cpl.thalesgroup.com

THALES
Building a future we can all trust

The Sentinel Envelope

White Paper

Contents

Executive Summary

One of the biggest issues software vendors 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 a very widespread problem which is 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 applications' 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 provider of the software protection solution 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 remaining 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 Sentinel LDK Envelope for protecting applications against software piracy and intellectual property theft.

Overview

Packers – Definition and Usage

A 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. The process of packing adds protection code to the original executable that 'unpacks' the program, or parts of it, before resuming execution. This code, referred to below as LDK Envelope Runtime, is required for the protected application to run and is responsible for tasks such as anti-debugging, tracing detection, license management, and background checks.

Enveloping combines encryption and code obfuscation for the utmost in file packer solutions for copy and IP protection.

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

Sentinel LDK Envelope is an automatic file packer that provides protection against software reverse-engineering through file encryption, code obfuscation, and anti-debugging. The process of packing (or wrapping) binaries (executable files and libraries) creates a strong binding between the licensing system and the application for the utmost in copy protection technology.

Sentinel LDK Envelope secures an application by adding a protective shield responsible for binding the application to either hardware-, software- or cloud-based protection keys (HL, SL and CL keys respectively).

Protecting an application with Sentinel LDK Envelope is a procedure that takes mere seconds, providing an extremely powerful solution for software vendors with practically no effort.

Packers can also be used by vendors who have no access to the application's source code. For example, a vendor can protect the application without requiring developers to be involved in protecting the software. This can also be used by distributors and resellers who may wish to protect the software for their local market.

When the protected application is launched, the LDK Envelope Runtime attempts to log in to an LDK license, and, if successful, utilizes the encryption of the HL, SL or CL protection key to decrypt the application, or parts of it, in memory. If the Sentinel protection key does not exist or the license is invalid, the binary cannot be decrypted into memory and the application cannot start. If the key or the license becomes invalid during runtime, the application will halt.

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 cracker (a hacker who tries to remove the protection) and the intellectual property of the software. Sentinel LDK Data Protection utility, in tandem with Sentinel LDK Envelope, utilizes data file encryption to pre-encrypt data files which are then encrypted or decrypted by the protected application. Following the encryption phase (performed using Sentinel LDK Data Protection utility), data files can only be accessed if a suitable protection key is available to the Envelope-protected application during runtime.

Specific data files can require dedicated licenses for decryption, providing the vendor, or a content publisher, the means to license documents, media files, and other content.

Under the Hood

Sentinel LDK 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, anti-debugging, AppOnChip, and more. The implementation of these additional protection technologies makes breaching the protection very complex and time-consuming for crackers, thus ensuring that software code is strongly bound to the end user license. Each feature takes into account the various techniques that crackers use when trying to bypass the protection of the application.

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 bypassed, will disconnect the link to the protection key containing the license, leaving the application completely unprotected. Consequently, this is the point that most attackers will attempt to strike. Crackers will study the protected file, analyzing the protection code and how it is linked to the protection key. Once they understand the code and recognize its location, they can then attempt to operate in one of the following manners:

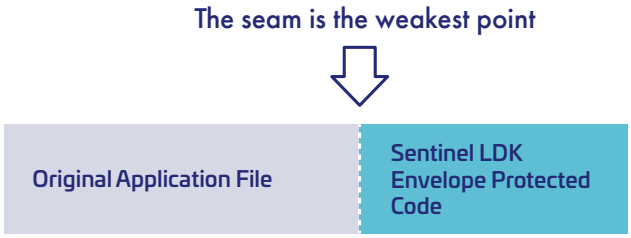
- **Application-Specific Crack** – Break the protection link for the specific application file.
- **Generic Crack** – Generate a fake license that works with all applications that depend on it to execute, or break the protection link for all other files protected by the same mechanism if the exact same 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, requiring a long and tiresome procedure for anyone trying to understand the protection mechanism. One of the strongest features of Sentinel LDK Envelope is its ability to protect the seam and present numerous obstacles that prevent the protection link from being broken.

This is achieved by changing parts of the application code to require the LDK Envelope Runtime code during execution. Removing the LDK Envelope Runtime would prevent the application from running correctly, and reverse engineering it is extremely difficult as it's heavily obfuscated and uses integrity checks that prevent modification.

In addition, when protecting .NET and Java applications, Sentinel LDK Envelope provides method-level protection, where methods get decrypted during runtime, just-in-time. This can be applied to any number of methods (even thousands), and is extremely time-consuming to circumvent.

Sentinel LDK Envelope automatically selects for protection only the methods that are less likely to impact performance, and the developer can override and deselect or select specific methods or classes in case of special security or performance needs. For native binaries, Envelope takes a different approach if performance is a concern, by allowing the software vendor to configure the percentage of code that gets encrypted. Normally, however, this is not an issue, and protected native applications perform very similarly to their unprotected versions.

Sentinel LDK Envelope also provides another layer of protection for these applications – assembly-level encryption (Windows shell protection) and class-level-encryption for Java. These provide the benefit of rendering decompilers and other reverse-engineering tools ineffective, because the encrypted files are no longer recognized as valid .NET assemblies or Java Class files by the reverse-engineering tools.



There is no resemblance between protected files, even if the original files are completely identical.

Sentinel LDK Envelope allows software vendors to encrypt certain pieces of an application or the entire application file – this is one of many configuration options that allows software vendors to tailor the protection to their individual needs.

Sentinel LDK Envelope Features and Benefits

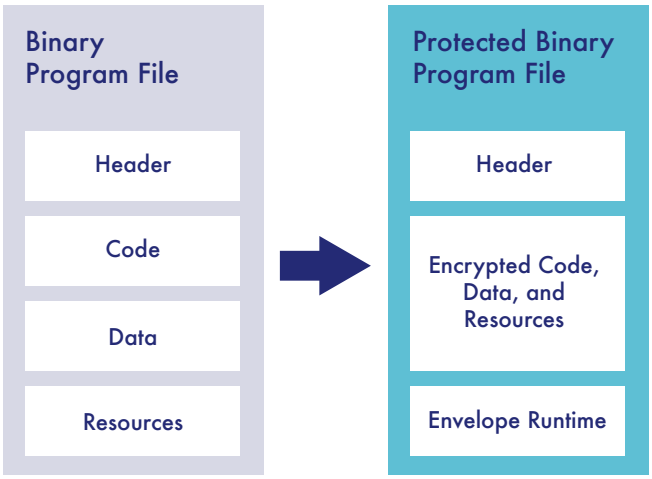
- **Automatic File Packer - Provides robust protection against software reverse-engineering through file encryption and code obfuscation**
- **Secure Communication Channel - Sentinel LDK Envelope eliminates man-in-the-middle attacks by providing a secure channel for communication between the protected application and the protection key.**
- **Runtime Decryption – Functions get decrypted during runtime, just-in-time, ensuring that the application cannot be dumped from memory in its entirety.**
- **Multiple Platform Support - Sentinel LDK Envelope secures your applications, ensuring strong copy protection and protection against reverse engineering and tampering on multiple platforms: Windows (x86, x86_64); Linux (x86, x86_64, ARM), Mac (x86_64). .NET and Java applications are supported on multiple operating systems.**

By encrypting the binary, adding licensing checks, detecting and stopping cracking attempts and implementing measures to hinder analysis of protected programs, Sentinel LDK Envelope is the tool of choice to protect software from illegitimate use and to protect valuable algorithms and trade secrets from prying eyes.

LDK Envelope Runtime code is dynamically created at protection time. It includes the vendor-specific secrets and is obfuscated differently each time.

This severely obstructs the ability to investigate the code and ensures that the use of disassemblers to analyze the protection mechanism or the disassembled code becomes a futile task.

The Envelope Runtime code may increase the size of the binary file, but in most cases the increase in size is negligible compared to the original binary size. In some cases, the Envelope file compression may even reduce the size of the binary.



Anti-Debugging and Anti-Tracing Methods

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

An extremely powerful feature of Sentinel LDK Envelope is its anti-debugging mechanism, which is constantly on the prowl for active debuggers. Applications that are running under the control of a debugger perform slightly differently. Envelope detects and exploits this, making debuggers an ineffective tool for analyzing the application to remove the protection. Sentinel LDK Envelope is architected to detect whether anti-tracing tools have been initiated and to stop the protected application from running when necessary.

Since both crackers and developers use the same debugging tools, Sentinel LDK Envelope must have the ability to distinguish between debugging activities of a legitimate 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 not try to bypass this message and will instead use the unprotected application for legitimate debugging. However, if this debugger check is bypassed, clearly this is the activity of a software cracker attempting to remove the protection of the software, and thus the application halts.

Cracking Attempt Detection

Additional techniques are used by Sentinel LDK Envelope to detect cracking attempts. For example, Envelope leaves "traps" in the binary which include code that would normally never get executed. If a cracker probes these execution code paths, it is an obvious sign of tampering which can lead to key or application disabling.

Integrity Checks

The protected application is signed with a digital signature. At run-time, the signature is checked to ensure that the entire binary (both code and resources) was not modified.

The Envelope program integrity protection is compatible with Microsoft Authenticode signatures. Both can be applied to the binary without affecting one another.

Automatic Key Disabling

A very effective response to a cracking attempt is to disable the protection key, stopping crackers in their tracks. This feature is supported by Sentinel HL Driverless and CL keys and is reversible by the vendor – the key can be re-enabled if the user provides a convincing reason to allow them access to the key again, at which point the vendor provides a digitally signed update file that re-enables the key. Although key disabling is not enabled by default, it is highly recommended to increase security. In Thales' experience, key disabling is extremely unlikely to result in false positive detection – no such events have been reported in the years since this feature was introduced.

Automatic Application Disabling

For Sentinel SL keys, where key disabling cannot be effectively implemented, Sentinel LDK Envelope uses an alternative technique. When a cracking attempt is detected (for example, through the use of 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 (for example, the application may quit with an internal error).

Vendor-specific API libraries

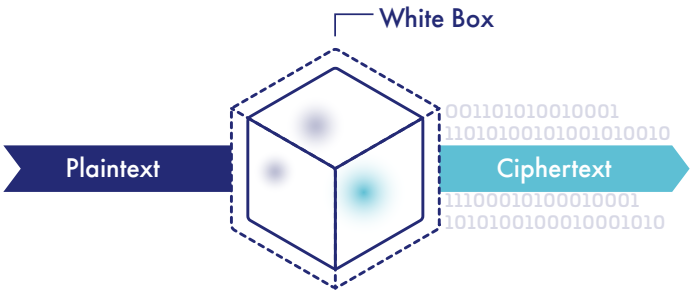
Most software protection vendors provide the same API library to all customers, making the library a single point of failure if a security breach occurs. Thales employs a far more secure solution – vendor-specific API libraries. These API libraries are built and customized on Thales servers, away from the prying eyes of crackers. They ensure that each vendor gets a structurally different component to be integrated into their application. As part of this process, the API libraries, which are customized differently for every software vendor, are augmented with unique white-box cryptography secrets and, finally, go through code obfuscation and other protection techniques. The resulting API 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 then expect to make any gains against other vendors. Sentinel LDK Envelope fetches these vendor-specific API libraries from a copy of downloaded APIs on the developer's machine and binds them into the application at protection time. These strongly protected vendor-specific APIs are then used by Envelope during runtime to enable legitimate access by the protected application to the protection key.

Multiple Calls to the Sentinel Protection Key Automatically Integrated by Sentinel LDK Envelope

The ingenuity of Sentinel LDK Envelope is that it is applied to a compiled file, which ensures there is no need to modify the source code of the application. Calls to the protection key are executed periodically by the protection code (the LDK Envelope Runtime) that gets added to the application file. Envelope allows the security integrator of the software vendor to specify and configure the time intervals at 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 vendor to be used during the protection phase.

White-Box Cryptography

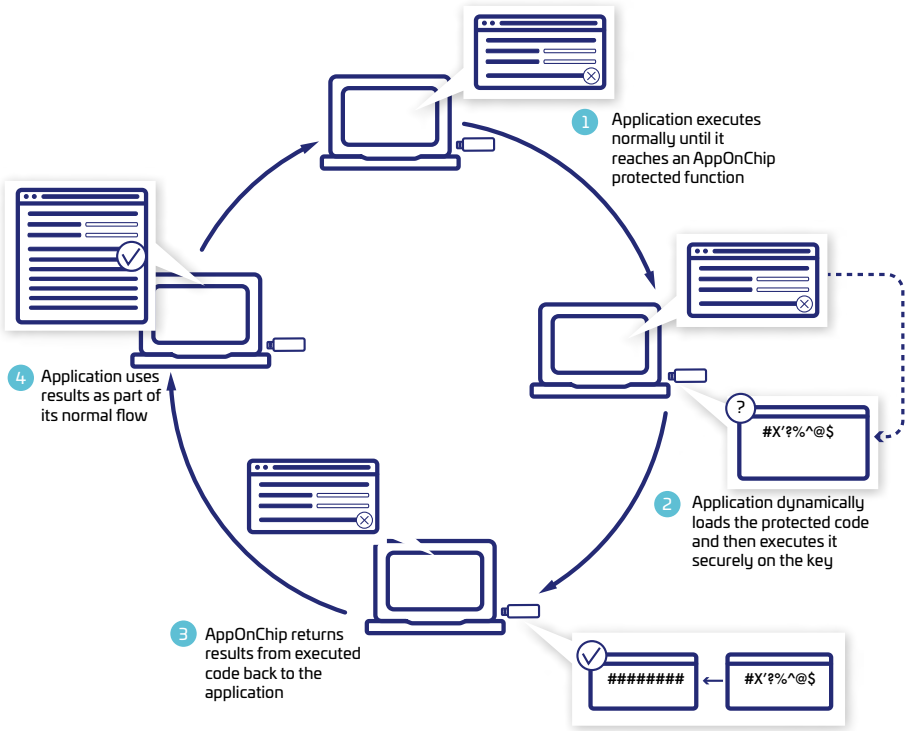
Thales is the industry pioneer in utilizing white-box cryptography to fully encrypt the communication channel as a means of preventing attackers from understanding the communication between the protected application and the protection key. The white-box-based secure channel communication utilizes vendor-specific components, ensuring that the secure channel encryption key cannot be extracted from the protected binaries, regardless of whether dynamic or static attack is used.



White Box Cryptography

AppOnChip

One of Thales' most advanced and innovative features of Sentinel LDK Envelope, AppOnChip, facilitates a virtually inseparable binding of the Sentinel hardware key to the application, providing software vendors with the most secure software protection solution available. This fully automated process presents the software vendor 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, and no operational burden, as the encrypted code is sent to the key for execution during runtime and does not require pre-loading.



The AppOnChip feature is available on Windows for native and .NET applications protected using HL Driverless keys. An automated runtime performance profiler is available to help vendors balance between the security and performance of AppOnChip integration.

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.

Unlike many packers, Sentinel LDK Envelope removes the Original Entry Point instructions from its default location and scatters pieces inside the LDK Envelope Runtime 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.

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

Method-Level Protection

Sentinel LDK Envelope enhances the protection of .NET and Java binaries by defining method-level protection. When a .NET assembly or Java archive is selected for protection, Envelope automatically determines the methods that are available for individual protection. This allows the vendor to select which methods to protect and how, resulting in the highest level of protection while minimizing any impact to performance.

Import Address Table Removal

An additional means of circumventing cracking attempts of Windows native binaries 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 LDK Envelope Runtime. This means that each import address operation is protected and handled internally by 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 traditional protection packers, the Import Address Table allows crackers to distinguish when they are done analyzing each entry in the table. With Sentinel LDK Envelope, the Import Address Table is not used, and therefore tools the cracker would normally use to reconstruct this table are rendered useless. In addition, Envelope uses various techniques to hide imports which will make the cracked application fail at a later stage, rendering a "successfully cracked" application partially broken and therefore undependable.

Strong Binding of Original Code and Envelope Code

In many common packers, there is no binding between the original application code and the packer's code. Sentinel LDK Envelope tightens the virtual bond between the packer and the protected application by integrating itself into the application flow based on a code flow analysis done at protection time. This allows an indistinguishable integration of protection measures into the application, preventing its removal by the attacker. At runtime, 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 that, in some circumstances, are able to provide crackers with insight on the original application logic. This is a crucial first step for any cracker attempting to circumvent the protected application and exactly where a successful anti-cracking 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 binary 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 LDK Envelope Runtime code by blurring where the original application code ends and the Envelope code starts.

Symbol and Code Obfuscation

Symbol Obfuscation is the process of turning meaningful strings into random strings of letters or numbers. Using Sentinel LDK Envelope, a software vendor can apply obfuscation as an anti-reverse-engineering security measure. By default, all symbol names with private visibility are obfuscated in protected .NET assemblies. In addition, software vendors can choose to obfuscate code of selected methods. Since code obfuscation may slow the performance of an application, it is not selected by default. Vendors can apply code obfuscation to a method regardless of whether it is selected for symbol obfuscation or encryption in the list of methods to be protected.

Conclusion

By allowing software vendors to configure the protection of their applications and select which protection technologies to integrate, Sentinel LDK Envelope can meet different individual needs, providing all the tools to implement the strongest copy protection. Security may come at a certain cost to performance and ease-of-use. It is therefore crucial that the vendor properly evaluates the required security level based on the level of threat (that is, the value of what needs to be protected) in conjunction with the incurred losses assumed by neglecting potential risks.

In addition to copy protection, by actively impeding the competition from accessing trade secrets and know-how, a software vendor can hamper industrial espionage and maintain competitive advantage. Enveloping combines encryption and native code obfuscation to provide strong protection for your valuable intellectual property.

Sentinel LDK Envelope is designed with ease-of-use in mind, providing out-of-the-box top-notch security without spending the time and effort to develop a protection solution from scratch while allowing your engineering teams to focus on their core competencies.

Download a FREE Sentinel LDK Demo Kit that includes Sentinel LDK Envelope. To explore the features of Sentinel LDK Envelope now, visit: https://www5.thalesgroup.com/sentinel-ldk-trial-en

Thales Sentinel Software Monetization Solutions

Thales is the market leading provider of software licensing and entitlement management solutions for on-premises, embedded and cloud-based software vendors. Thales 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 Thales are clients able to address each and every aspect of the software monetization lifecycle—from copy and intellectual property protection to product catalog management and ongoing end user experience improvement.

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

About Thales

The people you rely on to protect your privacy rely on Thales to protect their data. When it comes to data security, organizations are faced with an increasing number of decisive moments. Whether the moment is building an encryption strategy, moving to the cloud, or meeting compliance mandates, you can rely on Thales to secure your digital transformation.

Decisive technology for decisive moments.

THALES

Building a future we can all trust

Contact us

For office locations and contact information, please visit
cpl.thalesgroup.com/software-monetization/contact-us

> cpl.thalesgroup.com/software-monetization <