DOI:10.14421/ijid.2021.2839 IJID (International Journal on Informatics for Development), *e-ISSN: 2549-7448*
*Vol. 10, No. 2, 2021, Pp. 102-111*

# Analysis of Remote Access Trojan Attack using Android Debug Bridge

Deco Aprilliansyah
Master Program of Informatics
Universitas Ahmad Dahlan
Yogyakarta, Indonesia
deco2007048003@webmail.uad.ac.id

Imam Riadi
Department of Information Sustem
Universitas Ahmad Dahlan
Yogyakarta, Indonesia
imam.riadi@is.uad.ac.id

Sunardi
Department of Electrical Engineering,
Universitas Ahmad Dahlan
Yogyakarta, Indonesia
sunardi@mti.uad.ac.id

**Article History**
Received July 25th, 2021
Revised January 6th, 2022
Accepted January 10th, 2022
Published February, 2022

*Abstract*—**The security hole in the Android operating system is sometimes not realized by users such as malware and exploitation by third parties to remote access. This study was conducted to identify the vulnerabilities of the Android operating system by using Ghost Framework. The vulnerability of the Android smartphone is found by using the Android Debug Bridge (ADB) with the exploitation method as well as to analyze the test results and identify remote access Trojan attacks. The exploitation method with several steps from preparing the tools and connecting the testing commands to the testing device has been conducted. The result shows that Android version 9 can be remotely accessed by entering the exploit via ADB. Some information has been obtained by third parties, enter and change the contents of the system directory by remote access like an authorized to do any activities on the device such as opening lock screen, entering the directory system, changing the system, etc.**

***Keywords—security hole; operating system; malware; ghost framework; vulnerability***

## 1 INTRODUCTION

Facing the era of industry 4.0 every person at least has one smartphone to communicate, exchange data, look for information, or to entertain such as playing video games, online studies, listening to music, and watching a video [1][2]. The data statistics from 2010 show 3,6 billion smartphone users [3]. There are many operating systems on smartphones, one of them is Android with a working framework based on Linux kernel and created by Google [4]. The operating system based on the open-source Linux kernel offers high flexibility because it can be customized by dominant operating systems. Initially, Android applications were modified within Java [5]. Android is open source so anyone can contribute to developing this system [6]. This advantage also raises security concerns [7]. In the Android operating system, there are no security holes not realized by users. Many kinds of security threats can happen such as malware and exploitation by third parties to remote access [8]. On the Android operating system, attacks occur through networks, applications, and firmware vulnerabilities in the operating system itself. Attacks on the Android system can be classified into attacks on hardware, attacks on the kernel, attacks on the hardware abstraction layer, and attacks on application packages [9]. Types of attacks through the network such as DDoS by flooding data on the FTP server, spoofing on voice conversion, and Trojan Horses on banking applications [10], [11],[12]. The types of attacks are through applications such as cracking, banking trojans, and eavesdropping.

Google proposes a security patch update every month for Android system users [13] with contains the list of security that has been fixed, every list has several common vulnerabilities and exposures [14]. Usually, the update is in a form of a binary file. Unfortunately, only a few types of smartphones can get it whereas getting the update of a security patch can minimalize the security risk on the device itself. It is caused by there are so many different active versions of Android making security updates and vulnerability responses make this issue difficult to control[15].

Vulnerability detection is one way to find security holes that aims to show potential security leaks. The security risk on Android can be minimalized by using security analysis on the vulnerability system. In a digital environment where there are various cyber threats, regular vulnerability assessments are the right choice to protect our data. In this way, sensitive data can be properly maintained and protected from various adverse effects of cyber-attacks. Various methods can be implemented such as exploitation, penetration tester, and man in the middle attack (MITM) [7],[16],[17]. Users can use several tools like Apktool, Metasploit Framework, TheFatRat, ADB-Toolkit, Ghost Framework, Hacktronian, etc.

The Ghost Framework tool can be used to test whether or not remote access [18]. These tools will be used in this study to do exploitation the smartphone with the Android system. After obtaining the result and analyzing the security risk of the Remote Access Trojan, the Android user is expected to be more aware and keep the security on their device.

Based on the background of the problem above, this research conducts a literature study using research sources that have been carried out previously as research references. Some of the sources used as references include the first research conducted by R. Mayrhofer, J. Vander Stoep, C. Brubaker, and N. Kralevich in 2021 entitled "The Android Platform Security Model" which is the research of security model on Android. This model requires a difficult balance between end-user security, privacy and usability, application developers' assurance, and system performance under strict hardware constraints. Based on the threat model definition and the context of the Android ecosystem in which it runs, they analyzed how different security measures in the past and current Android implementations work together to mitigate current threats. This article aims to document the Android security model and determine its impact in the context of ecosystem constraints and historical development. In particular, this research is encouraging contribution and, for the first time, defined an Android security model based on security principles and the broader environment in which Android runs. This article focuses on the security and privacy measures of the Android platform itself, that is, the code that runs on user devices belonging to AOSP.

The second one is research conducted by A. Amin, A. Eldessouki, MT Magdy, N. Abdeen, H. Hindy, and I. Hegazy in 2019 with the research title "AndroShield: Automated Android Applications Vulnerability Detection, a Hybrid Static, and Dynamic Analysis Approach" by conducted research related to Android security testing with the Automated Vulnerability Detection Technique. The test results conclude that the shorter trial period of published apps can make apps more vulnerable to malicious users and hackers who can gain unauthorized access to personal data. They then propose a framework for Android vulnerability detection that can be used by developers and ordinary users. The web application is designed to be easy to use. The framework analyzes uploaded APK files using two methods: static analysis and dynamic analysis and generates analysis reports. The types of vulnerabilities found in the project were information leaks, intent crashes, insecure network requests (HTTP requests), exported Android components, enabled backup mode, and enabled debugging mode. The framework is available at https://github.com/AmrAshra.

The third one is research conducted by P.A. Bhat and K. Dutta in 2019 with the research title "A survey on various threats and current state of security in Android platform" examined attacks on various layers of the Android system. Researchers and software developers hope to formulate strategies for detecting and preventing attacks. This article conducts comprehensive research on the state of Android security domains. This research investigated various threats and security measures related to this category and conducted an in-depth analysis of potential issues in the Android security field. This research aims to help researchers acquire knowledge about Android security from all aspects, and build a more comprehensive, robust, and efficient solution to the threats faced by Android. Although the latest Android system now has strong security mechanisms, malware developers are discovering new techniques to bypass system security policies. Most malicious applications are designed to gain root access to the system. Malware developers are built so well that they are difficult to detect, and they continue to work silently; users do not know that malware has compromised their system. An important conclusion of this research is that lack of understanding of the use of security protocols and poor developer practices are the main reasons why Android faces security threats. The research can help researchers acquire knowledge in the Android security field from all aspects, and build a more comprehensive, robust, and efficient solution to the threats faced by Android.

The fourth one is research conducted by J. Wu and M. Yang in 2017 with the research title "LaChouTi: Kernel vulnerability responding framework for the fragmented Android devices" researched fragmentation in the Android ecosystem. They respond to kernel vulnerabilities in fragmented devices. In particular, they also proposed and implemented LaChouTi which is an automated kernel security update framework consisting of cloud services and end application updates. LaChouTi tracks and identifies security vulnerabilities based on CVEPatch for the target Android kernel then generates a differential binary patch for the specified result and is finally applied to the patch on the kernel. Subsequently implemented LaChouTi on new commercial devices in collaboration with four internationally renowned manufacturers. The results show that LaChouTi is effective for manufacturer security updates.

The fifth is research conducted by R.D. Putra and I. Mardianto in 2019 with the research title “Exploitation with Reverse_tcp Method on Android Device using Metasploit” researched Android security by testing with the exploitation method. They say that Android uses a kernel-level sandbox and isolation mechanism to separate apps from each other and control communication between apps or access to resources. The motivation of this research is to determine the characteristics of the exploit based on the reverse TCP attack method on Android devices and to determine the time variable required for the payload to enter the system from the target. The contribution of this research is to understand how the reverse TCP method works on Android system exploit attacks and how to avoid it. The results obtained from an exploit attack are understanding how to perform an exploit attack, analyzing the interaction between the attacker and the victim, analyzing how the exploit attack works, and the time it takes to prepare the payload using a bash script shell. All the details of the analysis will provide conclusions and strategies to make the Android operating system more secure.

Remote access Trojan is a malware type of trojan horse that attackers enable to remote administrative toward the infected target. The testing steps of remote access consists of connecting the IP testing device by using Ghost Framework tools, doing the exploitation, applying the command of ADB shell, and doing the remote access [21],[22].

The first step for the testing is to connect the IP testing device to Ghost Framework [23]. Exploitation is an activity to plant the exploit code into the system. This step is done by exploiting the system through ADB. ADB shell is to write a series of commands to remote access and other activity on the testing device [24]. The purpose of remote access on the testing device is to understand what can be done when the device is on remote access.

This research is conducted with the scenario of exploiting the Android device through Android Debug Bridge (ADB). The process of exploiting remote access on the Android system can help us to understand how the third parties doing the attack of remote access Trojan [19]. An exploit is a series of commands, data, or software that exploits the vulnerability of the target computer [20]. In short, exploit is code to attack the aim target.

## 2 METHOD

### 2.1 *Post-Exploitation*

Post-exploitation refers to any actions taken after a session is opened. A session is an open shell from a successful exploit or brute force attack. A shell can be a standard shell or Meterpreter [25]. Some of the actions that can take in an open session include Collect System Information, Pivot, Run Meterpreter Modules, and Searching the File System. Post-Exploitation is used to determine the underlying capabilities and values of the target system.

The Post-Exploitation process aims to be able to access all parts of the target system without being detected by the victim. Penetration testers are used to exploit the target computer system without validating and analyzing the value of the data presented on the victim's system [26]. Testers can dig deep for more information about the target system if they find the information useful. Penetration testers can also analyze system configuration settings, communication modes, registry settings, and connection methods that connect device-specific networks. In this process, the methods and requirements may differ depending on the situation and the rules of cooperation.

### 2.2 *Android*

Android is a Linux-based operating system designed primarily for touch screen mobile devices, such as smartphones and tablets. From black and white cell phones to the latest smartphones or minicomputers, operating systems have changed a lot in the last 15 years. One of the most widely used mobile operating systems today is Android. An android is a software that was founded in Palo Alto, California in 2003. Android is made for everyone like developers, designers, and device manufacturers. This means that many people will be able to experiment, imagine, and create things that have never existed in the world. The version history of the Android OS begins with the discharge of Android 1.0 beta in November 2007. Since April 2009, each Android version has been developed with a code name that supported a dessert item. These versions are released in alphabetical order: Cupcake, Donut, Eclair, Froyo, Gingerbread, Honeycomb, frozen dessert Sandwich, Jelly Bean, KitKat, Lollipop, Marshmallow, Nougat, Oreo, and Pie. Since Android 10 google does not use fruit names for naming Android versions. The latest Android version is Android 12 which was launched in May 2021 with SDK Build-Tools version 31.

The Android architecture consists of five parts as follow:

- Linux kernel: Android uses a powerful Linux kernel and supports a variety of hardware drivers. The kernel is the core of the operating system. It manages the input and output requests from the software. This provides basic system functions, such as process management, memory management, device management (such as camera, keyboard, monitor, etc.), and the kernel handles everything.
- Libraries: There is a set of libraries in the Linux kernel, including open-source web browsers, such as the WebKit and libc libraries. The library is used to play and record audio and video. SQLite is a useful database for storing and sharing application data. The SSL library is responsible for Internet security, etc.
- Android Runtime: The android runtime provides a key component called the Dalvik virtual machine, which is a Java virtual machine. It is specially designed and optimized for Android. Dalvik VM is a process virtual machine in the Android operating system. It is software to run applications on Android devices. Dalvik VM uses core Linux functions, such as memory management and multithreading in the Java language. Dalvik VM allows each Android application to run its process. Dalvik VM executes files index format.
- Application Framework: The application framework layer provides many advanced services for applications, such as window manager, display system, package manager, resource manager, etc. Allow app developers to use this service in their apps.
- Applications: All android apps can be found at the top level and write your apps and install them at that level. Examples of such applications include contacts, books, browsers, services, etc. Each application plays a different role in the application.

## 2.3 *Ghost Framework Application*

Ghost Framework is an Android post-exploitation framework that uses the Android Debug Bridge to remotely access an Android device. By using this Framework tool, we can control Android devices. We can use this framework to control old Android devices with the debug bridge turned on in "Developer Options". Now, this becomes very harmful because the attacker gains full administrative control of the vulnerable Android device. If you forget the password of the remote Android device, you can use the Ghost framework to delete it. It can also be used to access the shell of a remote Android device without using OpenSSH or other protocols. Using the ghost framework, we can perform various activities, such as viewing device activity information, viewing device network information, clicking the specified screen device, clicking the specified x-axis and y-axis, controlling the device keyboard, pressing/simulating keys on the target device, and opening the URL on the device, Open the device shell and restart the device.

## 2.4 *Tools and Software Testing*

The tools and software used to exploit Remote Access testing devices via the Android Debug Bridge can be seen in Table 1.

Table 1. Tools and Software Testing

| No. | Tools & Software | Version | Function |
|---|---|---|---|
| 1. | Laptop | MSI-GF639SCSR | Test Equipment |
| 2. | Smartphone 1 | Xiaomi Redmi Note 5 Android 9 & Android 12 (AOSP) | Test Equipment |
| 3. | Smartphone 2 | Xiaomi Redmi 5 plus Android 8.1 (MIUI) | Test Equipment |
| 4. | Smartphone 3 | Vivo Y71 Android 8.1 (FuntouchOS) | Test Equipment |
| 5. | Kali Linux | 21.1 | Operating System |
| 6. | Ghost Framework | 6.0 | Exploitation & Remote Access |

Tools and software as listed in Table 1 are needed for the exploitation so that remote access can be done through ADB. The tools and software that are used consist of Laptop MSI-GF639SCSR, Xiaomi Redmi Note 5 brand smartphone with Android 9 & Android 12 (AOSP) version, Xiaomi Redmi 5 Plus brand smartphone with Android 8.1 version, Vivo Y71 brand Smartphone with Android 8.1 version, and Kali Linux 21.1 as operating system [18]. The Ghost Framework is a tool to do the exploitation process for remote access devices.

## 2.5 *Scenario Attack*

This section provides an overview of the Attack scenario on an Android smartphone. Smartphones act as targets for attacks. The following is an overview of the attack process as shown in Figure 1.

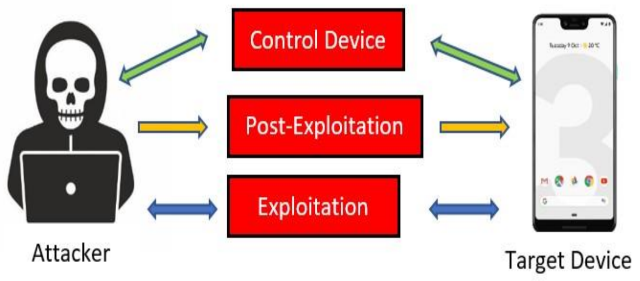


Figure 1. An attack scenario against the target

Figure 1 provides information on the attack scenario which consists of three main stages. The first stage is the exploitation process which is marked with a blue arrow, if the process is successful, there will be feedback to the attacker so that they can proceed to the next stage. The second stage is the post-exploitation process which is marked with an orange arrow. The third stage is an attempt to obtain access rights to control the victim's smartphone as shown in the green arrow.

## 2.6 *Attack Process*

In the attack section, the process functions to determine the extent of threats that can occur because mobile applications for data resource sharing services are still not ready to receive malware attacks. Malware attacks can covertly take over a mobile device and use it for nefarious purposes ranging from stealing data and spying on its users. Therefore, to avoid the worst possible scenario, testing is carried out through an attack process on mobile devices with the Ghost Framework. The following are the stages of attack in this study which consist of six stages as shown in Figure 2.

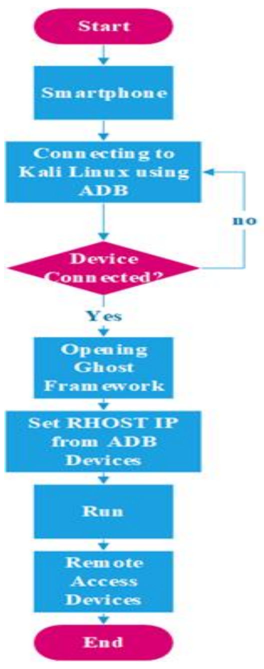


Figure 2. A flowchart of an attack process

The flow of the testing system can be seen in Figure 2 with described as follow:

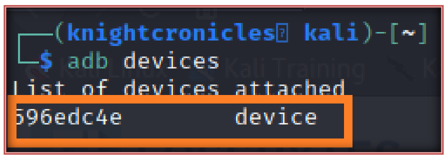
- Testing devices that have been confirmed to be connected to Kali Linux using the ADB
- Check whether the device is connected or not by opening the Ghost Framework.
- Set the IP RHOST obtained from the device under test
- Perform the command "run"
- If already connected, the device can be remotely accessed and given the ADB shell command.

## 2.7 *Connecting Device*

The Android testing device will be connected to the software testing. There are many steps of connecting which are a connection to Kali Linux and Connect IP & PORT Device to Ghost Framework.

### 2.7.1 *Connection to Kali Linux*

The Android device will be connected to Linux through ADB. Before testing device must turn on the USB mode debugging. Next is connected using the USB cable to Linux. The status of the device that is successfully connected can be seen in Figure 3 Android device code 596edc9e is connected with the ADB server this time Linux command "ADB devices".

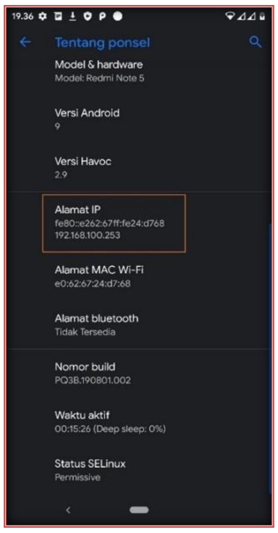


```
┌──(knightcronicles㉿ kali)-[~]
└─$ adb devices
List of devices attached
596edc4e          device
```

Figure 3. ADB devises interface

### 2.7.2 *Connect IP & PORT Device to Ghost Framework*

The Android device successfully connected through ADB will be connected using IP and port ADB on the Android device itself. The process of connecting to Android with Ghost Framework can be seen in Figures 4, 5, and 6.



19.36

Tentang ponsel

Model & hardware
Model: Redmi Note 5

Versi Android
9

Versi Havoc
2.9

Alamat IP
fe80::e262:67ff:fe24:d768
192.168.100.253

Alamat MAC Wi-Fi
e0:62:67:24:d7:68

Alamat bluetooth
Tidak Tersedia

Nomor build
PQ3B.190801.002

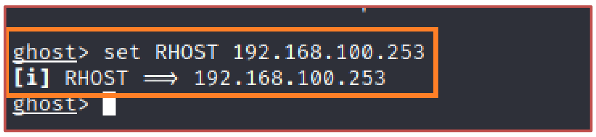
Waktu aktif
00:15:26 (Deep sleep: 0%)

Status SELinux
Permissive

Figure 4. The IP address of a smartphone

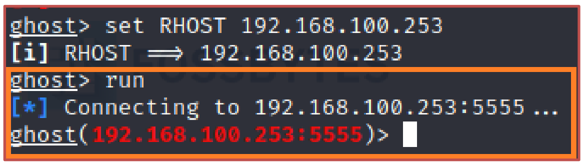
The IP on the testing device is 192.168.100.253 (Figure 4), then the IP will be connected to Ghost Framework. In Figure 5, before connecting to the IP ADB, on terminal Ghost Framework first perform the command "set RHOST IP". The IP of the device being tested is 192.168.100.253 so that the command written is "set RHOST 192.168.100.253". It can be seen in Figure 6, after setting up RHOST, to connect to the Ghost Framework just only do the "run" command.

After performing the command, see whether the device being tested is connected or not. There will be a statement saying failed to connect to <IP>. In the first experiment, the tested device failed to connect with the description "[-] Failed to connect to 192.168.100.253:5555!". Conversely, if successfully connected, see ghost information (IP: PORT) with the description "ghost (192.168.100.253:5555)>".List Command Tools.



```
ghost> set RHOST 192.168.100.253
[i] RHOST ⟹ 192.168.100.253
ghost>
```

Figure 5. SET RHOST IP address



```
ghost> set RHOST 192.168.100.253
[i] RHOST ⟹ 192.168.100.253
ghost> run
[*] Connecting to 192.168.100.253:5555 ...
ghost(192.168.100.253:5555)>
```

Figure 6. Connecting to IP address ADB

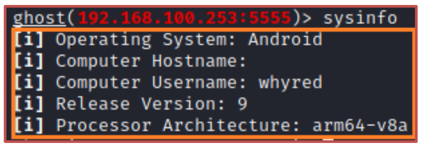
Many commands can be run on the Ghost Framework to remote access. The device shell command can be seen in Table 2. After the testing device is connected to the Ghost Framework accordingly remote access and also various kinds of commands on Android can be performed. Some activities that can be done by remote access to the testing device after connecting to the Ghost Framework include accessing system information, entering the system/data directory, changing system contents, accessing the ADB shell, opening the URL, etc.

Table 2. A List of Commands

| Command | Description |
|---|---|
| Clear | Clear terminal window |
| Activity | Show device activity information |
| Netstat | Show device network information |
| Sysinfo | Show device system information |
| Wifi | Control device wifi services |
| Openurl | Open URL on the device |
| Screen | Control device screen |
| Screenshot | Take device screen shoot |
| Shell | Open device shell |
| Upload | Upload local file |
| Click x,y | Clicks the specified x and y-axis |

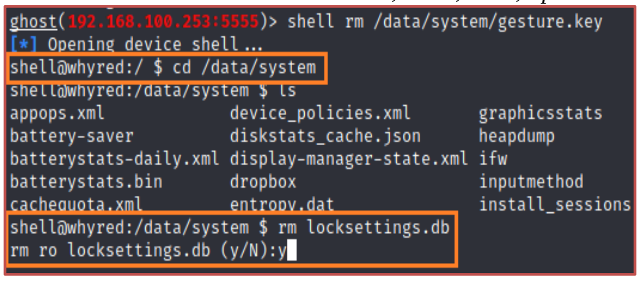
### 2.7.3 *Access to System Information*

Remote access makes the device accessible to the system. Some of the commands can be done as shown in Figures 7 and 8. By using the sysinfo command in Figure 6, system information will appear which are Operating System Android, Computer Username whyred with the codename for the Xiaomi Redmi Note 5 AI, the Release Version is the Android version which is version 9, as well as information on the Processor Architecture, namely arm64-v8a.



```
ghost(192.168.100.253:5555)> sysinfo
[i] Operating System: Android
[i] Computer Hostname:
[i] Computer Username: whyred
[i] Release Version: 9
[i] Processor Architecture: arm64-v8a
```

Figure 7. A sysInfo display



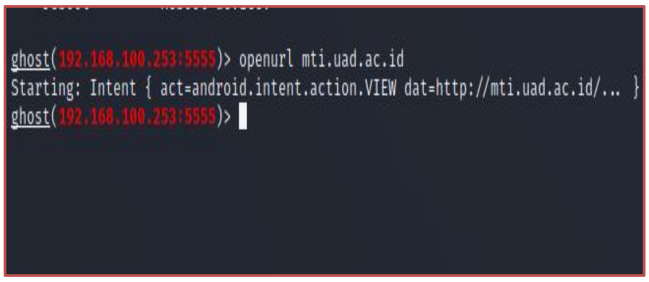
```
ghost(192.168.100.253:5555)> shell rm /data/system/gesture.key
[*] Opening device shell ...
shell@whyred:/ $ cd /data/system
shell@whyred:/data/system $ ls
appops.xml              device_policies.xml       graphicsstats
battery-saver           diskstats_cache.json      heapdump
batterystats-daily.xml  display-manager-state.xml ifw
batterystats.bin        dropbox                   inputmethod
cachequota.xml          entropy.dat               install_sessions
shell@whyred:/data/system $ rm locksettings.db
rm ro locksettings.db (y/N):y
```

Figure 8. A device shell for removing a file

In Figure 8 with the command shell, we can access, add, and delete contents of the system. Some many activities can be done such as deleting the lock screen database or removing the lock screen security.

### 2.7.4 *Access URL*

Remote access devices can access the URL using a browser without opening the lock screen. As an example in Figure 9, using the openurl command will make an intent to the browser to access the destination website.



```
ghost(192.168.100.253:5555)> openurl mti.uad.ac.id
Starting: Intent { act=android.intent.action.VIEW dat=http://mti.uad.ac.id/... }
ghost(192.168.100.253:5555)>
```

Figure 9. An Openurl display

In figure 9 the test is done by writing the command "openurl mti.uad.ac.id" and automatically Ghost Framework will start the data view intent to the address http://mti.uad.ac.id/.

## 3 RESULT AND DISCUSSION

### 3.1 *Attack Analysis*

Based on the tests that have been carried out, the test device on Android 9.0 was successfully remote access. Some of the commands that can be done can be seen in Table 3. Some information is obtained and allows to enter and change the contents of the system directory. The testing device can be remotely accessed from the computer to carry out the test like an authorized user.

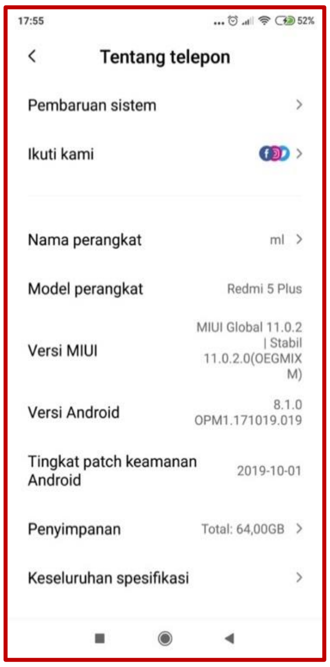
Table 3. A List of Activities of Android 9.0

| Activity | Result | Description |
|---|---|---|
| **Connect to Device** | True | Connecting IP device to Ghost Framework |
| **Access system info** | True | Obtain system information |
| **Access battery info** | True | Obtain test battery information |
| **Access the system directory** | True | Enter the system directory & change the contents of the system |
| **Access the website** | True | Open URL |
| **Screenshot device** | True | Take screenshot |
| **Remote device** | True | Remote access device |

In Table 3, the test device on Android 9.0 is successfully connected using the IP device to the Ghost Framework. Once connected, a session will open which allows the tester to post-exploitation. There are various kinds of information obtained including system information on the device used, battery information, entering and changing content in the system, accessing URLs, getting a screenshot, and remotely accessing targets.

### *3.1.1 Test Results on Xiaomi Redmi 5 Plus Android 8.1*

It can be seen on the Android 9.0 AOSP version that the exploitation and post-exploitation process was successfully carried out without any problems. However, in the tests carried out on android 8.1, the exploitation process failed because the device IP could not connect to the Ghost Framework. For more details, see Figures 10 and 11.



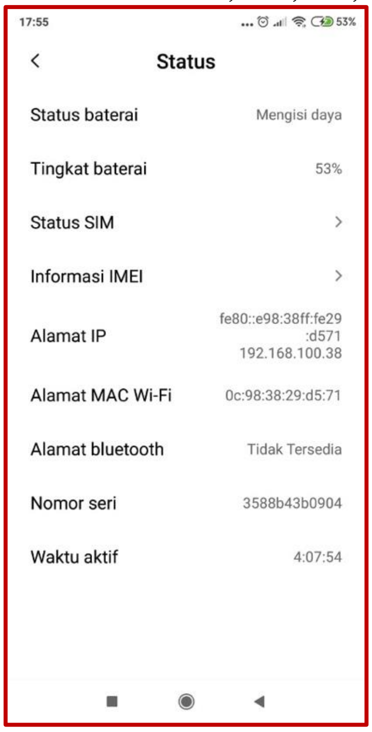
17:55

Tentang telepon

| | |
|---|---|
| Pembaruan sistem | |
| Ikuti kami | |
| Nama perangkat | ml |
| Model perangkat | Redmi 5 Plus |
| Versi MIUI | MIUI Global 11.0.2 \| Stabil 11.0.2.0(OEGMIXM) |
| Versi Android | 8.1.0 OPM1.171019.019 |
| Tingkat patch keamanan Android | 2019-10-01 |
| Penyimpanan | Total: 64,00GB |
| Keseluruhan spesifikasi | |

Figure 10. Properties of Xiaomi Redmi 5 Plus

Figure 10 shows a description of the Xiaomi Redmi 5 Plus device that uses MIUI 11.2 with the Android 8.1 version.



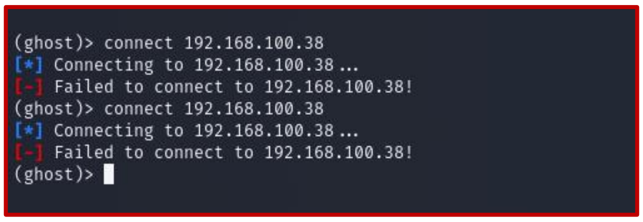
17:55

Status

| | |
|---|---|
| Status baterai | Mengisi daya |
| Tingkat baterai | 53% |
| Status SIM | |
| Informasi IMEI | |
| Alamat IP | fe80::e98:38ff:fe29:d571 192.168.100.38 |
| Alamat MAC Wi-Fi | 0c:98:38:29:d5:71 |
| Alamat bluetooth | Tidak Tersedia |
| Nomor seri | 3588b43b0904 |
| Waktu aktif | 4:07:54 |

Figure 11. IP Address Xiaomi Redmi 5 Plus

Figure 11 shows a description of the IP Address of the device that will be connected to the Ghost Framework. The connection process for this device can be seen in Figure 12.



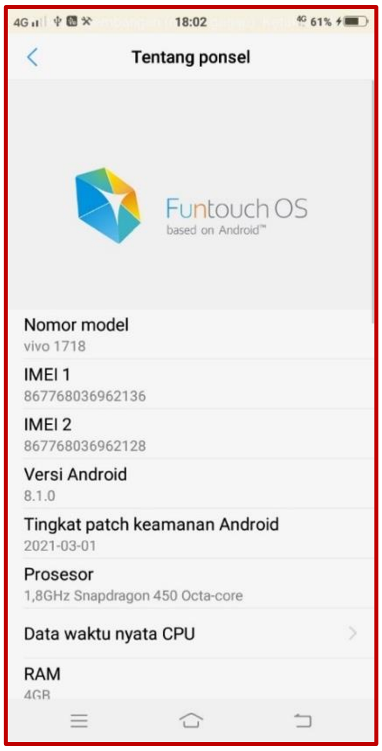
```
(ghost)> connect 192.168.100.38
[*] Connecting to 192.168.100.38 ...
[-] Failed to connect to 192.168.100.38!
(ghost)> connect 192.168.100.38
[*] Connecting to 192.168.100.38 ...
[-] Failed to connect to 192.168.100.38!
(ghost)>
```

Figure 12. Connecting to IP address ADB on Xiaomi Redmi 5 Plus

It can be seen in Figure 12 that the IP Address connection process between the Xiaomi Redmi 5 Plus failed.

### *3.1.2 Test Results on Vivo Y71 Android 8.1*

In Android Vivo Y71 Android 8.1 the exploitation process failed because the device IP could not connect to the Ghost Framework. For more details, see Figures 13 and 14.


18:02
Tentang ponsel
Funtouch OS
based on Android™

Nomor model
vivo 1718
IMEI 1
867768036962136
IMEI 2
867768036962128
Versi Android
8.1.0
Tingkat patch keamanan Android
2021-03-01
Prosesor
1,8GHz Snapdragon 450 Octa-core
Data waktu nyata CPU
RAM
4GB

Figure 13. Properties of Vivo Y71

Figure 13 shows a description of the Vivo Y71 device that uses FuntouchOS with the Android 8.1 version. The last security patch update was updated in March 2021.

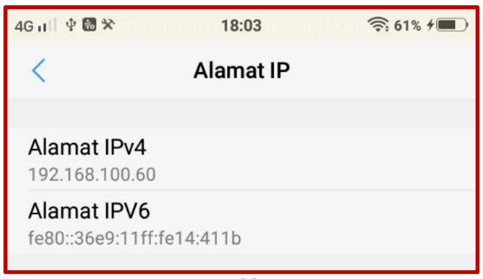
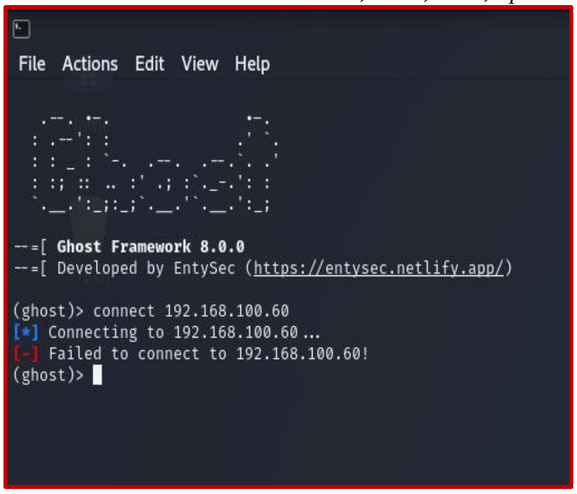

18:03
Alamat IP
Alamat IPv4
192.168.100.60
Alamat IPV6
fe80::36e9:11ff:fe14:411b

Figure 14. IP Address Vivo Y71

Figure 14 shows a description of the IP Address of the device that will be connected to the Ghost Framework. The connection process for this device can be seen in Figure 15.

It can be seen in Figure 15 that the IP Address connection process between Vivo Y71 Plus failed because the device vendor disabled the wireless debugging process. This causes this device to be unable to exploit and remotely access.

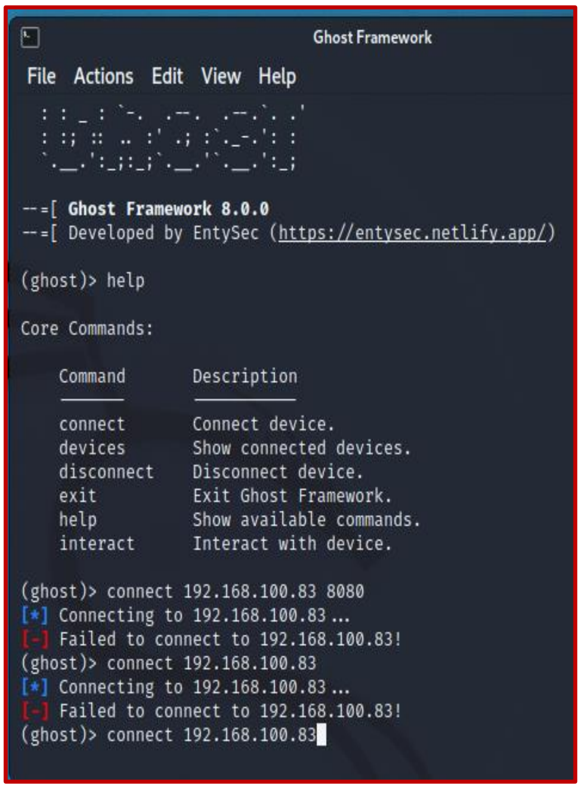


```
--=[ Ghost Framework 8.0.0
--=[ Developed by EntySec (https://entysec.netlify.app/)

(ghost)> connect 192.168.100.60
[*] Connecting to 192.168.100.60 ...
[-] Failed to connect to 192.168.100.60!
(ghost)>
```

Figure 15. Connecting to IP address ADB on Vivo Y71

### *3.1.3 Test Results on Android 12*

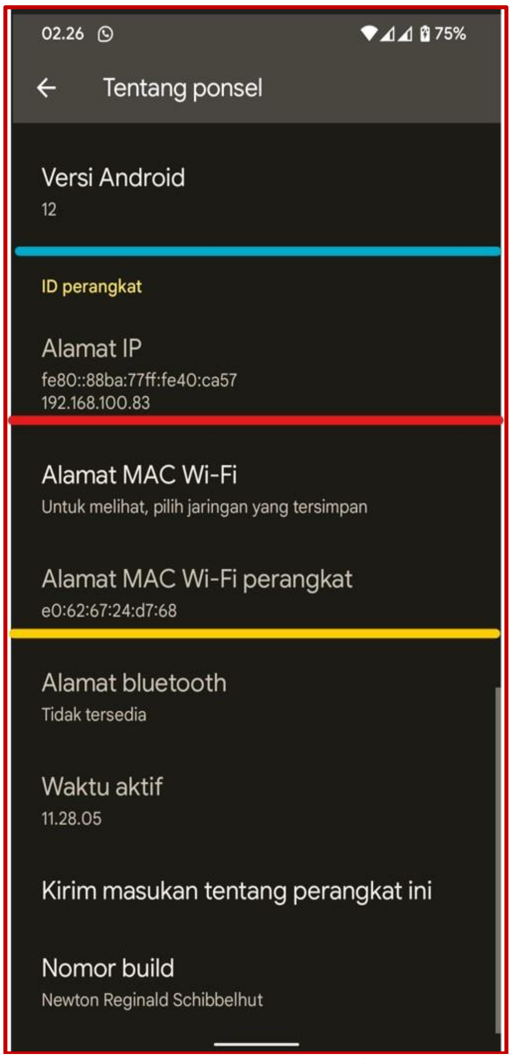
In Android 12, the exploitation process failed because the device IP could not connect to the Ghost Framework. For more details, see Figures 16 and 17.



```
--=[ Ghost Framework 8.0.0
--=[ Developed by EntySec (https://entysec.netlify.app/)

(ghost)> help

Core Commands:

    Command       Description
    -------       -----------
    connect       Connect device.
    devices       Show connected devices.
    disconnect    Disconnect device.
    exit          Exit Ghost Framework.
    help          Show available commands.
    interact      Interact with device.

(ghost)> connect 192.168.100.83 8080
[*] Connecting to 192.168.100.83 ...
[-] Failed to connect to 192.168.100.83!
(ghost)> connect 192.168.100.83
[*] Connecting to 192.168.100.83 ...
[-] Failed to connect to 192.168.100.83!
(ghost)> connect 192.168.100.83
```

Figure 16. Connecting to IP address ADB on AOSP Android 12

In Figure 16, it can be seen that the IP on the Android 12 device failed to connect to the Ghost Framework in the exploitation process. This causes the session to not open so it can't post-exploitation.



02.26 75%

Tentang ponsel

Versi Android
12

ID perangkat

Alamat IP
fe80::88ba:77ff:fe40:ca57
192.168.100.83

Alamat MAC Wi-Fi
Untuk melihat, pilih jaringan yang tersimpan

Alamat MAC Wi-Fi perangkat
e0:62:67:24:d7:68

Alamat bluetooth
Tidak tersedia

Waktu aktif
11.28.05

Kirim masukan tentang perangkat ini

Nomor build
Newton Reginald Schibbelhut

Figure 17. IP address & version of Android

Figure 17 shows a screenshot of the phone on Android 12. The blue color shows the Android version used, namely Android 12. The red color shows the IP Address of the device that will be used to connect to the Ghost Framework. Next, the yellow color indicates the MAC Address of the device.

As a comparison, the test results on Android 12 can be seen in Table 4.

Table 4. List of Activities Android 12.0

| Activity | Result | Description |
|---|---|---|
| **Connect to Device** | False | Connecting IP device to Ghost Framework |
| **Access system info** | False | Obtain system information |
| **Access battery info** | False | Obtain test battery information |
| **Access the system directory** | False | Enter the system directory<br>Change the contents of the system |
| **Access the website** | False | Open URL |
| **Screen shoot Device** | False | Take Screenshot |

It can be seen in Table 4 that the connection process between the test device and the Ghost Framework did not work because Android 12 by default blocks wireless debugging processes therefore the session does not open and causes the post-exploitation process to fail. Remote access trojan which is the purpose of the research cannot be done through ADB on Android 12 because smartphone vendors lock the wireless debugging process on their devices.

## *3.2 Test Comparison Table*

A comparison of the results of tests carried out on three devices can be seen in Table 5.

Table 5. A Comparison of Test Device Results

| Activity | Redmi Note 5 Android 9 (AOSP) | Redmi Note 5 Android 12 (AOSP) | Redmi 5 Plus Android 8.1 (MIUI) | Vivo Y71 Android 8.1 (Funtouch OS) |
|---|---|---|---|---|
| **Connect to Device** | True | False | False | False |
| **Access system info** | True | False | False | False |
| **Access battery info** | True | False | False | False |
| **Access the system directory** | True | False | False | False |
| **Access the website** | True | False | False | False |
| **Screen shoot Device** | True | False | False | False |
| **Remote device** | True | False | False | False |

It can be seen in Table 5 that the exploitation and post-exploitation process was successfully carried out on the Xiaomi Redmi Note 5 Android 9 Based AOSP ROM. Other devices cannot be successfully connected to Ghost Framework and cannot be exploited because device vendors and MIUI & FuntouchOS based ROMs by default block access to wireless debugging.

## 4 CONCLUSION

This research succeeded in showing that the test device with the Android 9.0 version and using the AOSP ROM has a hole in ADB which makes the device vulnerable to attack by Remote Access Trojan. The use of commands in the Ghost Framework and ADB shell commands allows third parties to perform activities on the device such as opening the lock screen, entering system directories, and changing systems that have been exploited. Therefore, anticipation and patching of these holes are necessary for security purposes. However, on devices using MIUI with the Android 8.1 version and devices using FuntouchOS with 8.1 version, exploitation and post-exploitation cannot also be done because smartphone vendors by default block wireless debug access. It is hoped that this research can be a reference for future mobile security research.

## AUTHOR'S CONTRIBUTION

Conducting a literature study of previous research, system design, data collection, analysis and interpretation.

## COMPETING INTERESTS

This paper is my original work. made as a requirement to complete a master's study in informatics engineering.

## ACKNOWLEDGMENT

Thanks to Universitas Ahmad Dahlan Yogyakarta for the support and facilitation of this research.

## REFERENCES

[1] N. A. Handoyono and R. Rabiman, "Development of android-based learning application in EFI materials for vocational schools," in *Journal of Physics: Conference Series*, Feb. 2020, vol. 1456, no. 1, p. 12050, DOI: 10.1088/1742-6596/1456/1/012050.

[2] R. Mayrhofer, J. Vander Stoep, C. Brubaker, and N. Kralevich, "The Android Platform Security Model," *ACM Trans. Priv. Secur.*, vol. 24, no. 3, 2021, doi: 10.1145/3448609.

[3] S. O'Dea, "Number of smartphone users worldwide from 2016 to 2023," *statista.com*, 2021. https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/ (accessed Apr. 29, 2021).

[4] R. Singh, "An Overview of Android Operating System and Its Security Features," *Eng. Res. Appl.*, vol. 4, no. 2, pp. 519–521, 2014.

[5] A. Sarkar, A. Goyal, D. Hicks, D. Sarkar, and S. Hazra, "Android Application Development: A Brief Overview of Android Platforms and Evolution of Security Systems," *Proc. 3rd Int. Conf. I-SMAC IoT Soc. Mobile, Anal. Cloud, I-SMAC 2019*, pp. 73–79, 2019, DOI: 10.1109/I-SMAC47947.2019.9032440.

[6] "Android Open Source Project." https://source.android.com/ (accessed Apr. 30, 2021).

[7] R. D. Putra and I. Mardianto, "Exploitation with Reverse_tcp Method on Android Device using Metasploit," *J. Edukasi dan Penelit. Inform.*, vol. 5, no. 1, p. 106, 2019, doi: 10.26418/jp.v5i1.26893.

[8] A. Amin, A. Eldessouki, M. T. Magdy, N. Abdeen, H. Hindy, and I. Hegazy, "AndroShield: Automated Android Applications Vulnerability Detection, a Hybrid Static and Dynamic Analysis Approach," *Information*, vol. 10, no. 10, p. 326, 2019, DOI: 10.3390/info10100326.

[9] P. Bhat and K. Dutta, "A survey on various threats and current state of security in android platform," *ACM Comput. Surv.*, vol. 52, no. 1, 2019, DOI: 10.1145/3301285.

[10] A. Iswardani and I. Riadi, "Denial of service log analysis using density K-means method," *J. Theor. Appl. Inf. Technol.*, vol. 83, no. 2, pp. 299–302, 2016.

[11] T. Kinnunen *et al.*, "The ASVspoof 2017 challenge: Assessing the limits of replay spoofing attack detection," *Proc. Annu. Conf. Int. Speech Commun. Assoc. INTERSPEECH*, vol. 2017-August, pp. 2–6, 2017, DOI: 10.21437/Interspeech.2017-1111.

[12] D. Kiwia, A. Dehghantanha, K. K. R. Choo, and J. Slaughter, "A cyber kill chain based taxonomy of banking Trojans for evolutionary computational intelligence," *J. Comput. Sci.*, vol. 27, pp. 394–409, 2018, DOI: 10.1016/j.jocs.2017.10.020.

[13] "Android Security Bulletins | Android Open Source Project." https://source.android.com/security/bulletin (accessed Apr. 30, 2021).

[14] "How Monthly Android Security Patch Updates Work." https://www.xda-developers.com/how-android-security-patch-updates-work/ (accessed May 03, 2021).

[15] J. Wu and M. Yang, "LaChouTi: Kernel vulnerability responding framework for the fragmented android devices," *Proc. ACM SIGSOFT Symp. Found. Softw. Eng.*, vol. Part F1301, pp. 920–925, 2017, DOI: 10.1145/3106237.3117768.

[16] A. Shanley and M. N. Johnstone, "Selection of penetration testing methodologies: A comparison and evaluation," *Aust. Inf. Secur. Manag. Conf. AISM 2015*, vol. 2015, pp. 65–72, 2015, DOI: 10.4225/75/57b69c4ed938d.

[17] A. Susanto and W. K. Raharja, "Simulation and Analysis of Network Security Performance Using Attack Vector Method for Public Wifi Communication," *Int. J. Informatics Comput. Sci.*, vol. 5, no. 1, pp. 7–15, Mar. 2021, DOI: 10.30865/ijics.v5i1.2764.

[18] H. Lu, X. Helu, C. Jin, Y. Sun, M. Zhang, and Z. Tian, "Salaxy: Enabling USB Debugging Mode Automatically to Control Android Devices," *IEEE Access*, vol. 7, pp. 178321–178330, 2019, DOI: 10.1109/ACCESS.2019.2958837.

[19] C. Guo, Z. Song, Y. Ping, G. Shen, Y. Cui, and C. Jiang, "PRATD: A Phased Remote Access Trojan Detection Method with Double-Sided Features," *Electronics*, vol. 9, no. 11, p. 1894, Nov. 2020, DOI: 10.3390/electronics9111894.

[20] "exploit - Definition." https://www.trendmicro.com/vinfo/us/security/definition/exploit (accessed May 04, 2021).

[21] D. Jiang and K. Omote, "An approach to detect remote access trojan in the early stage of communication," in *Proceedings - International Conference on Advanced Information Networking and Applications, AINA*, Apr. 2015, vol. 2015-April, pp. 706–713, DOI: 10.1109/AINA.2015.257.

[22] I. M. M. Matin and B. Rahardjo, "Malware Detection Using Honeypot and Machine Learning," Nov. 2019, DOI: 10.1109/CITSM47753.2019.8965419.

[23] R. N. Manda Vy and H. Z. T. R. Zafimarina Stefana, "Bridge implementation between IP network and GSM network," *2013 World Congr. Comput. Inf. Technol. WCCIT 2013*, vol. 4, no. 2, pp. 69–74, 2013, DOI: 10.1109/WCCIT.2013.6618671.

[24] "ADB Shell Commands List and Cheat Sheet." https://technastic.com/adb-shell-commands-list/ (accessed Apr. 30, 2021).

[25] "About Post-Exploitation | Metasploit Documentation." https://docs.rapid7.com/metasploit/about-post-exploitation/ (accessed Nov. 08, 2021).

[26] R. Umar, I. Riadi, and R. S. Kusuma, "Analysis of Conti Ransomware Attack on Computer Network with Live Forensic Method," *IJID (International J. Informatics Dev.*, vol. 10, no. 1, pp. 53–61, 2021, DOI: 10.14421/ijid.2021.2423.