SIMECK에 대한 양자회로 최적화 구현

https://youtu.be/H4rLPqXEkPo

송경주

SIMECK

- SIMECK
  - SIMECK : SIMON과 SPECK의 장점을 결합하여 설계한 경량암호
  - SIMON의 라운드 함수를 수정하여 SPECK과 유사하게 키 스케줄 함수에서 재사용
  - Feistel 구조의 라운드 함수와 키 스케줄 함수로 동작함

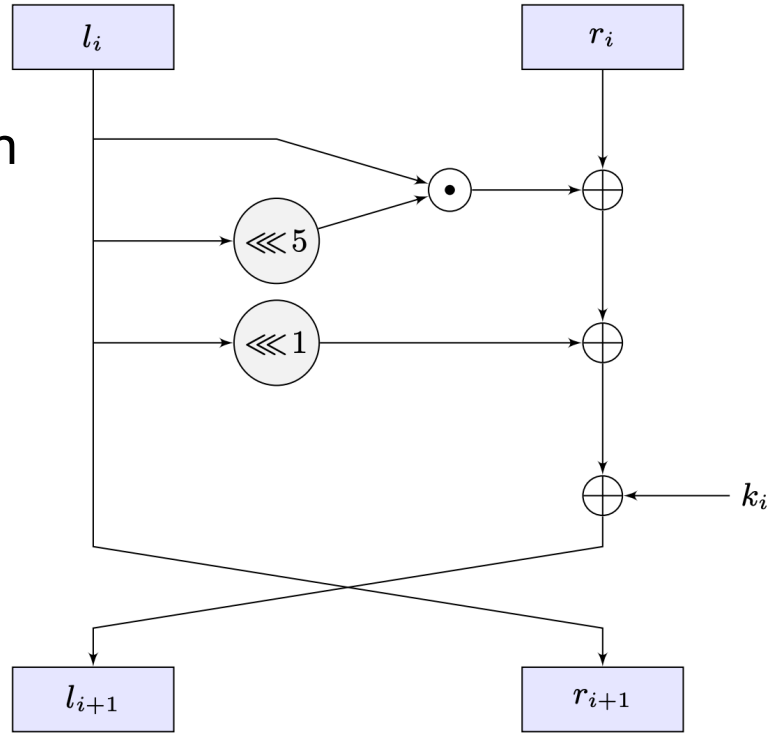
<SIMECK 라운드 함수>

- 평문이 두개의 word l_i, r_i 로 나뉘며 최상위 n 비트가 l_0, 최하위 n 비트가 r_0이 된다.

$$R_{k_i}(l_i, r_i) = (r_i \oplus f(l_i) \oplus k_i, l_i)$$

- 라운드 함수의 내부 함수는 $f(x) = (x \odot (x \lll 5)) \oplus (x \lll 1)$ 로 동작한다.

- 내부함수에서는 두 개의 입력 중 첫 번째 입력의 rotation을 통한 AND 값을 두 번째 입력에 XOR하는 연산이 수행된다.



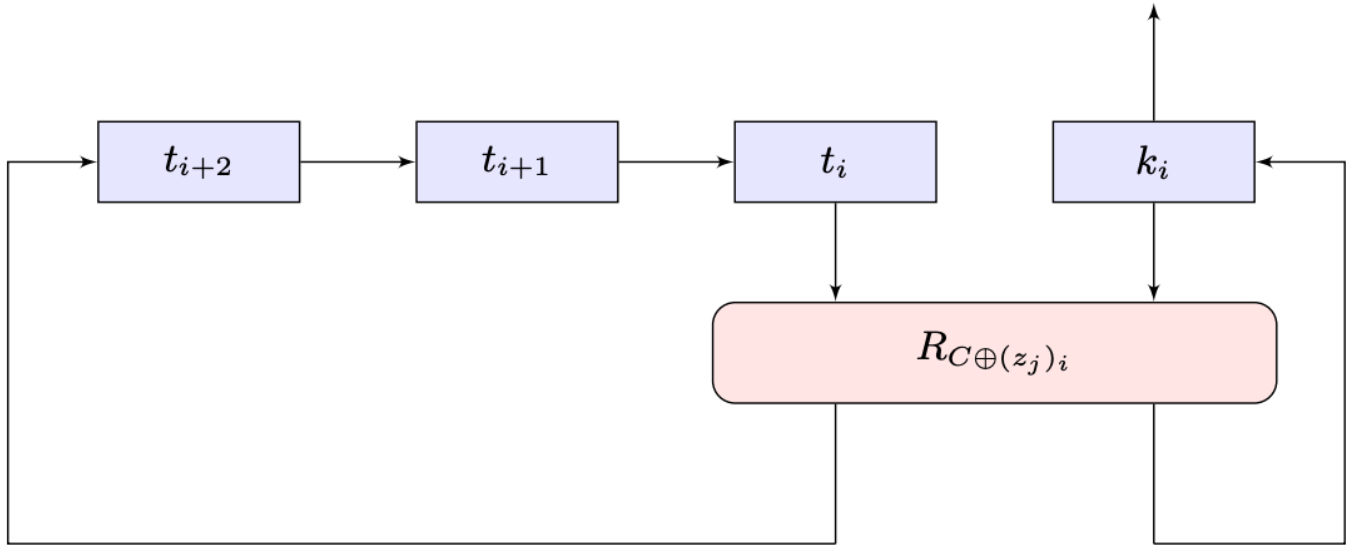
SIMECK

<SIMECK 키 스케줄링 함수>

- 키 스케줄링 함수에서는 라운드 함수에서 사용할 라운드 키를 생성한다.
- 키 스케줄에서 생성된 키는 라운드 함수의 두번째 입력에 XOR 된다.
- 마스터 키 K를 4개의 word (t_2, t_1, t_0, k_0)로 나누어 라운드 키를 생성한다.

$$\begin{cases} k_{i+1} = t_i, \\ t_{i+3} = k_i \oplus f(t_i) \oplus C \oplus (z_j)_i \end{cases}$$

- 키 스케줄링 함수의 내부 함수 $f(x)$ 는 라운드 함수의 내부함수와 동일함



SIMECK 양자회로 구현

- SIMECK 양자회로에서는 양자자원을 줄이기 위한 효율적인 방법을 사용하였음

< SIMECK 라운드 함수 양자회로 구현>

- Ancilla 큐비트를 줄이기 위해 $x \odot (x \lll 5)$ 연산에서 발생하는 temp 값을 따로 저장하지 않고 Toffoli gate를 사용하여 연산 대상 큐비트에 바로 저장되도록 구현함

- Shift 연산에 대해서는 Swap 게이트를 사용하지 않고 연산 큐비트의 인덱스를 변경하여 동작하도록 함

Algorithm 1 : SIMECK quantum circuit for round function

Input : $l_r,\ r_r$

Output : $l_r,\ r_r,\ k_r$

for i in range(length(r_r)):

    $r_r \leftarrow$ Toffoli($l_r,\ l_r \ggg 11,\ r_r$)

    $r_r \leftarrow$ CNOT($l_r \ggg 15,\ r_r$)

for i in range(length(r_r)):

    $r_r \leftarrow$ CNOT($k_r,\ r_r$)

Swap($l_r,\ r_r$)

SIMECK 양자회로 구현

< SIMECK 키 스케줄 함수 양자회로 구현 >

- 양자자원을 줄이기 위해 키 스케줄 함수를 변경하여 사용함
- 기존 키 스케줄 함수에서는 key 대신 constant와 CNOT연산을 진행함 → CNOT 게이트를 사용하기 위해서는 constant 를 양자자원(큐비트) 으로 할당하여 사용해야 함
- 제안하는 양자회로에서는 constant와 큐비트 classic to quantum 연산을 진행함

→ Constant를 classic 자원인 정수로 할당하여 큐비트 및 양자 게이트를 사용하지 않고 업데이트 함

→ constant의 1인 인덱스 위치에 X 게이트가 동작하도록 설계

Algorithm 2 : SIMECK quantum circuit for key schedule

Input : l_r, r_r

Output : l_r, r_r, constant

for i **in range**(length(r_r)):
  $r_r \leftarrow$ Toffoli(l_r, $l_r \ggg 11$, r_r)
  $r_r \leftarrow$ CNOT($l_r \ggg 15$, r_r)

if constant[i] = 1
  X (r_r)

for i **in range**(length(r_r)):
  $r_r \leftarrow$ CNOT(k_r, r_r)

Swap(l_r, r_r)

평가

- 개발 환경 : IBM 에서 제공하는 양자 프로그래밍 툴 ProjectQ
- 양자 자원 추정: ProjectQ 에서 제공하는 ResourceSimulation 사용

- <표 1>은 SIMECK (블록 크기)/(키 길이)에 대한 암호 양자회로 자원 추정 결과를 보여줌
- 표에 제시된 양자자원 결과는 암호화가 한번 진행될 때 필요한 양자자원을 나타냄
- SIMECK 암호에 대한 brute-force attack 수행에는 Grover's algorithm 내부의 Oracle 반복 만큼 암호화를 진행 : $2 \times$ <표 1> $\times \left\lfloor \frac{\pi}{4} \times \sqrt{2^{key\ length}} \right\rfloor$ 의 양자 자원 필요

Algorithm	Quantum gates			
	X	CNOT	Toffoli	Depth
SIMECK 32/64	465	1,536	1,024	214
SIMECK 48/96	792	2,592	1,728	232
SIMECK 64/128	1,320	4,224	2,816	361

<표 1> SIMECK (블록 크기)/(키 길이)에 대한 암호 양자회로 자원 추정 결과

평가

- 개발 환경 : IBM 에서 제공하는 양자 프로그래밍 툴 ProjectQ
- 양자 자원 추정: ProjectQ 에서 제공하는 ResourceSimulation 사용

- SIMECK 양자회로 구현에 대한 선행연구가 없으므로 비슷한 경량암호 SIMON와 비교함
- SIMECK과 동일한 (블록 사이즈)/(키 사이즈) 를 가지는 SIMON과 비교했을 때, SIMECK 이 SIMON 보다 더 많은 Toffoli 게이트를 사용하였지만 Depth가 훨씬 작게 구현 되었음

Algorithm	Quantum gates			
	X	CNOT	Toffoli	Depth
SIMECK 32/64	465	1,536	1,024	214
SIMECK 48/96	792	2,592	1,728	232
SIMECK 64/128	1,320	4,224	2,816	361

<표 2> SIMECK 양자회로 자원 추정 결과

Algorithm	Quantum gates			
	X	CNOT	Toffoli	Depth
SIMECK 32/64	448	2,816	512	946
SIMECK 48/96	768	4,800	864	1,597

<표 3> SIMON 양자회로 자원 추정 결과

Q & A