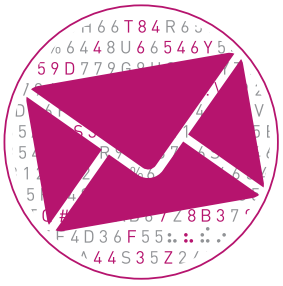
gemalto
security to be free



WHITE PAPER

The Sentinel Envelope

Table of Contents

Executive Summary

One of the biggest issues software publishers face in today's computing environment is how to prevent unauthorized use of their software without creating unnecessary obstacles for customers who wish to legitimately purchase and use it.

Copyright infringement of software, also known as software piracy, is facilitated by the abundance of reverse-engineering information found online, providing easily available tools and knowledge to everyone. It is a well-known fact that most countries have copyright laws that apply to software, but the degree of enforcement and compliance varies making some countries more "fertile" in terms of infringement practices.

Software piracy is an ever increasing problem as it is widespread, difficult to trace and even harder to prevent and negate. Software piracy stunts revenue potential and negatively impacts paying customers, who ultimately bear the cost of illegal product use. Software vendors who proactively protect their software are on the right track but may not be fully protected against the ever-growing hacking attempts that can compromise their application's security. A common misconception is that once a certain application is protected and distributed it is completely "bullet-proof" against software piracy and Intellectual Property theft. It is crucial that software vendors work with the licensing vendor and/or hardware protection manufacturer to constantly update and improve the level of security. Incorporating innovative protection and security measures, as part of the product lifecycle, can greatly contribute to being steps ahead of potential threats.

> Software piracy is an ever increasing problem as it is widespread, difficult to trace and even harder to prevent and negate. Software piracy stunts revenue potential and negatively impacts paying customers, who ultimately bear the cost of illegal product use.

This paper examines a variety of mechanisms available as part of the Sentinel Envelope for protecting applications against software piracy and Intellectual Property theft.

Overview

Packers – Definition and Usage

Packer is, as the name suggests a tool that modifies executables and creates new equivalent files for the purpose of compression or as a reverse engineering protection method. Packers operate in a similar manner to the Russian Matryoshka doll – a doll within a doll within a doll principal. The process of packing can add one (or more) sections to the original executable in addition to attaching a loader which 'unpacks' the program before resuming normal execution. The loader is the part of the operating system responsible for loading the application while performing real-time tasks such as anti-debugging, tracing detection, license management, and background checks.

> Enveloping combines encryption and native code obfuscation to provide the strongest protection to date enabling the protection of your valuable Intellectual Property.

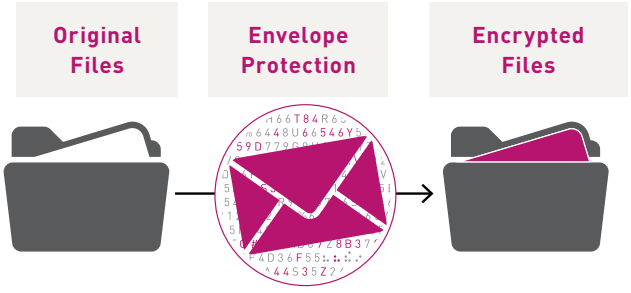
The Sentinel Envelope - One-Click, Easy-to-Use Solution

The Sentinel Envelope is an automatic file packer that provides protection against software reverse engineering through file encryption, code obfuscation and system-level anti-debugging. The process of packing (or wrapping) executable files and dynamically linked libraries ensures that algorithms, trade secrets, and professional know-how are secured against crackers.

The Sentinel Envelope secures an application by adding a protective shield responsible for binding the application to either **hardware** or **software**-based protection keys.

Protecting with the Sentinel Envelope is a procedure that takes only a few seconds, assuming that the default protection scheme is chosen. The process is slightly extended if additional steps and measures are taken in order to use some or all of its available options, providing an extremely powerful platform for software vendors who have no access to the application's source code. For example, resellers and dealers that sell unprotected software can use the basic default Envelope settings in order to protect the products for their local markets—an easy and rapid process.

When the protected application is launched, the Sentinel Envelope loader (part of the Envelope run-time) sends a query to the protection key validating its existence. If a valid Sentinel protection key exists, the Envelope loader, in tandem with the encryption engine of the protection key, decrypts the application file which was previously encrypted by the developer. If the Sentinel protection key does not exist or is invalid, the application halts and will not execute. Additionally, if the protection key is not available the binary will not be decrypted.



The Data File Encryption Utility

Aside from protecting the executable of an application, encrypting the data files being accessed by the application ensures protection of the Intellectual Property. This places an added layer of security between the hacker and the Intellectual Property of the software. DataHASP, the Data File Encryption utility, in tandem with the Sentinel Envelope, utilizes data file encryption to pre-encrypt data files which are then encrypted or decrypted by the protected application. Following the encryption phase, data files can only be accessed if a suitable protection key is detected.

Under the Hood

The Sentinel Envelope as a whole provides robust Intellectual Property (IP) protection against reverse engineering through the use of its highly advanced features such as: file encryption, code obfuscation, system-level anti-debugging, White-box Cryptography, AppOnChip and more. The implementation of these additional features make it extremely complex and time consuming for hackers to breach, thus ensuring that software code is safe from exposure while en-route to its end-user destination. Each feature takes into account the various techniques that hackers use when trying to infiltrate an application.

Anti-Debugging and Anti-Tracing Methods

Normally, debuggers are used by software developers to detect bugs and trace problems during the application development process. However, hackers trying to gain illegal access to software use the same debuggers to detect and trace the implanted protection code with the ultimate goal of changing, disabling, or removing it altogether.

An extremely powerful feature of the Sentinel Envelope is its debugger detection mechanism, which is constantly on the prowl for active debuggers. By sending misleading commands and false information to the debuggers, the Envelope succeeds in distracting them from their prime job. Moreover, the Envelope easily identifies debuggers, allowing the ISV to decipher between friend and foe. The Sentinel Envelope is also architected to detect whether anti-tracing tools have been initiated and halt the protected application from running when necessary.

Since both hackers and developer groups use the same debugging tools, the Sentinel Envelope must have the ability to distinguish between debugging activities of an innocent developer and that of someone intending to do harm. This is achieved by displaying a message that a debugger has been detected and preventing the protected application from loading. A developer will turn off the debugger at this stage to enable the application to load properly and run. However, if a debugger is activated after the application loads and runs, clearly this is the activity of a software "pirate" attempting to crack the software, and thus the application halts.

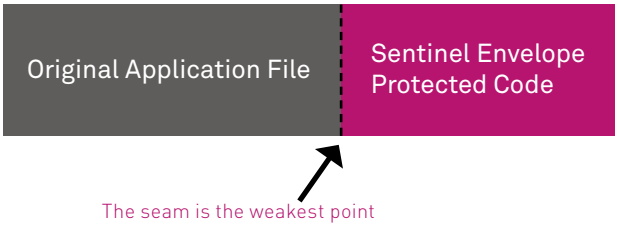
Securing the Weakest Point

The weakest point in an application protected with any packing mechanism is the seam between the application file and the added protection code. This is the point that, once annulled, will disconnect the link to the protection key containing the license, leaving the application completely unprotected. Consequently, this is the point at which most attackers will attempt to strike. Crackers will study the protected file analyzing the protection code and how it is linked to the attached protection key. Once they understand the code and recognize its location, they can then operate in one of the following manners:

Sentinel Envelope Features and Benefits

- **Automatic File Wrapper -** Provide robust protection against software reverse engineering through file encryption and native code obfuscation
- **Reconnection of the Application to the Hardware -** The application is now tightly coupled with the Hardware by means of a protection key
- **Secure Communication Channel -** Sentinel eliminates man-in-the-middle attacks by providing a secure channel for communication between the protected application and the protection key. The Java Envelope uses this ability to prevent a hacker from intercepting communications to access data sent back from the protection key
- **Runtime Decryption -** Because Sentinel decrypts files as they are requested at runtime rather than loading all the .class files into the virtual machine at once, it prevents hackers from rebuilding the entire application
- **Multiple Operating System Support -** Sentinel Envelope protects your Intellectual Property against reverse engineering and tampering on multiple platforms: Windows(x86); Linux(x86, x86_64) on ARM; Android (for Java apps).

- **Application-Specific Crack** – Break the protection link for the specific application file.
- **Generic Crack** – Break the protection link for all other files protected by the same mechanism if the exact same method appears in all of them repeatedly.It is, therefore, essential that the seam between the protected file and the added protection code be ambiguous and untraceable, presenting a long and tiresome procedure for anyone trying to understand the protection mechanism. One of the strongest features of the Sentinel Envelope is its ability to protect the seam and present numerous obstacles that prevent the protection link from being broken. This is achieved by supplying a multi-layered protection code, added onto the application during the protection process. These layers are pieces of code specially designed to fit one-after-the-other like train cars. In each protection session, the Sentinel Envelope ensures that the various layers constructing the entire code are organized in a different sequence when added to the original application file.

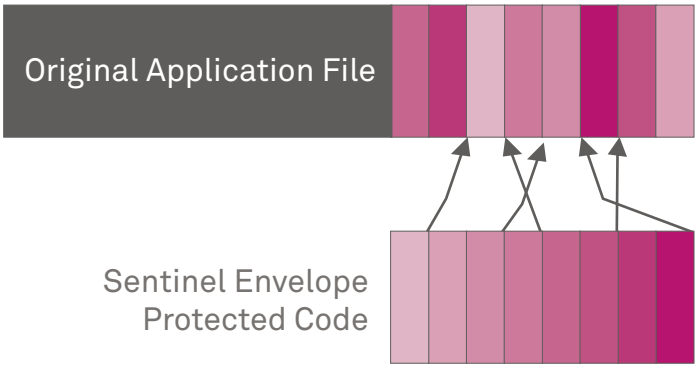


The dynamic arrangement of the layers differs in every Envelope protection session ensuring that each protected file is unique. There is no resemblance between protected files, even if the original files are completely identical. The transition from the last instruction in the Envelope code to the first instruction in the application code differs between protected applications. For each application, the original code starts at a different place making the Envelope application seam almost impossible to trace. Learning and understanding the different layers and their layout within the protected file implies nothing about the layout in the same file protected in another Envelope session. To make it

even more difficult to break, the Envelope not only arranges the layers differently, it also selects a different number of layers for each file it protects. Furthermore, the layers are encrypted, each one in a different way. And, during application runtime, each layer is responsible for decrypting the next layer in the sequence using a random encryption key.

> By allowing ISVs to encrypt certain pieces of an application or the entire application file, the Sentinel Envelope provides a multitude of security feature configurations based around individual needs.

Confused? There's more! The code in each layer is obfuscated, by using dummy op-codes, which are inserted between valid code instructions. This severely obstructs the ability to investigate the code and ensures that using disassemblers to analyze the protection mechanism or the disassembled code becomes a futile task.



By wrapping the source code, the Sentinel Envelope provides robust protection against reverse engineering protecting valuable algorithms and trade secrets. Each file protected with the Envelope is encrypted using a different random seed, resulting in very different files following the protection phase, even if the originals were identical. The application file is divided into multiple blocks, which are scalable and can be predetermined by the developers during the protection phase. Each block is encrypted using AES encryption using different arbitrary seeds.

Vary Behavior when Cracking Attempt is Detected

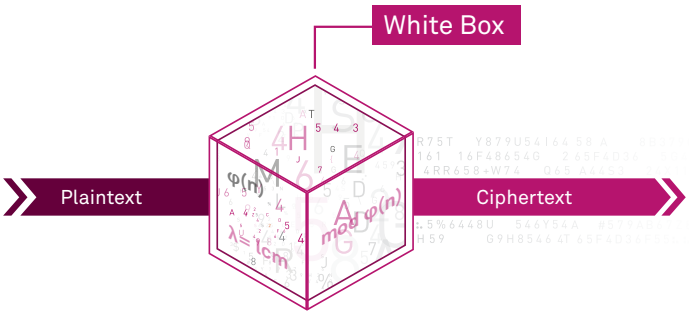
Another technique used by the Sentinel Envelope to fight debuggers is referred to as "behavior alteration". The protection keys containing the license employ a sophisticated code design that takes advantage of the fact that the operating system and the debugger execute applications differently. When a cracking attempt is detected (for example, through using an integrity check), the reactive behavior of the software is delayed, thus breaking the logical connection between "cause" and "effect." Delayed reaction confuses the cracker by obscuring the true logical link between the cracking attempt and the negative reaction of the software to that specific attempt. Behavior such as impairing program functionality when a cracking attempt is detected can be very effective.

Private API

Most software protection vendors provide the same API library to all customers; making the library a single point of failure if a security breach occurs. Gemalto employs a far more secure solution – ISV-specific API libraries. These Private APIs are built and customized on Gemalto servers, away from the prying hands of crackers. These Private APIs make sure each vendor gets a structurally different component to be integrated into their application. As part of this process, the Private API's which are customized differently for every ISV, are augmented with unique white-box cryptography secrets and finally go through heavy obfuscation and protection techniques. The resulting libraries are virtually immune to generic cracks and ensure that, as a rule, crackers cannot make progress breaching the security of one vendor API library and expect to make any gains against other vendors. Sentinel Envelope fetches these Private APIs from a copy of downloaded APIs on the developer's machine and injects them into the application at protection time. These strongly protected ISV-specific APIs are then used by the Envelope Runtime for legitimate access to ISVs protected application.

Multiple, Non-obtrusive Calls to the Sentinel Protection Key

The beauty of Sentinel Envelope is that it is applied to a compiled file which ensures there is absolutely no need to modify the source code of the application. Calls to the protection key are executed periodically by the protection code that gets added to the application file (the Envelope run-time). The Sentinel Envelope allows the security integrator of an ISV to specify and configure the time intervals in which the Sentinel protection keys are checked, challenging their presence using cryptographic means. This is just one of the many parameters that are fully configurable by the ISV to be used during protection phase.

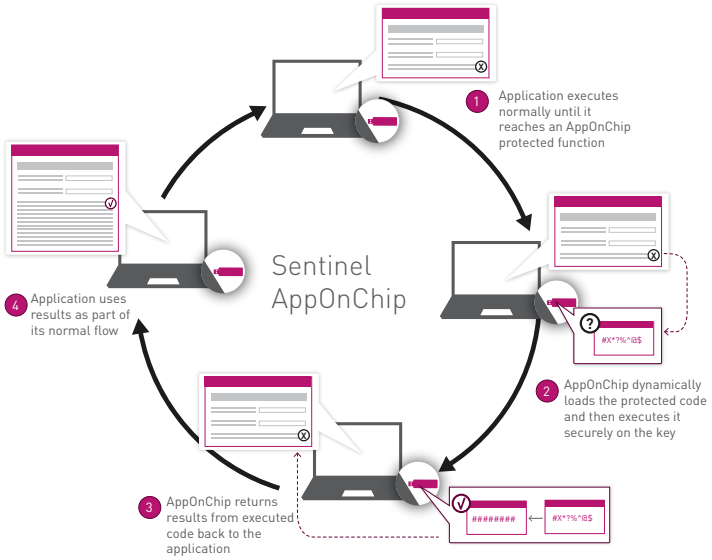


White-Box Cryptography

Gemalto is the industry pioneer in utilizing white-box cryptography to fully encrypt the communication channel as a means of preventing attackers from understanding the communication between the protected application and the protection key. The white-box-based secure channel communication utilizes vendor-specific components ensuring that the secure channel encryption key cannot be extracted from the protected binaries—both dynamically and statically.

AppOnChip

One of our most secure features of Sentinel Envelope, AppOnChip, facilitates an inseparable binding of the Sentinel hardware key to the application, providing software publishers with the most secure software protection solution available. This fully automated process presents software vendors with a list of functions from their application that contains code blocks that are compatible with the AppOnChip feature. The protected code blocks, encrypted and signed, can then be loaded and executed on the hardware key itself. This additional security measure makes it the most secure software licensing implementation in the market. Features and Benefits of AppOnChip include Stronger Security, Easy Implementation, Maximum Licensing Flexibility, End User Transparency, and No Operational Burden.



Original Entry Point (OEP) Protection

Original Entry Point (OEP) refers to the startup address of any application from which the Operating System (OS) starts to run the application. In order for crackers to unpack the protected application they must locate this address, remove the packer code and attempt
to start the application from the Original Application Entry Point.

> Sentinel Envelope provides out-of-the-box top-notch security without spending the time and effort to develop a solution from scratch while allowing your engineering teams to focus on their core competencies.

Unlike other packers, the Sentinel Envelope removes the Original Entry Point instructions from its default location and scatters pieces inside the Envelope run-time code. A cracker attempting to find and reconstruct the Original Entry Point from the spread out chunks will
hit a brick wall as this is virtually impossible considering the randomness of the location and chunk sizes.

Method-Level Protection

The Sentinel Envelope enhances the protection of .NET and Java executable by defining Method-level protection. When a .NET or Java assembly is selected for protection, Sentinel Envelope automatically determines the methods that are available for individual protection. This further ensures security to the Intellectual Property and provides the best protection for any application.

Import Address Table Removal

An additional means of circumventing cracking attempts is the process of removing the Import Address Table which contains addresses to functions in external DLLs that are used by the protected application. The process of packing the original application removes the Import Address Table so that it doesn't exist on disk or in memory and scatters this information inside the Envelope code. This means that each import address operation is protected and handled internally by the Envelope. In addition, each import operation resolves to a different memory location with a different obfuscated code so that the cracker has to analyze and understand each import operation separately in order to get a piece of the puzzle. In classic protection packers, the Import Address Table allows crackers to distinguish when they are done analyzing each entry in the table. In the case of the Sentinel Envelope, the Import Address Table is not used and the cracker has no table to enumerate through – therefore they will always have uncertainty as to whether they were able to complete the task or not. To top it all, the Sentinel Envelope uses various techniques to hide import, which will make the cracked application, fail at a later stage, rendering a "successfully cracked" application totally unusable in the long run.

Strong Binding of Original Code and Envelope Code

In most common packers, there is no binding between the original application code and the packer's code. The Sentinel Envelope tightens the virtual bond between the packer and the protected application; Sentinel Envelope integrates itself into the application flow based on a code flow analysis done during protection time. This allows an indistinguishable integration of protection measures into the application preventing its removal by the attacker. At run-time once the control flow reaches these designated addresses, an explicit execution sequence performs various validation and verification operations while continuing with the original application's code. If the flow is intact, the application will run; otherwise, if the application's integrity is in question – the process halts.

Stolen Bytes

Memory snapshots and dumping are commonly used techniques, which in some circumstances, are able to provide crackers with insight on the original application source code. This is a crucial first step for any hacker attempting to crack the protected application and exactly where a successful anti-hacking solution needs to excel.

The concept of "stolen bytes" refers to strengthening the dependency between the protected application and the Envelope code. The act of stealing bytes refers to selecting chunks of random bytes from different locations of the original source code and scattering them randomly inside the Envelope code. These chunks of code (stolen bytes) execute in new random locations while executing the protected application's original code. This mechanism enhances the dependency of the original application code to the Envelope code by blurring where the original application code ends and the Envelope code starts.

Code and Symbol Obfuscation

Obfuscation is the process of turning meaningful strings into random strings of letters or numbers. Using Sentinel Envelope, an ISV can apply obfuscation as an anti-reverse engineering security measure. By default, all symbol names in the protected .NET assembly are obfuscated as part of the protection process. In addition, ISV can choose to obfuscate the entire code of a selected method. Since code obfuscation may slow the performance of an application, it is not selected by default. An ISV can apply Code obfuscation to a method regardless of whether it is selected for protection in the list of Methods to be protected.

Conclusion

By allowing ISVs to encrypt certain pieces of an application or the entire application file, the Sentinel Envelope provides a multitude of security feature configurations based around individual needs. Security comes at a certain cost and, as a direct result, cannot be air tight. It is therefore crucial that one properly evaluates the required security level as dictated by the application itself i.e. the value of what needs to be protected in conjunction with the incurred losses assumed by neglecting potential risks.

By actively preventing the competition from apprehending trade secrets and know-how an ISV can truly prevent industrial espionage and enhance competitive advantage. Enveloping combines encryption and native code obfuscation to provide the strongest protection to date enabling the protection of your valuable Intellectual Property. Furthermore, the Sentinel Envelope provides out-of-the-box top-notch security without spending the time and effort to develop a solution from scratch while allowing your engineering teams to focus on their core competencies.

Gemalto Sentinel Software Monetization Solutions

Gemalto, through its acquisition of SafeNet, is the market leading provider of software licensing and entitlement management solutions for on-premises, embedded and cloud-based software vendors. Gemalto's Sentinel is the most trusted brand in the software industry for secure, flexible, and future-proof software monetization solutions.

Easy to integrate and use, innovative, and feature focused, the company's family of Sentinel Software Monetization Solutions are designed to meet the unique license enablement, enforcement, and management requirements of any organization, regardless of size, technical requirements, or organizational structure. Only with Gemalto are clients able to address each and every aspect of the software monetization lifecycle—from copy and intellectual propertyprotection to product catalog management and ongoing enduser experience improvement.

With a proven history of adapting to new requirements and introducing new technologies to address evolving market conditions, Gemalto's customers around the globe know that by choosing Sentinel, they choose the freedom to evolve how they do business today, tomorrow, and beyond.

Download a FREE Sentinel LDK Demo Kit which includes Sentinel Envelope, Sentinel EMS and Sentinel HL Keys. To explore the features of Sentinel Envelope now, visit:

www5.gemalto.com/sentinel-ldk-trial-en

Join the Conversation

- **> Facebook**
  www.facebook.com/licensinglive
- **> LinkedIn**
  bit.ly/LinkedInLicensingLive
- **> Twitter**
  twitter.com/LicensingLive
- **> Google+**
  plus.google.com/u/2/106533196287944993975/posts
- **> Sentinel Video Cloud**
  sentinelvideos.safenet-inc.com/
- **> Blog**
  http://www.licensinglive.com/
- **> Sentinel Customer Community**
  sentinelcustomer.gemalto.com

Contact Us: For all office locations and contact information, please visit www.gemalto.com/software-monetization

Follow Us: www.licensinglive.com

GEMALTO.COM

gemalto
security to be free