***Assignment: Cryptography Analysis and Implementation***

***NAME:* Sankalpa Swain**

***REG.NO*:20BCN7012**

***CAMPUS:* VITAP UNIVERSITY**

***Symmetric cryptography:***

Using the same cryptographic keys for encryption and decryption is known as symmetric-key cryptography. Here, a single secret key is shared by the sender and all recipients. With the use of a certain encryption key, the plaintext communications are converted into cypher text. The shared secret key can be used by the recipient to decrypt the message using the same encryption key.

**Ex:**

Advanced Encryption Standard (AES)

Data Encryption Standard (DES)

Triple Data Encryption Standard (Triple DES)

International Data Encryption Algorithm (IDEA)

TLS/SSL protocol

**I chose triple DES.**

***Triple DES:***

By employing three runs of the DES instead of one, the Triple Data Encryption normal (DES) algorithm offers higher security than the normal DES method. Standard DES was discovered to have poorer security than AES. The keys are 56 bits long. Man in the middle attacks can be avoided thanks to the development of Triple DES.

As the shortcomings of standard DES became increasingly obvious, 3DES was employed in a variety of applications. Before the popularity of AES, it was one of the most widely used encryption techniques.

It is recommended to adopt another algorithm because 3DES will be deprecated in the next years. For many purposes, keying option one is still regarded as safe, although there aren't many compelling arguments to favor it over an alternative like AES.

***Strength and advantages:***

In 1998, DES was strengthened with the introduction of 3DES since brute-forcing 56-bit keys was now possible. 3DES is just three DES encryptions, either with two distinct keys for a practical key size of 112 bits or with three different keys for a practical key size of 168 bits.

The banking sector makes extensive use of the encryption technique 3DES.

simpleness of hardware and software implementation, as well as the wide use of its cryptographic protocols and libraries.

***Weakness:***

If 3DES is used to encrypt large quantities of data with the same key, block collision attacks may be possible due to the tiny block size of 64 bits. The Sweet32 attack demonstrates how TLS and OpenVPN may be used to exploit this.

For a complete assault, a practical Sweet32 attack against 3DES-based cipher-suites in TLS requires 236.6 blocks (785 GB).

Since version 1.1.0 (August 2016), 3DES has been excluded by default and is regarded as a "weak cypher" by OpenSSL.

**Some real-world examples of its implementations included:**

Microsoft Office

Firefox

EMV payment systems

***Asymmetric cryptography***

It also goes by the name "public-key cryptography," and it employs several cryptographic keys for both encryption and decoding. A private key and a public key are shared by the sender and the recipient.

All parties involved in communications with the sender are given access to the public key.

Each side is kept in the dark about the private key.

These private key and public key pairs are mathematically connected, but the public key cannot create the private key.

For message authentication, public key cryptography is frequently employed in digital signatures. To verify the legitimacy of their messages, senders digitally sign them using their private keys. As a result, the recipient is fully aware that the sender is a reliable third party.

A few instances include:

Elliptical Curve Cryptography (ECC)

Rivest Shamir Adleman (RSA)

The Diffie-Hellman exchange method

TLS/SSL protocol

***RSA algorithm (Rivest-Shamir-Adleman):***

The foundation of a cryptosystem is a collection of cryptographic algorithms used for certain security functions or objectives. These algorithms support public key encryption, which is frequently used to safeguard sensitive data, especially when it is transferred over an unsecured network like the internet.

The Massachusetts Institute of Technology's Ron Rivest, Adi Shamir, and Leonard Adleman published the first public description of RSA in 1977. However, the 1973 development of a public key method by British mathematician Clifford Cocks was kept secret by the United Kingdom's GCHQ until 1997.

Asymmetric encryption like RSA employs two unique but connected keys. Both the public and the private keys can encrypt a message using RSA cryptography. A message is decrypted using the opposite key to that which was used to encrypt it.

**How does it work?**

Alice generates her RSA keys by selecting two primes: p=11 and q=13. The modulus is n=p×q=143. The totient is n ϕ(n)=(p−1) x(q−1) =120. She chooses seven for her RSA public key e and calculates her RSA private key using the Extended Euclidean algorithm, which gives her 103.

Bob wants to send Alice an encrypted message, M, so he obtains her RSA public key (n, e) which, in this example, is (143, 7). His plaintext message is just the number 9 and is encrypted into ciphertext, C, as follows:

Me mod n = 97 mod 143 = 48 = C

When Alice receives Bob's message, she decrypts it by using her RSA private key (d, n) as follows:

Cd mod n = 48103 mod 143 = 9 = M

key strength and advantages:

Security based on the computational difficulty of factoring large composite numbers.

Public-key cryptography enables secure communication without a shared secret key.

Widely adopted and standardized, with applications in key exchange and digital signatures.

**Strong security:**

Based on the difficulties of factoring big composite numbers, RSA encryption offers robust security, making it challenging for unauthorized parties to decrypt encrypted data.

Public-key infrastructure: RSA encryption uses a public-key system, which dispenses with the necessity for a shared secret key to enable safe communication and key exchange.

Applications: Because RSA encryption is extensively used and standardized, it may be used in a broad range of systems, including digital signatures, secure data transfer, and other cryptographic protocols.

**Weakness:**

**Key size requirements:**

For the same security levels, RSA encryption requires bigger key sizes than symmetric encryption techniques. The key size exponentially expands as the required level of security rises, which might strain storage and compute capacity.

RSA encryption is susceptible to assaults from quantum computers, which have the capability to factor enormous numbers far more quickly than classical computers. If quantum-resistant solutions are not adopted, RSA encryption may become less secure as quantum computing develops.

**Known vulnerability:**

Its susceptibility to attacks based on the factorization of big composite numbers is one of the recognized flaws in RSA encryption. An attacker may be able to gain the private key and unlock encrypted data if they are successful in factorizing the modulus used by RSA. Factoring huge numbers is a computationally challenging issue, which is the foundation of RSA's security. However, if the key size employed is insufficiently big, developments in computing power and factoring techniques, such as the General Number Field Sieve (GNFS), may pose a danger to the security of RSA encryption.

**Real world examples:**

**Secure online transactions:** RSA encryption is used in e-commerce platforms, online banking systems, and payment gateways to secure sensitive information, such as credit card details, during online transactions.

**Secure messaging and chat applications:** Many messaging and chat applications, including popular ones like WhatsApp and Signal, employ RSA encryption to protect the confidentiality of messages and ensure secure communication between users.

**Virtual private networks (VPNs**): RSA encryption is utilized in VPNs to establish secure connections between remote users and private networks, safeguarding data transmission and ensuring privacy.

and in digital certificates.

**Hash functions:**

Hash functions compute a fixed-length hash value or a “fingerprint” on the plain text message. These hashes are unique to each plaintext. Therefore, this type of cryptography does not use a cryptographic key. Hash functions help ensure data integrity between communicating parties. If the hash produces the same output, it indicates that the information has not been altered, compromised or damaged.

Hash functions are used in many cryptographic algorithms and protocols, including MAC algorithms, digital signature algorithms, and authentication protocols.

**Some of the most common hashing algorithms include:**

SHA-1

SHA-2

SHA-3

MD5

Whirlpool

Blake 2

Blake 3

Sha-256

**SHA-256 (Secure Hash Algorithm 256-bit):**

The SHA-2 family of hash algorithms includes SHA-256. It offers a high level of security by producing a 256-bit hash result. It is frequently utilized in many different applications, such as password storage, digital signatures, and blockchain technology.

**Briefly explain how the algorithm works?**

Message Padding: In order to meet the algorithm's requirement that the input message be a multiple of 512 bits, the message is padded.

Initialization: An initial hash value (also known as the "initial state") and a set of initial constants, sometimes known as "hash constants" or "round constants," are defined.

Processing of the message: Blocks of 512 bits are used to split the padded message. The algorithm then goes over each block in turn.

Compression Function: To iteratively update the hash value, each block is further broken into 32-bit words. These words are then subjected to a number of operations, including bitwise logical functions, modular addition, and logical shifts.

Rounds: The compression algorithm is composed of a number of rounds (64 rounds for SHA-256), each of which uses a distinct set of constants and various operations. The hash function is protected from cryptographic assaults thanks to these actions, which guarantee a high level of dispersion and confusion.

Final Hash Value: After processing all blocks, the updated hash values from each round are concatenated to get the final hash value.

The 256-bit hash value that is produced is exclusive to the input message. A drastically different hash value will result from even a slight modification to the input message. Due to this characteristic, SHA-256 may be used for a variety of tasks, including creating digital signatures, ensuring data integrity, and safely storing passwords.

**Discuss the key strengths and advantages of the algorithm:**

This hashing algorithm is a variant of the SHA2 hashing algorithm, recommended and approved by the National Institute of Standards and Technology (NIST). It generates a 256-bit hash value. Even if it’s 30% slower than the previous algorithms, it’s more complicated, thus, it’s more secure.

**Strong security:** SHA-256 offers a high level of security and collision resistance. It is designed to be resistant to various cryptographic attacks, including pre-image attacks, second pre-image attacks, and collision attacks. The 256-bit output size provides a large search space, making it computationally infeasible to find two different inputs that produce the same hash value.

**Widely adopted and standardized:** SHA-256 is a widely adopted and standardized hash function. It is implemented in many cryptographic libraries and systems, making it compatible with a wide range of applications. Its popularity and extensive analysis by the cryptographic community contribute to its reputation and security.

**Identify any known vulnerabilities or weaknesses:**

One of the main weaknesses of SHA-256 (and other conventional cryptographic hash algorithms) is that they are vulnerable to attacks from quantum computers. The mathematical underpinnings of SHA-256's security might be compromised by quantum computers, rendering the algorithm unsecure. It is necessary to switch to quantum-resistant hash functions as quantum computing technology develops.

Limitations on collision resistance: While computationally impossible in practice, discovering collisions in SHA-256 is theoretically possible owing to the birthday paradox. The likelihood of discovering a collision rises with an adequate number of hash operations. This restriction emphasizes the significance of routinely reviewing and maybe switching to better hash algorithms in order to preserve long-term security.

**Provide real-world examples of where the algorithm is commonly used.**

Cryptocurrency: SHA-256 is the underlying hash function used in Bitcoin and many other cryptocurrencies. It is used for mining, where miners perform computational work to find a hash value below a certain target, ensuring the integrity and security of the blockchain.

Password Hashing: SHA-256, along with other iterations of SHA-2, is often used for secure password hashing. When storing user passwords, SHA-256 helps protect sensitive information by producing a fixed-length hash representation that is difficult to reverse-engineer.

Digital Signatures: SHA-256 is commonly used in digital signature algorithms, such as the Digital Signature Algorithm (DSA) and the Elliptic Curve Digital Signature Algorithm (ECDSA). It helps ensure the authenticity and integrity of digital documents, messages, and software updates.

**IMPLEMENTATION:**

Scenario: We want to securely store user passwords in a database by hashing them using SHA-256. This ensures that even if the database is compromised, the original passwords remain secure.

import hashlib

def hash\_password(password):

# Create a SHA-256 hash object

sha256\_hash = hashlib.sha256()

# Convert the password string to bytes and update the hash object

sha256\_hash.update(password.encode('utf-8'))

# Get the hexadecimal representation of the hashed password

hashed\_password = sha256\_hash.hexdigest()

# Return the hashed password

return hashed\_password

# Sample password to be hashed

password = "ITS\_SHAENCRYPTIONSECURITY"

# Hash the password using SHA-256

hashed\_password = hash\_password(password)

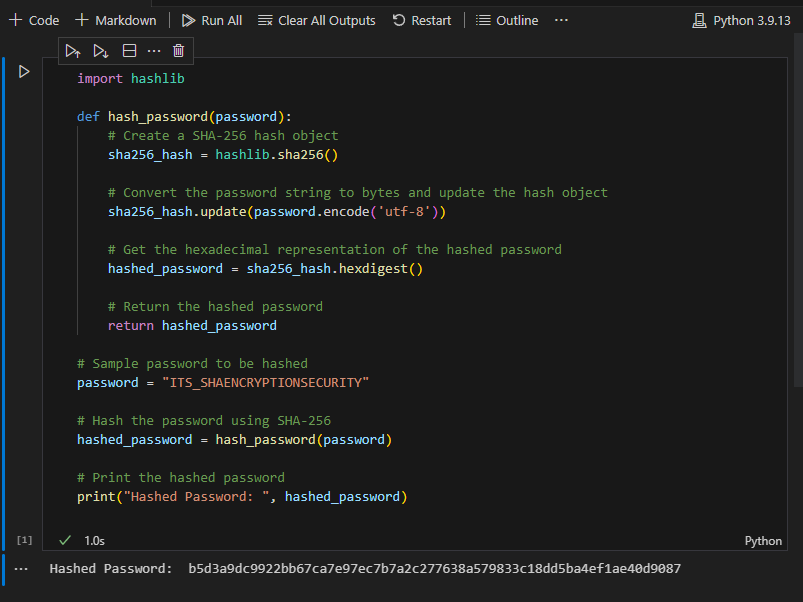
# Print the hashed password

print("Hashed Password: ", hashed\_password)

**Results**:

When you run the code, it will output the hashed password using the SHA-256 algorithm.

Refer the below pic.



**Security analysis:**

**Potential threat:**

The development of quantum computers is one factor that might endanger the SHA-256 algorithm. The mathematical foundations upon which the security of SHA-256 is based are susceptible to attack by quantum computers.

Currently, factoring huge numbers is impractical on classical computers because of the processing complexity involved. However, since they can take advantage of quantum phenomena like superposition and entanglement, quantum computers may be able to carry out some tasks, such integer factorization, a lot quicker.

The security of SHA-256 and other cryptographic algorithms based on related ideas might possibly be compromised if sufficiently effective quantum computers become accessible. This would make it possible for an attacker to quickly calculate collisions, pre-image assaults, and other flaws that can jeopardize the security and integrity offered by SHA-256.

**Counter measures and best practices:**

Moving to post-quantum cryptography: Follow developments in this field and switch to quantum-resistant algorithms when they become standardized.

When hashing passwords, use salting and iterations: Use methods like salting and iterations to strengthen password security and defend against brute-force attacks and precomputed hash tables.

Make careful to maintain keys securely: Use key derivation functions to create secure keys from shared secrets or passwords, generate strong and random keys, store and safeguard them securely, and adhere to suggested key rotation procedures.

**What are its limitations:**

Vulnerability to Quantum Computing: The mathematical foundations that underpin the security of the SHA-256 algorithm are susceptible to assaults by quantum computers, which have the potential to compromise that security. The security of SHA-256 and other comparable algorithms could be jeopardized as quantum computer technology develops, necessitating the use of quantum-resistant cryptographic methods.

Deterministic Output: For a given input, SHA-256 consistently generates the same hash value. The use cases where non-deterministic behavior is required, such as producing unique IDs or random values, may be restricted by this attribute, despite the fact that it is desired for data integrity verification.

Fixed Block and Hash Sizes: The fixed block size for SHA-256 is 512 bits, while the fixed hash size is 256 bits. This may be constricting when dealing with bigger data files or when it's necessary to be compatible with systems that employ various hash sizes. To handle bigger inputs or various output requirements, it can be necessary to take extra processes, such as chunking or truncation.

**Conclusion:**

As a result, SHA-256 is a popular and reliable cryptographic hash function that provides good security and collision resistance for the majority of real-world applications. It offers a dependable way to create digital signatures, check the accuracy of data, and save hashed passwords safely.

It's crucial to recognize that SHA-256 has its limits. Its deterministic nature might not be ideal for situations needing non-deterministic behavior, and it is susceptible to possible assaults by quantum computers. Additionally, dealing with bigger data sets or needing compatibility with alternative hash sizes may be difficult due to its fixed input and output widths.

Despite these drawbacks, SHA-256 is still a reliable and popular algorithm. It is essential to keep up with new developments in the field of cryptography and to be ready to switch to post-quantum cryptographic algorithms as needed. The usage of SHA-256 can offer robust security for data integrity and other cryptographic applications by adhering to recommended practices, such as effective key management, safe implementation, and frequent security audits.

**Resources:**

<https://www.techtarget.com/searchsecurity/tip/Expert-advice-Encryption-101-Triple-DES-explained#:~:text=The%20Triple%20DES%20encryption%20process,a%20last%20time%20with%20K3>.

<https://www.tutorialspoint.com/cryptography/triple_des.htm>

<https://www.comparitech.com/blog/information-security/3des-encryption/>

<https://cyberhoot.com/cybrary/3des-encryption/>

<https://www.splunk.com/en_us/blog/learn/triple-des-data-encryption-standard.html>

<https://www.simplilearn.com/tutorials/cyber-security-tutorial/sha-256-algorithm#:~:text=CybersecurityExplore%20Program-,What%20is%20the%20SHA%2D256%20Algorithm%3F,strength%20against%20brute%20force%20attacks>.

<https://sectigostore.com/blog/sha-256-algorithm-explained-by-a-cyber-security-consultant/>

<https://en.wikipedia.org/wiki/SHA-2>

<https://codesigningstore.com/what-is-the-best-hashing-algorithm#:~:text=To%20protect%20passwords%2C%20experts%20suggest,your%20best%20hashing%20algorithm%20choices>.

<https://www.splunk.com/en_us/blog/learn/cryptography.html>

<https://en.wikipedia.org/wiki/Data_Encryption_Standard>

<https://www.techtarget.com/searchsecurity/definition/RSA#:~:text=RSA%20is%20a%20type%20of,is%20used%20to%20decrypt%20it>.

<https://www.simplilearn.com/tutorials/cryptography-tutorial/rsa-algorithm>

<https://www.simplilearn.com/tutorials/cryptography-tutorial/rsa-algorithm>

<https://www.simplilearn.com/tutorials/cryptography-tutorial/rsa-algorithm>

<https://www.simplilearn.com/tutorials/cryptography-tutorial/rsa-algorithm>

<https://www.javatpoint.com/rsa-encryption-algorithm>

<https://www.javatpoint.com/rsa-encryption-algorithm>