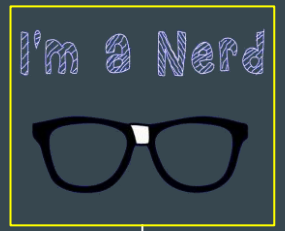
Android reverse-engineering

•••

find a trojan information in a closed source apps

Chi sono



Fabio Zito (@ZiteLOG)

Che cos’è il reverse-engineering?

Reverse engineering (in italiano "ingegneria inversa", "ingegnerizzazione inversa") è un anglicismo che indica quell'insieme di analisi delle funzioni, degli impieghi, della collocazione, dell'aspetto progettuale, geometrico e materiale di un manufatto o di un oggetto che è stato rinvenuto (ad esempio un reperto, un dispositivo, componente elettrico, un meccanismo, software). (Wikipedia)

Sviluppo

1

- Spaghetti 12 OZ
- Guanciale 4 OZ
- Very fresh egg yolks 4
- Aged Pecorino Romano cheese
- Grated e TBSP Grana Padano cheese
- Grated salt black pepper
- **Secret Sauce**

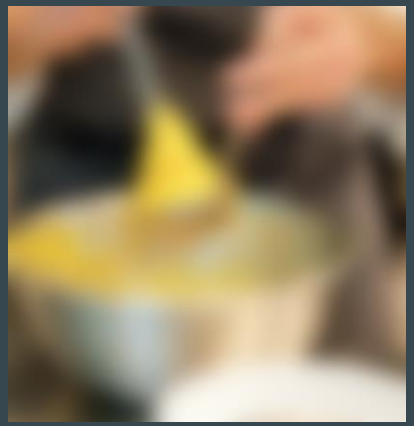
2

3

Reverse-Engineering

3

2



1

- [illegible]
- [illegible]
- [illegible]
- [illegible]
- [illegible]
- [illegible]
- [illegible]
- [illegible]

Ma a cosa serve?

audit/modify/malware analysis...

Che cos'è la Malware Analysis?

La Malware Analysis è un ramo della Cyber Security che si occupa di scovare e neutralizzare minacce informatiche quali virus, worm, bot, rootkit, spyware, backdoor, Trojan Horse, etc.

Ma a cosa serve esattamente quando si parla di Trojan?

A rispondere a domande come:

- Sono Spiato?
- Chi mi sta spiando e come ha fatto ad infettarmi?
- Dove vanno a "finire" i miei dati?

Attraverso l'identificazione:

- Del malware.
- Delle modalità di inoculazione.
- Delle informazioni acquisite e della loro destinazione finale

Che cos'è un Trojan?

Il Trojan è un programma maligno mascherato da qualcosa di benigno (il nome infatti deriva da Cavallo di Troia, in esplicito riferimento alla leggenda Greca). Sono scritti principalmente per sottrarre informazioni e esercitare controllo sul dispositivo infettato attraverso

due componenti:

- **Un client** installato sul dispositivo dell'utente che si occupa di inviare dati e/o ricevere comandi.

- **Uno o più server** che si occupano di ricevere e memorizzare i dati inviati dal client e inviare comandi verso quest'ultimo.

Cose da conoscere prima di iniziare

APK

L'estensione APK indica un file Android Package. Questo formato di file, una variante del formato .JAR, è utilizzato per la distribuzione e l'installazione di componenti in dotazione sulla piattaforma per dispositivi mobili Android. (Wikipedia)

Cartelle:

META-INF
res

File:

AndroidManifest.xml
classes.dex
resources.arsc

Componenti:

Activity
Receiver
Service
Provider

Compilare: Source Code → DEX

```
public class ComponentActivity extends v4 implements v9, ia, oa, c {
    public ha e;
    public int g;
    public final w9 c = new w9(this);
    public final na d = new na(this);
    public final OnBackPressedDispatcher f = new OnBackPressedDispatcher(new a());

    /* loaded from: classes.dex */
    public class a implements Runnable {
        public a() {
            ComponentActivity.this = r1;
        }

        @Override // java.lang.Runnable
        public void run() {
            ComponentActivity.super.onBackPressed();
        }
    }

    /* loaded from: classes.dex */
    public static final class b {
        public ha a;
    }

    public ComponentActivity() {
        if (a() != null) {
            if (Build.VERSION.SDK_INT >= 19) {
```

```
$ xxd -u classes.dex | tail -30
00369f10: C0AE 3500 5F4B 0000 C8AE 3500 604B 0000  ..5._K....5.`K..
00369f20: D0AE 3500 614B 0000 D8AE 3500 624B 0000  ..5.aK....5.bK..
00369f30: E0AE 3500 E8AE 3500 0000 0000 0000 0000  ..5...5.........
00369f40: 0000 0000 00AF 3500 0000 0000 0100 0000  ......5.........
00369f50: 0000 0000 344C 0000 F8AE 3500 0CAF 3500  ....4L....5...5.
00369f60: 0000 0000 0000 0000 0000 0000 1CAF 3500  ..............5.
00369f70: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00369f80: 0000 0000 0100 0000 0000 0000 7C61 0000  ............|a..
00369f90: 607D 3500 0000 0000 0000 0000 0100 0000  `}5.............
00369fa0: 0000 0000 A561 0000 607D 3500 0000 0000  .....a..`}5.....
00369fb0: 0000 0000 0100 0000 0000 0000 AB61 0000  .............a..
00369fc0: 782D 3500 2CAF 3500 0000 0000 0000 0000  x-5.,.5.........
00369fd0: 0000 0000 34AF 3500 0000 0000 0000 0000  ....4.5.........
00369fe0: 0000 0000 3CAF 3500 0000 0000 0000 0000  ....<.5.........
00369ff0: 0000 0000 44AF 3500 0000 0000 0000 0000  ....D.5.........
0036a000: 0000 0000 0000 0000 0000 0000 0100 0000  ................
0036a010: 0000 0000 B061 0000 607D 3500 1100 0000  .....a..`}5.....
0036a020: 0000 0000 0100 0000 0000 0000 0100 0000  ................
0036a030: D07D 0000 7000 0000 0200 0000 AA11 0000  .}..p...........
0036a040: B0F7 0100 0300 0000 541A 0000 583E 0200  ........T...X>..
0036a050: 0400 0000 923E 0000 487A 0300 0500 0000  .....>..Hz......
0036a060: AE6D 0000 D86E 0500 0600 0000 210D 0000  .m...n......!...
0036a070: 48DC 0800 0120 0000 0B52 0000 6880 0A00  H.... ...R..h...
0036a080: 0320 0000 721B 0000 F0DC 2200 0110 0000  . ..r.....".....
0036a090: AE0F 0000 2C6A 2400 0220 0000 D07D 0000  ....,j$.. ...}..
0036a0a0: E207 2500 0420 0000 870E 0000 C309 3100  ..%.. ........1.
0036a0b0: 0020 0000 7B0C 0000 0011 3200 0520 0000  . ..{.....2.. ..
0036a0c0: F501 0000 A4CB 3400 0310 0000 BE0D 0000  ......4.........
0036a0d0: 1425 3500 0620 0000 0C09 0000 4CAF 3500  .%5.. ......L.5.
0036a0e0: 0010 0000 0100 0000 1CA0 3600            ..........6.
```

Disassemblare: DEX → Smali

```
$ xxd -u classes.dex | tail -30
00369f10: C0AE 3500 5F4B 0000 C8AE 3500 604B 0000  ..5._K....5.`K..
00369f20: D0AE 3500 614B 0000 D8AE 3500 624B 0000  ..5.aK....5.bK..
00369f30: E0AE 3500 E8AE 3500 0000 0000 0000 0000  ..5...5.........
00369f40: 0000 0000 00AF 3500 0000 0000 0100 0000  ......5.........
00369f50: 0000 0000 344C 0000 F8AE 3500 0CAF 3500  ....4L....5...5.
00369f60: 0000 0000 0000 0000 0000 0000 1CAF 3500  ..............5.
00369f70: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00369f80: 0000 0000 0100 0000 0000 0000 7C61 0000  ............|a..
00369f90: 607D 3500 0000 0000 0000 0000 0100 0000  `}5.............
00369fa0: 0000 0000 A561 0000 607D 3500 0000 0000  .....a..`}5.....
00369fb0: 0000 0000 0100 0000 0000 0000 AB61 0000  .............a..
00369fc0: 782D 3500 2CAF 3500 0000 0000 0000 0000  x-5.,.5.........
00369fd0: 0000 0000 34AF 3500 0000 0000 0000 0000  ....4.5.........
00369fe0: 0000 0000 3CAF 3500 0000 0000 0000 0000  ....<.5.........
00369ff0: 0000 0000 44AF 3500 0000 0000 0000 0000  ....D.5.........
0036a000: 0000 0000 0000 0000 0000 0000 0100 0000  ................
0036a010: 0000 0000 B061 0000 607D 3500 1100 0000  .....a..`}5.....
0036a020: 0000 0000 0100 0000 0000 0000 0100 0000  ................
0036a030: D07D 0000 7000 0000 0200 0000 AA11 0000  .}..p...........
0036a040: B0F7 0100 0300 0000 541A 0000 583E 0200  ........T...X>..
0036a050: 0400 0000 923E 0000 487A 0300 0500 0000  .....>..Hz......
0036a060: AE6D 0000 D86E 0500 0600 0000 210D 0000  .m...n......!...
0036a070: 48DC 0800 0120 0000 0B52 0000 6880 0A00  H.... ...R..h...
0036a080: 0320 0000 721B 0000 F0DC 2200 0110 0000  . ..r.....".....
0036a090: AE0F 0000 2C6A 2400 0220 0000 D07D 0000  ....,j$.. ...}..
0036a0a0: E207 2500 0420 0000 870E 0000 C309 3100  ..%.. ........1.
0036a0b0: 0020 0000 7B0C 0000 0011 3200 0520 0000  . ..{.....2.. ..
0036a0c0: F501 0000 A4CB 3400 0310 0000 BE0D 0000  ......4.........
0036a0d0: 1425 3500 0620 0000 0C09 0000 4CAF 3500  .%5.. ......L.5.
0036a0e0: 0010 0000 0100 0000 1CA0 3600            ..........6.
```

```
# virtual methods
.method public onCreate(Landroid/os/Bundle;Landroid/os/PersistableBundle;)V
    .locals 0

    .line 1
    sget p1, Landroid/os/Build$VERSION;->SDK_INT:I

    const/16 p2, 0x1c

    if-lt p1, p2, :cond_0

    .line 2
    invoke-virtual {p0}, Landroid/app/Activity;->getWindow()Landroid/view/Window;

    move-result-object p1

    const/high16 p2, 0x4000000

    invoke-virtual {p1, p2}, Landroid/view/Window;->addFlags(I)V

    .line 3
    invoke-virtual {p0}, Landroid/app/Activity;->getWindow()Landroid/view/Window;

    move-result-object p1
```

Offuscamento

Spyme.getConnectionParams() == b.b()

Smali?

The names "smali" and "baksmali" are the Icelandic equivalents of "assembler" and "disassembler" respectively. Why Icelandic you ask? Because dalvik was named for an Icelandic fishing village. (https://github.com/JesusFreke/smali/wiki)

Registri

v0, v1, v2... vN registri locali
p0, p1, p2... pN argomenti (p0 == this)

https://github.com/JesusFreke/smali/wiki/Registers

AndroidManifest.xml

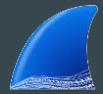
Ogni applicazione Android dev'essere accompagnata da un file chiamato **AndroidManifest.xml** nella sua cartella principale. Il **Manifest** raccoglie informazioni basilari sull'app, informazioni necessarie al sistema per far girare qualsiasi porzione di codice della stessa.

- il nome del package dell'applicazione, che è anche un identificatore univoco della stessa;
- il codice della versione
- la versione del SDK minima per l'utilizzo dell'applicazione
- Descrive le componenti dell'applicazione (attività, servizi, receiver, provider, ecc.), nomina le classi e pubblica le loro "competenze".
- i permessi dell'app, e i permessi necessari alle altre app per interagire con la stessa
- le librerie necessarie all'app

Gli strumenti

 APK Studio,  IntelliJ IDEA,

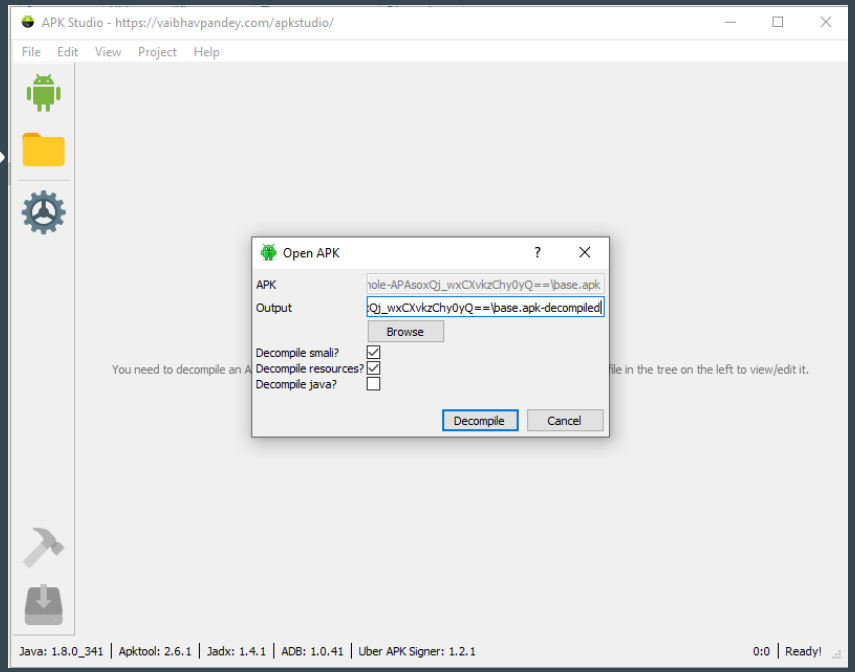
 Android Studio (LLDB), ADB Tools,

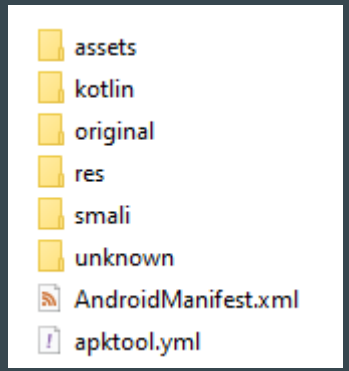
 Wireshark e tcpdump

grep è il nostro migliore amico

Iniziamo!

#1 Disassemblare





#2 Analisi del Manifest

Info base e Permessi

```
<?xml version="1.0" encoding="utf-8"?>
<manifest
    xmlns:android="http://schemas.android.com/apk/res/and
    android:permission="android.permission.RECEIVE_BOOT_C
    android:versionCode="1"
    android:versionName="2.4.2"
    android:allowBackup="false"
    android:installLocation="1"
    android:compileSdkVersion="29"
    android:compileSdkVersionCodename="10"
    package="com.
    platformBuildVersionCode="23"
    platformBuildVersionName="6.0-2438415">

    <uses-sdk
        android:minSdkVersion="16"
        android:targetSdkVersion="22" />

    <uses-permission
        android:name="android.permission.WRITE_CALL_LOG"

    <uses-permission
        android:name="android.permission.REQUEST_INSTALL_

    <uses-permission
        android:name="android.permission.FOREGROUND_SERVI

    <uses-permission
        android:name="android.permission.RECEIVE_BOOT_CO

    <uses-permission
```

Application e Activity

```
<application
    android:theme="@ref/0x7f090000"
    android:label="
    android:icon="@ref/0x01080093"
    android:name="net.        wyxrrwlh.Vgsftpjp"
    android:allowBackup="false"
    android:hardwareAccelerated="false"
    android:largeHeap="true"
    android:supportsRtl="true"
    android:extractNativeLibs="false"
    android:resizeableActivity="true"
```

```
<activity
    android:theme="@ref/0x7f0e010b"
    android:label="@ref/0x7f0d0036"
    android:name="qfv.cpvenz.spmy"
    android:screenOrientation="1">

    <intent-filter>

        <action
            android:name="android.intent.action.MAIN" />

        <category
            android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
```

Receiver, Service e Provider

```
<receiver
    android:name="nlf.ibypoo.ntqn"
    android:enabled="true"
    android:exported="true">

    <intent-filter>

        <action
            android:name="android.intent.action.BOOT_COMPLETED" />

        <action
            android:name="android.intent.action.QUICKBOOT_POWERON" />

        <action
            android:name="com.htc.intent.action.QUICKBOOT_POWERON" />
    </intent-filter>
</receiver>
```

```
<service
    android:name="noc.fybnbt.nsuc">

    <intent-filter>

        <action
            android:name="com.google.firebase.MESSAGING_EVENT" />
    </intent-filter>
</service>
```

```
<provider
    android:name="com.google.firebase.provider.FirebaseInitProvider"
    android:exported="false"
    android:authorities="com        .firebaseinitprovider"
    android:initOrder="100" />
```

#3 Analisi statica

È la «lettura» delle istruzioni contenute nel codice disassemblato per capire come il malware o parte di esso si comporta all'interno del dispositivo infetto. Generalmente per fare questo si individuano ***porzioni di codice di interesse*** *attraverso una serie di ricerche di tipo testuale utilizzando determinate parole chiave. Di seguito alcuni esempi ricordandoci sempre che* ***grep*** è il nostro migliore amico.

Network I/O

il Trojan deve inviare e ricevere dati da un server remoto. Sappiamo che generalmente tali informazioni vengono trasmesse/ricevute tramite protocolli criptati come ad esempio HTTPS, nel quale possono «viaggiare» sia i dati da trasferire dal dispositivo al server (messaggi/registrazione audio e video, etc.), sia le informazioni di «servizio» che il server invia al dispositivo (comandi, programmazioni, etc) quest'ultime generalmente sono in formato JSON.

#4 Analisi dinamica

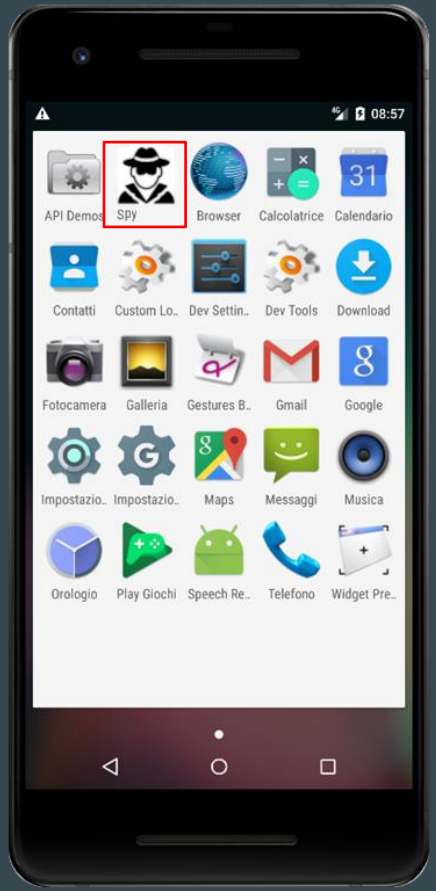
*Consiste nell'installare l'app in un dispositivo diverso (emulatore/muletto) da quello su cui è stato estratto il malware . Lanciarla eseguendola passo, passo utilizzando un software di debugging, in particolare *LLDB del progetto **LLVM (Low Level Virtual Machine) analizzandone man mano i valori contenuti nei registri.*

NB: In qualche caso, tale analisi è possibile solo dopo modifica al codice dissasemblato.

NB1: È utile anche lo sniffing e l'analisi del traffico di rete.

**https://en.wikipedia.org/wiki/LLDB_(debugger)*
***https://it.wikipedia.org/wiki/LLVM*

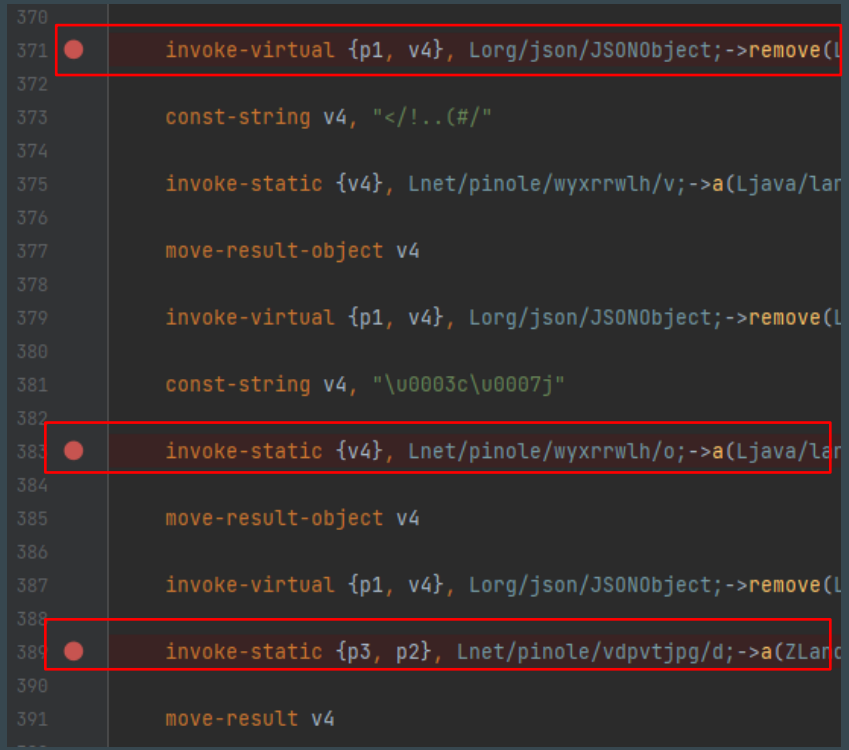
1



Per prima cosa è necessario installare il file "*.apk*" su un dispositivo con caratteristiche simili (emulatore/muletto) a quello da dove è stato individuato e successivamente estratto il Trojan. Questo al duplice obbiettivo di:

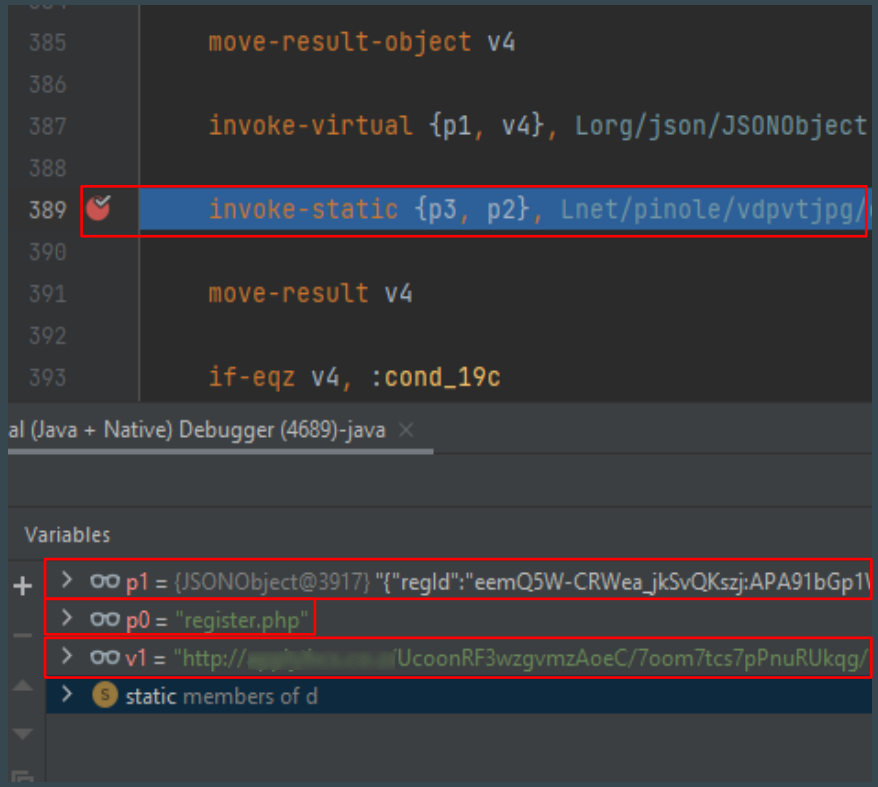
1. non operare alcuna alterazione sul dispositivo originale;
2. consentire la ripetibilità di tutte le operazioni fatte.

2



Una volta installata, è necessario «collegare» il dispositivo al codice dissasemblato (i file *.smali*) attraverso l'IDE utilizzata (Android studio/Intellij IDEA) e qui impostare , nei punti individuati in fase di analisi statica, i *breakpoint* che consentiranno di «bloccare» l'applicazione una volta lanciata.

3



Eseguire l'applicazione "bloccata" un'istruzione alla volta (passo-passo) leggendo il contenuto memorizzato all'interno dei registri, ogni qualvolta viene eseguita un'istruzione fino ad individuare quella di interesse.

Esempio su caso reale

Cambio di IMEI

```
# virtual methods
.method public run()V
    .locals 7

    .line 1
    iget-object v0, p0, Lg/a/a/f/a;->a:Landroid/content/Context;

    .line 2
    new-instance v1, Ljava/util/HashMap;

    invoke-direct {v1}, Ljava/util/HashMap;-><init>()V

    .line 3
    invoke-static {v0}, Lsys/android/core/common/util/Utility;->getIMEI(Landroid/c

    move-result-object v0

    const-string v2, "imei"

    invoke-virtual {v1, v2, v0}, Ljava/util/HashMap;->put(Ljava/lang/Object;Ljava/
```

```
# virtual methods
.method public run()V
    .locals 7

    .line 1
    iget-object v0, p0, Lg/a/a/f/a;->a:Landroid/content/Context;

    .line 2
    new-instance v1, Ljava/util/HashMap;

    invoke-direct {v1}, Ljava/util/HashMap;-><init>()V

    .line 3
    invoke-static {v0}, Lsys/android/core/common/util/Utility;->getIMEI(Landroid/c

    move-result-object v0

    const-string v0, "353539[redacted]"

    const-string v2, "imei"

    invoke-virtual {v1, v2, v0}, Ljava/util/HashMap;->put(Ljava/lang/Object;Ljava/
```

Grazie per l’attenzione