

swissbit®

White Paper

An Introduction to: Security in Storage Systems

© Swissbit AG 2024 – All rights reserved.



1. Abstract

Not a day passes without major news of data breaches. Whether it's database theft or taking control of systems, the interconnectivity the IoT brought with it, has made these threats even more dire.

Ensuring data is secure from unauthorized access during storage and transfer is one of the main challenges of the connected world we live in today. This white paper will give a brief introduction into the main concepts of security in storage systems.

Table of Contents

1. Abstract
2. Introduction
3. Cryptography
4. Security in Storage
5. Conclusion

2. Introduction

In most systems, there are one or more forms of data storage. Securing this data is complex and covers numerous aspects, from legal compliance and system authentication to data protection.

As security is related to keeping communication and data secret from others, cryptography will be a main topic in this paper where symmetric and asymmetric cryptography are explained. Communicating with a large number of parties requires an efficient identification process. This paper explores how Public Key Infrastructure (PKI) addresses and solves this issue. Following this, the elements and devices needed in a NAND flash storage system to ensure the secure storage of the data are presented. These range from random number generators to hardware accelerators for encryption. Finally, there will be an overview of methods that make sure that data once stored on a NAND flash storage system is irretrievably deleted once it's not needed anymore.



3. Cryptography

Cryptography is the research and practice of techniques for secure communication between two parties in the presence of third parties. It is about constructing and analyzing protocols that prevent third parties or the public from reading private messages.

In cryptography, encryption is the process of encoding a message in a way that only authorized parties, like the recipient, can access it and those who are not authorized cannot. Encryption does not itself prevent interference, but prevents a possible interceptor from reading the message.

In an encryption scheme, the message, which is called plaintext, is encrypted using an encryption called a cipher or a key. This generates cipher text that can only be read if decrypted. It is possible to decrypt the message without possessing the key, but, for a well-designed encryption scheme, considerable computational resources and skills are required. A communication protocol is secure if its cryptographic analysis implies so much time and effort that it cannot be executed in practice. For technical reasons, an encryption scheme usually uses a pseudo-random encryption key generated by an algorithm.

Cryptography can be divided into two major branches: symmetric key cryptography and asymmetric cryptography. Both have their advantages and disadvantages.

The main disadvantage of the symmetric key approach is that the sender and the receiver both encrypt and decrypt their messages using the same shared secret key. An important prerequisite for secure symmetrical communication is therefore that the key has already been exchanged in a secure manner, for example by a trustworthy courier or at a direct meeting. The key exchange problem now poses the following problem: For example Alice wants to communicate with Bob, who is far away, using a symmetric encryption method. The two are connected via an insecure channel, e.g. the internet, and have not exchanged a key. So how do Alice and Bob agree on a shared secret key over an insecure channel?

In addition to that, if Alice wants to communicate with another person, a new symmetric key is

needed. Communicating with a large group of people quickly becomes demanding, as a separate key is needed for every additional person.

Asymmetric encryption was introduced to eliminate the inherent problem of the need to share the secret key in a symmetrical encryption model by using a pair of public-private keys. While this is a strong advantage, asymmetric encryption is a lot slower and needs much more computational power than the symmetric encryption. On the other hand, only one set of public and private keys is needed to establish an encrypted communication with any number of people.

Symmetric Key Cryptography

Symmetric encryption is a conventional method of encryption. It is also the simpler of two techniques. Symmetric encryption is executed by means of only one secret key that is possessed by both parties. The plain text (the message) is encrypted to ciphertext using a key and an encryption algorithm. The ciphertext is converted back to plaintext (decrypted) using the same key as was used for the encryption.

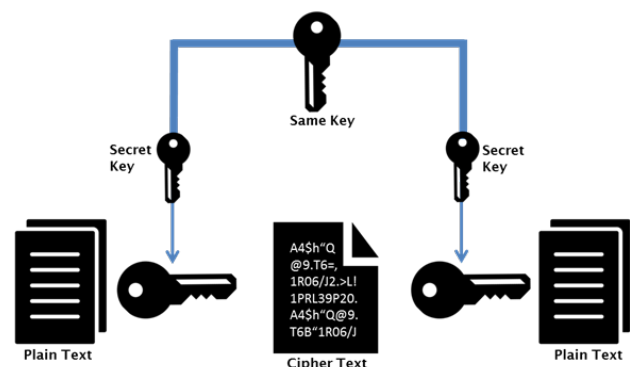


Figure 1: Symmetric Key

Symmetric encryption algorithms are fast and require little computational power. When encrypting large amounts of data, symmetrical encryption is therefore preferred.

Advanced Encryption Standard

The Advanced Encryption Standard (AES) is a specification for the encryption of electronic data established by the U.S. National Institute of Standards and Technology (NIST) in 2001 under the FIPS 197 standard. It is also part of the ISO/IEC 18033-3 standard. AES is a symmetric encryption algorithm. It is extensively used in storage systems such as SSDs.

AES is used by a variety of applications: It is used by the IEEE 802.11i encryption standard for wireless LAN and its Wi-Fi equivalent WPA2. Furthermore, it is used in IP telephony like SRTP and Skype, operating systems like Mac OS X and Microsoft Windows later than XP SP 1 and in software like 7-Zip, RAR and PGP.

The AES algorithm is supported in several CPUs by Intel or AMD by additional specialized machine commands, whereby encryption is performed many times faster than with unspecialized machine commands. This makes AES also suitable for mobile applications and battery-friendly for mass use.

Some programming software libraries such as OpenSSL automatically detect whether the hardware supports AES and then use the hardware AES implementation instead of the slower software implementation.

Asymmetric Key Cryptography

Asymmetric key cryptography utilizes algorithms that are based on the intractability of certain mathematical problems. Regarding the complexity, intractable problems are problems for which no efficient algorithms exist to solve them. Most of these problems have an algorithm that provides a solution, and that algorithm is the brute-force search. Conclusively, these are problems that can be solved in theory but in practice take too long for their solutions to be useful.

With asymmetric encryption it is computationally easy to generate public and private keys, encrypt messages with the public key, and decrypt messages with the private key. However, it is extremely difficult, i.e. it takes a very long time, for anyone to derive the private key based only on the public key.

The idea is that the private key is kept secret and never shared with any other party. The public key is exchanged through the communication channel. There is no need to find complicated ways to hide the public key or to secretly exchange it. It can be sent to everyone. With this public key, the sender can encrypt the message. Afterwards the message is sent through the communication channel to the recipient. Even if a third person intercepted the message and is in possession of the public key, the message cannot be decrypted. Only the intended recipient who is in possession of the private key, which is never shared with anybody else, can decrypt the message.

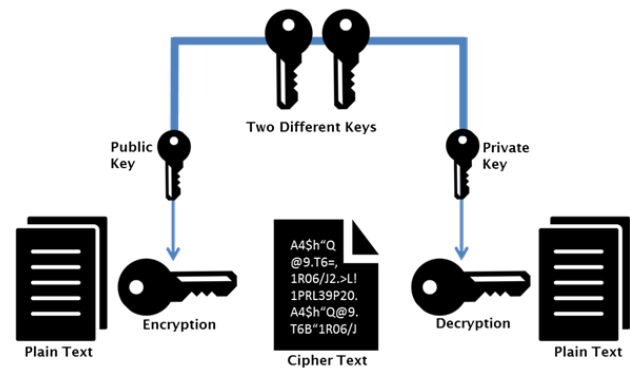


Figure 2: Encryption and decryption with asymmetric keys

One major advantage of asymmetric key cryptography is that for each participant, only one public/private key pair is needed – regardless of how many other participants the first one wants to communicate with.



Key Generation & Key Lengths

A large and random number is used to begin generation of an acceptable pair of keys for an asymmetric key encryption scheme.

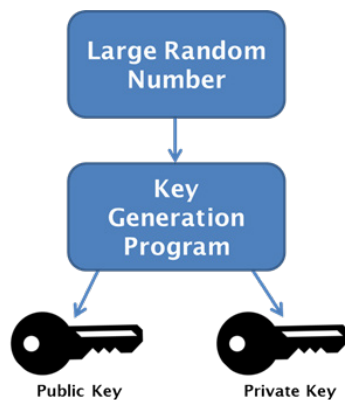


Figure 3: Key generation

Another difference between symmetric and asymmetric encryption is the length of the keys, which are measured in bits and are directly related to the level of security provided by each encryption method.

In symmetric encryption, the keys are randomly selected, and their lengths are usually set at 128 or 256 bits, depending on the required level of security. In asymmetric encryption, however, there must be a mathematical relationship between the public and private keys, meaning that there is a mathematical pattern between the two or that one is derived from the other. As this connection can potentially be exploited by attackers to crack the encryption, asymmetric keys need to be much longer to reach the same level of security as symmetric encryption. The difference in key length is so pronounced that a 2,048-bit asymmetric key is similar in a level of security to 128-bit symmetric key.

Transport Layer Security (TLS) and its predecessor Secure Sockets Layer (SSL), widely used to secure all internet communication between web browsers and servers, uses the asymmetric encryption (public/private key pair) to deliver the shared session key, to finally achieve a communication with symmetric encryption.

Message Signing

In addition to encrypting messages, it may be needed to ensure that a message was not altered during transit over the potentially insecure communication channel and that it actually came from the purported sender. This can be accomplished with a digital signature, which is a technique that binds a person to data sent. The signature is a cryptographic value that is calculated from the data and a secret key known only by the signer. The signature can be independently verified by the receiver as well as a third party.

A hash is calculated over the data to be sent. Cryptographic hash functions are known to be collision-resistant, which means that different input is highly unlikely to produce the same output. Since the hash of data is a unique representation of data, it is sufficient to sign the hash instead of the actual data. The most important reason to use a hash instead of data itself is efficiency – signing long messages requires significant computing power. The hash value and the private key are then used as input for the signature algorithm which produces the digital signature, which is appended to the data. Both are sent to the recipient.

The recipient recalculates the hash of the message and then uses the public key of the sender to verify the signature he received.

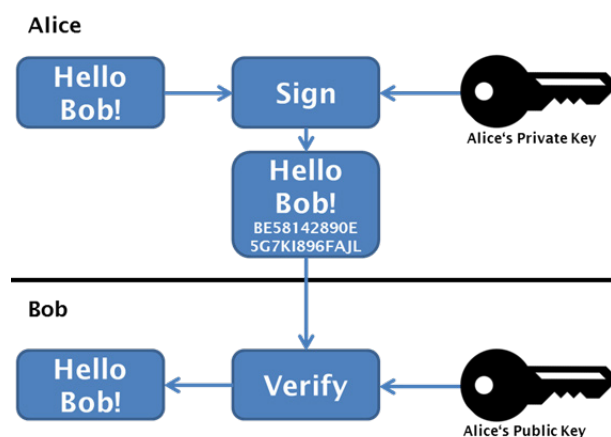


Figure 4: Signing of messages

Generally, the key pairs used for encryption/decryption and signing/verification are different. The private key used for signing is referred to as the signature key and the public key as the verification key.

RSA

RSA (Rivest–Shamir–Adleman) is one of the first public-key cryptosystems. It is widely used for secure data transmission and anyone who uses the Internet is utilizing RSA cryptography in some form. As all asymmetric encryption schemes, RSA uses a public key and a private key. The public key is based on two large prime numbers and an auxiliary value. The prime numbers have to remain secret. RSA's main security foundation relies upon the fact that given two large prime numbers, a composite number can very easily be generated by multiplying the two primes. But, given only the composite number, there is no known algorithm to efficiently determining its prime factors. In fact, it is classified as a NP-hard problem which translates into the fact that it's almost impossible to solve. Anyone can use the public key, the composite number, to encrypt a message, but only someone with knowledge of the private key, the prime numbers, can decode the message.

There are currently no known methods to break the encryption if a large enough key is used. RSA is a relatively slow algorithm, and therefore it is less commonly used to directly encrypt user data. Instead, RSA is often used to exchange keys for symmetric key cryptography.

Key Take-Away: The RSA algorithm contains one operation that is easy to do but difficult to undo. The easy part of the algorithm multiplies two prime numbers while the difficult part is factoring the product of the multiplication into its two component primes.

Diffie–Hellman Key Exchange

The Diffie–Hellman key exchange is a method of securely exchanging cryptographic keys over a public and insecure channel. It is a way of generating a shared secret, a key, between two people in a way that the secret can't be seen or obtained by observing the communication. Even if the traffic is recorded and later analyzed, it is impossible to determine the key, even though the exchanges that created it may have become visible. The key itself is never transmitted.

As in most of the asymmetric key schemes, sender and recipient both possess a pair of public and private keys. Alice would send her public key to Bob and Bob would send his public key to Alice. Alice would then use the public key from Bob and her own private key to generate a key using the Diffie–Hellman algorithm. Bob would also use the very same algorithm as Alice and in turn produce the exact same key as Alice.

As a result, they both now possess an identical key without having exchanged it over the communication channel. This key can then be used for symmetric encryption thus enabling them to communicate securely over the insecure internet or other communication channel.



Public Key Infrastructure

The public key infrastructure (PKI) is the foundation that enables the use of technologies, such as digital signatures and encryption, across large user populations. The purpose of public key infrastructure is to help establish the identity of users, i.e. to reliably tie a public key to a person, service or thing – in case of the internet of things (IoT). Although it is easy to encrypt messages without such infrastructure, it is not easy to verify the identity of the communication partner. In other words, the infrastructure helps to verify that the other person is who they claim to be. This enables controlled access to systems and resources, protection of data, and accountability in transactions.

A typical infrastructure consists of policies, standards, hardware and software that manage the creation, distribution, administration and revocation of digital certificates.

The heart of a public key infrastructure is a certificate authority, which is a trusted entity that ensures the trustworthiness of the digital certificates. Digital certificates are the credentials that facilitate the verification of identities between users. In the infrastructure, the user is negotiating trust with a root which then communicates all the other entities that the user can trust by default. The central idea of public key infrastructure is that some identities that are already trusted by the user can delegate their trust (and hence the trust of the user) to other identities the user doesn't know yet.

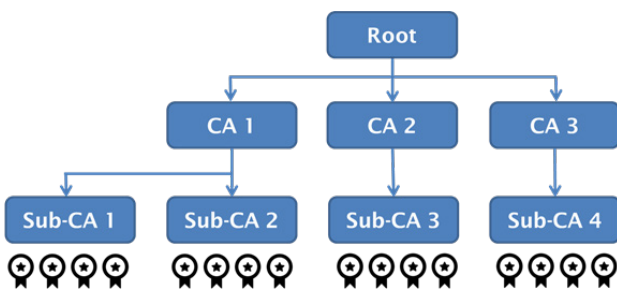


Figure 5: Key Distribution and Certification

Typical elements of a public key infrastructure, as shown in Fig. 5, are: The root certificate authority is an entity that is the “root of trust” and is responsible for authenticating all identities in the ecosystem. The subordinate certificate authority is certified by a root authority. Digital certificates are typically issued and signed by subordinate authority.

These certificates are a digital identity embedded in a device that provides secure authentication between devices allowing access to services, files or other remote resources. The certificate is a file containing information such as identification, a serial number and expiration dates. Most importantly, the certificate contains the certificate holder’s public key and the digital signature of the authority that issued the certificate.

The 4 main functions of PKI are:

- Generation of public/private key pairs
- Management of the certification process
- Certification of public keys and certificate publication
- Certificate revocation

As a result, the public key infrastructure provides security services such as authentication, confidentiality, integrity checking and non-repudiation.

Authentication is the process of verifying that the users, i.e. the sender and recipient, are who they say they are. Confidentiality means that data streams and messages are encrypted that only an intended recipient can read (decrypt) the transferred items. Integrity is achieved by public key infrastructure’s built-in ways to validate that all the outputs are equivalent to the inputs. Any change of the data can be immediately detected and prevented. Non-Repudiation stands for legal non-deniability. It means that a sender of a message cannot refute that he signed or encrypted a particular message once it has been sent, as he is the only one with access to his private key with which the message has been signed or encrypted. This assumes of course that the private key is secured.

The revocation of security certificates is shown in Fig. 6. As depicted, a single certificate can be revoked, a subordinate authority and a whole certification authority. As a result of the revocation of authorities, all certificates issued by this authority are also revoked.

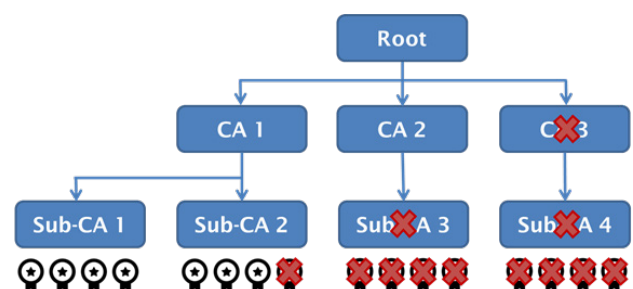


Figure 6: Removing all related certificates after a security breach.

Elliptic Curve Cryptography

The Diffie–Hellman key exchange and the RSA algorithm are based on the creation of keys by using very large prime numbers. This requires a lot of computational power. Elliptic curve cryptography (ECC) is part of the public key cryptography (therefore an asymmetric encryption) based on the elliptic curve theory that can be used to create smaller and more efficient cryptographic keys. It generates keys through the properties of the elliptic curve equation instead of the traditional method of generation as the product of very large prime numbers.

Mathematically, multiplying a point on the curve by a number will produce another point on the curve. This operation is easy to perform and takes very little computational power. Reversing this operation to find what number was used is done by searching for the discrete logarithm of the elliptic curve element with respect to a publicly known base point. It is assumed that this is infeasible, which means an immense amount of computational power would be needed – in the order of magnitudes more than for the factoring problem in the RSA and Diffie–Hellman methods.

As a result, elliptic curve cryptography shows significantly better performance, also with short key lengths: 384 bits are sufficient according to the National Institute of Standards and Technology (NIST) for top–secret encryption in American government agencies, whereas a key size of 7680 bits with RSA would achieve the same level of security.

This is especially relevant, since strong public–key cryptography is often considered to be too computationally expensive for small devices if not accelerated by crypto–graphic hardware. This changes with elliptic curve cryptography as a high level of security can be achieved with lower computing power and battery power.

4. Security in Storage

The last chapter introduced concepts of security during communication. Sending and receiving data during communication at some point also implies storing data in a memory like e.g. a NAND flash memory like an SSD.

Swissbit designs and manufactures secure storage modules and applications for IoT devices. We have taken security requirements into consideration from overall general security that is built into every firmware that comes with a controller up to advanced highly certifiable security applications.

Many of the algorithms and methods like the Advanced Encryption Standard, Elliptic Curve Cryptography and support for Public Key Infrastructure are built into swissbit products. In addition to that, they mostly offer the following hardware and firmware specific to security in storage systems.

Self Encrypting Drive

The Opal specifications were developed by the Trusted Computing Group (TCG). The standards describe a set of specifications for features of SSDs that enhance their security, like an encryption to protect the data on the drive from unauthorized access.

The host into which the device is plugged into doesn't have any overload as all the security functions are executed by the storage device. A number of hardware requirements are described in the specifications.

Opal encompasses these functions:

- Security provider support
- Interface communication protocol
- Cryptographic features
- Authentication
- Table management
- Access control and personalization
- Issuance
- Discovery

The Trusted Computing Group also later released two additional specifications:

Opalite is a subset of OPAL features that guarantee data-at-rest protection, control access to data, has secure boot authentication capability and allow crypto erase for repurposing of the drive.

Pyrite is a subset of OPALITE features that doesn't guarantee data-at-rest protection (no required encryption), but takes care of the access grant.

True Random Number Generator

The basis for cryptography and encryption are keys which are derived from random numbers, as explained in chapter 3.2.1. Therefore, generating random numbers of high quality is an essential prerequisite. A random number generator is a device that generates a sequence of numbers that cannot be predicted better than by random chance.

Random number generators can be true random number generators, which generate sequences of completely random numbers, or pseudo-random number generators, which produce sequences of numbers that look random, but are actually deterministic. The quality of a random number generator is measured by its entropy, which is the randomness that is collected by it.

For cryptography, either true random number generators or cryptographically secure pseudo-random number generators are needed. The latter are designed to have properties that make it suitable for use in cryptography. The random number sequence is often used to generate seeds for key generators to be used in symmetric and asymmetric encryption.

To prove that a random number generator in a system performs well enough for cryptographic use, it can be certified according to a standard like AIS-31, NIST SP 800-90, FIPS 140-2, ISO/IEC 18031. For this, a test mechanism must also be put in place to characterize its entropy according to certification requirements.

Physical Unclonable Function

A physical unclonable function (PUF) uses the physical characteristics of an integrated circuit to create a unique number that can be used e.g. as a device ID. The function usually utilizes minor imperfections in hardware modules produced using the exact same manufacturing process, even on the same wafer. The differences and imperfections do not affect the normal operation of the hardware module, but introduce unique secondary characteristics. A common secondary characteristic utilized by PUFs in integrated circuits is the start-up values of SRAM and DRAM memory. These are in an undefined state when powered up. This undefined value is often the same, thus providing a value that is both random and unique to the chip or device.

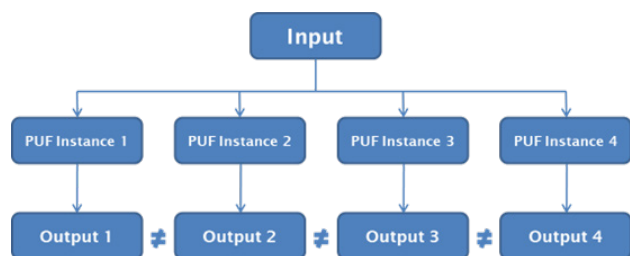


Figure 7: The same input to different PUFs in different chips will provide a different output.

Common applications of PUFs are secure boot, identification, authentication and secure key-exchange protocols. Additionally, PUFs can also serve as the basis for the implementation of true random number generators. A well-known system using PUFs are credit cards.

Secure Hash Algorithm

The Secure Hash Algorithm (SHA) is one of a number of cryptographic hash functions. A hash is a mathematical algorithm that maps data of any size to a bit string of a fixed size called the "hash" or "hash value". Hashing is a so-called one-way function which means that it is practically infeasible to invert.

A cryptographic hash is like a signature for a text or a data file. The SHA-256 algorithm for example generates a fixed size 256-bit (32-byte) hash. This makes it suitable for password validation, challenge hash authentication and digital signatures.

Root of Trust

The Root of Trust defines the security perimeter within a cryptographic system. It enables safe management of keys, safe execution of related firmware within a trusted execution environment. Any cryptographic environment depends on its keys that will be used to perform different actions. The root of trust protects the keys, during their generation or utilization to perform cryptographic functions. Any security certification is relative to the defined root of trust. The root of trust is not accessible by external resources and gives a way for the system to guarantee that the information is valid and has not been hacked. It is also a major element in the security lifecycle management. Sometimes, the term Trusted Platform Module (TPM) is used although in most cases, it is used for higher system levels.

The root of trust can also be used to support a secure boot. A secure boot in the context of a memory controller is to ensure that the right (unmodified) boot program is executed. For a storage system, during pre-format, a boot-firmware is loaded onto the flash memory. It is important to make sure that this firmware is legitimate. A secure boot process is in place to invalidate the boot firmware if it is tampered which makes the drive unusable.

ISO/IEC 7816 interface to secure elements

ISO/IEC7816 is an interface standard essentially used to connect smartcards to a host system. A smartcard can also be used as an external secure element for storage media and can be connected to NAND flash controllers that offer such an interface. In the secure element, keys for encryption can be stored and managed. Using these high-security additions, it is possible to achieve the highest levels of security certifications.

Deletion of Data

While it is important to protect data as long as it is used, it is equally important to delete data as soon as it becomes obsolete or other circumstances occur. For this, there are several methods.

Secure erase: For NAND flash memory secure erase often only deletes the mapping table, which defines where chunks of data are stored, but will not erase all blocks of data that have been written to. Thus, Secure Erase is faster to complete than Sanitize, but prone to possible data partial or full recovery, even though the task should be very difficult to perform.

Sanitize will delete the mapping table and will erase all blocks that have been written to. The drive should be restored to factory default settings. Depending on the flash controller vendor, the implementation of the feature might differ, and different performance can be expected. In some cases, it is even required to rewrite random data to the blocks, one time or multiple time, to ensure that all persistent information is removed.

A cryptographic erase can only be done if the data has been encrypted before being stored on the memory. In this case, only the encryption keys will be erased. Therefore, this process is really fast as very small amount of information, the encryption keys, must be erased. This is generally the erase option used if erase is required on system power down.

Secure TRIM: In NAND flash storage systems, the deletion of files or data on file-system level does not result in a physical erase on flash level. TRIM is a feature that has become meaningful for flash storage systems. A trim command allows an operating system to inform a solid-state drive (SSD) which blocks of data are no longer considered in use and can be wiped internally. However, a regular TRIM command does not trigger an erase of invalid data automatically so that fragments can still be present on the flash memory, which can be a security risk. A secure trim makes sure that obsolete data is actually deleted.

Write Once Read Many

On a "Write Once Read Many" (WORM) drive, data can only be written only once. It ensured that data cannot be changed later or tempered with. This is sometimes a requirement from regulatory institutions, like e.g. in cash registers. However, it can be important for example to implement a secure boot in a WORM drive, to guarantee that the firmware is not tampered to change the characteristics of the drive.

5. Conclusion

In this paper, the basis for understanding security has been explained. The methods, advantages and disadvantages of symmetric and asymmetric encryption standards have been outlined as well as the challenges of key distribution. Of the asymmetric encryption algorithms, the first standard and the RSA encryption have been outlined. Furthermore, the Diffie-Hellmann key-exchange up to the latest and highly efficient Elliptic Curve Encryption were explained.

The solution to trusting other users over an insecure communication channel is solved by public key infrastructure which provides a network of trust over the whole internet. From security in communication, the paper continued to security in storage. In order to enable security, privacy and authentication features for storage media, it is integral to consider security on many different levels.



Do you have any questions? Get in touch!

Europe

+49 (30) 936 954 400
sales@swissbit.com

USA

+1 (978) 490 3252
salesna@swissbit.com

About Swissbit

Swissbit AG is the leading European manufacturer of storage, security and embedded IoT solutions for demanding applications. As trusted partner, Swissbit empowers the digital and connected world by reliably storing and protecting data in industrial, security and IoT applications.

www.swissbit.com

© Swissbit AG 2024 – All rights reserved.

Disclaimer:

No part of this document may be copied or reproduced in any form or by any means, or transferred to any third party, without the prior written consent of an authorized representative of Swissbit AG ("SWISSBIT"). The information in this document is subject to change without notice. SWISSBIT assumes no responsibility for any errors or omissions that may appear in this document and disclaims responsibility for any consequences resulting from the use of the information set forth herein. SWISSBIT makes no commitments to update or to keep current information contained in this document. The products listed in this document are not suitable for use in applications such as, but not limited to, aircraft control systems, aerospace equipment, submarine cables, nuclear reactor control systems and life support systems. Moreover, SWISSBIT does not recommend or approve the use of any of its products in life support devices or systems or in any application where failure could result in injury or death. If a customer wishes to use SWISSBIT products in applications not intended by SWISSBIT, said customer must contact an authorized SWISSBIT representative to determine SWISSBIT willingness to support a given application. The information set forth in this document does not convey any license under the copyrights, patent rights, trademarks or other intellectual property rights claimed and owned by SWISSBIT.

ALL PRODUCTS SOLD BY SWISSBIT ARE COVERED BY THE PROVISIONS APPEARING IN SWISSBIT'S TERMS AND CONDITIONS OF SALE ONLY, INCLUDING THE LIMITATIONS OF LIABILITY, WARRANTY AND INFRINGEMENT PROVISIONS. SWISSBIT MAKES NO WARRANTIES OF ANY KIND, EXPRESS, STATUTORY, IMPLIED OR OTHERWISE, REGARDING INFORMATION SET FORTH HEREIN OR REGARDING THE FREEDOM OF THE DESCRIBED PRODUCTS FROM INTELLECTUAL PROPERTY INFRINGEMENT AND EXPRESSLY DISCLAIMS ANY SUCH WARRANTIES INCLUDING WITHOUT LIMITATION ANY EXPRESS, STATUTORY OR IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.