FSE 2013

Near Collision Attack on the Grain v1 Stream Cipher

Bin Zhang* and Zhenqi Li†

*Institute of Information Engineering,
Chinese Academy of Sciences, Beijing, 100093, China.
†Institute of Software,
Chinese Academy of Sciences, Beijing, 100190, China.
{zhangbin, lizhenqi}@is.iscas.ac.cn

March 13, 2013

Outline

- Introduction
- Description of Grain v1
- Main idea & some key observations
- The general attack model: NCA-1.0
- NCA-2.0 & NCA-3.0
- Simulations
- Conclusions

Introduction

- Grain v1, designed by Martin Hell, Thomas Johansson and Willi Meier, is a stream cipher for restricted hardware environments. It was selected into the final portfolio by the eSTREAM project.

Introduction

- Grain v1, designed by Martin Hell, Thomas Johansson and Willi Meier, is a stream cipher for restricted hardware environments. It was selected into the final portfolio by the eSTREAM project.
- Grain v1 is immune to the correlation and distinguishing attacks that successfully broke the former version Grain v0.

Introduction

- Grain v1, designed by Martin Hell, Thomas Johansson and Willi Meier, is a stream cipher for restricted hardware environments. It was selected into the final portfolio by the eSTREAM project.
- Grain v1 is immune to the correlation and distinguishing attacks that successfully broke the former version Grain v0.
- De Cannière. C. *et al.* discovered a slide property in the initialization phase of Grain v1, reduce half of the cost of exhaustive key search for a fixed IV.

Introduction

- Grain v1, designed by Martin Hell, Thomas Johansson and Willi Meier, is a stream cipher for restricted hardware environments. It was selected into the final portfolio by the eSTREAM project.
- Grain v1 is immune to the correlation and distinguishing attacks that successfully broke the former version Grain v0.
- De Cannière. C. *et al.* discovered a slide property in the initialization phase of Grain v1, reduce half of the cost of exhaustive key search for a fixed IV.
- A related-key chosen IV attack has also been proposed by Lee, Y *et al.*

Introduction

- Grain v1, designed by Martin Hell, Thomas Johansson and Willi Meier, is a stream cipher for restricted hardware environments. It was selected into the final portfolio by the eSTREAM project.
- Grain v1 is immune to the correlation and distinguishing attacks that successfully broke the former version Grain v0.
- De Cannière. C. *et al.* discovered a slide property in the initialization phase of Grain v1, reduce half of the cost of exhaustive key search for a fixed IV.
- A related-key chosen IV attack has also been proposed by Lee, Y *et al.*
- The companion cipher, Grain-128 is designed in a similar way with low algrbraic degree feedback function, resulting in a dynamic cube attack on the full initialization rounds by Dinur *et al.*

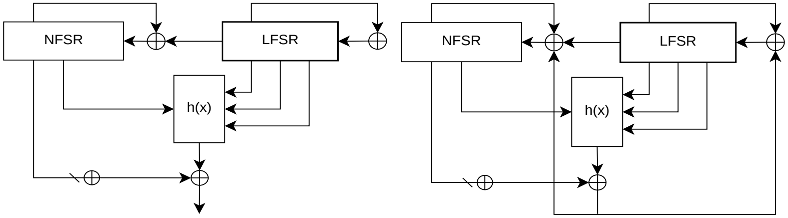
Introduction

- Grain v1, designed by Martin Hell, Thomas Johansson and Willi Meier, is a stream cipher for restricted hardware environments. It was selected into the final portfolio by the eSTREAM project.
- Grain v1 is immune to the correlation and distinguishing attacks that successfully broke the former version Grain v0.
- De Cannière. C. *et al.* discovered a slide property in the initialization phase of Grain v1, reduce half of the cost of exhaustive key search for a fixed IV.
- A related-key chosen IV attack has also been proposed by Lee, Y *et al.*
- The companion cipher, Grain-128 is designed in a similar way with low algrbraic degree feedback function, resulting in a dynamic cube attack on the full initialization rounds by Dinur *et al.*
- A new variant, Grain-128a with optional authentication was proposed by Ȧgren *et al.*

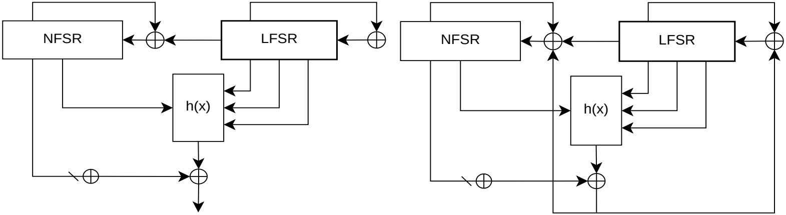
Outline

- Introduction
- Description of Grain v1
- Main idea & some key observations
- The general attack model: NCA-1.0
- NCA-2.0 & NCA-3.0
- Simulations
- Conclusions

Description of Grain v1

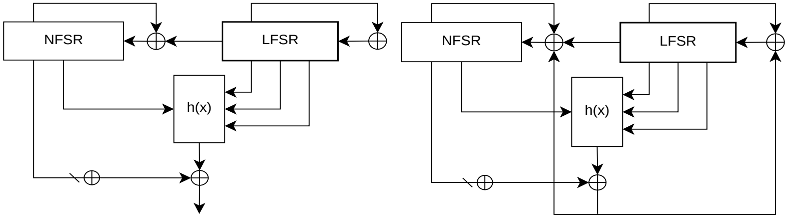


Description of Grain v1



- The non-linear filter function $h(x)$ is balanced and correlation immune of the first order, defined as:
  $h(x) = x_1 + x_4 + x_0x_3 + x_2x_3 + x_3x_4 + x_0x_1x_2 + x_0x_2x_3 + x_0x_2x_4 + x_1x_2x_4 + x_2x_3x_4.$

Description of Grain v1



- The non-linear filter function $h(x)$ is balanced and correlation immune of the first order, defined as:
  $h(x) = x_1 + x_4 + x_0x_3 + x_2x_3 + x_3x_4 + x_0x_1x_2 + x_0x_2x_3 + x_0x_2x_4 + x_1x_2x_4 + x_2x_3x_4.$
- The output function is taken as $z_i = \sum_{k\in\mathcal{A}} n_{i+k} + h(l_{i+3}, l_{i+25}, l_{i+46}, l_{i+64}, n_{i+63})$, where $\mathcal{A} = \{1, 2, 4, 10, 31, 43, 56\}$.

Outline

- Introduction
- Description of Grain v1
- Main idea & some key observations
- The general attack model: NCA-1.0
- NCA-2.0 & NCA-3.0
- Simulations
- Conclusions

Main idea

- In this paper, a new key recovery attack, called near collision attack is proposed, utilizing the compact NFSR-LFSR combined structure of Grain v1.

Main idea

- In this paper, a new key recovery attack, called near collision attack is proposed, utilizing the compact NFSR-LFSR combined structure of Grain v1.
- It is observed that the NFSR and LFSR are of length exactly 80-bit (with no redundance) and the LFSR updates independently in the keystream generation phase.

Main idea

- In this paper, a new key recovery attack, called near collision attack is proposed, utilizing the compact NFSR-LFSR combined structure of Grain v1.
- It is observed that the NFSR and LFSR are of length exactly 80-bit (with no redundance) and the LFSR updates independently in the keystream generation phase.
- It is observed that the LFSR state bits can be easily recovered, given the internal state difference at two different time instants.

Main idea

- In this paper, a new key recovery attack, called near collision attack is proposed, utilizing the compact NFSR-LFSR combined structure of Grain v1.
- It is observed that the NFSR and LFSR are of length exactly 80-bit (with no redundance) and the LFSR updates independently in the keystream generation phase.
- It is observed that the LFSR state bits can be easily recovered, given the internal state difference at two different time instants.
- It is observed that the distribution of the keystream segment differences is non-uniform, given a low Hamming weight internal state difference.

Main idea

- In this paper, a new key recovery attack, called near collision attack is proposed, utilizing the compact NFSR-LFSR combined structure of Grain v1.
- It is observed that the NFSR and LFSR are of length exactly 80-bit (with no redundance) and the LFSR updates independently in the keystream generation phase.
- It is observed that the LFSR state bits can be easily recovered, given the internal state difference at two different time instants.
- It is observed that the distribution of the keystream segment differences is non-uniform, given a low Hamming weight internal state difference.
- Three attacks has been proposed: NCA-1.0, NCA-2.0 combined with BSW sampling, NCA-3.0 utilizing the non-uniform distribution of the internal state differences for a fixed keystream difference.

Preliminaries

Definition

Two n-bit strings s, s' are d-near-collision, if $w_H(s \oplus s') \leq d$.

Similar to the birthday paradox, which states that two random subsets of a space with 2^n elements are expected to intersect when the product of their sizes exceeds 2^n, we present the following lemma of d-near-collision.

Lemma

Given two random subsets A, B of a space with 2^n *elements, then there exists a pair* (a, b) *with* $a \in A$ *and* $b \in B$ *that is an d-near-collision if*

$$|A| \cdot |B| \geq \frac{2^n}{V(n,d)} \tag{1}$$

holds, where $|A|$ *and* $|B|$ *are the size of A and B respectively.*

$V(n,d) = \sum_{i=0}^{d} \binom{n}{i}$.

Observation I-State recovery with known state difference

- Denote the LFSR state as $L^{t_1} = (l_0^{t_1}, l_1^{t_1}, ..., l_{79}^{t_1})$ at time t_1 and $L^{t_2} = (l_0^{t_2}, l_1^{t_2}, ..., l_{79}^{t_2})$ at time t_2 $(0 \leq t_1 < t_2)$. Then, we can derive

$$\begin{cases} l_0^{t_2} = c_0^0 l_0^{t_1} + c_1^0 l_1^{t_1} + ... + c_{79}^0 l_{79}^{t_1} \\ l_1^{t_2} = c_0^1 l_0^{t_1} + c_1^1 l_1^{t_1} + ... + c_{79}^1 l_{79}^{t_1} \\ \qquad \vdots \\ l_{79}^{t_2} = c_0^{79} l_0^{t_1} + c_1^{79} l_1^{t_1} + ... + c_{79}^{79} l_{79}^{t_1}, \end{cases}$$

Observation I-State recovery with known state difference

- Denote the LFSR state as $L^{t_1} = (l_0^{t_1}, l_1^{t_1}, ..., l_{79}^{t_1})$ at time t_1 and $L^{t_2} = (l_0^{t_2}, l_1^{t_2}, ..., l_{79}^{t_2})$ at time t_2 $(0 \leq t_1 < t_2)$. Then, we can derive

$$\begin{cases} l_0^{t_2} = c_0^0 l_0^{t_1} + c_1^0 l_1^{t_1} + ... + c_{79}^0 l_{79}^{t_1} \\ l_1^{t_2} = c_0^1 l_0^{t_1} + c_1^1 l_1^{t_1} + ... + c_{79}^1 l_{79}^{t_1} \\ \qquad\vdots \\ l_{79}^{t_2} = c_0^{79} l_0^{t_1} + c_1^{79} l_1^{t_1} + ... + c_{79}^{79} l_{79}^{t_1}, \end{cases}$$

- Suppose that we know the difference $\Delta L = (l_0^{t_1} \oplus l_0^{t_2}, ..., l_{79}^{t_1} \oplus l_{79}^{t_2}) = (\Delta l_0, \Delta l_1, ..., \Delta l_{79})$ with the time interval $\Delta t = t_2 - t_1$. Then,

$$\begin{cases} \Delta l_0 = l_0^{t_2} \oplus l_0^{t_1} = (c_0^0 + 1) l_0^{t_1} + x c_1^0 l_1^{t_1} + ... + c_{79}^0 l_{79}^{t_1} \\ \Delta l_1 = l_1^{t_2} \oplus l_1^{t_1} = c_0^1 l_0^{t_1} + (c_1^1 + 1) l_1^{t_1} + ... + c_{79}^1 l_{79}^{t_1} \\ \qquad\vdots \\ \Delta l_{79} = l_{79}^{t_2} \oplus l_{79}^{t_1} = c_0^{79} l_0^{t_1} + c_1^{79} l_1^{t_1} + ... + (c_{79}^{79} + 1) l_{79}^{t_1}. \end{cases}$$

Observation I-State recovery with known state difference

- Denote the LFSR state as $L^{t_1} = (l_0^{t_1}, l_1^{t_1}, ..., l_{79}^{t_1})$ at time t_1 and $L^{t_2} = (l_0^{t_2}, l_1^{t_2}, ..., l_{79}^{t_2})$ at time t_2 $(0 \leq t_1 < t_2)$. Then, we can derive

$$\begin{cases} l_0^{t_2} = c_0^0 l_0^{t_1} + c_1^0 l_1^{t_1} + ... + c_{79}^0 l_{79}^{t_1} \\ l_1^{t_2} = c_0^1 l_0^{t_1} + c_1^1 l_1^{t_1} + ... + c_{79}^1 l_{79}^{t_1} \\ \qquad \vdots \\ l_{79}^{t_2} = c_0^{79} l_0^{t_1} + c_1^{79} l_1^{t_1} + ... + c_{79}^{79} l_{79}^{t_1}, \end{cases}$$

- Suppose that we know the difference $\Delta L = (l_0^{t_1} \oplus l_0^{t_2}, ..., l_{79}^{t_1} \oplus l_{79}^{t_2}) = (\Delta l_0, \Delta l_1, ..., \Delta l_{79})$ with the time interval $\Delta t = t_2 - t_1$. Then,

$$\begin{cases} \Delta l_0 = l_0^{t_2} \oplus l_0^{t_1} = (c_0^0 + 1) l_0^{t_1} + x c_1^0 l_1^{t_1} + ... + c_{79}^0 l_{79}^{t_1} \\ \Delta l_1 = l_1^{t_2} \oplus l_1^{t_1} = c_0^1 l_0^{t_1} + (c_1^1 + 1) l_1^{t_1} + ... + c_{79}^1 l_{79}^{t_1} \\ \qquad \vdots \\ \Delta l_{79} = l_{79}^{t_2} \oplus l_{79}^{t_1} = c_0^{79} l_0^{t_1} + c_1^{79} l_1^{t_1} + ... + (c_{79}^{79} + 1) l_{79}^{t_1}. \end{cases}$$

- The next step is to recover the NFSR state at t_1 and t_2, the time complexity is bounded by $2^{20.3}$ cipher ticks.

Observation II-the Distribution of the KSD

- The distribution of keystream segment differences (KSDs) is biased, given a specific internal state differential (ISD).

Observation II-the Distribution of the KSD

- The distribution of keystream segment differences (KSDs) is biased, given a specific internal state differential (ISD).

Table: The distribution of KSDs

ISD	KSD	Proportion	ISD	KSD	Proportion
Δs_1	0xa120	49.4%	Δs_4	0x0000	52.0%
	0xe120	50.6%		0x0080	48.0%
Δs_2	0x0000	12.9%	Δs_3	0x0001	13.2%
	0x0001	13.8%		0x0201	12.1%
	0x2000	38.3%		0x0801	37.2%
	0x2001	35.1%		0x0a01	37.5%

The distribution of KSDs.
$l = 16, d = 4$.

Observation II-the Distribution of the KSD

- The distribution of keystream segment differences (KSDs) is biased, given a specific internal state differential (ISD).

Table: The distribution of KSDs

ISD	KSD	Proportion	ISD	KSD	Proportion
Δs_1	0xa120	49.4%	Δs_4	0x0000	52.0%
	0xe120	50.6%		0x0080	48.0%
Δs_2	0x0000	12.9%	Δs_3	0x0001	13.2%
	0x0001	13.8%		0x0201	12.1%
	0x2000	38.3%		0x0801	37.2%
	0x2001	35.1%		0x0a01	37.5%

The distribution of KSDs.
$l = 16, d = 4$.

- The results also show that there exists some impossible differences for most of (d, l) pairs.

Observation III-Complexity of the brute force attack

- The complexity of the brute force attack is higher than 2^{80} ticks and such an attack can only be mounted for each fixed IV.

Observation III-Complexity of the brute force attack

- The complexity of the brute force attack is higher than 2^{80} ticks and such an attack can only be mounted for each fixed IV.
- For each enumerated k_i, $1 \leq i \leq 2^{80} - 1$, the attacker first needs to proceed the initialization phase which needs 160 ticks.

Observation III-Complexity of the brute force attack

- The complexity of the brute force attack is higher than 2^{80} ticks and such an attack can only be mounted for each fixed IV.
- For each enumerated k_i, $1 \leq i \leq 2^{80} - 1$, the attacker first needs to proceed the initialization phase which needs 160 ticks.
- If each keystream bit is treated as a random variable, then for each k_i, the probability that the attacker need to generate l ($1 \leq l \leq 80$) bits keystream is 1 for $l = 1$ and $2^{-(l-1)}$ for $l > 1$

Observation III-Complexity of the brute force attack

- The complexity of the brute force attack is higher than 2^{80} ticks and such an attack can only be mounted for each fixed IV.
- For each enumerated k_i, $1 \leq i \leq 2^{80} - 1$, the attacker first needs to proceed the initialization phase which needs 160 ticks.
- If each keystream bit is treated as a random variable, then for each k_i, the probability that the attacker need to generate l $(1 \leq l \leq 80)$ bits keystream is 1 for $l = 1$ and $2^{-(l-1)}$ for $l > 1$
- Let N_w be the expected number of bits needed to generate for each enumerated key, which is $N_w = \sum_{l=1}^{80} l \cdot P_l = \sum_{l=1}^{80} l \cdot 2^{-(l-1)} \approx 4$. Then, the total time complexity is $(2^{80} - 1) \cdot (160 + 4) \approx 2^{87.4}$ cipher ticks.

Outline

- Introduction
- Description of Grain v1
- Main idea & some key observations
- The general attack model: NCA-1.0
- NCA-2.0 & NCA-3.0
- Simulations
- Conclusions

The General Attack Model

- Main concern: Identify the correct d-near-collision ISD.

The General Attack Model

- Main concern: Identify the correct d-near-collision ISD.
- Off-line stage: some well structured differential tables are pre-computed. The table structure can be illustrated as follows.

The General Attack Model

- Main concern: Identify the correct d-near-collision ISD.
- Off-line stage: some well structured differential tables are pre-computed. The table structure can be illustrated as follows.

$$\text{table-0x0000}\begin{cases}\Delta s_4 & 52.0\% \\ \Delta s_2 & 12.9\% \\ & \vdots\end{cases} \qquad \text{table-0x0001}\begin{cases}\Delta s_2 & 13.8\% \\ \Delta s_3 & 13.2\% \\ & \vdots\end{cases} \qquad \ldots$$

$$\text{table-0x0080}\begin{cases}\Delta s_4 & 48.0\% \\ & \vdots\end{cases} \qquad \ldots$$

- The total number of tables is $Q(n, d, l)$ and the average number of rows in each table is $R(n, d, l)$.

The General Attack Model

- Main concern: Identify the correct d-near-collision ISD.
- Off-line stage: some well structured differential tables are pre-computed. The table structure can be illustrated as follows.

$$\text{table-0x0000}\left\{\begin{array}{ll}\Delta s_4 & 52.0\% \\ \Delta s_2 & 12.9\% \\ & \vdots \end{array}\right. \qquad \text{table-0x0001}\left\{\begin{array}{ll}\Delta s_2 & 13.8\% \\ \Delta s_3 & 13.2\% \\ & \vdots \end{array}\right. \qquad \cdots$$

$$\text{table-0x0080}\left\{\begin{array}{ll}\Delta s_4 & 48.0\% \\ & \vdots \end{array}\right. \qquad \cdots$$

- The total number of tables is $Q(n, d, l)$ and the average number of rows in each table is $R(n, d, l)$.
- Due to the non-uniform distribution of the KSDs for a fixed ISD, we only consider at most 100 KSDs whose proportions are the first 100 largest among all the KSDs. Hence $R(n, d, l)$ is upper bounded by $100 \cdot V(n, d)/Q(n, d, l)$.

The General Attack Model

- Main concern: Identify the correct d-near-collision ISD.
- Off-line stage: some well structured differential tables are pre-computed. The table structure can be illustrated as follows.

$$\text{table-0x0000}\left\{\begin{array}{ll}\Delta s_4 & 52.0\% \\ \Delta s_2 & 12.9\% \\ & \vdots\end{array}\right. \qquad \text{table-0x0001}\left\{\begin{array}{ll}\Delta s_2 & 13.8\% \\ \Delta s_3 & 13.2\% \\ & \vdots\end{array}\right. \qquad \ldots$$

$$\text{table-0x0080}\left\{\begin{array}{ll}\Delta s_4 & 48.0\% \\ & \vdots\end{array}\right. \qquad \ldots$$

- The total number of tables is $Q(n,d,l)$ and the average number of rows in each table is $R(n,d,l)$.
- Due to the non-uniform distribution of the KSDs for a fixed ISD, we only consider at most 100 KSDs whose proportions are the first 100 largest among all the KSDs. Hence $R(n,d,l)$ is upper bounded by $100 \cdot V(n,d)/Q(n,d,l)$.

On-line Stage

- Now we discuss how to obtain the ISD by utilizing the pre-computed tables and the truncated keystreams.

On-line Stage

- Now we discuss how to obtain the ISD by utilizing the pre-computed tables and the truncated keystreams.
    1. Randomly collect two keystream segments sets A and B, satisfying $|A| \cdot |B| \geq 2^n / V(n, d)$.

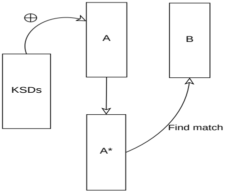
On-line Stage

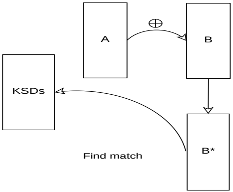
- Now we discuss how to obtain the ISD by utilizing the pre-computed tables and the truncated keystreams.
  1. Randomly collect two keystream segments sets A and B, satisfying $|A| \cdot |B| \geq 2^n / V(n, d)$.
  2. Sort A and B with respect to the value of the first l bits. Divide them into m different groups $G_1^A, G_2^A, ..., G_m^A$ and $G_1^B, G_2^B, ..., G_m^B$ respectively.

On-line Stage

- Now we discuss how to obtain the ISD by utilizing the pre-computed tables and the truncated keystreams.
  1. Randomly collect two keystream segments sets A and B, satisfying $|A| \cdot |B| \geq 2^n / V(n, d)$.
  2. Sort A and B with respect to the value of the first l bits. Divide them into m different groups $G_1^A, G_2^A, ..., G_m^A$ and $G_1^B, G_2^B, ..., G_m^B$ respectively.
  3. Identify the candidate (s_i^A, s_j^B) pairs that is d-near-collision. Two strategies:

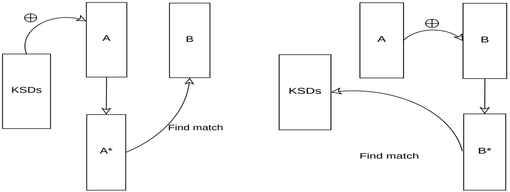
- Now we discuss how to obtain the ISD by utilizing the pre-computed tables and the truncated keystreams.
  1. Randomly collect two keystream segments sets A and B, satisfying $|A| \cdot |B| \geq 2^n/V(n,d)$.
  2. Sort A and B with respect to the value of the first l bits. Divide them into m different groups $G_1^A, G_2^A, \ldots, G_m^A$ and $G_1^B, G_2^B, \ldots, G_m^B$ respectively.
  3. Identify the candidate (s_i^A, s_j^B) pairs that is d-near-collision. Two strategies:





On-line Stage

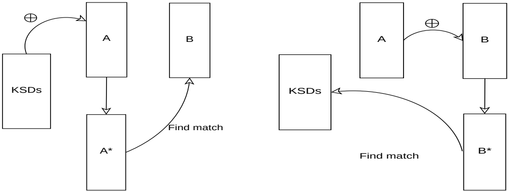
- Now we discuss how to obtain the ISD by utilizing the pre-computed tables and the truncated keystreams.
  1. Randomly collect two keystream segments sets A and B, satisfying $|A| \cdot |B| \geq 2^n/V(n,d)$.
  2. Sort A and B with respect to the value of the first l bits. Divide them into m different groups $G_1^A, G_2^A, \ldots, G_m^A$ and $G_1^B, G_2^B, \ldots, G_m^B$ respectively.
  3. Identify the candidate (s_i^A, s_j^B) pairs that is d-near-collision. Two strategies:



  4. This step is to filter out pseudo-collisions and find the real ISD by utilizing observation I.

On-line Stage

- Now we discuss how to obtain the ISD by utilizing the pre-computed tables and the truncated keystreams.
    1. Randomly collect two keystream segments sets A and B, satisfying $|A| \cdot |B| \geq 2^n / V(n, d)$.
    2. Sort A and B with respect to the value of the first l bits. Divide them into m different groups $G_1^A, G_2^A, \ldots, G_m^A$ and $G_1^B, G_2^B, \ldots, G_m^B$ respectively.
    3. Identify the candidate (s_i^A, s_j^B) pairs that is d-near-collision. Two strategies:



    4. This step is to filter out pseudo-collisions and find the real ISD by utilizing observation I.

Complexity Analysis of NCA-1.0

- Pre-computation time: $P = 2 \cdot N \cdot V(n, d) \cdot l$. The data complexity is $D = |A| + |B|$ $\hat{l}$-bit keystream segments and the memory requirement is $M = M_1 + M_2 = V(n, d) \cdot 2^{6.6} + |A| + |B|$ entries.

Complexity Analysis of NCA-1.0

- Pre-computation time: $P = 2 \cdot N \cdot V(n,d) \cdot l$. The data complexity is $D = |A| + |B|$ $\hat{l}$-bit keystream segments and the memory requirement is $M = M_1 + M_2 = V(n,d) \cdot 2^{6.6} + |A| + |B|$ entries.
- Step 2: $T_1 = (|A| \cdot \log |A| + |B| \cdot \log |B|)/\Omega$ cipher ticks.

Complexity Analysis of NCA-1.0

- Pre-computation time: $P = 2 \cdot N \cdot V(n,d) \cdot l$. The data complexity is $D = |A| + |B|$ $\hat{l}$-bit keystream segments and the memory requirement is $M = M_1 + M_2 = V(n,d) \cdot 2^{6.6} + |A| + |B|$ entries.
- Step 2: $T_1 = (|A| \cdot \log|A| + |B| \cdot \log|B|)/\Omega$ cipher ticks.
- Step 3: $T_2 = min\{Q(n,d,l) \cdot m \cdot \log m/\Omega,\ m^2 \cdot \log Q(n,d,l)/\Omega\}$ ticks.

Complexity Analysis of NCA-1.0

- Pre-computation time: $P = 2 \cdot N \cdot V(n,d) \cdot l$. The data complexity is $D = |A| + |B|$ $\hat{l}$-bit keystream segments and the memory requirement is $M = M_1 + M_2 = V(n,d) \cdot 2^{6.6} + |A| + |B|$ entries.
- Step 2: $T_1 = (|A| \cdot \log|A| + |B| \cdot \log|B|)/\Omega$ cipher ticks.
- Step 3: $T_2 = min\{Q(n,d,l) \cdot m \cdot \log m/\Omega,\ m^2 \cdot \log Q(n,d,l)/\Omega\}$ ticks.
- Step 4: $T_3 = |A| \cdot |B| \cdot V(n,d) \cdot 2^{6.6} \cdot T_K/Q(n,d,l)$.

Table: The attack complexity with various l

l	P	T_1	T_2	T_3	T
102	$2^{95.7}$	$2^{40.9}$	$2^{85.8}$	$2^{86.4}$	$2^{86.4}$
104	$2^{95.7}$	$2^{40.9}$	$2^{85.9}$	$2^{84.4}$	$2^{85.9}$
106	$2^{95.7}$	$2^{40.9}$	$2^{85.9}$	$2^{72.4}$	$2^{85.9}$

$n = 160,\ d = 16,\ D = 2^{45.8},\ M = 2^{78.6}$

Strategy II is chosen in Step 3.

Complexity Analysis of NCA-1.0

- Pre-computation time: $P = 2 \cdot N \cdot V(n,d) \cdot l$. The data complexity is $D = |A| + |B|$ $\hat{l}$-bit keystream segments and the memory requirement is $M = M_1 + M_2 = V(n,d) \cdot 2^{6.6} + |A| + |B|$ entries.
- Step 2: $T_1 = (|A| \cdot \log|A| + |B| \cdot \log|B|)/\Omega$ cipher ticks.
- Step 3: $T_2 = min\{Q(n,d,l) \cdot m \cdot \log m/\Omega,\ m^2 \cdot \log Q(n,d,l)/\Omega\}$ ticks.
- Step 4: $T_3 = |A| \cdot |B| \cdot V(n,d) \cdot 2^{6.6} \cdot T_K/Q(n,d,l)$.

Table: The attack complexity with various l

l	P	T_1	T_2	T_3	T
102	$2^{95.7}$	$2^{40.9}$	$2^{85.8}$	$2^{86.4}$	$2^{86.4}$
104	$2^{95.7}$	$2^{40.9}$	$2^{85.9}$	$2^{84.4}$	$2^{85.9}$
106	$2^{95.7}$	$2^{40.9}$	$2^{85.9}$	$2^{72.4}$	$2^{85.9}$

$n = 160,\ d = 16,\ D = 2^{45.8},\ M = 2^{78.6}$

Strategy II is chosen in Step 3.

- We name this basic attack as NCA-1.0. The pre-computation complexity $P = 2^{95.7}$ exceeds the brute force attack complexity of $2^{87.4}$.

Outline

- Introduction
- Description of Grain v1
- Main idea & some key observations
- The general attack model: NCA-1.0
- NCA-2.0 & NCA-3.0
- Simulations
- Conclusions

NCA-2.0

- The first improvement is designated by combining the sampling resistance property of Grain with NCA-1.0.

NCA-2.0

- The first improvement is designated by combining the sampling resistance property of Grain with NCA-1.0.

Lemma

Given the value of 139 *particular state bits of Grain and the first* 21 *keystream bits produced from that state, another* 21 *internal state bits can be deduced directly.*

NCA-2.0

- The first improvement is designated by combining the sampling resistance property of Grain with NCA-1.0.

Lemma

Given the value of 139 *particular state bits of Grain and the first* 21 *keystream bits produced from that state, another* 21 *internal state bits can be deduced directly.*

- The sampling resistance of Grain is $R = 2^{-21}$. Thus we define a restricted one-way function $\tau : \{0,1\}^{139} \to \{0,1\}^{139}$ by choosing a prefix of 0^{21}.

NCA-2.0

- The first improvement is designated by combining the sampling resistance property of Grain with NCA-1.0.

Lemma

Given the value of 139 *particular state bits of Grain and the first* 21 *keystream bits produced from that state, another* 21 *internal state bits can be deduced directly.*

- The sampling resistance of Grain is $R = 2^{-21}$. Thus we define a restricted one-way function $\tau : \{0,1\}^{139} \rightarrow \{0,1\}^{139}$ by choosing a prefix of 0^{21}.
    1. For each 139-bit input value x, the remaining 21-bit internal state can be determined by Lemma 2 and the prefix of 0^{21}.

NCA-2.0

- The first improvement is designated by combining the sampling resistance property of Grain with NCA-1.0.

Lemma

Given the value of 139 particular state bits of Grain and the first 21 keystream bits produced from that state, another 21 internal state bits can be deduced directly.

- The sampling resistance of Grain is $R = 2^{-21}$. Thus we define a restricted one-way function $\tau : \{0,1\}^{139} \to \{0,1\}^{139}$ by choosing a prefix of 0^{21}.
  1. For each 139-bit input value x, the remaining 21-bit internal state can be determined by Lemma 2 and the prefix of 0^{21}.
  2. Run the cipher forward for 160 ticks, generate an 160-bit segment $0^{21}||y$, output y.

NCA-2.0

- The first improvement is designated by combining the sampling resistance property of Grain with NCA-1.0.

Lemma

Given the value of 139 *particular state bits of Grain and the first* 21 *keystream bits produced from that state, another* 21 *internal state bits can be deduced directly.*

- The sampling resistance of Grain is $R = 2^{-21}$. Thus we define a restricted one-way function $\tau : \{0,1\}^{139} \rightarrow \{0,1\}^{139}$ by choosing a prefix of 0^{21}.
    1. For each 139-bit input value x, the remaining 21-bit internal state can be determined by Lemma 2 and the prefix of 0^{21}.
    2. Run the cipher forward for 160 ticks, generate an 160-bit segment $0^{21}||y$, output y.
- Now, the searching space is reduced to a special subset of the internal states.

Complexity Analysis of NCA-2.0

- Now, the goal is to recover the $n^* = 139$ bits ISD which contains 60 NFSR state bits and 79 LFSR state bits, instead of the $n = 160$ bits ISD.

Complexity Analysis of NCA-2.0

- Now, the goal is to recover the $n^* = 139$ bits ISD which contains 60 NFSR state bits and 79 LFSR state bits, instead of the $n = 160$ bits ISD.
- We need to collect those keystream segments with the prefix pattern 0^{21}, the data complexity is $D = (|A| + |B|) \cdot 2^{21}$.

Complexity Analysis of NCA-2.0

- Now, the goal is to recover the $n^* = 139$ bits ISD which contains 60 NFSR state bits and 79 LFSR state bits, instead of the $n = 160$ bits ISD.
- We need to collect those keystream segments with the prefix pattern 0^{21}, the data complexity is $D = (|A| + |B|) \cdot 2^{21}$.

Table: The attack complexities with various l based on sampling resistance

l	P^*	T_1	T_2	T_3	T
92	$2^{83.4}$	$2^{35.9}$	$2^{76.1}$	$2^{75.4}$	$2^{76.1}$
94	$2^{83.4}$	$2^{35.9}$	$2^{76.2}$	$2^{73.4}$	$2^{76.2}$
96	$2^{83.4}$	$2^{35.9}$	$2^{76.2}$	$2^{71.4}$	$2^{76.2}$

$n^* = 139,\ d = 13,\ D = 2^{62},\ M = 2^{65.9}$.

Strategy II is chosen in Step 3.

Complexity Analysis of NCA-2.0

- Now, the goal is to recover the $n^* = 139$ bits ISD which contains 60 NFSR state bits and 79 LFSR state bits, instead of the $n = 160$ bits ISD.
- We need to collect those keystream segments with the prefix pattern 0^{21}, the data complexity is $D = (|A| + |B|) \cdot 2^{21}$.

Table: The attack complexities with various l based on sampling resistance

l	P^*	T_1	T_2	T_3	T
92	$2^{83.4}$	$2^{35.9}$	$2^{76.1}$	$2^{75.4}$	$2^{76.1}$
94	$2^{83.4}$	$2^{35.9}$	$2^{76.2}$	$2^{73.4}$	$2^{76.2}$
96	$2^{83.4}$	$2^{35.9}$	$2^{76.2}$	$2^{71.4}$	$2^{76.2}$

$n^* = 139,\ d = 13,\ D = 2^{62},\ M = 2^{65.9}$.
Strategy II is chosen in Step 3.

- Compared to NCA-1.0, our improved attack reduces P by a factor of $2^{12.3}$ and it saves 10-bit storage for each entry in A and B.

Complexity Analysis of NCA-2.0

- Now, the goal is to recover the $n^* = 139$ bits ISD which contains 60 NFSR state bits and 79 LFSR state bits, instead of the $n = 160$ bits ISD.
- We need to collect those keystream segments with the prefix pattern 0^{21}, the data complexity is $D = (|A| + |B|) \cdot 2^{21}$.

Table: The attack complexities with various l based on sampling resistance

l	P^*	T_1	T_2	T_3	T
92	$2^{83.4}$	$2^{35.9}$	$2^{76.1}$	$2^{75.4}$	$2^{76.1}$
94	$2^{83.4}$	$2^{35.9}$	$2^{76.2}$	$2^{73.4}$	$2^{76.2}$
96	$2^{83.4}$	$2^{35.9}$	$2^{76.2}$	$2^{71.4}$	$2^{76.2}$

$n^* = 139,\ d = 13,\ D = 2^{62},\ M = 2^{65.9}$.
Strategy II is chosen in Step 3.

- Compared to NCA-1.0, our improved attack reduces P by a factor of $2^{12.3}$ and it saves 10-bit storage for each entry in A and B.
- All the complexities are under the brute force attack complexity of $2^{87.4}$. We name this combined attack as NCA-2.0.

NCA-3.0

- The second improvement is based on NCA-2.0 by utilizing the non-uniform distribution of KSDs among all the tables. Some observations (Example in Section 3.2):

NCA-3.0

- The second improvement is based on NCA-2.0 by utilizing the non-uniform distribution of KSDs among all the tables. Some observations (Example in Section 3.2):
    1. Some tables like table-0x0000, table-0x0008, table-0x0004 contains more rows than those like table-0x0012 and table-0x0048.

NCA-3.0

- The second improvement is based on NCA-2.0 by utilizing the non-uniform distribution of KSDs among all the tables. Some observations (Example in Section 3.2):
    1. Some tables like table-0x0000, table-0x0008, table-0x0004 contains more rows than those like table-0x0012 and table-0x0048.
    2. Table-0x0000 contains the most rows among all the tables.

NCA-3.0

- The second improvement is based on NCA-2.0 by utilizing the non-uniform distribution of KSDs among all the tables. Some observations (Example in Section 3.2):
  1. Some tables like table-0x0000, table-0x0008, table-0x0004 contains more rows than those like table-0x0012 and table-0x0048.
  2. Table-0x0000 contains the most rows among all the tables.
  3. Most tables like table-0xfe00, table-0xfd68 and table-0xfad1 only contain a single row.

NCA-3.0

- The second improvement is based on NCA-2.0 by utilizing the non-uniform distribution of KSDs among all the tables. Some observations (Example in Section 3.2):
    1. Some tables like table-0x0000, table-0x0008, table-0x0004 contains more rows than those like table-0x0012 and table-0x0048.
    2. Table-0x0000 contains the most rows among all the tables.
    3. Most tables like table-0xfe00, table-0xfd68 and table-0xfad1 only contain a single row.
    4. The tables with low Hamming weight indexes satisfying $w_H(\text{KSD}) \leq 3$ (special tables) contain about 80% of all the $V(n,d)$ different ISDs.

NCA-3.0

- The second improvement is based on NCA-2.0 by utilizing the non-uniform distribution of KSDs among all the tables. Some observations (Example in Section 3.2):
    1. Some tables like table-0x0000, table-0x0008, table-0x0004 contains more rows than those like table-0x0012 and table-0x0048.
    2. Table-0x0000 contains the most rows among all the tables.
    3. Most tables like table-0xfe00, table-0xfd68 and table-0xfad1 only contain a single row.
    4. The tables with low Hamming weight indexes satisfying $w_H(\text{KSD}) \leq 3$ (special tables) contain about 80% of all the $V(n, d)$ different ISDs.

Assumption

On average, the special tables can cover 50% of all the $V(n^*, d)$ different ISDs, when d and l becomes larger.

NCA-3.0

- The second improvement is based on NCA-2.0 by utilizing the non-uniform distribution of KSDs among all the tables. Some observations (Example in Section 3.2):
    1. Some tables like table-0x0000, table-0x0008, table-0x0004 contains more rows than those like table-0x0012 and table-0x0048.
    2. Table-0x0000 contains the most rows among all the tables.
    3. Most tables like table-0xfe00, table-0xfd68 and table-0xfad1 only contain a single row.
    4. The tables with low Hamming weight indexes satisfying $w_H(\text{KSD}) \leq 3$ (special tables) contain about 80% of all the $V(n, d)$ different ISDs.

Assumption

On average, the special tables can cover 50% *of all the* $V(n^*, d)$ *different ISDs, when d and l becomes larger.*

- The assumption indicates that in the off-line stage, we only need to construct those special tables.

Complexity Analysis of NCA-3.0

- All the complexities remain unchanged except $T_2 = min\{l^3 \cdot m \cdot \log m,\ m^2 \cdot \log l^3\}$.

Complexity Analysis of NCA-3.0

- All the complexities remain unchanged except $T_2 = min\{l^3 \cdot m \cdot \log m,\ m^2 \cdot \log l^3\}$.

Table: The attack complexity on Grain with various l based on special tables

l	P^*	T_1	T_2	T_3	T
92	$2^{73.1}$	$2^{41.9}$	$2^{60.5}$	$2^{75.4}$	$2^{75.4}$
94	$2^{73.1}$	$2^{41.9}$	$2^{60.6}$	$2^{73.4}$	$2^{73.4}$
96	$2^{73.1}$	$2^{41.9}$	$2^{60.7}$	$2^{71.4}$	$2^{71.4}$

$n^* = 139,\ d = 10,\ M = 2^{62.8}$ bits, $D = 2^{67.8}$ bits keystream. Strategy I is chosen in Step 3.

Complexity Analysis of NCA-3.0

- All the complexities remain unchanged except $T_2 = min\{l^3 \cdot m \cdot \log m,\ m^2 \cdot \log l^3\}$.

Table: The attack complexity on Grain with various l based on special tables

l	P^*	T_1	T_2	T_3	T
92	$2^{73.1}$	$2^{41.9}$	$2^{60.5}$	$2^{75.4}$	$2^{75.4}$
94	$2^{73.1}$	$2^{41.9}$	$2^{60.6}$	$2^{73.4}$	$2^{73.4}$
96	$2^{73.1}$	$2^{41.9}$	$2^{60.7}$	$2^{71.4}$	$2^{71.4}$

$n^* = 139,\ d = 10,\ M = 2^{62.8}$ bits, $D = 2^{67.8}$ bits keystream. Strategy I is chosen in Step 3.

- We can obtain an attack of $T = 2^{71.4}$, $M = 2^{62.8}$ and $D = 2^{67.8}$ with the pre-computation complexity $P = 2^{73.1}$. We name this enhanced attack as NCA-3.0.

Outline

- Introduction
- Description of Grain v1
- Main idea & some key observations
- The general attack model: NCA-1.0
- NCA-2.0 & NCA-3.0
- Simulations
- Conclusions

Simulation and Results-Reduced Version

- The reduced version of Grain v1 cipher consists of an LFSR of 32 bits and an NFSR of 32 bits. The update functions of LFSR and NFSR are designed in a similar way as full version.

Simulation and Results-Reduced Version

- The reduced version of Grain v1 cipher consists of an LFSR of 32 bits and an NFSR of 32 bits. The update functions of LFSR and NFSR are designed in a similar way as full version.
- LFSR update function: $l'_{i+32} = l'_{i+30} + l'_{i+25} + l'_{i+16} + l'_i$. NFSR update function:

$$
\begin{aligned}
n'_{i+32} &= l'_i + n'_{i+25} + n'_{i+23} + n'_{i+15} + n'_{i+8} + n'_i + n'_{i+25}n'_{i+23} + n'_{i+15}n'_{i+8} \\
&+ n'_{i+25}n'_{i+23}n'_{i+15} + n'_{i+23}n'_{i+15}n'_{i+8} + n'_{i+25}n'_{i+23}n'_{i+15}n'_{i+8}.
\end{aligned}
$$

Simulation and Results-Reduced Version

- The reduced version of Grain v1 cipher consists of an LFSR of 32 bits and an NFSR of 32 bits. The update functions of LFSR and NFSR are designed in a similar way as full version.
- LFSR update function: $l'_{i+32} = l'_{i+30} + l'_{i+25} + l'_{i+16} + l'_i$. NFSR update function:

$$\begin{aligned} n'_{i+32} &= l'_i + n'_{i+25} + n'_{i+23} + n'_{i+15} + n'_{i+8} + n'_i + n'_{i+25}n'_{i+23} + n'_{i+15}n'_{i+8} \\ &+ n'_{i+25}n'_{i+23}n'_{i+15} + n'_{i+23}n'_{i+15}n'_{i+8} + n'_{i+25}n'_{i+23}n'_{i+15}n'_{i+8}. \end{aligned}$$

- The output function as $z'_i = \sum_{k\in\mathcal{A}'} n'_{i+k} + h(l'_{i+3}, l'_{i+11}, l'_{i+21}, l'_{i+25}, n'_{i+24})$, where $\mathcal{A} = \{1, 4, 10, 21\}$.

Simulation and Results-Reduced Version

- The reduced version of Grain v1 cipher consists of an LFSR of 32 bits and an NFSR of 32 bits. The update functions of LFSR and NFSR are designed in a similar way as full version.
- LFSR update function: $l'_{i+32} = l'_{i+30} + l'_{i+25} + l'_{i+16} + l'_i$. NFSR update function:

$$\begin{aligned} n'_{i+32} &= l'_i + n'_{i+25} + n'_{i+23} + n'_{i+15} + n'_{i+8} + n'_i + n'_{i+25}n'_{i+23} + n'_{i+15}n'_{i+8} \\ &+ n'_{i+25}n'_{i+23}n'_{i+15} + n'_{i+23}n'_{i+15}n'_{i+8} + n'_{i+25}n'_{i+23}n'_{i+15}n'_{i+8}. \end{aligned}$$

- The output function as $z'_i = \sum_{k\in\mathcal{A}'} n'_{i+k} + h(l'_{i+3}, l'_{i+11}, l'_{i+21}, l'_{i+25}, n'_{i+24})$, where $\mathcal{A} = \{1, 4, 10, 21\}$.
- Given the value of 53 particular state bits (including 32 bits LFSR and 21 bits NFSR) and the first 11 keystream bits, another 11 internal state bits can be deduced directly. Then the sampling resistance is $R' = 2^{-11}$.

Verification of Assumption 1

- We randomly chose 10^4 ISDs with Hamming weight $d \leq 4$ and generate their corresponding KSDs with the proportions. For each ISD, N random internal states were generated to determine the projection from ISD to KSD.

Verification of Assumption 1

- We randomly chose 10^4 ISDs with Hamming weight $d \leq 4$ and generate their corresponding KSDs with the proportions. For each ISD, N random internal states were generated to determine the projection from ISD to KSD.
- Only those KSDs satisfying $w_H(\text{KSD}) \leq 3$ will be recorded and their corresponding ISDs will be stored in a text file named with KSD.

Verification of Assumption 1

- We randomly chose 10^4 ISDs with Hamming weight $d \leq 4$ and generate their corresponding KSDs with the proportions. For each ISD, N random internal states were generated to determine the projection from ISD to KSD.
- Only those KSDs satisfying $w_H(\mathrm{KSD}) \leq 3$ will be recorded and their corresponding ISDs will be stored in a text file named with KSD.
- Similar to the process of the off-line stage, we only consider at most η KSDs whose proportions are the first η largest among all the KSDs. Finally, we count the number of different ISDs in these special tables.

Verification of Assumption 1

- We randomly chose 10^4 ISDs with Hamming weight $d \leq 4$ and generate their corresponding KSDs with the proportions. For each ISD, N random internal states were generated to determine the projection from ISD to KSD.
- Only those KSDs satisfying $w_H(\text{KSD}) \leq 3$ will be recorded and their corresponding ISDs will be stored in a text file named with KSD.
- Similar to the process of the off-line stage, we only consider at most η KSDs whose proportions are the first η largest among all the KSDs. Finally, we count the number of different ISDs in these special tables.

Table: Verification of Assumption 1

η	l	No. of ISDs	Proportion
50	24	9842	98.4%
1000	24	9851	98.5%
50	32	9202	92.0%
1000	32	9153	91.5%

$n = 53, \ d = 4.$

Verification of Assumption 1

- We randomly chose 10^4 ISDs with Hamming weight $d \leq 4$ and generate their corresponding KSDs with the proportions. For each ISD, N random internal states were generated to determine the projection from ISD to KSD.
- Only those KSDs satisfying $w_H(\mathrm{KSD}) \leq 3$ will be recorded and their corresponding ISDs will be stored in a text file named with KSD.
- Similar to the process of the off-line stage, we only consider at most η KSDs whose proportions are the first η largest among all the KSDs. Finally, we count the number of different ISDs in these special tables.

Table: Verification of Assumption 1

η	l	No. of ISDs	Proportion
50	24	9842	98.4%
1000	24	9851	98.5%
50	32	9202	92.0%
1000	32	9153	91.5%

$n = 53,\ d = 4.$

Simulations

- In the off-line stage, we set $\eta = 50$, $N = 2^{12}$ and $d = 4$.

Simulations

- In the off-line stage, we set $\eta = 50$, $N = 2^{12}$ and $d = 4$.

Table: Theoretical complexity on reduced version of Grain

Attack	l	P	D	M	T
NCA-2.0	24	$2^{36.3}$	$2^{29.2}$	$2^{23.9}$	$2^{36.2}$
NCA-3.0	24	$2^{36.3}$	$2^{29.2}$	$2^{23.9}$	$2^{36.2}$
NCA-2.0	32	$2^{36.7}$	$2^{29.2}$	$2^{23.9}$	$2^{31.4}$
NCA-3.0	32	$2^{36.7}$	$2^{29.2}$	$2^{23.9}$	$2^{28.2}$

$\eta = 50,\ N = 2^{12},\ d = 4.$

Simulations

- In the off-line stage, we set $\eta = 50$, $N = 2^{12}$ and $d = 4$.

Table: Theoretical complexity on reduced version of Grain

Attack	l	P	D	M	T
NCA-2.0	24	$2^{36.3}$	$2^{29.2}$	$2^{23.9}$	$2^{36.2}$
NCA-3.0	24	$2^{36.3}$	$2^{29.2}$	$2^{23.9}$	$2^{36.2}$
NCA-2.0	32	$2^{36.7}$	$2^{29.2}$	$2^{23.9}$	$2^{31.4}$
NCA-3.0	32	$2^{36.7}$	$2^{29.2}$	$2^{23.9}$	$2^{28.2}$

$\eta = 50,\ N = 2^{12},\ d = 4.$

Table: Pre-computation time of NCA-2.0 & NCA-3.0

Attack	l	Time	Memory	No. of tables
NCA-2.0	24	9 hours, 50 mins	643 MB	8192
NCA-3.0	24	6 hours, 35 mins	216 MB	378
NCA-2.0	32	27 hours,41 mins	4.45 GB	2097152
NCA-3.0	32	6 hours, 37 mins	11.6 MB	1562

$\eta = 50,\ N = 2^{12},\ d = 4.$

Simulations

- We apply NCA-2.0 and NCA-3.0 to the reduced version of Grain respectively for 140 randomly generated (K,IV) pairs.

- We apply NCA-2.0 and NCA-3.0 to the reduced version of Grain respectively for 140 randomly generated (K,IV) pairs.

Table: The simulation results on reduced version of Grain

Attack	l	Average Attack Time[a]	Success Probability
NCA-2.0	24	1 hours, 53 mins	9%
NCA-3.0	24	1 hours, 31 mins	7%
NCA-2.0	32	2 hours, 12 mins	6%
NCA-3.0	32	41 mins	4%

[a]This is the average time for each on-line attack.

Simulations

- We apply NCA-2.0 and NCA-3.0 to the reduced version of Grain respectively for 140 randomly generated (K,IV) pairs.

Table: The simulation results on reduced version of Grain

Attack	l	Average Attack Time[a]	Success Probability
NCA-2.0	24	1 hours, 53 mins	9%
NCA-3.0	24	1 hours, 31 mins	7%
NCA-2.0	32	2 hours, 12 mins	6%
NCA-3.0	32	41 mins	4%

[a] This is the average time for each on-line attack.

- The experimental time is based on running an non-optimized C++ program on a 1.83 GHz CPU with 2GB RAM and 1TB harddisk.

Simulations

- We apply NCA-2.0 and NCA-3.0 to the reduced version of Grain respectively for 140 randomly generated (K,IV) pairs.

Table: The simulation results on reduced version of Grain

Attack	l	Average Attack Time[a]	Success Probability
NCA-2.0	24	1 hours, 53 mins	9%
NCA-3.0	24	1 hours, 31 mins	7%
NCA-2.0	32	2 hours, 12 mins	6%
NCA-3.0	32	41 mins	4%

[a]This is the average time for each on-line attack.

- The experimental time is based on running an non-optimized C++ program on a 1.83 GHz CPU with 2GB RAM and 1TB harddisk.
- The success probability is the proportion of the number of the correct internal state difference stored in the KSD tables.

Simulations

- We apply NCA-2.0 and NCA-3.0 to the reduced version of Grain respectively for 140 randomly generated (K,IV) pairs.

Table: The simulation results on reduced version of Grain

Attack	l	Average Attack Time[a]	Success Probability
NCA-2.0	24	1 hours, 53 mins	9%
NCA-3.0	24	1 hours, 31 mins	7%
NCA-2.0	32	2 hours, 12 mins	6%
NCA-3.0	32	41 mins	4%

[a]This is the average time for each on-line attack.

- The experimental time is based on running an non-optimized C++ program on a 1.83 GHz CPU with 2GB RAM and 1TB harddisk.
- The success probability is the proportion of the number of the correct internal state difference stored in the KSD tables.

Further Discussion

- The success probability needs to be stabilized.

Further Discussion

- The success probability needs to be stabilized.
- Our attack need to be refined further and we indeed get some improvements by reducing the complexity of recovering the NFSR given the LFSR and the state difference. We will provide the details in the upcoming papers.

Further Discussion

- The success probability needs to be stabilized.
- Our attack need to be refined further and we indeed get some improvements by reducing the complexity of recovering the NFSR given the LFSR and the state difference. We will provide the details in the upcoming papers.
- We can also see that the experimental success probability of NCA-2.0 is lower than estimated in theory. The reason is that we choose a restricted value of η and N. These two parameters directly influence the size and the number of the pre-computed tables, hence affect the success probability.

Further Discussion

- The success probability needs to be stabilized.
- Our attack need to be refined further and we indeed get some improvements by reducing the complexity of recovering the NFSR given the LFSR and the state difference. We will provide the details in the upcoming papers.
- We can also see that the experimental success probability of NCA-2.0 is lower than estimated in theory. The reason is that we choose a restricted value of η and N. These two parameters directly influence the size and the number of the pre-computed tables, hence affect the success probability.
- How to theoretically derive the relationship between the success probability and these two parameters is our future work.

Outline

- Introduction
- Description of Grain v1
- Main idea & some key observations
- The general attack model: NCA-1.0
- NCA-2.0 & NCA-3.0
- Simulations
- Conclusions

Conclusions

- In this paper, we have proposed a key recovery attack, called near collision attack on Grain v1.

Conclusions

- In this paper, we have proposed a key recovery attack, called near collision attack on Grain v1.
- Based on some key observations, we have presented the basic attack called NCA-1.0 and further enhance it to NCA-2.0 and NCA-3.0 by combining the sampling resistance of Grain v1 and the non-uniform distribution of the KSD table size respectively.

Conclusions

- In this paper, we have proposed a key recovery attack, called near collision attack on Grain v1.
- Based on some key observations, we have presented the basic attack called NCA-1.0 and further enhance it to NCA-2.0 and NCA-3.0 by combining the sampling resistance of Grain v1 and the non-uniform distribution of the KSD table size respectively.
- Our attack has been verified on a reduced version of Grain v1 and an extrapolation of the results indicates an attack on the original Grain v1.

Conclusions

- In this paper, we have proposed a key recovery attack, called near collision attack on Grain v1.
- Based on some key observations, we have presented the basic attack called NCA-1.0 and further enhance it to NCA-2.0 and NCA-3.0 by combining the sampling resistance of Grain v1 and the non-uniform distribution of the KSD table size respectively.
- Our attack has been verified on a reduced version of Grain v1 and an extrapolation of the results indicates an attack on the original Grain v1.
- Our attack is just a starting point for further analysis of Grain-like stream ciphers and hopefully it provides some new insights on the design of such compact stream ciphers.

Thanks for your attention!