Dhaval Patel

M.Sc Cyber Security, Sem 2

032200300002034

FRIDA

Frida is an open-source dynamic instrumentation toolkit primarily used for reverse engineering,
debugging, and analyzing the behavior of software applications. Frida can be operated on multiple platforms, including Windows, Linux, Android and so on. Frida allows to inject custom
code into running processes, enabling them to intercept and modify function calls, explore the memory of the target process, and monitor and manipulate various aspects of its execution. Frida
is often used for bypassing security mechanisms, performing dynamic analysis of malware, analyzing and modifying the behavior of mobile apps, and developing patches or modifications
for software.

Installation:

● Open command prompt on the computer.
● Ensure that we have Python installed on the system.
● Install Frida using pip, the Python package manager. Run the command :
'pip install frida'.
● Wait for the installation to complete. Pip will download the necessary packages for Frida.
● Once the installation is finished, Frida will be successfully installed on the system.

Python can be used to install frida in your system.
Command:- pip3 install frida-tools
To use frida we have to install the frida server in our android mobile.
Commands to install frida server in android using adb.

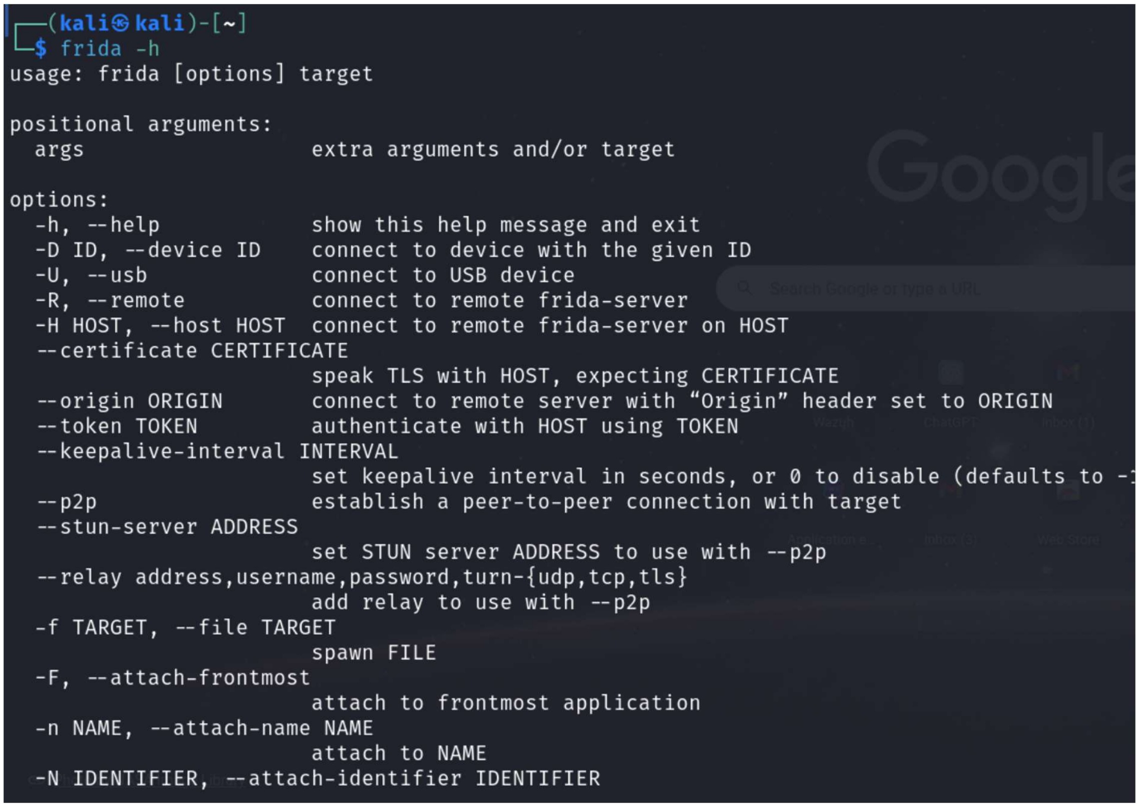
```
adb root
adb push frida-server /data/local/tmp/
adb shell "chmod 755 /data/local/tmp/frida-server"
adb shell "/data/local/tmp/frida-server &"
```

Steps to use frida:-

1) Set up Frida Server on the Target Device:- To interact with a mobile application on a target
device we need to install the Frida server on it.
2) Choose a Scripting Language:- Decide on the scripting language you want to use with Frida.
Frida primarily supports JavaScript, but it also offers bindings for other languages like Python and
Swift.

3) Connect your development machine to the target device or emulator running the Frida server.
You can establish a connection using various methods, such as USB, Wi-Fi, or remote connections.
4) Use the Frida client to inject the Frida script into the target application. The script injection process varies slightly between platforms.
5) For Android: Use the frida-ps -U command to list the running processes on the device. Identify
the process ID (PID) of the target application. Then, use the **frida -U -l <script.js> -f <package_name>** command to inject the script into the target application.

Figures:-

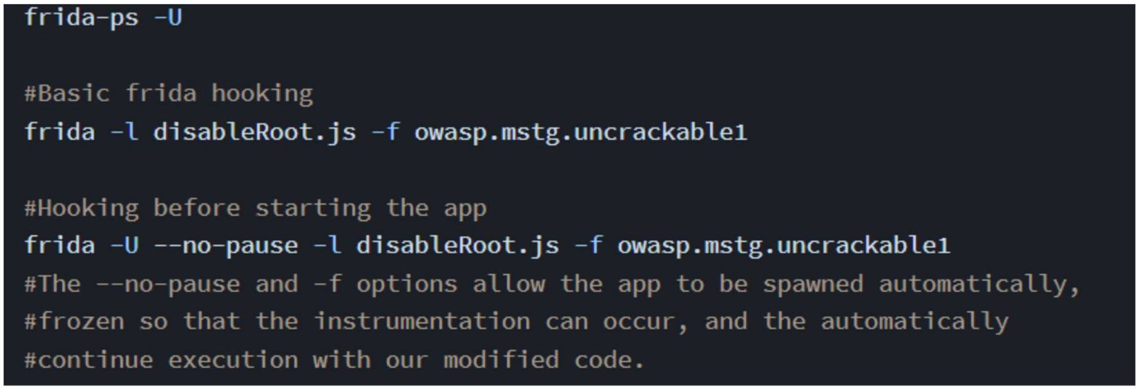


```
┌──(kali㉿kali)-[~]
└─$ frida -h
usage: frida [options] target

positional arguments:
  args                  extra arguments and/or target

options:
  -h, --help            show this help message and exit
  -D ID, --device ID    connect to device with the given ID
  -U, --usb             connect to USB device
  -R, --remote          connect to remote frida-server
  -H HOST, --host HOST  connect to remote frida-server on HOST
  --certificate CERTIFICATE
                        speak TLS with HOST, expecting CERTIFICATE
  --origin ORIGIN       connect to remote server with "Origin" header set to ORIGIN
  --token TOKEN         authenticate with HOST using TOKEN
  --keepalive-interval INTERVAL
                        set keepalive interval in seconds, or 0 to disable (defaults to -
  --p2p                 establish a peer-to-peer connection with target
  --stun-server ADDRESS
                        set STUN server ADDRESS to use with --p2p
  --relay address,username,password,turn-{udp,tcp,tls}
                        add relay to use with --p2p
  -f TARGET, --file TARGET
                        spawn FILE
  -F, --attach-frontmost
                        attach to frontmost application
  -n NAME, --attach-name NAME
                        attach to NAME
  -N IDENTIFIER, --attach-identifier IDENTIFIER
```

Fig 1: various options available in Friday



```
frida-ps -U

#Basic frida hooking
frida -l disableRoot.js -f owasp.mstg.uncrackable1

#Hooking before starting the app
frida -U --no-pause -l disableRoot.js -f owasp.mstg.uncrackable1
#The --no-pause and -f options allow the app to be spawned automatically,
#frozen so that the instrumentation can occur, and the automatically
#continue execution with our modified code.
```

Fig 2: using frida from command-line

Advantages:

Cross-platform support: Frida supports multiple platforms, including Windows, macOS, Linux and Android. This cross-platform compatibility allows us to analyze and manipulate applications on various operating systems.

Interoperability: Frida offers seamless interoperability with other tools and frameworks commonly used in the field of security research and reverse engineering. By combining Frida with tools like Burp Suite, IDA Pro, Radare2, and more, can enhance the analysis capabilities and integrate Frida into existing workflows.

JavaScript-based scripting: Frida provides a JavaScript API that allows to write scripts to interact with the target application.

Dynamic instrumentation: Frida enables us to perform runtime manipulation and dynamic instrumentation of running processes. We can inject custom code into processes, hook function calls, modify behavior on-the-fly, and analyze the application's execution in real-time.

Disadvantages:

Detection: As a dynamic instrumentation framework, Frida can potentially be detected by anti-debugging and anti-tampering techniques employed by the target application.

Limited visibility in obfuscated code: If the target application employs strong code obfuscation techniques, Frida may face challenges in effectively analyzing and manipulating the obfuscated code. Complex obfuscation can make it harder to identify relevant functions and data structures, potentially limiting the extent of dynamic analysis that can be performed.

Complexity: Working with Frida can require a certain level of technical expertise and familiarity with concepts like dynamic analysis, reverse engineering, and scripting. Users who are new to these domains may find a learning curve associated with understanding Frida's concepts, APIs, and usage.

Compatibility and stability: The effectiveness and stability of Frida can vary depending on the target application, operating system, and device. Not all applications may be fully compatible with Frida, and certain limitations may arise when working with specific platforms or versions.

QARK

Qark stands for "Quick Android Review Kit" and is an open-source tool designed for Android
application security testing and static analysis. Qark aims to assist developers and security researchers in identifying potential security vulnerabilities and privacy issues in Android apps.

Qark performs static analysis on the APK (Android Application Package) files of Android apps
to detect security vulnerabilities and provide a comprehensive report of potential risks. It combines multiple security scanning techniques and checks against known security best practices
to identify common security issues that can impact the privacy and security of Android applications.

Some of the key features and checks performed by Qark include:

Permission-related issues: Qark examines the permissions requested by the app and identifies any potential overprivileged or underprivileged permissions. It also checks for dangerous permissions that may pose a risk to user privacy or security.

Exported components: Qark examines exported components like activities, services, and broadcast receivers, and identifies any potential security risks associated with them, such as unauthorized access or misuse.

Code-related vulnerabilities: Qark analyzes the app's code to identify common security vulnerabilities, such as insecure data storage, input validation issues, insecure communication (HTTP instead of HTTPS), improper use of cryptographic functions, and more.

Intent-related vulnerabilities: Qark checks for potential security issues related to the usage of Android Intents, such as insecure intent handling, intent spoofing, or intent injection vulnerabilities.

WebView vulnerabilities: Qark inspects the usage of WebView within the app and looks for potential vulnerabilities, such as JavaScript injection, insecure WebView settings, or potential vulnerabilities in the WebView implementation.

Third-party libraries: Qark scans for the usage of known vulnerable third-party libraries within
the app and provides information about any associated security risks.

Installation:

- Ensure that we have Python installed on the system.
- Open a terminal or command prompt on your computer.
- Install Qark using pip, the Python package manager. Run the command:

‘pip install qark’.

- Wait for the installation to complete. Pip will download and install the necessary

1) Using python

```
pip3 install –user qark
```

2) Using github

```
git clone https://github.com/linkedin/qark.git
cd qark
pip install -r requirements.txt
pip install . –
```

How to use Qark:

To use Qark, follow these steps:

- Ensure Qark is installed on the system.
- Open command prompt.
- Navigate to the directory where APK file is located. Use the cd command to change directories in the terminal.
- Run Qark by executing the command: ‘qark --apk (path)’.
- Qark will start analyzing the APK file and perform various security checks.
- Once the analysis is complete, Qark will generate a report detailing any security vulnerabilities, privacy issues, or best practice violations it detected in the Android application.
- Review the Qark report to understand the identified issues. The report will provide information on

Running Qark on Android application:-

Running qark on your Android apps is quick and easy.
Simply navigate to the directory where your app is installed and use the command "python qark.py
<path_to_apk_file>" to start the analysis. qark will then produce a detailed report of potential vulnerabilities and security risks within your app.

Output And Analysis:-

qark produces an HTML report that highlights each potential vulnerability and explains the steps to
remediate it. This report includes a detailed analysis of the security risks present in your app and
includes a scorecard indicating the level of risk your app faces.
qark analysis can be tailored to your specific needs, whether you want to focus on one specific issue or analyze all potential vulnerabilities comprehensively. In either case, you'll be left
with a detailed understanding of your app's security posture.

Limitation:-

qark is a powerful tool, but it isn't without its limitations and challenges. One
limitation is that it is limited to Android applications, so it cannot be used to analyze other types of
software. Additionally, it may not always be able to detect very specific vulnerabilities or issues
with a particular application.