Analysis of Grain's Initialization Algorithm

Christophe De Cannière[1,2] Özgül Küçük[1] Bart Preneel[1]

Katholieke Universiteit Leuven, Dept. ESAT/SCD-COSIC
Département d'Informatique École Normale Supérieure

Casablanca – June 12, 2008

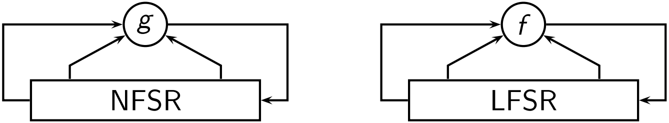
Outline

Outline

Grain

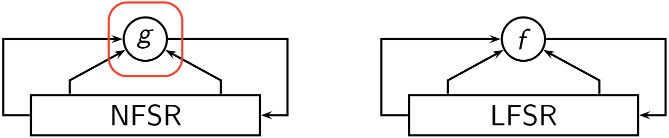
- Family of stream ciphers designed in 2005 by M.Hell, T. Johansson and W. Meier
- Has two members Grain v1 and Grain-128:
  - **Grain v1** accepts 80-bit key and 64-bit IV value
  - **Grain-128** accepts 128-bit key and 96-bit IV value
- One of 4 hardware ciphers in eSTREAM Portfolio

Keystream Generation



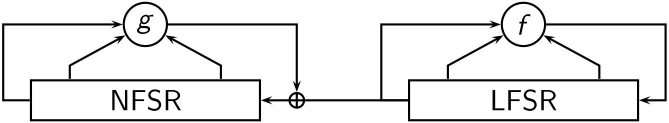
- **Grain v1:** 80-bit NFSR and 80-bit LFSR
- **Grain-128:** 128-bit NFSR and 128-bit LFSR

Keystream Generation



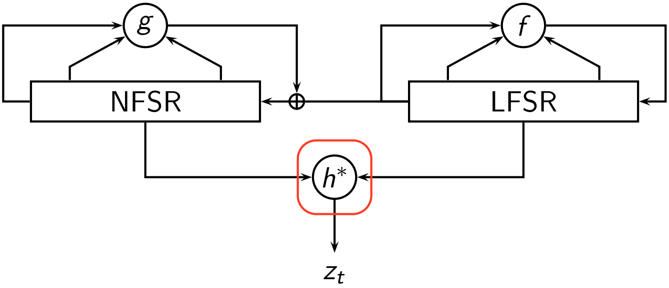
- **Grain v1:** $g(x_1 \ldots x_{13})$ is a function of degree 6
- **Grain-128:** $g(x_1 \ldots x_{19})$ is a very sparse quadratic function

Keystream Generation



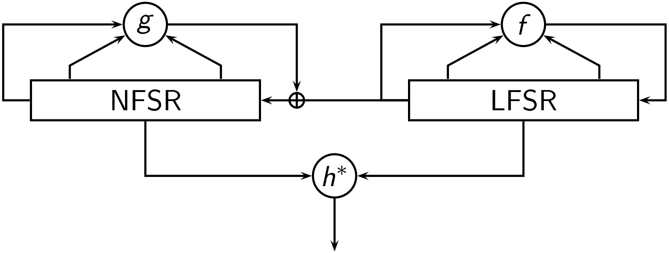
- **Grain v1:** $g(x_1 \ldots x_{13})$ is a function of degree 6
- **Grain-128:** $g(x_1 \ldots x_{19})$ is a very sparse quadratic function

Keystream Generation



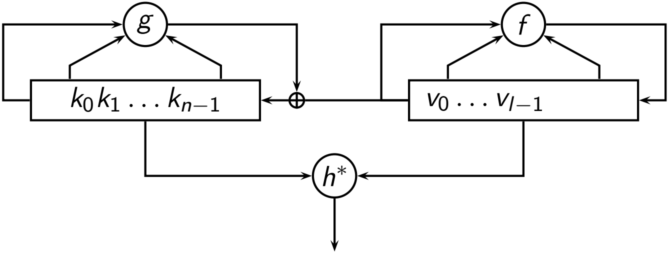
- **Grain v1:** $h^*(x_1 \ldots x_{12})$ is a function of degree 3
- **Grain-128:** $h^*(x_1 \ldots x_{17})$ is a function of degree 3

Key and IV Initialization



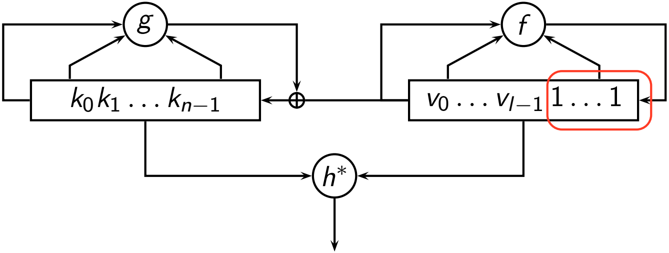
- **Grain v1:** 80-bit key and 64-bit IV
- **Grain-128:** 128-bit key and 96-bit IV

Key and IV Initialization



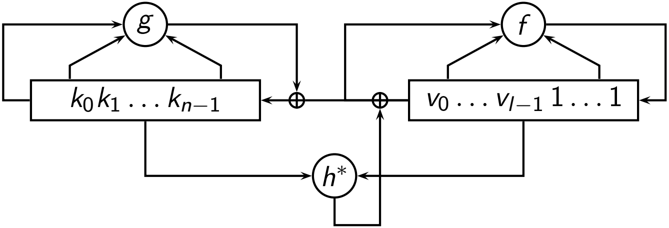
- **Grain v1:** 80-bit key and 64-bit IV
- **Grain-128:** 128-bit key and 96-bit IV

Key and IV Initialization



- **Grain v1:** 80-bit key and 64-bit IV
- **Grain-128:** 128-bit key and 96-bit IV

Key and IV Initialization



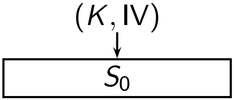
- **Grain v1:** 160 initialization rounds
- **Grain-128:** 256 initialization rounds

Outline

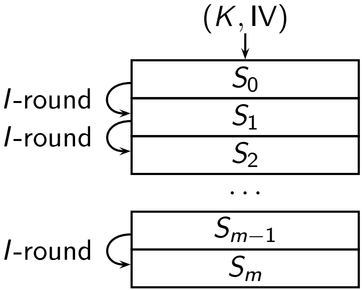
Slide Attacks

- Introduced by A. Biryukov and D. Wagner in 1999
- Mainly used to attack block ciphers
- Exploits the self-similarity of the rounds of a cipher
- Complexity is not affected by the number of rounds

Slid Pairs in Stream Ciphers

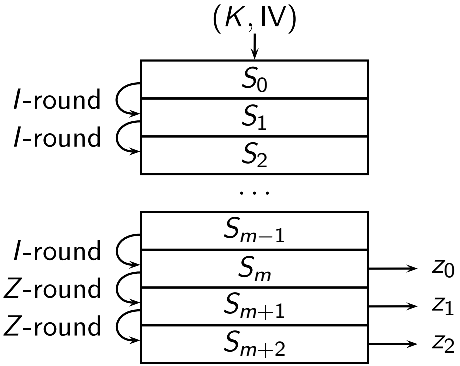


Slid Pairs in Stream Ciphers



(initialization rounds)

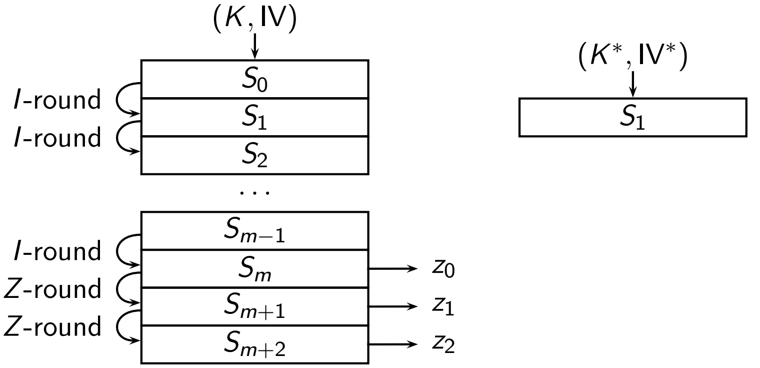
Slid Pairs in Stream Ciphers



(initialization rounds)

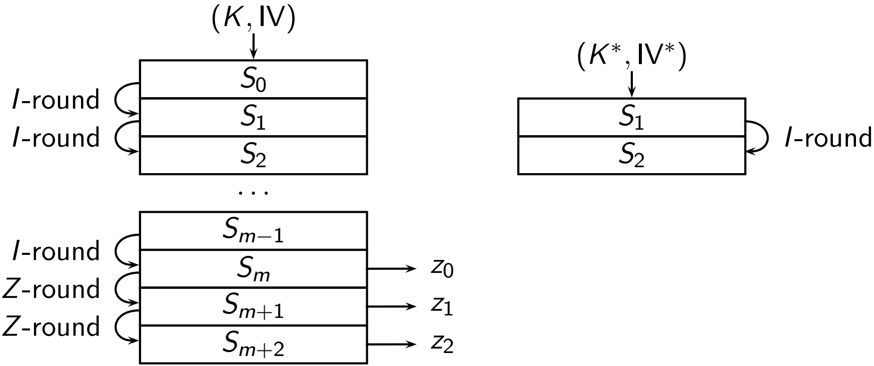
(keystream generation)

Slid Pairs in Stream Ciphers



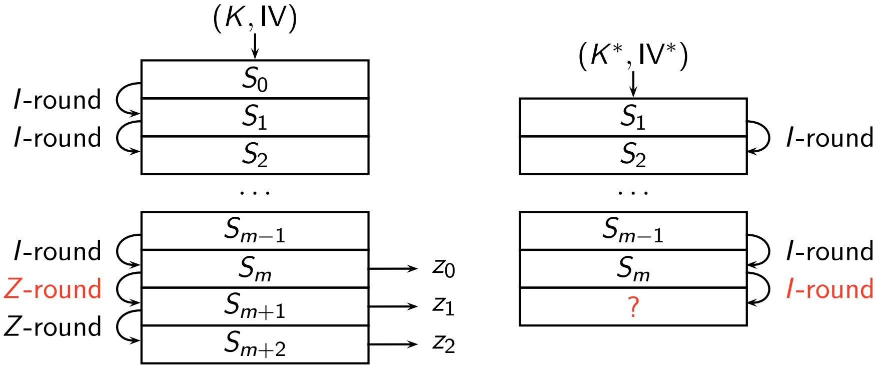
- **Condition 1:** S_1 is the initial state of a pair (K^*, IV^*)

Slid Pairs in Stream Ciphers



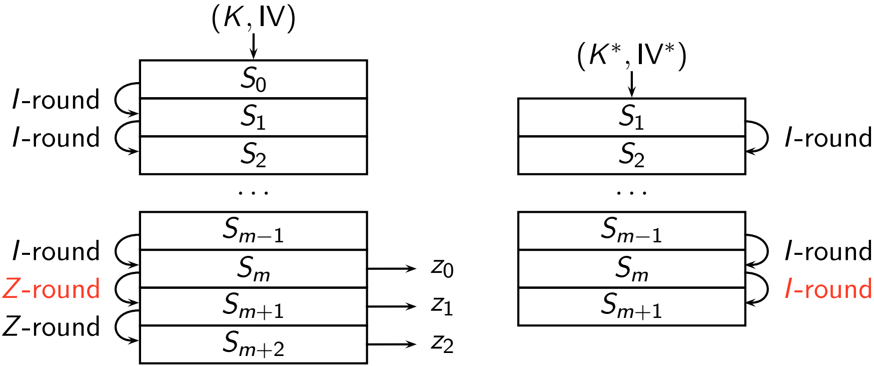
- **Condition 1:** S_1 is the initial state of a pair (K^*, IV^*)

Slid Pairs in Stream Ciphers



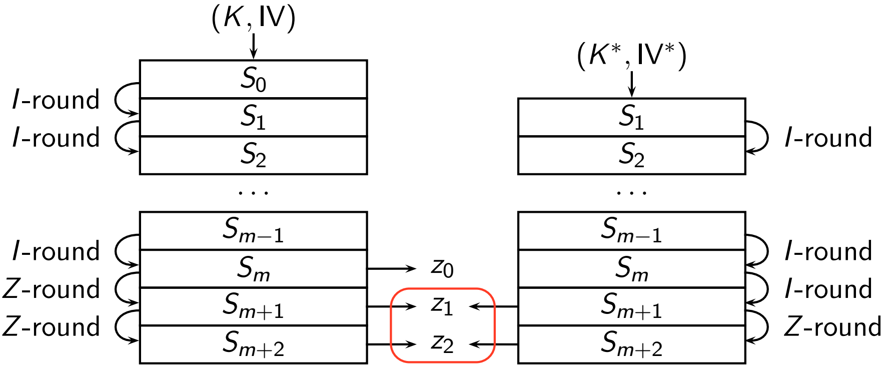
- **Condition 1:** S_1 is the initial state of a pair (K^*, IV^*)

Slid Pairs in Stream Ciphers



- **Condition 1:** S_1 is the initial state of a pair (K^*, IV^*)
- **Condition 2:** $I\text{-round}(S_m) = Z\text{-round}(S_m)$

Slid Pairs in Stream Ciphers



- **Condition 1:** S_1 is the initial state of a pair (K^*, IV^*)
- **Condition 2:** $I\text{-round}(S_m) = Z\text{-round}(S_m)$

Outline

Application to Grain

- **Condition 1:** S_1 is the initial state of a pair (K^*, IV^*)

Application to Grain

- **Condition 1:** S_1 is the initial state of a pair (K^*, IV^*)

$$S_0: \quad [k_0 \ldots k_{78} k_{79}] \quad [v_0 \ldots v_{62} v_{63} 1 \ldots 1 \quad 1]$$

Application to Grain

- **Condition 1:** S_1 is the initial state of a pair (K^*, IV^*)

$$
\begin{array}{llll}
S_0: & [k_0 \ldots k_{78} k_{79}] & [v_0 \ldots v_{62} v_{63}\, 1 \ldots 1 & 1] \\
S_1: & [k_1 \ldots k_{79} b_{80}] & [v_1 \ldots v_{63} & 1\, 1 \ldots 1\ s_{80}]
\end{array}
$$

Application to Grain

- **Condition 1:** S_1 is the initial state of a pair (K^*, IV^*)

$$
\begin{array}{llll}
S_0: & [k_0 \dots k_{78} k_{79}] & [v_0 \dots v_{62} v_{63}\, 1 \dots 1 & 1] \\
S_1: & [k_1 \dots k_{79} b_{80}] & [v_1 \dots v_{63}\quad 1\, 1 \dots 1 & 1]
\end{array}
$$

Application to Grain

- **Condition 1:** S_1 is the initial state of a pair (K^*, IV^*)

$$\begin{array}{llll} S_0: & [k_0 \ldots k_{78} k_{79}] & [v_0 \ldots v_{62} v_{63}\, 1 \ldots 1 & 1] \\ S_1: & [k_1 \ldots k_{79} b_{80}] & [v_1 \ldots v_{63} \quad 1\, 1 \ldots 1 & 1] \end{array}$$

$\Rightarrow$ occurs with probability $1/2$.

Application to Grain

- **Condition 1:** S_1 is the initial state of a pair (K^*, IV^*)

$$\begin{array}{llll} S_0: & [k_0 \ldots k_{78} k_{79}] & [v_0 \ldots v_{62} v_{63}\, 1 \ldots 1 & 1] \\ S_1: & [k_1 \ldots k_{79} b_{80}] & [v_1 \ldots v_{63} \;\; 1\, 1 \ldots 1 & 1] \end{array}$$

$\Rightarrow$ occurs with probability $1/2$.

- **Condition 2:** I-round$(S_m) = Z$-round(S_m)

Application to Grain

- **Condition 1:** S_1 is the initial state of a pair (K^*, IV^*)

$$\begin{array}{lll} S_0: & [k_0 \ldots k_{78} k_{79}] & [v_0 \ldots v_{62} v_{63} 1 \ldots 1 \quad 1] \\ S_1: & [k_1 \ldots k_{79} b_{80}] & [v_1 \ldots v_{63} \quad 1 1 \ldots 1 \quad 1] \end{array}$$

$\Rightarrow$ occurs with probability $1/2$.

- **Condition 2:** I-round(S_m) = Z-round(S_m)
  - I-round(S_m) = Z-round(S_m) $\Leftrightarrow h^*(S_m) = 0$

Application to Grain

- **Condition 1:** S_1 is the initial state of a pair (K^*, IV^*)

$$\begin{array}{llll} S_0: & [k_0 \dots k_{78} k_{79}] & [v_0 \dots v_{62} v_{63}\, 1 \dots 1 & 1] \\ S_1: & [k_1 \dots k_{79} b_{80}] & [v_1 \dots v_{63}\;\; 1\, 1 \dots 1 & \color{red}{1}] \end{array}$$

$\Rightarrow$ occurs with probability $1/2$.

- **Condition 2:** I-round$(S_m) = Z$-round(S_m)
  - I-round$(S_m) = Z$-round$(S_m) \Leftrightarrow h^*(S_m) = 0$

$\Rightarrow$ occurs with probability $1/2$.

Application to Grain

- **Condition 1:** S_1 is the initial state of a pair (K^*, IV^*)

$$\begin{array}{llll} S_0: & [k_0 \dots k_{78} k_{79}] & [v_0 \dots v_{62} v_{63} 1 \dots 1 & 1] \\ S_1: & [k_1 \dots k_{79} b_{80}] & [v_1 \dots v_{63} \; 1 1 \dots 1 & {\color{red}1}] \end{array}$$

$\Rightarrow$ occurs with probability 1/2.

- **Condition 2:** I-round(S_m) = Z-round(S_m)
  - I-round(S_m) = Z-round(S_m) $\Leftrightarrow h^*(S_m) = 0$

$\Rightarrow$ occurs with probability 1/2.

- **Remark:** What if Condition 2 is not fulfilled?

Application to Grain

- **Condition 1:** S_1 is the initial state of a pair (K^*, IV^*)

$$\begin{array}{llll} S_0: & [k_0 \ldots k_{78} k_{79}] & [v_0 \ldots v_{62} v_{63}\, 1 \ldots 1 & 1] \\ S_1: & [k_1 \ldots k_{79} b_{80}] & [v_1 \ldots v_{63} \;\; 1\, 1 \ldots 1 & 1] \end{array}$$

$\Rightarrow$ occurs with probability $1/2$.

- **Condition 2:** I-round$(S_m) = Z$-round(S_m)
  - I-round$(S_m) = Z$-round$(S_m) \Leftrightarrow h^*(S_m) = 0$

$\Rightarrow$ occurs with probability $1/2$.

- **Remark:** What if Condition 2 is not fulfilled?

$\Rightarrow$ Difference in right-most bit of NFSR and LFSR

Application to Grain

- **Condition 1:** S_1 is the initial state of a pair (K^*, IV^*)

$$\begin{array}{llll} S_0: & [k_0 \dots k_{78} k_{79}] & [v_0 \dots v_{62} v_{63}\, 1 \dots 1 & 1] \\ S_1: & [k_1 \dots k_{79} b_{80}] & [v_1 \dots v_{63} \;\; 1\, 1 \dots 1 & \color{red}{1}] \end{array}$$

$\Rightarrow$ occurs with probability $1/2$.

- **Condition 2:** I-round(S_m) = Z-round(S_m)
  - I-round(S_m) = Z-round(S_m) $\Leftrightarrow h^*(S_m) = 0$

$\Rightarrow$ occurs with probability $1/2$.

- **Remark:** What if Condition 2 is not fulfilled?

$\Rightarrow$ Difference in right-most bit of NFSR and LFSR

$\Rightarrow$ Only affects output stream after 16 (32) steps

Application to Grain

- **Condition 1:** S_1 is the initial state of a pair (K^*, IV^*)

$$\begin{array}{llll} S_0: & [k_0 \dots k_{78} k_{79}] & [v_0 \dots v_{62} v_{63}\, 1 \dots 1 & 1] \\ S_1: & [k_1 \dots k_{79} b_{80}] & [v_1 \dots v_{63}\; 1\, 1 \dots 1 & {\color{red}1}] \end{array}$$

$\Rightarrow$ occurs with probability $1/2$.

- **Condition 2:** I-round$(S_m) = Z$-round(S_m)
  - I-round$(S_m) = Z$-round$(S_m) \Leftrightarrow h^*(S_m) = 0$

$\Rightarrow$ occurs with probability $1/2$.

- **Remark:** What if Condition 2 is not fulfilled?

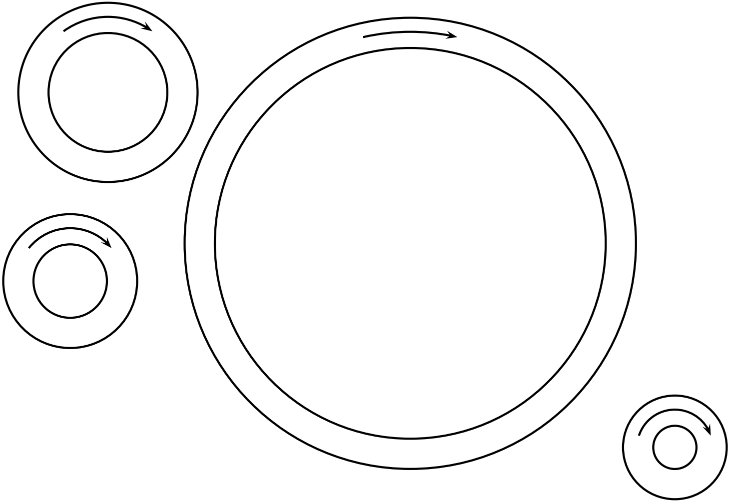
$\Rightarrow$ Difference in right-most bit of NFSR and LFSR
$\Rightarrow$ Only affects output stream after 16 (32) steps
$\Rightarrow$ First 15 (31) keystream bits are still equal (but shifted)

Related (K, IV) Pairs in Grain

Property

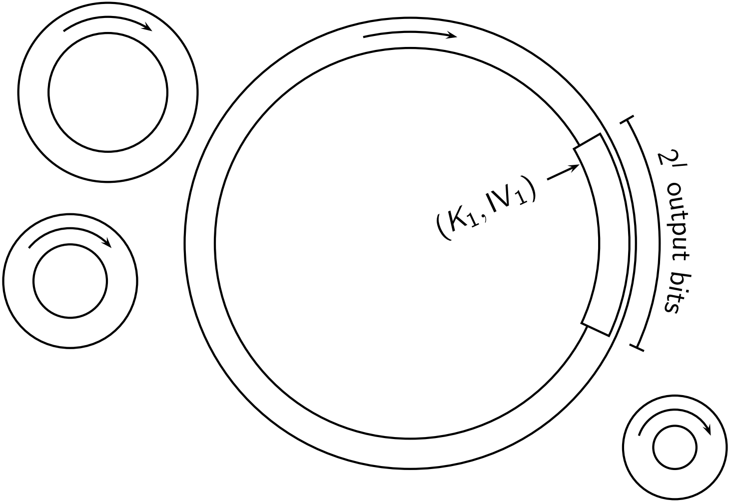
For a fraction $2^{-2 \cdot n}$ of pairs (K, IV), there exists a related pair (K^*, IV^*) which produces an identical but n-bit shifted key stream.

Is This Surprising?



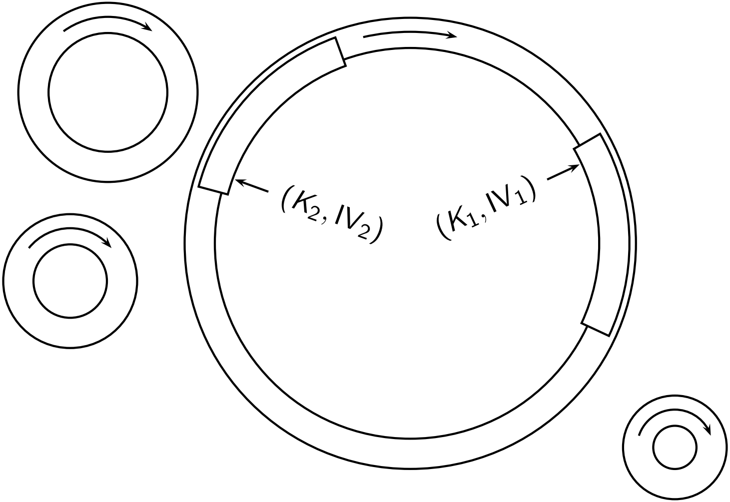
2^{160} possible states form cycles

Is This Surprising?



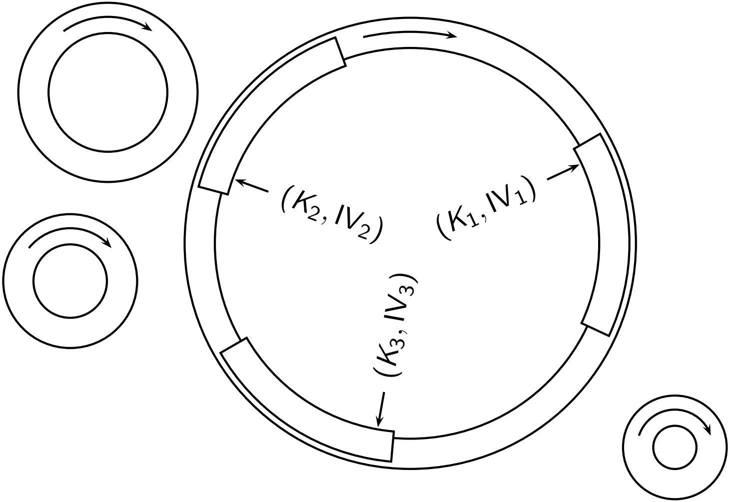
initialization algorithm defines starting point

Is This Surprising?



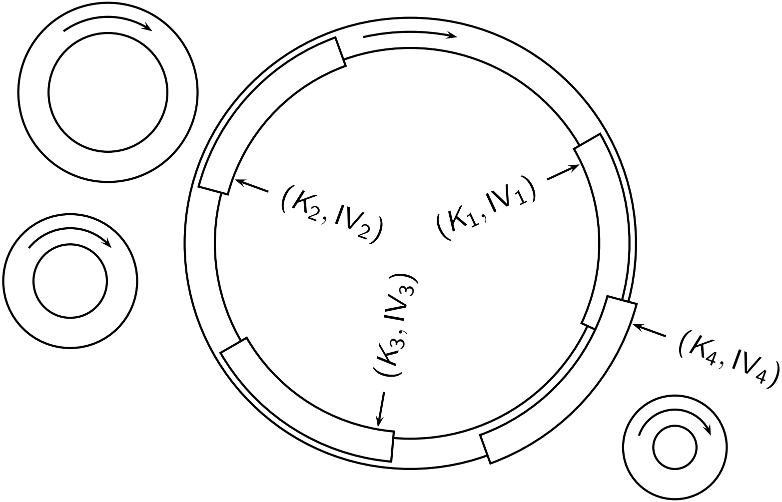
$2^{80} \times 2^{64}$ starting points

Is This Surprising?



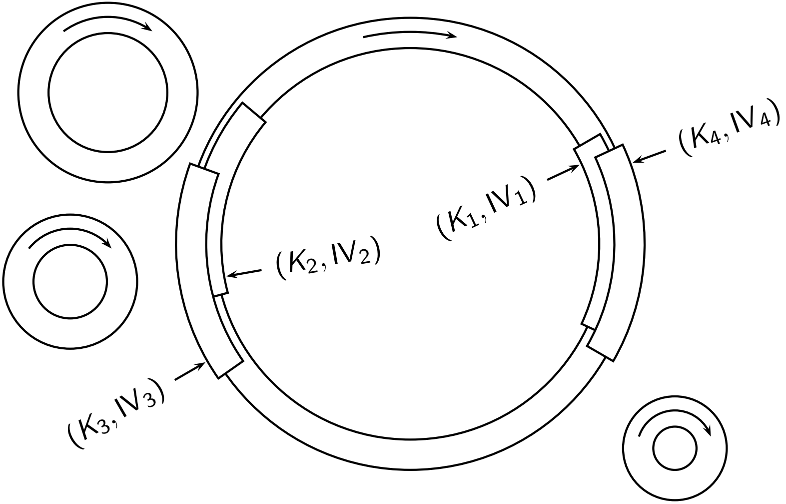
$2^{80} \times 2^{64}$ starting points

Is This Surprising?



if $2^{80} \times 2^{64} \times 2^{l} > 2^{160}$ $\longrightarrow$ overlap unavoidable

Is This Surprising?



special feature of Grain: clustering of starting points

Outline

Related Key Attack

- Assume that adversary manages to obtain keystream sequences from two shifted (K, IV) pairs

Related Key Attack

- Assume that adversary manages to obtain keystream sequences from two shifted (K, IV) pairs
  - $\Rightarrow$ With probability $1/4$, sequences are identical but shifted

Related Key Attack

- Assume that adversary manages to obtain keystream sequences from two shifted (K, IV) pairs
  - $\Rightarrow$ With probability $1/4$, sequences are identical but shifted
  - $\Rightarrow$ This indicates that $s_{80} = 1$, which yields simple (non-linear) equation in secret key bits

Related Key Attack

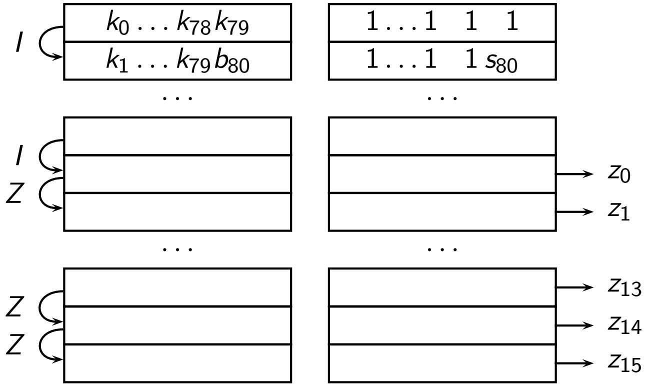
- Assume that adversary manages to obtain keystream sequences from two shifted (K, IV) pairs
  - $\Rightarrow$ With probability $1/4$, sequences are identical but shifted
  - $\Rightarrow$ This indicates that $s_{80} = 1$, which yields simple (non-linear) equation in secret key bits
- Unlikely to happen in practice, unless
  - Session keys are derived from master key in funny way
  - Adversary can cause synchronization errors

Speeding up Exhaustive Search when $IV = [1 \ldots 1]$

$k_0 \ldots k_{78} k_{79}$	$1 \ldots 1 \quad 1 \quad 1$

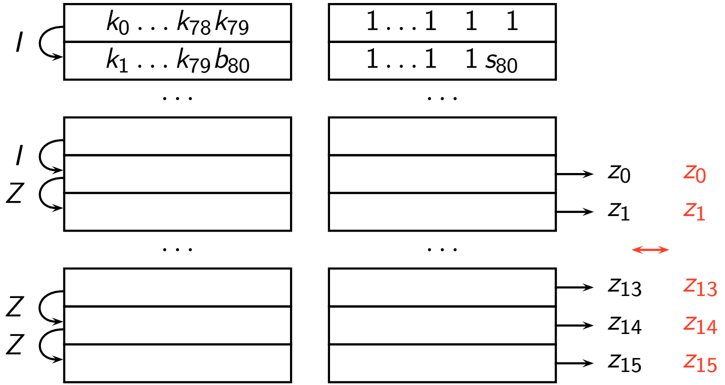
initialize Grain with arbitrary key K

Speeding up Exhaustive Search when $IV = [1 \ldots 1]$



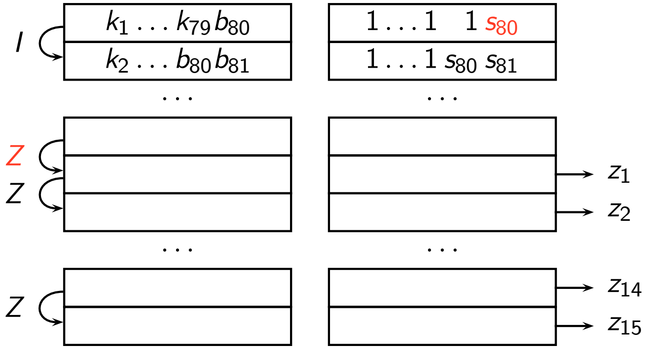
initialize Grain with arbitrary key K

Speeding up Exhaustive Search when $IV = [1 \dots 1]$



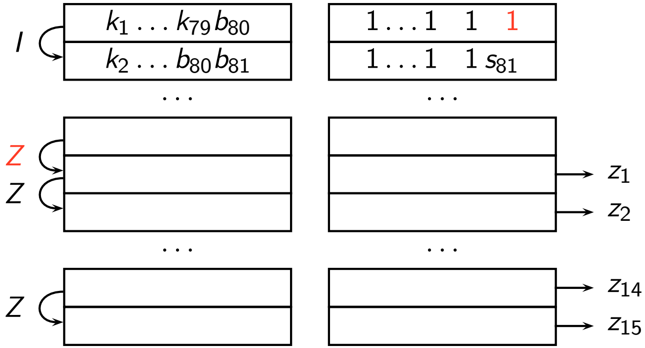
compare output with known keystream

Speeding up Exhaustive Search when $IV = [1 \dots 1]$



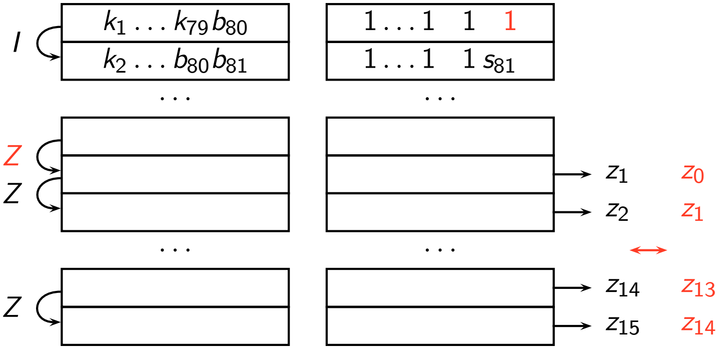
if no match, shift everything up by one step

Speeding up Exhaustive Search when $IV = [1\ldots1]$



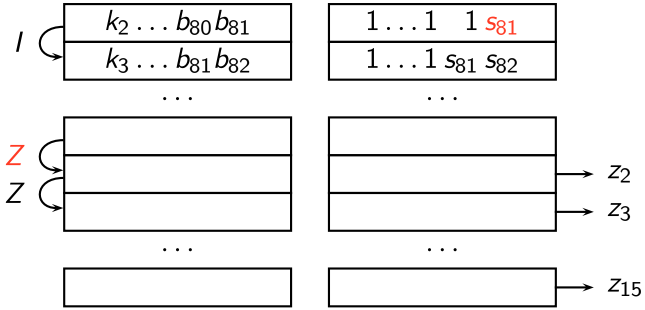
if $S_{80} = 1 \Rightarrow$ no need to recompute anything

Speeding up Exhaustive Search when $IV = [1 \ldots 1]$



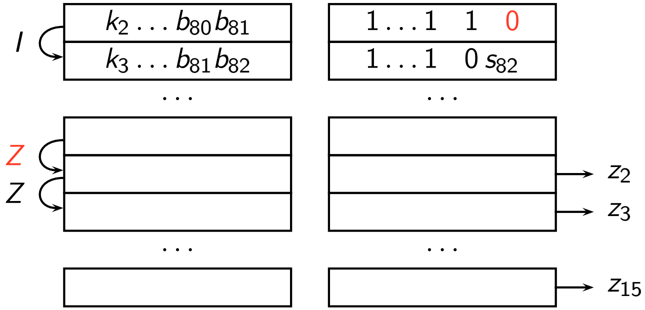
compare (shifted) output with known keystream

Speeding up Exhaustive Search when $IV = [1 \dots 1]$



if no match, shift everything up by one step

Speeding up Exhaustive Search when $IV = [1 \ldots 1]$



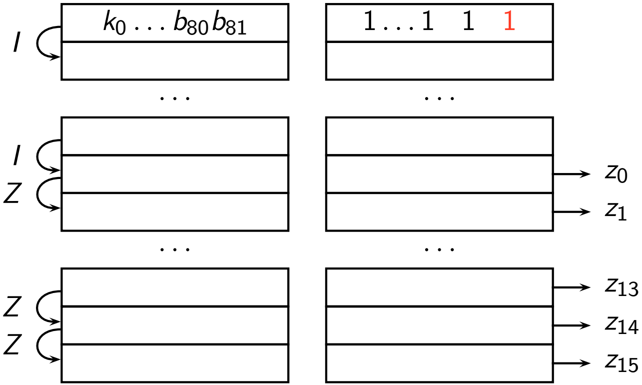
if $S_{81} = 0 \Rightarrow$ correct and rerun initialization

Speeding up Exhaustive Search when $IV = [1 \dots 1]$

$k_0 \dots b_{80} b_{81}$	$1 \dots 1 \quad 1 \quad 1$

if $S_{81} = 0 \Rightarrow$ correct and rerun initialization

Speeding up Exhaustive Search when $IV = [1 \dots 1]$



if $S_{81} = 0 \Rightarrow$ correct and rerun initialization

Speeding up Exhaustive Search

- Keys are checked in a complex order, but form a big cycle with an expected length of 2^{79}

Speeding up Exhaustive Search

- Keys are checked in a complex order, but form a big cycle with an expected length of 2^{79}
- On average, initialization algorithm only needs to be rerun for 1 out of 2 keys

  $\Rightarrow$ twice as fast as regular exhaustive search

Speeding up Exhaustive Search

- Keys are checked in a complex order, but form a big cycle with an expected length of 2^{79}
- On average, initialization algorithm only needs to be rerun for 1 out of 2 keys

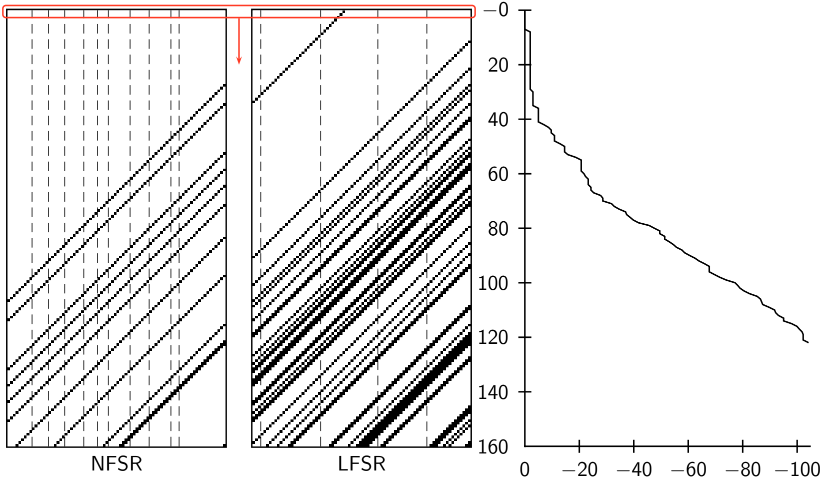
  $\Rightarrow$ twice as fast as regular exhaustive search
- Only works when $IV = [1 \ldots 1]$

Outline

Sparse Characteristics in Grain

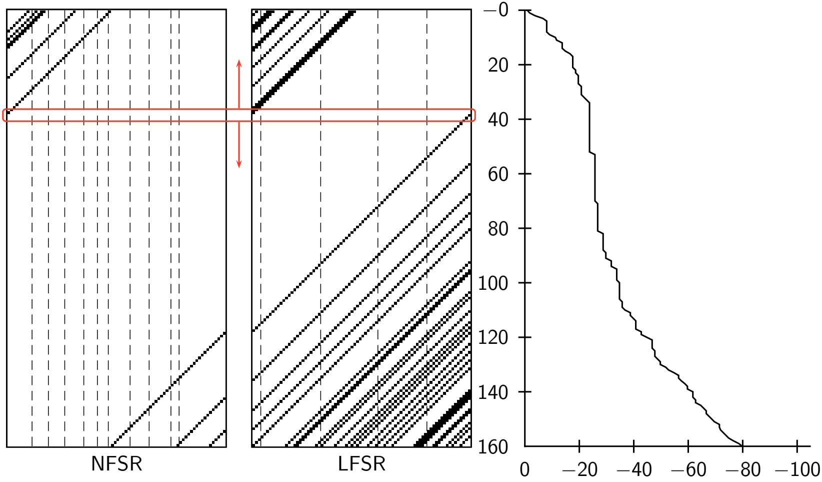
- Start with a single bit difference in the state at step t
- Propagate backwards and forwards
- Each time a difference enters the non-linear functions we have to make a choice
  - $\longrightarrow$ **Our approach:** choose the difference which introduces as few differences as possible in the next steps and in particular in NFSR

Illustration – Grain v1



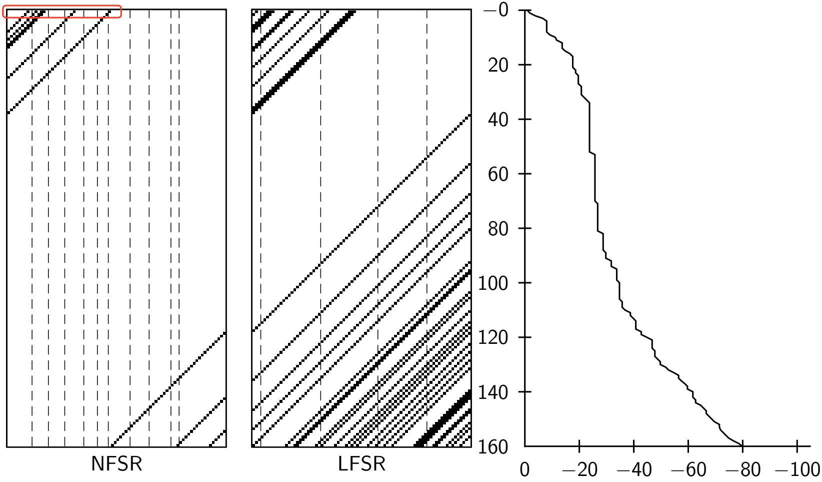
single bit difference at step 0

Illustration – Grain v1



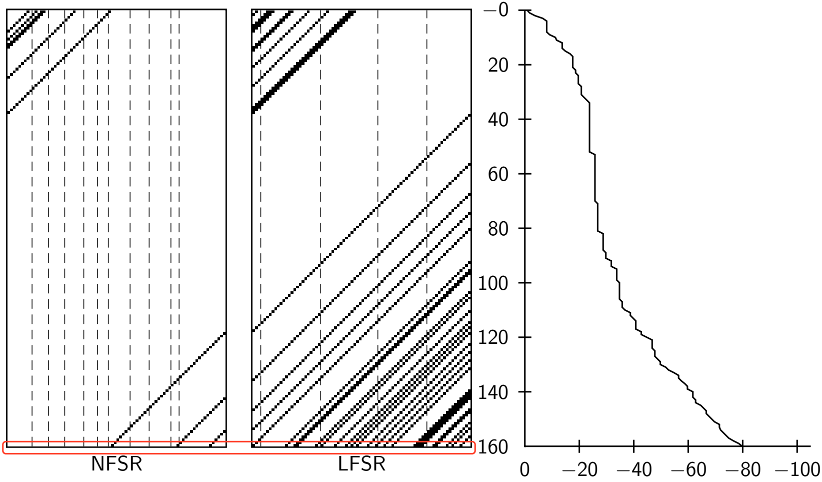
single bit difference at step 38

Illustration – Grain v1



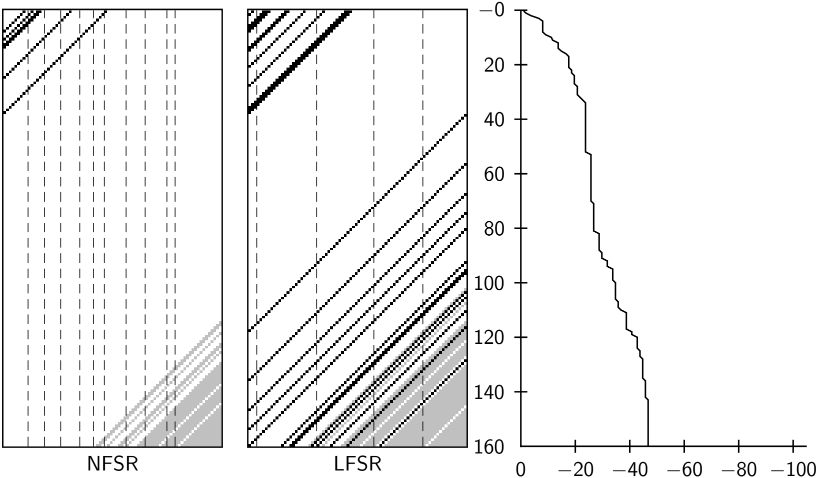
differences in NFSR at step 0 → related keys

Illustration – Grain v1



equalities in final state → equalities in a few keystream positions

Truncated Differentials



focus on first keystream position; ignore rest

Data Complexity

- Keystream can be distinguished from random by initializing with N different related pairs (K, IV_i) and $(K + K', \mathrm{IV}_i + \mathrm{IV}')$, and counting number of 0- and 1-differences in z_0.
- How large should N be to observe a bias?

Data Complexity

- Keystream can be distinguished from random by initializing with N different related pairs (K, IV_i) and $(K + K', \mathrm{IV}_i + \mathrm{IV}')$, and counting number of 0- and 1-differences in z_0.
- How large should N be to observe a bias?
  - $p_C = P(\text{characteristic is followed}) = 2^{-47}$
  - $p_R = P(\text{equality in } z_0 \text{ in random case}) = 1/2$

Data Complexity

- Keystream can be distinguished from random by initializing with N different related pairs (K, IV_i) and $(K + K', \mathrm{IV}_i + \mathrm{IV}')$, and counting number of 0- and 1-differences in z_0.
- How large should N be to observe a bias?
  - $p_C = P(\text{characteristic is followed}) = 2^{-47}$
  - $p_R = P(\text{equality in } z_0 \text{ in random case}) = 1/2$
- **Regular differential attack:** $p_C \gg p_R$

  $\Rightarrow N > 1/p_C$
- **In our case:** $p_C \ll p_R$

  $\Rightarrow N > 1/p_C^2$

Data Complexity

- Keystream can be distinguished from random by initializing with N different related pairs (K, IV_i) and $(K + K', \mathrm{IV}_i + \mathrm{IV}')$, and counting number of 0- and 1-differences in z_0.
- How large should N be to observe a bias?
  - $p_C = P(\text{characteristic is followed}) = 2^{-47}$
  - $p_R = P(\text{equality in } z_0 \text{ in random case}) = 1/2$
- **Regular differential attack:** $p_C \gg p_R$

  $\Rightarrow N > 1/p_C$
- **In our case:** $p_C \ll p_R$

  $\Rightarrow N > 1/p_C^2$

  $= 2^{94} \gg 2^{63}$ (total number of possible IV pairs)

Outline

How to Reduce N?

- Split characteristic into two parts:
  - **Part 1:** steps 0 to t (probability p_1)
  - **Part 2:** steps t to 160 (probability p_2)

How to Reduce N?

- Split characteristic into two parts:
  - **Part 1:** steps 0 to t (probability p_1)
  - **Part 2:** steps t to 160 (probability p_2)
- Try to separate the pairs (K, IV_i) and $(K + K', \mathrm{IV}_i + \mathrm{IV}')$ which satisfy Part 1 from those which do not

How to Reduce N?

- Split characteristic into two parts:
  - **Part 1:** steps 0 to t (probability p_1)
  - **Part 2:** steps t to 160 (probability p_2)
- Try to separate the pairs (K, IV_i) and $(K + K', \mathrm{IV}_i + \mathrm{IV}')$ which satisfy Part 1 from those which do not

$\Rightarrow$ **Effect:** reduces N from $(p_1 p_2)^{-2}$ to $p_1'^{-1} p_2^{-2}$

Partitioning the Key and IV Space

	IV_1	IV_2	IV_3	IV_4	...					$IV_{2^{64}}$
K_1	1	0	1	1	1	1	...	0	0	1
K_2	1	0	1	0	1	1	...	0	0	0
K_3	0	0	1	0	1	1	...	0	0	0
K_4	1	1	1	1	0	0	...	0	1	1
K_5	1	0	1	0	1	1	...	1	1	0
K_6	1	0	0	1	0	1	...	0	1	0
$\vdots$	$\vdots$					$\vdots$		$\vdots$		
	0	1	0	0	0	1	...	1	1	0
	1	0	1	0	1	1	...	1	0	1
$K_{2^{80}}$	0	1	0	1	0	0	...	1	0	1

$F_t(K_i, IV_i)$ for differences K' and IV'

Partitioning the Key Space

	IV_1	IV_2	IV_3	IV_4	...					$IV_{2^{64}}$
K_a	0	0	0	0	0	0	...	0	0	0
⋮	⋮					⋮		⋮		
K_c	0	0	0	0	0	0	...	0	0	0
K_d	0	1	1	0	0	1	...	1	0	0
⋮	⋮					⋮		⋮		
	0	1	1	0	0	1	...	1	0	0
	⋮					⋮		⋮		
	1	0	1	0	1	1	...	1	0	1
	⋮					⋮		⋮		
	1	0	1	0	1	1	...	1	0	1

sorting rows $\longrightarrow$ equivalent key classes

Partitioning the Key Space

	IV_1	IV_2	IV_3	IV_4	…					$IV_{2^{64}}$
K_a	0	0	0	0	0	0	…	0	0	0
⋮	⋮					⋮		⋮		
K_c	0	0	0	0	0	0	…	0	0	0
K_d	0	1	1	0	0	1	…	1	0	0
⋮	⋮					⋮		⋮		
	0	1	1	0	0	1	…	1	0	0
	⋮					⋮		⋮		
	1	0	1	0	1	1	…	1	0	1
	⋮					⋮		⋮		
	1	0	1	0	1	1	…	1	0	1

weak keys

sorting rows → equivalent key classes

Partitioning the IV Space

	IV_a	…	IV_c	IV_d	…					
K_a	0	…	0	0	…	0	…	0	…	0
⋮	⋮									
K_c	0	…	0	0	…	0	…	0	…	0
K_d	0	…	0	1	…	1	…	0	…	0
⋮	⋮									
	0	…	0	1	…	1	…	0	…	0
	⋮		⋮	⋮		⋮		⋮		⋮
	0	…	0	1	…	1	…	1	…	1
	⋮									
	0	…	0	1	…	1	…	1	…	1

sorting columns $\rightarrow$ equivalent IV classes

Partitioning the IV Space

weak IVs (spanning from IV_d to the last column)

	IV_a	…	IV_c	IV_d	…		…		…	
K_a	0	…	0	0	…	0	…	0	…	0
⋮	⋮									
K_c	0	…	0	0	…	0	…	0	…	0
K_d	0	…	0	1	…	1	…	0	…	0
⋮	⋮									
	0	…	0	1	…	1	…	0	…	0
	⋮		⋮	⋮		⋮		⋮		⋮
	0	…	0	1	…	1	…	1	…	1
	⋮									
	0	…	0	1	…	1	…	1	…	1

sorting columns $\rightarrow$ equivalent IV classes

Partitioning the IV Space

	IV_a	…	IV_c	IV_d	…					
K_a	0	…	0	0	…	0	…	0	…	0
⋮	⋮									
K_c	0	…	0	0	…	0	…	0	…	0
K_d	0	…	0	1	…	1	…	0	…	0
⋮	⋮									
	0	…	0	1	…	1	…	0	…	0
	⋮		⋮	⋮		⋮		⋮		⋮
	0	…	0	1	…	1	…	1	…	1
	⋮									
	0	…	0	1	…	1	…	1	…	1

when t increases $\rightarrow$ number of classes increases

How to use this?

How to use this?

- Assume that secret key is weak

How to use this?

- Assume that secret key is weak
- **Stage 1:**
  - Initialize Grain with N different weak related pairs
  - Count number of 0- and 1-differences in z_0

How to use this?

- Assume that secret key is weak
- **Stage 1:**
  - Initialize Grain with N different weak related pairs
  - Count number of 0- and 1-differences in z_0

  ⇒ Keep separate counters for each IV equivalence class

How to use this?

- Assume that secret key is weak
- **Stage 1:**
  - Initialize Grain with N different weak related pairs
  - Count number of 0- and 1-differences in z_0
  - ⇒ Keep separate counters for each IV equivalence class
- **Stage 2:**
  - Guess key equivalence class and combine counters of all IV equivalence classes for which Part 1 is satisfied
  - If no bias is detected, discard guess

Computing N

Computing N

- Probability of Part 1 can be written as $p_1 = p_K \cdot p_{\mathrm{IV}} \cdot p'_1$
  - p_K: fraction of keys which are weak
  - p_{IV}: fraction of IVs which are weak
  - p'_1: probability that Part 1 is satisfied for weak key and IV

Computing N

- Probability of Part 1 can be written as $p_1 = p_K \cdot p_{\mathrm{IV}} \cdot p'_1$
  - p_K: fraction of keys which are weak
  - p_{IV}: fraction of IVs which are weak
  - p'_1: probability that Part 1 is satisfied for weak key and IV
- For given key class guess, $M = p'_1 \cdot N$ pairs satisfy Part 1

Computing N

- Probability of Part 1 can be written as $p_1 = p_K \cdot p_{\mathrm{IV}} \cdot p'_1$
  - p_K: fraction of keys which are weak
  - p_{IV}: fraction of IVs which are weak
  - p'_1: probability that Part 1 is satisfied for weak key and IV
- For given key class guess, $M = p'_1 \cdot N$ pairs satisfy Part 1
- In order to detect bias after Part 2:

$\Rightarrow$ $M > p_2^{-2} \Rightarrow N > {p'_1}^{-1} p_2^{-2}$

Outline

Attack Complexities

Cipher	v1	v1	128	128	128
Rounds	160	112	256	224	192
Related keys	yes	no	yes	no	no
# Weak keys	2^{71}	2^{80}	2^{87}	2^{126}	2^{126}
# Weak IVs	2^{57}	2^{63}	2^{84}	2^{93}	2^{93}
N	2^{55}	(2^{72})	2^{73}	(2^{96})	2^{35}
t	33	28	75	78	76
p_1	2^{-23}	2^{-3}	2^{-64}	2^{-6}	2^{-6}
p_2	2^{-24}	2^{-35}	2^{-31}	2^{-47}	2^{-17}
# Key classes	2^{22}	8	2^{27}	72	72
# IV classes	2^{21}	8	2^{32}	64	64

Conclusions

- Sliding property in Grain allows to speed up exhaustive search

Conclusions

- Sliding property in Grain allows to speed up exhaustive search
  - $\Rightarrow$ Could be avoided if initialization used different constant

Conclusions

- Sliding property in Grain allows to speed up exhaustive search
  - $\Rightarrow$ Could be avoided if initialization used different constant
- Related key attacks against both versions of Grain

Conclusions

- Sliding property in Grain allows to speed up exhaustive search
  - $\Rightarrow$ Could be avoided if initialization used different constant
- Related key attacks against both versions of Grain
- Chosen IV attacks against reduced variants

Conclusions

- Sliding property in Grain allows to speed up exhaustive search
  - $\Rightarrow$ Could be avoided if initialization used different constant
- Related key attacks against both versions of Grain
- Chosen IV attacks against reduced variants

$\Rightarrow$ Attacks have limited practical impact, but can nonetheless be considered as non-ideal behavior