T.C
FIRAT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

KAOTİK İKEDA HARİTASI ile GÜÇLÜ KRİPTOGRAFİK ÖZETLEME FONKSİYONU ELDE EDİLMESİ

YÜKSEK LİSANS TEZİ

KEVSER KIRKIL

(08229101)

Tezin Enstitüye Verildiği Tarih : 18 Ocak 2012
Tezin Savunulduğu Tarih : 2 Şubat 2012

Tez Danışmanı : Yrd. Doç. Dr. A.Bedri ÖZER (F.Ü)
Diğer Jüri Üyeleri : Yrd. Doç. Dr. Galip AYDIN (F.Ü)
Yrd. Doç. Dr. Mustafa TÜRK (F.Ü)

ŞUBAT – 2012

ÖNSÖZ

Elektronik ortamda bulunan bilgilerin önemliliğinin artması güvenliğine olan ihtiyacın artmasını gerektirmiştir. Gelişen teknolojilere bağlı olarak elektronik imza, mobil imza gibi güvenlik anahtarları geliştirilmiştir.

Bu çalışmada bilgi güvenliğini sağlayan bir çok uygulamanın alt yapısını oluşturan özetleme algoritmaları kaotik fonksiyonlar kullanılarak uygulanmış ve testleri yapılmıştır. Bu süreçte her türlü yardımını esirgemeyen danışmanım Yrd.Doç.Dr. A. Bedir Özer'e ve Arş.Grv. Fatih Özkaynak'a teşekkürlerimi sunarım. Ayrıca sevgili aileme olan minnettarlığımı bir kez daha ifade etmek isterim.

KEVSER KIRKIL

ELAZIĞ – 2012

İÇİNDEKİLER

Sayfa No

ÖZET

Elektronik ortamda güvenliğimizi sağlayacak araçlara olan ihtiyacımız hızla artmıştır. Bu sebeple bilgisayar ortamında güvenliğimizi sağlayacak unsurlar ve hayatımıza etkileri giderek önem kazanmıştır. Güvenlik uygulamalarının gücü bu uygulamaların temelini oluşturan algoritmaların dayanıklılığına, hızına ve etkinliğine bağlıdır. Bu noktada elektronik imza, mesaj doğrulama gibi önemli sistemlerin alt yapısında bulunan özetleme algoritmalarındaki çakışmaları önlemek için son yıllarda kaotik fonksiyonlarla entegrasyonu çokça yapılmıştır. Kaotik fonksiyonların başlangıç koşullarındaki küçük değişiklikler sonuç değerini çok fazla değiştirir. Bu da değişen giriş verisine göre farklı özetler üreten özetleme algoritması için bir avantajdır.

Kaotik özetleme fonksiyonu açık metinde özet üretirken kaotik fonksiyonları kullanır. Bu çalışmada İkeda haritası tabanlı kaotik fonksiyon kullanılmış ve sonuçları test edilmiş, diğer kaotik özetleme algoritmalarıyla karşılaştırılmıştır. Test sonuçları geliştirilen algoritmanın çakışma ihtimalinin daha düşük olduğunu göstermektedir bu da geliştirilen algoritmanın daha dayanıklı olduğunu belirtmektedir.

Anahtar Kelimeler : Kaotik Özetleme Algoritmaları, İkeda Harita, Özetleme Algoritmaları, Bilgi Güvenliği

SUMMARY

Need of us to tools that provide security on digital word have incrased rapidly. Due to technological development the need for information security has increased rapidly. For this reason, providing security elements and influences to our life has become more important. The power of security applications depends on the strength, speed and efficiency of the underlying algorithm. At this point in recent years, for prevent collisions of hash algorithm which has been background of important systems such as digital signature and message authentication chaotic functions are used as integrated with hash algorithms. Change of initialization conditions of chaotic functions influence as extreme to output stations. This is advantage for hash algorithms because of it must be generate different hash for different plain text.

Chaotic hash functions use chaotic functions when product hash from plain text. In this study, Product hash with use chaotic Ikeda map and tested with compare other chaotic hash functions. Test results show conflict probability of developed algorithm's is lower than compared algorithms and that is indicate developed algorithm more strength than others.

Key Words : Chaotic Based Hash Functions, Ikeda Map, Hash Functions, Information Security

ŞEKİLLER LİSTESİ

Sayfa No

TABLOLAR LİSTESİ

Sayfa No

SEMBOLLER LİSTESİ

P : Açık Mesaj (Plain text)

C : Şifreli Mesaj (Chipper text)

u : Ekseni ifade eden değişken

t : Zamanı ifade eden değişken

v(t) : Hızı ifade eden değişken

R : Reel sayılar

M : Faz düzlemi (kaos analizi)

ß : Dinamik sistemler için geliştirme fonksiyonu (Kaos analizi)

α : Belirli bir noktadan periyodik iz (Kaos analizi)

S : Poincare bölgesi (Kaos analizi)

H : Özetleme fonksiyonu

M : Açık mesaj (Kriptografik özetleme algoritmalarının özellikleri)

h : Özet verisi

$\oplus$: Mantıksal XOR işlemi

bij : Bloklara ayrılmış mesajda i. bloğun j. bitini ifade eder

K : Mesajlama algoritmasının anahtar değeri

IV : Özetleme algoritmasında kullanılan başlangıç vektörü

D : Şifreleme (decryption) fonksiyonunu ifade eder

E : Şifre çözme (encryption) fonksiyonunu ifade eder

f : Sıkıştırma fonksiyonu

R : Özetleme fonksiyonunun hızı

$\mathbf{E}_{g(Hi-1)}(m_i)$: g fonksiyonundan geçirilen H_{i-1} 'in m_i ile birlikte E yani şifreleme işlemi

$\mathbf{p}(G_{i-1})$: Rastgele serbest sabit nokta permutasyonudur

c : 0 olmayan herhangi bir sabit sayı

g ve $\tilde{g}$: Des'e uygun anahtar üretmek için fonksiyonlar (Diğer sıkıştırma fonksiyonlarından örnekler)

Hi ve $\widetilde{Hi}$	: MDC-2 için başlangıç değerleri
T(x)	: Kaotik tent harita fonksiyonu
Asc(x)	: x değerinin Ascii karşılığı
Bmin	: B dizisinin en küçük değeri
Bmax	: B dizisinin en büyük değeri
$\bar{B}$	: B dizisinin ortalaması
P	: Değişen bit sayısının olasılık değeri (Kaotik özetleme algoritmaları güvenlik testleri)
Delta B	: Değişen bit sayısının standart sapması
Delta P	: Değişen bit sayısının olasılığının standart sapması
N	: Tekrar sayısı

KISALTMALAR LİSTESİ

ASCII	: Bilgi Alışverişi İçin Amerikan Kodlama Standardı
HMAC	: Hash Based Message Authentication Code
SHA	: Secure Hash Algorithm
CBC	: Chiper Blok Chaining
NIST	: National Institue of Standards and Technology

1.GİRİŞ

Bilgi güvenliği konusunda gerek kişisel gerekse kurumsal olarak bilinçlenme yayıldıkça çözümler de buna paralel olarak daha hızlı gelişmeye başlamıştır. Sahip olduğumuz değerlere dışarıdan gelen tehditlerin neden olabileceği zararın boyutu arttıkça korumak için kullanılan yollar da buna bağlı olarak çeşitlilik göstermeye başlamıştır. Tarihi süreç içerisinde önem arz eden bilgilerin korunması için kullanılan yöntemler internetin kullanılmaya başlamasıyla birlikte çok daha önemli bir noktaya gelmiştir. Bu kapsamda geliştirilen güvenlik teknolojilerinden en önemlisi e-imza'dır.

Bu tez çalışmasında e-imzanın temelinde bulunan özetleme algoritması kaotik fonksiyonlar kullanılarak geliştirilecek ve test edilecektir. E-imza; elektronik belgelerde kimlik doğrulama, içerik doğrulama, kullanıcıya özgü olma ve kullanıcının imzaladığı belgeyi inkâr edememe işlevlerini içerisinde barındırır ve her geçen gün kullanım alanı yasal dayanaklarıyla genişlemektedir.

Özetleme algoritmaları bir metinden sabit uzunlukta ve metinle ilişkisi kurulamayan, rastgele karakterlerden oluşmuş gibi görünen, o metne özgü özet veri üretmektedirler. Kaotik fonksiyonların başlangıç şartlarına göre çok fazla değişkenlik göstermesi ve çıktılarının kolaylıkla tahmin edilememesi gibi özellikler özetleme fonksiyonlarının çalışma prensipleriyle örtüşmektedir.

Bir özetleme algoritmasının farklı mesajlar için aynı özeti üretmesine çakışma denir. Zamanla hızları iyileşen bilgisayarların çakışma bulma ihtimalleri artmaktadır. Bu çalışmada amaç çakışma ihtimali düşük, daha güçlü, daha karmaşık özetleme algoritması geliştirmektir. Bu amaçla kaotik tabanlı fonksiyonlar kullanılmıştır. Algoritma daha önce yapılan çalışmalarla karşılaştırılmıştır ve test sonuçları sunulmuştur. İncelenen algoritmaların zayıflıklarını giderici yöntemler önerilmiştir. Temel amacımız bilgi güvenliğini sağlamak olduğu için bu bölümde bilgi güvenliği ve önemi anlatılacaktır.

1.1.Bilgi Güvenliği ve Önemi

İnternet kullanımının oranı sanal dünyada karşı karşıya olduğumuz tehditlerden ve bu tehditlerin farkındalık oranından daha fazla olduğu için kullanım arttıkça saldırılar ve buna

bağlı olarak zararın maliyeti artmıştır. Microsoft'un yapmış olduğu araştırmaya göre 2003 yılında 31.000.000.000 e-posta mesajından %33.3'ü istenmeyen yani spam, reklam içerikli vs. iken, 2006 yılında 60.000.000.000 e-postada bu oran %50'ye çıkmıştır. Computer Crimeand Security Survey (Bilgisayar Suçları ve Güvenlik Araştırması) ise saldırıların %60'ının kurum içi, %85'inin virüs kaynaklı, %95'inin ise yanlış planlama sebebiyle gerçekleştiğini açıklamıştır. Anti - Phishing Çalışma Grubu'na (APWG) göre de tanınmış bankalar, internet mağazaları ve kredi kartı şirketleri gibi güven duyulan kurumları kullanarak, sahte kimlikle (phishing) saldırıda bulunanlar, gönderdikleri e-postaları alanların yaklaşık %5'ini kişisel bilgilerini göndermeleri için kandırmayı başarmıştır. Anti-Phishing Çalışma Grubu her ay sahte kimlik saldırılarında bir önceki aya oranla %110'luk bir artış yaşandığını da açıklamıştır. Symantec ve Insight Express'in anket araştırmasında internet kullanıcılarının yaklaşık yarısının, %45.8'inin sanal dolandırıcılığa karşı nasıl önlem alacağı konusunda bir fikrinin olmadığı tespit edilmiştir. İnternet kullanıcılarının %75.7'sinin casus yazılım (spyware) uygulamalarının farkında olmasına karşın sadece %27.9'unun sahte kimlik saldırıları konusunda bilgi sahibi olması da yine aynı araştırmanın bir diğer önemli sonuçlarındandır. Ankete katılanların %43.5'i günde birkaç kez talep edilmediği halde gönderilen ve kişisel bilgilerini isteyen e-postalar aldığını, %44.2'si emin olmasa da sahte bir web sitesini ziyaret ettiğini, %19.3'ü ise sahte bir web sitesini kesinlikle ziyaret ettiğini söylemiştir. Ankete katılanların %74.3'ü sadece güvenli sitelerden ürün satın aldığını, %45'i internette kişisel gizli bilgileri paylaşmadığını, %31.5'i bankacılık işlemleri için interneti kullanmadığını ve %14.9'u internete güvenmediğini vurgulamışlardır [1].

Araştırmaların gösterdiği gibi aslında mevcut tehditler kişisel, kurumsal ve özellikle toplumsal olarak tahmin edildiğinden çok daha fazla zarar verme potansiyeline sahiptir. Bunun önüne geçmek için öncelikle bireysel ve toplumsal olarak farkındalık artırmalı sonrasında da geliştirilen yöntemlerle riskleri en az seviyeye getirmek hedeflenmelidir. Bireysel ve toplumsal olarak bilgi teknolojileri kullanarak hayatımızı kolaylaştırarak daha yaşanılabilir hale getirmek, gelişmek, bilgiye sahip olmak, kontrol etmek, kontrol altında tutmak, yönetmek, geleceği şekillendirmek, zenginleşmek, hâkimiyet kurmak amaçlanır.

Bilgi casusluğunun tarihsel sürecine baktığımızda, 1952'de Amerikan Ulusal Güvenlik Ajansı'nın (NSA) Amerikan karar vericilere ve askeri liderlere zamanında bilgi sağlamak için Başkan Truman tarafından kurulduğunu, 1960'da Rusya'ya iltica eden iki NSA görevlisi ABD'nin 40 ülkenin haberleşmesini dinlediğini açıklamıştır. Ayrıca ABD iç

savaşı sırasında General Stuart'ın telgraf hatlarını dinlettiği, 1974'te William Gibson ilk kez "cyber space" (siber alan) sözcüğünü kullandığı, 1992'de Pentagon Bilgi Savaşları konulu ilk "Top Secret" Yönergesi yayınlandığı, 1994'de ABD Haiti operasyonunda ağ saldırılarını kullandığı, 1995'te FBI'ın en çok arananlar listesine ilk kez bir hacker girdiği, 1997'de ABD Hava Kuvvetleri Bilgi Savaşları Laboratuvarı kurulduğu, 1997'ded ABD'nin ilk geniş çaplı siber saldırı tatbikatı yaptığı, 1998'de ABD Yugoslavya hava radarlarına sahte hedefler yerleştirdiği, 1998'de siber casus avı Operasyon Moonlight Maze'in başladığı, 2000'de Alman İçişleri Nazi sitelerine karşı DDOS saldırıları gerçekleştirebileceklerini söylediği ancak alınan tepki üzerine bu demecin geri çekilmek zorunda kaldığı, 2001'de AB, ABD'nin baskılarına rağmen GPS'e rakip Galileo'yu finanse etme kararı aldığı, 2002'de New York Devlet Üniversitesi ordu için beynine yerleştirilen devrelerle kontrol edilebilen canlı fare projesini açıkladığı bilinmektedir [1].

Ulusal ve uluslar arası düzeyde mevcut bu tehditlere karşı ülkelerin bazı önlemler alması gerekir. Bu önlemler şöyle sıralanabilir, ağ yapılarını ağ üst yapılarla kontrol ederek dağıtık fakat koordineli düzenleme ve kontrol noktaları oluşturulmalıdır bunun için sınıflandırılmış ve tanımlanmış Savunma ve Saldırı Politikaları ortaya konulmalıdır. Toplumsal ve yasal mutabakata varılarak Devlet-vatandaş bilgi paylaşımı güvenilir bir yapıda olmalıdır, merkezi ve/veya yasakçı yapılar yerine dağıtık ve otokontrollü yapılar kurulmalıdır çünkü vatandaş devleti kadar güvenli, devlet vatandaşı kadar özgürdür. Bunlar için de öncelikle kritik alt yapı tanımlanmalı ve yerli teknoloji zorunluluğu göz önünde bulundurulmalıdır. Alt yapılar arası koordinasyonu sağlayacak üst kurul ile sektörler arası bilgi paylaşımına önem verilmelidir. Burada iş birliği ve profesyonel kadro yönetimi ihtiyaç duyulan özelliklerdir [1]. Yerli teknolojileri kullanmada, Çin Halk Cumhuriyeti yerli arama motoru kullanma zorunluluğu getirerek konunun önemine de dikkat çekmektedir. Arama motorlarından yapılan analizlerde toplumun her türlü sosyal ve psikolojik eğilimleri tespit edilebilmekte hangi olaylara nasıl tepkiler verebileceği sosyal mühendislik ile önceden belirlenebilmektedir. Bunun çok ciddi bir tehdit olarak düşman ülkeler tarafından kullanılabileceği unutulmamalıdır. Tüm bunlardan dolayı sahip olunan bilgi korunmalı ve sanal savaş kuralları bilinmeli ve hazırlıklar yapılmalıdır.

Gerek bireysel gerekse kurumsal olarak yukarıda bahsedilen tehditlere karşı saldırıların hangi açıklardan gelebileceği belirlendikten sonra güvenliğin korunması için yapılması gerekenler üzerinde durulmalıdır. Bunu sağlamak için kurumların risk durumlarını ve bu durumlarda uygulanacak eylem politikalarını belirlemeleri gerekmektedir. Bilgi güvenliği

konusunda ISO 27001 ve CMMI gibi standartları gözetmeleri önerilmektedir. Bireysel ve kurumsal güvenlik için kullanılan şifre doğrulama ve gizliliği, veri doğrulama, kimlik doğrulama ve inkar edememe protokolleri, elektronik imza, veri içine veri gömme gibi uygulamalar için şifreleme ve özetleme fonksiyonları kullanılmalıdır. Bu bölümde son olarak Modern Kriptoloji (şifreleme) kısaca açıklanarak temel nitelikleri üzerinde durulacaktır. Bir sonraki bölümde geliştirilen algoritma kaos tabanlı olduğu için Kaos Teoremi hakkında bilgi verilecektir.

1.2. Modern Kriptoloji

Kriptoloji, iletişim halindeki iki veya daha çok tarafın bilgi alışverişini güvenli olarak yapmasını sağlayan, temeli matematiksel denklemlere dayanan yöntemlerin ve uygulamaların bütününe verilen genel bir addır. Kriptoloji günümüzde bir bilim dalı olarak kabul edilmektedir. Bu alanda matematik, elektronik, bilgisayar bilimleri, optik gibi birçok disiplini kullanan özelleşmiş bir daldır. Kriptolojinin Kriptografi ve Kriptoanaliz olmak üzere iki temel konusu vardır. Kriptografi, verilerin şifrelenmesi (encryption) ve şifresinin çözülmesi (decryption) için kullanılan metodlara verilen addır. Kriptoanaliz, kriptografik sistemlerin kurduğu yapıları inceler ve çözmeye çalışır. Oluşturulan bir şifreleme sistemini inceleyerek, zayıf ve kuvvetli yönlerini ortaya koymak için kripto analiz kullanılır. Haberleşmede iki tarafın da emniyetli bir şekilde veri alışverişinde bulunması için gerekli temel öğeler aşağıda listelenmiştir.

1. Gizlilik (Confidentiality) : Taşınan bilgi içeriğinin gizli kalmasıdır. Bilgiye erişim yetkisi olan taraf yetkisi kapsamında erişilebilir. Pratikte bu mühürlü zarflarla sağlanır.
2. Bütünlük (Integrity) : Taşınan bilginin içeriğinin paket alışverişi sırasında değiştirilmemesidir. Araya girip değişiklik yapma, zarar verme, veri taklidi, veri ekleme saldırılarına karşı doğruluk, değiştirilememe veya sadece daha önce belirlenmiş kurallar kapsamında yetkili kişiler ve işlemlerle değiştirilebilme, tutarlı ve anlamlı olma özelliklerinin gerekliliğini esas alır. Pratikte bu özellik imza veya damga ile sağlanır.
3. Kimlik Doğrulama (Authentication) : Bilgiyi gönderen tarafın kimliğinin doğruluğundan emin olmaktır. Okuma ve yazma işlemlerinde doğru kişi

tarafından yapıldığından emin olma, erişilebilirlik ve kullanılabilirlik özelliklerinin gerekliliğini esas alır. Pratikte bu özellik noter, kimlik kartı ile sağlanır.

4. İnkâr Edememe (Nonrepudation) : Bilgiyi gönderen veya işlemi yapan kişinin yaptığı işi sonradan inkâr edememesidir. Günlük hayatta bu özellik için imza, alındı ve onay metotları kullanılır.
5. Haberleşmenin Sürekliliği (Contunity of Communication) : İletişimin kesintiye uğramadan yapılabilmesidir. Sisteme saldıranların haberleşen iki taraf arasındaki hattı veya haberleşme cihazlarını kullanılmaz hale getirerek haberleşmenin sürekliliğini engellemeye çalışması durumunda haberleşmenin sürekliliği sağlanamayabilir. Pratikte birbirine alternatif iletişim yolları sunarak ve haberleşme cihazları yüksek seviyede korunarak bu özellik sağlanır.
6. Erişim Denetimi (Access Control) : Bir sistemde hangi kullanıcının hangi işlemleri yapmada yetkilerinin olduğunun net bir şekilde tanımlanarak yetkisiz erişimleri kontrol eden, kayıt altına alan bir üst sistemin gerekliliğini ifade eder.
7. Erişilebilirlik (Availability) : Bir sistemin kurulma amacını gerçekleştirebilmek için gerektiğinde erişilebilmeyi esas alır. [1,2]

Bu unsurlardan modern kriptoloji ile gizlilik için şifreleme, bütünlük için özetleme, kimlik doğrulama ve inkâr edememe için elektronik imza yöntemleri kullanılabilir.

Modern kriptoloji başlığı altında PGP (Pretty Good Privacy)'ye bakıldığında, PGP'nin e-postalarda kişisel gizliliği ve kimlik doğrulamayı sağlamak için e-postaları şifreleyen program olduğu görülmektedir, 1991 yılında insan hakları aracı olarak Phillip Zimmerman tarafından internette ücretsiz olarak yayınlanmıştır. Daha sonra Internet Engineering Task Force (IETF) tarafından PGP'ye dayanan bir internet standardı Open PGP geliştirilmiştir. Kullanıcı bir düz metni PGP ile şifrelediğinde, PGP öncelikle düz metni sıkıştırır. Veri sıkıştırma iletim zamanından ve disk boşluğundan kazandırır ve daha da önemlisi kriptografik güvenliği artırır. Birçok kripto analiz tekniği şifreyi kırmak için düz metin içindeki desenleri kullanır. Sıkıştırma bu tür desenleri azaltır ve bu şekilde kripto analize karşı daha dayanıklı hale getirilir. [1,3]

Geleceğin teknolojilerinden DNA bilgisayarlar, Kuantum bilgisayarların şifreleme ve şifre çözmeye katkıları ile meydana getireceği tehlikeler ise hala tartışma konusudur.

1.3. Şifreleme ve Özellikleri

Şifreleme, bir bilginin belirli özel yöntemler kullanarak değiştirilmesi ve farklı bir şekle sokulması olarak tanımlanabilir. Şifreleme sonucunda ortaya çıkan yeni şekildeki bilgi, şifre çözme işlemi uygulanarak başlangıçtaki haline dönüştürülebilir. Bir şifreleme yönteminden aşağıdaki yeterliliklere sahip olması beklenir:

1. Şifreleme ve şifre çözme işleminin maliyeti ihtiyaç duyulan güvenliğin getirdiği yarar ile orantılı olmalıdır. Önemli olmayan bir verinin şifrelenmesi için çok fazla işgücü ve zaman harcanması verimli olmayacaktır. Ayrıca uygulanabilir olmalıdır.
2. Şifreleme metodu her tür bilgi çeşidi için aynı şekilde çalışmalıdır.
3. Süreç mümkün olduğunca yalın halde olmalıdır. Çok fazla karışık bir sistemin gerçeklenmesi hem pratiğe dönüştürmede zor olabilir hem de performans açısından tatmin edici olmayabilir.
4. Şifrelemede bir kısımda yapılan hatalar sonraki adımlara yansıtılmamalı ve mesajın tamamını bozmamalıdır. Saldırılara karşı bu nitelik koruyucu olarak öngörülmektedir. Aksi durumda haberleşme hattında meydana gelen bir hata tüm mesajın bozulmasına neden olabilir.
5. Şifreli mesaj ile açık mesaj arasında ilişki kurulması çok zor olmalıdır.
6. Mesajın açık hali şifrelenirken içerdiği alt dizinler şifreli mesajın içinde olabildiğince dağıtılmalıdır.
7. İhtiyaç duyulan güvenlik seviyesine göre güvenlik seviyesi seçilebilmelidir.
8. Şifreleme yaklaşımları açıkların tespit edilebilmesi için mümkün olduğunca geniş bir ortamda test edilmelidir.

Şifreleme algoritmalarını algoritma tipine, mesaj uzunluğuna ve anahtar sayısına göre gruplandırılabilir. Algoritma tekniğine göre gizli ve açık algoritmalı şifreleme algoritmaları olarak sınıflandırılabilir.

Gizli algoritmalı şifreleme tekniklerinde şifreleme ve şifre çözme algoritması birbirinin tersidir, haberleşecek iki grup arasında gizli bir algoritma tespit edilerek güvenli bir iletişim kanalı ve güvenli taşıyıcılarla mesajlaşırlar. Kullanılan algoritmanın gizli kalmasını sağlamak zor olduğu için sınırlı kullanıma sahiptirler, kullanımı ve yaygınlaşması zordur. Şifreleme algoritmasının standart hale gelmesi için herkes tarafından bilinmesi gerekir bu da açık şifreleme algoritmalı teknikleri doğurmuştur. Açık şifreleme

algoritmalı tekniklerde P mesajı gönderen ile alanın bildiği K anahtarı ile şifrelenir. Modern şifreleme teknikleri anahtar alt yapısına dayanan açık algoritmalı sistemlerdir. Anahtar tipine göre şifreleme algoritmalarını simetrik ve asimetrik olarak ayırabiliriz. Kullanıcı bir mesajı göndermeden önce k1 anahtarı ile şifreler ve açık bir kanaldan gönderir. Mesajı okumak için alıcı k2 anahtarını kullanarak şifreyi çözer ve mesajı elde eder. Bu iki k1 ve k2 anahtarı aynı ise sistem simetriktir, eğer farklılarsa sistem asimetriktir. DES, 3DES, DESX, IDEA, XOR, SKIPJACK, RC2, RC4, RC5 bilinen simetrik şifreleme algoritmalarındandır. Asimetrik algoritmalı şifreleme tekniklerinde tek yönlü bir mesajlaşma söz konusudur. Mesaj sadece kendi bileceği bir gizli anahtar ve diğer kişiler için duyuracağı bir açık anahtar belirler. Mesaj göndericisi alıcıya ait herkes tarafından bilinen açık anahtarı kullanarak mesajı şifreler ve alıcıya gönderir, mesajı alıcı kendi gizli anahtarı ile deşifre eder. Asimetrik anahtarlı şifreleme, simetrik anahtarlı şifrelemeden daha uzun zaman almaktadır. Bu zaman, mesajın büyüklüğüne ve anahtara bağlı olarak üssel olarak artar ve kırılması daha zor olur.

Mesaj tipine göre şifreleme algoritmalarını dizi şifreleyiciler ve blok şifreleyiciler olarak ayrılabilmektedir. Dizi şifreleyicilerde algoritmanın girdisi anahtardır, algoritma anahtardan rastgele olarak bir diziye çok benzeyen kayan anahtar dizisi üretir sonrasında kayan anahtar dizisi elemanları ile açık metin veya şifreli metin dizisinin elemanları ikili tabanda özel OR işlemine tabi tutularak (XOR) şifreleme ve şifre çözme işlemi tamamlanır. Blok şifreleyiciler ise şifreleme ve şifre çözme işleminde metin sabit uzunluktaki dizilere bölünüp blok temelli şifrelemeye tabi tutulurlar [1,2].

Bilgi güvenliği ve kriptolojinin inceleneceği bu bölümün ardından ikinci bölümde kaos teoremi, üçüncü bölümde kriptografik özetleme algoritmaları, dördüncü bölümde kaotik özetleme algoritmaları, beşinci bölümde geliştirilen İkeda haritası tabanlı kaotik tabanlı özetleme algoritması ve testleri bulunmaktadır. Altıncı bölümde ise sonuçlar değerlendirilecektir.

2. KAOS TEOREMİ

Bu bölümde genel olarak kaotik fonksiyonlar ve farklı kullanım yöntemleri anlatılacaktır. Ayrıca özetleme algoritmalarıyla birlikte nasıl kullanıldıkları hakkında bilgi verilecektir.

2.1. Kaos Teoremi Tarihçesi

Kaos teorisi uygulamalarda fizik, ekonomi, biyoloji ve sosyolojiyi kapsayan matematiğin bir dalıdır. Kaos teoreminde dinamik sistemlerin davranışı başlangıç koşullarına çok fazla duyarlıdır, bu kelebek etkisi olarak da bilinir. Başlangıç şartlarındaki ufak değişiklikler örneğin sayısal hesaplamalardaki yuvarlamalara bağlı olarak değişen durumlarda kaotik sistemler için çok farklı çıktılar ortaya çıkarabilir bu da genelde tahmin edilmesi imkânsız durumlardır. Bir sistem eğer kaotik değilse birbirine çok yakın iki başlangıç noktasından başlatıldığında, sonuçlarındaki farklılık zamanla doğrusal olarak artar. Eğer kaotik bir sistemse bu fark üstel olarak artar. Bu davranış belirlenimci (deterministik) kaos veya kısaca kaos olarak bilinir.

Sistemin kaotik olup olmadığına bakabilmemiz için iki koşul bulunmaktadır, birincisi sistemin doğrusal olmayan bir sistem olması gerekliliğidir. İkinci koşul da sistemin derecesiyle ilgili olarak eğer sürekli zamanlı bir sistemse en az üçüncü derece bir sistem olmalıdır, ayrık zamanlı bir sistem ise birinci derecede de olabilmektedir. Bu şartlar kaos olabileceğini ifade eder ancak garanti etmez. Belirlenimci bir sistemde bulunduğunuz durumdan bir sonraki duruma geçiş için gerekli şartlar bellidir, hangi şartlarda hangi durumlara sistemin geçeceği açıktır. Bir belirlenimci sistemde rastgele olaylara benzer durumlarla karşılaşılması ise kaosu ifade eder.[4]

Kaosu ilk ifade edilişi ise 1890 yılına kadar dayanmaktadır. Güneş-dünya-ay cisimlerinin hareketini hesaplayabilmek için bilim adamlarının çalıştığı ancak çözüm üretemediği konuya Henry Poincare probleme niteliksel olarak yaklaşmıştır ve gezegenlerinin konumlarıyla ilgilenmek yerine, güneş sisteminin davranışlarını kararlı şekilde sürdürüp sürdürmediğine odaklanmıştır. Bu noktada geliştirdiği yöntem birçok fiziksel probleme uygulanabilir nitelikte geometrik bir yaklaşımdır.

Poincare'den sonra uzun bir süre kaos hak ettiği ilgili görememiştir, genelde oluşan rastgele durumlar gürültü olarak nitelendirilmiştir. 1963 yılında ise Edward Lorenz bilgisayar ile kaos üzerinde ilk sayısal çalışmayı yapan kişi olarak bilinmektedir. Kendisi MIT'de çalışan bir meteorolog ve hava durumu tahminlerini küçük bir modellemesini yaptığı atmosfer üzerinden yapmaya çalışmaktaydı. Oluşturduğu modelin bilgisayarda benzetimini (simülasyonunu) yaparak çıkartmıştır ve belirli parametreler vererek elde ettiği sonuçların periyodik bir yörüngede hareket etmediğini ve düzensiz olduğunu gözlemlemiştir. Sonrasında aynı deneyi birbirine çok yakın başlangıç koşullarına sahip iki farklı durum için tekrarladığında deney sonuçlarının birbirinden çok farklı olduğunu görmüştür [5].

2.2. Kaos Analizi

Kaotik davranış hava olayları gibi diğer doğa olaylarında da sıkça görülebilmektedir. Bu şekilde davranışının açıklamasını sistemin kaotik olup olmadığına bakarak yapılabilmektedir bu da farklı yöntemler kullanılarak yapılabilmektedir. Bu yöntemlerden en çok kullanılanları, zaman serileri, faz uzayı, Lyapunov üstelleri, Poincare haritaları, güç spektrumu ve çatallaşma diyagramıdır. Bu yöntemler aşağıda kısaca açıklanmıştır.

1. Yörüngenin izlenmesi yönteminde sistemin durum değişkenlerinden biri zaman ekseninde izlenerek kaosa girip girmediği tespit edilebilir. Sistemin izlendiği durumun geçici durum davranışı kalktıktan sonra olmasına dikkat etmek gerekmektedir, bu diğer analiz yöntemleri için de geçerlidir.
2. Faz uzayı yönteminde durum değişkenlerinin birbirlerine göre durumları geometrik olarak çizdirilerek sistemin kaosa girip girmediğine bakılmaktadır. Limit çevrimi de denilen (limit cycle) durumda, değişkenlerin birbirine göre yeri çizildiğinde kapalı bir çevrim olarak görünür. Bu sistemin periyodik olduğunu gösterir. Bu çizim sonucu karışık ve anlamsız ise bu tip karışık şekiller kaosun varlığını işaret eder. Faz düzlemi ifadesini biraz genişletirsek; doğrusal veya doğrusal olmayan türden denklem 2.1.'deki denklem teorisine göre;

$$u'' = f(t,u,u') \tag{2.1}$$

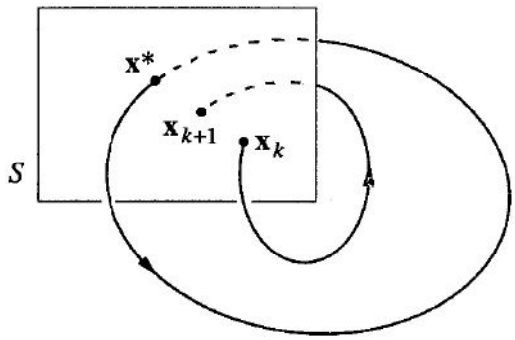
Burada u(t) çoğu kez u ekseninde hareket eden bir noktanın t anındaki yerini gösterir ve denklem 2.2'ye göre;

$$v(t)=u'(t) \qquad (2.2)$$

Burada da t anındaki hızını tanımlar. (u(t),v(t)) ifadesi ise sistemin t anındaki durumunu gösterir. Sistemin nasıl davranacağı, (u,v) düzleminde (u(t),v(t)) noktasının geometrik yeri ile ifade edilebilir. Bu şekilde diferansiyel denklem ile ilişkilendirilen (u,v) düzlemi fazdüzlemi olarak adlandırılır. (u(t),v(t)) parametrik çözüm eğrisine yörünge ve onun görüntüsüne de orbit veya iz denir. Bir yörünge ile orbit arasındaki ilişki ise yörüngenin çözüm eğrisinin oryantasyonunu veren t parametresi ile donatılmış olmasıdır [6].

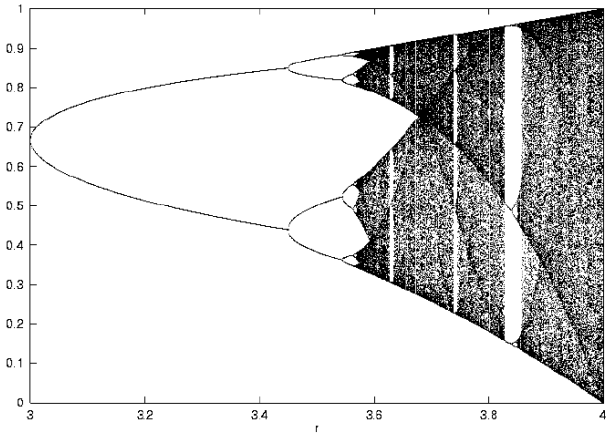
3. Poincare haritalama yönteminde eğer davranış periyodikse Poincare haritalama yönteminde sabit bir nokta elde edilir, eğer sistem periyodumsuysa kapalı bir çevrim ortaya çıkar. Sistem kaotik ise de kapalı olmayan rastgele bir fraktal şekil olur. Bu yöntem n. dereceden sürekli zamanlı bir sistemin (n-1). dereceden ayrık bir sistem olarak ifade edilmesine karşılık gelir. Faz uzayında bir yüzey seçilerek bu yüzeyde yörüngenin geçtiği noktalar işaretlenerek bir harita elde edilir. Faz uzayı izlenirken belirli periyotlarda örnekler alınarak Poincare harita çıkartılır. Karmaşık sistemleri daha basit şekilde ifade etmek için de kullanılır. Poincare haritasının matematiksel tanımı aşağıdaki gibidir.
    - (R,M,ß) dinamik bir sistemi ifade etsin, R real sayılar, M faz düzlemini ve ß da geliştirme fonksiyonu olsun. α da p noktasından periyodik orbit olsun, S de Poincare bölgesi olarak isimlendirilir, yerel olarak ayırtedilebilirdir ve ß nin p vasıtasıyla enine ayrılmıştır.
    - P : U→ S Poincare haritasının α izi için S Poincare bölgesinde p noktası vasıtasıyla tanımlanması için;
    - P(p) = p ; P(U) p nin komşuluğunda ve P:U→P(U) bir tersiyle ayırt edilebilen (diffeomorphism) olmalıdır.

- U içindeki her bir x noktası için, x in pozitif yarı-izi (positive semi-orbit) S'i P(x)'in ilk durumuna göre böler [7]. Poincare haritası bir örneği Şekil 2.1. 'de gösterilmiştir.



Şekil 2.1 Poincare harita örneği

4. Güç spektrumuna bakarak kaotik analiz yönteminde sinyalin güç spektrumundaki haline bakarak karar verilir, kaotik sinyaller geniş bantlı sinyaller olduğu için güç spektrumunda süreklilik hali gözlenir. Buna karşılık sistem kaotik değilse sadece belirli frekanslarda sıçrama gözlenir.



Şekil 2.2 Çatallaşma diyagramı örneği

5. Çatallaşma diyagramı yönteminde oluşan grafiğin y ekseninde sistemin asimptotik çözümleri, x ekseninde ise kontrol parametresinin değişim değerleri bulunur. Bu çizimlere çatallaşma diyagramı denilmektedir. Şekil 2.2'de görünen resimde örnek bir diyagram bulunmaktadır. Burada görülen noktalardan r=3.2 için düşey eksene paralel bir kesit yaptığımızda iki çizgi ile kesiştiğini görünmektedir bu çift periyotlu olduğunu ifade etmektedir. Sistemde r=3.6'dan sonra kaos görüldüğü, r=4'ten sonra ise sistemin tamamen kaosa gittiği görünmektedir. Kesişen çizgilerin artışı kaosun varlığına işaret etmektedir.

2.3.Kaotik Fonksiyonlarının Özetleme Algoritmalarında Kullanılması

Geleneksel özetleme algoritmaları olarak bilinen MD5, SHA-1 ve SHA-2 algoritmalarında metnin özetini alırken metin belli uzunluklarda bloklara ayrıldıktan sonra mantıksal ve matemetiksel fonksiyonlar kullanılarak o metne özgü özet ifade oluşturulmaktadır. Bu aşamada mantıksal ve matematiksel yöntemlerle oluşturulmaya çalışan karmaşıklık ve haritalama işlemi için kaotik fonksiyonlar kullanılması; başlangıç şartlarındaki küçük değişikliklere göre sonucunda büyük değişiklikler meydana geldiği için tercih edilmiştir. Kaotik fonksiyonların bu özelliğinin özetleme algoritmaları için büyük bir avantaj sağlayacak olması bu alanda yapılan çalışmaları hızlandırmıştır [8,9]. İncelenen çalışmalarda kaotik fonksiyonların özetleme algoritması inşa ederken farklı şekillerde kullanıldığı görülmüştür. Genelde kaotik fonksiyonun çalıştırılacağı iterasyon sayısı bloklara ayrılan metinden elde edilmiştir, ayrıca fonksiyonun başlangıç değerleri da blokların işlenmesi sonucunda hesaplanmıştır. Çoğu çalışmada kaotik fonksiyon olarak tek boyutlu tent harita, iki boyutlu cat harita gibi fonksiyonlar [10,11] kullanılmıştır. Bu tez çalışmasında çok boyutlu ikeda harita kullanılarak özetleme algoritması oluşturulmuş ve analiz edilmiştir. Sonraki bölümlerde kaotik özetleme algoritmaları detaylı şekilde incelenecektir.

3. KRİPTOGRAFİK ÖZETLEME ALGORİTMALARI

Kriptografik özetleme (cryptographic hash) fonksiyonları veriyi bloklara ayırarak belli işlemlerden geçirir, tek yönlü (one-way) olan bu fonksiyonlar algoritmanın özelliğine göre değişen sabit uzunlukta kriptografik özet çıkartır. Bakıldığında metinle ilişki kurulamayana, anlam bütünlüğü içermeyen ve rastgele seçilmiş sayılar gibi görünen bu çıktı o dosya veya bilgiye özeldir. Özetleme işlemi her tekrarlandığında aynı sonucu verir. Ancak dosya veya bilgide 1 bit bile değişiklik olması durumunda özet bilgisi değişir. Böylece veri bütünlüğünün bozulmasından endişe edilen iletişimlerde kontrol sağlar. Girilen verinin büyüklüğü ne kadar olursa olsun özet bilginin uzunluğu değişmez, özet fonksiyonundan giriş verisine ulaşılması veya özet fonksiyondan giriş verisi ile ilgili bilgi edinme teorik olarak imkânsızdır. Bir mesajın her durumda özeti aynı olmalıdır ve tabi ki farklı mesajların özetleri de farklı olmalıdır. İki farklı mesajın aynı özeti vermesi özetleme algoritmaları için çakışma (collision) olarak tanımlanır. Kriptografik özetleme fonksiyonları başta elektronik imza, mesaj doğrulama kodları (MAC) ve kimlik doğrulama formları gibi bir çok bilgi güvenliği uygulamasında kullanıldığından modern kriptolojide önemli bir rol oynamaktadır. Bu fonksiyonlar özet tablolarda veriyi indeksleyen basit bir özetleme fonksiyonu olarak kullanıldığı gibi çift kayıtları veya aynı isimli dosya tanımlamalarını engellemek için ya da veri bozulmalarını tespit edilmesi için kullanılabilir.

3.1. Kriptografik Özetleme Algoritmalarının Özellikleri

Daha önceki bölümlerde modern güvenlik, şifreleme ve şifrelemede özetleme algoritmalarının önemi incelenmiştir. Bu bölümde ise özetleme algoritmalarının özelliklerinden bahsedilecektir. Özetleme algoritmaları değişken uzunlukta girdileri bölüp bloklarda ardışık olarak bir takım fonksiyonlardan geçirerek sabit uzunlukta çıktılar üreten algoritmalardır. Üretilen bu çıktılar özet olarak isimlendirilecektir. Tek yönlü özetleme algoritması olarak bilinen MD5 ve SHA aileleri gibi algoritmalarda her girdi için yalnızca çıktı vardır ve giriş verisinde 1 bitlik değişim sonucu çıkış verisini ilişki kurulamayacak şekilde değiştirir. Tek yönlü denilmesinin nedeni özet veriden mesajı elde etmenin mümkün olmayışıdır. Özetleme fonksiyonlarının amacı bir dosyanın, mesajın veya bir veri

bloğuna ait ayırt edici özet bilgi (finger print) üretmektir.

M uzunluklu giriş verisini H(M) fonksiyonu ile h özet verisi haline getirecek olan özetleme algoritmasının taşıması gereken özellikler şöyledir:

1. H fonksiyonu her uzunlukta veri bloğuna uygulanabilmelidir.
2. H sabit uzunluklu bir çıkış üretmelidir.
3. H(x) verilen herhangi bir x için pratikte hem donanımsal hem de yazılımsal olarak kolaylıkla uygulanabilir nitelikte olmalıdır.
4. Bilinen herhangi bir h için, H(x) = h olan herhangi bir x değerini hesaplayabilmek mümkün olmamalıdır. Tek yönlü (One-way) algoritmalar denilmesinin nedeni budur.
5. Bilinen herhangi bir x bloğu için, y ile x eşit olmadığı müddetçe H(y) = H(x) eşitliğini sağlayan y değerinin bulunması mümkün olmamalıdır. Bu özellik literatürde zayıf çakışma direnci (weak collision resistance) olarak geçer.
6. H(x) = H(y) olacak (x, y) ikilisinin bulunması mümkün olmamalıdır. Bu özellik literatürde güçlü çakışma direnci (strong collision resistance) olarak geçer.
7. İlişki kurulamama (non-corelation) olarak bilinen özellik x giriş mesajı için elde edilen s özeti arasında herhangi bir şekilde ilişkilendirme olmamasını kapsar. Giriş mesajının her bir biti s özetinin tüm bitlerini etkilemelidir.
8. Yakın çakışma (near-collision) olarak bilinen özellik, özetleri birbirine çok benzer olan birbirinden farklı x ve x' mesajlarını bulmak mümkün olmaması özelliğidir.
9. Kısmi ön-imge direnci (partial-preimage resistance) olarak bilinen özellik çıkış dizisi için giriş dizisinin alt dizilerini bulmak mümkün olmaması özelliğidir [12].

Literatür taramalarında nitelenen bu özelliklerin farklı adlarla da kullanıldığı göz önünde tutulmalıdır. Dördüncü özellik ön-imge direnci (pre-image/pre image resistance), beşinci özellik de ikincil ön-imge direnci (second pre-image/preimage resistance) olarak anılır.

Sayılan bu özelliklerden ilk üçü pratikte mesaj doğrulatma özetleme algoritmalarının kullanıldığı durumlar için gereklidir. Dördüncü özellikte bir mesajın özetini elde etmenin kolay fakat verilen özetten mesajı elde etmek imkânsız olduğu nitelenmiştir. Beşinci özellik verilen mesajla aynı özet çıktısına sahip başka bir mesajın bulunamayacağını garanti eder. Sisteme saldıran kişi mesajı okuyup özet verisini elde etse bile gizli anahtara sahip olmadığından mesajda değişiklik yapamaz. Altıncı özellik doğum günü paradoksu olarak

bilinen saldırıya karşı özetleme algoritmalarının nasıl karşı koyduğuyla ilgilidir. Doğum günü paradoksu 23 kişilik bir toplulukta aynı doğum gününe sahip iki kişinin olma olasılığının %50'nin üzerinde olmasını ifade eder çünkü bu oran beklenenin çok fazla üstündedir. 23 kişiyi 253 (22+21+20+..1 = 253) farklı şekilde eşleştirebileceğimiz için oran da %50'nin üstünde çıkmaktadır [13].

3.2. Kriptografik Özetleme Algoritmalarının Genel Prensipleri

Tüm özetleme fonksiyonları şu genel prensiplere göre hareket ederek işlevlerini yerine getirirler. Giriş verisine ön işlemler uygulanır, n bitlik bloklar halinde okunur, her blok bir iterasyon zamanda işlenerek sonunda birleştirilir ve n bitlik özet kodu elde edilir. İşleyişin şeması Şekil 3.1'deki gibidir [14].

Bu şekilde, ön işlemler M mesajının sonraki bölümlerde işlenebilmesi için gerekli düzene getirecek işlemlerdir. Ön işlemlerde belirli bir bit sayısının tam katına sahip bloklar haline getirilen mesaj parçalara ayrılarak iteratif yapı içinde döngüsel işlemlerde matematiksel işlemlere tabi tutulur. Sonlandırma işleminde ise iteratif işlemler sonucu yapılan dönüşümlerden sonra mesaj dizisinin istenilen özetleme uzunluğuna getirilir.

En basit özetleme fonksiyonlarından biri, her bloğu bit bit XOR işlemine tabi tutmaktır. Eşitlik 3.1'deki gibi açıklanabilir:

$$h_i = b_{i1} \oplus b_{i2} \oplus \ldots \oplus b_{im} \tag{3.1}$$

Burada,

h_i: $1 \leq i \leq n$ olmak üzere özet verinin i. bitini

m: Girişteki n bitlik blokların sayısını

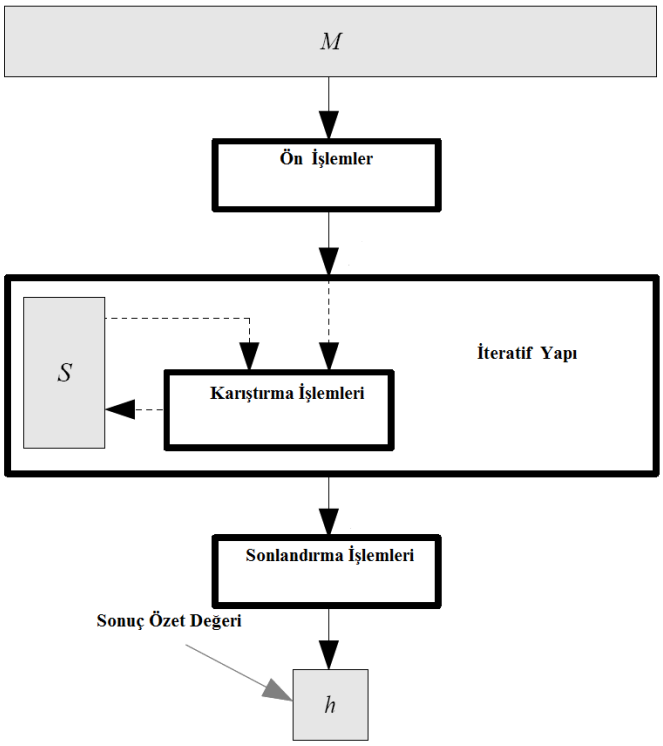
b_{ij}: j. bloktaki i. biti

$\oplus$: XOR işlemini ifade eder.

XOR mantıksal fonksiyonunun doğruluk tablosu Tablo 3.1'deki gibidir.

Bu işlem her bir bitin pozisyonu için basit bir eşlik (parity) biti üretir, bu dikey artıklık kontrolü olarak bilinir ve rastgele üretilen verinin doğruluk kontrolü için uygundur. Veride

olması muhtemel hatalar özet veride de değişikliğe sebep olacaktır. Örneğin normal bir metin dosyasında, her bir sekizliğin (octet) yüksek öncelikli biti her zaman sıfırdır. Eğer 128 bit özet değer kullanılırsa, 2^{128} etkililik yerine bu tipteki verinin özetleme fonksiyonunun etkinliği 2^{112} olmaktadır.



Şekil 3.1 Özetleme algoritmaları genel işleyiş şeması

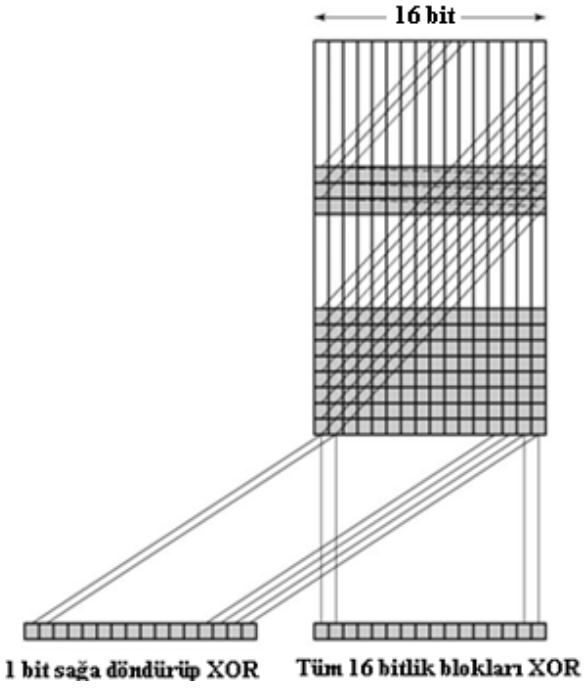
Tablo 3.1 XOR doğruluk tablosu

Giriş-1	Giriş-2	Çıkış
0	0	0
0	1	1
1	0	1
1	1	0

Bir diğer basit özetleme fonksiyonunda ise sırasıyla şu işlemler yapılır; n bit özet değeri sıfırlanır, verinin n bitlik bloğu için şu iki adım tekrarlanır sırasıyla geçerli özet verisi bir bit sola kaydırılır ve özet veri içindeki blok XOR işlemine tabi tutulur. Girişte herhangi bir kuralın olmamasını daha fazla sağlamak için rastgele olmasını sağlar. Bu iki işlemin gösterildiği şema 16 bit için Şekil 3.2'de görünmektedir. Bu tekniğin orijinalinde Ulusal Standart ve Teknoloji Enstitüsü (NIST) tarafından mesajın 64 bitlik bloklarına XOR uygulanmış daha sonra giriş verisi CBC (Cipher Block Chaining) blok zincirlemeli şifreleme kullanılarak şifrelenmiş. İşlem adımlarını şu şekilde sıralayabiliriz: Verilen mesaj X_1, X_2,..., X_N ardışık 64 bitlik bloklar içerir, C özet çıktısını blok blok tüm blokların XOR işlemine tutulmasıyla tanımlanır, son blokta özet veri 3.2'deki bağıntı ile elde edilir.

$$C = X_{N+1} = X_1 \oplus X_2 \oplus \ldots \oplus X_N \tag{3.2}$$

Daha sonra giriş verisine özet veri eklenerek şifrelenir, Y_1, Y_2,...,Y_{N+1} şifreli mesajı CBC kullanarak üretilir. CBC şeması Şekil 3.3' teki gibidir.



Şekil 3.2 Temel özetleme algoritmaları örneği

Şekil 3.1 ’deki eşitlikler :

$$X_1 = IV \oplus D(K, Y_1) \tag{3.3}$$

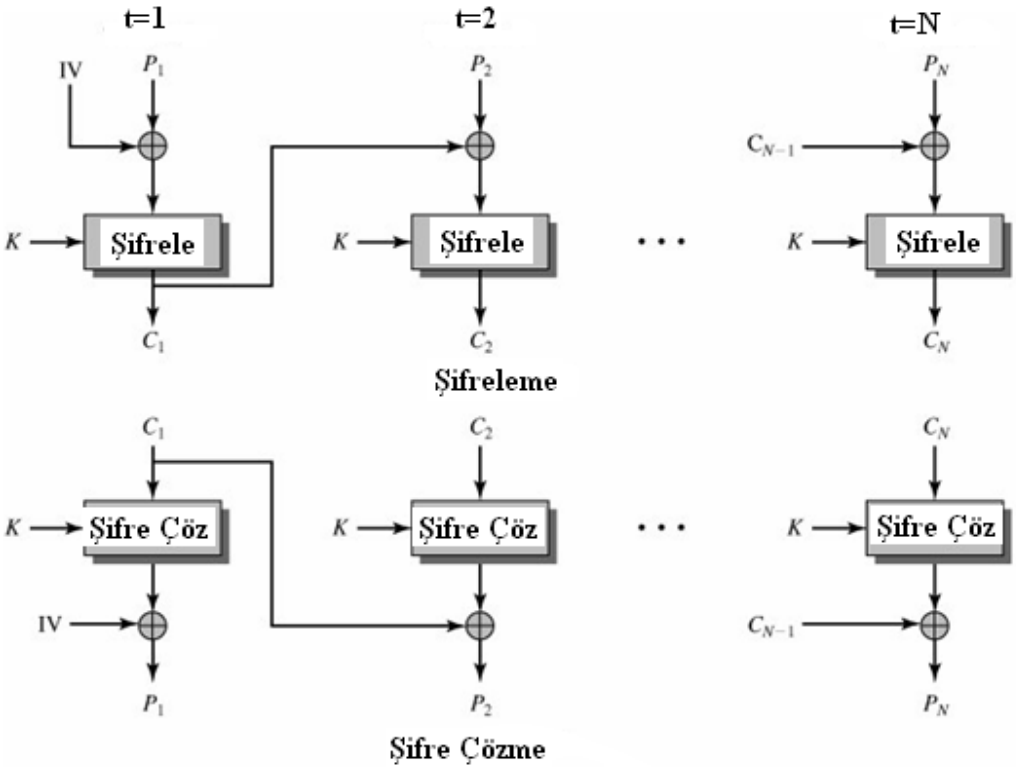
$$X_i = Y_{i1} \oplus D(K, Y_i) \tag{3.4}$$

$$X_{N+1} = Y_N \oplus D(K, Y_{N+1}) \tag{3.5}$$

$$X_{N+1} = X_1 \oplus X_2 \oplus \dots \oplus X_N \tag{3.6}$$

$$= [\, IV \oplus D(K, Y_1)\,] \oplus [Y_1 \oplus D(K, Y_2)] \oplus \dots \oplus [Y_{N1} \oplus \dots \oplus D(K, Y_N)] \tag{3.7}$$

olarak ifade edilir.



Şekil 3.3 Blok zincirlemeli şifreleme ve şifre çözme (CBC)

Şekil 3.3.'teki eşitlikler ise

$$C_j = E(K, [C_{j-1} \oplus P_j]) \quad (3.8)$$

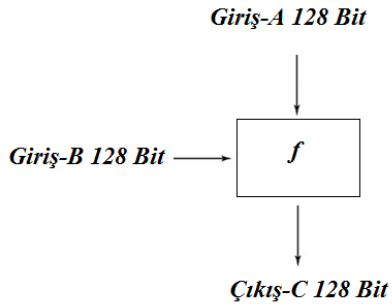
$$D(K, C_j) = D(K, E(K, [C_{j-1} \oplus P_j])) \quad (3.9)$$

$$D(K, C_j) = C_{j-1} \oplus P_j \quad (3.10)$$

$$C_{j-1} \oplus D(K, C_j) = C_{j-1} \oplus C_{j-1} \oplus P_j = P_j \quad (3.11)$$

olarak ifade edilir [13].

Burada özetleme algoritmaları içinde kullanılan tek yönlü sıkıştırma algoritmalarından bahsedilecektir, özetleme şeması (hashing scheme) olarak da bilinen yapıların özelliği iki sabit uzunluklu girişi girişlerden birinin uzunluğunda bir çıkışa dönüştürür, tek yönlü sıkıştırma fonksiyonları veri sıkıştırma ile ilgili değildir. Örnek bir tek yönlü sıkıştırma fonksiyonunu Şekil 3.4.'te bulunmaktadır.



Şekil 3.4 Tek yönlü sıkıştırma fonksiyonu genel şeması

Tek yönlü sıkıştırma fonksiyonları Merkle-Damgard iç yapısında olduğu gibi kriptografik özetleme algoritmalarında kullanılırlar. Blok şifreleyiciler genellikle tek yönlü sıkıştırma algoritmalarından oluşur. Blok şifrelemede de tek yönlü sıkıştırma algoritmalarında olduğu

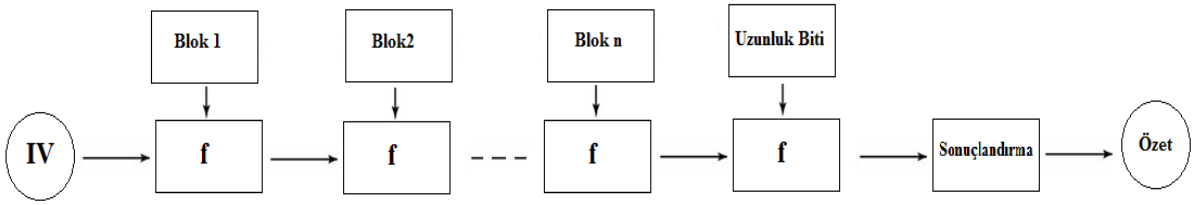
gibi A girişi yerinde açık mesaj B girişi yerinde şifreleme anahtarı ve C çıkışı yerinde ise şifreli mesaj bulunur. Bu şemaların yapıları kısaca incelenecektir.

Merkle-Damgard yapısı MD5 ve SHA-1 gibi kriptografik özetleme algoritmalarının içinde kullanılır, bir özetleme fonksiyonu rastgele uzunlukta mesajı sabit uzunlukta bir çıktı üretecek şekilde işleyebilmelidir bunu da ancak mesajı eşit uzunlukta bloklara böldükten sora ardışıl tek yönlü sıkıştırma fonksiyonları uygulayarak elde edebilir bu sıkıştırma işlemleri özetleme fonksiyonuna özgü olabileceği gibi hem blok şifreleme için hem de özetleme fonksiyonları için kullanılabilir. MD-güçlendirici olarak da bilinen uzunluk bitlerinin eklenmesi doğum günü paradoksunda ifade edilenden daha hızlı şekilde çakışma bulmayı engeller [15,16]. Bir ikincil dirence (second preimage) saldırısında yani verilen bir m1 mesajı için saldırganlar tarafından Hash(m1) = Hash(m2) olan bir m2 mesajının bulunması Kelsey ve Schneier'e göre [17] 2^k mesaj bloğu için eşitlik 3.12'deki kadar zaman alır.

$$k \times 2^{n/2+1} + 2^{n-k+1} \tag{3.12}$$

Burada karmaşıklığın $2^{n/2}$ ' den büyük 2^n ' den ise düşük olduğu görülmektedir. Birkaç yapısal zayıflıktan özellikle de uzunluk ilave probleminden dolayı, Stefan Lucks Merkle-Damgard yapısı yerine geniş-hatlı özetleme (wide-pipe hash) kullanmayı önerir [18]. Geniş-hatlı özetleme Merkle-Damgard yapısına çok benzer sadece onda iç yapıdaki mesaj boyutları daha geniştir yani içeride kullanılan bit uzunluğu çıkıştaki bit uzunluğundan daha geniştir. Bundan dolayı, son adımda ikinci bir sıkıştırma fonksiyonu içerideki son özet değerini sıkıştırarak sonuç özet değeri elde eder. SHA-224 ve SHA-384 bu formu kullanır ve onları sırasıyla SHA-256 ve SHA-512 den ayıran özelliklerden biri budur. Merkle-Damgard genel yapısı Şekil 3.5.'tegösterilmiştir.

Tek yönlü sıkıştırma fonksiyonlarının genellikle blok şifrelemede kullanıldığını daha önce söylenmişti, blok şifreleme de tek yönlü sıkıştırma fonksiyonlar gibi anahtar ve açık mesaj olmak üzere iki giriş alır ve çıkış olarak şifreli mesajı verir. Modern blok şifreleme ise kısmen tek yönlüdür, verilen bir açık mesaj ve bir şifreli mesaj için açık mesajı şifreli mesaja dönüştüren anahtarı bulmak mümkün olmamalıdır. Fakat verilen bir şifreli mesaj ve bir anahtar ile eşleşen açık mesaj blok şifreyi çözen fonksiyon kullanılarak kolaylıkla bulunabilir.



Şekil 3.5 Merkle-Damgard genel yapısı

Bir blok şifreleyiciyi tek yönlü sıkıştırma fonksiyonlarına dönüştürmek için bir takım işlemler eklenmelidir. Bazı metodlar normal bir blok şifreleyiciyi tek yönlü sıkıştırma fonksiyonuna dönüştürürler, bunlar tek blok uzunluklu sıkıştırma fonksiyonları olarak Davies-Meyer, Matyas-Meyer-Oseas, Miyaguchi-Preneel ve çift blok uzunluklu sıkıştırma fonksiyonları olarak MDC-2, MDC-4, Hirose'dur. Tek blok uzunluklu sıkıştırma fonksiyonları blok şifrelemenin temelinde olduğu gibi çıkış bitleri, işlenen bitlerle aynı sayıdadır, çift blok uzunluklu sıkıştırma fonksiyonlarında ise bitlerin sayısı iki katıdır. Eğer bir blok şifreleyici 128 bit tek blok uzunluklu methodu varsa 128 bit uzunlukta bir özetleme fonksiyonu oluşturur ve 128 bit bir özet üretir. Çift blok uzunluklu metotlarda ise 128 bit blok şifreleyici 256 bit özetleme fonksiyonunda dönüştürülebilir. Bir blok şifreleyiciyi özetleme algoritmasının sözü edilen 128 bit tek blok uzunluklu methodları varsa 128 bit blok boyutlu özetleme fonksiyonu oluşturur ve 128 bitlik bir özet elde eder.

Bu methodlar daha sonra Merkle-Damgard yapısında asıl özet fonksiyonu elde etmek için kullanılmıştır. Tek yönlü sıkıştırma fonksiyonları oluşturmak için blok şifreleyiciler kullanmak genelde bir miktar yavaşlamaya neden olur. Bu anahtar programlamanın (key scheduling) mesajın her bir bloğu için yapılmasından kaynaklanır. Anahtar programlama verilen bir anahtar için döngülerde kullanılacak alt anahtarlar üreten algoritmalar olarak tanımlanabilir. Black, Cochran ve Shrimpton bunun bir tek sabit anahtar ile bir blok şifreleme için bir kez çağırarak sıkıştırma işlemini yapabileceğini göstermişlerdir [19]. Uygulamada makul hızlarda seçilen blok şifreleyici ve anahtar programlamanın çok fazla yük getiren bir işlem olmadığı sonucu elde edilmiştir. Fakat, bazı durumlarda kolaylık sağlar çünkü blok şifreleyicinin tek bir uygulaması hem blok şifreleme için hem de özetleme fonksiyonu için

kullanılabilir, örneğin akıllı kartlarda veya arabalarda çok küçük gömülü sistemler için kodun kapladığı alanda yer kazanabilir.

Özetleme hızı veya oranı geçerli sıkıştırma fonksiyonunu baz alan özetleme fonksiyonunun verimliliğini belirtir. İteratif bir özetleme fonksiyonunun oranı blok şifreleme işlemlerinin sayısının çıkışa oranını ifade eder. Yani n blok şifreleyicinin çıkışındaki bit uzunluğunu, m giriş bitlerinin işlenme sayısını ifade ederken hız, m ile çıkış bitleri n ve gerekli blok şifreleme işlemleri s arasındaki orandır. Özetleme hızının formül olarak ifadesi eşitlik 3.13'de ifade edildiği gibidir.

$$R_h = \frac{|mi|}{s.n} \tag{3.13}$$

3.3. Diğer Sıkıştırma Fonksiyonlarından Örnekler

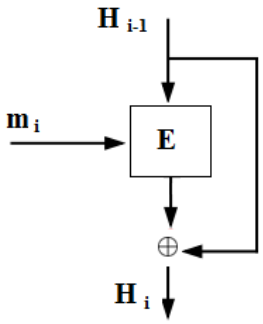
Merkle-Damgard'ın dışındaki diğer yapılar Davies-Meyer, Matyas-Meyer-Oseas, Miyaguchi-Preneel, Hirose olarak sıralanabilir. [20,21]. Bu yapıların işleyişlerini ve hız oranları aşağıdaki gibidir.

Davies-Meyer genel yapısı Şekil 3.6.'da görünen algoritma, tek blok uzunluklu tek yönlü sıkıştırma fonksiyonudur. Mesajın her bir bloğu m_i ve önceki özet değer H_{i-1} giriştir ve şifreli çıkış Xor işleminden sonra elde edilir. İlk durumda Ho bir sabit olarak alınır. Matematiksel gösterimi eşitlik 3.14'deki gibi tanımlanabilir.

$$H_i = E_{mi}(H_{i-1}) \oplus H_{i-1} \tag{3.14}$$

Hız ise k anahtar boyutu olmak üzere eşitlik 3.15'deki gibi tanımlanabilir [14]

$$R = \frac{k}{1.n} = \frac{k}{n} \tag{3.15}$$



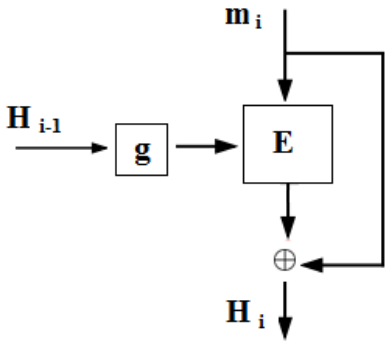
Şekil 3.6 Davies-Meyer genel yapısı

Matyas-Meyer-Oseas tek blok uzunluklu tek yönlü sıkıştırma fonksiyonunun genel yapısı Şekil 3.7.'de görülmektedir. Eğer blok şifreleyici farklı blok ve anahtar boyutlarına sahipse bu durumda H_{i-1} özet değeri anahtar olarak kullanmak için yanlış bir uzunluğa sahip olur, şifreleyici diğer özel gereklilikleri de anahtarda sağlamalıdır. Bu yüzden özet değer g fonksiyonundan geçirilerek gerekli dönüşümler ve eklemeler yapılarak şifreleme için uygun anahtar elde edilir. Matyas-Meyer-Oseas yapısının matematiksel gösterimi ise eşitlik 3.16'daki gibidir.

$$H_i = E_{g(Hi-1)}(m_i) \oplus m_i \tag{3.16}$$

Hız oranı ise şöyle ifade edilir:

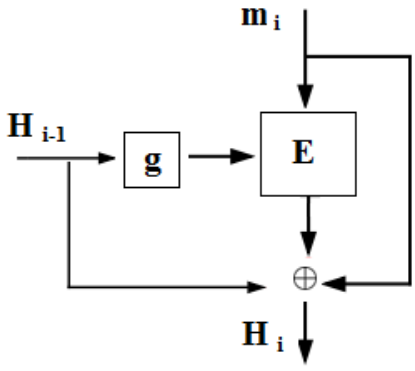
$$R = \frac{n}{1.n} = 1 \tag{3.17}$$



Şekil 3.7. Matyas-Meyer-Oseas genel yapısı

Miyaguchi-Preneel yapısı ise yine tek blok uzunluklu tek yönlü sıkıştırma fonksiyonudur, Matyas-Meyer-Oseas yapısının genişletilmiş halidir. Shoji Miyaguchi ve Bart Preneel tarafından bağımsız olarak sunulmuştur. Yapısı Şekil 3.8.'de gösterilmiştir. Matematiksel gösterimi eşitlik 3.18'deki gibidir, hız oranı ise Matyas-Meyer-Oseas için hesaplanan hız değeri ile aynıdır.

$$H_i = E_{g(Hi-1)}(m_i) \oplus H_{i-1} \oplus m_i \qquad (3.18)$$



Şekil 3.8 Miyaguchi-Preneel genel yapısı

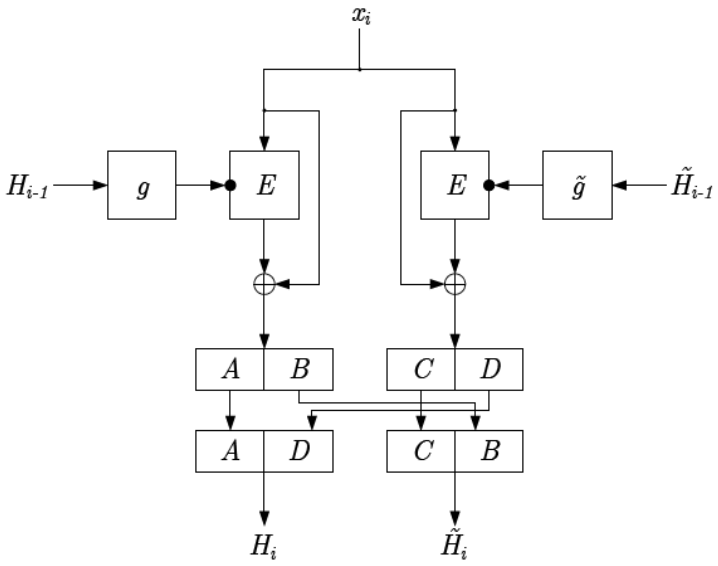
MDC-2 (Manipulation Detection Code) algoritması ise Matyas-Meyer-Oseas şemasının iki iterasyona ayırır, bu nedenle Şekil 3.9.'da görüldüğü gibi çift blok uzunlukludur. Aslında orijinali DES şifreleme algoritması için tasarlanmıştır fakat bu yapıya uygun istenilen özellikte

blok şifreleyici olamamıştır. Bu sebeple g ve $\tilde{g}$ fonksiyonları DES' e uygun anahtarları üretmek içindir. İşlem adımları ise şu şekildedir, U = $u_1u_2u_3\ldots u_{64}$ girişi için g(U) ve $\tilde{g}$(U) fonksiyonlarında her sekizinci bit u_8 'den başlanarak silinir, ikinci ve üçüncü bitler ise g fonksiyonunda 10 ile $\tilde{g}$ fonksiyonunda ise 01 bit dizisi ile yer değiştirilir.

$$g(U) = u_1 10 u_4 u_5 u_6 u_7 u_9 \ldots u_{63}, \tag{3.19}$$

$$\tilde{g}(U) = u_1 01 u_4 u_5 u_6 u_7 u_9 \ldots u_{63}. \tag{3.20}$$

Başlangıç vektörü IV = 0x5252525252525252 karşılığı olarak $\widetilde{IV}$ ise IV vektörünün 4 bit kaydırılmış halidir ve değeri 0x2525252525252525 dir. Bu değerler özelleştirilmek istenildiği takdirde değiştirilebilir. Her bir Matyas-Meyer-Oseas iterasyonunundan sonra iki çıkışın sağ yarıları değiştirilir. Hi ve $\widetilde{Hı}$ iki başlangıç çıkışı oluşturulmuş olur.



Şekil 3.9 MDC-2 genel yapısı

Ht || $\widetilde{Ht}$ son çıkışı eşitlik 3.21 ve 3.22'de tanımlandığı gibidir.

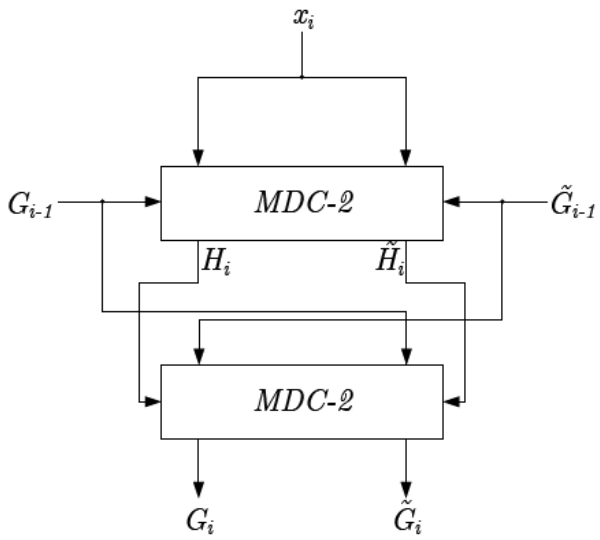
$H_i = C_i^L \parallel \check{C}_i^R$; $C_i = E_{g\,(Hi-1)(xi)} \oplus x_i$ (3.21)

$\check{H}_i = \check{C}_i^L \parallel C_i^R$; $\check{C}_i = E\tilde{g}_{(\tilde{H}i-1)(xi)} \oplus x_i$ (3.22)

for $1 \leq i \leq t$, $H_0 = IV$ ve $\widetilde{Ho} = \widetilde{IV}$ döngü ifadesi ile tekrarlanır. MDC-2 sıkıştırma fonksiyonu kullanılan özetleme fonksiyonunun hızı ise eşitlik 3.23'teki gibidir, burada bir giriş için iki çıkışa gerek duyulduğundan paydada 2.n bulunmaktadır.

$$R_{MDC2} = \frac{n}{2.n} = \frac{1}{2} \quad (3.23)$$

MDC-4 ise MDC-2 nin genişletilmiş halidir, yine Matyas-Meyer-Oseas şemasının iterasyonlara ayrılmış halidir. Şekil 3.10'da görüldüğü gibidir.



Şekil 3.10 MDC-4 genel şeması

MDC-2 den farklı olarak G_{i-1} ve $\tilde{g}i-1$ geri besleme değerleri vardır.Hi ve $\widetilde{Hı}$ başlangıç çıkışları MDC-2 sıkıştırma fonksiyonuna bağlanır. Çıkış Gt || $\widetilde{Gt}$ eşitlik 3.24 ve 3.25'deki gibi tanımlanabilir,

$H_i = C_i^L \parallel \check{C}_i^R$; $C_i = E_{g\,(Gi-1)(xi)} \oplus x_i$ (3.24)

$\check{H}_i = \check{C}_i^L \parallel C_i^R$; $\check{C}_i = E\tilde{g}_{(}\tilde{G}_{i-1)(xi)} \oplus x_i$ (3.25)

for $i \leq i \leq t$, G_0 = IV ve $\widetilde{Go} = \widetilde{IV}$ olarak tekrarlanır. MDC-4 sıkıştırma fonksiyonu kullanılan özetleme fonksiyonunun hızı eşitlik 3.26'deki gibidir, 4.n olmasının nedeni bir çıkış için dört blok şifreleme gerektirmesidir.

$$R_{MDC4} = \frac{n}{4.n} = \frac{1}{4} \quad (3.26)$$

Hirose çift blok uzunluklu tek yönlü sıkıştırma fonksiyonudur, bir blok şifreleme ve p permutasyonu içerir. Shoichi Hirose tarafından 2006'da önerilmiştir [22], Mridul Nandi tarafından ise bir çalışmada temel alınmıştır [23]. Her bir döngü m_i mesaj parçasını kullanarak G ve H durumlarını günceller. Öncelikle, m_i ve H_{i-1} birleştirilerek K_i anahtarını oluşturur. Sonra da bu iki geri beslemeli eşitlik 3.27 ve 3.28'daki ifadelere göre durumları günceller.

$Gi = E_{Ki}\,(G_{i-1}) \oplus G_{i-1}$ (3.27)

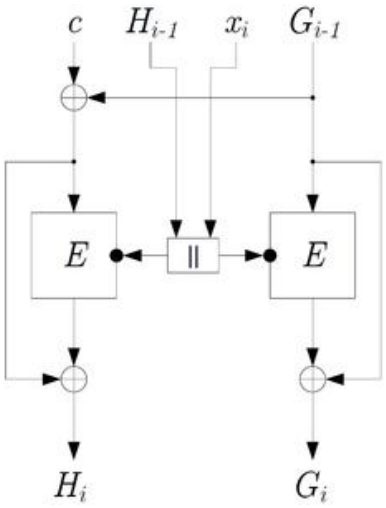
$Hi = E_{Ki}\,(p(G_{i-1})) \oplus p(G_{i-1})$ (3.28)

$p(G_{i-1})$ rastgele serbest sabit nokta permutasyonudur, genel olarak eşitlik 3.29'daki gibi ifade edilebilir

$p(x) = x \oplus c$ (3.29)

burada c sabiti sıfır olmayan herhangi bir değer seçilebilir. Hirose algoritmasının genel yapısı Şekil 3.11'de görüldüğü gibidir.

Genel yapısı anlatılan kriptografik özetleme algoritmalarının ardından bir sonraki bölümde kaotik özetleme algoritmaları anlatılacaktır.



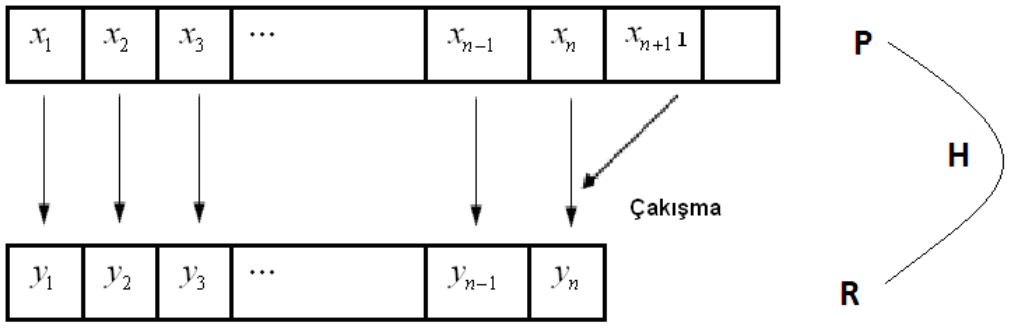
Şekil 3.11 Hirose genel yapısı

4. KAOTİK ÖZETLEME ALGORİTMALARI

Bu bölümde kaotik fonksiyonların özetleme algoritmaları içinde nasıl kullanıldığı incelenecektir. Örnekler incelenerek algoritmaları ve analiz yöntemleri üzerinde durulacaktır.

4.1. Kaotik Özetleme Algoritmaları ve Özellikleri

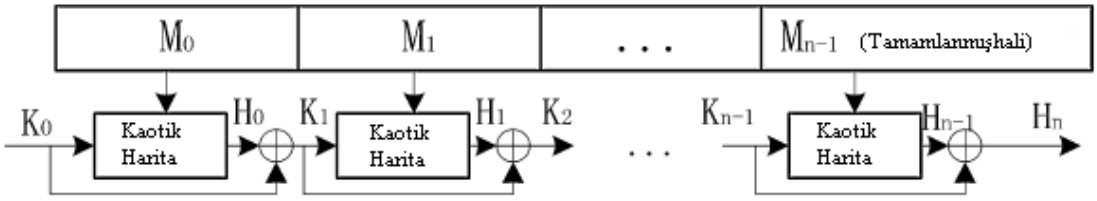
Bölüm 3 'de ayrıntılı olarak anlatıldığı gibi, özetleme algoritmaları bir metni giriş olarak alır ve sabit uzunlukta bir özet üretir. Bir özetleme algoritması H: P → R olarak tanımlanabilir ki burada P kümesinden bir değer R kümesinden bir değere eşleştirilir. Değişken uzunlukta metinlere karşı sabit uzunlukta özetler olduğundan, aradaki bu ilişki Şekil 4.1'de de gösterildiği gibi çoktan-bire şeklindedir, birebir değildir ancak örtendir. [24,25].



Şekil 4.1 Özetleme algoritması çoktan-bire ilişkisi

Kaotik özetleme algoritmalarında geleneksel özetleme algoritmalarında mantıksal işlevlerin ve sabit sayıların yerini, kaotik haritalar alır. Uygulanan yönteme göre değişmekle birlikte genelde iterasyon sayısı veya başlangıç şartları kaotik haritanın sonuç değerine göre düzenlenir. İncelenecek algoritmalar genellikle daha önce çalışılmış algoritmaların istatistikî değerlerini iyileştirmek amacıyla kuvvetlendirilmiş algoritmalardır. Bu tez çalışmasında iyileştirilen Xiao ve ekibince geliştirilen algoritma da

Amin ve arkadaşlarının çalışmasının [26] iyileştirilmesi sonucunda ortaya çıkmıştır. Kaotik özetleme algoritmalarının genel yapısı Şekil 4.2'de gösterilmiştir.



Şekil 4.2 Kaotik özetleme algoritmaları genel yapısı

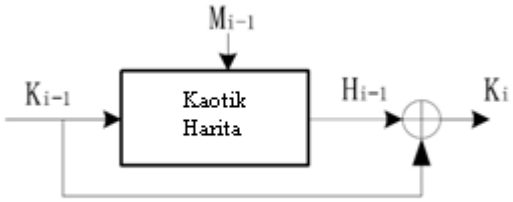
4.2. Tent Harita Tabanlı Kaotik Özetleme Algoritmasının Yapısı

Kaotik harita seçimi ve algoritmadaki kullanım şekline karar verilerek, aşağıdaki adımlar sırayla uygulanır.

1. Algoritmada kaotik harita olarak Tent harita seçilmiştir, denklemi aşağıdaki gibidir.

$$T(x) = \begin{cases} rx & 0 \le x \le 0.5 \\ r(1-x) & 0.5 \le x \le 1 \end{cases} \quad (4.1)$$

2. Açık mesaj ikilik sisteme dönüştürülüp, uzunluğu 1024'ün katı oluncaya kadar bir tane 1 ve tam katı olacak şekilde 0 eklenir.
3. Mesaj 1024 uzunlukta bloklara bölünür.
4. Her bir mesaj bloğu Şekil 4.3'de gösterilen yapıya göre işlenir ve özet değer oluşturulur.
5. 1024 bit uzunluktaki mesaj 8 adet 128 bitlik bloklar içermektedir.
6. Bir bloğun çıkış değerini hesaplamak için Tent harita fonksiyonu aşağıdaki gibi haritalanır.
   - Bloktaki her bir karakterin Ascii karşılığı Tent haritanın iterasyon sayısı olarak kullanılır.
   - Tent haritanın çıkışı 0,5'ten küçükse 0, büyükse 1 olarak haritalanır.



Şekil 4.3 Her bir birimin iç yapısı

7. Bir bloğun çıkışı olarak 128 karaktere karşılık olarak 128 değer üretilmiş olur, bu değer ile önceki bloktan gelen değer mantıksal XOR işleminden geçirilerek sonraki bloğun giriş değerini oluşturur. Başlangıç için giriş değeri sabit anahtar değeri olarak bilinen değerdir.
8. Tüm bloklar için işlem zincirleme devam ettiğinde, en son elde edilen değer açık metnin kaotik algoritmayla üretilmiş özet değeridir.

4.3. Algoritma Çakışma Analizi

Bu tez çalışmasında incelenmiş olan makalede Xiao ve arkadaşları [10] kaos tabanlı özetleme algoritmasının çakışma durumunu da analiz etmişlerdir. Denklem 4.1.'de ifade edilen tent harita için başlangıç durumlarını x_0=0.3 ve r_0=1.8 olarak almışlardır. Çakışmanın olduğu durumu ifade etmek için ilk iki karakteri farklı sonraki karakterleri aynı olan M = a,z, s3,..,si-1,si ve M'=b,y,s3,..,si-1,si şeklinde iki mesajın özetindeki çakışma durumu incelenmiştir. M mesajının ilk iki karakteri için tent harita fonksiyonuna göre hesaplanan değerler denklem 4.2 ve 4.3'de ifade edildiği gibidir. Burada mesajın ilk karakteri a olduğu için Ascii karşılığı 97 olarak hesaplanmış ve 0.3 başlangıç değeriyle 97 iterasyon sonucu 0.2635 olarak bulunmuştur, ikinci karakter de aynı şekilde 0.8645 olarak hesaplanmıştır.

$$x_1 = T^{asc(a)}(x_0) = T^{97}(0.3) = 0.2635 \quad (4.2)$$

$$x_2 = T^{asc(z)}(x_1) = T^{122}(0.2635) = 0.8645 \quad (4.3)$$

M' mesajı için ise hesaplanan değerler denklem 4.4 ve 4.5'de görünmektedir, iki mesaj için de hesaplanan değerler incelendiğinde ikinci karakterden sonra başlangıç değeri aynı olacağından sonraki karakterler de aynı olduğu için Ascii değerleri de her iki mesaj için de aynı iterasyon sonucunu verecektir. Böylece iki farklı mesaj için aynı özet değer elde edilmiştir. Bu durum çakışma olarak bilinmektedir.

$$x_1' = T^{asc(b)}(x_0) = T^{98}(0.3) = 0.4743 \quad (4.4)$$

$$x_2' = T^{asc(y)}(x_1') = T^{121}(0.4743) = 0.8645 \quad (4.5)$$

4.4. Kaotik Özetleme Algoritmalarının Güvenlik Testleri

Özetleme algoritmalarının güvenilirliğini test etmek için belirli istatistiksel yöntemler bulunmaktadır. Bu testlerin temeli metin ile özet arasındaki ilişkinin istatistikî olarak ortaya çıkartılamadığının test edilmesine dayanır [26]. Açık metin ile anahtarda yapılan en ufak değişikliğin özete yansıması oranını görebilmek için aşağıdaki yöntemler uygulanır.

1.En az değişen bit sayısı , $Bmin = \min(B_1, B_2, .., B_N)$ (4.6)

2. En çok değişen bit sayısı, $Bmaks = \max(B_1, B_2, .., B_N)$ (4.7)

3. Ortalama değişen bit sayısı, $\bar{B} = \frac{1}{N}\sum_{i=1}^{N} B_i$ (4.8)

4. Ortalama bit değişme olasılığı, $P = \left(\bar{B}/128\right) \times 100\%$ (4.9)

5. Değişen bit sayısının standart sapması, $\Delta B = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(B_i - \bar{B})^2}$ (4.10)

6. Standart sapma , $\Delta P = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(B_i/128 - P)^2} \times 100\%$ (4.11)

Test için mesaj içinde rastgele bir kısım seçilerek özeti elde edilir daha sonra içinden rastgele bir bit seçilir ve değiştirilerek yeni özet değer elde edilir. Değişen bit sayısı B_i olarak ifade edilir. Bu test N kez tekrarlanır. Özetleme algoritmasının gücünü gösteren değerlerin niteliği değişim olasılığının %50 'ye yakın olması, ortalama değişen bit sayısının özet uzunluğunun yarısına yakın olması ve standart sapmaların küçük değerler almasıdır. Örneğin 128 bit mesaj üreten bir algoritma için beklenen değer, ortalama değişen bit sayısı için 64'tür. Değişen bit sayısının olasılığı da bu sebeple %50'ye yakın olması gerekmektedir. Sapma değerlerinin küçük olmasının istenmesi mantığı ise değişen bit sayısının her örnek için birbirine yakın olması gerekliliğidir. N iterasyon için yapılan testlerin her birinde değişen bit sayısının 64'e yakın olması beklenmektedir.

Bu bölümde kaotik özetleme algoritmalarının genel yapısı örnek üzerinden anlatılmış ve test prensiplerinin dayandığı unsurlar belirtilmiştir. Bir sonraki bölümde geliştirilen algoritmanın yapısı ve testlerinin diğer algoritmalarla karşılaştırıldığı tablolar sunulacaktır.

5. GELİŞTİRİLEN ALGORİTMANIN YAPISI ve TESTLERİ

Bu bölümde öncelikle çakışma ihtimalini düşürmek için geliştirilen yöntem anlatılacaktır, sonra da İkeda harita kullanılarak üretilen kaotik özetleme algoritmasının içyapısı sunulacaktır. Son olarak test sonuçları ve değerlendirmeler vurgulanacaktır.

5.1. Çakışmayı Azaltmaya Yönelik Önlemler

Çakışma durumu incelenen algoritmanın dezavantajı birbirine yakın Ascii değerler üretecek karakterlerin mesajda bulunması durumunda yakın özet değerler çıkma olasılığının yüksek olmasıdır. Ayrıca bir blok sonunda elde ettiği değerleri 0 ya da 1'e atarken, 0 ile 0.5 arasının 0 'a, 0.5 ile 1 arasının da 1'e haritalanması ile elde edilen eş değerler çıkma olasılığı yüksektir. Bu problemleri aşmak için mesaj içindeki c_n değerine karşılık çalıştırılacak iterasyon sayısını Ascii (c_n)'den elde etmek yerine, denklem 5.1'deki eşitlik ile elde etmenin daha uygun olduğu ön görülmektedir.

$$n_{new} = (n_{old} + ASCII(c_n) + ABS(ASCII(c_n) - ASCII(c_{n-1}))) \bmod 1000 \qquad (5.1)$$

Denklemde de görüldüğü gibi ilgili karakter için kaotik haritanın kaç iterasyon çalıştırılacağı, kendisinden önceki değerlerin ürettiği iterasyon sayısına (n_{old}), ilgili karakterin Ascii değerine ve bir önceki karakterin Ascii değeri ile kendi Ascii değeri arasındaki farka bağlıdır. Bir açık metindeki karakterlerden birinin değişimi özeti o noktadan itibaren tamamen değiştirecektir. Değişecek karakter sayısını da matematiksel olarak ifade etmek istersek, n uzunlukta bir dizi için birinci karakterin değişimi n karakteri, ikinci karakterin değişimi n-1 karakteri etkileyecektir. N karakterli bir dizide değişimin etkileyeceği karakter ve ortalama ile toplam değişen karakter sayıları Tablo 5.1'de verilmiştir.

Çakışma ihtimalini artıra önemli diğer bir neden nihai özet değerini elde ederken mümkün oldukça karmaşıklığı sağlamaktır bunu yaparken uygulamaya geçirilmesini imkansız hale getirmemeye dikkat edilmelidir. Çünkü sonuç değerlerin elde edilmesinde ve haritalanmasında kullanılan basitlik çakışma ihtimalini artırır.

Tablo 5.1 Metin değişen karakter ile özet değişen karakter ilişkisi

Değişen karakter no	1	2		n-1	n
Etkilediği karakter No	n	n-1		2	1
Toplam	n*(n+1) / 2				
Ortalama	(n+1) / 2				

5.2. Önerilen Algoritmanın Yapısı

Önerilen kaos tabanlı haritalama algoritmasında kaotik sistemler olarak üç boyutlu kaotik sistemler kullanılmıştır. Algoritmada kaotik sistem olarak üç boyutlu kaotik sistem kullanılarak bir bloktan elde edilen sonuçları 0 ya da 1'e haritalarken üç farklı parametre değerlendirilmiş olmaktadır. İterasyon sayısı yine incelenen çalışmada olduğu gibi Ascii karşılığı ile elde edilir. Önerilen algoritmanın adımları aşağıdaki gibi sıralanmıştır.

1. Mesaj uzunluğu hesaplanarak sonuna 1 ve belli sayıda 0 eklenir. Eklenen bitlerle mesajın uzunluğu 1024'ün katı olacak şekildedir.
2. Elde edilen mesaj 128 bitlik bloklar haline getirilir.
3. Anahtar değeri üretilir.
4. Her bir blok için karakter temelli ilerlenerek, iterasyon sayısı karakterin Ascii karşılığından bulunur.
5. Üç boyutlu kaotik sistemin üç farklı durum değişkeni için başlangıç değerleri ile denklem 5.2'deki gibi elde edilir.

$$(x,y,z) = \text{Kaotik Harita}^{\text{Iterasyon Sayısı}}(x_0, y_0, z_0) \tag{5.2}$$

6. x,y ve z değerleri normal aralıkta değilse [0,1] aralığına aşağıdaki 5.3 denklemi ile normalize edilir.

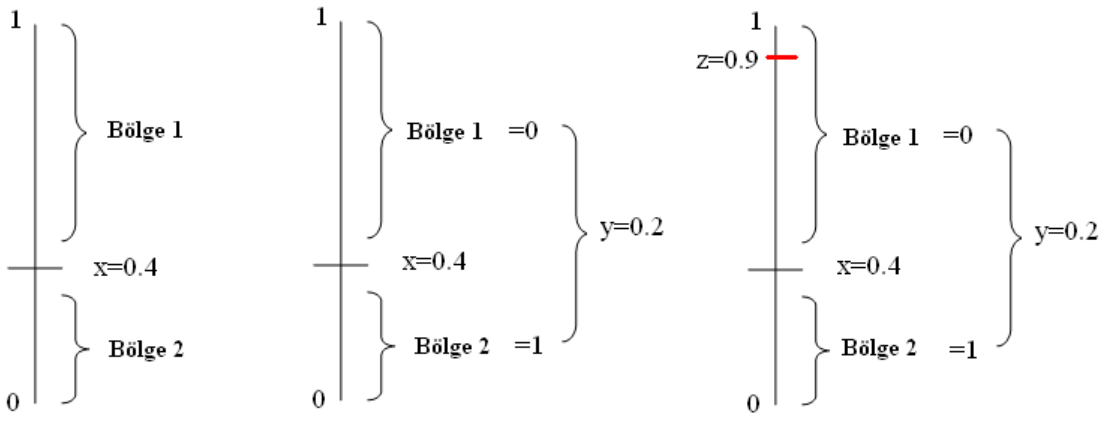
$$x_{normalize} = (x - x_{min}) / (x_{max} - x_{min}) \tag{5.3}$$

7. x, y ve z değerleri kullanılarak karakterlerin 0 veya 1 bit değerlerine dönüşmesi sağlanır.
8. x değeri [0,1] aralığını iki bölgeye ayıran sınır değerini belirlemektedir.
9. y değeri, x değeri ile iki bölgeye ayrılan aralıklardan hangi bölgenin 0 veya 1 olacağını belirlemekte kullanılır, bunun içinde denklem 5.4'de görülen haritalama yöntemi kullanılır.

   Bölge 1 = 1, $y \geq 0.5$; Bölge 1 = 0, $y < 0.5$ (5.4)

10. z değerinin hangi aralıkta olduğuna bakarak da karakterin 0 ya da 1 olacağına karar verilir.
11. Her bir blok anahtar değeri ile XOR işlemi uygulanır ve on altılık sayı sistemine çevrilerek karıştırma işlemine tabi tutulur.
12. Blokların sonucunda elde edilen bitler on altılık sayı sistemine çevrilerek özet değer elde edilir.
13. Karıştırma işlemi bir karakter dizisinde karakterlerin yerini değiştiren ve bir karakterin yeni yer indeksini bulurken kaos lojistik (logistic) harita kullanılmıştır. Bloktaki değerlerden 0 ya da 1 edilmesi şematik olarak Şekil 5.1'de gösterilmiştir.

Önerilen algoritma ile herhangi bir üç boyutlu kaotik sistem için n Ascii değerine sahip c karakteri için x_0, y_0, z_0 başlangıç koşulları sisteme girilerek kaotik sistem n kez çalıştırılır. Elde edilen 0 ile 1 aralığında değerlerin 0.4, 0.2 ve 0.9 olduğunu varsayılırsa, algoritmaya göre x=0.4 sınır değeri için [0,1] aralığı iki parçaya ayrılmıştır. Ardından y=0.2 değeri 0.5 değerinden küçük olduğu için bölge 1 0'a bölge 2 ise 1'e haritalanmış olmaktadır. Son olarak z=0.9 değeri bölge 1 içerisinde olduğundan c karakterine bölge 1'in haritalandığı 0 biti karşılık gelmektedir.



Şekil 5.1. Önerilen algoritmanın haritalanma yöntemi

5.3. Algoritmada Kullanılan Kaotik İkeda Harita

Bu çalışmada önerilen üç boyutlu kaotik harita olarak ikeda kullanılmıştır. Haritanın matematiksel gösterimi denklem 5.5'de gösterildiği gibidir.

$$I(x,y,z) = \begin{cases} z_n = 0.4 - \dfrac{6}{1+x_n^2+y_n^2} \\ x_{n+1} = 1+u(x_n\cos(z_n) - y_n\sin(z_n)) \\ y_{n+1} = u(x_n\sin(z_n) + y_n\cos(z_n)) \end{cases} \tag{5.5}$$

Önerilen sistemle çakışmanın olduğu algoritmaya göre iyileştirme yapıldığının tespiti için yine ilk iki karakterden itibaren aynı değerlere sahip karakter dizisi örneği üzerinde çalışılabilir. M = a,z, s3,..,si-1,si ve M'=b,y,s3,..,si-1,si için M mesajının ilk iki karakteri için hesaplanan değerler denklem 5.6 ve 5.7'da gösterilmiştir. İlk karakter için hesaplanan x_1, y_1, ve z_1 değerleri [0, 1] aralığına normalize edilirse sırasıyla 0.68, 0.27 ve 0.78 değerleri elde edilmektedir. Önerilen haritalama algoritmasına göre x_1=0.68 değeri 0-1 aralığını iki parçaya böler y_1=0.27 değeri 0.5 değerinden küçük olduğu için 0-1 aralığında alt bölge 1 ile üst bölge

0 ile etiketlenmektedir. Son olarak z_1=0.78 değeri 0 ile etiketlenmiş üst bölgede olduğundan mesajın a karakteri 0 bit değerine dönüştürülmüş olur. Mesajın ikinci karakteri olan b için ise x_2, y_2, ve z_2 değerlerinin normalize edilmiş biçimleri 0.84, 0.91 ve 0.41'dir. Yine x_2=0.84 değeri için 0-1 aralığı iki parçaya bölünür. y_2=0.91 değeri 0.5 değerinden büyük olduğu için ikiye ayrılan bölgelerden üst bölge 1 değeri ile etiketlenmiştir. Son olarak z=0.41 değeri 0 etiket değerine sahip alt bölgede olduğundan mesajın ikinci karakteri olan z değeri 0 bit değerine dönüştürülmüş olmaktadır.

$$(x_1,y_1,z_1) = I^{asc(a)}(x_0,y_0,z_0) = T^{97}(0.1,0.1,0) = (0.9967,-1.211,-17340) \quad (5.6)$$

$$(x_2,y_2,z_2) = T^{asc(z)}(x_1,y_1,z_1) = T^{122}(0.9967, -1.2113, -17340) = (1.2400,0.6068,-3.5326) \quad (5.7)$$

M' = b,y,s3,..,si-1,si mesajının ilk iki karakteri için hesaplanan değerler ise denklem 5.8 ve 5.9'da ifade edilmiştir.

$$(x_1,y_1,z_1) = I^{asc(b)}(x_0,y_0,z_0) = T^{98}(0.1,0.1,0) = (0.1510, -1.1281, -1.3336) \quad (5.8)$$

$$(x_2,y_2,z_2) = T^{asc(y)}(x_1,y_1,z_1) = T^{121}(0.1510,-1.1281,-1.3336) = (-2.5810,0.8868,-0.5326) \quad (5.9)$$

Görüldüğü gibi birbirine yakın Ascii değerler olsa dahi farklı başlangıç değerleri üretilmektedir, bu sonraki karakterler aynı olsa bile farklı sonuçlar üreteceğini göstermektedir.

5.4. Geliştirilen Algoritma ile Elde Edilen Özet Değerler

Önerilen algoritma kullanılarak "Fırat Üniversitesi Bilgisayar Mühendisliği Bölümü 1995 yılında kurulmuş olup, 1997 yılında lisans eğitimine başlamıştır. Bölümümüz ilk mezunlarını 2000-2001 Eğitim-Öğretim yılında vermiştir. Bölümümüzün hem örgün hem de ikinci öğretim programları mevcuttur. Bu programlara her yıl 80 örgün öğretim, 80 ikinci öğretim olmak üzere toplamda 160 öğrenci kayıt yaptırmaktadır." Metninin değiştirilmiş halleri için ürettiği özetler aşağıdaki gibidir.

1. Metnin orijinal halinin özet değeri : D388F3C5260832DF33C7EA4F702E8D65
2. İlk harfi “F” yerine “f” olarak değiştirildiğinde özet değeri: 408D88784F3DA6909D8AB2A5E4C6E641
3. Metin içinde geçen “1997” ifadesi “1987” olarak değiştirildiğinde özet değeri: E8EDC9DB4482655EEE3B6E100C7F7F15
4. Metnin sonundaki nokta işareti virgül olarak değiştirildiğinde özet değeri: 44536C977E070C9F6A4D252B14DD1E96

Ayrıca geliştirilen algoritmada anahtar değeri de rastgele üretilmiş bit dizisinden oluşmaktadır. Program anahtar değerini tekrar üreten bir fonksiyon eklenmiştir. Ancak bir mesajın farklı zamanlarda aynı özeti üretilebilmesi için aynı anahtar değeri ile oluşturulması gerektiği unutulmamalıdır

5.5. Güvenlik Testleri

Güvenlik testleri başlığı altında geliştirilen algoritmanın kabul görülen istatistik değerlendirmelere dayanan test sonuçları ve değerlendirilmesi anlatılacaktır. Bu testler mesajda bir bit değiştiği zaman bu değişikliğin özete yansıma oranını ifade eder. Hedeflenen oran %50 değişimin meydana gelmesidir. Ayrıca mesajda rastgele değiştirilen bitlerin sonucunda özette değişen bit sayısının birbirine yakın olması beklenmektedir. Bu homojen dağılımın bir sonucudur. Örneğin 128 bit özet üreten bir algoritma için, mesajda bir bit değiştiği zaman özette değişen bit sayısı 64 ise bu işlem bir kez daha tekrarlandığında yine 64’e yakın değişen bit sayısı tespit edilmesi gerekmektedir. Tekrar sayısı 256, 512, 1024 ve 2048 olarak yapılarak değişen ortalama bit sayıları ve ortalamadan sapma değerleri elde edilmektedir. Test sayıları diğer algoritmalar ile karşılaştırma yapabilmek için bu şekilde tutulmuştur. Geliştirilen programda test sayısı da bu rakamlar arasından seçimliktir. Tekrar sayısına göre geliştirilen algoritmanın ortalama değişen bit sayısı, standart sapma değerleri ve değişen en az ve en fazla bit sayısı değerleri Tablo 5.2’de gösterilmiştir. Bu tabloda da görüldüğü gibi ortalama değişen bit sayısının 64 olması algoritmanın güvenli olduğu hakkında bilgi vermektedir. Bu sonuç iki farklı mesajdan aynı özeti üretme ihtimalinin düşük olduğunu göstermektedir. Aynı şekilde standart sapma değerlerinin mümkün oldukça küçük rakamlar

olması mesajda değiştirilen her bitin özete yakın şekillerde etki ettiğini gösterdiği için bu değeri daha küçük olan algoritma çakışmalara karşı daha dayanıklıdır. Tablo 5.3'de ise bu konuyla ilgili yapılan diğer araştırmaların sonuçları bulunmaktadır. Buradaki algoritmaların farklı kaotik haritalar ve farklı haritalama yöntemleri kullandığı unutulmamalıdır.

Tablo 5.2 Geliştirilen algoritma için istatistiksel değeler

N	256	512	1024	2048
$\bar{B}$	64	63	64	64
P	50	49.2	50	49.2
ΔB	5.63	5.51	5.77	5.70
ΔP	4.94	4.87	4.94	4.87
En az değişim	49	49	47	50
En çok değişim	82	78	81	83

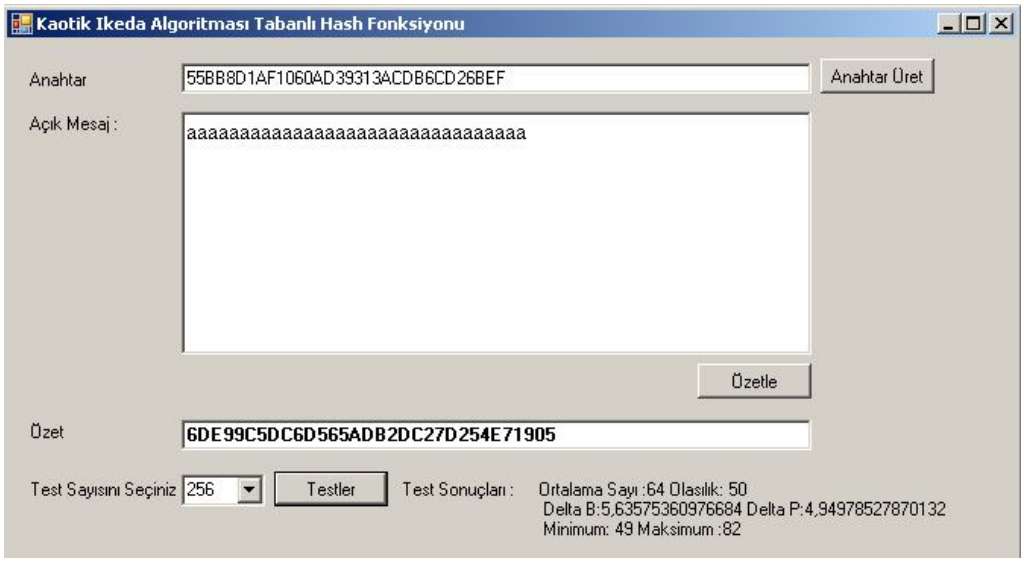
Tablo 5.3 Geliştirilen algoritmanın diğer algoritmalarla karşılaştırılması

		N=256	N=512	N=1024	N=2048
Ortalama	Önerilen	64	63	64	64
	Kaynak-27	64.441	63.801	63.839	63.850
	Kaynak-28	63.35	64.62	63.41	64.63
	Kaynak-29	64.09	63.92	64.08	63.98
Olasılık	Önerilen	50	49.99	50	49.95
	Kaynak-27	50.34	49.84	49.87	49.88
	Kaynak-28	49.32	50.53	49.49	49.46
	Kaynak-29	50.07	49.94	50.06	49.98
Ortalama Sapma	Önerilen	5.63	5.51	5.77	5.70
	Kaynak-27	5.522	5.708	5.708	5.789
	Kaynak-28	5.934	5.816	5.675	5.568
	Kaynak-29	5.38	5.62	5.56	5.53
Olasılığın Sapma Değeri	Önerilen	4.94	4.87	4.94	4.87
	Kaynak-27	4.31	4.46	4.46	4.52
	Kaynak-28	5.013	4.927	4.684	4.506
	Kaynak-29	4.21	4.39	4.34	4.33
En az değişim	Önerilen	49	49	47	50
	Kaynak-21	-	-	-	-
	Kaynak-27	51	48	47	44
	Kaynak-28	50	44	47	47
En çok değişim	Önerilen	82	78	81	83
	Kaynak-21	-	-	-	-
	Kaynak-27	69	73	78	85
	Kaynak-28	82	81	81	83

Tablo 4.3'de istatistiksel değerleri verilen algoritmaların ideal duruma yakın değerler çıkartmış olduğu görülmektedir. Kaynak-27 ile geliştirilen algoritmanın birçok değerlendirme kriteri için en iyi durumları gösterdiği sonucu elde edilmektedir. Değişen bit sayısının ortalama değerinin 64'e, değişen bit sayısı olasılığının %50'ye, standart sapma değerlerinin 0'a yaklaşmasına göre değerlendirdiğimiz için geliştirilen algoritmanın çakışma direncinin kuvvetli olduğunu söylenebilmektedir.

5.6. Özetleme Algoritması Programı Tanıtımı

İkeda tabanlı özetleme algoritmasının uygulaması Microsoft Visual Studio .Net ortamında C# dili ile yazılmıştır. Anahtar üretme fonksiyonu rastgele 0 ve 1 üretebilmek için lojistik kaotik fonksiyon kullanılmıştır. Aynı mesaj için aynı özeti elde etmek için anahtar değerinin ilkiyle aynı olması gerekmektedir. İstatistik testlerini yapmak için de seçimlik test sayıları bulunmaktadır. Burada 256, 512, 1024 ve 2048 test sayıları seçilebilmektedir. Test sonuçlarında da ortalama değişen bit sayısı, olasılık değeri, standart sapma değerleri ile en az ve en fazla değişen bit sayısı elde edilmektedir. Ekran ara yüzü örneği Şekil 5.2'de görünmektedir.



Şekil 5.2 Ekran ara yüz örneği

6. SONUÇ

Bu tez çalışmasında veri özetleme algoritmaları kaotik fonksiyonlar kullanılarak geliştirilmiş ve analiz edilmiştir. Özetleme algoritmaları bir veriye özgü ve metinle ilişkisi kurulamayacak derecede karışık karakterlerden oluşan özet veri üretirler. Bir metinden üretilen özet veri o metine özgüdür. Ayrıca giriş verisi metin yerine fotoğraf ya da video gibi veriler de olabilmektedir.

İletişiminde veri bütünlüğünün ve veri değiştirilmesinden endişe edilen durumlarda özet veri orijinal metinden elde edilir ve veri ile birlikte özet bilgisi de karşı tarafa gönderilir. Böylece veri karşı tarafa iletildiğinde tekrar özet bilgisi elde edilir, gönderilen özet ile elde edilen özet aynı ise verinin değiştirilmediğinden emin olunur. Bu özelliklerinden dolayı özetleme algoritmaları e-imza gibi uygulamaların temelini oluşturmaktadır.

Özetleme fonksiyonlarının giriş ve çıkış kümelerine baktığımızda, daha uzun metinden daha kısa uzunlukta veriler elde edilmektedir. Böyle bir yapıda daha dar bir alana eşleştirme yapıldığı için farklı mesajların aynı özetin çıkartılması kaçınılmazdır. Buna çakışma denir. Hedeflenen oluşturulan yapıda çakışma ihtimalini düşürmektir. Md5, Sha-1 ve Sha-2 gibi geleneksel özetleme algoritmaları denilen yapılarda metinden özet elde edilirken mantıksal ve matematiksel işlemler yapılarak metin karmaşık hale getirilerek özet veri oluşturulmaktadır. Son yıllarda geliştirilen yeni yaklaşımda ise matematiksel ve mantıksal karıştırma yöntemleri yerine kaotik fonksiyonların çıktıları kullanılmaktadır. Bunlara kaotik özetleme algoritmaları denilmektedir.

Kaotik fonksiyonların seçilme nedeni, bu fonksiyonların başlangıç şartlarına göre çıktılarının hesaplanamaz şekilde farklılık göstermesidir. Bu da giriş verisinde en ufak bir değişiklik yapıldığında aynı özet verinin hesaplanamaması amacıyla örtüşmektedir.

Geliştirilen algoritmada kaotik fonksiyon olarak İkeda haritalama yöntemi seçilmiştir. Kaotik İkeda haritasının üç parametresi kullanılarak bloklara ayrılan verilerden 0 ya da 1 değeri elde edilmiştir. Giriş verisi kaotik haritanın çalışma sayısına etki etmiştir. Ayrılan bloklarda fonksiyon sonucunda çıkan değerlerden elde edilen bit dizileri XOR işleminden geçirilerek nihai özet değeri elde edilmiştir.

Geliştirilen algoritmanın karmaşıklığının fazla olması, giriş verisinde değişen bit sayısına göre özet verisinde meydana gelen değişme oranının yüksek olması algoritmanın

çakışma ihtimalinin düşük olduğunu göstermektedir. Algoritmanın kalitesini değerlendirebilmek için bu alanda yapılan diğer çalışmalarla karşılaştırılmıştır. Kıyaslanan bazı algoritmalarla aynı değerleri üretmesi onlara alternatif olarak kullanılabilir olduğunu, bazılarından da daha iyi sonuçlar vermesi kötü değerler üreten algoritmalardan daha iyi olduğunu göstermiştir.

7. KAYNAKLAR

[1] **Sağıroğlu, Ş.,** 2008. Gazi Üniversitesi Bilgisayar Mühendisliği Bilgi ve Bilgisayar Güvenliği Ders Notu, Ankara.

[2] **Tübitak UEKAE,** 2006. Açık Anahtar Altyapısı Eğitim Kitabı.

[3] http://en.wikipedia.org/wiki/Pretty_Good_Privacy PGP. Nisan 2010.

[4] **Baker, G.L. and Gollub, J.P.,** 1990, Chaotic Dynamics, Cambridge University Press, Cambridge.

[5] **Özer, B.,** 2005 . Elektriksel Sürücü Sistemlerinde Doğrusal Olmayan Olguların Kaotik Analizi ve Yumuşak Hesaplama Yöntemleri İle Denetimi, *Doktora Tezi*, F.Ü. Fen Bilimleri Enstitüsü, Elazığ.

[6] MIT, Açık Ders Sistemi, İleri Diferansiyel Denklemler, 2009.

[7] **Strogatz, S. H.,** 1994, Nonlinear Dynamics and Chaos, Perseus Books, New York

[8] **Ren H., Yang W., Xie Q., Yang H.**, 2009 A novel method for one-way hash function construction based on spatiotemporal chaos, *Chaos, Solutions and Fractals*, **42**, 2014 - 2022

[9] **Akhavan A., Samsudin A., Akhshani A.**, 2009 Hash function based on piecewise nonlinear chaotic map, *Chaos, Solutions and Fractals*, **42**, 1046-1053

[10] **Xiao D,. Peng W.,Liao X., Xiang T.**, 2010. Collision analysis of one kind of chaos-based hash function, *Physics Letters A*, **374**, 1228-1231

[11] **S. Deng, Y. Li, Xiao D.,** 2010. Analysis and improvement of a chaos-based Hash function construction, *Commun Nonlinear Sci Number Simulat*, **15**, 1338-1347

[12] **Barkewits, T.,** 2009. Building Hash Functions from Block Ciphers, Their Security and Implementation Properties, Bochum.

[13] **Stallings, W.,** 2005. Cryptography and Network Security Principles and Practices, Fourth Edition, ABD.

[14] **Hensen, J.,** 2009. A Four Component Cryptographic Hash Framework, Milwaukee

[15] **Damgard, I.,** 1989. A design principle for hash functions, *Crypto*, **435,** 416–427.

[16] **Merkle, R.,** 1989. One way hash functions and DES, *Crypto*, **435,** 428–446.

[17] **Kelsey, J., Schneier, B.,** 2005. Second preimages on n-bit hash functions for much less than 2n work, EUROCRYPT 2005, Aarhus Üniversitesi, Aarhus, Danimarka,22-26 Mayıs, s. 474–490.

[18] **Lucks, S.,** 2004. Design Principles for Iterated Hash Functions, Cryptology ePrint Archive, http://eprint.iacr.org/2004/253.pdf

[19] **Black, J., Rogaway, P., ve Shrimpton, T.,** 2002. Black-Box Analysis of the Block-Cipher-Based Hash-Function Constructions from PGV. Advances in Cryptology, *CRYPTO '02*, Lecture Notes in Computer Science, **2442,** 320-335.

[20] **Hirose, J.,** 2006. Some Plausible Constructions of Double-Block-Length Hash Functions, http://www.iacr.org/archive/fse2006/40470213/40470213.pdf

[21] **Alfred, J., Paul C., Scott A.,** 2001.Handbook of Applied Cryptography Fifth Printing, CRC Press, United States of America.

[22] **Winternitz, R.,** 1984. A secure one-way hash function built from DES. *Information Security and Privacy, IEEE Symposium*, Oakland, California, USA. 22-25 Mayıs.

[23] **Nandi, M.,** 2005. Towards optimal double-length hash functions*, 6th International Conference on Cryptology in India* , Bangalor, Hindistan. 10-12 Aralık

[24] **Rosen, K. H.,** 1999. Discrete Mathematics and Its Applications, Mc-Graw Hill Press, USA.

[25] **Katz J. Lindell Y.,** 2008. Introduction to modern cryptography : principles and protocols, Chapman & Hall/CRC, United States of of America.

[26] **Amin M., Faragallah O.S.**,2009 . Chaos-based hash function (CBHF) for cryptographic applications, *Chaos Solitons and Fractals*, **42**, 767-772

[27] **Xiao D, Liao X, Deng S.,** 2005, One-way Hash function construction based on the chaotic map with changeable-parameter, *Chaos, Solitons and Fractals,***24**, 65–71.

[28] **H. Zhang, X. Wang, Z. Li, D. Liu**., 2005, One-way hash function construction based on spatiotemporal chaos, *Acta Physica Sinica* , **54**, 4006-11

[29] **Wang Y, Liao X, Xiao D, Wong K.,** 2008. One-way hash function construction based on 2D coupled map lattices, *Information Sciences*, **178**,1391–406.

ÖZGEÇMİŞ

KEVSER KIRKIL

Hazine Müsteşarlığı,
Ekonomik Araştırmalar Genel Müdürlüğü,
Emek, Ankara

kevserkirkil@gmail.com

1986	Elazığ'da doğdu
2004	Elazığ Anadolu Lisesi'nden mezun olduğu yıl Fırat Üniversitesi Bilgisayar Mühendisliği'ni kazandı
2008	Bilgisayar Mühendisliği lisans eğitimini tamamladı, aynı yıl Anadolu Üniversitesi İşletme Fakültesi'nde İşletme Bölümü'nde eğitime başladı
2008	Ekim ayında Odtü Teknokent bünyesinde YD Yazılım'a yazılım geliştirici olarak göreve başladı
2009	Ocak ayında Fırat Üniversitesi Bilgisayar Mühendisliği Yazılım A.B.D.'de yüksek lisansa kabul edildi
2009	Eylül ayında Sağlık Bakanlığı merkez teşkilatında Programcı olarak göreve başladı
2009	Güz yarı yılında yüksek lisans derslerinin bir kısmını Gazi Üniversitesi Bilgisayar Mühendisliği'nden özel öğrenci olarak alarak ders dönemini tamamladı
2011	Aralık ayında Hazine Müsteşarlığı'nda Ekonomik Araştırmalar Genel Müdürlüğü'nde göreve başladı, halen aynı görevdedir.