ELGAMAL ŞİFRELEME ALGORİTMASINI KULLANAN GÜVENLİ BİR E-POSTA UYGULAMASI: MD MESSAGE CONTROLLER

Mustafa DÜLGERLER[1] **M. Nusret SARISAKAL**[2]

[1,2]*İstanbul Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliği Bölümü*
34850, Avcılar, İstanbul

[1] e-posta: mdulgerler@hotmail.com [2]e-posta: nsarisakal@istanbul.edu.tr

Anahtar Kelimeler: *Güvenlik, Şifreleme, ElGamal Algoritması, Hash Fonksiyonu, E-mail*

ÖZET

Bu çalışmada, elektronik posta ile bilgilerin güvenli bir şekilde alıcıya nasıl gönderileceği konusu incelenmiştir. Bu uygulamada veriler karşı tarafa gönderilirken ElGamal şifreleme algoritması kullanılmıştır. Uygulama programında kullanıcı kimliği doğrulaması ise özgün bir hash fonksiyonu ile sağlanmaktadır. Bu uygulama programı windows ortamında çalışan Md Message Controller adında bir ara yüzden oluşmaktadır.

1. GİRİŞ

Günümüz teknolojileri sürekli değişmekte ve gelişmektedir. Tüm bu teknolojik gelişmeler beraberinde bir çok sorunu da beraberinde getirmektedir. Internet uygulamaları günümüzde oldukça yaygınlaşmıştır. Bu uygulamaların en büyük sorunu ise güvenliktir. Hızla gelişen teknolojide güvenlik ile ilgili yeni mekanizmalar ve yöntemler geliştirilmeye çalışılmaktadır. Yazılım geliştiriciler, sürekli olarak yeni yöntemler denemek zorunda ve başarılı oldukları taktirde, bunları uygulamalarına yansıtmaktalar. Herkesin istediği, hızlı ve güvenli iletişim, 1970'lerde ilk kez Amerikan Savunma Bakanlığınca kullanılmaya başlanılan mesajlaşma, günümüzdeki yaygın adıyla e-mail oldukça yaygın olarak kullanılmaktadır. Günümüzde şirket içi yazışmalardan, uluslararası bir çok görüşmelere kadar bir çok işlem bu yolla yapılmaktadır. Zira gönderilen mesajın, her iki tarafta da yazılı bir kopyanın olması, dolayısıyla belgelenebilirliği çok önemli bir özelliktir. Bunun yanı sıra ucuz ve diğer hizmetlere göre çok daha hızlı, etkin olması da onu öne çıkaran diğer özellikleridir. Bu kadar üstün özellikleri ve kullanıldığı çevrelerin çokluğu ile popüler olan bu uygulamalar, doğal olarak kötü niyetli kişilerinde ilgisini fazlasıyla çekmektedir. Onlara karşı alınabilecek ilk önlem, sağlam bir şifreleme yapısı kullanmaktır. Günümüzde bilinen bir çok şifreleme algoritması mevcuttur ancak bunları aynı zamanda bu kötü niyetli kişiler de bilmekteler, yapılması gereken ise bu algoritmalardan yeni algoritmalar veya yeni yöntemler türetmektir. Bu yeni yöntem veya algoritmaları, gelişmiş programlama dilleri ile kodlayarak kullanıcı uyumlu yazılımlar geliştirmek güvenlik problemlerini bir miktar çözecektir.

Bu güvenlik algoritmalarını tasarlarken, kriptografi biliminden yararlanılmaktadır. Kriptografi, çeşitli şifreleme, anahtarlama ve çözme algoritmaları sunmaktadır. Bu çalışmada, veriler şifrelenirken, ElGamal Şifreleme algoritması kullanıldı. Programın çeşitli bölümlerinde gereken kimlik doğrulama işlemi ise özgün bir SHA (Secure Hashing Algoritm) algoritması ile sağlandı.

Öncelikle Kriptoloji literatüründe kullanılan temel kavramaları açıklayalım; Gönderici, bir mesajı gönderen, alıcı ise gönderilen mesajı alması beklenen kişidir. Şifreleme, bir mesajı doğrusal olmayan fonksiyonlar yardımıyla, okunduğunda gerçek anlamının çıkarılması çok zor hatta imkansız bir forma dönüştürme işlemidir. Şifrelenmiş bir metni, şifrelemek için kullanılan fonksiyonların, matematiksel ters fonksiyonları yardımıyla, ilk haline dönüştürme işlemine de ters şifreleme yada çözme adı verilir.

Göndericiden, alıcıya iletilmesi beklenen metni M ile, herhangi bir şifreleme algoritması yardımıyla şifrelenmiş metini Ş ile şifreleme fonksiyonunu da F ile ifade edecek olursak. Şifrelenmiş metnin uzunluğu genellikle, gönderilecek metnin uzunluğundan fazladır. Bu durumda, şifrelenmiş metinler, sıkıştırma algoritmaları yardımı ile uzunluk olarak küçültmeye çalışılır. Bu tanımlamalara göre şifreleme işleminin matematiksel modeli (1) deki gibi

$$F(M)=Ş \qquad (1)$$

gösterilebilir. Şifrenin çözümlenmesi işlemi ise (2), F^{-1}

fonksiyonu (F fonksiyonunun matematiksel tersini göstermektedir),

$$F^{-1}(Ş)=M \qquad (2)$$

biçiminde ifade edilebilir. Ayrıca F^{-1} fonksiyonu ters şifreleme fonksiyonu olarak da adlandırılmaktadır.

Genel olarak, her şifreleme ve şifre çözme algoritması birbirinin ters fonksiyonları olmalıdır. Sağlam bir şifreleme yapısında asla doğrusal fonksiyonlar kullanılmamalıdır. Aksi taktirde, algoritmanın kötü niyetli kişilerce çözümlenme zamanı çok kısalmış olur.

Şifreleme bilimi çok uzun yıllar boyunca sadece askeri ve diplomatik haberleşmede kullanılmaktaydı. Bu algoritmaların kullanım alanları arttıkça, standart kurallara dönüştürülmeye başlandı. Bu çalışmada yararlandığımız ElGamal Şifreleme algoritması ise, bir genel-anahtar algoritması olarak 1985'de T. ElGamal tarafından tasarlandı. Günümüze kadar, ElGamal algoritmanın kullanıldığı bir sistem üzerine hiçbir başarılı saldırı bilinmemektedir. 1994'te bağımsız birkaç araştırma grubu, genel-anahtar, sayısal imza algoritmaları tabanlı tüm farklı logaritmaların, genelleştirilmiş "meta" algoritmanın farklı sonuçları olduğu bulgularını sundular. ElGamal'ın içerdiği bu farklı logaritmalar, onun oldukça güvenli olduğunun bir göstergesidir[1,2].

Bu tip yöntemler, genel anahtardan, özel anahtarın zor bulunması prensibine dayanır. Bu zorluk, genel-özel anahtar çiftinin uzunluğuna ve ayrıca bu anahtar çiftinin ne olduğunu tahmin için yapılacak hesaplamaların güçlüğüne bağlıdır.

ElGamal şifreleme algoritmasının anahtar uzunluğu 256 bitten, rasgele seçilmiş bir bit boyutuna kadar genişletilebilir. 1024 bitten 2048 bite kadar değer alabilen bir anahtar uzunluğu gelecek 20 yıl için güvenli olarak düşünülmektedir. Bu tahmin günümüz hesaplama yöntemlerine, donanım alt yapısına ve gelecekteki kriptografi ve kripto-analizdeki ilerlemelere bakılarak yapılmaktadır. Özel anahtar için baktığımızda ise 160 bitten 240 bite kadar bir genişleyebilme özelliği bulunmaktadır[3,4].

Yapılan analizler gösteriyor ki eşit uzunluklu anahtar değerleri için RSA ve ElGamal şifreleme algoritmaları benzer güvenliğe sahiptir.

2. ELGAMAL ALGORİTMASI

ElGamal Sistemi farklı logaritmik problem üzerine dayandırılmış bir genel anahtarlama kripto sistemidir. Şifreleme ve sayısal imza algoritmalarından oluşur. Şifreleme algoritması temelde Diffie-Hellman anahtar protokolüne benzer.

Sistem parametreleri, bir asal sayı olan ***p*** ve modül p'e göre elemanların sayısını veren ***g*** tamsayısından oluşur. Alıcı, bir özel anahtar olan a değerine ve bir genel anahtar olan y değerine sahiptir ve $y=g^a \pmod p$'dir. Varsayalım ki *gönderici* bir m mesajını *alıcıya* göndermek istiyor. *Gönderici* öncelikle p'den küçük olmak şartıyla rasgele bir k sayısı üretir. Daha sonra (3) deki işlemi gerçekleştirerek y_1 ve y_2 değerlerini hesaplar[5,6].

$$y_1 = g^k \pmod p \quad \text{ve} \quad y_2 = m \text{ xor } y^k \qquad (3)$$

Buradaki xor bit seviyesinde yapılan "özel veya" işlemini ifade etmektedir. Bu işlemden sonra *gönderici* (y_1,y_2) değer çiftini *alıcıya* gönderir. Şifrelenmiş mesajı alan *alıcı*, (4) deki işlemi gerçekleştirir.

$$m = (y_1^a \bmod p) \text{ xor } y_2 \qquad (4)$$

Bu hesaplamadan sonra orijinal metne ulaşmış olunur.

2.1. Uygulamadaki Şifreleme İşlemi

Bu temel şifreleme ve şifre çözme işlemlerinden sonra, bu uygulamada kullanılan adımları inceleyelim.

Şifrelenecek metni aldıktan sonra, ilk olarak metin bloklara ayrılır. Şifreleme işlemi daha sonra bu bloklar üzerine uygulanacak ve blok şifreleme sağlanacaktır. Blok boyutunu belirlemek, yani bloklarda yer alacak eleman sayısını bulmak için aşağıdaki hesaplama yapılır.

$$Blok=int(log(p)/log(strlen(alph))) \qquad (5)$$

(5)'deki ***alph*** uygulamada kullanılan alfabeyi ifade etmektedir. Bu uygulamada kullandığımız alfabe temel simgeleri içeren, 96 karakterden oluşan bir alfabedir. *Strlen* alfabenin kaç elemanlı olduğunu bulan fonksiyonu gösterir. P değeri de anahtarı ifade eder. P değerinin ve alfabedeki karakter sayısının logaritmalarının oranı, metnin kaç elemanlı bloklara ayrılacağını hesaplamaktadır. Alfabe boyutu küçüldükçe veya p değerinin boyutu büyüdükçe blokların boyutu büyümektedir. Bloklara ayırmanın faydası ise, bilindiği gibi iyi bir şifreleme algoritması şifrelenecek bir girdi verisine karşılık, bir şifrelenmiş çıktı verisi ve birde karakter üretir. Örneğin 1 karakter şifrelenirse, şifreleme sonucu 2 karakter üretilir. Bunun anlamı bloklar 1 elemanlı tutulursa, şifreleme sonucunda şifrelenmemiş mesajdaki karakter sayısının iki katı kadar karakterden oluşan bir şifrelenmiş mesaj elde edilir. Ancak bloklara ayrılırsa bu artış bir miktar azalmış olur. Örneğin, 40 karakterlik bir mesaj 5'erli bloklara ayrıldığında şifreleme sonucunda 48 karakterden oluşan bir şifreli mesaj elde edilir. Bloklara ayırma sayesinde bellek tasarrufu sağlanmış olur. Blok boyutunu arttırmanın dezavantajı ise algoritmanın daha basit hale getirilmesidir.

Dolayısıyla bu uygulamada blok değerini 1 yapacak şekilde bir p değeri belirlendi.

Bu adımdan sonra karakterler birer birer taranmaya başlanır. İlk karakter alınır. k rasgele değeri belirlenir. Bu değer p değerinden küçük olacak şekilde düzeltilir. Alınan k, p ve g parametrelerine göre y_1 değeri hesaplanır(3). Bulunan y_1 değeri, alfabenin uzunluğuna göre modu alınarak küçültülür(6).

$y_1 = y_1 \pmod{\text{strlen(alph)}}$ (6)

cipher1[i]=alph[y_1] (7)

Alfabedeki y_1 inci eleman cipher1 isimli bir diziye atanır(7). (Burada kullanılan i indisi, girilen metindeki kaçıncı karakterin şifrelendiğini gösteren indistir.) Bu dizi şifrelenmiş karakteri tutmaktadır. İlk alınan karakterin alfabedeki kaçıncı karaktere denk düştüğü bulunur. Buradan bir indis döndürülür. Alınan indis, k, p, ve y değerine göre y_2 değeri hesaplanır(3). Yine benzer şekilde y_2 değerinin de alfabenin uzunluğuna göre modu alınarak küçültülmesi sağlanır(8).

$y_2 = y_2 \pmod{\text{strlen(alph)}}$ (8)

cipher2[i]=alph[y_2] (9)

Alfabedeki y_2 inci eleman da *cipher2* isimli bir diziye atanır(9). Bu dizide şifrelenmiş karaktere bir anahtar görevi gören ikincil şifreleme anahtarlarını tutmaktadır. Bu işlemler bittikten sonra, *cipher1* ve *cipher2* dizilerinde tutulan veriler; şifreli metni tutması beklenen ***cipher*** isimli dizide birleştirilirler(10).

cipher[i]=cipher1[i]+cipher2[i] (10)

Tüm karakterlerin taranması tamamlandığında, ***cipher*** dizisi şifrelenmiş metni tutmaktadır.

2.2. Uygulamadaki Şifre Çözme İşlemi

Öncelikle (5) yardımıyla şifreli metin içerisindeki blok sayısı hesaplanır. *Strlen(cipher)/2*blok+2* defa çalışacak bir döngüye sokulur. Karakterin sayısal karşılığı plain1 isimli diziye alınır. Bu sayısal karşılık, karakterin alfabedeki kaçıncı sembole karşılık geldiğidir. Bu sayıdan yola çıkılarak y_1 değeri hesaplanır. Daha sonra gelen karakterin sayısal karşılığı plain2 dizisine atanır. Hatırlarsak bu ikincil karakterler, metin şifrelenirken, şifreli karakterlerin yanında üretilmiş yardımcı şifreleme anahtarlarıdır. Plain2 dizisine alınan sayısal değer yardımı ile y_2 değeri hesaplanır. y_1 ve y_2 değerleri elde edildikten sonra, p ve a değerleri yardımıyla; $(y_1^a \bmod p)$ xor y_2 işleminin sonucu hesaplanır. Bu işlem bize bir indisi geri döndürür. Alfabede bu indise karşılık gelen karakter bizim çözümlenmiş karakterimizdir. Sonucu plain isimli bir diziye atarız. Tüm karakterlerin taranması işlemi bittiğinde, şifreli metnin tamamı çözümlenmiş biçimde, plain isimli dizide tutulur.

3. HASH ALGORİTMASI

Bu algoritma, yetkili kullanıcılar kayıt edilirken kullanılmaktadır. Kullanıcı, programa önceden kayıt edilmemişse uygulamayı kullanamamaktadır. Bu kayıt girişinde kullanıcı, bir şifre girer. Bu şifre doğrudan veri tabanına kaydedilmez. Bu şifre, hash fonksiyonları ile geri dönüşümü olmayan başka bir veri haline dönüştürülerek veri tabanına kaydedilir. Bu yöntemle veri tabanına erişen kötü niyetli kişilerin bu şifreleri ele geçirmeleri zorlaştırılmıştır. Bu uygulamada kullanılan hash fonksiyonu ise şu şekildedir. Girilen şifre, ilk karakteri 0 indisine sahip olacak şekilde her karaktere indis verilerek alınır. Karakterleri sahip oldukları indis değeri 2'nin kuvveti olarak alınarak, karakterlerin ASCII değerleri ile çarpılır. Elde edilen tüm sonuçlar toplanır. Çıkan sayı eğer rasgele alınan bir epsilon değerinden büyükse, sayı ortadan ikiye bölünerek, diğer parçanın üzerine toplanır. Bu işlem, alınan epsilon değerinden küçük oluncaya kadar tekrar edilir.

Örneğin, girilecek şifre, "mustafa" olsun. Şifrenin küçük harflerden oluştuğu görülmektedir. Bilindiği gibi küçük harflerin ve büyük harflerin ASCII kodları farklıdır. Böylece daha sonraki zamanlarda programa giriş yapmak istersek, bu şekilde şifrenin girilmesi gerekmektedir. "Mustafa" bu şekilde yazıldığında anlam olarak aynı olmasına rağmen farklı ASCII kodlara sahip bir hash değeri hesaplanacağından sistem bu şifreyi kabul etmeyecektir. Girilen şifredeki karakterler indislere ayrılır, m karakteri 0 indisine, u 1, s 2, t 3, a 4, f 5, ve a karakteri de 6 indisine sahiptir. Şimdi bu karakterlerin ASCII karşılıklarına bakalım. m:109, u:117, s:115, t:116, a:97 ve f=102 ASCII koduna sahiptir. Şimdi aşağıdaki hesaplama sonucunda,
$109*2^0+117*2^1+115*2^2+116*2^3+97*2^4+102*2^{5+}97*2^6$
12755 değeri ortaya çıkmaktadır. Epsilon değerinin ise 1000 olarak seçildiğini farz edelim. Hesapladığımız değer epsilon değerinden daha büyük olduğu için değer ortadan ayrılarak üzerine toplanacaktır. Sayı sağ-önemli (right-most) olarak ikiye ayrılır. Yani sayıda tek rakam varsa, sağ tarafta, sol tarafa göre 1 adet fazla rakam kalacak şekilde ayırma işlemi gerçekleştirilir. Örneğimize bakarsak, 12755 sayısı, beş adet rakam içermektedir. Dolayısıyla sağ-önemli olarak ikiye ayrılırsa; soldan ilk iki rakam bir sayıyı, sağdan son üç rakam ise diğer bir sayıyı oluşturacaktır. 12 ve 755 sayıları gibi. Sonra bu iki sayı toplanacak. 12+755=767 çıkan sonuç epsilon değerinden küçük olduğu için işlem burada sonlandırılır. Artık şifre veri tabanına yazılırken bu yeni veriden yararlanılmaktadır. Kullanıcı programa giriş yapmaya çalıştığında, o alıştığı gibi şifresini yazacak ve "tamam" düğmesine bastığında, yazılan şifre alınıp yukarıdaki işlemler gerçekleştirilecektir.

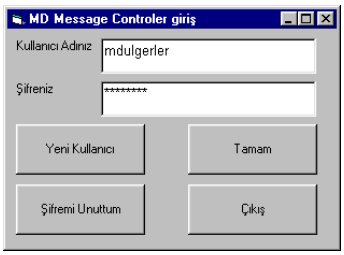
Program son haline getirdiğinde, veri tabanında o kullanıcıya ait olan şifre değeri okunacak eğer bu iki değer tutuyorsa, kullanıcıya programa erişim hakkı verilecektir.

4. E-POSTA UYGULAMASI : MD MESSAGE CONTROLLER

Bu uygulamada, dağıtılmış sistemler için tasarlanmış şifreleme algoritmalarından ElGamal'ı kullanarak, onun e-posta imzalamadaki üstün gücünü, mesajın tamamını şifrelemede kullanıldı. Ayrıca yerel bir işletim sisteminde program çalıştırıldığında, programın kullanıcısını tanıması amacıyla gerçekleştirilen kullanıcı girişi işleminde güvenliği sağlamak için ise bir hash algoritması kullanıldı.

4.1. Uygulamaya Giriş

Uygulama çalıştırıldığında, kullanıcı giriş formu gelmektedir. Kullanıcı bu forma, kullanıcı ismini ve şifresini girmektedir(Şekil 1).

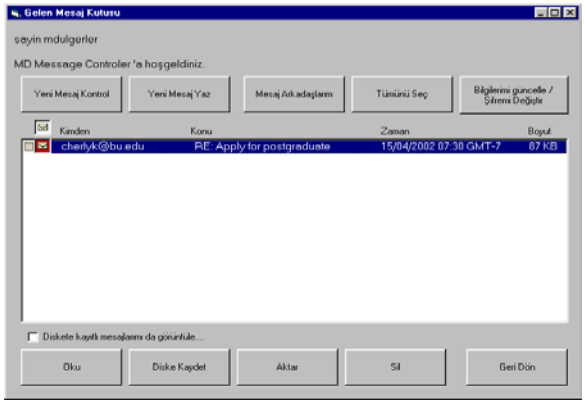


Şekil 1. Kullanıcı Giriş Formu

Girilen şifre yukarıda bahsedilen yöntemle bir hash değerine dönüştürülerek veri tabanına kaydedilir veya daha önce kayıtlı bir kullanıcı ise veri tabanındaki kayıtlı değer ile karşılaştırılır. Karşılaştırılan değerler eşitse, kullanıcıya programa erişim hakkına sahip olmaktadır. Kullanıcı programı kullanmaya başladığı anda, program arka planda, daha önceden o kullanıcı adına kayıt edilmiş, SMTP sunucusuna ve POP3 sunucusuna bağlanır.

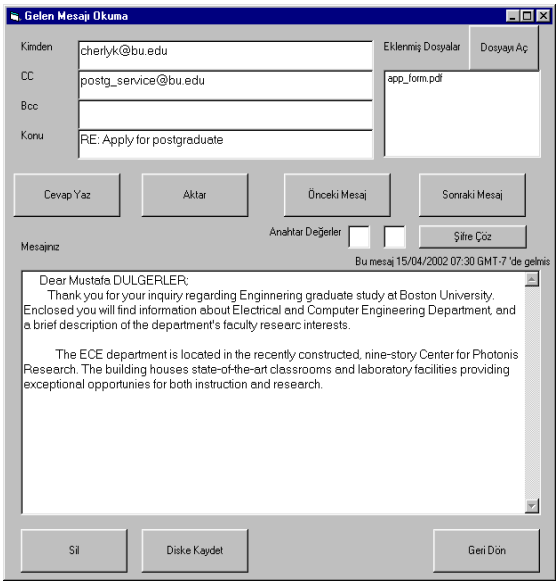
4.2. Gelen Mesajlar

POP3 sunucusundan, kullanıcının hesabına gelmiş olan mesajlar okunur ve listelenir (Şekil 2).



Şekil 2. Gelen Mesaj Kutusu

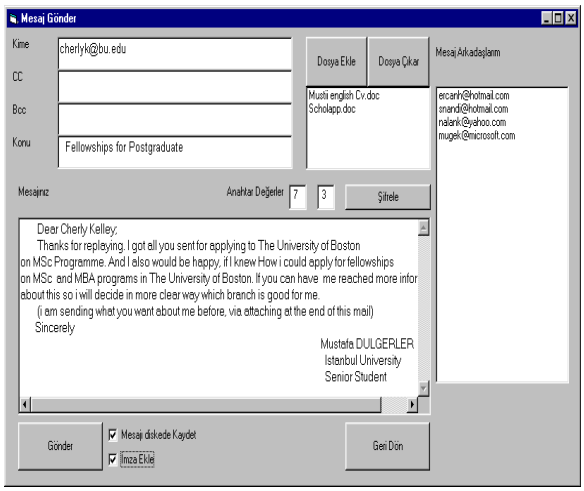
Kullanıcı bu mesajlardan herhangi birini, mesajın üzerine fare ile çift klik yaparak veya form üzerindeki "OKU" düğmesine basarak okuyabilir(Şekil 3). Mesaj, bu programı kullanan başka bir kullanıcı tarafından şifreli olarak gönderilmiş ise, mesaj oku formundaki "ŞİFRE ÇÖZ" düğmesi yardımıyla, şifre çözülmektedir.



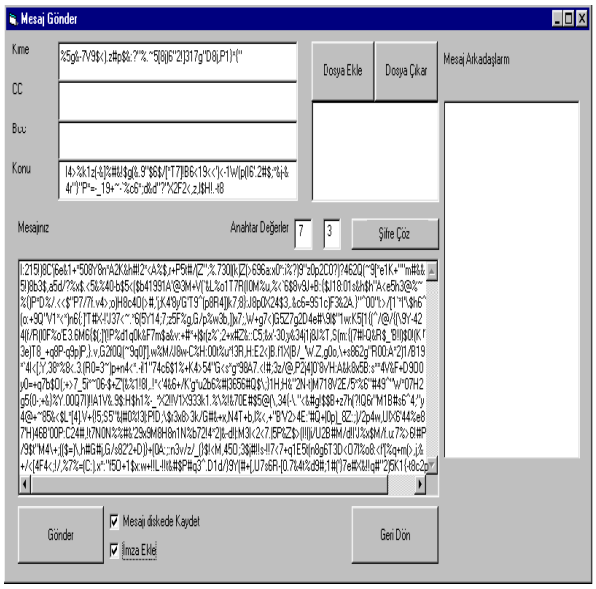
Şekil 3. Gelen Mesajı Okuma Formu

4.3. Gönderilecek Mesajlar

Mesaj gönderilirken, mesaj yazıldıktan sonra form üzerindeki, "ŞİFRELE" düğmesine basılırsa, mesaj ELGAMAL algoritma ile şifrelenir, "GÖNDER" düğmesi yardımıyla mesaj şifrelenmiş olarak alıcıya gönderilir (Şekil 4). Ancak form üzerindeki anahtar değerleri için iki değer girilmelidir. Mesajın şifreli hali ise Şekil 5'te görülmektedir.



Şekil 4. Mesaj Gönderme Formu



Şekil 5. Şifreli Mesajın Görüldüğü Form

4.4. Uygulamadan Çıkış

Çıkış bölümünde ise bağlantı yapılmış olunan sunucularla bağlantı kesilir. Bu uygulama kullanılarak gönderilmiş olan bir şifreli mesajın alıcıda çözümlenmesi için, alıcının da aynı uygulamaya sahip olması gerekir. Anahtar değerler mesaja eklenerek gönderildiğinden, alıcının bu parametreleri önceden bilmesi gereği ortadan kalkmıştır.

5. SONUÇ VE İLERİYE YÖNELİK ÇALIŞMALAR

Bu uygulama, ElGamal Şifreleme Algoritması ve özgün bir Hash algoritması kullanılarak tasarlanmıştır. Uygulamanın performansı kullanılan donanıma bağlı olarak değişebilir. Programda anlaşılırlığından dolayı, basit bir hash algoritması kullanılmıştır. Kullanılan bu yapı 16 bitlik değerler için düşünülmüştür. Bu değerin arttırılması, performansı azaltırken, güvenliği arttırmaktadır. ElGamal şifreleme algoritması bu uygulamada 8 bitlik veriler üzerinde kullanılmıştır. Bu değeri arttırmak da güvenliği arttırır. Bu uygulamanın klasik e-posta uygulamalarından tek farkı şifreleme ve şifre çözme işlemlerinin olmasıdır. Kullanıcı bu uygulama ile diğer e-posta uygulamalarındaki işlemleri gerçekleştirebilmektedir.

Bu çalışmanın sonuçlarını, diğer şifreleme algoritmalarındaki performans değerleri ile karşılaştırdığımızda, RSA algoritmasıyla arasında fazla bir fark olmadığını görüldü [7,8,9].

Bir sonraki adımda, programa bir span sözlüğünün eklenmesi planlanmaktadır. Bu sözlük yardımıyla şifreleme işlemi tamamlandıktan sonra, şifrelenmiş her karaktere karşılık, sözlükten bir cümle (sözcük) bulunacak ve yer değiştirilecektir. Bu, program performansını düşürecek ve mesajın boyutunu arttıracaktır ancak güvenliği de en üst seviyeye taşıyacaktır. Mesaj boyutunu düşürmek için sıkıştırma algoritmalarının kullanılması düşünülmektedir.

KAYNAKLAR

1. Stinson D. R., Cryptography Theory and Practice, CRC Press, 1995, Florida
2. William S., Network And Internetwork Security Principles And Practice, Prentice-Hall, Inc. 1995, New Jersey.
3. http://www.idssoftware.com/jsecure.html
4. http://www-d.connect.ti.com/dsp/tpcat/tpcodec.nsf/SoftwareForExternal/59BEAA0765A00DA4862569F2005645E1
5. http://www.soriac.de
6. http://www.cryptome.org
7. SERTBAŞ Ahmet and SARISAKAL M. Nusret (2001) : "VMAIL / An Application For A Secure E-Mail Transmission Using Encrypting Techniques" ELECO'2001 International Conference on Electrical and Electronics Engineering, 7 - 11 November 2001, Bursa, TURKEY, Proceedings (pp. 363-367, ISBN: 975 359-479-4).
8. SARISAKAL M. Nusret, SEVGEN Selcuk and ACAR Dogal (2001) : "Developing An Application Of RSA Algorithm With JAVA", ELECO'2001 International Conference on Electrical and Electronics Engineering, 7 - 11 November 2001, Bursa, TURKEY, Proceedings (pp. 332-335, ISBN: 975 359-479-4).
9. SARISAKAL M. Nusret, SAVAŞAN Volkan, SERTBAŞ Ahmet (2001) : "RSA ve IDEA Algoritmalarını Birlikte Kullanan Güvenli Bir E-Posta Uygulaması : VMAIL", I.U. Journal of Electrical & Electronics, Vol. 1, No. 2, pp 297-305, 2001, ISSN 1303 – 0914.