PROJE PLANI

Proje Başlığı: nihani Milli Siber Güvenlik Sistemleri

1. Amaç ve Kapsam: Bilgi güvenliği günümüzde firmalar ve bireyler için büyük önem taşımaktadır. Verilerin güvenliğini sağlamak için şu zamana dek birçok farklı algoritma/teknik geliştirilmiştir. Ancak geliştirilen bu algoritmalar günden güne güvenilirliklerini kaybetmektedir. Hazırladığımız nihani projesinde de Türkiye'ye özgü kriptoloji algoritmaları geliştirmek, Türkiye'yi de bilgi güvenliği alanında söz sahibi yapmak ve geliştirilen algoritmaları test ederek bilgi güvenliğini sağlamak amaçlanmıştır.

2. Yöntem ve Gereçler: Proje geliştirme aşamasında, ilk önce geliştirilecek olan algoritmaların pseudo kodları (sözde kod) yazılmıştır. Ardından hangi dilde geliştirilirse daha kolay ve performansı yüksek olacağına karar verilmiştir. Gerekli kaynak taraması yapıldıktan sonra proje C# diliyle, Visual Studio IDE'siyle geliştirilmeye başlanmıştır. Yüz tanıma sistemi olarak EmguCV kullanılmıştır. Uygulamanın geliştirmesi bittikten sonra olası exceptionlar için testler yapılmıştır.

3. İş-Zaman Tablosu

İşin tanımı	Aylar									
	NİSAN	MAYIS	HAZİRAN	TEMMUZ	AĞUSTOS	EYLÜL	EKİM	KASIM	ARALIK	OCAK
LİTERATÜR TARAMASI			x	x	x	x	x			
VERİ TOPLANMASI				x	x	x	x			
ARAZİ ÇALIŞMASI			x	x	x	x	x	x	x	
PROJE RAPORU YAZIMI									x	x

Proje Başlığı: nihani Milli Siber Güvenlik Sistemleri

Özet: Projemizde Türkiye'ye özgü bilgi şifreleme ve şifre çözme algoritmaları geliştirilmiştir. Proje C# diliyle geliştirilmiş olup VisualStudio2017 kullanılmıştır. Geliştirilen algoritmalar şu şekildedir;

- Simetrik Base64 Şifreleme ve Çözme: Metin.
  - Şifresiz metni Base64 ile şifrele,
  - Anahtarla simetrik şifrele,
  - Tekrar Base64 ile şifrele,
  - Ters Çevir.
- Anakart ID'sine Göre Dosya Şifreleme ve Çözme: Dosya.
  - Anahtar ile anakart ID'sini simetrik şifrele,
  - Oluşan şifreyi şifrelenecek dosya ile şifrele.
- Anakart ID'sine Göre Metin Şifreleme ve Çözme: Metin.
  - Şifresiz metni Base64 ile şifrele,
  - Anahtar ile Anakart ID'sini simetrik şifrele,
  - Oluşan iki metni de simetrik şifrele,
  - En son oluşan çıktıyı Base64 ile şifrele.

  ✓ ***NOT:*** *Eğer bir dosya bir bilgisayarda şifrelendiyse, yalnızca o bilgisayarda çözülür.*

- Değişken Güvenlikli Dosya Şifreleme ve Çözme: Metin.
  - 1 anahtar; anahtar ve o anahtarın tersiyle,
  - 2 anahtar; 1. ve 2. anahtarın simetrik haliyle,
  - 3 anahtar;1. ve 3. anahtarın simetrik haliyle 2. anahtarın şifrelenmesi sonucu oluşan anahtar ile,
  - 4 anahtar;1. ve 3. ile 2. ve 4. anahtarın simetrik şifrelemesiyle oluşan anahtarla metni şifrele.
- Analog Sinyalleri Kullanarak Şifreleme ve Çözme: Her türlü metin ve dosya.
  - Arduino'dan alınan analog sinyalleri çevir,
  - Dosya;
    - Analog string ile dosyayı AES'le şifrele.
  - Metin;
    - Metni Base64 ile şifrele,
    - Oluşan metni girilen anahtar ile simetrik şifrele,
    - En son oluşan metni bir kez daha Base64 ile şifrele.

Giriş yüz tanıma ve PIN koduyla sağlanmıştır. Yüz tanıma sistemi için EmguCV tercih edilmiştir. Projede esinlenilen algoritmalar AES, Base64 ve simetrik şifrelemelerdir.

Anahtar Kelimeler: Kriptoloji, Şifreleme, Kriptolog, Encryption, Bilgi Güvenliği, C Sharp, Arduino, Kripto, Encode, Decode, Aes, Base64, Net Framework, Milli Siber Güvenlik Sistemi, Nihani

PROJE RAPORU

Projenin Adı: nihani Milli Siber Güvenlik Sistemleri

İÇİNDEKİLER

1) Giriş

Kriptoloji, şifre bilimidir. Çeşitli iletilerin, yazıların belli bir sisteme göre şifrelenmesi, bu mesajların güvenli bir ortamda alıcıya iletilmesi ve iletilmiş mesajın deşifre edilmesidir. Geçmiş dönemlerde, örneğin 2. Dünya Savaşı sırasında, devletler birbirleriyle haberleşirken şifreli mesajları kullanmışlardır. Bu da bir kriptoloji uygulamasıdır. Günümüzde ise kriptoloji, bilgi güvenliği, ağ güvenliği, veri iletimi gibi farklı amaçlar için kullanılmaktadır.

Bu projede yerli ve milli kriptoloji algoritmaları geliştirilmiştir. nihani kelimesi, Osmanlı Türkçesinde gizli, saklı anlamına gelen bir sözcüktür. 5 farklı algoritma geliştirilen uygulama C# diliyle yazılmıştır. Uygulama geliştirme sürecinde esinlenilen algoritmalar AES (Advanced Encryption Standard) Algoritması, Base64 Algoritması, Simetrik Şifreleme Algoritmalarıdır. Uygulamaya giriş yüz tanıma sistemi ile ya da PIN kodu ile yapılmaktadır. Yüz tanıma EmguCV ile sağlanmıştır. Eğer algılanan yüz dosya sisteminde kayıtlı ise şifreleme ekranına giriş yapılır. Eğer önceden tanımlı bir yüz değil ise giriş yapılamaz. Aynı şekilde girilen PIN kodu doğru ise şifreleme ekranına giriş yapılır.

Yüz tanıma sistemi ve PIN kodu sistemi eklenerek uygulamanın güvenliği artırılmıştır.

1.1) Projenin Amacı

Bilgi güvenliği günümüzde firmalar ve bireyler için büyük önem taşımaktadır. Verilerin güvenliğini sağlamak için şu zamana dek birçok farklı algoritma/teknik geliştirilmiştir. Ancak geliştirilen bu algoritmalar günden güne güvenilirliklerini kaybetmektedir. Hazırladığımız nihani projesinde de Türkiye'ye özgü kriptoloji algoritmaları geliştirmek, Türkiye'yi de bilgi güvenliği alanında söz sahibi yapmak ve geliştirilen algoritmaları test ederek bilgi güvenliğini sağlamak amaçlanmıştır.

1.2) C#

C#, Microsoft'un geliştirmiş olduğu yeni nesil programlama dilidir. Yine Microsoft tarafından geliştirilmiş Dot NET Teknolojisi için geliştirilmiş dillerden biridir. Özellikle nesne yönelimli programlama kavramının gelişmesine katkıda bulunan en aktif programlama dillerinden biridir. Projede analog sinyal işleme dışında -orada Arduino kullanılmıştır- her yerde C# kullanılmıştır. Dot NET ile uyumsuzluk sorunu yaşanmaması ve performansın düşmemesi adına olabildiğince güncel ve kararlı sürümler kullanılmıştır.

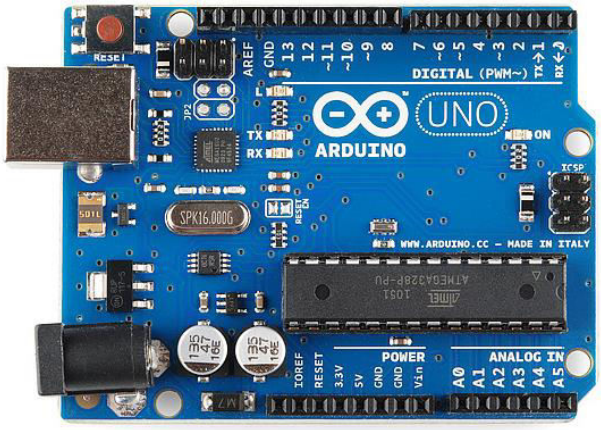
1.3) Microsoft Visual Studio

Microsoft Visual Studio, Microsoft tarafından geliştirilen bir tümleşik geliştirme ortamıdır (IDE). Visual Studio, değişik programlama dillerini destekler, bu da kod editörü ve hata ayıklayıcısının neredeyse tüm programlama dillerini desteklemesini sağlamaktadır. Projedeki C# kodları Visual Studio 2017 Community versiyonu ile geliştirilmiştir.

1.4) Arduino

Arduino bir G/Ç (I/O) kartı ve Processing/Wiring dilinin bir uygulamasını içeren geliştirme ortamından oluşan bir fiziksel programlama platformudur. Arduino tek başına çalışan interaktif nesneler geliştirmek için kullanılabileceği gibi bilgisayar üzerinde çalışan yazılımlara da bağlanabilir. Hazır üretilmiş kartlar satın alınabilir veya kendileri üretmek isteyenler için donanım tasarımı ile ilgili bilgiler mevcuttur. Projede ise A1 portundan alınan analog sinyalleri işleyip seri port aracılığıyla bilgisayara iletir. Arduino'nun teknik özellikleri şu şekildedir;

- Bu platform sayesinde çevresi ile kolayca etkileşime giren sistemler hazırlanabilmekte ve bu sistemler dilendiği şekilde kullanılabilmektedir.
- Arduino tamamen açık kaynaklı olan geliştirme platformlarından bir tanesidir. Bu da birçok yerde kullanılmasını sağlar.
- Arduino kartları üzerinde farklı firmaların kartları bulunmaz.
- Kartlar üzerinde Atmega firmasına ait olan 8 ve 32 bit kapasitesinde mikro denetleyeciler bulunmaktadır.
- Arduino kütüphaneleri oldukça geniştir. Bu geniş kütüphaneler sayesinde mikro denetleyicileri kolayca programlama şansınız bulunmaktadır.
- Bu platformda analog ve dijital girişler bulunmaktadır. Bu sayede analog ve dijital verileri platforma kolayca işleyebilirsiniz.
- Sensörler aracılığı ile gelen verileri kullanabilirsiniz. Dış dünyaya ses, ışık, hareket vb. gibi çıktılar üretebilir ve bunları kullanabilirsiniz.



Şekil 1. Arduino UNO Kartı

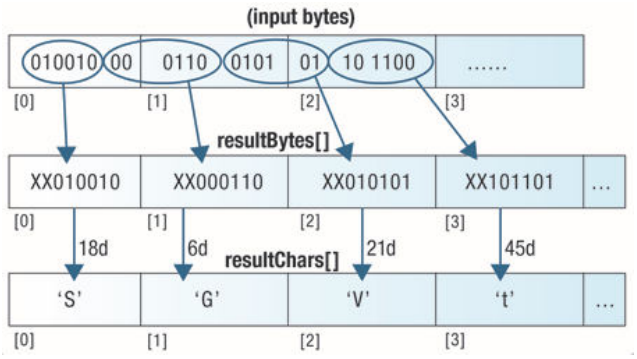
1.5) EmguCV

Emgu CV bir Open CV wrapperidir. Yani Open CV Framework'ünün Dot NET dilleri üzerinde de kullanılabilmesi için oluşturulmuş bir kütüphanedir. Görüntü işleme için kullanılır. Projede yüz tanıma sistemi için OpenCV tercih edilmiştir. OpenCV'nin tercih edilme sebebi ise kolay kullanımı ve çoklu platforma uyumlu olmasıdır.

1.6) Base64

Base64 ikili verilerin sadece ASCII karakterlerini kullanan ortamlarda iletilmesine ve saklanmasına olanak tanıyan bir kodlama şemasıdır. Projede metin şifreleme işlemleri genellikle Base64 ile yapılmıştır.

- Örneğin "TUBITAK" yazısı Base64 ile şifrelenirse çıktı "VFVCSVRBSw==" olacaktır.
- "VFVCSVRBSw==" decode (şifre çözme) işleminden geçerse "TUBITAK" çıktısı alınır.



Şekil 2. Base64 Şifreleme Mekanizması

1.7) AES

AES (Gelişmiş Şifreleme Standardı), elektronik verinin şifrelenmesi için sunulan bir standarttır. Amerikan hükümeti tarafından kabul edilen AES, uluslararası alanda da defacto şifreleme (kripto) standardı olarak kullanılmaktadır. Projede dosya şifreleme algoritması olarak kullanılmıştır. AES algoritmasında giriş, çıkış ve matrisler 128 bitliktir. Matris 4 satır, 4 sütun (4×4), 16 bölmeden oluşur. Bu matrise 'durum' denmektedir. Durumun her bölmesine bir baytlık veri düşer. Her satırda 32 bitlik bir kelimeyi meydana getirir. AES algoritması, 128 bit veri bloklarını 128, 192 veya 256 bit anahtar seçenekleri ile şifreleyen bir blok şifre algoritmasıdır. Anahtar uzunluğu bit sayıları arasındaki farklılık AES tur döngülerinin sayısını değiştirmektedir.

	Kelime Uzunluğu	Tur Sayısı
AES-128	4	10
AES-192	6	12
AES-256	8	14

Şekil 3. AES şifrelemede kelimenin ve tur sayısının şifreleme türüne göre karşılaştırması.

2) Yöntem

Proje geliştirme aşamasında, ilk önce geliştirilecek olan algoritmaların pseudo kodları (sözde kod) yazılmıştır. Ardından hangi dilde geliştirilirse daha kolay ve performansı yüksek olacağına karar verilmiştir. Gerekli kaynak taraması yapıldıktan sonra proje C# diliyle, Visual Studio IDE'siyle geliştirilmeye başlanmıştır.

2.1) Proje Kapsamında Geliştirilen Algoritmalar

a. Simetrik Base64 Metin Şifreleme ve Şifre Çözme Algoritması:

Metin tipindeki verileri şifrelerken ve çözerken kullanılır.

Simetrik Base64 Metin Şifreleme işlemi; kullanıcıdan form yardımıyla şifresiz metin ve anahtar istenir. Eğer kullanıcı anahtar üretmek isterse formda bulunan "ÜRET" butonuna da tıklayabilir. Anahtar ve şifresiz metin girildikten sonra formdaki "ŞİFRELE" butonuna tıklanır. Şifresiz metin ilk önce Base64 ile şifrelenir. Daha sonra girilen anahtar ile simetrik olarak şifrelenir. Oluşan metin tekrar Base64 ile şifrelendikten sonra ters çevrilir. En son oluşan çıktı formdaki "ŞİFRELİ METİN" başlıklı metin kutusuna yazdırılır.

Simetrik Base64 Metin Şifre çözme işlemi; kullanıcıdan Windows formu yardımıyla şifreli metin ve anahtar istenir. Eğer kullanıcı anahtar üretmek isterse formda bulunan "ÜRET" butonuna da tıklayabilir. Anahtar ve şifreli metin girildikten sonra formdaki "ŞİFRE ÇÖZ" butonuna tıklanır. Şifreli metin ilk önce ters çevrilip Base64 ile çözülür. Oluşan metin girilen anahtar ile simetrik olarak çözülüp bir kez daha Base64 ile çözülür. En son oluşan çıktı formdaki "ŞİFRESİZ METİN" başlıklı metin kutusuna yazdırılır.

b. Ana kart ID'sine Göre Dosya Şifreleme ve Şifre Çözme Algoritması:

Müzik, görsel, pdf ya da herhangi bir formattaki dosyayı şifrelerken ve çözerken kullanılır.

Ana kart ID'sine Göre Dosya Şifreleme işlemi; kullanıcı formdaki "AÇ" butonu ile şifrelenecek dosyanın yolunu seçer, "KAYDET" butonu ile yeni (şifreli) dosyanın yolunu seçer ve bir anahtar yazar. Dosya yolları ve anahtar girildikten sonra formdaki "ŞİFRELE" butonuna tıklanır. Girilen anahtar ile ana kartın kendine özgü kodu simetrik olarak şifrelenir. Oluşan şifre, şifrelenecek dosya ile AES şifrelemeye tabi tutulur. Yeni dosya yolunda şifreli dosya oluşturulur.

Ana kart ID'sine Göre Dosya Şifre çözme işlemi; kullanıcı formdaki "AÇ" butonu ile şifresi çözülecek dosyanın yolunu seçer, "KAYDET" butonu ile yeni (şifresiz) dosyanın yolunu seçer ve bir anahtar yazar. Dosya yolları ve anahtar girildikten sonra formdaki "ŞİFRE ÇÖZ" butonuna tıklanır. Girilen anahtar ile ana kartın kendine özgü kodu simetrik olarak şifrelenir. Oluşan şifre, şifresi çözülecek dosya ile AES şifre çözmeye tabi tutulur. Yeni dosya yolunda şifresiz dosya oluşturulur.

NOT: *Eğer bir dosya bir bilgisayarda şifrelendiyse, kullanıcının girdiği anahtar doğru olsa bile şifre çözme işlemi gerçekleşmez.*

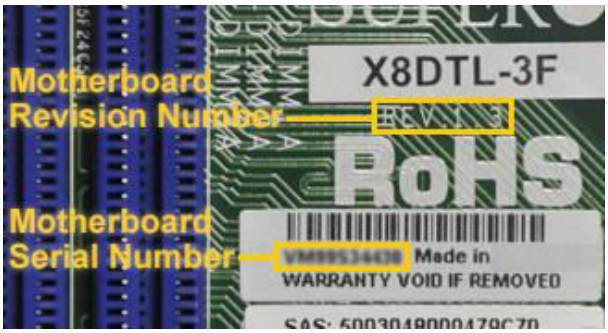
c. Ana kart ID'sine Göre Metin Şifreleme ve Şifre Çözme Algoritması:

Text tipindeki verileri şifrelerken ve çözerken kullanılır.

Ana kart ID'sine Göre Metin Şifreleme işlemi; kullanıcıdan form yardımıyla şifresiz metin ve anahtar istenir. Anahtar ve şifresiz metin girildikten sonra formdaki "ŞİFRELE" butonuna tıklanır. Şifresiz metin ilk önce Base64 ile şifrelenir. Kullanıcının girdiği anahtar ile ana kartın kendine özgü kodu simetrik olarak şifrelenir. Daha sonra simetrik şifreleme sonucu oluşan 2. anahtar, şifresiz metin ile simetrik olarak şifrelenir. Oluşan metin tekrar Base64 ile şifrelenir. En son oluşan çıktı formdaki "ŞİFRELİ METİN" başlıklı metin kutusuna yazdırılır.

Ana kart ID'sine Göre Metin Şifre çözme işlemi; kullanıcıdan form yardımıyla şifreli metin ve anahtar istenir. Anahtar ve şifreli metin girildikten sonra formdaki "ŞİFRE ÇÖZ" butonuna tıklanır. Şifreli metin ilk önce Base64 ile çözüldükten sonra kullanıcının girdiği anahtar ile ana kartın kendine özgü kodu simetrik olarak şifrelenince oluşan 2. anahtar ile simetrik olarak çözülür. Bu metin tekrar Base64 ile çözüldükten sonra en son oluşan çıktı formdaki "ŞİFRESİZ METİN" başlıklı metin kutusuna yazdırılır.

NOT: *Eğer bir metin bir bilgisayarda şifrelendiyse, kullanıcının girdiği anahtar doğru olsa bile şifre çözme işlemi gerçekleşmez.*



Şekil 4. Ana kart üzerinde bulunan ID numarası ve revizyon kimliği

d. Değişken Güvenlikli Dosya Şifreleme ve Şifre Çözme Algoritması:

Kullanıcının isteği doğrultusunda 1 ya da 2 ya da 3 ya da 4 anahtarlı olarak dosya şifrlerken ve şifre çözerken kullanılır.

Değişken Güvenlikli Dosya Şifreleme işlemi; kullanıcı formdan 1 ya da 2 ya da 3 ya da 4 anahtar seçeneklerinden birini seçerek seçtiği sayı doğrultusunda anahtar girer. Daha sonra "AÇ" butonu ile şifrelenecek dosyanın yolunu seçer, "KAYDET" butonu ile yeni (şifreli) dosyanın yolunu seçer. Eğer 1 anahtar seçildiyse, girilen anahtar ile girilen anahtarın tersi bir ortak anahtar oluşturur. Bu ortak anahtar Base64 ile şifrelendikten sonra, oluşan hem şifreli hem ortak anahtar ile şifresiz dosya AES şifreleme tabi tutulur. Eğer 2 anahtar seçildiyse, girilen 1. anahtar ile 2. anahtar simetrik olarak şifrelenir, ortak anahtar Base64 ile de şifrelendikten sonra şifresiz dosya ile AES şifrelemeye tabi tutulur. Eğer 3 anahtar seçildiyse, 1. ve 3. anahtar kendi arasında şifrelenir, daha sonra oluşan ortak şifre, 2. anahtar ile şifrelenir. En sonda oluşan anahtar Base64 ile şifrelendikten sonra şifresiz dosya ile AES şifrelemeye tabi tutulur. Eğer 4 anahtar seçildiyse, 1. ve 3. anahtar kendi arasında, 2. ve 4. anahtar kendi arasında şifrelenir. Oluşan 2 anahtar da kendi arasında şifrelendikten sonra oluşan ortak anahtar Base64 ile şifrelenir ve dosya ile AES şifrelemeye tabi tutulur. Bu işlemler bittikten sonra yeni dosya oluşturulur.

Değişken Güvenlikli Dosya Şifre çözme işleminde de, şifreli dosya seçildikten sonra; 1 anahtar varsa sadece o anahtar ve o anahtarın tersinin oluşturduğu ortak anahtarın Base64 ile şifreli hali, 2 anahtar varsa 1. ve 2. anahtarın oluşturduğu ortak anahtarın Base64 ile şifreli hali, 3 anahtar varsa 1. ve 3.'nün oluşturduğu anahtar ile 2. anahtarın oluşturduğu ortak anahtarın Base64 ile şifreli hali, 4 anahtar varsa 1. ve 3. anahtarın oluşturduğu anahtar ile, 2. ve 4. anahtarın oluşturduğu anahtarın şifrelenerek oluşturduğu ortak anahtarın Base64 ile şifreli hali şifreli dosya ile AES şifre çözmeye tabi tutulur.

e. Analog Sinyallerle Dosya ve Metin Şifreleme ve Şifre Çözme Algoritması:

Analog sinyaller Arduino tarafından işlenip bilgisyara iletildikten sonra dosya ve metin şifrelerken ve çözerken kullanılır.

Metin şifreleme işlemi; kullanıcı Arduino'nun bağlı olduğu seri porta bağlanır. Metin başlıklı metin kutusuna şifrelenecek metni girdikten sonra "ŞİFRELE" butonuna tıklar. Arduino'dan alınan analog sinyaller sayısal veri olarak işlenir. Girilen metin ilk önce Base64 ile şifrelenir, daha sonra simetrik olarak analog sinyallerin sayılsal halleri ile şifrelenir. En son oluşan metin bir kez daha Base64 ile şifrelendikten sonra Şifreli Metin başlıklı metin kutusuna yazdırılır.

Metin şifre çözme işlemi; kullanıcı Arduino'nun bağlı olduğu seri porta bağlanır. Şifreli Metin başlıklı metin kutusuna şifresi çözülecek metni girdikten sonra "ŞİFRE ÇÖZ" butonuna tıklar. Arduino'dan alınan analog sinyaller sayısal veri olarak işlenir. Girilen metin ilk önce Base64 ile çözülür, daha sonra simetrik olarak analog sinyallerin sayılsal halleri ile çözülür. En son oluşan metin bir kez daha Base64 ile çözüldükten sonra Metin başlıklı metin kutusuna yazdırılır.

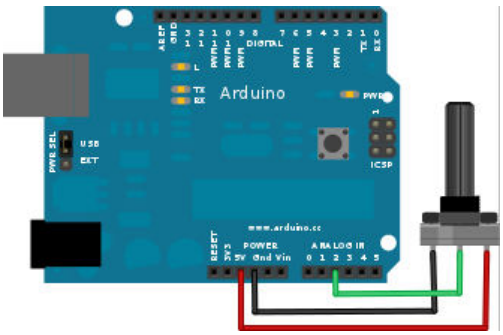
Dosya şifreleme işlemi; kullanıcı Arduino'nun bağlı olduğu seri porta bağlanır. Şifrelencek dosyayı "AÇ" butonu ile seçer ve "ŞİFRELE" butonuna tıklar. Arduino'dan alınan analog sinyaller sayısal veri olarak işlenir. Analog sinyaller temel alınarak AES şifrlemeye tabi tutulur. Şifreli dosya yeni yolda oluşturulur.

Dosya şifre çözme işlemi; kullanıcı Arduino'nun bağlı olduğu seri porta bağlanır. Çözülecek dosyayı "AÇ" butonu ile seçer ve "ŞİFRE ÇÖZ" butonuna tıklar. Arduino'dan alınan analog sinyaller sayısal veri olarak işlenir. Analog sinyaller temel alınarak AES şifre çözmeye tabi tutulur. Şifresi çözülmüş dosya yeni yolda oluşturulur.

NOT: *Sabit bir değerde şifrelenen metin/dosya ancak tekrar şifrelendiği sırada gönderilen analog sinyaller ile çözülebilir.*

- **Analog Sinyal İşleme Birimi**

Arduino kartı kullanılarak kolaylıkla bilgisyara veri gönderimi yapmak veya bilgisyardan veri alıp işlemek mümkündür. Bu projede de örnek olması açısından potansiyometre ile değiştirlen analog sinyaller Arduino ile bilgisayara gönderilmiş, analog şifrelemede anahtar baz alınarak işlemler yapılmıştır. Daha önce de belirtlildiği gibi yalnızca örnek olması açısından potansiyometre kullanılmış; istenildiği takdirde daha farklı şekillerde analog sinyal işlenmesi mümkün kılınmıştır.



Şekil 5. Arduino ve potansiyometre

2.2) Yüz Tanıma Sistemi ve PIN Kodu Sistemi

Bu uygulama bir siber güvenlik uygulamasıdır. Bu nedenle bilgi güvenliği sağlamanın yanında uygulamanın güvenliğini sağlamak da çok önemlidir.

Uygulamanın geliştirilen ilk kararsız versiyonunda uygulamaya direkt giriş yapılırken, güvenliği sağlamak düşüncesiyle yeni versiyonda bu sistemler de ana uygulamaya eklenmiştir.

a) Yüz Tanıma ile Giriş Sistemi

Projenin güvenliğini artırmak amacıyla, uygulamaya giriş anlık olarak yüz tanıma ile de yapılabilmektedir.

Yüz tanıma işlemi için bir OpenCV wrapperi olan EmguCV tercih edilmiştir. Yüz tanıma için web kameradan alınan görüntü kullanılmaktadır.

Uygulamaya istenildiği kadar yüz tanıtma imkânı vardır. Eğer taranan yüz önceden kaydedildi ise şifreleme ekranına giriş yapılır.

EmguCV'nin tercih edilmesinin sebeplerini şu şekilde sıralayabiliriz;

- C# ile uyumludur.
- Hızlıdır.
- Kolayca uygulama geliştirilebilir.
- Cross Platform geliştirmeye olanak sağlar.
- Sürekli güncellenen bir kütüphanedir, bünyesinde farklı lisanslamaları barındırır.

b) PIN Kodu ile Giriş Sistemi

Yüz tanımanın kullanılabilir olmadığı durumlarda da güvenliği sağlamak amacıyla 4 haneli bir PIN kodu ile uygulamaya giriş sağlanmıştır. Kullanıcı doğru kodu girene kadar uygulamada şifreleme ekranı açılmayacaktır.

PIN'deki Hane Sayısı	Mümkün PIN Sayısı
4	10*10*10*10=10000 (10^4)

Şekil 5. PIN hane sayısı ve denemelere göre maksimum mümkün PIN sayısı

2.3) Proje Yapım Basamakları

a) Windows Executable (Çalıştırılabilir Uygulama) Dosyası

- Simetrik Base64 Şifrelemenin oluşturulması, Ana kart ID'sine Göre Metin Şifrelemenin oluşturulması, Ana kart ID'sine Göre Dosya Şifrelemenin oluşturulması, Değişken Güvenlikli Dosya Şifrelemenin oluşturulması, Analog Sinyallerle Şifrelemenin oluşturulması
- Yüz tanıma ve PIN kodu ile giriş formunun oluşturulması
- Açılış ve hakkında formunun oluşturulması

b) Analog Sinyal İşleyici Aygıt

- Arduino ile kullanılacak malzemelerin bir araya getirilmesi
- Kodun Arduino IDE'sinde yazılması
- Yazılan kodun USB yardımıyla Arduino kartına yüklenmesi
- Analog Şifreleme formu ile oluşturulan aygıtın test edilmesi

c) Uygulamada Oluşturulan ve Kullanılan Fonksiyonlar

- Base64Decode(string base64EncodedData)
- Base64Encode(string plainText)
- Reverse(string s)
- uret()
- anahtarlisifre(string text1, string text2)
- EncryptFile(string sourceFile, string targetFile, string key)
- DecryptFile(string sourceFile, string targetFile, string key)
- getMotherBoardID()
- sif(string metin, string anahtar)
- coz(string metin, string anahtar)

3) Bulgular ve Gerçekleşme

3.1) Windows Executable (Çalıştırılabilir Uygulama) Dosyası

a) Simetrik Base64 Şifrelemenin Oluşturulması

Metni hem simetrik olarak hem de Base64 ile şifreleyen ve çözen algoritmanın kodlarının ve formunun oluşturulmasıdır. Kullanıcı şifreleme için şifresiz metin ve anahtar girer, şifre çözme için ise şifreli metin ve anahtar girer.

b) Ana kart ID'sine Göre Metin Şifrelemenin Oluşturulması

Bilgisayarların kimlik numarası sayılan ID numarasını anahtar baz alarak metin şifreleyen ve çözen algoritmanın kodlarının ve formunun oluşturulmasıdır. Kullanıcı şifreleme için şifresiz metin ve anahtar girer, şifre çözme için ise şifreli metin ve anahtar girer.

Şifreleme ve şifre çözme yalnızca aynı bilgisayarda mümkündür.

c) Ana kart ID'sine Göre Dosya Şifrelemenin Oluşturulması

Bilgisayarların kimlik numarası sayılan ID numarasını anahtar baz alarak her türlü dosyayı şifreleyen ve çözen algoritmanın kodlarının ve formunun oluşturulmasıdır. Kullanıcı şifreleme için şifresiz dosya ve anahtar girer, şifre çözme için ise şifreli dosya ve anahtar girer.

Şifreleme ve şifre çözme yalnızca aynı bilgisayarda mümkündür.

d) Değişken Güvenlikli Dosya Şifrelemenin Oluşturulması

Kullanıcının seçimi doğrultusunda 1, 2, 3 ya da 4 anahtarlı bir şekilde dosya şifreleyen ve çözen algoritmanın kodlarının ve formunun oluşturulmasıdır. Kullanıcı şifreleme için anahtar seçim kısmından 1'den 4'e kadar sayıda anahtar seçer ve şifresiz dosyayla anahtar(lar)ı girer, şifre çözme işlemi için ise şifreli dosyayı ve anahtar(lar)ı girer. Unutulmamalıdır ki, birden fazla anahtar kullanılan işlemlerde anahtarların yeri önemlidir.

e) Analog Sinyallerle Şifrelemenin Oluşturulması

Arduino'dan aldığı analog sinyalleri anahtar baz alarak dosya ve metin şifreleyen ve çözen algoritmanın kodlarının ve formunun oluşturulmasıdır. Kullanıcı analog sinyal işleme biriminin bağlı olduğu seri porta bağlanır ve analog sinyallerin sayılsal hallerinin bilgisyara USB kablo ile iletilmesini sağlar. Daha sonra metin ya da dosya şifreleme kısımlarından ilgili işlemleri gerçekleştirir. Analog sinyalin (örnek olarak uygulamada potansiyometre kullanılmıştır) değişmesi şifrelemenin sonucunu da değiştirecektir. Analog sinyaller map fonksiyonu ile 0, 1023 aralığından 0,100000 aralığına kadar yükseltilebilmekte, bu da daha kompleks bir şifreleme imkanı sunmaktadır.

f) Yüz tanıma ve PIN Kodu İle Giriş Formunun Oluşturulması

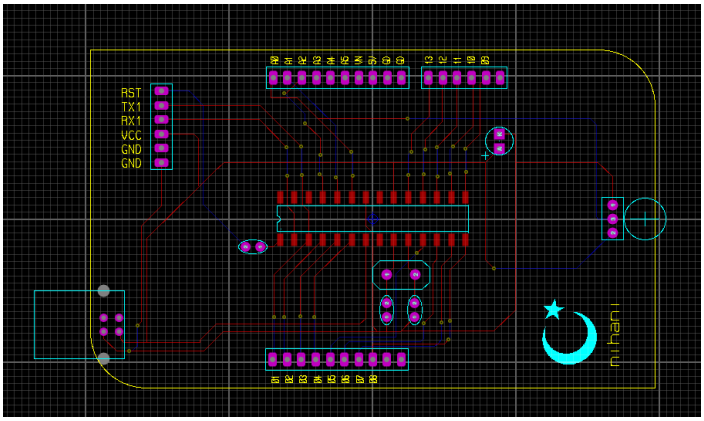
Uygulamaya giriş esnasında güvenliği sağlayan sistemlerin kodlarının ve formunun oluşuturulmasıdır. Kullanıcı kendi isteği doğrultusunda yüz tanıma ile ya da 4 haneli pin kodu ile (*örnek olarak 6687 alınmıştır*) giriş yapabilir. Yüz tanıma sistemine yeni yüzler eklemek de mümkündür.

g) Açılış ve Hakkında Formlarının Oluşturulması

Kullanıcının algoritma seçimi yapması için kullanılan açılış formu ve uygulama hakkında bigi verilen hakkında formunun kodlarının ve formunun oluşturulmasıdır.

3.2) Analog Sinyal İşleyici Aygıt

Atmel ATMEGA328P mikro işlemcisi bulunduran devrede değişken değer elde etmek için potansiyometre (3 bacaklı) kullanılmıştır. Arduino'nun avantajı olarak kolay bir şekilde bilgisayar ile haberleşilip, veri bilgisayara aktarılmıştır. Eğer istenirse analog sinyalleri değiştirmek için potansiyometreden başka çözümler de üretilebilir. Projede bu sistemi kullanabilmek için bir PCB çizilmiştir;



Şekil 6. Projede kullanılan devre kartının Proteus çizimi

3.3) Uygulamada Oluşturulan ve Kullanılan Fonksiyonlar

- **Base64Decode(string base64EncodedData)**

Daha önceden Base64 ile şifrelenmiş veriyi base64EncodedData parametresine tanımlayarak, Base64 ile şifreli metinden string tipinde çıktı veren fonksiyondur. Simetrik Base64 şifrelemede, Ana kart ID'sine göre metin şifrelemede, değişken güvenlikli dosya şifrelemede ve analog sinyallerle dosya ve metin şifrelemede kullanılmıştır. Kodu şu şekildedir;

```
public static string Base64Decode(string base64EncodedData)
{
var base64EncodedBytes = System.Convert.FromBase64String(base64EncodedData);
return System.Text.Encoding.UTF8.GetString(base64EncodedBytes);
}
```

- **Base64Encode(string plainText)**

String tipindeki metni plainText parametresine tanımlayarak düz metinden byte tipine Base64 ile şifreli çıktı veren fonksiyondur. Simetrik Base64 şifre çözmede, Ana kart ID'sine göre metin şifre çözmede, değişken güvenlikli dosya şifre çözmede ve analog sinyallerle dosya ve metin şifre çözmede kullanılmıştır. Kodu şu şekildedir;

```
public static string Base64Encode(string plainText)
{
var plainTextBytes = System.Text.Encoding.UTF8.GetBytes(plainText);
return System.Convert.ToBase64String(plainTextBytes);
}
```

- **Reverse(string s)**

"s" parametresine atanan string tipindeki veriyi simetrik olarak ters fonksiyondur. Örnek olarak, Reverse("TUBITAK") = "KATIBUT". Simetrik Base64 şifrelemede ve şifre çözmede kullanılmıştır. Kodu şu şekildedir;

```
public static string Reverse(string s)
{
char[] charArray = s.ToCharArray();
Array.Reverse(charArray);
return new string(charArray);
}
```

- **uret()**

İstenilen karakterlede ve uzunlukta rastgele keilme üretmek için kullanılan fonksiyondur. Simetrik Base64 şifrelemede ve şifre çözmede anahtar üretici olarak kullanılmıştır. Projede kullanıldığı şekilde tüm alfabe harflerinden (büyük harfler, küçük harfler) ve tüm rakamlardan, 6 karakter üretilmektedir. Kodu şu şekildedir;

```
Random rastgele = new Random();
string harfler = "abcdefghijklmnoprstuvyzABCDEFGHIJKLMNOPRSTUVYZ0123456789";
string uret = "";
for (int i = 0; i < 6; i++)
{
uret += harfler[rastgele.Next(harfler.Length)];
}
simbas_a.Text = uret; //simbas_a, formda bir elemanın adıdır.
```

- **getMotherBoardID()**

Ana kart ID'si ile şifreleme yapılan algoritmalarda kullanılan ve ana kartın ID numarasını regedit üzerinden çekmeye yarayan fonksiyondur. Kodu şu şekildedir;

```
public String getMotherBoardID()
{
String serial = "";
try
{
ManagementObjectSearcher mos = new ManagementObjectSearcher("SELECT SerialNumber FROM
Win32_BaseBoard");
ManagementObjectCollection moc = mos.Get();
foreach (ManagementObject mo in moc)
{
serial = mo["SerialNumber"].ToString();
}
return serial;
}
catch (Exception) { return serial; }

}
```

- **sif(string metin, string anahtar)**

Şifresiz metnin metin, anahtarın da anahtar parametresine atandığı ve simetrik şifrelemenin gerçekleştiği fonksiyondur. Tüm algoritmalarda kullanılmıştır. Kodu şu şekildedir;

```
public static string sif(string metin, string anahtar)
{
string sifreli_metin = "";
int j = 0;
for (int k = 0; k <= metin.Length - 1; k++)
{sifreli_metin = sifreli_metin + Convert.ToChar((Convert.ToInt32(metin[k]) +
Convert.ToInt32(anahtar[j])) % 255);
j = j + 1;
if (j == anahtar.Length){j = 0;}
}
return sifreli_metin;

}
```

- **coz(string metin, string anahtar)**

Şifreli metnin metin, anahtarın da anahtar parametresine atandığı ve simetrik şifre çözmenin gerçekleştiği fonksiyondur. Tüm algoritmalarda kullanılmıştır. Kodu şu şekildedir;

```
public static string coz(string metin, string anahtar)
{
string cozulu_metin = "";
int kod = 0;
int j = 0;
for (int k = 0; k <= metin.Length - 1; k++)
{
kod = Convert.ToInt32(metin[k]) - Convert.ToInt32(anahtar[j]);
if (kod <= 0)
kod = kod + 255;
else
kod = kod % 255;
cozulu_metin = cozulu_metin + Convert.ToChar(kod);
j = j + 1;
if (j == anahtar.Length)
j = 0;}return cozulu_metin;}
```

- **EncryptFile(string sourceFile, string targetFile, string key)**

Şifresiz dosya yolunun sourceFile, şifreli dosya yolunun targetFile ve anahtarın da key parametresine atanıp AES şifreleme yapılmasını sağlayan fonksiyondur. AES şifreleme kullanılan tüm algoritmalarda kullanılmıştır. Kodu şu şekildedir;

```
private void EncryptFile(string sourceFile, string targetFile, string key){
AesManaged AES = new AesManaged();
using (MD5CryptoServiceProvider MD5 = new MD5CryptoServiceProvider()){
AES.KeySize = MD5.HashSize; // they are 128 bit compatible
AES.BlockSize = MD5.HashSize; // they are 128 bit compatible
AES.IV = MD5.ComputeHash(ASCIIEncoding.ASCII.GetBytes(getMotherBoardID()));
AES.Key = MD5.ComputeHash(ASCIIEncoding.ASCII.GetBytes(key));
}
using (FileStream reader = new FileStream(sourceFile, FileMode.Open,
FileAccess.Read)){
using (FileStream writer = new FileStream(targetFile, FileMode.OpenOrCreate,
FileAccess.Write)){
using (CryptoStream cs = new CryptoStream(writer, AES.CreateEncryptor(),
CryptoStreamMode.Write)){
int bufferSize = 4096;
byte[] buffer = new byte[bufferSize];
int bytesRead;
do{
bytesRead = reader.Read(buffer, 0, bufferSize);
if (bytesRead != 0){
cs.Write(buffer, 0, bytesRead);
}
}
while (bytesRead != 0);
cs.FlushFinalBlock();}}}}
```

- **DecryptFile(string sourceFile, string targetFile, string key)**

Şifreli dosya yolunun sourceFile, şifresiz dosya yolunun targetFile ve anahtarın da key parametresine atanıp AES şifre çözme yapılmasını sağlayan fonksiyondur. AES şifre çözme kullanılan tüm algoritmalarda kullanılmıştır. Kodu şu şekildedir;

```
private void DecryptFile(string sourceFile, string targetFile, string key){
AesManaged AES = new AesManaged();
using (MD5CryptoServiceProvider MD5 = new MD5CryptoServiceProvider()){
AES.KeySize = MD5.HashSize; // they are 128 bit compatible
AES.BlockSize = MD5.HashSize; // they are 128 bit compatible
AES.IV = MD5.ComputeHash(ASCIIEncoding.ASCII.GetBytes(Reverse(key)));
AES.Key = MD5.ComputeHash(ASCIIEncoding.ASCII.GetBytes(key));}
using (FileStream reader = new FileStream(sourceFile,FileMode.Open,FileAccess.Read)){
using (FileStream writer = new FileStream(targetFile, FileMode.OpenOrCreate,
FileAccess.Write)){
using (CryptoStream cs = new CryptoStream(reader, AES.CreateDecryptor(),
CryptoStreamMode.Read)){
int bufferSize = 4096;
byte[] buffer = new byte[bufferSize];
int bytesRead;
do{
bytesRead = cs.Read(buffer, 0, bufferSize);
if (bytesRead != 0){
writer.Write(buffer, 0, bytesRead);}}
while (bytesRead != 0);}}}}
```

3.4) Proje Kodlarının Açıklanması

3.4.1) Windows Form Kodları

Proje kapsamında 8 adet Windows form oluşturulmuştur. Bunlar; acilis.cs, analog.cs, degisken.cs, giris.cs, hakkinda.cs, iddosya.cs, idmetin.cs ve simbas.cs'dir. *Kodlar ek olarak da jüri huzuruna sunulacaktır.*

a) acilis Formundaki Kodlar: Uygulamanın açılış sayfasıdır.

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
```

İlk kod bloğumuz kullanılan sınıfları göstermektedir.

```
private void açilis_Load(object sender, EventArgs e)
{
    try
    {

    }
    catch
    {
        MessageBox.Show("YAZILIM AÇILIRKEN BİR HATA OLUŞTU!");
    }

}
```

acilis_Load isimli blok, form açıldığı sırada yapılacak işlemleri içerir. Bu kod bloğunda ise formun açılması sırasında bir sorun çıkar ise MessageBox ile "YAZILIM AÇILIRKEN BİR SORUN OLUŞTU!" yazılması kodlanmıştır.

```
private void simbas_Click(object sender, EventArgs e)
{
    this.Hide();
    simbas simbas = new simbas();
    simbas.Show();
}

private void iddosya_Click(object sender, EventArgs e)
{
    this.Hide();
    iddosya iddosya = new iddosya();
    iddosya.Show();
}

private void degisken_Click(object sender, EventArgs e)
{
    this.Hide();
    degisken degisken = new degisken();
    degisken.Show();
}

private void analog_Click(object sender, EventArgs e)
{
    this.Hide();
    analog analog = new analog();
    analog.Show();
}

private void simbas1_Click(object sender, EventArgs e)
{
    this.Hide();
    simbas simbas = new simbas();
    simbas.Show();
}
```

```
private void iddosya1_Click(object sender, EventArgs e)
{
    this.Hide();
    iddosya iddosya = new iddosya();
    iddosya.Show();
}

private void idmetin1_Click(object sender, EventArgs e)
{
    this.Hide();
    idmetin idmetin = new idmetin();
    idmetin.Show();
}

private void degisken1_Click(object sender, EventArgs e)
{
    this.Hide();
    degisken degisken = new degisken();
    degisken.Show();
}

private void analog1_Click(object sender, EventArgs e)
{
    this.Hide();
    analog analog = new analog();
    analog.Show();
}

private void hakkinda_Click(object sender, EventArgs e)
{
    this.Hide();
    hakkinda hakkinda = new hakkinda();
    hakkinda.Show();
}
```

Bu kod bloklarında ise eğer simbas isimli butona basılırsa simbas isimli formun açılmasını, iddosya isimli butona basılırsa iddosya isimli formun açılmasını, degisken isimli butona basılırsa degisken isimli formun açılmasını, analog isimli butona basılırsa analog isimli formun açılmasını, idmetin isimli butona basılırsa idmetin isimli formun açılmasını ve hakkinda isimli butona basılırsa hakkinda isimli formun açılmasını sağlayan kodlar vardır.

```
private void çIKIŞToolStripMenuItem_Click(object sender, EventArgs e)
{
    Application.Exit();
}
```

Bu formun son kod bloğu ise, çıkış isimli tool strip butonuna tıklanırsa uygulamadan çıkılmasını sağlayan kodları içermektedir.

b) analog Formundaki Kodlar: Analog sinyallerle şifreleme kısmıdır.

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.IO;
using System.Management;
using System.Security.Cryptography;
```

Bu ilk kod bloğu da formda kullanılan sınıfları göstermektedir. Önceki formdan farklı olarak System.IO sınıfı, System.Management ve System.Security.Cryptography sınıflarını içerir. Bu sınıflar sırasıyla giriş/çıkış işlemleri, dosya yönetimi işlemleri ve AES şifreleme işlemleri için kullanılır.

```
public static string Base64Encode(string plainText)
{
    var plainTextBytes = System.Text.Encoding.UTF8.GetBytes(plainText);
    return System.Convert.ToBase64String(plainTextBytes);
}
```

Bir metni Base64 ile şifrelemek için kullanılan Base64Encode fonksiyonunun bulunduğu kodlardır. plainText değişkeninde string tipinde şifrelenecek veriler girilir. Önce plainText değilşeninin byte değeri hesaplanır, ardından Base64'e çevrilir. Daha sonra return deyimi ile bu şifrelenmiş veri yazdırılacağı yere yazdırılır.

```
public static string Base64Decode(string base64EncodedData)
{
    var base64EncodedBytes = System.Convert.FromBase64String(base64EncodedData);
    return System.Text.Encoding.UTF8.GetString(base64EncodedBytes);
}
```

Base64 ile şifrelenmiş metnin şifresini çözmek için kullanılan Base64Decode fonksiyonunun bulunduğu kodlardır. base64EncodedData önce System.Convert.FromBase64String fonksiyonu ile base64EncodedBytes isimli var veri tipindeki değişkene aktarılır. Daha sonra return deyimi ile bu çözülmüş veri yazdırılacağı yere yazdırılır.

```
public static string Reverse(string s)
{
    char[] charArray = s.ToCharArray();
    Array.Reverse(charArray);
    return new string(charArray);
}
```

Reverse fonksiyonu ile girilen metnin tersini alan kodların bulunduğu kod bloğudur. String tipindeki s değişkeni array yani karakter dizisine dönüştürülür ve ters çevrilir. Return ile de yazdırılır.

```
private void EncryptFile(string sourceFile, string targetFile, string key, string key2)
{
    AesManaged AES = new AesManaged();
    using (MD5CryptoServiceProvider MD5 = new MD5CryptoServiceProvider())
    {
        AES.KeySize = MD5.HashSize;
        AES.BlockSize = MD5.HashSize;
        AES.IV = MD5.ComputeHash(ASCIIEncoding.ASCII.GetBytes(key2));
        AES.Key = MD5.ComputeHash(ASCIIEncoding.ASCII.GetBytes(key));
    }
    using (FileStream reader = new FileStream(sourceFile, FileMode.Open, FileAccess.Read))
    {
        using (FileStream writer = new FileStream(targetFile, FileMode.OpenOrCreate, FileAccess.Write))
        {
            using (CryptoStream cs = new CryptoStream(writer, AES.CreateEncryptor(), CryptoStreamMode.Write))
            {
                int bufferSize = 4096;
                byte[] buffer = new byte[bufferSize];
                int bytesRead;
                do
                {
                    bytesRead = reader.Read(buffer, 0, bufferSize);
                    if (bytesRead != 0)
                    {
                        cs.Write(buffer, 0, bytesRead);
                    }
                }
                while (bytesRead != 0);
                cs.FlushFinalBlock();
            }
        }
    }
}
```

AES şifrelemenin gerçekleşmesini sağlayan EncryptFile fonksiyonunun bulunduğu kodlardır. AES şifreleme, System.Security.Cryptography sınıfı ile gerçekleştirilir. Program için EncrpytFile fonksiyonuna 4 adet parametre girilmiştir; sourceFile: kaynak dosya, targetFile: hedef dosya, key: 1. anahtar, key2: 2. anahtar.

```
private void DecryptFile(string sourceFile, string targetFile, string key, string key2)
{
    AesManaged AES = new AesManaged();
    using (MD5CryptoServiceProvider MD5 = new MD5CryptoServiceProvider())
    {
        AES.KeySize = MD5.HashSize;
        AES.BlockSize = MD5.HashSize;
        AES.IV = MD5.ComputeHash(ASCIIEncoding.ASCII.GetBytes(key2));
        AES.Key = MD5.ComputeHash(ASCIIEncoding.ASCII.GetBytes(key));
    }
    using (FileStream reader = new FileStream(sourceFile, FileMode.Open, FileAccess.Read))
    {
        using (FileStream writer = new FileStream(targetFile, FileMode.OpenOrCreate, FileAccess.Write))
        {
            using (CryptoStream cs = new CryptoStream(reader, AES.CreateDecryptor(), CryptoStreamMode.Read))
            {
                int bufferSize = 4096;
                byte[] buffer = new byte[bufferSize];
                int bytesRead;
                do
                {
                    bytesRead = cs.Read(buffer, 0, bufferSize);
                    if (bytesRead != 0)
                    {
                        writer.Write(buffer, 0, bytesRead);
                    }
                }
                while (bytesRead != 0);
            }
        }
    }
}
```

Bir üstteki kodlarda olduğu gibi, bu kodlar da AES şifre çözmenin gerçekleşmesini sağlayan DecryptFile fonksiyonunun bulunduğu kodlardır. AES şifre çözme de şifreleme gibi System.Security.Cryptography sınıfı ile gerçekleştirilir. Program için decryptFile fonksiyonuna 4 adet parametre girilmiştir; sourceFile: kaynak dosya, targetFile: hedef dosya, key: 1. anahtar, key2: 2. anahtar.

```
public analog()
{
    InitializeComponent();
}
```

analog isimli form açılışında gerekli komponentlerin (butonlar, labeller vs.) yüklenmesini sağlayan kodlardır.

```
private void anamenüToolStripMenuItem_Click(object sender, EventArgs e)
{
    this.Hide();
    acilis acilis = new acilis();
    acilis.Show();
}
```

anamenü isimli tool strip butonuna tıklanınca açık olan analog isimli formun kapanmasını, acilis ismindeki formun açılmasını sağlayan kodlardır.

```
private void analog_Load(object sender, EventArgs e)
{
    CenterToScreen();
    baglantidurumu.ForeColor = Color.Red;
    for (int i = 0; i < System.IO.Ports.SerialPort.GetPortNames().Length; i++)
    {
        comboBox1.Items.Add(System.IO.Ports.SerialPort.GetPortNames()[i]);
    }
}
```

analog isimli form yüklenince yapılacak işlemleri içeren kodlardır. CenterToScreen fonksiyonu formun tam ortaya sabitlenmesini sağlar. baglantidurumu isimli labelin rengi kırmızı yapılır. comboBox1'e de seri portların isimleri yazdırılır.

```
private void timer1_Tick(object sender, EventArgs e)
{
    try
    {
        serialPort1.Write("1");
        int receiveddata = Convert.ToInt16(serialPort1.ReadExisting());
        string receiveddata_b64 = Base64Encode(receiveddata.ToString());
        //receiveddata = ((receiveddata * 5000) / 1023) / 10;
        label4.Text = receiveddata.ToString();
        label11.Text = Base64Encode(receiveddata_b64);
        System.Threading.Thread.Sleep(100);
    }
    catch (Exception ex) { }
}
```

timer1isimli zamanlayıcı belirtilen sıklıklarda gerekli işlemleri yapması için programlanmıştır. Bu işlemler; serialPort1 isimli porta 1 yazdırmak, integer tipindeki receiveddata değişkenine seri porttan okunan verileri atamak, bu verileri base64'e çevirmek, label11'e seri porttan alınan son veriyi girmek ve 100 ms'de bir sistemi tekrarlatmak. Ayrıca try-cath komutlarıyla da hatalar çözümlendirilmeye çalışılmıştır.

```
private void button1_Click(object sender, EventArgs e)
{
    if (comboBox1.Text != "")
    {
        timer1.Start();
        try
        {
            serialPort1.PortName = comboBox1.Text;
            if (!serialPort1.IsOpen)
                serialPort1.Open();
            baglantidurumu.ForeColor = Color.Green;
            baglantidurumu.Text = "BAĞLI";
        }
        catch
        {
            MessageBox.Show("ZATEN BAĞLI!");
        }
    }
    else
    {
        MessageBox.Show("PORT SEÇİNİZ");
    }
}
```

Eğer button1 isimli butona, ki bu seri porta bağlanma butonudur, tıklanınca yapılacak işlemleri içeren kodlardır. Eğer seri port ismi giriliyse timer1 zamanlayıcısını başlatır, bağlanılan seri portu comboboxa yazdır. Eğer seri porta bağlı değilse bağlanır, baglantidurumu isimli labelin rengini yeşil yapıp "BAĞLI" yazdırır. Eğer bağlıysa "ZATEN BAĞLI!" şeklinde bir uyarı verir. Eğer port seçili değilse de "PORT SEÇİNİZ" şeklinde uyarı verir.

```
private void button2_Click(object sender, EventArgs e)
{
    try
    {
        string metin = Base64Encode(richTextBox2.Text);
        string anahtar = Base64Encode(label11.Text);
        string sifreli_metin = "";
        int j = 0;
        for (int k = 0; k <= metin.Length - 1; k++)
        {
            sifreli_metin = sifreli_metin + Convert.ToChar((Convert.ToInt32(metin[k]) + Convert.ToInt32(anahtar[j])) % 255);
            j = j + 1;
            if (j == anahtar.Length)
                j = 0;
        }
        richTextBox3.Text = Base64Encode(sifreli_metin);
        MessageBox.Show("ŞİFRELEME TAMAMLANDI");
    }
    catch
    {
        MessageBox.Show("BİR HATA OLUŞTU");
    }
}
```

button2 isimli butona, ki bu metin şifreleme butonudur, tıklanınca yapılacak işlemleri içeren kodlardır. richTextBox2'deki metni yani şifrelencek metni önce base64 ile şifreleyip metin isimli string tipindeki değişkene atar, seri porttan alınan veriyi base64 ile şifreleyip anahtar isimli değişkene atar ve metin değişkeni ile anahtar değişkenini simetrik olarak şifreler. Daha sonra oluşan çıktıyı da base64 ile şifreleyip "ŞİFRELEME TAMAMLANDI" şeklinde uyarı verir. Eğer bir hata oluşursa "BİR HATA OLUŞTU" şeklinde hata verir.

```
private void button3_Click(object sender, EventArgs e)
{
    try
    {
        if (richTextBox3.Text != "")
        {
            string sifreli_metin = Base64Decode(richTextBox3.Text);
            string anahtar = Base64Encode(label11.Text);
            string metin = "";
            int kod = 0;
            int j = 0;
            for (int k = 0; k <= sifreli_metin.Length - 1; k++)
            {
                kod = Convert.ToInt32(sifreli_metin[k]) - Convert.ToInt32(anahtar[j]);
                if (kod <= 0)
                    kod = kod + 255;
                else
                    kod = kod % 255;
                metin = metin + Convert.ToChar(kod);
                j = j + 1;
                if (j == anahtar.Length)
                    j = 0;
            }
            richTextBox2.Text = Base64Decode(metin);
            MessageBox.Show("ŞİFRE ÇÖZME TAMAMLANDI");
        }
    }
    catch
    {
        MessageBox.Show("BİR HATA OLUŞTU");
    }
}
```

Bu kodlarda ise, metin şifre çözme butonu olan button3'e tıklayınca gerçekleşecek işlemler verilmiştir. Metin şifrelemenin tam tersi olarak önce base64 şifre çözme, daha sonra simetrik şifre çözme ve en son da yine base64 şifre çözme işlemi gerçekleşir. Bir hata oluşmazsa "ŞİFRE ÇÖZME TAMAMLANDI", aksi halde "BİR HATA OLUŞTU" şeklinde uyarı verilir.

```
private void iddosya_ac1_Click(object sender, EventArgs e)
{
    OpenFileDialog dosyaac1 = new OpenFileDialog();
    if (dosyaac1.ShowDialog() == DialogResult.OK)
    {
        analog_e1.Text = dosyaac1.FileName;
    }
}
```

Bu kodlar, şifrelenecek dosyayı açan iddosya_ac1 isimli butona basılınca gerçekleşecek olan işlemleri içerek kodlardır. Butona basınca dosyaac1 isminde bir OpenFileDialog nesnesi oluşturulur ve diyalog başarılı sonuçlanırsa seçilen dosyanın yolu analog_e1 isimli metin kutusuna yazdırılır.

```
private void iddosya_kaydet1_Click(object sender, EventArgs e)
{
    SaveFileDialog dosyakaydet1 = new SaveFileDialog();
    if (dosyakaydet1.ShowDialog() == DialogResult.OK)
    {
        analog_y1.Text = dosyakaydet1.FileName;
    }
}
```

Bu kodlarda ise iddosya_kaydet1 isimli butona tıklanınca gerçekleşecek işlemler yazılmıştır. İlk önce dosyakaydet1 isimli bir SaveFileDialog nesnesi oluşturulur ve diyalog başarılı sonuçlanırsa seçilen dosyanın yolu analog_y1 isimli metin kutusuna yazdırılır.

```
private void button4_Click(object sender, EventArgs e)
{
    try
    {
        EncryptFile(analog_e1.Text, analog_y1.Text, Base64Encode(label11.Text), label11.Text);
        MessageBox.Show("ŞİFRELEME TAMAMLANDI");
    }
    catch
    {
        MessageBox.Show("ŞİFRELEME TAMAMLANAMADI");
    }
}
```

Dosya şifreleme butonu olan button4'e tıklanınca gerçekleşecek olaylar bu kodlarda verilmiştir. analog_e1.Text olan eski dosya yolu, analog_y1.Text olan yeni dosya yolu, alınan analog verinin base64'e çevrilmiş halleri AES şifrelemeye tabi tutulur. Bir hata olmadıysa "ŞİFRELEME TAMAMLANDI" şeklinde uyarı verir. Eğer bir hata oluşursa "BİR HATA OLUŞTU" şeklinde uyarı verir.

```
private void button7_Click(object sender, EventArgs e)
{
    OpenFileDialog dosyaac1 = new OpenFileDialog();
    if (dosyaac1.ShowDialog() == DialogResult.OK)
    {
        analog_e2.Text = dosyaac1.FileName;
    }
}
```

Bu kodlar, şifresi çözülecek dosyayı açan button7 isimli butona basılınca gerçekleşecek olan işlemleri içerek kodlardır. Butona basınca dosyaac1 isminde bir OpenFileDialog nesnesi oluşturulur ve diyalog başarılı sonuçlanırsa seçilen dosyanın yolu analog_e2 isimli metin kutusuna yazdırılır.

```
private void button6_Click(object sender, EventArgs e)
{
    SaveFileDialog dosyakaydet1 = new SaveFileDialog();
    if (dosyakaydet1.ShowDialog() == DialogResult.OK)
    {
        analog_y2.Text = dosyakaydet1.FileName;
    }
}
```

Bu kodlarda ise button6 isimli butona tıklanınca gerçekleşecek işlemler yazılmıştır. İlk önce dosyakaydet1 isimli bir SaveFileDialog nesnesi oluşturulur ve diyalog başarılı sonuçlanırsa seçilen dosyanın yolu analog_y2 isimli metin kutusuna yazdırılır.

```
private void button5_Click(object sender, EventArgs e)
{
    try
    {
        DecryptFile(analog_e2.Text, analog_y2.Text, Base64Encode(label11.Text), label11.Text);
        MessageBox.Show("ŞİFRE ÇÖZME TAMAMLANDI");
    }
    catch
    {
        MessageBox.Show("ŞİFRE ÇÖZME TAMAMLANAMADI");
    }
}
```

Dosya şifre çözme butonu olan button5'e tıklanınca gerçekleşecek olaylar bu kodlarda verilmiştir. analog_e2.Text olan eski dosya yolu, analog_y2.Text olan yeni dosya yolu, alınan analog verinin base64'e çevrilmiş hallesi AES şifre çözmeye tabi tutulur. Bir hata olmadıysa "ŞİFRE ÇÖZME TAMAMLANDI" şeklinde uyarı verir. Eğer bir hata oluşursa "BİR HATA OLUŞTU" şeklinde uyarı verir.

c) **degisken Formundaki Kodlar:** Değişken güvenlikli dosya şifreleme kısmıdır.

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.IO;
using System.Management;
using System.Security.Cryptography;
```

Bu ilk kod bloğu formda kullanılan sınıfları göstermektedir.

```
public static string Base64Encode(string plainText)
{
    var plainTextBytes = System.Text.Encoding.UTF8.GetBytes(plainText);
    return System.Convert.ToBase64String(plainTextBytes);
}
```

Bir metni Base64 ile şifrelemek için kullanılan Base64Encode fonksiyonunun bulunduğu kodlardır. plainText değişkeninde string tipinde şifrelenecek veriler girilir. Önce plainText değilşeninin byte değeri hesaplanır, ardından Base64'e çevrilir. Daha sonra return deyimi ile bu şifrelenmiş veri yazdırılacağı yere yazdırılır.

```
public static string Base64Decode(string base64EncodedData)
{
    var base64EncodedBytes = System.Convert.FromBase64String(base64EncodedData);
    return System.Text.Encoding.UTF8.GetString(base64EncodedBytes);
}
```

Base64 ile şifrelenmiş metnin şifresini çözmek için kullanılan Base64Decode fonksiyonunun bulunduğu kodlardır. base64EncodedData önce System.Convert.FromBase64String fonksiyonu ile base64EncodedBytes isimli var veri tipindeki değişkene aktarılır. Daha sonra return deyimi ile bu çözülmüş veri yazdırılacağı yere yazdırılır.

```
public static string Reverse(string s)
{
    char[] charArray = s.ToCharArray();
    Array.Reverse(charArray);
    return new string(charArray);
}
```

Reverse fonksiyonu ile girilen metnin tersini alan kodların bulunduğu kod bloğudur. String tipindeki s değişkeni array yani karakter dizisine dönüştürülür ve ters çevrilir. Return ile de yazdırılır.

```
public static string sif(string metin, string anahtar)
{
   string sifreli_metin = "";
    int j = 0;
    for (int k = 0; k <= metin.Length - 1; k++)
    {
        sifreli_metin = sifreli_metin + Convert.ToChar((Convert.ToInt32(metin[k]) + Convert.ToInt32(anahtar[j])) % 255);
        j = j + 1;
        if (j == anahtar.Length)
            j = 0;
    }
    return sifreli_metin;
}
```

Bu kodlarda ise string tipindeki metin değişkeni ile yine string tipindeki anahtar değişkeni simetrik olarak şifrelenmiştir.

```
public static string coz(string metin, string anahtar)
{
    string cozulu_metin = "";
    int kod = 0;
    int j = 0;
    for (int k = 0; k <= metin.Length - 1; k++)
    {
        kod = Convert.ToInt32(metin[k]) - Convert.ToInt32(anahtar[j]);
        if (kod <= 0)
            kod = kod + 255;
        else
            kod = kod % 255;
        cozulu_metin = cozulu_metin + Convert.ToChar(kod);
        j = j + 1;
        if (j == anahtar.Length)
            j = 0;
    }
    return cozulu_metin;
}
```

Bu kodlar, simetrik olarak şifrelenmiş metnin metin değişkeninde yazıldığı ve anahtar ile beraber simetrik olarak çözüldüğü kodlardır.

```
private void EncryptFile(string sourceFile, string targetFile, string key, string key2)
{
    AesManaged AES = new AesManaged();
    using (MD5CryptoServiceProvider MD5 = new MD5CryptoServiceProvider())
    {
        AES.KeySize = MD5.HashSize;
        AES.BlockSize = MD5.HashSize;
        AES.IV = MD5.ComputeHash(ASCIIEncoding.ASCII.GetBytes(key2));
        AES.Key = MD5.ComputeHash(ASCIIEncoding.ASCII.GetBytes(key));
    }
    using (FileStream reader = new FileStream(sourceFile, FileMode.Open, FileAccess.Read))
    {
        using (FileStream writer = new FileStream(targetFile, FileMode.OpenOrCreate, FileAccess.Write))
        {
            using (CryptoStream cs = new CryptoStream(writer, AES.CreateEncryptor(), CryptoStreamMode.Write))
            {
                int bufferSize = 4096;
                byte[] buffer = new byte[bufferSize];
                int bytesRead;
                do
                {
                    bytesRead = reader.Read(buffer, 0, bufferSize);
                    if (bytesRead != 0)
                    {
                        cs.Write(buffer, 0, bytesRead);
                    }
                }
                while (bytesRead != 0);
                cs.FlushFinalBlock();
            }
        }
    }
}
```

AES şifrelemenin gerçekleşmesini sağlayan EncryptFile fonksiyonunun bulunduğu kodlardır. AES şifreleme, System.Security.Cryptography sınıfı ile gerçekleştirilir. Program için EncrpytFile fonksiyonuna 4 adet parametre girilmiştir; sourceFile: kaynak dosya, targetFile: hedef dosya, key: 1. anahtar, key2: 2. anahtar.

```
private void DecryptFile(string sourceFile, string targetFile, string key, string key2)
{
    AesManaged AES = new AesManaged();
    using (MD5CryptoServiceProvider MD5 = new MD5CryptoServiceProvider())
    {
        AES.KeySize = MD5.HashSize;
        AES.BlockSize = MD5.HashSize;
        AES.IV = MD5.ComputeHash(ASCIIEncoding.ASCII.GetBytes(key2));
        AES.Key = MD5.ComputeHash(ASCIIEncoding.ASCII.GetBytes(key));
    }
    using (FileStream reader = new FileStream(sourceFile, FileMode.Open, FileAccess.Read))
    {
        using (FileStream writer = new FileStream(targetFile, FileMode.OpenOrCreate, FileAccess.Write))
        {
            using (CryptoStream cs = new CryptoStream(reader, AES.CreateDecryptor(), CryptoStreamMode.Read))
            {
                int bufferSize = 4096;
                byte[] buffer = new byte[bufferSize];
                int bytesRead;
                do
                {
                    bytesRead = cs.Read(buffer, 0, bufferSize);
                    if (bytesRead != 0)
                    {
                        writer.Write(buffer, 0, bytesRead);
                    }
                }
                while (bytesRead != 0);
            }
        }
    }
}
```

Bir üstteki kodlarda olduğu gibi, bu kodlar da AES şifre çözmenin gerçekleşmesini sağlayan DecryptFile fonksiyonunun bulunduğu kodlardır. AES şifre çözme de şifreleme gibi System.Security.Cryptography sınıfı ile gerçekleştirilir. Program için decryptFile fonksiyonuna 4 adet parametre girilmiştir; sourceFile: kaynak dosya, targetFile: hedef dosya, key: 1. anahtar, key2: 2. anahtar.

```
private void anamenüToolStripMenuItem_Click(object sender, EventArgs e)
{
    this.Hide();
    acilis acilis = new acilis();
    acilis.Show();
}
```

anamenü isimli tool strip butonuna tıklayınca gerçekleşecek işlemler verilmiştir. Açık olan form kapatılacak ve acilis isimli form açılacaktır.

```
private void radioButton1_CheckedChanged(object sender, EventArgs e)
{
    a1.Enabled = true; a11.Enabled = true;
    a2.Enabled = false; a21.Enabled = false;
    a3.Enabled = false; a31.Enabled = false;
    a4.Enabled = false; a41.Enabled = false;
}
```

Bu kodlarda radioButon1 işaretlenince gerçekleşecek işlemler yazılmıştır. a1 ve a11 isimli anahtar kutuları aktif, diğerleri pasif duruma getirilecektir.

```
private void radioButton2_CheckedChanged(object sender, EventArgs e)
{
    a1.Enabled = true;a11.Enabled = true;
    a2.Enabled = true;a21.Enabled = true;
    a3.Enabled = false;a31.Enabled = false;
    a4.Enabled = false;a41.Enabled = false;
}
```

Bu kodlarda radioButon2 işaretlenince gerçekleşecek işlemler yazılmıştır. a1, a11, a2 ve a21 isimli anahtar kutuları aktif, diğerleri pasif duruma getirilecektir.

```
private void radioButton3_CheckedChanged(object sender, EventArgs e)
{
    a1.Enabled = true;
    a2.Enabled = true;
    a3.Enabled = true;
    a4.Enabled = false;
    a11.Enabled = true;
    a21.Enabled = true;
    a31.Enabled = true;
    a41.Enabled = false;
}
```

Bu kodlarda radioButon3 işaretlenince gerçekleşecek işlemler yazılmıştır. a1, a11, a2, a21, a3 ve a31 isimli anahtar kutuları aktif, diğerleri pasif duruma getirilecektir.

```
private void radioButton4_CheckedChanged(object sender, EventArgs e)
{
    a1.Enabled = true;
    a2.Enabled = true;
    a3.Enabled = true;
    a4.Enabled = true;
    a11.Enabled = true;
    a21.Enabled = true;
    a31.Enabled = true;
    a41.Enabled = true;
}
```

Bu kodlarda radioButon4 işaretlenince gerçekleşecek işlemler yazılmıştır. Tüm anahtar kutuları aktif duruma getirilecektir.

```
private void ac1_Click(object sender, EventArgs e)
{
    OpenFileDialog dosyaac = new OpenFileDialog();
    if (dosyaac.ShowDialog() == DialogResult.OK)
    {
        e1.Text = dosyaac.FileName;
    }
}
```

ac1 isimli butona tıklanınca gerçekleşecek işlemler bu kodda yazılmıştır; dosyaac adında bir OpenFileDialog nesnesi oluşturmak ve diyalogun sonucuna göre seçilen dosyayı e1 isimli metin kutusuna yazdırmak. e1 isimli metin kutusu şifrelenecek dosyanın yoludur.

```
private void kaydet1_Click(object sender, EventArgs e)
{
    SaveFileDialog dosyakaydet = new SaveFileDialog();
    if(dosyakaydet.ShowDialog() == DialogResult.OK)
    {
        y1.Text = dosyakaydet.FileName;
    }
}
```

Bu kodlarda da kaydet1 isimli butona tıklanınca dosyakaydet adında bir SaveFileDialog nesnesi oluşturup diyalogun sonucunu y1 isimli metin kutusuna yazacak komutlar verilmiştir. y1 isimli metin kutusu şifreli dosyanın yeni konumudur.

```
private void ac2_Click(object sender, EventArgs e)
{
    OpenFileDialog dosyaac = new OpenFileDialog();
    if (dosyaac.ShowDialog() == DialogResult.OK)
    {
        e2.Text = dosyaac.FileName;
    }
}
```

ac2 isimli butona tıklanınca gerçekleşecek işlemler bu kodda yazılmıştır; dosyaac adında bir OpenFileDialog nesnesi oluşturmak ve diyalogun sonucuna göre seçilen dosyayı e1 isimli metin kutusuna yazdırmak. e2 isimli metin kutusu şifresi çözülecek dosyanın yoludur.

```
private void kaydet2_Click(object sender, EventArgs e)
{
    SaveFileDialog dosyakaydet = new SaveFileDialog();
    if (dosyakaydet.ShowDialog() == DialogResult.OK)
    {
        y2.Text = dosyakaydet.FileName;
    }
}
```

Bu kodlarda da kaydet2 isimli butona tıklanınca dosyakaydet adında bir SaveFileDialog nesnesi oluşturup diyalogun sonucunu y1 isimli metin kutusuna yazacak komutlar verilmiştir. y2 isimli metin kutusu şifresi çözülmüş dosyanın yeni konumudur.

```
private void sifrele_Click(object sender, EventArgs e)
{
    try
    {
        //1 anahtar:
        if (a2.Enabled == false &&
            a3.Enabled == false &&
            a4.Enabled == false &&
            a1.Enabled == true &&
            a1.Text != "" &&
            e1.Text != "" &&
            y1.Text != "")
        {
            string tekanahtar = sif(a1.Text, Reverse(a1.Text));
            EncryptFile(e1.Text, y1.Text, Base64Encode(tekanahtar));
            MessageBox.Show("ŞİFRELEME TAMAMLANDI");
        }
```

Bu kodlardan itibaren şifreleme işlemleri başlamaktadır. Bu kod bloklarında sifrele isimli butona basılınca yapılacak işlemler kodlanmıştır. Eğer yalnızca 1. anahtar kutusu aktifse, string tipinde, tekanahtar isminde bir değişken oluşturulur ve bu değişkene, girilen anahtar ve anahtarın tersinin simetrik olarak şifrelenmesi sonucu oluşan ortak anahtar atanır. AES şifreleme ile de e1 yani şifresiz dosya, y2 yolunda yani yeni dosya yolunda, ortak anahtarın Base64'e çevrilmiş haliyle şifrelenir. Herhangi bir sorun oluşmaz ise "ŞİFRELEME TAMAMLANDI" şeklinde uyarı verilir. (Devam ediyor)

```
//2 anahtar:
else if (a2.Enabled == true &&
    a1.Enabled == true &&
    a3.Enabled == false &&
    a4.Enabled == false &&
    a1.Text != "" &&
    a2.Text !="")
{
    string tekanahtar = sif(a1.Text, a2.Text);
    EncryptFile(e1.Text, y1.Text, Base64Encode(tekanahtar));
    MessageBox.Show("ŞİFRELEME TAMAMLANDI");
}
```

Eğer 1. ve 2. anahtar kutusu aktifse, string tipinde, tekanahtar isminde bir değişken oluşturulur ve bu değişkene, girilen 1. anahtar ve 2. anahtarın simetrik olarak şifrelenmesi sonucu oluşan ortak anahtar atanır. AES şifreleme ile de e1 yani şifresiz dosya, y2 yolunda yani yeni dosya yolunda, ortak anahtarın Base64'e çevrilmiş haliyle şifrelenir. Herhangi bir sorun oluşmaz ise "ŞİFRELEME TAMAMLANDI" şeklinde uyarı verilir.

```
//3 anahtar:
else if (a2.Enabled == true &&
    a3.Enabled == true &&
    a4.Enabled == false &&
    a1.Enabled == true &&
    a1.Text != "" &&
    a2.Text != "" &&
    a3.Text != "" &&
    e1.Text != "" &&
    y1.Text != "")
{
    string tekanahtar = sif(sif(a1.Text, a3.Text), a2.Text);
    EncryptFile(e1.Text, y1.Text, Base64Encode(tekanahtar));
    MessageBox.Show("ŞİFRELEME TAMAMLANDI");
}
```

Eğer 1., 2. ve 3. anahtar kutusu aktifse, string tipinde, tekanahtar isminde bir değişken oluşturulur ve bu değişkene, girilen 1. anahtar ve 3. anahtarın simetrik olarak şifrelenmesi sonucu oluşan ortak anahtar ile 2. anahtarın simetrik olarak şifrelenmesi sonucu oluşan 2. ortak anahtar atanır. AES şifreleme ile de e1 yani şifresiz dosya, y2 yolunda yani yeni dosya yolunda, 2. oluşan ortak anahtarın Base64'e çevrilmiş haliyle şifrelenir. Herhangi bir sorun oluşmaz ise "ŞİFRELEME TAMAMLANDI" şeklinde uyarı verilir.

```
        //4 anahtar:
        else if (a2.Enabled == true &&
            a3.Enabled == true &&
            a4.Enabled == true &&
            a1.Enabled == true &&
            a1.Text != "" &&
            a2.Text != "" &&
            a3.Text != "" &&
            a4.Text != "" &&
            e1.Text != "" &&
            y1.Text != "")
        {
            string tekanahtar = sif(sif(a1.Text, a3.Text), sif(a2.Text, a4.Text));
            EncryptFile(e1.Text, y1.Text, Base64Encode(tekanahtar));
            MessageBox.Show("ŞİFRELEME TAMAMLANDI");
        }
        else
        {
            MessageBox.Show("TÜM ALANLARI DOLDURUN");
        }

    }
    catch
    {
        MessageBox.Show("BİR HATA OLUŞTU");
    }
}
```

Eğer 1., 2., 3. ve 4. anahtar kutusu aktifse, string tipinde, tekanahtar isminde bir değişken oluşturulur ve bu değişkene, girilen 1. anahtar ve 3. anahtarın simetrik olarak şifrelenmesi sonucu oluşan ortak anahtar ($ortakAnahtar_{1ve3}$ olsun) ile 2. anahtar ve 4. anahtarın simetrik olarak şifrelenmesi sonucu oluşan ortak anahtar ($ortakAnahtar_{2ve4}$ olsun) 'ın simetrik olarak oluşturduğu 3. bir ortak anahtar ($ortakAnahtar_{13\text{-}24}$ olsun) atanır. AES şifreleme ile de e1 yani şifresiz dosya, y2 yolunda yani yeni dosya yolunda, 3. oluşan ortak anahtarın ($ortakAnahtar_{13\text{-}24)}$ Base64'e çevrilmiş haliyle şifrelenir. Herhangi bir sorun oluşmaz ise "ŞİFRELEME TAMAMLANDI" şeklinde uyarı verilir. Tüm bu işlemler bittikten sonra tekrar kontrol edilir ve bir hata oluşursa "BİR HATA OLUŞTU" şeklinde son bir uyarı verilir. Aynı işlemler şifre çözmede de geçerlidir.

d) giris Formundaki Kodlar: Uygulamanın giriş kısmıdır.

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using Emgu.CV;
using Emgu.CV.Structure;
using Emgu.CV.CvEnum;
using System.IO;
```

Bu kodlarda da diğer formlarda olduğu gibi kullanılan sınıflar gösterilmiştir. Ancak farklı olarak Emgu.CV sınıfını da barındırmaktadır. EmguCV'nin yüz tanıma için gerekli olduğu öncekl bölümlerde belirtilmişti.

```
Image<Bgr, Byte> currentFrame;
Capture grabber;
HaarCascade face;
HaarCascade eye;
MCvFont font = new MCvFont(FONT.CV_FONT_HERSHEY_TRIPLEX, 0.5d, 0.5d);
Image<Gray, byte> result, TrainedFace = null;
Image<Gray, byte> gray = null;
List<Image<Gray, byte>> trainingImages = new List<Image<Gray, byte>>();
List<string> labels = new List<string>();
List<string> NamePersons = new List<string>();
int ContTrain, NumLabels, t;
```

Bu kodlarda ise EmguCV için gerekli olan birkaç ayar yapılmıştır; grabber isminde bir Capture nesnesi oluşturmak, McvFont nesnesinin parametrelerini girmek ve tanınmış yüzleri bir dizide tutmak gibi.

```
private void ekle_Click(object sender, EventArgs e)
{
    try
    {
        ContTrain = ContTrain + 1;
        gray = grabber.QueryGrayFrame().Resize(320, 240, Emgu.CV.CvEnum.INTER.CV_INTER_CUBIC);
        MCvAvgComp[][] facesDetected = gray.DetectHaarCascade(
        face,
        1.2,
        10,
        Emgu.CV.CvEnum.HAAR_DETECTION_TYPE.DO_CANNY_PRUNING,
        new Size(20, 20));
        foreach (MCvAvgComp f in facesDetected[0])
        {
            TrainedFace = currentFrame.Copy(f.rect).Convert<Gray, byte>();
            break;
        }
        TrainedFace = result.Resize(100, 100, Emgu.CV.CvEnum.INTER.CV_INTER_CUBIC);
        trainingImages.Add(TrainedFace);
        labels.Add(textBox1.Text);
        imageBox1.Image = TrainedFace;
        File.WriteAllText(Application.StartupPath + "/TrainedFaces/TrainedLabels.txt", trainingImages.ToArray().Length.ToString() + "%");
        for (int i = 1; i < trainingImages.ToArray().Length + 1; i++)
        {
            trainingImages.ToArray()[i - 1].Save(Application.StartupPath + "/TrainedFaces/face" + i + ".bmp");
            File.AppendAllText(Application.StartupPath + "/TrainedFaces/TrainedLabels.txt", labels.ToArray()[i - 1] + "%");
        }
        MessageBox.Show(textBox1.Text + " kişisinin yüzü kaydedildi.", "Kaydedildi.", MessageBoxButtons.OK, MessageBoxIcon.Information);
    }
    catch
    {

    }
}
```

Bu kodlarda yeni yüz eklemek için kullanılan ekle butonuna tıklanırsa gerçekleşecek işlemler verilmiştir. Eğer kaydetme başarıyla sonuçlanırsa "TrainedFaces" isimli klasörün içindeki "TrainedLabels" isimli metin dosyasının içine kişi adı ve ID'si fotoğrafının .bmp formatındaki haliyle beraber kaydedilir, kaydetmenin tamamlandığına dair bir de uyarı verilir.

```
private void anamenüToolStripMenuItem_Click(object sender, EventArgs e)
{
    this.Hide();
    acilis acilis = new acilis();
    acilis.Show();
}
```

Bu kodlarda ise anamenü isimli tool strip menü butonuna tıklanırsa açık formun kapatılmasını ve acilis isimli formun açılmasını sağlayan kodlar bulunur. Bu özellik tamamen proje yarışmasına özgü eklenmiştir. Yüz tanımayı ve PIN kodunu es geçmek için eklenen butona basıldığı an şifreleme ekranı açılır.

```
private void gÖSTERToolStripMenuItem_Click(object sender, EventArgs e)
{
    this.Size = new System.Drawing.Size(854, 576);
    this.Location = new Point((Screen.PrimaryScreen.WorkingArea.Width - this.Width) / 2,
    (Screen.PrimaryScreen.WorkingArea.Height - this.Height) / 2);
    groupBox4.Size = new System.Drawing.Size(820, 282);
    groupBox3.Size = new System.Drawing.Size(814, 210);
}
```

Bu kodlar, göster isimli tool strip menü butonuna tıklandığı zaman gerçekleşecek kodlardır. Bu buton, yeni yüz eklemek için kullanılmaktadır. Butona tıklandığı zaman formun boyutu System.Drawing.Size fonksiyonu ile 854 pixele 576 pixel olarak büyütülür, aynı şekilde yüz tanıma bölmesiyle PIN kodu bölmesi de belirli bir oranda büyütülür.

```
private void gİZLEToolStripMenuItem_Click(object sender, EventArgs e)
{
    this.Size = new System.Drawing.Size(466, 576);
    this.Location = new Point((Screen.PrimaryScreen.WorkingArea.Width - this.Width) / 2,
                  (Screen.PrimaryScreen.WorkingArea.Height - this.Height) / 2);
    groupBox4.Size = new System.Drawing.Size(430, 282);
    groupBox3.Size = new System.Drawing.Size(430, 210);
}
```

Bu kodlar da, üstteki kodların tam tersi olarak gizle butonuna tıklanırsa gerçekleşecek işlemlerdir. Form 466 pixele 576 pixele küçültülür, yüz tanıma kısmı ve PIN kodu kısmı da aynı şekilde küçültülür. Ayrıca bu ölçüler formun açılıştaki ölçüleridir.

```
private void pin_TextChanged(object sender, EventArgs e)
{
    if(pin.Text == "6687")
    {
        this.Hide();
        acilis acilis = new acilis();
        acilis.Show();
    }
}
```

Eğer girilen PIN kodu 6687 ise (ki bu değer değiştirilebilir) açılış formunun açılmasını sağlayan kodlardır.

```
private void sil_Click(object sender, EventArgs e)
{
    pin.Text = "";
}
```

PIN kodu ile girişteki PIN kodu kutusunu temizlemek için kullanılan kodlardır.

```
public giris()
{
    InitializeComponent();
    face = new HaarCascade("haarcascade_frontalface_default.xml");
    try
    {
        string Labelsinfo = File.ReadAllText(Application.StartupPath + "/KayitliYuzler/KayitliIsimler.txt");
        string[] Labels = Labelsinfo.Split('%');
        NumLabels = Convert.ToInt16(Labels[0]);
        ContTrain = NumLabels;
        string LoadFaces;
        for (int tf = 1; tf < NumLabels + 1; tf++)
        {
            LoadFaces = "face" + tf + ".bmp";
            trainingImages.Add(new Image<Gray, byte>(Application.StartupPath + "/KayitliYuzler/" + LoadFaces));
            labels.Add(Labels[tf]);
        }
    }
    catch (Exception e)
    {
    }
}
```

Bu kodlar ise giris formunun temel kodlarıdır. Öncelikle komponentler yüklenir. Daha sonra face adında bir haarcascade nesnesi oluşturulur. Bu nesnenin kaynağı kırmızı çerçeve ile verilmiştir. Diğer kodlar isi EmguCv kütüphaneleri kullanarak yüz algılamaya yarar.

```
void FrameGrabber(object sender, EventArgs e)
{
    label3.Text = "0";
    NamePersons.Add("");
    currentFrame = grabber.QueryFrame().Resize(320, 240, Emgu.CV.CvEnum.INTER.CV_INTER_CUBIC);
    gray = currentFrame.Convert<Gray, Byte>();
    MCvAvgComp[][] facesDetected = gray.DetectHaarCascade(
  face,
  1.2,
  10,
  Emgu.CV.CvEnum.HAAR_DETECTION_TYPE.DO_CANNY_PRUNING,
  new Size(20, 20));
    foreach (MCvAvgComp f in facesDetected[0])
    {
        t = t + 1;
        result = currentFrame.Copy(f.rect).Convert<Gray, byte>().Resize(100, 100, Emgu.CV.CvEnum.INTER.CV_INTER_CUB
        currentFrame.Draw(f.rect, new Bgr(Color.Red), 2);
        if (trainingImages.ToArray().Length != 0)
        {
            MCvTermCriteria termCrit = new MCvTermCriteria(ContTrain, 0.001);
            EigenObjectRecognizer recognizer = new EigenObjectRecognizer(
               trainingImages.ToArray(),
               labels.ToArray(),
               3000,
               ref termCrit);
            name = recognizer.Recognize(result);
            currentFrame.Draw(name, ref font, new Point(f.rect.X - 2, f.rect.Y - 2), new Bgr(Color.LightGreen));
        }
        NamePersons[t - 1] = name;
        NamePersons.Add("");
        label3.Text = facesDetected[0].Length.ToString();
    }
    t = 0;
```

Bu kodlarda framegrabber deyimi ile algılanan yüz tespit edilip kırmızı bir çerçeve içine alınmıştır. Eğer algılanan yüz KayitliYuzler klasörü içindeyse yapılacak eylemler belirtilmiştir.

```
for (int nnn = 0; nnn < facesDetected[0].Length; nnn++)
{
    names = names + NamePersons[nnn] + ", ";
    for (int i = 0; i < 1; i++)
    {
        if (durum == false)
        {
            if (name == null)
            {

            }
            else
            {
                DialogResult dialogResult = MessageBox.Show("Şifreleme Ekranı Açılsın Mı?",
                    names + "Kişisinin Yüzü Tanındı", MessageBoxButtons.YesNo);
                if (dialogResult == DialogResult.Yes)
                {
                    if (Application.OpenForms["acilis"] == null)
                    {
                        this.Hide();
                        acilis acilis = new acilis();
                        acilis.Show();
                        acilis.name = names;
                    }
                    else
                    {

                    }
                }
            }
        }
        else
        {

        }
    }
            for (int nnn = 0; nnn < facesDetected[0].Length; nnn++)
            {
                names = names + NamePersons[nnn] + ", ";
            }
            imageBoxFrameGrabber.Image = currentFrame;
            label4.Text = names;
            names = "";
            NamePersons.Clear();
        }

        private void yuztani_Load(object sender, EventArgs e)
        {
            this.Size = new System.Drawing.Size(466, 576);
            this.Location = new Point((Screen.PrimaryScreen.WorkingArea.Width - this.Width) / 2,
                        (Screen.PrimaryScreen.WorkingArea.Height - this.Height) / 2);
            try
            {
                grabber = new Capture();
                grabber.QueryFrame();
                Application.Idle += new EventHandler(FrameGrabber);
            }
            catch
            {
                MessageBox.Show("KAMERA BAŞLATILIRKEN HATA OLUŞTU!");
            }
        }
    }
}
```

Üstte de belirtildiği gibi eğer algılanan yüz daha önceden kayıtlıysa formun açılıp açılmayacağını sorar. Kullanıcı açılmasını isterse form açılır. Aynı zamanda açılış formunda da şu anda uygulamayı kimin kullandığının bilgisi bulunmaktadır. Bu da güvenliği bir üst seviyelere çekmektedir. En sondaki kodlar ise sırasıyla, tespit edilen yüzün forma yazdırılmasını, daha sonra da kameranın başlatılmasını sağlayan kodlardır. Eğer kamerayı başlatırken hata olduysa, bu durum mesaj kutusuyla kullanıcıya bildirilir.

e) hakkinda Formundaki Kodlar: Uygulama hakkında bilgi verilen kısımdır.

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace nihani
{
    public partial class hakkinda : Form
    {
        public hakkinda()
        {
            InitializeComponent();
        }

        private void tamam_Click(object sender, EventArgs e)
        {
            this.Close();
            acilis acilis = new acilis();
            acilis.Show();
        }

        private void hakkinda_Load(object sender, EventArgs e)
        {

        }

        private void hakkinda_Closing(object sender, System.ComponentModel.CancelEventArgs e)
        {
            acilis acilis = new acilis();
            acilis.Show();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            this.Close();
            acilis acilis = new acilis();
            acilis.Show();
        }
    }
}
```

En az kod bu formda vardır. Bu form uygulama hakkında bilgi verir. Açılış sayfasından ulaşılabilir, çıkmak için ÇIK butonuna basılırsa acilis formunu açar.

f) iddosya Formundaki Kodlar: Anakart ID'siyle dosya şifreleyip çözmek için kullanılan kısımdır. *Tekrar edilen kısımlar eklenmemiştir.*

```
public static string anahtarlisifre(string text1, string text2)
{
    string encry = "";
    int j = 0;
    for (int k = 0; k <= text1.Length - 1; k++)
    {
        encry = encry + Convert.ToChar((Convert.ToInt32(text1[k]) + Convert.ToInt32(text2[j])) % 255);
        j = j + 1;
        if (j == text2.Length)
            j = 0;
    }
    return encry;
}
```

Bu kodlar anahtarliSifrele fonksiyonunun kodlarıdır. Text1 ve text2 parametreleri input edilir, daha sonra simetrik bir şekilde şifrelenir. Encry değişkeni ile de yazdırılır.

```
private void button2_Click(object sender, EventArgs e)
{
    try
    {
        if(iddosya_e1.Text != "")
        {
            EncryptFile(iddosya_e1.Text, iddosya_y1.Text, anahtarlisifre(getMotherBoardID(),
                anahtar1.Text));
            MessageBox.Show("ŞİFRELEME TAMAMLANDI");
        }
        else
        {
            MessageBox.Show("TÜM ALANLARI DOLDURUN");
        }
    }

    catch
    {
        MessageBox.Show("BİR HATA OLUŞTU");
    }
}
```

Button2'ye tıklanılınca çalışacak kodlar bu kodlardır. Eğer iddosya_e1 boş değilse şifreleme yapılır ve bu kullanıcıya bildirilir. Eğer hata oluşursa bu durum da uyarı şeklinde verilir.

```
private void button3_Click(object sender, EventArgs e)
{
    try{
        if (iddosya_e2.Text != ""){
            DecryptFile(iddosya_e2.Text, iddosya_y2.Text, anahtarlisifre(getMotherBoardID(),
                anahtar2.Text));
            MessageBox.Show("ŞİFRE ÇÖZME TAMAMLANDI");
        }
        else{
            MessageBox.Show("TÜM ALANLARI DOLDURUN");
        }}
    catch
    {
        MessageBox.Show("BİR HATA OLUŞTU");
    }
}
```

Button3'e tıklanılınca çalışacak kodlar da bu kodlardır. Eğer iddosya_e2 boş değilse şifreleme yapılır ve bu kullanıcıya bildirilir. Eğer hata oluşursa bu durum da uyarı şeklinde verilir.

```
private void iddosya_ac1_Click(object sender, EventArgs e)
{
    OpenFileDialog dosyaac1 = new OpenFileDialog();
    if (dosyaac1.ShowDialog() == DialogResult.OK)
    {
        iddosya_e1.Text = dosyaac1.FileName;
    }
}

private void iddosya_ac2_Click(object sender, EventArgs e)
{
    OpenFileDialog dosyaac2 = new OpenFileDialog();
    if (dosyaac2.ShowDialog() == DialogResult.OK)
    {
        iddosya_e2.Text = dosyaac2.FileName;
    }
}
```

```
private void iddosya_kaydet1_Click(object sender, EventArgs e)
{
    SaveFileDialog dosyakaydet1 = new SaveFileDialog();
    if (dosyakaydet1.ShowDialog() == DialogResult.OK)
    {
        iddosya_y1.Text = dosyakaydet1.FileName;
    }
}

private void iddosya_kaydet2_Click(object sender, EventArgs e)
{
    SaveFileDialog dosyakaydet2 = new SaveFileDialog();
    if (dosyakaydet2.ShowDialog() == DialogResult.OK)
    {
        iddosya_y2.Text = dosyakaydet2.FileName;
    }
}
```

Bu kodlar OpenFileDialog ve SaveFileDialog ile dosya açıp kaydetmek için kullanılan kodladır. Kullanıcının seçtiği dosyayı şifreler, yine kullanıcının seçtiği şifreli dosyanın şifresini çözerler.

g) idmetin Formundaki Kodlar: Anakart ID'siyle metin şifrelemek ve çözmek için kullanılan kısımdır. *Yine üsteki gibi, sayfa sayısını artırmamak amacıyla yinelenen kodlar gizlenmiştir.*

```
private void sifrele2_Click(object sender, EventArgs e)
{
    try
    {
        if (sifresizMetin2.Text != "") {
            string metin = Base64Encode(sifresizMetin2.Text);
            string anahtar = anahtarlisifre(getMotherBoardID(), anahtar1.Text);
            string sifreli_metin = "";
            int j = 0;
            for (int k = 0; k <= metin.Length - 1; k++)
            {
                sifreli_metin = sifreli_metin + Convert.ToChar((Convert.ToInt32(metin[k]) +
                    Convert.ToInt32(anahtar[j])) % 255);
                j = j + 1;
                if (j == anahtar.Length)
                    j = 0;
            }
            sifreliMetin2.Text = Base64Encode(sifreli_metin);
            MessageBox.Show("ŞİFRELEME TAMAMLANDI");
        }
    }
    catch
    {
        MessageBox.Show("BİR HATA OLUŞTU");
    }
}
```

Bu kodlar sifrele2 butonuna tıklanınca şifreleme işleminin başlamasını sağlayan kodlardır. Önce sifresizmetin2 textboxunun boş olup olmadığı kontrol edilir. Eğer boş değilse metin sırasıyla basee64, simetrik ve tekrar base64 olmak üzere şifrelenir.

```
private void sifrecoz2_Click(object sender, EventArgs e)
{
    try
    {
        if (sifreliMetin2.Text != "")
        {
            string sifreli_metin = Base64Decode(sifreliMetin2.Text);
            string anahtar = anahtarlisifre(getMotherBoardID(), anahtar2.Text);
            string metin = "";
            int kod = 0;
            int j = 0;
            for (int k = 0; k <= sifreli_metin.Length - 1; k++)
            {
                kod = Convert.ToInt32(sifreli_metin[k]) - Convert.ToInt32(anahtar[j]);
                if (kod <= 0)
                    kod = kod + 255;
                else
                    kod = kod % 255;
                metin = metin + Convert.ToChar(kod);
                j = j + 1;
                if (j == anahtar.Length)
                    j = 0;
            }
            sifresizMetin2.Text = Base64Decode(metin);
            MessageBox.Show("ŞİFRE ÇÖZME TAMAMLANDI");
        }
    }
    catch
    {
        MessageBox.Show("BİR HATA OLUŞTU");
    }
}
```

Bu kodlar da üstteki kodların tersi olarak şifreli metni base64, simetrik ve tekrar base64 yöntemiyle çözer.

h) simbas Formundaki Kodlar: Metni hem simetrik hem de base64 ile birden fazla şifreleyen ve çözen kısımdır.

```
private void uret1_Click(object sender, EventArgs e)
{
    Random rastgele = new Random();
    string harfler = "abcdefghijklmnoprstuvyzABCDEFGHIJKLMNOPRSTUVYZ0123456789";
    string uret = "";
    for (int i = 0; i < 6; i++)
    {
        uret += harfler[rastgele.Next(harfler.Length)];
    }
    simbas_a.Text = uret;
}
```

Üret butonuna tıklanınca 5 karakterli ve "abcdefghijklmnoprstuvyz", "ABCDEFGHIJKLMNOPRSTUVYZ" ve "0123456789" karakterlerinden herhangi beşinden oluşan bir anahtar üretmeye yarayan kodlardır.

```
private void şifrele1_Click(object sender, EventArgs e)
{
    try
    {
        if(simbas_a.Text != "")
        {
            if(simbas_0.Text != "")
            {
                string base64 = Base64Encode(simbas_0.Text);
                string metin = base64;
                string anahtar = this.simbas_a.Text;
                string sifreli_metin1 = "";
                int j = 0;
                for (int k = 0; k <= metin.Length - 1; k++)
                {
                    sifreli_metin1 = sifreli_metin1 + Convert.ToChar((Convert.ToInt32(metin[k]) +
                        Convert.ToInt32(anahtar[j])) % 255);
                    j = j + 1;
                    if (j == anahtar.Length)
                        j = 0;
                }
                string base64_2 = Base64Encode(sifreli_metin1);
                string ters_1 = Reverse(base64_2);
                simbas_1.Text = ters_1;
                MessageBox.Show("ŞİFRELEME BAŞARILI");
            }
            else
            {
                MessageBox.Show("ŞİFRESİZ METNİ GİRİNİZ");
            }
        }
        else
        {
            MessageBox.Show("ANAHTARI GİRİNİZ");
        }

    }
    catch
    {
        MessageBox.Show("BİR HATA OLUŞTU");
    }
}
```

Bu kodlarda ilk önce anahtarın olup olmadığı daha sonra şifresiz metnin olup olmadığı kontrol edildikten sonra şifresiz metin ilk önce base64 ile şifrelenir. Daha sonra anahtar değişkeni ile simetrik olarak şifrelenir. Oluşan metin bir kez daha base64 ile şifrelendikten sonra ters çevrilip şifreli metin kutusuna yazdırılır. Eğer anahtar ya da şifresiz metin yoksa veya bir hata oluştuysa hata mesajının verilmesi sağlanır.

i) Arduino Kodları: Analog işleyici aygıtın kodlarıdır.

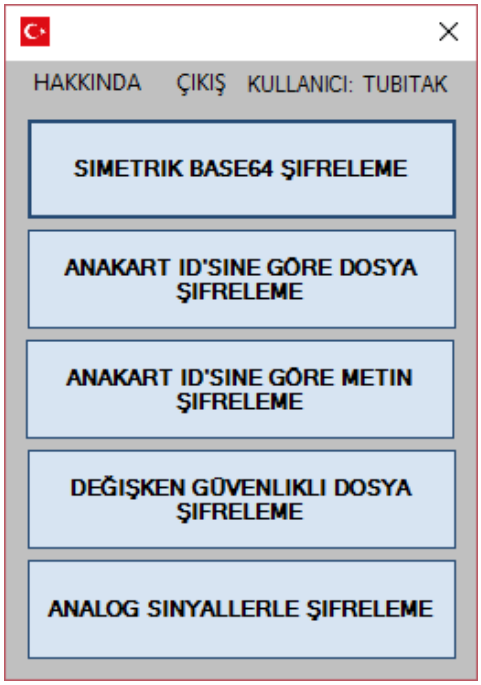
```
char t; //t değişkeni chat tipinde yazıldı
int veri = 0; //veri değişkeni int tipinde yazıldı, değeri 0'a eşitlendi.
void setup() //1 kere çalışacak kodlar
{
Serial.begin(9600); //Seri iletişim 9600 RaundBate'de başlatıldı.
}

void loop() //Sonsuz kere çalışacak kodlar
{
if(Serial.available()>0) //Eğer seri iletişim kullanılabilirse;
t=Serial.read(); //t değişkenine seri okumanın değerini ata
if(t=='1') //eğer t 1 olursa;
{
veri = analogRead(A0); //veri değişkenine analog 0 portunun değerini ata
veri = map(veri, 0,1023,0,50000); //veri değişkeninin aralığını 0-50000 yap
Serial.println(veri); //Seri ekranda veri değişkenini yazdır.
delay(100); //Bu işlemleri 1/10 saniyede (100ms) tekrarla.
  }
}
```

3.5) Proje Ekran Görüntüleri

a) Giriş Formunun Ekran Görüntüsü

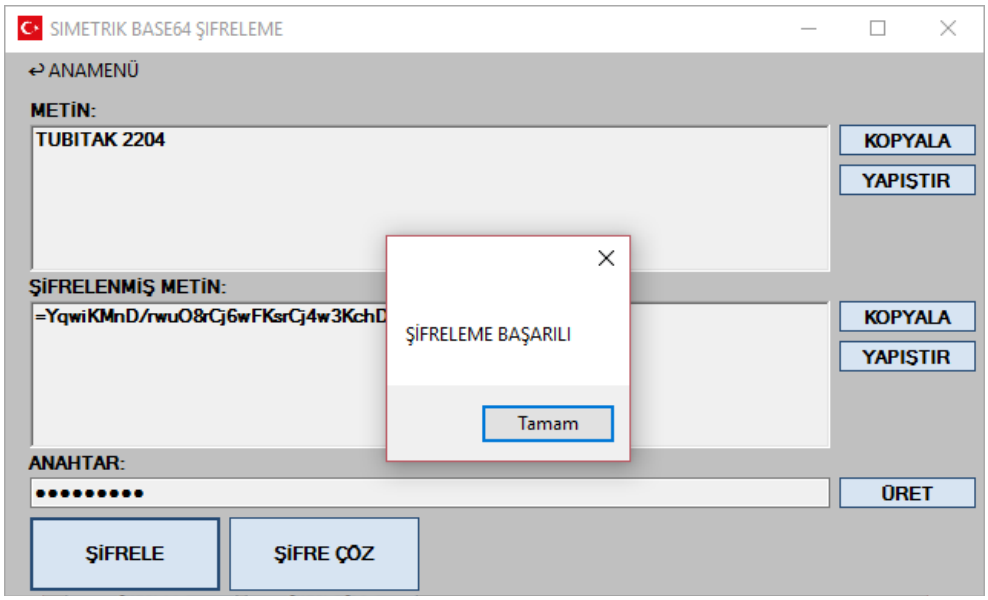
Giriş formunun ekran görüntüsü. Üstte yüz tanıma sistemi, altta PIN kodu sistemi görülmektedir. Kırmızı ile işaretli buton, sunum için demo olarak koyulmuştur.

b) Açılış Formunun Ekran Görüntüsü



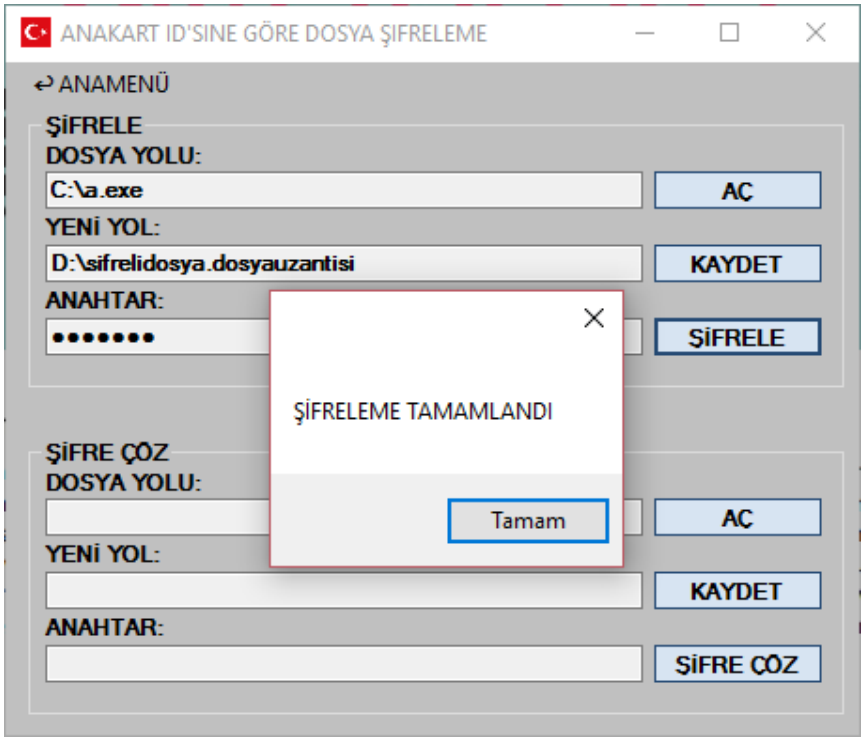
Açılış Formunun Ekran Görüntüsü. Üst menüde Hakkında ve Çıkış butonları ile Kullanıcı bilgisi bulunmaktadır.Bu bilgi yüz tanıma ile giriş yapıldıysa aktiftir. Kullanıcı bu formdan beş algoritmadan birini seçer ve şifreleme ya da şifre çözme işlemine geçer.

c) Simetrik Base64 Şifreleme Formunun Ekran Görüntüsü



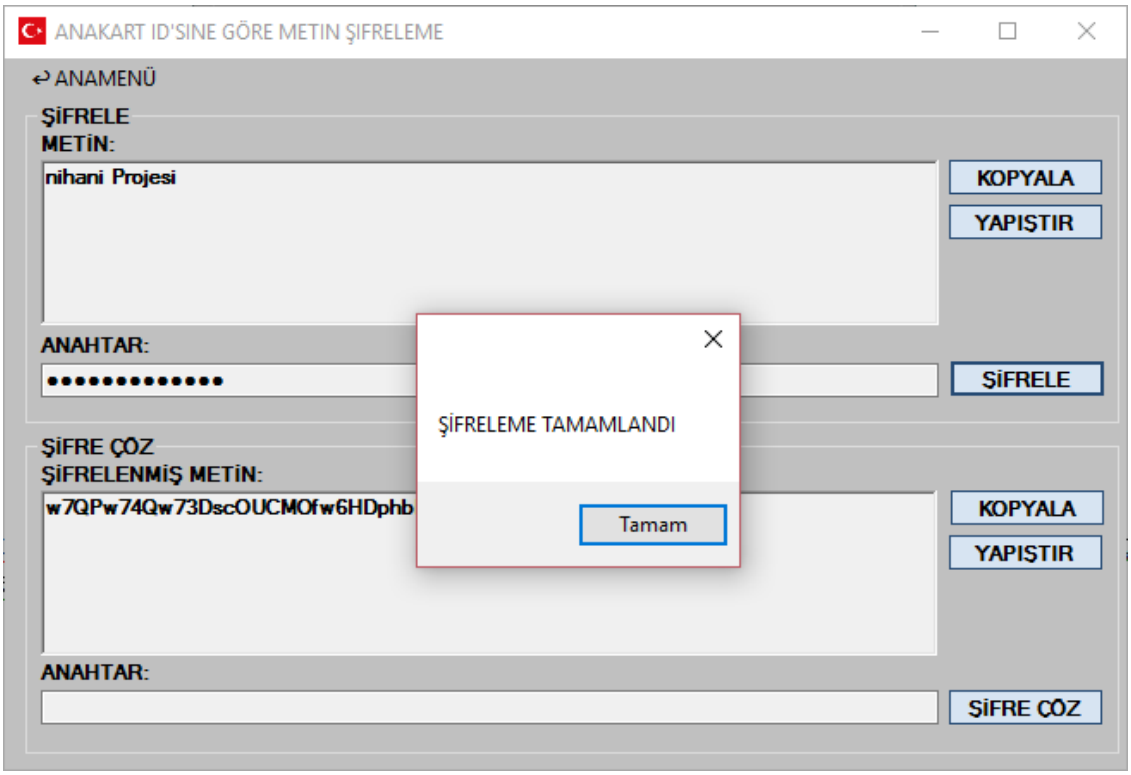
Simetrik Base64 Şifreleme Formunun Ekran Görüntüsü. Şifresiz metin ve şifreli metin kutuları bulunmaktadır. Bu metinlerin kopyalanıp yapıştırılmaları için KOPYALA ve YAPIŞTIR butonlar, anahtar üretilmesi için ÜRET butonu bulunmaktadır. Üst menüden açılış formuna geri dönülebilmektedir.

d) Anakart ID'sine Göre Dosya Şifreleme Formunun Ekran Görüntüsü

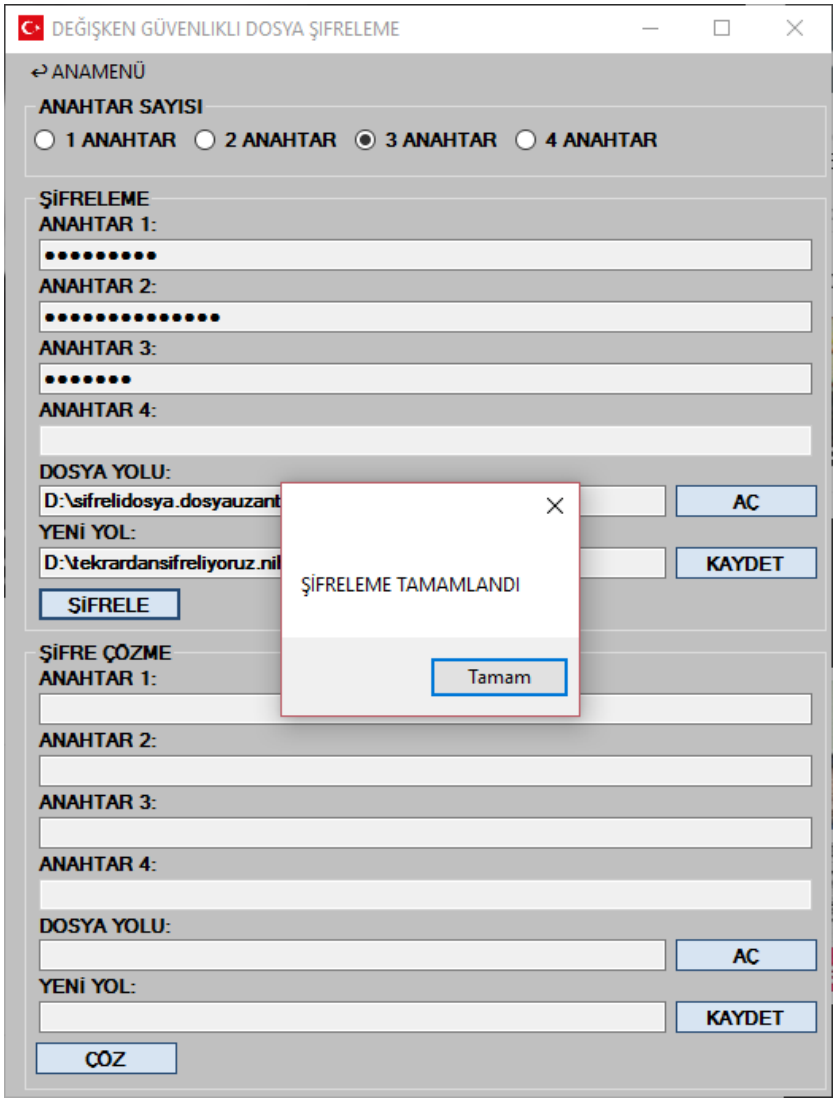


Anakart ID'sine Göre Dosya Şifreleme Formunun Ekran Görüntüsü. Dosya yolu seçmek için , şifrelemek ve şifre çözmek için butonlar bulunur.

e) Anakart ID'sine Göre Metin Şifreleme Formunun Ekran Görüntüsü

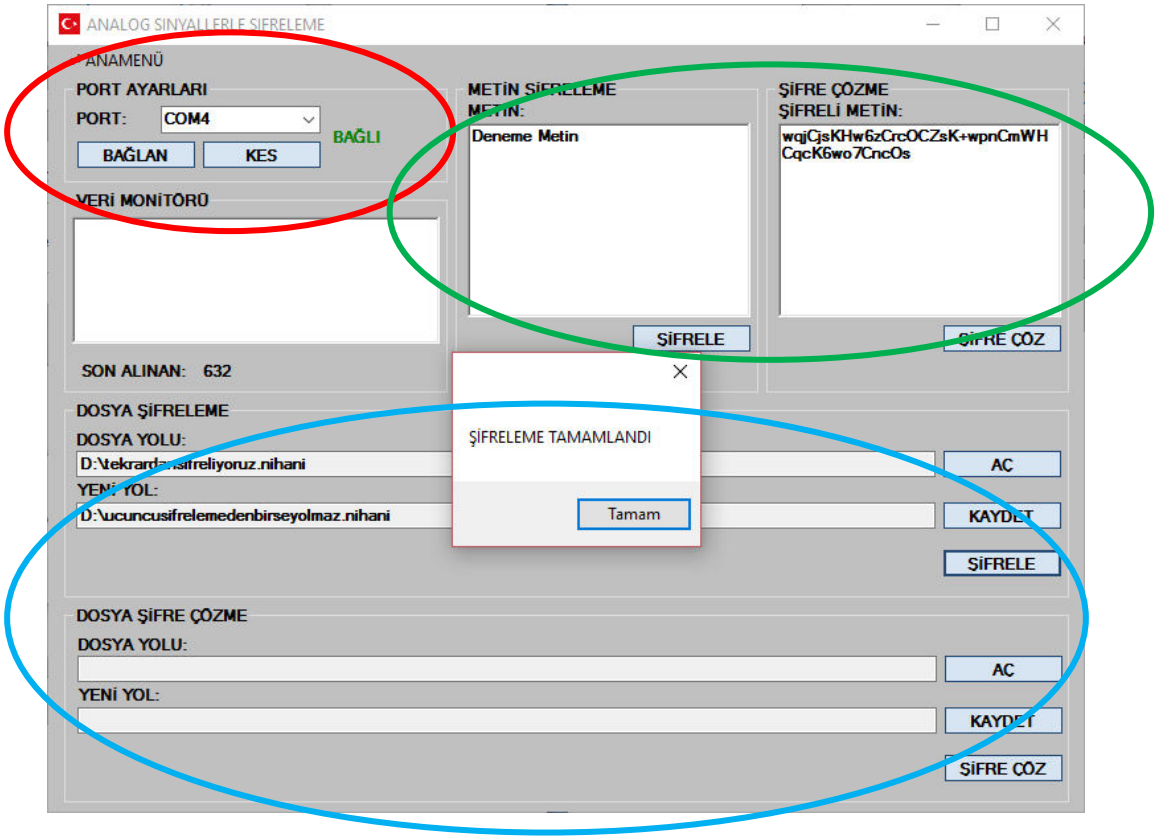


f) Değişken Güvenlikli Dosya Şifreleme Formunun Ekran Görüntüsü



g) Hakkında Formunun Ekran Görüntüsü

h) Analog Sinyallerle Şifreleme Formunun Ekran Görüntüsü



Kırmızı ile gösterilen kısmından Port ayarı yapılmaktadır. Kullanıcı portu seçer ve bağlan butonuna basar. Eğer bağlıysa BAĞLI yazısı yeşil olur. Yeşil ile gösterilen yerde ise metin şifrelemesi yapılır. Algoritmasından daha önce bahsetmiştik. Son olarak, mavi ile gösterilen yerde ise dosya şifrelemesi yapılmaktadır. Bu algoritmadan da daha önce bahsedilmişti.

4) Sonuçlar ve Tartışma

nihani projesi sonucunda dosya ve metin şifreleme/şifre çözme konusunda alternatif çözümler ürettik. Türkiye'nin de kullanabilmesi için üretilen algoritmalar tek bir uygulama altında toplandı. Böylelikle bilgi güvenliğini büyük ölçüde sağlamış olduk.

Uygulamanın ilk demosunda uygulamanın açıldığı sırada şifreleme ekranına direkt geçiş yapması güvenliği olumsuz etkiliyordu bu nedenle yüz tanıma sistemi eklendi. Daha sonra yüz tanımanın kullanım dışı olduğu durumlar için de bir yöntem geliştirilmesi gerektiğinin farkında vardık. Bu neden 4-digit (4 haneli) PIN kodu sistemini de projenin girişine ekledik.

Daha sonraları uygulama kapsamında bazı algoritmalarda exception (özel durumlar) geliştiğini fark ettik. Bu nedenle sıkı bir testing çalışmasından sonra gerekli önlemler alındı.

İlk başta analog sinyal işleyici olarak Arduino kartı kullanılırken daha sonra Proteusta çizilen devre kullanılmaya başladı.

Bu projenin diğer benzeri projelerden en önemli farkı Türkiye adına geliştirilmiş olmasıdır. Ayrıca henüz literatürde kullanılmayan teknikler de geliştirilmiştir. Bu nedenle geliştirilen algoritmaların ilk kullanım yerleri de bu uygulamadır.

5) Öneriler

- Bu projeye parmak izi ile giriş ve parmak izi ile şifreleme/şifre çözme de eklemeyi düşünüyorum.
- Eğer bir algoritma geliştirilecekse/modifiye edilecekse önceden planlamasını yapmayı ihmal etmeyiniz. Bu işinizi kolaylaştıracaktır.
- Geleceğin Python gibi dillerde olduğunu göz önünde bulundurarak projeye başlayın.

Kaynakça

- https://www.arduino.cc/en/Tutorial/AnalogInput
- https://www.arduino.cc/reference/en/language/functions/analog-io/analogread/
- https://msdn.microsoft.com/en-us/library/system.security.cryptography(v=vs.110).aspx
- https://msdn.microsoft.com/en-us/library/system.security.cryptography.aes(v=vs.110).aspx
- https://www.eetimes.com/document.asp?doc_id=1275908
- http://bilgisayarkavramlari.sadievrenseker.com/2009/06/03/aes-ve-rijndeal-sifreleme/
- https://en.wikipedia.org/wiki/Base64
- https://docs.microsoft.com/tr-tr/dotnet/csharp/programming-guide/
- https://www.pluralsight.com/browse/software.../c-sharp
- https://stackoverflow.com/questions/273452/using-aes-encryption-in-c-sharp