Table Of Contents

- 1. Introduction to AES
- 2. Design Principles behind AES
  - Linear Cryptanalysis
  - Differential Cryptanalysis
  - Square Attack
  - Biclique Attack
- 3. Quantum Cryptanalysis of AES
  - Applying Grover’s Algorithm – Circuit Cost Estimates
  - Quantum Differential and Linear Cryptanalysis
  - Breaking Symmetric Cryptosystems using Quantum Period Finding
- 4. References
- 5. Questions

1.1 Introduction to AES

- Ref: “The Design of Rijndael” (Joan Daemen and Vincent Rijmen) – Springer 2002
- Rijndael announced as the winner of the AES competition in October 2000
  - Designed by Rijmen and Daemen
- AES is a symmetric cipher
  - Blocksize is 128 bits viewed as a 4 by 4 matrix of bytes
  - Keys are 128, 192 or 256 bits
- In this talk we concentrate on AES-128 with 128 bit key
- AES-128 has 10 rounds
- AES-192 has 12 rounds
- AES-256 has 14 rounds
- The 128, 192 or 256 bit key is expanded to provide enough key bits to encrypt each round

1.2 AES-128 Round Structure

- AES-128 works on a State that is 4 by 4 matrix of 8 bit bytes
- AES(State, CipherKey)

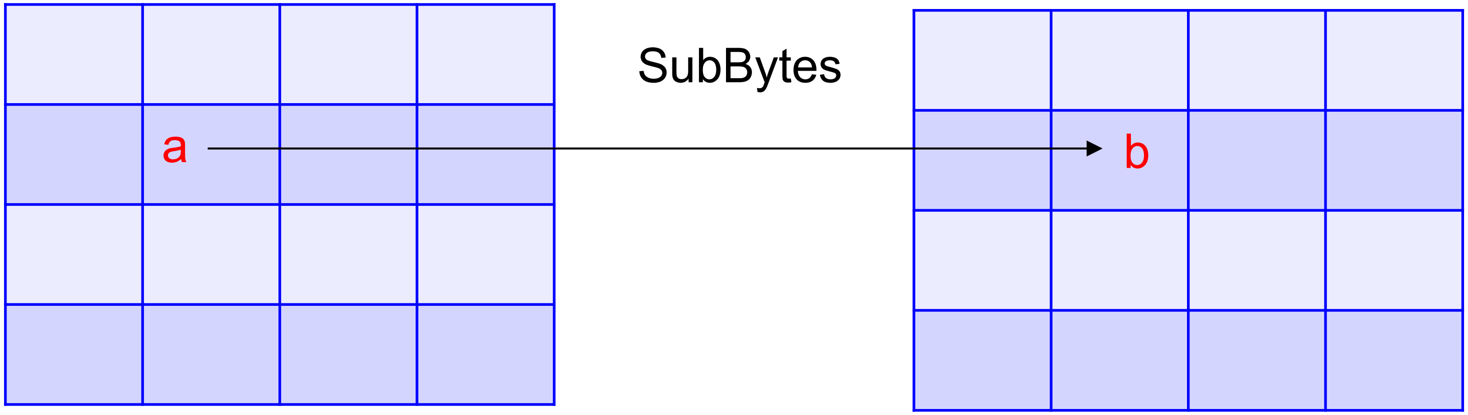
```
{
  KeyExpansion(CipherKey, ExpandedKey)
  AddRoundKey(State, ExpandedKey[0])
  for i=1:9
       Round(State,ExpandedKey[i])
   end
   FinalRound(State, ExpandedKey[10])
 }
```

1.3 AES-128 Round Structure

```
Round(State, ExpandedKey[i])
{
Subbytes(State)
ShiftRows(State)
MixColumns(State)
AddRoundey(State, ExpandedKey[i]
}

FinalRound(State, ExpandedKey[i])
{
SubBytes(State)
ShiftRows(State)
AddRoundKey(State, ExpandedKey[i])
}
```

1.4 AES-128 Round Structure - SubBytes



SubBytes

a

b

- SubBytes: $a \rightarrow f(a^{-1})$=b

  find inverse of a in $GF(2^{8})$ followed by an affine transformation f

  An affine transformation is a linear mixing and shift of the bits of a^{-1}

- Note 1: SubBytes is the only non-linear part of AES and operates on each byte individually

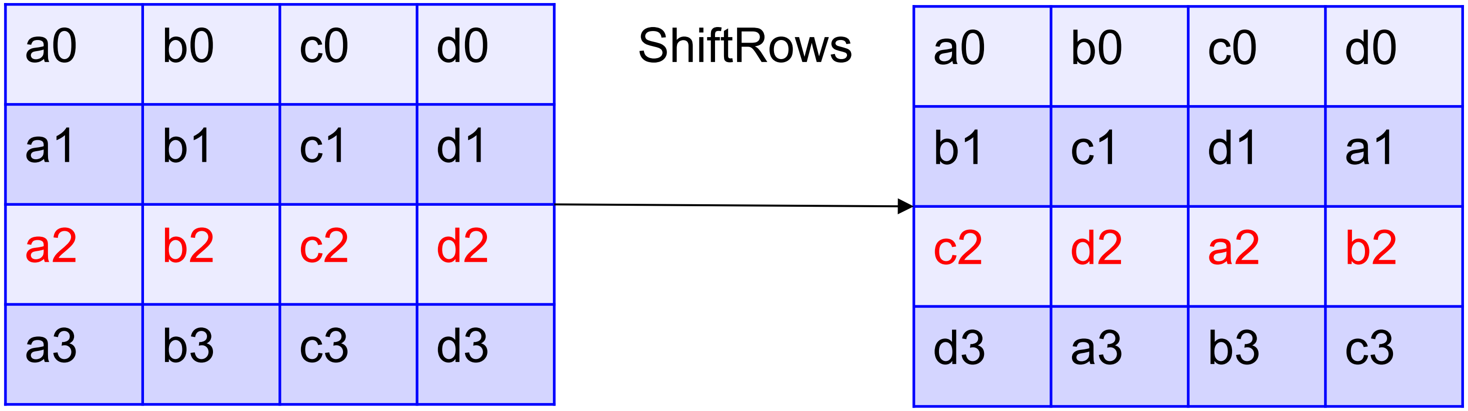
- Note 2: XOR is a linear function in $GF(2)\ and\ GF(2^{8})$ which is defined by the irreducible polynomial $m(x)=x^{8}+x^{4}+x^{3}+x+1$

1.5 AES-128 Round Structure – SubBytes (Contd)

- SubBytes Affine Transformation

- [■■■1&1@0&1 &■1&1@1&1 @■0&0@0&0 &■1&1@0&1 &■■1&0@1&1 &■0&0@0&0
  @■1&1@1&1 &■1&0@1&1 @■■1&0@1&1 &■0&0@0&0 @■1&1@1&1 &■1&0@1&1
  &■■1&1@0&1 &■1&1@1&1 @■0&0@0&0 &■1&1@0&1][■■■a7@a6 @■a5@a4
  @■■a3@a2 @■a1@a0]+[■■■0@1 @■1@0 @■■0@0 @■1@1]=[■■■b7@b6
  @■b5@b4 @■■b3@b2 @■b1@b0]

1.6 AES-128 Round Structure - ShiftRows



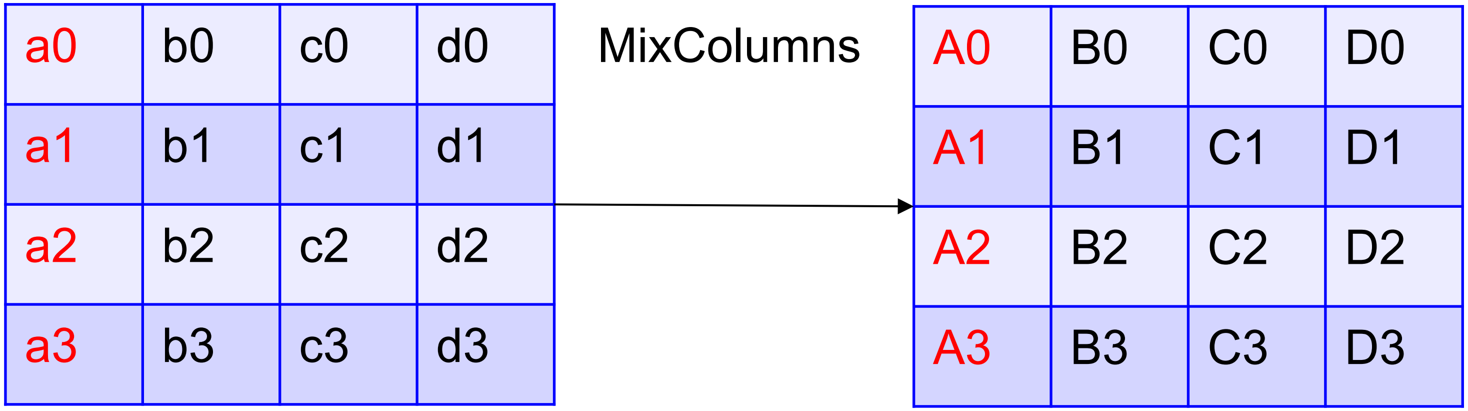
a0	b0	c0	d0
a1	b1	c1	d1
a2	b2	c2	d2
a3	b3	c3	d3

ShiftRows

a0	b0	c0	d0
b1	c1	d1	a1
c2	d2	a2	b2
d3	a3	b3	c3

Row 0 is not shifted
Row 1 is circularly left shifted by 1
Row 2 is circularly left shifted by 2
Row 3 is circularly left shifted by 3
Note: ShiftRows preserves the XOR of the row bytes and the total XOR of all the bytes

1.7 AES-128 Round Structure - MixColumns



a0	b0	c0	d0
a1	b1	c1	d1
a2	b2	c2	d2
a3	b3	c3	d3

MixColumns

A0	B0	C0	D0
A1	B1	C1	D1
A2	B2	C2	D2
A3	B3	C3	D3

(■■2&3@1&2 &■1&1@3&1 @■1&1@3&1

&■2&3@1&2)(■■a0@a1 @■a2@a3)=(■■A0@A1

@■A2@A3)

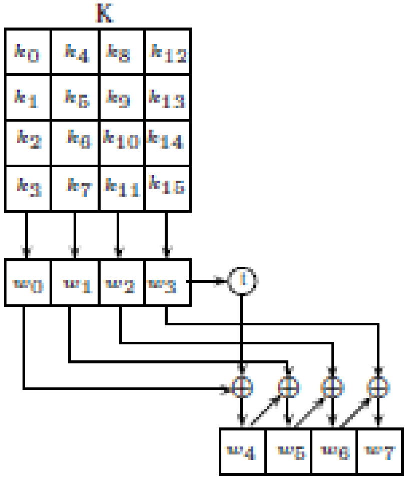
Addition is performed over $GF(2)$

Multiplication over $GF(2\uparrow 8)$

Note: MixColumns preserves the XOR of the columns and the total XOR of all the bytes

1.8 Key Expansion

- Ref: “Analysis of Block Cipher Constructions against Biclique and Multiset Attacks”, PhD Thesis by Mohona Ghosh, 2016
- Key Expansion of AES-128 (example)
- Begin with 128 bit key prepared as 4x4 byte state array
- Form 4 32-bit words from the columns. Iterate key schedule to obtain enough key for each of the rounds



2.1 Design Principles of AES – Linear Cryptanalysis

- **Linear Cryptanalysis: Correlations and Linear Trails**
- The correlation between two Boolean functions f and g is
- $C(f,g)=2.Prob(f(a)=g(a))-1$
- $-1\leq C(f,g)\leq 1$
- A parity function is the XOR of a number of bits
- Given a non-linear function S we can calculate the correlation of a parity function to it
- SubBytes() has been chosen such that the maximum correlation of a parity function is $2\uparrow -3$
- Given a sequence of rounds we can identify a corresponding linear trail through it which is a sequence of parity functions. We multiply the corresponding correlations of the parity functions to get the correlation of the linear trail. For AES, this gives the maximum of $2\uparrow -75$ for the correlation for any four round linear trail.

2.2 Design Principles of AES – Differential Cryptanalysis

- **Differential Cryptanalysis**: Consider two n bit vectors a and a^{*} where $a+a^{*}=a'$ a fixed difference pattern.

- Let $b=h(a), b^{*}=h(a^{*})$ and $b+b^{*}=b'$ the difference a' propagates to the difference b'.

- The difference propagation probability is

  $Prob(a',b')=2^{-n}\sum_{a}\delta(b'+h(a+a')+h(a))$

- The weight of a difference propagation is $w(a',b')=-\log_{2}(Prob(a',b'))$

- A differential trail is a sequence of difference patterns:

  D=$(d^{0},d^{1},d^{2},\ldots d^{r-1},d^{r})$

  The weight of a differential trail is the sum of the weights its differential steps

  $w(D)=w(d^{0},d^{1})+w(d^{1},d^{2})+\ldots w(d^{r-1},d^{r})$

  SubBytes has a differential weight of at least 6 meaning a differential propagation probability of at least 2^{-6}. This gives a minimum weight of 150 for any four round differential trail.

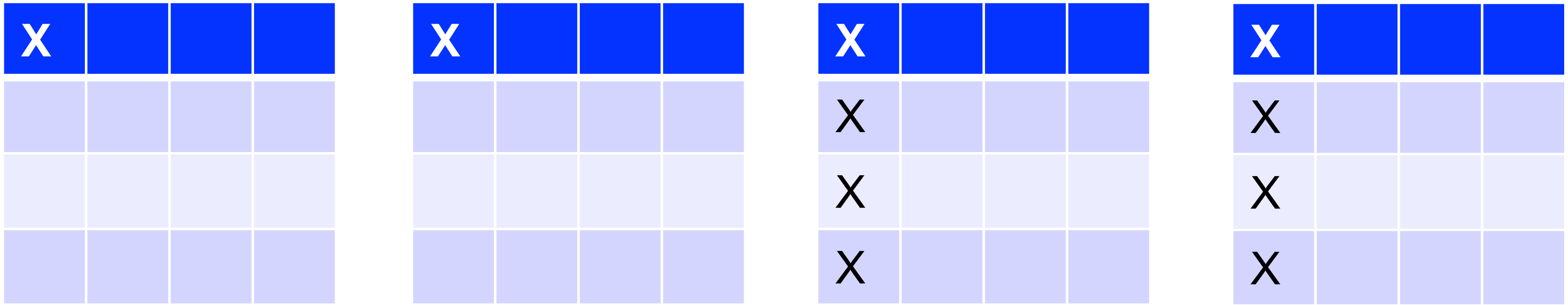
2.3 Design Principles of AES

- **Results**
- 1. There are no 8 round correlations above $2\uparrow-80$
- 2. There are no 8 round differential trails with a weight below 300
- “We consider this sufficient to resist differential and linear attacks” (designers of Rijndael)

2.4 AES - Square Attack – 3 Round Property

- Square Attack (aka Saturation Attack, Integral Attack, Partial Sums) – Chosen Plaintext attack originally identified by the designers of AES and improved upon since then (Ferguson et al 2000). We first describe a 3-round property. (Ref Gosh PhD Thesis)

- We input 2↑8 plaintexts which are all 0 except for the first byte which varies over all 2↑8 values. We track the XOR of the bytes at each byte position through 3 rounds of AES. The XOR of the bytes is 0 at every byte position through 3 rounds. X means byte position is active.



SubBytes

X			

ShiftRows

X			

MixColumns

X			
X			
X			
X			

AddRoundKey

X			
X			
X			
X			

Round 1
Input 2↑8 plaintexts
First step is AddRoundKey (not shown)

2.5 AES - Square Attack – 3 Round Property

X			
X			
X			
X			

SubBytes

X			
			X
		X	
	X		

ShiftRows

X	X	X	X
X	X	X	X
X	X	X	X
X	X	X	X

MixColumns

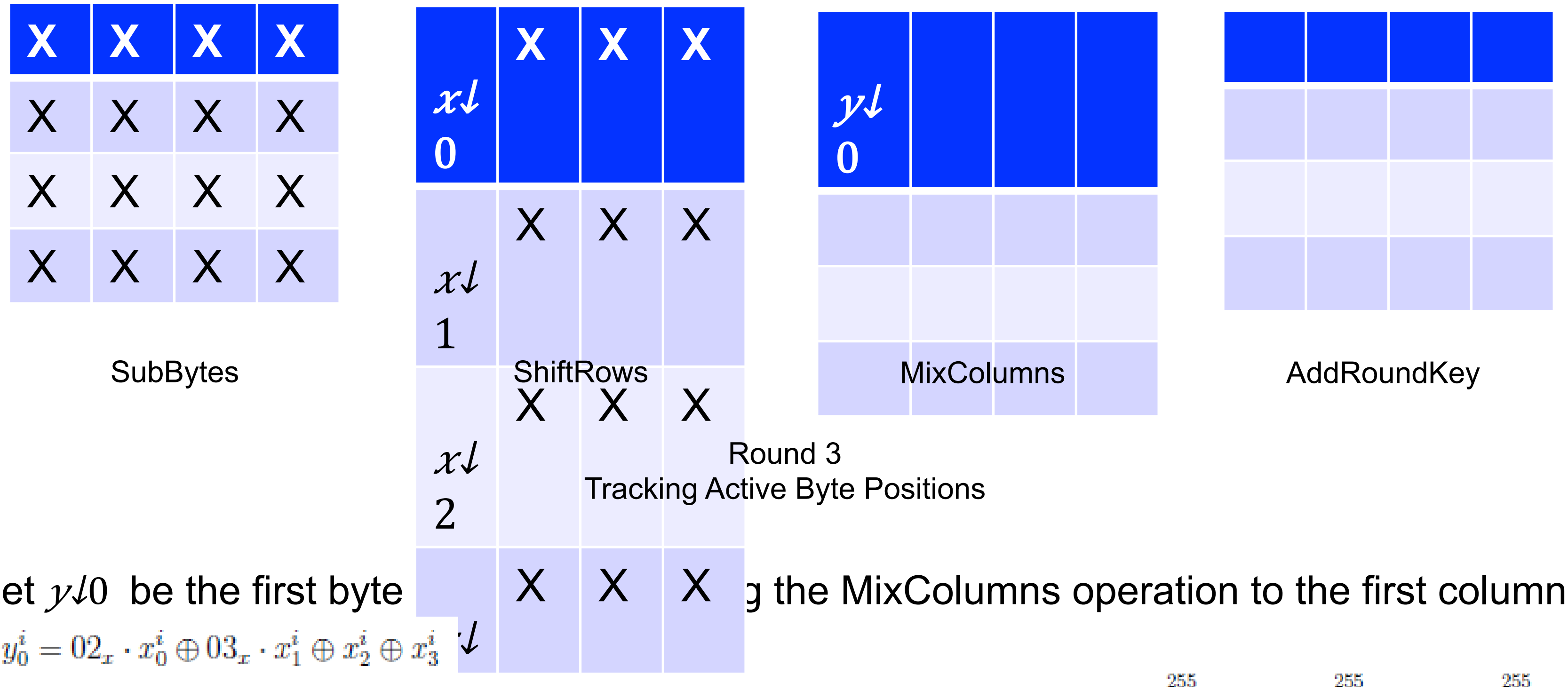
X	X	X	X
X	X	X	X
X	X	X	X
X	X	X	X

AddRoundKey

Round 2
Tracking Active Byte Positions

- Turns out the property of XOR byte positions = 0 goes through to 3 rounds by analyzing the properties of AES's third round application of the linear function MixColumns (next slide)

2.5 AES - Square Attack – 3 Round Property



SubBytes

ShiftRows

MixColumns

AddRoundKey

Round 3
Tracking Active Byte Positions

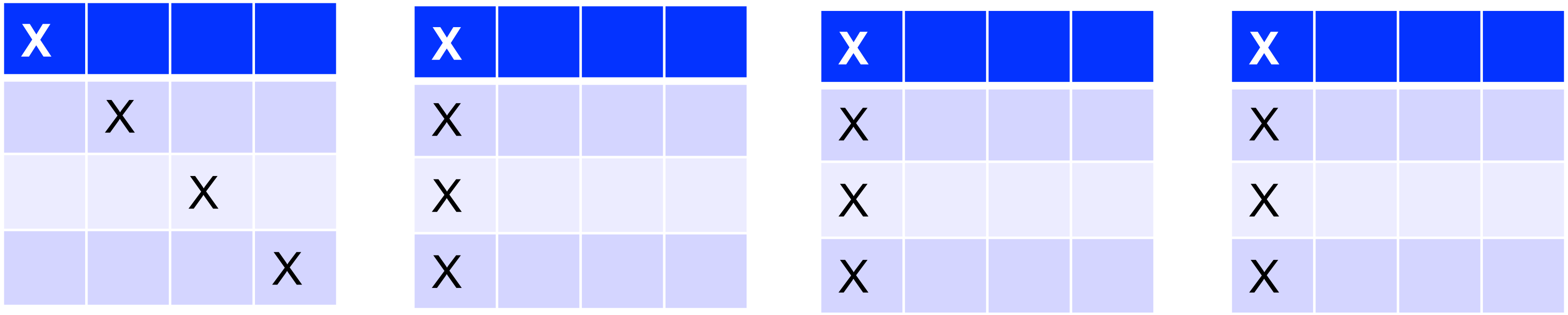
Let $y↓0$ be the first byte g the MixColumns operation to the first column

$$y_0^i = 02_x \cdot x_0^i \oplus 03_x \cdot x_1^i \oplus x_2^i \oplus x_3^i$$

$$\begin{aligned} y_0^0 \oplus y_0^1 \oplus \ldots y_0^{255} \; = \; & 02_x \cdot x_0^0 \oplus 03_x \cdot x_1^0 \oplus x_2^0 \oplus x_3^0 \oplus \\ & 02_x \cdot x_0^1 \oplus 03_x \cdot x_1^1 \oplus x_2^1 \oplus x_3^1 \oplus \\ & \vdots \\ & 02_x \cdot x_0^{255} \oplus 03_x \cdot x_1^{255} \oplus x_2^{255} \oplus x_3^{255} \end{aligned}$$

$$\begin{aligned} y_0^0 \oplus y_0^1 \oplus \ldots y_0^{255} &= 02_x \cdot \bigoplus_{i=0}^{255} x_0^i \oplus 03_x \cdot \bigoplus_{i=0}^{255} x_1^i \oplus 01_x \cdot \bigoplus_{i=0}^{255} x_2^i \oplus 01_x \cdot \bigoplus_{i=0}^{255} x_3^i \\ &= 02_x \cdot 00_x \oplus 03_x \cdot 00_x \oplus 00_x \oplus 00_x \\ &= 00_x \end{aligned}$$

2.6 AES - Square Attack – 4 Round Property



X			
	X		
		X	
			X

SubBytes

X			
X			
X			
X			

ShiftRows

X			
X			
X			
X			

MixColumns

X			
X			
X			
X			

AddRoundKey

4-Round Property
Add New First Round 2↑32 Plaintexts at Beginning of 3-Round Property

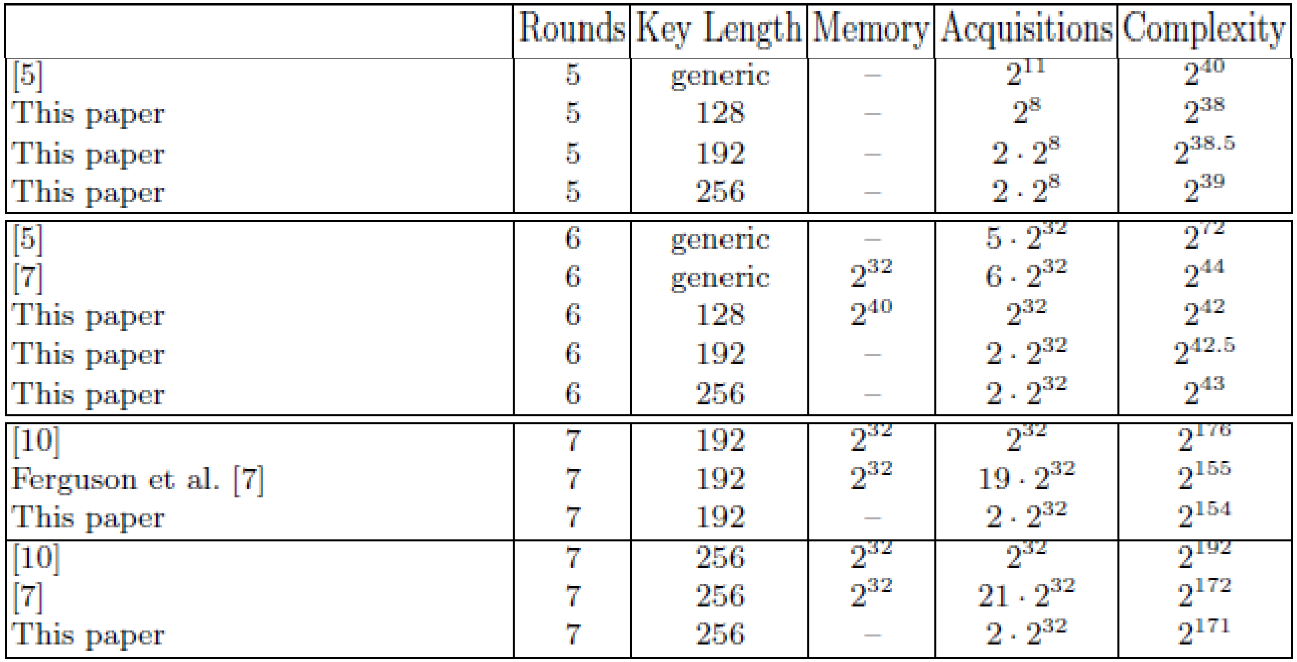
- We can extend this to a 4-round property by using 2↑32 plaintexts where the 4 active bytes vary over all possible 2↑32 byte combinations

- At the end of 4-rounds the XOR of the bytes at each byte position is still 0

- We can add a key assumption at rounds 5 and 6 to get a 6 round attack – work factor 2↑44 an improvement due to Ferguson et. al. 2001 from the original 2↑72

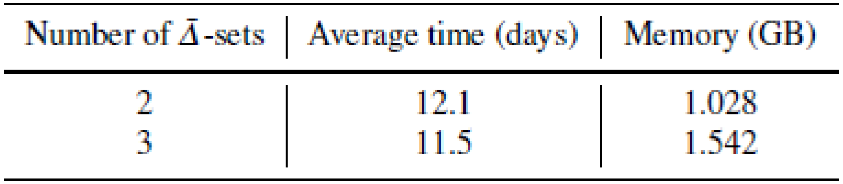
2.7 AES - Square Attack

- Ref: “Improved “Partial Sums” – based Square Attack on AES”, Tunstall (2012)



	Rounds	Key Length	Memory	Acquisitions	Complexity
[5]	5	generic	–	2^{11}	2^{40}
This paper	5	128	–	2^{8}	2^{38}
This paper	5	192	–	$2 \cdot 2^{8}$	$2^{38.5}$
This paper	5	256	–	$2 \cdot 2^{8}$	2^{39}
[5]	6	generic	–	$5 \cdot 2^{32}$	2^{72}
[7]	6	generic	2^{32}	$6 \cdot 2^{32}$	2^{44}
This paper	6	128	2^{40}	2^{32}	2^{42}
This paper	6	192	–	$2 \cdot 2^{32}$	$2^{42.5}$
This paper	6	256	–	$2 \cdot 2^{32}$	2^{43}
[10]	7	192	2^{32}	2^{32}	2^{176}
Ferguson et al. [7]	7	192	2^{32}	$19 \cdot 2^{32}$	2^{155}
This paper	7	192	–	$2 \cdot 2^{32}$	2^{154}
[10]	7	256	2^{32}	2^{32}	2^{192}
[7]	7	256	2^{32}	$21 \cdot 2^{32}$	2^{172}
This paper	7	256	–	$2 \cdot 2^{32}$	2^{171}

Ref: “Implementation and Improvement of the Partial Sum Attack on 6-round AES”, Alda, Aragona, Nicolodi, Sala (2014)

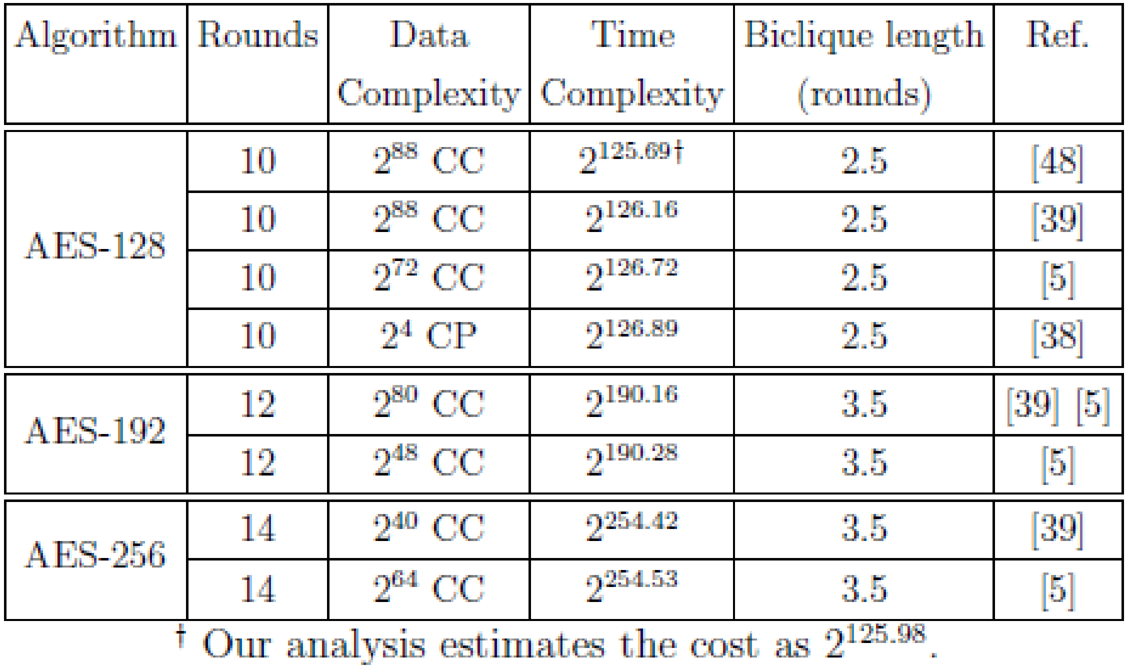


Number of $\bar{\Lambda}$-sets	Average time (days)	Memory (GB)
2	12.1	1.028
3	11.5	1.542

2.8 Full AES - Biclique Attack

- Ref: Analysis of Block Cipher Constructions against Biclique and Multiset Attacks, PhD Thesis by Mohona Ghosh, 2016

- Summary: Key recovery with bicliques for full AES. CC=Chosen Ciphertext, CP=Chosen Plaintext



Algorithm	Rounds	Data Complexity	Time Complexity	Biclique length (rounds)	Ref.
AES-128	10	2^{88} CC	$2^{125.69\dagger}$	2.5	[48]
	10	2^{88} CC	$2^{126.16}$	2.5	[39]
	10	2^{72} CC	$2^{126.72}$	2.5	[5]
	10	2^{4} CP	$2^{126.89}$	2.5	[38]
AES-192	12	2^{80} CC	$2^{190.16}$	3.5	[39] [5]
	12	2^{48} CC	$2^{190.28}$	3.5	[5]
AES-256	14	2^{40} CC	$2^{254.42}$	3.5	[39]
	14	2^{64} CC	$2^{254.53}$	3.5	[5]

$\dagger$ Our analysis estimates the cost as $2^{125.98}$.

3.1 Quantum Cryptanalysis of AES

- Ref: “Applying Grover’s algorithm to AES: quantum resource estimates”, Grassl, Langenberg, Roetteler, Steinwandt Quant-ph 1512.04965 15 December, 2015

- Problem: Given AES-128 and 3 plaintext/ciphertext pairs, find the secret key K

- Solution (roughly):
  - Step 1: $|\psi\downharpoonleft 1 > = 1/\sqrt{2}\upharpoonleft 128 \;\; \sum\upharpoonleft ▒ |K>|0>$ uniform superposition of all $2\upharpoonleft 128$ keys
  - Step 2: $|\psi\downharpoonleft 2 > = 1/\sqrt{2}\upharpoonleft 128 \;\; \sum\upharpoonleft ▒ |K>|AES\downharpoonleft k\,(m)>$ computation of AES with key K to plaintext m
  - Step 3: $|\psi\downharpoonleft 2 > = 1/\sqrt{2}\upharpoonleft 128 \;\; \sum\upharpoonleft ▒ |K>|AES\downharpoonleft k\,(m){=}c\downharpoonleft m >$ test equality with known ciphertext $c\downharpoonleft m$
  - Step 4: apply Grover’s algorithm to find which key K gives the equality value 1 versus the inequality value 0

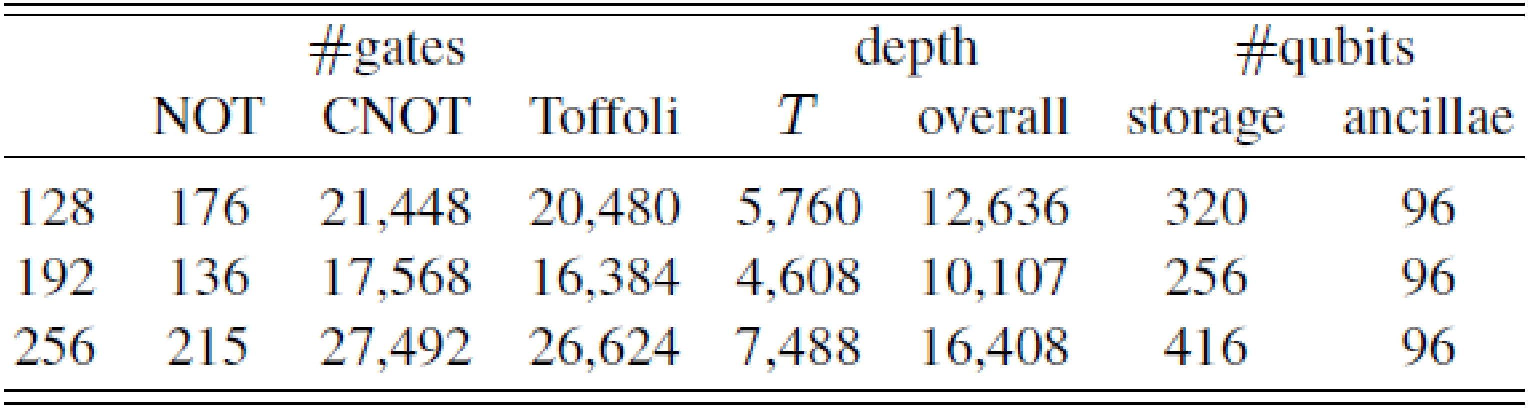
- Main costs: Computation of AES, Computation of Grover’s algorithm

3.2 Quantum Cryptanalysis of AES (Contd)

- Ref: “Applying Grover’s algorithm to AES: quantum resource estimates”, Grassl, Langenberg, Roetteler, Steinwandt Quant-ph 1512.04965 15 December, 2015

- Grover’s Cost (well known):
  - To find 1 item in an N long list costs $O(\sqrt{N})$. Here $N = 2 \uparrow 128$, $2 \uparrow 192$, $2 \uparrow 256$

- AES-k cost (Note quantum circuits are fully reversible)

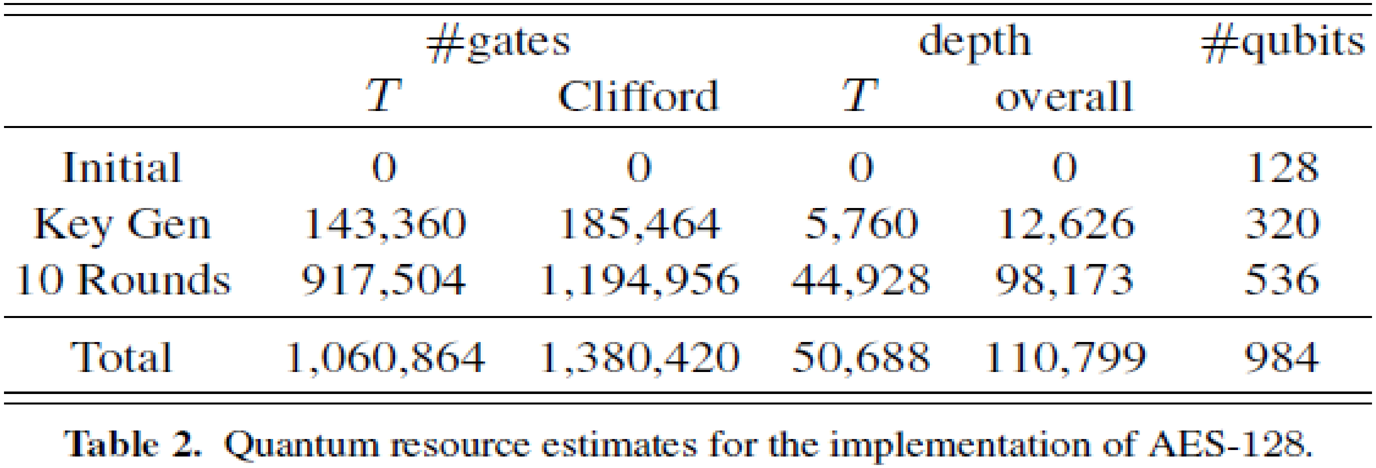


	#gates			depth		#qubits	
	NOT	CNOT	Toffoli	T	overall	storage	ancillae
128	176	21,448	20,480	5,760	12,636	320	96
192	136	17,568	16,384	4,608	10,107	256	96
256	215	27,492	26,624	7,488	16,408	416	96

Table 1. Quantum resource estimates for the key expansion phase of AES-k, where $k \in \{128, 192, 256\}$.

3.3 Quantum Cryptanalysis of AES (Contd)

- Ref: “Applying Grover’s algorithm to AES: quantum resource estimates”, Grassl, Langenberg, Roetteler, Steinwandt Quant-ph 1512.04965 15 December, 2015

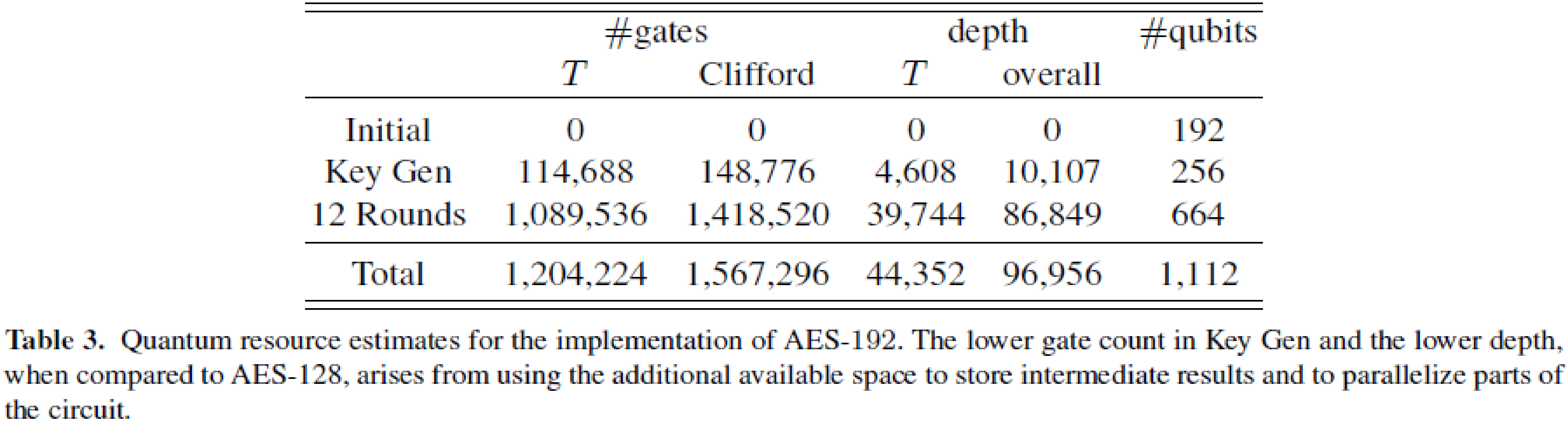


	#gates		depth		#qubits
	T	Clifford	T	overall	
Initial	0	0	0	0	128
Key Gen	143,360	185,464	5,760	12,626	320
10 Rounds	917,504	1,194,956	44,928	98,173	536
Total	1,060,864	1,380,420	50,688	110,799	984

Table 2. Quantum resource estimates for the implementation of AES-128.

3.4 Quantum Cryptanalysis of AES (Contd)

- Ref: “Applying Grover’s algorithm to AES: quantum resource estimates”, Grassl, Langenberg, Roetteler, Steinwandt Quant-ph 1512.04965 15 December, 2015
-



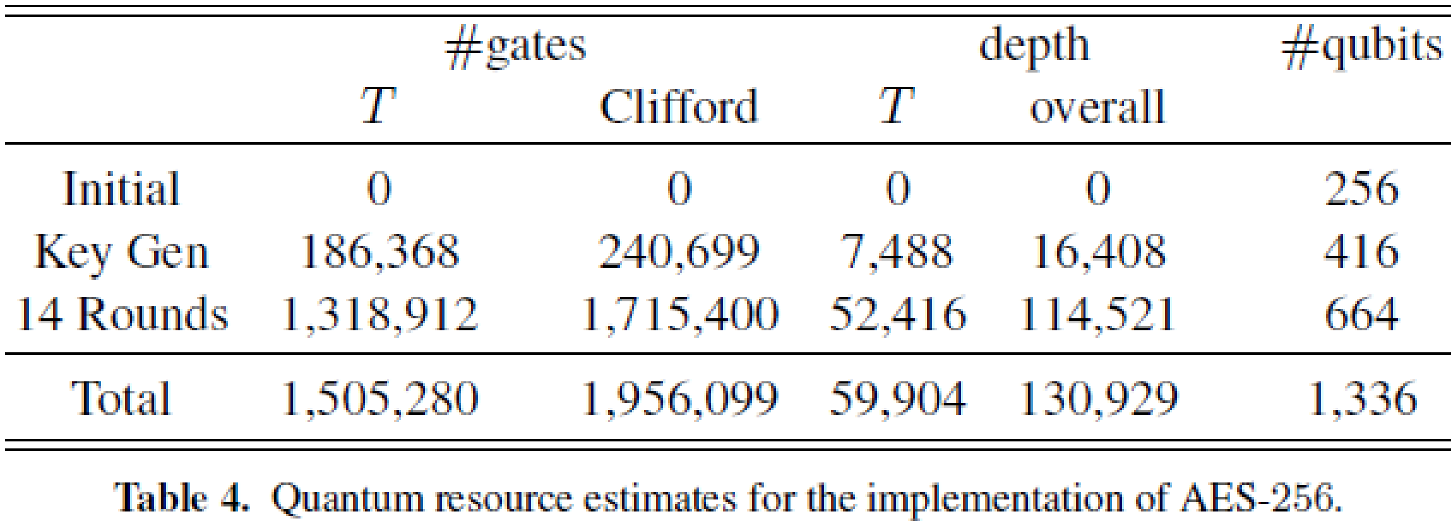
	#gates		depth		#qubits
	T	Clifford	T	overall	
Initial	0	0	0	0	192
Key Gen	114,688	148,776	4,608	10,107	256
12 Rounds	1,089,536	1,418,520	39,744	86,849	664
Total	1,204,224	1,567,296	44,352	96,956	1,112

Table 3. Quantum resource estimates for the implementation of AES-192. The lower gate count in Key Gen and the lower depth, when compared to AES-128, arises from using the additional available space to store intermediate results and to parallelize parts of the circuit.

3.5 Quantum Cryptanalysis of AES (Contd)

- Ref: “Applying Grover’s algorithm to AES: quantum resource estimates”, Grassl, Langenberg, Roetteler, Steinwandt Quant-ph 1512.04965 15 December, 2015

- :



	#gates		depth		#qubits
	T	Clifford	T	overall	
Initial	0	0	0	0	256
Key Gen	186,368	240,699	7,488	16,408	416
14 Rounds	1,318,912	1,715,400	52,416	114,521	664
Total	1,505,280	1,956,099	59,904	130,929	1,336

Table 4. Quantum resource estimates for the implementation of AES-256.

3.6 Quantum Differential and Linear Cryptanalysis

- Ref: “Quantum Differential and Linear Cryptanalysis”, Kaplan, Leurent, Leverrier, Naya-Plasencia (2017)
- The authors examine differential and linear cryptanalysis in two settings defined as follows

  (PRP = Pseudo Random Permutation

  PRF = Pseudo Random Function)
- **Standard security**: a block cipher is *standard secure* against quantum adversaries if no efficient quantum algorithm can distinguish the block cipher from PRP (or a PRF) by making only *classical* queries (Q1) (i.e. can make classical encryption queries)
- **Quantum security**: a block cipher is *quantum secure* against quantum adversaries if no efficient quantum algorithm can distinguish the block cipher from PRP (or a PRF) even by making *quantum* queries (Q2) (i.e. can make quantum encryption queries)

3.7 Quantum Differential and Linear Cryptanalysis

- Ref: “Quantum Differential and Linear Cryptanalysis”, Kaplan, Leurent, Leverrier, Naya-Plasencia (2017)

- Results:

- Differential cryptanalysis and linear cryptanalysis usually offer a quadratic gain in the Q2 model over the classical model.
  - If a block cipher is resistant to a classical linear or differential cryptanalysis attack costing at least $2 \uparrow k$ then it is also resistant the corresponding quantum linear or differential cryptanalysis attacks cost at least $2 \uparrow k/2$

- In the Q1 model cryptanalytic attacks might offer little gain over the classical model when the key-length is the same as the block length (e.g. AES-128)

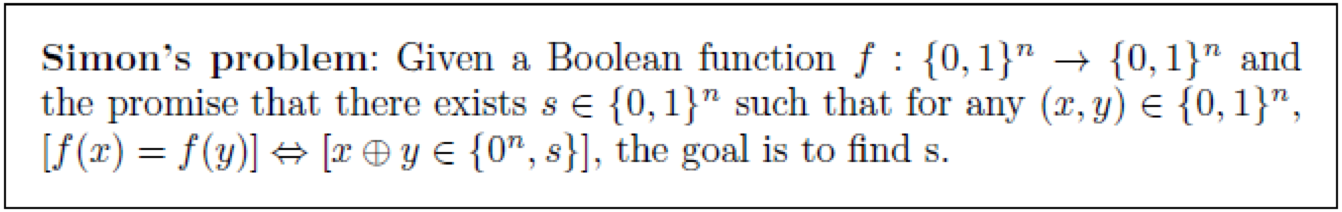
- The gain of cryptanalytic attacks in the Q1 model can be quite significant (similar to the Q2 model) when the key length is longer (e.g. AES-256) than the block length

3.8 Breaking Symmetric Cryptosystems using Quantum Period Finding

- Ref: “Breaking Symmetric Cryptosystems using Quantum Period Finding”, Kaplan, Leurent, Leverrier, Naya-Plasencia (2016)

- Results: CBC-MAC, GMAC, GCM and PMAC, OCB are all broken by forgery attacks using their method

- The author’s take advantage of Simon’s problem and algorithm:



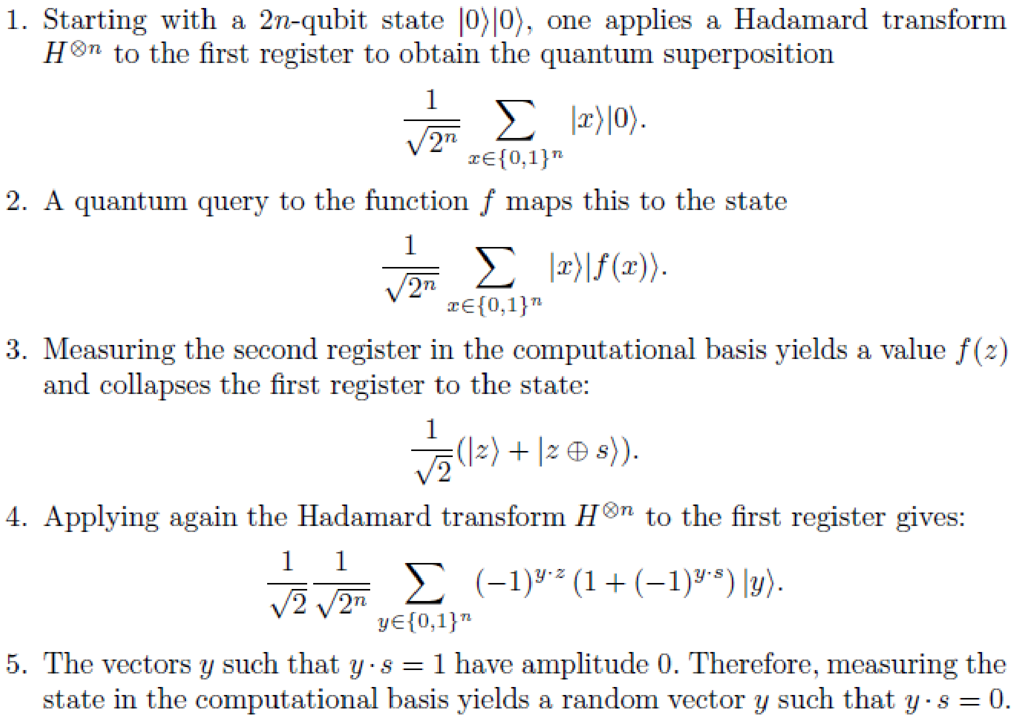
Simon’s problem: Given a Boolean function $f : \{0,1\}^n \to \{0,1\}^n$ and the promise that there exists $s \in \{0,1\}^n$ such that for any $(x,y) \in \{0,1\}^n$, $[f(x) = f(y)] \Leftrightarrow [x \oplus y \in \{0^n, s\}]$, the goal is to find s.

- Note: Simon’s algorithm (next slide) was one of the first to show exponential speedup of a quantum algorithm.

3.9 Breaking Symmetric Cryptosystems using Quantum Period Finding

- Ref: “Breaking Symmetric Cryptosystems using Quantum Period Finding”, Kaplan, Leurent, Leverrier, Naya-Plasencia (2016)

- Simon’s Problem can be solved using Simon’s Algorithm which is repeated O(n) times



1. Starting with a $2n$-qubit state $|0\rangle|0\rangle$, one applies a Hadamard transform $H^{\otimes n}$ to the first register to obtain the quantum superposition
$$\frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |x\rangle|0\rangle.$$
2. A quantum query to the function f maps this to the state
$$\frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |x\rangle|f(x)\rangle.$$
3. Measuring the second register in the computational basis yields a value $f(z)$ and collapses the first register to the state:
$$\frac{1}{\sqrt{2}}(|z\rangle + |z \oplus s\rangle).$$
4. Applying again the Hadamard transform $H^{\otimes n}$ to the first register gives:
$$\frac{1}{\sqrt{2}} \frac{1}{\sqrt{2^n}} \sum_{y \in \{0,1\}^n} (-1)^{y \cdot z} \left(1 + (-1)^{y \cdot s}\right) |y\rangle.$$
5. The vectors y such that $y \cdot s = 1$ have amplitude 0. Therefore, measuring the state in the computational basis yields a random vector y such that $y \cdot s = 0$.

3.10 Breaking Symmetric Cryptosystems using Quantum Period Finding

- Ref: “Breaking Symmetric Cryptosystems using Quantum Period Finding”, Kaplan, Leurent, Leverrier, Naya-Plasencia (2016)

- Example: CBC-MAC

$$x_0 = 0 \qquad x_i = E_k(x_{i-1} \oplus m_i) \qquad \text{CBC-MAC}(M) = E_{k'}(x_\ell)$$

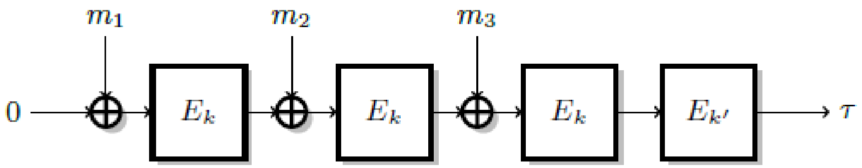


Fig. 9. Encrypt-last-block CBC-MAC.

- Fix two arbitrary message blocks α_0, α_1, with $\alpha_0 \neq \alpha_1$ define the function f:

$$\begin{aligned} f : \{0,1\} \times \{0,1\}^n &\to \{0,1\}^n \\ b, x \quad &\mapsto \text{CBC-MAC}(\alpha_b \,\|\, x) = E_{k'}\left(E_k(x \oplus E_k(\alpha_b))\right) \end{aligned}$$

- Note: Assumption is that we have quantum access to CBC-MAC using *E↓k and E↓k↑′*

3.11 Breaking Symmetric Cryptosystems using Quantum Period Finding

- Ref: “Breaking Symmetric Cryptosystems using Quantum Period Finding”, Kaplan, Leurent, Leverrier, Naya-Plasencia (2016)
- Example: CBC-MAC
- Simon’s algorithm returns “s” which is defined as:

$$1 \,\|\, E_k(\alpha_0) \oplus E_k(\alpha_1)$$

- $CBC-MAC(\alpha_{} \,||m)=E_k^{\prime}\,(E_k\,(E_k\,(\alpha)\oplus m))$
- Let $T_0 = CBC-MAC(\alpha_0 \,||m_1\,)$ for an arbitrary message block m_1
- Let $T_1 = CBC-MAC(\alpha_1 \,||m_1 \oplus E_k\,(\alpha_0\,)\oplus E_k\,(\alpha_1\,))$
- Then $T_0 = T_1$ (i.e. a forgery) since we know $E_k\,(\alpha_0\,)\oplus E_k\,(\alpha_1\,)$

4. References

- “The Design of Rijndael”, Joan Daemen and Vincent Rijmen – Springer 2002
- “Improved Cryptanalysis of Rijndael”, Ferguson, Kelsey, Lucks, Schneier, Stay, Wagner, Whiting, 2000
- “Improved “Partial Sums” – based Square Attack on AES”, Tunstall, 2012
- “Implementation and Improvement of the Partial Sum Attack on 6-round AES”, Alda, Aragona, Nicolodi, Sala, 2014
- “Analysis of Block Cipher Constructions against Biclique and Multiset Attacks”, PhD Thesis by Mohona Ghosh, 2016
- “Applying Grover’s algorithm to AES: quantum resource estimates”, Grassl, Langenberg, Roetteler, Steinwandt Quant-ph 1512.04965 15 December, 2015
- “Quantum Differential and Linear Cryptanalysis”, Kaplan, Leurent, Leverrier, Naya-Plasencia, 2017
- “Breaking Symmetric Cryptosystems using Quantum Period Finding”, Kaplan, Leurent, Leverrier, Naya-Plasencia, 2016
- “Differential cryptanalysis of DES-like cryptosystems”, Biham and Shamir, Springer CRYPTO 1990
- “Linear Cryptanalysis method for DES cipher”, Matsui, Springer EUROCRYPT 1993

5. Questions?

- Contact Information:

  David Cornwell, PhD
  Cornwell_david@bah.com

  (410) 684 6579

  Address:
  Booz Allen Hamilton
  304 Sentinel Drive (NBP)
  Annapolis Junction
  MD 20701
  USA