Analyzing Permutations for AES-like Ciphers: Understanding ShiftRows

Christof Beierle[1] Philipp Jovanovic[2] Martin M. Lauridsen[3]
Gregor Leander[1] Christian Rechberger[3]

[1]Horst Görtz Institute for IT Security, Ruhr-Universität Bochum, Germany

[2]Fakultät für Informatik und Mathematik, Universität Passau, Germany

[3]DTU Compute, Technical University of Denmark, Denmark

CT-RSA 2015
San Francisco

Outline

Motivation

Current status

- AES-like designs are **very frequent in practice**:
  LED, mCrypton, PRINCE, ECHO, Grøstl, LANE, PHOTON, PAEQ, PRIMATEs, Prøst, STRIBOB, ...

Motivation

Current status

- AES-like designs are **very frequent in practice**:
  LED, mCrypton, PRINCE, ECHO, Grøstl, LANE, PHOTON, PAEQ, PRIMATEs, Prøst, STRIBOB, ...
- **Crucial:** Understanding properties of diffusion and resistance to differential/linear attacks

Motivation

Current status

- AES-like designs are **very frequent in practice**:
  LED, mCrypton, PRINCE, ECHO, Grøstl, LANE, PHOTON, PAEQ, PRIMATEs, Prøst, STRIBOB, ...
- **Crucial:** Understanding properties of diffusion and resistance to differential/linear attacks
- `MixColumns`-like step
  - Well understood: Pick sufficiently high branch number

Motivation

Current status

- AES-like designs are **very frequent in practice**: LED, mCrypton, PRINCE, ECHO, Grøstl, LANE, PHOTON, PAEQ, PRIMATEs, Prøst, STRIBOB, ...
- **Crucial:** Understanding properties of diffusion and resistance to differential/linear attacks
- `MixColumns`-like step
  - Well understood: Pick sufficiently high branch number
- `ShiftRows`-like step:
  - **Unclear; no structured approach**
  - **Choice remains ad-hoc**

Motivation

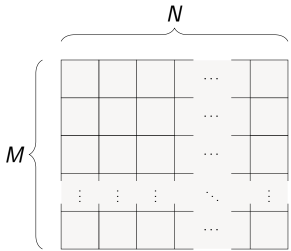
Our goal

Contribute to the understanding of picking **optimal** `ShiftRows`-like operations for **generalized** AES-like ciphers

Notation and definitions

AES-like cipher

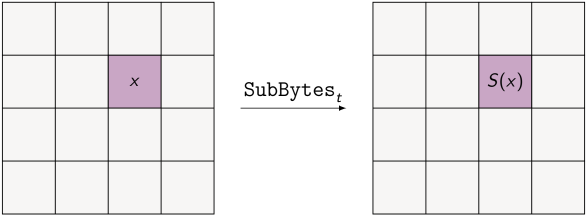
- State of size $M \times N$ of m-bit words



- Round t equals

$$R_t = \texttt{AddRoundKey}_t \circ \texttt{Permute}_{\pi_t} \circ \texttt{MixColumns}_t \circ \texttt{SubBytes}_t$$

AES-like cipher: $\texttt{SubBytes}_t$

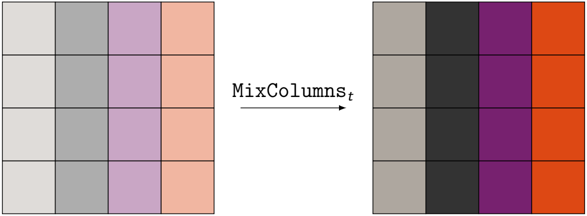


- Substitutes each state word according to one or more S-boxes

$$S^t_{i,j} : \mathbb{F}_2^m \to \mathbb{F}_2^m.$$

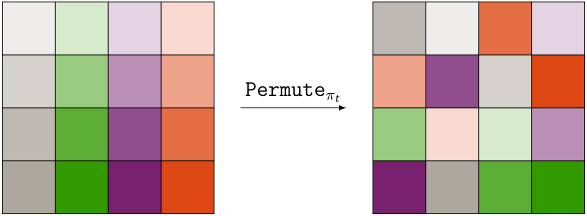
- Allow independent S-boxes for each word $x_{i,j}$ in each round

AES-like cipher: $\texttt{MixColumns}_t$



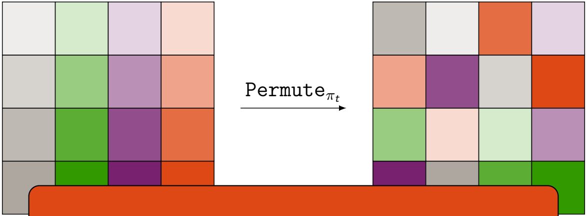
- Left-multiplies column j in round t by an $M \times M$ matrix M_j^t over $GF(2^m)$
- Allow independent M_j^t for each column in each round

AES-like cipher: $\texttt{Permute}_{\pi_t}$



- Shuffles state words according to a permutation π_t on $\mathbb{Z}_M \times \mathbb{Z}_N$
- Assume independent permutations π_t in each round
- We say $\pi = (\pi_0, \ldots, \pi_{T-1})$ is a **permutation sequence** for the T-round AES-like cipher

AES-like cipher: $\texttt{Permute}_{\pi_t}$

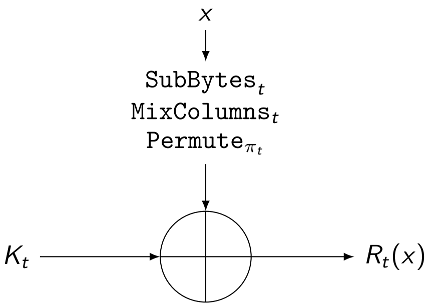


Our focus in this work:

optimize the choice of π

- Shuffles ... $\mathbb{Z}_M \times \mathbb{Z}_N$
- Assume ...
- We say ... e for the T-round AES-like cipher

AES-like cipher: $\texttt{AddRoundKey}_t$



- A round key is added to the state using $\oplus$ in each round
- Does not affect the properties we investigate, thus not considered further!

Difference and activity pattern

Difference

A (non-zero) **difference** is a value $X \in (\mathbb{F}_2^m)^{M \times N} \backslash \{0\}$

Difference and activity pattern

Difference

A (non-zero) **difference** is a value $X \in (\mathbb{F}_2^m)^{M \times N} \setminus \{0\}$

Activity pattern

For a difference X, let the **activity pattern** $\tilde{X}$ be defined s.t.

$$\tilde{X}_{i,j} = \begin{cases} 1 & X_{i,j} \neq 0 \\ 0 & X_{i,j} = 0 \end{cases},$$

Difference and activity pattern

Difference

A (non-zero) **difference** is a value $X \in (\mathbb{F}_2^m)^{M \times N} \setminus \{0\}$

Activity pattern

For a difference X, let the **activity pattern** $\tilde{X}$ be defined s.t.

$$\tilde{X}_{i,j} = \begin{cases} 1 & X_{i,j} \neq 0 \\ 0 & X_{i,j} = 0 \end{cases},$$

for example

X

00	00	CA
F2	00	24

$\tilde{X}$

0	0	1
1	0	1

Trails

Trail

For an AES-like cipher, a T-round **trail** is a $(T+1)$-tuple of differences

Trails

Trail

For an AES-like cipher, a T-round **trail** is a $(T+1)$-tuple of differences

Trail weight

The **trail weight** of $X = (X^0, \ldots, X^T)$ is defined as

$$\sum_{t=0}^{T-1} \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} \tilde{X}_{i,j}^{t}$$

Branch number

Differential branch number

For a linear automorphism $\theta : (\mathbb{F}_2^m)^M \rightarrow (\mathbb{F}_2^m)^M$, we define the (differential) **branch number** B_θ as the minimum number of non-zero words, in the input- and output differences $(X \oplus Y)$ respectively $(\theta(X) \oplus \theta(Y))$, when taken across all pairs of inputs $X, Y \in \mathbb{F}_2^m$.

Branch number

Differential branch number

For a linear automorphism $\theta : (\mathbb{F}_2^m)^M \to (\mathbb{F}_2^m)^M$, we define the (differential) **branch number** B_θ as the minimum number of non-zero words, in the input- and output differences $(X \oplus Y)$ respectively $(\theta(X) \oplus \theta(Y))$, when taken across all pairs of inputs $X, Y \in \mathbb{F}_2^m$.

For an AES-like cipher

The **branch number** for an AES-like cipher is B_θ if and only if it is the **minimum of the branch numbers** obtained by left multiplication by any M_j^t.

Problem modeling

We are interested in determining

$$\max_{\pi} \min_{\text{trail } X} \textbf{weight}(X)$$

for a T-round AES-like cipher of dimensions $M \times N$.

Problem modeling

We are interested in determining

$$\max_{\pi} \min_{\text{trail } X} \mathbf{weight}(X)$$

for a T-round AES-like cipher of dimensions $M \times N$.

In our modeling of the problem, we consider the following as black-box operations

- $\texttt{SubBytes}_t$
- $\texttt{MixColumns}_t$, under the requirement of a specific branch number

Bounds and trail-optimality

Tightly guaranteed active S-boxes

Consider an AES-like cipher with branch number B_θ. We say that a permutation sequence π **tightly guarantees** k active S-boxes, denoted $\pi \xrightarrow{B_\theta} k$, if and only if, when using π for the `Permute` operation,

- There exists a valid trail of weight k and
- There is no valid trail of positive weight $k' < k$.

Bounds and trail-optimality

Tightly guaranteed active S-boxes

Consider an AES-like cipher with branch number B_θ. We say that a permutation sequence π **tightly guarantees** k active S-boxes, denoted $\pi \xrightarrow{B_\theta} k$, if and only if, when using π for the `Permute` operation,

- There exists a valid trail of weight k and
- There is no valid trail of positive weight $k' < k$.

Trail-optimality

A permutation sequence π is said to be **trail-optimal** if and only if there exists no $\pi' \neq \pi$ such that $\pi' \xrightarrow{B_\theta} k'$ where $k' > k$

Equivalences for permutation sequences π

Defining equivalence

The goal

Classify permutation sequences π **incurring the same bound** on the trail weight

$$\Pi(k) = \left\{ \pi = (\pi_0, \ldots, \pi_{T-1}) \mid \pi \xrightarrow{B_\theta} k \right\},$$

and thus **reducing the search space** for a brute-force approach

Defining equivalence

The goal

Classify permutation sequences π **incurring the same bound** on the trail weight

$$\Pi(k) = \left\{ \pi = (\pi_0, \ldots, \pi_{T-1}) \mid \pi \xrightarrow{B_\theta} k \right\},$$

and thus **reducing the search space** for a brute-force approach

Equivalence of permutation sequences

Informally, we say that two permutation sequences π, π' are **equivalent**, denoted $\pi \sim \pi'$, if and only if $\pi \xrightarrow{B_\theta} k \Leftrightarrow \pi' \xrightarrow{B_\theta} k$

- Note: stronger notion of equivalence in paper

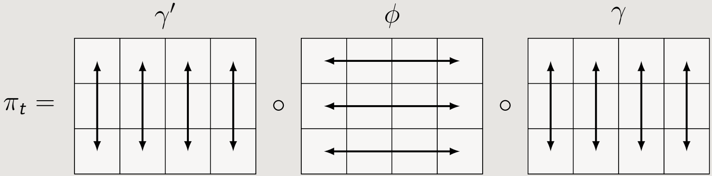
Structure of a state permutation

Lemma

Any permutation π_t on the words of an $M \times N$ state can be written as

$$\pi_t = \gamma' \circ \phi \circ \gamma,$$

where γ, γ' permute inside each column and ϕ permutes inside each row



(Thanks to John Steinberger for aiding in this proof)

Reduction to permuting in the rows

Lemma

Let π be a permutation sequence for an AES-like cipher and let γ, γ' be **arbitrary** permutations inside the columns of the state. Then

$$(\pi_0, \ldots, \pi_t, \ldots, \pi_{T-1}) \sim (\pi_0, \ldots, \gamma' \circ \pi_t \circ \gamma, \ldots, \pi_{T-1})$$

holds for all $t = 0, \ldots, T-1$.

Reduction to permuting in the rows

Lemma

Let π be a permutation sequence for an AES-like cipher and let γ, γ' be **arbitrary** permutations inside the columns of the state. Then

$$(\pi_0, \ldots, \pi_t, \ldots, \pi_{T-1}) \sim (\pi_0, \ldots, \gamma' \circ \pi_t \circ \gamma, \ldots, \pi_{T-1})$$

holds for all $t = 0, \ldots, T-1$.

Theorem

Given **any** permutation sequence π, one can construct π' s.t.

- $\pi \sim \pi'$ and
- The permutations of π' permute only the words within each row

Reduction to permuting in the rows

Lemma

Let π be a permutation sequence for an AES-like cipher and let γ, γ' be **arbitrary** permutations inside the columns of the state. Then

$$(\pi_0, \ldots, \pi_t, \ldots, \pi_{T-1}) \sim (\pi_0, \ldots, \gamma' \circ \pi_t \circ \gamma, \ldots, \pi_{T-1})$$

holds for all $t = 0, \ldots, T-1$.

Theorem

Given **any** permutation sequence π, one can construct π' s.t.

- $\pi \sim \pi'$ and
- The permutations of π' permute only the words within each row

- Search space (per round) reduced from $(M \cdot N)!$ to $(N!)^M$

Rotation matrices

In the following, we restrict ourselves to **rotation matrices**

- $\texttt{Permute}_{\pi_t}$ becomes $\texttt{ShiftRows}_{\sigma_t}$
- Much nicer for implementations

Rotation matrices

In the following, we restrict ourselves to **rotation matrices**

- $\texttt{Permute}_{\pi_t}$ becomes $\texttt{ShiftRows}_{\sigma_t}$
- Much nicer for implementations

Rotation matrix

A **rotation matrix** for an an AES-like cipher is a $\rho \times M$ matrix σ over $\mathbb{Z}_N$,

$$\sigma = \begin{pmatrix} \sigma_{0,0} & \cdots & \sigma_{0,M-1} \\ \sigma_{1,0} & \cdots & \sigma_{1,M-1} \\ \vdots & \ddots & \vdots \\ \sigma_{\rho-1,0} & \cdots & \sigma_{\rho-1,M-1} \end{pmatrix}$$

Rotate row j of the state in round t by $\sigma_{i,j}$ where $t \equiv i \bmod \rho$

Rotation matrices

In the following, we restrict ourselves to **rotation matrices**

- $\texttt{Permute}_{\pi_t}$ becomes $\texttt{ShiftRows}_{\sigma_t}$
- Much nicer for implementations

Rotation matrix

A **rotation matrix** for an an AES-like cipher is a $\rho \times M$ matrix σ over $\mathbb{Z}_N$,

$$\sigma = \begin{pmatrix} \sigma_{0,0} & \cdots & \sigma_{0,M-1} \\ \sigma_{1,0} & \cdots & \sigma_{1,M-1} \\ \vdots & \ddots & \vdots \\ \sigma_{\rho-1,0} & \cdots & \sigma_{\rho-1,M-1} \end{pmatrix}$$

Rotate row j of the state in round t by $\sigma_{i,j}$ where $t \equiv i \bmod \rho$

- Search space (per round) reduced from $(N!)^M$ to N^M

Rotation matrices: Example

Consider an AES-like cipher of dimension 3×4 with $\rho = 2$ using

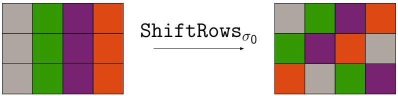
$$\sigma = \begin{pmatrix} 0 & 3 & 1 \\ 1 & 0 & 2 \end{pmatrix}$$

Rotation matrices: Example

Consider an AES-like cipher of dimension 3×4 with $\rho = 2$ using

$$\sigma = \begin{pmatrix} 0 & 3 & 1 \\ 1 & 0 & 2 \end{pmatrix}$$

When t is even

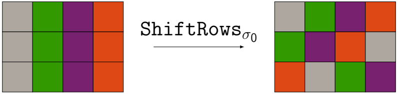


Rotation matrices: Example

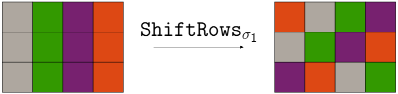
Consider an AES-like cipher of dimension 3×4 with $\rho = 2$ using

$$\sigma = \begin{pmatrix} 0 & 3 & 1 \\ 1 & 0 & 2 \end{pmatrix}$$

When t is even



When t is odd



Equivalences for rotation matrices σ

Equivalences for rotation matrices

Lemma: Re-arranging row entries of σ

Let σ be a rotation matrix and let $\vartheta_0, \ldots, \vartheta_{\rho-1}$ denote permutations on each of the ρ rows of σ. Define $\sigma'_t = \vartheta_t(\sigma_t)$ for all t. Then $\sigma \sim \sigma'$.

Equivalences for rotation matrices

Lemma: Re-arranging row entries of σ

Let σ be a rotation matrix and let $\vartheta_0, \ldots, \vartheta_{\rho-1}$ denote permutations on each of the ρ rows of σ. Define $\sigma'_t = \vartheta_t(\sigma_t)$ for all t. Then $\sigma \sim \sigma'$.

Lemma: Row-wise constant addition

Let σ be a rotation matrix and let $c_0, \ldots, c_{\rho-1} \in \mathbb{Z}_N$. Define $\sigma'_t = \sigma_t + c_t \bmod N$ for all t. Then $\sigma \sim \sigma'$.

Rotation matrix normalized form

Theorem: Equivalences

Let σ be any rotation matrix. Then there exists an equivalent rotation matrix σ', s.t.

1. Each row σ'_t is ordered lexicographically
2. Each $\sigma'_{t,0} = 0$, i.e. 1^{st} element in each row is zero
3. $\sigma'_{t,1} \leq N/2$ for all t, i.e. 2^{nd} element in each row is $\leq N/2$

Rotation matrix normalized form

Theorem: Equivalences

Let σ be any rotation matrix. Then there exists an equivalent rotation matrix σ', s.t.

1. Each row σ'_t is ordered lexicographically
2. Each $\sigma'_{t,0} = 0$, i.e. 1^{st} element in each row is zero
3. $\sigma'_{t,1} \leq N/2$ for all t, i.e. 2^{nd} element in each row is $\leq N/2$

Normal form

We define the **rotation matrix normal form** to satisfy 1-3, and heuristically also require that

- When N is even, σ' should have at **least one odd entry**
- The elements in each row σ'_t are **distinct**

- Search space (for full σ) reduced to $\left[\frac{N}{2} \cdot \binom{N/2}{M-2}\right]^{\rho}$

Experiments

Experiments

Goal

Determine **optimal rotation matrices** σ for a range of parameters (M, N, T, ρ)

Experiments

Goal

Determine **optimal rotation matrices** σ for a range of parameters (M, N, T, ρ)

Approach

Given a fixed rotation matrix σ, we

- Focus on the MDS case, i.e. $B_\theta = M + 1$
- Formulate a MIP problem of determining k s.t. $\sigma \xrightarrow{M+1} k$
- Combine **brute-forcing the normal forms** with solving the MIP model using CPLEX

Findings

Rounds	Rijndael-192	Rijndael-256	PRIMATEs-80	Prøst-256
5	–	–	54/56	–
6	42/45	50/55	–	85/90†
7	46/48	–	–	96/111†
8	50/57	–	–	–
10	–	85/90	–	–
12	87/90	105/111	–	–

Increased ρ from 1 to 2

- Many more results in paper
- † Searched only among diffusion-optimal solutions

Conclusion and open problems

What we did

- Took steps to analyze the problem of picking the best permutation for AES-like ciphers
- Focus on rotations as in `ShiftRows` due to implementation characteristics
- Reduced to normal form and combined with optimization using MIP
- Improve parameters for some existing designs

Conclusion and open problems

What we did

- Took steps to analyze the problem of picking the best permutation for AES-like ciphers
- Focus on rotations as in `ShiftRows` due to implementation characteristics
- Reduced to normal form and combined with optimization using MIP
- Improve parameters for some existing designs

Open problems

- Formulating optimization problem with trail-optimal σ as decision variable (bi-level optimization)
- Analysis w.r.t. combining diffusion-optimality with trail-optimality

Thanks for you attention

Questions?

RSAConference2015

San Francisco | April 20-24 | Moscone Center

CHANGE

Challenge today's security thinking

SESSION ID: CRYP-T08

Improved Attacks on Reduced-Round Camellia-128/192/256

Xiaoyang Dong[1], Leibo li[1], Keting Jia[2] and Xiaoyun Wang[1,3]*

[1]Key Lab of Cryptologic Technology and Information Security, Ministry of Education,Shandong University, China, Ph.D Student

[2]Department of Computer and Technology, Tsinghua University

[3]Institute for Advanced Study, Tsinghua University

#RSAC

Outline

- Introduction
  - Description of Camellia
- Key-Dependent Attacks
  - Key-dependent 8-round differentials
  - Key-dependent multiple differential attack for 10-round Camellia-128
- Meet-in-the-Middle Attacks
  - New 7-round property and 12-round attack for Camellia-192
  - New 8-round property and 13-round attack for Camellia-256

RSAConference2015

San Francisco | April 20-24 | Moscone Center

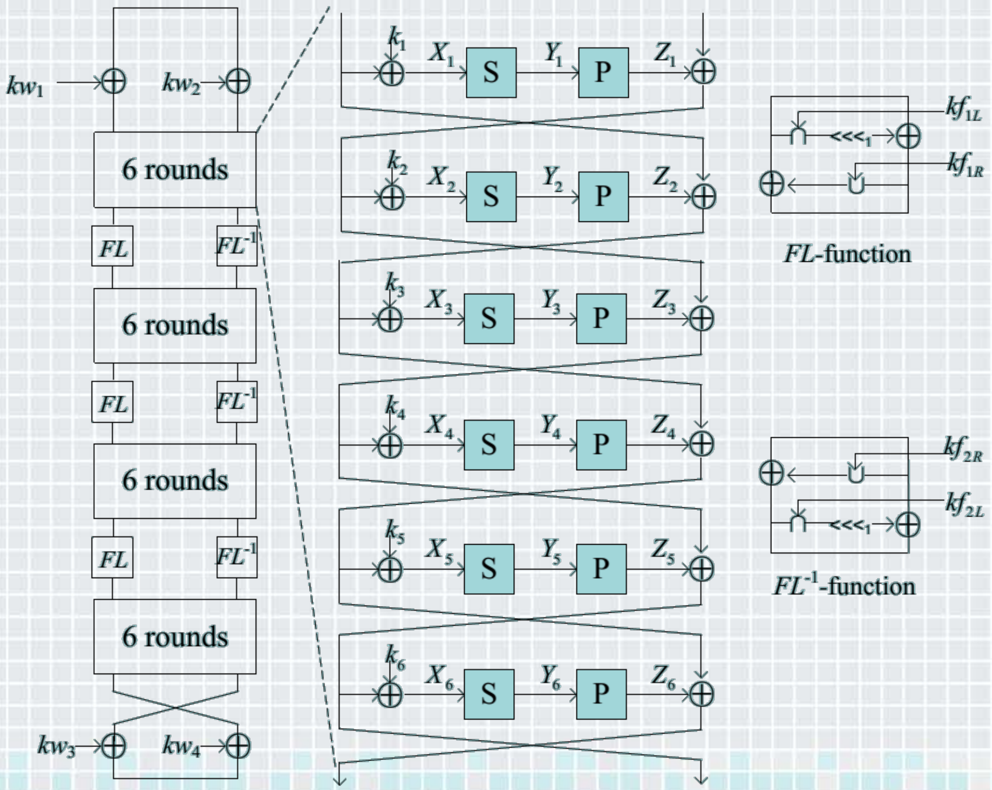
#RSAC

Description of Camellia

Description of Camellia

- In 2000, Proposed by NTT and Mitsubishi.
- Adopted as an international standard ISO/IEC 18033-3, NESSIE block cipher portfolio, as well as an e-Government recommended cipher by CRYPTREC project
- Basic Information
  - Block Size: 128
  - Key Sizes: 128/192/256(denoted as Camellia-128/192/256)
  - Number of Rounds: 18/24/24 for Camellia-128/192/256
  - Structure: Feistel structure with key-dependent *FL* layers

Description of Camellia



RSA Conference 2015

San Francisco | April 20-24 | Moscone Center

#RSAC

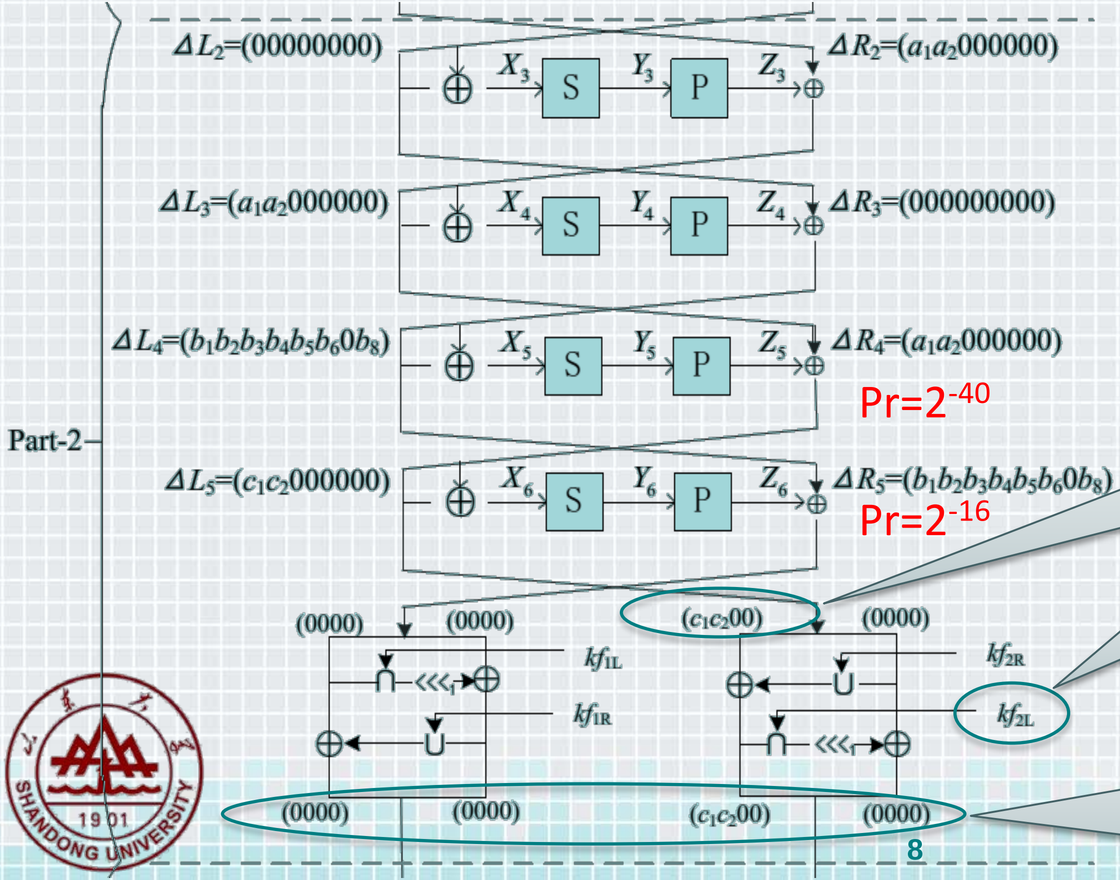
Key-Dependent Attacks

Key-dependent truncated differentials

- 4-Round Truncated Differentials with probability 2^{-56}

Active S-boxes: $0 \to 2 \to 7 \to 2$			
Case-1	Case-2	Case-3	Case-4
$(00000000, \ast\ast000000)$	$(00000000, 0\ast\ast00000)$	$(00000000, \ast00\ast0000)$	$(00000000, 00\ast\ast0000)$
$(\ast\ast000000, 00000000)$	$(0\ast\ast00000, 00000000)$	$(\ast00\ast0000, 00000000)$	$(00\ast\ast0000, 00000000)$
$(\ast\ast\ast\ast\ast\ast\ast0\ast, \ast\ast000000)$	$(\ast\ast\ast\ast\ast\ast0, 0\ast\ast00000)$	$(\ast\ast\ast\ast\ast0\ast\ast, \ast00\ast0000)$	$(\ast\ast\ast\ast0\ast\ast\ast, 00\ast\ast0000)$
$(\ast\ast000000, \ast\ast\ast\ast\ast\ast0\ast)$	$(0\ast\ast00000, \ast\ast\ast\ast\ast\ast\ast0)$	$(\ast00\ast0000, \ast\ast\ast\ast\ast0\ast\ast)$	$(00\ast\ast0000, \ast\ast\ast\ast0\ast\ast\ast)$
$(00000000, \ast\ast000000)$	$(00000000, 0\ast\ast00000)$	$(00000000, \ast00\ast0000)$	$(00000000, 00\ast\ast0000)$

Key-dependent truncated differentials



1. Choose some value for c_1,c_2
2. Choose a key subset correspondingly
3. Output difference determined

If we choose Some fixed value, like c_1=0x40,c_2=0x80

kf2L{2}=0,kf2L{9}=0 correspondingly

Output difference can be determined with $Pr=2^{-40}\times 2^{-16}\times 2^{-16}=2^{-72}$

Key subsets

i	(c_1^i, c_2^i)	i	(c_1^i, c_2^i)	i	(c_1^i, c_2^i)	i	(c_1^i, c_2^i)	i	(c_1^i, c_2^i)	i	(c_1^i, c_2^i)	i	(c_1^i, c_2^i)	i	(c_1^i, c_2^i)
1	01, 02	8	02, 01	15	04, 01	22	08, 01	29	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
2	01, 04	9	02, 04	16	04, 02	23	08, 02	30	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
3	01, 08	10	02, 08	17	04, 08	24	08, 04	31	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
4	01, 10	11	02, 10	18	04, 10	25	08, 10	32	10, 08	[illegible]	20, 08	46	40, 08	53	80, 08
5	01, 20	12	02, 20	19	04, 20	26	08, 20	33	10, 20	40	20, 10	47	40, 10	54	80, 10
6	01, 40	13	02, 40	20	04, 40	27	08, 40	34	10, 40	41	20, 40	48	40, 20	55	80, 20
7	01, 80	14	02, 80	21	04, 80	28	08, 80	35	10, 80	42	20, 80	49	40, 80	56	80, 40

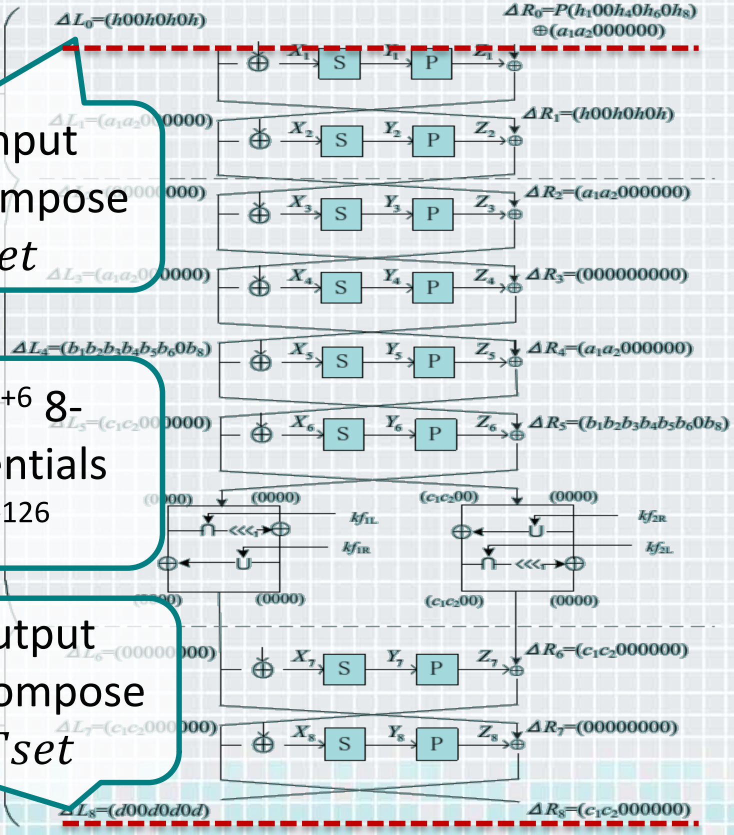
There are 56 such (c_1, c_2).

56 pairs of (c_1, c_2);
4 cases differentials.
Produce 224 key
Dependent differentials.

$$KDset_i^1 = \{K | kf_{2L} = (\neg c_1^i \wedge *, \neg c_2^i \wedge *, *, *), * \in F_2^8\},$$
$$KDset_i^2 = \{K | kf_{2L} = (*, \neg c_1^i \wedge *, \neg c_2^i \wedge *, *), * \in F_2^8\},$$
$$KDset_i^3 = \{K | kf_{2L} = (\neg c_1^i \wedge *, *, *, \neg c_2^i \wedge *), * \in F_2^8\},$$
$$KDset_i^4 = \{K | kf_{2L} = (*, *, \neg c_1^i \wedge *, \neg c_2^i \wedge *), * \in F_2^8\}.$$

Produce 224 key subsets as well.
And denote the other keys as *RK*set

Key-dependent 8-round differentials



About 2^{37} input differences compose the $\Delta INset$

Produce 2^{37+6} 8-round differentials with $Pr=2^{-126}$

About2^{6} output differences compose the $\Delta OUTset$

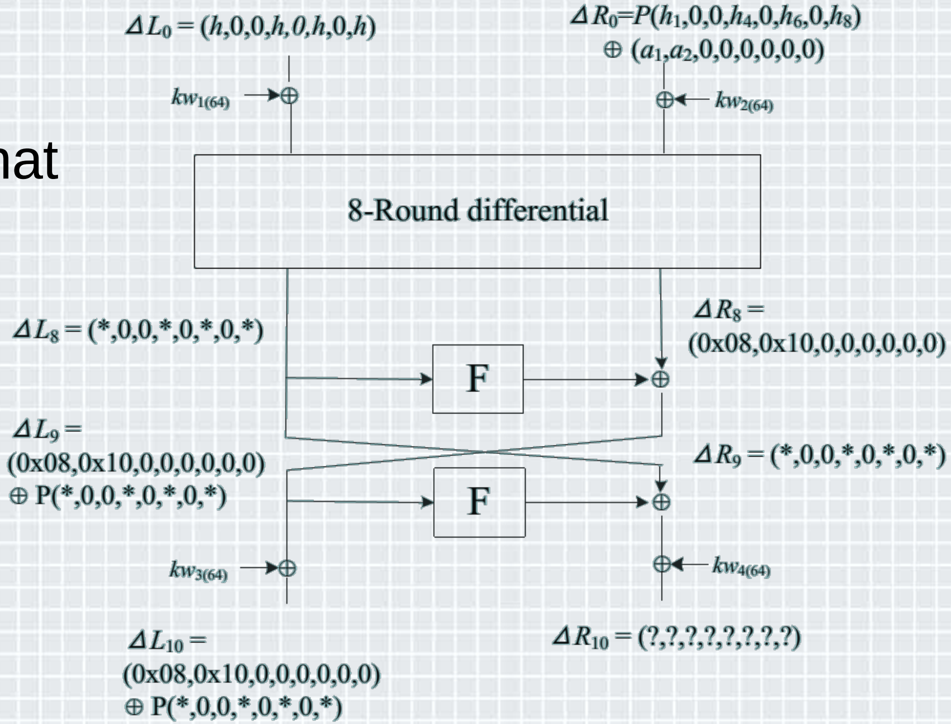
Extend two rounds

4-round key-dependent truncated differential with FL/FL^{-1} layer

Append two rounds

Key-dependent attack on10-round Camellia-128

- We launch an example attack
- Choose 8-round differentials that
  - c_1=0x08,c_2=0x10
  - Cover $KDset^1_{32}$
- Append 2 rounds on the bottom



Key-dependent attack on10-round Camellia-128

- Data collection:
  - Structure:$L_0 = (\alpha_1, x_1, x_2, \alpha_1, x_3, \alpha_1, x_4, \alpha_1)$ and

  $R_0 = P(\alpha_2, x_5, x_6, \alpha_3, x_7, \alpha_4, x_8, \alpha_5) \oplus (\alpha_6, \alpha_7, x_9, x_{10}, x_{11}, x_{12}, x_{13}, x_{14})$, where x_i are constant, α_j take all values.
  - Encrypt to get 2^{56} ciphertext C, store them indexed by $P^{-1}(C_L)[1,4,6,8]$,
  - Construct pairs by choosing C indexed by $P^{-1}(C_L)[1,4,6,8]$ and C' indexed by $P^{-1}(C_L)[1,4,6,8] \oplus P^{-1}(0x08, 0x10, 0,0,0,0,0,0)[1,4,6,8]$.
  - Choose 2^{33} structures, $2^{33+111-32}=2^{112}$ pairs constructed.
  - Delete the pairs whose input difference do not belong to $\Delta INset$, and about 2^{93} pairs left

Key-dependent attack on10-round Camellia-128

- Key Guessing
  - For each pair and each possible ΔR_9, do

A. Deduce 64-bit key $kw_3 \oplus k_{10}$

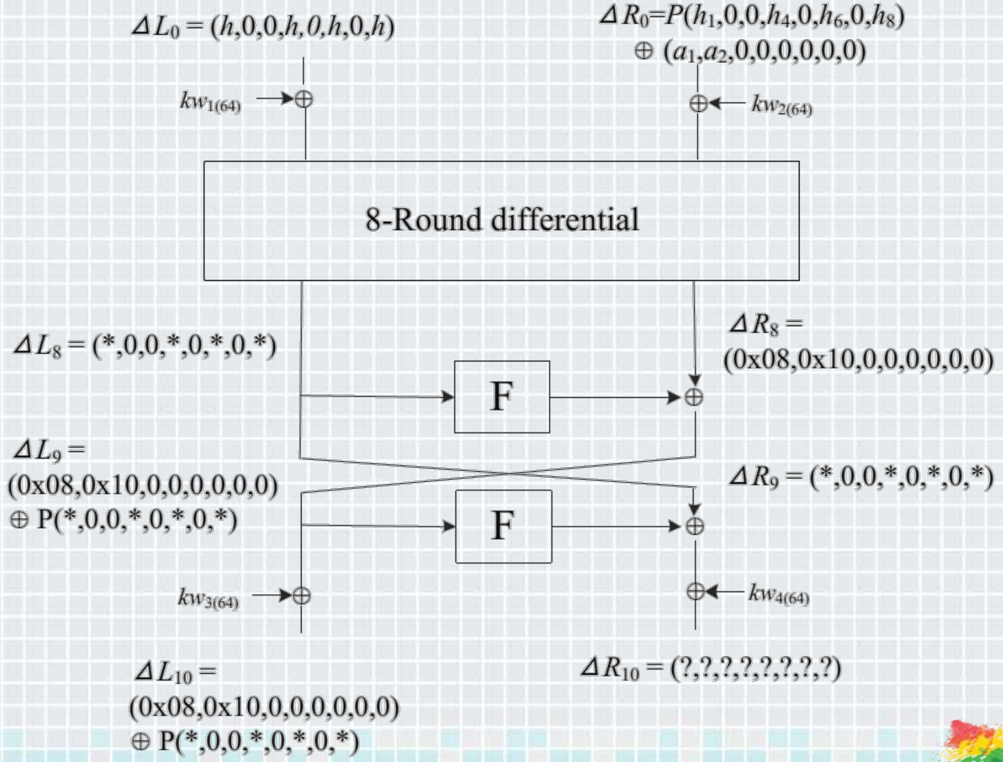
B. Deduce 32-bit key $kw_4 \oplus k_9[1,4,6,8]$

C. Increase the counter of 96-bit subkey "$kw_3 \oplus k_{10}$, $(kw_4 \oplus k_9)[1,4,6,8]$"

- If the right key recovered, then terminate the attack;
- Else replace the attack by choosing other 8-round differentials.
- Search the *Rk*set to find the right key.

Key-dependent attack on10-round Camellia-128

- 99.99% of Key Space
  - Data: 2^{91} CP
  - Time: $2^{104.5}$ ENC
  - Memory: 2^{96} Bytes
- Full Key Space
  - Data: 2^{91} CP
  - Time: 2^{113} ENC
  - Memory: 2^{96} Bytes



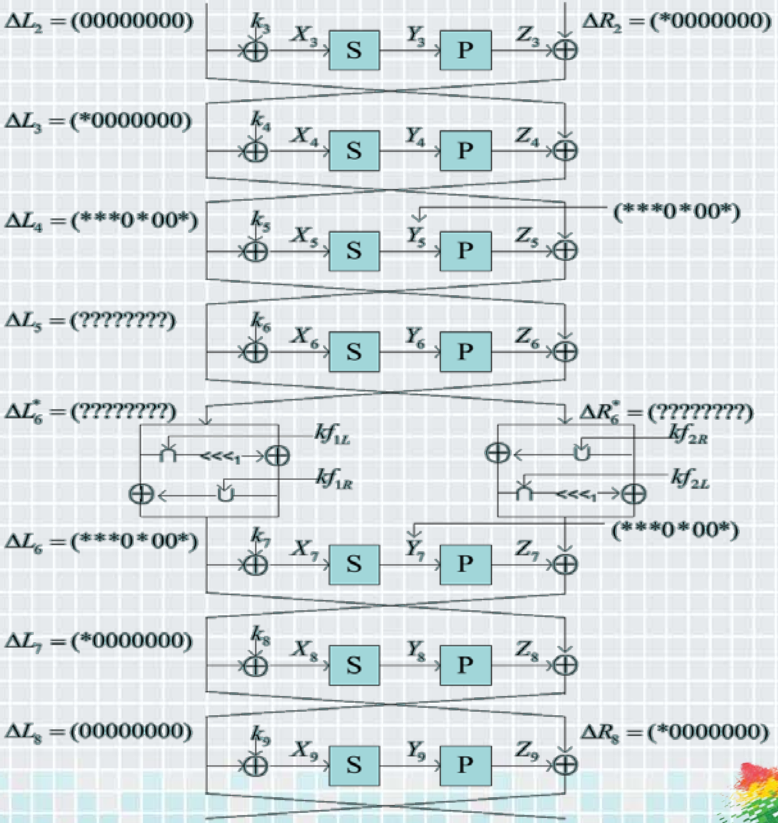
Meet-in-the-Middle Attacks

Meet-in-the-Middle Attacks

- Idea borrowed from Dunkelman etc.and Derbez-Selçuk’s attacks on AES
  - δ-set
  - Multiset

New 7-round property for Camellia-192

- 7-round property
  - $R_2[1]$ is the active byte
  - Multiset of $(P^{-1}(\Delta L_8))[6]$ only takes about 2^{128} values
  - $\Delta X_4[1] \| \Delta Y_4[1] \| \Delta Y_5[1,2,3,5,8] \| \Delta X_8[1] \| \Delta Y_8[1] \| X_7[7] \| X_8[6] \| kf_1$ where…



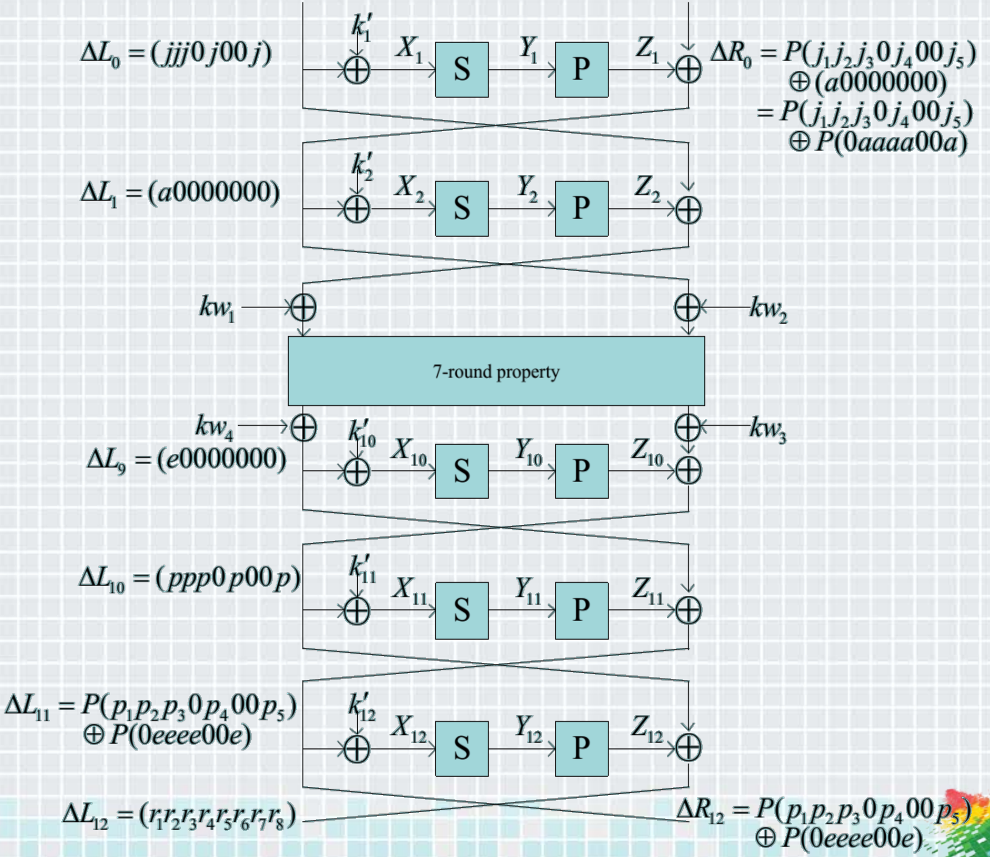
Proof of 7-round property

- Obviously, Multiset of $(P^{-1}(\Delta L_8))[6]$ is determinted by 36-byte value
  - $X_4[1]||X_5[1,2,3,5,8]||X_6||kf_1||kf_2||X_7[2,3,5,7,8]||X_8[6]$
- If a pair comforms to the truncated differential, then
  - 18-byte "$X_4[1]||X_5[1,2,3,5,8]||X_6||X_7[2,3,5,7,8]$" determined by 9-byte "$\Delta X_4[1]||\Delta Y_4[1]||\Delta Y_5[1,2,3,5,8]||\Delta X_8[1]||\Delta Y_8[1]$" and 128-bit "$kf_1||kf_2$"
    - $Pr(\Delta Y_7[4,6,7]=0)=2^{-24}$ and
    - $\Delta Y_7=P^{-1}(FL^{-1}(P(\Delta Y_5)\oplus\Delta L_3))\oplus P^{-1}(\Delta L_7)$
- So there are 2^{128} values for multiset of $(P^{-1}(\Delta L_8))[6]$

Only has 64-bit information for Camellia-192

12-round attack for Camellia-192

- Add two round on the top
- Three round on the bottom



12-round attack for Camellia-192

- Precomputation Phase
  - Get 2^{128} possible values for multiset of $(P^{-1}(\Delta L_8))[6]$ stored in $\mathcal{H}$
- Online Phase
  - Data collection
    - Structure: each contains 2^{56} plaintexts

$L0 = (\alpha, \alpha \oplus x_1, \alpha \oplus x_2, x_3, \alpha \oplus x_4, x_5, x_6, \alpha \oplus x_7)$ and

$R0 = P(\beta_1, \beta_2, \beta_3, \beta_4, \beta_5, y_1, y_2, \beta_6)$, where $xi(i = 1, \ldots, 7), y_1, y_2$ are constant, $\alpha, \beta j(j = 1, \ldots, 6)$ take all values.

    - Choose 2^{57} structure, $2^{57+111-16} = 2^{152}$ pairs satisfy $P^{-1}(\Delta R_{12})[6,7] = 0$

12-round attack for Camellia-192

- Key Guessing to find a pair conforming to the truncated differential

A. For $l = 2,3,4,5,6,7,8$, guess $k'_{12}[l]$ one by one, paritially decrypt, use $\Delta Y_{11}[l] = P(\Delta L_{12}[l])$ to filter and $2^{152-7\times 8} = 2^{96}$ pairs left.

B. For $l = 2,3,5,8$, guess $k'_{11}[l]$, paritially decrypt, use $\Delta Y_{11}[l] = P^{-1}(\Delta R_{12})[l] \oplus P^{-1}(\Delta R_{12})[4]$ to filter. Then guess $k_{11}'[1]$ and keep the $\Delta Y_{11}[1] = P^{-1}(\Delta R_{12})[1]$ hold. $2^{96-40} = 2^{56}$ pairs remain.

C. For $l = 1,2,3,5,8$, guess $k'_1[l]$, make $\Delta Y_1[1] = P^{-1}(\Delta R_0)[1]$ and $\Delta Y_1[l] = P^{-1}(\Delta R_0)[l] \oplus P^{-1}(\Delta R_0)[4]$ hold. 2^{16} pairs remain.

SHANDONG UNIVERSITY 1901

12-round attack for Camellia-192

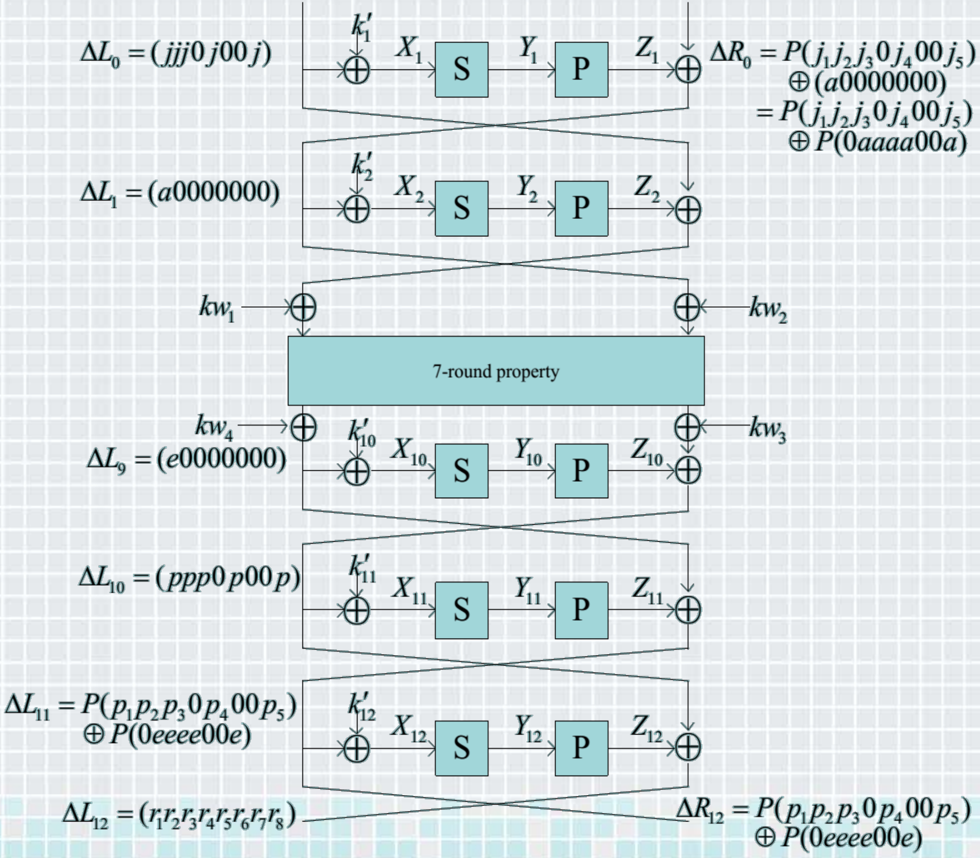
- Construct δ-set for every remaining pair
  A. Deduce $X_2[1]||Y_2[1]$ by difference distribution table of s_1
  B. For pair $(L_0||R_0, L'_0||R'_0)$, corresponding to $(X_2[1], X'2[1])$, change $X'2[1]$ to a different $X''_2[1]$, compute $\Delta Y'_2[1]$, get the difference $\Delta L'_0[1,2,3,5,8]$. Get $L''_0 = L_0 \oplus \Delta L'_0$.
  C. Compute $\Delta Y'_1[1,2,3,5,8]$by the guessed key $k'_1[1,2,3,5,8]$, obtain $\Delta R'0$then get $R''_0 = R_0 \oplus \Delta R'_0$
  D. Get δ-set.

12-round attack for Camellia-192

- For each δ-set under 144-bit key guesses
  - Compute $Y_{11}[2,3,5,8]$, $P^{-1}(L_{10})[6]$ for every (P, C) pair.
  - Guess $k'_{11}[7]$ to compute $X_{10}[6]$.
  - Guess $k'_{10}[6]$ to compute the multiset of $P^{-1}(\Delta L_8)[6]$
  - Check the multiset belongs to $\mathcal{H}$ or not, the wrong value could pass the check with $Pr = 2^{128} \times 2^{-467.6} = 2^{-339.6}$.

12-round attack for Camellia-192

- Full Key Space
  - Data: 2^{113} CP
  - Time: 2^{180} ENC
  - Memory: 2^{158} Bytes



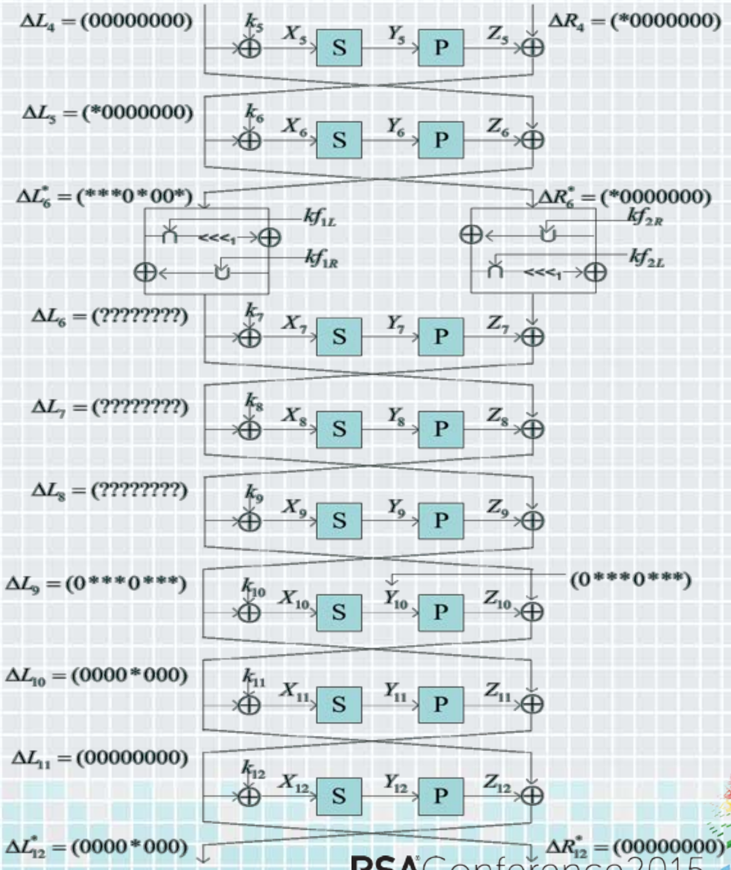
New 8-round property for Camellia-256

- 8-round property
  - $L_{12}[5]$ is active
  - Multiset of bytes$(P^{-1}(\Delta L_4))[1]$ Only take 2^{225} values
- Determined by the 225-bit：

  $\Delta X_{11}[5]||\Delta Y_{11}[5]||\Delta Y_{10}[2,3,4,6,7,8]||$

  $\Delta Y_9||\Delta X_6[1]||\Delta Y_6[1]||kf_1||kf_{2L}[1]||$

  $kf_{2R}[1]||kf_{2L}\{9\}$



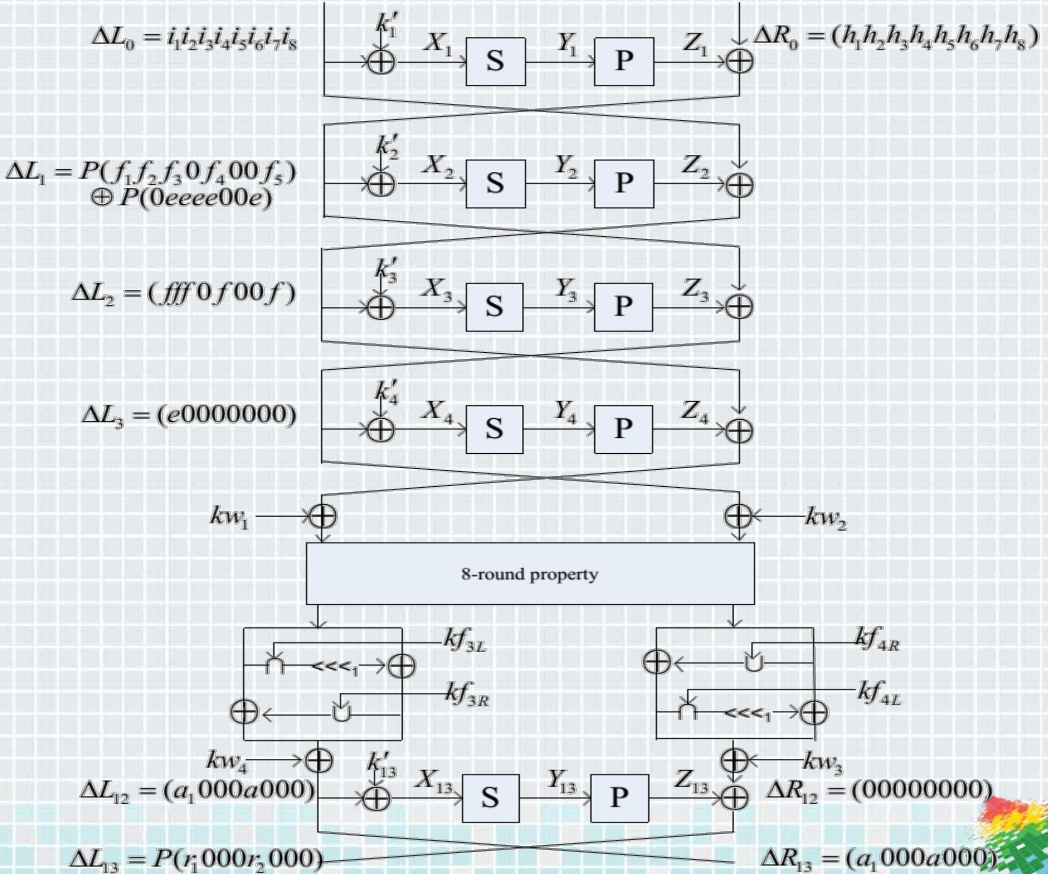
Proof of 8-round property

- Multiset of $(P^{-1}(\Delta L_8))[6]$ is determinted by 321-bit value
  - $X_{11}[5]||X_{10}[2,3,4,6,7,8]||X_9||X_8||X_7\ ||kf_1\{9\text{-}33,42\text{-}64\}||kf_{2L}[1]||kf_{2R}[1]||kf_{2L}[9]\ ||X_6[1]$
- If a pair comforms to the 8-round truncated differential, then
  - 312-bit "$\Delta X_{11}[5]||\Delta X_{10}[2,3,4,6,7,8]||X_9||X_8||X_7||X_6[1]||kf_1\{9\text{-}33,42\text{-}64\}||kf_{2L}[1]$" determined by 216-bit "$\Delta X_{11}[5]||\Delta Y_{11}[5]||\Delta Y_{10}[2,3,4,6,7,8]||\Delta Y_9||\Delta X_6[1]||\Delta Y_6[1]||kf_1||kf_{2L}[1]$"
  - kf2R[1]||kf2L{9} are also needed to compute Multiset of $(P^{-1}(\Delta L_8))[6]$
- Totally, 225-bit determines the Multiset of $(P^{-1}(\Delta L_8))[6]$

13-round attack for Camellia-256

- Add 4 rounds on the top

  1 rounds on the bottom



13-round attack for Camellia-256

- Precomputation Phase
  - Compute 2^{225} values of multiset store them in hash table $\mathcal{H}$.
- Online Phase
  - Data collection
    - Structure: each contains 2^{32} ciphertexts

$L_{13} = (\alpha_1, x_1, x_2, x_3, \alpha_2, x_4, x_5, x_6)$ and

$R_{13} = P(\beta_1, y_1, y_2, y_3, \beta_2, y_4, y_5, y_6)$, where $x_i, yi(i = 1, \ldots, 6)$are constant, $\alpha_j, \beta_j(j = 1,2)$ take all values. Decrypt to get the plaintexts.

    - Choose 2^{81} structure to get 2^{144} pairs

13-round attack for Camellia-256

- Key Guessing to find a pair conforming to the truncated differential

A. Guess k'_1 , compute $P^{-1}(\Delta L_1)$,eliminate pairs that do not satisy $P^{-1}(\Delta L_1)[6,7]=0$, $2^{144-16}=2^{128}$ pairs left.

B. For $l=2,3,4,6,7,8$, guess $k'_2\ [l]$, paritially encrypt, make $\Delta Y_2[l]=P^{-1}(\Delta L_0)[l]$ hold. Then guess $k_2'\ [1]$ to compute L_2 . $2^{128-7*8}=2^{72}$pairs remain.

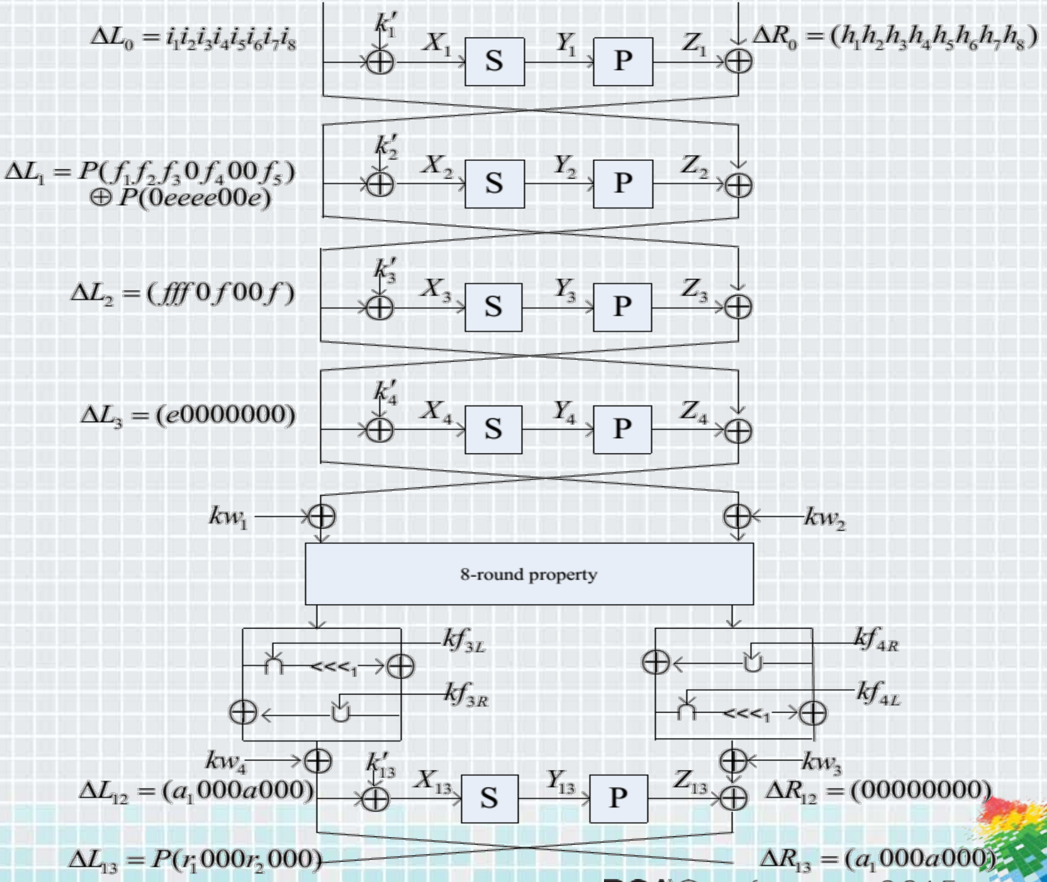
C. For $l=2,3,5,8$, guess $k'_3\ [l]$, make $\Delta Y_3[1]=P^{-1}(\Delta L_1)[l]\oplus P^{-1}(\Delta L_1)[4]$ Then guess $k_3'\ [1]$ and keep the pairs satisfy $\Delta Y_3[1]=P^{-1}\ (\Delta L_1)[1]$. 2^{32}pairs left for every 168-bit key guess.

13-round attack for Camellia-256

- Key Guessing to find a pair conforming to the truncated differential
  - D. For $l = 1{,}5$, guess $k'_{13}[l]$, partially decrypt, make $\Delta Y_{13}[l] = P^{-1}(\Delta L_{13})[l]$ hold. Then guess $kf_{3R}[1]$ to compute $\Delta L_{12}{}^{*}[1]$and delete the pairs when $\Delta L_{12}{}^{*}[1] \neq 0$. 2^8pairs remain.
  - E. Compute the value L_3 by guessing $k'_3\,[4{,}6{,}7]$ and deduce $k'_4[1]$ for each pair.
- Construct δ-set for each pair, and compute the corresponding multiset to check it whether belongs to $\mathcal{H}$, and recover the right key.

13-round attack for Camellia-256

- Full Key Space
  - Data: 2^{113} CC
  - Time: $2^{232.7}$ ENC
  - Memory: 2^{231} Bytes



That is all

Thank you!