Funciones Resumen

U.D. Computación

DSIC - UPV

Contenido

Bibliografía

- Handbook of applied crytography. *A. J. Menezes, P. C. van Oorshot and S. A. Vanstone*. CRC Press. 1996. (Capítulo 9)
- Encyclopedia of cryptography and security. *Henk C. A. van Tilborg (Ed.)*. Springer. 2005.

Hashing

Funciones resumen
Descripción

- Sea un alfabeto binario A. Una función hash es una función h donde el dominio está formado por mensajes de longitud arbitraria y el rango es $R \subseteq A^n$ para un valor n prefijado

Funciones resumen
Descripción

- Sea un alfabeto binario A. Una función hash es una función h donde el dominio está formado por mensajes de longitud arbitraria y el rango es $R \subseteq A^n$ para un valor n prefijado
- $|D| >> |R|$, por lo tanto h no es inyectiva, produciendose colisiones

Funciones resumen
Descripción

- Sea un alfabeto binario A. Una función hash es una función h donde el dominio está formado por mensajes de longitud arbitraria y el rango es $R \subseteq A^n$ para un valor n prefijado
- $|D| >> |R|$, por lo tanto h no es inyectiva, produciendose colisiones
- Para una h aleatoria, si $D = A^t$ $(t > n)$, entonces a cada resumen le corresponden 2^{t-n} mensajes. La probabilidad de que dos mensajes den como resultado la misma salida es 2^{-n}

Funciones resumen
Descripción

- Sea un alfabeto binario A. Una función hash es una función h donde el dominio está formado por mensajes de longitud arbitraria y el rango es $R \subseteq A^n$ para un valor n prefijado
- $|D| >> |R|$, por lo tanto h no es inyectiva, produciendose colisiones
- Para una h aleatoria, si $D = A^t$ $(t > n)$, entonces a cada resumen le corresponden 2^{t-n} mensajes. La probabilidad de que dos mensajes den como resultado la misma salida es 2^{-n}

Propiedades:

- Fácil de computar
- Compresión: Cualquier entrada x de longitud arbitraria, se convierte en $h(x)$ de longitud fija

Funciones resumen
Propiedades (deseables) de las funciones hash

- Funciones hash unidireccionales (OWHF)
- Funciones hash resistentes a colisiones (CRHF)

Funciones resumen
Propiedades (deseables) de las funciones hash

- Funciones hash unidireccionales (OWHF)
  - Resistencia a la 1^a preimagen (one way hashing) Dado un valor hash y, es computacionalmente intratable encontrar un $x \in D$ tal que $h(x) = y$
  - Resistencia a la 2^a preimagen (Weak colision resistance) Dado un mesaje x es computacionalmente intratable encontrar un segundo mensaje x', $(x \neq x')$, tal que $h(x) = h(x')$
- Funciones hash resistentes a colisiones (CRHF)

Funciones resumen
Propiedades (deseables) de las funciones hash

- Funciones hash unidireccionales (OWHF)
  - Resistencia a la 1^a preimagen (one way hashing) Dado un valor hash y, es computacionalmente intratable encontrar un $x \in D$ tal que $h(x) = y$
  - Resistencia a la 2^a preimagen (Weak colision resistance) Dado un mesaje x es computacionalmente intratable encontrar un segundo mensaje x', $(x \neq x')$, tal que $h(x) = h(x')$
- Funciones hash resistentes a colisiones (CRHF)
  - Resistencia a las colisiones (strong colision resistance) Encontrar dos mensajes $x, x' \in D$, $(x \neq x')$, tales que $h(x) = h(x')$ es computacionalmente intratable

Funciones resumen
Relación entre propiedades

- Resistencia a la primera preimagen no asegura resistencia a la segunda preimagen.
- Resistencia a la segunda preimagen no asegura resistencia a la primera preimagen.
- CRHF implica resistencia a la segunda preimagen.
- CRHF no garantiza resistencia la primera preimagen.

Seguridad

Funciones resumen
Seguridad: El ataque del cumpleaños

- Sea $h : X \to Z$ donde $|X| \geq 2|Z|$ donde $|X| = m$ y $|Z| = n$. Por lo tanto existirán al menos n colisiones
- Asumimos que para cualquier $z \in Z$ se cumple que $|h^{-1}(z)| = m/n$. En otro caso se incrementa la probabilidad de encontrar una colisión

Funciones resumen
Seguridad: El ataque del cumpleaños

- Sea $h : X \to Z$ donde $|X| \geq 2|Z|$ donde $|X| = m$ y $|Z| = n$. Por lo tanto existirán al menos n colisiones
- Asumimos que para cualquier $z \in Z$ se cumple que $|h^{-1}(z)| = m/n$. En otro caso se incrementa la probabilidad de encontrar una colisión
- Dados k mensajes al azar $(x_1, x_2, \ldots, x_k)$, queremos calcular la probabilidad de que no se produzca colisión:

Funciones resumen
Seguridad: El ataque del cumpleaños

- Sea $h : X \to Z$ donde $|X| \geq 2|Z|$ donde $|X| = m$ y $|Z| = n$. Por lo tanto existirán al menos n colisiones
- Asumimos que para cualquier $z \in Z$ se cumple que $|h^{-1}(z)| = m/n$. En otro caso se incrementa la probabilidad de encontrar una colisión
- Dados k mensajes al azar $(x_1, x_2, \ldots, x_k)$, queremos calcular la probabilidad de que no se produzca colisión:

$$\left(1 - \frac{1}{n}\right)\left(1 - \frac{2}{n}\right)\ldots\left(1 - \frac{k-1}{n}\right) = \prod_{i=1}^{k-1}\left(1 - \frac{i}{n}\right)$$

Funciones resumen
Seguridad: El ataque del cumpleaños

- Sea $h : X \to Z$ donde $|X| \geq 2|Z|$ donde $|X| = m$ y $|Z| = n$. Por lo tanto existirán al menos n colisiones
- Asumimos que para cualquier $z \in Z$ se cumple que $|h^{-1}(z)| = m/n$. En otro caso se incrementa la probabilidad de encontrar una colisión
- Dados k mensajes al azar $(x_1, x_2, \ldots, x_k)$, queremos calcular la probabilidad de que no se produzca colisión:

$$\left(1 - \frac{1}{n}\right)\left(1 - \frac{2}{n}\right)\ldots\left(1 - \frac{k-1}{n}\right) = \prod_{i=1}^{k-1}\left(1 - \frac{i}{n}\right)$$

$$e^{-x} = 1 - x + \frac{x^2}{2!} - \frac{x^3}{3!}\ldots$$

Funciones resumen
Seguridad: El ataque del cumpleaños

- Sea $h : X \to Z$ donde $|X| \geq 2|Z|$ donde $|X| = m$ y $|Z| = n$. Por lo tanto existirán al menos n colisiones
- Asumimos que para cualquier $z \in Z$ se cumple que $|h^{-1}(z)| = m/n$. En otro caso se incrementa la probabilidad de encontrar una colisión
- Dados k mensajes al azar $(x_1, x_2, \ldots, x_k)$, queremos calcular la probabilidad de que no se produzca colisión:

$$\left(1 - \frac{1}{n}\right)\left(1 - \frac{2}{n}\right)\ldots\left(1 - \frac{k-1}{n}\right) = \prod_{i=1}^{k-1}\left(1 - \frac{i}{n}\right)$$

$e^{-x} = 1 - x + \frac{x^2}{2!} - \frac{x^3}{3!}\ldots$, por lo que, si $x \to 0$ entonces $1 - x \approx e^{-x}$

Funciones resumen
Seguridad: El ataque del cumpleaños

- Sea $h : X \to Z$ donde $|X| \geq 2|Z|$ donde $|X| = m$ y $|Z| = n$. Por lo tanto existirán al menos n colisiones
- Asumimos que para cualquier $z \in Z$ se cumple que $|h^{-1}(z)| = m/n$. En otro caso se incrementa la probabilidad de encontrar una colisión
- Dados k mensajes al azar $(x_1, x_2, \ldots, x_k)$, queremos calcular la probabilidad de que no se produzca colisión:

$$\left(1 - \frac{1}{n}\right)\left(1 - \frac{2}{n}\right)\ldots\left(1 - \frac{k-1}{n}\right) = \prod_{i=1}^{k-1}\left(1 - \frac{i}{n}\right)$$

$e^{-x} = 1 - x + \frac{x^2}{2!} - \frac{x^3}{3!}\ldots$, por lo que, si $x \to 0$ entonces $1 - x \approx e^{-x}$, con lo que la probabilidad de no colisión queda:

$$\prod_{i=1}^{k-1}\left(1 - \frac{i}{n}\right) \approx \prod_{i=1}^{k-1} e^{\frac{-i}{n}} = e^{\frac{-k(k-1)}{2n}}$$

Funciones resumen
Seguridad: El ataque del cumpleaños

- La probabilidad de colisión se aproxima a $1 - e^{\frac{-k(k-1)}{2n}}$. ¿Cuantos mensajes hay que escoger para que la probabilidad de colisión sea ϵ?

Funciones resumen
Seguridad: El ataque del cumpleaños

- La probabilidad de colisión se aproxima a $1 - e^{\frac{-k(k-1)}{2n}}$. ¿Cuantos mensajes hay que escoger para que la probabilidad de colisión sea ϵ?

$$e^{\frac{-k(k-1)}{2n}} \approx 1 - \epsilon$$

Funciones resumen
Seguridad: El ataque del cumpleaños

- La probabilidad de colisión se aproxima a $1 - e^{\frac{-k(k-1)}{2n}}$. ¿Cuantos mensajes hay que escoger para que la probabilidad de colisión sea ϵ?

$$e^{\frac{-k(k-1)}{2n}} \approx 1 - \epsilon \Rightarrow \frac{k(k-1)}{2n} \approx -\ln(1-\epsilon)$$

Funciones resumen
Seguridad: El ataque del cumpleaños

- La probabilidad de colisión se aproxima a $1 - e^{\frac{-k(k-1)}{2n}}$. ¿Cuantos mensajes hay que escoger para que la probabilidad de colisión sea ϵ?

$$e^{\frac{-k(k-1)}{2n}} \approx 1 - \epsilon \Rightarrow \frac{k(k-1)}{2n} \approx -\ln(1-\epsilon) \Rightarrow$$

$$\Rightarrow k^2 - k \approx 2n \ln \frac{1}{1-\epsilon}$$

Funciones resumen
Seguridad: El ataque del cumpleaños

- La probabilidad de colisión se aproxima a $1 - e^{\frac{-k(k-1)}{2n}}$. ¿Cuantos mensajes hay que escoger para que la probabilidad de colisión sea ϵ?

$$e^{\frac{-k(k-1)}{2n}} \approx 1 - \epsilon \Rightarrow \frac{k(k-1)}{2n} \approx -\ln(1-\epsilon) \Rightarrow$$

$$\Rightarrow k^2 - k \approx 2n \ln \frac{1}{1-\epsilon} \Rightarrow k \approx \sqrt{2n \ln \frac{1}{1-\epsilon}}$$

Funciones resumen
Seguridad: El ataque del cumpleaños

- La probabilidad de colisión se aproxima a $1 - e^{\frac{-k(k-1)}{2n}}$. ¿Cuantos mensajes hay que escoger para que la probabilidad de colisión sea ϵ?

$$e^{\frac{-k(k-1)}{2n}} \approx 1 - \epsilon \Rightarrow \frac{k(k-1)}{2n} \approx -\ln(1-\epsilon) \Rightarrow$$

$$\Rightarrow k^2 - k \approx 2n \ln \frac{1}{1-\epsilon} \Rightarrow k \approx \sqrt{2n \ln \frac{1}{1-\epsilon}}$$

- Si $\epsilon = 0.5$ entonces $k \approx \sqrt{n(2 \ln 2)} \approx 1.17\sqrt{n}$

Funciones resumen
Seguridad: El ataque del cumpleaños

- La probabilidad de colisión se aproxima a $1 - e^{\frac{-k(k-1)}{2n}}$. ¿Cuantos mensajes hay que escoger para que la probabilidad de colisión sea ϵ?

$$e^{\frac{-k(k-1)}{2n}} \approx 1 - \epsilon \Rightarrow \frac{k(k-1)}{2n} \approx -\ln(1-\epsilon) \Rightarrow$$

$$\Rightarrow k^2 - k \approx 2n \ln \frac{1}{1-\epsilon} \Rightarrow k \approx \sqrt{2n \ln \frac{1}{1-\epsilon}}$$

- Si $\epsilon = 0.5$ entonces $k \approx \sqrt{n(2\ln 2)} \approx 1.17\sqrt{n}$
- Tomando $n = 365$ obtenemos $k = 23$

Funciones resumen
Seguridad: El ataque del cumpleaños

- La probabilidad de colisión se aproxima a $1 - e^{\frac{-k(k-1)}{2n}}$. ¿Cuantos mensajes hay que escoger para que la probabilidad de colisión sea ϵ?

$$e^{\frac{-k(k-1)}{2n}} \approx 1 - \epsilon \Rightarrow \frac{k(k-1)}{2n} \approx -\ln(1-\epsilon) \Rightarrow$$

$$\Rightarrow k^2 - k \approx 2n \ln \frac{1}{1-\epsilon} \Rightarrow k \approx \sqrt{2n \ln \frac{1}{1-\epsilon}}$$

- Si $\epsilon = 0.5$ entonces $k \approx \sqrt{n(2 \ln 2)} \approx 1.17\sqrt{n}$
- Tomando $n = 365$ obtenemos $k = 23$
- Con el mismo $n = 365$, tomando $\epsilon = 0.95$ entonces $k = 47$

Funciones resumen
Seguridad: Algoritmo de Yuval

Firma de un mensaje ilegítimo con la firma de un mensaje inofensivo

Require: Par de mensajes legítimo e ilegítimo: x_l, x_i
Require: Función hash h de m bits
Ensure: Mensajes x'_l y x'_i (con cambios menores respecto la entrada), tales que $h(x'_l) = h(x'_i)$

Firma de un mensaje ilegítimo con la firma de un mensaje inofensivo

Require: Par de mensajes legítimo e ilegítimo: x_l, x_i
Require: Función hash h de m bits
Ensure: Mensajes x'_l y x'_i (con cambios menores respecto la entrada), tales que $h(x'_l) = h(x'_i)$

1: Generar $t = 2^{m/2}$ modificaciones menores de x_l
2: Computar el resumen y almacenar los pares $(x'_l, h(x'_l))$
3: **while** no se encuentre colisión **do** {probable en t intentos}
4: Generar x'_i modificación menor de x_i y computar $h(x'_i)$
5: Buscar si existe x'_l tal que $h(x'_l) = h(x'_i)$
6: **end while**
7: **return** (x'_l, x'_i)

Funciones resumen
Seguridad: Algoritmo de Yuval

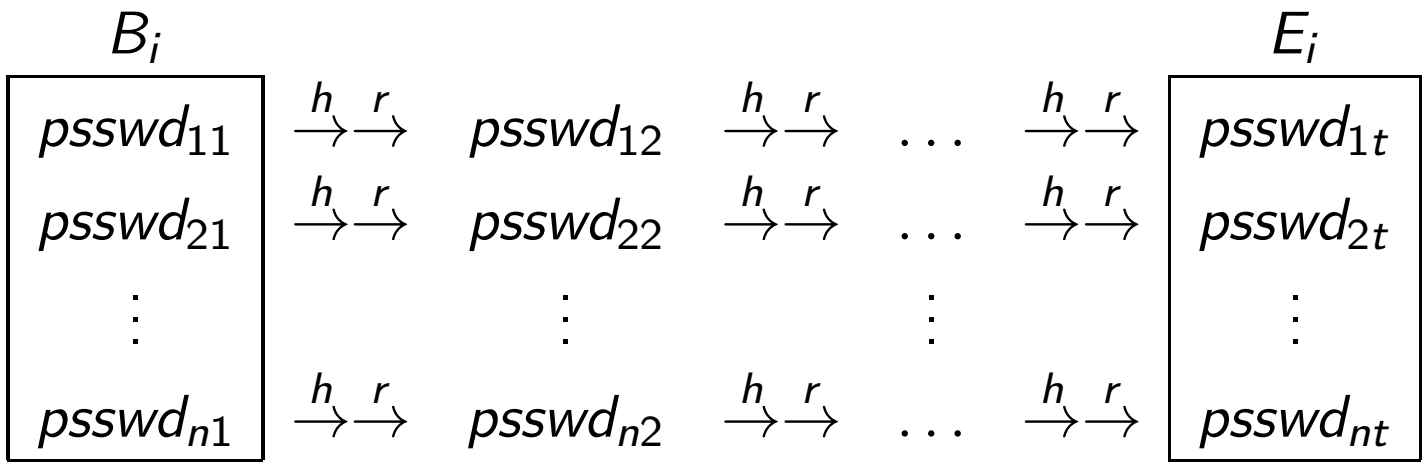
- El ataque del cumpleaños acota inferiormente la talla del resumen
- Si el resumen es de 40 bits, entonces con probabilidad 0.5, se puede obtener una colisión con 2^{20} mensajes al azar (aproximadamente 10^6 mensajes)
- Para una OWHF de n bits se necesita exploración exhaustiva ($n = 80$ se considera mínimo)
- Una CRHF de n bits es sensible a ataques basados en el algoritmo de Yuval. ($n = 160$ se considera mínimo)
- MD4/5 obtiene un resumen de 128 bits
- DSS considera 160 bits. La familia SHA permite desde 256/512 bits

Funciones resumen
Seguridad: Ataque arcoiris

- Diseñado para encontrar colisiones de passwords
- Utiliza una aproximación time-memory trade-off (TMTO)
- Sensible a la función resumen utilizada. Distintas funciones necesitan tablas arcoiris distintas

Require: Una función resumen h
Require: Una función recodificante r
Require: t longitud de la secuencia
Require: n número de entradas
Ensure: Un vector rainbow para la función h.
Método
while La tabla no contenga n entradas **do**
Escoger P_1 un password al azar.
for $i = 1$ **to** $t - 1$ **do**
$P_{i+1} = r(h(P_i))$
end for
Almacenar $\langle P_1,\ P_t \rangle$ en la tabla
end while
FinMétodo.

Funciones resumen
Seguridad: Ataque arcoiris



Funciones resumen
Seguridad: Ataque arcoiris

- El inconveniente principal es la obtención de colisiones durante la construcción de la tabla
- Para resolver esto se consideran varias soluciones (distintas funciones recodificantes, introducción de *puntos de sincronización*...)
- Mediante *salting* puede prevenirse el uso de estas técnicas

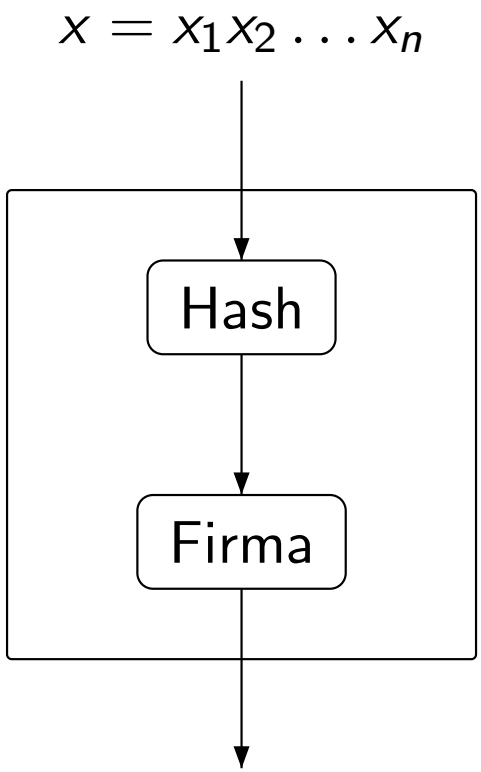
Funciones resumen
Randomized Hashing

- *Strengthening digital Signatures via Randomized Hashing*. Shai Halevi and H. Krawczyk. Cripto 2006.
- Propuesta que complementa el proceso de firma contra ataques por colisión de las funciones de resumen.
- Independientemente de la función resumen, la seguridad se mantiene aunque se disponga de mensajes cuyo resumen colisione.
- NIST Special Publication 800-106 (2009) recoge el procedimiento pese a no incluirlo en el estandar de firma.

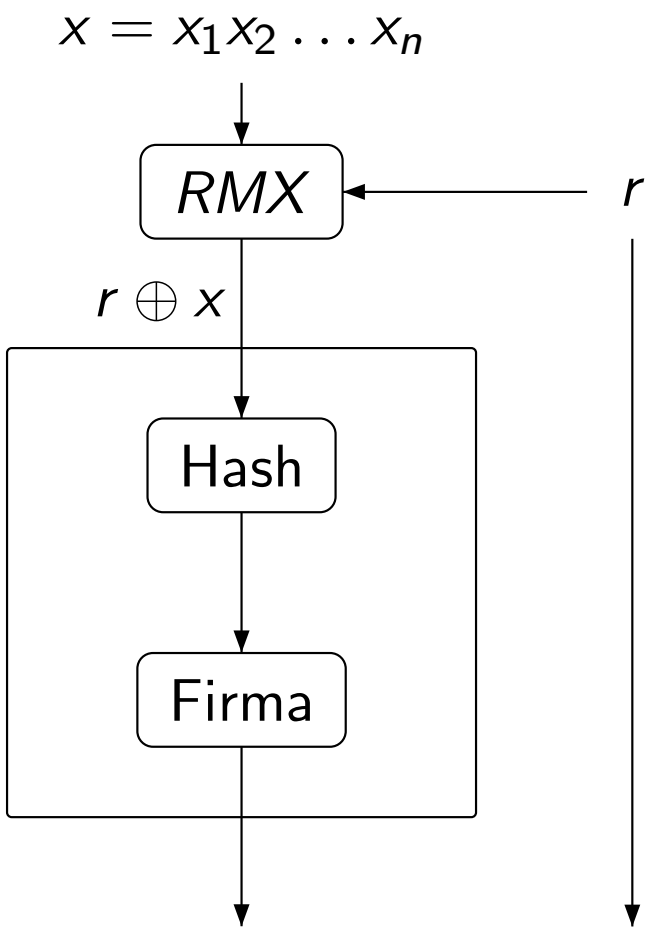
Funciones resumen
Randomized Hashing

- Considera una secuencia binaria aleatoria r que se genera para cada mensaje firmado.
- Esta secuencia modifica el mensaje (mediante una operación *XOR*) previamente al cálculo de su resumen, y su posterior firma.
- El ataque al esquema resultante equivale a, dado un mensaje, encontrar una primera preimagen.

Funciones resumen
Randomized Hashing



Funciones resumen
Randomized Hashing



Implementación de funciones hash

Implementación de funciones hash
Esquemas de implementación

- Habitualmente se computa el resumen considerando un *'estado'* que se modifica iterativamente al procesar el mensaje.
- No consideramos aproximaciones no diseñadas específicamente como funciones de resumen.

Implementación de funciones hash
El esquema Damgard-Merkle

Require: Mensaje x (binario) de longitud arbitraria n
Ensure: Resumen del mensaje $h(x)$ de longitud fija k

$//n >> k$

1: Dividir el mensaje en una serie de bloques de tamaño fijo t $//x = x_1 x_2 \ldots x_m$
2: Completar el último bloque con una secuencia de bits 0 si es necesario $//$*padding* del mensaje
3: Añadir un bloque extra que contenga la longitud del mensaje original módulo t $//$*length-block*
4: $h_0 = 0_1 0_2 \ldots 0_k$ // Iniciación del resumen
5: **for** $i = 1$ **to** $m + 1$ **do**
6: $\quad h_i = f(h_0, x_i)$
7: **end for**
8: **return** h_i

Require: Mensaje x (binario) de longitud arbitraria n.
Ensure: Mensaje x' (binario) de longitud múltiplo de 512.

1: Obtener x' extendiendo el mensaje (añadiendo un 1 seguido de suficientes 0s) para que su longitud sea congruente con 448 módulo 512

//Padding

2: Añadir a x' la representación binaria del mensaje módulo 2^{64} (los 64 bits menos significativos)

//*length-block*

Implementación de funciones hash
MD4: Algoritmo

Require: Mensaje x (binario) de longitud n múltiplo de 512.
Ensure: $MD4(x)$ //Resumen de 128 bits

1: $A = 67452301_{hex}$; $B = EFCDAB89_{hex}$;
   $C = 98BADCFE_{hex}$; $D = 10325476_{hex}$;
2: **for all** bloque de 512 bits **do**
3: Dividir el $Bloque_i$ en 16 palabras de 32 bits
   $Bloque_i = X[0], X[1], \ldots, X[15]$
4: $AA = A$ $BB = B$ $CC = C$ $DD = D$
5: **Round 1**
6: **Round 2**
7: **Round 3**
8: $A = A + AA$ $B = B + BB$
   $C = C + CC$ $D = D + DD$
9: **end for**
10: **return** $ABCD$

Implementación de funciones hash
MD4: Round 1

1. $A = (A + f(B, C, D) + X[0]) <<< 3$
2. $D = (D + f(A, B, C) + X[1]) <<< 7$
3. $C = (C + f(D, A, B) + X[2]) <<< 11$
4. $B = (B + f(C, D, A) + X[3]) <<< 19$
5. $A = (A + f(B, C, D) + X[4]) <<< 3$
6. $D = (D + f(A, B, C) + X[5]) <<< 7$
7. $C = (C + f(D, A, B) + X[6]) <<< 11$
8. $B = (B + f(C, D, A) + X[7]) <<< 19$
9. $A = (A + f(B, C, D) + X[8]) <<< 3$
10. $D = (D + f(A, B, C) + X[9]) <<< 7$
11. $C = (C + f(D, A, B) + X[10]) <<< 11$
12. $B = (B + f(C, D, A) + X[11]) <<< 19$
13. $A = (A + f(B, C, D) + X[12]) <<< 3$
14. $D = (D + f(A, B, C) + X[13]) <<< 7$
15. $C = (C + f(D, A, B) + X[14]) <<< 11$ // $f(X, Y, Z) = (X \ AND \ Y) \ OR \ (\overline{X} \ AND \ Z)$
16. $B = (B + f(C, D, A) + X[15]) <<< 19$ // Si X entonces Y, si no Z

Implementación de funciones hash
MD4: Round 2

1. $A = (A + g(B, C, D) + X[0] + 5A827999) <<< 3$
2. $D = (D + g(A, B, C) + X[4] + 5A827999) <<< 7$
3. $C = (C + g(D, A, B) + X[8] + 5A827999) <<< 11$
4. $B = (B + g(C, D, A) + X[12] + 5A827999) <<< 19$
5. $A = (A + g(B, C, D) + X[1] + 5A827999) <<< 3$
6. $D = (D + g(A, B, C) + X[5] + 5A827999) <<< 7$
7. $C = (C + g(D, A, B) + X[9] + 5A827999) <<< 11$
8. $B = (B + g(C, D, A) + X[13] + 5A827999) <<< 19$
9. $A = (A + g(B, C, D) + X[2] + 5A827999) <<< 3$
10. $D = (D + g(A, B, C) + X[6] + 5A827999) <<< 7$
11. $C = (C + g(D, A, B) + X[10] + 5A827999) <<< 11$
12. $B = (B + g(C, D, A) + X[14] + 5A827999) <<< 19$
13. $A = (A + g(B, C, D) + X[3] + 5A827999) <<< 3$
14. $D = (D + g(A, B, C) + X[7] + 5A827999) <<< 7$
15. $C = (C + g(D, A, B) + X[11] + 5A827999) <<< 11$
16. $B = (B + g(C, D, A) + X[15] + 5A827999) <<< 19$

// $g(X, Y, Z) = (X\ AND\ Y)\ OR$
// $(X\ AND\ Z)\ OR\ (Y\ AND\ Z)$
// función mayoría

Implementación de funciones hash
MD4: Round 3

1. $A = (A + h(B, C, D) + X[0] + 6ED9EBA1) <<< 3$
2. $D = (D + h(A, B, C) + X[8] + 6ED9EBA1) <<< 9$
3. $C = (C + h(D, A, B) + X[4] + 6ED9EBA1) <<< 11$
4. $B = (B + h(C, D, A) + X[12] + 6ED9EBA1) <<< 15$
5. $A = (A + h(B, C, D) + X[2] + 6ED9EBA1) <<< 3$
6. $D = (D + h(A, B, C) + X[10] + 6ED9EBA1) <<< 9$
7. $C = (C + h(D, A, B) + X[6] + 6ED9EBA1) <<< 11$
8. $B = (B + h(C, D, A) + X[14] + 6ED9EBA1) <<< 15$
9. $A = (A + h(B, C, D) + X[1] + 6ED9EBA1) <<< 3$
10. $D = (D + h(A, B, C) + X[9] + 6ED9EBA1) <<< 9$
11. $C = (C + h(D, A, B) + X[5] + 6ED9EBA1) <<< 11$
12. $B = (B + h(C, D, A) + X[13] + 6ED9EBA1) <<< 15$
13. $A = (A + h(B, C, D) + X[3] + 6ED9EBA1) <<< 3$
14. $D = (D + h(A, B, C) + X[11] + 6ED9EBA1) <<< 9$
15. $C = (C + h(D, A, B) + X[7] + 6ED9EBA1) <<< 11$
16. $B = (B + h(C, D, A) + X[15] + 6ED9EBA1) <<< 15$

// $h(X, Y, Z) = (X \oplus Y \oplus Z)$

Implementación de funciones hash
La familia MDx: Seguridad

- Muy pronto se desarrollan ataques a MD4 que no ejecutaban la primera o tercera vuelta. Se considera que MD4 está roto.
- MD5 preprocesa del mismo modo que MD4 el mensaje. El algoritmo sigue básicamente el esquema MD4 ejecutando un cuarto round adicional (cada uno con 20 operaciones).
- En 2004 se publica un resultado que describe un procedimiento para encontrar mensajes distintos x y x' tales que $h_{MD5}(x) = h_{MD5}(x')$.
- MD5 sigue siendo fiable como medio para validar la no-modificación de ficheros propios cuyo contenido conocemos. Deja de ser fiable como base para cualquier tipo de firmas o certificados.
- DNI electrónico utiliza SHA-1/RSA.
  Se recomienda no firmar hashes MD5

Funciones resumen
La función SHA-1

- Desarrollada por la NSA en 1993. Considera mensajes de longitud menor de 2^{64} bits, produciendo inicialmente un resumen de 160 bits.

Funciones resumen
La función SHA-1

- Desarrollada por la NSA en 1993. Considera mensajes de longitud menor de 2^{64} bits, produciendo inicialmente un resumen de 160 bits.
- En 1995, NSA sustituye SHA por SHA-1, básicamente igual al original (SHA-0) con cambios menores.

Funciones resumen
La función SHA-1

- Desarrollada por la NSA en 1993. Considera mensajes de longitud menor de 2^{64} bits, produciendo inicialmente un resumen de 160 bits.
- En 1995, NSA sustituye SHA por SHA-1, básicamente igual al original (SHA-0) con cambios menores.
- Preproceso del mensaje identico al efectuado por MD4 excepto en que añade la longitud del mensaje (2 bloques de 32 bits, el más significativo primero) en lugar de los 64 bits menos significativos.

Funciones resumen
La función SHA-1. Algoritmo

Require: Mensaje x (binario) de longitud n múltiplo de 512.
Ensure: $SHA-1(x)$ //Resumen de 160 bits

1: $A = 5A827999_{hex}$ $B = 6ED9EBA1_{hex}$ $C = 8F1BBCDC_{hex}$
$D = CA62C1D6_{hex}$ $E = C3D2E1F0_{hex}$
2: **for all** bloque de 512 bits **do**
3: $Bloque_i = X[0], X[1], \ldots X[15]$ //palabras de 32 bits.
4: $AA = A \quad BB = B \quad CC = C \quad DD = D \quad EE = E$
5: **Round 1**
6: **Round 2**
7: **Round 3**
8: **Round 4** //20 operaciones cada bloque
9: $A = A + AA \quad B = B + BB \quad C = C + CC$
$D = D + DD \quad E = E + EE$
10: **end for**
11: **return** $ABCDE$

- Se ha propuesto un algoritmo *de cumpleaños* que encuentra colisiones en SHA-1 con un esfuerzo menor a la *fuerza bruta* (orden de 2^{69} operaciones en lugar de 2^{80})

Funciones resumen
La familia SHA. Seguridad

- Se ha propuesto un algoritmo *de cumpleaños* que encuentra colisiones en SHA-1 con un esfuerzo menor a la *fuerza bruta* (orden de 2^{69} operaciones en lugar de 2^{80})
- Este algoritmo encuentra colisiones en SHA-0 con 2^{39} operaciones (!!)

Funciones resumen
La familia SHA. Seguridad

- Se ha propuesto un algoritmo *de cumpleaños* que encuentra colisiones en SHA-1 con un esfuerzo menor a la *fuerza bruta* (orden de 2^{69} operaciones en lugar de 2^{80})
- Este algoritmo encuentra colisiones en SHA-0 con 2^{39} operaciones (!!)
- Pese a esto, atacar SHA-1 es 4 ordenes de magnitud más dificil que atacar DES (...)

Funciones resumen
La familia SHA. Seguridad

- SHA-2 (publicada en 2001) contiene versiones de 224, 256, 384 y 512 bits. Última actualización en 2012 (FIPS PUB 180-4)

Funciones resumen
La familia SHA. Seguridad

- SHA-2 (publicada en 2001) contiene versiones de 224, 256, 384 y 512 bits. Última actualización en 2012 (FIPS PUB 180-4)
- Estado del algoritmo de 256 (8×32) o 512 bits (8×64) bits frente a 160 bits de SHA-1 (5×32)

Funciones resumen
La familia SHA. Seguridad

- SHA-2 (publicada en 2001) contiene versiones de 224, 256, 384 y 512 bits. Última actualización en 2012 (FIPS PUB 180-4)
- Estado del algoritmo de 256 (8×32) o 512 bits (8×64) bits frente a 160 bits de SHA-1 (5×32)
- No se conocen ataques demostrados.

Funciones resumen
La familia SHA. Seguridad

- SHA-2 (publicada en 2001) contiene versiones de 224, 256, 384 y 512 bits. Última actualización en 2012 (FIPS PUB 180-4)
- Estado del algoritmo de 256 (8×32) o 512 bits (8×64) bits frente a 160 bits de SHA-1 (5×32)
- No se conocen ataques demostrados.
- Pese a la ausencia de ataques, en 2006 se promueve un concurso para un nuevo estandar. Se publica el resultado en 2015 (SHA-3, FIPS 202 - 2014).

Funciones resumen
La familia SHA. Seguridad

- SHA-2 (publicada en 2001) contiene versiones de 224, 256, 384 y 512 bits. Última actualización en 2012 (FIPS PUB 180-4)
- Estado del algoritmo de 256 (8×32) o 512 bits (8×64) bits frente a 160 bits de SHA-1 (5×32)
- No se conocen ataques demostrados.
- Pese a la ausencia de ataques, en 2006 se promueve un concurso para un nuevo estandar. Se publica el resultado en 2015 (SHA-3, FIPS 202 - 2014).
- SHA-3 no es evolución de SHA-2. Considera un estado matricial de 1600 bits ($5 \times 5 \times 64$)

Keyed hash

Funciones resumen
Keyed hash

- Modificación de una proceso de resumen que considera una clave privada
- La propuesta de estandar (RFC-2104) considera MD5 (recoge la debilidad de este)
- El estandar admite el uso de distintas funciones resumen, obteniendo un HMAC (*hash message authentication code*)
- Un resultado similar puede obtenerse utilizando determinados algoritmos de cifrado de clave privada

Funciones resumen
Keyed hash

- Considera una función resumen con estado interno de B bytes y hash de t bits
- Si la clave k es mayor al resumen, entonces $k = h(k)$
- Si la clave es menor, entonces la clave se considera como los bytes de menor orden en una secuencia de t bits (el resto 0s)

Funciones resumen
Keyed hash: Implementación

$$HMAC(x) = h(\ outterkey \mid h(\ innerkey \mid x\ )\ )$$

donde:

$$\begin{cases} innerkey = k \oplus ipad \\ outterkey = k \oplus opad \end{cases} \qquad \begin{cases} ipad = (36_{HEX})^B \\ opad = (5C_{HEX})^B \end{cases}$$