SafeNet HSM Design Principles

Better Hardware Security Modules Through Better Design

www.safenet-inc.com

Table of Contents

SafeNet Secure Hardware Design Practices

Better Hardware Security Modules Through Better Design

Hardware Security Modules (HSMs) are purpose-built hardware devices designed to store sensitive cryptographic private keys and perform cryptographic operations. HSMs work in conjunction with host applications, for example, a Certificate Authority application that issues digital identity certificates, to secure the application's private keys and cryptographic operations.

The HSM and its host system are two physically separate entities that must communicate with each other via a cable or system bus (e.g. PCI) at the hardware level, while software on the host system invoke the services of the HSM through an Application Programming Interface (API). HSMs use APIs designed specifically for cryptographic operations, such as RSA's PKCS#11, Microsoft's CryptoAPI, Intel's CDSA, Sun's JCE, and Open SSL.

During normal usage, the host application and the HSM exchange requests and data needed to perform cryptographic operations. However, due to the sensitive nature of these operations, special precautions must be taken to prevent compromise of the cryptographic key materials stored within the HSM, through a compromised application or API vulnerability.

Cryptographic API Security Risks

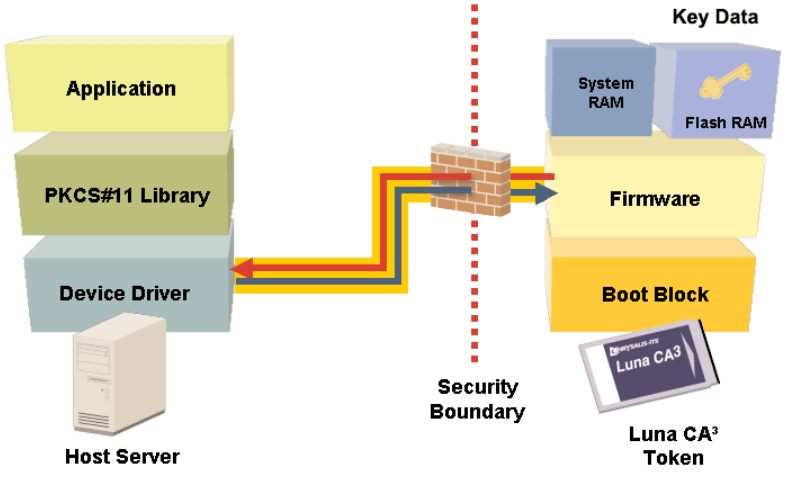
Cryptographic APIs are designed to provide a common interface between diverse system components to simplify design and integration while accommodating a variety of software and hardware components. In order to provide flexibility and compatibility with the largest number of possible implementations, cryptographic APIs feature flexible configurations and a wide variety of options to adapt to the broadest range of application scenarios. However, this flexibility can also create security exposures when a cryptographic API is misconfigured or implemented incorrectly. While the cryptographic API specifies the syntax required to invoke and manipulate cryptographic operations, it is not designed to provide security safeguards to protect against the introduction of insecure design practices; the responsibility to ensure that the API is implemented correctly falls solely on the shoulders of the system architect or software designer who specifies how the API is used in the context of a specific security application. As HSMs are deployed specifically to secure sensitive cryptographic keys and operations, it is of extreme importance that all aspects of the HSM system design, including API implementation, are beyond compromise.

SafeNet HSM Design Philosophy and Practices

SafeNet has established its reputation as the leading HSM vendor by providing rock-solid security hardware; the Luna family of HSM products has been used since 1996 by hundreds of organizations in government, military, financial, and enterprise sectors as the trusted solution to secure PKI applications. Chrysalis has also been an integral contributor to the development of industry standards governing HSM best practices to help increase the security and operation of HSMs, including work on the PCKS#11 API itself. Based on this experience garnered from designing and building hardware security products, coupled with extensive deployment experience with partners and customers, Chrysalis has developed a rigorous set of HSM design best practices designed to maintain the highest levels of security for HSMs.

Security Boundary

To provide the highest levels of assurance and security, Chrysalis HSMs operate independently of their host application, APIs, and supporting hardware. To accomplish this, Chrysalis defines a clearly demarcated Security Boundary that isolates all firmware, software, and memory locations related to cryptographic processing and private key data storage on the HSM from external elements. The areas within the Security Boundary are only accessible through the HSM's firmware to prevent exposure of cryptographic elements to the outside world. All other hardware and software components, including cables, client software, APIs, communications protocols, and authentication devices necessary for the HSM's operation reside outside of the HSM security boundary and cannot directly access protected elements behind the security boundary.



The security boundary prevents components on the host system, which may be comprimised, from gaining direct access to key data and cryptographic operations on the HSM.

All cryptographic processing is performed within the Security Boundary to prevent unauthorized, or illegitimate access. For example, if a Chrysalis HSM was stolen and placed in a new environment for the purpose of gaining illegitimate access to the cryptographic material contained within it, Chrysalis HSMs will override requests from external sources if they contravene the internal security policy settings of the HSM.

This design practice eliminates the possibility that rogue or defective software could be inserted into the host system in an attempt to extract sensitive keys or data from the HSM. The security of a Chrysalis HSM is completely independent of the surrounding environment including the client software that communicates with the HSM.

The autonomous nature of Chrysalis HSMs means that host application servers do not need to be disconnected from the network during HSM configuration or maintenance operations. Chrysalis HSMs do not allow the host application or any other intervening component to compromise the security of the sensitive data that they are designed to protect.

API Restriction

PKCS#11 is the native API for all Chrysalis HSMs. All other APIs supported by Chrysalis are layered on top of this PKCS#11 fundamental architecture. Chrysalis selected PKCS#11 as the basis for our products due to its secure architecture and direct mapping to an HSM's cryptographic services which allows API implementation without ambiguity. Unlike other cryptographic APIs, PKCS#11 has been designed from the ground up as an API intended to support external hardware security modules, whereas other cryptographic APIs were initially developed to support software-based cryptographic processes. This underlying architectural difference provided by PKCS#11 reduces potential exposure, and hence security risk, by ensuring the accuracy and security of the underlying hardware implementation relative to the features and functions of a software-biased API.

Cryptographic APIs contain capabilities that allow key material to be passed back and forth in very liberal ways between the application and the cryptographic library. In fact, all of these APIs allow for the keys to be provided by the application in "plain text" format to the cryptographic library and vice versa. As a result, it is possible that applications using these APIs could attempt to perform direct manipulation of the cryptographic material to expose, tamper with, or steal private keys. In order to protect against such a risk, Chrysalis HSMs restrict API access to prevent exposure of sensitive key materials or cryptographic processes. Chrysalis has applied the expertise gained over years of working with cryptographic APIs and HSM security to establish design practices design criteria to selectively restrict and disable potentially dangerous features offered by cryptographic APIs. These design practices protect against inadvertent or malicious misuse of API calls to HSMs. Although these design decisions may potentially result in additional development effort when integrating with security applications, SafeNet' attention to security detail is meant to protect HSM customers from accidental security holes created by lack of familiarity with cryptographic APIs.

Chrysalis pays particular attention to this issue and sets very high internal design standards for implementing APIs that are fail-safe and do not provide opportunities for inexperienced programmers to misuse the API, and hence the underlying HSM, in an insecure fashion.

Best Practices

As a pioneering developer of HSMs, Chrysalis has been bringing secure HSMs to market since 1994. SafeNet Luna products incorporate features developed through extensive operational experience, implementing Best Practices in hardware, software, and operations that make the deployment of a secure HSMs as easy as possible. Chrysalis not only incorporates rigorous design requirements, but must also pass through stringent product verification testing followed by real-world application testing to verify the security and integrity of every SafeNet product.

To provide an objective yardstick to measure HSM security, in 2001, SafeNet introduced a set of 10 HSM Best Practices that define the recommended hardware, software, and operational best practices necessary to the establish and maintain a secure HSM. These Best Practices represent the core values inherent in every Chrysalis HSM design.

Summary

These design best practices take Chrysalis HSMs beyond the expectations and requirements of certifications such as FIPS 140-2 Level 3 validation and Common Criteria EAL4+ certification.

Chrysalis HSM offer fully autonomous security processing within a well-defined Security Boundary, combined with failsafe overrides, and defaults to protect application developers from creating accidental security holes, yields a level of security in an HSM that is unmatched in the industry.

For more Information Regarding SafeNet Products:

Please Visit http://www.safenet-inc.com or send us an email info@safenet-inc.com.

Beyond FIPS 140-1: Essential Best Practices for Hardware Security Modules
http://www.safenet-inc.com/news/library/white_papers/Beyond FIPS 140-1-best practices.pdf

4690 Millennium Drive, Belcamp, Maryland 21017 USA

Tel: +1 410.931.7500 or 800.533.3958
eMail: info@safenet-inc.com

www.safenet-inc.com