AKIŞ ŞİFRELEME TEKNİĞİNE YENİ BİR YAKLAŞIM: DÜZENSİZ ŞİFRELEME

Oğuzhan TAŞ[1] **Bilal ALATAŞ[2]** **Erhan AKIN[3]**

{oguzhantas, balatas, eakin} @firat.edu.tr
[1,2,3] Bilgisayar Mühendisliği Bölümü
Mühendislik Fakültesi
Fırat Üniversitesi, 23119, Elazığ

Anahtar Kelimeler: Kriptografi, Şifreleme, Deşifreleme.

ÖZET

Bu makalede önerilen şifreleme tekniği, bitsel olarak şifreleme/deşifreleme yapmaktadır. Metindeki her bir karakter için rasgele olarak belirlenen ekleme sabiti denilen sayı (r) o karakterin şifrelenmesini 2^r farklı şekilde değiştirmektedir. Bu işleme ilaveten ekleme sabitinin değerlerine göre de orijinal metindeki karakter, bitsel forma çevrilip parçalanarak bitsel değişim yapılmaktadır. Daha önce literatürde kusursuz şifreleme olarak yer alan One Time Pad (Bir seferlik ıstampa)'den farklı olarak aynı anahtarla farklı şifreli metinlerin elde edilebilmesi, önerilen tekniğin daha avantajlı olduğunu göstermektedir. Diğer bitsel algoritmalara göre yapısal bakımdan daha basit olması, önerilen tekniğin donanımsal ve yazılımsal olarak geliştirimini de kolaylaştırmaktadır.

1.GİRİŞ

Kriptografi bilimi, askeri, finansal, vb. birçok alanda veri gizliliğinin, bütünlüğünün, güvenliğinin sağlanması amacıyla yıllardır kullanılmaktadır. Kriptografi'de işlemler, beş ana bileşenle yapılmaktadır. Bunlar, şifreleme fonksiyonu, deşifreleme fonksiyonu, şifreleme anahtarı, şifreli metin ve orijinal metin'dir. Farklı anahtar değerlerine göre şifreleme fonksiyonları farklı değerler üretirler. Dolayısıyla şifreyi çözmeye çalışan saldırgan, anahtar olmadan orijinal metni elde edemez. Bu anlatımdan da anlaşıldığı gibi kriptografide şifreleme fonksiyonu, deşifreleme fonksiyonu ve şifreli metin gizlenmez, anahtar ve orijinal metin ise gizli tutulur [1,2,3]. Anahtar olmadan yapılacak şifreleme işleminde, şifreleme ve deşifreleme fonksiyonunun gizli tutulması gerekecektir. Bir gurup arasında yapılan şifreleme işlemlerinde bu şifreleme tarzının kullanıldığını düşünelim. Guruptan birisi şifreleme fonksiyonunu açıkladığında guruptaki tüm üyelerin yeni bir şifreleme fonksiyonu üzerinde anlaşmaları gerekecektir [1]. Eğer gurup, anahtarlı şifreleme tarzını kullansaydı, şifreleme/deşifreleme fonksiyonu değil anahtar değiştirilerek güvenli haberleşme yeniden sağlanabilecekti.

Kriptografide, şifreleme ve deşifreleme işlemi için aynı anahtarın kullanıldığı simetrik anahtar sistemlerin yanı sıra şifreleme ve deşifreleme için farklı anahtarın kullanıldığı asimetrik sistemler de geliştirilmiştir [1,3,4]. Asimetrik sistemlerde farklı anahtarlar kullanıldığından dolayı uzak mesafedeki kişilerin haberleşmesinde anahtar dağıtımı sorun olmamaktadır. Simetrik anahtar sistemlerinde ise anahtar dağıtımı sorun olmasına rağmen şifreleme ve deşifreleme daha hızlı yapılmaktadır. SSL [21], SET [20] gibi güvenlik protokollerinde halen simetrik anahtar yöntemi kullanılmaktadır. Bu çalışmada önerilen teknik, simetrik anahtar sistemine uygun olarak geliştirilmiştir.

Kriptografide, blok şifreleme ve akış (stream) şifreleme olmak üzere iki temel simetrik algoritma tipi vardır. Blok şifreleme, orijinal metni ve şifreli metni bloklayarak şifreleme/deşifreleme işlemini yapar. Blok uzunluğu için, genellikle 64 bit bazı zaman daha fazla uzunluk seçilir. Akış şifrelemede ise, orijinal metnin ve şifreli metnin bir biti veya baytı (bazı zaman bir 32 bitlik kelime) üzerinde işlem yapılır. Blok şifrelemede daima aynı anahtar kullanarak aynı orijinal metin bloğundan aynı şifreli metin bloğu elde edilir. Blok şifreleme algoritmalarına örnek olarak DES[6], RC5[7], SAFER[8], Blowfish[9], TEA[16] ve FEAL[10] verilebilir. Akış şifrelemede ise aynı orijinal metindeki bir bit veya bayt her seferinde farklı bir bit veya bayt olarak şifrelenir. Bu algoritma tipine örnek olarak RC4[1], A5/1[12], ORYX[5] ve SEAL[11] verilebilir. Blok şifreleme daha çok yazılım uygulamalarında, akış şifreleme ise bitsel olarak işlem yapmasından dolayı daha çok donanım uygulamalarında tercih edilir. Önerilen şifreleme tekniği, donanım uygulamalarında olduğu kadar yazılım uygulamalarında da hızlı çalışacak şekilde geliştirilmiştir.

2. DÜZENSİZ ŞİFRELEME

Düzensiz şifreleme tekniği, bitsel olarak çalışan bir şifreleme tekniğidir. Önerilen bu teknikte şifreleme ve deşifreleme işlemi için aynı anahtar (secret key,

single key) kullanılmaktadır. Anlatım boyunca aşağıdaki notasyon kullanılacaktır:

P : Orijinal Metin
C : Şifreli Metin
E( ) : Şifreleme fonksiyonu.
D( ) : Deşifreleme fonksiyonu.
K : Anahtar
N : Orijinal metindeki karakter sayısı.
r : Ekleme sabiti.
S : Şifreli metindeki bit sayısı.
y : Şifreli metnin bitleri.
x : Orijinal metin karakterinin bitleri.
n : Orijinal metindeki her bir karakterin kaç bitle ifade edildiği.
Karakter( ) : ASCII değerin karakter karşılığını veren fonksiyon.

2.1. ŞİFRELEME İŞLEMİ

Bu şifreleme tekniğinde, ilk önce orijinal metindeki her bir karakterin kaç bitle ifade edileceği (*n*) seçilir. Ekleme sabiti, 8 veya daha büyük bir sayı olmalıdır. Çünkü ASCII tablodaki karakterler, en az 8 bitle ifade edilmektedir. Bundan sonraki örneklerde *n* değeri varsayılan olarak 8 alınacaktır. Sonra, her bir orijinal metin karakteri için rasgele olarak bir ekleme sabiti (*r*) seçilir. Ekleme sabitinin değerleri $1 \le r \le n$ aralığından seçilecektir. Orijinal metindeki karakterler $P=\{p_1,p_2,p_3,...,p_N\}$ şeklinde ifade edilsin. ($\forall p_i \in \{$ *Karakter*(1...255) $\}$, $1 \le i \le N$ Orijinal metnin her bir karakterinin bitleri de $C=\{x_1,x_2,x_3,...,x_n\}$ ve $\forall x_j \in \{0,1\},\ 1 \le j \le n$ şeklinde ifade edilsin. Sonuçta elde edilen şifreli metin bitleri $\{y_1,y_2,y_3,....,y_s\}$ *ve* $\forall y_t \in \{0,1\}, 1 \le t \le S$ şeklinde olacaktır.

Ekleme sabiti seçildikten sonra bir orijinal metin karakterinin bitsel karşılığı, ekleme katsayısının değeri kadar eşit parçalara ayrılır. Eşit parçalama işleminin gerçekleştirilemediği durumda en büyük parça başta bırakılmak üzere ondan sonraki parçalar eşit alınır. Örneğin ekleme katsayısı 6 olarak belirlensin. 8 bitlik orijinal metin 6 parçaya ayrılır. İlk iki bit bir parça, sonraki iki bit bir parça geri kalan 6 bitin her biri bir parça olarak alınır. Örneğin orijinal metindeki bir karakter "A" olsun. A = *Karakter(*65*)* karakterinin bitsel karşılığı 10000001'dir. Ekleme katsayısı *(n)* 6 olarak belirlenmişse aşağıdaki şekilde parçalanacaktır.

10 ⟶ 1.Parça
00 ⟶ 2.Parça
0 ⟶ 3.Parça
0 ⟶ 4.Parça
0 ⟶ 5.Parça
1 ⟶ 6.Parça

Şekil 1. Metin bitlerinin parçalanması.

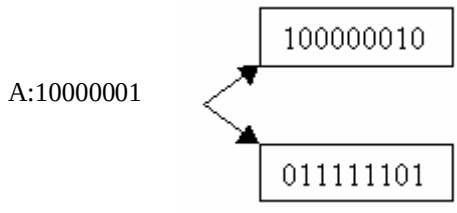
Yukarıdaki işlem program tasarımında formüle edilmelidir. Bunun için *n/r* işlemi yapılır. Bu bölümden kalan sayı *k* bölüm de *q* ise;

$$n = q^* r + k \quad (1)$$
$$k = n \bmod r \quad (2)$$
$$k^* (q+1) \quad (3)$$
$$(r-k)^*q \quad (4)$$

Eşitlik 3'teki ifade ilk önce *k* tane *(q+1)* bitlik parçaya bölüneceğini, daha sonra da *(r-k)* tane *q* bitlik parçaya bölüneceğini gösterir. eşitlik (3) ve eşitlik (4) toplandığında da eşitlik (1) elde edilir. Eşitlik (2) ise kalanın modülasyon işlemiyle elde edilişini belirtmektedir. Yukarıdaki formülsel ifadeler *n* ve *r* ifadelerinin her değeri için geçerlidir. Örneğin *n* = 8 ve *r* = 6 olursa;

$k = 8 \bmod 6 = 2$
$q = 8 / 6 = 1$
$r - k = 6 - 2 = 4$

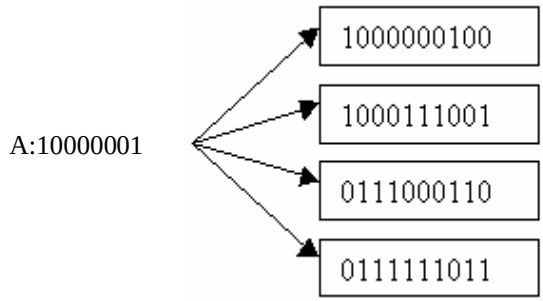
Buradan 2 tane 2 bitlik sonra da 4 tane 1 bitlik parçaya ayrılacağını belirlemiş oluruz. Parçalama işleminden sonra ekleme sabitinin değeri kadar ikilik sistemde bit, her bir karakterin ikilik sistemdeki karşılığının sonuna eklenilir. Ekleme sabiti 5 ise 2^5 farklı şekilde bit sona eklenebilir. Dolayısıyla orijinal metindeki "A" karakteri, 2^5=32 farklı şekilde şifrelenmiş olacaktır. Ekleme sabitinin büyük değerler alması, şifreleme işleminin karmaşıklığını artıracağından daha güvenli sonuç elde edilecektir. Sonraki adımda, ekleme sabitinin ikilik değerlerine göre parçalarda bitsel değişiklik yapılır. Örneğin ekleme sabitindeki bit "0" ise karşı gelen parçada aynen bırakılır, "1" ise karşı gelen parçadaki bitler ters çevrilir. Örneğin Şekil 1'deki örnek için ekleme sabiti 101011 (6 bit) olsun. Bu sabit değerdeki ilk bit "1" dir. Bu da 1. parçadaki "10" değerinin "01" olacağını belirtir. Sonra ekleme sabitindeki ikinci bite bakılır. Bu bit "0" olduğundan 2.parçadaki "00" değeri aynen bırakılır. Sonraki bit "1" olduğundan üçüncü parçadaki "0" değeri "1" yapılır ve bu işlem aynı şekilde devam ettirilir.Ekleme sayısının alacağı değerler ve bu değerlere göre orijinal karakterin şifrelenmesini şu şekilde açıklayabiliriz:
Ekleme sabiti (*r)=1* durumunda karakterin kendisi ve tüm bitlerinin tersi şifreli A' yı oluşturmada kullanılır:



Şekil 2. Ekleme sabiti *(r)* =1 için durumlar.

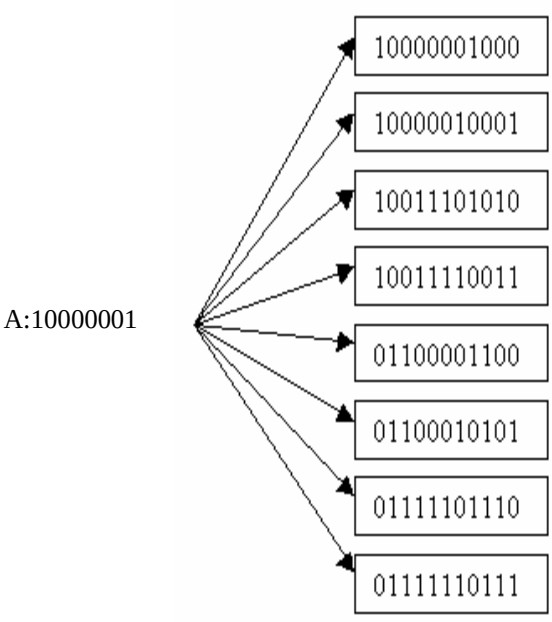
Şekil 2'de bir karakter (8 bit) ekleme sabitiyle 9 bit olmuştur, en sondaki bir bit ekleme bitidir. 0 eklenmesi durumunda tüm bitler aynı kalmış, 1 eklenmesi durumunda ise tüm bitlerin tümleyeni alınmıştır.

r=2 durumunda ise karakterin bitsel ifadesi iki parçaya ayrılır ve ekleme sayısı dört farklı değer alabilir. 00 eklenmesi durumunda tüm bitler aynı kalır, 01 eklenmesi durumunda yukarıda anlatıldığı şekilde ilk parçanın kendisi, ikinci parçanın ise tersi alınır; 10' da ise ilk parçanın tersi ve ikincinin kendisi; 11 durumunda ise tüm bitlerin tersi alınır ve ekleme bitiyle beraber şifrelenmiş karakteri temsil eder.



Şekil 3. Ekleme sabiti *(r)*=2 için durumlar.

r=3 durumunda karakter bitleri yukarıdaki anlatıldığı gibi üç parçaya ayrılır ve ekleme bitinin durumuna göre 8 farklı şekilde gösterilebilir. Ekleme bitleri 000, 001, 010, 011, 100, 101, 110, 111 durumunu alır. Bu durumda A harfi 11 bit olarak aşağıdaki görüldüğü gibi 8 farklı şekilde şifrelenebilir:



Şekil 4. Ekleme sabiti *(r)*=3 için durumlar.

2.1. DEŞİFRELEME İŞLEMİ

Deşifreleme işlemi için, şifrelemede kullanılan anahtarın deşifreleme yapacak tarafta bulunması gerekmektedir. Anahtar dosyasındaki sayılar her bir karakterin sonuna kaç bit eklendiğini (ekleme sabitlerini) sırayla belirttiği için deşifreleme işlemi buradan yararlanarak kolayca yapılabilecektir.

Deşifreleme işlemini yapan kişinin anahtara ek olarak *n* değerini de bilmesi gerekmektedir. *n* değerinin büyük seçilirse, şifreli metin daha da karmaşıklaşacak böylece güvenlik daha da artacaktır. ASCII tablodaki her karakter 8 bitle ifade edilebilmesine rağmen *n*=10 seçilirse her bir karakter 10 bitle ifade edilecek, şifreli metin uzunluğu da buna bağlı olarak artacaktır. *n* değeri deşifreleyici tarafından öğrenildikten sonra seçilen *n* değerine anahtar dosyasındaki değerler sırayla eklenerek her karakterin kaç bitle şifrelendiği bulunur. Sonra şifreli metinden sırayla her bir karakter için kodlanan bit dizileri alınarak çözülür. Örneğin *n*=8 seçilsin ve şifreleme anahtarı;

5423714234561423756435322365315 2.....
şeklinde olsun. Şifreli metin de;

10111101010011110001010101110001 1...
şeklinde olsun.

Deşifreleme yapılırken ilk karakterin şifresinin çözülmesi için *n*=8 değeri ile anahtarın ilk değeri olan 5 değeri toplanır. Elde edilen 13 sayısı, orijinal metnin ilk karakterinin 13 bitle şifrelendiğini belirtir. Sonraki karakter için *n*=8 değeri ile anahtarın ikinci değeri olan 4 toplanır ve buradan 14.bit ile 26. bit arasının 2. karakterin şifrelenmiş değerine karşı geldiği anlaşılır ve bu işlem aynı şekilde devam ettirilerek tüm karakterlerin şifreli metinde kaç bitle ve hangi bitlerle ifade edildiği bulunur.

İlk karaktere karşı gelen bit grubunun 1011110101001 olduğunu bulduktan sonra son 5 biti (ekleme sayısı) ayrıştırılır. Bu işlemden sonra eşitlik (3) ve eşitlik (4) kullanılarak 8 bitlik kısım, (2,2,2,1,1) şeklinde 5 parçaya ayrılır. Bundan sonra, elde edilen ekleme sayısının bitlerine göre (01001) daha önce yapılan şifreleme işleminin tersi alınır. Örneğin ekleme sayısının ilk biti "0" olduğundan "10" aynen kalır, ikinci biti "1" olduğundan "11" in tersi alınarak "00" elde edilir. Bu işlem aynı şekilde devam ettirildikten sonra 10001100 = î orjinal metin karakteri elde edilir.

2.2.ŞİFRELEME ve DEŞİFRELEME ANAHTARI

Şifreleme ve deşifreleme işlemleri için kullanılan anahtar, her bir karakter için kullanılan ekleme sabitlerinin yan yana yazılmasından meydana gelir. Dolayısıyla anahtar uzunluğu, orijinal metindeki

karakter sayısı kadardır. Anahtar uzunluğunun fazla olması, tüm şifreleme algoritmalarında olduğu gibi burada da işlemin karmaşıklığını artırmakta çözümü zorlaştırarak daha güvenli yaklaşım sunmaktadır.

Literatürde kusursuz şifreleme olarak yer alan One Time Pad (Bir seferlik ıstampa) tekniğinde tekrarlanmayan anahtarlar kümesi ile şifreleme yapılarak maksimum güvenlik sağlanmaktadır. Fakat bu tekniğin de iki dezavantajı bulunmaktadır. Mesajı gönderen ve alan arasında tam bir senkronizasyon sağlanmalı ve birbirinden farklı çok sayıda anahtar üretilmesi gerekmektedir. Her seferde kullanılan anahtarın daha önce kullanılıp kullanılmadığını kontrol edilmelidir. Ayrıca daha önce kullanılan anahtarların saklanması için bir veritabanı kullanılması gerekmektir [22]. Bir seferlik ıstampa (one-time pad) tekniği için bu sorun olmasına rağmen önerilen şifreleme tekniğinde anahtar tekrar edilse bilse şifreleme içeriği yine değişeceğinden sorun değildir.

3.SONUÇ

Bu şifreleme tekniğinde, daha önce literatürde kusursuz şifreleme olarak yer alan One-Time Pad (Tek Seferlik Istampa) tekniğine benzer fakat bu teknikten daha avantajlı bir şifreleme algoritması geliştirilmiştir. Tek Seferlik Istampa'da aynı anahtardan aynı şifreli metin elde edilmektedir. Önerilen teknikte ise aynı anahtarla birbirinden farklı şifreli metinler elde edilmektedir. Ayrıca önerilen teknikte, şifreli metin uzunluğu orijinal metin uzunluğundan daima daha büyük olacaktır. Ekleme sabitinin rasgele olması şifreleme işleminin kırılmasını imkansız hale getirmektedir. Ekleme katsayısına göre karakter bitlerinin de değişmesi şifreleme işleminin güvenliğini bir kat daha artırmaktadır.

KAYNAKLAR

1. B. Schneier, Applied Cryptography 2nd Edition, John Willey & Sons Inc, New York, 1996.
2. A. Menezes, Van Oorschot O., Vanstone S., Handbook of Applied Cryptography, CRC Press, 1997.
3. W. Stallings, Network Security Essentials Applications and Standards, Prentice Hall, New Jersey, 2000.
4. D.R. Stinson, Cryptography Theory and Practice, CRC Press, 1995.
5. D. Wagner, L.Simpson, E.Dawson, J.Kelsey, W.Millan, B. Schneier, Cryptanalysis of ORYX, Fifth Annual Workshop on Selected Areas in Cryptography, Springer Verlag, Ağustos 1998.
6. ANSI X3.106, "American National Standard for Information Systems – Data Encryption Standard- Modes of Operation", American National Standard Institute, 1983.
7. "The RC5 Encryption Algorithm", B. Prencel, Fast Software Encryption, Second International Workshop (LNCS 1008) 86-96 Springer-Verlag, 1995.
8. "SAFER K-64: One year later", B.Preneel, editor, Fast Software Encryption, Second International Workshop (LNCS 1008), 212–241, Springer-Verlag, 1995.
9. B. Schneier, "Description of a new variable-length key, 64-bit block cipher (Blowfish)", R. Anderson, editor, Fast Software Encryption, Cambridge Security Work-shop (LNCS 809), 191–204, Springer-Verlag, 1994.
10. S. Miyaguchi, "The FEAL cipher family", Advances in Cryptology–CRYPTO '90 (LNCS 537), 627–638, 1991.
11. P. Rogaway, D. Coppersmith, "A Sofware Optimized Encryption Algorithm", Journal of Cryptology, 273-287, 1998.
12. Golic J. Dj., Cryptanalysis of Alleged A5 Stream Cipher , Proceedings of Eurocrypt 97, Springer LNCS 1233, 239-255, 1997.
13. D. Kahn, "The Code Breakers- The Comprehensive History of Secret Communication from Ancient Times to the Internet" , Revised and Updated Edtion, Scribner, 1996, USA.
14. M.J.B. Robsaw, "Stream Ciphers", RSA Laboratories Technical Report, 1995.
15. J. Kahanek, "Protecting Business Application with Encryption Symmetric and Asymmetric", 2000.
16. D.J.Wheeler, R.M.Needham, "TEA, a Tiny Encryption Algorithm", Cambridge University, England.
17. C.M.Adams, "Simple and Effective Key Scheduling fo Symmetric Ciphers", Workshop on selected Areas in Cryptography Workshop Record, Kingston, Ontario, 5-6 May 1994, pp. 129-133.
18. C.M. Adams "Symmetric Cryptographic System for data encryption", U.S Patent 5,511,123, 23 Nisan 1996.
19. Rivest, Shamir, ve Adleman. "A method for obtaining digital signatures and public-key cryptosystems". Comm. ACM, Şubat 1978, 120-126.
20. MasterCard Inc. SET Secure Electronic Transaction Specification, Book 1: Business Description, MasterCard Inc., May 1997.
21. A. O. Freier, P. Karlton, and P. C. Kocher. The SSL Protocol Version 3, Netscape Communications Corp., 1996, available from http://home.netscape.com/eng/ssl3.
22. C.P.Pfleeger, "Security in Computing" Second Edition, Prentice Hall, 1997.