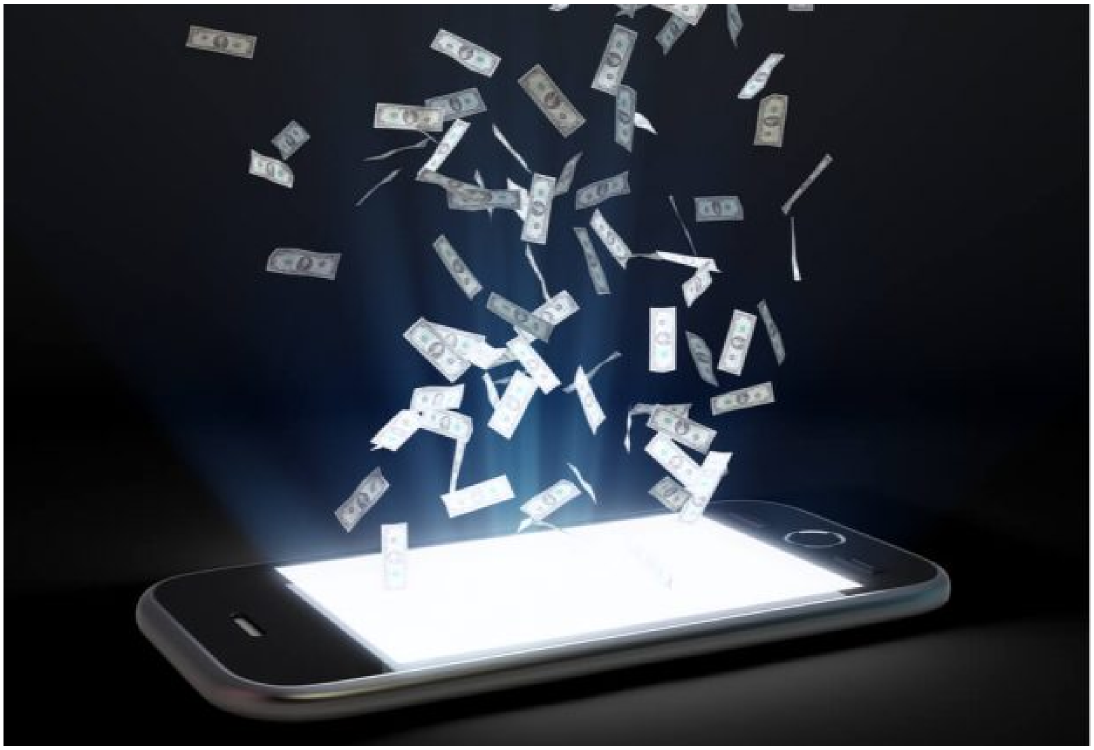
Insidious Android malware gives up all malicious features but one to gain stealth

welivesecurity.com/2020/05/22/insidious-android-malware-gives-up-all-malicious-features-but-one-gain-stealth/

May 22, 2020



ESET researchers detect a new way of misusing Accessibility Service, the Achilles' heel of Android security

Lukas Stefanko

22 May 2020 - 03:00PM

ESET researchers detect a new way of misusing Accessibility Service, the Achilles' heel of Android security

ESET researchers have analyzed an extremely dangerous Android app that can perform a host of nefarious actions, notably wiping out the victim's bank account or cryptocurrency wallet and taking over their email or social media accounts. Called "DEFENSOR ID", the banking trojan was available on Google

Play at the time of the analysis. The app is fitted with standard information-stealing capabilities; however, this banker is exceptionally insidious in that after installation it requires a single action from the victim – enable Android's Accessibility Service – to fully unleash the app's malicious functionality.

The DEFENSOR ID app made it onto the heavily guarded Google Play store thanks to its extreme stealth. Its creators reduced the app's malicious surface to the bare minimum by removing all potentially malicious functionalities but one: abusing Accessibility Service.

Accessibility Service is long known to be the Achilles' heel of the Android operating system. Security solutions can detect it in countless combinations with other suspicious permissions and functions, or malicious functionalities – but when faced with no additional functionality nor permission, all failed to trigger any alarm on DEFENSOR ID.

By "all" we mean all security mechanisms guarding the official Android app store (including the detection engines of the members of the App Defense Alliance) and all security vendors participating in the VirusTotal program (see Figure 1).

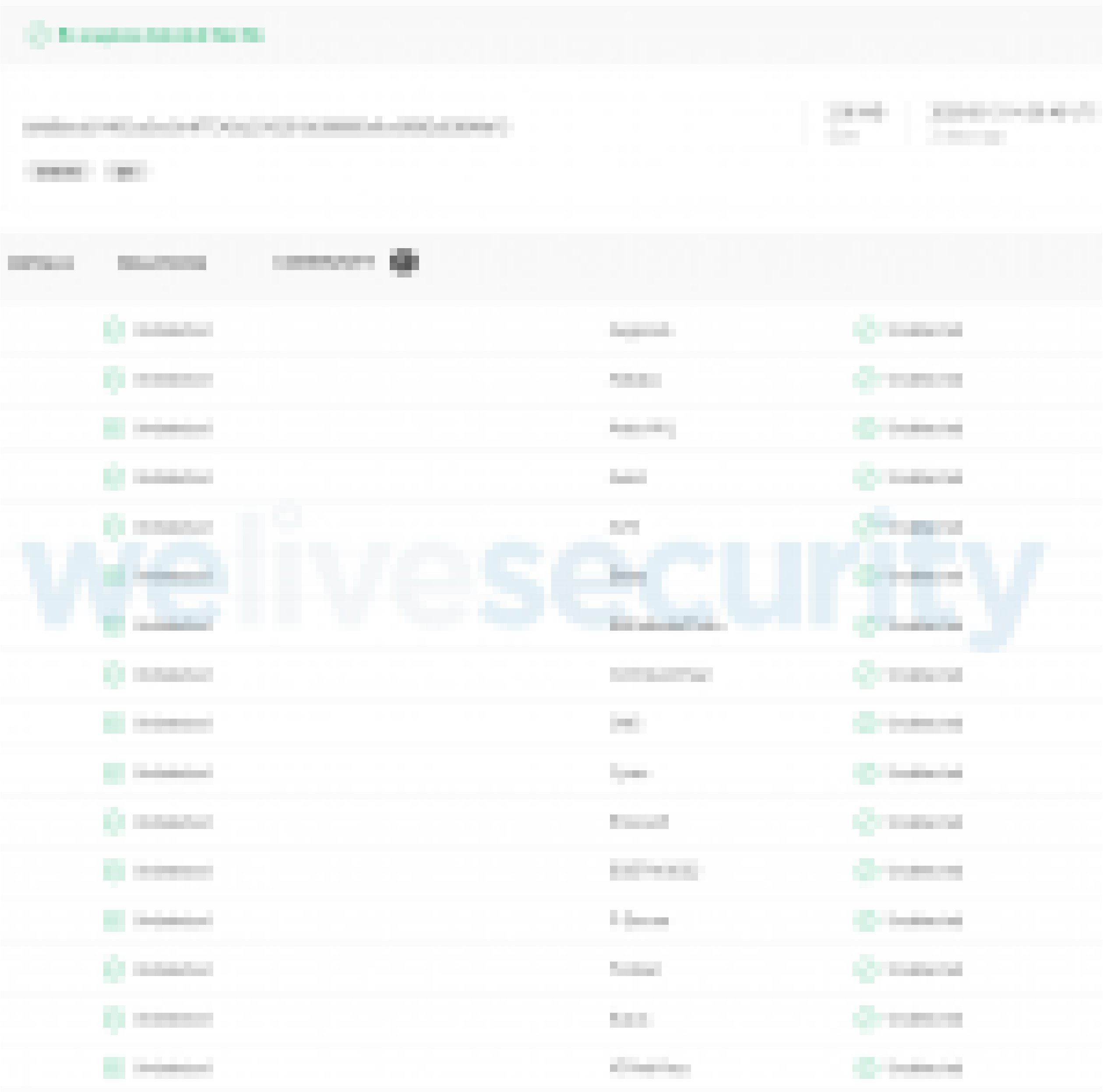


Figure 1. According to the VirusTotal service, no security vendor detected the DEFENSOR ID app until it was pulled off the Play store

DEFENSOR ID was released on Feb 3, 2020 and last updated to v1.4 on May 6, 2020. The latest version is analyzed here; we weren't able to determine if the earlier versions were also malicious. According to its profile at Google Play (see Figure 2) the app reached a mere 10+ downloads. We reported it to Google on May 16, 2020 and since May 19, 2020 the app has no longer been available on Google Play.

The developer name used, GAS Brazil, suggests the criminals behind the app targeted Brazilian users. Apart from including the country's name, the app's name is probably intended to imply a relationship with the antifraud solution named GAS Tecnologia. That security software is commonly installed on computers in Brazil as several banks require it to log into their online banking. However, there is also an English version of the DEFENSOR ID app (see Figure 3) besides the Portuguese one, and that app has neither geographical nor language restrictions.

Playing further off the suggested GAS Tecnologia link, the app promises better security for its users. The description in Portuguese promises more protection for the user's applications, including end-to-end encryption. Deceptively, the app was listed in the Education section.

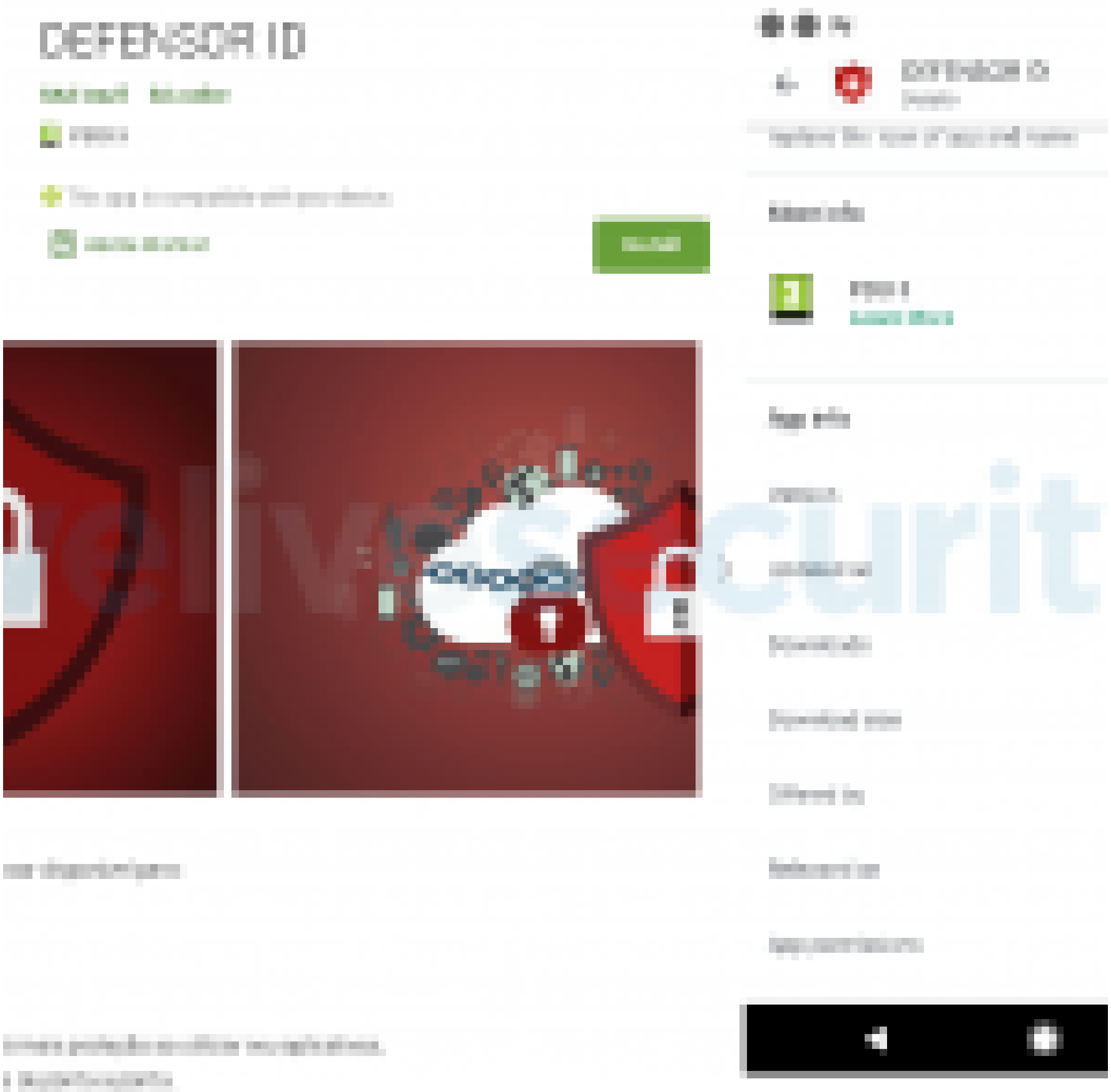


Figure 2. The DEFENSOR ID app on Google Play – Portuguese version (translates roughly as: "Your new Defensor app available for: / Individuals / Legal entities / From now on you will have more protection when using your applications, encryption for end-to-end users")

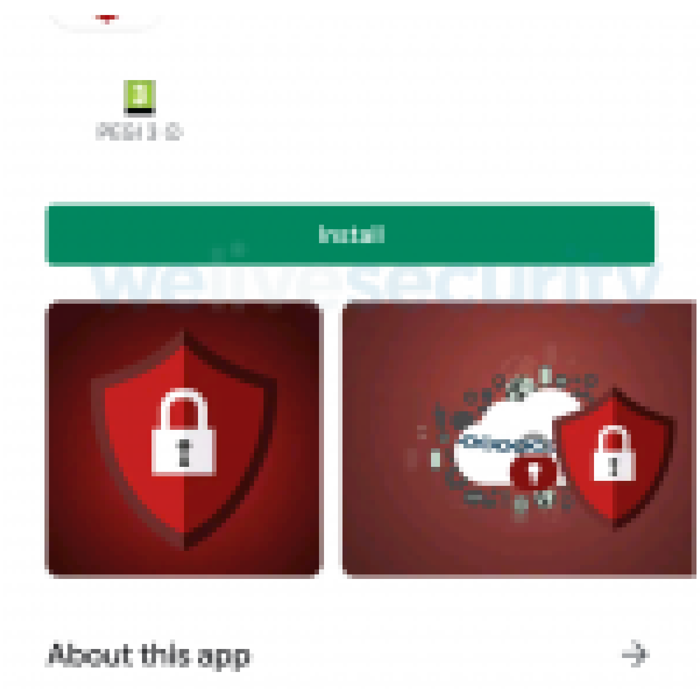


Figure 3. The DEFENSOR ID app on Google Play – English version

Functionality

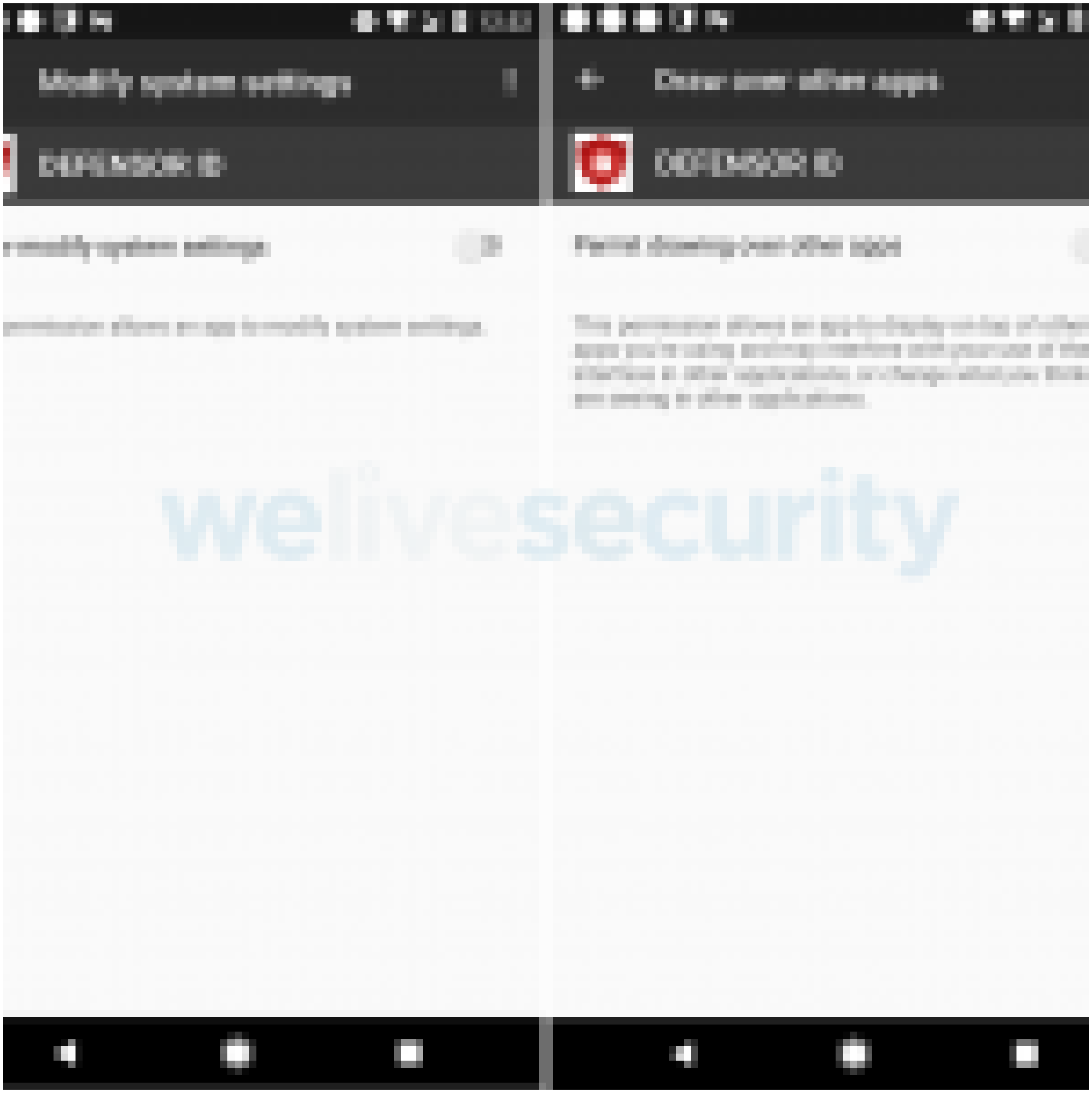
After starting, DEFENSOR ID requests the following permissions:

- allow modify system settings
- permit drawing over other apps, and
- activate accessibility services.

If an unsuspecting user grants these permissions (see Figure 4), the trojan can read any text displayed in any app the user may launch – and send it to the attackers. This means the attackers can steal the victim's credentials for logging into apps, SMS and email messages, displayed cryptocurrency private keys, and even software-generated 2FA codes.

The fact the trojan can steal both the victim's credentials and also can control their SMS messages and generated 2FA codes means DEFENSOR ID's operators can bypass two-factor authentication. This opens the door to, for example, fully controlling the victim's bank account.

To make sure the trojan survives a device restart, it abuses already activated accessibility services that will launch the trojan right after start.



Modify system settings

DEFENSOR ID

modify system settings

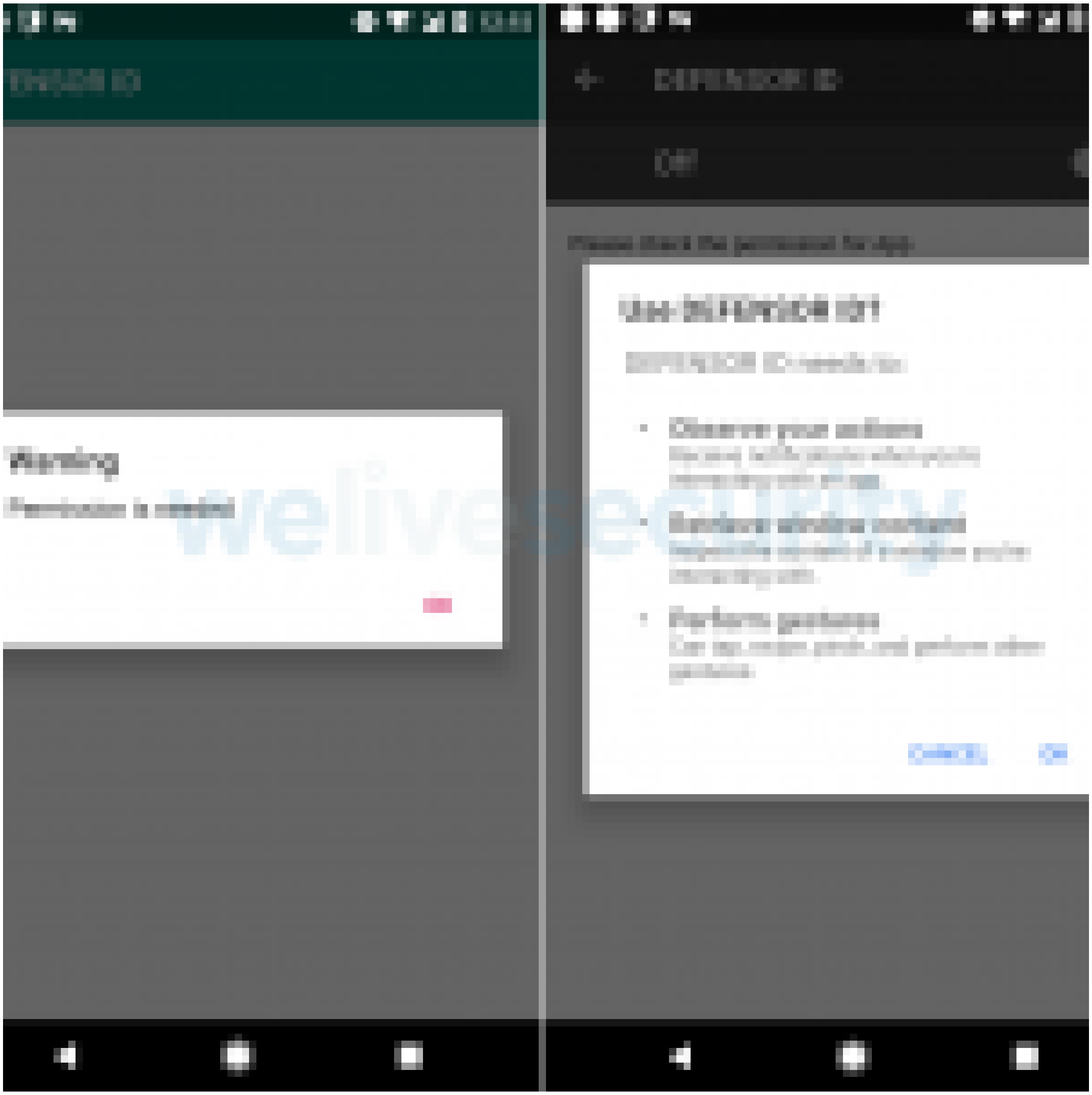
permission allows an app to modify system settings.

Draw over other apps

DEFENSOR ID

Permit drawing over other apps

This permission allows an app to display on top of other apps you're using and may interfere with your use of the interface in other applications, or change what you think you are seeing in other applications.



Warning

Permission is needed

Please check the permission for app

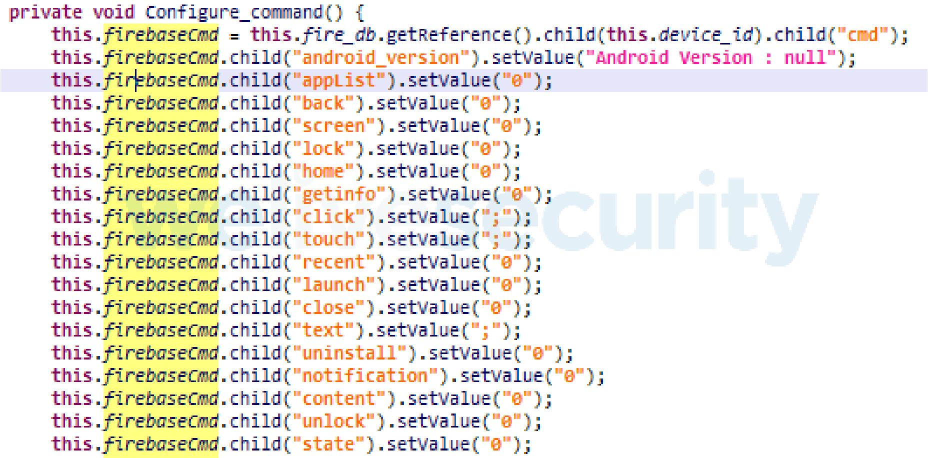
Use DEFENSOR ID?

DEFENSOR ID needs to:

- Observe your actions
  Receive notifications when you're interacting with an app.
- Retrieve window content
  Inspect the content of a window you're interacting with.
- Perform gestures
  Can tap, swipe, pinch, and perform other gestures.

Figure 4. The permission requests by DEFENSOR ID

Our analysis shows the DEFENSOR ID trojan can execute 17 commands received from the attacker-controlled server such as uninstalling an app, launching an app and then performing any click/tap action controlled remotely by the attacker (see Figure 5).



```
private void Configure_command() {
    this.firebaseCmd = this.fire_db.getReference().child(this.device_id).child("cmd");
    this.firebaseCmd.child("android_version").setValue("Android Version : null");
    this.firebaseCmd.child("appList").setValue("0");
    this.firebaseCmd.child("back").setValue("0");
    this.firebaseCmd.child("screen").setValue("0");
    this.firebaseCmd.child("lock").setValue("0");
    this.firebaseCmd.child("home").setValue("0");
    this.firebaseCmd.child("getinfo").setValue("0");
    this.firebaseCmd.child("click").setValue(";");
    this.firebaseCmd.child("touch").setValue(";");
    this.firebaseCmd.child("recent").setValue("0");
    this.firebaseCmd.child("launch").setValue("0");
    this.firebaseCmd.child("close").setValue("0");
    this.firebaseCmd.child("text").setValue(";");
    this.firebaseCmd.child("uninstall").setValue("0");
    this.firebaseCmd.child("notification").setValue("0");
    this.firebaseCmd.child("content").setValue("0");
    this.firebaseCmd.child("unlock").setValue("0");
    this.firebaseCmd.child("state").setValue("0");
```

Figure 5. The list of commands DEFENSOR ID may get from its C&C server

In 2018, we saw similar behavior, but all the click actions were hardcoded and suited only for the app of the attacker's choice. In this case, the attacker can get the list of all installed apps and then remotely launch the victim's app of their choice to either steal credentials or perform malicious actions (e.g. send funds via a wire transfer).

We believe that this is the reason the DEFENSOR ID trojan requests the user to allow "Modify system settings". Subsequently, the malware will change the screen off time-out to 10 minutes. This means that, unless victims lock their devices via the hardware button, the timer provides plenty of time for the malware to remotely perform malicious, in-app operations.

If the device gets locked, the malware can't unlock it.

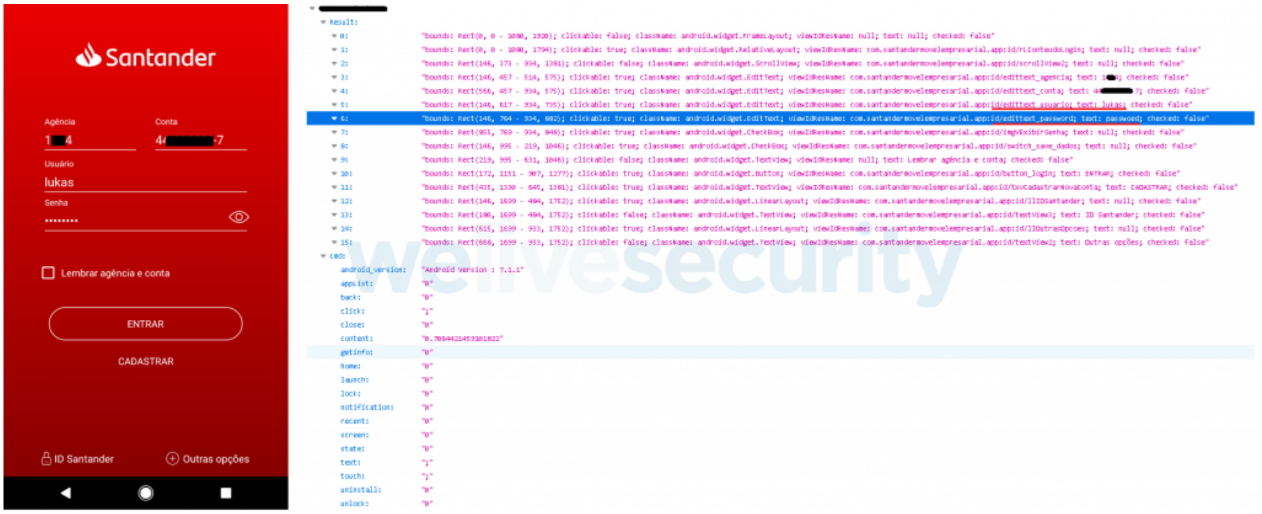
Malware data leak

When we analyzed the sample, we realized that the malware operators left the remote database with some of the victims' data freely accessible, without any authentication. The database contained the last activity performed on around 60 compromised devices. We found no other information stolen from the victims to be accessible.

Thanks to this data leak, we were able to confirm that the malware really worked as designed: the attacker had access to the victims' entered credentials, displayed or written emails and messages, etc.

Once we reached the non-secured database, we were able to directly observe the app's malicious behavior. To illustrate the level of threat the DEFENSOR ID app posed, we performed three tests.

First, we launched a banking app and entered the credentials there. The credentials were immediately available in the leaky database – see Figure 6.



Agência
1[redacted]4

Conta
44[redacted]7

Usuário
lukas

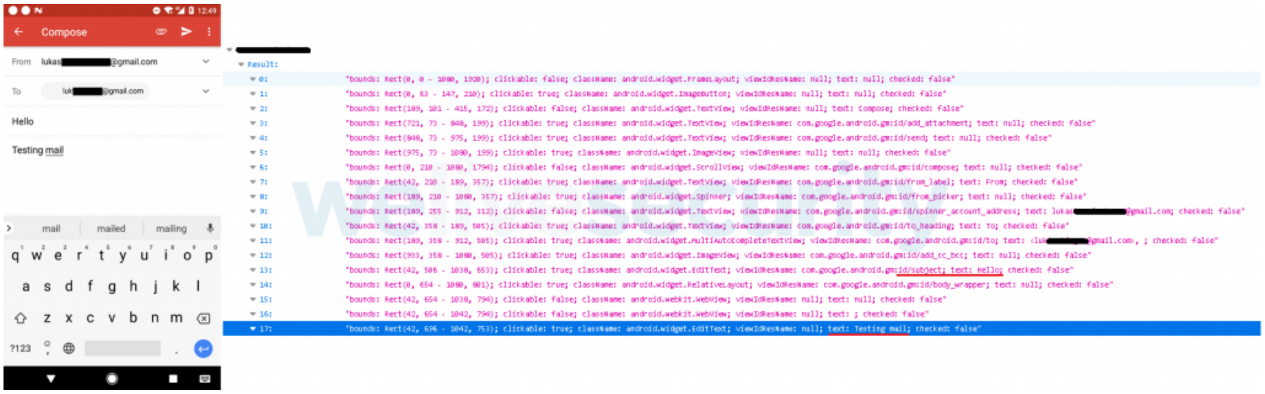
Senha
••••••••

□ Lembrar agência e conta

```
▼ [redacted]
  ▼ Result:
    ▼ 0:    "bounds: Rect(0, 0 - 1080, 1920); clickable: false; className: android.widget.FrameLayout; viewIdResName: null; text: null; checked: false"
    ▼ 1:    "bounds: Rect(0, 0 - 1080, 1794); clickable: true; className: android.widget.RelativeLayout; viewIdResName: com.santandermovelempresarial.app:id/rLConteudoLogin; text: null; checked: false"
    ▼ 2:    "bounds: Rect(146, 373 - 934, 1381); clickable: false; className: android.widget.ScrollView; viewIdResName: com.santandermovelempresarial.app:id/scrollView2; text: null; checked: false"
    ▼ 3:    "bounds: Rect(146, 457 - 514, 575); clickable: true; className: android.widget.EditText; viewIdResName: com.santandermovelempresarial.app:id/edittext_agencia; text: 1[redacted]; checked: false"
    ▼ 4:    "bounds: Rect(566, 457 - 934, 575); clickable: true; className: android.widget.EditText; viewIdResName: com.santandermovelempresarial.app:id/edittext_conta; text: 44[redacted]7; checked: false"
    ▼ 5:    "bounds: Rect(146, 617 - 934, 735); clickable: true; className: android.widget.EditText; viewIdResName: com.santandermovelempresarial.app:id/edittext_usuario; text: lukas; checked: false"
    ▼ 6:    "bounds: Rect(146, 764 - 934, 882); clickable: true; className: android.widget.EditText; viewIdResName: com.santandermovelempresarial.app:id/edittext_password; text: password; checked: false"
    ▼ 7:    "bounds: Rect(855, 769 - 934, 848); clickable: true; className: android.widget.CheckBox; viewIdResName: com.santandermovelempresarial.app:id/imgVExibirSenha; text: null; checked: false"
    ▼ 8:    "bounds: Rect(146, 995 - 219, 1046); clickable: true; className: android.widget.CheckBox; viewIdResName: com.santandermovelempresarial.app:id/switch_save_dados; text: null; checked: false"
    ▼ 9:    "bounds: Rect(219, 995 - 631, 1046); clickable: false; className: android.widget.TextView; viewIdResName: null; text: Lembrar agência e conta; checked: false"
    ▼ 10:   "bounds: Rect(172, 1151 - 907, 1277); clickable: true; className: android.widget.Button; viewIdResName: com.santandermovelempresarial.app:id/button_login; text: ENTRAR; checked: false"
    ▼ 11:   "bounds: Rect(435, 1330 - 645, 1381); clickable: true; className: android.widget.TextView; viewIdResName: com.santandermovelempresarial.app:id/txvCadastrarNovaConta; text: CADASTRAR; checked: false"
    ▼ 12:   "bounds: Rect(146, 1699 - 494, 1752); clickable: true; className: android.widget.LinearLayout; viewIdResName: com.santandermovelempresarial.app:id/llIDSantander; text: null; checked: false"
    ▼ 13:   "bounds: Rect(180, 1699 - 494, 1752); clickable: false; className: android.widget.TextView; viewIdResName: com.santandermovelempresarial.app:id/textView3; text: ID Santander; checked: false"
    ▼ 14:   "bounds: Rect(615, 1699 - 933, 1752); clickable: true; className: android.widget.LinearLayout; viewIdResName: com.santandermovelempresarial.app:id/llOutrasOpcoes; text: null; checked: false"
    ▼ 15:   "bounds: Rect(668, 1699 - 933, 1752); clickable: false; className: android.widget.TextView; viewIdResName: com.santandermovelempresarial.app:id/textView2; text: Outras opções; checked: false"
  ▼ cmd:
      android_version:   "Android Version : 7.1.1"
      appList:           "0"
      back:              "0"
      click:             ";"
      close:             "0"
      content:           "0.7064421459101822"
      getinfo:           "0"
      home:              "0"
      launch:            "0"
      lock:              "0"
      notification:      "0"
      recent:            "0"
      screen:            "0"
      state:             "0"
      text:              ";"
      touch:             ";"
      uninstall:         "0"
      unlock:            "0"
```

Figure 6. The banking app test: the credentials as entered (left) and as available in the database (right)

Second, we wrote a test message in an email client. We saw the message uploaded to the attackers' server within a second – see Figure 7.



From lukas[redacted]@gmail.com

To luk[redacted]@gmail.com

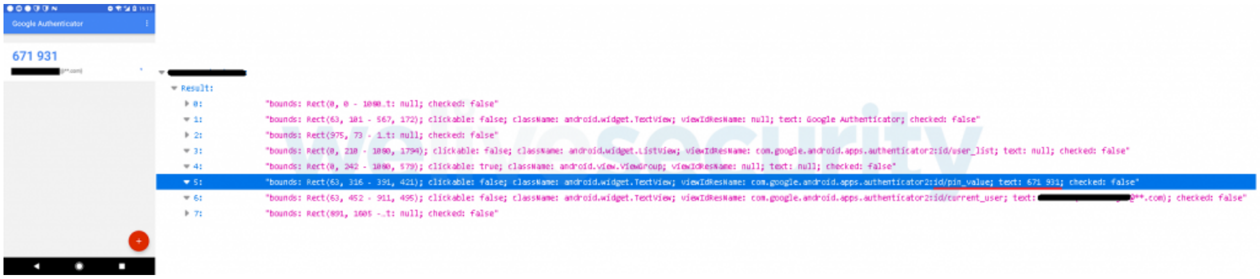
Hello

Testing mail

```
▼ [redacted]
  ▼ Result:
    ▼ 0:    "bounds: Rect(0, 0 - 1080, 1920); clickable: false; className: android.widget.FrameLayout; viewIdResName: null; text: null; checked: false"
    ▼ 1:    "bounds: Rect(0, 63 - 147, 210); clickable: true; className: android.widget.ImageButton; viewIdResName: null; text: null; checked: false"
    ▼ 2:    "bounds: Rect(189, 101 - 415, 172); clickable: false; className: android.widget.TextView; viewIdResName: null; text: Compose; checked: false"
    ▼ 3:    "bounds: Rect(721, 73 - 848, 199); clickable: true; className: android.widget.TextView; viewIdResName: com.google.android.gm:id/add_attachment; text: null; checked: false"
    ▼ 4:    "bounds: Rect(848, 73 - 975, 199); clickable: true; className: android.widget.TextView; viewIdResName: com.google.android.gm:id/send; text: null; checked: false"
    ▼ 5:    "bounds: Rect(975, 73 - 1080, 199); clickable: true; className: android.widget.ImageView; viewIdResName: null; text: null; checked: false"
    ▼ 6:    "bounds: Rect(0, 210 - 1080, 1794); clickable: false; className: android.widget.ScrollView; viewIdResName: com.google.android.gm:id/compose; text: null; checked: false"
    ▼ 7:    "bounds: Rect(42, 210 - 189, 357); clickable: true; className: android.widget.TextView; viewIdResName: com.google.android.gm:id/from_label; text: From; checked: false"
    ▼ 8:    "bounds: Rect(189, 210 - 1080, 357); clickable: true; className: android.widget.Spinner; viewIdResName: com.google.android.gm:id/from_picker; text: null; checked: false"
    ▼ 9:    "bounds: Rect(189, 255 - 912, 312); clickable: false; className: android.widget.TextView; viewIdResName: com.google.android.gm:id/spinner_account_address; text: lukas[redacted]@gmail.com; checked: false"
    ▼ 10:   "bounds: Rect(42, 358 - 189, 505); clickable: true; className: android.widget.TextView; viewIdResName: com.google.android.gm:id/to_heading; text: To; checked: false"
    ▼ 11:   "bounds: Rect(189, 358 - 912, 505); clickable: true; className: android.widget.MultiAutoCompleteTextView; viewIdResName: com.google.android.gm:id/to; text: <luk[redacted]@gmail.com>, ; checked: false"
    ▼ 12:   "bounds: Rect(933, 358 - 1080, 505); clickable: true; className: android.widget.ImageView; viewIdResName: com.google.android.gm:id/add_cc_bcc; text: null; checked: false"
    ▼ 13:   "bounds: Rect(42, 506 - 1038, 653); clickable: true; className: android.widget.EditText; viewIdResName: com.google.android.gm:id/subject; text: Hello; checked: false"
    ▼ 14:   "bounds: Rect(0, 654 - 1080, 801); clickable: true; className: android.widget.RelativeLayout; viewIdResName: com.google.android.gm:id/body_wrapper; text: null; checked: false"
    ▼ 15:   "bounds: Rect(42, 654 - 1038, 794); clickable: false; className: android.webkit.WebView; viewIdResName: null; text: null; checked: false"
    ▼ 16:   "bounds: Rect(42, 654 - 1042, 794); clickable: false; className: android.webkit.WebView; viewIdResName: null; text: ; checked: false"
    ▼ 17:   "bounds: Rect(42, 696 - 1042, 753); clickable: true; className: android.widget.EditText; viewIdResName: null; text: Testing mail; checked: false"
```

Figure 7. The email message test: the message as written (left) and as available in the database (right)

Third, we documented the trojan retrieving the Google Authenticator 2FA code.



671 931

[redacted]

```
▼ [redacted]
  ▼ Result:
    ▶ 0:    "bounds: Rect(0, 0 - 1080…t: null; checked: false"
    ▼ 1:    "bounds: Rect(63, 101 - 567, 172); clickable: false; className: android.widget.TextView; viewIdResName: null; text: Google Authenticator; checked: false"
    ▶ 2:    "bounds: Rect(975, 73 - 1…t: null; checked: false"
    ▼ 3:    "bounds: Rect(0, 210 - 1080, 1794); clickable: false; className: android.widget.ListView; viewIdResName: com.google.android.apps.authenticator2:id/user_list; text: null; checked: false"
    ▼ 4:    "bounds: Rect(0, 242 - 1080, 579); clickable: true; className: android.view.ViewGroup; viewIdResName: null; text: null; checked: false"
    ▼ 5:    "bounds: Rect(63, 316 - 391, 421); clickable: false; className: android.widget.TextView; viewIdResName: com.google.android.apps.authenticator2:id/pin_value; text: 671 931; checked: false"
    ▼ 6:    "bounds: Rect(63, 452 - 911, 495); clickable: false; className: android.widget.TextView; viewIdResName: com.google.android.apps.authenticator2:id/current_user; text: [redacted]@**.com); checked: false"
    ▶ 7:    "bounds: Rect(891, 1605 -…t: null; checked: false"
```

Figure 8. The software generated 2FA code as it appeared on the device's display (left) and as available in the database (right)

Along with the malicious DEFENSOR ID app, another malicious app named Defensor Digital was discovered. Both apps shared the same C&C server, but we couldn't investigate the latter as it had already been removed from the Google Play store.

Indicators of Compromise (IoCs)

Package Name	Hash	ESET detection name

Package Name	Hash	ESET detection name
com.secure.protect.world	F17AEBC741957AA21CFE7C7D7BAEC0900E863F61	Android/Spy.BanBra.A
com.brazil.android.free	EA069A5C96DC1DB0715923EB68192FD325F3D3CE	Android/Spy.BanBra.A

MITRE ATT&CK techniques

Tactic	ID	Name	Description
Initial Access	T1475	Deliver Malicious App via Authorized App Store	Impersonates security app on Google Play.
T1444	Masquerade as Legitimate Application	Impersonates legitimate GAS Tecnologia application.	
Discovery	T1418	Application Discovery	Sends list of installed apps on device.
Impact	T1516	Input Injection	Can enter text and perform clicks on behalf of user.
Collection	T1417	Input Capture	Records user input data.
Command and Control	T1437	Standard Application Layer Protocol	Uses Firebase Cloud Messaging for C&C.

22 May 2020 - 03:00PM

Sign up to receive an email update whenever a new article is published in our Ukraine Crisis – Digital Security Resource Center

Newsletter

Discussion