iOS 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

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, FlyJB 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

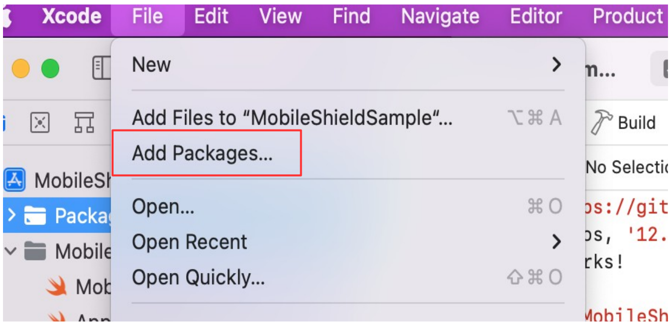
c. Compile and Environment

Item	Content
xCode	14.0.1 and Above
Swift	5.0 and Above
Device	iphone 6 and Above (arm64)
OS	iOS 12.4 and Above

2. Installation of Package

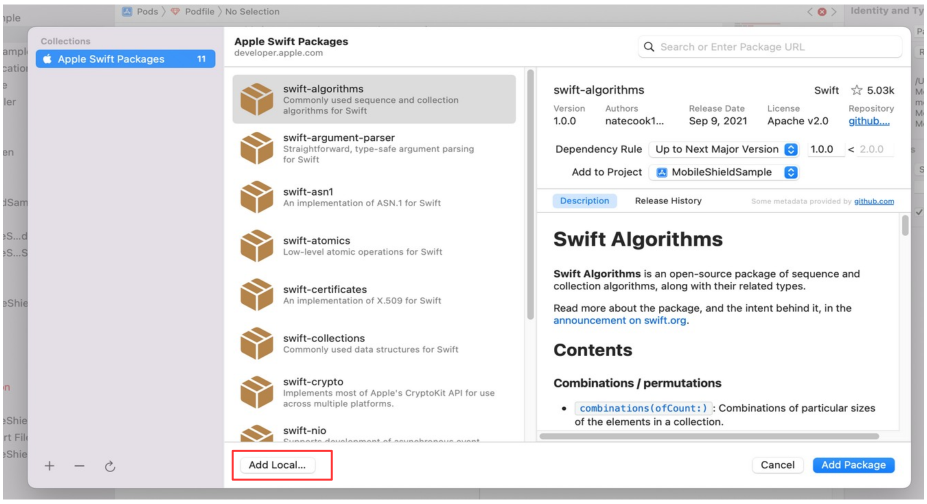
A. Adding Packages to the Project

Select "Add Packages" from the top menu in xCode.



B. Adding Local Packages

In the popped-up dialog box, select "Add Local".



Apple Swift Packages
developer.apple.com

swift-algorithms
Commonly used sequence and collection algorithms for Swift

swift-argument-parser
Straightforward, type-safe argument parsing for Swift

swift-asn1
An implementation of ASN.1 for Swift

swift-atomics
Low-level atomic operations for Swift

swift-certificates
An implementation of X.509 for Swift

swift-collections
Commonly used data structures for Swift

swift-crypto
Implements most of Apple's CryptoKit API for use across multiple platforms.

swift-nio

swift-algorithms Swift ☆ 5.03k

Version	Authors	Release Date	License	Repository
1.0.0	natecook1...	Sep 9, 2021	Apache v2.0	github....

Dependency Rule: Up to Next Major Version 1.0.0 < 2.0.0

Add to Project: MobileShieldSample

Description | Release History | Some metadata provided by github.com

Swift Algorithms

Swift Algorithms is an open-source package of sequence and collection algorithms, along with their related types.

Read more about the package, and the intent behind it, in the announcement on swift.org.

Contents

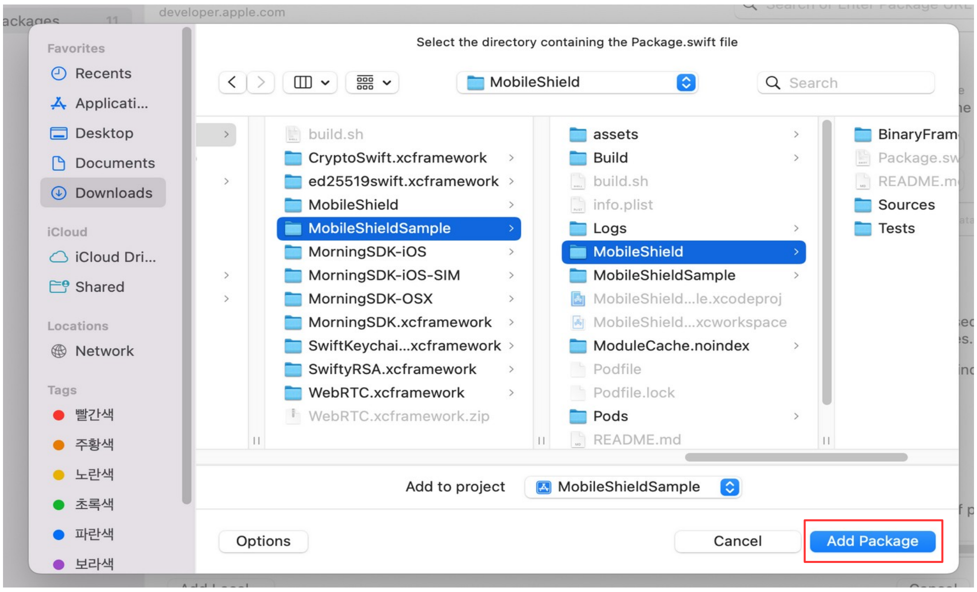
Combinations / permutations

- `combinations(ofCount:)`: Combinations of particular sizes of the elements in a collection.

C. Adding Packages to the Project

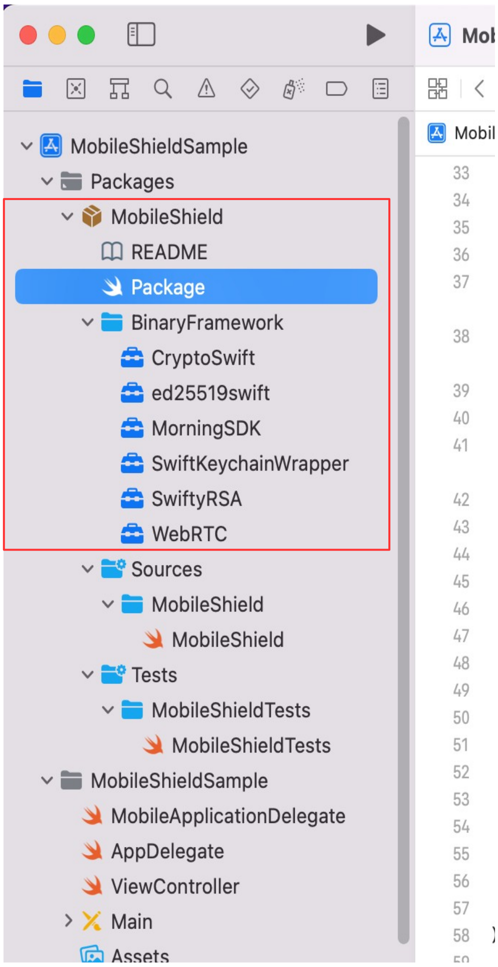
Select the "MorningSDK" folder containing the package and click the "Add Package" button.

Extract MorningSDK/Library/iOS/Swift/MorningSDK.zip



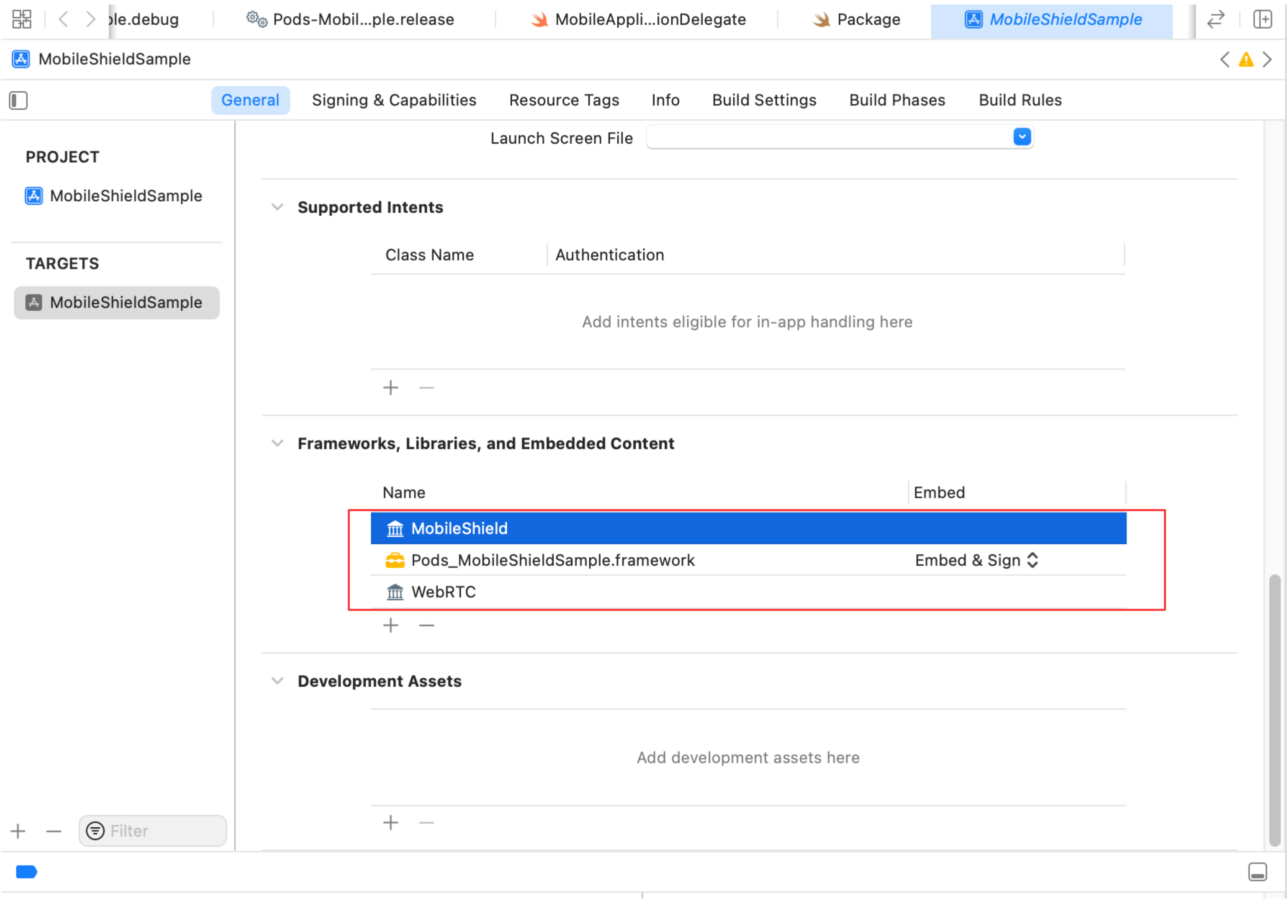
D. Confirm adding the workspace package

You can see the added MorningSDK package in the workspace window of xCode.



E. Adding Frameworks for Use in the Project

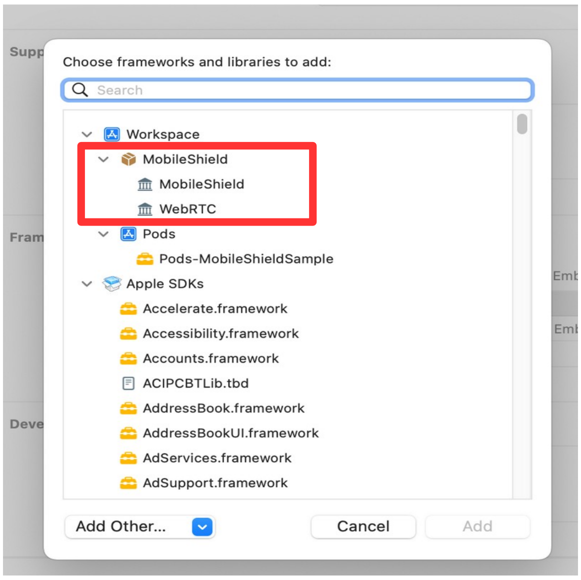
Navigate to General → Frameworks, Libraries, and Embedded Content, then add the framework.



Name	Embed
MobileShield	
Pods_MobileShieldSample.framework	Embed & Sign
WebRTC	

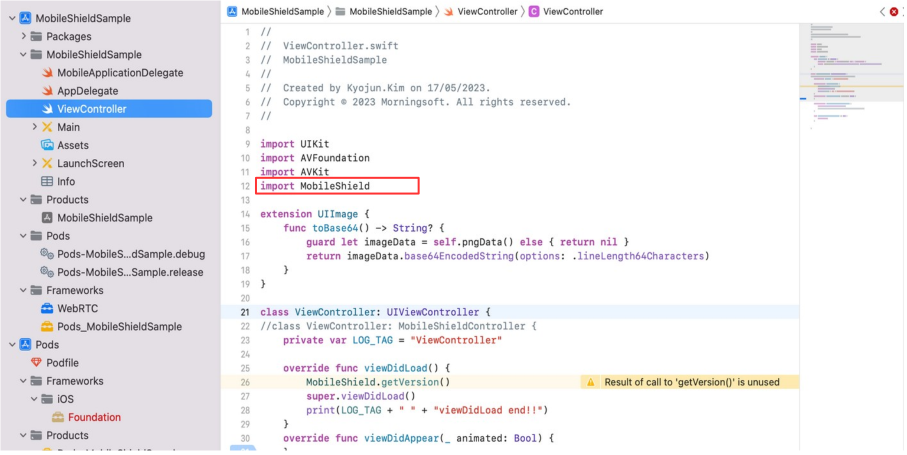
F. Select the framework from the list.

Add the MobileShield Framework from the package.



G. Use the framework in the source code.

The added package can be used in the project by using the statement "import MorningSDK".



```swift
//
//  ViewController.swift
//  MobileShieldSample
//
//  Created by Kyojun.Kim on 17/05/2023.
//  Copyright © 2023 Morningsoft. All rights reserved.
//

import UIKit
import AVFoundation
import AVKit
import MobileShield

extension UIImage {
    func toBase64() -> String? {
        guard let imageData = self.pngData() else { return nil }
        return imageData.base64EncodedString(options: .lineLength64Characters)
    }
}

class ViewController: UIViewController {
//class ViewController: MobileShieldController {
    private var LOG_TAG = "ViewController"

    override func viewDidLoad() {
        MobileShield.getVersion()
        super.viewDidLoad()
        print(LOG_TAG + " " + "viewDidLoad end!!")
    }
    override func viewDidAppear(_ animated: Bool) {
```

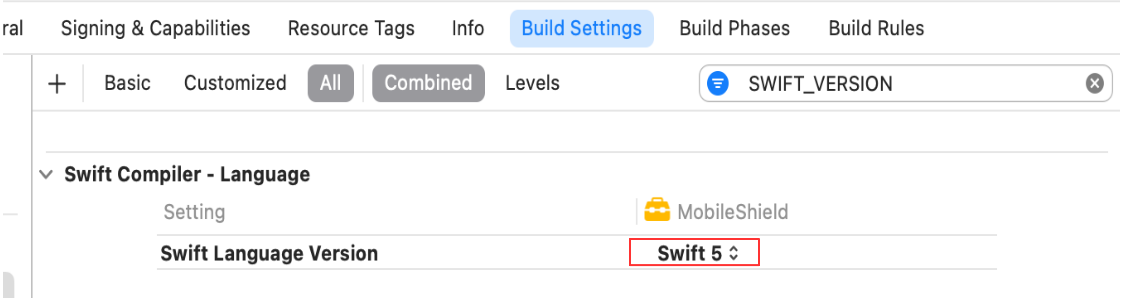
⚠ Result of call to 'getVersion()' is unused

H. Checking the required settings when using the library

In the source code, the following mandatory settings need to be configured to use it.

```
override func viewDidLoad() {
        MorningSDK.setLicense(licenseData : “Your-License-Data”, configData : “voip.yourdomain.com\ncom.yourdomain.voip\nyourdomain.com\nYour-Developer-License”)
...
```

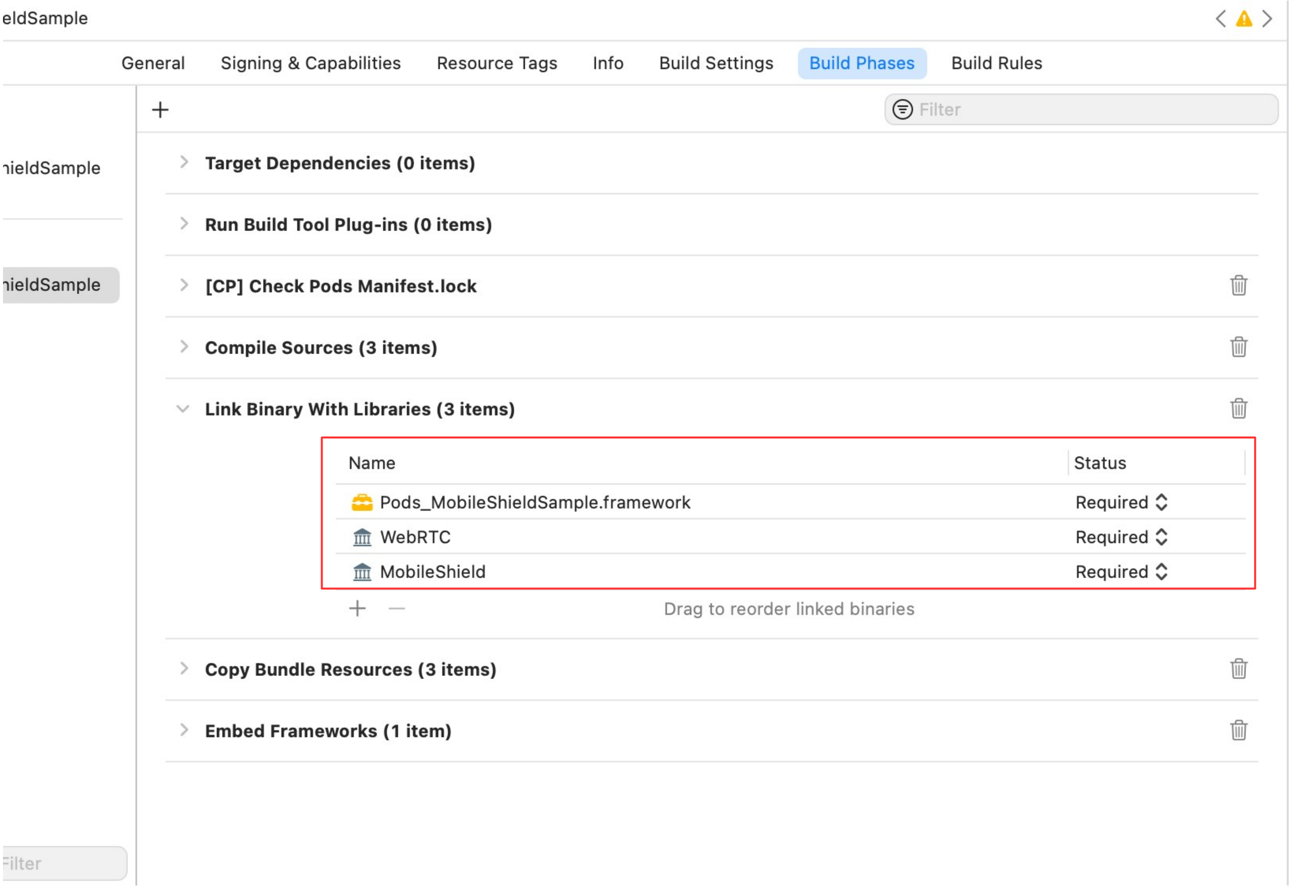
I. Checking the settings for project build
Select Swift 5 for the project's default settings.



Swift Compiler - Language

Setting	MobileShield
Swift Language Version	Swift 5

Checking project settings



Target Dependencies (0 items)

Run Build Tool Plug-ins (0 items)

[CP] Check Pods Manifest.lock

Compile Sources (3 items)

Link Binary With Libraries (3 items)

Name	Status
Pods_MobileShieldSample.framework	Required
WebRTC	Required
MobileShield	Required

Drag to reorder linked binaries

Copy Bundle Resources (3 items)

Embed Frameworks (1 item)

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 iOS Keychain

```
// To store data in the app, it must be encrypted without exception.
// MorningSDK utilizes iOS’s Keychain for encryption, ensuring safety.
// The Keychain's lock state matches the device's lock state, based on whether the device is locked or unlocked.
var  mUserPassword : String = “pasword##!!”
print("User Password : \(mUserPassword) ")
MorningSDK.writeString(key : “UserPassword”, value : mUserPassword)
print("User Password : \(MorningSDK.readString(key : “UserPassword”)) ")
var  mUserNumber : Int = “12341234”
print("User Number : \(mUserNumber) ")
MorningSDK.writeInteger(key : “UserNumber”, value : mUserNumber)
print("User Number : \(MorningSDK.readInteger(key : “UserNumber”)) ")
var mUserLogin : Bool = true
MorningSDK.writeBool(key : “UserLogin”, value : mUserLogin)
print(“User Login : \(MorningSDK.readBool(key : “UserLogin”)) “)
```

c. String Obfuscation

```
// Legacy Code
var  mAESPassword : String = “pasword##!!”
print("AES Password : \(mAESPassword) ")

// New Code
print(“AES Password : \(MorningSDK.getObfuscation(IDS_AES_PASSWORD)) “)
```

4. List up iOS API

a. Prevention of app integrity and tampering

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

Prototype	func isJailbrokenEnabled() -> Bool
Description	Whether to enable the Jailbreak prevention feature
Parameter	None
Result	true : enabled false : disabled
Remarks	

Prototype	func isRunInEmulatorEnabled() -> Bool
Description	Whether to enable the feature that prevents execution on emulators
Parameter	None
Result	true : enabled false : disabled
Remarks	

Prototype	func isDebuggedEnabled() -> Bool
Description	Whether to enable the feature that prevents debugging execution
Parameter	None
Result	true : enabled false : disabled
Remarks	

Prototype	func isTamperedEnabled() -> Bool
Description	Whether to enable the feature that prevents file tampering
Parameter	None
Result	true : enabled false : enabled
Remarks	

Prototype	func isReverseEngineeredEnabled() -> Bool
Description	Whether to enable the feature that prevents reverse engineering
Parameter	None

Result	true : enabled false : disabled
Remarks	

Prototype	func isRuntimeHookedEnabled() -> Bool
Description	Whether to enable the feature that prevents real-time hooking
Parameter	None
Result	true : enabled false : disabled
Remarks	

Prototype	func isProxiedEnabled() -> Bool
Description	Whether to enable the feature that prevents proxy
Parameter	None
Result	true : enabled false : disabled
Remarks	

Prototype	func isBreakpointAtEnabled() -> Bool
Description	Whether to enable the feature that prevents setting breakpoints
Parameter	None
Result	true : enabled false : disabled
Remarks	

Prototype	func setJailbrokenEnabled(enable : Bool?)
Description	Setting the jailbreak prevention feature
Parameter	enable : true or false
Result	None
Remarks	

Prototype	func setRunInEmulatorEnabled(enable : Bool?)
Description	Setting the feature to prevent execution on emulators
Parameter	enable : true or false
Result	None
Remarks	

Prototype	func setDebuggedEnabled(enable : Bool?)
Description	Setting the feature to prevent debugging execution
Parameter	enable : true or false
Result	None

Remarks	

Prototype	func setTamperedEnabled(enable : Bool?)
Description	Setting the feature to prevent file tampering
Parameter	enable : true or false
Result	None
Remarks	

Prototype	func setReverseEngineeredEnabled(enable : Bool?)
Description	Setting the feature to prevent reverse engineering
Parameter	enable : true or false
Result	None
Remarks	

Prototype	func setRuntimeHookedEnabled(enable : Bool?)
Description	Setting the feature to prevent real-time hooking
Parameter	enable : true or false
Result	None
Remarks	

Prototype	func setProxiedEnabled(enable : Bool?)
Description	Setting the feature to prevent proxy
Parameter	enable : true or false
Result	None
Remarks	

Prototype	func setBreakpointAtEnabled(enable : Bool?)
Description	Setting the feature to prevent setting breakpoints
Parameter	enable : true or false
Result	None
Remarks	

b. String Obfuscation

Prototype	func getObfuscation(symbol: String) -> String?
Description	Decrypted obfuscated strings
Parameter	symbol : Key value to distinguish strings
Result	Decrypted plaintext
Remarks	

Prototype	func setObfuscation(obfuscationData: String) -> void
Description	Obfuscated data configuration
Parameter	obfuscationData : Obfuscated data
Result	None
Remarks	

Prototype	func isStringObfuscationEnabled() -> Bool
Description	Whether to enable the string obfuscation feature
Parameter	None
Result	true : enabled false : disabled
Remarks	

Prototype	func setStringObfuscationEnabled(enable : Bool?)
Description	Setting the string obfuscation feature
Parameter	enable : true or false
Result	None
Remarks	

c. Sensitive Data storage

Prototype	func writeString(key: String?, value: String?)
Description	Safely stores data in the format of (key, text) using the keychain.
Parameter	key: Key value associated with the data value : Text data
Result	None
Remarks	

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

Prototype	func writeInteger(key: String?, value: Int?)
Description	Safely stores data in the format of (key, integer) using the keychain.
Parameter	key: Key value associated with the data value : Integer data
Result	None
Remarks	

Prototype	func 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	func writeBool(key: String?, value: Bool?)
Description	Safely stores data in the format of (key, binary) using the keychain.
Parameter	key: Key value associated with the data value : Binary data
Result	None
Remarks	

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

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

Prototype	func 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 keychain.

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

Prototype	func 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 text
Remarks	The key for decryption is automatically generated and securely stored in the keychain.

Prototype	func initECC25519KeyPair()
Description	Generating public/private keys for the 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	func computeSHA256String(plainText: String) -> String
Description	Creating a hash string using SHA256.
Parameter	plainText : Plaintext text
Result	String-formatted hash value
Remarks	

e. Issuing a unique identifier

Prototype	func getAUUID()-> String
Description	Generating Application UUID (unique identifier), which can be used as the identifier for mobile devices.
Parameter	None
Result	AUUID
Remarks	AUUID is regenerated upon app reinstallation or deletion of app data.

f. malware detection

Prototype	func runScan()
Description	Start malware scan
Parameter	None
Result	None
Remarks	Just scan jailbreaken.

g. Preventing screen capture

Prototype	func isScreenProtectEnabled() -> Bool
Description	Whether to enable the screen capture prevention feature
Parameter	None
Result	true : enabled false : disabled
Remarks	

Prototype	func setScreenProtectEnabled(enable : Bool?)
Description	Setting the screen capture prevention feature
Parameter	enable : enabled
Result	None
Remarks	

h. Self-testing

Prototype	func testMorningSDK()

Description	Self-testing
Parameter	None
Result	None
Remarks	You can check the test results in the Xcode Console.

i. License configuration

Prototype	func setLicense(licenseData: String, configData: String) -> void
Description	License data configuration
Parameter	LicenseData : issued license data configData : Configuration for usage
Result	None
Remarks	

j. Logging

Prototype	func 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