



SENTINEL SHELL: A Technical Overview

For Anti-Piracy, Licensing, and Code Obfuscation



INTRODUCTION: THE NEED FOR SOFTWARE PROTECTION

Reverse engineering is the process of discovering the technological principles of a device through analysis of its structure or operation. For software applications, this means hackers are capable of analyzing the components of a software program with the intent of creating a new one with similar functionalities.

By reusing source code for software applications, competitors have the opportunity to steal the intellectual property of software developers. By violating application security measures, hackers are able to publish trade secrets or even redistribute the application with trojan code present. Stealing intellectual property means software vendors are not receiving revenue for the products they developed. To ensure software applications are in the hands of authorized users, software vendors must secure their intellectual property and their revenues using anti-piracy technologies.

SafeNet's security offerings, including [Sentinel Hardware Keys](#) and [Sentinel RMS](#), provide anti-piracy protection and license management solutions for software vendors. Through hardware or software-based solutions, software vendors can protect their applications from unlicensed use and distribution. For additional security, both Sentinel Keys and Sentinel RMS offer the Sentinel Shell to protect intellectual property from sophisticated hacking techniques and to shield the source code quickly and effectively.

The purpose of this document is to demonstrate the benefit of the Sentinel Shell for both hard and soft-based licenses in securing intellectual property and revenues.

SENTINEL SHELL

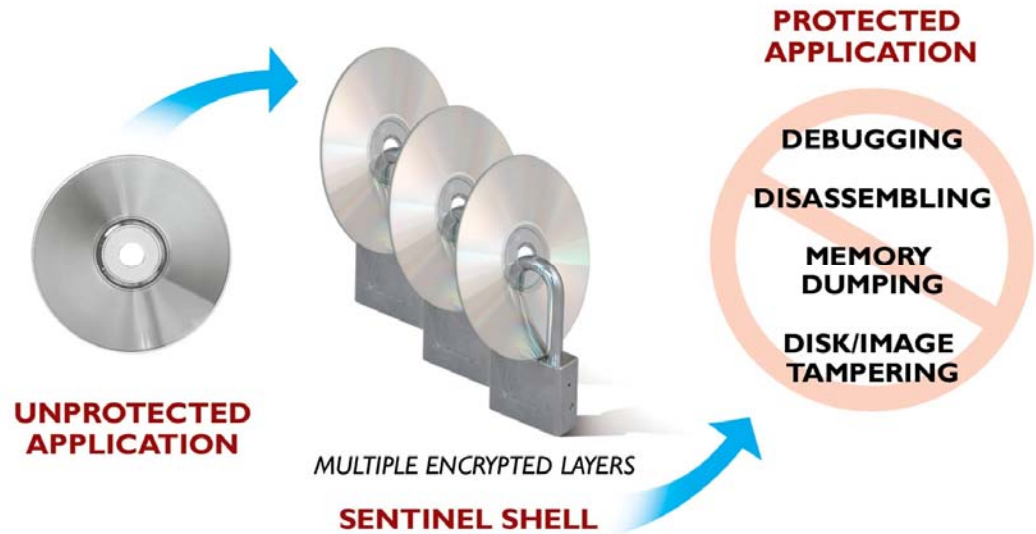
The Sentinel Shell offers easy implementation and superior protection to Windows executables and DLLs. The implementation is quick, automatic and does not require the developer to modify the source code or change the logic of the application.

The Sentinel Shell prevents debugging, static and run-time code analysis, and memory dumping, providing software vendors with advanced protection for their intellectual property.

The Sentinel Shell is most commonly utilized when:

- The software application is already built
- Additional anti-hacking security is required
- Time and abilities are limited to implement a complete, API-level protection scheme

Image 1: The Sentinel Shell Protection – Protecting Applications from Advanced Hacking Techniques



SENTINEL SHELL PROTECTION FEATURES

Multi-Layer Protection

The Sentinel Shell has a multi-layer architecture, which provides extra protection to the application, similar to what multiple locks provide to a door. Each layer implements a different security measure, with a correlating check to ensure the application is executed normally. The previous layer, if executed successfully, will only decrypt the successive layer. Breaking this layer requires additional resources, time, and skill that could potentially deter hacking attacks. Further, due to the random pattern of the layers, there are no protected binaries that are the same—neither in the disk, nor in the memory.

Multi-layer protection ensures that the internal code process and the algorithms within the software applications are securely protected from various anti-reversing techniques. Enabling multi-layer protection has the following benefits:

- The ability to harden and protect the software application from anti-reversing techniques, such as memory dumping - one of the most commonly used techniques hackers use to remove protection from the software application
- Each shelled application is uniquely and randomly protected, reducing the piracy rate by preventing hackers from learning how the protection works and creating a generic hack for the software application



File Integrity Protection

To ensure content integrity, a checksum redundancy check should be performed to identify any possible errors or corrupt data. The Sentinel Shell is able to compute a unique checksum value for the protected application, which is authenticated at runtime. Any tampering or modification of the application code on the disk can be detected, thus preventing the application from running.

Redundancy checking provides the following benefits toward ensuring file integrity:

- Anti-tampering protection to ensure authenticity and security of the application
- Protection from viruses and Trojan attacks, which anti-virus software applications fail to detect

Memory Image Integrity Protection

The Sentinel Shell maintains several checksum values for its multiple layers, as well as the original application image in memory, to ensure protection from code injection attacks. Any changes to the authenticity of the image will be detected and the protected application will fail to execute.

The Sentinel Shell uses a set of security functions for randomly encrypting and decrypting application routines. Benefits of memory image integrity protection include:

- Integrity of runtime application and authenticity from various malicious attacks, such as code injection
- Safeguarding runtime code by preventing anti-debugging tools to attach and decouple the protection from the application

Anti-disassembling Protection

The Sentinel Shell uses various security measures to thwart and prevent disassembling tools such as IDAPro and SoftIce. Unlike various software protection tools on the market that only provide simple encryption wrappers, the Sentinel Shell ensures protection from static analysis of the software application.

Enabling this feature has the following benefits:

- Protection from reverse engineering tools, which disassemble and allow access to intellectual property residing in the software application
- Ensures safe execution and tamper protection to thwart hackers from embedding malware or malicious code into the software application



Anti-debugger Protection

A debugger is a software application that allows the user to break in and single-step through the program execution. It also enables the user to stop the program execution, inject or change the execution path, and dump the memory content to a file. The Sentinel Shell is capable of detecting various debugger tools, such as SoftICE and OllyDbg. It can also provide strong protection from any break interruption and deviation of the program execution.

Enabling this feature has the following benefits:

- Ensures continuous, safe execution and detects any deviation from the normal program execution
- Provides a proactive approach for preventing and detecting various hacking attempts, such as changing, inserting, or performing any technique to capture the program runtime
- Prevents various debugger tools from analyzing the protected application at runtime

Memory Dump Protection


Each Sentinel Shell layer implements a security measure and performs a specific task. During application runtime, only one layer is decrypted at a time. When attempting to dump the memory of the protected application, the dumped image will not work. This is because the rest of the layers, as well as the program, are still encrypted.

In addition, the Sentinel Shell is able to encrypt import functions of the application. This ensures that the import table for the original application cannot be reconstructed after the memory dump.

The key benefit of this feature is the ability to prevent hackers from locating and defeating security measures by re-assembling the decrypted memory content and saving it to another file. The Sentinel Shell can prevent memory dumping by automatically locating and decrypting only a subset of the application's functions during runtime, thus providing a strong defense against any reverse engineering.

Binding Application with Sentinel Shell Engine

In addition to securing software applications without modifications to the source code, the Sentinel Shell provides a Shell software development kit (SDK) to allow the developer to tightly integrate with and provide additional security measures to the protected application. The Sentinel Shell SDK allows the developer to add runtime calls directly to the Sentinel Shell Engine to decrypt and encrypt important code fragments, constant data type, and strings that require additional security. The Sentinel Shell SDK provides an intuitive Application Programming Interface (API) that the developer can use for various functions throughout the application that require additional protection.



During program execution, the Sentinel Shell Engine will only attempt to decrypt and encrypt in various locations as defined by the developer. If the Sentinel Shell protection is bypassed or removed by the hacker, the encrypted code fragments, strings, and constant data will remain encrypted, and the protected application will fail to execute.

Note: Using the Sentinel Shell SDK requires modification to the original application source code. This feature is available only for Microsoft Visual C/C++, Microsoft Visual Basic, and Borland Delphi compilers.

External Data File Protection

The Sentinel Shell has additional capabilities to protect external data files used by the protected application. Any external files that are defined in Sentinel Shell will be encrypted and securely protected, and only the protected application will be able to read and write to these external data files.

The main benefit of extending the protection to external data files is the ability to provide a one-to-one mapping scheme, so that only the protected application can decrypt the required external data file. Without the correct protected application, the required external data files cannot be used. This ensures data integrity and prevents hackers from replacing external data files, which can be harmful to the software application.

.NET APPLICATION PROTECTION

Software programs written for the Common Language Runtime (CLR) are easier to reverse engineer. For such applications, the Sentinel Shell protects the application through the use of code obfuscation, the process of scrambling code and data to thwart a hacker from reverse engineering the application. The Sentinel Shell uses various security measures to protect and hide critical information, which makes it more difficult for hackers to defeat the protection scheme.

A basic introduction to compiled code in a .NET application does not contain native machine instructions. Instead, it manages assemblies or code (Microsoft Intermediate Language – MSIL) that the CLR uses to interpret and execute on behalf of the application. This makes reverse engineering simple and straightforward, enabling a hacker with the right tools to defeat the protection measures.

The Sentinel Shell uses advanced techniques, as mentioned above, to protect .NET applications. (Exceptions include binding the application to the Sentinel Shell Engine and external data file protection.)

The Sentinel Shell is capable of providing enhanced security for managed .NET applications by:

- Hiding the original entry point method, requiring new classes (with types and methods) added into MSIL assemblies
- Encrypting strings in MSIL assemblies using standard AES 128-bit algorithms
- Encrypting constants (32-bit and 64-bit integers) in MSIL assemblies using standard AES 128-bit algorithms

SUPPORTED COMPILERS

The Sentinel Shell can protect applications built with the following compilers:

Compiler/Tool	Version
Microsoft Visual C++	5.0, 6.0, 7.0, 7.1, 8.0
Microsoft Visual Basic	5.0, 6.0
Microsoft Visual FoxPro	5.0, 6.0, 7.0, 8.0, 9.0
Borland C++ Builder	6.0, v 2006
Borland Delphi	7.0, v2006
Power Builder	6.0, 7.0, 8.0, 9.0, 10.0, 10.5
Director	5.0, 6.0, 8.0, 8.5, 9.0, 10.1, MX 2004
Microsoft Visual Basic .NET	7.0, 7.1 and 8.0 with .NET Framework version 1.0, 1.1, 2.0 and 3.0.
Microsoft C#	7.0, 7.1 and 8.0 with .NET Framework version 1.0, 1.1, 2.0 and 3.0.
Delphi .NET	v2006
Borland C#	v2006
MFC	6.0, 7.0, 7.1, 8.0
WinDev	11
LabView	7.1
Authorware	6.0, 7.0

PLATFORM SUPPORT

The Sentinel Shell supports the following Windows platforms:

- Windows Me
- Windows 98
- Windows 2000, 2000 Server, 2000 Advanced Server
- Windows XP 32-bit
- Windows XP 64-bit on AMD64, EM64T
- Windows 2003 Server, 2003 Advanced Server 32-bit
- Windows 2003 Server, 2003 Advanced Server 64-bit on AMD64, EM64T
- Windows Vista 32-bit
- Windows Vista 64-bit on AMD64, EM64T



SENTINEL SHELL LIMITATIONS

The Sentinel Shell has the following limitations:

- Cannot protect memory-mapped files
- Cannot protect executables or DLLs utilizing Thread Local Storage (TLS)
- As the Sentinel Shell adds additional layers of protection to the application, there is an increase in the size of the protected application, varying from approximately 1.5MB to 4.5MB, depending on the level of protection selected. This also results in a slight increase in the load time of the application.

SENTINEL SHELL IMPLEMENTATIONS

The Sentinel Shell protection can be implemented with or without hardware and software licensing. With licensing, the application execution depends on the presence of either software or hardware-based licensing. A software-based license is generated using the Sentinel RMS Development Kit and a hardware-based license is generated using the Sentinel Hardware Keys (SHK) toolkit. Without licensing, the Sentinel Shell implementation works as a code obfuscator for the application—protecting it against the debuggers, memory dumps, and code analysis.

CONCLUSION

See <http://www.safenet-inc.com/sentinel> for more information about these products.