**🡺 Cryptography**

Cryptography is the practice of concealing information by converting plaintext (readable format) into ciphertext (unreadable format) using a key or encryption scheme. It is the process of converting data into a scrambled code that is encrypted and sent across a private or public network. Plaintext (readable format) is encrypted by means of encryption algorithms such as RSA, DES, and AES, resulting in a ciphertext (unreadable format) that, on reaching the destination, is decrypted into readable plaintext.

Objectives of cryptography: Confidentiality, Integrity, Authentication, Nonrepudiation

**Types of Cryptography**

* **Symmetric Encryption:** Symmetric encryption requires that both the sender and the receiver of the message possess the same encryption key. The sender uses a key to encrypt the plaintext and sends the resultant ciphertext to the recipient, who uses the same key (used for encryption) to decrypt the ciphertext into plaintext. Symmetric encryption is also known as secret-key cryptography, as it uses only one secret key to encrypt and decrypt the data.
* **Asymmetric Encryption:** An asymmetric-key system is an encryption method that uses a key pair comprising a public key available to anyone and a private key held only by the key owner, which helps to provide confidentiality, integrity, authentication, and nonrepudiation in data management. No one but the holder of the private key can decrypt a message encrypted with the corresponding public key. Asymmetric encryption uses the following sequence to send a message:
* An individual finds the public key of the person he or she wants to contact in a directory.
* This public key is used to encrypt a message that is then sent to the intended recipient.
* The receiver uses the private key to decrypt the message and reads it.

**Government Access Key (GAK)**

Government Access to Keys (GAK) refers to the statutory obligation of individuals and organizations to disclose their cryptographic keys to government agencies. It means that software companies will give copies of all keys (or at least enough of the key such that the remainder can be cracked) to the government. The government promises that it will hold on to the keys in a secure manner and only use them when a court issues a warrant to do so. Government agencies often use key escrow for uninterrupted access to keys. Key escrow is a key exchange arrangement in which essential cryptographic keys are stored with a third party in escrow. The third party can use or allow others to use the encryption keys under certain predefined circumstances.

**🡺 Encryption Algorithm**

**Ciphers**

In cryptography, a cipher is an algorithm (a series of well-defined steps) for performing encryption and decryption. Encipherment is the process of converting plaintext into a cipher or code; the reverse process is called decipherment. A message encrypted using a cipher is rendered unreadable unless its recipient knows the secret key required to decrypt it.

Types of Ciphers:

* **Classical Cipher:** Classical ciphers are the most basic type of ciphers, which operate on letters of the alphabet (A–Z).
* **Substitution cipher:** The user replaces units of plaintext with ciphertext according to a regular system. The units may be single letters, pairs of letters, or combinations of them, and so on. Examples include the Beale cipher, autokey cipher, Gronsfeld cipher, and Hill cipher. “HELLO WORLD” can be encrypted as “PSTER HGFST” (H=P, E=S, etc.).
* **Transposition cipher:** Here, letters in the plaintext are rearranged according to a regular system to produce the ciphertext. Examples include the rail fence cipher, route cipher, and Myszkowski transposition. “CRYPTOGRAPHY” when encrypted becomes “AOYCRGPTYRHP.”
* **Modern Cipher:** Modern ciphers are designed to withstand a wide range of attacks. They provide message secrecy, integrity, and authentication of the sender.
* Based on the type of key used:

**Symmetric Key Algorithm**

**Asymmetric Key Algorithm**

* Based on the type of input data:**Block cipher:** Deterministic algorithms operating on a block (a group of bits) of fixed size with an unvarying transformation specified by a symmetric key. They are widely used to encrypt bulk data. Examples include DES, AES, IDEA, etc.

**Stream cipher:** Symmetric-key ciphers are plaintext digits combined with a key stream (pseudorandom cipher digit stream). Here, the user applies the key to each bit, one at a time. Examples include RC4, SEAL, etc.

**Data Encryption Standard (DES)**

DES is a standard for data encryption that uses a secret key for both encryption and decryption (symmetric cryptosystem). DES uses a 64-bit secret key, of which 56 bits are generated randomly and the other 8 bits are used for error detection. DES is the archetypal block cipher—an algorithm that takes a fixed-length string of plaintext bits and transforms it into a ciphertext bit string of the same length. DES provides 72 quadrillion or more possible encryption keys and chooses a random key for the encryption of each message.

**Triple Data Encryption Standard (3DES)**

Tt performs DES three times with three different keys. 3DES uses a “key bundle” that comprises three DES keys, K1, K2, and K3. Each key is a standard 56-bit DES key. It then performs the following process: DES encrypt with K1, DES decrypt with K2, DES encrypt with K3 There are three options for the keys. In the first option, all three keys are independent and different. In the second option, K1 and K3 are identical. In the third option, all three keys are the same.

**Advanced Encryption Standard (AES)**

The Advanced Encryption Standard (AES) is a National Institute of Standards and Technology (NIST) specification for the encryption of electronic data. AES consists of a symmetric-key algorithm: both encryption and decryption are performed using the same key. It is an iterated block cipher that works by repeating the defined steps multiple times. It has a 128-bit block size, with key sizes of 128, 192, and 256 bits for AES-128, AES-192, and AES-256, respectively. It works simultaneously at multiple network layers.

Initially, the system copies the cipher input into the internal state and then adds an initial round key. The system transforms the state by iterating a round function in a number of cycles. The number of cycles may vary with the block size and key length.

**RC4, RC5, RC6 Algorithm**

* **RC4** is a variable key-size symmetric-key stream cipher with byte-oriented operations, and it is based on the use of a random permutation. According to some analyses, the period of the cipher is likely to be greater than 10,100. Each output byte uses 8 to 16 system operations; thus, the cipher can run fast when used in software.
* **RC5** is a fast symmetric-key block cipher designed by Ronald Rivest for RSA Data Security (now RSA Security). The algorithm is a parameterized algorithm with a variable block size, a variable key size, and a variable number of rounds. The block sizes can be 32, 64, or 128 bits. The range of the rounds can vary from 0 to 255, and the size of the key can vary from 0 to 2,040 bits. The routines used in RC5 are key expansion, encryption, and decryption.In the key expansion routine, the secret key that a user provides is expanded to fill the key table. RC5 uses a key table for both encryption and decryption. The encryption routine has three fundamental operations: integer addition, bitwise XOR, and variable rotation.
* **RC6** is a symmetric-key block cipher derived from RC5. It is a parameterized algorithm with a variable block size, key size, and number of rounds. Two features that differentiate RC6 from RC5 are integer multiplication (which is used to increase the diffusion, achieved in fewer rounds with increased speed of the cipher) and the use of four 4-bit working registers rather than two 2-bit registers. RC6 uses four 4-bit registers instead of two 2-bit registers because the block size of the AES is 128 bits.

**Blowfish**

Blowfish is a type of symmetric block cipher algorithm designed to replace DES or IDEA algorithms. It uses the same secret key to encrypt and decrypt data. This algorithm splits the data into a block length of 64 bits and produces a key ranging from 32 bits to 448 bits. t is a 16-round Feistel cipher working on 64-bit blocks.

This algorithm has two parts. The first part handles the expansion of the key. The second part actually encrypts the data. The key expansion is handled in several steps. The first step is to break the original key into a set of subkeys. Specifically, a key of no more than 448 bits is separated into 4,168 bytes. There is a P-array and four 32-bit S-boxes. The P-array contains 18 32-bit subkeys, while each S-box contains 256 entries. The round function splits the 32-bit input into four 8-bit quarters and uses the quarters as input to the S-boxes. The outputs are added modulo 232 and XORed to produce the final 32-bit output.

**Twofish**

Twofish was designed by Bruce Schneier, John Kelsey, Doug Whiting, David Wagner, Chris Hall, and Niels Ferguson. It is a 128-bit block cipher. It is one of the most conceptually simple algorithms that uses a single key for both encryption and decryption for any length up to 256 bits. It is a Feistel cipher. It not only works fast for CPU or hardware but is also flexible for network-based applications.

**Threefish**

Threefish was developed in 2008 and it is a part of the Skein algorithm. It was enrolled in NIST’s SHA-3 (hash function) contest. It is a large tweakable symmetric-key block cipher in which the block and key sizes are equal, i.e., 256, 512, and 1024. Threefish involves only three operations, i.e., ARX (addition-rotation-XOR), which makes the coding simple, and all these operations work on 64-bit words. Threefish blocks 256, 512, and 1024 involve 72, 72, and 80 rounds of computations, respectively, to achieve the final security goal. This algorithm does not use S-boxes to prevent cache timing attacks.

**Serpent**

Serpent is a symmetric-key block cipher that was a finalist in the AES contest. This algorithm was designed by Ross Anderson, Eli Biham, and Lars Knudsen. It uses a 128-bit symmetric block cipher with key sizes of 128, 192, or 256 bits. It can be integrated into software or hardware programs without any restrictions. Serpent involves 32 rounds of computational operations that include substitution and permutation operations on four 32-bit word blocks using 8-variable S-boxes with 4-bit entry and 4-bit exit. All S-boxes work parallelly 32 times. Serpent minimizes the correlation between encoded images or plaintexts to a greater extent compared to Twofish and Rijndael.

**TEA**

The tiny encryption algorithm (TEA) was created by David Wheeler and Roger Needham, and it was publicly presented for the first time in 1994. It is a simple algorithm, easy to implement in code. It is a Feistel cipher that uses 64 rounds or lesser. The number of rounds should be even since they are implemented in pairs called cycles. TEA uses a 128-bit key operating on a 64-bit block. It also uses a constant that is defined as 232/the golden ratio. This constant is referred to as delta, and in each round, a multiple of delta is used. The 128-bit key is split into four different 32-bit subkeys labeled K[0], K[1], K[2], and K[3]. Instead of using the XOR operation, TEA uses addition and subtraction, but with mod 232. The block is divided into two halves, R and L. R is processed through the round function.

The round function takes the R half and performs a left shift of 4. Then, the result of this operation is added to K[0]. Next, the result of this operation is added to delta (recall that delta is the current multiple of 232/the golden ratio). The result of this operation is then shifted right by 5 and added to K[1]. This is the round function. As with all Feistel ciphers, the result of the round function is XORed with L, and L and R are then swapped for the next round.

**CAST-128**

CAST-128, also called CAST5, is a symmetric-key block cipher having a classical 12-or 16-round Feistel network with a block size of 64 bits. CAST-128 uses a key size varying from 40 bits to 128 bits in 8-bit increments. The CAST-128 components include large 8×32-bit S-boxes (S1, S2, S3, S4) based on bent functions, modular addition and subtraction, key-dependent rotation, and XOR operations. CAST-128 uses a masking key (Km1) and a rotation key (Kr1) for performing its functions. The round function consists of three alternating types to perform addition, subtraction, or XOR operations in different stages. It used as a default cipher in GPG (GNU Privacy Guard) and PGP (Pretty Good Privacy).

CAST-256 is an extension of CAST-128 that uses the same design procedure. CAST-256 has a 128-bit block size, and it uses key sizes varying from 128 to 256 bits. Furthermore, it uses zero-correlation cryptanalysis, which can break 28 rounds with time = 2246.9 and data = 298.8.

**GOST Block Cipher**

The GOST (Government Standard) block cipher, also called Magma, is a symmetric-key block cipher having a 32-round Feistel network working on 64-bit blocks with a 256-bit key length. It consists of an S-box that can be kept secret and it contains around 354 bits of secret information. GOST is a simple encryption algorithm, where the round function 32-bit subkey modulo 232 is added and put in the layer of S-boxes and the rotate left shift operation is used for shifting 11 bits, thereby providing the output of the round function.

The key scheduling of the GOST block cipher is performed by breaking the 256-bit key into eight 32-bit subkeys, where each subkey is used four times. In this algorithm, the key words are used in order for the first 24 rounds and they are used in reverse order for the last 8 rounds. Kuznyechik is the latest extension of GOST, which uses 128-bit blocks.

**Camellia**

Camellia is a symmetric-key block cipher having either 18 rounds (for 128-bit keys) or 24 rounds (for 256-bit keys). It is a Feistel cipher with a block size of 128 bits and a key size of 128, 192, and 256 bits. Camellia uses four 8×8-bit S-boxes that perform affine transformations and logical operations. A logical transformation layer FL-function or its inverse is applied every six rounds. Camellia uses the key whitening technique for increased security. Camellia is a part of the Transport Layer Security (TLS) protocol, which is used to deliver secure communication. Camellia cannot be brute-forced even with the latest technology although it uses a smaller key size of 128 bits, thus making it a safe cipher.

**DSA and Related signature scheme**

The Digital Signature Algorithm (DSA) is a Federal Information Processing Standard for digital signatures. The NIST proposed the DSA for use in the Digital Signature Standard (DSS), adopted as FIPS 186. It creates a 320-bit digital signature with 512–1024-bit security. A digital signature is a mathematical scheme used for the authentication of digital messages.

Processes involved in DSA:

* **Signature Generation Process:** The private key is used to know who has signed it.
* **Signature Verification Process:** The public key is used to verify whether the given digital signature is genuine.

DSA Algorithm:

Each entity A does the following:

* Select a prime number q such that 2159 < q < 2160
* Choose t such that 0 ≤ t ≤ 8, and select a prime number p where 2511+64t<p< 2512+64twith the property that q divides (p-1)
* Select a generator α of the unique cyclic group of order q in Z\*p g ∈Z\*p and then computing α = g(p−1)/q mod p until α ≠1
* Select a random integer d such that 1 ≤ d ≤ q-1
* Compute y = αd mod p
* A’s public key is (p, q, α, y); A’s private key is d.

**Rivest Shamir Adleman (RSA)**

Ron Rivest, Adi Shamir, and Leonard Adleman formulated RSA, a public-key cryptosystem for Internet encryption and authentication. RSA uses modular arithmetic and elementary number theories to perform computations using two large prime numbers.

RSA works as follows:

* Two large prime numbers are taken (a and b), and their product is determined (c = ab, where “c” is called the modulus).
* RSA chooses a number “e” that it is less than “c” and relatively prime to (a-1)(b-1). Therefore, e and (a-1)(b-1) have no common factor except 1.
* Furthermore, RSA chooses a number “f” such that (ef - 1) is divisible by (a-1)(b-1).
* The values “e” and “f” are the public and private exponents, respectively.
* The public key is the pair (c, e); the private key is the pair (c, f).
* It is difficult to obtain the private key (c, f) from the public key (c, e). However, if someone can factor “c” into “a” and “b”, then that person can decipher the private key (c, f).

RSA Signature Scheme

The RSA signature scheme is the first technique used to generate digital signatures. It is a deterministic digital signature scheme that provides message recovery from the signature itself, making it the most practical and versatile technique available.

RSA Key Generation

* Generate two large distinct primes p and q arbitrarily, each with roughly the same bit length
* Compute n = pq and k = (p-1)(q-1)
* Choose a random integer e, 1 < e < k such that gcd(e, k) = 1
* Use the extended Euclidean algorithm to compute the unique integer d, 1 < d < k, such that ed ≡ 1 (mod k)
* A’s public key is (n, e); A’s private key is d

Destroy p and q at the end of the key generation

**Diffie–Hellman**

It is a cryptographic protocol that allows two parties to establish a shared key over an insecure channel. It was developed and published by Whitfield Diffie and Martin Hellman in 1976.

Algorithm:

* Alice generates a random private value a, and Bob generates a random private value b. Both a and b are drawn from the set of integers
* They derive their public values using parameters p and g and their private values. Alice's public value is ga mod p, and Bob's public value is gb mod p.
* They exchange their public values
* Alice computes gab = (gb)a mod p, and Bob computes gba = (ga)b mod p
* Since gab = gba = k, Alice and Bob now have a shared secret key k

**YAK**

YAK is a public-key-based Authenticated Key Exchange (AKE) protocol. The authentication of YAK is based on public key pairs, and it needs PKI to distribute authentic public keys. YAK is a variant of the two-pass Hashed Menezes‐Qu‐Vanstone (HMQV) protocol using zero‐knowledge proofs (ZKP) for proving the knowledge of ephemeral secret keys from both parties. The YAK protocol lacks joint key control and perfect forward secrecy attributes.

Algorithm:

* Alice chooses a random number x such that x∈ R[0, q − 1], computes X = gx, and generates ZKP of x, denoted by KP{x}. Alice sends X and KP{x} to Bob.
* Bob chooses a random number y such that y∈ R[0, q − 1], computes Y = gy, and generates ZKP of y, denoted by KP{y}. Bob sends Y and KP{y} to Alice.
* Alice verifies the received KP{x} and computes the session key after verification as k = H ((Y.PKB)x + a), where H is a hash function.
* Bob verifies the received KP{y} and computes the session key after verification as k = H ((X.PKA)y + b).
* They authenticate each other, and both obtain the same session key k = H (g(x + a)(y + b)).

**Message Digest (One-way Hash) Functions**

Hash functions calculate a unique fixed-size bit string representation, called a message digest, of any arbitrary block of information. Message digest functions distill the information contained in a file (small or large) into a single fixed-length number, typically between 128 and 256 bits. If any given bit of the function’s input is changed, every output bit has a 50% chance of changing.

Message digest functions are also called one-way hash functions because they produce values that are nearly impossible to invert, resistant to attack, mostly unique, and widely distributed. Message digest algorithms themselves do not participate in encryption and decryption operations. They allow the creation of digital signatures and message authentication codes (MACs) as well as the derivation of encryption keys from passphrases. The main role of a cryptographic hash function is to provide integrity in document management. Their characteristic feature is to calculate the signature of the document’s hash value, which is smaller than the document.

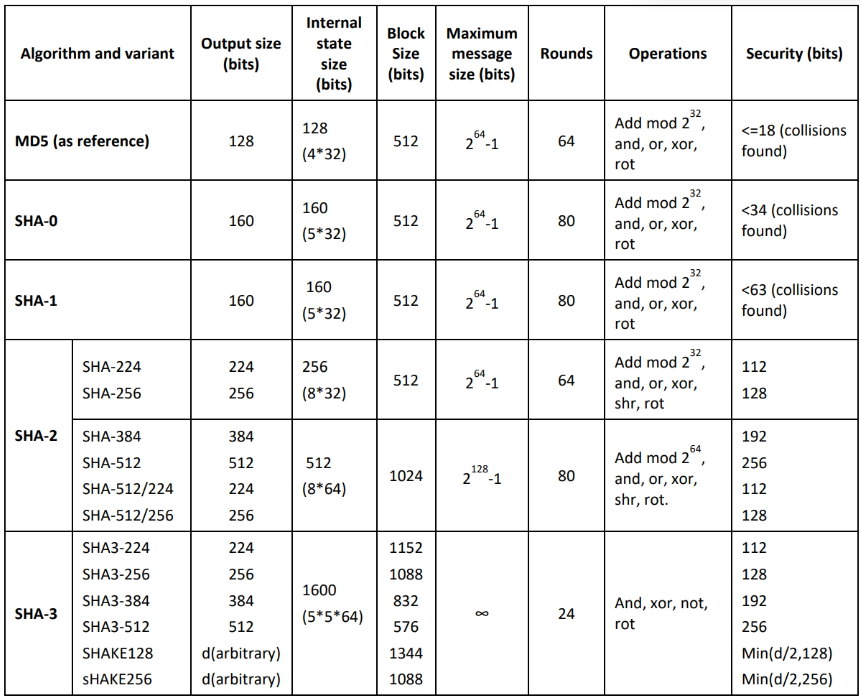
**Message Digest Function: MD5 and MD6**

MD2, MD4, MD5, and MD6 are message digest algorithms used in digital signature applications to compress a document securely before the system signs it with a private key. The algorithms can be of variable length, but the resulting message digest always has a size of 128 bits.The structures of all three algorithms (MD2, MD4, and MD5) appear similar, although the design of MD2 is reasonably different from that of MD4 and MD5. MD2 supports 8-bit machines, while MD4 and MD5 support 32-bit machines. The algorithm pads the message with extra bits to ensure that the number of bits is divisible by 512. The extra bits may include a 64-bit binary message.

MD5 is a widely used cryptographic hash function that takes a message of arbitrary length as input and outputs a 128-bit (16-byte) fingerprint or message digest of the input.

MD6 uses a Merkle-tree-like structure to allow for large-scale parallel computation of hashes for very long inputs. It is resistant to differential cryptanalysis attacks.

**Message Digest Function: Secure Hashing Algorithm (SHA)**

The NIST has developed the Secure Hash Algorithm (SHA), specified in the Secure Hash Standard (SHS) and published as a federal information-processing standard (FIPS PUB 180). It generates a cryptographically secure one-way hash. Rivest developed the SHA, which is similar to the message digest algorithm family of hash functions. It is slightly slower than MD5 

**RIPEMD-160**

RACE Integrity Primitives Evaluation Message Digest (RIPEMD) is a 160-bit hash algorithm developed by Hans Dobbertin, Antoon Bosselaers, and Bart Preneel. There exist 128-, 256-, and 320-bit versions of this algorithm, called RIPEMD-128, RIPEMD-256, and RIPEMD-320, respectively. These algorithms replace the original RIPEMD, which was found to have a collision issue. RIPEMD-160 is a more secure version of the RIPEMED algorithm. In this algorithm, the compression function consists of 80 stages, i.e., 5 blocks that execute 16 times each. This process repeats twice by combining the results at the bottom using modulo 32 additions.

**HMAC**

Hash-based message authentication code (HMAC) is a type of message authentication code (MAC) that uses a cryptographic key along with a cryptographic hash function. It is widely used to verify the integrity of data and authentication of a message. This algorithm includes an embedded hash function such as SHA-1 or MD5. The strength of HMAC depends on the embedded hash function, key size, and size of the hash output. HMAC includes two stages for computing the hash. The input key is processed to produce two keys, namely the inner key and the outer key. The first stage of the algorithm inputs the inner key and message to produce an internal hash. The second stage of the algorithm inputs the output from the first stage and outer key, and produces the final HMAC code. As HMAC executes the underlying hash function twice, it offers protection against various length extension attacks. The size of the key and the output depends on the embedded hash function; e.g., 128 or 160 bits in the case of MD5 or SHA-1, respectively.

**CHAP**

The Challenge-Handshake Authentication Protocol (CHAP) is an authentication mechanism used by Point-to-Point Protocol (PPP) servers to authenticate or validate the identity of remote clients or network hosts. It is more secure and effective compared to Password Authentication Procedure (PAP), as it regularly verifies the identity of the client using a three-way handshake and provides protection against replay attacks.

**EAP**

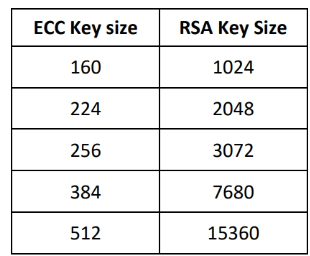
The Extensible Authentication Protocol (EAP) is an authentication protocol that was originally designed for point-to-point connections. It is used as an alternative to the CHAP and PAP authentication protocols, as it is more secure and supports different authentication mechanisms such as passwords, smart tokens, one-time passwords (OTPs), secure ID card, digital certificates, and public-key encryption mechanisms. After the selection of the EAP authentication mechanism, a session is established and messages are exchanged between the client and the authenticating server.

**GOST – Hash Function**

This hash algorithm was initially defined in the Russian national standard GOST R 34.11-94 “Information Technology - Cryptographic Information Security - Hash Function.” It produces a fixed-length output of 256 bits. The input message is broken up into chunks of 256-bit blocks. If a block is less than 256 bits, then the message is padded by appending as many zeros to it as are required to make the length of the message 256 bits. The remaining bits are filled with a 256-bit integer arithmetic sum of all previously hashed blocks.

**Elliptic Curve Cryptography (ECC)**

ECC is a modern public-key cryptography developed to avoid larger cryptographic key usage. The asymmetric cryptosystem depends on number theory and mathematical elliptic curves (algebraic structure) to generate short, quick, and robust cryptographic keys. RSA is an incumbent public-key algorithm, but its key size is large. The speed of the encryption always depends on the key size: a smaller key length allows faster encryption.



**Quantum Cryptography**

Quantum cryptography has been introduced to protect data from theft midway (e.g., MITM attacks). This cryptography is processed based on quantum mechanics, such as quantum key distribution (QKD), using photons instead of mathematics as a part of encryption. In quantum cryptography, the data are encrypted by a sequence of photons that have a spinning trait while traveling from one end to another end. These photons keep changing their shapes during their course through filters: vertical, horizontal, forward slash, and backslash. Here, vertical and backslash spins imply “ones,” while horizontal and forward slash spins imply “zeros.”

**Homomorphic Encryption**

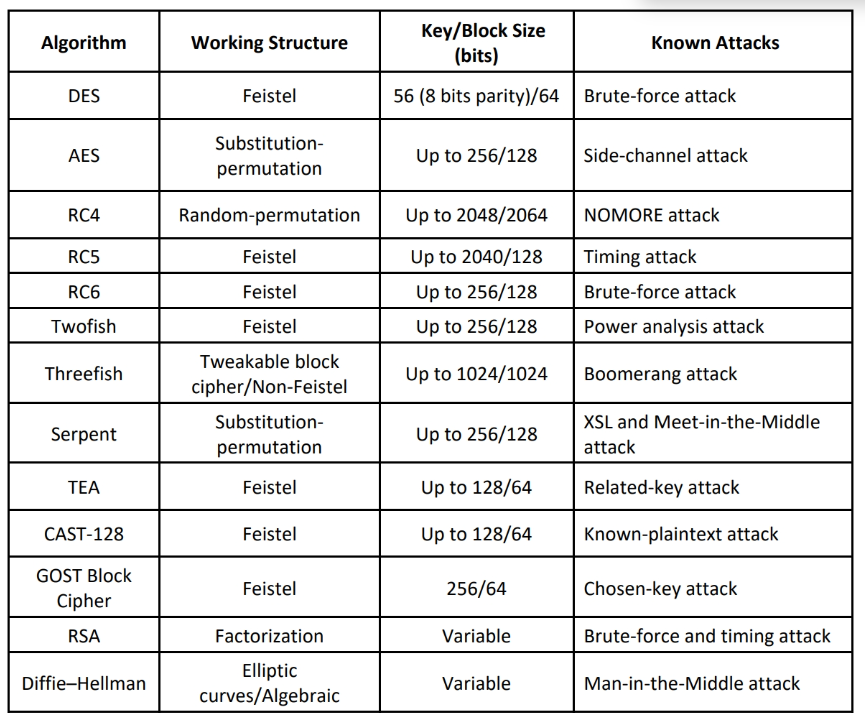
Homomorphic encryption differs from conventional encryption mechanisms, where math operations are performed to encrypt the plaintext. Homographic encryption allows users to secure and leave their data in an encrypted format even while it is being processed or manipulated. In this technique, encryption and decryption are performed by the same key holder. The homomorphic mechanism enables the user/sender to encrypt the confidential data and out-source it to an enterprise via cloud services to process the given data.

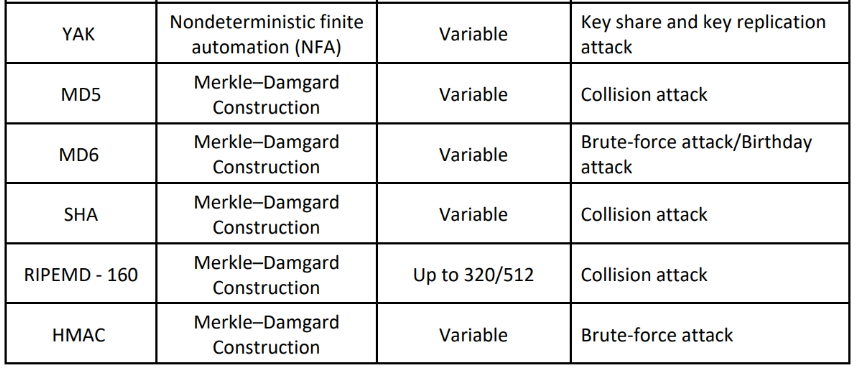
**Hardware-Based Encryption**

Hardware-based encryption is a technique that uses computer hardware for assisting or replacing the software when the data encryption process is being performed. Devices that offer encryption techniques can be considered as hardware-based encryption devices. In the implementation of hardware-based encryption, the cryptography technique workload is transferred to the hardware processors, making the system resources free for performing other functions.

Types of hardware encryption devices

* **TPM:** Trusted Platform Module (TPM) is a crypto-processor or a chip that is present in the motherboard. It can securely store the encryption keys and perform many cryptographic operations.
* **HSM:** A hardware security module (HSM) is an additional external security device that is used in a system for crypto-processing, and it can be used for managing, generating, and securely storing cryptographic keys. HSM offers enhanced encryption computation that is useful for symmetric keys longer than 256 bits.
* **USB Encryption:** USB encryption is an additional feature for USB storage devices, which offers onboard encryption services. Encrypted USB devices need an on-device credential system or software-or hardware-based credentials from a computer. USB encryption provides protection against malware distribution over USB and helps in preventing data loss and data leakage.
* **Hard Drive Encryption:** Hard drive encryption is a technology whereby the data stored in the hardware can be encrypted using a wide range of encryption options. Hard drive encryption devices cannot use an on-device keyboard or fingerprint reader; instead, they need a TPM or an HSM.





**🡺 Public Key Infrastructure (PKI)**

PKI is a security architecture developed to increase the confidentiality of information exchanged over the insecure Internet. It includes hardware, software, people, policies, and procedures required to create, manage, distribute, use, store, and revoke digital certificates. In cryptography, the PKI helps to bind public keys with corresponding user identities by means of a certification authority (CA). PKI is a comprehensive system that allows the use of public-key encryption and digital signature services across a wide variety of applications. PKI authentication depends on digital certificates (also known as public-key certificates) that CAs sign and provide. A digital certificate is a digitally signed statement with a public key and the subject (user, company, or system) name in it

In public-key cryptography, a CA generates public and private keys with the same algorithm simultaneously. The private key is held only by the subject (user, company, or system) mentioned in the certificate, while the public key is made publicly available in a directory that all parties can access. The subject keeps the private key secret and uses it to decrypt the text encrypted by someone else using the corresponding public key (available in a public directory).

Components of PKI

* Certificate Management System: Generates, distributes, stores, and verifies certificates
* Digital Certificates: Establishes credentials of a person when performing online transactions
* Validation Authority (VA): Stores certificates (with their public keys)
* Certification Authority (CA): Issues and verifies digital certificates
* End User: Requests, manages, and uses certificates
* Registration Authority (RA): Acts as the verifier for the CA

The steps involved in the PKI process are as follows:

* The subject (user, company, or system) intending to exchange information securely applies for a certificate to the registration authority (RA).
* The RA receives the request from the subject, verifies the subject’s identity, and requests the CA to issue a public key certificate to the user.
* The CA issues the public key certificate binding the subject’s identity with the subject’s public key; then, the updated information is sent to the validation authority (VA).
* When a user makes a transaction, the user duly signs the message digitally using the public key certificate and sends the message to the client.
* The client verifies the authenticity of the user by inquiring with the VA about the validity of the user’s public key certificate.
* The VA compares the public key certificate of the user with that of the updated information provided by the CA and determines the result (valid or invalid).

**Certification Authorities**

Certification authorities (CAs) are trusted entities that issue digital certificates. The digital certificate certifies the possession of the public key by the subject (user, company, or system) specified in the certificate. This aids others to trust signatures or statements made by the private key that is associated with the certified public key.

* **Comodo** offers a range of PKI digital certificates with strong SSL encryption (128/256 available) with Server-Gated Cryptography (SGC). It ensures standards of confidentiality, system reliability, and pertinent business practices as judged via qualified independent audits.
* **Symantec Corporation (NASDAQ: SYMC)** provides solutions that allow companies and consumers to engage in communications and commerce online with confidence. Symantec offers SSL/TLS certificates.
* **IdenTrust** is a trusted third party that provides CA services for many sectors such as banks,corporates, governments, and healthcare. It provides solutions such as digital signing and sealing, compliance with NIST SP 800-171, global identity networks, and managed PKI hosting service

**Signed Certificate**

CAs sign and issue signed certificates. These certificates contain a public key and the identity of the owner. The corresponding private key is not made publicly available; instead, it is kept secret by the authorized user. By issuing the certificate, the CA confirms or validates that the public key contained in the certificate belongs to the person, company, server, or other entity mentioned in the certificate. CA verifies an application’s credentials; thus, users and relying parties trust the information in the CA’s certificates.

**Self-Signed Certificate**

A self-signed certificate is an identity certificate signed by the same entity whose identity it certifies. In general, self-signed certificates are widely used for testing servers. In a self-signed certificate, a user creates a pair of public and private keys using a certificate creation tool such as Adobe Acrobat Reader, Java's keytool, and Apple's Keychain, and signs the document with the public key. The receiver requests the sender for the private key to verify the certificate.

**🡺 Email Encryption**

**Digital Signature**

A digital signature uses asymmetric cryptography to simulate the security properties of a signature in digital form rather than in written form. A digital signature is a cryptographic means of authentication. Public-key cryptography uses asymmetric encryption and helps the user to create a digital signature.

A hash function is an algorithm that helps a user to create and verify a digital signature. This algorithm creates a digital representation, also known as the message fingerprint. This fingerprint has a hash value that is much smaller than the message. To verify the digital signature, one needs the hash value of the original message and the encryption algorithm used to create the digital signature.

**Secure Sockets Layer (SSL)**

The Secure Sockets Layer (SSL) protocol is an application layer protocol developed by Netscape for managing the security of message transmission on the Internet. It is used to provide a secure authentication mechanism between two communicating applications, such as a client and a server. SSL requires a reliable transport protocol, such as TCP, for data transmission and reception. It uses RSA asymmetric (public-key) encryption to encrypt data transferred over SSL connections. Any application-layer protocol that is higher than SSL, such as HTTP, FTP, and telnet, can form a transparent layer over SSL. SSL acts as an arbitrator between the encryption algorithm and the session key.

SSL uses both asymmetric and symmetric authentication mechanisms. Public-key encryption verifies the identities of the server, the client, or both. Once authentication has occurred, the client and server can create symmetric keys, allowing them to communicate and transfer data rapidly.

SSL also offers “channelsecurity” with three basic properties:

* Private channel – All the messages are encrypted after a simple handshake is used to define a secret key.
* Authenticated channel – The server endpoint of the conversation is always encrypted, whereas the client endpoint is optionally authenticated.
* Reliable channel – Message transfer has an integrity check.

**Transport Layer Security (TLS)**

The Transport Layer Security (TLS) protocol is used to establish a secure connection between a client and a server and ensure the privacy and integrity of information during transmission. It uses a symmetric key for bulk encryption, asymmetric key for authentication and key exchange, and message authentication codes for message integrity. It uses the RSA algorithm with strengths of 1024 and 2048 bits.

TLS consists of two layers:

* **TLS Record Protocol:** The TLS Record Protocol is a layered protocol. It provides secured connections with an encryption method such as DES.
* **The connection is private:** Uses symmetric cryptography for data encryption (e.g., DES). The protocol generates unique keys for symmetric encryption for each connection, depending on a secret negotiated by another protocol.
* **The connection is reliable:** It provides a message integrity check at the time of message transport using a keyed MAC.

The TLS Record Protocol does the following:

* Fragments outgoing data into manageable blocks and reassembles incoming data
* Optionally compresses outgoing data and decompresses incoming data
* Applies MAC to the outgoing data and uses MAC to verify the incoming data
* Encrypts outgoing data and decrypts incoming data
* **TLS Handshake Protocol:** The TLS Handshake Protocol allows the client and server to authenticate each other and select an encryption algorithm and cryptographic keys prior to data exchange by the application protocol. The TLS Handshake Protocol operates on top of the TLS Record Protocol and is responsible for producing cryptographic parameters of the session state.

It provides connection security with three basic properties:

* The peer’s identity can be authenticated using asymmetric cryptography. This can be made optional but is mostly required for at least one of the peers.
* The negotiation of a shared secret is secure.
* The negotiation is reliable.

**Cryptography Toolkits**

Cryptography toolkits include cryptographic primitives, algorithms, and schemes used to provide security for various applications.

* **OpenSSL** is an open-source cryptography toolkit implementing the SSL and TLS network protocols and the related cryptography standards required by them. It is a command-line tool for using the various cryptography functions of OpenSSL’s crypto-library from the shell.

**Pretty Good Privacy (PGP)**

Pretty Good Privacy (PGP) is a protocol used to encrypt and decrypt data with authentication and cryptographic privacy. It is often used for data compression, digital signing, encryption and decryption of messages, emails, files, and directories, and to enhance the privacy of email communications. The algorithm used for message encryption is RSA for key transport and IDEA for bulk-message encryption. PGP uses RSA for computing digital signatures and MD5 for computing message digests.

* Encrypting a message or file prior to transmission so that only the recipient can decrypt and read it
* Clear signing of the plaintext message to ensure the authenticity of the sender
* Encrypting stored computer files so that no one besides the person who encrypted them can decrypt them
* Deleting files rather than just removing them from the directory or folder
* Data compression for storage or transmission

PGP Encryption Decryption

* First PGP compresses the data
* Then it creates random key that is one-time-only secret key
* Encrypt the text using random key
* Then random key is encrypted using recipients public key
* Encrypted random key and cyphertext both are sent to the recipient
* Decryption is just reverse of encryption

**GNU Privacy Guard (GPG)**

GNU Privacy Guard (GPG) is a software replacement of PGP and free implementation of the OpenPGP standard that is used to encrypt and decrypt data. GPG is also called a hybrid encryption software program, as it uses both symmetric-key cryptography and asymmetric-key cryptography for improved speed and secure key exchange, which is achieved using the receiver’s public key for encrypting the session key. GPG also supports S/MIME and Secure Shell (SSH). The latest version of GPG supports most cryptographic functions such as elliptic curve cryptography (ECDSA, ECDH, and EdDSA), and it also supports the cryptography library Libgcrypt.

GPG Encryption

* GPG encrypts messages individually by using asymmetric-key pairs. o The user sends the raw file, and
* GPG is used for signing the file using the sender’s private key for confirming the file content at the time of signing.
* Then, the file is encrypted using the receiver’s public key. Now, the file can be decrypted only with the receiver’s private key.
* After encrypting the data, the encrypted file can be stored locally, distributed to the FTP servers, or sent to email recipients.
* Decryption is just reverse of encryption.

**Web of Trust (WOT)**

Web of trust (WoT) is a trust model of PGP, OpenPGP, and GnuPG accessible systems. It is an idea of decentralizing the key distribution among PGP users. In the PKI, only a centralized power such as the CA signs certificates in the network, ensuring authenticity between the public key and its owner. In WoT, everyone in the network is a CA, and they can sign for other trusted entities. WoT is a network chain in which individuals intermediately validate each other’s certificates using their signatures. These signatures verify the ownership of keys from various trust levels.

In WOT, every PGP user in the network has a ring of public keys to encrypt the data, and they introduce many other users whom they trust. In this trust model, a user encodes the data with the receiver’s public key that is decrypted only by the receiver’s private key. Then, every user in this model digitally signs the data with their private keys; when the recipient is validating it against the user’s public key, he/she can confirm the user’s authenticity.

**🡺 Disk Encryption**

Disk encryption encrypts every bit of data stored on a disk or a disk volume, thus preventingillegal access to data storage. This section deals with disk encryption concepts and various disk encryption tools. Disk encryption is a technology that protects the confidentiality of the data stored on a disk by converting it into an unreadable code using disk encryption software or hardware, thus preventing unauthorized users from accessing it. Disk encryption provides confidentiality and privacy using passphrases and hidden volumes. Disk encryption works similarly to text-message encryption and protects data even when the OS is not active.

Disk encryption is useful when the user needs to physically send sensitive information. In addition, disk encryption can protect the real-time exchange of information from compromising threats.

**🡺 Cryptanalysis**

Attackers may implement various cryptography attacks to evade the security of a cryptographicsystem by exploiting vulnerabilities in code, ciphers, cryptographic protocols, or key management schemes. This process is known as cryptanalysis. Cryptanalysis is the study of ciphers, ciphertext, or cryptosystems with the ability to identify vulnerabilities in them and thus extract plaintext from ciphertext even if the cryptographic key or algorithm used to encrypt the plaintext is unknown.

**Cryptanalysis Methods**

* **Linear cryptanalysis** is based on finding affine approximations to the action of a cipher. It is commonly used on block ciphers. This technique was invented by Mitsarue Matsui. It is a known plaintext attack and uses a linear approximation to describe the behavior of the block cipher. Given sufficient pairs of plaintexts and corresponding ciphertext, bits of information about the key can be obtained. cryptanalysis is an attempt to crack cryptography. For example, with the 56-bit data encryption standard (DES), key brute-forcing could take up to 256 attempts. Linear cryptanalysis requires 243 known plaintexts.
* **Differential cryptanalysis** is a form of cryptanalysis applicable to symmetric-key algorithms. It was invented by Eli Biham and Adi Shamir. Essentially, it is the examination of differences in input and how that affects the resultant difference in the output. The differential analysis looks at pairs of inputs that differ in only one bit position, with all other bits being identical.
* **Integral cryptanalysis** was first described by Lars Knudsen. This attack is particularly usefulagainst block ciphers based on substitution-permutation networks as an extension of differential cryptanalysis. Integral analysis for block size b holds b-k bits constant and runs the other k bits through all 2k possibilities. For k = 1, this is just differential cryptanalysis, but with k > 1, it is a new technique.

**Code Breaking Techniques**

* **Brute Force:** Code breakers or cryptanalysts work to recover the plaintext of a message without knowing the required key in advance. They may first try to recover the key, or they may go after the message itself. A common cryptanalytic technique is a brute-force attack, or exhaustive search, in which the keys are determined by trying every possible combination of characters.
* **Frequency analysis** is the study of the frequency of letters or groups of letters in a ciphertext. Frequency analysis of letters and words is another method used to crack ciphers. It works on the principle that, in any given stretch of written language, certain letters and combinations of letters occur with varying frequencies. This technique examines the number of times that a particular symbol appears in a ciphertext. Encrypted source code is more vulnerable to these types of attacks because words such as “#define,” “struct,” “else,” and “return” are repeated frequently in code.
* **Trickery and deceit** require a high level of mathematical and cryptographic skills. It involves the use of social engineering techniques to extract cryptography keys.
* **One-time pad** mostly contains a non-repeating set of letters or numbers, which the system chooses randomly. The user writes them on small sheets of paper and then pastes them together in a pad. The key length is the same as that of the message, thus making it impossible to encrypt and send large messages.

**Cryptography Attacks**

* **Ciphertext-only Attack:** It is less effective but much more likely for the attacker. The attacker only has access to a collection of ciphertexts. This is much more likely than known plaintext but is also the most difficult. The attack is completely successful if the corresponding plaintexts (or even better, the key) can be deduced.
* **Adaptive Chosen-plaintext Attack:** An attacker has complete access to the plaintext message including its encryption, and he/she can also modify the content of the message by making a series of interactive queries, choosing subsequent plaintext blocks based on the information from the previous encryption queries and functions.
* **Chosen-Plaintext Attack:** the attacker obtains the ciphertexts corresponding to a set of plaintexts of his/her own choosing. This allows the attacker to attempt to derive the key used and thus decrypt other messages encrypted with that key. Basically, since the attacker knows the plaintext and the resultant ciphertext, he/she gains many insights into the key used.
* **Related-key Attack:** It is similar to the chosen plaintext attack, except that the attacker can obtain ciphertexts encrypted under two different keys. The attack requires that the differing keys be closely related.
* **Dictionary Attack:** The attacker constructs a dictionary of plaintext along with its corresponding ciphertext that he/she has analyzed and obtained for a certain period of time. After building the dictionary, if the attacker obtains the ciphertext, he/she uses the already built dictionary to find the corresponding plaintext.
* **Known-plaintext Attack:** In this attack, the only information available to the attacker is some plaintext blocks along with the corresponding ciphertext and algorithm used to encrypt and decrypt the text. Using this information, the key used to generate the ciphertext is deduced so as to decipher other messages.
* **Chosen-ciphertext Attack:** The attacker obtains the plaintexts corresponding to an arbitrary set of ciphertexts of his own choosing. Using this information, the attacker tries to recover the key used to encrypt the plaintext. To perform this attack, the attacker must have access to the communication channel between the sender and the receiver.
* **Rubber Hose Attack:** Attackers extract cryptographic secrets (e.g., the password to an encrypted file) from a person by coercion or torture.
* **Chosen-key Attack:** In this type of attack, an attacker not only breaks a ciphertext but also breaks into a larger system, which is dependent of that ciphertext. The attacker usually breaks an n-bit key cipher into 2n/2 operations. Once an attacker breaks the cipher, he gets access to the system.
* **Timing Attack:** It is based on repeatedly measuring the exact execution times of modular exponentiation operations. The attacker tries to break the ciphertext by analyzing the time taken to execute the encryption and decryption algorithm for various inputs.
* Man-in-the-Middle Attack: This attack is performed against a cryptographic protocol. Here, an attacker intercepts the communication between a client and a server and negotiates the cryptographic parameters. An attacker can also inject commands that can modify the data in transit.

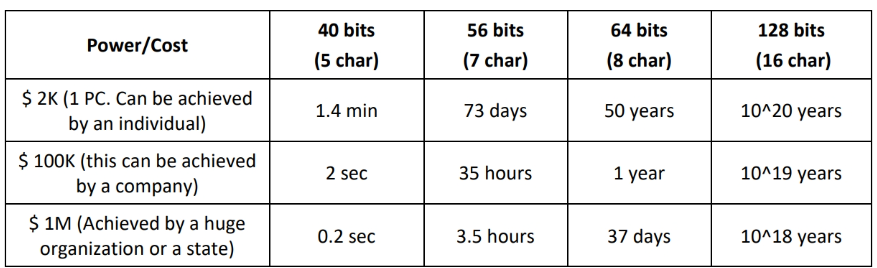
**Brute-Force Attack**

Cryptographic algorithms use a key to encrypt or decrypt messages. In cryptography, this key is the important parameter that specifies the transformation of plaintext to ciphertext and vice versa. If you are able to guess or find the key used for decryption, then you can decrypt the messages and read them in clear text. 128-bit keys are common and considered strong.

You can attempt to decrypt a message using all possible keys until you discover the key used for encryption. This method of discovering a key is called a brute-force attack. However, doing so requires a massive amount of processing power. It is a resource-intensive and time-intensive process. For any non-flawed protocol, the average time needed to find the key in a brute-force attack depends on the length of the key.

The difficulty of a brute-force attack depends on various factors, such as

* The length of the key
* The number of possible values each component of the key can have
* The time it takes to attempt each key
* If there is any mechanism that locks the attacker out after a certain number of failed attempts

If a system could brute-force a DES 56-bit key in one second, then for an AES 128-bit key, it takes approximately 149 trillion years. 

**Birthday Attack**

A birthday attack refers to a class of brute-force attacks against cryptographic hashes that renders brute-forcing easier to performs. This attack depends on the birthday paradox, which is the probability of two or more people in a group of 23 sharing the same birthday is greater than 0.5.

The basic idea is as follows: How many people would you need to have in a room to have a strong likelihood that two amongst them would have the same birthday (same day and month). Obviously, if you put 367 people in a room, at least two of them must have their birthdays on the same day and month since there are only 365 days in a year, and an additional day in the case of a leap year. The paradox is not the number of people you need to guarantee a match, but the number of people you need to have a strong probability. Even with 23 people in a room, there is a 50% chance that two them will have their birthdays on the same day and month.

**Birthday Paradox: Probability**

The probability that the first person does not share a birthday with any previous person is 100% because there are no previous people in the set. This can be written as 365/365. The second person has only one preceding person, and the odds that the second person has a birthday different from the first are 364/365. we can compute the probability as follows: 365/365 \* 364/365 \* 363/365 \* 362/365 ... \* 342/365 (342 is the probability of the 23rd person who shares a birthday with a preceding person). When we convert these to decimal values, it yields (truncating at the third decimal point) 1 \* 0.997 \* 0.994 \* 0.991 \* 0.989 \* 0.986 \* ... 0.936 = 0.49 or 49%. This is the probability that 23 people will not have any birthdays in common; thus, there is a 51% (better than even odds) chance that two of the 23 will have a birthday in common.

The idea behind the birthday attack is to attempt to find a collision for a given hash. Now, assume that the hash is MD5 with a 128-bit output. You would have to try 2^128 possible hashes to guarantee a collision, which is a very large number. In decimal notation, it is 3.4028236692093846346337460743177e+38Now, from the birthday paradox, we need 1.174√2^128 or 21656477542535013597.184 hashes to guarantee a collision.

**Meet-in-the-Middle Attack on Digital Signature Schemes**

A meet-in-the-middle attack is the best attack method for cryptographic algorithms using multiple keys for encryption. This attack reduces the number of brute-force permutations required to decode text encrypted by more than one key. A meet-in-the-middle attack uses space-time trade-off; it is also a type of birthday attack because it exploits the mathematics behind the birthday paradox, and the attack consumes less time than an exhaustive attack. It is called a meet-in-the-middle attack because it works by encrypting from one end and decrypting from the other end. The attacker has access to both the plaintext as well as the respective encrypted text. This attack is performed by attackers for forging messages that use multiple encryption schemes.

**Hash Collision Attack**A hash collision attack is performed by finding two different input messages that result in the same hash output. For example, in a hash collision attack, “hash(a1) = hash(a2)”, where a1 and a2 represent some random messages. Since the algorithm itself randomly selects these messages, attackers have no role in the content of these messages. One of the most popular hash functions is SHA-1, which is widely used as a digital signature algorithm. SHA-1 converts an input message into a constant length of unstructured strings of numbers and alphabets, which act as a fingerprint for the sent file. Therefore, the attacker tries to identify similar hashed output to get the digital signatures of the victim.

**DUHK Attack**

Don't Use Hard-Coded Keys (DUHK) is a cryptographic vulnerability that allows attackers to obtain encryption keys used to secure VPNs and web sessions. This attack mainly affects any hardware/software using the ANSI X9.31 Random Number Generator (RNG). Pseudorandom number generators (PRNGs) algorithm generates cryptographic keys that are used to establish a secure communication channel over the VPN. In some cases, the seed key is hardcoded into the implementation. Both the factors are key issues of the DUHK attack, as any attacker can combine ANSI X9.31 with the hard-coded seed key to decrypt the encrypted data sent or received by that device. Man-in-the-middle attackers use the DUHK attack to learn the seed value, observe the current session, and obtain the current state value.

**Rainbow Table Attack**

A rainbow table attack is a type of cryptography attack whereby an attacker uses a rainbow table for reversing cryptographic hash functions. A rainbow table attack uses the cryptanalytic time-memory trade-off technique, which is less time consuming than other techniques. In the rainbow table attack, the attacker creates a table of all the possible passwords and their respective hash values, called a rainbow table, in advance. A rainbow table contains word lists such as dictionary files and brute-force lists and their hash values. It is a lookup table particularly used for recovering a plaintext password from a ciphertext. An attacker computes the hash for a list of possible passwords and compares it with the pre-computed hash table (rainbow table). If a match is found, then he/she can crack the password.

**Related-Key Attack**

The failure in the WEP cryptogram, i.e., when used in wireless networks, is the best example of this attack. In this attack, each AP and user interface device uses the same key. The encryption used in WEP is a stream cipher known as RC4; it is important to note that the same keys should not be repeated in the stream cipher. To avoid this, WEP integrates a 24-bit initial vector (IV) in every packet transferred. The RC4 key for that particular packet is the IV associated with the WEP key. WEP keys need to be changed manually, however, this is rarely done. Hence, the attacker notices that the keys used for encryption are often the same. This drawback poses various risks on WEP, especially using the birthday paradox, because for every 4096 packets, two parties will share the same IV and hence the same RC4 key.

**Padding Oracle Attack**

In a padding oracle attack, the attackers exploit the padding validation of an encrypted message to decipher the ciphertext. Such an attack is also known as a Vaudenay attack. In many cryptographic algorithms based on a block cipher, the messages are padded with additional random bits so that the length of the last block is of the required size. Padding oracle is a function of such encryption that verifies if a message was correctly padded. This attack is mainly performed on algorithms that operate in the CBC (Cipher Block Chaining) mode. In this attack, the server (oracle) reveals information about whether the padding of an encrypted message was correctly done. In some cases, this information allows attackers to decrypt and optionally encrypt messages using the server’s key (oracle’s key) without having access to the corresponding encryption key.

**DROWN Attack**

Decrypting RSA with Obsolete and Weakened eNcryption (DROWN) is a grave vulnerability that can affect important cryptographic protocols such as HTTPS and other cryptographic services that depend on SSL and TSL. The DROWN attack is a cross-protocol weakness that can communicate and initiate an attack on servers supporting recent SSLv3/TLS protocol suites. It is a new form of cross-protocol Bleichenbacher padding oracle attack. The DROWN attack allows the attacker to decrypt the latest TLS connection between the victim client and the server by launching malicious SSLv2 probes using the same private key. Using this attack, the attacker can also force the victim client and server to use the RSA key exchange.

The server is critically vulnerable to the DROWN attack if

* The server permits SSLv2 connection, which is mostly caused by a misconfiguration or incorrect default settings.
* The same private key certificate is used on a different server that allows SSLv2 connection, and it also makes the TLS server vulnerable, as the SSLv2 server can leak the key information.

Attackers perform the DROWN attack as part of an online man-in-the-middle (MITM) attack, breaking encrypted keys, sniffing or stealing sensitive information such as passwords and bank account details, and accessing personal emails or messages. By performing this attack, the attacker can also masquerade as a secure website.

🡺 Key Stretching

Key stretching refers to processes used to make a weak key stronger, usually by making it longer. This technique helps in defending against brute-force attacks.

In the key stretching technique, the initial key is given as input to an algorithm that generates an enhanced key. The key must be sufficiently resistant to brute-force attacks.

There are many functions and libraries that perform key stretching as part of their working:

* Password-Based Key Derivation Function 2 (PBKDF2) is part of PKCS #5 v. 2.01. It applies some function (such as a hash or HMAC) to the password or passphrase along with Salt to produce a derived key.
* bcrypt is used with passwords, and it essentially uses a variant of the Blowfish algorithm, converted to a hashing algorithm, to hash a password and add Salt to it.