SNS COLLEGE OF TECHNOLOGY

(Autonomous)
COIMBATORE – 35

DEPARTMENT OF COMPUTER SIENCE AND ENGINEERING (UG & PG)

Third Year Computer Science and Engineering, 5th Semester

UNIT I – CYBER SCURITY FUNDAMENTALS

Topic Name : Cryptography

Basic Cryptography

The English word cryptography derives from Greek and translates roughly to "hidden writing." The ancient Egyptians began the first known practice of writ-ing secret messages, using nonstandard hieroglyphs to convey secret messages as early as 1900 bc. Since that time, people have developed many methods of hiding the content of a message. These methods are known as ciphers. The most famous classical cipher is the substitution cipher. Substitution ciphers work by substituting each letter in the alphabet with another one when writing a message. For instance, one could shift the letters of the English alphabet as shown:

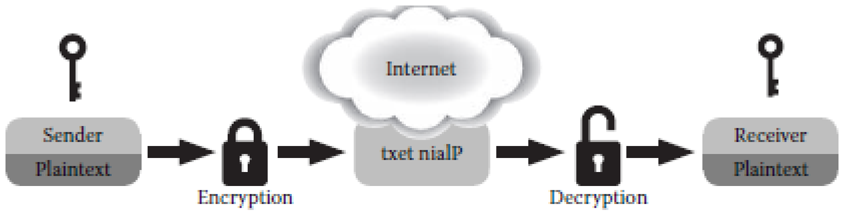
abcdefghijklmnopqrstuvwxyz
nopqrstuvwxyzabcdefghijklm

Using this cipher, the message "the act starts at midnight" would be written as "gur npg fgnegf ng zvqavtug." The text above, showing how to decode the message, is known as the key. This is a very simple substitution cipher known as the Caesar cipher.

Cryptography is driven by the constant struggle between people who want to keep messages secret and those who work to uncover their meanings. Substitution ciphers are very vulnerable to cryptanalysis, the practice of breaking codes. With enough text, it would be simple to begin replacing characters in the ciphertext with their possible cleartext counterparts.

Symmetric Encryption

Symmetric encryption is a class of reversible encryption algorithms that use the same key for both encrypting and decrypting messages. Symmetric encryption, by definition, requires both communication endpoints to know the same key in order to send and receive encrypted messages (below mentioned diagram). Symmetric encryption depends upon the secrecy of a key. Key exchanges or pre-shared keys present a challenge to keeping the encrypted text's confidentiality and are usually performed out of band using different protocols.



Symmetric encryption: the sender and receiver use the same key.

Example of Simple Symmetric Encryption with Exclusive OR (XOR)

At its most basic level, symmetric encryption is similar to an exclusive OR (XOR) operation, which has the following truth table for input variables p and q,

P	Q	P XOR Q
TRUE	TRUE	FALSE
TRUE	FALSE	TRUE
FALSE	TRUE	TRUE
FALSE	FALSE	FALSE

The XOR operation is nearly the same as one would expect for OR, except when both p and q are true. The properties of XOR make it ideal for use in symmetric cryptography because one of the inputs (p) can act as the message and the other input (q) can act as the key. The recipient of an encrypted message (p XOR q) decrypts that message by performing the same XOR operation that the sender used to encrypt the original message (p).

P XOR Q	Q	P XOR Q XOR Q
FALSE	TRUE	TRUE
TRUE	FALSE	TRUE
TRUE	TRUE	FALSE
FALSE	FALSE	FALSE

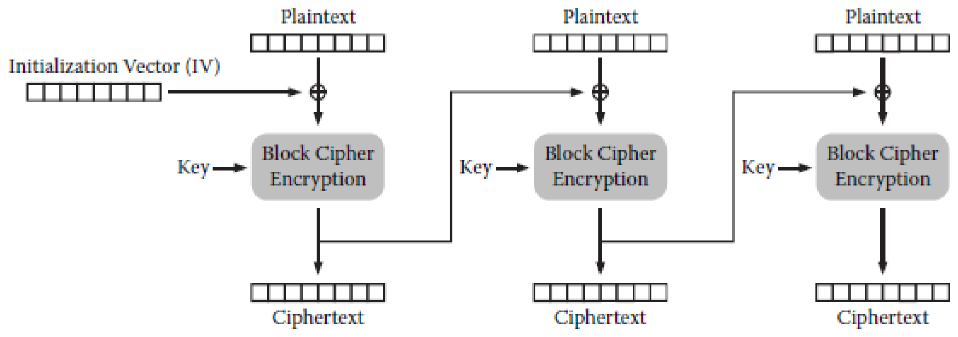
The operation above shows how to decrypt the encrypted message (p XOR q) to obtain the original message (p). Applying this technique to larger values by using their individual bits and agreeing on a common key (q) represents the most basic symmetric encryption algorithm. Encryption using XOR is surprisingly common in unsophisticated malicious code, including shellcode, even as a means to hide logging or configuration information. Due to its simplicity, many unsophisticated attackers use either one-byte XOR keys or multibyte XOR keys to hide data.

The XOR operation is an example of a stream cipher, which means that the key operates on every bit or byte to encrypt a message. Like traditional substitution ciphers, XOR leaves patterns in ciphertext that a cryptanalyst could use to discover the plaintext. Performing an XOR operation on the same data twice with the same key will always result in the same ciphertext.

Modern stream ciphers like RC4, designed by Ron Rivest in 1987, avoid this problem by using a pseudo-random number generation (PRNG) algorithm. Instead of performing an XOR on each byte of data with a key, a PRNG receives a chosen key, used as a "seed." A PRNG generates numbers that are close to random but will always be the same given the same seed. RC4 uses the PRNG to create an infinitely long, one-time pad of single-byte XOR keys. This technique allows the sender to encrypt a message with a single (relatively short) key, but for each individual byte, the XOR key is different.

Improving upon Stream Ciphers with Block Cipher

Block ciphers are more common in symmetric encryption algorithms because they operate on a block of data rather than each character (bit or byte). PRNG algorithms used in stream ciphers are typically time intensive. Block ciphers are the best choice for bulk data encryption. Stream ciphers remove patterns from ciphertext using PRNGs, but block ciphers use a more efficient method called cipher block chaining (CBC). When using a block cipher in CBC mode, both a key and a random initialization vector (IV) convert blocks of plaintext into ciphertext. The initialization vector and plaintext go through an XOR operation, and the result is an input to the block cipher with the chosen key (below mentioned diagram). This ensures that the resulting ciphertext is different, even if the same key was used to encrypt the same plaintext, as long as the IV is different and sufficiently random with each execution of the algorithm.



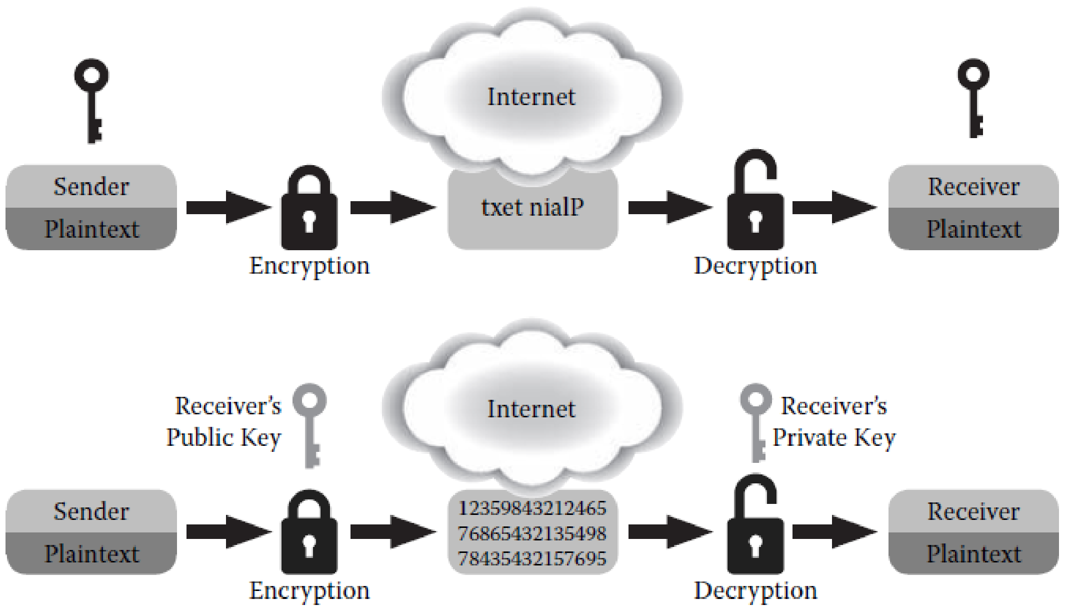
Cipher block chaining (CBC) mode encryption

Symmetric encryption can be very fast and protect sensitive information provided the key remains secret. The grouping of larger blocks of data in the encryption algorithm makes it more difficult to decrypt without the key. Key exchange and protection are the most important aspects of symmetric cryptography because anyone who has the key can both encrypt and decrypt messages. Asymmetric algorithms are different because they use different keys for encryption and decryption, and in this way, public key encryption can solve other goals beyond symmetric algorithms that protect confidentiality.

Public Key Encryption

Public key encryption represents a branch of cryptography for which the distinguishing attribute of the system is the use of two linked keys for encryption and decryption, rather than a single key. While a variety of public key encryption solutions have been proposed, with some implemented and standardized, each system shares one common attribute: each public key system uses one key, known as the public key, to encrypt data, and a second key, known as the private key, to decrypt the encrypted data.

Public key encryption solves one of the major issues with symmetric key encryption, namely, the use of a shared key for both sides of the conversation. In public key systems, the intended recipient of a secure communication publishes his or her public key. Anyone wishing to send a secure datagram to the recipient uses the recipient's public key to encrypt the communication; however, those in possession of the public key cannot use the key to decrypt the communication. The use of a public key is a one-way cryptographic operation. This allows recipients to give out their public keys without the risk of someone using the same public keys to reveal the original content of the messages sent. This is the most obvious advantage over symmetric encryption. To decrypt the encrypted message, the recipient uses his or her private key.



Symmetric encryption (top) versus public key encryption (bottom).

The private key has a mathematical relationship to the public key, but this relationship does not provide an easy way for an attacker to derive the private key from the public key. Given the fact that the recipient uses the private key to decrypt messages encoded with the public key, it is paramount that the owner of the private key keeps it secure at all times.

Visually, the process of encrypting and decrypting a message using the public key method is similar to the process of using symmetric encryption with the notable exception that the keys used in the process are not the same.

Researchers (Ron Rivest, Adi Shamir, and Leonard Adleman) at MIT expanded on this research to develop one of the widest used public key encryption systems in use today. Known as the RSA system, a name derived from the original inventors' last names, the system uses large prime numbers to encrypt and decrypt communication.

RSA process works as such:

1. The recipient generates three numbers: one to be used as an exponential (e), one as a modulus (n), and one as the multiplicative inverse of the exponential with respect to the modulus (d). The modulus n should be the product of two very large prime numbers, p and q. Thusly, n = pq.

2. The recipient publishes his or her public key as (e, n).

3. The sender transforms the message (M) to be encrypted into an integer whose value is between 0 and (n−1). If the message cannot fit within the confines of this integer space, the mes-sage is broken into multiple blocks.

4. The sender generates the ciphertext (C) by applying the fol-lowing mathematical function:

$$C = M^e \bmod n$$

5. The sender transmits the ciphertext to the recipient.

6. The recipient uses the pair (d, n) as the private key in order to decrypt the ciphertext. The decryption process uses the following mathematical transform to recover the original plaintext:

$$M = C^d \bmod n$$

The power of the RSA scheme lays in the use of the large prime numbers p and q. Factoring an extremely large prime number (on the order of 2^{1024} or 309 digits) is an exceedingly difficult task—a task for which there is no easy solution. To understand how the RSA scheme works in simpler terms, it is best to use a simpler, smaller example,

1. The recipient chooses two prime numbers: for example, p = 17 and q = 11.

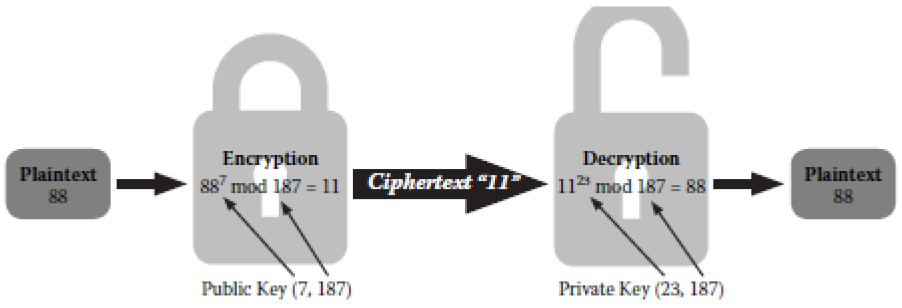
2. The recipient calculates n by multiplying the two prime numbers together: (n = 187).

3. The recipient chooses an exponent such that the exponent is less than (p−1)(q−1), which is 160, and the exponent is relatively prime to this number. In this scenario, a recipient could choose the number 7, as it is less than 160 and relatively prime to 160.

4. The value of d is calculated by solving de = 1 (mod 160) with d < 160. The math behind this calculation is beyond the scope of this book; however, in this scenario, d has the value of 23.

5. At this point in the scenario, the recipient could have developed a private key of (23, 187) and a public key of (7, 187).

If the sender were to encrypt the message of 88 (which is between 0 and 186) using the RSA method, the sender would calculate 88^7 mod 187, which equals 11. Therefore, the sender would transmit the number 11 as the ciphertext to the recipient. To recover the original message, the recipient would then need to transform 11 into the origi-nal value by calculating 11^{23} mod 187, which equals 88.



An RSA encryption–decryption example.