SENSITIVE DATA EXPOSURE TYPES

## Data in transit is data that we send from one system to another. This includes personal projects, work, confidential data, or business. It is very important that when our data is in transit from one system to another, it is 100% protected.

## Here below I will leave tips and examples of how to protect the data that is in transit.

## Implementation of secure key and certificate management

## In a world where there are so many people and automated machines that know how to steal confidential data, it is important to be alert, since an encryption that used to protect your data is now no longer useful because it has been violated by an algorithm.

## Simple substitution encryption should not be used because this encryption replaces each letter in the message with another letter of the alphabet. For example, D would be replaced by A, F, or G, and this encryption technique using simple substitution cipher is an easy cipher to break using frequency analysis techniques, which what this frequency does is identify the most common letter in the encrypted message and what this technique does is assume that it corresponds to the most common letter of the alphabet. Substitution Ciphers - Detailed Guide 2022 - Privacy Canada

## For example, if you use simple substitution encryption for facial data transit, your data will be exposed or stolen and if you are a company you can get a big fine due to the GDPR law and if you are a person you can end up in jail or with a fine for using an obsolete encryption, and the recommendation that I am going to give you is that you use encryption such as AES (Advanced Encryption Standard), this encryption uses a key to encrypt and decrypt the data and uses a block structure to encrypt the data in 128-bit blocks. AES is highly secure and resistant to known cryptographic attacks, making it widely used in applications such as file encryption, disk encryption, and network encryption. What is Advanced Encryption Standard (AES)❓

## In addition, to protect the privacy and integrity of data in transit through communication networks, it is recommended to use the TLS (Transport Layer Security) protocol that uses AES encryption and other encryption algorithms to protect data. TLS is used by many web applications, such as e-mail and e-commerce, and is essential for ensuring the security of online communications.

## Ciphers that I would not recommend using under any conditions.

| **Cipher Name** | **Reasons not to recommend** |
| --- | --- |
| Simple substitution cipher | It is very easy to break using frequency analysis techniques. |
| Vigenère cipher | It is also vulnerable to frequency analysis techniques, plus its key can be easily discovered using cryptanalytic methods. |
| DES encryption | Its 56-bit key length is very short compared to current encryption standards, and it is vulnerable to brute force attacks and other cryptanalysis methods. |
| RSA encryption with keys less than 1024 bits | Shorter keys are easier to break using prime factorization techniques. |

## Ciphers that I would recommend using so that the data is not stolen while transiting from one system to another:

| Cipher Name | Reasons to recommend it |
| --- | --- |
| AES (Advanced Encryption Standard) | It uses a key to encrypt and decrypt data and is highly secure and resistant to known cryptographic attacks. It is widely used in applications such as file encryption, disk encryption, and network encryption. |
| RSA with keys of at least 2048 bits | It is a widely used and secure asymmetric encryption algorithm, but it is recommended to use keys of at least 2048 bits for better security. |
| SHA-256 (Secure Hash Algorithm 256) | It is a secure hash algorithm used to verify data integrity. It is highly resistant to data tampering attempts and is widely used in computer security applications. |
| blowfish | It is a very secure and fast symmetric block cipher that is widely used in computer security applications such as file encryption and password encryption. |

## Understanding Personal and Sensitive Data

Have you ever wondered what personal data and sensitive data are? Personal data is information that can identify an individual, such as their name, ID number, age, and address. On the other hand, sensitive data is personal information that is more private and intimate.

This information should be kept confidential and only accessible to those who have permission to do so. Unlike personal data, the mismanagement of sensitive data can result in discrimination or direct harm to a person's reputation and well-being.

## How Cyberattacks Can Expose Sensitive Data

Cyberattacks can expose sensitive data, and there are various types of cyberattacks. Phishing is one of the most common types of attacks, which involves tricking the recipient into providing personal information through fake emails or websites.

Malware, such as viruses and trojans, is malicious software that can be installed on your computer without your knowledge. Social engineering involves psychological manipulation to get confidential information from you, and brute force attacks try every possible password combination until the right one is found. Software vulnerabilities in programs can also be exploited by attackers to gain access to private data.



## Protecting Yourself from Data Exposure

As technology advances, our personal data and systems are increasingly at risk of being compromised by malicious computer threats. However, there are steps we can take to safeguard ourselves from these attacks.

Firstly, it's important to exercise caution when receiving and opening emails. Always verify the sender's identity and be wary of clicking on any links or giving out sensitive information. Using reliable antivirus software can also help to detect and block dangerous emails.

Keeping your operating system and applications up to date with the latest security updates is also essential in preventing malware from infiltrating your device. It's best to avoid downloading or opening files from unknown or suspicious sources to minimize the risk of exposure.

Social engineering tactics are also commonly used by cybercriminals to trick people into divulging confidential information. Be wary of unexpected requests for sensitive information and never give out personal details over the phone or by email. Additionally, using strong and frequently changed passwords, as well as enabling multi-factor authentication, can help to prevent brute force attacks.

Lastly, it's crucial to keep all software up to date and apply necessary security patches to minimise vulnerabilities. Always download software from reputable sources and avoid pirated or untrusted websites.

Overall, protecting we from computer threats is an ongoing effort that requires a combination of technological solutions, user education, and awareness. By taking these steps, we can minimise our vulnerability to data exposure and keep our systems and personal information safe.

Figure 1:

#Here what we are doing is generate a random encryption key of 256-bit

let key = window.crypto.getRandomValues(new Uint8Array(32));

What we are doing in this line of code is generating a random encryption key of 256 bits. We achieve this by using the getRandomValues function of the window.crypto API and creating a Uint8Array object called key with a length of 32 bytes to store the generated key.

Figure 2:

# Here what I’m doing is to convert the file to a Uint8Array

let file = new Uint8Array([student-confidencial.txt]);

What we are doing in this line of code is creating a Uint8Array object called file that contains the data of the file we want to encrypt. We accomplish this by providing the data as a byte array, which is represented by the array [student-confidential.txt].

Figure 3:

# Here what we are doing is to create an initialization vector (IV)

let iv = window.crypto.getRandomValues(new Uint8Array(16));

What this line of code does is generate an initialization vector (IV) - a random vector of 16 bytes. We create it using the getRandomValues() function of the window.crypto API. The IV is used to randomize the encryption process, making it much more secure.

Figure 4:

#Here what we are doing is create the cipher object.

let cipher = await window.crypto.subtle.encrypt({

name: 'AES-CBC',

iv: iv

}, key, file);

What we are doing here is creating a cipher object using the encrypt() function of the window.crypto API. This function encrypts the data in the file we created in Figure 2 using the values of the key and IV, and the AES-CBC encryption algorithm. The 'await' keyword is used to wait for the encryption process to complete before moving on to the next line of code. Once the encryption is complete, the encrypted data is stored in the cipher object.

Figure 5:

# Here what we are doing is to convert the cipher object to an ArrayBuffer

let encryptedFile = new Uint8Array(await new Response(cipher).arrayBuffer());

What this line of code does is convert the cipher object into an ArrayBuffer object called encryptedFile. We accomplish this by creating a new Response object with the cipher object as input, and then using the arrayBuffer() method on the Response object.

Figure 6:

# Here what we do is store the key, IV, and encrypted file in an object

let encryptedData = {

key: key,

iv: iv,

data: encryptedFile

};

In this line of code, we create an object called encryptedData. This object contains the values of the key, IV, and encryptedFile, which are stored as properties of the data object.

Figure 7:

# Here what we are doing is to convert the encryptedData object to a JSON string

let encryptedJSON = JSON.stringify(encryptedData);

Finally, what we do here is convert the encryptedData object to a JSON string using the stringify() method on the JSON object. This allows us to transmit or save the encrypted data as a string.