
radware

Radware Mobile SDK Integration – Android v5.1.3

Objective:

This is an integration document for the Radware Bot Manager Android SDK. You can use it to integrate with your Android App and protect your API servers from the malicious bots originating from Mobile App.

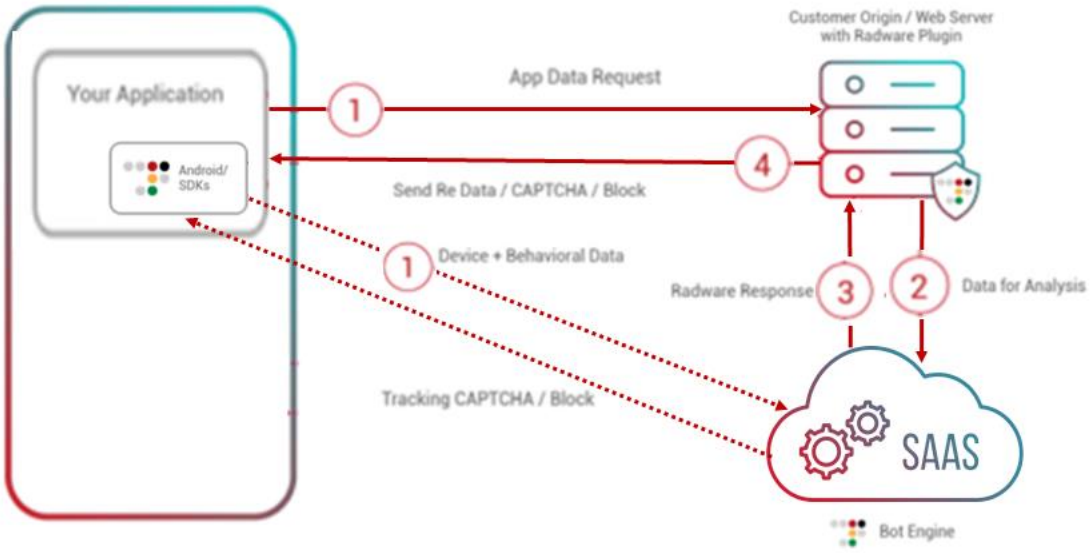
Prerequisite & Compatibility:

1. Ensure you have created an account for the Radware Bot Manager
2. Ensure you have already integrated the Radware Bot Manager plugin on your Web or Application server
3. Operating System Supported: Android 4.4 or above
4. Device supported: Smartphones &Tablets
5. Android studio version supported: 2.3 or above
6. Applications should be using retrofit for Rest calls

Reach out to Radware Support Team for any clarifications.

How it works?

The following image shows the data flow from Radware Bot Manager Mobile SDK which is integrated on customer's App to Radware Bot Manager Server and Origin/Web Server with Radware Bot Manager Plugin.



1. When your mobile app makes a REST call to your server/API, the data request is sent to the Radware Bot Manager connector integrated at your Origin/Web server. And Bot Manager headers are added from the sdk for the application API requests. In parallel, Radware Bot Manager SDK asynchronously collects data and sends it to the Radware Bot Manager bot engine for analysis. Data consists of device details (characteristics, orientation & acceleration) and behavioural details (events: buttons clicked, ads clicked, articles read/shared/liked, comments posted, screens viewed, purchases made, levels completed etc.,) based on your business logic from the interactions of the user with your Mobile App.
2. Radware Bot Manager connector sends the data to the Radware bot engine via Radware Bot Manager endpoint (deployed across the world using the global load balancer for minimal latency). Radware bot engine analyses the data from the endpoint and responds with appropriate action to humans and bots.
3. Your origin/webserver either sends the requested App data to the user or challenge with CAPTCHA or block page based on the response from Radware Bot Manager. Radware Bot Manager SDK has the capability to render in-app CAPTCHA / Block page.

Note:

- You must be in **'Active mode'** for receiving response codes for CAPTCHA/BLOCK by configuring the responses of different categories of bad bots in **'Bot response page'** in Radware Bot Manager portal.
- If required, the Radware bot engine can also be configured to send response code (CAPTCHA/BLOCK) directly to Radware Bot Manager SDK.

- Now Radware Bot Manager is integrated with Google attestation services. To enable this integration at client (sdk) refer **Steps to Integrate with Google Attestation service** section.

  Though Attestation feature is add-on on top of SDK, we highly recommend to turn on App Attestation feature for better protection. Please get in touch with Radware support team to enable App Attestation service on Bot Manager backend.

Integration steps

The integration involves 2 steps as listed below.

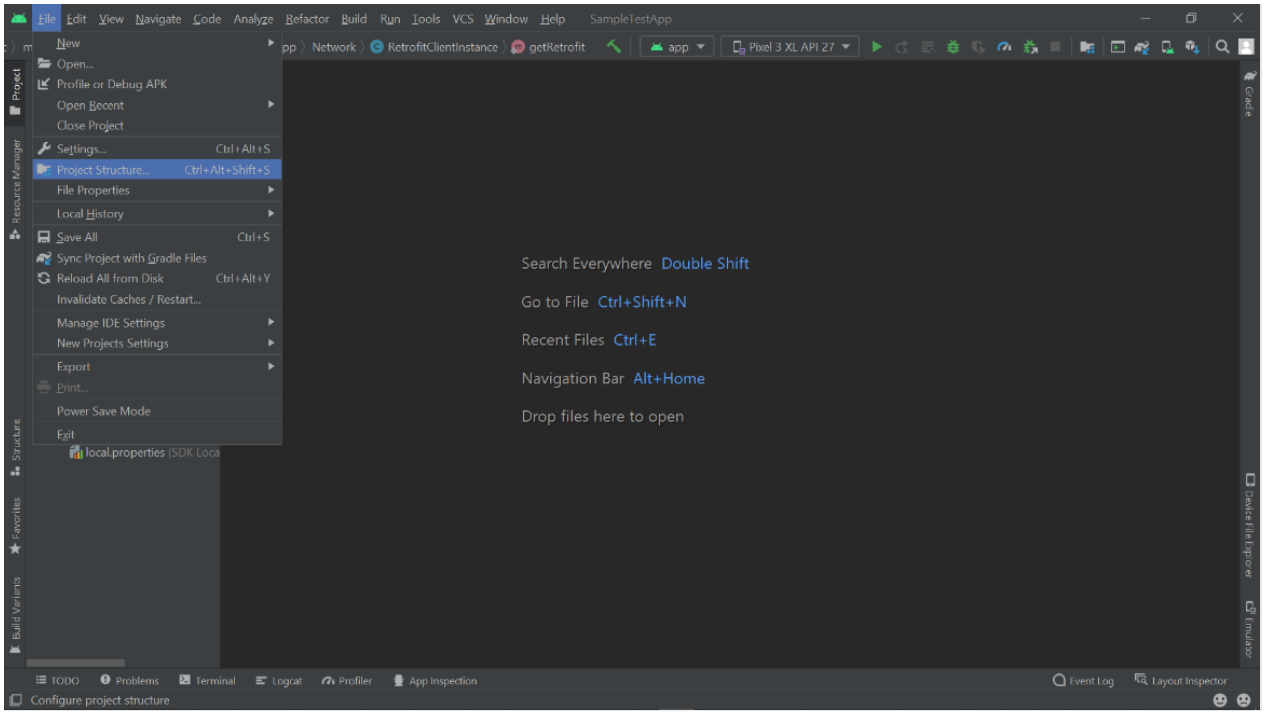
1. Integrate Radware Bot Manager SDK into your project
2. Configure the Radware Bot Manager SDK

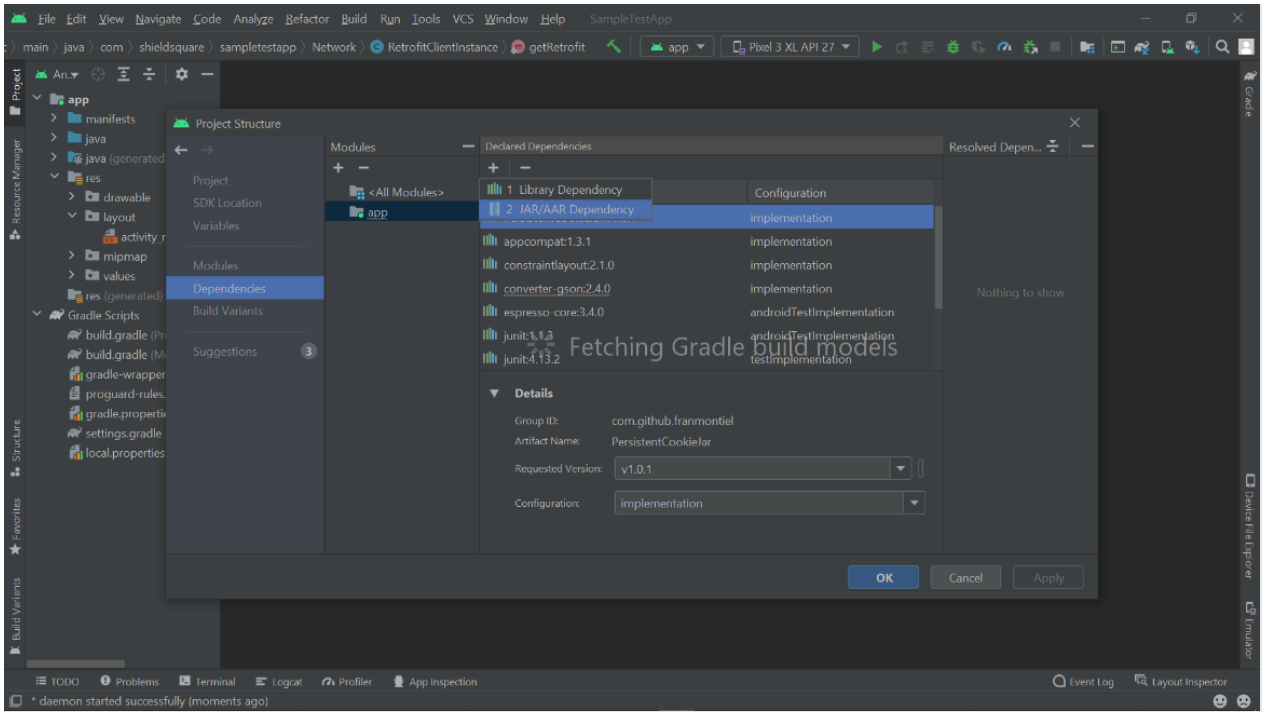
Integrate Radware Bot Manager SDK into your project

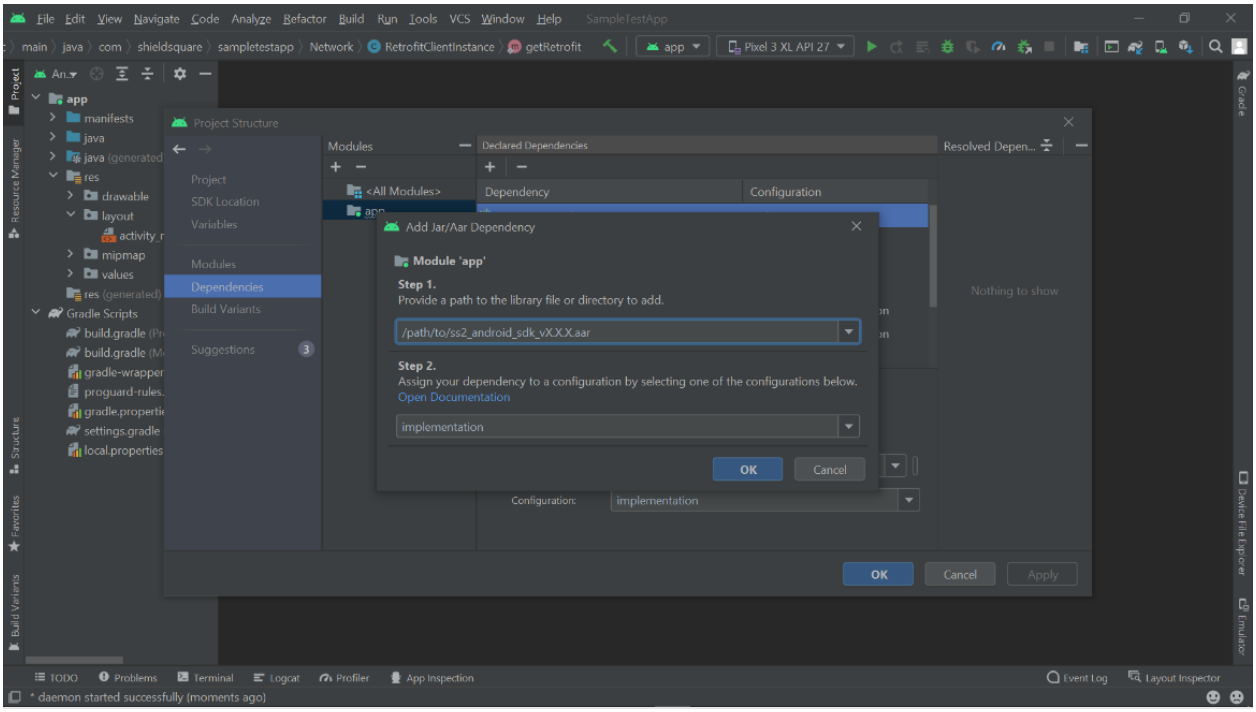
i. Unzip the package **ss2_android_sdk_vX.X.X.zip** which would contain **ss2_android_sdk_vX.X.X_debug.aar** and **ss2_android_sdk_vX.X.X_release.aar** files. Recommended to copy the SDK to **libs** directory in your project's **app** module.

Note that now SDK package has both debug and release version of SDK. Radware recommend to use debug version first to ensure the successful SDK integration and to use release version of SDK for production rollout.

ii. Import the **ss2_android_sdk_vX.X.X.aar** file as a dependency to your project. You can do that by going to *File -> Project Structure -> Dependencies (Left Panel) -> Add Dependencies -> JAR/AAR Dependencies*.
For better clarity, you can refer screenshots below or you can visit this link by **developer.android.com**. It is the official web resource of Google to help with Android Integrations.


File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help SampleTestApp
New
Open...
Profile or Debug APK
Open Recent
Close Project
Settings... Ctrl+Alt+S
Project Structure... Ctrl+Alt+Shift+S
File Properties
Local History
Save All Ctrl+S
Sync Project with Gradle Files
Reload All from Disk Ctrl+Alt+Y
Invalidate Caches / Restart...
Manage IDE Settings
New Projects Settings
Export
Print...
Power Save Mode
Exit
local.properties (SDK Loca
Search Everywhere Double Shift
Go to File Ctrl+Shift+N
Recent Files Ctrl+E
Navigation Bar Alt+Home
Drop files here to open
TODO Problems Terminal Logcat Profiler App Inspection Event Log Layout Inspector
Configure project structure


Project Structure
Project
SDK Location
Variables
Modules
Dependencies
Build Variants
Suggestions
Modules
<All Modules>
app
Declared Dependencies
1 Library Dependency
2 JAR/AAR Dependency
Configuration
implementation
appcompat:1.3.1 implementation
constraintlayout:2.1.0 implementation
converter-gson:2.4.0 implementation
espresso-core:3.4.0 androidTestImplementation
junit:1.1.3 androidTestImplementation
junit:4.13.2 testImplementation
Resolved Depen...
Nothing to show
Details
Group ID: com.github.franmontiel
Artifact Name: PersistentCookieJar
Requested Version: v1.0.1
Configuration: implementation
OK Cancel Apply
Fetching Gradle build models
TODO Problems Terminal Logcat Profiler App Inspection Event Log Layout Inspector
* daemon started successfully (moments ago)



iii. Add the following dependencies to your app-level **build.gradle** file. This dependency is to add Cookie storing capability to your App (if it doesn't support cookie storing currently) which will help Radware Bot Manager to set four first-party cookies to identify patterns of the user interactions.

```
implementation 'com.squareup.retrofit2:retrofit:2.9.0'
implementation 'com.squareup.okhttp3:logging-interceptor:4.9.0'
implementation 'com.squareup.retrofit2:converter-gson:2.9.0'
implementation 'com.google.code.gson:gson:2.8.8'
implementation 'com.github.franmontiel:PersistentCookieJar:v1.0.1'
 // below dependency is only for reCaptcha
implementation 'com.google.android.gms:play-services-safetynet:18.0.1'
// integrity manager (new) [if attestation is disabled, no need to add this
    dependency
implementation 'com.google.android.play:integrity:1.0.2'
// DSAEncoding [must required to generate keyPair]
implementation 'org.bouncycastle:bcpkix-jdk15on:1.70'
```

iv. Change in gradle.properties if using jetifier and gradle version below(7.0.0)

Add the line **`android.jetifier.blacklist=bcprov-jdk15on`** to your build gradle.properties file.

Note:

If you encounter this error:

Failed to resolve: com.github.franmontiel:PersistentCookieJar:v1.0.1

- Open your build.gradle(Project:) file.
- Add the following line in the 'all project' section.

  maven { url "https://jitpack.io" }

Configure Radware Bot Manager SDK

i. Initialize the Radware Bot Manager SDK in your application file as below.

Use code block 1 if you want to initialize Simple CAPTCHA:

```
//Code Block 1: Text CAPTCHA
private final String CID = "xxxx";
private final String SECRET = "yyyyyyyyy";

//Define below parameters to enable Google attestation
public static final boolean ENABLE_ATTESTATION = true;
public static final boolean ATTESTATION_ON_RELAUNCH = false;
public static final long ATTESTATION_EXPIRY = 86400;


SimpleCaptcha simpleCaptcha = new SimpleCaptcha.Builder()
                    .build();

ShieldSquare shieldSquare = new ShieldSquare.Builder(this)
                  .setTrackingEnabled(true)
                  .setSubscriberID(CID)
                  .setShieldSecret(SECRET)
                  .setCaptchaOption(simpleCaptcha)
                  .setShieldSquareServiceUrl("https://rwscus.shieldsquare.net/")

                 //setting below configuration parameter is optional. Set this
                   configuration from your application only if you want to
                   change the default interval values set in the SDK. The default
                   Interval values set in the SDK are 60 and 300 as shown below.
                  .setTrackingInterval(60,300)


                   // setting below configuration parameters are required only if
                      You are enabling integration with Google attestation service.
                   // Refer 'Steps to Integrate with Google Attestation service'
                      section for more details
                  .setAttestationConfig(ENABLE_ATTESTATION,ATTESTATION_ON_RELAUNCH,
                   ATTESTATION_EXPIRY).build()

//Code Block 1 Ends Here.
```

--------- OR ---------

Use code block 2 if you want to initialize Text CAPTCHA:

```
//Code Block 2: Text CAPTCHA
private final String CID = "xxxx";
private final String SECRET = "yyyyyyyyy";

// Redefine below parameters to enable Google attestation
public static final boolean ENABLE_ATTESTATION = true;
public static final boolean ATTESTATION_ON_RELAUNCH = false;
public static final long ATTESTATION_EXPIRY = 86400;


TextCaptcha textCaptcha = new TextCaptcha.Builder()
                    .build();

ShieldSquare shieldSquare = new ShieldSquare.Builder(this)
                  .setTrackingEnabled(true)
                  .setSubscriberID(CID)
```

```
                .setShieldSecret(SECRET)
                .setCaptchaOption(textCaptcha)
                .setShieldSquareServiceUrl("https://rwscus.shieldsquare.net/")

                //Setting below configuration parameter is optional. Set this
                configuration from your application only if you want to change
                the default interval values set in the SDK. The default interval
                values set in the SDK are 60 and 300 as shown below.
                .setTrackingInterval(60,300)


               //Setting below configurataion parameters are required only if
                 You are enabling integration with Google attestation service.
               //Refer 'Steps to Integrate with Google Attestation service'
                 section for more details

               .setAttestationConfig(ENABLE_ATTESTATION,ATTESTATION_ON_RELAUNCH,
                ATTESTATION_EXPIRY).build()

//Code Block 2 Ends Here.
```

---------- OR ----------

Use code block 3 if you want to initialize Google reCAPTCHA:

```
//Code Block 3: Google reCAPTCHA
private final String RECAPTCHA_SITE_KEY = "Your-reCAPTCHA-Key";
private final String RECAPTCHA_VERIFICATION_ENDPOINT =
"https://www.google.com/recaptcha/api/siteverify";

private final String CID = "xxxx";
private final String SECRET = "yyyyyyyyy";

// Redefine below parameters to enable Google attestation
public static final boolean ENABLE_ATTESTATION = true;
public static final boolean ATTESTATION_ON_RELAUNCH = false;
public static final long ATTESTATION_EXPIRY = 86400;


ReCaptcha reCaptcha = new ReCaptcha.Builder()
.setSiteKey(RECAPTCHA_SITE_KEY)
.setServerVerificationUrl(RECAPTCHA_VERIFICATION_ENDPOINT)
.build();

ShieldSquare shieldSquare = new ShieldSquare.Builder(this)
                .setTrackingEnabled(true)
                .setSubscriberID(CID)
                .setShieldSecret(SECRET)
                .setCaptchaOption(reCaptcha)
                .setShieldSquareServiceUrl("https://rwscus.shieldsquare.net/")

               //setting below configuration parameter is optional. Set this
                configuration from your application only if you want to change
                the default interval values set in the SDK. The default interval
                values set in the SDK are 60 and 300 as shown below.
                .setTrackingInterval(60,300)


               //setting below configuration parameters are required only if you
                  are enabling integration with Google attestation service.
               // Refer 'Steps to Integrate with Google Attestation service'
                  section for more details
               .setAttestationConfig(ENABLE_ATTESTATION,ATTESTATION_ON_RELAUNCH,
                ATTESTATION_EXPIRY).build()
```

```
//Code Block 3 End Here.
```

---------- AND ----------

Code Block 4: Block Option

```
private final String CID = "xxxx";
private final String SECRET = "yyyyyyyyy";

// Redefine below parameters to enable Google attestation
public static final boolean ENABLE_ATTESTATION = true;
public static final boolean ATTESTATION_ON_RELAUNCH = false;
public static final long ATTESTATION_EXPIRY = 86400;


BlockPage blockPage = new BlockPage.Builder().build();

ShieldSquare shieldSquare = new ShieldSquare.Builder(this)
                .setTrackingEnabled(true)
                .setSubscriberID(CID)
                .setShieldSecret(SECRET)
                .setBlockPage(blockPage)

                 //setting below configuration parameter is optional. Set this
                 configuration from your application only if you want to change
                 the default interval values set in the SDK. The default
                 Interval values set in the SDK are 60 and 300 as shown below.
                .setTrackingInterval(60,300)


                 //setting below configuration parameters are required only if
                   you are enabling integration with Google attestation
                   service.
                 //Refer 'Steps to Integrate with Google Attestation service'
                  section for more details
                .setAttestationConfig(ENABLE_ATTESTATION,
                 ATTESTATION_ON_RELAUNCH, ATTESTATION_EXPIRY).build()

//Code Block 4 End Here.
```

NOTE:

a. Radware Bot Manager support team will provide a unique customer ID for your account.
b. Ensure you can configure a required response 'CAPTCHA' or 'Block' from the Bot Response page in the Radware Bot Manager portal for different types of bots.
c. You can pass either 'textCaptcha' for TEXT CAPTCHA or 'simpleCaptcha' for SIMPLE CAPTCHA or 'reCaptcha' for Google ReCAPTCHA service while initializing Radware Bot Manager.
d. If you're setting up Google reCAPTCHA as captcha type in your application, select 'reCAPTCHA Android' as reCAPTCHA type in Google settings.
e. SDK supports customization of CAPTCHA and BLOCK pages as per your requirements. Refer **Android Mobile SDK Captcha Customization** document in **ss2_android_sdk_vX.X.X.zip** file. The document covers the complete set of customization options provided across multiple elements in the CAPTCHA and BLOCK page.

Add Interceptor

When initializing Retrofit, include **ShieldSquareInterceptor, ShieldSquareCookiePicker** and **ShieldSquareCookieManager** as below. Use below code block shown to add the interceptor to your Retrofit Client. If you are already using Radware Bot Manager SDK version less than 5.1.0. then please follow instructions mentioned under Customer's Using SDK Version Less Than 5.1.0 section and then implement the below interception code block.

```
//Retrofit Code Block
OkHttpClient client = new OkHttpClient.Builder()
                      .addInterceptor(new ShieldSquareInterceptor())
                      .cookieJar(CookieManager.provideCookieJar())
                      .build();

Retrofit retrofit = new Retrofit.Builder()
        .baseUrl(BASE_URL)
        .client(client)
        .build();

//Retrofit Code Block Ends Here.
```

> ***Note:*** If you are blocking Other Headers at your Server/CDN or at Radware Bot Manager Portal, please whitelist Shieldsquare headers in your firewall. These headers are essentials for identifying anomalies better.

The above code will intercept all responses received by the retrofit engine, enables cookie setting on the app, and acts based on the response from the Server.

Customer's Using SDK Version Less Than 5.1.0

If you have integrated with Radware Bot Manager version less than v5.1.0, make sure you remove the below interception code block from your existing integration while integrating with v5.1.0 and then follow the steps mentioned in Add Interceptor section. From SDK v5.1.0, adding additional headers is done within the SDK.

```
.addInterceptor(chain -> {
Request original = chain.request();
String BASE_URL = "xxxx";
String timestamp = String.valueOf(System.currentTimeMillis());
Request.Builder builder = original.newBuilder()
.addHeader("__uzmaj",CookieManager.getCookieForHeader("__uzmaj"))
.addHeader("__uzmbj",CookieManager.getCookieForHeader("__uzmbj"))
.addHeader("__uzmcj",CookieManager.getCookieForHeader("__uzmcj"))
.addHeader("__uzmdj",CookieManager.getCookieForHeader("__uzmdj"))
.addHeader("zpsh1",
ShieldSquare.getInstance().getCombinedHeader(timestamp))
.addHeader("zpsh2",ShieldSquare.getInstance().getAdditionalHeader(
timestamp))
.addHeader("zpset61",ShieldSquare.getInstance().getDeviceInfo()));
Request request = builder.build();
return chain.proceed(request);  })
```

ii. To collect events data, insert below analytics snippet whenever there is an event triggered in your app (Eg: App open, User signed in, Pages/screens viewed, an article read/liked/shared/commented, app version updated, etc.,). Radware Bot Manager collects the events data in a batch process at fixed time intervals.

```
//Event Tracker Code Block Begins Here
```

```
String eventName = "login_success";
String activityName = getTitle().toString();
JSONObject eventParams = new JSONObject();
try {
    eventParams.put("email", "user@example.com");
    eventParams.put("sessionId", "1234-1234-1234");
} catch (JSONException e) {
    e.printStackTrace();
}
ShieldSquare.trackEvents(eventName, activityName, eventParams);

//Event Tracker Code Block Ends Here.
```

Steps to Integrate with Google Attestation service:

To enable this feature, configurations need to be made on client side (your application) as well as on server side i.e., in Bot Manager Engine for the feature to be fully functional. Also refer Prerequisite's (Google Cloud Console and Google Play Console) section for the settings need to be made/enabled on Google Cloud Console and Play Console to enable attestation service.

Client-Side Configurations

1. Add the Google Play integrity dependency library to your application's build.gradle file
   implementation 'com.google.android.play:integrity:1.0.2'
2. Set below configuration parameters.
   //By default integration with Google Attestation service is not enabled in SDK, set below configuration parameter to true to enable this integration in the SDK
   public static final boolean *ENABLE_ATTESTATION* = true;

   //Once the integration with Google Attestation service is enabled, it is mandatory to define the below configuration parameters in your application either with the default values or with your custom set values.

   //Be default attestation service at first time application launch or called every 24 (86400 seconds) hours, you can reset this value as per your requirement.
   public static final long *ATTESTATION_EXPIRY* = 86400; *//seconds*

   //Be default calling attestation service at every app relaunch is tuned off. You can set below configuration parameter to true to turn attestation to happen at every application relaunch. We recommend not to turn on this flag until there is a real need.
   public static final boolean *ATTESTATION_ON_RELAUNCH* = false;

3. To set configuration parameters (mentioned in above step2) *ENABLE_ATTESTATION, ATTESTATION_EXPIRY and ATTESTATION_ON_RELAUNCH to ShieldSquare instance created from your application.*
   ShieldSquare shieldSquare = new ShieldSquare.Builder(this)//line 1
   .setAttestationConfig(***ENABLE_ATTESTATION***, A***TTESTATION_ON_RELAUNCH***, ***ATTESTATION_EXPIRY***).build() //line 2

Data to be configured on Server Side

Service account with roles Service Account User and Service Usage Consumer need to be configured on the BOT Manager backend and this data needs to be provided by the Mobile SDK customers. The service account credentials are required to decrypt and verify attestation token on the Google's Servers when the attestation token verification request is sent to Google's Servers from Radware backend (Refer https://cloud.google.com/iam/docs/creating-managing-service-accounts and

https://developer.android.com/google/play/integrity/verdict#decrypt-verify-google-servers Google's official documentation on how to create service account and the roles that service account should have).

Note: 1. Contact Radware Bot Manager support team to enable Google Play Integrity API integration in SDK.
2. Google Play Integrity API is supported from Android Version 4.4(API level 19) or higher (https://developer.android.com/google/play/integrity/overview)

Prerequisite's (Google Cloud Console and Google Play Console):

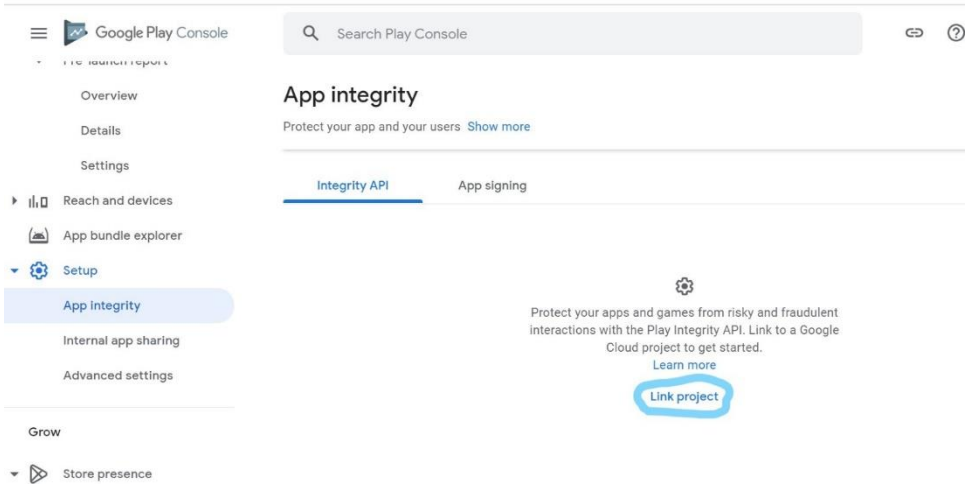
There is dependency on Google Cloud Console and Google Play Console for this feature to be functional.

Google Cloud Console:

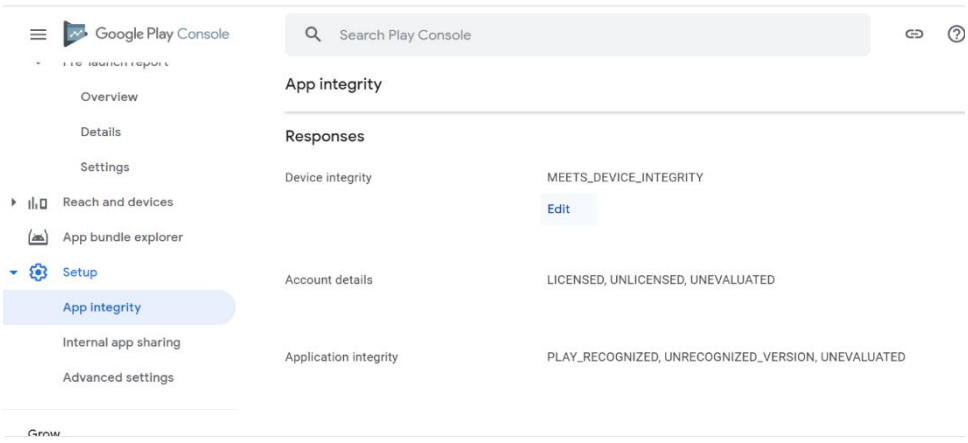
1. Enable **Google Play Integrity API** from google cloud console. https://console.developers.google.com/apis/api/playintegrity.googleapis.com/overview?project=**PROJECT_NAME**
2. Update the Usage Tier as per the application traffic.

Google Play Console:

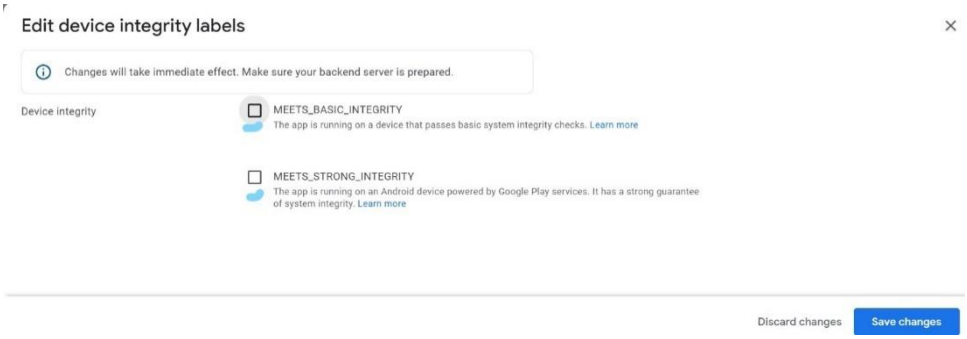
1. Customer app should be published in Google play store.
2. Enable App Integrity in the Play Console by linking project. Navigate to the Release section of the left menu. Go to Setup > App integrity. Select the Integrity API tab to get started. Click on Link Project, see below screenshot,



3. Once the project is linked click on edit, in responses section, see below screenshot:



4. A pop up will be opened as shown below:



Select MEETS_BASIC_INTEGRITY and MEETS_STRONG_INTEGRITY and click on Save Changes.

Now your application is ready to request and decode play integrity tokens.

Contact Us

Follow below mentioned steps to open a support case for any clarifications.

- Go to the link: https://support.radware.com/
- Click on the Support Cases
- Open a new case.
- Select Bot Manager in **Queue Routing**.