Android MorningSDK v1.1.0
- Developer Manual -

2024. 3.

Morningsoft

Document History

No.	Date	Contents
1	2022.05	Android Manual Publishing
2	2023.05	Add Developer License API
3	2024.01	Modify License API
4	2024.03	Support sign/verify, HTTP 2.0

Contents

1. Introduction MorningSDK
2. Installation of Library
3. SDK Example
4. List up Android API

1. Introduction MorningSDK

a. MorningSDK

As mobile app hacking techniques continue to advance, cases of apps being hacked using the latest technology are occurring in the app market. Organizations and companies providing services to customers need measures to protect personal information from hacking tools that incorporate these latest technologies. The MorningSDK mobile library includes features to prevent the execution of apps requiring protection of sensitive information by verifying the installation of well-known hacking tools such as Frida, Magisk, and Magisk Hide worldwide. Additionally, it has features to prevent debugging or memory intrusion attempts using some hacking software in advance. Therefore, it includes tools not only to block already known hacking tools but also to block new hacking tools. Furthermore, it includes a feature to obfuscate sensitive information (user IDs, resident registration numbers) strings in the source code to make it difficult to extract such information from already distributed apps.

b. Feature

Include	Class
App integrity and tampering prevention	Integrity
String obfuscation	Obfuscation
Sensitive information storage	Secure Storage
Cryptography	Cryptography
Unique identifier issuance	AUUID
Malware detection	Vaccine
Screen capture prevention	DLP
mVoIP security	VOIP
document tampering prevention	Sign

- AUUID : Application Universally Unique IDentifier
- DLP : Data Loss Prevention

c. Compile and Environment

Item	Content
Android Studio	Android Studio Flamingo | 2023.2.1 and Above
Kotlin	v1.9.20 and Above
Device	Galaxy S6 and Above
OS	Android 6.0 and Above (api 23)

2. Installation of Library

After extracting the compressed file of MorningSDK and adding the MorningSDK.aar file located in Library/Android/kotlin to the 'libs' folder, initialize the MorningSDK functionality by calling the initialization function of MorningSDK.

1. Copy MorningSDK.aar

```
cp ./Library/Android/kotlin/MorningSDK.aar
./Sample/Android/kotlin/MorningSDKSample/app/libs/
```

2. Edit app gradle

```
implementation fileTree(include: [‘*.jar’, ‘*.aar’], dir: ‘libs’)
implementation files(‘libs/MorningSDK.aar’)

repositories {
    ...
    flatDir { dirs 'libs'}
}
```

3. Initialize license and setting in MainApplication

The License should be initialized in the global settings of the program. This allows MorningSDK to be accessible from all activities.

3.1 kotlin

```
getAppContext()?.let{
        MorningSDK().setLicense(mContext, “ Your-License-Data”,”voip.yourdomain.com\n
com.yourdomain.voip\nyourdomain.com\nYour-Developer-License”)
}
```

3.2 java

```
public static com.morningsoft.security.MorningSDK mMorningSDK;
...
      if(mContext != null){
         mMorningSDK.setLicense(mContext, "Your-License-Data","voip.morningsoft.com\
ncom.morningsoft.voip\nmorningsoft.com\nYour-Developer-License");
      }
```

4. Setting inheritance for Activity

Ensure the integrity check of the app and use MorningActivity for activities requiring security. Prevent unauthorized access to activities by bypassing the main activity through unauthorized routes.

4.1 kotlin

```
import com.morningsoft.game.activity.MorningActivity;
...
public class MainActivity extends MorningActivity {
```

4.2 java

```
import com.morningsoft.security.MorningSDKActivity
```

```
...
class MainActivity : MorningSDKActivity() {
```

5. AndroidManifest.xml Configuration

To use the sensitive information storage API, the following settings must be configured in the AndroidManifest.xml. This is for utilizing the Android EncryptedSharedPreferences feature.

```
<application
    ...
    android:allowBackup="false"
    android:fullBackupContent="false"
    …
```

3. SDK Example

a. Initialize Library

```
MorningSDK().setTamperProtectEnable(true)
MorningSDK().setMagiskHideProtectEnable(true)
MorningSDK().setDebuggerProtectEnable(true)
MorningSDK().setSignatureVerifyEnable(true)
MorningSDK().setFridaProtectEnable(true)
MorningSDK().setRootingCheckEnable(true)
MorningSDK().setScreenProtectEnable(true)
MorningSDK().setMarketVerifyEnable(true)
MorningSDK().setDebuggableProtectEnable(true)
```

b. Save sensitivity data using Android Keystore

```
 // To store data in the app, it must be encrypted without exception.
 // MorningSDK utilizes Android's Keystore for encryption, ensuring safety.
 // Note: Existing data will not be restored if the app is reinstalled or its data is deleted.
  var  mUserPassword : String = “pasword##!!”
  Log.d(TAG, "User Password : “ + mUserPassword)
  MorningSDK().writeString(“UserPassword”, mUserPassword)
  Log.d(TAG, "User Password : “ + MorningSDK().readString(“UserPassword”))
  var  mUserNumber : Int = “12341234”
  Log.d(TAG, "User Number : “ + mUserNumber)
  MorningSDK().writeInteger(“UserNumber”, mUserNumber)
  Log.d(TAG, "User Number : “ + MorningSDK().readInteger(“UserNumber”))
  var mUserLogin : Boolean = true
  MorningSDK().writeBool(“UserLogin”, mUserLogin)
  Log.d(TAG, “User Login : “ + MorningSDK().readBool(“UserLogin”))
```

c. String Obfuscation

```
// Legacy Code
var  mAESPassword : String = “pasword##!!”
Log.d(TAG, "AES Password : “ + mAESPassword)

// New Code
Log.d(TAG, “AES Password : “ + MorningSDK().getObfuscation(IDS_AES_PASSWORD))
```

4. List up Android API

a. Prevention of app integrity and tampering

Prototype	fun securityCheck()
Description	When inheriting MorningSDKActivity in Activity, it is internally called.
Parameter	None
Result	The app will terminate if integrity verification fails or malware is detected.
Remarks	

Prototype	fun isMarketVerifyEnabled() : Boolean
Description	Whether to use the market verification feature
Parameter	None
Result	true : enable false : disable
Remarks	

Prototype	fun isRootingCheckEnabled() : Boolean
Description	Whether to use the rooting check feature
Parameter	None
Result	true : enable false : disable
Remarks	

Prototype	fun isSignatureVerifyEnabled() : Boolean
Description	Whether to use the app signature verification feature
Parameter	None
Result	true : enable false : disable
Remarks	

Prototype	fun isDebuggableProtectEnabled() : Boolean
Description	Whether to use the feature to check for debugable settings
Parameter	None
Result	true : enable false : disable
Remarks	

Prototype	fun isFridaProtectEnabled() : Boolean
Description	Whether to use the Frida prevention feature

Parameter	None
Result	true : enable false : disable
Remarks	

Prototype	fun isDebuggerProtectEnabled() : Boolean
Description	Whether to use the debugger prevention feature
Parameter	None
Result	true : enable false : disable
Remarks	

Prototype	fun isMagiskHideProtectEnabled() : Boolean
Description	Whether to use the Magisk Hide feature
Parameter	None
Result	true : enable false : disable
Remarks	

Prototype	fun isTamperProtectEnabled() : Boolean
Description	Whether to use the tamper prevention feature
Parameter	None
Result	true : enable false : disable
Remarks	

Prototype	fun setMarketVerifyEnable(enable : Boolean?)
Description	Setting for market verification feature
Parameter	enable : true or false
Result	None
Remarks	

Prototype	fun setRootingCheckEnable(enable : Boolean?)
Description	Setting for rooting check feature
Parameter	enable : true or false
Result	None
Remarks	

Prototype	fun setSignatureVerifyEnable(enable : Boolean?)
Description	Setting for app signature verification feature

Parameter	enable : true or false
Result	None
Remarks	

Prototype	fun setDebuggableProtectEnable(enable : Boolean?)
Description	Setting for preventing debugging capability
Parameter	enable : true or false
Result	None
Remarks	

Prototype	fun setFridaProtectEnable(enable : Boolean?)
Description	Setting for Frida prevention feature
Parameter	enable : true or false
Result	None
Remarks	

Prototype	fun setDebuggerProtectEnable(enable : Boolean?)
Description	Setting for debugger execution prevention feature
Parameter	enable : true or false
Result	None
Remarks	

Prototype	fun setMagiskHideProtectEnable(enable : Boolean?)
Description	Setting for Magisk Hide prevention feature
Parameter	enable : true or false
Result	None
Remarks	

Prototype	fun setTamperProtectEnable(enable : Boolean?)
Description	Setting for tamper prevention feature
Parameter	enable : true or false
Result	None
Remarks	

b. String Obfuscation

Prototype	fun getObfuscation(symbol: String) : String?
Description	Decryption of obfuscated strings
Parameter	symbol : string key

Result	Decrypted plaintext
Remarks	

Prototype	fun setObfuscation(obfuscationData : String) -> Void
Description	Obfuscated data setting
Parameter	obfuscationData : Obfuscated string data
Result	None
Remarks	

Prototype	fun isStringObfuscationEnabled() : Boolean
Description	Whether to use string obfuscation feature
Parameter	None
Result	true : enable false : disable
Remarks	

Prototype	fun setStringObfuscationEnable(enable : Boolean?)
Description	Setting for string obfuscation feature
Parameter	enable : true or false
Result	None
Remarks	

c. Save Sensitivity Data

Prototype	fun writeString(key: String?, value: String?)
Description	Safely stores data in the format of (key, text) using a keystore
Parameter	key: Key value associated with the data value : String-formatted text
Result	None
Remarks	

Prototype	fun readString(key: String) : String?
Description	Retrieves text-format data corresponding to the key.
Parameter	key: Key value associated with the data
Result	String-formatted text
Remarks	

Prototype	fun writeInteger(key: String?, value: Int?)
Description	Safely stores data in the format of (key, integer) using a keystore.
Parameter	key: Key value associated with the data

	value : Integer data
Result	None
Remarks	

Prototype	fun readInteger(key: String) : Int
Description	Retrieves integer-formatted data corresponding to the key.
Parameter	key: Key value associated with the data
Result	Integer data
Remarks	

Prototype	fun writeBool(key: String?, value: Bool?)
Description	Safely stores data in the format of (key, binary) using a keystore.
Parameter	key: Key value associated with the data value : true or false
Result	None
Remarks	

Prototype	fun readBool(key: String) : Bool
Description	Retrieves binary-formatted data corresponding to the key.
Parameter	key: Key value associated with the data
Result	true or false
Remarks	

d. Cryptography

Prototype	fun initRSA2048String()
Description	Generating public/private keys (2048 bits) for RSA encryption of text (String).
Parameter	None
Result	None
Remarks	

Prototype	fun encryptRSA2048String(plainText: String) : String
Description	Performing RSA encryption of text (String) to generate cipher text.
Parameter	plainText : Plaintext text
Result	Base64-formatted ciphertext text
Remarks	The public key for encryption is retrieved from the latest key stored in the keystore. If the key has not been generated, it is automatically created before encryption.

Prototype	fun decryptRSA2048String(cipherText: String) : String
Description	Performing RSA decryption of text (String) to generate plaintext.

Parameter	cipherText : Base64-formatted cipher text text
Result	Plaintext text
Remarks	The public key for decryption is retrieved from the latest key stored in the keystore.

Prototype	fun initAES256String()
Description	Function to generate a key for encrypting plaintext text (String) with AES (256 bits).
Parameter	None
Result	None
Remarks	The previous key is invalidated and can no longer be used.

Prototype	fun encryptAES256String(plainText: String) : String
Description	Encrypting plaintext text (String) with AES (256 bits) to generate cipher text.
Parameter	plainText : Plaintext
Result	None
Remarks	The key for encryption is automatically generated and securely stored in the keystore.

Prototype	fun decryptAES256String(cipherText: String) : String
Description	Decrypting cipher text text (String) with AES (256 bits) to obtain plaintext
Parameter	cipherText : Base64-encoded cipher text
Result	Plaintext
Remarks	The key for decryption is automatically generated and securely stored in the keystore.

Prototype	func initECC25519KeyPair()
Description	Generating public/private keys for ECC 25519 algorithm
Parameter	None
Result	None
Remarks	

Prototype	func signECC25519WithBase64(message : String, publicKey : String) : String
Description	Calculating the signature value (signature) in Base64 format
Parameter	message : Original message for generating the signature value in Base64 format publicKey : Public key for signature
Result	Base64-formatted signature value
Remarks	

Prototype	func verifyECC25519WithBase64(message : String, signature : String, privateKey : String) : Bool
Description	Checking whether the message has been tampered with in Base64 format
Parameter	message : Original message for verifying the signature value in Base64 format signature : Base64-formatted signature value

	publicKey : Private key for signature
Result	true : The message has not been tampered with (signature successful) false : The message has been tampered with (signature failed)
Remarks	

Prototype	fun computeSHA256String(plainText: String) : String
Description	Generate a hash string using SHA256.
Parameter	plainText : Plaintext
Result	Text-formatted hash value
Remarks	

e. Unique identifier issuance

Prototype	fun getAUUID() : String
Description	Generating AUUID, which can be used as a unique identifier for mobile devices.
Parameter	None
Result	AUUID
Remarks	AUUID is regenerated upon app reinstallation or deletion of app data.

f. Malware detection

Prototype	fun runScan()
Description	Start malware scan
Parameter	None
Result	None
Remarks	Just rooting check

g. Screen capture prevention

Prototype	fun isScreenProtectEnabled() : Boolean
Description	Whether to use the screen capture prevention feature
Parameter	None
Result	true : enabled false : disabled
Remarks	

Prototype	fun setScreenProtectEnable(enable : Boolean?)
Description	Setting for the screen capture prevention feature.
Parameter	enable : true or false
Result	None

Remarks	

h. Self-Testing

Prototype	fun testMorningSDK()
Description	Self-testing of the library
Parameter	None
Result	None
Remarks	You can verify the test results via Logcat.

i. Set up License Information

Prototype	fun setLicense(mContext : ApplicationContext, licenseData: String, configData : String) -> Void
Description	License configuration
Parameter	mContext : The ApplicationContext acquired during program execution. licenseData : The issued license data configData : Configuration for usage
Result	None
Remarks	

j. Logging

Prototype	fun setLogLevel(logLevel : Int) -> Void
Description	Logging level configuration.
Parameter	logLevel : Setting the logging level
Result	None
Remarks	LogUtil.LEVEL_NONE LogUtil.LEVEL_ERROR LogUtil.LEVEL_WARN LogUtil.LEVEL_INFO LogUtil.LEVEL_DEBUG