**Data Security**

Table of Contents

[Terminology 1](#_Toc63380098)

[Single-Key Encryption 2](#_Toc63380099)

[Symmetric/Private Key Encryption 3](#_Toc63380100)

[Asymmetric/Public Key Encryption 4](#_Toc63380101)

[Choosing Numbers 5](#_Toc63380102)

The main concept of data security is that data being sent is encoded on the sender’s end and decoded on the receiver’s end. In this way, even if a third party were to obtain the message, they would be unable to understand its content without knowing how to decode it.

## Terminology

* **Cleartext or Plaintext** – This refers to the original, non-encoded data.
* **Encryption** – This is the process of transforming data so that it is no longer understandable.
* **Decryption** – This is the reverse process of encryption.
* **Ciphertext** – This is the encrypted data.
* **One-Way Cipher** – This refers to an encryption that cannot easily be decrypted. One-way ciphers are used for passwords, where the user will enter a password, the password will be encrypted using strong encryption algorithms, and the encrypted form will be stored in the database for authentication purposes. In this way, not even administrators that have access to the database and can see the encrypted password that is stored will be able to get into the user’s account.
* **Cryptosystem** – This is the complete system of encryption and decryption methods being used.
* **Cryptography** – The study of encryption methods. We are currently at a point where 256-bit encryption is used, which would take thousands of years to break, even using super-computers. As such, research into encryption methods is becoming a little pointless.
* **Cryptoanalysis** – The study of decryption methods.

## Single-Key Encryption

Conventional encryption algorithms are single-key encryption algorithms. There is a fixed key, , based on which the data is transformed.

One of the first single-key encryption algorithms is Caesar’s cipher. The algorithm replaces every letter in a message with the letter that is 3 steps away in the alphabet. Thus, A is replaced by D, B is replaced by E and so on. Here, the value of is .

Caesar’s cipher is a specific form of a more general group of encryption algorithms called substitution algorithms. In substitution algorithms, every letter of a message is substituted by another letter, based on the value of the key. The receiver must know the key with which the sender encrypted the text to be able to decrypt it.

The shortcoming of substitution algorithms is that they are very naïve, so it is also very easy to break them. Anyone can easily realize that a simple shift has been done by analysing the encrypted text for a while. Common words that appear frequently in the text will also have their encrypted versions appear frequently in the encrypted text, which will also make the key easier to find.

There are some other substitution algorithms that assign random substitutions, perhaps based on the position of the letter within the text. These ciphers are a little harder to crack, but overall, it is still doable, even by a human.

Nowadays, such simple encryption algorithms are no longer used. Most encryption algorithms used now use mathematical formulae and a much stronger encryption key to encrypt data. The length of the key determines how secure the encryption is, because the longer the key, the longer it will take to find the key using a brute force approach. Key lengths are generally between 40 to 128 bits, with military grade encryption being a minimum of 256 bits. Even though these numbers are mind-bogglingly large and completely impossible to find through a brute force approach, they are so common that most web browsers support them.

## Symmetric/Private Key Encryption

In Private Key Encryption, the sender and receiver both have a single key. This key is used to encrypt and decrypt the data. The most widely used standard for Private Key Encryption is called the Data Encryption Standard (DES).

However, this method has a problem. Every pair of sender and receiver must have a unique key. This means for entities to be able to communicate with each other, each entity would have to have keys, meaning there would have to be keys in total. Due to the huge number of keys involved, this type of encryption is generally used by government entities. E-commerce transactions over the internet rarely use this method.

Another complicacy is that there needs to be a secure way to communicate the key in the first place so that both parties know the key.

## Asymmetric/Public Key Encryption

In Public Key Encryption, two numeric keys are used. Every entity has a public key, which is known to everyone, and a private key, which is known to only themselves. The public key of the receiver is used by the sender to encrypt the message. The receiver uses their private key to decrypt the message. In this way, even though the public key of the receiver is publicly visible, no one is able to decrypt the message without knowing their private key.

The advantage of using Public Key Encryption as opposed to Private Key Encryption is that a single entity only needs to know two keys, their own public and private keys, instead of having to keep track of different keys. In total, there would just be keys.

For example, say the public key is and the private key is . If the message is the number , the sender will make the following calculation:

Thus, the number is sent. The mod has been done since, generally, the keys are so large that the actual results cannot be sent.

On the receiver’s end, it is decrypted as:

Therefore, the formulae are:

where is the original message, is the encrypted message, is the public key, is the private key and is the multiplication of two very large prime numbers. Private and public keys must be chosen in such a way so that these equations are valid.

### Choosing Numbers

The process of creating keys we are about to see comes from the RSA algorithm, named after its creators.

We need to choose two very large prime numbers, and . Next, we make the following calculations:

is called the RSA modulos and is called the Totient function of , i.e. .

For the calculated value of , we need to find a number , that is relatively prime to it. This means that is not a prime factor of or a multiple of it. Thus, is the co-prime factor of .

A co-prime factor is a number for which the only common divisor is . For , a few possible co-prime factors are , , , etc. Keep in mind that the chosen number must be less than .

The chosen value of is the public key.

Next, we need to calculate the private key, .

Say was chosen to be . One way of finding now is to use the Euclidean algorithm. In the Euclidean algorithm, we try to find the greatest common divisor.

We got as the remainder. Next, we repeat this step for and .

Once we get a as the remainder, we need to work backwards, expressing the in terms of the previous equations.

Thus, the value of is , our private key.

Sometimes, it is possible that the value of we get is negative. Say we have a situation where and . In that case, we need to find a multiple of that is less than (more negative) than . In this case, . Thus,

Say we want to send the character ‘B’. We could give the character the value . Thus, when we want to send the message , we create the encrypted message as

On the receiver’s end, we decrypt the message as

Example

To use the RSA algorithm, we first need to choose two large prime numbers. It is difficult to perform calculations on large numbers by hand, so for now, we will choose two small prime numbers.

Possible choices for values that are relatively prime to and less than are , , , , etc. Say .

Using the Euclidean algorithm,

Using the extended Euclidean algorithm,

Essentially, we expressed the value in terms of and . Thus, the value of should be

The primary benefit of public key encryption is that people with no pre-existing security arrangement are able to exchange messages securely. They do not need to ensure a secure channel via which a secret key can be exchanged, which requires special services like VPNs and HTTPS certificates and are an additional cost. Additionally, since all communication only involves public keys, the private keys are never shared, which increases security.

Some algorithms that use public key encryption are:

* Elgamal
* RSA
* Diffie-Hellman
* Digital Signature Algorithm (DSA), which allows users to attach a digital signature and thus verify their identity
* Pretty Good Privacy (PGP), which is a hybrid that uses both public key encryption and private key encryption

## Digital Certificates

It is possible to use public key encryption to create digital signatures. These signatures are used on the Internet to authenticate users and vendors. They are sent as encrypted attachments along with electronic messages to verify the sender’s identity.

The interesting part about digital signatures is that the roles of the public and private keys are reversed. The sender encrypts the data using their private key and anyone receiving data from them decrypts it using their public key. In this way, if it is not possible to decrypt a certain message using the public key for the expected sender, we will know that the message did not actually come from them. Since no one has their private key, no one can replicate their encryption.

Digital signatures are actually sent as a digital certificate. These digital certificates are unique identifiers assigned to users and vendors by a certification authority. Anyone wanting a digital certificate must contact the certification authority and verify their identity before they can be assigned a digital certificate. A certification authority is a private company that works together with credit card verification companies and other financial institutions to verify the identity of the person or entity requesting certification. One of the most famous certification authorities is VeriSign.

## Pretty Good Privacy

PGP combines the best features of both private key encryption and public key encryption to create a hybrid crypto system.

The first step is to compress the plaintext. Data compression saves transmission time and disk space, and more importantly, strengthens cryptographic security. Most cryptoanalysis techniques exploit patterns found in ciphertext to crack them. Compressing the plaintext before encrypting it ensures that such patterns no longer exist, which enhances resistance to cryptoanalysis.

The second step is to encrypt the compressed plaintext using a key, as in private key encryption. In PGP, this key is called a session key. This is a one-time key that is generated from the random mouse movements and key strokes of the user. Thus, it is quite secure and not something that can be guessed or found via a brute force attack.

Finally, the encrypted ciphertext is sent to the receiver. However, the receiver does not know the random one-time session key required to decrypt the text, so this key needs to be sent as well. The key is sent using public-key encryption. The receiver’s public key is used to encrypt the session key and then send it.

On the receiver’s end, they must use their private key to first decrypt the session key, and then use the session key to decrypt the ciphertext. Finally, they can decompress the text to get the original plaintext.

The advantages of doing all this are two-fold. Private key encryption is 1000 times faster than public key encryption, so encrypting the entire plaintext using this means the encryption process itself is faster. Additionally, using public key encryption ensures that key distribution and data transmission issues faced by traditional private key encryption algorithms can be avoided.