1. BÖLÜM

1. GİRİŞ

Günlük hayatta karşımıza çıkan ve gerçek hayat problemleri olarak isimlendirilen problemlerin (çizelgeleme, sıralama, atama, gezgin satıcı, sırt çantası vb.) bazılarını çözmek için etkili algoritmalar geliştirilmesine karşın evet veya hayır cevabı gerektiren ve NP-Sınıfı problemleri olarak adlandırılan TSP(travelling salesman problem), KP(knapsack problem) ve bunun gibi problemler modellemesi kolay fakat çözümü zor bir yapıdadır. Yani, bu problemleri çözmek için etkili algoritmalar geliştirmek günümüzün gelişen teknolojisine uygun olarak önem kazanmıştır.

Dolayısıyla bu tip problemleri ele almak için algortimalar ve onların performans analizini iyi irdelemek gerektiği ortadadır. Hem bu sebepten hem de özellikle son on yılda bilgisayar teknolojisindeki hızlı gelişmeye paralel olarak finans sektörü, elektronik ticaret, bilgi güvenliği ve güvenlik sistemleri bilişim sektöründe çok önemli bir role sahip olması çalışmamızın ikinci bölümü bu bahsedilen konuların genel bir tasnifinden ibarettir.

Kredi kartlarından cep telefonlarına, ulusal güvenlikten kişisel bilgisayarların şifrelenmesine kadar her sektörde karşımıza güvenlik sistemleri ve bilgi güvenliği çıkmaktadır. Bu nedenle; bilgi güvenliğini sağlayacak algoritmaların (şifreleme, sıralama, arama algoritmaları vb.) ve bunların performansı son yıllarda tüm bilim dallarının kapsamına alınmasına rağmen özellikle matematik, bilgisayar ve elektronik ana bilim dallarında bu konuyla ilgili kürsüler ve programlar açılmıştır. Bunları dikkate alarak bu çalışma, algoritmalar ve performans analizi olarak isimlendirilmiş ve bu başlık altında aşağıdaki gibi genel başlıklar detaylı olarak incelenmiştir.

1. Algoritmanın tanımı ve algoritmalar ile ilgili temel kavramlar
2. Algoritmaların performans analizi ve kodlanması
3. Algoritmaların karmaşıklığı ve buna ait semboller
4. Fonksiyonların büyüme oranları ve O, Ω, Φ sembollerinin anlatılması
5. Modüler aritmatik ve ilgili kavramlar
6. Algoritma tipleri ve performanslarının incelenmesi

7. Benzerlik ve uygulamaları
8. Dijital İmza Projelerinin (RSA, Rabin, El Gamma vb.) detaylı incelenmesi ve örneklerle desteklenmesi
9. Algoritmanın tanımı ve matematikteki yeri
10. Algoritmaların performans analizi ve kodlanması
11. Algoritmaların karmaşıklığı ve buna ait semboller
12. Dijital imzalar ve şifreleme algoritmaları
13. RSA, DSA ve ilgili imza projeleri
14. Kriptografinin tanımı ve temel terimler
15. Ulusal bilgi güvenliği
16. Gizli ve açık anahtar uygulamaları
17. Güvenlik metotlarında Smart Kart uygulaması
18. Elektronik ticarette güvenlik
19. Oturum temelli güvenlik yöntemleri
20. İnternette iletişim güvenliği ve PGP

Önsözde anlatıldığı gibi algoritmaların ve performans analizlerinin önemine binaen bu çalışmanın ikinci bölümünde algoritmanın tanımı ve temel bilgiler ele alınmıştır. Bu bağlamda algoritmaların yapısı, karmaşıklığı, performans ölçüm çeşitleri, karmaşıklık sınıfları ve genel olarak algoritmaların matematikteki yeri ve önemi vurgulanmıştır.

Algoritmaların matematikteki yeri ve önemini daha iyi kavrayabilmek için üçüncü bölüm "Matematik Alt Yapı" olarak isimlendirilmiş ve bu başlık altında fonksiyonların büyüme oranları, modüler aritmatik, sayı teorisi ve uygulamaları belli bir hiyerarşi göz önüne alınarak sunulmuştur. Özellikle bu bölümde indirgeme bağıntıları ve onun uygulamaları dikkat çekicidir.

Üçüncü bölümde bu alt yapı hazırlandıktan sonra dördüncü bölümde belli başlı algoritma tipleri anlatılmış ve bunların performans analizi incelenmiştir. Ele alınan belli başlı algoritma tipleri ve performans analizleri, arama algoritmaları, sıralama algoritmaları ve indirgeme algoritmaları olarak sıralanmıştır.

Dijital İmza Projeleri ve buna ilişkin şifreleme algoritmaları çok kapsamlı bir konu olduğu için tek başına bir bölüm olarak beşinci bölümde incelenmiştir. Bu bölümde özellikle RSA Dijital İmza Projelerinin etkinliği vurgulanmıştır. Şifreleme algoritmalarını avantajları ve dezavantajları açıklanmıştır.

Altıncı bölümde Kriptografinin tanımı, ülkelerin kiptografi politikaları ve ülkemizdeki durum incelenmiştir.Yedinci bölümde yapılan çalışmanın elektronik ticaretteki yerinin genel bir incelemesi ve internette güvenlik ele alınmıştır.

Ekler kısmında ise ilk ek olarak kitabımızda geçen konuların detaylarının bulunabileceği internet adresleri, , ikinci ek olarak çalışmamızda ele aldığımız belli başlı konulara ait temel terimlerin anlamlarını içeren küçük bir sözlük, üçüncü ek olarak belli başlı bazı algoritmalar kendi kodlamasına göre ve son ek olarak bir RSA örneğinin şifrelenmesi ve şifrenin çözümü verilmiştir.

Sonuç olarak, hem kendi alanındaki ilk türkçe kitap olması hemde okulumuzun Bilgisayar programında okutulan Algoritma dersinin kaynak ihtiyacını karşılaması bakımından bu çalışmanın başta öğrencilerimize olmak üzere bu konuda lisans,yüksek lisans ve doktora çalışmaları yapanlara ışık tutacak bir nitelikte olduğuna inanıyoruz ve bu çalışmanın daha geniş kapsamlı olması için çalışmalarımıza devam ediyoruz.

BÖLÜM II

2. ALGORİTMALAR

2.1. Algoritmanın Tanımı, Matematikteki Yeri ve Önemi

Endüstri ve hizmet sektörü organizasyonlarının karmaşıklığının artan bir yapıda olması, büyük ölçekli optimizasyon problemleri için çözümleri ve mevcut verimliliği devam ettirmede yeni alternatiflerin belirlenmesini gerektirir. Bu büyük ölçekli optimizasyon problemlerinin formülasyonu, kompleks matematik modellere yol açar ve bu modellerin çözümü yalnızca çok güçlü hesaplama kaynakları kullanılarak elde edilebilir.

Böyle bir matematik uygulamanın kesikli bir yapıda olması gerekir. Bundan dolayı, algoritma fikri bu tip uygulamalarda önemli bir rol oynar. Algoritmanın kesikli bir yapıya sahip uygulamalardaki rolü, matematiğin klasik branşlarında bir fonksiyon bilgisinin önemi gibidir. Matematikte yeri ve önemi böylesine net olan algoritmalara genel anlamda bir prosedür gözü ile bakılabilir. Bu anlamda, bir algoritma için yaygın olarak kullanılan tanımlardan bazıları şunlardır:

- Algoritmalar, problemleri çözmek için adım adım prosedürlerdir.
- Algoritma, bilgisayarda problemlerin bir sınıfını çözmek için bir metoddur.
- Algoritma, bir mekanik kural veya otomatik metod veya bazı matematiksel işlemlerin düzenlenmesi için programdır.
- Algoritma, soruların herhangi verilen bir sınıfına cevaplar bulmakta kullanılabilen bir hesaplama prosedürü (nümerik olması gerekmeyen) için etkili komutların kümesidir.
- Algoritma, açık olarak tanımlanmış olan ve herhangi bir bilgisayara icra edilen bir prosedürdür.

Algoritma için bu tanımları verdikten sonra, algoritmanın, matematikteki yeri ve önemini şöyle özetleyebiliriz:

Her algoritmanın bünyesinde aritmetik ve mantıki adımları bulundurması, tersine matematiksel çözüm isteyen ve analitik yapıda olmayan bütün problemlerin bir algoritma ile ifade edilmesi ve algoritmaların çalışma sürelerinin fonksiyonel bir yapıda olması, algoritmaların matematikteki yeri ve önemini belirtir .

2.2.Algoritmalar Sistemi

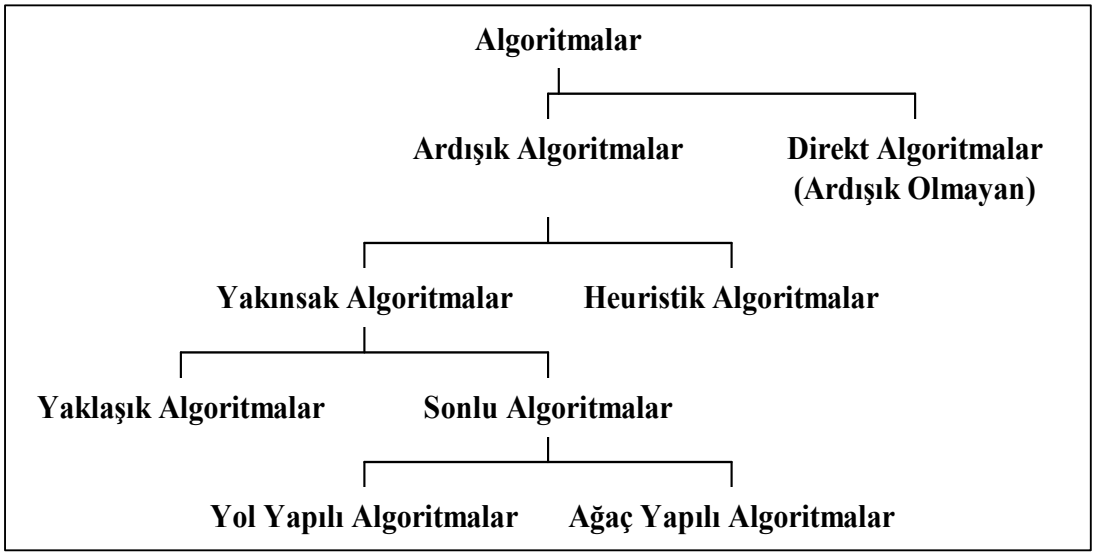
Bir algoritma, belirli bir problem için tatmin edici bir çözüme yakınsamayı sağlayan bir prosedürdür. Endüstrideki problemlerin bazıları çok kompleks olduğu için, Yöneylem Araştırmasının standart modelleri, bu tip problemler için uygun değildir. Bu durum;

- Problemi tarif etmek için gereken verinin çok fazla olduğu
- Problem için doğru bilgi toplamanın zor olduğu durumlarda ortaya çıkar.

Buna karşılık algoritmalar, matematik terimlerle kurulan bir problemin çözümü için prosedürlerdir. Değişik algoritmalar arasında bazı temel farklılıklar vardır.

2.2.1. Algoritma Tipleri

Algoritmalar, gerek yapıları, gerekse çalışma durumlarına göre farklılıklar gösterirler ve şekildeki gibi altı tipe ayrılırlar.



Şekil 2.1. Algoritma Sisteminin Ağaç Diyagramla Gösterimi

a) Direkt (Ardışık Olmayan) Algoritmalar

İterasyonlarda çalışmayan algoritmalar, direkt algoritmalar olarak adlandırılır.

$y= ax^b$, $(a>0)$ polinomunun türevi $y'= abx^{b-1}$ bunun basit bir örneğidir.

b) Ardışık Algoritmalar

Belirli alt prosedürlerin pek çok kez tekrarlandığı algoritmalardır ve ardışık olarak çalışırlar. Algoritmaların çoğu ardışık olarak çalışır.

c) Yakınsak Algoritmalar

Aranılan çözüme doğru yakınsayan ardışık algoritmalardır.

d) Yaklaşık Algoritmalar

Bazı yakınsak algoritmalar kesin çözümü elde edemezler, fakat bu çözüme yaklaşık bir değeri kesin çözüm alırlar. Yaklaşık algoritmalar sonlu değillerdir; fakat herbir ileri iterasyon onları kesin çözüme biraz daha yaklaştırır. Yaklaşık algoritmalara Değişken Kesen Metodu, Arama Teknikleri vb. çok bilinen birkaç örnek verilebilir.

e) Sonlu Algoritmalar

Bu algoritmalar, iterasyonların sonlu bir sayısında kesin çözümü garanti eden yakınsak algoritmalardır ve kendi arasında yol yapılı ve ağaç yapılı olmak üzere ikiye ayrılırlar.

1. Yol Yapılı Algoritmalar: Sonlu algoritmaların pekçoğu yol yapısına sahiptir; bu yol yapısında bir iterasyon bir önceki iterasyonu iterasyon dizilerinde farklı dallar üretmeksizin takip eder.

Böyle algoritmaların örnekleri, Tamsayı Programlamada Kesme Düzlem Algoritmalarının pekçoğu, Şebekelerde Maksimum Akış Algoritmaları, pekçok En Kısa Yol Algoritmaları ve Lineer Programlamanın Simpleks Algoritmaları ve herhangi pivot tekniği ile matris tersleridir.

2. Ağaç Yapılı Algoritmalar: Diğer sonlu algoritmalarda iterasyon dizileri, pekçok paralel dalları içeren bir ağaç şeklindedir. Bir çok ağaç arama algoritmaları bu sınıfa aittir. Bu arama algoritmalarının bazıları, Dinamik Programlama, Dal ve Sınır, Sınırlı Sayım, Kapalı Sayım, Dal ve Çıkarma, Dal ve Atma vb. olarak sıralanabilir.

Yaklaşık algoritmalar bu yol yapısına doğru yönelir. Diğer yandan heuristikler ise bir yol yapısı ile son bulabilen indirgenmiş bir ağaç yapısı olarak gözönüne alınabilir.

2.3. Algoritmanın Yapısı

Bir problemi çözmek için adım adım uygulanacak bir prosedür olarak tanımlanan algoritmanın tipik adımları;

i. Atama adımları, (Bir değişkene bazı değerlerin atanması gibi)
ii. Aritmetik adımlar, (Toplama, bölme, çıkarma, çarpma gibi)
iii. Mantıki adımlardır. (İki sayının karşılaştırılması gibi)

Genel yapısı bu şekilde kısaca özetlenen algoritmalar, belirli bir hata kabülü içinde yaklaşık cevap verirler; uygun programlama dili seçildiğinde genellikle hızlı çalışırlar.

Bir algoritmanın sahip olduğu adımların sayısı (ki bu, algoritmanın gereksinimi olan zamanı büyük ölçüde belirler) problemin bir örneğinden diğerine farklılık gösterir. Problemin bazı "iyi" örnekleri, bu algoritma yardımıyla hızlı olarak çözülebilirse de, problemin bazı "kötü" örneklerini bu algoritma yardımıyla çözmek çok uzun zaman alabilir. Bu, ilgili algoritmanın performansına bağlı bir faktördür. Bu performansın nasıl ölçüleceği Bölüm 2.6 da incelenecektir.

2.4. Algoritmaların Karmaşıklığı

Bir algoritmanın karmaşıklığı, bu algoritmanın çalışma zamanı ile ilgili bir kavramdır. Bir algoritmanın çalışma zamanı olarak da isimlendirilen algoritmanın gereksinimi olan zaman, verinin hem yapısına hem de boyutuna bağlıdır.

Büyük problemler çok daha fazla çözüm zamanı gerektirirken; verideki farklılıklara bağlı olarak aynı boyuttaki değişik problemler belirgin olarak farklı çözüm zamanları gerektirirler. Bir algoritmanın karmaşıklık fonksiyonu, problem uzayı boyutunun bir fonksiyonudur ve verilen boyutun zamanının herhangi bir problem örneğini çözmek için algoritma tarafından ihtiyaç duyulan en geniş zamanı belirtir. Bir başka deyişle, zaman karmaşıklığı fonksiyonu, problemin boyutu arttıkça çözüm zamanının gelişme hızını ölçer.

Burada dikkat edilmesi gereken konu, zaman karmaşıklığı fonksiyonunun, verilen bir problem uzayı boyutunda herhangi bir problem örneğini çözmek için gereken çalışma zamanını, çözüm için gerekli en büyük çalışma zamanını ölçerek hesapladığıdır. Karmaşıklık fonksiyonu, algoritmanın performansının ölçümünde, problemin giriş verisinin uygun ölçülmesine bağlı olarak bir performans garantisi sağlar.

Bu nedenle zaman karmaşıklığı fonksiyonu aynı zamanda bir algoritmanın en kötü durum karmaşıklığı veya sadece karmaşıklığı olarak bilinir. Bir algoritmanın en kötü durum analizinde bu detaylı bir şekilde anlatılmıştır .

2.5. Polinom ve Üstel Zaman Algoritmaları

Bir algoritmanın etkinliği, en kötü durumda algoritma için gereken adımların sayısını sayarak ve bunu problem uzayı boyutu olan n'nin bir fonksiyonu olarak, n büyüdükçe çalışma zamanının davranışını saptamaktır. Algoritma için gereken adımların sayısı n'nin bir polinom fonksiyonu olduğunda, "*O*" sembolu kullanılır ve algoritmanın çalışma zamanı $O(n^k)$'olarak gösterilir. Burada, *k*, büyük *n*'ler için diğer terimler anlamsız olduğundan, polinomun en büyük derecesidir.

Bir başka deyişle, eğer bir algoritmanın zaman karmaşıklığı fonksiyonu $O(n^k),(k \in N)$ (fonksiyon *k*.nci derecedendir) ise bu algoritmaya **Polinom Zaman Algoritması** denir ve **algoritma $O(n^k)$**'dır denir. Örneğin, *k=1* ise algoritmaya *Lineer*, *k=2* ise algoritmaya *karesel* algoritma denir.

Giriş verisi bir polinom fonksiyon ile sınırlandırılan bir çalışma zamanına sahip olan bir algoritma genellikle *"iyi"* olarak adlandırılır. Bu algoritma ile çözülen problemlere de *"kolay"* denir. Diğer yandan, çalışma zamanı bazı *c>1* için en az c^n kadar hızla artan bir algoritmaya "**Üstel Zaman Algoritması**" denir.

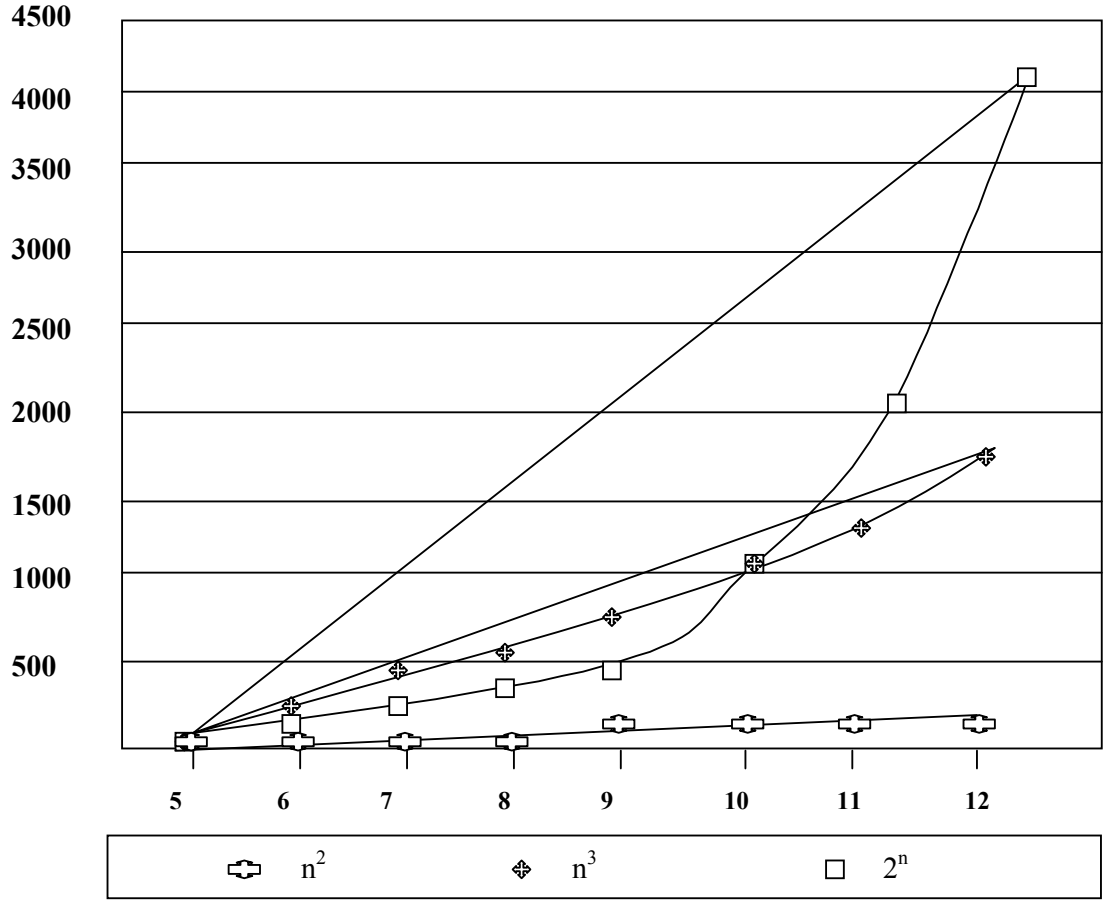
Nlogn gibi ne polinom ne de üstel bir fonksiyon ile sınırlandırılmamış yarı üstel fonksiyonlar olmasına karşılık en kötü durum kriteri bir polinom fonksiyon ile sınırlandırılmamış herhangi bir algoritmaya kolaylık için "*Üstel Zaman Algoritması*" denir. **$O(2n), O(n!), O(n^{logn})$** üstel zaman algoritmalarına örnek olarak gösterilebilir. Polinom ve üstel zaman algoritmaları arasındaki farklılık, saniyede 10^6 işlem düzenleme kapasitesine sahip bir bilgisayarın farklı algoritmaları icra etme zamanı ile hesaplama gücü arttığında en büyük çözülebilir problemin problem uzayı boyutunun karşılaştırılmasıyla ortaya çıkar.

Aşağıdaki tabloda bazı polinom ve üstel fonksiyonların problem uzayının boyutunun *n*'ye bağlı büyüme oranları verilmiştir.

Tablo 2.1. Bazı üstel ve polinom fonksiyonların büyüme oranları

n	**log n**	**$n^{0.5}$**	**N^2**	**n^3**	**2^n**	**n!**
10	332	316	10^2	10^3	10^3	$3{,}6 \times 10^6$
100	664	10	10^4	10^6	$1{,}27 \times 10^{30}$	$9{,}33 \times 10^{157}$
1000	997	31.62	10^6	10^9	$1{,}07 \times 10^{301}$	$4{,}02 \times 10^{2{,}567}$
10000	1329	100.00	10^8	10^{12}	$0{,}99 \times 10^{3010}$	$2{,}85 \times 10^{35{,}659}$

Aşağıdaki şekilde de problem uzayının boyutu olan n'ye göre f(n) hesaplama adımlarının sayısı, $O(n^2)$,$O(n^3)$ ve $O(2^n)$ algoritmalar için gösterilmiştir.



Şekil 2.1. Problem uzayının boyutu n'ye göre bir algoritmayı gerçekleştirmede f(n) hesaplama adımlarının sayısı.

Gerek Tablo 2.1 gerekse Şekil 2.1 den de anlaşılacağı gibi genelde polinom zaman algoritmaları üstel zaman algoritmalarına göre daha küçük derecelere sahip olduklarından daha iyi düzenlenirler.

2.6. Algoritmaların Performans Ölçümü

Bir algoritmanın performansı, o algoritmanın performansının nasıl ölçüleceği sorusundan doğar. Bu sorun, bir problemin çözümü için ortaya konulan algoritmalar arasından "en iyi" algoritmayı seçme ile ortaya çıkar. Bir algoritmanın performans ölçümü için üç temel yaklaşım yaygın olarak kullanılır: Bunlar *deneysel analiz, olasılık (ortalama durum) analizi* ve *en kötü durum analizidir*. Şimdi bu analizleri kısaca açıklayalım:

2.6.1. Deneysel Analiz

Deneysel analiz, örnek problemlerde denenmiş bir algoritmada hesaplama deneyiminde dayanır. Bu analizin amacı, pratikte algoritmanın nasıl davrandığını tahmin etmektir. Bu analizde, algoritma için bir bilgisayar programı yazılır ve problem örneklerinin bazı sınıfları üzerinde programın performansı test edilir .

Bu nedenle deneysel analiz, araştırmacı ve uygulamacıların en çok güvendiği analizlerden biridir ve bilimsel yaklaşımdan çok, uygulamaya yöneliktir .

Deneysel analizin başlıca dezavantajları şunlardır:

i. Bir algoritmanın performansının, programı yazan programcının tekniği kadar kullanılan bilgisayara, derleyiciye ve programlama diline bağlı olması.
ii. Bu analizin çok zaman alması ve düzenlenmesinin pahalıya malolması
iii. Algoritmaların karşılaştırılması; farklı algoritmaların problem örneklerinin farklı sınırlarını daha iyi düzenlemesi ve farklı deneysel çalışmaların birbirini tutmayan sonuçlar vermesi anlamında sıkça kesinlik taşımamasıdır. Bu analizin başlıca avantajı, güncel problemler için kesin ve geçerli olmasıdır .

2.6.2. Olasılık (Ortalama Durum) Analizi

Bu analiz son zamanlarda yoğun bir şekilde kullanılmaya başlanmıştır. Bu analizin amacı, algoritmanın alması beklenen adımlarının sayısını tahmin etmektir. Olasılık analizinde, problem örnekleri için bir olasılık dağılımı seçilir ve algoritma için beklenen asimptotik çalışma zamanlarını üreten istatistik analiz kullanılır .

Olasılık analizinin başlıca dezavantajları şunlardır:

i. Belirli analizin, problem örneklerini temsil etmek için seçilmiş olasılık dağılımına kesin olarak bağlı olması ve olasılık dağılımındaki farklı seçimlerin algoritmaların yapısından kaynaklanan avantajlar nedeni ile ilgili farklı değerlendirmelere yol açabilmesi.
ii. Pratikte karşılaşılan problemler için uygun olasılık dağılımlarını belirlemenin genellikle zor olması.
iii. Olasılık analizinin, algoritmanın en basit tipini değerlendirmek için bile oldukça yoğun matematik alt yapı gerektirmesi ve bu analizin tipik olarak çok kompleks algoritmalar için başarılmasının oldukça zor olması.

Bir algoritmanın performansının önceden tahmini, o algoritmanın olasılık analizine dayanmasına rağmen, analistin problem örneklerinin büyük bir çoğunluğunu çözmesini gerektirmesi ve sonuçların dağılımı hakkında bilgi sağlamaması, bu analizin rağbet görmemesine yol açar.

2.6.3. En Kötü Durum Analizi

En kötü durum analizi, özel bir *H* heuristiği ile bir *P* probleminin örneklerine uygulandığında ortaya çıkan optimalden sapma ile ilgili analizdir. Bu analiz, verilen bir algoritmanın herhangi bir problem örneği üzerinde alabileceği adımların sayısına bir üst sınır getirir.

En kötü durum analizinde diğer iki analizde bulunan dezavantajların çoğu yoktur. Bu analizin iki dezavantajı, bir algoritmanın performansını belirlemek için pratikte son derece nadiren ortaya çıkan anlamsız örneklere izin vermesi, ve bir algoritmanın en kötü durum ortalama performansının önceden bilinmemesidir. En kötü durum analizi, hesaplama çevresinden bağımsızdır ve düzenlenmesi diğer analizlere nisbeten kolaydır. Ayrıca bu analiz, bir algoritmanın adımları (çalışma zamanı) üzerinde bir üst sınır sağlar.

Bu nedenlerden dolayı bir algoritmanın performans ölçümü için bu üç analizden bilimsel literatürde en popüler olanı en kötü durum analizidir. Performans garantisi, en kötü durum oranı olan *r* ile ifade edilir. Buna göre, bazı $r>1$ için $C_h(I)$ ve $C(I)$ sırasıyla bir $I \in P$ minimizasyon problemi örneğinin heuristiği ve optimal çözümü olmak üzere;

$$C_h(I) = r \,.\, C(I) \qquad (\exists\, r>1) \qquad \textbf{(1)}$$

ile ifade edilir. Ayrıca heuristiğin performansı en kötü durum bağıl hatası $\varepsilon=r-1$ ile degerlendirilir. ε ve *r* ikilisi çok kullanılır ve duruma göre bunlardan önemli olanı seçilir [4].

2.7. "O", "Ω", "Φ" Sembolleri

O, Ω, Φ sembolleri, araştırmacıların algoritmaların analizinde kullandığı sembollerdir. Şimdi sırasıyla bunları tanıyalım:

"O" : Eğer *c* ve n_0 sayıları için, bir algoritmanın gereksinimi olan zaman en fazla her $n \geq n_0$ için $c.f(n)$ ise bu algoritma $O(f(n))$ çalışma zamanına sahiptir denir.

Yani, *O* sembolü algoritmanın performansı üzerinde bir üst sınırı belirtir. Problem uzayı boyutu olan n'nin yeterince büyük değerleri için algoritmanın çalışma zamanında bir üst

sınır belirlendiğinde *O(f(n))* karmaşıklık ölçümü çalışma zamanının asimptotik bir büyüme oranına sahip olmasına yol açar .

"Ω" : Ω sembolü, algoritmanın çalışma zamanı üzerinde bir alt sınırı belirtir. Eğer bazı *c'* ve n_0 sayıları ve her $n \geq n_0$ için bir algoritmanın çalışma zamanı bazı problem örneklerinde en az *c'.f(n)* ise bu algoritma *Ω (f(n))* olarak isimlendirilir. *"O"* ve *"Ω"* sembollerini kısaca şu şekilde özetlenebilir :

Eğer bir algoritmanın çalışma zamanı, *O(f(n))* ise sabit bir c sayısı için problemin herbir örneği en çok *c.f(n)* zamanında çalışır. Benzer şekilde, eğer bir algoritma *Ω(f(n))* zamanında çalışırsa, problemin bazı örnekleri *c'* bir sabit olmak üzere en az *c'.f(n)* zamanında çalışır.

"Φ" : *Φ* sembolu bir algoritmanın performansı üzerinde hem alt hem de üst sınırı belirtir. Yani eğer bir algoritma hem *O(f(n))* hem de *Ω(f(n))* ise bu algoritma *Φ(f(n))* olarak isimlendirilir.

Bir algoritmanın performansının alt veya üst sınırı algoritmanın çalışma zamanında bir üst sınır veya bir alt sınır belirleme anlamındadır .

2.8. Optimizasyon

Optimizasyon, en basit anlamı ile eldeki kısıtlı kaynakların en uygun şekilde kullanılması olarak tanımlanır.

Matematiksel bir terim olarak ifade etmek gerekirse optimizasyon, kısaca bir fonksiyonun maksimize (en büyükleme) veya minimize (en küçükleme) edilmesi olarak tanımlanır.Başka bir ifade ile, optimizasyon, "en iyi amaç kriterinin en iyi değerini veren kısıtlardaki değişkenlerin değerini bulmak" şeklinde tanımlanır.

Pratikte bir optimizasyon problemi üç evrede çözülür. Bunlar;

i. Gerçek hayat problemleri için iyi bir model veya formülasyonun bulunması. Örneğin;
   a) model için matematik çözümler, uygulamaya uygun olmalıdır ve
   b) matematik çözümler elde edilebilir olmalıdır.
   c) İleri sürülen model için bir çözüm üretecek etkili metodların bulunması,

ii. Çözüm metodlarının uygulanması,

iii. Sonuçların yorumlanmasıdır.

Gerçek hayat problemlerinin modellenmesi bir sanat olup, çok kompleks problemlerin bunlarla ilgili temel modellerin anlaşılmasına bağlıdır.

2.8.1. Birleşik Optimizasyon Problemleri - BOP

Birleşik Optimizasyon (Combinatorial Optimization), kesikli nesnelerin seçimi, ya da bir optimal dizinin, grubun veya sıralmanın bulunmasındaki matematik çalışmadır.

Birleşik optimizasyon, uygun çözümlerin sonlu bir sayısı ile karakterize edilen problemlerle ilgilenir. Birleşik optimizasyonda karşılaşılan tipik problemler şunlardır: Verilen konumlarda imkanların en uygun kullanımı, müşterilerin en iyi gruplaması, makineler de işlerin optimal (en uygun) sıralaması, işlerin (müşterilerin) acentelere(araç rotalarına) optimal dağıtımı, değişik yatırım ihtimalleri arasında optimal seçim v.b.

BOP'lar, eğer uygun çözümlerin kümesi boş değilse, bir optimal çözümün olması anlamında ekseriye iyi tanımlıdır. Bu problemlerin çoğunda optimal çözümü hesap yoluyla bulmak zordur. Bu nedenle uygun bir hesaplama süresinde büyük ölçekli problemler için optimal çözümleri sağlayabilecek yaklaşık algoritmalara (*heuristik*) sahip olmak önemlidir.

Birleşik optimizasyon, bazen *kesikli programlama (discrete programming)* olarak da ifade edilir. Böylece bir *BOP,* uygun çözümlerin kesikli bir kümesi üzerinde optimize (maksimize veya minimize) amaç fonksiyonunun bir çözümünü araştırır.

Genel bir BOP ;

BOP:min f(x) **(2)**

Kısıtlar $x \in X$

şeklinde tanımlanır.

Burada, $X \subseteq \Omega$, Ω uzayında bir uygun bölgeyi belirtir. Bir başka deyişle, X, zorunlu kısıtları sağlayacak uygun çözümlerin bir kümesidir. $f: X \rightarrow R$ fonksiyonu amaç fonksiyonu olarak bilinir ve R gerçel değerlerin kümesidir. $x \in X$ uygun çözümü, eğer diğer bir y uygun çözümü $f(y)<f(x)$ eşitsizliğini sağlamazsa, optimal çözümdür. X ve Ω kümeleri, özel uygulamalara göre değişiklik gösterirler. Eğer, X ve Ω kümeleri birleşik veya kesikli ise, bir P problemi, *BOP* olarak adlandırılır. Örneğin, n elemanlı bir sonlu kümenin ailesi, 2^n sonlu elemanı içerir.

2.8.2. BOP'ların Karmaşıklığı ve Problem Sınıfları

BOP problemleri genellikle tanımlanması çok kolay; fakat çözmesi çok zor olan problemlerdir. Hesaplama karmaşıklığı teorisi Cook tarafından ortaya konulmuştur. Buna göre, algoritmaların hesaplama gereksinimleri ve pratikte karşılaşılan sınıflandırılmış problemler *zor* veya *kolay* olarak tasnif edilmiştir.

Hesap karmaşıklığı teorisi, pratikte karşılaşılan problemlerin algoritmaları ve önemli örneklerinin her ikisinin de hesaplamalarının sınıflandırılmasını konu alır. Bu teorinin en önemli konusu, *NP Complete (Non Deterministic Polynomial Time Complete) sınıfı* problemlerdir .

Eğer, belirli bir problemin her örneğini çözecek bir polinom zaman algoritması geliştirebilirsek, BOP'ların bu sınıfı *"kolay"* olarak adlandırılır ve *"P"* ile gösterilir. Polinom zaman algoritması, etkili bir algoritma olarak bilinir. Bir algoritmanın zaman karmaşıklığı, verilen bir problem örneğini çözmek için algoritmanın gereksinimi olan aritmetik işlem adımlarının sayısıyla ölçülür .

Bu durum, genellikle en kötü durum kriteri olarak bilinir. k, bir sabit ve n problem uzayının boyutunu göstermek üzere, eğer bir problem $O(n^k)$ hesap karmaşıklığında bir algoritmaya sahipse, bu problem kolaydır. Burada $O(n^k)$;

$a_k n^k + a_{k-1} n^{k-1} + \ldots\ldots\ldots\ldots\ldots + a_1 n + a_0$ (a_i'ler sabit) **(3)**

şeklinde bir fonksiyonu ifade eder. Bu şekilde bir karmaşıklık, polinom mertebeden olarak ifade edilirken; algoritma k. mertebeden polinom zaman algoritması olarak adlandırılır. Burada, k=3 olması tercih edilir. Eğer bir problem için o problemi çözecek etkili algoritmalar bulunamazsa, bu problem *"zor"* veya *"çözülemez"* olarak adlandırılır. Polinom zaman algoritması ile çözülemeyen *"zor"* problemler veya üstel işlem zamanı gerektiren, bilinen bütün problemler, üstel zaman algoritması ister.Üstel zaman algoritması isteyen problemlerden sadece küçük ölçekli olanlar çözülebilir. Dünyada bu konuda çalışma yapan araştırmacıların ortak görüşü şudur:

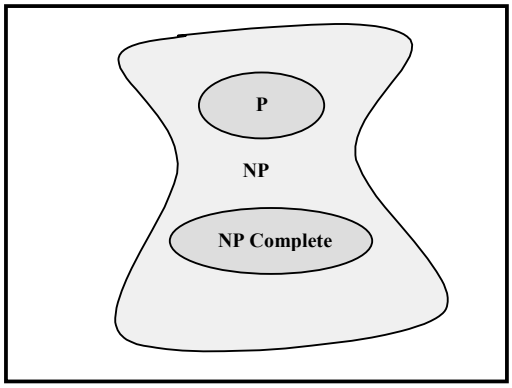
Zor olarak adlandırılan problemler için bu problemlerin herhangi birini çözebilecek etkili bir algoritma geliştirilmemiştir. Bu nedenle, bu zor BOP'ların çözümü için heuristikler ele alınmış ve algoritmalar sistemi içinde heuristik algoritmalar önemli bir yer bulmuştur.
Deterministik olmayan algoritmalar yardımıyla polinom zamanda çözülebilen karar problemlerinin sınıfı *NP(Non Deterministic Polynomial Time)* sınıfı olarak adlandırılır.

NP, *NP Complete (Non Deterministic Polynomial Time Complete)* olarak isimlendirilen problemlerin bir alt kümesini içerir. Bu alt kümedeki her bir problem, NP sınıfına aittir. Eğer, bu problem için etkili bir algoritma mevcutsa, NP sınıfındaki herbir problem için etkili bir algoritma mevcuttur. (Örneğin, P=NP) Bunun anlamı, NP Complete sınıfı problemlerin NP sınıfındaki en zor problem sınıfı olmasıdır.

Karşılaşılan yöneylem araştırması problemlerinin çoğu NP Complete sınıfındadır. Bir probleme uygulanan heuristikler, bazı sebeplerden ötürü NP complete olarak ele alınabilir

Eğer NP sınıfındaki bütün problemler polinom olarak bir probleme indirgenebilirse, bu problem *NP Hard* sınıfına aittir denir. Başka bir deyişle, eğer bir problem NP Hard sınıfına aitse, sadece *P=NP* olan bir polinom zaman algoritması ile çözülebilir .

Problemlerin bu sınıfları arasındaki ilişkiyi basit bir şema ile aşağıdaki gibi ifade edebiliriz.



Şekil 2.2 Farklı problem sınıfları arasındaki ilişki

NP Complete teorisi *"evet"* veya *"hayır"* cevabı gerektiren problemlerden oluşur. Bu evet/hayır cevabı isteyen problemlere, karar problemi denir. Herbir *BOP*, $\overline{BOP}$ ile gösterilen bir karar problemi ile birleştirilir. Verilen bir çözüm uzayı *X*, bir amaç fonksiyonu *f ve* α bir baslangıç değer ise, karar problemi, $f(x) \leq \alpha$ *? Gezgin Satıcı Problemi (Travelling Salesman Problem) (TSP)* şeklinde bir $x \in X$ uygun çözümü mevcut mu şeklinde bir soruyla tanımlanır.

Bunu TSP(Gezgin Satıcı Problemi) üzerinde tanımlayalım:Verilen bir grafik *G(N,A)* ve tamsayı yay uzunluğu C_{ij} olmak üzere herbir $(i,j) \in A$ yayı ile birleştirilsin.

Bu durumda TSP, networkde herbir düğümü dolaşarak;

$$f(w)= \sum_{(i,j) \in w} C_{ij} \qquad \textbf{(4)}$$

tur uzunluğunun mümkün en küçük değerinin tam olarak belirleneceği bir W turunu elde etmektir. TSP için karar problemi $\overline{TSP}$, W turunu içeren bir networkte f(W) toplam uzunluğunun α verilen bir tamsayı iken α'dan küçük olmasıdır. ($f(W)< \alpha$)

Eğer biz TSP'yi polinom zamanda çözebiliyorsak, TSP'yi optimal turun α'dan küçük olup olmadığını kontrol ederek TSP'nin karar problemine ($\overline{TSP}$'ye) bir cevap bulmak için kullanabiliriz.

Fakat bunun tersi doğru değildir. Yani, karar problemi ($\overline{TSP}$),TSP'nin bir cevabını bulmada kullanılamaz. Böylece,TSP ve onun karar problemi olan $\overline{TSP}$, polinom zamanda çözümlerinin olup olmamasına göre eşittirler.

Eğer BOP2 için bir algoritma kullanarak BOP1'i çözebilirsek, BOP1, BOP2'ye indirgendi denir. Buradan da TSP, $\overline{TSP}$'ye indirgenebilir. Eğer BOP1'in herhangi bir örneği polinom zamanda BOP2'nin bir örneğine dönüştürülebilirse, BOP1 problemi, BOP2 problemine polinom olarak indirgenebilir. Bu, şunu gerektirir:

Eğer BOP1 polinom olarak BOP2'ye indirgenir ve bazı polinom zaman algoritması BOP2'yi çözerse, bu polinom-zaman algoritması BOP1'i de çözer.

Problemlerin dört sınıfı vardır: Bunlar sırasıyla P sınıfı, NP sınıfı, NP hard sınıfı ve NP Complete sınıfıdır .

P-Sınıfı: Eğer bazı polinom-zaman algoritması $\overline{BOP}$ karar problemini çözerse, BOP, P-sınıfına aittir denir. P-sınıfı problem örneklerinin bazıları; Minimum Maliyetle Şebeke (network) Akış Problemi, Atama Problemi, En Kısa Yol Problemi ve bunun gibi verilebilir.

NP Sınıfı: Bu sınıfa ait problemler iki şekilde karakterize edilirler;

- NP'deki her problem, sadece bütün sorulara birer birer cevap vererek çözülebilir.

- Karar problemlerinin ($\overline{BOP}$) herbiri, polinom zamanda çözülebilir.

NP Hard Sınıfı: Eğer NP sınıfındaki herhangi bir *BOP1* problemi *BOP*'a indirgenirse, $\overline{BOP}$ karar problemi NP hard sınıfına aittir denir.

NP Complete Sınıfı: Eğer bir $\overline{BOP}$ karar problemi NP ve NP hard sınıflarının her ikisine de aitse, BOP, NP Complete sınıfına aittir denir.

Bu problem sınıfları arasında $P \subseteq NP$ ve $NP\ Complete \subseteq NP$ şeklinde bir ilişki vardır. Bundan dolayı, *NP Complete sınıfı, NP* sınıfındaki en zor problem sınıfıdır.

2.9.Algoritmaların Dili

Kodlama :

1) Procedure = Bir algoritmanın kodlanmasına başlanan ilk ifadedir .Bu ifadede algoritmanın ad Örnek: Procedure max(L = list of integers)

Burada algoritmanın adı max iken tamsayıların L listesinin maksimumunu bulur.

2) Assignments=Atamalar ve ifadelerin diğer tipleri

Assigments ifadesi değişkenlere değer atamada kullanılır .Bu ifadede sol taraf değerin adını alırken sağ taraf ise prosedürlerle tanımlanan fonksiyonları , değerleri atanan değişkenleri , sabitleri içeren bir ifade yada deyimdir .Sağ tarafta ayrıca aritmetik işlemlerin herhangi biride bulunabilir.

: = atamalar için semboldür.

Böylece ;

Variable : = expression

Max: = a (a'nın değeri max değişkenine atanır.)

Ayrıca, x : = Largest integer in the list L şeklinde bir ifade de kullanılır.

x ' i L'de en büyük tamsayı olarak atar.

Güncel bir programlama diline bu ifadeyi dönüştürmek için birden fazla ifade kullanmalıyız . Bunun için ; interchange a and b kullanılır.

Karmaşık prosedürler için ifade blokları kullanılır . Bu begin ile başlar ve end ile sona erer.

```
Begin
    Statement 1
    Statement 2
    .............
    Statement n
end.
```

Şartlı Yapılar

1) **if condition then statement**

   veya

   if condition then

   begin

   blok of statements

end

Eğer durum doğru ise verilen ifade gerçekleştirilir

2) if condition then statement1 else statement 2

Genel form: İf condition 1 then statement 1 else if condition 2 then statement 2 else if condition 3 then Statement 3……..

Else if condition n then statement n else statement n+1

Eğer durum 1 doğruysa ifade 1 gerçekleştirilir v e programdan çıkılır .Eğer, durum1 yanlışsa program durum2 'nin doğruluğunu kontrol eder .Eğer durum2 doğruysa ifade2 gerçekleştirilir . Böylece devam edilir. Sonuç olarak ,eğer durum1 ile durum – n'nin hiçbiri doğru değilse (n+1).ifade yerine getirilir.

Döngü Yapıları

1) for

2) while

1) for variable : = initial value to final value

statement

veya

for variable : = initial value to final value

begin

block of statements

end.

Eğer başlangıç değer sonuç değerden küçük yada eşitse değişken olarak başlangıç değer atanır ve sonuç değer değişken atanana kadar bu döngü gerçekleştirilir.Eğer, başlangıç değer sonuç değeri aşarsa döngü olmaz .Örneğin;

sum : = 0

for i : = 1 to n

sum : = sum+i

2) while condition statement

veya while condition

begin

block of statements

end şeklindedir. Burada durum doğruysa ifade gerçekleştirilir ve durum yanlış olana kadar bu döngü sürdürülür .

3.BÖLÜM

3.MATEMATİK ALT YAPI

Bu bölümde algoritmaların performans ölçüsünde bize yardımcı olacak fonksiyonların büyüme oranları, modüler aritmetik ve buna ait temel kavramlar incelenmiştir.

3.1.Fonksiyonların Büyüme Oranları

İkinci bölümde kısaca bahsetdiğimiz "*O*" notasyonu fonksiyonlarının gelişiminde en yaygın kullanılan notasyon oldugundan burada bu notasyonu daha iyi anlamak için tanım ve örnekleri ele alınacaktır. "*O*" sembolü ilk olarak Alman matematikçi Paul Gustav Heinrich Bachmann (1837-1920) tarafından 1892 de sayı teorisi üzerine önemli bir kitapta kullanılmış olmasına rağmen çalışmasının her aşamasında bu sembolü kullanan Edmund Landau (1877-1938) sayesinde "*O*"sembolü "Landau Sembolü" olarakda adlandırılır.

Tanım 3.1: f ve g fonksiyonları reel sayılardan reel sayılara veya tam sayılardan reel sayılara tanımlı iki fonksiyon olsun. *C* ve *k* sabitleri için ;

$$|f(x)| \le C\,|g(x)| \qquad (x > k)$$

sağlanıyorsa ;

$$\boldsymbol{f(x) = O\,(g\,(x))} \quad \text{denir.}$$

Örnek 3.1 : $f(x) = x^2 + 2x + 1$ için $O(x^2)$ olduğunu gösterelim.

Çözüm 3.1 : $x > 1$ için ;

$0 \le x^2 + 2x + 1 \le x^2 + 2x^2 + x^2 = 4x^2$ olduğundan $C = 4$ ve $k = 1$ için $f(x) = O(x^2)$ olur.

$x > 2$ için ;

$2x \le x^2$ olduğundan

$0 \le x^2 + 2x + 1 \le x^2 + x^2 + x^2 = 3x^2$ burada da $C = 3$ ve $k = 2$ için $f(x) = O(x^2)$ olduğu görülür.

Böylece , $f(x) = x^2 + 2x + 1$ ve $g(x) = x^2$ için $f(x) = O(g(x))$ dir.

Genelleyecek olursak geçişme özelliğinden aşağıdaki sonucu yazabiliriz.

$f(x) = O(g(x))$ ve x' in yeterince büyük değerleri için $h(x)$ fonksiyonu $g(x)$'den daha büyük mutlak değerlere sahip olsun.

Bu durumda ;

$$|f(x)| \le C\,|g(x)| \qquad x > k\,,$$

ve eğer bütün x > k 'lar için $|h(x)| > |g(x)|$ oluyorsa;

$|f(x)| \leq C|h(x)|$ x > k olur.Bu nedenle ;

$f(x) \leq O(h(x))$ elde edilir.

3.2.Fonksiyonların Kombinasyonunun Büyüme Oranı

Algoritmaların çoğu iki veya daha fazla alt prosedürden oluşur.Böyle bir algoritmayı kullanarak belirli ölçüdeki bir problemi çözmek için bilgisayar tarafından kullanılan adımların sayısı bu alt prosedürler tarafından kullanılan adımların sayıları toplamına eşittir. Kullanılan adımların sayısının tahmini için 'O' notasyonu kullanılır.Bunun için iki fonksiyonun toplamı ve çarpımı için 'O' notasyonu nasıl bulunur onu inceleyelim.

$f_1(x) = O(g_1(x))$ ve $f_2(x) = O(g_2(x))$ olsun. C_1, C_2 , k_1 ve k_2 sabitleri için;

$|f_1(x)| \leq C_1|g_1(x)|$ x > k_1 ,

$|f_2(x)| \leq C_2|g_2(x)|$ x > k_2 ,

olur.Bu iki fonksiyonun toplamı ise;

$|(f_1 + f_2)(x)| = |f_1(x) + f_2(x)|$

$\leq |f_1(x) + f_2(x)|$ (üçgen eşitsizliği $|a + b| \leq |a| + |b|$)

x > k_1 , x > k_2 durumu için;

$|f_1(x)| + |f_2(x)| < C_1|g_1(x)| + C_2|g_2(x)|$

$\leq C_1|g(x)| + C_2|g(x)|$

$= (C_1 + C_2)|g(x)|$

$= C|g(x)|$,

burada, $C = C_1 + C_2$ ve $g(x) = \max(|g_1(x)|, |g_2(x)|)$ alındığından k=max(k_1,k_2) iken

$|(f_1 + f_2)(x)| \leq C|g(x)|$ x > k

olarak elde edilir.

Teorem.3.1.

$f_1(x) = O(g_1(x))$ ve $f_2(x) = O(g_2(x))$ olsun.Bu durumda;

$(f_1 + f_2)(x) = O(\max(g_1(x)), g_2(x)))$ .Burada, $\max(g(x), g(x)) = g(x)$ olduğundan

$(f_1 + f_2)(x)$ yine $O(g(x))$'dir.

Sonuç.3.1. $f_1(x) = O(g(x))$ ve $f_2(x) = O(g(x))$ olduğunu varsayalım.Böylece;

$(f_1 + f_2)(x) = O(g(x))$'dir.

Teorem.3.2: $f_1(x) = O(g_1(x))$ ve $f_2(x) = O(g_2(x))$ olsun.Buna göre; x > k_1 , x > k_2 ve C= C_1 C_2 olmak üzere;

$|(f_1\ f_2)(x)| = |(f_1(x)|.|(f_2(x)|$

$\leq C_1|g_1(x)|\ C_2|g_2(x)|$

$\leq C_1\ C_2|(g_1g_2)(x)|$

$$\leq C\,|\,(g_1 g_2)\,(x)\,|$$

$$|\,(f_1\ f_2)(x)\,| \leq\ C\,|\,(g_1\,(x)\,g_2\,(x)\,|\qquad x > k$$

Teorem.3.3:

$f_1(x) = O(g_1(x))$ ve $f_2(x) = O(g_2(x))$ olsun. Buna göre;

$(f_1\ f_2)(x) = O(g_1(x)(g_2(x))$ elde edilir.

3.3. Asimptotik Notasyonlar

i) (asimptotik üst sınır) Her $n \geq n_0$ için . $0 \leq f(n) \leq c.g(n)$ olacak şekilde c pozitif sabiti ile n_0 pozitif tamsayısı bulunursa $f(n) = O(g(n))$'dir.

ii) (asimptotik alt sınır) Her $n \geq n_0$ için . $0 \leq c.g(n)) \leq f(n)$ olacak şekilde c pozitif sabiti ile n_0 pozitif tamsayısı bulunursa $f(n) = \Omega(g(n))$'dir.

iii) (asimptotik sıkı sınır) Her $n \geq n_0$ için . $c_1.g(n) \leq f(n) \leq c_2.g(n))$ olacak şekilde c_1 ve c_2 pozitif sabitleri ile n_0 pozitif tamsayısı bulunursa $f(n) = \theta(g(n))$ 'dir.

iv) (O-notasyonu) Her $n \geq n_0$ için . $0 \leq f(n) \leq cg(n)$ olacak şekilde c pozitif sabiti ile n_0 pozitif tamsayısı bulunursa $f(n) = O(g(n))$'dir

3.3.1. Asimptotik Notasyonların Özellikleri

$f(n)$,$g(n)$,$h(n)$, and $l(n)$, herhangi fonksiyonlar olsun.Buna göre, aşağıdaki özellikler doğrudur.

i) Eğer, $g(n) = \Omega(f(n))$ ise $f(n) = O(g(n))$.

ii) Eğer, $f(n) = O(g(n))$ ve $f(n) = \Omega(g(n))$.ise $f(n) = \theta(g(n))$

iii) Eğer, $f(n) = O(h(n))$ ve $g(n) = O(h(n))$ iken $f + g)(n) = O(h(n))$

iv) Eğer, $f(n) = O(h(n))$ ve $g(n) = O(l(n))$ iken $(f\,.\,g)(n) = O(h(n)l(n))$

v) (Yansıma) $f(n) = O(f(n))$.

vi) (Geçişkenlik) Eğer, $f(n) = O(g(n))$ ve $g(n) = O(h(n))$ iken $f(n) = O(h(n))$

3.3.2. (Bazı Fonksiyonların Karmaşıklık Değerleri)

i) (Polinom fonksiyonlar) Eğer, $f(n)$, k. dereceden bir polinom fonksiyon ise $\boldsymbol{f(n) = \theta(n^k)}$.

ii) Herhangi bir $c > 0$ sabiti için , $\boldsymbol{\log_c n = \theta(\lg n)}$.

iii)(Stirling's formülü) $n \geq 1$ tamsayıları için,

$$\sqrt{2\pi n}\left(\frac{n}{e}\right)^n \leq n! \leq \sqrt{2\pi n}\left(\frac{n}{e}\right)^{n+(1/12n)}$$

ve böylece; $\boldsymbol{n! = \sqrt{2\pi n}\left(\frac{n}{e}\right)^n (1 + \theta(1/n))}$. Aynı zamanda, $\boldsymbol{O(n^n)}$ **and** $\boldsymbol{n! = \Omega(2^n)}$.

iv) $\boldsymbol{lg(n!) = \theta\,(n lg\, n)}$.

Örnek.3.2 : ***e*** ve ***c*** $0 < e < 1 < c$ özelliğindeki keyfi sabitler olsunlar.Bu durumda,

$$1 < \ln\ln n < \ln n < \exp(\sqrt{\ln n \ln\ln n}) < n^e < n^c < n^{\ln n} < c^n < n^n < c^c .$$

3.4.Modüler Aritmetiğe Ait Temel Kavramlar

Tanım3.2: a = 0 olmak üzere a ve b tamsayılar ise ve b = ac şeklinde bir c tamsayısı varsa a, b' yi böler diyebiliriz. a b' yi böldüğünde b'nin faktörü a' dır ve a' nın çarpanı b' dir. a/b şeklinde gösterilir. a / b şeklinde gösterildiğinde a b' yi bölemez anlamındadır.

Tanım.3.3: 1' den büyük pozitif p tamsayısının çarpanları sadece 1 ve kendisi ise böyle sayılara asal denir. 1' den büyük ve asal olmayan pozitif tamsayılara "composite" denir.

Tanım.3.4: Bölüm algoritmasında verilen eşitlikte d bölen, a bölünen, q bölüm ve r kalan olarak adlandırılır.

Tanım.3.5: a ve b, 0' dan farklı tamsayılar olsun. d / a ve d / b olmak üzere en büyük d tamsayısı a ve b' nin en büyük ortak bölen' i olarak adlandırılır. a ve b' nin en büyük ortak böleni gcd(a,b) şeklinde gösterilir.

Tanım.3.6: a ve b tamsayılarının en büyük ortak böleni 1 ise bu sayılara (nispeten) asal sayılar denir.

Tanım.3.7: a_1 , a_2 ,...., a_n 'nin $1 \leq i < j \leq n$ şartıyla $gcd(a_i , a_j) = 1$ ise böyle sayılara ikişerli aralarında asal sayılar denir.

Tanım.3.8: a ve b pozitif tamsayılarının en küçük ortak çarpanı, a ve b'nin her ikisiyle bölünebilen en küçük pozitif tamsayılardır. a ve b' nin en küçük çarpanı lcm(a,b) şeklinde gösterilir.

Tanım.3.9: a, bir tamsayı ve m, bir pozitif tamsayı olsun. a, m' ye bölündüğünde kalanı a mod m şeklinde gösteririz.

Tanım.3.10: a ve b tamsayılar ve m bir pozitif tamsayı olsun. Bu durumda, eğer m a-b' yi bölerse, modul m' ye göre a, b' ye denkdir denir ve a = b (mod m) ile gösterilir. Eğer a ve b modul m'ye göre denk değillerse a ≠ b (mod m) ile gösterilir.

Teorem.3.4: a, b ve c tamsayılar olsun.

1. Eğer a / b ve a / c ise a / (b+c);
2. Eğer a / b ise tüm sayılar için a / bc' dir.
3. Eğer a / b ve b / c ise a / c' dir.

Ispat: a / b ve a / c olduğunu dğüşünelim. Bölünebilirlik tanımından aşağıdaki b = as ve c = at ile s ve t tamsayıları vardır. Bununla beraber;

b + c = as + at = a(s + t)' dir.

Bundan başka; a, b + c' yi böler.

a tamsayısı 1' e ve kendine bölünebildiğinde 1' den büyük her pozitif tamsayı en küçük iki tamsayı ile bölünür. İki farklı pozitif çarpanları içeren tamsayılara asal denir.

Teorem.3.5: Bölüm Algoritması: a ve d pozitif tamsayılar olsun. a = dp + r olmak şartıyla 0 ≤ r < d iken; q ve r tek tamsayılardır.

Teorem.3.6: a ve b pozitif tamsayılar olsun. ab = gcd(a,b)lcm(a,b)' dir.

Teorem.3.7: m bir pozitif tam sayı olsun. a = b + km olmak üzere sadece bir k tamsayısı varsa a ve b modul m' ye göre denkdir.

Ispat: a ≡ b (mod m) ise, m / (a-b)' dir. Bunun anlamı; a – b = km iken bir k tamsayısı vardır. Böylece a = b + km' dir. a = b + km olmak üzere bir k tamsayısı varsa km = a – b' dir. bununla beraber m, a – b' yi böler, böylece a ≡ b (mod m)' dir.

Teorem.3.8: m bir pozitif tamsayı olsun. a ≡ b (mod m) ve c ≡ d (mod m) ise;

a + c ≡ b + d (mod m) ve ac ≡ bd (mod m)' dir.

Ispat: a ≡ b (mod m) ve c ≡ d (mod m) iken, b = a + sm ve d = c + tm ile s ve t tamsayıları vardır. Bununla beraber ;

b + d = (a + sm) + (c + tm) = (a + c) + m(s + t) ve

db = (a + sm)(c + tm) = ac + m(at + cs + stm)' dir.

Ayrıca;

a + c ≡ b + d (mod m) ve

ac ≡ db (mod m)' dir.

3.4.1.Bölünebilirliğin Özellikleri: Her a,b,c Є Z için aşağıdakiler doğrudur.

i) a | a

ii) Eğer, a | b ve b | *c* ise a | *c* olur.

iii) Eğer, a | b ve a | c ise her x,y Є Z için a | (bx + cy) ‘dir.

iv) Eğer, a | b ve b | a ise a = ± b olur.

Teorem.3.9: π(x), x’den küçük asal sayıları göstersin. x ≥ 17 için;

$$\pi(x) > x / \ln x$$

ve x > 1 için;

$$\pi(x) < 1{,}25506\,(x / \ln x)$$

Teorem.3.10 : (Aritmetiğin Temel Teoremi) Her $n \geq 2$ tamsayısı için; p_i’ler farklı asallar ve e_i’ler pozitif tamsayılar olmak üzere;

$$n = p_1^{e_1} p_2^{e_2} \ldots\ldots p_k^{e_k}$$

olup bu yazılış tektir.

Özellik.3.1: Her bir $e_i \geq 0$ ve $f_i \geq 0$ için, Eğer, $a = p_1^{e_1} p_2^{e_2} \ldots\ldots p_k^{e_k}$, $b = p_1^{f_1} p_2^{f_2} \ldots\ldots p_k^{f_k}$ ise;

$gcd(a,b) = p_1^{\min(e_1, f_1)} p_2^{\min(e_2, f_2)} \ldots\ldots p_k^{\min(e_k, f_k)}$

and

$lcm(a,b) = p_1^{\max(e_1, f_1)} p_2^{\max(e_2, f_2)} \ldots\ldots p_k^{\max(e_k, f_k)}$.

Örnek.3.3: $a=4864=2^8.19$ ve $b=3458=2.7.13.19$ olsun.Buna göre,

$gcd(4864,3458)=2.19=38$ ve $lcm(4864,3458)= 2^8.197.13=442624$

3.4.2:Euler phi fonksiyonunun özellikleri

i) Eğer, p asal sayı ise **ø (*p*) = *p* – 1**

ii) Eğer, gcd(m,n) = 1 ise ø ($m.n$) = ø (m) . ø (n)

iii)Eğer, $n= p_1^{e_1} p_2^{e_2} \ldots\ldots p_k^{e_k}$ n’in asal çarpanları ise;

$$ø(n) = n\left(1 - (1/p_1)\right)\left(1 - (1/p_2)\right) \ldots\ldots \left(1 - (1/p_k)\right)$$

Özellik.3.2: Her $n \geq 5$ tamsayısı için,

$$ø(n) > (n / 6 \ln \ln n)$$

3.5.Z 'de Algoritmalar

a ve b negatif olmayan tamsayılar olsun ve her biri *n*'den küçük veya eşit olsun.

Tablo.3.1.Z'de dört işlemin karmaşıklığı

İşlem		Bit Karmaşıklığı
Toplama	a + b	O (lg a + lg b) = O (lg n)
Çıkarma	a – b	O (lg a + lg b) = O (lg n)
Çarpma	a . b	O((lg a) (lg b)) = O ((lg n)^2)
Bölme	a = qb + r	O((lg a) (lg b)) = O ((lg n)^2)

Özellik.3.3 : Eğer, a ve b, a > b şeklindeki pozitif tamsayılar ise gcd(a,b) = gcd(b,a mod b)

Algoritma.3.1: İki tamsayının en büyük ortak bölenini bulan Euclidean algoritması

İnput : a ve b, a ≥ b özelliğindeki iki pozitif tamsayı olsun.

Output: a ve b'nin en büyük ortak böleni;

1. b ≠ 0 iken aşağıdakini gerçekleştir.
   1.1 set r←a mod b, a←b, b←r .
2. return(a)

Sonuç.3.2: Eucledian algoritmasının çalışma zamanı $O((\lg n)^2)$ bit işlemdir.

Örnek.3.4: Eucledian algoritmasından 4854 ve 3458 sayılarının en büyük ortak bölenini bulun.

gcd(4864,3458) = 38:

4864 = 1 . 3458 + 1406

3458 = 2 . 1406 + 646

1406 = 2 . 646 + 114

646 = 5 . 114 + 76

114 = 1 . 76 + 38

76 = 2 . 38 + 0

Eucledian algoritması sadece iki tamsayının en büyük ortak bölenini bulmanın yanısıra x ve y tamsayıları için ax + by = d eşitliği bulmak istendiğinde genişletilebilir.

Algoritma.3.2: Genişletilmiş Euclidean Algoritması

İnput : a ve b, $a \geq b$ özelliğindeki iki pozitif tamsayı olsun.

Output: d = gcd(a,b) olan x,y tamsayıları için $ax + by = d$ sağlanır.

1. if b = 0 then set d←a, x←1, y←0, and return(d,x,y).
2. set x_2←1, x_1←0, y_2←0, y_1←1.
3. while b > 0 do
   3.1 q←[a / b] , r←a – qb, x←$x_2 – qx_1$, y←$y_2 – qy_1$
   3.2 a←b, b←r, x_2←x_1, x_1←x, y_2←y_1 , and y_1←y
4. set d←a, x←x_2 , y←y_2 , and return (d,x,y)

Genişletilmiş Euclidean Algoritmasının çalışma zamanı $O((\lg n)^2)$ bit işlemdir.

Örnek.3.5: Tablo.3.2 'de a = 4864 ve b = 3458 girişleri için Genişletilmiş Euclidean Algoritması kullanılarak gcd(4864,3458) = 38 ve (4864)(32) + (3458) – (-45) = 38 olarak bulunur. İnput: a = 4864 ve b = 3458.

Böylece, gcd(4864,3458) = 38 ve (4864)(32) + (3458) – (-45) = 38

Tablo.3.2. a = 4864 ve b = 3458 girişleri için Genişletilmiş Euclidean Algoritması

Q	r	x	y	a	b	x_2	x_1	y_2	y_1
-	-	-	-	4864	3458	1	0	0	1
1	1406	1	-1	3458	1406	0	1	1	-1
2	646	-2	3	1406	646	1	-2	-1	3
2	114	5	-7	646	114	-2	5	3	-7
5	76	-27	38	114	76	5	-27	-7	38
1	38	32	-45	76	38	-27	32	38	-45
2	0	-91	128	38	0	32	-91	-45	128

3.5.1: Denkliğin özellikleri: Her $a,a_1,b,b_1,c \in Z$ için aşağıdakiler doğrudur.

i) $a \equiv b \pmod{n}$ sadece *n* 'ye bölündüklerinde aynı kalanı verdiklerinde doğrudur.

ii) (yansıma özelliği) $a \equiv a \pmod{n}$

iii) (simetri özelliği) $a \equiv b \pmod{n}$ ise $b \equiv a \pmod{n}$

3.6. Z_n^* Üretecinin Özellikleri

i) Yalnız ve yalnız, $n = 2,4\ p^k$ veya $2p^k$ ve p tek asal sayı ($k \geq 1$) iken Z_n^* üreteçtir.Kısmen de , p asalsa Z_p^* üreteçtir.

ii) Eğer, α, Z_n^* in üreteci ise $Z_n^* = \{ \alpha^i \bmod n \mid 0 \leq i \leq \emptyset(n) - 1 \}$

iii)Varsayalım ki, α, Z_n^* in üreteci olsun.Eğer, gcd(i,Ø(*n*)) = 1 ise b= α^imodn ' de Z_n^* in üreteci olur.Böylece, Z_n^* periyodiktir üretilenlerin sayısı Ø(Ø(*n*))'dir.

iv) Yalnız ve yalnız, Ø(*n*)'in herbir asal böleni için $\alpha^{\emptyset(n)/p} \equiv 1 \pmod n$ olursa $\alpha \in Z_n^*$ Z_n^* 'ın bir üretecidir.

3.6.1.Z_n 'de Algoritmalar: n pozitif bir tamsayı olsun. Buna göre,

Z_n={ 0,1,2,... *n* – 1 }kümesi ile tanımlanır.Eğer, **a,b Є Z_n** ise**;**

$$(a + b) \bmod n = \begin{cases} a+b, & a+b < n \\ a+b-n, & a+b \geq n \end{cases} \text{ olur.}$$

Böylece, Modüler toplama veya çıkarma bölme gerektirmeden düzenlenebilir.

Algoritma.3.3. Z_n'de Çarpmanın Terslerinin Hesabı

Giriş: a Є Z_n

Çıkış: $a^{-1} \bmod n$ 'nin bulunması.

1. $ax + ny = d$ ve $d = \gcd(a,n)$ olan x , y tamsayıları Genişletilmiş Euclidean Algoritması kullanılarak bulunur.

2. Eğer, d > 1 ise $a^{-1} \bmod n$ mevcut değildir.Aksi halde,x'e geri dön.

Not.3.1 : $a^k = \prod_{i=0}^{t} a^{k_i 2^i} = \left(a^{2^0}\right)^{k_0} \left(a^{2^1}\right)^{k_1} \ldots\ldots\ldots \left(a^{2^t}\right)^{k_t}$ şeklindeki modüler üs alma işlemi aşağıdaki algoritma ile etkili olarak düzenlenir ve bu özellik kriptografi protokollerinin çoğu için çok önemli bir araçtır.

Algoritma.3.4: Z_n'de üs alma işlemi için tekrarlanan kare alma ve çarpma algoritmaları

Input : a Є Z_n ve 2'li gösterimi $k = \sum_{i=0}^{t} k_i.2^i$ olan $0 \geq k_n$ olsun

Output : $a^k \bmod n$

1. Set b←1.if k = 0 then return(b).
2. Set A←a .

3. if $k_0 = 1$ then set b←a

4.For i = 1 to t do

4.1 Set A←A

4.2 if $k_i = 1$ then set b←A **.** b mod *n*

5. return (b)

Örnek.3.6:Aşağıdaki tabloda 5^{596} mod 1234 = 1013 'ın hesaplanmasındaki adımlar gösterilmiştir.

Tablo 3.3. 5^{596} mod 1234 = 1013'ün hesaplanması

i	0	1	2	3	4	5	6	7	8	9
k_i	0	0	1	0	1	0	1	0	0	1
A	5	25	625	681	1011	369	421	947	947	925
b	1	1	625	625	67	67	1059	1059	1059	1013

3.6.2:Z_n'de temel işlemlerin karmaşıklığı:Tablo 3.4 ile Z_n'de toplama **,** çıkarma , çarpma, tersini bulma ve üs alma işlemlerine ait bit karmaşıklığı verilmiştir.

Tablo.3.4. Z_n'de temel işlemlerin bit karmaşıklığı

İşlem		Bit Karmaşıklığı
Modüler toplama	(a + b) mod *n*	$O(\lg n)$
Modüler çıkarma	(a – b) mod *n*	$O(\lg n)$
Modüler çarpma	(a **.** b) mod *n*	$O((\lg n)^2)$
Modüler ters	a^{-1} mod *n*	$O((\lg n)^2)$
Modüler üs alma	a^k mod *n*,$k < n$	$O((\lg n)^3)$

4.BÖLÜM

4.ALGORİTMALARIN KARMAŞIKLIK HESABI(PERFORMANS ANALİZİ)

Bu bölümde Arama,Sıralama ve İndirgeme algoritmaları ele alınarak bunların performans analizi incelenmiştir.Algoritmanın performans analizi, algoritmanın karmaşıklığı ile ilgili bir kavram olduğundan, bu konu ilk olarak açıklanmıştır.

4.1.Algoritmaların Karmaşıklığı

Bir algoritmanın bir problem için ne zaman tatminkar bir çözüm sağladığı önemli bir sorudur. Bu sorunun cevabı şudur. İlk olarak, bir algoritma her zaman doğru cevaplar ortaya koymalıdır. İkinci olarak, etkili ve etkin çalışmalıdır. Bu bölümde algoritmaların etkinliği konusu üzerinde duracağız.

Bir algoritmanın etkinliği nasıl incelenir. Etkinliğin ölçümlerinden birisi, giriş değerleri belirtildiğinde, bu algoritmayı kullanarak belirli problemi çözmek için bilgisayar tarafından kullanılan zamandır. İkinci bir ölçüm, giriş değerleri özel olarak belirtildiğinde algoritmayı tamamlamak için gereken bilgisayar hafıza miktarının ne kadar olduğudur.

Böyle sorunlar **algoritmaların hesaplama karmaşıklığı** ile ilgilidir**.** Boyutu belirli bir problemi çözmek için gereken zamanın analizi, algoritmanın zaman karmaşıklığını içerir. Gerekli olan bilgisayar hafızasının analizi de algoritmanın uzay karmaşıklığını içerir. Bir algoritmanın zaman ve uzay karmaşıklığının ele alınması algoritmaları tamamlayabilmekte önemlidir. Bir algoritmanın bir cevabı bir mikrosaniye, bir dakika yada bir milyar yılda üretip üretemeyeceğini bilmek çok önemlidir. Ayrıca bir problemin çözümünde gerekli hafızanın mevcut olma şartı vardır.

Uzay karmaşıklığının ele alınması algoritmanın tamamlanmasında kullanılan kısmi veri yapıları ile bağlantılıdır. Veri yapıları bu çalışmada yer almayacağından, uzay karmaşıklığı dikkate alınmayacaktır. Özellikle zaman karmaşıklığı üzerinde durulacaktır.

Bir algoritmanın zaman karmaşıklığı, giriş özel bir ölçüye sahip olduğunda algoritmayla kullanılan işlem sayısına göre ifade edilebilir. Algoritmaların zaman karmaşıklığının ölçümünde kullanılan işlemler, tamsayıların karşılaştırılması, toplanması, çarpılması, bölünmesi yada diğer temel işlemler olabilir. Karmaşıklık, temel işlemleri düzenlemek için farklı bilgisayarlarda farklı zamanlara ihtiyaç olduğundan sahip olunan bilgisayara göre gereken işlemlerin sayısına bağlı olarak tanımlanır. Ayrıca, karmaşık bit işlemleri yapılacağı zaman bilgisayar kullanılır. En hızlı bilgisayarlar temel

işlemleri(toplama,çarpma vb..)1 nano saniye, kişisel bilgisayarlar 1 mikrosaniye de yaparlar.Yani, aynı işlemi kişisel bilgisayardan 1000 kez daha hızlı icra ederler.

4.2.Algoritmaların Oluşturulması: Sonlu bir tamsayı dizisinin en büyük elemanını bulmak için bir örnek algoritmayı ele alalım.

Örnek.4.1: Bir dizideki en büyük elemanı bulma problemi oldukça açık olmasına rağmen , algoritma kavramının anlaşılması açısından güzel bir örnek oluşturur . Ayrıca, sonlu bir tamsayı dizisindeki en büyük elemanı bulmayı gerektiren birçok algoritma örneği vardır . Örnek olarak bir üniversitede binlerce öğrencinin katıldığı rekabete dayalı bir yarışmada en yüksek skorun bulunmasına ihtiyaç duyulabilir .veya bir spor organizasyonunda her ay en yüksek reyting alan üyenin tanıtılması istenebilir . Biz sonlu bir tamsayı dizisindeki en büyük elemanı bulmakta kullanılacak bir algoritma geliştirmek istiyoruz .

Bu problemi birçok yolla çözmek için prosedür belirtebiliriz. Şöyle bir çözüm sağlayabiliriz .

Çözüm.4.1 : Aşağıdaki adımları yerine getiriyoruz .

1 . Maksimum değerini geçici olarak dizinin ilk elemanına eşitliyoruz .(Geçici maksimum değeri prosedürün herhangi bir adımında sınanmış en büyük tamsayıdır)

2 . Dizinin bir sonraki elemanıyla geçici maksimum değerini karşılaştırıyoruz, eğer dizinin bir sonraki elemanı geçici maximum değerinden büyükse, geçici maximum değerine bu sayıyı atıyoruz .

3 . Eğer dizide başka sayılar varsa önceki adımları tekrarlıyoruz .

4 . Dizide karşılaştıracak tamsayı kalmadığında duruyoruz.Bu noktada geçici maksimum dizideki en büyük tamsayıdır.

Ayrıca bilgisayar dili kullanılarak algoritma tanımlanabilir. Böyle bir şey yapıldığında sadece o dildeki komutların kullanılmasına izin verilir. Bu da algoritma tanımını karmaşık ve anlaşılması zor bir hale getirir . Üstelik birçok programlama dilinin genel kullanımında , özel bir dil seçmek istenmeyen bir durumdur . Bu yüzden algoritmaları belirtirken özel bir bilgisayar dili yerine pseudocode kullanılır.(Ayrıca bütün algoritmalar ingilizce tanımlanır)

Pseudocode bir algoritmanın ingilizce dil tanımı ve bu algoritmanın programlama diline dönüştürülmesi arasında bir ara basamak oluşturur . Programlama dilinde kullanılan komutlar bu noktada geçici maximum değeri dizinin en büyük

elemanıdır.İyi tanımlanmış işlemleri ve durumları içerir . Herhangi bir bilgisayar dilinde başlangıç noktasında psedeucode tanımlanarak bir bilgisayar programı üretilebilir .

Bu çalışmada pascal programlama dili temel alınarak pseudocode kullanılmıştır . Bununla birlikte pascal ve diger programlama dillerinin söz dizimi kullnılmayacaktır .Daha ilerde iyi tanımlı komutlar bu pseudocotta kullanılabilir . Aşağıda sonlu bir dizideki maximum elemanı bulan pseudocode algoritması tanımlanmıştır .

Algoritma .4.1 : Sonlu bir dizideki maksimum elemanı bulan algoritma

Procedure max(a_1,a_2,.....a_n: integers)

max : =a_1

for i:=2 to n

if max<a_i then max:=a_i

{maksimum dizinin en büyük elemanıdır}

Bu algoritma önce dizinin ilk elemanını maksimum değişkenine atar. For döngüsü dizinin elemanlarını ardışık bir şekilde incelemekte kullanılır. Eğer bir terim max'ın geçerli değerinden daha büyük ise maksimumun yeni değeri olarak atanır.Algoritmalarda genel olarak kullanılan bir çok özellik vardır . Algoritmalar tanımlanırken bunları akılda tutmak yararlıdır .

4.3.Algoritmaların Özellikleri :

Giriş : Bir algoritma açıkça belirtilen bir kümeden giriş değerlere sahiptir .

Çıkış : Bir algoritmanın her bir giriş değerinin kümesinden, çıkış değerlerinin kümesi üretilir . Çıkış değerleri problemin sonucunu içerir

Tanımlılık : Algoritmanın adımları tam olarak tanımlanmalıdır.

Sonluluk: Bir algoritma herhangi bir giriş kümesi için sonlu sayıdaki adımlardan sonra istenilen sonucu üretmelidir .

Etkinlik: Algoritmanın her bir adımı tam olarak ve sınırlı bir zamanda gerçekleşebilmelidir.

Genellik : Prosedür, sadece belli giriş değerleri için değil istenilen formdaki bütün problemler için uygulanabilir olmalıdır.

Sonlu bir tamsayı dizisindeki maksimum elemanı bulan algoritmanın özellikleri şöyle ifade edebiliriz.Algoritmadaki girdi tamsayı dizisidir.Çıktı dizideki en büyük tamsayıdır. Algoritmadaki her adım tam olarak tanımlanmıştır , atamalar sonlu döngü ve şarta bağlı durumlar yer almaktadır.

Algoritma sınırlı bir zamanda karşılaştırma ve atama adımlarının herbirini gerçekleştirmektedir. Sonuç olarak algoritma geneldir ve herhangi bir sınırlı tamsayı dizisindeki maximum sayıyı bulmak için kullanılabilir .

4.4.Arama algoritmaları ve performans analizi

Bu bölümde ve ikinci bölümde değinildiği gibi bir algoritmanın performans analizi o algoritmanın karmaşıklık hesabı anlamındadır. Şimdi sırasıyla Lineer ve İkili arama algoritmalarının performans analizini ele alalım. Karmaşıklık analizinin ikinci bölümde belirtildiği gibi üç çeşidi vardır. Bunlar, En Kötü Durum Analizi, Ortalama Durum Analizi ve Deneysel analizdir.Burada sadece En Kötü Durum Analizi ve Ortalama Durum Analizi ele alınmıştır.Ortalama durum analizinde işlemlerin ortalama sayısı kullanılarak verilen tüm girişlerden problem çözülür. Ortalama durum zaman karmaşıklığı analizi genellikle en kötü durum analizinden daha karmaşıktır. En Kötü Durum Analizi en yaygın kullanılan analizdir.

Sıralanmış bir listedeki bir elemanın yerini bulma problemi pekçok konuda karşımıza çıkar.Sıralanmış kelimelerin listesinin bulunduğu bir sözlükte, kelimelerin hecelenmesini kontrol eden bir program bunun en basit örneklerinden biridir.Bu tür problemler arama problemleri olarak adlandırılır. Bu bölümde arama için birkaç algoritma ele alınacaktır.

Genel olarak arama problemleri şöyle tanımlanır. Farklı $a_1,a_2,.....a_n$ elemanlarının listesinden x'in yerini bulur, ya da listede olmadığını belirler. Bu arama problemi için çözüm dizi içerisinde x'e eşit olan sayının yeridir. Eğer $x=a_i$ ise çözüm 1'dir. x=0 ise dizide yoktur. İlk algoritma sıralı arama algoritmadır . Sıralı arama algoritması x ve a_1'i karşılaştırarak başlar . $x=a_1$ ise çözüm a_1'in yeridir . Eğer, x a_1'e eşit olmazsa x , a_2 ile karşılaştırılır. Çözüm , a_2'nin yeridir. Eğer, x, a_2'ye eşit olmazsa x,a_3 ile karşılaştırılır. Bir eşleme oluncaya kadar listedeki her bir eleman ile x'i karşılaştırma işlemine devam edilir. Çözüm x'e eşit olan terimin yeridir. Eğer, tüm liste arandığında x'in yeri yoksa sonuç 0'dır.

Algoritma.4.2. Sıralı Arama Algoritması

```
Procedure lineer search(x:integer , a1,a2,....,an: distinct integer)
i :=1
while (i≤n and x≠ai)
i:=i+1
if i≤n then location :=i
else location :=0
{location x'e eşit olan terim , eğer "x" 0 ise bulunamadı}
```

Yeni bir arama algoritması incelenecek . Bu algoritma listedeki terimlerin artan bir şekilde sıralı olması halinde kullanılabilir . (Terimler sayı ise küçükten büyüğe , eğer kelime ise alfabetik olarak sıralamalıdır) . Bu ikinci algoritma ikili arama algoritması olarak adlandırılır. Eleman listenin ortasına yerleştirilmiş terimler karşılaştırılarak ilerler . Liste aynı boyutta iki küçük listeye ayrılır veya küçük listelerden birindeki terim sayısı diğerinden bir tane az olabilir. Diğer bölümde ikili arama algoritmasının sıralı arama algoritmasından daha etkili olduğu gösterilecektir . İkili aramanın nasıl çalıştığı aşağıdaki örnekte gösterilmiştir.

19 sayısı için arama yapılması :

1 2 3 5 6 7 8 10 12 13 15 16 18 19 20 22

16 terimden oluşan listeyi 8 terimli iki listeye ayırıyoruz.

1 2 3 5 6 7 8 10 12 13 15 16 18 19 20 22

İlk listedeki en büyük terimle 19 sayısı karşılaştırılıyor. 10<19 olduğundan arama orijinal listedeki 9. Ve 16. Terime sınırlandırılıyor. Daha sonra bu liste 4 terimden oluşan iki listeye ayrılıyor.

12 13 15 16 18 19 20 22

ilk listedeki en büyük terimle 19 karşılaştırılıyor. 16<19 olduğundan arama orijinal listedeki 13. ve 16. terim arasına sınırlandırılıyor. Liste iki parçaya ayrılıyor.

18 19 20 22

İlk listenin en büyük terimi olan 19 sayısı 19'dan büyük olmadığından dolayı arama ilk listedeki 18 ile 19 sayıları ile sınırlandırılıyor. (Orijinal listedeki 13. Ve 14.terim). Sonra iki terimin listesi tek bir terim olarak ayrılıyor. 18<19 olduğundan arama ikinci listeye sınırlanıyor. Liste 14.terimi içeren 19 sayısıdır. Şu an arama bir terime indiriliyor , karşılaştırma yapılıyor ve 19 sayısı orijinal listenin 14. terimine yerleştiriliyor.

4.4.1.Lineer arama algoritması

Algoritma.4.3 : The linear search algorithm.

procedure linear search (x **:** integer , a_1 , a_2 , **.....** , a_n **:** distinct integers)

i **:** 1

while ($i \geq n$ ve $x \neq a_i$)

i **:** = ***i*** + 1

if $i \leq n$ **then** location **:** = ***i***

{ location is the subscript of term that equals x , or is 0 if x is not found }

4.4.1.1: Lineer arama algoritmasının zaman karmaşıklığı(en kötü durum analizi)

Zaman karmaşıklığının ölçümü karşılaştırma sayısı kullanılarak yapılacaktır. Algoritmada çevrimin her adımında, iki karşılaştırma yapılacaktır. Birincide listenin sonuna gelindiğinde ve listenin birinci terimiyle ***x*** elemanı karşılaştırılacaktır. Sonuçta bir dahaki karşılaştırmada çevrimden çıkılır. Dolayısıyla, eğer ***x= a_i , 2i+1*** karşılaştırması kullanıldı. En çok karşılaştırma ***2n***, listedeki olmayan eleman olduğu zamandır. Bu bölümde ***2n*** karşılaştırması ***x*** hesabında kullanıldıysa ***a_i*** değildir ve sonraki karşılaştırmalarda çevrimden çıkılır. ***x*** listede değilse ***2n+2*** karşılaştırmaları toplamı kullanılır. Böylece ***O(n)*** karşılaştırmaları. Bir lineer seçim olmuştur. Bu karmaşıklık analizi **en kötü durum analizidir**. Bu analiz bize bir algoritmada kaç tane işlem yapılırsa çözüme ulaşılacağının garantisini verir.

4.4.1.2. Lineer arama algoritmanın ortalama durum performansının hesabı

Listede x bilindiği zaman mümkün n tip giriş vardır. Listede x ilk terimse, 3 karşılaştırmaya ihtiyaç duyulur. Birinci hesap liste sonuna gelinip gelinmediğini, birinci karşılaştırma x ve ilk terim ve ilk çevrim dışına çıkılır. Eğer listede x ikinci terimse, 2' den fazla karşılaştırmaya ihtiyaç duyulur. Bu yüzden toplam 5 karşılaştırma yapılır. Genelde eğer x i' ninci terimse çevrimden i adımın herbirinde iki karşılaştırma kullanılır ve bir çevrim dışına çıkılır. Bu yüzden 2i+1 karşılaştırmaya ihtiyaç duyulur. Böylece karşılaştırmaların ortalama sayısı :

$$\frac{3+5+7+.....+(2n+1)}{n} = \frac{2(1+2+3.....+n)+n}{n}$$

'dir.

$$1+2+3+.....+n = \frac{n(n+1)}{2}$$

Böylece lineer arama algoritmada kullanılan karşılaştırmaların ortalama sayısı:

$$\frac{2(n(n+1)/2)}{n}+1 = n+2 = O(n)$$

4.4.2.İkili arama algoritması

Algoritma.4.4 : The Binary search algoritma.

procedure binary search (x **:** integer , a_1 , a_2 , **.....** , a_n **:** increasing integers)

i **:** 1 {***i*** is left endpoint of search interval}

j **:** *n* {***j*** is right endpoint of search interval}

while (***i*** < ***j***)

m **:** = [(***i*** + ***j***) / 2]

if $x > a_m$ **then** ***i*** **: =** $m + 1$

else ***j*** **: =** m

end

if $x = a_i$ **then** location **: =** ***i***

else locations **: =** 0

{ location is the subscript of term that equals x , or is 0 if x is not found }

4.4.2.1.İkili arama algoritmanın zaman karmaşıklığının hesabı

Kolaylık için listede $n=2^k$ eleman olduğunu farz edelim. ***k*** pozitif tamsayı listede $k=\log n$ geçmekte, 2' nin kuvveti değildir. 2^{k+1} elemanla daha büyük bir liste dikkate alınabilir. Burada $2^k<n<2^{k+1}$ 'dir.

Algoritmanın her bölümünde , i ve j ilk terimin yerinde ve en son terimle sınırlandırılmıştır. Karşılaştırmada bir terimden daha fazla terim sınırlandırılamayacağı gözlenir. Eğer i<j , sınırlandırılmış listenin orta teriminden, x' in daha büyük olup olamayacağı hesaplanır.

İlk kısımda, seçim 2^{k-1} terimle sınırlandırılarak yapılır. Şimdiye kadar 2 karşılaştırma kullanıldı. Bu prosedüre devam edersek, bir listede her bölgenin sınırlandırılmasında iki karşılaştırma kullanarak birçok terimin yarısıyla bir liste seçiminde.

Diğer durumlarda, liste 2^k elemana sahip olduğu zaman algoritmanın ilk kısmında iki karşılaştırma kullanılır. İkiden fazla seçim yapılacağı zaman 2^{k-1} elemanla liste yapılır. Sonuç olarak listede bir terim ayrılacağı zaman, birinci karşılaştırmada takip eden terimler atılır ve bir daha karşılaştırmada eğer terim x ise hesaplamada kullanılır.

Böylece,en çok 2k+2=2logn+2 karşılaştırmayla, 2^k elemana sahip seçilen bit listede ikili bir ikili seçimi gerçekleştirdik. Sonuç olarak bir ikili seçim yapılırken O(logn) karşılaştırma yapılır. Bu analiz bize göstermiştir ki algoritmada ikili seçim lineer seçimden daha etkindir.

4.4.3.Algoritmaların Karmaşıklığı için Kullanılan Terminoloji ve İşlem Zamanı

Tablo4.1.algoritmaların zaman karmaşıklığının tanımında kullanılan terminolojiyi gösterir. Örnek olarak, $O(n^b)$ zaman karmaşıklığıyla belirtilen bir algoritma dendiğinde polinom karmaşıklıktan bahsedilir. Lineer arama algoritması, lineer (en kötü yada ortalama durumu) karmaşıklığa ve ikili arama algoritması logaritmik (en kötü durum) karmaşıklığa sahiptir. 'O' sembolü bir algoritmanın zaman karmaşıklığının tahminini ifade eder. Bununla beraber bilgi zamanın etkin kullanımında 'O' zaman karmaşıklığının tahmini direk yapılamaz. Birinci neden bir büyük 'O' tahmini f(n)=O(g(n)), burada f(n) bir algoritmanın zaman karmaşıklığını ve g(n) bir referans fonksiyondur. n>k olduğu zaman f(n)<Cg(n) dir. C ve k

sabitlerdir. Eşitsizlikte C ve k bilinmeksizin bu tahmin kullanılan işlemlerin sayısında bir üst sınır hesaplanamaz.Bununla beraber eğer tüm işlemler bilgisayar kullanılarak bit işlemler şeklinde yapılacaksa problem çözümünde kullanılacak algoritmanın zaman hesabı yapılabilir.

Tablo4.1.algoritmaların zaman karmaşıklığının tanımında kullanılan terminoloji

Karmaşıklık	Terminoloji
O(1)	*Sabit Karmaşıklık*
O(log n)	*Logaritmik Karmaşıklık*
O(n)	*Lineer Karmaşıklık*
O(n logn)	*nlogn*
$O(n^b)$	*Polinom Karmaşıklık*
$O(b^n)$, $b>1$	*Üstel Karmaşıklık*
O(n!)	*Faktöriyel Karmaşıklık*

Tablo 4.2 bit işlem sayısını göstermektedir. 10^{100} yıldan büyük zamanlar asterisk (*) ile gösterilmiştir. Bu tablo yapılırken her bit işlemi 10^{-9} saniye alınmıştır. Gelecekte bit işlem hızı gelişen bilgisayar teknolojisiyle daha artacaktır.

Problem çözümünde nasıl bilgisayara ihtiyaç duyduğumuz önemli bir noktadır. Örnek olarak bir algoritma 10 saat ise bu problemin çözümü için bilgisayar zamanına ve para harcamaya değer. Fakat eğer algoritma 10 milyar yıl ise bu algoritmanın sonuçlanması olanaksızdır. Gelişen bilgisayar teknolojisi hızları ve hafıza kapasitelerini artırmıştır. Özellikle paralel işlem özelliği algoritmaların ve problem çözümlerini kolaylaştırmıştır.

Tablo.4.2.Algoritmalar tarafından kullanılan zaman

Problem Uzunluğu	Kullanılan bit işlem					
n	*logn*	*n*	*nlogn*	n^2	2^n	*n!*
10	$3x10^{-9}$	10^{-8}	$3x10^{-8}$	10^{-7} s	10^{-6}	$3x10^{-3}$s
10^2	$7x10^{-9}$	10^{-7}	$7x10^{-7}$	10^{-5} s	$4x10^{13}$yıl	*
10^3	$1x10^{-8}$	10^{-6}	$1x10^{-5}$	10^{-3} s	*	*
10^4	$1.3x10^{-8}$	10^{-5}	$1x10^{-4}$	10^{-1} s	*	*
10^5	$1.7x10^{-8}$	10^{-4}	$2x10^{-3}$	10 s	*	*
10^6	$2x10^{-8}$	10^{-3}	$2x10^{-2}$	17 dakika	*	*

4.5.Denkliğin Uygulamaları

Sayı teoresi kavramı içine giren **Denklik** pek çok alanda kullanılır. Bunların en önemlileri :

1.Hafıza bölgelerini bilgisayar dosyalarına atamak için benzerlik kullanımı .

2.Rasgele sayıların üretimi .

3.Modüler aritmetiğe dayalı şifreleme sistemi .

4.5.1.Hashing Fonksiyonları

Okuldaki her bir öğrenci için kayıtları devam ettiren merkezi bir bilgisayarımız olduğunu düşümelim.Bu öğrencilerin kayıtlarına hızlı bir şekilde yeniden ulaşmak için ne kadarlık bir hafıza bölgesi atanmalıdır.Böyle bir problemi çözmek için uygun bir hashing fonksiyonu seçilmiştir. Her bir öğrencinin kaydı bir anahtar kullanılarak tanımlansın. Örneğin bu anahtaröğrencinin sosyal güvenlik numarası olarak seçilebilir. Buna göre "h" hashing fonksiyonu "k" anahtarına sahip olan h (k) hafıza bölgesine atar.

Pratikte pek çok farklı hashing fonksiyonları kullanılır. Bunun en yaygın kullanımı h(k)=k(modm)olanıdır. Burada m elde edilebilir hafıza bölgesinin sayısıdır. Hashing fonksiyonları kolayca değerlendirilebilir. Bu sayede verilere hızlı bir şekilde ulaşılabilir. h(k)=k mod m hashing fonksiyonları bu gereksinimleri karşılar. H(k)'yı bulmak için k'nın m'ye bölümünden kalanı hesaplamamız yeterli olacaktır. Ayrıca bütün hafıza bölgelerine ulaşmak bu fonksiyonla mümkündür.

Örnek.4.2: 111 hafıza bölgesine sahip olalım. Sosyal güvenlik numarası 064212848 olan öğrenci h(064212848)=064212848 mod 111=14 hashing fonksiyonu yardımıyla 14.hafıza bölgesine atanır.Sosyal güvenlik numarası 037149212 olan öğrenci ise benzer yolla

h(037149212)=037149212 mod 111=65

hashing fonksiyonu yardımıyla 65.hafıza bölgesine atanır.Hashing fonksiyonu birebir olmadığında (hafıza bölgesinden daha çok anahtar olduğunda) birden fazla dosya bir hafıza bölgesine atanır. Bu olduğunda çakışma oluştu denir. Hashing fonksiyonları yardımıyla atanan son hafıza bölgesinde ilk boş alanına atanır. Örneğin önceden 2 atama yapıldıktan sonra sosyal güvenlik numarası 107405723 olan öğrencinin kaydını 15.hafıza bölgesine atar.

h(107405723)=107405723 mod 111=14 olduğu gözönüne alınır ve 14 nolu hafıza bölgesinin 064212848 nolu öğrenciye ait olduğu bilindiğinden müteakip ilk boş hafıza bölgeside 15 olduğundan buradaki çakışma önlenmiş olur.

4.5.2.Yarı Rastgele Sayılar

Rasgele seçilen sayılar bilgisayarların simulasyonunda gereklidir. Rasgele seçilen sayıların özelliklerine sahip üretilmiş sayılar için farklı metodlar mevcuttur. Sistematik metodlar yardımıyla üretilen ayılar gerçekten rasgele olmadığından olayı bu sayılara yalancı rasgele sayılar denir.

Yalancı sayıların üretiminde kullanılan en yaygın işlem lineer benzerlik metodudur. Bu metodta 4 tamsayı seçilir. Bunlardan m modülü , a çarpanı , c aralığı ,(x0)? Asıl sayıyı göstermek üzere 2≤a<m , 0≤c<m ve 0≤xn<m ? şeklindedir.Bütün n'ler için 0≤ xn<m aralığında yalancı rasgele sayıların {xn} dizisini üretmek için (xn+1=(axn+c)) mod m benzerliği ardışık olarak kullanılır. Bu bağıntı aynı zamanda indirgeme bağıntının tanımıdır. İlerki bölümlerde bu bağıntıdan bahsedilecektir. Birçok bilgisayar uzmanı 0-1 arasındaki yalancı rasgele sayıların üretimine ihityaç duyar.Bu sayıları modüller yardımıyla bir lineer benzerlik bağıntısı kurup üretilen sayıları bölerek elde ederiz. Yani xn/m sayılarını kullanırız.

Örnek olarak m=9 , a=7 , c=4 ve x_0=3 seçilerek yarı rastgele sayıların kümesi aşağıdaki gibi üretilir.

Örnek.4.3. $x_9=x_0$ ve her bir terim bir önceki terime bağlı olduğundan 3 ,7 ,8 ,6 ,1 ,2 ,0 , 4, 5 ,3 , 7 , 8 , 6 ,1 , 2 , 0 , 4 , 5 , 3..... dizisi üretilir. Bu dizi 9 tane farklı sayının sonsuz tekrarından ibarettir. Pek çok bilgisayar bunu kullanır. En çok kullanılan * ifadesinde c'nin 0 olduğu bağıntıdır. Buna pure multiplicative genarator denir. Örneğin 231-2 ? modülü ve 7^5 =16807 çarpanı ile pure multiplicative generator yaygın olarak kullanılır. Bu değerler ile 2^31 −2 tane sayı tekrarsız olarak üretilir.

4.5.3.Kriptoloji (Şifreleme)

Benzerlikler kesikli matematik ve bilgisayar biliminde çok yaygın kullanılır. Kongrüansın en önemli uygulamalarından biri gizli mesajları içeren kriptolojilerdir. Kriptoloji' yi kullananlardan biri Julius Caesor'dır. Caesor alfabedeki (ingiliz alfabesi) her bir harfi üç harf ileri kaydırarak gizli mesajlar oluşturmuştur. Bu şifrelemenin bir örneğidir. Yani gizli bir mesaj yapma işlemidir. Caesor'ın bu şifreleme işlemini matematiksel olarak ifade etmek için alfabedeki pozisyonuna bağlı olarak her bir harfi 0'dan 25'e bir tamsayıya eşliyoruz. Yani A→0 , K→10 , Z→25 şeklinde. Böylece Caesor'ın şifreleme metodu p≤25 olmak üzere negatif olmayan bir p tamsayısı

yardımıyla f(p) =(p+3) mod 26 şeklinde tanımlanır. Burada f(p) bir tamsayı olup {0, 1, 2,3, ...,25} kümesinden değer alır.

Örnek.4.4: **"MEET YOU IN THE PARK"** mesajından Caesar'ın şifreleme yöntemini kullanarak gizli mesaj üretebiliriz.

Çözüm.4.4: İlk olarak mesajın harflerini numaralara çeviriz.

12 4 4 19 24 14 20 8 23 19 7 4 15 0 17 10

Her numarayı f(p)=(p+3) mod 26 fonksiyonuyla değiştiririz.

15 77 22 1 17 23 11 16 22 10 7 18 3 20 13

Numaraları harflere çevirerek şifrelenmiş mesaj PHHW BRX LQ WKH SDUN olarak üretilir. Caesar'ın şifreleme yöntemiyle şifrelenmiş gizli mesajdan orijinal mesajı elde etmek için f^-1 fonksiyonu kullanılır. F^-1 fonksiyonu {0,1,2.....,25} sayılarından bir p tamsayısı gönderir.

F^-1(p)=(p-3) mod 26

Orijinal mesajı bulmak için alfabedeki her bir harf üç harf geriye kaydırılır. Şifrelenmiş mesajdan orijinal mesaj belirleme işleminede şifre çözme(description) denir. Caesor'ın şifreleme yöntemini genelleştirebiliriz. Bunun basit örneği ;

f(p)=(p+k) mod 26 olup bu tip şifreye kaydırma şifre denir. Böyle bir şifreyi çözmek içinde f^-1 (p)=(p-k) mod 26 kullanılır. (Burada k alfabedeki ilerletilecek sayıdır) Bununla birlikte Caesor'ın bu şifreleme yöntemi çok güvenli değildir. Bunun güvenirliliğini arttırmak için f(p)=(ap+b) mod 26 fonksiyonu seçilir. Burada a ve b tamsayılardır.

Örnek.4.5: f(p)=(7p+3) mod 26 fonksiyonu kullanıldığında k harfi hangi harfe karşılık gelir.

Çözüm.4.5: k=10.harf ise f(10)=(7.10+3) mod 26 =21=v

Böylece k'ya karşılık gelen v harfi bulunur.

4.5.4.Çin Kalan Teoremi

Bu sistemleri çok eski Çin ve Hindu matematikçilerinin yazdığı kelime bulmacalarında bulabiliriz. Böyle bir bulmaca örneği aşağıda verilmiştir:

1. yy.da Çinli matematikçi, Sun-Tsu soruyor:3' e bölündüğü zaman 2 kalanını veren, 5' e bölündüğü zaman 3 kalanını veren, 7' ye bölündüğü zaman 2 kalanını veren sayı kaçtır? Bu bulmacanın çözümü;

$x \equiv 2 \pmod 3$,

$x \equiv 3 \pmod 5$,

$x \equiv 2 \pmod 7$,

sistemin çözümü ile eşdeğer olup Çin kalan teoreminden sonra verilecektir.

Çin kalan teoremi, adını Çinlilerden miras kalan, lineer denklik sistemlerini içeren bulmaca problemlerden almıştır. Lineer denklik sistemleri modül alma işlemine dayanan ve ikişerli aralarında asal olan sayılardan oluşur. Böyle bir sistemin çözümü tektir.

Teorem.4.1: Çin Kalan Teoremi: $\mathbf{m_1,m_2,.....m_n}$ ikişerli aralarında asal pozitif tamsayılar olsun. Buna göre;

$\mathbf{x \equiv a_1 \pmod{m_1}}$,

$\mathbf{x \equiv a_2 \pmod{m_2}}$,

.

$\mathbf{x \equiv a_n \pmod{m_n}}$

sistemi $\mathbf{modm= m_1.m_2.....m_n'}$ e göre tek bir çözüme sahiptir. $0 \leq x < m$ aralığında bir x çözümü vardır. Diğer tüm çözümler, bu tek çözümdeki m modülüne denkdir.

Ispat.4.1: Teoremi kanıtlamak için, bu çözümün varolduğunu ve **m** modülüne göre tek olduğunu göstermemiz gerekir. Bunun için;

$\mathbf{k=1,2,....n}$ için $\mathbf{M_k = m / m_k}$ alalım. Burada, $\mathbf{m_k}$ hariç modüllerin çarpımı $\mathbf{M_k}$' dır.

$\mathbf{m_i}$ ile $\mathbf{m_k}$, $\mathbf{i \neq k}$ iken 1' den büyük hiç ortak çarpana sahip değillerse $\mathbf{gcd(m_k,M_k)=1}$' dir. Sonuç olarak, modül m_k ' ya göre, M_k' nın tersi y_k ;

$\mathbf{M_k.y_k \equiv 1 \pmod{m_k}}$ olacak şekilde mevcuttur. Buna göre, sistemin tek çözümü

$\mathbf{x \equiv a_1.M_1.y_1 + a_2.M_2.y_2 + ...+a_n.M_n.y_n}$

şeklinde tanımlanır. x çözümünün similtane çözümü olduğunu gösterelim.

Bunun için, $j \neq k$ iken, $M_j \equiv 0 \pmod{m_k}$ olduğundan, bu toplamdaki k. terim hariç bütün terimler modül m_k ' ya göre 0' dır. $M_k.y_k \equiv 1 \pmod{m_k}$ olduğundan k=1,2,....n için

$\mathbf{x \equiv a_k.M_k.yk \equiv a_k \pmod{m_k}}$

olduğu görülür. Böylece x istenen tek çözümdür.

Aşağıdaki örnekler, teorem.4.1'in ispatındaki oluşumu denklik sistemlerin çözümlerinde nasıl kullanılacağını açıklar. Sun-Tsu bulmacasında ortaya çıkan sistemleri örnek.4.6'daki gibi çözeceğiz.

Örnek.4.6: Öncelikle m=3.5.7 = 105 oluşturulur.Sonra,

$M_1 = m/3 = 35$, $M_2 = m/5 = 21$, $M_3 = m/7 = 15$ eşitliklerini yazdıktan sonra modül m_k ' ya göre M_k ' nın tersi y_k ' yı bulmak için $M_k.y_k \equiv 1$ denkliğinden yararlanırız. Buna göre;

$M_1.y_1 \equiv 1 \pmod 3$ ise $35.\ y_1 \equiv 1 \pmod 3$ ve buradan $y_1 = 2$

$M_2.y_2 \equiv 1 \pmod 5$ ise $21.\ y_2 \equiv 1 \pmod 5$ ve buradan $y_2 = 1$

$M_3.y_3 \equiv 1 \pmod 7$ ise $15.\ y_3 \equiv 1 \pmod 7$ ve buradan $y_3 = 1$ bulunur.

Örneğin; birinci denklikte 35 ile öyle bir sayıyı çarpalım ki sonucun mod 3' e göre kalanı 1 olsun. Diğer denklikleri de bu şekilde ifade edebiliriz. y değerleri bulunduktan sonra sistemin aşağıdaki gibi x çözümüne ulaşırız:

$x \equiv a_1.M_1.y_1 + a_2.M_2.y_2 + a_3.M_3.y_3 = 2.35.2 + 3.21.1 + 2.15.1 \pmod{105}$

$= 233 \equiv 23 \pmod{105}$.

Sonuç olarak, 23 en küçük pozitif tam sayısı bir similtane çözümdür. 3' e bölündüğünde 2 kalanını veren, 5' e bölündüğü 3 kalanını veren, 7' e 2 kalanını veren en küçük pozitif tam sayı 23 olarak bulunur.

4.5.5.Büyük Sayılarla Bilgisayar Aritmetiği

Aralarında ikişerli asal ve 2' den büyük veya eşit tam sayılar $m = m_1, m_2, m_3,....m_n$ olsun. Çin kalan teoremine göre gösterebiliriz ki; m_i , i=1,2,....n $0 \leq a < m$ aralığında olan bir a tamsayısı (a mod m_1, a mod m_2, ...a mod m_n) olacak şekilde ifade edilir.

Örnek.4.7: 12' den küçük pozitif tam sayıların 3 ve 4 ile bölümünden kalanları nokta çiftleri ile bulunuz:

Çözüm.4.7: 3 ve 4' e bölündüğü zaman her sayıdan kalanları bularak çözüme ulaşabiliriz.

0 = (0,0)	4 = (1,0)	8 = (2,0)
1 = (1,1)	5 = (2,1)	9 = (0,1)
2 = (2,2)	6 = (0,2)	10 = (1,2)
3 = (0,3)	7 = (1,3)	11 = (2,3)

Örneğin; 6 sayısını ele alalım. Bu sayı 3' e bölündüğünde 0 kalanını, 4' e bölündüğünde 2 kalanını verecektir. Bu şekilde 12' e kadar olan sayıların sırasıyla 3 ve 4' e bölümünden kalanları bulunarak, yukarıdaki gibi bir çözüm elde edilir.

Büyük tamsayılarla aritmetik uygulaması için, her m_i sayısı 2' den büyük olmak koşuluyla, $i \neq j$ iken gcd(i,j) = 1 olduğunda $m_1, m_2,...m_n$ modüllerini seçeriz ve $m = m_1.m_2.....m_n$, gerçekleştirmek istediğimiz aritmetik işlemlerin sonuçlarından daha büyüktür.

İlk olarak, modülü seçeriz. m_i' ye, i=1,2,...n' e kadar sayıların bölümünden kalan n adet sayı bulunur. Sayıların işlem bileşenlerinde yerine getirilmesiyle büyük tamsayılarla aritmetik işlemleri gerçekleştirebiliriz. Sonuçta her bileşenin değerini hesaplarız. m_i, i = 1,2,....n modülünde denk bir n sistemini çözerek değerleri tekrar elde ederiz. Büyük tam sayılarla uygulanan aritmetik metodun önemli birkaç özelliği vardır. Birincisi, genellikle bilgisayarda uygulanan tamsayılardan daha geniş tamsayılarla aritmetik uygulaması kullanılabilir. İkincisi, farklı modüllerle ilgili olarak hesaplamalar yapılabilir, buna paralel olarak işlem hızı artar.

Örnek.4.8: Büyük tamsayılarla yapılan aritmetikten daha hızlı, kesin bir yöntem üzerinde 100' den küçük tamsayılarla bir işlem uygulaması varsayalım. 100' den küçük ardışık asal tamsayıların modülünden kalanların kullanıldığı sayıları gösterirsek, 100' den küçük tamsayılar için hemen hemen tüm hesaplamalarımızı sınırlayabiliriz. Örneğin; 99, 98, 97 ve 95' in modülünü kullanabiliriz. (Bu sayılar, 1' den başka ortak çarprnı bulunmayan ikişerli aralarında asal sayılardır.)

Çin kalan teoremi ile, 99.98.97.95 = 89,403,930' dan küçük tüm pozitif tamsayılar bu 4 modüle sırasıyla bölündüğünde, kalanları olarak tek gösterilebilir. Örneğin;

123684 mod 99 = 33

123684 mod 98 = 8

123684 mod 97 = 9

123684 mod 95 = 89

olduğu için 123684 sayısını (33,8,9,89) şeklinde de ifade edebiliriz. Buna denk, 413456 sayısını (32,92,42,16) gibi de ifade edebiliriz. 123684 ve 413456 sayılarının toplamını bulmak için 4 sayı yerine iki tamsayıyla çalışırız. 4 bileşen ekler ve her bileşenin, uygun modüllerini bularak indirgeriz.

(33,8,9,89) + (32,92,42,16)= (65 mod 99, 100 mod 98, 51 mod 97, 105 mod 95)

= (65,2,51,10).

Bu toplamı bulmak için;

$x \equiv 65 \pmod{99}$

$x \equiv 2 \pmod{98}$

$x \equiv 51 \pmod{97}$

$x \equiv 10 \pmod{95}$

sistemi Çin kalan teoremi yardımıyla bulunur.537140 , 89403930 sayısından küçük, bu sistemin tek pozitif çözümüdür. Sonuç olarak, 537140 toplamdır. Sayıları (65,2,51,10) olarak çevirdiğimiz zaman 100' den büyük sayılarla işlem yapmak zorundayız.

Büyük sayılarla işlem yapmak için, modülün iyi seçimleri, k pozitif tamsayı olmak şartıyla, 2^k -1 şeklindeki tam sayılar kümesidir. Çünkü, bu tür sayılarla ikili işlem modülü yapmak ve aralarında ikişerli asal sayılar kümesini bulmak kolaydır.

İkinci sebep ise $\gcd(2^a-1,2^b-1) = 2^{\gcd(a,b)} - 1$ olan gerçek bir denkliktir. Örneğin; bilgisayarımızda 2^{35} ' ten küçük sayılarla işlem yapabileceğimizi varsayalım. Fakat, büyük sayılarla çalışmak özel prosedürler gerektirir. Büyük sayılarla işlem uygulaması için 2^{35}' ten küçük ikişerli aralarında asal sayı modülünü kullanabiliriz.

Mesela; 2^{35}-1, 2^{34}-1, 2^{33}-1, 2^{31}-1,2^{29}-1 ve 2^{23}-1 sayıları ikişerli aralarında asal sayılardır. Bu altı modülün sonuçları 2^{184}' ü aşacağı için, 2^{184} kadar büyük sayılarla işlemi, bu altı modülden hiçbiri 2^{35}' i geçmeyecek şekilde aritmetik modülü yaparak düzenleyebiliriz.

Bir sayının asal olduğuna karar vermek için daha etkili yollar var mıdır? Eski Çin matematikçisi sadece $2^{n-1} \equiv 1 \pmod{n}$ ise n' nin asal olduğuna inanırdı.

Eğer bu doğruysa, etkili bir test uygulanacaktır. Neden bir sayının asal olduğuna karar vermek için kullanılabilen bu denkliğe inandılar? Birincisi, onlar n asal olduğu zaman denkliğin tuttuğunu gözlediler. Örneğin, 5 asaldır ve $2^{5-1} = 2^4 = 16 \equiv 1 \pmod{5}$.

İkincisi, denkliğin tuttuğu çok karmaşık olan bir n sayısı bulamadılar. Eski Çin matematikçileri kısmen haklıydı. n asal olduğu zaman denkliğin tuttuğunu doğru düşündüler. Fakat, denklik tutarsa n' in gerekli asal olduğu sonucuna varmaları yanlıştı.

Büyük Fransız matematikçisi Fermat, n asal olduğu zaman denkliğin tuttuğunu gösterdi. Bunu takiben daha genel sonuçlara da ulaştı.

(Pierre de Fermat (1601-1665). 17. yy.da önemli matematikçilerden biri olan Fermat profesyonel bir hukukçuydu. Tarihte çok ünlü amatör bir matematikçidir. Fermat kendi matematik buluşlarını yayınladı. Buna rağmen, diğer matematikçilerle çalışmaları da olduğunu biliyoruz. Fermat analitik geometriyi ve hesaplamanın birkaç temel mantığını geliştiren bir bilim adamıdır. Fermat, Pascalla temel matematik teorilerini vermiştir. Fermat matematikte çözülmeyen ünlü problemleri formüle etmiştir. n 2' den büyük bir tamsayı olduğunda, önemsiz pozitif tamsayı çözümlerine sahip olan $x^n+y^n = z^n$ eşitliğiyle yarar sağladı.)

Teorem.4.2.Fermat Teoremi: p asalsa ve a, p ile bölünmeyen bir tamsayı ise,

$$a^{p-1} \equiv 1 \pmod{p}$$

Bunun dışında, sahip olduğumuz her a sayısı için,

$$a^p \equiv a \pmod{p}$$

Maalesef, $2^{n-1} \equiv 1 \pmod{n}$ gibi karmaşık bir n tamsayısı vardır. Böyle tamsayılara yarı asallar denir.

Örnek.4.9: $2^{340} \equiv 1 \pmod{341}$ gibi gösterildiği ve (341=11.31) olarak tanımlandığı zaman 341 bir yarı asal sayıdır.

4.6.Bölüm ve Denklik Bağıntılarının Karmaşıklığı ile İlgili Teoremler ve Örnekleri

Varsayalım ki bir algoritma n- boyutlu bit problemi a tane alt probleme bölsün ve bu alt problemlerin herbirinin boyutu $^{n}/_{b}$ olsun. (n , b'yi böler varsayalım. Gerçekte , küçük problemler eşit boyutlu alt problemlere sahiptirler ve bu $^{n}/_{b}$ sayısı en yakın tamsayıya dönüştürülür.

n boyutlu problem küçük problemlere ayrıldığında gerekli olan ekstra işlemlerin sayısına **g(*n*)** diyelim. Böylece , ***f*(n)** bir problemi çözmek için gereken işlemlerim sayısını ifade ettiğinde;

$$f(n) = a.f(^{n}/_{b}) + g(n)$$

bağıntısı sağlanır ve bu bağıntı **bölüm ve denklik bağıntısı** olarak adlandırılır.

Örnek.4.10 : ikili arama algoritmasında;

$$f(n) = f\left(\frac{n}{2}\right) + 2 \qquad \text{(n çift)}$$

Örnek.4.11: n-elemanlı bir dizide bu dizinin maksimum ve minimum elemanını bulan algoritmada;

$$f(n) = 2.f\left(\frac{n}{2}\right) + 2 \qquad \text{(n çift)}$$

4.7.İndirgeme Bağıntılarının Çözümü

k.dereceden sabit katsayılı Lineer homojen bir indirgeme bağıntısı;

$a_n = c_1\, a_{n-1} + c_2\, a_{n-2} + \ldots\ldots\ldots\ldots + c_k\, a_{n-k}$

$c_1, c_2, \ldots\ldots\ldots, c_k$ reel sayılar ve $c_k \neq 0$ ' dır.

Başlangıç koşulları ;

$H_n = 2\, H_{n-1} + 1$ homojen değil.

$B_n = n.\, B_{n-1}$ sabit katsayılı değil.

$a_{n-1} + a_{n-2}$ Lineer değil.

Bu bağıntılar sistematik olarak çözüldüğü ve problemlerin modellemesinde sıklıkla ortaya çıktığından dolayı seçilmiştir.

Çözümü : $a_n = r^n$, $a_n = c_1\, a_{n-1} + c_2\, a_{n-2} + \ldots\ldots\ldots\ldots + c_k\, a_{n-k}$

İndirgeme bağıntılarının çözümü ise ;

$$\mathbf{r^n = c_1.r^{n-1} + c_2.r^{n-2} + \ldots\ldots + c_k.r^{n-k}}$$

yazılır. Her iki taraf r^{n-1} 'ya bölünürse

$$\mathbf{r^k - c_1.r^{k-1} - c_2.r^{k-2} - \ldots\ldots - c_{k-1}.r - c_k} = 0 \text{ elde edilir.}$$

Bu denkleme karakteristik denklem köklerinede karakteristik denklem kökleri denir.

Teorem.4.3 : c_1 ve c_2 reel sayılar olsun.

$r^2 - c_1 r - c_2 = 0$ 'ın r_1 ve r_2 iki farklı kökü olsun.

$a_n = c_1 a_{n-1} + c_2 a_{n-2}$

$a_n = \alpha_1 r_1^n + \alpha_2 r_2^n$ $\quad n = 0,1,2,.....(\alpha_1, \alpha_2$ sabitler)

için $\{a_n\}$ dizisi çözümdür.

Teorem.4.4 : c_1 ve $c_2 \in \mathbb{R}$, $c_2 \neq 0$ olsun.

$r^2 - c_1 r - c_2 = 0$ $\quad r_1 = r_2 = r_0$ olsun.

Eğer, $a_n = \alpha_1 r_0^n + \alpha_2 . n r_0^n$ $\quad n = 0,1,2,.....(\alpha_1, \alpha_2$ sabitler)

İse $a_n = c_1 a_{n-1} + c_2 a_{n-2}$

İndirgeme bağıntılarının çözümü $\{a_n\}$ dizisidir.

Teorem.4.5: $\mathbf{c_1, c_2 , ... c_k}$ reel sayılar olsun. ve

$\mathbf{r^k - c_1 r^{k-1} - ..... - c_k = 0}$ karakteristik denklemi r_1 , r_2 ,....., r_k farklı köklere sahip olsun.

Eğer, $a_n = \alpha_1 r_1^n + \alpha_2 . r_2^n + ... + \alpha_k . r_k^n$ $\quad n = 0,1,2,.....(\alpha_1, \alpha_2 , \alpha_k$ sabitler)

İse $a_n = c_1 a_{n-1} + c_2 a_{n-2} + ... + c_k a_{n-k}$

Bağıntısının çözümü $\{a_n\}$ dizisidir.

Genelleme : Eğer , n, b'yi böler ve $n = b^k$, $k \in Z^+$ ise

$$f(n) = a \,.\, f(n/b) + g(n)$$

$$= a^k \,.\, f(n/b^k) + \sum_{j=0}^{k-1} a^j .g(n/b^j)$$

$$\frac{n}{b^k} = 1 \Rightarrow f(n) = a^k \,.\, f(1) + \sum_{j=0}^{k-1} a^j .g(n/b^j)$$

Teorem.4.6 : f artan bir fonksiyon , n , b' yi bölsün $a \geq 1$ ve b, 1'den büyük bir tamsayı ve c pozitif bir reel sayı olmak üzere ;

$f(n) = a \,.\, f\left(\frac{n}{b}\right) + c$ sağlansın . Bu durumda ;

$$f(n) = \begin{cases} O\left(n^{\log_b a}\right) & , a > 1 \\ O(\log n) & , a = 1 \end{cases}$$

Örnek.4.12 : $f(n) = 5 . f(n/2) + 3$ ve $f(1) = 7$ olsun. Buna göre, $f(2^k)$, $k \in Z^+$'yı bulunuz ve $f(n)$ için f artan bir fonksiyon iken bir tahmin yapınız?

Çözüm : $\left.\begin{matrix} a=5 \\ b=2 \\ c=3 \end{matrix}\right\}$ $n = 2^k$

$$f(n) = a^k\left[f(1)+\frac{c}{a-1}\right]-\frac{c}{a-1} \quad = \quad 5^k\left[7+\frac{3}{4}\right]-\frac{3}{4}$$

$f(n) = 5^k\left(\frac{31}{4}\right)-\frac{3}{4}$ ve örtendir. $a>1 \Rightarrow f(n) = O(n^{\log b^a}) = O(n^{\log 5})$

Örnek.4.13 : İkili arama algoritması: $f(n) = f(\frac{n}{2}) + 2$

a = 1 , b = 2 , c = 3 → ***f* (n) = O (Log n)**

Örnek.4.14 : $f(n) = 2.f(\frac{n}{2}) + 2$ a = 2, b = 2, c = 2 → a > 1

$f(n) = O(n^{\log b^a}) = O(n^{\log 2^2}) = O(n)$

4.8. İndirgeme Bağıntıları ile Modelleme

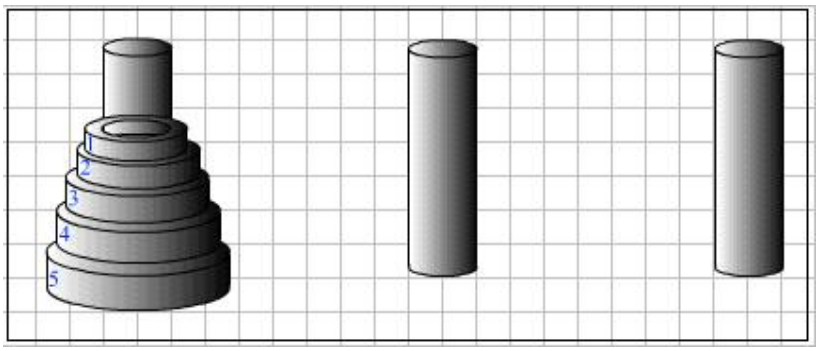
4.8.1.Hanoi Kulesi

Hanoi Kulesi 19–yy'ın meşhur bir puzzle'ı olup üç tane çubuktan oluşan bir düzlemde bir çubuktan diğer bir çubuğa tek bir hareketle farklı ölçülerdeki diskleri en küçük disk en üstte en büyük disk en altta olmak şartıyla sırayla dizmektir.

n tane diskten oluşan Hanoi Kulesi problemini ele alalım. n tane diskin hareket sayısını bir defada bir diski hareket ettirmek şartıyla H_n olarak tanımlayalım.Buna göre,

{ H_n }dizisi için bir indirgeme bağıntısı bulalım.

Aşağıdaki üç tane çubuğu göz önüne alalım.(Şekil.4.1)



Çubuk.1 Çubuk.2 Çubuk.3

Şekil4.1'de de görüldüğü gibi 1. çubukta n tane diskimiz var. Bu **n** tane diskten üsteki **(n – 1)** tanesini birer hareketle 2. çubuğa transfer ediyoruz. Bunun için $\mathbf{H_{n-1}}$ hareket yapıyoruz.

Yani, $\mathbf{H_1 = 1}$ kabul ediyoruz. Böylece en büyük disk 1. çubukta kalıyor. Bundan sonra 1. çubuktaki en büyük diski $H_1 = 1$ olduğu hatırlanarak1 hareketle 3. çubuğa taşıyoruz ve 2. çubuktaki (n-1) diski yine H_{n-1} hareketle 3. çubuktaki en büyük diskin üstüne taşıyoruz.

Sonuç olarak n tane diski bir çubuktan diğer bir çubuğa en küçük disk en üstte en büyük disk en altta olmak ve bir hareketle bir disk taşınmak şartıyla;

$\mathbf{H_n = 2H_{n-1} + 1}$ şeklinde ifade edebiliriz.

Şimdi bu son ifadeden hanoi kulesi problemi için bir indirgeme bağıntısı bulmak için yine açıkladığımız mantığa göre H_{n-1}'den H_1'e kadar olan degerleri aşağıdaıki gibi yerlerine yazarsak ;

$$\mathbf{H_n = 2H_{n-1} + 1}$$
$$\mathbf{= 2(2H_{n-2} + 1) + 1 = 2^2.H_{n-2} + 2 + 1}$$
$$\mathbf{= 2^2(2H_{n-3} + 1) + 1 = 2^3.H_{n-3} + 2^2 + 2^2 + 2 + 1}$$
$$\mathbf{.}$$
$$\mathbf{.}$$
$$\mathbf{.}$$
$$\mathbf{= 2^{n-1}.\ H_1 + 2^{n-2} + 2^{n-3} + ... + 2 + 1}$$
$$\mathbf{= 2^{n-1} + 2^{n-2} + ... + 2 + 1}$$
$$\mathbf{= 2^{n-1} - 1}$$

4.8.2.Fibonacci Sayıları:

Bir adada bir genç tavşan çifti var. Bunlar ilk 2 aya kadar üremiyorlar. 2 ay sonra her bir çift başka bir çift olarak ürüyorlar. Hiçbir tavşanın ölmediği var sayılarak n ay sonra adadaki tavşan çiftlerinin sayısı için bir rekurans (indirgeme) bağıntısı bulalım.

AYLAR	ÜRETİLEN ÇİFT	GENÇ ÇİFT	TOPLAM ÇİFT
1	0	1	1
2	0	1	1
3	1	1	2
4	1	2	3
5	2	3	5
6	3	5	8

$f_1 = 1$, $f_2 = 1$, $f_3 = 2$, $f_4 = 3$ ise $f_3 = f_1 + f_2$ ve $f_4 = f_3 + f_2$ olarak bulunur. Buna göre;

$\mathbf{n \geq 3}$ için $\mathbf{f(n) = f_{n-2} + f_{n-1}}$ olduğu kolayca görülebilir.

Fibonacci sayıları için bir indirgeme formülü bulalım.

$f_n = f_{n-1} + f_{n-2} \Rightarrow r^2 = r + 1 \Rightarrow r^2 - r - 1 = 0 \quad f_0 = 1 \; f_1 = 1$

$r_1 = (1+5^{1/2})/2 \qquad r_2 = (1-5^{1/2})/2 \qquad f_n = \alpha_1 .(1+5^{1/2}/2)^n + \alpha_2 (1-5^{1/2}/2)^n$ olursa

$f_0 = 1 \Rightarrow \alpha_1 + \alpha_2 = 1$

$f_1 = 1 \Rightarrow \alpha_1 (1+5^{1/2}/2) + \alpha_2 (1-5^{1/2}/2) = 1$ olur. Buradan,

$\alpha_1 = 1/5^{1/2}$ ve $\alpha_2 = -1/5^{1/2}$ bulunur. Böylece,Fibonacci sayıları için bir indirgeme formülü ;

$$f_n = (1/5)^{1/2} .((1+5^{1/2})/2)^n - (1/5)^{1/2} ((1-5^{1/2})/2)^n$$

olarak bulunur.

4.8.3.İndirgeme Bağıntılarının Karmaşıklığına Ait Çeşitli Örnekler

Örnek.4.15 : Aşağıdaki fonksiyonların karmaşıklığını tahmin ediniz?

a) $(n\log n + n^2)(n^3+2)$ b) $(n! + 2^n)(n^3 + \log(n^2+1))$

Çözüm.4.15:

a) $\log n < n \Rightarrow n \log n < n^2 \quad (n^2 + n^2)$

$n^3 + 2 \leq 2n^3$ ise $(2n^2)(2n^3)$ olduğundan $O(n^2) \cdot O(n^3) = 0(n^5)$

b) $n! < n^n$ $(n > 2)$ olduğundan

$\log(n^2 + 1) < \log 2n^2$ ve $\log(n^2 + 1) < \log 2 + 2.\log n$

$\log(n^2 + 1) < 3.\log n = O(\log n)$

$(n^2+2^n)(n^3 + \log(n^2 + 1))$

$n^n . n^3 = n^{3+n} = O(n^{3+n})$

Örnek.4.16 : Genel terimi $a_n = a_{n-1} + n$ ve başlangıç koşulu $a_0 = 1$ olan indirgeme bağıntısını ve karmaşıklığı bulunuz.

Çözüm.4.16:

$a_1 = a_0 + 1 \Rightarrow 1 + 1 = 2 = 1 + 1$

$a_2 = a_1 + 2 \Rightarrow 2 + 2 = 4 = 1 + 3$

$a_3 = a_2 + 3 \Rightarrow 4 + 3 = 7 = 1 + 6$

$a_4 = a_3 + 4 \Rightarrow 7 + 4 = 11 = 1 + 10$ ve buradan

$a_n = (1 + n . (n + 1)) / 2$ ise karmaşıklığı $n>2$ için $O(n^2)$ olur.

Örnek.4.17 : Genel terimi $a_n = 2n . a_{n-1}$ ve başlangıç koşulu $a_0 = 1$ olan indirgeme bağıntısı ve karmaşıklığı nedir.

Çözüm.4.17:

$a_1 = 2 . 1 = 2 . 1 . 1 = 2^1 . 1$

$a_2 = 4 . 2 = 2 . 2 . 2 = 2^2 . 2 . 1_3$

$a_3 = 6 . 8 = 2 . 3 . 2 . 2 . 2 = 2^3 . 3 . 2 .1$

$a_4 = 2 . 4 . a_3 = 2 . 4 . 2 . 3 . 2 . 2 . 2 = 2^4 . 4 . 3 . 2 .1$

.

$a_n = 2^n . n!$ ve karmaşıklığı n>2 için $O(n^{n+2})$ olur.

4.9. Sıralama Algoritmaları ve Karmaşıklığı (Ağaç Yapılar)

4.9.1.Giriş: Bir kümedeki elemanların sıralanması problemi çok çeşitli konularda ortaya çıkar.Örneğin,telefon idaresi bir rehber bastırmak istese abonelerinin isimlerini alfabetik olarak sıralaması gerekir.

Bir kümedeki elemanların hepsinin sıralı olduğunu varsayalım.Sıralama ,bu elemanların artan bir sırada bir liste içinde yeniden sıralaması olarak tanımlanır. Örneğin,7,2,1,4,5,9 elemanlarından oluşan bir listeden 1,2,4,5,7,9 şeklinde sıralanmış yeni bir liste üretebiliriz.Başka bir örnek olarak,d,h,k,m,n,a harflerinden oluşan listeden alfabetik sıra gözönüne alınarak a,d,h,k,m,n şeklinde yeni bir liste elde ederiz.

Bilgisayar kullanımının büyük bir oranı bir şeyi veya diğer bir şeyi sıralamaya bağlıdır.Bu nedenle,etkili sıralama algoritmaları geliştirmek için daha çok çaba sarfetmemiz gerekir.Bunun için,burada bazı sıralama algoritmaları ve onların karmaşıklık analizi incelenmiştir.Bu inceleme için ağaç yapıları (Trees) kullanılmıştır.

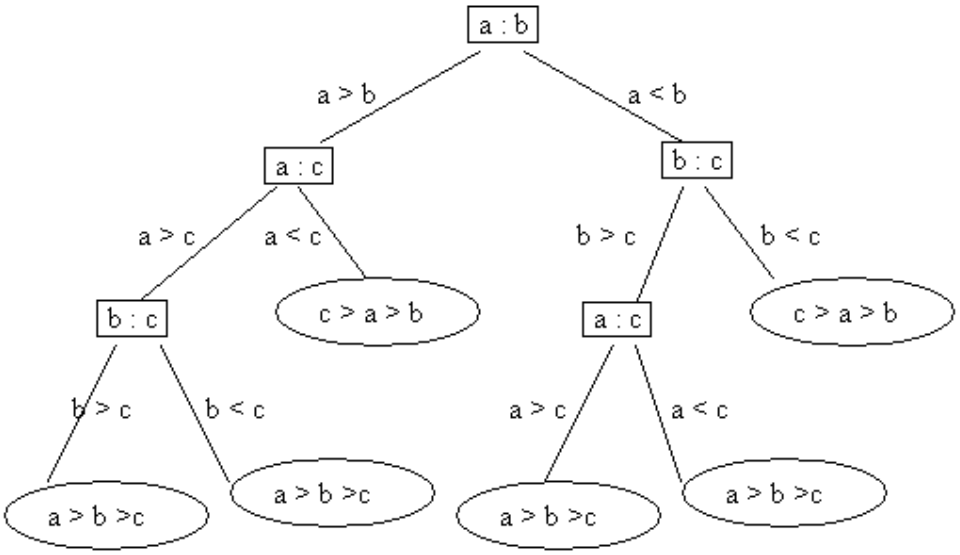
4.9.2.Sıralama Algoritmalarının Karmaşıklığı

Literatürde pekçok sıralama algoritması geliştirilmiştir.Belirli bir sıralama algoritmasının etkili olup olmadığına karar vermek için onun karmaşıklığının belirlenmesi gerekir. Ağaç Yapılı Algoritmalar**,** diğer sonlu algoritmalarda olduğu gibi iterasyon dizileri, pekçok paralel dalları içeren bir ağaç şeklindedir. Bir çok ağaç arama algoritmaları bu sınıfa aittir. Bu arama algoritmalarının bazıları, Dinamik Programlama, Dal ve Sınır, Sınırlı Sayım, Kapalı Sayım, Dal ve Çıkarma, Dal ve Atma vb. olarak sıralanabilir.

Modeller olarak ağaçları kullanarak sıralama algoritmalarının en kötü durum karmaşıklığı için bir alt sınır bulunabilir. n elemanlı bir kümenin permütasyon sayısı ,aynı zamanda n elemanın mümkün olan sıralama sayısı olup n!'dir.Sıralama algoritmalarının karmaşıklık hesabı için ikili karşılaştırmalara dayalı bir yol izlenecektir. Yani,bir zamanda iki eleman karşılaştırılacaktır. Buna göre,böyle karşılaştırmaların herbirinin sonucu olarak aşağıya doğru daralan mümkün sıralama kümeleri oluşur.

Böylece, sıralama algoritması ikili karşılaştırmalara dayalı ikili bir karar ağacı yardımıyla tarif edilebilir. İkili karar ağacı herbir iç tepe noktası iki elemanın karşılaştırılması esasını içerir.Her bir yaprak, n elemanlı bir kümenin n! permütasyonunun birini temsil eder.

Teorem.4.7:Aşağıdaki şekilde de ifade edilen üç elemanlı bir kümenin ikili karşılaştırmalarına dayalı bir sıralama algoritması için en az **(logn!)** karşılaştırmaya ihtiyaç vardır.(Şekil.4.2)



Şekil.4.2.Üç farklı elemanın sıralaması için Karar Ağacı

(logn!)= O(nlogn) olduğu hatırlanırsa karşılaştırmaları kullanarak O(nlogn)'den daha iyi en kötü durum zaman karmaşıklığına sahip bir sıralama algoritması bulunamaz.Sonuç olarak, O(nlogn) zaman karmaşıklığına sahip böyle etkili bir algoritma bulmak zordur.

5.BÖLÜM

5.DİJİTAL İMZALAR VE ŞİFRELEME ALGORİTMALARI

5.1.Giriş: Bu bölüm, bir el imzası için dijital kopya sağlayarak tasarlanan teknikleri inceler. Bir mesajın dijital imzası, sadece işareti ve mesaj kapasitesinin başlangıç işareti bilinen birkaç gizliliğe dayanan bir sayıdır. İmzalar soruşturabilir olmalıdır. İşaretlenmiş bir dökümanla ilgili bir tartışma çıkarsa, önyargısız üçüncü bir parti işaretçinin gizil bilgilerine girmeden meseleyi tekrar çözebilmelidir.

Dijital imzalar, veri bütünlüğü, izin ve güvenirliği içeren bilgi güvenliğindeki bzaı uygulamalara sahiptir. Dijital imzanın önemli uygulamalarından bir tanesi de, geniş bir network'te açık anahtar belgesidir. Belge, güvenilen bir üçüncü partide (TTP), açık anahtara gereken kullanıcı kimliğini bağlamak için bir vasıtadır. Böylece, diğer mevcut olanlar güvenilen üçüncü bir partinin yardımı olmadan bir açık anahtara güvenebilir.

Dijital imza kavramı ve yararı çok daha önceden tanınmasına rağmen son birkaç yıldır popüler olmuştur. Bu imza projelerinin en pratik olanı ve en geniş uygulama alanına sahip olanı RSA imza projesidir. Sonraki çalışmalarda bazı alternatif dijital imza tekniklerinin ortaya çıkması ile sonuçlanmıştır ki bunlar tanımlamalara ve görevlere göre önemli avantajlar önerir.

5.2. Dijital İmza Mekanizmasının Çatısı

Bu bölüm dijital imza projeleri için genel bir model tanımlar. RSA ve ElGamal zamanın aşırı harcanmamasını tavsiye eder. Fonksiyon fikri, geri alınan mesajla dijital imzaların verildiği algoritmaları anlamak için gereklidir.

5.2.1.Temel Tanımlar:

1- Dijital imza, birkaç başlangıç değeri ile dijital formda bir mesajı birleştirilen olan veri dizisidir.

2- Dijital imza üreteç algoritması dijital imzayı oluşturmak için bir işlemdir.

3- Dijital imza Doğrulama(Gerçekleme) algoritması, dijital imzanın doğruluğunu kanıtlamak için bir işlemdir.

4- Dijital imza projesi, birleştirilmiş Doğrulama(Gerçekleme) algoritması ve imza üreteç algoritmasının oluşumudur.

5- Dijital imza işaretli prosedürü, işaretlenebilen mesajdaki veriyi formatlamak için bir metot ile dijital imza üreteç algoritmasının oluşumudur.

6- Dijital imza Doğrulama(Gerçekleme) prosedürü; mesajdan veriyi geri almak için bir metot ile bir Doğrulama(Gerçekleme) algoritmasından oluşur.

Bu bölüm, dijital imza projeleri ile ilgilidir. Pratikte dijital imza projesini kullanmak dijital imza işlemlerini elde etmek için gereklidir. Çeşitli projeler için bahsedilen birkaç işlem, ticariye uygun standartlarla ortaya çıkmıştır. Bunlardan iki işlemin adı 5.3.5 ve 5.3.6 'da tarif edilen ISO/IEC 9796 ve PKCS #1 'dir. Tablo 5.1.'de kullanılan semboller gösterilmiştir. Tablo 5.1'de listelenen kümelerle fonksiyonlar herkesçe bilinir.

Tablo. 5.1 : Dijital İmza Mekanizmaları İçin Kullanılan Notasyonlar

Notasyon	**Anlamı**
$\mathcal{M}$	Mesaj uzayı olarak adlandırılan elemanların kümesi .
$\mathcal{M}s$	İşaretleme uzayı olarak adlandırılan elemanların kümesi .
S	İmza uzayı olarak adlandırılan elemanların kümesi .
$\mathcal{R}$	Redundancy fonksiyon olarak adlandırılan $\mathcal{M}$ 'den $\mathcal{M}s$'ye fonksiyon.
$\mathcal{M}_R$	$\mathcal{R}$'nin sanalı($\mathcal{M}_R$ = **Im** ($\mathcal{R}$)) **.**
R^{-1}	$\mathcal{R}$'nin tersi ($R^{-1} : \mathcal{M}_R \rightarrow \mathcal{M}$)
R	İşaretleme için index kümesi olarak adlandırılan elemanların kümesi
H	$\mathcal{M}$ bölgesinde tek-yön fonksiyonu
$\mathcal{M}_h$	h 'nin sanalı($h : \mathcal{M} \rightarrow \mathcal{M}_h$) ; $\mathcal{M}_h \subseteq \mathcal{M}s$ hash değer uzayı

5.1 Not:

i) (Mesajlar) Dijital imzayı bağlayabilen bir işaret için elemanların kümesi M'dir.

ii) (İşaretleme uzayı) İmza çevirilerinin uygulandığı elemanların kümesi M_s 'dir. İmza çevirileri direk olarak M kümesine uygulanamaz.

iii) (İmza uzayı) M'deki mesajlar ile birleştirilen elemanların kümesi S'dir. Bu elemanlar işaretçiyi mesaja bağlamakta kullanılır.

iv) (İçerik kümesi) R, özel belirli çevirileri tanımlamakta kullanılır.

5.2.2.Dijital İmza Projesinin Sınıflandırılması:

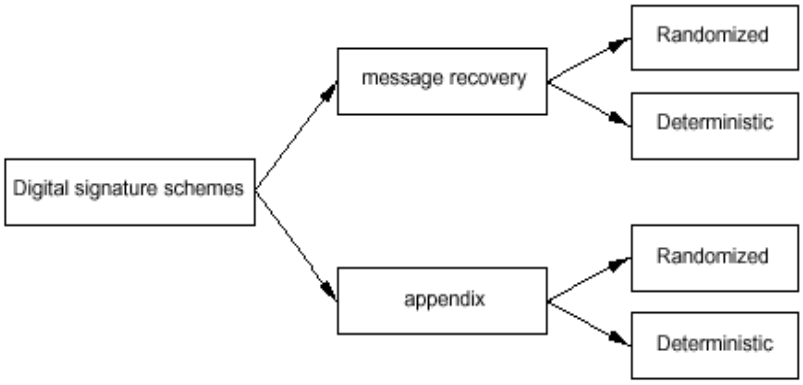
5.2.2 ve 5.2.3 'te aşağıda kısaca özetlenen dijital imza projesinin iki genel sınıfı tarif edilmektedir.

1- İlaveli dijital imza projesi Doğrulama(Gerçekleme) algoritmasına girdi (entıty) olarak orijinal mesajı gerektirir.

2- Geri alınan mesajlı dijital imza projesi, Doğrulama(Gerçekleme) algoritmasına girdi (girdi(entıty)) yapıldığında orijinal mesajı gerektirmez. Bu sebeble orijinal mesaj imzadan kendi kendine geri alınır. Bu sınıflar $|R|=1$ olup olmaması durumu ile ilgili olarak alt bölümlere ayrılabilir.

5.2.Tanım: Dijital imza projesi , $|R|>1$ ise rasgele dijital imza projesi olduğunu ifade eder. Aksi halde, dijital imza projesi deterministiktir.

Şekil 5.1 'de bu sınıflandırma gösterilmiştir. Deterministik dijital imza mekanizması tek-zaman imza projesi ve çoklu-kullanılan proje olarak ikiye ayrılır.



Şekil 5 .1. : Dijital İmza Projelerinin Sınıflandırılması

5.2.3. İlaveli Dijital İmza Projeleri

Pratikte genel olarak kullanılan sınıftır. Bunlar rastgele fonksiyonlardan çok şifrelenebilen hash fonksiyonlarına dayanır.

5.3. Tanım: Doğrulama(Gerçekleme) algoritmalarına girdi (entıty) yapıldığı gibi mesaj gerektiren dijital imza projeleri "İlaveli dijital imza projeleri" olarak adlandırılır.

İlaveli dijital imzaların sağladığı mekanizma örnekleri DSA, El Gamal ve Schnorr imza projeleridir.

5.4. Algoritma: İlaveli Dijital İmza Projeleri için Anahtar Üretimi

Özet:Her bir girdi(entıty) işaretli mesajlar için bir gizli anahtar oluşturur ve uygun bir açık anahtar imzaların doğruluğu için diğer girdiler(entıty) tarafından kullanılır.

1-Her A değeri $S_A = \{S_{A,k}: k \in R\}$ şeklinde dönüşümlerin kümesiyle tanımlanan özel bir anahtar seçmelidir.Herbir $S_{A,k}$, M_h'den S'ye 1-1 tasvir olup işaretçi dönüşüm olarak adlandırılır.

2-S_A, M_h x S'den {true, false}'a uygun çizilen V_A 'yi tanımlar. Şöyleki

$$V_A(\tilde{m},s^*)=\begin{cases}\text{True}, & S_{A,k}(\tilde{m})=s^* \text{ ise} \\ & \text{diğer yönden} \\ \text{False}, & \end{cases} \Longrightarrow$$

Her $\tilde{m} \in M_h$ $s^* \in S$ için geçerlidir.
Burada $m \in M$ için $\tilde{m}=h(m)$ 'dir.

V_A; Doğrulama(Gerçekleme) dönüşümleri olarak adlandırılır ve kurulur. Şöyleki işaretçinin özel anahtar bilgisi olmadan hesaplanabilir.

3-A'nın açık anahtarı V_A 'dır. A'nın özel anahtarı S_A kümesidir.

5.5. Algoritma :İmza Üretimi ve Doğrulama(Gerçekleme)(İlaveli Dijital İmza Projeleri)

Özet: A girdi(entıty)i $m \in M$ olan bir mesaj için $s \in S$ imzasını oluşturur. Daha sonra herhangi bir B girdisi(entıty) ile doğrulanır.

1- İmza üretimi: Girdi (entıty) A, aşağıdakileri yapmalıdır.

a) $k \in R$ olan bir eleman seçer.

b) $\tilde{m}=h(m)$ ve $s^*=S_{A,k}(\tilde{m})$ hesaplanır

c) M'nin s* için A'nın imzası m ve s^*'nin herikisi, imzayı Doğrulama(Gerçekleme)yı isteyebilen girdi(entıty)ler için geçerli yapılabilir.

2- Doğrulama(Gerçekleme): Girdi(entıty) B aşağıdakileri yapmalıdır.

a) A'nın gerçek açık anahtarı V_A'yı elde etmek.

b) $\tilde{m}=h(m)$ ve $u=V_A(\tilde{m},s^*)$ hesaplamak.

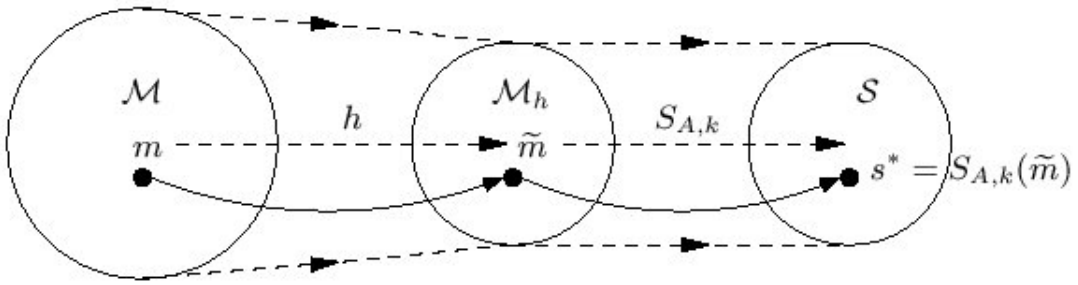
c) Sadece u=true ise imzayı kabul eder.

Şekil 5.2; ilaveli bir dijital imza projesinin şemasını verir. Takibeden özellikler, Doğrulama(Gerçekleme) dönüşümlerini ve işaretleri gerektirir.

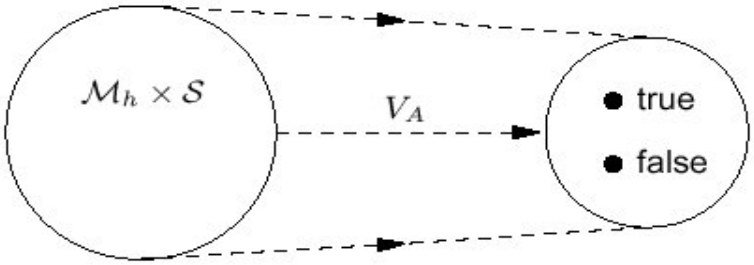
i) Her $k \in R$ için $S_{A,k}$ hesaplamak için etkili olmalıdır.

ii) V_A hesaplamak için etkili olmalıdır ve

iii) Bir $s^* \in S$ şöyle ki $\tilde{m}=h(m)$ iken $V_A(\tilde{m},s^*)$=true ve bir $m \in M$'yi bulmak için A'dan başka bir girdi(entıty)in mümkün olmadığı hesaplama olması gerekir.



a)



Şekil.5.2. ilaveli bir dijital imza projesinin şeması

5.6.Not:(hash fonksiyonlarının kullanımı**)** geri alınan mesajlarla dijital imza projeleri; ilaveli dijital imzaların keyfi uzunlukla mesajlara uygulandığında, sabit uzunluklu mesajlara uygulanır. Algoritma 5.5'teki tek-yol fonksiyonu h, serbest çarpma hash fonksiyonu olmak için seçilir. Hash için bir alternatif, geri alınan mesajlarla bir imza projesi kullanarak bireysel olarak işaretlenebilen sabit uzunlukta bloklar içindeki mesajları kırmaktır. İmza üretimi, pekçok proje için nispeten yavaş olduğunda ve çoklu işaret bloklarının geri sıralanışı bir güvenlik riskini gösterdiğinde tercih edilen metot **hash**'tir.

5.2.3. Mesajı geri almalı Dijital İmza Projeleri

Bu bölümde açıklanan dijital imza projeleri, kendi kendine imzadan, işaretli mesaja çevrilebilen gibi bir özelliğe sahiptir. Pratikte, bu özellik kısa mesajlar için kullanılır.

5.7. Tanım: Mesajı geri almalı dijital imza projeleri, doğrulama(gerçekleme) algoritmasında gerekli olmayan mesajın önceki bilgisi için bir dijital imza projesidir.

Mesajı geri almalı dijital imzaların sağladığı mekanizmaların örnekleri RA, Rabin ve Nyberg-Rueppel açık-anahtar imza projesidir.

5.8. Algoritma: Mesajı geri almalı dijital imza projeleri için anahtar üretimi

Özet: Her girdi (entıty), işaretli mesajları kullanmak için bir özel anahtar oluşturur ve doğrulanan imzalar için diğer girdiler (entıty) tarafından uygun bir açık anahtar kullanılır.

1- Her A girdisi (entıty) dönüşümlerin $S_A=\{S_{A,k}:k\in R\}$ kümesi seçmelidir. Her $S_{A,k}$ M_s' den S' ye 1-1 çizimdir ve işaretlenmiş dönüşüm denir.

2- S_A, her $k\in R$ için M_S üzerinde birim tasvir olan V_{Ao} $S_{A,k}$ özelliğinde uygun bir V_A tasvirini tanımlar. V_A, doğrulama (gerçekleme) dönüşümü olarak adlandırılır ve işaretçinin özel anahtar bilgisi olmadan hesaplanabilecek şekilde oluşturulabilir.

3- A'nın açık anahtarı V_A, A'nın özel anahtarı S_A kümesidir.

5.9.Algoritma:Mesajı geri almalı projeler için doğrulama(gerçekleme) ve imza üretimi.

Özet: Girdi (entıty) A, $m\in M$ mesajı için $s\in S$ imzası üretir. Daha sonra herhangi bir B girdisi(entıty) tarafından doğrulanır. M mesajı s'den geri alınır.

1- İmza üretimi: A girdi (girdi(entıty))i aşağıdaki gibi takip etmelidir.

a) $k\in R$ olan bir eleman seçer.

b) $\tilde{m} = R(m)$ ve $s^* = S_{A,k}(\tilde{m})$ hesaplanır.

c) A'nın imzası s^* 'dir. Bu, imzayı doğrulamak(gerçeklemek) isteyebilen girdiler (entıty) için geçerli kılınır ve s^* 'dan m'ye çevrilir.

2- Doğrulama(Gerçekleme): B girdisi(entıty) aşağıdaki gibi takip edilmelidir.

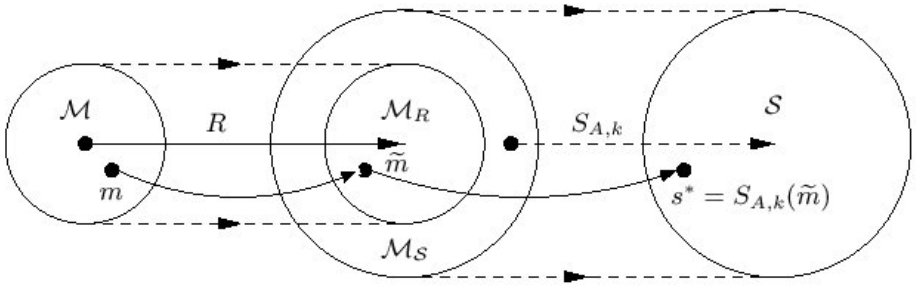
a) A'nın doğruluk açık anahtarı V_A 'yı hesaplanır.

b) $\tilde{m} = V_A(s^*)$ hesaplanır.

c) ($\tilde{m} \notin M_R$ ise imza reddedilir.) $\tilde{m} \notin M_R$'yi oluşturur.

d) $R^{-1}(\tilde{m})$ hesabıyla $\tilde{m}$ 'den m 'ye çevrilir.

Şekil 5.3 , mesajı geri almalı dijital imza projesinin görünümünü şematik olarak verir. Takip edilen özellikler, doğrulama(gerçekleme) dönüşümlerini ve işaretlenenleri gerektirir.



şekil.5.3. mesajı geri almalı dijital imza projesinin şematik görünümünü

i) Her $k \in R$ için $S_{A,k}$ hesaplama için etkili olmalıdır.

ii) V_A, hesaplama için etkili olmalıdır.

iii) $V_A(s^*) \in M_R$ olan $s^* \in S$ 'yi bulmak için A'dan başka bir girdi (girdi(entıty))in mümkün olmayan hesaplama olması gerekir.

5.10. Not:(redundancy fonksiyon) redundancy fonksiyon R ve onun tersi R^{-1}, açıkça bilinir. Uygun R'nin seçimi, sistem güvenliği için önemlidir.Bunu ifade etmek için, $M_R=M_S$ olduğunu varsayalım. Varsayılan R ve $S_{A,k}$ sırasıyla M'den M_R'ye ve M_S'den S'ye 1-1,örten 'dır.Bu, M ve S'nin aynı eleman sayısına sahip olduğuna işaret eder. Daha sonra $s^* \in S$, $V_A(s^*) \in M_R$ için aşağıdaki dönüşüm Doğrulama(Gerçekleme) algoritmalarıyla kabul edilecek m mesajı ve karşılığı olan s* imzasını bulmak önemsizdir.

1- Rastgele $k \in R$ ve rastgele $s^* \in S$ seçer.

2- $\tilde{m} = V_A(s^*)$ hesaplanır.

3- $m=R^{-1}(\tilde{m})$ hesaplanır.

Eleman s*, m mesajı için geçerli imzadır ve işaretlenmiş dönüşüm S_A kümesinin bilgisi olmadan oluşturulur.

5.11. Örnek:(redundancy fonksiyon) Birkaç sabit pozitif n tamsayısı ve $M_S=\{t:t \in \{0,1\}^{2n}\}$için $M=\{m:m \in \{0,1\}^n\}$olduğunu varsayalım. $M_R=\{m \| m:m \in M\} \underline{C} M_5$ olan $R(m)= m \| m$ tarafından $R:M \to M_s$ tanımlanır. n'nin büyük değerleri için $|M_R|/|M_S|=\left(\frac{1}{2}\right)^n$ ihmal edilebilir. R redundancy fonksiyonu $_A(s^*) \in M_R$ 'deki s^* seçimi için uygundur.

5.12:(Redundancy fonksiyon seçimi) R redundancy fonksiyonu açık bilgidir ve R^{-1}'i hesaplamak kolaydır. R'nin seçimi önemlidir ve S_A içinde işaretlenmiş dönüşüm seçimi bağımsız olarak yapılmamalıdır. 5.21 örneği; imza projelerinin güvenliğini tehlikeye atan redundancy fonksiyonun özel bir örneğini verir. Uluslararası standart olarak kabul edilen bir redundancy fonksiyon örneği 5.3.5'te verilmiştir. Bu redundancy fonksiyon tüm mesajı geri alan dijital imza projeleri için uygun değildir. Fakat RSA'ya ve Rabin dijital imza projelerine uygulanır.

5.13.(Mesajı geri alma projelerinin özel bir sınıfı)

5.13.1.

i) Tersi alınabilen açık anahtar şifrelemesinden dijital imzalar

Bu bölümde belirli açık anahtar şifreleme sistemlerinde bir takım dijital imza projeleri dikkate alınmıştır.

Farzedelim ki E_e nin mesajları açık anahtar şifreleme dönüşümü ile M uzayına ve şifre dosyasınıda C uzayına çeviriyor. Daha da ötesi M=C'dir. Eğer D_d kod çözümlemesi ile dönüşerek E_e' ye denk ise E^{e}'' ye kadar D_d'nin tüm permütasyonları tüm $m \in M$ için,

$D_d(E_e(m))= E_e(D_d(m))=M_d$' dir.

Bu tipteki açık anahtar şifreleme projelerine tersine çevrilebilir denir. Bu eşitliğin geçerli olabilmesi için M=C ve tüm $m \in M$ olması gereklidir. Aksi takdirde $D_d(m)$ işlemi $m \notin C$ durumunda anlamsızdır.

ii)Bir dijital imza projesinin oluşturulması

1. M bir mesaj aralığı olsun, dijital imza projesi için

2. C=M, S aralığında bir imza olsun

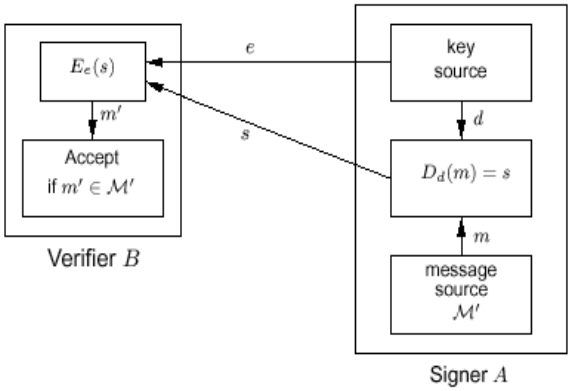
3. (e,d) ninde açık anahtar şifreleme projesinde bir anahtar parçası olduğunu farzedelim.

4. S_a' dan D_d ye bir imza fonksiyonu tanımlayalım $m \in M$ ve $s=D_d(m)$ iken bu bir imzadır.

5. Doğrulama(Gerçekleme)fonksiyonunu

$$V_A(m,s)=\begin{cases} \text{doğru } E_e(s)=m \\ \text{yanlış diğer durumlarda} \end{cases}$$

İmza projesi basitleştirilebilir ve ayrıca eğer A mesajları sadece özel bir yapıda imzalıyorsa ve bu yapı açıkça biliyorsa M' M'in bir altkümesi olsun eğer M' bu özel yapı tarafından iyi bir şakilde tanımlanmıştır. Öyle ki M' ihmal edilebilir. Mesaj franksiyonları içerir. Örneğin; varsayalım ki t' nin bazı pozitif değerleri için M'nin oluşturduğu ikililerinin 2t' ye yaklaştığını düşünün.Bu yeni senaryoların altında A sadece s imzasını $M=E_e(s)$ olana dek Doğrulama(Gerçekleme)fonksiyonlarına uyup karsayarak taşımaya ihtiyaç duyar. Bu tip projelere mesaj korumalı dijital imza projeleri denir. Şekil 5.4'de bu tip imza fonksiyonlarının nasıl kullanıldığı anlatılıyor. Bu özel yapının seçici mesajlar özelliği gereksiz mesajları seçip ayıklar.



Şekil.5.4.Mesajı geri alan dijiral imza yapısı

Yukarıda gösterilen bu değiştirme basit bir sadeleştirmeden fazlasını içerir.Eğer,kümelerden biri b'nin özelliğindeki gibi bir imzalama ve doğrulama(Gerçekleme)fonksiyonlarını taşımak isterse bu kesinlikle çok can alıcı olur. Bunun neden olduğunu görmek için B girişi rastgele bir eleman ($s \in S$ olmak üzere) seçerek S=M olana kadar E_e ile $u=E_e(s)$ kabullenebilir ve E_e bir açık bilgidir. B, daha sonra m=u mesajını s olası için m üzerindeki imzayı alabilir ve (m,s)'yi geçirir. m için A tarafından oluşturulan bir imzayı s'nin soruşturacağını kontrol etmek kolaydır. Fakat A hiçbir gruba sahip değildir.

Sonuç olarak B, A'nın işlenmiş bir imzasıdır. Buna sahte existendial denir.

Eğer M' sadece M'den küçük mesaj parçaları içeriyorsa A'nın imzasını işleyebilen birkaç giriş bu konuda küçüktür.

Not (Dijital imzalar güvenilirliği**):** Her ne kadar dijital imza projeleri ters dönüştürülebilir açık anahtar şifrelemesine göre çekici kılınsada aslında basit, ilkel şifreleme metodlarına ihityaç duymaktadır. Bazı durumlar var ki, bazı dijital imza mekanizmasının kullanımına ihtiyaç vardır fakat şifreleme yasaklanmıştır. Bu tip sebeplerle bu dijital imza projeleri uygunsuz kalmıştır.

iii) Pratikte Dijital İmzalar

Dijital imzalamayı pratikte kullanılabilir hale getirmek için bir takım işlemlerinin özelliklerine katılması gerekir. Bir dijital imza mutlaka;

1- İmzalayan tarafından kolayca uygulanır olmalı.(imzalama fonksiyonunun kolayca uygulaması eklenmeli)

2- Başkaları tarafından rahatlıkla tanımlamalıdır.(tanımlama fonksiyonu rahatlıkla tanınmalıdır)

3- Çok keskin zarif çizgileri olmalıdır. Orjinalliği korunmalı, sabit kalmalıdır.

iv)Tartışma Çözümleri

Dijital imzanın amacı tartışma çözümlerine izin vermektir. Örneğin; giriş A, birkaç noktada işaretlenmiş bir mesajı veya A tarafından üretilen mesaj üzerindeki imzayı diğer B girişinin yanlışlıkta iddia edebildiğini reddedebilir. Güvenilen 3. grup veya hakim problemlerinin üstesinden gelmek için gereklidir. TTP, ilerlemede aynı fikirleri tüm grupların içerdiği birkaç giriş olmalıdır.

Eğer A, A tarfından üretilmiş, B tarafından tutulan mesajı reddederse, B yalnız m ile TTP için m'in S_A imzasını gösterebilmelidir. TTP kuralları, $V_A(m,S_A)$=true ise B'nin lehine değilse A'nın lehinedir. A'nın yaptığı gibi aynı soruşturulmuş çeviri V_A' ya sahip TTP gibi B güvenli ise B kararı kabul edilecektir. Uzlaşmanın olmadığı ve TTP'nin V_A' yı kullandığı gibi A güvenli iseA kararı kabul edilecektir. Bundan başka yanlış tartışma çözümü aşağıdaki kriterleri gerektirir.

v)Tartışma imzalarının çözümleri için gerekenler

1- S_A ve V_A sayfa 23'ün a ve b özelliğine sahiptir.

2- TTP, V_A' nın kopyasının bir doğruluğudur.

3- İşaretlenmiş çeviri S_A gizli tutulmalı ve güvenliği sağlamalıdır.

Bu özellikler gereklidir fakat pratikte onları garanti altına almak mümkün olamyabilir. Örneğin;???

, tersi alınabilen açık anahtar şifreleme metotlarından doğan mesajı geri almalı dijital imza projelerinin bir sınıfını açıklar. Aşağıda, RSA ve RABIN şifreleme projelerinin içeriği detaylı olarak açıklanmıştır. Karşılığı olan imza mekanizmaları , bölüm **5.3.1** ve **5.3.4**'te anlatılır.

5.13.2) RSA Açık Anahtar Şifrelemesi

Bulucuların soyisimleriyle adlandırılan RSA şifreleme sistemi en çok kullanılan açık anahtar şifreleme sistemidir. Dijital imzalarda ve gizliliği sağlamakta kullanılır ve güvenliği tamsayı çarpanlara ayırma problemlerinin zorluğuna dayandırılmıştır. Bu bölüm RSA şifreleme sistemi, onun güvenliği ve bazı sonuçları tanımlamaktadır. RSA imza taslakları 5.3.1'in içinde yer almaktadır.

5.13.2.1 Tanım

5.13.2Algoritma RSA açık anahtar şifrelemesi için anahtar genellemesi

Özet: Herkes RSA açık anahtarı ve buna uyan gizli anahtarı yapabilir. Şu yol izlenmelidir.

1) Rasgele iki tane büyük, her biri kabaca aynı büyüklükte olan p ve q asal sayıları üretilir.

2) $(p-1)(q-1) = \phi$ olacak şekilde $n = p.q$ hesaplanır.

3) $qcd(e, \phi) = 1$ ve $1< e < \phi$ olacak şekilde rastgele e sayısı seçilir.

4) Uzatılmış öklit algoritması kullanılarak $ed = 1(mod\phi)$ ve $1< d < \phi$olacak şekilde d sayısı hesaplanır.

5) A'nın açık anahtarı (n,e), gizli anahtarı d'dir.

5.13.2.2 Tanım: RSA anahtar üretiminde e ve d şifreleme deşifreleme osleri, n ise modül olarak adlandırılır.

5.13.2.3 Algoritma RSA Açık Anahtar Şifreleme

Özet: B A için m mesajını şifreler, A çözer.

1. Şifreleme: B şunları izlemeli;
   a) A'nın esas açık anahtarı olan (n,e) elde etmeli.
   b) Mesajı [0, n-k] aralağındaki m tamsayısı olarak göstermeli.
   c) $C = m^e \pmod n$ hesaplanmalı (algoritma 2.143 kullanılarak).
   d) C şifrelenmiş metni A'ya göndermeli
2. Şifre Çözme: c'den açık metin m'i elde etmek için a şunları izlemelidir

a) D gizli anahtarını kullanarak m elde edilir. $m=c^d \pmod n$

Deşifreleme Çalışmalarının Kanıtı: $ed = 1+ k\phi$ olacak şekilde k tamsayısı vardır. Eğer $gcd(m,p) = 1$ise Fermat teoreminden

$$m^{p-1} \equiv 1(\text{mod } p)$$

Eşitliğin her iki tarafı k(q-1) kuvveti ile büyütülür ve m ile çarpılır.

$$m^{1+k(p-1)(q-1)} \equiv m \text{ (mod } p)$$

Diğer taraftan eğer gcd (m,p) = p ise denkliğin her iki tarafı mod p'ye göre 0 olduğu sürece en son denlik geçerlidir.

Böylece bütün durumlarda:

$$m^{ed} \equiv m(\text{mod } p) \text{ olur.}$$

Aynı delilden

$$m^{ed} \equiv m(\text{mod } p) \text{ olur.}$$

Sonuç olarak p ve q farklı asallar olursa şu şekilde olur

$$m^{ed} \equiv m(\text{mod } n)$$

ve böylece

$$c^d \equiv (m^e)^d \equiv m \ (\text{mod } n)$$

5.13.2.4.Örnek: (yapay küçük parametrelerle RSA şifreleme)

i)Anahtar Üretme:

A kişisi p = 2357 ve q = 2551 olacak şekilde p ve q asallarını seçer ve

$\Phi=(p-1)(q-1) = 6007800$ olacak şekilde

d = 422191 sayısını bulur. A'nın gizli anahtarı d = 422191 iken açık anahtarı (n=6012707, e=3674911) çiftidir.

m = 5234673 mesajını şifrelemek için B modüler üs algoritmasını kullanır.

$c = m^e \text{ mod } n = 5234673^{3674911} \text{ mod } 6012707 = 3650502$ ve bunu A'ya gönderir.

Şifre Çözme: C'yi çözmek için A

$C^d \text{ mod } n = 3650502^{422191} \text{ mod } 6012707 = 5234673$ hesaplanır.

Not: (Evrensel üs): λ = Icm(p-1,q-1) sayısı bazen n sayısının evrensel üssü olarak adlandırılır ve RSA anahtar üretiminde $\phi = (p-1)(q-1)$ yerine kullanılabilir. λ'nin ϕ'nın böleni olmasına dikkat edilmelidir. λ kullanılarak daha küçük d deşifreleme üssü elde edilir ve deşifrelemenin daha hızlı olmasını sağlar.

Bununla birlikte eğer p ve q rasgele seçilirse gcd(p-1,q-1)'in küçük olası beklenir. Ve sonuç olarak ϕ ve λ kabaca aynı büyüklüktedir.

5.13.2.5. RSA'nın Güvenliği

Bu bölümde RSA şifreleme ile ilgili çeşitli güvenlik sonuçları ele alınmıştır. Literatürde yer alan çeşitli hırsızlıklar ve bu davranışlara karşı alınan önlemler üzerinde çalışılmıştır.

(i) Çarpanlara Ayırma

Pasif bir hırsız ile karşılaşıldığında yapılacak iş A alıcısının genel bilgilerini (n,e) neren c şifreli metninden m açık metnini elde etmektir. Bu RSA problemi (RSAP) olarak adlandırılır. Bu problemle ilgili bilinen bir algoritma yoktur.

Hırsızın RSA problemini çözmek ilk olarak n faktörünü kullanması ve sonra ϕ ve d'yi A'nın algoritma 8.1'de yaptığı gibi hesaplaması uygun bir yaklaşımdır. d ele geçirildiğinde Aiçin tasarlanan şifre metni hırsız tarafından deşifre edilebilir.

Diğer bir tarfta, eğer hırsız d'yi hesaplayabilirse, şu şekilde n'ide hesaplayabilir. $ed \equiv 1 \ (\text{mod } \phi)$ olduğunda, $ed-1 = k\phi$ olacak şekilde k tamsayısı vardır. Böylece 2.126 daki gibi bütün $a \in Zn^*$ sayıları için $a^{ed-1} \equiv 1 \ (\text{mod } n)$ olur. t tek tamsayı olacak şekilde $ed-1 = 2^s t$

elde edilir. $a^{2^{i-1}t} \not\equiv \pm 1 \pmod{n}$ ve $a^{2^i t} \equiv 1 \pmod{n}$ a'nın en az yarısı $a \in Z_n^*$ olacak şekilde bir iE[1,5] sayısının varlığı gösterilmiş olur. a ve i böyle tamsayılar ise gcd($a^{2^{i-1}t}$ -1.n) n'in önemli bir faktörüdür. Böylece düşman $a \in Z_n^*$ olacak şekilde rastgele sayılar seçer ve $i \in [1,5]$ satısının yukarıda özelliklere sahip olup olmadığını kontrol eder. Bu durum aşğıdaki gibidir.

Açık anahtar (n,e)'den RSA deşifreleme üssü d'yi hesaplama problemi ve n'i çarpanlara ayırma roblemlerinin hesaplanması eşdeğeridir. RSA anahtarı üretildiğinde, n = pq yoluyla seçilen p ve q asallarının hesaplanabilmesi mümkün değildir.

(ii) Küçük Şifreleme Üssü

Bir şifrelemenin verimliliğini kanıtlayabilmek için e=3 biçimde küçük bir üs seçilebilir. Gruptaki herkes aynı e şifreleme üssüne sahip olabilir, fakat herkesin kendine ait farklı bir modülü olmalıdır. Eğer A kişisi m mesajını kişisel modülleri n_1, n_2, n_3 olan 3 kişiye göndermek isterse ve şifreleme üsleri e=3 ise i=1,2,3 değerleri ise A mesajı $c_i=m^3 \bmod n_i$ olacak şekilde gönderir. Eğer modüller aralarında asal ise c_1, c_2, c_3 Gaussis algoritmasını kullanarak 3 denkliğe $0 \le x < n_1 n_2 n_3$ olacak şekilde x sonucu bulunur.

$$x \equiv c_1 \pmod{n_1}$$
$$x \equiv c_2 \pmod{n_2}$$
$$x \equiv c_3 \pmod{n_3}$$

Kalan teoreminden $m^3 < n_1 n_2 n_3$ olursa $x = m^3$ durumu olur. Böylece casus x'in küp köklerini hesaplayarak m açık metnini elde edebilir.

Bu nedenle e=3 olacak şekilde küçük bir şifreleme üssü aynı mesaj için yada bilinen değişikliklerle bir çok kişiye gönderilmemelidir. Alternatif olarak, bu tür hırsızlıkları önlemek için, uygu nuzunlukta yarı rastgele üretilmiş sayılar şifreleme yapmadan önce açık metnin mesajına eklemelidir. Bu işlemden mesajı tuzlamak olarak bahsedilmektedir.Küçük şifreleme üsleri m küçük mesajları için problem oluşturur. Eğer $m < n^{1/e}$ olursa c'nin kökü e^{+4} basit bir şekilde hesaplanarak $c=m^e \bmod n$ şifreli metninden m elde edilebilir. Tuzlama bu tür problemlerden bir kaçış yoludur.

(iii) İleri Arama Hırsızlıkları

Eğer mesaj alanı küçük yada tahmin edilebilir ise, hırsız c'yi elde dinceye kadar bütün olasılıklarla mesajı şifreler ve c şifreli metnini elde eder. Yukarıda tanımlandığı şekilde mesajı tuzlama işlemi bu tür hırsızlıkları önlemek için bir metoddur.

(iv) Küçük Deşifre Üssü d

Verimli bir deşifreleme geliştirebilmek için, e şifreleme üssünde kullanılan durumlar d içinde olabilir. Bununla birlikte eğer gcd(p-1,q-1) küçük ise kötü bir durumdur, eğer d (n,e) genel bilgisiden d'yi hesaplamak verimli bir algoritma olur. bu algoritma d'nin n ile aynı büyüklükte olması durumunda genişletilemez. Böylece, hırsızlıktan korunmak için, deşifreleme üssü d'nin kabaca n ile aynı olması gerekir.

(v)Çarpma Özellikleri

m_1 ve m_2 2 tane açık mesaj, c_1 ve c_2 ise onların herbiri için RSA şifresi olsun.

$(m_1m_2)^e \equiv m_1^e\, m_2^e \equiv c_1\, c_2 \pmod n$olduğunu gözlemle

Diğer bir deyişle $m=m_1m_2 \pmod n$ açık metni $c=c_1c_2 \pmod n$ şifreli metne uyar. Bu RSA'nın homommorphic özelliği olarak gösterilmiştir. Bu gözlem bizi RSA üzerinde uygun seçilmiş şifreli metin hırsızlıklarına götürür.

Farzedelim bir hırsız $c=m^e \bmod n$ şekilde A için tasarlanmış metni deşifre etmek istiyor. Farzedelim A keyfi olarak metni deşifre edecek. Hırsız $x \in Z^*n$ olacak şekilde rastgele bir tamsayı seçerek c'yi gizleyebilir ve $\hat{c}=cx^e \bmod n$ sayısını hesaplayabilir. $\hat{c}$' yi göstermenin ötesinde A hırsız için $m=(e)^d \bmod n$ sayısı hesaplayabilir.

$$m \equiv (\hat{c})^d \equiv c^d\,(x^e)^d \equiv mx \pmod n$$

böylece hırsız $m=mx^{-1} \bmod n$ sayısını hesaplayabilir. Bu adaptasyon açık metin mesajları üzerinde yapılan zorlamalardan kaçınmak için yapılmalıdır. Eğer deşifre edilen c şifreli metni bu yapıya sahip değilse c bir dolandırıcı tarafından reddedilecektir. Eğer m mesajı dikkatli bir şekilde şeçilmiş bu yapıya sahipse, büyük bir ihtimalle $x \in Zn^*$ için mx mod n sayısı olmayacaktır. Bu nedenle bir önceki paragrafta tanımlanan uygun seçilmiş şifreli metin hırsızlıkları başarısızlıkla sonuçlanacaktır, çünkü A hırsız için c'yi deşifre edemeyecektir. Note 8.63 uygun seçilmiş şifreli metin hırsızlıkları ve hırsızlık çeşitlerine karşı koruma için güçlü teknikler sağlanmaktadır.

(Vİ) Ortak Modül Hırsızlıkları

İlerleyen konuda niçin herbir kişinin kendi özel RSA n modülünü seşmesinin zorunluluğu olduğu tartışılmaktadır. Bazen merkezi bir tek RSA modülünün ve network içerisinde yer her bir kişiye şifreleme , deşifreleme için (ei,di) çiftleri dağıtılarak networke girmeleri önerilmektedir. Bunula birlikte yukarıda (I)'de gösterildiği gibi bir (ei, di) çiftinin bilinmesi n modüllerinin çarpanlarının bulunmasına olanak verir ve böylece networkw giren herhangi bir kişi diğerlerinin deşifreleme üslerini belirleyebilir. Eğer

küçük bir mesaj networ içerisindeki 2 yada daha fazla kişiye gönderilirse, casus tarfından bazı genel bilgiler kullanılarak büyük bir oranda mesaj elde edilebilir.

(Vİİ) Devir Hırsızlıkları

c=m^e mod n şifre metni olsun. $c^{e^k} \equiv$c (mod n) olacak şekilde k pozitif tamsayı olsun. Şifreleme mesaj üzerinde k $\{0,1,...,n-1\}$ olacak şekilde (k'nınolması zorunlu) bir permatasyondur. Bu sebepten dolayı $c^{e^{k-1}} \equiv$ m (mod n) durumu olmalıdır. Bu gözlemler RSA şifrelemesi üzerinde devir hırsızlıklarına götürür. Hırsız, c'nin ilk halini elde edinceyi kadar c^e mod n, c^{e^2} mod n, c^{e^3} mod n sayılarını hesaplanır. Eğer c^{e^k} mod n=c ise bir önceki numara $c^{e^{k-1}}$ mod n olarak adlandırılır ve m açık metnine eşittir.

F=gcd($c^{e^u} - c, n$) > 1 olacak şekilde en küçük pozitif u tamsayısını bulmak için devir hırsızlıkları genellenir. Eğer, $c^{e^u} \equiv$ c (mod p) ve $c^{e^u} \not\equiv c$ (mod q) ise f=p aynı şekilde eğer

$c^{e^u} \not\equiv c$ (mod p) ve $c^{e^u} \equiv$ c (mod q) ise f=p olur.Diğer bir durumda n çarpanlara ayrılırsa, hırsız d'yi ve sonra m'i elde edebilir. Diğer taraftan her iki $c^{e^u} \equiv c$ (mod p) ve $c^{e^u} \equiv$ c (mod q) ise f=n olur. ve $c^{e^u} \equiv c$ (mod n) olur. e $c^{e^k} \equiv c$ (mod n) için u en küçük pozitif tam sayısı olmalıdır. Bu durumda temel devir hırsızlıkları başarılmış ve böylece Ve $m \equiv c^{e^{v-1}}$ mod n etkili bir şekilde hesaplanmış olur. (B.3)'ün (8.1 ve 8.2)'den daha az sıklıkla olması beklenir, genelleştirilmiş devir hırsızlıklarında, hırsızlık yapılmadan önce bitirilir. Bu sebepten dolayı genelleştirilmiş devir hırsızlıkları çarpanlara ayırma algoritmalarında gösterilmiştir.n sayısının çarpanlara ayırımı zor olduğu sürece, devir hırsızlıkları RSA şifreleme güvenliğinde tehlike oluşturmaz.

(Vİİİ) Mesajların Gizlenmesi

$0 \le m \le n-1$ olacak şekilde m açık metin mesajı RSA açık metin mesajı şifreleme içerisinde kendi kendini şifrelerse gizlenmemiş olarak adlandırılır, $m^e \equiv$ m (mod n) olur. (m=0, m=1 ve m= n-1) olduğu durumlarda her zaman gizlenmemiş mesaj vardır. Gerçekten gizlenmemiş mesajların sayısı:

[1+gcd (e-1, p-1)].[1+gcd (e-1, q-1)]

e-1, p-1 ve q-1'in hepsinin tam olduğu durumlarda gizlenmemiş mesaj sayısı en az 9'dur. Eğer p ve q rastgele asallarsa ve e küçük bir sayı olarak seçilmişse RSA'da şifrelenmiş gizlenmemiş mesajların oranı küçük olacaktır, böylece gizlenmemiş mesajlar RSA şifreleme güvenliğinde tehdit oluşturmayacaklartır.

5.13.2.6) Pratikte RSA Şifreleme

RSA şifreleme ve deşifrelemeyi hızlandırmanın yazımsal ve donanımsal olarak geliştirilmiş bir çok yolu vardır. Bu tekniklerin bir kısmı bölüm 14'te belirtilmiştir. Hızlı modüler çarpanlara ayırma (14,3), hızlı modüler üs alma (14,6), deşifreleme çın kalan teoreminin hızlı kullanımı (14,75)' te yer almaktadır. Bu gelişmelere rağmen RSA şifreleme / deşifreleme DES gibi genellikle kullanılan simetrik anahtar şifreleme algotirmalarında daha yavaştır. Pratikte RSA şifrelemesi simetrik anahtarın dönüşümünde şifreleme algoritması ve küçük verilerin şifrelenmesinde yaygın olarak kullanılmaktadır.

RSA şifreleme sistemi U.S.A ve Canada tarafından patentlenmiştir. Bazı standart organizasyonlar yazılmış veya yazma işlemi içinde şifreleme için RSA'nın şifreleme sistemlerinin kullanım adresleri, dijital imzalar ve anahtar kurulumu standartlaştırılmıştır.

Not: (Tavsiye edilen modül büyüklüğü): bülüm 3.2' deki tamsayıların çarpanlara ayrılması ile ilgili en son geliştirilmiş algoritmalar 512 bitlik n modülündedir ve planlanmış hırsızlıklara karşı sadece marjinal bir güvenlik sağlar. 1996'da işi bozmak için güçlü quadratic elek (3.2.6), sayı alan eleği (3.2.7), çarpanlara ayırma algoritmaları ve n modülü için enaz 768 bit tavsiye edilmiştir. Uzun dönemli güvenlik için 1024 bit ve daha geniş modüller kullanılmalıdır.

5.13.2.7) Asalların Seçilmesi

(i) 8.2.2'd bahsedildiği gibi, p ve q asalları öyle bir şekilde seçilmeli ki n=p.q olacak biçimde hesaplanması mümkün olmamalı. p ve q üzerindeki en büyük sınırlama eliptik eğri çarpan algoritmalarındadır (3.2.4). p ve q yeterli ve aşağı yukarı aynı bit uzunluğunda olmalıdır. Örneğin n modülü 1024 bit uzunluğundaysa p ve q 'nun her biri 512 bit uzunluğunda olmalıdır.

(ii) Diğer bir sınırlama ise p v q asalları arasındaki fark çok küçük olmamalıdır. Eğer p-q sayısı küçük olursa $p \approx q$ olur ve böylece $p \approx \sqrt{n}$ olur. Böylelikle $\sqrt{n}$ sayısına bütün tek sayılar deneme yoluyla bölünerek basit bir şekilde n sayısı elde edilir. Eğer p ve q rasgele seçilirse çok büyük ihtimalle p-q arasındaki fark büyük olacaktır.

(iii) Bu sınırlamara ilave olarak, bir çok yazar p ve q asallarının kuvvetli sayılar olmasını tavsiye etmişlerdir. Eğer aşağıdaki üç durum gerçekleşirse p kuvvetli bir sayıdır;

a) p-1 r ile gösterilen büyük bir asal çarpan

b) p+1 büyük bir asal çarpan

c) r-1 büyük bir asal çarpan olmalı

Kuvvetli asal çarpanlar üretmek için algoritma 4.2.2' de gösterilmiştir. a) koşulunun nedeni Pollard' ın verimliliği sadece p asalına dayanan ve p-1 düz sayı olan n modülünün işini zorlaştırmaktadır. b) koşulu bölüm 3.12' beliritlen verimliliği p asalına bağlı olan ve p+1 düz sayı olan (p+1) çarpan algoritmaların işini zorlaştırır. c) koşulu ise 8.2.2'de anlatılan devir hırsızlıklarını engellemektedir.

Eğer, p rastgele seçilmiş ve yeterli büyüklükte ise p-1 ve p+1'in ikisininde geniş asal çarpanlara sahip olması beklenir. Eğer güçlü asallar p-1 ve p+1 çarpan algoritmalarına karşı koruma sağlarlarsa, onları eliptik eğri çarpan algoritmalarına karşı koruyamazlar. Eğer rastgele seçilmiş sayı p ile aynı büyüklükte ise küçük asal çarpanlara sahiptir. Ve n çarpanını başarılı kılar. İlaveten 8.2.2 de belirtildiği gibi p ve q asalları rastgele seçilirse devir hırsızlıklarındaki şansın az olduğunuı gösterir. Bu nedenle güçlü asalların, rastgele seçilen asallardan daha az koruma sağladığı sunulmaktadır. Çatrpan algoritmaları ile ilgili verilen bilgilere, RSA anahtar üretiminde güçlü asalların kullanımın zorunlu olamadığı belirtilmiştir.

Diğer taraftan güçlü asallar rastgele asallardan daha az güvenlik sağlama zve çalışma zamanının hesaplamak için küçük eklemeler gereklidir.

Not (Küçük Şifreleme Üsleri):

i) Eğer e şifre üssü rastgele seçilirse

ii) Pratikte e şifreleme üssü e=3 olarak kullanılır. Bu durumda p-1 ve q-1 in üçe bölünebilmesi zorunlu değildir. Bu sonuçlar bir modüler çarpma ve rakamla bir modüler kare alma ilemi ile şifreleme işlemlerini çok hızlı hale getirir. Pratikte diğer şifreleme üsleri $e=2^{16} + 1= 65537$şeklinde kullanılır. Bu sayı binary gösteriminde iki tane bire sahiptir. Bu yüzden şifrelemede tekrarlı kare ve çarpma algoritmalarında gerekli olan 16 modüler kare alma ve 1 modüler çarpma kullanır. 8.2.2 de gösterilen hırsızlık çeşitlerine düzenlemede şifreleme üssünü $e=2^{16} + 1$ olması, e=3 olmasından

aynı mesaj gönderilen $2^{16}+1$ alıcı olması durumunda daha avantajlıdır.

5.13.3. Rabin Açık Anahtar Şifreleme

Herhangi bir şifreleme planında istenen özellik; ayrık logaritma ve tamsayı çarpanlara ayırma işlemlerinin, problemlerin çözümünün zorluğu ve buna bağlı olarak şifrenin kırılabilmesinin zorluğudur. RSA şifrelerinin kırılması n modülünün çarpanlarının zorluğuyla ilgili olduğuna inanılmaktadır. Rabin açık anhatr şifreleme taslakları, açık

anahtar şifreleme taslaklarının güvenliğinin kanıtlanmasının ilk örneğidir. Problem bir hırsızın şifreli metinden çarpanları hesaplayarak açık metni elde etmesi sırasında görülür.

5.13.3.1. Algoritma : Rabin Açık Anahtar Şifresi İçin Anahtar Üretme

Özet: Herkes açık anahtar ve buna uyan gizli bir anahtar üretebilir. Şu yol izlenmelidir;

1- Her biri eşit boyutta olan farklı, rasgele p ve q asal sayıları üretilmesi

2- n= p.q satısı hesaplanmalı

3- A'nın açık anahtarı n, gizli anahtarı ise (p.q)

5.13.3.2. Algoritma: Rabin Açık Anahtar Şifrelemesi

Özet: B m mesajını A için şifreler, A şifreyi çözer.

1. Şifreleme: B şu yolu izlemelidir.

a) A'nın açık anahtarı olan n'i ele geçirmeli

b) Mesajı {0,1,...n-1} aralığında m tamsayısı ile göstermeli

c) $c=m^2 \bmod n$ sayısını hesaplamalı

d) c şifreli metnini A'ya göndermeli

2- Şifre Çözme : C'den m açık metnini elde etmek için, A şu yolu izlemeli:

a) n^2 modülüne göre c'nin m_1, m_2, m_3, m_4 kareköklerini bulmak için algoritma 3.44 kullanılmalı

b) mesaj m_1, m_2, m_3, m_4 olarak kullanılabilir. A hangisinin büyük olduğuna karar verir.

Not : $p \equiv q \equiv 3 \pmod 4$ olduğundan n modülüne göre c'nin karekökünün bulunması. Eğer p ve q $\equiv 3 \pmod 4$ olacak şekilde seçilirse, c'nin n modülüne göre 4 karekökünün hesaplanması için şu yol izlenmelidir:

1) ap+bq=1 olacak şekilde a ve b sayılarını bulmak için uzayılmış oklit algoritması kullanılmalıdır. a ve b tüm anahtar üretim aşamaları için bir defa hesaplanabilir.

2) $r = c^{(p+1)/4} \bmod p$ sayısı hesaplanmalıdır.

3) $s = c^{(q+1)/4} \bmod q$ sayısı hesaplanmalıdır.

4) $x = (aps+bqr) \bmod n$ sayısı hesaplanmalıdır.

5) $y = (aps-bqr) \bmod n$ sayısı hesaplanmalıdır.

6) c'nin 4 kökü mod n'e göre x, -x, y ve –y' dir

5.13.3.3.Rabin Açık Anahtar Şifrelemenin Güvenliği

i) Hırsızla karşılaşıldığında yapılacak iş c şifre metninden m açık metnini elde etmektir. Bu bölüm 3.5.2'deki karekök problemleridir. n çarpanlarının problemleri ve n modülüne göre karekök hesaplama problemleri aynıdır. n faktörünün zor olduğu var sayılırsa rabin açık anhatrı şifreleme taslakları pasif hırsızlara karşı güvenlidir.

ii) Herhangi bir hırsıza karşı korunduğu sürece, rabin açık anahtar şifreleme taslakları seçilmiş şifreli metin hırsızlıklarına karşı yenilmemektedir. Bu tarz hırsızlıklar şu şekilde ortaya çıkmaktadır. Hırsız $m \in Zn^*$ olacak şekilde bir n tamsayısı seçer ve $c=m^2$ mod n sayısını hesaplanır. Sonra hırsız c' nin şifresini çözen A'nın deşifre makinasına c'yi gösterir ve y açık metnin geri verir. M rasgele seçildiği ve a tarafından bilinmediği zaman y açık metninin m ile aynı olması gerekli değildir. n'nin asal çarpanlarının gcd(m-y.n) olması durumunda ½ olasılıkla $y \not\equiv \pm m$ mod n olur eğer $y \equiv \pm m$ mod n olursa hırsızlık yeni bir m^3 ile tekrarlanr. (iii) Rabin açık anahtar şifreleme taslakları bölüm 8.2.2 (ii), 8.2.2(iii) ve 8.2.2(v)'te tanımlanan RSA il aynı hırsızlıklardan etkilenmektedir. RSA ile ilgili (ii) ve (iii) durumlarında hırsızlıklardan açık metin mesajına tuzlama yapılarak kaçınılabilir. (v) durumunda ise şifrelemeye uygum gereksiz eklemeler yapılarak hırsızlıklar önlenebilir.

Not: Redundancy'nin Kullanımı

(i) Rabin açık anahtar şifreleme taslaklarının dezavantajı alıcı kişinin 4 olasılık arasından doğru açık metni seçmesidir. Şifrelemede orijinal mesaja gereksiz eklemeler yapılarak deşifrelemedeki bu anlamsızlığın üstesinden gelinebilir. (örneğin en son 64 bitlik mesaj kopyalanabilir) Büyük bir olasılıkla, c yasal şifreli metnin m_1, m_2, m_3, m_4 kareköklerinden biri bu aşırılığa sahip olacaktır ve alıcı tasarlana bu açık metni seçebilecektir. Eğer kareköklerin hiç biri bu aşırılığa sahip değilse, alıcı c'yi hileli olarak reddetmelidir.

(ii) Eğer redundoncy yukaridaki gibi kullanılırsa, Rabin'in taslakları 8.13 (ii)'deki şifre hırsızlıklarından kolaylıkla etkilenmez. Eğer hırsız redundoncy içeren m mesajını seçerse ve $c = m^2$ mod n' i A'nın deşifre makinasına gönderirse, büyük bir olasılıkla makine yeni bir bilgi içermeyen m açık metnini hırsıza gönderecektir. (c'nin diğer 3 karekökü redundoncy içermediği sürece). Diğer taraftan eğer hırsız gereksiz eklemeler içermeyen m mesajının seşerse büyük bir olasılıkla $c=m^2$ mod n'in kareköklerinden hiçbiri bu aşırılıklara sahip olmayacaktır. Bu durumda deşifreleme makinası c'yi deşifrelemeyecek ve hırsız da herhangi bir yanıt veremeyecektir.

Eğer rabin deşifreleme iki işlemin birleşimi ise ilk olarak mod n'e göre c'nin 4 karekökü bulunur, 2i olarak önemli olan karekök açık metin olarak seçilir ve tutulan eşitlik kanıtlanır. Böylece rabin açı anahtar şifrelemesi gereksiz eklemeler yapılarak, pratik olarak uygun bir şekilde değiştirilir.

5.13.3.4. Örnek: (Küçük yapay parametrelerle rabin açık anahtar şifrelemesi)

i) Anahtar Üretimi: A kişsi p=277 ve q=331 olacak şekilde p ve q asallarını seçer n=p.q=91687'yi hesaplanır. A'nın gizli anahtarı (p=277,q=331) iken , açık anahtarı 91687'dir.

ii) Şifreleme: Farzedelim 8.14 (1)' de olduğ ugib orjinalmesajın son 6 bitinin şifrelemek için kopyalanması gerekiyor. 10 bitlik m=1001111001 mesajını şifrelemek için , B son 6 biti m'e ilave eder ve m=1001111001111001 olacak şekilde 16 bit olur ve desimal rotasyonu m=40569'dur. Sonra B $c=m^2 \bmod n= 40569^2 \bmod 91687= 62111$'i hesaplanır ve A'ya gönderir.

iii) Deşifreleme: Şifreyi çözmek için A algoritma 3.44'ü kullanır ve mod n'e göre c'nin 4 kareköknü n'nin çarpanlarının bilgisiyle hesaplanır.

m_1=69654 m_2=22033 m_3 = 40569 m_4= 51118

binary olarak

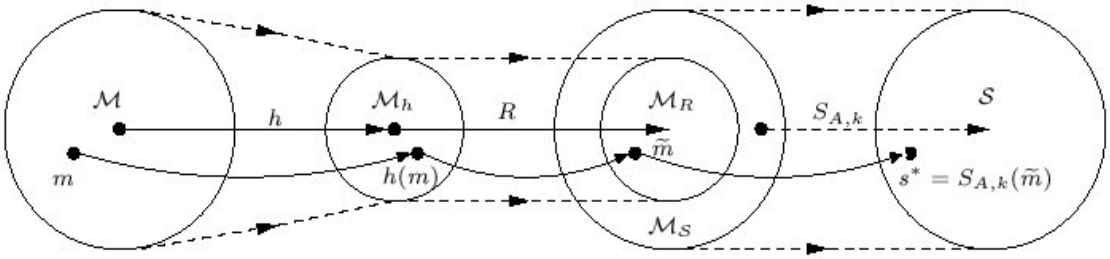
m_1= 10001000000010110 m_2= 101011000010001

m_3= 1001111001111001 m_4= 1100011110101110

m_3 mesajı aşırılığa sahip olduğundan, A m_3 için c'yi deşifre eder ve orijinal mesajdan m = 1001111001 sayısını elde eder.

Not (Verimlilik): Rabin şifrelemede sadece 1 tane modüler kare alma işlemi olduğundan işlemler oldukça hızlıdır. Karşılaştırma olarak olarak e = 3 olduğu RSA şifrelemesinde 1 modüler çarpma ve 1 modüler kare alma işlemi vardır. Rabin'in deşifrelemesi şifrelemmesinden daha yavaştır, RSA şifreleme hızı ile karşılaştırılabilir.

5.14.Not: (Mesajı geri almayı sağlayan projelerden ilaveli imzalar) Herhangi bir mesajı geri almalı dijital imza projeleri, basitçe hashlenen mesaj ve daha sonra işaretlenen hash değeri tarafından ilaveli dijital imza projesine çevrilebilir. Şimdi mesaj,Doğrulama(Gerçekleme) algoritmalarına girdi (entıty) olarak gereklidir. Bu durumun şeması 5.3'ten çıkarılır ve şekil 5.5'te anlatılır. Redundancy fonksiyon R'nin imza projesinin güvenliği için daha fazla önemi yoktur ve M_h'ten M_s'ye herhangi bir 1-1 fonksiyon olabilir.



Şekil .5.5. Mesajı geri almayı sağlayan projelerden ilaveli imzalar

5.2.4. İmza Projelerinde Saldırı Tipleri

Rakibin hedefi, diğer girdiler (entıty) gibi kabul edilecek üretilen imzaları taklit etmektir. Aşağıdaki maddeler bir imza projesini kırmanın anlamını belirten özelliklerin bir kümesini gösterir.

1- Tam kırmak: Bir rakip, geçerli işaretlenmiş algoritmaya fonksiyon olarak eşit olan işaretlenmiş etkili bir algoritmayı bulabilir veya işaretçinin özel anahtar bilgisini hesaplanır. (Örneğin, 5.3.2 (i).)

2- Sahte(taklit) seçim: Bir rakip, öncelikli seçilen mesaj sınıfı veya özel mesajı için geçerli bir imza oluşturabilir. Imza oluşumu direk yasal işaretçi içermez.(Örnek, 5,21)

3- Sahte(taklit) varlığı: Bir rakip en az bir mesaj için bir imza taklit edebilir. Rakip, kimin imzasını elde ettiğini bilmediğinden mesaj üzerinde az veya hiç kontrolü yoktur ve yasal işaretçi hile de içerebilir.(Not 5.66(iii)).

Açık anatar dijital imza projesine karşı iki temel saldırı vardır.

1- Sadece anahtar saldırıları : Bu saldırılarda rakip sadece işaretçinin açık anahtarını bilir.

2- Mesaj saldırıları : Burada rakip imzalara karşılık bilinen veya seçilen mesajlardan birini deneyebilir. Mesaj saldırıları 3 sınıfa ayrılmıştır.

a) Bilinen mesaj saldırıları : Rakip, bilinen mesaj kümesi için imzalara sahiptir. Bunları bilir fakat onun tarafından seçilemez.

b) Seçilen mesaj saldırıları : Rakip,imza projesini kırma teşebbüsünden önce mesajların seçilmiş bir listesinden geçerli imzalar elde eder. Bu saldırı,imzalar görülmeden önce mesaj seçilmelidir anlamında uyarlanamaz. Imza projelerine karşı seçilen mesaj saldırıları, açık anahtar imza şifreleme projelerine karşı seçilmiş ciphertext saldırılarına benzerdir.

c) Uyarlanarak seçilmiş mesaj saldırıları : Rakip bir oracle olarak işaretçi kullanımını kabul eder. Rakip, işaretçinin açık anahtarına dayanan mesaj imzalarını isteyebilir ve o, özellikle sağlanan imza veya mesajlara dayanan mesaj imzalarını isteyebilir.

5.15. Not: (Uyarlanarak seçilmiş mesaj saldırısı) Prensipte, uyarlanarak seçilmiş mesaj saldırısını önlemek için en zor saldırı tipidir. Bir rakip, bir model çıkartabilir ve daha sonra seçtiği imzayı işleyebilir. Uyarlanarak seçilmiş mesaj saldırısı pratikte kırmak için mümkün olamadığında en iyi tasarlanmış imza projesi imkana rağmen korumak için tasarlanmalıdır.

5.16. Not: (Güvenlik faktörleri) Djital imza projesinde gereken güvenlik seviyesi uygulamayla ilişkili olarak değiştirilebilir. Örneğin; rakibin sadece tek anahtar saldırısı kurumunda yetenekli olduğu durumlarda, sahte(taklit) seçimde başarılı olandan rakibi korumak için projeyi tasarlamak yeterli olabilir. Rakibin mesaj saldırısında yetenekli olduğu durumlarda sahte(taklit) varlık'ın imkanına rağmen korumak gereklidir.

5.17. Not: (Hash fonksiyonları ve dijital imza işlemleri) Bir hash fonksiyon h, dijital imza projesinde kullanıldığında, h imza işleminin sabit bir bölümü olmalıdır. Böylece rakip geçerli imzayı alamaz, h ile zayıf bir h fonksiyonu yer değiştiremez ve sahte(taklit) seçim saldırısını kıramaz.

5.3.RSA ve İLGİLİ İMZA PROJELERİ

Bu bölüm RSA imza projesini ve diğer ilişkili metotlrı anlatır. Burada gösterilen projelerin güvenliği faktöriyel alma problemlerinin intractability 'sinde büyük dereceye dayanır. Gösterilen projeler ilaveli ve mesajı geri alma dijital imzaların he ikisi kapsar.

5.3.1. RSA İmza Projesi

RSA açık şifreleme projelerinin mesaj aralığı ve ciphertext aralığının her ikisi rastgele seçilen farklı iki asal sayının ürünü olan n=pq iken $Z_n=\{0,1,2,...,n-1\}$'dir. Şifreleme dönüşümleri 1-1,örten olduğunda, dijital imzalar şifreleme ve şifre çözme rollerinin tersiyle oluşturulabilir. RSA imza projesi mesajı geri almaın sağladığı, deterministik dijital imza projesidir. İşaretlenmiş uzay M_s ve imza aralığı S her ikisi de Z_n'dir. $R:M \rightarrow Z_n$ seçilmiş ve açık bilgidir.

5.18. Algoritma: RSA İmza Projesi İçin Anahtar Üretimi

ÖZET: Her girdi(entıty) RSA açık anahtar ve karşılığı olan özel anahtarları oluşturur. Her A girdisi(entıty) aşağıdaki gibi olmalıdır.

1- Aynı büyüklükteki rastgele iki farklı M asal sayı p ve q meydana getirilir.

2- N=pq ve Φ=(p-1)(q-1) hesaplanır.

3- Gcd(e, Φ)=1 şeklinde 1<e< Φ aralığında rastgele bir e sayısı seçilir.

4- $ed \equiv 1 (\mathrm{mod}\, \Phi)$ şeklinde 1<d< Φ aralığında tek tamsayı d'yi hesaplamak için genişletilmiş Eucliden algoritma kullanılır.

5- A'nın açık anahtarı (n,e) ; A'nın özel anahtarı d'dir.

5.19. Algoritma: RSA İmza Üretimi ve Doğrulama(Gerçekleme)

ÖZET: A girdisi (entıty), $m \in M$ mesajını işaretler. Herhangi bir B girdisi(entıty) A'nın imzasını soruşturabilir ve imzadan m mesajına çevirebilir.

1- İmza üretimi: A girdi(entıty)i aşağıdaki gibi olmalıdır.

a) $\tilde{m}$ =R(m), [0,n-1] aralığında bir tamsayı hesaplanır.

b) $S=\tilde{m}^{d} \bmod n$ hesaplanır.

c) M için A'nın imzası s'dir.

2- Doğrulama(Gerçekleme) : A'nın imzası s'yi değiştirmek ve mesaj m'yi çevirmek için B şöyle olmalıdır:

a)A'nın doğruluk açık anahtarı (n,e) sağlanır.

b) $\tilde{m}=s^{e} \bmod n$ hesaplanır.

c) $\tilde{m} \in M_R$ doğrulanır, eğer değilse imza reddedilir.

d) $M=R^{-1}(\tilde{m})$ geri alınır.

İmza Doğrulama(Gerçekleme) çalışmalarının kanıtı; eğer s, mesaj m için bir imzaysa $\tilde{m}$=R(m) iken $S=\tilde{m}^{d} \bmod n$' dir. $ed \equiv 1 (\mathrm{mod}\, \Phi)$ iken $s^{e} \equiv \tilde{m}^{ed} \equiv \tilde{m} \pmod n$' dir. Sonuç olarak

$R^{-1}(\tilde{m})=R^{-1}(R(m))=m$

5.20. Örnek: Küçük Parametrelerle RSA İmza Üretimi

Anahtar Üretimi: A girdisi (entıty) p=7927, q=6997 asallarını seçer ve n=pq=55465219 ve Φ=7926*6996=55450296 hesaplanır. A, e=5 seçer ve $ed=5d \equiv 1 \pmod{55450296}$ çözer, d=44360237 meydana çıkar. A'nın açık anahtarı (n=55465219, e=5); A'nın özel anahtarı d=44360237'dir.

İmza Üretimi: $M=Z_n$ narsayılır ve $R: M \rightarrow Z_n$, tüm $m \in M$ için birim dönüşüm R(m)=m'dir. m=31229978 mesajını işaretlemek için A $\tilde{m}$=R(m)=31229978 hesaplanır ve $s=\tilde{m}^{d} \bmod n=3122978^{44360237} \bmod 55465219=30729435$ imzasını hesaplanır.

İmza Gerçekleme (doğrulaması): B, $\tilde{m}=s^{e} \bmod n=30729435^{5} \bmod 55465219=31229978$ hesaplanır. Sonuç olarak B, $\tilde{m}$, M_R 'nin elemanı olduğundan ve

$m=R^{-1}(\tilde{m})=31229978$ geri alındığından ötürü imzayı kabul eder.

5.3.2. RSA İmzalar üzerinde mümkün olababilen saldırılar

i) Tamsayı faktörü

Eğer rakip, birkaç A girdisinin (entıty) açık modülü n'in faktörünü alabiliyorsa rakip Φ'yi hesaplayabilir ve genişletilmiş Euclidean algoritma kullanılarak Φ'den özel anahtar d'çıkarılır ve $ed \equiv 1 (\mathrm{mod}\, \Phi)$ çözümüyle açık üs e bulunur. Bu sistemin toplam kırılması demektir. Buna karşı koruma için A, p ve q'yu öyle seçmelidir ki böylece faktörü alınan n çarpımını hesaplamak mümkün olmasın.

ii) Çarpılabilir RSA Özelliği:

RSA imza projesi aşağıdaki çarpılabilir özelliğe sahiptir. Bazen homomorphic özellik gibi gösterilir. Eğer, $s_1 = m_1^d \bmod n$ ve $s_2 = m_2^d \bmod n$ değerleri m_1 ve m_2 üzerindeki imzalarsa $s = s_1 s_2 \bmod n$'den $s = (m_1 m_2)^d \bmod n$ şeklinde özelliğe sahiptir. Eğer, $m = m_1 m_2$ düzgün redundancy'ye sahipse($m \in M_R$ gibi), s, onun için geçerli bir imza olacaktır. Bu nedenle, redundancy fonksiyon R'nin tüm $a,b \in M$ çiftleri için $R(a,b) \neq R(a)R(b)$ olması önemlidir. Örnek 5.21'de gösterildiği gibi R üzerindeki bu durum güvenlik için gereklidir fakat yeterli değildir.

5.21. Örnek: (Emniyetsiz redundancy fonksiyon**)** n, bir RSA modülü ve d özel anahtar olsun. k=[logn], n'nin bit uzunluluğu olsun ve t, t<k/2 şeklinde sabit pozitif tamsayı olsun. $w = 2^t$ ve mesajlar $[1.n2^{-t} - 1]$ aralığında m tamsayıları olsun. Redundancy fonksiyon R, $R(m) = m2^t$ olarak alınır. N'nin çoğu seçimleri için R çarpılabilir özellikte olmayacaktır.

Not 5.10' da açıklanan genel sahte(taklit) varlık saldırısı $\left(\frac{1}{2}\right)^t$ olasılığında başarı imkanına sahip olabilecektir. Fakat bu redundancy fonksiyon için sahte(taklit) seçim saldırısı mümkündür.

Rakibin mesaj m üzerinde bir imzayı taklit etmek istediğini varsayalım. Rakip n'yi bilir fakat d'yi bilmez. Rakip, m üzerinde imzayı elde etmek için takip eden seçilmiş mesaj saldırısını kurabilir. Bunun için, n'ye ve $\tilde{m} = R(m) = m2^t = mw$'e genişletilmiş Euclidean algoritma uyugulanır. Genişletilmiş Euclidean algoritmasının her safhasında x,y ve r tamsayıları $xn + y\tilde{m} = r$ şeklinde hesaplanır. Algoritmanın her aşamasında, Eğer y>o ise $m_2 = rw$ ve $m_3 = yw$ tamsayıları $w \leq \sqrt{n}$, $r < n/m$ ve $|y| < n/m$ olacak şeklinde y ve r sayıları mevcuttur. Eğer,y<0 ise $m_2 = rw$ ve $m_3 = -yw$ tamsayıları düzenlenir. Her iki sonuçta m_2 ve m_3 gereken redundancy'ye sahiptir. Eğer, imzalar $s_2 = m_2^d \bmod n$ ve $s_3 = m_3^d \bmod n$ imzaları yasal işaretçiden sağlanıyorsa, rakip m için bir imzayı aşağıdaki gibi hesaplayabilir.

- Eğer y>0 ise $\frac{s_2}{s_3} = \frac{m_2^{\,d}}{m_3^{\,d}} = \left(\frac{rw}{yw}\right)^d = \left(\frac{r}{y}\right)^d = \tilde{m}^{\,d} \bmod n$ hesaplanır.

- Eğer y<0 ise $\frac{s_2}{-s_3} = \frac{m_2^{\,d}}{(-m_3)^d} = \left(\frac{rw}{yw}\right)^d = \left(\frac{r}{y}\right)^d = \tilde{m}^{\,d} \bmod n$ hesaplanır.

Her iki durumda, rakip gereken redundancy ile onun seçimi işaretlenmiş mesaja sahiptir. Bu saldırı, sahte(taklit) seçimi sağlayan seçilmiş mesaj saldırısının bir örneğidir. Redundancy fonksiyon R'nin seçimi akıllı olmalıdır.

5.3.3.Pratikte RSA İmzaları

i)Problemi Yeniden Bloke Etmek

Önerilen bir RSA kullanımı, mesajı işaretlemek ve daha sonra imza sonucunu şifrelemektir. Bu prosedür tamamlandığında içerdiği modülün boyutları ile ilgili olmalıdır. A'nın, B'nin mesajını işaretlemek daha sonra şifrelemek istediğini düşünelim.(n_A,e_A) ve (n_B,e_B) 'nin sırasıyla A'nın ve B'nin açık anahtarları olduğunu varsayalım. $n_A > n_B$ ise, örnek 5.22'de gösterildiği gibi B tarafından mesajın geri alınmaması gibi bir şans vardır.

5.22 Örnek: n_A=8387 * 7499 = 6289453 e_A = 5 ve d_A = 37726937 ve n_B = 54465219 e_B = 5, d_B = 44360237 olsun. $n_A > n_B$ olduğu unutulmamalıdır. m= 1368797, A'nın özel anahtarı altında işaretlenmiş ve daha sonra B'nin açık anahtar kullanarak şifrelenmiş olduğu için redundancy bir mesajdır. $m \neq \tilde{m}$ incelenir. Bunun sonucunda s, modül n_B'den daha büyüktür. Burada, bu problemin ortaya çıkma olasılığı $\mathbf{(n_A - n_B)/\, n_A \approx 0.12}$'dir.

Problemi yeniden bloke etmenin üstesinden gelmek için çeşitli yollar vardır.

1- Yeniden Sıralama: Yanlış şifre çözme problemi ise daha küçük modüllerin kullanıldığı işlemler ilk olarak uygulanırsa asla oluşmayacaktır.Yani, eğer $n_A > n_B$ ise girdi (girdi(entıty)) A B'nin açık anahtarının kullandığı mesajı öncelikle şifreler, daha sonra A'nın özel anahtarının kullandığı sonuçlanan ciphertexti işaretler. Tercih edilen işlem sırası yine de her zaman öncelikle mesajı işaretlemek sonra imzayı şifrelemektir. A'nın önce şifreleyip sonra işaretlemesi için rakip imzayı taşıyabilir ve kendi imzasıyla onu yerdeğiştirebilir. Rakip neyin işaretlenmiş olduğunu bilmese bile, rakip için bunun avantajlı olduğu durumlar da olabilir. Böylece yeniden sıralama tedbirli bir çözüm değildir.

2- Girdi(Entity) Başına İki Modül: Her bir girdi (entıty), şifreleme ve işaretleme için ayrı modüller üretmeye haizdir. Eğer, her kullanıcının işaretlenmiş modülleri mümkün olan şifreleme modüllerinin hepsinden daha küçükse, şifre çözme mümkün olmaz. Bu

işaretlenmiş modül t-bit sayıları ve (t+1) bit sayıları için gereken şifreleme modülü tarafından garantilenebilir.

3- Modüllerin Tavsiye Edilen Formları: Bu metotta, p ve q asalları seçilir. Şöyle ki, n modülünün özel bir formu en yüksek-mertebeli bit 1'dir ve takip eden k bitin tümü 0'dır.Bu formun t-bitli n modülü aşağıdaki gibi bulunabilir. n için $2^{t-1} \le n < 2^{t-1}+2^{t-k-1}$ aralığında gereken forma sahip olmaktır. Rastgele [t/2] bitlik p asalı seçilir ve $[2^{t-1}/p]$ ve $[(2^{t-1}+2^{t-k-1})/p]$ aralığında bir q asalı için arama yapılır. Daha sonra, n=pq gereken tipin bir modülüdür.(5.23) n modülü için bu seçim, yanlış şifre çözme problemini tamamen önlemesede ihmal edilebilecek kadar küçük bir sayının ortaya çıkma olasılığını azaltabilir. n_A' nın böyle bir mpdül olduğunu ve $s = m^{d_A} \bmod n_A$'nın m üzerinde bir imza olduğunu varsayalım. Yüksek mertebeli k+1 bit pozisyonlarından birinde s'nin 1 olduğunu farzedelim. Daha sonra ,s, n_A'dan daha küçük olduğunda yüksek-mertebeli bit pozisyonunda 0 olmalıdır ve böylece benzer bir formdaki herhangi diğer modüllerinden daha küçüğü gereklidir. s'nin yüksek mertebeli (k+1) bit pozisyonlarında 1 olmama olasılığı, k, 100 civarında seçildiyse $\left(\frac{1}{2}\right)^k$ 'dan daha küçüktür ve ihmal edilebilir.

5.23 Örnek: Yüksek mertebeli bitin 1 olduğu ve sonraki k=3 bitin 0 olduğu 12 bitlik n modülünün oluşturmak istendiğini düşünelim. 6-bitlik p=37 asalının seçimiyle başlanır. $[2^5/p]=56$ ve $[(2^4+2^8)/p]=62$ aralığında bir q asalı seçilir. q'nun olasılıkları 59 ve 61'dir. q=59 seçilirse n=37*61=2257'dir. İkilik gösterimi 10005010001'dir.

ii) Redundancy Fonksiyonlar: RSA imza projesi üzerinde sahte(taklit) varlık saldırısından (5.2.4) sakınmak için uygun redundancy fonksiyon R gereklidir. Bölüm 5.3.5'te uluslararası standart olarak kabul edilen böyle bir fonksiyon tarif edilir. Redundancy fonksiyonunun uygun seçimi sistemin güvenliği için önemlidir.(5.3.2.ii)

iii) Ilaveli Dijital İmza Projesi:

5.14' te tarif edilen mesajı geri almalı dijital imza projesi ilaveli bir dijital imza projesini vermek için düzenlenebilir. Örneğin; MD5 128 uzunluklu bit dizileri için rastgele bit uzunluklu mesajları hash'lemekte kullanılıyorsa algoritma 5.9 hash değerlerini işaretlemekte kullanılabilir.Eğer, n, k-bit RSA modülü ise uygun redundancy fonksiyon R, 128-bit tamsayıları k-bit tamsayılara atamak için gereklidir.5.3.6'da pratikte sıkça kullanılan bu durum için bir yöntem tarif edilmiştir.

iv) İmza üretimi ve gerçeklemenin (doğrulama) performans karakteristikleri

p ve q'nun her biri k-bit asallar iken n=pq 2k-bit RSA modülü olsun. modüler üs alma için ve m mesajı $O(k^3)$ bit işlemleri çarpımını gerektirdiği için $s=m^d$ modn imzası hesaplanır. İşaretçi genellikle p ve q'yı bildiğinde $s_1=m^d$ mod p, $s_2=m^d$ mod q hesaplayabilir ve Çinkalan Teoremi kullanılarak s tespit edilebilir. Bu prosedürün $O(k^3)$ karmaşıklığına rağmen bazı durumlarda daha etkilidir.

Açık üs küçük bir sayı için seçilmişse imza gerçeklemesi (doğrulama) işaretlemekten daha hızlıdır. Bu yapılırsa, Doğrulama(Gerçekleme) ,$O(k^2)$,bit işlemleri gerektirir. e için önerilen değerler pratikte 3 veya $2^{16}+1$'dir. p ve q **gcd(e,(p-1)(q-1))=1** şeklinde seçilmelidir.

RSA imza projesi, düzenlenen üstün işlemlerin imza doğrulaması olduğu durumlara uydurulur. Örneğin; güvenilir 3. Grup bir A girdisi (entıty) için açık anahtar belgesi oluşturulduğunda bu sadece imza üretimini gerektirir ve bu imza çeşitli girdilerle birkaç kez doğrulanabilir.

v) Parametre seçimi: RSA imza modülü için minimum 768 bit tavsiye edilir. En az 1024 bitlik modüller, daha uzun yaşam süresine ihtiyaç veya geniş bir networkün güvenliğinin önemli olduğu imzalar için önerilir.Bildirilen RSA imza projesinde açık üse 3 veya $2^{16}+1$ gibi küçük bir sayı seçildiğinde kuvvetsiz değildir. İmza üretiminin etkinliğini arttırmak için özel üs d'nin boyutunu sınırlamak önerilmez.

vi) Band genişliği etkisi: Mesajı geri almalı dijital imza için band genişliği etkisi,

Log_2 M_S'nin Log_2 M_R'ye oranını gösterir.Burada, M_S, işaratleme uzayı iken M_R redundancy fonksiyonu R'nin görüntü uzayıdır. Bununla beraber band genişliği etkisi, redundancy R ile tespit edilir. RSA için ISO/IEC 9796 tarafında özelleştirilen redundancy fonksiyon k-bit mesajları alır ve uygulanan 2k-bit imzadan M_S içindeki 2k-bit elemanlarını çözer. Band genişliği etkisi, sonuçta ½'dir. Örneğin, 1024 bit boyutundaki modüller ile,işaretlenebilen maximum boyutlu mesaj 512 bittir.

vii) Sistem-geniş(açık) parametreleri: Her girdi(entıty) farklı RSA modülüne sahip olmalıdır. Sistem-geniş modüllerini kullanmak emniyetsizdir. Açık üse,sistem açık parametresi olabilir ve birkaç uygulamada olabilir.

viii) Kısa veya uzun mesajlar: k-bit mesajlarını işaretlemek için Algoritma. 5.19'da kullanılan 2k-bit RSA modüllerinin n olduğunu varsayalım. A girdisinin(entıty) kt-bit mesaj m'i işaretlemek istediğini düşünelim. $M=m_1 \| m_2 \| \ldots \| m_t$ şeklindeki k-bit blokları içinde m'i bölmek ve her bloğu işaretlemek bir yoldur. Bunu için bant genişliğinin ihtiyacı

2kt-bit'tir. $1 \leq k$ uzunluklu bir bit dizisi için A, m mesajını hashleyebilir ve hash değerini işaretleyebilir. Bu imzanın bant genişliği ihtiyavı gönderilen m mesajından gelen kt süresinde kt+2k 'dır. Ne zaman $t \geq 2$ iken kt+2k $\leq 2kt$ olduğunda bant genişliğinin etkili metotu : ilaveli RSA dijital imzalarını kullanmaktır. En fazla k-bit boyutundaki bir mesaj için mesajı geri almalı RSA önerilir.

5.3.4. Rabin açık-anahtar imza projesi:

Rabin açık anahtar imza projesi RSA'ya benzer(5.19) fakat, üs olarak **e** kullanır. Kolaylık için e=2 olduğunu varsayalım. İşaretleme uzayı M_S,Q_N(Kuadrdtik Rezidü) ve imzalar bunların karekökleridir. Mesaj aralığı M'den M_S'ye olan R redundancy fonksiyonu seçilir ve açık bilgidir.

Algoritma 5.25 Rabin açık anahtar imza projesine basit bir versiyonunu anlatmaktadır. Pratikte daha yararlı ve kullanışlı olan versiyon algoritma 5.30'da gösterilir.

5.24. Algoritma: ve Rabin Açık Anahtar İmza Projesi için Anahtar Üretimi

Özet: Her girdi(entıty) bir açık anahtar ve karşılığı olan özel anahtar oluşturur. Her A girdisi (entıty) aşağıdakileri yapmalıdır.

1- Her biri yaklaşık aynı boyutta iki farklı rastgele p ve q asal sayıları üretilir.
2- n=p.q hesaplanır.
3- A açık anahtarı n, A'nın özel anahtarı (p,q) dur.

5.25. Algoritma: Rabin İmza Üretimi ve Doğrulama(Gerçekleme):

Özet: A girdisi(girdi(entıty)) $m \in M$ olan bir imza işaretler. Herhangi bir B girdisi (entıty) A'nın imzasını doğrular ve imzadan m mesajını geri alır.

1- **İmza üretimi:** Girdi (entıty) A, aşağıdakileri yapmalıdır.
   a) $\tilde{m} = R(m)$ hesaplanır.
   b) $\tilde{m}$ mod n 'nin bir karekökü (s) hesaplanır.
   c) m için A'nın imzası s'dir.
2- **Doğrulama(Gerçekleme):** A'nın imzası s'yi doğrulamak(gerçeklemek) ve mesaj m'yi geri almak için B şunları yapmalıdır.

   a) A'nın doğru açık anahtarı n elde edilir.
   b) $\tilde{m} = s^2$ mod n hesaplanır.
   c) $\tilde{m} \in M_R$ doğrulanır. Değilse imza redddedilir.
   d) $M = R^{-1}(\tilde{m})$ geri alınır.

5.26. Örnek: Anahtar üretimi: Girdi (entıty) A, p=7 ,q=5 asallarını seçer ve n=p.q= 77 olur. A'nın açık anahtarı n=77, özel anahtarı (p=7,q=5) 'dir. işaretleme uzayı

$M_S = Q_{77}$ = {1,4,9,15,16,23,25,36,37,53,58,60,64,67,71} 'dir. Kolaylık için $M=M_S$ ve redundancy fonksiyon R,(Not.5.27 dikkate alınarak) birim dönüşüm alınır.

İmza üretimi: m=23 mesajını işaretlemek için A, $R(m) = \tilde{m} = 23$ hesaplanır ve daha sonra $\tilde{m}$ modülü 77'nin karekökünü bulur.Eğer,böyle bir s karekök olarak alınırsa,

$s \equiv \mp 3 (\text{mod } 7)$ ve $s \equiv \mp 1 (\text{mod } 5)$'dir s=10,32,45 veya 67'ye karşılık gelirBu dört kökten herhangi birisi alınabilir.. m'nin imzası örneğin,s=45 olarak seçilir.

İmza Doğrulaması(Gerçekleme): B, $\tilde{m} = s^2$ mod 77 = 23 hesaplanır. $\tilde{m} = 23 \in M_R$ olduğundan m imzayı kabul eder $m=R^{-1}(\tilde{m}) = 23$'ü geri alır.

5.27. Not:

i) RSA(5.21), imza projesi ile redundancy fonksiyon R'nin uygun seçimi Rabin imza projesinin güvenliği için önemlidir. Örneğin; tüm $m \in M$ için $M=M_S=Q_n$ ve R(m) = m olduğunu varsayalım. Eğer,, rakip herhangi bir $s \in Z_n^*$ tam sayısı seçer ve $\tilde{m} = s^2$ mod n alınarak karekökünü hesaplanırsa, s , $\tilde{m}$ için geçerli bir imzadır ve özel anahtar bilgisi olmadan sağlanır. Bu durumda, sahte(taklit) varlık önemsiz hale gelir.

ii)Mesajı geri almalı dijital imza projesinin en pratik uygulamaları , M mesaj uzayının bazısı sabit uzunluklu bit dizilerinden oluşmasıdır. Rabin projesi için belirlenecek redundancy fonksiyon R, örneğin, m mesajı bir bit dizisi ise R ikilik gösterimindeki mesajı tam sayı atayabilir. Sonuç, tam sayının modül n'nin kuadratik bir rezidüsü olduğunun garantisi yoktur. Böylece, karekök hesabı mümkün değildir. m için rastgele bitlerin küçük bir sayı eklemesini deneyebilir $R(m) \in Q_n$ umuduyla tekrar R'yi uygulayabiliriz. Ortalama, iki kez deneme yetecektir. Fakat deterministik metot tercih edilebilir.

Değiştirilmiş Rabin İmza Projesi:

Not 5.27(ii)'de verilen problemin üstesinden gelebilmek için temel Rabin imza projesinin değiştirilmiş versiyonu kulanılır. Tarif edilen teknik ISO/IEC 9796 'da kullanılan dijital imza standartına(5.3.5) benzerdir. İşaretleme uzayı M_s'de elemanlarla mesajları birleştirmek için karekök hesabının her zaman mümkün olduğu deterministik bir metod kullanılır.

5.28: p ve q herbiri 3mod4'e denk olan farklı asallar ve n=pq olsun.

i) Eğer, gcd(x,n)=1 ise $x^{(p-1)(q-1)/2} \equiv 1 (\text{mod } n)$ 'dir.

ii) Eğer,$x \in Q_n$ ise $x^{(n-p-q+5)/8}$ mod n (x modül n)'in bir kareköküdür.

iii) x, jacobi sembolü $\left(\frac{x}{n}\right)$=1 'e sahip bir tamsayı olsun ve d=(n-p-q+5)/8 alalım.

Buna göre, $x^{2d} \bmod n = \begin{cases} x, & x \in Q_n \\ n-x, & x \notin Q_n \end{cases}$

vi) Eğer, $p \neq q \pmod 8$ ise $\left(\frac{2}{n}\right)$=-1 'dir. Bununla beraber, herhangi bir x sayısının 2 ile veya 2^{-1}(mod n) ile çarpımı x'in jacobi sembolünün tersidir. Ayrıca,n=p.q olan $p \equiv q \equiv 3 \pmod 4$ ve $p \neq q \pmod 8$ şeklindeki sayılara **Williams** tamsayılarıda denir.

Algoritma 5.30, Rabin dijital imza projesinin değiştirilmiş bir versiyonudur. İşaretlenen mesajlar $M_S = \{m=Z_n : m \equiv 6 \pmod{16}\}$ 'dendir. Tablo 5.2'de semboller ile gösterimi verilmiştir. Pratikte, redundancy fonksiyon R sahte(taklit) varlığı önlemek için daha karışık olmalıdır. (örnek için 5.3.5'e bakılabilir.)

Tablo 5.2 : Algoritma.5.30'daki fonksiyon ve kümelerin tanımı

Notasyon	Anlamı	Tanımlama
$\mathcal{M}$	Mesaj Uzayı	$\{m \in Z_n : m \leq ((n-6)/16)\}$
$\mathcal{M}_S$	İşaretleme Uzayı	$\{m \in Z_n : m \equiv 6 \pmod{16}\}$
$\mathcal{S}$	İmza Uzayı	$\{s \in Z_n (s^2 \bmod n) \in \mathcal{M}_S\}$
$\mathcal{R}$	Redundancy Fonksiyon	Her $m \in \mathcal{M}$ *için* $R(m)=16m+6$
$\mathcal{M}_R$	$\mathcal{R}$'nin Görüntüsü $\mathcal{M}_R$ = **Im** ($\mathcal{R}$)) .	$\{m \in Z_n : m \equiv 6 \pmod{16}\}$

5.29. Algoritma:Değiştirilmiş Rabin imza projesi için anahtar üretimi:

Özet: Her girdi(entıty) açık anahtarı ve karşılığı özel anahtarı oluşturur. Her A girdisi(entıty) aşağıdaki yolu izlemelidir:

1- Rastgele $p \equiv 3 \pmod 8$,$q \equiv 7 \pmod 8$ asallarını seçer ve n=pq hesaplanır.
2- A'nın açık anahtarı n, özel anahtarı d=(n-p-q+5)/8'dir.

5.30. Algoritma: Değiştirilmiş Rabin açık-anahtar imza üretimi ve Doğrulaması(Gerçekleme)

Özet: A girdisi(entıty),herhangi $m \in M$ mesajını işaretler. Herhangi bir B girdisi(entıty), A'nın imzasını doğrulayabilir ve imzadan m mesajını geri alabilir.

1- İmza üretimi: A girdisi(entıty) aşağıdakileri yapmalıdır:

a) $\tilde{m}$ =R(m)=16m+6'yı hesaplanır.

b) Jacobi sembolü $J=\left(\frac{\tilde{m}}{n}\right)$ hesaplanır.

c) J=1 ise $s=\tilde{m}^{d} \bmod n$ hesaplanır.

d) J=-1 ise $s=(\tilde{m}/2)^{d} \bmod n$ hesaplanır.

e) A'nın imzası m için s'dir.

2- Doğrulama(Gerçekleme): A'nın s imzasını doğrulamak(gerçeklemek) ve mesaj m'yi geri almak için girdi B şunları yapmalıdır.

a) A'nın gerçek açık anahtarı n'i elde etmelidir.

b) $m' = s^2 \bmod n$ hesaplanmalıdır.

c) Eğer,$m' = 6 \pmod 8$ ise $\tilde{m} = m'$ almalıdır.

d) Eğer,$m' = 3 \pmod 8$ ise $\tilde{m} = 2m'$ almalıdır.

e) Eğer,$m' = 7 \pmod 8$ ise $\tilde{m} = n-m'$ almalıdır.

f) Eğer,$m' = 2 \pmod 8$ ise $\tilde{m} = 2(n-m')$ almalıdır.

g) $\tilde{m} \in M_R$ doğrulamalıdır(Gerçekleme) olmuyorsa imza reddedilir.

h) $m = R^{-1}(\tilde{m}) = (\tilde{m}-6)/16$ geri alınmalıdır.

İmza Doğrulama(Gerçekleme) çalışmalarının ıspatı: İmza üretimi safhası, jacobi sembol 1'e sahip olan $v=\tilde{m}$ veya $v=\tilde{m}/2$ 'den birisini işaretler. 5.28 (iv)'de $\tilde{m}, \tilde{m}/2$ biri jacobi sembolü 1'e sahiptir. İşaretlenen v değeri $v \equiv 3$ veya 6(mod 8) 'dir. 5.28(iii)'de $v \in Q_n$ olup olmamasına bağlı olarak $s^2 \bmod n = v$ veya n-v 'dir. $n \equiv 5 \pmod 8$ iken bu durum nadir olarak ayrılır.

5.31. Örnek:Küçük Parametrelerle Değiştirilmiş Rabin İmza Projesi

Anahtar üretimi: A, p=19,q=31'i seçer ve n=pq =589 olur. d=(n-p-q-+5)/8=68 hesaplanır. A'nın özel anahtarı d=68 iken açık anahtarı n=589'dur.

İşaretleme uzayı M_S, her elemanın jacobi sembolü ile aşağıdaki tabloda verilmiştir.

m	6	22	54	70	86	102	118	134	150	166
$\left(\frac{m}{589}\right)$	−1	1	−1	−1	1	1	1	1	−1	1
m	182	198	214	230	246	262	278	294	326	358
$\left(\frac{m}{589}\right)$	−1	1	1	1	1	−1	1	−1	−1	−1
m	374	390	406	422	438	454	470	486	502	518
$\left(\frac{m}{589}\right)$	−1	−1	−1	1	1	1	−1	−1	1	−1
m	534	550	566	582						
$\left(\frac{m}{589}\right)$	−1	1	−1	1						

İmza üretimi: m = 12 mesajını işaretlemek için A girdisi $\tilde{m}$ =R(12)=198, $\left(\frac{\tilde{m}}{n}\right)=\left(\frac{198}{589}\right)=$ 1 ve $s = 198^{68} \bmod 589=102$ hesaplanır. m=12 için A'nın imzası s=102 'dir.

İmza Doğrulama(Gerçekleme): B girdisi, $m'=s^2 \bmod n = 102^2 \bmod 589 =391$ hesaplanır. $m' \equiv 7 (\text{mod } 8)$ olduğundan B, $\tilde{m}$ =n-m' = 589-391 = 198 alır. Sonuç olarak,

$m = R^{-1}(\tilde{m}) = (198-6)/16=12$ hesaplanır ve imzayı kabul eder.

5.32. Not: Değiştirilmiş Rabin İmza Projesisinin Güvenliği

i) Algoritma. 5.30 kullanıldığında n'in çarpanları söz konusu olacağından jacobi sembolü –1'e sahip olan v değerini asla işaretlememelidir. Bunu anlamak için $y=v^{2d}=s^2$ jacobi sembolü 1 olması gerektiği incelenir. Fakat, 5.28(iii)'de $y^2 \equiv (v^2)^{2d} \equiv v^2 (\text{mod } n)$ 'dir. Bundan dolayı, $(v-y)(v+y) \equiv 0 \pmod n$'dir. v ve y zıt jacobi sembollerine sahip olduğundan;

$v \neq y \pmod n$ ve böylece $\gcd(v-y,n)=p$ veya q'dur.

ii) Sahte(taklit) varlık , orijinal Rabin projesi gibi değişen Rabin projesi için kolayca başarılır. Sadece, $1 \leq s \leq n-1$ aralığında bir s bulmaya ihtiyaç vardır. Şöyle ki, s^2 veya $n-s^2$ veya $2(n-s^2)$ değerlerinden birisi $6 (\text{mod } n16)$'ya denk olmalıdır.Bu durumların herhangi birinde, s, $m'=s^2 \bmod n$ için geçerli bir imzadır.

5.33:Not: Rabin İmza Projesisinin Performans Karakteristikleri

Algoritma. 5.25, genellikle jacobi sembolü hesabını içeren M'den $M_S=Q_n$'e bir redundancy fonksiyon gerektirir. İmza üretimi daha sonra en az bir jacobi sembolü(Not.5.27) ve modül n'in bir karekökünü içerecektir. Jacobi sembolü hesabı modüler çarpımların küçük bir sayısına eşit olduğunda Rabin imza üretimi, aynı modül boyutlarıyla bir RSA imza üretiminden daha etkili hesaplanamaz. İmza doğrulaması(gerçekleme), e=2 ise çok hızlıdır, sadece bir modüler çarpım gerektirir. Kök alma genel modüler çarpımdan daha etkili düzenlenebilir. Bu da RSA açık üssü e=3 olduğu zaman RSA imza doğrulamasıyla (gerçekleme) elverişli bir şekilde karşılaştırır. Değiştirilmiş Rabin imza projesi mesaj uzayını ve redundancy fonksiyonu açıkça belirtir. İmza üretimi jacobi sembolünün ve bir modüler üs alma işleminin değerlendirilmesini gerektirir.

5.34:Not: (Bant Genişliği Etkisi): Rabin dijital imza projesi bant genişliği etkisi açısından(5.3.3.vi) RSA projesine benzer.

5.3.5. ISO/IEC 9796 Formatı: ISO/IEC 9796 dijital imzalar için ilk uluslararası standart olarak **International Standarts Organization** tarafından 1991'de yayımlanmıştır. Mesajı geri almayı sağlayan dijital imza mekanizmasını kullanan dijital imza işlemini belirtir.

ISO/IEC 9796 ‘nın temel özellikleri:

i) Açık anahtar şifrelemesine dayanır.

ii) Özel imza Algoritması belirtilmemiştir fakat, k bitten k bite tasvir olmalıdır.

iii) Sınırlı uzunluktaki mesajları işaretlemekte kullanılır ve şifrelenebilen hash fonksiyon gerektirmez.

iv) Mesajı geri almayı sağlar (Not.5.14).

v) Gerektiği yerde doldurulan mesajı belirtir.

Standarda uygun mekanizma örnekleri RSA(Alg.5.19) ve değiştirilmiş Rabin(Alg.5.30)’dir. ISO/IEC 9796’da yuvarlama, redundancy ve doldurma için kullanılan özel metotlar imzayı taklit etmek için çeşitli araçlar sağlarlar. Tablo 5.3’de bu bölümün sembolleri gösterilmiştir.

Tablo.5.3. ISO/IEC 9796 notasyonu

Sembol	Anlamı
k	0^l = l bir uzunluğundaki dizinin bütün 0’ ları
d	$0^{l-e}\|\|b_{e-1}...b_1b_0$ burada $b_{e-1}...b_1b_0$ i’ nin ikili gösterimidir.
z	l bit uzunluğundaki dizinin bir kümesi
r	Bir anahtar uzayı K tarafından indekslenen şifre dönüşümlerinin kümesi
t	Bir şifreleme dönüşümüdür ve her bir E_t l bit diziden l bit diziye tasvirdir.

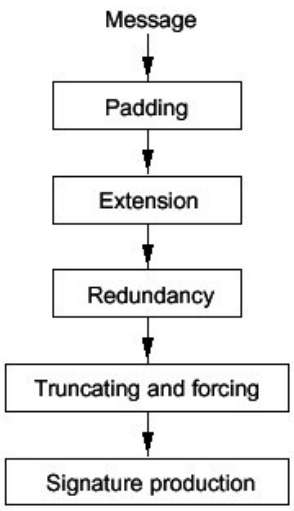
5.35. Örnek: ISO/IEC 9796 için örnek parametre tablosu

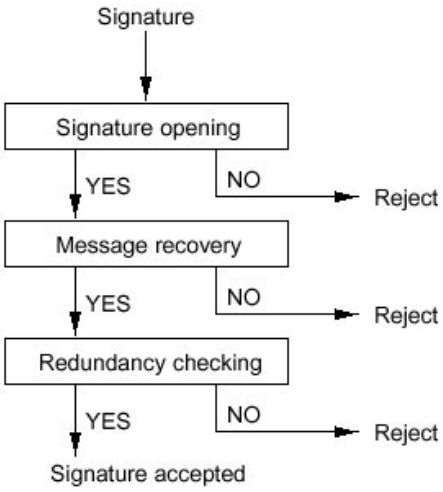
Aşağıdaki tablo, 150 bit mesaj ve 1024 bit imza için işaretlenmiş işlemde parametrelerin örnek değerleri listelemiştir.

Parametre	k-bit	d-bit	z-byte	r-bit	t-byte
Değer	1024	150	19	3	64

i) ISO/IEC 9796 için imza işlemi

İmza işlemi şekil.5.5.(a)' daki gibi 5 adımdan oluşur.





Şekil.5.5.a. ISO/IEC 9796 imza işlemi Şekil.5.5.b. ISO/IEC 9796 doğrulama işlemi

1- Doldurma: eğer,m mesaj ise form $MP=0^{1-1} \| m$ mesajı doldurur.burada, $1 \le r \le 8$ iken MP'deki bitlerin sayısı 8'in bir çarpanıdır. MP'deki byte'ların sayısı z'dir.Her m_i bir byte iken $MP=m_z \| m_{z-1} \| ... m_2 \| m_1$

2- Mesaj uzatma: Uzatılmış mesaj ME ile gösterilir , dizideki t byte kadar kendisiyle MP'nin solunda tekrarlanan kademeli bağlantı sayesinde MP'den sağlanır.

$ME=ME_t \| ME_{t-1} \| ... \| ME_2 \| ME_1$(her ME_i bir byte'tır). Eğer, t, z'nin bir çarpanı değilse kademeli bağlanmış olan en son byte'lar ki bu bytelar sağdan MP'nin ardarda gelen byteları iken MP'den byteların kısmi bir kümesidir. Tam olarak;

$0 \le i \le t\text{-}1$ için $ME_{i+1}=m_{(i \bmod z)+1}$'dir.

3- Redundancy mesaj: Redundanc, $MR=MR_{zt} \| MR_{zt-1} \| ... \| MR_2 \| MR_1$ byte dizisi oluşturularak ME'ye eklenir. Her ME_i bir byte'tır. MR, fazla olan t byteları ile ME'nin t bytelarını ayırarak ve daha sonra sonuç dizisinin byte MR_{2z} 'si uyarlanarak elde edilir.

$1 \le i \le t$ aralığında byte u'nun gölge fonksiyonu S(u) olarak adlandırılır.Burada,

$1 \le i \le t$ aralığında $MR_{2i-1}=MR_i$ ve $MR_{2i}=2(ME_i)$'dir. Eğer $u=u_2 \| u_1$ ise(u_1 ve u_2 4 bit uzunluklu dizilerdir) $S(u)= \pi(u_2) \| \pi(u_1)$ ve π permütasyon olup aşağıda tanımlanmıştır.

$$\pi = \begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & A & B & C & D & E & F \\ E & 3 & 5 & 8 & 9 & 4 & 2 & F & 0 & D & B & 6 & 7 & A & C & 1 \end{pmatrix}$$

Sonuç olarak MR, $r \oplus M_{2z}$ ile M_{2z} yer değiştirerek elde edilir.

4- Yuvarlama ve işleme: MR'den k-bit aradaki tamsayı Iraşağıdaki gibi biçimlendirilir.

a) MR'nin en az (k-1) anlamlı biti tek bit 1'in soluna eklenir.

b) Sonucun en az anlamlı $u_2 \| u_1$ byte, $u_1 \| 050$ ile yer değiştirerek düzenlenir.

(IR≡6 (mod 16) garanti edilecek şekilde yapılır)

5- İmza üretimi: Bir imza mekanizması k-bit tamsayıdan k-bit tamsayıya mesajı geri almaya izin veren bir tasvir kullanır. IR, bu mekanizma kullanılarak işaretlenir. s, sonuçtaki imzayı gösterir.

5.36.Not:(RSA,Rabin): ISO/IEC 9796, Rabin(5.25) ve RSA(5.19) dijital imza mekanizmasıyla ile kullanmak için tasarlandı. Bu özel projeler için imza üretimi çok açık olarak oluşturulur. **e**, RSA veya Rabin algoritmaları için açık üs, n modül ve d özel üs olsun. Öncelikle RR elemanı aşağıdaki gibi oluşturulur.

i) Eğer, e, tek ise veya çiftse ve IR'nin n modülüne göre jacobi sembolü 1 ise IR' dir.

ii) Eğer, e, çiftse ve IR'nin n modülüne göre jacobi sembolü -1 ise IR/2'dir. m'nin imzası $s=(RR)^d \bmod n$'dir. ISO/IEC 9796, imza s'nin $n-((RR)^d \bmod n)$ ve $(RR)^d \bmod n$'nin daha az olması gerektiğini belirtir.

ii) ISO/IEC 9796'nın Doğrulama(Gerçekleme) işlemi:

ISO/IEC 9796 dijital imzanın doğrulama(gerçekleme) işlemi şekil.5.5.b'deki üç bölüme ayrılabilir.

1- imza açılımı: s, imza olsun. Buna göre, aşağıdaki adımlar düzenlenir.

a) IR' tamsayısını geri almak için s'ye açık doğrulama(gerçekleme) dönüşümü uygulanır.

b) IR', en anlamlı biti bir olan ile k bitlik dizi değilse veya en az anlamlı olan 4'lü bit dizisi 0110 değerine sahip değilse imza reddedilir.

2- Mesajı geri alma: 2 byte'lı MR' dizisi aşağıdaki adımlar uygulanarak IR''den oluşturulur.

a) X, IR''nün en az anlamlı (k-1) biti olsun.

b) Eğer $u_4 \| u_3 \| u_2 \| 0110$,X'in en az anlamlı 4'lü dizisi ise $\pi^{-1}(u_4) \| u_2$ ile X'in en az anlamlı byte'ı yer değiştirir.

c) MR', X'in 0 ve 15 arasındaki 0 bitleri doldurularak sağlanır. Böylece sonuç dizisi 2t byte'tır.

❖ r ve z değerleri aşağıdaki gibi hesaplanır.

a) MR' ,2t byte'ından $1 \le i \le t$ aralığındaki t toplam $MR'_{2i} \oplus S(MR'_{2i-1})$ hesaplanır. Tüm toplamlar 0 ise imza reddedilir.

b) $MR'_{2i} \oplus S(MR'_{2i-1}\ MP')\neq 0$ için i'nin en küçük değeri **z** olsun.

c) Adım(b)'nin bulunan toplamın en az anlamlı 4'lü dizisi **r** olsun. **r**'nin hexadecimal değeri 1 ile 8 arasında değilse imza reddedilir.

❖ MR''den z -byte MP' dizisi aşağıdaki gibi oluşturulur.

a) $1 \le i \le z$ aralığı için $MP'_i = MR'_{2i-1}$

b) MP''nün en anlamlı (r-1) bitin hepsi 0 değilse imza reddedilir.

c) M', MP''nün 8z-r+1 sayıda en az anlamlı bitleri olsun.

3- Redundancy Kontrol: S imzası aşağıdaki gibi doğrulanır.

a) M' 'den , mesaj doldurma, mesaj uzatma ve işaretlenmiş işlemin redundancy mesaj adımları uygulanarak MR'' dizisi oluşturulur.

b) Sadece ve sadece MR''' ün (k-1) en az anlamlı biti MR''in (k-1) en az anlamlı bitine eşitse imza kabul edilir.

5.3.6. PKCS#1 Formatı

Açık anahtar şifreleme standartları **(PKCS-Publıc Key Cryptography Standarts)** RSA şifreleme ve imzaları için teknikleri içerir. PKCS#1'de dijital imza mekanizması, RSA imza projesinin mesajı geri alma özelliğini kullanmaz. Bir hash fonksiyonu gerektirir ve böylece ilaveli dijital imza projesi olur. Tablo 5.4'te bu bölümde kullanılan semboller listelenmiştir. Büyük harfler octet dizileri ifade eder.Eğer, X bir octet dizi ise X_i , soldan sayılan octet **i**'dir.

Tablo.5.4. PKCS#1 Notasyonu

Sembol	Anlamı	Sembol	Anlamı
k	(k≥11)octetinde n'in uzunluğu	EB	Şifreleme bloğu
n	$2^{8(k-1)} \le n < 2^{8k}$	ED	Şifreleme verisi
p,q	N'in asal çarpanları	Octet	8 uzunluğunda bitdizisi
e	Açık üs	ab	Hexadesimal octet değeri
d	Özel üs	BT	Blok tipi
M	Mesaj	PS	Doldurma dizisi
Md	Mesaj özeti	S	İmza
Md'	Mesaj özetlerinin karşılaştırılması	$\|S\|$	Octet içerisindeki x'in uzunluğu

i) PKCS#1 veri formatı

Veri, $\|D\| \le$ k-11 olan bir octet D dizisidir. BT, hexadecimal gösterimini 00 veya 01 olan tek octet'tir. PS, $\|PS\|$=k-3-$\|D\|$ olan bir octet dizisidir. Eğer,BT = 00 ise PS'deki tüm

octet'ler 00'dır. Eğer,BT=01 ise PS'deki tüm octet'ler ff 'dir. Formatlanan veri bloğu (şifreleme bloğu denir) EB=00 ‖BT‖PS‖00‖D'dir.

5.37. Not:(Veri Formatlama Mantığı)

i) Bir tamsayı olarak yorumlandığında octet EB dizisini sağlayan 00 ana bloğu modül n'den daha küçüktür.

ii) Eğer**,**blok tipi BT=00 ise sırasıyla EB'nin belirli çözümüne izin vermek için D, 0 olmayan bir octet veya kendi uzunluğu bilinen biri ile başlamalıdır.

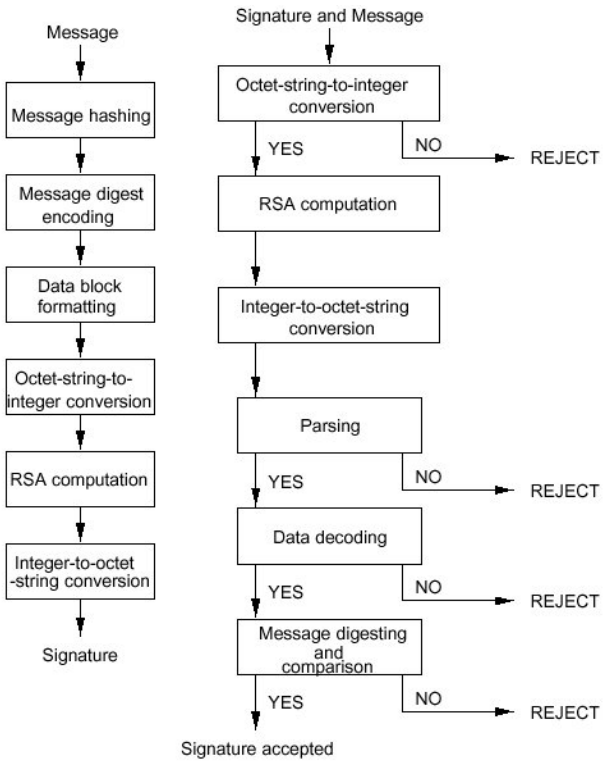
iii) Eğer**,** BT=01 ise belirli çözüm her zaman mümkündür.

iv) (iii)'de verilen sebep için ve imza mekanizması üzerinde kesin potansiyel saldırıları önlemek için BT=01 tavsiye edilir.

5.38. Örnek: (Belirli değerlerin PKCS#1 veri formatı) n'nin 1024-bitlik bir modül (k=128) olduğunu varsayalım.Buna göre, ‖D‖=20 octet ise ‖PS‖=105 octet ve ‖EB‖=128 octet olur.

ii) PKCS#1 için imza işlemi

İmza işlemi şekil 5.6(a)'daki adımları içerir.İmza işlemi için giriş m mesajıdır ve işaretçinin özel üssü d, modülü n'dir.



Şekil.5.6.a. PKCS#1 için İmza işlemi Şekil.5.6.b.. PKCS#1 için Doğrulama İşlemi

1- Hashlenen Mesaj: MD octet dizisini oluşturmak için seçilen mesaj-özet algoritması kullanılarak mesaj m hashlenir.

2- Mesaj-özet Şifrelemesi: MD ve hash algoritması ASN.1(Soyut syntax notasyonu) değeri içinde birleştirilir. Daha sonra, octet bir D veri dizisi vermek için BER(Basıc Encoding Rules) şifre çözer.

3- Veri Blok Formatı: D girişli bir veri dizisi octet EB dizisini biçimlendirmek için bölüm 5.3.6(i)'den veri formatı kullanır.

4- Octet Diziyi Tamsayıya Değiştirme: EB octetleri, $EB_1 \| EB_2 \| .... \| EB_k$ olsun. ikilik gösterimi octet EB_i olan $\widetilde{EB}_i$ tamsayısını tanımlayalım.

EB tamsayı gösterimi;

$m=\sum_{i=1}^{k} 2^{8(k-1)} \widetilde{EB}_i$ 'dir.

5- RSA hesabı: $s=m^d$ mod n'i hesaplanır.

6- Tamsayıyı octet diziye değiştirme: ED_i octetleri $s=\sum_{i=1}^{k} 2^{8(k-1)} \widetilde{ED}_i$'yi sağladığında s, $ED=ED_1 \| ED_2 \| .... \| ED_k$ octet dizisine dönüştürülür. İmza, S=ED'dir.

iii) PKCS#1'in Doğrulama(Gerçekleme) Işlemi

Doğrulama(Gerçekleme) işlemi şekil 5.6(b) adımlarını içerir. Doğrulama(Gerçekleme) işlemine girmek için mesaj **M**, imza **S**, açık üs **e**, modül **n**'dir.

1- Octet Diziyi Tamsayıya Çevirme:

a) S'nin bit uzunluğu 8'in bir çarpanı değilse S reddedilir.

b) İmza işleminin 4. adımındaki gibi s tamsayısını S'ye dönüştürür.

c) Eğer,s>n ise imza reddedilir.

2- RSA Hesaplaması: $m=S^e$ (modn) hesaplanır.

3- Tamsayıyı Octet Diziye Çevirme: İmza işleminin 6. adımındaki gibi m'yi k- octet uzunluklu EB octet dizisine çevirir.

4- Çözümleme: BT blok tipinin içine EB, doldurulmuş dizi PS ve D verisini çözümler.

a) EB anlamlı bir şekilde çözümlenmemiş ise imza reddedilir.

b) BT 00 veya 01 den biri değilse imza reddedilir.

c) PS 8 octetten küçük oluşuyor veya BT ile uyuşmuyorsa imza reddedilir.

5- Veri Çözme:

a) Özet mesaj MD ve hash algoritma kimliğini elde etmek için BER ,D'yi çözer.

b) Hashlenen algoritma kimliği MD2'nin veya MD5'i tanımazsa imza reddedilir.

6- Mesaj Özeti ve Karşılaştırma:

a) MD'yi elde etmek için seçilen özet mesaj algoritma ile mesaj M hashlenir.

b) Eğer,sadece ve sadece MD^1=MD ise M üzerindeki S imzası kabul edilir.

5.4.Fiat-Shamir İmza Projesi

5.4.1. Feige-Fiat-Shamir İmza Projesi

Feige-Fiat-Shamir imza projesi fiat ve shamir'in önceki imza projelerinin değişik şeklidir ve birkaç sabit pozitif tamsayı k için tek-yol hash fonksiyonu h: $\{0,1\}^* \rightarrow \{0,1\}^k$ gerektirir. Burada, $\{0,1\}^k$, k-bit uzunluklu bit dizilerini gösterir ve $\{0,1\}^*$ rastgele bit uzunluklu tüm bit dizilerini gösterir. Metot, ilaveli dijital imzayı sağlar ve rastgele bir mekanizmadır.

5.39 Algoritma: Feige-Fiat-Shamir İmza Projesi için Anahtar Üretimi

Özet: Her girdi(entıty) açık anahtarı ve karşılığı özel anahtarı oluşturur. Her A girdisi (entıty) aşağıdaki yolu izlemelidir.

1- Rastgele faklı p,q gizli asallarını oluşturur ve n=pq'yu düzenler.

2- k pozitif tamsayısını ve $s_1,s_2,\ldots s_k \in Z_n^*$ olan farklı tamsayıları seçer.

3- $1 \le j \le k$ için $V_j=S_j^{-2} \pmod n$ hesaplanır.

4- A'nın açık anahtarı sıralı k'lı $(v_1,v_2,\ldots v_k)$ ve modül n'dir. Özel anahtarı sıralı k'lı $(s_1,s_2,\ldots s_k)$'dır.

5.40. Algoritma: Feige-Fiat-Shamir İmza Üretimi ve Doğrulaması(Gerçekleme)

Özet: A girdisi (entıty) keyfi uzunluklu ikilik sistemdeki M mesajını işaretler. Herhangi bir B girdisi(entıty), A'nın açık anahtarını kullanarak bu imzayı doğrular.

1- İmza üretimi: Girdi (entıty) A, aşağıdaki yolu izlemelidir.

a) $1 \le r \le n-1$ aarlığında rastgele bir r tamsayısı seçer.

b) $u=r^2 \pmod n$ hesaplanır.

c) Her $e_i \in \{0,1\}$ için $e=(e_1,e_2,\ldots e_k)=h(m \| u)$ hesaplanır.

d) $S=r.\prod_{j=1}^{k} s_j^{e_j} \bmod n$ hesaplanır.

e) A'nın imzası m için (e,s)'dir.

2- Doğrulama(Gerçekleme): m üzerinde A'nın imzasını Doğrulamak(Gerçeklemek) için B aşağıdaki yolu izlemelidir.

a) A'nın gerçek açık anahtarı $(v_1,v_2,\ldots v_k)$ ve n'i elde etmelidir..

b) $w=s^2.\prod_{j=1}^{k} v_j^{e^j} \bmod n$ hesaplanır.

c) $e^{\prime} = h(m \| w)$ hesaplanır.

d) Eğer,sadece ve sadece $e= e^{\prime}$ ise imzayı kabul eder.

5.41. Örnek: Küçük Parametreler ile Feige-Fiat-Shamir İmza Üretimi

Anahtar üretimi: Girdi (entıty) A, p=3571, q=4523 asallarını oluşturur ve n=pq=16151633. Aşağıdaki tablo (A'nın özel anahtarı) s_j'nin seçimlerini ve yalnız aradaki değerleri s_j^{-1} ile (A'nın açık anahtarı) v_j tamsayılarını gösterir.

j	1	2	3	4	5
s_j	42	73	85	101	150
S_j^{-1} mod n	4999315	885021	6270634	13113207	11090788
$V_j=s_j^{-2}$mod n	503594	4879739	7104483	1409171	6965302

İmza üretimi: h: $\{0,1\}^k \rightarrow \{0,1\}^5$ bir hash fonksiyon olsun. A rastgele r=23181 tamsayısını seçer ve $u=r^2$ mod n=4354872 hesaplanır. Mesaj m'yi işaretlemek için $e=h(m \| u)$=1050 değerlendirilir. A, $s=rs_1s_3s_4$ mod n (23181)(42)(85)(101) mod n=7978909 düzenler m için imza (e=1050, s=7978909)'dir.

İmza Doğrulaması(Gerçekleme): Girdi B, s^2 mod n=2926875 ve $v_1v_3v_4$ mod n=(533594)(7104483) (1409171) mod n=15668174 hesaplanır. Daha sonra,

B $w=s^2$ ve $v_1v_3v_4$ mod n=4354872 hesaplanır. w=u olduğundan $e^{\prime} = h(m \| w)=h(m \| u)=e$ olur ve böylece B imzayı kabul eder.

5.42. Not:(FFS imza projesinin güvenliği)

i) RSA imza projesine benzemez. Tüm girdiler (entıty) aynı modül n'i kullanabilir. Bu senaryoda, güvenilir 3 grup (TTP-Trusted Third Party) p ve q asalları ve aynı zamanda her girdi (entıty) için açık ve özel anahtarların üretimine ihtiyaç duyar.

ii) FFS projesinin güvenliği modül n'in kareköklerinin hesabının etkinliğine dayanır. Uyarlanarak seçilmiş mesaj saldırısına karşı güvenliği sağlanmıştır, çarpanların etkileşimi sağlanmıştır. h rastgele bir fonksiyondur ve s_i'ler farklıdır.

5.43. Not:(Parametre seçimi ve anahtar depolama gereksinimleri)Eğer**,** n, t-bitlik bir tamsayı ise algoritma 5.39'da oluşturulan özel anahtar kt bit büyüklüğündedir. Bu, $t^{\prime}< t$ bit uzunluğundaki sayılar gibi $1 \le j \le k$ aralığında rastgele s_j değerlerini seçerek azaltılabilir. $t^{\prime}$, yine de s_j' nin tahmini mümkün iken çok küçük seçilmemelidir. Açık anahtar (k+1)t- bit

büyüklüğündedir. Örneğin; t=768 ve k=128 ise özel anahtar 98304 bit, açık anahtar 99072 bit gerektirir.

5.44. Not:(Özdeşlik-tabanlı Feige-Fiat-Shamir İmzaları)

Varsayalımki,TTP, p ve q asallarını ve modül n'i oluştursun.. Modüller sistemdeki tüm girdiler(entıty) için geneldir. Algoritma 5.39 özdeşliğe dayalı bir proje olarak değiştirilebilir. A girdisinin(entıty) bitdizisi I_A,A'nın kimlik bilgisini içerir. j, ikilik sistemde gösterilmek şartıyla, TTP, $1 \le j \le k$ aralığında $v_j=f(I_A \| j)$ hesaplanır ve $1 \le j \le k$ iken v_j^{-1} modül n'in karekökü s_j'yi hesaplanır.Burada,h: $\{0,1\}^k \rightarrow Q_n$ 'e tek-yol Hash fonksiyondur., A'nın özel anahtarı sıralı-k'lı $(s_1,s_2,... s_k)$ iken A'nın açık anahtarı I_A'dır. h,f fonksiyonları ve modül n geniş-sistemli niceliklerdir.

Bu işlem algoritma 5.39'dakinden daha küçük bir nicelik olan I_A'dan oluşturulan açık anahtar avantajına sahiptir.Potansiyel olarak depolama ve gönderme maliyetini azaltabilir. Dezavantajı ise TTP için girdilerin (entıty)özel anahtarlarının bilinmesi ve modüln'in sistem-genişliğinin daha çekici bir hedef haline gelmesidir.

5.45. Not:(Feige-Fiat-Shamir İmzalarının küçük asal değişimi)

Bu gelişme, açık anahtarın boyutunu azaltmayı ve imza doğrulamanın(gerçekleme) etkinliğini arttırmayı amaçlar. Not 5.44'de tarif edilen değişime benzemez. Her A girdisi (entıty) kendi modül n_A'sını ve $v_1,v_2,... v_k \in Q_n$ olan k küçük asallarının kümesini oluşturur. Girdi(entıty) A, $1 \le j \le k$ aralığındaki her j için v_j^{-1} modül n'in karekökleri olan s_j'den birini seçer. Bunlar özel anahtarı oluşturur. Açık anahtar, n_A ve $v_1,v_2,...,v_k$ değerlerinden oluşur. Daha küçük sayılarla hesaplamalar yapıldığında imzaların doğrulaması(gerçekleme) daha etkili bir şekilde ilerler.

5.46. Not:(Feige-Fiat-Shamir İmzalarının performans karakteristikleri)

RSA projesi t=768 uzunluklu modüller ile saf tekniklerde kullanılan imza üretimi ortalama 1152 modüler çarpım gerektirir. FFS projesi için imza üretimi ortalama k/2 modüler çarpım gerektirir. Bu proje ile bir mesaj işaretlemek için t=768 ve k=128 uzunluklu modül ortalama 64 modüler çarpım veya RSA'nın saf tamamlanmasında gereken çalışmanın %6'sından daha azını gerektirir. Açık üs e=3 ve FFS için ortalama 64 modüler çarpım gerekli ise imza doğrulaması(gerçekleme) RSA için sadece bir modüler çarpıma ihtiyaç duyar. İmza üretiminin çabuk uygulanması gerektiği ve anahtar uzayı depolamanın sınırlı olmadığı durumda uygulamalar için FFS projesi RSA'ya tercih edilebilir.

5.4.2. GQ İmza Projesi

GQ kimlik protokolü bir dijital imza mekanizmasına dönüştürülebilir. n pozitif tamsayı iken $h:\{0,1\}^* \rightarrow Z_n$ bir hash fonksiyonu olsun.

5.47. Algoritma: GQ İmza Projesi İçin Anahtar Üretimi

Özet: Her girdi(entıty), açık anahtar (n,e,J_A) ve karşılığı a özel anahtarını oluşturur. A girdisi (entıty) aşağıdaki yolu izlemelidir.

1- Rastgele farklı p ve q gizli asalları seçer ve n=pq kurar.

2- $Gcd((e(p-1)(q-1))=1$ şeklinde $e \in \{1,2,...,n-1\}$ olan bir tamsayı seçer.

3- A'nın bir kimliği gibi hizmet eden $1<J_A<n$ aralığında J_A tamsayısını seçer. Şöyle ki $gcd(J_A,n)=n^+$ 'dir.

4- $a \in Z_n$ olan bir tamsayı tespit eder. Şöyleki $J_A a^e \equiv 1 (mod\ n)$ aşağıdaki gibi takip eder.

4.1- J_A^{-1} mod n hesaplanır.

4.2- $d_1=e^{-1} mod\ (p-1)$ ve $d_2=e^{-1}\ mod(q-1)$ hesaplanır.

4.3- $a_1 \equiv (J_A^{-1})^{d_1}$ mod p ve $a_2 \equiv (J_A^{-1})^{d2}$ mod q hesaplanır.

4.4- $a \equiv a_1 (mod\ p)$, $a \equiv (mod\ q)$ simultane uygunluğa bir a çözümü bulur.

5- A'nın açık anahtarı (n,e,J_A) , özel anahtarı a'dır.

5.48. Algoritma: GQ İmza Üretimi ve Doğrulama(Gerçekleme)

Özet: Girdi (entity) A, keyfi uzunluklu ikilik sistemdeki m mesajını işaretler. Herhangi bir B girdisi(entity) A'nın açık anahtarını kullanarak bu imzayı doğrulayabilir.

1-İmza üretimi: A girdisi (entity) aşağıdaki yolu izler.

a) Rastgele tamsayı k'yı seçer ve $r=k^e$ mod n hesaplanır.

b) $l=h(m \| r)$ hesaplanır.

c) $s = ka^l$ mod n hesaplanır.

d) A'nın imzası m için (s,l) çiftidir.

2- Doğrulama(Gerçekleme): m üzerinde A'nın (s,l) imzasını doğrulamak(gerçeklemek) için B aşağıdaki yolu izlemelidir.

a) A'nın gerçek açık anahtarı (n,e,J_A)'yı elde eder.

b) $u= s^e J_A^l$ mod n ve $l'=h(m \| u)$ hesaplanır.

c) Sadece $l=l'$ ise imzayı kabul eder.

5.49. Örnek:Küçük parametreler ile GQ İmza Üretimi

Anahtar üretimi: A girdisi (entıty) p=2089,q=27457 asallarını seçer ve n=pq=572450993 hesaplanır. A, e=47 tamsayısını, J_A=1091522 kimliğini seçer ve a=21465724 olarak $J_A a^e \equiv 1 (\text{mod } n)$ denkliğini çözer. A'nın özel anahtarı a=21465724 iken (n=572450993 ,e=47,J_A=1091522)'dir.

İmza üretimi: m= 1101110001'i işaretlemek için A rastgele k=42134 tamsayısını seçer ve $r=k^e \text{ mod } n=297543350$ hesaplanır. Daha sonra , A $L= h(m \| r)=2713833$ ve $s=ka^1 \text{ mod } n=(42134)21465724^{2713833} \text{ mod } n=252000854$ hesaplanır. m için A'nın imzası (s=252000854, L=2713833) çiftidir.

İmza Doğrulama(Gerçekleme): B, $s^e \text{ mod } n=252000854^{47} \text{ mod } n=398641962$, $J_A^1 \text{ mod } n=1091522^{2713833} \text{ mod } n=50523867$ ve sonuç olarak $u=s^e J_A^1 \text{ mod } n=297543350$ hesaplanır. u=r olduğundan , $L^1= h(m \| u)= h(m \| r)=L$ ve böylece, B imzayı kabul eder.

5.50. Not: GQ İmzaProjesinin Güvenliği

Algoritma 5.47'de e taklit ve saldırılara karşı yeterince büyük olmalıdır. Potansiyel saldırılar yalnız takip eden çizgilerle ilerler. Rakip m mesajını seçer ve $L \equiv t (\text{mod } e)$ olana kadar t'nin pekçok değeri için $l= h(m \| S_A^l)$ hesaplanır. Bunun $O(\sqrt{e})$ deneme ile oluşması beklenir. Belirlenen (l,t) çiftine sahipken rakip t=xe+l şeklinde x tamsayısını belirler ve $s=J_A^x \text{ mod } n$ hesaplanır. $s^e J_A^l \equiv (J_A^x)^e J_A^l \equiv J_A^{xe+l} \equiv J_A^t (\text{mod } n)$ incelenir ve bununla beraber $h(m \| J_A^t)=l$'yi inceler. Böylece (s,l) m mesajı için geçerli bir sahte imzadır.

5.51.Not:Parametre Seçimi

Tamsayı çarpanları için geçerli metotlar en az 768 bit boyutlu bir modül n'in seçimini önerir. Not 5.50,e'nin en az 128 bit boyutunda olması gerektiğini tavsiye eder. Güvenli hash fonksiyonlarının çıkışları için genel değerler 128 veya 160 bit'tir. 768 bit modülleri ve 128 bit e ile GQ projesinin açık anahtarı J_A olarak temsil eden bit sayıları u iken 896 + u bit boyutundadır. Özel anahtar a 768 bit boyutundadır.

5.52.Not: GQ İmzalarının Performans Karakteristikleri

GQ'nun imza üretimi(alg.5.48) iki modüler üs alma ve 1 modüler çarpım gerektirir. 768 bit modül n'i 128 bit e değerini ve 128 bit l çıkışlı hash fonksiyonunu kullanılarak imza üretimi ortalama 384 modüler çarpıma ihtiyacı vardır. İmza doğrulaması(gerçekleme) benzer çalışma miktarını gerektirir. İmza üretimi için RSA ile FFS karşılaştırılırsa :GQ, FFS'den hesaplama olarak daha yoğundur fakat, daha küçük anahtar depolama uzayı gerektirir.(Not 5.51)

5.53.Not: : GQ İmzalarının Mesajı Geri Almadaki Değişimi

Algoritma 5.48 mesajı geri almak için aşağıdaki gibi düzenlenebilir. İşaretleme uzayı $M_S=Z_n$ ve $m \in M_S$ olsun. imza üretiminde gcd(k,n)=1 olmak şartıyla rastgele bir k seçilir ve $r=k^e \bmod n$ ve $l=mr \bmod n$ hesaplanır. İmza $s=ka^l \bmod n$'dir. Doğrulama (gerçekleme) işlemi $s^e J_A{}^l \equiv k^e a^{el} J_A{}^l \equiv k^e \equiv r \pmod n$ 'yi verir. Mesaj m $l_r{}^{-1} \bmod n$'den geri alınır. Mesajı geri almalı tüm dijital imza projelerinde olduğu gibi uygun redundancy fonksiyon R, sahte(taklit) varlık saldırılarına önlem almak için gereklidir.

5.5. DSA ve İlgili İmza Projeleri

Bu bölüm dijital imza Algoritması(DSA-Digital Signature Algorithm) ve ilgili imza projelerini inceler. Bunların çoğu birkaç büyük p asalı için Z_p^* üzerinde sunulur. Fakat bu mekanizmaların hepsi sınırlı devirsel grup ile genelleştirilebilir. Bu, 5.5.2'deki ElGamal imza projesi ile tarif edilir.

5.5.1. Dijital İmza Algoritması (DSA - Digital Signature Algorithm)

1991'in Ağustosunda NIST (Natioal Institute of Standarts and Technology) DSA(dijital imza algoritması) 'yı hedeflemiştir. DSS (Digital Signature Standart- dijital imza standardı) olarak adlandırılan FIPS (Federal Information Processing Standart) Amerika'nın ilk standartı olmuştur ve ilk olarak herhangi bir hükümet tarafından tanınan dijital imza projesidir. Algoritma ElGamal projesinin değişik bir şeklidir ve ilaveli dijital imza projesidir.

İmza mekanizması birkaç q tamsayısı için bir hash fonksiyon $h:\{0,1\}^* \rightarrow Z_q$ gerektirir.

5.54. Algoritma: DSA İçin Anahtar Üretimi

Özet: Her girdi(entity) bir açık ve karşılığı olan özel anahtarı oluşturur. Her A girdisi (entity) aşağıdaki yolu izlemelidir.

1- $2^{159} < q < 2^{160}$ aralığında bir q asal sayısını seçer.

2- $0 \le t \le 8$ aralığında t'yi ve $2^{511+64t} < p < 2^{512+64t}$ aralığında, q'nun (p-1)'i böldüğü p asal sayısını seçer.

3- Z_p^*, deki q. mertebeden tek devirli grubun α üretecini seçer.

3.1 $g \in Z_p^*$ bir eleman seçer ve $\alpha=g^{(p-1)/q} \bmod p$ hesaplanır.

3.2α=1 ise 3.1 adımına geri döner.

4-$1 \le \alpha \le q-1$ aralığında rastgele bir αtamsayısını seçer.

5-$y=\alpha^a \bmod p$'yi hesaplanır.

6-A'nın açık anahtarı (p,q, α,y), özel anahtarı a'dır.

5.55. Not:p ve q DSA Asallarının Üretimi

Algoritma 5.54 öncelikle q asalını seçmelidir, daha sonra q, (p-1)'e bölündüğünde p asalını bulmaya çalışır.

5.56. Algoritma: DSA imza üretimi ve Doğrulama(Gerçekleme)

Özet: A girdisi (entity) keyfi uzunluktaki ikilik gösterime sahip m mesajını işaretler. Herhangi bir B, A'nın açık anahtarını kullanarak bu imzayı doğrulayabilir.

1-İmza üretimi: A girdisi aşağıdaki yolu izlemelidir.

a) $0 < k < q$ aralığında rastgele gizli bir k tamsayısı seçer.
b) $r= (\alpha^k \bmod p) \bmod q$ hesaplanır
c) $k^{-1} \bmod q$ hesaplanır
d) $s=k^{-1} \{h(m)+ar\} \bmod q$ hesaplanır
e) m için A'nın imzası (r,s) çiftidir.

2- Doğrulama(Gerçekleme): m üzerinde A'nın imzası (r,s)'yi doğrulamak (gerçekleme) için, B aşağıdaki yolu izlemelidir.

a) A'nın gerçek anahtarı (p,q, α,y) elde etmek
b) $0 < r < q$ ve $0 < s < q$ olup olmadığı doğrulanır, değilse imzayı reddeder.
c) $w=s^{-1} \bmod q$ ve h(m) hesaplanır
d) $u_1=w.h(m) \bmod q$ ve $u_2 =rw \bmod q$ hesaplanır
e) $v=(\alpha^{u_1} y^{u_2} \bmod p) \bmod q$ hesaplanır.
f) Sadece v=r ise imzayı kabul eder.

5.57 Örnek:Küçük Parametreler İle DSA İmza Üretimi

Anahtar Üretimi:

A girdisi, p=124540019 ve q=17389 asallarını q, (p-1)'i bölecek şekilde seçer.(p-1)/q=7162 dir. Z_p^* da rasgele bir g elemanı g=110217528 olarak seçilir ve $\alpha=g^{7162} \bmod p=10083255$ hesaplanır. $\alpha \neq 1$ olduğundan, α, Z_p^* de q. mertebeden devirli alt grup için bir üreteçtir.

Bir sonraki rasgele α tamsayısının seçimi $1 \leq \alpha \leq q-1$ aralığında α=12496 olarak seçelim ve ve $y= \alpha^a \bmod p=10083255^{12496} \bmod 124540019=119946265$ hesaplanır.A'nın özel anahtarı α=12496 iken A'nın açık anahtarı(p=124540019,q=17389, α=10083255,y=119946265) olur.

İmza Üretimi:m'yi işaretlemek için A girdisi k=9557 rasgele tamsayısını seçer ve r=(α^k mod p) mod q=(10083255^{9557} mod 12450019) mod 17389=27039929 mod 17389=34 hesaplar. Daha sonra bu örnek için k^{-1} mod q=7631, h(m)=5246 hesaplanır.Sonuç olarak, s=(7631){5246+(12496).(34)}mod q=13049 olur.Böylece m için imza (r=34,s=13049) çiftidir.

İmza Doğrulama (gerçekleme): B girdisi $w=s^{-1}$ mod q=1799,u_1=w.h(m) mod q=(5246)(1799) mod 17389=12716 ve u_2=rw mod q=(34)(1799) mod 17389=8999 hesaplanır.Daha sonra ,B,v=($\alpha^{u_1}y^{u_2}$ mod p) mod q=(10083255^{12716}.119946265^{8999} mod 12450019) mod 17389=27039929 mod 17389=34 hesaplanır.v=r olduğundan dolayı B imzayı kabul eder.

5.58. Not: (DSA güvenliği) DSA güvenliği farklı fakat ilgili ayrık logaritma problemlerine dayanır. Biri, güçlü ilaveli-hesap metotlarının uygulandığı Z_p^* içindeki logaritma problemidir, diğeri ise en iyi geçer metotlarının karekök süresince çalıştığı q sırasının devirli alt gruplarındaki logaritma problemidir. DSA s denklemiyle ilişkili olarak ElGamal imzalarının özel durumu iken sonraki için güvenlik faktörleri burada uygundur.

5.59. Not:Önerilen Parametre Ölçüleri

p'nin boyutu 512 ve 1024 bit arasında 64 çarpım olabilirken q'nun boyutu 160 bitte algoritma 5.54 yardımıyla sabite alınır. 512 bit p asalı birlik saldırısına karşı marjinal güvenliği sağlar. 1996'da olduğu gibi en az 768 bit modüller tavsiye edilir. FIPS 186 1024 bitten daha büyük p asallarına izin vermez.

5.60. Not:DSA'nın Performans Karakteristikleri

Varsayalım ki p, 768 bitlik bir tamsayı olsun. imza üretimi, ortalama 240 modüle çarpım olarak 160 bitlik modül ile bir modüler ters alma, iki 160-bit modüler çarpım ve bir toplama gerektirir. 160 bit işlemlerde üs alma ile karşılaştırıldığında nispeten daha küçüktür. DSA'nın imza üretiminin zamanında yapılmasına gerek duymayan ve üs almanın önceden hesaplanabilmesi gibi bir avantajı vardır. RSA imza projesinde önceden hesaplama DSA da olduğu gibi mümkün değildir. İmza gerçeklemesi (doğrulaması) için çalışmanın daha büyük kısmı her biri 160 bitlik üs olan modül p'ye göre iki üs almaişleminden ibarettir. Ortalama, bunların her biri 240 modüler çarpım veya toplamda 480 çarpım gerektirir. Bazı güvenlik durumlarında iki üs alma işlemi aynı zamanlı olarak incelenebilir. Bu durumda, ortalama 280 modüler çarpım yapılır.

5.61. Not: (Sistem – Genişliği Parametreleri)

Her bir girdi(entity) için kendi p, q asallarını seçmek gerekli değildir. DSS, geniş sistem paremetleri olması için p,q ve α 'yı kabul eder. Bu yine bir rakip için daha çekici hedeftir.

5.62. Not: Başaramama Olasılığı

Doğrulama(gerçekleme), s^{-1} mod q'nun hesaplamasını gerektirir. Eğer, s=0 ise s^{-1} oluşmaz. Bu durumdan sakınmak için işaretçi $s \neq 0$'ı konrol edebilir fakat s, Z_p içinde rastgele bir eleman olarak varsayılırsa s=0 olma olasılığı $\left(\frac{1}{2}\right)^{160}$ dır. Pratikte, Buna benzer bir durum oluşmaz. İşaretçi ayrıca r**≠0** 'ı kontrol edebilir. Eğer işaretçi r=0 veya s=0'dan birini bulursa k'nın yeni değeri üretilmelidir.

5.5.2. ElGamal İmza Projesi

ElGamal imza projesi rastgele imza proje mekanizmasıdır. Rastgele uzunluklu ikilik mesajlarda ilaveli dijital imzalar oluşturur. Ve p büyük asal sayı olmak üzere $h:\{0,1\} \rightarrow Z_p$ şeklinde bir hash fonksiyona ihtiyaç duyar. DSA, ElGamal imza mekanizmasından farklıdır.

5.63. Algoritma: ElGamal İmza Projesi İçin Anahtar Üretimi

Özet: Herbir girdi(entity) açık anahtar ve ona uyan özel anahtaroluşturur.A girdisi aşağıdakileri yapmalıdır:

1.Rasgele büyük bir p asal sayısı ve Z_p^* çarpma grubunun α üreteci üretilir

2.$1 \leq a \leq p-2$ aralığında rasgele bir tamsayı seçilir

3.$y=\alpha^a \bmod p$ hesaplanır

4.A'nın açık anahtarı (p, α,y) ,özel anahtarı a'dır.

5.64. Algoritma: ElGamal İmza Üretimi ve Doğrulaması (gerçeklemesi)

Özet: A girdisi (entity) keyfi uzunluktaki ikilik gösterime sahip m mesajını işaretler. Herhangi bir B, A'nın açık anahtarını kullanarak bu imzayı doğrulayabilir

1-İmza üretimi: A girdisi aşağıdaki yolu izlemelidir.

a) $1 \leq k \leq p-2$ aralığında gcd(k,p-1)=1 olan rastgele gizli bir k tamsayısı seçer.

b) $r= (\alpha^k \bmod p)$ hesaplanır

c) $k^{-1} \bmod (p-1)$ hesaplanır

d) $s=k^{-1} \{h(m)-ar\} \bmod (p-1)$ hesaplanır

e) m için A'nın imzası (r,s) çiftidir.

2- Doğrulama(Gerçekleme): m üzerinde A'nın imzası (r,s)'yi doğrulamak (gerçekleme) için, B aşağıdaki yolu izlemelidir.

a) A'nın gerçek anahtarı (p, α,y) elde etmek

b) $1 \le r \le p-1$ olup olmadığı doğrulanır, değilse imzayı reddeder.

c) $v_1=y^r r^s \bmod p$ hesaplanır

d) h(m) ve $v_2 = \alpha^{h(m)} \bmod p$ hesaplanır

e) $v=(\alpha^{u_1} y^{u_2} \bmod p) \bmod q$ hesaplanır.

f) Sadece $v_1=v_2$ ise imzayı kabul eder.

5.65.Örnek:Küçük Parametrelerle ElGamal İmza Üretimi

Anahtar Üretimi:

A girdisi, p=**2357** ve Z^*_{2357} 'nın α=2 üretecini seçer.A ,a=1751 özel anahtarını seçer ve y= $\alpha^{\alpha} \bmod p=2^{1751} \bmod 2357=1185$ hesaplanır.A'nın açı k anahtarı (p=2357, α =2,y=1185)'dir.

İmza Üretimi:

İmza Doğrulama (gerçekleme):

B, $v_1=1185^{1490}.1490^{1777} \bmod 2357=1072$, h(m)=1463, ve $v_2=2^{1463} \bmod 2357=1072$ 'yi hesaplar.$v_1=v_2$ olduğunda B imzayı kabul eder.

5.66.Not: El Gamal İmzalarının Güvenliği:

i) Rakip, $r=\alpha^k \bmod p$ hesaplayarak ve rastgele bir k tamsayısını geçerek m üzerinden A'nın imzasını işlemeye teşebbüs edebilir. Daha sonra rakip $s=k^{-1}\{4(m)-ar\} \bmod (p-1)$ tespit etmelidir. Discrete logaritma probleminin hesabı mümkün değilse rakip rastgele bir s seçiminden daha iyisini yapamaz. Mümkün olan başarı sadece büyük p için ihmal edilebilir olan 1/p'dir.

ii) Farklı k işaretlenmiş her mesaj için seçilmelidir. Diğer yandan özel anahtar yüksek ihtimalle aşağıdaki gibi tespit edilebilir.

($s_1 = k^{-1}$...aynısı)düşünelim. Daha sonra (...aynısı),daha sonra(k=...aynısı) Bir kez k bilinen, a kolayca bulunandır.

iii) Hash fonksiyon h hiç kullanılmıyorsa işaretlenmiş denklem (s=...aynısı) 'dır. Daha sonra sahte(taklit) varlık saldırısını aşağıdaki gibi kurmak rakip için kolaydır.

gcd(v,p-1)=1 ile (v;v) çifti seçilir. R=(aynısı) hesaplanır. (r,s) çifti $((\alpha^m \alpha^{-ar})^{s^{-1}}$ aynısı) iken m=sumod (p-1) mesajı için geçerli bir imzadır.

iv) Algoritma. 5.64'te 2b adımı o<r<p aralığını kontrol etmek için bir sorgulayana ihtiyaç duyar. Bu kontrol yapılmazsa , rakip aşağıdaki gibi A girdisinin(entıty) oluşturduğu geçerli imzayı sağlayan onun seçiminin mesajını işaretleyebilir. A tarafından sağlanan m mesajının (r,s) imzası olduğunu düşünelim. Rakip kendi seçimini m mesajını seçer ve (h(m')... aynısı) hesaplanır. daha sonra (r'≡...aynısı) ve (r'≡...aynısı) olmak koşulu ile (s'≡...aynısı) ve r' hesaplanır. daha sonraki her zaman çin kolon teoremi ile mümkündür. (r',s') çifti, sorgulmama algoritması tarafından kabul edilebilecek m' mesajının bir imzasıdır.

5.67.Not: Parametre Seçimlerine Dayanan Güvenlik:

i) (İndex-calculus saldırısı) İndex-calculus saldırılarının etkin kullanımını önlemek için p asalları yeterince büyük olmalıdır.

ii) (Pohlig-Hellman saldırısı) Pohlig-Hellman discrete logaritma saldırısını önlemek için q asal sayısı ile bölünebilen p-1 yeterince büyük olmalıdır.

iii) (Zayıf üretecileri) p≡1 mod 4 ve α üretecisinin aşağıdaki durumları belirttiğini düşünelim:

a) α (p-1)'i böler ve

b) zp içinde α sırasının salt grubundaki logaritmaların hesabı etkili bir şekilde yapılabilir.

Daha sonra Doğrulama(Gerçekleme) algoritmaları ile kabul edilecek imzalara oluşturmak (A'nın özel anahtarı bilinmeden) rakip için mümkündür. Bunu şöyle görebiliriz: p-1 = α q olduğunu düşünelim. Mesaj m'i işaretlemek için rakip aşağıdaki maddeleri uygular.

a) t= (p-3)/2 hesaplanır ve r=q kurar.

b) A'nın açık anahtarı y iken $x^{q^z} \equiv y^q \pmod p$ belirler.

) (s=...aynısı) hesaplanır.

d) algortima 5.64'ün ikinci adımı tarafından kabul edilecek m üzerindeki imza (r,s)'dir. bu saldırı $r^s, y^r \equiv \alpha^{h(m)} \pmod p$ ile belirlenen Doğrulama(Gerçekleme) denklemi sayesinde çalışır. Öncelikle (α a≡...aynısı) ve ($q^{(p-1)/2}$...aynısı)incelenir. Bunlardan (q^t=...) sonucu çıkarılır. Şimdi ($r^s y^n$=...aynısı)'dir.

α ,zp^x 'nin kendi üretecisine rağmen asal sıranın zp^x alt grubunun üreteci gibi seçilirse bu saldırı sağlanabilir.

5.68.Not:(El Gamal İmzasının Karakterlerinin Performansı)

i) Algoritma. 5.64 ile sağlanan imza üretimi hızlıdır ve bir modüler üs almayı, geliştirilmiş Euclidean Algoritma.(k^{-1} mod(p-1) hesabı için) ve modüler çarpımları gerektirir. (modüler çıkarma, modüler çarpımlarla karşılaştırıldığında uygundur.)Geliştirilmiş Euclidean

algoritma uygulamaları ve üs alması sadece iki modüler çarpımı gerekli olduğu imza üretimi durumunda yapılabilir.

ii) İmza gerçeklemesi (doğrulaması) daha önemlidir ve 3 üs almayı gerektirir. Her üs alma 9/2 [lgp]çarpımlarının toplam değeri için oratalam 3/2[lgp] modüler çarpımlarına ihtiyaç duyar. Hesaplama değerleri değişen Doğrulama(Gerçekleme)larla indirgenebilir. $v_i = \alpha^{-h(m)} y^r r^s \bmod p$ hesaplanır ve sadece $v_1=1$ ise geçerli imzayı kabul eder. Şimdi v_1,3üs alma uygulanarak daha etkili hesaplanabilir. Toplam değer şimdi 5/8 [lgp] modüler çarpımı civarlarındadır. İmza Doğrulama(Gerçekleme) hesaplamaları imza üretimi hesaplamaları modül p ve modül ile yapıldığında modül p ile tamamen uygulanır.

5.69.Not:(Önerilen Parametre Boyutları) Zp^x içinde discrete logaritma problemi üzerinde verilen en son işlem ve 512 bit modül p sadece birleştirilmiş saldırılarından kenarda olan güvenliği sağlarlar. 1996'daki gibi en az 768 bitlerin p modülü önerilir. Uzun süre güvenlik için 1024 bit veya daha büyük modül kullanılmalıdır.

5.70.Not:(Geniş Sistem Parametreleri): Tüm girdiler (entıty), p ve α açık anahtar bölümlerinde gerekmediği durumda α üretecisi ve aynı asal p sayısını kullanmak için seçebilir.

i) El Gamal proje değişkenleri: Temel El Gamal imza projesinin birkaç değişkeni hedeflenmiştir. Bunların çoğu genel olarak işaretlenen denklemle ilgilidir. Uygun düzenlemeden sonra bu işaretlenen denklem u=h(m),v=r ve w=s iken u=av+kw mod(p-1) şeklinde yazılır. Diğer işaretlenmiş denklem , farklı sıralarda s,r ve h(m) değerleri üzerine almak için u,v,m izniyle sağlanabilir. Tablo 5.5'te 6 ihtimal listelenmiştir.

Tablo 5.5 Elgamal İşaretleme Denkleminin Değişimleri İşaretleme Denklemleri modül (p-1)'e Göre Doğrulama ise modül p'ye göre yapılır.

	u	**v**	**w**	**işaretleme denklemi**	**doğrulama**
1	h(m)	r	s	h(m)=ar+ks	$\alpha^{h(m)}=(\alpha^{\alpha})^r r^s$
2	h(m)	s	r	h(m)=as+kr	$\alpha^{h(m)}=(\alpha^{\alpha})^s r^r$
3	s	r	h(m)	s=ar+kh(m)	$\alpha^{s}=(\alpha^{\alpha})^r r^{h(m)}$
4	s	h(m)	r	s=ah(m)+kr	$\alpha^{s}=(\alpha^{\alpha})^{h(m)} r^r$
5	r	s	h(m)	r=as+kh(m)	$\alpha^{r}=(\alpha^{\alpha})^s r^{h(m)}$
6	r	h(m)	s	r=ah(m)+ks	$\alpha^{r}=(\alpha^{\alpha})^{h(m)} r^s$

5.71.Not:(El Gamal İmza Projesinin Değişikliklerinin Karşılaştırılması):

i) Tablo 5.5'te listelenen işaretleme denklemlerinin birkaçı Algoritma 5.64'te orginal El Gamal denkleminden daha etkili bir hesaplamadır. Örneğin Tablo 5.5' teki denklem 3 ve 4, s imzasını belirlemek için modüler ters almayı gerektirmez. Denklem 2 ve 5 a^{-1}mod(p-1)'i hesaplamak için işaretçiye ihtiyaç duyar fakat bu sabit özellik sadece bir kez hesaplama gerektirir.

ii) (2 ve 4 ile verilen) Doğrulama(Gerçekleme) denklemleri, r^r ifadesini içerir. İşaretleme denklemlerine dayanan imza projesinin güvenliğinin bir bölümü c sabitken $x^x \equiv c(\text{mod } p)$ şeklindeki bir ifadenin çözümünü bulmada yetersizdir.

iii)Genelleştirilmiş ElGamal İmza Projesi: Zp^x Çarpım grubunun oluşturulmasında esas olarak tanımlanan ElGamal dijital imza projesi herhangi bir sonlu (abelian) G grubunda çalışmak için doğru bir şekilde genelleştirilebilir. Algoritma 5.73 şifrelenebilen bir hash fonksiyonu n G'deki elemanların sayısı iken h: $\{0,1\}^x \rightarrow Z_n$ ihtiyaç duyar. H(r) olarak tanımlanan G' nin her bir r elemanı ikilik sistemde gösterildiği varsayılıyor.

5.72. Algoritma : Genelleştirilmiş ElGamal İmza Projesi İçin Anahtar Üretimi

Özet: Her bir girdi(entity) sonlu bir G grubunu,G' nin üretecini, açık ve özel anahtarları seçer. Her bir A girdisi aşağıdakileri yapmalıdır.

1. Üreteci α olan n. mertebeden devirli bir uygun G grubu seçer.
2. Rastgele bir gizli tamsayısı $1 \leq \alpha \leq n-1$ aralığında seçilir.
3. A' nın açık anahtarı (α,y), özel anahtarı a' dır.

5.73. Algoritma : Genelleştirilmiş ElGamal İmza Projesi İçin Anahtar Üretimi ve Doğrulama(gerçekleme)

Özet: A girdisi (entity) keyfi uzunluktaki ikilik gösterimdeki m mesajını işaretler. Herhengi bir B girdisi(entity) de A'nın açık anahtarını kullanarak bu imzayı doğrulayabilir.

1. İmza Üretimi:

A girdisi (entity) aşağıdakileri yapmalıdır.

a) Gcd(k,n)=1 ile $1 \leq k \leq n-1$ olmak şartıyla rastgele gizli bir k sayısı seçer.
b) Grup elemanı $r=\alpha^k$ hesaplanır
c) k^{-1} mod n hesaplanır
d) h(m) ve h(r) hesaplanır
e) $s=k^{-1}\{h(m)-ah(r)\}$ mod n hesaplanır
f) m için A'nın imzası (r,s) çiftidir.

2. Doğrulama(gerçekleme) : m üzerinde A'nın imzası (r,s) yi doğrulamak (gerçeklemek) için B aşağıdakileri yapmalıdır:

a) A'nın gerçek açık anahtarı (α,y)' yi elde eder.

b) h(m) ve h(r) hesaplanır.

c) $v_1=y^{h(r)}. r$ hesaplanır

d) $v_2\ \alpha^{h(m)}$ hesaplanır

e) sadece $v_1=v_2$ ise imzayı kabul eder.

5.74. Örnek Anahtar Üretimi: Küçük parametrelerle genelleştirilmiş ElGamal imzaları:

Anahtar Üretimi: F_2^5 üzerinde indirgenemez polinom ($f(x) = x^5+x^2+1$)'den oluşturulan sonlu F_2^5 halkasını ele alalım.Bu halkanın elemanları yalnız 00000 ile tablo 5.6'da gösterilen 31 tane ikilik gösterimde sıralı 5' lidir. α = (00010) elemanı, halkanın çarpılabilir devirli grubu olan G= F* için bir üretecidir. Bu G grubunun mertebesi n=31 dir. $h:\{0,1\}^x \rightarrow Z_{31}$ bir hash fonksiyonu olsun. A girdisi (entity) a=19 özel anahtarını seçer. ve $y=\alpha^a = (00010)^{19} = (00110)$ olur.A'nın açık anahtar (α=(00010) ; y =(00110)) 'dır.

İmza Üretimi :

Mesaj m =10110101 işaretlemek için A rastgele k=24 sayısını seçer ve ($r= \alpha^{24} = (11110)$ ve $k^{-1} \bmod 31 = 22$) hesaplanır. Daha sonra A, h(m)=16, h(r)=7 ve $s=22*\{16-(19)(7)\} \bmod 31 = 30$ hesaplar. m mesajı için A' nın imzası (r = (11110), s = 30).

İmza Gerçeklemesi (doğrulaması):

B, h(m)=16, h(r) = 7, $v_1 = v^{h(r)}r^s = (00110)^7 . (11110)^{30} = (11011)$ ve $v_2 = \alpha^{h(m)} = \alpha^{16} = (11011)$' i hesaplar. B,$v_1=v_2$ iken imzayı kabul eder.

Tablo 5.6: Bir α üretecinin kuvvetleri olarak F elemanları

i	α^i	i	α^i	i	α^i	i	α^i
0	00001	8	01101	16	11011	24	11110
1	00010	9	11010	17	10011	25	11001
2	00100	10	10001	18	00011	26	10111
3	01000	11	00111	19	00110	27	01011
4	10000	12	01110	20	01100	28	10110
5	00101	13	11100	21	11000	29	01001
6	01010	14	11101	22	10101	30	10010
7	10100	15	11111	23	01111		

5.75. Not: (Genelleştirilmiş Elgamal Güvenliği) Algoritma 5.73 güvenliğinin bir bölümü grup G de ayrık logaritma problemin çözülemezliğine dayanır. Not 5.66 daki güvenlik yorumlarının çoğu genelleştirilmiş Elgamal projesini uygular.

5.76. Not:(İşaretleme ve Doğrulama(Gerçekleme) İşlemleri) İmza üretimi grup G deki ve Z_n deki işlemleri gerektirir. İmza gerçeklemesi (doğrulaması) ise sadece grup G deki işlemleri gerektirir.

5.77. Not (Eliptik Eğrilerin kullanan Genelleştirilmiş Elgamal)

Bu tip gruplardaki ayrık logaritma problemi sonlu halka F_q üzerindeki eliptik eğrinin noktalarının kümasinden oluşturulan sonlu G (abelian) grubuna algoritma 5.73 uygulanması önerilir. Bu tip grublardaki ayrık logaritma problemi bir sonlu F_q halkasının çarpılabilir grubundaki ayrık logaritma probleminden çok daha zordur. $G=F_q^x$ şeklindeki gruplarda uygun tamamlamalar için q' dan daha küçüğü seçilebilir.

5.5.3. Schnorr İmza Projesi

Diğer iyi bilinen Elgamal projesinin(Alg.5.64) değişiği Schnorr imza projesidir. DSA ile bu teknik, bazı büyük p asal sayıları için Z_p^*'de q. mertebeden alt grupta çalışır. Bu metod ayrıca hash fonksiyon $h:\{0,1\}^x \rightarrow Z_p$ gerektirir. Schnorr imza projesinin anahtar üretimi, p ve q boyutları üzerinde kısıtlamaların olmaması haricinde DSA anahtar üretimi ile aynıdır.

5.78. Algoritma Schnorr İmza Üretimi ve Doğrulama(Gerçekleme)

Özet: A girdisi(entity), keyfi uzunlukta ikilik m mesajını işaretler. Herhangi bir B A'nın açık anahtarını kullanarak bu imzayı doğrulayabilir.

1. **İmza üretimi :** A girdisi (entity) aşağıdakileri yapmalıdır.
   a) $1 \le k \le q-1$ aralığında rastgele gizli bir k sayısı seçer.
   b) $r = \alpha^k \bmod p$, $e = h(m||r)$ ve $s = ae+k \bmod q$ hesaplanır
   c) m için Anın imzası (s,e) çiftidir.
2. **Doğrulama(Gerçekleme):** m üzerinde A'nın imzası (s,e)' yi doğrulamak (Gerçekleme) için B aşağıdakileri yapmalıdır.
   a) A'nın gerçek açık anahtarı (p,q, α, y) sağlar.
   b) $v = \alpha^s y^{-e}$ ve $e' = h(m||v)$ hesaplanır.
   c) Sadece $e' = e$ ise imzayı kabul eder.

5.79.Örnek(Küçük parametreler ile Schnorr İmza Projesi):

Anahtar üretimi: A, p = 129841 ve q = 541 asallarını seçer. Burada (p-1)/q = 240. Daha sonra rastgele g = 26346 $\in Z_p^x$ bir tamsayı seçer ve $\alpha = 26346^{240}$ hesaplanır. $\alpha \neq 1$ olduğunda α, Z_p^x 'da α ,541. mertebeli tek alt grubu üretir. Daha sonra A özel anahtar a=423 seçer ve $y = 26^{423}$ hesaplanır. A'nın açık anahtarı p = 129841, q = 541, α = 26, y = 115917'dir.

İmza üretimi: m=11101101 mesajını işaretlemek için, A , $1 \leq k \leq 540$ aralığında rastgele bir k=327 sayı seçer ve $r = 26^{327} \bmod p = 49375$ ve e = h(m||r) = 155 hesaplanır. Sonuç olarak, A s = 423.155+327 mod 541=431 hesaplanır. m için imza s = 431,e = 155'dir.

İmza gerçeklemesi (doğrulaması): B, $v = 26^{431}.115917^{-155} \bmod p = 49375$ ve e'=h(m||v)=155 hesaplanır. e=e' iken B imzayı kabul eder.

5.80.Not:(Schnorr Projesinin Performans Karakteristikleri) Algoritma 5.78 'deki imza üretimi modül p üs almayı ve modül q çarpımı gerektirir. Kulllanılan Hash algoritmasına bağlı olarak h(m‖r) 'yi hesaplamak için küçük olmalıdır. Doğrulama(Gerçekleme) iki üs alma modülü p'yi gerektirir. q mertebeli alt grup algoritma 5.64 ile etkin hesaplanamaz fakat ElGamal metodu yardımıyla üretilen imzalardan daha küçük imzalar sağlar.

5.5.4. Mesajı geri almalı ElGamal İmza Projesi:

5.5.2' de ElGamal imza projesi ve onun değişiklikleri rastgele ilaveli dijital imza projeleridir. Algoritma 5.81'in imza mekanizması imzadan kendiliğinden geri alınabilme özelliğine sahiptir. Bununla beraber bu ElGamal değişkliği rastgele mesajı geri almalı dijital imzayı sağlar.

Bu proje için işaretleme uzayı ms= Z_p^*, (p bir asal) ve imza uzayı S=Z_p x Z_q (q bir asaldır ve p-1'i böler) M' den M_S ' e kümesine bir R redundancy fonksiyonu alalım. Algoritma 5.81 için anahtar üretimi , p ve q boyutları üzerinde sınırlamalar haricinde DSA (Alg.5.54) anahtar üretimi ile aynıdır.

5.81. Algoritma : Nyberg-Rueppel İmza Üretimi ve Gerçeklemesi (doğrulaması):

Özet: A girdisi(entity) $m \in M$ mesajını işaretler. Herhangi bir B girdisi(entity) A'nın imzasını doğrulayabilir ve imzadan m mesajını geri alır.

1- **İmza üretimi**: A girdisi (entity) aşağıdakileri yapmalıdır:

a) $\hat{m} = R(m)$ hesaplanır.

b) $1 \leq k \leq q-1$ aralığında rastgele gizli bir k sayısını seçer ve $r = \alpha^{-k} \bmod p$ hesaplanır.

c) e = mr mod p hesaplanır.

d) $s = ae+k \bmod q$ hesaplanır.

e) m için A'nın imzası (e,s) çiftidir.

2- **Doğrulama(Gerçekleme):** m üzerinde A'nın imzası (e,s)'yi Doğrulamak (Gerçekleme) için B aşağıdakileri yapmalıdır:

a) A'nın gerçek açık anahtarı (p,q, α ,y) sağlar.

b) $a \le e \le p$ aralığını doğrular. Bu aralık sağlanmıyorsa imzayı reddeder.

c) $0 \le s \le q$ aralığını doğrular, sağlamıyorsa imzayı reddeder.

d) $v = \alpha^s y^{-e} \bmod p$ ve $\hat{m} = ve \bmod p$ hesaplanır.

e) $\hat{m} \in MR$ doğrular, $\hat{m} \notin MR$ 'nin imzayı reddeder.

f) $m=R^{-1}(\hat{m})$ geri alır.

5.82.Örnek: (Küçük parametrelerle Nyberg-Rueppel İmza Üretimi)

Anahtar üretimi: A girdisi (entity) p = 1256993 ve q = 3571 asallarını seçer. q,(p-1) böler; burada (p-1)/q =382'dir. Daha sonra A $g = 42077 \in Z_p^*$ rastgele sayısını seçer ve $\alpha = 42077^{352} \bmod p = 441238$ hesaplanır. $\alpha \neq 1$ iken α, Z_p^* içinde 3571 mertebeli tek devirli alt grubu üretir. Sonuç olarak A, a =2774 rastgele sayısını seçer ve $y = \alpha^a \bmod p = 1013657$ hesaplar. A'nın özel anahtarı a=2774 iken açık anahtarı p = 1256993, q = 3571, α = 441238, y = 1013657 'dır.

İmza üretimi: Mesaj m'i işaretlemek için A $\hat{m} = R(m)=1147892$ hesaplar. Daha sonra k=1001 seçer ve $r = \alpha^{-k} \bmod p = 441238^{-1001} \bmod p =1188935$, $e= \hat{m} r \bmod p = 138207$ ve $s = (2774)(138207)+1001 \bmod q = 1088$ hesaplar.

İmza doğrulaması: B, $v = 441238^{1088} \cdot 1013657^{-138207} \bmod 1256993 = 504308$ ve $\hat{m} =$ v.138207 mod 1256993 hesaplar. $\hat{m} \in$ MR'yi doğrular ve $m = R^{-1}(\hat{m})$ geri alır.

5.83.Not: Nyberg-Rueppel İmza Projesinin Güvenliği:

i) Algoritma. 5.81 temel ElGamal projesinin değişkliği olduğundan not 5.66' daki güvenlik aynen uygulanır. DSA gibi bu mesajı geri almalı ElGamal mekanizması iki tane bağlantılı fakat birbirinden farklı ayrık logaritma problemlerinin (not 5.58) zorluğuna dayanır.

ii) Algoritma.5.81 mesajı geri almayı sağladığından uygun redundancy fonksiyon r, sahte(taklit) varlık 'a karşı korunmayı gerektirir(not 5.10). RSA' daki durum gibi bu imza projesinin çarpılabilir yapısı redundancy r fonksiyonu seçildiğinde dikkatlice ele alınmalıdır. Aşağıdaki mümkün saldırı unutulmamalıdır. $m \in M$, $\hat{m} = R(m)$ ve (e,s) m bir imza olduğunu varsayalım. Daha sonra bazı k tamsayılar için $e= \hat{m} \alpha^{-k} \bmod p$ ve $s=ae+k \bmod q$ alınır. Bazı l tamsayılar için $\hat{m}^* = \hat{m} \alpha^l$ olsun. Eğer $s^* = s+l \bmod q$ ve $\hat{m}^* \in$ MR ise

$m^A = R^{-1}(\hat{m}^x)$ için (e,s*) geçerli bir imzadır. Bunu anlamak için, algoritma 5.81' in 2. adımındaki doğrulama algoritması göz önüne alır. $v \equiv \alpha^{s^*} y^{-e} \equiv \alpha^{s+1} \alpha^{-ae} \equiv \alpha^{k+l} \pmod p$ ve $ve \equiv \alpha^{k+l} \hat{m} \alpha^{-k} \equiv \hat{m} \alpha^{l} \equiv \hat{m}^{*}$' dır. $\hat{m}^* \in$ MR olduğundan taklit imza (e,s*) m^* için geçerli bir imza kabul edilecektir.

iii) Algoritma.5.81'in 2b adımında verilen 0<e<p doğrulaması (gerçekleme) önemlidir. m mesajı için A'nın imzasını (e,s) olduğunu düşünelim. $e = \hat{m}\, r \bmod p$ ve $s = ae+k \bmod q$ 'dir. Bir rakip kendi seçiminin m^* mesajı üzerindeki bir imzayı hesaplamak için bu imzayı kullanabilir. e^* 'i belirler ; şöyle ki $e^* \equiv \hat{m}^*\, r \pmod p$ ve $e^* \equiv e \pmod q$. (e^*,s) çifti , $0<e^*<p$ 'nin kontrolünün sağlanmadığı doğrulama(gerçekleme) algoritmalarını geçecektir.

5.84. Not:(Mesajı geri almalı ElGamal İmzasının Genelleştirilmesi) : Algoritma.5.81'in 1c adımında $e = \hat{m}\, r \bmod p$ ifadesi r anahtarı ile $\hat{m}$' yi şifrelemekte basit bir yol sağlar ve simetrik anahtar algoritmasını genelleştirebilir. $E = \{Er : r \in Z_p\}$ elemanıyla indekslendiğinde şifreleme çevirilerinin bir kümesi olsun ve $M_S = Z_p^*$ 'den $Z_p^{*'}$e bir 1-1,örten 'dir. $m \in M$ için $1 \le k \le q-1$ aralığında rastgele bir k sayısı seçer, $r = \alpha^k \bmod p$, $e = E_r(\hat{m})$ ve $s = ae+k \bmod q$ hesaplanır. (e,s) çifti m için bir imzadır.

Temel imza denklemi $s = ae+k \bmod q$, sonraki zamanda diğer girdiyle (entity) mesajı çevirmekte kullanılabilen simetrik anahtar için mesaj m ve A girdisinin (entity) özel anahtarını bağlayan bir araçtır.

5.6. Tek Zaman Dijital İmzalar: Tek zaman dijital imza projeleri genelde bir mesajı işaretlemekte, diğer yandan imzaları taklit edilebilen dijital imza mekanizmalarıdır. Yeni bir açık anahtar işaretlenmiş her mesaj için gereklidir. Tek-zaman imzaları doğrulamak (gerçekleme) için gereken açık bilgi geçerli parametreler olarak adlandırılır. Geçerlilik parametrelerinin asallanması için tek-zaman imzaları ile birleştirildiğinde, çoklu imzalar mümkündür.

Çoğu fakat hepsi değil tek zaman dijital projeleri imza üretimi ve doğrulamanın (gerçekleme) çok etkili olduğu avantajına sahiptir. Tek-zaman dijital imza projeleri, düşük hesaplama karmaşığının gerektiği yerlerde şifre kartları gibi uygulamalar da kullanışlıdır.

5.6.1. Rabin Tek-Zaman İmza Projesi:

Rabin'in tek zaman imza projesi herhangi bir dijital imza çeşidi için ilk önerilerden biriydi. Tek mesajın işaretlenmesine izin verir. İmza gerçeklemesi (doğrulaması),işaretçi ve sorgulayıcı arasındaki etkileşime ihtiyaç duyar. Diğer dijital imza projelerine benzemez.

Doğrulama(Gerçekleme) sadece bir kez yapılabilir. Pratik değişken tarihi sebebleri için burada gösterilir. Bu bölümde kullanılan semboller tablo 5.7'de verilir.

Tablo 5.7 Rabin tek-zamanlı imza projesi için notasyon

Sembol	Anlamı
M_0	0^l = l bir uzunluğundaki dizinin bütün 0' ları
$M_0(i)$	$0^{l-e}\|\|b_{e-1}...b_1b_0$ burada $b_{e-1}...b_1b_0$ i' nin ikili gösterimidir.
K	l bit uzunluğundaki dizinin bir kümesi
E	Bir anahtar uzayı K tarafından indekslenen şifre dönüşümlerinin kümesi
E_t	Bir şifreleme dönüşümüdür ve her bir E_t l bit diziden l bit diziye tasvirdir.
h	$\{0,1\}^*$' dan $\{0,1\}^l$' ye açıkca bilinen bir hash fonksiyonudur.
n	Güvenlik parametresi olarak kullanılan sabir bir pozitif tamsayıdır.

5.85.Algoritma Rabin Tek Zaman İmza projesinin Anahtar Üretimi

Özet: Her A girdisi(entity) simetrik anahtar şifreleme projesi E'yi seçer, rastgele 2n bit dizilerini genelleştirir ve değişken parametrelerinin kümesini oluşturur.

Her A girdisi(entity) aşağıdaki gibidir:

1- Simetrik anahtar projesi E'yi seçer.

2- Rastgele gizli 2n dizileri $k_1,k_2,....,k_{2n} \in K$'yı genelleştirir. Her biri 1 bit uzunluğundadır.

3- $1 \le i \le 2n$ aralığında $y_i = E_{k_i}(M_0(i))$ hesaplanır.

4- A'nın açık anahtarı $(y_1,y_2.....y_{2n})$; Özel anahtarı $(k_1,k_2,....,k_{2n})$ dır.

5.86. Algoritma Rabin Tek Zaman İmza Üretimi ve Doğrulama(Gerçekleme)

Özet: A girdi (girdi(entity))i keyfi uzunlukta ikilik m mesajını işaretler. İmza gerçeklemesi (doğrulaması) A ile interaktiftir.

1- İmza Üretimi : A girdi (girdi(entity))i aşağıdaki gibidir.

a) h(m)'i hesaplanır.

b) $1 \le i \le 2n$ aralığında $s_i = E_{k_i}(h(m))$ hesaplanır.

c) m için A'nın imzası $(s_1,s_2.....s_{2n})$ dir.

2- Doğrulama(Gerçekleme) : m üzerinde A'nın imzası $(s_1,s_2.....s_{2n})$'nı doğrulamak (Gerçekleme) için B aşağıdakileri yapmalıdır.

a) A'nın gerçek açık anahtarı $(y_1,y_2.....y_{2n})$ sağlar.

b) h(m)'i hesaplanır.

c) $1 \le j \le n$ için $1 \le rj \le 2n$ olmak şartıyla n farklı rastgele rj sayılarını seçer.

d) $1 \le j \le n$ olmak şartıyla A'dan k_{rj} anahtarlarını tahmin eder.

e) $1 \le j \le n$ aralığında herbiri $Z_j = y_{rj}$ kontroluyle ve $z_j = E_{k_{rj}}(M_0(r_j))$ hesabıyla alınan anahtarların doğruluğunu doğrular.

f) $1 \le j \le n$ olmak üzere $S_{r_j} = E_{k_{r_j}}(h(m))$ doğrular.

5.87. Not (Rabin'in Tek Zaman İmzalarının Anahtar Boyutları) E_t l bit çıkış yaptığında algoritma 5.86 da açık ve özel anahtarların herbiri 2nl bitlerinden oluşur. n=80 ve l= 64 için anahtarların herbiri 1280 byte uzunluğundadır.

5.88. Not (Tartışma Çözümü) Algoritma 5.86'da kullanılan işaretci A ve sorgulayıcı B arasındaki potansiyel tartışmayı çözmek için aşağıdaki işlem izlenir.

1- B, m ve imza $(s_1, s_2 ..... s_{2n})$ ile TTP'yi sağlar.

2- TTP, A'dan $k_1, k_2 ..... k_{2n}$'i sağlar.

3- TTP, $1 \le i \le 2n$ olmak koşuluyla $y_i = z_i$ kontrolu ve $(z_i = \_\_\_)$ hesabıyla özel anahtarın doğruluğunu doğrular. Bu başarısız sa TTP kuralları B'nin lehinedir.

4- TTP $u_i = E_{k_i}(h(m))$ hesaplanır. $1 \le i \le 2n$ olmak üzere i'nin çoğu n değeri için $u_i = s_i$ ise imza sahte(taklit)liğini bildirir ve TTP kuralları A'nın lehinedir. n+1 veya i'nin daha fazla değeri $u_i = s_i$ yi veriyorsa imzanın geçerliliği varsayılır. Ve TTP kuralları B'nin lehinedir.

5.89. Not (Tartışma Çözüm Protokolünün Mantığı) Eğer B, yeni m' mesajı üzerinde A'nın imzasını işlemeye teşebbüs etmişse B en az k' anahtarlarını tesbit etmeye ihtiyaç duyar. Böylece en az i'nin n+1 değerleri $u_i = s_i$ verir veya m' belirler, şöyleki $h(m) = h(m')$ dir. Simetrik anahtar algoritması ve Hash fonksiyonu uygun seçilirse bu mimkün olmamalıdır. Eğer A daha sonra reddedilebilir. İmzayı oluşturmaya teşebbüs ederse i'nin n değerleri için $u_i = s_i$'yi sağlamalıdır. Ve B'nin Doğrulama(Gerçekleme) prosedürünün 2c adımında bu n değerlerini seçtiğini ummalıdır. Sadece $1/(2_n n)$ iken mümkündür.

5.90. Not: A, Rabinin tek zaman projesinde verilen özel anahtar ile n çok bir mesaj işaretleyebilir, diğer yandan A, yeni mesaj üzerinden imzaları taklit etmek için B'ye yetki verecek ve özel anahtar değerlerinin n+1'ini veya daha fazlasını gösterecektir. Bir imza, 2n özel değerlerinin n den daha fazlasını göstermeden sadece bir kez sorgulanır.

5.6.2. Merkle Tek Zaman İmza Projesi:

Merkle tek zaman imza projesi imza gerçeklemesi (doğrulaması)nda Rabin'den farklıdır. İşaretçiyle interaktif değildir. Bir TTP veya diğer güvenilen araçlar algoritma 5.9'da oluşturulan değişken parametrelerin doğruluğunu gerektirir.

5.91. Algoritma: Merkle Tek Zaman İmza Projesinin Anahtar Üretimi

Özet: n bit mesajları işaretlemek için A t=n+[lgn]+1 değişken parametrelerini genelleştirir. Her A girdisi(entity) aşağıdaki yolu izlemelidir.

1- $t = n+[\lg n]+1$ rastgele gizli dizileri $k_1,k_2,...k_t$ seçer. Herbiri l bit uzunluktadır.

2- $1 \le i \le t$ olmak koşuluyla $v_i = h(k_i)$ hesaplanır. Burada h görüntü dirençli hash fonksiyon $h:\{0,1\}^x \rightarrow \{0,1\}^{l'}$dir.

3- A'nın açık anahatrı $(v_1,v_2,...v_t)$, özel anahtarı $(k_1,k_2,...k_t)$dir.

n bit mesaj m'i işaretlemek için 1 bit dizisi $w=m\|e$, m'de 0'ların sayısı için e ikilik gösterildiğinde düzenlenir. eğer gerekli ise C'nin 0 la doldurulan yüksek mertebeli bitler ile [lgn]+1 bit uzunluklu bir bit dizisi olduğu farz edilir. Bununla beraber w, t=n+[lgn]+1 bit uzunluklu bir bit dizisidir.

5.92. Algoritma Merkle Tek zaman İmza Üretimi ve Gerçeklemesi (doğrulaması)

Özet: A girdi(entity)i n bit uzunluklu bir ikilik m mesajını işaretler. B girdisi(entity) A'nın açık anahtarını kullanarak bu imzayı doğrulayabilir.

1- İmza Üretimi : A girdi (girdi(entity))i aşağıdakileri yapmalıdır.

a) A'nın gerçek açık anahtarı$(v_1,v_{2,}.....v_u)$ sağlar.

b) m'de 0'ın sayıları için ikilik gösterimdeki e'yi hesaplanır.

c) $w=m||c=(a_1,a_2...a_t)$ kurar.

d) $A_{ij} = 1$, $1 \le j \le u$ iken w'de $i_1<i_2<...<i_u$ koordinat pozisyonunu belirler.

e) $1 \le j \le u$ hepsi için sadece $v_{ij}=h(s_j)$ ise imza kabul edilir.

5.93. Not:Algoritma.5.92'nin Depolama ve Hesaplama Gereksinimleri

i) Özel anahtarın l.(n+[lgn]+1) bitleri ve geçerli parametreler için birikim bitlerini gerektiren k, n bit m mesjını işaretlemek içindir. İmza n-k ikilik gösteriminde birlerin sayısı k' iken birikimin l.(k+k') bitlerine ihtiyaç duyar. Örneğin; n=128, l=64 ve k=72 ise açık ve özel anahtarların herbiri 8704 bit (1388 byte) gerektirir. İmza ise 4800 bit (600 byte) gerektirir.

ii) Özel anahtar k_i'ler tek seed değerinden kurularak daha küçük yapılabilir. Örneğin; k^x en az l bit uzunluklu bir dizisi ise $1 \le i \le t$ iken $k_i=h(k^x\|i)$ kurulur. Sadece seed k^x birikime ihtiyaç duyduğunda özel anahtarın boyutu şiddetle indirgenir.

iii) İmza üretimi, hesaplamaya ihtiyaç yokken çok hızlıdır. İmza gerçeklemesi (doğrulaması) n+[lgn]+1 değerlerinden daha az hash fonksiyon değerlerine ihtiyaç duyar.

5.94. Not : Algoritma 5.92 Açık ve özel anahtarlarının herbiri l.(n+[lgn]+1) bitlerine ihtiyaç duyar. Açık anahtar mesaj bitlerinin hesaba katıldığı işaretleme algoritma sayesinde bu genişlikte olmalıdır. İşaretlenmiş algoritma belli bir zaman da bir bitten daha fazlasını hesaba dizisi proje daha etkili yapılabilir.

A girdisinin (entity) kt.bit m mesajını işaretlemek istediğini düşünelim.Her biri 0 ve 2k-1 arasını kapsayan bir tamsayıyı gösterdiğinde ve mi k.bit uzunluğunda olmak şartıyla $m = m_1||m_2||...||m_t$ yazılır. $U = \sum_{i=1}^{t}(2^k - m_i) \le t2^k$ .U tanımlanır.u, $lgU \le [lgt]+1+k$ bitleriyle gösterilebilir. Eğer $r = [([lgt]+1+k)/k]$ ise u,ui k.bit uzunluğundayken $U = u_1||u_2||...||u_r$ şeklinde yazılabilir. $w = m_1||m_2||...m_t||u_1||u_2||...||u_r$ bit dizisi kurulur.t + r rastgele bit dizileri $(k_1.k_2...k_{t+r})$ genişletilir ve $1 \le i \le t+r$ iken $v_i = h^{2^k-1}(k_i)$ hesaplanır. Değişen projenin özel anahtarı $(k_1.k_2...k_{t+r})$ ve açık anahtarı $(v_1.v_2...v_{t+r})$'dir.m için imza $1 \le i \le z$ aralığında $s_i = h^{m_i}(k_i)$ ve $1 \le i \le r$ aralığında $s_i = h^{u_i}(k_{t+i})$ iken $(s_1.s_2...s_{t+r})$'dir. Bununla beraber h^c,h'nin c kat düzenini kendi kendine ifade eder. Orginal proje ile kontrol toplam gibi mesaja eklenen bitler aşağıdaki gibidir.

$s_i = h^a(k_i)$ elemanı verilir.Rakip, $0 \le \delta \le 2^k$-a için $h^{a+\delta}(k_j)$ kolayca hesaplayabilir.Fakat h tek yol hash fonksiyon ise $\delta > 0$ için $h^{a-\delta}$ hesaplanamaz.Yeni bir mesajın üzerindeki imzayı taklit etmek için rakip sadece,eklenen kr bitleri üzerindeki gerekli hash değerlerini hesaplayarak onun için imkasızı yapacak kontrol-toplam değerlerini indirgeyebilir.

5.7 Diğer İmza Projeleri

5.7.1.Keyfi Digital İmzalar:

5.95.Tanım: Bir keyfi dijital imza projesi, imza üretimi ve doğrulama(gerçekleme) bölümleri gibi TTP gerektirmesi şartıyla dijital imza mekanizmasıdır.

Algoritma. 5.109 K anahtar uzayı iken $E=\{E_k:k \in K\}$ simetrik anahtar şifrelenmiş algoritmaya ihityaç duyar.Herbir E_k'nın girişleri (entity) ve çıkışlarının l-bit dizileri olduğunu farzedelim ve $h:\{0,1\}^* \rightarrow \{0,1\}^l$ tek-yol bir hash fonksiyon olsun. TTP gizli tutulan $k_T \in K$ anahtarını seçer. Sırasıyla imzayı doğrulamak(gerçekleme) için girdi (entity) TTP ile simetrik anahtarı paylaştırmalıdır.

5.96. Algoritma: Keyfi İmzalar İçin Anahtar Üretimi

Özet: Her girdi(entity) bir anahtar seçer ve TTP için doğruluk ile gizliliği taşır. A girdisi(entity):

1- $k_{A,}$ E_K keyfi gizli anahtarı seçer.

2- Gizlice ve birkaç ortalamayla TTP için geçerli k_A elde edilebilirliğine çalışılır.

5.97.Algoritma:Keyfi İmzalar İçin İmza Üretimi ve İmza Doğrulaması(gerçeklemesi)

Özet: A girdisi(entity) Ek_A' yı kullanarak imzaları genelleştirir. Herhangi bir B girdisi (entity), A'nın imzasını TTP'nin zıt işlemiyle sorgulayabilir.

1- İmza üretimi: m mesajı işaretlemek için A girdisi (entity),

a) H= h(m) hesaplanır.

b) u= E_{KA} (H) alınarak E ile H 'yi şifreler,

c) TTP için birkaç kişisel I_A dizisiyle yalnız u gönderir.

d) TTP H'ı alarak Ek_A^{-1} (u) hesaplanır.

e) TTP s=E_{kT} (H||I_A)hesaplanır ve s'yi A'ya gönderir.

f) m için A'nın imzası S'dir.

2- Doğrulama(Gerçekleme): B girdisi (entity) aşağıdakiler yapılarak m üzerinde A'nın imzasını sorgular.

a) B (v=E_{kB}(s)) hesaplanır.

b) B, TTP için v'y ive birkaç kişisel I_B dizisini gönderir.

c) TTP s'yi alarak Ek_B^{-1}(v) hesaplanır

d) TTP E^{-1}_{kA} (u)hesaplanır ve w'yu B'ye gönderir.

e) B‖I_A alarak Ek_B^{-1}(w) hesaplanır.

f) B, m'den H'=h(m) hesaplanır.

g) sadece H'=H ise B imzayı kabul eder.

5.98. Not: (Keyfi İmza Projesinin Güvenliği): Algoritma. 5.109'un güvenliği simetrik-anahtar şifreleme projesinin seçimine ve doğru bir tarzda paylaşılan anahtar dağılım yeteneğine dayanır.

5.99. Not: (Keyfi İmzaların Performans Karakteristiği): Simetrik anahtar algoritması genellikle açık anahtar tekniklerinden daha hızlıyken Algoritma. 5.109 tarafından imza üretimi ve gerçeklemesi (doğrulaması) nispeten etkilidir.

5.7.2 ESIGN

Esign, sayıların çarpanlarının alınma zorluğuna dayanan güvenliğin diğer bir dijital imza projesidir. Ilaveli bir imza projesidir ve tek-yol hash fonksiyon($h:\{0,1\}^* \rightarrow Z_n$) gerektirir.

5.100. Algoritma: ESIGN'ın Anahtar Üretimi

Özet: Her girdi(entity) açık ve özel anahtarı oluşturur. A girdisi(entity):

1- $p \geq q$ p ve p aynı uzunlukta olmak şartıyla keyfi p, q asallarını seçer.

2- $n=p^2q$ hesaplanır.

3- $k \geq 4$ pozitif bir sayı seçer.

4- A'nın açık anahtarı (n,k): özel anahtarı (p,q)'dur.

5.101. Algoritma: ESIGN İmza Üretimi ve Gerçeklemesi (Doğrulaması)

Özet: İşaretlenmiş Algoritma., mesajla belirlenen kesin bir anayüzde yararlana s^k mod n şeklinde bir s sayısını hesaplanır. doğrulama(gerçekleme) belirlenen anayüzde gerçekten yalanlayan s^k mod n ispat eder.

1- İmza Üretimi: Keyfi uzunluklu bir bit dizisi olan m mesajını işaretlemek için A girdisi (entity):

a) v=h(m) hesaplanır

b) $0 \leq x < pq$ aralığında keyfi gizli x sayısını seçer.

c) $w=[((v-x_k) \bmod n)/(pq)]$ ve $y=w.(kx^{k-1})^{-1} \bmod p$ hesaplanır

d) $s=x + ypq \bmod n$ hesaplanır

e)m için A'nın imzası s'dir.

2- Doğrulama(Gerçekleme): m üzerinde A'nın imzasını doğrulamak(gerçekleme) için B:

a) A'nın doğruluk açık anahtarını (n,k) sağlar.

b) $u=s^k \bmod n$ ve z=h(m) hesaplanır.

c) $(z \leq u \leq z+2^{[2/3 \log n]})$ ise imzayı kabul eder. Değilse reddeder.

5.102. Örnek: Algoritma. 5.53'te, $0 \leq m < n$ aralığında sayılarının olduğu mesajları h(m)=m olan hash Fonksiyon alır.

Anahtar Üretimi: A, p=17389 ve q=15401 asallarını seçer ve $n=p^2q=4656913120721$ hesaplanır. A'nın açık anahtarı (n=4656913120721,k=4) özel anahtarı (p=17389, q=15401)'dir.

İmza Üretimi: m=3111527988477 mesajını işaretlemek için, A v=h(m)= 3111527988477 hesaplanır ve $0 \leq x < pq$ aralığında x=14222 sayısını seçer. Daha sonra A

$w=[((v-x^k) \bmod n)/(pq)] = [2848181921806/267807989]=810635.16414]=10636$ ve $y=w(kx^{k-1})^{-1} \bmod p=10636(4*14222^3)^{-1} \bmod 17389=9567$ hesaplanır. Sonuç olarak A, $s= x+ypq \bmod n =2562119044985$ imzasını hesaplanır.

İmza Gerçeklemesi (Doğrulaması): B, A'nın açık anahtarı (n=4656913120721) 'i sağlar ve $u=s^k \bmod n=3111751837675$ hesaplanır. $3111527988477 \leq 3111751837675 \leq 3111527988477+2^{29}$ olduğundan B imzayı kabul eder.

5.103.Not: Esign Güvenliği

i) Algoritma. 5.53'teki $n=p^2q$ modülü, p'nin tekrarlanan faktörü sayesinde RSA modülünden farklıdır. Bu formun bilinmeyen modülünün faktöriyelini almak iki farklı asalın çarpımı olan sayılardan daha kolay olup olmadığı bilinmiyor.

ii) m mesajı için geçerli imza s verilir. Rakip, $h(m), h(m') \leq u \leq h(m') + 2^{[2/3 \lg n]}$aralığındaysa m' mesajı için imzayı işleyebilir. Bulunan bu özellik ile m' , s, onun için bir imza olacaktır. h(m) ve h(m') yüksek mertebeli (lgn)/3 bitlenirse düzenlenirse bu oluşacaktır. h'ın keyfi bir fonksiyon olarak tanımlandığını düşünelim. İncelemeden önce m' nün $2^{(lgn)/3}$ faklı değerlerini denemesi bekleyecektir.

iii) h(m) ve h(m') yüksek mertebeli (lgn)/3 bitlerinde düzenlenen m' ve m mesaj çiftlerini bulmak taklit etmek için mümkündür. Birtday paradox ile ile 0 $(2^{(lgn)/6})$ denemelerindeki bir çifti bulması beklenebilir. Eğer rakip m'i işaretlemek için yasal işaretçiyi alabilirse aynı imza m' için bir imza olacaktır.

iv) Gerekli n sayısının boyutu için n'in faktöriyelini almak mümkün değildir. (ii) ve (iii) tamamen ihtimalleri benzemez.

5.104. Not: Algoritma. 5.53'teki imza üretimi çok etkilidir. k'nın küçük değerleri için yoğun hesaplamalar gerektirir. Bölüm 1c adımında gereken ters modülerdir. Bu işlemi tamamlamak dayanan bu p modülüyle modüle çarpımlarının küçük sayılarının karşılığıdır. k=4 ve 768 bit n modülü için Esign imza üretimi, modüllerin boyutlarının bir eşiyle RSA imza üretiminden daha hızlı, büyüklük sırası bir ve iki arasında olabilir. İmza sorgusu ayrıca çok etkilidir ve küçük açık üs ile RSA karşılaştırılabilir

5.8. İlave Fonksiyonel İmzalar: Bu bölümde tanımlanan mekanizma asıllama ve kabul etmenin ötesinde fonksiyonellik sağlar.

5.8.1. Kör İmza Projeleri

Kör imza projeleri gönderici A ve işaretleyici B arasında iki grup protokoldür. A, B'nin işaretlediği ve A'ya geri döndürdüğü bilgiyi gönderir. Bu imzadan, A, A'nın ilk

seçimi m mesajı üzerinde B'nin imzasını hesaplayabilir. Protokolün tamamında B, ne mesajı ne de onunla birleşmiş imzayı bilir.

Kör imzanın amacı, B işaretçisinin, imzanın ve işaretlenen mesajın incelenmesine engel olmaktır .Bununla beraber daha sonra A göndericisiyle işaretlenmiş mesajı birleştiremez.

5.105. Örnek: (Kör İmza Uygulamaları): Kör imza projesi, gönderici A'nın (müşterinin) protokolün özel aralığı için $S_B(m)$ imzasını ve m mesajını birleştirebilme yeteneği olan B işaretçisini (bankayı) istemediği uygulamalar vardır. A'nın harcayabildiği parasal değerinin gösterebildiği m mesajı gibi elektronik kasa uygulamalarında önemli olabilir. m ve $s_B(m)$ödeme için B'ye sunulduğunda B grubun verdiği işaretlenmiş mesajı anlayamaz. Bu adı bilinmeyen kalan için A'ya izin verir. Böylece harcama örnekleri gösterilemez.

Bir kör imza protokolü aşağıdaki companentlere ihtiyacı vardır.

1- B işaretçisi için bir dijital imza mekanizması $S_B(x)$, x üzerinde B'nin imzasını gösterir.

2- Sadece göstericiye bildirilen f ve g fonksiyonları şöyle ki $g(S_B(f(m))= SB(m).f$ görünmeyen fonksiyon g görünen fonksiyon ve f(m) görünmemiş mesajdır.

Özellik 2, S_B ve g seçimlerinde birkaç sınırlama yer alır.

5.106. Örnek: (RSA'ya Dayalı Görünmez Fonksiyon) n=pq iki büyük keyfi asal sayının ürünü olsun. B girdisinin(entity) işaretlenmiş Algoritma. S_B'si özel anahtar d ve açık anahtar (n,e) ile RSA imza projesidir. k, gcd(n,m)=1 ile sabit sayıdır. f: $Z_n \rightarrow Z_n$ görünmeyen fonksiyon $f(m)=m.k^e$ mod n ile ifade edilir. Ve görünen fonksiyon g: $Z_n \rightarrow Z_n$ $g(m)=k^{-1}$ m mod n ile ifade edilir. F,g ve S_B, $g(S_B(f(m)))= g(S_B(mk^e \text{ mod } n))=g(m^dk \text{ mod } n)=m^d \text{ mod } n=s_B(m)$'nin bu seçimleri 2 özellik gerektirir.

Protokol 5.59 örnek 5.58' de tarif edilen f ve g fonksiyon ve dijital mekanizmasının kullandığı bir imza projelerini sunar.

5.107. Chaum's Kör İmza Protokolü

Özet: A gönderisi bir görünmeyen mesaj üzerinde B'nin imzasını alır. Bundan, A, kendi tarafından öncelikle seçtiği m mesajı üzerinde B'nin imzasını hesaplanır. ($0 \le m \le n-1$ olmak şartıyla) B, ne m ile birleştirilmiş imza bilgisine ne de m'in bilgisine sahip değildir.

1- Sembole gösterimi: B'nin RSA açık ve özel anahtarları (n,e) ve d'dir. k, $0 \le k \le n-1$ ve gcd(n,k)=1 şeklinde belirlenen A tarafından keyfi seçilmiş gizli bir sayıdır.

2- Protokol hareketleri

a) (görünmeyen) A $m^*=mk^e$ hesaplanır ve bunu B'ye gönderir.

b) (işaretlenen) B, A'ya gönderdiği $s^*=(m^*)^d$ hesaplanır.

c) (görülen) A, m üzerindeki B'nin imzası olan $s=k^{-1}s^*$ mod n hesaplanır.

5.8.2. İnkar Edilemez İmza Projeleri

Bu projeler, imza doğrulama(gerçekleme) protokolünün ihtiyaç duyduğu işaretçinin ters işlemlerinde dijital imzalardan farklıdır. Aşağıda bu projenin uygulandığı iki senaryo tarif edilmiştir.

5.108. Örnek: (İnkar Edilemez İmzaların Senaryosu)

i) A girdisi(entity) (müşteri), B girdisi (entity) (banka) tarafından kontrol edilen güvenlik bölgesini kazanmak ister. Güvenlik bölgesi, örneğin güvenli-heposit bir kutu odası olsun. B, girdisi (entity) kabul edilmeden önce zamanı ve veri dökümanlarını işaretlemek için A'ya ihtiyaç duyar. Eğer A inkar edilemez imzayı kullanırsa B, imza doğrulama(gerçekleme) işlemine A'nın gerektirdikleri olmadan A'nın kolaylıkla kullanmasını sağlayamaz.

ii) Büyük birlik A'nın yazılım paketini oluşturduğunu farzedelim. A, bu paketin kopyasını almaya karar veren B girdisi(entity) için paketi işaretler ve satar ve 3.c grubuna tekrar satar. C, A'nın ters işlemi olmadan yazılımın doğruluğunu denetleyemez. Tabii ki bu senaryo, kendi imzasıyla geri işaretlenen paketten B'yi sağlamaz. Fakat B için kayıp birlik A adı ile birleştirilmiş market avantajını sağlar. Ayrıca B'nin aktivitelerini izlemek kolay olacaktır.

5.109. Algoritma: Algoritma. 5.122 için anahtar üretimi:

Özet: Her girdi(entity) özel ve açık anahtarı seçer. A girişi:

1- q asal olduğu p=2q+1 keyfi asalını seçer.

2- Zp^* içinde q mertebesinin alt grubu için α yöneticisi seçer.

2.1 $\beta \in Zp^*$ keyfi bir eleman seçer ve $\alpha = \beta^{(p-1)/g}$ hesaplanır.

2.2 α=1 ise 2.1 adıma gider.

3- $\alpha \in \{1,2,.....q-1\}$ keyfi bir sayı seçer ve $y=\alpha^a$ mod p hesaplanır.

4- A'nın açık anahtarı (p,α,y) iken özel anahtarı a'dır.

5.110. Chaum-Von Antverpen İmkansız İmza Projesi

Özet: A, Zp^* içinde q mertebesinin alt grubuna ait m mesajını işaretler. B girdisi (entity) A'nın ters işlemiyle bu imzayı sorgular.

1- İmza Üretimi: A girdisi (entity):

a) $s=m^a$ mod p hesaplanır

b) m mesajı üzerinde A'nın imzası S'dir.

2- Doğrulama(Gerçekleme): B'nin protokolü m üzerinden A'nın imzası s'yi doğrulamak(gerçekleme) için;

a) B A'nın doğruluk açık anahtarı (p, α, y) sağlar.

b) B $x_1x_2 \in \{1,2,.....,q-1\}$ gizli keyfi sayılar seçer.

c) B, $z=s^{x1}y^{x2} \bmod p$ hesaplanır ve A'ya z'yi gönderir.

d) A, $(aa^{-1} \equiv 1 \pmod q$ iken) $w=(z)^{a-1}$ hesaplanır ve w'yu B'ye gönderir.

e) B, $w'=m^{x1}\alpha^{x2}$ hesaplanır ve sadece w=w' ise imzayı kabul eder.

5.111. Not:İnkar Edilemez İmzaların Taklitlerinin Bulunması: m mesajı için A'nın imzasını sahteliğinin (taklit) s olduğunu $(s \neq m^a \bmod p)$ düşünelim. Daha sonra Algoritma. 5.122. de kabul edilen imzanın B ihtimali sadece 1/q olsun bu ihtimal rakibin hesaplama kaynaklarından bağımsızdır.

5.112. Not:İmzaların Saklanması: İşaretçi A, 3 yoldan biriyle Algoritma. 5.122 tarafından oluşturulan geçerli imzayı reddetmek için teşebbüsde bulunabilir.

i) Algoritma. 5.122.'nin doğrulama(gerçekleme) protokolune katılması reddedilir.

ii) Doğrulama(gerçekleme) protokolu yanlış uygulanır veya

iii) Doğrulama(gerçekleme) protokolünün başarılı olmasına rağmen imzanın sahte(taklit) olduğu iddia edilir.

Protokol 5.125, Algoritma. 5.122'nin doğrulama(gerçekleme) protokolünü iki kez uygular ve daha sonra A'nın doğru uyguladığı protokolü doğrulamak (gerçekleme) için bir kontrol uygular.

5.113.Protokol: Chaum–Von Antverper inkar edilemez imza projesi için reddedilen protokol **Özet:** Bu protokol, ne Algoritma. 5.122 kullanarak A reddedilen imza s için teşebbüste bulunduğuna karar verir. Ne de imza sahteliğine(taklit)

1- B, A'nın doğruluk açık anahtarı (p, α, y) sağlar.

2- B, $x_1,x_2 \in$ ... keyfi gizli sayılar seçer, $z=s^{x1}y^{x2}$ hesaplanır. z'yi A'ya gönderir.

3- A ,($aa^{-1} \equiv 1 \pmod q$) iken w = hesaplanır ve w'yi B'ye gönderir.

4- $w= m^{x}{}_1 \alpha^{x}{}_2$ ise B s imzasını kabul eder ve protokol kırılır.

5- B,$x_1,x_2 \in \{1,2,....,q-1\}$ keyfi gizli sayılar seçer, $z'=s^{x'\ 1}y^{x'\ 2}$ hesaplanır. z' A'ya gönderir.

6- $w'= (z')^{a-1}$ hesaplanır ve w'nü B'ye gönderir.

7- $w'= m^{x'\ 1}\alpha^{x'\ 2}$ ise B s imzasını kabul eder. Ve protokol kırılır.

8- B $c=(w\alpha^{-x2})^{x'1}$ve $c'=(w'\alpha^{-x'2})^{x1}$ hesaplanır. c=c' ise B s'nin sahteliğini(taklit) sonuçlandırır. Diğer yandan B imzasının geçerliliğini sonuçlandırır ve A, reddedilen s için teşebbüste bulunur.

5.114.Not:m bir mesaj olsun ve s'nin m üzerinde A'nın imzası olduğunu düşünelim.

i) s sahte(taklit) ise (örneğin $s \neq m^a \bmod p$ gibi) A ve B protokol 5.125 doğru olarak izlerse w= w'dür.

ii) S'nin gerçekten m üzerinde A'nın imzası olduğunu düşünelim. ($s=m^a \bmod p$ gibi) B'ninde protokol 5.125 doğru olarak izlediğini farzedelim fakat büyük A'nın değil. Daha sonra w=w' ihtimali sadece 1/q dur.

5.115.Not: İnkar Edilemez İmzanın Güvenliği:

i) Algoritma. 5.122.'nin güvenliği Zp^* içinde q mertebesi devirli alt grubundaki ayrık logaritma probleminin algoritma probleminin intractability dayanır.

ii) Sorgulayıcı B'nin Algoritma. 5.122.'nin 2. adımındaki karşılıklı mesajları ve ayrıca protokolde kullanılan x_1,x_2 keyfi değerlerinin kaydettiğini düşünelim. Bir 3-grup C, imza s'nin gerçeklemesi (doğrulaması) gibi B'den bu transcript'i asla kabul etmemelidir. İşaretçi A'nın paylaşımı olmadan B'nin Algoritma. 5.122'nin 2. Adımının başarılı transcriptini nasıl başarabildiğini göstermesi yeterlidir. B, m mesajını x_1,x_2 sayılarını ve [1, a-1] aralığında L'yi seçer ve $s=.((m^{x1}\alpha^{x2})^{l-1}y^{-x2})x_1^{-1} \bmod p$ hesaplanır. B'den A'ya protokol mesajı $z=s^{x1}y^{x2}$ olacaktır ve A'dan B'ye $w=z^l \bmod p$ olacaktır. Algoritma. 5. 122. m mesajı için A'nın geçerli imzası gibi s'yi kabul edecektir.

Bu argüman, işaretçiyle direkt olarak interacting tarafından imzanın sadece sorgulanabileceğini ispat eder.

5.8.3. Başarısızlıkla Sonuçlanan İmza Projesi:

Bu proje, sahte(taklit) tarafından işaretlenen bir imzayı sağlamak için girdi (entity) A'ya izin veren dijital imzadır. Bu uzlaşmaya dayanan imza mekanizması üzerinde tahminin önemini göstererek yapılır. Sahteliği(taklit) kanıtlayan Yetenek şifreleme tahminine dayanmaz. Fakat küçük birkaç ihtimalle başarısız olabilir. Bu başarısız ihtimal işleyenin gücünün hesaplanmasından bağımsızdır. Fail-stop imza projesi çok güçlü bir rakibin tek imzayı işleyebilme, Sahteliği(taklit) kontrol edebilme ve işaretlenmiş mekanizmasının daha uzun kullanılamama avantajlarına sahiptir. Bununla beraber fail - then - stop ayrıca uygundur. Fail – stop özelliği aşağıdaki özelliklere sahiptir.

1- eğer bir işaretçi mekanizma ile ilgili mesajı işaretlerse kontrolden sonra imza sorgulayıcıyı kontrolden sonra imza sorgulaycıyı kontrol eder.

2- Bir işleyici çalışmanın üs alma hesabını yapmadan doğrulama(gerçekleme) algoritmasını geçen imzayı oluşturamaz.

3- Eğer işleyici, doğrulama(gerçekleme) testinden geçen imzayı oluşturmada başarılıysa yüksek ihtimalle doğru işaretçi sahteliğin(taklit) ispatını üretebilir.

4- Bir işaretçi birkaç zaman sonra sahteliği(taklit) iddia eden imzayı oluşturamaz.

Algoritma. 5.118, Fail Stop mekanizmasının bir örneğidir. Tarif edildiği gibi tek zaman imza projesidir. Fakat bir ihtimal doğruluk ağaçlarını kullanarak işaretli çarpımlara izin vermek için genelleştiren yollar vardır. Sahteliğin(taklit) ispatı algoritması Algoritma. 5.134 te sunulur.

5.116.Algoritma : Algoritma 5.130'un anahtar üretimi :

Özet: Anahtar üretimi TTP ve girdi (entity) A arasında bölünür.

1. TTP :

a) P ve q asalarını seçer. q (p-1) ve aktif olan Zp içindeki ayrık logaritma problemini böler.

b) Q mertebesi Zp'nin G devirli altgrubu için α yöneticisini seçer.

i) g $\in$ Zp keyfi bir eleman seçer ve α=.... hesaplanır.

ii) α=1 ise i. Adıma gider.

c) $1 \le a \le q-1$ aralığında keyfi a gizli sayısını seçer. ve β=... hesaplanır. a TTP ile gizli tutulur.

d) Girdi(entity) A'ya açıkta (p, q, α, β)gönderir.

2. A girdi(entity):

a) (0, q-1) aralığında x_1, x_2, y_1, y_2 keyfi gizli sayıları seçer.

b) β_1=..... ve β_2=..... hesaplanır.

c) A'nın açık anahatrı (β_1.....) özel anahtarı $\bar{x}$=.... dır.

5.117.Not : (TTP'nin Gizli Bilgisi) Algoritma. 5.128'deki aktif olan Zp'nin içinde q mertebesinin altgrubunda ayrık logaritma problemi tahmin edilir. Sadece a'yı bilen girdi (entity) α tabanlı β ayrık logaritması TTP'dir.

5.118.Algoritma 5.117 'daki Başarılı Taklitlerin Olasılığı:

Rakibin, birkaç m' mesajı üzerinde A'nın imzasını çıkarmak istediğini düşünelim.

i) İşleyişi, sadece işaretçinin açık anahtar girdisi(entity) vardır. rakip tarafından oluşturulan imza ihtimali, m' için A'nın anahtarının sadece $q/q^2=1/9$ olduğu gibi aynıdır. Bu ihtimal rakibin hesaplama kaynaklarından bağımsızdır.

ii) işleyicinin işaretçinin oluşturduğu bir imza (sn, m, sz, m) ve bir m mesajı için bir girdi (entity) vardır. Rakip tarafından oluşturulan imza ihtimali, m için A'nın imzasının sadece 1/q olması gibi aynıdır. Yine bu ihtimal rakibin hesaplama kaynaklarından bağımsızdır. Şimdi bir rakibin bir mesaj üzerinde A'nın imzasını işlediği ve imzanın Algoritma.5.130 daki doğrulama(gerçekleme) bölümünü geçtiğini düşünelim. A'nın imzanın sahteliğini (taklit) kanıtlayabilmesi objektiftir. Aşağıdaki algoritma yüksek ihtimalle A'nın gizli a'yı çıkmak için işlenmiş imzayı nasıl kullanabildiğini gösterir.

5.9. Simetrik-anahtar ve açık-anahtar şifrelemesi

simetrik anahtar ve açık anahtar projeleri, çeşitli avantajlara ve dezavantajlara sahiptir. Bu bölümde, bir önceki bölümlerde özetle gösterilen özellikleri ve bunların sayıları belirtilmektedir.

5.9.1.Simetrik Şifrelemenin Avantajları

1- Simetrik anahtar şifreleri veri oranlarının yüksek olması için tasarlanabilir. Bazı donanım tanımlamaları, yazılım tanımlamaları megabyte'taki oranları elde edebildiğinde megabyte'ın yüzlercesinin şifreleme oranlarını tamamlar.

2.Simetrik anahtar şifreleri için olan anahtarlar nispeten kısadır.

3.Simetrik anahtar şifreleri keyfi sayı jeneratörlerini, hash fonksiyonlarını ve dijital imza projesinin etkili mesajını içeren çeşitli şifreleme mekanizmalarını kurmak için kabaca çalıştırabilir.

4.Simetrik anahtar şifreleri, daha güçlü şifreler meydana getirmek için oluşturabilir. İncelenmesi kolay olan basit çeviriler güçlü şifreleri kurmakta kullanılabilir.

5.Simetrik anahtar şifrelemesi pahalıya malour.

5.9.2.Simetrik–Anahtar Şifrelemesinin Dezavantajları

1- İki-parti haberleşmesinde, anahtar her ikisinin sonunda gizli kalmalıdır.

2- Geniş bir network 'te kullanılan birkaç anahtar çifti vardır. Etkili anahtar idaresi şartsız güvenilen TTP kullanımı gerektirir.

3- Mevcut olan A ve B arasındaki iki-parti haberleşmede ses şifreleme pratiği sürekli değişen anahtarları ve herhalde her bir haberleşme oturumunu yazdırır.

4- Dijital imza mekanizmaları, bir TTP kullanımı veya açık araştırma fonksiyonları için genellikle büyük anahtarlar gerektiren simetrik-anahtar şifrelemesinden çıkmaktadır.

5.9.3.Açık Anahtar Şifrelemesinin Avantajları

1- Sadece özel anahtar gizli tutulmalıdır.

2- Bir network'te anahtarların yönetimi, şartsız güvenilen bir TTP ile karşılaştırılmış gibi sadece işlevsel güvenilen bir TTP' nin hazır bulunmasını gerektirir. Kullanım moduna bağlı olarak; TTP, gerçek zamanda karşılaştırılmış gibi bir "off-line" tarzını gerektirebilir.

3- Kullanım moduna bağlı olarak; özel anahtar/açık anahtar çifti önemli zaman periyotları gibi bazı oturumlar için değişmeden kalabilir.

4- Bazı açık anahtar proje ürünleri dijital imza mekanizmalarına nispeten daha etkilidir. Açık doğrulama(gerçekleme)fonksiyonlarının tarifinde kullanılan anahtar genellikle simetrik-anahtar counterport(?) için daha küçüktür.

5- Geniş bir network'te gerekli anahtar sayıları simetrik anahtar senaryosunda daha küçük olabilir.

5.9.4.Açık Anahtar Şifrelemesinin Dezavantajları

1- En popüler açık-anahtar şifreleme metotlarının oranları, en iyi bilinen simetrik projelerinden daha yavaş olan büyüklük sıralarıdır.

2- Anahtar boyutları genellikle simetrik-anahtar şifrelemesi için gerekenlerden daha büyüktür ve açık-anahtar imza boyutu, simetrik anahtar tekniklerinden veri başlangıç belgesini sağlayan takılardan daha büyüktür.

3.Hiçbir açık anahtar projesinin güvenli olması sağlanmamıştır. En etkili açık anahtar şifreleme projesi, sayı problemlerinin küçük bir kümesinin varsayılan zorluğuna dayanan sahip oldukları kendi güvenliğini buldu.

4.Açık-anahtar şifresi, simetrik-anahtar şifrelemesi kadar geniş bir tarihi yoktur. 1970'lerin ortalarında bulunmuştur.

5.10.Karşılaştırma Özeti

Simetrik ve açık anahtar şifrelemesi tamamlayıcı avantajlarının bir sayısına sahiptir. Bu günkü şifreleme sistemleri herbirinin kuvvetinin sömürür.

Açık anahtar şifreleme teknikleri, bildirilmiş A ve B değerleriyle kullanılabilen simetrik anahtar sistemi için bir anahtar kurulumunda kullanılabilir. Bu senaryoda, A ve B açık anahtar projesinin açık/özel anahtarlarının uzun süre avantajlarını ve simetrik-anahtar projelerinin etkili performansını alabilir. Veri şifrelemesi, şifreleme yöntemlerinin tüketildiği bölümlerde sürekli olduğunda anahtar kurulumu için açık anahtar projesi, A ve B arasında toplam şifreleme yönteminin küçük bir parçasıdır.

Günümüzde, açık anahtar şifrelemesinin hesaplanabilen performansı, simetrik-anahtar şifrelemesi için ikinci derecedir. Yine de böyle olması için bir kanıt yoktur. Pratikte iki önemli nokta şudur;

1- **A**çık anahtar şifrelemesi etkili imzaları ve anahtar yönetimini kolaylaştırır.

2- Simetrik anahtar şifrelemesi şifreleme ve birkaç veri bütünlüğü uygulamaları için etkilidir

Açık anahtar sistemlerindeki özel anahtarlar, simetrik anahtar sitemlerindeki gizli anahtarlardan daha büyüktür. Çünkü simetrik anahtar sistemleri üzerinden en etkili hücum, ayrıntılı anahtar araştırması iken tüm bilinen açık anahtar sistemleri, ayrıntılı araştırmalardan daha etkili "kısa-kesim " hücumlarına bağlıdır. Eşit güvenlik için simetrik anahtarlar, 10 veya daha fazla faktörlerle açık-anahtar sistemlerindeki özel anahtarlardan daha küçük bit uzunluklarına sahiptir.

6.BÖLÜM

6. KRİPTOGRAFİ

6.1. Kriptografinin Tanımı

Kriptografi bilgi güvenliğini inceleyen bilim dalıdır. Güvenilirlik, veri bütünlüğü, kimlik doğrulama gibi bilgi güvenliği konularıyla ilgilenen matematiksel yöntemler üzere yapılan çalışmalar olarak da tanımlanabilir. Kriptografi kelimesi, Yunanca gizli anlamına gelen "kryptos" kelimesinden türemiştir.

Modern kriptografinin ilgilendiği ana konular:
- **Gizlilik;** Bilgi istenmeyen kişiler tarafından anlaşılamaz
- **Bütünlük;** Bilgi saklanması veya iletilmesi sırasında, farkına varılmadan değiştirilemez
- **Reddedilemezlik;** Bilgiyi oluşturan ya da gönderen, daha sonra bilgiyi kendisinin oluşturduğunu veya gönderdiğini inkar edemez
- **Kimlik belirleme ;** Gönderen ve alıcı, birbirlerinin kimliklerini doğrulayabilirler

"Bilgi güvenliği" başkası tarafından dinlenme, bilginin değiştirilmesi, kimlik taklidi gibi tehditlerin ortadan kaldırılması ile sağlanır ve bu amaçla kullanılan temel araç kriptografidir.

Bilgi toplumlarına bakıldığında, teknolojinin de yardımıyla, milyonlarca insanın gözetiminin hükümetler tarafından yapılmakta olduğunu görürüz. Kriptografi dijital dünyadaki kişilere özelliği sağlayan başlıca araçtır. Kriptografi artık sadece resmi özelliğe sahip olan veya uzak durulması gereken askeri bir araç değildir. Kriptografiyi öğrenmek ve onun modern toplumlara sağladığı avantajlardan yararlanmak hem kişisel gizliliğimiz hem de elektronik dünyadaki güvenliğimiz için kaçınılmazdır.

6.1.1. Temel Terimler

Kriptografik teknolojide bir bilginin içeriğinin başkalarının anlamayacağı hale getirilmesine **şifreleme** denir. Bu bilgi bir yere iletilmek amacıyla şifrelenen bir mesaj veya saklanmak amacıyla şifrelenen bir bilgi olabilir. Şifreleme işlemi, birbirinden farklı matematiksel özelliklere sahip olan farklı **kriptografik algoritmalardan** biri kullanılarak

yapılır. Şifreli mesajdan ana mesajı elde etme işlemine **şifre çözme** denir. Şifreleme ve şifre çözme bir veya iki farklı anahtarın yardımıyla yapılır ve anahtar olmadan bir mesajı şifrelemek veya şifresini çözmek mümkün değildir. Burada sözü edilen **anahtar**, kriptografi algoritmasının kullandığı uzun sayı dizisi anlamına gelmektedir. **SSL** (**S**ecure **S**ocket **L**ayer) şifreli bilginin TCP/IP üzerinden aktarılmasını sağlayan bir protokoldür.

6.2. Ulusal Bilgi Güvenliği Raporu ve Teknolojik Gelişmeler

Bilgi ve iletişim teknolojilerindeki hızlı gelişmeler yeni bir çağ yaratmıştır. Bilgi çağı olarak adlandırılan bu çağda ekonomide ve sosyal yaşamda klasik paradigmalar yetersiz kalmakta; teknolojik gelişmeler yeni yapılar, yaklaşımlar yaratmaktadır. Bu nedenle, bilgi güvenliğine ilişkin ulusal bir politika oluşturmanın temel koşullarından birisi, bilgi ve iletişim teknolojilerinde gözlenen gelişmelerin bilinmesidir. Bu noktada, söz konusu teknolojik gelişmelerin ne olduğunu ve ne yönde olacağını doğru anlamak ve içeriğini doğru belirlemek son derece önemlidir:

Kriptoloji, İnternet'in yaygınlaşması ve bir ticaret medyası halini almaya başlamasıyla, bilgi güvenliğinin sivil uygulamalarına tanık olmaya başlamıştır. Bu uygulamalar kısaca "açık anahtarlı altyapılar **PKI** (**P**ublic **K**ey **I**nfrastructure)" olarak isimlendirilebilir. Açık anahtarlı altyapılar, bir gizli ve açık anahtar çifti ve bu çiftle sağlanan "elektronik kimlik, sayısal imzalama ve şifreleme" işlevleriyle, gerekli kurumsal ve yasal yapılanma üzerine kurulmuştur. Burada şu noktalar önemlidir:

Kripto işlemleri donanım üzerinden yazılıma kaymıştır. Örneğin kripto teçhizatı, saklanması, imhası ve benzeri unsurlar bu altyapıda karşılaşılan ve kullanılan terimler değildir.

Söz konusu yazılımlarda kullanılan algoritmalar tümüyle kamuya açıktır. Bunun anlamı, bu algoritmaların çok sayıda taraf tarafından testinin yapılabilir olması ve bu yolla güvenilirliğinin artmasıdır.

Anahtarların üretilmesi, saklanması ve tüm işlevlerin yürütülmesi için uluslararası açık standartlar belirlenmektedir (**X.509** elektronik kimlik belgesi standardı, **PKCS** açık anahtarlı altyapılar standartları gibi). Bu standartların dışına çıkmak pratik değildir.

Açık anahtarlı altyapıların gerektireceği yasal düzenlemeler de, (onay kurumlarının yapılandırılması, sayısal imzanın kabulü, vb.) uluslararası uyumun gereği olarak yerine getirilmelidir.

Konu ile ilgili ilk yasa ABD'de Utah eyaletinde kabul edilmiştir (1995). Bunu Almanya (1997) ve Singapur (1998) sayısal imza yasaları izlemiştir. Halen Avrupa Topluluğu üye ülkelerinde böyle bir yasa için çalışmalar sürmektedir. Genel eğilimlerin dışında geliştirilmeye çalışılacak uygulamalar sosyal ve ekonomik açıdan zararlı olabilecek, uluslararası standartlara uymayan yapılanmalar ülkeleri yalnızlığa itecektir.

Internet yalnızca bir iletişim altyapısı değildir. Kimi yorumlara göre bilginin yaratılması ve paylaşılması için bir "özgürlük ortamı" anlamına da gelmektedir. Gerçekten de Internet sınırsız bilginin yayılmasının medyası olmuştur. Bu bağlamda, kriptografik ürünler hızla ve kolaylıkla yayılabilmektedir. Bu durum, ABD'deki aksi yönde gayretlere karşın değişmemiş aksine gelişmiştir. Örneğin, ABD tarafından sınırlandırılmış pek çok ürünü, ABD üzerinden ya da dünyanın başka bir köşesinden Internet aracılığıyla edinmek "teknik olarak" son derece kolaydır. "Teknik yöntemlerle" bunun önüne geçilmesi olanaksızdır. Dolayısıyla, "İnternet'teki bu durumun denetlenmesi" amacıyla yapılacak girişimler "teknik açıdan geçersizdir".

Bunun yanında şu noktalar da dikkat çekicidir:

Internet protokolü IP'nin gelecekteki versiyonu IP v6, IP düzeyinde "şifreleme" sağlayacaktır. Bu durumda anahtarların saklanması neredeyse imkansız olacaktır.

Bilgi ve iletişim teknolojilerinde, ulusal devlet politikaları "teknolojik yansızlık" ilkesini benimsemişlerdir. Bunun anlamı kamunun, gelişme aşamasında bir teknolojiyi diğerlerine üstün saymaması ya da tercih etmemesidir. Diğer bir deyişle, teknolojilerin açık rekabet ortamında birbirlerine üstünlüklerinin sağlanmasıdır.

Bütün bu teknolojik gelişmeler göz önüne alındığında oluşturulacak politika ilkelerinin ve kurumsal yapılanma önerilerinin, ileride sorunlar yaratabilecek uygulamalardan kaçınılması gerektiğini göstermektedir. Yukarıda sözü edilen teknolojik tarafsızlık ilkesi, teknolojik açık rekabet ortamının sağlanması gibi konular göz önüne alındığında oluşturulacak politikaların bunlarla çatışmaması gerekmektedir. IP v6 ve sonrası gelişmelerin

ışığında, teknik açıdan gerçekleştirilemeyecek görevler, kurumsal önerilerde yer almamalıdır. Bu nedenle, gelişmiş ülkeler bu yeni gelişmelere karşı tedbirler almak konusunda çok temkinli davranmaktadırlar. Mevcut yasalarla sahip oldukları yetkileri de, kulanım alanlarına açıklık getirerek sınırlamaktadırlar.

6.2.1. Diğer Ülkelerdeki Durum

Birkaç yüzyıllık geçmişi olan demokrasinin, çağımızda ortaya çıkan çok yoğun bilgi transferi ihtiyacına ayak uydurmasını sağlayacak arayışlar sürmektedir. Demokratik ülkeler, kriptografik yazılım ve donanım kullanımı söz konusu olduğunda kişisel/ticari özgürlüklerle ve devlet/kamu güvenliği arasında bir denge bulmaya çalışmaktadırlar. Bilgi güvenliği, uzun yıllar boyunca ve özellikle soğuk savaş döneminde, askeri ve diplomatik haberleşmenin önemli bir parçası olarak ele alınmıştır. Bu açıdan bakıldığında kavram, bilginin güvenli iletimi kadar, "hasım ulusların" elektronik istihbarat yöntemleriyle dinlenmesi olarak anlaşılmıştır. Özellikle gelişmiş uluslar, bu amaçlarla soğuk savaş döneminin hemen başlarında çeşitli kurumlar ihdas etmişlerdir.

Gelişmiş ülkelerdeki gelişmeler aşağıda özetlenmiştir.

6.2.1.1. ABD

Kişisel özgürlüklerin zedelenmemesi prensibinin neredeyse anane haline geldiği bu ülkede, devlet, yurttaşlarının haberleşmenin mahremiyeti konusunda hakları ile terörizm, kaçakçılık ve devlete karşı işlenen diğer suçların önlenmesi konusunda dengeleri de kurmuş bulunmaktadır. ABD'de 1952 yılında kurulan "Ulusal Güvenlik Teşkilatı (**National Security Agency**)" bu dengenin içinde kendine özgü bir yere sahiptir. **NSA**, ABD çıkarları doğrultusunda bir yanda uluslararası elektronik istihbarat yapmak ve öte yanda Amerikan devletinin bilgi güvenliğini sağlamaktan sorumludur. NSA, uzman teknik fonksiyonları sağlamaktan sorumlu kılınan Savunma Bakanı'nın yetki, kontrol ve yönlendirmesinde ve Savunma Bakanlığı bünyesinde bağımsız bir teşkilat olarak kurulmuştur.

NSA'nın günümüzde aldığı biçim hakkında şu noktalar önemlidir:

NSA, tam olarak bir "elektronik istihbarat" örgütüdür. Hatta bu öyledir ki; NSA "ABD İstihbarat Topluluğu (**US Intelligence Community**)" içinde, **CIA**, **FBI**, "Ordu İstihbarat (**Army Intelligence**)" ve Savunma Bakanlığı gibi toplam 13 federal kurumdan birisidir ve bu kuruluşundan beri değişmemiştir. Ayrıca, 1972 yılında kurulan "Merkezi Güvenlik Birimiyle (**Central Security Service**)" NSA ve ordu istihbarat birimleri arasında tam bir işbirliği

sağlanarak Savunma Bakanlığının kriptografik çalışmaları tek bir bünyede verilmeye başlanmıştır.

NSA "diğer uluslar ve onların tarafları için istihbarat ve karşı istihbarat" in istihbarat etkinlikleriyle sınırlıdır. Amerika'da oturma izni olan yabancılar, Amerikan vatandaşları ve Amerikan özel sektör kurumlarının gizlilik haklarına aykırı yasadışı istihbarat yürütmesi anayasa ve NSA'in kuruluş yasasıyla kesinlikle yasaklanmıştır. Bunun ötesinde, gizlilik ve haberleşme özgürlüğü yasalarıyla kişiler kendileri hakkında NSA gizlilik yasasında tanımlana "kimlik bilgilerine" erişme hakkına sahiptirler. Haberleşme özgürlüğü kapsamına giren "hükümet kayıtlarının" NSA tarafından tutulması yasayla engellenmiştir.

NSA "ithalat ve ihracat politikalarının" belirlenmesinde görevli değildir ve kendi dışında oluşan politikalara tabiidir. ABD'de bu politikalar "**Başkanlık**" düzeyinde saptanır.

NSA'in ABD'deki "kriptografi üretimini" kontrol altında tutmak, söz konusu ürünleri sertifikalandırmak türünden hiç bir işlevi ve görevi yoktur. Hatta, özel üreticiler kamuya ürünlerini satarlarken dahi, NSA'in onayını almak zorunda değildirler. Tam aksine, NSA, gelişmiş yeteneklerinden ABD özel sektörünün de yararlanmasını sağlamak amacıyla, özel sektörün isteğine bağlı olarak danışmanlık vermektedir.

Kamuda bilgi güvenliği standartlarının belirlenmesi ve uygulanması da NSA'in görevleri arasında değildir. ABD'de bu işlevi, Amerika Standartlar Enstitüsü (**National Institute of Standarts and Technology**) "Federal Bilgi İşleme Standartları (**Federal Information Processing Standarts**)" yayınlarıyla yerine getirmektedir. Bu bağlamda Madde 8'in çeşitli bentlerinde ve farklı biçimlerde tekrar edilen "usuller ve yöntemler belirlemek, altyapıları korumak" fiilleriyle ifade edilen ve temelde kamuda bilgi güvenliği standartlarının altyapılar, ürünler ve uygulamalar açısından saptanmasına dönük olduğu anlaşılan işlevler bu kapsama girmektedir.

1993 yılı Nisan ayında Clinton yönetimi NSA tarafından hazırlanan/önerilen yeni bir kriptoloji politikasını ortaya koymuştur. Başkan Bush zamanından başlayarak yürütülen bu çalışmaların odak noktası hükümet tarafından geliştirilen Clipper adlı bir kripto çipidir. Özel sektör tarafından üretilen her türlü güvenli iletişim ürünlerine yerleştirilmesi önerilen bu çipe karşı çok büyük bir tepki oluşmuştur. Bu tepkilerin kaynağında, her çip için özel olarak

üretilen anahtarların bir kopyasının hükümet tarafından tutulması ve tamamen yasal olmayan hiçbir nedene dayanarak bu çipler üzerinden geçen trafiğin dinlenmeyeceğinin hükümet tarafından garanti edilmesine rağmen, yeterli güvenin oluşturulamaması bulunmaktadır. Ayrıca, hükümet tarafından geliştirilen anahtar algoritmasının açıklanmaması sonucunda algortimanın denenmesinin ve güvenilirliğinin kanıtlanamayacak olması, bunun sonucunda tüketicilerin bu ürünlere rağbet etmeyeceğinin ortaya çıkması, gittikçe önem kazanan kriptoloji sanayiinde dünya pazarlarında rekabet şansının kapalı bir algoritma kullanılarak üretilen ürünlere dayanarak korunamayacağı gibi endişeler ortaya çıkmış ve bu çip istenen kullanım yaygınlığına ulaşamamıştır.

Öte yandan "yaşamsal altyapıların korunması" (**critical infrastructure protection**) yeni bir kavram olarak dünya ülkelerinin gündemine girmiştir. Yaşamsal altyapıların korunması kavramı, ekonominin ve devletin minimum düzeyde işleyişi için gerekli fiziksel ve ağsal sistemleri kapsamaktadır. Amaç, ülkenin düşmanlarının, bunlar ister başka ülkeler, ister ülke içindeki gruplar ve bireysel olsun, "geleneksel olmayan" yöntemlerle yapacakları saldırıların engellenmesidir. Açıktır ki, bu tehditler, devletin elektronikleşmesi ve açık ağları kullanmasının yaygınlığıyla doğru orantılıdır. Yaşamsal altyapıların korunması için ABD Başkanı Bill Clinton, 1998 yılında bir Beyaz Belge direktifleri yayınlamıştır. Bu belgede dikkat çeken unsurlar şunlardır:
Devlet içinde öncü kurumlar tespit edilmiştir. Her öncü kurum çeşitli sektörleri paylaşmışlardır. Örneğin, Ticaret Bakanlığı iletişim ve enformasyon sektörüne; Hazine Bakanlığı bankacılık ve finans sektörüne, FBI polis, acil durum ve adalet konularına; CIA dış istihbarata; Dışişleri Bakanlığı dışişleri sektörüne; Savunma Bakanlığı savunma sektörüne liderlik edecek kurumlar olarak belirlenmişlerdir. Ayrıcı, Bilim ve Teknoloji Politika Genel Müdürlüğü, Ulusal Bilim ve Teknoloji Konseyi'nin programları aracılığıyla araştırma ve geliştirme çalışmalarını eşgüdümlemekle görevlendirilmiştir. Öncü kurumların seçilmesinin nedeni, bu konuda, özel sektör/kamu sektörü işbirliğini gerektirmesi ve gereksiz hükümet düzenlemeleri yaratmaktan kaçınılmasıdır.

Ulusal Eşgüdümcü: Güvenlik, Altyapı Koruma ve Karşı-terörizm Ulusal Koordinatörü bu direktifin eşgüdümünden sorumludur.

Uyarma ve Bilgi Merkezleri: Başkan, bu görevle FBI'yı sorumlu kılmıştır.

National **I**nfrastructure **P**rotection **C**enter **(NIIPC)**: Bu heyeti, FBI, bilgisayar suç uzmanları, Savunma Bakanlığı, İstihbarat topluluğu ve önder kurumların temsilcilerinden oluşur.

6.2.1.2. İNGİLTERE ve ALMANYA

NSA dışında gelişmiş ülkelerden anılan iki diğer örnek kurum İngiltere'deki Kamu Haberleşmesi Koordinasyonu (**Government Communications Headquarters**) ve Almanya'daki Enformasyon Teknolojileri Güvenlik Kurumudur (**Bundesamt für Sicherheit in der Informationstechnik**). İngiltere'deki Kamu Haberleşmesi Koordinasyonu (**Government Communications Headquarters**), NSA'ya çok yakın işlevler yürütmektedir. **GCHQ**'nun da oluşumu soğuk savaş dönemine dayandırılmaktadır.
Enformasyon Teknolojileri Güvenlik Kurumu (**BSI**) ise **NSA** ve **GCHQ** örneklerinden farklı olarak, istihbarat işlevi olmayan bir kurumdur. BSI bir Alman kamu kurumu olarak, bilgi ve bilgisayar sistemleri güvenliği konularında araştırma yürüten bir kurumdur. Araştırma sonuçları, kamuda söz konusu güvenlik uygulamalarının yapılmasına yarar sağlamaya çalışmaktadır. Kurum adli olaylarda da emniyete talep olduğu takdirde danışmanlık hizmeti verebilmektedir.

Alman Hükümeti de söz konusu teknolojiyi yasaklamak üzere girişimlerde bulunmaya başlamıştır. 4 Mayıs 1995 tarihinde Alman Parlamentosu "Telekomünikasyon İzleme Kanunu" adı altında ülkede tasarlanan ve kullanılan telefon, GSM, ISDN ve bilgisayar şebekesi tarayıcılarının, devlet birimleri tarafından gerektiğinde dinlenmesini sağlamak için standart bir ara bağlantı sağlamaları konusunda bir kanun teklifini onaylamıştır. Kanunun özünü, güçlü delillere dayanarak bağımsız bir yargıç kararı olmadan özel haberleşmenin dinlenememesi oluşturmaktadır. Yargıcın gerekli gördüğü durumlarda, bu kanuna göre çağrı oluşturma bilgilerine ve GSM kullanıcılarının hücreler arasında izlenmesini sağlayacak bilgilere erişilmesi mümkün hale gelmektedir. Almanya, bu düzenlemesiyle, Avrupa ülkeleri arasında bireyi devlete karşı üst düzeyde güvenceye alan bir ülkelerin öncülüğünü yapmaktadır.

6.2.1.3. NORVEÇ

Norveç'te kriptografik ürünlerin yurt içerisinde kullanımında herhangi bir yasaklama yoktur ve ithalatında da kontrol bulunmamaktadır.

Norveç Wassenaar anlaşmasının bir üyesidir ve bu bağlamda kriptografik ürünlerin ihracatını 1987 tarihli bir yasayla kontrol etmektedir. İhracat kontrolü Dış İşleri Bakanlığı'nın sorumluluğundadır. Norveç yasaları Wassenaar antlaşmasında yer alan genel yazılım istisnasını gözetmektedir. Ayrıca, Internet'in genel yazılım notunun uygulanması ve kriptografik ürünlerin iletilmesi için yeterli bir temeli teşkil ettiği benimsenmektedir.

Norveç'te yürütülmekte olan Kamu Sektörü Ağ Projesinde, sayısal imza sayısal kimlik belgesi ürünlerinin oluşturulması hedeflenmektedir. Bu projenin "açık anahtarlı altyapılar" (**PKI**) için yasal, teknik ve organizasyonel düzenlemeler için deneme olduğu ve temel teşkil edeceği planlanmaktadır.

SEIS-Secure **E**lectronic **I**nformation in the **S**ociety, kamu ve özel sektör örgütlerinin ortaklaşa çalıştığı bir gruptur. Elektronik kimlik kartları için kamuya açık standart geliştirmişlerdir. (**PAS**) Norveç ulusal standartlar kurumu **SEIS** temelli İsveç standardını Norveç ulusal standardı olarak kabul etmeyi planlamaktadır.

Norveç, bir dizi şifreleme sisteminden oluşan ve çalınamaz biçimde silisyuma yazdığı NSK adlı milli algoritma yazmıştır. **NSK** (**N**orwegian **S**tandart for **C**ryptography) algoritmasını kullanan NX1000 gibi kriptografik cihazlar Norveç pazarında bulunabilmektedir.

ABD'de kriptografik teçhizat Wassenauer ilkeleri gereğinde "mühimmat" gibi görmektedir. "Mühimmat kavramı" askeri bir terim olarak, kriptografik ürünlerin, 1998 Wassenaar düzenlemesinde, "askeri ve ticari olmak üzere çift kullanımlı" teknolojilerden sayılmaya başlanmasıyla ilgilidir. Bunun anlamı, bu düzenlemeye imza koyan ülkelerin, kriptografi ürünlerinin ithalat ve ihracında "uyumlu" politikalar izleme niyetinde olduklarıdır. Ancak Wassenaar bir anlaşma değildir ve yaptırımı yoktur. Nitekim, İsviçre hükümeti 1998 düzenlemesinin "liberal politikalarında" bir değişikliğe neden olmayacağını ve İsviçre firmalarının dünya pazarından en yüksek payı almaları konusundaki desteğinin devam edeceğini açıklamıştır (Bkz. Crytography and Liberty 1999, EPIC). Wassenaar üyesi İrlanda ise, en başından beri ABD ve İngiltere'nin aksi yönünde politikalar izlemiştir. İrlanda'nın bir firması olan "Baltimore Inc." bugün dünyanın en önde gelen bilgi güvenliği firmalarından birisidir.

ABD, 1998 Wassenaar düzenlemesinin aksi yönünde, ihraç politikalarını 2000 yılıyla beraber değiştirmiştir. Wassenaar 1998 açık bir biçimde Internet tarayıcılarında 56 bit anahtar uzunluğunda kriptoya izin verirken, ABD bu yılın başından itibaren 128 bit anahtar

uzunluğunu ihraç etmeye başlamıştır. ABD kökenli web sunucuları da paralel bir biçimde 128 bit SSL destekler bir biçimde ihraç edilmeye başlanmıştır. Ülkemizde bazı bankaların internet sayfalarında bu konunun uygulaması görülmektedir.
Bu değişimde, "ticari çıkarların" ağır bastığı vurgulanmaktadır. Değişiklik, tümüyle liberal bir politika değilse de, geçmiş uygulamalara göre liberalleşme anlamına gelmektedir.

Fransa'da da kriptografik yazılım ve donanım "mühimmat" olarak tanımlamakta ve işlem görmektedir. Yasa ile kriptografik teçhizatın ihracatı ve kullanımı devlet denetimine tabi kılınmıştı. Bir süre, Fransa'da faaliyet gösteren yabancı şirketler "ulusal güvenlik" nedeni ile kullandıkları anahtarları Fransız hükümetine bildirmek zorunda kaldılar. Ancak daha sonra bu politikalardan vazgeçilmiştir. Almanya, Finlandiya ve İtalya en başından beri liberal politikalar, izlerken, Fransa 1999 yılında gösterdiği değişimle ilginç bir uyum örneği sergileşmiştir. Fransa'da da artık sadece "beyan" yöntemi uygulanmaktadır. Gerçekten de Fransa yakın zamana kadar, "sertifikalandırma" yöntemiyle ihracat ve ithalatı kontrol altında tutmaya çalışırken, 1999 yılı başında Başbakan'ın birinci ağızdan açıklamalarıyla sadece ticari politikalarında değil ancak tüm ulusal politikalarında önemli değişikliklere gittiğini duyurmuştur.

Avrupa Birliği, diğer ulusal politika konularında olduğu gibi, ihracat ve ithalat izinlerinde de ABD'ye oranla çok daha liberal politikalar izlemektedir. Son olarak, 22 Mayıs 2000 tarihinde Lizbon'daki Dışişleri Bakanları toplantısında onaylanması beklenen bir düzenlemeyle, kriptolojik ürünlerin ihracatına getirilen sınırlamalar büyük ölçüde kaldırılacaktır. Bu düzenlemeyle, ihracat yapacak firmaların, ürünün son kullanıcısının Avrupa Birliği ülkelerinde veya aralarında Kanada, Japonya, ABD, Avustralya ve Yeni Zelanda'nın da bulunduğu 10 ülkeden birinde olduğunu beyan etmeleri yeterli olacaktır. Avrupa Birliği ve adı geçen diğer ülkeler bu alanda dünya pazarının %80'ini oluşturmaktadırlar.

Bu hamleyle, AB ülkeleri firmaları bu pazarda özellikle ABD'li firmalara karşı büyük avantaj sağlamış olacaktır. Bu nedenle, çok yakında ABD'li firmaların hükümete baskı yaparak kendi ülkelerinde de benzer bir düzenlemenin yapılmasını istemeleri beklenmektedir.

OECD'nin 1998 yılında gerçekleştirdiği "Kriptografi Teknolojilerindeki Kontroller" başlıklı envanter çalışmasının raporu 1999 yılında yayınlanmıştır. Bu envantere göre

kriptografik ürünlerin ihracat ve ithalatından sorumlu kurumlar büyük çoğunlukla ekonomiden ya da sanayi ve ticaretten sorumlu bakanlıklardır. Envantere göre yalnızca Avustralya ve Türkiye'de sorumlu bakanlıklar olarak Savunma Bakanlıkları belirtilmiştir. Türkiye'de ayrıca Dış Ticaret Müsteşarlığı'da sorumlu kurum olarak belirtilmektedir.

6.2.1.4. TÜRKİYE' deki Durum

Ulusal Güvenliği ilgilendiren bilginin örgütlenmesi açısından,. "gizlilik dereceli" bilginin ne olduğunu nasıl üretileceği, korunacağı, nakledileceği, kullanılacağı ve imha edileceği konusunda T.C. Devleti'nin yasal hazırlığı ve düzenlemesi bulunmamaktadır. Bu yasal düzenlemeler ise geçici olarak kurulup sonra dağıtılan birtakım platformların üstesinden gelebileceği gibi basit bir yasa ya da mevzuat değil çok karmaşık bir yasalar zinciridir ve gelişen teknoloji nedeni ile sürekli güncellenmek ve teknolojiye uygun hale getirilmek zorundadır.

Ulusal bir bilgi güvenliği politikanın olmayışı ülkenin ulusal güvenliğini hassas hale getirmektedir.

Bakanlıklar, kamu kurum ve kuruluşları arasında ulusal güvenlik ihtiyaçları doğrultusunda bilgi güvenliğini koordine edecek, yönlendirecek ve ulusal bilgi güvenlik sistemini işletecek bir yapı yoktur.

Ulusal bilgi güvenliği gibi karmaşık ve konunun nasıl çözülebileceği hususunun müzakere etmek için ortak anlayışı kolaylaştıracak, üzerinde mutabakata varılmış tanımlar mevcut değildir.

Hassas bilgi altyapısına bağımlılık ve bu altyapıya tehdit ve riskler, kurum ve kuruluşlarca iyi anlaşılamamıştır.

Kriptografik ürünlerin ithalatı ve ihracatı ile ilgili kontrol esasları net olarak belirlenmemiş olup kontrol mekanizması tam olarak tesis edilememiştir.

6.2.2. Politika Açısından Kriptoloji

Kamu kurum ve kuruluşlarında ulusal kullanım amaçlı onaysız hiçbir kripto cihazının kullanılmaması,

Ulusal kriptoloji politikasının, icra ve yasama organları tarafından geliştirilmesi,

Kriptoloji üzerindeki icraat kontrolünün, dış ticareti arttıracak ancak ulusal savunma gereklerini dikkate alacak biçimde tespit ve uygulanması,

Devletin, özel sektörde de, gelecek talep üzerine bilgi güvenliğini geliştirecek mekanizmalarının kurulmasını teşvik etmesi ve desteklemesi,

Haberleşme ve bilgi sistemleri ile kripto teçhizatının milli imkanlarla yurtiçi kaynaklardan sağlanması,

Bilgi teknolojilerinin gelişimi için Araştırma-Geliştirmenin devletçe desteklenmesi gerekmektedir.

6.2.3. Ülkelerin Kriptografi Politikalarındaki Farklılıklar

Dünya üzerindeki değişik ülkelerin kriptografi politikaları oldukça önemli farklılıklar göstermektedir. ABD, kriptografik ürünlerin dış satımına kısıtlamalar koymakta, örneğin çok yaygın olarak kullanılan bir açık(çift) anahtarlı şifreleme yöntemi olan RSA algoritmasının anahtarı olan n sayısının 512 ikili'yi aşmasına izin vermemektedir. Ülke içinde 1024 ikili uygulamaları yasaklamadığı halde, bu durumdan endişe duymakta ve sade vatandaşla birlikte, yasadışı örgütlerin de çok önemli bir gizli iletişim gücü ele geçirmesini sakıncalı bulmaktadır. ABD'nin bu soruna bulduğu çözüm, güvenilir üçüncü kuruluşlar, yani GÜK'ler yardımıyla, herkesin gizli (özel) anahtarına, yasalar gerektirdiği zaman ulaşılmasını sağlayacak yöntemler geliştirilmesidir. Fransa, İsrail, Belçika, Çin gibi ülkeler de, kriptografik ürünlerin dışalımını kısıtlamış, ve gizli anahtarların GÜK'ler tarafından saklanmasını zorlayıcı önlemler almışlardır. İngiltere'deki kriptografi politikaları da benzer endişelerle planlanmaktadır. En uç önlem ise bir Güneydoğu Asya ülkesinden gelmiş ve Birmanya, 1996'nın Eylül ayında Internet bağlantılarını yasaklamıştır.

Öte yandan birçok Avrupa ülkesi ve Avustralya, Japonya gibi ülkeler, kısıtlamalar ve yasaklamalardan yana değildirler. Özellikle Avrupa Birliği ülkeleri, ikili gizliliği tehdit eden her türlü önlemin, insan haklarına aykırı olduğunu ve elektronik ticaretin serbest gelişimini engelleyeceğini ileri sürmektedirler. Japonya, kriptografiye öncelikle ekonomiyi canlandırması açısından bakmakta, ulusal güvenlik yönünden kriptografiyi bir tehdit olarak algılamamaktadır. Bu ülkelerden, GÜK'ler konusundaki politikası belli olmayan Japonya dışındakiler, GÜK'lerin gerekliliğine inanmamakta, üçüncü kuruluşlara güvenilmesi gereksinimi yaratan bir toplumsal düzenlemenin, sonunda GÜK'lerin güvenilirliğini zorlayacak, sarsacak yöntemler gelişmesine yol açabileceğini düşünmektedirler. Bu durum ise yasalara uyan vatandaşların haklarını zedeleyecek, sayısal imzaların taklit edilebilmesine, gizli mesajların istenmeyen kişiler tarafından okunmasına, ekonominin zarar görmesine yol açabilecek ve E-ticaretin gelişmesini engelleyebilecektir.

6.2.4. Kriptoloji Hakkında Değerlendirme Sonuçları

Kriptolojik ürün geliştirme tek başına bir sanayi olarak görülmelidir. Bu konuda uygulanacak akıllı politikalarla bu tür ürünler ülkemiz için önemli bir ticari meta haline gelebilir. Ayrıca birkaç yıl içinde geliştirilen her yazılım için o yazılıma özgü kriptolojik bir modül olabileceği gibi, Internet sayfalarındaki erişim de kriptolojij yöntemler içerebilecektir. Bütün bu muhtemel gelişmeler kriptolojinin artık bir ticari ürün olduğunu ortaya koymaktadır. Bu tür ürünlerin geliştirilmesine konulacak kısıtlar ülkemizdeki herhangi bir ticari ürünün geliştirilmesine karşı konulmuş kısıt gibi, ticari iş hacmini engelleyecektir.

Ayrıca, konu uzmanlarından London School of Economics akademisyenlerinden Stuart J.D. Schwartzstein tarafından da belirtildiği gibi, gizli anahtarın elde edilmesi, ve güvenilir üçüncü tarafta tutulması gibi çalışmaların önümüzdeki dönemde politik, ekonomik ve teknik nedenlerden dolayı başarılı olması beklenmemektedir.

Kanunun bu yanları da düşünüldüğünde, ulusal politikalar açısından yeni kurumsal ve yasal yapılanmanın aşağıdaki ilkeleri de göz önüne alması gerekmektedir:

"Ulusal bilgi güvenliğini sağlamaya ilişkin bilginin" tanımı belirli olmalıdır. Tersi durumda, belirsiz tanım altında, kamu ve özel kurumlarda bilgiye "Ulusal Bilgi Güvenliği Teşkilatı" tarafından erişme hakkı tanınmış olur ve gereğini yapmayan kamu ve özel kurum sorumlularına hapis cezası öngörülebilir. Bu durumun yukarıda özetlenen eğilimlerle ve dünyadaki durumla taban tabana zıt olduğu açıktır.

"Anahtarlama materyalinin üretim esaslarını" hangi durumlar ve koşullar altında belirlenmesinin geçerli olduğu açık olmalıdır. Burada "kişilerin gizli anahtarlarının" yasa kapsamı dışında olduğu açıkça belirtilmelidir.
Kamu ve özel kurum üreticilerini sertifikalandırma yoluna gidilmemelidir. Daha önceden de sözü edildiği gibi, sertifikalandırma ihracat ve kimi yerlerde de kamuya satımlarda uygulanan bir yöntemdir. Yine açıklandığı gibi, bu uygulamalar yok olmaya başlamıştır ve sadece ihracat ve ithalat için "beyan usulüne" geçilmiştir. Bu maddede sadece kamu alımları öngörüyorsa bu açıkça ifade edilmelidir. O durumda dahi, aşağıda ifade edilecek sakıncalardan uzak durulması gereği göz önünde bulundurulmalıdır.

"Ulusal Bilgi Güvenliği Teşkilatının" işlevleri ve ilgi alanı netleştirilmelidir. Bu teşkilatın aynı anda bir istihbarat örgütü, bir araştırma-geliştirme kurumu, bir standartlar enstitüsü, bir ürün sertifikalandırma kurumu, bir kamu düzenleme kurumu olmadığı açıkça belirtilmelidir.

Bu bağlamda "ulusal bilgi güvenliği", "ulusal güvenliği ilgilendiren bilgi" tanımları belirsizlikten kurtarılmalıdır. Bu açıdan bakıldığında Devlet Haberleşme ve Bilgi Teşkilatı altında örgütlenmek daha uygun olabilir.

Kurum yukarıda sayılan işlevlerden hangisine çevrilirse çevrilsin, özel kişi, kurum ve kuruluşlar kapsam dışında tutulmalıdır. "Özel kişi ve kurumlarla" ilgili hükümler koyulmamalıdır.

Yasa haksız rekabet ve tekelci uygulamalara yol açabilecek, teknolojik açık rekabet ortamını zedeleyecek bir yapılanmadan arındırılmalıdır. Buna göre, özel sektörün üretiminin denetimi söz konusu olmamalıdır. Kamu alımlarında, teknolojik tercih dayatmasına gidilemeyeceği açıkça beyan edilmelidir.

Ticari politikalar, kişisel ve ticari gizlilik hakları tümüyle ulusal politikalar düzeyinde ele alınabilecek konulardır. Bu bağlamda kararlar "siyasi otoriteye" bırakılmalıdırlar. Kurumsal yapı, ancak talep olduğunda danışmanlık veren bir kurum statüsünde olmalıdır. Yasanın bu konudaki ilgili maddeleri değiştirilmelidir.

Bilgi güvenliğinin sivil uygulamaları dünyada henüz çok yenidir. Ülkemizde de bilgi teknolojilerinin gelişmişlik düzeyi üretim, tüketim ve içerik açısından gelişmiş ülkelerin çok altındadır. Bu iki gerçek, bilgi güvenliğinin sivil uygulamalarında bir düzenlemeye gitmek için henüz çok erken olduğunu göstermektedir. Bu bir yana, bu düzenlemeleri yukarıda da ifade edilmeye çalışıldığı gibi kapsamı ve işlevleri belirsiz bir kurum marifetiyle yapmaya çalışmak, ekonomik ve toplumsal yaşamımızı önemli ölçüde kısıtlar altına sokacaktır. Bugünden öngörülemeyecek ölçüde zararlar verebilir. Ülkemizde bilgi teknolojilerinin üretimini, tüketimini ve içeriğini zenginleştirmek üzere yapılanmalara gitmek, elektronik belge ve sayısal imzalar gibi gerekli öncül yasaların çıkarılmasını sağlamak çok daha acil sorunlar olarak karşımıza çıkmaktadır. Bu bağlamda, bilgi güvenliğinin sivil uygulamalarını kapsam dışında tutmalıdır.

6.3. Şifreleme Algoritmaları

6.3.1. Şifreleme :

Şifreleme tekniği, sizin okuduğunuz bilgiyi bir başkasının okuyamayacağı bir yapıya dönüştürmek için kullanılır. Bu yöntemde bilgi, alıcı dışında başka bir kişi tarafından okunamaması yada değiştirilememesi için kodlanır. Bilgi, transfer sırasında bir başkasının eline geçse bile şifrelenmiş olduğundan okunması güçtür. Şifreleme ve şifreyi çözme için bir matematiksel algoritma ve bir anahtar gereklidir.

6.3.2. Anahtar :

Anahtar, şifrelemek veya deşifre etmek için kullanılan sayısal karakterler dizisidir. Simetrik anahtar algoritmasında şifrelemek ve deşifre etmek için aynı anahtar; açık anahtar algoritmasında şifrelemek için açık anahtar, deşifre etmek için ise gizli anahtar kullanılır. Dijital imzalar açık anahtar algoritmasını kullanır. Dijital imza imzanın sahibinin gizli anahtarı kullanılarak oluşturulur. Alıcı da imza sahibinin açık anahtarını kullanarak imzasını kontrol eder.

6.3.3. Şifrelenmesi Gereken Bilgiler :

Şifreleme konusunda karar verilmesi gereken konulardan biri de, nelerin şifreleneceğidir; Sadece bilgi içeren kısımlar mı şifrelenecektir yoksa bunların yanında kaynak ve hedef IP' lerini içeren başlık kısımları da şifrelenecek midir? Eğer sadece veri içeren kısımlar şifrelenirse – Cisco, Cylink ve NEC tarafından bu yaklaşım kullanılmaktadır – kaynak ve hedef IP adresleri değişmeden internet ortamına çıkarılacaktır. Bu paketlerden birini ele geçirecek kişi, adres bilgilerini elde edebilecektir. Bu bilgilerin başkaları tarafından ele geçirilmesi sorun yaratmayacakmış gibi gözükse de, bu bilgilerden yararlanılarak, firewall' un diğer tarafında yer alan routerlara veya başka cihazlara ilişkin bilgiler elde edilebilir. Bu bilgiler de kullanılarak firewall' u kandırmak, sanki şirket içinden bir makineymiş gibi ağa ulaşmak mümkün olabilecektir.

Tüm paketlerin şifrelenmesi durumunda böyle bir olasılık olmayacaktır. Şu an için bu işi yapan bir ürün IP **ESP (Encapsulating Security Payload)**' dir. Kısaca buradaki yaklaşım; Bir güvenlik aygıtı vasıtasıyla IP adres paketlerine yeni kaynak ve hedef adreslerinin eklenmesidir. Yalnız bu adresler, sadece firewall' ları adreslerler. Dolayısıyla

dışarıdan birisinin, firewall' un arkasında yer alan herhangi bir cihaza ilişkin bilgileri elde edebilmesi mümkün değildir. ESP yaklaşımını destekleyen şirketler; Border, Checkpoint, IBM, Raptor, V_One ve Western Datacom' dur.

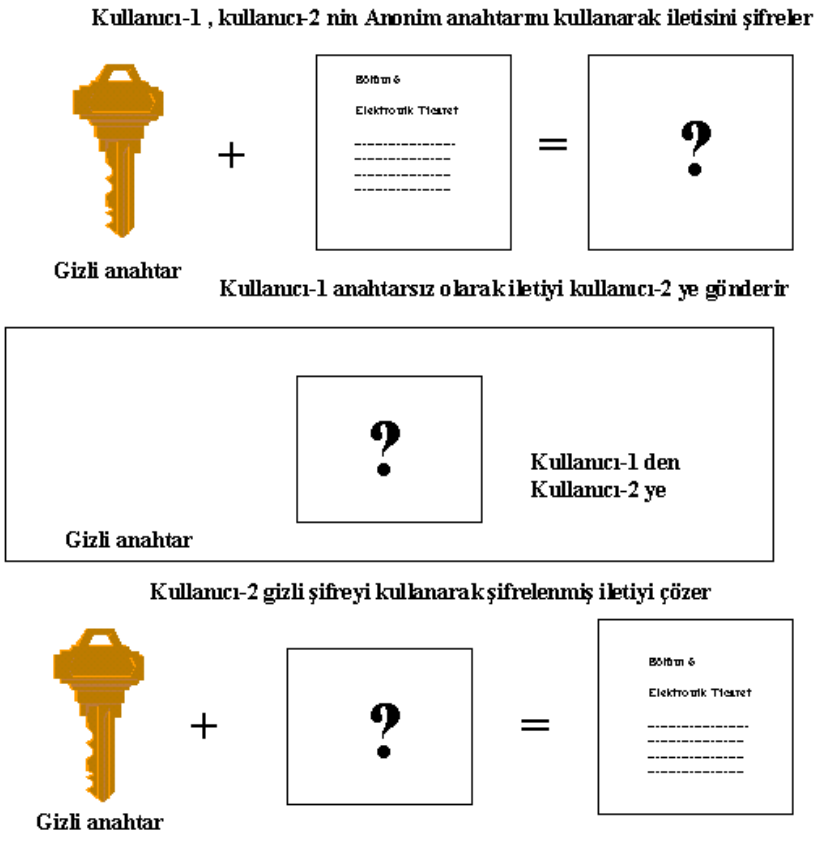
Fakat bu yaklaşım için de bir problem mevcuttur. IP paketine sürekli yeni adreslerin eklenmesi, bu paketin oldukça fazla büyümesine, hatta müsaade edilen sınırı aşmasına sebep olabilecektir. Böyle bir durumda da bu paketi bölmek gerekecektir. Bölümlenmiş paketlerin hedefe ulaştığında kodlarının çözülmesi biraz daha zor olacaktır. Ayrıca bölünmüş paketler hedefe uygun sırada gelmeyebileceklerinden bunlar da sorunlara sebebiyet verebilecektir.
IP şifrelemesi yukarıda bahsedilen sorunu önlerken aynı zamanda çalma olaylarının da önüne geçecektir. Burada yapılan işlem şudur; Dışarıdan ağa müdahale etmek isteyen kişi, düğümler arasında giden paketleri takip ederek, amacına uygun bir düğüme ilişkin bir paket yakalamaya çalışır. Böyle bir paket yakaladığında da, haberleşen iki düğümün arasına girip, hedef düğüm ile konuşmaya başlar. Hedeflenen düğüm hala daha kendi ağından bir makine ile haberleştiğini sanarak, veri alış – verişini sürdürür. Bu sayede de bu düğüm tarafından üretilen paketler alınırken, hedef düğüme de kendi kodlarını göndererek, bunların burada koşması, dolayısıyla buraya zarar verilmesine sebep olunabilir. IP şifrelemesi vasıtasıyla, bu sorunların da önüne geçilmiş olunur.

6.4. Şifreleme Algoritmalarına Giriş

Geçmişte ilgilenilen kriptografi algoritmaları algoritmanın gizliliğine dayanmaktaydı. Günümüzde kullanılmakta olan modern ve güçlü şifreleme algoritmalar ise artık gizli değildir. Bu algoritmalar güvenliklerini kullandıkları farklı uzunluk ve yapılardaki anahtarlarla sağlarlar. Bütün modern algoritmalar şifrelemeyi ve şifre çözmeyi kontrol için anahtarları kullanır. Bir anahtar ile şifrelenmiş bilgi, kullanılan algoritmaya bağlı olarak, ilgili anahtar ile çözülebilir. Anahtar yapısı bakımdan, şifreleme algoritmaları iki grupta incelenebilir:

6.4.1. Simetrik (Gizli) Anahtar Algoritmaları

Simetrik kripto-algoritmalar şifreleme ve çözmede aynı anahtarı kullanma prensibine dayalı olarak çalıştıklarından "simetrik" olarak nitelendirilirler. Bu tip algoritmalar gizli anahtar algoritması diye de adlandırılırlar. Başka bir deyişle şifreli iletinin çözülebilmesi için alıcının, kullanılan anahtarı daha önceden bilmesi gerekir.



Şekil.6.1 : Gizli Anahtar Algoritma

6.4.1.1. DES – (Data Encryption Standard)

En bilinen ve yaygın biçimde kullanılan simetrik algoritma **DEA (Data Encryption Algorithm)** dir. Bu algoritma **DES** (veri şifreleme standardı) olarak da bilinir.

6.4.1.2. DES ' e Giriş

1960'ların sonunda IBM'de çalışan bir araştırmacı olan Horst Feistel başkanlığındaki bir grup LUCIFER adı verilen bir şifreleme sistemi geliştirmiştir. 1973 yılında ABD standartlar enstitüsü NIST sivil kullanım için bir standart saptamak için firmaları davet etti. Yapılan incelemeler sonucu amaca en yakın çözüm LUCFIFER bulundu. 128 bitlik bir şifre anahtarına sahip LUCIFER üzerinde çalışan ABD güvenlik teşkilatı (**NSA**) uzmanları bazı düzenlemeler yaptılar ve anahtar uzunluğunu 56 bit'e indirdiler. Bu yeni algoritma 1977 yılında DES olarak yayınlandı ve kısa bir süre içinde başta finans endüstrisi olmak üzere birçok alanda **de facto standard** halinde kullanıma alındı.

DES klasik şifreleme sistemleri içinde efsanevi bir yere sahiptir ve bugün bile VISA, MASTERCARD, BKM, v.s. tüm kart sistemlerinin şifreleme omurgasını oluşturmaktadır. DES karıştırma, yerine koyma işlemlerini son derece dikkatli ve sistematik olarak yapacak şekilde tasarlanmıştır. Bunun yanı sıra en küçük değişikliğin, çok büyük farklar yarattığı çığ etkisi (**avalanche effect**) bulunmaktadır. Yani tek bir bitlik bir değişiklik bile sonucu tamamen değiştirmekte ve değişiklikler önceden tahmin edilememektedir.

NSA incelemeleri sırasında bu yerleştirme tablolarını kendisine göre değiştirmiştir (**s-box**) ve neden, nasıl yaptığını açıklamamaktadır. Ancak anlaşılan IBM araştırmacıların bilmeden NSA tarafından uzun zamandır bilinen bazı sırları keşfetmiş olmalarıdır. NSA bu değişiklikleri, DES'in gücünü artırmak için yapılmış optimum sayılar şeklinde açıklamaktadır. Bir firmanın kendine özel bir DES yaratmak için bu sayıları kendine göre rasgele düzenlemesi tavsiye edilmemekte ve büyük bir olasılıkla şifrenin zayıflayabileceği belirtilmektedir.

Bu düzenlemeler nedeniyle kripto dünyasında yoğun eleştiri ve kuşkular doğmuş, NSA'ın DES içine gizli kapılar yerleştirdiği iddia edilmiştir. Diğer bir eleştiri de anahtar uzunluğunun 128 bit yerine 56 bit uzunluğa indirilmesidir. Bunun nedeni olarak da NSA bilgi işlem gücünün 56 bit şifreleri kırabilecek olması gösterilmiştir.

Buna karşın NSA, 56 bit uzunluğun kolay hardware DES tasarımına olanak sağlamak için yapıldığını belirtmiştir. Bunun da ötesinde bir süre sonra NSA bir yanlış anlama sonucu DES standardının kamuya açıklandığını, oysa onların bunu sadece hardware olarak uygulanacak bir sistem olarak düşündüklerini, bunu güçlendirmeye ve optimize etmeye çalıştıklarını, ancak açık olarak yayınlanan DES standardının yayınlanmasının ardında normal ticari kullanıma vermek istediklerinden güçlü bir şifreleme yönteminin piyasa çıkmasından rahatsızlık duyduklarını belirtmişlerdir.

Şifre sistemlerinin güvenliğini şifre anahtarlığının uzunluğu belirlemektedir. Genel olarak belirli bir yönteme de 10 yıl ömür biçilmektedir. DES 1999 yılı itibariyle 22 yaşındadır ve ömrü birkaç uzatılmıştır. Artık yeteri derecede güvenli değildir. Birçok batı ülkesinin güvenlik servislerinin bilgi işlem kapasitesi DES şifrelerini birkaç dakika içinde kırabilecek düzeye gelmiştir. Bunun da ötesinde amatör şifre kırıcılar aranacak anahtar dilimini on

binlerce PC' ye bölerek Internet üzerinden dağıtmakta ve aynı anda on binlerce PC çalışarak birkaç gün sonra sonuca erişebilmektedir. Bakılacak şifre adedi 2^{56} adettir. Bu bir PC için çok büyük bir görevdir ama on binlerce PC veya çok güçlü bilgisayarlar aranan bilgiyi fizibl bir sürede bulabilirler.

DES algoritması GİZLİ ANAHTAR yöntemini kullanan SİMETRİK bir kriptodur. Yani her iki tarafta aynı anahtarları bilmek zorundadır. BKM + 40 BANKA veya VISA + Binlerce Banka için zorda olsa uygulanabilir bir sistemdir. Ama anahtarları gizlemenin ve dağıtmanın zorluğu uygulamayı kısıtlamaktadır.

Aynı anahtar hem şifrelemeye, hem de çözmeye yarar yani şifreleyen algoritmaya şifreli bilgi verilip aynı anahtar kullanılarak algoritma tersten işletildiğinde şifre çözülür.

6.4.13. 3DES (Triple DES)

Standart DES'in 112 veya 168 bitlik iki veya üç anahtar ile artarda çalıştırılması ile oluşturulan bir şifreleme tekniğidir. Anahtar alanı 2^{112} veya 2^{168} sayısına ulaşınca bugün için veya tahmin edilebilir bir gelecekte çözülmesi mümkün olmayan bir kod olmaktadır. Buna "**strong crypto**" denilir, en güçlü teknik güç olan ABD'nin bile çözmesi mümkün değildir.

Bu nedenle kullanımı ve yayılması sınırlanmak istenmektedir. ABD ve birçok batı ülkesi 40 bit' ten daha güçlü kripto sistemlerin ihracını kısıtlamaktadır. 40 bit, birkaç saniyede ABD güvenlik kurumu NSA tarafından çözülebilmektedir ve böylece "**real-time**" dinlemeyi olanaklı kılmaktadır. 56-bit bile bugün için belirli bir süre gerektirmektedir. Şüphesiz anında çözebilecek güçte bazı sistemler vardır ama sayısı sınırlıdır ve maliyeti yüksektir. Bu nedenle bu gücün ciddi işlere tahsis edilebilmesi için 40 bit üzeri şifreler kısıtlanmaktadır. Türkiye'de bankalara verildiği söylenen 128 bit gücündeki sistemler, sözde SET ile çalışan siteler, v.s. hepsi bir aldatmacadır. Bu sistemlerdeki tüm şifrelerin anası firmalar tarafından NSA' ya teslim edilmektedir. NSA istediği bankanın bilgisayarına girerek kendi şifresiyle istediği şifre anahtarını çekebilmektedir. Başka türlü bir firmanın ABD'den ihracat izni alması mümkün değildir.

Türkiye'de 128-bit şifreleme ile çalıştığını iddia eden veya SET olduğunu öne süren tüm siteler 40-bit SSL ile çalışmaktadır, yapılan iddialar tümüyle gerçek dışı veya sanaldır.

Bunu ilgili siteye bağlandığınızda çıkan kapalı anahtar üzerine tıklayarak görebilirsiniz. 128 bit görünen siteler ise sanki 128 bitmiş gibi gözüken ama aslında anahtar sahalarının belirli bölümü ABD tarafından bilinen veya tümü arkadan dolaşılarak öğrenilebilen sitelerdir.

DEA simetrik–blok şifreleme algoritmasıdır. Şifreli metin ile açık metin aynı uzunluktadır, yani bu algoritma açık metnin uzunluğuna ek katmaz. Blok uzunluğu, anahtar gibi 64 bittir (8 byte).

Anahtar, 8 parity biti içerir. Soldan sağa 8, 16, 24, …, 64 nolu bitler parity bitleridir. Parity bitlerinden dolayı DEA'nın şifre uzayı 2^{56} dır. Bu yaklaşık olarak $7.2x10^{16}$ farklı anahtar olasılığı demektir. Buna rağmen hızlı bir bilgisayarla açık/şifreli (public/secret) metin çiftini elinde bulunduran birisi bu olasılıkları deneyerek gizli anahtarı bulabileceğinden DEA sanıldığı kadar güçlü değildir.

DEA şifreleme algoritması donanımsal gerçeklenmek üzere tasarlanmıştır. Bununla birlikte, DEA modülü içeren smartkart olmadığından dolayı yazılımsal olarak gerçeklenmektedir. Bu da 1 kbyte lık bir assembler kodu demektir.

Güvenlik seviyesini daha da arttırmak için üçlü-DES (**Triple DES**) kullanılabilir. Triple Des 168 bitlik bir anahtarın üç kez Des kullanımıdır. Burada üç DEA işlemi şifre ve çözme zinciri içerisinde şekil 1.1 de görüldüğü gibi birbirine bağlı olarak yürütülmektedir. Burada anahtar uzunluğu 16 byte tır.

Şifreleme yöntemi olarak DES' ten başka algoritmalar da piyasada bulunmakta ve kullanılmaktadır. Bunlar; Checkpoint tarafından üretilen FWZ1, IRE tarafından üretilen Atlas ve IBM tarafından üretilen Command Masking Data Facility (CMDF) dir. Bunların tamamı 40 bitlik algoritmalardır. Northern Telecom Ltd. tarafından geliştirilen CAST, 40 ila 64 bit aralığında değişen uzunlukta anahtar kullanmaktadır.

6.4.2. Asimetrik Kripto-Algoritmalar

Whitfield Diffie ve Martin E.Hellman 1976 da iki farklı anahtara dayalı algoritma geliştirilebileceğini gösterdiler. Anahtarlardan biri açık (**public**), diğeri ise gizli (**secret**) olmalıdır. Bu yöntemde şifreleme için kullanılan anahtarlar ile şifre çözümünde kullanılan

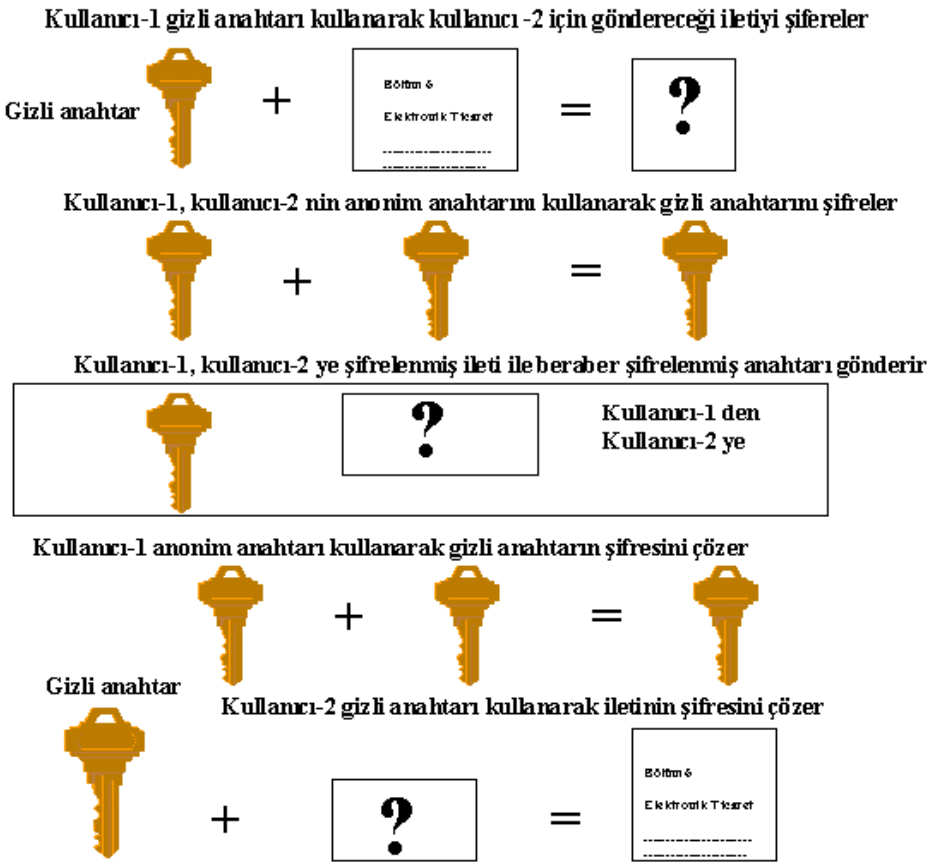
anahtarlar farklıdır. Bu nedenle bu algoritma tipinde şifreleme için kullanılan anahtar bilinse dahi (zaten bu anahtar açıktır ve herkes tarafından kolaylıkla öğrenilebilir) , bu anahtarı kullanarak şifreyi çözmek için kullanılan anahtarı elde etmek olanaksızdır. Böylece ilk defa, gizli simetrik anahtar dağıtım problemi çözülmüş oldu ve herkes tarafından gerçeklenebilecek sayısal imza gibi yeni yöntemler mümkün oldu.

Açık anahtarlı altyapılar, bir gizli ve açık anahtar çifti ve bu çiftle sağlanan "elektronik kimlik, sayısal imzalama ve şifreleme" işlevleriyle, gerekli kurumsal ve yasal yapılanma üzerine kurulmuştur.

Burada şu noktalar önemlidir:

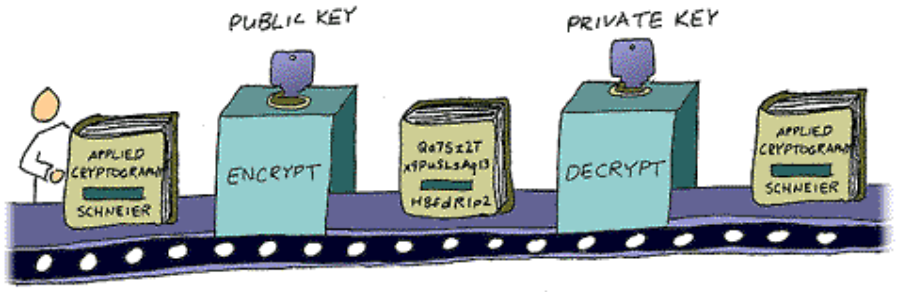
* Kripto işlemleri donanım üzerinden yazılıma kaymıştır. Örneğin kripto teçhizatı, saklanması, imhası ve benzeri unsurlar bu altyapıda karşılaşılan ve kullanılan terimler değildirler.
* Söz konusu yazılımlarda kullanılan algoritmalar tümüyle kamuya açıktır. Bunun anlamı, bu algoritmaların çok sayıda taraf tarafından testinin yapılabilir olması ve bu yolla güvenilirliğinin artmasıdır.
* Anahtarların üretilmesi, saklanması ve tüm işlevlerin yürütülmesi için uluslararası açık standartlar belirlenmektedir (X.509 elektronik kimlik belgesi standardı, PKCS açık anahtarlı altyapılar standartları gibi). Bu standartların dışına çıkmak pratik değildir.

Bu tip algoritmalarda gizli anahtarla şifrelenen iletilerin anonim anahtarlar ile çözülme zamanı , gizli anahtar algoritmalardakine kıyasla çok daha uzundur. Bu zaman ileti uzunluğu ile üssel olarak artar. Bu nedenle anonim anahtarla şifreleme yöntemi birçok uygulamada gizli anahtarla şifreleme yöntemi ile birlikte kullanılır. Örneğin elektronik postaların şifrelenip gönderilmesinde en yaygın kullanılan yöntemler hem gizli hem de anonim anahtar algoritma yöntemlerini içerenlerdir.



Şekil.6.2 :Açık anahtar algoritmaları ile şifreleme ve çözme

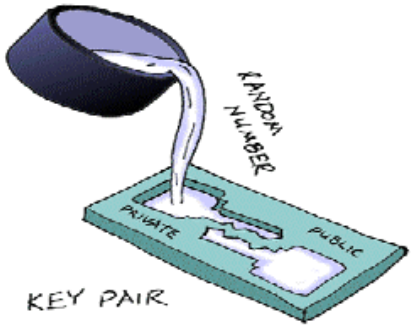
Şifreleme mekanizması için karar verilmesi gereken bir nokta da, private-key mi yoksa public-key mi kullanılacağıdır. (Aslında public-key ile anlatılan bir public-key ve bir private-key içerir. Karışıklığı önlemek ve yöntemleri bir birinden ayırmak için böyle bir isimlendirme yapılmıştır.)



Şekil.6.3 : Açık Anahtar Mekanizması

Esas olarak bir anahtar, aktarılacak verinin kodlanması ve kodlanmış verinin çözülmesi için kullanılan bir algoritmadır. Private-key şifreleme yaklaşımında, her iki işlem için de aynı anahtar kullanılır. Public-key şifreleme yaklaşımında ise public-key ve private-key birlikte kullanılırlar.

Private-key yaklaşımı kullanılarak geliştirilmiş ürünlerde, ağ üzerinde çalışmakta olan herkese bir anahtar verilir. Buradaki önemli nokta, bu anahtarların, istenmeyen ellere geçmesine mani olmaktır. Burada da, private-key' lerin kullanıcılara nasıl dağıtılacağı konusu gündeme gelmektedir. Çünkü bu dağıtım işlemi esnasında da anahtarlar istenmeyen ellere geçebilir. Bu iş için e-posta 'dan yararlanılabileceği gibi (gerekli güvenlik önlemleri alınmak kaydıyla), telefonla da bu anahtarlar sahiplerine aktarılabilir. Tabi tüm bu işlemler sırasında güvenliğe azami derecede dikkat etmek gerekmektedir.



Şekil.6.4 : Anahtar Oluşturma

Şifreleri kişilere atamanın bir başka yolu da, bir sunucu vasıtasıyla dağıtma işlemini yapmaktır. Bu durumda, şifreleri ele geçirmek isteyenlere direkt bir hedef sunulmakta, burada sağlanacak yeterli bir güvenlik mekanizması ile, bu tehlike de önlenebilmektedir.

Private-key yaklaşımı ile ilgili olarak yaşanabilecek bir başka sorun da, ortak kullanılacak bir bilginin karşılıklı olarak paylaşıma açılması esnasında yaşanabilecektir. Bunun için öncelikle private-key' lerin değiş-tokuş edilmesi gerekmektedir. Bunun için e-mail sisteminin kullanıldığını düşünürsek, bu zaman alabilecek,bu gecikmeler de sorunlara sebep olabilecektir.

Public-key yaklaşımında ise, ağ üzerinde yer alan her kullanıcıya iki anahtar atanır : private-key ve public-key. Private-key' ler, yukarıda anlatıldığı gibi kullanıcılara dağıtılır.

Burada gizlilik yine esastır. Bütün kullanıcılara ait public-key' ler ise, herkesin erişimine ve kullanımına açıktır.
Bu şekilde her kullanıcıya iki ayrı anahtar atamanın temel sebebi, private-key ile yaşanan, verilerin karşılıklı olarak aktarılması esnasında zorunlu olan, private-key' lerin değiş-tokuş'u işleminden kurtulmak, dolayısıyla bunun getirdiği zorlukları bertaraf etmektir. Yöntemin nasıl çalıştığını anlatarak, mekanizmayı daha iyi anlayabiliriz;

Farklı yerlerde bulunan iki kullanıcıdan birinin diğerine bilgi aktarmak istediğini düşünelim. Bu durumda şifrelemede ihtiyaç duyacağı anahtarlar, göndereceği kişinin public-key'i ve kendisinin private-key' idir. Bu iki anahtarı kullanarak şifreleme işlemini yapar ve verileri karşı tarafa gönderir. Şifrelenmiş verileri almış olan kişi ise, şifreyi çözmek için, göndericinin public-key' ini ve kendisinin private-key' ini kullanır. Dolayısıyla bir kişinin public-key' ini bilmek, ancak private-key' ini de bilmekle anlamlıdır, aksi taktirde hiç bir işe yaramaz. Ayrıca bu yöntem private-key' lerin karşılıklı aktarılmasını da gerektirmez.

Görüldüğü gibi public-key yaklaşımı bize büyük bir avantaj getirmiştir. Ancak bunun da dezavantajları vardır. Bunların başında da, birbirlerine şifrelenmiş veri gönderecek bilgisayarların her defa yürütmeleri gereken el sıkışma protokolleri, ilgili anahtarları kullanarak bilgileri çözme işlemlerinin CPU zamanından çok fazla almasıdır. Bu işlemler, hesap yoğunluğu olan işlemler olduğundan sistemi oldukça yavaşlatacak, ağ performansının %20 azalmasına sebep olabilecektir.

Şu an için private-key yaklaşımı ile ilgili olarak üzerinde çalışılan konuların başında, anahtar değiştirme işlemini otomatik yaparak, zaman kaybını en aza indirmektir.

Sun Microsystems Inc. tarafından geliştirilmiş olan **SKIP** (**S**imple **K**ey Exchange **I**nternet **P**rotocol) yaklaşımı ile de, public-key yönteminin getirdiği her seferinde güvenli erişim için el sıkışma protokollerinin koşturulması ve dolayısıyla işlemciye fazladan yük getirilmesi durumundan kurtulmaktadır. Bunun yerine yapılan işlem şöyledir; Bir kere güvenli haberleşme oturumu başlayınca, **SKIP** tarafından bir oturum anahtarı oluşturulur. Bu anahtar geçici bir anahtar olup, bu oturum süresince ilgili kaynaktan veri aktarımı için kullanılır.

SKIP ilk bakışta gerçekten çok güzel bir yaklaşım olarak gözükmektedir. Ne yazık ki burada da güvenlik açısından bazı sorunlar vardır, şöyle ki; Oturum anahtarı kullanıcı tarafından belirlenen bir periyot boyunca geçerlidir. Bu periyot bir saat olduğu gibi, bir kaç gün de olabilir. Anahtar, network ortamında bir yerde tutulduğundan, buna ulaşacak kişiler, bu anahtarın geçerli olduğu süre zarfında, istedikleri verilere ulaşabilirler.
SKIP yaklaşımından yola çıkılarak, değişik yöntemler geliştirilmiştir. SKIP' in güvenlik konusundaki açığını kapatmak ve daha güvenli bir iletişim sağlamak amacıyla **P**hoturis **S**ession **K**ey **M**anagement **P**rotocol (**PSKMP**) geliştirilmiştir. Burada yapılan, bir kere güvenli oturum başlatıldıktan sonra oturum anahtarını sabit tutmamak, bazı rasgele sayılar, haberleşen bilgisayarların IP bilgilerinden oluşan değerlerin bir karışımını kullanarak, her yeni veri transferinde bunların belirlediği bir oturum anahtarını kullanmaktır. Bunun sayesinde de güvenlik SKIP' e göre daha iyi bir düzeye getirilmiş olur, hız olarak da pek çok public-key ürününden daha iyi bir seviyeye ulaşılır.

6.4.3. Kriptografik Algoritmaların Gücü

Teorik olarak, bir anahtar kullanan kriptografik algoritmalar, olası bütün anahtarları sırasıyla denemek yoluyla kırılabilir. Olası anahtarların denenmesi işlemi de, anahtarın uzunluğu arttıkça güçleşecektir. Örneğin 5 bitlik (5 adet 0 veya 1'den oluşan) bir anahtarın, en fazla 2^5=32 farklı anahtarı olabileceğinden, bu anahtarların sırayla denenmesi, şifrenin çözülmesine yeterli olacaktır. 32 bitlik bir anahtar için 2^32 yani 10^9 deneme yapılması gerekir. Bu bir amatörün kendi ev bilgisayarıyla deneyebileceği bir şeydir. 40 bitlik anahtara sahip bir sistem için 2^40 adim gerekir ki, bu da birçok üniversitenin ve hatta bazı küçük şirketlerin bile sahip olabileceği bir güçle mümkündür. 56 bitlik anahtarlı bir sistem (DES gibi) oldukça fazla bir çaba gerektirir fakat özel bir donanım yardımıyla bu da kırılabilir. Bu özel donanımın maliyeti oldukça fazla olmasına rağmen,organize suçluların, büyük şirketlerin ve hükümetlerin sahip olabilecekleri bir güçtür. 64 bitlik anahtarlı sistemler, büyük hükümetlerce kırılabilmektedir ve yakın zamanda organize suçlular, büyük firmalar ve daha küçük hükümetler tarafından da kırılabilecektir.

Saniyede 3 trilyon anahtarı deneyebilen bir makine ile (yaklaşık 1 milyon dolar yatırımla elde edilebilecek bir makine), 56 bitlik DES algoritmasını 3,5 saat içinde kırılabilir. Aynı makine 80 bitlik anahtar kullanan bir algoritmayı 6.655 yılda, 128 bitlik anahtar kullanan bir algoritmayı ise 2.000.000.000.000.000.000 yılda kırabilir. Gelişen teknoloji ve

işlemci hızları ile bu tip işlemlerin daha hızlı gerçekleştirilebileceği açıktır.

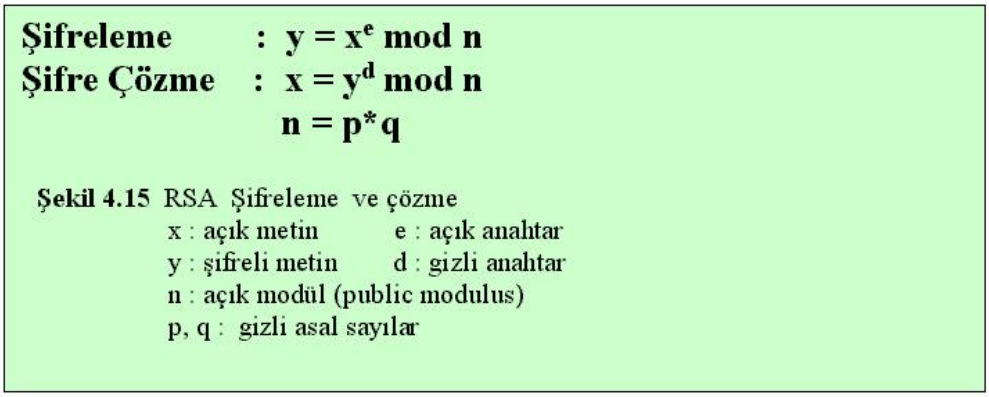
250 bitlik anahtar kullanan bir algoritmanın kırılması için gereken çabayı hesaplamak için termodinamik yasaları kullanılabilir. Buna göre, bir bit işlemi için gerekli minimum enerji kullanılarak yapılan hesaplamada, 250 bitlik tek bir anahtarı kırmak için güneşin tahmin edilen ömrü boyunca dışarı verdiği tüm enerjinin gerekli olduğu ortaya çıkar. Güneşin sadece bir yıl boyunca dışarıya verdiği enerji ile yetinecek olursak, en fazla 192 bitlik bir anahtarı kırmamız mümkün olur. Görüldüğü gibi, teorik olarak kırılabilir kabul edilen algoritmaların
pratikte kırılması, anahtar büyüdükçe imkansızlaşmaktadır.

Çift anahtarlı kriptografide kullanılan anahtarların uzunlukları simetrik anahtarlı kriptografide kullanılanlardan genellikle çok daha büyüktür. Açık anahtar kriptografisini kırabilmek için, doğru anahtarı tahmin etmek değil, açık anahtardan ona uygun gizli anahtarı elde etmek gerekmektedir. Örneğin RSA için bu, iki büyük asal çarpanı olan büyük bir sayının çarpanlarına ayrılması anlamına gelmektedir. Bu işlem de teorik olarak yapılabilir olmasına karşın, pratik olarak gerçekleştirilmesi günümüz teknolojisiyle imkansızdır.

Bir kriptosistemin gücü genellikle en zayıf noktasına eşittir. Dolayısıyla, algoritma seçiminden, anahtar dağıtımına ve kullanım koşullarına kadar hiç bir şey göz ardı edilmemelidir.

6.4.4. RSA Algoritmaları

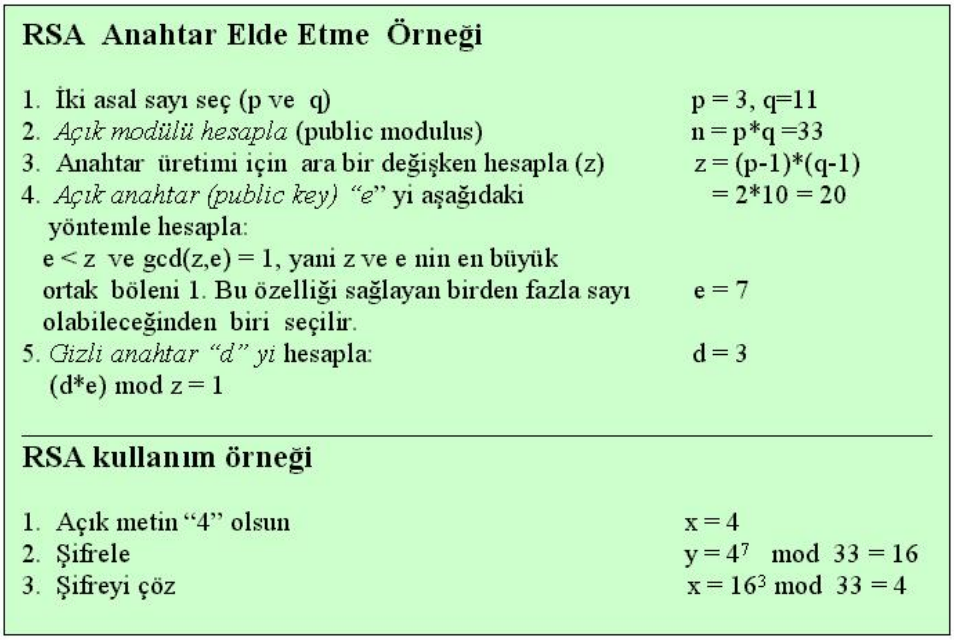
1978 yılında, Ronald L.Rivest, Adi Shamir ve Leonard Adleman yukarıdaki kriterleri sağlayan bir algoritma önerdiler. RSA olarak bilinen ve dünyada en yaygın biçimde kullanılan asimetrik algoritma, ismini mucitlerinin baş harflerinden almıştır. Büyük sayıların aritmetiğine dayalı çok basit bir prensibi vardır. Anahtarlar, iki büyük asal sayıdan üretilir
Metotları geniş çapta kabul edildi.
Ayrıca RSA dijital imza projesi dördüncü bölümde diğer bilinen imza projeleri ile birlikte detaylı olarak açıklanmıştır.



Şekil.6.5: RSA

Şifrelemeden önce, açık metnin uzunluğu uygun bir değere gelecek şekilde ek yapılmalıdır. Eklenecek alan kullanılan anahtar uzunluğuna bağlıdır. Şifreleme açık metnin exponansiyelinin (üssünün) alınması ve bunu takiben bir modül alma işleminden ibarettir. Bu işlem sonucunda şifreli metin elde edilir ve ancak gizli anahtarın bilinmesiyle benzer bir işlemle açılır.

Dolayısıyla, algoritmanın güvenliği büyük sayı üretme problemine dayalıdır. İki asal sayının çarpımından hareketle açık modülü hesaplamak çok kolaydır, bununla birlikte açık modülü oluşturan asal çarpanları bulmak oldukça zordur. RSA algoritmasının anahtar üretim algoritması aşağıdaki basit örnekte görülmektedir.



Şekil6.6 : RSA Anahtar Elde Etme Örneği

Smartkartın RAM'ı büyük sayıların eksponansiyelinin hesaplanması durumunda yetersiz olacağından *"modülo-üsleme"* (modulo exponentiation) denilen, ve hesaplanan değerlerin asla modülü geçmesine izin vermeyen bir yöntem kullanılır. Örneğin x^2 mod n değerinin hesaplanmasında yöntem çok büyük sayılarla uğraşılması gerekeceğinden (x*x) mod n işleminde biçiminde çalışmaz, bunun yerine aynı sonucu veren (((x mod n)*(x mod n)) mod n) yol tercih edilir. Bu daha küçük sayılarla uğraşılması demek olduğundan RSA için kullanılan bellek ve adım sayısı indirgenmiş olur.

Saldırıları zorlaştırmak için anahtar uzunluğu mümkün olduğunca büyük seçilmelidir. Açık ve gizli anahtar uzunlukları farklı olabilir, açık anahtarın uzunluğunun az olduğu durumlarda sayısal imzanın kontrolü için gereken zaman önemli ölçüde azaldığından genel olarak açık ve gizli anahtar uzunlukları farklı seçilir.

Bir saldırgan açık modülü (public modulus) asal çarpanlarına doğru biçimde ayırırsa anahtarı elde edebilir. Küçük bir sayı için bu kolay olmakla birlikte büyük bir sayı için bu oldukça zordur. RSA anahtarları bu yüzden yeterince büyük olarak seçilir. RSA de (512 bit=64 byte) anahtarlar yeterince büyük olarak kabul edilir. Bununla birlikte 768 bit ve 1024 bitlik anahtarlarda kullanılır. Anahtar uzunluğu şifreleme ve çözme zamanını doğrusal olarak değil eksponansiyel olarak arttırmaktadır.

RSA hesaplanmasını 8-bitlik mikroişlemcili bir smartkartta gerçeklemek bir kaç dakikadan uzun sürecektir. Bu yüzden RSA hesaplamayı destekleyici aritmetik birimler içermektedirler. Donanım destekli bir RSA algoritmasının program kodu 300 bayt uzunluğundadır. 768 bit ya da 1024 bitlik RSA için 1 kbyte lık bir assembler kodun kartta yer alması gerekir.

RSA bütün gücüne rağmen uzun işlem gerektirdiğinden veri şifreleme için ender kullanılır. Asıl kullanım alanı sayısal imzalardır.

6.4.4.1. RSA ile Kredi Kartı Şifreleme

Kredi Kartı şifrelemede M kart numaramız olsun ; M= 6882 3268 7966 6683,
Önce bu sayı küçük sayılara ayrılır. Örneğin;

m1=688 m2 =232 m3=687 m4=966 m5=668 m6=3
teker teker her biri şifrelenir.

688^{79} (mod 3337) = 1570 = c1
Aynı işlemleri diğer m_i değerleri için yaptığımızda

C = 1570 2756 2714 2276 2423 158 değerini elde ederiz.

Bu mesaj ağ üzerindeki terminale gönderilir. Mesajın şifre çözümü terminalde yapılır. Terminal müşterinin bilmediği gizli anahtar değerini ("d") bilir. Böylece terminal yukarıdaki mesajın şifresini çözer.
C1 : 1570^{1019} (mod 3337) = 688 = m1

Bu yolla mesajın arta kalan kısımları geri alınabilir.

6.4.4.2. Veri Alanı Geniş Rakamlarla RSA Örneği

Aralarında asal iki küçük rakam seçilir (p,q) p=5 q=17
Açık modül hesaplanır (public modulus) n=p*q
Ara bir değişken hesaplanır (phi(n)) phi(n)= (p-1)*(q-1)=4*16=64
Bir sonraki adım şifreleme anahtarı seçmek .Bu örnekte e=13 değeri kullanıldı.
Mesaj olarak M=3 değerini kullanmayı öne sürüyoruz. Fakat istediğiniz mesaj değerini seçebilirsiniz.5 'ten küçük bir değer olması hesap makinesinden kontrolü mümkün kılar.
Örneğin; Eğer 120 'nin 13 ile bölümünü bilmek istiyorsak hesap makinemizi
120/13=9.23076923077 değerini elde etmek için kullanırız.
Sonra tamsayı kısmını ondalık kısımdan (9) ayırmak için çıkarılır. Sonunda sonuç 13 ile çarpılır. 13*23076923077=3.0000000001 sonucu elde edilir.

Birinci adım :Mesajın üssü alınır. e=12 olsun
M^e =3^13=1 594 323
Şimdi n=85 ile bölümünden kalan kısım bulunur. Sonuç 63 olur.
M^e =3 ^13 =63 (mod n)

İkinci adım :Mesajı çözmek.

Sonuç ^5=63^5 =992 436 543

Ve hesaplama ;

63^5=992 436 543 =3 (mod n)

Eğer bu gerçel sayılarla yapılacaksa hesaplamayı kolaylaştırmak için birkaç teknik kullanmalı. Açık anahtar (e,n) ve gizli anahtar (d,n)

6.4.5. Özet Olarak Açık Anahtar Kriptografisi

Bir açık anahtar şifreleme ve ayırma farklı gizli anahtar mesajı kırmak için kullanılır.
Her bir grup bir çift anahtar üretimini gerçekleştirir. Her bir grup açık anahtarlarını ilan eder. Her bir grup gizli anahtarlarını koruma altına alır. A 'nın B ye bir mesaj göndereceğini varsayalım. İlk olarak B yi açık anahtar olarak kullanarak mesajı şifreler. B gizli anahtar kullanarak mesajın şifresini çözebilir. B 'nin gizli anahtar olduğunu kimse bilemez. Bu tamamen korunur ve şifre kırılamaz. Hala açık anahtarların asıllanması konusunda problemler vardır.

6.5. Güvenlik Metotları

6.5.1. Smart Kartlarda Güvenlik

Güvenlik metotları günlük hayatta çok geniş bir uygulama alanına sahip olup para çekme makinelerinden telefon kartlarındaki PIN kodlarına kadar çok farklı ortamlarda karşımıza çıkarlar. Biz bu çalışmada, uygulama alanlarının önemli bir parçasını oluşturan smart kartların yapısı ve içeriğinde kullanılan şifreleme/deşifreleme tekniklerini inceledik.

Smartkartların manyetik kart ve disklere göre en büyük avantajlarından birisi veri koruma ve güvenliğidir.

6.5.1.1. Kullanıcı Tanıma

Genel olarak bir kullanıcıyı tanımak için üç farklı seçenek kullanılabilir:
Güvenlik bilgisi (örnek: PIN girişi)

Fiziksel nesneler (örnek: kapıyı açmak için anahtar taşımak)
Biyolojik özelliklerin ölçümü (örnek: parmak izi, retina okuma)

İlk iki yöntemin tanınacak kişinin PIN ezberlemek zorunda olması ve /veya yanında anahtar taşıması gerekliliğinin olması gibi dezavantajları vardır. Üçüncü yöntemde böyle bir durum yoktur, ancak boy ve ağırlık gibi basit biyolojik özelliklerin değişmesinden dolayı çoğu durumda karmaşık teknikler kullanmak gerektirmektedir.

6.5.1.2. Asıllama (Authentication)

Asıllamanın amacı haberleşen nesnelerin kimlik ve gerçek olma özelliklerini kontrol etmektir (doğrulamaktır). Bu kavram smartkartta, kart ve terminalin birbirleri açısından doğru nesneler olduğunun denetimi için kullanılır.

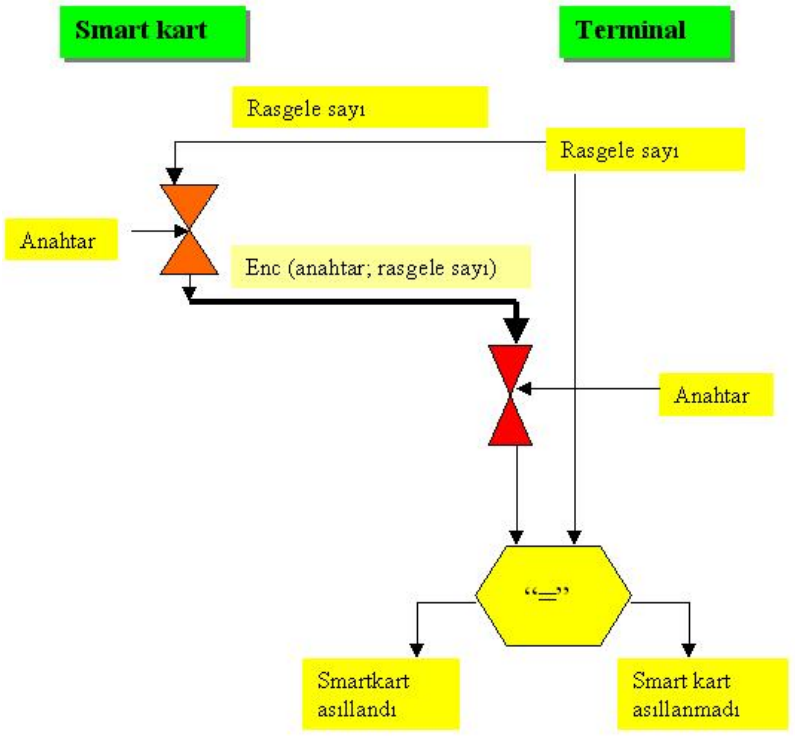
İki taraf da asıllama yönteminde kullanacakları bir ortak gizli bilgiye sahip olmalıdır. Bu basit bir PIN denetiminden daha güvenlidir. Klasik PIN kullanımında PIN karta açık olarak gönderilir, bu gizli bağlantı kurularak PIN'in kolayca elde edilebilmesini sağlar.
Asıllama yönteminde ise, ortak gizin (PIN) bağlantılar üzerinden elde edilmesi imkansız olmalıdır, dolayısıyla herkese açık ortamda gönderilemez. Asıllama yöntemleri statik ve dinamik olmak üzere ikiye ayrılır. Statik yöntemde, her zaman aynı statik data kullanılır. Dinamik asıllama, bir önceki oturumda (**session**) elde edilen verinin tekrar gönderilme olasılığına karşı bile korumalı bir yöntemdir, çünkü her oturumda farklı bir ortak giz (**anahtar/key**) kullanır.

Asıllamanın yönü açısından tek-yönlü (bidirectional) ve çift yönlü (unidirectional) iki tür asıllamadan bahsedilebilir. Tek yönlü de taraflardan birisi asıllanırken, diğerinde iki tarafta asıllanır (**authenticated**).
Kriptografik algoritmalara dayalı olan asıllamalar simetrik ve asimetrik olmak üzere ikiye ayrılabilir. Smart kartlı sistemler yaygın biçimde simetrik yöntemler kullanmaktadır. RSA ve benzeri tabanlı olan asimetrik yöntemler ise yavaş olmalarından dolayı pratik kullanım için uygun değildirler.

Smartkartlarda asıllamada, taraflardan biri diğerine rasgele bir sayı üretip gönderir, diğer taraf bir algoritma kullanarak cevap hazırlar ve bunu gönderir. Doğal olarak, tercih edilen algoritma iki tarafın da sahibi olduğu ortak bir anahtar kullanarak şifreleme yapan algoritmadır.

6.5.1.2.1. Tek Yönlü Simetrik Asıllama

Tek yönlü asıllama ikinci tarafın kimliğini sorgulamak için kullanılır. Terminal rasgele bir sayı üretir ve bunu karta gönderir. Kart bunu şifreler, bu işlemde kullanılan anahtar sadece kart ve terminal tarafından bilinir. Yöntemin güvenliği bu anahtara dayalıdır, zira doğru cevap ancak anahtar sahipleri tarafından üretilebilir. Kart şifrelemenin sonucunu terminale gönderir. Terminal gizli anahtarla bu şifreyi açar ve elde ettiği değeri karta ilk gönderdiği değerle karşılaştırır. İki değer aynı ise, terminal kartta doğru gizli anahtarın (secret key) olduğunu anlar ve kartın hakiki olduğu sonucuna varır.

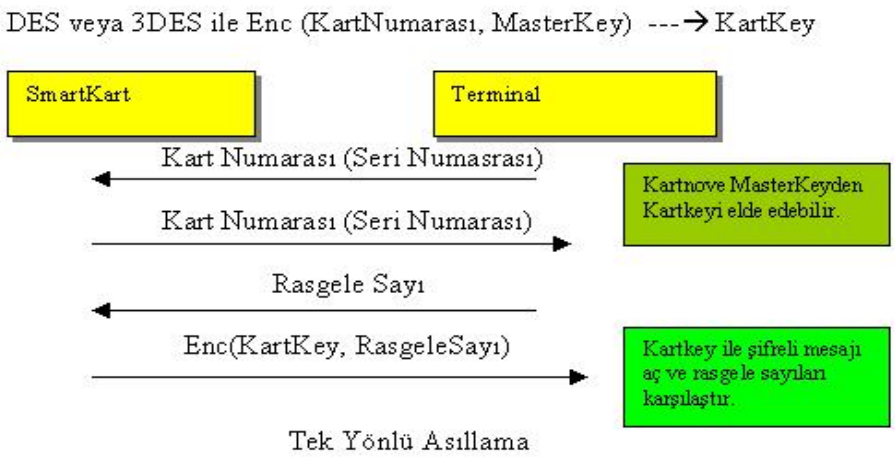


Tek yönlü simetrik asıllama (Terminal üzerinden smart kartın asıllanması)

Şekil 6.7 Smart Kart

Uygulamadaki bütün kartlar aynı anahtar bilgisini tutarlarsa, anahtar ele geçirildiği zaman, bütün sistem tehlikeye girecektir. Bu yüzden pratik uygulamalarda anahtar kart

bazlıdır. Her kartın gizli-olmayan bir kart özelliğinden üretilmiş ayrı bir anahtarı vardır. Bu çipin seri numarası gibi bir özellik olabilir. Tekil anahtarı hesaplayabilmek için, terminal karttan çip numarasını ister. Her karta özgü olan gizli asıllama anahtarı, sadece terminal tarafından bilinen master anahtar ve kart numarasının fonksiyonudur. Kart numarasının bir kısmı master anahtarla şifrelenir, sonuç kartın özgün asıllama anahtarıdır. DES ya da Trible-DES kullanılabilir. Bununla birlikte, sadece terminal tarafından bilinen master anahtarın yetkisiz kişiler tarafından ele geçirilmesi halinde, güvenlik sisteminin tümü tehlikeye girecektir. Bu yüzden, master anahtar terminal içerisinde çok güvenli biçimde korunmalı (güvenlik modülü içerisinde saklanabilir) ve bir saldırı durumunda silinmelidir.

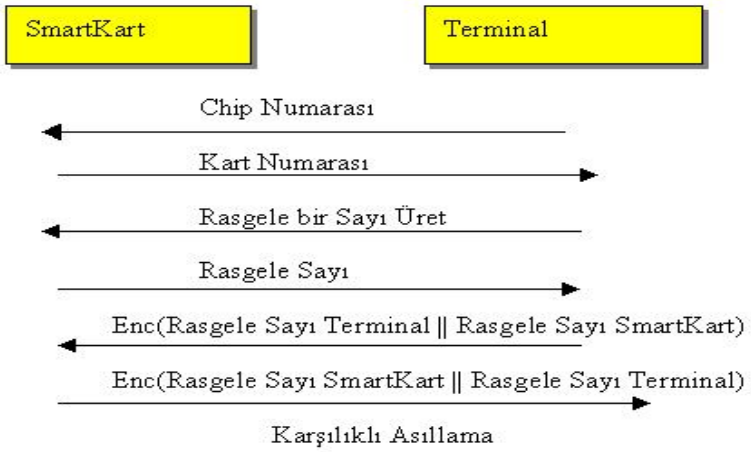


Şekil 6.8 Tek Yönlü Asıllama

Kart hesaplama anahtarı terminal tarafından hesaplandıktan sonra klasik sorgu-yanıt fazı başlar. Kart terminalden bir rasgele sayı alır, bunu kendi özgün anahtarıyla şifreler ve terminale gönderir. Terminal şifreyi açar ve gönderdiği değerle karşılaştırır. İki değer de aynı ise, kart terminal tarafından asıllanmıştır.

6.5.1.2.2. Karşılıklı Simetrik Asıllama

Karşılıklı asıllama, tek-yönlü asıllamanın iki defa yapılmasına dayanır. Terminalin kartın asıllama anahtarını hesaplayabilmesi için, öncelikle kart numarasını elde etmesi gerekmektedir. Kart numarası elde edilince, karta ilişkin tekil anahtar hesaplanır. Daha sonra terminal karttan bir sayı üretmesini ister, ayrıca terminal de bir rasgele sayı üretir. Terminal iki rasgele sayıyı arka arkaya yerleştirir, gizli asıllama anahtarıyla şifreler ve sonucu karta gönderir.



Şekil.6.9 : Karşılıklı Asıllama

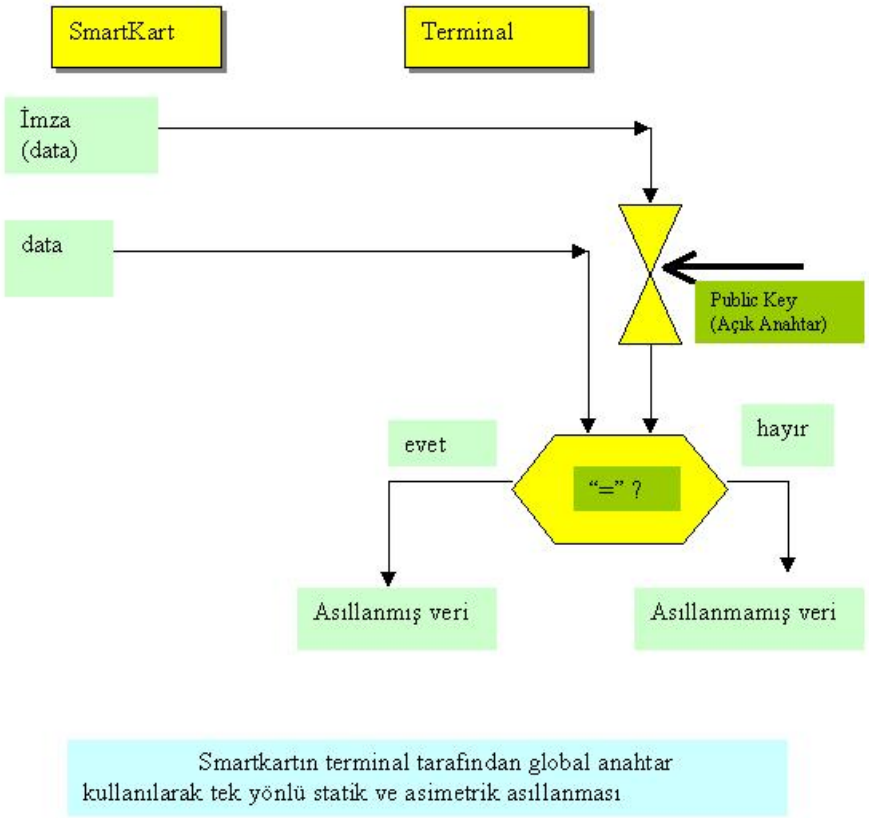
Kart şifrelenmiş bloğu çözer ve terminale göndermiş olduğu sayının mesaj içerisinde gelen sayıyla aynı olup olmadığını kontrol eder. Aynı ise kart terminalin anahtarın sahibi olduğunu anlar. Böylece terminal açısından kart asıllanmış olur. Sonraki aşamada kart iki sayıyı şifreler ve sonucu terminale gönderir. Terminal şifreli bloğu çözer, daha önce gönderdiği rasgele sayı ile gelen sayıyı karşılaştırır. Aynı iseler, terminal kartı asıllar. Böylece, kart ve terminal birbirini asıllamış olur.

6.5.1.2.3. Statik Asimetrik Asıllama

SmartKart mikroişlemcilerinin çok azı RSA hesaplamaları yapan aritmetik birimlere sahiptir. Bu böyle bir birimin fiyatı artırmasından kaynaklanmaktadır.

Bununla birlikte, ek asimetrik asıllama yöntemi arttırılmış koruma anlamına geldiğinden, saldırgan bir yerine iki kriptografik algoritmayı kırması gerekecektir, dolayısıyla genellikle bu ek güvenlik seviyesinin kartta olması istenir. Kart üzerinde RSA yapan bir aritmetik birimin olmaması problemi, kartın terminal tarafından statik asıllanmasıyla çözülebilir. Bu terminal tarafında bir imza (**signature**) rutinine gereksinim duyar. Terminal üzerinde ek bir aritmetik işlemci, toplam fiyat düşünüldüğünde önemsiz bir artış getirir, dolayısıyla bu çözüm pahalı özel smartkart işlemcileri yerine kullanılır. Ayrıca, dinamik asıllamayla kıyaslandığında, iki yerine bir asimetrik şifreleme gerektirdiğinden daha hızlıdır. Ama bu asıllama yönteminde indirgenmiş güvenlik demektir. Statik metot, tekrara (playback) koruma sağlamaz.

Kısaca metot aşağıdaki gibi çalışır. Personalize edildiğinde her smartkart kendi datasını tutar. Bu data örneğin kart numarası, kart sahibinin ismi ve adresi olabilir. Kart personalizasyonu esnasında gizli anahtar kullanılarak bu data için hiç değişmeyecek bir sayısal imza hesaplanır. Kart terminalde kullanıldığında, terminal imzayı ve imzalı datayı karttaki bir dosyadan okur. Terminal bütün kartlarda geçerli olan açık anahtara (public key) sahiptir, imzalı datayı açabilir ve gelen data ile karşılaştırabilir. Karttan gelen data ile imzalı datanın açılımı sonucunda elde edilen data aynı ise kart terminal tarafından asıllanır.



Şekil.6.10 : Global Anahtar Kullanılarak Asıllanma

Tekrara (playback) karşı bir korumasının olmaması yanısıra, yukarıdaki yöntemin diğer bir dezavantajı daha vardır. Global anahtarlar, imzaları üretmek ve kontrol etmek için kullanılmaktadır. Terminalde duran anahtar, açık (public) olduğundan hiçbir koruma sağlamamaktadır. Prensip olarak büyük sistemler bütün kartlar için aynı anahtarı kullanmamalıdır. Bu anahtar bir biçimde zarar görür ya da bilinirse, bu asıllama yöntemi değersiz olacaktır. Dolayısıyla her kart için ayrı anahtar-çifti üretilmelidir.

Bu durum, terminalin bütün kartlara ilişkin açık anahtarlarını (public key) tutmasını gerektireceğinden beraberinde terminal saklama kapasitesi problemini de getirecektir. 512-bit RSA anahtar kullanan 1 milyon kartlı bir sistemde terminalin bu durum için 64MB bellek harcaması gerekecektir (512-bit = 64 byte; 1M * 64 byte = 64 MB) . Bu terminallerin fiyatını önemli ölçüde arttıracaktır.

Simetrik metotlarda, master key den karta özgü anahtar üretmek kolay olmakla birlikte, asimetrik yöntemlerde böyle değildir. Açık anahtar (public key), kartta imza ile birlikte tutulur. Bu çözümde hala toplamda 64MB alana gereksinim duyulmakla birlikte bu alan 64-baytlık paketler halinde kartlar arasında dağıtılmıştır. Terminal açık anahtarı karttaki bir dosyadan okur ve imzayı kontrol etmek için kullanır. Bu durum sistemin bütün açık anahtarlarını terminalde tutması problemini çözer.

Bununla birlikte, yetkisiz birisi taklit bir karttaki datayı imzalamak için anahtar-çifti yaratabilir. Terminal açık anahtarı okuyacaktır ve kartın hakiki olduğu sonucuna varacaktır. Dolayısıyla bu metoda ek düzeltmeler yapmak gerekiyor. Her kartta saklı olan açık anahtar bir global gizli anahtarla imzalanmalıdır ve bu imza kartta tutulmalıdır. Terminal bu durumda aşağıdaki gibi çalışacaktır.

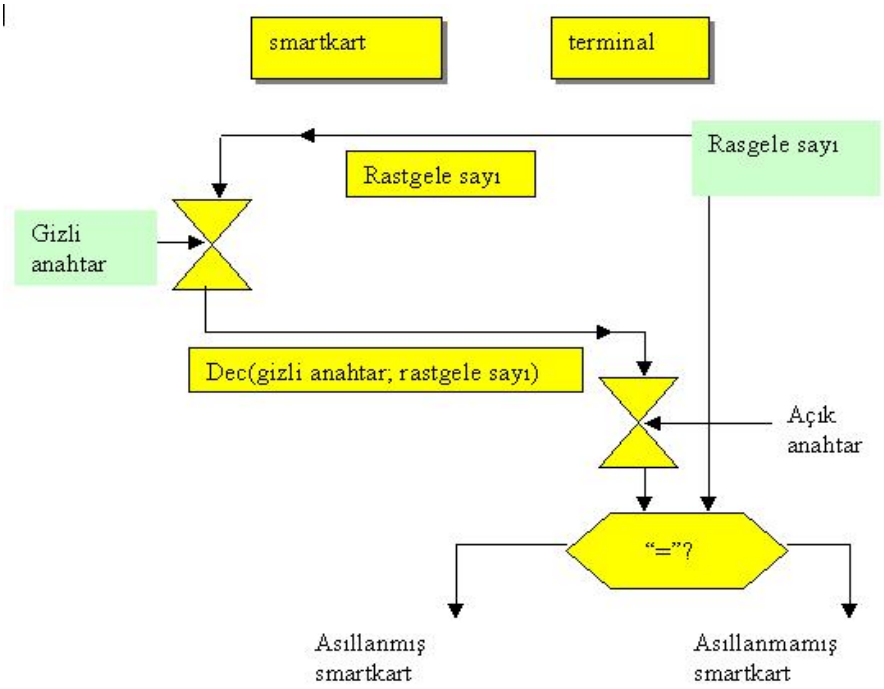
Öncelikle açık anahtarı karttan okur ve global açık anahtarı kullanarak okuduğu anahtarın doğruluğunu denetler. Eğer doğru ise, gerçek datayı okur ve datanın doğruluğunu kart üzerinde saklı olan açık anahtarla denetler.

Yukarıdaki yöntemler bir çok sistemde kullanılmaktadır. Bununla birlikte, asimetrik kripto-algoritma işlemcilerinin fiyatının düşmesiyle çekiciliklerini kaybedeceklerdir. En önemli dezavantajları tekrarlamaya (playback) karşı korumasız olmalarıdır.

6.5.1.2.4. Dinamik Asimetrik Asıllama

Dinamik asimetrik asıllama sistemde daha önce kaydedilmiş bir işlemin tekrarlanmasına (playback) karşı koruma getirmektedir. Klasik yaklaşım, kriptografik algoritma için başlangıç noktası hizmeti görecek bir rasgele sayı kullanır. Bununla birlikte,

bu tipteki bir asıllama yöntemi smartkartta bu işlemi yapabilecek bir aritmetik üniteye ihtiyaç duyar. Şekil.6.10 : Global açık anahtar kullanan tek yönlü asıllamayı göstermektedir.



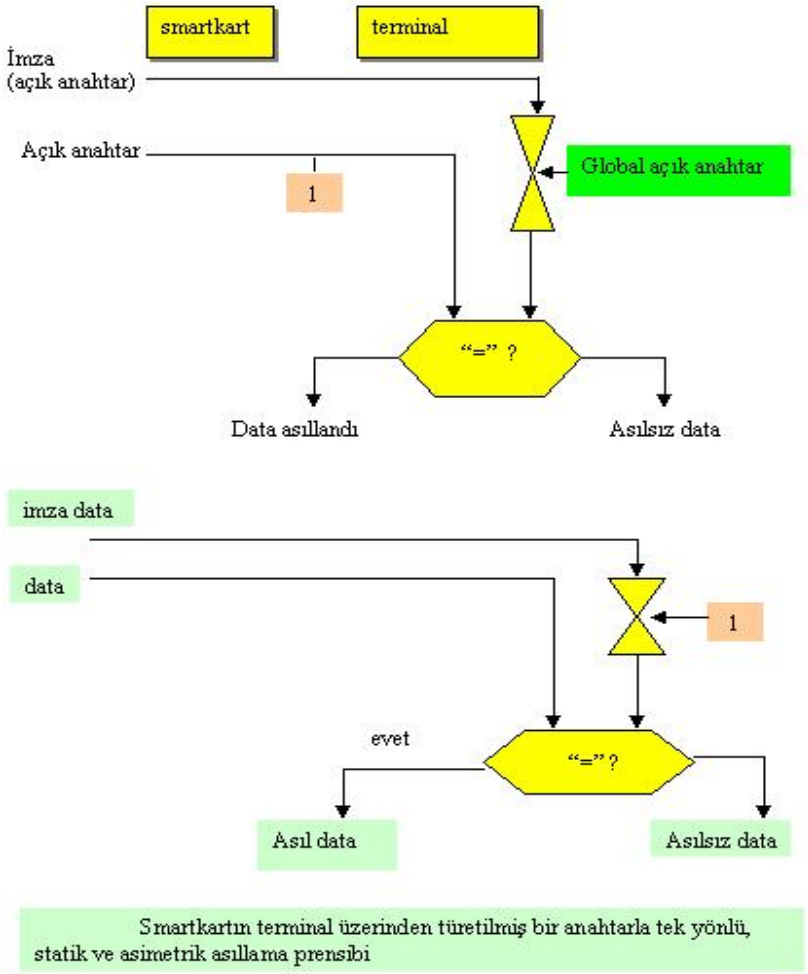
Smartkartın terminal üzerinden dinamik ve asimetrik asıllanması

Şekil.6.11 : Global açık anahtar kullanan tek yönlü asıllama

Simetrik metoda benzer biçimde, terminal rasgele bir sayı üretir ve bunu karta gönderir. Kart gizli anahtarı kullanarak bunu çözer ve sonucu terminale gönderir.

Buradaki çözme (decryption) işleminin nedeni asimetrik kripto-algoritmalarda imza üretimi esnasında, çözmenin her zaman gizli anahtar, şifrelemenin ise açık anahtar kullanılarak yapılmasından kaynaklanmaktadır.

Terminal gelen rasgele sayının şifrelenmesinde kullanılan global açık anahtarı tutmaktadır. Bu hesaplamanın sonucunda elde edilen değer karta gönderilen değere eşit ise kart terminal tarafından asıllanacaktır.



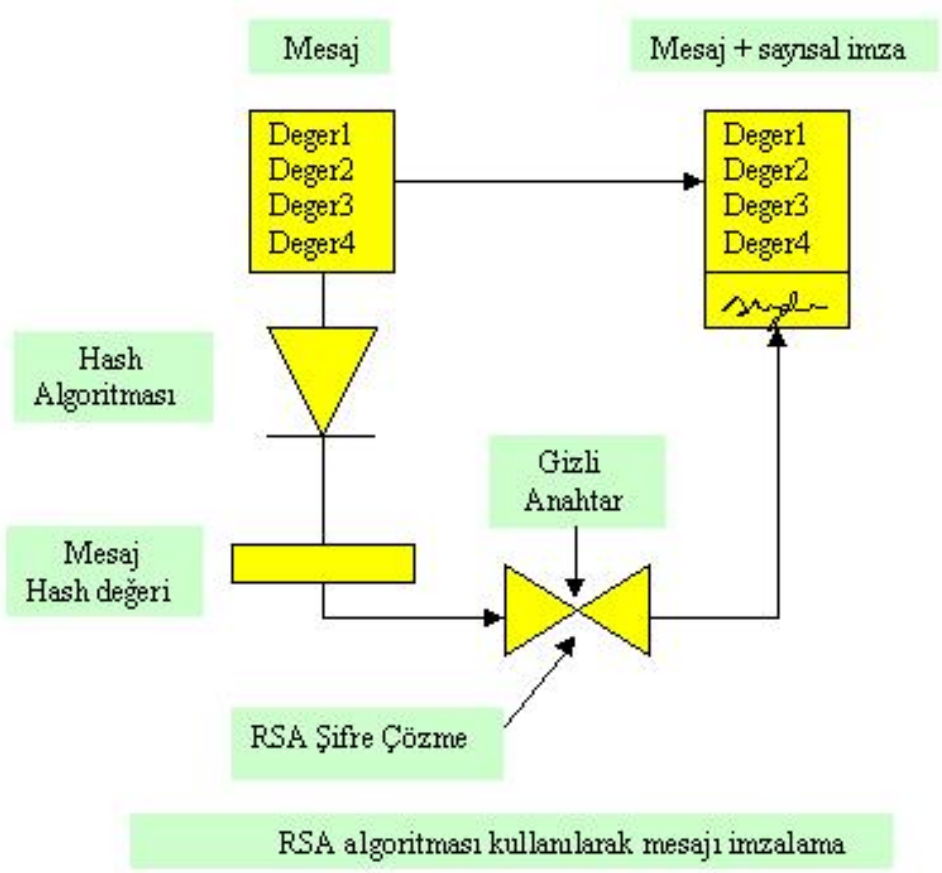
Şekil.6.12 : Türetilmiş Kartla Asıllama

6.5.1.3. Sayısal İmzalar

Sayısal imzalar, elektronik olarak iletilen mesaj veya dokümanları asıllamak için kullanılır. Sayısal imzalar mesaj ya da dökümlerin içeriğinin değişip değişmediğini sorgulamayı mümkün kılar.

Sayısal imza doğru olarak tek bir kişi tarafından yaratılır, fakat bütün alıcılar ya da asıl imzayı bilen kişiler tarafından kontrol edilebilir. Bu sayısal imzaların temel özelliğidir. Sadece tek kişi ya da smartkart bir dökümleri imzalayabilir, fakat herkes imzanın hakiki olup olmadığını denetleyebilir. Bundan dolayı, asimetrik kriptografik yöntemler sayısal imzalar için uygundur. Sayısal imza bir katar üzerinde elde edilen **MAK** (**M**esaj **A**sıllama **K**odu) gibi de düşünülebilir. Bu katarlar binlerce bayt uzunluğundadır. Hesaplama süresini kabul edilebilir sınırlar içinde tutmak için bütün katar üzerinde işlem yapmak yerine en başta bir hash değeri hesaplanır. Hash fonksiyonlar, basit anlamda, tek-yönlü data-dönüştürücü fonksiyonlardır. Bu dönüşüm tersinir değildir, yani asıl katar dönüştürülen katardan elde edilemez. Hash değer hesaplanması çok hızlı olduğundan, sayısal imza hesaplanmasında optimum çözümdür.

Sayısal imza üretmekte kullanılan rutin aşağıdaki gibidir. Mesaj, örneğin bir kelime–işlemci programı ile yazılmış bir döküm, Hash algoritması kullanılarak mesajdan hash değeri elde edilmesi için kullanılsın. Hash değeri, RSA gibi bir asimetrik algoritma kullanılarak şifrelenir. Bu şifrelemenin sonucunda mesajın sonuna eklenecek olan sayısal imza bulunmuş olur.



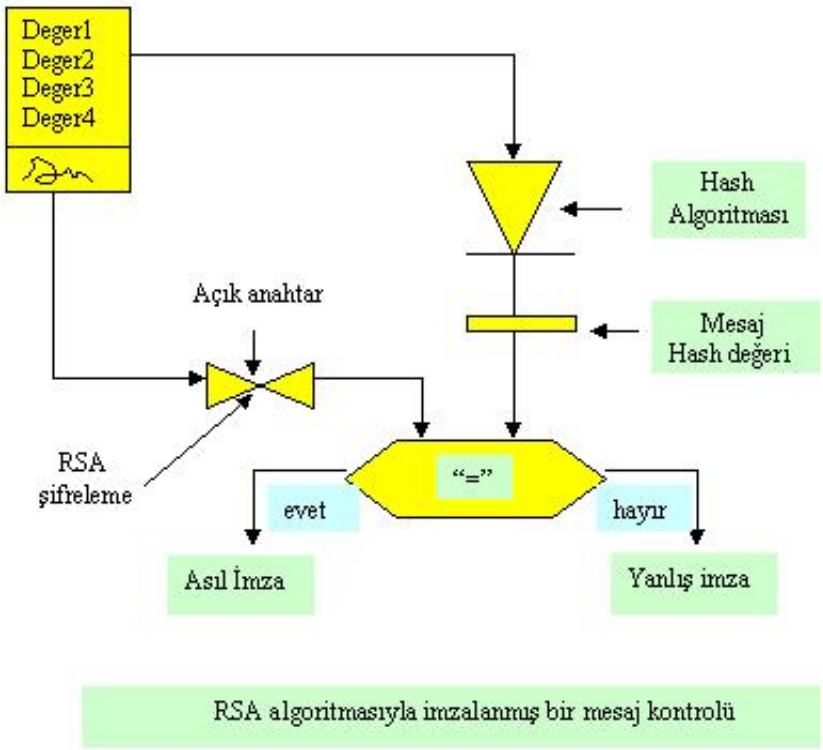
Şekil.6.13 : RSA ile Mesaj İmzalama

İmzayı içeren mesaj, alıcıya güvensiz bir kanaldan gönderilebilir, alıcı mesaj ve imzayı ayırır. Mesaj alıcıda, göndericide kullanılan aynı hash algoritması kullanılarak dönüştürülür. Sayısal imza RSA açık anahtarı kullanılarak elde edilir ve hash hesaplanması sonucunda elde edilen ile karşılaştırılır. İki değer aynı ise, iletim esnasında mesaj değişmemiştir, aksi halde mesaj ya da imza değişmiştir.

Yukarıdaki senaryoda kartın görevi basit biçimde açıklanabilir. En azından gizli RSA anahtarını saklar ve mesajın hash değerini çözer, böylece imzayı üretir. Hash değeri üretilmesi veya imza kontrolü PC tarafından yapılabilir.

En uygun durum, bununla birlikte, kartın mesajı alıp, hash değerini hesaplaması ve imzayla birlikte geri göndermesidir. İmza kontrolü kart tarafından da yapılabilir. Bu yöntem terminalin imza kontrolü yapmasından daha güvenli olmamasına rağmen, RSA anahtarları ve

hash algoritmaları kart değiştirince değişeceğinden terminal programlarında herhangi bir değişikliğe gerek olmamasından dolayı daha kullanışlıdır.



Şekil.6.14 : RSA ile imzalanmış Mesaj Kontrolü

7.BÖLÜM

7. ELEKTRONİK TİCARETTE GÜVENLİK

7.1. Tanıtım

Güvenlik ile ilgili çalışmalar çok yavaş bir tempo ile devam etmektedir. Diğer cephede (kötü amaçlı olanlar) çok büyük gelişme meydana gelirken, güvenlik konusunda ulaşılan nokta henüz arzulanan düzeye gelememiştir.

Güvenli kredi kart iletişimleri, halen E-ticaret kullanıcılarının bir numaralı endişe kaynağı olmaya devam etmektedir. Gerçi güvenli kart iletişimi çözümleri gelişmeleri sürdürmeye devam etmekte, kişiler çevrim-içi sipariş vermede giderek kendilerini daha güvenli olduklarını his etmekte iseler de, gelişme halen yeterli değildir.

Bunda en önemli neden güvenliğin, bir anahtarın kapalı ve aşık oluşuyla sağlanamayacak kadar karmaşık oluşudur. Her hangi bir kuruluş sadece bir güvenlik duvarı satın almak suretiyle tüm networkünü güvenli hale getiremez. Güvenli ağ katmansal bir yaklaşımla sağlanabilir ki, bu da pek çok güvenlik konusunu dikkate almayı gerektirir.

7.2 Güvenlikte Dikkate Alınması Gereken Noktalar

Nasıl zincirin gücü en zayıf halkası ile temsil edilir ise güvenliğin gücü de en zayıf noktası ile belirlenir.

7.2.1.Virüsler ve güvenlik duvarları

Dışarıdan gelecek virüs saldırıları için güvenlik duvarlarına ek olarak daima yeni üreyen virüslere karşı güncelleştirilen virüs savıcıları kullanmak gerekir. Bugün iki saat'te bir yeni bir virüs üretildiği düşünülür ise, bu güncelleştirmenin ne kadar önemli olduğu ortaya çıkacaktır.

Ayrıca kurumlar için güvenlik duvarlarının ne kadar önemli ve yararlı olduğu pek çok kişi ve kurum tarafından bilinmesine karşın, yapılan anket çalışmaları kurumların halen % 50 sinin güvenlik duvarına sahip olmadığını göstermiştir.

7.3. İletişim Güvenliği ve Ödeme Teknolojisi

7.3.1. Şifreleme

7.3.1.1. Tanıtım

Bugün için ağ üzerinde bilgi iletişiminde kullanılan en önemli silah şifrelemedir. Özellikle sağlık ile ilgili kayıtlar, finansal iletimler, araştırma ve geliştirme planları, askeri haberleşme gibi çok hassas konuların aktarımında veriler şifrelenerek karşı tarafa gönderilirler ve karşı kullanıcı tarafından çözümlenerek kullanılırlar.

7.3.1.2. Şifrelemenin Kullanımı

Şifreleme teknolojisi her hangi bir elektronik iletimin güvenli olarak yapılmasını sağlar. Ancak şifrelenmiş bir iletinin alıcısının, iletinin şifresini çözebilme yeteneğine sahip olması gerekir. Bu iş için geliştirilmiş pek çok metot bulunmaktadır.

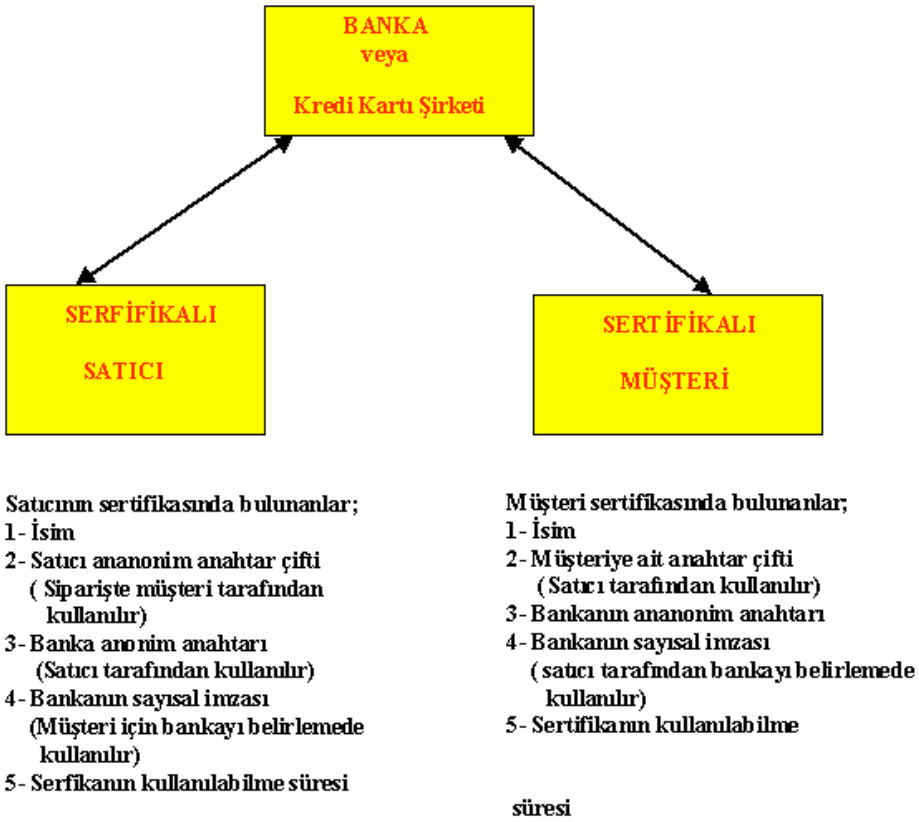
Aktarım Seviyesi	İki uzaktan erişim noktası arasındaki tüm trafik	VPV (Virtual Privat Network)
Oturum Seviyesi	Web tarayıcı ile Web sunucu arasında iletilen veriyi korumak üzere şifreleme uygulanmaktadır	• Netscape'nin SSL (Secure Sockets Layer) • EIT'nin SHHTP (Secure-HHTP)
Uygulama Seviyesi	Bir siparişte; tüm ödeme sürecinde , ödeme yöntemi dahil, işlemler şifrelenir	• MasterCard ve VİZA tarafından SET uygulaması • CyberCash Wallet
Kütük Seviyesinde	E-posta veya kütük aktarımı oturumu tümüyle şifrelenir	• Nortel'in Entrust • Phil Zimmermans'ın PGP (Prety Good Privacy)

Tablo.7.1. Şifreleme Seviyeleri

Tablo.7.1 incelendiğinde görülecektir ki, farklı bilgilerin farklı şifreleme seviyesinde gönderilmesi uygun olmaktadır. Örneğin; iki uzaktan erişim noktası arasında tüm ağ trafiğinin iletimi, iletişim (transmission) seviyesinde iletim uygulanması gerekmektedir ve buna örnek te **VPN** (**V**irtual **P**rivate **N**etwork) dir. E-Posta veya kütük aktarımı oturumunda, kütük seviyesinde şifreleme uygun bulunmaktadır ve buna örnek Phil Zimmerman'nın **PGP** (**P**retty **G**ood **P**rivacy) dir.

7.3.1.3. Gizli ve anonim anahtar algoritmaların birleştirilmesi

Anonim anahtar algoritmalar, gizli anahtar algoritmalarına kıyasla çok fazla bilgisayar kullanımını gerektirdiğinden, gerek şifreleme gerekse şifre çözümünde çok zaman kaybına neden olurlar. Buna karşılık anonim anahtar algoritmalar, gizli anahtar algoritmalara kıyasla çok daha güvenilirdirler. Bu nedenle pek çok anahtar algoritma bu iki yöntemi birlikte kullanmaktadır.Şekil-7.1 de bu uygulama görülmektedir.



Şekil.7.1 : Gizli ve Anonim Anahtar Algoritmaların Kombinasyonu

7.4.Oturum Temelli Güvenlik

Tarayıcı ve sunucular arasındaki iletişimi şifreleyen oturum temelli şifreleme, güvenli bir çevrim içi siparişin temelini oluşturur. Bu güvenlik yöntemi içinde en belli başlı olanları: Netscape'nin **SSL** (**S**ecure **S**ocket **L**ayer-SSL) ve EIT'nin **SHTTP** (**S**ecure-**HTTP**) ve Microsoft'un **PCT** (**P**rivate **C**ommunication **T**echnology) dir ve hepsinde SSL kullanılmaktadır.

7.4.1. SSL (Secure Socket Layer)

SSL (güvenilir soket katmanı) network üzerindeki bilgi transferi sırasında güvenlik ve gizliliğin sağlanması amacıyla Netscape tarafından geliştirilmiş bir güvenlik protokolüdür. 1996 yılında 3.0 versiyonunun çıkarılmasıyla hemen bütün Internet tarayıcılarının (Microsoft Explorer, Netscape Navigator vb) desteklediği bir standart haline gelmiş ve çok geniş uygulama alanları bulmuştur.

SSL teknolojisi TCP/IP protokolü üzerinden çalışan, web sunucusu ve web tarayıcısı arasındaki tüm bilgi akışını koruyan bir güvenlik protokolüdür. Bütün popüler web tarayıcılarda ve web sunucularda uygulanmaktadır. Bugünün web üzerinden elektronik ticaret ve elektronik iş uygulamalarında önemli bir rolü vardır. SSL gönderilen bilginin kesinlikle ve sadece doğru adreste deşifre edilebilmesini sağlar. Bilgi gönderilmeden önce otomatik olarak şifrelenir ve sadece doğru alıcı tarafından deşifre edilebilir. Her iki tarafta da doğrulama yapılarak işlemin ve bilginin gizliliği ve bütünlüğü korunur. SSL protokolü iki taraf arasında güvenli ve gizli iletişimin sağlanmasında elektronik kimlik belgelerini kullanır. SSL bağlantısı üzerinden gönderilen veriler üçüncü şahıslar tarafından bozguna uğratıldığında, bundan tarafların anında haberi olur. Bir web sunucusunun kullanıcılarıyla SSL bağlantısı sağlayabilmesi için öncelikle sunucu tarafında bir sunucu e-kimliğinin bulunması gereklidir. SSL ile güvenliği sağlanmış bir siteye bağlanan kullanıcı sitenin URL adresinin "**https:**" ile başladığını görür. Bundan sonraki aşamalarda sunucu kendi e-kimliğini kullanıcıya göndererek kendini tanıtır. Eğer sunucunun e-kimliği geçerliyse ve kullanıcının tarayıcısında sunucuya e-kimlik veren Onay Kurumunun e-kimliği tanımlıysa tarayıcı kullanıcıya güvenilir bir siteye bağlandığına dair bilgi verir. Daha sonra SSL bağlantısı kurulur ve sunucu-tarayıcı arasındaki tüm veriler üçüncü şahısların mesajı okumasını önlemek amacıyla şifrelenir, mesaj içeriğinin

yolda herhangi bir şekilde değiştirilme olasılığına karşı olarak da sayısal imza ile imzalanır ve şifreli mesaj imzasıyla birlikte gönderilir veya alınır. **"SSL / Sunucu Kimlik Dogrulamasi"** olarak adlandırılan bu işlemlerin amacı kullanıcıya bağlandığı sitenin gerçekten bağlandığını düşündüğü site olduğunun ve sunucuya gönderilen bilgilerin gerçekten de sadece o sunucu tarafından okunabileceğinin ispatinin sağlanmasıdır. Kullanıcı bundan emin olmak için SSL bağlantısı süresince sunucudan gelen her bilgiyi web sayfasının güvenlikle ilgili özelliklerine bakarak kontrol etmelidir. Web sayfalarının güvenlik bilgileri sunucunun e-kimliğini kontrol imkanı sağlar. Eğer yabancı veya farklı bir sunucunun kimliğiyle karşılaşılırsa bağlanılan sunucu bağlanıldığı sanılan sunucu değildir ve iletişimin güvenliği tehdit altındadır.

SSL'in sağladığı imkanlardan bir diğeri de kullanıcı e-kimliğinin sunucuya gönderilerek kullanıcının kendisini sunucuya tanıtmasıdır. **"SSL / Kullanici Kimlik Dogrulamasi"** olarak adlandırılan bu işlemde ise sunucu kendini kullanıcıya tanıttığı gibi kullanıcıdan da kendisini tanıtmasını ister. Bunun için kullanıcının bir e-kimliğinin olması gerekmektedir. Sunucu tarafında yapılacak bazı ayarlar ile e-kimliği olan hangi kullanıcıların siteye bağlanmalarına izin verileceği tanımlanabilir.

7.4.1.1. SSL'in Gücü

Veri akışında kullanılan şifreleme yönteminin gücü kullanılan anahtar uzunluğuna bağlıdır. Anahtar uzunluğu bilginin korunması için çok önemlidir. Örneğin; 8 bit üzerinden bir iletimin çözülmesi son derece kolaydır. Bit, ikilik sayma düzeninde bir rakamı ifade eder. Bir bit, 0 veya 1 olmak üzere 2 farklı değer alabilir. 8 bit ise sadece 28=256 olası farklı anahtar içerir. Bir bilgisayar bu 256 farklı olasılığı sıra ile inceleyerek bir sonuca ulaşabilir. SSL protokolünde 40 bit ve 128 bit şifreleme kullanılmaktadır. 128 bit şifrelemede 2128 değişik anahtar vardır ve bu şifrenin çözülebilmesi çok büyük bir maliyet ve zaman gerektirir. Kötü niyetli bir kişinin 128 bit'lik şifreyi çözebilmesi için 1 milyon dolarlık yatırım yaptıktan sonra 67 yıl gibi bir zaman harcaması gerekir. Bu örnekten anlaşıldığı gibi SSL güvenlik sistemi tam ve kesin bir koruma sağlar.

7.4.2. SET (Secure Electronic Transaction)

SET banka kartları ve ödemeler ile ilgili bilgilerin güvenliğini sağlamak amacıyla Visa, Mastercard, Microsoft, Netscape, GTE, IBM, SAIC, Terisa Systems

ve Verisign'ın katılımıyla oluşan bir konsorsiyum tarafından geliştirilmiştir. SET uyumlu ilk alışveriş, 18 Temmuz 1997'de San Francisco'da yapılan tanıtımla İspanya ve Singapur'da bulunan sanal mağazalardan gerçekleştirilmiştir. **Garanti Bankası** Şubat 1998'de gerçekleştirdiği **SET** uyumlu alışverişle, bu protokolü kullanmaya başlayan Dünya'da yedinci, Avrupa'da dördüncü ve Türkiye'de ilk kuruluş olmuştur.
SET üzerindeki çalışmalar yoğun bir şekilde devam etmektedir. Örneğin akıllı kartlar ve elektronik ödemenin İş-ten-işe (**B2B** - business-to-business) uygulamasında SET'in destek vermesi çalışmaları sürmektedir.

SET protokolünde alışveriş, sanal cüzdan ve sertifika aracılığı ile daha güvenli bir ortamda gerçekleştirilir. SET, alışveriş işlemi sırasında ödeme bilgisi gizliliğini, kart kullanıcısının gerçek kart sahibi olduğunu ve işyerinin banka ile anlaşmalı bir işyeri olduğunu garantiler.
SET sisteminde provizyon işlemi müşteri alışveriş seçimini yaptıktan sonra müşterinin sanal cüzdanı ile mağazanın Sanal POS'unun (**V-POS**) birbirlerinin gerçekliklerini dijital sertifikalar aracılığıyla kontrol etmeleri ile başlar. Mağazanın Sanal POS yazılımı sipariş tutarını ve sanal cüzdanda bulunan ve alışveriş için seçilen kredi kartının sertifika bilgilerini bankaya iletmesi ile devam eder. Banka yapılan alışverişin içeriğini (malın ne olduğu, kaç tane alındığı vb.) görmeksizin provizyon verir. Müşterinin kredi kartı bilgilerini görmeyen sanal mağaza ise bankadan gelecek onayı bekler. Onayı aldıktan sonra da ürünü alıcısına gönderir.

SET sistemi de SSL'de olduğu gibi kullanıcı, işyeri ve banka arasındaki veri akışı sırasında bilgilerin şifrelenerek gönderilmesi esasına dayanır. Bu sistemden faydalanabilmek için kullanılmak istenen kredi kartının SET uyumlu olması gerekir. SET protokolünü kullanmak isteyen kredi kartı sahipleri iki ön koşulu yerine getirmek zorundadırlar:

Öncelikle kullanmak istedikleri her bir kredi kartı için sertifikasyon kurumu (**Certificate Authority**) ayrı birer SET sertifikası almalıdırlar. Ardından kart sahipleri yine kredi kartı veren bir bankadan sanal cüzdan adı verilen bir programı alıp bilgisayarlarına yüklemeli ve bu yükleme sırasında SET sertifikalı kredi kartlarını programa tanıtmalıdırlar. SET uyumlu alışverişler sanal cüzdanın yüklü olduğu bilgisayar kullanılarak SET uyumlu mağazalardan yapılabilecektir. Sanal cüzdan

programı en fazla üç kez yüklenmek üzere yazıldığından en fazla üç bilgisayarda kullanılabilecektir. SET protokolünün SSL'e göre çok daha yüksek denebilecek güvenliğine rağmen yeterince yaygınlaşamaması sanal cüzdanın mobilitesinin olmamasına bağlanabilir. Bu yüzden Garanti Bankası sistemi SET uyumlu olmasına karşın SET protokolünü tam olarak uygulamamaktadır.
Sanal mağazalar ise Sanal **POS** (**P**oint **O**f **S**ale) olarak adlandırılan V-POS yazılımını yükledikten sonra bir sertifikasyon kurumundan (http://www.verisign.com, http://www.gte.com) dijital bir sertifika alarak alışverişlerin güvenliğini sağlarlar.

SET ile gerçekleşen alışveriş sırasında gerçekleşen işlemler sırasıyla aşağıdaki gibidir:

SET protokolü, kart sahibi İnternet üzerinde araştırmasını tamamlayıp seçimini yaptıktan ve siparişini verdikten sonra devreye girmektedir. SET işleminin başlamasından önce kart sahibi sipariş formunu doldurmuş ve onaylamış olmalıdır. Kart sahibi ayrıca kart türünü de seçmiş olmalıdır.

1. Kart sahibinin yazılımı satıcı firmaya kullanılacak kredi kartını belirten ve ödeme altyapısını sağlayan kuruluşun sertifikalı açık anahtarının kopyasını isteyen bir mesaj gönderir.

2. Satıcı firmanın yazılımı mesajı aldığında, sadece o mesaja özel bir işlem tanımlama numarası belirler. Daha sonra bu özel tanımlama numarasıyla beraber kart sahibine satıcı firmanın açık anahtarını ve ödeme altyapısını sağlayan kuruluşun (genelde bankalar) onaylı açık anahtarını gönderir.

3. Kart sahibinin yazılımı satıcı firmanın ve ödeme altyapısını sağlayan kuruluşun sertifikalarını kontrol eder ve sipariş sürecinde kullanmak üzere bunları kaydeder. Kart sahibinin yazılımı sipariş bilgisini ve ödeme talimatlarını oluşturur. Yazılım satıcı firma tarafından belirlenen özel tanımlama numarası ile sipariş bilgisini ve ödeme talimatlarını ilişkilendirir. Bu tanımlama daha sonra satıcı firma tarafından ödeme talebi yapıldığında, ödeme altyapısını sağlayan kuruluş tarafından sipariş bilgisini ve ödeme talimatlarını ilişkilendirmede kullanılacaktır.

4. Kart sahibinin yazılımı sipariş bilgisi ve ödeme talimatları için bir dijital imza oluşturur. Yazılım daha sonra ödeme altyapısını sağlayan kuruluşun açık anahtarını kullanarak dijital olarak imzalanan ödeme talimatlarını şifreler. Son olarak yazılım imzalanmış ve şifrelenmiş sipariş bilgisini ve ödeme talimatlarını bir mesajla satıcı firmaya gönderir.

5. Satıcı firmanın yazılımı siparişi alır ve kart sahibinin açık anahtarı üzerindeki dijital sertifikayı kontrol eder. Bundan sonra gene bu açık anahtarı kullanarak siparişin gerçekten kart sahibinden geldiğinden ve mesajın gönderim esnasında değiştirilmediğini teyit eder (Satıcı firma ödeme talimatları ödeme altyapısını sağlayan firmanın açık anahtarı ile şifrelendiği için deşifre edemez).

6. Bu işlemlerin ardından satıcı firmanın yazılımı ödeme onayı istenmesi de dahil olmak üzere siparişle ilgili işlemlere başlar (lütfen 9. Maddeye bakınız)

7. Sipariş bilgisi işleme alındıktan sonra, satıcı firmanın yazılımı bir cevap mesajı hazırlar ve dijital olarak imzalar (satıcı firmanın onaylı açık anahtarı ile). Kart sahibinin siparişinin alındığının ve işleme konulduğunun bildirilmesi amacıyla hazırlanan cevap mesajı kart sahibine gönderilir.

8. Kart sahibinin yazılımı satıcı firmadan cevap mesajını aldığı zaman dijital sertifikasını kontrol eder. Bunun ardından bu mesajı kullanarak kart sahibine bir teyit mesajı gösterir veya siparişin durumunu günceller.

9. Kart sahibinden gelen siparişlerin işleme konulması esnasında (lütfen 6. maddeye bakınız) satıcı firmanın yazılımı ödenmesi talep edilen tutarı, sipariş bilgisindeki işlemi belirleyen özel tanımlama numarasını ve işlemle ilgili diğer bilgileri içeren bir ödeme onay talebini hazırlar ve bu mesajı dijital olarak imzalar. Ardından bu talep ödeme altyapısını sağlayan kuruluşun açık anahtarı kullanılarak şifrelenir. Satıcı firmanın ödeme onay talebi ve kart sahibinin şifrelenmiş ödeme talimatları ödeme altyapısını sağlayan kuruluşa gönderilir.

10. Ödeme altyapısını sağlayan kuruluş onay talebini aldığı zaman satıcı firmadan gelen onay talebini kendi gizli anahtarını kullanarak deşifre eder. Ardından satıcı

firmanın açık anahtarı üzerindeki dijital sertifikayı kontrol eder ve sertifikanın geçerlilik süresinin dolup dolmadığını belirler.

11. Ödeme altyapısını sağlayan kuruluş kart sahibinin satıcı firmadan gelen onay talebiyle birlikte gönderilen ödeme talimatlarını kart sahibinin açık anahtarını kullanarak deşifre eder. Ardından bu açık anahtarı kullanarak kart sahibinin ödeme talimatları üzerindeki dijital imzasını kontrol eder ve böylece ödeme talimatlarının kart sahibi tarafından imzalandığından ve iletim esnasında değişikliğe uğramadığından emin olur.

12. Ödeme altyapısını sağlayan kuruluş satıcı firma tarafından gönderilen işlem tanımlayıcısı ile kart sahibinden gelen ödeme talimatlarındaki tanımları karşılaştırarak her ikisinin de aynı olup olmadığını kontrol eder. Kontrolün ardından ödeme altyapısının sağlayan kuruluş, kredi kartını veren bankaya Internet üzerinden çalışmayan bir ödeme sistemiyle bir onay talebi gönderir.

13. Kartı veren banka onay talebini işleme alır ve ödeme altyapısını sağlayan kuruluşa güvenli ödeme sistemi aracılığıyla bir cevap gönderir.

14. Onay cevabını aldıktan sonra ödeme altyapısını sağlayan kuruluş kartı veren bankanın cevabını ve onaylı açık anahtarını içeren bir onay cevap mesajı yaratır ve dijital olarak imzalar. Cevap satıcı firmanın açık anahtarını kullanarak şifrelenir ve satıcı firmaya gönderilir.

15. Satıcı firmanın yazılımı ödeme altyapısını sağlayan kuruluştan onay cevabını aldığı zaman kendi gizli anahtarıyla deşifre eder. Ardından ödeme altyapısını sağlayan kuruluşun açık anahtarı üzerindeki dijital sertifikayı kontrol eder ve bu açık anahtarı kullanarak ödeme altyapısını sağlayan kuruluşun onay cevap mesajındaki dijital imzayı kontrol eder. Satıcı firmanın yazılımı, sipariş tamamen yerine getirildikten sonra ödeme talebinde bulunulabilmesi için (gün sonu işlemi ile) bu onay cevap mesajını kaydeder.

16. Satıcı firma onay cevabını aldıktan sonra kart sahibinin siparişi tamamlar ve ilgili ürünü sevk eder veya söz konusu hizmeti verir.

17. Siparişi yerine getirdikten sonra satıcı firma ödeme talebinde bulunur (Siparişin tamamlanması esnasındaki gecikmeler onay talebi ile ödeme talebi mesajları arasında önemli zaman aralıkları oluşmasına yol açabilir).

18. Ödeme talebinde bulunmak için satıcı firmanın yazılımı işlemin nihai tutarını, sipariş bilgisindeki işlem tanım numarasını ve işlem hakkındaki diğer bilgileri içeren bir gün sonu işlemi oluşturur ve dijital olarak imzalar. Bu talep ödeme altyapısı sağlayan kuruluşun açık anahtarı ile şifrelenir ve ödeme sağlayan kuruluş gönderilir.

19. Ödeme altyapısını sağlayan kuruluş gün sonu işlemi talebini aldığı zaman, kendi açık anahtarını kullanarak talebi deşifre eder. Daha sonra satıcı firmanın açık anahtarını kullanarak gün sonu işlemindeki dijital imzayı kontrol eder. Satıcı firmadan gelen gün sonu işlemiyle, daha önce işleme alınan onay talebini karşılaştırır ve bir tahsilat talebi oluşturarak bunu kredi kartını veren bankaya güvenli ödeme sistemiyle gönderir.

20. Ödeme altyapısını sağlayan kuruluş kendi onaylı açık anahtarını içeren bir gün sonu cevap mesajı oluşturur ve bunu dijital olarak imzalar. Bu cevap satıcı firmanın açık anahtarı ile şifrelenerek satıcı firmaya gönderilir. Bu mesaj sayesinde gün sonu işleminin ödeme altyapısını sağlayan kuruluş tarafından alındığını ve işleme konulduğunu satıcı firmaya bildirir.

21. Satıcı firmanın yazılımı ödeme altyapısını sağlayan kuruluştan gün sonu işleminin cevabını alınca, mesajı kendi gizli anahtarını kullanarak deşifre eder. Ardından ödeme altyapısını sağlayan kuruluşun açık anahtarı üzerindeki dijital sertifikayı kontrol eder ve yine bu açık anahtarı kullanarak ödeme altyapısını sağlayan kuruluşun dijital imzasını kontrol eder. Son olarak satıcı firmanın yazılımı gün sonu işlemi cevabını yapılan ödemeler için gönderilen gün sonu talep mesajları ile mutabakat için kaydeder.

7.4.2.1. SET 'in Güvenlik Seviyesi

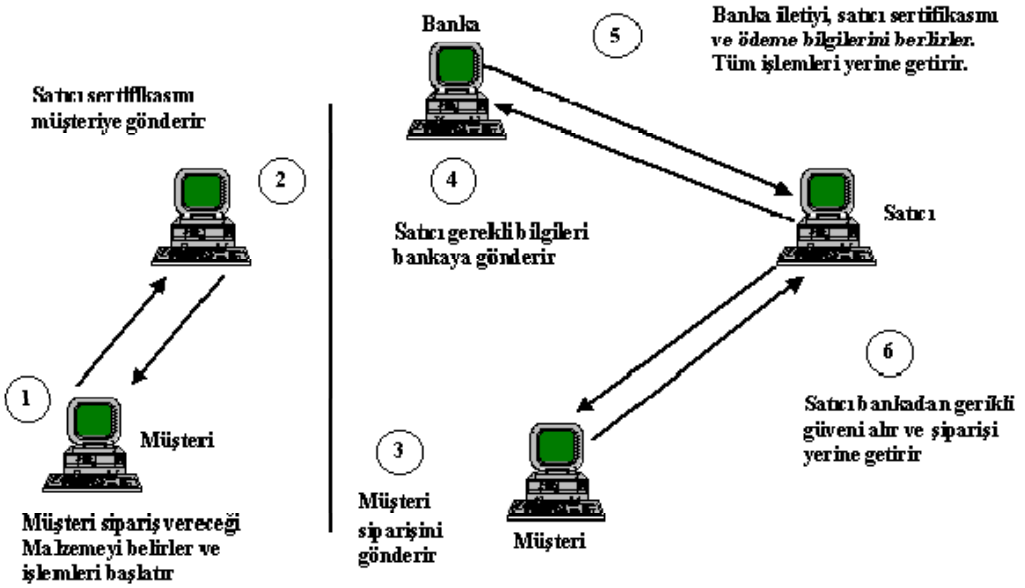
SET aşağıda verilen çeşitli seviyelerde güvenlik sağlar.

- Bir siparişin gönderilmesi ve yanıtının gelmesi işlemlerini tümleştirerek yerine getirir,
- Sipariş gizlidir ve müşteri hiç bir zaman müşteri numarasını bilmez,
- Satıcı ödemenin yapılacağı hakkında güven altındadır. Zira bu ödeme üçüncü parti (örneğin banka) tarafından güven altına alınmaktadır.

SET iletiminin yapılabilmesi için uygulamaya giren tüm partilerin **SET yazılımını** sağlayıp sistemlerine yüklenmesi gerekmektedir.

Müşteri her hangi bir bankada bir hesap açtığında bir sertifika alır ve kendisine iki anahtar sağlanır. Bunlardan biri şifreleme, sayısal imza ise sipariş için kullanılırken, diğeri kişilik belirleme ve ödeme bilgileri için kullanılır.

Şekil.7.2 : de görüldüğü gibi banka veya kredi kart şirketi, müşteri ve satıcı için elektronik ortamda (**kağıt üzerine yazılı olmayan**) sertifikalar üretir.

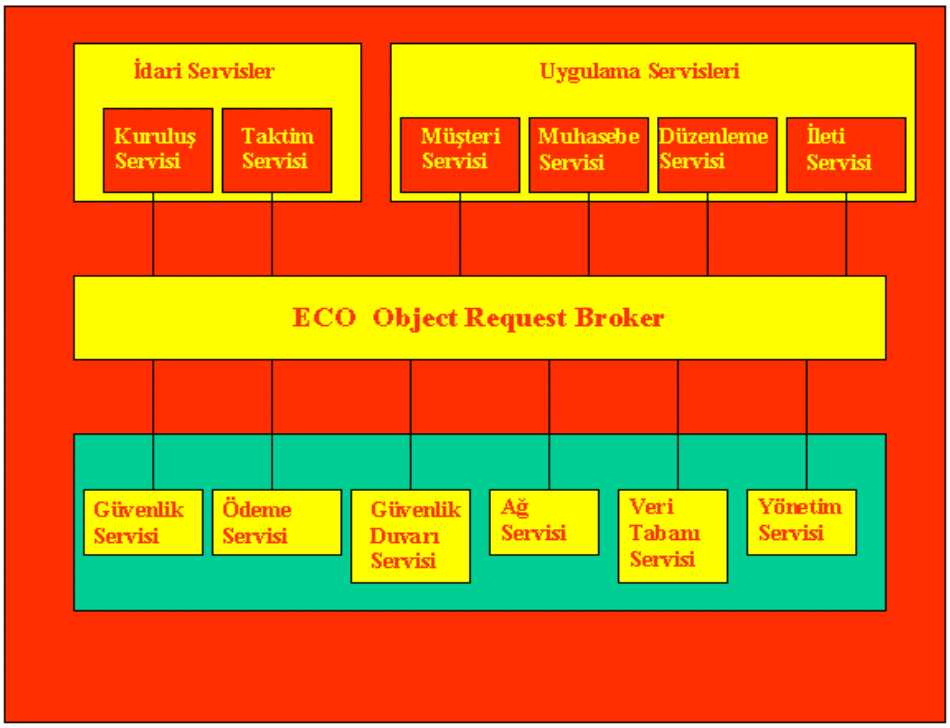


Şekil.7.2 : SET Hesabının Oluşturulması

Sertifikaların kopyaları her siparişte alıcı ve satıcı arasında karşılıklı olarak birbirlerine iletilir. Bu bilgiler şifrelidir ve ancak yetkililer tarafından okunabilir.

Kullanıcı istediği parçayı seçer ve sipariş eder, satıcı daha önce edindiği sertifikayı müşteriye iletir ve müşteri böylece satıcıyı belirler. Müşteri siparişini birinci adımda gizli anahtar ile şifreler ve daha sonra müşterinin anonim anahtarını kullanarak ikinci şifrelemeyi gerçekleştirir. Bu durumda satıcı, sadece müşteri siparişinin şifre çözümünü yapabilir. Doğal olarak siparişin parasal kısmı, kredi kart numarası da şifrelenmiş olarak aktarılır.

Satıcı kendi özel anahtarını kullanarak gizli anahtarının ve daha sonra da siparişin şifresini çözer. Bu işlemleri izleyerek, satıcı siparişin kopyası ile birlikte ödeme bilgilerini bankaya iletir.
Banka önce müşteriye ait bilgileri kontrol eder ve daha sonra bilgileri açar. Sonra müşterinin ödeme bilgilerini kontrol eder ve satıcıya gerekli garantiyi verir. Satıcı bunun üzerine müşterisinin siparişini yerine getirir. Şekil.7.3 : de bu uygulama görülmektedir.



Şekil.7.3 : Güvenli Bir SET Uygulaması

Ağ Güvenliği ile kaynaklar;

Neyi ?	Nereden ?
Pacific Bell Telecommunity Security Checklist	http://www.pacbell.com
Open Computing Security Survay Result	http://www.wcmh.com
Forum of Incident Response and Security Teams	http://www.first.org
Security tools for DOS , Macintosh and UNIX	http:// www.ce.perdue.edu
Privacy issues	http://www.ce.perdue.edu
SHTTP	http://www.eit.com
Security Dynamics	http://www.securid.com

Tablo.7.2: Ağ Güvenlik Kaynakları

8.BÖLÜM

8. İNTERNETTE GÜVENLİK

8.1. İnternette Güvenliğin Önemi

İnternetin hızla yaygınlaşması ve gelişmesi evlerde, okullarda, kamu kuruluşlarında, finansal işlemlerde, kısaca hayatın her alanında radikal değişiklikleri de beraberinde getirmiştir. İnternet kullanımının artmasıyla beraber başta finansal kurumlar olmak üzere birçok kuruluş ticari işlemlerini İnternet'e taşımaya başlamıştır. Böylece bilgilerin herkese açık bir ağ üzerinde dolaşmasının yarattığı haklı tedirginlik gündeme gelmiş, bu konu yoğun bir şekilde tartışılmaya başlanmıştır.

İnternet'te güvenlik konusu iki temel başlık altında incelenebilir:

8.1.1. Yazılım ve Ağ Altyapısı

Yazılım ve ağ altyapısından dolayı meydana gelebilecek sorunlar. Bu kapsamda İnternet'e bağlandığınız servis sağlayıcıların ve kişisel bilgilerinizi verdiğiniz kurumların bu bilgileri sakladıkları ortamlar ve iç güvenlikleri ile bilgisayarların İnternet'e bağlı oldukları sürede maruz kaldıkları riskler ve önlemler yer alır. ISS'ler, finansal kurumlar, bankalar ve hizmet verebilmek için duyarlı bilgiler toplaması gereken siteler (elektronik ticaret siteleri, sağlık danışmanlığı hizmeti veren siteler vb.) müşterilerinden aldıkları kişisel bilgilerin güvenliğini sağlamak için gereken altyapıyı oluşturmakla yükümlüdürler. Bu tür kuruluşlar şirket içi güvenlik prosedürleri, yetki ve sorumluluk sınıfları ve ulaşım izinleri, detaylı loglama gibi geleneksel tedbirlerin yanı sıra, kendi iç ağlarını İnternet yolu ile gelecek tehlikelere karşı korumak için firewall çözümleri, virus gateway çözümleri gibi teknolojik önlemler de almaktadırlar. Bunlara ilave olarak güvenilir kurumlar, veri tabanlarında tutulan bilgilerden önemlilerini şifrelenmiş olarak tutmakta, bu bilgileri şifrelenmiş olarak şirket içindeki işlemlerde kullanmaktadır.

8.1.2. Veri Akışı

İnternet üzerinde veri akışı sırasında, bilgilerin çeşitli teknik açıklar değerlendirilerek kötü amaçlı kullanım için ele geçirilme tehlikesi Genel bir İnternet kullanıcısının verebileceği bilgilerin arasında kendi güvenliği ve mahremiyeti açısından sakıncalı olabilecek kredi kartı bilgileri, kullanıcı isimleri şifreler, adres ve telefon numaraları bulunmaktadır. Bu bilgiler

genellikle, elektronik ticaret veya finansal işlemler (örneğin bireysel bankacılık) sırasında İnternete açılır, ve ele geçirilme olasılıkları belirir. Elektronik ticaret ve finansal işlemlerin yürütüldüğü siteler İnternet'ten bilgi alışverişini şifreleyerek gerçekleştirdikleri için genel olarak güvenlidirler. Şifrelenmiş bilgi İnternet üzerinde iletilirken ele geçirilse bile, şifrenin kırılması çok büyük bir yatırım ve oldukça uzun bir zaman dilimi gerektirdiğinden güvenli olduğu kabul edilebilir Şifreleme amacıyla yaygın olarak SSL güvenlik standardı kullanılmaktadır. Ayrıca güvenliğin sağlanması amacıyla yaygın olmamakla birlikte SET protokolü adı verilen ve güvenliği bir kat daha arttıran bir sistem de kullanılmaktadır.

İnternet kullanımında sisteminizin güvenliğini artırmak için aşağıdaki ipuçları yardımcı olacaktır:

· En önemli nokta dikkatli olmaktır. Tanımadığınız kişilerden e-mail veya chat gibi güvensiz yollarla dosyalar almayınız.

· Güvenilir ve tanınır siteler haricindeki sitelerden dosyalar indirmeyiniz ve bu tür dosyaları bilgisayarınızda çalıştırmayınız.

· Bilmediğiniz ve güvenliğinden emin olmadığınız sitelere kişisel bilgilerinizi kesinlikle vermeyiniz.

· İnternet üzerindeki güvenlik ile ilgili konu ve bilgileri yakından takip ediniz. İşletim sisteminizi ve programlarınızı korumak için yeni teknolojileri kullanınız.

8.2. Internet Güvenliğine Detaylı Bir Bakış

Firewall' lar güvenlik mekanizmalarının ilk aşamalarıdır, ancak bilgilerimizin duyarlılığı arttıkça bununla beraber şifreleme ve kullanıcı doğrulama (**authentication**) tekniklerinden de yararlanmak gerekmektedir.

Ağ yöneticileri için ağın güvenliği, başkaları tarafından – özellikle dışarıdan ağa dahil olacaklar tarafından - zarara uğratılmaması, her zaman için en önemli, en zor ve en çok sıkıntı yaratan konulardan biri olmuştur. Bir yolunu bulup ağa dahil olmuş kişiler tarafından ağa zarar verilmesi, belki bundan daha da önemli olmak üzere, gizli bilgilere ulaşması, ağ yöneticileri için korkuların başında gelmekte idi. Bugün için ise durum daha da kötüdür. Bahsetmiş olduğumuz korkular, internet kavramı bu kadar yaygın değilken ve kuruluşlar bu ortama dahil olmak konusunda bu kadar istekli değilken yaşanan korkulardı. Oysa internet

denilen ve çok güçlü olması gerekmeyen bir bilgisayar ile basit bir programdan başka bir şey istemeyen ortamın insanların kullanımına açılması, bunun kuruluşlar tarafından da çok cazip bir ortam olarak görülmesi, ağ yöneticileri için yeni kaygıların başlangıcı olmuştur. Bu gün dünyanın hemen her tarafındaki şirketler internete dahil olmak , Web sayfaları arasında kendilerine ait olanı da görmek istemektedir. Bazıları ise şirket için bir intranet kurulması isteniyorken bu ortamdan yararlanmak istemektedir. Bu da tabi gizliliği yüksek olan bilgilerin, genel kullanıma açık bir ağ ortamına çıkarılması manasına gelmektedir ki ağ yöneticileri için yeni zorlukların da başlangıcının işaretçileridir.

Tabi bu durumlar karşısında, ağ güvenliğini sağlayacak ürünler devreye girmiş, bu konuda değişik teknolojiler geliştirilmiştir. Bu çalışmalardan biri de firewall' lardır. Yalnız maalesef bunlar da ağ yöneticilerinin beklentilerini tam olarak karşılayamamışlardır. **Computer Security Institute** (Bilgisayar Güvenliği Enstitüsü)' ün açıklamalarına göre internet üzerindeki şirketlerin beşte biri istenmeyen dış müdahalelere uğramıştır. Bunların da üçte biri kurulu bir firewall' a sahipti. Yani firewall' lar, ağ yöneticilerinin beklediği güvenliği sağlayamamıştır.

Internet üzerinden dünyanın dört bir tarafından erişilebilir hale gelmiş şirketler için böyle tehlikeler söz konusu iken, bazı ağ yöneticileri bu tehlikelerden habersizdir. Bu konuda çalışmalarda bulunmuş bazı uzmanların belirttiğine göre, ağ yöneticilerinin bir kısmı, internet ortamında veri çalmanın ne kadar kolay olduğunu bilmemektedir ve aslında gerçek tehlikeyi oluşturan da budur.

Güvenliği sağlama konusunda yararlanılabilecek bir başka araç da Şifreleme' dir. Pek çok firewall üreticisi, şifreleme araçları ile çalışmayı desteklemektedir. Yönlendirici (router) üreticileri de bu konuda çalışmakta olup, şifrelenmiş bilgileri yönlendirebilen routerlar da üretilmiştir.

Uygun bir güvenlik sistemi oluşturmak için, ağa ve iletilecek bilgilere ilişkin bazı unsurların göz önüne alınması gerekmektedir. Karar verilmesi gereken konulardan ilki, şifreleme işleminin nerede yapılacağıdır. Bazı çözümler uygulama seviyesinde bu işi yapıyorken bazıları IP yığınında yapmaktadır. Uygulama seviyesinde yapılan şifrelemede ağ yöneticilerine şifrelemeyi istedikleri şeyleri seçme şansı da verilmektedir.

Cevaplanması gereken ikinci soru, verilerin nasıl şifreleneceğidir. Şu anda en çok kullanılan iki yöntemden biri 56 bitlik şifreleme anahtarı, diğeri ise 128 bitlik şifreleme anahtarı kullanmaktadır. Güvenlik danışmanlarının kabul edebildiği, yeterli olabilecek minimum anahtar uzunluğu 56 bittir. Burada üzerinde durulması gereken bir diğer konu da, paketin nerelerinin şifrelenmesinin yeterli olacağıdır. Bazı ürünler, tüm paketi, başlık kısmı da dahil olmak üzere şifrelerler. Diğerleri ise sadece bilgi kısmını şifrelerler.

Genel Çerçeve paragrafında da belirtildiği gibi, iyi bir güvenlik sağlamak için kullanıcı doğrulama mekanizmasının da kurulması gerekmektedir. Bunun manası, internet üzerinden birisi ile bağlantı kurulduğunda, irtibat halinde bulunulanın kimliğinin tam olarak belirlenmesidir. Bunun için üreticiler değişik çalışmalarda bulunmaktadırlar.

Bu noktada, güvenlik konusu ile ilgili yapılan çalışmalarda sıkça karşılaşılan ve hala daha yeterli düzeye gelinemeyen bir konudan bahsetmek gerekir ; Standart. Internet güvenliği ile ilgili pek çok çalışma yapılmasına rağmen, bu alandaki hızlı gelişmeler, hala daha bir standardın oturtulamamasına sebep olmuştur.

Günümüzde, internet ortamına açılma konusunda hemen hemen tüm şirketler isteklidir. Her ne kadar bu ağ yöneticileri için çözümü çok zor sorunları beraberinde getirse de kaçınılmaz olarak bu yöne doğru hızlı bir yönlenme olmaktadır. Ancak bu demek değildir ki internet ortamında güvenlik sağlanmıştır ve kuruluşlar aynı mantık ve rahatlıkta davranmaktadır. Bu konudaki rahatlık derecesi doğal olarak yapılan işin ve bununla da bağlantılı olarak taşınması gereken bilginin mahiyeti ile ilgilidir. Örneğin bir banka için bilgilerini internet ortamında gezdirmek, şu an için çok akıllıca bir işlem değildir. Internet ortamından müşterilere sunulan ev bankacılığı hizmetleri de, alt düzeylerde kalmaktadır. Önemli verilerin aktarılması işlemi için bankalar genellikle özel hatları tercih etmektedirler. Internet ortamının sağladığı güvenlik ortamına inançları, şu an için bu bilgilerin bu ortama aktarılması için yeterli olamamaktadır.

Bunun yanında, bu konularda biraz daha rahat davranabilen kuruluşlar da vardır. Bunlar belki biraz da kendi geliştirdikleri güvenlik mekanizmasının da yardımıyla, internet ortamından veri aktarımında yararlanmaktadırlar. Bununla ilgili olarak söylenen, kuruluşların ihtiyaçları doğrultusunda, değişik ürünlerin değişik modüllerinden yararlanmak suretiyle, kabul edilebilir bir güvenliğin sağlanabileceğidir. Fakat her şeye rağmen şu an için internet

ortamının güvenliği konusu üzerinde daha çok düşünmek gerekmektedir. Mevcut hali ile pek çok tehlikeye açık durumdadır.

8.3. PGP (Pretty Good Privacy)

8.3.1. Giriş

İnternet'in geniş çevrelerce yaygın olarak kullanılması birtakım güvenlik sorunlarını da beraberinde getirmiştir. Bu bildiride, Internet'in genel güvenlik sorunlarının yanı sıra özellikle E-posta uygulamalarındaki güvenlik sorunlarından bahsedilmiş ve çözüm olarak yaygın olarak kullanılan **PGP** (**Mükemmel Şifreleme ve İmzalama)** yazılımının temel işlev ve özellikleri anlatılmıştır.

Son yılarda Internet kullanımının yaygınlaşması birtakım güvenlik sorunlarını 1da beraberinde getirmiştir. Bunun başlıca sebepleri, Internet'in açık bir sistem olması ve üzerinde dolaşan verinin gasp edilmeye uygun olmasıdır. Değişik Internet uygulamalarının değişik güvenlik isterleri olabilir. Örneğin, Elektronik posta (E-posta) uygulamasında bilgi gizliliği önemli iken WWW uygulamalarında bilginin gizli olmasının bir anlamı yoktur.

Bu bildiride öncelikle Internet'in genel güvenlik sorunlarından bahsedilecektir. Daha sonra E-posta uygulamalarındaki güvenlik sorunları ve genel çözüm yaklaşımları verilecektir. En sonda ise PGP E-posta güvenlik yazılımının genel özellikleri anlatılacaktır.

8.3.2. İnternet'te Genel Güvenlik Sorunları

İnternet'te güvenlik denince ilk olarak yetkisiz kişilerin paylaşımlı bilgisayarlara sızıp bilgi hırsızlığı yapması veya bilgilere zarar vermesi akla gelmektedir. Gerçekten de en ciddi zararlar bu şekilde verilmektedir. Ancak buradaki sorun, iletişim ağından çok kullanılan uygulama katmanı yazılımlarının (telnet, ftp, http, sendmail vb.) ve sunucu (server) tarafındaki işletim sisteminin tasarım hatalarıdır. Bu tür güvenlik sorunları "uzaktan erişim" sorunları olarak adlandırılır. Günümüzde bu sorunların çözümü olarak firewall'lar yaygın olarak kullanılmaktadır. Firewall iç ağı dış ağdan (Internet'ten) ayıran bir duvar olarak düşünülebilir. Temel işlevi güvenlik gediği olan uygulamalara ait veri paketlerinin iç ağa ulaşmasını engellemektir. Böylelikle, iyi veya kötü niyetli olduğuna bakılmaksızın hiç kimse ağ dışından ağ içine izin verilen uygulamalar dışındaki uygulamalar için ulaşamayacaktır. Başka bir deyişle kurunun yanında yaş da yanacaktır.

Firewall'lar kolay uygulanabilirliği yüzünden yaygın olarak kullanılmaktadır. Ancak "ya hep ya hiç" mantığındaki bu çözümün kesin çözüm olmadığı da açıktır. Çoğu kuruluşlar sadece kendi izin verdikleri kişilerin kendilerine ulaşmasına olanak tanıyan sistemler kurmak istemektedir. Bu da ancak, kimlik kanıtlama (authentication) özelliği olan sistemlerle mümkün olabilmektedir. Kimlik kanıtlama sistemleri şifrelemeye dayalı sistemlerdir. İstemci ile sunucu bir protokolle olarak aralarında haberleşerek ortak bir sırrı paylaşırlar. Böylelikle, birbirlerinin kimliklerinden emin olurlar. Açık kanallardan giden bilginin yetkisiz insanlar tarafından öğrenilmesini engellemek için şifreleme yöntemleri kullanılır. Bu şekilde insanların birbirlerinin kimliklerini kullanması önlenir. Bunun dışında, iletişimde bulunan taraflar ortak bir oturum anahtarı üzerinde anlaşarak aralarında ilettikleri veriyi şifrelerler. MIT tarafından geliştirilen Kerberos yaygın olarak kullanılan özel anahtar (private key) tabanlı bir kimlik kanıtlama sistemidir. Bunun dışında açık anahtar (public key) tabanlı kimlik kanıtlama sistemleri de vardır (SecureID, SSL, SET, vb.). Açık anahtar şifreleme algoritmaları yapılarından dolayı yavaştırlar, fakat kırılmaları güç ve anahtar dağıtım sorunu da özel anahtar tabanlı sistemlere göre daha azdır. Bankacılık, elektronik alışveriş ve elektronik ödeme gibi paraya dayalı Internet uygulamalarında daha güvenli olduğu için açık anahtar tabanlı sistemler tercih edilmektedir. Ancak, performans düşüklüğü bu açık anahtar tabanlı sistemlerin gerçek zamanlı uygulamalardaki şansını azaltmaktadır.

Genel olarak ağ güvenliğinden bahsedildiğinde akla gelen diğer bir sorun da açık kanallarda dolaşan bilginin gizliliği ve bütünlüğüdür. Bilgi gizliliği, verinin alıcısı dışında hiç kimse tarafından okunamaması, bilgi bütünlüğü ise, verinin değişmeden alıcısına ulaşması anlamına gelmektedir. Kimlik kanıtlama sistemleri oluşturdukları oturum anahtarları ile bu sorunları çözebilmektedirler. İlk bakışta pek önemsenmeyen ama bazı uygulamalarda elzem olan diğer bir güvenlik sorunu ise inkar edememe (non-repudiation) sorunudur. Özellikle doğrudan parayla ilgili uygulamalarda ortaya çıkan bu sorun, gönderenin gönderdiği bir mesajı daha sonra inkar edememesi, etse bile alıcının gönderenin mesajı gönderdiğini üçüncü kişilere ispat edebilmesi zorunluluğundan kaynaklanmaktadır. Kerberos gibi uzaktan erişim için kullanılan özel anahtar tabanlı sistemlerin inkar edememeyi sağlaması ortak anahtar kavramından dolayı imkansızdır. Çünkü, gönderici gönderdiği bir mesajın alıcı tarafından uydurulduğunu iddia edebilir, alıcı da aynı anahtarı bildiği için bu durumun aksini ispatlayamaz. Açık anahtar tabanlı sistemlerde ise şifreleme anahtarı ile şifreyi çözme

anahtarı farklı anahtarlar olduğundan inkar edememe sağlanabilir. Bunun detayları bir sonraki kısımda verilecektir.

8.3.3. E-Posta uygulamalarının güvenlik sorunları

Gönderilen mesajın içeriğine ve iletişim uçlarının isteğine bağlı olmakla beraber, E-posta uygulamalarında bir önceki kısımda bahsi geçen kimlik kanıtlama, bilgi gizliliği, bilgi bütünlüğü ve inkar edememe sorunlarının hepsinin varlığından bahsedilebilir. Kimlik kanıtlama, gönderenin ve alıcının E-posta adreslerinin gerçekten de bahsi geçen kişilere ait olup olmadığı ile ilgilidir ve iki taraf da birbirlerinin gerçek kimliklerinden emin olmak isteyebilir. İnkar edememe ise alıcının gönderenin kimliğini gerektiğinde üçüncü bir kişiye E-posta mesajını göstererek ispatlayabilmesidir. Bu sorunu çözen bir sistemde kimse başkasının adını kullanarak E-posta gönderemeyecektir. Bilgi gizliliği, gönderilen E-posta mesajının sadece alıcı tarafından okunabilmesinin sağlanması sorunudur. Bilgi bütünlüğü ise E-posta mesajının değişmeden alıcısına ulaşması ile ilgilidir.

İnkar edememe ve dolayısıyla başkasının adını kullanarak E-posta mesajı göndereememe özelliği, ancak açık anahtar tabanlı şifreleme algoritmaları kullanan sayısal imzalar (**digital signatures**) destekli sistemler ile mümkündür. Açık anahtar tabanlı şifreleme algoritmalarında şifreleme anahtarı (**açık anahtar**) ve şifre çözme anahtarı (**gizli anahtar**) farklıdır ve şifreleme anahtarı herkese açıktır. Ancak, şifreleme anahtarından şifreyi çözme anahtarını elde etmek pratik olarak imkansızdır. Bundan başka, bu iki anahtar birbirlerinin tersi işlemler yaparlar. Birinin kodladığını diğeri çözer. Bu özelliği sayesinde açık anahtar tabanlı sistemler hem mesaj gizleme, hem de sayısal imzalama yapabilmektedir. Bir mesaj, alıcının açık anahtarı (public key) ile kodlandığında, ancak alıcının gizli anahtarı (secret key) ile açılabileceğinden ve bu anahtara sadece alıcı sahip olduğundan, esas mesajı sadece alıcı okuyabilecektir. O yüzden, bu kodlama bilgiyi gizleyen bir şifreleme işlemidir. Böylelikle, bilgi gizleme ve bilgi bütünlüğü sorunları çözümlenebilir. Öte yandan, bir mesajın gönderenin gizli anahtarı ile kodlandığını düşünelim. Söz konusu gizli anahtarı sadece gönderen bildiğinden, bu kodlama işlemi o mesaja gönderenin attığı bir sayısal imza olarak değerlendirilebilir. İmzanın kontrolü ise kodlanmış mesajın gönderenin açık anahtarı ile açılmasından başka bir şey değildir. Açık anahtar da herkes tarafından bilindiğinden, herkes sayısal imza kontrolü yapabilmektedir. Böylelikle, inkar edememe, başkasının yerine mesaj göndereememe ve gönderenin alıcıya kimliğini kanıtlaması sorunları çözülebilmektedir.

Geriye kalan tek sorun alıcının gönderene kimliğini kanıtlaması sorunudur. Başka bir deyişle, gönderenin mesajı, alıcıya ait olduğuna emin olduğu bir açık anahtarla şifrelemesi gerekir. Aksi halde alıcı şifreyi çözemeyecektir ve doğru olmayan açık anahtarın gizli anahtarını bilen fakat gerçekte istenilen alıcı olmayan biri gizli mesajı okuyabilecektir. Benzer bir durum sayısal imza doğrulanmasında da yaşanabilir. Eğer alıcı yanlış bir açık anahtar biliyorsa, sayısal imzayı doğrulayamayacaktır. O yüzden, hem alıcının hem de gönderenin kullandıkları açık anahtarların doğruluğundan emin olmaları gerekir. Bunun için de en iyi yöntem açık anahtarları sahibinden doğrudan almaktır. Bunun mümkün olmadığı durumlarda kefalet (**certification**) mekanizmaları kullanılabilir. Kefalet, bir açık anahtar, açık anahtarın sahibinin kimliği ve E-posta adresinden oluşan bir veriye herhangi birinin koyduğu sayısal imzadır. Eğer bir başkası kefaleti imzalayan insana güveniyorsa, kefaletin içindeki açık anahtara ve açık anahtarın sahibinin kimliğine de güvenecektir.

PGP yukarıda sözü edilen çözümleri içeren bir mesaj şifreleme ve sayısal imzalama programıdır. PGP aynı zamanda açık ve gizli anahtarlarla ilgili işlemleri de yapabilmektedir. Sonraki kısımda PGP'nin genel yapısı ve işlevlerinden bahsedilecektir.

8.3.4. PGP 'nin Genel Yapısı ve İşlevleri

PGP tamamen Phil Zimmerman'ın kişisel çabaları ile oluşturulmuş bir yazılımdır. Aslında temel olarak bakıldığında, dosya bazında işlem yapan bir şifreleme ve sayısal imzalama programıdır. E-posta gönderme özelliği yoktur. Sadece E-posta olarak gönderilebilir halde dosyalar yaratır. Değişik platformlarda (MS-DOS, UNIX, Macintosh, Atari, Amiga, OS/2, VMS) çalışabilmektedir.

PGP şu anda dünyada kullanılan en yaygın E-posta şifreleme programıdır. Ancak, bu gelişim süreci içinde yaratıcısı Phil Zimmermann'ın başını oldukça ağrıtmıştır. PGP güçlü şifreleme algoritmaları içerdiğin-den, ABD gümrük kurallarına göre ABD dışına çıkartılması yasak bir üründür. Ancak, henüz bilinmeyen (!) bir şekilde ABD dışına çıkmıştır. Şu anda PGP'nin uluslararası sürümü ABD ve Kanada dışındaki ülkelerde WWW ve FTP sitelerinden dağıtılmaktadır. PGP'nin uluslararası sürümü ile Amerikan sürümü arasında işlevsel hiç bir fark yoktur. Tek fark Amerikan sürümünün hukuki olarak ABD dışında bulunamamasından kaynaklanan isim farklılığıdır. Amerikan hükümeti, PGP'nin ABD dışına çıkartılmasından Phil Zimmermann'ı sorumlu tutmuş ve hakkında soruşturma başlatmıştı. Ancak, Phil Zimmermann bu olaydan hakkında dava açılmasını gerektirecek deliller bulunamadığından aklandı. PGP'nin nasıl ABD dışına çıkartıldığı ise henüz belirlenemedi. PGP'nin uluslararası

sürümünün ABD ve Kanada dışında kullanımı kanunidir. Ancak bazı ülkelerin (İran, Irak, Çin, Fransa, vs.) şifreleme konusunda birtakım yasaklayıcı düzenlemeleri vardır.

PGP'nin ticari olamayan kişiler ve kuruluşlar tarafından edinilmesi ve kullanımı tüm dünyada ücretsizdir. PGP'nin tüm ticari hakları **ViaCrypt** adlı bir ABD şirketinindir. O yüzden, ticari kullanım için bu şirketten lisans alınması gereklidir. PGP'nin ABD dışına çıkması yasak olduğu için ViaCrypt'in ABD dışına lisans vermesi de şu anda mümkün gözükmemektedir. PGP, RSA ve IDEA adlı iki patentli şifreleme algoritması kullanmaktadır. RSA'nın patenti ABD dışında geçerli değildir. Ancak IDEA'nın patenti Ascom-Tech AG isimli bir İsviçreli şirkete aittir. ABD ve Kanada dışındaki ülkelerde, bu şirketten IDEA lisansı alarak ticari olarak PGP'yi kullanmak mümkündür.

PGP emir bazında işlem yapan bir arayüze sahiptir. Orjinal PGP'nin grafik arayüzü yoktur. O yüzden kullanımı gayet zordur. PGP'nin program kodunun da dağıtılması, Phil Zimmermann'ın dışında bu konu ile ilgilenen başka insanları PGP uyumlu programlar yazmaya teşvik etmiştir. Böylelikle, PGP Eudora ve Pine gibi E-posta işlemleri yapan diğer programlarla entegre edilmiş ve birtakım Windows ön uçları üretilmiştir. Ancak, bu bildiride orijinal PGP'nin özelliklerinden bahsedilecektir.

PGP mesaj şifreleme ve imzalama özelliklerinin yanı sıra açık ve gizli anahtarlarla ilgili işlemler de yapabilmektedir. Bu özellikler aşağıda anlatılacaktır.

8.3.5. PGP' de Şifreleme ve İmzalama

PGP, temel olarak Rivest, Shamir ve Adleman tarafından geliştirilen RSA açık anahtar şifreleme algoritmasına dayanır. Ancak PGP, RSA'nın performansının diğer özel anahtar tabanlı şifreleme algoritmalarına göre düşük olması nedeniyle, mesaj şifrelerken IDEA özel anahtar algoritmasını kullanır. Bu şifrelemede kullanılan 128 bitlik anahtar ise RSA kullanılarak şifrelenir ve mesaj paketinin başına eklenir. IDEA'nin kullandığı 128 bitlik bu oturum anahtarı rasgele yaratılır ve sadece bir kere kullanılır. Mesaj imzalanırken ise mesajın tümü değil 128 bitlik bir özü (hash) imzalanır ve bu imza mesaj paketine konur. Öz yaratmak için ise MD5 tek yönlü öz yaratma algoritması kullanılır. Bu yöntemle elde edilen özden ana mesajı elde etmenin yolu yoktur. Şifreleme ve imza yaratmada gidilen bu yöntem PGP'nin performansını önemli ölçüde artırmaktadır.

PGP'nin başka bir özelliği ise mesajı şifrelemeden önce ZIP yöntemiyle sıkıştırmasıdır. Bu şekilde, hem saklama alanından, hem de bant genişliğinden tasarruf sağlanmış olur. Sıkıştırmanın diğer bir yararı ise harflerin sıklığını kullanan şifre çözümleyici ataklara karşı şifreyi koruyabilmesidir.

PGP, şifrelenmiş mesajı ve sayısal imzayı aynı pakete koyabildiği gibi ayrı ayrı şifrelenmiş mesaj ve imza da üretebilmektedir. Hatta istenirse sadece mesaj şifreleme veya sadece sayısal imzalama da yapabilir. Sadece imzalama modunda sıkıştırma yapıp yapmaması kullanıcıya bağlıdır. Sıkıştırma yapıldığında, sadece PGP kullanıcıları mesajı açabildiğinden, herkesin okuyabilmesi için haber gruplarına veya E-posta listelerine gönderilen imzalı E-postalarda mesajın açık (şifresiz ve sıkıştırmadan) gitmesi tercih edilmektedir. Bu durumda, imza mesajın sonuna eklenir.

Şifreleme ve sıkıştırma sonunda elde edilen dosya ikili (binary) bir dosyadır ve ikili bir dosyanın E-posta kanallarından mesaj olarak gönderilmesi imkansızdır. O yüzden, PGP bu dosyaları radix 64 çevrimini kullanarak ASCII formatına çevirir. Ancak, bu çevrim dosyanın boyunu 1/3 oranında artırır. Bu çevrim isteğe bağlıdır, eğer dosya E-posta ile gönderilmeyecekse ASCII formatına çevrilmesine de gerek yoktur.

PGP değişik işletim sistemlerinde çalışabilen bir programdır. Her işletim sisteminin de text dosyaları işleme yöntemi farklıdır. O yüzden, PGP E-postanın bir text dosyası olduğu ve hangi işletim sisteminden geldiği bilgisini de mesaj paketine ekleyebilmektedir. Bu işlem isteğe bağlı bir işlemdir ve sadece text dosyalar için yapılmalı, ikili dosyalar için yapılmamalıdır.

E-posta haberleşmesinin başka bir güvenlik tehdidi ise, şifrelenmiş ve imzalanmış bile olsa bir mesajın Internet dinlenerek kopyalanması ve daha sonraki bir zamanda tekrar alıcıya gönderilmesidir. Böyle bir durumda mesajın üzerindeki şifre ve imza doğru olduğu için, alıcı mesajın gönderenden geldiğini zannedebilir. Bu sorunu çözmek için imzanın ve şifrelenmiş mesajın başına zaman damgaları (**time stamp**) konur. Böylelikle, alıcı mesajın ve imzanın ne zaman gönderildiğinden emin olur.

Eğer çok gizli bir mesaj gönderilecekse, gönderen mesajı şifreledikten sonra bilgisayarındaki açık mesajı içeren dosyayı silmek isteyecektir. Ancak, işletim sistemlerindeki dosya silme işlemleri genelde geriye dönüşümlü işlemlerdir. PGP buna çözüm olarak kullanıcıya, şifreleme işleminden sonra açık mesajın olduğu dosyanın içini tamamen 0 ile doldurup silme olanağı tanımaktadır.

PGP'de eğer bir mesaj hem imzalanıp hem de şifrelenecekse, önce gönderenin gizli anahtarı ile mesajın özü imzalanır, sonra mesaj ile imza birlikte sıkıştırılır, daha sonra da şifrelenir. Bu şifrelemede kullanılan IDEA oturum anahtarı ise alıcının açık anahtarı ile şifrelendikten sonra paketin başına konur. ASCII formatına çevriliş ise bu eklemeden sonra yapılır.

Alıcı tarafındaki PGP programı şifrelenmiş ve imzalanmış bir mesajı aldığında, önce ASCII formatından ikili formata çevirir. Daha sonra, kendi gizli anahtarını kullanarak IDEA oturum anahtarını çözer ve imza ile birlikte mesajı ortaya çıkarır. Artık, açık mesaj alıcının elindedir. PGP, imzayı doğrulamak için ise E-posta ile gelen imzaya gönderenin açık anahtarını kullanarak RSA algoritmasını uygular. Sonuçta ortaya çıkan mesajın özü olmalıdır. Bu kontrolü de gerçek mesajın özünü bularak yapar. Bu iki öz birbirinin aynı ise mesaj üzerindeki imza doğrudur. Şifrelenmeden imzalanmış bir mesajın imzasını kontrol etmek için ise sadece yukarıda belirtilen imza kontrolü ile ilgili işlemler uygulanır. PGP, ayrı dosyalardaki imza ile mesajı kontrol edebildiği gibi, bir imzayı imzalanmış bir mesajdan ayırıp ayrı bir dosyaya da kaydedebilir. Bundan başka, kullanıcı şifrelenip imzalanmış bir mesajı açarken imzayı mesaj üzerinde bırakma hakkına da sahiptir.

PGP, şifreleme ve imzalamanın yanı sıra açık ve gizli anahtarla ilgili işlemler de yapmaktadır. Bunlarla ilgili özellikler aşağıda anlatılmıştır.

8.3.6. Anahtar İşlemleri

PGP kullanarak RSA anahtarları yaratmak mümkündür. Bu şekilde gizli anahtar ve ona karşılık gelen açık anahtar yaratılır. RSA anahtarlarının boyu kullanıcının güvenlik gereksinimine bağlı olarak 512 ile 2048 bit arasında değişebilir. O yüzden, gizli anahtar-ları ezberlemek ve her gerektiğinde klavyeyi kullanarak girmek nerdeyse imkansızdır. O yüzden, PGP'de gizli anahtarlar şifrelenerek bir dosyada saklanır ve gerektiğinde bu dosyadan okunarak işlem yapılır. Bu dosyanın adı 'secring.pgp'dir. Gizli anahtarı şifrelemek için IDEA algoritması kullanılır. Şifreleme anahtarı 128 bit uzunluğundadır. Söz konusu anahtar kullanıcının belirlediği bir şifre cümlenin (**passphrase**) MD5 kullanılarak çıkartılan özüdür. Şifre çözücü ataklara karşı koyabilmek için bu şifre cümlenin mümkün olduğunca uzun ve anlamsız olması önerilir. Gizli anahtarın ele geçirilmesini önleyecek diğer bir önlem ise, secring.pgp dosyasının sabit disk üzerinde değil de taşınabilir disketler üzerinde tutulması ve bu disketin sıkı bir şekilde korunmasıdır. Kullanıcı isterse birden fazla gizli anahtar yaratıp şifreleyerek secring.pgp dosyası içinde saklayabilir. PGP'de bir gizli anahtarı secring.pgp'den silmek de mümkündür. Bundan başka, kullanıcı bir gizli anahtarı ayrı bir dosya olarak da tutabilir, gerektiğinde ayrı bir dosyadaki gizli anahtarı secring.pgp içine taşıyabilir. PGP'de gizli anahtara ait kullanıcı bilgilerini ve şifre cümleyi değiştirme özellikleri de vardır.

PGP, açık anahtarları 'pubring.pgp' isimli bir dosyada saklar. Açık anahtarların gizlilik gibi bir gereksinimleri olmadıkları için şifrelenmelerine de gerek yoktur. Açık anahtarlarla ilgili tek sorun yanlış açık anahtar sorunudur. Gönderenin veya alıcının şifreleme ve imza kontrollerinde yanlış açık anahtar kullanmaları yanlış sonuçlara sebep olabilir. PGP bu sorunu aşmak için açık anahtar imzalama (**public key signatures**), başka bir deyişle kefalet (certification) mekanizmasını kullanır. Aslında en iyi açık anahtar edinme yolu sahibinden doğrudan edinmektir. Bu şekilde elde edilen açık anahtarlar kullanıcı tarafın-dan imzalanarak pubring.pgp dosyasına kaydedilir. Bunun imkansız olduğu durumlarda ise kefalet (açık anahtar imzalama) yöntemi kullanılır. Kefalet, bir açık anahtarın doğruluğuna bir başkasının verdiği garantidir. Bu garanti, açık anahtar ve sahibinin kimliğinden oluşan bir bilginin kefaleti verenin gizli anahtarıyla imzalanması ile verilir. Bir açık anahtar ve bu anahtara atılan imzalar (kefaletler) bir arada pubring.pgp dosyası içinde tutulur.

PGP'de kefalet mekanizması şu şekilde işler: bir kullanıcı bir açık anahtara doğrudan güvenemiyorsa, o anahtar için verilmiş anahtar imzalarının sahiplerini tanıyıp tanımadığını kontrol eder. Bu kontrol pubring.pgp dosyası içinde imzalayıcı ile ilgili bir anahtarın olup olmadığıdır. Eğer imzalayıcının açık anahtarı varsa ve o anahtar üzerinde kullanıcının tanıdığı ve güvendiği başka birinin (veya kendisinin) imzası varsa, kullanıcının imzalayan insanı tanıdığına hükmedilir. Bun-dan sonraki kontrol ise tanınan imzalayıcının imzaladığı açık anahtarlara güvenilip güvenilmeyeceğinin kontrolüdür. Eğer o anahtarı güvenilen biri imzalamışsa, kullanıcı söz konusu açık anahtarı güvenerek kullanabilir. Bu mekanizmanın işlemesi için pubring.pgp içindeki her anahtar için bir güven bilgisi tutulur ve bu bilgiye göre kullanıcının açık anahtar sahibinin verdiği imzalara güvenip güvenmeyeceği hesaplanır. Bu hesaplama PGP tarafından otomatik olarak yapılır. Eğer sonuçta açık anahtarın doğruluğuna karar verilmişse kullanıcının söz konusu anahtarı imzalayarak pubring.pgp dosyasına kaydetmesinde bir sakınca yoktur. Burada önemli olan diğer bir konu ise zincirleme güven konusudur. Bir kullanıcı bir başka kullanıcıya güveniyorsa onun güvendiği başka kullanıcılara da güvenmeli midir? Bu sorunun cevabı evet ise kullanıcı zincirleme kefalete izin veriyor demektir. Burada hiyerarşik bir geçiş söz konusudur. Bu hiyerarşide kaç seviye gidileceğini ise kullanıcı belirler.

Özetlemek gerekirse, bir kullanıcının doğruluğuna güvenmediği bir açık anahtarı çekinmeden kullanabilmesi için ortakça tanınan ve güvenilen bir veya birkaç kişinin, ya doğrudan ya da hiyerarşik bir düzende birkaç seviyede geçişli olarak, söz konusu açık anahtarı imzalayarak kefalet vermesi gerekir.

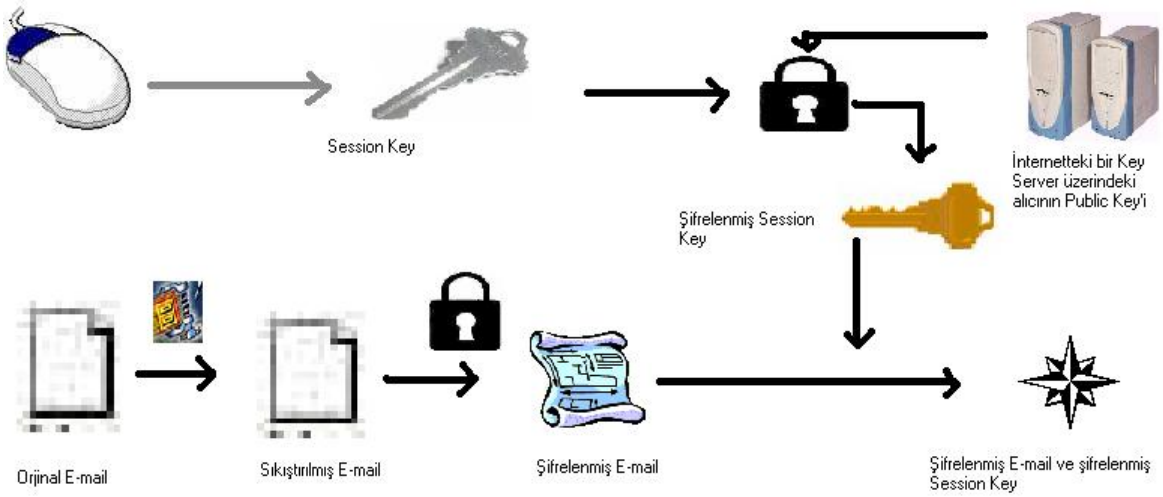
Açık anahtarlar ve üzerlerindeki imzalar, Internet üzerinden açık anahtar sunucuları vasıtasıyla istem bazında dağıtılırlar. Biri bir başkasının açık anahtarını arıyorsa, bir anahtar kelime vererek açık anahtar sunucu-sunun veritabanını arattırır ve sunucunun döndürdükleri arasından kendi istediği kişiye ait olan açık anahtarı kopyalar. PGP, bir dosya içindeki açık anahtarı üzerindeki imzaları kontrol ederek pubring.pgp içine koyabilmektedir. Bir kullanıcı, kendi açık anahtarını başkalarının kullanımına sunmak istiyorsa söz konusu anahtarı açık anahtar sunucusuna kaydettirmelidir. Açık anahtar sunucuları, uç kullanıcıların kullandığı PGP'den bağımsız çalışan bir veritabanı arama ve kaydetme sistemidir. Bu veritabanında kullanıcıların açık anahtarları ve bunların kefaletleri saklanır. Açık anahtar sunucuları anahtarların doğruluğunu garanti edemezler. Bir açık anahtarın herhangi bir sunucuya kaydettirilmesi yeterlidir. Açık anahtar sunucuları kendi aralarında haberleşerek tutarlılık sağlarlar. Bir kullanıcı açık anahtarını iptal etmek isterse bunu açık anahtar sunucusuna bildirmek zorundadır. Yetkisiz kişilerin açık anahtarları silmesini önlemek için, açık anahtarını sunucudan silmek isteyen biri önce PGP'yi ve kendi gizli anahtarını kullanarak bir açık anahtar silme mesajı hazırlar ve imzalar. Daha sonra bu mesajı açık anahtar sunucusuna gönderir. Açık anahtar sunucusu da söz konusu açık anahtarı iptal eder. Buradaki en önemli sorun, diğer PGP kullanıcılarındaki pubring.pgp'lerde bulunan geçersiz açık anahtarların nasıl iptal edileceğidir. PGP bunun için bir çözüm önermemektedir.
PGP, pubring.pgp üzerinde herhangi bir açık anahtarı silme, güven durumunu kontrol edip parametrelerini değiştirme, geçici olarak devre dışı bırakma, bir dosya olarak kaydetme, üzerindeki imzaları listeleme özelliklerine de sahiptir. Bundan başka, kullanıcı PGP kullanarak herhangi bir açık anahtarı imzalayabilir veya imzasını silebilir. PGP, pubring.pgp'deki açık anahtarları, üzerindeki imzaları ve güven durumlarını da listeleye-bilmektedir.

Kefalet mantığı, PGP dışında diğer açık anahtar tabanlı sistemlerde de kullanılmak-tadır. Ancak PGP'nin yaklaşımı, diğerlerin-den farklı olarak, tamamen dağıtıktır. PGP, açık anahtar imzalama yetkisini belirli kefalet otoritelerine vermektense herkesi kefalet otoritesi yapma yolunu seçmiştir. Bu durumun faydaları ve zararları tartışılabilir. Ancak, günümüzde açık anahtar tabanlı sistemler daha merkeziyetçi bir yapıya doğru standartlaşmaktadır**. ITU** (**I**nternational **T**elecommunications **U**nion) tarafından geliştirilen X.509 standart önerisi bu yönde yapılmış bir çalışmadır. Günümüzde açık anahtar tabanlı sistemler X.509 uyumlu olmaya çalışmaktadırlar, ama PGP tamamıyla bu standartlara uymayan bağımsız bir yapı

sergilemektedir. E-posta güvenliği konusun-da üretilen bir başka ürün ise **PEM** (Internet **P**rivacy **E**nhanced **M**ail yazılımıdır. PEM, PGP kadar yaygın kullanılmasa da, E-posta güvenliği konusunda standart olmaya çalışan X.509 uyumlu bir üründür.

8.3.7. Bir E-Postayı PGP ile Şifreleme

Gönderen tarafında: Teknik açıdan PGP önce bir e-mail'in şifrelenecek metnini sıkıştır (**compress**). Böylece yerden tasarruf edildiği gibi belli bir kalıptaki ifadeleri tanıyan kriptoanalitik saldırı denemelerinin de başarıya ulaşmasını engeller. PGP, kullanıcının fare hareketlerinden ve klavye girişlerinden şimdi bir yarı **random** (tesadüfi) karekter kodu oluşturur, bu o anki oturumun anahtarı (**session key**) olarak sadece bir kullanımlık için belirlenir. Sıkıştırılmış mail metni, oturum anahtarıyla geleneksel (simetrik) bir yöntemle şifrelenir. Şimdi alıcının açık anahtarı (**puplic key**) devreye girer: Tesadüfi bir şekilde yerel olarak kaydedilmemişse, PGP programı bu anahtarı internetteki bir sunucudan alır.
Public key ile şimdi session key'in kendisi (asimetrik yöntemle) şifrelenir, yani mesaj metni burada şifrelenmez. Şifrelenmiş metin ve şifrelenmiş session key e-mail programı tarafından alıcıya gönderilir.



Şekil.8.1 : PGP ile Şifreleme

Alıcı tarafında: Alıcıya ulaştığında şifrelerin çözülmesi tam tersi sırada gerçekleşir: Alıcının sadece yerel sabit diskinde kayıtlı olan kişisel anahtarıyla (private key) PGP programı session key'in şifresini çözer. Session key çözüldükten sonra yine bunun sayesinde şifrelenmiş

metinde çözlür. Sıkıştırılan mesaj metni açıldıktan (decompress)sonra orijinal metin alıcı e-mail programı için hazır durumdadır. Session key ise artık bir işle yaramaz ve atılır.

8.3.9. PGP de Güvenlik Seviyesi

PGP'nin şu andaki en son uluslararası sürümü 7.0.3 sürümüdür. Şimdiye kadar bu sürümle ilgili güvenliği etkileyecek herhangi bir hata rapor edilmemiştir. PGP'nin güvenliği, temel olarak RSA ve IDEA algoritmalarının ne kadar güvenli olduğu ile ilgilidir. Her iki algoritma da geniş çevreler tarafından yeterli anahtar uzunluğu var olduğunda güvenli olarak nitelendirilen algoritmalardır. O yüzden, PGP şifreleme bilimi açısından tamamen güvenlidir. Yine de, PGP'nin güvenliğini artırmak ve işletim sisteminin hatalarından doğabilecek güvenlik gediklerini en aza indirmek için aşağıdaki önlemleri almak gereklidir.

Gizli anahtarın güvenliğini pekiştirmek için secring.pgp dosyası bilgisayarın sabit diski üzerinde bırakılmamalıdır. Bununla beraber, gizli anahtarı şifrelemek için kullanılan şifre cümlesi mümkün olduğunca uzun (128 karakter) ve anlamsız olmalı ve hiç bir yere kaydedilmemelidir. Yanlış açık anahtar kullanımını önlemek için ise kefalet kurallarına sıkı bir şekilde uyulmalıdır. Bunun mümkün olmadığı durumlarda ise güvensiz açık anahtarın MD5 kullanılarak çıkartılacak bir özünün E-posta dışı bir iletişim kanalıyla sahibinden doğrulanmasını istemek en uygun davranıştır. Bir açık anahtarın özünü çıkarmak için PGP kullanılabilir. Öte yandan, pubring.pgp dosyasının sabit disk üzerinde tutulmaması bu dosyanın yetkisiz kişilerce silinip değiştirilmesini önleyecektir.

Şekil.8.2 : İzleniyorsunuz

Ek.1.İnternet Adresleri

http://www.globalsign.com.tr/
http://www.verisign.com/
http://e-kimlik.bilten.metu.edu.tr/tknyrd/secure.html
http://www.cardeurope.demon.co.uk/
www.rsasecurity.com
www.pgp.com
www.crypto.com
www.guvenlikhaber.com
www.securityfocus.com
www.authentica.com
http://exchange.ups.com
www.cylink.com
www.pki.org
www.sanalpos.com
http://csrc.nist.gov
http://tuath.pair.com
www.cl.cam.ac.uk
www.counterpane.com
www.nai.com
www.sigaba.com
www.eweek.com.tr
www.baltimore.com

PGP www ;

http://world.std.com/~franl/pgp/pgp.html
http://www.pgp.net/pgpnet/
http://www.ifi.uio.no/pgp/
http://www.prairienet.org/~jalicqui/pgpfaq.txt
http://axion.physics.ubc.ca/crypt.html
http://www.mantis.co.uk/pgp/pgp.html
http://www.efh.org/pgp/pgpwork.html
http://www.yahoo.com/Computers/Security_and_Encryption/PGP___Pretty_Good_Privacy/

PGP news ;

comp.security.pgp.announce
comp.security.pgp.discuss
comp.security.pgp.resources
comp.security.pgp.tech
alt.security.pgp

Smart-card ;

http://www.smartcrd.com/
http://www.smartcardcentral.com/
http://www.ioc.ee/atsc/faq.html

Ek.2.Terimler Sözlüğü

açık anahtar (public key): Açık anahtarlı bir kriptografik yöntem (algoritma) kullanan bir kullanıcının kendisine ait olan iki anahtarından kamuya açık olanı.

açık anahtar altyapısı-AAA (public key infrastructure-PKI): Bilgi iletişiminde açık anahtarlı kriptografinin yaygın ve güvenli olarak kullanılabilmesini sağlamaya yarayan ve birbirleriyle eşgüdüm içinde çalışan anahtar üretimi, anahtar yönetimi, onay kurumu, sayısal noterlik, zaman damgası gibi hizmetlerin tümü.

açık anahtarlı kriptografi (public key cryptography): Her kullanıcıya, sürekli kullanım için biri açık diğeri gizli iki anahtarın verildiği şifreleme/şifre çözme yöntemlerinin tümü. Asimetrik kriptografi ya da çift anahtarlı kriptografi adını da alır.

açık bilgisayar ağı (open computer network): İsteyen herhangi bir bilgisayar kullanıcısının bağlanabileceği ve diğer kişilerle bilgisayar üzerinden iletişim kurabileceği, herkese açık elektronik iletişim ortamı. Örnek: Internet.

anahtar (key): Şifreleme ve şifre çözme sırasında kullanılan sayı dizisi.

anahtar üretimi (key generation): Açık anahtarlı kriptografide, her kullanıcının açık/gizli anahtar çiftinin, kullanılan kriptografik yönteme bağlı matematiksel işlemlerle hazırlanması

anahtar yönetimi (key management): Açık anahtarlı kriptografide her kullanıcıya farklı anahtar çiftleri verilmesi, kullanıcıların açık anahtarlarının herkesin ulaşımına açık olarak saklanması ve kullanıcıların gizli anahtarlarının mutlak gizliliğinin sağlanmasından sorumlu düzen

çift anahtarlı kriptografi (double key cryptography): Açık anahtarlı kriptografi veya asimetrik kriptografi.

elektronik kimlik belgesi-EKB (digital certificate): Onay kurumunun hazırladığı ve sayısal olarak imzaladığı, hangi açık anahtarın hangi kişiye ait olduğunu gösteren belge.

elektronik veri değişimi-EVD (electronic data interchange-EDI): Standart bir formda yazılmış olan bilgilerin bilgisayarlar arasında aktarımı ve otomatik olarak yorumlanıp işlenebilmesi.

Elektronik Veri Değişimi: (Electronic Data Interchange) Standart bir yapıda bilgisayardan – bilgisayara veri (ticari) transferi.

erişim (access): Herhangi bir sistemi kullanmaya başlama, örneğin bir elektronik ticaret sistemine bilgisayar üzerinden bağlanarak iletişim kurma.

gizli -özel,kişisel- anahtar (private key): Açık anahtarlı kriptografi kullanan bir kullanıcının, kendisine ait olan iki anahtarından gizli tutulanı.

gizlilik (privacy): İletişim kuran iki taraf arasındaki yazışmaların üçüncü kişilerden gizli tutulması, veya bir kişiye ait bilgilerin kendisi dışında herkesten gizli tutulması.

kanal (channel): Bilginin bir kullanıcıdan diğerine iletimi için gereken fiziksel iletişim ortamı, örneğin, bilgisayar bağlantısı, telefon kablosu, radyolink ve uydu üzerinden diğer kullanıcıya ulaşan bağlantının tümü

kapalı bilgisayar ağı (closed computer network): Kullanıcılarından biri olmak için belirli koşulların sağlanması gerektiği, herkese açık olmayan bilgisayar ağları. Örnek: Bankalar ve bankamatikler arasındaki bağlantı.

kimlik belirleme (authentication): Herhangi bir servisi almak isteyen birinin, gerçekten de kendi iddia ettiği kişi olduğunun belirlenmesi.

Kod: (a) Bilginin kısaltılarak kayıt edildiği ya da tanımlandığı karakter dizisi **(b)** Bilgisayarın tanıyacağı formda özel semboller kullanılarak bilginin gösterilmesi ya da tanımlanması.

kriptografik algoritma (cryptographic algorithm): Şifreleme / şifre çözmede kullanılan belirli bir yöntemin ayrıntılı içeriği, bu içeriğin matematiksel adımları.

kriptoloji (cryptology): Güvenli bilgi iletişimi ve/veya saklanması için sifreleme ve sifre çözme yöntemleri türeten, geliştiren, inceleyen bilim dalı.

Mesaj çizeneği: (Message diagram) Bir mesaj içindeki bölüm dizisinin grafiksel gösterimi.

Mesaj kodu: (Message code) Mesaj tipini tanımlayan ve tek olan alfabetik referans (isim).

Mesaj rehberi: (Message directory) İsimlendirilmiş, tanımlandırılmış ve tarif edilmiş mesaj tiplerinin listesi.

Mesaj tipi: (Message type) Belirlenmış işlem tipi için ihtiyaçları kapsayan, tanımlanmış ve planlanmış veri kümesi (seti).

Mesaj: (Message) Bilgiyi taşımak üzere planlanmış sıralı (düzenli) karakter serisi

sayısal imza (digital signature): Elektronik ortamdaki yazışmalara eklenen, yazıyı gönderenin kimliğini ve gönderilen yazının iletim sırasında bozulmadığını kanıtlamaya yarayan bölüm. Sayısal imza, yazının içeriğine ve imzalayanın gizli anahtarına bağlı bir kriptografik yöntemle atıldığı için, sayısal imzanın doğrulanmasında, imzayı atanın açık anahtarı kullanılır.

sayısal noter (digital notary): Bilgisayar ağlarında iletilen bilgileri tarafların isteği ile saklayıp, kendisine başvurulduğunda belgeleyebilen kuruluş.

tek anahtarlı kriptografi (single key cryptography): Şifreleme ve şifre çözme için aynı anahtarı kullanan kriptografik yöntemlerin tümü. Simetrik kriptografi veya gizli anahtarlı kriptografi adını da alır. Kullanılan gizli anahtarı mesajı gönderen ve alan kişilerin paylaşması gerektiği için, tek anahtarlı kriptografinin güvenilirliği, her kullanıcı çiftine ayrı bir anahtar verilebilmesine bağlıdır. Bu durumda, bir kullanıcı, haberleşeceği herkes için farklı bir anahtar kullanmak zorundadır; bu ise önemli bir anahtar dağıtımı problemine yolaçar. Çift anahtarlı kriptografi , bu sorunu ortadan kaldırmıştır.

zaman damgası (time stamp): Bilgisayar ağlarında iletilen mesajlara eklenen ve mesajın yazıldığı zamanı güvenli olarak belgeleyen damga.

Ek.3. Çeşitli Algoritma Örnekleri

1) Algoritma: Sonlu bir dizideki en büyük elemanı bulan Algoritma

procedure max (a_1 , a_2 , ... , a_n : integers)
max : = a_1
for *i* : = 2 to n
** if max < a_i then max : = a_i**
{ max en büyük eleman }

2) Algoritma: Euclidean Algoritma

procedure gcd(a , b : positive integers)
***x* : = a**
***y* : = b**
while $y \neq 0$
begin
** r : = *x* mod *y***
*** x* : = *y***
*** y* : = *r***
end { gcd (a ,b) is *x* }

3) Algoritma : Constructing Base *b* Expansions

procedure base *b* expansion (n : positive integer)
***q* : = n**
***k*: = 0**
while $q \neq 0$
begin
** a_k : = *q* mod *k***
*** q* : = [*q* / *b*]**
*** k* : = *k* + 1**
end { the base *b* expansion of *n* is (a_{k-1} ..... a_1 a_0)$_b$ }

4) Algoritma:Tamsayıların toplamı

procedure add(*a* ,*b* : positive integer)
{ the binary expansions of *a* and *b* are (a_{n-1} , a_{n-2} , ... a_1a_0)$_2$
** and (b_{n-1} , b_{n-2} ..... b_1b_0)$_2$, respectively }**
***c* : = 0**
for *j* : = 0 to *n* - 1
begin
** d : = [(a_j + b_j + *c*) / 2]**
** s_j : = a_j + b_j + *c* – 2*d***
*** c* : = *d***
end
s_n : = *c*
{ the binary expansion of the sum is (s_n s_{n-1} ..... s_0)$_2$ }

5) Algoritma:Tamsayıların çarpımı

procedure multiply(a ,b : positive integer)
{ the binary expansions of a and b are (a_{n-1} , a_{n-2} , ... a_1a_0)$_2$
and (b_{n-1} , b_{n-2} ..... b_1b_0)$_2$, respectively }
for j : = 0 to n - 1
begin
if b_j : = 1 then c_j : = a shifted j places
else c_j : = 0
end
{ c_0 , c_1 , ... , c_{n-1} are the partial products }
p : = 0
{ p is the value of $a.b$ }

6) Algoritma: Matris çarpımı

procedure matrix multiplication(A ,B : matrices)
for i : = 1 to m
begin
for j : = 1 to n
begin
c_{ij} : = 0
for q : 1 to k
c_{ij} : = c_{ij} + a_{iq} b_{qj}
end
end { C = [c_{ij}] is the product of A and B }

7) Algoritma: Boolean çarpımı

procedure Boolean Product (A ,B : ziro-one matrices)
for i : = 1 to m
begin
for j : = 1 to n
begin
c_{ij} : = 0
for q : 1 to k
c_{ij} : = c_{ij} $\vee$ a_{iq} $\wedge$ b_{qj}
end
end { C = [c_{ij}] is the Boolean product of A and B }

Ek.4.RSA Şifreleme Örneği

p = 61 <= first prime number (destroy this after computing e and d)
q = 53 <= second prime number (destroy this after computing e and d)
p.q = 3233 <= modulus (give this to others)
e = 17 <= public exponent (give this to others)
d = 2753 <= private exponent (keep this secret!)
Your public key is **(p.q,e)**
Your private key is **d**
The encryption function is: **encrypt(T) = (T^e) mod(p.q)**
= (t^17) mod 3233
The decryption function is: **decrypt(C) = (C^d) mod(p.q)**
= (C^2753) mod 3233
To encrypt the plaintext value 123, do this:

encrypt(123) = (123^17) mod 3233
= 337587917446653715596592958817679803 mod 3233
= 855
To decrypt the ciphertext value 855, do this:

decrypt(855) = (855^2753) mod 3233 =

504328889584160687344228991273944663145387836003550931555496756450105562861208255997874424542811005438349865428933638493024645144150785172091796654782635307099638035387326500896686074771829745822950342950407903581845940956377938586598936883808360284013250976862076697739667533250542826093475735137988063256482639334453092594385562429233017519771900169249169128091505960191787601713497254392792156967017899021343071464689712796102771813783945869677289869342365240311693217089269617643726521315665833158712459759803042503144006837883246101784830717585474547252069688925995892544366701432205469543174002285500923863694244485597333306305160738530286321930291350374547194675777671357954965202919790505781532871558392070303159585937493663283548602090830635507044556588963193180119341220178269233441013301164806963340240750469525886698765866900622402410208846650753026395387052663193358473481094876156227126037327597360375237388364148088948438096157757045380081079469800667348777958837582899851327930703533551275090439948178979054899338121732945853544741326805698108726334828546381688504882434658897839333466254454006619645218766694795528023088412465948239275105770491133290256843065052292561427303898320890070515110552506189941712317779515797942971179547529630183784386291397787766129820738907279676720235011399271581964273076407418989190486860748124549315795374377124416014387650691458681964022760277668695309039513149683190973245054523459447725658788769269335391869235481851854242092306499640682218449011913571088542442852112077371223831105455431265307394075927890822606043171133395752266034451645259763161842774590432019134528932993216130744053222747057289481214358683197841559727649635709090121513130415756920979851832104115596935784883366531595132734467524394087576977789084901269153228420809496307929724713044221942439065903081428939308

291584830873687450789770869218452967411463211556678655283381648067954559418910069509196589908545679807239237084630255354568691923554629957157358790622745861957217211107882865756385970941907763205097832395713464119025004702084856040821750949107716553117652974738031767658205876731402889103288343185088447211644271939037404131556498699591373651621084511374022433518599576657753969362812542539006855262454561419258809437402128886669744109721845342218171980899119537070545542033911964539366461792968165342652234639936742330970183533904623677693670380534264482173582384219251590438148524738896864244370318665419961537791396964900303958760654915244945043600135939277133952101251928572092597887511601959629615690271164318946373426500236310045557180036935860552649100009072451837866895644171649072783562810097085452413546966084481161338780654854515176167308605108065782936524108723263667228054003879410864348226750090778265121013728195831653139698309088731741747453598868429855980718519221597004650810606844559536480892249440542766329674592308898484868435865479850511542844016462352696931799377844302178570191970987516296546651302780099665800521782081393172323790132324946826092008199810376848471678749891936949979148247163450609371256541225019537951668976018550875993133677977939527822273233375295802631226653589482055665152894663690320832876804323906115493509545909340667640225867084833760536998679410262047090571567447056531112428629073548884929899835609996360921411284977458614696040287029670701478179490248282907484160083680458666855076046192252094349804715745268818131850859150194852763596503458153641656549316013061330407434457965108380304062240278898042825189094716292266898016684480963645198090510905796513075703792459580744797523712667610114738787421441491548135917439279949695641565386688389171544630561180536972834347021920634899953191764016110392490439179803398975491765395923608511807653184706473318015782074127647875927390874929557168536651859126663738312359458912678709583800022451509424457564874484086877530845395521730636693891702394037184780362774643171470855830491959895146776294392143100245613061114299370005577513397172825491100560089408984196713197091181655429087610900832499783133824078696157849234198629916800867749593407759306602207814943807854996798945399364063685722697422361858411425048372451244655802708591797955910865230997565198382779529457569965742455786883835444236857223681399021261363744082131478483203563615611346287019851423901842909741638620232051039712184983355286308685184282634615027441873586395040422815123995059959836537922272858474220716778366794513436380708657977421985359539316627998878972169596345534633649794922113017661316207477266113107012321403713882270221723233085472679533015079980622538354589480248200431447261915961905260340690619309392907241028494870016717296951770346790997944097506376492963567555800711621827727603182921790350290486090976266285396627024392536890256337101471683274045045830602286763142158159900791642627700054612322919219299716990769016902594646810414121420447240266165827568052416686147339332265951270064563044741608529167218700704514464979322666873214634674904118588676083684030619069578699009652139067520501974407677651043885151941619318479919134924388152822038464729269446084915299958818598855

19514906630731177723813226751694588259363878610724302565980914901032
78384821401136556784934102431512482864529170314100400120163648299853
25166349056053794585089424403855252455477792240104614890752745163425
13992163738356814149047932037426337301987825405699619163520193896982
54478631309773749154478427634532593998741700138163198116645377208944
00285485000269685982644562183794116702151847721909339232185087775790
95933267631141312961939849592613898790166971088102766386231676940572
95932538078643444100512138025081797622723797210352196773268441946486
16402961059899027710532570457016332613431076417700043237152474626393
99011899727845362949303636914900881060531231630009010150839331880116
68215163893104666659513782749892374556051100401647771682271626727078
37012242465512648784549235041852167426383189733332434674449039780017
84689726405462148024124125833843501704885320601475687862318094090012
63241969092252022679880113408073012216264404133887392600523096072386
15855496515800103474611979213076722454380367188325370860671331132581
99227975522771848648475326124302804177943090938992370938053652046462
55147267884961527773274119265709116613580084145421487687310394441054
79639308530896880365608504772144592172500126500717068969428154627563
70458838904219177398190648731908014828739058159462227867277418610111
02763247972904122211994117388204526335701759090678628159281519982214
57652796853892517218720090070389138562840007332258507590485348046564
54349837073287625935891427854318266587294608072389652291599021738887
95773647738726574610400822551124182720096168188828493894678810468847
31265541726209789056784581096517975300873063154649030211213352818084
76122990409576427857316364124880930949770739567588422963171158464569
84202455109029882398517953684125891446352791897307683834073696131409
74522985638668272691043357517677128894527881368623965066654089894394
95161912002160777898876864736481837825324846699168307281220310791935
64666840159148582699993374427677252275403853322196852298590851548110
40229657916338257385513314823459591633281445819843614596306024993617
53097925561238039014690665163673718859582772525683119989984646027216
46279764077057074816406450769779869955106180046471937808223250148934
07851137833251073753823403466269553292608813843895784099804170410417
77608463062862610614059615207066695243018438575031762939543026312673
77406936404705896083462601885911184367532529845888040849710922999195
65539701911191919188327308603766775339607722455632113506572191067587
51186812786344197572392195263333856538388240057190102564949233944519
65959203992392217400247234147190970964562108299547746193228981181286
05556588093851898811812905614274085809168765711911224763288658712755
38928438126611991937924624112632990739867854558756652453056197509891
14578114735771283607554001774268660965093305172102723066635739462334
13638045914237759965220309418558880039496755829711258361621890140359
54234930424749053693992776114261796407100127643280428706083531594582

30594632682786127020335698034614324569702148437 5 mod 3233 = **123**