KARADENİZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

MATEMATIKSEL İFADELERİN ÜRETİMİ VE ÇÖZÜMÜNE DAYALI BİR KRİPTOLOJİ YÖNTEMİN TASARIMI VE GERÇEKLEMESİ

DOKTORA TEZİ

Sahereh HOSSEINPOUR

AĞUSTOS 2018
TRABZON

KARADENİZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

MATEMATİKSEL İFADELERİN ÜRETİMİ VE ÇÖZÜMÜNE DAYALI BİR KRİPTOLOJİ YÖNTEMİN TASARIMI VE GERÇEKLEMESİ

Sahereh HOSSEINPOUR

Karadeniz Teknik Üniversitesi Fen Bilimleri Enstitüsünce
"DOKTOR (BİLGİSAYAR MÜHENDİSLİĞİ)"
Unvanı Verilmesi İçin Kabul Edilen Tezdir.

Tezin Enstitüye Verildiği Tarih : 24 / 07 /2018
Tezin Savunma Tarihi : 13 / 08 /2018

Tez Danışmanı : Dr. Öğr. Üyesi Hüseyin PEHLİVAN

Trabzon 2018

KARADENİZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

Bilgisayar Mühendisliği Anabilim Dalında
Sahereh HOSSEİNPOUR Tarafından Hazırlanan

MATEMATİKSEL İFADELERİN ÜRETİMİ VE ÇÖZÜMÜNE DAYALI BİR KRİPTOLOJİ YÖNTEMİN TASARIMI VE GERÇEKLEMESİ

başlıklı bu çalışma, Enstitü Yönetim Kurulunun 24 /07/2018 gün ve 1763 sayılı kararıyla oluşturulan jüri tarafından yapılan sınavda
DOKTORA TEZİ
olarak kabul edilmiştir.

Jüri Üyeleri

Başkan : Prof. Dr. Yasin SOYLU .....

Üye : Prof. Dr. Halil İbrahim OKUMUŞ .....

Üye : Doç. Dr. Bekir DİZDAROĞLU .....

Üye : Doç. Dr. Burhan ERGEN .....

Üye : Dr. Öğr. Üyesi Hüseyin PEHLİVAN .....

Prof. Dr. Sadettin KORKMAZ
Enstitü Müdürü

ÖNSÖZ

Kriptografi gizli iletişim için yaygın olarak kullanılan bir tekniktir ve ceşitli şifreleme ve deşifreleme yöntemleri ile uygulanır. Ancak, şifreli metin dikkat çekebileceğinden, şifreleme sadece güvenli bilgi aktarımı için yeterli değildir. Bu problemi çözmek için steganografi, en küçük bir şüphe uyandıran stegometinleri kullanarak güvenli bir şekilde bilgi göndermenin yollarından biridir. Tipik olarak, bu sistemlerde, metinler çeşitli kapaklarda mesaj olarak gizlenebilir. Girilen mesaj sadece kapaklarda küçük değişikliklere neden olmalıdır. Genel olarak, kapaklar dört kategoriye ayrılabilir; metin, resim, video ve ses. Bu dört kategoride yer almayan, yürütülebilir dosyalar gibi başka kapaklar da olabilir. Bu tezde matematik ifadeler bir başka veri gizleme kapağı olarak ele alınmıştır. Bu ifadelerin özel yapılarından dolayı diğer kapaklara göre daha farklı yaklaşımların kullanımını gerektirmektedir. Yapılan çalışmada matematik ifadelere veri gömme ve yeni ifadeler üretme olmak üzere iki ana yöntemden söz edilmektedir.

Doktora çalışmamda danışmanlığımı üstlenerek bana çalışmalarımı yürütmemde her zaman sınırsız destek veren hocam Dr.Öğr. Üyesi Hüseyin PEHLİVAN'a, çalışma sürecinde değerli görüş ve katkılarını esirgemeyen sayın Prof. Dr. Halil İbrahim OKUMUŞ'a ve Doç.Dr. Bekir DİZDAROĞLU'ya teşekkürü bir borç bilirim. Doktora tezimde yer alan başta sevgili eşim Dr.Öğr.Üyesi Muhammed MİLANİ olmak üzere diğer tüm arkadaşlarıma ve ayrıca ilgileri ve desteklerini üzerimden eksik etmeyen sevgili babam ve anneme çok teşekkür ederim.

Sahereh HOSSEINPOUR

Trabzon, 2018

TEZ ETİK BEYANNAMESİ

Doktora Tezi olarak sunduğum "MATEMATIKSEL İFADELERİN ÜRETİMİ VE ÇÖZÜMÜNE DAYALI BİR KRİPTOLOJİ YÖNTEMİN TASARIMI VE GERÇEKLEMESİ" başlıklı bu çalışmayı baştan sona kadar danışmanım Dr. Oğr. Üyesi Hüseyin PEHLİVAN'nin sorumluluğunda tamamladığımı, verileri/örnekleri kendim topladığımı, deneyleri/analizleri ilgili laboratuvarlarda yaptığımı/yaptırdığımı, başka kaynaklardan aldığım bilgileri metinde ve kaynakçada eksiksiz olarak gösterdiğimi, çalışma sürecinde bilimsel araştırma ve etik kurallara uygun olarak davrandığımı ve aksinin ortaya çıkması durumunda her türlü yasal sonucu kabul ettiğimi beyan ederim. 13/08/2018

Sahereh HOSSEINPOUR

İÇİNDEKİLER

Sayfa No

Doktora Tezi

ÖZET

MATEMATİKSEL İFADELERİN ÜRETİMİ VE ÇÖZÜMÜNE DAYALI BİR KRİPTOLOJİ YÖNTEMİN TASARIMI VE GERÇEKLEMESİ

Sahereh HOSSEINPOUR

Karadeniz Teknik Üniversitesi
Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı
Danışman: Dr. Öğr. Üyesi Hüseyin PEHLIVAN
2018, 131 Sayfa

Veri gizleme son zamanlarda çok dikkat çeken bir konu haline gelmiştir. Steganografi için çeşitli yöntemler geliştirilmiştir ve eşzamanlı olarak gizli verileri saptamak için de uygun steganaliz tasarlanmıştır. Ancak, steganaliz analizlerine dikkat edilmemesi nedeniyle yeni tipte bir kapağa getirilen bir yaklaşım daha az şüpheli olabilir. Bu tezde, matematik ifadeler içerisinde mesaj gizleme ve bir mesajı matematiksel ifadeye dönüştürebilecek yöntemler önerilmiştir. Oluşturulan matematiksel ifade, mesajı güvenli bir metinle birlikte iletmek için bir kapak olarak kullanılabilir. Tez kapsamında, bilgisayar cebir sistemine dahil edilebilecek problemler için bir metodoloji önerilmiştir. Başlıca amacımız matematiksel ifadelerin içerisinde mesaj gizleme işlemi için yeni yöntemler geliştirmektir. Metodolojinin ilk aşaması matematiksel ifadelerin sözdizimini ve anlamını açıklayan biçimsel bir gramerin geliştirilmesini kapsar. Bir derleyici oluşturma aracı olan JavaCC kullanılarak, gramer bildiriminden bir parser üretilir. Parser, belirli bir matematiksel ifadeyi, diğer metodoloji aşamaları içerisinden düğümleri gezinerek değerlendirilen bir Soyut Sözdizim Ağacı'na (AST) dönüştürmek için kullanılır. Önerilen yöntemlere uygun gramerler geliştirilerek, değişik tiplerden matematiksel ifadeler üretilmiştir. Ayrıca, tezde, gizlenen mesajların daha güvenli olması için gömülmeden önce geliştirilen yeni yöntem ile şifrelenmesi sağlanmıştır.

Anahtar Kelimeler: Veri gizleme, Stokastik gramer, Matematik ifade üretmek, Abstract Syntax Tree, Derleyici, Parser, javaCC

PhD Thesis

SUMMARY

DESIGN AND REALIZATION OF A CRYPTOLOGIC METHOD BASED ON THE PRODUCTION AND SOLUTION OF MATHEMATICAL EXPRESSIONS

Sahereh HOSSEINPOUR

Karadeniz Technical University
The Graduate School of Natural and Applied Sciences
Computer Engineering Graduate Program
Supervisor: Dr. Öğr. Üyesi Hüseyin PEHLIVAN
2018, 131 Pages

Recently data hiding has been a very remarkable issue. Various methods have been developed for steganography, and at the same time appropriate steganalysis has been designed to detect confidential data. However, an approach to a new type of cover may be less susceptible because of the lack of attention to steganalysis analyzes. In this thesis, two methods are proposed that ones can hide messages in mathematical expressions and others transform a message into a mathematical expression. The generated mathematical expression can be used as a new cover for communicating the message with a secure text. In the direction of the thesis we propose a methodology for the problems that can be incorporated into the computer algebra system. Our main goal is to develop new methods for message hiding within mathematical expressions. The first step of methodology involves the development of a formal grammar that explains the syntax and meaning of mathematical expressions. Using a compiler construction tool such as JavaCC, a parser is generated from the grammar description. The parser is used to transform a particular mathematical expression into an Abstract Syntax Tree (AST) that is evaluated by traversing its nodes through the other stages of the methodology. Enhancing grammars in accordance with the proposed methods, mathematical expressions of various types are generated. Moreover, in the thesis, the messages are encrypted with the new developed method before being embedded so that they can be more secure.

Key Words: Data hiding, Stokastic grammar, Math expression generation, Abstract Syntax Tree, Compiler, Parser, javaCC

ŞEKİLLER DİZİNİ

Sayfa No

TABLOLAR DİZİNİ

Sayfa No

LİSTELERİN DİZİNİ

Sayfa No

SAMBOLLER DİZİNİ

CAS	Computer Algebra System
JavaCC	Java Compiler Compiler
YACC	Yet Another Compiler-Compiler
BNF	Backus-Naur Form
CFG	Context-Free Grammar
NLP	Natural Language Programming
MPL	Mathematical Pseudo Language
LL	Left-to-right and Left-most production
LR	Leftmost and Rightmost Derivations
AST	Abstract Syntax Tree
MathML	Mathematical Markup Language
PKE	Public Key Encryption
AES	Advanced Encryption Standard
DES	Data Encryption Standard
PGP	Pretty Good Privacy
RSA	Rivest Shamir Adleman
IDES	International DES
JIT	Just-In-Time
D2S	Data –to- Speech

1. GENEL BİLGİLER

1.1. Giriş

İnsan evrim tarihinde matematik önemli bir kökene sahiptir. Doğru komutlar kullanıldığında bilgisayarlar problem çözmede çok iyidir. Programlama dillerinin ilk nesli, (örneğin, assembly dili) toplama, çıkarma, çarpma ve bölme gibi dört temel işlem haricinde matematiksel ifadelerle çalışmak için özel ifadelere sahip değildi. Temel işlemler, donanım ile çalışmak için kullanılabilmesine rağmen, üst düzey programlama dilleri dikkate alındığında, daha karmaşık ifadeler desteklenir ve bu ifadeler alt işlemlere ayrıştırılır. Ayrıştırma işlemi genellikle yüksek seviyeli bir programlama dilinde tüm ifadeleri düşük seviyeli veya makine koduna çeviren derleyiciler tarafından yerine getirilir. Sürecin çok etkili sonuçları nedeniyle birçok ileri düzey dil ve sistem ortaya çıkmıştır.

Sembolik hesaplama veya cebirsel hesaplama, matematiksel ifadelerin geliştirilmesi ve değerlendirilmesi anlamına gelir. Sembolik hesaplama, değişkenler veya semboller içerebilen ifadelerle tam çözümler sunar. Eğer semboller herhangi bir değere ayarlanmamışsa, çıktı da değişkenlerin başka bir ifadesi olacaktır. Sembolik hesaplamalar yapabilen bilgisayar uygulamalarına Bilgisayar Cebir Sistemleri (CAS) veya sembol manipülasyon sistemleri denir [1]. CAS sistemlerinin kullanımının artması, matematik öğretiminde bilgisayar sistemlerinin rolünü genişletmiştir [2-4]. Örneğin, bilgisayar cebiri, sayısal programlarda gerekli olan formüllerin tasarlanması ve denenmesinde önemli bir role sahiptir.

Genelde CAS sistemlerinde İçerikten Bağımsız Gramerler (CFG) matematiksel ifadeleri ayrıştırmak ve üretmek için kullanılır. Girdi dizesi, o girdinin sözdizimini doğruladığı bir ayrıştırıcıdan geçer. Aynı zamanda, matematiksel ifadelerle çalışmak için uygun bir yapı olan bir soyut sözdizim ağacı (AST) oluşturulur. Bu ağaç, giriş dizgisinde çeşitli iyileştirme algoritmalarını uygulamak için önerilen metodolojinin ana çekirdeği görevi görür.

Genel amaçlı bilgisayar cebir sistemlerine ek olarak, belirli bir alandaki problemleri çözmek için geliştirilen matematik veya fizik için özel amaçlı yazılımlar vardır. Bunlara örnek olarak, grup teorisi [5, 6] için PARI, SIMATH ve KANT, sayı teorisi için CoCoA, cebirsel geometri için CoCoA [7], cebirsel geometri için Macaulay, ve Lie teorisi için LiE gibi geliştirilen GAP yazılımları verilebilir [8].

Problemleri çözmenin yanı sıra, Doğal Dil Üretimi (NLG) de şablon kullanarak ifade üretimi için geliştirilen farklı sistemlerdir. YAG [9], gerçek zamanlı ve genel amaçlı biçiminde

Şablon Tabanlı dizeler üretir. D2S (Data-to-Speech) [10], rota açıklaması, müzik, futbol raporu ve İngilizce dahil olmak üzere farklı dillerdeki farklı uygulamalar için geliştirilmiştir. EXEMPLARS [11], dinamik metin oluşturucuyu destekleyen, JAVA için bir üst kümesi olan, nesne tabanlı, kural tabanlı bir çatıdır ve HTML / SGML şablonlarını kullanabilir. XtraGen [12], NLG için diğer uygulamalarla kolayca entegre edilebilen XML ve JAVA tabanlı bir yazılım sistemidir.

Rastgelelik, geçmiş zamanlardan beri bilim insanları için ilginç bir konudur. Rastgele sayı üretimi için önerilen farklı yöntemler vardır [13-16]. Rasgele sayılar, kriptografi [17-19], Bilgisayar Simülasyonu [20, 21] ve hatta Hayvan Bilimleri [22] gibi farklı bilimlerde farklı uygulamalara sahiptir. Rastgele sayı üretimi açısından, kaotik işlevler [22] veya elektronik sesler [23] kullanan yöntemler gibi rasgele ve sahte sayılar üretmek için de farklı yöntemler önerilmiştir. Ayrıca, rastgele jenerasyonlar ve ilgili olasılıklar ile ilgili endişeleri olan E.N.GILBERT [24] çalışmaları da dahil olmak üzere rastgele jenerasyonlar büyük ilgi görmektedir. Rastgele üretim de Doğal Dil İşleme (NLP) olarak ele alınmış ve NLG adı altında, Langkilde [19] tarafından ve NLG için stokastik teknikler kullanılarak farklı sistemler geliştirilmiştir. Bu sistemler gerçek ve Şablon Tabanlı olmak üzere iki kategoriye ayrılmıştır. Bu kategoriler Kees Van Deemter tarafından karşılaştırmıştır [25].

Tillman Bechar [26], TAG kullanarak şablon tabanlı NLG için bir üretim yöntemi sundu. Temel ağaç düğümlerini bütünleştirerek rastgele bir dil nesline geçti. Bu konu için başka uygulamalar da geliştirilmiştir. Örneğin, Amruth N. Kumar [27] problemleri ve çoğunlukla programları üretmek için şablonlar kullandı. Test vakası üretimi, rastgele nesil başka bir uygulamadır. Takahide Y. et.al [28], çalışma zamanı testi olarak Just-In-Time (JIT) derleyicileri için bir araç tasarladı. Rasgele nesil konusu da otomatik testlerin tasarlanması için sunulmuştur. Öğretmenlerin internette soru dosyaları oluşturmasına yardımcı olmak için çeşitli sistemler oluşturulmuştur [29-31]. [32]'da, Joao et.al, test üretmek için tasarlandığı basit cevaplarla otomatik matematiksel testler üretecek bir sistem oluşturdu. [33]'de Ana Paula, Mantıksal Programlamayı Kısıtlamaya (CLP) dayanan otomatik matematik araştırmaları jenerasyonu için bir sistem tasarladı. Bu tür sistemler, internet ve sanal eğitim sistemleri gibi ortamlarda otomatik testler için olanaklar sağlar.

Ayrıca bilgisayar ağları ve internetin gelişmesi ve sürekli artan kullanımıyla, veri güvenliği ve bilgi alışverişinin bütünlüğü, yetkili ve yetkisiz kişilerin aktarılan verilere kolay erişimi sayesinde araştırmacıların karşılaştığı önemli bir sorun haline gelmiştir. Veri paketleri internet ortamında birkaç ara katmandan geçerek hedeflerine ulaştıklarından, üçüncü taraf

insanlar bunları yakalayabilirler ve değerlendirip kullanabilirler. Bu nedenle, internet üzerinden güvenliği sağlamanın temel yapı taşı veri paketlerinin korunmasıyla ilişkilidir.

Yeni uygulamaların, özellikle ticari, askeri uygulamaların ve video konferansların geliştirilmesi gibi dijital ortamların aktarımı için hızlı ve güvenli sistemlerin kullanılması gereklidir. Verilerin şifrelenmesi, geleneksel veya modern kriptografi, DNA tabanlı ve kaos tabanlı teknikler vb. gibi gruplarda sınıflandırılan çok farklı şifreleme algoritmaları ve teknikleri ile gerçekleştirilir. Bazı teknikler kuantum fourier dönüşümleri ve eliptik eğri gibi matematiksel kavramlara dayanır [34, 35]. Bazıları, uygulamada nadiren kullanılmış olan 1979'da Shamir tarafından önerilen gizli paylaşım kavramlarına [36] dayanmaktadır. Sıkıştırma tabanlı yöntemler, genellikle, SCAN dili [37] gibi sıkıştırmaya veya vektör nicelleştirmeye dayalı tekniklere dayanan dijital görüntüler için önerilen başka bir şifreleme yöntemidir [38].

DES (Veri Şifreleme Standardı), AES (Gelişmiş Şifreleme Algoritması), IDES (Uluslararası DES) veya RSA (Rivest–Shamir–Adleman) dahil olmak üzere çeşitli teknikler önerilmiştir. Bununla birlikte, bu yöntemler görüntüyü zaman ve hız açısından kodlamak için süre etkinliğine sahip değildir [39-41]. Dönüşüm tabanlı görüntü şifreleme tekniği, içsel olarak dönüşüm alanında gerçekleştirilen bir işleme dayanan bir yöntemdir. Bu işlem Fourier dönüşümü [42] ve / veya Dalgacık dönüşümü [43] gibi tekniklere dayanmaktadır.

Kriptografi gizli iletişim için yaygın olarak kullanılan bir tekniktir [44] ve ceşitli şifreleme ve deşifreleme yöntemleri ile uygulanır. Ancak, şifreli metin dikkat çekebileceğinden, şifreleme sadece güvenli bilgi aktarımı için yeterli değildir. Bu problemi çözmek için steganografi, en ufak bir şüphe uyandıran stegometinleri kullanarak güvenli bir şekilde bilgi göndermenin yollarından biridir. Tipik olarak, bu sistemlerde, metinler çeşitli kapaklarda mesaj olarak gizlenebilir. Girilen mesaj sadece kapaklarda küçük değişikliklere neden olmalıdır. Genel olarak, kapaklar dört kategoriye ayrılabilir: metin, resim, video ve ses [45]. Tabii ki, bu dört kategoride yer almayan, yürütülebilir dosyalar [46] gibi başka kapaklar da olabilir.

Bu kategoriler arasında, görüntü bazlı steganografi daha yaygın olarak kullanılmış ve görüntünün daha az tanınabilen kısımlarına veri dahil edilmiştir [47]. Video, mesajı eklemek için çok sayıda işlem gerektiren bir kapak olarak da bilinir [48]. Ancak, büyük hacimli mesajlar ekleme yeteneği nedeniyle, dikkat çekmişlerdir [49]. Sesli bir mesaj eklemek de mümkündür. Genellikle, insan kulağı 20HZ'den 20KHZ'ye kadar olan sesleri anlayabilir. Bu nedenle, bazı ses tipi kapaklarda, bu özellik tanınmayan bir şekilde veri eklemek için kullanılır [50].

Bazı steganografi yöntemlerinde de metinler kapak olarak kullanılmıştır [46]. Metin tabanlı yöntemlerin sayısı diğer yöntemlerden daha az olsa da, bu yöntemlerde büyük miktarda metin veri türü onları çekici hale getirmiştir.

Steganografinin aksine, steganaliz [51] yerleştirilen mesajın yetkisiz olarak çıkarılması bilimidir. Steganaliz ve steganografinin saklambaç oyununa benzer olduğu söylenebilir [52]. Steganografi için çeşitli yöntemlerin geliştirilmesi ile steganaliz için çeşitli yöntemler önerilmiştir [53]. Açıkçası, yeni yöntemler güvenli bilgi aktarımında daha güvenilirdir, çünkü steganaliz henüz yeni kapaklara odaklanmamıştır.

Bu tezde, matematiksel ifadelerde verilerin steganografisi için yeni bir kapak üretim yöntemi önerilmiştir. Matematiksel ifadeler, özellikleri göz önüne alındığında, bilgi gizlemek için iyi bir yer olabilir. Gerçek gibi görünen metinleri üretmek için çaba sarf edilmiştir [54]. Aynı bakış açısı ile belirsiz olmayan matematiksel ifadeler üretmenin yollarını önermek mümkündür.

Hassas verileri gizlemek için CAS'da önerilen teknikleri kullanabilir ve steganografi sistemleri için uygun bir platform sağlayabiliriz. Bu tezde, CAS kavramı kullanılarak, veri gizlemede etkili bir kapak üretim (Cover Generation) ve yerine koyma yöntemleri önerilmiştir.

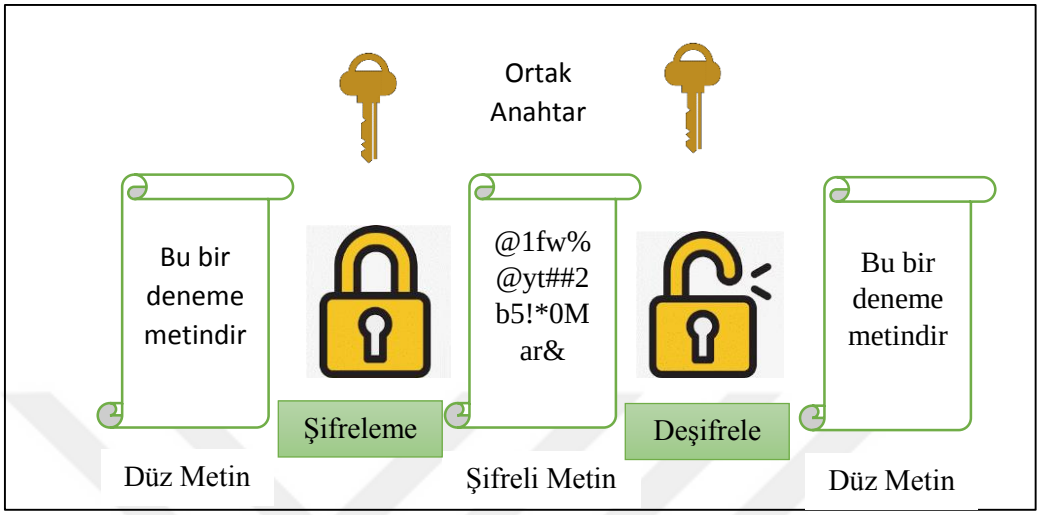
1.2. Tezin Kapsamı ve Amacı

Bu tezde kriptoloji biliminde veri gizleme (Information Hiding) konusu ele alınmıştır. Burada steganografi için matematiksel ifadeler yeni KAPAK olarak önerilmiştir. Matematik ifadelerin özellikleri bu çalışmada dikkata alınmıştır ve yeni gramerler tasarlanmıştır. Önerilen yöntemlere uygun gramerler geliştirilerek, değişik tiplerden matematiksel ifadeler üretilmiştir. Ayrıca, tezde, gizlenen mesajların daha güvenli olması için gömülmeden önce geliştirilen yeni yöntem ile şifrelenmesi sağlanmıştır.

1.3. Kriptografi ve Şifreleme

Şifreleme terminolojisinde mesaja düz metin veya temiz metin, mesajın bilgilerini kişilerden saklamak için kullanılan yönteme şifre ile kodlamaya şifreleme denir. Şifrelenmiş mesaja, şifreli metin, şifreli mesajdan metin elde etme işlemine çözme denir. Şifreleme ve çözmede genellikle bir anahtar kullanılır ve çözme işlemi ancak doğru anahtarın doğru bilinmesiyle gerçekleşir. Teorik olarak, bir şifreleme yaklaşımının güvenli olabilmesi için

anahtar olmadan kriptogramın anahtarını çözmek imkansız olmalıdır. Şifreleme bilimi ile uğraşanlara kriptograf adı verilir. Şifrelenmiş metnin üçüncü kişilerce kırılması kriptanaliz kelimesi ile ifade edilirken, bu işle meşgul olanlara kriptanalist denir. Şekil 1 bir metnin şifrelenmesi modelini göstermektedir.



Şekil 1. Bir metnin şifrelenmesi

Ağlarda bir merkezden diğer merkezlere gönderilen ve alınan veya gönderilen yerde saklanan verilerin korunması, yetkisi olmayan kişilerin bu verilere ulaşmasının önüne geçilmesi, günümüz teknolojisinde şifrelemeye ayrılan zaman ve önemi sürekli arttırmaktadır.

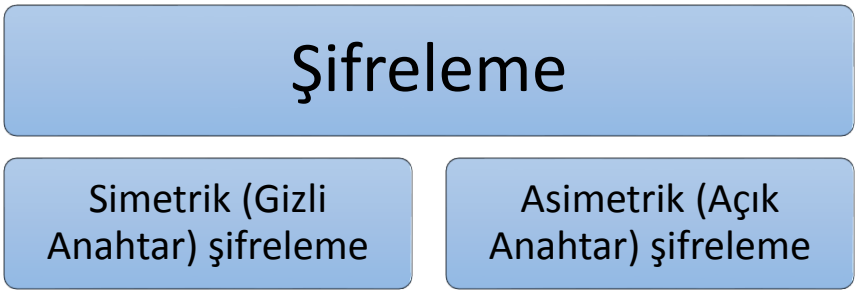
İnternet ve intranet uygulamalarında; e-posta, banka işlemleri, kişisel işlem ve bilgilerin saklanması, sayısal imza ve kimliklerin üretimi, veri tabanı dosyalarının korunumu, video şifreleme, elektronik oyun ve program şifrelemesi, faks ve telefon şifrelemesi vb. uygulamalar sıkça kullanılır durumdadır.

Haberleşme verileri üzerinde yapılabilecek saldırılar üç şekilde olabilir.

- Verilerin belirtilen hedefe gitmesini engelleme,
- Veriyi sadece okuma,
- Veriyi değiştirme.

Bu işlemler bilginin güvenliğini ciddi şekilde tehlikeye atmaktadır. Modern şifreleme, verilerin okunmasını ve değiştirilmesini engelleme ve verinin belirtilen kişi tarafından yollandığını garanti altına almayı amaçlar. Genelde Şekil 2'de gösterdiği gibi şifreleme teknikleri iki ana sınıfta olabilir:

- Gizli Anahtar (Simetrik)
- Açık Anahtar (Asimetrik)



Şekil 2. Şifreleme türleri

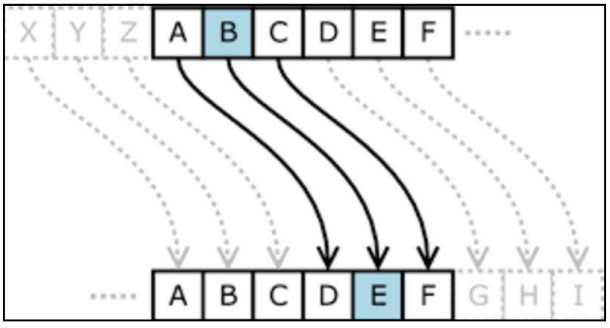
1.3.1. Gizli Anahtarlı (Simetrik)

Simetrik şifrelemede ve şifre çözme adımlarında aynı anahtar kullanılır. Sezar, AES, DES, 3DES, RC5, IDEA, Blowfish, SAFER ve Vigenere başlıca simetrik şifreleme yöntemlerindendir. Şekil 3 simetrik şifreleme yöntemini göstermektedir.

Şekil 3. Simetrik şifreleme

1.3.1.1. Sezar Yöntemi

Bu yöntem ile şifrelemek istediğiniz bir metin ve anahtar değeri olarak bir sayı belirlenir. Şifreli metni üretmek için, verilen metnin harfleri anahtar değer kadar kaydırılır.



Şekil 4.Sezar Yöntemi

Örneğin Şekil 4'deki gibi metnimiz ABCDEF ve anahtar değerimiz 3 olsun. A harfi 3 basamak kayarak yeni değeri D, B harfi 3 basamak kayarak yeni değeri E olur. Bu biçimde harfleri kaydırılarak şifrelenmiş metin elde edilir.

1.3.1.2. Vigenere Yöntemi

Daha önceden tanımlı olan A → 0, B → 1, C → 2,....., Z → 28 ilişkileri kullanılır ve her K anahtarı; anahtar kelime denilen m uzunluklu bir alfabetik dizi ile eşleştirilebilir. Vigenere şifresi, her mesaj elemanı m alfabetik karaktere eşit olduğu zaman bu karakterleri şifreleyebilir.

1.3.1.3. DES

Data Encryption Standart algoritması elektronik verilerin şifrelenmesi için simetrik anahtar algoritmasıdır. Güvensiz olmasına rağmen, modern kriptografinin ilerlemesinde oldukça etkili olmuştur. 1970'lerin başında IBM'de geliştirilen ve Horst Feistel'in daha önceki bir tasarımına dayanan algoritma, ajansın hassas, sınıflandırılmamış elektronik devlet verilerinin korunması için bir aday önerme davetini müteakip Ulusal Standartlar Bürosu'na (NBS) sunulmuştur. 1976 yılında, Ulusal Güvenlik Ajansı (NSA) ile görüştükten sonra, NBS nihayetinde, resmi bir Federal Bilgi İşleme Standardı (FIPS) olarak yayınlanmış, hafif bir modifiye edilmiş versiyonu seçmiştir. 1977'de Amerika Birleşik Devletleri için. NSA onaylı bir şifreleme standardının aynı anda yayınlanması, hızlı uluslararası kabulü ve yaygın akademik incelemesi ile sonuçlandı. Tartışmalar, sınıflandırılmış tasarım öğelerinden, simetrik anahtar blok şifre tasarımının nispeten kısa bir anahtar uzunluğundan ve NSA'nın katılımı ile bir arka kapı ile ilgili şüphelere yol açmıştır. Bugün bu şüpheleri ortaya çıkaran S-kutularının aslında NSA tarafından gizlice bildikleri bir arka kapıyı (diferansiyel kriptanaliz) kaldırmaları için tasarlandıkları bilinmektedir.

Bununla birlikte, NSA, anahtar büyüklüğünün kaba kuvvet saldırısıyla kırılabileceği şekilde büyük ölçüde azaltılmasını sağlamıştır. Algoritmanın zamanla aldığı yoğun akademik incelemeler, blok şifrelere ve onların kriptanalizine dair modern anlayışa yol açtı. DES güvensizdir. Bu, esas olarak 56 bitlik anahtar boyutunun çok küçük olmasından kaynaklanır.

Ocak 1999'da, dağıtık.net ve Electronic Frontier Foundation, bir DES anahtarını 22 saat 15 dakika içinde kırmak için işbirliği yaptılar Ayrıca, pratikte monte edilemez olmalarına

rağmen, şifre içindeki teorik zayıflıkları gösteren bazı analitik sonuçlar da vardır. Teorik saldırılara rağmen algoritmanın Triple DES şeklinde pratik olarak güvenli olduğuna inanılmaktadır. Bu şifreleme, Gelişmiş Şifreleme Standardı tarafından yerine getirilmiştir.

DES blok, sabit uzunlukta bir düz metin bit dizisi alan ve bir dizi karmaşık işlem sırasında aynı uzunluktaki başka bir şifreye dönüştüren bir algoritmadır. DES durumunda, blok boyutu 64 bittir. DES ayrıca, şifrelemeyi yalnızca şifrelemek için kullanılan belirli anahtarı bilen kişiler tarafından gerçekleştirilebilmesi ve dönüşümü özelleştirmek için bir anahtar kullanır. Anahtar görünümünde 64 bit oluşur; Bununla birlikte, sadece 56 tanesi algoritma tarafından kullanılmaktadır. Sekiz biti sadece parite kontrol etmek ve sonra atmak için kullanılır. Bu nedenle etkili anahtar uzunluğu 56 bittir. Anahtar, her biri tek bir parite olan nominal 8 baytta saklanır veya iletilir. Diğer blok şifreler gibi, DES tek başına güvenli bir şifreleme aracı değildir, bunun yerine bir operasyon modunda kullanılmalıdır.

1.3.1.4. AES

Gelişmiş Şifreleme Standardı veya AES, ABD hükümetinin gizli bilgileri korumak için seçtiği ve hassas verileri şifrelemek için dünya çapında yazılım ve donanımda uygulanan simetrik bir blok şifredir. Bu yeni simetrik anahtar algoritmasının seçim süreci, kamu incelemesine ve yorumuna tamamen açıktı. Bu, sunulan tasarımların eksiksiz ve şeffaf bir analizini sağladı. NIST, yeni gelişmiş şifreleme standardı algoritmasının 128, 192 ve 256 bit boyutlarında tuşları kullanarak 128 bit blokları işleyebilen bir blok şifresi olması gerektiğini belirtti. Bir sonraki gelişmiş şifreleme standardı algoritması olarak seçilmek için diğer kriterler:

Güvenlik: Rakip güçlerin, diğer rakiplerine kıyasla, saldırıya karşı direnme yetenekleri üzerinde yargılanmaları gerekiyordu, ancak güvenlik gücü rekabetin en önemli unsuru olarak görülüyordu.

Maliyet: Küresel, münhasır olmayan ve telifsiz bir temelde serbest bırakılması amaçlanan aday algoritmalar, hesaplama ve bellek verimliliği üzerinde değerlendirilmelidir.

Uygulama: Değerlendirilecek algoritma ve uygulama özellikleri, algoritmanın esnekliğini içerir; donanım veya yazılımda uygulanacak algoritmanın uygunluğu ve genel olarak, uygulamanın göreceli basitliği.

AES üç tip şifreleme barındırır: AES-128, AES-192 ve AES-256. Her şifre, sırasıyla 128, 192 ve 256 bit şifreleme anahtarlarını kullanarak 128 bitlik bloklardaki verileri şifreler.

Rijndael şifresi ek blok boyutları ve anahtar uzunluklarını kabul etmek için tasarlanmıştır, ancak AES için bu işlevler kabul edilmemiştir.

1.3.1.5. Blowfish

Blowfish, DES veya IDEA algoritmalarının yerini almak için kullanılabilen bir şifreleme algoritmasıdır. Değişken uzunlukta bir anahtar kullanan, 32 bitten 448 bite kadar simetrik (yani, gizli veya özel anahtar) bir blok şifrelidir; bu, hem yerel hem de ihraç edilebilir kullanım için yararlı kılar. (ABD hükümeti, özel durumlar hariç 40 bitten daha büyük anahtarlar kullanarak şifreleme yazılımının ihracını yasaklamaktadır.) Blowfish, 1993'te mevcut şifreleme algoritmalarına alternatif olarak Bruce Schneier tarafından tasarlanmıştır. 32-bit komut işlemcileri göz önünde bulundurularak tasarlanmıştır, DES'den önemli ölçüde daha hızlıdır. Kökeni olduğundan, önemli ölçüde analiz edilmiştir. Blowfish, patentsiz, lisanssız ve tüm kullanımlar için ücretsiz olarak kullanılabilir.

Blowfish, anahtarların değiştirilmesi dışında, yaygın kullanımda en hızlı blok şifrelerinden biridir. Her yeni anahtar, diğer blok şifrelere göre çok yavaş olan yaklaşık 4 kilobayt metin şifrelemek için ön işleme gerektirir. Bu, belirli uygulamalarda kullanımını engeller, ancak SplashID gibi başkalarında bir sorun değildir. Blowfish herhangi bir patente tabi değildir ve bu nedenle herkesin kullanımına açıktır. Bu, kriptografik yazılımlardaki popülerliğine katkıda bulunmuştur.

1.3.1.6. RC5

Birçok şemanın aksine, RC5 değişken blok büyüklüğüne (32, 64 veya 128 bit), anahtar büyüklüğüne (0 ila 2040 bit) ve tur sayısına (0 ila 255) sahiptir. Orijinal önerilen parametre seçimi, 64 bit blok 128 bitlik bir anahtar ve 12 tur idi. RC5'in önemli bir özelliği, veriye bağımlı rotasyonların kullanılmasıdır; RC5'in amaçlarından biri, bu tür işlemlerin kriptografik bir ilkel olarak değerlendirilmesidir. RC5 ayrıca bir dizi modüler eklenti ve exclusive OR (XOR) içerir. Algoritmanın genel yapısı Feistel benzeri bir ağdır. Şifreleme ve şifre çözme rutinleri birkaç satırlık kod ile belirtilebilir. Bununla birlikte, anahtar program daha karmaşıktır ve anahtarı, "kol numaramdan başka bir şey" kaynağı olarak hem e hem de altın oranın ikili açılımlarıyla esasen tek yönlü bir işlev kullanarak genişletmektedir. Algoritmanın verilere bağlı dönüşlerinin yenilikle birlikte basitleştirilmesi, RC5'i kriptanalistler için çekici bir çalışma nesnesi haline

getirmiştir. RC5 temel olarak RC5-w/r/b olarak ifade edilir; burada w word boyutunu bit cinsinden, r tur sayısını, b anahtardaki 8 bitlik bayt sayısını temsil eder.

1.3.1.7. IDEA (Internal Data Encryption Algorithm)

IDEA'nın en popüler şifreleme programı PGP'de kullanılan blok şifreleme algoritması olarak bilinmesiyle birlikte, algoritmalarının ücretsiz ticari olmayan kullanımına izin verme konusunda cömert davranmışlardır. IDEA algoritması kendi başına ilginçtir. İlk başta, bir blok şifresi yerine tersinmez bir hash işlevi olabileceğini belirten bazı adımlar içerir. Ayrıca, herhangi bir arama tablosunun veya S-kutusunun kullanılmasını tamamen engellememesi ilginçtir.

IDEA, her 16 bit uzunluğunda 52 alt anahtar kullanır. IDEA'daki düz metin bloğu, her 16 bit uzunluğa sahip dört bölüme ayrılmıştır. IDEA'da, 16 bitlik bir sonuç, toplama, XOR ve çarpma oluşturmak için iki 16 bit değeri birleştirmek için üç işlem kullanılır. Ekleme, taşıma, modulo 65,536 ile normal ilavesidir. IDEA'da kullanıldığı gibi çarpma, bazı açıklamalar gerektirir. Sıfırla çarpma her zaman sıfır üretir ve tersine çevrilemez. Multiplication modulo n, n'ye nispeten asal olmayan bir sayı olduğunda da tersine çevrilemez. Çoğullama, IDEA'da kullanıldığında, her zaman tersine çevrilebilir olmalıdır. Bu çarpma IDEA tarzı için geçerlidir.

2 ^ 16 + 1 olan 65.537 sayısı asal sayıdır. (Tesadüfen, 2 ^ 8 + 1 veya 257, aynı zamanda en yüksek değerdir ve 2 ^ 4 + 1 veya 17'dir, ancak 2 ^ 32 + 1 temel değildir, bu yüzden IDEA 128 bit'e kadar önemsiz olarak ölçeklendirilemez.) Böylece, 1'den 65.536'ya kadar olan sayılar için bir çarpım tablosunu oluşturuyorsa, her bir satır ve sütun her sayıyı yalnızca bir kez içerecek ve bir Latin karesi oluşturacak ve tersine çevrilebilir bir işlem sağlayacaktır. Normal olarak 16 bitin temsil ettiği sayılar 0'dan 65.535'e kadardır (veya belki daha da yaygın olarak -32.768'den 32.767'ye kadar). IDEA'da, çoğaltma amacıyla, tümü sıfırları içeren 16 bitlik bir kelimenin 65.536 sayısını temsil ettiği düşünülmektedir; Diğer sayılar geleneksel işaretsiz gösterimde temsil edilir ve çoğaltma modulodur, ana sayı 65.537'dir.

1.3.2. Asimetrik Şifreleme

Ortak anahtar şifrelemesi veya asimetrik kriptografi, anahtar çiftlerini kullanan herhangi bir şifreleme sistemidir; yaygın olarak dağıtılabilen genel anahtarlar ve yalnızca sahibi tarafından bilinen özel anahtarlar. Bu, iki işlevi yerine getirir; ortak anahtarın, eşlenmiş özel

anahtarın sahibinin iletiyi gönderdiğini doğruladığı kimlik doğrulama ve şifreleme, yalnızca eşlenmiş özel anahtar sahibinin, ortak anahtarla şifrelenmiş iletinin şifresini çözebilme.

Ortak anahtar şifreleme sisteminde, herhangi bir kişi alıcının ortak anahtarını kullanarak bir mesajı şifreleyebilir. Bu şifreli mesaj sadece alıcının özel anahtarı ile çözülebilir. Pratik olmak gerekirse, kamusal ve özel bir anahtarın oluşturulması hesaplama açısından ekonomik olmalıdır. Ortak anahtar şifreleme sisteminin gücü, ortak anahtarın eş anahtarını bulmak için gereken hesaplama çabasına (kriptografide çalışma faktörü) dayanır. Etkin güvenlik sadece özel anahtarı gizli tutmayı gerektirir; genel anahtar güvenlikten ödün vermeksizin açıkça dağıtılabilir.

Açık anahtarlı kriptografi sistemleri, çoğu zaman, belirli bir tamsayı faktorizasyonu, ayrık logaritma ve eliptik eğri ilişkilerinde sayılırken, şu anda etkin bir çözüm kabul etmeyen matematiksel problemlere dayanan kriptografik algoritmalardır. Ortak anahtar algoritmaları, simetrik anahtar algoritmalarından farklı olarak, taraflar arasındaki bir veya daha fazla gizli anahtarın ilk kez değiştirilebilmesi için güvenli bir kanal gerektirmez.

Asimetrik şifrelemenin hesaplama karmaşıklığından dolayı, genellikle sadece küçük veri blokları için kullanılır; tipik olarak simetrik bir şifreleme anahtarının (örneğin, bir oturum anahtarı) transferi. Bu simetrik anahtar daha sonra potansiyel olarak uzun mesaj dizisinin kalanını şifrelemek için kullanılır. Simetrik şifreleme / şifre çözme, daha basit algoritmalara dayanır ve çok daha hızlıdır.

Bir genel anahtar imza sisteminde, bir kişi, mesaj üzerinde kısa bir dijital imza oluşturmak için bir mesajı özel bir anahtarla birleştirebilir. İlgili genel anahtara sahip olan herkes, bir imzayı, üzerinde varsayılan bir dijital imza ve bilinen açık anahtarı birleştirerek, imzanın geçerli olup olmadığını, yani ilgili özel anahtarın sahibi tarafından yapıldığını doğrulayabilir. İletinin değiştirilmesi, tek bir harfin yerine bile, doğrulama işleminin başarısız olmasına neden olur. Güvenli bir imza sisteminde, açık anahtardan veya herhangi bir sayıda imzadan çıkarılması için özel anahtarı bilmeyen veya herhangi bir imzanın görülmediği herhangi bir mesajda geçerli bir imza bulmak için hesaplanabilir bir şekilde mümkün değildir. Böylece özel anahtarın sahibinin özel anahtarın gizli kalması kaydıyla, bir mesajın gerçekliliği imzayla gösterilebilir.

Ortak anahtar algoritmaları, kripto sistemlerinde, uygulamalarda ve protokollerde temel güvenlik bileşenleridir. Aktarım Katmanı Güvenliği (TLS), S / MIME, PGP ve GPG gibi çeşitli İnternet standartlarını desteklerler. Bazı genel anahtar algoritmalar, anahtar dağıtımı ve gizliliği

(örneğin, Diffie-Hellman anahtar değişimi), bazıları dijital imzaları (örneğin, Dijital İmza Algoritması) ve bazıları ise her ikisini de (örneğin RSA) temin etmektedir.

Bilgi güvenliği (IS), elektronik bilgi varlıklarının güvenlik tehditlerine karşı korunmasının tüm yönleriyle ilgilidir. Açık anahtar şifrelemesi, elektronik haberleşmenin ve veri depolamanın gizliliğini, gerçekliğini ve güvenilmezliğini sağlamak için bir yöntem olarak kullanılır.

Genel Anahtar Şifreleme (PKE) hedefi, gönderilen iletişimin geçiş sırasında gizli tutulmasını sağlamaktır. PKE'yi kullanarak bir mesaj göndermek için mesajın göndereni, mesajın içeriğini şifrelerken alıcının ortak anahtarını kullanır. Şifreli mesaj daha sonra alıcıya elektronik olarak iletilir ve alıcı mesajı çözmek için kendi özel anahtarlarını kullanabilir. Alıcının genel anahtarını kullanmanın şifreleme işlemi, mesajın gizliliğini korumak için sadece alıcının mesajın şifresini çözmek için eşleşen özel anahtara sahip olması açısından yararlıdır. Bu nedenle, mesajın göndereni, alıcının açık anahtarı kullanılarak şifrelenmiş bir mesajın şifresini çözemez. Ancak, mesajın alıcının açık anahtarına erişimi olan herhangi bir kişi tarafından gönderilebildiği için PKE, reddetme sorununu çözmez.

1.3.2.1. Diffie-Helman

Diffie-Hellman anahtar değişimi kriptografik anahtarların bir kamusal kanal üzerinden güvenli bir şekilde değiştirilmesine yönelik bir yöntemdir ve ilk olarak Ralph Merkle tarafından kavramsallaştırılan ve Whitfield Diffie ve Martin Hellman'ın adını taşıyan ilk halka açık protokollerden biridir. Diffie – Hellman, kriptografi alanında uygulanan en erken pratik anahtar uygulamalardan biridir.

Geleneksel olarak, iki taraf arasında güvenli şifreli iletişim, güvenilir bir kurye tarafından taşınan kağıt anahtar listeleri gibi bazı güvenli fiziksel kanallar tarafından anahtarları ilk olarak değiştirmelerini gerektirdi. Diffie – Hellman anahtar değişim metodu, güvensiz bir kanal üzerinden ortak bir gizli anahtar oluşturmak için birbirlerini önceden bilmeyen iki tarafa izin verir. Daha sonra simetrik anahtar şifrelerini kullanarak sonraki iletişimleri şifrelemek için kullanılabilir.

Diffie-Hellman, çeşitli İnternet servislerini güvence altına almak için kullanılır. Bununla birlikte, Ekim 2015'te yayınlanan araştırmalar, o zamandaki birçok DH İnternet uygulamasında kullanılan parametrelerin, büyük hükümetlerin güvenlik hizmetleri gibi çok iyi finanse edilen saldırganların taviz vermesini önleyecek kadar güçlü olmadığını göstermektedir.

Diffie-Hellman Key Exchange, bir açık ağ üzerinden veri alışverişi için gizli iletişim için kullanılabilecek iki taraf arasında paylaşılan bir sır kurmaktadır. Aşağıdaki kavramsal diyagram, çok büyük sayıların yerine renkleri kullanarak anahtar değişiminin genel fikrini göstermektedir. Süreç, iki tarafın, Alice ve Bob'un, gizli tutulmaya ihtiyaç duymayan, keyfi bir başlangıç rengini kabul etmeleriyle başlar (her seferinde farklı olmalıdır).

Bu örnekte renk sarıdır. Her biri kendilerine sakladıkları gizli bir renk seçer; bu durumda, turuncu ve mavi-yeşil. Sürecin en önemli kısmı, Alice ve Bob'un, kendi ortak renkleriyle birlikte kendi gizli renklerini bir araya getirip, sırasıyla turuncu-tan rengi ve açık mavi karışımlarla sonuçlanması ve ardından iki karışık rengi alenen almalarıdır. Son olarak, her bir karışım, ortakdan aldıkları rengi kendi özel renkleriyle bir araya getirir.

Sonuç, partnerin renk karışımına özdeş olan son bir renk karışımı sarı-kahverengidir. Eğer üçüncü bir taraf değiş tokuş dinlediyse, gizli renkleri belirlemeleri onlar için hesaplama açısından zor olurdu. Aslında, renkler yerine büyük sayılar kullanıldığında, bu eylem, modern süper bilgisayarların makul bir süre içinde yapması için hesaplama açısından pahalıdır.

1.3.2.2. RSA (Rivest-Shamir-Adleman)

RSA (Rivest-Shamir-Adleman) ilk açık şifreleme sistemlerinden biridir ve güvenli veri iletimi için yaygın olarak kullanılmaktadır. Böyle bir şifreleme sisteminde, şifreleme anahtarı geneldir ve gizli (özel) tutulan şifre çözme anahtarından farklıdır. RSA'da, bu asimetri, iki büyük asal sayı olan çarpımın "faktoring problemi" nin çarpıtılmasının pratik zorluğuna dayanmaktadır.

RSA kısaltması, ilk önce 1978'de algoritmayı halka açıklayan Ron Rivest, Adi Shamir ve Leonard Adleman'ın soyadlarının ilk harflerinden oluşur. İngiliz istihbarat ajansı Hükümet İletişim Merkezi (GCHQ) için çalışan bir İngiliz matematikçi olan Clifford Cocks, 1973'te eşdeğer bir sistem geliştirmişti, ancak bu 1997 yılına kadar sınıflandırılmamıştı.

Bir RSA kullanıcısı, bir yardımcı değer ile birlikte iki büyük asal sayı temelinde bir ortak anahtar oluşturur ve yayınlar. Asal sayılar gizli tutulmalıdır. Herkes, bir mesajı şifrelemek için herkese açık anahtarı kullanabilir, ancak şu anda yayınlanmış yöntemlerle ve genel anahtar yeterince büyükse, yalnızca asal sayıları bilgisi olan biri mesajın şifresini çözebilir. RSA şifrelemesinin kırılması RSA problemi olarak bilinir. Faktoring problemi kadar zor olup olmadığı açık bir sorudur. RSA nispeten yavaş bir algoritmadır ve bu nedenle, kullanıcı verilerini doğrudan şifrelemek için daha az kullanılır. Daha sık olarak, RSA, şifrelenmiş

paylaşımlı anahtarları simetrik anahtar şifrelemesi için geçirir, bu da daha yüksek hızda toplu şifreleme-şifre çözme işlemlerini gerçekleştirebilir.

RSA algoritması asimetrik kriptografi algoritmasıdır. Asimetrik aslında iki farklı anahtarda, yani ortak anahtar ve özel anahtar üzerinde çalıştığı anlamına gelir.

Bir istemci (örneğin tarayıcı) genel anahtarını sunucuya gönderir ve bazı veriler ister. Sunucu, müşterinin genel anahtarını kullanarak verileri şifreler ve şifrelenmiş verileri gönderir. Müşteri bu verileri alır ve şifresini çözer. Bu, asimetrik olduğundan, tarayıcı dışında başka hiç kimse, üçüncü tarafın tarayıcıda açık anahtarı olsa bile verilerin şifresini çözemez.

RSA fikri, büyük bir tamsayıyı hesaplamak zor olduğu gerçeğine dayanmaktadır. Ortak anahtar, bir sayının iki büyük asal çarpanı şeklindedir. Ve özel anahtar aynı iki asal sayıdan da elde edilir. Yani, eğer birisi çok sayıda faktörü hesaplayabilirse, özel anahtar tehlikeye düşer. Bu nedenle, şifreleme gücü tamamen anahtar boyutunda yatar ve anahtar boyutunu iki katına çıkarırsak, şifreleme gücü katlanarak artar. RSA anahtarları genellikle 1024 veya 2048 bit uzunluğunda olabilir, ancak uzmanlar yakın gelecekte 1024 bit anahtarın kırılabileceğine inanmaktadırr. Ancak şimdiye kadar mümkün olmayan bir görev gibi görünmektedir.

1.3.2.3. PGP (Pretty Good Privacy)

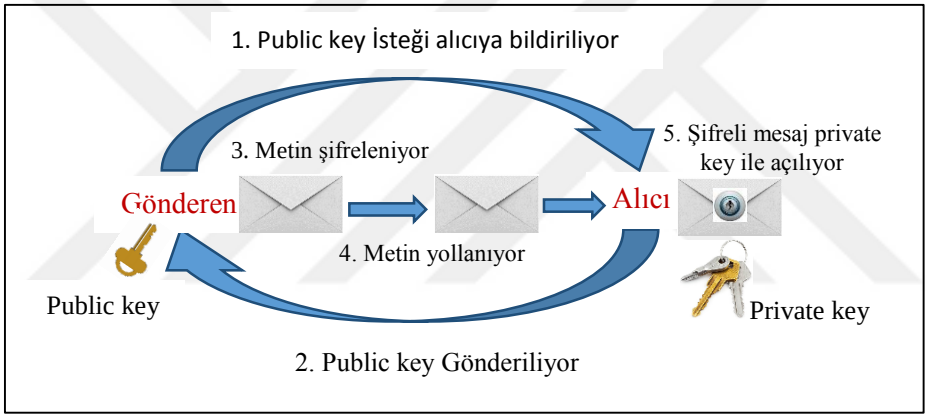
Pretty Good Privacy (PGP), veri iletişimi için kriptografik gizlilik ve kimlik doğrulaması sağlayan bir şifreleme programıdır. PGP, metinleri, e-postaları, dosyaları, dizinleri ve tüm disk bölümlerini imzalamak, şifrelemek ve şifresini çözmek ve e-posta iletişiminin güvenliğini artırmak için kullanılır. Phil Zimmermann tarafından 1991 yılında geliştirilmiştir.

PGP şifreleme, karma, veri sıkıştırma, simetrik anahtarlı şifreleme ve nihayet açık anahtar şifrelemesinden oluşan seri bir kombinasyonu kullanır; her adım, desteklenen birkaç algoritmadan birini kullanır. Her bir ortak anahtar, bir kullanıcı adına veya bir e-posta adresine bağlanır. Bu sistemin ilk versiyonu, sertifika yetkilisine dayanan ve daha sonra PGP uygulamalarına eklenen bir hiyerarşik yaklaşım kullanan X.509 sistemi ile karşılaştırmak için bir güven ağı olarak bilinmekteydi. PGP şifrelemesinin mevcut sürümleri, otomatik bir anahtar yönetim sunucusu aracılığıyla her iki seçeneği de içerir.

PGP mesajları gizli olarak göndermek için kullanılabilir. Bunun için PGP, simetrik anahtar şifrelemeyi ve genel anahtar şifrelemeyi birleştirir. Mesaj, simetrik bir anahtar gerektiren bir simetrik şifreleme algoritması kullanılarak şifrelenir. Her simetrik anahtar sadece bir kez kullanılır ve ayrıca bir oturum anahtarı olarak adlandırılır. Mesaj ve oturum anahtarı

alıcıya gönderilir. Oturum anahtarı alıcıya gönderilmelidir, böylece mesajın şifresinin nasıl çözüleceği bilinir, ancak aktarım sırasında onu korumak için alıcının açık anahtarı ile şifrelenir. Sadece alıcıya ait özel anahtar, oturum anahtarının şifresini çözebilir.

PGP, mesaj doğrulama ve bütünlük kontrolünü destekler. İkincisi, bir mesajın tamamlandığından (mesaj bütünlüğü özelliği) ve eskiden gönderenin (dijital imza) olduğunu iddia eden kişi veya kuruluş tarafından gerçekten gönderilip gönderilmediğini belirlemek için bir mesajın değiştirilip değiştirilmediğini tespit etmek için kullanılır. İçerik şifrelendiğinden, iletideki herhangi bir değişiklik şifre çözme işleminin uygun anahtarla başarısız olmasına neden olur. Gönderen, RSA veya DES algoritmaları ile mesaj için dijital imza oluşturmada PGP'yi kullanır. Bunu yapmak için, PGP düz metinden bir hash (bir mesaj özeti de denir) hesaplar ve daha sonra göndericinin özel anahtarını kullanarak bu hashtan gelen dijital imzayı oluşturur. Şekil 5' de bu model genel olarak gösterilmiştir.



Şekil 5. PGP programın genel modeli

1.3.3. Kaotik Sistemleri

Günümüzde, kriptografide kullanılan en önemli yöntemler kaos sistemleri gibi doğrusal olmayan sistemlere dayanmaktadır [55-58]. Bu sistemler, başlangıç noktasına güçlü duyarlılık göstermesi açısından görüntüleri şifrelemek için verimli bir şekilde kullanılır. Ayrıca, bu sistemleri birleştirerek, tahmin etmek ve analiz etmek zordur, bu da, bu yöntemin şifrelenmesindeki gittikçe artan popülaritesine yol açan, sözde rasgele bir sayı zinciri üretilebilir. Genel olarak, kaotik sistemlerin kullanılmasının şifreleme sistemi güvenliğini artırabildiğini söylemek mümkündür [59].

Kaos sistemleri, başlangıç koşullarına ve rastgele davranışların hassasiyetleriyle orantılı değildir. Kaotik sinyaller gürültüye benzer bir görünüme sahiptir ve bu rastgele davranışlarına rağmen başlangıç değerlerine ve haritalama fonksiyonlarına ait önceden üretilmiş değerler elde edilebilir. Eğer bu sistemler Lyapunov üstel denklem koşullarına uyarsa kaotik durumdadır. Bu sistemlerdeki sözde rastgele özellikler, şifreleme yapmak için güçlü bir alternatif oluşturmaktadır.

Kaos teorisine dayanan birçok şifreleme algoritması ve yöntemi vardır [60]. Diğer şifreleme yöntemleri bazı özel algoritmaları kullanarak kaos fonksiyonlarından üretilen rastgele sayıların bir kısmı ile düz verileri şifrelenmiş veriye dönüştürür.

Kaotik şifreleme sistemleri ve bu sistemlere saldırmak, şifrelenmiş verileri çözmek için oluşturulmuş sistemlerin de çeşitli yöntemleri bulunmaktadır.

Kaotik tabanlı şifreleme sistemlerinde kilit nokta, başlangıç koşullarının ve kaotik özellikler sergilediği alanların incelenmesi ve analiz edilmesi için bir haritalama fonksiyonunun belirlenmesidir. Haritalama fonksiyonu için doğru ve uygun seçim şifreleme sisteminin verimliliğinin artmasına ve korsanlara karşı şifrelenmiş verilerin daha fazla güvenliğine yol açacaktır.

1.3.3.1. Lojistik Harita (Logistic Map)

Lojistik haritalama fonksiyonu, Pierre Franois Verhulst tarafından 1838 yılında bir model olarak önerilen doğrusal olmayan sistem işlevlerinden biridir [61]. Sonraki yıllarda bu fonksiyon rasgele sayıları [62, 63] üretmek ve görüntüleri S-kutu tablolarına alternatif olarak şifrelemek için kullanılmıştır [64,65]. Fonksiyon denklem 1 olarak formüle edilmiştir:

$$x_{n+1} = a.x_n.(1-x_n) \ , \ a \in [1,4] \text{ and } x_n \in (0,1] \quad (1)$$

Bu denkleme göre, işlev parametresi ve 'a' nın farklı değerlerine göre sistem farklı davranışlara sahip olabilir. Başlangıç değeri olan x_0 ın aralığı (0,1] dir ve şifreleme sistemindeki bir anahtar değerle hesaplanabilir. Lojistik haritalama fonksiyonunun 1. presibinde lojistik harita 2D formatındadır ve denklemi şöyledir;

$$\begin{cases} x_{i+1} = \alpha_1 x_i(1-x_i) + \beta_1 y_i^2 \\ y_{i+1} = \alpha_2 y_i(1-y_i) + \beta_2(x_i^2 + x_i y_i) \end{cases} \quad (2)$$

$A_1 \in (2.75, 3.4)$, $A_2 \in (2.75, 3.45)$, $\beta_1 \in (0.15, 0.21)$, $\beta_2 \in (0.13, 0.15)$ değerleri bu aralıklardayken kaotik davranışları vardır ve iki kaotik duruma neden olur.

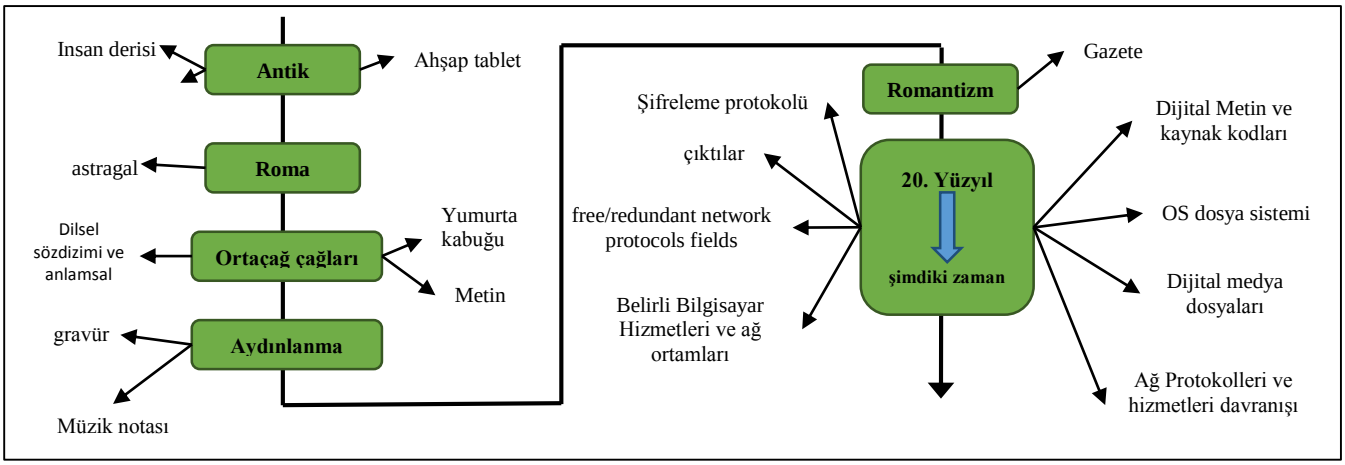
1.4. Veri Gizleme Bilimi

Bilgi veya veri herkes için çok önemli kaynaklardır. Bu yüzden onları korumak aynı zamanda önemli ve zordur. İletişim medyası güvenli değildir ve herhangi biri ona ulaşabilir, bu yüzden veriyi gizli tutmak için başka yöntemlere ihtiyaç duyarız. Bilginin saklanması önemli bir role sahiptir ve bilgilerin gizlenmesi için yöntemler sağlanmıştır, böylece bilgi bir insan veya makine için okunamaz hale gelir.

Veri gizleme biliminin diğer önemli konusu telif hakkı korumasıdır (Sahipliği Onayla). Bilgi gizleme, kopyanın (bilginin) asıl sahibini bilmek ve kimin kopya yaptığını belirlemek için kullanılır. Böylece gizli mesajlar kriptografi kullanmadan gönderilir, kopyalama önleme veya kontrol kimlik doğrulaması ve meta veri gizleme (izleme bilgilerini saklamak) yapılır.

Bilginin saklanma yöntemleri iletişim kadar eskidir ve veri gizleme yöntemlerini kullanmanın birkaç yöntemi vardır. Bu yöntemlerden biri Roma dönemine kadar uzanır. Askerin saçı kesilir ve üzerine gizli mesaj yazılır, sonra askerin saçları uzayana kadar beklenir ve diğer tarafa gönderilir ve gizli mesajı görmek için yeni bir saç kesimi vardır. Başka bir hikaye, Demeratus adlı bir askerin Xerxes mesajını Sparta'ya göndermek istediğini ve bu mesajın Yunanistan'ın işgali anlamına geldiğini söylüyor. Xerxes mesajı, mumla kaplı tablet üzerine yazılmıştır. Demeratus, gerekli mesajı bir panele yazdı ve sonra onu gönderecek boş bir tablet gibi görünmesi için onu mumla kapladı. Antik Romalılar tarafından kullanılan yaygın yöntemlerden biri, normal mesajlar gönderdikleri ve gizli mesajların satırlar arasında yazıldığı gizli (görünmez) mürekkeptir. Bunlar, meyve suyu, süt vb. gibi gizli mürekkepler kullanılarak yazılır. Bu malzemelerle yazarken, hiçbir şey yazılmamış gibi görünür, ancak ısıtıldığında, karanlık olur ve gizli mesajı okuyabiliriz.

Modern kriptografinin kurucularından biri olarak kabul edilen Johannes Trithemius, şüpheli metinler içinde gizli mesajların saklanması için kapsamlı bir sistemi anlatmaktadır (Steganografi). Geçmişte kullanılan birçok yöntem (Şekil 6'de olduğu gibi) vardır ve modern yollarla benimsenmiştir.



Şekil 6. Bilgi gizlemenin evrimi

1.4.1. Genel Bakiş

Gizli bilgilerin güvenliği, geçmiş zamanlardan günümüze kadar her zaman önemli bir konu olmuştur. Araştırmacıların alıcıdan başka kimseye göstermeden verileri göndermek için güvenli teknikler geliştirmeleri her zaman ilginç kalmaya devam etmiştir. Bu nedenle zaman zaman araştırmacılar verilerin güvenli transferini gerçekleştirmek için birçok teknik geliştirmişlerdir; steganografi ve filigranlar bunlardan bazılarıdır.

Son zamanlarda, önemli bilgileri korumak için çeşitli yöntemler geliştirilmiştir. Geliştirilen yöntemler iki kategoriye ayrılabilir: steganografi ve filigran. Hem steganografi hem de filigran, veri yerleştirme yöntemleridir.

Steganografi, metin, görüntü, ses ve video gibi multimedya taşıyıcıya çok miktarda gizli veri yerleştirmeyi amaçlamaktadır. Öte yandan, esas olarak telif haklarını kanıtlamak için kullanılan damgalama, bir multimedya taşıyıcısında az miktarda gizli veriyi gizlemeyi amaçlar. Steganografi ve kriptografinin ortak bir amacı olmasına ve ilgili kavramlara sahip olmasına rağmen, her ikisinin de kullanımı ve yolu biraz farklıdır.

Çoğumuz kriptografinin bilgiyi gizlemeye bağlı olduğunu bilir; verileri bir şifreleme anahtarı ile şifreleriz ve sadece şifre çözme anahtarına sahip olanlar şifrelenmış veri içeriğini bilebilir. Ancak bu şekilde yetkisiz kişiler hala bu şifrelenmış bilginin varlığını bilebilirler. Bazen bu zararlı olabilir çünkü herhangi bir gözlemci iletim işlemlerini yine de izleyebilir, meta verileri belirleyebilir ve kaydedebilir, hatta şifrelemeyi kesmeye çalışabilir.

Bilginin gizlenmesinin amacı bunun ötesine geçmek iletimin ya da kaydetmede bilginin varlığını gizlemektir ve bilgiyi tespit etmeyi zorlaştırmaktır. Steganografi, Filigran

(watermarking), Gizli Kanallar (covert channels), Anonimlik… vb. “Bilgi Gizleme” terimi altında toplanabilir. Bilgi gizlemesiyle ilgili ilk uluslararası çalıştayda kabul edilen sınıflandırma Şekil 7'de gösterilmiştir.

Şekil 7. Bilgi gizleme sınıflandırması

Şekil 7'ye bakıldığında bilgi gizleme teknikleri dört kategoriye ayrılır:

- Gizli Kanal
- Adsızlık (Anonymity)
- Telif Hakkı İşaretlemesi
- Steganografi

1.4.1.1. Gizli Kanallar

Gizli kanallar, izin verilen sistem kaynaklarını kullanarak ancak gizli (farkedilemez) ve yasa dışı bir şekilde iki taraf arasında veri iletme ve aktarmanın bir yoludur. Steganografide olduğu gibi, bir taraftan diğerine normal bir fotoğraf gönderilir, ancak bu fotoğrafta, sadece amaçlanan tarafların görebileceği gizli bir mesaj vardır.

Gizli kanalın, bu tür bilgileri göndermek için tasarlanmamış bir kanaldan gizlice bilgi gönderen bir mekanizma olduğunu söyleyebiliriz. Veya sızan bilgileri normal içerikten erişilebilen bir dosya içinde saklar. "Bilgi gizleme" terimi yeni terimdir ve bu terimden önce "gizli kanallar" terimini kullanılıyordu. Lampson bir iletişim kanalı hiçbir şekilde bilgi aktarımı için tasarlanmamışsa ya da istenmemişse, bu kanal gizlidir. Gizli kanallar farklı çeşitlerde bulunurlar:

1.4.1.1.1. Depolama Kanalları

Gizli mesajın, hem gönderenin hem de alıcının bu konuma veya kaynaklara erişebildiği depo yerinden yazma ve okuma yoluyla gönderme yöntemleridir. Bu, gönderici işleminin paylaşılan depolama konumuna ve ondan okunacak alıcı işlemine, gizli iletinin varlığını kimse fark etmeyecek şekilde yazması anlamına gelir.

1.4.1.1.2. Zamanlama Kanalları

Mesajı başka bir kelimeye ve zaman kavramına göre, belirli bir bilgiyi gönderme, göndericiden alıcıya sinyal verilmesi anlamına gelir. Örneğin, göndericinin süreç kontrolü, alıcının mesajın gerçekleşmesi için zamanlamadaki bir farklılığı gözlemleyebildiği bir şekilde olur. Bu iki tip arasındaki temel fark, birincisinin daha az zaman almasıdır, ancak depolama yerlerine yazdığı için arkasında bir kanıt bırakır ve daha sonra tespit edilebilir. İkincisi, hiçbir kanıt bırakmaz, ancak mesajı almak için sürekli dinleme ihtiyacı duyar.

Gönderen süreci diğer bir deyişle, sistem kaynaklarının kendi kullanımını değiştirerek, ikinci işlem tarafından gözlemlenen gerçek tepki süresini etkileyecek şekilde bir başka bilgi gönderir. Gürültü, kanallar üzerinde büyük etki yarattığı için iletişimin önlenmesi veya bilgi aktarım oranının azaltılması için bir araç olarak kullanılabilir.

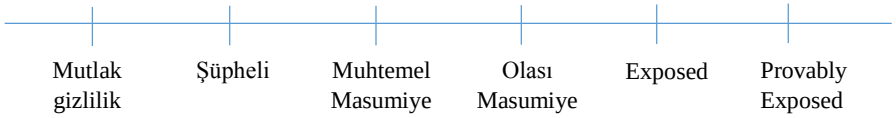
1.4.1.2. Adsızlık (Anonymity)

"Adsızlık" veya "isimsizlik" anlamına gelen Yunanca kelimeden (anonim) türetilmiştir. İletişim dünyasında Anonimlik kimlik gizleme teknikleridir. Anonimlik, bir kişinin gerçek kimliğinin bilinmemesini ve kişinin internette takip edilmesini engeller. Örneğin, anonimlik size gizliliğinizi korumak ve World Wide Web'de serbestçe dolaşmak için bir takma ad verir. Anonimlik, bir mesajın gerçek sahibini tespit etmek için çok zordur.

1.4.1.2.1. Anonimlik Seviyeleri

Anonimik Şekil 8'de gösterildiği gibi bir kaç seviyede olabilir.

- Şüpheli (Beyond Suspicion): Bir gözlemci veya saldırgan gönderilen bir mesajın varlığını veya kanıtını not edebilir, ancak hangi sistem kullanıcılarının mesajın göndericisi olduğuna karar veremez.
- Muhtemel Masumiyet (Probable Innocence): Gönderici diğer gönderenlerden daha çok kaynaklıdır, ancak gönderenin gönderici olmadığı bir % 50 olasılık vardır.
- Olası Masumiyet (Possible Innocence): Gönderenin gönderen kişi olması daha muhtemel görünmektedir, ancak kaynağın başka biri olması önemli bir olasılıktır.



Şekil 8. Anonimik seviyeleri

1.4.1.3. Telif Hakkı İşaretlemesi

Dijital medyanın başkaları tarafından kopyalanmasının kolay olması nedeniyle bu dijital dünyadaki bilginin sahibinin telif haklarını garanti altına almak için çalışma yöntemleri ve tekniklerine yol açmaktadır; Genel olarak, telif hakkı iki kategoriye ayrılmıştır:

- Filigran
- Parmak izi.

1.4.2. Filigran (Watermarking)

Gerçek dünyada olduğu gibi dijital dünyada bir filigran vardır. Filigran, dijital damga (sahiplik bilgileri) dijital dosyaya (görüntü, video ve ses) veya dosyaların telif hakkı bilgilerini tanımlayan sinyale gömme işlemidir. Dijital damga herhangi bir dijital veri tipi ve hem görünür hem de görünmez olabilir.

Alice'in göndermeden önce Bob'a bir görüntü göndermek istediğini varsayalım, Alice görüntüyü filigran edecektir. Görüntü Bob'a gittiğinde, Bob kopyasını almak istiyorsa ve yayınladığını iddia ederse, o zaman Alice filigranı Alice'in telif haklarını koruyacaktır, böylece Bob onun görüntünün kendisi olduğunu iddia edemez.

Genel olarak, gizli mesaj kapak hakkında bir şeyler söyler ve kapak korumaya ihtiyaç duyan en önemli şeydir. “Filigran” terimi, “wassermarke” Almanca teriminden türetilmiştir. Filigranın markanın yaratılmasında hiçbir önemi olmadığı için, bu isim muhtemelen verilir çünkü işaretler suyun kağıt üzerindeki etkilerine benzer. Filigran, bir kapağın içine bilgi

yerleştirme işlemidir ve bu bilgiler, telif hakları, seri numarası, kimlik doğrulama kodu, parmak izi vb. kapak hakkında bir şeyler açıklar.

Steganografinin temel amacı, aktarılan dosyayı elde etmek için kapak verisindeki mesajını gizlemektir. Bu nedenle, aktarılan dosyada gizli bir mesajın varlığını belirlemek için dinleyicinin bulunması zordur. Filigrandaki ana amaç, aktarılan dosyayı elde etmek için mesajı (damga) kapağa gömmektir. Bu nedenle, aktarılan dosyadaki gizli mesajı kaldırmak veya değiştirmek için dinleyicinin bulunması zordur.

Filigran için temel olarak aşağıdaki sınıflandırmalar yapılabilir:

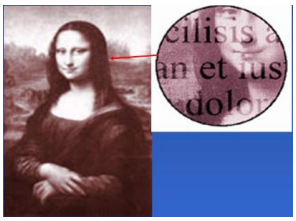
1.4.2.1. Görünür Filigran

Mülkiyet ve telif hakkı korumasını göstermek için kullanılır. Bu tipte filigran izleyicinin açıkça dikkatini çeker ve onu ayırt etmek zordur. Görünür filigran Şekil 9'da gösterildiği gibi katı veya saydam olabilir.

Şekil 9. Görüntü Filigran

1.4.2.1.1. Görünmez Filigranlama

Dijital damga (filigran), görüntüde görülemeyecek şekilde (video, ses), örneğin filigranı görüntünün LSB'sinde saklayacak şekilde buraya gömülür. Görüntü insan gözüyle fark edilmeyecektir. Görünmez filigran, filigranın kendisi hakkında bir bilgi olabilir. Bazen gizli verilere veya gizli filigranlara erişmek için anahtar kullanmalı ve bu tip filigran anahtar olarak adlandırılmalıdır. Filigranın başka sınıflandırması da (Şekil 10) olabilir:



Şekil 10. Filigranın başka sınıfları

1.4.2.1.1.1. Sağlam

En iyi tiplerden biridir. Onu manipüle etmek, yok etmek veya kaldırmak zordur. Telif hakkı koruması için kullanılır. Kanal sahibinin kendi logosunu koyduğu televizyon kanalında bulunabilir.

Bazı filigran uygulamalarının sağlam olması için filigranı gerekir (kaldırmak zor). Bu eleme işlemleri geometrik bozulma veya filtreleme ile gerçekleştirilebilir. Yani, saldırılar uygulansa bile sağlam filigran hayatta kalmalıdır.

Ancak diğer uygulamada, filigranın kimlik doğrulaması için kullanılan filigranlar gibi kırılgan (kaldırılması kolay) olması gerekir.

1.4.2.1.1.2. Semifragile

Görüntü doğrulama ve bütünlük doğrulamasında kullanılır. Gevşek sıkıştırmanın görüntü işlemesine karşı güçlüdür, ancak görüntü içeriği değiştiğinde bu yöntem kırılgan olacaktır.

1.4.2.1.1.3. Fragile

Görüntü içeriğindeki küçük değişikliklere çok duyarlıdır. Herhangi bir değişiklik filigranı yok edebilir ve görüntü bütünlüğünü etkileyebilir.

1.4.2.1.1.4. Saf / Gizli anahtar / Genel Anahtar Filigran

Saf filigranda algılama için anahtarın olması gerekmez. Gizli anahtar filigranı, hem yerleştirme hem de algılama işlemi için gizli bir anahtara ihtiyaç duyar. Genel anahtar şemaları, algılama işlemi için gömme ve gizli anahtar için bir gizli anahtar kullanır.

Filigran teknikleri steganografi tekniklerinde olduğu gibi aynı prensibe sahiptir. Bu tekniklerden bazıları aşağıda verilmiştir:

1.4.2.1.1.5. Uzamsal Alan (Spatial Domain)

Yüksek taşıma kapasitesi ile frekans uzayından daha az karmaşıktır.

LSB: uygulaması basit ve daha az zaman alır. Görüntü pikselinin en az anlamlı bitini filigran ile değiştirerek uygulanır. Bu tekniğin birçok dezavantajı vardır, basit manipüle bile filigranı kaldırabilir veya yok edebilir.

1.4.2.1.1.6. Frekans Uzayı (Frequency Domain)

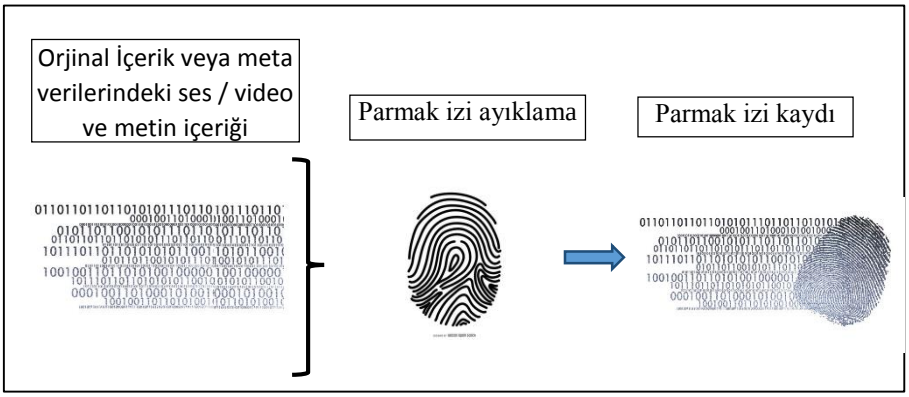
Bu teknikte sinyal, filigranı yerleştirmeden önce uzamsal alandan frekans alanına dönüştürülür. DCT (Ayrık Kosinüs Dönüşümü): Mekansal alan tekniğinden daha iyi ve hızlıdır. Bu teknikte, görüntü, farklı frekans bantları düşük, orta ve yüksek olmak üzere ayrılmıştır. Filigran düşük frekans uzayına gömülü ise filigran insan gözüyle fark edilebilir. Yüksek frekansa gömülü ise kenarları ile birlikte yerel bozulmaya neden olabilir. Orta frekans bandı, yerleştirme için en iyi bölgedir, görüntü kalitesini etkilemez. Bu teknik sıkıştırma, gürültü ve benzeri saldırılardan kurtulabilir.

- DWT (Ayrık Dalgacık Dönüşümü)
- FT/DFT (Ayrık Fourier Dönüşümü)

1.4.2.1.2. Parmak İzi

Parmak izi, altındaki şeklin gözlemlenmesinde, parmak izini damgalamadan farklıdır. Filigran tekniğinde Filigran, içeriği tanımlamak için dijital dosyaya veya sinyale gömülüdür. Parmak izi tekniğinde parmak izi, gelecekte kullanılacak içeriklerin karakterini belirtmek için dijital dosyadan çıkarılır.

Bu teknik, içerik sahibinin, telif hakkıyla korunan içerikleri üzerinde tanımlama ve izleme yaparak kontrolünü sağlar. Parmak izi dijital veriden (dosya) çıkarılır ve küçük bir boyuta sahip olduğu için daha sonra kullanmak için veritabanında saklanır.



Şekil 11. Parmak izi modeli

Şekil 11 deki parmak izi sistemi aşağıdaki şekilde çalışır:

- Özelliklerin çıkarımı: Dijital içeriğin parmak izi ve haklara kayıtların yanı sıra dosya meta verileri ile birlikte kayıt.
- İçeriğin tanımlanması: İhlali kontrol etmek için veritabanında parmak izleriyle karşılaştırma.
- Sonuç: Karşılaştırmaya dayalı olarak, üzerinde anlaşılan iş kuralları uyarınca uygun eylemi gerçekleştirir (blok, silme veya yetkilendirme).

Filigranlama, içeriğinize yerleştirme veya tanımlama konusunda yardımcı olacak bir tekniktir ve tek bir içerik için yalnızca bir filigranımız vardır. Parmak izi, dosyalarınıza kolayca erişilebilmesini ve diğer dosyalarla karşılaştırılabilmesini sağlar. Bu durumda bir içerik için ona çok sayıda parmak izi olabilir.

1.4.3. Steganografi

Bilginin gizlemesinin en önemli kısmı steganografidir. "Kapalı yazı" anlamına gelen Yunanca sözcüklerden türetilmiştir. Steganografi ile kriptografi arasındaki fark, mesajın sadece şifrelenmesi yerine mesajın varlığını gizlemesidir. Steganografi modern bir şey değildir, antik çağlardan beri vardır. Örneğin, Demeratus, Pers kralı Xerxes tarafından yakın bir Sparta istilası hakkında bir uyarı mesajı göndermiştir. Balmumu yazı tahtasından çıkarmış ve mesajını yazmış, sonra boş bir ahşap tahta gibi görünmesi için yine balmumuyla kaplamıştır.

Bir mesajın bir diğerini gizleme sanatıdır, böylece gizli mesajın varlığı gerçekleşemez. Steganografinin arkasındaki anahtar kavram, gönderilen mesajın kişi tarafından tespit edilememesidir. Steganografi (gizli verileri gömmek için kullanılan kapak medyasına bağlı olarak) görüntü, metin, ses, video ve ağ protokolü steganografisi olarak sınıflandırılabilir.

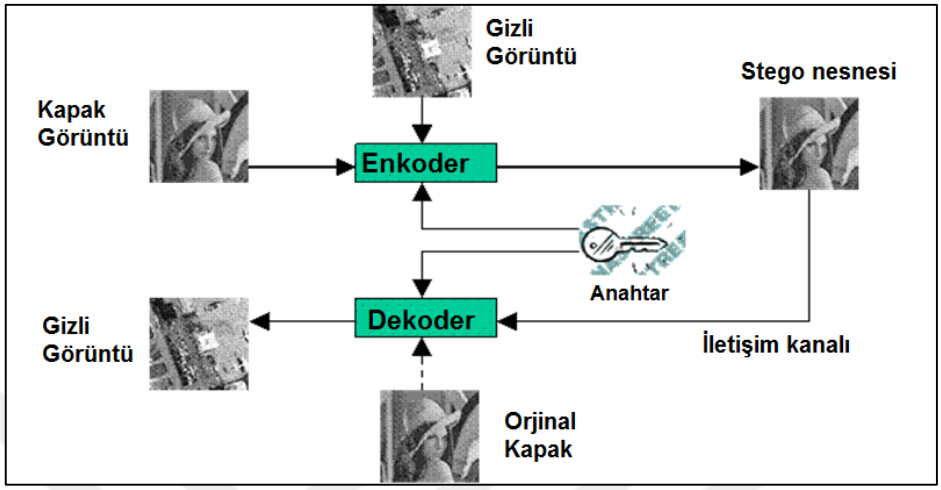
Etkili bir Steganografi tekniği aşağıdaki özelliklere sahip olmalıdır:

- Gizlilik,
- Emperceptibility: Belirlenememe yeteneği,
- Kapasite: Gizli mesajın maksimum boyutu,
- Doğruluk: Gizli verilerin çıkarılması doğru ve güvenilir olmalıdır.

Steganografi, mesajı korumak ve mesajın varlığını gizlemek için bir ortama veya sinyale (metin, resim, ses, video, 3D nesne, vb.) yerleştirerek saklamak veya göndermek istediğiniz bir mesajı gizleme işlemidir.

Mesaj steganografisi genel modelini aşağıdaki şekilde açıklayabiliriz (Şekil 12); gizli görüntümüzün olduğu ve hedefin, diğer görüntünün (kapak resmi) içine gizlenmiş alıcıya

gönderilmesi. Gizli görüntüyü, kapak resminin içine yerleştirmeden önce şifrelemek için paylaşılan bir anahtar kullanacağız. Şifreleme işleminden sonra ortaya çıkan stego nesnesini, iletişim kanalından alıcıya göndereceğiz. Alıcı gizli görüntüyü çıkarabilir ve aynı paylaşılan anahtarla şifresini çözebilir.



Şekil 12. Steganografinin Genel Modeli

Kapak her zaman bir görüntü değildir, herhangi bir dijital ortam (Ses, Görüntü, Video, Grafik, Metin, Yazılım, Zamanlama veya Ağ trafiği gibi Dijital olaylar) olabilir.

1.4.3.1. Tarihçe

Steganografi, gizli mesajı, hedeflenen alıcı ve gönderenden başka kimsenin varlığından bile şüphe etmeyeceği şekilde gizlemenin sanatı ve bilimidir. Steganografi, 25. yüzyıldan beri çeşitli şekillerde kullanılmaktadır.

1.4.3.1.1. Eski Yunanlılar ve Romalılar

Antik Yunan tarihçi Herodotus, bir Yunan tiranı olan Histiaeus'un hikayesiyle ilk kaydedilen anıdan sorumludur. En güvenilir kölesinin kafasını traş ederek kafatasına bir gizli mesaj yazmış ve habercinin saçları geri gelene kadar beklemesini söylemiştir. Elçi gönderildikten sonra, Histiaeus'un iletişimiyle buluşmak için kölesini tekrar gizlice uyandırmıştır. Burada, gizli mesajın içeriğini (yani Fars istilasıyla ilgili uyarıları, vb.) ortaya çıkarmak için başını traş etmiştir.

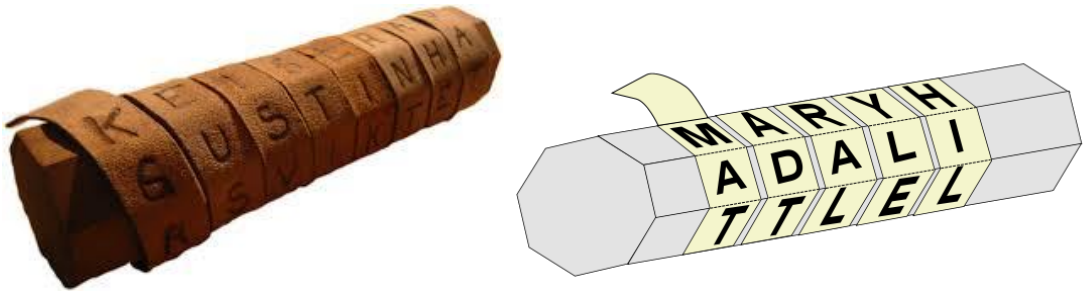
Bir başka erken kayıt, gizli mesajları gizlemek için kullanılan balmumu tabletlerini rapor eder. Bu süre zarfında, tabletler bir yazma yüzeyi olarak balmumunun tekrar kullanılabilirliği nedeniyle tercih edilmiştir. Ancak, balmumu katmanının altındaki ahşap üzerine gizli mesajlar yazılacak ve nereye bakılacağını bilenler tarafından kolayca erişilebilecektir (Şekil 13).

Şekil 13. Wax Tablet (Amazon)

Diğer örnekler arasında Yunan mitolojisinin ünlü Truva atı (Şekil 14), Yunan askerlerinin devasa tahta bir at üzerinde kendilerini gizledikleri yer almaktadır. Heykel, bir hediyenin kisvesi altında Spartan kampına getirilmiş, ancak askerler gece heykelin dışına çıkarak Spartalı askerleri uykularında öldürmüşlerdir.

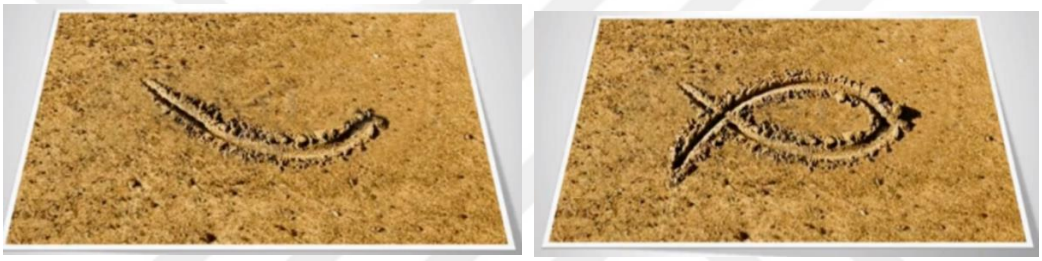
Şekil 14. Yunan Truva atı (greekboston.com)

Şekil 15'de görünen scytales, deri şeritler üzerindeki mesajların gizlenmesinin popüler bir yoluydu. Ne zaman ortaya konulursa, şerit rastgele bir harf grubu olarak görünecektir, ancak belirli bir genişlikte bir kadro etrafında dikkatli bir şekilde raptedildiğinde, harfler istenen mesajı belirleyecektir.



Şekil 15. Scytales (Wikipedia and wikimedia.org)

Erken Hıristiyanlar, günümüzde sıklıkla bir İsa balığı olarak adlandırılan ICTUS'un sembolünü kabul ettiler. Bu zamanlarda bastonlar yaygın olduğu için, eğer bir Hıristiyan bir yabancıyla konuşuyorsa, ICTUS'un bir yarısını toprağa çizecekti (Şekil 16). Yabancı aynı zamanda bir Hıristiyan ise, çizim sinyalini dostça niyetle bitirecekti.



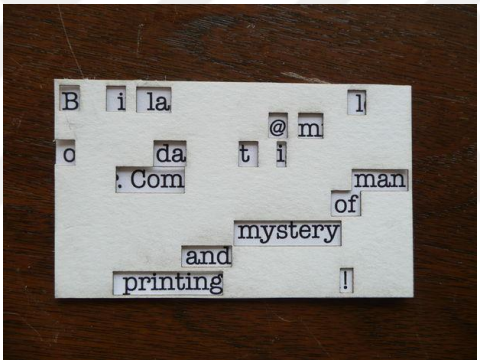
Şekil 16. ICTUS'nun sembollerinden (jesuswalk.com)

1.4.3.1.2. Orta Çağlar

Steganografi terimi, Alman başrahip Johannes Trithemius tarafından yayınlanan "Steganographia" başlıklı üçleme dönemine dayanmaktadır. Üçleme Latince yazılmıştı. İlk iki cilt, kriptografi sanatına ve gizli mesajların gizlenmesine odaklanmışken, bir bütün olarak, üçüncü ciltte ise yayınlandıktan sonra büyük miktarda okült ile ilişkili malzeme nedeniyle Katolik kilisesi kara listeye alınmıştır. 19. yüzyılın sonlarına doğru modern araştırmacılar, üçüncü kitapların sayı şemaları ve kara büyücülükle ilgili görünüşleri ile dolup taştığını ve kitabın içeriğindeki önceki ciltlerin kriptografinin bir devamı olarak gizlediğini fark etmemişti (Şekil 17). Sayı çizelgeleri ve konuları, sadece kriptografinin değil, aynı zamanda steganografinin gücünü gösteren bir işlev gördü. Steganografinin geleceği takip edilecek ve muazzam çığır açan steganografik teknikler ortaya çıkacaktı.

Şekil 17. Steganografi kitabı (hermesantiquarian.h-e-r-m-e-s.org)

15. yüzyılda Gerolamo Cardano, "bir ızgara kullanarak gizli mesajların gizlenmesi için bir yöntem önerdi. Şekil 18'de gösterilen kardan ızgarası, yazı yazmak için açıklıklara sahip bir parşömen tabakasıdır. Metnin tamamı, masum görünen metne dönüştürülen steganogramı oluşturur".



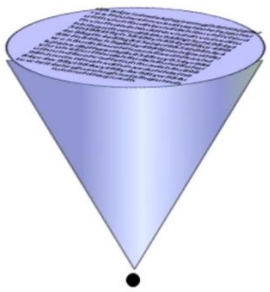
Şekil 18. Cardan grille (instructables.com)

Görünmez mürekkep, özellikle Amerikan devrimi sırasında 18. yüzyılda popüler hale getirildi. Gizli mesajlar, bir yazarın süt, meyve suyu, şarap, sirke veya kimyasal ajanları kullanabileceği satırlar arasında, neredeyse tespit edilemeyen gizli mesajları gizlemek suretiyle okunmuştur. Gönderenler, tüyü küçük bir odun parçasına veya parmaklarını mürekkebe batırıp kağıda mesajı yazarlardı. Kuruduğunda, mesajı alıcıya gönderirlerdi. Mesajı aldıktan sonra, alıcı mektuba (örneğin bir mumdan) ısı uygulayacak ve mesaj görünür hale gelecektir; dolayısıyla okunabilir.

1.4.3.1.3. Daha Yakın Tarih

Erkek izci hareketinin kurucusu Robert Baden-Powell, İngiltere tarafından Boer savaşı sırasında bir izci olarak istihdam edildi. Boer'in topçu pozisyonlarının haritalarını kelebeklerin kanatlarında çizerdi. İşvereninin ihtiyaç duyduğu tüm istihbaratları rakipler tarafından fark edilmeden gönderebiliyordu. Bir başka steganografik teknik, yakın bir zamanda bir şüphelinin, ihanet olayı olarak adlandırılan, 1910'un sosyalist bir planına katılmamasını detaylandıran bir mektuptu. Mektup boş görünüyordu ama sansürden sakınma amaçlı yazılı bir mesaj oluşturmak için yazılmış küçük iğneler ortaya çıkardı.

Şaşırtıcı steganografik tekniklerden biri, dünya savaşları sırasında Alman ordusu tarafından icat edilen ve bugün hala kullanımda olan mikrodot (Şekil 19) teknolojisidir. Gizli metin mesajlarını, görüntüleri ve hatta diyagramları standart bir daktilodaki noktalama işaretiyle aynı boyutta bir noktaya gizleyebildiler. Alıcı mikrodottaki gizli bilgileri ortaya çıkarmak için bir büyüteç veya mikroskop kullanabilir.



Şekil 19. Microdot

19. ve 20. yüzyıllarda bir başka iyi steganografi tekniği null şifresiydi. Bu, gizli bir bilgiyi büyük bir metin gövdesinde saklamayı sağlar. Alıcı, gizli mesajın gömülü olduğu cümlenin belirli konumlarından (yani her bir kelimenin 2 ya da 3 harfinden) harf seçerek bilgiyi çıkarabilmektedir. Teknik, uzun saçma cümlelerin şüphe uyandırabilmesi açısından mükemmel sayılmazdı.

Spread spektrum teknikleri de bir steganografik teknik olarak kullanılmıştır. Bu teknoloji günümüzde çıkarım ve gürültüye dayanıklılık sağlamanın bir aracı olarak kabul edilmektedir. Ancak orijinalleri, 1950 yılına kadar Hedy Lamarr ve George Antheil, torpidolar için bir frekans atlama içeren telsiz sistemini önerdiğinde ortaya çıktı. Ana inovasyon, steganografi olarak düşünülebilecek bu kontrolün saptanamazlığıydı.

Vücut dili, özellikle şekil jestleri ve göz kırpmaları da bir steganografik teknik olarak kullanılabilir. 1960'larda ABD askerleri Kuzey Kore tarafından ele geçirildi, askerler iyi davranıldığını göstermek için propaganda fotoğraflarını çekmek zorunda kaldılar. Mürettebat, hükümetlerine mesaj göndermek için parmak hareketlerini kullandı ve Kuzey Koreliler mesajı hemen yakalayamadılar.

1.4.3.1.4. Bilgisayar Çağı

Bir mesaj ters sırayla oluşturulduğunda veya bir ses geriye doğru kaydedildiğinde ve bu bilgi, aksi halde oynatılmadıkça veya geriye bakılmadıkça hiçbir anlam ifade etmediğinde, Backmasking adında bir tekniğe ihtiyaç duyarız. Başka bir deyişle, mesaj veya ses geriye dönük olarak izlenir veya oynatılırsa, net bir bilgi elde edilebilir. Bu teknik, herhangi bir şüpheye yer bırakmadan alıcılara gizli mesajlar göndermek için kullanıldı, ancak günümüzde mükemmel ve güvenilir olmadığı kanıtlandı. Bu dijital çağda, bilgi dijital metin dosyalarına, videoya, görüntüye, hatta ses dosyalarına gömülebilir. Çıplak gözle neredeyse tespit edilemeyen orijinal kaplama nesnesinde değişiklikler yapan çeşitli teknikler ve algoritmalar geliştirilmiştir. Bununla birlikte, bazı yazılım uygulamaları bir kapak nesnesinin gizli bilgi taşıdığını ya da bir dereceye kadar taşıyamayacağını tespit edebilmektedir. Frekans uzayı, uzamsal alan ve aynı zamanda dil teknikleri genellikle günümüzde sahip olduğumuz güçlü makinelerle ilgilenen örtü nesnelerinde gizli bilginin varlığını tespit etmeye yarayan bazı gelişmiş steganografik teknikleri uygulamak için kullanılmaktadır.

Ağ steganografisi, aynı zamanda steganografi alanında da yeni ortaya çıkan bir trenddir. Bilgi, protokoller arası veya protokol içi steganografi ve telekomünikasyon ağlarındaki stegramlar kullanılarak iletilebilir. Inter-protokol birden fazla protokolü kullanırken, protokol içi bilgiyi gizlemek için sadece bir protokol kullanır. Yazıcı üreticileri, yazıcılarıyla basılmış kağıtlara seri numarası ve üretim numarası gibi yazıcı bilgilerini gizlemek için steganografi kullanır. Belirli desenlerde sarı renkli noktalar, yalnızca mavi ışık altında görülebilen kağıtlara yazdırılır. Bu teknik, sahtecilik takibinde çok yardımcı olmuştur.

Sonuç olarak steganografik teknikler zamanla çok yönlü olmasına rağmen, bu bilimi yöneten ana ilke aynı kalmıştır. Tüm gizli mesajlar kapak nesneleriyle gömülür; dolayısıyla iletişim gizli kalmıştır.

1.4.3.2. Steganografinin Yarar ve Amacı

Daha önce de belirtildiği gibi, steganografinin ana hedefi, gömülü mesajların gizliliğini veya mesajların iletilmesini sağlamaktır. Steganografi iletilecek mesajları gizlemek için masum olmayan veya normal görünümlü medyayı kullanır. Amaç, bir gözlemcinin mesajın varlığını gerçekleştirmesini veya keşfetmesini engellemektir. İyi bir stego-medya (yani, görüntü, metin, video veya ses) orijinal ortamdan ayırt etmek zor olmalıdır. Stego medyasının sağlam olması da çok önemlidir. Bunun anlamı, stego-medyanın, birkaç manipulatif süreçten geçse bile, sağlam kalması yeteneğine sahip olması gerektiğidir.

Dürüstlük (Integrity): Veri bütünlüğü, bir miktar veri parçasının bazı referans sürümlerinden değiştirilmediğinden emin olunmasıyla ilgilidir. Sosyal medya ve sahte medyanın bu gündeki yaşamda çok önemlidir. Orjinal görüntüden farklı bir gündemi temsil etmek için değiştirilmiş birçok görüntü görülebilir. Steganografi, bir parçacık ortamının orijinal mi yoksa değiştirilmiş mi olduğunu tespit etmek için kullanılabilir. Gizli bir iletinin resimler gibi genel olarak paylaşılan bir içeriğin gizlenmesi, medyanın orjinal sahibinin ortamın photoshop aracılığıyla değiştirilip değiştirilmediğini belirlemesine izin verecektir. Bu bir filigran şekli olarak düşünülebilir.

Kimlik Doğrulama (Authentication): Kimlik Doğrulama, bir şeyin gerçekten iddia ettikleri şeyin olup olmadığını belirlemede yardımcı olur. Bu iletişim ve bilgi paylaşımında çok önemlidir. Gizli mesajların örnek videoların gömülmesiyle, bir medyanın gerçek olup olmadığını belirleyebiliriz. Aynı şeyi entelektüel mülklerin, tıbbi kayıtların vb. doğrulanmasına yardımcı olan filigran şeklinde steganografi kullanımıyla da görebiliriz.

Steganografi, bazen oldukça yeni olduğu düşünülse de modern bir şey değildir. Bunun kanıtları, eski Yunan'da, köleler arasındaki dövmelerin taraflar arasındaki iletişim için kullanıldığı yerlerde mevcuttur. Günümüzde steganografi veya stego-medyanın deşifre edilmesi birçok önemli uygulamaya sahiptir.

Steganografinin kullanım alanları aşağıdaki gibidir:

• Ağ iletişimi: Bir ağa sahip düğümler içindeki iletişim, veri aktarımı ve değişimi için çok önemlidir. Bununla birlikte, virüsleri ve diğer tehditleri yaymak için de kullanılabilir. Steganografik teknikler kullanılarak, tehditler gizlenebilir ve dolayısıyla güvenlik duvarı ve güvenlik ayarlarını atlayabilir.

• Güvenli veri iletişimi: Steganografinin kullanıldığı ana alanlar, verilerin güvenli aktarımını sağlamaktır. Askeri haberleşme alanlarında hassas bilgilerin veya istihbaratın iletilmesi önemlidir ki bu da kesişimden korunmalıdır ve bazen bilgi alışverişini gizlemek için çok önemlidir. Bu nedenle, steganografi ile kombine edilmiş kriptografi, bu gibi durumlarda birlikte kullanılabilir. Ne yazık ki, teröristler ve diğer suçlular, ceza infaz kurumlarını ve yasadışı eylemlerini bildirmek için benzer teknikleri kullanabilirler.

• Gizli anahtarların değişimi: Steganografi bazen gizli anahtarlar gibi gizli bilgileri iletmede yardımcı olabilecek iyi bir mekanizma olabilir. Kriptografi anahtarlarının paylaşımı için kriptografik iletişim sırasında steganografi kullanılabilir. Bu çok önemlidir, çünkü şifrelenmiş mesaj tuşlara sahip kişinin elinde ise, anahtarların elde edilmesi şifrelenmiş bir mesajın esasen çözülebilir olmasını sağlayacaktır. Bu nedenle, steganografi tarafından sağlanan mesaj iletiminin gizliliği, bu amaç için çok önemlidir.

• Veri değişikliğinin ve veri güvenliğinin korunması: Özellikle güvenli iletişim kanallarında veri doğrulaması konusu çok önemlidir, davetsiz misafirlerin kanalda yanlış bilgi vermemelerini sağlamak önemlidir. Ayrıca, bilginin mesajın başlatıcısından mesajın alıcısı üzerinden doğru bir şekilde güvenilir olmasını temin etmektir. Steganografi de kullanmaktadır

• Dijital içerik dağıtımının izlenmesi (Radyo Reklamlarının İzlenmesi): Dijital medyanın yayın oranını izlemek için steganografi kullanılabilir; radyo reklamı. Böyle bir ortama eklenen gömülü bir kod, yayıncının yayınlanmakta olduğu süreyi takip etmesini sağlayarak, reklam verenin ödedikleri tutar için değer aldığından emin olmasını sağlayabilir.

Bunun bir başka uygulaması da korsanlığın önlenmesi veya dijital içeriğin yetkisiz çoğaltılması için dijital medya kopya korumasındadır. Bu durumda, yetkisiz bir çoğaltma meydana geldiğinde izlemeyi etkinleştirmek için dijital ortama gizli bir mesaj yerleştirilir.

1.4.3.3. Mahkûm Problemi

Mahkûm problemi (Prisoner's problem) 1984'te Gustavus Simmons tarafından bir dönüm noktası olarak bilinen bir makalede iyi çalışılmıştı. Bugünkü resimleme, mahkûmu problemi açısından yaygın olarak steganografi olarak bilinmektedir (Dilemma).

Aynı yıl Gustavus Simmons, mahkûm problemi açısından genel olarak steganografi olarak bilinen şeyin kısa bir resmini çizdi. İki mahkûmu (yani Bob ve Alice) nerede kullandı? Her ikisi de ayrı hücrelerde. Bob ve Alice, kaçış planı hakkında birbirleriyle iletişim kurmak istiyor. Ve onların tek iletişim araçları hapishane amiridir (müdür). Bunu yapabilmeleri için,

aynı zamanda özel bir plan (bu durumda, herhangi bir veri güvenliği aracı) bulmak zorundadırlar. Denetim otoritesi, aralarında herhangi bir düzensiz iletişim tespit ederse, onları bir daha asla iletişim kuramayacakları bir maksimum tesise (hapse) götürecektir [66].

Yapılan birçok araştırmaya göre, bu mahkûm probleminin (iletişim araçları) günümüz dünya iletişim yolunda daha önce hiç olmadığı kadar yaygın bir şekilde kullanıldığına inanılmaktadır. Aynı sorun, bilgi açıklamalarına karşı savaşmaya çalışan vatandaşları andırmaktadır; her biri herkesin özgürlüğünü ve düşüncesini, onların izlenmesine gerek duymadan ifade etmek ister. Buna, bugünün dünyasında, 2001 yılında dünya ticaret merkezinin bombalanmasından sonra terörist lansmanına karşı yürütülen mücadele eklenebilir. O zamandan beri, güvenlik ajanlarının her iletişimi izlemek için başka seçeneği yoktur. Hâlihazırda, teröristler mahkum problemini başka yerlerdeki sokaklarla iletişim kurmak için kullanırlar [66].

Simmons'un yazdığı makalede mahkum probleminin çözümünde başka bir seçeneğin olmadığı gözlemciye (yani Bob ve Alice) dayanıyor, ancak sadece denetçiyi (müdür) aldatmanın bir yolunu bulmaktır. Bu noktada, gizlice iletişim kurmanın bir yolunu bulmak veya bulmaya çalışmak. Bunu başarmanın bir yolu, bilinçaltı kanalın kullanılmasıdır. Örneğin, Bob'un belirli bir datagram tarafından temsil edilen bir mesaj göndermeye çalıştığı bir durumda ve bu datagram sadece Alice ve Warden için kullanılabilir. Bu düşüncenin ardındaki sır, iletici mesajın farklı yorumlamalar içermesi veya müdürün Bob ile karşılaştırması için farklı bir anlam içermesidir.

1.4.3.3.1. Subliminal Kanal Kullanmanın Nedeni

Müdürün, İngilizce mahkemenin iki mahkumun (Bob ve Alice) iletişim aracı olarak İngilizce metnine izin verdiğini ve bazı İngilizlerin rastgele oluşturulduğu ihtimaline gözlerini vuracağını varsayalım. Bilgiyi iletmek için kullanılan metin araçlarını kullanmak, özellikle ingilizce alfabe kullanarak kolayca hata yapabilir. 26 harfi vardır ve İngilizce'yi kontrol ettiğimizde, 'E' çoğunlukla kullanılır. Ve bu kolayca izlenebilir.

1.4.3.3.2. Subliminal Kanal

Subliminal kanalla ilgili herhangi bir referans yapmadan mahkum problemini inceleyemeyiz. Önceki bölümümüzde, Bob ve Alice'den ayrı hücrelerden bahsetmiştik.

Hapishaneden kaçma planlarını kimsenin (müdür) planlarını tespit etmeksizin nasıl iletebilirler?

Subliminal Kanal nedir? Veri güvenliğinde aynı zamanda, güvensiz bir kanalda ya da yollarda tespit edilmeden haberleşmek için kullanılan bir dönüşüm kanalıdır. 1984 yılında Gustavus Simmons gazetesinde Subliminal kanalın sayısal imzalı kripto sistemi ile ilgili olduğuna değinilmiştir. Aynı zamanda, mahkumun problemi dijital imza algoritmalarındaki ikame parametresi aracılığıyla nasıl ele alınabileceğini de açıklar. [67]

Sistem güvenliği (bilinçaltı kanal), Bob'un kendisi ve Alice arasındaki sır üzerindeki gerçek anlam ve bağlam tabanını anlayabilmesi veya yorumlanması anlamında Warden'a karşı bir avantaj sağladığına dair belirli varsayımlara dayanır. Danışman, elindeki mesaji sadece kapak olarak anlayabilir. Bu durumda, Bob diğer mahkumun (Alice) mesajı nerede tuttuğunu bilmelidir. Bu noktada, çoğu sırrı kantinde ya da hatta cümlenin huzurunda paylaştılar.

Buna ek olarak, denetim otoritesi (bu durumda iletişim sistemi) bir iletişim kaynağı olarak steganografinin kullanıldığını tespit ettiğinde veya şüphe ettiğinde veya şüphelendiğinde, herhangi bir bilgiyi kendisine iletmek için güvenli olmayan bir iştir. Bu nedenle, sözdizimsel ayrıştırma problemi için çok önemli olduğu düşünülür.

1.4.3.3.3. Mahkum İkilemi (The Prisoner’s Dilemma)

İki oyuncu arasında sıfır olmayan bir oyun olarak tanımlanır. Sıfır olmayan toplamın anlamı, oyunculara uygulanan avantajların, diğer oyuncuya zorunlu olarak uygulanmaması anlamına gelir. Başka bir deyişle, kooperatif olmayan / paylaşım / işbirliği olarak da bilinir. Her oyuncu kendi başına hareket eder.

Bu problemde iki tür eylem vardır:

1. Özverili olmak ve

2. Bencil olmak

1. Özürsüz olmak, her ikisi de aynı sayfada yer almayı reddetmektedir. Bu durumda, her ikisi de iyi bir ödül verecektir.

2. Diğer yandan, birbirimize ihanet eden bencil bir anlam özgür olacaktır (bir diğerinin konuşmadığı bir durumda) veya her ikisi de sonuçlarla yüzleşecektir. [68]

1.4.3.3.4. Mahkum İkilemine Örnek

İki mahkumun probleminde, her birine ihanet etmeleri istendiğini varsayalım. Birincisi, yargıcın onları cezalandırmak için somut bir kanıtı yoktur. Her iki yerde de sorgulama için ayrı bir oda kullanma ve şartlar yerine getirildiğinde. İlk koşul:

I. Eğer A ibaresi B (ikinci mahkum) ve B sessiz kalırsa A özgürdür ve 3 yıl hizmet eder (tam tersi)

II. İkisi de (A ve B) birbirilerine ihanet ederse, her ikisi de 2 yıl boyunca cümle kuracaktır.

III. Ve eğer ikisi de sessiz kalırsa, her biri sadece bir yıl olacak.

İletişim hayatımızda çok farklıdır. Aynı zamanda dışarıda insanı aldatmaya çalışan, bir başkasının banka detaylarını bu kadar çok talep eden insanlar var. İletişim etiği, eksileri temelinde kendimizi eğitmek zorundayız. Her ülke vatandaşların özgür olmasını ister, ancak tam aksine bugün dünya artık güvende değildir. Terör, toplumdaki büyük seçkinler tarafından sadece bencilce faydası için destekleniyor. Bu noktada, güvenlik ajanları halkının iletişim yolunu dinlemekle suçlanamaz. Mahkûm problemi, iyi tarafları kadar da zararlı değildir. Söylendiği gibi "her kötü şeyin iyi bir yanı vardır".

Dahası, bu alanda daha fazla araştırma yapılması gerekmektedir (mahkum problemi, genel olarak veri güvenliği). Mahkûm ikileminde, bu birliğin / birlikteliğin nasıl güçlü olduğunu gösterir. Bencillik verimi kötü sonuç ve beraberlik mükemmel sonuç verir. Mahkûm problemine göre, bu konunun steganografi ile ilgili olduğunu ve Simmons gazetesinde, bu konunun steganografiye gelmeye başladığını öğrendik. Böylece, bu konuyu steganografiyi geride bırakarak inceleyebiliriz. Bu bizim steganografi ve özellikleri hakkında daha fazla bilgi edinmemizi sağlar.

1.4.3.4. Steganografinin Temel Modeli

Steganografi, kullanılan alana göre farklı şekillerde sınıflandırılabilir, şifreleme ve şifre çözme işlemlerine göre dönüşüm ve şifre çözme prosedürlerine göre kategorize edilebilir; Saf Steganografi, Kamusal Anahtar Steganografi ve Özel Anahtar Steganografi. Veri gizlemek için kullanılan ortam göz önüne alındığında, Steganografi şu şekilde gruplandırılabilir: Ses, Video,

Görüntü… vb. Son olarak, ve en önemlisi, Şekil 20'deki gibi veri gizleme tekniklerine göre sınıflandırılabilir: İkame sistemleri, Dönüşüm Alanı, Distorsiyon tekniği… vb.

Şekil 20. Steganografi'nin farklı yöntemleri

Bu yollar için, teknikler, ortamlar, steganografik sistem aşağıdaki özellikleri göz önünde bulundurmalıdır: Kapasite, sağlamlık ve algılanamazlık.

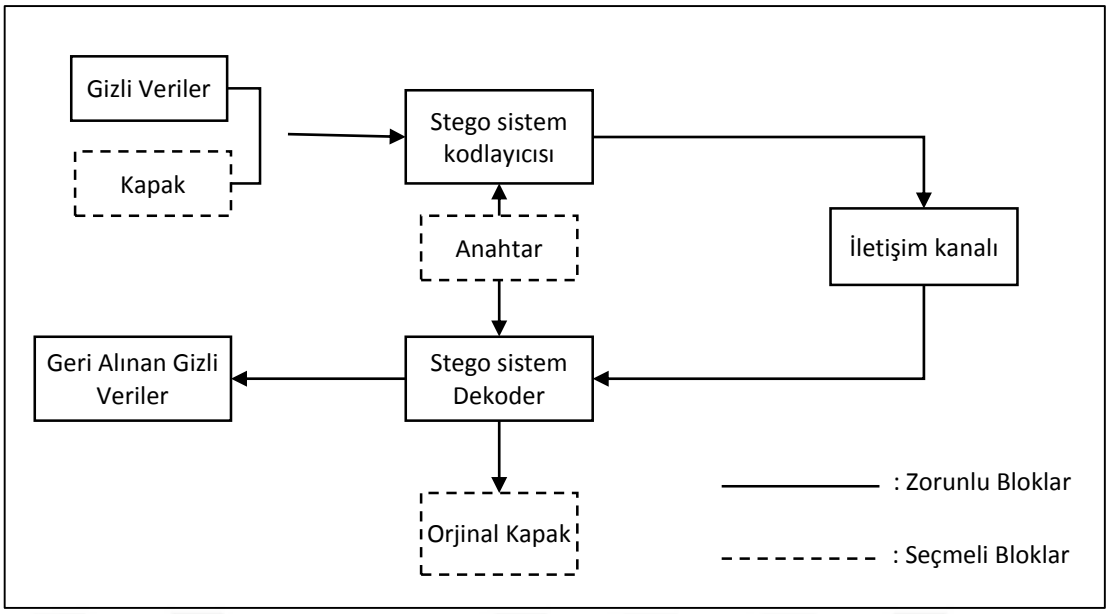
Kapasite: Kullanılan teknik, iyi miktarda verinin gömülmesine izin vermelidir.

Görünmezlik: Gizli verilerin ayıklanması ve hatta tahmin edilmesi kolay olmamalı ve orijinal ile stego görüntüleri arasında büyük bir fark olmamalıdır.

Sağlamlık: Kullanılan ortama istatistiksel analiz uygulandığında Stego nesneleri keşfedilmemeli veya tespit edilmemelidir.

Temelde, steganografi her türlü belli adımları takip ederek uygulanabilir, ancak bazı adımları bir yöntem (steganografi yöntemi) diğerine göre farklı olabilir, sürecinde (process) kullanılan teknik ve kapağa (cover) bağlı olduğu için, her yöntemde farklı süreci görebiliriz. Genelde, temel modelde, süreç şu şekilde ilerler: Bir kapak (cover) seçilecek (veya üretilecek) ve gizli bir mesaj, birlikte, bu kapakta, gizli bir mesaj enkode ettikten sonra kodlama işlemi (encoding) ile birlikte bir stego-object oluşturulacaktır.

Son olarak, kod çözme işleminde, kodlama sürecinde yapılanları tersine çevirerek, gizli mesajı Stego nesnesinden alınabilir. Başka bir güvenlik seviyesi ekleyebilmek için, kapakta gizli mesaj, gizlenmeden önce şifrelenebilir. Genel Steganografi Modeli Şekil 21'de gösterilmektedir.



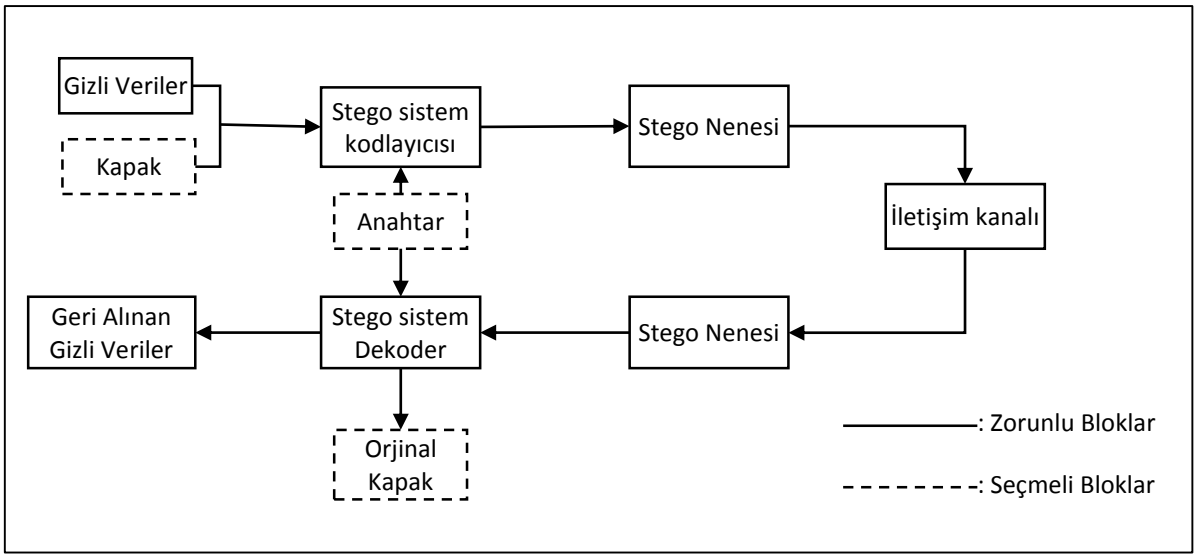
Şekil 21. Steganografinin Genel Modeli

Unutmamak gerekir ki, kapakta (cover) gizli mesaj kodlama işlemi, uzaysal alan'de ve frekans uzayında aynı olmayacaktır, Frekans alanı tekniğinde kapak ilk önce frekans uzayına (DCT, DFT,…) dönüştürülecek, daha sonra gizli mesaj kodlanacaktır.

1.4.3.4.1. Spatial Domain

Spatial domain'da, kapak (cover), frekans uzayı gibi başka herhangi bir uzaya dönüştürülmez. Bu nedenle, kapaktaki ana öğeler (bitler) asıl halinde doğrudan manipüle edilerek veriler (gizli mesaj) gizlenecektir. Spatial domain steganografisinin bir örneği (LSB), bu yöntemde kapağın bazı baytlarının son bitini, gizli mesajın bir kısmı ile değiştirilecektir. Örneğin, kapak bir görüntü dosyasıysa (image), görüntünün belirli piksellerin son biti, mesajın bir kısmı ile değiştirilir (Şekil 3'de gösterildiği gibi).

Spatial domain Steganografi genel modeli Şekil 22'de gösterilmektedir.



Şekil 22. Spatial alan steganografisinin Temel Modeli

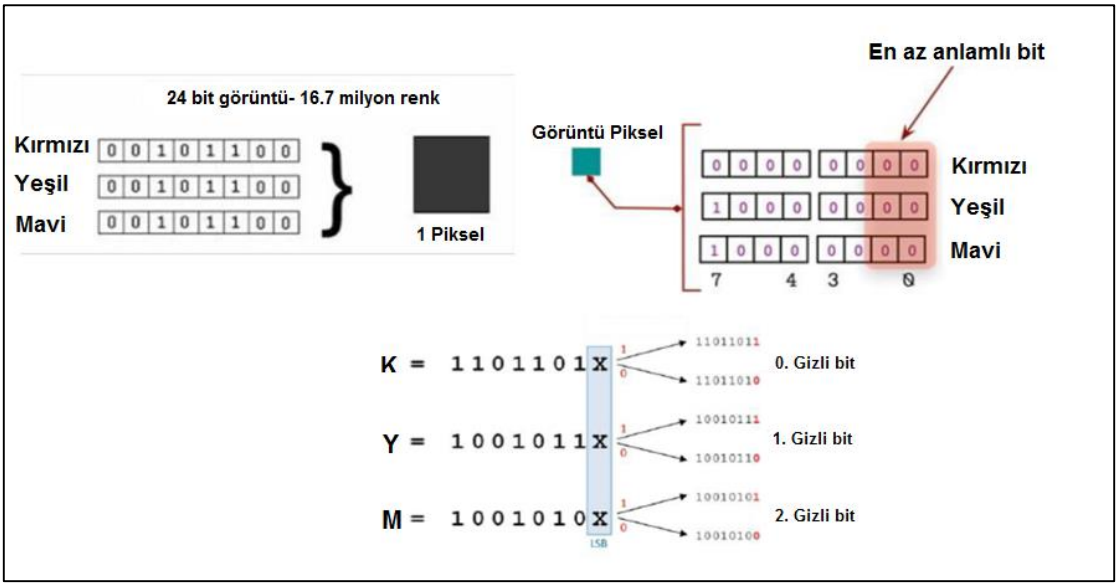
İlk olarak, bir kapak seçilecek veya üretilecek, bu kapakla gizli mesajın, kodlama işlemlerinden birini (ex: LSB) kullanarak bir gizli nesne (Stego-Object) oluşturulacaktır. Seçmeli blok (key), mesaj gizlemeden önce şifreleme imkanı gösterir, Kripto sisteminin sağlamlığını arttırabilmek için kullanılır. Haberleşme kanalını geçtikten sonra, gizli mesaj bir kod çözücü kullanılarak, stego nesnesinden çıkarılacaktır, bu kod çözücü (decoder), kodlayıcıda (encoder) işlemleri tersine çevirerek tasarlanmıştır.

Spatial domain steganografinin bir örneği LSB, LBS'de bir görüntü (image) kapağındaki metni (text) gizlemek için, kapağın belirli öğelerinde değiştirerek, mesaj bitlerle (0,1) gömülecektir. Bu yerine koyma sadece elementin en önemsiz bitini etkileyecektir, (Şekil 23), spatial domain steganografisinde görüntü (image) kapağında metnin (text) gizlenmesi sürecini göstermektedir.

Aşağıda LSB Kodlama işlemi açıklanmıştır:

Örneğin, gri bir görüntüdeki her pikselin 8-bit (0-255 arasındaki tamsayılardan) ile temsil edilecektir. Gizli mesajı kodlamak için, LSB algoritması seçilen piksellerin en önemsiz bitini, mesaj bitleri ile değiştirecektir (Şekil 23).

Bazı teknikler, iki veya üç en önemsiz bit kullanır, bu teknikler daha fazla kapasite sağlayabilir, ancak, steganografik sistemin sağlamlığı azaltılacaktır.



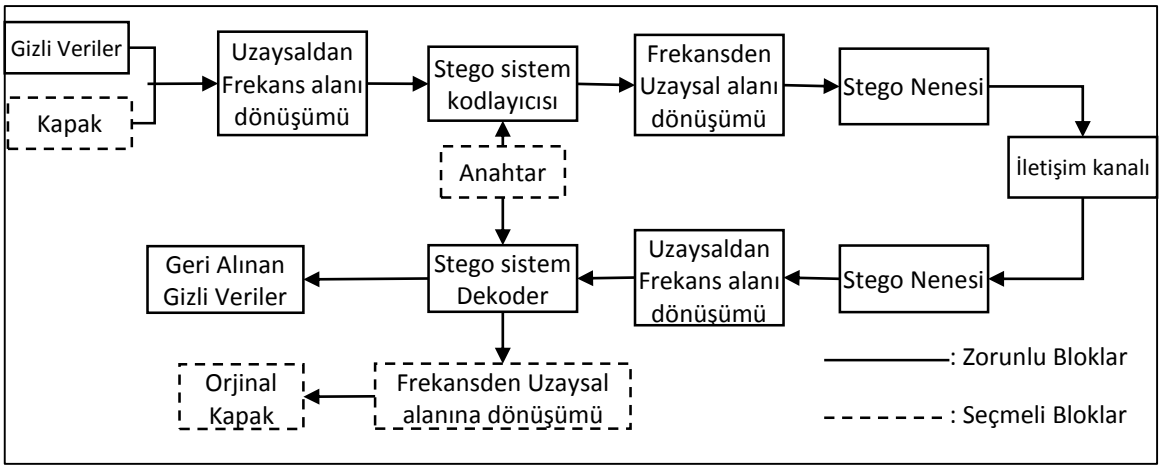
Şekil 23. LSB süreçleri

1.4.3.4.2. Frekans

Frekans uzayı steganografi içindeki süreç biraz farklıdır, burada gizli mesajı gizlemeden önce (Kodlama İşlemi), kapak uzaysal alan'dan frekans uzayına dönüştürülecektir. Mesaj sakladıktan sonra, şüpheli olmamak için, yine kapak uzaysal alan'a dönüştürülecektir. Daha fazla güvenlik sağlayabilmek için gizli mesaj, kapakta gizlenmeden önce şifrelenebilir, Şekil 24'de gösterilen seçmeli blok (key), gizli mesaj, kapakta gizlenmeden önce şifrelenme imakanini gösterir.

Gizli mesajı alabilmek için, stego-sistemi kod çözücüsü (decoder), stego-object frekans uzayına dönüştürüldükten sonra, gizli mesaj ayıklanabilecektir. Eğer alıcı orijinal kapağı geri almak istiyorsa, stego nesnesi uzaysal alana geri dönüştürülmelidir.

Şekil 24'de gösterilen seçmeli blok (Frequency to Spatial Domain Transformer), orijinal kapağın geri alınması imkanı gösterir. Genelde, Frekans uzay steganografi üç farklı yol ile uygulanabilir, (DWT): discrete wavelet transform, (DCT): discrete cosine transform, (DFT): discrete Fourier transform. Üçünün her biri, kapağın (nesne) frekans dağılımı hakkında farklı bir bakış açısı verecektir. Frekans uzay steganografi yöntemleri, uzaysal alan yöntemleriyle karşılaştırıldığında daha sağlamdır, ancak uzaysal alan yöntemleri daha fazla kapasite sağlar. Ek olarak, frekans uzay steganografisinin kullanılmasının diğer avantajı, bir nesnenin frekans dağılımının formatından tamamen bağımsız olmasıdır. Frekans uzay steganografi genel modeli aşağıda gösterilmiştir.



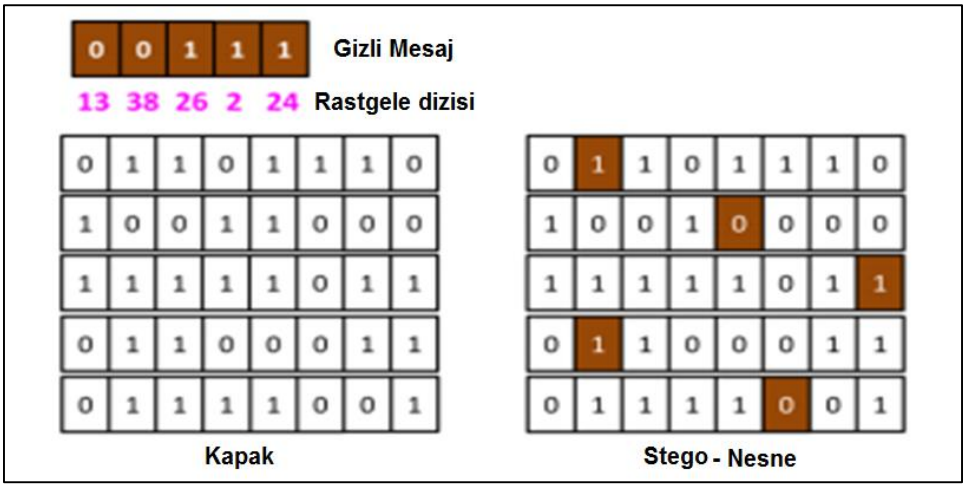
Şekil 24. Frekans domain steganografisinin Temel Modeli

1.4.3.4.3. Yerine Koyma Yöntemleri

Yerine koyma sistemi steganografide uygulanan altı steganografi sınıflandırma yönteminden birisidir. Bu sistem bir kapağın bir parçasını veya daha az önemli kısımlarını gizli bir mesajla değiştirir. Eğer gizli bilginin gömülü olduğu yer biliniyorsa alıcı bilgiyi alabilir. Aşağıda genel olarak bilinen ve steganografi araçlarında uygulanan yerine koyma sistemi altındaki yöntemler sırası ile ele alınmıştır [69].

1.4.3.4.3.1. Sözde Rastlantısal Permütasyon Yöntemi

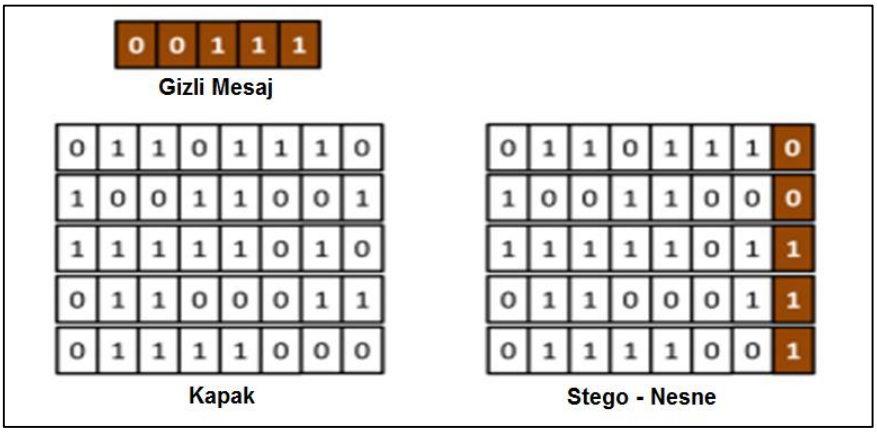
Gizli mesajın dağıtımı tüm kapak elemanları üzerinde rasgele yapılmaktadır. Asıl amaç, bir saldırı karşısında karmaşıklığın arttırılmasıdır. Sonraki mesaj bitlerinin aynı sıraya yerleştirileceğinin bir garantisi yoktur. Bir sıralı sayı üreteci kullanılarak dizi oluşturulur. Gizli mesaj bitleri, oluşturulan dizi tarafından belirlenen kapak elemanlarının bit pozisyonuna göre saklanır [70]. Bu yöntem Şekil 25'te gösterilmiştir.



Şekil 25. Sözde Rastlantısal Permütasyon

1.4.3.4.3.2. En Düşük Değerli Bit Yerine Koyma Yöntemi

Şekil 26'da gösterildiği üzere, bu yöntem, kapak elemanının en düşük değerli bitinin (LSB) bir gizli mesajın bitiyle değiştirilmesiyle gerçekleştirilir [71]. Gizli mesajın yeniden yapılandırılması için, seçilen kapak elemanlarının LSB'si çıkarılır ve sıralanır. Bu yöntem kullanılarak, bilgiler taşıyıcılara çok az etkiyle gizlenebilir. Görüntü ve ses içine uygulama basitliği nedeniyle de, bu yöntem steganografi için yaygın olarak kullanılmaktadır.

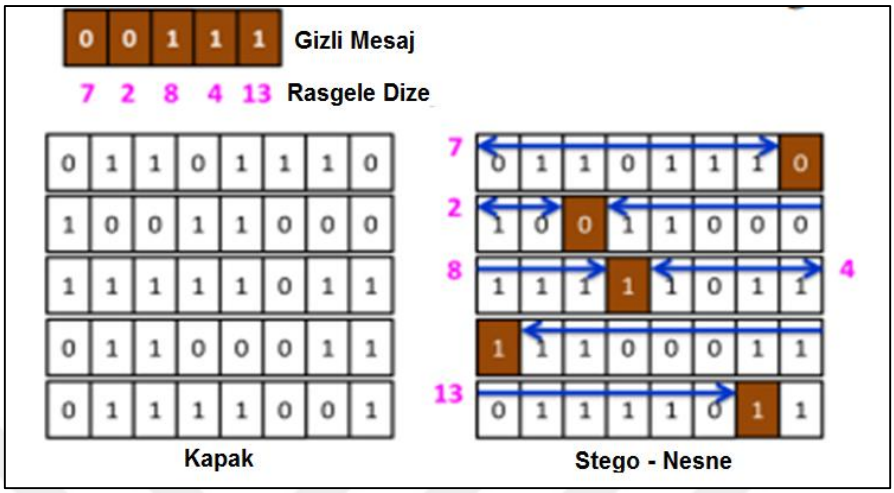


Şekil 26. En Düşük Değerli Bit Değiştirme

1.4.3.4.3.3. Rastgele Aralık Yöntemi

Bu yöntemde, gizli mesajın kapak-elemanları üzerine oldukça rastgele bir şekilde yayılması için bir sözde rasgele sayı üreteci kullanılmıştır [69]. Hem gönderici hem de alıcı bir

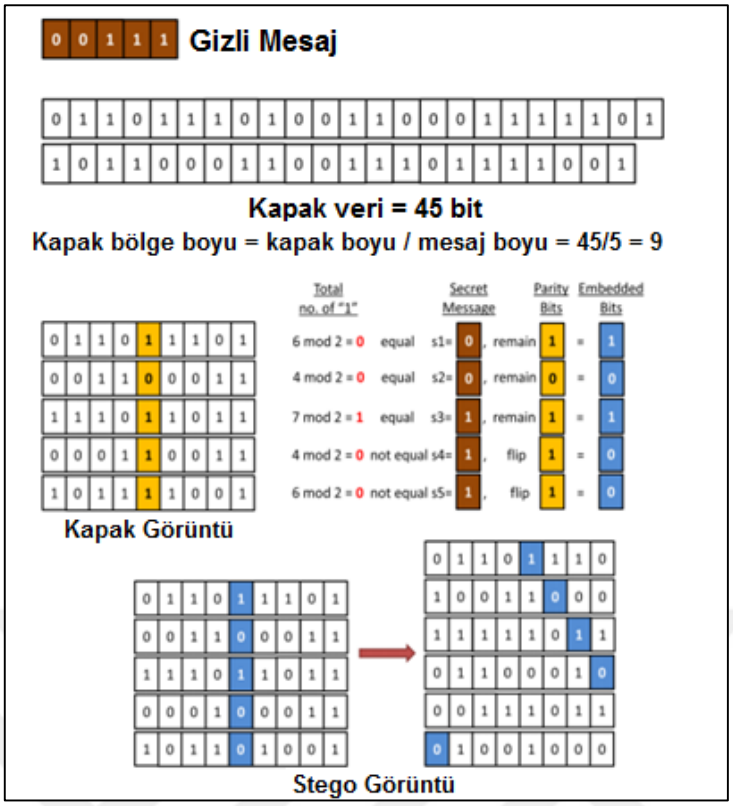
rasgele sayı üreteci için çekirdek olarak bir 'stegokey'i paylaşmaktadır. Daha sonra iki gömülü bit arasındaki aralığın belirlendiği rasgele bir dizi oluşturulmaktadır. Böylece gizli mesaj bitleri, iki gömülü bit arasındaki aralığa göre saklanır. Bu yöntem Şekil 27'de gösterilmiştir.



Şekil 27. Rastgele Aralık Metodu

1.4.3.4.3.4. Kapak-bölgeleri ve Güvenlik (Parity) Bitleri Yöntemi

Ayrık kapak bölgelerinin sözde rastgele dizilimini üretmek için çekirdek olarak bir stegokey kullanılır. Kapak bölgesi uzunluğu, kapak görüntüsü uzunluğunun mesaj uzunluğuna bölünmesiyle hesaplanabilir. Gizli mesajın sadece bir kısmı, tek bir elemandan ziyade bütün bir kapak bölgesinde depolanır. Gömme işleminde, her biri parite biti içinde bir gizli biti kodlayan ayrık kapak bölgeleri seçilir. Bir bölgenin parite biti, toplam sayı 1 olarak sayılarak ve iki değerle modüle edilerek hesaplanabilir. Eğer rastgele seçilen bir kapak elemanının LSB'si, kapak bölgesinin parite biti kodlanacak gizli bit ile uyuşmazsa çevrilir. Seçilen tüm kapak bölgelerinin parite bitleri, alıcıdaki mesajı yeniden oluşturmak için hesaplanır ve sıralanır [72]. Bu işlem Şekil 28'de gösterilmiştir.



Şekil 28. Kapak-bölgeleri ve Güvenlik (Parity) Bitleri

1.4.3.4.4. Kapak Üretim Yöntemleri

Dış dünyadaki muazzam oranda bilgi hacmi nedeniyle, bir kişinin dünyadaki tüm iletişimleri gözlemlemesi imkansızdır. Bunun için çeşitli steganografi teknikleri ile mesajlar güvenli bir şekilde iletilebilir. Gizlenecek mesaj bir kapak data içerisine gizlenerek stego data elde edilir. Burada kapak data bir metin, ses, logo, görüntü veya animasyon olabilir. Bazı durumlarda kapak data elimizde bulunmaz. Gizlenecek mesajın, bir gizleme algoritması yardımıyla belirli bir kapağa gizlendiği sistemlerin aksine, kapak oluşturma teknikleri, mesajın gizleneceği bir kapağın olmadığı durumlarda bu mesajı gizlemek için kapak olması amacıyla dijital bir nesne üretir. Böyle durumlar için steganografide bazı kapak oluşturma teknikleri geliştirilmiştir. Bu tekniklerden en yaygın bilineni mimic fonksiyondur [73].

1991 yılında Peter Wayner tarafından icat edilen Mimic Functions, context free gramerlerin Steganografi alanına uygulanmasıdır. Mimik Fonksiyonları kullanılarak, zararsız formdaki bilgiler gizlenebilir.

Mimik Fonksiyonların diğer steganografik teknikler üzerindeki gücü, gizli mesajın aldığı formun, yalnızca dilbilgisi yapısı oluşturan kişinin hayal gücü ve becerikliliği ile ilintili

olmasıdır. Mimik Fonksiyonların steganalize karşı koyduğu bir başka yol da gizli mesajın zararsız kaynağın istatistiksel özelliklerine sahip olmasıdır [74].

Mimic fonksiyonlar, istatistiksel yapısını, masum gibi görünen bir metin profil ile eşleşecek şekilde değiştirerek mesajın kimliğini gizlemek için kullanılabilir. Mesela ingilizce dili birkaç istatistiksel özelliklere sahiptir. Örneğin, karakterlerin dağılımı eşit değildir, e, z'den çok daha sık görülür. Bu gerçek Huffman kodlaması gibi veri sıkıştırma şemalarında kullanılır. Huffman sıkıştırma işlevlerinden bir mimik işlev elde edilebilir. Bu işlevler sadece makineleri kandırabilir; yani bir insan gözlemci için taklit edilen metin dilbilgisel hatalarından dolayı tamamen anlamsız görünecektir. Bu problemlerin üstesinden gelmek için, içerikten bağımsız gramerlerin uygulanması ile taklitçilik geliştirilmiştir. İçerikten bağımsız gramerler, konuşmanın farklı bölümlerinden dillerdeki cümleleri kurma kurallarını açıklamaktadır. İçerikten bağımsız gramerler, mesajları gizlemek için dilbilgisi açısından doğru bir İngilizce metin oluşturmak için kullanılabilir.

1.4.3.5. Steganografi Çeşitleri

Steganografinin çeşitlerine girmeden önce, genel olarak görüntü, ses ve video steganografisi aynı çalışma prensibine sahip, hepsi de Mekansal ve Dönüştürme alanına bağlı olduğu söylenmelidir.

1.4.3.5.1. Görüntü Steganografi

Görüntü steganografisi, görüntü dosyasını kullanarak mesajı gizleyen bir tür steganografidir. Bu türde görüntülerde gizli mesaj gizlemek için pikseller kullanılır. Görüntü steganografi, steganografinin en popüler şeklidir. Burada, gizli mesaj insan gözü tarafından algılanması neredeyse imkansız olan bir görüntüye gömülür. Görüntü steganografide önemli kavramlar aşağıdaki gibidir:

Kapak-Görüntü: Gizli bilgi için taşıyıcı olarak kullanılan orijinal görüntü.

Mesaj: Gizlemek istediğimiz mesaj. Mesaj düz bir metin veya başka bir görüntü olabilir.

Stego-Görüntü: Kapak görüntüsüne mesaj gömdükten sonra bu stego-görüntü olarak bilinir.

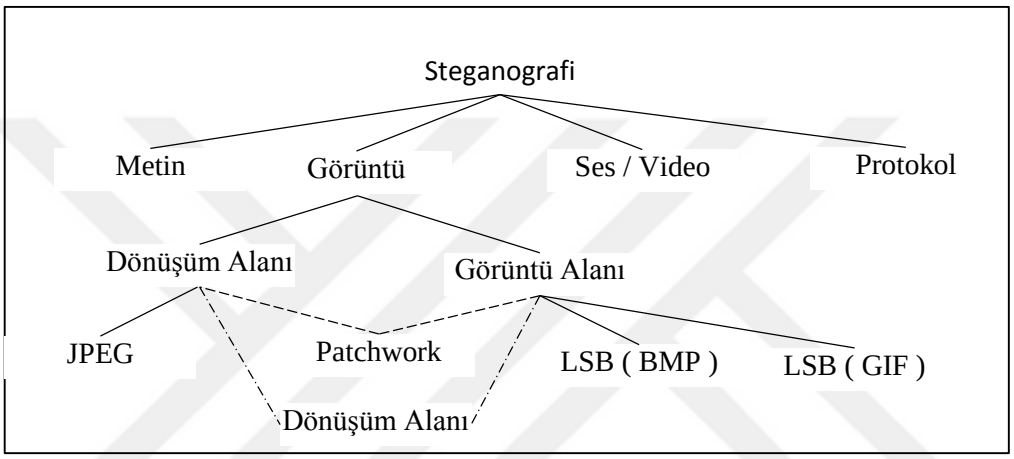
Stego-Anahtar: Mesajların gömülmesi veya çıkarılması için bir anahtar kullanılır (isteğe bağlı).

Görüntü steganografide önce bilgiyi Kapak-Görüntü'ye gizleyin ve bazı algoritmaları kullanarak bir Stego-Görüntü'yü yaratıp, bu Stego-Görüntü daha sonra üçüncü tarafın bu görüntünün gizli bir mesajı olduğunu bilmediği diğer tarafa gönderilir. Alıcı tarafında mesaj, Stego-görüntüden stego-anahtarsız veya stego-anahtar olarak (algoritmaya bağlı olarak) basitçe çıkarılabilir.

Görüntü steganografi teknikleri (Şekil 29) genel olarak iki gruba ayrılabilir:

Görüntü Alanı (uzamsal alan): Piksele bağlıdır ve mesaj doğrudan kapağa gömülür.

Dönüştürülmüş Alanı (frekans bölgesi): Görüntü önce frekans alanına dönüştürülür, ardından mesaj görüntüye gömülür.



Şekil 29. Görüntü steganografi türleri+

1.4.3.5.1.1. Mekansal Alanı:

Mekansal alan adı teknikleri şu şekilde sınıflandırılmıştır:

- Least significant bit (LSB)
- Piksel değeri farklı (PVD)
- Kenarlara dayalı veri yerleştirme yöntemi (EBE)
- Rastgele piksel gömme yöntemi (RPE)
- Histogram kaydırma yöntemleri.

Least significant bit (LSB) bilgiyi görüntüye katmak için basit bir yaklaşımdır. Bir resmin içindeki baytların bir kısmının ya da tamamının en az anlamlı bit'i (8. bit), gizli mesajın bir kısmı dönüştürülür. 24 bit görüntü kullanırken, kırmızı, yeşil ve mavi renk bileşenlerinin her birinin bir kısmı kullanılabilir.

Avantajı: Çok sayıda gizli mesajın bitlerini gizleyebilecek.

İyi seçilmiş bir görüntü ile mesajı en az ve ikinci en az anlamlı bit içinde gizleyebiliriz ve yine de görüntüde bir değişiklik olmaz.

1.4.3.5.1.2. Dönüştürme Alanı (Transform Domain):

Bilgiyi saklamanın daha karmaşık bir yoludur; çünkü öncelikle görüntüyü frekans uzayına aktarır ve daha sonra bilgiyi ona katıştırır. Güçlü steganografik sistemler dönüşüm alanı içinde çalışır. Gizli verinin bir sinyalin frekans uzayına gömülmesi, görüntü alanına gömülmekten çok daha güçlüdür.

Dijital görüntü, yüksek ve düşük frekanslı bileşenlerde bulunan piksellerdir. Kenar pikselleri yüksek frekans bileşenini temsil eder ve yoğun olmayan pikseller düşük frekans bileşenini temsil eder.

Uzamsal alan teknikleri üzerinde avantaj, bilgiye, sıkıştırmaya, kırpmaya veya herhangi bir işlem türüne daha az maruz kalan alanlarda bilgi gizlemedir.

Dönüşüm alan teknikleri şu şekilde sınıflandırılır:

- Ayrık Fourier Dönüşümü (DFT).
- Ayrık Kosinüs Dönüşümü (DCT).
- Ayrık Dalgacık Dönüşümü (DWT).

1.4.3.5.2. Ses Steganografisi

Kapak olarak ses dosyasını kullanarak bilgi gizleme sürecidir. Ses steganografisi, steganografi için WAVE, AVI MPEG vb. gibi dijital ses formatlarını kullanır. Ses steganografi yöntemleri aşağıdaki gibidir:

LSB Kodlama: Gizli mesaj baytlarının bir dizisini gizlemek için ses dosyasının bazı bitlerindeki en az anlamlı bitin yerini alan çok popüler bir yöntemdir (resimdeki gibi).

Avantajı: Bu yöntemde, kapak baytları gizli bit ile değiştirilirken, aynı bitin olması önemli bir olasılıktır, bu nedenle bit'i değiştirmeye gerek yoktur, bu nedenle bu yöntem önemli kalite bozukluklarına neden olmaz.

Eko Gizleme (Echo Hiding): *Avantajları:* Yüksek veri iletim hızı ve çok iyi sağlamlık sağlar. Kodlama işleminden önce, orijinal işaret bloklarına ayrılır. Kodlama işlemi tamamlandıktan sonra, stego-audio (son sinyal) oluşturmak için bloklar birleştirilir.

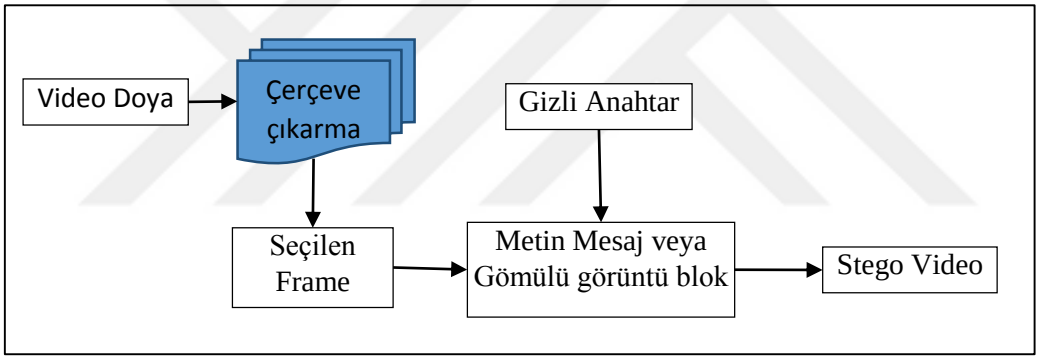
Yayılı Spektrum (Spread Spektrum) : Bu yöntem ses sinyalinin frekans spektrumu boyunca gizli bilgileri yaymaya çalışır.

1.4.3.5.3. Video Steganografi

Bu tip steganografide kapak ortamı bir video dosyasıdır (Şekil 30). Videonun büyük avantajları, içeride saklanabilecek büyük miktarda veri ve sürekli bir bilgi akışı nedeniyle görüntülerin ve seslerin devam eden bir akış olması ve izleyici tarafından gözlemlenmeyen küçük değişikliklerdir. Video steganografi için bazı önemli yöntemler aşağıda verilmiştir:

Mekansal Alan Tabanlı Yöntem: Bilgileri video karelerinin piksellerinde gizler, bu yöntem daha yüksek yerleştirme kapasitesine ve uygulama kolaylığına sahiptir. LSB yöntemi, rastgele piksel seçimi ile gizli mesajı en az anlamlı bitini gömer.

Dezavantajları: Görüntü döndürme, kırpma ve ölçekleme gibi saldırılara karşı yüksek derecede savunmasızdır.



Şekil 30. Video Steganografi

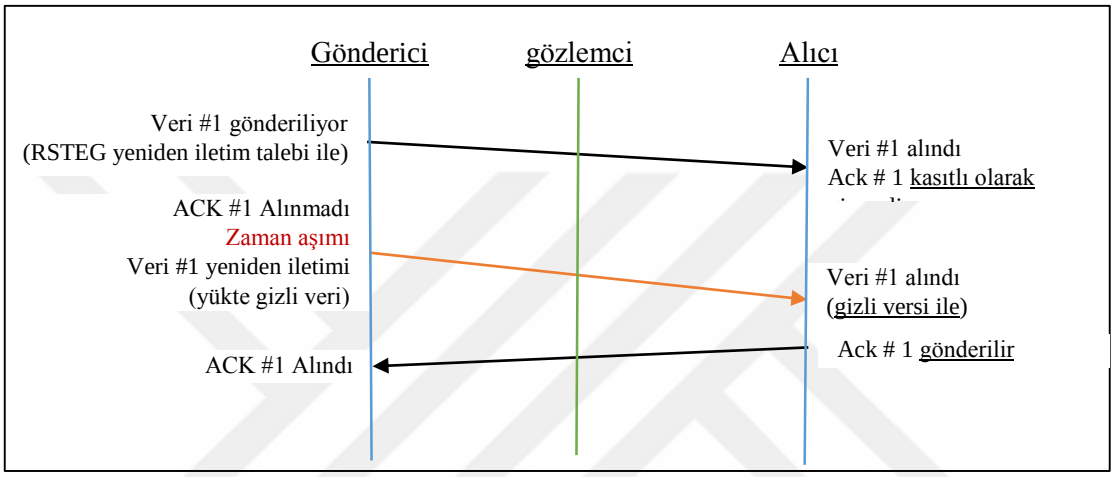
Dönüştürme Alan Tabanlı Yöntem: Saldırılara daha az eğilimli yüksek güvenlik. Ayrık Kosinüs Dönüşümü (DCT) ve Ayrık Dalgacık Dönüşümü (DWT), video steganografisindeki en popüler dönüşümdür.

1.4.3.5.4. Ağ Protokolü Steganografi

Bu türde, gizli mesajı gizlemek için kullanılan protokoller TCP, UDP, IP vb. dir. Ağ steganografisinde, bant genişliği oldukça esnektir ve protokol türüne ve ağ kapasitesine bağlıdır. Bant genişliğinin kapak büyüklüğü ile sınırlı olduğu diğer steganografi türlerinden

farklı olur. Ağ steganografisinin temel özellikleri bant genişliği, tespit edilemezlik ve sağlamlığıdır. Ağ protokolü yaklaşımları aşağıdaki türlerden olabilir:

RSTEG (Yeniden İletim Steganografisi): Bu yöntem, alıcıdan alındı bildiriminin alınmasına bağlıdır. Onay, gönderilen mesajın kendisine ulaştığını onaylamak için alıcı tarafından gönderilen bir mesajdır. Şekil 31 RSTEG tekniğinin nasıl çalıştığını göstermektedir. Gönderen alıcıya 1 numaralı veriyi gönderiyor ve alıcı kasıtlı olarak verinin alınmadığını gösteriyor; bu gönderen nedeniyle 1 numaralı veriyi almanın onayını almaz, gizli verileri ekleyerek verileri yeniden iletir ve alıcı için alındı bilgisi gönderilir.



Şekil 31. Timeout alıcının onayını alma zamanının gelmesi

Şekil 31'de, Timeout alıcının onayını alma zamanının geldiğini gösterir. Bu yöntemin avantajı yöntemi tespit etmesinin çok zor olmasıdır.

TransSteg: Veri sıkıştırma işleminden kalan alana bağlıdır.

StegSuggest: Bu yöntem, gizli iletilerin Google önerilerinde gizlenmesine dayanmaktadır.

HICCUPS (Bozuk Ağlar için Gizli İletişim Sistemi).

1.4.3.5.5. Metin Steganografi

Metin steganografi, metni başka bir metin dosyasının içerisine gizler. Metin dosyalarında gizli mesajı saklamak için kullanılan en güvenilir steganografi türlerinden birisidir. Metin steganografi, verileri gizlemek için bir araçtır. Bu en zorlu steganografidir. Bunun nedeni, bir görüntü veya ses dosyasıyla farklılaştırılmış bir metin dosyasında tekrarlanan verilerin göreceli olarak daha az olmasıdır. Metin belgelerinin yapısı, gözlemlediklerimizle ayırt edilemezken,

farklı türde belgelere, örneğin resimlerde dokümanın yapısı, gözlemlediklerimize göre benzersizdir. Bu nedenle, bu belgelerde, istenen çıktıda kayda değer bir değişiklik oluşturmadan belgenin yapısındaki değişiklikleri uygulayarak verileri gizleyebiliriz. Metin steganografisi, mesajı başka bir metin mesajı kullanarak gizleyen bir steganografi türüdür, metin dosyasının gizlenmesi için gizli mesajı gizleyeceği bir kapak ortamı olarak adlandırılan metin mesajıdır. Mesaj, varolan bir metnin biçimlendirmesini ve bir metindeki sözcüklerin değiştirmesini, rasgele karakter dizileri veya okunabilir metinler oluşturmak için dilbilgileri kullanılarak birçok şekilde gizlenebilir. Metin steganografisinde avantaj, bellek gereksinimlerinin iletişim için daha küçük ve hızlı olmasıdır.

Metin steganografi genel olarak üç kategoride sınıflandırılmaktadır. Bunlar, Dilbilimsel, Format tabanlı, Rasgele ve İstatistiksel nesil yöntemlerdir [75]. Aşağıda metin steganografi kategorileri verilmiştir.

1.4.3.5.5.1. Dil Yöntemleri

Dilbilgisel steganografi, özellikle üretilen ve düzenlenen metnin semantik özelliklerini dikkate alır. Çoğu zaman mesajların gizlendiği alan olarak etimolojik yapıyı kullanır. CFG, sol dalın "0" ve sağ dalın '1' ile ilişkili olduğu kısımları gizlemek için kullanılabilecek ağaç yapısını oluşturur. Bu tekniğin kullanılmasının bir kaç dezavantajı vardır. Küçük bir cümle yapısı üzerinde yapılan değişimler söz kalabalığına neden olmaktadır. Metin dilbilgisel olarak kusursuz olmasına rağmen, anlamsal bir bozukluk oluşur. Sonuçta birbiriyle ilişkisi olmayan bir dizi cümleler meydana gelir. Bu yöntem aşağıdaki kategorilere bölünebilir:

1.4.3.5.5.1.1. Sözdizimsel (Syntactic) Yöntem:

Bu teknikte 0 ve 1 bitlerini gizlemek için noktalama işaretleri, örneğin, nokta (.) veya virgül (,) kullanılır [76]. Bu teknikteki sorun, doğru yerlerde noktalama işaretlerinin tanımlanmasının gerekmesidir.

1.4.3.5.5.1.2. Semantik Yöntem:

Bu yöntem, bilgileri gizlemek için belirli kelimelerin eşanlamlılığını (başka bir kelime) kullanır. Avantaj: Yeniden tanımlarken veya karakter tanıma programlarını kullanırken gizli bilgileri korur.

1.4.3.5.5.1.3. Metin Kısaltması (Text Appreviation):

0 bitini gizlemek için, kelimenin tam şekli kullanılır ve 1 bitini gizlemek için kısaltma kullanılır.

1.4.3.5.5.1.4. Yazımın Değiştirilmesi:

Gizli bilgileri saklamak için İngiliz ve Amerikan İngilizcesinde yazılan kelimeler kullanılır.

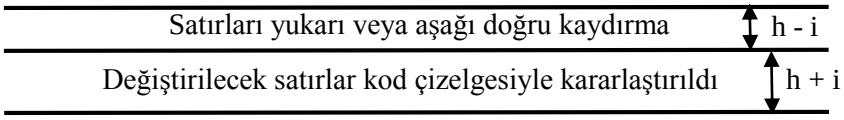
1.4.3.5.5.2. Biçim Tabanlı Yöntemler

Biçime Dayalı Yöntemlerde verileri gizlemek için metnin fiziksel yapısının düzenlenmesi gerekir [77]. Bu tekniğin sakıncası, eğer stego dosyası bir kelime işlemcisi ile açılırsa, yanlış yazılmış kısımlar ve ek boşluklar ayırt edilir. Metin stili boyutlarının değişmesi, stego analist için şüphe uyandırabilir. Dahası, eğer ilk düz metine ulaşılırsa, bu düz metnin ve şüpheli steganografik metnin zıt olması metnin gizlenmiş kısımlarını tamamen görünür hale getirecektir.

Biçime dayalı yöntemler, mesajı gizlemek için metnin biçimini değiştirmeyi içerir. Bu yöntemin bazı kusurları vardır. Stego dosyası bir kelime işlemcisi ile açılırsa, yanlış yazımlar ve fazladan beyaz boşluklar algılanır. Değişen yazı tipi boyutları şüpheyi bir insan okuyucusuna yönlendirebilir ve hızlı bir şekilde tespit edilebilir. Ayrıca, orijinal düz metin mevcutsa düz metin ile steganografik metin karşılaştırılarak metnin manipüle edilen kısımları oldukça görünür hale getirilecektir. Bu yöntemin avantajları bilgiyi saklamak için yüksek kapasiteye ihtiyaç duyulmasıdır. Bazı format tabanlı yöntemler aşağıda verilmiştir.

1.4.3.5.5.2.1. Satır Kayma (Line Shift)

Bu yöntemde, gizli mesaj metin satırlarının bir dereceye kadar yukarı veya aşağı kaydırılmasıyla gizlenir (örneğin, her bir satır n inç olarak değiştirilir). Bit 0'ı gizlemek için bir satır yukarı kaydırılır ve bit 1'i gizlemek için satır aşağı kaydırılır. Metin yeniden yazılırsa veya bir karakter tanıma programı kullanılıyorsa gizli bilgiler yok olur (Şekil 32).



Şekil 32. Satır kayma yöntemi

Bu teknik, metin satırlarının konumunu dikey olarak taşıyarak bir belgeyi değiştirir [78]. Belli bir belge için yeniden atanan kod sözcüğü, bu belgede taşınacak metin satırlarını belirler. Bir satırın yukarı taşınması için "0" ve bir satırın aşağı taşınması için "1" değeri kullanılabilir. Bu strateji, performans ölçümü için diferansiyel kodlama yöntemini kullanır. Bu tekniğin kusuru, metnin yeniden yazılması veya bir OCR programı kullanılması durumunda, gizli verilerin yok edilmesidir.

1.4.3.5.5.2.2. Kelime Kayma (Word Shift)

Bu teknikte, kelimeleri yatay olarak hareket ettirerek ve mesafeyi değiştirerek, gizli mesaj metinde gizlenir. Bu teknik, kelimeler arasındaki mesafenin değiştiği metinler için önemlidir. Kelimeler kayması, korelasyon teknikleri kullanılarak tanımlanabilir. Bu teknik daha az fark edilebilir, çünkü kelimeler arasındaki mesafenin değişmesi çok normaldir.

Dezavantajı: Bu yöntem çok zaman alıcıdır. Metni yeniden yazmak veya karakter tanıma programını kullanmak gizli bilgileri yok edecektir.

1.4.3.5.5.2.3. White Steg:

Bu prosedür gizli bir mesajı gizlemek için beyaz alanları (boşluk) kullanır [79]. Beyaz boşlukları kullanarak bilgi gizlemek için üç teknik vardır. Bunlar, cümleler arası boşluk, satır sonu boşluğu ve kelimeler arası boşluktur. Cümleler arası boşlukta, bir bit 0'ı gizlemek için, tek bir boşluk koyarız ve bit 1'i gizlemek için her bir bitiş karakterinin sonuna iki boşluk koyarız. Satır sonu boşlukta, her satırın sonuna doğru yerleştirilmiş boşluklar mevcuttur. Örneğin, satır başına bir bit kodlamak için iki boşluk, iki biti kodlamak için dört boşluk vb. Kelimeler arası boşluk sisteminde, bit 0, bir sözcükten sonra bir boşluk ile temsil edilir ve bit 1, bir sözcükten sonra iki boşluk ile temsil edilir. Bu teknik, gizli bir mesajı gizlemek için beyaz boşlukları kullanır. Gizlemek için üç yöntem vardır;

Inter Sentence Spacing tekniği: Bit 0'ı ve her sonlandırma karakterinin sonunda bit 1'i gizlemek için ve iki alanı gizlemek için tek boşluk.

End of Line Spaces tekniği: Her satırın sonunda sabit sayıda boşluk eklenir.

Inter Word Spacing tekniği: Bir kelime bittikten sonra bir boşluk, bit 0'ı ve iki kelimeden sonraki iki boşluk, bit 1'i temsil eder.

1.4.3.5.5.2.4. Paragraflarda Verileri Gizleme Tekniği

Bu teknik önceden belirlenmiş bir örtülü dosyayı kullanır [80]. Bir örtülü dosyanın kelimelerinin başlangıç ve bitiş harflerini kullanarak bir mesajı gizleyerek çalışır. Şifre metnini bit akışına dönüştürdükten sonra, her bit, örtülü dosyadan bir sözcük seçerek ve gizlenecek bit sayısına bağlı olarak o kelimenin başlangıç veya son harfini kullanarak gizlenir. Bit 0 veya 1, sırasıyla, örtülü dosyadan ve sırasıyla stego anahtarındaki sözcüğün başlangıç harfini veya bitiş harfini içeren bir kelimeyi okuyarak gizlenir. Kapak (örtülü metinde) herhangi bir değişiklik yapılmadığı için örtülü dosya ve ilgili stego dosyası tam olarak aynıdır.

1.4.3.5.5.2.5. MS Excel'de Metin Dönüştürme Teknikleri

Bu teknikte ilk önce gizli mesajı ikili biçime dönüştürürüz ve sonra her bir biti ikili formattan sırayla seçeriz [81]. Daha sonra bitin 1 veya 0 olup olmadığını kontrol ederiz. Eğer bit 1 ise, MS Excel dosyasının bir hücresini 1 derece döndürerek başka bir dereceye kadar 0 dereceyi koruruz. Bu işlem tüm bitler gizlenene kadar tekrarlanır. Hücrenin seçilmesi ayrıca sırayla yapılır. Gizleme işlemi tamamlandıktan sonra stego dosyası oluşturulur ve alıcıya gönderilir. Alıcı daha sonra sıralı olarak stego dosyasındaki hücre dönüşünü kontrol eder [82]. Hücre rotasyonu 1 dereceyse, gizli bit 1'dir, yoksa gizli bit 0'dır. Bu 1 ve 0 başka bir dosyaya konur ve daha sonra orijinal gizli mesajı geri almak için karakter biçimine dönüştürülür.

1.4.3.5.5.2.6. HTML Tags

HTML Etiketleri türü mesajı gizlemek için kullanılabilir. Örneğin,

Stego anahtarı:	Sego verileri:
<img> </ img> 0'a bakın	<img src = g1.jpg> </ img>
<img /> 1 bakın	
	<img src = g3.jpg> </ img>
	Gizli bitler: 010

1.4.3.5.5.3. Rastgele ve İstatistiksel Üretim

Bilinen bir plaintext ile karşılaştırma problemini gidermek için steganograflar sıklıkla kendi özel örtülü metinlerini üretirler. Kullanılan teknikler içerisinde karakterlerin düzensiz görünümlü bir grupta saklanması ve kelime uzunluğu ve harf frekanslarının sayısal özellikleri, gerçek kelimelerle aynı sayısal özelliklere sahip gibi görünen kelimeler seçilerek kullanılmaktadır.

1.4.3.6. Steganografi Saldırıları

Steganografide iki genel saldırı türü vardır.

1.4.3.6.1. Pasif Saldırılar

Saldırgan, gönderen ve alıcı arasındaki iletişimi izleyebilir. İletişim sırasında herhangi bir gizli mesajı tespit etmeye çalışır. Gizli mesajın varlığını tespit edebilecek olursa, bunu pasif saldırı olarak adlandırabiliriz. Bu saldırı gizli bir mesajın var olduğuna dair bazı istatistiksel kanıtlar sağlayan pratik steganografi araçlarıyla yapılabilir veya matematiksel bir steganografi modelini tanımlamaya çalışarak yapılabilir. Güvenlik ve belli bir şemanın bu güvenlik özelliğini sağladığını ispatlayabiliriz.

1.4.3.6.2. Aktif Saldırılar

Saldırgan, iletişim sırasında kapağı değiştirebilir. Böylece, gizli mesajı değiştirmek veya yok etmek için kapakta bazı değişiklikler yapabilir. Ya da mesajı tespit etmeye çalışır. Bunu yapmak için görüntü tipini değiştirebilir, biraz gürültü ekleyebilir veya bazı sinyal işleme işlemlerini uygulayabilir. Steganografi sistemlerinin çoğu aktif saldırganlar düşünülerek tasarlanmamıştır.

1.5. Matematiksel İfadeler

Matematiksel bir ifade, sayılar, sabitler, değişkenler, işlemler, işlevler ve diğer sözdizimsel öğeler gibi matematiksel sembollerin kümesidir. Her matematiksel ifade, bir

operatörün sembolü olarak takip edilen bir işlenenin dizisi olarak görülebilir. Bilgisayar cebir sistemlerinde, ifadelerin bu temsili çok esnektir. Bir denklem, bir "=" sembolüyle ifade edilir. Bir matris, bir operatör "matrisi" ile bir ifadesi ve onun işlenenleri de satırları olarak temsil edilir. Bilgisayar programları, en az iki işleneni, parametrelerin listesini ve gövdeyi içeren "yordam" gibi işleçlerle ifadeler olarak da gösterilebilir. Bu şekilde, matematiksel bir ifadenin kendisi bir program olarak görülebilir. Örneğin, "a + b" ifadesi, a ve b parametrelerinin toplamını gerçekleştiren bir programdır.

1.5.1. Matematiksel İfadelerin Türleri

Çeşitli matematiksel ifadeler vardır. Matematiksel bir ifadedeki her öğe, bu ifadenin türünü değiştirebilir. Tüm bu unsurların bir listesi [83] Tablo 1'de sunulmuştur. Bu tezde, tüm kapalı olmayan formlar nedeniyle sabit, değişken, basit aritmetik işlem, faktöriyel, tamsayı üssü, root kök ve rasyonel üslerden oluşan cebirsel ifadelere odaklanacağız.

Tablo 1. Matematiksel ifade türleri

	Arithmetic expressions	Polynomial expression	Algebraic expressions	Closed-form expressions	Analytical expressions	Mathematical expressions
Constant	Y	Y	Y	Y	Y	Y
Variable	Y	Y	Y	Y	Y	Y
Elementary arithmetic operation	Y	Y	Y	Y	Y	Y
Factorial	Y	Y	Y	Y	Y	Y
Integer exponent	N	Y	Y	Y	Y	Y
Nth root	N	N	Y	Y	Y	Y
Rational exponent	N	N	Y	Y	Y	Y
Irrational exponent	N	N	N	Y	Y	Y
Logarithm	N	N	N	Y	Y	Y
Trigonometric function	N	N	N	Y	Y	Y
Inverse trigonometric function	N	N	N	Y	Y	Y
Hyperbolic function	N	N	N	Y	Y	Y
Inverse hyperbolic function	N	N	N	Y	Y	Y
Gamma function	N	N	N	N	Y	Y
Bessel function	N	N	N	N	Y	Y
Special function	N	N	N	N	Y	Y
Continued fraction	N	N	N	N	Y	Y
Infinite series	N	N	N	N	Y	Y
Formal power series	N	N	N	N	N	Y
Differential	N	N	N	N	N	Y
Limit	N	N	N	N	N	Y
Integral	N	N	N	N	N	Y

1.5.2. Matematiksel İfadelerin Gösterimi

Matematiksel ifadeler String, List ve Tree gibi çeşitli veri türlerinde temsil edilebilir. Bu bölüm ifadelerin temsil biçimlerinin kısa bir özetini sunmaktadır.

1.5.2.1. İfade Yapısı

Bu tezde cebirsel ifadelere odaklandığımızda, önce hiyerarşik doğası ve özyinelemeli formları olan cebirsel ifadelerin yapısını tartışalım.

Cebirsel bir ifade *u* bu kurallardan birine girmelidir:

• u bir sayıdır

• u bir sembol

• İşlenenler başka bir cebirsel ifade olan bir toplam, ürün veya işlevdir.

Son kural, tekrarlayan bir cebirsel ifadeden çok tanımını yapar, çünkü her bir işleneninin de cebirsel bir ifade olmasını gerektirir. Cebirsel ifadeler, ön cebirin dönüşüm kuralları ile manipüle edilebilir. Bir cebirsel ifadenin doğru şeklini göstermek için bazı örnekler:

1, 1 + 2/3, cos (x2), g (f (x, y)), doğru

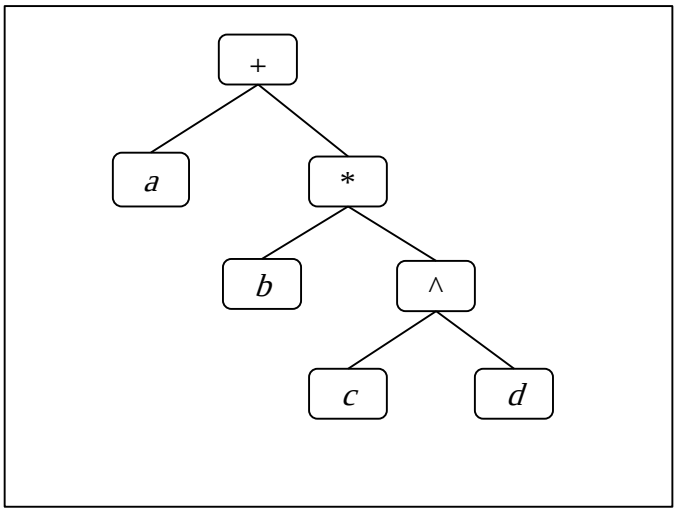
Bu nedenle, aşağıdaki örnekler yukarıdaki kurallara uymaz:

[x, y, z], x = 1, doğru veya yanlış

1.5.2.2. İfade Ağacı (Expression Tree)

Bir ifadenin operatörleri ve işlenenleri ile yapısı arasındaki ilişki, ifade ağaçları kullanılarak grafik olarak gösterilebilir.

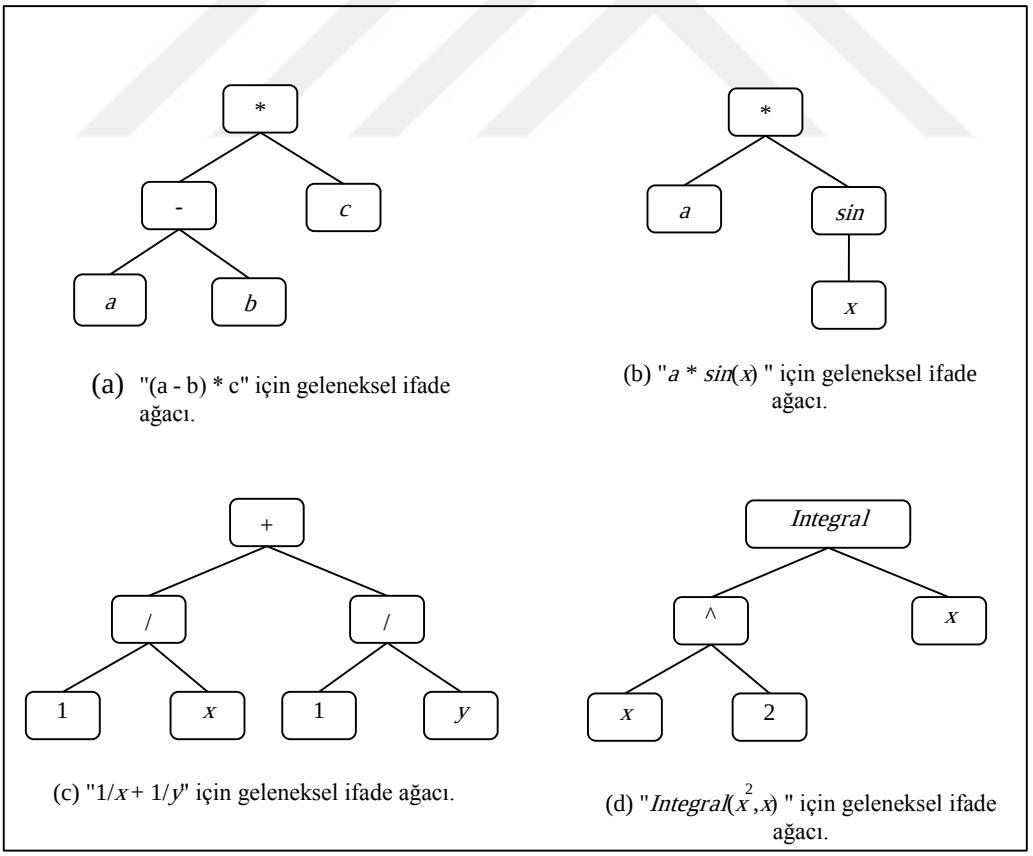
Şekil 33, "a + b * c ^ d" ifadesinin ağacını gösterir.



Şekil 33. "a + b * c ^ d" ifadesinin geleneksel ifadesi

Her operatör bir düğüm tarafından temsil edilir. En düşük önceliğe sahip operatörler ağacın üstünde görünürken, daha yüksek önceliğe sahip operatörler ağacın dibine konur. Her seviye, bir operatör ile onun işleneni veya işlenenleri arasındaki bağlantıyı gösterir.

Şekil 34'de, farklı işlenenlerle farklı ifadeler için daha fazla ağaç gösterilmiştir.



Şekil 34. Bazı geleneksel ifade ağaçları

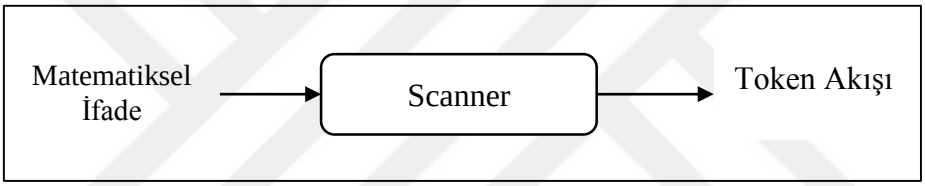
1.5.2.3. Soyut Sözdizim Ağacı (AST)

Matematiksel ifadelerin dize formu, bir sözcük analizcisi tarafından matematiksel tokenlere dönüştürülmelidir. Giriş dizesini ve belirteci akışını göstermek için Tablo 2'de bir örnek gösterilmiştir.

Tablo 2. Giriş dizesi ve token akışı

Giriş dizesi	Token akışı
$3x^2+2(x-12)$	`Num(3) Var Power Num(2) Plus Num(2) LPR Var Minus Num(12) RPR`

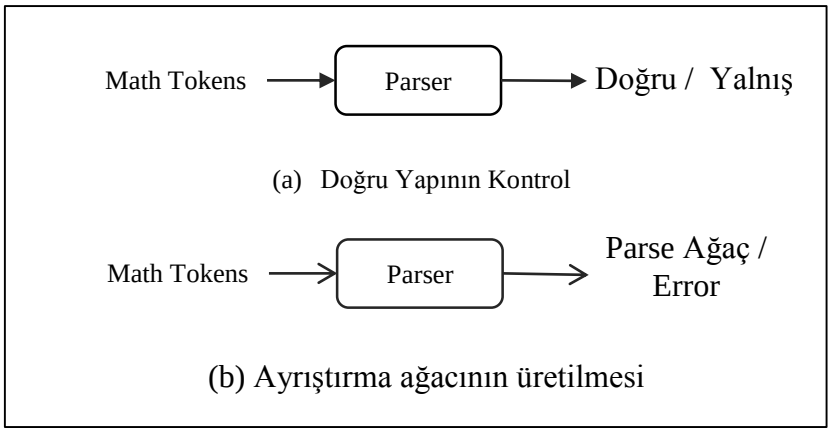
Tablo 2'de, belirteç akışını tarif etmek için kullanılan terimler sunum amaçlıdır. Giriş dizgesini tokenlere dönüştürme işlemi Şekil 35'te gösterilmektedir. Bundan sonra, sözcük analizcisi için sözcük terimini kullanacağız.



Şekil 35. Giriş dizgesini belirteçlere dönüştürme

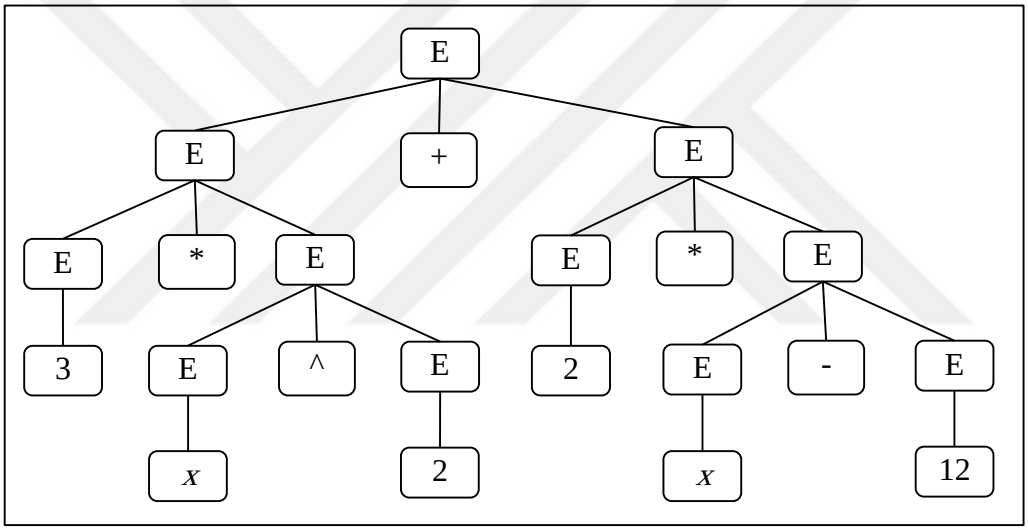
Token akışı, örneğin parantezlerin çift halinde olup olmadığını ve her operatörün yeterince işlenip işlenmediğini kontrol eden matematiksel kurallarla ele alınmalıdır. Bu kurallar, bir sonraki bölümde ele alınacak matematiksel bir dilbilgisi ile uygulanacaktır.

Token akışı, matematiksel bir dilbilgisi tarafından tanımlanan tüm kuralları karşılayıp karşılamadığı analiz edilir. Gerekirse, sözdizimi analizcisi, giriş akımının doğru yapısı için doğru veya yanlış bildirebilir veya Şekil 36'da gösterilen sonraki aşamalara bir girdi olarak bir ayrıştırma ağacı oluşturabilir.



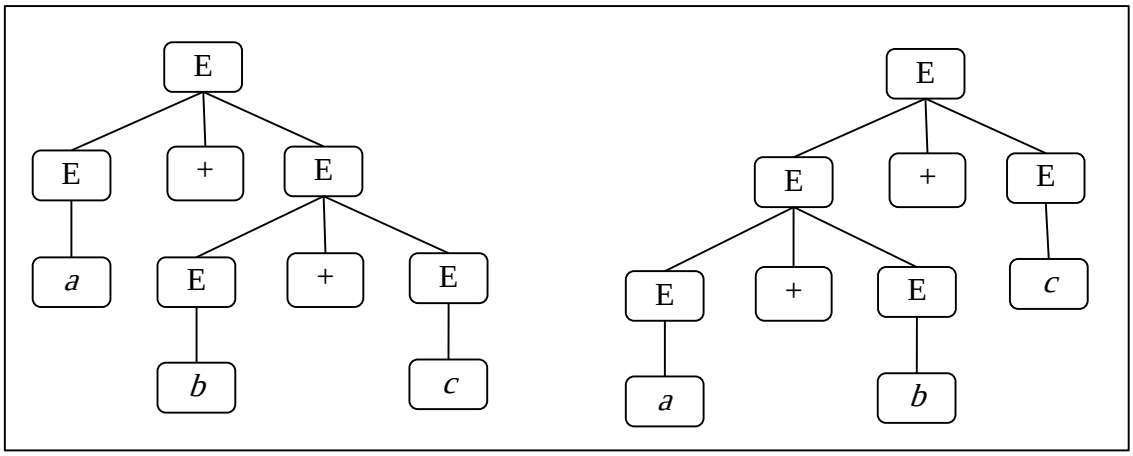
Şekil 36. Yapısal analiz için iki amaç

Ayrıştırma ağacı, ayrıştırıcı olarak adlandırılan bir birim tarafından oluşturulur. Yanlış token akışı için bir hata bildirilecektir. Şekil 37, Tablo 2'nin ayrışmasını göstermektedir.



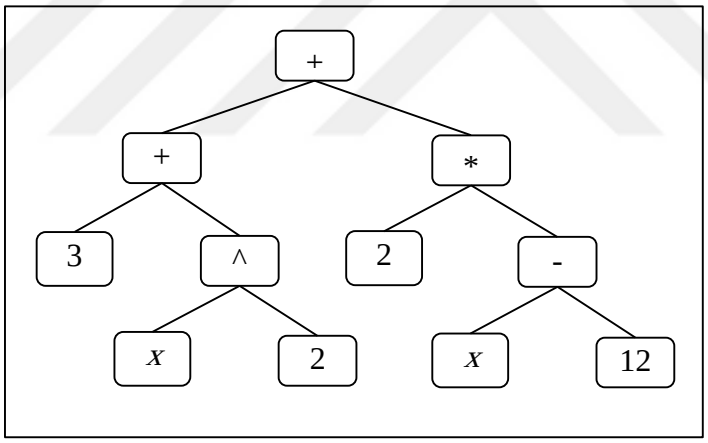
Şekil 37. Tablo 2'nin ayrışması ağacı

Dilbilgisine bağlı olarak, farklı ayrışma ağaçları olabilir. Şekil 38, "a + b + c" için iki ayrıştırma ağacını göstermektedir.



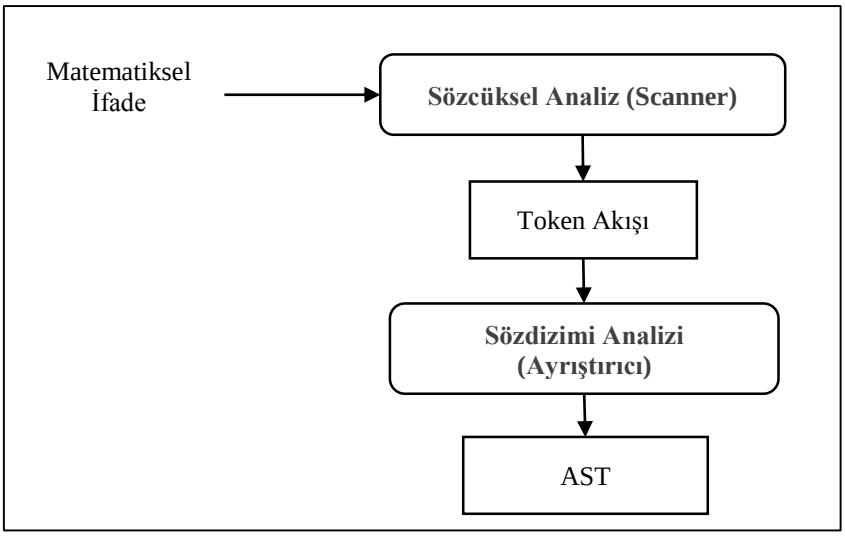
Şekil 38. "a + b + c" için iki farklı ayrıştırma ağaçları

Ayrıştırma ağacındaki yapraklar terminallerdir ve diğer düğümler non-terminallerdir. Bununla birlikte, sözdizimi ağacından başka bir ağaç türünü kullanmak uygundur. Temel fark, yaprakların işlenen ve düğümlerin operatörler olmasıdır. Ayrıca matematiksel ifadelerle çalışmak için basit ve verimli bir ağaç sunar. Şekil 39, Şekil 37'in sözdizimi ağacını göstermektedir.



Şekil 39. Şekil 38'in sözdizimi ağacı

Sözdizimi ağacına ayrıca Soyut Sözdizimi Ağacı veya kısaca AST denir. Şekil 40, bir giriş dizisinden matematiksel bir ifade olarak AST'nin üretilmesi için adımları göstermektedir.



Şekil 40. AST oluşturmak için adımlar

Matematiksel ifadelerde tutarsızlık oluşmaz. Bu nedenle semantik bir analiz bileşeni kullanmayacağız. Bununla birlikte, sadeleştirme için bölünme sırasına göre kontrol etmeliyiz. Bu tezde, AST'yi üretiyoruz ve bir derleyicinin kalan aşamalarını dikkate almıyoruz. Bir sonraki bölümde gramer ve ayrıştırma konuları ele alınmaktadır.

1.5.3. Diller İçin Dilbilgisi (Grammar)

Belirli bir dilin Backus Normal Formu (BNF) veya Meta Dili, sözdizimi kuralları veya bir dilin dilbilgisi ile belirlenir. C dili, dilbilgisini tanımlamak için BNF'yi kullanan ilk programlama dilidir. Bu tezde matematiksel kurallara odaklanacağız ve onlarla çalışmak için bazı gramerleri tanıtacağız. Tablo 3 bir matematik dilbilgisinin önemsiz bir örneğini göstermektedir. Tablo 3'te matematiksel ifadelerin dilbilgisi yapısı için bir örnek verilmiştir.

Tablo 3. Matematiksel ifadelerin dilbilgisi yapısı için bir örnek

exp → *exp op exp* | *func*(*exp*) | (*exp*) | *Num* | *Id*
op → + | - | * | / | ^
func → *Sin* | *Cos* | *Tan* | *Cot*
Id → *x* | *y* | *z* | *a* | *b* | *c*
Num → [*Digit*]* [.[*Digit*]*]?
Digit → 0 | 1 | 2 | 3 | .. | 9

Her kuralın sol tarafında terminal olmayan bir kısmı ve sağda bir dizi kural vardır. Dil oluşturma sırasında, yukarıdan aşağıya veya aşağıdan yukarıya ayrıştırma yöntemine bağlı olarak, bir kural genişletilecek veya daraltılacaktır. Bu iki yöntemi daha sonraki bölümlerde tartışacağız. Tablo 3'teki kuralları veya üretimleri kullanarak, aşağıdaki ifadeler elde edilebilir:

- a + b - c
- 12 * (*Sin* (*x* - 2) +6)
- *x* + *y* ^ (6 - 2)

İşte "a + b - c" nin en soldaki türevi:

$$exp \rightarrow exp + exp \rightarrow exp + exp - exp \rightarrow a + exp - exp$$
$$\rightarrow a + b - exp \rightarrow a + b - c$$

Bununla birlikte, sözdizimsel olarak doğru olduğundan, "a / (6 - 6)" gibi anlamsal olarak yanlış türetmeler üretmek mümkündür. Dilbilgisi, sözdizimi değil, sözdizimi için bir araçtır. Matematiksel ifadelerle onay anlambilimsel konu olan Fortunatey, ifadenin değerlendirilmesi sırasında kontrol edilebilen sıfıra bölünür.

1.5.3.1. Dilbilgisi Hiyerarşisi

Noam Chomsky'nin tanıttığı Chomsky Hiyerarşisinde dört çeşit resmi dilbilgisi vardır. Bu bölümde, bu dilbilgilerini kısaca gözden geçireceğiz.

- Tip 0 - Serbest veya kısıtlanmamış dilbilgisi (unrestricted grammars): Bu tür en genel biçimdir. Üretimlerin biçimi u → v, u ve v'nin rasgele bir sembol kümesi olduğu durumda. Ancak, null olamaz. u ve v'de nelerin göründüğü konusunda herhangi bir kısıtlama yoktur.
- Tip 1 - içeriğe duyarlı dilbilgileri (context-sensitive grammars): Üretimlerin biçimi uXw → uvw, burada u, v ve w'nin, v'de rastgele bir sembol kümesi olduğu durumda, v boş olamaz ve X, tek bir non-terminaldir. Başka bir deyişle, X, yalnızca çevrelenmiş olduğunda v ile değiştirilebilir.
- Tip 2 – İçerikten bağımsız dilbilgileri (CFG): Üretimlerin biçimi, X'in v'de rastgele bir sembol kümesi olduğu ve X'in tek bir non-terminal olduğu X → v şeklindedir.

- Tip 3 - düzenli dilbilgisi (regular grammars): Üretimlerin biçimi X → a veya X → Y'dir, burada X ve Y tek non-terminaldir ve a tek bir uçbirimdir. Bu dilbilgisi ifade gücü bakımından en sınırlı dilbilgisidir.

Tip 3, tip 2'nin bir alt kümesidir. Ayrıca tip 2, tip 1 ve benzeridir. Özyinelemeli yapıların eksikliği nedeniyle, tip 3 dilbilgisinin ayrıştırılması özellikle kolaydır. Tip 2 dilbilgisinin birçok sınıfı için verimli ayrıştırıcılar vardır. Tip 1 ve Tip 0 dilbilgileri Tip 2 ve 3'ten daha güçlü olmasına rağmen, onlar için uygun ayrıştırıcılar oluşturamadığımızdan çok daha az kullanışlıdırlar.

Bu tezde içerikten bağımsız dilbilgisi veya CFG kullanacağız. Bağlamsız dilbilgisi tanımlamalarındaki (CFG) kurallar aşağıdaki şekildedir:

$$V \rightarrow w$$

burada V, terminal olmayan tek bir semboldür ve w, terminal olmayan veya boş bir dizidir. Sol taraftaki terminal ile sağ taraf değiştirilebildiğinde, resmi bir dilbilgisi "bağlam özgürlüğü" olarak kabul edilir.

1.5.3.2. Ayrıştırma Teknikleri

Ayrıştırıcı belirli bir CFG dizeleri üzerinde çalışır. Bir ayrıştırıcı, bir giriş arabelleği, bir yığın ve ayrıştırma tablosundan oluşur. Giriş akışını okuyarak, ayrıştırıcı kurallarını ayrıştırma tablosunda uygular.

Sonunda, yığın boş olduğunda veya diğer bazı özelliklere sahip olduğunda ayrıştırma tamamlanacaktır ve sadece bitiş sembolü, girişte kalmıştır. Sembol $, giriş akışı için bitiş sembolünü belirtmek için kullanılır.

Sembol $, ayrıştırıcı yığının S$ ile ayrıştırılmaya başladığında kullanılır; burada S, terminal olmayan bir başlangıçtır ve $, bitiş işaretidir.

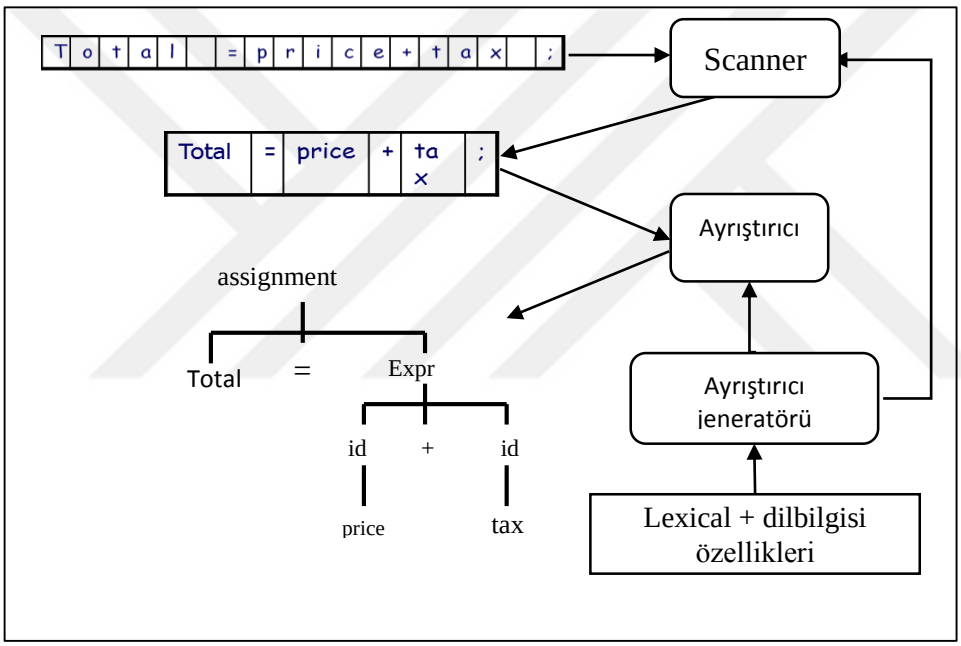
Ayrıştırma iki teknikle yapılabilir: yukarıdan aşağıya ve aşağıdan yukarıya. Burada, aşağı-yukarı tekniğini kullandıkları LL parserleri olarak adlandırılan bu iki ana ayrıştırıcıyı ve aşağıdan yukarıya tekniği kullandıkları LR ayrıştırıcılarını inceleyeceğiz. Ayrıca, [84,85] 'de bulunabilecek başka ayrıştırıcı varyasyonları da vardır.

1.5.3.3. Ayrıştırıcı Üretme Araçları

Derleyici inşası, bilgisayar bilimlerinde programlama dilleri geliştirme teori ve pratiğini ele alan geniş bir alandır. Bir derleyici-derleyici, dilbilgisinden belirli bir dil için herhangi bir

tarayıcı ve ayrıştırıcı üretmeye yardımcı olur. Giriş dilbilgisi derleyici-derleyici kurallarına uymalıdır. Bir derleyici-derleyici aslında, gramer kurallarını girdi olarak alan ve verilen bir dilbilgisi için çıktı olarak bir derleyici üreten bir derleyici üretecidir. Şekil 41, bir derleyici-derleyicinin, verilen dilbilgisi için tarayıcıyı ve ayrıştırıcıyı nasıl oluşturduğunu gösterir.

Otomatik olarak ayrıştırıcıları oluşturmak için, çeşitli dillerde kaynak kodları oluşturabilen birçok farklı üretim aracı vardır. Tipik olarak bu araçların örnekleri YACC [86] ve Bizon [87], zorunlu diller için, ML-YACC [88] ve fonksiyonel diller için Happy'dir [89], T-Gen [90], JavaCup [91] ve nesne yönelimli diller için JavaCC [92,93]. Giriş olarak özel bir gramer türü gerektirirler ve LL(k) veya LR(k) ayrıştırıcıları üretirler. Bu nedenle, bir ayrıştırıcı üreteci seçtikten sonra, o üretecin özelliklerine göre uygun bir dilbilgisi tanımlanmalıdır.



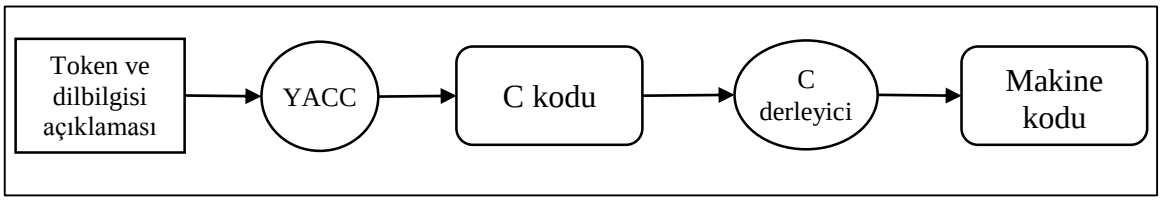
Şekil 41. Derleyici-Derleyici nasıl çalışır

Bu tez çalışmasında herhangi bir derleyici-derleyiciyi gözden geçirmeyeceğiz. Ancak, YACC ve JavaCC'yi bir dilin giriş dilbilgisinden derleyicileri üretmek için en popüler iki araç olarak tanıtıyoruz.

1.5.3.3.1. YACC

En tanınmış ve popüler derleyici derleyicilerinden biri, YACC'dir. YACC'nin çıktısı C dilinde bir derleyici programının kaynak kodudur (Şekil 42). YACC, LALR ayrıştırıcıları

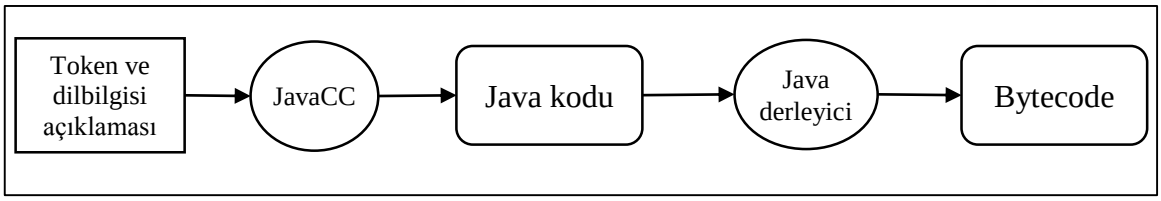
üretir. Bununla birlikte, tam bir sözdizimsel analiz için, Lex veya Flex gibi harici bir sözcük analizcisi gereklidir.



Şekil 42. YACC ile ayrıştırıcı üretimi

1.5.3.3.2. JavaCC

JavaCC, yukarıdan aşağı çözümleyici üreten bir Java tabanlı çözümleyici üretecidir. Yukarıdan aşağıya doğru ayrıştırıcılar veya özyineli terbiyeciler, daha genel dilbilgisi kullanımına izin verir. Bu ayrıştırıcıların tek sınırlaması, sola dönüşüme izin verilmemesidir, çünkü bu sonsuz tekrarlamaya yol açabilir. Yukarıdan aşağı ayrıştırıcıların dilbilgisi özelliklerine özdeş bir yapısı vardır ve bu nedenle hata ayıklaması daha kolaydır. Kullanıcı kodunun, yukarıdan aşağıya ayrıştırıcıda oluşturulmuş soyut sözdizimi ağacına gömülmesi, ayrıştırma ağacının düğümleri arasındaki bağımsız değişkenleri ve değerleri geçme kolaylığı nedeniyle basittir (Şekil 43).



Şekil 43. JavaCC ile ayrıştırıcı üretimi

JavaCC, giriş akışında bir tokeni önde bekleyen bir varsayılan mekanizmaya sahiptir. Bununla birlikte, birden fazla tokenin ileriye bakılabildiği bir yetenek sağlar. JavaCC ayrıca bazı belirsizlikleri ortadan kaldırmak için hem sözdizimsel hem de anlamsal bir görünüm sağlar.

JavaCC ayrıca Lex gibi sözcüksel analizör sağlar. Token Manager, deterministik olmayan sonlu otomasyonun bir uygulaması olduğu gramerin belirteçlerini tanımak için kullanılan JavaCC bileşenidir. JavaCC ve özellikleriyle ilgili daha fazla bilgi [94]'te bulunabilir.

1.5.4. Matematik İfadeler İçin Diller

Programlama dillerine benzer şekilde, matematiksel ifadeleri tanımlamak için de çeşitli matematiksel diller vardır. En iyi bilinen üç matematik dili aşağıda tartışılmıştır; MPL, MathML ve LaTex.

1.5.4.1. Matematiksel Sözde Dil (MPL)

Matematiksel sözde dil (MPL), Maple, Mathematica ve MuPAD'de kolayca ifade edilen bir algoritmik dildir. Özellikle matematiksel algoritmaları geliştirmek, test etmek ve iletmek için tasarlanmış, yüksek seviyeli, kullanıcı odaklı bir programlama dilidir.

1.5.4.2. MathML

MathML veya Mematsal İşaretleme Dili matematiksel ifadeleri tanımlamak için XML tabanlı bir dildir. Amacı matematiksel ifadeleri ve formülleri wep sayfalarına ve diğer belgelere entegre etmektir. MathML, Content MathML'nin anlamsal anlamını açıkladığı iki bölümden oluşur ve Presentation MathML ifadesi, ekran görünümünü ve ifadenin düzenini açıklar.

Matematiksel ifadelerin MathML'den daha yaygın ve daha fazla insan tarafından okunabilir bir temsili, ifadelerin sunumu olduğu ve Presentation MathML'ye kolayca dönüştürülebileceği LATEX tarafından sağlanır.

1.5.4.3. LaTex

Lamport Tex'in LaTeX kısaltması, bir belge hazırlama sistemi ve belge biçimlendirme dilidir. Bilimsel belgelerin matematik, fizik, bilgisayar bilimi, vb. dahil pek çok alanda iletişim ve yayınlanması için yaygın olarak kullanılmaktadır. Aynı zamanda Sanskrit gibi karmaşık çok dilli materyalleri içeren kitap ve makalelerin hazırlanmasında ve yayınlanmasında da projeksiyon rolü vardır. LaTeX, belirli bir düzenleme programının adı değildir, ancak LaTeX belgelerinde kullanılan kodlama veya etiketleme kurallarına atıfta bulunur.

TeX gibi, LaTeX de matematikçiler ve bilgisayar bilimcileri için bir yazma aracı olarak başladı, ancak gelişiminin başlangıcından itibaren karmaşık matematik ifadeleri ve Latince olmayan alfabeler içeren belgeleri yazması gereken bilim adamları tarafından da ele geçirildi. Şekil 44 Latex'te bir örneği göstermektedir.

```
\lim_{x \to \infty} \exp(-x) = 0
```

$$\lim_{n\to\infty} exp(-x) = 0$$

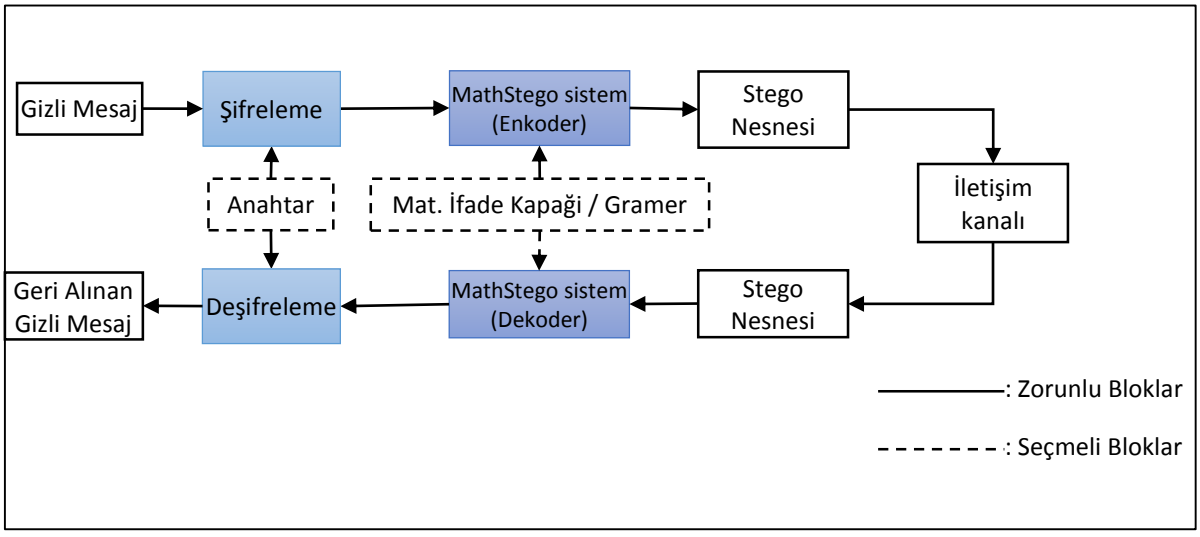
Şekil 44. LaTex'te bir matematik ifadesi örneği

2. YAPILAN ÇALIŞMA

2.1. Giriş

Bu çalışmada kriptoloji biliminde bulunan veri gizleme işlemi ele alınmıştır. Literatürde veri gizleme işlemini farklı medyalar üzerinden gerçekleştiren çok sayıda çalışma vardır ve bu tezde yeni bir veri gizleme medyası olarak matematiksel ifadeler kullanılmıştır. Genel olarak matematiksel ifadeler iki şekilde göz önüne alınabilir. Birincisi, var olan bir ifade kullanılarak, istenilen mesaj belli bir metodoloji üzerinden ifadeye gömülebilir. Burada matematiksel ifade önce ağaç üzerine taşınıp, daha sonra mesaj ifadeye gömülebilir.

İkinci yöntemde ise, mevcut bir ifade yerine, genişletilmiş bir grameri kullanarak gömülecek mesaj verileri ile yeni bir matematiksel ifadenin üretimi düşünülebilir. Her iki yöntemde de matematiksel ifadeler ile çalışmak gerekmektedir. Dolayısıyla, çalışmanın ilk kısmında matematiksel ifadeler ile çalışmak için gereken metotlar geliştirilmiştir. Daha sonra bu metotlar kullanılarak önerilen yöntemlere yer verilmiştir. Ayrıca, mesajlar gömülmeden önce iyi bir yöntemle şifrelenmesi için, önerilen yöntemin güvenliği daha da artırılabilir. O yüzden bu çalışmada kaotik özelliği taşıyan yeni bir şifreleme yöntemi önerilmiştir. Şekil 45 tezin genel yapısını göstermektedir.



Şekil 45. Tez çalışmasının genel yapısı

2.2. Matematiksel İfadelerin Değerlendirilmesi

Bu tezde, çok taraflı bir sistem şeklinde tasarlanan ve matematiksel ifadeler üzerinde değerlendirme yapabilen bir metodoloji önerilmiştir. Sistem, girdi verilerini sözdizimsel kurallara göre ayrıştırmak (parsing) için genişletilmiş bir gramer kullanır ve daha sonra belirli bir matematiksel ifadeyi kolayca işleyebilmek için uygun bir model sağlayan ağaç veri yapısına dönüştürür. Ağacın her düğümü, giriş ifadesinin bir sembolünü veya terimini ve sembollerin operatör, sayı vb. olabileceği diğer semboller veya terimlerle ilişkilerini içerir. Ağaç gösterimi ayrıca bir düğüm ve çocukları üzerinde tekrarlanan işlemlerin uygulanmasına yardımcı olur. Örneğin, bir düğümün değerlendirilmesi öncelikle o düğümün çocuklarını değerlendirmeyi gerektirir. Belirli bir düğüm kombinasyonunun yanı sıra, diğer belirli işlemlerin uygulanması gereken bir model tanımlayabilir. Metodoloji aşağıdaki aşamaları ve adımları içerir:

Aşama 1: Ayrıştırıcı (parser) tanımı

Adımlar: Gramer tasarımı, Gramer dönüşümü, Derleyici-derleyici (Compiler-Compiler) giriş yapısı, kod üretimi

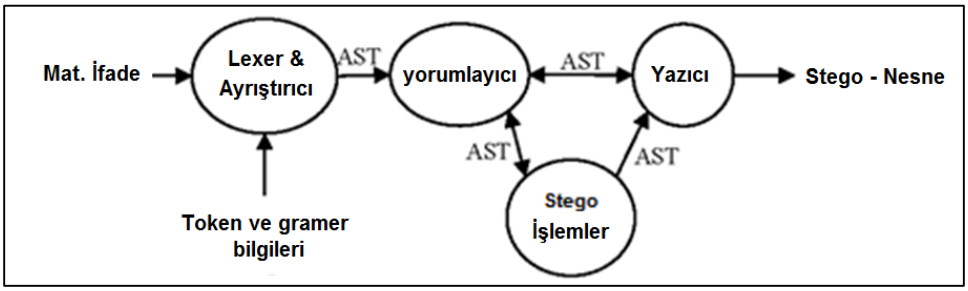
Aşama 2: Giriş veri analizi

Adımlar: token üretimi, sözdizimi (syntax) analizi, AST üretimi

Aşama 3: Değerlendirme ve yorumlama

Adımlar: İfadeleri çözme, sadeleştirme, ara değerlendirme sonuçlarını yazdırma

Bu aşamaların ve adımların bazıları Şekil 46'de de gösterilmiştir.



Şekil 46. Önerilen metodolojinin bazı önemli aşamaları ve adımları

Şekil 46'da gösterilen adımları kullanarak, matematiksel ifadeleri işlemek için bir çatı (framework) geliştirilebilir. Bu tezde, metodolojinin bazı uygulamalarını göstermek için cebirsel ifadeler üzerinde çalışacağız. Metodoloji, eğitim, iş dünyası veya endüstride birçok pratik veya gerçek dünya problemini çözmek için geliştirilen sistemlerle entegre edilebilir. Bu bir matematiksel modelin geliştirilebileceği tüm sistemlerde önemli bir bileşen olarak hizmet

verme potansiyeline sahiptir. Örneğin olası bir kullanım alanı, elektrik devrelerinin analizinden elde edilen doğrusal veya doğrusal olmayan denklem sistemlerinin çözümü olabilir. Diğerleri, kök bulma, fonksiyon enterpolasyonu, polinom yaklaşımı, eğri uydurma dahil olmak üzere birçok sayısal analiz yönteminin değerlendirilmesi, düzeltilmesi ve uygulanması olabilir. Çeşitli tiplerdeki diferansiyel denklemleri tanımlamak ve çözmek için de kullanılabilir.

2.2.1. BNF Gramer Tanımı

BNF notasyonu, özellikle programlama dilleri olmak üzere, biçimsel diller için içerikten bağımsız (başka bir ifadeyle, bağlam-içermeyen) gramerler tanımlamada kullanılır. Basit notasyonlara ve özyinelemeli yapılara sahiptir. YACC [86], LEX ve JavaCC [92,93] gibi birçok derleyici oluşturma aracı bir kaynak dilin BNF benzeri tanımını kullanır. Matematiksel ifadeler toplama veya çıkarma, *sin* veya *cos* gibi işlevler, integral gibi özel semboller gibi işlemleri içerebilir. Belirli bir matematiksel ifade için genişletirilmiş bir gramer verildiğinde, tüm operatörler, fonksiyonlar, semboller, değişkenler ve sayılar terminal setinin üyeleri olacaktır. Non-terminal küme gramerin üretim kurallarına göre belirlenir.

Liste 1. Matematiksel ifade için Genişletilmiş-BNF grameri

```
G = {Σ, T, V, P, S}
V = {expr, op, func, var, number, digit}⊆ Σ
T = {x, Sin, Cos, Tan, Log, Exp, Sqrt, +, -, *, /}⊆Σ
Σ = T ∪ V, S = { expr }
<expr>::= <expr> <op> <expr>  |(<expr>)
         |<func>(<expr>)   |<var>     |<Number>
<op>::='+' | '-' | '*' | '/' | '^'
<func>::= 'Sin' | 'Cos' | 'Tan' | 'Log' |'Exp' | 'Sqrt’
<var>::='x'
<number>::= '-' ? <digit> + ('.' <digit> +)?
<digit>::=['0'-'9']
```

Tasarlanan gramer, bir operatör ve onun işlenenleriyle (operands) veya argümanlarıyla bir matematiksel işleve sahip aritmetik ifadeler üretmelidir. Bir işlenenin (operand) veya argümanın kendisi bir sayı, bir değişken veya başka bir matematiksel ifade olabilir. Gramerin üretim kuralları özyinelemeli olabilir, çünkü "expression" türünde işlenenler, kısa bir süre "expr" olarak adlandırılır. Ek olarak, ondalık sayılar (veya tamsayılar) istenilen basamak sayısına kadar oluşturulabilir. Her farklı matematik ifadesi farklı gramerin kullanılmasını

gerektirir. Bu bölümde, Liste 1'de gösterilen ve [95]'te sunulan birtakım değişiklikler ile genişletilmiş bir BNF grameri geliştirilerek bir türev sistem uygulanmaktadır.

Liste 1'de verilen gramer, basit matematiksel ifadede yer alabilen beş operatör ve altı fonksiyona sahiptir. Ancak, diğer bazı operatörlerin, fonksiyonların ve simgelerin eklenmesiyle değiştirilebilir. Yukarıdaki gramer, örneğin, "3 * x + x ^ 2/2" ifadesini temsil etmek için kullanıldığında, bir anlamsal (semantik) problemle karşılaşacaktır. Gramer tanımı operatörlerin önceliğini göz önünde bulundurmadığı için, ifadeye matematikte yapılandan farklı bir anlam verilecek ve bu nedenle işlemler doğru sırada değerlendirilmeyecektir. Bir sonraki bölümde, bu çeşit anlamsal problemleri çözmek için gramer değiştirilecektir.

2.2.2. Gramer Dönüşümü

Bir ifadeyi ayrıştırmak (parse), bir gramer yoluyla ifade yapısını işlemek anlamına gelir. Gramer ve ifade arasındaki en önemli bağlantılardan biri, parse ağacının türetilmesidir. Bu bağlantı, bazı belirli koşullar altında çalışan bir ayrıştırma yöntemi ile belirlenebilir. Bu çalışmada, LL ayrıştırıcılarına (parser) odaklanırız. Liste 1'de tasarlanan gramer, geleneksel parse yöntemlerinden birini kullanmak üzere dönüştürülmelidir. Derleyici-Derleyici (DD) veya literatürde compiler-compiler (CC) adı verilen parser üretimi için geliştirilmiş çeşitli araçlar vardır. Her bir CC, belirli bir parse yöntemi için uygundur. Aslında bu araçlar, girdi olarak bir biçimsel dilin sözdizim kurallarını kullanarak çıktı olarak bu dil için bir derleyiciyi döndüren üreteçleridir. Bu tezde, yukarıdan aşağıya parse işlemi için geliştirilmiş bir CC aracı (veya ayrıştırıcı üreticisi) olarak JavaCC kullanılmıştır. Yukarıdan aşağıya doğru ilerleyen parserler, genel gramerler ile kolayca çalışabilir. Liste 1'deki gramer göz önüne alındığında, tüm operatörlerin aynı öncelik seviyesine sahip olduğunu görmek kolaydır. Bilindiği gibi, bu onu belirsiz bir gramer yapacaktır. JavaCC gibi bir CC aracı kullanmak için gramer LL(1) olmalıdır. Liste 2, operatörün önceliği ve birleşme yönü dikkate alınarak değiştirilen eşdeğer LL(1) gramerini göstermektedir.

Liste 2'deki gramer, normal LL(1) gramerin optimize edilmiş bir versiyonu olup, tüm ekstra terminaller, orijinal olmayan terminale geri birleştirilir. Gramer tarafından oluşturulan ağaç yapısı göz önünde bulundurulduğunda, daha yüksek önceliğe sahip operatörler ilk önce ve daha düşük öncelikli olanlar daha sonra değerlendirilecektir. Diğer matematiksel operatörlere ihtiyaç varsa, öncelik düzeyi ile birleşme yönü dikkate alınarak gramere eklenmesi gerekir. Bu şekilde, "3 * x + x ^ 2/2" ifadesi değerlendirilecektir.

Liste 2. Matematiksel ifadeler için bir LL(1) grameri

```
G={Σ, T, V, P, S}
V={expr, element, term, unary, power, func, number, digit}⊆ Σ
T={x, Sin, Cos, Tan, Log, Exp, Sqrt, (,), +, -, *, /, ^}⊆Σ
Σ = T ∪ V
S = { expr }
<expr>⟶<unary> <term> [("+"|"-") <term> ]*
<unary>⟶("+"|"-")?
<term>⟶<power> [("*"|"/")<power>]*
<power>⟶<element> ("^"<power>)?
<element>⟶<func> (<expr>) | <number> | "x"
<func>⟶"Sin" | "Cos" | "Tan" | "Log" | "Exp" | "Sqrt"
<number>⟶"-"? <digit> + ("."<digit>+)?
<digit>⟶["0"-"9"]
```

2.2.3. Sözdizimi Sınıflarının Tanımı

Sözdizimi bir dilin yazma kurallarıdır. Matematiksel ifadelerin diğer diller yanı sıra, bileşenlerin sınırlı bir kombinasyonu olduğunda belirli bir sözdizimine sahiptir. Matematik ifadelerinde sayılar, değişkenler, işlemler, fonksiyonlar, gruplama sembolleri ve diğer sözdizimsel semboller kullanılır; her bileşen tanımlanması gereken belirli bir yapıya sahiptir. Çalışmada, Java'nın nesne yönelimli (Object-Oriented) kavramları kullanılarak uygulanan sözdizimi sınıfları kullanılmıştır. Tablo 4, bazı sözdizimi sınıflarını ve çalışmalarımıza uyarlanmış niteliklerini göstermektedir.

Tablo 4. Bazı matematiksel bileşenler için sözdizimi sınıfları

Bileşen	**Sözdizimi Biçimi**	**Sözdizimi Sınıf**	**Özellikler**
+	Exp + Exp	Plus	exp1, exp2
*	Exp * Exp	Times	exp1, exp2
^	Exp ^ Exp	Power	exp1, exp2
Sin	Sin(Exp)	Sin	exp
Cos	Cos(Exp)	Cos	exp
Numbers	*n*	Num	*n*
Variables	----	Var	----

Tablo 4'de olduğu gibi, diğer operatörler, fonksiyonlar veya semboller için ek sözdizimi sınıfları tanımlanabilir. Nesne yönelimli programlamada, her kural genellikle bir sınıf

tarafından tanımlanır ve daha sonra ifadeleri değerlendirmek için kullanılır. Liste 3, Tablo 4'de verilen bazı sınıfların tanımlarını göstermektedir. Sözdizimi sınıfları, bazı yapıları oluşturmak için kullanılabilir. AST ağaç yapısı, matematiksel ifadeler için en uygun olanıdır. Parse ağaçlarından farklı olarak, bir AST bir ağaç şeklinde girdinin önemli kısımlarını tutabilir. Bir AST onu üreten bir parser ile eş zamanlı olarak tanımlanabilir.

Liste 3. Tablo 4'de gösterilen bazı sözdizimi sınıflarının tanımı

```
public abstract class Exp
{
    public Exp exp1, exp2;
    public Exp (Exp e1, Exp e2){
        this.exp1 = e1;
       this.exp2 = e2;
   }
}
public class Plus extends Exp
{ ... }
...
public class Sin extends Exp {
    public Sin(Exp e) {
        super(e, null);
    }
} ...
public class Num extends Exp {
    public double num;
    public Num(double n){
        this.num = n;
    }
}
public class Var extends Exp {
    private String var;
    public Var() {
        super(null, null)
    }
    public Var(String var) {
        super(null, null);
        this.var = var;
    }
}
```

Bu aşamada tanımlanan sözdizimi sınıfları, matematikteki ilgili operatörlerin özelliklerine uygun olarak kodlanmalıdır. Liste 4'te görüldüğü gibi, Exp sınıfı abstract olarak tanımlanmıştır ve ifadelerde gerçekleştirilebilecek çeşitli işlemlere bağlı olarak, bu sınıfa metotlar eklenebilir. Örneğin, bir ifadenin sonucunu hesaplamak için, *public abstract double Eval (int x)* gibi bir yöntem; Exp sınıfına eklenebilir. Liste 4, bazı operatörler için bu yöntemin tanımlarını gösterir.

Liste 4. Eval () yöntemi için bazı sözdizimi sınıflarının tanımı

```
abstract class Exp {
    …
    public abstract double Eval(int x);
}
class Plus extends Exp {
    Exp exp1, exp2, tmp;
    public Plus(Exp e1, Exp e2) {
        exp1 = e1;     exp2 = e2;
    }
    public double Eval(int x) {
        return (exp1.Eval(x)+exp2.Eval(x));
    }  …
}

class Minus extends Exp {
    Exp exp1, exp2;
    Data val2;
    Exp tmp;
    public Minus(Exp e1, Exp e2) {
        exp1 = e1;       exp2 = e2;
    }
    public double Eval(int x) {
        return (exp1.Eval(x)- exp2.Eval(x));
    }  …
}
class Times extends Exp {
     Exp exp1, exp2, tmp;
     Data val2;
     public Times(Exp e1, Exp e2) {
        exp1 = e1;       exp2 = e2;
     }
     public double Eval(int x) {
        return (exp1.Eval(x)*exp2.Eval(x));
     }
     …
}
class Var extends Exp {
     public Var(String var) {
         this.var = var;
     }
     public double Eval(int x) {
         return (x);
     }  …
}
```

Liste 4'te tanımlanan yöntemi kullanarak, "*x* * *x* + *x* ^ 2/2" gibi bir matematiksel ifadenin sayısal değerini, *x* değişkeni için belirli bir değerle hesaplamak kolaydır.

2.2.4. Parser Yapılışı

2.2.4.1. Belirteç (Token) Özellikleri

Kelime perspektifinden girdi analizi ile bir token üreticisi (tarayıcı) bir token dizisi üretir. Giriş verilerindeki, daha küçük parçalara bölünemeyen herhangi bir kelimeye token denir. Token'ler, Liste 2'de verilen gramer ile tanımlanan terminal kümesindeki ($T \subseteq \Sigma$) tüm öğelerdir. Liste 2'deki gramer için, Token'ler, JavaCC'nin bildirim (specification) kurallarını izleyerek Liste 5'te gösterilmiştir.

Liste 5. Liste 2'deki gramer terminallerinin JavaCC token bildirimleri

```
TOKEN : {
        < NUMBER : (["0"-"9"])+("." (["0"-"9"])+)? >
}
TOKEN : {
       < EOL : "\n" >
}
TOKEN : /* OPERATORS */ {
        < PLUS: "+" > | < MINUS: "-" > | < TIMES: "*" >
       | < DIV : "/" > | < POW : "^" >
        ...
}
TOKEN : /* FUNCTIONS */ {
        < SQRT: "sqrt" > | < SIN: "sin" > | < COS: "cos" >
        ....
}
TOKEN : /* SYMBOLS */ {
       < X: "x" > | < LPR: " (" > | < RPR: ")" >
}
SKIP : { " " | "\t" | "\r" }
```

Liste 5'te her token, TOKEN anahtar kelimesi ile tanımlanır. Basitlik için, tüm token'leri sayı, operatör, vb. gruplara ayırırız. SKIP anahtar kelimesi, atılması gereken karakterleri belirtir. Benzer şekilde, gramerdeki diğer token'ler Liste 5'e eklenebilir. Aslında, bu aşamada, token'ler, JavaCC ile üretilen bir sözcük (lexical) analizcisi tarafından tanınan düzenli ifadelerle tanımlanır. Örneğin, "3 * *x* + *x* ^ 2/2" ifadesi, aşağıdaki token'lerin bir akışına (stream) sahip olacaktır: *NUMBER TIMES VAR PLUS VAR POW NUMBER DIV NUMBER*

2.2.4.2. Parser Tanımı

Bir parser, girdi verilerini dizisl oluşturulmuş tokenler'e dayanarak üretilebilirlik açısından analiz eder. Gramer kurallarına göre inceleme yaparak token dizisinin oluşturulup oluşturulamadığını kontrol eder. Bu nedenle, token'lerin doğrulanması için bir mekanizmaya ve dil kurallarına göre onların görünüş sırasına ihtiyaç vardır. Elbette, analiz sürecinde, sistem değerlendirme yapmadan önce kabul edilebilir girdi verileri için bir anlamsal analiz yapmalıdır. Ancak, sözdizimi analizcisi (parser) matematiksel ifadelerin yapıları nedeniyle bu verilerin değerlendirilebilirliğini garanti edebilir. Parser elle yazılmış (hand-written) fonksiyonlar veya ayrıştırıcı üretici araçları kullanılarak tasarlanabilir. Bir parser oluşturma aracının kullanılması durumunda, bu parser'in açıklama koşullarına dayalı olarak istenen gramerin geliştirilmesini içerir. Örneğin, Yacc için bir gramer LR'de ve CppCC için LL biçiminde geliştirilmelidir. Bu tezde girdilerin parser üretimi ve geçerlilik testi için JavaCC kullanılmıştır. JavaCC bildirimi (declaration) içindeki işlevlerin veya metotların adı, Liste 2'deki gramer içindeki non-terminal kümeye göre belirlenir. Genel olarak, bir gramerde her non-terminal için bir metot tanımlanmalıdır. Bazı durumlarda, non-terminal birkaç sembolü birleştirmek yararlı olabilir, sadece onlar için bir metot tanımlanır. Örneğin, Liste 1'deki <expr> non-terminalini göz önüne alalım. Bu non-terminal için, gramer içerisinde <expr> ⟶ <term> <expr'> gibi bir kural vardır. Bu nedenle, parser'de bir metot tanımlanabilir. Ancak ilgili non-terminal <expr' > için parse ağacı üretiminde bazı zorluklar ortaya çıkabilir. Bu gibi durumlar için, bu terminaller birleştirilebilir, bir non-terminale indirgeme yapılabilir ve parser üretici aracında sadece bir metot ile temsil edilebilir. Tablo 5 birleştirme işleminin tipik örneklerini göstermektedir

Tablo 5. Birleştirme işleminin tipik örnekleri

Normal kurallar	Birleşik Kurallar
< *expr* >⟶< *term* > < *expr′* > < *expr′* > ⟶("+" | "-") < *term* > < *exprv′* > < *expr′* > ⟶λ	< *expr* >⟶< *term* > { ("+" | "-") < *term* > }*
< *term* >⟶< *unary* > < *term′* > < *term′* >⟶("*" | "/") < *unary* > < *term′* > < *term′* >⟶λ	< *term* >⟶ < *unary* >{ ("*" | "/") < *unary* >}*

Bütün parser metotları gramer kurallarının yapısına uygun olarak tanımlanır. Tablo 6'te, LL(1) gramerinin bazı JavaCC tanımlı metotları gösterilmiştir.

Gramer kuralları için diğer metotlar Tablo 6'te görüldüğü gibi tanımlanabilir. Token'lerin ve kuralların bütün tanımları, girdi olarak JavaCC tarafından okunan ".jj" uzantılı bir dosya içerisinde belirtilmelidir. JavaCC'nin çıktısı, ilgili gramer için bir parser görevi gören Java kodudur. Bu parser, girdi verilerinin doğrulamasını yapmak için kullanılabilir. Tablo 6'te, token'ler <EOL> ve <EOF>, sırasıyla satır sonu ve veri girişlerinin sonunu temsil eder. Bu aşama, ifade yapısının doğruluğunu kontrol etmek veya sözdizimi ağacının üretimine yönlendirmek için kullanılabilir. Önceki aşamada tanımlanan matematiksel ifade, belirli bir gramere dayalı olarak değerlendirilir. "3 * x + x ^ 2/2" ifadesinin sözdizimi hatası içermediğini ve iyi oluşturulmuş olduğunu görmek kolaydır.

Tablo 6. LL(1) gramer kuralları için örnek JavaCC metot tanımları

Gramer kuralı	İlgili Yöntemi

< start >⟶< expr > $	void parse() : { } {
	expr() (<EOF> | <EOL>)
	}

< expr >⟶< term > { ("+" | "-v) < term > }*	void expr() : { }{
	term()(<PLUS> term() | <MINUS> term())*
	}

< power >⟶< element > ("^v< power >)?	void power() : { }{
	element() (<POWER> power())?
	}

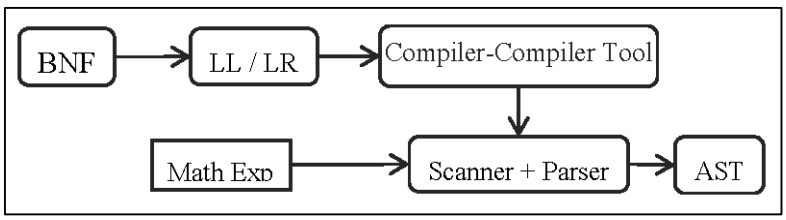
2.2.5. Soyut Sözdizim Ağacı Oluşturma (Abstract Syntax Tree-AST)

Bir önceki bölümde, veri girişinin doğruluğunu belirlemek için bir parser oluşturma süreci tartışıldı. Ancak, matematiksel ifadeler üzerinde bazı işlemleri yapabilmek için, sadece doğruluğu değil, aynı zamanda istenen veri yapısının da oluşturulması gereklidir. Matematiksel ifadeler için yararlı yapılardan biri, sözdizim ağacıdır. Bir sözdizimi ağacı (veya esas olarak nesne ağacı), hiyerarşik bir yapıda birbirine bağlanan birçok düğümden oluşur. Her düğüm bir sözdizimi sınıfından türetilir ve kaynak verilerinin bir ifadesini temsil eden bir nesne tarafından oluşturulur. Sözdizimi ağacı genellikle bir süper sınıf tür yapısına sahiptir. Aynı süper sınıftan miras alan alt sınıflar, bir nesne ağacının düğümlerini oluşturmak için kullanılabilir, ancak ağaç üzerindeki bu düğümler, süper sınıfın referansıyla temsil edilebilir. JavaCC'de çeşitli Java ifadeleri ekleyerek, AST oluşturmak mümkündür. Liste 6, AST düğümlerini oluşturmak için Tablo 6'daki koda eklenecek komutları gösterir.

Liste 6. AST üretmek için eklenen Java ifadeleri

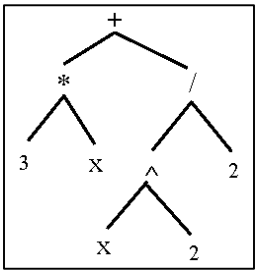
```
Exp parse() : { Exp a; }
{
     a = expr() (<EOF> | <EOL>) { return a; }
}
Exp expr() : { Exp a, b; }{
     a = term() (
        <PLUS> b = term()  { a = new Plus(a, b); }
      | <MINUS> b = term() { a = new Minus(a, b); }
     )*
     { return a; }
}
...
Exp power() : { Exp a, b; }{
     a = element() (
         <POWER> b = power() { a = new Power(a, b); }
    )?
    { return a; }
}
...
Exp element() : { Token t; Exp a; } {
    t=<NUMBER>
       { return (new Num(Double.parseDouble(t.image))); }
    | <X> { return (new Var()); }
    | <LPR> a = expr() <PPR> { return a;}
    | <SIN> <LPR> a=expr() <PPR> { return new Sin(a);}
    ...
}
...
```

Tablo 6'ya eklenen kodlar, giriş dizgisini doğrulayarak çıktı verilerini üretir. Şekil 47 matematiksel gramere ait AST'yı üretmek için gerekli adımları göstermektedir.



Şekil 47. Matematiksel gramer için AST üretme adımları

Örneğin, "3 * x + x ^ 2/2" ifadesi, Şekil 48'de grafiksel olarak gösterildiği gibi bir AST'ye dönüştürülecektir.



Şekil 48. "3x + x ^ 2/2" ifadesi için üretilen AST

Aslında, bu AST, parser tarafından aşağıda verilen Java ifadesi biçiminde oluşturulur:

Exp ast = new Plus(new Time(new Num(3), new Var()), new Divide(new Pow(new Var(), new Num(2)), new Num(2)))

Bu tezde tüm AST örneklerinin okunabilirliğini artırmak için, yukarıdaki AST'nın daha kullanışlı bir temsilini şöyle uyarlayabiliriz:

Exp ast = Plus(Time(Num(3), Var()), Divide(Pow(Var(), Num(2)), Num(2)))

2.2.6. Matematiksel İfadelerin Değerlendirilmesi

Giriş verilerinin doğrulanması aşaması, ilgili AST'nin olşturulmasıyla sonuçlanır. Bir sonraki aşama bu AST'nin değerlendirmesi ile ilgilidir. Genel olarak, matematiksel bir ifade, ya doğrudan ya da bu ifadenin AST'sini kullanarak iki yaklaşımla değerlendirilebilir. İlk yaklaşım, parser'in doğrulama aşaması sırasında ifadenin token bileşenlerinin değerlendirilmesini ifade eder. Diğer yaklaşım, token'in bulunduğu düğümler (residing node) ziyaret edilerek, ilgili ifadenin AST temsilini değerlendirmeyi içerir. Bu iki yaklaşım aşağıdaki bölümlerde açıklanmıştır.

2.2.6.1. Matematiksel İfadelerin Doğrudan Değerlendirilmesi

Basit matematiksel ifadelerin değerlendirilmesi, parse sırasında yapılabilir. Bu durumda, bir girdi ifadesinden herhangi bir veri yapısının yaratılmasına gerek yoktur ve istenen değerlendirme doğrudan parser'e gömülebilir. Bir önceki bölümde görüldüğü gibi, parser üretici araçlarını kullanma sürecinde programlama dili ifadelerinden yararlanmak mümkündür. Böylece istenen işlem değerlendirme kodu parser metotlarına ekleyerek gerçekleştirilebilir. Örnek olarak, bir değişkenin belirli bir değerine dayanarak matematiksel bir ifadeyi hesaplamayı düşünelim. Bu amaçla, değişkenin değeri metotlara bir argüman olarak iletilir.

Her bir metotta, ilgili işlem ifadenin ilgili parçası üzerinde gerçekleştirilir ve sonuç geri döndürülür ki bir double olabilir. Liste 7, JavaCC'deki bu metotlardan bazılarını göstermektedir. Liste 5 ve Liste 6'da verilen fonksiyonlar arasında anlamlı bir fark, dönüş değerlerinin tipi ile ilişkilidir. Liste 6'daki işlevler her zaman bir sayı döndürmelidir. $x = 2$ için değerlendirilen "$3x^3 + 7x + 1$" ifadesinin basit bir örneği göz önüne alındığında, parametre geçişi ve değerlendirmesinin birleştirilmiş işlemi 39 değerini döndürecektir.

Liste 7. String tabanlı değerlendirme için bazı JavaCC fonksiyonları

```
double parse(double x) : { double a; }
{
     a = expr(x) (<EOF> | <EOL>) { return a; }
}
double expr(double x) : { double a, b; }{
     a = term(x) (
       <PLUS> b = term(x) { a = a+b; }
     | <MINUS> b = term(x) { a = a-b; }
     )*
     { return a; }
}
double power(double x) : { double a, b; }{
     a = element(x) (
       <POWER> b = power(x) { a = Math.pow (a, b); }
     )?
     { return a; }
}
double element(double x) : { Token t; double a;} {
     t = <NUMBER> { return Double.parseDouble(t.image); }
     | <X> { return x; }
     | <LPR> a = expr(x) <RPR> { return a; }
     | <SIN> <LPR> a = expr(x) <RPR> { return Math.sin(a); }
     ...
}
```

2.2.6.2. AST Kullanarak Matematiksel İfadelerin Değerlendirilmesi

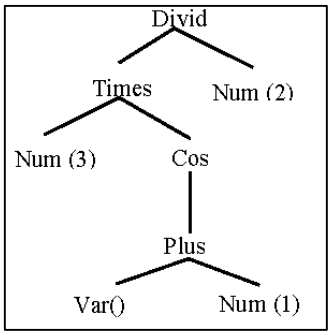
Bu yaklaşım, bu tezdeki metodolojinin bazı özel uygulamaları aracılığıyla verilen ifadelerin ara gösterimi üzerinde çalışır. Bilindiği gibi, parse süreci, kaynak veri olarak alınan bir ifade bileşenleriyle nesnelerden oluşan hiyerarşik bir yapıda AST üretir. Örnek olarak "3*x + x ^ 2/2" giriş dizgesi, parser tarafından aşağıdaki gibi bir AST'ye dönüştürülür:

Plus(Time(Num(3), Var()), Divide(Pow(Var(), Num(2)), Num(2))).eval()

Sınıf yapıcısı (constructor) *Var*, *x* değişkenini gösterir. Birden fazla değişkenin olduğu ifadeler için, değişkenin adını yapıcıya argüman olarak iletebiliriz. Bu tezde temel olarak sözdizimi sınıflarına metot ekleme yaklaşımı kullanılmıştır. Ancak, bazı durumlarda, değerlendirmeyi uygulamak için iki yaklaşımın bir kombinasyonu gereklidir. Bu şekilde, instanceof operatörü esas olarak sözdizimi sınıflarına eklenen metotlarda kullanılır. Bu ihtiyaç, düğümler ziyaret edildiğinde, her düğümün çocuğunun türünü belirleme gereksiniminden kaynaklanır. Ara çözüm adımlarını göstermek için, bir düğüm ziyaret edildiğinde, bu düğümün daha ileri değerlendirme için çocuğa sahip olup olmadığını bilmemiz gerekir. Örneğin, alt düğümün tipi *Num* ise, bu düğümlerin daha fazla değerlendirmeye gerek duymayacağı açıktır.

2.2.7. Matematiksel İfadelerin Gösterimi

AST ile temsil edilen bir matematiksel ifadenin her değerlendirme aşamasında, ortaya çıkması gereken ifade, *LaTex* veya *MathML* gibi insan tarafından okunabilir formatlardan birine dönüştürülmelidir. Bu, özellikle AST'deki bir ifade değerlendirme süreci sonucunda değişime uğradığında, yeni ifadeyi görüntülemek için gereklidir. İkili arama tekniği ile AST düğümleri ziyaret edilerek, değerlendirme sonucu ortaya çıkan ifadenin terimlerini elde etmemizi sağlayacaktır. Şekil 49, "(3*cos (x + 1)) / 2" ifadesi için AST yapısını göstermektedir.



Şekil 49. "(3*cos (x + 1)) / 2" ifadesinin AST'si

Şekil 49'daki AST'nin temsili aşağıdaki gibi verilebilir.

Divide (Times(Num(3), Cos(Plus(Var(), Num(1)))), Num(2))

AST'deki matematiksel ifadeleri görüntülemek için, aşağıdaki bölümlerde ele alınan farklı formatlara dönüşüm yapılabilir.

2.2.7.1. İnsan-Okunabilir Format

Biçim dönüştürme metotlarını her bir sınıfa uygulayarak, gerekli çıktı formatı oluşturulabilir. Matematiksel ifadelerin insan tarafından okunabilir formatta görüntülenmesi, AST'nin ikili traversiyle elde edilebilen bir dizgeye dönüştürülmesini gerektirir. Tablo 4, bazı sınıflar için Yazdırma metotlarını göstermektedir. Tablo 7'de insan tarafından okunabilir çıktı oluşturmak için sınıflara eklenen *Print()* metotları gösterilmiştir.

Tablo 7. İnsan tarafından okunabilir ifadeler üreten *Print()* metotları

Sınıf adı	*Print* Metodu

Exp	public String Print() { return ""; }
	public static String Out(String str) {
	if(str.contains("-") || str.contains("+") ||
	str.contains("*") || str.contains("/"))
	str = " (" + str + ") ";
	return str;
	}

Plus	public String Print() {
	String a = exp1.print();
	String b = exp2.print();
	return Exp.Out(a) + "+" + Exp.Out(b);
	}

Minus	public String Print() {
	String a = exp1.print();
	String b = exp2.print();
	return Exp.Out(a) +"-" + Exp.Out(b);
	}

Times	public String Print() {
	String a = exp1.print();
	String b = exp2.print();
	return Exp.Out(a) + "*" + Exp.Out(b);
	}

Functions	public String Print() {
	return "sin(" + exp.Print() + ")";
	}

Num	public String Print() {
	return Integer.toString(num);
	}

Bu metotların çıktısı doğrudan kullanıcıya gösterilebilir. Daha güzel sonuçlar elde etmek için ek kodlar ve metotlar eklenebilir. Metotların çıktısı, örneğin, " (3*cos (x + 1)) / 2" gibi bir ifadedir.

2.2.7.2. LaTex Biçimi

LaTex, bilimsel topluluk tarafından yaygın olarak kullanılmaktadır. Çıktıda AST'yi görüntüleyen diğer formatlara benzer olarak, parser kodlarına çıktı kodları ekleyerek bir *LaTex* belgesi oluşturulabilir. Tablo 8, bazı sözdizimi sınıfları tarafından sağlanan ve *LaTex* formatlı verileri çıkaran LaTex metodunu göstermektedir. *MathML* ayrıca *LaTex* girişini de kabul etmektedir. Bu nedenle, sonuç *LaTex* formatıyla dışarıya aktarılabilir ve daha sonra görüntülemek için *MathML* işleyicisini (renderer) kullanabilir. Daha önce bahsedildiği gibi, *LaTex* çok popüler bir formattır ve diğer birçok araç tarafından desteklenir.

Tablo 8. LaTex ifadeleri oluşturmak için *LaTex()* metotları

Sınıf adı	*LaTex* Metodu	Sınıf adı	*LaTex* Metodu

Exp	public String LaTex() {
	return "";
	}
	public static String Out(String str) {
	if(str.contains("-") ||
	str.contains("+") ||
	str.contains("*") ||
	str.contains("/")
	)
	str = " (" + str + ")";
	return str;
	}
		Plus	public String LaTex() {
			String a = exp1.LaTex();
			String b = exp2.LaTex();
			return Exp.Out(a) + "+" +
			Exp.Out(b);
			}

Divide	public String Print() {
	String a = exp1.LaTex();
	String b = exp2.LaTex();
	return "\\frac {" + a + "} {" + b + "}";
	}
		Power	public String LaTex() {
			String a = exp1.LaTex();
			String b = exp2.LaTex();
			return " (" + a + ") ^ {" + b + "}";
			}

Functions	public String LaTex() {
	return "\\sin {" + exp.Print() + "}";
	}
		Num	public String LaTex() {
			return Integer.toString(num);
			}

LaTex çıkışını anlamak kolaydır; örneğin, LaTex formatında " (3*cos (x + 1)) / 2" çıktısı "\ frac {(3 \ sin (x + 1))} {2}" şeklinde üretilir.

2.2.7.3. MathML Biçimi

MathMl, web sayfalarında kullanımı nedeniyle çok popüler olmuştur. *MathML* formatında bir dizgi üretmek için *MathML* SDK içindeki gömülü (built-in) fonksiyonları kullanmak mümkündür. Ancak, AST üzerinde tanımlanan bazı basit metotlar eklenerek de aynı format verileri üretilebilir. Önceki bölümde olduğu gibi, her bir sözdizimi sınıfının kendi yazdırma metodu vardır, ancak çıktı *MathML* etiketleri (tags) arasında sarılır. Örneğin, *Num* ve *Var* sınıfları sırasıyla <mi> ve <mo> etiketleriyle çevrelenmelidir. Tablo 9, bazı sınıflar için *MathML()* metodunu göstermektedir.

Tablo 9. *MathML* verisi oluşturmak için *MathML()* metotları

Sınıf adı	MathML Metodu

Exp	public String MathML() {
	return "";
	}

Plus	public String MathML() {
	String a = exp1.MathML();
	String b = exp2.MathML();
	return "<mi>" + a + "<mo>+</mo>" + b + "</mi>";
	}

Divide	public String MathML() {
	String a = exp1.MathML();
	String b = exp2.MathML();
	return "<mfrac>" + a + b + "</mfrac>";
	}

Times	public String MathML() {
	String a = exp1.MathML();
	String b = exp2.MathML();
	return "<mi>" + a + "*" + b + "</mi>";
	}

Functions	public String MathML() {
	return "<mi>sin</mi>" + "<mi>" + exp.MathML() + "<mi>";
	}

Num	public String MathML() {
	return "<mn>" + Integer.toString(num) + "<mn>";
	}

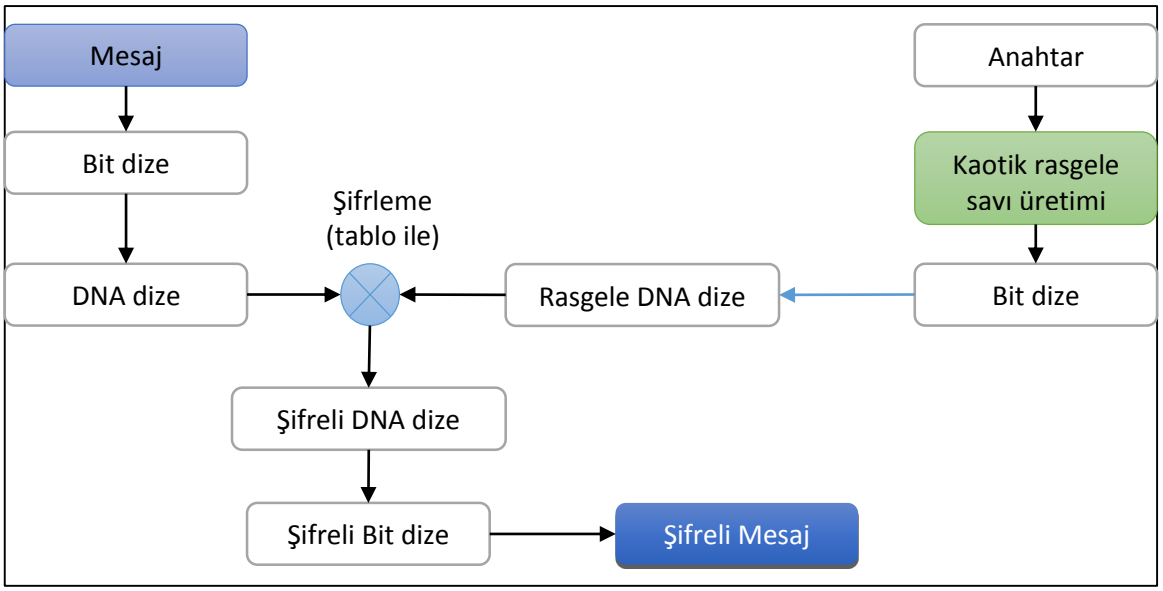
Bu metodun çıktısı, web sayfası içerisine bağlanması gereken *MathML* parser gerektirir. Örnek olarak, "(3*cos (x + 1)) / 2" ifadesinin ham (raw) ve işlenmiş formunun görünümü Tablo 10'da gösterilmektedir.

Tablo 10. "(3*cos (x + 1)) / 2" ifadesinin *MathML* örneği

Raw View	Rendered View
<math xmlns="http://www.w3.org/1998/Math/MathML"> <mfrac> <mrow> <mo>(</mo> <mn>3</mn> <mi>cos</mi> <mo>(</mo> <mi>x</mi> <mo>+</mo> <mn>1</mn> <mo>)</mo> <mo>)</mo> </mrow> <mn>2</mn> </mfrac> </math>	3cos(x+1)/2

2.3. Şifreleme Yöntemi

Dijital görüntüler internet üzerinden erişilebilir çok sayıda veriye sahip olabilirler. Bu bölümde, modele dayalı şifreleme yöntemi olarak adlandırılan kaotik sistemler aracılığıyla dijital bir görüntünün şifrelemesi için yeni bir yöntem sunulmaktadır. Temel fikir, şifrelenmiş bir dizinin, kaos işlevleri ve ana görüntüden üretilen bir yöntemle üretilen bir sözde rasgele verinin birleştirilmesiyle yaratılmasıdır. Yöntem, önerilen algoritmanın karmaşıklığını arttırmak için çoklu şifreleme tabloları kullanır. Tablolar görüntü ve rastgele iplik değerlerine göre dinamik olarak belirlenir. Bu özellik, kaotik sistemlerin özellikleriyle birleştiğinde, şifreleme algoritmasının güvenliğini ve verimliliğini artırır. Ayrıca, yüksek güvenlik seviyesine ulaştıran yöntem değerlendirme ve güvenlik değerlendirme deneylerinin sonuçlarını sunmaktadır. Şekil 50'de tezin genel yapısı gösterilmektedir. Burada anahtar ile kast edilen, 0 ile 1 arasında bir başlangıç değeridir. Kullanıcı anahtar yerine bir kelime girerse, ileride gösterilecek yöntem ile girilen karakter dizisinin, 0 ile 1 araında bir değere dönüştürülmesi gerekecektir.



Şekil 50. Şifreleme Genel Yapısı

2.3.1. DNA Desenleri

DNA, nükleotitlerden yapılan deoksiribonükleik asitin kısaltmasıdır. DNA'daki bilgi, dört tip nükleik asit, adenin (A), sitozin (C), guanin (G) ve timin (T) içeren bir kod olarak saklanır.

Gehani ve diğ. [96], geleneksel elektronik ortamlarla sınırlı olan DNA kullanarak görüntü şifrelemesi için OTP tabanlı bir yöntem önermişlerdir; Bununla birlikte, uygun medyanın DNA ve biyolojik yeteneklerini kullanarak muazzam bilgi taşımak için OTP'de bir anahtar olarak kabul edildi. Son zamanlarda, bazıları devasa veri transferinde kullanılmış olan DNA hesaplamalarına dayanan başka yöntemler de önerilmiştir [97, 98].

DNA yöntemini kullanarak görüntü şifreleme, görüntü bilgisini ondalık sistemden dörtlü bir alfabetik sisteme aktarabilir. Bunu yapmak için, önce, her bir pikselin sayısal değerleri 8 hane ile ikiliye dönüştürülür ve daha sonra her iki basamak bir harfe dönüştürülür. Bu dönüştürme prosedürü Denklem (3) olarak bir transfer fonksiyonu kullanılarak gerçekleştirilebilir:

$$f: x \to y\,; x \in \{0,1..255\} \; and \; y \in \{A,T,C,G\} \tag{3}$$

Denklem (3)'te, ikili değerin her 2 biti Denklem (4) ile verilen bir homomorfizm fonksiyonuna göre ikame edilebilir:

$$\begin{matrix} h(11) \to A & h(10) \to T \\ h(01) \to G & h(00) \to C \end{matrix} \tag{4}$$

Denklem (4)'ü kullanarak ve mesajin boyutlarının N'e eşit olduğunu varsayarak, DNA alfabesinden 4 * N boyutuna kadar bir dizi üretilir.

Çalışmada, *DNA* kalıpları ve çoğu uygulamada kullanılan standart *XOR* operatörü dışındaki farklı operatörleri kullanarak bir yöntem önerilmiştir. DNA kalıplarını ve farklı operatörleri uygulamak, önerilen yöntemin güvenliğini diğerlerinden daha ideal hale getirmiştir.

2.3.2. Tablo/Operatör Veritabanı

DNA bilişim alanında, bazı biyolojik ve cebirsel operatörler çıkarılmaktadır [99]. Bu işlemler, üretilen DNA dizisi üzerinde gerçekleştirilebilir. DNA iplikçikleri için (⊗) operasyonu Tablo 11'de çizilir.

Tablo 11. DNA iplikçikleri için özel veya operasyon (⊗)

⊗	A	T	G	C
A	C	G	T	A
T	G	C	A	T
G	T	A	C	G
C	A	T	G	C

Tablo 11, DNA düzeninin harita kurallarını, XOR işleyişinin uygun şekilde ve denklemin homomorfizma fonksiyonuna göre tanıtmaktadır. Denklem (4)'te transfer fonksiyonu Watson-Crick temel ayrıştırma kurallarına dayanılarak oluşturulmuştur [100]. Tablo 12, Watson-Cricks kurallarına göre sekiz olası transfer fonksiyonu biçimini göstermektedir.

Tablo 12. Watson-Cricks siparişine göre muhtemel transfer fonksiyonları

	Order1	Order2	Order3	Order4	Order5	Order6	Order7	Order8
A	00	00	11	11	10	01	10	01
T	11	11	00	00	01	10	01	10
G	10	01	10	01	00	00	11	11
C	01	10	01	10	11	11	00	00

Bu kurallar, XOR operatörü kullanılarak 8 farklı tablo ile sonuçlanabilir. Bu tür tablolara sahip olmak için Watson-Crick kurallarına dayanmayan diğer formları kullanmak da

mümkündür. Bu nedenle 2-bit çiftlerini DNA alfabesine aktarmak için 24 farklı fonksiyon düşünülmelidir. Elde edilen tablolar aynı özelliklere sahiptir:

- Son satır ve sütunlar (karşılık gelen 00'a eşdeğer) tabloların nötr üyeleridir
- Tabloların çapı nötr üyeye eşittir (00'a karşılık gelen harf).

Bu özellikler bu çalışmada kullanılmıştır. Ayrıca, tablo sayısını arttırmak için benzer özelliklere sahip diğer operatörler de kullanılabilir. Xingyuan Wang ve diğ. [101] toplama (+) operatörleri dahil etmişlerdir. Bu operatörler Tablo 13 ve Tablo 14'de özetlenmiştir.

Tablo 13. DNA dizisi için Toplama (+) işlemi

+	A	T	G	C
A	T	G	C	A
T	G	C	A	T
G	C	A	T	G
C	A	T	G	C

Denklem (4)'teki aynı fonksiyonlar Tablo 13 ve Tablo 14'ün tasarımında kullanılmıştır. Her operatör için 24 durumun bulunması, sistem için 48 farklı operatör tablosunun tasarlanabileceği anlamına gelir. Bu tablolar, önerilen sistemde daha fazla kullanılan operatör veritabanı olarak adlandırılan bir veritabanında numaralandırılır ve saklanır.

Tablo 14. Deoksiribonükleik asit sekansı için çıkarma (-) operasyonu

-	A	T	G	C
A	C	A	T	G
T	G	C	A	T
G	T	G	C	A
C	A	T	G	C

2.3.3. Önerilen Yöntem

Bu yöntem, önce operatörün veritabanıyla ilgili olan ve önceki bölümde açıklanan dört farklı aşamaya sahiptir. Daha sonra, ikinci aşama olarak, piksellerin sayısal değerleri bir DNA dizisine dönüştürülür. Her bir piksel 8 bitlik ikili biçime dönüştürülür ve daha sonra bit düzlemlerinin her biri, Denklem (4)'e göre DNA kodlama kuralı uygulanarak DNA sekansına

çıkarılır ve dönüştürülür. Örneğin, pikselin sayısal değeri 64 olduğunda, ikili biçim numarası Denklem (4) kullanılarak 10100100, TTGC gibi bir DNA dizisi ile sonuçlanır.

Görüntülerin ana özelliklerinden biri, en yakın veya komşu resim elemanının yüksek korelasyon faktörüdür. En yakın resim elemanının etkileşim faktörünün düşürülmesi durumunda, pikselin konumu, kendi değerinden etkilenebilir. Denklem (5), P etkisinin resim öğesinin sayısal değerine atıfta bulunduğunu ve her pikselin indeksini ifade ettiğini varsayarak bu etkiyi gösterir.

$$(p_i + i) \bmod 256 \rightarrow P_i' \rightarrow P_{DNA} \tag{5}$$

Bu nedenle, düz görüntüden 4*N büyüklüğünde bir DNA dizisi elde edilir ve bu DNA_P olarak gösterilir. 3. aşamada, 4*N boyutunda rastgele bir DNA ipliği üretilir. Bu yazıda, rastgele diziyi üretmek için 2 boyutlu bir lojistik harita kullanılmıştır.

2 boyutlu lojistik harita fonksiyonunun özellikleriyle ilgili olarak, x_0 ve y_0 başlangıç değeri 0 ile 1 arasında bir sayıdır. Fonksiyon tarafından üretilen diğer numaralar başlangıç değerine, x_0, y_0 ve diğer parametrelere çok hassastır. .

Başlangıç değeri x_0 ve y_0 şu şekilde hesaplanır:

$$\begin{cases} x_0 = \frac{\sum_{i=0}^{n-1}\sum_{j=0}^{7} k_{i,j} * 2^{i*8+j}}{2^{n*8}} \\ y_0 = \frac{\sum_{i=n-1}^{0}\sum_{j=0}^{7} k_{i,j} * 2^{i*8+j}}{2^{n*8}} \end{cases} \tag{6}$$

Önerilen yöntem, üretilen dizinin büyüklüğüne göre rasgele bir dizi gerektirir. Bu nedenle, lojistik haritalama kullanılarak, x_0 ve y_0 başlangıç değeri ile 0 ve 3 arasında 4*N sözde-rasgele sayı üretilir. Bu sayılar, faz ikiyle aynı prosedür kullanılarak DNA dizilerine dönüştürülür. Bu yazıda 2 boyutlu lojistik harita üzerinden rassal sayılar üretmek için Denklem (2)'de β_0 ve β_1 sırasıyla 0.17 ve 0.14 olarak belirlenmiştir. Ayrıca, anahtardan verilen diğer parametreler α_1 ve α_2'dir. 2-D lojistik fonksiyonunun sonucu, 0 ve 1 arasında iki sayı dizi üretmektir. Bu, Denklem (7) kullanılarak bir dizi ikili sayıya dönüştürülebilir. Bu iki görüntünün bitlerini birleştirerek, rastgele bir iplik oluşturulabilir.

$$R \leftarrow (x_i * C)\%\, 2 \tag{7}$$

Qian ve diğ. [102], rasgele sayıları belirli bir aralıkta bir sayıya dönüştürmek için bir dönüştürme faktörünün kullanılmasını önermişlerdir, yani (0, 255); bu durumda Denklem (7)'de C = 1000000 olarak ayarlanmıştır. Böylece, üretilen rasgele sayılar daha düzgün dağılımda olacaktır. Bu aşamada üretilen rastgele iplikçik, şifrelemede ve şifrelerin çözülmesinde önemli bir rol oynar ve DNA_R olarak adlandırılır.

Liste 8. Şifreleme aşamaları

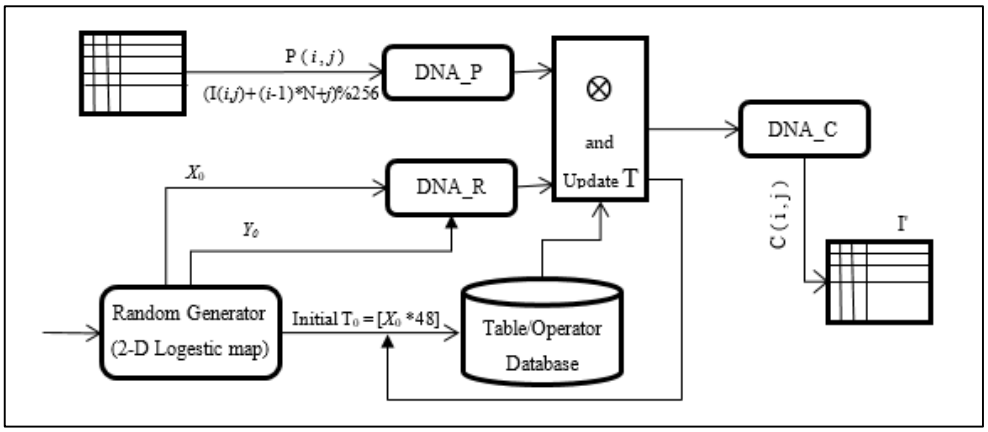
```
Step 1: ind = int (x0 * (Table number) )
// a random number in the vicinity of 0 and table number
Step 2: for each element in DNA_P
Step 2.1 : DNA_Ci = DNA_Pi ⊗ind DNA_Ri
Step 2.2: if DNA_Ci is a neutral member in ind Table.
Step 2.2.1 : Increase ind by 1
Step 2.2.2 : if ind = Table number then ind =0
```

Liste 9. Deşifreleme aşamaları

```
Step 1: ind = int (x0 * (Table number) )
// a random number in the vicinity of 0 and table number
Step 2: for each element in DNA_P
Step 2.1 : DNA_Ci = DNA_Pi ⊗ind DNA_Ri
Step 2.2 : if DNA_Pi = DNA_Ri then
Step 2.2.1 : Increase ind by 1
Step 2.2.2 : if ind = Table number then ind =0
```

DNA_P ve operatörlerin veritabanı kullanılarak şifrelenmiş bir DNA_C dizisi üretilebilir. Şifreleme ve deşifrelemede kilit nokta, operatör tablolarını nasıl takip edeceğidir. Liste 8 ve Liste 9 şifreleme ve deşifreleme algoritmalarını özetlemektedir

Önerilen aşamalardan geçerek, bir görüntü DNA dizilerinde şifrelenebilir. Bu sekanslar biyolojik olarak nakledilebilir veya ondalık eşdeğerlere dönüştürülebilir ve şifrelenmiş bir görüntü olarak kullanılabilir. Şekil 51, önerilen yöntemin genel yapısını göstermektedir.



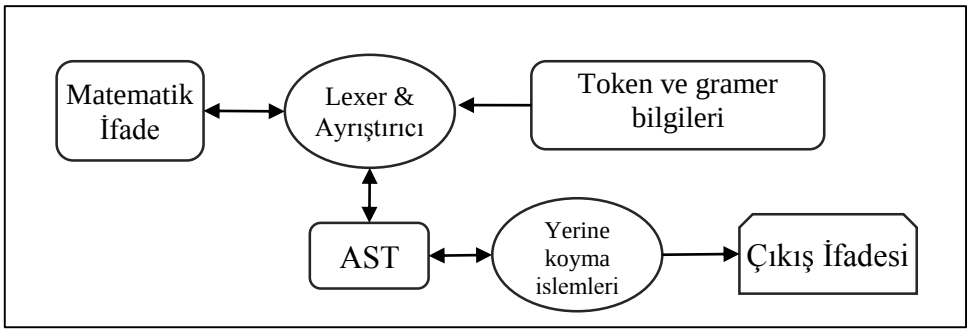
Şekil 51. Önerilen algoritmanın genel yapısı

2.4. Matematik İfadelere Dayalı Veri Gizleme

Bilgi saklaması son zamanlarda çok dikkat çekmiştir. Steganografi için çeşitli yöntemler geliştirilmiştir ve aynı zamanda gizli verilerden kuşkulanmak için uygun steganaliz tasarlanmıştır. Ancak, steganaliz analizlerine dikkat edilmemesi nedeniyle yeni tipte bir kapağa getirilen bir yaklaşım daha az şüpheli olabilir. Bu bölümde, bir mesajı matematiksel bir ifadeye dönüştürebilecek bir yöntem önerilmektedir. Oluşturulan matematiksel ifade, mesajı güvenli bir metinle birlikte iletmek için bir kapak olarak kullanılabilir. Uygun bir stokastik dilbilgisi kullanarak önerilen yöntem, belirli bir metne dayalı bir matematiksel ifade oluşturma yeteneğine sahiptir. Dilbilgisi kurallarının olasılığı, taşıyıcı metne göre ve matematiksel ifadelerin türüne göre belirlenebilir.

2.4.1. Yerine Koyma Yöntemi

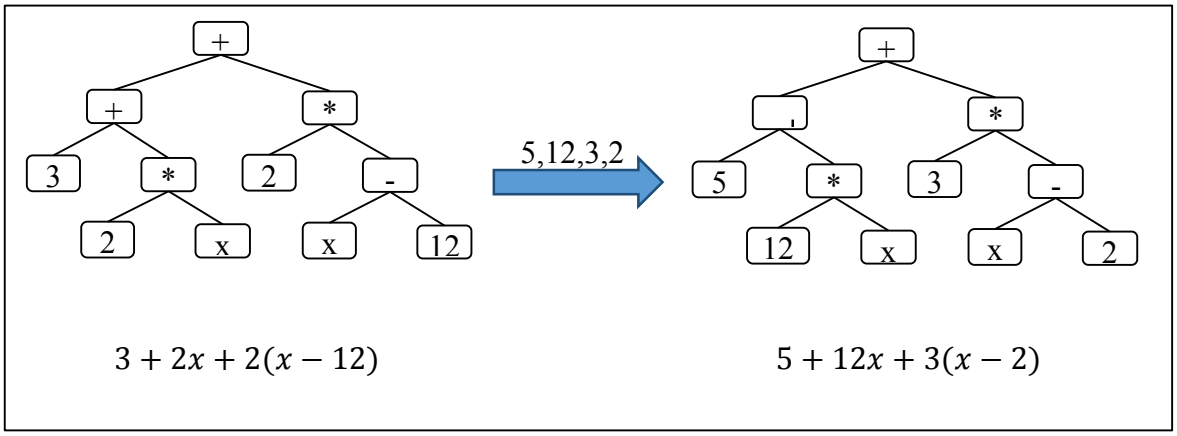
Veri gizleme yöntemlerinden ilk akla gelen yerine koyma olabilir. Bu yaklaşımda mesaj verileri bir ifadenin sayı değerleri veya işlevler içine gömüleblir. Gömme işlemleri çeşitli metin yöntemlerine benzer yöntemler ile gerçekleşebilir. Metin ile matematiksel ifadeler arasındaki fark aslında metinlerin karakterleri ardışık ve matematiksel ifadelerin özel yapıları olmasıdır. Dolaysıyla matematiksel ifadelerin üzerinde işlem yapılmadan önce, AST ağacına taşınması gerekir. Şekil 52 matematiksel ifadelerin üzerinde veri gizleme metodolojisini göstermektedir.



Şekil 52. Yerine Koyma Yönteminin genel modeli

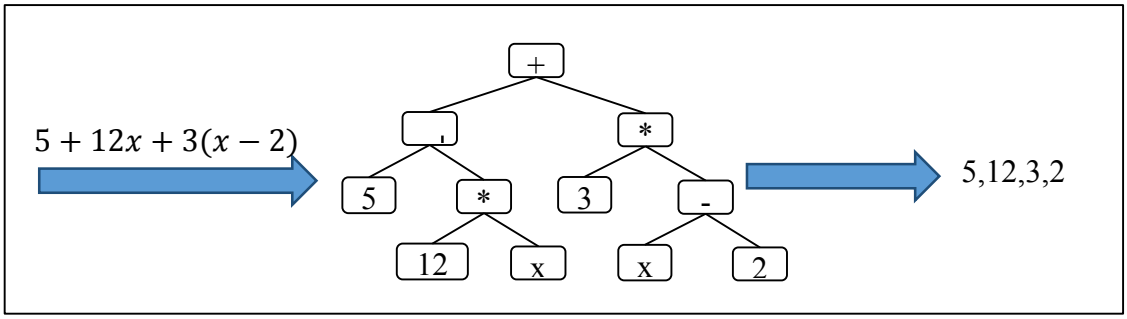
Şekil 52'de göründüğü gibi metematiksel ifadeler belirli kurallar üzerinden AST ağacına dönüşülüp, üzerlerinde farklı yerine koyma yöntemleri uygulanır. Bunun için çeşitli yerine koyma yöntemler kullanılabilir. Aşağıda bir kaç örnek verilmiştir:

Örnek 1: Bu yöntemde matematiksel ifade AST ağacına taşındıktan sonra, AST ağacı pre-order, in-order veya post-order şeklinde okunur ve rasladığı her sayı, mesaj verileri ile yerdeğiştirilir. Şekil 53 bu yöntemi bir örnek üzerinden göstermektedir.



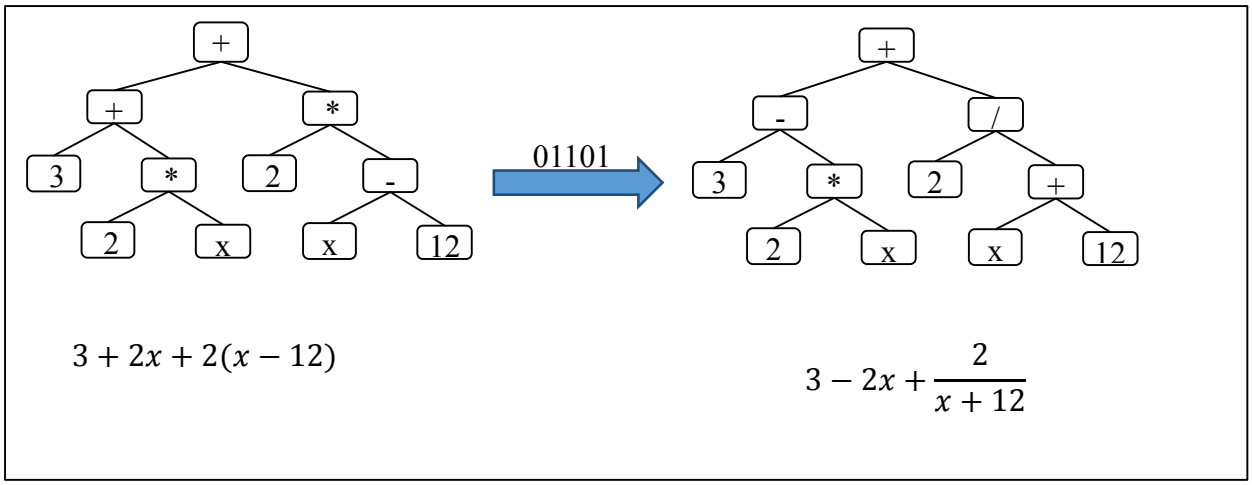
Şekil 53. In-order trace ile yerine koyma yönteminden örnek

Şekil 53'deki örnekte mesaj verileri ile ağactaki sayılar eşit olduğundan her hangi bir sorun ile karşılaşılmaz. Eğer ağaçtaki sayılar mesaj verilerinden daha fazla olsaydı, ağacın geriye kalan sayıları extract aşamasında mesaj verilerine eklenebilir. Bu sorunu gidermek için mesaj verilerinin sonuna -1 (veya herhangi eksi değer) eklenir. Extract aşamasında bu veri okunduktan sonra mesaj verilerinin sonu olduğu anlaşılır. Şekil 54, önceki örneğin extraction (veri çıkarma) işlemini göstermektedir.



Şekil 54. In-order trace ile sabstitution yöntemindeki örneğin extraction işlemleri

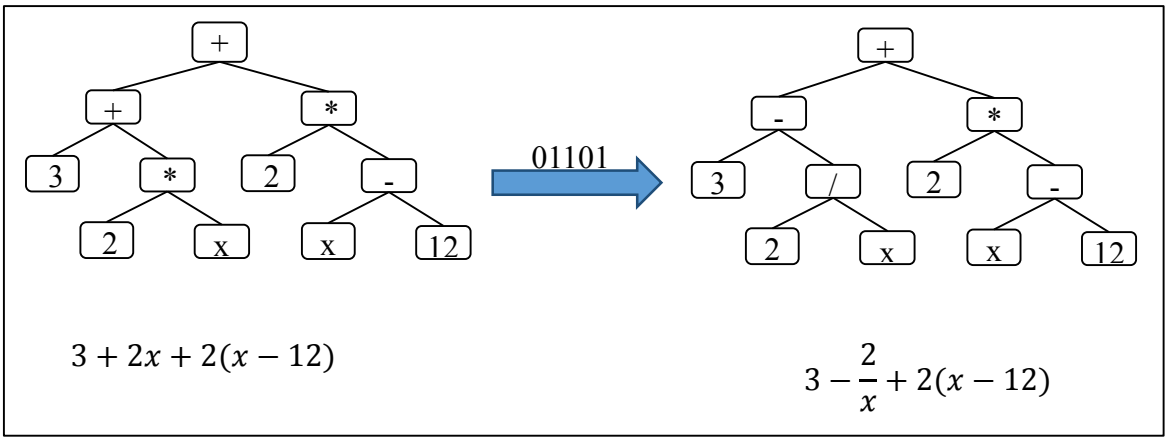
Örnek 2: Bu yöntemde AST üzerinde bulunan ifade belirli bir kuralla taranır, örneğin pre-order veya satır satır soldan sağa) ve mesaj bitlerine göre 0 ise rasladığı işlev değişmez, 1 ise + işlevi – ile ve * işlevi / ile yerdeğiştirilir. Şekil 55 bu yöntemi bir örnek üzerinden anlatmaktadır.



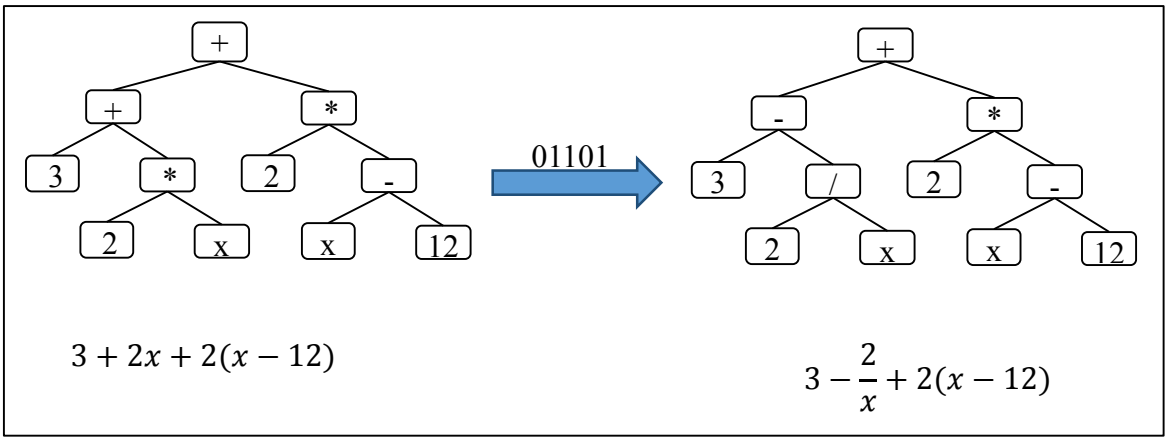
Şekil 55. İşlev değiştirerek mesaj gömme örneği

Şekil 55'deki örnekte AST satırları soldan sağa sıra ile taranmıştır. Bu yöntem ile gömulen mesajları cıkarması için karşi tarafde orginal ifadenin bulunması gerekiyor. Extraction işlemi yaparken, her iki ifadenin AST ağacı üretilir, değişilen işlevler 1, değişmeyen işlevler ise 0 olarak kayd edilir. Bitlerin sırası ise TRACE yöntemine tabi tutulur. Karşi tarafde ifadenin olma gerekçesi ortadan karksın diye aşağıdakı yöntemi kullanabiliriz.

Örnek 3: Bu yöntem bir önceki yönteme benzerdir, ancak burada + ve * işlevi 0, - ve / işlevi ise 1 anlamında kullanılır. Bu yöntemi kullanarak karşı tarafta orijinal ifadenin olması gerekmez. Yalnız işlevlere karşı gelen bitleri tarama yöntemine göre sıralanırsa mesaj elde edilebilir.



Şekil 56 bu yöntemi bir örnek üzerinden göstermektedir.



Şekil 56. İşlev değiştirerek diğer mesaj gömme örneği

Şekil 56'daki örnekte AST ağacı pre-order yöntemi ile taranmıştır.

2.4.2. Kapak Üretim Yöntemi

İnsan evrim tarihinde matematik önemli bir kökene sahiptir. Doğru komutlar kullanıldığında bilgisayarlar problem çözmede çok iyidir. Sembolik hesaplama veya cebirsel hesaplama, matematiksel ifadelerin geliştirilmesi ve değerlendirilmesi anlamına gelir. Bilgisayar Cebir Sistemleri (CAS) veya sembol manipülasyon sistemleri olarak adlandırılan sembolik hesaplamaları yapabilen bilgisayar uygulamalarıdır. CAS sistemlerinin artan kullanımı, çeşitli alanlarda bilgisayar sistemlerinin rolünü genişletmiştir. Örneğin, bilgisayar cebiri, sayısal programlarda gerekli olan formüllerin tasarlanması ve denenmesinde önemli bir role sahiptir. Hassas verileri gizlemek için CAS'da önerilen teknikleri kullanabilir ve steganografi sistemleri için uygun bir platform sağlayabiliriz.

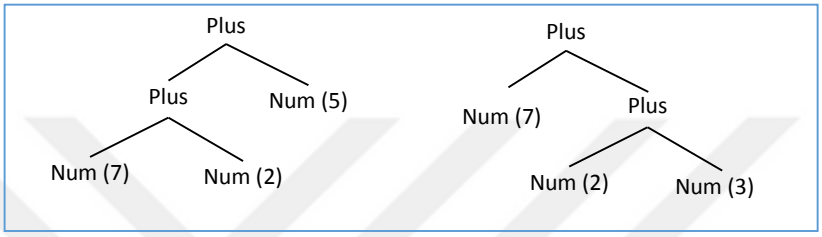
Bu tezde, matematiksel ifadelerde verilerin steganografisi için yeni bir kapak üretim yöntemi önerilmiştir. Matematiksel ifadeler, özellikleri göz önüne alındığında, bilgi gizlemek için iyi bir yer olabilir. Gerçek gibi görünen metinleri üretmek için çaba sarf edilmiştir [54]. Aynı bakış açısı ile belirsiz olmayan matematiksel ifadeler üretmenin yollarını önermek mümkündür.

Genel bilgiler bölümünde, Kapak Üretimi Steganografi sistemlerine yönelik yapılan işlemler anlatılmaktadır. Bu bölümün devamında matematiksel ifadeler ve önerilen yönteme uygun uygulanabilen işlemler anlatılmaktadır. Bu işlemler matematik cümleciklerini üretmeyi, ayrıştırmayı ve görüntülemeyi içerir. Ayrıca, bu bölümde bir mesajın yerleştirilmesi ve geri

elde edilmesi için yeni bir yöntem önerilecektir. Önerilen yöntemin değerlendirilmesi bulgular ve sonuçlar bölümünde yapılacaktır.

2.4.2.1. Dilbilgisi Dönüşümü

Önceki bölümde tanımlanan dilbilgisi geneldir ve herhangi bir matematik ifadesi oluşturabilir. Bazı durumlarda, belirli bir ifade için bu dilbilgisini kullanarak birden fazla ifade ağacı ortaya çıkabilmektedir. Basit bir örnek olarak, Şekil 57'de "7 + 2 + 5" için iki farklı ifade ağacı verilmiştir.



Şekil 57. "7 + 2 + 5" ifadesi için iki farklı AST

Bu durum, bazı sistemlerde sorun olmasa bile, bu çalışmada önerilen yöntem için büyük sorunlara neden olabilir. Steganografide bu, gömülü mesajdan farklı mesajın çıkarılmasına yol açar. Bu nedenle, dilbilgisini uygun şekilde geliştirmeye ihtiyaç vardır.

Liste 10. Matematiksel ifade için değiştirilmiş dilbilgisi

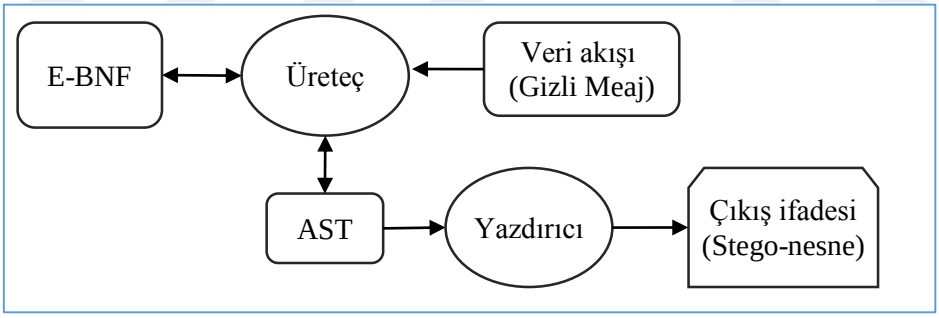
```
0. < E >::= '(' < E > ')'
1.  < E >::= < E > '+' < E >
2.  < E >::= < E > '-' < E >
3.  < E >::= < E > '*' < E >
4.  < E >::= < E > '/' < E >
5.  < E <::= < E > '^' < E >
6.  < E >::= 'Sin (' < E > ')'
7.  < E >::= 'Cos (' < E > ')'
8.  < E >::= 'Tan (' < E > ')'
9.  < E >::= 'Cot (' < E > ')'
10. < E >::= 'Log (' < E > ')'
11. < E >::= 'Exp (' < E > ')'
12. < E >::= 'Sqrt (' < E > ')'
13. < E >::= 'Abs (' < E > ')`
14. < E >::='Lim (' < E > '),(' < E > ')'
15. < E >::='Fact(' < E > ')'
     < E >::= < C >
    < C > ::= < Var > | < Num > < Var >
    < Var >::='x'
    < Num >::= '-' ? < Digit > + ('.' < Digit > +)?
    < Digit >::=[' 0'-'9']
```

2.4.2.2. Matematiksel İfadelerin Oluşturulması

Önerilen yönteme göre, dilbilgisini kullanarak matematiksel ifadeler oluşturmak için, ifade oluşturma cümleleri, numaralandırılmış kuralların mesaj verilerine göre seçilmesi gerekir. Verilerin bir kısmını matematiksel ifadelerde katsayı olarak kabul edilen sayılara dahil etmek de mümkündür. Yöntem kapsamında kullanılan matematiksel ifadeler üretme metodolojisi aşağıda sunulmuştur.

2.4.2.3. Veri Gizleme Yöntemi

İkili ağaç, cebirsel operatörlerin ve tek parametre işlevlerinin dahil edilmesini destekleyen uygun bir veri yapısıdır. Operatör önceliğini temsil etmenin basit bir yolunu sunar. Ayrıca, ağaç yapısını belge formatlayıcılarının gereksinimleri için diğer veri formatlarına dönüştürmek kolaydır. Diğer bir avantaj, gelişimsel gramerlerin geliştirilmesidir, çünkü ağacın her bir düğümü, bir dilbilgisi kuralının oluşturduğu ebeveyn ve çocuk arasında tekrarlayan bir çağırma önerir. Şekil 58, ağaçları kullanarak bir matematiksel ifade oluşturmak için gerekli matematiksel bileşenlerin bir blok diyagramını göstermektedir.



Şekil 58. Ağaç tabanlı ifadelerin üretilmesi için metodolojik bileşenler

Şekil 58'de, "Üreteç" ilk önce bir dilbilgisinin başlangıç kuralını AST'ye yerleştirir ve daha sonra giriş kurallarına göre giriş verisine göre bu kuraldaki diğer kurallarla değiştirmeden terminal olmayanları tekrar tekrar genişletir. Liste 10'de sunulan kuralların, önerilen yöntemin konusu olmadığı için operatörlerin önceliğini göz önünde bulundurmadığını unutmayın. Liste 11, Şekil58'deki Üreteç bileşeni tarafından gerçekleştirilen metodolojinin ana algoritmasını gösterir.

Liste 11. İfade üreten algoritma

```
AST ← Start symbol E
While exists any data in input stream
   N     ← Get a data from stream
   expr  ← Select N^th rule
   AST   ← Replace E with expr in AST
In-order traverse the AST and print mathematical
expression
```

Liste 11'de sunulan gramer temel problemi, sol özyinelemeli kuralların varlığıdır. Bu kural türü, oluşturulan matematiksel ifadelerde belirsizlik yaratır. Bu nedenle, kullanılan dilbilgisi kuralların bu özelliğe sahip olmadığı bir şekilde geliştirilmelidir. Burada tartışılan terimlerin üretilmesinde, operatörlerin önceliğini gözlemlemeye gerek yoktur. Çünkü bu çalışmadaki ana amaç, ifadenin üretilmesi olup değerlendirilmesine odaklanılmamıştır. Ek olarak, terminal olmayanlar için yazılan kurallar, bu sayılara dayanarak algoritmanın yürütülmesi sırasında kullanılmak üzere numaralandırılmalıdır. Buna göre, istenen dilbilgisi Liste 10'da gösterildiği gibi geliştirilmiştir. Aşağıdaki örnek Liste 11'deki algoritmayı kullanarak mesaj üzerinden ifade üretmektedir.

Örnek: Mesaj ='Example'

1. Example → 69, 120, 96, 109, 112, 108, 101 → 01000101, 01111000, 01100000, 01101101, 01110000, 01101100, 01100101 → 4,5,7,8,6,0, 6,13, 2,0, 6,12,6,5
2. < E > →**(**4.kural E için**) < E$_1$ > / < E$_2$ >→ (**5. Kural E_1 yerine**) < E$_3$ > ^ < E$_4$ >** / < E_2 >

→ **(**7. Kural E_2 yerine**)** < E_3 > ^ < E_4 > / **Cos(< E$_5$ >)**

→ **(**8. Kural E_3 yerine**) Tan(< E$_6$ >)** ^ < E_4 > / Cos(< E_5 >)

→ **(**6. Kural E_4 yerine**)** Tan(< E_6 >) ^ **Sin(< E$_7$ >)** / Cos(< E_5 >)

→ **(**0. Kural E_5 yerine**)** Tan(< E_6 >) ^ Sin(< E_7 >) / Cos**((< E$_8$ >))**

→ **(**6. Kural E_6 yerine**)** Tan(**Sin(< E$_9$ >)**) ^ Sin(< E_7 >) / Cos((< E_8 >))

→ **(**13. Kural E_7 yerine**)** Tan(Sin(< E_9 >)) ^ Sin(**Abs(< E$_{10}$ >)**) / Cos((< E_8 >))

→ **(**2. Kural**)** Tan(Sin(< E_9 >)) ^ Sin(Abs(< E_{10} >)) / Cos((**< E$_{11}$ > - < E$_{12}$ >**))

→ **(**0. Kural**)** Tan(Sin(**(< E$_{13}$ >)**)) ^ Sin(Abs(< E_{10} >)) / Cos((< E_{11} > - < E_{12} >))

→ **(**6. Kural**)** Tan(Sin((< E_{13} >))) ^ Sin(Abs(**< E$_{14}$ > / < E$_{15}$ >**)) / Cos((< E_{11} > - < E_{12} >))

→ **(**12.**)** Tan(Sin((< E_{13} >))) ^ Sin(Abs(< E_{14} > / < E_{15} >)) / Cos((**Sqrt(< E$_{16}$ >)** - < E_{12} >))

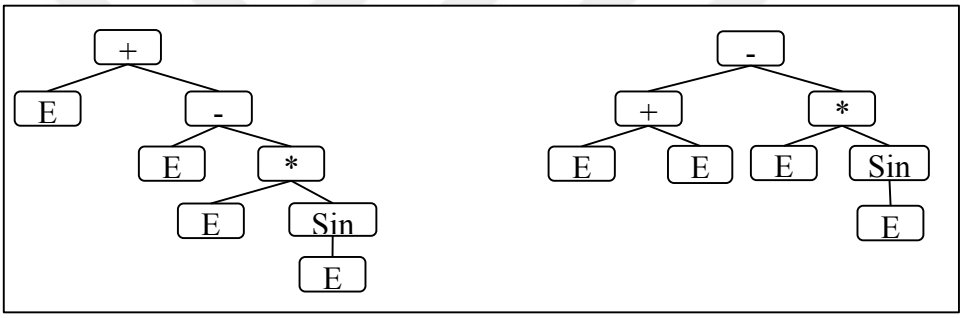
→ **(**6.**)** Tan(Sin((< E_{13} >))) ^ Sin(Abs(< E_{14} > / < E_{15} >)) / Cos((Sqrt(< E_{16} >) - **< E$_{17}$ >/ < E$_{18}$ >**))

→ **(**5.**)** Tan(Sin((**< E$_{19}$ >^ < E$_{20}$ >**))) ^ Sin(Abs(< E_{14} > / < E_{15} >)) / Cos((Sqrt(< E_{16} >) - < E_{17} >/ < E_{18} >))

Üretilen ifade en son kural uygulandıktan sonra tüm < E > için rasgele olarak gramerin numarasız kurallarını kullanarak tamamlanır. Bu işlemlerle aşağıda verilen ifadenin üretimi yapılabilir.

$$\frac{tan(sin(x^{3x})^{sin\left|\frac{12}{x}\right|})}{cos\left(|4x| - \frac{x}{3}\right)}$$

Yukarıdaki örnekte göründüğü gibi üretilen ifadeler uygundur. Ancak bu ifadelerin sözdizimleri açısından istenmeyen durumlarla karşılaşılabilir. Üretilme aşamasındaki AST ile daha sonra oluşan AST (extraction aşamasında) aynı olmayabilir. Bu farklılık veri çıkarmada yalnış mesaj elde edilmesine neden olur. Şekil59'de E + E – E * Sin(E) ifadesinin farklı AST leri gösterilmiştir. Bu ağacların biri üretim, diğeri extraction aşamalarında oluşursa farklı mesaj elde edilir.



Şekil 59. E + E – E * Sin(E) ifadesinin farklı ağaçları

Şekil 59'da gösterilen sorun grameri değiştirerek düzeltilebilir. O yüzden Liste 10'daki gramer uygun şekilde değiştirilmelidir. Liste 12'de gramerin geliştirilmiş hali sunulmaktadır.

Önceki örnek ile Liste 12'deki grameri kullanılarak aşağıda verilen ifade üretilir.

Örnek : Mesaj ='Example'

1. Example → 69, 120, 96, 109, 112, 108, 101 → 01000101, 01111000, 01100000, 01101101, 01110000, 01101100, 01100101 → 4,5,7,8,6,0, 6,13, 2,0, 6,12,6,5
2. < E > →(4.Kural) < C > / < E > → (5.Kural) < C > / < C > ^ < E >
   → (7.Kural) < C > / < C > ^ Cos(< E >) → (8) < C > / < C > ^ Cos(Tan(< E >))
   → (6.Kural) < C > / < C > ^ Cos(Tan(Sin(< E >)))
   → (0.Kural) < C > / < C > ^ Cos(Tan(Sin((< E >))))
   → (6.Kural) < C > / < C > ^ Cos(Tan(Sin((Sin(< E >)))))
   → (13.Kural) < C > / < C > ^ Cos(Tan(Sin((Sin(Sin(Abs(< E10 >))))))

→ (2.Kural) < C > / < C > ^ Cos(Tan(Sin((Sin(Sin(Abs(< C > - < E >))))))

→ (0.Kural) < C > / < C > ^ Cos(Tan(Sin((Sin(Sin(Abs(< C > - (< E >)))))))

→ (6.Kural) < C > / < C > ^ Cos(Tan(Sin((Sin(Sin(Abs(< C > - (< C > / < E >)))))))

→ (12.) < C > / < C > ^ Cos(Tan(Sin((Sin(Sin(Abs(< C > - (< C > / Sqrt(< E >)))))))

→ (6.) < C > / < C > ^ Cos(Tan(Sin((Sin(Sin(Abs(< C > - (< C > / Sqrt(< C >/ < E >)))))))

→ (5.) < C > / < C > ^ Cos(Tan(Sin((Sin(Sin(Abs(< C > - (< C > / Sqrt(< C >/ < C >^ < E >)))))))

Üretilen ifade en son kural uygulandıktan sonra tüm < C > ve < E > için rasgele olarak gramerin numarasız kuralarını kullanarak tamamlanır. Bu işlemlerle aşağıda verilen ifadenin üretimi yapılabilir.

$$\frac{3x}{5^{\cos(\tan(\sin(\sin(\sin(\left|x-\left\langle\frac{12}{\sqrt{\frac{5x}{4^5}}}\right\rangle\right|)))))}}$$

Liste 12. Matematiksel ifade için değiştirilmiş dilbilgisi

```
0. < E >::= '(' < E > ')'
1. < E >::= < C > '+' < E >
2. < E >::= < C > '-' < E >
3. < E >::= < C > '*' < E >
4. < E >::= < C > '/' < E >
5. < E <::= < C > '^' < E >
6. < E >::= 'Sin (' < E > ')'
7. < E >::= 'Cos (' < E > ')'
8. < E >::= 'Tan (' < E > ')'
9. < E >::= 'Cot (' < E > ')'
10. < E >::= 'Log (' < E > ')'
11. < E >::= 'Exp (' < E > ')'
12. < E >::= 'Sqrt (' < E > ')'
13. < E >::= 'Abs (' < E > ')'
14. < E >::='Lim (' < E > '),(' < E > ')'
15. < E >::='Fact(' < E > ')'
  < E > ::= < Var > | < Num > < Var >
  < Var >::='x'
  < Num >::= '-' ? < Digit > + ('.' < Digit > +)?
  < Digit >::=[' 0'-'9']
```

Örnekte göründüğü gibi üretilen ifadeler uygun görünmemektedir. Bunun asıl nedeni, gramerin her zaman işlevin sağ tarafından ilerlemesidir. Bu sorunun giderilmesi için Liste 13'deki gramer geliştirilmiştir.

Liste 13. Matematiksel ifade için geliştirilmiş dilbilgisi

```
0. < E >::= '(' < E > ')'
1. < E >::= < T > '+' < E >
2. < E >::= < T > '-' < E >
3. < E >::= < F > '*' < E >
4. < E >::= < F > '/' < E >
5. < E <::= < E > '^' < E >
6. < E >::= 'Sin (' < E > ')'
7. < E >::= 'Cos (' < E > ')'
8. < E >::= 'Tan (' < E > ')'
9. < E >::= 'Cot (' < E > ')'
10. < E >::= 'Log (' < E > ')'
11. < E >::= 'Exp (' < E > ')'
12. < E >::= 'Sqrt (' < E > ')'
13. < E >::= 'Abs (' < E > ')'
14. < E >::='Lim (' < C > '),(' < E > ')'
15. < E >::='Fact(' < C > ')'

0. < T >::= < F > '*' < T >
1. < T >::= < F > '/' < T >
2. < T <::= < T> '^' < T >
3. < T >::= 'Sin (' < T > ')'
4. < T >::= 'Cos (' < T > ')'
5. < T >::= 'Tan (' < T > ')'
6. < T >::= 'Abs (' < T > ')'
7. < T >::= '(' < T > ')'

0. < F >::= 'Sin (' < F > ')'
1. < F >::= 'Cos (' < F > ')'
2. < F >::= 'Tan (' < F > ')'
3. < E >::= 'Cot (' < F > ')'
4. < E >::= 'Log (' < F > ')'
5. < E >::= 'Sqrt (' < F > ')'
6. < F >::= 'Abs (' < F > ')'
7. < F >::= '(' < F > ')'
< E > ::= < Var > | < Num > < Var >
< E > ::= < Var > | < Num > < Var >
< E > ::= < Var > | < Num > < Var >
< Var >::='x'
< Num >::= '-' ? < Digit > + ('. ' < Digit > +)?
< Digit >::=[' 0'-'9']
```

Önceki örnek ile Liste 13'deki gramere göre aşağıda verilen ifade üretilebilmektedir:

Örnek : Mesaj ='Example'

1. Example → 69, 120, 96, 109, 112, 108, 101
2. → 01000101011110000110000001101101011100000110110001100**010**
3. < E > →**(E nin 3. kuralı) < T_1 > - < E_2 >**

   01000101011110000110000001101101011100000110110001**100**010

   → **(T nin 6. kuralı) Abs(< T_3 >)** - < E_2 >

   01000101011110000110000001101101011100000110**1000**1100010

   → **(E nin 8. kuralı)** Abs(< T_3 >) - **Tan(< E_4 >)**

   01000101011110000110000001101101011100000**101**10001100010

   → **(T nin 5.kuralı)** Abs(**Cos(< T_5 >)**) - Tan(< E_4 >)

   01000101011110000110000001101101011100**0001**10110001100010

   → **(E nin 1.kuralı)** Abs(Cos(< T_5 >)) - Tan(**< T_6 > + < E_7 >**)

   01000101011110000110000001101101011**100**0001101100011000 10

   → **(T nin 4.kuralı)** Abs(Cos(**Tan(< T_8 >)**)) - Tan(< T_6 > + < E_7 >)

   ....

   01000101011110000110000001101101011100000110110001100010

   → **(F in 1. kuralı)** Abs(Cos(Tan(Abs(Sin(< F_{20} >) / < **F_{21}** > * < **T_{22}** >)))) - Tan(Sin(Log(< F_{17} >) *(< T_{18} >)) + Abs(Tan(Log(< E_{19} >))))

Üretilen ifade en son kural uygulandıktan sonra tüm < E >, < T > ve < F > için rasgele olarak gramerin numarasız kurallarını kullanarak tamamlanır. Bu işlemlerle aşağıda verilen ifadenin üretimi yapılabilir.

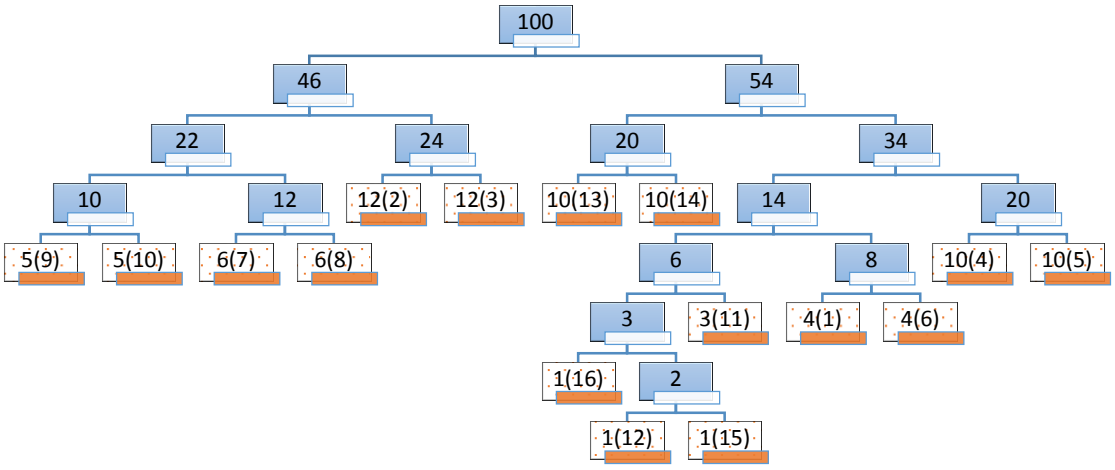
$$\left|cos\left(tan\left(\left|\frac{sin\,3x}{4*x}\right|\right)\right)\right| \text{-} tan(sin(log(12) * (5x) + |tan(log(2x))|))$$

Liste 13, Liste 10'da sunulan yöntem kullanılarak üretilen matematiksel ifadeler, eşit olarak dağıtılmış ve giriş verilerine dayandırılmıştır. Kaynaklarda yaygın olarak kullanılan matematiksel ifadeler farklı dağılımlara sahiptir. Örneğin, üretilen ifadelerde *Cot* fonksiyonu genellikle daha fazla bulunmaktadır. Stokastik dilbilgisi, çağrılan kuralların durumunu kontrol etmek için kullanılabilir. Algoritmadaki her kural için bir başlatma olasılığını başlatma adımları, ifade adımlarıyla kontrol edilir, daha gerçekçi matematiksel ifadeler oluşturulabilir. Liste 14'te, Liste 10'nin dilbilgisi varsayımsal olasılıklarla temsil edilir.

Liste 14. Matematiksel anlatım için her bir üretimin yanında olasılıklarla stokastik bir dilbilgisi

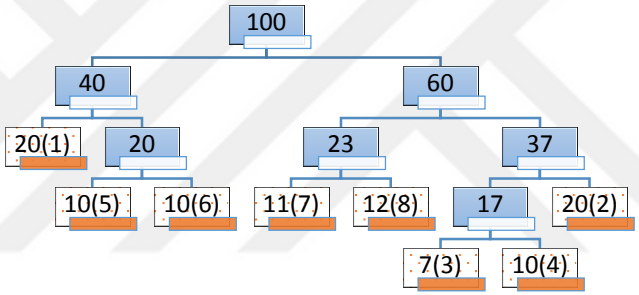
```
0. < E >::= '(' < E > ')'                          0.04
1. < E >::= < T > '+' < E >                        0.12
2. < E >::= < T > '-' < E >                        0.12
3. < E >::= < F > '*' < E >                        0.10
4. < E >::= < F > '/' < E >                        0.10
5. < E <::= < E > '^' < E >                        0.04
6. < E >::= 'Sin (' < E > ')'                      0.06
7. < E >::= 'Cos (' < E > ')'                      0.06
8. < E >::= 'Tan (' < E > ')'                      0.05
9. < E >::= 'Cot (' < E > ')'                      0.05
10.< E >::= 'Log (' < E > ')'                      0.03
11.< E >::= 'Exp (' < E > ')'                      0.01
12.< E >::= 'Sqrt (' < E > ')'                     0.10
13.< E >::= 'Abs (' < E > ')'                      0.10
14.< E >::='Lim (' < C > '),(' < E > ')'           0.01
15.< E >::='Fact(' < C > ')'                       0.01

0. < T >::= < F > '*' < T >                        0.20
1. < T >::= < F > '/' < T >                        0.20
2. < T <::= < T> '^' < T >                         0.07
3. < T >::= 'Sin (' < T > ')'                      0.10
4. < T >::= 'Cos (' < T > ')'                      0.10
5. < T >::= 'Tan (' < T > ')'                      0.10
6. < T >::= 'Abs (' < T > ')'                      0.11
7. < T >::= '(' < T > ')'                          0.12

0. < F >::= 'Sin (' < F > ')'                      0.14
1. < F >::= 'Cos (' < F > ')'                      0.14
2. < F >::= 'Tan (' < F > ')'                      0.12
3. < E >::= 'Cot (' < F > ')'                      0.12
4. < E >::= 'Log (' < F > ')'                      0.05
5. < E >::= 'Sqrt (' < F > ')'                     0.15
6. < F >::= 'Abs (' < F > ')'                      0.15
7. < F >::= '(' < F > ')'                          0.13

< E > ::= < Var > | < Num > < Var >
< Var >::='x'
< Num >::= '-' ? < Digit > + ('. ' < Digit > +)?
< Digit >::=[' 0'-'9']
```

Wayner, Huffman kodlamasından yararlanarak olasılıksal gramerleri kullanmak için bir yöntem önermiştir. Aynı zamanda önerilen yöntemimizde de aynı yol izlenebilir. Bu amaçla, her kuralın olasılığı o kuralın sıklığı olarak görülebilir ve her kural için Huffman ağacını kullanarak benzersiz bir kod elde edilebilir. Şekil 60 < E > kurallarının olasılığına göre Huffman ağacını göstermektedir.

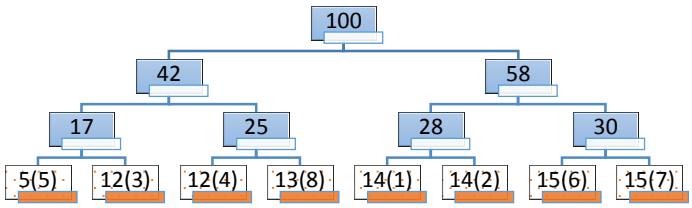


Şekil 60. < E > kurallarının olasılığına göre Huffman ağacı

Şekil 60'a benzer < T > ve < F > kuralları için de ağaçlar üretilebilir. Şekil 61 ve Şekil 62 bu ağacları göstermektedir.



Şekil 61. < T > kurallarının olasılığına göre Huffman ağacı



Şekil 62. < F > kurallarının olasılığına göre Huffman ağacı

Huffman ağaçlarını kullanarak her kural için özel kod geliştirilebilir. Liste 15 Huffman algoritması ile Liste 14'deki dilbilgisini gösterir.

Liste 15. Huffman kodları ile ilişkilendirilen kurallara sahip bir stokastik gramer

```
0. < E >::= '(' < E > ')'                          11010
1. < E >::= < T > '+' < E >                        010
2. < E >::= < T > '-' < E >                        011
3. < E >::= < F > '*' < E >                        1110
4. < E >::= < F > '/' < E >                        1111
5. < E <::= < E > '^' < E >                        11011
6. < E >::= 'Sin (' < E > ')'                      0010
7. < E >::= 'Cos (' < E > ')'                      0011
8. < E >::= 'Tan (' < E > ')'                      0000
9. < E >::= 'Cot (' < E > ')'                      0001
10.< E >::= 'Log (' < E > ')'                      11001
11.< E >::= 'Exp (' < E > ')'                      1100010
12.< E >::= 'Sqrt (' < E > ')'                     100
13.< E >::= 'Abs (' < E > ')'                      101
14.< E >::='Lim (' < C > '),(' < E > ')'           1100011
15.< E >::='Fact(' < C > ')'                       110000

0. < T >::= < F > '*' < T >                        00
1. < T >::= < F > '/' < T >                        111
2. < T <::= < T> '^' < T >                         1100
3. < T >::= 'Sin (' < T > ')'                      1101
4. < T >::= 'Cos (' < T > ')'                      010
5. < T >::= 'Tan (' < T > ')'                      011
6. < T >::= 'Abs (' < T > ')'                      100
7. < T >::= '(' < T > ')'                          101

0. < F >::= 'Sin (' < F > ')'                      100
1. < F >::= 'Cos (' < F > ')'                      101
2. < F >::= 'Tan (' < F > ')'                      001
3. < E >::= 'Cot (' < F > ')'                      010
4. < E >::= 'Log (' < F > ')'                      000
5. < E >::= 'Sqrt (' < F > ')'                     110
6. < F >::= 'Abs (' < F > ')'                      111
7. < F >::= '(' < F > ')'                          011

< E > ::= < Var > | < Num > < Var >
< Var >::='x'
< Num >::= '-' ? < Digit > + ('. ' < Digit > +)?
< Digit >::=[' 0'-'9']
```

Liste 15'in dilbilgisinden üretilen ifadelerin daha gerçekçi bir dağılımı olacaktır. Tabi ki, matematiksel ifadelerin yerleştirileceği kapaklara bağlı olarak dilbilgisi olasılıkları belirlenmelidir. Bu amaçla, gerçek matematiksel ifadeleri inceleyerek ve her birinin tekrar sayısını hesaplayarak olasılıkları hesaplamak kolaydır.

Önceki örnek ile Liste 15'deki gramere göre aşağıda verilen ifade üretilebilmektedir:

Örnek : Mesaj ='Example'

1. Example → 69, 120, 96, 109, 112, 108, 101
2. → **010**00101011110000110000001101101011100000110110001100010

3. < E > → **< T_1 > + < E_2 >**

010**00**10101111010011000000110110101110000011011000110010

→ **< F_3 > * < T_4 >** + < E_2 >

01000**101**0111101001100000011011010111000001101100011000 10

→ < F_3 > * < T_4 > **- Abs(< E_5 >)**

0100010**011**1101001100000011011010111000001101100011000 10

→ **(< F_6 >)** * < T_4 > - Abs(< E_5 >)

0100010101**1101**0011000000110110101110000011011000110010

→ (< F_6 >) * **Sin(< T_7 >)** – Abs(< E_5 >)

0100010101111**0011**000000110110101110000011011000110010

→ (< F_6 >) * Sin(< T_7 >) – Abs (**Cos(< E_8 >)**)

010001010111101001**000**0011011010111000001101100011000 10

→ (**Log(< F_9 >)**) * Sin(< T_7 >) – Abs (Cos(< E_8 >))

0100010101111010011000**00**11011010111000001101100011000 10

→ (Log(< F_9 >)) * Sin(**< F_{10} > *< T_{11} >**) – Abs (Cos(< E_8 >))

0100010101111010011000000**011**011010111000001101100011000 10

→ (Log(< F_9 >)) * Sin(< F_{10} > *< T_{11} >) – Abs (Cos(**< T_{12} > - < E_{13} >**))

0100010101111010011000000110**1101011100000**110110001100010

→ (Log(**(< F_{14} >)**)) * **Sin(Cot(< F_{15} >)** * **< F_{16} >/ < T_{17} >**) – Abs (Cos(**< F_{18} > *< T_{19} > - Cot(< E_{20} >)**))

0100010101111010011000000110110101110000011**0110001100010**

→ (Log((**Cos(< F_{21} >)**))) * Sin(Cot(**Sin(< F_{22} >)**) * (**< F_{23} >)/ < F_{24} > * < T_{25} >**) – Abs (Cos(**Sqrt(< F_{26} >) ***< T_{19} > - Cot(< E_{20} >)))

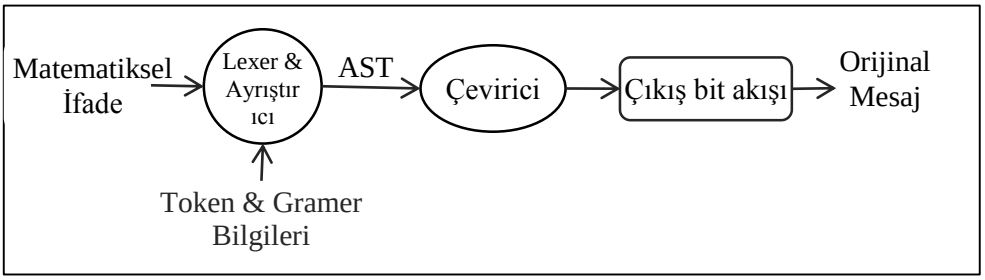
Üretilen ifade en son kural uygulandıktan sonra tüm < T > ve < F >'ler < E >'e dönüştürülerek aşağıdaki ifadenin üretimi yapılabilir

$$(log(cos(E))) * sin(cot(sin(E * \frac{(E)}{E * E}))) + \left|cos\left(\sqrt{E} * E - cot(E)\right)\right|$$

2.4.2.4. Veri Çıkarma (Extractıın)

Mesajın matematiksel ifadelerden ayrıştırılması ve ayıklanması için bir dilbilgisi tabanlı metodoloji önerilmiştir. Önerilen metodoloji ile gerekli işlemi gerçekleştirmek için çok parçalı bir sistem kullanılır. Giriş verileri matematiksel ifade için genişletilmiş bir dilbilgisi kullanılarak ayrıştırılır. Ayrıştırıcının çıktısı, girdi verilerini veri ayıklamak için

kullanılabilecek şekilde modellediği bir ağaç veri yapısıdır. Bu ağacın her düğümü, giriş verilerinin bir sembolünü veya bir kısmını ve sembollerin operatör, sayı vb. olabileceği diğer sembollerle bağlantısını temsil eder. Metodolojimizin adımları Şekil 63'de gösterilmiştir.

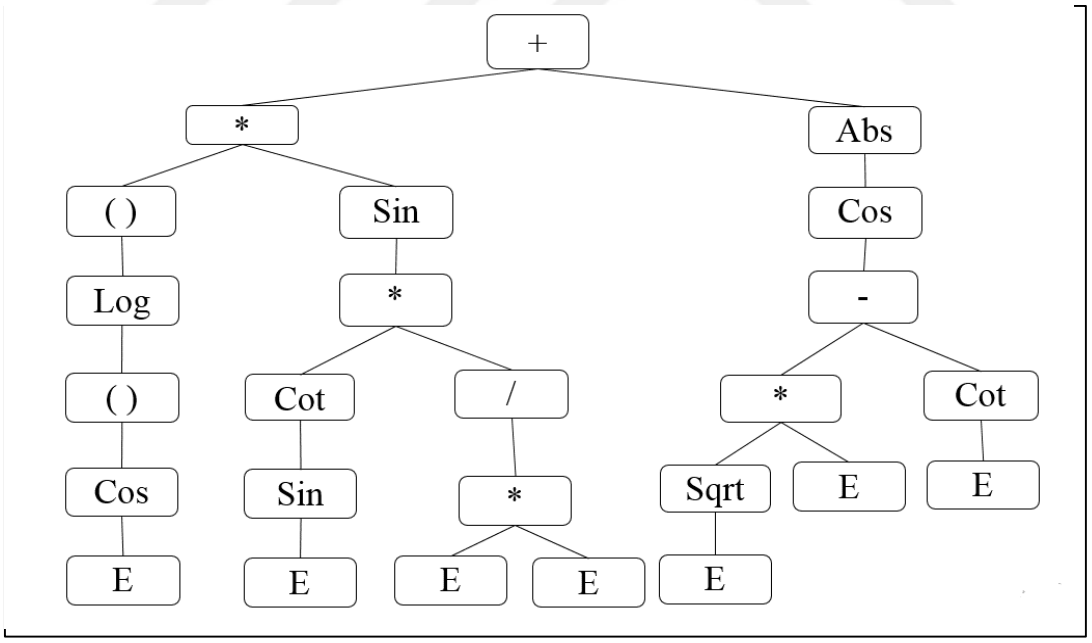


Şekil 63. Önerilen metodolojinin aşamaları ve adımları

Gizlenilen verilerin EXTRACTION yapılması için, matematiksel ifade önce AST üzerine taşınır. Daha sonra ağaç üzerinde dolaşarak her işleme karşı gelen Huffman Kodlar toplanarak orjınal veri elde edilebilir. Önceki bölümde ifadesi üretilen örneği ele alarak veri çıkarma işlemi anlatılmıştır. Soz konusu ifade aşağıdaki gibidir.

$$(\log(\cos(E))) * \sin(\cot(\sin(E * \frac{(E)}{E * E}))) + \left|\cos\left(\sqrt{E} * E - \cot(E)\right)\right|$$

Bu ifade için üretilen AST ağacı Şekil 64 de gösterilmiştir.



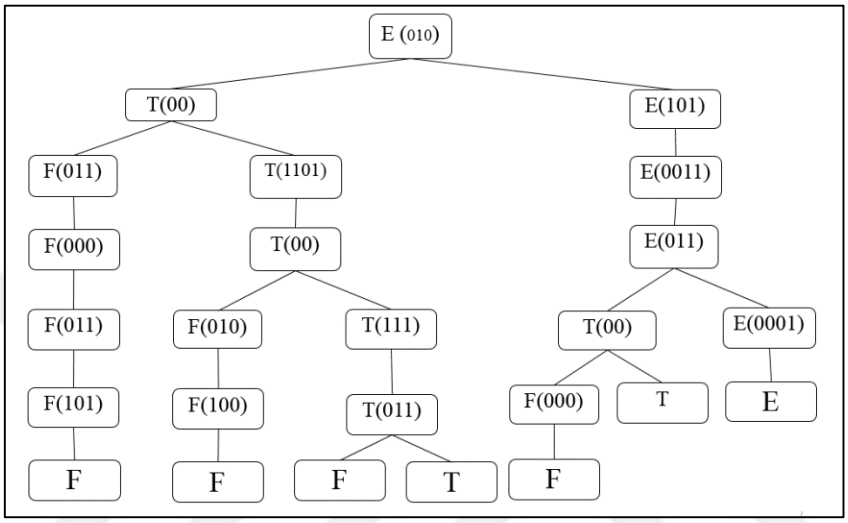
Şekil 64. Örnek ifadenin AST ağacı

Şekil 64'teki ağaçtan veri çıkarmak için aynı yapıda boş AST ağacı üretilir. Daha sonra orijinal ağaç kökünden başlayarak, hangi kural uygulandığı belirlenerek kodu ve çocuklarının

türü boş AST ağacına kaydedilir. Gezinme esnasında her düğümün ilk başta kural türü belirlenip, işlemine göre ona karşı gelen kod Liste 15'den elde edilebilir. Ağacın tümü gezindikten sonra, geçici AST ağacının düğümlerinde yazılan kodlar, üsten aşağıya, soldan sağa toplanıp orijinal mesaj elde edilebilir. Şekil 65 işlem sonundaki geçici AST ağacını göstermektedir. Bu ağacdan elde edilem kod aşağıdaki şekildedir:

0100010101111000011000000110110101110000011011000110010

Bu koda karşı gelen mesaj 'Example' mesajıdır.



Şekil 65. Extraction aşamasında kullanılan AST

2.4.2.5. Matematiksel İfadenin Gösterimi

Programlama dillerine benzer şekilde, matematiksel ifadeleri tanımlamak için çeşitli matematiksel diller vardır. En iyi bilinen üç matematik dili aşağıda tartışılmıştır: MPL, *MathML* ve *LaTex*. Sözdizimi ağacının sonucu matematiksel bir ifadeyse, örn. Türetme, onu düzgün bir şekilde göstermemiz gerekecektir. AST bir ağaç biçimine sahiptir ve bu nedenle insan tarafından okunamaz, LaTex veya MathML formuna dönüştürülmelidir. Infix tarama tekniğindeki AST geçişi, istenen çıktıyı oluşturmak için uygun terimleri elde etmemizi sağlayacaktır.

3. BULGULAR VE İRDELEME

Bu bölümde tezde yapılan çalışmalar 3 ana kısımda irdelenecektir. İlk kısımda önerilen şifreleme yöntemi, bilinen şifreleme değerlendirme araçları ile değerlendirilecektir. Bu değerlendirmede, önerilen yöntem gorüntüler üzerinde uygulanıp tartışılacaktır. Bunun nedeni yöntemin ne kadar etkili olduğunun görüntü üzerinden daha kolay görünmesidir. İkinci bölümde veri gizleme için önerilen matematiksel ifadelerin yerine koyma yöntemi ve üçüncü bölümde ise matematiksel ifade üertim yöntemi ele alınacaktır.

3.1. Şifreleme Yöntemi Değerlendirmesi

Önerilen yöntemin verimliliğini göstermek için Matlab ortamında bir sisteme uygulanmıştır. Bu sistemde, kullanıcı bir görüntü ve bir güvenlik anahtarı seçer. Önerilen algoritmaya göre, renkli görüntüler kırmızı, yeşil ve mavi olmak üzere üç renge dönüştürülmüş ve daha sonra bunları ayrı ayrı şifrelemek / deşifre etmektedir. Şekil 66.a'da, kullanıcının şifreleme için bir güvenlik anahtarını “Exam” olarak kullanıp, deşifrelemede aynı girdiği bir durum için programın bir ekran görüntüsü gösterilmektedir. Şekildeki gibi, deşifre edilen görüntü düz görüntüye çok benzemektedir. Şifre çözme için güvenlik anahtarı şifrelenecek güvenlik anahtarından farklıysa, elde edilen görüntü net görüntüden çok farklı olacaktır. Bu, şifrelemede anahtarın "Exam"e olduğu ve deşifrede ilgili anahtarın "Exap" olduğu durum Şekil 66.b'de gösterilmektedir.

Bu raporda, karşılığı olan önerilen sistemle ilgili etkinliği göstermek için, sistem bir dizi basit görüntü ile test edilir. Görüntü USC-SIPI ile ilgili olarak seçilmiştir. Sonuçlar sistemin yüksek performansını göstermektedir.

(a) (b)

Şekil 66. Başarılı (a) ve Başarısız (b) Şifreleme

3.1.1. Diferansiyel Saldırı Analizi

Uygun bir şifreleme yöntemi, her türlü kriptanalize karşı tutarlı olmalıdır. Şifrelenmiş görüntüler için ortaya çıkabilecek saldırı türlerinden biri de diferansiyel saldırıdır. Diferansiyel kriptanaliz keşfi genellikle Eli Biham ve Adi Shamir'e atfedilir [103, 104].

Başka bir deyişle, görüntü şifreleme sistemleri şifre görüntüsünde sadece düz görüntüye küçük değişiklikler uygulanarak büyük ölçüde değişir. Diferansiyel saldırıya karşı şifreli algoritmaların savunma gücünü değerlendirmenin önemli yöntemlerinden biri, NPCR (değişen piksel oranı sayısı) ve UACI (birleştirilmiş ortalama değiştirilmiş yoğunluk) testidir.

Bu testler ilk kez 2004 yılında uygulanmıştır [105] ve günümüzde görüntü şifrelemesi için güvenlik analizinde yaygın olarak kullanılmaktadır [106]. P_1(i, j) ve P_2 (i, j) 'nin sırasıyla P_1 ve P_2 , iki görüntülerinin (i, j) piksel olduğunu varsayalım, NPCR [107]

$$NPCR = \frac{\sum_{i=1}^{M}\sum_{j=1}^{N} D(i,j)}{M \times N} \times 100\%$$

Burada, M ve N görüntü için genişlik ve uzunluk olmak üzere D (i, j) aşağıdaki gibi elde edilir;

$$D(i,j) = \begin{cases} 0 & if\ P_1(i,j) = P_2(i,j) \\ 1 & if\ P_1(i,j) \neq P_2(i,j) \end{cases}$$

UACI şöyle tanımlanır;

$$UACI = \frac{\sum_{i=1}^{M}\sum_{j=1}^{N} |P_1(i,j) - P_2(i,j)|}{255 \times M \times N} \times 100\%$$

Araştırmalar [108, 109], 256 seviyeli gri görüntülerde, NPCR ve UACI için beklenen değerlerin sırasıyla 99.6938 ve 33.4862 olduğunu göstermektedir. Tablo 15 ve Tablo 16, [110] 'te kritik değerlere göre 8 standart gri görüntü için NPCR ve UACI değerlerini özetlemektedir. Bu değerleri hesaplamak için, P_1 ve P_2'nin şifreli görüntüleri, sadece bir piksel değiştirilmiş iki görüntü ile ilgili olarak dikkate alınır.*

Tablo 15. (226,346) 'da bir piksel değiştirerek NPCR değerleri

Görüntü boyutu: 256*256 NPCR değeri: α (0.05)=99.5693% α (0.01)=99.5527% α (0.001)=99.5341%				
Görüntüler	NPCR %	α (0.05)	α (0.01)	α (0.001)
Girl	99.6938	authorize	authorize	authorize
Couple	99.6205	authorize	authorize	authorize
Görüntü boyutu: 512*512 NPCR değeri: α (0.05)= 99.5893% α (0.01)=99.5810% α (0.001)=99.5717%				
Görüntüler	NPCR %	α (0.05)	α (0.01)	α (0.001)
Mandrill(a.k.a Baboon)	99.6123	authorize	authorize	authorize
Girl (Lena)	99.6326	authorize	authorize	authorize
Sailboat on lake	99.7312	authorize	authorize	authorize
Peppers	99.6433	authorize	authorize	authorize
Görüntü boyutu: 1024*1024 NPCR değeri: α (0.05)=99.5994% α (0.01)=99.5952% α (0.001)=99.5906%				
Görüntüler	NPCR %	α (0.05)	α (0.01)	α (0.001)
Aerial	99.6828	authorize	authorize	Authorize
Girl (Elaine)	99.6314	authorize	authorize	Authorize

Tablo 16. (226,346) 'da bir piksel değiştirerek UACI değerleri

Görüntü boyutu: 256*256 UACI deüeri: α^- (0.05)= 33.2824%, α- (0.01)= 33.2255%, α- (0.001)= 33.1594% α+ (0.05)= 33.6447%, α+ (0.01)= 33.7016%, α+ (0.001)= 33.7677%				
Görüntüler	UACI %	α (0.05)	α (0.01)	α (0.001)
Girl	33.4862	authorize	authorize	authorize
Couple	33.5812	authorize	authorize	authorize
Görüntü boyutu: 512*512 UACI değeri: α^- (0.05)= 33.2824%, α- (0.01)= 33.2255%, α- (0.001)= 33.1594% α+ (0.05)= 33.6447%, α+ (0.01)= 33.7016%, α+ (0.001)= 33.7677%				
Görüntüler	UACI %	α (0.05)	α (0.01)	α (0.001)
Mandrill(a.k.a Baboon)	33.4901	authorize	authorize	Authorize
Girl (Lena)	33.6211	authorize	authorize	Authorize
Sailboat on lake	33.4632	authorize	authorize	Authorize
Peppers	33.5072	authorize	authorize	Authorize
Görüntü boyutu: 1024*1024 UACI değeri: α^- (0.05)= 33.2824%, α- (0.01)= 33.2255%, α- (0.001)= 33.1594% α+ (0.05)= 33.6447%, α+ (0.01)= 33.7016%, α+ (0.001)= 33.7677%				
Görüntüler	UACI %	α (0.05)	α (0.01)	α (0.001)
Aerial	33.4564	authorize	authorize	authorize
Girl (Elaine)	33.5120	authorize	authorize	authorize

3.1.2. İstatistiksel Saldırı

Bu yöntemin yeteneklerini değerlendirmek için diferansiyel saldırının üzerinde istatistiksel bir analiz yapılmıştır. Aşağıda, sistemin hem etkilerini araştırmak için hem düz görüntü hem de şifre görüntüsü için Korelasyon Katsayısı ve Histogram analizleri kullanılmıştır.

3.1.2.1. Korelasyon Katsayısı

Bitişik piksellerde yüksek korelasyon, görüntüdeki özelliklerden biridir. Difüzyon ve karışıklığı araştırmak için, şifre içindeki düz görüntünün bitişik piksellerinin korelasyon faktörü azaltılmalıdır. Bu nedenle, 3000 çift bitişik piksel rasgele seçilerek, yatay, dikey ve diyagonal korelasyon faktörleri hem görüntü hem de şifre olarak aşağıdaki gibi hesaplanmıştır:

$$r_{x,y} = \frac{E\{[x - E(x)][y - E(y)]\}}{\sqrt{D(x)}\sqrt{D(y)}}$$

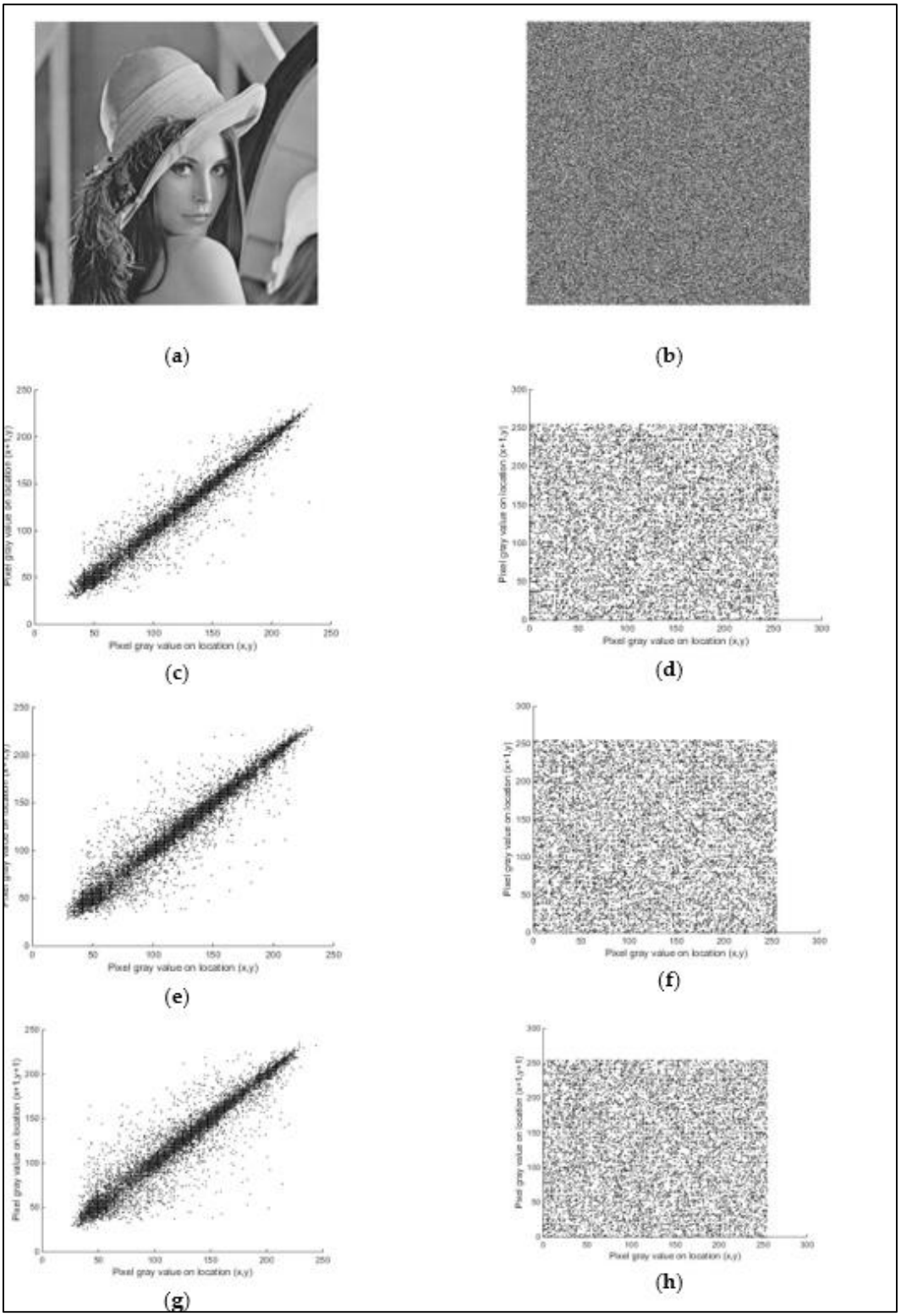
$$E(x) = \frac{1}{s}\sum_{i=1}^{s} x_i$$

$$D(x) = \frac{1}{s}\sum_{i=1}^{s} [x_i - E(x)]^2$$

Burada, x ve y en yakın piksellerin sayısıdır. E(x) beklentidir ve D(x) x varyansını gösterir. Tablo 17 ve Şekil 67, bu analizin sonuçlarını göstermektedir.

Tablo 17. Korelasyon katsayısı analizi

Test Görüntüleri	Düz görüntüü			Şifreli resmi		
	Dikey	Yatay	Diyagonal	Dikey	Yatay	Diyagonal
Couple	0.9551	0.9295	0.9052	0.0030	0.0134	0.0026
Mandrill	0.8519	0.9046	0.8196	0.0011	0.0041	0.0052
Lena	0.9852	0.9711	0.9584	0.0047	0.0057	0.0011
Sailboat on lake	0.9667	0.9675	0.9513	0.0026	0.0041	0.0012
Peppers	0.9818	0.9777	0.9701	0.0029	0.0056	0.0017
Aerial	0.8598	0.9040	0.8233	0.0057	0.0044	0.0036



Şekil 67. (a) ikili görüntü; (b) şifreli görüntü; (c) düz görüntünün yatay korelasyonu; (d) şifreli görüntünün yatay korelasyonu; (e) düz görüntünün dikey korelasyonu; (f) şifrelenmiş resmin dikey korelasyonu; (g) düz görüntünün diagonal korelasyonu; (h) şifrelenmiş resmin yatay diagonal korelasyonu.

Tablo 18, önerilen yöntemin ve diğer bazı yöntemlerin korelasyon katsayısını karşılaştırmaktadır. Bu tabloyu oluşturarak, düz görüntüde güçlü bir etkileşimin, bu önerilen program ile anlaşılır şekilde azaldığını görmekteyiz.

Tablo 18. Sunulan yöntem ve bazı farklı yöntemler için korelasyon katkısı

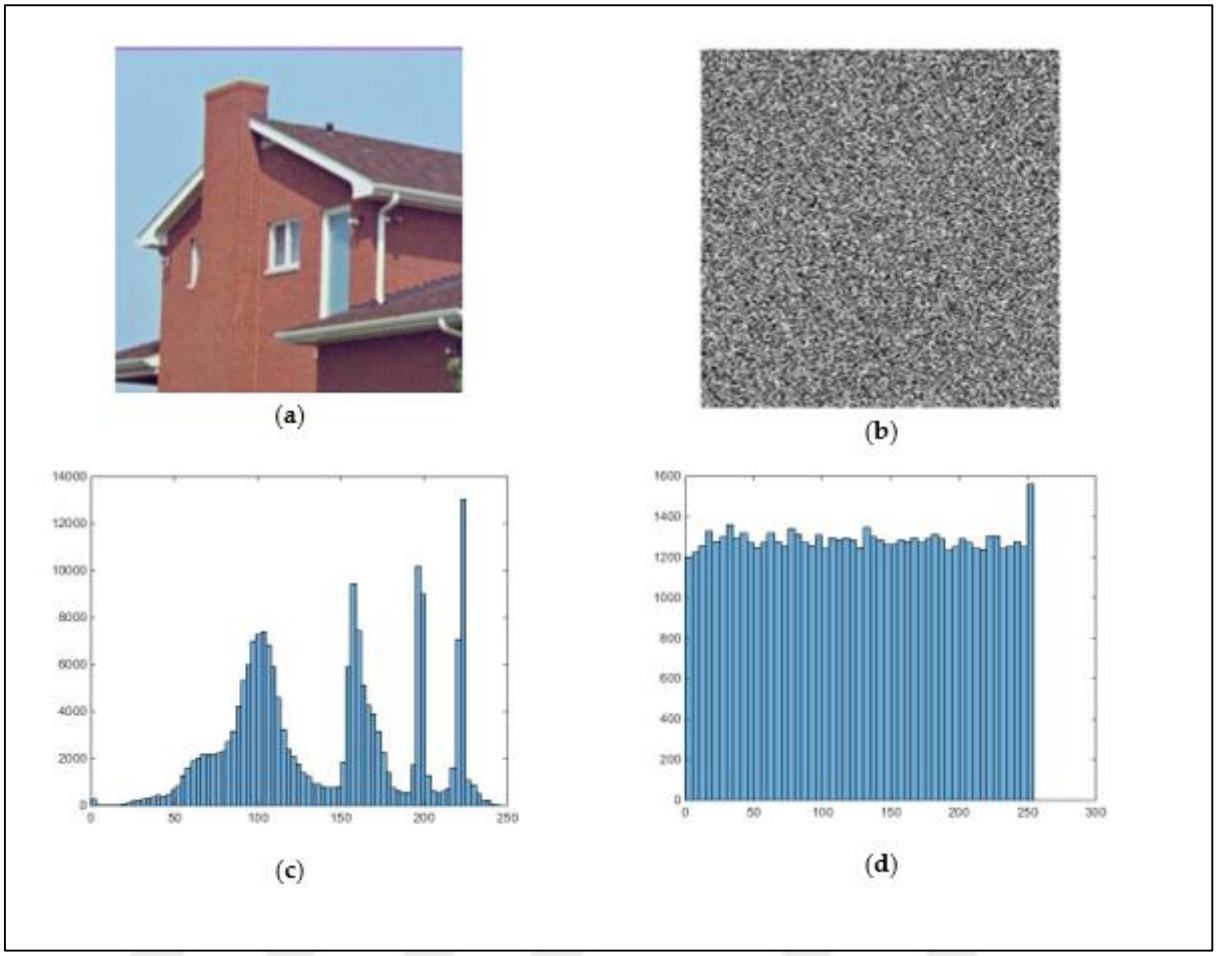
Yöntem	Düz Görüntü			Şifreli Görüntü		
	Dikey	Yatay	Diyagonal	Dikey	Yatay	Diyagonal
Ref.[111]	0.9883	0.9906	0.9823	-0.0160	-0.0093	0.0096
Ref.[112]	0.9883	0.9906	0.9823	0.0092	0.0043	0.0098
Ref.[113]	0.9883	0.9906	0.9823	-0.0344	0.0162	0.0547
Ref.[114]	0.9883	0.9906	0.9823	0.0256	-0.0061	0.03712
Proposed Method	0.9883	0.9906	0.9823	0.0047	0.0057	0.0011

3.1.2.2. Histogram

Histogram analizi, istatistiksel yönteme karşı önerilen yöntemin kalitesini gösteren başka bir testtir. Bir histogram, piksellerin görüntünün gri alanındaki dağılımını gösterir. Şifrenin histogram tekdüze hale getirilmesi, düz görüntülerde şifreleme algoritmalarının etkilerinden biridir. Şekil 68'in histogramları dikkate alındığında, düz görüntüye kıyasla şifrelenmiş görüntünün homojenleştirilmesi açıktır. Bu homojenleştirme, aşağıdaki gibi ki-kare analiziyle incelenir [115]:

$$X_{test}^2 = \sum_{i=1}^{t} \frac{(o_i - e_i)^2}{e_i}$$

İncelenen görüntülerde t = 256, görüntülerin büyüklüğü (256 * 256) ile eşleştirilmiş, e_i eşittir. $\frac{256*256}{256} = 256$. Tablo 19 bazı görüntülerin ki kare değerini göstermektedir.



Şekil 68. (a) Orjinal House (b) şifreleniş House (c) düz görüntü histogramı (d) şifreli görüntünün histogramı

Tablo 19. Bazı görüntülerin ki-kare değeri

Görüntü	$\chi^2_{calculated}$	$\chi^2_{tabulated}$
House	248.0391	293
Aerial	232.4238	293
clock	251.6875	293
couple	253.6328	293

3.1.3. Anahtar Analizi

Gizli anahtar için iki test kullanılmaktadır. Bu testler, anahtar için hassasiyet ve anahtar alan içerir. Muazzam anahtar alan ve yüksek hassasiyet, kaba kuvvet gibi saldırılara karşı önemli bir rol oynamaktadır.

3.1.3.1. Anahtar Mekan Analizi

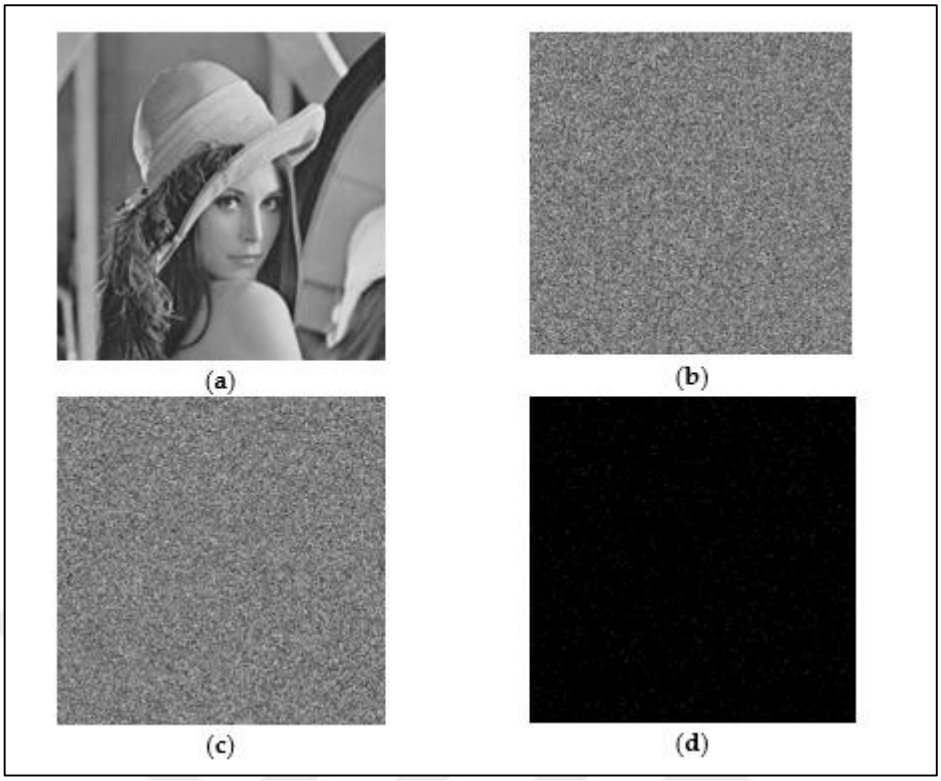
Saldırganlar tarafından kullanılan yöntemlerden biri, kaba kuvvet saldırısı olarak adlandırılır. Bu tür saldırılarda, gizli anahtar için olası tüm durumlar otomatik olarak kontrol edilir ve sonuçları özel bir yazılım ile analiz edilir. Şifrelerin bu tür saldırılara karşı durması için gizli anahtar alanı mümkün olduğunca büyük olmalıdır. Tartışılan yöntemde, rasgele görüntülerin üretilmesiyle birlikte, lojistik harita için altı parametre (x_0, y_0, α_0, α_1, β_0 ve β_1) ve şifreleme işleminin başlangıcında seçilen 48 tablo vardır. Sayıların kesinliği 10^{-12} ise, önerilen gizli anahtar alanı, kaba kuvvet saldırısına karşı $48 * (10^{12})^{6} = 48 * 10^{72}$'dir.

3.1.3.2. Anahtar Hassasiyet Testi

Duyarlılık değerlendirme aşamasında iki konu dikkate alınmalıdır:

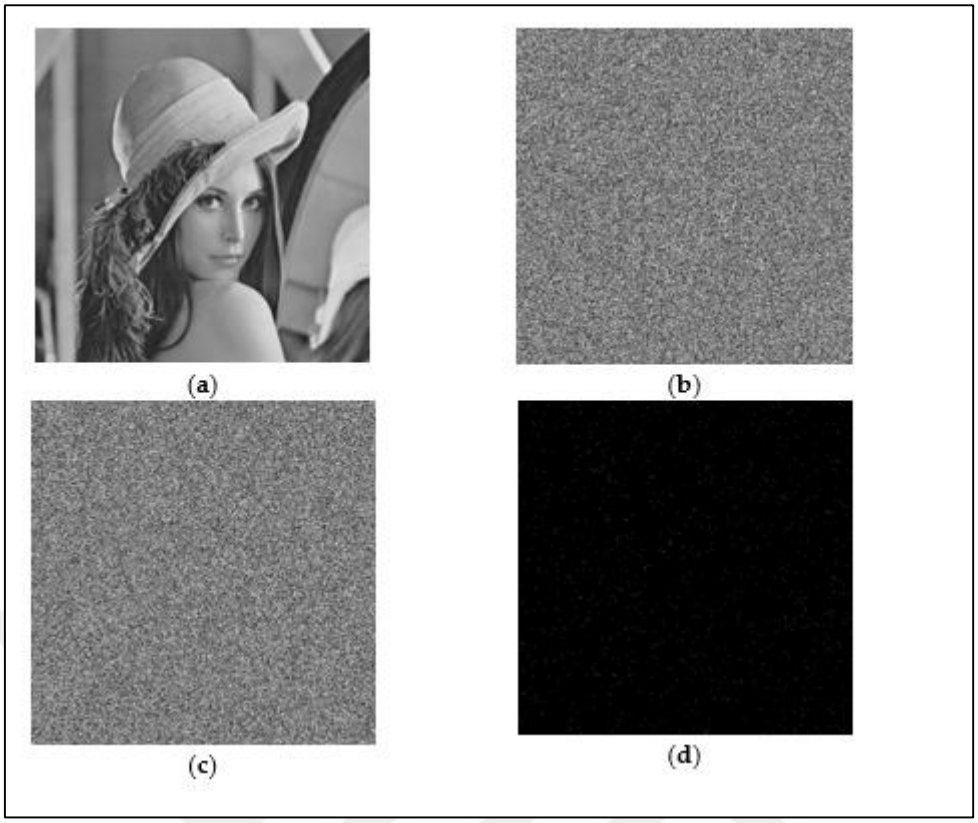
- Düz bir görüntü, yalnızca bir bit değişikliği ile iki anahtarla şifrelenirse, şifreli görüntülerin önemli farklılıkları olmalıdır.
- Sadece bir-bit değişikliği ile bir şifre görüntüsü deşifre edilirse; Elde edilen görüntünün, düz görüntünün kendisi ile önemli farklılıkları olması gerekir.

Önerilen sistemin performansını incelemek için (512 * 512) boyutuna sahip bir görüntü seçilmiştir (Şekil 69.a). Bu görüntü, x_0 ve y_0 başlangıç değerinde yalnızca bir-bit değişikliği olan farklı tuşlarla iki kez şifrelenmiştir (Şekil 69.b ve Şekil 69.c). Şekil 69.d, iki şifrelenmiş görüntü arasındaki farkı göstermektedir. Bu resimde, siyah renk farklı değerlere sahip pikselleri temsil eder ve beyaz renk benzer renklere sahip pikselleri temsil eder; bu nedenle, piksellerin% 99.62'sinin farklı değerleri vardır.



Şekil 69. (a) Lena'nın düz görüntüsü; (b) Başlangıç değeri x0 = 0,1 ve y0 = 0,2 ile şifrelenen görüntü ; (c) Başlangıç değeri x0 = 0.1000000001 ve y0 = 0.2000000001 ile şifrelenen görüntü; (d) iki şifreli görüntü arasındaki farklar

Ayrıca, Şekil 70'de, düz görüntü, k_1 kullanarak şifrelenmiş görüntü, k_2 kullanılarak şifresi çözülmüş görüntü ve düz ve deşifre edilmiş görüntü arasındaki farkı göstermektedir. k_1 ve k_2, sadece bir bit değiştirme ile, düz ve deşifre edilmiş görüntüde % 99.58 piksel farklı değerlere sahiptir. Bu testlerden elde edilen sonuçlar ve analizler, sistemin gizli anahtarlara yüksek hassasiyetini göstermektedir.



Şekil 70. (a) Lena'nın düz görüntüsü ; (b) İlk değer $x_0 = 0.1$ ve $y_0 = 0.2$ ile şifreligörüntü; (c) Başlangıç değeri $x_0 = 0.100000001$ ve $y_0 = 0.2000000001$ ile şifrelenen görüntü; (d) iki şifrelenmiş görüntü arasındaki farklar

3.1.4. Bilgi Entropisi

Prensip olarak, enformasyon entropisi bir olayın ne kadar rasgele olduğunu belirler. Shannon entropi olarak da bilinen enformasyon entropisi, rassal olma derecesini matematiksel bir değerle rapor eder. Bilgi entropisi, sistem belirsizliğinin belirlenmesi için bir araç olarak kullanabilir ve bir kaynağın rastlantısallığını ve tahmin edilemezliğini gösteren bir ölçektir [116]. Görüntü bilgisi ile ilgili belirsizliği ifade ederken, bu ilişkiyi bir değerlendirme olarak kabul edıp ve görüntü piksel değerlerinin dağılımını ölçmek için kullanabilir. Entropiye ilişkin değer daha fazlaysa, piksel değerlerinin daha düzgün dağılımını temsil eder. Genel olarak, bilgi entropisi [117] aşağıdaki gibi tanımlanmıştır:

$$H(S) = \sum_{i=0}^{2N-1} P(s_i) \log\left(\frac{1}{P(s_i)}\right)$$

Denklemin değeri 8'e yaklaştıkça, piksel dağılımı daha fazladır. Örnek bir görüntü üzerinde yapılan bir çalışma, bir örnek kaynak görüntüsü için denklem değerinin 7.408.491

olduğunu ve aynı görüntüye ilişkin şifrelenmiş bir görüntü için bu değerin 7.995.178 olduğunu göstermiştir.

Tablo 20, düz ve şifrelenmiş bir dizi görüntü için düz ve şifreli entropi sonuçlarını göstermektedir. Bu bölümde yapılan incelemeler göz önünde bulundurulduğunda, sistemin saldırılara karşı direncini gösteren o sistem tarafından şifrelenmiş görüntülerin tekdüze ve uygun dağılımını tanıyabilir.

Tablo 20. Düz ve şifreli görüntülerin entropi sonuçları

	Test Görüntüleri							
	Girl	Couple	Mandrill	Lena	Sailboat on lake	Peppers	Aerial	Elaine
Düz görüntü	7.0533	6.4215	7.3579	7.4472	7.4847	7.5944	3.2915	3.4578
Şifreli görüntü	7.9970	7.9973	7.9993	7.9993	7.9993	7.9991	7.9971	7.9994

Ayrıca, Tablo 21'de önerilen yöntem ile diğer bazı yöntemler arasında bilgi entropisi karşılaştırması görülebilir.

Tablo 21. Düz ve şifreli görüntülerin entropi sonuçları

Gçrüntü	Düz	Önerilen Yöntem	Ref.[111]	Ref.[112]	Ref.[113]	Ref.[114]
Lena	7.4472	7.9993	7.9973	7.9993	7.9991	7.9992
Aerial	6.9939	7.9991	7.9969	7.9992	7.9991	7.9992
Peppers	7.5944	7.9991	7.9974	7.9992	7.9992	7.9992
Boat	7.1914	7.9994	7.9970	7.9993	7.9991	7.9993
Elaine	7.4664	7.9994	7.9972	7.9993	7.9991	7.9992
Average	7.33866	7.99926	7.99716	7.99926	7.99912	7.99922

3.1.5. Şifreleme Kalitesi Ölçümü

Şifreleme kalitesini (EQ) değerlendirmek için, [118] ile tanımlanan iki görüntüdeki karşılık gelen pikseller arasındaki farka dayalı olarak aşağıdaki nicel analiz ölçüsünü kullanırız:

$$EQ = \frac{\sum_{i=0}^{255} |o_i(P) - o_i(C)|}{256} \quad (16)$$

Burada $O_i(P)$ ve $O_i(C)$, düz görüntü P'de bayt seviyesi i ve şifrelenmiş görüntü C için gözlenen olaylardır. Bu ölçüm, şifreleme yönteminin güvenlik seviyesinin, EQ'nun değeri arttıkça yükseldiğini ima eder. L çizgileri ve C sütunları olan gri bir görüntü için, EQ_upper olarak gösterilen EQ'nun üst değeri denklem 16, aşağıdaki gibidir:

$$EQ_{upper} = \frac{510 \times L \times C}{256^2} \quad (17)$$

EQ, EQ_{upper} ile kıyaslandığında, önerilen yöntemle şifrelenmiş bazı standart görüntüler için EQ değerleri Tablo 22'de listelenmiştir.

Tablo 22. Şifreleme kalitesi analizi

Görüntü	EQ	EQ_{upper}
House - 256	274.2891	510
Couple -256	317.9063	510
Mandrill - 512	776.9688	2040
Lena - 256	275.2266	510
Sailboat - 512	662.8125	2040
Peppers - 512	568.3047	2040
Elaine-512	646.4347	2040

3.1.6. Zaman Performansı

Bir şifreleme sisteminin zaman performansı için, algoritmanın hem karmaşıklığı hem de performansına ihtiyaç vardır. Karmaşıklık matematiksel ve mantıksal hesaplamalar ve hafıza hücreleri üzerindeki okuma-yazma işlemleri perspektifinden analiz edilir. Performans, şifreleme çıktısı tarafından uygulanan çalışma zamanı ek yükleri ve bir byte şifreleme için gereken döngü sayısı açısından değerlendirilir. Aşağıdaki tanımlar, şifreleme çıktısını (ET) ve tek baytlık bir şifreleme veya şifre çözme işleminde döngü sayısını verir.

$$ET = \frac{\text{Image}_{\text{Size}}(\text{Byte})}{\text{Encryption}_{\text{Time}}(\text{second})} \quad (18)$$

$$\text{Number of cycles per Byte} = \frac{\text{CPU Speed(Hertz)}}{\text{ET(Byte)}} \quad (19)$$

Denk. (19) ayrıca, farklı platformlarda gerçekleştirilen çeşitli şifreleme sistemlerinin çalışma hızını karşılaştırmak için de kullanılabilir. Tablo 23, bazı özel görüntüler için Denk. (18) ve (19) ile hesaplana değerleri göstermektedir.

Tablo 23. Verim ve hız analizi

Görüntü	Encryption throughput (ET)(MB/s)	Bayt başına döngü sayısı
Couple	11.294117	165.5729
Mandrill	10.346666	180.7345
Lena	10.777777	173.5051
airplane	9.264462	201.8465
Average	10.420755	180.4147

3.2. Yerine Koyma ile Veri Gizleme Yönteminin Değerlendirmesi

Bu değerlendirme aslında matematiksel ifadeleri uygun biçime dönüştürüp diğer bilinen substitution yöntemleri uygulanabilecek vaziyete getirme amacıyla yapılmıştır. Görüntü veya metin kapakları üzerinde yapılan yöntemler AST ağacı üzerine taşınabilir. Bu tezde bir kaç örnek verilerek bunun açıklaması yapılmıştır.

Genelde matematiksel ifadeler özel yapıya sahıp olduğundan herhangi bir sıra söz konusu değldir. Ancak tezde önerilen metodoloji ile ve belli bir gezinme yöntemi kullanarak sıralı biçimde duruma bakılabilir.

3.3. İfade Üretim Yönteminin Değerlendirilmesi

Matematiksel ifadelerin sahip olduğu özel yapılar açısından, şüpheli olmayan matematiksel ifadelerin, mesaj için geri dönüş metodolojisi ile üretilmesi mümkündür. Bu amaçla, önerilen yönteme göre uygun bir dilbilgisi tasarlanmıştır. Bu metinlerde farklı kapak metinleri seçebilir ve matematiksel ifadeleri gömebiliriz. Bu metin ve eklenen matematiksel ifade, izleyicide en ufak bir şüphe yaratmamak için bir Stego metni olarak kullanılabilir. Seçilen metinde matematiksel ifadeleri yerleştirmek için *Math Expression Lexer*, *Parser Token, Grammar description Interpreter, Output bitstream AST* ve *Original Message* oluşturulan matematiksel ifadelerin bazı özellikleri olmalıdır. Bu amaçla, istenen dilbilgisi, verilen matematiksel ifadelerin türüne bağlı olarak her kuralın olasılığını belirleyen olasılıksal bir dilbilgisine dönüştürülür. Her kuralın olasılığını belirlemek için, metne göre bir dizi gerçek matematiksel ifadeyi inceleyebiliriz. Oluşturulan matematiksel ifadeleri daha doğal yapmak için farklı sınıflar tanımlanabilir. Üretim zamanında, bu sınıflardan biri rastgele seçilebilir.

Tablo 24, cebirsel bir ifadenin sahip olabileceği desteklenen bazı formların bir listesini göstermektedir.

Tablo 24. Kapak ifadelerin çeşitli biçimleri

Kapak formu	***Kapak formu***
exp	*f*(*x*) = *exp*
f(*exp*) = *exp*	*exp* = *exp*
f(*x*) = *exp* = *exp*	*f*(*exp*) = *exp* = *exp*
exp , *exp*	*f*(*x*) = *exp* , *exp*
f(*x*)=*exp*, *g*(*x*)=*exp*	*f*(*exp*)=*exp*, *g*(*x*)=*exp*
f(*exp*)=*exp*, *g*(*exp*)=*exp*	*exp*=*exp*,*exp*=*exp*
f(*x*)=*exp*=*exp*, *g*(*x*)=*exp*=*exp*	

Bu sınıflardan bazılarını seçerek, oluşturulan matematiksel ifadeler daha doğal görünür ve eklenen mesajın kapasitesini artırabilir. Matematiksel ifadelerin gerçekleştirilmesi için diğer durumlar da düşünülebilir. Örneğin, üretilen ifadeler, Tablo 25'te gösterilen formlara da dönüştürülebilir.

Tablo 25. Oluşturulan ifadelerin bazı biçimleri

İfade formu	**İfade formu**
$\int exp\ dx$	$\lim_{x\to 0} exp$
$\iint exp\ dx\, dy$	$\oint \exp dx$

Önerilen yöntem, bir kapak oluşturma yöntemi olarak, mesajı bir matematiksel ifadeye dönüştürmektedir. Önceki Steganografi yöntemleri, matematiksel ifadeleri bir kapak olarak kullanmadığından, matematiksel ifadelerin özelliklerini inceleyen hiçbir steganaliz örneği geliştirilmemiştir. Bu, tezde önerilen yöntemin gücü olarak düşünülebilir, çünkü bu türden veriler yoluyla güvenli bir şekilde mesaj gönderilebilmektedir.

4. TARTIŞMA VE SONUÇLAR

Bu tezde, matematiksel ifadelerin kullanımına yönelik gramer tabanlı bir metodoloji önerilmiştir. Temel yapı matematiksel problem çözme sistemlerinin geliştirilmesi için kullanılabilir. İfadelerin biçimi, kesin bir yön sergileyen LL(1) gramerleriyle temsil edilir. JavaCC aracını kullanarak, her zaman matematiksel ifadeler için eşsiz (unique) bir türevi izleyen LL(1) parser'leri geliştirilebilir. Bu parser'lerin uygulanması, gramer kurallarının ilgili metotlarla eşlenmesini içerdiğinden nispeten kolaydır. Hiyerarşik bir yapıya sahip Soyut Sözdizim Ağacı (AST) kullanarak bir giriş ifadesi modellenmiştir. Ağaç, operatörlerin önceliğini ve birleşme yönünü (associativity) temsil eder ve böylece ağaç düğümleri arasında anlamsal ilişki kurar. Uygulanan dönüşümleri ve ortaya çıkan ifadeleri yazdırmak için çeşitli belgeler veya raporlar üretilebilir. Tezde, AST içeriğinin yazılabildiği bir belge formatlama sisteminin kullanılması tercih edilmiştir. AST'nin değerlendirilmesi oldukça basittir. Ağaç değişkenler ve sayılar içerir, bu nedenle x, a ve b gibi değişkenlerin bazı değerlere başlatılmasıyla (initializing), sayısal sonuç hesaplanabilir.

AST yapısını kullanarak bu tezde iki farklı ana veri gizleme yöntemi önerilmiştir. Bu yöntemlerin her ikisi de AST yapısından yararlanarak, biri var olan ifade içerisinde mesaj gizlerken, diğeri mesajdan ifade üretmeye çalışır. Her iki önerilen metodoloji ile daha farklı stratejiler kullanarak yeni yöntemleri ortaya çıkarılabilir.

Bu çalışmada gömülen verilerin daha güvenilir olması için kaos özellikleri ve DNA modelleri kullanılarak bir şifreleme yöntemi sunulmuştur. DNA modellerinin kullanımı, sistem verimliliğinin artmasına ve çok yüksek hacimli görüntülerin hızlı şifrelenmesine neden olacaktır. Sistemde, saldırılara karşı artan güvenliğe iki şekilde odaklanılmıştır.

Rasgele DNA dizileri, algoritmanın kodlanmasında uygulanmakta olan kaotik sistem vasıtasıyla yaratılmaktadır. Kaotik sistemlerin içsel doğası ve başlangıç değerine olan duyarlılıkları, önerilen yöntemin performansını ve aynı zamanda anahtar güvenliğine olan duyarlılığını çok cazip kılacaktır. Üstelik, bu yöntem, operatörlerin, düz görüntü ve başlangıç değerlerine büyük ölçüde bağımlı olan operatörler olarak 48 tabloyu kapsamaktadır. Bu da, kriptanalistlere karşı önerilen şifreleme sisteminin stabilitesini, tüm görüntünün kolayca erişilemeyeceği bir şekilde arttıracaktır. Bu nedenle önerilen sistem, anahtar ve algoritma açısından yüksek güvenliğe sahip olacaktır.

KAYNAKLAR

1. Van Hulzen, J. and J. Calmet, Computer algebra systems, in Computer Algebra. 1983, Springer. (1983) 221-243.

2. Dunham, P.H. and T.P. Dick, Research on graphing calculators. The Mathematics Teacher, 87(1994): 440.

3. Palmiter, J.R., Effects of computer algebra systems on concept and skill acquisition in calculus. Journal for Research in Mathematics Education, (1991) 151-156.

4. Artigue, M., Learning mathematics in a CAS environment: The genesis of a reflection about instrumentation and the dialectics between technical and conceptual work. International Journal of Computers for Mathematical Learning, 7 (2002) 245.

5. Butler, G. and J. Cannon. The design of cayley—a language for modern algebra. in International Symposium on Design and Implementation of Symbolic Computation Systems. (1990). Springer.

6. Schönert, M., et al. groups algorithms and programming. in Lehrshuhl D fur Mathematik, RWTH Aachen. (1994). Citeseer.

7. Capani, A. and G. Niesi, CoCoA User's Manual (v. 3.0 b). Dept. of Mathematics, University of Genova, (1996).

8. Van Leeuwen, M., LIE, a software package for Lie group computations. Euromath Bull, 1 (1994) 83-94.

9. McRoy, S.W., S. Channarukul, and S.S. Ali. YAG: A template-based generator for real-time systems. in Proceedings of the first international conference on Natural language generation-Volume 14. (2000). Association for Computational Linguistics.

10. Theune, M., et al., From data to speech: a general approach. Natural Language Engineering, 7(2001) 47-86.

11. White, M. and T. Caldwell, EXEMPLARS: A practical, extensible framework for dynamic text generation. Natural Language Generation, (1998).

12. Stenzhorn, H. XtraGen: a natural language generation system using XML-and Java-technologies. in Proceedings of the 2nd workshop on NLP and XML-Volume 17.(2002). Association for Computational Linguistics.

13. Schellekens, D., B. Preneel, and I. Verbauwhede. FPGA vendor agnostic true random number generator. in Field Programmable Logic and Applications, FPL'06. International Conference on IEEE. (2006).

14. Tkacik, T.E. A hardware random number generator. in International Workshop on Cryptographic hardware and embedded systems. (2002). Springer.

15. Jun, B. and P. Kocher, The Intel random number generator. Cryptography Research Inc. white paper, (1999).

16. Cohn, C.E., Random number generator. (1972), Google Patents.

17. Jakimoski, G. and L. Kocarev, Chaos and cryptography: block encryption ciphers based on chaotic maps. IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications, 48(2001) 163-169.

18. Kocarev, L., Chaos-based cryptography: a brief overview. IEEE Circuits and Systems Magazine,1 (2001) 6-21.

19. Langkilde, I. and K. Knight. Generation that exploits corpus-based statistical knowledge. in Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics-Volume 1. (1998). Association for Computational Linguistics.

20. Coates, R.F., G.J. Janacek, and K.V. Lever, Monte Carlo simulation and random number generation. IEEE Journal on Selected Areas in Communications, 6(1988) 58-66.

21. Ferrenberg, A.M., D. Landau, and Y.J. Wong, Monte carlo simulations: Hidden errors from ‘‘good’’random number generators. Physical Review Letters, 69, 23 (1992) 3382.

22. Schaeffer, L., Application of random regression models in animal breeding. Livestock Production Science, 86,1-3(2004) 35-45.

23. Epstein, M., et al. Design and implementation of a true random number generator based on digital circuit artifacts. in International Workshop on Cryptographic Hardware and Embedded Systems. (2003). Springer.

24. Gilbert, E.N., Random graphs. The Annals of Mathematical Statistics, 30,4 (1959) 1141-1144.

25. Deemter, K.V., M. Theune, and E. Krahmer, Real versus template-based natural language generation: A false opposition? Computational Linguistics, 31, 1 (2005) 15-24.

26. Becker, T. Practical, template–based natural language generation with tag. in Proceedings of the Sixth International Workshop on Tree Adjoining Grammar and Related Frameworks (TAG+ 6). (2002).

27. Kumar, A.N., Explanation of step-by-step execution as feedback for problems on program analysis, and its generation in model-based problem-solving tutors. Technology, Instruction, Cognition and Learning (TICL) Journal, 4, 1 (2006).

28. Yoshikawa, T., K. Shimura, and T. Ozawa. Random program generator for Java JIT compiler test system. in Quality Software, Proceedings. Third International Conference on IEEE. (2003).

29. Melis, E., et al., ActiveMath: A generic and adaptive web-based learning environment. International Journal of Artificial Intelligence in Education (IJAIED), 12(2001) 385-407.

30. Hoffman, D., et al., Two case studies in grammar-based test generation. Journal of Systems and Software, 83, 12 (2010) 2369-2378.

31. Klai, S., T. Kolokolnikov, and N. Van den Bergh. Using Maple and the web to grade mathematics tests. in Advanced Learning Technologies, IWALT 2000. Proceedings. International Workshop on. (2000). IEEE.

32. Almeida, J.J., et al. Math exercise generation and smart assessment. in Information Systems and Technologies (CISTI), 8th Iberian Conference on. (2013). IEEE.

33. Tomás, A.P. and J.P. Leal. A CLP-based tool for computer aided generation and solving of maths exercises. in International Symposium on Practical Aspects of Declarative Languages. (2003). Springer.

34. Gong, L.-H., et al., Single Channel Quantum Color Image Encryption Algorithm Based on HSI Model and Quantum Fourier Transform. International Journal of Theoretical Physics, 57, 1 (2018) 59-73.

35. Wu, J., X. Liao, and B. Yang, Color image encryption based on chaotic systems and elliptic curve ElGamal scheme. Signal Processing, 141(2017) 109-124.

36. Wu, X., J. Weng, and W. Yan, Adopting secret sharing for reversible data hiding in encrypted images. Signal Processing, 143 (2018) 269-281.

37. Maniccam, S. and N.G. Bourbakis, Lossless image compression and encryption using SCAN. Pattern Recognition, 34, 6 (2001) 1229-1245.

38. Yan, B. and S. Bai. Design of image confusion-diffusion cryptosystem based on vector quantization and cross chaotic map. in Image, Vision and Computing (ICIVC), 2nd International Conference on IEEE. (2017).

39. Dalal, M. and M. Juneja. Video steganography techniques in spatial domain—a survey. in Proceedings of the International Conference on Computing and Communication Systems. (2018). Springer.

40. Zhang, G. and Q. Liu, A novel image encryption method based on total shuffling scheme. Optics Communications,. 284, 12 (2011) 2775-2780.

41. Ran, Q., et al., Vector power multiple-parameter fractional Fourier transform of image encryption algorithm. Optics and Lasers in Engineering, 62 (2014) 80-86.

42. Liu, Z., et al., Image security based on iterative random phase encoding in expanded fractional Fourier transform domains. Optics and Lasers in Engineering, 105 (2018) 1-5.

43. Mondal, B., et al., A secure image encryption scheme using chaos and wavelet transformations. Recent Patents on Engineering, 12, 1 (2018) 5-14.

44. Gupta, B., D.P. Agrawal, and S. Yamaguchi, Handbook of research on modern cryptographic solutions for computer and cyber security. (2016) IGI Global.

45. Bhattacharyya, S., I. Banerjee, and G. Sanyal, A novel approach of secure text based steganography model using word mapping method (WMM). International Journal of Computer and Information Engineering, 4, 2 (2010) 96-103.

46. Lockwood, R. and K. Curran, Text based steganography. International Journal of Information Privacy, Security and Integrity, 3, 2 (2017) 134-153.

47. Li, J., et al., Color image watermarking scheme based on quaternion Hadamard transform and Schur decomposition. Multimedia Tools and Applications, 77, 4 (2018) 4545-4561.

48. Balaji, R. and G. Naveen. Secure data transmission using video Steganography. in Electro/Information Technology (EIT), International Conference on IEEE. (2011).

49. Nikam, G., et al., A Survey of Video Steganography Techniques. Journal of Network Communications and Emerging Technologies (JNCET) www. jncet. org,7, 5 (2017).

50. Mishra, S., et al., Audio Steganography Techniques: A Survey, in Advances in Computer and Computational Sciences. (2018), Springer. 581-589.

51. Johnson, N.F. and S. Jajodia. Steganalysis of images created using current steganography software. in International Workshop on Information Hiding. (1998). Springer.

52. Provos, N. and P. Honeyman, Hide and seek: An introduction to steganography. IEEE security & privacy, 99, 3 (2003) 32-44.

53. Czaplewski, B., Current trends in the field of steganalysis and guidelines for constructions of new steganalysis schemes. Przegląd Telekomunikacyjny+ Wiadomości Telekomunikacyjne, (2017) 1121-1125.

54. Wayner, P., Mimic functions. Cryptologia, 16, 3 (1992) 193-214.

55. Yan, B., & Bai, S. Design of image confusion-diffusion cryptosystem based on vector quantization and cross chaotic map. In Image, Vision and Computing (ICIVC), 2nd International Conference on IEEE.(2017) 639-644.

56. Mondal, B., Mandal, T., Khan, D. A., & Choudhury, T. A secure image encryption scheme using chaos and wavelet transformations. Recent Patents on Engineering, 12, 1 (2018), 5-14.

57. Chai, X., Zheng, X., Gan, Z., Han, D., & Chen, Y. An image encryption algorithm based on chaotic system and compressive sensing. Signal Processing, 148, (2018) 124-144.

58. Wang X, Liu L, Zhang Y. A novel chaotic block image encryption algorithm based on dynamic random growth technique. Optics and Lasers in Engineering.66 (2015)10-8.

59. Zhang Y. Cryptanalysis of a novel image fusion encryption algorithm based on DNA sequence operation and hyper-chaotic system. Optik-International Journal for Light and Electron Optics.126, 2 (2015) 223-9.

60. Patel, A., & Parikh, M. A Survey on Multiple Image Encryption Using Chaos Based algorithms And DNA Computing. (2018).

61. Verhulst, P. F. Notice sur la loi que la population suit dans son accroissement. correspondance mathématique et physique publiée par a. Quetelet, 10, (1838). 113-121.

62. Wang X-y, Qin X. A new pseudo-random number generator based on CML and chaotic iteration. Nonlinear Dynamics.70, 2 (2012) 1589-92.

63. Xingyuan W, Xue Q, Lin T. A novel true random number generator based on mouse movement and a one-dimensional chaotic map. Mathematical Problems in Engineering. (2012).

64. Tang G, Liao X, Chen Y. A novel method for designing S-boxes based on chaotic maps. Chaos, Solitons & Fractals.23, 2 (2005) 413-9.

65. Jakimoski G, Kocarev L. Chaos and cryptography: block encryption ciphers based on chaotic maps. IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications.48, 2 (2001) 163-9.

66. Richard Bergmair and Stefan Katzenbeisser, "Content-Aware Steganography: About Lazy Prisoners and Narrow-Minded Wardens" IEEE Dec, (2005).

67. Wikipedia.org/wiki/subliminal_channel 26-04-(2018)

68. M. Gaudesi, E. Piccolo & G. Squillero, "Evolving non-deterministic players for the iterated prisoner's dilemma" Daun, Politecnico di Torio Italy

69. Johnson and Katzenbeisser. A survey of Steganographic techniques. Information hiding Techniques for Steganography and Digital Watermarking. (2000) 43-78.

70. Hossain M.J. Information Hiding using Image Steganography with Pseudorandom Permutation. Bangladesh Research Publications Journal. 9, 3 (2014) 215-225.

71. Rodrigues J.M., Rios J.R. and Puech W. SSB-4 System of Steganography using Bit. 5th International Workshop on Image Analysis for Multimedia Interactive Services (2004).

72. Bandyopadhyay S.K. and Banik B.G. Multi-Level Steganographic Algorithm for Audio Steganography using LSB Modification and Parity Encoding Technique. International Journal of Emerging Trend & Technology in Computer Science (IJETTCS). 1, 2 (2012) 71-74

73. Andaç ŞAHİN, Ercan BULUŞ, M.Tolga SAKALLI, 24-BİT RENKLİ RESİMLER ÜZERİNDE EN ÖNEMSİZ BİTE EKLEME YÖNTEMİNİ KULLANARAK BİLGİ GİZLEME, Trakya Univ J Sci, 7, 1 (2006) 17-22

74. Shashikala Channalli, Ajay Jadhav, Steganography An Art of Hiding Data, Shashikala Channalli et al. International Journal on Computer Science and Engineering1, 3 (2009), 137-141

75. Dhanani, C., K. Panchal, and M. Scholar, Steganography using web documents as a carrier: A Survey. (2013).

76. Kumar, K.A., S. Pabboju, and N.M.S. Desai, Advance text steganography algorithms: an overview. International Journal of Research and Applications, 1,1 (2014). 31-35.

77. Shirali-Shahreza, M.H. and M. Shirali-Shahreza. A new approach to Persian/Arabic text steganography. in Computer and Information Science, 2006 and 2006 1st IEEE/ACIS International Workshop on Component-Based Software Engineering, Software Architecture and Reuse. ICIS-COMSAR. 5th IEEE/ACIS International Conference on. IEEE. (2006)

78. Popa, R., An analysis of steganographic techniques. The Politehnica University of Timisoara, Faculty of Automatics and Computers, Department of Computer Science and Software Engineering, (1998).

79. Por, L.Y., T. Ang, and B. Delina, Whitesteg: a new scheme in information hiding using text steganography. WSEAS Transactions on Computers, 7, 6 (2008) 735-745.

80. Agarwal, M., Text steganographic approaches: a comparison. arXiv preprint arXiv:1302.2718, (2013).

81. Kingslin, S. and N. Kavitha, Evaluative approach towards text steganographic techniques. Indian Journal of Science and Technology, 8, 29 (2015).

82. Pamulaparty, L., TEXT STEGANOGRAPHY

83. Algebraic expression. N.d. In Wikipedia, The Free Encyclopedia. Retrieved January 23, (2015), from http://en.wikipedia.org/wiki/Algebraic_expression

84. Pulman, S. G. Basic Parsing Techniques: an introductory survey. (1991).

85. Jacobs, C. J., & Grune, D. Parsing Techniques-A Practical Guide. (1990).

86. Johnson, S.C., Yacc: Yet another compiler-compiler. 32 (1975) Bell Laboratories Murray Hill, NJ.

87. Donnelly, C. and R. Stallman, Bison. The YACC-compatible Parser Generator. (2004).

88. Earley, J., An efficient context-free parsing algorithm. Communications of the ACM, 13, 2 (1970) 94-102.

89. Gill, A. and S. Marlow, Happy: the parser generator for Haskell. University of Glasgow, (1995).

90. Graver, J.O., The evolution of an object-oriented compiler framework. Software: Practice and Experience, 22, 7 (1992) 519-535.

91. Hudson, S., JAVACUP: LALR parser generator for Java, (1999).

92. Kodaganallur, V., Incorporating language processing into java applications: A JavaCC tutorial. Software, IEEE, 21, 4 (2004) 70-77.

93. Viswanadha, S. and S. Sankar, Java compiler compiler (JavaCC)-The java parser generator. Java. net, https://javacc. dev. java. net/, accessed Aug, (2009). 23.

94. Lundberg, J. Getting started with JavaCC. Vaxjo University, Available at http://uogonline. com/drlee/CS410/Projects (2006).

95. Ryan, C., M. O'Neill, and J. Collins. Grammatical evolution: solving trigonometric identities. in Proceedings of Mendel. (1998). Citeseer

96. Gehani A, LaBean T, Reif J. DNA-based cryptography. Aspects of Molecular Computing: Springer; (2004) 167-88.

97. Zhang, Q., Guo, L., & Wei, X. Image encryption using DNA addition combining with chaotic maps. Mathematical and Computer Modelling, 52, 11-12 (2010) 2028-2035.

98. Kalsi, S., Kaur, H., & Chang, V. DNA Cryptography and Deep Learning using Genetic Algorithm with NW algorithm for Key Generation. Journal of medical systems, 42, 1 (2018), 17.

99. Zhang Q, Guo L, Wei X. Image encryption using DNA addition combining with chaotic maps. Mathematical and Computer Modelling.52, 11 (2010) 2028-35

100. Cisse II, Kim H, Ha T. A rule of seven in Watson-Crick base-pairing of mismatched sequences. Nature structural & molecular biology.19, 6 (2012) 623-7

101. Wang, X., & Liu, C. A novel and effective image encryption algorithm based on chaos and DNA encoding. Multimedia Tools and Applications, 76, 5 (2017), 6229-6245

102. Wang Q, Zhang Q, Wei X, editors. Image encryption algorithm based on DNA biological properties and chaotic systems. Bio-Inspired Computing: Theories and Applications (BIC-TA), Fifth International Conference on IEEE (2010)

103. Biham E, Shamir A. Differential cryptanalysis of DES-like cryptosystems. Journal of CRYPTOLOGY. 4, 1 (1991) 3-72.

104. Biham, E., Shamir, A.: Di_erential cryptanalysis of the full 16-round DES. Springer, (1993)

105. Mao, Y., Chen, G., Lian, S.: A novel fast image encryption scheme based on 3D chaotic baker maps. International Journal of Bifurcation and Chaos 14, 10 (2004) 3613-3624.

106. Wu, Y., Noonan, J.P., Agaian, S.: NPCR and UACI randomness tests for image encryption. Cyber journals: multidisciplinary journals in science and technology, Journal of Selected Areas in Telecommunications (JSAT), (2011) 31-38.

107. Huang, C., Nien, H.: Multi chaotic systems based pixel shu_e for image encryption. Optics Communications 282, 11 (2009) 2123-2127.

108. Fu, C., Chen, J.-j., Zou, H., Meng, W.-h., Zhan, Y.-f., Yu, Y.-w.: A chaos-based digital image encryption scheme with an improved di_usion strategy. Optics Express 20, 3 (2012) 2363-2378.

109. Wu, Y., Zhou, Y., Saveriades, G., Agaian, S., Noonan, J.P., Natarajan, P.: Local Shannon entropy measure with statistical tests for image randomness. Information Sciences 222 (2013) 323-342.

110. Wu, Yue, Joseph P. Noonan, and Sos Agaian. "NPCR and UACI randomness tests for image encryption." Cyber journals: multidisciplinary journals in science and technology, Journal of Selected Areas in Telecommunications (JSAT) 1.2 (2011) 31-38

111. Zhu, Hegui, Cheng Zhao, and Xiangde Zhang. "A novel image encryption–compression scheme using hyper-chaos and Chinese remainder theorem." Signal Processing: Image Communication 28.6 (2013) 670-680.

112. Zhu, Zhi-liang, et al. "A chaos-based symmetric image encryption scheme using a bit-level permutation." Information Sciences 181.6 (2011) 1171-1186

113. Zhou, Yicong, Long Bao, and CL Philip Chen. "Image encryption using a new parametric switching chaotic system." Signal processing 93.11 (2013) 3039-3052.

114. Zhu, Hegui, et al. "An image encryption scheme using generalized Arnold map and affine cipher." Optik-International Journal for Light and Electron Optics 125.22 (2014) 6672-6677.

115. Kwok, H., Tang, W.K.: A fast image encryption system based on chaotic maps with _nite precision representation. Chaos, Solitons & Fractals 32, 4 (2007) 1518-1529.

116. Shannon, C.E.: Communication theory of secrecy systems*. Bell system technical journal 28, 4 (1949) 656- 715.

117. Amin, M., Faragallah, O.S., El-Latif, A.A.A.: A chaotic block cipher algorithm for image cryptosystems. Communications in Nonlinear Science and Numerical Simulation 15, 11 (2010) 3484- 3497.

118. H. E H Ahmed, H. M. Kalash, and O.S.F. Allah. Encryption Efficiency Analysis and Security Evaluation of RC6 Block Cipher for Digital Images. In International Conference on Electrical Engineering (ICEE), 28 (2007) 1–7

ÖZGEÇMİŞ

Sahereh HOSSEİNPOUR 1981 yılında İran'nin Tabriz şehrinde doğdu. İlköğrenimini 1993 yılında Tabriz'de tamamladıktan sonra 2000 yılında Tabriz'in Sedige Kübra lisesinden mezun oldu. 2001 yılında Tabriz Azad Üniversitesi Matematik Bölümü'ne girmeye hak kazanmış ve lisans öğrenimini 2006 yılında tamamlanmıştir. Üniversiteden mezun olduktan sonra Tabriz'deki lise ve teknik liselerde matematik ve bilgisayar derslerinde öğretmen olarak görev aldı. 2009 yılında Karadeniz Teknik Üniversitesi Fen Bilimleri Bilgisayar Mühendisliği Anabilim Dalında Yüksek Lisans öğrenimine başlamıştır ve 2011 aralık ayında tamamlanmış. Daha sonra 2012 yılının bahar döneminde Karadeniz Teknik Üniversitesi Fen Bilimleri Bilgisayar Mühendisliği Anabilim Dalında Doktora öğrenimine başlamıştır. Evli olan Sahereh HOSSEİNPOUR iyi derecede Azeri, Türkçe, Farsça ve orta derecede İngilizce bilmektedir.