Criptografía Simétrica (II)

M.I. GONZÁLEZ VASCO / GRADO EN INGENIERÍA DE LA CIBERSEGURIDAD

UNIVERSIDAD REY JUAN CARLOS

Qué vamos a aprender

1. Primitivas útiles: HASH/MACs
2. Cifradores históricos
3. Cifrado en Flujo
4. Cifrado en Bloque

Referencia: Capítulo 3, 5, 7 y 8 del Smart

Esquema de cifrado de clave privada.

Consta de 3 algoritmos **(Gen, Enc, Dec).**

- **Gen**: generación de claves, pptm, recibe como entrada 1^n y genera la clave secreta k.
- **Enc:** algoritmo de cifrado, pptm, recibe como entrada una clave k y un texto claro y da como salida un cifrado c . (obs: usaremos $Enc(k,m)$ o $Enc_k(m)$ alternativamente para denotar su salida)
- **Dec:** algoritmo de descifrado, recibe como entrada la clave k y un cifrado c,

  su salida es m o $\perp$.

Corrección: m= **Dec**(k, **Enc**(k, m))

2. Cifradores históricos

Smart, capítulo 3

- Cifrado por trasposición (shift) – César (26 claves)
- Cifrado por sustitución (substitution) – (26! claves)
- Cifrado Vigenere (evitando el análisis de frecuencias)
- Cifrados por permutación (inicios del cifrado en bloque…)

3. Cifrado en flujo

(STREAM CIPHERS)

Pieza base: Generadores pseudoaleatorios.

Informalmente, son funciones que tienen como entrada una semilla de longitud corta y devuelven una secuencia mucho más larga.

Su distribución de salida ha de ser difícil de distinguir de una distribución aleatoria.

Pieza base: Generadores pseudoaleatorios.

Γ es un algoritmo determinista que corre en tiempo polinomial.

Para cada input $s \in \{0,1\}^n$ se devuelve una secuencia de bits $\Gamma(s)$ de longitud $l(n)$.

La función l recibe el nombre de **factor de expansión**.

Generadores pseudoaleatorios.

Tiene que cumplir:

1. **Expansión**: l(n)>n
2. **Pseudoaleatorio**: para cualquier pptm D (distinguidor) se cumple:

$$|Pr[D(r)=1]-Pr[D(\Gamma(s))=1]| = negl(n)$$

con $|s|=n$, $|r|=l(n)$, siendo r un valor elegido con una distribución uniforme (real).

Función despreciable (negligible).

Decimos que una función $f : N \longrightarrow R$ es **despreciable** si para cada polinomio p(.) existe n_0 tal que si $n>n_0$ entonces

$$f(n)<1/p(n)$$

¿Cómo usarlos para cifrar?

Tomamos Γ generador pseudoaletorio con factor de expansión l y construímos un esquema de cifrado como sigue:

- $Gen(1^n)$ elige $k \in \{0,1\}^n$
- Dado $m \in \{0,1\}^{l(n)}$

$$Enc_k(m) := \Gamma(k) \oplus m$$

- Dado $c \in \{0,1\}^{l(n)}$

$$Dec_k(c) := \Gamma(k) \oplus c$$

Cifrado en flujo

La construcción anterior suele recibir el nombre de cifrado en flujo (stream cipher).

Teorema: si Γ es un generador pseudoaleatorio, entonces el cifrado en flujo es seguro en un cierto sentido (IND-EAV seguro).

Múltiples cifrados

- El cifrado en flujo no se mantiene seguro cuando se usa varias veces con la misma clave
- Problema: es determinista. Se detecta, por ejemplo, cuando un mensaje se envía dos veces.
- Solución: la clave nunca se reutiliza; el esquema mantiene una secuencia de clave que se va generando y actualizando para poder cifrar secuencialmente.

4. Cifrado en bloque

Funciones pseudoaleatorias

- En realidad consiste en una familia de funciones indexadas por una clave secreta.

$$F : \{0,1\}^l \times \{0,1\}^n \longrightarrow \{0,1\}^n$$

$$(k,x) \longrightarrow F_k(x)$$

- Se pide que al elegir una clave al azar la función correspondiente sea difícil de distinguir de una función elegida completamente al azar.
- También se pide que cada función se pueda computar eficientemente.

Permutaciones pseudoaleatorias

- Funciones pseudoaleatorias a las que se les pide que cada $F_k(.)$ sea una permutación de $\{0,1\}^n$
- Se pide que, para cada clave, tanto la permutación como la inversa sean eficientemente computables.

Cifrador en bloque

Es el nombre que reciben las permutaciones pseudoaletorias cuando se usan en la práctica. Parámetros que los definen:

- l : longitud de clave.
- n : longitud de bloque.
- Se usan como primitivas para la construcción de esquemas de cifrado de clave secreta.

Modos de operación

- Determina la forma de utilizar un cifrador en bloque.
- Las partes que se comunican comparten una clave secreta k y usan la permutación F_k
- Se supone que se tiene un mensaje cuya longitud es múltiplo de la longitud de bloque n (si no es así, se completa con información supeflua).
- Se divide el mesaje en bloques de longitud n

$$m=m_0 \,||\, m_1 \,||\, \ldots \,||\, m_s$$

Modos de operación

1. **Electronic Code Book** (ECB)

   $c_j := F_k(m_j)$

   Obs: es inseguro

2. **Cipher Block Chaining** (CBC)

   Se elige al azar un IV de n bits

   $c_0 := IV$

   $c_j := F_k(m_j \oplus c_{j-1})$

Modos de operación

3. **Output feedback** (OFB)

   Se elige al azar un IV de longitud n

   $r_0 := IV$

   $r_j := F_k(r_{j-1})$

   $c_0 := IV$

   $c_j := m_j \oplus r_j$

Modos de operación

4. **Counter** (CTR)

Se elige al azar un IV de longitud n

ctr:=IV , c_0:=IV

Para j=1 hasta n hacemos;

r_j:=F_k(ctr+j)

c_j:= $m_j \oplus r_j$

Seguridad de los modos

Teorema: los modos CBC, OFB y CTR son seguros si se emplea una permutación pseudoaleatoria (seguridad IND-CPA)

Obs: OFB y CTR sólo necesitan, en realidad, una función pseudoaleatoria.

Modos avanzados (AE)

- Existen modos de operación especiales que añaden a la funcionalidad de confidencialidad la de autenticación, se llaman modos de cifrado autenticado (Authenticated Encryption)
- Algunos ejemplos:
  - GCM
  - OCB
  - EAX
  - ChaCha20 + Poly1305

4.2. Construcciones. Ejemplos: DES y AES.

Técnicas generales de diseño para cifradores en bloque

Paradigma confusión-difusión.

Confusión: Supongamos que la longitud de bloque es 128.

- La clave determina 16 permutaciones f_1, f_2, …, f_{16} cuya longitud de bloque es 8.
- Se divide $x=x_1||x_2||\ldots||x_{16}$ en bloques de 8 bits.
- Se define $F_k(x):=f_1(x_1)||\ldots||f_{16}(x_{16})$

Técnicas generales de diseño

Difusión: los bits del output anterior se mezclan de acuerdo a una cierta permutación (permutaciones de mezcla).

Los pasos confusión-difusión se repiten un cierto número de **rondas**.

Implementación

Red de sustitución permutación:

- Las permutaciones $\{f_j\}$ son fijas y no dependen de la clave. Reciben el nombre de S-cajas (cajas de sustitución).
- La dependencia de la clave se introduce al principio de cada ronda mediante una agenda de clave (key schedule). Por ejemplo haciendo XOR del input con una subclave obtenida a partir de la clave.

Otro prinicipio de diseño

Efecto avalancha: cambiar un único bit del input debería afectar a todos los bits del output. Para conseguirlo es suficiente con:

1. Para cada S-caja, si se cambia un único bit del input, cambian al menos dos bits del output.
2. Para cada S-caja, la permutación mezcladora reparte cada bit del output a una caja distinta para la siguiente ronda.

Redes de Feistel (Feistel networks)

Es un enfoque alternativo para construir cifradores de bloque.

No se requiere que las S-cajas sean permutaciones, pero se consigue que cada ronda se pueda invertir.

Incialmente se divide el input en dos bloque L_0 y R_0 (parte izquierda y parte derecha).

Redes de Feistel (Feistel networks)

En la ronda j se calcula el output mediante una S-caja f_j (que dependerá de la clave):

- $L_j := R_{j-1}$
- $R_j := L_{j-1} \oplus f_j(R_{j-1})$

Es fácil ver que cada una de estas rondas se puede invertir; compruébalo.

El output suele darse en orden invertido (R_n, L_n), para facilitar el proceso de descifrado.

DES (Data Encryption Standard)

- Cifrador de bloque.
- Desarrollado en los 70 por IBM con la ayuda de la NSA (National Security Agency) estadounidense.
- Adoptado como estándar por EEUU en 1977.
- Actualmente se considera inseguro por su baja longitud de clave (56 bits).

El diseño de DES

- Consiste en una red de Feistel de 16 rondas.
- Longitud de bloque: 64 bits.
- Longitud de clave: 56 bits.
- Agenda de clave: en cada ronda se genera una subclave de 48 bits eligiendo 24 de la parte izquierda y 24 de la parte derecha de la clave original.
- ¡La agenda de clave es fija y pública!

El diseño de DES

- Cada función de ronda f_j se comporta del siguiente modo:
- El input de 32 bits se expande a 48 bits (duplicando la mitad). Al resultado se le aplica XOR con la subclave k_j
- A continuación se procede con una ronda de sustitución-permutación:
  1. La parte de sustitución consta de 8 S-cajas fijas que tienen entrada de 6 bits y salida de 4 bits. Por tanto se reduce la entrada de 48 bits a 32 bits de nuevo.
  2. Se aplica una permutación mezcladora a los 32 bits.

Seguridad de DES

- El mejor ataque conocido en la práctica es búsqueda exhaustiva en el espacio de claves. ¡Pero el tamaño del espacio, 2^{56}, es demasiado pequeño hoy en día!
- Un par de retos propuestos en 1997, en forma de pares input/output, con objetivo recuperar la clave, fueron resueltos (96 días, 41 días).

Seguridad de DES

- En 1998 *Deep Crack* (una máquina de 250.000 $, diseñada ad hoc) resuelve otro en 56 horas.
- Por tanto, en la forma en que hemos descrito, DES no se considera seguro actualmente.

Aumentando la longitud de clave.

- No es aconsejable modificar la estructura interna porque se puede perder la seguridad.
- DES iterado: se aplica varias veces el cifrador utilizando distintas claves

Variantes de DES

- $2DES(k_1k_2, m) := DES(k_2, DES(k_1, m))$
- $3DES3(k_1k_2k_3, m) := DES(k_3, DES^{-1}(k_2, DES(k_1, m)))$
- $3DES2(k_1k_2, m) := 3DES3(k_1k_2k_1, m)$
- $DESX(k\ k_1k_2, m) := k_2 \oplus DES(k, k_1 \oplus m)$

Variantes de DES, inconvenientes

Los iterados son poco eficientes.

La longitud de clave efectiva baja por ataques meet-in-the-middle.

La longitud de bloque se mantiene baja y esto también es una debilidad.

¡¡Se hace necesario buscar un nuevo estándar, con mayores longitudes de clave y bloque (y más eficiente)!!

AES (Advanced Encryption Standard)

- En 1997 el National Institute of Standards and Technology (NIST) estadounidense convocó una competición para seleccionar un nuevo cifrador en bloque que sustituya a DES.
- La competición fue de carácter abierto y 15 equipos de expertos de todo el mundo presentaron sus propuestas.

AES (Advanced Encryption Standard)

- Cada una de las propuestas fue detenidamente analizada por el NIST y, sobre todo, por el resto de los equipos.
- En 1998 y 1999 se celebraron congresos en los que se expusieron los criptoanálisis de las propuestas. Tras ellos se eligieron 5 finalistas.

AES (Advanced Encryption Standard)

- En octubre de 2000 se eligió el algoritmo ganador: Rijndael. Diseñado por los belgas Daemen y Rijmen, pasaría a ser el AES.
- Se reconoció que los 5 candidatos finalistas eran excelentes cifradores en bloque y no se encontraron vulnerabilidades para ninguno de ellos.

El diseño de AES

- Longitud de bloque: 128 bits.
- Longitud de clave: variable (128, 192 ó 256 bits).
- Número de rondas:

  10 para clave de 128 bits.

  12 para clave de 192 bits.

  14 para clave de 256 bits.

El diseño de AES

- Es, esencialmente, una red de sustitución-permutación.
- Un array 4x4 de bytes, llamado *estado* (*state*) se va modificando a lo largo de las rondas.
- El estado inicial es el input del cifrador.
- En cada ronda se aplican 4 operaciones al estado.

El diseño de AES

1. AddRoundKey: se genera una subclave de 128 bits a partir de la clave, se interpreta como un array 4x4 de bytes y se hace XOR con el estado.
2. SubBytes: cada byte del estado se sustituye por otro mediante una S-caja fija para todo el algoritmo.

El diseño de AES

3. ShiftRows: los bytes de cada fila del estado se desplazan hacia la izquierda
    - 0 posiciones en la primera fila
    - 1 posición en la segunda fila
    - 2 posiciones en la tercera fila
    - 3 posiciones en la cuarta fila
4. MixColumns: se multiplica el estado por una matriz 4x4 invertible fija.

El diseño de AES

En la última ronda se sustituye el último paso de MixColumns por un AddRoundKey adicional.

Esto evita que se puedan invertir los 3 últimos pasos.

Obs: para cada clave AES es una permutación; esto se sigue de que cada uno de los pasos se puede invertir.

Seguridad de AES

AES ha sufrido un intenso escrutinio tanto durante la fase de selección como posteriormente y no se ha encontrado ninguna vulnerabilidad.

Los únicos ataques no triviales son a variantes con menos rondas de AES y, aún así, no son eficientes.

Seguridad de AES

- Actualmente AES se considera un cifrador en bloque muy seguro y eficiente. Además es libre y está estandarizado.
- Buena elección para cualquier protocolo en el que se necesite una permutación pseudoaleatoria.

Tablas/Resumen de seguridad (para BC y SC

Table 4.2: Block Cipher Summary

Primitive	Classification	
	Legacy	Future
AES	✓	✓
Camellia	✓	✓
Serpent	✓	✓
Three-Key-3DES	✓	✗
Two-Key-3DES	✓	✗
Kasumi	✓	✗
Blowfish$^{\geq 80}$-bit keys	✓	✗
DES	✗	✗

Table 4.4: Stream Cipher Summary

Primitive	Classification	
	Legacy	Future
HC-128	✓	✓
Salsa20/20	✓	✓
ChaCha	✓	✓
SNOW 2.0	✓	✓
SNOW 3G	✓	✓
SOSEMANUK	✓	✓
Grain 128a	✓	✓
Grain	✓	✗
Mickey 2.0	✓	✗
Trivium	✓	✗
Rabbit	✓	✗
A5/1	✗	✗
A5/2	✗	✗
E0	✗	✗
RC4	✗	✗

Tablas/Resumen de seguridad (para BC y SC

Table 5.1: Symmetric Key Encryption Summary Table

Scheme	Legacy	Future	Notes
Block Cipher Modes of Operation			
OFB	✓	✗	No padding required
CFB	✓	✗	No padding required
CTR	✓	✗	No padding required
CBC	✓	✗	
ECB	✗	✗	
XTS	✓	✗	
EME	✓	✓	
FFX	✓	✓	
Authenticated Encryption			
Generic Composition	✓	✗	Encrypt-then-MAC, and other variants
CCM	✓	✗	Superseded by EAX
CWC	✓	✗	Superseded by GCM
OCB	✓	✓	
EAX	✓	✓	
GCM	✓	✓	
ChaCha20+Poly1305	✓	✓	