Computing AES Related-Key Differential Characteristics with Constraint Programming

D. Gérault[(1)], P. Lafourcade[(1)], **M. Minier**[(2)], C. Solnon[(3)]

[(1)] - LIMOS, Université Clermont Auvergne
[(2)] - LORIA, Université de Lorraine
[(3)] - LIRIS, Université de Lyon

Code and Data Protection Day - December 2018

Revisiting AES RKD Characteristics with CP

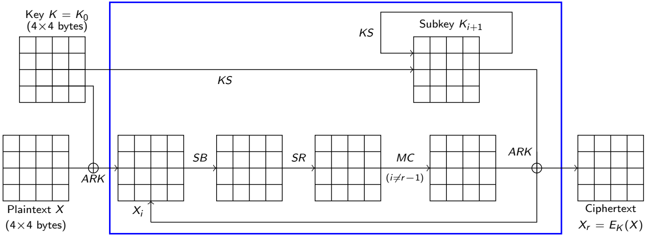
- **Differential cryptanalysis of the AES**
- First CP model for Step 1
- Second CP model for Step 1
- Third CP model for Step 1
- CP model for Step 2
- Results
- Conclusion

AES (Advanced Encryption Standard)

Block cipher standard since 2001

- Input:
  - A plaintext X = 128 bits = 4x4 bytes
  - A key K = 128, 192, or 256 bits = 4x4, 4x6, or 4x8 bytes
- Output: a ciphertext $E_K(X)$ such that $X = E_K^{-1}(E_K(X))$
- Iterative process of r rounds: $r = 10\ (12, 14)$ when $|K| = 128\ (192, 256)$

Operations applied at each round $i \in [0, r-1]$ for AES-128:

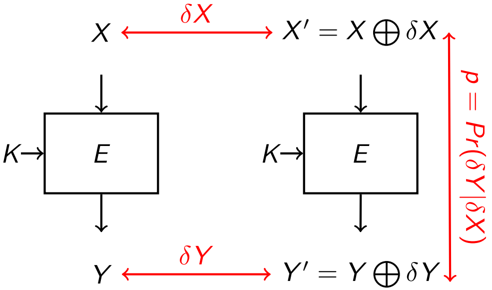


Cryptanalysis of the AES Block Cipher (1/2)

Differential Cryptanalysis [Biham and Shamir 1991]:

Track XOR differences through the ciphering process to recover the key:

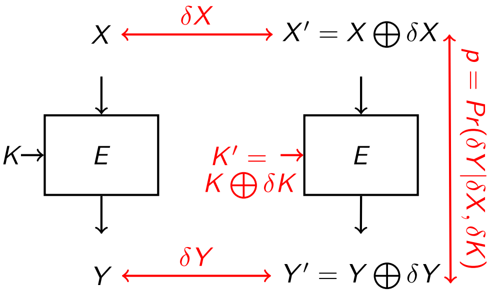
- Let $\delta X = X \oplus X'$ be an input plaintext difference
- Let $\delta Y = E_K(X) \oplus E_K(X')$ be the output difference
- The cipher is weak if $\exists$ δX and δY such that $Pr[\delta Y|\delta X] >> 2^{-|K|}$
  $\rightsquigarrow$ Key recovery in $\mathcal{O}(1/Pr[\delta Y|\delta X])$



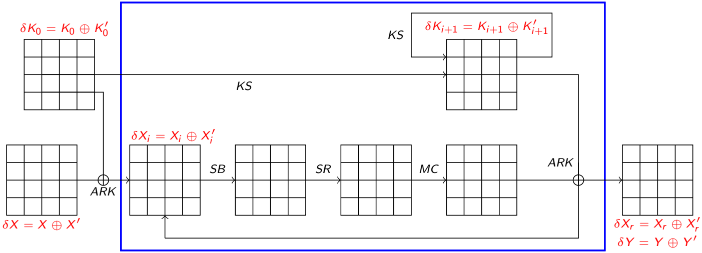
Cryptanalysis of the AES Block Cipher (2/2)

Related-Key Attack [Biham 1993]: Inject differences in texts and keys

- Let $\delta X = X \oplus X'$ be an input plaintext difference
- Let $\delta K = K \oplus K'$ be an input key difference
- Let $\delta Y = E_K(X) \oplus E_{K'}(X')$ be the output difference
- The cipher is weak if $\exists\ \delta X, \delta K$, and δY such that $Pr[\delta Y | \delta X, \delta K] >> 2^{-|K|}$
  $\rightsquigarrow$ Key recovery in $\mathcal{O}(1/Pr[\delta Y | \delta X, \delta K])$



Related-Key Differential of AES



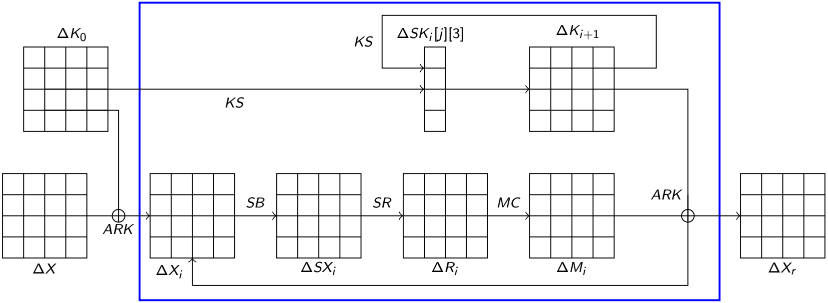
Goal: Find δX, δK_0, and δY that maximizes $Pr[\delta Y|\delta X, \delta K_0]$:

- ARK, SR, and MC are linear: $op(B_i) \oplus op(B_j) = op(B_i \oplus B_j)$
  $\rightsquigarrow$ Probabilities are equal to 1 (or 0) for these operators
- SB is not linear:
  - Let $Pr[\delta_o|\delta_i] = \frac{\#\{(B_1,B_2)\in[0,256]^2 \mid \delta_i = B_1 \oplus B_2 \text{ and } \delta_o = S(B_1) \oplus S(B_2)\}}{256}$
    $\rightsquigarrow$ Probability to have output difference δ_o given input difference δ_i
  - Perfect cipher: $\forall \delta_i, \delta_o, Pr[\delta_o|\delta_i] = \frac{1}{256}$... but this is impossible!
  - SB of AES: if $\delta_o = \delta_i = 0$ then $Pr[\delta_o|\delta_i] = 1$ else $Pr[\delta_o|\delta_i] \in \{0, \frac{2}{256}, \frac{4}{256}\}$

Two step solving process [Biryukov et al. 2010, Fouque et al. 2013]

Step 1: Asbtract differential bytes $\delta B = B \oplus B'$ to booleans ΔB

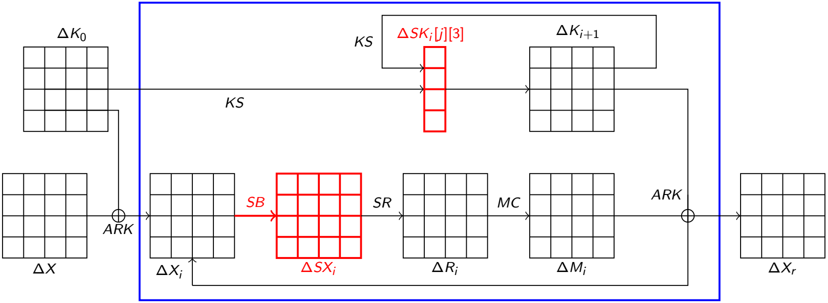
- For each differential byte δB: $\Delta B = 0$ if $\delta B = 0$; $\Delta B = 1$ if $\delta B \in [1, 255]$



Two step solving process [Biryukov et al. 2010, Fouque et al. 2013]

Step 1: Asbtract differential bytes $\delta B = B \oplus B'$ to booleans ΔB

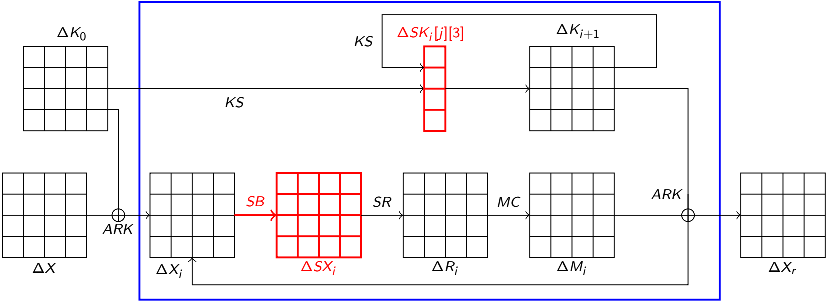
- For each differential byte δB: $\Delta B = 0$ if $\delta B = 0$; $\Delta B = 1$ if $\delta B \in [1, 255]$
- Minimize the nb of boolean variables $\Delta X_i[j][k]$ and $\Delta K_i[j][3]$ set to 1:
  - If $\delta X_i[j][k] = \delta SX_i[j][k] = 0$ then $Pr[\delta SX_i[j][k] | \delta X_i[j][k]] = 1$
  - Otherwise $Pr[\delta SX_i[j][k] | \delta X_i[j][k]] \in \{0, \frac{2}{256}, \frac{4}{256}\}$



Two step solving process [Biryukov et al. 2010, Fouque et al. 2013]

Step 2: Concretize booleans to differential bytes

- If $\Delta B = 0$ then set δB to 0; otherwise search for $\delta B \in [1, 255]$
  - If not possible: Solution byte-inconsistent
  - If possible: Solution byte-consistent
    $\rightsquigarrow$ Maximize the probability $Pr[\delta X_r | \delta X, \delta K_0]$



Existing approaches

Biryukov et al. 2010:

$\rightsquigarrow$ Branch & Bound for Step 1

- $|K| = 128$: Several days of CPU time
- $|K| = 192$: Several weeks of CPU time

Fouque et al. 2013:

$\rightsquigarrow$ Graph traversal for Step 1

- $|K| = 128$: 30mn of CPU time (on 12 cores) but 60 GB of memory
- Not extended to $|K| = 192$ or 256

In both cases: Difficult and time-consuming programming work

$\rightsquigarrow$ Checking the correctness of the program is not straightforward...

What about Constraint Programming (CP)?

Solving a problem with CP:

- Define the problem with a declarative language:
  - Variables (unknowns) and their domains
  - Constraints (relations between variables)
  - Optionally: Objective function to optimize
- Use generic engines to search for solutions

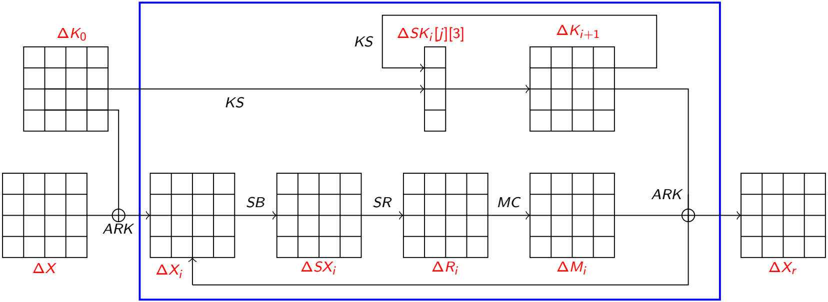
Using CP to compute related-key differentials:

- Less than 5 hours for most of instances
- Less than 15 hours for the hardest instance
- Prove inconsistency of a solution proposed by Biryukov et al. 2010
- New related-key differentials:
  - $|K| = 128$: $p = 2^{-79}$ (instead of 2^{-81}) for 4 rounds
  - $|K| = 192$: $p = 2^{-188}$ for 10 rounds
  - $|K| = 256$: $p = 2^{-146}$ (instead of 2^{-154}) for 14 rounds

Revisiting AES RKD Characteristics with CP

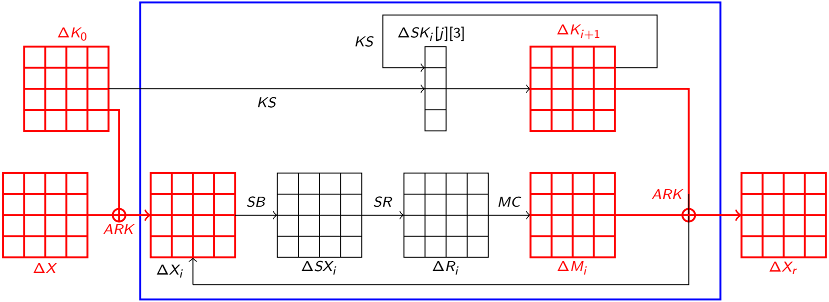
- Differential cryptanalysis of the AES
- **First CP model for Step 1**
- Second CP model for Step 1
- Third CP model for Step 1
- CP model for Step 2
- Results
- Conclusion

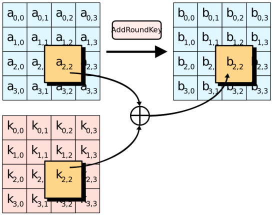
CP_{Basic}: First CP model for Step 1



- For each round i, for each row j and each column k: $\Delta X[j][k], \Delta X_i[j][k], \Delta SX_i[j][k], \Delta R_i[j][k], \Delta M_i[j][k], \Delta K_i[j][k], \Delta SK_i[j][3]$
- Boolean variables $\rightsquigarrow$ Domains $= \{0, 1\}$

CP_{Basic}: First CP model for Step 1





ARK performs XOR operations:

- $\forall j,k \in [0,3]: XOR(\Delta X[j][k], \Delta K_0[j][k], \Delta X_0[j][k])$
- $\forall i \in [0, r-1], \forall j,k \in [0,3]:$ $XOR(\Delta M_i[j][k], \Delta K_{i+1}[j][k], \Delta X_{i+1}[j][k])$

CP_{Basic}: First CP model for Step 1

XOR at the byte level: $\delta B_1 \oplus \delta B_2 \oplus \delta B_3 = 0$

$$
\begin{array}{lllllll}
(\delta B_1, \delta B_2, \delta B_3) \in & \{(0, & 0, & 0)\} & & \\
\cup & \{(0, & x, & x) & | & x \in [1,255]\} \\
\cup & \{(x, & 0, & x) & | & x \in [1,255]\} \\
\cup & \{(x, & x, & 0) & | & x \in [1,255]\} \\
\cup & \{(x, & y, & z) & | & x, y, z \in [1,255], x \neq y \neq z\}
\end{array}
$$

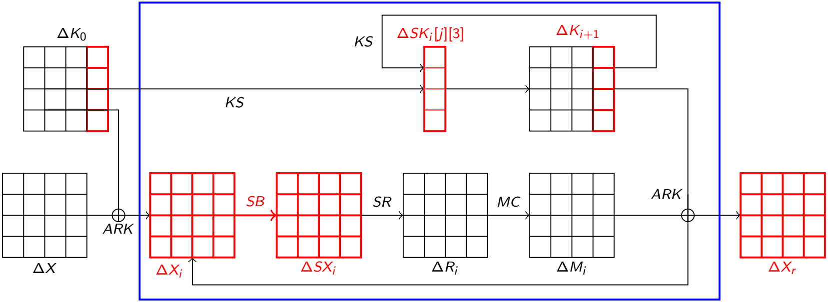
XOR at the boolean level:

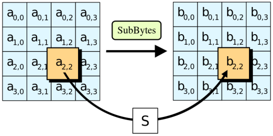
$$
\begin{array}{llll}
(\Delta B_1, \Delta B_2, \Delta B_3) \in \{ & (0, & 0, & 0), \\
& (0, & 1, & 1), \\
& (1, & 0, & 1), \\
& (1, & 1, & 0), \\
& (1, & 1, & 1)\}
\end{array}
$$

Definition of the $XOR(\Delta B_1, \Delta B_2, \Delta B_3)$ constraint:

$$\Delta B_1 + \Delta B_2 + \Delta B_3 \neq 1$$

CP_{Basic}: First CP model for Step 1

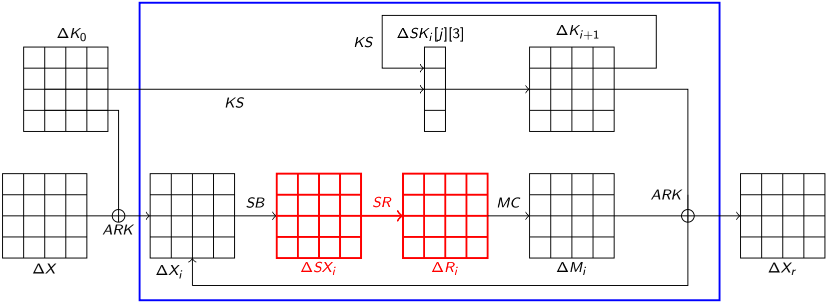


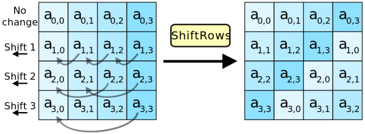


SubBytes does not introduce nor remove differences (because $B_i \oplus B_j = 0 \Leftrightarrow S(B_i) \oplus S(B_j) = 0$)

- $\forall i \in [0, r], \forall j, k \in [0, 3]$: $\Delta X_i[j][k] = \Delta SX_i[j][k]$
- $\forall i \in [0, r], \forall j \in [0, 3]$: $\Delta K_i[j][3] = \Delta SK_i[j][3]$

CP_{Basic}: First CP model for Step 1

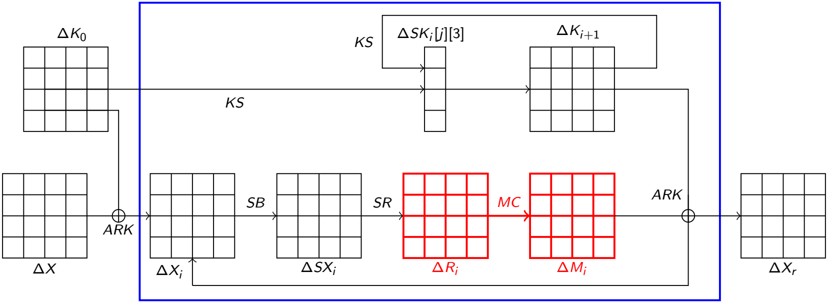


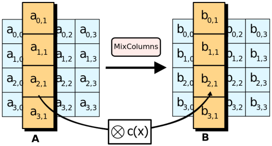


SR shifts bytes: $\forall i \in [0, r-1], \forall j, k \in [0, 3]$:

$$\Delta R_i[j][k] = \Delta SX_i[j][k + j\%4]$$

CP_{Basic}: First CP model for Step 1

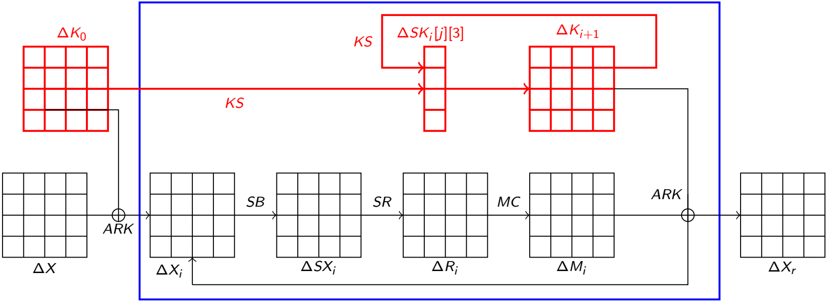


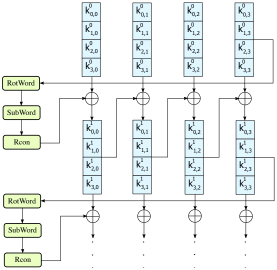


- MC multiplies each column by a fixed matrix
- Ensures the MDS property:
  $\forall i \in [0, r-1], \forall k \in [0, 3]$

$$\sum_{j=0}^{3} \Delta R_i[j][k] + \Delta M_i[j][k] \in \{0, 5, 6, 7, 8\}$$

CP_{Basic}: First CP model for Step 1



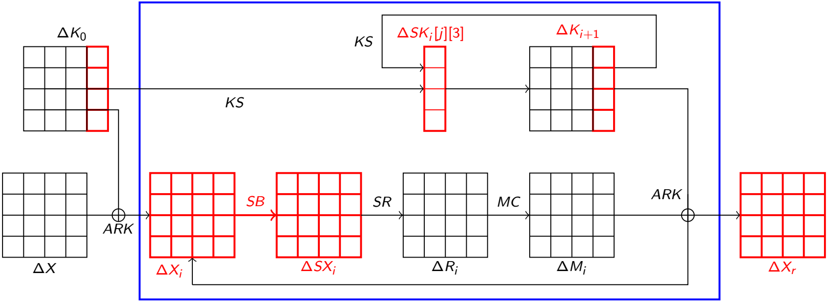


KS performs XOR, byte shifts, and SB operations

For AES-128: $\forall i \in [0, r-1], \forall j \in [0,3]$:

- Column 0:
  $XOR(\Delta K_{i-1}[j][0], \Delta SK_{i-1}[(j+1)\%4][3], \Delta K_i[j][0])$
- Columns $k \in [1,3]$:
  $XOR(\Delta K_{i-1}[j][k], \Delta K_i[j][k-1], \Delta K_i[j][k])$

CP_{Basic}: First CP model for Step 1



Goal: Minimize the number of differences that pass through SubBytes:

$$obj_{Step1} = \sum_{i=0}^{r-1} \sum_{j=0}^{3} (\Delta K_i[j][3] + \sum_{k=0}^{3} \Delta X_i[j][k])$$

Ordering heuristics:

- First choose variables that occur in the objective function

CP_{Basic}: Limitations

- BUT too many binary solutions that are NOT byte-consistent
- Example: $r = 4$, $obj_{Step1} = 11 \rightsquigarrow$ 90 millions of Boolean solutions, none byte-consistent

Revisiting AES RKD Characteristics with CP

- Differential cryptanalysis of the AES
- First CP model for Step 1
- **Second CP model for Step 1**
- Third CP model for Step 1
- CP model for Step 2
- Results
- Conclusion

CP_{EQ}: Second CP model for Step 1

What's wrong with CP_{Basic}?

XOR constraints do not propagate equality relationships at the byte level

- For example, if $\delta a \oplus \delta b \oplus \delta c = 0$ and $\delta a \oplus \delta b \oplus \delta d = 0$ then $\delta c = \delta d$
- However, at the boolean level, we only propagate: $\Delta A + \Delta B + \Delta C \neq 1$ and $\Delta A + \Delta B + \Delta D \neq 1$

New variables and constraints to model byte equalities:

- For each couple of differential bytes $(\delta A, \delta B)$:
  - $EQ_{\delta A, \delta B} = 1$ if $\delta A = \delta B$
  - $EQ_{\delta A, \delta B} = 0$ if $\delta A \neq \delta B$
- Symmetry: $EQ_{\delta A, \delta B} = EQ_{\delta B, \delta A}$
- Transitivity: $EQ_{\delta A, \delta B} = EQ_{\delta B, \delta C} = 1 \Rightarrow EQ_{\delta A, \delta C} = 1$
- Relation with Δ variables:
  - $EQ_{\delta A, \delta B} = 1 \Rightarrow \Delta A = \Delta B$
  - $EQ_{\delta A, \delta B} = 0 \Rightarrow \Delta A + \Delta B \neq 0$

CP_{EQ}: Second CP model for Step 1

Definition of XOR in CP_{Basic}: $\Delta B_1 + \Delta B_2 + \Delta B_3 \neq 1$

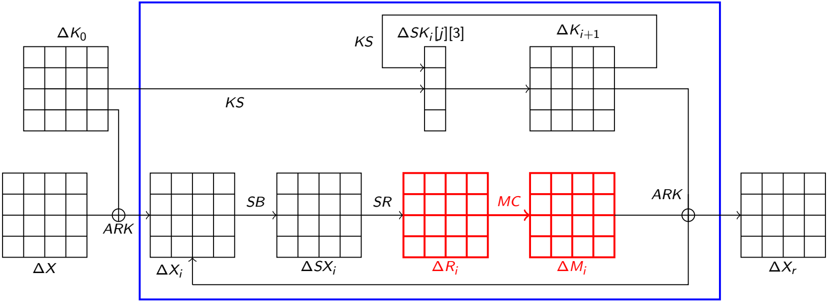
Can we strengthen it by exploiting byte equalities?
Yes, because:

- $\Delta B_1 = 0 \Leftrightarrow \delta B_2 = \delta B_3$
- $\Delta B_2 = 0 \Leftrightarrow \delta B_1 = \delta B_3$
- $\Delta B_3 = 0 \Leftrightarrow \delta B_1 = \delta B_2$

New definition of XOR:

$$\begin{aligned}
\mathtt{XOR}(\Delta B_1, \Delta B_2, \Delta B_3) \Leftrightarrow \quad & ((\Delta B_1 + \Delta B_2 + \Delta B_3 \neq 1) \\
\wedge \quad & (EQ_{\delta B_1, \delta B_2} = 1 - \Delta B_3) \\
\wedge \quad & (EQ_{\delta B_1, \delta B_3} = 1 - \Delta B_2) \\
\wedge \quad & (EQ_{\delta B_2, \delta B_3} = 1 - \Delta B_1))
\end{aligned}$$

CP_{EQ}: Second CP model for Step 1



MDS also holds when XORing different columns of δR and δM:

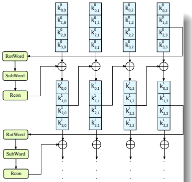
$\forall i_1, i_2 \in [0, r-1], \forall k_1, k_2 \in [0, 3]$, the number of bytes equal to 0 in

$$\delta R_{i_1}[j][k_1] \oplus \delta R_{i_2}[j][k_2] \text{ and } \delta M_{i_1}[j][k_1] \oplus \delta M_{i_2}[j][k_2] \in \{0, 1, 2, 3, 8\}$$

New constraints to ensure MDS: $\forall i_1, i_2 \in [0, r-1], \forall k_1, k_2 \in [0, 3]$

$$\sum_{j=0}^{3} EQ_{\delta R_{i_1}[j][k_1], \delta R_{i_2}[j][k_2]} + EQ_{\delta M_{i_1}[j][k_1], \delta M_{i_2}[j][k_2]} \in \{0, 1, 2, 3, 8\}$$

CP_{EQ}: Second CP model for Step 1



KS (mainly) performs XOR operations:

- Column 0: $K_i[j][0] = K_{i-1}[j][0] \oplus SK_{i-1}[(j+1)\%4][3]$
- Columns $k \in [1,3]$: $K_i[j][k] = K_i[j][k-1] \oplus K_{i-1}[j][k]$

$\rightsquigarrow$ Each byte of K_i is eq. to a XOR of bytes of K_0 and SK_{i-1}

Ex: $K_2[1][1] = K_2[1][0] \oplus K_1[1][1]$
$= K_1[1][0] \oplus SK_1[2][3] \oplus K_1[1][0] \oplus K_0[1][1] = SK_1[2][3] \oplus K_0[1][1]$

New constraints:

- Pre-compute sets $V_{i,j,k}$ such that $\delta K_i[j][k] = \bigoplus_{\delta B \in V_{i,j,k}} \delta B$
- Introduce set variables $S_{i,j,k}$ and post the following constraints:
  - $S_{i,j,k} = \{\delta B \in V_{i,j,k} | \Delta B = 1\}$
  - If $S_{i,j,k} = \emptyset$ then $\Delta K_i[j][k] = 0$
  - If $S_{i,j,k} = \{\delta B\}$ then $EQ_{\delta K_i[j][k], \delta B} = 1$
  - If $S_{i,j,k} = \{\delta B_1, \delta B_2\}$ then $XOR(\Delta B_1, \Delta B_2, \Delta K_i[j][k])$
  - If $\exists i', j', k'$ s.t. $S_{i,j,k} = S_{i',j',k'}$ then $EQ_{\delta K_i[j][k], \delta K_{i'}[j'][k']} = 1$

Revisiting AES RKD Characteristics with CP

- Differential cryptanalysis of the AES
- First CP model for Step 1
- Second CP model for Step 1
- **Third CP model for Step 1**
- CP model for Step 2
- Results
- Conclusion

CP_{XOR}: Third CP model for Step 1

Key Schedule Modeling

- Generate all possible equations from the key schedule with 2 or 3 XORs: sets called *XOReq*
- All those equations could be generated from the original equations with 2 or 3 XORs
- for AES-128, 1104 equations; for AES-192, 1696 equations; for AES-256, 1256 equations;
- Keep all the constraints of CP_{EQ} and add the following constraints:
  - $\forall(\delta B_1 \oplus \delta B_2 \oplus \delta B_3 = 0) \in XOReq$:
    $EQ_{\delta B_1,\delta B_2} = 1 - \Delta B_3) \wedge (EQ_{\delta B_1,\delta B_3} = 1 - \Delta B_2) \wedge (EQ_{\delta B_2,\delta B_3} = 1 - \Delta B_1)$
  - $\forall(\delta B_1 \oplus \delta B_2 \oplus \delta B_3 \oplus \delta B_4 = 0) \in XOReq$:
    $EQ_{\delta B_1,\delta B_2} = EQ_{\delta B_3,\delta B_4} \wedge EQ_{\delta B_1,\delta B_3} = EQ_{\delta B_2,\delta B_4} \wedge EQ_{\delta B_1,\delta B_4} = EQ_{\delta B_2,\delta B_3}$

Revisiting AES RKD Characteristics with CP

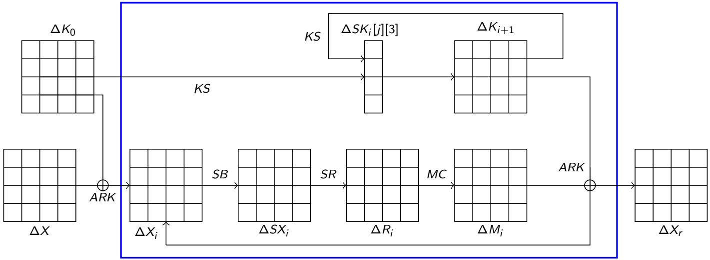
- Differential cryptanalysis of the AES
- First CP model for Step 1
- Second CP model for Step 1
- Third CP model for Step 1
- **CP model for Step 2**
- Results
- Conclusion

CP model for Step 2

1. Initialize Obj_{Step1} to 1
2. Step 1: Search for all boolean solutions
3. For each boolean solution of Step 1 for values of ΔX_i and of $\Delta K_i[j][3]$:
   - Step 2: Search for byte values that maximize $Pr[\delta X_r | \delta X, \delta K_0]$ (or detect inconsistency and set Pr to 0)

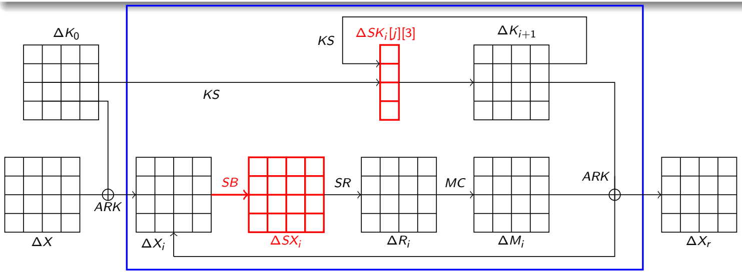
$\rightsquigarrow$ Let Pr_{max} be the largest probability wrt all boolean solutions of Step 1

4. If $Pr_{max} < 2^{-6(Obj_{Step1}+1)}$ then increment Obj_{Step1} and go to (2)
   Otherwise, return Pr_{max}



CP model for Step 2

- For each boolean variable ΔB: Integer variable δB
  - If $\Delta B = 0$ in the Step 1 solution then: $D(\delta B) = \{0\}$
  - Otherwise: $D(\delta B) = [1, 255]$
- For each byte A on which SB is applied: Integer variable P_A
  $\rightsquigarrow$ Base 2 logarithm of $\Pr(\delta SA|\delta A)$
  - If $\Delta A = \Delta SA = 0$ then: $D(P_A) = \{0\}$ because $\Pr(0|0) = 1$
  - Otherwise: $D(P_A) = \{-7, -6\}$ because $Pr(\delta SA|\delta A) \in \{\frac{2}{256}, \frac{4}{256}\}$
- Objective function: Maximize $obj_{Step2} = \sum_{A \text{ on which SB is applied}} P_A$



CP model for Step 2

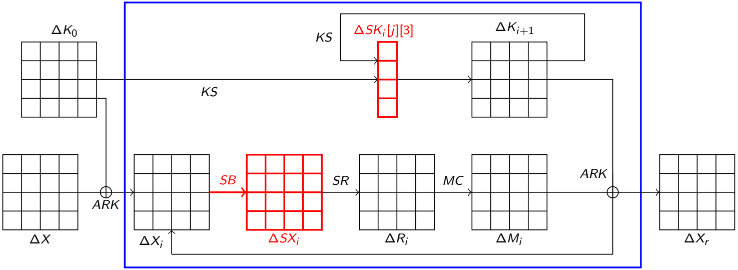
Table constraint related to SB:

For each byte A on which SB is applied:

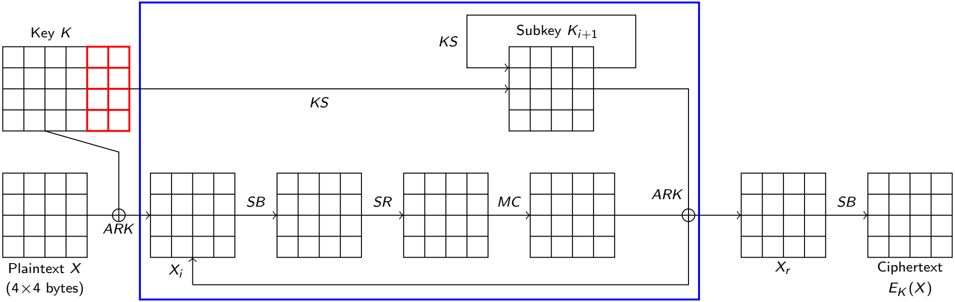
$$(\delta A, \delta SA, P_A) \in \{(X, Y, P) | \quad \exists (B_1, B_2) \in [0,255] \times [0,255], X = B_1 \oplus B_2, \\ Y = S(B_1) \oplus S(B_2), P = \log_2(\Pr(Y|X))\}$$

Constraints related to KS, ARK, SR, and MC:

$\rightsquigarrow$ Straightforward definition with table constraints



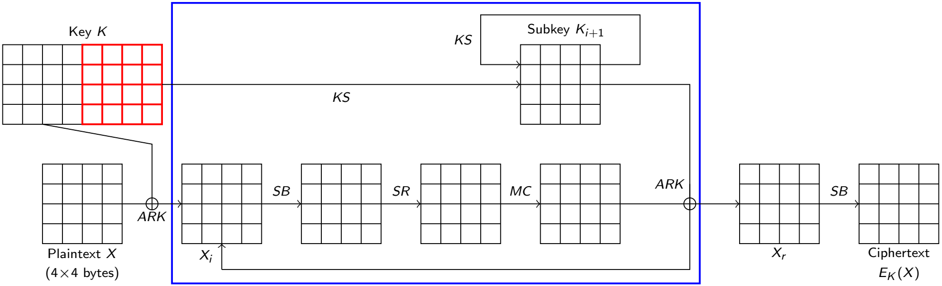
Extension to AES-192 and AES-256



Update constraints related to KeySchedule:

- Step 1: XOR constraints combined with byte shifts
- Step 2: XOR constraints combined with byte shifts
  + SubBytes on some columns

Extension to AES-192 and AES-256



Update constraints related to KeySchedule:

- Step 1: XOR constraints combined with byte shifts
- Step 2: XOR constraints combined with byte shifts
  + SubBytes on some columns

Revisiting AES RKD Characteristics with CP

- Differential cryptanalysis of the AES
- First CP model for Step 1
- Second CP model for Step 1
- Third CP model for Step 1
- CP model for Step 2
- **Results**
- Conclusion

Experimental setup

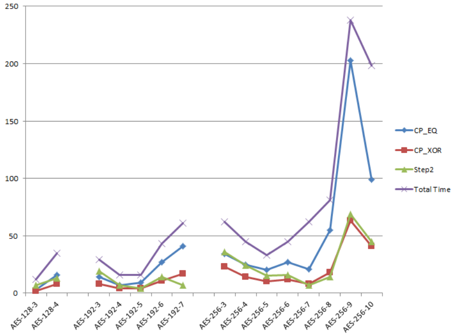
Languages and Solvers

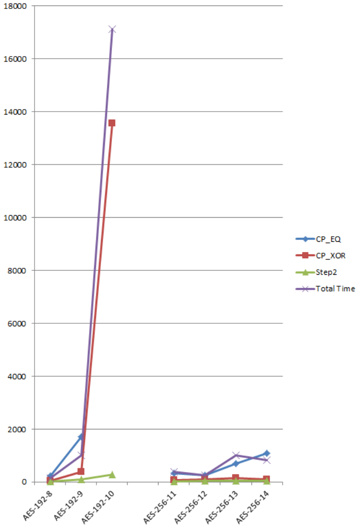
- CP models for Step 1 implemented in MiniZinc
  - Benchmark for the 2016 MiniZinc Challenge
  - Best results are obtained with Picat-Sat
- The CP model for Step 2 is defined in Choco 3 (Java CP library)

Time to solve the hardest instances

- Less than 5 hours for all instances EXCEPT AES-128-5
- AES-128-5 solved in 15 hours

Experimental Results: time (in seconds)





Experimental Results: Nb of solutions

	AES-128			AES-192							
r	3	4	5	3	4	5	6	7	8	9	10
Opt bound	5	12	17	1	4	5	10	13	18	24	29
Nb sol bin	2	1	103	14	2	1	2	1	1	3	7
Nb sol byte	2	1	27	14	2	1	2	1	1	3	7
Best p	2^{-31}	2^{-75}	2^{-105}	2^{-6}	2^{-24}	2^{-30}	2^{-60}	2^{-78}	2^{-108}	2^{-146}	2^{-176}

	AES-256											
r	3	4	5	6	7	8	9	10	11	12	13	14
Opt bound	1	3	3	5	5	10	15	16	20	20	24	24
Nb sol bin	33	10	4	3	1	2	4	1	1	1	1	1
Nb sol byte	33	10	4	3	1	2	4	1	1	1	1	1
Best p	2^{-6}	2^{-18}	2^{-18}	2^{-30}	2^{-30}	2^{-60}	2^{-92}	2^{-98}	2^{-122}	2^{-122}	2^{-146}	2^{-146}

Revisiting AES RKD Characteristics with CP

- Differential cryptanalysis of the AES
- First CP model for Step 1
- Second CP model for Step 1
- Third CP model for Step 1
- CP model for Step 2
- Results
- **Conclusion**

Conclusion (1/2): Better RK Diff Characteristics

AES-192

Attack	Nb rounds	Nb keys	Data	Time	Memory	Source
RK rectangle	10	64	2^{124}	2^{183}	N/A	[Kim et al. 07]
RK amplified boomerang	12	4	2^{123}	2^{176}	2^{152}	[Biryukov et al. 09]
RK distinguisher	10	2^{80}	$2^{108}*$	$2^{108}*$	-	CP
basic RK differential	10	2^{44}	2^{156}	2^{156}	2^{65}	CP

AES-256

Attack	Nb rounds	Nb keys	Data	Time	Memory	Source
RK boomerang	14	4	$2^{99.5}$	$2^{99.5}$	2^{77}	[Biryukov et al. 09]
RK distinguisher	14	2^{35}	$2^{119}*$	$2^{119}*$	-	[Biryukov et al. 09]
basic RK differential	14	2^{35}	2^{131}	2^{131}	2^{65}	[Biryukov et al. 09]
q-multicollisions	14	$2q$	$2q$	$q2^{67}$	-	[Biryukov et al. 09]
RK distinguisher	14	2^{32}	$2^{114}*$	$2^{114}*$	-	CP
basic RK differential	14	2^{32}	2^{125}	2^{125}	2^{65}	CP
q-multicollisions	14	$2q$	$2q$	$q2^{66}$	-	CP

Table: $*$ means for each key.

Conclusion (2/2): go further ?

First Results for Rijndael

block sizes	Key sizes 128	160	192	224	256
128	5, 2^{-105}	8, 2^{-144}	10, 2^{-176}	**13**, 2^{-217}	**14**, 2^{-146}
160	4, 2^{-106}	6, 2^{-138}	9, 2^{-177}	10, 2^{-202}	11, 2^{-198}
192	3, 2^{-54}	5, 2^{-112}	7, 2^{-153}	10, 2^{-222}	9, 2^{-173}
224	3, 2^{-54}	4, 2^{-122}	6, 2^{-160}	7, 2^{-161}	9, 2^{-222}
256	3, 2^{-54}	4, 2^{-121}	5, 2^{-142}	7, 2^{-207}	7, 2^{-172}

Declarative framework for Cryptanalysis?

CP models describe problems, not how to solve them:

- Easier to define and check than a full program
  $\leadsto$ Better solutions than [Biryukov et al 2009] and [Fouque et al 2013]
- Models are defined with the MiniZinc language:
  $\leadsto$ We can use different CP solvers to solve them

Thanks for Your Attention !

Questions ?