Bu, CBC tekniğine benzer, ancak bu durumda gizli bir anahtar yoktur. Herhangi bir karma kodda olduğu gibi, bu şema da doğum günü saldırısına tabidir ve şifreleme algoritması DES ise ve yalnızca 64 bitlik bir karma kod üretiliyorsa, sistem savunmasızdır.

Ayrıca, rakibin yalnızca bir mesaja ve geçerli imzasına erişimi olsa ve birden fazla imza elde edemese bile doğum günü saldırısının başka bir versiyonu kullanılabilir. Senaryo şu şekildedir: Rakibin şifrelenmiş karma kod biçiminde bir imzaya sahip bir mesajı yakaladığını ve şifrelenmemiş karma kodun m bit uzunluğunda olduğunu varsayıyoruz.

1. Bu alt bölümün başında tanımlanan algoritmayı kullanarak şunu hesaplayın: Şifrelenmemiş karma kod G.
2. Q1, Q2, c , QN-2 biçiminde istediğiniz mesajı oluşturun .
3. 1 ... i ... (N - 2) için Hi = E(Qi , Hi-1) değerini hesaplayın .
4. 2m/2 rastgele blok üret ; her X bloğu için E(X, HN-2)'yi hesapla. Ek 2m/2 rastgele blok üret; her blok Y için , D(Y, G)' yi hesapla , burada D, E'ye karşılık gelen şifre çözme fonksiyonudur.
5. Doğum günü paradoksuna dayanarak, yüksek olasılıkla X ve Y olacaktır . böylece E(X, HN-2) = D(Y, G).
6. Q1, Q2, c , QN-2, X, Y mesajını oluşturun. Bu mesajın karma kodu G'dir. ve bu nedenle yakalanan şifreli imza ile birlikte kullanılabilir.

Bu saldırı biçimine orta yol saldırısı denir . Birçok araştırmacı, temel blok zincirleme yaklaşımını güçlendirmeyi amaçlayan iyileştirmeler önermiştir. Örneğin, Davies ve Price [DAVI89] varyasyonu şöyle açıklar:

$$\text{Merhaba} = E(M_i , \text{Merhaba}_{-1}) \quad \text{Merhaba}_{-1}$$

[MEYE88]'de önerilen bir diğer varyasyon ise

$$\text{Merhaba} = E(H_{i-1}, M_i) \quad M_i$$

Ancak, bu şemaların her ikisinin de çeşitli saldırılara karşı savunmasız olduğu gösterilmiştir [MIYA90]. Daha genel olarak, gizli bir anahtar olmadan şifreli blok zincirleme kullanımını içeren herhangi bir karma şemasına karşı bir tür doğum günü saldırısının başarılı olacağı gösterilebilir, bunun için ya ortaya çıkan karma kodunun yeterince küçük olması (örneğin, 64 bit veya daha az) ya da daha büyük bir karma kodunun bağımsız alt kodlara ayrıştırılabilmesi gerekir [JUEN87].

Bu nedenle, karma işlemine yönelik başka yaklaşımların bulunmasına dikkat çekilmiştir. Bunların birçoğunun zayıf yönleri olduğu da gösterildi [MITC92].

11.5 GÜVENLİ HASH ALGORİTMASI (SHA)

Son yıllarda en yaygın kullanılan karma işlevi Güvenli Karma Algoritması (SHA) olmuştur. Gerçekten de, yaygın olarak kullanılan diğer karma işlevlerinin neredeyse hepsinin önemli kriptoanalitik zayıflıkları olduğu tespit edildiğinden, SHA 2005'e kadar az çok son kalan standart karma algoritmasıydı. SHA,

Tablo 11.3 SHA Parametrelerinin Karşılaştırılması

Algoritma	Mesaj Boyutu	Blok Boyutu	Kelime Boyutu	Mesaj Özet Boyutu
SHA-1	6 264	512	32	160
SHA-224	6 264	512	32	224
SHA-256	6 264	512	32	256
SHA-384	6 2128	1024	64	384
SHA-512	6 2128	1024	64	512
SHA-512/224	6 2128	1024	64	224
SHA-512/256	6 2128	1024	64	256

Not: Tüm boyutlar bit cinsinden ölçülmüştür.

Ulusal Standartlar ve Teknoloji Enstitüsü (NIST) tarafından yayınlanmış ve 1993 yılında federal bilgi işleme standardı (FIPS 180) olarak yayınlanmıştır. SHA'da zayıflıklar keşfedildiğinde, artık SHA-0 olarak bilinen, 1995 yılında FIPS 180-1 olarak revize edilmiş bir versiyonu yayınlanmış ve SHA-1 olarak anılmıştır . Gerçek standartlar belgesinin adı "Güvenli Karma Standart"tır. SHA, karma işlevi MD4'e dayanmaktadır ve tasarımı MD4'ü yakından modellemektedir.

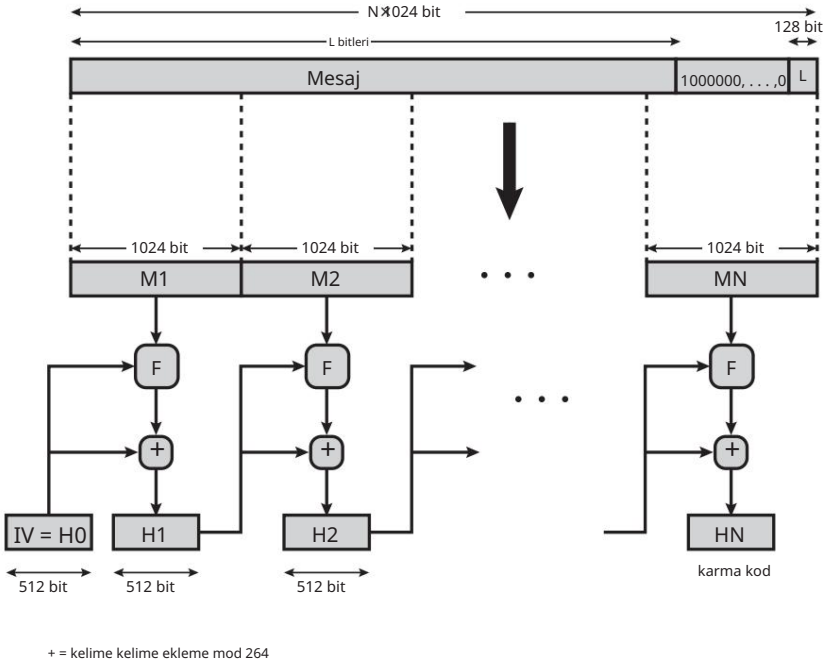
SHA-1, 160 bitlik bir karma değer üretir. 2002'de NIST, sırasıyla SHA-256, SHA-384 ve SHA-512 olarak bilinen 256, 384 ve 512 bitlik karma değer uzunluklarına sahip üç yeni SHA sürümünü tanımlayan FIPS 180-2 standardının revize edilmiş bir sürümünü üretti. Bu karma algoritmaları toplu olarak SHA-2 olarak bilinir.
Bu yeni sürümler aynı temel yapıya sahiptir ve SHA-1 ile aynı türde modüler aritmetik ve mantıksal ikili işlemleri kullanır. 2008'de 224 bitlik bir sürüm ekleyen FIP PUB 180-3 olarak revize edilmiş bir belge yayınlandı (Tablo 11.3).
NIST, 2015 yılında iki ek algoritma ekleyen FIPS 180-4'ü yayınladı: SHA-512/224 ve SHA-512/256. SHA-1 ve SHA-2, esasen FIPS 180-3'teki materyali kopyalayan ancak bir C kodu uygulaması ekleyen RFC 6234'te de belirtilmiştir.

2005 yılında NIST, SHA-1'in onayını aşamalı olarak kaldırma ve 2010 yılına kadar SHA-2'ye güvenmeye geçme niyetini duyurdu. Kısa bir süre sonra bir araştırma ekibi, 269 işlem kullanarak aynı SHA-1 karma değerini ileten iki ayrı mesajın bulunabileceği bir saldırıyı tanımladı; bu, daha önce bir SHA-1 karma değeriyle çarpışmayı bulmak için gereken 280 işlemden çok daha azdı [WANG05]. Bu sonuç, SHA-2'ye geçişi hızlandırmalıdır.

Bu bölümde SHA-512'nin bir tanımını sunuyoruz. Diğer sürümler oldukça benzerdir.

SHA-512 Mantığı

Algoritma, girdi olarak 2128 bitten daha az maksimum uzunluğa sahip bir mesaj alır ve çıktı olarak 512 bitlik bir mesaj özeti üretir. Giriş, 1024 bitlik bloklarda işlenir. Şekil 11.9, bir özet üretmek için bir mesajın genel işlenmesini gösterir.



Şekil 11.9 SHA-512 Kullanılarak Mesaj Özeti Oluşturma

Bu, Şekil 11.8'de gösterilen genel yapıyı takip eder. İşleme aşağıdaki adımlardan oluşur.

Adım 1 Dolgu bitleri ekleyin. Mesaj, uzunluğu 1024 modül 896'ya [uzunluk K 896(mod 1024)] denk olacak şekilde doldurulur. Mesaj zaten istenen uzunlukta olsa bile dolgu her zaman eklenir. Bu nedenle, dolgu bitlerinin sayısı 1 ila 1024 aralığındadır. Dolgu, gerekli sayıda 0 biti izleyen tek bir 1 bitinden oluşur.

Adım 2 Uzunluğu ekle. Mesaja 128 bitlik bir blok eklenir. Bu blok, işaretsiz 128 bitlik bir tam sayı (önce en önemli bayt) olarak ele alınır ve orijinal mesajın uzunluğunu bit cinsinden (dolgudan önce) içerir.

İlk iki adımın sonucu, 1024 bit uzunluğunda bir tam sayı katı olan bir mesaj üretir. Şekil 11.9'da, genişletilmiş mesaj, 1024 bitlik blokların dizisi olarak gösterilir; M1, M2, c , MN, böylece genişletilmiş mesajın toplam uzunluğu N * 1024 bittir.

Adım 3 Karma tamponunu başlatın. Karma işlevinin ara ve son sonuçlarını tutmak için 512 bitlik bir tampon kullanılır. Tampon sekiz adet 64 bitlik kayıt (a, b, c, d, e, f, g, h) olarak temsil edilebilir. Bu kayıtlar aşağıdaki 64 bitlik tam sayılara (onaltılık değerler) başlatılır:

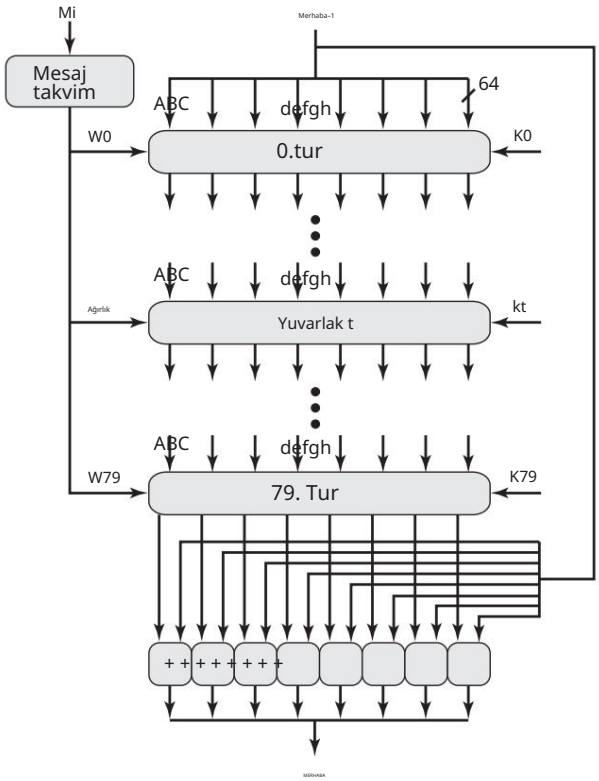
a = 6A09E667F3BCC908 e = 510E527FADE682D1
b = BB67AE8584CAA73B f = 9B05688C2B3E6C1F
c = 3C6EF372FE94F82B g = 1F83D9ABFB41BD6B
d = A54FF53A5F1D36F1 h = 5BE0CD19137E2179

Bu değerler, düşük adresli (en soldaki) bayt konumundaki bir kelimenin en önemli baytı olan büyük uçlu biçimde saklanır . Bu kelimeler, ilk sekiz asal sayının kareköklerinin kesirli kısımlarının ilk altmış dört biti alınarak elde edildi.

Adım 4 Mesajı 1024 bitlik (128 baytlık) bloklarda işleyin. Algoritmanın kalbi, 80 turdan oluşan bir modüldür; bu modül Şekil 11.9'da F olarak etiketlenmiştir. Mantık Şekil 11.10'da gösterilmektedir.

Her tur, 512 bitlik tampon değeri abcdefgh'yi girdi olarak alır ve tamponun içeriğini günceller. İlk tura girdi olarak, tampon ara karma değerin değeri Hi-1'e sahiptir. Her tur t, işlenen mevcut 1024 bitlik bloktan (Mi) türetilen 64 bitlik bir değer Wt kullanır . Bu değerler, daha sonra açıklanan bir mesaj programı kullanılarak türetilir. Her tur ayrıca , 0 ... t ... 79'un 80 turdan birini gösterdiği bir katkısal sabit Kt kullanır . Bu kelimeler, ilk 80 asal sayının küp kökünün kesirli kısımlarının ilk 64 bitini temsil eder.

Sabitler, giriş verilerindeki herhangi bir düzenliliği ortadan kaldırması gereken 64 bitlik desenlerin "rastgele" bir kümesini sağlar. Tablo 11.4 bu sabitleri onaltılık biçimde (soldan sağa) gösterir.



Şekil 11.10 Tek Bir 1024 Bitlik Bloğun SHA-512 İşlemesi

Tablo 11.4 SHA-512 Sabitleri

428a2f98d728ae22 7137449123ef65cd b5c0fbcfec4d3b2f e9b5dba58189dbbc
3956c25bf348b538 59f111f1b605d019 923f82a4af194f9b ab1c5ed5da6d8118
d807aa98a3030242 12835b0145706fbe 243185be4ee4b28c 550c7dc3d5ffb4e2
72be5d74f27b896f 80deb1fe3b1696b1 9bdc06a725c71235 c19bf174cf692694
e49b69c19ef14ad2 efbe4786384f25e3 0fc19dc68b8cd5b5 240ca1cc77ac9c65
2de92c6f592b0275 4a7484aa6ea6e483 5cb0a9dcbd41fbd4 76f988da831153b5
983e5152ee66dfab a831c66d2db43210 b00327c898fb213f bf597fc7beef0ee4
c6e00bf33da88fc2 d5a79147930aa725 06ca6351e003826f 142929670a0e6e70
27b70a8546d22ffc 2e1b21385c26c926 4d2c6dfc5ac42aed 53380d139d95b3df
650a73548baf63de 766a0abb3c77b2a8 81c2c92e47edaee6 92722c851482353b
a2bfe8a14cf10364 a81a664bbc423001 c24b8b70d0f89791 c76c51a30654be30
d192e819d6ef5218 d69906245565a910 f40e35855771202a 106aa07032bbd1b8
19a4c116b8d2d0c8 1e376c085141ab53 2748774cdf8eeb99 34b0bcb5e19b48a8
391c0cb3c5c95a63 4ed8aa4ae3418acb 5b9cca4f7763e373 682e6ff3d6b2b8a3
748f82ee5defb2fc 78a5636f43172f60 84c87814a1f0ab72 8cc702081a6439ec
90befffa23631e28 a4506cebde82bde9 bef9a3f7b2c67915 c67178f2e372532b
ca273eceea26619c d186b8c721c0c207 eada7dd6cde0eb1e f57d4f7fee6ed178
06f067aa72176fba 0a637dc5a2c898a6 113f9804bef90dae 1b710b35131c471b
28db77f523047d84 32caab7b40c72493 3c9ebe0a15c9bebc 431d67c49c100d4c
4cc5d4becb3e42b6 597f299cfc657e2a 5fcb6fab3ad6faec 6c44198c4a475817

Sekseninci turdaki çıktı, Hi üretmek için ilk turdaki girdiye (Hi-1) eklenir . Toplama, tampondaki sekiz kelimenin her biri için, karşılık gelen kelimelerin her biri Hi-1'de olmak üzere, modül 264 toplama kullanılarak bağımsız olarak yapılır .

Adım 5 Çıktı. Tüm N adet 1024 bitlik blok işlendikten sonra, N'inci aşamadan gelen çıktı 512 bitlik mesaj özetidir.

SHA-512'nin davranışını şu şekilde özetleyebiliriz:

$$H_0 = 4$$

$$\text{Merhaba} = \text{SUM64}(H_{i-1}, \text{abcdefghi})$$

$$MD = H_N$$

Neresi

IV = 3. adımda tanımlanan abcdefgh tamponunun başlangıç değeri

abcdefghi = i'inci mesajın son işlem turunun çıktısı blok = mesajdaki blok sayısı

N = (dolgu ve uzunluk alanları dahil)

SUM64 = giriş çiftinin her bir kelimesi üzerinde ayrı ayrı gerçekleştirilen modül 264 ekleme

MD = son mesaj özeti değeri

SHA-512 Yuvarlak Fonksiyon

Bir 512-bit bloğun işlenmesinin 80 adımının her birindeki mantığa daha detaylı bakalım (Şekil 11.11). Her tur aşağıdaki denklem kümesiyle tanımlanır:

$$T1 = h + Ch(e, f, g) + (a512\ 1\ e) + Ağırlık + Kt$$
$$T2 = (a512\ 0\ a) + Maj(a, b, c)$$
$$h = g$$
$$g = f$$
$$f = e$$
$$e = d + T1$$
$$d = c$$
$$c = b$$
$$b = bir$$
$$bir = T1 + T2$$

Neresi

^{T} = adım sayısı; 0 ... t ... 79

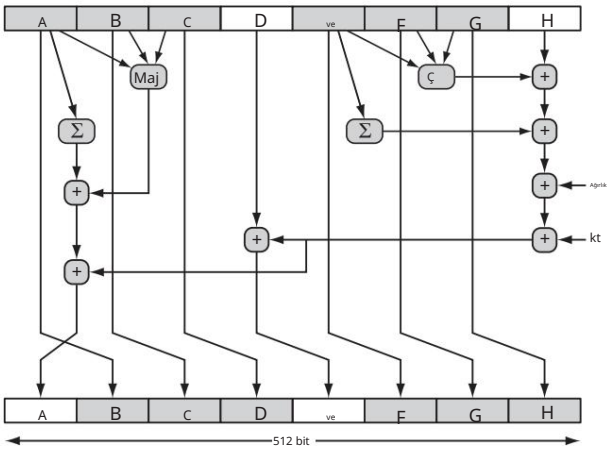
Ch(e, f, g) = (e VE f) (e VE g DEĞİL) koşullu fonksiyon: Eğer e ise f değilse g

Maj(a, b, c) = (a VE b) (a VE c) (b VE c) fonksiyon yalnızca argümanların çoğunluğunun (iki veya üçünün) doğru olması durumunda doğrudur

(Σ512 0 a) = ROTR28(a) ROTR34(a) ROTR39(a)

(Σ512 1 e) = ROTR14(e) ROTR18(e) ROTR41(e)

ROTRn (x) = 64 bitlik argüman x'in n bitlik dairesel sağa kaydırılması (döndürülmesi)



Şekil 11.11 Temel SHA-512 İşlemi (tek tur)

Wt = geçerli 1024 bitlik giriş bloğundan türetilen 64 bitlik bir sözcük

Kt = 64 bitlik bir ekleme sabiti + =
modül 264 ekleme

Yuvarlak fonksiyon hakkında iki gözlem yapılabilir.

1. Yuvarlama fonksiyonunun çıktısının sekiz kelimesinden altısı , rotasyon yoluyla basitçe (b, c, d, f, g, h) permutasyonunu içerir . Bu, Şekil 11.11'deki gölgelendirme ile gösterilir.

2. Çıktı sözcüklerinden yalnızca ikisi (a, e) ikameyle üretilir. e sözcüğü , girdi değişkenlerinin (d, e, f, g, h) yanı sıra yuvarlak sözcük Wt ve sabit Kt'nin bir fonksiyonudur. a sözcüğü , d hariç tüm girdi değişkenlerinin yanı sıra yuvarlak sözcük Wt ve sabit Kt'nin bir fonksiyonudur .

Geriye 64 bitlik sözcük değerleri Wt'nin 1024 bitlik mesajdan nasıl türetildiğini göstermek kalıyor. Şekil 11.12 eşlemeyi göstermektedir. Wt'nin ilk 16 değeri doğrudan geçerli bloğun 16 sözcüğünden alınmıştır. Kalan değerler şu şekilde tanımlanmıştır

$$\text{Ağırlık} = s1_{512}(\text{Ağırlık}-2) + \text{Ağırlık}-7 + s0_{512}(\text{Ağırlık}-15) + \text{Ağırlık}-16$$

Neresi

$$s0_{512}(x) = \text{ROTR1}(x) \quad \text{ROTR8}(x) \quad \text{SHR7}(x)$$

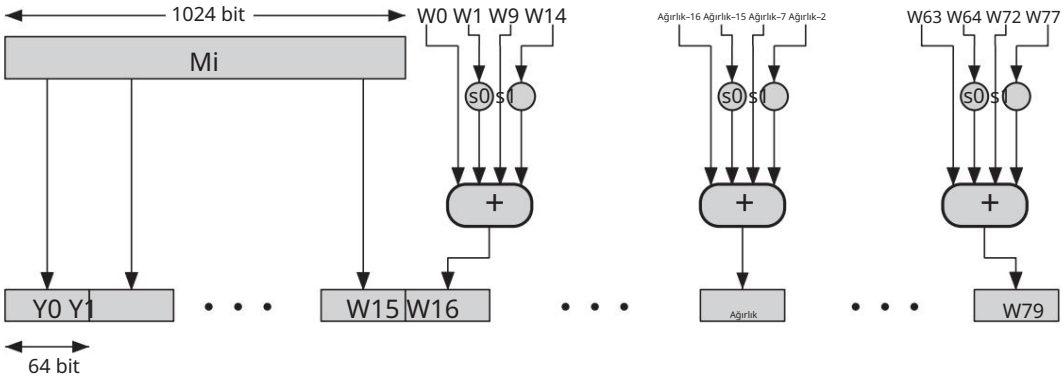
$$s1_{512}(x) = \text{ROTR19}(x) \quad \text{ROTR61}(x) \quad \text{SHR6}(x)$$

ROTRn (x) = 64 bitlik argüman x'in n bitlik dairesel sağa kaydırılması (döndürülmesi)

SHRn (x) = 64 bitlik argüman x'in n bit sağa kaydırılması ve solda sıfırlarla doldurulması + = modül 264'e göre

toplama

Böylece, işlemenin ilk 16 adımında, Wt değeri mesaj bloğundaki karşılık gelen kelimeye eşittir. Kalan 64 adım için, Wt değeri, Wt'nin önceki değerlerinden dördünün XOR'unun bir bitlik dairesel sola kaydırılmasından oluşur ve bu değerlerden ikisi kaydırma ve döndürme işlemlerine tabi tutulur.



Şekil 11.12 Tek Bloğun SHA-512 İşlemesi için 80 Kelimelik Giriş Dizisinin Oluşturulması

Bu, sıkıştırılan ileti bloklarına çok fazla yedeklilik ve karşılıklı bağımlılık getirir ve bu da aynı sıkıştırma işlevi çıktısına eşlenen farklı bir ileti bloğu bulma görevini karmaşıklaştırır. Şekil 11.13, SHA-512 mantığını özetlemektedir.

SHA-512 algoritması, karma kodunun her bitinin girdinin her bitinin bir fonksiyonu olduğu özelliğine sahiptir. Temel fonksiyon F'nin karmaşık tekrarı, iyi karıştırılmış sonuçlar üretir; yani, benzer düzenlilikler gösterseler bile, rastgele seçilen iki mesajın aynı karma koduna sahip olması olası değildir. SHA-512'de henüz yayınlanmamış gizli bir zayıflık olmadığı sürece, aynı mesaj özetine sahip iki mesaj bulma zorluğu 2256 işlem mertebesindeyken, belirli bir özete sahip bir mesaj bulma zorluğu 2512 işlem mertebesindedir .

Örnek

Burada FIPS 180'deki bir örneğe dayalı bir örnek ekliyoruz. Üç ASCII karakterinden oluşan tek bloklu bir mesajın karmasını oluşturmak istiyoruz: "abc", aşağıdaki 24 bitlik ikili dizeye eşdeğerdir:

01100001 01100010 01100011

SHA algoritmasının 1. adımında, mesajın 1024 modülüne göre 896'ya denk gelen bir uzunluğa doldurulduğunu hatırlayalım. Bu tek blok durumunda, doldurma 896 - 24 = 872 bitten oluşur ve bu da bir "1" biti ve bunu izleyen 871 "0" bitinden oluşur. Daha sonra mesaja, orijinal mesajın uzunluğunu bit cinsinden (dolgudan önce) içeren 128 bit uzunluğunda bir değer eklenir. Orijinal uzunluk 24 bit veya onaltılık değer olarak 18'dir. Tüm bunları bir araya koyduğumuzda, onaltılık olarak 1024 bit uzunluğundaki mesaj bloğu

6162638000000000 0000000000000000 0000000000000000 0000000000000000
0000000000000000 0000000000000000 0000000000000000 0000000000000000
0000000000000000 0000000000000000 0000000000000000 0000000000000000
0000000000000000 0000000000000000 0000000000000000 0000000000000018

Bu blok, aşağıdaki gibi görünen mesaj planının W0, . . . , W15 sözcüklerine atanmıştır.

W0 = 6162638000000000	W8 = 0000000000000000
W1 = 0000000000000000	W9 = 0000000000000000
W2 = 0000000000000000	W10 = 0000000000000000
W3 = 0000000000000000	W11 = 0000000000000000
W4 = 0000000000000000	W12 = 0000000000000000
W5 = 0000000000000000	W13 = 0000000000000000
W6 = 0000000000000000	W14 = 0000000000000000
W7 = 0000000000000000	W15 = 0000000000000018