

# Reporte de Vulnerabilidad

<b>Document Creator</b>	Omar Alejandro Hinojosa Ochoa
<b>Revision</b>	1
<b>Last updated</b>	07/12/2023
<b>Owner</b>	
<b>Reviewer</b>	

# Contents

<b>Findings</b>	<b>2</b>
Critical severity vulnerabilities	2
Evidencia	2
High severity vulnerabilities	3
Evidencia	3
Mid severity vulnerabilities	5
Evidencia	5
Low severity vulnerabilities	6
Evidencia	7
<b>Summary</b>	<b>8</b>

# Findings

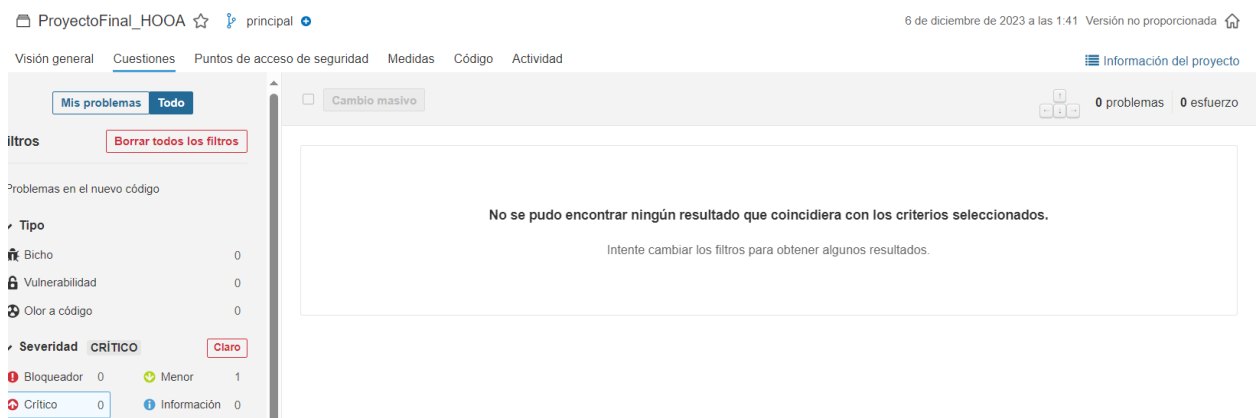
## Critical severity vulnerabilities

SEVERITY: **Critical**

No fueron detectadas vulnerabilidades de severidad critica en el escaneo del proyecto.

Evidencia:

1.



## Recommendations

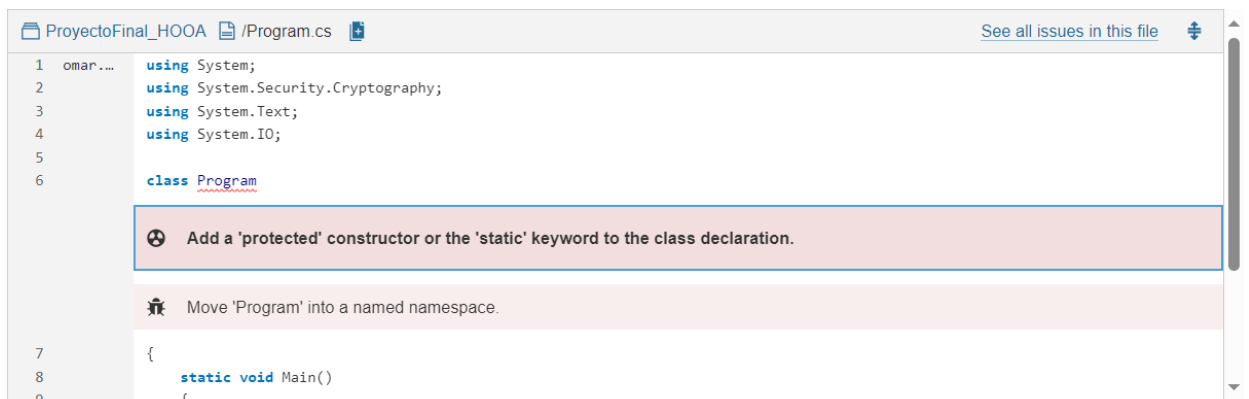
- La gestión de vulnerabilidades crítica es una parte crucial de la seguridad de cualquier sistema o aplicación.  
Por lo tanto, debemos priorizar estas vulnerabilidades.

# High severity vulnerabilities

SEVERITY: **High**

Un constructor estático es un tipo especial de constructor que se utiliza para inicializar cualquier dato estático o realizar otras tareas que deben ejecutarse una sola vez cuando la clase se carga por primera vez en la memoria. Los constructores estáticos no pueden tener modificadores de acceso y se ejecutan antes de que se acceda a cualquier miembro estático de la clase o se crea una instancia de la clase.

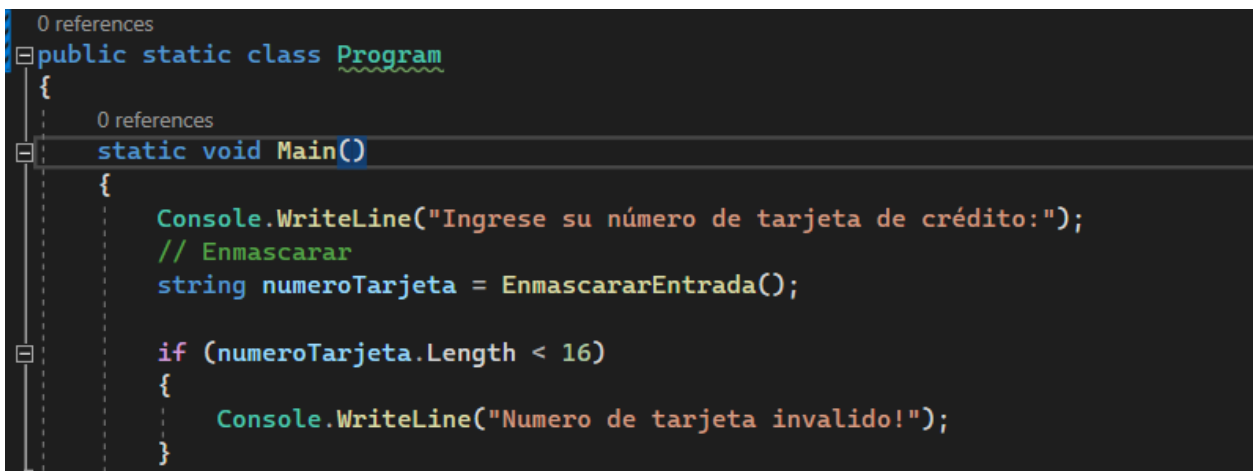
Evidencia:



```
1  using System;
2  using System.Security.Cryptography;
3  using System.Text;
4  using System.IO;
5
6  class Program
7  {
8      static void Main()
9      {
```

Add a 'protected' constructor or the 'static' keyword to the class declaration.

Solucion:



```
0 references
public static class Program
{
    0 references
    static void Main()
    {
        Console.WriteLine("Ingrese su número de tarjeta de crédito:");
        // Enmascarar
        string numeroTarjeta = EnmascararEntrada();

        if (numeroTarjeta.Length < 16)
        {
            Console.WriteLine("Numero de tarjeta invalido!");
        }
    }
}
```

## **Recommendations**

- Los constructores permiten inicializar los miembros de una clase cuando se crea una instancia. Esto es esencial para que los objetos tengan un estado valida desde el principio.

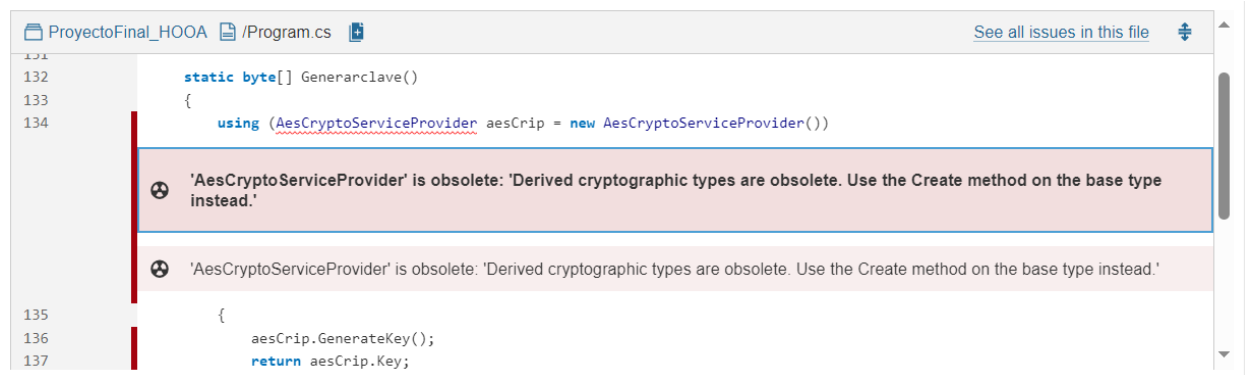
# Mid severity vulnerabilities

SEVERITY: **Medium**

El código obsoleto se refiere a partes de un programa que han sido marcadas como desaconsejadas o en desuso. Aunque el código obsoleto sigue siendo funcional, no es recomendable seguir utilizándolo.

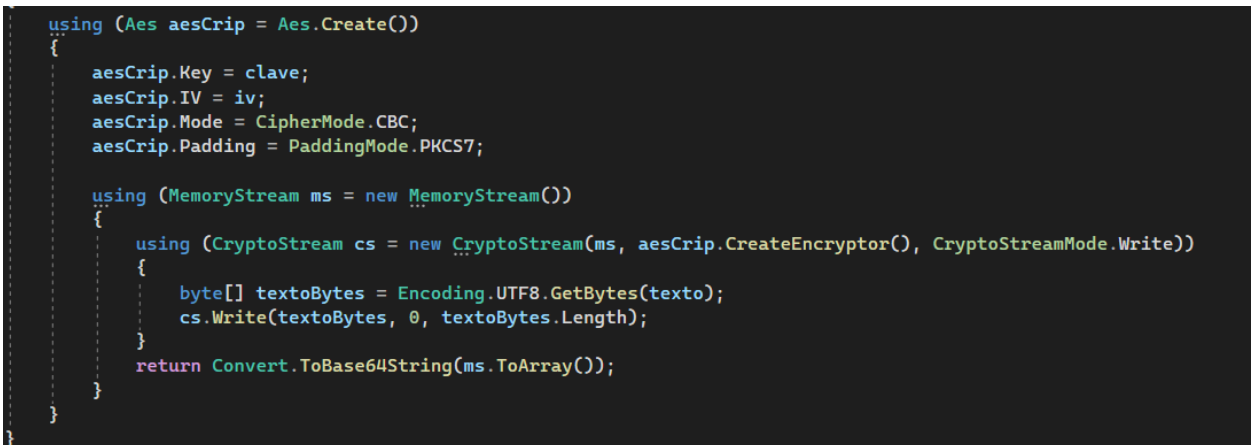
Evidencia:

2.



The screenshot shows a code editor window titled 'ProyectoFinal\_HOOA' with a file named 'Program.cs'. The code is in C# and shows a static method 'GenerarClave()' that creates an instance of 'AesCryptoServiceProvider'. A red squiggly line under the class name indicates an issue. A warning box is displayed, stating: 'AesCryptoServiceProvider' is obsolete: 'Derived cryptographic types are obsolete. Use the Create method on the base type instead.'

3.



```
using (Aes aesCrip = Aes.Create())
{
    aesCrip.Key = clave;
    aesCrip.IV = iv;
    aesCrip.Mode = CipherMode.CBC;
    aesCrip.Padding = PaddingMode.PKCS7;

    using (MemoryStream ms = new MemoryStream())
    {
        using (CryptoStream cs = new CryptoStream(ms, aesCrip.CreateEncryptor(), CryptoStreamMode.Write))
        {
            byte[] textoBytes = Encoding.UTF8.GetBytes(texto);
            cs.Write(textoBytes, 0, textoBytes.Length);
        }
        return Convert.ToBase64String(ms.ToArray());
    }
}
```

## Recommendations

- Evitar el uso de código obsoleto es esencial para mantener la calidad, la seguridad y la mantenibilidad del programa o sistema.

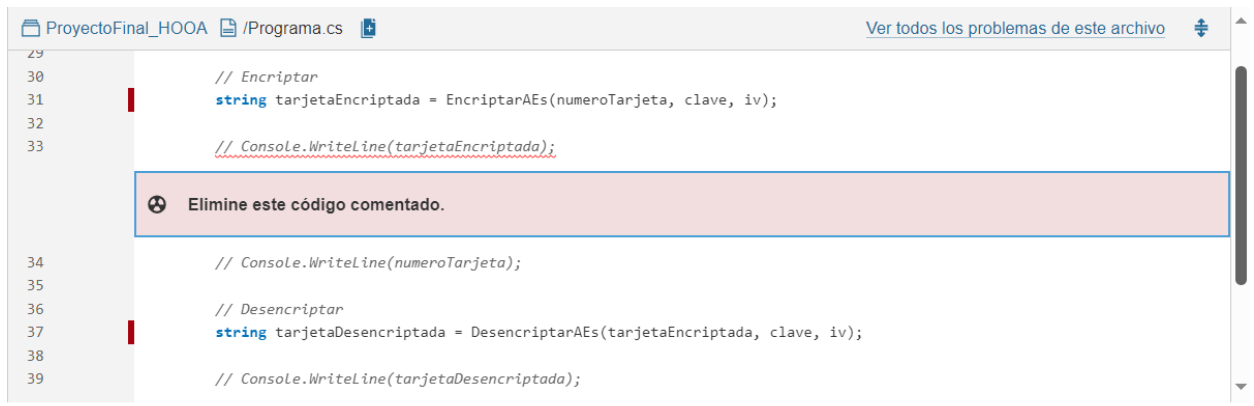
# Low severity vulnerabilities

SEVERITY: **Low**

Los programadores no deben comentar el código, ya que infla los programas y reduce la legibilidad. El código no utilizado debe eliminarse y se puede recuperar del historial de control de código fuente si es necesario. Yo considero que esto puede ser tomado como un falso positivo.

Evidencia:

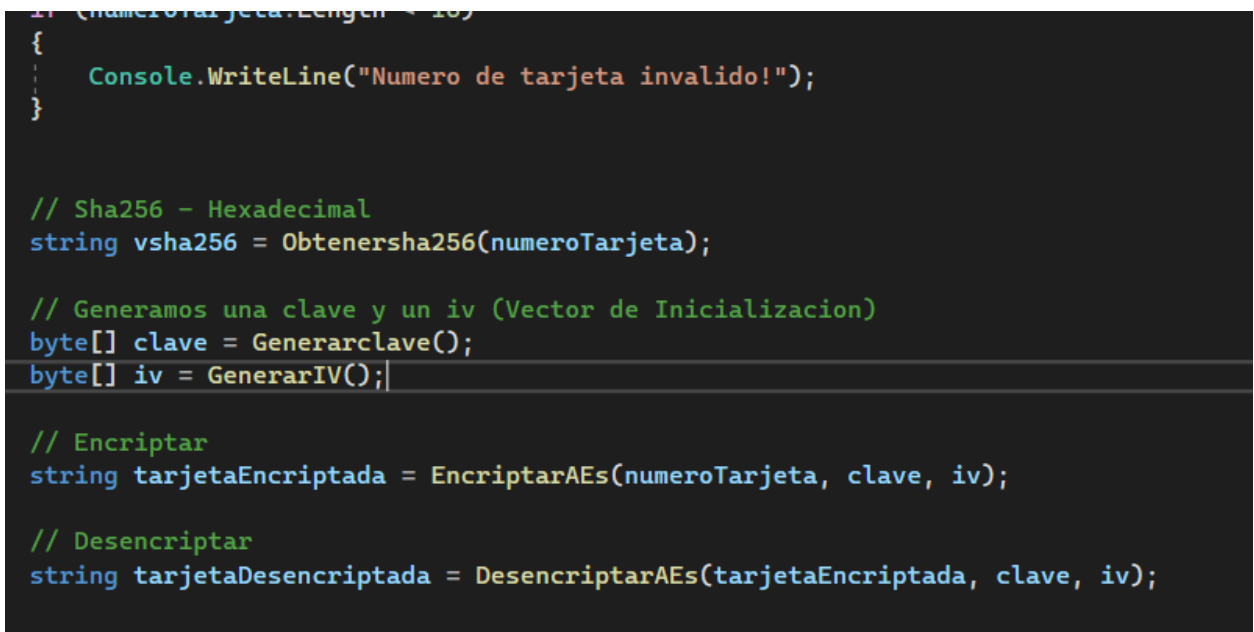
4.



The screenshot shows a Visual Studio editor window titled 'ProyectoFinal\_HOOA' with the file 'Programa.cs' open. The code contains several commented-out lines. A red squiggly line under the comment '// Console.WriteLine(tarjetaEncriptada);' on line 33 is highlighted. A tooltip box appears over this line with the text 'Elimine este código comentado.' (Remove this commented code).

```
29
30 // Encriptar
31 string tarjetaEncriptada = EncriptarAes(numeroTarjeta, clave, iv);
32
33 // Console.WriteLine(tarjetaEncriptada);
34
35 // Console.WriteLine(numeroTarjeta);
36
37 // Desencriptar
38 string tarjetaDesencriptada = DesencriptarAes(tarjetaEncriptada, clave, iv);
39
40 // Console.WriteLine(tarjetaDesencriptada);
```

Solución:



The screenshot shows the solution code in Visual Studio. It includes a validation check for the card number length, SHA256 hashing, key and IV generation, and the encryption/decryption process.

```
17 if (numeroTarjeta.Length < 10)
18 {
19     Console.WriteLine("Numero de tarjeta invalido!");
20 }
21
22 // Sha256 - Hexadecimal
23 string vsha256 = ObtenerSha256(numeroTarjeta);
24
25 // Generamos una clave y un iv (Vector de Inicializacion)
26 byte[] clave = GenerarClave();
27 byte[] iv = GenerarIV();
28
29 // Encriptar
30 string tarjetaEncriptada = EncriptarAes(numeroTarjeta, clave, iv);
31
32 // Desencriptar
33 string tarjetaDesencriptada = DesencriptarAes(tarjetaEncriptada, clave, iv);
```

## **Recommendations**

- Tener código comentado dentro de tu programa es código que no se utiliza por lo tanto lo recomendable sería eliminarlo, ya que esto ayuda a que el tamaño del código no incremente más de lo necesario y su lectura sea más clara.



## MICROSOFT TREAT

# Threat Modeling Report

Created on 12/21/2023 8:00:44 PM

Threat Model Name:

Owner:

Reviewer:

Contributors:

Description:

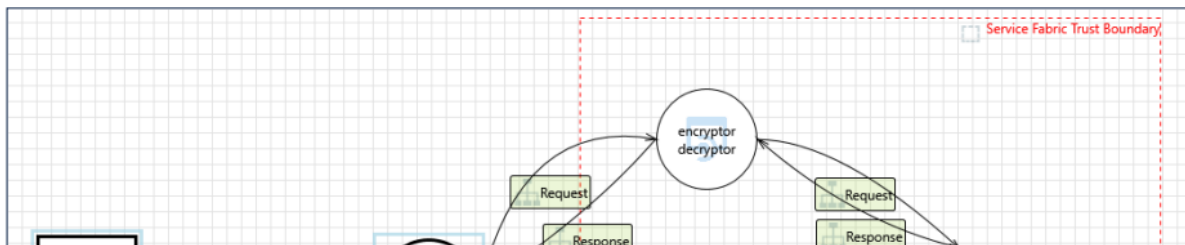
Assumptions:

External Dependencies:

### Threat Model Summary:

Not Started	156
Not Applicable	0
Needs Investigation	0
Mitigation Implemented	0
Total	156
Total Migrated	0

### Diagram: Diagram 1



## MITRE ATT&CK

Fueron seleccionado los grupos de amenazas FIN5, FIN6 y CARBANAK.

FIN5: es un grupo de amenazas motivado financieramente que se ha dirigido a la información de identificación personal y a la información de las tarjetas de pago.

FIN6: es un grupo de ciberdelincuencia que ha robado datos de tarjetas de pago y los ha vendido con fines de lucro en mercados clandestinos.

CARBANAK: es un grupo de ciberdelincuentes que ha utilizado el malware Carbanak para atacar instituciones financieras desde al menos 2013.

Estos grupos fueron seleccionados con su relación a ataques financieros que va relacionado un con el proyecto realizado para el entregable 2. Ya que en nuestro proyecto obtenemos información de la tarjeta de crédito por lo que estos grupos podrían ser una amenaza.

### Tecnica:

Valid accounts: La superposición de credenciales y permisos en una red de sistemas es motivo de preocupación porque el adversario puede ser capaz de pivotar entre cuentas y sistemas para alcanzar un alto nivel de acceso.

### Mitigación:

Políticas de contraseñas: Las aplicaciones y los dispositivos que utilizan el nombre de usuario y la contraseña predeterminados deben cambiarse inmediatamente después de la instalación y antes de la implementación en un entorno de producción.

Gestión de cuentas de usuario: Asegúrese de que los usuarios y los grupos de usuarios tengan los permisos adecuados para sus roles.

Reporte ATT&amp;CK

Enterprise ATT&amp;CK v14

Linux, macOS, Windows,  
Network, PRE, Containers,  
Office 365, SaaS, Google  
Workspace, IaaS, Azure AD

[illegible]

# Summary

Para la realización de esta actividad fue necesario escanear el programa realizado en el entregable 2, el cual se trató implementar un poco de la teoría obtenida durante el curso "Seguridad en Aplicaciones y Mejores Prácticas de Seguridad en Codificación".

En este trabajo se encuentran algunas de las vulnerabilidades encontradas al escanear el programa desarrollado para el segundo entregable. Estas vulnerabilidades estarán organizadas por los niveles Críticos, Altos, Medios y Bajos. También se agregaron algunos falsos positivos.

Lo agregado en este documento fue obtenido del análisis realizado en SonarQube del proyecto.

También se agregó lo realizado con las herramientas MICROSOFT THREAT MODELLING y MITRE ATT&CK, para complementar la información de este entregable.

## Security risk matrix

Risk Matrix	Qty
Critical	
High	
Medium	
Low	