Done By: SAPTARSHI CHATTERJEE(MCA student)
Date: 06 th April/2020
TARGET: Aarogyasetu Android App
Email: sapchatterjee1998@gmail.com
Contact:+919748858085

At first I want to tell I have done this for noble purpose I didn't have any bad motive so I will be writing certain Critical and some not that Critical bugs so I will be Highlighting the CRITICAL ones with RED colour.

Platform: Android
Package Name: nic.goi.aarogyasetu
Package Version Name: 1.0.1
Package Version Code: 1037
Min Sdk: 23
Target Sdk: 29
MD5 : bd664dcee0dd77a4e22ea98d7b35ce66
SHA1 : 7f56cdfed5ead53a9d9f774218944d1a55112048
SHA256: b0d7d011071dc2e6238e90530252f930b8f24a1c4c4115dc861063f520d1525e
SHA512:
2b514ce162acce746ade21f6451e65058500a4052394b14c3a7abd1aa170fe174289c1e1fef558
9b0132fc302f8d2697536f57d38b76cca390251ba247552227
Analyze Signature:
48fe9be5a1cb47d29886e2ef311bc7b31ea76ad464473a87a7740ae4a11dbc568fb8b01625832e
535300da2d47caf9fa7c717d189d13feffe77b5d945d486dd6

[Critical] <Command> Runtime Command Checking:
This app is using critical function 'Runtime.getRuntime().exec("...")'.
Please confirm these following code secions are not harmful:
=>
Lnic/goi/aarogyasetu/views/SplashActivity;->onCreate(Landroid/os/Bundle;)V (0x124) --->
Ljava/lang/Runtime;->exec(Ljava/lang/String;)Ljava/lang/Process;
=>
Lnic/goi/aarogyasetu/views/SplashActivity;->onCreate(Landroid/os/Bundle;)V (0x29e) --->
Ljava/lang/Runtime;->exec(Ljava/lang/String;)Ljava/lang/Process;

[Critical] <WebView><Remote Code Execution><#CVE-2013-4710#> WebView RCE Vulnerability Checking:
Found a critical WebView "addJavascriptInterface" vulnerability. This method can be used to allow JavaScript to control the host
application.
This is a powerful feature, but also presents a security risk for applications targeted to API level JELLY_BEAN(4.2) or below,
because JavaScript could use reflection to access an injected object's public fields. Use of this method in a WebView containing
untrusted content could allow an attacker to manipulate the host application in unintended ways, executing Java code with the
permissions of the host application.
Reference:

1."http://developer.android.com/reference/android/webkit/WebView.html#addJavascriptInterface(java.lang.Object,
java.lang.String) "

2.https://labs.mwrinfosecurity.com/blog/2013/09/24/webview-addjavascriptinterface-remote-code-execution/

3.http://50.56.33.56/blog/?p=314

4.http://blog.trustlook.com/2013/09/04/alert-android-webview-addjavascriptinterface-code-execution-vulnerability/

Please modify the below code:

=>
Lnic/goi/aarogyasetu/views/WebViewActivity;->onCreate(Landroid/os/Bundle;)V (0x102)
--->

Landroid/webkit/WebView;->addJavascriptInterface(Ljava/lang/Object; Ljava/lang/String;)V

[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.

receiver => nic.goi.aarogyasetu.utility.SmsReceiver

[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

=> Lk/a/a/a/o/b/i;->h(Landroid/content/Context;)Z (0xc) --->

Landroid/provider/Settings$Secure;->getString(Landroid/content/ContentResolver; Ljava/lang/String;)Ljava/lang/String;

=> Li/c/a/b/i/a/r9;->b(Li/c/a/b/i/a/m; Li/c/a/b/i/a/ca;)V (0xfd2) --->

Landroid/provider/Settings$Secure;->getString(Landroid/content/ContentResolver; Ljava/lang/String;)Ljava/lang/String;

[Warning] <SSL_Security> SSL Certificate Verification Checking:

Please make sure this app has the conditions to check the validation of SSL Certificate. If it's not properly checked, it MAY

allows self-signed, expired or mismatch CN certificates for SSL connection.

This is a critical vulnerability and allows attackers to do MITM attacks without your knowledge.

If you are transmitting users' username or password, these sensitive information may be leaking.

Reference:

(1)OWASP Mobile Top 10 doc:

https://www.owasp.org/index.php/Mobile_Top_10_2014-M3
(2)Android Security book: http://goo.gl/BFb65r

(3)https://www.securecoding.cert.org/confluence/pages/viewpage.action?pageId=134807561
This vulnerability is much more severe than Apple's "goto fail" vulnerability: http://goo.gl/eFlovw
Please do not try to create a "X509Certificate" and override "checkClientTrusted", "checkServerTrusted", and "getAcceptedIssuers"
functions with blank implementation.
We strongly suggest you use the existing API instead of creating your own X509Certificate class.
Please modify or remove these vulnerable code:

[Maybe Vulnerable (Please manually confirm)]
=> Lk/a/a/a/o/e/d;
-> used by:

Li/c/d/o/e;->a(Lcom/crashlytics/android/core/CrashlyticsPinningInfoProvider;)Ljavax/net/ssl/SSLSocketFactory;

[Warning] <WebView> WebView Local File Access Attacks Checking:
Found "setAllowFileAccess(true)" or not set(enabled by default) in WebView. The attackers could inject malicious script into
WebView and exploit the opportunity to access local resources. This can be mitigated or prevented by disabling local file system
access. (It is enabled by default)
Note that this enables or disables file system access only. Assets and resources are still accessible using file:///android_asset
and file:///android_res.
The attackers can use "mWebView.loadUrl("file:///data/data/[Your_Package_Name]/[File]");" to access app's local file.
Reference:
(1)https://labs.mwrinfosecurity.com/blog/2012/04/23/adventures-with-android-webviews/

(2)http://developer.android.com/reference/android/webkit/WebSettings.html#setAllowFileAccess(boolean)
Please add or modify "yourWebView.getSettings().setAllowFileAccess(false)" to your WebView:

Lnic/goi/aarogyasetu/views/WebViewActivity;->onCreate(Landroid/os/Bundle;)V

[Warning] <WebView> WebView Potential XSS Attacks Checking:
Found "setJavaScriptEnabled(true)" in WebView, which could exposed to potential XSS attacks. Please check the web page code
carefully and sanitize the output:
=>
Lnic/goi/aarogyasetu/views/WebViewActivity;->onCreate(Landroid/os/Bundle;)V (0x8a)
--->
Landroid/webkit/WebSettings;->setJavaScriptEnabled(Z)V

[Notice] <Command> Executing "root" or System Privilege Checking:
The app may has the code checking for "root" permission, mounting filesystem operations or monitoring system:

Lnic/goi/aarogyasetu/views/SplashActivity;->onCreate(Landroid/os/Bundle;)V => 'su'
Ld/a/a/l/a0;-><clinit>()V => '/system/bin/failsafe/'
Ld/a/a/l/a0;-><clinit>()V => '/system/bin/'
Ld/a/a/l/a0;-><clinit>()V => '/system/bin/.ext/'
Ld/a/a/l/a0;-><clinit>()V => '/system/bin'

[Notice] <Database><#CVE-2011-3901#> Android SQLite Databases Vulnerability Checking:
This app is using Android SQLite databases but it's "NOT" suffering from SQLite Journal Information Disclosure Vulnerability.
[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
=> Lcom/crashlytics/android/core/CrashlyticsController;->deleteSessionPartFilesFor(Ljava/lang/String;)V (0x14) --->
Ljava/io/File;->delete()Z
=> Lcom/crashlytics/android/core/CrashlyticsController;->recursiveDelete(Ljava/io/File;)V (0x2c) --->
Ljava/io/File;->delete()Z
=> Lcom/crashlytics/android/core/CrashlyticsController;->retainSessions([Ljava/io/File; Ljava/util/Set;)V (0x5a) --->
Ljava/io/File;->delete()Z
=> Lcom/crashlytics/android/core/CrashlyticsController;->retainSessions([Ljava/io/File; Ljava/util/Set;)V (0x9e) --->
Ljava/io/File;->delete()Z
=> Lcom/crashlytics/android/core/CrashlyticsController;->doCleanInvalidTempFiles([Ljava/io/File;)V (0x14a) --->
Ljava/io/File;->delete()Z
=> Lcom/crashlytics/android/core/CrashlyticsFileMarker;->remove()Z (0x8) ---> Ljava/io/File;->delete()Z
=> Lcom/crashlytics/android/core/LogFileManager;->discardOldLogFiles(Ljava/util/Set;)V (0x38) ---> Ljava/io/File;->delete()Z
=> Lcom/crashlytics/android/core/Utils;->capFileCount(Ljava/io/File; Ljava/io/FilenameFilter; I Ljava/util/Comparator;)I
(0x28) ---> Ljava/io/File;->delete()Z
=> Lh/h/g/i;->a(Landroid/content/Context; Landroid/content/res/Resources; I Ljava/lang/String; I)Landroid/graphics/Typeface;
(0x1c) ---> Ljava/io/File;->delete()Z
=> Lh/h/g/i;->a(Landroid/content/Context; Landroid/content/res/Resources; I Ljava/lang/String; I)Landroid/graphics/Typeface;
(0x34) ---> Ljava/io/File;->delete()Z
=> Lh/h/g/i;->a(Landroid/content/Context; Landroid/content/res/Resources; I Ljava/lang/String; I)Landroid/graphics/Typeface;
(0x3e) ---> Ljava/io/File;->delete()Z
=> Lh/h/g/i;->a(Landroid/content/Context; Landroid/content/res/Resources; I Ljava/lang/String; I)Landroid/graphics/Typeface;
(0x46) ---> Ljava/io/File;->delete()Z

=> Lh/h/g/i;->a(Landroid/content/Context;
Ljava/io/InputStream;)Landroid/graphics/Typeface; (0x1c) --->
Ljava/io/File;->delete()Z
=> Lh/h/g/i;->a(Landroid/content/Context;
Ljava/io/InputStream;)Landroid/graphics/Typeface; (0x34) --->
Ljava/io/File;->delete()Z
=> Lh/h/g/i;->a(Landroid/content/Context;
Ljava/io/InputStream;)Landroid/graphics/Typeface; (0x3e) --->
Ljava/io/File;->delete()Z
=> Lh/h/g/i;->a(Landroid/content/Context;
Ljava/io/InputStream;)Landroid/graphics/Typeface; (0x46) --->
Ljava/io/File;->delete()Z
=> Li/b/a/l/a;->a(Ljava/io/File; I I J)Li/b/a/l/a; (0x44) --->
Ljava/io/File;->delete()Z
=> Li/b/a/l/a;->a(Ljava/io/File;)V (0xc) ---> Ljava/io/File;->delete()Z
=> Li/b/a/l/a;->d(Ljava/lang/String;)Z (0x44) --->
Ljava/io/File;->delete()Z
=> Li/b/a/l/a;->m()V (0x190) ---> Ljava/io/File;->delete()Z
=> Li/b/a/l/c;->a(Ljava/io/File;)V (0x2a) ---> Ljava/io/File;->delete()Z
=>
Li/c/a/b/i/a/e;->getWritableDatabase()Landroid/database/sqlite/SQLiteDatabase; (0xaa) --->
Ljava/io/File;->delete()Z
=>
Li/c/a/b/i/a/l4;->getWritableDatabase()Landroid/database/sqlite/SQLiteDatabase; (0x3c) --->
Ljava/io/File;->delete()Z
=> Lcom/crashlytics/android/core/InvalidSessionReport;->remove()V
(0x54) ---> Ljava/io/File;->delete()Z
=> Lcom/crashlytics/android/core/NativeSessionReport;->remove()V
(0x58) ---> Ljava/io/File;->delete()Z
=> Lcom/crashlytics/android/core/NativeSessionReport;->remove()V
(0xa0) ---> Ljava/io/File;->delete()Z
=> Lcom/crashlytics/android/core/QueueFileLogStore;->deleteLogFile()V
(0xa) ---> Ljava/io/File;->delete()Z
=> Lcom/crashlytics/android/core/SessionReport;->remove()V (0x4c) --->
Ljava/io/File;->delete()Z
=> Lh/h/g/d;->a(Landroid/content/Context; Lh/h/f/b/c;
Landroid/content/res/Resources; I)Landroid/graphics/Typeface; (0x48)
---> Ljava/io/File;->delete()Z
=> Lh/h/g/d;->a(Landroid/content/Context; Lh/h/f/b/c;
Landroid/content/res/Resources; I)Landroid/graphics/Typeface; (0x6c)
---> Ljava/io/File;->delete()Z
=> Lh/h/g/d;->a(Landroid/content/Context; Lh/h/f/b/c;
Landroid/content/res/Resources; I)Landroid/graphics/Typeface; (0x74)
---> Ljava/io/File;->delete()Z
=> Lh/h/g/d;->a(Landroid/content/Context; Lh/h/f/b/c;
Landroid/content/res/Resources; I)Landroid/graphics/Typeface; (0x82)
---> Ljava/io/File;->delete()Z
=> Lh/h/g/d;->a(Landroid/content/Context; Lh/h/f/b/c;
Landroid/content/res/Resources; I)Landroid/graphics/Typeface; (0x8a)
---> Ljava/io/File;->delete()Z
=> Lh/h/g/e;->a(Landroid/content/Context; Lh/h/f/b/c;
Landroid/content/res/Resources; I)Landroid/graphics/Typeface; (0x54)
---> Ljava/io/File;->delete()Z
=> Lh/h/g/e;->a(Landroid/content/Context; Lh/h/f/b/c;
Landroid/content/res/Resources; I)Landroid/graphics/Typeface; (0xb0)

---> Ljava/io/File;->delete()Z
=> Lh/h/g/e;->a(Landroid/content/Context; Lh/h/f/b/c;
Landroid/content/res/Resources; I)Landroid/graphics/Typeface; (0xde)
---> Ljava/io/File;->delete()Z
=> Lk/a/a/a/o/d/g;->a(Ljava/io/File; Ljava/io/File;)V (0x42) --->
Ljava/io/File;->delete()Z
=> Lk/a/a/a/o/d/g;->a(Ljava/io/File; Ljava/io/File;)V (0x6a) --->
Ljava/io/File;->delete()Z
=> Lk/a/a/a/o/d/g;->a(Ljava/util/List;)V (0x4a) --->
Ljava/io/File;->delete()Z
=> Lk/a/a/a/o/d/g;->b()V (0xe) ---> Ljava/io/File;->delete()Z
[Notice] <Debug><Hacker> Codes for Checking Android Debug Mode:
Found codes for checking "ApplicationInfo.FLAG_DEBUGGABLE" in
AndroidManifest.xml:
=> Lk/a/a/a/o/b/i;->f (Landroid/content/Context;)Z
=> Li/c/a/b/d/h;->a (I)Z
[Notice] <Hacker> APK Installing Source Checking:
This app has code checking APK installer sources(e.g. from Google Play, from
Amazon, etc.). It might be used to check for whether
the app is hacked by the attackers.
=> Li/c/a/b/i/a/d6;->run()V (0x1d6) --->

Landroid/content/pm/PackageManager;->getInstallerPackageName(Ljava/lang/String;)Ljava/
lang/String;
=> Lk/a/a/a/o/b/t$a;->load(Landroid/content/Context;)Ljava/lang/Object;
(0x10) --->

Landroid/content/pm/PackageManager;->getInstallerPackageName(Ljava/lang/String;)Ljava/
lang/String;
=> Li/c/a/b/i/a/j4;->u()V (0x56) --->

Landroid/content/pm/PackageManager;->getInstallerPackageName(Ljava/lang/String;)Ljava/
lang/String;
[Notice] <KeyStore><Hacker> KeyStore Protection Information:
The Keystores below are "protected" by password and seem using SSL-pinning
(Total: 1). You can use "Portecle" tool to manage the
certificates in the KeyStore:
=> Lk/a/a/a/o/e/e;-><init>(Ljava/io/InputStream; Ljava/lang/String;)V
(0x24) --->
Ljava/security/KeyStore;->load(Ljava/io/InputStream; [C)V
[Notice] <Signature><Hacker> Getting Signature Code Checking:
This app has code checking the package signature in the code. It might be used
to check for whether the app is hacked by the
attackers.
=> Lh/h/j/b;->a(Landroid/content/Context; Landroid/os/CancellationSignal;
Lh/h/j/a;)Lh/h/j/b$e; (0x42) --->

Landroid/content/pm/PackageManager;->getPackageInfo(Ljava/lang/String;
I)Landroid/content/pm/PackageInfo;
=> Li/c/a/b/d/g;->honorsDebugCertificates(Landroid/content/Context;)Z
(0x28) --->

Landroid/content/pm/PackageManager;->getPackageInfo(Ljava/lang/String;
I)Landroid/content/pm/PackageInfo;
=> Li/c/a/b/d/g;->zza(Landroid/content/Context; Z I)I (0x50) --->

```
Landroid/content/pm/PackageManager;->getPackageInfo(Ljava/lang/String;
I)Landroid/content/pm/PackageInfo;
                    => Li/c/a/b/d/h;->a(I)Z (0x5a) --->
Landroid/content/pm/PackageManager;->getPackageInfo(Ljava/lang/String;
                        I)Landroid/content/pm/PackageInfo;
                    => Li/c/a/b/d/l/q;->a(Landroid/content/Context; I)Z (0x20) --->

Landroid/content/pm/PackageManager;->getPackageInfo(Ljava/lang/String;
I)Landroid/content/pm/PackageInfo;
                    => Li/c/d/l/p/c;->a(Ljava/net/URL;)Ljava/net/HttpURLConnection; (0xd6)
--->

Landroid/content/pm/PackageManager;->getPackageInfo(Ljava/lang/String;
I)Landroid/content/pm/PackageInfo;
                    =>
Lnic/goi/aarogyasetu/utility/AppSignatureHelper;->getAppSignatures()Ljava/util/ArrayList;
(0x1e) --->

Landroid/content/pm/PackageManager;->getPackageInfo(Ljava/lang/String;
I)Landroid/content/pm/PackageInfo;
                    => Li/c/a/b/i/a/aa;->b(Landroid/content/Context; Ljava/lang/String;)Z
(0x26) --->

Landroid/content/pm/PackageManager;->getPackageInfo(Ljava/lang/String;
I)Landroid/content/pm/PackageInfo;
[Notice]    AndroidManifest Exported Components Checking 2:
            Found "exported" components(except for Launcher) for receiving Google's
"Android" actions (AndroidManifest.xml):
                activity => nic.goi.aarogyasetu.views.WebViewActivity
                receiver => nic.goi.aarogyasetu.background.BootReceiver
[Info]    AndroidManifest Adb Backup Checking:
            This app has disabled Adb Backup.
[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] <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]    External Storage Accessing:
            External storage access not 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] <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] <Hacker> Code Setting Preventing Screenshot Capturing:
Did not detect this app has code setting preventing screenshot capturing.
[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] Native Library Loading Checking:
No native library loaded.
[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] <Implicit_Intent> Implicit Service Checking:
No dangerous implicit service.
[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):