UNIT II

Introduction to Number Theory:

A number of concepts from number theory are essential in the design of public-key cryptographic algorithms

Prime Numbers:

- Prime numbers are a central concern of number theory.
- An integer is a prime number if its only divisors are 1 and itself.
- Any integer can be factored in a unique way as a product of prime numbers and their positive integer exponents and this is known as the fundamental theorem of arithmetic.

$$a = p_1^{a_1} \times p_2^{a_2} \times \cdots \times p_t^{a_t}$$

- Any positive integer can be written uniquely as a product over all possible prime numbers.

Fermat's and Euler's Theorems:

Fermat's Theorem: Fermat's theorem states that if p is prime and a is a positive integer not divisible by p, then

$$a^{p-1} \equiv 1 \pmod{p}$$

Euler's Theorem: Euler's theorem states that for every a and n that are relatively prime:

$$a^{\phi(n)} \equiv 1 \pmod{n}$$
$$\phi(n) = (n - 1)$$

Principles of Public Key Crypto System:

Public-key cryptography was developed to address two difficult problems associated with symmetric encryption: key distribution and digital signatures.

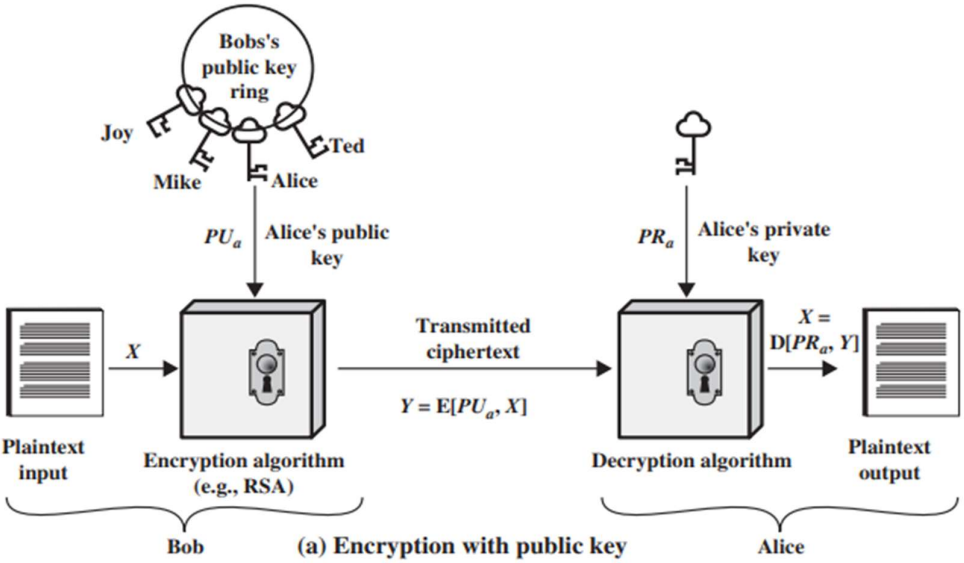
- Key distribution under symmetric encryption requires the two communicants to share a key or use a key distribution center, which may compromise the secrecy of their communication.
- Digital signatures are needed for electronic messages and documents to verify the sender's identity.
- Whitfield Diffie and Martin Hellman developed a method that addressed both problems and was radically different from all previous approaches to cryptography.
- Public-key cryptography uses a pair of keys, one public and one private, for encryption and decryption.
- The encryption and decryption algorithms are asymmetric and based on mathematical functions that are easy to compute in one direction but difficult to compute in the other direction.

Public-Key Cryptosystems:

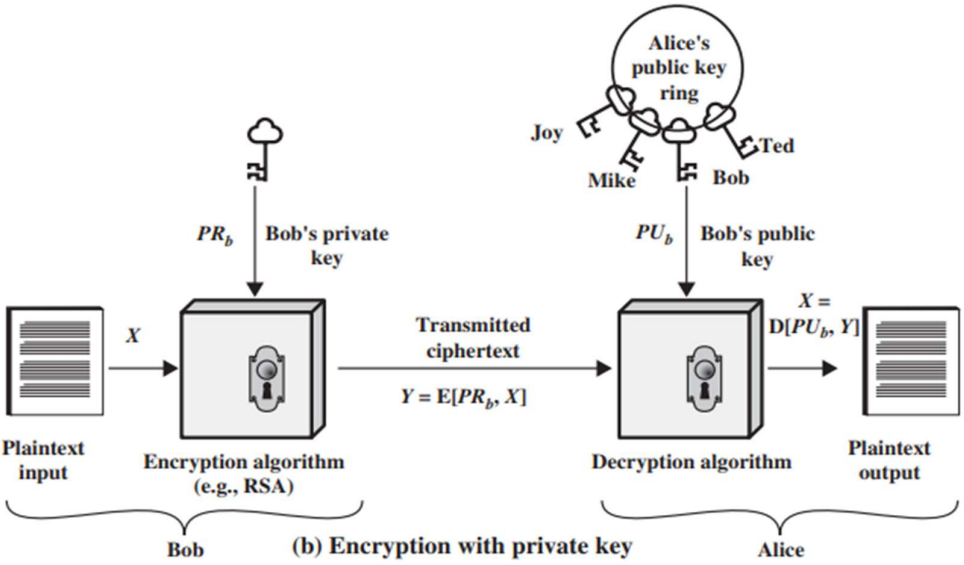
Public key cryptography uses two keys, a public key and a private key, to encrypt and decrypt messages.

- Anyone can use the public key to encrypt a message, but only the owner of the private key can decrypt it.
- This eliminates the need for secure distribution of secret keys, which is a challenge in symmetric key cryptography.

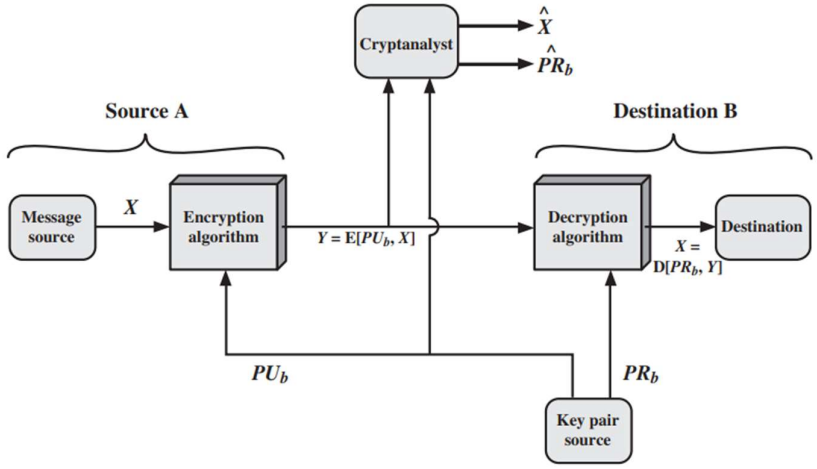
- In a public key encryption scheme, each user generates a pair of keys and places one key in a public registry while keeping the other key private.
- When someone wants to send a message to a user, they use the recipient's public key to encrypt the message.
- The recipient then uses their private key to decrypt the message.
- As long as the private key remains secure, incoming communication is secure.
- Conventional encryption uses the same algorithm and key for encryption and decryption, which must be shared by the sender and receiver.
- The key must be kept secret, and it should be impossible to decipher a message without the key.
- Public-key encryption uses a pair of keys, one for encryption and one for decryption.
- The sender and receiver must each have one of the matched pair of keys, and one of the keys must be kept secret.
- It should be impossible to decipher a message without the key, and knowledge of the algorithm plus one of the keys and samples of cipher text must be insufficient to determine the other key.

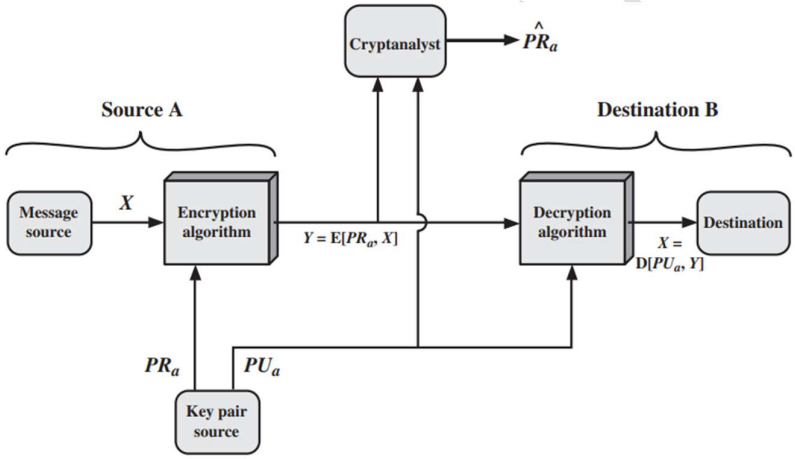


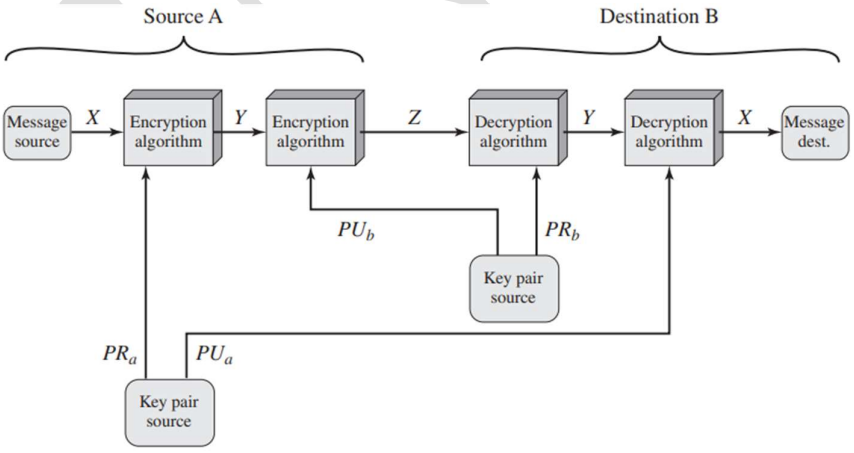
(a) Encryption with public key



(b) Encryption with private key







Applications for Public-Key Cryptosystems:
There are three main categories of using public-key cryptosystems:

- **Encryption/decryption:** The sender encrypts a message using the recipient's public key.
- **Digital signature:** The sender signs a message using their private key.
- **Key exchange:** Two sides cooperate to exchange a session key using their private keys.

Requirements for Public-Key Cryptography:
Diffie and Hellman proposed a cryptosystem that relies on a cryptographic algorithm with two related keys: a public key and a private key. They laid out the following requirements for such an algorithm:

- It should be easy for a party to generate a pair of keys: one public and one private.
- It should be easy for a sender to encrypt a message with the recipient's public key.
- It should be easy for the recipient to decrypt the cipher text using their private key.
- It should be computationally infeasible for an adversary to determine the private key from the public key.
- It should be computationally infeasible for an adversary to recover the original message from the public key and the cipher text.
- The two keys should be interchangeable.

These requirements are difficult to meet, and only a few algorithms have achieved widespread acceptance, including RSA, elliptic curve cryptography, Diffie-Hellman, and DSS.

- The requirements can be boiled down to the need for a trap-door one-way function.
- A one-way function is a function that is easy to compute in one direction but infeasible to compute in the other direction, where "infeasible" means that the effort to solve it grows faster than polynomial time as a function of input size.
- A trap-door one-way function is a family of invertible functions that have an additional piece of information (the trapdoor) that makes it easy to compute the inverse.
- The development of a practical public-key scheme depends on the discovery of a suitable trap-door one-way function.

Public-Key Cryptanalysis:
Public-key encryption is vulnerable to brute-force attacks, so large keys are necessary to make it impractical for attackers.

- However, the complexity of mathematical functions used in public-key encryption may not scale linearly with key size, which can result in slow encryption/decryption speeds.
- Therefore, public-key encryption is primarily used for key management and signature applications.
- There is a possibility that an attacker may find a way to compute the private key from the public key, although it has not been mathematically proven for any specific public-key algorithm, including RSA.
- A probable-message attack is a type of attack where an attacker can use the public key to encrypt all possible messages and compare them to the transmitted cipher text to find the original message.
- This attack can be thwarted by appending random bits to simple messages.

RSA algorithm:

RSA (Rivest-Shamir-Adleman) is a widely used public-key encryption algorithm that allows secure communication over an insecure channel. Here are the basic steps involved in the RSA algorithm:

- **Key Generation:**
  Choose two large prime numbers, p and q, and calculate their product, n = pq.
  Then, compute the totient of n, phi(n) = (p-1)(q-1).

Choose an integer e, such that 1 < e < phi(n), and e is relatively prime to phi(n).
Calculate the integer d, such that d is the multiplicative inverse of e modulo phi(n), i.e., de ≡ 1 (mod phi(n)).
The public key is (e, n) and the private key is (d, n).

- **Encryption:**
  To encrypt a message M, convert it to an integer m such that 0 <= m < n.
  Then, compute the cipher text C as C = m^e mod n.
  The sender transmits the cipher text C to the receiver.
- **Decryption:**
  To decrypt the cipher text C, the receiver uses the private key (d, n) and then computes m = C^d mod n.
  The decrypted message is then obtained by converting m back to its original form.

The security of RSA depends on the difficulty of factoring large integers i.e. the larger the prime numbers p and q used in key generation, the more secure the algorithm is.

Diffie-Hellman Key Exchange:

The Algorithm:

The Diffie-Hellman Key Exchange is a method for two parties to agree on a shared secret key over an insecure communication channel.

Global Public Elements	
q	prime number
α	$\alpha < q$ and α a primitive root of q

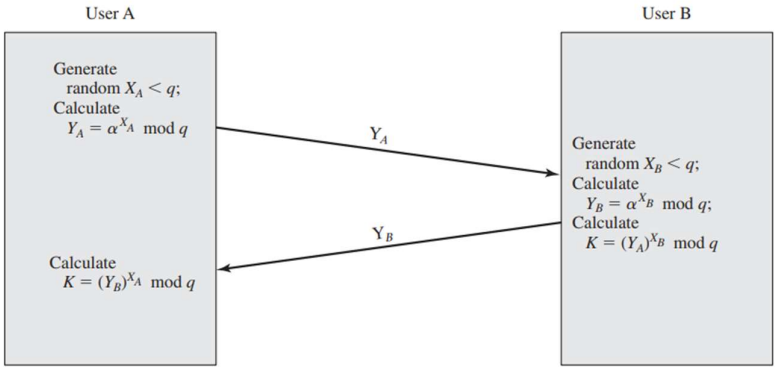
User A Key Generation	
Select private X_A	$X_A < q$
Calculate public Y_A	$Y_A = \alpha^{X_A} \bmod q$

User B Key Generation	
Select private X_B	$X_B < q$
Calculate public Y_B	$Y_B = \alpha^{X_B} \bmod q$

Calculation of Secret Key by User A
$K = (Y_B)^{X_A} \bmod q$

Calculation of Secret Key by User B
$K = (Y_A)^{X_B} \bmod q$

Key Exchange Protocols:



- User A generates a one-time private key, calculates a public value, and sends it to user B.
- User B responds by generating a private value, calculating a public value, and sending it to user A.
- Both users can now calculate the key.
- The public values need to be known ahead of time or included in the first message.
- The Diffie-Hellman algorithm can also be used by a group of users to establish secret keys with each other using long-lasting private values and a central directory.
- The technique provides confidentiality and a degree of authentication.
- Only the two users involved can determine the key, so no other user can read the message.
- The recipient knows that only the sender could have created a message using this key, providing authentication.
- However, the technique does not protect against replay attacks.

Man-in-the-Middle Attack:
The protocol above is insecure against a man-in-the-middle attack.

- An attacker, Darth, intercepts the communication between Alice and Bob.
- Darth generates two random private keys and calculates the corresponding public keys.
- Darth substitutes Alice's public key with his own and sends it to Bob, and substitutes Bob's public key with his own and sends it to Alice.
- Alice and Bob calculate their shared secret keys based on the public keys they receive, but instead they share a secret key with Darth.
- Darth can eavesdrop on all future communication between Alice and Bob and can modify the messages going to Bob.
- The vulnerability can be overcome with the use of digital signatures and public-key certificates.

Elliptic Curve Cryptography:

Global Public Elements	
$E_q(a, b)$	elliptic curve with parameters a, b, and q, where q is a prime or an integer of the form 2^m
G	point on elliptic curve whose order is large value n

User A Key Generation	
Select private n_A	$n_A < n$
Calculate public P_A	$P_A = n_A \times G$

User B Key Generation	
Select private n_B	$n_B < n$
Calculate public P_B	$P_B = n_B \times G$

Calculation of Secret Key by User A
$K = n_A \times P_B$

Calculation of Secret Key by User B
$K = n_B \times P_A$

- Elliptic Curve Cryptography (ECC) is a type of public-key cryptography that uses the arithmetic of elliptic curves to provide security.
- In ECC, addition is the equivalent of modular multiplication in RSA, and multiple addition is the equivalent of modular exponentiation.
- To form a secure cryptographic system using elliptic curves, a "hard problem" is needed that corresponds to the factoring of the product of two primes or taking the discrete logarithm.
- The discrete logarithm problem for elliptic curves involves finding the value of k in the equation kP = Q, where P and Q are known points on the curve and k is an unknown integer.
- It is relatively easy to calculate k given P and Q, but it is relatively hard to determine P or Q given k and the other point.

Encryption and Decryption:
An encryption/decryption system requires a point G and an elliptic group Eq(a,b) as parameters.

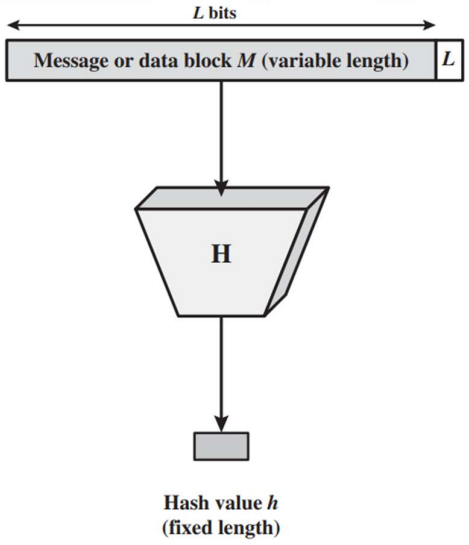
- Each user A selects a private key nA and generates a public key pA=nAxG.
- To encrypt and send a message to Pm to B, A chooses a random positive integer k and produces the cipher text consisting of the pair of points:

$$C_m = \{kG, P_m + kP_B\}$$

- A has used B's public key.
- To decrypt the cipher text, B multiplies the first point in the pair by B's secret key and subtracts the result from the second point:

$$P_m + kP_B - n_B(kG) = P_m + k(n_BG) - n_B(kG) = P_m$$

Cryptographic Hash Functions:



A hash function H accepts a variable-length block of data as input and produces a fixed-size hash value.

- A good hash function has the property that the results of applying the function to a large set of inputs will produce outputs that are evenly distributed and apparently random.
- The principal object of a hash function is data integrity i.e. a change to any bit or bits in results, with high probability, in a change to the hash code.

- A cryptographic hash function is an algorithm that is computationally infeasible to find either a data object that maps to a pre-specified hash result or two data objects that map to the same hash result.
- The input to a cryptographic hash function is typically padded out to an integer multiple of some fixed length, and the padding includes the value of the length of the original message in bits.

Applications:
A cryptographic hash function is a versatile tool used in many security applications, such as verifying message integrity.

- **Message Authentication:**
    1. When a hash function is used for message authentication, the hash value is called a message digest.
    2. Encrypting messages can be slow, expensive, and less efficient for small amounts of data.
    3. Instead, a message authentication code (MAC) is often used.
    4. A MAC function takes a secret key and data block as input and produces a hash value.
    5. If the integrity of the message needs to be checked, the MAC function can be applied to the message and compared to the stored MAC value.
    6. Combining hashing and encryption results in a MAC function that is generally more efficient than an encryption algorithm alone.
- **Digital Signatures:**
    1. Digital signature is similar to message authentication but uses encryption with a user's private key.
    2. Anyone with the user's public key can verify the integrity of the message associated with the digital signature.
    3. An attacker would need to know the user's private key to alter the message.
    4. Digital signatures have implications beyond message authentication.
    5. Digital signatures use a hash code that is encrypted using public-key encryption with the sender's private key to provide authentication and digital signature.
    6. If confidentiality is desired, the message plus the private-key-encrypted hash code can be encrypted using a symmetric secret key.

Requirements and Security:

A preimage is a data block that produces a specific hash value when passed through a hash function.

- A collision occurs when two different data blocks produce the same hash value.
- Collisions are undesirable because we use hash functions for data integrity.
- The number of preimages for a given hash value depends on the length of the hash code and the length of the input data blocks.
- On average, each hash value corresponds too many preimages.
- The security risks associated with hash functions are not as severe as they may seem, and their security requirements need to be precisely defined.

Requirement	Description
Variable input size	H can be applied to a block of data of any size.
Fixed output size	H produces a fixed-length output.
Efficiency	$H(x)$ is relatively easy to compute for any given x, making both hardware and software implementations practical.
Preimage resistant (one-way property)	For any given hash value h, it is computationally infeasible to find y such that $H(y) = h$.
Second preimage resistant (weak collision resistant)	For any given block x, it is computationally infeasible to find $y \neq x$ with $H(y) = H(x)$.
Collision resistant (strong collision resistant)	It is computationally infeasible to find any pair (x, y) such that $H(x) = H(y)$.
Pseudorandomness	Output of H meets standard tests for pseudorandomness.

Secure Hash Algorithm (SHA):

SHA is the most widely used hash function.

- SHA was developed by the National Institute of Standards and Technology (NIST).
- SHA-0 was found to have weaknesses.
- A revised version, SHA-1 produces a hash value of 160 bits.
- Later NIST produced a revised version of the standard that defined three new versions of SHA, known as SHA-256, SHA-384, and SHA-512.
- Later a 224-bit version of SHA-2 was added.
- SHA-2 uses the same underlying structure and operations as SHA-1 but with different hash value lengths.
- Later NIST announced the intention to phase out approval of SHA-1 and move to a reliance on SHA-2.
- Later a research team described an attack in which two separate messages could be found that deliver the same SHA-1 hash using fewer operations than previously thought.

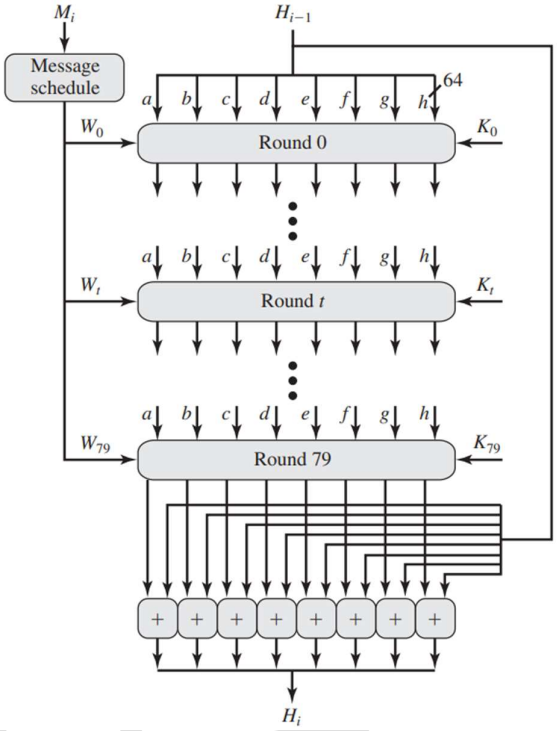
	SHA-1	SHA-224	SHA-256	SHA-384	SHA-512
Message Digest Size	160	224	256	384	512
Message Size	$< 2^{64}$	$< 2^{64}$	$< 2^{64}$	$< 2^{128}$	$< 2^{128}$
Block Size	512	512	512	1024	1024
Word Size	32	32	32	64	64
Number of Steps	80	64	64	80	80

SHA-512 Logic:

SHA-512 is a cryptographic hash function that takes an input message and produces a fixed-size output (512 bits). The process consists of several steps:

- **Padding:** The input message is padded with 1 followed by 0s so that its length is a multiple of 1024 bits.

- **Length:** A 128-bit block is appended to the padded message, which contains the length of the original message.
- **Initialization:** A 512-bit buffer is initialized with specific values.
- **Processing:** The padded message is processed in 1024-bit blocks using a module that consists of 80 rounds and each round updates the buffer using a message schedule, an additive constant, and the current 1024-bit block.
- **Output:** After processing all blocks, the output is a 512-bit message digest.



SHA-512 Round Function:

The SHA-512 Round Function is the core operation in the SHA-512 hash algorithm.

- The round function operates on a 512-bit buffer consisting of eight 64-bit registers labelled a, b, c, d, e, f, g, and h.
- Each round uses a 64-bit value derived from the current 1024-bit block being processed, and an additive constant derived from the first 80 prime numbers.
- The round function is designed to produce a highly randomized output, which makes it resistant to various cryptographic attacks.