**Converted from: UNIT II NETWORK HEENA.pdf**

**--- Page 1 ---**

UNIT II ASY M METRIC CRYPTOGRAPHIC ALGORITHM & MESSAGE AUTHENTICATION Public Key Encryption/Asymmetric key Cryptography Public key cryptography provides a secure way to exchange information and authenticate users by using pairs of keys. The public key is used for encryption and signature verification, while the private key is used for decryption and signing. When the two parties communicate with each other to transfer the intelligible or sensible message, referred to as plaintext, is converted into apparen tly random unreadable for security purposes referred to as ciphertext. What is Public Key Cryptography? Public key cryptography is a method of secure communication that uses a pair of keys, a public key, which anyone can use to encrypt messages or verify signatures, and a private key, which is kept secret and used to decrypt messages or sign documents. This system ensures that only the intended recipient can read an encrypted message and that a signed message truly comes from the claimed sender. Public key cryptography is essential for secure internet communications, allowing for confidential messaging, authentication of identities, and verification of data integrity. What is a Cryptographic Key? A cryptographic key is a piece of information used by cryptographic algorithms to encrypt or decrypt data, authenticate identities, or generate digital signatures . It serves as a parameter to control cryptographic operations, ensuring the security and privacy of digital communications and transactions. How Does TLS/SSL Use Public Key Cryptography? TLS/SSL uses public key cryptography to keep our internet connections secure. It does this in two main ways: 1. Encryption: When you visit a secure website ( HTTPS ), TLS/SSL helps encrypt data exchanged between your browser and the website’s server. It uses a combination of public and private keys to create a secure connection. Your browser and the server agree on a secret key for this session, which keeps your data safe from eavesdroppers. 2. Authentication: TLS/SSL verifies the identity of websites. When you connect to a site, it presents a digital certificate signed by a trusted authority. Your browser checks this certificate to ensure you’re really connecting to the right site and not a fake one trying to steal your information. By using public key cryptography, TLS/SSL protects our privacy online and ensures that the websites we visit are genuine and trustworthy. Encryption The process of changing the plaintext into the ciphertext is referred to as encryption. The encryption process consists of an algorithm and a key. The key is a value independent of the plaintext.

**--- Page 2 ---**

The security of conventional encryption depends on the major two factors 1. The Encryption algorithm 2. Secrecy of the key Once the ciphertext is produced, it may be transmitted. The Encryption algorithm will produce a different output depending on the specific key being used at the time. Changing the key changes the output of the algorithm. Once the ciphertext is produced, i t may be transmitted. Upon reception, the ciphertext can be transformed back to the original plaintext by using a decryption algorithm and the same key that was used for encryption . Decryption The process of changing the ciphertext to the plaintext that process is known as decryption. Public Key Encryption : Asymmetric is a form of Cryptosystem in which encryption and decryption are performed using different keys - Public key (known to everyone) and Private key (Secret key). This is known as Public Key Encryption. Difference Between Encryption and Public - Key Encryption Public - Key Basis Encryption Encryption • One algorithm is used for • Same algorithm with the encryption and a related same key is used for algorithm decryption with pair of Required encryption and decryption. keys, one for encryption and for Work • The sender and receiver other for decryption. must share the algorithm • Receiver and Sender must and key. each have one of the matched pair of keys (not identical) . • One of the two keys must be • Key must be kept secret. kept secret. • If the key is secret, it is • If one of the key is kept secret, very impossible to it is very impossible to decipher Required decipher message. message. for • Knowledge of the • Knowledge of the algorithm Security algorithm plus samples of plus one of the keys plus ciphertext must be samples of ciphertext must be impractical to impractical to determine the determine the key. other key. Characteristics of Public Encryption key • Public key Encryption is important because it is infeasible to determine the decryption key given only the knowledge of the cryptographic algorithm and encryption key. • Either of the two keys (Public and Private key) can be used for encryption with other key used for decryption. • Due to Public key cryptosystem, public keys can be freely shared, allowing users an easy and convenient method for encrypting content and verifying digital signatures, and private keys can be kept secret, ensuring only the owners of the private keys can decrypt content and create digital signatures. • The most widely used public - key cryptosystem is RSA (Rivest – Shamir – Adleman). The difficulty of finding the prime factors of a composite number is the backbone of RSA. Example: Public keys of every user are present in the Public key Register. If B wants to send a confidential message to C, then B encrypt the message using C Public key. When C receives the message from B then C can decrypt it using its own Private key. N o other recipient other than C can decrypt the message because only C know C’s private key.

**--- Page 3 ---**

Public Key Encryption Components of Public Key Encryption • Plain Text: This is the message which is readable or understandable. This message is given to the Encryption algorithm as an input. • Cipher Text: The cipher text is produced as an output of Encryption algorithm. We cannot simply understand this message. • Encryption Algorithm: The encryption algorithm is used to convert plain text into cipher text. • Decryption Algorithm: It accepts the cipher text as input and the matching key (Private Key or Public key) and produces the original plain text • Public and Private Key: One key either Private key (Secret key) or Public Key (known to everyone) is used for encryption and other is used for decryption Weakness of the Public Key Encryption • Public key Encryption is vulnerable to Brute - force attack. • This algorithm also fails when the user lost his private key, then the Public key Encryption becomes the most vulnerable algorithm. • Public Key Encryption also is weak towards man in the middle attack. In this attack a third party can disrupt the public key communication and then modify the public keys. • If user private key used for certificate creation higher in the PKI (Public Key Infrastructure) server hierarchy is compromised, or accidentally disclosed, then a “man - in - the - middle attack” is also possible, making any subordinate certificate wholly insecure. This is also the weakness of public key Encryption. Applications of the Public Key Encryption • Encryption/Decryption: Confidentiality can be achieved using Public Key Encryption. In this the Plain text is encrypted using receiver public key. This will ensure that no one other than receiver private key can decrypt the cipher text. • Digital signature: Digital signature is for senders authentication purpose. In this sender encrypt the plain text using his own private key. This step will make sure the authentication of the sender because receiver can decrypt the cipher text using sende rs public key only. • Key exchange: This algorithm can use in both Key - management and securely transmission of data.

**--- Page 4 ---**

RSA Algorithm in Cryptography RSA(Rivest - Shamir - Adleman) Algorithm is an asymmetric or public - key cryptography algorithm which means it works on two different keys: Public Key and Private Key. The Public Key is used for encryption and is known to everyone, while the Private Key is used for decryption and must be kept secret by the receiver. RSA Algorithm is named after Ron Rivest, Adi Shamir and Leonard Adleman, who published the algorithm in 1977. Example of Asymmetric Cryptography: If Person A wants to send a message securely to Person B: • Person A encrypts the message using Person B’s Public Key. • Person B decrypts the message using their Private Key. RSA Algorithm RSA Algorithm is based on factorization of large number and modular arithmetic for encrypting and decrypting data. It consists of three main stages: 1. Key Generation: Creating Public and Private Keys 2. Encryption: Sender encrypts the data using Public Key to get cipher text. 3. Decryption: Decrypting the cipher text using Private Key to get the original data. 1. Key Generation • Choose two large prime numbers, say p and q. These prime numbers should be kept secret. • Calculate the product of primes, n = p \* q. This product is part of the public as well as the private key. • Calculate Euler Totient Function Φ( n) as Φ( n) = Φ( p \* q) = Φ( p) \* Φ( q) = (p – 1) \* (q – 1). • Choose encryption exponent e, such that o 1 < e < Φ( n), and o gcd(e, Φ( n)) = 1, that is e should be co - prime with Φ( n). • Calculate decryption exponent d, such that o (d \* e) ≡ 1 mod Φ( n), that is d is modular multiplicative inverse of e mod Φ( n). Some common methods to calculate multiplicative inverse are: Extended Euclidean Algorithm , Fermat’s Little Theorem , etc. o We can have multiple values of d satisfying (d \* e) ≡ 1 mod Φ( n) but it does not matter which value we choose as all of them are valid keys and will result into same message on decryption. Finally, the Public Key = (n, e) and the Private Key = (n, d). 2. Encryption To encrypt a message M, it is first converted to numerical representation using ASCII and other encoding schemes. Now, use the public key (n, e) to encrypt the message and get the cipher text using the formula: C = Me mod n, where C is the Cipher text and e and n are parts of public key.

**--- Page 5 ---**

3. Decryption To decrypt the cipher text C, use the private key (n, d) and get the original data using the formula: M = Cd mod n, where M is the message and d and n are parts of private key. Advantages • Security: RSA algorithm is considered to be very secure and is widely used for secure data transmission. • Public - key cryptography: RSA algorithm is a public - key cryptography algorithm, which means that it uses two different keys for encryption and decryption. The public key is used to encrypt the data, while the private key is used to decrypt the data. • Key exchange: RSA algorithm can be used for secure key exchange, which means that two parties can exchange a secret key without actually sending the key over the network. • Digital signatures: RSA algorithm can be used for digital signatures, which means that a sender can sign a message using their private key, and the receiver can verify the signature using the sender’s public key. • Widely used: Online banking, e - commerce, and secure communications are just a few fields and applications where the RSA algorithm is extensively developed. Disadvantages • Slow processing speed: RSA algorithm is slower than other encryption algorithms, especially when dealing with large amounts of data. • Large key size: RSA algorithm requires large key sizes to be secure, which means that it requires more computational resources and storage space. • Vulnerability to side - channel attacks: RSA algorithm is vulnerable to side - channel attacks, which means an attacker can use information leaked through side channels such as power consumption, electromagnetic radiation, and timing analysis to extract the pr ivate key. • Limited use in some applications: RSA algorithm is not suitable for some applications, such as those that require constant encryption and decryption of large amounts of data, due to its slow processing speed. • Complexity: The RSA algorithm is a sophisticated mathematical technique that some individuals may find challenging to comprehend and use. • Key Management: The secure administration of the private key is necessary for the RSA algorithm, although in some cases this can be difficult. • Vulnerability to Quantum Computing: Quantum computers have the ability to attack the RSA algorithm, potentially decrypting the data. Diffie - Hellman key exchange (exponential key exchange) What is Diffie - Hellman key exchange (exponential key exchange)? Diffie - Hellman key exchange is a method of digital encryption that securely exchanges cryptographic keys between two parties over a public channel without their conversation being transmitted over the internet. The two parties use symmetric cryptography to encrypt and decrypt their messages. Published in 1976 by Whitfield Diffie and Martin Hellman, it was one of the first practical examples of public key cryptography. Diffie - Hellman key exchange raises numbers to a selected power to produce decryption keys. The components of the keys are never directly transmitted, making the task of a would - be code breaker mathematically overwhelming. The method doesn't share informati on during the key exchange. The two parties have no prior knowledge of each other, but the two parties create a key together. Where is Diffie - Hellman key exchange used? Diffie - Hellman key exchange's goal is to securely establish a channel to create and share a key for symmetric key algorithms. Generally, it's used for encryption, password - authenticated key agreement and forward security. Password - authenticated key agreeme nts are used to prevent man - in - the - middle ( MitM ) attacks. Forward secrecy - based protocols protect against the compromising of keys by generating new key pairs for each session. Diffie - Hellman key exchange is commonly found in security protocols, such as Transport Layer Security ( TLS ), Secure Shell ( SSH ) and IP Security ( IPsec ). For example, in IPsec, the encryption method is used for key generation and key rotation.

**--- Page 6 ---**

Even though Diffie - Hellman key exchange can be used for establishing both public and private keys, the Rivest - Shamir - Adleman algorithm, or RSA algorithm , can also be used, since it's able to sign public key certificates . How does Diffie - Hellman key exchange work? To implement Diffie - Hellman, two end users, Alice and Bob, mutually agree on positive whole numbers p and q , such that p is a prime number and q is a generator of p . The generator q is a number that, when raised to positive whole - number powers less than p , never produces the same result for any two such whole numbers. The value of p may be large, but the value of q is usually small. Once Alice and Bob have agreed on p and q in private, they choose positive whole - number personal keys a and b . Both are less than the prime number modulus p . Neither user divulges their personal key to anyone; ideally, they memorize these numbers and don't write them down or store them anywhere. Next, Alice and Bob compute public keys a\* and b\* based on their personal keys according to the following formulas: a\* = q mod p a b\* = q mod p b The two users can share their public keys a\* and b\* over a communications medium assumed to be insecure, such as the internet or a corporate wide area network. From these public keys, a number x can be generated by either user on the basis of their own personal keys. Alice computes x using the following formula: x = ( b\* ) mod p Bob computes x using the following formula: x = ( a\* ) mod p The value of x turns out to be the same according to either of the above two formulas. However, the personal keys a and b , which are critical in the calculation of x , haven't been transmitted over a public medium. Because it's a large and apparently random number, a potential hacker has almost no chance of correctly guessing x , even with the help of a powerful computer to conduct millions of trials. The two users can, therefore, in theory, communicate privately over a public medium with an encryption method of their choice using the decryption key x . - Vulnerabilities of Diffie - Hellman key exchange The most serious limitation of Diffie - Hellman in its basic form is the lack of authentication . Communications using Diffie - Hellman by itself are vulnerable to MitM. Ideally, Diffie - Hellman should be used in conjunction with a recognized authentication method, such as digital signatures , to verify the identities of the users over the public communications medium. Diffie - Hellman key exchange is also vulnerable to logjam attacks, specifically against the TLS protocol . Logjam attacks downgrade TLS connections to 512 - bit cryptography, enabling an attacker to read and modify data that's passed through the connection. Diffie - Hellman key exchange can still be secure if implemented correctly. For example, logjam attacks won 't work with a 2,048 - bit key.

**--- Page 7 ---**

Diffie - Hellman key exchange works by having both parties choose a private key, calculate a public key, share both computed keys and compute a shared secret key for secured communication. Examples of Diffie - Hellman key exchange If two people, say Alice and Bob, want to communicate sensitive data over an open public network but want to avoid hackers or eavesdroppers, they can use Diffie - Hellman key exchange method for encryption. This open public network could be at a cafe, for ex ample. Alice and Bob privately choose a secret key, and a function is run on these keys to create a public key. The results -- and not the function -- are shared. Even if a third party is listening in, that third party won't have all the involved numbers, making it difficult to derive the function the numbers came from. From here, Alice and Bob each run a new function using the results they received from the opposite party, their own secret number and the original prime value. Alice and Bob then arrive at a common shared secret key that a third party can't deduce. Alice a nd Bob are now free to communicate without worrying about third parties. What is One Way Function in Cryptography? One - way functions in cryptography are easy to compute, whereas inverse functions are complicated to compute. There is no mathematical proof that one - way functions exist. The existence of such one - way functions is still a not - resolved question. Their existence indicates that the complexity classes are not equivalent. In this article, we wil l learn about the one - way function in cryptography, the trapdoor one - way function, the one - way hash function, and how the one - way functions work. A one - way function is a widely used mathematical function that is easier to compute in one direction. The forward direction of a function can be computed fast, while the inverse can take months. One - way functions in cryptography fulfill all conditions for one - way functions. It is easy to find their values using input data, but having merely a hash value prevents one from determining the original input sequence. A one - way hash function should be free of collisions. This means it c an be extremely difficult to locate two sequences that give the same result. It converts input messages of different lengths into output sequences of fixed length. The output sequence is often referred to as a hash value. Hash values are frequently used to label input sequences, that is, to assign them unique values that identify them. Trapdoor One Way Function A trapdoor one - way function is easy to evaluate in one direction but difficult to compute in the opposite direction. When some additional information is revealed, such a function becomes a trapdoor one - way function. In mathematical terms, if "f" is a trapdoor function, then there is some hidden information "t" that allows "x" to be easily computed given f(x) and "t". Consider the padlock and its key. To change the padlock from open to closed without using the key, sim ply push the shackle into the lock mechanism. The padlock can be readily opened, but the key is required. The key is the trapdoor, and the padlock performs the trapdoor function. One Way Hash Function One - way hash function algorithms provide security as their properties are one - way functions. Changing one bit of input data results in changing around half of the output bits. These are used to protect data from intentional or unintended changes. It satisfies all of the requirements for one - way functions. It is simple to calculate their values using input data, but having merely a hash value prevents one from determining the original input sequence.

**--- Page 8 ---**

One Way Function How Do One Way Functions Work? • A one - way function mainly starts with a group of letters known as a key, which is then mapped to a hash of a specific length. • Modern hashes have 128 bits or more, however, the hash value is shorter than the original string of characters. • The hash value is often referred to as a message digest . When you scramble this message digest, it becomes unique each time. • A one - way function should be free of collisions. This means it should be extremely difficult to locate two distinct sequences that give the same hash result. • Furthermore, one - way functions are used to store data. Data can be accessed by locating hash values, and are stored in computer memory. Types of One Way Functions in Cryptography • Strong One - Way Function: A Strong One - Way function is easy to compute but can only be inverted with negligible probability on random input, or difficult to invert on all but a small number of inputs. • Weak One - Way Function: A Weak One - Way function is easy to compute but difficult to invert for random inputs, or easy to invert for a significant fraction of inputs. Why Use One Way Functions in Cryptography? • One - way hash functions were first used in databases because it is faster to locate a short piece of information in a hash than a whole, long string of characters. Since their inception, one - way hash functions have become more popular in encryption . • Hashing enables people to encrypt and decrypt digital signatures . The digital signature is converted into a hash value and transmitted to the recipient. • After receiving the encrypted value, the computer generates the hash value using the same hash function . • Then compares it to the message. If the two are the same, the message was delivered without mistakes. Conclusion In computer science, a one - way function is easy to compute in one direction but hard to reverse. This means you can quickly get the output from an input, but it's tough to figure out the input from the output. The terms "easy" and "hard" here refer to how quickly these tasks can be done by a computer, specifically relating to polynomial time problems. For a function to be considered one - way, it must be hard to reverse even if it's not one - to - one (where one input gives a unique output).

**--- Page 9 ---**

Message Digest in Information security A Message is any information that flows via the network, such as files, emails, and financial transactions, from one device to another or from a set of devices. When a message is sent over a network, it must be secure to ensure that it is safe from anybody sitting in the middle listening to the conversation and having the ability to access, alter, or modify the message. What is a Message Digest or Hash Value? A message digest or hash value is a numeric string generated using the cryptographic hash function. The message string is passed to the hash function. Hash function computes a unique hash value for the provided message and this hash value acts as digital f ingerprint of the message. The cryptographic function creates the message digest, which may then be encrypted to give a second layer of security. This function is one - way; it can only be used to generate the message digest from the message, not the other way around. If the message d igest is created using a symmetric key then it is known as MAC or Message Authentication Code. Encrypted message digests work as digital fingerprints and the receiver needs to decrypt the digests first to compare them. Properties of a Message Digest • The message digest is always a unique numeric hash value. It cannot be the same for two or more messages. Once the message digest value is generated for a message, it's only associated with that message. • The size of the hash value is fixed. For any length of message a fixed length hash value is generated and that depends on the implementation of the hash function. • If you use the hash function multiple times for the same function again and again, then you only get the same digest value each time. For example, if you send "hii" to the hash function the generated message digest value will be the same each time when you pass "hii" to the hash function. It's not going to change. • We cannot generate the message by passing the message digest value to the hash function. A hash function is a one - way function. How Does a Message Digest Work? Message digest works as a digital fingerprint for the message. In the communication channel, the sender and receiver communicate with each other so both must receive the right message. to ensure the integrity of the message digest is sent by the sender to the receiver along with the message. The receiver receives the message and the message digest value and the receiver uses the same hash function to generate a new message for the message he/she received. once generated receiver compares both the message di gest values to verify that the message is received without any modifications and corrections. If both digest values are the same, it proves that the message has not been modified in the network by any person. Different digest values indicate that the recei ved message is not the actual message send by the sender. This way we could check and verify the integrity of the message using message digest. For more clarity refer to the below example: For example, if Alice wants to send a message to Bob, first of all, the message will be sent to the hash function , which will create the message digest. The created message digest is a unique string that cannot be converted back into a message. Alice will send this message digest and message to Bob over the communication channel. To ensure that Bob received the corre ct message digest, he will use the same hash function on the message and generate a new digest. Bob will compare the digest values; if they match, the message is original; otherwise, it has been modified. Message Digests and Integrity Protection The message digest is sent with the message. The receiver may generate a digest for the message and compare it to the sender's. If both message digests are the same, the message's integrity has been verified. Any modification of the message during transmis sion probably generates a different message digest. So With the help of message digests, we can check the integrity of the messages, which means we can confirm whether the message is an original message or a modified message through the comparison of the message digests generated at different places. The me ssage digest value for a particular message will always be the same whether it's generated multiple

**--- Page 10 ---**

times, so it ensures that every message has a unique digest and identifies a single message. The message digest can be encrypted using cryptographic symmetric or asymmetric algorithms . The encrypted message digest acts as a digital fingerprint of the message. when the sender sends the message and encrypted message digest, the sender also sends the secret key by which the message digest is encrypted along with them. The receiver receive s the message, the message digest, and the key and the receiver decrypts the digest with the key and compares both the digest values to check the integrity of the message. Encryption ensures that only authorized persons can access the digest value and decrypt it. Malicious attackers and middle men can not decrypt the digest without knowing the secret key and the secret key is only known to the sender and receiver. Conclusion In conclusion, it is important to understand that any piece of data sent via any communication channel can be intercepted or altered unless other proper measures are put in place hence there is a need to ensure data integrity using what we call a message d igest that gives the digital fingerprint for any given message or information. What is the MD5 Algorithm? MD5 is a cryptographic hash function algorithm that takes the message as input of any length and changes it into a fixed - length message of 16 bytes. MD5 algorithm stands for the Message - Digest algorithm . MD5 was developed in 1991 by Ronald Rivest as an improvement of MD4, with advanced security purposes. The output of MD5 (Digest size) is always 128 bits. MD5 is still the most commonly used message digest for non - cryptographic functions, such as used as a checksum to verify data integrity, compressing large files int o smaller ones securely, etc . Working of the MD5 Algorithm MD5 algorithm follows the following steps 1. Append Padding Bits: In the first step, we add padding bits in the original message in such a way that the total length of the message is 64 bits less than the exact multiple of 512. Suppose we are given a message of 1000 bits. Now we have to add padding bits to the original message. Here we will add 472 padding bits to the original message. After adding the padding bits the size of the original message/output of the first step will b e 1472 i.e. 64 bits less than an exact multiple of 512 (i.e. 512\*3 = 1536). Length(original message + padding bits) = 512 \* i - 64 where i = 1,2,3 . . . 2. Append Length Bits: In this step, we add the length bit in the output of the first step in such a way that the total number of the bits is the perfect multiple of 512. Simply, here we add the 64 - bit as a length bit in the output of the first step. output of first step = 512 \* n - 64 length bits = 64. After adding both we will get 512 \* n i.e. the exact multiple of 512. 3. Initialize MD buffer: Here, we use the 4 buffers i.e. A B, C, and D. The size of each buffer is 32 bits. - A = 0x67425301 - B = 0xEDFCBA45 - C= 0x98CBADFE - D = 0x13DCE476 4. Process Each 512 - bit Block: This is the most important step of the MD5 algorithm. Here, a total of 64 operations are performed in 4 rounds. In the 1st round, 16 operations will be performed, 2nd round 16 operations will be performed, 3rd round 16 operations will be performed, and in the 4th round, 16 operations will be performed. We apply a different function on each round i.e. for the 1st round we apply the F function, for the 2nd G function, 3rd for the H function, and 4th for the I function. We perfo rm OR, AND, XOR, and NOT (basically these are logic gates) for calculating functions. We use 3 buffers for each function i.e. B, C, D. After applying the function now we perform an operation on each block. For performing operations we need • add modulo 2 32 • M[i] – 32 bit message. • K[i] – 32 - bit constant.

**--- Page 11 ---**

• <<<n – Left shift by n bits. Now take input as initialize MD buffer i.e. A, B, C, D. Output of B will be fed in C, C will be fed into D, and D will be fed into J. After doing this now we perform some operations to find the output for A. • In the first step, Outputs of B, C, and D are taken and then the function F is applied to them. We will add modulo 2 32 bits for the output of this with A. • In the second step, we add the M[i] bit message with the output of the first step. • Then add 32 bits constant i.e. K[i] to the output of the second step. • At last, we do left shift operation by n (can be any value of n) and addition modulo by 2 32 . After all steps, the result of A will be fed into B. Now same steps will be used for all functions G, H, and I. After performing all 64 operations we will get our message digest. Output: After all rounds have been performed, the buffer A, B, C, and D contains the MD5 output starting with the lower bit A and ending with Higher bits D. Applications of MD5 Algorithm • MD5 is used as a checksum to verify the integrity of files and data by comparing the hash of the original file with the file received to check if the files or data has been altered. • MD5 is used for data security and encryption e.g. Secure password of users in database and non - sensitive data. • It is used in version control systems to manage different versions of files. • It was earlier used in digital signatures and certificate but due it’s vulnerabilities, it has been replaced by more secure algorithms like SHA - 256. Advantages of MD5 Algorithm • MD5 is faster and simple to understand. • MD5 algorithm generates a strong password in 16 bytes format. All developers like web developers, etc. use the MD5 algorithm to secure the password of users. • To integrate the MD5 algorithm, relatively low memory is necessary. • It is very easy and faster to generate a digest message of the original message. Disadvantages of MD5 Algorithm • MD5 generates the same hash function for different inputs ( hash collision ). • MD5 provides poor security over SHA1 , SHA256 and other modern cryptographic algorithms. • MD5 has been considered an insecure algorithm. So now we are using SHA256 instead of MD5. • MD5 is neither a symmetric nor asymmetric algorithm. Alternatives to MD5 in Modern Cryptography As MD5 has been found to have vulnerabilities, several more secure cryptographic hash functions are commonly used in modern applications: 1. SHA - 256 (Secure Hash Algorithm 256 - bit) is a part of the SHA - 2 family that produces a 256 - bit hash value. It is widely used in blockchain, SSL certificates, digital signatures, and certificates. 2. SHA - 3 offers an alternative to SHA - 2 with a different internal structure, making it more resilient to certain attacks. While SHA - 2 is still more common, SHA - 3 is used for secure applications requiring the highest cryptographic standards. 3. RIPEMD - 160 produces a 160 - bit hash value, providing better security than MD5 but not as widely used as SHA - 2. It is sometimes used in digital signatures and certificates in specific security systems. 4. Whirlpool is a cryptographic hash function designed for high security, producing a 512 - bit hash output. It’s used particularly in cases where very strong security is required. SHA - 1 Hash SHA - 1 or Secure Hash Algorithm 1 is a cryptographic algorithm that takes an input and produces a 160 - bit (20 - byte) hash value. This hash value is known as a message digest. This message digest is usually then rendered as a hexadecimal number which is 40 di gits long. It is a U.S. Federal Information Processing Standard and was designed by the United States National Security Agency. SHA - 1 is been considered insecure since 2005. Major tech

**--- Page 12 ---**

giants browsers like Microsoft, Google, Apple, and Mozilla have stopped accepting SHA - 1 SSL certificates by 2017. How SHA - 1 Works The block diagram of the SHA - 1 (Secure Hash Algorithm 1) algorithm. Here’s a detailed description of each component and process in the diagram: Components and Process Flow: 1. Message (M) : • The original input message that needs to be hashed. 2. Message Padding : • The initial step where the message is padded to ensure its length is congruent to 448 modulo 512. This step prepares the message for processing in 512 - bit blocks. 3. Round Word Computation (WtW\_tWt ) : • After padding, the message is divided into blocks of 512 bits, and each block is further divided into 16 words of 32 bits. These words are then expanded into 80 32 - bit words, which are used in the subsequent rounds. 4. Round Initialize (A, B, C, D, and E) : • Initialization of five working variables (A, B, C, D, and E) with specific constant values. These variables are used to compute the hash value iteratively. 5. Round Constants (KtK\_tKt ) : • SHA - 1 uses four constant values (K1K\_1K1 , K2K\_2K2 , K3K\_3K3 , K4K\_4K4 ), each applied in a specific range of rounds: o K1K\_1K1 for rounds 0 - 19 o K2K\_2K2 for rounds 20 - 39 o K3K\_3K3 for rounds 40 - 59 o K4K\_4K4 for rounds 60 - 79 6. Rounds (0 - 79) : • The main computation loop of SHA - 1, divided into four stages (each corresponding to one of the constants K1K\_1K1 to K4K\_4K4 ). In each round, a combination of logical functions and operations is performed on the working variables (A, B, C, D, and E) using the words generated in the previous step. 7. Final Round Addition : • After all 80 rounds, the resulting values of A, B, C, D, and E are added to the original hash values to produce the final hash. 8. MPX (Multiplexing) :

**--- Page 13 ---**

• Combines the results from the final round addition to form the final message digest. Summary of Steps: • Input (Message M) : The process starts with the input message MMM. • Message Padding : The message is padded to meet the length requirements. • Word Computation : The padded message is divided into blocks and further into words, which are expanded for use in the rounds. • Initialization : Initial hash values are set. • Round Processing : The main loop performs 80 rounds of computation using the message words and round constants. • Final Addition : The results from the rounds are added to the initial hash values. • Output (Hash Value) : The final message digest is produced. Applications: • Cryptography : The main application of SHA1 is to protect communications from being intercepted by outside parties. From a given data input, SHA1 generates a fixed - size, singular, and irreversible hash value. The integrity of the data can then be confirmed by comparing this hash value to the original hash value. This makes it possible to confirm that the data was not changed or tampered with in any manner during transmission. • Data Integrity : In many industries, such as finance, healthcare, and government, data integrity is a major concern. Data integrity in a system is checked using the SHA1 algorithm. A fingerprint of the original data is created using a hash value produced by the SHA1 algo rithm. If the data changes in any way, the hash value will also change, indicating that the data has been tampered with. • Digital Signatures : Digital signatures are used to confirm the legitimacy of digital documents and messages. The digital document or communication is hashed using the SHA1 technique, and its hash value is subsequently encrypted with the sender’s private key. Using the sende r’s public key to decode the message, the recipient can then compare the hash value to the original value. • Digital Forensics : In digital forensics, a hash of a file containing digital evidence can be produced using the SHA1 algorithm. To ensure that the evidence hasn’t been altered with during the investigation, utilize this hash value as proof. It gives proof that the file has not been altered if the hash values of the original file and the evidence file match. • Password Storage : SHA1 can be used to save passwords. A hash of the password is generated using SHA1 when a user creates a password. The password itself is then substituted in a database for the hash value. The user’s password is hashed with SHA1 when they attempt to log in, and the resulting hash is compared to a previously generated hash. • Software Updates : The integrity of software updates can be guaranteed using SHA1. The SHA1 hash of the update file can be made public on the software vendor’s website when an update is made available. By comparing the hash of the downloaded file with the published hash, u sers can download the update and ensure its integrity. Message Authentication Code (MAC) MAC algorithm is a symmetric key cryptographic technique to provide message authentication. For establishing MAC process, the sender and receiver share a symmetric key K. Essentially, a MAC is an encrypted checksum generated on the underlying message that is sent along with a message to ensure message authentication. The process of using MAC for authentication is depicted in the following illustration −

**--- Page 14 ---**

Let us now try to understand the entire process in detail − • The sender uses some publicly known MAC algorithm, inputs the message and the secret key K and produces a MAC value. • Similar to hash, MAC function also compresses an arbitrary long input into a fixed length output. The major difference between hash and MAC is that MAC uses secret key during the compression. • The sender forwards the message along with the MAC. Here, we assume that the message is sent in the clear, as we are concerned of providing message origin authentication, not confidentiality. If confidentiality is required then the message needs encryption . • On receipt of the message and the MAC, the receiver feeds the received message and the shared secret key K into the MAC algorithm and re - computes the MAC value. • The receiver now checks equality of freshly computed MAC with the MAC received from the sender. If they match, then the receiver accepts the message and assures himself that the message has been sent by the intended sender. • If the computed MAC does not match the MAC sent by the sender, the receiver cannot determine whether it is the message that has been altered or it is the origin that has been falsified. As a bottom - line, a receiver safely assumes that the message is not th e genuine. Limitations of MAC There are two major limitations of MAC, both due to its symmetric nature of operation − • Establishment of Shared Secret. o It can provide message authentication among pre - decided legitimate users who have shared key. o This requires establishment of shared secret prior to use of MAC. • Inability to Provide Non - Repudiation o Non - repudiation is the assurance that a message originator cannot deny any previously sent messages and commitments or actions. o MAC technique does not provide a non - repudiation service. If the sender and receiver get involved in a dispute over message origination, MACs cannot provide a proof that a message was indeed sent by the sender. o Though no third party can compute the MAC, still sender could deny having sent the message and claim that the receiver forged it, as it is impossible to determine which of the two parties computed the MAC. Digital Signatures D igital signatures and certificates are two key technologies that play a crucial role in ensuring the security and authenticity of online activities. They are essential for activities such as online banking, secure email communication, software distribution, and electronic document signing. By providing mechanisms for authentication, integrity, and non - repudiation, these technologies help protect against fraud, data breaches, and unauthorized access. Digital Signature A digital signature is a mathematical technique used to validate the authenticity and integrity of a message, software, or digital document. These are some of the key features of it.

**--- Page 15 ---**

1. Key Generation Algorithms : Digital signatures are electronic signatures, which assure that the message was sent by a particular sender. While performing digital transactions authenticity and integrity should be assured, otherwise, the data can be altered or someone can also act as if he were the sender and expect a reply. 2. Signing Algorithms : To create a digital signature, signing algorithms like email programs create a one - way hash of the electronic data which is to be signed. The signing algorithm then encrypts the hash value using the private key (signature key). This encrypted hash along with other information like the hashing algorithm is the digital signature. This digital signature is appended with the data and sent to the verifier. The reason for encrypting the hash instead of the entire message or document is that a hash function converts any arbitrary input into a much shorter fixed - length value. This saves time as now instead of signing a long message a shorter hash value has to be signed and hashing is much faster than signing. 3. Signature Verification Algorithms : The Verifier receives a Digital Signature along with the data. It then uses a Verification algorithm to process the digital signature and the public key (verification key) and generates some value. It also applies the same hash function on the received d ata and generates a hash value. If they both are equal, then the digital signature is valid else it is invalid. The steps followed in creating a digital signature are: 1. Message digest is computed by applying the hash function on the message and then message digest is encrypted using the private key of the sender to form the digital signature. (digital signature = encryption (private key of sender, message digest) and mess age digest = message digest algorithm (message)). 2. A digital signature is then transmitted with the message. (message + digital signature is transmitted) 3. The receiver decrypts the digital signature using the public key of the sender. (This assures authenticity, as only the sender has his private key so only the sender can encrypt using his private key which can thus be decrypted by the sender’s public key). 4. The receiver now has the message digest. 5. The receiver can compute the message digest from the message (actual message is sent with the digital signature). 6. The message digest computed by receiver and the message digest (got by decryption on digital signature) need to be same for ensuring integrity. Message digest is computed using one - way hash function , i.e. a hash function in which computation of hash value of a message is easy but computation of the message from hash value of the message is very difficult. Assurances About Digital Signatures The definitions and words that follow illustrate the kind of assurances that digital signatures offer. 1. Authenticity : The identity of the signer is verified. 2. Integration: Since the content was digitally signed, it hasn’t been altered or interfered with.

**--- Page 16 ---**

3. Non - repudiation: demonstrates the source of the signed content to all parties. The act of a signer denying any affiliation with the signed material is known as repudiation. 4. Notarization: Under some conditions, a signature in a Microsoft Word, Microsoft Excel, or Microsoft PowerPoint document that has been time - stamped by a secure time - stamp server is equivalent to a notarization. Benefits of Digital Signatures • Legal documents and contracts: Digital signatures are legally binding. This makes them ideal for any legal document that requires a signature authenticated by one or more parties and guarantees that the record has not been altered. • Sales contracts: Digital signing of contracts and sales contracts authenticates the identity of the seller and the buyer, and both parties can be sure that the signatures are legally binding and that the terms of the agreement have not been changed. • Financial Documents: Finance departments digitally sign invoices so customers can trust that the payment request is from the right seller, not from a attacker trying to trick the buyer into sending payments to a fraudulent account. • Health Data: In the healthcare industry, privacy is paramount for both patient records and research data. Digital signatures ensure that this confidential information was not modified when it was transmitted between the consenting parties. Drawbacks of Digital Signature • Dependency on technology: Because digital signatures rely on technology, they are susceptible to crimes, including hacking . As a result, businesses that use digital signatures must make sure their systems are safe and have the most recent security patches and upgrades installed. • Complexity: Setting up and using digital signatures can be challenging, especially for those who are unfamiliar with the technology. This may result in blunders and errors that reduce the system’s efficacy. The process of issuing digital signatures to senior citizens can occasionally be challenging. • Limited acceptance: Digital signatures take time to replace manual ones since technology is not widely available in India, a developing nation. What is a Digital Envelope? • A digital envelope is a method used in cryptography to securely send a message along with its decryption key. It combines the speed of symmetric encryption with the security of asymmetric encryption. The process involves encrypting the message using a symm etric key (which is fast but less secure) and then encrypting that key with the recipient's public key (which is more secure). The recipient uses their private key to decrypt the symmetric key, which in turn decrypts the message. This method ensures that t he message stays secure during transit, even if intercepted. Practical Application of Digital Envelopes In practice, digital envelopes are used in various applications, from securing email communications to protecting sensitive data in cloud storage. For example, when you send an encrypted email, a digital envelope is often used to secure the content. The em ail service encrypts your message with a symmetric key, then encrypts that key with the recipient's public key, ensuring that only the intended recipient can access the message. The Benefits of Digital Envelopes Digital envelopes offer numerous advantages, making them an essential tool in digital security. They provide a high level of security by combining two types of encryption methods, making it extremely difficult for unauthorized parties to access the encrypt ed data. They also ensure the authenticity and integrity of the data, as the recipient can verify the sender's identity through the public key infrastructure. Additionally, digital envelopes facilitate secure key exchange, which is crucial in many cryptogr aphic protocols.