Superposition Meet-in-the-Middle Attacks: Updates on Fundamental Security of AES-like Hashing

Zhenzhen Bao, Jian Guo, Danping Shi, Yi Tu

Crypto 2022, August 13–18

Outline

1. Background and Preliminaries
2. Enhanced MITM-MILP Modeling
   - Superposition states and separate attribute-propagation (SupP)
   - Bi-directional attribute-propagation and cancellation (BiDir)
   - Guess-and-determine (GnD)
   - Multiple ways of `AddRoundKey` (MulAK)
   - Applications to collision and key-recovery attacks
3. Updates on Fundamental Security of AES-like Hashing
4. Conclusions and Future Work

Meet-in-the-Middle (MITM) attacks [DH77; MH81]

Generic attack

At the top-level of abstraction, regardless of the internal details of the primitive.

Double Encryption (e.g., Double DES)

$C = S_{K_2}(S_{K_1}(P))$

n: Number of keys in key space of S

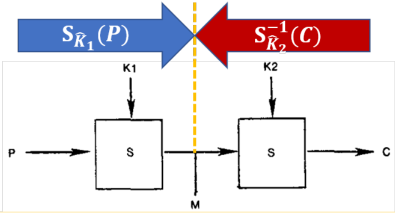
Cplx: $T: n, M: n$

(1) For $i = 1$ to n Do
 (a) Table[i] = $\langle S_i(P), i$, "encrypt"$\rangle$
 (b) Table[$n+i$] = $\langle S_i^{-1}(C), i$, "decrypt"$\rangle$

(2) Sort the table on the first field.

(3a) Search the table for adjacent entries of the form
 $\langle$value, $\hat{K}1$, "encrypt"$\rangle$
 $\langle$value, $\hat{K}2$, "decrypt"$\rangle$

(3b) Test to see if $\hat{K}1$ and $\hat{K}2$ are the correct keys by encrypting one additional plaintext–ciphertext pair.



Unicity distance arguments: For DES, 56-bit key, 64-bit state
One (P, C): $2^{2\times 56-64} = 2^{48}$; Second (P, C): $2^{48-64} = 2^{-16}$.

Triple Encryption (e.g., Triple DES)

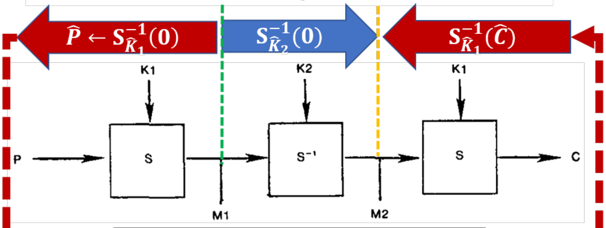
$C = S_{K_1}(S^{-1}_{K_2}(S_{K_1}(P)))$

(1) For $i = 1$ to n Do
 (a) $\hat{M}2 = S_i^{-1}(0)$
 (b) Table[i] = $\langle \hat{M}2, i$, "middle"$\rangle$
 (c) $\hat{M}2' = S_i^{-1}(\mathrm{Enc}(S_i^{-1}(0)))$
 (d) Table[$n+i$] = $\langle \hat{M}2', i$, "ends"$\rangle$

(2) Sort the table on the first field.

(3a) Search the table for adjacent entries of the form
 $\langle$value, $\hat{K}2$, "middle"$\rangle$
 $\langle$value, $\hat{K}1$, "ends"$\rangle$

(3b) Test to see if $\hat{K}1$ and $\hat{K}2$ are the correct keys by checking an additional plaintext–ciphertext pair.



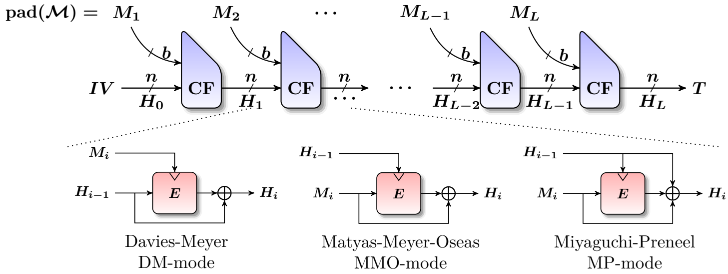
$Enc(\hat{P}) = S_{K_1}(S^{-1}_{K_2}(S_{K_1}(\hat{P}))) \rightarrow \hat{C}$

MITM from key-recovery to preimage attacks on hash functions

Generic MITM preimage attacks on block cipher-based hash functions [EC:LaiMas92].

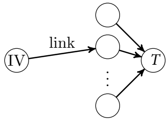
- Block cipher-based hash functions
  e.g., follow Merkle-Damgård construction

$$H_0 = IV,\ H_i = \mathsf{CF}(H_{i-1}, M_i),\ T = \mathsf{CF}(H_{L-1}, M_L)$$



- Compression Function (CF)
  e.g., secure PGV modes

- Preimage attack on $\mathcal{H}$
  - given an n-bit T, find $\mathcal{M}$, s.t. $\mathcal{H}(\mathcal{M}) = T$, Cplx. $< 2^n$.
- Pseudo-preimage attack on CF
  - given an n-bit T, find H and M, s.t. $\mathsf{CF}(H, M) = T$, Cplx. $< 2^n$.
  - converted to preimage attack on $\mathcal{H}$ use generic MITM procedures.

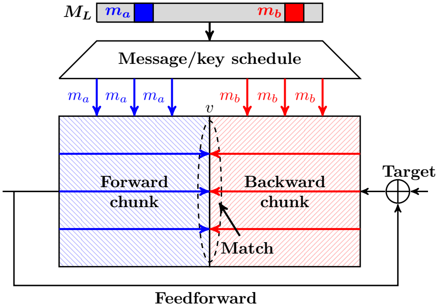


#link : $2^{(n+\ell)/2}$; #PP : $2^{(n-\ell)/2}$;
Cplx: $2^{(n+\ell)/2+1}$

MITM from generic to dedicated attacks on hash functions

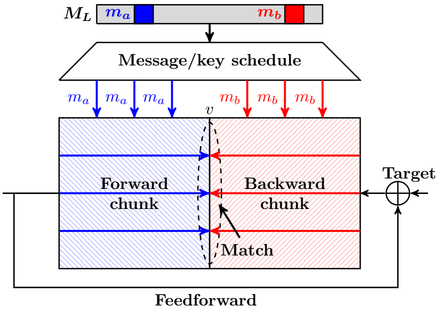
- The MITM idea was used to devise dedicated attacks on hash functions by Saarinen [INDOCRYPT:Saarinen07b], Aumasson *et al.* [SAC:AumMeiMen08], Sasaki and Aoki [AC:SasAok08].
- Applications with several novel techniques:
  - Full preimage attacks:
    - MD4 [AC:GLRW10]
    - MD5 [EC:SasAok09]
    - Tiger [AC:GLRW10]
    - HAVAL [AC:SasAok08]
    - Haraka-512 v2 [EC:BDGLSSW21]
    - ...
  - Best preimage attacks:
    - SHA-1 [C:KneKho12]
    - SHA-2 [AC:GLRW10; FSE:KhoRecSav12]
    - Whirlpool [AC:SWWW12]
    - Grøstl [IWSEC:MLHL15; JISE:ZouWWD14]
    - AES hashing modes [EC:BDGLSSW21]
    - ...
  - Convert preimage into collision attacks: pseudo collision on SHA-2 [FSE:LiIsoShi12]

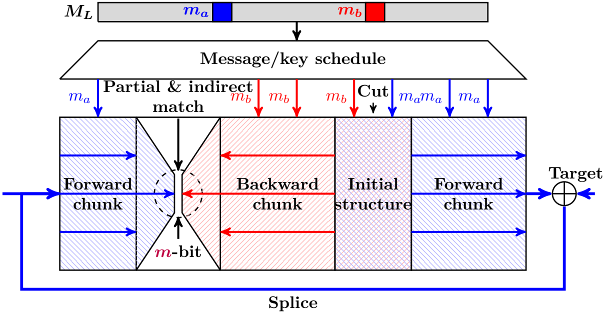
The meet-in-the-middle pseudo-preimage attacks on **CF**



- For $2^{n-(d_1+d_2)}$ values of M_L/ $\{m_a, m_b\}$
  - For 2^{d_1} values of m_a, forward compute to get a list $\overrightarrow{\mathcal{L}}$ of v.
  - For 2^{d_2} values of m_b, backward compute to get a list $\overleftarrow{\mathcal{L}}$ of v.
  - If find a match between $\overrightarrow{\mathcal{L}}$ and $\overleftarrow{\mathcal{L}}$, return the correspondence M_L.

The meet-in-the-middle pseudo-preimage attacks on **CF**

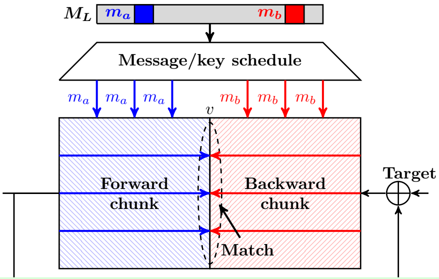


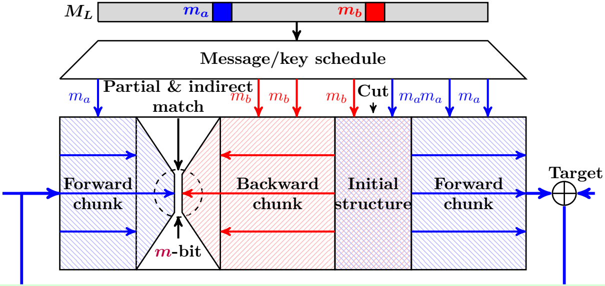


- For $2^{n-(d_1+d_2)}$ values of $M_L / \{m_a, m_b\}$
  - For 2^{d_1} values of m_a, forward compute to get a list $\overrightarrow{\mathcal{L}}$ of v.
  - For 2^{d_2} values of m_b, backward compute to get a list $\overleftarrow{\mathcal{L}}$ of v.
  - If find a match between $\overrightarrow{\mathcal{L}}$ and $\overleftarrow{\mathcal{L}}$, return the correspondence M_L.

- Splice-and-cut: better chunk separations
- Initial structure: more rounds
  - neutral words appear simultaneously
  - local-collision-like cancellation of impact
- Partial & indirect matching: more rounds
  - filtering using partial state ($m < n$ bits)
  - indirect matching via linear relations.

The meet-in-the-middle pseudo-preimage attacks on **CF**





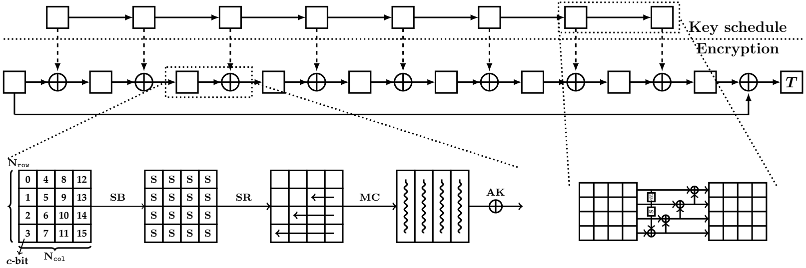
Complexity

$$2^{n-(d_1+d_2)} \cdot (2^{\max(d_1,d_2)} + 2^{d_1+d_2-m}) \simeq 2^{n-\min(d_1,d_2,m)}.$$

- For 2^{d_1} values of m_a, forward compute to get a list $\overrightarrow{\mathcal{L}}$ of v.
- For 2^{d_2} values of m_b, backward compute to get a list $\overleftarrow{\mathcal{L}}$ of v.
- If find a match between $\overrightarrow{\mathcal{L}}$ and $\overleftarrow{\mathcal{L}}$, return the correspondence M_L.

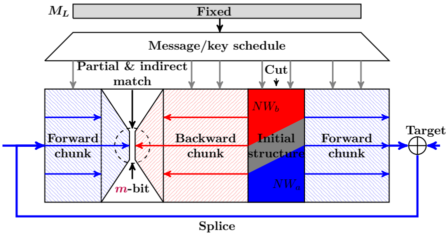
- Initial structure: more rounds
  - neutral words appear simultaneously
  - local-collision-like cancellation of impact
- Partial & indirect matching: more rounds
  - filtering using partial state ($m < n$ bits)
  - indirect matching via linear relations.

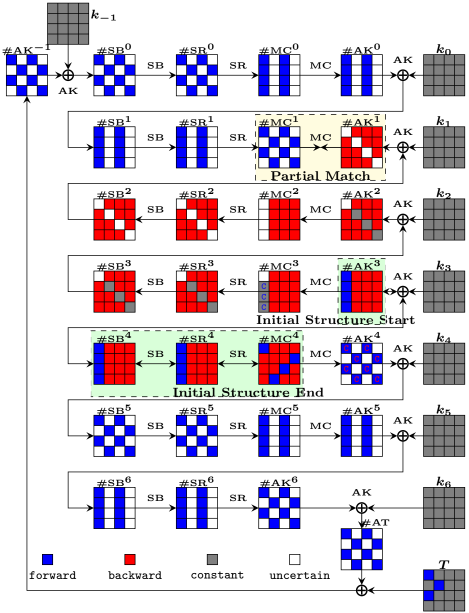
AES-like hashing



- **SubBytes (SB).** Substitute each cell according to an S-boxes $S : \mathbb{F}_{2^c} \to \mathbb{F}_{2^c}$.
- **ShiftRows$_{\pi_t}$ (SR).** Permute the cell positions according to the permutation π_t.
- **MixColumns (MC).** Update each column by left-multiplying an $\mathrm{N}_{\texttt{row}} \times \mathrm{N}_{\texttt{row}}$ matrix.
- **AddRoundKey (AK).** XOR a round key or a round-dependent constant into the state.

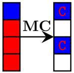
The MITM preimage attacks on **AES** hashing [FSE:Sasaki11; FSE:WFWGDZ12]





Initial structure: add constraints to cancel impact

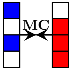
Constraints on #MC^4[1, 2, 3] to build the initial structure:



$$i.e., \begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix} \times \begin{bmatrix} 0 \\ \#\mathrm{MC}^4[1] \\ \#\mathrm{MC}^4[2] \\ \#\mathrm{MC}^4[3] \end{bmatrix} = \begin{bmatrix} c_0 \\ - \\ c_1 \\ - \end{bmatrix} \Rightarrow$$

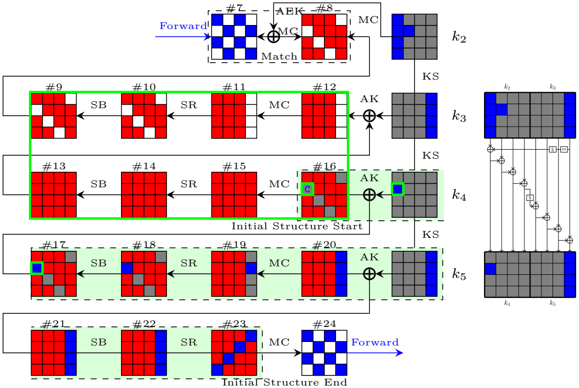
$$\begin{bmatrix} 3 \cdot \#\mathrm{MC}^4[1] & \oplus\, 1 \cdot \#\mathrm{MC}^4[2] & \oplus\, 1 \cdot \#\mathrm{MC}^4[3] \\ 1 \cdot \#\mathrm{MC}^4[1] & \oplus\, 2 \cdot \#\mathrm{MC}^4[2] & \oplus\, 3 \cdot \#\mathrm{MC}^4[3] \end{bmatrix} = \begin{bmatrix} c_0 \\ c_1 \end{bmatrix}$$

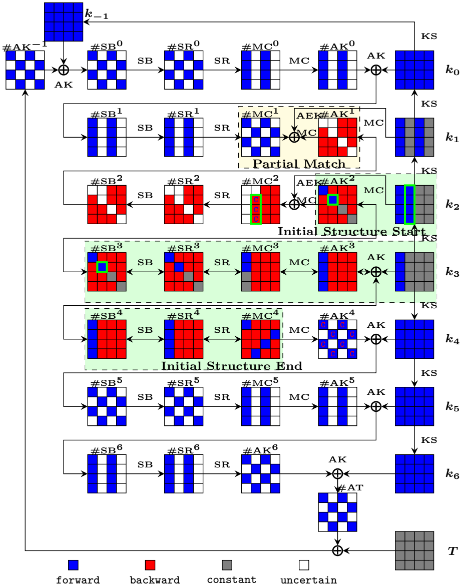
Partial & indirect matching



Known any 4 out of the 8 input and output elements of the MDS matrix, the remaining 4 elements can be computed
($m = 8 \cdot (|\{\blacksquare, \blacksquare, \blacksquare\}| - 4)$)

The MITM preimage attacks on **AES** hashing [ToSC:BDGWZ19]





Introduce Neutral Bytes in Key

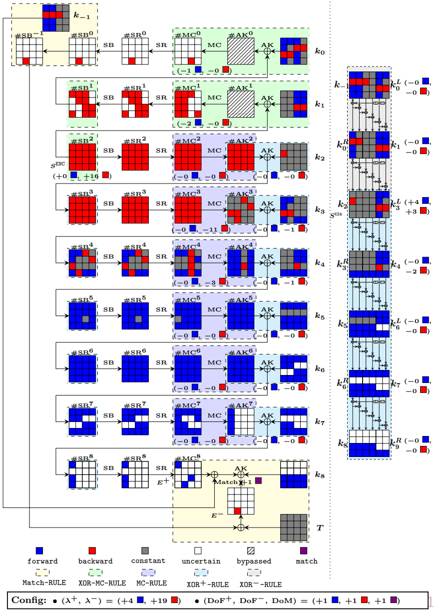
- Reduce complexity: add degrees of freedom
- Cover more rounds: cancel impacts

Combine AK and MC

- More ways to cancel impacts

Automatic search of MITM attacks with MILP [EC:BDGLSSW21]

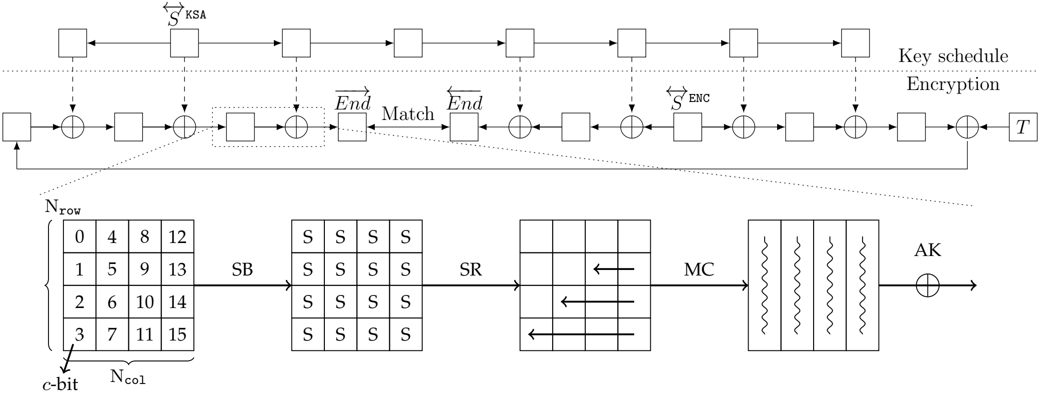
- Generalization
  - remove artificial limitations,
  - extend attack space:
    - cover all possible combinations of starting and matching points, in encryption and key-schedule
    - select neutral bytes from both encryption and key states for both chunks
    - apply the essential idea behind initial structure to every possible round
- Translation
  - translate the formalized attack configurations into Mixed-Integer-Linear-Programming (MILP) models
  - reduce the search for the best attacks into solving optimization problems under constraints in MILP.



Basic MILP model for MITM preimage attack [EC:BDGLSSW21]

- The top-level of a search:
  - enumerate all high-level configurations
- A high-level configuration is determined by four parameters
  - $\texttt{total}_r$: the total number of targeted rounds
  - $\texttt{init}_r^{\mathrm{E}}$: the position of the round from where we select the neutral words in encryption
  - $\texttt{init}_r^{\mathrm{K}}$: the position of the round from where we select the neutral words in key-schedule
  - $\texttt{match}_r$: the position of the round at where we match
- For each combination of $(\texttt{total}_r, \texttt{init}_r^{\mathrm{E}}, \texttt{init}_r^{\mathrm{K}}, \texttt{match}_r)$, build an individual MILP model

Basic MILP model for MITM preimage attack



When building an individual MILP model, constraints are imposed on propagation of attributes

- starting from some initial states (*i.e.*, $\overleftrightarrow{S}^{\texttt{ENC}}$ and $\overleftrightarrow{S}^{\texttt{KSA}}$ in round $\texttt{init}_r^{\mathrm{E}}$ and $\texttt{init}_r^{\mathrm{K}}$),
- terminating at some ending states (*i.e.*, $\overrightarrow{End}$ and $\overleftarrow{End}$ in round $\texttt{match}_r$) from two directions.

Basic MILP model for MITM preimage attack [EC:BDGLSSW21]

Encoding the attribute of the i-th cell of a state S: two 0-1 variables x_i^S, y_i^S

- $x_i^S = 1 \iff$ the i-th cell of state S is computable in the forward chunk, *i.e.*, {`Blue` (■), `Gray` (■)}
- $y_i^S = 1 \iff$ the i-th cell of state S is computable in the backward chunk, *i.e.*, {`Red` (■), `Gray` (■)}

$$(x_i^S, y_i^S) = \begin{cases} (1,1) & \texttt{Gray}\ (■)\text{: computable in both chunks} \\ (1,0) & \texttt{Blue}\ (■)\text{: computable only in the forward chunk} \\ (0,1) & \texttt{Red}\ (■)\text{: computable only in the backward chunk} \\ (0,0) & \texttt{White}\ (□)\text{: Incomputable in both chunks} \end{cases}, \quad \begin{cases} x_i^S - \beta_i^S \geq 0; \\ y_i^S - \beta_i^S \geq 0; \\ x_i^S + y_i^S - \beta_i^S \leq 1; \\ \omega_i^S + x_i^S + y_i^S - \beta_i^S = 1. \end{cases}$$

- $\beta_i^S = 1 \iff$ the i-th cell of state S is computable in both chunks, *i.e.*, `Gray` (■)
- $\omega_i^S = 1 \iff$ the i-th cell of state S is incomputable in both chunks, *i.e.*, `White` (□)

Starting states $\overleftrightarrow{S}^{\texttt{ENC}}$ and $\overleftrightarrow{S}^{\texttt{KSA}}$, and initial degrees of freedom $\overrightarrow{\iota}^{■}$ and $\overleftarrow{\iota}^{■}$

$$\begin{cases} \overrightarrow{\iota}^{■} = \sum_i (x_i^{\overleftrightarrow{S}^{\texttt{ENC}}} - \beta_i^{\overleftrightarrow{S}^{\texttt{ENC}}}) + \sum_i (x_i^{\overleftrightarrow{S}^{\texttt{KSA}}} - \beta_i^{\overleftrightarrow{S}^{\texttt{KSA}}}), & \text{initial DoF for forward, ■'s in } \{\overleftrightarrow{S}^{\texttt{ENC}}, \overleftrightarrow{S}^{\texttt{KSA}}\} \\ \overleftarrow{\iota}^{■} = \sum_i (y_i^{\overleftrightarrow{S}^{\texttt{ENC}}} - \beta_i^{\overleftrightarrow{S}^{\texttt{ENC}}}) + \sum_i (y_i^{\overleftrightarrow{S}^{\texttt{KSA}}} - \beta_i^{\overleftrightarrow{S}^{\texttt{KSA}}}), & \text{initial DoF for backward, ■'s in } \{\overleftrightarrow{S}^{\texttt{ENC}}, \overleftrightarrow{S}^{\texttt{KSA}}\} \end{cases}$$

Basic MILP model for MITM preimage attack [EC:BDGLSSW21]

Degree of Match ($\overset{\rightarrow\leftarrow}{m}{}^{\color{purple}\blacksquare}$), Degree of Freedom for forward ($\overrightarrow{d_b}{}^{\color{blue}\blacksquare}$) and for backward ($\overleftarrow{d_r}{}^{\color{red}\blacksquare}$)

$$\overset{\rightarrow\leftarrow}{m}{}^{\color{purple}\blacksquare} = \sum_{j=0}^{\mathrm{N}_{\mathtt{col}}-1} \max\{0, (\mathrm{N}_{\mathtt{row}} - \sum_{i=0}^{\mathrm{N}_{\mathtt{row}}-1} \square_{i,j}^{\overrightarrow{End}}) + (\mathrm{N}_{\mathtt{row}} - \sum_{i=0}^{\mathrm{N}_{\mathtt{row}}-1} \square_{i,j}^{\overleftarrow{End}}) - \mathrm{N}_{\mathtt{row}}\}$$

$$\begin{cases} \overset{\rightarrow\leftarrow}{m}{}^{\color{purple}\blacksquare} = \sum_{j=0}^{\mathrm{N}_{\mathtt{col}}-1} \max\{0, \mathrm{N}_{\mathtt{row}} - \sum_{i=0}^{\mathrm{N}_{\mathtt{row}}-1} \omega_{(i,j)}^{\overrightarrow{End}} - \sum_{i=0}^{\mathrm{N}_{\mathtt{row}}-1} \omega_{(i,j)}^{\overleftarrow{End}}\}; \\ \overset{\rightarrow\leftarrow}{m}{}^{\color{purple}\blacksquare} \geq 1. \end{cases}$$

$$\begin{cases} \overrightarrow{d_b}{}^{\color{blue}\blacksquare} = \overrightarrow{\iota}{}^{\color{blue}\blacksquare} - \overrightarrow{\sigma}{}^{\color{blue}\blacksquare}, \\ \overleftarrow{d_r}{}^{\color{red}\blacksquare} = \overleftarrow{\iota}{}^{\color{red}\blacksquare} - \overleftarrow{\sigma}{}^{\color{red}\blacksquare}, \\ \overrightarrow{d_b}{}^{\color{blue}\blacksquare} \geq 1, \\ \overleftarrow{d_r}{}^{\color{red}\blacksquare} \geq 1. \end{cases}$$

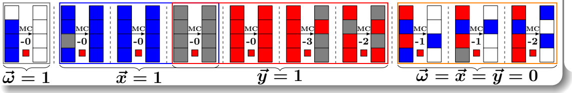
The Objective Function

$$\tau_{\mathtt{Obj}} := \min\{\overrightarrow{d_b}{}^{\color{blue}\blacksquare}, \overleftarrow{d_r}{}^{\color{red}\blacksquare}, \overset{\rightarrow\leftarrow}{m}{}^{\color{purple}\blacksquare}\} \Rightarrow \begin{cases} \tau_{\mathtt{Obj}} \leq \overrightarrow{d_b}{}^{\color{blue}\blacksquare}; \\ \tau_{\mathtt{Obj}} \leq \overleftarrow{d_r}{}^{\color{red}\blacksquare}; \\ \tau_{\mathtt{Obj}} \leq \overset{\rightarrow\leftarrow}{m}{}^{\color{purple}\blacksquare}. \end{cases}$$

Cplx: $2^{n-\min(\boldsymbol{d_1},\boldsymbol{d_2},\boldsymbol{m})} \Longrightarrow$ Objective : Maximize $\tau_{\mathtt{Obj}}$

Basic MILP model for MITM preimage attack [EC:BDGLSSW21]

Translate the rules of attribute-propagation into MILP: *e.g.*, `MC-RULE` in the forward chunk



Introduce 0-1 indicator variables for each input column

$$\begin{cases} \vec{\omega}=1 \Leftrightarrow \text{exists incomputable (exists } \square) \\ \vec{x}=1 \Leftrightarrow \text{are all computable for forward (only } \blacksquare,\blacksquare) \\ \vec{y}=1 \Leftrightarrow \text{are all computable for backward (only } \blacksquare,\blacksquare) \end{cases}$$

Indicator variables

$$\begin{cases} \vec{\omega}=\max\limits_{i=0}^{\mathtt{N_{row}}-1}(\omega_i^I), \\ \sum\limits_{i=0}^{\mathtt{N_{row}}-1} x_i^I - \mathtt{N_{row}}\cdot\vec{x} \geq 0, \\ \sum\limits_{i=0}^{\mathtt{N_{row}}-1} x_i^I - \vec{x} \leq \mathtt{N_{row}}-1. \\ \sum\limits_{i=0}^{\mathtt{N_{row}}-1} y_i^I - \mathtt{N_{row}}\cdot\vec{y} \geq 0, \\ \sum\limits_{i=0}^{\mathtt{N_{row}}-1} y_i^I - \vec{y} \leq \mathtt{N_{row}}-1. \end{cases}$$

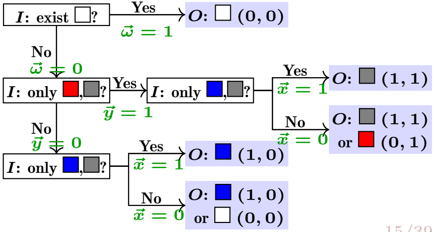
Attribute-propagation through MC

$$\begin{cases} \sum\limits_{i=0}^{\mathtt{N_{row}}-1} x_i^O + \mathtt{N_{row}}\cdot\vec{\omega} \leq \mathtt{N_{row}}, \\ \sum\limits_{i=0}^{\mathtt{N_{row}}-1} y_i^O + \mathtt{N_{row}}\cdot\vec{\omega} \leq \mathtt{N_{row}}, \\ \sum\limits_{i=0}^{\mathtt{N_{row}}-1} y_i^O - \mathtt{N_{row}}\cdot\vec{y} = 0, \\ \sum\limits_{i=0}^{\mathtt{N_{row}}-1} (x_i^O + x_i^I) - \mathtt{Br_n}\cdot\vec{x} \leq \mathtt{N_{iosum}} - \mathtt{Br_n}, \\ \sum\limits_{i=0}^{\mathtt{N_{row}}-1} (x_i^O + x_i^I) - \mathtt{N_{iosum}}\cdot\vec{x} \geq 0. \end{cases}$$

Canceling impact by consuming DoF

$$\left\{ \sum\limits_{i=0}^{\mathtt{N_{row}}-1} x_i^O - \mathtt{N_{row}}\cdot\vec{x} - c_{\vec{y}} = 0. \right.$$

$(x_0^I, y_0^I, \cdots, x_{\mathtt{N_{row}}-1}^I, y_{\mathtt{N_{row}}-1}^I, \vec{\omega}, \vec{x}, \vec{y}) \rightarrow (x_0^O, y_0^O, \cdots, x_{\mathtt{N_{row}}-1}^O, y_{\mathtt{N_{row}}-1}^O)$



Outline

1. Background and Preliminaries
2. Enhanced MITM-MILP Modeling
   - Superposition states and separate attribute-propagation (SupP)
   - Bi-directional attribute-propagation and cancellation (BiDir)
   - Guess-and-determine (GnD)
   - Multiple ways of `AddRoundKey` (MulAK)
   - Applications to collision and key-recovery attacks
3. Updates on Fundamental Security of AES-like Hashing
4. Conclusions and Future Work

Outline

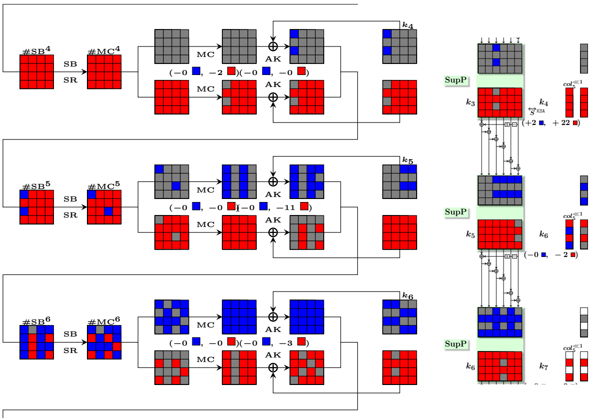
1. Background and Preliminaries
2. Enhanced MITM-MILP Modeling
   - Superposition states and separate attribute-propagation (SupP)
   - Bi-directional attribute-propagation and cancellation (BiDir)
   - Guess-and-determine (GnD)
   - Multiple ways of `AddRoundKey` (MulAK)
   - Applications to collision and key-recovery attacks
3. Updates on Fundamental Security of AES-like Hashing
4. Conclusions and Future Work

Superposition states and separate attribute-propagation (SupP)

Superposition States

- Every intermediate state is viewed as a combination of two virtual states.
- Each virtual state carries one attribute propagation through linear operations independently of the other.
- Two virtual states are combined only when going through non-linear operations.

SupP preserves linear combinations

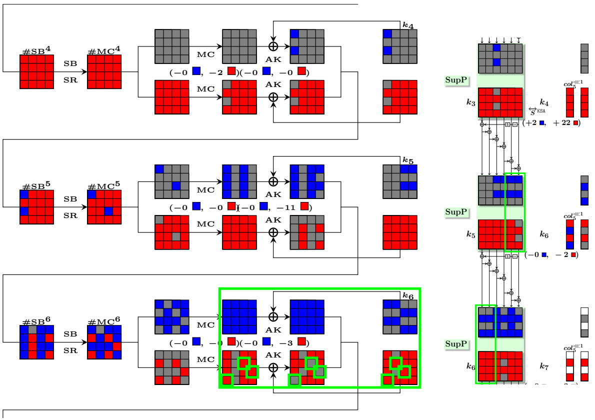


Superposition states and separate attribute-propagation (SupP)

Superposition States

- Every intermediate state is viewed as a combination of two virtual states.
- Each virtual state carries one attribute propagation through linear operations independently of the other.
- Two virtual states are combined only when going through non-linear operations.

SupP preserves linear combinations

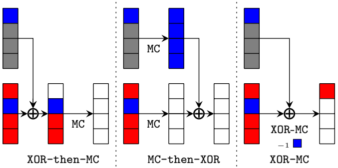


Superposition states and separate attribute-propagation (SupP)

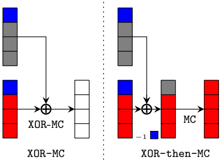
Superposition States

- Every intermediate state is viewed as a combination of two virtual states.
- Each virtual state carries one attribute propagation through linear operations independently of the other.
- Two virtual states are combined only when going through non-linear operations.

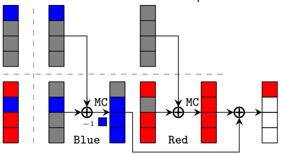
SupP facilitates modeling cancellation



(a) `XOR-MC-RULE` is necessary if not in superposition



(b) `XOR-MC-RULE` does not cover a propagation



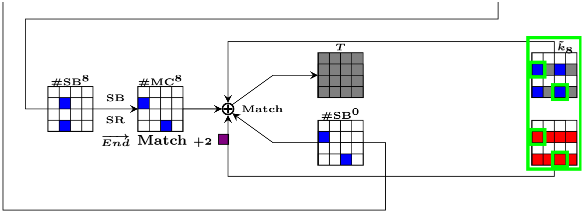
(c) `XOR-RULE` and `MC-RULE` are sufficient once in superposition

Superposition states and separate attribute-propagation (SupP)

Superposition States

- Every intermediate state is viewed as a combination of two virtual states.
- Each virtual state carries one attribute propagation through linear operations independently of the other.
- Two virtual states are combined only when going through non-linear operations.

SupP facilitates modeling matching

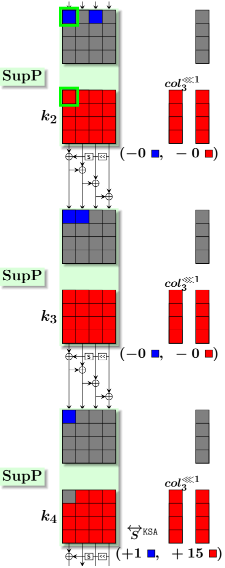


Notes on SupP

SupP representation of the cells is at a level of abstraction, where it presents just the right amount of details, *i.e.*, whether a cell carries a degree of freedom for each of the two chunks.

- compared with algebraic representations, SupP
  - ⋆ compacts many terms into a single indicator term,
  - ⋆ is at a higher level of abstraction, so that the *efficiency* of the search is better.
- compared with representations in previous color scheme, SupP
  - ⋆ expands the formula of each cell into two terms,
  - ⋆ is at a lower level of abstraction, so that the *quality* of the search is better.

$$k_2[0] \leftarrow \underbrace{k_4[0]}_{\blacksquare} \oplus \underbrace{\mathrm{Sbox}(k_4[9] \oplus k_4[13]) \oplus \mathrm{Sbox}(k_4[5] \oplus k_4[13])}_{\blacksquare}$$



Outline

1 Background and Preliminaries

2 Enhanced MITM-MILP Modeling
- Superposition states and separate attribute-propagation (SupP)
- Bi-directional attribute-propagation and cancellation (BiDir)
- Guess-and-determine (GnD)
- Multiple ways of `AddRoundKey` (MulAK)
- Applications to collision and key-recovery attacks

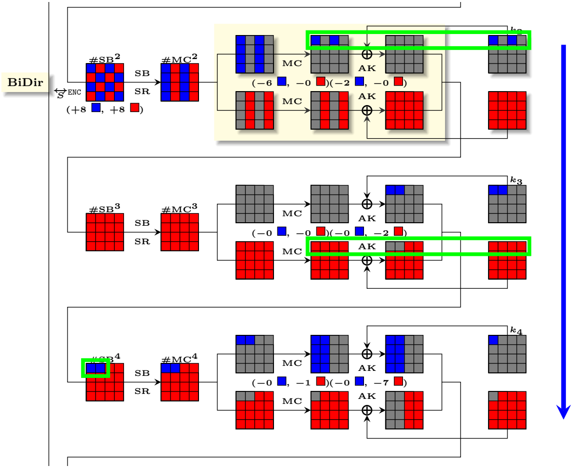
3 Updates on Fundamental Security of AES-like Hashing

4 Conclusions and Future Work

Bi-directional attribute-propagation and cancellation (BiDir)

BiDir enables remote cancellation of impact via `AddRoundKey`

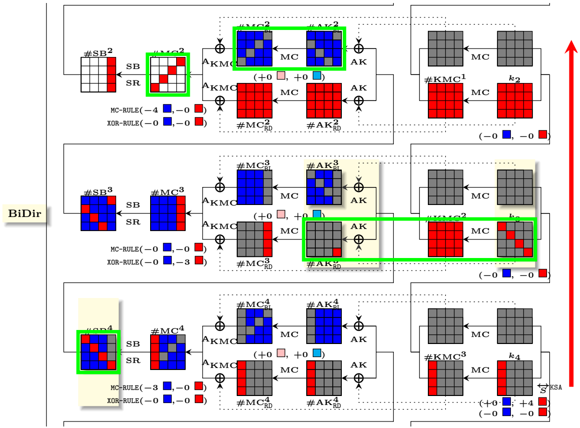
- letting `Blue` be consumed and preserve some local `Red`-cells,
- enable a remote cancellation between the preserved `Red` cells and that introduced `Red` cells from key state through `AddRoundKey`,
- `Blue`-propagation can be continued.



Bi-directional attribute-propagation and cancellation (BiDir)

BiDir enables remote cancellation of impact via `MixColumns`

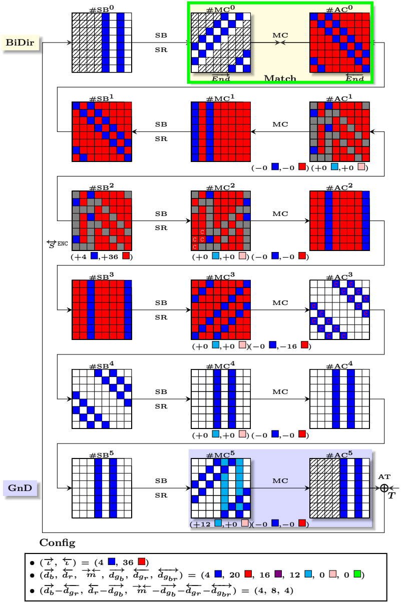
- letting **Red** be consumed and preserve some local **Blue** cells,
- enable these local **Blue** cells to propagate and combine with other **Blue** cells at a remote point such that their impacts on certain cells be mutually canceled through `MixColumns`,
- **Red**-propagation can be continued.



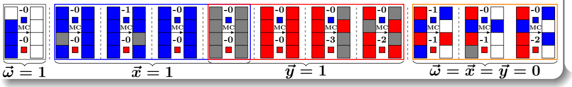
Bi-directional attribute-propagation and cancellation (BiDir)

BiDir contributes to gain degree of matching

- Once an attribute of `Blue` or `Red` propagates to the ending states, no matter from which direction, it provides source of degree of matching.



Translate the rules of bi-directional attribute-propagation into MILP: *e.g.*, `MC-RULE`



Introduce 0-1 indicator variables for each input column

$$\begin{cases} \vec{\omega}=1 \Leftrightarrow \text{exists incomputable (exists } \square) \\ \vec{x}=1 \Leftrightarrow \text{are all computable for forward (only } \blacksquare,\blacksquare) \\ \vec{y}=1 \Leftrightarrow \text{are all computable for backward (only } \blacksquare,\blacksquare) \end{cases}$$

Indicator variables

$$\begin{cases} \vec{\omega}=\max_{i=0}^{\mathrm{N_{row}}-1}(\omega_i^I), \\ \sum_{i=0}^{\mathrm{N_{row}}-1} x_i^I - \mathrm{N_{row}}\cdot\vec{x} \geq 0, \\ \sum_{i=0}^{\mathrm{N_{row}}-1} x_i^I - \vec{x} \leq \mathrm{N_{row}} - 1. \\ \sum_{i=0}^{\mathrm{N_{row}}-1} y_i^I - \mathrm{N_{row}}\cdot\vec{y} \geq 0, \\ \sum_{i=0}^{\mathrm{N_{row}}-1} y_i^I - \vec{y} \leq \mathrm{N_{row}} - 1. \end{cases}$$

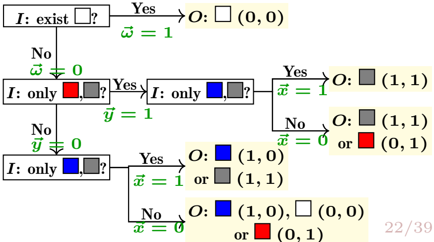
Attribute-propagation through MC

$$\begin{cases} \sum_{i=0}^{\mathrm{N_{row}}-1} x_i^O + \mathrm{N_{row}}\cdot\vec{\omega} \leq \mathrm{N_{row}}, \\ \sum_{i=0}^{\mathrm{N_{row}}-1} (x_i^O + x_i^I) - \mathrm{N_{iosum}}\cdot\vec{x} \geq 0, \\ \sum_{i=0}^{\mathrm{N_{row}}-1} (x_i^O + x_i^I) - \mathtt{Br_n}\cdot\vec{x} \leq \mathrm{N_{iosum}} - \mathtt{Br_n}, \\ \sum_{i=0}^{\mathrm{N_{row}}-1} y_i^O + \mathrm{N_{row}}\cdot\vec{\omega} \leq \mathrm{N_{row}}, \\ \sum_{i=0}^{\mathrm{N_{row}}-1} (y_i^O + y_i^I) - \mathrm{N_{iosum}}\cdot\vec{y} \geq 0, \\ \sum_{i=0}^{\mathrm{N_{row}}-1} (y_i^O + y_i^I) - \mathtt{Br_n}\cdot\vec{y} \leq \mathrm{N_{iosum}} - \mathtt{Br_n}. \end{cases}$$

Canceling impact by consuming DoF

$$\begin{cases} \sum_{i=0}^{\mathrm{N_{row}}-1} x_i^O - \mathrm{N_{row}}\cdot\vec{x} - c_{\vec{y}} = 0, \\ \sum_{i=0}^{\mathrm{N_{row}}-1} y_i^O - \mathrm{N_{row}}\cdot\vec{y} - c_{\vec{x}} = 0. \end{cases}$$

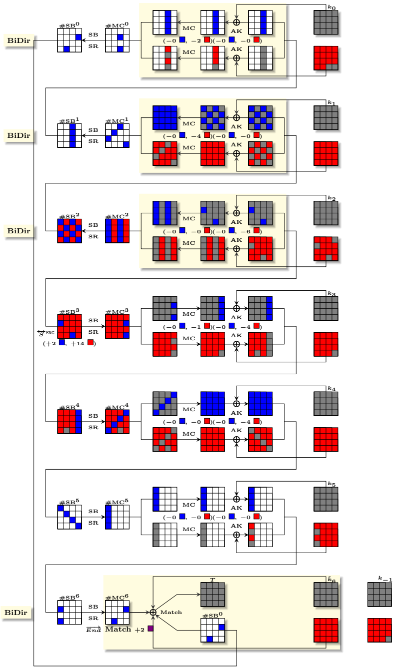
$(x_0^I, y_0^I, \cdots, x_{\mathrm{N_{row}}-1}^I, y_{\mathrm{N_{row}}-1}^I, \vec{\omega}, \vec{x}, \vec{y}) \rightarrow (x_0^O, y_0^O, \cdots, x_{\mathrm{N_{row}}-1}^O, y_{\mathrm{N_{row}}-1}^O)$



Notes on BiDir

Bi-directional attribute-propagation and cancellation (BiDir)

- With BiDir, the computation is divided not only *horizontally* but also *vertically* (irregularly).
- With BiDir, the selection of neutral bits evolved into the most generalized form.

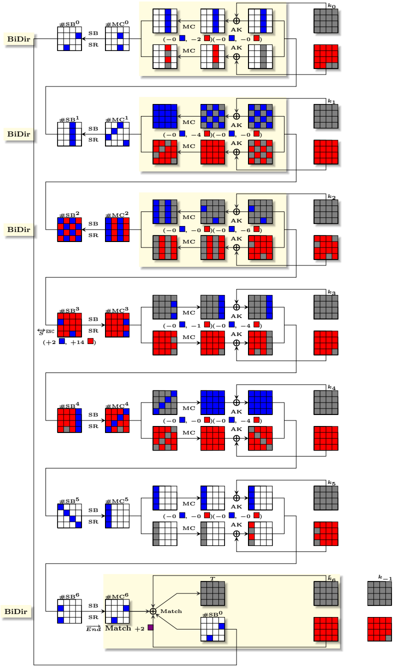


Notes on BiDir

Bi-directional attribute-propagation and cancellation (BiDir)

The evolution of the selection of neutral bits:

- From a cutting point to an initial structure:
  - ⋆ selecting *standard bases* to form the space of initially guessed values;
  - ⋆ selecting *arbitrary bases* to form an affine subspace to be initially guessed values.
- From the initial structure to BiDir:
  - ⋆ selecting an *affine subspace* to be initially guessed values.
  - ⋆ selecting a *non-linearly constrained system of equations*, whose solutions form the space of the initially guessed values



Outline

1. Background and Preliminaries

2. Enhanced MITM-MILP Modeling
   - Superposition states and separate attribute-propagation (SupP)
   - Bi-directional attribute-propagation and cancellation (BiDir)
   - Guess-and-determine (GnD)
   - Multiple ways of `AddRoundKey` (MulAK)
   - Applications to collision and key-recovery attacks

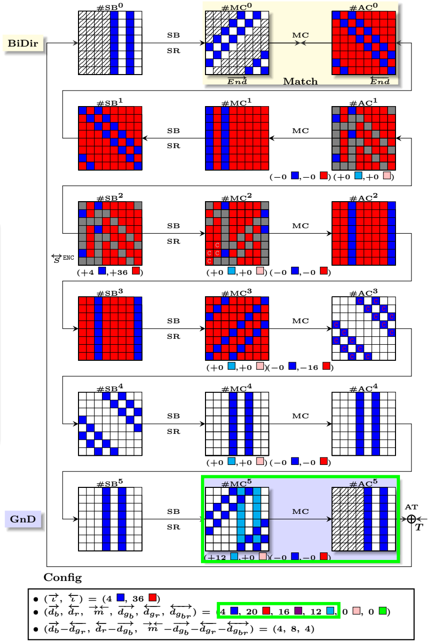
3. Updates on Fundamental Security of AES-like Hashing

4. Conclusions and Future Work

Guess-and-Determine (GnD)

Guess-and-Determine [AC:SWWW12]

- Guess values of a few unknown cells to continue the propagation of attribute to reach the matching point;
- after (partial) matching, check the consistency of the few guessed cells.
- If the gained degree of matching is sufficient and the required guesswork is very little, one can still achieve a better attack complexity than a brute-force attack.



The complexity of the MITM attack with GnD is

$$\varsigma^n \cdot \max(\varsigma^{-(\overleftarrow{d_r}-\overrightarrow{d_{g_b}})},\ \varsigma^{-(\overrightarrow{d_b}-\overleftarrow{d_{g_r}})},\ \varsigma^{-(\overrightarrow{\overleftarrow{m}}-\overrightarrow{d_{g_b}}-\overleftarrow{d_{g_r}}-\overleftrightarrow{d_{g_{br}}})})$$

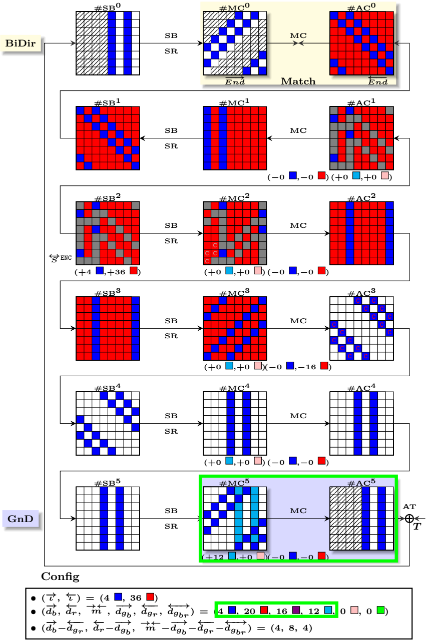
which is determined by

$$\min(\overleftarrow{d_r}-\overrightarrow{d_{g_b}},\ \overrightarrow{d_b}-\overleftarrow{d_{g_r}},\ \ \overrightarrow{\overleftarrow{m}}-\overrightarrow{d_{g_b}}-\overleftarrow{d_{g_r}}-\overleftrightarrow{d_{g_{br}}}).$$

To model the mechanism of GnD

three binary variables, g_b, g_r, g_{br}, are introduced for each cell in the input state of MixColumns (invMixColumns for the backward computation), indicating whether the cells should be guessed.

$$\begin{cases} \overrightarrow{d_{g_b}} = \sum\limits_{r=0,i=0}^{\texttt{total}_r-1,n-1} {g_b}_i^r, \\ \overleftarrow{d_{g_r}} = \sum\limits_{r=0,i=0}^{\texttt{total}_r-1,n-1} {g_r}_i^r, \\ \overleftrightarrow{d_{g_{br}}} = \sum\limits_{r=0,i=0}^{\texttt{total}_r-1,n-1} {g_{br}}_i^r, \end{cases} \qquad \begin{cases} \tau_{\texttt{Obj}} \le \overrightarrow{d_b} - \overleftarrow{d_{g_r}}, \\ \tau_{\texttt{Obj}} \le \overleftarrow{d_r} - \overrightarrow{d_{g_b}}, \\ \tau_{\texttt{Obj}} \le \overrightarrow{\overleftarrow{m}} - \overrightarrow{d_{g_b}} - \overleftarrow{d_{g_r}} - \overleftrightarrow{d_{g_{br}}}. \end{cases}$$



Outline

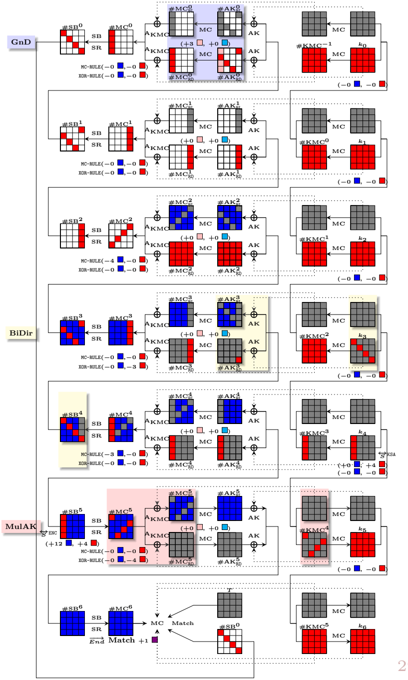
1. Background and Preliminaries
2. Enhanced MITM-MILP Modeling
   - Superposition states and separate attribute-propagation (SupP)
   - Bi-directional attribute-propagation and cancellation (BiDir)
   - Guess-and-determine (GnD)
   - Multiple ways of `AddRoundKey` (MulAK)
   - Applications to collision and key-recovery attacks
3. Updates on Fundamental Security of AES-like Hashing
4. Conclusions and Future Work

Multiple ways of `AddRoundKey` (MulAK)

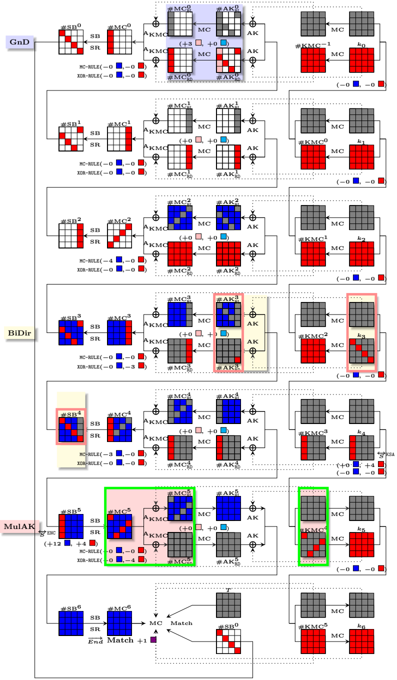
Multiple ways of `AddRoundKey` (MulAK)

- In Whirlpool, the key-schedule shares the same round function with the encryption.
- The AddRoundKey can be easily moved around MixColumns[a] using an equivalent key state (#KMC) already involved in the key-schedule.

[a] For simplicity, we denote MixRows in Whirlpool by MixColumns, and describe the state in its transpose.



Multiple ways of `AddRoundKey` (MulAK)



Multiple ways of `AddRoundKey` (MulAK)

- `AK-MC-RULE`: Moving AddRoundKey before MixColumns and using **#KMC** can bring more advantages in some cases (*e.g.*, round 5).
- `MC-AK-RULE`: It is also possible that adding **#AK** with the real round key $\boldsymbol{k}$ has more advantages than adding **#MC** with **#KMC** (*e.g.*, round 3).
- The integration of both scenarios into one model is in the form of indicator constraints that is available in Gurobi, *e.g.*, `AK-MC-RULE` $= 1 \rightarrow$ **constraints** on **#KMC**, **#MC**, **#AK**, `AK-MC-RULE` $= 0 \rightarrow$ **constraints** on $\boldsymbol{k}$, **#AK**, **#SB**

Outline

1. Background and Preliminaries
2. Enhanced MITM-MILP Modeling
   - Superposition states and separate attribute-propagation (SupP)
   - Bi-directional attribute-propagation and cancellation (BiDir)
   - Guess-and-determine (GnD)
   - Multiple ways of `AddRoundKey` (MulAK)
   - Applications to collision and key-recovery attacks
3. Updates on Fundamental Security of AES-like Hashing
4. Conclusions and Future Work

Applications to collision and key-recovery Attacks

The MILP models for searching for preimage attacks can be directly transformed to search for collision attacks on hash functions and key-recovery attacks on block ciphers, as pointed by Dong *et al.* in [C:DHSLWH21].

- A MITM partial-target preimage attack whose matching point is at the last round can be transformed into a collision attack [FSE:LiIsoShi12];
- Thus, the searching of MITM preimage attacks can be constrained to search for such partial-target preimage attacks, and then translated into a valid collision attack.
- For a valid attack, the following should be fulfilled

$$\left\{\overrightarrow{d_b} - \overleftarrow{d_{g_r}} > \overrightarrow{\overleftarrow{m}}/2, \quad \overleftarrow{d_r} - \overrightarrow{d_{g_b}} > \overrightarrow{\overleftarrow{m}}/2, \quad \overrightarrow{d_{g_b}} + \overleftarrow{d_{g_r}} + \overleftrightarrow{d_{g_{br}}} < \overrightarrow{\overleftarrow{m}}/2\right\}.$$

- To find the best attack, the objective function is the same as that for preimage attack, *i.e.*,

$$\text{maximize} \min\left\{\overrightarrow{d_b} - \overleftarrow{d_{g_r}},\ \overleftarrow{d_r} - \overrightarrow{d_{g_b}},\ \ \overrightarrow{\overleftarrow{m}} - \overrightarrow{d_{g_b}} - \overleftarrow{d_{g_r}} - \overleftrightarrow{d_{g_{br}}}\right\}.$$

Applications to collision and key-recovery Attacks

For key-recovery attacks, upon the MILP models for preimage attack, one simply needs to

- constrain that the degrees of freedom in both forward and backward only source from the key states,
- relax the degrees of matching such that it is not included in the objective but simply be non-zero, and
- constrain that the plaintext or ciphertext contains only `Red` and `Gray` cells or only `Blue` and `Gray` cells. Besides,
- the objective can be set to maximize the number of `Gray` cells in the plaintext or ciphertext, which can reduce the data complexity.

Outline

1. Background and Preliminaries
2. Enhanced MITM-MILP Modeling
   - Superposition states and separate attribute-propagation (SupP)
   - Bi-directional attribute-propagation and cancellation (BiDir)
   - Guess-and-determine (GnD)
   - Multiple ways of `AddRoundKey` (MulAK)
   - Applications to collision and key-recovery attacks
3. Updates on Fundamental Security of AES-like Hashing
4. Conclusions and Future Work

Summary of applications to (pseudo-) preimage attacks

Cipher (Target)	#R	Time-1	Time-2	$(\overrightarrow{d_b}, \overleftarrow{d_r}, \overrightarrow{\overleftarrow{m}}, \overrightarrow{d_{g_b}}, \overleftarrow{d_{g_r}})$	Critical Tech.	Ref.
Whirlpool (Hash)	5/10	2^{416}	2^{448}	$(16, 12, 16, 0, 0)$	Dedicated	**[AC:SWWW12]**
	5/10	$\mathbf{2^{352}}$	$\mathbf{2^{433}}$	$(20, 20, 20, 0, 0)$	MILP, BiDir, MulAK	**[This]**
	6/10	2^{448}	2^{481}	$(32, 8, 32, 0, 24)$	Dedicated, GnD	**[AC:SWWW12]**
	6/10	$\mathbf{2^{440}}$	$\mathbf{2^{477}}$	$(9, 24, 24, 15, 0)$	MILP, GnD	**[This]**
	7/10	$\mathbf{2^{480}}$	$\mathbf{2^{497}}$	$(16, 4, 16, 0, 12)$	MILP, GnD, MulAK	**[This]**
Grøstl-256 (CF+OT)	5/10	2^{192}	$2^{234.67}$	$(8, 8, 8, 0, 0)$	Dedicated	‡ **[IWSEC:MLHL15; JISE:ZouWWD14]**
	5/10	$\mathbf{2^{184}}$	$\mathbf{2^{232}}$	$(9, 9, 16, 0, 0)$	MILP, BiDir	**[This]**
	6/10	2^{240}	2^{252}	$(8, 2, 8, 0, 6)$	Dedicated, GnD	‡ **[IWSEC:MLHL15; JISE:ZouWWD14]**
	6/10	$\mathbf{2^{224}}$	$\mathbf{2^{245.33}}$	$(4, 20, 16, 12, 0)$	MILP, GnD, BiDir	**[This]**
Grøstl-512 (CF+OT)	7/14	2^{416}	2^{480}	$(19, 12, 19, 0, 7)$	MILP, GnD, BiDir	**[This]**
	8/14	2^{472}	2^{504}	$(10, 10, 18, 5, 5)$	Dedicated	† **[JISE:ZouWWD14]**
	8/14	2^{472}	$\mathbf{2^{500}}$	$(9, 5, 10, 0, 4)$	MILP, GnD, BiDir	**[This]**
AES-192 Hashing	9/12	2^{120}	2^{125}	$(1, 1, 1, 0, 0)$	MILP	**[EC:BDGLSSW21]**
	9/12	$\mathbf{2^{112}}$	$\mathbf{2^{121}}$	$(2, 2, 2, 0, 0)$	MILP, SupP, BiDir	**[This]**
Kiasu-BC Hashing	8/10	2^{120}	2^{123}	$(1, 4, 4, 0, 0)$	Dedicated	**[ToSC:BDGWZ19]**
	9/10	$\mathbf{2^{120}}$	$\mathbf{2^{125}}$	$(1, 1, 1, 0, 0)$	MILP, SupP, BiDir	**[This]**

Summary of applications to collision and key-recovery attacks

(Free-start) Collision						
Cipher (Target)	#R	Time	Mem	Setting & Type	Critical Tech.	Ref.
Grøstl-256 (OT)	6/10	2^{124}	2^{124}	classic collision	MILP	[C:DHSLWH21]
	6/10	$\mathbf{2^{116}}$	$\mathbf{2^{116}}$	classic collision	MILP, BiDir	[This]
Grøstl-512 (OT)	8/14	2^{248}	2^{248}	classic collision	MILP	[C:DHSLWH21]
	8/14	$\mathbf{2^{244}}$	$\mathbf{2^{244}}$	classic collision	MILP, BiDir	[This]
AES-128 Hashing	6/10	2^{56}	2^{32}	classic collision	Dedicated	[FSE:GilPey10; AC:LMRRS09]
	7/10	$2^{42.5}$	(2^{48})	quantum collision	Dedicated	[EC:HosSas20]
	7/10	$\mathbf{2^{56}}$	$\mathbf{2^{56}}$	classic free-start	MILP, BiDir	[This]
Key-recovery						
Cipher (Target)	#R	Time	Mem	Data	Critical Tech.	Ref.
SKINNY-64-192	23/40	2^{188}	2^{4}	2^{52}	MILP	[C:DHSLWH21]
	23/40	$\mathbf{2^{184}}$	2^{8}	2^{60}	MILP, SupP	[This]
	23/40	2^{188}	2^{4}	$\mathbf{2^{28}}$	MILP, SupP	[This]
SKINNY-128-384	23/56	2^{376}	2^{8}	2^{104}	MILP	[C:DHSLWH21]
	23/56	$\mathbf{2^{368}}$	2^{16}	2^{120}	MILP, SupP	[This]
	23/56	2^{376}	2^{8}	$\mathbf{2^{56}}$	MILP, SupP	[This]

Outline

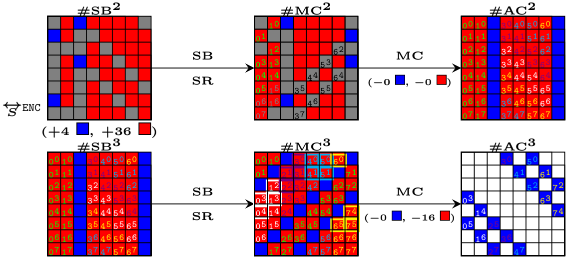
1. Background and Preliminaries
2. Enhanced MITM-MILP Modeling
   - Superposition states and separate attribute-propagation (SupP)
   - Bi-directional attribute-propagation and cancellation (BiDir)
   - Guess-and-determine (GnD)
   - Multiple ways of `AddRoundKey` (MulAK)
   - Applications to collision and key-recovery attacks
3. Updates on Fundamental Security of AES-like Hashing
4. Conclusions and Future Work

Conclusions

- Simple and detailed tricks (SupP, BiDir, GnD, MulAK) are combined with automatic search (MILP), achieving non-negligible improvements:
  - conquers one of the remaining four rounds of the ISO/IEC standard hash function Whirlpool in terms of preimage resistance, and
  - achieves the best classical attacks on round-reduced Grøstl in terms of both preimage and collision resistance.
- The automatic search model has many applications:
  - in terms of the types of attacks: preimage, collision, key-recovery
  - in terms of the targets of attacks: AES hashing modes, Whirlpool, Grøstl, SKINNY, and many other AES-like ciphers.

Future work

- Expand it into an automatic tool to efficiently search a *recursive* MITM procedure with consideration of the computation of initial values of neutral bits.
  - Since the constraints on neutral bits evolved to a much more complicated form, the computation of their initial values becomes not trivial.
  - In some cases, local MITM procedures can be used to compute the initial values of neutral bits, thus, the final attacks can be viewed as *recursive* MITM procedures.



- Improve the efficiency of the search;
- Investigate the security of hashing modes of AES with tweaked key-schedule;
- Adapt it to search for attacks on bit-oriented primitives.

Thanks for your attention!