



Promon SDK Protection™

Unlock the full potential of your SDKs and stay ahead of attackers

Promon SDK Protection™ delivers advanced security measures designed specifically for Software Development Kits (SDKs) across mobile platforms. By embedding robust code obfuscation and runtime controls directly into the compiled code, the solution ensures your SDKs—and the apps that rely on them—are shielded from reverse engineering, tampering, and unauthorized access.

Compatibility and support*

Platforms

-  iOS
-  Android (native, Java)

Languages

-  Swift
-  Rust
-  Objective-C
-  C
-  C++
-  Java
-  JavaScript
-  Kotlin

Architectures

-  ARM (arm64, armv7 32-bit)

In today's mobile ecosystem, a typical app integrates around 30 SDKs, with up to 90% of its code derived from external sources. This exposes SDK providers and their customers to security risks, increasing the need to safeguard their software's integrity. The threat is real: SDK misuse or compromise can trigger not only financial repercussions but also tarnish a company's reputation.

This widespread reliance puts SDK producers—including software companies, banking entities, game engine developers, streaming services, and specialized firms in identity verification, DRM, and security—under intense scrutiny. They face the challenge of adhering to stringent regulations like eIDAS 2.0, DORA, and PSD2 while also protecting their intellectual property from reverse engineering and mitigating security risks such as repackaging attacks.

A staggering 92% of companies experienced a breach in the previous year due to vulnerabilities in applications developed in-house. These apps can expose your SDKs to hackers, enabling them to uncover vulnerabilities they can exploit to target your business.

For SDK providers like you, it's not just about protecting your products. Reducing the chance of vulnerabilities being passed to host apps adds real value to your SDKs. You also need to boost security, address your customers' concerns confidently, and maintain trust.

*Support for additional platforms and CPU architectures is upcoming.

Your strategic outcomes with Promon SDK Protection™

Mitigate security vulnerabilities

- ✔ **Protect your valuable intellectual property**
SDK Protection uses binary code obfuscation to prevent unauthorized access, reverse engineering, adversarial analysis, and proprietary code theft. This means your unique algorithms and sensitive data remain exclusive.
- ✔ **Strengthen your security posture**
As an SDK producer, securing your SDKs directly mitigates the risk associated with operating on untrusted devices and within vulnerable applications. This proactive approach allows you to detect and react to attacks, effectively making your SDKs more resilient.

01

Reduce development and maintenance efforts

- ✔ **Seamless deployment**
SDK Protection uses a post-compile approach that can reduce the integration time and effort in your development frameworks. This requires no additional training, allowing your developers to focus on what they do best without worrying about security implementation details.
- ✔ **Lower maintenance costs**
SDK Protection is more cost-effective than typical security solutions that require adding obfuscation pre-compile into your development process. Changes in your framework, code, or workflow won't drive up your security expenses.

02

Enable uniform security across platforms

- ✔ With SDK Protection, you get equal security for Android and iOS SDKs, with plans to add more platforms and architectures. As your product lineup grows into new platforms, the SDK Protection solution seamlessly adapts, keeping your technology secure without requiring additional security expertise from your developers.

03

What makes Promon SDK Protection™ different?

Minimal developer impact

SDK Protection's post-compile approach streamlines its integration, preserving your existing codebase and workflows. Developers can concentrate on their work, free from the complexities of adding security measures.

Rapid deployment

Designed for efficiency, SDK Protection can be deployed quickly, integrating into your workflow with minimal disruption. This hassle-free process accelerates the path from development to secure deployment.

Runtime protection

Besides code obfuscation, which provides protection from static analysis, SDK Protection enhances security when the app is in use. It guards against dynamic threats like code injection, hooking, and unauthorized debugging activities, adding a critical layer of protection for apps using your SDKs.

Scalable security

SDK Protection grows with your business. It keeps pace with new and evolving platforms, providing a robust security framework that supports your innovation and growth. A simple command line tool update is all it takes to ensure this security solution remains adaptable and up-to-date.

Promon SDK Protection™ code obfuscation techniques



Section encryption

Section encryption ensures that a binary (executable or library) cannot be statically analyzed (i.e., understood while on disk). Much of the binary file is encrypted by "sections" to prevent such analysis. Decryption begins shortly after start-up and after runtime integrity has been proven. The derivation of the keys used to decrypt each section depends on whether the Shield library is in a valid state.

On the Java bytecode, it encrypts static strings like error messages, file names, API keys, URLs, etc.*



Control flow flattening

Control flow abstraction diverts call instructions within the code sections to a central dispatch function that hides the links between code blocks. With control flow abstraction, an attacker won't be able to see where a code jump goes to and whether it's to an external dependency or another internal symbol. Trying to extract a call graph from the code will be of limited use because all calls go to the same place, and the graph is effectively flattened.



Block splitting

However, block splitting can be useful if you have a large symbol with few or no dependencies or want to increase the obfuscation of particular symbols. Block splitting takes the code blocks in one or more symbols/functions, splits them into smaller fragments, shuffles them with unrelated code, and inserts jumps to reconnect the control flow. Block splitting happens before control flow abstraction, so if indirect links are chosen, this flow is hidden by control flow abstraction.



Integrity checking

To ensure that your code has not been tampered with (e.g., patched, hooked, or a breakpoint inserted), a checksum network covering all the application code is embedded. This feature is tightly integrated with control flow abstraction, so you must have control flow abstraction enabled to use integrity checking. By default, both features are enabled to a limited level.



Debug stripping

Binary libraries and executables contain a surprising amount of debug information, even on "release" builds and especially on "debug" builds. Debug stripping removes debug information from the binary. Effective use of the debug stripping tool requires that symbols are not stripped from the release binaries before they are protected.



Renaming (Java and Kotlin)

On Java and Kotlin, it renames classes and their members (i.e., fields and methods) to have meaningless names. It also flattens classes into a single package.*

Promon SDK Protection™ runtime controls



Hooking protection

Attackers can inject code into an Android or iOS app via code hooks, commonly facilitated by hooking frameworks. These injections are then used to modify the app, intercept messages, and read user input. SDK Protection detects the presence of code hooks and, based on your configurations, outright crashes the app and optionally calls an internal function for custom handling.



Root/Jailbreak protection

When an Android device is "rooted," it gains admin/root access to file locations that the manufacturer and/or carrier had originally restricted. On iOS devices, "jailbreaking" has a similar effect but is more about removing restrictions on the apps that can be downloaded and installed. These are actions that might be performed directly by an attacker, or they might be performed by a user who simply wants to customize their device. However, a rooted or jailbroken device is much more susceptible to malware, so it is essential to be aware of the risks. For both platforms, SDK Protection detects that the device's default restrictions are compromised and can be configured to act accordingly.



Debug protection

Attackers can run a debugger on an Android or iOS application to extract sensitive information and help them reverse engineer the app. SDK Protection detects the use of such debuggers and, based on your configurations, outright crashes the app and optionally calls an internal function for custom handling.

About Promon

Promon is the leader in proactive mobile app and SDK security. We make the world a little bit safer, one app at a time. Since 2006, some of the world's most impactful companies have trusted Promon to secure their mobile apps. Today, more than 1 billion people use a Promon-protected app. Promon is headquartered in Oslo, Norway, with offices throughout the globe.

[Learn more](#)

Would you like to talk to an expert?

Mobile app security is crucial to preserve and improve your business reputation.

Request pricing or talk to an expert to learn more today.

[Book a meeting](#)

Promon AS
Cort Adellers Gate 30
0251 Oslo
Norway