Profile Weight Calculations Using AES Encryption

Faik Doruk Akgüney
Department of Computer Engineering
Cankaya University
Ankara, Turkey
dorukakguney@hotmail.com

***Abstract*—The primary goal of this project is to calculate the weights of steel profiles. During these calculations, weight information can be incorrectly obtained or accidentally changed. To eliminate this issue, the user of the program will select the profile they want to calculate, enter the quantity and length of this profile, and then calculate the weight of the material. The program has a screen for the user and a data entry screen where data can be entered with encryption. The person who wants to enter data will be able to upload data to the program after entering their username and password. When the user uploads the weight library to the program, the program will take this data and encrypt it using the AES encryption method. With this encryption, no one except the person who entered the data will be able to make changes to the weight library. The person using the program will be able to use all the uploaded data for their operations as they wish, but they will not be able to see or change the data.**

***Index Terms*—AES Encryption, C#, Data Security, Profile Weight Calculations**

I. Introduction

In previously used calculation programs, users accidentally changed the values in the weight libraries, leading to calculation errors. Additionally, when the values that needed to be changed were not uploaded to the system, the correct data was present in some users' systems while others had incorrect data. To solve this issue, it was necessary to store the data in one place and ensure that everyone used that data. This way, any change made in the library would affect everyone's calculations. Consequently, different values would not be present, and the results would be calculated with updated values. There were two important points here. First, the weight library should not be changeable. Second, the changes made should take effect immediately. This project aims to address these two issues.

II. Team Members and Project Details

- **Team Member:** Faik Doruk Akgüney
- **Student ID:** 202371202
- **Project Name:** Profile Weight Calculations Using AES Encryption

III. Project Idea

The project aims to provide a secure method for calculating and managing profile weights using AES encryption. Users can:

- Log in to the application.
- Encrypt and save profile weight data.
- Decrypt and load profile weight data.
- Perform calculations on the profile weight data.

A. Use Cases and User Roles

- **Admin:** Manages the overall application and has access to all features.
- **User:** Can log in, load data, save data, and perform profile weight calculations.

IV. Literature Review

AES encryption has been widely studied and implemented in various applications for data security. It is known for its simplicity, efficiency, and strong security properties. AES is used in numerous applications, ranging from securing communication channels to protecting data in databases.

The choice of AES encryption in this project is due to its proven reliability and efficiency. The literature shows that AES is a robust encryption standard that provides high levels of security with relatively low computational overhead. This makes it suitable for applications where both security and performance are critical.

V. Project Plan

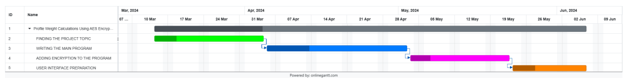

Fig. 1. Gantt Chart of Project Plan

A. Work Breakdown Structure

- Initialization
- Data Encryption
- Data Decryption
- User Interface Development
- Testing and Validation

VI. Application Interface

The application is a desktop application developed using Windows Forms in C#. Users interact with the application through a graphical user interface (GUI) to manage and calculate profile weights securely.

A. Running Platform Details

- **Platform:** Windows Desktop
- **IDE:** Visual Studio 2022
- **Framework:** .NET Framework 4.7.2

VII. Technologies Used

- **Programming Languages:** C#
- **Platforms:** .NET Framework
- **Frameworks:** Windows Forms
- **Libraries:** System.Security.Cryptography

VIII. Cryptographic Properties

A. Algorithms and Protocols

The project uses AES (Advanced Encryption Standard) for encryption and decryption of profile weight data. AES is a symmetric encryption algorithm, meaning the same key is used for both encryption and decryption.

B. Implementations and Libraries

The System.Security.Cryptography library in .NET provides the necessary classes and methods to implement AES encryption and decryption.

C. Crypto Design Plan and Implementation

The encryption and decryption processes are implemented as follows:

- **Encryption:**

```
static string EncryptValue(double value)
{
    using (Aes aesAlg = Aes.Create())
    {
        aesAlg.Key = Key;
        aesAlg.IV = IV;

        ICryptoTransform encryptor = aesAlg.
    CreateEncryptor(aesAlg.Key, aesAlg.IV);

        using (MemoryStream msEncrypt = new
    MemoryStream())
        {
            using (CryptoStream csEncrypt = new
    CryptoStream(msEncrypt, encryptor,
    CryptoStreamMode.Write))
            {
                using (StreamWriter swEncrypt =
    new StreamWriter(csEncrypt))
                {
                    swEncrypt.Write(value.
    ToString());
                }
                return Convert.ToBase64String(
    msEncrypt.ToArray());
            }
        }
    }
}

```

Listing 1. AES Encryption

- **Decryption:**

```
static double DecryptValue(string encryptedValue
    )
{
    byte[] cipherText = Convert.FromBase64String
    (encryptedValue);

    using (Aes aesAlg = Aes.Create())
    {
        aesAlg.Key = Key;
        aesAlg.IV = IV;

        ICryptoTransform decryptor = aesAlg.
    CreateDecryptor(aesAlg.Key, aesAlg.IV);

        using (MemoryStream msDecrypt = new
    MemoryStream(cipherText))
        {
            using (CryptoStream csDecrypt = new
    CryptoStream(msDecrypt, decryptor,
    CryptoStreamMode.Read))
            {
                using (StreamReader srDecrypt =
    new StreamReader(csDecrypt))
                {
                    string plaintext = srDecrypt
    .ReadToEnd();
                    return double.Parse(
    plaintext);
                }
            }
        }
    }
}

```

Listing 2. AES Decryption

IX. Results and Discussion

A. Results

The application successfully encrypts and decrypts profile weight data using AES encryption. The user interface is intuitive, allowing users to manage their data securely. The following results were observed:

- Successful encryption and decryption of data.
- User-friendly interface that simplifies data management.
- Secure storage of sensitive information.

B. Screenshots

Below are the screenshots of the application's user interface:

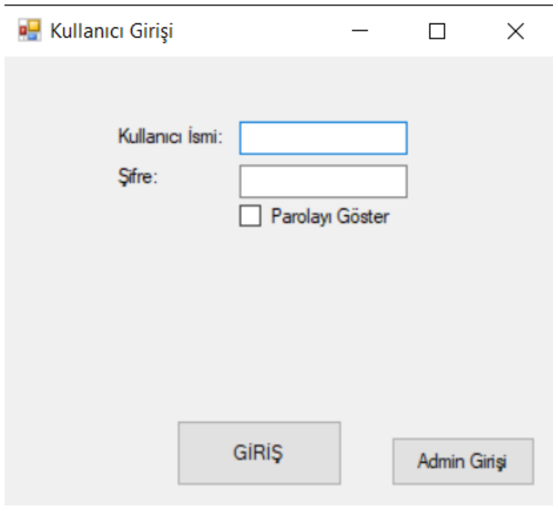

Fig. 2. User Login Form

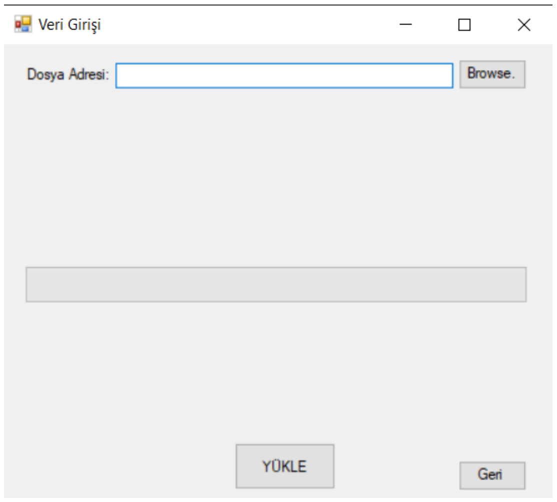

Fig. 3. Data Entry Form

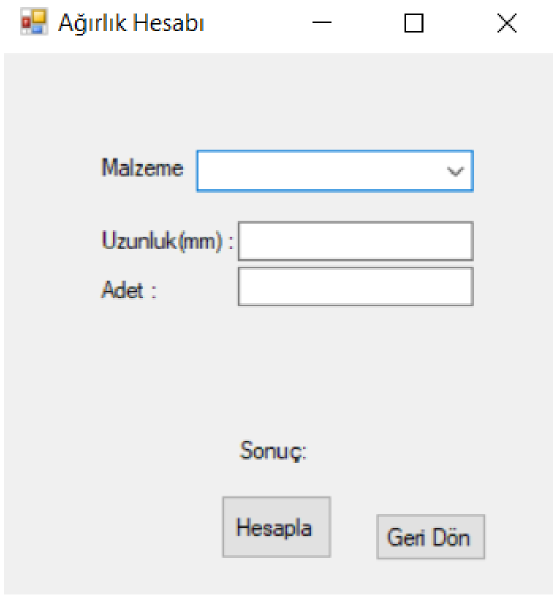

Fig. 4. Profile Weight Calculation Form

C. Discussion

The use of AES encryption ensures that the data remains secure while being stored and processed. AES encryption provides a robust mechanism to protect data from unauthorized access, ensuring that the confidentiality and integrity of the data are maintained throughout its lifecycle.

In this project, AES encryption was chosen for its simplicity and efficiency. It is a symmetric encryption algorithm, meaning the same key is used for both encryption and decryption. This makes it relatively straightforward to implement and manage, especially for applications where performance is a critical consideration.

X. Future Work

The future work section outlines potential improvements and extensions for the project:

- **Advanced Security Features:** Enhance the security features by adding multi-factor authentication and advanced encryption methods. This will provide additional layers of security to protect sensitive data.
- **Performance Optimization:** Optimize the performance of the encryption and decryption processes for larger datasets. This will improve the application's efficiency and scalability.
- **User Interface Improvements:** Enhance the user interface to include more features and make it more user-friendly. This could involve adding more data management tools and improving the overall user experience.
- **Integration with Cloud Services:** Integrate the application with cloud storage services to allow for secure data storage and retrieval over the internet. This would provide greater flexibility and accessibility for users.

XI. Conclusion

This project demonstrates the implementation of profile weight calculations using AES encryption in a C# application. The application provides a secure way to handle sensitive data, ensuring data privacy and security. Future work includes exploring advanced security features, optimizing performance, and enhancing the user interface.

The project successfully met its objectives by providing a secure and efficient method for encrypting and decrypting data. The results demonstrate the effectiveness of AES encryption in protecting sensitive information. By addressing the outlined future work, the application can be further improved to offer even greater security and functionality.

References

[1] FIPS PUB 197: Advanced Encryption Standard (AES). National Institute of Standards and Technology, U.S. Department of Commerce, November 2001.

XII. Appendices

A. Program.cs

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Security.Cryptography;
using System.Text;
using System.Windows.Forms;

namespace Profile_Weight_Calculations
{
    internal static class Program
    {
        public static class GlobalData
        {
            public static Dictionary<string, double>
 Dataset { get; set; } = new Dictionary<string,
double>();
        }
    private static readonly byte[] Key = Encoding.UTF8.GetBytes("A3CDE9ABFC4E0B6CDBE6F1DAE9F29C16"); // 32 bytes for AES-256
    private static readonly byte[] IV = Encoding.UTF8.GetBytes("A1B2C3D4E5F6G7H8"); // 16 bytes for AES

    [STAThread]
    static void Main()
    {
        Application.EnableVisualStyles();
        Application.SetCompatibleTextRenderingDefault(false);
        LoadData();
        Application.Run(new Form1());
        SaveData();
    }

    static void LoadData()
    {
        string txtFilePath = "veriler.txt";

        if (File.Exists(txtFilePath))
        {
            GlobalData.Dataset.Clear();

            try
            {
                string[] lines = File.ReadAllLines(txtFilePath);
                foreach (string line in lines)
                {
                    string[] parts = line.Split(';');
                    if (parts.Length == 2)
                    {
                        string key = parts[0];
                        string encryptedValue = parts[1];
                        double value = DecryptValue(encryptedValue);
                        GlobalData.Dataset[key] = value;
                    }
                }
            }
            catch (Exception ex)
            {
                MessageBox.Show($"Veri okuma hatasi: {ex.Message}");
            }
        }
        else
        {
            GlobalData.Dataset = new Dictionary<string, double>();
            SaveData();
        }
    }

    static void SaveData()
    {
        string txtFilePath = "veriler.txt";

        try
        {
            using (StreamWriter writer = new StreamWriter(txtFilePath, false))
            {
                foreach (var entry in GlobalData.Dataset)
                {
                    string encryptedValue = EncryptValue(entry.Value);
                    writer.WriteLine($"{entry.Key};{encryptedValue}");
                }
            }
        }
        catch (Exception ex)
        {
            MessageBox.Show($"Veri kaydetme hatasi: {ex.Message}");
        }
    }

    static string EncryptValue(double value)
    {
        using (Aes aesAlg = Aes.Create())
        {
            aesAlg.Key = Key;
            aesAlg.IV = IV;

            ICryptoTransform encryptor = aesAlg.CreateEncryptor(aesAlg.Key, aesAlg.IV);

            using (MemoryStream msEncrypt = new MemoryStream())
            {
                using (CryptoStream csEncrypt = new CryptoStream(msEncrypt, encryptor, CryptoStreamMode.Write))
                {
                    using (StreamWriter swEncrypt = new StreamWriter(csEncrypt))
                    {
                        swEncrypt.Write(value.ToString());
                    }
                    return Convert.ToBase64String(msEncrypt.ToArray());
                }
            }
        }
    }

    static double DecryptValue(string encryptedValue)
    {
        byte[] cipherText = Convert.FromBase64String(encryptedValue);

        using (Aes aesAlg = Aes.Create())
        {
            aesAlg.Key = Key;
            aesAlg.IV = IV;

            ICryptoTransform decryptor = aesAlg.CreateDecryptor(aesAlg.Key, aesAlg.IV);

            using (MemoryStream msDecrypt = new MemoryStream(cipherText))
            {
                using (CryptoStream csDecrypt = new CryptoStream(msDecrypt, decryptor, CryptoStreamMode.Read))
                {
                    using (StreamReader srDecrypt = new StreamReader(csDecrypt))
                    {
                        string plaintext = srDecrypt.ReadToEnd();
                        return double.Parse(plaintext);
                    }
                }
            }
            }
        }
    }
}
```

B. Form1.cs

```
using System;
using System.Windows.Forms;

namespace Profile_Weight_Calculations
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
            this.FormClosing += Form_Closing;
        }

        private void btnLogin_Click(object sender,
    EventArgs e)
        {
            string username = txtUserName.Text;
            string password = txtPassword.Text;

            if (string.IsNullOrEmpty(username) ||
    string.IsNullOrEmpty(password))
            {
                MessageBox.Show("Lutfen Kullanici
    Adinizi ve Sifreyi Girin");
                return;
            }

            CheckCredentials(username, password);
        }

        private void CheckCredentials(string
    username, string password)
        {
            if (this.Text == "Admin Girisi" &&
    username == "doruk" && password == "123456")
            {
                OpenForm(new Form2());
            }
            else if (this.Text == "Kullanici Girisi"
     && username == "Doruk" && password == "123456")
            {
                OpenForm(new Form3());
            }
            else
            {
                MessageBox.Show("Kullanici adi veya
    sifre yanlis!");
            }
        }

        private void OpenForm(Form form)
        {
            this.Hide();
            form.Show();
        }

        private void Form_Closing(object sender,
    FormClosingEventArgs e)
        {
            if (e.CloseReason == CloseReason.
    UserClosing)
            {
                Application.Exit();
            }
        }

        private void btnChange_Click_1(object sender
    , EventArgs e)
        {
            if (this.Text == "Admin Girisi")
            {
                this.Text = "Kullanici Girisi";
                btnChange.Text = "Admin Girisi";
            }
            else
            {
                this.Text = "Admin Girisi";
                btnChange.Text = "Kullanici Girisi";
            }
        }
    }
}
```

C. Form2.cs

```
using System;
using System.IO;
using System.Windows.Forms;
using static Profile_Weight_Calculations.Program;

namespace Profile_Weight_Calculations
{
    public partial class Form2 : Form
    {
        public Form2()
        {
            InitializeComponent();
            this.FormClosing += Form_Closing;
        }

        private void btnDownload_Click(object sender
    , EventArgs e)
        {
            string filePath = txtBrowse.Text.Trim();

            if (string.IsNullOrEmpty(filePath))
            {
                MessageBox.Show("Lutfen bir dosya
    secin.");
                return;
            }

            try
            {
                string[] lines = File.ReadAllLines(
    filePath);
                int lineCount = 0;

                foreach (string line in lines)
                {
                    string[] parts = line.Split(';')
    ;
                    if (parts.Length == 2)
                    {
                        string key = parts[0].Trim()
    ;
                        if (double.TryParse(parts
    [1].Trim(), out double value))
                        {
                            GlobalData.Dataset[key]
    = value;
                            lineCount++;
                        }
                    }
                }

                MessageBox.Show($"Dosya Okuma
    Tamamlandi. Eklenen Veri Sayisi: {lineCount}");
            }
            catch (Exception ex)
            {
                MessageBox.Show($"Dosya okuma hatasi : {ex.Message}");
            }
        }

        private void Form_Closing(object sender, FormClosingEventArgs e)
        {
            if (e.CloseReason == CloseReason.UserClosing)
            {
                Application.Exit();
            }
        }

        private void btnBrowse_Click_1(object sender, EventArgs e)
        {
            OpenFileDialog openFileDialog = new OpenFileDialog();
            openFileDialog.Filter = "Metin Dosyalari (*.txt)|*.txt";

            if (openFileDialog.ShowDialog() == DialogResult.OK)
            {
                txtBrowse.Text = openFileDialog.FileName;
            }
        }
    }
}
```

D. *Form3.cs*

```
using System;
using System.Windows.Forms;
using static Profile_Weight_Calculations.Program;

namespace Profile_Weight_Calculations
{
    public partial class Form3 : Form
    {
        public Form3()
        {
            InitializeComponent();
            this.FormClosing += Form_Closing;
        }

        private void btnCalculate_Click_1(object sender, EventArgs e)
        {
            string selectedValue = cmbDataset.SelectedItem?.ToString();
            if (string.IsNullOrEmpty(selectedValue))
            {
                MessageBox.Show("Lutfen malzeme degerini secin.");
                return;
            }

            if (double.TryParse(txtLength.Text, out double length) && double.TryParse(txtCount.Text, out double count))
            {
                double weight = GlobalData.Dataset[selectedValue];
                double result = length * count * weight;
                lblResult.Text = $"Sonuc: {result} kg/m.";
            }
            else
            {
                MessageBox.Show("Gecersiz uzunluk veya adet degeri.");
            }
        }

        private void Form_Closing(object sender, FormClosingEventArgs e)
        {
            if (e.CloseReason == CloseReason.UserClosing)
            {
                Application.Exit();
            }
        }
    }
}
```