LEA-128 Quantum implementation

https://youtu.be/GbfHeHeYtYA

장경배

한성대학교
HANSUNG UNIVERSITY

CryptoCraft LAB
https://crypto.modoo.at

LEA

- ARX 연산으로 구성 된, 128 - bit 블록암호
- 128, 192, 256 – bit 의 키 사이즈가 있음

LEA – 128 / 128 : Key Schedule

- 키 스케쥴을 수행하면 192 - bit 의 라운드 키 집합 RK_i 생성
- 라운드 키를 생성하는 과정에 다음 Constant value 사용

$$\begin{array}{ll} \delta[0] = 0xc3efe9db, & \delta[1] = 0x44626b02, \\ \delta[2] = 0x79e27c8a, & \delta[3] = 0x78df30ec, \\ \delta[4] = 0x715ea49e, & \delta[5] = 0xc785da0a, \\ \delta[6] = 0xe04ef22a, & \delta[7] = 0xe5c40957. \end{array}$$

→ ASCII 코드로 LEA를 뜻함

- 128 – bit 키 $K = (K[0], K[1], K[2], K[3])$ 로 키 스케쥴링 진행 Round key $RK_i = (RK_i[0], RK_i[1], ..., RK_i[5])$
- 라운드 키 RK_i 는 $0 \leq i < 24$ 에 대해, 다음과 같이 생성

$$\begin{aligned} T[0] &\leftarrow \mathrm{ROL}_1(T[0] \boxplus \mathrm{ROL}_i(\delta[i \bmod 4])), \\ T[1] &\leftarrow \mathrm{ROL}_3(T[1] \boxplus \mathrm{ROL}_{i+1}(\delta[i \bmod 4])), \\ T[2] &\leftarrow \mathrm{ROL}_6(T[2] \boxplus \mathrm{ROL}_{i+2}(\delta[i \bmod 4])), \\ T[3] &\leftarrow \mathrm{ROL}_{11}(T[3] \boxplus \mathrm{ROL}_{i+3}(\delta[i \bmod 4])), \\ RK_i &\leftarrow (T[0], T[1], T[2], T[1], T[3], T[1]). \end{aligned}$$

LEA – 128 / 128 : Key Schedule Quantum implementation

- Keywords, Constant Value 큐빗 할당

```
# Qubit
t0 = eng.allocate_qureg(32)  #Key
t1 = eng.allocate_qureg(32)
t2 = eng.allocate_qureg(32)
t3 = eng.allocate_qureg(32)

a0 = eng.allocate_qureg(32)  #Constant
a1 = eng.allocate_qureg(32)
a2 = eng.allocate_qureg(32)
a3 = eng.allocate_qureg(32)


c0 = eng.allocate_qubit()    #carry qubit

text0 = eng.allocate_qureg(32)  # plaintext --> ciphertext
text1 = eng.allocate_qureg(32)
text2 = eng.allocate_qureg(32)
text3 = eng.allocate_qureg(32)
```

LEA – 128 / 128 : Key Schedule Quantum implementation

```
################################ RK(0) ############################################

# ROTL(0) (a0)
Add(eng, a0, t0, 0, 31, 0, 31, c0)

# ROTL(1) (t0)
# 31
# 30

#ROTL(1) (a1)
Add(eng, a1, t1, 31, 30, 0, 31)

#ROTL(3) (t1)
#29
#28

#ROTL(2) (a2)
#30
#29
Add(eng, a2, t2, 30, 29, 0, 31)

#ROTL(6) (t2)
#26
#25

#ROTL(3) (a3)
#29
#28
Add(eng, a3, t3, 29, 28, 0, 31)

#ROTL(11) (t3)
#21
#20
```

for $0 \leq i < 24$

$$
\begin{aligned}
T[0] &\leftarrow \boxed{\mathrm{ROL}_1}(T[0] \boxed{\boxplus} \boxed{\mathrm{ROL}_i}(\delta[i \bmod 4])),\\
T[1] &\leftarrow \mathrm{ROL}_3(T[1] \boxplus \mathrm{ROL}_{i+1}(\delta[i \bmod 4])),\\
T[2] &\leftarrow \mathrm{ROL}_6(T[2] \boxplus \mathrm{ROL}_{i+2}(\delta[i \bmod 4])),\\
T[3] &\leftarrow \mathrm{ROL}_{11}(T[3] \boxplus \mathrm{ROL}_{i+3}(\delta[i \bmod 4])),\\
RK_i &\leftarrow (T[0], T[1], T[2], T[1], T[3], T[1]).
\end{aligned}
$$

Add Function : Ripple Carry Addition

```python
#################### function ########################
def MAJ(eng,a,b,c):

    CNOT | (a, b)
    CNOT | (a, c)
    Toffoli | (c,  b, a)

def UMA(eng,a,b,c):

    Toffoli | (c,  b, a)
    CNOT | (a, c)
    CNOT | (c, b)

def Add(eng, a, b, a_first, a_last, b_first, b_last, c0):
    MAJ(eng, a[a_first], b[b_first], c0)
    for i in range (31):
        MAJ(eng, a[(i + a_first+1)%32], b[(i+b_first+1)%32], a[(a_first+i)%32])

    for i in range (31):
        UMA(eng, a[(a_last - i)%32], b[(b_last - i)%32], a[(a_last-1-i)%32])
    UMA(eng, a[a_first], b[b_first], c0)
```

LEA – 128 / 128 : Encryption Quantum implementation

for $0 \leq i \leq r-1$ (Round r = 24)

X_0 = Plain Text

3 $X_{i+1}[0] \leftarrow \mathrm{ROL}_9((\boxed{X_i[0]} \oplus RK_i[0]) \boxplus (\boxed{X_i[1]} \oplus RK_i[1])),$

2 $X_{i+1}[1] \leftarrow \mathrm{ROR}_5((\boxed{X_i[1]} \oplus RK_i[2]) \boxplus (\boxed{X_i[2]} \oplus RK_i[3])),$

1 $X_{i+1}[2] \leftarrow \mathrm{ROR}_3((\boxed{X_i[2]} \oplus RK_i[4]) \boxplus (X_i[3] \oplus RK_i[5])),$

4 $X_{i+1}[3] \leftarrow \boxed{X_i[0]}.$

- X_{i+1} 의 값을 어디에 저장할지, Quantum 구현 시 값을 덮어씌울 수 없음
- $X_i[0]$ $X_i[1]$ $X_i[2]$ 는 값이 재사용 되지만 $X_i[3]$ 은 재사용 되지 않음

$$\begin{aligned} X_{i+1}[2] &= X_i[3] \\ X_{i+1}[1] &= X_i[2] \\ X_{i+1}[0] &= X_i[1] \\ X_{i+1}[3] &= X_i[0] \end{aligned}$$

LEA – 128 / 128 : Encryption Quantum implementation

for $0 \leq i \leq r-1$ (Round r = 24)

3 $X_{i+1}[0] \leftarrow \mathrm{ROL}_9((X_i[0] \oplus RK_i[0]) \boxplus (X_i[1] \oplus RK_i[1])),$

2 $X_{i+1}[1] \leftarrow \mathrm{ROR}_5((X_i[1] \oplus RK_i[2]) \boxplus (X_i[2] \oplus RK_i[3])),$

1 $X_{i+1}[2] \leftarrow \mathrm{ROR}_3(\boxed{(X_i[2] \oplus RK_i[4])} \boxed{\boxplus} \boxed{(X_i[3] \oplus RK_i[5])}),$

4 $X_{i+1}[3] \leftarrow X_i[0].$

```
################################ ENC(0), X(1) ############################################

for i in range(32):
    CNOT | (t1[ (i+29) % 32], text3[i])

for i in range(32):
    CNOT | (t3[ (i+21) % 32], text2[i])

Add(eng, text2, text3, 0, 31, 0, 31)

#Reverse
for i in range(32):
    CNOT | (t3[ (i+21) % 32], text2[i])

#text2 = text3 ( 3, 2 )
```

* $RK_i \leftarrow (T[0], T[1], T[2], T[1], T[3], T[1]).$

```
#ROTL(3) (t1)
#29
#28

#ROTL(11) (t3)
#21
#20
```

LEA – 128 / 128 : Encryption Quantum implementation

for $0 \leq i \leq r-1$ (Round r = 24)

3 $X_{i+1}[0] \leftarrow \mathrm{ROL}_9((X_i[0] \oplus RK_i[0]) \boxplus (X_i[1] \oplus RK_i[1])),$

2 $X_{i+1}[1] \leftarrow \mathrm{ROR}_5(\boxed{(X_i[1] \oplus RK_i[2])} \boxed{\boxplus} \boxed{(X_i[2] \oplus RK_i[3])}),$

1 $X_{i+1}[2] \leftarrow \mathrm{ROR}_3((X_i[2] \oplus RK_i[4]) \boxplus (X_i[3] \oplus RK_i[5])),$

4 $X_{i+1}[3] \leftarrow X_i[0].$

```
for i in range(32):
    CNOT | (t1[i+29 % 32], text2[i])

for i in range(32):
    CNOT | (t2[ (i+26) % 32 ], text1)

Add(eng, text1, text2, 0, 31, 0, 31)

#Reverse
for i in range(32):
    CNOT | (t2[ (i+26) % 32 ], text1)

#text1 = text2 (5, 4)
```

* $RK_i \leftarrow (T[0], T[1], T[2], T[1], T[3], T[1]).$

```
#ROTL(3) (t1)
#29
#28

#ROTL(6) (t2)
#26
#25
```

LEA – 128 / 128 : Encryption Quantum implementation

for $0 \le i \le r-1$ (Round r = 24)

3 $X_{i+1}[0] \leftarrow \mathrm{ROL}_9(\boxed{(X_i[0] \oplus RK_i[0])} \boxed{\boxplus} \boxed{(X_i[1] \oplus RK_i[1])}),$

2 $X_{i+1}[1] \leftarrow \mathrm{ROR}_5((X_i[1] \oplus RK_i[2]) \boxplus (X_i[2] \oplus RK_i[3])),$

1 $X_{i+1}[2] \leftarrow \mathrm{ROR}_3((X_i[2] \oplus RK_i[4]) \boxplus (X_i[3] \oplus RK_i[5])),$

4 $X_{i+1}[3] \leftarrow X_i[0].$

```
for i in range(32):
    CNOT | (t1[i+29 % 32], text1[i])

for i in range(32):
    CNOT | (t0[ (i+31) % 32 ], text0)

Add(eng, text0, text1, 0, 31, 0, 31)

#Reverse
for i in range(32):
    CNOT | (t0[(i + 31) % 32], text0)

#text0 = text1 (23, 22)
```

* $RK_i \leftarrow (T[0], T[1], T[2], T[1], T[3], T[1]).$

```
#ROTL(3) (t1)
#29
#28

# ROTL(1) (t0)
# 31
# 30
```

LEA – 128 / 128 : Encryption Quantum implementation

for $0 \leq i \leq r-1$ (Round r = 24)

3 $X_{i+1}[0] \leftarrow \mathrm{ROL}_9((X_i[0] \oplus RK_i[0]) \boxplus (X_i[1] \oplus RK_i[1])),$

2 $X_{i+1}[1] \leftarrow \mathrm{ROR}_5((X_i[1] \oplus RK_i[2]) \boxplus (X_i[2] \oplus RK_i[3])),$

1 $X_{i+1}[2] \leftarrow \mathrm{ROR}_3((X_i[2] \oplus RK_i[4]) \boxplus (X_i[3] \oplus RK_i[5])),$

4 $\boxed{X_{i+1}[3] \leftarrow X_i[0].}$

Step 3 에서 $X_i[0]$ 에 대해 Reverse 연산을 해 주었음

`#text3 = text0` (0, 31)

Round 0 끝

$$
\begin{aligned}
X_{i+1}[2] &= X_i[3] \\
X_{i+1}[1] &= X_i[2] \\
X_{i+1}[0] &= X_i[1] \\
X_{i+1}[3] &= X_i[0]
\end{aligned}
$$

다음 Round 1 은 이 상태로 진행하고 사전에 RK_1 에 대한 Key Schedule 수행

감사합니다