Graphical user interface, application

Description automatically generatedGraphical user interface, application, Word

Description automatically generated

* 1. Equipment theft threatens the confidentiality, availability, and integrity of computer systems and data.
  2. Unauthorised copying of software threatens the copyright of the software and its associated intellectual property, as well as the availability of the software.
  3. Modification of existing files threatens the integrity of the files and the data stored therein.
  4. Destroying messages threatens the confidentiality, availability, and integrity of the messages and any data contained therein.
  5. Observing traffic patterns of messages threatens the confidentiality and privacy of the messages, as well as the availability of the data contained therein.

1. Source 2 is more random on average. This is because the sum of the squared probability values for Source 2 (0.09 + 0.01 + 0.25 + 0.01 = 0.36) is greater than the sum of the squared probability values for Source 1 (0.09 + 0.04 + 0.16 + 0.01 = 0.30). This indicates that Source 2 has a more unpredictable distribution, making it more random on average.
2. A binary source with two events only has a maximum entropy of 1 bit. When plotted on a graph of entropy against binary probability distribution, the graph will be a straight line with a slope of -1, with the entropy decreasing as the probability of the event increases.

Graphical user interface, application, Word

Description automatically generatedTable

Description automatically generated with low confidence

Pros:

-The cryptosystem does not require a key exchange and is based on a one-time keypad, making it secure.

-It is computationally efficient and does not require a large amount of resources.

Cons:

-The system is vulnerable to a man-in-the-middle attack.

-Alice and Bob must be online at the same time in order to exchange messages.

John's cryptosystem does work, as it provides a secure way for Alice and Bob to communicate. By using the one-time keypad and the XOR operation, Alice and Bob are able to send and receive messages in a secure manner. The one-time keypad ensures that the same key cannot be used more than once, which makes it difficult for an attacker to break the encryption. Additionally, since no key exchange is required, Alice and Bob can communicate without having to worry about sharing a key. John's cryptosystem is secure, but it is vulnerable to man-in-the-middle attacks and requires Alice and Bob to be online at the same time, making it less than ideal for secure communication.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Alice |  | Charlie |  | Bob |
| m |  |  |  |  |
| ↓ |  |  |  |  |
| Encrypt(m) | **→** | **Encrypt(Encrypt(m))** | **→** | **Decrypt(Encrypt(Encrypt(m)))** |
|  |  |  |  | **↓** |
| Decrypt(Decrypt(Encrypt(Encrypt(m)))) | **←** | **Encrypt(m)** | **←** | **Decrypt(Encrypt(m))** |
| ↓ |  | **↓** |  |  |
| Encrypt(m) |  | **Encrypt(m)** |  |  |
| ↓ |  |  |  |  |
| Encrypt(Encrypt(m)) | **→** | **Encrypt(Encrypt(Encrypt(m)))** | **→** | **Decrypt(Encrypt(Encrypt(Encrypt(m))))** |
|  |  | **↓** |  | **↓** |
|  |  | **Encrypt(m)** |  | **Encrypt(m)** |

Graphical user interface, application, Word

Description automatically generated

1. To generate an identical new random password of up to 15 bits without sending the new password, Alice and Bob can use a cryptographic hash function. The hash function takes the password seed as an input and produces a unique output (the new random password). For example, Alice and Bob could use the SHA-2 hash algorithm, which produces an output of 256 bits (32 bytes). Alice and Bob can then select the leftmost 15 bits of the output as their new password.
2. The main risk of using a cryptographic hash function is that, if the same input is used more than once, the same output will be produced each time. This means that Alice and Bob’s new passwords may become predictable after a certain number of iterations. Additionally, if the password seed is ever compromised, then anyone with knowledge of the seed could easily generate the same new password, as well as any future passwords Alice and Bob generate.

Graphical user interface, text, application

Description automatically generated

1. The Vernam cipher works by XORing each character of the plaintext with the key characters. In the example of "computer" and the one-time pad (5 20 0 9 17 16 22 18), the ciphertext is 7 4 20 14 1 8 4 5. This is found by XORing each character of the plaintext with the corresponding character of the key:

c (99) XOR 5 (0101) = 94 (0101 1110) = 7 (0111)

o (111) XOR 20 (10100) = 127 (0111 1111) = 4 (0100)

m (109) XOR 0 (0000) = 109 (0110 1101) = 20 (10100)

p (112) XOR 9 (1001) = 103 (0110 0111) = 14 (1110)

u (117) XOR 17 (10001) = 100 (0110 0100) = 1 (0001)

t (116) XOR 16 (10000) = 132 (1000 0100) = 8 (1000)

e (101) XOR 22 (10110) = 123 (0111 1011) = 4 (0100)

r (114) XOR 18 (10010) = 102 (0110 0110) = 5 (0101)

The cipher is hopeless in practice because it relies on a one-time pad, which is a randomly generated key of the same length as the plaintext. This key must be kept secret and never reused, which is impossible in most practical applications. Additionally, the key must be perfectly synchronized between the sender and receiver, making it difficult to use in practice.

Graphical user interface, text, application

Description automatically generated

1. The transposition cipher works by rearranging the order of the letters in the plaintext.

In this example, the plaintext can be decrypted by first dividing the ciphertext into blocks of 7 characters (as the key is 7 characters long).

HKFPRZN

IWUVLGU

OJOEO T

CNMEAOE

BOETYCQ

RXDHDE

The next step is to replace the letters in each block with the letters in the key.

IAMTHE

IAMTHE

IAMTHE

IAMTHE

IAMTHE

IAMTHE

The final step is to rearrange the letters in each block to match the order of the letters in the key.

HIAMTTE

EAMTHHU

EOMAITI

TEAMHEO

ECOETYM

HRDEDXN

The plaintext is HIAMTTEEAMTHHUEOMAITITEAMHEOECOETYMHRDEDXN.

Graphical user interface, text, application

Description automatically generated

1. n = pq = 5\*7 = 35
2. r = φ(n) = (p-1)(q-1) = (5-1)(7-1) = 24
3. The decryption exponent d is found by using the equation: d = e^-1 mod r. e^-1 mod r is the modular inverse of e mod r. To find this we use the Extended Euclidean Algorithm:

7\*d = 1 mod 24

7\*d = 24k + 1

From the algorithm,

d = 17

1. Bob's private key consists of p, q, e and d. Private Key = {5, 7, 7, 17}
2. Bob's public key consists of n and e. Public Key = {35, 7}

Graphical user interface, application

Description automatically generated

1. To show that g = 3 is a generator for p = 17, we must show that every element in the group Z\*17 is generated by 3.

We can do this by showing that every element in Z\*17 can be written as 3n for some n ∈ [1, p − 1].

For example, 3^1 mod 17 = 3, 3^2 mod 17 = 9, 3^3 mod 17 = 5, 3^4 mod 17 = 15, 3^5 mod 17 = 1, 3^6 mod 17 = 11, 3^7 mod 17 = 13, 3^8 mod 17 = 7, 3^9 mod 17 = 16, 3^10 mod 17 = 4, 3^11 mod 17 = 14, 3^12 mod 17 = 2, 3^13 mod 17 = 12, 3^14 mod 17 = 8, 3^15 mod 17 = 6, and 3^16 mod 17 = 10.

Therefore, 3 is a generator for p = 17.

1. To show that g = 2 is not a generator for p = 17, we must show that not every element in the group Z\*17 can be written as 2n for some n ∈ [1, p − 1].

For example, 2^1 mod 17 = 2, 2^2 mod 17 = 4, 2^3 mod 17 = 8, 2^4 mod 17 = 16, 2^5 mod 17 = 13, 2^6 mod 17 = 7, 2^7 mod 17 = 14, 2^8 mod 17 = 11, 2^9 mod 17 = 1, 2^10 mod 17 = 5, 2^11 mod 17 = 10, 2^12 mod 17 = 3, 2^13 mod 17 = 6, 2^14 mod 17 = 12, 2^15 mod 17 = 9, and 2^16 mod 17 = 15. Notice that 2^16 mod 17 = 15 ≠ 2.

Therefore, 2 is not a generator for p = 17.