SADS AY 2020/2021 April Semester CRYPTOGRAPHY GROUP PROJECT

Written and Collated By:
Cheryl Toh Wen Qi | 1900954G
Tan Yi Ching | 1902249J
Toh Yun Zhen | 1906146D
Zachary Phoon Jun Ze | 1900353B

Diploma & Class:
Diploma in Cybersecurity and Digital Forensics
P02

Declaration of plagiarism:

We understand that plagiarism is the blatant copying of information, codes or any other material (including ideas, music, pictures etc) and passing it off as our own original work. We undertake to ensure that all information in all the assessments and assignments that we undertake will be original.

We shall not include information without giving proper citations and reference lists. All pictures (if any) will be either original or from a free source.

We realize the serious nature of plagiarism and know that we will fail a subject, and possibly all subjects this semester if we plagiarize.

Signatures:

Cheryl Toh Wen Qi | 1900954G	[illegible]
Tan Yi Ching | 1902249J	[illegible]
Toh Yun Zhen | 1906146D	[illegible]
Zachary Phoon Jun Ze | 1900353B	[illegible]

Content Page

Executive Summary

The open-source Linux machines of different distributions have grown to become a significant force in today's computing environment; establishing itself as a preferred method and server platform for business environments and cloud infrastructures as a high-performing, secure and reliable host and platform that provides both flexibility and agility with low operating costs.

To better secure the Linux environment, different cryptographic securities are being developed and/or implemented widely by many businesses and enterprises in their data centers, and private and public clouds.

This report is mainly research on cryptography and its latest developments of different applications that are either in development or have been implemented in Linux hosts within a span of ten years.

Each subtopic in cryptography consists of a brief description, explanation, latest development, application in Linux and security. The topics are digital signatures, cryptographic algorithms, quantum technology, post quantum technology, cloud computing, cryptocurrency, blockchain technology, data storage, data transmission, disk encryption and endpoint encryption. In addition to that, we will illustrate and demonstrate the several usage of cryptography that can be applied or are already built in the Linux system.

Latest Development of Cryptography

1 Digital signatures

1.1 What is it

A digital signature is similar to a digital fingerprint that is unique and is used for verifying the authenticity of digital messages or documents. For example, user A sends data to user B via email. User B can be sure that the received data is the same data that was sent by user A through the use of digital signature.

1.2 How is it applied currently in Linux

In Linux, digital signatures are being used in MD5 source binary file generation. The creation of two keys which includes private and public (certificate) and binary file signing (ELF which is Executable and Linkable Format) which consists of the MD5 of the binary file being encrypted with the help of the private key.

While there is MD5, there is also SHA1 and SHA2 as examples. These are some of the types of hashing algorithms there are. The MD5 hash function produces a 128-bit hash value, the SHA1 produces 160-bit hash value and SHA2 produces a hash value of 256-bits, or 64 hexadecimal digits. Lastly, there is SHA3, however as it is still somewhat new, it is not widely adopted. Its algorithm is unrelated to the one used by its predecessor, SHA-2. Technically speaking, the longer the hash value, the harder it is to make it for someone to brute force the hash hence making it more secure. With that being said, the arrangement of the least secure to the better secure is: MD5, SHA1, SHA2, SHA3.

Additionally, Linux comes with the GNU Privacy Guard (GnuPG or GPG) encryption and authentication utility which encrypts the files with a key, and digitally signs a message to authenticate it. Users are also able to create their private and public key using that software.

Digital Signature is considered to be a much safer option as compared to hashes due to the hashes being able to be easily replaced by attackers if the hackers are able to replace files on the website. Digital Signature is a better option as for attackers to bypass the security mechanism, it would mean that the attackers have to steal the private key which would be usually secured and protected properly.

In Linux, there is a special kernel feature called Integrity Measurement Architecture (IMA) Appraisal. It is an extension of Secure Boot signature verification in Linux OS which reduces attack surface by only allowing signed software from trusted repositories to run. Additionally, it also keeps a list of executed applications and their measurements and signatures. However, it does not prevent malware from running and abuse of the signed software (such as malicious code was in the software but it was signed as trusted)

1.3 How is cryptography used in this?

Digital signatures use symmetric or public key encryptions to create their signatures. Digital Signature often uses symmetric cryptography as with symmetric cryptography, it is much faster and more suitable for encrypting large amounts of information.

Public-key encryption is used to solve the problem of delivering the symmetric encryption key for example, to the bank in a secure manner. To do so, the encryption of the symmetric key using the bank's public key is required. Since only the intended recipient has the corresponding private key, only the intended recipient will be able to recover the symmetric key and decrypt the check.

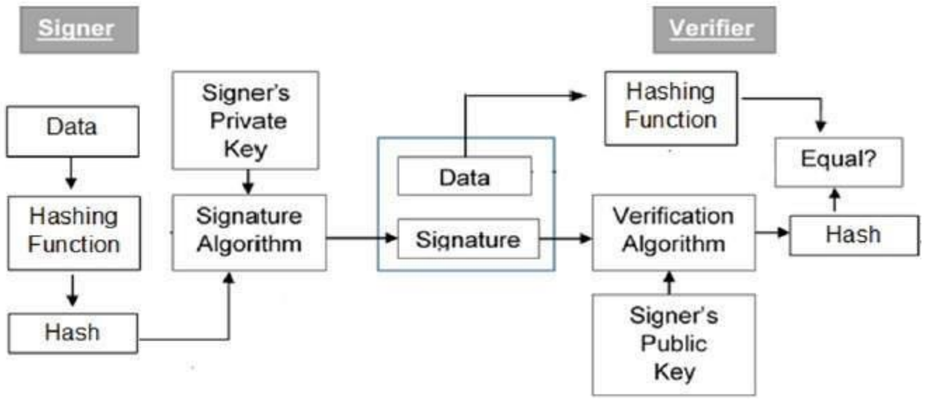


Fig. 1

In this scenario, Cheryl wants to send company information to her boss. Cheryl the signer and Cheryl's boss, the verifier both have a public-private key pair each. The key pairs used for encryption, decryption and signing,verifying are different as they both each serve different purposes. The private key that is used for signing is called signature key and public key as verification key.

In this scenario, Cheryl (signer) sends data to the hash function and generates a hash data. Then, the hash value and signature key are then sent to the signature algorithm which produces the digital signature. The signature is appended to the data and sent to Cheryl's boss (verifier) and Cheryl's boss (verifier) send the digital signature and verification key into the verification algorithm. To verify, the verification algorithm will generate a hash then will be used to compare with the first hash to determine if the digital signature is valid.

Asymmetric key encryption is more often used because if we use symmetric key encryption it can cause many problems. For example in a situation: If two person, Cheryl and Bob are sharing a secret symmetric key, which is only known by them. If Bob send to Cheryl the encrypted message M with key K, there will only be proof to show that the message M is created by Bob but whether the key is from Bob is unknown to Cheryl. This is because only Bob knows secret key K, so only he could encrypt the message M with this key.

Only one problem for Cheryl and Bob is how to share key K. This problem has led to the rise of public key cryptography which is also known as asymmetric cryptography. But because all conversations between Alice and Bob can be modified by Man in the middle attack, they have no opportunity to exchange secret key K using symmetric key encryption. This is why it is better to use public key cryptography. This way, Bob shared his public key and Cheryl can use it to prove that the received message was created by him.

It is possible to use symmetric key K for proof that the message was created by the specific person, but for this you first need to share the key. And for this to happen, users would need to use a public key or trusted Key distribution center, which is not as convenient as public key cryptography.

1.4 The latest development of Digital signatures technology.

Top industries that uses this technology

- **Finance & Insurance**
    - Added layer of security in protecting business against fraud.
    - Ensure privacy in data and encryption of data.
- **Healthcare and Life sciences**
    - Confidentiality and security for documents / records
    - Can be used as a court evidence/material
    - Ensure privacy and encrypt the data.

1.5 How to ensure the trust online [security wise]

Digital Signatures are generally used to bind signatory to the message.

Privacy: The information transmitted from a sender to a receiver is not accessed by others

Message authentication: The sender can be assured that the signature is created only by the sender who possesses the corresponding secret private key.

Data Integrity: The receiver can assure that the data received has not been breached and if it has been, he/she can safely deny the message.

Non-repudiation: Since it is assumed that only the signer has the knowledge of the signature key, he can only create unique signatures on a given data. Thus the receiver can present data and the digital signature to a third party as evidence if any dispute arises in the future.

By adding public-key encryption to digital signature schemes, we can create a system that can provide the four essential elements of security which is Privacy, Authentication, Integrity, and Non-repudiation.

In a public-key encryption scheme, a public key of sender is available in the open domain, and hence anyone can spoof his identity and send an encrypted message to the receiver. This makes it essential for users employing Public-key cryptography for encryption to seek digital signatures along with encrypted data to be assured of message authentication and non-repudiation. **This can be achieved by combining digital signatures with encryption schemes.** There are two possibilities for this situation, sign-then-encrypt and encrypt-then-sign schemes. However, the cryptosystem based on sign-then-encrypt can be exploited by the receiver to spoof the identity of the sender and send that data to a third party. Hence, this method is not preferred. The process of encrypt-then-sign is more reliable and chosen for this situation.

2 Cryptographic algorithms

2.1 What it is?

A cryptographic algorithm which also can be known as a cipher, alters data from a readable form (plain text) to a protected form (ciphertext) and back to readable form. Changing plaintext to ciphertext is known as encryption, whereas changing ciphertext to plaintext is known as decryption.

2.2 How is it applied currently in Linux

Crypto API is a cryptography framework in the Linux kernel, for various parts of the kernel that deal with cryptography

The kernel crypto API provides different API calls for the following cipher types:

• Symmetric ciphers
The symmetric cipher commands allow data to be encrypted or decrypted using various block and stream ciphers using keys based on passwords. Base64 encoding or decoding can also be performed.

• AEAD ciphers (Authenticated Encryption with Associated Data)
It simultaneously assures the confidentiality and authenticity of data and the ability to check the integrity and authentication of additional authenticated data that is sent in the clear.

• Message digest
Message digest algorithms rely on cryptographic hash functions to generate a unique value that is computed from data and a unique symmetric key

• Random number generation
Cryptographically secure pseudorandom number generator or Cryptographically pseudorandom number generator. Pseudo-random number generators (PRNGs) are algorithms that can create long runs of numbers with good random properties but eventually the sequence repeats. Thus, the term ‘pseudo’ random number generators. The algorithms essentially generate numbers that, while not being truly random, are random enough for cryptographic applications.

• User space interface
User space can only act as a consumer and never as a provider of a transformation or cipher algorithm. They can only use it but not make changes to the algorithm and settings.

2.3 The latest development of cryptographic technology.

It can also be seen in the usage in image data such as :

- Digital Watermarking
- Steganography

There is an ongoing development in the field of authentication using biometric data. There is also an ongoing development and research on post-quantum cryptography:

- As several computationally difficult problems for classical computing models are susceptible to attacks in quantum computing models. The post-quantum cryptographic algorithms are made to handle those challenges. Currently, it is still under development and research.

Internet Of Things (IoT): Television,fridge, microwave, washing machines, microwaves, smoke detectors and home assistant.

- All of these need to communicate with each other to relieve the end-user from manual interventions in many real-time processes.

- These devices usually use lightweight encryption/decryption algorithms since they are resource-constrained but ultimately, their main goal is not to compromise the security and authenticity of the communicated data too much. As IoT becomes more popular, this has led to the development of lightweight cryptography for resource-constrained devices.

2.4 How to ensure the trust online [security wise]

It changes the plaintext to a protected form which is known as ciphertext and then back to plaintext. This ensures security as normally people would think that it is meaningless as it looks like a bunch of letters, symbols and characters all mashed together into a weird sentence/phrase. It also makes it harder for other people who managed to steal information or the ciphertext to figure out the actual contents, thus making it more secure.

2.4.1 Advantages

It provides the four basic services of cybersecurity

- **Confidentiality** − Encryption technique can guard the information and communication from unauthorized revelation and access of information.

- **Authentication** − The cryptographic techniques such as MAC and digital signatures can protect information against spoofing and forgeries.

- **Data Integrity** − The cryptographic hash functions are playing a vital role in assuring the users about the data integrity.

- **Non-repudiation** − The digital signature provides the non-repudiation service to guard against the dispute that may arise due to denial of passing messages by the sender.

All these fundamental services offered by cryptography have enabled the conduct of business over the networks using the computer systems in an extremely efficient and effective manner.

2.4.2 Disadvantages

Difficult to access:

- A strongly encrypted, authentic, and digitally signed information can be **difficult to access** even for a legitimate user at a crucial time of decision-making. The network or the computer system can be attacked and rendered non-functional by an intruder.

Limitations:

- Cryptography does not guard against the vulnerabilities and threats that emerge from the poor design of systems, protocols, and procedures. These need to be fixed through proper design and setting up of a defensive infrastructure.

- Addition of cryptographic techniques in information processing leads to delay.

3 Quantum Technology

3.1 What is it?

Quantum cryptography is the science of exploiting quantum mechanical properties to perform cryptographic tasks. BB84 is a quantum key distribution scheme developed by Charles Bennett and Gilles Brassard in 1984. It's the first quantum cryptography protocol

3.2 How does it work?

Quantum Phenomena refers to processes showing quantum behavior at the microscopic scale than atomic scale as quantum effects are prevalent.

It uses this Phenomena to create a noise source with a higher level of entropy (example: Randomness) compared to other techniques.

Quantum cryptography is the only known method for transmitting a secret key over distance that is secure in principle and based on the laws of physics. Current methods for communicating secret keys are all based on unproven mathematical assumptions. These same methods also are at risk of becoming cracked in the future, compromising today's encrypted transmissions retroactively. This matters very much if you care about long-term security.

3.3 How is it applied currently to Linux?

Not applied into Linux as It runs on its own OS and machine. It will affect Linux as it can easily cipher daily crypto much faster and easier. For example, a regular computer in cubits , it will either be one or a zero however for a quantum computer , it can be with one or zero or both at the same time. This means it can easily crack any algorithm used.

3.4 Latest Development

Google has claimed quantum supremacy. Their quantum computer can perform a specific task in 200 seconds that would take the world's best supercomputer 10,000 years to complete.

3.5 Is it trustable?

Quantum Key Distribution uses quantum mechanics to guarantee secure communication. It enables two parties to produce a shared random secret key known only to them, which can then be used to encrypt and decrypt messages.

The system relies mainly on the secrecy of the password, the correctness and completeness of quantum theory and the reliability of the devices in the quantum communication system. So if the password is leaked the message can be easily compromised.

4 Post-Quantum

4.1 What is it?

Post-quantum cryptography refers to cryptographic algorithms that are thought to be secure against an attack by quantum computers. However, it isn't accurate for most popular public-key algorithms as it can be efficiently broken by a sufficiently strong quantum computer. But the race to make a algorithm to protect sensitive electronic information is still ongoing.

4.2 How does it work?

Currently, research is mainly focused on six Different approaches.

4.2.1 Lattice-based Cryptography

- This approach includes cryptographic system such as learning with errors, ring learning with errors(ring-LWE) , the ring learning with errors key exchange and the ring learning with errors signature, the older NTRU or GGH encryption scheme, and the newer NTRU signature and BLISS signatures.
- Some of the Schemes like NTRU encryption have been studied for many years without anyone finding a feasible attack while others like ring-LWE algorithms have proof that their security reduces to a worst-case problem
- NSA has redid reviewed this algorithm and agreed with the NIST assessment, documented in NISTIR 8309: Status Report on the Second Round of the NIST Post-Quantum Cryptography Standardization Process, that these are among the most efficient post-quantum designs. Based on their history of analysis and implementation efforts, NSA CSD expects that a NIST-candidate lattice-based signature and a NIST-candidate lattice-based key encapsulation mechanism will be approved for NSS.

4.2.2 Multivariate Cryptography

- This includes cryptographic systems such as the Rainbow (Unbalanced Oil and Vinegar) scheme which is based on the difficulty of solving systems of multivariate equations. Various attempts to build secure multivariate equation encryption schemes have failed. However, multivariate signature schemes like Rainbow could provide the basis for a quantum secure digital signature.

4.2.3 Hash-based cryptography

- This includes cryptographic systems such as Lamport signatures and the Merkle signature scheme and the newer XMSS and SPHINCS schemes. Hash-based digital signatures were invented in the late 1970s by Ralph Merkle and have been studied ever since as an interesting alternative to number-theoretic digital signatures like RSA and DSA. Their primary drawback is that for any hash-based public key, there is a limit on the number of signatures that can be signed using the corresponding set

of private keys. This fact had reduced interest in these signatures until interest was revived due to the desire for cryptography that was resistant to attack by quantum computers. There appear to be no patents on the Merkle signature scheme and there exist many non-patented hash functions that could be used with these schemes.
- However, the stateful versions have a limited number of allowable signatures per public key and require the signer to maintain an internal state. Because of this, they are not suitable for all applications. NSA CSD expects that the stateful signatures LMS and XMSS will be standardized by NIST in NIST SP 800-208 and approved for NSS solutions for certain niche applications where maintaining state is not a problem.

4.2.4 Code-based cryptography

- This includes cryptographic systems which rely on error-correcting codes, such as the McEliece and Niederreiter encryption algorithms and the related Courtois, Finiasz and Sendrier Signature scheme. The original McEliece signature using random Goppa codes has withstood scrutiny for over 30 years. However, many variants of the McEliece scheme, which seek to introduce more structure into the code used to reduce the size of the keys, have been shown to be insecure

4.2.5 Supersingular elliptic curve isogeny cryptography

- This cryptographic system relies on the properties of supersingular elliptic curves and supersingular isogeny graphs to create a Diffie-Hellman replacement with forward secrecy.
- Cryptographic system uses the well-studied mathematics of supersingular elliptic curves to create a Diffie-Hellman like key exchange that can serve as a straightforward quantum computing resistant replacement for the Diffie-Hellman and elliptic curve Diffie–Hellman key exchange methods that are in widespread use today. Because it works much like existing Diffie–Hellman implementations, it offers forward secrecy which is viewed as important both to prevent mass surveillance by governments but also to protect against the compromise of long-term keys through failures

4.2.6 Symmetric key quantum resistance

- Provided one uses sufficiently large key sizes, the symmetric key cryptographic systems like AES and SNOW 3G are already resistant to attack by a quantum computer.
- Further, key management systems and protocols that use symmetric key cryptography instead of public-key cryptography like Kerberos and the 3GPP Mobile Network Authentication Structure are also inherently secure against attack by a quantum computer.
- Given its widespread deployment in the world already, some researchers recommend expanded use of Kerberos-like symmetric key management as an efficient way to get post-quantum cryptography today.

4.3 How is it applied currently to Linux?

No articles regarding application to Linux however some of these algorithms may be implemented in the future to help improve and stabilize the security of the internet and network.

4.4 Latest Development

NIST (National Institute of Standards and Technology) has yet to provide any new developments or algorithms that have proven to be post-quantum ready as they are still reviewing public key submissions.

At the present time, NSA CSD does not anticipate the need to approve other post-quantum cryptographic technologies for NSS usage but recognizes circumstances could change going forward. A variety of factors—including confidence in security and performance, interoperability, systems engineering, budgeting, procurement, and other requirements—could affect such decisions.

4.5 Is it trustable?

Not 100% trustable as it is still in development, but it is available as most open source codes can easily be cracked however there are a few algorithms that have proved to be successful.

5 Cryptography in Cloud Computing

5.1 What is Cloud Computing?

Cloud computing, in simple terms, means the storing and accessing of data and programs over the Internet instead of your computer's hard drive. It is a technology that uses the Internet and central remote servers to maintain its data and applications.

Its' main purpose serves as an application-based software infrastructure that stores data, applications and programs on remote servers, which allows consumers and businesses alike to use applications without installation and access their personal files at any computer that has internet access.

5.2 Types of Cloud Deployment and their Services

On the point of cloud computing deployment, all deployments operate with the same principle, which is to virtualise the computing power of servers that are being hosted into segmented and software-driven applications that provide both processing and storage capabilities without decreasing its original competence.

Cloud deployment refers to the method on how a cloud platform is being hosted, implemented, accessed and used on the Internet; the enablement of cloud solution services that may be accessed on demand by consumers.

There are four different cloud deployment models as listed;

- Public Cloud
- Private Cloud
- Community Cloud
- Hybrid Cloud

With the different cloud deployments, there are different types of services mainly catered towards public cloud deployments for businesses as listed with each of their own explanations;

- Software-as-a-Service (SaaS) – *businesses or consumers subscribe to an application it accesses over the Internet, whereby the application is from a third-party provider*
- Platform-as-a-Service (PaaS) – *businesses can create their own custom applications for use by their company's users and consumers with hardware and software tools available over the Internet*
- Infrastructure (IaaS) –*company and players—eg. Amazon, Google—provides online services that can be used by other companies with pay-as-you-go payments*

5.3 Applying Cryptography in Cloud Computing

Many different businesses and organisations are adopting the usage of cloud computing and its technologies due to its efficiency and flexibility that provides organisations' businesses many benefits. However, public and shared infrastructure always have security issues and loopholes. It is important to not take security for granted. Cloud computing is fundamentally the delivery of digital services and data is its greatest asset. As the main controller of the public's data, it is important that we need to retain the trust of the service's security.

For Linux machines and/or applications that are deployed on a cloud platform, key management services (KMSs) are used, which is provided by the cloud service provider.

The services are low in cost or cost-free. However, the disadvantage is that most KMSs on the cloud platforms use proprietary interfaces, which locks consumers, users and/or businesses in that particular service provider, making it hard to migrate to other platforms. It causes difficulties in implementing dedicated KMSs outside of the cloud platform as well.

One means of better integrating cryptography into cloud computing is by **encrypting data-at-rest** in the cloud. This can be done by encrypting the data at the edge of the cloud, or within the cloud itself.

> **Data-At-Rest Encryption**
> Data-at-rest encryption ensures that files stored on disk are always in an encrypted form/format. Data and files will be available to the OS and applications only when the system is running/unlocked by a trusted user.
> In addition, data-at-rest encryption can be used to add extra security measures against unauthorised attempts to tamper with one's OS. *For example, the installation of keyloggers or attackers implanting viruses or trojan horses to gain physical access.*

On the other hand, data-at-rest encryption should only be viewed as additional security to the existing security mechanisms of the OS or servers. Regardless of what method was used, the vital point is to ensure that the **encryption keys are properly managed and kept secure**. This is important since different organisations use different cloud services.

However, there is the issue with encrypting all the data in the cloud—you can no longer process it inside there.

5.4 Cloud Computing Cryptography with Hardware Security Modules (HSMs)

To better utilise and integrate cryptography on top of cloud computing, this is where Hardware Security Module (HSM) comes into play. HSM is a physical computing device which manages digital keys and keeps it secure, has encryption and decryption capabilities for digital signatures, has strong authentication and other cryptographic functions; all performed at high speed within a tamper-resistant device typically with a recognised independent certification such as **FIPS**

140-2 or **Common Criteria**. (*For more information on FIPS 140-2 and Common Criteria, please refer to the documentation below that is indented.)*

FIPS 140-2

The Federal Information Processing Standard 140-2 (FIPS 140-2) is an information technology computer security standard used to prove and validate cryptographic modules produced by private sector companies to ensure that their product meets well-defined security standards.
The publication of this requirements and standards for this certification is titled *Security Requirements for Cryptographic Modules*.
FIPS 140-2 has 4 levels:

- Level 1: Provides the lowest level of security with basic security requirements specified for a cryptographic module. No physical security features required.
- Level 2: Improves upon physical security mechanisms of a Level 1. Tamper-evident physical security features included in this Level.
- Level 3: Improves upon Level 2 tamper-evident physical security mechanism. Additional tamper-responding features in Level 3 designed to notify of unauthorised access and modification on the module as well as identity-based authentication.
- Level 4: Provides highest level of security. Improvement from Level 3, this Level has a complete tamper-responding envelope of protection that immediately deletes all plaintext keys upon detection of unauthorised access as well as protection against attacks using extreme voltage or temperature changes from outside.

Common Criteria

Common Criteria is a globally recognised HSM certification, which guarantees the assurance level of an HSM. CC is an international standard (ISO/IEC 15408) for IT Security Evaluation. The CC certification standard reduces the need for multiple evaluations in international markets; allowing products being sold into international markets to not need re-evaluation beforehand.

HSM is designed to withstand both **physical** and **logical attacks** on top of its capability to perform cryptographic operations and protect the keys.

Functions of HSM

- Onboard secure cryptographic key generation and storage—of at least the top-level and most master keys
- Key management
- Use of cryptographic and sensitive data material – performing encryption or digital signature functions
- Offloading application servers for complete asymmetric and symmetric cryptography

Knowing where the security keys are located will help in identifying what are relevant to the organisation’s security requirements and provide solid risk reduction while lessening complexity of migrating between servers on top of ensuring better management.

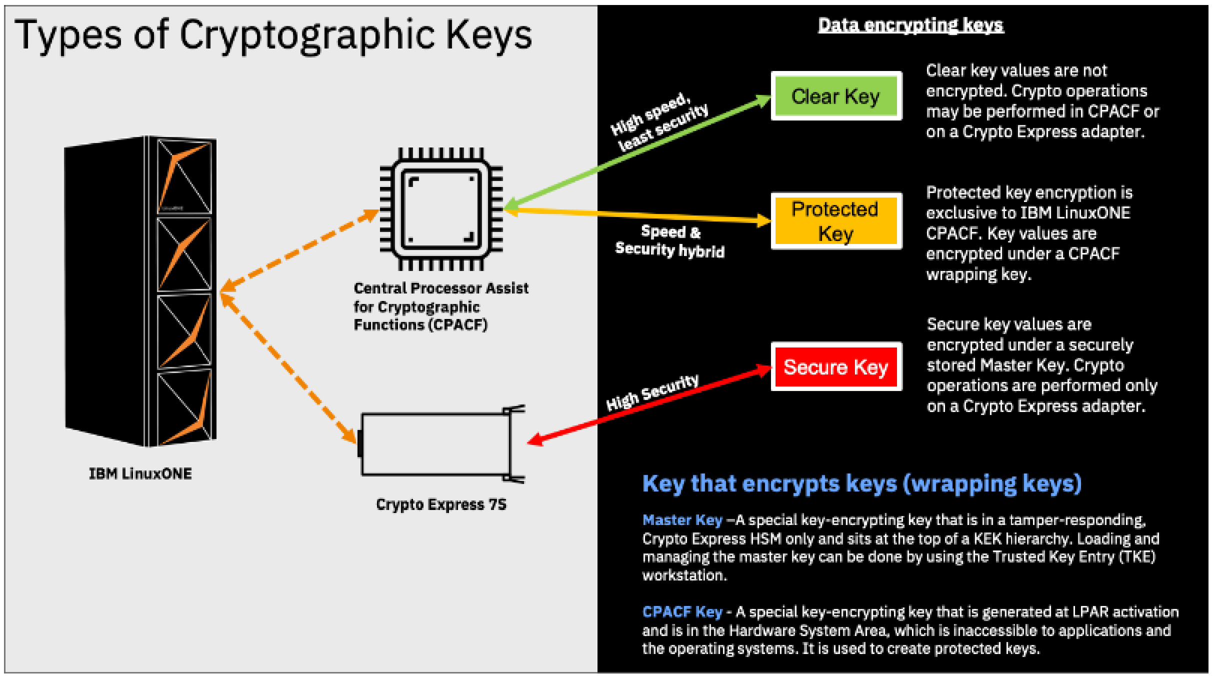
HSMs could be used as a Service Model in the organisation’s enterprise with inclusion of integrating KMS service features as well. Utilising HSM as a Service provides an additional layer of security that eliminates the need for physical hardware devices to be located on-site.

This implementation of integrating cryptography into cloud computing with HSMs for Linux have already been in some developments in the current world. Please refer to section 5.5 for more information.

5.5 Latest Development

5.5.1 LinuxONE by IBM

LinuxONE by IBM has two types of cryptographic keys. *Please refer to the screenshot below*.



Clear key values are not encrypted. Crypto operations may be performed in CPACF or on a Crypto Express adapter.

Protected key encryption is exclusive to IBM LinuxONE CPACF. Key values are encrypted under a CPACF wrapping key.

Secure key values are encrypted under a securely stored Master Key. Crypto operations are performed only on a Crypto Express adapter.

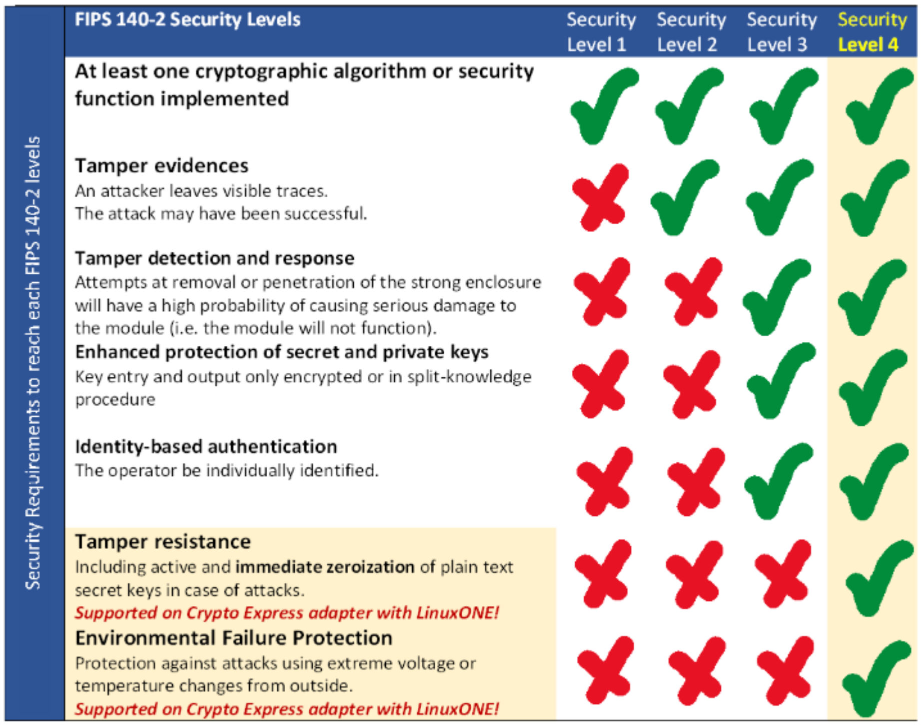
Key that encrypts keys (wrapping keys)

Master Key –A special key-encrypting key that is in a tamper-responding, Crypto Express HSM only and sits at the top of a KEK hierarchy. Loading and managing the master key can be done by using the Trusted Key Entry (TKE) workstation.

CPACF Key - A special key-encrypting key that is generated at LPAR activation and is in the Hardware System Area, which is inaccessible to applications and the operating systems. It is used to create protected keys.

Fig. 3 *Encryption key implementations and tradeoffs*

This portion of the report will be focusing on the Crypto Express adapter. The IBM Crypto Express adapter is a HSM that is certified for PCI-HSM and FIPS 140-2 Level 4--the highest level available in the industry and one that very few devices are able to attain.



Security Requirements to reach each FIPS 140-2 levels

FIPS 140-2 Security Levels	Security Level 1	Security Level 2	Security Level 3	Security Level 4
At least one cryptographic algorithm or security function implemented	✓	✓	✓	✓
Tamper evidences An attacker leaves visible traces. The attack may have been successful.	✗	✓	✓	✓
Tamper detection and response Attempts at removal or penetration of the strong enclosure will have a high probability of causing serious damage to the module (i.e. the module will not function).	✗	✗	✓	✓
Enhanced protection of secret and private keys Key entry and output only encrypted or in split-knowledge procedure	✗	✗	✓	✓
Identity-based authentication The operator be individually identified.	✗	✗	✓	✓
Tamper resistance Including active and **immediate zeroization** of plain text secret keys in case of attacks. *Supported on Crypto Express adapter with LinuxONE!*	✗	✗	✗	✓
Environmental Failure Protection Protection against attacks using extreme voltage or temperature changes from outside. *Supported on Crypto Express adapter with LinuxONE!*	✗	✗	✗	✓

Fig. 4

These Crypto Express adapters can be interchanged between a series of different other adapters under the same category. These are called **CryptoCards - HSMs**. They are as listed below:

- ❖ IBM 4769 HSM - referred to as IBM Crypto Express7S or CEX7S
- ❖ IBM 4768 HSM - referred to as IBM Crypto Express6S or CEX6S
- ❖ IBM 4767 HSM - referred to as IBM Crypto Express5S or CEX5S
- ❖ IBM 4765 HSM - referred to as IBM Crypto Express4S or CEX4S

The HSM in LinuxONE is accessed from a host computer that uses a set of generic or specialised API functions. The HSM is able to support the following:

- IBM Common Cryptographic Architecture (CCA) -- primary secure key cryptographic mode
  - This mode is often called *cryptographic co-processor mode*
  - Primarily to support applications' finance industry payments
- IBM Enterprise PKCS #11 (EP 11) mode -- API
  - EP11 Is designed to support open standards and enhanced security for customers; offering various general purpose, secure-key-only cryptography functions.
- Accelerator -- key accelerator mode
  - Enables the IBM's CryptoExpress hardware--their HSMs--become an asymmetric cryptographic function accelerator.
  - This mode is suggested to be useful for enterprise web services.

- *Tested with CEX6S running on LinuxONE in accelerator mode with 8 concurrent processes, it is able to process about 9k SSL/TLS handshakes per second with a 2048-bit RSA key.*

Below is an image of the platforms supported by each of the IBM CryptoCards HSM.

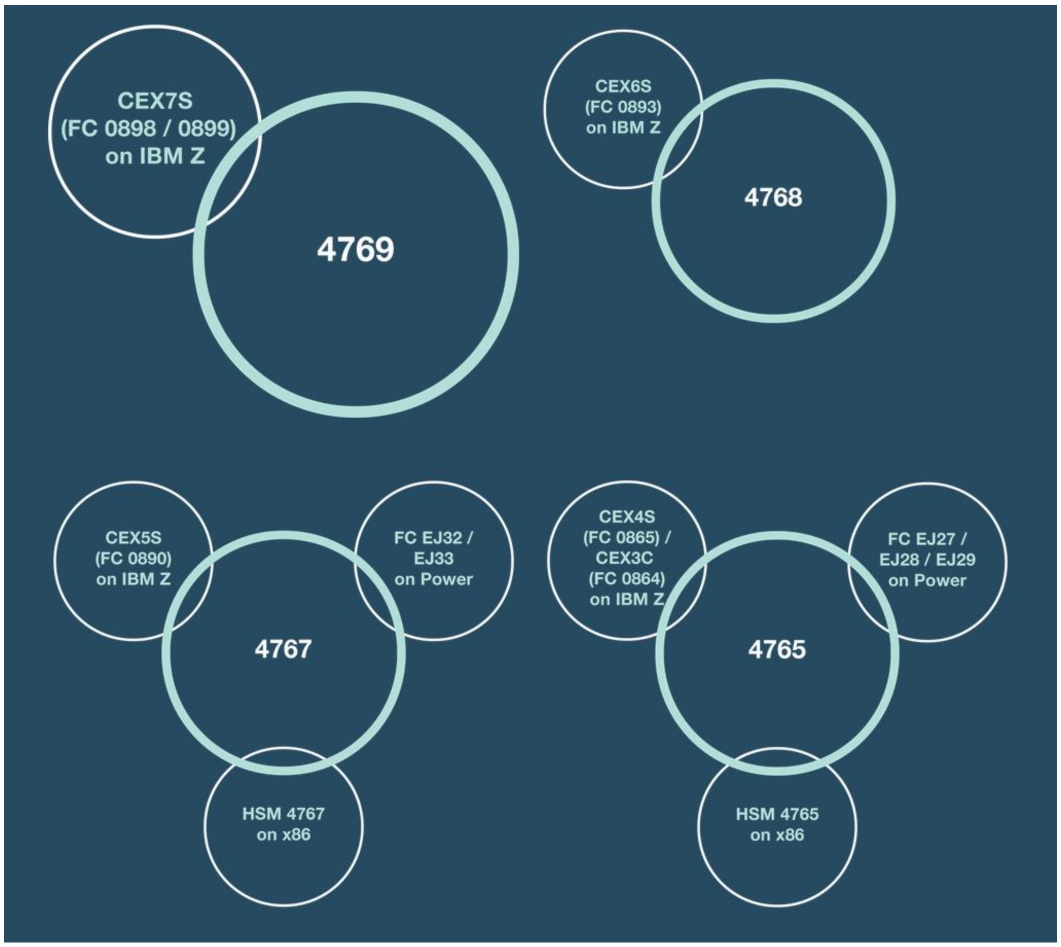


Fig. 5

To know more about each of IBM's CryptoCards series and their own specialized functions, please refer to https://www.ibm.com/security/cryptocards/hsms

5.6 Is It Trustable?

If used, HSMs hold a critical role in securing applications and the infrastructure. HSMs and/or the cryptographic modules inside are typically certified to internationally recognised standards such as FIPS 140 or Common Criteria to provide users and consumers with independent assurance that the implementation of the product and cryptographic algorithms are sound.

Although HSMs were mainly designed to manage and protect cryptographic keys both at rest and in use, they have the features of providing tamper evidence and/or event logging and alert, and has tamper resistance which makes tampering with data inside the servers difficult without making the HSM inoperable, or tamper responsiveness such as deleting security keys upon tamper detection.

Additionally, some HSM systems are hardware cryptographic accelerators. With performance ranges from 1 to 10,000 1024-bit RSA signs per second, HSMs can provide significant CPU offload for asymmetric key operations.

Assurances in the security with integration of cryptography with HSMs for cloud computing is thus one of the best choices and will probably be for the many coming years.

6 Elliptic Curve Cryptography

6.1 What is Elliptic Curve Cryptography

Elliptic Curve Cryptography (ECC) is an approach to public-key cryptography and encryption based on the complex algebraic structure of elliptic curves over finite fields to be doing **asymmetric cryptography**. ECC typically uses the same types of algorithms as that of Diffie-Hellman Key Exchange and RSA Encryption, where ECC is more widely based on the said former algorithm. ECC allows smaller keys compared to non-EC cryptography to provide equivalent security as well.

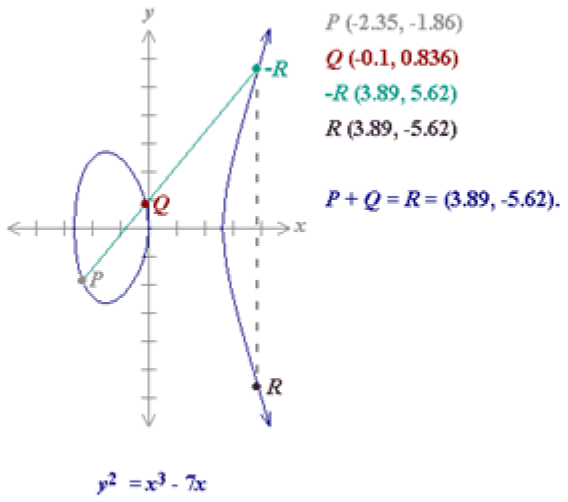


Fig. 6

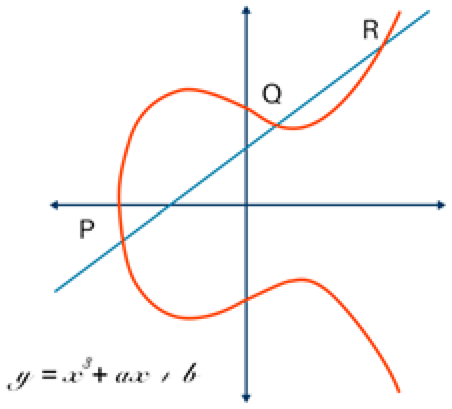


Fig. 7

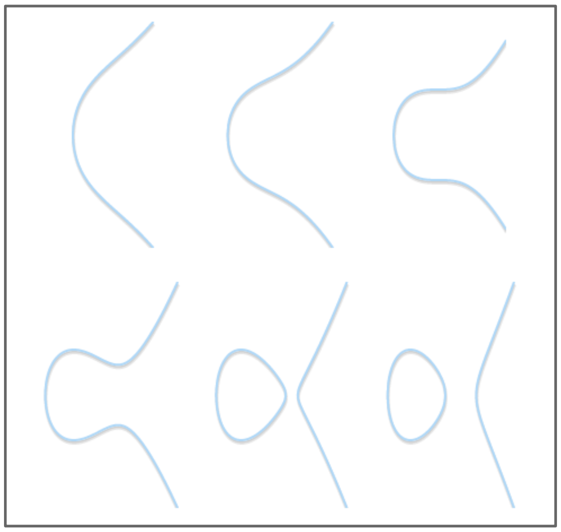


Fig. 8 Different shapes for different elliptic curves; **b = 1**, **α** varying from 2 to -3.

Figures 6 and 7 show an example of an elliptic curve. Figure 6 could be used in conjunction with an RSA-type algorithm, where primes “P” and “Q” were chosen. When these two primes are chosen using a predefined elliptic curve in a finite field, the key sizes can be much smaller and still yield the same amount of security. This causes the time it takes to perform the encryption and decryption to be **drastically reduced**, allowing an **increase in the amount of data being passed with equal security.**

Depending on the value of a and b, elliptic curves may assume different shapes on the graph plane. Elliptic curves can be easily verified since they are symmetrical on the x-axis as seen in figure 8.

Elliptic curves are applicable for key agreement, digital signatures, pseudo-random generators and other tasks. They can be used for encryption by combining the key agreement with a symmetric encryption scheme, as well as used in several integer factorization algorithms based on elliptic curves that have applications in cryptography.

Elliptic curve cryptographic schemes were proposed independently by Neal Koblitz and Victor Miller in 1985. *See **section 6.1.1** for more information.*

Further reading on the explanation of the mathematical formula and concept behind elliptic curves used in cryptography can be read in these links below:

- ❖ https://andrea.corbellini.name/2015/05/17/elliptic-curve-cryptography-a-gentle-introduction/
- ❖ https://wizardforcel.gitbooks.io/practical-cryptography-for-developers-book/content/asymmetric-key-ciphers/elliptic-curve-cryptography-ecc.html
- ❖ https://www.linuxjournal.com/content/elliptic-curve-cryptography

6.1.1 ECC Algorithms

ECC provides several groups of algorithms based on the discrete logarithm problem, where the underlying main objective is the group of points defined over a finite field. Listed below are a few discrete logarithm-based protocols that have elliptic curves adapted into them:

- ECC **Digital Signature algorithms** like ECDSA and EdDSA (twisted Edwards Curve) schemes
- ECC **Encryption algorithms** and **hybrid encryption** schemes like ECIES integrated encryption and EEECC (EC-based ElGamal)
- ECC **key agreement algorithms** like ECDH, X25519 and FHMQV

(Please refer to 6.1.2 for the relative cryptographic schemes)

6.1.2 Cryptographic schemes

- ❖ **Elliptic Curve Diffie-Hellman (ECDH)** key agreement -- which is based on the Diffie-Hellman scheme

- **Elliptic Curve Integrated Encryption Scheme (ECIES)**, or also known as *Elliptic Curve Augmented Encryption Scheme* and the *Elliptic Curve Encryption Scheme*
- **Elliptic Curve Digital Signature Algorithm (ECDSA)** -- which is based on the Digital Signature Algorithm
  - Concepts of ECDSA algorithms revolves around private and public keys, as well as the digital numeric signature(s)
- **Deformation scheme** -- which uses Harrison's p-adic Manhattan metric
- **Edwards-curve Digital Signature Algorithm (EdDSA)** -- which is based on Schnoor signature and uses twisted Edwards curves
- **Elliptic Curve Menezes-Qu-Vanstone (ECMQV)** key agreement -- which is based on the Meneze-Qu-Vanstone (MQV) key agreement scheme
- **Elliptic Curve Qu-Vanstone (ECQV)** implicit certificate scheme

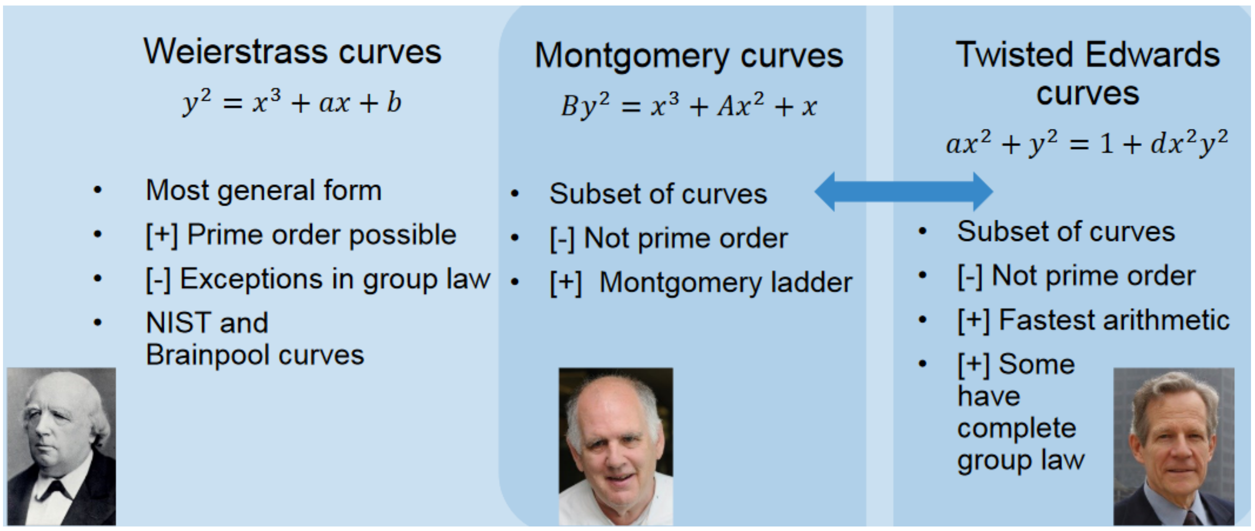
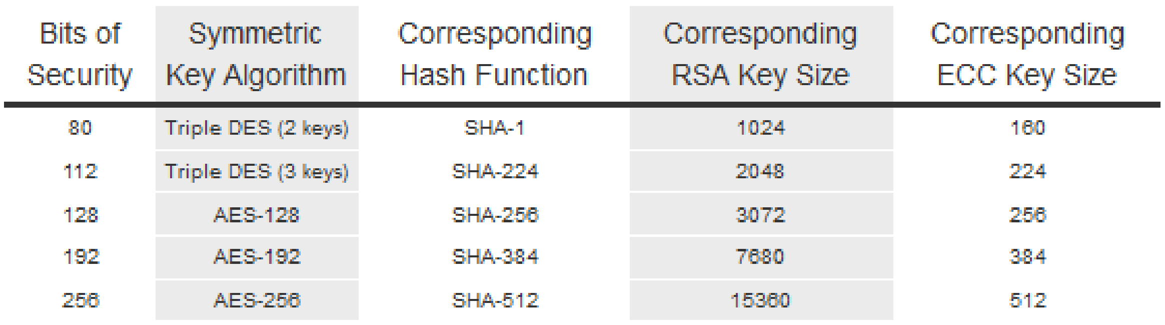


Fig. 9 Some examples of the ECC schemes and the formula behind them

6.2 Advantage for Using Elliptic Curve Cryptography

Businesses are integrating Elliptic Curve Cryptography more than RSA because of the key size and cryptographic strength comparisons between them.



Bits of Security	Symmetric Key Algorithm	Corresponding Hash Function	Corresponding RSA Key Size	Corresponding ECC Key Size
80	Triple DES (2 keys)	SHA-1	1024	160
112	Triple DES (3 keys)	SHA-224	2048	224
128	AES-128	SHA-256	3072	256
192	AES-192	SHA-384	7680	384
256	AES-256	SHA-512	15360	512

Fig. 10

The advantage of using ECC is the reduced key size. From the chart above, you can see that ECC is able to provide the same cryptographic strength with a much smaller key size. This appeals to businesses, especially for those who have limited processing power and storage but still need the same cryptographic strength as an RSA encryption.

6.3 Latest Development

ECC algorithms entered wide use in 2004 to 2005. As the years went by, more research and developments were released.

6.3.1 RSA Suite B

At RSA Conference 2005, the National Security Agency(NSA) announced **Suite B** which exclusively uses ECC for digital signature generation and key exchange. It provides a security level of 128 bits or higher, which is significantly higher than many commonly used standards, securing information travelling over networks. The suite is intended to protect both classified and unclassified national security systems and information.

Suite B algorithms makes it easier to collaborate in environments where information gathering and/or sharing is hindered due to costs or logistics. Suite B also helps public and private sector organisations meet compliances including Payment Card Industry (PCI), Health Insurance Portability and Accountability Act (HIPAA), Federal Information Processing Standards (FIPS) and others.

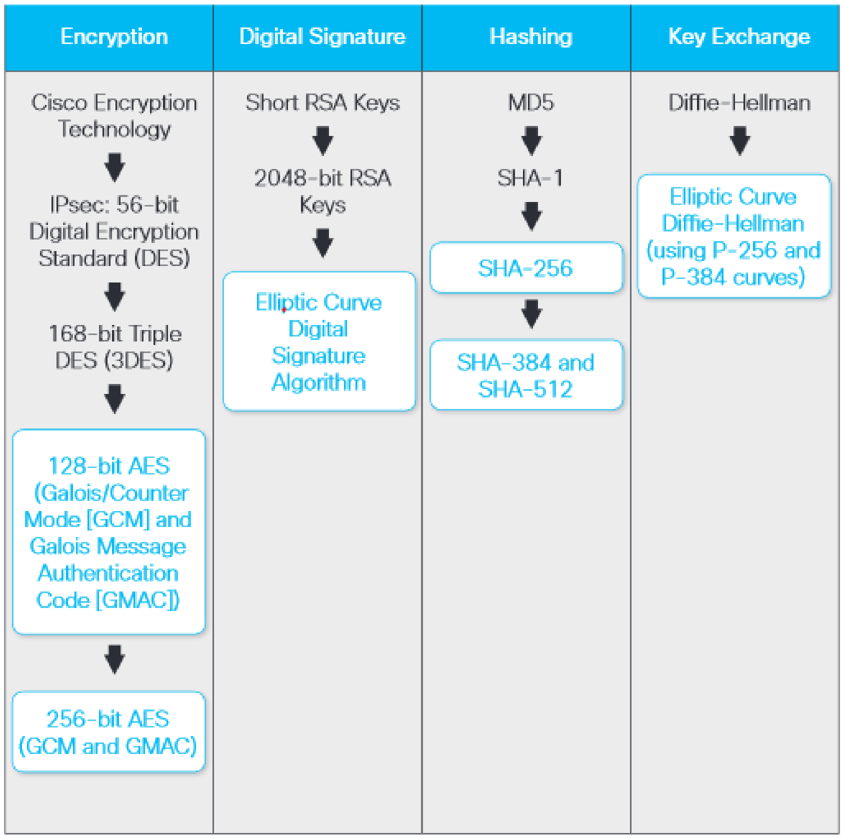
Suite B's components used to secure data over the network are:

- ❖ Elliptic Curve Digital Signature Algorithm (ECDSA) - digital signatures
- ❖ Elliptic Curve Diffie-Hellman (ECDH) - key agreements
- ❖ Advanced Encryption Standard (AES) - symmetric encryption
- ❖ Secure Hash Algorithm 2 | SHA-256 and SHA-384 - message digest

These four public-domain cryptographic algorithms working as a whole provides adequate security over classified data.

Suite B is actually used in many products. One of the named products is Cisco. Cisco had integrated and used Suite B in their solutions and coming up with better security. Leading the design and standardisation of Suite B, they came up with several solutions to up the security level of Suite B. One is Galois/Counter Mode (GCM) of operation for AES that is used for symmetric encryption in Suite B, where GCM provides both confidentiality and authentication. More recently, Cisco contributed to the practical implementation methods of the ECC algorithms that provide the public key cryptography encryption for Suite B.

Below is a table picture which shows the evolution of cryptographic technologies in Cisco Solutions. Suite B technologies are shown in blue.



Encryption	Digital Signature	Hashing	Key Exchange
Cisco Encryption Technology	Short RSA Keys	MD5	Diffie-Hellman
IPsec: 56-bit Digital Encryption Standard (DES)	2048-bit RSA Keys	SHA-1	Elliptic Curve Diffie-Hellman (using P-256 and P-384 curves)
168-bit Triple DES (3DES)	Elliptic Curve Digital Signature Algorithm	SHA-256	
128-bit AES (Galois/Counter Mode [GCM] and Galois Message Authentication Code [GMAC])		SHA-384 and SHA-512	
256-bit AES (GCM and GMAC)			

Fig. 11

Integrating Suite B cryptography standards in Cisco’s own VPN products, it is able to gain additional security, scalability and operational efficiencies not available in other products that implements Suite B as well. One good example is Cisco IOS Software.

Cisco IOS Software

This software includes built-in security technologies such as IPsec standards with integrated solutions that is built on top of those standards such as Dynamic Multipoint VPN (DMVPN). Both IPsec and DMVPN can use Suite B to provide strong data authentication and confidentiality, entity authentication services and anti-replay services.

Using IPsec in combination with Cisco IOS Software capabilities, companies can build and make VPNs more secure and reliable, have it be able to prioritise latency-sensitive traffic--voice and video.

6.3.2 Bitcoin

The ECC used in bitcoin is the central operation which addresses the public keys and authenticates transactions using digital signatures. The public keys and digital signatures are accessible to all as part of the publicly available and auditable block chain to prevent double-spending.

Bitcoin mainly uses the **ECDSA** cryptographic algorithm to ensure that funds can only be spent by their rightful owners. The elliptic curve used by Bitcoin is a mix of Ethereum and many other cryptocurrencies and is thus called secp256k1. The equation for the secp256k1 curve is $y^2 = x^3+7$, which looks like:

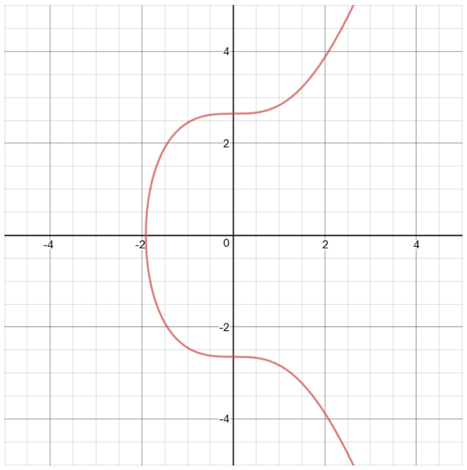


Fig 12. Image of secp256k1's elliptic curve

In Bitcoin, ECDSA curve is an additional enhancement security layer to the cryptocurrencies' security. ECDSA private key typically serves as a user's account. How it works is, the transaction of bitcoins from user A to user B is authenticated and acknowledged by attaching a digital signature--using user A's private key--of the hash of the previous transaction and information about the public key of user B at the end of the new transaction. The signature can be verified with the help of user A's public key.

As the ledger grows, the time taken to verify the transactions will also increase *(please refer to section **7 Cryptocurrency** for more information).* Thus ECC is better with bitcoin; helping to boost the exchange and transaction rate of verification.

6.3.3 OpenSSH

In year 2012, OpenSSH 5.7 added support for ECC-based cryptography, and now versions of OpenSSH that support ECC are **OpenSSH 5.7 or later**, and **OpenSSL version 0.9.8g or later**.

This development has yet to be distributed into every Linux's distribution, but ECC's usage in the machines' systems is becoming more widespread. There are reports of people migrating over. Its introduction into open-source tools like OpenSSL and OpenSSH is definitely a great step toward gaining more widespread use.

ECC private key--which is named **ssh_host_ecdsa_key**--and corresponding public key--which is named **ssh_host_ecdsa_key.pub**--are both stored in **/etc/ssh** when OpenSSH is installed in the Linux machine.

Client authentication is used to authenticate the client against the server and ECC keys are used to do the authentication for security and convenience sake. **ssh-keygen** is used to create the ECC keys for the authentication; similar to how RSA keys are set up and created.

6.4 Is It Trustable?

The current developments of ECC have already gained favour in many areas of businesses and technology worldwide--becoming increasingly common in IoT surprisingly. ECC's reliability, security and cryptographic strength with smaller key sizes are incredibly appealing.

RSA keys are strong and enough for most businesses' application and environment, but ECC is remarkably more secure. With the developments still ongoing, its security will get stronger on low-powered devices and become even more widespread.

While you could likely continue to keep RSA encryption secure by increasing its key sizes and length which comes with a slower cryptographic performance and speed, ECC appears to be a better tradeoff in this case--**higher security with short and fast keys.**

7 Cryptocurrency

7.1 What is it?

A cryptocurrency is a digital or virtual currency that is secured by cryptography, which makes it nearly impossible to counterfeit or double-spend. Many cryptocurrencies are decentralized networks based on blockchain technology. This structure allows them to exist outside the control of governments and central authorities. They are systems that allow for secure payments alone which are denominated in terms of virtual “tokens”, which are represented by ledger entries internal to the system. They emulate the concept of the real-world signature by using cryptography techniques and the encryption keys.

7.1.1 Cryptocurrency Public Ledger

A cryptocurrency public ledger is a record-keeping system. The ledger keeps the participants’ identity anonymously, their cryptocurrency balances, and a record of all the transactions executed between them.

The transaction details on a cryptocurrency public ledger can be verified and queried by the two transacting participants. However, since this is a decentralised system, no central authority or network participants has the capability to know the identity of the participants. Transactions are allowed and recorded only after suitable verification of the sender’s identity.

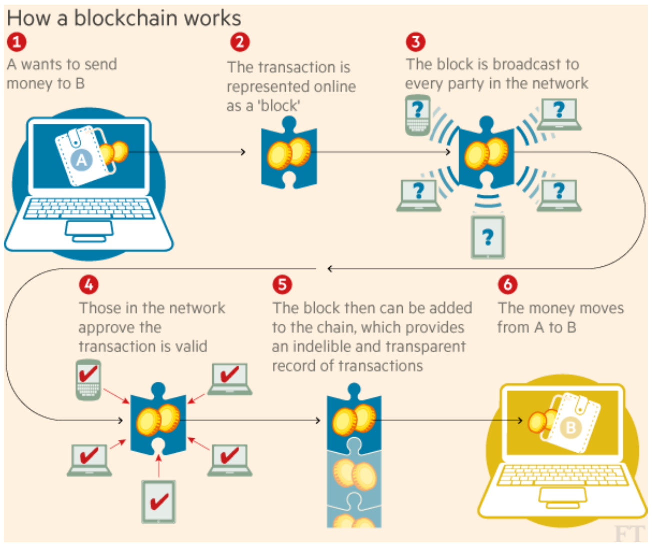
A blockchain is a form of public ledger, which is a series of blocks on which transaction details are recorded after participants have verified and authenticated them. The recording and storage of all confirmed transactions on such public ledgers begin at the creation and start of a cryptocurrency. Once the block is filled to the max with transaction details, the new ones are mined and added to the blockchain by the participants.

Select network participants maintain a copy of the whole ledger on their devices that are connected on the cryptocurrency network. Depending on the participants’ interest and their spread across the globe, the public ledger is distributed as participants connect and contribute to the blockchain network activities keeping it agile and functional.

7.1.2 Cryptocurrency Mining

Cryptocurrency mining is a process in which transactions for various forms of cryptocurrency are verified and added to the blockchain digital ledger. It enforces a chronological order in the blockchain, protects the neutrality of the network, and allows different computers to agree on the state of the system.
Whenever a cryptocurrency transaction is made, the cryptocurrency miner is responsible to make sure that the transaction is authentic with the required information and is updating the blockchain. This involves using computer power to solve complex math problems with

cryptographic hash functions that are interlinked with blocks containing transaction data. These rules are created to prevent the blocks from being modified since should any of the blocks be amended, the following blocks will be invalid.


How a blockchain works
1
A wants to send money to B
2
The transaction is represented online as a 'block'
3
The block is broadcast to every party in the network
4
Those in the network approve the transaction is valid
5
The block then can be added to the chain, which provides an indelible and transparent record of transactions
6
The money moves from A to B
A
B
FT

7.2 How is it applied currently in Linux

There is no form of cryptocurrency transactions on Linux which explains why it does not apply currently in Linux. However, in the future, if cryptocurrency is going to use Linux as a platform to make transactions, I think that it may be applied.

7.3 How is cryptography used in this?

7.3.1 Symmetric Encryption Cryptography

In the cryptocurrency system, when a message is transmitted from one user to another, the message will be encrypted then decrypted once it reaches the designation. By using this method, when data is encrypted, it cannot be understood by anyone who does not have the secret key to decrypt it. Once the message lies in the intended recipient who possesses the key, the algorithm will reverse its action and the content of the message returns to its original and understandable form.

For example, Jennifer wants to send a confidential message to her co-partner, Sarah, who is working on the same project as her. She uses the secret key to encrypt the raw message to make sure that nobody can understand the message. The authorized person who has the secret key has the ability to decrypt the message. If the person who receives the message is not Sarah, the person will not be able to decrypt the message unless he or she has the knowledge of the encryption methodology. If the person receiving the encrypted message is Sarah who is the intended recipient who has the key, she will be able to reverse the action of the encryption and is able to uncover the content of the message.

7.3.2 Asymmetric Encryption Cryptography

In modern cryptocurrency systems, a user's "wallet" or account address, has a public key, while the private key is known only to the owner and is used to sign transactions. Once the private key is verified by the public key, the paired key holder can successfully decrypt the encrypted message. This shows that the message will be authentic, without any form of tampering or anybody from being able to see the content of the encrypted message.

For example, Jennifer wants to send some bitcoins to James but she does not want anyone to tamper with that amount or to let the bitcoins land onto some stranger's hands. Using asymmetric encryption, James' public key will be used to encrypt the transaction and to decrypt it, it will be his private key. This is to ensure that that it will be authentic and only the intended recipient is able to see the content.

7.3.3 Hashing

With this method used in many cryptocurrency systems, it allows the system to prevent fraudulent transactions and double spending of the currency. The transactions in cryptocurrency are taken as input and run though a hashing algorithm which gives an output of a fixed output. Since each block header contains a target hash, once there is a minor change in it, the entire hash value will be totally different from the original value.

7.4 The latest development of Cryptocurrency

7.4.1 Bitcoin

Bitcoin, which was launched in 2009, is a digital currency that offers the promise of lower transaction fees than traditional online payment mechanisms and is operated by a decentralised authority. There are no physical versions of them and everything is verified by a massive amount of computer power and it is currently the world's largest cryptocurrency by market cap.

7.4.2 Litecoin

Litecoin was released in 2011 and it is the most similar to Bitcoin, but it has moved in a more faster pace to develop new innovations which includes faster payment and processes to allow many more transactions to happen. It is a peer to peer Internet currency that enables instant, near-zero cost payments to anyone in the world. Not only that, it is an open-source, global payment network that is fully decentralised without any central authorities.

7.4.3 Ripple

It was founded in 2012 and is a real-time gross settlement system, currency exchange and remittance network. It is built upon a distributed open-source protocol and supports tokens representing fiat currency, cryptocurrency, commodities, etc. Besides that, it can be used to track a variety of transactions, other than cryptocurrencies.

7.4.4 Ethereum

It was launched in 2015. It is a decentralised application that uses the advantage of cryptocurrency and blockchain technology to make it always run as programmed. It can control digital assets in order to create new kinds of financial applications. It is run on ethereum blockchain and coded transactions are stored in a decentralised ledger, the blockchain, and they can be seen for everyone on the network.

7.5 Is It Trustable?

Since every transaction made is recorded and encryption is used to verify them, there are security and safety features inside the network. It is also built using blockchain technology which is a complex, technical process, which is difficult for hackers to tamper or launch an attack.

The transactions also require a two-factor authentication process where one needs to enter a username and password then enter an authentication code that is sent through text to one’s personal phone to be able to have a successful transaction.

For example, when Jennifer wants to do a transaction, she will have to enter her username and password to have her identity. After that, there will be an authentication code for her to make sure that the identity that was given is truly hers by going through this process of verifying her identity.

7.5.1 Disadvantages

Cryptocurrency blockchains are highly secure, but other aspects of a cryptocurrency ecosystem are not immune to the threat of hacking and this is a great host of illegal activities such as money laundering and tax evasion.

8 Blockchain technology

8.1 What is it?

Blockchain Technology, also known as Distributed Ledger Technology (DLT), makes the history of any digital asset unalterable and transparent through the use of decentralization and cryptographic hashing. It is a chain of blocks that are made up of digital pieces of information. These blocks store information about transactions, the people participating in transactions, and information that distinguishes them from other blocks.

8.2 How is it applied currently in Linux

No articles related to Linux

8.3 How is cryptography used in this?

8.3.1 Public-Key Cryptography

Public-Key Cryptography is known as Asymmetric Encryption Cryptography. Once a user creates a wallet on the blockchain, a pair of private and public keys will be generated. The address of the wallet, which is a string of numbers and letters generated from the public key, is public to everyone. On the other hand, the private key which is associated with a wallet is used to prove the ownership and has the capability to control the wallet. Once the transaction is confirmed, it will be written into the ledger and the balance is updated. This shows that there is accountability for every transaction made. However, it requires a signature from the private key of the sending wallet to be valid, which confirms the authenticity and the identity of the user.

8.3.2 Hashing

Once there is an edit in the data, the hash value will change into something different. In blockchain technology, every new block of data contains a hash output of all the data in the previous block. This is for integrity purposes, so if someone were to try to change the data in any block, it would not only alter the hash output of that block's data but every block after it. Once the resulting hashes are different from the original version of the chain, it would reject the change.

8.4 The latest development of Blockchain Technology

Facebook commits to starting a blockchain group and also hints at the possibility of creating its own cryptocurrency in 2018.

Facebook could use blockchain technology to enhance privacy for users who frequent its platforms every day. For instance, blockchain could be implemented to prove a person's identity and could be used to provide users with more control over their data. Facebook could create contracts that stipulate who can receive information on the platform.

IBM develops a blockchain-based banking platform with large banks like Citi and Barclays signing on in 2018.
With blockchain technology, it can speed up processes within the financial industry, making them more efficient and cheaper.

8.5 Is It Trustable?

It reduces risk, stamps out fraud and brings transparency in a scalable way for a variety of uses. Once a block has been added to the end of the blockchain, it is very difficult to go back and alter the contents of the block since each block contains its own hash, along with the hash of the block before it. If any information is changed, the hash code changes as well. It is also difficult to edit and impossible to delete every single block on the blockchain.

Blockchain networks have also implemented tests for computers that want to join and add blocks to the chain. It is not impossible for hackers but there is no point to do so since they would need to control more than half of all the computing power on the blockchain to make an attack. Besides that, the size of the blockchain is so immense that it requires the control over millions of computers just to execute an attack which makes it pointless, not worth the effort, and therefore nearly impossible for anyone to do it.

Applications used in Linux

1 Data Storage

Linux encrypt Filesystem:

- Ecryptfs
- Encfs

eCryptfs is a “pseudo-file system” which provides data and filename encryption on a per-file basis. The term "pseudo-file system" refers to the fact that eCryptfs does not have an on-disk format. It is a file system layer that resides on top of an actual file system. The eCryptfs layer provides encryption capabilities.

eCryptfs intercepts the file operations that are being written to the encrypted file system. The eCryptfs layer adds a header to the metadata of files in the encrypted file system. This metadata describes the encryption for that file, and eCryptfs encrypts file data before it is passed to the encrypted file system. Optionally, eCryptfs can also encrypt filenames.

When mounting a file directory using eCryptfs, there are two passphrases. The login passphrase is used for the user every time they want to mount the encrypted directory. However, if the user wants to auto-mount on login, the passphrase has to be the same password as the login password to the user account.

Other than the login passphrase, there is the mount passphrase. This is used to derive the actual file encryption master key hence it is best to leave it to auto-generate a secure random one. It will be encrypted using the login passphrase and stored in the encrypted form in a directory that was created. Later on, it will automatically be decrypted (“unwrapped”) again in RAM when needed. It is important to make sure that the file does not get lost, otherwise the user might not be able to access the encrypted folder again. One way of preventing that from happening is to run a “ecryptfs-unwrap-passphrase” to see the mount passphrase in unencrypted form and write the passphrase or keep it somewhere safe. The eCryptfs hence provides two different approaches for the accessing of data, one being the on demand (manual key in of passphrase) and auto which is through user login.

EncFS provides an encrypted filesystem in user-space. It runs without any special permissions and uses the FUSE library and Linux kernel module to provide the filesystem interface. It is created on top of an existing filesystem and not replacing an existing filesystem. The Encfs is a demand-based encryption, meaning if i zip up a directory or folder or files, i would need to provide a passphrase every time i would want to access it.

Tools to encrypt/decrypt password-protected files in Linux: (can be installed through a command-line interface)

- GnuPG (usually comes by default, so it is already installed inside the OS)
- Bcrypt (not recommended due to the blowfish cipher algorithm ability to be attacked.)
- Ccrypt (Designed as a replacement of UNIX crypt, ccrypt is a utility for files and streams encryption and decryption. It uses Rijndael cypher.)
- Zip (It is one of the most famous archive formats and it is so famous that we generally call archive files as zip files in day-to-day communication. It uses a pkzip stream cipher algorithm.)
- Openssl (Openssl is a command line cryptographic toolkit which can be used to encrypt messages as well as files.)

As Gzip, bzip2 and tar tools do not support password protection hence there is GnuPG.

2 Data Transmission (Security Data Transmission)

2.1 Overall Design of System Software

It adopts an embedded Linux System, this is because it has characteristics of small size and stable performance. The flow will be from Embedded Linux Operating System to Network Transmission Module to Identity Authentication Module to either Data Encryption Module or Image Watering Module.

2.2 Network Transmission Module

The network transmission module is the basic module of the whole platform network transmission as it is responsible for the network transmission. This will be used as a node server in the network, the corresponding security operations are carried on it. Different transmission modules for file data and digital image improves the efficiency of network transmission.

2.3 Identity Authentication Module

It uses the technology of combining digital certification and signature, adopts the form of simple PKI which can effectively solve the problem of identity authentication of network users. When the TCP connection is established, each embedded node server will send their digital certificate issued by CA and CA root certificate. It will communicate both their digital signatures first using CA after that CA root will validify if it is legal or illegal. If it is considered illegal it will send out the certificate to other parties to check its validity, if it is proven to be legitimate, the connection will be established, and it will begin. The combination of hash functions with a public key algorithm will provide data integrity as well as data authenticity. Therefore, the problem of authentication for network users can effectively be solved.

2.4 Data Encryption Module

It implements the function of encrypting and decrypting of data. It can realize the commonly used symmetric and asymmetric encryption algorithms, provide the protection of data security, and realize the commonly used digest algorithm, provide the protection of data integrity. Data encryption module mainly includes symmetrical encryption sub-module, asymmetrical encryption sub-module, digest sub-module and key anti-quotient sub-module. Module uses OpenSSL standard library function to implement various encryption algorithms.

2.5 Where it is being used?

It is mainly used in OpenSSL standard library functions to understand the data encryption module of the system which ensures security. This is mainly used with all Linux operating systems.

3 Disk Encryption

Disk encryption is often referred to as ‘at-rest-encryption’ in security compliance guides and many other compliance regimes—such as PCI. At rest encryption can be an important factor of system-hardening.

One common practice with respect to Linux servers, is to leave the root unencrypted and only add cryptographic encryption to specific disks that contain sensitive data and files.

There are two main disk encryption capabilities built-in Linux:

1. Device Mapper Crypt (dm-crypt)
2. Linux Unified Key Setup (LUKS)

Dm-crypt

This capability works with:

- Linux kernel v2.6+ and later,
- DragonFly BSD,

Dm-crypt is a transparent disk encryption subsystem within the Linux kernel where the block device based abstraction can be inserted on top of other block devices. The actual encryption utilises the kernel’s crypto API framework and device-mapper subsystem. It is able to encrypt whole disks, removable media, partitions, software RAID volumes, logical volumes and files.

Some Linux distributions support the use of dm-crypt on the root file system. These distributions use initrd to prompt the user to enter a passphrase at the console or insert a smart card prior to the normal boot process of the system.

LUKS

This capability works with any Linux distribution, provided that the respective machine installer includes a **LUKS encryption** option. LUKS utilises the standard on-disk-format and provides not only secure management of multiple user passwords but also facilitates compatibility among the distributions.
LUKS stores all necessary setup information in the partition header which enables the user to transport or migrate his or her data seamlessly.

An example of disk encryption used in the current world is NVIDIA drive OS SDK security development for Linux-based machines.

> *NVIDIA Drive OS delivers a safe and secure execution environment for critical applications, providing services such as secure boot, security services, firewall and over-the-air updates.*

They have the /home directory encrypted and the user data available when the system is running. The user /home partition is mounted on a separate disk partition and block-level encryption is enabled for that disk.

*For more information on NVIDIA Drive OS SDK security development, refer to the **Appendix** section of this report.*

4 Endpoint Encryption

Endpoint encryption protects the operating system from any form of attacks such as installing attacks that can install a keylogger and corrupting boot files and locks files stored on laptops, servers, tablets, and other endpoints to prevent unauthorised users from accessing the data.

4.1 Full-disk encryption

Full-disk encryption solutions often include pre-boot authentication capabilities which require the end-user to first authenticate to the device before unlocking the device. This means that the device will not boot or the removable device cannot be read until authentication is completed. Two main capabilities built in Linux are LUKS and dm-crypt. (Full reference at 3 Disk Encryption)

4.2 Folder encryption

Folder-level encryption is helpful if one does not want to encrypt the entire disk but just wants to hide the content of the folder from another party.
Folder encryption is accomplished by using a user-specific key which has the advantage of preventing access by other users on the same machine.

4.3 Transparent file encryption

Files that have this encryption are usually encrypted based on a policy that identifies sensitive data such as credit card number. This is well positioned to prevent data loss by ensuring that the data remains encrypted, be it on the server, at rest, in motion or on the endpoint.
In addition, there is often a policy information embedded in the file that controls the file permission. This policy is embedded in the file itself so that, no matter where the file is accessed from, the policy is enforced.

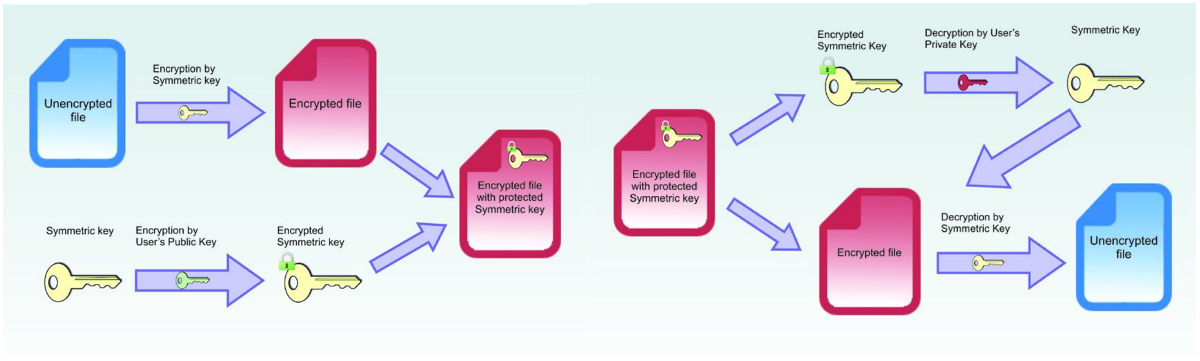
With this type of encryption, it requires a transparent file encryption client on every device, so it knows how to read and enforce the policy and can decrypt the file to allow the correct client to read the file. This means that the encryption and decryption will be automatically without requiring the user to do anything.

4.3.1 Encryption Process

This feature executes a special driver of the file system, which carries out decryption of files when certain applications request it as well as re encryption after the application is finished with the files. The driver also encrypts all new files added into a secure folder.

Firstly, the file will be encrypted using the AES algorithm and a 256 bit symmetric key generated randomly by the program. The symmetric key is RSA encrypted via the user’s Public Key, which can be up to 8192 bits in length and is stored in an alternate NTFS data stream.

When an appropriate program is used to open the file, it is automatically decrypted. The symmetric key stored in ADS is decrypted using the user's Private Key and the desired file is decrypted using the symmetric key.


Unencrypted file
Encryption by Symmetric key
Encrypted file
Symmetric key
Encryption by User's Public Key
Encrypted Symmetric key
Encrypted file with protected Symmetric key
Encrypted file with protected Symmetric key
Encrypted Symmetric Key
Decryption by User's Private Key
Symmetric Key
Encrypted file
Decryption by Symmetric Key
Unencrypted file

- **Filesystem-level encryption (fscrypt)**
  It is a library which filesystems can hook into to support transparent encryption of files and directories.

  fscrypt operates at the filesystem level rather than at the block device level. This allows it to encrypt different files with different keys and to have unencrypted files on the same filesystem. Besides that, it is also useful since each user's data-at-rest is cryptographically isolated from the others. However, it does not encrypt filesystem metadata except for filenames.

  fscrypt is integrated directly into supported filesystems such as ext4. This allows encrypted files to be read and written without caching both the decrypted and encrypted pages in the pagecache, which requires lesser memory used. fscrypt API can also be used by unprivileged users, with no need to mount anything.

- **Gocryptfs**
  Gocryptfs is a Filesystem in Userspace (FUSE)-mounted file-level encryption program. FUSE-mounted means that the encrypted files are stored in a single directory tree that is mounted which allows any user to do the mount. This is because it encrypts at the file level, synchronisation operations that copy the files can work efficiently on each file.

  When the user uses this in its normal mode, the files will be stored on the disk in an encrypted format. However, when the user mounts the encrypted files, he will get unencrypted access to his files, which means that all the programs and applications inside the computer will have access to the file. Changes of the files and deletions are reflected in real-time in the encrypted version of the filesystem stored on your disk.

Conclusion

As we lean towards a society where the amount of automated information resources are increasing rapidly, cryptography will continue to increase in importance as a security mechanism. As time progresses, technology advances and this leads to issues with older platforms where a rise of vulnerabilities and machines’ exploitations occur. Hence, we require constant growth along with technology to better improve security in order to ensure the safety of internet users.

The word ‘cryptography’ is a very broad word. Cryptography is not only being used in industries but also in our daily lives. The essential need for security compliance and incorporating confidentiality, validation, integrity and nonrepudiation is needed. To give such security, most frameworks and applications in the Linux machines need to use noteworthy cryptographic algorithms. This is where Cryptographic Algorithms, Disk Encryption, Endpoint Encryption and Data Transmission comes into play.

The other main factor in developing cryptographic applications is to trim down the operational speed of the algorithm, and this is where Elliptic Curve Cryptography comes into play with their smaller key sizes. The selection of a cryptographic system depends on the type of tasks that are needed to be done and the situation, and that is where Quantum and Post Quantum Technology, Cryptography in Cloud Computing, Cryptocurrency and Blockchain Technology have arised.

Presently, although some applications and features are not widely distributed in every Linux distribution, in the near future, cryptography would be heavily used since many countries are starting to go towards the digital era as of now and more highly advanced technology are surfacing. For example, many parts in China have digitized payment through the use of QR code scanners on wechat. In Singapore, we can easily pay our fees through online transactions instead of the traditional method of notes and coins.

Even though cryptography has played an important role in many industries, cyber crimes will be the next top crimes around the world. For example, in the finance industry, with the increase of usage of cryptocurrency, the number of corruption and illegal transactions will be the next top crimes around the world in the next few years. This shows the power of cryptography and how it affects the future.

Appendix

NVIDIA Drive OS SDK Linux-based Disk Encryption

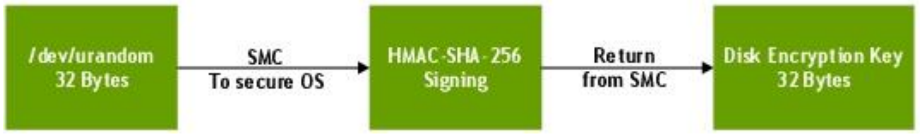
NVIDIA implements the usage of **dm-crypt** kernel module in their security which is inserted between the disk driver and the file system to encrypt and decrypt the data (blocks). Dm-crypt is managed with cryptsetup which sets up the disk encryption based on the CMCrypt kernel module. NVIDIA security implements the setting up of encrypted data disk as a LUKS partition and configures it with a random passphrase. The basic configuration of NVIDIA cryptsetup can be found in this documentation: https://gitlab.com/cryptsetup/cryptsetup.

In addition, the cipher mode used for NVIDIA disk encryption security for Linux-based machine is aes-cbc-essiv:sha256, where the cipher generator consists of 3 parts--cipher-chain-mode-IV generator. The base documentation of the algorithm can be found in https://wiki.archlinux.org/index.php/Disk_encryption#Ciphers_and_modes_of_operation.

NVIDIA disk encryption key management for the encrypted file system (EFS) has two types of keys provided:

1. Disk Encryption Key (DEK), also known as master key. It ensures that data flowing between the file system and the disk is encrypted and decrypted.
2. Passphrase/Password pattern/string input by the user to setup disk encryption is later used to enter, unlock and decrypt user's data. *The system sets and stores a passphrase used on every boot.*

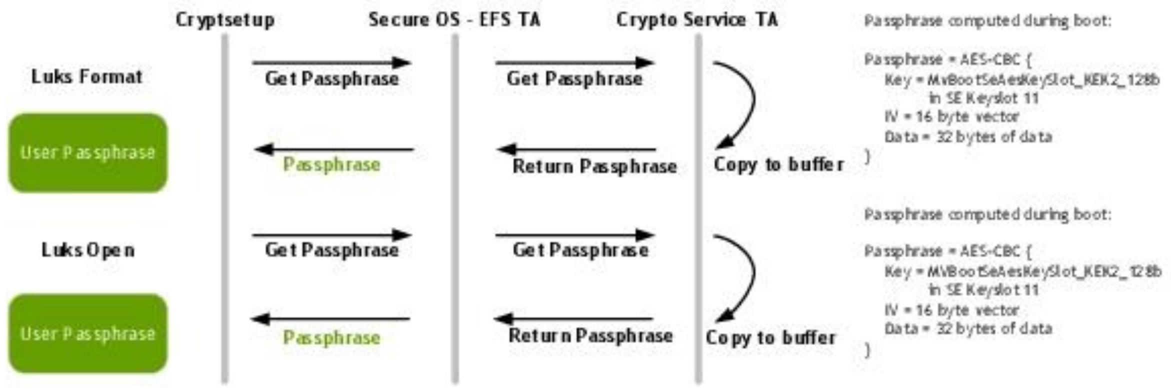
NVIDIA's DEK is derived from a /dev/urandom buffer with 32 bytes of random data.



The random buffer is then signed with **HMAC-SHA-256** algorithm to enhance security before returning back to cryptsetup after adding randomness. The secret key for the Hash Message Authentication Code (HMAC) tag is derived from the EKS blob provisioned into the device as per the image above.

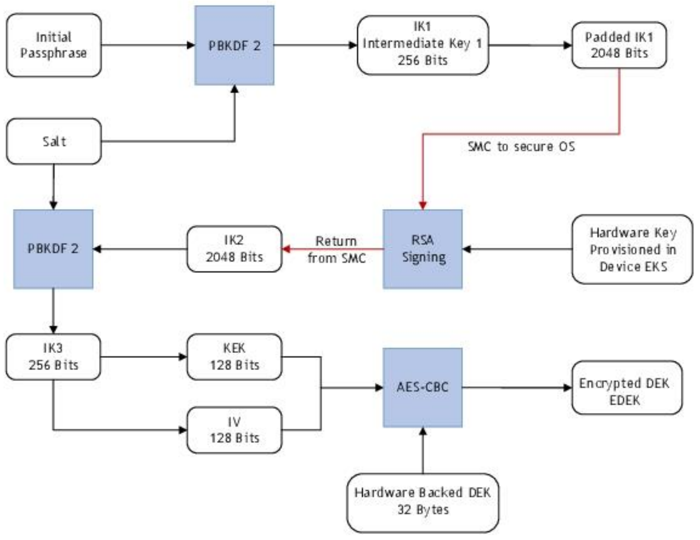
The initial passphrase is obtained by performing AES-CBC-128 fixed known buffer encryption--also known as IV. The key used for the encryption is stored in Key Encryption Key 2 (KEK2) hardware fused provisioned by Original Equipment Manufacturer (OEM).

Below is a picture of how one could be able to obtain the initial passphrase:



Additionally, a keyfile will be created during the 1st booting of the machine with a 32 byte random vector from /dev/urandom which is accessible by the kernel. On the other hand, the passphrase in the keyfile will be used as a backup option for the initial passphrase if obtaining the passphrase is *not supported* on the user machine's platform.
Note: The keyfile is stored in the **/usr** partition of the root-file system.

To derive the hardware backed passphrase, the initial passphrase is run through the machine. Below are the steps the machine will take:



The cryptsetup encrypts the hardware backed DEK using the hardware backed passphrase and stores the encrypted DEK onto the encrypted partition.

References

[Cheryl]
https://gpgtools.tenderapp.com/kb/how-to/introduction-to-cryptography
https://www.kernel.org/doc/html/v4.10/crypto/index.html
https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/6/html/storage_administration_guide/ch-efs
https://www.esat.kuleuven.be/cosic/publications/article-2337.pdf
https://www.laits.utexas.edu/~anorman/BUS.FOR/course.mat/SSim/life.html
https://www.howtoforge.com/tutorial/encrypt-your-data-with-encfs-on-ubuntu/
https://www.kernel.org/doc/html/v4.10/crypto/userspace-if.html
https://mchehab.fedorapeople.org/kernel_docs_latex/crypto-api.pdf
https://study.com/academy/lesson/what-is-cryptography-definition-uses.html
https://www.sciencedirect.com/topics/computer-science/cryptographic-algorithm
https://www.intechopen.com/books/recent-advances-in-cryptography-and-network-security/introductory-chapter-recent-advances-in-cryptography-and-network-security
http://www.crypto-it.net/eng/simple/index.html
https://www.go4expert.com/articles/how-ciphers-work-t415/
https://www.cryptomathic.com/news-events/blog/summary-of-cryptographic-algorithms-according-to-nist#:~:text=Symmetric%2DKey%20Algorithms%20for%20Encryption,used%20for%20encryption%2Fdecryption%20services.
https://www.kernel.org/doc/html/v4.10/crypto/index.html
https://unixmen.com/encryption-methods-linux/
https://www.esat.kuleuven.be/cosic/publications/article-2337.pdf
https://unixmen.com/encryption-methods-linux/
https://www.ukessays.com/essays/information-technology/the-evolution-of-cryptography-information-technology-essay.php
https://wiki.archlinux.org/index.php/ECryptfs
https://crypto.stackexchange.com/questions/14654/digital-signature-using-symmetric-key-cryptography
https://www.freecodecamp.org/news/md5-vs-sha-1-vs-sha-2-which-is-the-most-secure-encryption-hash-and-how-to-check-them/

[Zachary]
ww.techrepublic.com/blog/it-security/how-quantum-cryptography-works-and-by-the-way-its-breakable/
https://fedtechmagazine.com/article/2020/03/what-difference-between-quantum-cryptography-and-post-quantum-cryptography-perfcon
https://en.wikipedia.org/wiki/Post-quantum_cryptography

https://en.wikipedia.org/wiki/Quantum_computing
https://en.wikipedia.org/wiki/Quantum_cryptography
https://www.fireeye.com/blog/threat-research/2020/04/kerberos-tickets-on-linux-red-teams.htm
https://www.bbc.com/news/science-environment-50154993#:~:text=Google%20says%20an%20advanced%20computer,supercomputer%2010%2C000%20years%20to%20complete.
https://www.researchgate.net/publication/335080509_Research_on_the_Security_Data_Transmission_based_on_Linux

[Yi Ching]
[Cloud Computing]
https://www.cryptomathic.com/news-events/blog/cryptography-the-next-10-years-part-3
https://www.hcltech.com/technology-qa/how-does-cloud-computing-work
https://searchcloudcomputing.techtarget.com/definition/Software-as-a-Service
https://sea.pcmag.com/networking-communications-software/2919/what-is-cloud-computing
https://www.vxchnge.com/blog/different-types-of-cloud-computing
https://www.sciencedirect.com/topics/computer-science/cloud-deployment-model
https://www.bigcommerce.com/blog/saas-vs-paas-vs-iaas/
https://blog.equinix.com/blog/2019/09/30/deconstructing-distributed-security-why-use-an-hsm-as-a-service-model/
https://link.springer.com/content/pdf/10.1007%2F3-540-39799-X_31.pdf
https://cs.brown.edu/~seny/slides/cc-RCLE11.pdf
https://wiki.archlinux.org/index.php/Data-at-rest_encryption
https://en.wikipedia.org/wiki/Hardware_security_module
https://en.wikipedia.org/wiki/FIPS_140-2 - FIPS 140-2 Certification
https://www.cryptomathic.com/news-events/blog/common-criteria-helping-to-choose-the-right-hsm - Common Criteria Certification
https://hsm.utimaco.com/solutions/compliance/certifications/common-criteria-cc/ - Common Criteria Certification
https://www.linuxjournal.com/content/rest-encryption
https://www.winmagic.com/blog/linux-servers-and-encryption/
http://www.redbooks.ibm.com/redpapers/pdfs/redp5535.pdf - IBM LinuxOne
https://www.ibm.com/security/cryptocards/hsms - IBM LinuxOne
[Elliptic Curve Cryptography]
https://en.wikipedia.org/wiki/Elliptic-curve_cryptography
https://www.nist.gov/news-events/events/2015/06/workshop-elliptic-curve-cryptography-standards
https://wizardforcel.gitbooks.io/practical-cryptography-for-developers-book/content/asymmetric-key-ciphers/elliptic-curve-cryptography-ecc.html
https://csrc.nist.gov/csrc/media/events/workshop-on-elliptic-curve-cryptography-standards/documents/presentations/session6-bos-joppe.pdf - Fig. 9
https://www.cisco.com/c/dam/en_us/solutions/industries/docs/gov/CIS-44830_Suite-B_Whitepaper_1c-hi_res.pdf - Fig 11
https://hackernoon.com/what-is-the-math-behind-elliptic-curve-cryptography-f61b25253da3 - Fig 12

https://andrea.corbellini.name/2015/05/17/elliptic-curve-cryptography-a-gentle-introduction/
https://www.globalsign.com/en/blog/elliptic-curve-cryptography
https://www.keycdn.com/support/elliptic-curve-cryptography
https://www.linuxjournal.com/content/elliptic-curve-cryptography
https://link.springer.com/referenceworkentry/10.1007%2F978-1-4419-5906-5_245
https://www.microsoft.com/en-us/research/wp-content/uploads/2013/11/734.pdf?from=http%3A%2F%2Fresearch.microsoft.com%2Fpubs%2F204914%2F734.pdf - Bitcoin
[Disk Encryption]
https://wiki.gentoo.org/wiki/Dm-crypt
https://guardianproject.info/archive/luks/
https://docs.nvidia.com/drive/drive_os_5.1.6.1L/nvvib_docs/index.html#page/DRIVE_OS_Linux_SDK_Development_Guide/Windows%20Systems/security_disk_encryption_lnx.html#
https://en.wikipedia.org/wiki/Dm-crypt - NIVIDA

[Yun Zhen]
https://www.cnbc.com/2018/07/30/ibm-trials-blockchain-platform-aimed-at-banks.html
https://linuxinsider.com/story/cryptocurrency-os-makes-it-easy-to-buy-and-spend-digital-cash-86201.html
https://crushcrypto.com/cryptography-in-blockchain/#:~:text=Cryptography%20is%20an%20integral%20part,blockchains%20to%20be%20more%20efficient.
https://techbeacon.com/security/3-endpoint-encryption-strategies-you-need-know
https://my.eng.utah.edu/~nmcdonal/Tutorials/EncryptionResearchReview.pdf
https://www.investopedia.com/terms/c/cryptocurrency.asp
https://www.investopedia.com/tech/explaining-crypto-cryptocurrency/
https://www.cryptomathic.com/news-events/blog/symmetric-key-encryption-why-where-and-how-its-used-in-banking
https://www.investopedia.com/terms/h/hash.asp#:~:text=A%20hash%20is%20a%20function,to%20blockchain%20management%20in%20cryptocurrency.
https://www.kaspersky.com/resource-center/definitions/what-is-cryptocurrency
https://www.investopedia.com/terms/b/bitcoin.asp
https://litecoin.org/
https://www.investopedia.com/tech/whats-difference-between-bitcoin-and-ripple/
https://ethereum.org/what-is-ethereum/
https://www.linuxjournal.com/content/blockchain-part-i-introduction-and-cryptocurrency
https://builtin.com/blockchain
https://www.investopedia.com/terms/b/blockchain.asp
https://www.linuxjournal.com/content/blockchain-part-ii-configuring-blockchain-network-and-leveraging-technology
https://www.investopedia.com/news/facebook-working-blockchain-why/
https://www.linux.com/training-tutorials/rookies-guide-ethereum-and-blockchain/
https://www.thegeekstuff.com/2012/10/gnupg-basics/

https://www.webopedia.com/TERM/C/cryptocurrency-mining.html#:~:text=Cryptocurrency%20mining%2C%20or%20cryptomining%2C%20is,to%20the%20blockchain%20digital%20ledger.
https://www.weforum.org/agenda/2017/07/blockchain-the-ledger-that-will-record-everything-of-value/
https://www.telegraph.co.uk/technology/0/cryptocurrency/#:~:text=How%20do%20cryptocurrencies%20work%3F,and%20held%20by%20currency%20holders.
https://www.genesis-mining.com/how-cryptocurrency-works
https://www.investopedia.com/tech/what-cryptocurrency-public-ledger/#:~:text=Key%20Takeaways%3A,transactions%20executed%20between%20network%20participants.
https://tradeix.com/distributed-ledger-technology/
https://bitcoin.org/en/how-it-works
https://opensource.com/article/19/8/how-encrypt-files-gocryptfs
https://www.reddit.com/r/crypto/comments/4fpr48/announcing_gocryptfs_v09_transparent_file/
https://nuetzlich.net/gocryptfs/
https://cybersafesoft.com/post.php?id=5273

Other links:
http://index-of.co.uk/Hacking-Coleccion/
https://csrc.nist.gov/Projects/Cryptographic-Research)
https://www.cryptomathic.com/news-events/blog/cryptography-the-next-10-years-part-1
https://www.cryptomathic.com/news-events/blog/cryptography-the-next-10-years-part-2