Improved Key Recovery Attacks on Reduced-Round AES with Practical Data and Memory Complexities

Achiya Bar-On, Orr Dunkelman, Nathan Keller,
Eyal Ronen, and Adi Shamir

CS 741 Project
160020018
160050017
160050087
160050096

AES

- AES is the best known and most widely used secret key cryptosystem
  - Almost all secure connections on the Internet use AES

- Its security had been analyzed for more than 20 years

- AES has either 10, 12, or 14 rounds depending on the key size (128, 192, 256 bits)

- To date there is no known attack on full AES which is significantly faster than exhaustive search

Reduced round AES

- Interesting as a platform for analyzing the remaining security margins
- Several Light Weight Cryptosystems and Hash functions use 4 or 5 rounds AES as a building block
  - 4-Round AES: ZORRO, LED and AEZ
  - 5-Round AES: WEM, Hound and ELmD

Recent attacks on 5 rounds AES

- In 2017 a new technique (the multiple-of-8 attack [GRR, EC'17]) was proposed, and in 2018 Grassi applied a special version of it (the mixture-differentials attack) to 5 round AES
- However, its complexity was not better than previous attacks
- In this work we improve the 20 year old record to 2^{22}

AES structure

- 10, 12, or 14 rounds, where each round of AES
  (

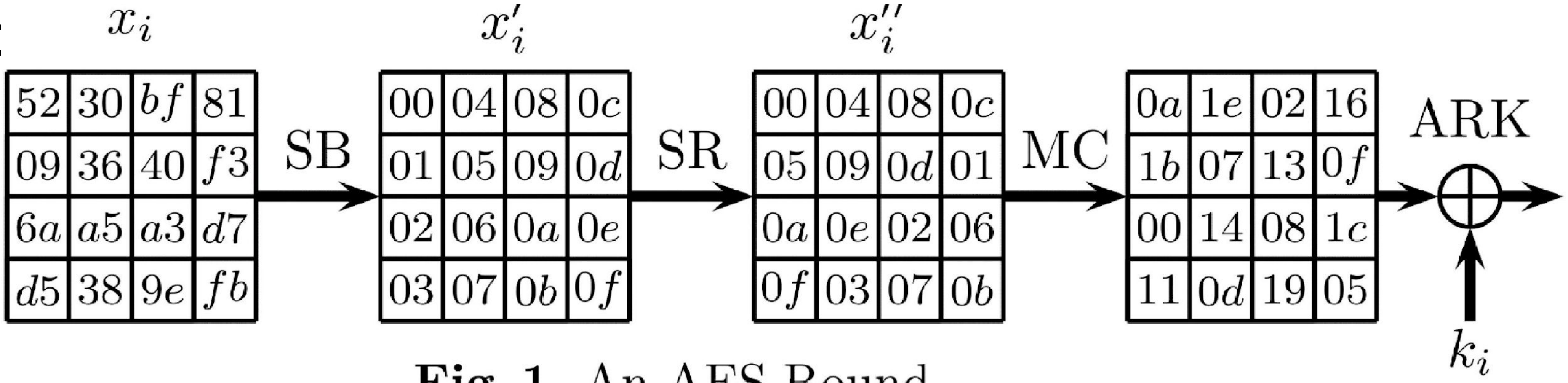
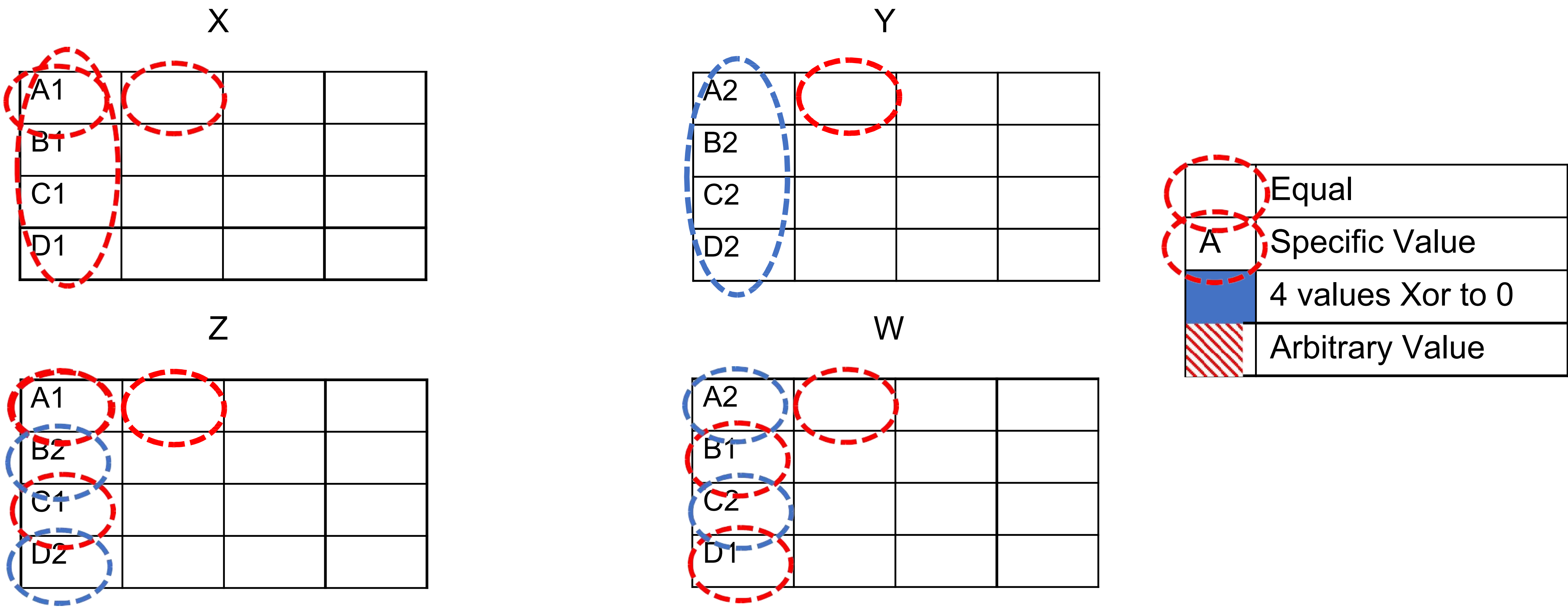


Fig. 1. An AES Round

- Extra ARK operation before the first round
- No Mix Column in the last round

The notation of mixtures (Grassi et. al 2017)

- What is a mixture of an AES state pair (x,y)?



The evolution of mixtures under AES

- Consider the following 4 inputs to round i

X

A1			
B1			
C1			
D1			

Y

A2			
B2			
C2			
D2			

Z

A1			
B2			
C1			
D2			

W

A2			
B1			
C2			
D1			

	Equal
A	Specific Value
	4 values Xor to 0
	Arbitrary Value

The evolution of mixtures under AES

- Round i after Sub Byte

X

A1*			
B1*			
C1*			
D1*			

Y

A2*			
B2*			
C2*			
D2*			

Z

A1*			
B2*			
C1*			
D2*			

W

A2*			
B1*			
C2*			
D1*			

	Equal
A	Specific Value
	4 values Xor to 0
	Arbitrary Value

The evolution of mixtures under AES

- Round i after Shift Rows

X

A1*			
			B1*
		C1*	
	D1*		

Y

A2*			
			B2*
		C2*	
	D2*		

Z

A1*			
			B2*
		C1*	
	D2*		

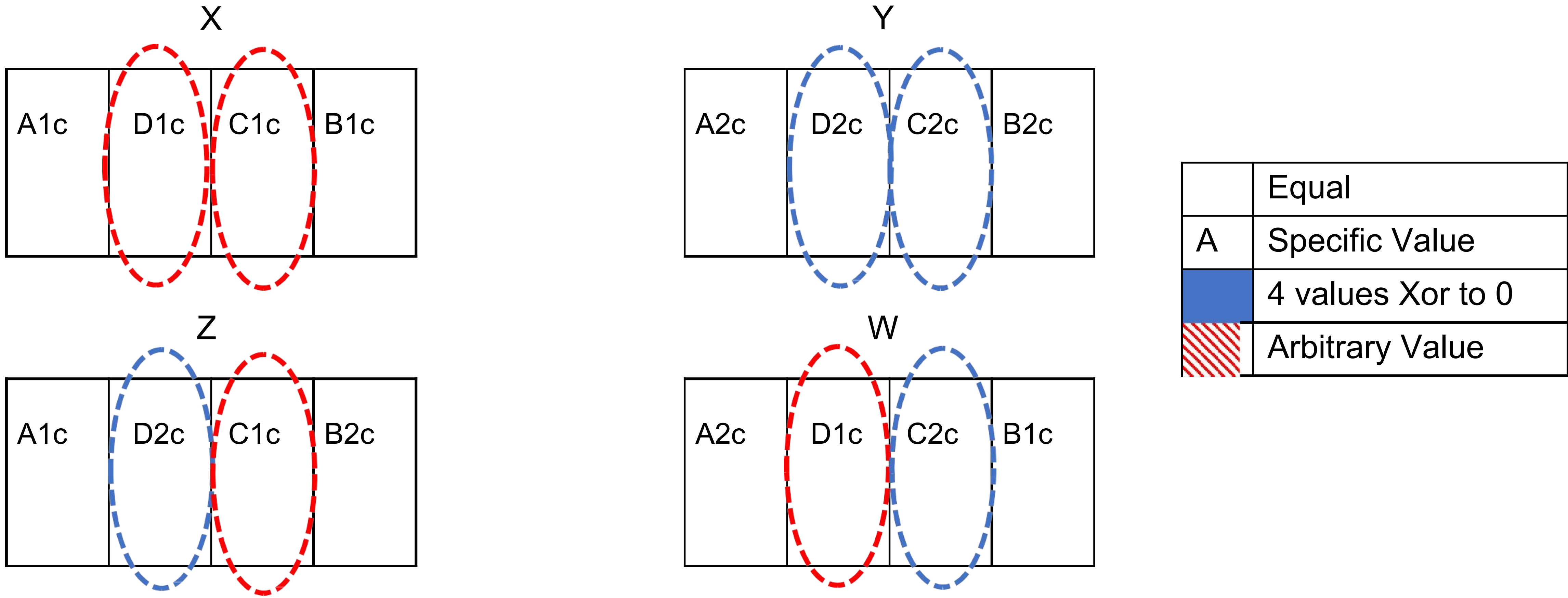
W

A2*			
			B1*
		C2*	
	D1*		

	Equal
A	Specific Value
	4 values Xor to 0
	Arbitrary Value

The evolution of mixtures under AES

- Round i after Mix Column



The evolution of mixtures under AES

- Round i after Add Round Key

X

A1c*	D1c*	C1c*	B1c*

Y

A2c*	D2c*	C2c*	B2c*

Z

A1c*	D2c*	C1c*	B2c*

W

A2c*	D1c*	C2c*	B1c*

	Equal
A	Specific Value
	4 values Xor to 0
	Arbitrary Value

The evolution of mixtures under AES

- Input to round i+1

X

A1c*	D1c*	C1c*	B1c*

Y

A2c*	D2c*	C2c*	B2c*

Z

A1c*	D2c*	C1c*	B2c*

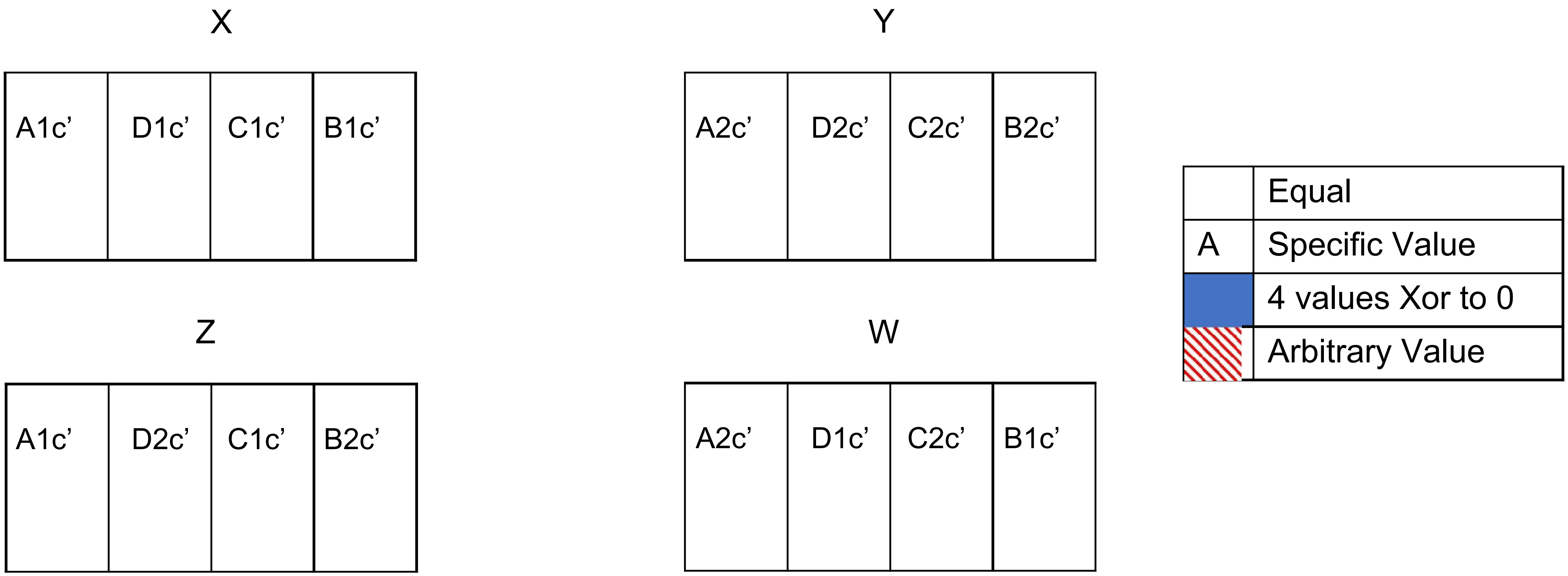
W

A2c*	D1c*	C2c*	B1c*

	Equal
A	Specific Value
	4 values Xor to 0
	Arbitrary Value

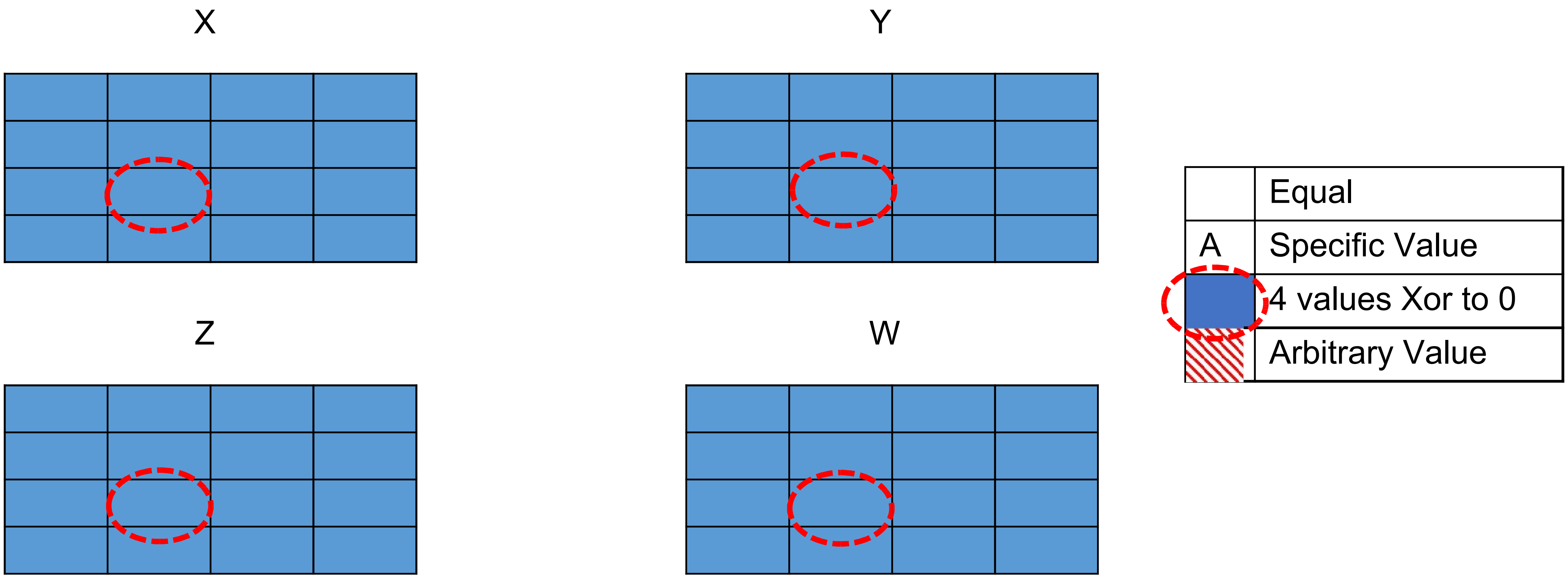
The evolution of mixtures under AES

- Round i+1 after Sub Byte



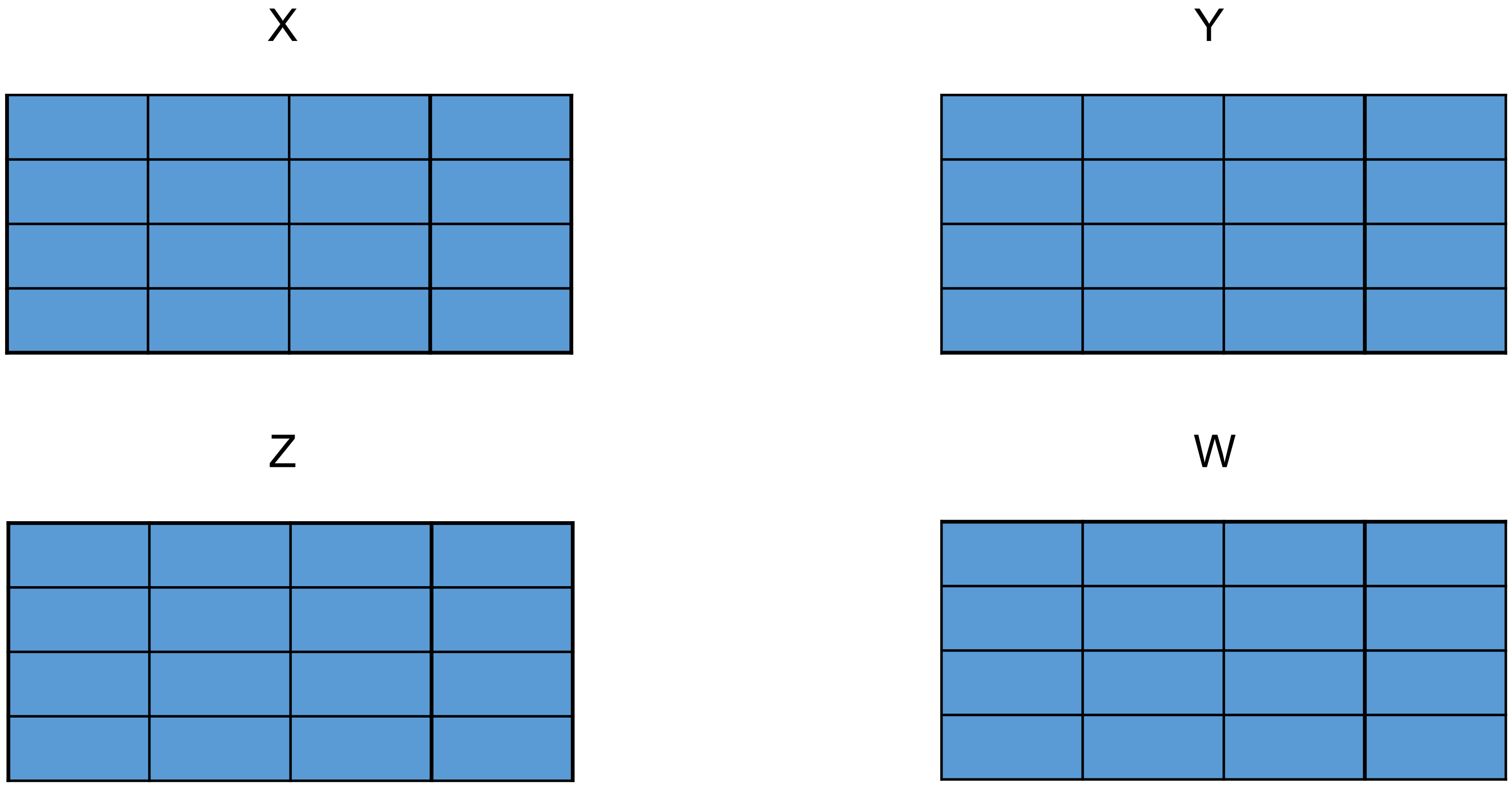
The evolution of mixtures under AES

- Implies weaker property in round i+1 after Sub Byte



The evolution of mixtures under AES

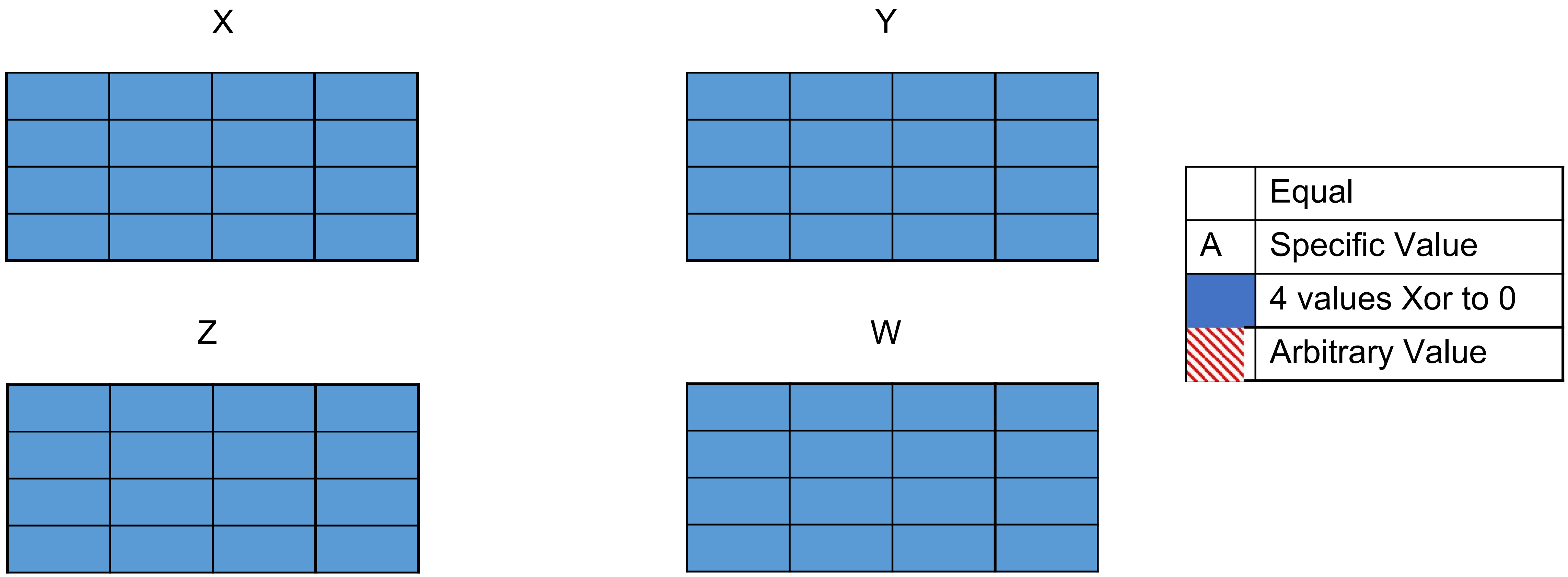
- Round i+1 after Shift Row, Mix Column and ARK



	Equal
A	Specific Value
	4 values Xor to 0
	Arbitrary Value

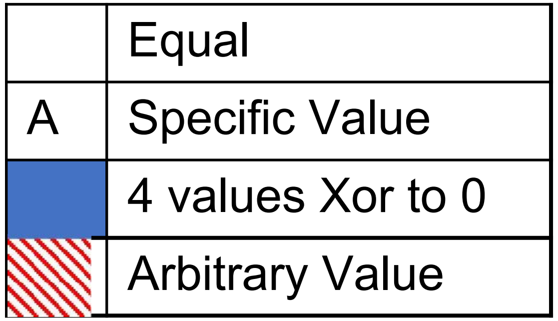
The evolution of mixtures under AES

- Input to round i+2



Extending this property to 4 rounds

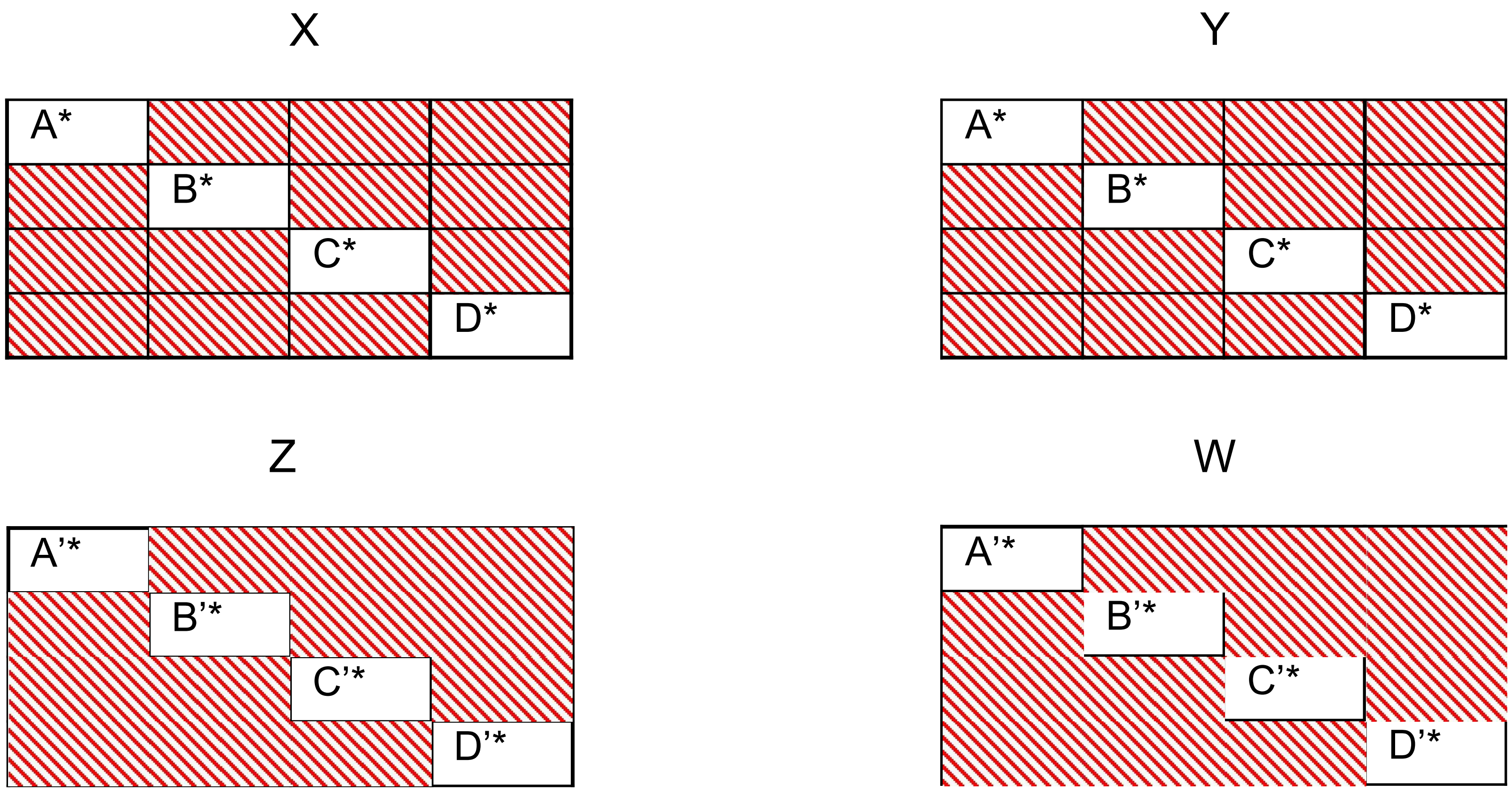
- Assume states (X,Y) are equal in one of their diagonals



- Then :

Extending this property to 4 rounds

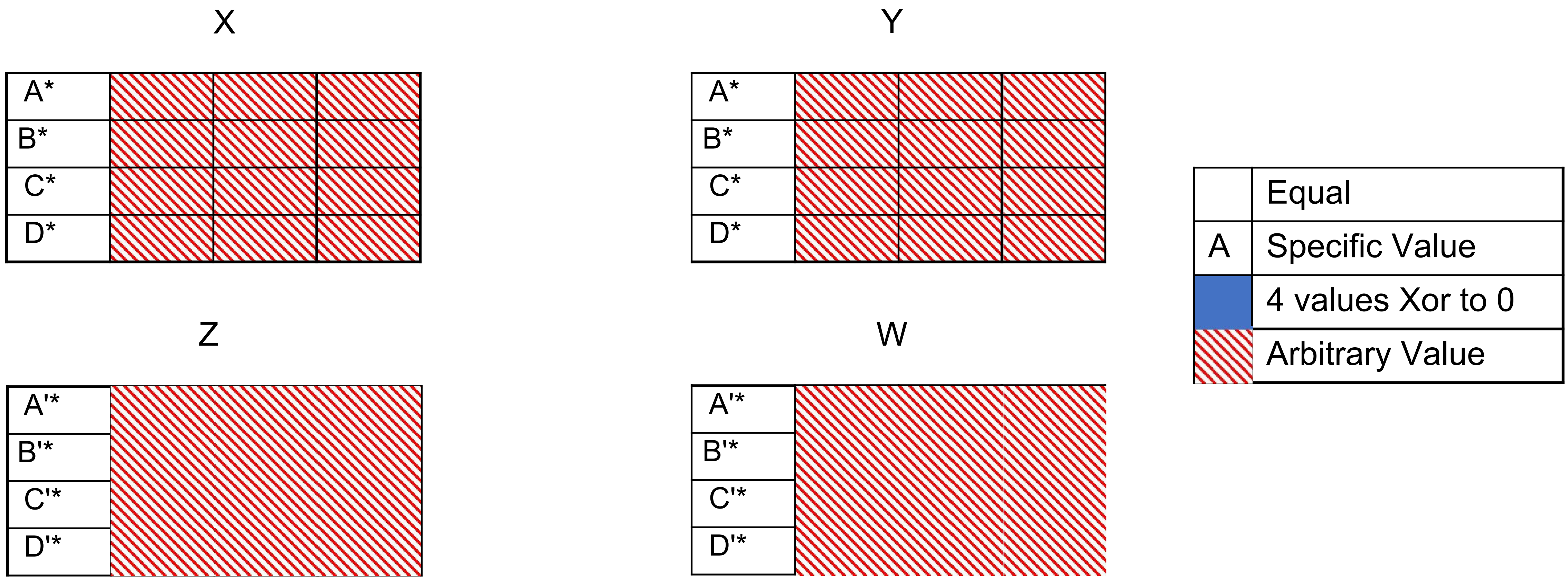
- Round i+2 after Sub Byte



	Equal
A	Specific Value
	4 values Xor to 0
	Arbitrary Value

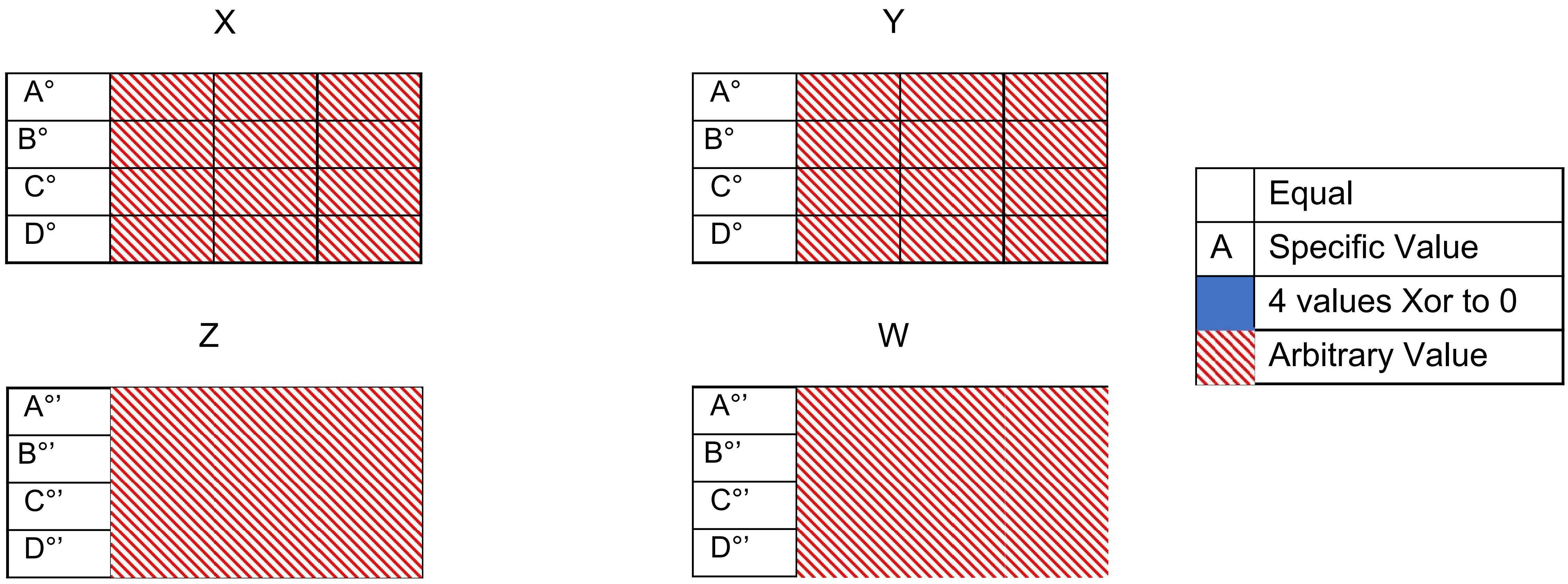
Extending this property to 4 rounds

- Round i+2 after Shift rows



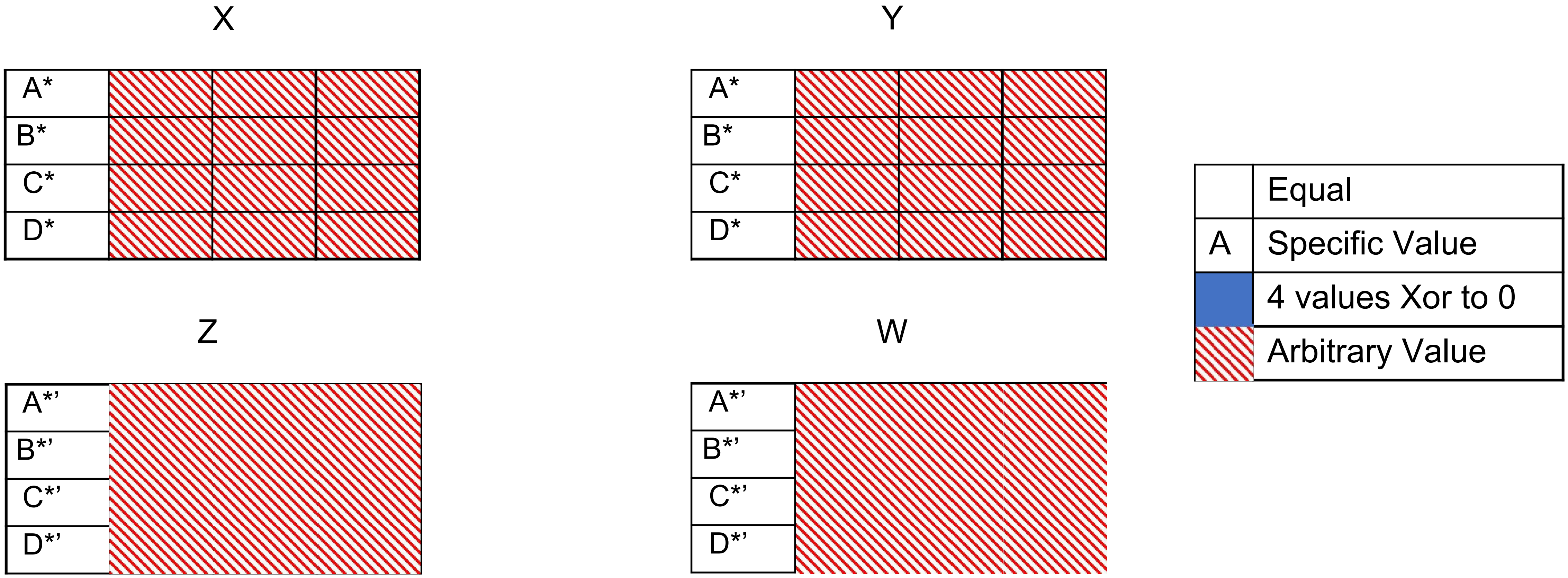
Extending this property to 4 rounds

- Round i+2 after Mix Column



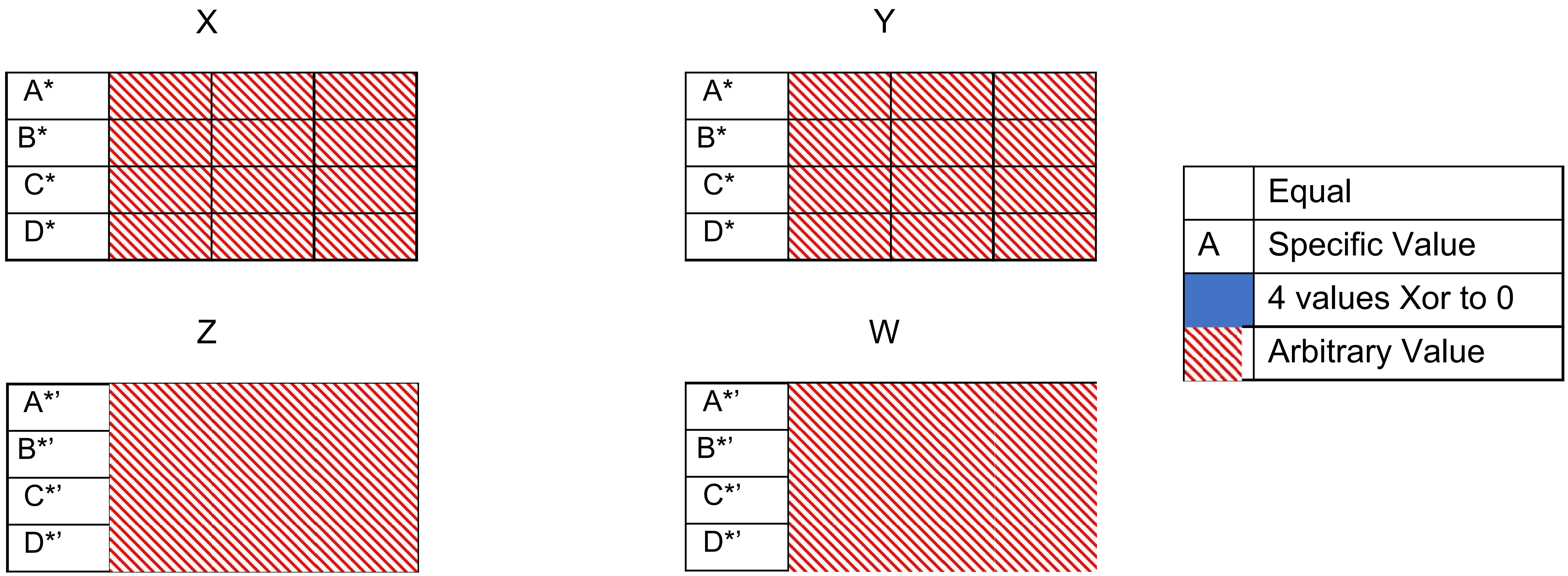
Extending this property to 4 rounds

- Round i+2 after Add Round Key



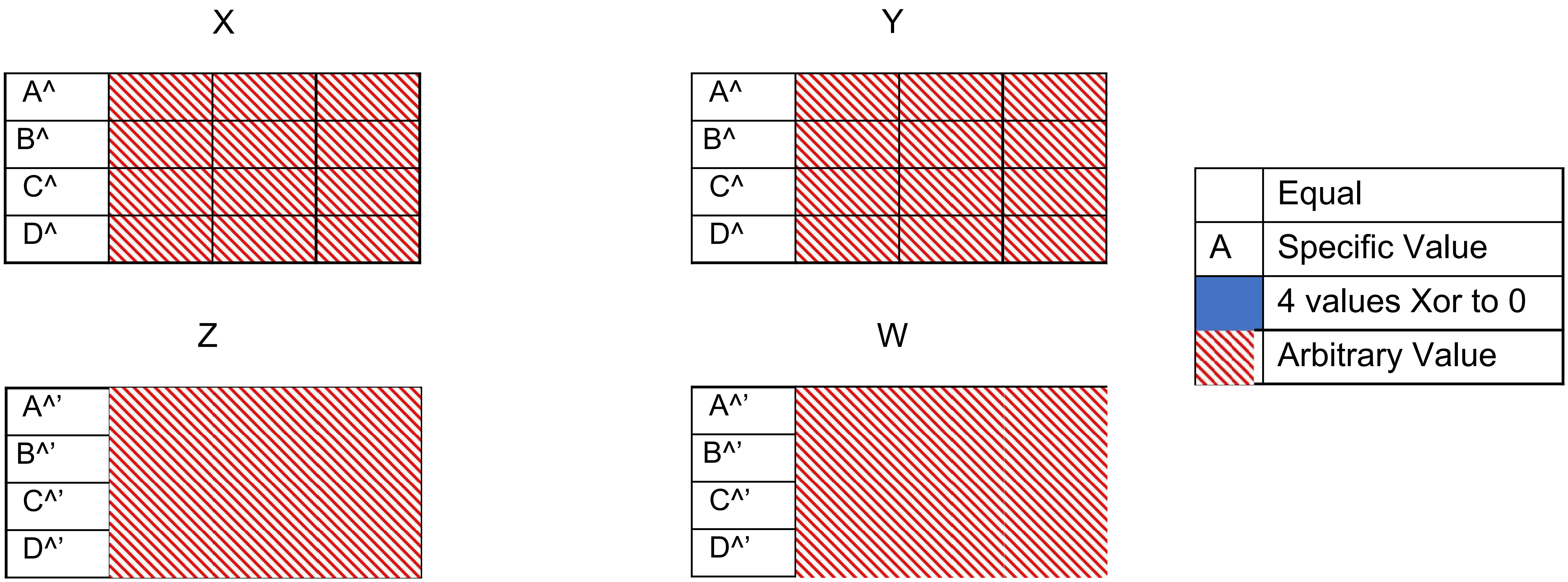
Extending this property to 4 rounds

- Then in the input to round i+3 we get



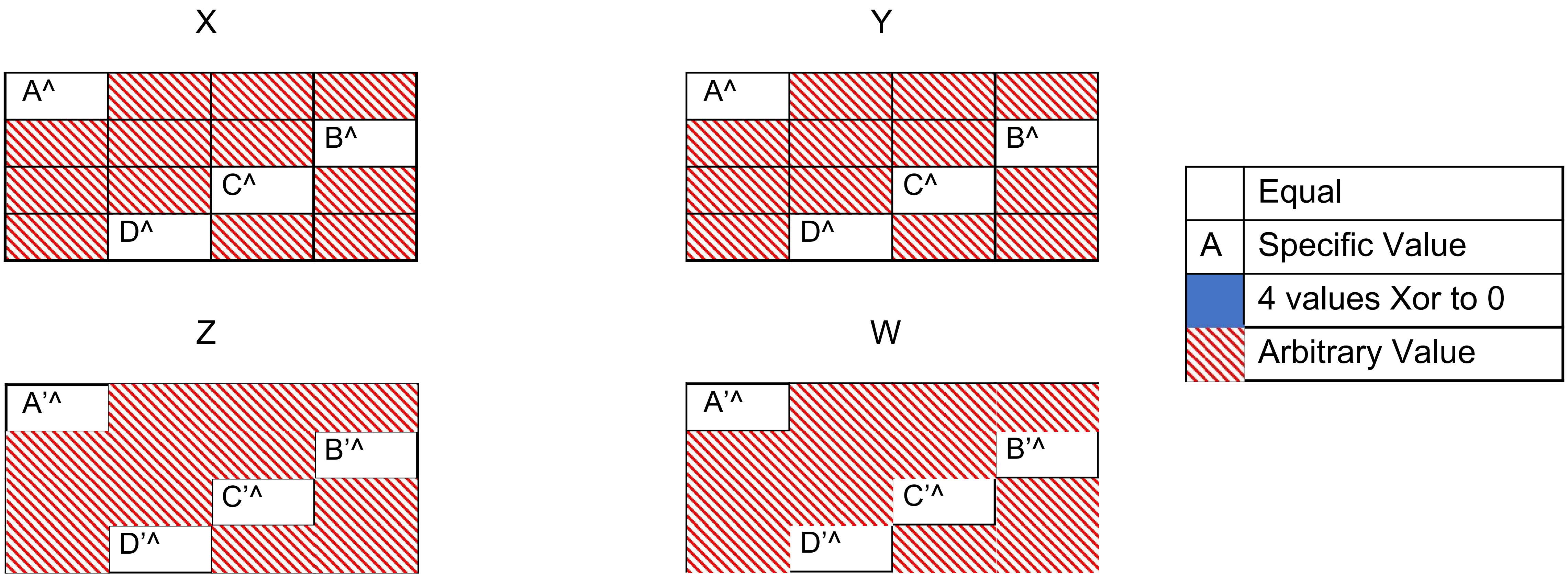
Extending this property to 4 rounds

- Round i+3 after sub byte



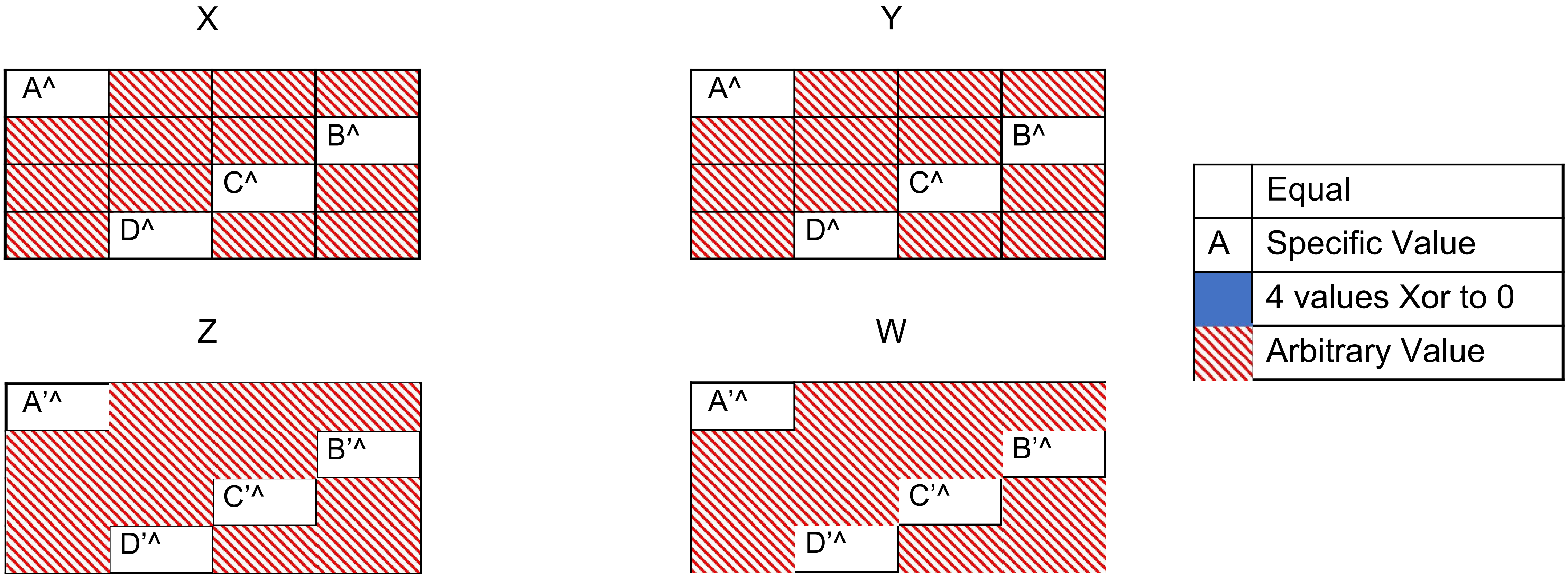
Extending this property to 4 rounds

- Round i+3 after Shift Rows and before Mix Column



AES 4 Round Distinguisher

- Last round of AES has no Mix Column



A 5 Round AES Attack (Grassi 18)

- Precede the 4 round distinguisher with an extra round before it
- We encrypt all possible values of A,B,C,D

A			
	B		
		C	
			D

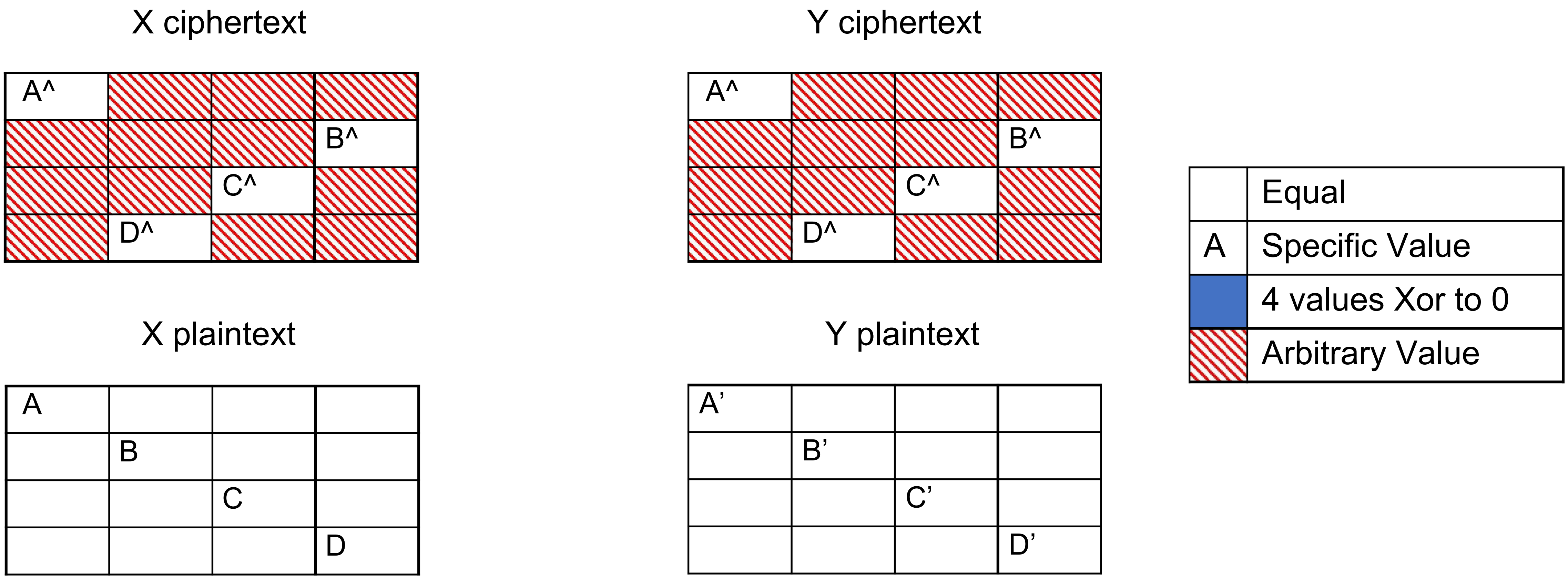
- Then as input to round 1 we get:

A’			
B’			
C’			
D’			

A’, B’, C’, and D’ is a permutation of A, B, C, D which depends only on 4 key bytes

A 5 Round AES Attack [Grassi 18]

- We look for a “good ciphertext pair”, and get the plaintext



A 5 Round AES Attack [Grassi 18]

- For all 2^{32} possible key bytes: partially encrypt (AKR, SB, SR, MC)

X partial round encryption

A*			
B*			
C*			
D*			

Y partial round encryption

A'*			
B'*			
C'*			
D'*			

X plaintext

A			
	B		
		C	
			D

Y plaintext

A'			
	B'		
		C'	
			D'

A 5 Round AES Attack [Grassi 18]

- Create a state mixture Z, W

X partial round encryption

A*			
B*			
C*			
D*			

Y partial round encryption

A'*			
B'*			
C'*			
D'*			

Z partial round encryption

A*			
B'*			
C*			
D'*			

W partial round encryption

A'*			
B*			
C'*			
D*			

A 5 Round AES Attack [Grassi 18]

- Partially decrypt Z and W

Z plaintext

A°			
	B°		
		C°	
			D°

W plaintext

A°’			
	B°’		
		C°’	
			D°’

Z partial round encryption

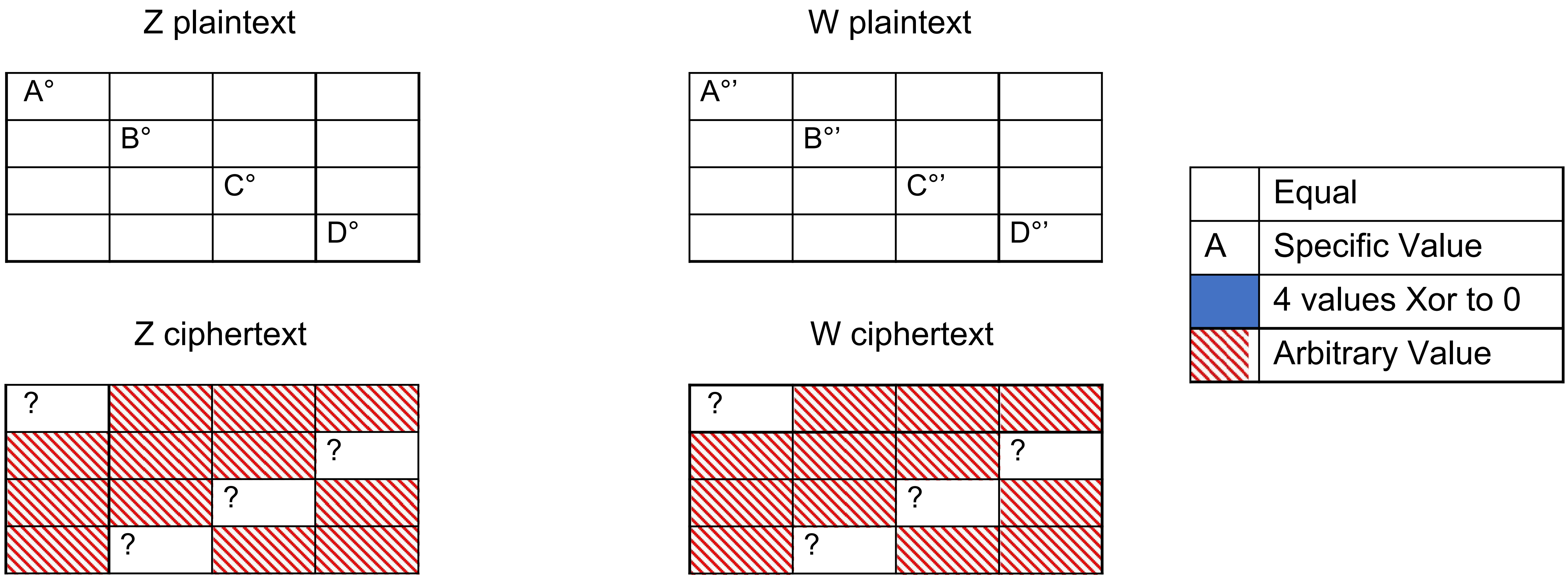
A*			
B’*			
C*			
D’*			

W partial round encryption

A’*			
B*			
C’*			
D*			

A 5 Round AES Attack [Grassi 18]

- Get Z and W ciphertexts, and check the equality condition



Improvements by Shamir et al

Attack	Complexity
Grassi's original attack	$T=2^{32}$, $D=2^{32}$, $M=2^{32}$
Reduce data to get one "good mixture"	$T=2^{47}$, $D=2^{24}$, $M=2^{24}$
Switch order to iterate over pairs	$T=2^{33}$, $D=2^{24}$, $M=2^{24}$
Use precomputed table	$T=2^{29}$, $D=2^{24}$, $M=2^{24}$
Smart selection of input structure	$T=2^{22}$, $D=2^{22}$, $M=2^{22}$

Idea 1 - Reduce Data:

- There are many mixtures, but we only need one of them
- Grassi used 2^{32} data
  - 2^{32} encryptions -> 2^{63} pairs -> 2^{31} good pairs
- We use only 2^{24} data
  - 2^{24} encryptions -> 2^{47} pairs -> 2^{15} good pairs
  - For each key and mixture type:
    We have the mixture in our data with probability $(2^{24}/2^{32})^2 = 2^{-16}$
  - There are 2^{15} pairs and 7 mixture types:
    We have a good mixture with probability $1-(1-2^{-16})^{(7*2\^{}15)} \sim 0.97$

Idea 1 - Reduce Data:

- We can thus reduce the data complexity

- However, we need to go over all 2^{15} pairs
  - So now $T = 2^{32} * 2^{15} = 2^{47}$

- This is only a time \ data tradeoff:
  - We reduce the data by a factor of 2^{8}
  - While increasing the time by a factor of 2^{15}

Idea 2 – Switch Order:

- We can change the order of operations, iterating over all pairs of pairs:
- If we have a mixture after ARK, SB , SR and MC operations:

$$X_0'' \oplus Y_0'' \oplus Z_0'' \oplus W_0'' = 0$$

- Holds for each byte separately, depending on a single key byte

$$SB(X_{0,0} \oplus k_0) \oplus SB(Y_{0,0} \oplus k_0) \oplus SB(Z_{0,0} \oplus k_0) \oplus SB(W_{0,0} \oplus k_0) = 0$$

- Can find a suggestion for each of the 4 key bytes independently
- Take the 4 key bytes and check for mixture after 1 round

Idea 2 – Switch Order:

- For each pair of pairs (quartet) we can get a 4 key bytes suggestion with $4*2^8$ S-Box applications
  - 2^{24} encryptions -> 2^{47} pairs -> 2^{15} “good pairs”
  - 2^{29} quartets * 4 * 2^8 S box= 2^{39} S-Box ~ 2^{33} encryptions

Idea 3 - Precomputed Table

- We can use an optimized precomputed table
- Consider quartet of bytes of the form (0, a, b, c)
  - For each quartet we find a k such as:
    $SB(k) \oplus SB(a \oplus k) \oplus SB(b \oplus k) \oplus SB(c \oplus k) = 0$
  - We get (0, a, b, c) by $(0, \mathrm{y} \oplus x, \mathrm{z} \oplus x, \mathrm{w} \oplus x)$
- We get a table of size 2^{24}
  - The order is irrelevant so we can arrange in increasing order: save a factor of 6 to get ~ $2^{(21.4)}$
  - Precomputation can be optimize to use ~ 2^{24} S Box applications

Idea 4 – Smart Input Structure

- So far we get data and memory 2^{24} and time 2^{29}
- If we select $2^{21.5}$ data from 2^{24} data arbitrarily then there is non negligible probability of existence of at least one good mixture.

const			
	A		
		B	
			C

- The probability is almost 51%

Idea 4 – Smart Input Structure

- $2^{21.5}$ encryptions -> 2^{42} pairs -> 2^{10} good pairs
- Number of quartets $(2^{10})^2/2 = 2^{19}$
- Then use precomputed table from improvement 3

Observation 5

- Data \ Memory trade off
- We can check for zero diff also in SR(Col(1)) and SR(Col(2)) …
- We can check 4 diagonals
  - Increase probability of success by 4
  - Amount of quartets = date^4
  - Reduces the data only by 4^(1/4) = sqrt(2)
  - Increases the amount of memory by factor of 4

References

- https://eprint.iacr.org/2018/527.pdf
- https://crypto.iacr.org/2018/slides/28838.pdf
- https://link.springer.com/content/pdf/10.1007%2F978-3-319-56614-6_10.pdf
- https://eprint.iacr.org/2017/832.pdf
- https://pdfs.semanticscholar.org/7405/c463a0d8477396c3a60408fb3ead0917bfb4.pdf
- https://gist.github.com/bonsaiviking/5571001