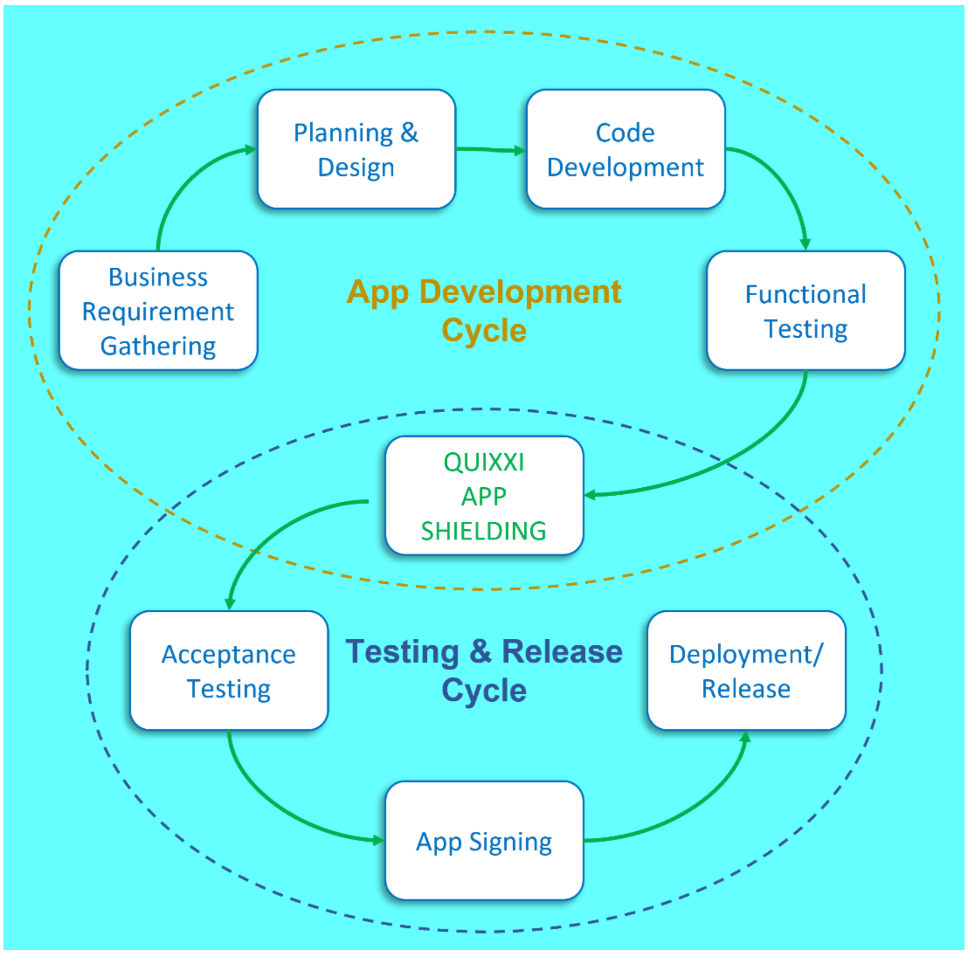
App Shielding- Read me!

1. Should you run a functional test on the app after shielding? Yes, please read below.

Quixxi Security is an application security solution that protects the app on a code level, so we recommend integrating the solution during the testing phase of your application development cycle. Before you upload your application distributable file for shielding, we recommend you do a functional testing and ensure all functionalities are working as expected, then applying Quixxi shield, test the application once again to ensure both results match and all functionalities are working as expected. Only after successful completion of this step consider signing the application for release in app store or play store. Following diagram will explain how to include Quixxi in your application development life cycle. We recommend to install the aab file in the device before publishing it to the device.



2. How to properly test the shield functions?

Following steps can be used to verify the effectiveness of the shielding on your application.

- Remove hardcoded strings

Unzip apk contents into a directory. Search for the file ends with .dex., convert them in to jar using jadx or dex2jar tool. Search for the hard coded string present in each class. It must be replaced with a function call.

- ## Use random classes and method names

Quixxi will insert multiple classes into the application binary (apk / aab) for increasing the application security. Name of these classes and method will vary for each shield. To verify this option, unzip the apk contents into a directory. Convert all .dex files into .jar using jadx or dex2jar tool. You can see the class names inserted by quixxi are very long and are different in each shield

- ## Insert spoof code

Quixxi will insert multiple classes and methods into the application binary (apk/aab) for increasing the application security. To perform each security operation, it will use a method with random name, but Quixxi will insert more methods and classes with similar instructions method call. This will increase the complexity of understanding the code flow after decompiling it. To verify this option unzip the apk contents into a directory, convert all .dex files into .jar files using jadx or dex2jar too. You can see that many methods in different classes are with same set of instructions.

- ## Remove app logs

Quixxi will remove method calls like system.out, Log.i, Log.d, Log.v from application binary to avoid the data leakage using the application debug logs. To verify this option, unzip the apk contents into a directory. Convert all .dex files into a jar, Open the jar file using jadx or dex2jar too. Open all classes in the application, you can see the statements like system.out, Log.i, Log.d, Log.v are removed from the application binary.

- ## Disable Copy & Paste Functionality

Disabling copy and paste on the application will increase the security of the application, it will help to prevent the data leakage using the clipboard. To verify this option, install the shielded application into the device, open it in the device, open a page with text box. Type something, long press on the text box to open the copy, paste context menu. It will not open on the native application, on some cross platform applications copy paste menu will displayed but the copied text cannot be pasted outside of the application.

- ## Disable screenshots capture & screen sharing

Disabling screenshots and screen sharing will help to prevent the data leakage by sharing the screens with sensitive data. To verify this option, install the shielded application into the device, open it in the device, try to capture the screenshot of the application by pressing the screenshot key combination. This key combination will vary from one model to another model.

- ## Terminate the app when running in rooted device

Prevent the application from running on the rooted device will help to prevent the data leakage by collecting all the data stored on the internal and external storage of the device. In rooted device, users can open all the folders in the device and easily collect all the files, database, images saved in the internal and external storage of the device. To verify this option, install and open the application in device, wait for few seconds. Application should display a screen with message “Rooted device not supported”, user can only exit this screen by tapping the exit button at the bottom of the screen.

- ## Terminate the app when connected to the emulator

Prevent the application from running on the rooted device will help to prevent the data leakage by collecting all the data stored on the internal and external storage of the device. In rooted device, users can open all the folders in the device and easily collect all the files, database, images saved in the internal and external storage of the device. To verify this option, install and open the application in emulator, wait for few seconds. Application should display a screen with message “Emulator not supported”, user can only exit this screen by tapping the exit button at the bottom of the screen.

- ## Integrate Malware Detector SDK

Scanning for the malwares install on the device will prevent the application from security attack. This option can be verified by installing the application in the device, Open the application on device after the installation, wait for few seconds, Quixxi will display a screen with list malwares detected on the device. User can exit or continue using the application based on the options selected during the application shield.

- ## Disable ADB Backup

Disabling the ADB Backup will prevent the data leakage by allowing anyone with physical access to the device to make a full backup of apps and their files/settings. This option can be verified by decompile the application using apktool and verifying the option android:allowBackup="false" in application tag in AndroidManifest.xml is set to false.

- ## Validate app integrity

Verifying the application integrity will help ensure the application is not modified after publishing it to store. To verify this option, decompile the application using any tool like apktool modify Manifest.xml, any classes.dex files present in the application, contents of the files in the assets folder, recompile the application using any tool. Install the recompiled application in the device and open it. Application should display a screen with a message “Application tampering detected”, An exit button is placed at the bottom of the screen to exit the application. User cannot use the application by escaping from this screen.

- ## Terminate the app when running with the debugger attached

Prevent attaching the application to a debugger will help to reduce the understanding the code flow of the application by an attacker. To verify this option, install the shielded application in a device and open it. Try to connect debugger using android studio. Application should display a screen with a message “Debugger not supported”. User cannot skip this screen. User can exit the application by tapping the Exit button placed at the bottom of the screen.

- ## Encrypt the "assets" folder

Encrypting the assets folder will reduce the code readability on the cross-platform apps. It will also help to protect code used to create the application for cross-platform apps. To verify this option, extract the shielded apk contents into a directory, open the files present in the assets folder of the application. Contents of the assets folder should not be readable. Install the application in device and open it to ensure the application working fine.

- ## Terminate the app when “USB debugging” is enabled

Detecting USB Debugging option in the device will help to prevent attaching the application to debugger. To verify this option, Install the shielded application into the device, open it, wait for few seconds. Application should display a screen with message “USB Debugging not supported”. User cannot skip this screen. User can exit the application by tapping the Exit button placed at the bottom of the screen.

- ## Terminate the app if installation from “Unknown Sources” is enabled

This option is useful for the applications with minimum sdk version below 8. This will be useful to ensure the application is downloaded from recognised stores. Enabling “Unknown Sources” option will allow the users to install the application downloaded using any application, disabling this option will ensure the application is downloaded from recognised stores. To verify this option, Install the shielded application into the device, open it, wait for few seconds. Application should display a screen with message “USB Debugging not supported”. User cannot skip this screen. User can exit the application by tapping the Exit button placed at the bottom of the screen.

- ## SSL certificate validation via SSL pinning

Validating the certificate used in network calls will help to prevent MiTM attack. To test this option install and open the shielded application in the device, configure a proxy in laptop, configure the mobile device to use proxy server for connection. Open the screen with API requests. Application should not accept the API response when connecting the device with proxy, it should work normal without proxy server.

3. How to Install aab file in device

1. Download bundle tool here

   https://github.com/google/bundletool/releases

2. Please use the following two commands to install the aab file into device, please connect the device you want to test with your laptop. USB debugging must be enabled on the device to install application.

   - Generate temp.apks file.

```
Java -jar bundletool.jar build-apks --output=/PATH/TO/SAVE/TEMPFILE/temp.apks --ks=/PATH/TO/YOUR/KEYSTORE/FILE --key-pass=pass:YOUR_PASSWORD --ks-key-alias=YOUR_COMMON_NAME --ks-pass=pass:YOUR_PASSWORD --bundle=/PATH/TO/AAB/FILE/apprelease.aab
```

   - Install apk file into device

```
java -jar bundletool.jar install-apks --apks=/PATH/TO/SAVE/TEMPFILE/temp.apks
```