COMMSCOPE®

CipherKnight™: Revolutionizing Application Security with White-Box Cryptography

Rafie Shamsaasef
Aaron Woods

August 22, 2024

Table of Contents

Introduction

In today's increasingly connected and digitally driven world, securing sensitive data within software applications has never been more critical. Traditional security measures often rely on hardware security modules (HSMs), operating system protections, or network security protocols. However, these approaches may not always be viable, especially in untrusted environments where software is vulnerable to attacks. Enter CipherKnight™, a cross-platform white-box security toolchain designed to protect cryptographic keys, certificates, and other sensitive data in applications, even when running in insecure or untrusted environments.

This white paper has outlined the protection needs, features, benefits, and use cases of CipherKnight™, demonstrating how it can play a crucial role in safeguarding applications across a wide range of industries.

What is CipherKnight™?

CipherKnight™ is a sophisticated white-box security toolchain that protects cryptographic keys and other sensitive data while implementing standard crypto algorithms. Unlike traditional security methods, CipherKnight™ does not depend on hardware security, operating system security, or network security. Instead, it integrates protection directly into the application, ensuring that sensitive data remains secure, even if the surrounding environment is compromised. CipherKnight™ white-box implementation represents a significant advancement in application security, providing a powerful solution for protecting sensitive data in environments where traditional security measures may not be sufficient. By embedding security directly into the application code, CipherKnight™ ensures that critical information remains secure, regardless of the surrounding environment.

Understanding the Need for White-Box Technology

White-box cryptography is a technique that fundamentally alters a program to ensure that it operates directly on encrypted and encoded secrets, without ever revealing these secrets in cleartext form. This approach means that even if an attacker has full visibility and control over the application, they cannot extract any meaningful information from it. By keeping the secrets always concealed, white-box cryptography protects sensitive data even in hostile environments. White-box cryptography implementations can be either static or dynamic. Static white-boxes have fixed secrets that are embedded within the application during the build process, making them resistant to extraction. On the other hand, dynamic white-boxes are more flexible—they can accept encoded secrets during runtime, allowing for a more adaptable security approach.

In most cases, White-box cryptography is implemented to protect cryptographic implementations in various applications running on open devices such as smartphones, PCs, and tablets when the developer needs to achieve the highest level of security without relying on elements of secure hardware or operating system being available. These applications store and process private and confidential data and can benefit greatly from white-box cryptography implementations over traditional methods with have well known attack surfaces. Data privacy is an integral aspect of application management and development, leveraging white-box based implementations to protect critical data totally mitigates the risk associated to the loss of that data.

CipherKnight™ white-box cryptography is table-based transformations. This technique involves combining three random bijections with the application’s functions. These combinations are then expressed as lookup tables (LUTs) of size 256x256, effectively concealing the underlying secrets and other sensitive state information. By using these tables, white-box cryptography ensures that the implementation remains secure and resistant to reverse engineering and side-channel attacks even when under direct observation.

Traditional security mechanisms often rely on external factors such as hardware security modules (HSMs), trusted execution environments (TEEs), or network security measures. While effective in some scenarios, these approaches can fall short in

environments where the software is exposed to potential threats, such as mobile devices, IoT devices, and cloud applications. CipherKnight™ White-Box technology addresses these gaps by embedding security directly within the software, making it resilient to attacks even when external protections are compromised.

Key Vulnerabilities and CipherKnight™ Protection Methods

CipherKnight™ is designed to mitigate a range of vulnerabilities that are commonly exploited in modern software applications. One of the primary challenges in application security is ensuring that sensitive data remains protected even when the application is running in an untrusted environment. CipherKnight™ addresses this challenge head-on by embedding protection mechanisms directly into the application code. Below is a summary of key vulnerabilities and the protection methods provided by CipherKnight™:

Vulnerability	Protection Methods
Insecure execution environment	Implement a secure white-box security context
Weak/low platform security or lack of HSM/TEE	Establish a software root-of-trust with a node-locked white-box
Insecure handling of highly sensitive data	Encrypt or white-box encode valuable data and keys at all times
Black-box cryptographic operations that expose keys and other sensitive data	Utilize corresponding white-box cryptographic operations with encrypted or encoded keys and data
Unauthorized access privileges	Move access control to the white-box domain
Bypass of licensing and other controls	Use of white-box based licensing system, such as KnightLicense
Credential extraction vulnerability	Utilize node-locked white-boxes to store and manage credentials
Insecure communication paths between applications or components	Utilize white-box chaining to communicate sensitive data

Benefits of Using CipherKnight™

- **Enhanced Security:** CipherKnight™ offers superior protection for sensitive data, even in untrusted environments.
- **Cross-Platform Support:** Developers can integrate CipherKnight™ white-box source code across various platforms without sacrificing security.
- **Scalability:** CipherKnight™ can be deployed in applications of any size, from small mobile apps to large enterprise systems. The size and speed optimization allows for flexibility needed to balance security vs performance.
- **Ease of Integration:** CipherKnight™ is designed to be easily integrated into existing applications, minimizing development overhead.

Key features of CipherKnight™

- **Cross-Platform Compatibility:** CipherKnight™ white-box primitives are generated as native C/C++ source code and can be integrated into applications across different platforms, ensuring consistent security regardless of the underlying operating system or hardware.
- **Data Locking:** Sensitive key and data are encoded and can be locked to specific devices or subsystems, preventing unauthorized access and ensuring that the data can only be used in its intended environment.
- **Algorithm Protection:** Table-based white-box transformations compose random bijections with an application's functions. These compositions are emitted as lookup tables (LUTs) to conceal the underlying secrets and other state values in the white-box implementation.
- **High-Level Security:** By embedding security directly within the application, CipherKnight™ provides a high level of protection against various threats, including reverse engineering and tampering.

Supported Cryptographic Algorithms

CipherKnight™ supports a wide range of cryptographic algorithms, ensuring robust protection for various applications:

White-box Algorithm	Description
AES 128, 256	• Modes: CBC, CCM, CMAC, CTR, ECB, GCM • Functions: Encryption, Decryption, Key Generation
CSPRNG	• Functions: Secure random number generation
DH 2048	• Functions: Key Agreement, Key Generation
ECC 256, 384, 521	• Functions: Sign, Verify, Key Agreement, Key Generation
RSA 2048, 4096	• Functions: Encrypt, Decrypt, Sign, Verify, Key Agreement, OAEP, Key Generation
SHA 256, 384, 512	• Functions: Hash, HMAC, HKDF, PRF
TRNG	• Functions: Generation of true random numbers

White-Box Chaining

In cryptography, the secure management and transition of cryptographic keys are paramount. CipherKnight™ addresses this need by establishing a robust key ladder mechanism within its white-box environment. This key ladder not only enhances security but also allows the integration of various salts or sources of entropy into the key, further strengthening its resistance against attacks. By injecting randomness and unpredictability at every step, CipherKnight™ ensures that each key derived in the process is unique and highly secure.

A critical aspect of this architecture is its ability to transition between keys seamlessly within the white-box chain while keeping intermediate keys completely concealed. This is essential for maintaining the integrity and confidentiality of the cryptographic process, particularly when executing complex operations like SSL/TLS handshakes. For example, CipherKnight™ enables the chaining of multiple cryptographic algorithms, such as RSA or ECC ciphers with Diffie-Hellman (DH), Digital

Signature Algorithm (DSA), Secure Hash Algorithm (SHA), and Advanced Encryption Standard (AES), all while ensuring that intermediate keys or sensitive data are never exposed.

The flexibility of CipherKnight™ is further demonstrated by its support for various key mixture combinations. Whether combining RSA with Diffie-Hellman (RSA-DH) or using Elliptic Curve Diffie-Hellman (ECDH) to derive a shared secret, CipherKnight™ allows users to customize their cryptographic processes to meet specific security requirements. Once a shared secret is derived, it can be used to generate an AES key, which is then employed to encrypt communications securely. This capability not only enhances security but also allows for the creation of complex, multi-layered encryption strategies that are resilient against sophisticated attacks.

White-Box Node-Locking

White-box Node-Locking is a critical feature of CipherKnight™ technology, providing a robust mechanism to tie the white-box implementation to a specific, secure node. This approach significantly reduces the risk of code-lifting attacks and ensures that sensitive cryptographic operations and data remain protected, even in the event of an attempted unauthorized migration. Node-Locking restricts the operation of a white-box implementation to a specific node, thereby preventing its use on any unauthorized systems. A node, in this context, is not limited to a single hardware device. It can refer to a variety of environments, including:

- **Hardware Devices:** Physical computing devices such as desktops, laptops, servers, or mobile devices.
- **Containerized Environments:** Isolated environments created using container technologies like Docker, ensuring that the white-box implementation runs only within a specific container setup.
- **Virtual Machine Instances:** Virtualized environments, where the white-box is locked to a specific virtual machine configuration.
- **Combinations of Application, Customer, and End-User Identifiers:** The node can also be defined by combining specific software or application identifiers, customer-specific elements, and end-user data to create a unique operating environment.

Node Fingerprinting

The developer configures the definition of a node based on selected data elements that together create the node's "fingerprint." This fingerprint is a unique identifier for the node and is crucial in ensuring that the white-box implementation functions only in the intended environment. The fingerprint could include:

- **Hardware Identifiers:** Such as CPU IDs, MAC addresses, or storage device serial numbers.
- **Software Identifiers:** Including OS version, installed application IDs, or specific environment variables.
- **User or Customer Data:** Information such as user credentials, customer IDs, or other personalized data that can help uniquely identify a node.

By leveraging this fingerprint, CipherKnight™ ensures that even if the white-box implementation is copied and attempted to be run on a different node, it will not function. This capability is essential in mitigating code-lifting attacks, where unauthorized entities try to run the software outside of the controlled, secure environment it was designed for.

True Random Number Generator

CipherKnight™ implements a true random number generator (TRNG) within its white-box cryptographic environment, utilizing advanced mixing functions rooted in "shapeless" quasigroup algebras. These algebras are composed of operations that are non-commutative, non-associative, and non-linear, which significantly enhance the unpredictability and randomness of the generated numbers. By leveraging these mathematical properties, CipherKnight™ ensures that the random numbers produced

are highly resistant to prediction or reverse engineering, providing a robust foundation for secure cryptographic processes. This innovative approach to TRNG design within the white-box framework further strengthens the security and integrity of the overall cryptographic system while compliant to NIST TRNG requirements.

Custom Key Derivation Function (KDF)

CipherKnight™ offers support for custom key derivation functions (KDFs) within a white-box cryptographic framework. This feature allows users to incorporate their own unique “secret sauce” into the KDF, enhancing the security and flexibility of the white-box ciphers. By leveraging the chaining capabilities of CipherKnight, users can construct key ladders that integrate various ciphers, all while ensuring that no intermediate keys are exposed during the process. This powerful combination of white-box KDF and chaining technology enables the creation of complex, multi-layered cryptographic systems with enhanced protection against key extraction and reverse engineering.

Use Cases for CipherKnight™

This section provides a practical overview of how white-box technology can be applied to address common security challenges, making it easier for readers to understand the real-world benefits of the solution.

Generic Application Environment

- **Mobile Applications:** In the mobile landscape, applications often run on devices that are outside the control of the developer, making them susceptible to a wide range of attacks. CipherKnight™ ensures that sensitive data within these applications remains secure, regardless of the device's security posture.
- **IoT Device Applications:** Internet of Things (IoT) devices frequently operate in environments where physical and network security cannot be guaranteed. CipherKnight™ secures sensitive data within these devices, protecting them from potential threats.
- **Cloud-Based Applications:** Cloud environments, while offering scalability and convenience, can also introduce security vulnerabilities. CipherKnight™ ensures that sensitive data in cloud-based applications is protected, even when traditional security measures are insufficient.

Emerging Application Examples

The following emerging application examples highlight the critical importance and wide applicability of white-box solutions across various markets. These examples demonstrate how white-box solutions are increasingly vital across various markets, providing enhanced security and protection in scenarios where traditional methods may fall short.

1. **IoT Chip and Application Security** – With the widespread availability of System on Chip (SoC) technology and accessible manufacturing processes, the integration of white-box implementations has become a viable alternative to hardware security modules (HSMs). This approach allows developers to achieve robust security in their IoT designs without the need to include an HSM in the bill of materials (BOM). As a result, the barrier to entry for innovative IoT solutions is significantly lowered, enabling greater flexibility and cost efficiency without compromising on security.

2. **Contactless Payments with NFC** - Today, many mobile payment applications utilize Near Field Communication (NFC) technology to transform standard smartphones into contactless payment terminals. This capability can be particularly beneficial for companies with limited resources, as it eliminates the need to invest in specialized point-of-sale systems. However, security remains a primary concern in these implementations. Full software-based white-box solutions offer an effective means of achieving the necessary security while maintaining low power consumption, all without the need for dedicated hardware security modules (HSMs).

3. **Medical Applications** - Most data on medical devices is encrypted and transmitted using strong encryption protocols. Additionally, this medical data is often signed to ensure its integrity. Typically, the encryption keys are securely stored within the medical device and on cloud servers. However, applications or programs running on smartphones or desktop PCs represent the weakest links in terms of security. White-box cryptography addresses this vulnerability by securing both the decryption and signing processes, ensuring that medical data and records are protected against theft or manipulation by attackers. Moreover, new micro medical IoT devices, which operate under tight resource constraints, can benefit from software-based white-box implementations. These solutions provide the necessary cryptographic security without the need for hardware security modules (HSMs), making them ideal for resource-limited environments.

4. **Securing the Software Supply Chain** – Whether in a CI/CD pipeline or on the floor of a chip manufacturing plant, a white-box security implementation plays a crucial role in detecting and preventing tampering with critical intellectual property (IP) or software deployments. This is achieved by securing access to sensitive assets and, in cases where physical tampering attempts are made, by detecting changes in binaries or signed code.
   In the CI/CD pipeline, white-box security can be integrated to protect the integrity of software deployments. By embedding cryptographic keys and sensitive operations within the software in a way that they are obfuscated and never exposed in plaintext, white-box security ensures that even if the build environment is compromised, attackers cannot easily extract or manipulate critical information. On the manufacturing floor, where physical access to hardware and software is more feasible, white-box security can prevent tampering by making it extremely difficult to alter the protected crypto algorithms and secrets embedded in software.
   In both environments, white-box security acts as a robust defense mechanism, not only preventing unauthorized access but also providing a critical layer of detection for any attempts to tamper with the protected assets. This ensures that the integrity and security of critical IP and software deployments are maintained throughout the entire lifecycle, from development to production.

5. **Secure Digital Signatures for Identity Theft** - Normally, digital signatures are used for security purposes as they facilitate undeniable user consent even for remote entity authentication. Traditionally, they have been employed to ensure that the identity of the signer is verified and that the signed content has not been altered. Recently, digital signatures have also enabled software-only solutions for tasks such as remote access control and contract signing. However, as these use cases expand, the risk of identity theft and unauthorized sharing of access rights becomes a significant concern.
   By adopting a white-box approach to digital signatures, the security of all parties involved is substantially enhanced. White-box cryptography ensures that the cryptographic keys and processes involved in generating digital signatures are protected within the software itself. This means that even if an attacker gains access to the device or application, they cannot easily extract the keys or manipulate the signing process. Consequently, white-box security prevents identity theft by safeguarding the signer's credentials, and it also deters the voluntary sharing of access rights, as the cryptographic operations are securely encapsulated within the software.

6. **Streaming Platforms** - The rapid rise of streaming services, often referred to as above-ground video services, has presented a significant challenge for those responsible for protecting video content from hackers while simultaneously ensuring that legitimate users enjoy seamless access and a user-friendly experience. This challenge is particularly relevant for both the applications and streaming devices utilized by service providers to deliver content to their subscribers. Hackers constantly attempt to intercept, duplicate, or redistribute premium content without paying, which can result in substantial financial losses for service providers. To counter these threats, robust white-box cryptography security measures are required that can protect the content and associated keys at every stage of its delivery while also maintaining the integrity and performance of the service.

For providers, the goal is to implement security strategies that are invisible to the user but effective in preventing unauthorized access. This includes the use of encryption, secure key management, and other technologies that safeguard the content from the point of origin to the moment it is viewed by the subscriber often in unsecure devices. CipherKnight™ white-box technology offers device and platform independent protection required for such environment.

Use Cases

CipherKnight™ white-box technology addresses several critical security challenges across various scenarios. Below are some specific use cases that highlight the effectiveness of the technology regardless of the specific application or target environment:

1. **Insecure Execution Environment**
    - **Challenge**: Running sensitive operations in an insecure environment can expose critical data and algorithms to attacks.
    - **Solution**: Implementing a secure white-box security context ensures that sensitive operations remain protected even in potentially compromised environments.
2. **Weak/Low Platform Security or Lack of HSM/TEE**
    - **Challenge**: Platforms lacking Hardware Security Modules (HSM), or Trusted Execution Environments (TEE) can leave cryptographic operations vulnerable to attacks.
    - **Solution**: Establishing a software root-of-trust using a node-locked white-box ensures that cryptographic operations are protected without relying on hardware-based security measures.
3. **Insecure Handling of Highly Sensitive Data**
    - **Challenge**: Sensitive data that is not securely handled can be exposed during processing or storage.
    - **Solution**: By ensuring that all valuable data and keys are encrypted or white-box encoded at all times, CipherKnight™ technology protects sensitive information from unauthorized access.
4. **Broken Cryptographic Operations**
    - **Challenge**: Cryptographic operations that expose keys or other sensitive data can be vulnerable to attacks.
    - **Solution**: Utilizing corresponding white-box cryptographic operations with encrypted or encoded keys and data ensures that these operations are secure, even if the cryptographic operation itself is exposed.
5. **Unauthorized Access Privileges**
    - **Challenge**: Unauthorized access to sensitive data or operations can lead to significant security breaches.
    - **Solution**: Moving access control to the white-box domain ensures that only authorized entities can access or perform sensitive operations.
6. **Bypass of Licensing and Other Controls**
    - **Challenge**: Attackers may attempt to bypass licensing controls or other restrictions to exploit software without authorization.
    - **Solution**: Utilizing a white-box based licensing system, such as KnightLicense, ensures that licensing controls are enforced, even in potentially compromised environments.

7. **Credentials Extraction Vulnerability**
    - **Challenge**: Credentials that are not securely stored or managed can be extracted by attackers, leading to unauthorized access.
    - **Solution**: Storing and managing credentials within node-locked white-boxes mitigates the risk of credential extraction, ensuring that sensitive credentials remain secure.
8. **Unsecure Cryptographic Algorithms**
    - **Challenge**: The use of weak or outdated cryptographic algorithms can compromise the security of sensitive operations.
    - **Solution**: CipherKnight™ white-box technology supports modern, secure cryptographic algorithms such as AES, RSA, and ECC, ensuring that cryptographic operations are both secure and compliant with current standards.

Conclusion

CipherKnight™ offers a sophisticated solution to these challenges through its White-Box technology, which embeds security directly into the software, ensuring that sensitive data is protected regardless of the operating environment. CipherKnight™ offers a robust solution by securing sensitive information directly within the application, eliminating the need to rely on external security measures.

CommScope CipherKnight™ white-box suite enables organization to protect keys, algorithms and secrets in unsecure and untrusted execution environment.