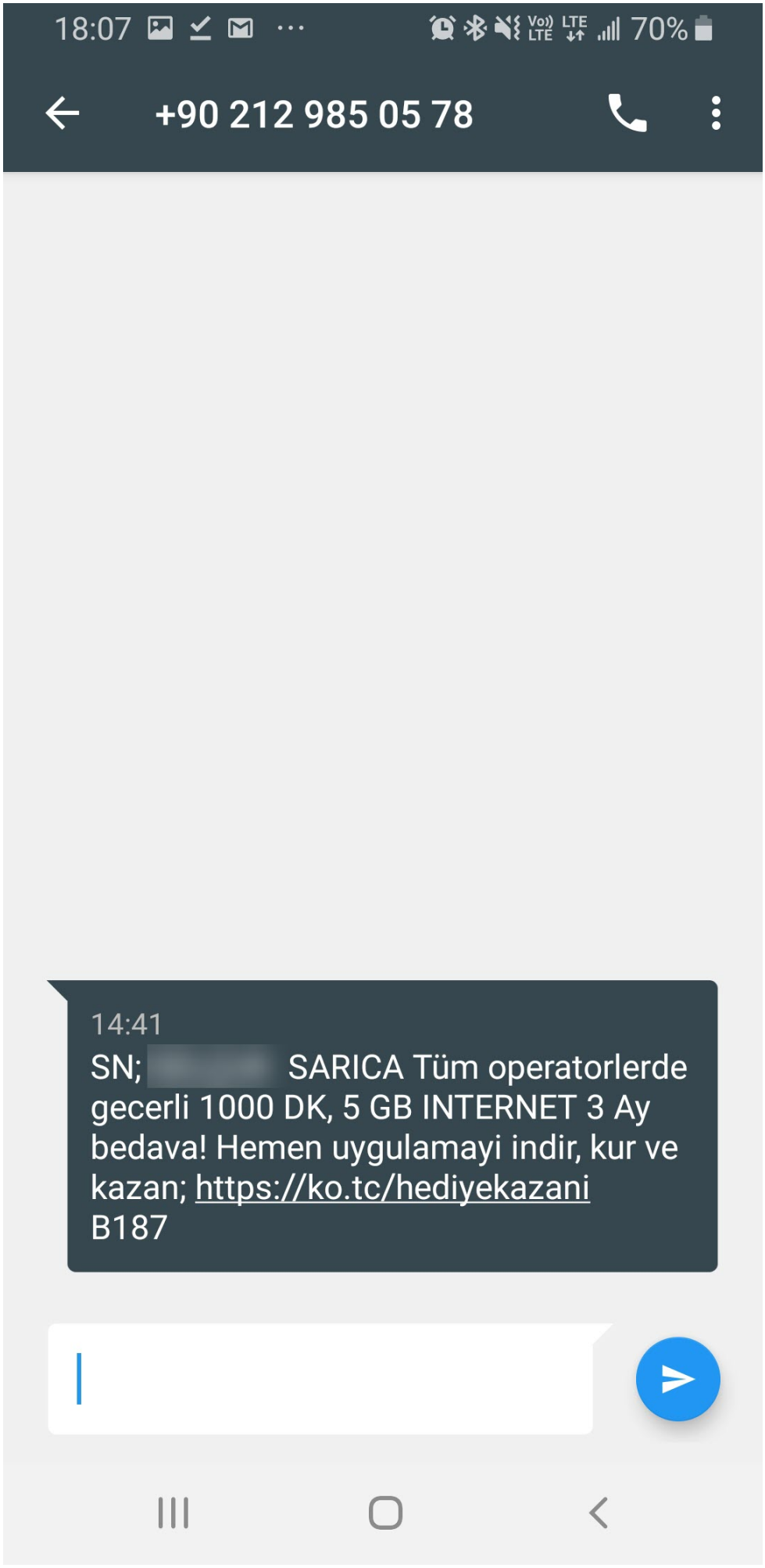
Cerberus Analysis

written by Mert SARICA | 1 December 2020

In February 2020, I received a SMS on my cell phone that made me quite suspicious. When I visited the https://ko[.]tc/hediyekazani web address mentioned in the message, I found that I was redirected to the http://www-bedavainternethediyeuygulama[.]com web address. A short time after receiving the SMS, when I visited the website again, I saw that the images on the site had changed. Saying “suspicion is the whip of a cyber security researcher,” I decided to take a closer look at this situation.


18:07
70%
+90 212 985 05 78
14:41
SN; SARICA Tüm operatorlerde gecerli 1000 DK, 5 GB INTERNET 3 Ay bedava! Hemen uygulamayi indir, kur ve kazan; https://ko.tc/hediyekazani
B187


18:04
71%
thediyeliuygulama.com
16
5G Uygulaması 5GB İnternet Veriyor
1)5G Uygulaması İnternet Kazandırıyor 5GB İnternet Kazanmak İçin Hemen İndir
İndir
Bu türden dosyalar cihazınıza zarar verebilir. Yine de 5GBeta.apk adlı dosyayı saklamak istiyor musunuz?
İptal
Tamam

18:20 68%

thediyeliuygulama.com

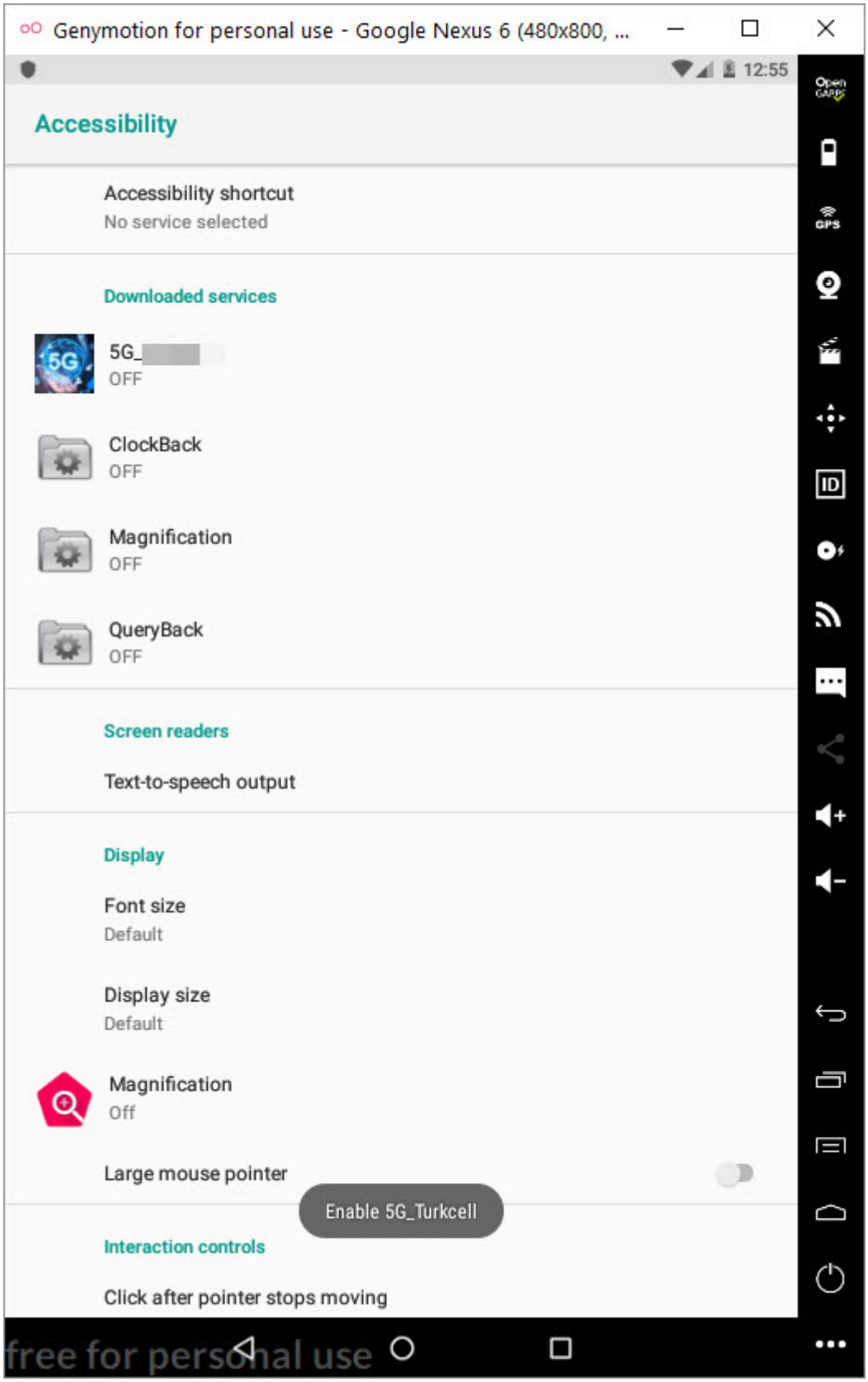
5G Beta Test

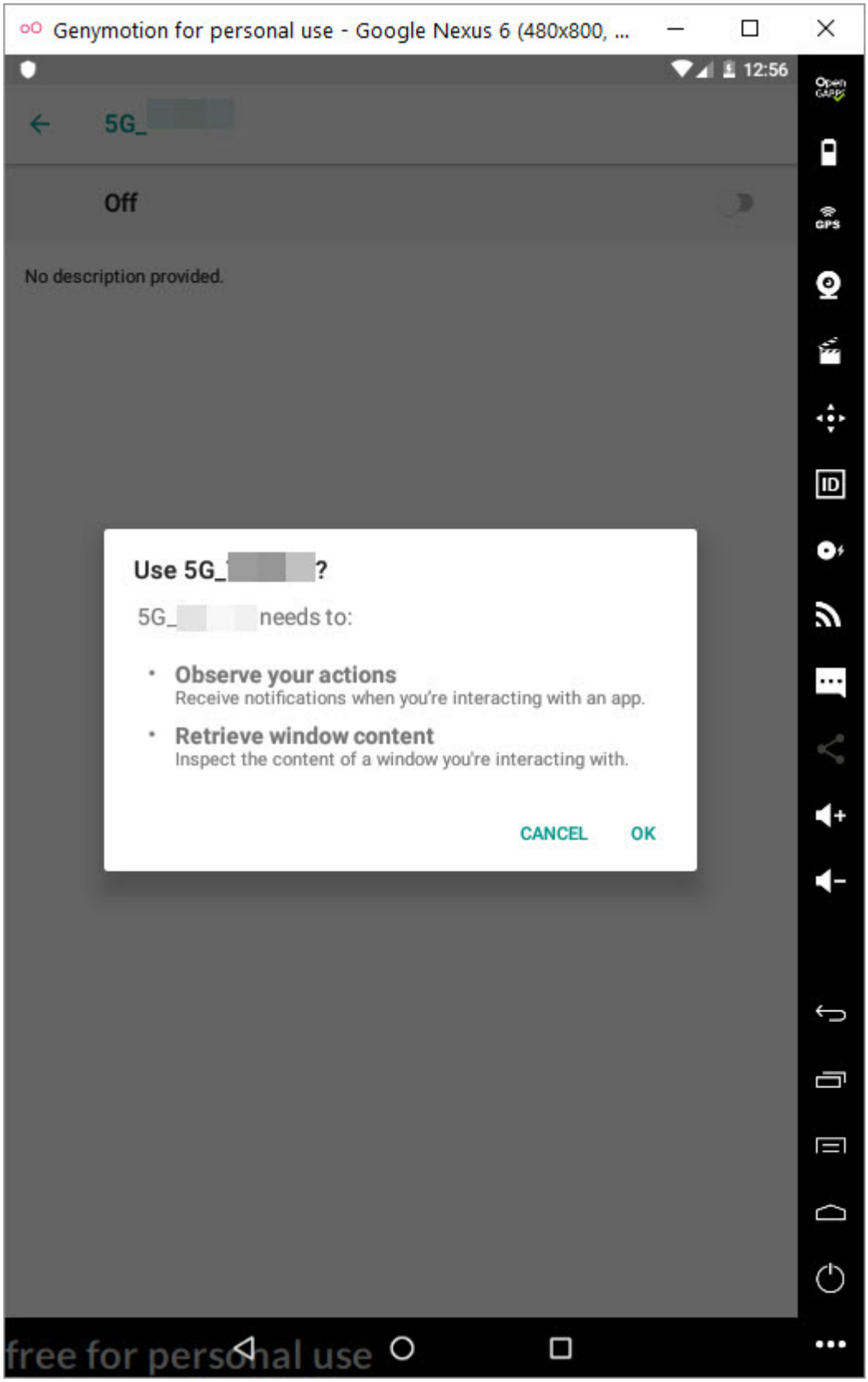
5G Uygulamasini İndirip Yükleyen Tüm Operatör Müşterilerine 5GB İnternet Hediye Sizde Hemen İndirin Yükleyin 5GB İnterneti Hemen Kazanın

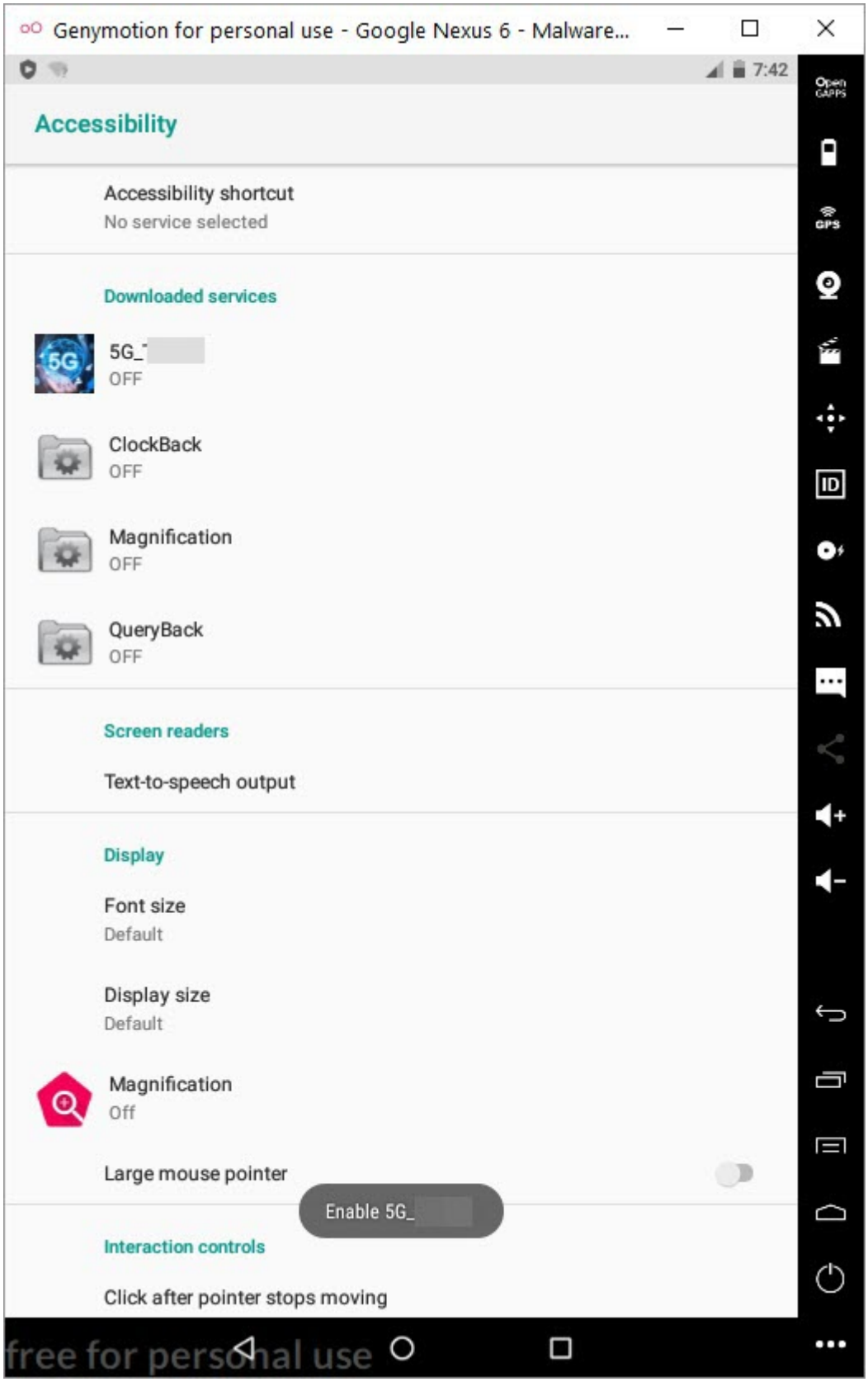
Uygulamayı Yükle

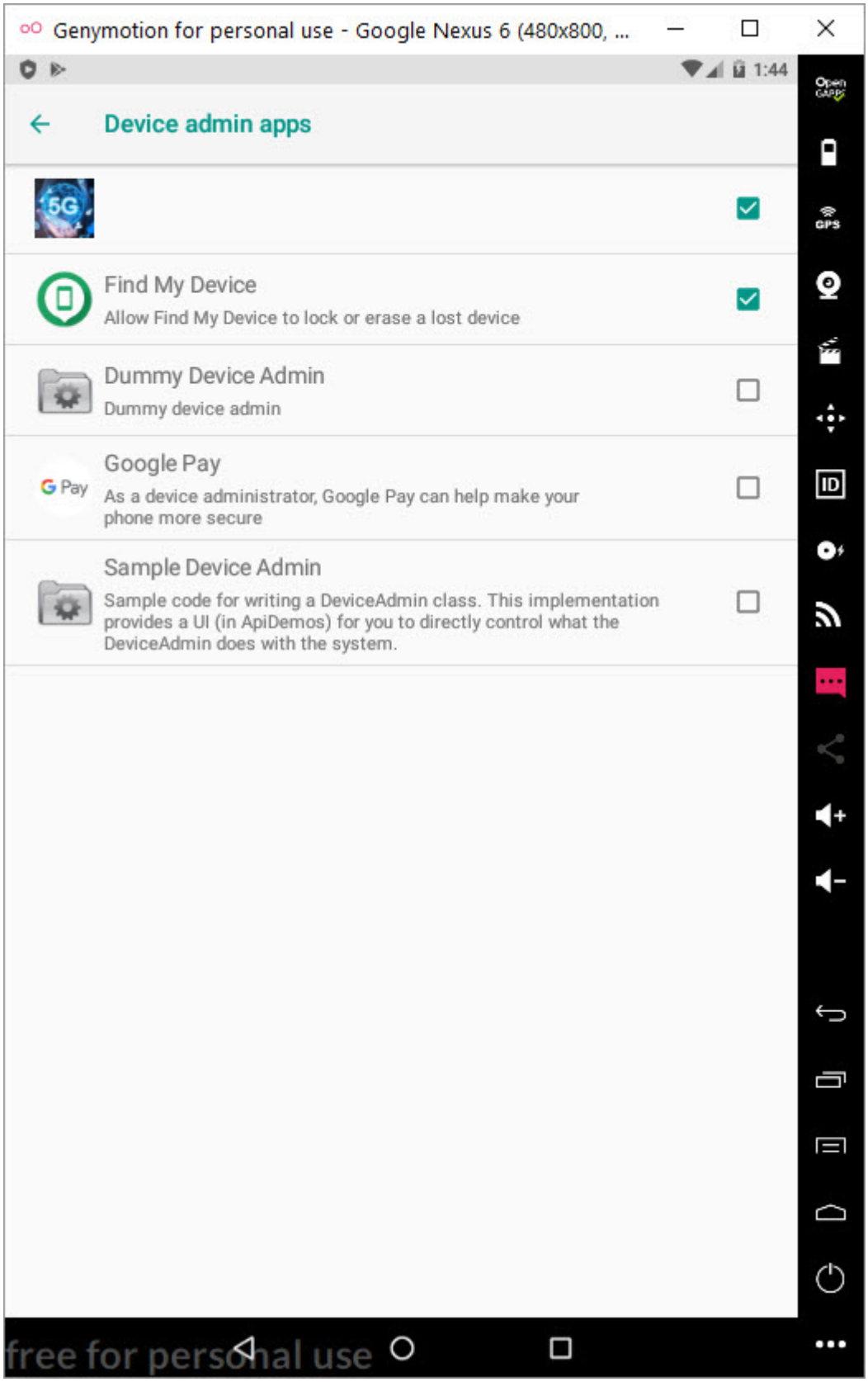
I downloaded the 5GBeta.apk file from the website and uploaded it to the Koodous web application, which is used for mobile malware analysis. The analysis failed. Then I uploaded this application to the VirusTotal web application and, although I encountered a clue that it was a banking malware (Cerberus), I did not see the address of the command and control center in the behavioral analysis output. I couldn’t find answers to the questions that came to my mind, so I decided to quickly analyze the 5GBeta.apk application dynamically using the Genymotion Android emulator.

As soon as the malicious application was installed on Android, it began to request permissions one by one to achieve its bad intentions. After obtaining the permissions and successfully completing the installation, it hid its icon and started working in the background and communicating with the command and control center with the kryll[.]ug (8[.]208.19.185) web address. When I searched the 8[.]208.19.185 IP address on VirusTotal, it was clear from the passive DNS information that it was not innocent at all.


Genymotion for personal use - Google Nexus 6 (480x800, ...
12:55
Accessibility
Accessibility shortcut
No service selected
Downloaded services
5G_
OFF
ClockBack
OFF
Magnification
OFF
QueryBack
OFF
Screen readers
Text-to-speech output
Display
Font size
Default
Display size
Default
Magnification
Off
Large mouse pointer
Enable 5G_Turkcell
Interaction controls
Click after pointer stops moving
free for personal use

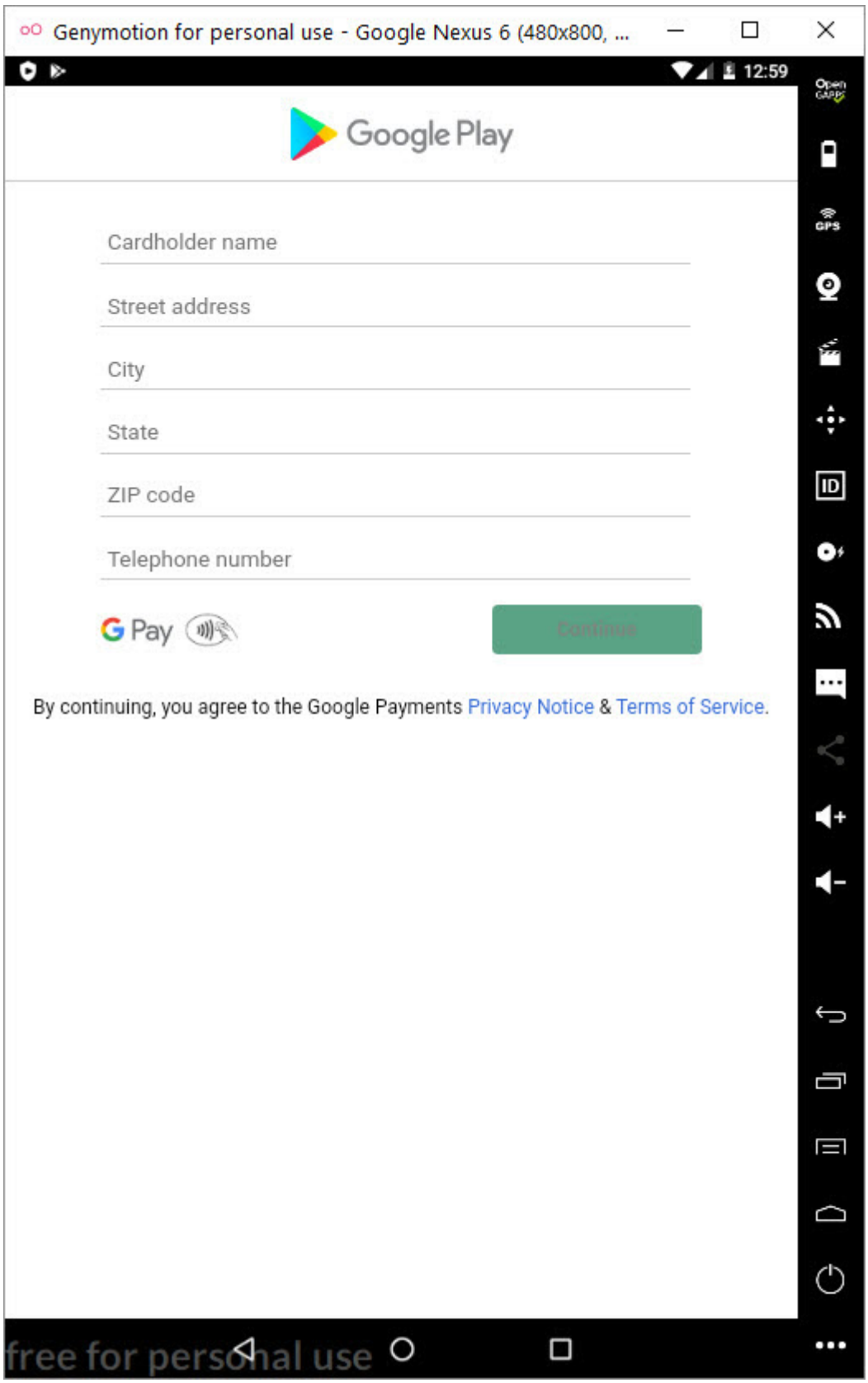

Genymotion for personal use - Google Nexus 6 (480x800, ...
12:56
5G_
Off
No description provided.
Use 5G_ ?
5G_ needs to:
Observe your actions
Receive notifications when you're interacting with an app.
Retrieve window content
Inspect the content of a window you're interacting with.
CANCEL
OK
free for personal use

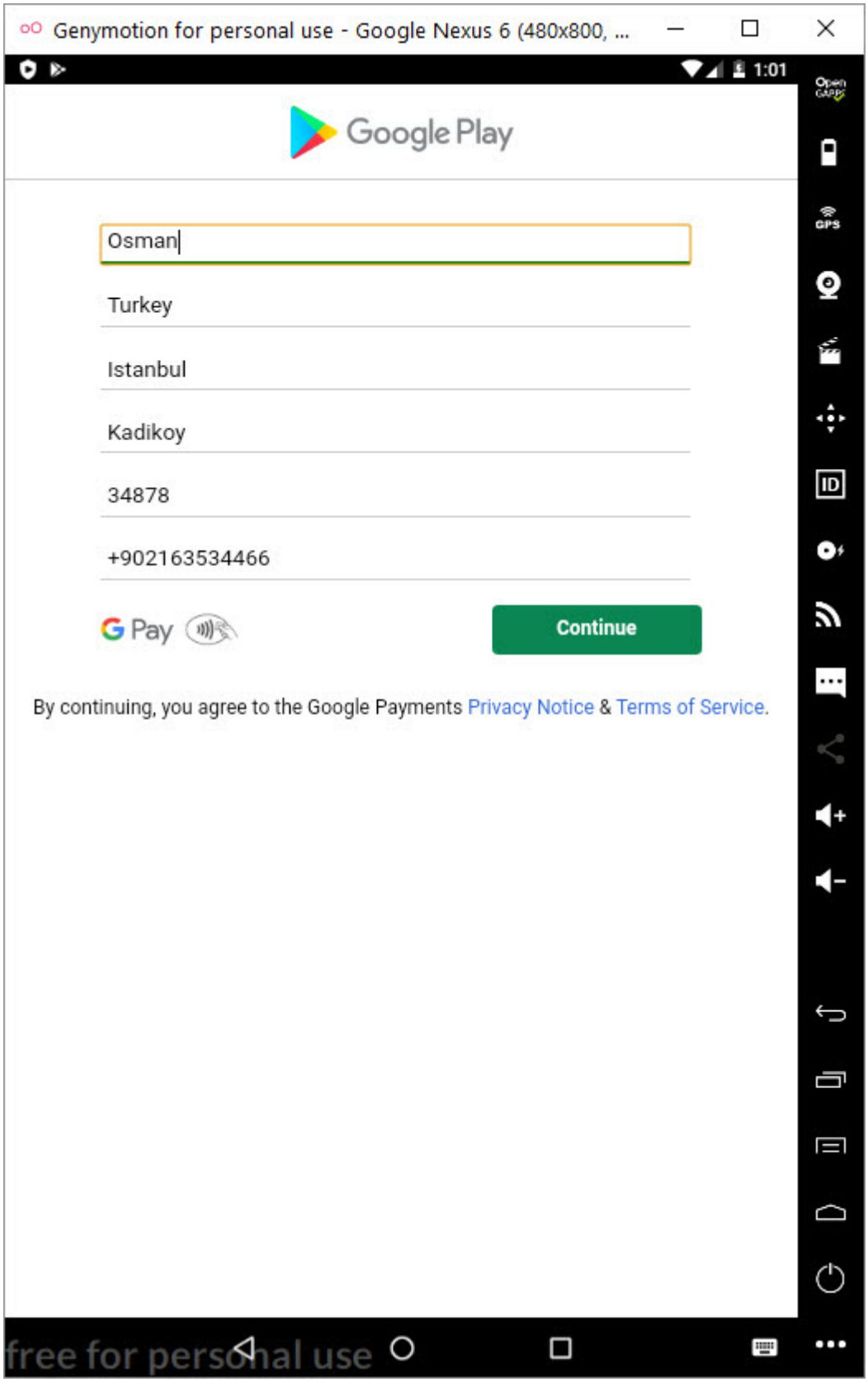

Genymotion for personal use - Google Nexus 6 - Malware...
7:42
Accessibility
Accessibility shortcut
No service selected
Downloaded services
5G_
OFF
ClockBack
OFF
Magnification
OFF
QueryBack
OFF
Screen readers
Text-to-speech output
Display
Font size
Default
Display size
Default
Magnification
Off
Large mouse pointer
Enable 5G_
Interaction controls
Click after pointer stops moving
free for personal use

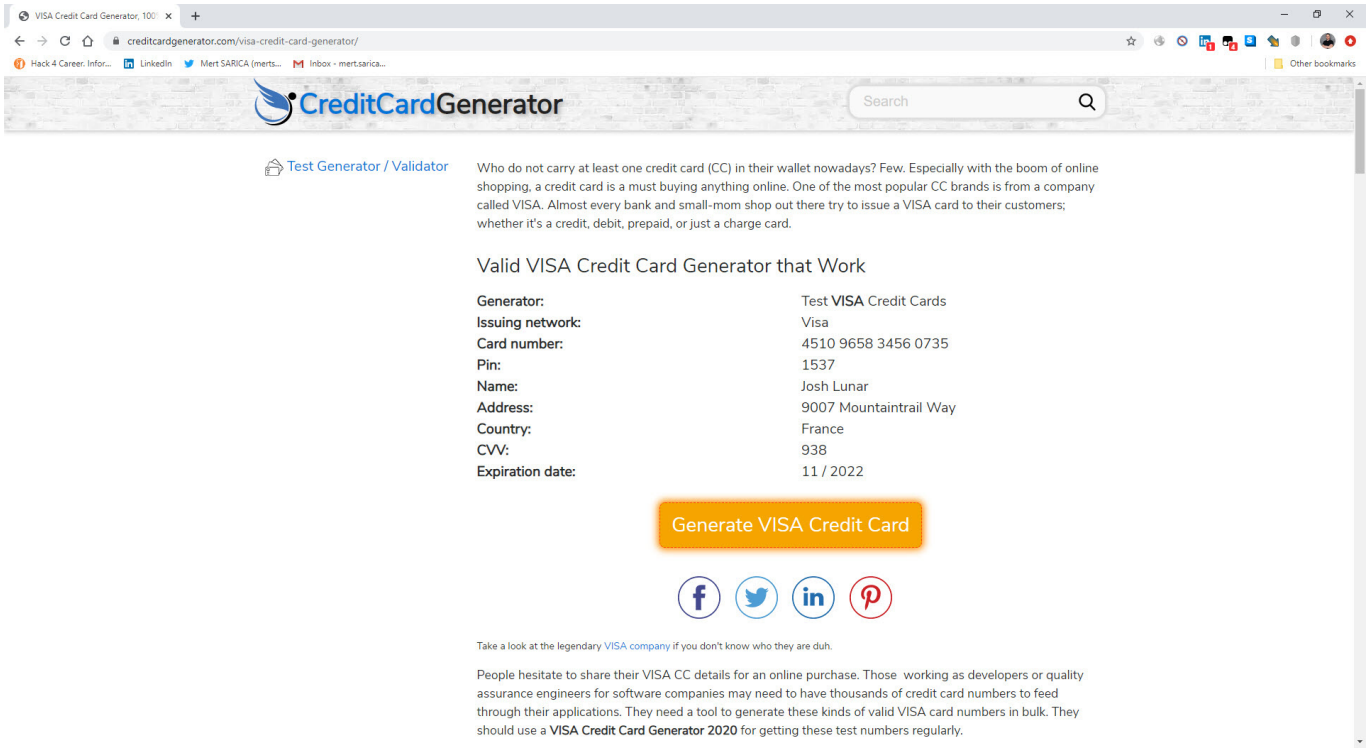


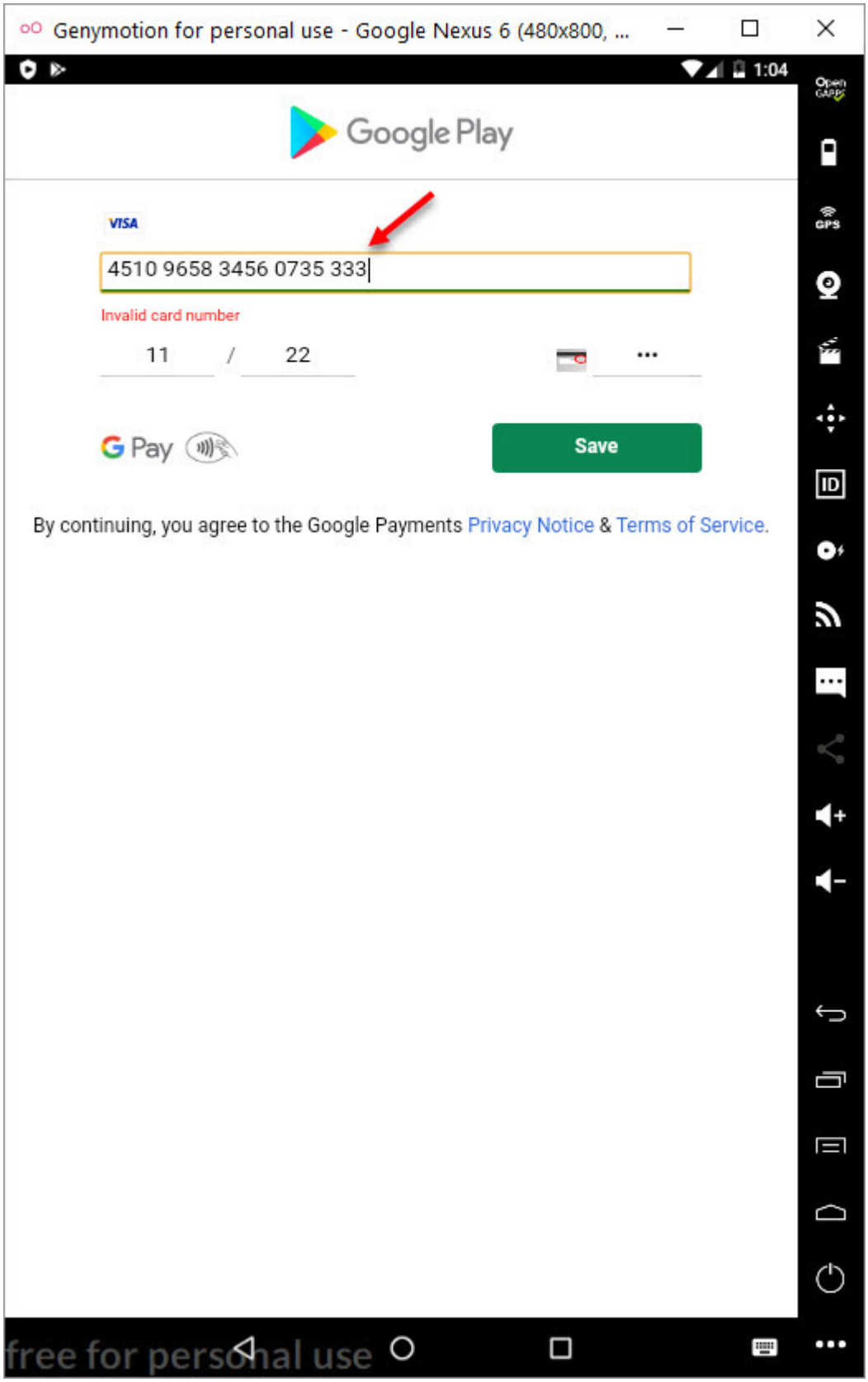
In my virtual Android operating system, a screen of Google Play, which was created to steal my credit card information, appeared as there was no banking application installed. When I entered a 16-digit credit card number created for testing, I saw that the save button (SAVE) was not activated. When I made the credit card field 19, the SAVE button became active. Probably the malicious person has made the control of the 3-digit CVV2 number specific to

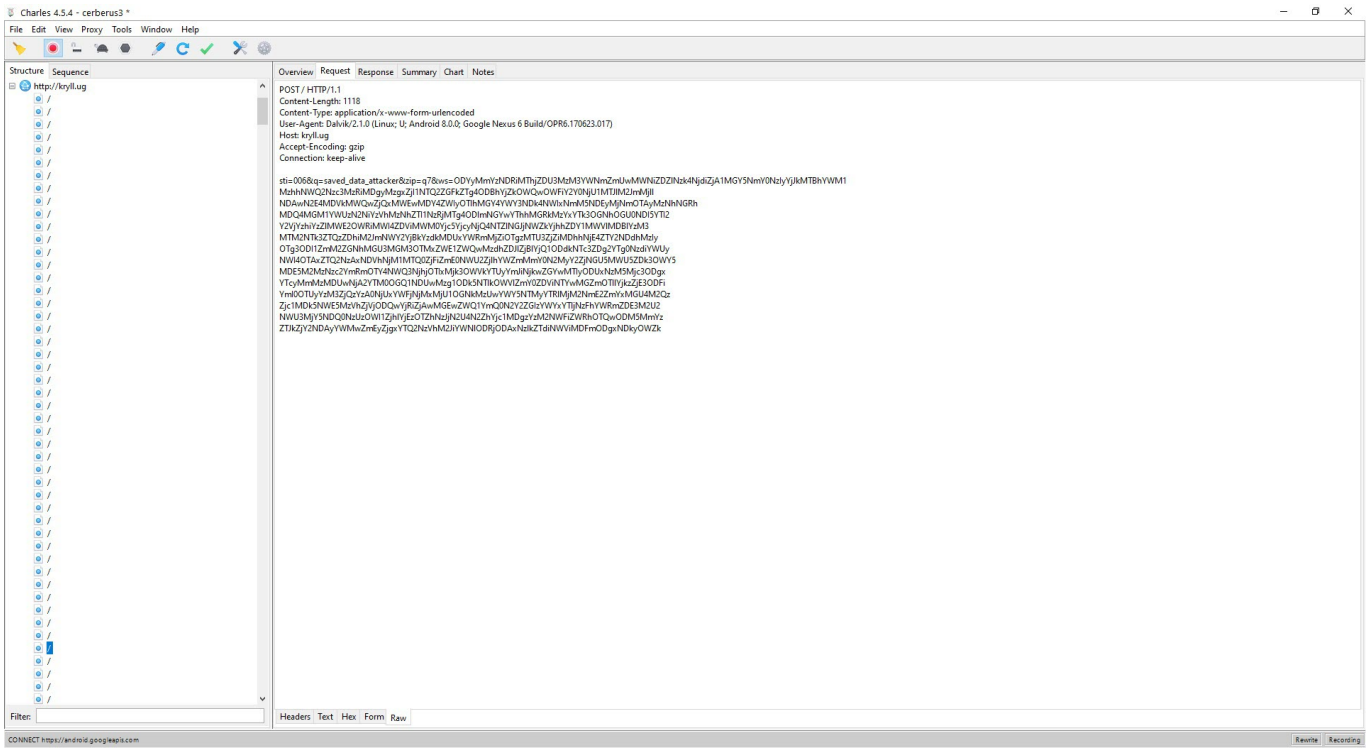
the form where the credit card number is entered and such an error has occurred. After seeing that all the information I entered went to the command and control center in encrypted form, I decided to pursue the encryption key.



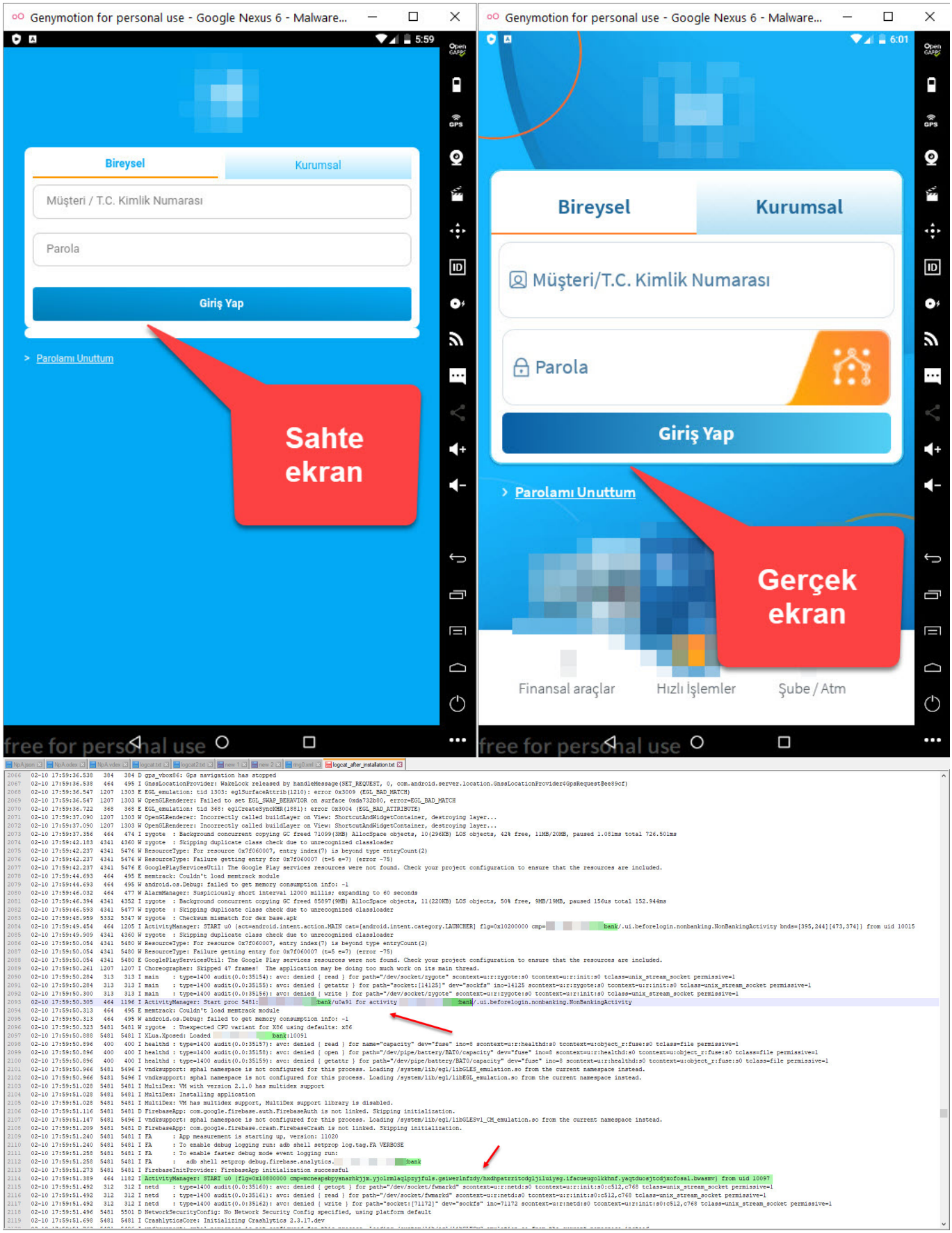

Genymotion for personal use - Google Nexus 6 (480x800, ...
1:01
Google Play
Osman
Turkey
Istanbul
Kadikoy
34878
+902163534466
G Pay
Continue
By continuing, you agree to the Google Payments Privacy Notice & Terms of Service.
free for personal use


VISA Credit Card Generator, 100
creditcardgenerator.com/visa-credit-card-generator/
Hack 4 Career. Infor...
LinkedIn
Mert SARICA (merts...
Inbox - mert.sarica...
Other bookmarks
CreditCardGenerator
Search
Test Generator / Validator
Who do not carry at least one credit card (CC) in their wallet nowadays? Few. Especially with the boom of online shopping, a credit card is a must buying anything online. One of the most popular CC brands is from a company called VISA. Almost every bank and small-mom shop out there try to issue a VISA card to their customers; whether it's a credit, debit, prepaid, or just a charge card.
Valid VISA Credit Card Generator that Work
Generator: Test VISA Credit Cards
Issuing network: Visa
Card number: 4510 9658 3456 0735
Pin: 1537
Name: Josh Lunar
Address: 9007 Mountaintrail Way
Country: France
CVV: 938
Expiration date: 11 / 2022
Generate VISA Credit Card
Take a look at the legendary VISA company if you don't know who they are duh.
People hesitate to share their VISA CC details for an online purchase. Those working as developers or quality assurance engineers for software companies may need to have thousands of credit card numbers to feed through their applications. They need a tool to generate these kinds of valid VISA card numbers in bulk. They should use a VISA Credit Card Generator 2020 for getting these test numbers regularly.


Genymotion for personal use - Google Nexus 6 (480x800, ...
1:04
Google Play
VISA
4510 9658 3456 0735 333
Invalid card number
11
/
22
G Pay
Save
By continuing, you agree to the Google Payments Privacy Notice & Terms of Service.
free for personal use

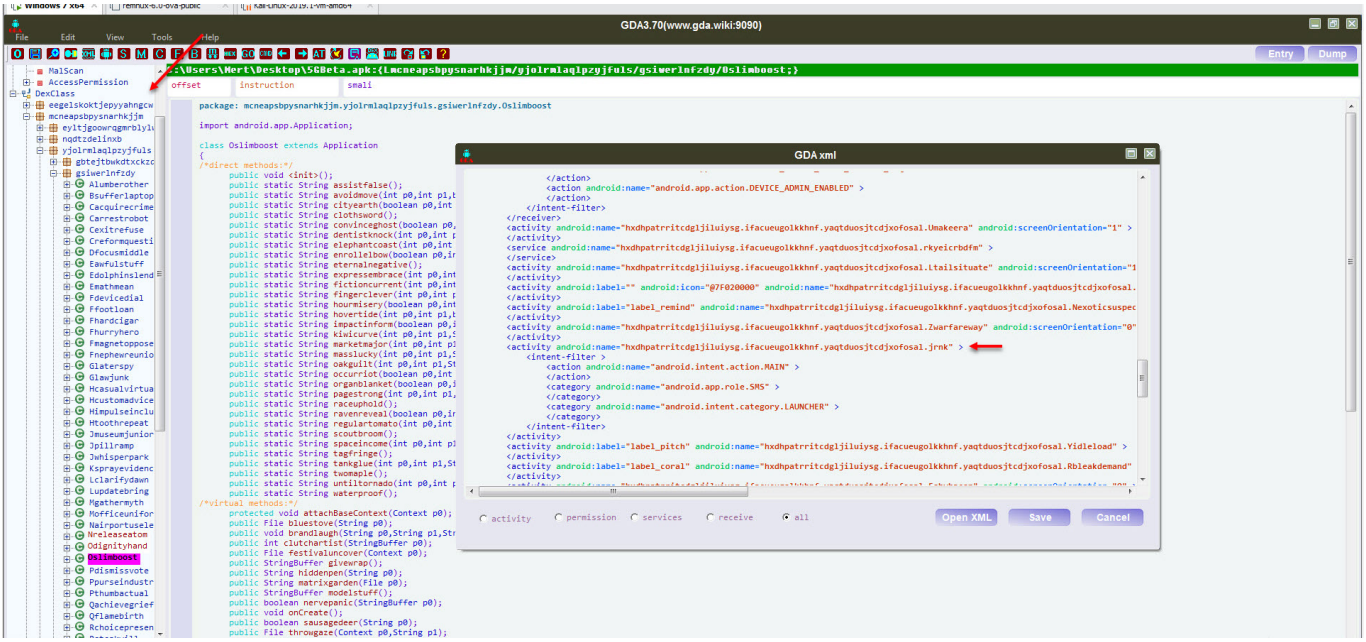


Before pursuing the encryption key, I installed and ran 10 mobile apps of different banks on my virtual system to confirm if the malicious application had stolen any banking information. As a result of my tests, the malicious application was able to steal all the information entered by the user by opening a fake screen over the login screen of the targeted mobile banking application when it was running.

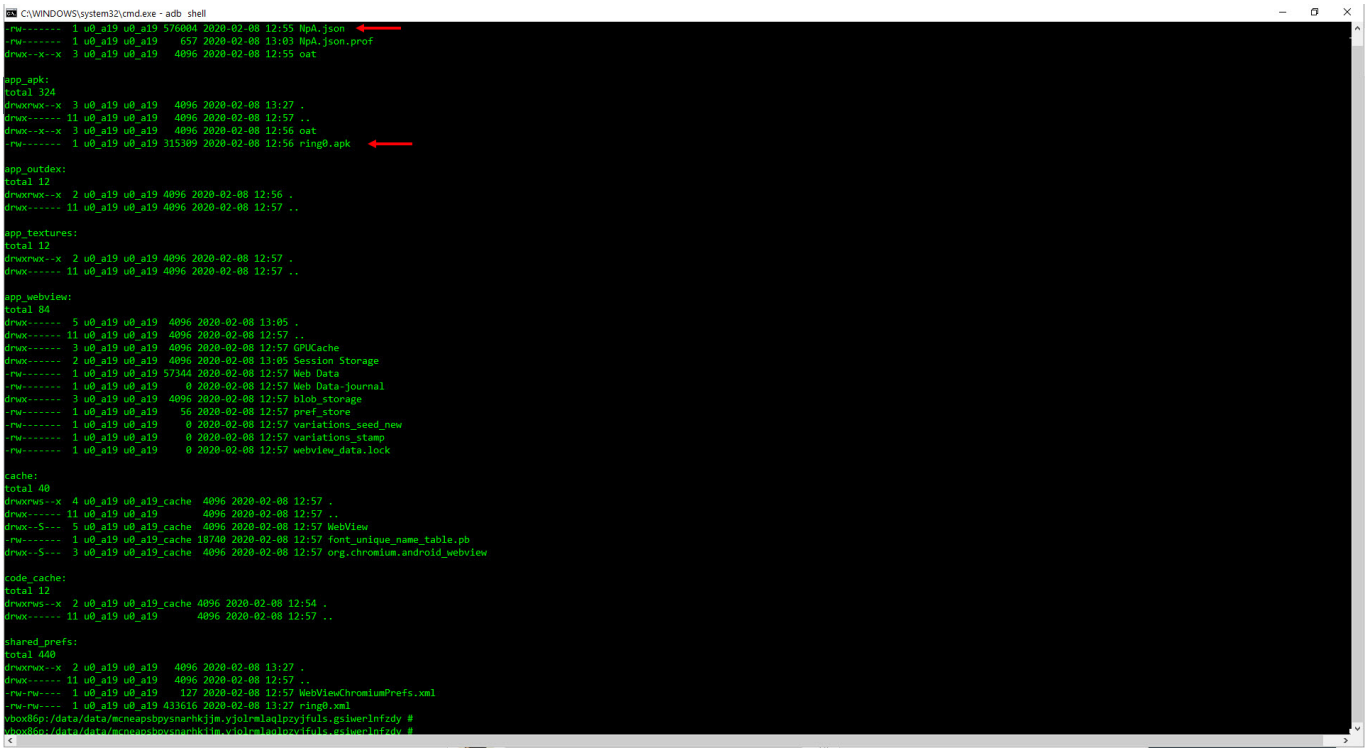


In order to find the encryption key, I used the GDA tool to decompile the 5GBeta.apk application, and by looking at the AndroidManifest.xml file which contains the basic information about the application, I saw that the classes between the MainActivity class, which will run first when the malicious

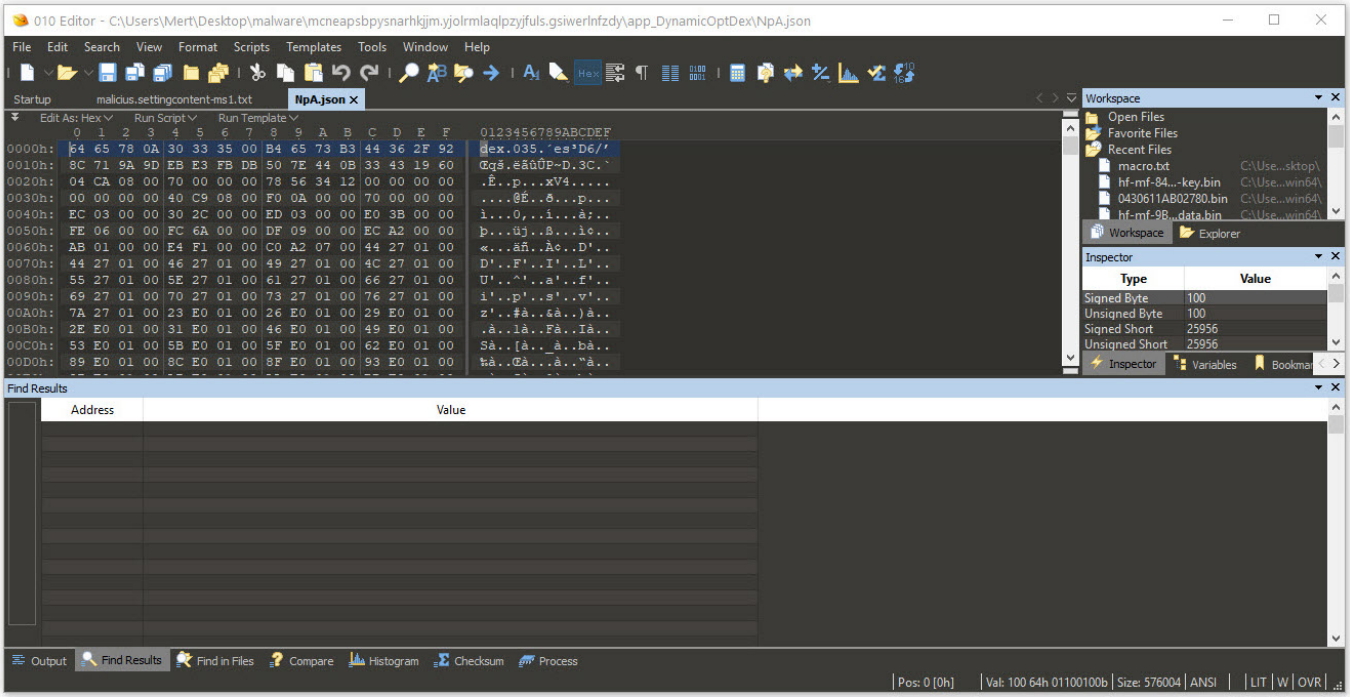
application starts, and the source code were different. This indicated that the malicious code block was loaded dynamically during runtime.

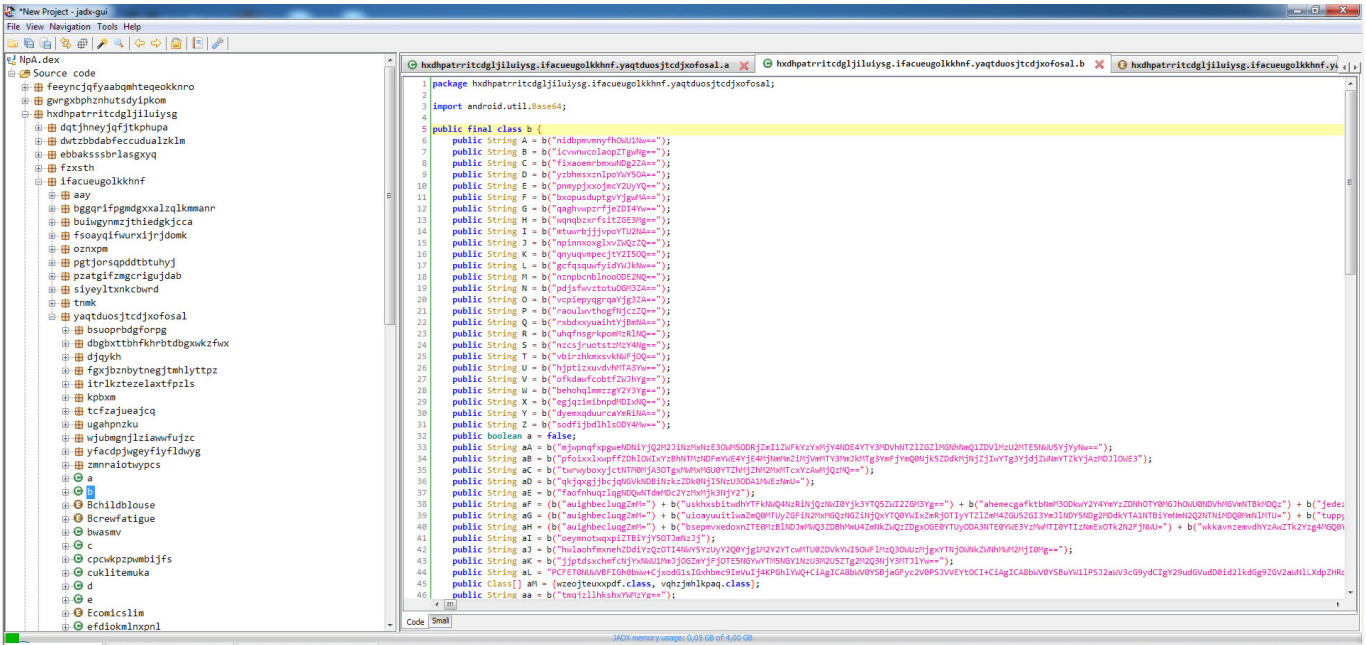


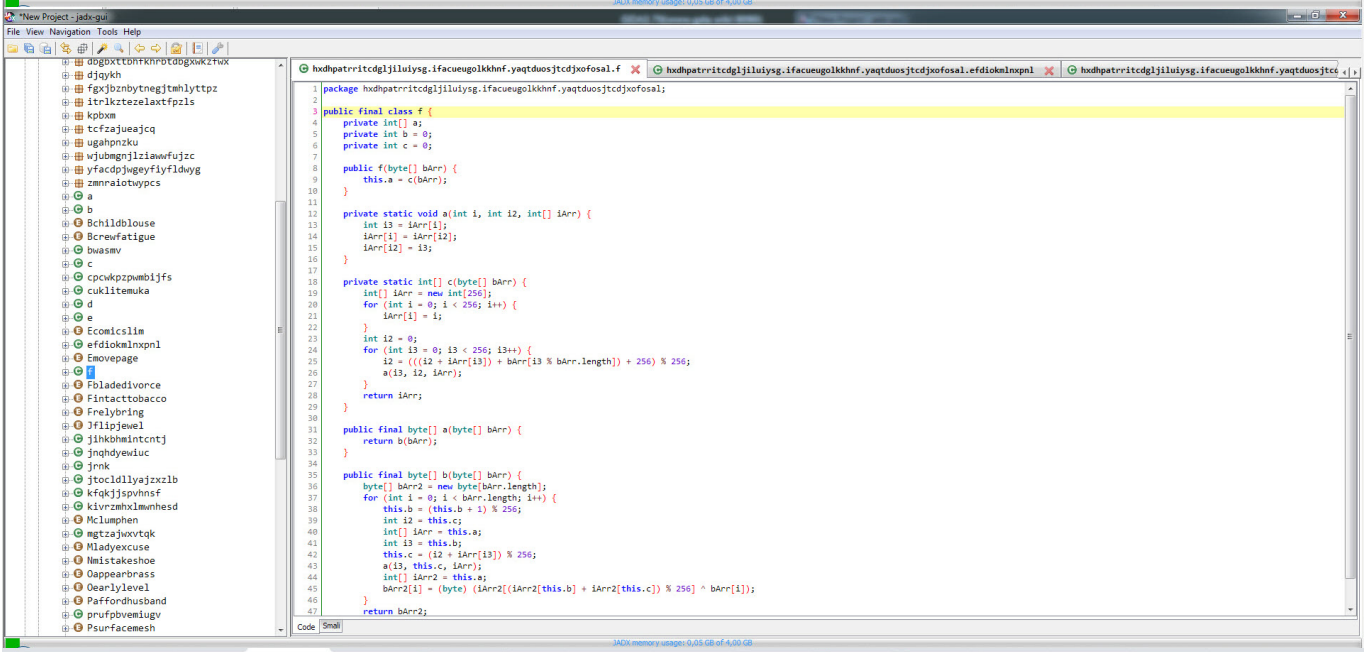
When I looked at the folder /data/data/mcneapsbpysnarhkjjm.yjolrmlaqlpzyjfuls.gsiwerlnfzdy where the malicious application was installed, I noticed the large files ring0.apk and NpA.json. I learned that NpA.json was actually a DEX file, and when I decompiled it using the jadx tool to see the source code, I encountered the MainActivity class that was present in the AndroidManifest.xml file.

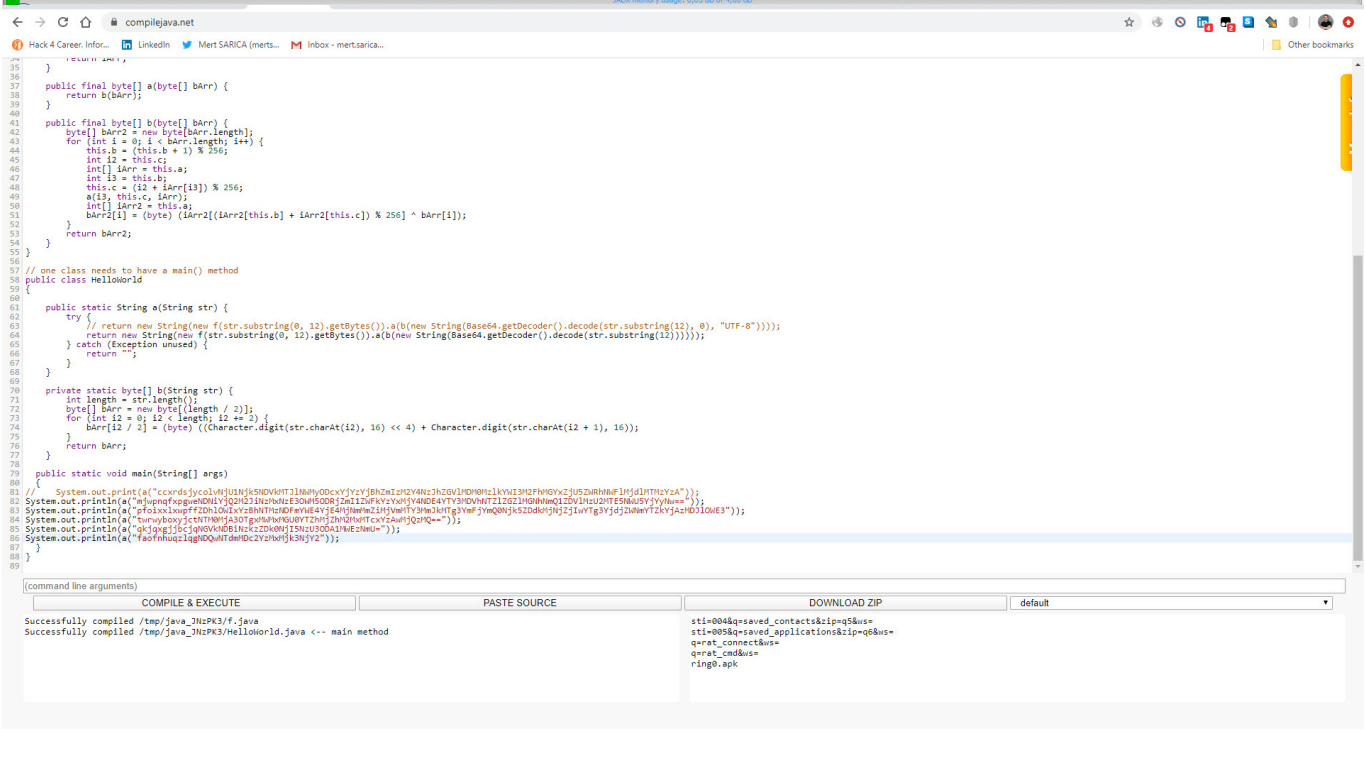


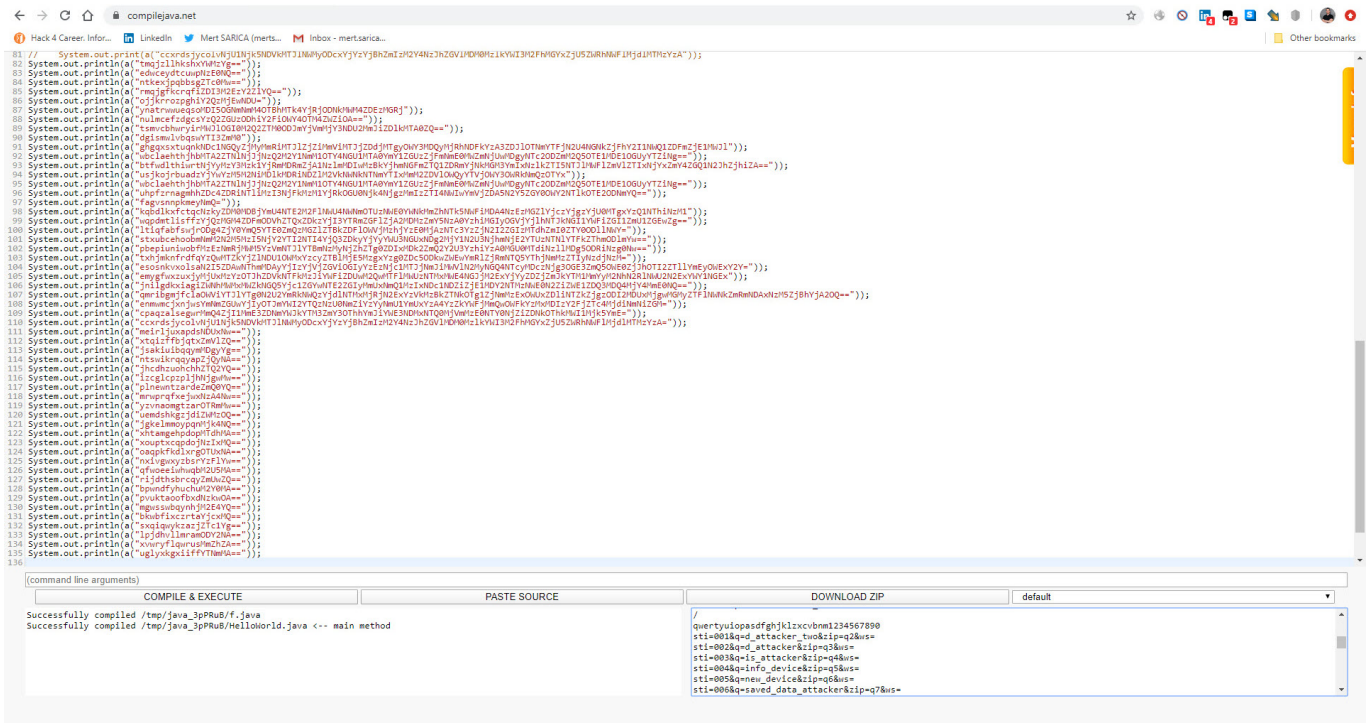
After analyzing the encrypted strings, I found that the f class is responsible for decrypting them. To do this, it takes the first 12 characters of the encrypted string as an RC4 key, and uses that key to decrypt the rest of the string, which is BASE64 decoded. (For example, if the encrypted string is mjwpnqfxpgweNDNiYjQ2M2JiNzMxNzE3OWM5ODRjZmI1ZWFkYzYxMjY4NDE4YTY3MDVhNTZlZGZlMGNhNmQ1ZDVlMzU2MTE5NWU5YjYyNw==, the RC4 encryption key is mjwpnqfxpgwe. Using this key, the rest of the characters (NDNiYjQ2M2JiNzMxNzE3OWM5ODRjZmI1ZWFkYzYxMjY4NDE4YTY3MDVhNTZlZGZlMGNhNmQ1ZDVlMzU2MTE5NWU5YjYyNw==) are decoded by BASE64 and then decrypted by the encryption key.) I was also able to easily decrypt all the strings by using the Java code in the f class and compiling it with the help of compilejava.net website.



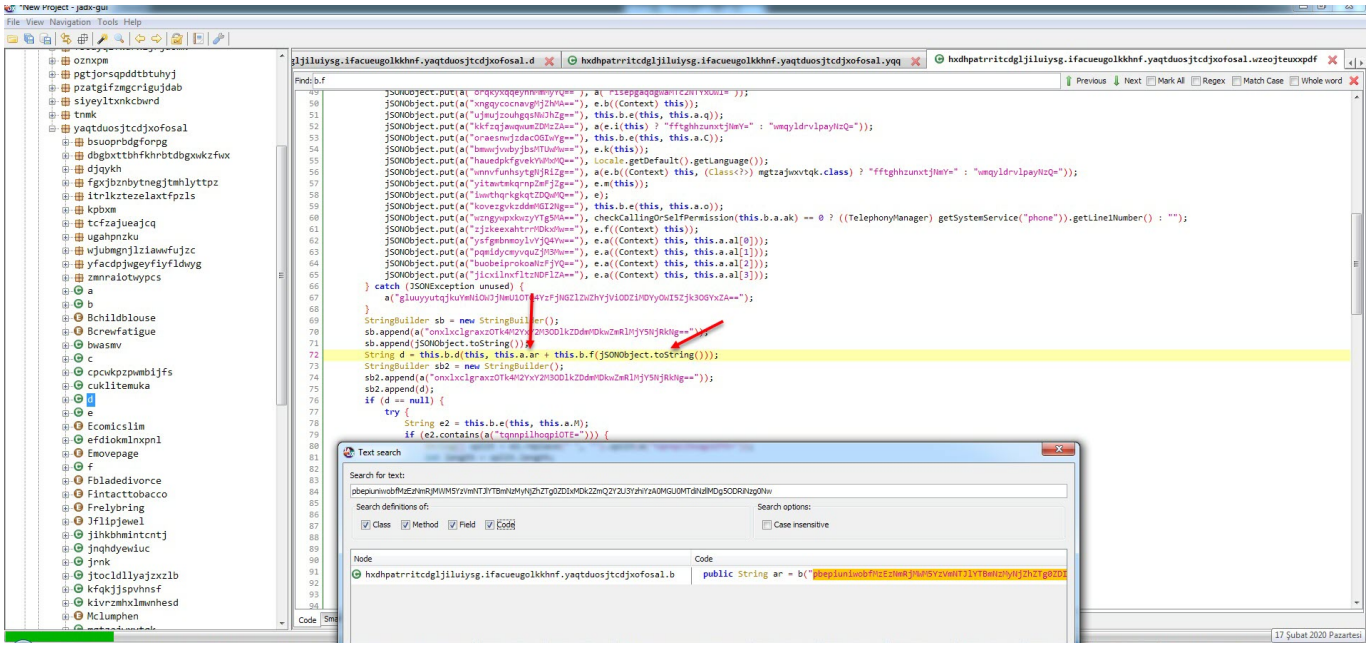

*New Project - jadx-gui
File View Navigation Tools Help
NpA.dex
Source code
hxdhpatrritcdgljliuiysg.ifacueugolkkhnf.yaqtduosjtcdjxofosal.a
hxdhpatrritcdgljliuiysg.ifacueugolkkhnf.yaqtduosjtcdjxofosal.b
package hxdhpatrritcdgljliuiysg.ifacueugolkkhnf.yaqtduosjtcdjxofosal;
import android.util.Base64;
public final class b {
Code
Smal
JADX memory usage: 0,05 GB of 4,00 GB

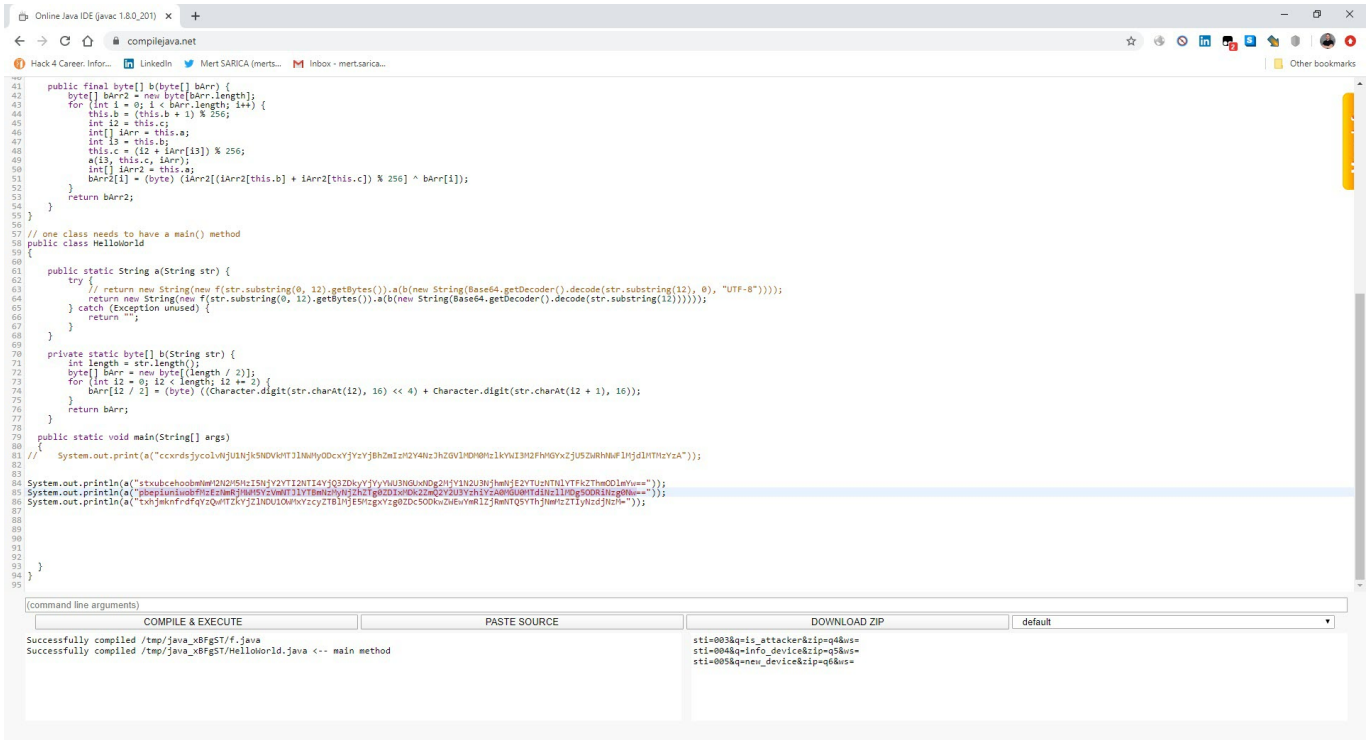

*New Project - jadx-gui
File View Navigation Tools Help
hxdhpatrritcdgljliuiysg.ifacueugolkkhnf.yaqtduosjtcdjxofosal.f
hxdhpatrritcdgljliuiysg.ifacueugolkkhnf.yaqtduosjtcdjxofosal.efdiokmlnxpnl
package hxdhpatrritcdgljliuiysg.ifacueugolkkhnf.yaqtduosjtcdjxofosal;
public final class f {
Code
Smal
JADX memory usage: 0,05 GB of 4,00 GB

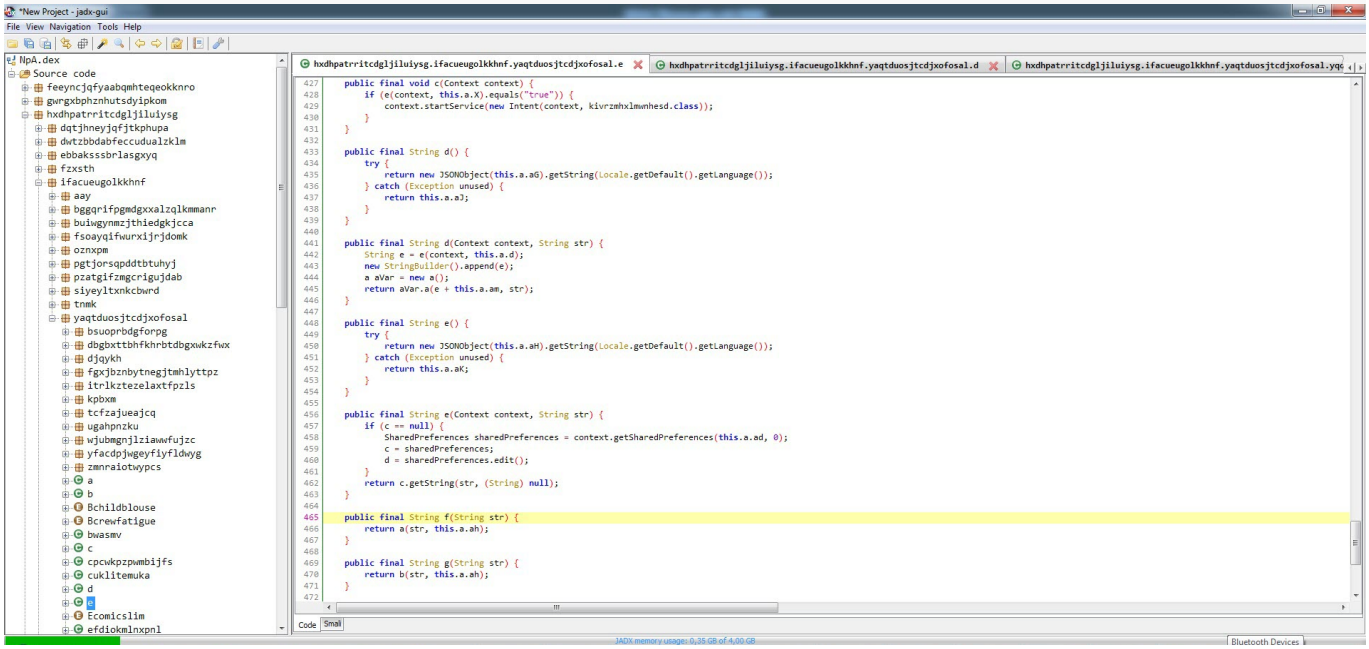

compilejava.net
(command line arguments)
COMPILE & EXECUTE
PASTE SOURCE
DOWNLOAD ZIP
default
Successfully compiled /tmp/java_JNzPK3/f.java
Successfully compiled /tmp/java_JNzPK3/HelloWorld.java <-- main method

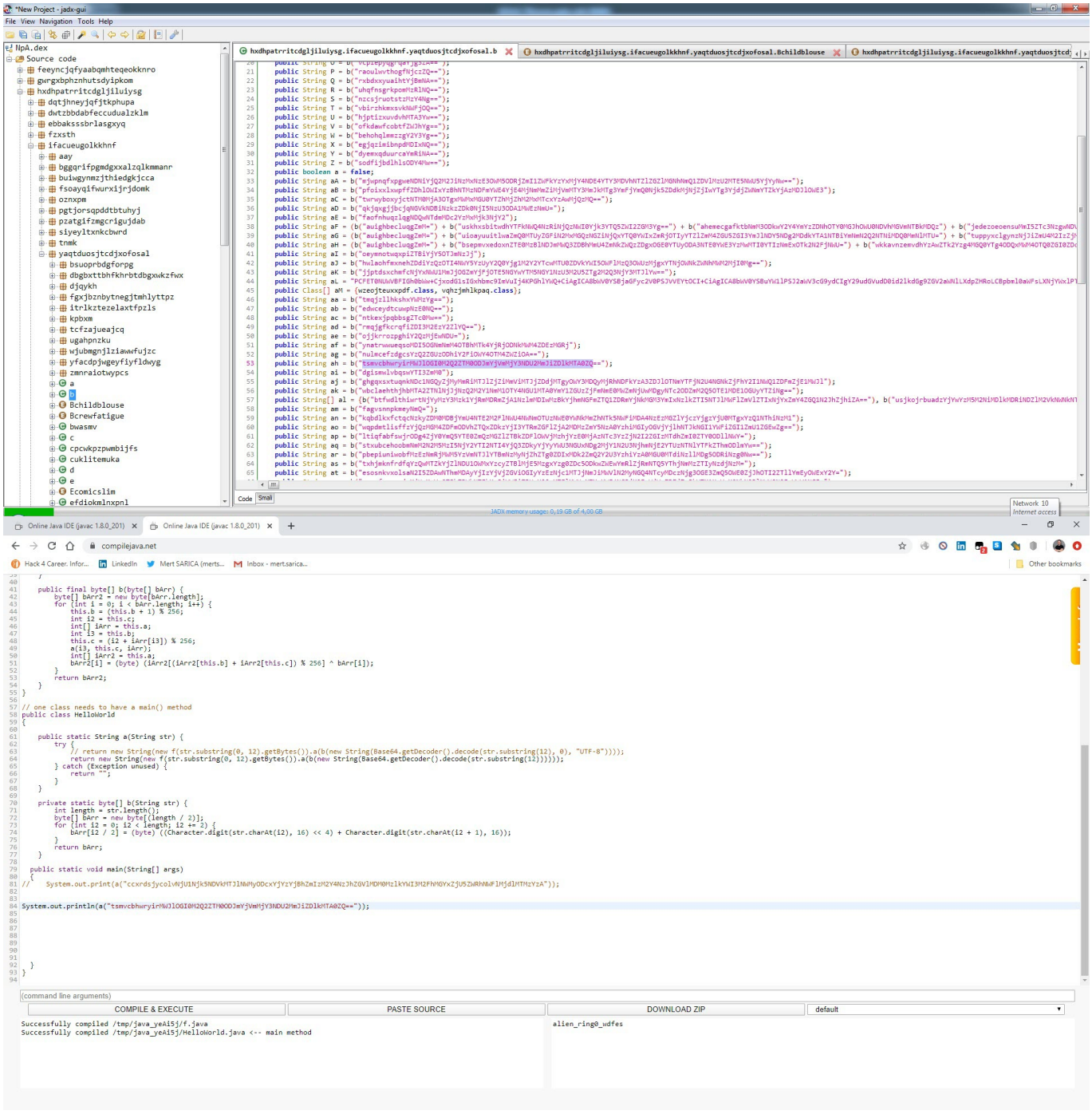


After observing that the encrypted data sent to the command and control center was in JSON format, I focused on the parts of the code where this class was used. Since I knew that the malicious application sent the sti=004&q=info_device parameter to the command and control center at certain intervals, and then the encrypted ws= parameter, I found the code block where these two values were concatenated. Upon analyzing this code block, I found that the ws parameter was encrypted with the alien_ring0_wdfes RC4 key.

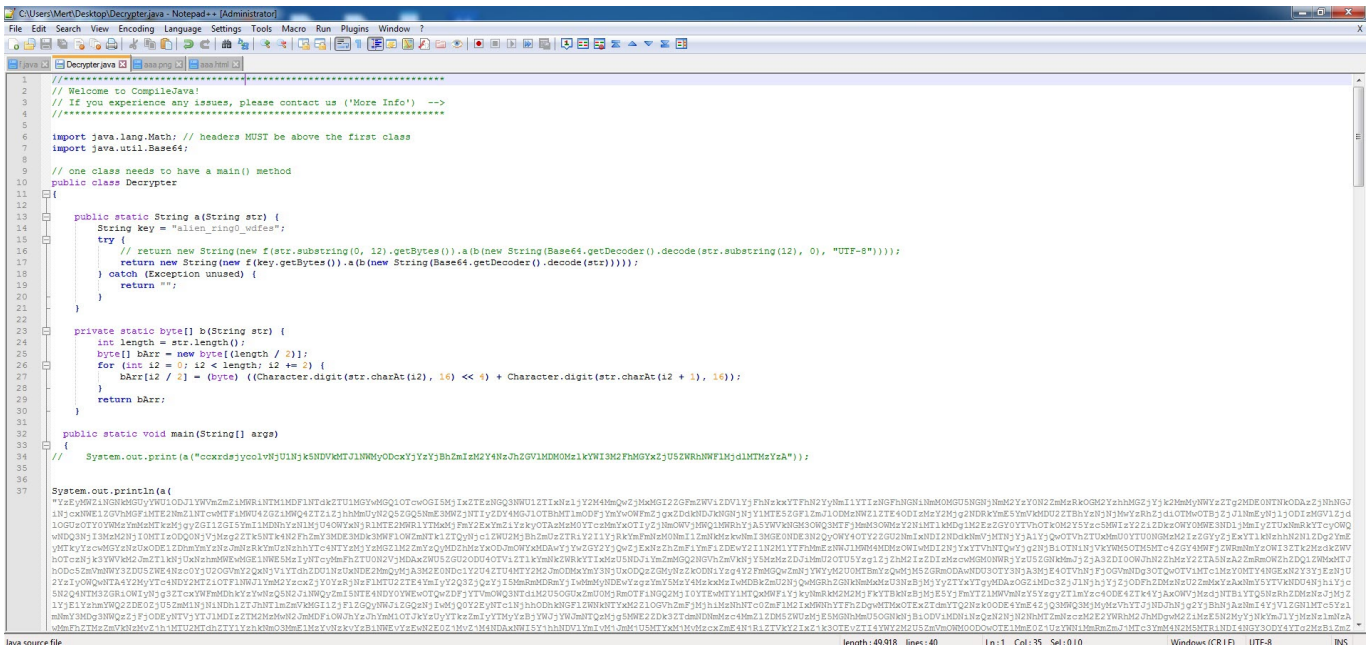


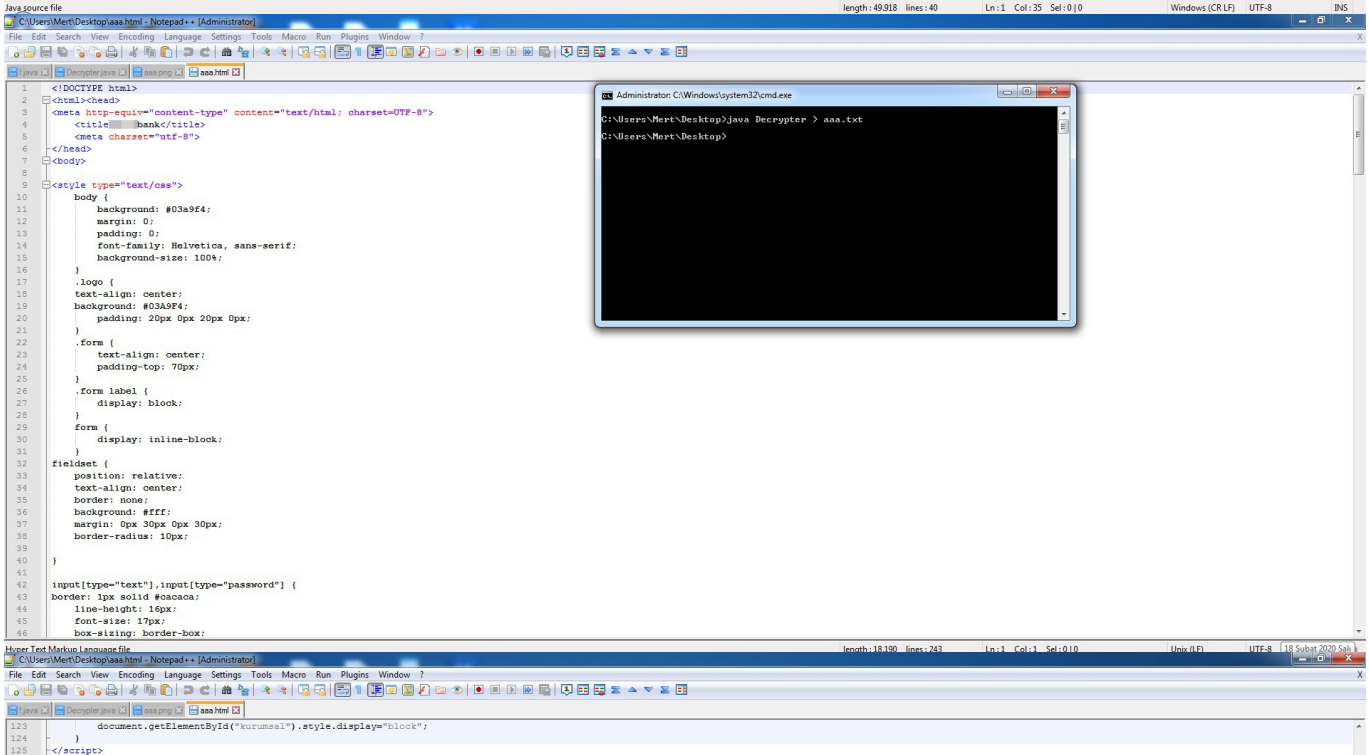

Online Java IDE (javac 1.8.0_201)
compilejava.net
Hack 4 Career. Infor...
LinkedIn
Mert SARICA (merts...
Inbox - mert.sarica...
Other bookmarks
(command line arguments)
COMPILE & EXECUTE
PASTE SOURCE
DOWNLOAD ZIP
default
Successfully compiled /tmp/java_xBFgST/f.java
Successfully compiled /tmp/java_xBFgST/HelloWorld.java <-- main method
sti=003&q=is_attacker&zip=q4&ws=
sti=004&q=info_device&zip=q5&ws=
sti=005&q=new_device&zip=q6&ws=

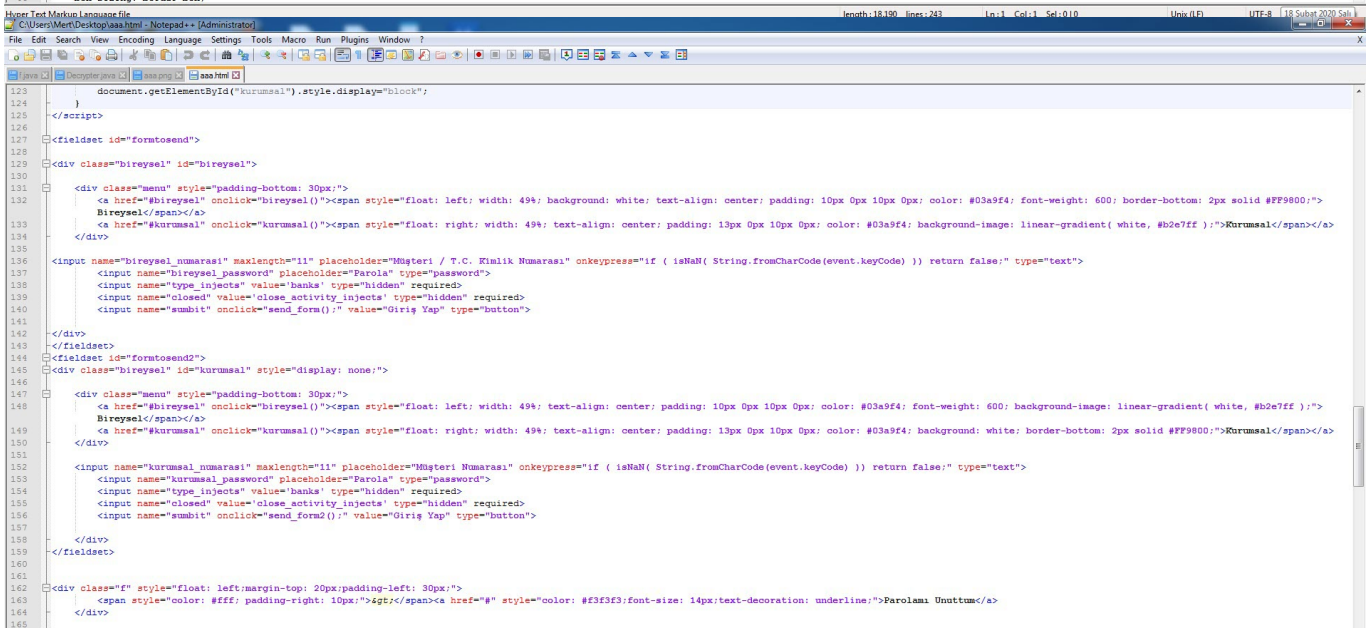

*New Project - jadx-gui
File View Navigation Tools Help
NpA.dex
Source code
hxdhpatrritcdgljiluiysg.ifacueugolkkhnf.yaqtduosjtcdjxofosal.e
hxdhpatrritcdgljiluiysg.ifacueugolkkhnf.yaqtduosjtcdjxofosal.d
Code
Smali
Bluetooth Devices



When it came to decrypting the encrypted data I previously obtained with the alien_ring0_wdfes encryption key, and confirming the validity of the encryption key, I also learned that the fake screens (html) came from the command and control center, as mentioned at the beginning of the text.”


C:\Users\Mert\Desktop\Decrypter.java - Notepad++ [Administrator]
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
Decrypter.java
aaa.png
aaa.html
//**********
// Welcome to CompileJava!
// If you experience any issues, please contact us ('More Info') -->
//**********
import java.lang.Math; // headers MUST be above the first class
import java.util.Base64;
// one class needs to have a main() method
public class Decrypter
{
public static String a(String str) {
String key = "alien_ring0_wdfes";
try {
// return new String(new f(str.substring(0, 12).getBytes()).a(b(new String(Base64.getDecoder().decode(str.substring(12), 0), "UTF-8"))));
return new String(new f(key.getBytes()).a(b(new String(Base64.getDecoder().decode(str))))); } catch (Exception unused) {
return "";
}
}
private static byte[] b(String str) {
int length = str.length();
byte[] bArr = new byte[(length / 2)];
for (int i2 = 0; i2 < length; i2 += 2) {
bArr[i2 / 2] = (byte) ((Character.digit(str.charAt(i2), 16) << 4) + Character.digit(str.charAt(i2 + 1), 16));
}
return bArr;
}
public static void main(String[] args)
{
//   System.out.print(a("ccxrdsjycolvNjUiNjk5NDVkMTJlNWMyODcxYjYzYjBhZmIzM2Y4NzJhZGVlMDM0Mzlk YWI3M2FhMGYxZjU5ZWRhNWFlMjdlMTMzYzA"));
System.out.println(a(
Java source file
length : 49,918 lines : 40
Ln : 1 Col : 35 Sel : 0 | 0
Windows (CR LF)
UTF-8
INS


C:\Users\Mert\Desktop\aaa.html - Notepad++ [Administrator]
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
aaa.html
<!DOCTYPE html>
<html><head>
<meta http-equiv="content-type" content="text/html; charset=UTF-8">
<title bank</title>
<meta charset="utf-8">
</head>
<body>
<style type="text/css">
body {
background: #03a9f4;
margin: 0;
padding: 0;
font-family: Helvetica, sans-serif;
background-size: 100%;
}
.logo {
text-align: center;
background: #03A9F4;
padding: 20px 0px 20px 0px;
}
.form {
text-align: center;
padding-top: 70px;
}
.form label {
display: block;
}
form {
display: inline-block;
}
fieldset {
position: relative;
text-align: center;
border: none;
background: #fff;
margin: 0px 30px 0px 30px;
border-radius: 10px;
}
input[type="text"],input[type="password"] {
border: 1px solid #cacaca;
line-height: 16px;
font-size: 17px;
box-sizing: border-box;
Administrator: C:\Windows\system32\cmd.exe
C:\Users\Mert\Desktop>java Decrypter > aaa.txt
C:\Users\Mert\Desktop>
Hyper Text Markup Language file
length : 18,190 lines : 243
Ln : 1 Col : 1 Sel : 0 | 0
Unix (LF)
UTF-8
18 Subat 2020 Sal


C:\Users\Mert\Desktop\aaa.html - Notepad++ [Administrator]
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
aaa.html
document.getElementById("kurumsal").style.display="block";
}
</script>
<fieldset id="formtosend">

Bireysel
Kurumsal

<input name="bireysel_numarasi" maxlength="11" placeholder="Müşteri / T.C. Kimlik Numarası" onkeypress="if (isNaN(String.fromCharCode(event.keyCode))) return false;" type="text">
<input name="bireysel_password" placeholder="Parola" type="password">
<input name="type_injects" value='banks' type="hidden" required>
<input name="closed" value='close_activity_injects' type="hidden" required>
<input name="sumbit" onclick="send_form();" value="Giriş Yap" type="button">

</fieldset>
<fieldset id="formtosend2">

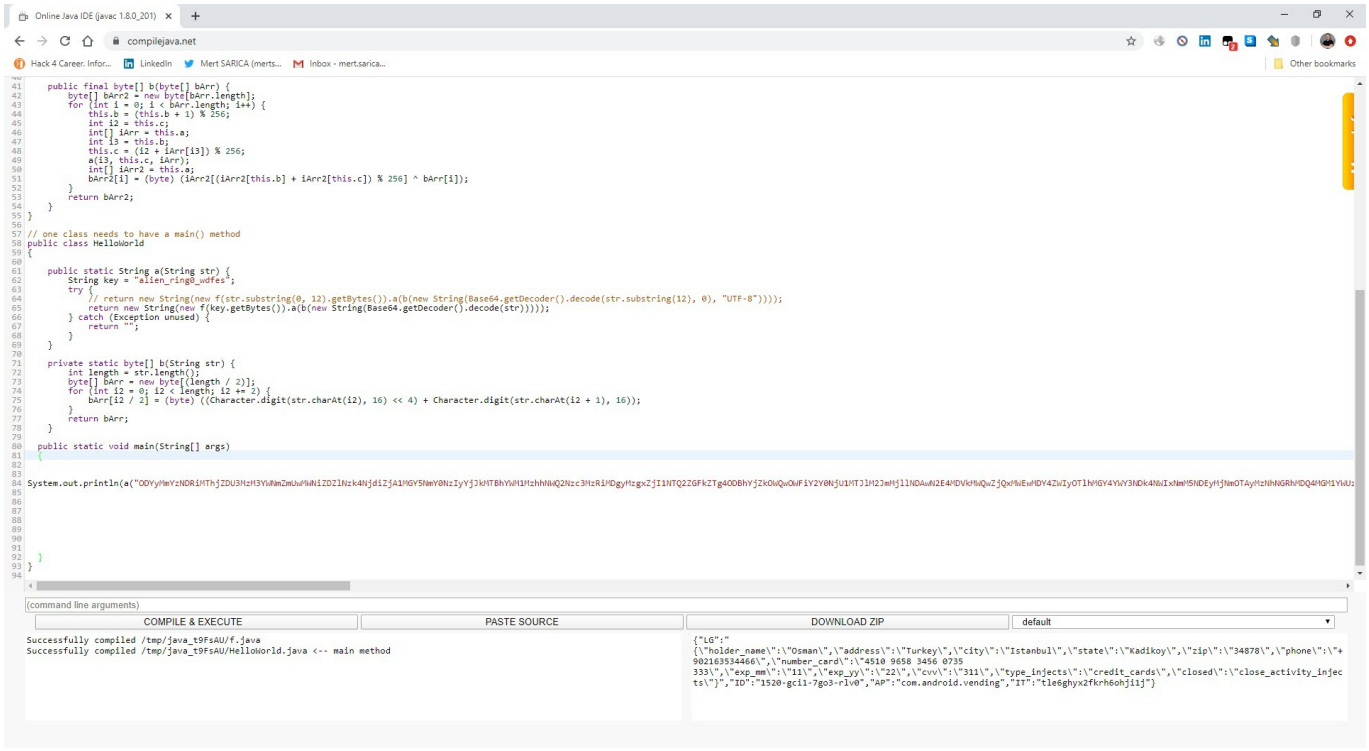
Bireysel
Kurumsal

<input name="kurumsal_numarasi" maxlength="11" placeholder="Müşteri Numarası" onkeypress="if (isNaN(String.fromCharCode(event.keyCode))) return false;" type="text">
<input name="kurumsal_password" placeholder="Parola" type="password">
<input name="type_injects" value='banks' type="hidden" required>
<input name="closed" value='close_activity_injects' type="hidden" required>
<input name="sumbit" onclick="send_form2();" value="Giriş Yap" type="button">

</fieldset>

>Parolamı Unuttum

<script>
Solve PC issues: 2 important messages



In conclusion, it was surprising and concerning to learn that the Cerberus mobile banking malware, which has been frequently talked about in recent years for its features and name, has started targeting citizens with SMS containing their name and surname. As always, I stress that Android users should avoid installing apps from unknown sources.

Hope to see you in the following articles.

Note:

1. After this security research (February 2020), it was revealed that (September 2020) this malware, commonly known as Cerberus, later branching off into a malware called Alien from the fork of Cerberus v1 version.
2. This text also includes the solution for the Pi Hediyem Var #18 cybersecurity game.