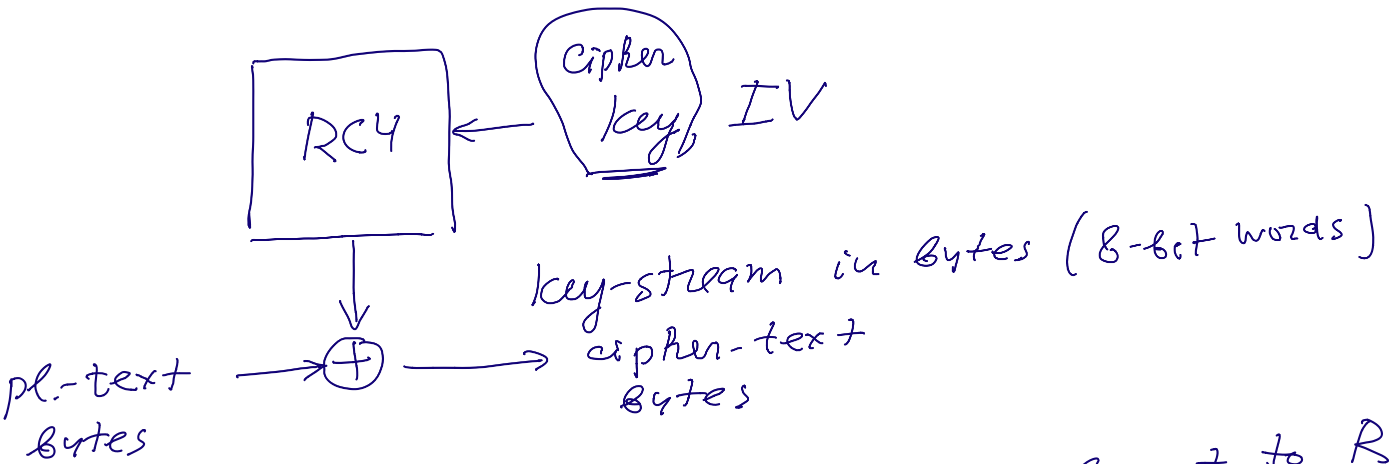
RC 4

in SSL/TLS

now not recommended due to weird correlation properties



More details. Notions relevant to RC4

1. KSA key-scheduling algorithm

2. PRGA pseudo random generation algorithm

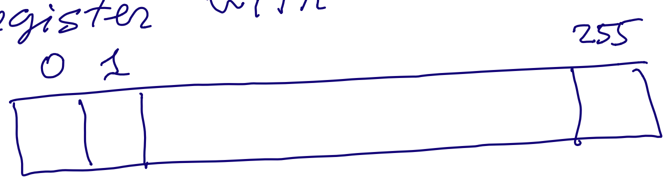
3. ℓ = keylength

length in bytes

key = key (Cipher key, IV)

4. S permutation on bytes

register with 256 cells



KSA

goal to define initial state ≡ initial permutation S.

initialization :

$i = 0, 1, \ldots, 255$

$S[i] = i$

result identity permutation $\boxed{0 \mid 1 \mid \ldots \mid 255}$

scrambling :

$j = 0$

loop $i = 0, \ldots, 255$

$j = (j + S[i] + key[i \bmod \ell]) \bmod 256$

swap $S[i] \leftrightarrow S[j]$

result 256 "random" transpositions applied to identity permutation

that permutation is initial state

encryption may start now.

PRGA

goal is to generate key-stream

initialisation : $i = 0$
$j = 0$

loop : $i = i + 1$

$j = j + S[i] \bmod 256$

swap $S[i] \leftrightarrow S[j]$

output $z = S[S[i] + S[j] \bmod 256]$

key-stream Byte.

used to encrypt current pl.-text Byte.

Biased output of RC4.

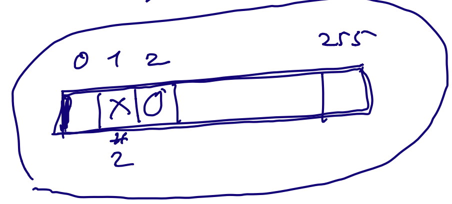
We'll find that Z_2 is biased towards 0.

we expect that $\Pr(Z_i = 0) = \frac{1}{256}$

<u>Theorem</u> (informal) Assume the initial permutation S_0 (after KS) is uniformly distributed. Then

$$\Pr(Z_2 = 0) \approx \frac{1}{128}$$

(twice larger than expected $\frac{1}{256}$).

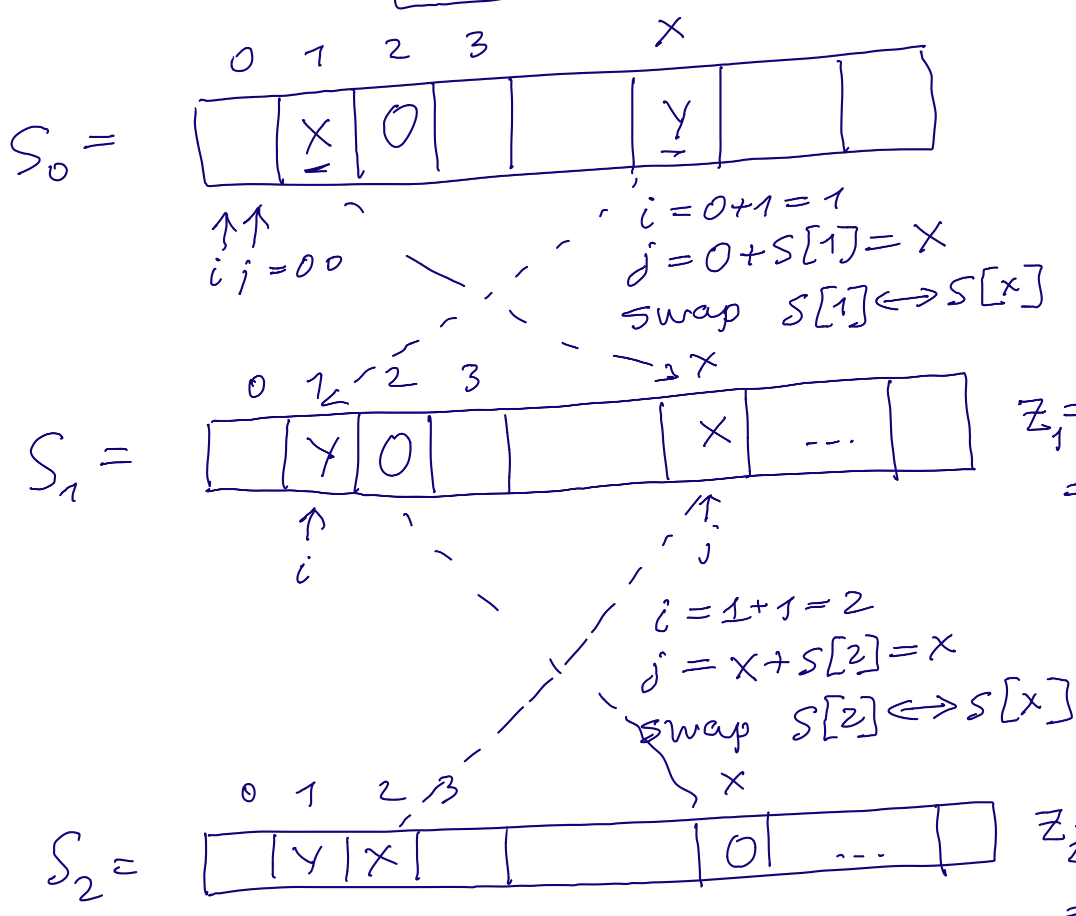


<u>Proof.</u>

observation on PRGA

assume $S_0[2] = 0$, $S_0[1] = x \neq 2$, then

$\boxed{Z_2 = 0}$ $\quad \frac{1}{128}$



$Z_1 = S\big[S[1] + S[x]\big] =$
$= S[y + x \bmod 256]$

$Z_2 = S\big[S[2] + S[x]\big] =$
$= S[x + 0] = 0$

Analyse probability

$$\Pr(Z_2=0) = \underbrace{\Pr(Z_2=0, S_0[2]=0) + \Pr(Z_2=0, S_0[2]\neq 0)}_{\text{complete probability formula}} =$$

ignore $x = S_0[1] \neq 2$

$$= \underbrace{\Pr(S_0[2]=0)}_{= 1/256} \cdot \underbrace{\Pr(Z_2=0 / S_0[2]=0)}_{\approx 1}$$

$$+ \underbrace{\Pr(S_0[2]\neq 0)}_{\approx (1-\frac{1}{256})} \cdot \underbrace{\Pr(Z_2=0 / S_0[2]\neq 0)}_{\approx \frac{1}{256}}$$

by conditional probability formula.

$$\approx \frac{1}{256} + \frac{1}{256}\left(1-\frac{1}{256}\right) \approx \boxed{\frac{1}{128}}$$

Broadcast Attack for RC4.

common attack when key-stream is not uniformly distributed.

let $M = M[1], M[2], M[3], \ldots$

message written by bytes

$C_1, C_2 \ldots C_k$ are RC4 encryptions of M on k different keys, IVs

$$C_i = RC4(M, key_i, IV_i)$$

goal observing the cipher-texts $C_1, \ldots, C_k$ get some part of M.

(*) $C_1[2], C_2[2], \dots, C_k[2]$

$\underbrace{x = M[2]}$ is the most frequent byte in (*)

why $\underbrace{\nu_0, \nu_1, \dots, \nu_{255}}$ frequencies of bytes in (*)

$$\nu_y \approx \frac{K}{256} \quad \text{if } y \neq x$$

$$\nu_x \approx \frac{K}{128}$$

even $\underbrace{\nu_x = \max(\nu_0\ \nu_1 \dots \nu_{255})}$,

that works if K is large enough.

Application of this broadcast attack.

M = Attack or Retreat

by observing $M[2]$ one recovers M.