Applications of Walsh-Hadamard transform in cryptanalysis.

n natural, $a = (a_1 \dots a_n)$ $a_i \in \{0,1\}$

$b = (b_1 \dots b_n)$ $b_i \in \{0,1\}$

$$a \cdot b = \sum_{i=1}^{n} a_i b_i \bmod 2$$

Hadamard matrix of size $2^n \times 2^n$

$$H_n = \begin{array}{c} \\ 0 \\ 1 \\ \vdots \\ a \\ \vdots \\ 2^n-1 \end{array} \begin{array}{c} \begin{array}{ccccc} 0 & 1 & \cdots & b & \cdots & 2^n-1 \end{array} \\ \left(\begin{array}{ccccc} & & & & \\ & & & & \\ & & & (-1)^{a \cdot b} & \\ & & & & \\ & & & & \end{array} \right) \end{array}$$

Example. $H_1 = \begin{array}{c} \\ 0 \\ 1 \end{array} \begin{array}{c} \begin{array}{cc} 0 & 1 \end{array} \\ \begin{pmatrix} (-1)^{0 \cdot 0} & (-1)^{0 \cdot 1} \\ (-1)^{1 \cdot 0} & (-1)^{1 \cdot 1} \end{pmatrix} \end{array} = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$

$$H_n = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}^{[n]}, \qquad H_n = H_1 \otimes H_{n-1} = \begin{pmatrix} H_{n-1} & H_{n-1} \\ H_{n-1} & -H_{n-1} \end{pmatrix}$$

$$H_2 = H_1^{[2]} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix}$$

property of H_n is

$$H_n \cdot H_n = 2^n \cdot I_{2^n}$$

I_{2^n}: identity matrix of size $2^n \times 2^n$.

Walsh-Hadamard transform

$A = (A_0 A_1 \dots A_{2^n-1})$ $A_i \in \mathbb{R}$

$W = A \cdot H_n$ of A.

Fast algorithm to compute WH transform

time complexity $n \cdot 2^n$ add/subtr. of reals.

memory 2^n locations to keep

numerous applications in cryptanalysis.

1. Boolean function $F = F(X_1 \dots X_n)$

task find best aff. approx. to F

max (min) $Pr(F = a \cdot X)$

$a \cdot X = \sum_{i=1}^{n} a_i X_i$ linear Boolean function.

WH spectrum for F

$(W_0 W_1 \dots W_{2^n-1}) = (\underbrace{(-1)^{F(0)}, (-1)^{F(1)}, \dots, (-1)^{F(2^n-1)}}_{2^n}) \cdot H_n$

$F(0) = F(0\dots0)$, $F(1) = F(0\dots01)$, $F(2^n-1) = F(1\dots1)$

$$Pr(F = ax) = \frac{1 - W_a}{2}$$

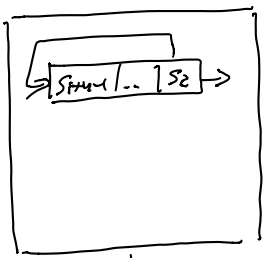
By fast algorithm of $n \cdot 2^n$ op.

$\Rightarrow$ max (min) this prob. for a.

2. Improve time complexity in correlation attacks.

LFSR based stream cipher

common 1) Filter generators



$X^{n} + c_1 X^{n-1} + \cdots + c_n$
gen. pol. for the LFSR

$\downarrow z_i$ key-stream

2) combiners

3) Grain-type stream cipher: LFSR + NFSR

a priori correlation between z_i, s_i

$$Pr(z_i = s_i) = q \neq \frac{1}{2}.$$

$\Rightarrow$ apply correlation attack with aim to recover LFSR initial state.

assume $z_0 z_1 \cdots z_{N-1}$ are known.

$X = (s_{n-1} \cdots s_1 s_0)$ initial state to find.

guess X generate $s_0' \cdots s_{N-1}'$ LFSR-seq. on X

$$N_1(X) = \#\{ s_i' + z_i = 1 \quad i = 0, 1, \cdots, N-1 \}$$

decide if X is correct based on

$$\frac{N_1(X)}{N} \to 1 - q = Pr(s_i \neq z_i) \text{ for correct } X.$$

time complexity of the attack.

$2^n \cdot N$

guesses for x ↑ cost to compute $N_1(x)$

We'll see that with WH transform time complexity is

$n \cdot 2^n + N$

price to pay memory size 2^n some values.

How to do?

$N_1(x) = \#\ s_i' \neq z_i$ $\quad s_i'$ LFSR seq. gen. on x.

$N_0(x) = \#\ s_i' = z_i$

$\Rightarrow N_1(x) + N_0(x) = N.$

We want to compute $\frac{N_1(x)}{N}$

$$N_0(x) - N_1(x) = \sum_{i=0}^{N-1} (-1)^{\overbrace{z_i + s_i}^{0}}$$

(+1 for N_0, −1 for N_1)

matrix form $s_i = (0 \ldots 01) \cdot \begin{pmatrix} s_{i+n-1} \\ \vdots \\ s_{i+1} \\ s_i \end{pmatrix} =$

$= (0 \ldots 01) A^i \begin{pmatrix} s_{n-1} \\ \vdots \\ s_1 \\ s_0 \end{pmatrix} = (0 \ldots 01) A^i x$

where $A = \begin{pmatrix} c_1 & \cdots & c_n \\ 1 & & 0 \\ & \ddots & \\ 0 & & 1\ 0 \end{pmatrix}$

transpose of companion matrix for LFSR generating polynomial.

$$\begin{pmatrix} s_{i+n} \\ \vdots \\ s_{i+2} \\ s_{i+1} \end{pmatrix} = A \begin{pmatrix} s_{i+n-1} \\ \vdots \\ s_{i+1} \\ s_i \end{pmatrix}$$ by definition of the LFSR.

$$\Rightarrow N_0(x) - N_1(x) = \sum_{i=0}^{N-1} (-1)^{z_i + tA^i x}$$

vector $C = (C(0), C(1), \ldots, C(2^n - 1))$

$$C(B) = \begin{cases} (-1)^{z_i} & t \cdot A^i = B \\ 0 & \text{otherwise} \end{cases}$$

$$N_0(x) - N_1(x) = \sum_{B=0}^{2^n - 1} C(B) \cdot (-1)^{B \cdot x}$$

the vector

$W = (N_0(0) - N_1(0), N_0(1) - N_1(1), \ldots, N_0(2^n - 1) - N_1(2^n - 1))$

is WH transform of C

apply H_n to C and compute W

$= C \cdot H_n$

$N_0(x) - N_1(x) = W_x$

$N_0(x) + N_1(x) = N$

$\Rightarrow \underline{\underline{N_1(x) = \frac{N - W_x}{2}}} \quad x = 0, 1, \cdots, 2^n - 1.$

complexity

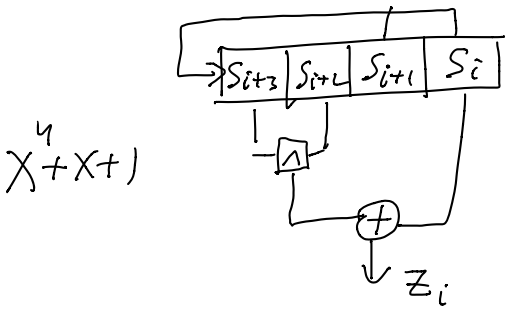
define vector C is $N + 2^n$

apply WHT transform $n \cdot 2^n$

$(n+1) \cdot 2^n + N \approx n \cdot 2^n + N$

keep vector C cost in memory locations is 2^n.

Example.



$N = 11$

key-stream given is

1 0 0 0 1 1 0 1 1 0 0

find LFSR initial state.

matrix $A = \begin{pmatrix} 0&0&1&1 \\ 1&0&0&0 \\ 0&1&0&0 \\ 0&0&1&0 \end{pmatrix}$

compute $\underset{\vee}{b} = (0001)A^i \quad i = 0,\dots,10$

i	$b=(0001)A^i$	$\underset{\vee}{b}$	z_i	$(-1)^{z_i}$
0	0001	1	1	−1
1	0010	2	0	1
2	0100	4	0	1
3	1000	8	0	1
4	0011	3	1	−1
5	0110	6	1	−1
6	1100	12	0	1
7	1011	11	1	−1
8	0101	5	1	−1
9	1010	10	0	1
10	0111	7	0	1

define vector C : $C(b) = (-1)^{z_i}$ $b = t \cdot A^i$, 0 otherwise

b	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$C(b)$	0	−1	1	−1	1	−1	−1	1	1	0	1	−1	1	0	0	0

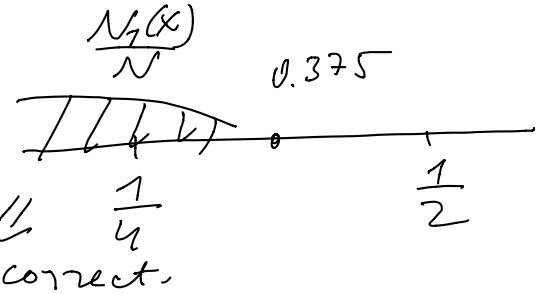
apply WHT transform to C

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	0	−1	1	−1	1	−1	−1	1	1	0	1	−1	1	0	0	0
	(−1	1)	(0	2)	(0	2)	(0	−2)	(1	1)	(0	2)	(1	1)	(0	0
	...															
W	1	7	1	3	−1	5	−1	−7	−3	−1	−3	3	−1	1	−1	−3

$\Rightarrow \quad N_1(x) = \dfrac{N - W_x}{2}$

x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$N_1(x)$	5	2	5	4	6	3	6	9	7	6	7	4	6	5	6	7

i/2 x correct $\quad \dfrac{N_1(x)}{N} \simeq \dfrac{1}{4}$



x correct.

$N_1(1) = 2$
$N_1(3) = 4$
$N_1(5) = 3$
$N_1(11) = 4$ / 11

1, 3, 5, 11 candidates

(0001)
(0011)
(0101)
(1011)

Brute force the candidates to find unique solution:

(0001)