



TECNOLÓGICO NACIONAL DE MÉXICO
INSTITUTO TECNOLÓGICO DE TLAXIACO

INGENIERÍA EN SISTEMAS COMPUTACIONALES.

SEGURIDAD Y VIRTUALIZACIÓN.

REPORTE DE PROYECTO FINAL

INTEGRANTES DEL EQUIPO:

JEANETTE ARLET SALAZAR NICOLÁS	21620202
NELSY ORTIZ LÓPEZ	21620165
MARIBEL LUCERO ZUÑIGA	21620139

SEMESTRE: SÉPTIMO.

GRUPO: 7 US

ASESOR: ING. EDWARD OSORIO SALINAS.

TLAXIACO, OAXACA A 06 DE DICIEMBRE DEL 2024.

CONTENIDO

1. INTRODUCCIÓN.....	3
2. MATERIALES	3
3. OBJETIVO GENERAL	3
3.1 OBJETIVOS ESPECIFICOS	3
4. MARCO TEÓRICO	4
4.1 ¿QUÉ ES NETBEANS?	4
4.2 ¿QUÉ ES EL CIFRADO?	4
4.3 CIFRADO FRENTE A CRIPTOGRAFÍA.....	4
4.4 ¿CÓMO FUNCIONA EL CIFRADO?	5
4.5 TIPO DE CIFRADO.....	5
4.5.1 ADVANCED ENCRYPTION STANDARD (AES)	5
4.5.2 CODIFICACIÓN BASE64.....	6
4.5.3 CIFRADO SIMÉTRICO VS. ASIMÉTRICO	6
5. DESARROLLO	7
5.1 EXPLICACIÓN Y ESTRUCTURA DEL PROGRAMA.....	7
6. RESULTADOS	10
7. CONCLUSIÓN.....	12
BIBLIOGRAFÍA	12

ÍNDICE DE ILUSTRACIONES

Ilustración 1 Prueba 1.....	10
Ilustración 2 Prueba 2 con símbolos y caracteres	11
Ilustración 3 Prueba 3 sin desenscriptar el texto	11

1. INTRODUCCIÓN

Para el desarrollo de esta práctica, se evaluará la protección de datos sensibles es una tarea crítica en los sistemas de información modernos. Se describe el desarrollo y funcionamiento de un programa elaborado en Java (NetBeans) que utiliza el algoritmo de cifrado simétrico AES (Advanced Encryption Standard).

Durante el desarrollo se optó por hacer uso del algoritmo hash haciendo que el código permita encriptar y desencriptar cadenas de texto, números y caracteres, lo que garantiza la seguridad de la información incluso en escenarios de transmisión o almacenamiento.

2. MATERIALES

- Equipo de cómputo
- Java Development Kit (JDK)
- IDE NetBeans
- Java Cryptography Extension (JCE)

3. OBJETIVO GENERAL

Implementar un programa en Java que permita cifrar y descifrar texto utilizando el algoritmo AES, asegurando la integridad de los datos y facilitando su uso mediante una interfaz de consola.

3.1 OBJETIVOS ESPECIFICOS

- Comprender y aplicar el algoritmo AES en un entorno de programación.
- Implementar la generación dinámica de claves de cifrado de 128 bits.
- Garantizar la compatibilidad con diferentes tipos de texto, incluyendo caracteres especiales y números.
- Probar el funcionamiento del código en distintos escenarios para validar su robustez y fiabilidad.

4. MARCO TEÓRICO

4.1 ¿QUÉ ES NETBEANS?

NetBeans es un entorno de desarrollo integrado libre, hecho principalmente para el lenguaje de programación Java.

La plataforma NetBeans permite que las aplicaciones sean desarrolladas a partir de un conjunto de componentes de software llamados módulos. Un módulo es un archivo Java que contiene clases de java escritas para interactuar con las API de NetBeans y un archivo especial (manifest file) que lo identifica como módulo. (apache, 2000)

4.2 ¿QUÉ ES EL CIFRADO?

El cifrado es un método de protección de datos que consiste en alterarlos hasta hacerlos ilegibles. Los datos pasan de ser texto sin formato a ser texto cifrado por medio de un método denominado algoritmo. Quien desee acceder a los datos cifrados debe descodificarlos primero con la clave de descifrado correcta.

El cifrado de datos aumenta la seguridad de sus datos confidenciales y su información personal, y le ofrece protección y privacidad en un mundo en línea totalmente imprevisible.

4.3 CIFRADO FRENTE A CRIPTOGRAFÍA

La práctica del cifrado es una forma de criptografía. «Criptografía» procede del griego y etimológicamente significa «escritura secreta». La gente lleva miles de años empleando sistemas criptográficos para ocultar información escrita, de modo que solo el destinatario pueda leerla.

El cifrado no es más que una forma de criptografía. Se trata de un proceso que utiliza cálculos y algoritmos para codificar texto plano de forma más eficiente y

compleja. El cifrado se utiliza principalmente para evitar el uso no autorizado de datos.

4.4 ¿CÓMO FUNCIONA EL CIFRADO?

El cifrado funciona pasando los datos originales (o texto plano) por un algoritmo (o cifra) que los convierte en texto cifrado. El nuevo texto resulta ilegible si no se utiliza la clave de descifrado adecuada para decodificarlo.

Esto es igualmente cierto para los datos en reposo (almacenados en algún sitio, como un disco duro) y en movimiento (en transferencia electrónica de un lugar a otro, por ejemplo, a través de una red o de Internet).

Los algoritmos de cifrado emplean claves criptográficas (cadenas de caracteres) para transformar los datos en un sinsentido aparentemente aleatorio. Los algoritmos modernos descomponen los datos de texto plano en grupos llamados bloques y luego cifran cada bloque como una unidad. Por este motivo se los denomina cifradores de bloques.

Los algoritmos actuales son tan complejos que el mismo texto plano puede terminar como un texto cifrado distinto cada vez que se aplican. Los datos solo se pueden decodificar con la clave para esa sesión de cifrado específica. En resumen, así es como se cifran los datos. (Avast, 2024)

4.5 TIPO DE CIFRADO

4.5.1 ADVANCED ENCRYPTION STANDARD (AES)

AES es un estándar de cifrado simétrico establecido por el Instituto Nacional de Estándares y Tecnología (NIST) en 2001. Es ampliamente utilizado en aplicaciones de seguridad debido a su balance entre velocidad y robustez.

AES-128: una clave de 128 bits en 10 rondas

AES-192: una clave de 192 bits en 12 rondas

AES-256: una clave de 256 bits en 14 rondas

- **Rapidez:** AES es eficiente en software y hardware.
- **Flexibilidad:** Se utiliza en diversas plataformas y dispositivos.
- **Seguridad:** Es resistente a la mayoría de los ataques criptográficos conocidos.

4.5.2 CODIFICACIÓN BASE64

Base64 es una técnica de codificación que convierte datos binarios en texto legible utilizando un conjunto limitado de caracteres ASCII. Es útil para almacenar o transmitir datos cifrados, ya que estos suelen incluir caracteres no imprimibles.

4.5.3 CIFRADO SIMÉTRICO VS. ASIMÉTRICO

- **Cifrado simétrico:** Utiliza la misma clave para cifrar y descifrar los datos. Es rápido y adecuado para grandes volúmenes de información, pero requiere un método seguro para compartir la clave.
- **Cifrado asimétrico:** Utiliza un par de claves (pública y privada). Es más seguro para la transmisión de datos, pero más lento que el cifrado simétrico. (López, 2024)

5. DESARROLLO

Dicho programa desarrollado implementa el cifrado y descifrado de texto en Java utilizando el algoritmo AES y una interfaz de consola para la interacción del usuario.

5.1 EXPLICACIÓN Y ESTRUCTURA DEL PROGRAMA

1. Primeramente, se emplea el generador de claves de Java para crear una clave de 128 bits, que es el parámetro fundamental para el cifrado y descifrado que hemos elegido.

```
private static final String ALGORITHM = "AES";
```

```
// Genera la clave secreta para AES
```

```
public static SecretKey generateKey() throws Exception {
```

```
    KeyGenerator keyGenerator = KeyGenerator.getInstance(ALGORITHM);
```

```
    keyGenerator.init(128); // Tamaño de clave puede ser ajustable
```

```
    return keyGenerator.generateKey();
```

```
}
```

2. En seguida encriptamos la cadena proporcionada por el usuario convirtiéndolo a String

```
public static String encrypt(String number, SecretKey secretKey) throws  
Exception {
```

```
    Cipher cipher = Cipher.getInstance(ALGORITHM);
```

```
    cipher.init(Cipher.ENCRYPT_MODE, secretKey);
```

```
    byte[] encryptedBytes = cipher.doFinal(number.getBytes());
```

```
        return Base64.getEncoder().encodeToString(encryptedBytes); // Convertir a
String
    }
}
```

3. Después implementamos un método donde lo denominamos decrypt y este se encarga de convertir un texto cifrado (en formato Base64) en su versión original utilizando el algoritmo AES y la clave secreta proporcionada.

```
public static String decrypt(String encryptedNumber, SecretKey secretKey) throws
Exception {
```

4. En seguida creamos una instancia de Cipher configurada para el algoritmo AES, asimismo lo inicializamos, decodificamos Base64 permitiendo que convierta el texto cifrado en un arreglo de bytes legible, en seguida para el descifrado utilizamos la clave secreta para revertir el cifrado y finalmente retorna el texto original como un String.

```
    public static String decrypt(String encryptedNumber, SecretKey secretKey)
throws Exception {
```

```
        Cipher cipher = Cipher.getInstance(ALGORITHM);
        cipher.init(Cipher.DECRYPT_MODE, secretKey);
        byte[] decodedBytes = Base64.getDecoder().decode(encryptedNumber);
        byte[] decryptedBytes = cipher.doFinal(decodedBytes);
        return new String(decryptedBytes); // Convertir de nuevo a String
    }
```

5. Finalmente, agregamos el método main inicializando así el objeto Scanner pidiéndole al usuario que ingrese datos desde consola para posteriormente crear la clave secreta AES convirtiendo el texto en una cadena cifrada en

Base64, después le preguntan al usuario se desea descifrar la cadena, ¿¿si el usuario acepta muestra el texto original que anteriormente haya ingresado, en casi contrario solo termina el programa mostrando un mensaje “Operación Terminada!!”.

```
public static void main(String[] args) {  
  
    try (Scanner scanner = new Scanner(System.in)) {  
  
        // Solicitar cadena de caracteres o numeros al usuario  
  
        System.out.println("Ingresa tu texto para encriptar:");  
  
        String number = scanner.nextLine();  
  
  
        // Generar clave AES  
  
        SecretKey secretKey = generateKey();  
  
  
        // Encriptar el número  
  
        String encryptedNumber = encrypt(number, secretKey);  
  
        System.out.println("Cadena encriptada: " + encryptedNumber);  
  
        // Preguntar al usuario si desea desencriptar  
  
        System.out.println("¿Quieres desencriptar tu cadena? (s/n):");  
  
        String response = scanner.nextLine();  
  
        if (response.equalsIgnoreCase("s")) {  
  
            // Desencriptar el número  
  
            String decryptedNumber = decrypt(encryptedNumber, secretKey);  
  
            System.out.println("Cadena de caracteres desencriptad@: " +  
decryptedNumber);  
        }  
    }  
}
```

```

    } else {

        System.out.println("Operación terminada!!");

    }

} catch (Exception e) {

    e.printStackTrace();

}

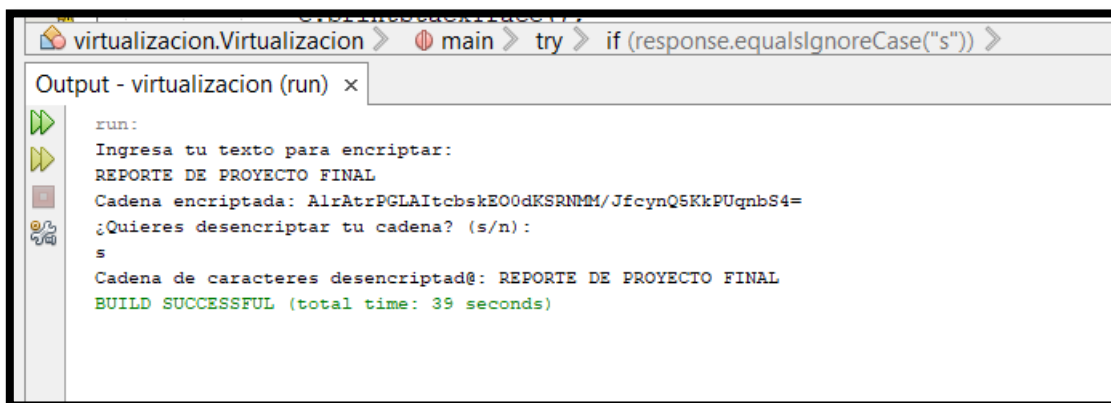
```

6. RESULTADOS

En la siguiente ilustración podemos observar que el programa permite cifrar un texto ingresado por el usuario. En este caso, el texto ingresado fue: REPORTE DE PROYECTO FINAL. Tras procesarlo, el sistema genera una cadena encriptada, que en este ejemplo es: AlrÄtrPGLAItcbskEO0dKSRNMf/JfcynQSKkPUqnbS4=.

Posteriormente, el programa consulta si se desea descriptar la cadena. Al seleccionar la opción s(sí), el texto original se recupera exitosamente, mostrando: REPORTE DE PROYECTO FINAL.

Finalmente, se observa que el proceso fue completado satisfactoriamente en un tiempo total de 39 segundos, demostrando la funcionalidad adecuada del sistema para encriptar y descriptar información.



```

virtualizacion.Virtualizacion > main > try > if (response.equalsIgnoreCase("s")) >
Output - virtualizacion (run) x
run:
Ingresa tu texto para encriptar:
REPORTE DE PROYECTO FINAL
Cadena encriptada: AlrÄtrPGLAItcbskEO0dKSRNMf/JfcynQSKkPUqnbS4=
¿Quieres descriptar tu cadena? (s/n):
s
Cadena de caracteres descriptad@: REPORTE DE PROYECTO FINAL
BUILD SUCCESSFUL (total time: 39 seconds)

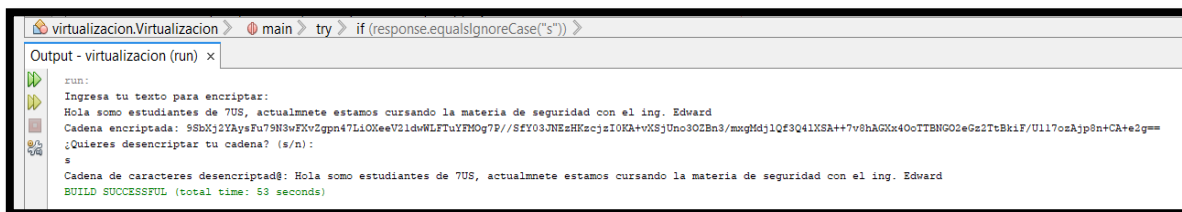
```

Ilustración 1: Prueba 1

En esta captura podemos observar que el programa permite cifrar un texto ingresado por el usuario. En este caso, el texto ingresado fue:

“Hola somos estudiantes de 7US, actualmente estamos cursando la materia de seguridad con el ing. Edward”

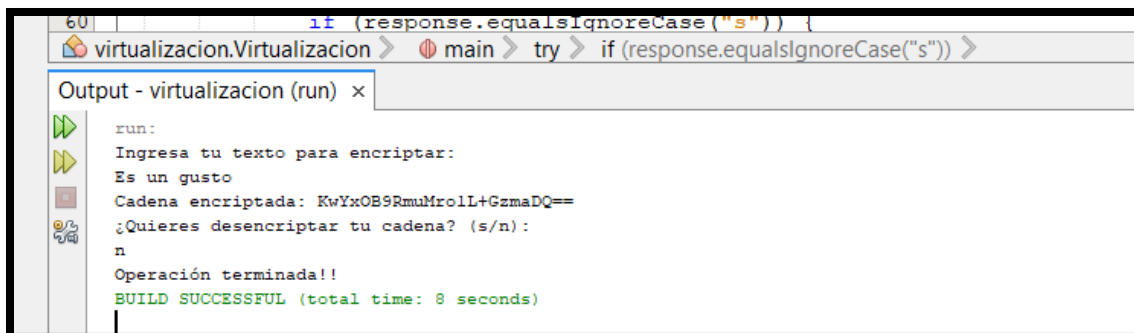
Tras procesarlo, el sistema genera la cadena encriptada, posteriormente, al elegir la opción s(sí), el programa desencripta la cadena y recupera correctamente el mensaje original.



```
virtualizacion.Virtualizacion > main > try > if (response.equalsIgnoreCase("s")) >
Output - virtualizacion (run) x
run:
Ingresa tu texto para encriptar:
Hola somos estudiantes de 7US, actualmente estamos cursando la materia de seguridad con el ing. Edward
Cadena encriptada: 9bKj2YAysFu79H3vFKv2gpn47L1OXeeV2ldwLFTuYFMog7P//SfY03JNEzHKcjsI0KA+vXSjUno302Bn3/mxgMdj1Qf9Q41XSA++7v8hAGX40oTTBNG02eGz2TcBkiF/U117ozAjp8n+CA+e2g==
¿Quieres desencriptar tu cadena? (s/n):
s
Cadena de caracteres desencriptad$: Hola somos estudiantes de 7US, actualmnete estamos cursando la materia de seguridad con el ing. Edward
BUILD SUCCESSFUL (total time: 53 seconds)
```

Ilustración 2: Prueba 2 con símbolos y caracteres.

El programa permite cifrar un texto ingresado por el usuario. En este caso, el texto ingresado fue: “Es un gusto”. Tras procesarlo, el sistema genera una cadena encriptada, que en este ejemplo es: KwXzO89RmuMrolL+GzmaDQ==. Posteriormente, al ser consultado si se desea desencriptar la cadena, se elige la opción n(no) para verificar que el programa funciona correctamente al no proceder con la desencriptación. Como resultado, el sistema finaliza la operación sin errores, demostrando que respeta la elección del usuario y no muestra el texto original. Este proceso se fabricó exitosamente en un tiempo total de 8 segundos.



```
60 if (response.equalsIgnoreCase("s")) {
virtualizacion.Virtualizacion > main > try > if (response.equalsIgnoreCase("s")) >
Output - virtualizacion (run) x
run:
Ingresa tu texto para encriptar:
Es un gusto
Cadena encriptada: KwYxOB9RmuMrolL+GzmaDQ==
¿Quieres desencriptar tu cadena? (s/n):
n
Operación terminada!!
BUILD SUCCESSFUL (total time: 8 seconds)
```

Ilustración 3: Prueba 3 sin desencriptar el texto

- **Funcionalidad:** El sistema encripta y desencripta correctamente diferentes tipos de entradas.
- **Eficiencia:** Los tiempos de ejecución varían según la longitud del texto.
- **Usabilidad:** La interacción es clara, permitiendo al usuario decidir si desea desencriptar el mensaje.

7. CONCLUSIÓN

Durante el desarrollo de dicho proyecto hemos notado como equipo la importancia que tiene la implementación del código ya que es un excelente punto de partida para aprender sobre criptografía simétrica. Aunque presenta limitaciones en términos de gestión de claves y persistencia de datos, logramos cumplir con el objetivo principal de demostrar el funcionamiento básico del cifrado y descifrado, proporcionando una solución efectiva y práctica para comprender el proceso utilizando el algoritmo AES (Advanced Encryption Standard).

Este tipo de solución es fundamental en el contexto de la ciberseguridad moderna, donde proteger la información personal y empresarial es una prioridad en la actualidad. La combinación de ambos enfoques facilita la optimización de recursos y asegura la entrega de productos de alta calidad, a tiempo y con buenos resultados, preparándonos para futuros proyectos o retos que se nos puedan presentar a lo largo de nuestra carrera.

BIBLIOGRAFÍA

- *apache*. (12 de junio de 2000). *apache*: <https://es.wikipedia.org/wiki/NetBeans>
- *Avast*. (2024). *Avast*: <https://www.avast.com/es-es/c-encryption>
- López, A. (23 de Octubre de 2024). *Redes Zone*. *Redes Zone*: <https://www.redeszone.net/tutoriales/seguridad/criptografia-algoritmos-clave-simetrica-asimetrica/>