Cryptanalysis of Rabbit

Yi Lu, Huaxiong Wang and San Ling

Nanyang Technological University, Singapore

17 September 2008

Outline

- Review on Stream Ciphers and Distinguishing Attacks
- Introduction on Rabbit
- Our Attack against Rabbit
  - new distinguishing attack
  - first key-recovery attack
- Conclusion

Stream Ciphers: Basics

- Concept

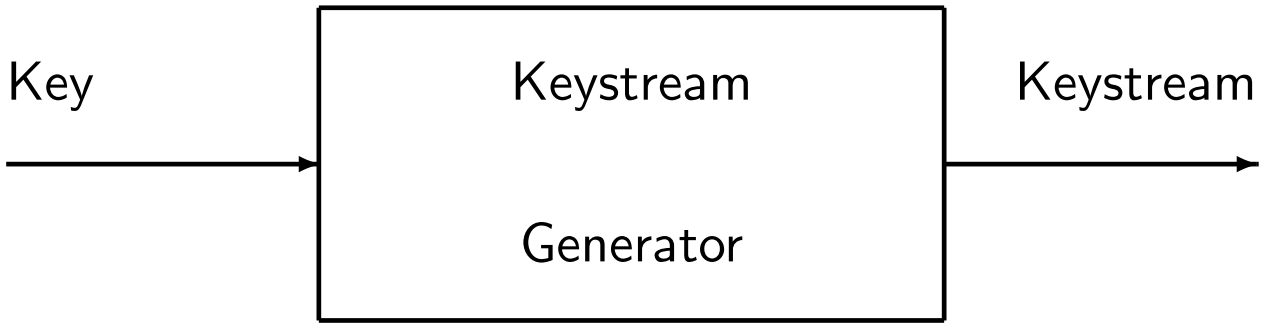
  Encryption: $\text{ciphertext} = \text{message} \oplus \text{keystream}$

  Decryption: $\text{message} = \text{ciphertext} \oplus \text{keystream}$

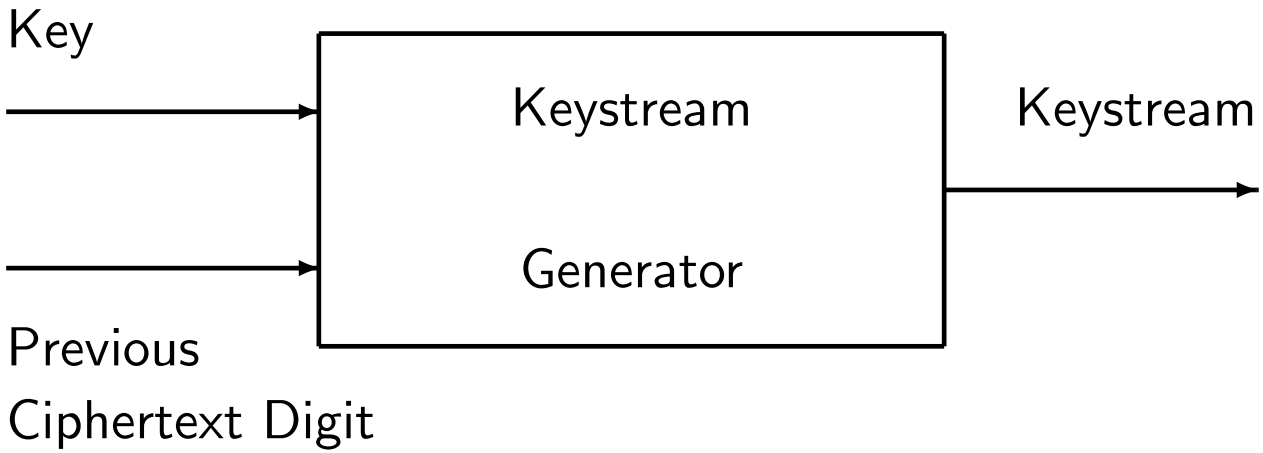
- Stream Ciphers: fast and low implementation cost in hardware.
- Main Problem: How to efficiently and securely generate keystream?

Stream Ciphers: Classification

1. synchronous stream ciphers (no error propagation):



2. asynchronous stream ciphers (limited error propagation):



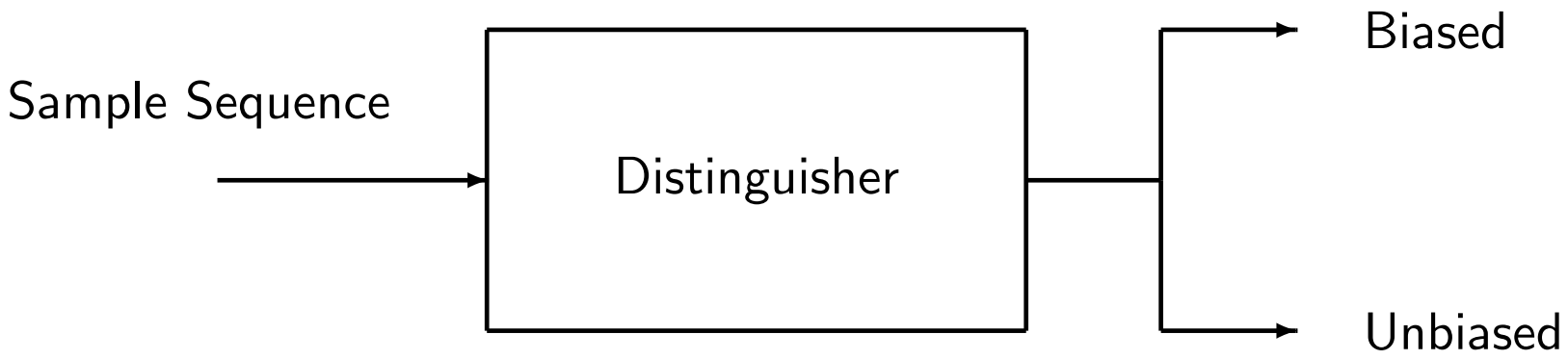
Stream Ciphers: Constructions

- Block ciphers with mode of operation
- t-functions
- Linear Feedback Shift Registers (LFSRs)
- Nonlinear Feedback Shift Registers
- etc.

Stream Ciphers: Attacks

- Attack Aims:
  - keystream distinguishing attack: to distinguish the output of keystream generator from a truly random sequence.
  - keystream predictting attack: to predict the output of the keystream generator with or without a keystream of certain length
  - key-recovery attack: to obtain the key
- Attack scenarios:
  - known-plaintext attacks (keystream is known)
  - ciphertext-only attacks (less common in cryptanalysis)

Simple Distinguisher

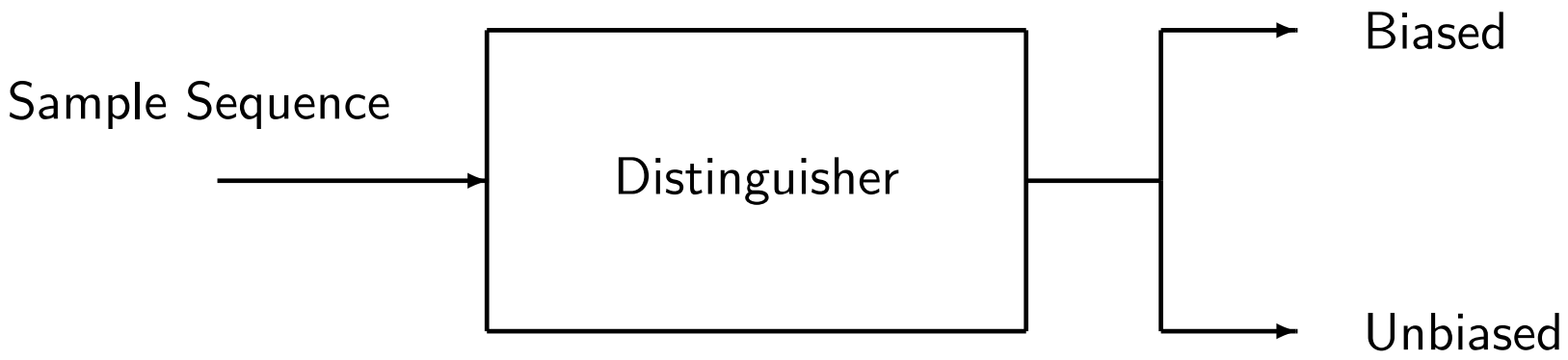


Sample length=1 bit:

$$\min \#\text{samples} = O\left(\frac{1}{\epsilon^2}\right),$$

where the bias $\epsilon \stackrel{\text{def}}{=} \Pr(\text{sample} = 1) - \Pr(\text{sample} = 0)$.

Advanced Distinguisher



Sample length$=r$ bits [BJV'04]:

$$\min \#\text{samples} = O\left(\frac{1}{\Delta(D)}\right),$$

where the Squared Euclidean Imbalance of the sample distribution D is defined by

$$\Delta(D) = 2^r \sum_a \left(D(a) - 2^{-r}\right)^2.$$

[BJV'04]: Thomas Baignères, Pascal Junod, and Serge Vaudenay, *How far can we go beyond linear cryptanalysis?*, ASIACRYPT2004.

Content

- Review on Stream Ciphers and Distinguishing Attacks
- Introduction on Rabbit
- Attack against Rabbit
- Conclusion

Rabbit: Introduction

- designed by M. Boesgaard, M. Vesterager, T. Pedersen, J. Christiansen and O. Scavenius and first presented at FSE 2003.
- one of the finalists of the ECRYPT Stream Cipher Project (eSTREAM) (April, 2008).

Profile 1 (SW)	Profile 2 (HW)
HC-128	F-FCSR-H v2
Rabbit	Grain v1
Salsa20/12	MICKEY v2
SOSEMANUK	Trivium

Rabbit: Introduction

- The design goals of Rabbit are:
  - **Security:** 128-bit security
  - **Speed:** faster than commonly used ciphers
- Rabbit was designed for both hardware and software implementations.
- Rabbit has a 128-bit key, 64-bit IV and 513-bit internal state.

Rabbit: Description

The internal state includes 16 words of 32-bits $x_{0,t}, \ldots, x_{7,t}$, $c_{0,t}, \ldots, c_{7,t}$ and 1-bit $\phi_{7,t}$. It's updated at each clock as follows.

$$c_{j,t} = \begin{cases} c_{0,t-1} + a_0 + \phi_{7,t-1} \mod 2^{32}, & \text{if } j = 0 \\ c_{j,t-1} + a_j + \phi_{j-1,t} \mod 2^{32}, & \text{if } j > 0 \end{cases}$$

where

$$\phi_{j,t+1} = \begin{cases} 1, & \text{if } j = 0 \text{ and } c_{0,t} + a_0 + \phi_{7,t} \geq 2^{32} \\ 1, & \text{if } j > 0 \text{ and } c_{j,t} + a_j + \phi_{j-1,t+1} \geq 2^{32} \\ 0, & \text{otherwise.} \end{cases}$$

And the a_i's are constants.

Rabbit: Next-State Function

$$
\begin{aligned}
x_{0,t+1} &= g_{0,t} + (g_{7,t} \lll 16) + (g_{6,t} \lll 16) \\
x_{1,t+1} &= g_{1,t} + (g_{0,t} \lll 8) + g_{7,t} \\
x_{2,t+1} &= g_{2,t} + (g_{1,t} \lll 16) + (g_{0,t} \lll 16) \\
x_{3,t+1} &= g_{3,t} + (g_{2,t} \lll 8) + g_{1,t} \\
x_{4,t+1} &= g_{4,t} + (g_{3,t} \lll 16) + (g_{2,t} \lll 16) \\
x_{5,t+1} &= g_{5,t} + (g_{4,t} \lll 8) + g_{3,t} \\
x_{6,t+1} &= g_{6,t} + (g_{5,t} \lll 16) + (g_{4,t} \lll 16) \\
x_{7,t+1} &= g_{7,t} + (g_{6,t} \lll 8) + g_{5,t}
\end{aligned}
$$

and

$$
g_{j,t} = (x_{j,t} + c_{j,t+1})^2 \oplus \left((x_{j,t} + c_{j,t+1})^2 \gg 32 \right),
$$

where all additions are modulo 2^{32}, and squarings modulo 2^{64}.

Rabbit: The Function g

- is essential for the security of Rabbit
- is to destroy linear relations in the cipher, *following initial ideas from chaos theory* (by the designers)
  - the modulo square in g is strongly non-linear
- is slow in hardware (of the 32-bit integer squaring)
- is fast in software environment

Rabbit: Keystream Generation

At each clock $t \geq 1$, a 128-bit keystream block s_t is produced:

$$
\begin{aligned}
s_t^{[15..0]} &= x_{0,t}^{[15..0]} \oplus x_{5,t}^{[31..16]} & s_t^{[31..16]} &= x_{0,t}^{[31..16]} \oplus x_{3,t}^{[15..0]} \\
s_t^{[47..32]} &= x_{2,t}^{[15..0]} \oplus x_{7,t}^{[31..16]} & s_t^{[63..48]} &= x_{2,t}^{[31..16]} \oplus x_{5,t}^{[15..0]} \\
s_t^{[79..64]} &= x_{4,t}^{[15..0]} \oplus x_{1,t}^{[31..16]} & s_t^{[95..80]} &= x_{4,t}^{[31..16]} \oplus x_{7,t}^{[15..0]} \\
s_t^{[111..96]} &= x_{6,t}^{[15..0]} \oplus x_{3,t}^{[31..16]} & s_t^{[127..112]} &= x_{6,t}^{[31..16]} \oplus x_{1,t}^{[15..0]}
\end{aligned}
$$

Content

- Review on Stream Ciphers and Distinguishing Attacks
- Introduction on Rabbit
- Attack against Rabbit
- Conclusion

Rabbit: Previous Attacks

- In a series of white papers, the designers gave convincing arguments that Rabbit is resistant against algebraic, correlation, differential, guess-and-determin, and statistical attacks.
- So far, only one paper [Aumasson'07] raised weakness on Rabbit and the work was on the bias.
- [Aumasson'07]: the core function g was shown to be unbalanced.
- The largest bias in the keystream, ie. $\epsilon(s_t^{[0]})$, $\epsilon(s_t^{[32]})$, $\epsilon(s_t^{[64]})$, $\epsilon(s_t^{[96]})$, was estimated to be $\approx 2^{-123.5}$

  $\Rightarrow$ the corresponding distinguishing attack has complexity $2^{247} (\gg 2^{128})$.

[Aumasson'07]: Jean-Philippe Aumasson, *On a bias of Rabbit*, SASC 2007 available at `http://www.ecrypt.eu.org/stream/papersdir/2007/033.pdf`

Rabbit: Our Distinguishing Attack

- Applying the techniques from [MJ05], the distribution of $x_{j,t}$ can be computed efficiently by Fast Fourier Transform (FFT) from the distributions of the three relevant g's.
- We have

$$D(x_0) = D(g) \otimes D(g \lll 16) \otimes D(g \lll 16),$$

  where $\otimes$ denotes convolution.
- Furthermore,

$$D(x_0) = \mathsf{FFT}^{-1}(\mathsf{FFT}(D(g)) \cdot \mathsf{FFT}(D(g \lll 16)) \cdot \mathsf{FFT}(D(g \lll 16))).$$

[MJ05]: Alexander Maximov, Thomas Johansson, *Fast computation of large distributions and its cryptographic applications*, ASIACRYPT'05.

Rabbit: Our Distinguishing Attack (cont'd)

- Large-scale computation was done to compute FFT for long sequence length of 2^{32}.
- To avoid loss of precision during computation, we computed all over integers instead of reals.
- Experiments (coded in C) used 4GB RAM, more than 400GB hard disk space; it took about 1 month to finish.

Rabbit: Our Distinguishing Attack Results

- First Results:

$$\Delta(D(s^{[15..0]})) = \Delta(D(s^{[47..32]})) = \Delta(D(s^{[79..64]})) = \Delta(D(s^{[111..96]})) \approx 2^{-158}.$$

  $\Rightarrow$ the distinguishing attack has complexity $2^{158} \ll 2^{247}$.

- This is much closer to the exhaustive search complexity 2^{128}.
- Comment: our result indicates that the approximation assumption in [Aumasson'07] is critical for estimation of the bias and cannot be ignored.

Rabbit: More about Experiments

Distribution	g	$x_1^{[31..16]}$	$x_1^{[15..0]}$	$x_1^{[31..0]}$	$x_0^{[15..0]}$	$s^{[15..0]}$
SEI	1.01	2^{-76}	2^{-100}	2^{-45}	2^{-100}	2^{-158}

Distribution	g	$x_0^{[31..16]}$	$x_0^{[15..0]}$	$x_0^{[31..0]}$	$x_1^{[15..0]}$	$s^{[31..16]}$
SEI	1.01	2^{-76}	2^{-100}	2^{-45}	2^{-100}	2^{-160}

- From above tables, the unbalanced g is somehow transformed to a nearly-balanced keystream output s.
- We consider 16-bits samples and apply advanced distinguisger.
- We gain an improvement with a reduced factor 2^{35} to the previous attack of 1-bit samples with simple distinguisher.

Rabbit: Extended Key-recovery Attack

- Problem: try to extend the distinguishing attack to a key-recovery attack.
- Assumption: we know many frames of keystreams, and we also know the relations between their internal states x's. More specifically,
  - we consider m ($m \approx 2^{51.5}$) frames and we know the first three subblocks of each frame
  - we know $d^i_{j,1}$ and $d^i_{j,2}$, where

$$\begin{aligned} x^1_{j,1} \oplus x^i_{j,1} &= d^i_{j,1} \\ x^1_{j,2} \oplus x^i_{j,2} &= d^i_{j,2} \end{aligned}$$

- Aim: we want to recover the keys for all frames from the known $d^i_{j,1}, d^i_{j,2}$'s and keystreams s^i's as short as possible.

Rabbit: Sketch on Key-recovery Attack

Step 1: solve $x^i_{j,1}$'s (resp. $x^i_{j,2}$'s) from $d^i_{j,1}$'s (resp. $d^i_{j,2}$'s).

Step 2: solve $g^i_{j,0}$'s (resp. $g^i_{j,1}$'s) from $x^i_{j,1}$'s (resp. $x^i_{j,2}$'s).

Step 3: recover $c^i_{j,2}$'s and $\phi^i_{7,2}$'s.

Step 4: recover the keys from the internal states.

Rabbit: Key-recovery Attack Summary

step	precomputation	memory	time
1	-	-	$2^{86.5}$
2	2^{32}	2^{32}	$2^{97.5}$
3	-	-	2^{58}
4	-	-	$2^{83.5}$
Total	2^{32}	2^{32}	$2^{97.5}$

Comment:

This is the first known key-recovery attack on Rabbit so far, though the attack assumption is stronger than usual.

Conclusion

- Thanks to FFT, we computed the exact bias of Rabbit's keystream sub-blocks.
- Our result leads to the best distinguishing attack with complexity $2^{158} \ll 2^{247}$.
- Assuming that we know the relation between part of the internal states of all frames, we extend our distinguishing attack to a multi-frame key-recovery attack.
- It remains an open challenge to further improve our distinguishing attack to complexity below 2^{128}.