Práctica 3
Cifrado en flujo

Índice

Para entregar

- Carpeta "flujo" con el código de las funciones $vernam()$, $LFSR()$, $cifvernam()$ y $decvernam()$ completado.

1. Introducción

Los criptosistemas de *Cifrado en flujo* son procedimientos que cifran la información del mensaje en claro bit a bit mediante una función de transformación que puede variar.

Requieren de una clave pseudoaleatoria secreta compartida por emisor y receptor. La clave se genera al tiempo que se cifra y se usa para cifrar y descifrar. Constan de dos componentes básicas: el **generador de claves aleatorias** (RKG) y el **cifrador** (E) propiamente dicho. La comunicación es segura si la clave y el dispositivo cifrador se mantienen secretos en emisor y receptor.

El generador de claves RKG produce una secuencia binaria pseudoaleatoria (z_i) a partir de una clave de inicialización K:

$$\mathrm{RKG}(K) = (z_i).$$

El cifrador E realiza operaciones booleanas reversibles entre un bit z_i de la secuencia y un bit m_i del mensaje para obtener el bit c_i del cifrado:

$$c_i = \mathrm{E}(z_i, m_i).$$

2. Cifrado de Vernam

Si el cifrado se efectúa mediante una función XOR bit a bit entre el mensaje en claro y la secuencia pseudoaleatoria obtenemos un *Cifrado de Vernam*.

- Programar una función *vernam*() que admita como entrada un vector de bits que representa un mensaje en claro y un vector de bits que representa una clave, y cuya salida sea el vector de bits que representa el mensaje cifrado. La longitud de la clave debe ser igual o mayor que la del mensaje (en caso contrario, se debe generar un mensaje de error).

  Observaciones:

  1. Sólo necesitaremos una línea de código en la que hagamos un XOR entre dos secuencias de bits.
  2. Para descifrar se emplea la misma función, puesto que

$$c_i = m_i \oplus z_i,$$

$$m_i = c_i \oplus z_i = m_i \oplus z_i \oplus z_i = m_i.$$

EJEMPLO.

```
> source("flujo/vernam.R")
> m <- c(0, 1, 1, 0, 1, 0, 0, 1)
> clave1 <- c(1, 1, 1, 1, 0)
> c1 <- vernam(m, clave1)
Error en vernam(m, clave1) :
  La longitud de la clave es menor que la del mensaje
> clave2 <- c(1, 1, 1, 1, 0, 0, 1, 0, 1, 1, 0)
> c2 <- vernam(m, clave2)
> c2
[1] 1 0 0 1 1 0 1 1
> vernam(c2, clave2)
[1] 0 1 1 0 1 0 0 1
```

3. Registros de desplazamiento (LFSRs)

Un registro de desplazamiento retroalimentado lineal (Linear Feedback Shift Register) genera una secuencia de n bits

$$s^n = (s_0, s_1, \ldots, s_{n-1}).$$

Si $s0$ es el estado inicial del registro:

$$s0 = (s_0, \ldots, s_{L-1})$$

y

$$C(D) = 1 + c_1 D + c_2 D^2 + ... + c_L D^L$$

es el polinomio de conexión, entonces

$$s_j \equiv c_1 s_{j-1} + \cdots + c_L s_{j-L} \quad \text{mód } 2, \quad j = L \ldots, n.$$

Las secuencias generadas por registros de desplazamiento son periódicas de período menor o igual que $2^L - 1$. Para que presenten período máximo el polinomio $C(D)$ debe ser primitivo.

- Programar una función ($LFSR()$) que admita como entradas el vector $c = (c_1, \ldots, c_L)$ de coeficientes del polinomio de conexión de un LFSR, el estado inicial $s0 = (s_0, \ldots, s_{L-1})$ y un número natural n, y devuelva como salida un vector de n bits generado por el LFSR.

  Observación: Notar que en la entrada omitimos el término independiente del polinomio. Es decir, en lugar de introducir todos los coeficientes del polinomio $(1, c_1, \ldots, c_L)$, introducimos como entrada sólo los "términos operativos" $(c_1, \ldots, c_L)$.

Algunos ejemplos de polinomios primitivos módulo 2:

$$C(D) = D^5 + D^3 + 1, \quad C(D) = D^7 + D^3 + 1, \quad C(D) = D^{11} + D^2 + 1,$$

$$C(D) = D^{19} + D^5 + D^2 + D + 1, \quad C(D) = D^{134} + D^{57} + 1.$$

EJERCICIOS. Generar las secuencias de longitud $2^L - 1$ producidas por un LFSR con polinomio de conexión $C(D)$ y estado inicial $s0$.

1. $C(D) = 1 + D + D^4$, $s0 = (s_0, s_1, s_2, s_3) = (0, 1, 1, 0)$.
   Solución: 011001000111101.

2. $C(D) = 1 + D + D^4$, $s0 = (s_0, s_1, s_2, s_3) = (0, 0, 1, 1)$.
   Solución: 001111010110010.

3. $C(D) = 1 + D^3 + D^5$, $s0 = (s_0, s_1, s_2, s_3, s_4) = (0, 0, 1, 1, 0)$.
   Solución: 00110111010100001001011001111110.

4. $C(D) = 1 + D + D^2 + D^3$, $s0 = (s_0, s_1, s_2) = (0, 0, 1)$.
   Solución: 0011001.

4. Cifrado de Vernam utilizando una secuencia generada por un LFSR

En esta sección cifraremos y descifraremos mensajes utilizando un cifrado de Vernam con una clave generada con un LFSR dado. Para ello necesitaremos los algoritmos de las secciones anteriores.

- Programar una función ($cifvernam()$) que admita como entradas un alfabeto, un mensaje escrito en dicho alfabeto, un número natural k y dos vectores $c = (c_1, \ldots, c_L)$ y $s0 = (s_0, \ldots, s_{L-1})$ que representan los coeficientes del polinomio de conexión $C(D) = 1 + c_1 D + c_2 D^2 + ... + c_L D^L$ y el estado inicial de un LFSR, y devuelva como salida el vector de bits que representa el mensaje cifrado con un cifrado de Vernam utilizando como clave la secuencia generada por el LFSR. La transformación del mensaje en bits se efectúa partiendo el mensaje en k-gramas y calculando el equivalente binario de cada k-grama.

Los pasos que habrá que seguir son:

- Convertir el mensaje en bits con la función $men2bit()$.
- Calcular el número de bits que debe tener la clave.
- Obtener la clave.
- Cifrar.

EJERCICIO. Utilizando la clave generada con un LFSR con polinomio de conexión $C(D) = 1 + D^3 + D^5$ y estado inicial $s0 = (s_0, s_1, s_2, s_3, s_4) = (0, 0, 1, 1, 0)$ cifrar el mensaje "PRUEBA" sabiendo que ha sido escrito en el alfabeto de 26 letras ($LETTERS$) y transformado en bits calculando el equivalente binario de cada letra.
Solución: 01001011001110001101011011111 1

- Programar una función ($decvernam()$) que admita como entradas un alfabeto, un vector de bits que representa un mensaje cifrado, un número natural k y dos vectores $c = (c_1, \ldots, c_L)$ y $s0 = (s_0, \ldots, s_{L-1})$ que representan los coeficientes del polinomio de conexión $C(D) = 1 + c_1 D + c_2 D^2 + ... + c_L D^L$ y el estado incicial de un LFSR, y devuelva como salida el mensaje en claro, sabiendo que ha sido cifrado con un cifrado de Vernam utilizando como clave la secuencia generada por el LFSR. La transformación del mensaje en bits se ha efectuado partiendo el mensaje en k-gramas y calculando el equivalente binario de cada k-grama.

Los pasos que habrá que seguir son:

- Calcular el número de bits que debe tener la clave.
- Obtener la clave.
- Descifrar.
- Convertir el descifrado en texto con la función $bit2men()$.

EJERCICIO. Descifrar los dos mensajes:

$$cif1 = 01001011001110001101011011111 1$$

$$cif2 = 10100010010011101101111011111100000011 10$$

sabiendo que han sido cifrados utilizando la clave generada con un LFSR con polinomio de conexión $C(D) = 1 + D^3 + D^5$ y estado inicial $s0 = (s_0, s_1, s_2, s_3, s_4) = (0, 0, 1, 1, 0)$. El mensaje en claro ha sido escrito en el alfabeto de 26 letras ($LETTERS$) y transformado en bits calculando el equivalente binario de cada letra.
Solución: "PRUEBA"; "SUPERADA".