



# Bilgi Güvenliđi ve Kriptografi

## Mesaj Doğrulama

Buraya kadar mesaj içeriğinin şifrelenmesiyle korunması üzerinde duruldu. Bu bölümde göndericinin doğrulanması yanında mesaj içeriğinin bütünlüğünün(değiştirmelere karşı) nasıl korunacağı üzerinde durulacaktır. Genellikle mesaj bütünlüğünün korunması elektronik ticaret uygulamalarında gizlilikten daha önde gelen bir husustur. Mesaj doğrulama şu kavramları içerir. Mesaj bütünlüğünün korunması, göndericinin kimliğinin geçerliliği ve mesaj kaynağının kendisini inkar edememesi(non repudation). Bir doğrulayıcı üretmek için kullanılabilen üç adet fonksiyon vardır. Bunlar;

**Mesaj şifreleme :**(Şifrelenen mesaj onun doğrulanması görevini yapar

**Mesaj doğrulama kodu(MAC):** (Bir fonksiyon ile bir anahtar ile sabit uzunluklu olarak üretilen değer doğrulama için kullanılır

**Hash(özet) fonksiyonu:** (Herhangi uzunluktaki bir mesajdan açık bir fonksiyon ile üretilen sabit uzunluktaki özet değer doğrulama için kullanılır.

## **Güvenlik gereksinimleri**

Bi ağdaki haberleşmenin içeriğinde aşağıda listelenen saldırılar tanınabilir.

İlk iki gereksinim mesaj gizliliği içerisinde değerlendirilir ve açıklanan şifreleme yöntemleriyle sağlanır. Diğer gereksinimler mesaj doğrulama içerisinde kalır. Bu noktada, kaynağından gelen mesajın değiştirilmemiş olması önemlidir. Aynı zamanda adres dizisi ve zamanlamadır. Sayısal imzanın kullanımı kaynağın inkar edilmesi ile ilgilidir.

- Mesaj içeriğini açıklama (Disclosure)
- Ağ trafiğinin analizi(traffic analysis)
- Gerçeği gizleme(masquerade)
- Mesaj içeriğini değiştirme(content modification)
- Mesaj sırasını değiştirme(sequence modification)
- Mesaj zamanlamasını değiştirme(timing modification)
- Kaynağın inkar edilmesi(source repudiation)
- Varışın inkar edilmesi(destination repudiation)

## Mesaj şifreleme

Mesaj şifrelemenin kendisi, doğrulama işlevini sağlayabilir. Burada ,mesajın bütünüünün şifrlenmesi, sadece uygun anahtarları bilenlerin mesajı şifreleyebilmesi nedeniyle onun doğrulayıcısı olabilir. Böylece geçerli olan mesaj anlaşılabilir.(mesajın uygun yapıda olması veya değişikliğe karşı denetim bilgisi(checksum) bulunması).

➤ Eğer simetrik şifreleme kullanılmış ise:

- Alıcı, mesajın gönderici tarafından oluşturulduğunu bilir.
- Kullanılan anahtarı sadece gönderici ve alıcı bilir
- Eğer mesaj, denetim bilgisi içeren uygun yapıda ise, mesajın içeriği değiştirilemez

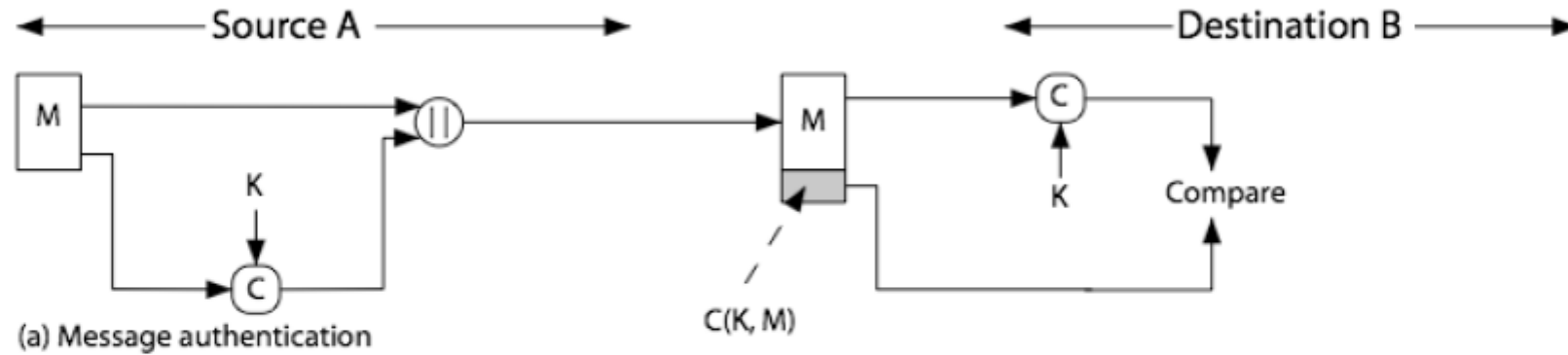
Açık anahtar teknikleri ile, sadece anahtar sahibi tarafından üretilebilen, sayısal imzalar kullanılabilir. Fakat mesajın sonunda iki açık anahtar işlemi gerekir.



## Mesaj Doğrulama Kodu(MAC)

Diğer bir doğrulama tekniği, bir gizli anahtar ile sabit uzunlukta üretilen ve kriptografik control verisi veya Mesaj doğrulama kodu olarak bilinen ve mesajın sonuna eklenen bir veri bloğu ile yapılır. Bu teknikte, A ve B olarak adlandırılan iki haberleşme grubu bir K ortak anahtarını paylaşır. Bir MAC fonksiyonu ,şifrelemeye benzeyen ve deşifrelemede olduğu gibi ters çevrilme gerektirmez. Bu sonuç mesajın sonuna eklenir.

- Alıcı mesaj üzerinde aynı hesaplamayı yapar ve sonucu MAC ile karşılaştırır.
- Böylece göndericiden gelen mesajın değiştirilmediğini garanti eder. (Şekil 7. 7)



Şekil 7.7 : Mesaj doğrulama kodunun çalışması

Doğrulama ve gizliliği birlikte sağlamak için MAC şifreleme ile birleştirilebilir. Sadece doğrulama gerekli ise MAC kullanılır.

Gönderici ve alıcının her ikisi de anahtarı paylaştığı ve üretebildiği için MAC sayısal imza değildir.

## MAC özellikleri

Bir C fonksiyonu tarafından üretilen bir MAC aynı zamanda kriptografik denetim bilgisidir. MAC kaynakta mesajın sonuna eklenir ve varışta yeniden hesaplanarak doğrulama yapılır.

MAC fonksiyonu, birçok farklı uzunluktaki mesajı aynı uzunluktaki özet değere dönüştürdüğü için çoktan bire bir fonksiyondur.

- Bir MAC kriptografik denetim değeridir(checksum)  
 $MAC = CK(M)$ 
  - Değişken uzunluktaki M mesajını bir gizli K anahtarı kullanarak sabit uzunluktaki bir doğrulayıcıya şeklinde sıkıştırır
- CK bir çoktan bire fonksiyondur, bu fonksiyon ile potansiyel olarak birçok mesaj aynı MAC değerine sahip olabilir fakat bu sonucu elde etmek çok zordur.

## MAC gereksinimleri

Bir MAC fonksiyonunun güvenliğini değerlendirmek için ona karşı olabilen saldırıları düşünmek gereklidir. Bundan sonra listelenen gereksinimleri sağlamak gereklidir.

İlk gereksinim, saldırganın anahtarı bilmesede dahi, verilen MAC ile uyuşan bir başka mesaj oluşturduğu mesaj yerine koyma saldırısı ile ilgilidir.

İkinci gereksinim, seçilen bir şifresiz metin tabanlı brute-force saldırısını önlemeyle ilgilidir.

Son gereksinim, doğrulama algoritmasının, mesajın belirli bir parçasının diğerlerine göre daha zayıf olmasını dikte eder. Özetle;

- Saldırı çeşitlerini dikkate alır
- MAC aşağıdakileri sağlamalıdır:
  1. Bilinen bir mesaj ve MAC için aynı MAC'a sahip bir başka mesaj bulunması verimsiz olmalıdır.
  2. MAC, düzenli dağıtılmış olmalıdır
  3. MAC, mesajın bütün bitlerine bağlı olmalıdır



## Hash Fonksiyonları

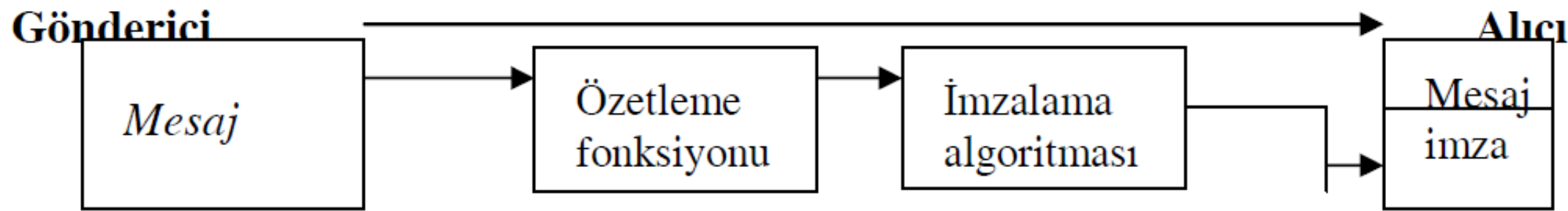
Temel kriptografik terimlerden biri “Kriptografik özetleme fonksiyonu” veya diğer adıyla tek yönlü özetleme fonksiyonu (one-way hash function) dur.

Tanım : Değişik uzunluktaki bit dizilerini sabit uzunluklu bit dizilerine taşıyan polinomsal zamanda kolay hesaplanabilen fonksiyona “Özetleme fonksiyonu” denir. Görüntü kümesindeki sabit uzunluklu oluşan bu bit dizisine ise “Özet-değer” (Hash-value) adı verilir.

Kriptografik olarak Özetleme fonksiyonları değişken  $m$  uzunluklu mesajları sabit  $n$  uzunluklu mesajlara indirgemek amacıyla kullanılırlar ( $m > n$ ). Seçilen  $h$  özetleme fonksiyonu iki farklı  $m_1$  ve  $m_2$  mesajlarını aynı özet değerine taşımamalı ( $h(m_1) \neq h(m_2)$ ) ve verilen bir  $y$  özet-değerinden bu değere ait  $m$  mesajı polinomsal zamanda hesaplanamamalıdır.


Özetleme fonksiyonlarının kriptografide kullanımı daha çok sayısal imza ve veri bütünlüğünün korunması alanlarında yaygındır. Sayısal imza uygulamalarında uzun mesajlar öncelikle bilinen bir Özetleme fonksiyonu ile sabit uzunluklu kısa bir diziye özetlenmeli ve bu özet-değer imzaya girmelidir.





**Şekil 7. 9** Şifrelenmemiş bir mesajın bütünlüğünün ve doğruluğunun korunması için özetlenip imzalanması.

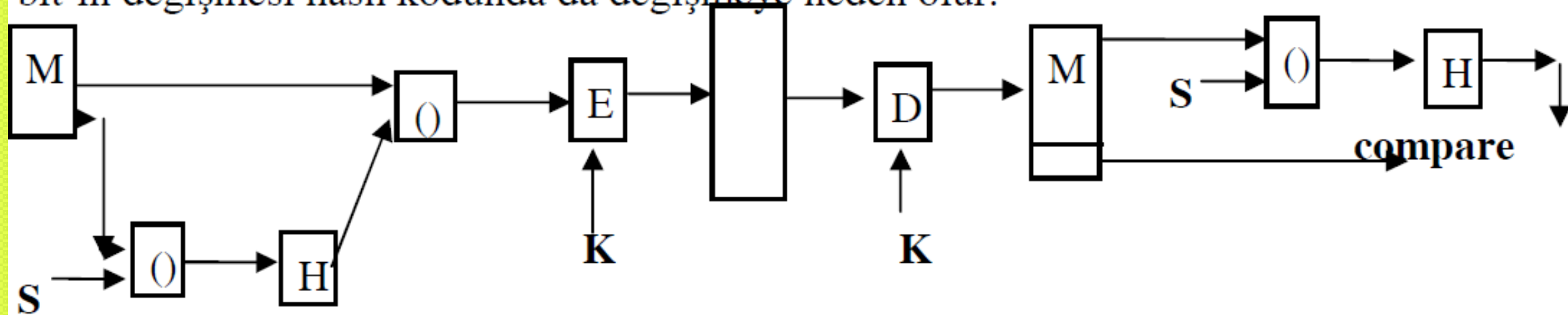
Özetleme fonksiyonu bilginin bütünlüğünü sağlamak için de kullanılabilir. Fonksiyonun tanım kümesindeki her mesajı görüntü kümesinde farklı bir özet-değerine taşıması gerektiği düşünülürse içeriği değiştirilmiş bir mesajın özet-değeri de farklı çıkacaktır. Dolayısıyla mesajı alan kişi mesajı kendisinin de bildiği Özetleme fonksiyonundan geçirecek ve elde ettiği özet-değerle kendisine gönderilen özet-değeri karşılaştıracaktır. Mesajın bütünlüğünün korunduğunun ispatı için bu iki değer uyuşması gerekmektedir. Virüs koruması ve yazılım korsanlığının önlenmesi Özetleme fonksiyonlarının diğer özel uygulamalarındandır.



Özetleme fonksiyonları yukarıda da belirtildiği gibi herkesçe bilinebilen ve gizli anahtarı olmayan tek-yönlü fonksiyon uygulamalarıdır. Eğer belli bir mesajın değiştirilip değiştirilmediğini tespit etmeye yönelik kullanılırlarsa “Değişiklik tespit kodları” (Modification Detection Codes) adını alırlar. Bu alanla ilgili olarak gizli bir anahtar içeren ve veri bütünlüğünün yanı sıra verinin kaynağının doğrulanması işleminde de kullanılan Özetleme fonksiyonlarına “Mesaj doğrulama kodları” (Message Authentication Codes) adı verilir.

## Hash Fonksiyonu

Mesaj yetkilendirme kodunun bir çeşidi, tek yönlü hash fonksiyonu olarak çok sık kullanılır. Mesaj yetkilendirme kodu olarak, bir hash fonksiyonu, değişken uzunluklu  $M$  mesajını giriş olarak alır ve çıkış olarak sabit uzunluklu, mesaj özeti denilen  $H(M)$  hash kodu üretir. Hash kodu, mesajın bütün bitlerinin bir fonksiyonudur ve hata bulma özelliği vardır. Mesajdaki bir veya birkaç bit'in değişmesi hash kodunda da değişmeye neden olur.



Şekil 7.10 Hash Fonksiyonunun temel Kullanımı: Yetkilendirme ve gizlilik sağlar.

Özetleme fonksiyonunun amacı, bir dosya, mesaj veya diğer bir veri bloğunun parmak izini üretmektir.

Güvenli özetleme(hash) fonksiyonlarının özellikleri aşağıda verilmiştir. Esas olarak iki mesaj için aynı özet değerini bulmak çok zor olmalı ve özet açık bir şekilde mesajla ilişkili olmamalıdır.(mesajın karmaşık doğrusal olmayan bir fonksiyonu olmalıdır). Özetleme fonksiyonları ve blok şifreleyicilerin tasarımı arasında birçok benzerlik bulunur.

- $H$  herhangi boyutlu bir  $M$  mesajına uygulanabilir
- $H$  sabit uzunlukta bir çıkış( $h$ ) üretir
- $H(M)$ , verilen bir  $M$  mesajı için kolay üretilebilir
- Verilen bir  $h$  değeri için,  $H(x)=h$  yı sağlayan  $x$ 'i bulmak hesaplama bakımından verimsizdir. Bu  $H(x)$ 'in tekyönlü özelliğidir
- Verilen bir  $x$  bloğu için  $H(y)= H(x)$  olan  $y \neq x$  gibi bir  $y$  değerini bulmak hesaplama bakımından verimsizdir. Bu özellik “weak collision resistance” dır.
- $H(y)=H(x)$  için, bir  $(x,y)$  çifti bulmak hesaplama bakımından verimsizdir. Bu özellik “strong collision resistance” dır.



## Basit Hash Fonksiyonları

Bütün hash fonksiyonları aşağıdaki genel prensipleri kullanarak çalışır. Giriş, n bitlik blok dizisi olarak görülür. Giriş her defasında bir blok olarak n-bitlik hash fonksiyonunu üretmek için işlenir. En basit hash fonksiyonu, her bloğun bit bit XOR işlemine tabi tutulmasıdır. Bu aşağıdaki gibi açıklanır:

$$C_i = b_{i1} \oplus b_{i2} \oplus \dots \oplus b_{im}$$

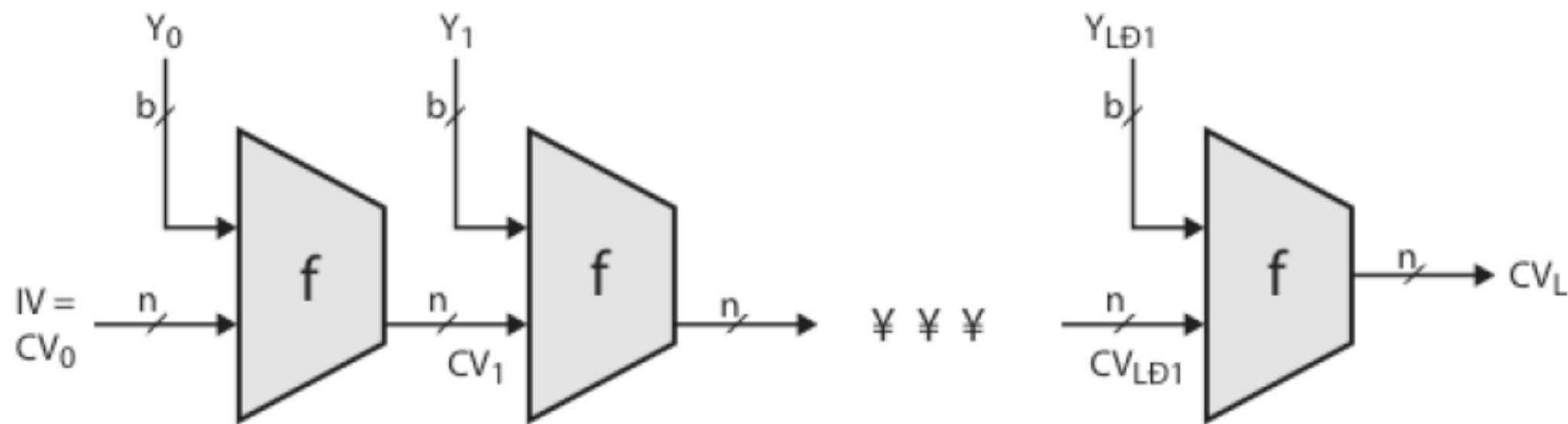
$C_i$ : hash kodunun i. biti,  $1 \leq i \leq n$  ;  $b_{ij}$  : j.'inci bloktaki i'inci. Bit

Genel hash algoritmasının blok şeması Şekil 7.11'de verilmiştir.

$$CV_0 = IV$$

$$CV_i = f(CV_{i-1}, Y_{i-1}) \quad 1 \leq i \leq L$$

$$H(M) = CV_L \quad (CV : \text{Chaining Value})$$



$IV$  = Initial value  
 $CV_i$  = chaining variable  
 $Y_i$  =  $i$ th input block  
 $f$  = compression algorithm

$L$  = number of input blocks  
 $n$  = length of hash code  
 $b$  = length of input block

Şekil 7.11. Özetleme fonksiyonu genel yapısı

Bugün sıklıkla kullanılan Özetleme fonksiyonlarına örnek olarak 1992’de Ron Rivest tarafından geliştirilmiş MD5 (message-digest algorithm) algoritması gösterilebilir. NIST ve NSA’nın ortaklaşa geliştirmiş olduğu SHA (secure hash algorithm) ise yine NIST’in Özetleme fonksiyonları için belirlediği standart olan SHS (secure hash standard) içerisinde yer almış ve kullanımı yaygın olan bir özetleme algoritmasıdır. Önemli özetleme fonksiyonlarının özellikleri Tablo 7. 3’te verilmiştir.

Adı	Blok Uz.(bit)	Max mesaj	Sonuç(bit)	Adım sayısı	Mantık F.	Ek sabit
MD5	512	$\infty$	128	64(4x16)	4	64
SHA1	512	$2^{64} - 1$	160	80(4x20)	4	4
RipeMD-160	512	$2^{64} - 1$	160	160(5x16 çift)	5	4

Tablo 7.3 Önemli Özetleme fonksiyonlarının özellikleri

# Sayısal İmza

**5070 sayılı Kanun'daki tanım:**

*“Başka bir elektronik veriye eklenen veya elektronik veriyle mantıksal bağlantısı bulunan ve kimlik doğrulama amacıyla kullanılan elektronik veri”*

**Yasal olmayan elektronik imzalar:**

- ❖ Kağıt üzerindeki imzanın elektronik ortama aktarılmasıyla oluşturulan resim
- ❖ Ekranı atılan imza
- ❖ Parmak izinin elektronik ortama aktarılması
- ❖ vs.. vs..



- ❖ Açık Anahtarlı Altyapı Teknolojisi
- ❖ Matematiksel şifreleme yöntemleri kullanılır. Şifreleme için büyük sayı dizilerinden oluşan 2 anahtar kullanılır.

**Özel anahtar (imza oluşturma verisi)**

E-imzayı oluşturmak için kullanılır.

Sadece kişinin kendisinde bulunur.

Akıllı kart içinde saklanır ve bu araçtan dışarıya çıkarılamaz.

**Açık anahtar (imza doğrulama verisi)**

E-imzayı doğrulamak için kullanılır.

Gizli olmayan, herkese açık bir veridir.

### **Anahtarların Kullanım Amaçları**

- ❖ Özel anahtar: Güvenli elektronik imza oluşturma amacıyla kullanılır. Başka bir amaç için kullanılmaz.
- ❖ Açık anahtar: Oluşturulan güvenli elektronik imzanın doğrulanması için kullanılır. Başka bir amaç için kullanılmaz.

### **Elektronik Sertifikanın Geçerlilik Süresi**

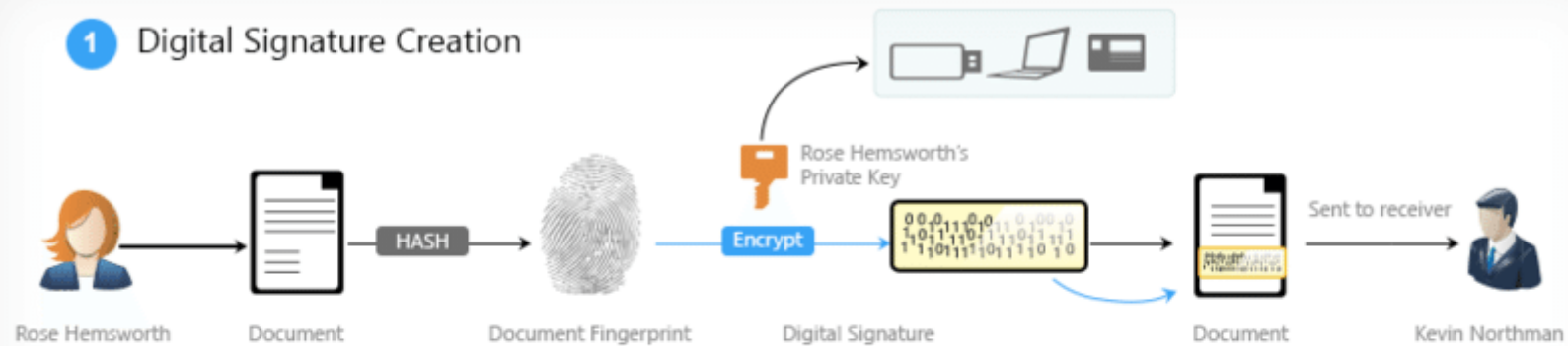
- ❖ Anahtarların kriptografik açıdan güvenlik süresi:

1024-bit : 1 yıl

2048-bit : 10 yıl

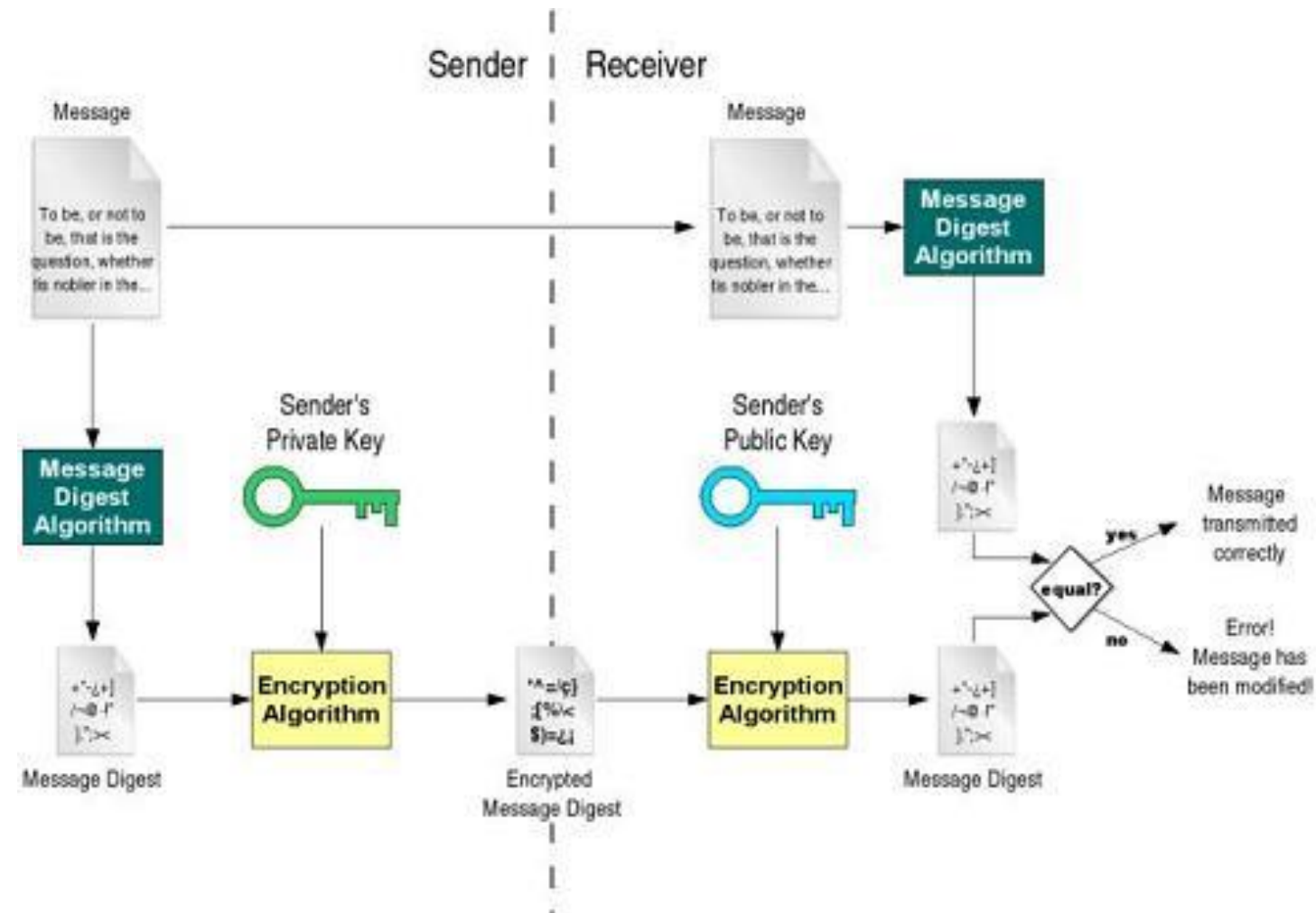
Geçerlilik süresi dolan elektronik sertifikaya ait özel anahtar imza oluşturma amaçlı kullanılmaz. Açık anahtar geçmişte oluşturulmuş imzaların doğrulanması için kullanılır.

## 1 Digital Signature Creation



## 2 Digital Signature Verification







## SIGNING



## VERIFICATION

