# Cifrado Parcial

## Conceptos y algoritmo

**Marcos Daniel C. Calderón**

2014

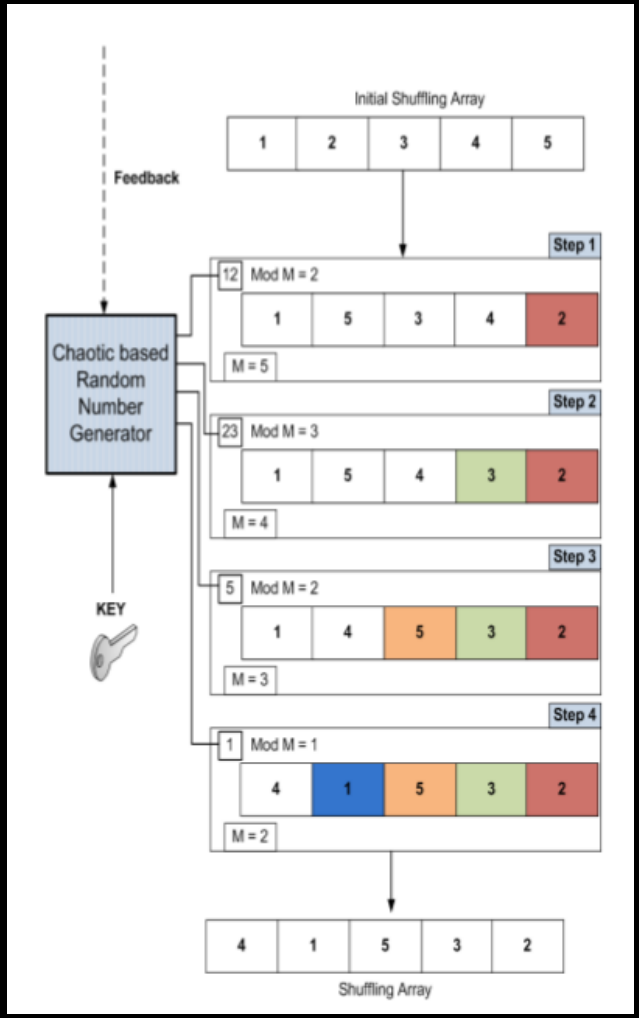
## Definición.

El cifrado parcial consiste en fragmentar la información en pequeños pedazos, después, se aplica un método de cifrado a cada una de las partes obtenidas.

# Ventajas del cifrado Parcial

- Reducción del tiempo de procesamiento.
- Reducción del ancho de banda requerido.
- Muy útil para esquemas de cifrado de video e imágemes.

# Parte 1. Permutación de segmentos.



# Parte 1. Un código para permutación...

```
/*========== Proceso de Permutacion ==================*/
srand (time(NULL));
M=10;
for(iSeg =1; iSeg < Lsegmentos; iSeg++){
M=M-1;
R= rand(); //Generate random number R
T= R%M;

/*Ahora, hacemos el intercambio que sea necesario.*/
for(iTamSeg=0; iTamSeg<tamSegmento; iTamSeg++){
  pos1=(tamSeg*(T)+iTamSeg) + (irtp*tamRTP);
  pos2=(tamSeg*(M)+iTamSeg) + (irtp*tamRTP);
  auxiliar = datosArchivo[pos1];
  datosArchivo[pos1]=datosArchivo[pos2];
  datosArchivo[pos2]= auxiliar;
        }
}
/*=====================================================*/
```

# Parte 1. Resultados

A continuación se muestra un ejemplo sencillo de los segmentos que se van a intercambiar en un paquete RTP que fué dividido en 10 segmentos.

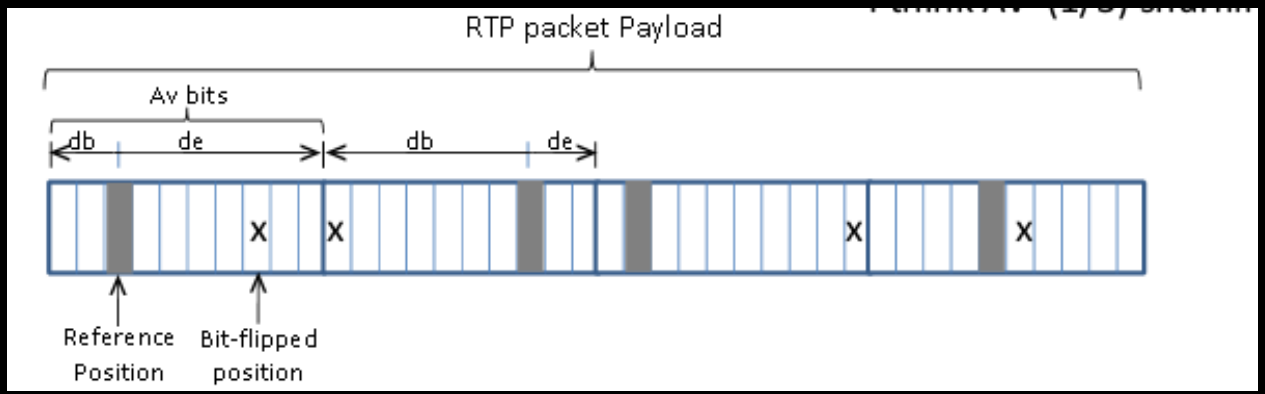
```
El paquente numero  1:
"Resultados similares"

El paquente numero  2:
T: 8     M: 9
T: 2     M: 8
T: 5     M: 7
T: 3     M: 6
T: 4     M: 5
T: 1     M: 4
T: 1     M: 3
T: 1     M: 2
T: 0     M: 1
```

# Parte 1. Recomendaciones para el proceso de permutación.

- Asegurarse que siempre haya intercambio. (T debe ser distinto de M en el esquema presentado).
- Justo antes de ejecutar el intercambio, aplicar el proceso de inversión de bits.

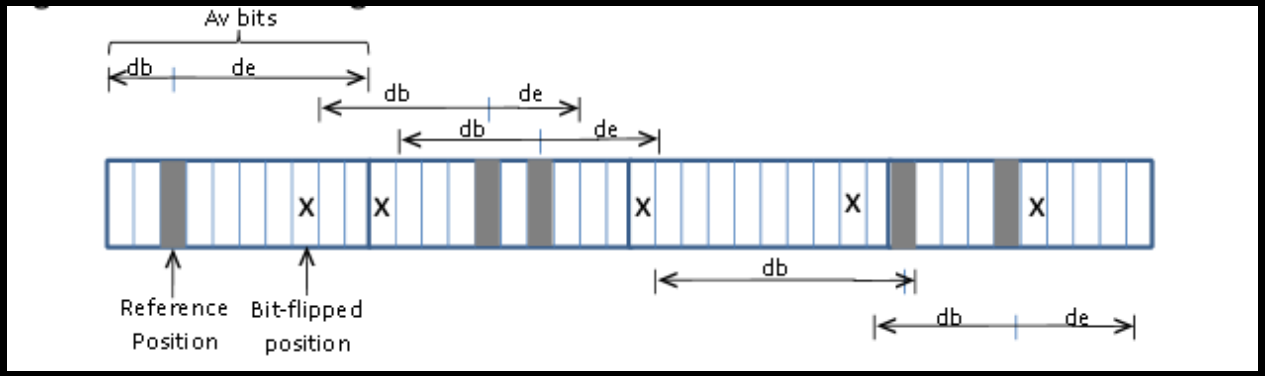
# Parte 2. Inversión de Bits. Criterio 1.

- ¿Qué pasa si el bit de referencia queda en un extremo? Mejor aseguarse SIEMPRE que la referencia no quede en un extremo para que de y db estén bien definidos.



# Parte 2. Inversión de Bits. Criterio 2.

- En este caso, el bit de referencia será aquel que se haya invertido en una iteración anterior. (Con este esquema se evitan huecos).



# Parte 2. Inversión de bits hasta el momento.

La inversión de bits en este momento consta de los siguientes pasos:

- Supongamos (como ocurre en el programa actual), que el tamaño del segmento **a intercambiar** es de 20 bytes (160 bits), entonces dividimos entre 32 bits, para obtener un total de 5 iteraciones.
- Ahora, para cada una de las 5 iteraciones, un apuntador se mueve a lo largo del segmento cada 4 bytes (32 bits) y se aplica la inversión de bits de acuerdo a los criterios mencionados.

```
p = IC_int[count]&7;
pos_f+= p;
if(p&1){
  step = IC_int[count+1]%p;
  pos_f -= step;
}
else{
  step = IC_int[count+1]%(BlockSize - p);
  pos_f += step;
}
Pack_temp[pos_byte + (pos_f/8)]^= 1<<(8 - (pos_f&7));
```

# Utilización de mapas caóticos acoplados. Un ejemplo...

- A continuación, se muestra un conjunto de cuatro mapas acoplados donde se utiliza un fragmento de texto plano.

```
for(jj=0; jj<4; jj++){
hh+=W_int[jj]^IC_int[jj]^PText[jj];
}
hh = hh%20;

for(jj=0; jj<4; jj++){
IC_int[jj] = Renyi_int(IC_int[jj],beta[jj]) + hh;
    PText[jj] = Pack_temp[pos_byte + jj];
}
```

# Utilización de mapas caóticos acoplados. Uso del texto plano.

En el código existente se utilizó el texto plano de la siguiente manera:

```
for(jj=0; jj<4; jj++){
hh+=W_int[jj]^IC_int[jj]^PText[jj];
}
hh = hh%20;
```

Pero, puede haber otras alternativas.

# Observaciones finales.

- Evitar el uso de memoria auxiliar, (arreglos auxiliares).
- Evitar operaciones innecesarias (verificar que siempre ocurran tareas significativas).