Analisis *Malware Trojan* Dalam *File* Undangan Pernikahan.Apk Pada *Smartphone* Android Dengan Metode *Hybrid Analysis*

Guntur Setya Agung
Teknik Informatika
Informatics and Computer Science Group
Purwokerto, Indonesia
guntursetyaagung@student.telkomuniversity.ac.id

Trihastuti Yuniati
Teknik Informatika
Informatics and Computer Science Group
Purwokerto, Indonesia
trihastutiy@telkomuniversity.ac.id

Ipam Fuaddina Adam
Teknik Informatika
Informatics and Computer Science Group
Purwokerto, Indonesia
ipamya@telkomuniversity.ac.id

***Abstrak* — Penggunaan smartphone Android yang mendominasi pasar Indonesia pada 2024 mencapai 88,46%, meningkatkan risiko keamanan siber, dengan lebih dari 98,3% responden melaporkan pernah menerima pesan penipuan digital, terutama melalui tautan berisi *malware* (65,2%). Penelitian ini bertujuan menganalisis *malware trojan* yang menyamar sebagai aplikasi undangan pernikahan, menggunakan metode *hybrid analysis* yang menggabungkan analisis statis dan dinamis. Analisis statis mempelajari struktur dan izin aplikasi, sedangkan analisis dinamis mengamati perilaku *malware* pada perangkat Android. Hasil penelitian menunjukkan bahwa *malware* ini dapat mencuri data pribadi, seperti pesan SMS dan informasi perangkat, serta mengirim data tersebut ke server eksternal melalui API Telegram. *Malware* ini juga dapat mengirim SMS tanpa sepengetahuan pengguna, berpotensi menimbulkan penyalahgunaan lebih lanjut.**

Kata kunci*— *Android, Hybrid Analysis, Malware, Penipuan Digital

I. PENDAHULUAN

Perangkat smartphone Android mendominasi pasar Indonesia dengan pangsa pasar mencapai 88,46% pada tahun 2024[1], memberikan kemudahan dalam komunikasi dan akses layanan. Namun, dengan meningkatnya penggunaan *smartphone*, risiko keamanan siber juga semakin besar, terutama dalam hal ancaman *malware* yang dapat mengakses data pribadi dan informasi sensitif pengguna[2]. Berdasarkan survei nasional, lebih dari 98,3% responden pernah menerima pesan penipuan digital, dengan salah satu modus yang paling banyak ditemukan adalah pengiriman tautan berisi *malware*[3]. *Malware* ini sering tersebar melalui aplikasi komunikasi populer seperti WhatsApp, yang digunakan oleh 90,9% pengguna internet di Indonesia[4]. Kejahatan siber ini menimbulkan tantangan besar dalam melindungi pengguna *smartphone*, terutama dalam mendeteksi dan menganalisis ancaman *malware* yang semakin kompleks.

Malware yang menyamar sebagai aplikasi undangan pernikahan menjadi salah satu bentuk penipuan yang banyak beredar melalui pesan WhatsApp[5]. Dalam penelitian ini, digunakan metode *hybrid analysis* yang menggabungkan analisis statis dan dinamis untuk memahami perilaku *malware* dan potensi kerentanannya. Penelitian ini bertujuan untuk mengetahui dampak serangan terhadap sistem Android dan mengidentifikasi jenis data sensitif yang dapat dicuri oleh penyerang. Hasil penelitian ini diharapkan dapat memberikan informasi yang lebih mendalam mengenai deteksi dan mitigasi ancaman siber pada perangkat android.

II. KAJIAN TEORI

A. APKTool

APKTool adalah alat sumber terbuka yang digunakan untuk mendekompilasi dan merekompilasi file APK, memungkinkan pengguna untuk menganalisis dan memodifikasi aplikasi Android. Dengan APKTool, pengguna dapat mengakses kode sumber, sumber daya, dan berkas manifest aplikasi, serta menganalisis elemen-elemen seperti gambar dan layout XML. Alat ini sangat berguna bagi pengembang dan peneliti keamanan untuk memahami, menganalisis, dan melakukan modifikasi aplikasi Android[6].

B. *Hybrid analysis*

Hybrid analysis adalah metode yang menggabungkan analisis statis dan dinamis untuk meningkatkan deteksi ancaman. Analisis statis memeriksa kode tanpa menjalankannya, mengidentifikasi struktur dan kerentanannya, sementara analisis dinamis mengamati perilaku perangkat lunak saat dijalankan dalam lingkungan terkendali, seperti interaksi dengan sistem dan akses jaringan. Kombinasi kedua metode ini mengatasi keterbatasan masing-masing, memberikan efisiensi dan keamanan dalam memeriksa kode, serta memungkinkan pengamatan langsung terhadap perilaku berbahaya atau mencurigakan, sehingga meningkatkan kemampuan deteksi ancaman[7].

C. JADX

JADX adalah alat dekompilasi yang mengonversi berkas APK dan DEX Android menjadi kode sumber Java yang lebih mudah dipahami. Alat ini membantu pengembang, peneliti keamanan, dan analis untuk memahami struktur dan fungsi aplikasi Android dengan menyajikan kode biner dalam format yang lebih manusiawi. Menyediakan antarmuka grafis (GUI) dan baris perintah (CLI), JADX memungkinkan pengguna untuk menavigasi kode yang didekompilasi, menyorot sintaks, dan mencari teks tertentu dalam kode. Alat ini berguna dalam debugging, analisis keamanan, dan pembelajaran pengembangan aplikasi Android[8].

D. *Malware*

Malware adalah perangkat lunak yang dirancang untuk merusak atau mengompromikan sistem komputer tanpa izin pengguna, sering digunakan oleh penjahat siber untuk mencuri informasi sensitif atau memperoleh keuntungan. Jenis-jenis *malware* yang umum antara lain virus, worm, Trojan, spyware, adware, dan ransomware. Virus dapat mereplikasi diri dan menyebar ke sistem lain dan mengganggu operasional sistem. Worm menyebar sendiri tanpa bantuan pengguna dan memanfaatkan celah keamanan untuk mengganggu sistem. Trojan menyamar sebagai file normal namun berisi kode jahat yang sering digunakan untuk mencuri data atau merusak sistem. Spyware mengumpulkan informasi pribadi korban tanpa izin, sementara adware menampilkan iklan otomatis yang bisa menyertakan spyware untuk melacak aktivitas pengguna. Ransomware mengenkripsi data dan meminta tebusan untuk mengembalikannya, sering menyerang bisnis dan individu dengan ancaman kehilangan data penting jika tebusan tidak dibayar[9].

E. Mobile Security Framework (MobSF)

Mobile Security Framework (MobSF) adalah platform riset yang fokus pada keamanan aplikasi mobile untuk sistem operasi Android, iOS, dan Windows Mobile. MobSF menyediakan analisis statis dan dinamis untuk kebutuhan keamanan, termasuk analisis aplikasi, pengujian penetrasi, analisis *malware*, dan analisis privasi. Platform ini juga mendukung integrasi dengan alur kerja DevSecOps dan pipeline CI/CD melalui REST API dan Command Line Interface (CLI), memungkinkan pemeriksaan keamanan secara otomatis dalam proses pengembangan aplikasi. Dengan fitur lengkapnya, MobSF membantu pengembang dan peneliti dalam mengidentifikasi, menganalisis, dan memperbaiki kelemahan keamanan aplikasi mobile[10].

F. Telegram

Telegram adalah platform pesan instan yang menawarkan komunikasi aman melalui enkripsi end-to-end. Fitur utamanya mencakup grup chat besar, pesan terjadwal, channel publik, dan bot untuk otomatisasi tugas. Telegram juga menyediakan API bot yang memungkinkan pengembang membuat bot khusus untuk berbagai keperluan, seperti layanan pelanggan atau pemberitahuan. Dengan fitur-fitur ini, Telegram menjadi ekosistem kuat untuk komunikasi personal dan bisnis, serta mendukung pengembangan aplikasi dan solusi melalui API bot yang lengkap dan efisien[11].

G. VirusTotal

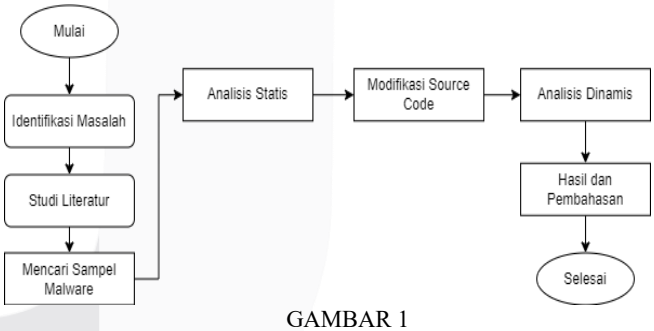
VirusTotal adalah platform analisis online yang memungkinkan pengguna memeriksa berkas dan URL terhadap ancaman *malware* dan masalah keamanan lainnya. Dengan mengunggah berkas atau URL, VirusTotal menggunakan berbagai mesin pemindaian dari vendor antivirus terkemuka untuk memberikan laporan yang menunjukkan apakah berkas atau URL tersebut terinfeksi. Dengan memanfaatkan berbagai mesin pemindaian dan kontribusi komunitas, VirusTotal meningkatkan kemampuan deteksi ancaman keamanan secara lebih efektif[12].

H. Whatsapp

WhatsApp adalah aplikasi komunikasi yang memanfaatkan internet untuk memungkinkan pengguna mengirim pesan teks, suara, gambar, video, dokumen, serta melakukan panggilan suara dan video. Aplikasi ini mengandalkan teknologi Voice over Internet Protocol (VoIP) untuk panggilan suara dan video, serta Signal Protocol untuk enkripsi end-to-end yang menjaga keamanan dan privasi pengguna. WhatsApp berfungsi melalui koneksi internet, baik Wi-Fi atau data seluler, memungkinkan komunikasi antarnegara tanpa biaya tambahan selain penggunaan data[13].

III. METODE

Penelitian ini dilakukan secara terstruktur dan sistematis melalui serangkaian tahapan yang jelas. Alur dari tahapan penelitian dapat dilihat pada gambar 1.



GAMBAR 1
(ALUR PENELITIAN)

Proses diawali dengan identifikasi masalah yang akan diteliti, kemudian dilanjutkan dengan studi literatur untuk memperoleh informasi yang relevan. Langkah berikutnya adalah mencari dan mengumpulkan sampel *malware* yang sesuai untuk dianalisis lebih lanjut.

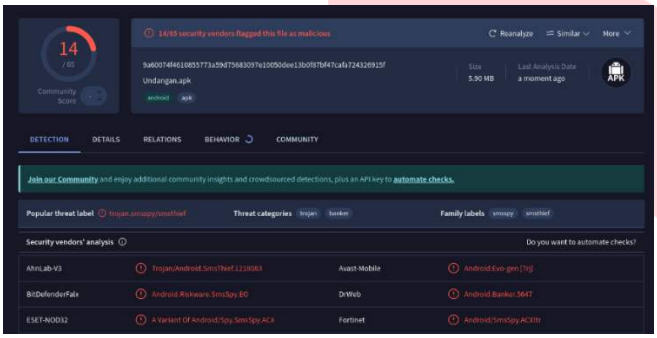
Setelah memperoleh sampel *malware*, penelitian ini menggunakan metode hybrid analysis, yang menggabungkan analisis statis dan dinamis untuk mendapatkan pemahaman yang lebih komprehensif terhadap karakteristik dan perilaku *malware*. Pada tahap analisis statis, *malware* terlebih dahulu dipindai menggunakan VirusTotal untuk mendeteksi keberadaannya. Selanjutnya, *malware* didekompilasi menggunakan JADX untuk menganalisis kode sumber dan izin guna memahami struktur serta fungsinya. Analisis tambahan dilakukan menggunakan MobSF untuk mengevaluasi keamanan aplikasi secara menyeluruh. Selain itu, *malware* juga didekompilasi menggunakan APKTool, di

mana dilakukan modifikasi pada kode sumbernya sebelum berlanjut ke tahap analisis dinamis.

Tahap analisis dinamis dilakukan dengan menginstal *malware* pada perangkat Android dalam lingkungan terisolasi guna mengamati interaksi dan perilakunya secara langsung saat dijalankan. Selama proses ini, aktivitas jaringan, perubahan sistem, serta komunikasi dengan server eksternal dipantau untuk mengidentifikasi pola-pola mencurigakan.

Hasil dari semua tahapan penelitian dianalisis secara menyeluruh dan didokumentasikan dalam bentuk laporan. Laporan ini mencakup bukti-bukti yang ditemukan selama penelitian, serta temuan dan rekomendasi yang dihasilkan dari seluruh proses analisis.

IV. HASIL DAN PEMBAHASAN



GAMBAR 2
(PEMINDAIAN MENGGUNAKAN VIRUSTOTAL)

Berdasarkan informasi yang diperoleh dari VirusTotal, sampel *malware* dengan nama Undangan.apk memiliki checksum SHA256 "9a60074f4610855773a59d75683097e10050dee13b0f87bf47cafa724326915f". Dari 65 perangkat lunak *antimalware* yang menganalisisnya, 14 di antaranya mengidentifikasi file ini sebagai ancaman berbahaya. *Malware* ini dikategorikan sebagai Trojan dengan tipe SMS Spy dan Banking Trojan, yang bertujuan untuk mencuri informasi sensitif, seperti pesan teks dan data kredensial perbankan.

APPLICATION PERMISSIONS

PERMISSION	STATUS	INFO
android.permission.INTERNET	normal	full Internet access
android.permission.RECEIVE_SMS	dangerous	receive SMS
android.permission.SEND_SMS	dangerous	send SMS messages
DYNAMIC_RECEIVER_NOT_EXPORTED_PERMISSION0wleuqgbs	unknown	Unknown permission

GAMBAR 3
(*PERMISSION* DARI UNDANGAN.APK)

Hasil analisis *permission* menggunakan JADX dan MobSF, terdapat 4 izin yang ditemukan pada aplikasi. Izin pertama adalah *android.permission.INTERNET*, yang digunakan untuk mengakses jaringan internet dan membuat koneksi jaringan. Izin ini memiliki status Normal dan umumnya tidak berbahaya. Selanjutnya, terdapat *android.permission.RECEIVE_SMS*, yang memiliki status Dangerous karena memungkinkan aplikasi menerima dan memproses pesan SMS, dengan potensi penyalahgunaan seperti memantau atau menghapus pesan tanpa sepengetahuan pengguna. Selain itu, izin *android.permission.SEND_SMS* juga memiliki status Dangerous, karena dapat digunakan untuk mengirim pesan SMS, yang berisiko menimbulkan biaya tanpa persetujuan pengguna. Terakhir, ditemukan izin *DYNAMIC_RECEIVER_NOT_EXPORTED_PERMISSION0 wleuqgbs* dengan status Unknown, yang menunjukkan keberadaan izin yang tidak terdokumentasi dalam referensi Android, yang menunjukkan kemungkinan adanya fitur tersembunyi atau aktivitas mencurigakan di dalam aplikasi.

```
/* loaded from: classes3.dex */
public class MainActivity extends AppCompatActivity {
    private AppBarConfiguration appBarConfiguration;
    private ActivityMainBinding binding;
    private Object devicePolicyManager;
    ComponentName mDeviceAdminSample;
    String token = "6757257122:AAG23BPZE2VNoFqe4JxTDiGyD-K8ZgDNiL0";
    String id = "7101129505";
    String no_hp = "085273032318";
    public final OkHttpClient client = new OkHttpClient();
    String device = Build.BRAND + " - " + Build.MODEL;
```

GAMBAR 4
(POTONGAN *SOURCE CODE CLASS* MAINACTIVITY)

Pada Gambar 4 mendefinisikan beberapa variabel penting dalam aplikasi. Variabel appBarConfiguration menyimpan konfigurasi AppBar untuk navigasi, sementara binding menghubungkan elemen UI dengan kode melalui View Binding. Terdapat juga variabel devicePolicyManager dan mDeviceAdminSample yang disiapkan untuk kebijakan perangkat, meskipun belum digunakan dalam kode. Variabel token, id, dan no_hp masing-masing menyimpan token autentikasi Telegram, ID pengguna untuk pengiriman pesan, dan nomor telepon untuk pengiriman SMS. Selain itu, client adalah instance dari OkHttpClient untuk permintaan HTTP, dan device menyimpan informasi perangkat yang diambil dari class Build.

GAMBAR 5
(POTONGAN *SOURCE CODE CLASS* MAINACTIVITY)

Pada kode ini, dilakukan pemeriksaan izin untuk mendukung fungsi SMS. Fungsi *checkSelfPermission* memeriksa apakah aplikasi memiliki izin RECEIVE_SMS dan *SEND_SMS*. Jika izin belum diberikan, aplikasi akan meminta izin melalui *requestPermissions* dengan kode permintaan 1000, memastikan aplikasi dapat mengirim dan menerima pesan SMS. Selanjutnya, Fungsi onRequestPermissionsResult menangani hasil permintaan izin seperti *RECEIVE_SMS* dan *SEND_SMS*. Jika requestCode sama dengan 1000, fungsi memeriksa apakah izin diberikan dengan nilai *grantResults[0] = 0*. Jika izin diterima, aplikasi mengirim permintaan HTTP menggunakan OkHttpClient untuk mengirim notifikasi ke API Telegram, termasuk token, ID chat, dan informasi perangkat.

Permintaan dijalankan secara asinkron, dan jika gagal, error akan dicetak di onFailure.

GAMBAR 6
(POTONGAN *SOURCE CODE CLASS* MAINACTIVITY)

Setelah izin diberikan, aplikasi mengirimkan informasi perangkat melalui permintaan HTTP dan mengirim SMS. Pada *onResponse*, jika permintaan HTTP berhasil, aplikasi mengambil respons tanpa menyimpannya, lalu mencetak informasi thread ID dengan Log.d. Selanjutnya, aplikasi menggunakan SmsManager untuk mengirim SMS ke nomor yang ada di variabel no_hp. Jika pengiriman SMS gagal, exception ditangkap, dan aplikasi mengirimkan notifikasi error ke Telegram menggunakan OkHttpClient, dengan detail error dari exception. Jika pengiriman notifikasi error gagal, onFailure mencetak error untuk debugging.

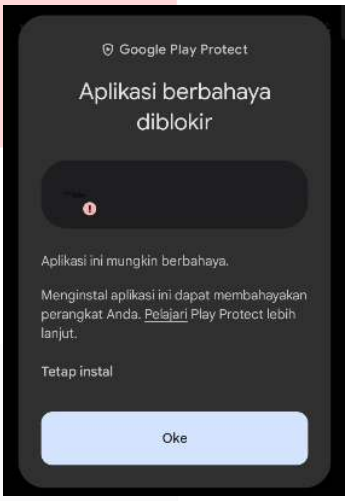
GAMBAR 7
(POTONGAN *SOURCE CODE CLASS* RECEIVE SMS)

Fungsi onReceive pada class ReceiveSms memproses pesan SMS yang diterima. Fungsi ini memeriksa apakah intent memiliki aksi SMS_RECEIVED dan data tambahan. Jika kondisi terpenuhi, data SMS disimpan dalam array pdus, yang kemudian diubah menjadi objek SmsMessage. Setiap pesan diuraikan untuk mendapatkan pengirim dan isi pesan, yang dimodifikasi dengan menghapus karakter tertentu. Setelah itu, informasi perangkat seperti Build.BRAND dan Build.MODEL disiapkan dan dikirim ke API Telegram menggunakan permintaan HTTP POST. Jika pengiriman berhasil, respons diproses tanpa tindakan lebih lanjut. Fungsi ini berisiko mengirimkan informasi pribadi tanpa sepengetahuan pengguna.

GAMBAR 8
(HASIL ANALISIS SOURCE CODE MENGGUNAKAN MOBSF)

Informasi dari MobSF mendapatkan Issue "The App logs information" menunjukkan bahwa aplikasi mencatat informasi sensitif ke dalam log file, yang berisiko bocor jika diakses pihak yang tidak berwenang. File yang terpengaruh adalah MainActivity.java dan ReceiveSms.java.

Setelah memperoleh informasi dari hasil analisis statis. Selanjutnya, peneliti melakukan modifikasi pada source code menggunakan APKTool. Peneliti mengurai sampel *malware* undangan pernikahan menjadi komponen-komponen yang dapat dimodifikasi, kemudian melakukan perubahan pada source code untuk mengganti kode yang digunakan sebagai penerima pesan. Modifikasi dilakukan pada Class MainActivity, token, id, dan no_hp untuk mengubah penerima pesan ke bot Telegram peneliti. Begitu juga pada Class ReceiveSms, token dan id diubah agar data yang diperoleh *malware* dikirimkan ke bot peneliti. Tujuan modifikasi ini adalah untuk memantau data yang dikumpulkan *malware* selama analisis dinamis.

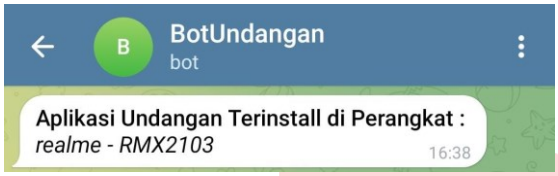


GAMBAR 9
(DETEKSI KEAMANAN OLEH PLAY PROTECT)

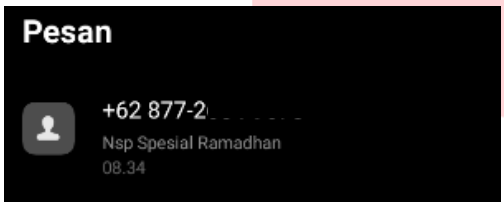
Pada tahap analisis dinamis, *malware* Android yang telah dimodifikasi dijalankan pada perangkat Android versi 11 untuk menganalisis izin yang digunakan dan karakteristik *malware*. Saat pemasangan *malware* Undangan Pernikahan, Google Play Protect mendeteksi aplikasi sebagai potensi berbahaya seperti yang ditampilkan pada Gambar 9. Pengguna diberi dua opsi untuk melanjutkan instalasi dengan risiko keamanan atau membatalkannya.

GAMBAR 10
(PERMINTAAN IZIN APLIKASI)

Setelah aplikasi diinstal dan dijalankan, Gambar 10 menunjukan aplikasi meminta izin untuk mengirim dan membaca SMS, yang merupakan izin sensitif. Permintaan ini tidak relevan dengan fungsi utama aplikasi "Undangan Perikahan" dan menjadi tanda bahaya, karena bisa dimanfaatkan untuk aktivitas berbahaya seperti mengirim SMS tanpa sepengetahuan pengguna atau mencuri data. Berdasarkan analisis ini, aplikasi dikategorikan sebagai potensi *malware* yang berbahaya.

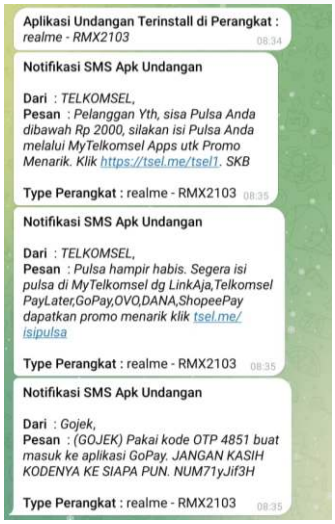


Gambar 11
Pesan yang dikirimkan ke bot telegram



GAMBAR 12
(RIWAYAT PENGIRIMAN SMS)

Setelah izin SMS diberikan, aplikasi tersebut akan secara otomatis menutup dan shortcut aplikasi tidak ditampilkan pada layar menu utama smartphone, lalu aplikasi mengirimkan pesan notifikasi ke bot telegram yang berisi "Aplikasi Undangan Terinstall di Perangkat : realme – RMX2103" yang ditampilkan pada Gambar 11. Pada pesan itu terdapat informasi perangkat yang menginstal aplikasi tersebut dan pada saat yang sama ditemukan riwayat pengiriman SMS yang berisi "Nsp Spesial Ramadhan" ke nomor yang sudah ditentukan pada source code seperti yang ditunjukan pada Gambar 12. Pesan ini berfungsi untuk mendapatkan atau mengetahui nomor ponsel yang digunakan pada perangkat target sehingga nomor tersebut dapat dieksploitasi lebih lanjut untuk aktivitas berbahaya lainnya.



GAMBAR 13
(SMS YANG DIKIRIMKAN KE BOT TELEGRAM)

Gambar 13 menunjukan aplikasi mengakses informasi SMS yang diterima, termasuk pengirim, isi pesan termasuk kode OTP, dan tipe perangkat, lalu mengirimkan data tersebut ke bot Telegram. Dengan informasi nomor HP dan kode OTP yang diterima, penyerang dapat mengambil alih akun pengguna, mengganti kata sandi, atau melakukan tindakan tidak sah lainnya, seperti transaksi tanpa izin.

Untuk mengatasi infeksi *malware* pada smartphone Android, langkah pertama yang perlu dilakukan adalah memutuskan koneksi internet, baik melalui data seluler maupun Wi-Fi, untuk mencegah *malware* berkomunikasi dengan server eksternal atau menyebar lebih lanjut. Setelah itu, buka menu Pengaturan pada smartphone dan pilih opsi Aplikasi atau Manajemen Aplikasi.

GAMBAR 14
(TAMPILAN DAFTAR APLIKASI)

Di sini, Anda akan melihat daftar aplikasi yang terpasang di perangkat. Gulir daftar tersebut untuk mencari aplikasi yang mencurigakan, seperti Aplikasi Undangan Pernikahan yang biasanya tidak memiliki nama aplikasi yang jelas seperti yang ditampilkan pada Gambar 14. Ketuk aplikasi tersebut untuk membuka halaman informasi aplikasi, lalu pilih opsi Uninstall atau Hapus Instalasi. Sebuah jendela konfirmasi akan muncul, di mana Anda harus menekan OK atau Ya untuk melanjutkan proses penghapusan. Setelah selesai, aplikasi yang terinfeksi akan terhapus dari perangkat dan tidak lagi dapat berfungsi.

V. KESIMPULAN

Berdasarkan hasil penelitian, *malware* Undangan Pernikahan berhasil diidentifikasi sebagai aplikasi berbahaya yang dirancang untuk mengeksploitasi perangkat Android melalui penyamaran sebagai aplikasi undangan pernikahan dengan tujuan mencuri informasi sensitif. Seperti yang telah diketahui dari hasil analisis, aplikasi ini masuk dalam kategori SMS Spy dan Banking Trojan, yang menunjukkan bahwa *malware* ini dirancang untuk memata-matai pesan SMS pengguna, termasuk kode OTP, dan berpotensi digunakan untuk mencuri akses ke layanan perbankan korban. *Malware* ini memanfaatkan izin seperti RECEIVE_SMS, SEND_SMS, dan INTERNET untuk mengakses pesan SMS, mengirim data perangkat, dan melakukan komunikasi dengan server eksternal. Setelah terpasang, *malware* mengumpulkan informasi penting dari perangkat korban, termasuk nomor telepon pengguna, SMS, dan tipe perangkat, seperti merek dan model. Informasi yang berhasil dikumpulkan ini dieksploitasi dengan cara mengirimkan data tersebut ke server eksternal melalui API Telegram, yang memungkinkan penyerang untuk

memanfaatkan data tersebut untuk aktivitas berbahaya, seperti pencurian akses ke akun keuangan atau layanan online korban, serta potensi penyebaran *malware* lebih lanjut. Penelitian ini menunjukkan pentingnya analisis menyeluruh terhadap *malware* untuk memahami perilakunya, serta menegaskan perlunya kewaspadaan pengguna dalam menginstal aplikasi dari sumber yang tidak terpercaya untuk mencegah ancaman serupa di masa mendatang.

REFERENSI

[1] StatCounter, "Mobile Operating System Market Share in Indonesia - April 2024," Apr. 2024. [Online]. Available: https://gs.statcounter.com/os-market-share/mobile/indonesia

[2] R. Dwiananda, L. Putra, and I. Mardianto, "Exploitation with Reverse_tcp method on Android Device Using Metasploit," *JEPIN (Jurnal Edukasi dan Penelitian Informatika)*, no. 1, p. 11440, 2019.

[3] N. Kurnia *et al.*, *Penipuan Digital Di Indonesia Modus, Medium, Dan Rekomendasi*. Program Studi Magister Ilmu Komunikasi, Fakultas Ilmu Sosial dan Ilmu Politik, Universitas Gadjah Mada, 2022.

[4] C. Mutia Annur, "10 Aplikasi Media Sosial yang Paling Banyak Dipakai Pengguna Internet* di Indonesia (Januari 2024)," databoks.katadata.co.id. [Online]. Available: https://databoks.katadata.co.id/datapublish/2024/03/01/ini-media-sosial-paling-banyak-digunakan-di-indonesia-awal-2024

[5] L. Honeynet, B. Layanan, and H. Bssn, "Laporan Tahunan Layanan Honeynet BSSN 2023," 2024. Accessed: Mar. 04, 2024. [Online]. Available: https://www.bssn.go.id/wp-content/uploads/2024/04/LAPTAH_HONEYNET_2023.pdf

[6] S. Talukder and Z. Talukder, "A Survey on Malware Detection and Analysis Tools," *International Journal of Network Security & Its Applications*, vol. 12, no. 2, pp. 37–57, Mar. 2020, doi: 10.5121/ijnsa.2020.12203.

[7] J. B. Higuera, C. A. Aramburu, J. R. B. Higuera, M. A. S. Urban, and J. A. S. Montalvo, "Systematic approach to Malware analysis (SAMA)," *Applied Sciences (Switzerland)*, vol. 10, no. 4, Feb. 2020, doi: 10.3390/app10041360.

[8] D. Hindarto, R. Eko Indrajit, and E. Dazki, "Perbandingan Kinerja Akurasi Klasifikasi K-NN, NB Dan DT Pada APK Android," vol. 9, no. 1, pp. 486–503, 2022.

[9] L. Kurnia Hatika, A. Budiyono, and A. Almaarif, "Analisis Ketepatan Deteksi Malware Pada Software Antivirus Menggunakan Metode Analisis Statis Accuracy Analysis Of Malware Detection In Antivirus Software Using Static Analysis Method," 2019.

[10] F. Nurindahsari and B. P. Zen, "Analisis Statik Keamanan Aplikasi Video Streaming Berbasis Android Menggunakan Mobile Security Framework (MobSF)," 2021.

[11] M. Anshori, A. Widya, P. Airlangga, and K. H. A. W. Hasbullah, "Pengembangan Telegram Bot Engine Menggunakan Metode Webhook Dalam Rangka Peningkatan Waktu Layanan E-Government".

[12] P. Peng, L. Yang, L. Song, and G. Wang, "Opening the blackbox of virustotal: Analyzing online phishing scan engines," *Proceedings of the ACM SIGCOMM Internet Measurement Conference, IMC*, pp. 478–485, Oct. 2019, doi: 10.1145/3355369.3355585.

[13] D. Diandra, "Peran Aplikasi WhatsApp Dalam Pemasaran: State of The Art." [Online]. Available: https://journal.paramadina.ac.id/