REPORT BY SAPTARSHI CHATTERJEE
DATE:02/4/2020
CLIENT:CYBER KINGS INDIA
APP NAME: SPIKE
Platform: Android
Package Name: com.pingapp.app
Package Version Name: 2.8.8.4
Package Version Code: 172
Min Sdk: 16
Target Sdk: 29
MD5 : c89356c0a431c74fda864b651e79ddd4
SHA1 : 19c0aa493cda1b872573cbe7ddd7d606e08ea10f
SHA256: f5da1bc8e0b87aca38128628acd417152ca58058233d3378e5efabcd9fb66a73
SHA512:
134907fc25bbc0a93e3cb48107c740e4beaefd838dc958ef5a24019a6d878d2ef5f3696543da80c9dd73
6a752002cba31156d16cfed157ccd42e29c487fb4a53
Analyze Signature:
ac392baaf8f7f40c3f5902f505098367f518e5d7909226e293f808474ef92419b3d08e486b1ce7410f4129
605f6c432008beac34c20ed669ec8f198a06bd930a

[Critical] <Implicit_Intent> Implicit Service Checking:
To ensure your app is secure, always use an explicit intent when starting a Service and DO NOT declare intent filters for your
services. Using an implicit intent to start a service is a security hazard because you cannot be certain what service will
respond to the intent, and the user cannot see which service starts.
Reference: http://developer.android.com/guide/components/intents-filters.html#Types
=> com.pingapp.gcmjs2.GcmListener
=> com.pingapp.gcmjs2.InstanceIDListener
=> com.google.firebase.iid.FirebaseInstanceIdService
=> com.google.firebase.messaging.FirebaseMessagingService
[Warning] External Storage Accessing:
External storage access found (Remember DO NOT write important files to external storages):
=> Lorg/appcelerator/titanium/util/TiFileHelper;->getDataDirectory(Z)Ljava/io/File; (0x2a) --->
Landroid/os/Environment;->getExternalStorageDirectory()Ljava/io/File;
=> Lti/modules/titanium/TitaniumModule;->dumpCoverage()V (0x28) --->
Landroid/os/Environment;->getExternalStorageDirectory()Ljava/io/File;
[Warning] AndroidManifest Exported Components Checking:
Found "exported" components(except for Launcher) for receiving outside applications' actions (AndroidManifest.xml).
These components can be initilized by other apps. You should add or modify the attribute to [exported="false"] if you don't want
to.
You can also protect it with a customized permission with "signature" or higher protectionLevel and specify in
"android:permission" attribute.
activity => com.pingapp.hopandroid2.SearchActivity
activity => com.pingapp.hopandroid2.NewMessageActivity
activity => com.pingapp.hopandroid2.SettingsActivity
activity => com.pingapp.hopandroid2.SendLogActivity
activity => com.pingapp.app.PreviewActivity
service => com.pingapp.gcmjs2.GcmListener
service => com.pingapp.gcmjs2.InstanceIDListener
service => com.google.firebase.iid.FirebaseInstanceIdService
service => com.google.firebase.messaging.FirebaseMessagingService

[Warning] <Sensitive_Information> Getting ANDROID_ID:
This app has code getting the 64-bit number "Settings.Secure.ANDROID_ID".
ANDROID_ID seems a good choice for a unique device identifier. There are downsides: First, it is not 100% reliable on releases of
Android prior to 2.2 (Froyo).
Also, there has been at least one widely-observed bug in a popular handset from a major manufacturer, where every instance has
the same ANDROID_ID.
If you want to get an unique id for the device, we suggest you use "Installation" framework in the following article.
Please check the reference:
http://android-developers.blogspot.tw/2011/03/identifying-app-installations.html
=> Lcom/appcelerator/aps/APSAnalyticsHelper;->init(Ljava/lang/String; Landroid/content/Context;)V (0xe) --->

Landroid/provider/Settings$Secure;->getString(Landroid/content/ContentResolver; Ljava/lang/String;)Ljava/lang/String;
[Notice] AndroidManifest Adb Backup Checking:
ADB Backup is ENABLED for this app (default: ENABLED). ADB Backup is a good tool for backing up all of your files. If it's open
for this app, people who have your phone can copy all of the sensitive data for this app in your phone (Prerequisite: 1.Unlock
phone's screen 2.Open the developer mode). The sensitive data may include lifetime access token, username or password, etc.
Security case related to ADB Backup:
1.http://www.securityfocus.com/archive/1/530288/30/0/threaded

2.http://blog.c22.cc/advisories/cve-2013-5112-evernote-android-insecure-storage-of-pin-data-bypass-of-pin-protection/
3.http://nelenkov.blogspot.co.uk/2012/06/unpacking-android-backups.html
Reference:
http://developer.android.com/guide/topics/manifest/application-element.html#allowbackup
[Notice] File Unsafe Delete Checking:
Everything you delete may be recovered by any user or attacker, especially rooted devices.
Please make sure do not use "file.delete()" to delete essential files.
Check this video: https://www.youtube.com/watch?v=tGw1fxUD-uY
=> Lio/requery/android/database/sqlite/SQLiteDatabase;->deleteDatabase(Ljava/io/File;)Z (0x14) ---> Ljava/io/File;->delete()Z
=> Lio/requery/android/database/sqlite/SQLiteDatabase;->deleteDatabase(Ljava/io/File;)Z (0x54) ---> Ljava/io/File;->delete()Z
=> Lio/requery/android/database/sqlite/SQLiteDatabase;->deleteDatabase(Ljava/io/File;)Z (0x96) ---> Ljava/io/File;->delete()Z
=> Lio/requery/android/database/sqlite/SQLiteDatabase;->deleteDatabase(Ljava/io/File;)Z (0xd8) ---> Ljava/io/File;->delete()Z
=> Lio/requery/android/database/sqlite/SQLiteDatabase;->deleteDatabase(Ljava/io/File;)Z (0x13a) ---> Ljava/io/File;->delete()Z
=> Lorg/appcelerator/kroll/common/LogFileCollector;->cycleFiles()V (0x120) ---> Ljava/io/File;->delete()Z
=> Lorg/appcelerator/kroll/common/LogFileCollector;->getLogfile()Ljava/io/File; (0x24) ---> Ljava/io/File;->delete()Z

```
                => Lorg/appcelerator/kroll/util/TiTempFileHelper;->doCleanTempDir()V (0x94) --->
Ljava/io/File;->delete()Z
                => Lorg/appcelerator/titanium/util/TiFileHelper;->wipeDirectoryTree(Ljava/io/File;
Ljava/util/SortedSet;)V (0x7a) --->
                        Ljava/io/File;->delete()Z
                => Lorg/appcelerator/titanium/util/TiFileHelper;->destroyTempFiles()V (0x24) --->
Ljava/io/File;->delete()Z
                =>
Lorg/appcelerator/titanium/util/TiFileHelper;->getTempFileFromFile(Ljava/lang/String; Z)Ljava/io/File;
(0xe8) --->
                        Ljava/io/File;->delete()Z
                =>
Lorg/appcelerator/titanium/util/TiFileHelper;->wipeDirectoryTree(Ljava/io/File;)V (0x80) --->
Ljava/io/File;->delete()Z
                =>
Lti/modules/titanium/media/MediaModule;->launchNativeCamera(Lorg/appcelerator/kroll/KrollDict;)
V (0x29e) --->
                        Ljava/io/File;->delete()Z
[Notice]    Native Library Loading Checking:
            Native library loading codes(System.loadLibrary(...)) found:
                [libsqlite3x.so]
                => Lio/requery/android/database/sqlite/SQLiteDatabase;-><clinit>()V (0x1e) --->
                        Ljava/lang/System;->loadLibrary(Ljava/lang/String;)V
                [libc++_shared.so]
                => Lorg/appcelerator/kroll/runtime/v8/V8Runtime;->initRuntime()V (0x20) --->
                        Ljava/lang/System;->loadLibrary(Ljava/lang/String;)V
                [libkroll-v8.so]
                => Lorg/appcelerator/kroll/runtime/v8/V8Runtime;->initRuntime()V (0x2a) --->
                        Ljava/lang/System;->loadLibrary(Ljava/lang/String;)V
[Notice]    AndroidManifest Exported Components Checking 2:
            Found "exported" components(except for Launcher) for receiving Google's "Android"
actions (AndroidManifest.xml):
                    activity => com.pingapp.hopandroid2.SendIntentReceiver
                    activity => com.pingapp.hopandroid2.SendIntentReceiverToMyself
                    activity => com.pingapp.widget2.HopWidgetSettings
                    receiver => com.pingapp.gcmjs2.LoopbackBroadcastReceiver
                    receiver => com.pingapp.widget2.HopWidgetProvider
                    receiver => com.appsflyer.SingleInstallBroadcastReceiver
                    receiver => com.pingapp.calendar.CalendarChangeReceiver
[Info] <Command> Runtime Command Checking:
            This app is not using critical function 'Runtime.getRuntime().exec("...")'.
[Info] <Command> Executing "root" or System Privilege Checking:
            Did not find codes checking "root" permission(su) or getting system permission (It's still
possible we did not find out).
[Info] <Database> SQLiteDatabase Transaction Deprecated Checking:
            Ignore checking "SQLiteDatabase:beginTransactionNonExclusive" because your set
minSdk >= 11.
[Info] <Database> Android SQLite Databases Encryption (SQLite Encryption Extension (SEE)):
            This app is "NOT" using SQLite Encryption Extension (SEE) on Android
(http://www.sqlite.org/android) to encrypt or decrpyt
            databases.
[Info] <Database> Android SQLite Databases Encryption (SQLCipher):
            This app is "NOT" using SQLCipher(http://sqlcipher.net/) to encrypt or decrpyt
databases.
[Info] <Database><#CVE-2011-3901#> Android SQLite Databases Vulnerability Checking:
            This app is "NOT" using Android SQLite databases.
[Info] <Debug> Android Debug Mode Checking:
```

DEBUG mode is OFF(android:debuggable="false") in AndroidManifest.xml.
[Info] Dynamic Code Loading:
No dynamic code loading(DexClassLoader) found.
[Info] <#BID 64208, CVE-2013-6271#> Fragment Vulnerability Checking:
Did not detect the vulnerability of "Fragment" dynamically loading into "PreferenceActivity" or "SherlockPreferenceActivity"
[Info] <Framework> Framework - MonoDroid:
This app is NOT using MonoDroid Framework (http://xamarin.com/android).
[Info] <Hacker> Base64 String Encryption:
No encoded Base64 String or Urls found.
[Info] <Database><Hacker> Key for Android SQLite Databases Encryption:
Did not find using the symmetric key(PRAGMA key) to encrypt the SQLite databases (It's still possible that it might use but we
did not find out).
[Info] <Debug><Hacker> Codes for Checking Android Debug Mode:
Did not detect codes for checking "ApplicationInfo.FLAG_DEBUGGABLE" in AndroidManifest.xml.
[Info] <Hacker> APK Installing Source Checking:
Did not detect this app checks for APK installer sources.
[Info] <KeyStore><Hacker> KeyStore File Location:
Did not find any possible BKS keystores or certificate keystore file (Notice: It does not mean this app does not use keysotre):
[Info] <KeyStore><Hacker> KeyStore Protection Checking:
Ignore checking KeyStore protected by password or not because you're not using KeyStore.
[Info] <Hacker> Code Setting Preventing Screenshot Capturing:
Did not detect this app has code setting preventing screenshot capturing.
[Info] <Signature><Hacker> Getting Signature Code Checking:
Did not detect this app is checking the signature in the code.
[Info] HttpURLConnection Android Bug Checking:
Ignore checking "http.keepAlive" because you're not using "HttpURLConnection" and min_Sdk > 8.
[Info] <KeyStore> KeyStore Type Checking:
KeyStore 'BKS' type check OK
[Info] Google Cloud Messaging Suggestion:
Nothing to suggest.
[Info] <#CVE-2013-4787#> Master Key Type I Vulnerability:
No Master Key Type I Vulnerability in this APK.
[Info] App Sandbox Permission Checking:
No security issues "MODE_WORLD_READABLE" or "MODE_WORLD_WRITEABLE" found on 'openOrCreateDatabase' or 'openOrCreateDatabase2' or
'getDir' or 'getSharedPreferences' or 'openFileOutput'
[Info] AndroidManifest Dangerous ProtectionLevel of Permission Checking:
No "dangerous" protection level customized permission found (AndroidManifest.xml).
[Info] AndroidManifest PermissionGroup Checking:
PermissionGroup in permission tag of AndroidManifest sets correctly.
[Info] AndroidManifest "intent-filter" Settings Checking:
"intent-filter" of AndroidManifest.xml check OK.
[Info] AndroidManifest Normal ProtectionLevel of Permission Checking:
No default or "normal" protection level customized permission found (AndroidManifest.xml).
[Info] <#CVE-2013-6272#> AndroidManifest Exported Lost Prefix Checking:
No exported components that forgot to add "android:" prefix.
[Info] AndroidManifest ContentProvider Exported Checking:
No exported "ContentProvider" found (AndroidManifest.xml).
[Info] <Sensitive_Information> Getting IMEI and Device ID:

Did not detect this app is getting the "device id(IMEI)" by "TelephonyManager.getDeviceId()" approach.
[Info] Codes for Sending SMS:
Did not detect this app has code for sending SMS messages (sendDataMessage, sendMultipartTextMessage or sendTextMessage).
[Info] <System> AndroidManifest sharedUserId Checking:
This app does not use "android.uid.system" sharedUserId.
[Info] <SSL_Security> SSL Implementation Checking (Verifying Host Name in Custom Classes):
Self-defined HOSTNAME VERIFIER checking OK.
[Info] <SSL_Security> SSL Implementation Checking (Verifying Host Name in Fields):
Critical vulnerability "ALLOW_ALL_HOSTNAME_VERIFIER" field setting or "AllowAllHostnameVerifier" class instance not found.
[Info] <SSL_Security> SSL Implementation Checking (Insecure component):
Did not detect SSLSocketFactory by insecure method "getInsecure".
[Info] <SSL_Security> SSL Implementation Checking (HttpHost):
DEFAULT_SCHEME_NAME for HttpHost check: OK
[Info] <SSL_Security> SSL Connection Checking:
Did not discover urls that are not under SSL (Notice: if you encrypt the url string, we can not discover that).
[Info] <SSL_Security> SSL Implementation Checking (WebViewClient for WebView):
Did not detect critical usage of "WebViewClient"(MITM Vulnerability).
[Info] <SSL_Security> SSL Certificate Verification Checking:
Did not find vulnerable X509Certificate code.
[Info] Unnecessary Permission Checking:
Permission 'android.permission.ACCESS_MOCK_LOCATION' sets correctly.
[Info] Accessing the Internet Checking:
This app is using the Internet via HTTP protocol.
[Info] AndroidManifest System Use Permission Checking:
No system-level critical use-permission found.
[Info] <WebView> WebView Local File Access Attacks Checking:
Did not find potentially critical local file access settings.
[Info] <WebView> WebView Potential XSS Attacks Checking:
Did not detect "setJavaScriptEnabled(true)" in WebView.
[Info] <WebView><Remote Code Execution><#CVE-2013-4710#> WebView RCE Vulnerability Checking:
WebView addJavascriptInterface vulnerabilities not found.
