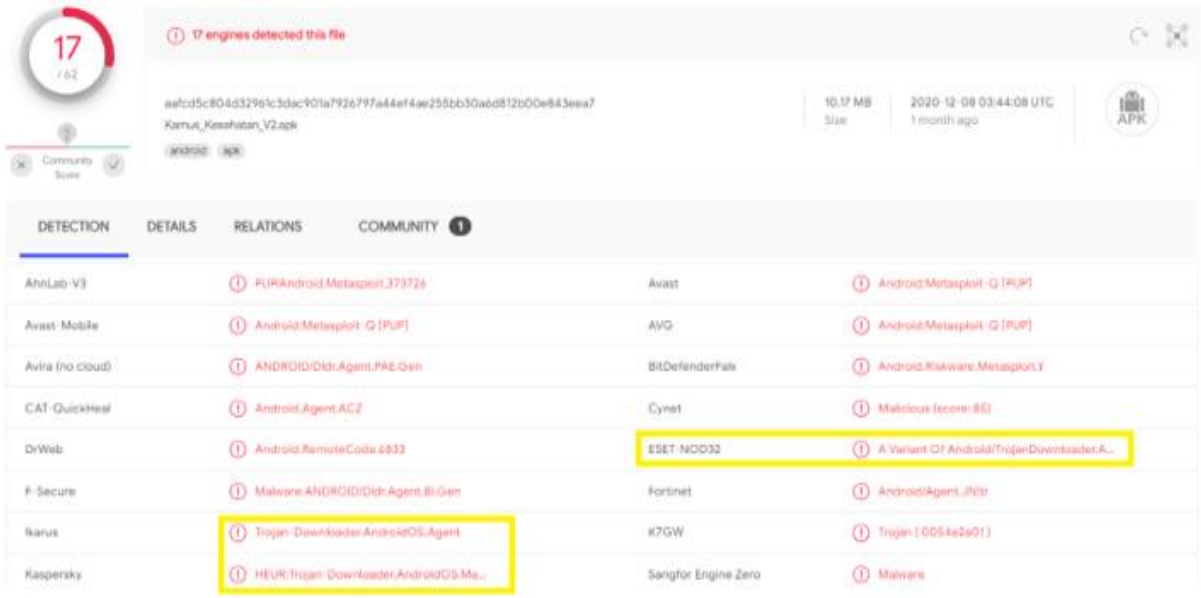
Building of Informatics, Technology and Science (BITS)
Volume 4, No 1, Juni 2022 Page: 69−79
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v4i1.1515

Alur penelitian dimulai dari pemilihan aplikasi android yang akan diteliti. *Android* merupakan sistem operasi berbasis linux untuk telepon seluler seperti *smartphone* dan tablet PC. *Android* menyediakan platform terbuka bagi para pengembang untuk menciptakan mereka sendiri yang akan digunakan oleh bermacam perangkat seluler. Sistem *android* menggunakan *database* untuk menyimpan informasi penting yang diperlukan agar tetap tersimpan meskipun device dimatikan. Untuk melakukan penyimpanan data pada *database*, sistem *android* menggunakan *SQLite* yang merupakan suatu *open- source database* yang cukup stabil dan banyak digunakan pada banyak *device* berukuran kecil [9]. Pada penelitian ini penulis memilih aplikasi "*kamus kesehatan v2.apk*" sebagai obyek penelitian kali ini. Setelah pemilihan aplikasi dilakukan, tahap selanjutnya adalah mendeteksi malware dengan virustotal. Jika aplikasi yang dipilih tidak terdeteksi malware, maka penulis akan memilih aplikasi lain yang terdeteksi adanya malware. *Malware (malicious software)* merupakan istilah umum untuk program atau kode yang berbahaya bagi sistem. *Malware* dapat menyusup ke sistem operasi sehingga dapat merusak sistem dan juga dapat mencuri file - file penting yang ada pada sistem. *Malware* diciptakan dengan maksud tertentu yaitu melakukan aktifitas berbahaya yang berdampak sangat merugikan bagi para korbannya, antara lain seperti penyadapan serta pencurian informasi pribadi [7]. Salah satu malware yang berbahaya adalah *Trojan*. *Trojan* merupakan program yang bersembunyi di dalam atau tampak seperti program sah atau legal.[15] *Trojan Downloader* merupakan *Trojan horse* yang mengunduh dan menjalankan malware lain pada sistem yang terpengaruh. *Trojan* yang memasang dirinya sendiri ke sistem dan menunggu hingga sambungan Internet tersedia untuk tersambung ke server atau situs web jarak jauh untuk mengunduh program tambahan (biasanya perangkat lunak perusak) ke komputer atau aplikasi yang terinfeksi. Jika aplikasi yang dipilih terdeteksi adanya malware maka proses penelitian dilanjutkan dengan decompile aplikasi, menggunakan MARA Framework. Proses *decompile* selesai dilanjutkan dengan checking permissions dari aplikasi "kamus kesehatan v2.apk". Setelah proses checking selesai, proses penelitian dilanjutkan dengan analisis source code, pada penelitian ini penulis menggunakan JD.GUI sebagai media/ alat untuk melakukan proses analisis source code. Pemetaan Mitre ATT&CK dilakukan setelah proses analisis source code selesai dilakukan. Proses diakhiri dengan pencatatan setelah proses pengujian aplikasi dengan emulator dan membandingkan aplikasi yang terdampak malware sebelum dan sesudah.

3. HASIL DAN PEMBAHASAN

3.1 Checksum Sample Malware

Checksum adalah item data kecil yang dihitung dari struktur data dan dapat digunakan untuk memverifikasi integritasnya, yaitu untuk memverifikasi bahwa struktur data mengalami perubahan atau tidak [11]. *Checksum* juga dikenal dengan *hash*. Sebuah file akan memiliki checksum MD5, SHA-1, dan SHA-256 yang berbeda. Pengecekan *checksum* dilakukan secara online menggunakan *virustotal.*



Gambar 2. Aplikasi terdeteksi Android Trojan Downloader

3.2 Decompiler aplikasi

Decompile adalah sebuah proses melakukan pembongkaran terhadap suatu aplikasi untuk mendapatkan *source code* agar bisa dibaca atau dipahami[12], sedangkan *decompiler* adalah alat yang digunakan dalam proses *decompile*, salah satu *tools decompiler* adalah *MARA Framework. MARA Framework* merupakan *tools* aplikasi pihak ketiga yang digunakan merekayasa balik *android binary*.

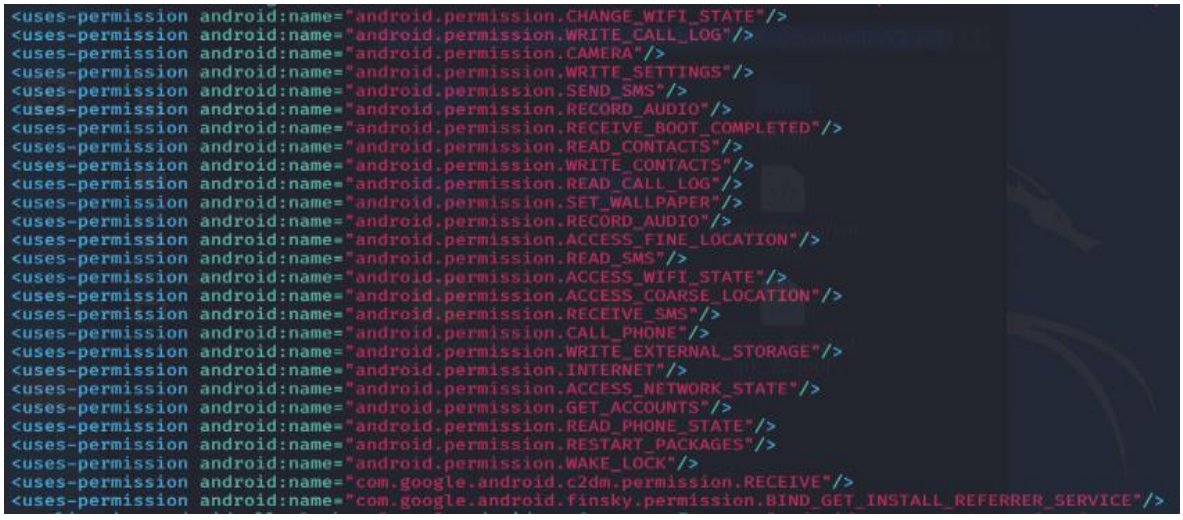
Building of Informatics, Technology and Science (BITS)
Volume 4, No 1, Juni 2022 Page: 69−79
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v4i1.1515

Gambar 3. Proses Decompiler Aplikasi Kamus Kesehatan

Decompiler malware android dengan menggunakan *MARA Framework* dengan sintak "*./mara.sh -s (decode) /home/kali/SKRIPSI/Kamus_Kesehatan_V2.apk (nama aplikasi)*. Dalam proses *decompiler*, *MARA Framework* melakukan *decoding* pada *AndroidManifest.xml*, *decoding* pada *file*-resources dan melakukan *disassembler baksmaling* pada *classes.dex* serta menyalin *assets, libs, unknown files,* dan *original files* pada APK "*Kamus Kesehatan V2".*

3.3 Application Permission Analysis

Permissions adalah sebuah hak akses yang ada pada aplikasi agar aplikasi tersebut dapat mengakses beberapa informasi dari *smartphone*. Sistem operasi Android menggunakan model berbasis izin untuk mengakses berbagai sumber daya dan informasi. Izin ini bukan permintaan melainkan deklarasi. Izin ini dideklarasikan dalam file AndroidManifest.xml. Setelah izin diberikan, izin tetap statis untuk versi Android 6.0 dan sebelumnya [17][18]. Namun, dalam versi Android, 7.0 dan diatasnya, izin aplikasi diklasifikasikan ke dalam izin normal [19] dan izin berbahaya [20]. File *android manifest* ini terletak pada *root* sebuah aplikasi.



Gambar 4. Application Permission Kamus Kesehatan V2.apk

Permission yang ada pada aplikasi ini didapatkan 27 buah *permissions.* 19 *permissions* berstatus *dangerous,* 7 *permissions* berstatus *normal* dan 1 *permission* berstatus *signature.* Informasi *permission* yang didapatkan selanjutnya dilakukan pembagian berdasarkan pada tingkat keamanan. Detail lengkap mengenai permission sample *malware trojan* pada aplikasi *"kamus kesehatan v2.apk"* dapat dilihat pada tabel 1.

Tabel 1. Penjabaran Application Permission Kamus Kesehatan V2.apk

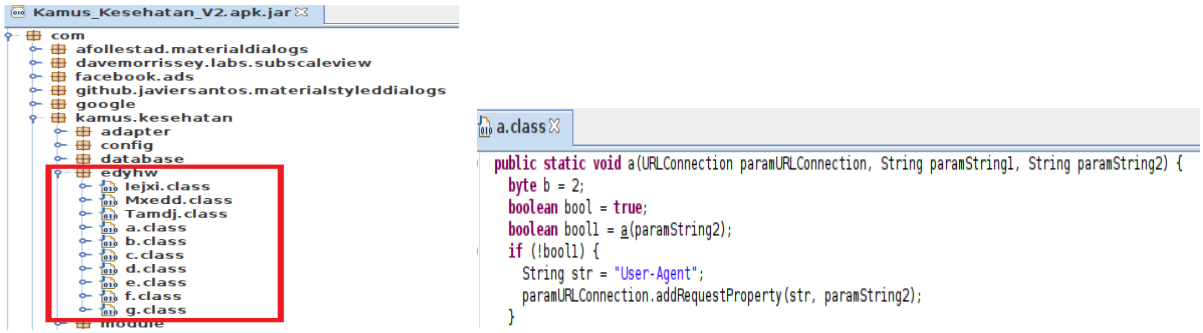
Status	Permissions
Dangerous	android.permission.ACCESS_COARSE_LOCATION
Dangerous	android.permission. ACCESS_FINE_LOCATION
Dangerous	android.permission.CALL_PHONE
Dangerous	android.permission.CAMERA
Dangerous	android.permission.CHANGE_WIFI_STATE
Dangerous	android.permission.INTERNET

Building of Informatics, Technology and Science (BITS)
Volume 4, No 1, Juni 2022 Page: 69−79
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v4i1.1515

Dangerous	android.permission.READ_CONTACTS
Dangerous	android.permission.READ_CALL_LOG
Dangerous	android.permission.WRITE_CALL_LOG
Dangerous	Android.permission.WRITE_CONTACTS
Dangerous	android.permission.READ_PHONE_STATE
Dangerous	android.permission.READ_SMS
Dangerous	android.permission.RECEIVE_SMS
Dangerous	android.permission.SEND_SMS
Dangerous	android.permission.RECORD_AUDIO
Dangerous	android.permission.WAKE_LOCK
Dangerous	android.permission.WRITE_EXTERNAL_STORAGE
Dangerous	android.permission.WRITE_SETTING
Dangerous	com.google.android.finsky.permission.BIND_GET_INSTALL_REFERRER_SERVICE
Normal	android.permission.ACCESS_NETWORK
Normal	android.permission.CHANGE_NETWORK_STATE
Normal	android.permission.GET_ACCOUNTS
Normal	android.permission.RESTART_PACKAGES
Normal	android.permission.RECEIVE_BOOT_COMPLETED
Normal	android.permission.ACCESS_WIFI_STATE
Normal	android.permission.SET_WALLPAPER
Signature	com.google.android.c2dm. permission.RECEIVE

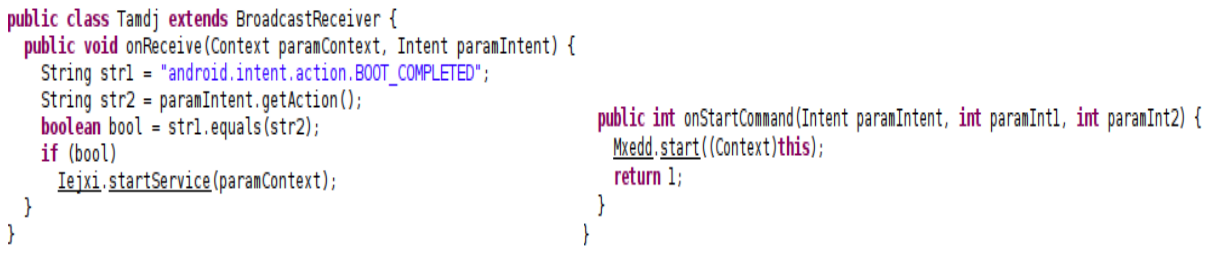
3.4 Analisis Source Code

Melakukan analisa *source code* merupakan hal penting dalam analisis *malware* dengan metode *reverse engineering*. *Reverse engineering* adalah *reverse code engineering* atau *backwards engineering* atau juga *back engineering*. Teknik ini digunakan untuk memeriksa fungsi program yang bertujuan untuk menghindari mekanisme keamanan. Teknik *reverse engineering* juga mampu memodifikasi sebuah program sesuai dengan keinginan *reverse engineer* [8]. Tahapan ini diperlukan dalam mencari kode yang bersifat *malicious* atau berbahaya.



Gambar 5. (a) Class yang dicurigai (b) Analisis file a.class

Pada gambar 6 (a) ditunjukan sebuah folder dengan penamaan yang membingungkan dan terlihat mencurigakan yaitu folder *edyhw*. Setelah dibuka, di dalam folder *edyhw* terdapat 8 class dengan penamaan yang mecurigakan. Pada gambar 6 (b) menjelaskan bahwa file *a.class* mengandung sebuah kode untuk mengetahui *User-Agent* dari perangkat korban dengan menggunakan tipe *string*. *User-Agent* merupakan aplikasi di sisi client yang secara otomatis mengirim informasi berupa string kepada server pada layanan web. String informasi ini bisa berisi OS yang digunakan, software vendor browsernya, versi *software*. Hal ini memungkinkan penyerang untuk mengambil alih mesin yang terkena dampak dan menjalankan perintah sewenang-wenang. Jenis serangan ini, bernama *Remote Code Execution (RCE)*. Remote Code Execution adalah sebuah celah yang membuat kita terhubung pada sebuah command atau terminal sistem operasi. Dengan celah ini kita dapat menggunakan perintah- perintah yang ada pada sistem operasi server tersebut. [16]



Gambar 7. (a) Analisis file Tamdj.class (b) Analisis file Iejxi.class

Building of Informatics, Technology and Science (BITS)
Volume 4, No 1, Juni 2022 Page: 69−79
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v4i1.1515

Pada gambar 7 (a) menjelaskan analisa pada file *Tamdj.class*. Diketahui bahwa file tersebut terdapat eksekusi *Broadcast Receivers*, yang dimana memungkinkan *malware* merespon tindakan dari aplikasi lain serta file *Tamdj.class* menjalankan *service* dari *class Iejxi*. Gambar 7 (b) menjelaskan analisa pada file *Iejxi.class* Diketahui bahwa file tersebut terdapat metode *onStartCommand()*, yang dimana metode ini akan dipanggil saat *service* dimulai dengan perintah *startService()* dan saat dimulai metode ini mengarah ke pemanggilan ke *class Mxedd.*

```
if (bool1) {
  serverSocket = new ServerSocket();
  this(i);
  socket = serverSocket.accept();
  serverSocket.close();
} else {
  socket = new Socket();
  this((String)serverSocket, i);
}
if (socket != null) {
  DataInputStream dataInputStream = new DataInputStream();
  InputStream inputStream = socket.getInputStream();
  this(inputStream);
  DataOutputStream dataOutputStream = new DataOutputStream();
  OutputStream outputStream = socket.getOutputStream();
  this(outputStream);
  Object[] arrayOfObject = h;
  a(dataInputStream, dataOutputStream, arrayOfObject);
}
```

Gambar 8. Analisis file Mxedd.class

File *Mxedd.class* dijelaskan pada gambar 10 menjelaskan analisa yang lebih advanced. Diketahui bahwa file tersebut memiliki kode untuk membuat *socket* jaringan baru. Dimana hal tersebut memungkinkan aplikasi melakukan initial akses kepada *CnC-nya*.

3.5 Pemetaan Metrik Mitre ATT&CK

Mitre Adversarial Tactics, Techniques, and Common Knowledge (ATT&CK) merupakan model yang menghitung dan mengkategorikan taktik, teknik, dan prosedur (TTP) untuk meningkatkan deteksi aktivitas program jahat. *ATT&CK* dibagi menjadi dua domain, yaitu *Enterprise (Linux, Windows, macOS)* dan *Mobile (Android, iOS)*.[13]

Mitre Att&ck Matrix

Initial Access	Execution	Persistence	Privilege Escalation	Defense Evasion	Credential Access	Discovery	Lateral Movement	Collection	Exfiltration	Command and Control	Network Effects	Remote Service Effects	Impact
Valid Accounts	Windows Management Instrumentation	Path Interception	Path Interception	Application Discovery 1	Access Stored Application Data 1	System Network Connections Discovery 1	Remote Services	Location Tracking 1	Exfiltration Over Other Network Medium	Encrypted Channel 1	Exploit SS7 to Redirect Phone Calls/SMS 2	Remotely Track Device Without Authorization	Delete Device Data 1
Default Accounts	Scheduled Task/Job	Boot or Logon Initialization Scripts	Boot or Logon Initialization Scripts	Obfuscated Files or Information 1	LSASS Memory	Location Tracking 1	Remote Desktop Protocol	Capture Audio 1	Exfiltration Over Bluetooth	Non-Standard Port 1	Eavesdrop on Insecure Network Communication 1	Remotely Wipe Data Without Authorization	Carrier Billing Fraud 1
Domain Accounts	At (Linux)	Logon Script (Windows)	Logon Script (Windows)	Obfuscated Files or Information	Security Account Manager	Application Discovery 1	SMB/Windows Admin Shares	Network Information Discovery 1	Automated Exfiltration	Non-Application Layer Protocol 1	Exploit SS7 to Track Device Location	Obtain Device Cloud Backups	Delete Device Data
Local Accounts	At (Windows)	Logon Script (Mac)	Logon Script (Mac)	Binary Padding	NTDS	Process Discovery 1	Distributed Component Object Model	Access Stored Application Data 1	Scheduled Transfer	Application Layer Protocol 2	SIM Card Swap		Carrier Billing Fraud

Gambar 9. Pemetaan Metrik Mitre ATT&CK

Penjelasan Mitre ATT&CK dapat dilihat pada table 2 dimana pada proses *Defense Evasion, Credential Access, Discovery, Collection, Command and Control, Network Effect dan Impact* sudah ter-mapping dengan masing masing ID beserta nama dan deskripsi.

Tabel 2. Penjelasan Mitre ATT&CK

Proses	ID	Nama	Deskripsi
Defense Evasion	T1418	Application Discovery	Memungkinkan aplikasi jahat untuk mengidentifikasi semua aplikasi yang diinstal pada perangkat.
	T1406	Obfuscated Files or Information	Aplikasi dapat berisi kode berbahaya dalam bentuk yang disamarkan atau dienkripsi, kemudian membatalkan penyamaran atau mendekripsi kode pada waktu proses untuk menghindari banyak teknik pemeriksaan aplikasi.
Credential Access	T1409	Access Stored Application Data	Aplikasi dapat mengakses dan mengumpulkan data aplikasi yang ada di perangkat.

Building of Informatics, Technology and Science (BITS)
Volume 4, No 1, Juni 2022 Page: 69−79
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v4i1.1515

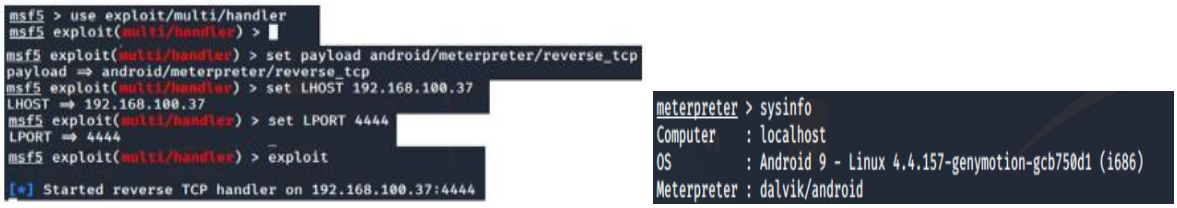
Discovery	T1421	System Network Connections Discovery	Di Android, aplikasi dapat menggunakan API standar untuk mengumpulkan daftar koneksi jaringan ke dan dari perangkat.
	T1430	Location Tracking	Penyerang dapat menggunakan aplikasi yang berbahaya atau dieksploitasi untuk secara diam-diam melacak lokasi fisik perangkat melalui penggunaan API sistem operasi standar.
	T1418	Application Discovery	Memungkinkan aplikasi jahat untuk mengidentifikasi semua aplikasi yang diinstal pada perangkat.
	T1424	Process Discovery	Pada versi Android sebelum 5, aplikasi bisa mengamati informasi tentang proses lain yang sedang berjalan melalui metode di kelas ActivityManager. Pada versi Android sebelum 7, aplikasi bisa mendapatkan informasi ini dengan menjalankan perintah ps, atau dengan memeriksa direktori / proc.
Collection	T1430	Location Tracking	Penyerang dapat menggunakan aplikasi yang berbahaya atau dieksploitasi untuk secara diam-diam melacak lokasi fisik perangkat melalui penggunaan API sistem operasi standar.
	T1429	Capture Audio	Penyerang dapat menangkap audio untuk mengumpulkan informasi tentang pengguna perangkat seluler menggunakan API sistem operasi standar. Penyerang dapat menargetkan informasi audio seperti percakapan pengguna, lingkungan sekitar, panggilan telepon, atau informasi sensitif lainnya.
	T1507	Network Information Discovery	Penyerang dapat menggunakan sensor perangkat untuk mengumpulkan informasi tentang jaringan terdekat, seperti Wi-Fi dan Bluetooth.
	T1409	Access Stored Application Data	Aplikasi dapat mengakses dan mengumpulkan data aplikasi yang ada di perangkat.
Command and Control	T1573	Encrypted Channel	Penyeran dapat menggunakan algoritma enkripsi yang dikenal untuk menyembunyikan perintah dan mengontrol lalu lintas daripada mengandalkan perlindungan bawaan yang disediakan oleh protokol komunikasi.
	T1571	Non-Standard Port	Penyerang dapat berkomunikasi menggunakan protokol dan port paring yang biasanya tidak terkait. Misalnya, HTTPS melalui port 8088 atau port 587 sebagai lawan dari port tradisional 443.
	T1095	Non-Application Layer Protocol	Penyerang dapat menggunakan protokol lapisan non-aplikasi untuk komunikasi antara host dan server C2 atau di antara host yang terinfeksi dalam jaringan.
	T1071	Application Layer Protocol	Penyerang dapat berkomunikasi menggunakan protokol lapisan aplikasi untuk menghindari deteksi / pemfilteran jaringan dengan membaur dengan lalu lintas yang ada.
Network Effects	T1449	Exploit SS7 to Redirect Phone Calls/SMS2	Penyerang dapat memanfaatkan kerentanan sistem pensinyalan untuk mengalihkan panggilan atau pesan teks (SMS) ke nomor telepon di bawah kendali penyerang kemudian dapat bertindak sebagai man-in-the-middle untuk mencegat atau memanipulasi komunikasi.
	T1439	Eavesdrop on insecure Network Communic-ation	Jika lalu lintas jaringan antara perangkat seluler dan server jarak jauh tidak dienkripsi atau dienkripsi dengan cara yang tidak aman, maka penyerang dapat menguping komunikasi.
Impact	T1447	Delete Device Data	Penyerang dapat menghapus perangkat atau menghapus file individu untuk memanipulasi hasil eksternal atau menyembunyikan aktivitas.
	T1448	Carrier Billing Fraud	Aplikasi berbahaya dapat memicu biaya penipuan pada laporan tagihan operator korban dengan beberapa cara berbeda, termasuk penipuan pulsa SMS.

3.6 Pengujian

a. Demonstrasi Attack Devices

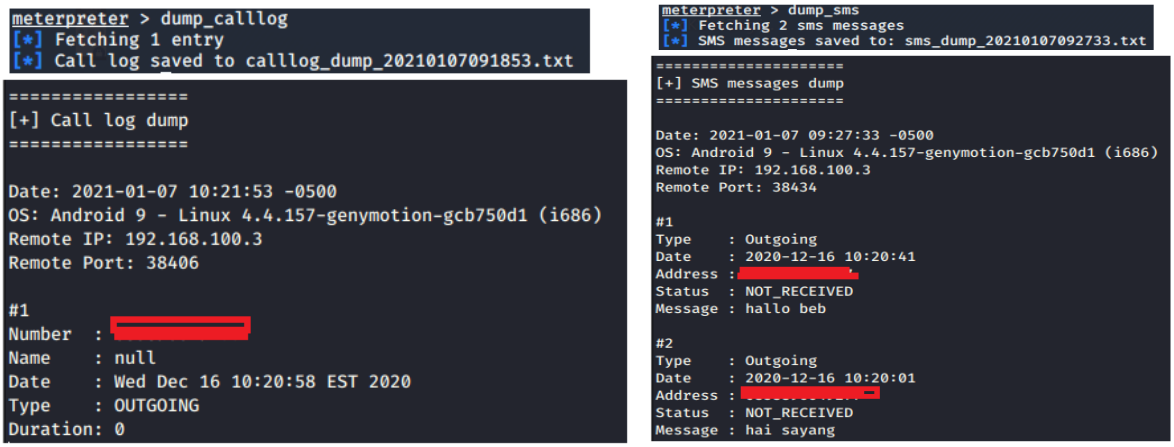
Penyerang melakukan serangan dengan menjalankan *payload* pada *tools metasploit. Payload* adalah kode yang kita inginkan agar system dieksekusi dan itu harus dipilih dan disampaikan oleh Framework [14]. *Payload* yang digunakan oleh penyerang adalah *reverse shell*, yaitu *payload* yang bertujuan membuat koneksi dari device target dan kembali kembali pada penyerang menjadi *Meterpreter console.*

Building of Informatics, Technology and Science (BITS)
Volume 4, No 1, Juni 2022 Page: 69−79
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v4i1.1515



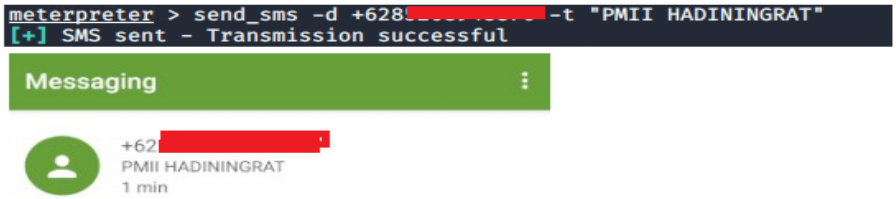
Gambar 10. (a) Proses exploitasi (b) Command Sysinfo

Proses exploitasi yang berakibat penyerang dapat melakukan *remote access* ke *devices* korban disajikan pada gambar 10 (a). Tahapan dimulai dengan perintah *"use exploit/multi/handler"* perintah ini digunakan sebagai rintisan yang menangani exploitasi yang dijalankan di luar *framework.* Proses dilanjutkan dengan *"set payload android/ meterpreter /reverse_tcp"* sebagai perintah untuk menjalankan *payload* pada aplikasi melalui meterpreter *reverse tcp*.Perintah selanjutnya adalah *"set LHOST 192.168.100.37"*, perintah ini difungsikan sebagai IP yang digunakan remote device korban. Dan dilanjutkan dengan perintah *"set LPORT 4444"* untuk difungsikan sebagai port yang digunakan *remote device* korban. Perintah *Exploit adalah perintah terakhir* untuk melakukan exploitasi device. Penyerang sudah mendapatkan hak akses *(Priviliges)* penuh terhadap *device* korban setelah perintah *exploit* sehingga penyerang leluasa melakukan perintah – perintah yang diinginkan, seperti perintah *"sysinfo"* yang digunakan untuk melihat informasi *operating system* pada device korban.



Gambar 11. (a) *Command dump_call log* (b) *Command dump_sms*

Perintah lain juga bisa di kirim ke target seperti perintah *"ifconfig"* untuk melihat informasi jaringan pada target host. Perintah *"app_list"* untuk melihat aplikasi apa saja yang terinstall di target host. Perintah *"play"* untuk memutar musik di target host. Penulis juga memberikan perintah *"screenshot"* untuk menangkap tangkapan layar di target host. Selain beberapa perintah sebelumnya yang bisa penulis input. Penulis juga dapat memberikan perintah *"dump_calllog"* untuk mengambil log panggilan dari perangkat korban. Setelah perintah *dump_calllog* dijalankan terlihat ada 1 entri atau data yang telah didapatkan, kemudian data tersebut tersimpan dalam file *calllog_dump_20210107091853.txt* yang dapat diakses dengan menggunakan tool text editor. Hasil dari file *calllog_dump_20210107091853.txt.* Perintah *"dump_sms"* untuk mengambil pesan SMS dari perangkat korban. Setelah perintah *dump_sms* dilakukan terlihat 2 pesan SMS berhasil diambil (Gambar 11 (b)) dan disimpan pada file bernama *sms_dump_20210107092733.txt* dan untuk melihat pesan SMS yang berhasil diambil sehingga perlu mengakses file *sms_dump_20210107092733.txt* menggunakan text editor.



Gambar 12. Command Send_sms

Perintah *"send_sms"* untuk melakukan pengiriman sms ke perangkat *devices.* Perintah *send_sms* dilakukan dengan format *send_sms – d +6287738466883 –t "PMII HADININGRAT"* yang diartikan sebagai berikut :

1. *send_sms* : Merupakan perintah untuk mengirim pesan SMS.
2. *–d* : Merupakan kode untuk *destination number.*
3. *+628xxx* : Merupakan nomor tujuan.

Building of Informatics, Technology and Science (BITS)
Volume 4, No 1, Juni 2022 Page: 69−79
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v4i1.1515

4. *–t* : Merupakan kode untuk body SMS atau isi dari pesan.
5. *"PMII HADININGRAT"* : Merupakan isi dari pesan, serta isi berada dalam kedua tanda kutip

b. Demonstrasi One-Group Pretest dan Posttest

Peneliti menguji perbandingan aplikasi *Kamus Kesehatan V2.apk* (yang *ter-embedded malware*) dengan aplikasi *Kamus Kesehatan dan Medis Offline.apk (aplikasi original)*. Pengujian dilakukan dengan menggunakan *tools MobSF (Mobile Security Framework)*, dimana pengujian tersebut meliputi 5 tahapan: *1) hashing, 2) permissions, 3) signer certificate, 4) services, 5) receivers.*

Gambar 13. (a) Perbandingan permission (b) Perbandingan Hashing

Gambar 6. (a) Perbedaan signer certificate (b) Services (c) Receivers

Building of Informatics, Technology and Science (BITS)
Volume 4, No 1, Juni 2022 Page: 69−79
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v4i1.1515

Peneliti menemukan beberapa perbedaan dari aplikasi sebelum dan sesudah *ter-embedded malware*. Berikut penjelasan dari setiap perbedaan yang dapat dilihat pada tabel 3.

Tabel 3. Penjelasan Perbandingan aplikasi (One-Group Pretest Posttest)

Aspect	Application	
	Pretest	**Posttest**
Hashing	34c64361210e7f911008d32979 88e3d7e1e06b4f3d6d505474a2 aab3250621f9a	aafcd5c804d32961c3dac901a792 6797a44ef4ae255bb30a6d812b00e 843eea7
Permissions	9 Permissions	27 Permissions
Signer Certificate	1181a4b40dfbfd05e4ce9d4d42a5 3f1d30466c315c185fabc9d80494 8d7cf3e9	Tidak terdapat *fingerprint*
Services	5 Services	6 Services
Receivers	3 Receirves	4 Receirves

4. KESIMPULAN

Hasil penelitian ini ditunjukkan dengan melihat dari beberapa sudut pandang perspektif analisis dari perbandingan analisis aplikasi kamus kesehatan v2 sebelum dan sesudah terinfeksi *malware trojan downloader*. Dari analisis statis menggunakan *MobSF* ditemukan perbedaan ukuran file pada aplikasi kamus kesehatan v2. Dimana setelah *ter-embedded malware*, ukuran file menjadi 10.17 MB yang sebelumnya adalah 10.05 MB. Tentunya dengan perubahan ukuran file, maka secara otomatis juga perubahan pada *hashing SHA256*. *MobSF* menemukan adanya perubahan pada *permissions* yang sebelumnya hanya ada 9 *permissions*, namun setelah *ter-embedded malware* ditemukan 18 *permissions* tambahan sehingga total keseluruhan menjadi 27 *permissions. MobSF* menemukan penghapusan *fingerprint* pada aplikasi kamus kesehatan yang terinfeksi *malware trojan downloader*. *MobSF* menemukan adanya perubahan pada *services* yang sebelumnya hanya ada 5 *services*, namun setelah terinfeksi *malware trojan downloader* ditemukan 6 *services.* Terakhir *MobSF* menemukan adanya perubahan pada *receivers* yang sebelumnya hanya ada 3 *receivers*, namun setelah terinfeksi *malware trojan downloader* ditemukan 4 *receivers.* Implementasi teknik analisis dinamis juga berhasil dilakukan dengan uji hasil kerja *malware trojan downloader*, yaitu berhasil menjalankan 8 perintah seperti *sysinfo, ifconfig, app_list, play, screenshot, dump_calllog, dump_sms, dan send_sms.* Peneliti menyadari sepenuhnya bahwa *analisis malware trojan downloader* pada android menggunakan teknik *reverse engineering* ini masih memiliki kekurangan, oleh karena itu saran yang dapat peneliti berikan yaitu *Malware* adalah topik penelitian yang masih sangat terbuka luas. Selain memanfaatkan *reverse engineering*, peneliti selanjutnya dapat dilakukan pula menggunakan *signature base detection, behaviour based, deep learning.*

REFERENCES

[1] G. Statcounter, 'Mobile Operating System Market Share Worldwide', 2020. [Online]. Available: Https://Gs.Statcounter.Com/Os-Market-Share/All/Worldwide/2020
[2] Ao Kaspersky Lab, 'It Threat Evolution Q3 2020 Mobile Statistics', Securelist, 2020. [Online]. Available: Https://Securelist.Com/It-Threat-Evolution-Q3-2020-Mobile-Statistics/99461/
[3] A. N. Iman And A. Budiyono, 'Analisis Malware Pada Sistem Operasi Android Menggunakan Permission-Based', P. 7.
[4] T. P. Setia, A. P. Aldya, And N. Widiyasono, 'Reverse Engineering Untuk Analisis Malware Remote Access Trojan', J. Edukasi Dan Penelit. Inform. Jepin, Vol. 5, No. 1, P. 40, Apr. 2019, Doi: 10.26418/Jp.V5i1.28214.
[5] S. Megira, A. R. Pangesti, And F. W. Wibowo, 'Malware Analysis And Detection Using Reverse Engineering Technique', J. Phys. Conf. Ser., Vol. 1140, P. 012042, Dec. 2018, Doi: 10.1088/1742-6596/1140/1/012042.
[6] T. Pajar Setia, N. Widiyasono, And A. Putra Aldya, 'Analysis Malware Flawed Ammyy Rat Dengan Metode Reverse Engineering', J. Inform. J. Pengemb. It, Vol. 3, No. 3, Pp. 371–379, Oct. 2018, Doi: 10.30591/Jpit.V3i3.1019.
[7] T. A. Cahyanto, V. Wahanggara, And D. Ramadana, 'Analisis Dan Deteksi Malware Menggunakan Metode Malware Analisis Dinamis Dan Malware Analisis Statis', Vol. 2, No. 1, P. 12, 2017.
[8] S. M. Myat And M. T. Kyaw, 'Analysis Of Android Applications By Using Reverse Engineering Techniques', Vol. 4, No. 3, P. 8, 2019.
[9] G. Developers, 'What Is Android', 2021. [Online]. Available: Https://Www.Android.Com/What-Is-Android/
[10] Sugiyono, Metode Penelitian Pendidikan Pendekatan Kuantitatif, Kualitatif, Dan R&D. Bandung: Alfabeta, 2017.
[11] F. Corella And K. Lewison, 'An Omission-Tolerant Cryptographic Checksum', P. 30.
[12] Yumnun, L., Kusyanti, A., & Kartikasari, D. (2020). Implementasi Owasp Mobile Security Testing Guide (Mstg) Untuk Pengujian Keamanan Pada Aplikasi Berbasis Android. Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer, 3, 10579–10585.
[13] Blake E. Strom Et Al. 2018. Mitre Att&Cktm: Design And Philosophy. Mclean, Va. [Online] Available: Https://Www.Mitre.Org/Sites/Default/Files/Publications/Pr-18-0944-11-Mitre-Attack-Design-And-Philosophy.Pdf
[14] Dwiananda, Rizky & Mardianto, Is. (2019). Exploitation With Reverse_Tcp Method On Android Device Using Metasploit. Jurnal Edukasi Dan Penelitian Informatika (Jepin). 5. 106. 10.26418/Jp.V5i1.26893.

Building of Informatics, Technology and Science (BITS)
Volume 4, No 1, Juni 2022 Page: 69−79
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)
DOI 10.47065/bits.v4i1.1515

[15] Prathivi, Rastri & Vydia, Vensy. (2017). Analisa Pendeteksian Worm Dan Trojan Pada Jaringan Internet Universitas Semarang Menggunakan Metode Kalsifikasi Pada Data Mining C45 Dan Bayesian Network. Jurnal Transformatika. 14. 77. 10.26623/Transformatika.V14i2.440.

[16] Yantu, Ramdan. (2014). Tutorial Celah Keamanan Pada Php Scripts. Available: Https://Dl.Packetstormsecurity.Net/Papers/General/Phpbugs-Tutorial.Pdf (Diakses Pada Tanggal 15 Februari 2022).

[17] L. Whitney, "Almost No One Is Using Android Marshmallow, Still," Cnet, 2016. [Online]. Available: Http://Www.Cnet.Com/News/Almost-No- One-Is-Using-Androidmarshmallow-Still. (Diakses Pada Tanggal 11 Februari 2022).

[18] V. Savov, "Only 7.5 Percent Of Android Phones Are Running Marshmallow," The Verge, 2016. [Online]. Available: Http://Www.Theverge.Com/Circuitbreaker/2016/5/4/1 1589630/ Android-6-Marshmallow-Os-Distribution- Statistics. (Diakses Pada Tanggal 15 Februari 2022).

[19] Normal Permissions,". [Online]. Available: Https://Developer.Android.Com/Guide/Topics/Security /Normalpermissions. Html (Diakses Pada Tanggal 10 Januari 2022).

[20] Dangerous Permissions,". [Online]. Available: Https://Developer.Android.Com/Guide/Topics/Security /Permissions.Html#Normal-Dangerous. (Diakses Pada Tanggal 14 Desember 2021).