BRATA is evolving into an Advanced Persistent Threat

Francesco Iubatti, Alessandro Strino

Introduction

Here we go with another episode about our (not so) old friend, BRATA. In almost one year, threat actors (TAs) have further improved the capabilities of this malware. In our previous blog post [1] we defined three main BRATA variants, which appeared during two different waves detected by our telemetries at the very end of 2021. However, during the last months we have observed a change in the attack pattern commonly used.

In fact, the modus operandi now fits into an Advanced Persistent Threat (APT) activity pattern. This term is used to describe an attack campaign in which criminals establish a long-term presence on a targeted network to steal sensitive information.

Threat Actors behind BRATA, now target a specific financial institution at a time, and change their focus only once the targeted victim starts to implement consistent countermeasures against them. Then, they move away from the spotlight, to come out with a different target and strategies of infections. At first glance, it seems to be a good strategy with a relevant pay off. However, it's important to point out also the struggles and the plan needed to apply this pattern.

As we highlighted through our metrics, when a new release comes out there are also new features that make it more dangerous. During the last months, a new **BRATA.A** variant has been spotted in EU territory posing as specific bank applications, including some internal changes, such as:

- A new phishing technique that is in charge of mimicking a login page of the targeted bank;
- Brand new classes in charge to acquire GPS, overlay, SMS and device management permissions;
- Sideloading a piece of code (second stage) downloaded from its C2 to perform Event Logging.

In this article, we give an overview of these new features in order to figure out their purposes and forecast new evolutions.

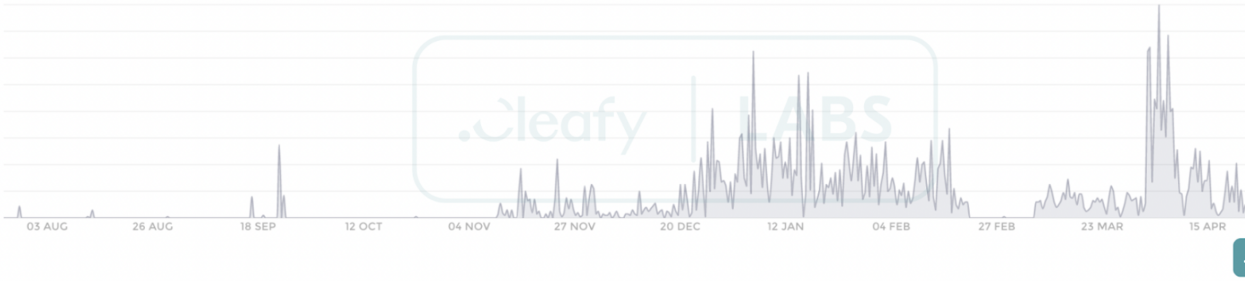

Figure 1 – BRATA activities during 2021/2022

[1] https://www.cleafy.com/labs

[2] https://www.cleafy.com/cleafy-labs/how-brata-is-monitoring-your-bank-account

[3] https://www.cleafy.com/cleafy-labs/mobile-banking-fraud-brata-strikes-again

BRATA updates

As we already mentioned in the previous paragraph, TAs are modifying their code in order to tailor their malware on specific banking institutions. This code refractory is actually doing small changes compared to old versions of BRATA, as there are a bunch of classes that have been added for very specific purposes.

Nevertheless, before proceeding with a deep dive into BRATA's features, in Figure 2 we have highlighted all new functions that we observed in the last month. Most of them are very specific and their purpose is crystal clear.

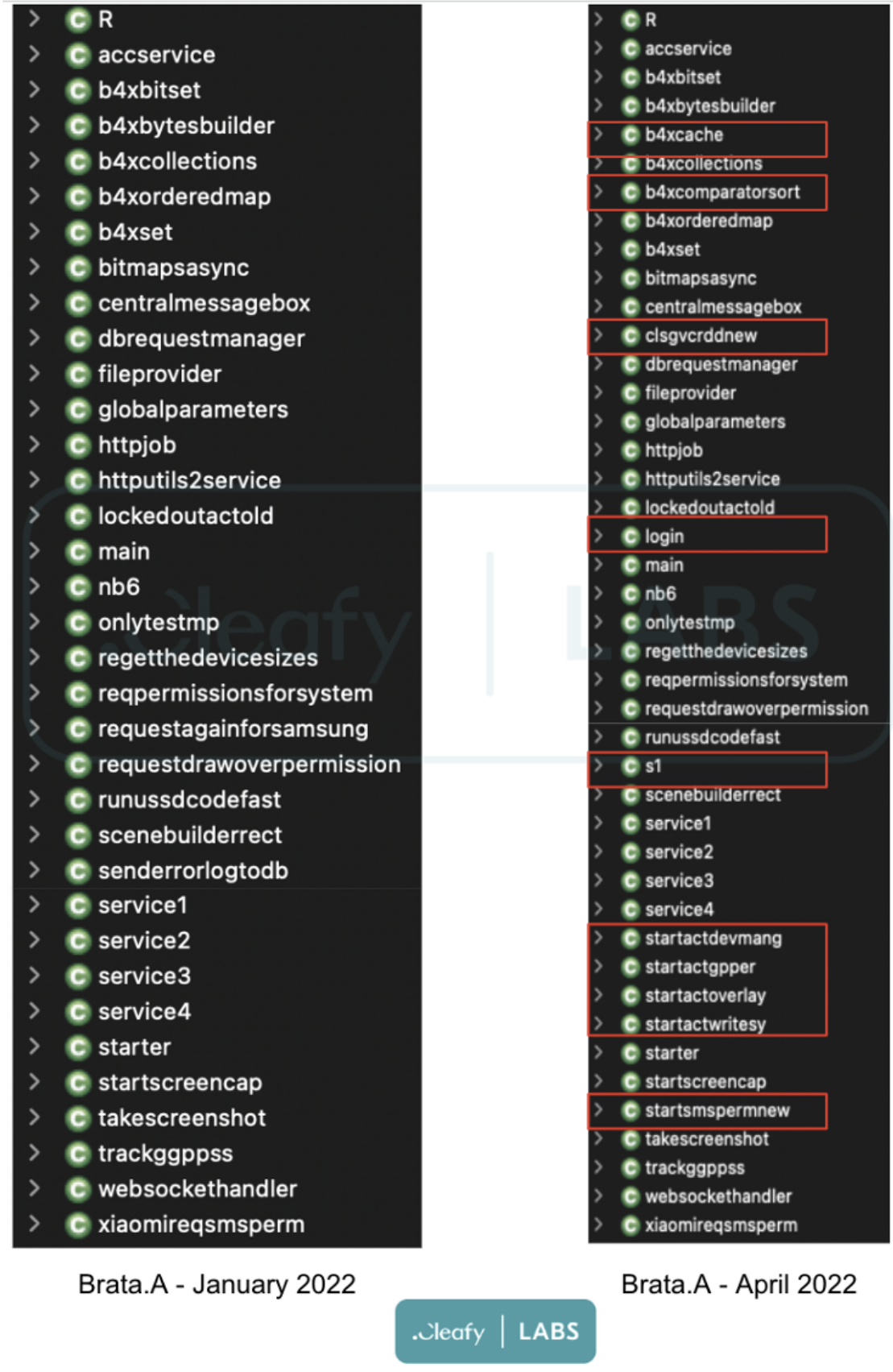

Figure 2 – Differences between BRATA variants

Speaking about examples, we could observe a login class that disguises a classic login page in order to harvest credentials from unaware users, as well as classes like *startactdevmang*, *startactgpper*, *startactoverlay* and *startsmspermnew* have been introduced to request additional permissions for later fraud phases (e.g., device administration, gps, overlay, SMS, etc.).

Credential Harvesting Attack

Investigation on this sample has led our researchers to discover that BRATA has been equipped with a phishing page that recreates a login page of a famous Italian bank. In this way, TAs are trying to steal sensitive information from their victims to perform some sort of social engineering in a later stage of the fraud. As shown in Figure 3, the victim is lured to type *Numero Cliente* and *PIN*. This information is the foundation of the authentication process commonly used by banks.

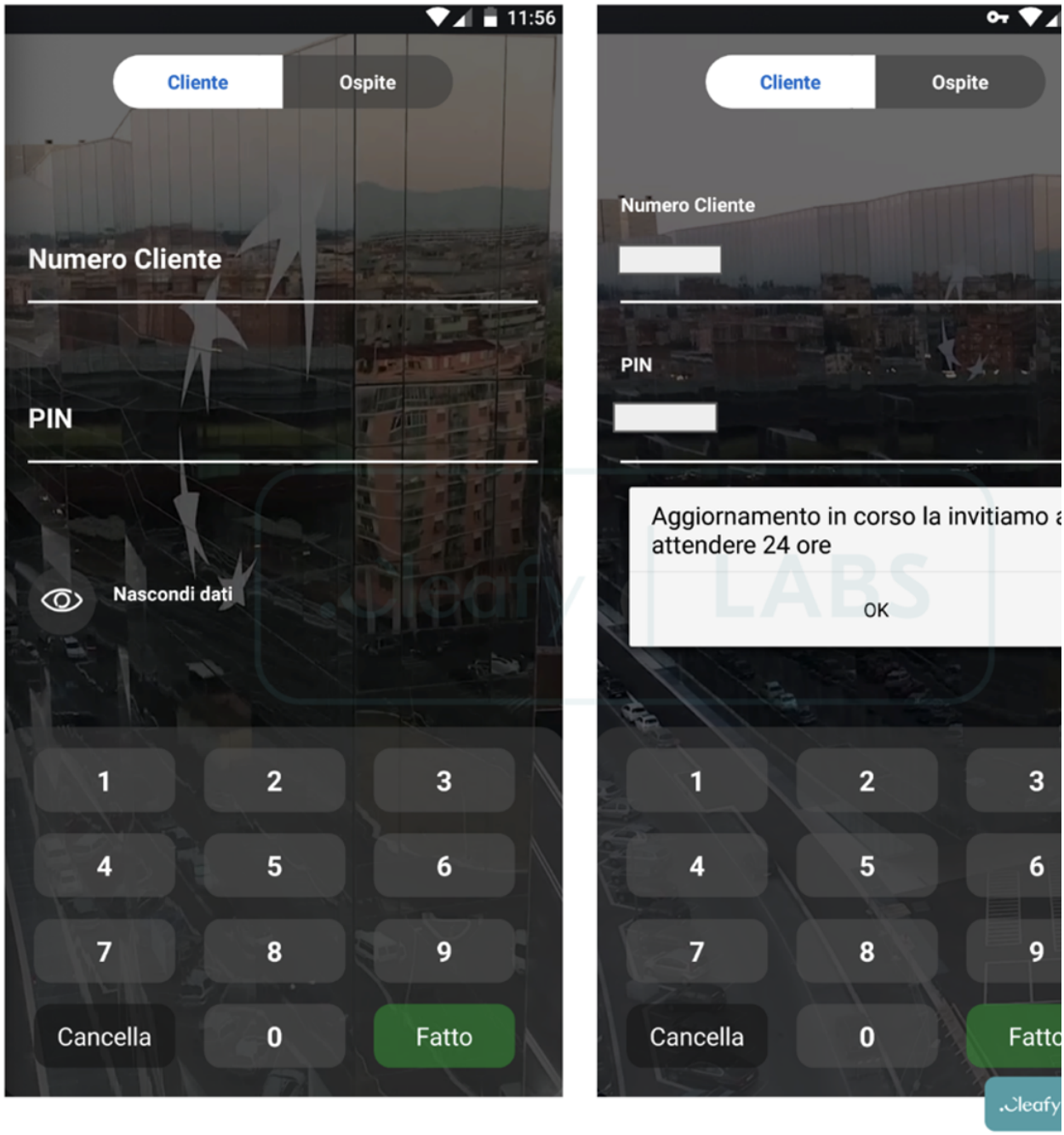

Figure 3 - BRATA phishing page

It's worth mentioning that, at the time of writing, this information seems to be under development. This hypothesis is supported by the fact that there is no data exchange between the victim device and the TA infrastructure. Moreover, log information and local databases did not show evidence that these information are stored somewhere on the device. Another technique is related to the usage of screen recording functionality that could avoid storing this information, however, adopting this strategy requires additional alerting mechanisms in order to properly see all codes typed.

Furthermore, the current version of BRATA has introduced two new permissions inside the AndroidManifest file, the RECEIVE_SMS and SEND_SMS. The combination of the phishing page with the possibility to receive and read the victim's sms could be used to perform a complete Account Takeover (ATO) attack.

```
public static String _activity_create(boolean arg8) throws Exception {
    login.mostCurrent._ime1.Initialize("");
    login.mostCurrent._ime1.HideKeyboard(login.mostCurrent.activityBA);
    login.mostCurrent._activity.setColor(-1);
    login.mostCurrent._activity.LoadLayout("LLoginBNL", login.mostCurrent.activityBA);
    login.mostCurrent._simpleexoplayerview1.setResizeMode("FIXED_HEIGHT");
    login.mostCurrent._player1.Initialize(login.processBA, "player");
    login.mostCurrent._player1.Prepare(login.mostCurrent._player1.CreateLoopSource(login.mostCurrent._player1.CreateFileSource("AssetsDir", "videoapp.mp4"), -1));
    login.mostCurrent._simpleexoplayerview1.setPlayer(login.mostCurrent._player1);
    login.mostCurrent._player1.Play();
    PanelWrapper v8 = new PanelWrapper();
    v8.Initialize(login.mostCurrent.activityBA, "");
    login.mostCurrent._activity.AddView(((View)v8.getObject()), 0, 0, Common.PerXToCurrent(100.0f, login.mostCurrent.activityBA), Common.PerYToCurrent(100.0f, login.mostCurren
    v8.setColor(Colors.ARGB(150, 0, 0, 0));
    v8.BringToFront();
    login._create_ui_txt1();
    login._create_ui_txt12();
    login._create_ui_keypad();
    login._create_ui_topoptions();
    login._create_ui_hider();
    return "";
}
```

Figure 4 – Function used to create the phishing page

External Payload - Event logging

As we already mentioned above, after being correctly installed on the victim's phone, this BRATA version is going to download a .zip file from its C2. This file contains a jar file named *unrar.jar*.

```
package com.gvcrf.gvsrcfiles;

public final class BuildConfig {
    public static final String APPLICATION_ID = "com.remotefile.gvsrcf
    public static final String BUILD_TYPE = "release";
    public static final boolean DEBUG = false;
    public static final int VERSION_CODE = 1;
    public static final String VERSION_NAME = "1.0";

}
```

Figure 5 – BRATA external payload

From the information retrieved, this *plugin* seems to be in charge of monitoring events that are generated from applications. More specifically, each time there is a change in a text view, it stores within a local database a pair of *Event Text* and a *Date* when the event occurred.

```
public void save_events_db(String text, String strappnamekeylog, SQL sqlevents) {
    if(text.toLowerCase().contains("type_view_text_changed")) {
        try {
            String v6 = strappnamekeylog.equals("KeyLogEverything") ? "" : strappnamekeylog;
            String v5 = "";
            if((text.contains("[ ClassName:")) && (text.contains("]; Content")) && (text.toLowerCase().co
                if(text.indexOf("[ ClassName:") < text.indexOf("]; Content")) {
                    v5 = text.substring(text.indexOf("[ ClassName:"), text.indexOf("]; Content"));
                }

                if(v5.trim().length() > 0) {
                    v5 = v5.replace(v5.substring(0, v5.indexOf("Text: [")), "");
                    if(v5.contains("Text: [")) {
                        v5 = v5.replace("Text: [", "");
                    }
                }
            }

            if(v5.trim().length() > 0) {
                String v4 = DateTime.Date(DateTime.getNow()) + " " + DateTime.Time(DateTime.getNow());
                sqlevents.BeginTransaction();
                sqlevents.ExecNonQuery2("insert into tblkeys (eventtext,event) values (?,?)", Common.Arra
                sqlevents.TransactionSuccessful();
                sqlevents.EndTransaction();
```

Figure 6 – BRATA keylogger function

At the time of writing, this feature seems to be under development too. However, our hypothesis is that TAs are trying to extend the functionality of the malware to get data from other applications, abusing the Accessibility Service.

SMS Stealer

During our analysis of the last BRATA campaign, we found a suspicious app connected to the same BRATA C2 infrastructure.

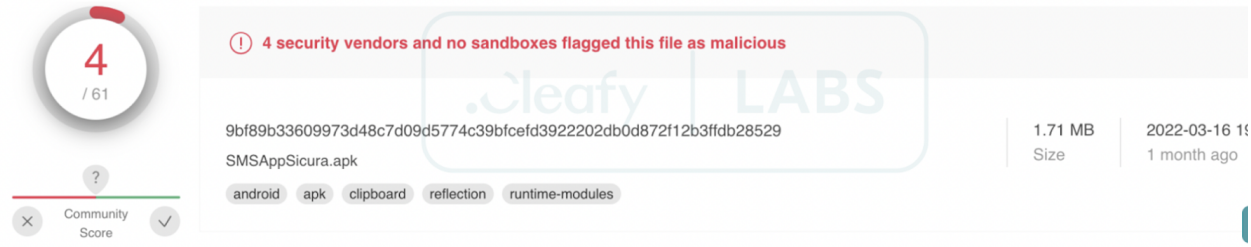

Figure 7 – Low detection of the SMS stealer app

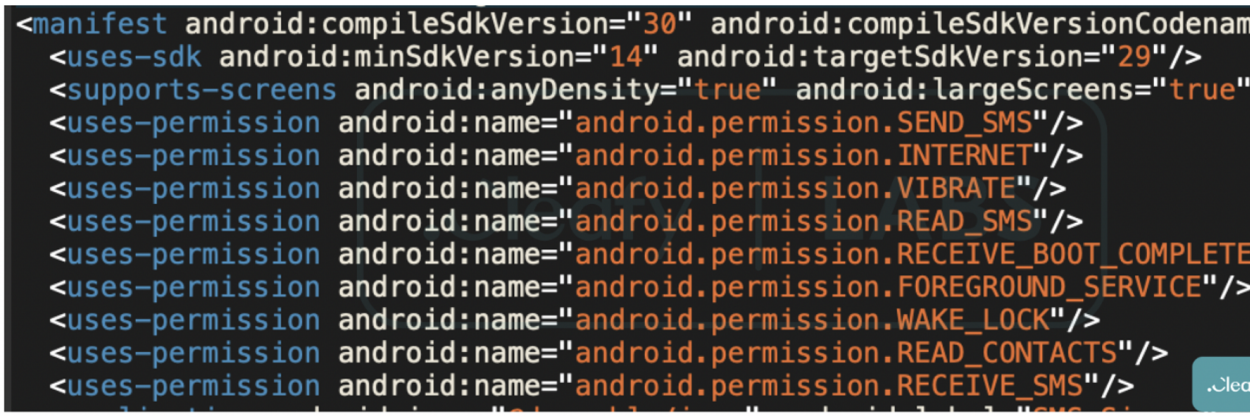

Figure 8 – Permissions declared inside the AndroidManifest file

Analyzing the suspicious app, we observed that the developer(s) of the app used the same framework used in BRATA malware and also the same names for different classes (Figure 9).

Thanks to a deeper analysis, it is possible to say that the TAs have used some portions of BRATA code's to create this new malicious app. Our hypothesis is that TAs are trying to develop new types of malware or they are just making some simple experiments to create new types of attacks, like contacts harvesting or SMS sniffers in order to stay undetected.

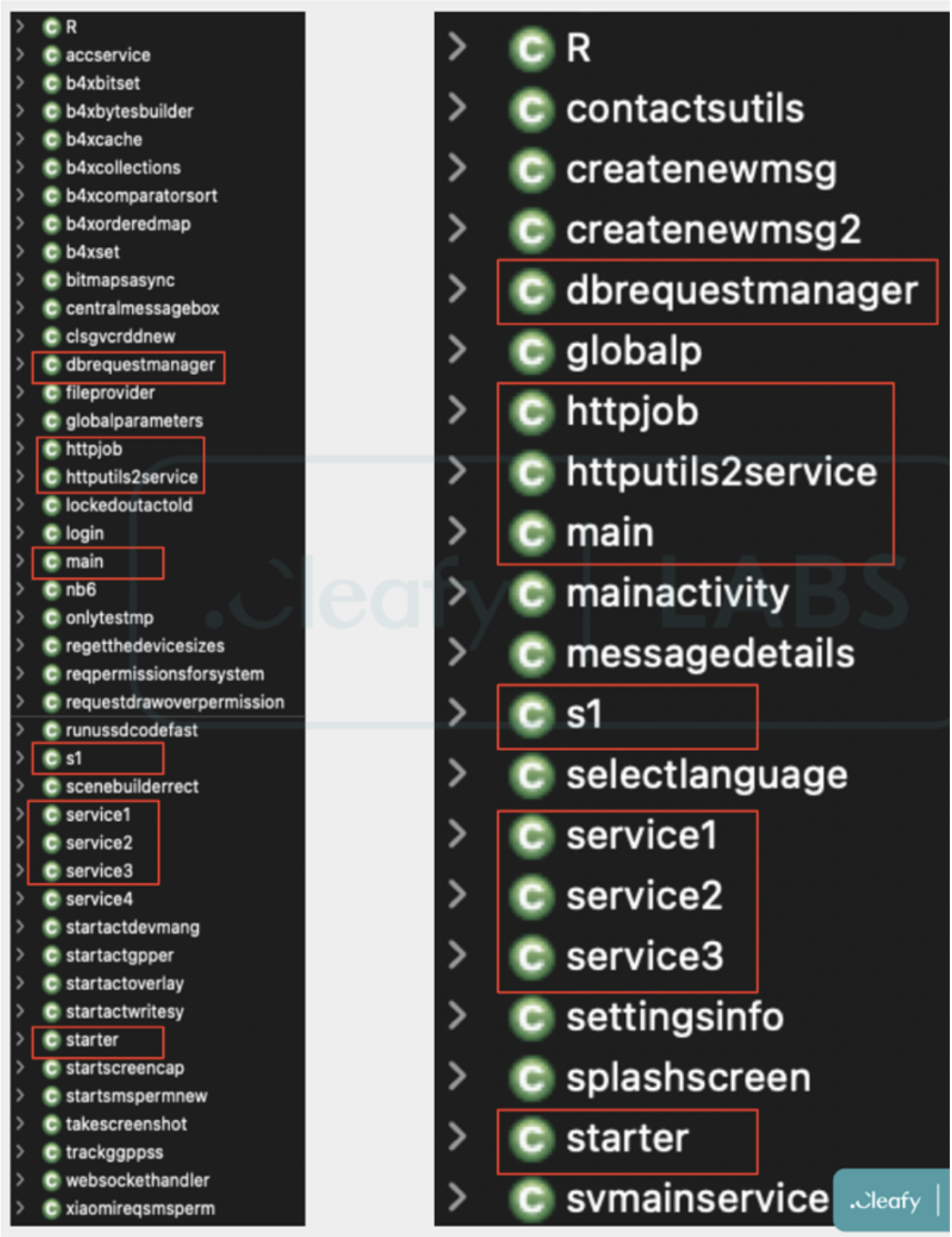

Figure 9 – Names similarities between BRATA and the SMS stealer

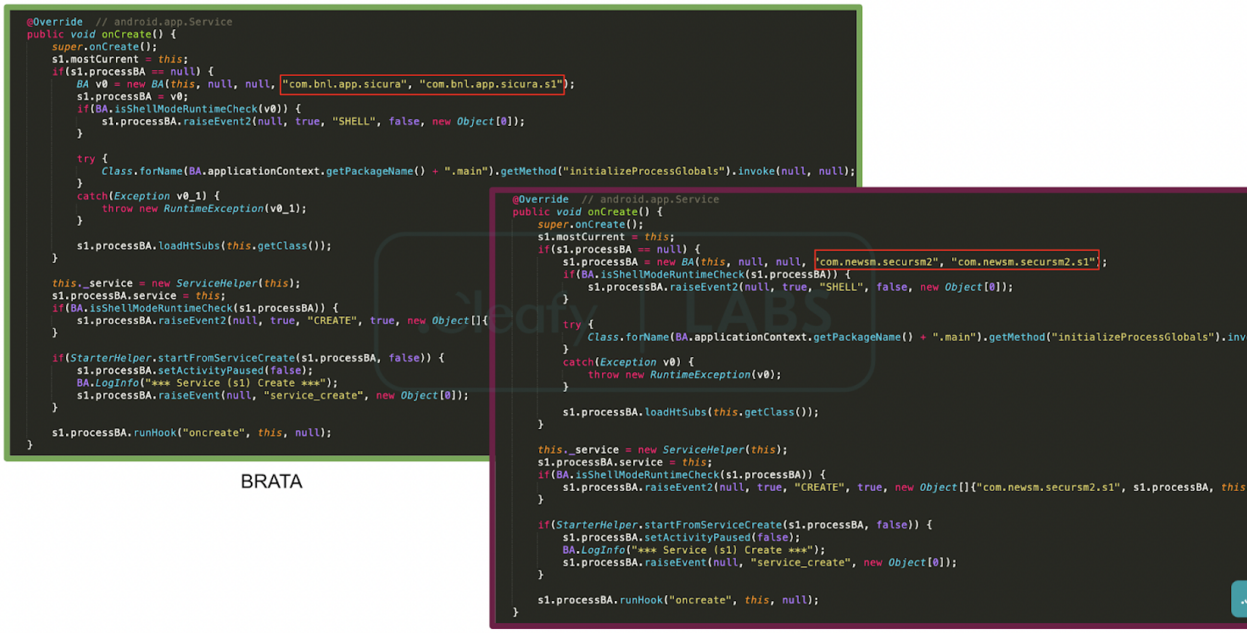

Figure 10 – Code similarities between BRATA and SMS stealer

The malicious app seems to target three different countries: Great Britain, Italy and Spain. During the installation phases, in fact, it requires you to choose the language of the app.

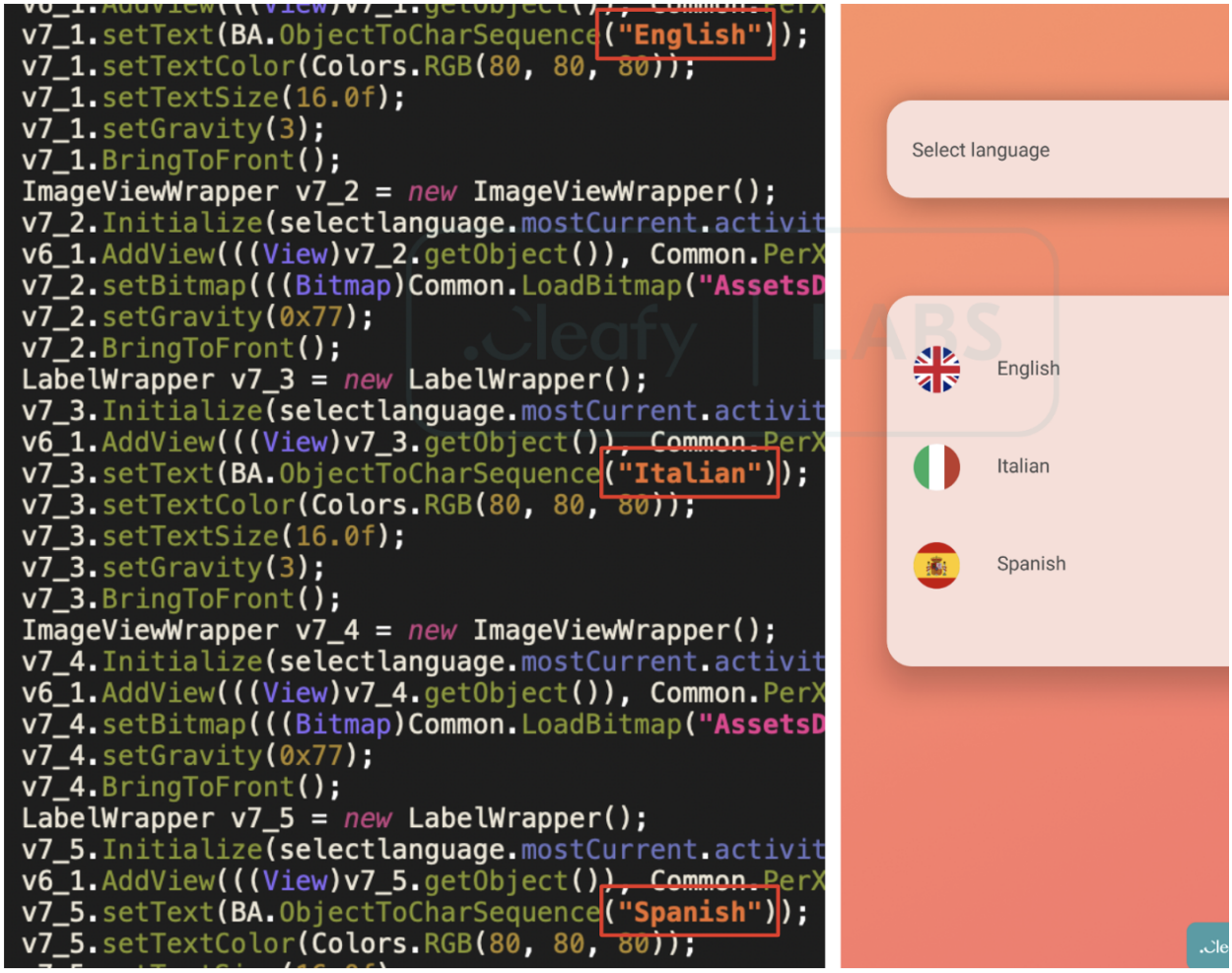

Figure 11 – SMS stealer targeted countries

Once installed, the pattern of the attack is similar to other SMS stealers. This consists in the malicious app asking the user to change the default messaging app with the malicious one to intercept all incoming messages, typically used by banks in PSD2 area for sending authorization codes (2FA/OTP).

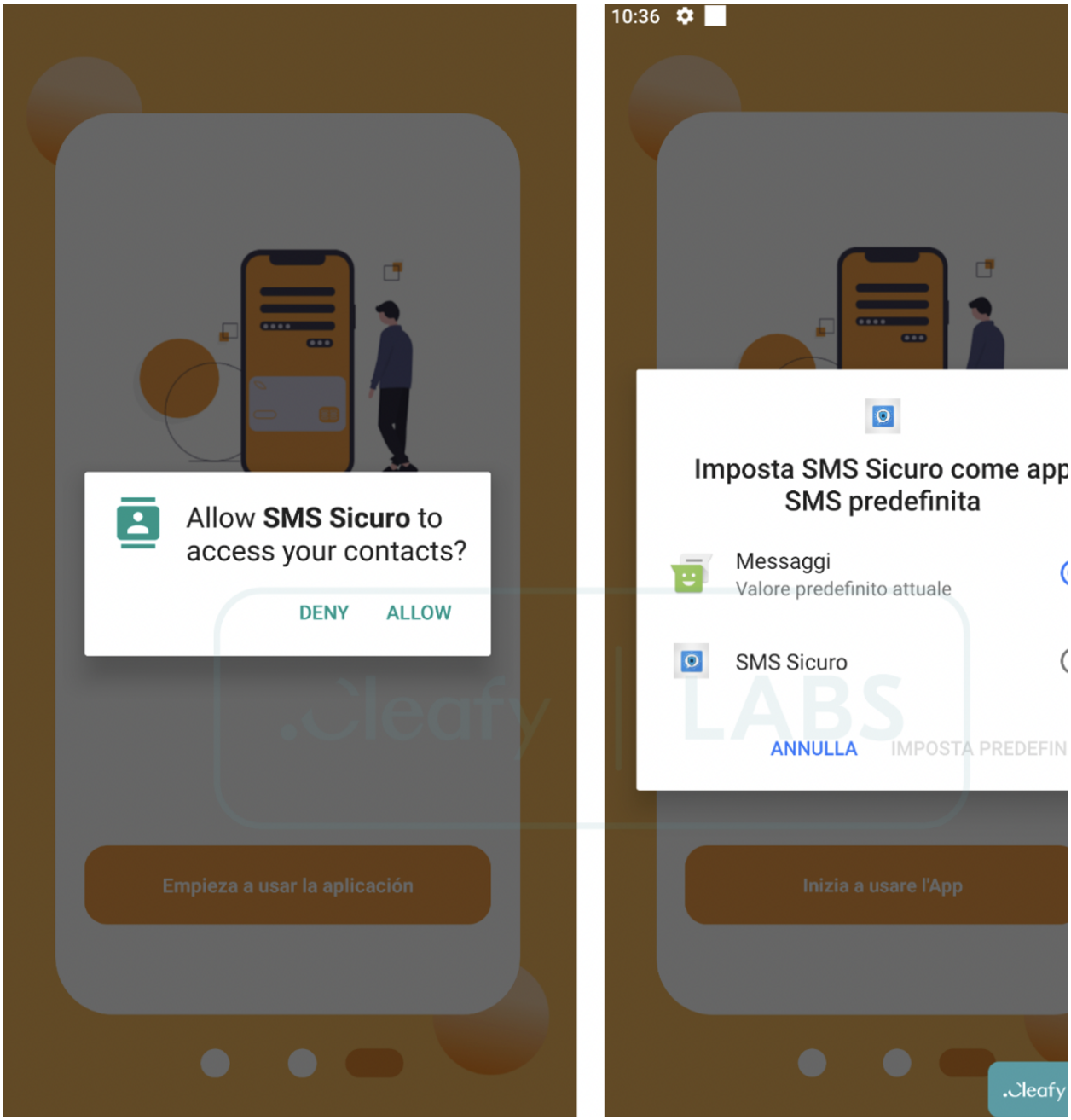

Figure 12 - factory reset command sent

The similarities between the SMS stealer and BRATA can be found in the network communication since both use the following two different ports and the endpoint "/rdc":

- Port 19999: used to notify to the C2 that the malicious app was installed on the victim device
- Port 18888: used to send SMS intercepted to the C2

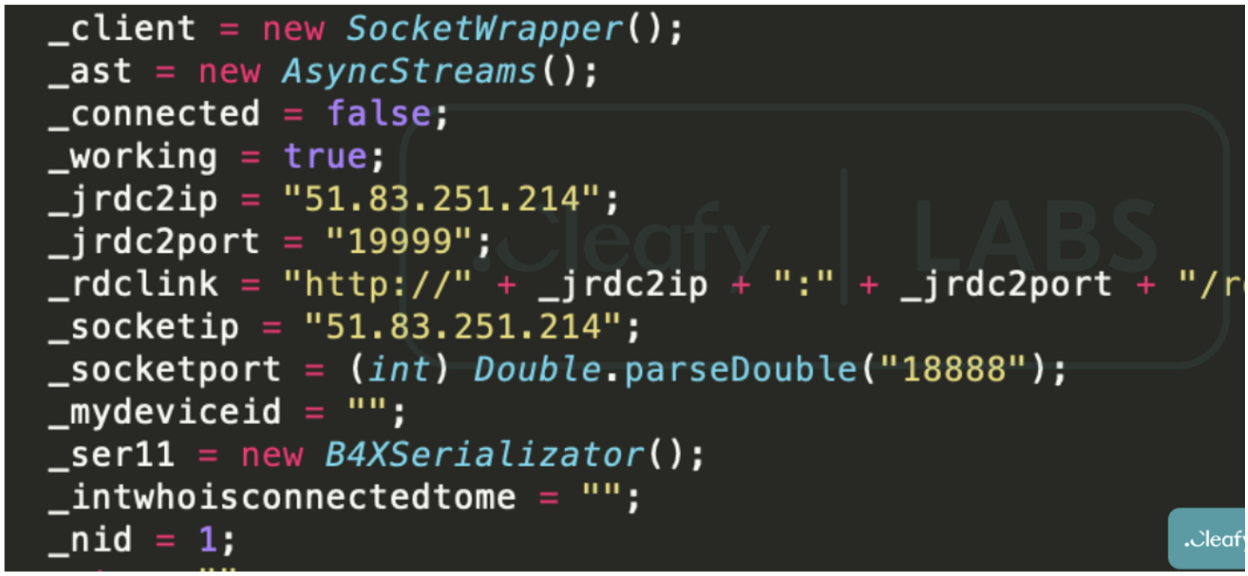

Figure 13 – Address and ports used to communicate with C2

Final Considerations

Starting from June 2021, when we first intercepted the BRATA campaigns in Italy, we observed an uninterrupted evolution of both the malware and the attack methodologies used by the TAs. The first campaigns of malware were distributed through fake antivirus or other common apps, while during the campaigns the malware is taking the turn of an APT attack against the customer of a specific Italian bank.

The latter trend, the so-called “Advanced Persistent Threat”, seems to be the attack pattern that TAs are going to use in the coming year.. They usually focus on delivering malicious applications targeted to a specific bank for a couple of months, and then moving to another target.

Appendix 1: IOCs

IoC	Description
1ae5fcbbd3d0e13192600ef05ba5640d	BRATA
69d3ce972e66635b238dc17e632474ec	SMS stealer
51[.]83[.]251[.]214	C2 server used by BRATA and the SMS stealer
51[.]83[.]225[.]224	Other BRATA C2 server