RUHR-UNIVERSITÄT BOCHUM

Security Arguments and Tool-based Design of Block Ciphers

PhD Defense

December 13th, 2019

Workgroup Symmetric Cryptography, Horst-Görtz-Institute for IT-Security, Ruhr University Bochum
Friedrich Wiemer

2019-12-07
Security Arguments and Tool-based Design of Block Ciphers

Topics of the Thesis

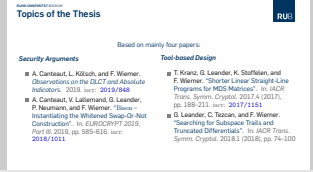
Based on mainly four papers:

Security Arguments

- A. Canteaut, L. Kölsch, and F. Wiemer. *Observations on the DLCT and Absolute Indicators*. 2019. iacr: 2019/848
- A. Canteaut, V. Lallemand, G. Leander, P. Neumann, and F. Wiemer. "Bison – Instantiating the Whitened Swap-Or-Not Construction". In: *EUROCRYPT 2019, Part III*. 2019, pp. 585–616. iacr: 2018/1011

Tool-based Design

- T. Kranz, G. Leander, K. Stoffelen, and F. Wiemer. "Shorter Linear Straight-Line Programs for MDS Matrices". In: *IACR Trans. Symm. Cryptol.* 2017.4 (2017), pp. 188–211. iacr: 2017/1151
- G. Leander, C. Tezcan, and F. Wiemer. "Searching for Subspace Trails and Truncated Differentials". In: *IACR Trans. Symm. Cryptol.* 2018.1 (2018), pp. 74–100



The General Setting

Block Ciphers and Security Notion

RUB

Definition: Block Cipher

- $E : \mathcal{K} \times \mathcal{M} \to \mathcal{C}$ family of permutations, with
  $\mathcal{K}$ the key space, $\mathcal{M}$ the message space, and $\mathcal{C}$ the cipher space
- In practice: $\mathcal{K} = \mathbb{F}_2^s$ and $\mathcal{M} = \mathcal{C} = \mathbb{F}_2^n$.
  $E_k = E(k, \cdot)$ the *encryption* under key k,
  $D_k = E_k^{-1}$ the *decryption*,
  s the *key length* and n the *block lengh*.

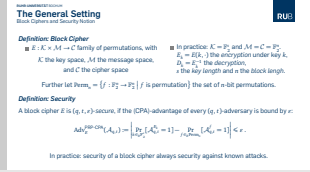
Further let $\mathrm{Perm}_n = \left\{ f : \mathbb{F}_2^n \to \mathbb{F}_2^n \;\middle|\; f \text{ is permutation} \right\}$ the set of n-bit permutations.

Definition: Security

A block cipher E is (q, t, ε)-*secure*, if the (CPA)-advantage of every (q, t)-adversary is bound by ε:

$$\mathrm{Adv}_E^{\text{PRP-CPA}}(\mathcal{A}_{q,t}) := \left| \Pr_{k \in_R \mathbb{F}_2^s}[\mathcal{A}_{q,t}^{E_k} = 1] - \Pr_{f \in_R \mathrm{Perm}_n}[\mathcal{A}_{q,t}^{f} = 1] \right| \leqslant \varepsilon \,.$$

In practice: security of a block cipher always security against known attacks.

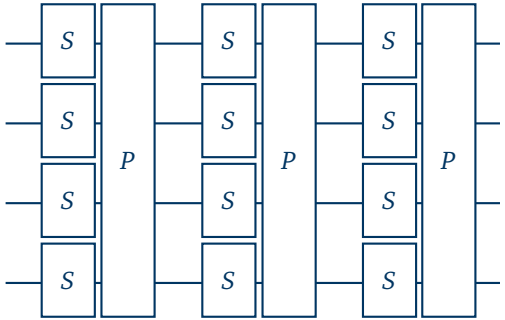


Substitution Permutation Networks

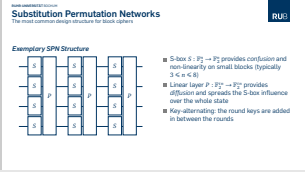
The most common design structure for block ciphers

RUB

Exemplary SPN Structure

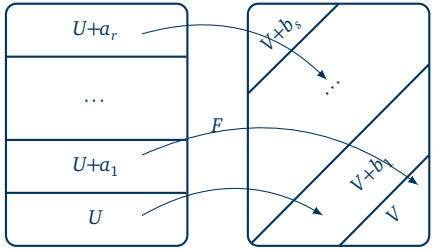


- S-box $S : \mathbb{F}_2^n \to \mathbb{F}_2^n$ provides *confusion* and non-linearity on small blocks (typically $3 \leqslant n \leqslant 8$)
- Linear layer $P : \mathbb{F}_2^{tn} \to \mathbb{F}_2^{tn}$ provides *diffusion* and spreads the S-box influence over the whole state
- Key-alternating: the round keys are added in between the rounds



Overview

Subspace Trail Cryptanalysis



Idea: Subspace Trails

Def.: Subspace Trails [GRR16] (FSE'16)

Let $U_0, \ldots, U_r \subseteq \mathbb{F}_2^n$, and $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$. If

$$\forall a \in U_i^{\perp} : \exists b \in U_{i+1}^{\perp} : \quad F(U_i + a) \subseteq U_{i+1} + b \,,$$

these subspaces form a *subspace trail* (ST).

Notation: $U_0 \to^F \cdots \to^F U_r$.

(We only look at the probability-one variant here)

The Problem/Our Goal

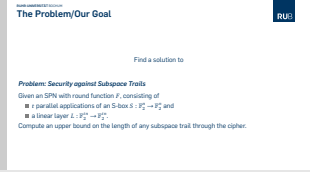
Find a solution to

Problem: Security against Subspace Trails

Given an SPN with round function F, consisting of

- t parallel applications of an S-box $S : \mathbb{F}_2^n \to \mathbb{F}_2^n$ and
- a linear layer $L : \mathbb{F}_2^{tn} \to \mathbb{F}_2^{tn}$.

Compute an upper bound on the length of any subspace trail through the cipher.



Subspace Propagation

Given a starting subspace U, we can efficiently compute the corresponding longest subspace trail.

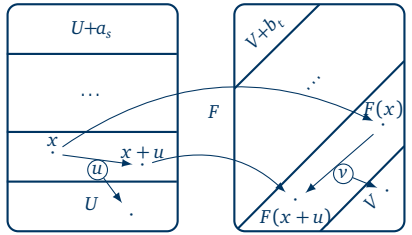
Lemma 1

Let $U \xrightarrow{F} V$ be a subspace trail. Then

$$\forall u \in U, x \in \mathbb{F}_2^m : \Delta_u(F)(x) := F(x)+F(x+u) \in V \, .$$

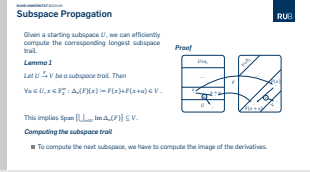
This implies $\mathrm{Span}\left\{\bigcup_{u \in U} \mathrm{Im}\, \Delta_u(F)\right\} \subseteq V$.

Proof



Computing the subspace trail

- To compute the next subspace, we have to compute the image of the derivatives.



Propagate a Basis

Actually it is enough to compute only the image of the derivatives in direction of U's basis vectors.

Lemma 2

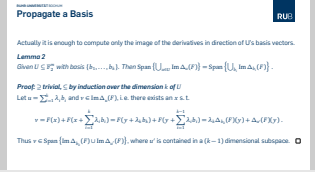
Given $U \subseteq \mathbb{F}_2^m$ *with basis* $\{b_1, \ldots, b_k\}$. *Then* $\operatorname{Span}\left\{\bigcup_{u \in U} \operatorname{Im} \Delta_u(F)\right\} = \operatorname{Span}\left\{\bigcup_{b_i} \operatorname{Im} \Delta_{b_i}(F)\right\}$.

Proof: $\supseteq$ ***trivial,*** $\subseteq$ ***by induction over the dimension*** k ***of*** U

Let $u = \sum_{i=1}^{k} \lambda_i b_i$ and $v \in \operatorname{Im} \Delta_u(F)$, i. e. there exists an x s. t.

$$v = F(x) + F(x + \sum_{i=1}^{k} \lambda_i b_i) = F(y + \lambda_k b_k) + F(y + \sum_{i=1}^{k-1} \lambda_i b_i) = \lambda_k \Delta_{b_k}(F)(y) + \Delta_{u'}(F)(y)\,.$$

Thus $v \in \operatorname{Span}\left\{\operatorname{Im} \Delta_{b_k}(F) \cup \operatorname{Im} \Delta_{u'}(F)\right\}$, where u' is contained in a $(k-1)$ dimensional subspace. □



ComputeTrail Algorithm

RUB

Computation of Subspace Trails

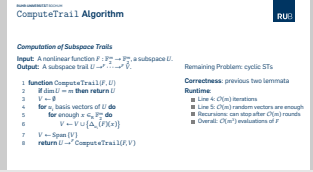
Input: A nonlinear function $F : \mathbb{F}_2^m \to \mathbb{F}_2^m$, a subspace U.
Output: A subspace trail $U \to^F \cdots \to^F V$.

```
1 function ComputeTrail(F, U)
2     if dim U = m then return U
3     V ← ∅
4     for u_i basis vectors of U do
5         for enough x ∈_R F_2^m do
6             V ← V ∪ {Δ_{u_i}(F)(x)}
7     V ← Span{V}
8     return U →^F ComputeTrail(F, V)
```

Remaining Problem: cyclic STs

Correctness: previous two lemmata
Runtime:

- Line 4: $\mathcal{O}(m)$ iterations
- Line 5: $\mathcal{O}(m)$ random vectors are enough
- Recursions: can stop after $\mathcal{O}(m)$ rounds
- Overall: $\mathcal{O}(m^3)$ evaluations of F



How many random vectors are enough:
https://math.stackexchange.com/questions/564603/probability-that-a-random-binary-matrix-will-have-full-column-rank
Draw n random vectors of length m, but because $n > m$ look at the probability

$\Pr[m \text{ vectors of length } n \text{ have rank } m] = \Pr[m \text{ vectors on length } n \text{ are linearly independent}]$.

The uniformly at random drawn $i+1$-th vector is linearly independent of the previous i vectors with probability

$$\frac{2^n - 2^i}{2^n} = 1 - 2^{i-n}$$

and thus for all vectors up to m we get the probability

$$\prod_{i=0}^{m-1} (1 - 2^{i-n}) .$$

How to Bound the Length of Subspace Trails

RUB

Goal

Give an upper bound on the length of any subspace trail.

Naïve Approach 1

$\forall U \subseteq \mathbb{F}_2^m$ run $\texttt{ComputeTrail}(F, U)$

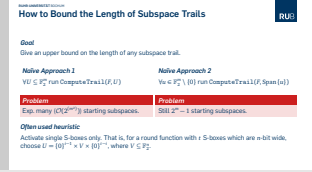
Naïve Approach 2

$\forall u \in \mathbb{F}_2^m \setminus \{0\}$ run $\texttt{ComputeTrail}(F, \mathrm{Span}\,\{u\})$

Problem	*Problem*
Exp. many $(\mathcal{O}(2^{(m^2)}))$ starting subspaces.	Still $2^m - 1$ starting subspaces.

Often used heuristic

Activate single S-boxes only. That is, for a round function with t S-boxes which are n-bit wide, choose $U = \{0\}^{i-1} \times V \times \{0\}^{t-i}$, where $V \subseteq \mathbb{F}_2^n$.



How many subspaces of $\mathbb{F}_2^n$:
https://math.stackexchange.com/questions/70801/how-many-k-dimensional-subspaces-there-are-in-n-dimensional-vector-space-over

$$\sum_{r=1}^{n} \frac{A(n,r)}{B(n,r)} = \sum_{r=1}^{n} \frac{\prod_{i=1}^{r}(1-2^{n-i+1})}{\prod_{i=1}^{r}(1-2^{i})} = \sum_{r=1}^{n} \binom{n}{r}_2$$

$A(n,r)$: Number of subsets with r linear independent elements:

$$A(n,r) = \underbrace{(2^n-1)}_{\text{choices for first element}} \overbrace{(2^n-2)}^{\text{choices for second element}} \cdots \underbrace{(2^n-2^{r-1})}_{\text{choices for } r\text{-th element}}$$

$B(n,r)$: Number of bases for one r-dimensional subspace:

$$B(n,r) = \underbrace{(2^r-1)}_{\text{choices for first basis vector}} \overbrace{(2^r-2)}^{\text{choices for second basis vector}} \cdots \underbrace{(2^r-2^{r-1})}_{\text{choices for } r\text{-th vector}}$$

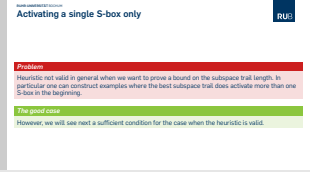
Activating a single S-box only

Problem

Heuristic not valid in general when we want to prove a bound on the subspace trail length. In particular one can construct examples where the best subspace trail does activate more than one S-box in the beginning.

The good case

However, we will see next a sufficient condition for the case when the heuristic is valid.



The Connection to Linear Structures

Let us observe how a single S-box S behaves regarding subspace trails:

Given a subpsace trail $U \xrightarrow{S} V$, this implies

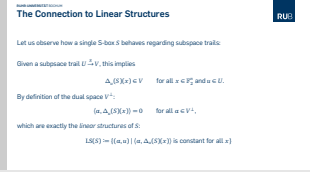
$$\Delta_u(S)(x) \in V \qquad \text{for all } x \in \mathbb{F}_2^n \text{ and } u \in U.$$

By definition of the dual space $V^\perp$:

$$\langle \alpha, \Delta_u(S)(x) \rangle = 0 \qquad \text{for all } \alpha \in V^\perp,$$

which are exactly the *linear structures* of S:

$$\mathrm{LS}(S) := \{(\alpha, u) \mid \langle \alpha, \Delta_u(S)(x) \rangle \text{ is constant for all } x\}$$



S-boxes without Linear Structures

This observation implies that S-boxes without linear structures (e. g. the AES S-box) exhibit only two important subspace trails:

$$\{0\} \to \{0\} \qquad \text{and} \qquad \mathbb{F}_2^n \to \mathbb{F}_2^n$$

We can further show that subspace trails over an S-box layer without linear structures are direct products of the above two subspace trails.

Theorem 3

Let F be an S-box layer of t parallel S-boxes $S : \mathbb{F}_2^n \to \mathbb{F}_2^n$. If S has no non-trivial linear structures, then for every subspace trail $U \to^F V$:

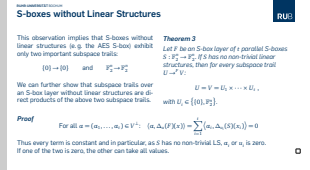
$$U = V = U_1 \times \cdots \times U_t \,,$$

with $U_i \in \{\{0\}, \mathbb{F}_2^n\}$.

Proof

$$\text{For all } \alpha = (\alpha_1, \ldots, \alpha_t) \in V^\perp: \quad \langle \alpha, \Delta_u(F)(x) \rangle = \sum_{i=1}^{t} \left\langle \alpha_i, \Delta_{u_i}(S)(x_i) \right\rangle = 0$$

Thus every term is constant and in particular, as S has no non-trivial LS, α_i or u_i is zero.
If one of the two is zero, the other can take all values. □



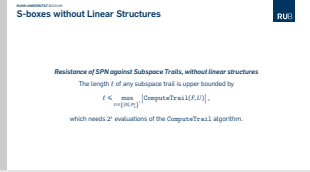
S-boxes without Linear Structures

Resistance of SPN against Subspace Trails, without linear structures

The length ℓ of any subspace trail is upper bounded by

$$\ell \leqslant \max_{U \in \left\{\{0\}, \mathbb{F}_2^n\right\}^t} \left|\texttt{ComputeTrail}(F, U)\right| ,$$

which needs 2^t evaluations of the `ComputeTrail` algorithm.



S-boxes with Linear Structures

Compared to the no-linear-structures-case, $V^{\perp}$ can now contain much more elements, namely all combinations of linear structures, such that their corresponding constants sum to zero.

Instead, we can show that (for any not-trivially-insecure S-box) the subspace after the first S-box layer contains at least one element of a specific structure:

$$W_{i,\alpha} = \{0\}^{i-1} \times \{0, \alpha\} \times \{0\}^{t-i} .$$

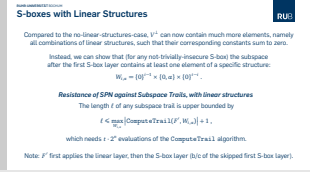
Resistance of SPN against Subspace Trails, with linear structures

The length ℓ of any subspace trail is upper bounded by

$$\ell \leqslant \max_{W_{i,\alpha}} \left|\texttt{ComputeTrail}(F', W_{i,\alpha})\right| + 1 ,$$

which needs $t \cdot 2^n$ evaluations of the `ComputeTrail` algorithm.

Note: F' first applies the linear layer, then the S-box layer (b/c of the skipped first S-box layer).



Conclusion

Thanks for your attention!

RUB

Applications of `ComputeTrail`

- Bound longest probability-one subspace trail
- Finding key-recovery strategies

Further Results on Subspace Trails

- Link to Truncated Differentials