

Promon Jigsaw Engine

Empowering organizations to harden their apps against emerging threats

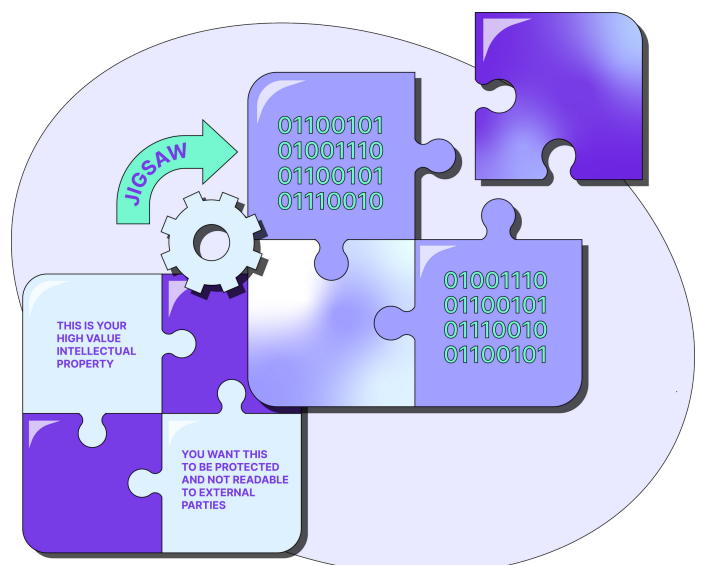
Promon Jigsaw is the world's first cross-platform, post-compile, native obfuscation engine that eliminates next-generation cyber threats. It integrates seamlessly with the Promon SHIELD™ app shielding platform, protecting apps from repackaging, reverse engineering, and manipulation. In addition, it blocks attackers from using techniques to remove SHIELD's in-app runtime protection. The Jigsaw engine allows for seamless configuration, as well as fine-grained control for more demanding use cases to help gaming, financial services, media and streaming, healthcare, and defense organizations.

The time for innovative binary obfuscation is now

The rise of binary obfuscation is vital for safeguarding intellectual property and code security, as traditional code obfuscation techniques have grown vulnerable to automated machine-learning attacks.

Current application shielding methods are either slow and costly or lack depth. Apple's recent deprecation of LLVM bitcode in Xcode further complicates MacOS and iOS app protection. To combat these challenges, advanced, user-friendly obfuscation techniques are urgently needed.

Promon's Jigsaw engine addresses this gap by providing comprehensive, low-impact code protection across platforms, streamlining integration for security developers and engineers.



The benefits of the Promon Jigsaw engine



Safeguard your valuable revenue streams

The Jigsaw engine's capabilities extend to protecting in-app purchases, subscription models, and premium features, ensuring your monetization strategies remain intact. With a hardened app, your business can grow without disruption.



Protect your prized intellectual property

The Jigsaw engine safeguards your app code by ensuring that proprietary algorithms, unique functionalities, and logic remain shielded from unauthorized access, imitation, or reverse engineering attempts.



Ensure data security and compliance

Minimize the risk of data breaches, ensuring your app aligns with regulations such as GDPR, PCI, PSD2, HIPAA, CPRA, CPA, and local industry-specific mandates. By incorporating the Jigsaw engine's robust security into your app development process, you can confidently meet compliance requirements and build trust with users concerned about their data privacy.



Reduce time to market with a simplified deployment

The Jigsaw engine is a low-code engine suitable to developers of varying expertise levels. Performance optimization is built in, and when specific configurations are needed, it allows explicit fine-tuning for an optimal balance between security and performance.



Benefit from cutting-edge protection against the latest threats

Driven by continuous research and development, the Jigsaw engine rapidly adapts to the latest threat trends and attack vectors. It will power future solutions, countering threats from reverse engineering and AI, thereby minimizing the potential for financial fraud.*

*Future capability

Why Promon Jigsaw engine

Binary obfuscation

The Jigsaw engine operates natively on the post-compile binary code. It is developer-tool agnostic, supporting a wide range of languages like Dart, Rust, Golang, C++, Obj-C, Swift, NDK, and Xamarin.iOS. It can also protect third party libraries and is not dependent of Bitcode.

Native implementation

As a native engine, the Jigsaw engine delivers reduced implementation friction. It runs quickly and efficiently compared to other market alternatives and integrates seamlessly into CI/CD pipelines.

Low-code versatility

Like all Promon products and tools, the Jigsaw engine is user-friendly and doesn't require specialized knowledge. Its default settings offer strong performance in most use cases. For specific needs, customization is straight forward.

Cross-platform

The Jigsaw engine's unified codebase offers the same features and experience across multiple platforms, including Android Native, Linux (desktop + embedded), MacOS, iOS, and Windows. It also supports ARM and Intel architectures.


Overview

Jigsaw vs. traditional obfuscation

	Source based	Intermediate based	Promon Jigsaw (Binary based)
Language support	Single e.g. C/C++	Limited by toolchain	Any compiled language
Platform support	Any	Limited by toolchain	Depends on product
Toolchain support	Limited by language support	Typically limited to LLVM-based toolchains	Any
Obfuscation level	Worst	Better	Best
Ease of integration	Medium – needs to be called before compiler but will need access to all compilation unites (source files)	Invasive, requires patching into toolchain	Easy – end of process typically applied to a single binary
Ease of use	Could be tricky to integrate with IDE and cause usability issues	Typically, invisible to end user all though may slow compilation down	Optionally can be applied or not and has no impact on the IDE or toolchain

Supported platforms and architectures

Platforms



Architecture

intel arm

Promon Jigsaw assists companies with

IP protection

Whether in gaming, defense, healthcare, or other sectors, the Jigsaw engine ensures advanced code protection against reverse engineering and unauthorized access and helps organizations remain in control of their data and intellectual property (IP).

Payment flow protection

By countering threats like reverse engineering and patching, the Jigsaw engine hardens the integrity of payment flows within native components in industries like banking and finance, retail, or other sectors reliant on secure financial transactions.

Data protection

The Jigsaw engine supports organizations to provide a high level of data and app security and integrity, particularly for industries dealing with personal data, such as healthcare and finance.

Content protection

Streaming and entertainment businesses can securely deliver their digital content, thwarting unauthorized access and preserving the integrity of their offerings.

The right obfuscation technique to keep your code protected

1

Control flow obfuscation hides the calls between functions and external dependencies. This makes it difficult to map the sequence of operation of the code or to gain understanding by looking at which library calls it is making.

2

Section encryption encrypts code and all static data. This makes initial disassembly of the code difficult and removes identifiable strings, structs, and constant values from the code.

3

Block splitting breaks up larger functions into smaller chunks, shuffles these, and links them via the control flow obfuscation, so they are difficult to follow.

4

Checksumming verifies the integrity of the code to protect it from modification, such as setting breakpoints or patching. By default, 100% coverage is achieved using multiple redundant ranges.

Get in touch!

Visit the [Promon Jigsaw engine page](#) to learn more or [schedule a demo](#) with one of our experts.