

Design Principles for Combiners with Memory

Frederik Armknecht
Matthias Krause
Dirk Stegemann

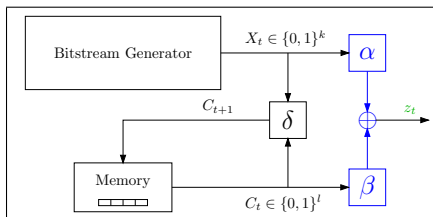
Theoretical Computer Science
University of Mannheim, Germany

INDOCRYPT 2005
December 10-12, Bangalore, India

Overview

- 1 Introduction
- 2 Design Principles against Correlation Attacks and Algebraic Attacks
- 3 Application to E_0
- 4 Conclusion

Combiners With Memory



Initialization in $t = 0$

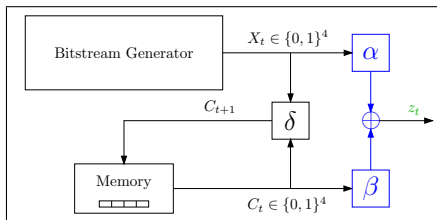
- n -bit **secret key \mathcal{K}**
→ Bitstream Generator
- initial value C_0
→ Memory

In each clock $t > 0$

- Produce keybit: $z_t = f(C_t, X_t)$
- Update memory bits: $C_{t+1} = \delta(C_t, X_t)$

Special case: $f(C_t, X_t) = \alpha(X_t) \oplus \beta(C_t)$

Practical Example: Bluetooth E_0



$$\alpha(X_t) = X_t^1 \oplus X_t^2 \oplus X_t^3 \oplus X_t^4$$

$$\beta(C_t) = C_t^2$$

Correlation Attacks and Algebraic Attacks

... exploit equations in internal bits and corresponding output bits

$$F(\underbrace{X_t, \dots, X_{t+r-1}}_{\text{internal bits}}, \underbrace{Z_t, \dots, Z_{t+r-1}}_{r \text{ output bits}}) = 0$$

in order to recover the **secret key** \mathcal{K} .

correlation attacks:

- $\deg(F) = 1$
- equations F biased, i.e. true with probability $\frac{1}{2} + \lambda$, $\lambda \neq 0$

algebraic attacks:

- $\deg(F) = d > 1$
- equations F true with probability 1

A Special Correlation Attack

Idea:

- Output bits are computed as $z_t = \alpha(X_t) \oplus \beta(C_t)$.
- Exploit biased linear combinations of the $\beta(C_t)$.

More precisely: Find $\gamma = (\gamma_0, \dots, \gamma_{r-1}) \in \{0, 1\}^r$ such that

$$\lambda(\gamma) = \left(Pr \left[\bigoplus_{i=0}^{r-1} \gamma_i \cdot \beta(C_{t+i}) = 0 \right] - Pr \left[\bigoplus_{i=0}^{r-1} \gamma_i \cdot \beta(C_{t+i}) = 1 \right] \right) \neq 0$$

Observation: Attack efficiency increases with $|\lambda|$.

Theorem (Lu, Vaudenay 2004)

For the E_0 generator, it holds that $|\lambda| \leq \frac{25}{256}$ for $r \leq 25$.

An Algebraic Attack

Scenario:

- φ many Z-functions F_Z of degree d for each clock t fulfilling

$$F_Z(X_t, \dots, X_{t+r-1}) = 0 \text{ for each } Z = (z_t, \dots, z_{t+r-1}) \in \{0, 1\}^r$$

- number of equations \approx number of monomials $\approx \binom{n}{d}$

Observation: Attack efficiency increases with decreasing d .

Theorem (Armknecht, Krause 2003)

For the E_0 generator, \exists Z-functions with $d = 4$ and $r = 4$.

Computation of λ and d

- For $\gamma = (\gamma_0, \dots, \gamma_{r-1})$, compute $\lambda(\gamma)$ by subsequent multiplication of the bias matrices B_{γ_i} .
→ Feasible for r up to 25 and more
- Find minimum d by systematically considering all possible outputs z_t, \dots, z_{t+r-1} over r clock cycles.
→ Feasible for $d \leq 7$

Resisting Correlation Attacks and Algebraic Attacks

In order to improve the combiner's security

- $|\lambda| \rightarrow \min$
- $d \rightarrow \max$

How to minimize $|\lambda|$

Theorem

If for all $1 \leq t \leq r$

- $z_t = \alpha_t(X_t) \oplus \beta_t(C_t)$, $\beta_t \neq 0$ for at least one t
- all X_t independent
- $\beta_t \neq 0 \Rightarrow \beta_t$ balanced, i.e. $|\beta_t^{-1}(0)| = |\beta_t^{-1}(1)|$
- δ balanced, i.e. $\Pr[C \rightarrow C']$ is equal for all C, C'

then $\lambda = \lambda(\gamma) = 0$ for all $\gamma \in \{0, 1\}^r$.

In the case of E_0 :

$$\checkmark \quad z_t = \underbrace{X_t^1 \oplus X_t^2 \oplus X_t^3 \oplus X_t^4}_{\alpha(X_t)} \oplus \underbrace{C_t^2}_{\beta(C_t)}$$

\checkmark X_t independent for $r \leq 25$

\checkmark β balanced

\ominus δ not balanced

How to maximize d

Definition

For a subset $A \subseteq \{0, 1\}^n$,

- we denote by $Ann(A)$ the set of all Boolean functions $f : \{0, 1\}^n \rightarrow \{0, 1\}$ such that $f(x) = 0$ for all $x \in A$
- we define $mindeg(A) = \min\{deg(f) : f \in Ann(A)\}$

Theorem

If for all $1 \leq t \leq r$

- $z_t = \alpha(X_t) \oplus \beta(C_t)$
- all X_t independent
- $mindeg(\alpha^{-1}(0)) = mindeg(\alpha^{-1}(1)) = d = AI(\alpha)$

then $deg(F_Z) \geq d$.

How to maximize d

Idea: Find a function α with
 $\text{mindeg}(\alpha^{-1}(1)) = \text{mindeg}(\alpha^{-1}(0)) = 1$ as large as possible.

How large is the largest possible?

Lemma

For each Boolean function $\alpha : \{0, 1\}^k \rightarrow \{0, 1\}$,

$$\text{mindeg}(\alpha^{-1}(0)), \text{mindeg}(\alpha^{-1}(1)) \leq \left\lceil \frac{k}{2} \right\rceil$$

This bound is matched by the majority function:

Lemma

For the Function

$$\begin{aligned} \text{maj} : \{0, 1\}^k &\rightarrow \{0, 1\} \\ x &\mapsto \begin{cases} 0 & \text{weight}(x) \leq \lfloor k/2 \rfloor \\ 1 & \text{otherwise} \end{cases} \quad \text{for } k \text{ odd} \\ &\quad \text{(similarly for } k \text{ even)} \end{aligned}$$

it holds that $\text{mindeg}(\text{maj}^{-1}(0)) = \text{mindeg}(\text{maj}^{-1}(1)) = \left\lceil \frac{k}{2} \right\rceil$

Improving E_0

Increase resistance

... against correlation attacks by decreasing λ

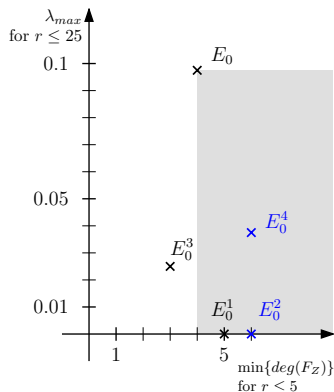
- Replace δ by a balanced function $\rightarrow \lambda = 0$ for $r \leq 25$

... against algebraic attacks by increasing $\min\{\deg(F_Z)\}$

- Replace $\alpha(X_t)$ by $\text{maj}(X_t)$ $\rightarrow \deg(F_Z) \geq 2$ for $r \leq 25$

Improved Variants of E_0

	$\alpha(X_t)$	$\beta(C_t)$	$\delta(X_t, C_t)$
E_0	$\bigoplus X_t^i$	C_t^2	<i>original</i>
E_0^1	$\text{maj}(X_t)$	C_t^2	$X_t + C_t$
E_0^2	$\text{maj}(X_t)$	$\text{maj}(C_t)$	$X_t + C_t$
E_0^3	$\bigoplus X_t^i$	$C_t^2 \oplus C_t^3 \oplus C_t^4$	<i>original</i>
E_0^4	$\bigoplus X_t^i$	$C_t^1 \oplus C_t^3 \oplus C_t^4$	<i>original</i>



The favourite candidates:

- E_0^2 : highest resistance
- E_0^4 : significant improvement with only small changes

Time Complexities

	Correlation Attack	Algebraic Attack
E_0	2^{40}	2^{70}
E_0^2	n/a	$2^{97.22}$
E_0^4	$2^{52.15}$	$2^{97.22}$

Conclusion

- Theoretical design principles against certain types of
 - correlation attacks
 - algebraic attacks

- Application to the E_0 generator yields
 - significantly improved resistance against these attacks
 - with only small changes of the design

But: What about other attacks?

Thank you!

{frederik.armknecht, matthias.krause, dirk.stegemann}
@th.informatik.uni-mannheim.de