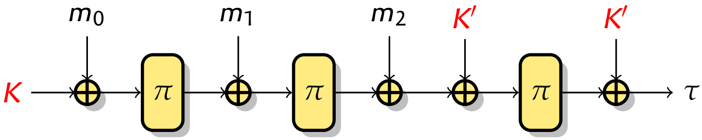
Improved Differential-Linear Cryptanalysis of 7-round Chaskey with Partitioning

Gaëtan Leurent

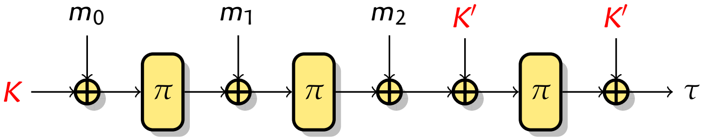
Inria, Paris

Eurocrypt 2016



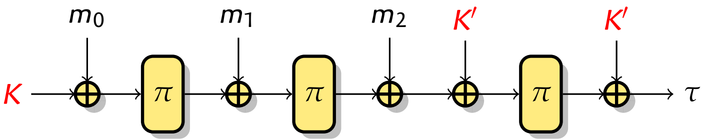
Chaskey

N. Mouha, B. Mennink, A. Van Herrewege, D. Watanabe, B. Preneel, I. Verbauwhede
Chaskey: An Efficient MAC Algorithm for 32-bit Microcontrollers
SAC 2014



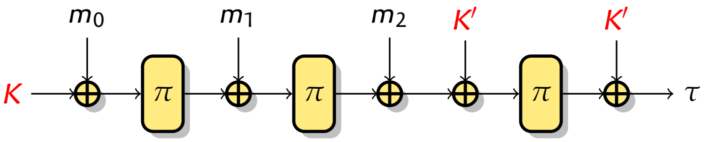
Chaskey

- Message Authentication Code
  - Authenticity
  - $\tau = \mathrm{MAC}_K(m)$
    1. Computed by Alice
    2. Transmitted with m
    3. Verified by Bob (same key)
- For microcontrollers
  - Typical use-case: sensor network (lightweight)
  - "Ten times faster than AES"
- Considered for ISO standardisation

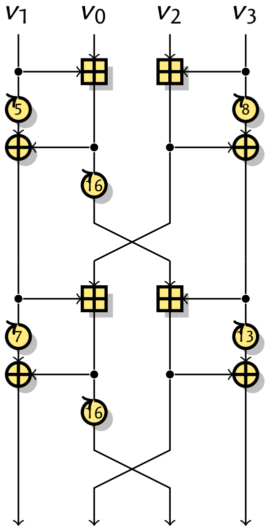


Chaskey

- CBC-MAC with an Even-Mansour cipher
  - Permutation based (sponge-like)
- Birthday security
  - 128-bit key ($K' = 2 \cdot K$)
  - 128-bit state
  - Security claim: 2^{48} data, 2^{80} time ($TD > 2^{128}$).



Chaskey permutation

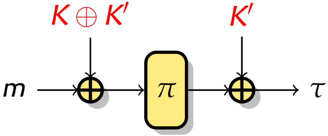


- 32-bit words
- 128-bit state
- ARX scheme
  - Additions $\pmod{2^{32}}$
  - Rotations (bitwise)
  - Xor
- Same structure as Siphash
- 8 rounds

Cryptanalysis of Chaskey

Exploiting properties of the π permutation

- Use single-block messages
  - Chaskey becomes an Even-Mansour cipher
  - No decryption oracle
- Previous work: 4-round bias by the designers
  - 5-round attack?



Main Cryptanalysis Techniques

Differential Cryptanalysis

Track difference propagation

[Biham & Shamir, 1990]

- Input/output differences δ_P, δ_C
- $E(x \oplus \delta_P) \approx E(x) \oplus \delta_C$

$p = \Pr\left[E(P \oplus \delta_P) = E(P) \oplus \delta_C\right]$

- Concatenate trails: $p = \prod p_i$
- Complexity $1/p$
  - Require $p \gg 2^{-n}$

Linear Cryptanalysis

Track linear approximations

[Matsui, 1992]

- Input/output masks χ_P, χ_C
- $E(x)[\chi_C] \approx x[\chi_P]$

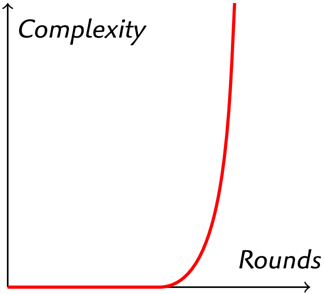
$\varepsilon = 2\Pr\left[E(x)[\chi_C] = x[\chi_P]\right] - 1$

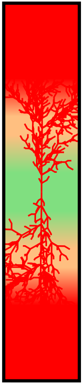
- Concatenate trails: $\varepsilon = \prod \varepsilon_i$
- Complexity $1/\varepsilon^2$
  - Require $\varepsilon \gg 2^{-n/2}$

$$x[\chi_1 \dots \chi_\ell] = x[\chi_1] \oplus x[\chi_2] \cdots x[\chi_\ell]$$

Cryptanalysis of ARX schemes

- No iterative differential/linear trails
- Small difference in the middle and propagate
- Only short trails
  with high probability

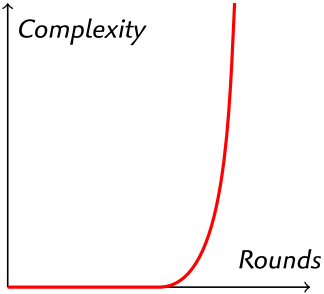




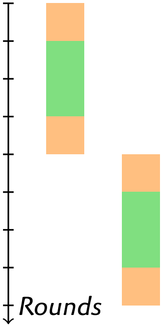
Cryptanalysis of ARX schemes

- No iterative differential/linear trails
- Small difference in the middle and propagate

- Only short trails
  with high probability



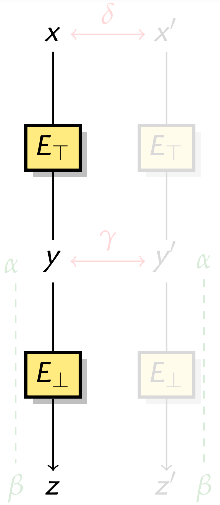
- Can we combine two trails?



Differential-Linear Cryptanalysis

[Langford & Hellman, 1994]
[Biham, Dunkelman & Keller, 2002]

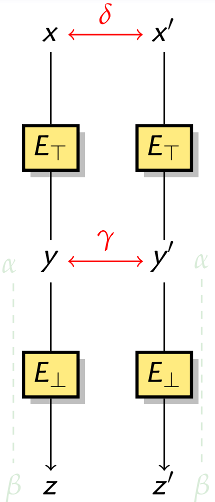
- Divide E in two sub-ciphers $E = E_\perp \circ E_\top$
  - Let $y = E_\top(x)$, $z = E_\perp(y)$
- Find a differential $\delta \to \gamma$ for $E_\top$
  - $\Pr[E_\top(x \oplus \delta) = E_\top(x) \oplus \gamma] = p$
- Find a linear approximation $\alpha \to \beta$ of $E_\perp$
  - $\Pr[y[\alpha] = E_\perp(y)[\beta]] = \frac{1}{2}(1+\varepsilon)$



Differential-Linear Cryptanalysis

[Langford & Hellman, 1994]
[Biham, Dunkelman & Keller, 2002]

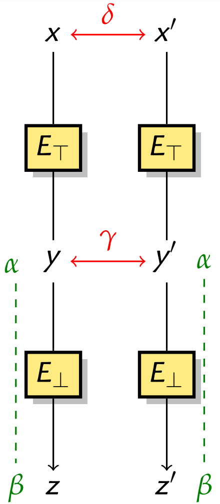
- Divide E in two sub-ciphers $E = E_\bot \circ E_\top$
  - Let $y = E_\top(x)$, $z = E_\bot(y)$
- Find a differential $\delta \to \gamma$ for $E_\top$
  - $\Pr\left[E_\top(x \oplus \delta) = E_\top(x) \oplus \gamma\right] = p$
- Find a linear approximation $\alpha \to \beta$ of $E_\bot$
  - $\Pr\left[y[\alpha] = E_\bot(y)[\beta]\right] = \frac{1}{2}(1 + \varepsilon)$



Differential-Linear Cryptanalysis

[Langford & Hellman, 1994]
[Biham, Dunkelman & Keller, 2002]

- Divide E in two sub-ciphers $E = E_\perp \circ E_\top$
  - Let $y = E_\top(x)$, $z = E_\perp(y)$
- Find a differential $\delta \to \gamma$ for $E_\top$
  - $\Pr\left[E_\top(x \oplus \delta) = E_\top(x) \oplus \gamma\right] = p$
- Find a linear approximation $\alpha \to \beta$ of $E_\perp$
  - $\Pr\left[y[\alpha] = E_\perp(y)[\beta]\right] = \frac{1}{2}(1+\varepsilon)$

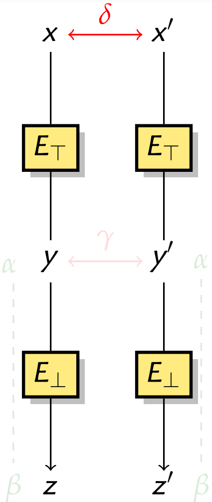


Differential-Linear Cryptanalysis

- Query a pair $(x, x' = x \oplus \delta)$:

$y \oplus y' = \gamma$	proba p
$(y \oplus y')[\alpha] = \gamma[\alpha]$	proba $\approx p + 1/2(1-p)$
$z[\beta] = y[\alpha]$	proba $1/2(1+\varepsilon)$
$z'[\beta] = y'[\alpha]$	proba $1/2(1+\varepsilon)$
$(z \oplus z')[\beta] = \gamma[\alpha]$	proba $1/2(1+p\varepsilon^2)$

- Distinguisher with complexity $\approx p^{-2}\varepsilon^{-4}$

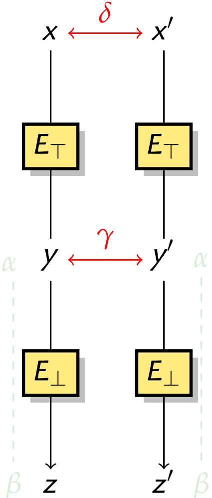


Differential-Linear Cryptanalysis

- Query a pair $(x, x' = x \oplus \delta)$:

$$
\begin{aligned}
y \oplus y' &= \gamma && \text{proba } p \\
(y \oplus y')[\alpha] &= \gamma[\alpha] && \text{proba} \approx p + 1/2(1-p) \\
z[\beta] &= y[\alpha] && \text{proba } 1/2(1+\varepsilon) \\
z'[\beta] &= y'[\alpha] && \text{proba } 1/2(1+\varepsilon) \\
(z \oplus z')[\beta] &= \gamma[\alpha] && \text{proba } 1/2(1+p\varepsilon^2)
\end{aligned}
$$

- Distinguisher with complexity $\approx p^{-2}\varepsilon^{-4}$

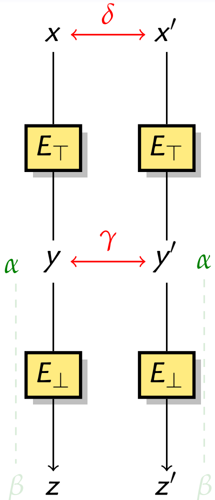


Differential-Linear Cryptanalysis

- Query a pair $(x, x' = x \oplus \delta)$:

$$
\begin{aligned}
y \oplus y' &= \gamma & &\text{proba } p \\
(y \oplus y')[\alpha] &= \gamma[\alpha] & &\text{proba} \approx p + 1/2(1-p) \\
z[\beta] &= y[\alpha] & &\text{proba } 1/2(1+\varepsilon) \\
z'[\beta] &= y'[\alpha] & &\text{proba } 1/2(1+\varepsilon) \\
(z \oplus z')[\beta] &= \gamma[\alpha] & &\text{proba } 1/2(1+p\varepsilon^2)
\end{aligned}
$$

- Distinguisher with complexity $\approx p^{-2}\varepsilon^{-4}$

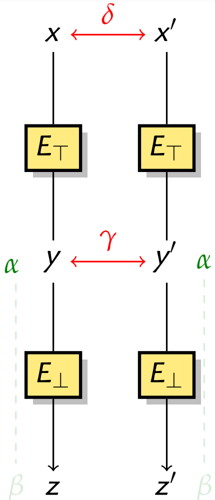


Differential-Linear Cryptanalysis

- Query a pair $(x, x' = x \oplus \delta)$:

$$
\begin{aligned}
y \oplus y' &= \gamma & &\text{proba } p \\
(y \oplus y')[\alpha] &= \gamma[\alpha] & &\text{proba} \approx 1/2(1+p) \\
z[\beta] &= y[\alpha] & &\text{proba } 1/2(1+\varepsilon) \\
z'[\beta] &= y'[\alpha] & &\text{proba } 1/2(1+\varepsilon) \\
(z \oplus z')[\beta] &= \gamma[\alpha] & &\text{proba } 1/2(1+p\varepsilon^2)
\end{aligned}
$$

- Distinguisher with complexity $\approx p^{-2}\varepsilon^{-4}$

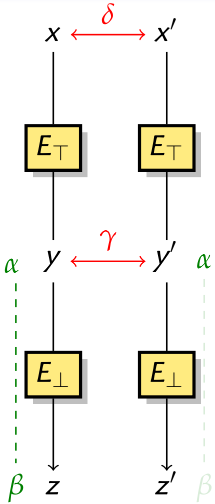


Differential-Linear Cryptanalysis

- Query a pair $(x, x' = x \oplus \delta)$:

$$
\begin{aligned}
y \oplus y' &= \gamma & &\text{proba } p \\
(y \oplus y')[\alpha] &= \gamma[\alpha] & &\text{proba} \approx 1/2(1+p) \\
z[\beta] &= y[\alpha] & &\text{proba } 1/2(1+\varepsilon) \\
z'[\beta] &= y'[\alpha] & &\text{proba } 1/2(1+\varepsilon) \\
(z \oplus z')[\beta] &= \gamma[\alpha] & &\text{proba } 1/2(1+p\varepsilon^2)
\end{aligned}
$$

- Distinguisher with complexity $\approx p^{-2}\varepsilon^{-4}$

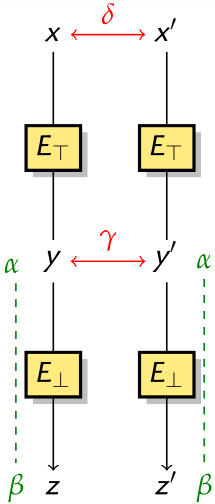


Differential-Linear Cryptanalysis

- Query a pair $(x, x' = x \oplus \delta)$:

$$\begin{aligned} y \oplus y' &= \gamma && \text{proba } p \\ (y \oplus y')[\alpha] &= \gamma[\alpha] && \text{proba} \approx 1/2(1+p) \\ z[\beta] &= y[\alpha] && \text{proba } 1/2(1+\varepsilon) \\ z'[\beta] &= y'[\alpha] && \text{proba } 1/2(1+\varepsilon) \\ (z \oplus z')[\beta] &= \gamma[\alpha] && \text{proba } 1/2(1+p\varepsilon^2) \end{aligned}$$

- Distinguisher with complexity $\approx p^{-2}\varepsilon^{-4}$

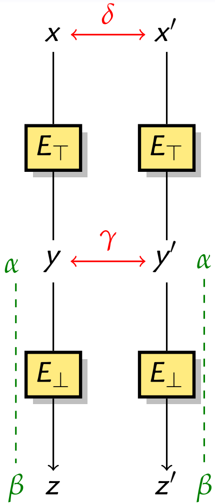


Differential-Linear Cryptanalysis

- Query a pair $(x, x' = x \oplus \delta)$:

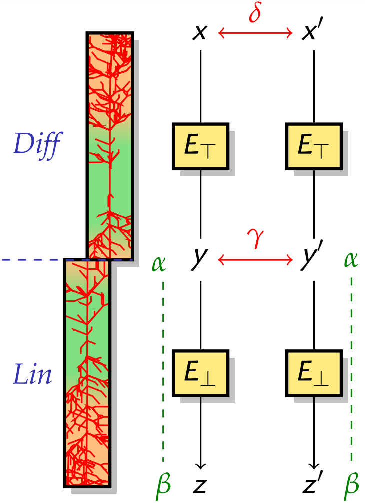
$$
\begin{aligned}
y \oplus y' &= \gamma && \text{proba } p \\
(y \oplus y')[\alpha] &= \gamma[\alpha] && \text{proba} \approx 1/2(1+p) \\
z[\beta] &= y[\alpha] && \text{proba } 1/2(1+\varepsilon) \\
z'[\beta] &= y'[\alpha] && \text{proba } 1/2(1+\varepsilon) \\
(z \oplus z')[\beta] &= \gamma[\alpha] && \text{proba } 1/2(1+p\varepsilon^2)
\end{aligned}
$$

- Distinguisher with complexity $\approx p^{-2}\varepsilon^{-4}$



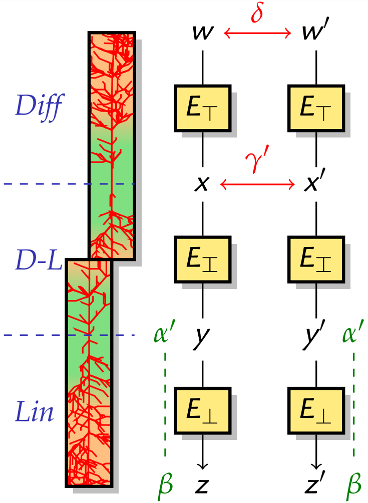
Improved Differential-Linear cryptanalysis

- Accurate analysis of differential-linear attack is hard [BLN, FSE '14]
  - Proba for wrong pair is not 1/2
  - Many differential trails with same δ
  - Many linear trails with same β
- Divide E in 3 parts
- Assuming there is a position with single bit γ', α'
  - Hourglass structure
- Eval. middle rounds experimentally
  - Small Differential-Linear
  - $\Pr[(E_{\bot}(x) \oplus E_{\bot}(x \oplus \gamma'))[\alpha'] = 1]$
- Try all single bit γ', α'



Improved Differential-Linear cryptanalysis

- Accurate analysis of differential-linear attack is hard [BLN, FSE '14]
  - Proba for wrong pair is not 1/2
  - Many differential trails with same δ
  - Many linear trails with same β
- Divide E in 3 parts
- Assuming there is a position with single bit γ', α'
  - Hourglass structure
- Eval. middle rounds experimentally
  - Small Differential-Linear
  - $\Pr\left[(E_{\top\!\!\bot}(x) \oplus E_{\top\!\!\bot}(x \oplus \gamma'))[\alpha'] = 1\right]$
- Try all single bit γ', α'



A 6-round distinguisher

Optimal choice for 6 rounds

- $E_\top$: 1 round, $p_\top = 2^{-5}$
  - $v_0[26], v_1[26], v_2[6, 23, 30], v_3[23, 30] \to v_2[22]$
- $E_\top$: 4 rounds, $\varepsilon_\top \approx 2^{-6.05}$
  - $v_2[22] \to v_2[16]$
- $E_\perp$: 1 round, $\varepsilon_\perp \approx 2^{-2.6}$
  - $v_2[16] \to v_0[5], v_1[23, 31], v_2[0, 8, 15], v_3[5]$

- Differential-linear bias $p_\top \cdot \varepsilon_\top \cdot \varepsilon_\perp^2 \approx 2^{-16.25}$
- Distinguisher with complexity $2/p_\top^2 \varepsilon_\top^2 \varepsilon_\perp^4 \approx 2^{33.5}$
- Implemented: analysis is verified

A 6-round distinguisher

Optimal choice for 6 rounds

- $E_\top$: 1 round, $p_\top = 2^{-5}$
  - $v_0[26], v_1[26], v_2[6, 23, 30], v_3[23, 30] \to v_2[22]$
- $E_\perp$: 4 rounds, $\varepsilon_\perp \approx 2^{-6.05}$
  - $v_2[22] \to v_2[16]$
- $E_\perp$: 1 round, $\varepsilon_\perp \approx 2^{-2.6}$
  - $v_2[16] \to v_0[5], v_1[23, 31], v_2[0, 8, 15], v_3[5]$

- Differential-linear bias $p_\top \cdot \varepsilon_\perp \cdot \varepsilon_\perp^2 \approx 2^{-16.25}$
- Distinguisher with complexity $2/p_\top^2 \varepsilon_\perp^2 \varepsilon_\perp^4 \approx 2^{33.5}$
- Implemented: analysis is verified

Partitionning

Main idea

- From distinguisher to key recovery
  - Last-round attack
  - Guess key bits, partial decryption
- Adapt technique to ARX ciphers

1. Guess some key bits
2. Deduce state bits, partition data according to state bits
3. Keep subsets with high expected bias

- Techniques inspired by:
  - Improved linear cryptanalysis of addition [Biham & Carmeli, SAC '14]
  - Salsa20 Probabilistic Neutral Bits [AFKMR, FSE '08]

Linear Cryptanalysis of Addition

Linear approximations of addition:

- $x_i = a_i \oplus b_i \oplus c_i$
- $c_i = \mathrm{MAJ}(a_{i-1}, b_{i-1}, c_{i-1})$
- $c_i = a_i$ with probability $3/4$ (bias $1/2$)

$$
\begin{array}{cccccc}
 & & c_i & & & \\
 & ? & a_i & a_{i-1} & ? & ? \\
+ & ? & b_i & b_{i-1} & ? & ? \\
\hline
 & ? & x_i & ? & ? & ?
\end{array}
$$

- Therefore $x_i \approx a_i \oplus b_i \oplus a_{i-1}$

Linear Cryptanalysis of Addition

Linear approximations of addition:

With partitionning

- If $(a_{i-1}, b_{i-1}) = (0, 0)$ there is no carry

$$\begin{array}{cccccc} & & & \overset{0}{\curvearrowleft} & & \\ & ? & a_i & 0 & ? & ? \\ + & ? & b_i & 0 & ? & ? \\ \hline & ? & x_i & ? & ? & ? \end{array}$$

- Therefore $x_i = a_i \oplus b_i$

- If $(a_{i-1}, b_{i-1}) = (1, 1)$ there is always a carry

$$\begin{array}{cccccc} & & & \overset{1}{\curvearrowleft} & & \\ & ? & a_i & 1 & ? & ? \\ + & ? & b_i & 1 & ? & ? \\ \hline & ? & x_i & ? & ? & ? \end{array}$$

- Therefore $x_i = a_i \oplus b_i \oplus 1$

- We throw out one half of the data [Biham & Carmeli, SAC '14]
- But the distinguisher requires 4 times less data

Linear Cryptanalysis of Addition

Linear approximations of addition:

With partitionning

- If $(a_{i-1}, b_{i-1}) = (0, 0)$ there is no carry

$$\begin{array}{cccccc}
 & & & 0 & 0 & \\
 & ? & a_i & 0 & 0 & ? \\
+ & ? & b_i & 1 & 0 & ? \\
\hline
 & ? & x_i & ? & ? & ?
\end{array}$$

- Therefore $x_i = a_i \oplus b_i$

- If $(a_{i-1}, b_{i-1}) = (1, 1)$ there is always a carry

$$\begin{array}{cccccc}
 & & & 1 & 1 & \\
 & ? & a_i & 0 & 1 & ? \\
+ & ? & b_i & 1 & 1 & ? \\
\hline
 & ? & x_i & ? & ? & ?
\end{array}$$

- Therefore $x_i = a_i \oplus b_i \oplus 1$

- We throw out one fourth of the data [New]
- But the distinguisher requires 4 times less data

Partitionning for Linear Cryptanalysis

- Further improvements
  - Guess more bits
  - Several active bits
  - Predict bits of the next addition
  - But it gets messy...

Experimental approach

- Identify candidate bits (by hand)
- Collect data:
  - Filter according to candidate bits
  - Measure bias
- Build vector of bias, and remove least useful bits
  - Symmetries allow the reduce the number of filtering bits

Partitionning for Differential Cryptanalysis

- Partitionning can also be used in the differential side

Main steps

1. Use structures and multiple differential
2. Guess key bits
3. Build pairs according to key guess

- Small gain for plain differential
- More interresting for differential-linear
- Experimental approach to deal with complex cases

Improved 6-round attack

- Partitioning on the linear side
  - 8 control bits
  - Gain a factor 2^8
- Partitioning on the differential side
  - Structures with 2^3 differences
  - 5 differential control bits
  - Gain a factor 36

- Data complexity: 2^{24} pairs (vs $2^{33.5}$)
- 13-bit subkey
  - 6-bit gain: average key rank 64
  - Repeat with another trail for more key bits...

- FFT to reduce the time complexity
- Time complexity: $2^{28.6}$ (elementary operations)
- Fully implemented

Time complexity

Attack steps (following multiple-linear cryptanalysis [BCQ04])

1. Filtering bits define subsets s
2. For each subset s, observed imbalance $\hat{\varepsilon}[s]$ (using counters).
3. For each subset s, key candidate k, expected imbalance $\varepsilon_k[s]$.
4. Compute distance $L(k) = \sum_s (\hat{\varepsilon}[s] - \varepsilon_k[s])^2$
5. Enumerate keys with smaller distance

- The key is only xored at the beginning and at the end

$$\varepsilon_k[s] = \varepsilon_0[s \oplus \phi(k)], \quad \text{where } \phi(k_{\text{diff}}, k_{\text{lin}}) = (0, k_{\text{lin}}, k_{\text{diff}}, k_{\text{diff}})$$

$$L(k) = \sum_s \hat{\varepsilon}[s]^2 + \sum_s \varepsilon_0[s \oplus \phi(k)]^2 - 2 \sum_s \hat{\varepsilon}[s]\varepsilon_0[s \oplus \phi(k)]$$

- $\sum_s \hat{\varepsilon}[s]\varepsilon_0[s \oplus \phi(k)]$ is a convolution: Compute with FFT [CSQ07]

Improved 6-round attack

- Partitioning on the linear side
  - 8 control bits
  - Gain a factor 2^8
- Partitioning on the differential side
  - Structures with 2^3 differences
  - 5 differential control bits
  - Gain a factor 36

- Data complexity: 2^{24} pairs (vs $2^{33.5}$)
- 13-bit subkey
  - 6-bit gain: average key rank 64
  - Repeat with another trail for more key bits...

- FFT to reduce the time complexity
- Time complexity: $2^{28.6}$ (elementary operations)
- Fully implemented

A 7-round distinguisher

The attack can be extended to 7 rounds

Optimal choice for 7 rounds

- $E_\top$: 1.5 rounds, $p_\top = 2^{-17}$
  - $v_0[8,18,21,30], v_1[8,13,21,26,30], v_2[3,21,26], v_3[21,26,27] \xrightarrow{E_\top} v_0[31]$
- E_I: 4 rounds, $\varepsilon_\text{I} = 2^{-6.1}$
  - $v_0[31] \xrightarrow{E_\text{I}} v_2[20]$
- $E_\perp$: 1.5 rounds, $\varepsilon_\perp = 2^{-7.6}$
  - $v_2[20] \xrightarrow{E_\perp} v_0[0,15,16,25,29], v_1[7,11,19,26], v_2[2,10,19,20,23,28], v_3[0,25,29]$

- Differential-linear bias: $p_\top \cdot \varepsilon_\text{I} \cdot \varepsilon_\perp^2 \approx 2^{-38.3}$
- Distinguisher with complexity $2/p_\top^2 \varepsilon_\text{I}^2 \varepsilon_\perp^4 \approx 2^{77.6}$

Improved 7-round attack

- Improved 7 round attack
- Partitioning on the linear side
  - 19 control bits
  - Gain a factor 2^{21}
- Partitioning on the differential side
  - Structures with 2^9 differences
  - 14 differential control bits
  - Gain a factor $4374 \approx 2^{12.1}$

- Data complexity: 2^{47} pairs (vs $2^{77.6}$)
- 33-bit subkey
  - theoretical gain 6.3 bits
  - Repeat with another trail for more key bits...
- FHT to reduce the time complexity
- Time complexity: 2^{67} (elementary operations)

Key-recovery attacks against Chaskey

	Rounds	Data	Time	Gain
Differential-Linear	6	2^{35}	2^{35}	1 bit
Differential-Linear with partitioning	6	2^{25}	$2^{28.6}$	6 bits
Differential-Linear	7	2^{78}	2^{78}	1 bit
Differential-Linear with partitioning	7	2^{48}	2^{67}	6 bits
Security Claim	*8*	2^{48}	2^{80}	

- 6-round attacks implemented
- Security margin of Chaskey rather slim (7/8 rounds broken)
- New Chaskey variant with 12-round

Key-recovery attacks against Chaskey

	Rounds	Data	Time	Gain
Differential-Linear	6	2^{35}	2^{35}	1 bit
Differential-Linear with partitioning	6	2^{25}	$2^{28.6}$	6 bits
Differential-Linear	7	2^{78}	2^{78}	1 bit
Differential-Linear with partitioning	7	2^{48}	2^{67}	6 bits
Security Claim	*8*	2^{48}	2^{80}	

- Differential-Linear attacks quite efficient for ARX designs
- Improvements: roughly half round at top and bottom for free
  1. Divide in three section, evaluate experimentally middle section
  2. Use partitionning to reduce data complexity
  3. Use FFT to reduce time complexity