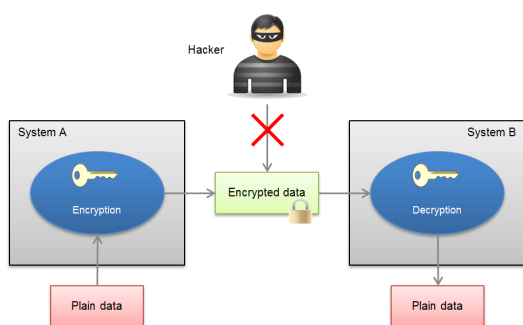


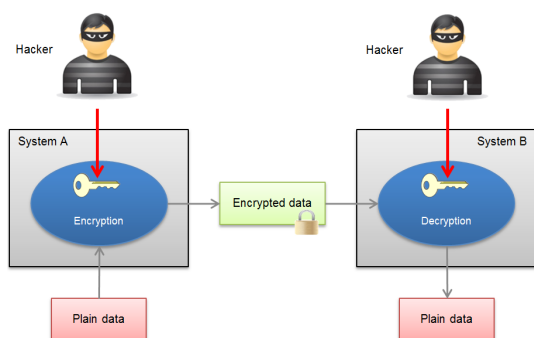
White-Box Cryptography Overview

Why We Need White-Box Cryptography

Security applications protect sensitive information by using cryptographic keys, which need to be available to the program code that decrypts data. Modern cryptographic algorithms are designed so that an encrypted message can be considered safe while it is travelling between the endpoints at which it is encrypted and decrypted.



However, commonly-used cryptographic algorithms were not designed to operate in an environment in which their execution could be observed. The crucial points are the endpoints of cryptographic communication, where a message is encrypted, signed, or decrypted, because the secret cryptographic keys are required for each such operation. If an adversary has access to the device on which a message is cryptographically processed, and if the cryptographic keys are not sufficiently protected, he can extract these keys. This can invalidate the whole security system.



Most hardware platforms, including personal computers, mobile phones, and embedded systems, provide an insecure execution environment for cryptographic operations. Adversaries can monitor the program

code and memory of such devices via special tools, and gain access to the secret cryptographic keys, which are usually internally revealed at some point during processing. White-box cryptography has emerged to specifically address the problem of untrusted endpoints in cryptographic communications.

White-Box Cryptography Overview

The term “white-box cryptography” describes a secure implementation of cryptographic algorithms in an execution environment, such as on a desktop computer or a mobile device, that is fully observable and modifiable by an attacker. It is different from black-box cryptography, where an algorithm’s internal processing data is unavailable to an attacker. The white-box environment puts hard additional restrictions on implementations of the cryptographic algorithms.

Who Is It Intended For?

White-box cryptography is intended for any security system that employs cryptographic algorithms and keys, and that is executed in an open and untrusted environment, such as on a desktop computer, mobile device, or embedded system. Examples of such systems are Digital Rights Management clients, Conditional Access Systems, game consoles, and set-top boxes.

Requirements for White-Box Cryptography

To be secure, practical, and cost-effective, white-box cryptography implementations need to satisfy the following objectives:

- Cryptographic keys must always be encrypted in the program code and device memory.
- Having full access to the cryptographic implementation should not present any advantage when compared to a black-box implementation.
- The performance cost must be reasonable in order for the implementation to be practical.

whiteCryption Technology

whiteCryption specializes in white-box cryptography solutions, and has over 20 years of experience in software security. It has developed several unique proprietary techniques that have been successfully deployed and used. The fundamental technology behind whiteCryption's white-box cryptography solutions is based on a combination of sophisticated mathematical techniques. The primary technique used is Multi-Channel Finite Automata Code Transformation (MCFACT).

MCFACT™

MCFACT is a method of protecting secure data and the sensitive areas of the program code (such as the cryptographic algorithms) by transforming them into composite finite-state automata. Program code that is transformed into such automata is able to perform computations on encrypted data without the data ever being decrypted.

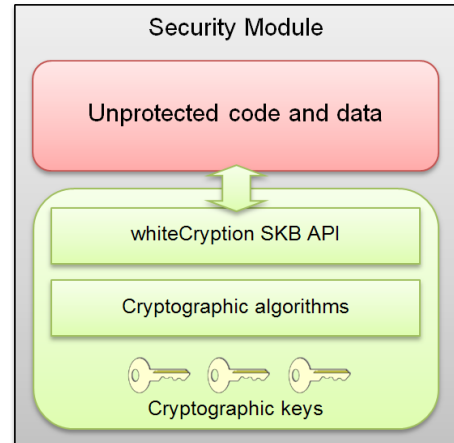
MCFACT is based on the proposition that once encryption, decryption, and computation finite automata are composed into a single finite automaton, the internal secrets of the computation are effectively hidden. The composition of the finite automata ensures that no intermediate plain data or information about the operation can be retrieved. An adversary cannot easily reverse engineer the protected parts of the binary program code, because a reverse transformation to program code would involve the decomposition of large finite automata, which is a near impossible mathematical problem.

whiteCryption Secure Key Box (SKB)

whiteCryption SKB is whiteCryption's flagship product that is based on the company's unique technologies, including MCFACT. It is a cryptographic library that provides secure white-box cryptography implementations of the most commonly-used algorithms, including AES, RSA, ECC, ECDSA, and SHA.

Systems integrated with whiteCryption SKB can be safely deployed in insecure environments, such as on mobile devices, game consoles, and desktop computers, where anyone could gain access to the program code and device memory.

whiteCryption SKB System Overview



whiteCryption SKB Features

- **Cryptographic keys are always encrypted.** Once keys are imported into whiteCryption SKB, debugging and reverse engineering will not reveal them in plain form. Algorithms operate directly on encrypted keys.
- **Support for industry-standard cryptographic algorithms.** whiteCryption SKB supports the most commonly-used cryptographic algorithms.
- **Implementation design with performance in mind.** whiteCryption SKB is customizable and provides separate implementations of cryptographic algorithms for different purposes and target platforms, considering the computational capabilities and memory available on the target devices.
- **Cross-platform support.** whiteCryption SKB supports a wide range of operating systems and hardware architectures.
- **Safe storage of cryptographic keys.** whiteCryption SKB ensures that cryptographic keys are exported, imported, and stored in a unique encrypted format to prevent adversaries from reading and altering them.
- **Diversified code and data.** By using whiteCryption's Trusted Deployment Service, you can obtain multiple whiteCryption SKB packages with different binary and data implementations, making it even harder to develop a universal tampering scheme.