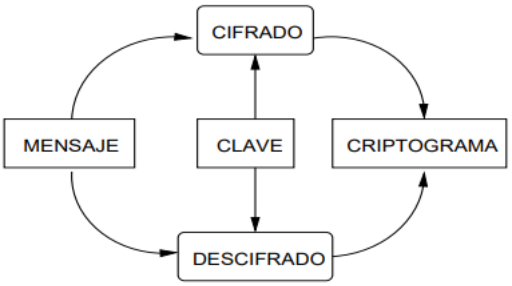
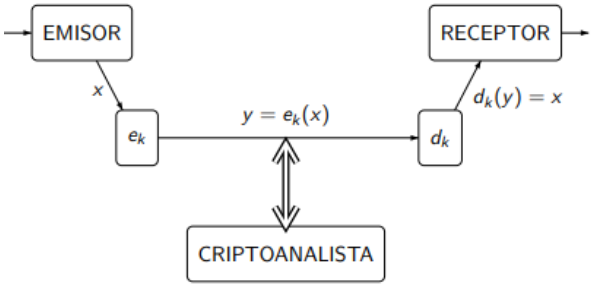
Criptografía

Tema 1





Tiene 5 objetivos:

- Confidencialidad: ocultar el contenido de la información salvo para aquellos autorizados.
- Accesibilidad: asegurar quien y en qué momento, puede acceder a una información.
- Autenticidad: el receptor de un mensaje debe poseer la certeza de su origen.
- Integridad: seguridad, para el receptor, de que el mensaje no ha sido modificado, así como posibilidad de detectar su posible manipulación.
- No repudio: imposibilidad por parte del emisor de negar la autoría de un mensaje.

Tiene 3 características básicas:

- Cifrado y descifrado eficiente independientemente de la clave escogida.
- Sistema fácilmente utilizable.
- La seguridad del sistema debe depender únicamente del secreto de las claves utilizadas y no del secreto de los algoritmos de cifrado y descifrado.

Hay 4 tipos de ataque:

- Texto cifrado.
- Mensaje conocido.
- Mensaje escogido.
- Criptograma escogido.

Aproximaciones:

- Uso de códigos
- Sistemas monoalfabéticos
- Sistemas polialfabéticos
- Sistemas poligráficos
- Cifrado por permutación
- Transformaciones variables en el tiempo
- Cifrado por bloques

Seguridad

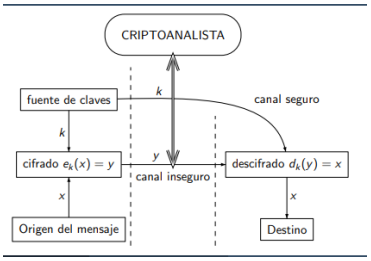
Incondicional: el sistema siempre es seguro

Computacional: el coste de obtener el mensaje es superior al valor de este o el tiempo necesario para obtener el mensaje supera la vida útil del mensaje.

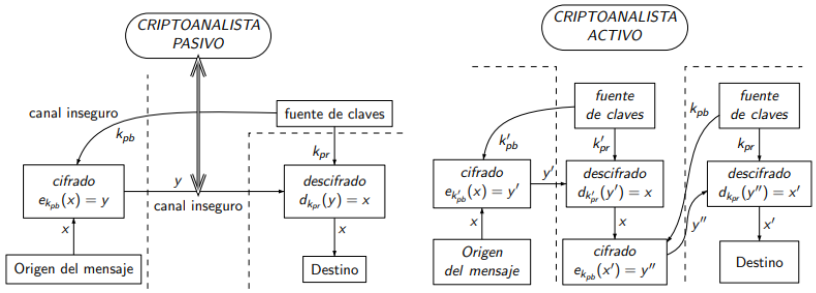
Protocolos

Envío de mensajes cifrados

- Clave simétrica:



- Clave asimétrica:



- Sistema híbrido: Pasar una clave privada mediante una clave pública y luego usar la clave privada para descifrar el resto de los mensajes.

Firma digital

- Clave simétrica: Emisor y receptor comparten clave de cifrado.
- Clave pública: se cifra con la clave privada y se comprueba la firma con la clave pública. No funciona bien para firmar textos largos, por eso se suele firmar el resumen del mensaje.

Voto electrónico

Propiedades:

- Democracia: Solo los inscritos en el censo pueden participar
- Privacidad: No puede relacionarse voto y elector.
- Seguridad: No se debe poder suplantar a un elector que decide no votar.
- Justicia: Nadie puede conocer el resultado antes de que finalice.
- Resistencia coercitiva: Ningún elector puede mostrar su voto
- Completitud: El resultado debe ser preciso
- Precisión: Un voto no puede ser alterado, los votos nulos no pueden contabilizarse de otro modo y el elector debe saber que su voto se ha considerado.
- Verificabilidad: Los electores pueden verificar que su voto ha sido considerado como se emitió.

- Se propone un sistema basado en doble autoridad: Mesa de Identificación (MI) y Mesa Electoral (ME).
- Estas se reparten la responsabilidad del registro, validación, depósito y escrutinio.
- Se asume que ambas autoridades son independientes, no teniendo relación excepto para la comunicación de claves.
- En todo momento se asumen canales de comunicación seguros. En cualquier caso puede considerarse un protocolo de clave pública para implementar dichos canales.

- La MI configura una clave pública (F_{MI}, V_{MI}) para ser utilizada como certificado, comunicando a la ME la parte pública (de verificación).
- Sea v la versión binaria del voto del elector. El elector genera un valor aleatorio h de la misma longitud de v. Sea $v' = v \oplus h$. El elector mantiene el valor de h secreto.
- El elector comunica a la MI el par $\langle v', id \rangle$
- La MI comprueba si el id del elector pertenece al censo. En este caso firma el voto del elector $RSA_{F_{MI}}(v')$ comunicando el resultado al elector.
- El elector puede comprobar en este momento la corrección del proceso, comunicando posteriormente a la urna el par $\langle RSA_{F_{MI}}(v'), h \rangle$.
- La urna verifica la firma del voto obteniendo v'. El voto se obtiene despues de computar $v = v' \oplus h$.

Tema 2

- Grupo $\langle G, \otimes \rangle$
  - $\otimes$ es de composición interna
  - $\otimes$ es asociativa
  - Existe un elemento neutro en G
  - Todo $a \in G$ tiene inverso respecto $\otimes$

$(\mathbb{Z}_n, +, \cdot)$ posea estructura de anillo conmutativo.

- Son operaciones cerradas y conmutativas.
- $(\mathbb{Z}_n, +)$ es un grupo (la operación es asociativa, tiene elemento neutro e inverso para todo valor en $\mathbb{Z}_n$).
- $(\mathbb{Z}_n, .)$ es un semigrupo (la operación es asociativa y tiene elemento neutro).
- El producto distribuye respecto la suma.

- Teorema (de congruencia lineal): $ax \equiv b \pmod{n}$ tiene una única solución sii $mcd(a, n)$ divide b.

- Teorema: $mcd(a, b)$ es el menor entero estrictamente positivo del conjunto $\{xa + yb \ : \ x, y \in \mathbb{Z}\}$ (combinaciones lineales de a y b)
- Corolario: $d|a$ y $d|b$ implica que $d|mcd(a, b)$

- a es invertible si $mcd(a, m) = 1$.
- Si $mcd(a, m) = 1$ entonces a y m son relativamente primos

- Teorema: $mcd(a, b) = mcd(b, a \bmod b)$

$a \in \mathbb{Z}_n^*$ es *resíduo cuadrático módulo n* si existe un $x \in \mathbb{Z}_n^*$ tal que $x^2 \equiv a \pmod{n}$. En caso de existir, x se comoce como *rai (de a) módulo n.*

Para solucionar factorizar el número

Tema 3

Sistemas monoalfabéticos

Polibio

	1	2	3	4	5
1	A	B	C	D	E
2	F	G	H	I	J
3	L	M	N	Ñ	O
4	P	Q	R	S	T
5	U	V	X	Y	Z

CAPTURADOAGENTEENDESTINO
⇓
13114145514311143511221533451533...

Caesar:

Se cifra el mensaje desplazando los caracteres del mensaje

e(x) = x + k mod 27 — e: encriptación

d(y) = y - k mod 27 — d: desencriptación

Espacio de claves: desplazamientos posibles.
Número de claves: talla del alfabeto.

Sustitución simple:

Se sustituye una letra por otra

A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O
O	C	T	A	V	I	P	Z	B	D	E	F	G	H	J	K

Espacio de claves: Permutaciones del alfabeto.
Número de claves: (Talla del alfabeto)!

Afín:

$e(x) = ax + b \bmod 27$

$d(y) = a^{-1} (y - b) \bmod 27$

Sistemas polialfabeticos

Vigenèrè

$e_k(x_i) = x_i + k_{i \bmod m} \bmod 27$

$d_k(y_i) = y_i - k_{i \bmod m} \bmod 27$

clave: clave, en la parte de abajo salen las posiciones de las letras

x:	T	E	X	T	O	A	C	I	F	R	A	R
k:	c	l	a	v	e	c	l	a	v	e	c	l

x:	20	4	24	20	15	0	2	8	5	18	0	18
k:	2	11	0	22	4	2	11	0	22	4	2	11
y:	22	15	24	15	19	2	13	8	0	22	2	2
y:	V	O	X	O	S	C	N	I	A	V	C	M

Criptoanálisis

Kasiski: un fragmento de mensaje que aparezca k veces será cifrado de k/n formas, donde n es el número de alfabetos. Dado ese fragmento las distancias entre las posiciones de ese fragmento será MCD del tamaño de la clave.

Índice de coincidencia: analiza la variación de la frecuencia relativa de cada letra respecto a una distribución uniforme, en español su valor es 0,072.

Códigos

Se cifran las palabras mediante un código.

Ataque: se busca identificar sílabas, los números suelen ser patrones.

Sistemas por transposición

O	A	N	C	D	S	I	C	N	I	E	B
P	C	C	E	A	T	R	U	T	O	N	L
E	I	A	L	D	R	D	M	A	N	S	E
R	O	N	A	E	U	O	E	C	S	I	X

Leer por columnas

Sistemas poligráficos

Playfair

- Cifra el mensaje considerando pares de dos símbolos (s_1, s_2)
- La clave es una matriz donde se disponen los símbolos del alfabeto (5×5)

Algoritmo: sean (x_1, y_1) y (x_2, y_2) las coordenadas de s_1 y s_2 en la matriz clave.

- Si $x_1 = x_2 \Rightarrow ((x_1, y_1 + 1 \text{ mód } 5), (x_2, y_2 + 1 \text{ mód } 5))$
- Si $y_1 = y_2 \Rightarrow ((x_1 + 1 \text{ mód } 5, y_1), (x_2 + 1 \text{ mód } 5, y_2))$
- Si $x_1 \neq x_2 \wedge y_1 \neq y2 \Rightarrow ((x_1, y_2), (x_2, y_1))$
- Si $s_1 = s_2$ insertar un símbolo sin significado
- Si al final queda un único caracter sin cifrar, insertar un símbolo sin significado

- Objetivo reconstrucción de la matriz clave
- Proceso iterativo que considera hipótesis iniciales basadas en el criptograma: repeticiones en el criptograma; bigramas XC-CX; ...
- Los bigramas con letras en común permiten ubicar más caracteres en la matriz
- La reconstrucción ha de considerar las tres posibles situaciones de los caracteres del bigrama
- La matriz resultante puede ser mayor de 5×5, pudiendo compactarse posteriormente
- El criptoanálisis Playfair es un caso particular del criptoanálisis del sistema *four squares*

Hill

- Cifra el mensaje considerando bloques de k símbolos
- La clave es una matriz $K_{k\times k}$ de valores en Z_m tal que $mcd(|K|, 27) = 1$

$$K = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1k} \\ a_{21} & a_{22} & \dots & a_{2k} \\ \dots & & & \\ a_{k1} & a_{k2} & \dots & a_{kk} \end{pmatrix};$$

Algoritmo: considerando el mensaje $x = x_1x_2x_3x_4\dots$

$$K \cdot \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_k \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_k \end{pmatrix}; \qquad K^{-1} \cdot \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_k \end{pmatrix} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_k \end{pmatrix}$$

- Ataque basado en *mensaje conocido*
- Sea $x = x_1x_2x_3\dots$ un fragmento del mensaje e $y = y_1y_2y_3\dots$ su correspondiente criptograma
- Suponiendo la clave de tamaño 3, es posible construir el sistema:

$$K \cdot \begin{pmatrix} x_1 & x_4 & x_7 \\ x_2 & x_5 & x_8 \\ x_3 & x_6 & x_9 \end{pmatrix} = \begin{pmatrix} y_1 & y_4 & y_7 \\ y_2 & y_5 & y_8 \\ y_3 & y_6 & y_9 \end{pmatrix}$$

- Es posible calcular la clave si la matriz *mensaje* es invertible

Tema 4

Cifrado en flujo

Los símbolos individuales se cifran mediante una transformación que varía en el tiempo, se implementan mediante hardware, basados en generadores de clave pseudoaleatorias.

Algoritmo:

- Obtener una codificación binaria del mensaje y la clave
- El mensaje cifrado aparece cuando se opera un "O exclusivo" bit a bit sobre el mensaje y la clave
- El mismo proceso sirve como descifrado del mensaje

Propiedades:

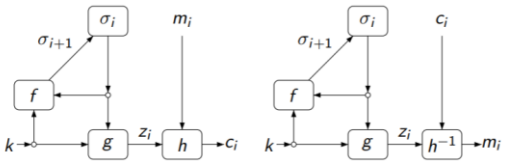
- Incondicionalmente seguro si:
  - Clave compuesta por una secuencia binaria de la misma longitud que el mensaje
  - Clave única para cada mensaje

Sistemas

Flujo síncrono

- Flujo de claves generado independientemente del mensaje y del criptograma Función de cambio de estado:

  $\sigma_{i+1} = f(\sigma_i, k)$
  Función de generación de clave: $z_i = g(\sigma_i, k)$
  Función de cifrado: $c_i = h(z_i, m_i)$



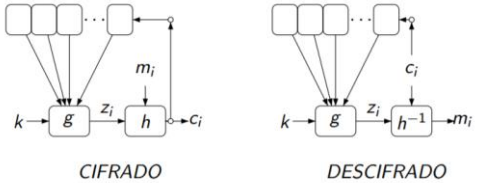
Necesita sincronización entre emisor y receptor, usa marcas a intervalos regulares para reiniciar la sincronización, gracias a eso no es sensible a errores de transmisión.

Es sensible a ataque activos, inserción, borrado, repetición de símbolos

Flujo autosíncrono

- Flujo de claves generado como una función de la clave de cifrado y un número prefijado de los últimos símbolos del criptograma

  Estado del sistema: $\sigma_i = (c_{i-t}, c_{i-t+1}, c_{i-t+2}, \dots, c_{i-1})$
  Función de generación de clave: $z_i = g(\sigma_i, k)$
  Función de cifrado: $c_i = h(z_i, m_i)$



Depende de los últimos símbolos del criptograma, la pérdida de sincronización se resuelve sola, Los errores se propagan de forma limitada, es menos sensible a los ataques activos y a los basados en redundancias (ya que se desvirtúan las propiedades estadísticas).

Secuencias binarias pseudoaleatorias

Ejemplo:

010110010001111

- *periodo* 15
- *Rachas*: 4 de longitud 1; 2 de longitud 2; una de longi 3; 1 de longitud 4
- *Función de autocorrelación*: $AC^{15}(4) = \frac{7-8}{15}$

Periodo: número de caracteres hasta que se repita una cadena

Racha: secuencias de bits igua.

Función de autocorrelación: medida de similitud entre una secuencia de periodo T y la secuencia resultante de desplazar esa secuencia d posiciones.

$AC_T(k)$ = (Coincidencias – fallos) / Periodo

Postulados de Golomb:

P1: En un periodo, la diferencia entre el número de bits distintos (0s y 1s) ha de ser a lo sumo 1

P2: Denotando el total de rachas con r, el número de rachas de longitud l en el periodo debe ser igual o superior a $\frac{r}{2^l}$

P3: Para cualquier valor de k no múltiplo de T (fuera de fase) el valor de $AC_T(k)$ es constante.

(Nótese que en caso que k sea múltiplo de T, $AC_T(k) = 1$)

Las secuencias que cumplen los Postulados de Golomb presenta una distribución uniforme, denomiandose *pseudo-noise* (PN)

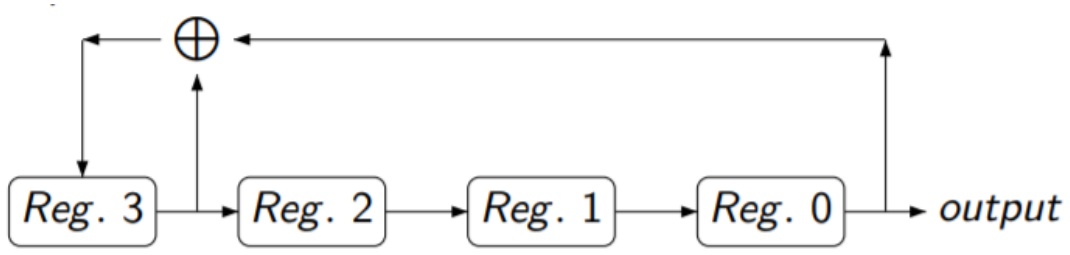
Generadores pseudoaleatorios

Generadores de congruencia lineal

$X_{i+1} = ax_i + b \pmod n$ a, b, n son la clave secreta

x_0 es la semilla, se usa el valor anterior de la secuencia en el resto de valores.
Se pueden obtener los parámetros del generador teniendo en cuenta un fragmento de la secuencia generada, no se considera de utilidad criptográficas.

LFSR



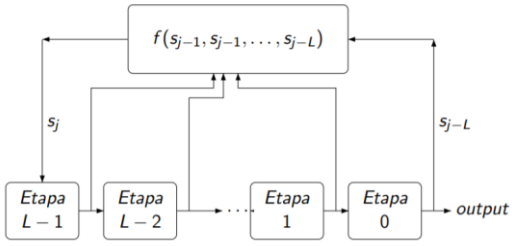
Cadenas de gran periodo, buenas propiedades estadísticas, fácil implementación hardware, se caracteriza por un polinomio módulo 2.

Tipos:

- Factorizables: pueden ser descompuestos, el T depende del estado inicial, varía entre n>= T >= 2^n-1, puede haber periodos de longitud divisor del principal.
- Irreducibles: T es un factor de 2^n-1.
- Primitivos: el polinomio es irreducible y genera el conjunto de configuraciones, T = 2^n-1, hay 2^n-1 polinomios de este tipo.

n es el número de etapas, se pueden romper con un segmento de 2n valores de la secuencia generada.

FSR



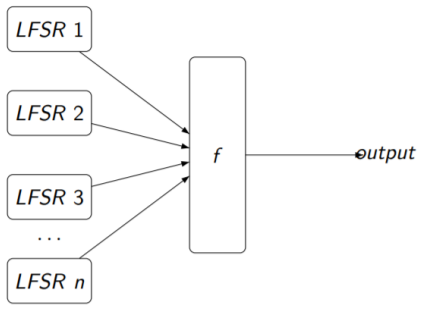
Propiedades:

- Basados en funciones booleanas
  - *función booleana*: función con n entradas y una salida, todas ellas binarias
  - Existen 2^{2^n} funciones booleanas distintas de n variables
  - Pueden tener pequeños ciclos que se repiten indefinidamente
  - Son más lentos que los sistemas lineales

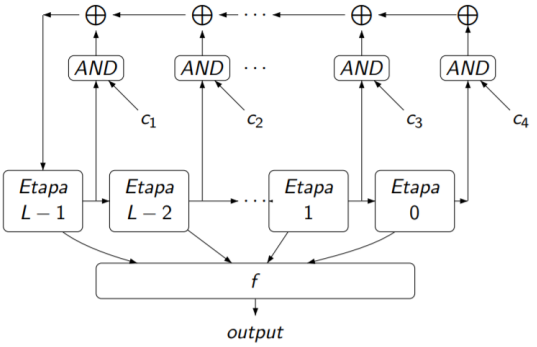
Combinaciones LFSR

Las secuencias LFSR son predecibles, en la realidad se plantean 3 aproximaciones

Función de salida booleana

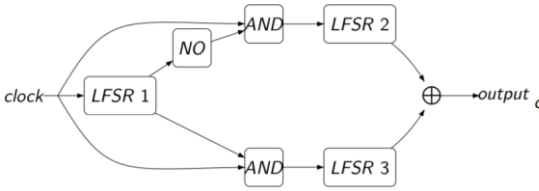


Nonlinear filter generators



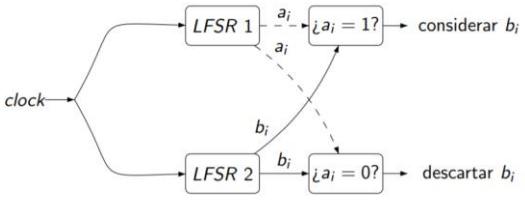
Clock controlled generators

Alternating step generator

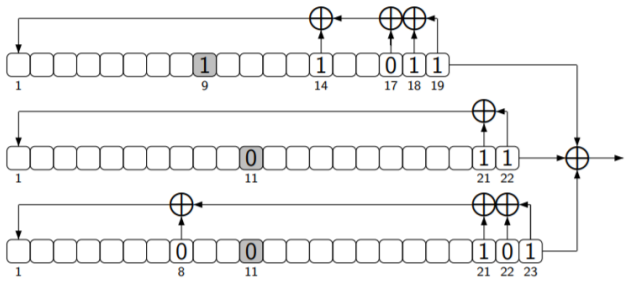


Clock controlled generators

Shrinking generator

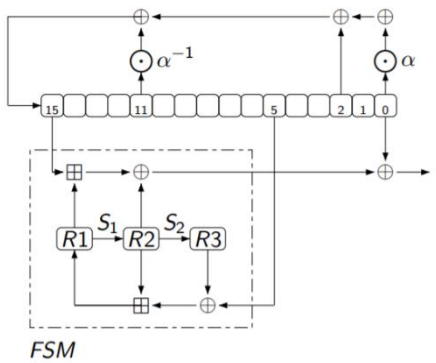


Cifrado A5



Los grises son para mirara que LFS se tienen que mover, aquellos que tengan el dígito mayoritario

Cifrado 4G



La FSM modifica dos de los 3 registros usando S-boxes

Cifrados basados en generadores pseudoaleatorios

Una secuencia aleatoria hace de clave para combinar con el mensaje mediante una función.

Salsa20/x: Implementación

$c_0 = 61707865_{HEX}$ $c_1 = 3320646E_{HEX}$
$c_2 = 79622D32_{HEX}$ $c_3 = 6B206574_{HEX}$ //Constantes
$k_0, k_1, \ldots, k_7 \in \{0,1\}^{32}$ //Clave
$j_0, j_1 \in \{0,1\}^{32}$ //Contador de posición
$n_0, n_1 \in \{0,1\}^{32}$ //Nonce

$$\omega = \begin{pmatrix} x_0 & x_1 & x_2 & x_3 \\ x_4 & x_5 & x_6 & x_7 \\ x_8 & x_9 & x_{10} & x_{11} \\ x_{12} & x_{13} & x_{14} & x_{15} \end{pmatrix} \Longleftarrow \begin{pmatrix} c_0 & k_0 & k_1 & k_2 \\ k_3 & c_1 & n_0 & n_1 \\ j_0 & j_1 & c_2 & k_4 \\ k_5 & k_6 & k_7 & c_3 \end{pmatrix}$$

Salsa20/20: Implementación

Permutación $\prod$ de Salsa 20

Require: $x_1 x_2 x_3 \ldots x_{15} \in \{0,1\}^{512}$ $//x_i \in \{0,1\}^{32}$
Require: Salsa20 $QRound(a, b, c, d, t)$
1: **Método**
2: **for** $i = 1$ **to** 10 **do**
3: $QRound(x_0, x_4, x_8, x_{12}, 1)$; $QRound(x_1, x_5, x_9, x_{13}, 2)$;
4: $QRound(x_2, x_6, x_{10}, x_{14}, 3)$; $QRound(x_3, x_7, x_{11}, x_{15}, 4)$;
5: $QRound(x_0, x_1, x_2, x_3, 1)$; $QRound(x_4, x_5, x_6, x_7, 2)$;
6: $QRound(x_8, x_9, x_{10}, x_{11}, 3)$; $QRound(x_{12}, x_{13}, x_{14}, x_{15}, 4)$;
7: **end for**
8: **FinMétodo.**

Salsa20/x: Implementación

Salsa 20 QuarterRound

Require: $a, b, c, d \in \{0,1\}^{32}$
Require: $t \in \{1, 2, 3, 4\}$
Método
Ejecuta las operaciones de forma circular empezando por la t-esima.
(i) $b \oplus = (a \boxplus d) <<< 7$;
(ii) $c \oplus = (b \boxplus a) <<< 9$;
(iii) $d \oplus = (c \boxplus b) <<< 13$;
(iv) $a \oplus = (d \boxplus c) <<< 18$;
FinMétodo.

Cada bloque viene de un hash de la clave (256 b) un nonce (único mensaje de 64 b) y un contador de posiciones (64 b)
La función resumen se basa en una permutación fija de 512b. Se sugiere Salsa20/12

ChaCha20/x: Implementación

$c_0 = 61707865_{HEX}$ $c_1 = 3320646E_{HEX}$
$c_2 = 79622D32_{HEX}$ $c_3 = 6B206574_{HEX}$ //Constantes
$k_0, k_1, \ldots, k_7 \in \{0,1\}^{32}$ //Clave
$j_0 \in \{0,1\}^{32}$ //Contador de posición
$n_0, n_1, n_2 \in \{0,1\}^{32}$ //Nonce

$$\omega = \begin{pmatrix} x_0 & x_1 & x_2 & x_3 \\ x_4 & x_5 & x_6 & x_7 \\ x_8 & x_9 & x_{10} & x_{11} \\ x_{12} & x_{13} & x_{14} & x_{15} \end{pmatrix} \Longleftarrow \begin{pmatrix} c_0 & c_1 & c_2 & c_3 \\ k_0 & k_1 & k_2 & k_3 \\ k_4 & k_5 & k_6 & k_7 \\ j_0 & n_0 & n_1 & n_2 \end{pmatrix}$$

ChaCha20/20: Implementación

Permutación $\prod$ de ChaCha 20

Require: $x_1 x_2 x_3 \ldots x_{15} \in \{0,1\}^{512}$ $//x_i \in \{0,1\}^{32}$
Require: ChaCha20 $QRound(a, b, c, d)$
1: **Método**
2: **for** $i = 1$ **to** 10 **do**
3: $QRound(x_0, x_4, x_8, x_{12})$; $QRound(x_1, x_5, x_9, x_{13})$;
4: $QRound(x_2, x_6, x_{10}, x_{14})$; $QRound(x_3, x_7, x_{11}, x_{15})$;
5: $QRound(x_0, x_5, x_{10}, x_{15})$; $QRound(x_1, x_6, x_{11}, x_{12})$;
6: $QRound(x_2, x_7, x_8, x_{13})$; $QRound(x_3, x_4, x_9, x_{14})$;
7: **end for**
8: **FinMétodo.**

ChaCha20/x: Implementación

ChaCha20 QuarterRound

Require: $a, b, c, d \in \{0,1\}^{32}$
1: **Método**
2: $a \boxplus = b$; $d \oplus = a$; $d <<< 16$;
3: $c \boxplus = d$; $b \oplus = c$; $b <<< 12$;
4: $a \boxplus = b$; $d \oplus = a$; $d <<< 8$;
5: $c \boxplus = d$; $b \oplus = c$; $b <<< 7$;
6: **FinMétodo.**

Mantiene arquitectura de Salsa20
Cada bloque viene de un hash de la clave (256 b), un nonce (mensaje único de 96 b) y un contador de 32 b

Tema 5

Hashing

Propiedades deseables de las funciones hash:

- Funciones hash unidireccionales (OWHF):
  - Resistentes 1º preimagen (one way hashing): dado un hash y no es computacionalmente posible encontrar una x | h(x) = y.
  - Resistentes a la 2º preimagen (weak colisión resistance): dado un mensaje x no se puede conseguir un mensaje x' | h(x) = h(x').
- Funciones hash resistentes a colisiones (CRHF):
  - Resistencia a las colisiones (strong colisión resistance): encontrar dos mensajes, x, x' |h(x)=h(x') es intratable.

Relación entre propiedades:

- Resistencia a la 1º preimagen no implica resistencia a la 2º preimagen ni a la inversa.
- CRHF implica resistencia a la 2º preimagen, pero no a la primera.

Seguridad

Ataque de 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:

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

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

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

El ataque acota inferiormente la talla del resumen, con 40 bitss se obtiene una colisión con probabilidad 0.5 con 2^{20} mensajes al azar.
Para OWHF se necesita exploración exhaustiva. Una CRHF de n bits es sensible al algoritmo de Yuval.

Algoritmo de Yuval:

Ensure = garantizas

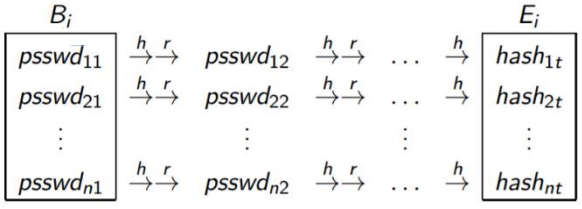
Firma de un mensaje ilegítimo con la firma de un mensaje inofensivo

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')

Ataque arcoiris

Diseñado para encontrar colisiones en contraseñas, sensible a la función de resumen utilizada, usa una aproximación time-memory trade-off, TMTO.

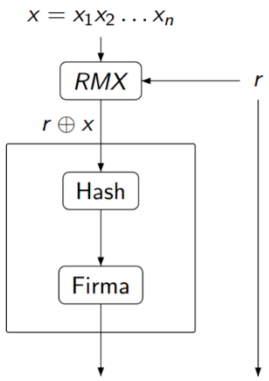
Require: Una función resumen h
Require: Una función recodificante r
Require: t longitud de la secuencia
Require: n número de entradas
Ensure: Un vector rainbow para la función h.
Método
while La tabla no contenga n entradas **do**
Escoger P_i un password al azar.
$P = P_i$
for $j = 1$ **to** $t - 1$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



El inconveniente es obtener colisiones para la tabla, mediante salting se puede prevenir el uso de las técnicas.

Randomized Hashing RMX

Complementa el proceso de firma contra ataques por colisión de las funciones resumen



r es una secuencia binaria aleatoria que se genera para cada mensaje firmado.

La secuencia modifica el mensaje, mediante XOR, antes del cálculo del resumen y su firma.

El ataque es equivalente a encontrar una 1º preimagen.

Implementación funciones hash

Se computa el resumen considerando un *estado* que se modifica iterativamente

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
$//x = x_1x_2 \ldots x_m$
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: $h_i = f(h_{i-1}, x_i)$
7: **end for**
8: **return** h_i

Familia MDx

MD4: Preproceso del mensaje

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*

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** $ABCD$

MD5 mismo esquema con un 4º round.

DNI electrónico utiliza SHA-1/RSA.
No se recomienda firmar hashes MD5

Familia SHA

Mismo preproceso que 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.

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: $AA = A$ $BB = B$ $CC = C$ $DD = D$ $EE = E$
5: **Round 1**
6: **Round 2**
7: **Round 3**
8: **Round 4** //20 operaciones cada bloque
9: $A = A + AA$ $B = B + BB$ $C = C + CC$
$D = D + DD$ $E = E + EE$
10: **end for**
11: **return** $ABCDE$

Keyed hash

Modificación del proceso que considera una clave privada

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)

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

donde:

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

Tema 7

Seguridad incondicional

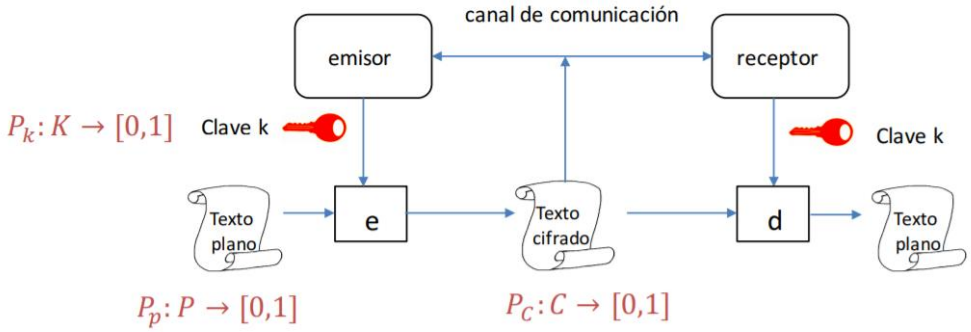
Asunción: los textos X y las claves K son variables independientes.

p(x,k) = p(x)p(k)

Prob condicional y conjunta: p(x,y) = p(x|y)p(y)

p(x,y) = p(y|x)p(x)

Regla de Bayes p(x|y) = p(y|x)p(x)/p(y)



- Probabilidad de aparición en el canal de comunicación del texto cifrado *y* dado el texto sin cifrar *x* : $P_C(y|x) = \sum_{\{k\,:\,x=d_k(y)\}} p_k(k)$

- Probabilidad de que un texto sin cifrar sea *x* dado que el texto cifrado sea *y* :

$$P_C(x|y) = \frac{p_p(x)p_c(y|x)}{p_c(y)} = \frac{p_p(x)\sum_{\{k:\ x=d_k(y)\}} p_x(k)}{\sum_{\{k:y\in C(k)\}} p_k(k)p_p(d_k(y))}$$

Un sistema de cifrado tiene privacidad perfecta (secreto perfecto, seguridad incondicional) si se cumple que

$$(\forall x \in P)\ \ (\forall\ y\ \in C)\ \ P_p(x|y) = P_p(x)$$

Hacia la seguridad incondicional: entropía

Sean X e Y dos variables aleatorias discretas con distribuciones de probabilidad condicional y conjunta. Definimos el valor medio ponderado de la cantidad de información (entropía)

$$H(X) = -\sum_{i=1}^{n} p(X = x_i) \log_2 p(X = x_i)$$

$$H(X|y) = -\sum_{x} p(x|y) \log_2 p(x|y)$$

$$H(X|Y) = -\sum_{y}\sum_{x} p(y)p(x|y) \log_2 p(x|y)$$

Algunos resultados sobre la entropía conjunta o condicional

Teorema. H(X,Y) = H(Y) + H(X | Y) = H(X) + H(Y | X)

Corolario. H(X | Y) ≤ H(X)
H(X | Y) = H(X) sii X e Y son independientes

Hacia la seguridad incondicional: entropía

Claves falsas y distancia de unicidad (I)

- Equivocación de clave $$H(K|C) = -\sum_{y c \in C}\sum_{k \in K} p(k,c) \log_2 p(k|c)$$
- Entropía de un lenguaje *L* $$H_L = \lim_{n\to\infty} \frac{H(P^n)}{n}$$
- Redundancia de un lenguaje *L* $$R_L = 1 - \frac{H_L}{\log_2 |P|}$$
- Claves consistentes con un texto cifrado *y*

$$K(y) = \{k \in K : \exists\, x \in P^n, p_{P^n}(x) > 0, e_k(x) = y\}$$

Claves falsas y distancia de unicidad (II)

- Número de claves falsas en promedio $$\overline{s_n} = \sum_{y \in C^n} p(y)\,|K(y)| - 1$$

Si |C|=|P| y $p_k(k) = \frac{1}{|K|}$ *entonces* $\overline{s_n} \geq \frac{|K|}{|P|^{nR_L}} - 1$

- Distancia de unicidad n_0 $$n_0 \approx \frac{\log_2 |K|}{R_L \log_2 |P|}$$

Algoritmos no deterministas

- Sea P un problema con un conjunto de restricciones R
- Sea S una estructura de soluciones para el problema P

Un algoritmo no determinista que resuelve P se basa en dos fases

(1) Fase de conjetura (aleatoria, no determinista)

conjetura(P,S)

(2) Fase de verificación (determinista)

verifica(S,R)

Medida de complejidad (tiempo/espacio)

Reducción polinómica entre problemas (la reducción de Karp)

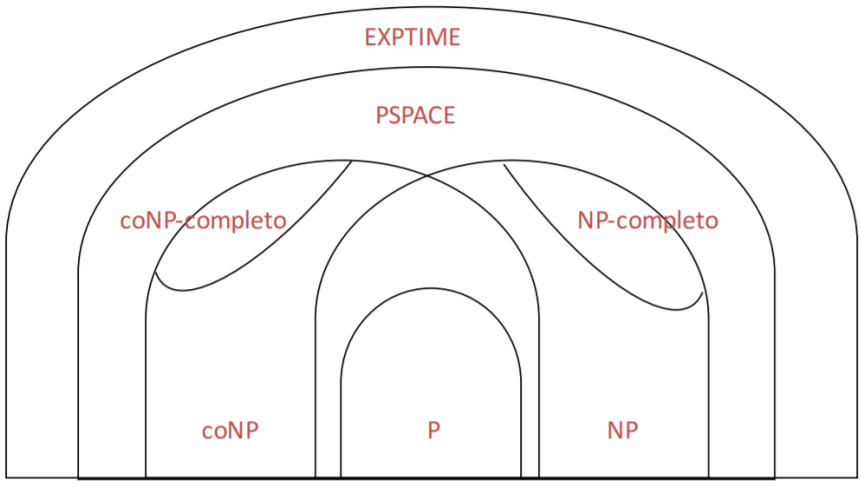
Dados los problemas A y b, diremos que A se reduce a B ($A \leq_p B$) si existe una función f tal que

1. f es calculable en tiempo polinómico
2. La instancia x del problema A tiene solución sii la instancia f(x) del problema B tiene solución

Dado un conjunto de problemas C, diremos que un problema A es ...

- C-duro si todos los problemas de C se pueden reducir a A
- C-completo si A ∈ C y A es C-duro
- coC si el problema complementario de A pertenece a C

Una clasificación de los problemas bajo la Teoría de la Complejidad



Asumimos que P ≠ NP

Asumiremos que un problema es computacionalmente difícil de resolver si es NP-completo

- El problema de la satisfacibilidad

Sea $U = \{ u_1, u_2, ..., u_n \}$ un conjunto de variables booleanas y sea $C = \{ c_1, c_2, ..., c_m \}$ un conjunto de cláusulas (disyunción de literales inducidos por las variables). Se pide establecer si existe una asignación de valores a las variables de U que hagan todas las cláusulas de C simultáneamente ciertas.

$$U = \{u_1, u_2, u_3, u_4\}$$

$$C = \{ \{u_1, \overline{u_2}, \overline{u_3}\}, \{u_2, u_3\}, \{\overline{u_4}\}, \{\overline{u_1}, u_3, u_4\}\}$$

- El problema de la isomorfía de subgrafos

Dados dos grafos G_1 y G_2 conexos y no dirigidos, se pide establecer si existe un subgrafo de G_1 isomorfo a G_2

Asumiremos que un problema es computacionalmente difícil de resolver si es NP-completo

- El problema de las congruencias simultáneas

Sea $C = \{ (a_1,b_1), (a_2,b_2), \dots , (a_n,b_n) \}$ donde a_i , b_i son enteros positivos y se cumple que $a_i \leq b_i$ para cualquier valor de i

"Dado el conjunto C. ¿ Existe un valor entero x tal que $x \equiv a_i \pmod{b_i}$ para $1 \leq i \leq n$?"

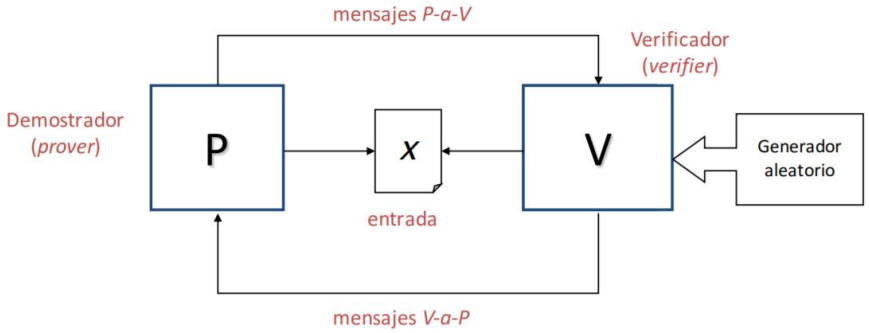
- El problema de las congruencias cuadráticas

Sean a , b y c enteros positivos

"Dados a, b y c. ¿ Existe un valor entero positivo $x < c$ tal que $x^2 \equiv a \pmod{b}$?"

El problema se vuelve fácil de resolver si el entero *b* es primo (asumiendo algunas condiciones de partida)

- Todos los problemas NP-completos contienen instancias fáciles de resolver (mediante algoritmos polinómicos deterministas)
- Toda instancia fácil de resolver se puede enmascarar como una instancia difícil
- El problema de desenmascarar la instancia se resuelve a partir de una información secreta (clave)



- La computación se establece por turnos (*rounds*) que comienza *V*
- La aceptación o rechazo de la entrada la establece *V*
- El número de mensajes y la longitud de los mismos están acotados polinómicamente en relación con la longitud de la entrada
- Un Sistema Interactivo decimos que presenta conocimiento nulo si la información que envía P a V puede ser simulada aleatoriamente (P no desvela ninguna información acerca de x)
- La demostración de x permite a P autenticarse ante V