

명세서

발명의 명칭: 동형 암호화를 수행하는 단말 장치와 그 암호문을 처리하는 서버 장치 및 그 방법들

기술분야

- [1] 본 발명은 동형 암호화를 수행하는 단말 장치와 그 암호문을 처리하는 서버 장치 및 그 방법과 링의 평문 공간을 실수 공간으로 조정하여 연산 속도가 향상된 동형 암호문을 생성하는 실수 평문에 대한 동형 암호화 방법에 관한 것이다.

배경기술

- [2] 전자 및 통신 기술의 발달에 힘입어 다양한 장치들 간에 데이터를 송수신하여 활용하는 다양한 서비스가 지원되고 있다. 그 중 일 예로, 사용자가 자신의 개인 정보 등을 서버에 저장하여 두고, 자신의 단말 장치를 이용하여 서버의 정보를 이용하는 클라우드 컴퓨팅 서비스도 활발하게 사용되고 있다.
- [3] 이러한 환경에서는 데이터 유출을 방지하기 위한 보안 기술의 사용이 필수적이다. 따라서, 서버는 암호화된 데이터를 저장하게 된다. 이 경우, 서버는 저장된 데이터를 검색하거나 그 데이터에 기초한 일련의 작업을 할 때마다, 암호화된 데이터를 복호화하여야 하므로, 자원 및 시간 낭비가 발생하게 된다.
- [4] 또한, 서버에서 연산을 위해 일시적으로 복호화한 상태에서 제3자의 해킹이 이루어지는 경우, 개인 정보가 제3자에게 손쉽게 유출될 수 있다는 문제점도 있었다.
- [5] 이러한 문제를 해결하기 위하여 동형 암호화 방법이 연구되고 있다. 동형 암호화에 따르면, 암호화된 정보를 복호화하지 않고 암호문 자체에서 연산을 하더라도, 평문에 대해 연산한 후 암호화한 값과 동일한 결과를 얻을 수 있다. 따라서, 암호문을 복호화하지 않은 상태에서 각종 연산을 수행할 수 있다.
- [6] 하지만, 종래의 동형 암호화 방법에 따라 정수 연산을 수행하게 되면, 평문의 비트수가 기하급수적으로 증가하여, 연산에 소요되는 시간이 크게 증가하는 문제가 있었다.
- [7] 한편, 기존의 동형 암호화 방식은 연산 과정에서 평문의 허수부가 커지는 단점이 있었다. 구체적으로, 기존의 동형 암호화 방식 복소수 공간에서 메시지를 동형 암호화하였다는 점에서, 실수 공간의 메시지를 동형 암호화하는 경우에도 허수부에 에러가 추가되었다. 만약 동형 암호문에 대한 연산이 반복되는 경우, 허수부의 에러가 점차 커지게 되고, 허수부의 에러가 많이 커지게 되면 이후의 동형 곱셈 연산 과정에서 실제 메시지에 해당하는 실수부의 메시지가 손상될 수 있었다.
- [8] 따라서, 암호문 상태에서도 연산 속도를 향상할 수 있으며, 연산 과정에서 평문의 허수부가 커지는 것을 차단할 수 있는 방법이 요구되었다.

발명의 상세한 설명

기술적 과제

- [9] 본 발명은 이상과 같은 문제점을 해결하기 위하여 고안된 것으로, 본 발명의 목적은 메시지에 스케일링 팩터를 반영한 후 암호화하여 동형 암호문을 생성하는 단말 장치와 그 암호문을 연산하는 서버 장치 및 그 방법들을 제공하고, 링의 평문 공간을 실수 공간으로 조정하여 연산 속도가 향상된 동형 암호문을 생성하는 실수 평문에 대한 동형 암호화 방법을 제공함에 있다.

과제 해결 수단

- [10] 이상과 같은 목적을 달성하기 위하여, 본 발명의 일 실시 예에 따른 단말 장치의 암호화 방법은, 스케일링 팩터를 설정하는 단계, 암호화할 메시지에 상기 스케일링 팩터를 반영한 후 공개키를 이용하여 암호화하여 동형 암호문을 생성하는 단계를 포함한다. 여기서, 상기 동형 암호문은, 복호화하였을 때 상기 메시지에 상기 스케일링 팩터를 반영한 값에 에러 값을 가산한 결과값이 복원되는 형태가 될 수 있다.
- [11] 그리고, 상기 동형 암호문의 모듈러스는, 하나의 스케일링 팩터의 지수승으로 설정될 수 있다.
- [12] 또는, 상기 동형 암호문의 모듈러스는, 복수의 서로 다른 스케일링 팩터들을 승산한 값으로 설정될 수도 있다. 여기서, 상기 복수의 서로 다른 스케일링 팩터들은 유사 범위 이내에서 서로 소 관계인 값들로 설정될 수 있다.
- [13] 또한, 상기 동형 암호문을 생성하는 단계는, 이산 가우시안 분포 또는 상기 이산 가우시안 분포와 통계적 거리가 가까운 분포로부터 에러를 산출하는 단계, 상기 메시지에 상기 스케일링 팩터를 승산하고 상기 에러를 가산한 후, 상기 공개키를 이용하여 암호화하여 상기 암호문을 생성하는 단계를 포함할 수 있다.
- [14] 또는, 상기 메시지가 복수의 메시지 벡터인 경우, 상기 복수의 메시지 벡터를, 병렬적으로 암호화할 수 있는 형태의 다항식으로 변환하는 단계를 더 포함할 수도 있다. 이 경우, 상기 동형 암호문을 생성하는 단계는, 상기 다항식에 상기 스케일링 팩터를 승산한 후, 상기 공개키를 이용하여 동형 암호화할 수 있다.
- [15] 한편, 본 발명의 일 실시 예에 따른 서버 장치의 암호문 처리 방법은, 개별적으로 암호화된 복수의 동형 암호문을 수신하는 단계, 상기 복수의 동형 암호문에 대하여 기 설정된 연산을 수행하는 단계, 상기 연산에 의해 산출된 결과 암호문으로부터 노이즈 영역을 제거하여 유효 영역의 데이터를 추출하는 단계를 포함한다. 여기서, 상기 복수의 동형 암호문 각각은 메시지에 스케일링 팩터를 승산 한 후 암호화한 암호문이며, 상기 노이즈 영역은 상기 스케일링 팩터의 크기에 대응되도록 결정된다.
- [16] 또한, 상기 복수의 동형 암호문 각각이 복수의 메시지 벡터를 패킹한 암호문이면, 상기 복수의 암호문 각각에 포함된 메시지 벡터들을 병렬적으로 연산 처리하는 단계를 더 포함할 수도 있다.

- [17] 이 경우, 상기 복수의 암호문에 포함된 각 메시지 벡터들 간의 순서를 로테이션시키는 단계를 더 포함할 수도 있다.
- [18] 또는, 상기 메시지가 복소수인 경우, 상기 복수의 암호문에 대하여 켄쥬게이션(conjugation) 연산을 수행하는 단계를 더 포함할 수도 있다.
- [19] 본 발명의 일 실시 예에 따른 단말 장치는, 스케일링 팩터 및 공개키가 저장된 메모리, 암호화할 메시지에 상기 스케일링 팩터를 반영한 후 상기 공개키를 이용하여 암호화하여 동형 암호문을 생성하는 프로세서 및 상기 동형 암호문을 외부 장치로 전송하는 통신부를 포함한다. 여기서, 상기 동형 암호문은, 복호화하였을 때 상기 메시지에 상기 스케일링 팩터를 반영한 값에 에러 값을 승산한 결과값이 복원되는 형태가 될 수 있다.
- [20] 또한, 상기 메시지 및 상기 스케일링 팩터를 입력받기 위한 입력부;를 더 포함할 수도 있다. 상기 프로세서는 상기 입력부를 통해 입력된 상기 메시지 및 상기 스케일링 팩터를 상기 메모리에 저장할 수도 있다.
- [21] 또한, 상기 프로세서는, 동형 암호문의 모듈러스를 상기 스케일링 팩터의 지수승으로 설정하여 상기 메모리에 저장할 수도 있다.
- [22] 또는, 상기 프로세서는, 상기 동형 암호문의 모듈러스를 복수의 서로 다른 스케일링 팩터들을 승산한 값으로 설정하여 상기 메모리에 저장할 수도 있다. 이 경우, 상기 복수의 서로 다른 스케일링 팩터들은 유사 범위 이내에서 서로 소 관계인 값들로 설정될 수 있다.
- [23] 또한, 상기 프로세서는, 이산 가우시안 분포 또는 상기 이산 가우시안 분포와 통계적 거리가 가까운 분포로부터 에러를 산출하고, 상기 메시지 및 상기 스케일링 팩터를 승산한 값에 상기 에러를 가산한 후, 상기 공개키를 이용하여 암호화할 수도 있다.
- [24] 또한, 상기 프로세서는, 상기 메시지가 복수의 메시지 벡터인 경우, 상기 복수의 메시지 벡터를, 병렬적으로 암호화할 수 있는 형태의 다항식으로 변환하며, 상기 다항식에 상기 스케일링 팩터를 승산한 후, 상기 공개키를 이용하여 동형 암호화할 수도 있다.
- [25] 한편, 이상과 같은 목적을 달성하기 위하여, 본 개시의 일 실시 예에 따른 단말 장치의 암호화 방법은, 원소들 사이에 덧셈과 곱셈이 정의되어 있으며 덧셈과 곱셈에 대해서 닫혀 있는 집합 중 평문 공간이 실수인 집합인 링(Ring)을 설정하는 단계, 상기 링으로부터 비밀키를 산출하는 단계, 상기 산출된 비밀키에 대응하여 공개키를 산출하는 단계, 및 메시지에 상기 공개키를 적용하여 동형 암호문을 생성하는 단계를 포함한다.
- [26] 이 경우, 상기 링은 아래의 수학식을 만족할 수 있다.
- [27] $a(X)=a(X^{-1})$ where $a(X) \in R, X^{-1} := -X^{n-1}$
- [28] 여기서, $a(X)$ 는 상기 링에 속하는 다항식, R 은 평문 공간이 복소수인 링.
- [29] 한편, 상기 동형 암호문은, 덧셈 또는 곱셈 연산에 대해서 $O(n \log n)$

복잡도(complexity)를 가질 수 있다.

- [30] 한편, 본 암호화 방법은 상기 링으로부터 제1 랜덤 다항식을 산출하는 단계, 에러를 추출하는 단계, 및 상기 제1 랜덤 다항식 및 상기 비밀키에 상기 에러를 모듈러 연산하여 제2 랜덤 다항식을 산출하는 단계를 더 포함하고, 상기 공개키를 산출하는 단계는, 상기 제1 랜덤 다항식 및 상기 제2 랜덤 다항식을 이용하여 상기 공개키를 산출할 수 있다.
- [31] 한편, 상기 동형 암호문을 생성하는 단계는, 상기 메시지를 상기 링에 속하는 다항식으로 산출하고, 상기 산출된 다항식에 상기 공개키를 적용하여 동형 암호문을 생성할 수 있다.
- [32] 한편, 본 암호화 방법은 상기 생성된 동형 암호문 및 기생성된 동형 암호문에 대해서 덧셈 및 곱셈 중 적어도 하나의 연산을 수행하는 단계를 더 포함하고, 상기 연산을 수행하는 단계는, 상기 생성된 동형 암호문 및 상기 기생성된 동형 암호문 각각을 제1함수 처리하고, 제1함수 처리된 동형 암호문 간에 연산을 수행하고, 상기 연산 수행된 동형 암호문을 상기 제1 함수에 역함수인 제2 함수 처리할 수 있다.
- [33] 이 경우, 상기 제1 함수는, NTT(Number Theoretic Transform) 함수일 수 있다.
- [34] 한편, 상기 동형 암호문은, 복호화하였을 때 상기 메시지에 에러 값을 가산한 결과값이 복원되는 형태일 수 있다.
- [35] 한편, 본 개시의 일 실시 예에 따른 단말 장치는 메시지를 저장하는 메모리, 및 원소들 사이에 덧셈과 곱셈이 정의되어 있으며 덧셈과 곱셈에 대해서 닫혀 있는 집합 중 평문 공간이 실수인 집합인 링(Ring)을 설정하고, 상기 링으로부터 비밀키를 산출하고, 상기 산출된 비밀키에 대응하여 공개키를 산출하고, 상기 메모리에 저장된 메시지에 상기 공개키를 적용하여 동형 암호문을 생성하는 프로세서를 포함한다.
- [36] 이 경우, 상기 링은 아래의 수학적식을 만족할 수 있다.
- [37] $a(X)=a(X^{-1})$ where $a(X) \in R, X^{-1} := -X^{n-1}$
- [38] 여기서, $a(X)$ 는 상기 링에 속하는 다항식, R 은 평문 공간이 복소수인 링.
- [39] 한편, 상기 동형 암호문은 덧셈 또는 곱셈 연산에 대해서 $O(n \log n)$ 복잡도(complexity)를 가질 수 있다.
- [40] 한편, 상기 프로세서는 상기 링으로부터 제1 랜덤 다항식을 산출하고, 에러를 추출하고, 상기 제1 랜덤 다항식 및 상기 비밀키에 상기 에러를 모듈러 연산하여 제2 랜덤 다항식을 산출하고, 상기 제1 랜덤 다항식 및 상기 제2 랜덤 다항식을 이용하여 상기 공개키를 산출할 수 있다.
- [41] 한편, 상기 프로세서는 상기 메시지를 상기 링에 속하는 다항식으로 산출하고, 상기 산출된 다항식에 상기 공개키를 적용하여 동형 암호문을 생성할 수 있다.
- [42] 한편, 상기 프로세서는 상기 생성된 동형 암호문 및 상기 기저장된 동형 암호문 각각을 제1함수 처리하고, 제1함수 처리된 동형 암호문 간에 덧셈 및 곱셈 중

적어도 하나의 연산을 수행하고, 상기 연산 수행된 동형 암호문을 상기 제1 함수에 역함수인 제2 함수 처리할 수 있다.

- [43] 이 경우, 상기 제1 함수는 NTT(Number Theoretic Transform) 함수일 수 있다.

발명의 효과

- [44] 이상과 같은 본 발명의 다양한 실시 예들에 따르면, 유효 영역을 포함하는 동형 암호문을 생성할 수 있게 되므로, 메시지 손실을 최소화하면서, 연산 시간 및 부담을 경감시킬 수 있게 된다.

- [45] 그리고, 링의 평문 공간을 실수 공간으로 조정하여 동형 암호문을 생성하는바, 연산 과정에서 평문의 허수부가 커지는 것을 원천적으로 차단할 수 있으며, 기존 HEAAN과 같은 안정성, 연산속도, 메모리(암호문, 키 사이즈)를 가짐과 동시에 2배 많은 병렬 실수 연산을 지원 가능하게 된다.

도면의 간단한 설명

- [46] 도 1은 본 발명의 일 실시 예에 따른 네트워크 시스템의 구조를 설명하기 위한 도면,
 [47] 도 2는 본 발명의 일 실시 예에 따른 단말 장치 및 서버 장치의 구성을 나타내는 블록도,
 [48] 도 3은 본 발명의 일 실시 예에 따른 단말 장치의 암호화 방법을 설명하기 위한 흐름도,
 [49] 도 4는 본 발명의 일 실시 예에 따른 서버 장치의 암호문 처리 방법을 설명하기 위한 흐름도,
 [50] 도 5 내지 도 7은 서버 장치의 암호문 처리 방법의 다양한 예들을 설명하기 위한 도면,
 [51] 도 8은 본 발명의 일 실시 예에 따른 네트워크 시스템의 암호문 처리 방법을 설명하기 위한 도면,
 [52] 도 9는 본 개시의 일 실시 예에 따른 네트워크 시스템의 구조를 설명하기 위한 도면,
 [53] 도 10은 본 개시의 일 실시 예에 따른 단말 장치의 간단한 구성을 도시한 블록도,
 [54] 도 11은 본 개시의 일 실시 예에 따른 단말 장치의 구체적인 구성을 도시한 블록도,
 [55] 도 12는 공개키 및 비밀키의 생성 동작을 설명하기 위한 도면,
 [56] 도 13은 연산 키의 생성 동작을 설명하기 위한 도면,
 [57] 도 14는 근사 동형 암호문의 생성 및 복호 동작을 설명하기 위한 도면,
 [58] 도 15는 동형 암호문의 덧셈 계산을 설명하기 위한 도면,
 [59] 도 16은 동형 암호문의 곱셈 계산을 설명하기 위한 도면,
 [60] 도 17은 본 개시의 암호화 방법을 설명하기 위한 흐름도, 그리고,
 [61] 도 18은 본 개시의 일 실시 예에 따른 네트워크 시스템의 암호문 처리 방법을

설명하기 위한 도면이다.

발명의 실시를 위한 최선의 형태

- [62] 이하에서는 첨부 도면을 참조하여 본 발명에 대해서 자세하게 설명한다. 본 명세서에서 수행되는 정보(데이터) 전송 과정은 필요에 따라서 암호화/복호화가 적용될 수 있으며, 본 명세서 및 특허청구범위에서 정보(데이터) 전송 과정을 설명하는 표현은 별도로 언급되지 않더라도 모두 암호화/복호화하는 경우도 포함하는 것으로 해석되어야 한다. 본 명세서에서 "A로부터 B로 전송(전달)" 또는 "A가 B로부터 수신"과 같은 형태의 표현은 중간에 다른 매개체가 포함되어 전송(전달) 또는 수신되는 것도 포함하며, 반드시 A로부터 B까지 직접 전송(전달) 또는 수신되는 것만을 표현하는 것은 아니다.
- [63] 본 발명의 설명에 있어서 각 단계의 순서는 선행 단계가 논리적 및 시간적으로 반드시 후행 단계에 앞서서 수행되어야 하는 경우가 아니라면 각 단계의 순서는 비제한적으로 이해되어야 한다. 즉, 위와 같은 예외적인 경우를 제외하고는 후행 단계로 설명된 과정이 선행단계로 설명된 과정보다 앞서서 수행되더라도 발명의 본질에는 영향이 없으며 권리범위 역시 단계의 순서에 관계없이 정의되어야 한다. 그리고 본 명세서에서 "A 또는 B"라고 기재한 것은 A와 B 중 어느 하나를 선택적으로 가리키는 것뿐만 아니라 A와 B 모두를 포함하는 것도 의미하는 것으로 정의된다. 또한, 본 명세서에서 "포함"이라는 용어는 포함하는 것으로 나열된 요소 이외에 추가로 다른 구성요소를 더 포함하는 것도 포괄하는 의미를 가진다.
- [64] 본 명세서에서는 본 발명의 설명에 필요한 필수적인 구성요소만을 설명하며, 본 발명의 본질과 관계가 없는 구성요소는 언급하지 아니한다. 그리고 언급되는 구성요소만을 포함하는 배타적인 의미로 해석되어서는 아니되며 다른 구성요소도 포함할 수 있는 비배타적인 의미로 해석되어야 한다.
- [65] 그리고 본 명세서에서 "값"이라 함은 스칼라값 뿐만 아니라 벡터도 포함하는 개념으로 정의된다.
- [66] 후술하는 본 발명의 각 단계의 수학적 연산 및 산출은 해당 연산 또는 산출을 하기 위해 공지되어 있는 코딩 방법 및/또는 본 발명에 적합하게 고안된 코딩에 의해서 컴퓨터 연산으로 구현될 수 있다.
- [67] 이하에서 설명하는 구체적인 수학적식은 가능한 여러 대안 중에서 예시적으로 설명되는 것이며, 본 발명의 권리 범위가 본 명세서에 언급된 수학적식에 제한되는 것으로 해석되어서는 아니된다.
- [68] 설명의 편의를 위해서, 본 명세서에서는 다음과 같이 표기를 정하기로 한다.
- [69] $a \leftarrow D$: 분포(D)에 따라서 원소(a)를 선택함
- [70] $s_1, s_2 \in R$: S1, S2 각각은 R 집합에 속하는 원소이다.
- [71] $\text{mod}(q)$: q 원소로 모듈(modular) 연산
- [72] $[\cdot]$: 내부 값을 반올림함

- [73] 이하에서는 첨부된 도면을 이용하여 본 발명의 다양한 실시 예들에 대하여 구체적으로 설명한다.
- [74] 도 1은 본 발명의 일 실시 예에 따른 네트워크 시스템의 구성을 나타내는 도면이다. 도 1에 따르면, 복수의 단말장치(100-1 ~ 100-n), 제1 서버 장치(200), 제2 서버 장치(300)가 네트워크(10)를 통해 서로 연결될 수 있다. 네트워크(10)는 다양한 형태의 유무선 통신 네트워크, 방송 통신 네트워크, 광통신 네트워크, 클라우드 네트워크 등으로 구현될 수 있으며, 각 장치들은 별도의 매개체 없이 와이파이, 블루투스, NFC(Near Field Communication) 등과 같은 방식으로 연결될 수도 있다.
- [75] 도 1에서는 단말 장치가 복수 개(100-1 ~ 100-n)인 것으로 도시하였으나, 반드시 복수 개의 단말 장치가 사용되어야 하는 것은 아니며 하나의 장치가 사용될 수도 있다. 일 예로, 단말 장치(100-1 ~ 100-n)는 휴대폰, 태블릿, 게임 플레이어, PC, 랩탑 PC, 홈서버, 키오스크 등과 같은 다양한 형태의 장치로 구현될 수 있으며, 이밖에 IoT 기능이 적용된 가전 제품 형태로도 구현될 수 있다.
- [76] 사용자는 자신이 사용하는 단말 장치(100-1 ~ 100-n)를 통해서 다양한 정보를 입력할 수 있다. 입력된 정보는 단말 장치(100-1 ~ 100-n) 자체에 저장될 수도 있지만, 저장 용량 및 보안 등을 이유로 외부 장치로 전송되어 저장될 수도 있다. 도 1에서 제1 서버 장치(200)는 이러한 정보들을 저장하는 역할을 수행하고, 제2 서버 장치(300)는 제1 서버 장치(200)에 저장된 정보의 일부 또는 전부를 이용하는 역할을 수행한다.
- [77] 각 단말 장치(100-1 ~ 100-n)는 입력된 정보를 동형 암호화하여, 동형 암호문을 제1 서버 장치(200)로 전송한다. 제1 서버 장치(200)는 전송된 동형 암호문을 복호화하지 않고, 암호문 상태로 저장할 수 있다.
- [78] 제2 서버 장치(300)는 동형 암호문에 대한 특정 처리 결과를 제1 서버 장치(200)로 요청할 수 있다. 제1 서버 장치(200)는 요청에 따라 특정 연산을 수행한 후, 그 결과를 제2 서버 장치(300)로 전송한다. 일 예로, 두 개의 단말 장치(100-1, 100-2)가 전송한 암호문 ct_1 , ct_2 가 제1 서버 장치(200)에 저장된 경우, 제2 서버 장치(300)는 두 단말 장치(100-1, 100-2)로부터 제공된 정보들을 합산한 값을 제1 서버 장치(200)로 요청할 수 있다. 제1 서버 장치(200)는 요청에 따라 두 암호문을 합산하는 연산을 수행한 후, 그 결과값($ct_1 + ct_2$)을 제2 서버 장치(300)로 전송할 수 있다.
- [79] 각 단말 장치는 동형 암호화를 수행하는 과정에서 산출되는 암호화 노이즈, 즉, 에러를 암호문에 포함시킨다. 구체적으로는, 각 단말 장치(100-1 ~ 100-n)에서 생성하는 동형 암호문은, 추후에 비밀키를 이용하여 복호화하였을 때 메시지 및 에러 값을 포함하는 결과값이 복원되는 형태로 생성될 수 있다.
- [80] 일 예로, 단말 장치(100-1 ~ 100-n)에서 생성하는 동형 암호문은 비밀키를 이용하여 복호화 하였을 때 다음과 같은 성질을 만족하는 형태로 생성된다.
- [81] [수학식 1]

- [82] $\text{Dec}(ct, sk) = \langle ct, sk \rangle = \Delta M + e \pmod{q}$
- [83] 여기서 $\langle, \rangle$ 는 내적 연산(usual inner product), ct 는 암호문, sk 는 비밀키, M 은 평문 메시지, e 는 암호화 에러 값, Δ 는 스케일링 팩터(scaling factor), 그리고 q 는 암호문의 모듈러스(Modulus)를 의미한다. q 는 스케일링 팩터가 메시지에 곱해진 결과값 ΔM 보다 크게 선택되어야 한다. 에러 값 e 의 절대값이 ΔM 에 비해서 충분히 작다면, 암호문의 복호화 값 $\Delta M + e$ 는 유효숫자연산에서 원래의 메시지를 동일한 정밀도로 대체할 수 있는 값이다. 복호화된 데이터 중에서 에러는 최하위 비트(LSB) 측에 배치되고, ΔM 은 차하위 비트 측에 배치된다.
- [84] 메시지의 크기가 너무 작거나 너무 큰 경우, 스케일링 팩터를 이용하여 그 크기를 조절할 수도 있다. 스케일링 팩터를 사용하게 되면, 정수 형태의 메시지 뿐만 아니라 실수 형태의 메시지까지도 암호화할 수 있게 되므로, 활용성이 크게 증대할 수 있다. 또한, 스케일링 팩터를 이용하여 메시지의 크기를 조절함으로써, 연산이 이루어지고 난 이후의 암호문에서 메시지들이 존재하는 영역, 즉, 유효 영역의 크기도 조절될 수 있다.
- [85] 실시 예에 따라, 암호문 모듈러스 q 는 다양한 형태로 설정되어 사용될 수 있다. 일 예로, 암호문의 모듈러스는 스케일링 팩터 Δ 의 지수승 $q = \Delta^L$ 형태로 설정될 수 있다. Δ 가 2라면, $q = 2^{10}$ 과 같은 값으로 설정될 수 있다.
- [86] 다른 예로, 암호문 모듈러스는 복수의 서로 다른 스케일링 팩터들을 곱한 값으로 설정될 수도 있다. 각 팩터들은 유사 범위 이내의 값, 즉, 서로 비슷한 크기의 값으로 설정된다. 구체적으로는, $q = q_1 \cdot q_2 \cdot q_3 \cdots q_x$ 로 설정되며, $q_1, q_2, q_3, \dots, q_x$ 각각은 스케일링 인수 Δ 와 비슷한 크기이며 서로 소 관계의 값으로 설정된다. 스케일링 팩터를 이와 같은 방식으로 설정하게 되면, CRT(Chinese Remainder Theorem)에 따라 전체 연산을 복수 개의 모듈러스 연산으로 분리하여 진행할 수 있게 되므로, 연산 부담을 경감시킬 수 있다. 또한, 서로 비슷한 크기의 팩터를 사용함에 따라, 후술하는 단계에서 라운딩 처리를 수행하였을 때, 앞선 예에서의 결과값과 거의 같은 결과를 얻을 수 있다.
- [87] 이상과 같이, 각 단말 장치(100-1 ~ 100-n)에서 생성한 암호문은 제1 서버 장치(200)에 의해 처리된 후, 암호문 상태로 제2 서버 장치(300)로 전송된다. 제2 서버 장치(300)는 전송된 암호문을 비밀키를 이용하여 복호화한다. 동형 암호화 처리가 되었기 때문에, 복호화된 데이터는 메시지 자체를 복호화한 후 연산 처리한 결과와 같은 결과가 될 수 있다. 결과적으로 중간 과정에서 제3자에게 유출될 위험을 방지할 수 있게 된다.
- [88] 도 1의 실시 예에서 암호화를 수행하는데 사용되는 공개 키는 각 단말 장치에 의해 생성될 수도 있고, 복호화를 수행하는 제2 서버 장치(300)에서 생성한 후 각 단말 장치로 배포할 수도 있다.
- [89] 키 생성 과정에서는 Ring-LWE 기법이 사용될 수 있다. 제2 서버 장치(300)가 키를 생성하는 경우로 가정하면, 제2 서버 장치(300)는 각종 파라미터 및 링을 설정한다. 구체적으로는, 평문 메시지 비트의 길이, 공개 키 및 비밀키의 크기

등과 같은 다양한 파라미터들을 설정할 수 있다.

[90] 링은 다음과 같은 수학적식으로 표현될 수 있다.

[91] [수학식 2]

$$[92] \quad R = \mathbb{Z}_q[x]/(f(x))$$

[93] 링(Ring)이란 기 설정된 계수를 가지는 다항식의 집합을 의미한다. 일 예로, 링은 계수가 $\mathbb{Z}_q$ 인 n 차 다항식의 집합을 의미한다. 수학식 2에서 $f(x)$ 는 n 차 다항식을 의미한다. 구체적으로는, n 이 $\Phi(N)$ 일 때, N 차 사이클로토믹 다항식 (N -th cyclotomic polynomial)을 의미한다. ($f(x)$)란 $f(x)$ 로 생성되는 $\mathbb{Z}_q[x]$ 의 이데알(ideal)을 나타낸다. Euler totient 함수 $\Phi(N)$ 이란 N 과 서로소이고 N 보다 작은 자연수의 개수를 의미한다. $\Phi_N(x)$ 를 N 차 사이클로토믹 다항식으로 정의하면, 링은 다음과 같은 수학적식으로도 표현될 수 있다.

[94] [수학식 3]

$$[95] \quad R = \mathbb{Z}_q[x]/(\Phi_N(x))$$

[96] 다음으로, 제2 서버 장치(300)는 링으로부터 비밀키(sk)를 산출한다.

[97] [수학식 4]

$$[98] \quad sk \leftarrow (1, s(x)), s(x) \in R$$

[99] $s(x)$ 는 작은 계수로 랜덤하게 생성한 다항식을 의미한다.

[100] 제2 서버 장치(300)는 링으로부터 제1 랜덤 다항식($a(x)$)을 산출한다. 제1 랜덤 다항식은 다음과 같이 표현될 수 있다.

[101] [수학식 5]

$$[102] \quad a(x) \leftarrow R$$

[103] 또한, 제2 서버 장치(300)는 이산 가우시안 분포 또는 그와 통계적 거리가 가까운 분포로부터 에러를 추출한다. 에러는 다음과 같은 수학적식으로 표현될 수 있다.

[104] [수학식 6]

$$[105] \quad e(x) \leftarrow \mathcal{D}_{\alpha q}^n$$

[106] 에러까지 산출되면, 제2 서버 장치(300)는 제1 랜덤 다항식 및 비밀키에 에러를 모듈러 연산하여 제2 랜덤 다항식을 산출할 수 있다. 제2 랜덤 다항식은 다음과 같이 표현될 수 있다.

[107] [수학식 7]

$$[108] \quad b(x) = -a(x)s(x) + e(x) \pmod{q}$$

[109] 최종적으로 공개키(pk)는 제1 랜덤 다항식 및 제2 랜덤 다항식을 포함하는 형태로 다음과 같이 설정된다.

[110] [수학식 8]

$$[111] \quad pk = (b(x), a(x))$$

- [112] 제2 서버 장치(300)는 생성된 공개키를 각 단말 장치들(100-1 ~ 100-n)로 전송한다. 각 단말 장치들(100-1 ~ 100-n)은 수신된 공개키를 저장한 후, 암호화 시에 사용한다.
- [113] 상술한 예에서는 제2 서버 장치(300)가 공개키 및 비밀키를 생성하는 것으로 설명하였으나, 다른 예에서는 각 단말 장치(100-1 ~ 100-n) 또는 기타 장치가 공개키 및 비밀키 중 적어도 하나를 생성한 후, 타 장치와 공유하는 것도 가능하다. 또한, 상술한 키 생성 방법은 일 예에 불과하므로, 반드시 이에 한정되는 것은 아니며, 이 밖에 다른 방법으로 공개키 및 비밀키를 생성할 수도 있음은 물론이다.
- [114] 도 2는 본 발명의 일 실시 예에 따른 단말 장치(100) 및 서버 장치(200)의 구성을 나타내는 블록도이다.
- [115] 도 2에 따르면, 단말 장치(100)는 메모리(110), 프로세서(120), 통신부(130)를 포함한다.
- [116] 메모리(110)는 단말 장치(100)를 구동하기 위한 O/S나 각종 소프트웨어, 데이터 등을 저장하기 위한 구성요소이다. 메모리(110)는 RAM이나 ROM, 플래시 메모리, HDD, 외장 메모리, 메모리 카드 등과 같은 다양한 형태로 구현될 수 있으며, 어느 하나로 한정되는 것은 아니다.
- [117] 메모리(110)는 공개키, 스케일링 팩터 등을 저장할 수 있다.
- [118] 프로세서(120)는 메모리(110)에 저장된 프로그램을 실행시켜 각종 연산 작업을 수행하기 위한 구성요소이다.
- [119] 통신부(130)는 외부 장치들과 통신을 수행하기 위한 구성요소이다. 통신부(130)는 와이파이, 지그비, 블루투스, NFC 등과 같은 무선 통신뿐만 아니라, 각종 유선 통신 인터페이스를 통한 유선 통신을 지원하는 형태로도 구현될 수 있다.
- [120] 프로세서(120)는 암호화할 메시지에 스케일링 팩터를 반영한 후, 공개 키를 이용하여 암호화하여 동형 암호문을 생성한다. 프로세서(120)는 생성된 동형 암호문을 외부 장치로 전송하도록 통신부(130)를 제어할 수 있다.
- [121] 동형 암호문은 상술한 바와 같은 수학적 1과 같은 성질을 만족하도록 생성된다. 구체적으로, 프로세서(120)는 암호화할 메시지 M 에 스케일링 팩터 Δ 를 곱하여 정수 또는 정수 계수의 다항식인 ΔM 을 계산한다.
- [122] 그리고 나서, 다항식 v 를 랜덤하게 결정한다. 프로세서(120)는 v 를 작은 분포(예를 들어, 이산 가우시안 또는 이와 유사한 분포)로부터 산출할 수 있다. 프로세서(120)는 에러 e_0, e_1 도 작은 에러 분포(예를 들어, 이산 가우시안 또는 이와 유사한 분포)로부터 뽑을 수 있다.
- [123] 프로세서(120)는 공개키 $pk = (b(x), a(x))$ 와, 다음과 같은 수학적식을 이용하여 암호문을 생성한다.
- [124] [수학적식 9]

- [125]
$$C_{\text{txt}} = (v \cdot b(x) + \Delta \cdot M + e_0, v \cdot a(x) + e_1) \in R \times R$$
- [126] 암호화할 메시지는 외부 소스로부터 수신할 수도 있고, 단말 장치(100)에 직접 구비 또는 연결된 입력 장치로부터 입력될 수도 있다. 또한, 스케일링 팩터도 사용자가 직접 입력하거나, 다른 장치를 통해 제공받을 수도 있다. 예를 들어, 단말 장치(100)가 터치 스크린이나 키 패드를 포함하는 경우, 프로세서(120)는 사용자가 터치 스크린이나 키 패드를 통해 입력하는 데이터를 메모리(110)에 저장한 후, 암호화할 수 있다. 생성된 동형 암호문은 복호화하였을 때 메시지에 스케일링 팩터를 반영한 값에 에러를 더한 결과값으로 복원되는 형태가 될 수 있다. 스케일링 팩터는 사전에 입력되어 설정된 값을 그대로 사용할 수도 있다.
- [127] 상술한 암호문의 모듈러스 q 는 실시 예에 따라 상이하게 설정될 수 있다. 일 예로, 프로세서(120)는 암호문 모듈러스를 스케일링 팩터 Δ 의 지수승 $q = \Delta^L$ 형태로 설정할 수 있다. 다른 예로, 프로세서(120)는 암호문 모듈러스를 복수의 서로 다른 스케일링 팩터들을 곱한 값으로 설정될 수도 있다. 프로세서(120)는 설정한 암호문 모듈러스를 메모리(110)에 저장할 수도 있다. 상술한 바와 같이 각 팩터들은 유사 범위 이내에서 서로 소인 관계를 가질 수 있다.
- [128] 프로세서(120)는 공개 키의 암호화를 위하여 에러를 사용할 수도 있다. 에러는 이산 가우시안 분포 또는 이산 가우시안 분포와 통계적 거리가 가까운 분포로부터 산출할 수 있다. 일 예로, 상술한 수학식 6과 같은 형태의 에러를 산출할 수 있다. 프로세서(120)는 메시지가 입력되면 스케일링 팩터를 그 메시지에 승산한 값에 에러를 가산한 후, 공개 키를 이용하여 암호화할 수 있다. 이 경우, 암호문을 복호화하였을 때 산출되는 에러 값은 암호화 시에 가산했던 초기 에러와 상이하게 도출된다.
- [129] 또는, 메시지 및 스케일링 팩터를 승산한 상태에서 바로 공개 키를 이용하여 암호화할 수도 있다. 이 경우, 암호화 과정에서 산출되는 에러가 메시지 및 스케일링 팩터를 승산한 결과값에 가산될 수 있다.
- [130] 프로세서(120)는 암호문의 길이를 스케일링 팩터의 크기에 대응되도록 생성할 수 있다.
- [131] 한편, 본 발명의 일 실시 예에 따르면, 패킹(packing)이 이루어질 수도 있다. 동형 암호화에서 패킹을 이용하게 되면, 다수의 메시지를 하나의 암호문으로 암호화하는 것이 가능해진다. 이 경우, 제1 서버 장치(200)에서 각 암호문들 간의 연산을 수행하게 되면, 결과적으로 다수의 메시지에 대한 연산이 병렬적으로 처리되므로 연산 부담이 크게 줄어들게 된다.
- [132] 구체적으로는, 프로세서(120)는 메시지가 복수의 메시지 벡터로 이루어지는 경우, 복수의 메시지 벡터를 병렬적으로 암호화할 수 있는 형태의 다항식으로 변환한 후, 그 다항식에 스케일링 팩터를 승산하고 공개 키를 이용하여 동형 암호화할 수도 있다. 이에 따라, 복수의 메시지 벡터를 패킹한 암호문을 생성할 수 있다.

- [133] 구체적으로는, 프로세서(120)는 N차 사이클로토믹 다항식 (N-th cyclotomic polynomial) $\Phi_N(x)$ 가 복소수($\mathbb{C}$)내에서 서로 다른 $n = \phi(N)$ 개의 근 $\zeta_1, \overline{\zeta_1}, \dots, \zeta_{n/2}, \overline{\zeta_{n/2}}$ (primitive N-th roots of unity)을 가진다는 점을 이용한다. 복소수 개념을 도입함으로써 후술하는 바와 같이 복수 개의 메시지를 동시에 동형 암호화하는 것이 가능하다.

- [134] 다음으로 기본 매장함수(canonical embedding function)를 변형하여 패킹 함수(σ)를 계산한다. 기본 매장함수는 다항식 $M(x) \in \mathbb{R}[x]/(\Phi_N(x))$ 를 $\Phi_N(x)$ 의 근 $\zeta_1, \overline{\zeta_1}, \dots, \zeta_{n/2}, \overline{\zeta_{n/2}}$ 중에서 복소수 켤레(complex conjugate) 관계가 아닌 (n/2)개의 근 $\zeta_1, \dots, \zeta_{n/2}$ 에서의 값들의 쌍

$(M(\zeta_1), \dots, M(\zeta_{n/2})) \in \mathbb{C}^{n/2}$ 으로 대응시키는 함수이다. 이 함수가

동형함수(homomorphism)임은 이 분야에 평균적 지식을 가진 자라면 쉽게 증명할 수 있다.

- [135] 기본 매장함수를 행렬(C)로 표현하면 다음과 같다.

- [136] [수학식 10]

- [137]
$$\begin{pmatrix} 1 & \zeta_1 & \dots & \zeta_1^{n-1} \\ 1 & \zeta_2 & \dots & \zeta_2^{n-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \zeta_{n/2} & \dots & \zeta_{n/2}^{n-1} \end{pmatrix}$$

- [138] 다항식 $M(x)$ 를 계수들의 열벡터 $M = (M_0, \dots, M_{n-1})$ 로 표현하면, 이 다항식의 패킹 함수 $\sigma(M) = (M(\zeta_1), \dots, M(\zeta_{n/2}))$ 와는 $C \cdot M = \sigma(M)$ 관계, 즉, 다음과 같은 관계를 가지게 된다.

- [139] [수학식 11]

- [140]
$$\begin{pmatrix} 1 & \zeta_1 & \dots & \zeta_1^{n-1} \\ 1 & \zeta_2 & \dots & \zeta_2^{n-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \zeta_{n/2} & \dots & \zeta_{n/2}^{n-1} \end{pmatrix} \begin{pmatrix} M_0 \\ M_1 \\ \vdots \\ M_{n-1} \end{pmatrix} = \begin{pmatrix} M(\zeta_1) \\ M(\zeta_2) \\ \vdots \\ M(\zeta_{n/2}) \end{pmatrix}$$

- [141] 이와 같이 기본 매장 함수를 산출한 상태에서, 프로세서(120)는 복수 개(예를 들어, n/2개)의 메시지 벡터

가 입력되면,

$$\mathbf{m} = (m_1, \dots, m_{n/2}) \in \mathbb{C}^{n/2}$$

상술한 기본 매장 함수를 이용하여 메시지 벡터들을 다항식으로 변환한다.

[142] [수학식 12]

[143] $M(x) = \sigma^{-1}(\mathbf{m})$

[144] 수학식 12와 같은 방식으로 변환된 다항식 $M(x)$ 는 $M(\zeta_i) = m_i$ 와 같은 관계를 만족한다.

[145] 프로세서(120)는 상술한 방식으로 변환된 다항식에 스케일링 팩터를 승산한 후, 공개 키를 이용하여 동형 암호화할 수 있다. 일 예로, 동형 암호화 방법 중 하나인 Ring-LWE(Learning with Errors) 방식을 이용하여 동형 암호화할 수 있다.

[146] 구체적인 암호문은 다음과 같은 형태로 표현될 수 있다.

[147] [수학식 13]

[148] $C_{\text{txt}} = (v \cdot b(x) + \Delta \cdot M + e_0, v \cdot a(x) + e_1) \in R \times R$

[149] 상술한 바와 같이 단말 장치(100)가 생성한 동형 암호문은 제1 서버 장치(200)로 제공될 수 있다. 제1 서버 장치(200)는 단말 장치(100)로부터 수신되는 동형 암호문을 저장하여, 이후 연산 과정에 사용할 수 있다. 제1 서버 장치(200)는 복수의 서로 다른 단말 장치로부터 전송되는 여러 개의 동형 암호문 뿐만 아니라, 하나의 단말 장치(100)에서 전송되는 여러 개의 동형 암호문에 대해서도 연산 처리를 수행할 수 있다.

[150] 제1 서버 장치(200) 역시 메모리(210), 프로세서(220), 통신부(230)를 포함하는 형태로 구현될 수 있다.

[151] 통신부(230)는 단말 장치(100)로부터 각종 동형 암호문을 수신할 수 있다. 각 동형 암호문은 서로 다른 단말 장치 또는 하나의 단말 장치에서 개별적으로 암호화된 데이터를 의미한다.

[152] 프로세서(220)는 수신된 동형 암호문을 메모리(210)에 저장한다. 도 2에서는 메모리(210)만을 도시하였으나, 제1 서버 장치(200)는 대용량의 데이터를 저장하기 위한 별도의 스토리지를 더 포함할 수도 있고, 이와 별도로 외부 저장 장치와 연결되어 그 외부 저장 장치에 각 동형 암호문들을 저장할 수도 있다.

[153] 프로세서(220)는 복수의 동형 암호문에 대하여 기 설정된 연산을 수행할 수 있다. 수행되는 연산의 종류는 제1 서버 장치(200)의 관리자에 의해 직접 선택될 수도 있고, 제2 서버 장치(300)로부터 요청을 받을 수도 있다.

[154] 일 예로, 연산은 승산, 제분, 가산, 감산 등과 같은 기본 연산으로 설정될 수도 있지만, 반드시 이에 한정되는 것은 아니다. 구체적으로는, 암호화된 메시지가 복소수 형태인 경우, 켄쥬게이트(conjugate) 연산이 수행될 수도 있고, 이밖에 통계나 소팅 등의 연산도 이루어질 수 있다.

[155] 복수의 동형 암호문 각각이 복수의 메시지 벡터를 패키징한 암호문인 경우에는, 프로세서(220)는 복수의 암호문 각각에 포함된 다항식들의 암호문을 병렬적으로 연산 처리할 수도 있다. 이 경우, 복수의 암호문들의 연산 순서를

로테이션시키거나, 랜덤하게 섞어서 연산을 수행할 수도 있다. 제2 서버 장치(200)의 구체적인 연산 동작에 대해서는 후술하는 부분에서 구체적으로 다시 설명한다.

- [156] 도 2에서는 단말 장치(100) 및 제1 서버 장치(200)가 각각 통신부, 메모리, 프로세서만을 포함하는 형태인 것으로 도시하였으나, 반드시 이러한 구성으로 한정되는 것은 아니다. 즉, 각 장치의 종류 및 사용 환경 등에 따라 입력 수단, 디스플레이, 버스 등과 같은 다양한 구성요소들을 더 포함할 수도 있다. 일 예로, 단말 장치(100)는 전송할 메시지나 스케일링 팩터 등을 입력받기 위한 입력부(미도시)를 더 포함할 수도 있다. 프로세서(120)는 입력부를 통해 입력된 메시지 또는 스케일링 팩터 등을 메모리(110)에 저장할 수 있다. 또한, 프로세서(120)는 입력된 스케일링 팩터를 이용하여 상술한 다양한 방법으로 암호문의 모듈러스를 설정한 후, 메모리(110)에 저장할 수도 있다.
- [157] 도 3은 본 발명의 일 실시 예에 따른 단말 장치의 암호화 방법을 설명하기 위한 흐름도이다.
- [158] 도 3에 따르면, 단말 장치(100)는 스케일링 팩터를 설정한다(S310). 스케일링 팩터의 설정 방법은 상술한 바와 같이 다양하게 구현될 수 있다.
- [159] 단말 장치(100)는 설정된 스케일링 팩터를 저장한다. 단말 장치(100)는 암호화할 메시지가 발생하면, 그 메시지에 스케일링 팩터를 반영한 후, 공개 키를 이용하여 동형 암호화를 수행한다(S320). 공개 키는 단말 장치(100)가 직접 생성할 수도 있고, 외부 장치로부터 제공받을 수도 있다.
- [160] 도 4는 본 발명의 일 실시 예에 따른 서버 장치의 암호문 처리 방법을 설명하기 위한 흐름도이다. 도 1에 도시한 바와 같이, 암호문 처리는 제1 서버 장치에서 수행되므로, 이하에서는 제1 서버 장치(200)가 수행하는 방법으로 설명한다.
- [161] 도 4에 따르면, 제1 서버 장치(200)는 복수의 동형 암호문을 수신한다(S410). 각 동형 암호문은 스케일링 팩터가 적용된 상태에서 공개 키를 이용하여 동형 암호화한 데이터를 의미한다.
- [162] 제1 서버 장치(200)는 외부 장치 또는 관리자로부터 연산 요청이 입력되면(S420), 복수의 동형 암호문을 대상으로 요청된 연산을 수행한다(S430). 연산이 완료되면, 제1 서버 장치(200)는 연산 결과 데이터로부터 유효 영역의 데이터를 검출한다(S440). 제1 서버 장치(200)는 연산 결과 데이터를 라운딩 처리를 수행하여 유효 영역의 데이터를 검출할 수 있다. 라운딩 처리란 암호화된 상태에서 메시지의 반올림(round-off)을 진행하는 것을 의미하며, 다르게는 리스케일링(rescaling)이라고 할 수도 있다. 구체적으로는, 제1 서버 장치(300)는 암호문 각각의 성분에 스케일링 인수의 역수인 Δ^{-1} 을 곱하고 반올림 하여, 노이즈 영역을 제거한다. 노이즈 영역은 스케일링 팩터의 크기에 대응되도록 결정될 수 있다. 결과적으로 노이즈 영역이 제외된 유효 영역의 메시지를 검출할 수 있다. 암호화 상태에서 진행되므로 추가적인 에러가 발생하지만 크기는 충분히 작으므로 무시할 수 있다.

- [163] 도 5는 도 4의 암호문 처리 방법을 구체적으로 설명하기 위한 모식도이다.
- [164] 예를 들어, 제1 동형 암호문(51)과 제2 동형 암호문(52)을 대상으로, 곱셈 연산이 요청되었다면, 제1 서버 장치(200)는 암호문 상태에서 곱셈(S510)을 진행하여 연산 암호문(53)을 생성한다. 산출된 연산 암호문(53)은 스케일링 팩터가 반영된 메시지부($\Delta^2 M_1 M_2$)와 에러 데이터(emult)를 포함한다. 곱셈 과정에서 에러 데이터의 크기가 커지기는 하지만, 후술하는 바와 같이 메시지부의 크기가 축소되므로 문제가 발생하지 않는다. 이는 덧셈에 대해서도 동일하게 작용한다.
- [165] 다음으로 메시지부($\Delta^2 M_1 M_2$)를 유효 숫자에 해당하는 부분, 즉 유효 영역으로 크기를 축소시켜서 최종 연산 암호문(54)을 생성한다. 상술한 바와 같이, 제2 서버 장치(200)는 연산 암호문(53)에 대하여 스케일링 팩터의 역수를 곱한 후 라운딩 처리하는 방식으로, 노이즈 영역을 제거할 수 있다. 결과적으로 유효 영역에 해당하는 데이터($(\Delta M_1 M_2)' + e'$)를 포함하는 최종 연산 암호문(54)을 생성할 수 있다. 이러한 처리 과정에 의해 추가적인 에러가 발생하여 에러 데이터(emult)가 e' 로 변경되지만 크기는 충분히 작으므로 무시할 수 있다.
- [166] 도 6은 위에서 기술한 패킹 방법을 통해 복수의 메시지를 하나의 암호문에 암호화한 경우 연산을 수행하는 방법을 설명하기 위한 방법이다. 설명의 편의를 위하여 복수의 메시지 벡터를 암호화한 암호문을 패킹 암호문으로 명명한다. 도 6에 따르면, 제1 패킹 암호문(ct)는 복수의 메시지(61-1 ~ 61-k)로 이루어진 벡터를 암호화하고 있고 제2 패킹 암호문(ct')는 복수의 메시지(62-1 ~ 62-k)로 이루어진 벡터를 암호화한 암호문이다.
- [167] 제1 서버 장치(200)는 제1 패킹 암호문(ct)과 제2 패킹 암호문(ct')에 대한 연산이 요청되면 위에서 기술한 실시예 등의 방법으로 두 암호문의 연산을 수행한다. 결과적으로 한 번의 암호문 연산으로 여러 데이터의 병렬 처리가 가능한 SIMD (Single Instruction Multiple Data) 연산이 이루어질 수 있다.
- [168] 도 6에 따르면, 제1 패킹 암호문(ct)의 첫번째 메시지 성분(61-1)과 제2 패킹 암호문(ct')의 첫번째 메시지 성분(62-1)이 곱해지게 된다. 그와 동시에 두번째 메시지 성분들(61-2, 62-2)의 곱셈도 이루어진다. 결과적으로 최종 패킹 암호문(ct_{mult})은 같은 위치에 있는 메시지 성분들의 곱셈 값들로 이루어진 벡터(63-1 ~ 63-k)를 암호화한 암호문으로 이해할 수 있다.
- [169] 도 7은 SIMD 연산을 수행함에 있어서, 일부 동형 암호문의 위치를 변경하여 처리하는 방법을 설명하기 위한 도면이다. 도 7에 따르면, 제1 패킹 암호문(ct)을 이용하여 새로운 제2 패킹 암호문(ct')를 생성할 수 있다. 이때 제2 패킹 암호문의 첫번째 성분은 제1 패킹 암호문의 두번째 메시지 성분과 같아지도록 제1 패킹 암호문(ct)을 쉬프트시킨다. 그와 동시에 제2 패킹 암호문의 두 번째 성분은 제1 패킹 암호문의 세번째 메시지 성분과 같아진다. 결과적으로 제2 패킹 암호문은 제1 패킹 암호문의 모든 메시지 위치를 이동시킨 새로운 메시지 벡터를 암호화한 패킹 암호문이다. 본 명세서에서는 이상과 같이 암호문의 위치를 이동시키는

작업을 로테이션 처리라 한다.

- [170] 도 8은 도 1의 네트워크 시스템의 동작을 종합적으로 설명하기 위한 타이밍도이다. 도 8에서는, 두 개의 단말 장치(100-1, 100-2)에서 암호문을 전송하는 경우를 도시하였다.
- [171] 먼저, 단말 장치 1(100-1)에서 제1 메시지가 입력되면(S810), 단말 장치 1(100-1)은 상술한 방식으로 제1 동형 암호문을 생성한 후(S815), 제1 서버 장치(200)로 전송할 수 있다(S820). 일 예로, 단말 장치 1(100-1)의 사용자가 사진을 촬영한 경우에는 그 사진 및 촬영 장소 등이 암호화될 수도 있고, 사용자가 음악 콘텐츠를 재생하거나 특정 URL로 접속하였다면, 그 재생 이력이나 URL 접속 이력 등이 암호화될 수도 있다.
- [172] 단말 장치 2(100-2)도 유사한 형태로 암호문을 생성하여 제1 서버 장치(200)로 전송할 수 있다(S825, S830, S835).
- [173] 제1 서버 장치(200)는 전송된 암호문들을 암호화된 그 상태 그대로 저장한다. 이러한 상태에서 제2 서버 장치(300)가 특정 연산을 요청한다면(S840), 제1 서버 장치(200)는 암호문 상태로 연산을 수행한 후(S845), 제2 서버 장치(300)로 전송한다(S850). 제2 서버 장치(300)는 전송된 암호문을 복호화하여 메시지를 검출한다.
- [174] 예를 들어, 수학적 식 9와 같은 형태로 암호화된 암호문을 제2 서버 장치(300)가 비밀키 $sk=(1, s)$ 로 복호화한다면, 그 결과는 다음과 같은 수학적 식으로 표현될 수 있다.
- [175] [수학적 식 14]
- [176] $\langle C_{txt}, sk \rangle = (v \cdot b(x) + \Delta \cdot M + e_0) + s(x)(v \cdot a(x) + e_1)$
- [177] $= v(b(x) + a(x)s(x)) + \Delta \cdot \Delta + e_0 + s(x) \cdot e_1$
- [178] $= \Delta \cdot M + (v \cdot e + e_0 + s(x) \cdot e_1)$
- [179] $= \Delta \cdot M + e_{small}$
- [180] 결과적으로 복호화된 데이터는 스케일링 팩터 및 메시지를 곱한 값에 작은 크기의 에러 e_{small} 를 더한 값을 포함하며, 에러는 LSB 측에 배치되고 ΔM 은 에러에 인접한 위치에 배치되게 된다.
- [181] 구체적인 예를 들어, 제1 메시지가 1.23이고 제2 메시지가 3.45이며, 스케일링 팩터 $\Delta=10^3$ 이라고 가정하면, $\Delta M_1 + e_1 \approx 1230$, $\Delta M_2 + e_2 \approx 3450$ 이 된다. 제1 서버 장치(200)에서 곱하기 연산을 수행하면, 데이터는 $\Delta^2 M_1 M_2 + e_{total} \approx 4243500$ 이 된다. 이 결과값에 Δ^{-1} 을 승산한 후 라운딩 처리하면 4243이 된다. 제2 서버 장치(200)는 복호화에 의해 이러한 결과값을 얻을 수 있다.
- [182] 제2 서버 장치(300)는 복호화된 메시지를 출력할 수 있다(S860). 예를 들어, 단말 장치 1, 2(100-1, 100-2)가 각각 사진을 제1 서버 장치(200)로 전송한 상태라면, 제2 서버 장치(300)는 각 장치(100-1, 100-2)에서 촬영된 사진을 한꺼번에 출력하여 줄 수 있다.
- [183] 상술한 실시 예에서는 제1 서버 장치(200)에서 암호문들 간의 연산을 수행하고,

노이즈 영역을 제거한 후 제2 서버 장치(300)로 전송하는 것으로 도시 및 설명하였으나 반드시 이에 한정되는 것은 아니다. 예를 들어, 제2 서버 장치(300)가 스케일링 팩터에 대한 정보를 알고 있다면, 직접 노이즈 영역을 제거하여 유효 영역을 검출한 후 복호화를 진행할 수도 있다.

- [184] 한편, 이상과 같은 암호화 방법 및 암호문 처리 방법은 실수 평면에 대해 적용될 수 있다.
- [185] 도 9는 본 개시의 일 실시 예에 따른 네트워크 시스템의 구성을 나타내는 도면이다.
- [186] 도 9를 참조하면, 네트워크 시스템은 복수의 단말 장치(100-1 ~ 100-n), 제1 서버 장치(200), 제2 서버 장치(300)를 포함할 수 있으며, 각 구성들은 네트워크(10)를 통해 서로 연결될 수 있다.
- [187] 네트워크(10)는 다양한 형태의 유무선 통신 네트워크, 방송 통신 네트워크, 광통신 네트워크, 클라우드 네트워크 등으로 구현될 수 있으며, 각 장치들은 별도의 매개체 없이 와이파이, 블루투스, NFC(Near Field Communication) 등과 같은 방식으로 연결될 수도 있다.
- [188] 도 9에서는 단말 장치가 복수 개(100-1 ~ 100-n)인 것으로 도시하였으나, 반드시 복수 개의 단말 장치가 사용되어야 하는 것은 아니며 하나의 장치가 사용될 수도 있다. 일 예로, 단말 장치(100-1 ~ 100-n)는 스마트폰, 태블릿, 게임 플레이어, PC, 랩톱 PC, 홈서버, 키오스크 등과 같은 다양한 형태의 장치로 구현될 수 있으며, 이밖에 IoT 기능이 적용된 가전 제품 형태로도 구현될 수 있다.
- [189] 사용자는 자신이 사용하는 단말 장치(100-1 ~ 100-n)를 통해서 다양한 정보를 입력할 수 있다. 입력된 정보는 단말 장치(100-1 ~ 100-n) 자체에 저장될 수도 있지만, 저장 용량 및 보안 등을 이유로 외부 장치로 전송되어 저장될 수도 있다. 도 1에서 제1 서버 장치(200)는 이러한 정보들을 저장하는 역할을 수행하고, 제2 서버 장치(300)는 제1 서버 장치(200)에 저장된 정보의 일부 또는 전부를 이용하는 역할을 수행할 수 있다.
- [190] 각 단말 장치(100-1 ~ 100-n)는 입력된 정보를 동형 암호화하여, 동형 암호문을 제1 서버 장치(200)로 전송할 수 있다. 제1 서버 장치(200)는 전송된 동형 암호문을 복호화하지 않고, 암호문 상태로 저장할 수 있다.
- [191] 제2 서버 장치(300)는 동형 암호문에 대한 특정 처리 결과를 제1 서버 장치(200)로 요청할 수 있다. 제1 서버 장치(200)는 요청에 따라 특정 연산을 수행한 후, 그 결과를 제2 서버 장치(300)로 전송한다. 일 예로, 두 개의 단말 장치(100-1, 100-2)가 전송한 암호문 ct_1 , ct_2 가 제1 서버 장치(200)에 저장된 경우, 제2 서버 장치(300)는 두 단말 장치(100-1, 100-2)로부터 제공된 정보들을 합산한 값을 제1 서버 장치(200)로 요청할 수 있다. 제1 서버 장치(200)는 요청에 따라 두 암호문을 합산하는 연산을 수행한 후, 그 결과값($ct_1 + ct_2$)을 제2 서버 장치(300)로 전송할 수 있다.
- [192] 각 단말 장치(100-1 ~ 100-n)는 동형 암호화를 수행하는 과정에서 산출되는

암호화 노이즈, 즉, 에러를 암호문에 포함시킬 수 있다. 구체적으로는, 각 단말 장치(100-1 ~ 100-n)에서 생성하는 동형 암호문은, 추후에 비밀키를 이용하여 복호화하였을 때 메시지 및 에러 값을 포함하는 결과값이 복원되는 형태로 생성될 수 있다.

- [193] 일 예로, 단말 장치(100-1 ~ 100-n)에서 생성하는 동형 암호문은 비밀키를 이용하여 복호화 하였을 때 다음과 같은 성질을 만족하는 형태로 생성될 수 있다.
- [194] [수학식 1]
- [195] $\text{Dec}(ct, sk) = \langle ct, sk \rangle = \Delta M + e \pmod{q}$
- [196] 여기서 $\langle \cdot, \cdot \rangle$ 는 내적 연산(usual inner product), ct 는 암호문, sk 는 비밀키, M 은 평문 메시지, e 는 암호화 에러 값, Δ 는 스케일링 팩터(scaling factor), 그리고 q 는 암호문의 모듈러스(Modulus)를 의미한다. q 는 스케일링 팩터가 메시지에 곱해진 결과값 ΔM 보다 크게 선택되어야 한다. 에러 값 e 의 절대값이 ΔM 에 비해서 충분히 작다면, 암호문의 복호화 값 $\Delta M + e$ 는 유효숫자연산에서 원래의 메시지를 동일한 정밀도로 대체할 수 있는 값이다. 복호화된 데이터 중에서 에러는 최하위 비트(LSB) 측에 배치되고, ΔM 은 차하위 비트 측에 배치된다.
- [197] 메시지의 크기가 너무 작거나 너무 큰 경우, 스케일링 팩터를 이용하여 그 크기를 조절할 수도 있다. 스케일링 팩터를 사용하게 되면, 정수 형태의 메시지 뿐만 아니라 실수 형태의 메시지까지도 암호화할 수 있게 되므로, 활용성이 크게 증대할 수 있다. 또한, 스케일링 팩터를 이용하여 메시지의 크기를 조절함으로써, 연산이 이루어지고 난 이후의 암호문에서 메시지들이 존재하는 영역, 즉, 유효 영역의 크기도 조절될 수 있다.
- [198] 실시 예에 따라, 암호문 모듈러스 q 는 다양한 형태로 설정되어 사용될 수 있다. 일 예로, 암호문의 모듈러스는 스케일링 팩터 Δ 의 지수승 $q = \Delta^L$ 형태로 설정될 수 있다. Δ 가 2라면, $q = 2^{10}$ 과 같은 값으로 설정될 수 있다.
- [199] 다른 예로, 암호문 모듈러스는 서로소인 복수의 정수 값으로 설정될 수도 있다. 구체적으로, $q = q_1 q_2 q_3 \dots q_x$ 로 설정되며, $q_1, q_2, q_3, \dots, q_x$ 각각은 스케일링 인수 Δ 와 비슷한 크기이며 서로 소 관계의 값으로 설정된다. 스케일링 팩터를 이와 같은 방식으로 설정하게 되면, CRT(Chinese Remainder Theorem)에 따라 전체 연산을 복수 개의 모듈러스 연산으로 분리하여 진행할 수 있게 되므로, 연산 부담을 경감시킬 수 있다. 서로소인 복수의 정수 값을 모듈러스로 이용하는 방식을 RNS HEAAN 방식으로 지칭하며, RNS HEAAN 방식에 대해서는 도 12 내지 16을 참조하여 보다 자세히 설명한다.
- [200] 이상과 같이, 각 단말 장치(100-1 ~ 100-n)에서 생성한 암호문은 제1 서버 장치(200)에 의해 처리된 후, 암호문 상태로 제2 서버 장치(300)로 전송된다. 제2 서버 장치(300)는 전송된 암호문을 비밀키를 이용하여 복호화할 수 있다. 동형 암호화 처리가 되었기 때문에, 복호화된 데이터는 메시지 자체를 복호화한 후 연산 처리한 결과와 같은 결과가 될 수 있다. 결과적으로 중간 과정에서

제3자에게 유출될 위험을 방지할 수 있게 된다.

- [201] 도 9의 실시 예에서 암호화를 수행하는데 사용되는 공개키는 각 단말 장치에 의해 생성될 수도 있고, 복호화를 수행하는 제2 서버 장치(300)에서 생성한 후 각 단말 장치로 배포할 수도 있다. 이하에서는 설명을 용이하게 하기 위하여, 단말 장치(100) 측에서 각종 키를 생성하는 것으로 설명한다.
- [202] 키 생성 과정에서는 Ring-LWE 기법이 사용될 수 있다.
- [203] 단말 장치(100)는 키 생성 및 링 설정을 위한 각종 파라미터를 입력받을 수 있다. 구체적으로, 평문 메시지 비트의 길이, 공개키 및 비밀키의 크기 등과 같은 다양한 파라미터들을 설정할 수 있다. 그리고 단말 장치(100)는 입력받는 파라미터를 기초로 비밀키 및 공개키 등의 생성에 필요한 설정 파라미터를 생성할 수 있다.
- [204] 링은 다음과 같은 수학식으로 표현될 수 있다.
- [205] [수학식 2]
- [206]
$$R = \mathbb{Z}_q[x]/(f(x))$$
- [207] 여기서 R은 링, $\mathbb{Z}_q$ 는 계수, $f(x)$ 는 n차 다항식이다.
- [208] 링(Ring)이란 기 설정된 계수를 가지는 다항식의 집합으로, 원소들 사이에 덧셈과 곱셈이 정의되어 있으며 덧셈과 곱셈에 대해서 닫혀 있는 집합을 의미한다. 이러한 링은 환으로 지칭될 수 있다.
- [209] 한편, 상술한 수학식 2에서 n이 $\Phi(N)$ 일 때, N차 사이클로토믹 다항식 (N-th cyclotomic polynomial)을 의미한다. ($f(x)$)란 $f(x)$ 로 생성되는 $\mathbb{Z}_q[x]$ 의 이데알(ideal)을 나타낸다. Euler totient 함수 $\Phi(N)$ 이란 N과 서로소이고 N보다 작은 자연수의 개수를 의미한다. $\Phi_N(x)$ 를 N차 사이클로토믹 다항식으로 정의하면, 링은 다음과 같은 수학식으로도 표현될 수 있다. 그리고 상술한 다항식을 상반 방정식을 이용하는 경우, 링은 수학식 15와 같이 표현될 수 있다.
- [210] [수학식 3]
- [211]
$$R = \mathbb{Z}_q[x]/(\Phi_N(x))$$
- [212] [수학식 15]
- [213]
$$R = \mathbb{Z}[X] / (X^n + 1)$$
- [214] 한편, 상술한 수학식 3 및 15의 링은 평문 공간에서 복소수를 갖는다. 한편, 본 개시에서는 동형 암호문에 대한 연산 속도를 향상하기 위하여, 상술한 링의 집합 중 평문 공간이 실수인 집합만을 이용할 수 있다. 여기서, 평문 공간이 실수인 링을 이하에서는 R'로 지칭하며, R'는 다음과 같은 수학식으로 표현될 수 있다. R'을 이용하는 방식을 Real HEAAN 방식이라고 지칭한다.
- [215] [수학식 16]
- [216]
$$R' = \{a(X) \in \mathbb{Z}[X] / (X^n + 1) : a(X) = a(X^{-1})\} \subset R \text{ where } X^{-1} := -X^{n-1}$$
- [217] 이와 같은 링이 설정되면, 단말 장치(100)는 링으로부터 비밀키(sk)를 산출할 수

있다.

[218] [수학식 17]

[219] $sk \leftarrow (1, s)$

[220] 여기서, S는 작은 계수로 랜덤하게 생성된 값을 의미한다.

[221] 그리고 단말 장치(100)는 링으로부터 제1 랜덤 다항식(a)을 산출한다. 제1 랜덤 다항식은 다음과 같이 표현될 수 있다.

[222] [수학식 18]

[223] $a \leftarrow U(R'_{q_L})$

[224] 또한, 단말 장치(100)는 에러를 산출할 수 있다. 구체적으로, 단말 장치(100)는 이산 가우시안 분포 또는 그와 통계적 거리가 가까운 분포로부터 에러를 추출할 수 있다. 이러한 에러는 다음과 같이 표현될 수 있다.

[225] [수학식 19]

[226] $e \leftarrow X_{err}$

[227] 에러까지 산출되면, 단말 장치(100)는 제1 랜덤 다항식 및 비밀키에 에러를 모듈러 연산하여 제2 랜덤 다항식을 산출할 수 있다. 제2 랜덤 다항식은 다음과 같이 표현될 수 있다.

[228] [수학식 20]

[229] $b \leftarrow -as + e \pmod{q_L}$

[230] 최종적으로 공개키(pk)는 제1 랜덤 다항식 및 제2 랜덤 다항식을 포함하는 형태로 다음과 같이 설정될 수 있다.

[231] [수학식 21]

[232] $pk \leftarrow (b, a) \in R'^2_{q_L}$

[233] 단말 장치(100)는 생성된 공개키를 다른 단말 장치들(100-2 ~ 100-n) 및/또는 서버 장치(200)로 전송할 수 있다.

[234] 상술한 예에서는 단말 장치(100)가 공개키 및 비밀키를 생성하는 것으로 설명하였으나, 다른 예에서는 서버 장치(200) 또는 기타 장치가 공개키 및 비밀키 중 적어도 하나를 생성한 후, 타 장치와 공유하는 것도 가능하다. 또한, 상술한 키 생성 방법은 일 예에 불과하므로, 반드시 이에 한정되는 것은 아니며, 이 밖에 다른 방법으로 공개키 및 비밀키를 생성할 수도 있음은 물론이다.

[235] 보다 구체적인 공개키 및 비밀키 생성동작에 대해서는 도 12를 참조하여 후술한다.

[236]

[237] 도 10은 본 개시의 일 실시 예에 따른 단말 장치의 간단한 구성을 도시한 블록도이다.

[238] 도 10을 참조하면, 단말 장치(100)는 메모리(110) 및 프로세서(120)로 구성될 수 있다.

- [239] 메모리(110)는 단말 장치(100)를 구동하기 위한 O/S나 각종 소프트웨어, 데이터 등을 저장하기 위한 구성요소이다. 메모리(110)는 RAM이나 ROM, 플래시 메모리, HDD, 외장 메모리, 메모리 카드 등과 같은 다양한 형태로 구현될 수 있으며, 어느 하나로 한정되는 것은 아니다.
- [240] 메모리(110)는 암호화할 메시지를 저장한다. 여기서 메시지는 사용자가 각종 인용한 각종 신용 정보, 개인 정보 등일 수 있으며, 단말 장치(100)에서 사용되는 위치 정보, 인터넷 사용 시간 정보 등 사용 이력 등과 관련된 정보일 수도 있다.
- [241] 그리고 메모리(110)는 공개키를 저장할 수 있으며, 단말 장치(100)가 직접 공개키를 생성한 장치인 경우, 비밀키뿐만 아니라, 공개키 및 비밀키 생성에 필요한 각종 파라미터를 저장할 수 있다.
- [242] 또한, 메모리(110)는 후술한 과정에서 생성된 동형 암호문을 저장할 수 있다.
- [243] 프로세서(120)는 단말 장치(100) 내의 각 구성을 제어한다. 이러한 프로세서(130)는 CPU와 같은 단일 장치로 구성될 수 있으며, 클럭 발생 회로, CPU, 그래픽 프로세서 등의 복수의 장치로 구성될 수도 있다.
- [244] 그리고 프로세서(120)는 메모리(110)에 저장된 프로그램을 실행시켜 동형 암호문 생성에 필요한 비밀키 및 공개키를 생성할 수 있다. 이를 위하여, 우선적으로 프로세서(120)는 기설정된 계수를 가지는 다항식의 집합인 링을 설정할 수 있다. 이때, 링은 상술한 수학적 15 또는 수학적 16을 만족할 수 있다.
- [245] 그리고 프로세서(120)는 링으로부터 비밀키를 산출할 수 있다. 구체적으로, 프로세서(120)는 아래와 같은 수학식에 기초하여 링으로부터 비밀키(SK)를 산출할 수 있다.
- [246] 그리고 프로세서(120)는 링으로부터 제1 랜덤 다항식(a)을 산출할 수 있으며, 이산 가우시안 분포 또는 그와 통계적 거리가 가까운 분포로부터 에러를 추출할 수 있다.
- [247] 또한, 프로세서(120)는 제1 랜덤 다항식 및 비밀키에 에러를 모듈러 연산하여 제2 랜덤 다항식을 산출하고, 최종적으로 제1 랜덤 다항식 및 제2 랜덤 다항식을 포함하는 형태로 공개키(pk)를 생성할 수 있다. 한편, 단말 장치(100)가 RNS HEAAN을 지원하는 경우, 프로세서(120)는 서로소인 복수의 정수 각각에 대응되는 복수의 공개키를 생성할 수 있다.
- [248] 그리고 프로세서(120)는 메시지에 대한 동형 암호문을 생성할 수 있다. 구체적으로, 프로세서(120)는 메시지에 대해서 앞서 생성된 공개키를 적용하여 동형 암호문을 생성할 수 있다.
- [249] 한편, 단말 장치(100)가 RNS HEAAN을 지원하는 경우, 프로세서(120)는 메시지에 서로소인 복수의 정수 각각에 대응되는 복수의 공개키를 이용하여 복수의 기저(basis)로 표현되는 동형 암호문을 생성할 수 있다.
- [250] 또한, 프로세서(120)는 암호화할 메시지를 링에 속하는 다항식으로 산출하고, 산출된 다항식에 공개키를 적용하여 동형 암호문을 생성할 수 있다. 암호화할 메시지는 외부 소스로부터 수신할 수도 있고, 단말 장치(100)에 직접 구비 또는

연결된 입력 장치로부터 입력될 수도 있다. 또한, 스케일링 팩터도 사용자가 직접 입력하거나, 다른 장치를 통해 제공받을 수도 있다.

- [251] 상술한 암호문의 모듈러스 q 는 실시 예에 따라 상이하게 설정될 수 있다. 일 예로, 프로세서(120)는 암호문 모듈러스를 스케일링 팩터 Δ 의 지수승 $q=\Delta^L$ 형태로 설정할 수 있다. 다른 예로, 프로세서(120)는 서로소인 복수의 정수 값으로 설정되는 값이 이용될 수 있다.
- [252] 또한, 프로세서(120)는 암호문의 길이를 스케일링 팩터의 크기에 대응되도록 생성할 수 있다.
- [253] 한편, 본 발명의 일 실시 예에 따르면, 패킹(packing)이 이루어질 수도 있다. 동형 암호화에서 패킹을 이용하게 되면, 다수의 메시지를 하나의 암호문으로 암호화하는 것이 가능해진다. 이 경우, 제1 서버 장치(200)에서 각 암호문들 간의 연산을 수행하게 되면, 결과적으로 다수의 메시지에 대한 연산이 병렬적으로 처리되므로 연산 부담이 크게 줄어들게 된다.
- [254] 구체적으로는, 프로세서(120)는 메시지가 복수의 메시지 벡터로 이루어지는 경우, 복수의 메시지 벡터를 병렬적으로 암호화할 수 있는 형태의 다항식으로 변환한 후, 그 다항식에 스케일링 팩터를 승산하고 공개키를 이용하여 동형 암호화할 수도 있다. 이에 따라, 복수의 메시지 벡터를 패킹한 암호문을 생성할 수 있다.
- [255] 프로세서(120)는 암호문에 대한 연산을 수행할 수 있다. 구체적으로, 프로세서(120)는 동형 암호문에 대해서 암호화된 상태를 유지한 상태에서 덧셈 또는 곱셈 등의 연산을 수행할 수 있다. 구체적으로, 프로세서(120)는 연산에 사용될 동형 암호문 각각을 제1함수 처리하고, 제1함수 처리된 동형 암호문 간에 덧셈 또는 곱셈 등의 연산을 수행하고, 연산 수행된 동형 암호문을 제1 함수에 역함수인 제2 함수 처리할 수 있다. 여기서 제1 함수는 NTT(Number Theoretic Transform) 함수일 수 있다.
- [256] 한편, 동형 암호문을 상술한 RNS 방식으로 생성한 경우, 프로세서(120)는 생성된 동형 암호문 내의 기저별도 덧셈 및 곱셈을 수행할 수 있다.
- [257] 한편, 단말 장치(100)는 연산이 완료되면, 연산 결과 데이터로부터 유효 영역의 데이터를 검출할 수 있다. 구체적으로, 단말 장치(100)는 연산 결과 데이터를 라운딩 처리를 수행하여 유효 영역의 데이터를 검출할 수 있다. 라운딩 처리란 암호화된 상태에서 메시지의 반올림(round-off)을 진행하는 것을 의미하며, 다르게는 리스케일링(rescaling)이라고 할 수도 있다. 구체적으로는, 단말 장치(100)는 암호문 각각의 성분에 스케일링 인수의 역수인 Δ^{-1} 을 곱하고 반올림하여, 노이즈 영역을 제거한다. 노이즈 영역은 스케일링 팩터의 크기에 대응되도록 결정될 수 있다. 결과적으로 노이즈 영역이 제외된 유효 영역의 메시지를 검출할 수 있다. 암호화 상태에서 진행되므로 추가적인 에러가 발생하지만 크기는 충분히 작으므로 무시할 수 있다.
- [258] 만약, 단말 장치(100)가 RNS HEAAN을 지원하는 경우, 프로세서(120)는

복수의 기저 중 어느 하나의 비중이 임계치를 초과하면, 생성된 동형 암호문 내의 복수의 기저 각각에 대한 메시지의 반올림 처리를 수행하여 동형 암호문을 리스케일링할 수 있다.

- [259] 또한, 단말 장치(100)는 연산 결과 암호문 내의 근사 메시지 비중이 임계치를 초과하면, 연산 결과 암호문의 평문 공간을 확장할 수 있다. 구체적으로, 상술한 수식 1에서 q 가 M 보다 작다면 $M+e(\bmod q)$ 는 $M+e$ 와 다른 값을 가지므로 복호화가 불가능해진다. 따라서, q 값은 항상 M 보다 크게 유지되어야 한다. 하지만, 연산이 진행됨에 따라 q 값은 점차 감소하게 된다. 평문 공간의 확장이란 암호문 ct 를 더 큰 모듈러스(modulus)를 가지는 암호문으로 변화시키는 것을 의미한다. 평문 공간을 확장하는 동작은 다르게는 재부팅(rebooting)이라 할 수도 있다. 재부팅을 수행함에 따라, 암호문은 다시 연산이 가능한 상태가 될 수 있다.
- [260] 한편, 이상에서는 단말 장치(100)를 구성하는 간단한 구성에 대해서만 도시하고 설명하였지만, 구현시에는 다양한 구성이 추가로 구비될 수 있다. 이에 대해서는 도 11을 참조하여 이하에서 설명한다.
- [261]
- [262] 도 11은 본 개시의 일 실시 예에 따른 단말 장치의 구체적인 구성을 도시한 블록도이다.
- [263] 도 11을 참조하면, 본 개시의 단말 장치(100)는 메모리(110), 프로세서(120), 통신부(130), 디스플레이(140) 및 조작 입력 장치(150)로 구성될 수 있다.
- [264] 메모리(110)에 대해서는 도 10과 관련하여 설명하였는바, 중복 설명은 생략한다. 그리고 프로세서(120)에 대해서도 도 10과 관련하여 설명하였는바, 도 10에서 설명한 내용은 중복 기재하지 않고, 도 11에 추가된 구성과 관련된 내용만 이하에서 설명한다.
- [265] 통신부(130)는 단말 장치(100)를 외부 장치(미도시)와 연결하기 위해 형성되고, 근거리 통신망(LAN: Local Area Network) 및 인터넷망을 통해 외부 장치에 접속되는 형태뿐만 아니라, USB(Universal Serial Bus) 포트 또는 무선 통신(예를 들어, WiFi 802.11a/b/g/n, NFC, Bluetooth) 포트를 통하여 접속되는 형태도 가능하다. 이러한 통신부(130)는 송수신부(transceiver)로 지칭될 수도 있다.
- [266] 통신부(130)는 공개키를 외부 장치로부터 수신할 수 있으며, 단말 장치(100)에서 생성한 공개키를 외부 장치로 전송할 수 있다.
- [267] 그리고 통신부(130)는 외부 장치로부터 메시지를 수신할 수 있으며, 생성한 동형 암호문을 외부 장치로 송신할 수 있다.
- [268] 또한, 통신부(130)는 암호문 생성에 필요한 각종 파라미터를 외부 장치로부터 수신할 수 있다. 한편, 구현시에 각종 파라미터는 후술하는 조작 입력 장치(150)를 통하여 사용자로부터 직접 입력받을 수 있다.
- [269] 디스플레이(140)는 단말 장치(100)가 지원하는 기능을 선택받기 위한 사용자 인터페이스 창을 표시한다. 구체적으로, 디스플레이(140)는 단말 장치(100)가 제공하는 각종 기능을 선택받기 위한 사용자 인터페이스 창을 표시할 수 있다.

이러한 디스플레이(140)는 LCD, CRT, OLED 등과 같은 모니터일 수 있으며, 후술할 조작 입력 장치(150)의 기능을 동시에 수행할 수 있는 터치 스크린으로 구현될 수도 있다.

- [270] 디스플레이(140)는 비밀키 및 공개키 생성에 필요한 파라미터의 입력을 요청하는 메시지를 표시할 수 있다. 그리고 디스플레이(140)는 암호화 대상이 메시지를 선택하는 메시지를 표시할 수 있다. 한편, 구현시에 암호화 대상은 사용자가 직접 선택할 수도 있고, 자동으로 선택될 수 있다. 즉, 암호화가 필요한 개인 정보 등은 사용자가 직접 메시지를 선택하지 않더라도 자동으로 설정될 수 있다.
- [271] 조작 입력 장치(150)는 사용자로부터 단말 장치(100)의 기능 선택 및 해당 기능에 대한 제어 명령을 입력받을 수 있다. 구체적으로, 조작 입력 장치(150)는 사용자로부터 비밀키 및 공개키 생성에 필요한 파라미터를 입력받을 수 있다. 또한, 조작 입력 장치(150)는 사용자로부터 암호화될 메시지를 설정받을 수 있다.
- [272] 프로세서(120)는 사용자로부터 비밀키 및 공개키 생성에 필요한 파라미터를 입력받으면, 입력받은 파라미터에 기초하여 설정 파라미터를 생성하고, 생성된 설정 파라미터에 기초하여 비밀키 및 공개키를 생성할 수 있다.
- [273] 그리고 메시지에 대한 암호문 생성이 필요하면 프로세서(120)는 메시지에 공개키를 적용하여 동형 암호문을 생성할 수 있다. 구체적으로, 프로세서(120)는 메시지를 다항식 형태로 변환하고, 변환된 다항식 형태의 메시지에 공개키를 적용하여 동형 암호문을 생성할 수 있다.
- [274] 그리고 프로세서(120)는 동형 암호문에 대한 복호가 필요한 경우, 동형 암호문에 비밀키를 적용하여 다항식 형태의 복호문을 생성하고, 다항식 형태의 복호문을 디코딩하여 메시지를 생성할 수 있다. 이때 생성한 메시지는 앞서 설명한 수학적 식 1에서 언급한 바와 같이 에러를 포함할 수 있다.
- [275] 또한, 프로세서(120)는 동형 암호문에 대한 연산이 필요한 경우, 사용자가 요청한 복수의 동형 암호문에 대해서 덧셈 또는 곱셈 연산을 수행할 수 있다. 구체적인 연산 동작에 대해서는 도 15 및 도 16을 참조하여 후술한다.
- [276] 이상과 같이 본 실시 예에 따른 단말 장치(100)는 메시지에 동형 암호문을 생성할 수 있는바, 연산이 필요한 경우에도 메시지의 안정성을 향상할 수 있다. 또한, 생성되는 동형 암호문은 에러를 포함하는바, 고도의 보안성이 요구되는 생체 정보 등에 대해서도 안정적인 보안을 유지할 수 있다. 또한, 본 실시 예에 따른 단말 장치(100)는 평문 공간이 실수인 링을 이용하는바 연산 속도를 향상할 수 있다. 즉, 기존의 평문 공간이 복소수인 링을 이용하는 경우보다 2배 많은 병렬 실수 연산이 가능하게 된다.
- [277]
- [278] 이하에서는 REAL HEAAN 방식에 알고리즘을 우선 설명한다.
- [279] 암호문 간의 연산은 R'_q 원소들 간의 덧셈, 곱셈 연산이기 때문에, R'_q 위에서의

연산 효율성을 증대시키는 것이 필요하다. R'_q 위에 곱셈을 가장

나이브(naive)하게 수행하면, $O(n^2)$ 의 복잡도(complexity)을 갖는다.

[280] 이러한 연산의 복잡도를 줄이기 위하여 본 개시에서는 number Theoretic Transform(NTT) 기법을 이용한다. NTT_m 이란 $Z_q[X]/(X^m-1)$ 에서 Z_q^m 로 가는 필드 동형(field isomorphism)이고, 이의 역함수를 $INTT_m$ 이라 지칭한다.

[281] 이러한 NTT_m , $INTT_m$ 은 페스트 푸리에 트랜스폼(Fast Fourier Transform) 알고리즘 적용시 $O(m \log m)$ 의 복잡도를 갖는다.

[282] 즉, $a(X), b(X) \in Z_q[X]/(X^m-1)$ 에 대해 $a(X) \cdot b(X)$ 을 계산하고 싶은 경우 $INTT_m(NTT_m(a(X)) \odot NTT_m(b(X)))$ 으로 연산 수행하면 $O(m \log m)$ 의 복잡도를 갖는다.

[283] 이하에서는 상술한 기법을 링에 적용하는 방식을 설명한다.

[284] 링에 속하는 원소 각각은 $a(X) \in R'_q$ 을 만족하고,

$$a_0, a_1, \dots, a_{\frac{n}{2}-1} \in \mathbb{Z}_q \text{ 일 때, 원소는 아래와 같은 수학식 22와 같이}$$

표현될 수 있다.

[285] [수학식 22]

$$[286] \quad a(X) = a_0 + \sum_{i=1}^{\frac{n}{2}-1} a_i(X^i - X^{n-i})$$

[287] 먼저, R' 에 대한 $NTT_m(NTT'_{n/2})$ 과 관련하여, $\Phi(a(X))$ 을 계산하고, function Φ 는 다음의 수학식 23과 같이 표현될 수 있다.

[288] [수학식 23]

$$[289] \quad \Phi: R'_q \rightarrow \mathbb{Z}_q^{n/2}$$

$$[290] \quad a(X) \rightarrow (a_0, a_1 \cdot w_m + a_{\frac{n}{2}-1} \cdot w_m^{1-\frac{n}{2}}, \dots, a_{\frac{n}{2}-1} \cdot w_m^{\frac{n}{2}-1} + a_1 \cdot w_m^{-1})$$

[291] 그리고 R' 에 대한 $INTT_m(INTT'_{n/2})$ 과 관련하여, $\Phi^{-1}(INTT_{n/2}(a))$ 을 계산하고, 수학식 24와 같은 값을 출력할 수 있다.

[292] [수학식 24]

$$[293] \quad \begin{aligned} \Phi^{-1}: \mathbb{Z}_q^{n/2} &\rightarrow R'_q \\ (a_0, \dots, a_{\frac{n}{2}-1}) &\rightarrow \tilde{a}_0 + \sum_{i=1}^{\frac{n}{2}-1} \tilde{a}_i(X^i - X^{n-i}), \end{aligned}$$

[294] 여기서,

$(\tilde{a}_0, \tilde{a}_1, \dots, \tilde{a}_{\frac{n}{2}-1}) = (a_0, 2^{-1} \cdot (a_1 \cdot w_m^{-1} + a_{\frac{n}{2}-1} \cdot w_m), \dots, 2^{-1} \cdot (a_{\frac{n}{2}-1} \cdot w_m^{1-\frac{n}{2}} + a_1 \cdot w_m^{\frac{n}{2}-1}))$
이다.

[295] 따라서, 상술한 수학식 23과 24를 정리하면 다음과 같다.

[296] [수학식 25]

[297] $INTT_{n/2}(NTT_{n/2}(a(X)) + NTT_{n/2}(b(X))) = a(X) + b(X)$

$INTT_{n/2}(NTT_{n/2}(a(X)) \odot NTT_{n/2}(b(X))) = a(X) \cdot b(X)$

[298] 즉, R'_q 에 대한 NTT와 Inverse NTT 모두 $O(n \log n)$ 의 복잡도를 가진다. 따라서,

$a(X), b(X) \in R'_q$ 을 만족하는 경우에, $a(X) \cdot b(X)$ 을 계산하고 싶은 경우에

$INTT_{n/2}(NTT_{n/2}(a(X)) \odot NTT_{n/2}(b(X)))$ 로 연산을 수행하는 경우, $O(n \log n)$ 의 복잡도로 연산할 수 있다.

[299] 이하에서는 REAL HEAAN이 적용되는 경우의 비밀키, 공개키 생성 알고리즘, 동형 암호문 생성 방식 및 연산 방식에 대해서 도 12 내지 도 16을 참조하여 보다 자세히 설명한다.

[300]

[301] 도 12는 공개키 및 비밀키의 생성 동작을 설명하기 위한 도면이다.

[302] 도 12를 참조하면, 셋업 모듈(71)은 사용자로부터 스케일링 팩터(Δ), 보안 파라미터(λ), 레벨 파라미터(L)를 입력받을 수 있다. 여기서 스케일링 팩터(Δ)는 메시지의 크기를 조절하는 파라미터로 1보다 크다. 그리고 보안 파라미터(λ)는 얼마나 스킴이 안전한지를 나타내는 지표로, 계산 복잡도로 지칭될 수 있다. 그리고 레벨 파라미터는 연산을 지원하는 탭스를 나타내는 지표이다.

[303] 상술한 바와 같은 파라미터를 입력받은 셋업 모듈(71)은 모듈리(moduli)를 설정하고, 정수(n)와 양수(P)를 랜덤하게 선택하고, $(n, (q_i)_{1 \leq i \leq L}, P, \Delta)$ 와 같은 설정 파라미터를 출력할 수 있다.

[304] 키 생성 모듈(72)은 앞서 셋업 모듈(71)에 생성된 설정 파라미터를 입력받고, 입력받은 설정 파라미터를 기초로 비밀키와 공개키를 생성할 수 있다.

[305] 구체적으로, 키 생성 모듈(72)은 $s \leftarrow X_{key}$, $sk \leftarrow (1, s)$ 에 기초하여 비밀키를 생성하고, $e \leftarrow X_{err}$ 에 따라 에러를 산출할 수 있다.

[306] 그리고 키 생성 모듈(72)은 $a \leftarrow U(R'_{q_L})$ 와 같이 제1 랜덤 다항식을 생성하고, $b \leftarrow -as + e \pmod{q_L}$ 와 같이 제2 랜덤 다항식을 생성할 수 있다. 그리고 앞서 만들어진 제1 랜덤 다항식과 제2 랜덤 다항식을 기초로 앞서 설명한 수학식 21과 같은 공개키를 생성할 수 있다.

[307]

[308] 도 13은 연산 키의 생성 동작을 설명하기 위한 도면이다.

[309] 도 13을 참조하면, 연산 모듈(73)은 $s_1, s_2 \in R'$ 을 만족하는 값과 $a' \leftarrow U(R'_{P, q_L})$,

$e' \leftarrow X_{err}$ 을 산출하고, $b' \leftarrow -a's_2 + e' + P \cdot s_1 \pmod{P \cdot q_L}$ 을 만족하는

$swk \leftarrow (b', a') \in R'_{P \cdot q_L}^2$ 를 산출할 수 있다. 이에 따라 연산 모듈(73)은

$evk \leftarrow KSGen(s^2, s)$ 와 같은 연산 키를 출력할 수 있다.

[310]

[311] 도 14는 근사 동형 암호문의 생성 및 복호 동작을 설명하기 위한 도면이다.

[312] 도 14를 참조하면, 인코딩 모듈(74)은 메시지와 스케일링 팩터를 입력받고, 메시지에 스케일링 팩터를 반영하여 다항식 형태로 변환할 수 있다.

[313] 구체적으로, 인코딩 모듈(74)은 $\vec{m} = (m_j)_{0 \leq j < n/2} \in \mathbb{R}^{n/2}$, 1 이상의 스케일링 팩터를 입력받으면, 메시지를 아래와 같은 수학식 26과 같은 다항식을 출력할 수 있다.

[314] [수학식 26]

[315] $m(X) = \tau^{-1}([\Delta \cdot \vec{m}]_{\tau(R)}) \in R'$

[316] 여기서 $m(x)$ 는 다항식 형태의 메시지이다.

[317] 그리고 암호화 모듈(75)은 다항식 형태의 메시지를 수신하고, 수신한 메시지에 공개키를 반영하여 동형 암호문을 생성할 수 있다. 구체적으로, 아래와 같은 수학식 27을 이용하여 동형 암호문을 생성할 수 있다.

[318] [수학식 27]

[319] $v \cdot pk + (m + e_0, e_1) \pmod{q_L}$

[320] 여기서, v 는 X_{enc} 에 따라서 선택된 원소이고, e_0, e_1 역시 X_{err} 에 따라 선택된 에러 값이다.

[321] 복호화 모듈(76)은 암호문과 비밀키를 입력받아, 암호문을 복호화하여 에러를 포함하는 메시지를 출력할 수 있다. 구체적으로, 복호화 모듈(76)은 입력된 암호문이 $ct = (c_0, c_1) \in R'_{q_1}^2$ 와 같은 경우, $m' = c_0 + c_1 \cdot s \pmod{q_1}$ 와 같은 메시지를 출력할 수 있다.

[322] 한편, 복호화 모듈(76)에서 출력되는 메시지는 다항식 형태의 메시지인바, 디코딩 모듈(77)에서 복호화 모듈(76)에서 출력된 메시지와 스케일링 팩터에 기초하여 메시지를 최종 출력할 수 있다. 구체적으로, 복호화 모듈(76)은 다항식 메시지가 $m(X) \in R'$ 을 만족하는 경우, $\vec{m} = (m_j = \Delta^{-1} \cdot m(\zeta_j))_{0 \leq j < n/2} \in \mathbb{R}^{n/2}$ 와 같은 메시지를 출력할 수 있다.

[323]

[324] 도 15는 동형 암호문의 덧셈 계산을 설명하기 위한 도면이다.

[325] 도 15를 참조하면, 제1 단말 장치(81-1)에서 제1 메시지에 대한 제1 동형 암호문($Ct1$)을 출력하고, 제2 단말 장치(81-2)에서 제2 메시지에 대한 제2 동형 암호문($Ct2$)을 출력한다.

- [326] 제1 동형 암호문(Ct1)과 제2 동형 암호문(Ct2)을 수신한 서버 장치의 연산 모듈(83)은 두 동형 암호문에 대한 덧셈 연산을 수행할 수 있다. 구체적으로, $ct = (c_0, c_1) \in R_{q_1}^2$ 을 만족하는 경우, 연산 모듈(83)은 $ct_{add} \leftarrow ct + ct' \pmod{q_1}$ 와 같이 두 동형 암호문에 대한 덧셈 연산을 수행할 수 있다.
- [327] 즉, 두 동형 암호문에 대해서 제1 함수 처리하고, 함수 처리된 동형 암호문에 대한 덧셈 연산을 수행하고, 덧셈 연산된 동형 암호문을 제1 함수에 역함수인 제2 함수 처리하여 두 동형 암호문에 대한 덧셈 연산을 수행할 수 있다.
- [328] 이후에 실제 값이 필요한 경우, 복호 모듈(85)은 덧셈 결과를 복호 및 디코딩하여 제1 메시지 및 제2 메시지의 합산 값을 출력할 수 있다.
- [329]
- [330] 도 16은 동형 암호문의 곱셈 계산을 설명하기 위한 도면이다.
- [331] 도 16을 참조하면, 제1 단말 장치(81-1)에서 제1 메시지에 대한 제1 동형 암호문(Ct1)을 출력하고, 제2 단말 장치(81-2)에서 제2 메시지에 대한 제2 동형 암호문(Ct2)을 출력할 수 있다. 제1 동형 암호문(Ct1)과 제2 동형 암호문(Ct2)을 수신한 서버 장치의 연산 모듈(87)은 두 동형 암호문에 대한 곱셈 연산을 수행할 수 있다. 구체적으로, $ct = (c_0, c_1), ct' = (c'_0, c'_1) \in R_{q_1}^2$ 을 만족하는 경우, 연산 모듈(87)은 $ct_{mult} \leftarrow (d_0, d_1) + [P^{-1} \cdot d_2 \cdot evk] \pmod{q_1}$ 와 같이 두 동형 암호문에 대한 덧셈 연산을 수행할 수 있다. 여기서,
- $$let (d_0, d_1, d_2) = (c_0 c'_0, c_0 c'_1 + c_1 c'_0, c_1 c'_1) \pmod{q_1} \text{ 이다.}$$
- [332] 즉, 두 동형 암호문에 대해서 제1 함수 처리하고, 함수 처리된 동형 암호문에 대한 곱셈 연산을 수행하고, 곱셈 연산된 동형 암호문을 제1 함수에 역함수인 제2 함수 처리하여 두 동형 암호문에 대한 곱셈 연산을 수행할 수 있다.
- [333] 연산에 의해 획득된 연산 결과 암호문 내의 근사 메시지 비중이 임계치를 초과하면, 리스케일링 모듈(88)은 연산 결과 암호문의 평문 공간을 확장할 수 있다. 구체적으로, 리스케일링 모듈(88)은 아래와 같은 수학식 28과 같이 암호문을 업데이트할 수 있다.
- [334] [수학식 28]
- [335] $ct' \leftarrow [(q_1' / q_1) \cdot ct] \pmod{q_1'}$
- [336] 여기서, $l' = l - 1$ 이다.
- [337] 이후에 복호 모듈(85)은 덧셈 결과를 복호 및 디코딩하여 제1 메시지 및 제2 메시지의 합산 값을 출력할 수 있다.
- [338] 이상에서는 Real HEAAN 방식에 대해서 설명하였으며, 이하에서는 RNS HEAAN 방식에 대해서 설명한다.
- [339] RNS HEAAN은 기존의 HEAAN 방식이 중국인의 나머지 정리와 같은 방법이

$R_{q_i}(q_i = \Delta^i)$ 을 $R_{q_i}(q_i = \prod p_i, p_i \approx \Delta)$ 으로 대체하여 사용하는 방식으로,

이에 따라 에리 비트사이즈가 5~10 정도 큰 근사계산 결과를 갖게 되지만, 연산 속도에서 3~10배의 성능 개선이 있다.

[340] 동작 방식은 앞서 설명한 Real HEAAN 방식과 유사한바, 차이가 있는 부분에 대해서만 이하에서 설명한다.

[341] 먼저, RNS HEAAN에서 모듈러스는 아래의 수학식 29와 같이 서로소인 복수의 정수로 정의될 수 있다.

[342] [수학식 29]

[343] $[P^{-1}]_{q_i} = (\prod_{i=0}^{k-1} p_i)^{-1} \pmod{q_i}$ for $0 \leq j \leq L$

[344] 이에 따라, 생성되는 평가키는 다음과 같은 수학식 30과 같으며, 공개키 역시, 수학식 31과 같이 복수의 서로소에 대응되는 복수의 기저로 표현될 수 있다.

[345] [수학식 30]

[346] $(\text{swk}^{(0)} = (b^{(0)}, a^{(0)}), \dots, \text{swk}^{(k+L)} = (b^{(k+L)}, a^{(k+L)})) \in \prod_{i=0}^{k-1} R_{p_i}^2 \times \prod_{j=0}^L R_{q_j}^2$

[347] 여기서, $b^{(i)} \leftarrow -a^{(i)} \cdot s_2 + e' \pmod{p_i}$ for $0 \leq i < k$ 이고,

$b^{(k+j)} \leftarrow -a^{(k+j)} \cdot s_2 + [P]_{q_j} \cdot s_1 + e' \pmod{q_j}$ for $0 \leq j \leq L$ 이다. 그리고,

$(a^{(0)}, \dots, a^{(L)}) \leftarrow U(\prod_{j=0}^L R_{q_j})$ 이다.

[348] [수학식 31]

[349] $pk \leftarrow (pk^{(j)} = (b^{(j)}, a^{(j)}) \in R_{q_j}^2)_{0 \leq j \leq L}$

[350] 여기서, $b^{(j)} \leftarrow -a^{(j)} \cdot s + e \pmod{q_j}$ for $0 \leq j \leq L$ 이고,

$(a^{(0)}, \dots, a^{(L)}) \leftarrow U(\prod_{j=0}^L R_{q_j})$ 이다.

[351] 따라서, 암호문을 생성하는 경우에 아래의 수학식 32와 같이 복수의 기저(basis)로 표현되는 동형 암호문을 생성된다.

[352] [수학식 32]

[353] $ct = (ct^{(j)})_{0 \leq j \leq L} \in \prod_{j=0}^L R_{q_j}^2$

[354] 여기서, $ct^{(j)} \leftarrow v \cdot pk^{(j)} + (m + e_0, e_1) \pmod{q_j}$ for $0 \leq j \leq L$ 이고, $v \leftarrow X_{enc}$,

$e_0, e_1 \leftarrow X_{err}$ 이다.

[355] 한편, RNS HEAAN 방식으로 생성된 동형 암호문($ct = (ct^{(0)}, \dots, ct^{(L)})$,

$ct' = (ct'^{(0)}, \dots, ct'^{(L)})$)에 대한 덧셈 연산은 다음과 같은 수학식 33과 같이 수행될

수 있다.

[356] [수학식 33]

$$[357] \quad ct_{add} = (ct_{add}^{(j)})_{0 \leq j \leq l}$$

[358] 여기서, $ct_{add}^{(j)} \leftarrow ct^{(j)} + ct'^{(j)} \pmod{q_j}$ for $0 \leq j \leq l$ 이다.

[359] 한편, RNS HEAAN 방식으로 생성된 동형 암호문($ct = (ct^{(j)} = (c_0^{(j)}, c_1^{(j)}))_{0 \leq j \leq l}$,

$ct' = (ct'^{(j)} = (c_0'^{(j)}, c_1'^{(j)}))_{0 \leq j \leq l}$)에 대한 연산은 다음과 같은 단계로 수행될 수

있다.

[360] 먼저, 수학식 34와 같은 연산을 1차적으로 수행한다.

[361] [수학식 34]

$$[362] \quad \begin{aligned} d_0^{(j)} &\leftarrow c_0^{(j)} c_0'^{(j)} \pmod{q_j}, \\ d_1^{(j)} &\leftarrow c_0^{(j)} c_1'^{(j)} + c_1^{(j)} c_0'^{(j)} \pmod{q_j}, \\ d_2^{(j)} &\leftarrow c_1^{(j)} c_1'^{(j)} \pmod{q_j}. \end{aligned}$$

[363] 이후에, 수학식 35와 같이 modup 동작을 수행할 수 있다.

[364] [수학식 35]

$$[365] \quad \text{ModUp}_{p_{c_1} \rightarrow D_1}(d_2^{(0)}, \dots, d_2^{(l)}) = (\tilde{d}_2^{(0)}, \dots, \tilde{d}_2^{(k-1)}, d_2^{(0)}, \dots, d_2^{(l)})$$

[366] 그리고 수학식 36과 같은 2차 연산을 수행할 수 있다.

[367] [수학식 36]

$$[368] \quad \tilde{ct} = (\tilde{ct}^{(0)} = (\tilde{c}_0^{(0)}, \tilde{c}_1^{(0)}), \dots, \tilde{ct}^{(k+l)} = (\tilde{c}_0^{(k+l)}, \tilde{c}_1^{(k+l)})) \in \prod_{i=0}^{k-1} R_{p_i}^2 \times \prod_{j=0}^L R_{q_j}^2$$

[369] 여기서, $\tilde{ct}^{(i)} = (\tilde{d}_2^{(i)} \cdot \text{evk}^{(i)} \pmod{p_i})^{\circ}$ 이고, $\tilde{ct}^{(k+j)} = \tilde{d}_2^{(j)} \cdot \text{evk}^{(k+j)} \pmod{q_j}$ 이다.

[370] 그리고 수학식 37과 같은 연산을 수행하고, 최종적으로 수학식 38과 같은 값을 최종 출력한다.

[371] [수학식 37]

$$[372] \quad \begin{aligned} (\hat{c}_0^{(0)}, \dots, \hat{c}_0^{(l)}) &\leftarrow \text{ModDown}_{D_1 \rightarrow C_1}(\tilde{c}_0^{(0)}, \dots, \tilde{c}_0^{(k+l)}), \\ (\hat{c}_1^{(0)}, \dots, \hat{c}_1^{(l)}) &\leftarrow \text{ModDown}_{D_1 \rightarrow C_1}(\tilde{c}_1^{(0)}, \dots, \tilde{c}_1^{(k+l)}). \end{aligned}$$

[373] [수학식 38]

$$[374] \quad ct_{mult} = (ct_{mult}^{(j)})_{0 \leq j \leq l}$$

[375]

$$ct_{mult}^{(j)} \leftarrow (\hat{c}_0^{(j)} + d_0^{(j)}, \hat{c}_1^{(j)} + d_1^{(j)}) \pmod{q_j}$$

[376] 한편, 연산 이후에 암호문($ct = (ct^{(j)} = (c_0^{(j)}, c_1^{(j)}))_{0 \leq j \leq l} \in \prod_{j=0}^l R_{q_j}^2$)의 적어도

하나 기저에 대한 비중이 임계치를 초과하면, 아래와 같은 수학식 39와 같이 복수의 기저 각각에 대한 메시지의 반올림 처리를 수행하여 동형 암호문을 리스케일링하고, 리스케일링된 암호문(

$$ct' \leftarrow (ct'^{(j)} = (c_0'^{(j)}, c_1'^{(j)}))_{0 \leq j \leq l-1} \in \prod_{j=0}^{l-1} R_{q_j}^2$$

[377] [수학식 39]

[378] $c_i'^{(j)} \leftarrow q_i^{-1} \cdot (c_i^{(j)} - c_i^{(j)}) \pmod{q_j}$ for $i = 0, 1$ and $0 \leq j < l$

[379]

[380] 도 17은 본 개시의 암호화 방법을 설명하기 위한 흐름도이다.

[381] 먼저, 기설정된 계수를 가지는 다항식의 집합인 링(Ring)을 설정한다(S910). 구체적으로, 구체적으로, 원소들 사이에 덧셈과 곱셈이 정의되어 있으며, 덧셈과 곱셈에 대해서 닫혀 있으며, 평문 공간이 실수인 수학식 16과 같은 링을 설정할 수도 있다.

[382] 그리고 링으로부터 비밀키를 산출한다(S920). Xkey에 기초하여 s를 선택하고, 선택된 s에 기초하여 비밀키(SK)을 산출할 수 있다.

[383] 그리고 산출된 비밀키에 대응하여 공개키를 산출한다(S930). 구체적으로, 링으로부터 제1 랜덤 다항식을 산출하고, 에러를 추출하고, 산출된 제1 랜덤 다항식과 비밀키에 에러를 모듈러 연산하여 제2 랜덤 다항식을 산출하고, 산출된 제1 랜덤 다항식과 제2 랜덤 다항식을 이용하여 공개키를 산출할 수 있다.

[384] 그리고 메시지에 공개키를 적용하여 동형 암호문을 생성한다(S940).

구체적으로, 메시지를 링에 속하는 다항식으로 인코딩을 수행하고, 인코딩된 메시지에 공개키를 적용하여 동형 암호문을 생성할 수 있다. 한편, RNS HEAAN을 이용하는 경우, 생성된 복수의 공개키를 이용하여 복수의 기저로 표현되는 동형 암호문을 생성할 수 있다.

[385] 이상과 같이 본 실시 예에 따른 암호화 방법은 에러를 포함하는 동형 암호문을 생성하는바, 고도의 보안성이 요구되는 생체 정보 등에 대해서도 안정적인 보안을 유지할 수 있다. 또한, 본 실시 예에 따른 암호화 방법은 평문 공간이 실수인 링을 이용하여, 연산 과정에서 평문의 허수부가 커지는 것을 원천적으로 차단할 수 있으며, 기존 HEAAN과 같은 안정성, 연산속도, 메모리(암호문, 키 사이즈)를 가짐과 동시에 2배 많은 병렬 실수 연산을 지원 가능하다.

[386]

- [387] 도 18은 본 개시의 일 실시 예에 따른 네트워크 시스템의 암호문 처리 방법을 설명하기 위한 도면이다. 구체적으로, 도 18에서는 두 개의 단말 장치(100-1, 100-2)에서 암호문을 전송하는 경우를 도시하였다.
- [388] 먼저, 단말 장치 1(100-1)에서 제1 메시지가 입력되면(S1010), 단말 장치 1(100-1)은 상술한 방식으로 제1 동형 암호문을 생성한 후(S1015), 제1 서버 장치(200)로 전송할 수 있다(S1020). 일 예로, 단말 장치 1(100-1)의 사용자가 사진을 촬영한 경우에는 그 사진 및 촬영 장소 등이 암호화될 수도 있고, 사용자가 음악 콘텐츠를 재생하거나 특정 URL로 접속하였다면, 그 재생 이력이나 URL 접속 이력 등이 암호화될 수도 있다.
- [389] 단말 장치 2(100-2)도 유사한 형태로 암호문을 생성하여 제1 서버 장치(200)로 전송할 수 있다(S1025, S1030, S1035).
- [390] 제1 서버 장치(200)는 전송된 암호문들을 암호화된 그 상태 그대로 저장할 수 있다. 이러한 상태에서 제2 서버 장치(300)가 특정 연산을 요청한다면(S1040), 제1 서버 장치(200)는 동형암호문에 대해서 제2 서버 장치(300)가 요청한 특정 연산을 수행한 후(S1045), 제2 서버 장치(300)로 전송할 수 있다(S1050). 제2 서버 장치(300)는 전송된 암호문을 복호화하여 메시지를 검출할 수 있다.
- [391] 그리고 제2 서버 장치(300)는 복호화된 메시지를 출력할 수 있다(S1060). 예를 들어, 단말 장치 1, 2(100-1, 100-2)가 각각 사진을 제1 서버 장치(200)로 전송한 상태라면, 제2 서버 장치(300)는 각 장치(100-1, 100-2)에서 촬영된 사진을 한꺼번에 출력하여 줄 수 있다.
- [392] 이상과 같은 암호화 방법 및 암호문 처리 방법은 각종 어플리케이션의 실행 결과뿐만 아니라 사용자가 직접 입력한 각종 신용 정보나 개인 정보들의 수집에도 사용될 수 있다. 제2 서버 장치(300)가 은행 서버인 경우, 은행은 고객들의 신용 정보 중 필요한 항목들만 모아서 확인할 수 있다.
- [393] 한편, 상술한 다양한 실시 예들에 따른 암호화 방법 및 암호문 처리 방법 등은, 각 단계들을 수행하기 위한 프로그램 코드 형태로 구현되어, 기록 매체에 저장되고 배포될 수도 있다. 이 경우, 기록 매체가 탑재된 장치는 상술한 암호화 또는 암호문 처리 등의 동작들을 수행할 수 있다.
- [394] 이러한 기록 매체는, ROM, RAM, 메모리 칩, 메모리 카드, 외장형 하드, 하드, CD, DVD, 자기 디스크 또는 자기 테이프 등과 같은 다양한 유형의 컴퓨터 판독 가능 매체가 될 수 있다.
- [395] 이상 설명한 바와 같이, 본 개시의 다양한 실시 예에 따라 암호화 및 암호문 처리 등을 수행하게 되면 양자 컴퓨터 시대에서도 안전하며, 암호화/복호화 속도를 크게 향상시킬 수 있고 암호문 크기도 줄일 수 있다. 특히, 암호화에 사용되는 메시지를 실수 영역까지 확장시킬 수 있어서 활용성이 크게 증대될 수 있다.
- [396] 이상 첨부 도면을 참고하여 본 개시에 대해서 설명하였지만 본 개시의 권리범위는 후술하는 특허청구범위에 의해 결정되며 전술한 실시 예 및/또는

도면에 제한되는 것으로 해석되어서는 안 된다. 그리고 특허청구범위에 기재된 개시의, 당업자에게 자명한 개량, 변경 및 수정도 본 개시의 권리범위에 포함된다는 점이 명백하게 이해되어야 한다.

[397]

청구범위

- [청구항 1] 단말 장치의 암호화 방법에 있어서,
스케일링 팩터를 설정하는 단계; 및
암호화할 메시지에 상기 스케일링 팩터를 반영한 후 공개키를 이용하여
암호화하여 동형 암호문을 생성하는 단계;를 포함하며,
상기 동형 암호문은, 복호화하였을 때 상기 메시지에 상기 스케일링
팩터를 반영한 값에 에러 값을 가산한 결과값이 복원되는 형태인, 암호화
방법.
- [청구항 2] 제1항에 있어서,
상기 동형 암호문의 모듈러스는,
하나의 스케일링 팩터의 지수승으로 설정되는, 암호화 방법.
- [청구항 3] 제1항에 있어서,
상기 동형 암호문의 모듈러스는,
복수의 서로 다른 스케일링 팩터들을 승산한 값이며,
상기 복수의 서로 다른 스케일링 팩터들은 유사 범위 이내에서 서로 소
관계인 값들로 설정되는, 암호화 방법.
- [청구항 4] 제1항에 있어서,
상기 동형 암호문을 생성하는 단계는,
이산 가우시안 분포 또는 상기 이산 가우시안 분포와 통계적 거리가
가까운 분포로부터 에러를 산출하는 단계; 및
상기 메시지에 상기 스케일링 팩터를 승산하고 상기 에러를 가산한 후,
상기 공개키를 이용하여 암호화하여 상기 동형 암호문을 생성하는
단계;를 포함하는, 암호화 방법.
- [청구항 5] 제1항에 있어서,
상기 메시지가 복수의 메시지 벡터인 경우, 상기 복수의 메시지 벡터를,
병렬적으로 암호화할 수 있는 형태의 다항식으로 변환하는 단계;를 더
포함하며,
상기 동형 암호문을 생성하는 단계는, 상기 다항식에 상기 스케일링
팩터를 승산한 후, 상기 공개키를 이용하여 동형 암호화하는 것을,
특징으로 하는 암호화 방법.
- [청구항 6] 서버 장치의 암호문 처리 방법에 있어서,
개별적으로 암호화된 복수의 동형 암호문을 수신하는 단계;
상기 복수의 동형 암호문에 대하여 기 설정된 연산을 수행하는 단계;
상기 연산에 의해 산출된 결과 암호문으로부터 노이즈 영역을 제거하여
유효 영역의 데이터를 추출하는 단계;를 포함하며,
상기 복수의 동형 암호문 각각은 메시지에 스케일링 팩터를 승산 한 후
암호화한 암호문이며,

- 상기 노이즈 영역은 상기 스케일링 팩터의 크기에 대응되도록 결정된, 암호문 처리 방법.
- [청구항 7] 제6항에 있어서,
상기 복수의 동형 암호문 각각이 복수의 메시지 벡터를 패킹한 암호문이면, 상기 복수의 동형 암호문 각각에 포함된 메시지 벡터들을 병렬적으로 연산 처리하는 단계;를 더 포함하는 암호문 처리 방법.
- [청구항 8] 제7항에 있어서,
상기 복수의 동형 암호문에 포함된 각 메시지 벡터들 간의 순서를 로테이션시키는 단계;를 더 포함하는, 암호문 처리 방법.
- [청구항 9] 제7항에 있어서,
상기 메시지가 복소수인 경우, 상기 복수의 동형 암호문에 대하여 켄주게이션(conjugation) 연산을 수행하는 단계;를 더 포함하는 암호문 처리 방법.
- [청구항 10] 단말 장치에 있어서,
스케일링 팩터 및 공개키가 저장된 메모리;
암호화할 메시지에 상기 스케일링 팩터를 반영한 후 상기 공개키를 이용하여 암호화하여 동형 암호문을 생성하는 프로세서; 및
상기 동형 암호문을 외부 장치로 전송하는 통신부;를 포함하며,
상기 동형 암호문은, 복호화하였을 때 상기 메시지에 상기 스케일링 팩터를 반영한 값에 에러 값을 승산한 결과값이 복원되는 형태인, 단말 장치.
- [청구항 11] 제10항에 있어서,
상기 메시지 및 상기 스케일링 팩터를 입력받기 위한 입력부;를 더 포함하며,
상기 프로세서는 상기 입력부를 통해 입력된 상기 메시지 및 상기 스케일링 팩터를 상기 메모리에 저장하는, 단말 장치.
- [청구항 12] 제10항에 있어서,
상기 프로세서는,
상기 동형 암호문의 모듈러스를 상기 스케일링 팩터의 지수승으로 설정하여 상기 메모리에 저장하는, 단말 장치.
- [청구항 13] 제10항에 있어서,
상기 프로세서는,
상기 동형 암호문의 모듈러스를 복수의 서로 다른 스케일링 팩터들을 승산한 값으로 설정하여 상기 메모리에 저장하며,
상기 복수의 서로 다른 스케일링 팩터들은 유사 범위 이내에서 서로 소 관계인 값들로 설정되는, 단말 장치.
- [청구항 14] 제10항에 있어서,
상기 프로세서는,

이산 가우시안 분포 또는 상기 이산 가우시안 분포와 통계적 거리가 가까운 분포로부터 에러를 산출하고, 상기 메시지 및 상기 스케일링 팩터를 승산한 값에 상기 에러를 가산한 후, 상기 공개키를 이용하여 암호화하는, 단말 장치.

- [청구항 15] 제10항에 있어서,
상기 프로세서는,
상기 메시지가 복수의 메시지 벡터인 경우, 상기 복수의 메시지 벡터를, 병렬적으로 암호화할 수 있는 형태의 다항식으로 변환하며,
상기 다항식에 상기 스케일링 팩터를 승산한 후, 상기 공개키를 이용하여 동형 암호화하는, 단말 장치.
- [청구항 16] 단말 장치에서의 암호화 방법에 있어서,
원소들 사이에 덧셈과 곱셈이 정의되어 있으며 덧셈과 곱셈에 대해서 닫혀 있는 집합 중 평문 공간이 실수인 집합인 링(Ring)을 설정하는 단계;
상기 링으로부터 비밀키를 산출하는 단계;
상기 산출된 비밀키에 대응하여 공개키를 산출하는 단계; 및
메시지에 상기 공개키를 적용하여 동형 암호문을 생성하는 단계;를 포함하는 암호화 방법.
- [청구항 17] 제16항에 있어서,
상기 링은, 아래의 수학식을 만족하는 암호화 방법,
$$a(X)=a(X^{-1}) \text{ where } a(X) \in R, X^{-1} := -X^{n-1}$$

여기서, $a(X)$ 는 상기 링에 속하는 다항식, R 은 평문 공간이 복소수인 링.
- [청구항 18] 제16항에 있어서,
상기 동형 암호문은,
덧셈 또는 곱셈 연산에 대해서 $O(n \log n)$ 복잡도(complexity)를 갖는 암호화 방법.
- [청구항 19] 제16항에 있어서,
상기 링으로부터 제1 랜덤 다항식을 산출하는 단계;
에러를 추출하는 단계; 및
상기 제1 랜덤 다항식 및 상기 비밀키에 상기 에러를 모듈러 연산하여 제2 랜덤 다항식을 산출하는 단계;를 더 포함하고,
상기 공개키를 산출하는 단계는,
상기 제1 랜덤 다항식 및 상기 제2 랜덤 다항식을 이용하여 상기 공개키를 산출하는 암호화 방법.
- [청구항 20] 제16항에 있어서,
상기 동형 암호문을 생성하는 단계는,
상기 메시지를 상기 링에 속하는 다항식으로 산출하고, 상기 산출된 다항식에 상기 공개키를 적용하여 동형 암호문을 생성하는 암호화 방법.

- [청구항 21] 제16항에 있어서,
상기 생성된 동형 암호문 및 기생성된 동형 암호문에 대해서 덧셈 및 곱셈 중 적어도 하나의 연산을 수행하는 단계;를 더 포함하고,
상기 연산을 수행하는 단계는,
상기 생성된 동형 암호문 및 상기 기생성된 동형 암호문 각각을 제1함수 처리하고, 제1함수 처리된 동형 암호문 간에 연산을 수행하고, 상기 연산 수행된 동형 암호문을 상기 제1 함수에 역함수인 제2 함수 처리하는 암호화 방법.
- [청구항 22] 제21항에 있어서,
상기 제1 함수는,
NTT(Number Theoretic Transform) 함수인 암호화 방법.
- [청구항 23] 제16항에 있어서,
상기 동형 암호문은, 복호화하였을 때 상기 메시지에 에러 값을 가산한 결과값이 복원되는 형태인 암호화 방법.