ETRI

전자통신동향분석 제 23권 제 4호 2008년 8월

검색 가능 암호 시스템 기술 동향

Technical Trend of the Searchable Encryption System

21세기를 대비하는 정보보호 특집

조남수 (N.S. Jho) 암호기술연구팀 선임연구원

홍도원 (D.W. Hong) 암호기술연구팀 팀장

목 차

검색 가능 암호 시스템은 암호화된 자료를 복호화하지 않고도 원하는 자료를 검색할 수 있도록 하는 암호 기반 기술이다. 검색 가능 암호 시스템은 개인의 정보가 외부 저장 공간에 저장되면서 발생하는 여러 문제점에 대한 해결 방법으로 지금까지 많은 연구가 진행되었다. 본 고에서는 검색 가능 암호 시스템의 기본 구조, 요구 조건, 안전성 모델 등을 살펴본 후 지금까지 제안된 중요 검색 가능 암호 시스템들에 대해 살펴보기로 한다.

I. 서론

현대 사회는 모든 정보를 디지털화하여 저장하고, 저장된 정보를 네트워크를 통해 공유하여 사용하는 사회로 변화하고 있다. 또한 처리하는 자료의 양이 증가하고 다양한 서비스에 대한 요구가 늘어나면서 특화된 외부 저장 공간의 활용이 늘고 있다. 이메일, 웹 하드 서비스, 카페나 개인 홈페이지 제공 서비스 등은 개인이 외부 저장 공간을 이용하는 대표적인 예라고 할 수 있다. 이러한 외부 저장공간은 제한된 자원을 지니고 있는 개인이 상대적으로 많은 자원을 지닌 서버를 통해 다양한 서비스를 제공 받을 수 있다는 장점때문에 널리 확산되고 있다. 이 외에도 기업이 고객 정보를 효율적으로 관리하기 위해 사용하는 데이터베이스 또한 개인의 정보가 외부 저장 공간에 저장되는 예로 볼 수 있다.

최근 기업 데이터베이스에서의 고객 정보 유출 사례나 개인 홈페이지에 저장된 사진 등의 개인 정보의 유출 사례가 보고되면서, 이러한 외부 저장 공간에 저장된 정보에 대한 보안 문제가 이슈가 되고 있다. 외부 저장 공간에서의 보안 문제는 과거 개인이 독립된 저장 공간을 이용하여 스스로 정보를 관리하던 때와는 차이를 보인다. 이는 근본적으로 정보의 소유자와 저장 공간을 관리하는 주체가 서로 다르기 때문이다. 데이터베이스 등에서 정보를 보호하기 위해 주로 사용되는 접근 제어나 키 관리 기법들은 외부 침입자를 막는 데 유효한 방법이지만 저장 공간의 소유자가 저장되어 있는 자료를 열람하는 것을 근본적으로 방지하지 못한다.

정보를 안전하게 저장하기 위한 다른 방법으로 암호화를 생각할 수 있다. 즉, 외부 저장 공간에 저장할 정보를 안전성이 증명된 암호 시스템을 이용하여 암호화하는 것이다. 안전성이 증명된 암호 시스템은 복호화 키를 소유하지 못한 공격자가 암호문으로부터 실제 저장된 정보를 얻을 수 없다는 것을 보장한다. 따라서, 외부 침입자 또는 저장 공간의 소유자가 외부 저장 공간에 저장된 암호문에 접근했다 하더라도 실제 의미 있는 정보를 얻지는 못한다는 것을 의미한다. 정보의 암호화는 저장된 정보의 유출을 완벽하게 방지하는 방법이지만, 기존 데이터베이스가 제공하는 많은 부가 기능 또한 적용할 수 없도록 만든다. 저장된 정보의 양이 많을수록 이를 효율적으로 활용하고 정리하기 위해서 다양한 데이터베이스 기능이 요구되기 때문에 단순히 정보를 암호화하여 저장하는 방법은 적당한 해법이라 보기 힘들다.

검색 가능 암호 시스템은 기존의 암호 기술과 같이 암호화된 정보에 대한 안정성을 보장하면서 동시에 특정 키워드를 포함하는 정보를 검색할 수 있도록 고안된 암호 기술이다. 데이터베이스에서 제공되는 다양한 기능 중 많은 경우가 특정 키워드를 포함하는 정보에 대한 검색을 바탕으로 이루어지기 때문에 검색 가능 암호 시스템은 앞에서 제기된 문제에 대한 해결 방안 중 하나로 여겨지고 있다. 또한 기본적인 검색 이외에도 범위 검색, conjunctive 검색 등의 다양한 검색 기능을 제공하는 검색 가능 암호 시스템[1]-[4]에 대한 연구도 진행중이다.

본 고에서는 검색 가능 암호 시스템의 기본 구조, 요구 조건, 안전성 모델 등을 살펴본 후 지금까지 제안된 중요 검색 가능 암호 시스템들에 대해서 정리하기로 한다.

Ⅱ. 기본 지식

1. 기본 구조

검색 가능 암호 시스템에서 암호화의 대상인 정보를 자료(document)라 부른다. 즉, 자료는 사용자가 숨기고 싶은 정보이다. 또한, 사용자가 자신이 원하는 자료를 검색하기 위해 서버에 제공하는 정보를 키워드(keyword)라고 부른다. 일반적으로 자료는 그 자료에 포함된 키워드들의 집합으로 정의된다. $D = \{W_1, W_2, ..., W_n\}$

검색 가능 암호 시스템은 키 생성(key generation), 암호화(build index), 트랩도어 생성(trapdoor generation), 검색(search)의 4가지 단계로

이루어진다. 키 생성 단계는 사용자가 앞으로 사용할 검색 가능 암호 시스템을 준비하는 단계이다. 키 생성 단계에서 각 사용자는 자신의 비밀키를 생성하여 저장하고 암호 시스템의 공개 정보는 서버나 다른 사용자들에게 공개한다.

암호화 단계에서 사용자는 주어진 자료에 대해 자료 자체를 암호화한 암호문과 자료에 포함된 키워드의 정보를 포함한 인덱스(index)를 생성한다. 암호문과 인덱스는 모두 서버에 저장된다. 암호화 단계에서 사용자의 비밀키가 사용되는 검색 가능 암호 시스템을 비밀키 기반 검색 가능 암호 시스템(symmetric-key searchable encryption)이라 부르고 공개키만을 이용하여 암호문 및 인덱스의 생성이 가능한 시스템을 공개키 기반 검색 가능 암호 시스템(public-key searchable encryption)이라 부른다.

트랩 도어 생성 단계는 사용자에 의해서 실행되며 주어진 키워드에 해당하는 트랩 도어를 생성한다. 트랩 도어는 오직 사용자의 비밀키로부터 생성이 가능하다.

마지막, 검색 단계는 주어진 트랩 도어에 대응하는 자료를 찾는 단계로 서버에 의해서 실행된다. 검색의 결과로 서버는 주어진 트랩 도어와 일치하는 자료의 암호문-또는 자료의 식별자(identifier)-을 사용자에게 전달한다.

검색 가능 암호 시스템은 다음과 같은 요구 조건을 만족시켜야 한다. 우선 검색 단계에서는 주어진 트랩 도어와 일치하는 모든 자료들이 검색되어야 하며, 검색에서 발생할 수 있는 오류는 최소화되어야 한다. 여기에서 오류란 주어진 트랩 도어에 대응하는 키워드를 포함하고 있지 않은 자료가 검색 결과에 포함될 확률을 의미한다.

정보 보호 측면에서 볼 때, 검색 과정에서 유출되는 정보의 양은 가능하면 작아야 한다. 좀 더 구체적으로 주어진 트랩 도어와 관계없는 또는 일부 키워드만을 포함하는 자료에 대한 정보는 유출이 되어서는 안된다.

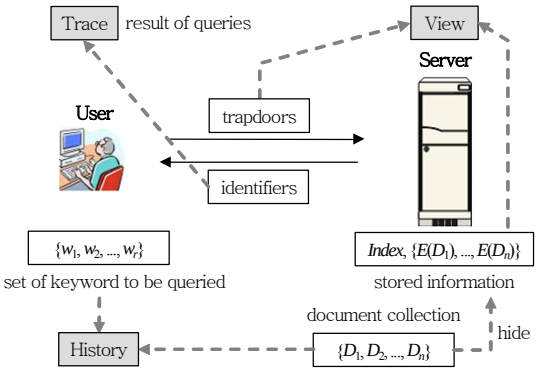
2. 안전성

검색 가능 암호 시스템의 안전성을 정의하기 위해서는 history, view, trace의 세 가지 용어의 정의가 필요하다((그림 1) 참조)[5].

우선 history는 사용자와 서버 사이의 상호 작용을 의미하는데, 이러한 상호 작용은 서버에 저장되어 있는 자료의 집합과 사용자가 원하는 검색의 내용에 의해서 유일하게 결정된다. 따라서, history는 자료와 질문된 키워드의 집합으로 표현할 수 있다. History는 사용자가 공격자로부터 숨기고 싶은 정보라는 의미를 지닌다. 저장된 자료의 내용이나 질문한 키워드들은 모두 사용자가 공격자로부터 숨기고 싶어하는 내용이다.

View는 공격자가 사용자와 서버 사이의 상호 작용을 관찰하여 얻는 정보를 의미한다. View에는 서버에 저장되어 있는 인덱스와 사용자가 질의를 위해서 서버에 보내는 트랩 도어, 마지막으로 서버를 관찰하여 얻을 수 있는 암호문과 자료의 개수 등의 공개정보가 포함된다.

Trace는 사용자가 주어진 history에 의해서 유출되는 것을 용인하는 정보로 검색 가능 암호 시스템이 안전하다는 의미는 실제로 trace 이외의 어떠한 정보도 공격자에 유출되지 않는 것을 의미한다. Trace에는 각각의 검색에 대응하는 결과물이 포함되고 더 나아가 저장된 암호문 중 어느 부분을 검색하였는지에 대한 정보 등이 포함된다.



(그림 1) History, View, Trace의 정의

요약해 보면, history는 우선 사용자가 저장하고 있는 자료, 사용자가 검색하고자 키워드를 의미하고, view는 주어진 history에 의해서 사용자가 검색을 수행하는 동안 공격자가 관찰할 수 있는 정보이며, 마지막으로 trace는 주어진 history에서 공격자에 제공되는 정보이다.

이러한 용어들을 이용하여 다음과 같은 검색 가능 암호 시스템의 안전성을 정의할 수 있다.

- Indistinguishable(IND): 어떠한 공격자도 동일한 trace를 가지는 두 history로부터 생성된 view를 구별할 수 없음을 의미한다.
- Chosen Keyword Attack(CKA): 공격자의 능력에 대한 가정으로 공격자는 공격 도중 자신이 선택한 키워드에 대해 사용자가 트랩 도어를 생성하도록 요구할 수 있다.
- Adaptive Chosen Keyword Attack(CKA2): CKA와 다른 점은 공격자가 키워드를 선택하는 시점이다. CKA2에서는 공격자가 언제든지 새로운 키워드를 선택하여 그에 대응하는 트랩 도어를 요구할 수 있다고 가정한다.

IND-CKA2의 안전성은 2004년 Boneh 등[6]에 의해서 제안되었으며, 그 이후 검색 가능 암호 시스템에는 IND-CKA2의 안전성을 요구하고 있다.

3. 효율성

검색 가능 암호 시스템의 효율성 측면으로는 암호문의 크기, 트랩 도어의 크기, 암호화 시간, 트랩 도어 생성 시간, 검색 시간 등을 고려한다. 암호문의 크기는 검색을 위해 추가로 저장하여야 하는 인덱스의 크기를 의미한다. 이 중 가장 중요한 척도는 암호문의 크기와 검색 시간이다.

일반적으로 계산 속도에서 비밀키 기반의 암호 시스템과 공개키 기반의 암호 시스템은 큰 차이를 보인다. 이는 검색 가능 암호 시스템에서도 마찬가지이며 따라서 연구 방향에 있어서도 차이가 나타난다. 비밀키 기반의 검색 가능 암호 시스템에서는 효율성을 최적화하는 것이 주된 목적이며 공개키 기반의 시스템에서는 다양한 부가기능이나 더욱 완벽한 안전성을 지니는 시스템의 연구가 중심이 되고 있다.

Ⅲ. 대칭키 검색 가능 암호 시스템

1. Oblivious RAM

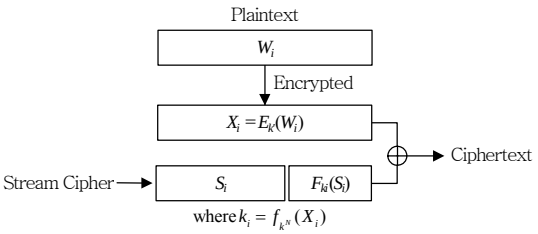
Oblivious RAM은 1996년 R. Ostrovsky와 P. Golle[7],[8]에 의해서 제안되었다. Oblivious RAM은 기본적으로 검색 가능 암호 시스템으로 보기는 힘들지만, 추가적인 정보를 유출하지 않으면서 원하는 정보를 검색할 수 있는 검색 방식을 제안했다는 점에서 검색 가능 암호 시스템의 초기 형태라는 의미를 지닌다.

초기의 Oblivious RAM은 사용자의 검색 키워드에 관계없이 검색을 시행한 시간에 의존하여 메모리를 검색하도록 하였다. 따라서 사용자가 검색한 위치로부터 사용자가 원하는 키워드에 대한 정보를 얻을 수 없도록 고안되었다. 하지만, 이러한 방법은 검색 결과에 관계 없이 많은 검색량을 지니게 되어 현실에서 사용하기에는 부적합하다고 알려져 있다.

2. Hidden Search

Hidden Search는 2000년 Song, Wagner, Perrig[9]에 의해서 제안된 시스템으로 평문의 정보를 유출하지 않으면서 검색할 수 있도록 고안되었다. 명확한 안전성이 정의되지는 않았지만 초기의 검색 가능 암호 시스템이라 할 수 있다.

(그림 2)는 Song 등이 제안한 방식을 간략하게 보여준다.



(그림 2) Hidden Search의 구조

기본적인 대칭키 암호 시스템을 이용하여 평문을 암호화하고 다시 스트림 암호에서 생성된 난수열과 결합하여 최종 암호문을 만들어 낸다.

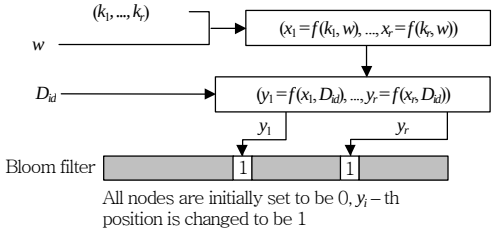
암호문을 검색하기 위해 필요한 트랩 도어는 $<X, k> = <E_{k'}(W), f_{k''}(E_{k'}(W))>$로 생성된다. 단 여기에서 k'과 k''는 사용자의 비밀키이다. 주어진 암호문 C에 대해서 $C \oplus X$를 계산하여 $T = <S, V>$를 계산하고 $V = F_k(S)$를 만족하는 C를 모아 사용자에게 전달한다. 즉 암호문 C를 해독하지 않고도 평문의 특정 정보를 이용한 검색이 가능하다.

3. Secure Index

처음으로 안전성이 증명된 검색 가능 암호 시스템으로 2003년 Goh[10]가 Bloom filter를 사용하여 설계하였다. 또한 Goh는 처음으로 검색 가능 암호시스템에 대한 명확한 안전성 정의를 제시하였으며, 이를 이용하여 제안된 검색 가능 암호 시스템이 안전함을 증명하였다. 하지만 Bloom filter를 사용하였기 때문에 검색 결과 중 오류가 포함될 확률을 지니고 있다. 이러한 오류의 확률을 줄이기 위해서는 더 큰 Bloom filter를 사용하여야 하는데 너무 큰 Bloom filter를 사용하는 경우 검색이 비효율적이 된다.

기본 구조는 (그림 3)과 같다. 사용자는 Bloom filter에 사용할 상수 r과 비밀키 $k_1, k_2, ..., k_r$을 임의로 선택한다. 또한 일방향 함수 f를 선택하여 주어진 키워드 w에 대해서 $x_1 = f(k_1, w), \ldots, x_r = f(k_r, w)$을 계산한다. 다시 각각의 자료에 대해서 자료의 식별자 D_{ID}를 이용하여 $y_1 = f(x_1, D_{ID}), \ldots, y_r = f(x_r, D_{ID})$를 계산한다.

자료 D_{ID}의 키워드 w에 대해 사용자는 n bits의 Bloom filter를 정의한다. 여기에서, f는 $f : \{0, 1\}^* \rightarrow \{0, 1\}^n$를 의미한다. 초기의 Bloom filter는 모두 0의 값으로 채워지고, 후에 D_{ID}와 w가 결정되면 앞에서 설명한 계산을 통해 얻어진 y_i번째 bit의 값을 1로 변경한다.



(그림 3) Secure Index

키워드 w에 대한 트랩 도어는 $x_1 = f(k_1, w), \ldots, x_r = f(k_r, w)$로 비밀키 $k_1, k_2, ..., k_r$를 모두 알고 있는 사용자만이 구성이 가능하다. 또한 주어진 트랩 도어로부터 다른 키워드에 대한 트랩 도어를 생성하는 것은 일방향 함수의 안전성으로부터 보장된다. 서버는 주어진 트랩 도어와 각 자료의 식별자인 D_{ID}를 이용하여 $y_i = f(x_i, D_{ID})$를 계산한다. 각 자료의 Bloom filter를 검사하여 모든 y_i번째 bit에 1의 값을 가지는 자료만을 결과로 사용자에 통보한다.

Secure index는 임의의 일방향 함수 f를 사용하기 때문에 사용자가 원하지 않는 자료가 검색될 확률이 존재한다. 즉, $y_i = f(x_i, D_{ID})$의 계산 과정에서 다른 x_i의 값으로부터 동일한 y_i가 계산될 수 있다. 이것은 검색 단계의 오류인데 보다 큰 r을 선택하여 이러한 오류의 확률을 줄일 수 있다. 단 이 경우 인덱스의 크기가 증가하고 또한 검색 과정의 계산량이 증가하여 검색에 더 많은 시간이 소요되어 효율성이 떨어진다.

● 용 어 해 설 ●

Bloom filter: 1970년 Burton H. Bloom에 의해서 고안된 확률적인 자료 구조이다. 주어진 집합에 특정 원소가 속해 있는지를 판단하는 데 사용된다. 구성에는 해시 함수를 사용하며 bit들의 집합인 filter에서 해시 함수로 계산된 위치를 1로 치환하여 자료를 저장한다.

4. Privacy Preserving Keyword Search

2005년 Chang 등이 제안한 방식[11]은 매우 작은 트랩 도어를 사용하는 등 검색 가능 암호 시스템에 대한 현실적인 요구를 반영하여 고안되었다. 또한 이 시스템은 유사 난수 생성 함수만을 사용하고 있다. 기존의 검색 가능 암호 시스템들이 각각의 키

워드에 대한 인덱스를 각각 별도로 저장했던 것에 비해서 이 시스템은 각각의 키워드마다 단지 1bit의 정보만을 사용한다. 즉, 자료가 주어진 키워드를 포함하는지의 여부만을 저장하는 것이다. 또한 해시 테이블을 이용한 방식으로 저장량을 최소화 할 수도 있다. 단 이 경우 검색과정 동안 사용자와 서버간의 통신 횟수가 증가한다.

사용자는 비밀키 s와 r을 선택한다. 인덱스를 구성하기 위해서 사용자는 각각의 자료에 대해서 2^d-bits의 배열을 구성하는데 초기에는 모든 배열은 0의 값을 가진다. 만약 자료가 i번째 키워드 w_i를 포함한다면, 유사 난수 생성 함수 $P_s(i)$의 값을 계산하여 배열의 $P_s(i)$번째 bit를 1로 치환한다.

만약 이러한 배열로 이루어진 인덱스를 서버에 그대로 저장한다면 서버는 각각의 배열 값을 보고 주어진 자료가 어떤 키워드를 포함하는지의 정보를 얻을 수 있다. 따라서 배열을 암호화할 필요가 있는데, 이때, $M_j[i] = I_j[i] \oplus G_{r_i}(j)$와 같이 각 bit에 대해 마스크를 만들어 적용하게 된다. 여기에서 $r_i = F_r(i)$로 F_k와 G_k는 각각 k에 의해서 결정되는 유사 난수 발생 함수 패밀리이다.

키워드 w_i에 대한 트랩 도어는 식 (1)과 같다.

$$T = < p = P_s(i),\ f = F_r(p) > \qquad (1)$$

사용자는 각 키워드에 대한 키워드 인덱스 i에 대한 사전을 별도로 저장하고 있어야 한다. 이러한 사전을 해시 함수 등을 사용하여 생성하는 것도 생각해 볼 수는 있지만, 해시 함수의 충돌쌍 문제에 의해서 오류가 발생할 수 있으므로 사전은 별도로 생성하여 저장하여야 한다. 이 사전을 서버에 저장하는 경우, 사용자의 저장량을 최소화시킬 수 있지만, 트랩 도어를 생성하기 위해서 서버에 키워드에 해당하는 키워드 인덱스를 질의하여야 하므로 통신 복잡도가 증가하게 된다.

주어진 트랩 도어에 대해서 서버는 다음과 같이 검색을 수행한다. 우선, 모든 j에 대해서 $I_j[p] = M_j[p] \oplus G_f(j)$를 계산하여 만약 $I_j[p] = 1$이면, 서버는 자료의 식별자를 사용자에게 전송한다.

5. Searchable Symmetric Encryption

Curtmola 등[12]은 대칭키 기반 검색 가능 암호 시스템에 대한 기존의 adaptive security 정의를 새롭게 수정하였으며, 이를 바탕으로 새로운 검색 가능 암호 시스템인 SSE를 제안하였다. SSE에 사용된 주요 기술은 해시 테이블과 링크드 리스트(linked list)이다. 기존에 제안된 모든 검색 가능 암호 기술이 서버에 저장된 모든 자료의 인덱스에 대해 검색을 수행하던 것에 비해서 SSE는 주어진 키워드에 대응하는 자료의 인덱스만을 검사한다. 따라서 검색 속도 면에서 기존의 시스템들과 차별성을 지닌다. 하지만, 하나의 자료가 여러 키워드와 관련성을 지닐 때, 중복 저장이 발생하여 저장 공간 면에서는 상당히 비효율적이라 할 수 있다.

키 생성 과정에서 사용자는 유사 난수 생성 함수 P와 비밀키 s, y, z를 선택한다. 주어진 키워드 w_i에 대해서 사용자의 모든 자료를 조사하여 키워드 w_i를 포함하는 자료들로 집합인 D_i를 구성한다. 이 D_i를 링크드 리스트의 형태로 서버에 저장한다.

링크드 리스트는 일종의 배열로 다음과 같은 구조를 지닌다. 사용자는 초기에 적당한 크기의 배열을 선언한다. 배열의 각 원소 또는 노드는 유일한 주소값이 정해져 있다. 또한 각각의 노드에는 자신의 노드 다음에 연결되어 따라올 노드의 주소를 저장하기 위한 공간이 할당되어 있는데 이러한 노드의 주소를 링크(link)라고 한다. 그리고 노드들이 링크에 의해서 연결되어 있기 때문에 링크드 리스트라고 불린다.

SSE에서 사용하는 노드는 (2)와 같이 구성되어 있다.

$$N_{i,j} = < ID(D_{i,j}) \,\|\, k_{i,j} \,\|\, P_s(ctr+1) > \qquad (2)$$

노드의 첫번째 값은 이 노드에 대응하는 자료의 식별자이다. 그 이후의 $k_{i,j} \| P_s(ctr+1)$은 다음에 따라올 노드를 위한 링크인데, 이 중 $P_s(ctr+1)$은 다음 노드의 주소를 나타내고, $k_{i,j}$는 다음 노드를 암호화하기 위한 비밀키를 의미한다. 유사 난수 생성 함수인 $P_s(ctr+1)$을 이용하여 각 자료의 위치를 배열 내에서 임의로 선택하고, $k_{i,j}$를 이용하여 각 노드를 암

호화하여 서버가 저장된 노드로부터 자료에 대한 정보를 얻을 수 없도록 한다.

사용자가 D_i를 링크드 리스트에 저장하고자 할 때는 우선 ctr 값을 0으로 초기화하고 링크드 리스트가 시작될 위치를 $P_y(w_i)$로 계산한다. D_i의 첫번째 자료를 $P_y(w_i)$를 주소값으로 갖는 노드에 할당하고 자료의 식별자를 저장한다. 다음 $P_s(ctr + 1)$을 계산하여 다음 노드를 선택하고 다음 자료를 저장한다. 이 과정을 D_i에 속한 모든 자료를 저장할 때까지 반복하며 마지막 노드의 링크는 NULL로 비워둔다. 마지막 단계로 D_i를 저장하는 데에 사용된 모든 노드를 각각 암호화한다. 사용자는 리스트의 첫번째 노드의 위치와 암호화 키를 $T[P_z(w_i)] = < P_y(w_i), k_{i,0} >$의 형태로 해시 테이블 T에 저장한다. 자료가 저장되어 있는 배열과 마지막에 구성한 해시 테이블은 서버에 저장된다.

위의 과정에서 만약 하나의 자료가 n개의 키워드를 포함하고 있다면 이러한 자료는 모두 n개의 서로 다른 링크드 리스트에 저장이 되어야 하므로 n번 중복이 발생한다. 이러한 이유로 실제 자료의 수에 비해 훨씬 큰 저장공간이 필요하게 된다.

주어진 키워드 w에 대한 트랩 도어는 $(P_z(w), P_y(w))$로 주어진다. 서버는 우선 해시 테이블에서 $T[(P_z(w)]$를 찾아내어 링크드 리스트의 첫 노드의 주소와 이에 해당하는 암호화 키를 찾아낸다. 서버는 링크를 이용하여 배열 내에서 링크드 리스트를 구성하는 모든 노드를 찾아내어 복호화 한다. 이 복호화를 통해서 서버는 링크드 리스트에 저장되어 있는 자료의 식별자를 얻을 수 있고, 링크가 NULL 값을 가지는 노드를 발견하면 검색을 중단하고 지금까지 발견한 모든 자료의 식별자를 사용자에게 전달한다.

Ⅳ. 공개키 검색 가능 암호 시스템

1. 기본 개념

공개키 기반의 검색 가능 암호 시스템은 공개키 기반 암호의 장점을 이어 받아 시스템의 공개 정보만을 이용하여 누구나 암호문을 생성할 수 있도록 한다. 즉, 서버에 저장되는 자료의 제공자와 검색을 하는 주체인 사용자가 서로 다르다는 것을 의미한다. 따라서 자료의 제공자와 사용자가 동일하던 대칭키 기반의 시스템보다 더 많은 응용 범위를 가진다. 가장 널리 이해되는 예로는 이메일 서버를 들 수 있다. 즉, 사용자가 자신의 시스템을 구축하여 공개키를 공개하면, 사용자에 메일을 전송하고 싶어하는 사람은 사용자의 공개키를 이용하여 메일을 암호화하고 암호화된 메일을 서버에 저장하는 방식이다. 사용자는 서버에 저장된 메일 중에서 자신이 원하는 키워드를 포함한 메일을 선택적으로 검색하여 처리할 수 있다.

공개키 기반 검색 가능 암호 시스템은 또한 대칭키 기반의 시스템이 제공하지 못한 구간 검색, conjunctive 검색 등 다양한 부가 기능을 구현하는 데에도 적합하다. 반면 계산 속도에 있어서는 큰 단점을 지니고 있다.

2. 겹선형 사상

지금까지 다양한 공개키 기반 검색 가능 암호시스템이 제안되었는데, 최근의 결과들은 대부분 시스템의 설계에 겹선형 사상(bilinear map)을 사용하였다. 겹선형 사상은 타원곡선 위에서 정의된 Weil pairing 또는 Tate pairing으로 대표되는데, 다음과 같은 성질을 지닌다.

겹선형 사상 $e : G_1 \times G_1 \rightarrow G_2$에 대해

1. Computable: 주어진 G_1의 두 원소 g, h에 대해서 $e(g, h)$를 계산할 수 있는 효율적인 다항시간 알고리듬이 존재한다.
2. Bilinear: 모든 정수 x, y와 모든 원소 g, h에 대해서, $e(g^x, h^y) = e(g, h)^{xy}$를 만족한다.
3. Non-degenerate: 만약 g가 G_1의 생성자라고 할 때, $e(g, g)$ 또한 G_2의 생성자이다.

겹선형 사상은 Joux가 삼자간 키 공유 시스템에 처음 사용한 이후 다양한 암호 응용 프로토콜의 설계에 사용되고 있다.

3. Public-key Encryption with Keyword Search

Boneh 등[6]은 처음으로 공개키 기반의 검색 가능 암호 시스템의 정의를 제시하였으며, 이를 만족하는 최초의 공개키 기반 검색 가능 암호 시스템을 제안하였다. 이들은 두 개의 알고리듬을 소개하였는데, 이 중 하나에서 겹선형 사상을 사용하였고, 다른 하나는 trapdoor permutation을 사용하여 시스템을 구성하였다. 또한 검색 가능 암호 시스템에 대한 안전성 모델을 완성하고 이를 바탕으로 한 IND-CKA2의 안전성을 지니는 것을 증명하였다. 여기에서는 겹선형 사상을 이용한 시스템을 소개한다.

사용자는 겹선형 사상이 잘 정의된 두 개의 군 G_1, G_2를 선택하고 비밀키 a와 공개키 $h = g^a$를 결정한다. 자료 D가 키워드 w를 포함하고 있다면 사용자는 임의의 r을 선택하여 $peks(w) = [g^r, h_2(e(h_1(w), h^r))]$을 계산하여 인덱스를 생성한다. 여기에서 h_1, h_2는 해시 함수이다. 각 자료의 인덱스 크기가 자료에 포함된 키워드의 수를 의미하므로 사용자는 이 정보를 숨기기 위해서 의미 없는 인덱스를 임의로 생성하여 모든 자료에 동일한 수의 인덱스를 할당한다.

키워드 w에 대한 트랩 도어는 $T_w = h_1(w)^s$이다. 서버는 주어진 트랩 도어 T_w와 인덱스로 저장된 모든 $peks(w) = [A, B]$에 대해서 $h_2(e(T_w, A)) = B$를 만족하는 모든 자료를 검색의 결과로 사용자에게 전송한다.

Boneh 등의 시스템은 이후 설계된 대부분의 공개키 기반 검색 가능 암호 시스템의 바탕이 되었으며, 안전성 정의와 증명 방법 또한 여전히 사용되고 있다.

4. Conjunctive, Subset, Range 검색

2007년 Boneh와 Waters[13]는 다양한 부가 검색 기능을 지니는 공개키 기반 검색 가능 암호 시스템을 제안하였다. 이들은 일반적인 키워드 인덱스에 대해 효율적으로 conjunctive 검색을 수행할 수 있도록 시스템을 고안하였고, 키워드 인덱스의 결합을 통해서 subset과 범위 검색을 효율적으로 처리하였다.

사용자는 겹선형 사상이 잘 정의된 두 개의 군 G_1, G_2를 선택한다. 단, G_1, G_2는 일반적인 겹선형 사상 군이 소수 위수를 갖는 것에 비해 $n = pq$인 합성수 위수를 갖도록 선택한다. G_p, G_q는 각각 p, q를 위수로 갖는 G_1의 부분 군이다. 사용자는 자연수 l을 선택하고, 비밀키 $(u_1, h_1, w_1), ..., (u_l, h_l, w_l) \in G_p^3$을 임의로 선택한다. 또한 $g, v \in G_p$, $g_q \in G_q$, $a \in Z_p$를 비밀키로 선택한다.

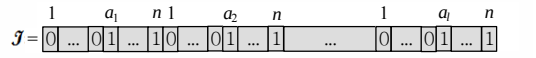
시스템의 공개키로 g_q, $V = vR_v$, $A = e(g, v)^a$와 각각의 $i \in [1, l]$에 대해서 $U_i = u_i R_{u,i}$, $H_i = h_i R_{h,i}$, $W_i = w_i R_{w,i}$를 계산하여 공개한다.

키워드는 l차원의 벡터로 표현되는데, $K = (k_1, k_2, ..., k_l)$에 대응하는 인덱스는 다음과 같이 생성된다. 사용자는 $s \in Z_n$과 $Z, (Z_{11}, Z_{12}), ..., (Z_{l1}, Z_{l2}) \in G_q$를 임의로 선택하여 $C' = MA^s$, $C_0 = V^s Z$ 그리고 각각의 $i \in [1, l]$에 대해서 $C_{i,1} = (U_i^{k_i} H_i)^s Z_{i,1}, C_{i,2} = W_i^s Z_{i,2}$를 생성한다.

이 시스템에서는 키워드로 미리 정의된 '*'를 사용할 수 있는데, '*'는 어떠한 키워드와도 일치한다는 의미를 지닌다. 사용자가 검색하고자 하는 키워드를 $K = (k_1, k_2, ..., k_l)$라 하자. 각각의 k_i는 '*'의 값을 가질 수 있다. 사용자는 각각의 $i \in [1,l]$에 대해서 임의의 $r_{i,1}, r_{i,2}$를 선택한 후, 트랩 도어로 $T = (K, K_0 = g^a \prod_{i \in [1,l]} (u_i^{k_i} h_i)^{r_{i,1}} w_i^{r_{i,2}}, K_{i,1} = v^{r_{i,1}}, K_{i,2} = v^{r_{i,2}})$를 생성한다.

서버는 주어진 트랩 도어 T에 대해서 $C'/(e(C_0, K_0)/\prod_{i \in [1,l]} e(C_{i,1}, K_{i,1}) e(C_{i,2}, K_{i,2}))$를 계산하여 자료가 키워드와 일치하는지의 여부를 확인한다.

이 시스템은 인덱스를 생성할 때 키워드를 변형하여 다양한 부가 검색 기능이 가능하도록 설계되었다. 예로 Conjunctive 비교 검색을 위해서는 다음과 같이 키워드를 선택한다. 사용자가 임의의 정수들 $a_1, a_2, ..., a_l \in [1, n]$에 대한 conjunctive 검색을 하고자 할 때, 키워드를 총 $l \times n$ bit로 선택하고 이를 l개의 n bit 블록으로 생각한다. 그 후 i번째 블록의 a_i번째 bit만을 1로 표현하고 나머지 bit는 모두 0을 입력하여 인덱스를 생성한다. 트랩 도어 또한 동일한 키워드에 대해서 생성하면 검색 결과는 모든 키워드 bit



(그림 4) Conjunctive 비교 검색을 위한 키워드

가 일치하는 자료만을 출력하므로 원하는 검색 결과를 얻을 수 있다. (그림 4)는 conjunctive 비교 검색을 위한 키워드의 한 예이다.

Subset 검색과 범위 검색 등에 대해서도 키워드를 조정하여 원하는 검색 결과를 쉽게 얻을 수 있다.

V. 결론

검색 가능 암호 시스템은 사용자가 암호화된 자료를 복호화하지 않고도 원하는 키워드를 포함하는 자료를 검색할 수 있도록 고안된 암호 기반 기술이다. 검색 가능 암호 시스템은 개인의 정보가 외부 저장 공간에 저장되면서 발생하는 문제점에 대한 해결 방법으로 지금까지 많은 연구가 진행되었다.

안전성 측면에서는 IND-CKA2의 안전성이 제안되고 또 이를 만족하는 시스템이 고안되면서 어느 정도 마무리 되었다고도 할 수 있다. 현재 제안되는 모든 검색 가능 암호 시스템은 기본적으로 IND-CKA2의 안전성이 요구된다.

앞으로의 연구 방향은 주로 효율성에 대한 것으로 대칭키 기반의 시스템의 경우 저장량과 계산 속도를 최적화하는 것이다. 반면 공개키 기반의 시스템의 경우는 범위 검색, conjunctive 검색 등의 부가 기능을 추가하는 것과 함께 실용적인 검색 속도를 지니는 시스템을 설계하는 것이라 할 수 있다.

약 어 정 리

CKA	Chosen Keyword Attack
CKA2	Adaptive Chosen Keyword Attack
IND	Indistinguishable
SSE	Symmetric-key Searchable Encryption

참 고 문 헌

[1] P. Golle, J. Staddon, and B. Waters, “Secure Conjunctive Keyword Search over Encrypted Data,” *In Applied Cryptography and Network Security Conference*, 2004.

[2] B. Waters, D. Balfanz, G. Durfee, and D. Smetters, “Building an Encrypted and Searchable Auditlog,” NDSS, 2004.

[3] R. Ostrovsky and W. Skeith, “Private Searching on Streaming Data,” Crypto 2005.

[4] J. Bethencourt, H. Chan, A. Perrig, E. Shi, and D. Song, “Anonymous Multi-Attribute Encryption with Range Query Conditional Decryption,” Technical Report, C.M.U. 2006.

[5] M. Abdalla, M. Bellare, D. Catalano, E. Kiltz, T. Kohno, T. Lange, J. Malone-Lee, G. Neven, P. Pailier, and H. Shi, “Searchable Encryption Revisited: Consistency Properties, Relation to Anonymous IBE, and Extensions,” Crypto 2005.

[6] D. Boneh, G. Crescenzo, R. Ostrovsky, and G. Persiano, “Public Key Encryption with Keyword Search,” Eurocrypt 2004.

[7] R. Ostrovsky, “Software Protection and Simulations on Oblivious RAMs,” *ACM Symp. on Theory of Computing*, 1990.

[8] P. Golle and R. Ostrovsky, “Software Protection and Simulation on Oblivious RAMs,” *Journal of ACM*, Vol.43, No.3, 1996, pp.431-473.

[9] D. Song, D. Wagner, and A. Perrig, “Practical Techniques for Searching on Encrypted Data,” *IEEE Symp. on Security and Privacy*, 2000.

[10] E.J. Goh, “Secure Indexes,” Technical Report 2003/216, IACR ePrint Crpytography Archive, 2003.

[11] Y.C. Chang and M. Mitzenmacher, “Privacy Preserving Keyword Searches on Remote Encrypted Data,” *In Applied Cryptography and Network Security Conf.*, 2005.

[12] R. Curtmola, J. Garay, S. Kamara, and R. Ostrovsky, “Searchable Symmetric Encryption: Improved Definitions and Efficient Constructions,” ACMCCS, 2006.

[13] D. Boneh and B. Waters, “Conjunctive, Subset and Range Queries on Encrypted Data,” *Theory of Cryptography Conf.*, 2007.