STTP를 활용한 공평한 비밀 매칭

(Fair Private Matching with Semi-Trusted Third Party)

김 이 용 † 홍 정 대 ††
(E-yong Kim) (Jeongdae Hong)
천 정 희 ††† 박 근 수 ††††
(Jung Hee Cheon) (Kunsoo Park)

요 약 비밀 매칭(Private Matching)은 각기 다른 두 참여자가 가진 데이타의 교집합을 구하는 문제이다. 이때 각 참여자는 교집합은 공유하되 그 이상의 정보는 감춰지기를 원한다. 2004년 Freedman 등[1]은 한 참여자만 비밀 매칭을 얻는 방법을 제안하였다. 경쟁관계의 회사와 같이 프로토콜 참여자가 동시에 Private Matching을 얻고자 하는 경우에는 Kissner와 Song[2]이 제안한 다자간의 교집합 계산 방법을 고려할 수 있다. 우리는 Kissner와 Song의 다항식 덧셈에 의한 교집합 계산 방법에 상당히 신뢰할 수 있는 제3자(Semi-Trusted Third Party)를 도입하여, 보다 효율적으로 비밀 매칭을 구할 수 있는 방법을 제안한다. 한편, STTP의 저장능력을 활용할 경우 프로토콜을 다시 시작하지 않고도 비밀 매칭을 업데이트 할 수 있는 방법 또한 제공한다.

키워드 : 비밀 매칭, 상당히 신뢰할 수 있는 제 3자, 준동형 암호시스템

Abstract Private Matching is the problem of computing the intersection of private datasets of two parties without revealing their own datasets. Freedman et al. [1] introduced a solution for the problem, where only one party gets private matching. When both parties want to get private matching simultaneously, we can consider the use of Kissner and Song [2]'s method which is a privacy-preserving set intersection with group decryption in multi-party case. In this paper we propose new protocols for fair private matching. Instead of group decryption we introduce a Semi-Trusted Third Party for fairness. We also propose an update procedure without restarting the PM protocol.

Key words : Private matching, semi-trusted third party, homomorphic cryptosystem

1. 서 론

서로 다른 데이타 집합을 가지고 있는 두 개의 조직이나 기관이 서로 자신의 집합을 노출하지 않으면서 공통의 데이타를 찾는 일을 비밀 매칭(Private Matching, 이하 PM)이라 한다[3,4]. 보험회사간 보험사기자 적발 또는 항공회사가 정부기관과 협조하여 탑승금지자 목록(do-not-fly list)을 만들고자 할 때 PM은 사용될 수 있다.

Freedman 등은 프로토콜 결과 둘 중 한 참여자만이 PM을 얻는 방법을 제안하였다[1]. Kissner와 Song은 그룹 복호화 기법[5]을 사용하여 다자간에 집합 연산을 수행할 수 있는 방법을 제안하였다[2].

PM에 참여하는 조직이 경쟁관계에 있는 회사나 조직이라면 먼저 또는 일방적으로 결과를 얻은 한쪽은 그 결과를 이용하여 어떤 유리한 행동을 할 수도 있다. 따라서 PM에 참여하는 조직은 누구나 먼저 또는 일방적인 결과를 얻기를 원할 것이다. 프로토콜 참여자 모두 동일한 시간에 PM을 얻기 위한 이상적인 방법은 완전히 신뢰할 수 있는 제3자(Trusted Third Party, 이하 TTP)의 도움을 받는 것이다. 그러나 TTP를 사용하는 방법은 TTP에게 PM의 결과가 노출되는 등의 문제로 인하여 실제 상황에서는 적용이 곤란하다.

TTP 대신에 상당히 신뢰할 수 있는 제3자가 있어서 자신은 PM의 결과를 모르면서 참여자들이 동시에 결과를 얻을 수 있도록 도와 준다면 프로토콜 참여 당사자도 이를 수용할 수 있을 것이다. 다자간 연산에서 이러한 참여자를 Semi-Trusted Third Party(이하 STTP)라 부른다. 이러한 STTP는 전자상거래, 부인방지 프로토콜 등에서 이미 사용되었다. Franklin과 Reiter[6]는 프로토콜의 적극적인 중개자로 사용하였으며, Bao 등은 프로토콜에 적절하지 않은 행동을 하는 참여자가 있을 때만 개입하는 Off-line TTP를 사용하였다[7].

기존 방법[1,4]에서는 프로토콜을 시작하는 참여자만

· 이 논문은 2007 한국컴퓨터종합학술대회에서 'STTP를 활용한 공평한 비밀 매칭'의 제목으로 발표된 논문을 확장한 것임

† 학생회원 : 서울대학교 전기,컴퓨터공학부
eykim@theory.snu.ac.kr
†† 비 회 원 : 서울대학교 전기,컴퓨터공학부
jdhong@theory.snu.ac.kr
††† 비 회 원 : 서울대학교 수리과학부 교수
jhcheon@snu.ac.kr
†††† 정 회 원 : 서울대학교 전기,컴퓨터공학부 교수
kpark@theory.snu.ac.kr

논문접수 : 2007년 9월 28일
심사완료 : 2008년 3월 24일

정보과학회논문지: 컴퓨팅의 실제 및 레터 제14권 제4호(2008.6)

PM을 얻는 반면, 우리는 공평하게 PM을 얻을 수 있는 새로운 방법을 제안한다. 우리의 프로토콜은 Kissner와 Song의 다항식 덧셈에 의한 교집합 계산 방법을 기본 도구로 사용하고 그룹 복호화 기법 대신에 STTP를 도입하여 참여자 모두가 동시에 PM을 얻는다. 또한, 참여자에게 새로운 원소가 추가되었을 때 프로토콜을 처음부터 시작하지 않고 STTP가 저장하고 있는 데이타를 이용하여 PM을 갱신(update)할 수 있는 방법을 제공한다.

2. 용어의 정의 및 공격자 모델

2.1 용어의 정의

클라이언트는 고유의 데이타 집합을 가지고 프로토콜에 참여하며, STTP 서버는 클라이언트들의 중간계산 결과를 받아 적절한 연산 후 중재한다. 서버는 프로토콜의 규칙을 잘 따르며 어느 클라이언트와도 공모하지 않으나 중간계산 결과를 저장할 수 있다.

평문의 정의역을 R(e.g., $\mathbb{Z}_N$)이라고 하고, R의 부분집합 P에서 데이타 집합의 원소를 뽑는다. 단, P의 원소들은 R에 고르게 분포되어 있고 그 성질이 특이하여 R에서 임의로 선택한 원소가 P에 포함될 확률은 매우 낮다. 우리는 Kissner와 Song이 제시한 $a \| h(a)$꼴의 원소를 집합 P로 사용한다. 이때 h(·)는 암호학적 해쉬함수이다.

환(ring) R에서 계수를 선택한 다항식 집합을 다항식환(polynomial ring) $R[x]$라고 할 때, $R[x]$에서 선택한 다항식 $f \in R[x]$는 아래와 같이 표현할 수 있다.

$$f(x) = \sum_{i=0}^{\deg(f)} f[i]x^i,$$

여기에서 $f[i]$는 x^i의 계수이며, $\deg(f)$는 다항식의 차수를 의미한다. 이 논문에서는 데이타 집합을 표시하기 위하여 다항식을 이용한다. 주어진 집합 $X \subseteq P$에 대하여 X의 다항식 표현은 $f(x) = \Pi_{x_i \in X}(x - x_i)$ 이다. 여기에서 $f \in R[x]$이다. 반대로 주어진 다항식 $f \in R[x]$로 표현되는 집합은 $X = \{x_i \mid f(x_i) = 0 \wedge x_i \in P\}$이다.

2.2 공격자 모델과 안전성

프로토콜의 공격자 모델은 Goldreich의 준정직한(Semi-Honest) 모델과 악의적인(Malicious) 모델을 따른다[8]. 준정직한 모델에서 참여자는 연산과정의 중간계산결과를 저장하는 행위를 제외하고는 모든 정해진 규칙을 잘 따른다. 악의적인 모델에서 참여자는 프로토콜이 의도하는 이상의 정보를 얻기 위해 자신의 능력 범위 내에서 규칙을 따르지 않는 어떠한 행동도 할 수 있다.

준정직한 모델에서 안전성은 어떠한 참여자 또는 공모자도 프로토콜에서 의도하는 이상의 결과를 얻지 못하면 만족하는 것으로 본다. STTP는 클라이언트들의 데이타를 알 수 없어야 하고, 각 클라이언트는 PM 이외의 상대방 정보를 얻을 수 없어야 한다.

악의적인 모델에서 안전성은 악의적인 행동을 하는 클라이언트의 결과값이 TTP가 있는 이상적인 모델에서 클라이언트가 얻게 되는 결과값과 구분할 수 없이 같도록 하는 시뮬레이터(simulator)를 만들 수 있으면 안전한 것으로 본다.

3. 배경지식

3.1 Paillier 암호시스템

암호시스템은 Paillier 스킴을 사용한다[9]. Paillier 스킴은 다음과 같은 준동형(homomorphic) 특성을 가지고 있다.

- 두 개의 서로 다른 암호문 $E_{pk}(a)$, $E_{pk}(b)$가 주어졌을 때 $E_{pk}(a+b) := E_{pk}(a) +_h E_{pk}(b)$를 계산할 수 있다.
- 임의의 상수 c와 암호문 $E_{pk}(a)$가 주어졌을 때 $E_{pk}(c \cdot a) := c \times_h E_{pk}(a)$를 계산할 수 있다.

여기에서 $+_h$와 $\times_h$는 두 암호문의 조합을 의미하며, 암호문 $E_{pk}(a+b)$와 $E_{pk}(c \cdot a)$는 re-randomized 된다. 한편, 다항식 $f(x)$의 암호문을 계수들의 암호문 순서쌍

$$E_{pk}(f(x)) := \{E_{pk}(f[0]), \ldots, E_{pk}(f[\deg(f(x))])\}$$

으로 정의할 때 다음의 연산들을 수행할 수 있다.

- 암호화된 다항식 $E_{pk}(f_1)$과 $E_{pk}(f_2)$가 주어졌을 때 $E_{pk}(g) := E_{pk}(f_1 + f_2)$를 계산할 수 있다.
- 암호화된 다항식 $E_{pk}(f_1)$과 암호화되지 않은 다항식 f_2가 주어졌을 때 $E_{pk}(g) := f_2 \times_h E_{pk}(f_1)$를 계산할 수 있다.
- 암호화된 다항식 $E_{pk}(f)$에 암호화되지 않은 어떤 값 $a \in R$가 주어졌을 때 $E_{pk}(f(a))$를 계산할 수 있다.

3.2 다항식 덧셈에 의한 교집합 계산

Kissner와 Song은 두 다항식을 더하여 그들의 최대공약수에 해당되는 항들로부터 교집합을 구하는 새로운 집합 연산 방법을 제안하였다. 데이타 집합 X와 Y를 나타내는 다항식을 각각 f와 g라고 할 때, 교집합 $X \cap Y$는 $r,s \leftarrow R^{\deg(f)}[x]$에 대하여 $f \cdot r + g \cdot s$ 형태로 나타낼 수 있다. 여기에서 $R^{\deg(f)}[x]$는 차수가 0에서 $\deg(f)$이고 계수는 R에서 고르게 독립적으로 선택된 모든 다항식의 집합이다. [2]의 보조정리 3.1은 이것을 증명하고 있다.

4. 공평한 비밀 매칭 프로토콜

4.1 준정직한 클라이언트에 대한 프로토콜

클라이언트 A와 B의 데이타 집합을 각각 X, Y라고

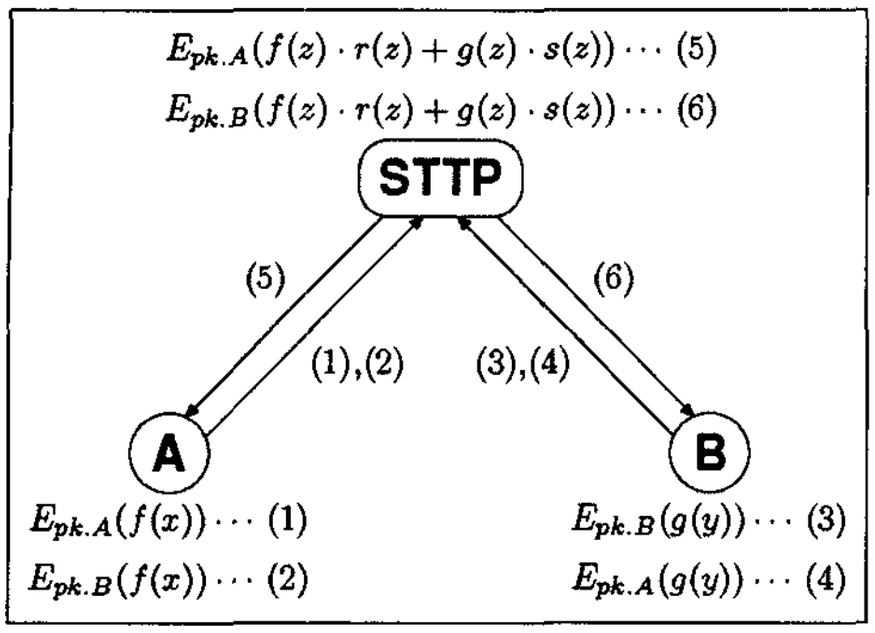

그림 1 FPM-SH 프로토콜

하고, $|X|=|Y|=k$라고 하자. A와 B는 그림 1과 같이 각자 자신의 데이타 집합을 표현하는 다항식 $f(x)$와 $g(y)$를 자신과 상대방의 공개키를 이용하여 암호화한다. 즉, A는 $E_{pk.A}(f(x))$와 $E_{pk.B}(f(x))$를, B는 $E_{pk.B}(g(y))$와 $E_{pk.A}(g(y))$를 계산한다. 각 클라이언트는 암호화된 다항식들을 서버에게 전송한다. 서버는 클라이언트로부터 받은 암호화된 다항식으로부터 다항식의 차수 k를 알 수 있다. 서버는 k보다 크거나 같은 임의의 다항식 r,s를 고르게 선택하여 각 다항식에 곱한 후 그 결과를 더하여 이를 복호화할 수 있는 개인키를 가진 클라이언트에게 각각 전송한다. 즉, A에게는 $E_{pk.A}(f(z)\cdot r(z)+g(z)\cdot s(z))$를, B에게는 $E_{pk.B}(f(z)\cdot r(z)+g(z)\cdot s(z))$를 보낸다. A와 B는 각각 자신의 데이타 집합 X,Y의 모든 원소를 대입한 후 복호화하였을 때 그 값이 '0'인 값을 PM으로 결정한다. 자세한 내용은 프로토콜 4.1에 기술되어 있다.

프로토콜 4.1 FPM-SH

입력: 클라이언트 A의 집합 $X=\{x_1,\ldots,x_k\}$
클라이언트 B의 집합 $Y=\{y_1,\ldots,y_k\}$
출력: $X\cap Y$

1. 클라이언트 A, B
   (a) 각자 자신의 데이터 집합을 표현하는 다항식 $f(x),g(y)$를 만든다.
   $f(x)=(x-x_1)\cdots(x-x_k)=\sum_{i=0}^{k}f[i]x^i$
   $g(y)=(y-y_1)\cdots(y-y_k)=\sum_{i=0}^{k}g[i]y^i$
   (b) 준동형 암호시스템 $\langle E,D\rangle$에 사용할 공개키 $pk.A$와 $pk.B$를 서버를 통해 주고받는다.
   (c) 클라이언트 A는 $E_{pk.A}(f(x))$와 $E_{pk.B}(f(x))$를, B는 $E_{pk.B}(g(y))$와 $E_{pk.A}(g(y))$를 계산하여 서버에게 전송한다.
   $E_{pk.A}(f(x))\cdots(1)$ $E_{pk.B}(f(x))\cdots(2)$
   $E_{pk.B}(g(y))\cdots(3)$ $E_{pk.A}(g(y))\cdots(4)$
2. 서버 $STTP$
   (a) 서로 다른 2개의 다항식 $r,s\leftarrow R^k[x]$를 선택하여 식 (1)과 (2)에는 r을 곱하고, 식 (3)과 (4)에는 s를 곱하여 이 값들을 서로 더한다.
   $E_{pk.A}(f(z)\cdot r(z)+g(z)\cdot s(z))\cdots(5)$
   $E_{pk.B}(f(z)\cdot r(z)+g(z)\cdot s(z))\cdots(6)$
   (b) 클라이언트 A에게는 (5)를, B에게는 (6)을 각각 전송한다.
3. 클라이언트 A, B
   (a) 클라이언트 A는 (5)에 집합 X의 원소들을, B는 (6)에 집합 Y의 원소들을 대입하여 각각 자신의 개인키로 복호화한다.
   (b) 결과값이 '0'인 원소를 PM에 포함시킨다.

4.2 악의적인 클라이언트에 대한 프로토콜

우리는 자신만 정확한 PM을 얻고 상대방에게 잘못된 PM을 얻게 하거나 상대방으로부터 PM 이상의 정보를 얻으려는 악의적인 클라이언트를 차단하는 프로토콜을 제안한다.

그림 1의 프로토콜에서 만약 A가 자신의 공개키로는 $f(x)$를 암호화하고, B의 공개키로는 다른 다항식 $\bar{f}(x)$를 암호화한다면, A는 정확한 PM을 얻게 되지만 B는 잘못된 결과를 얻게 된다. 또한 A가 다항식의 모든 계수를 0으로 하여 전송하면 자신은 정확한 PM을 얻을 수 있지만 B는 모든 원소를 PM으로 착각하게 된다.

이를 막기 위한 방법은 다음과 같다. 우선 프로토콜 4.1과 동일하게 A는 $E_{pk.A}(f(x))$와 $E_{pk.B}(f(x))$를, B는 $E_{pk.B}(g(y))$와 $E_{pk.A}(g(y))$를 계산하여 서버에게 전송한다. [2]에서와 마찬가지로 최고차항의 계수를 1로 제한하여 모든 계수가 0이 될 수 없도록 한다. 이 때 최고차항에 대한 계수는 보내지 않아도 되기 때문에 암호화하여 보내는 다항식 계수의 개수는 다항식의 차수보다 하나 작아진다.

서버는 A와 B로부터 받은 암호화된 다항식에 랜덤다항식 t,u를 선택하여 각각 곱한 후 임의의 값 λ,μ를 각각 대입하여 이를 복호화가 가능한 클라이언트에게 보낸다. 즉, $E_{pk.A}(f(\lambda)\cdot t(\lambda))$와 $E_{pk.A}(g(\mu)\cdot u(\mu))$를 A에게, $E_{pk.B}(f(\lambda)\cdot t(\lambda))$와 $E_{pk.B}(g(\mu)\cdot u(\mu))$를 B에게 보낸다. 각 클라이언트는 복호화한 값을 서버에게 돌려준다. A와 B로부터 받은 값들이 서로 일치하면 A가 암호화한 다항식은 동일한 것임을 확인할 수 있고, B에 대해서도 마찬가지이다. 클라이언트들이 보낸 다항식에 이상이 없으면 이후의 프로토콜은 프로토콜 4.1과 동일하게 진행한다.

4.3 FPM-SH 갱신 프로토콜

프로토콜 수행 후 클라이언트의 데이타 집합에 새로운 원소가 추가된 경우에도 유효한 PM을 유지하려면 갱신 작업이 필요하다. 우리는 STTP가 중간 계산결과

를 저장할 수 있는 특성을 활용하여 프로토콜을 처음부터 다시 시작하지 않으면서 상대적으로 적은 계산 및 통신량으로 PM을 갱신할 수 있는 방법을 제안한다.

그림 2와 같이 A는 추가 데이타로 다항식 $f'(x)$을 만든다. 그리고 기본 프로토콜에서와 동일하게 자신의 공개키와 B의 공개키로 암호화하여 서버에게 전송한다. 서버는 이전 프로토콜 수행에서 B로부터 받은 다항식을 활용하여 (3)과 (4)를 계산하고 이를 각 클라이언트에게 보낸다. 이후 A는 추가 데이타를, B는 PM이 아닌 원소를 대입하여 그 값이 '0'인 원소를 PM에 추가한다. 추가하는 원소의 개수가 $k' < k$라고 하면 클라이언트와 서버간 교환하는 암호문의 개수는 다항식의 차수와 일치하므로 통신량도 줄어들게 된다.

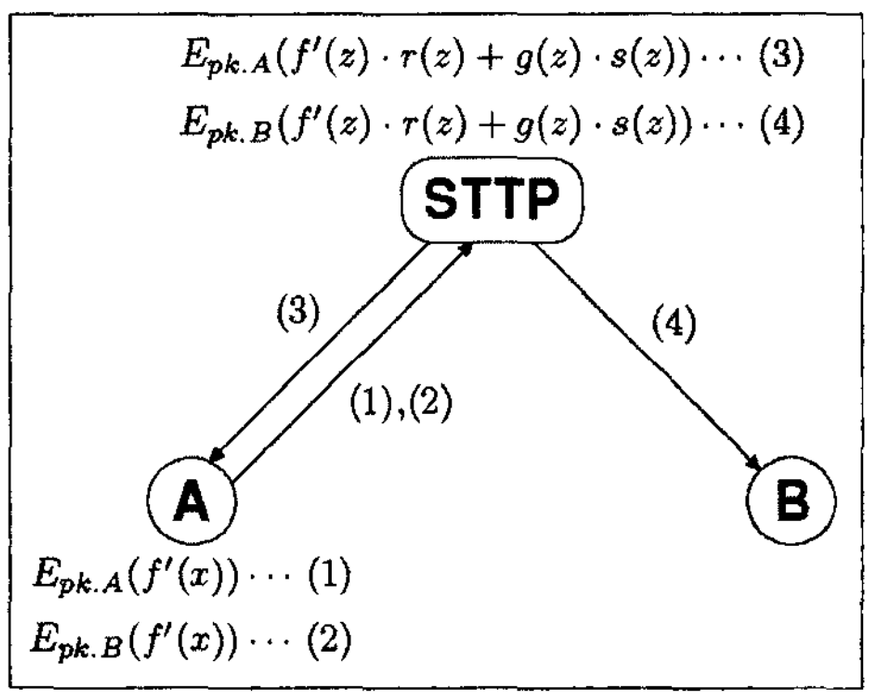

그림 2 FPM-SH 갱신 프로토콜

5. 안전도 분석 및 효율성

5.1 안전도 분석

FPM-SH 우리는 먼저 프로토콜이 정확함을 보인다. 다음으로 서버 STTP는 중간 계산결과로부터 클라이언트들의 입력을 알아낼 수 없으며, 클라이언트 A와 B는 준정직한 모델에서 안전함을 보인다.

보조정리 5.1 (정확성). 클라이언트 A와 B는 높은 확률로 정확한 PM을 얻을 수 있다.

증명. 프로토콜 결과 A와 B는 동일한 다항식 $p(z) = f(z)\cdot r(z)+g(z)\cdot s(z)$을 갖는다. 만약 $a \in X \cap Y$이면 $f(a)=g(a)=0$에 의해 $p(a)=0$이 된다. 한편, 보조정리 3.1에 의해 다항식 $p(z)$의 근은 PM이다. □

보조정리 5.2 (클라이언트의 서버에 대한 보안성). 서버 STTP는 중간 계산결과로부터 클라이언트들의 원소를 구분할 수 없다.

증명. 프로토콜 4.1에서 사용되는 Paillier 암호시스템은 semantically secure하다[10]. 서버가 얻게 되는 정보는 오직 이 암호시스템에 의해 암호화된 다항식들이므로 다항식의 차수 k 이외에 어떤 정보도 얻을 수 없게 된다. □

이제 준정직한 모델에서 각 클라이언트는 PM 이외의 상대방 정보는 아무것도 얻을 수 없음을 보인다. 다음 보조정리 5.3은 클라이언트들이 프로토콜 수행 후 얻게 되는 PM이 이상적인 모델에서 TTP의 중재로 얻게 되는 PM과 구분할 수 없이 같음을 보인다.

보조정리 5.3 (클라이언트 상호간의 보안성). 어떤 확률적인 다항시간 연산을 하는 준정직한 클라이언트도, TTP가 존재하는 이상적인 모델에서 같은 데이타 집합을 가지고 얻게 되는 정보 이상의 것을 프로토콜 4.1이 돌아가는 실제 모델에서 얻을 수 없다.

증명. 이상적인 모델에서 각 클라이언트는 TTP로부터 PM을 얻는다. 한편, 실제 모델에서 각 클라이언트는 서버로부터 $p(z) = f(z)\cdot r(z)+g(z)\cdot s(z)$를 얻게 된다. 여기에서 r,s는 서버가 랜덤하게 선택하였기 때문에 각 클라이언트의 관점에서 볼 때 $R^k[x]$ 상에 고르게 분포하고 그 내용을 알 수도 없다. 보조정리 3.1에 의해 각 클라이언트는 상대방 데이타 집합에 대해서 PM에서 얻을 수 있는 것 이외에 어떠한 정보도 p로부터 얻을 수 없다. □

FPM-MAL 우리는 실제 모델에서 악의적인 클라이언트 A^*의 어떤 행동에 대해서도 TTP가 존재하는 이상적인 모델에서 A^*가 따를 수 있는 행동으로 변환할 수 있는 시뮬레이터 S를 만들 수 있음을 보인다. A^*의 관점에서 두 행동은 계산적으로 구분할 수 없고, 따라서 TTP로부터 얻게 되는 PM 이외에 어떤 정보도 A^*는 얻을 수 없음을 보이게 된다.

정리 5.4 실제 모델에서 악의적인 행동을 하는 클라이언트 A^*가 얻게 되는 PM과 TTP가 있는 이상적인 모델에서 클라이언트들이 얻게 되는 PM이 구분할 수 없이 같도록 하는, 이상적인 모델에서 작동하는 시뮬레이터 S를 만들 수 있다.

증명(스케치). TTP가 있는 이상적인 모델에서 시뮬레이터 S가 A^*와 그림 3과 같은 프로토콜을 수행하면 A^*는 실제 모델과 이상적인 모델에서 모두 같은 결과를 얻으며, 실제 모델에서 STTP와 통신을 하는지 이상적인 모델에서 S와 통신을 하는지 구분하지 못한다. □

5.2 효율성

표 1은 양자간에 각 프로토콜을 수행할 경우 클라이언트당 필요한 라운드수 및 총통신량을 비교한 것이다. 여기에서 k는 각 클라이언트의 원소 개수, k'은 추가되는 원소의 개수, 그리고 P는 원소의 정의역이다. FPM-SH 갱신 프로토콜의 경우 프로토콜을 시작한 클라이언트(A)에 대해서는 1라운드가 필요하지만 다른 클라이언트(B)에 대해서는 0.5라운드가 필요하다.

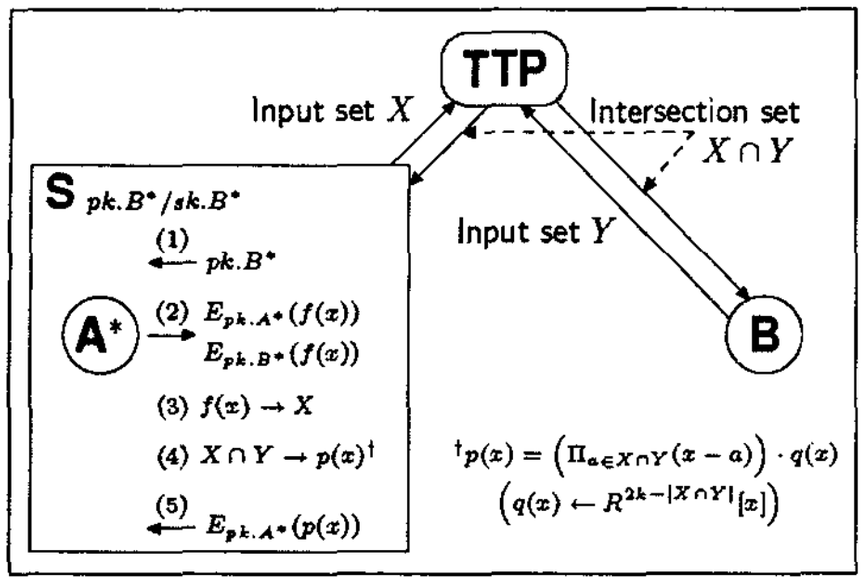

그림 3 시뮬레이션 프로토콜

표 1 프로토콜에 따른 총통신량 비교

프로토콜	FPM SH	FPM SH 갱신	Set Intersection (HBC) [2]
라운드수	1	1 / 0.5	3
총통신량 (bit 길이)	$4k \log \mid P \mid$	$2(k+k')\log \mid P \mid$ / $2k \log \mid P \mid$	$10k \log \mid P \mid$

표 2는 양자간에 각 프로토콜을 수행할 경우 연산량을 비교한 것이다. 여기에서 t_E를 암호화 시간, t_D를 복호화 시간, t_{+_h}을 준동형 덧셈 시간, $t_{\times_h}$를 준동형 곱셈 시간, 그리고 t_{GD}를 그룹 복호화 시간이라고 하자. 그룹 복호화는 일반 복호화에 비하여 훨씬 많은 연산 시간을 필요로 한다.

표 2 프로토콜에 따른 연산량 비교

프로토콜	연산량
FPM	$4t_E + 2t_D + 2t_{+_h} + 4t_{\times_h}$
FPM-SH	$2t_E + 2t_D + 2t_{+_h} + 4t_{\times_h}$
Set-Intersection (HBC) [2]	$2t_E + t_{GD} + 3t_{+_h} + 4t_{\times_h}$

6. 결 론

우리는 Kissner와 Song의 다항식 덧셈을 이용한 교집합 계산 방법에 STTP를 도입하여 양자간에 공평하게 PM을 구하는 방법을 제안하였다. STTP가 별도의 저장 서버를 운용한다면 적은 계산량으로 PM의 갱신도 가능함을 보였다. 우리의 방법은 먼저 PM을 얻은 쪽이 상대방에게 잘못된 결과 또는 아무것도 얻지 못하도록 하려는 행동을 막을 수 있다. 한편, 우리의 프로토콜을 다자간으로 확장하는 문제는 효율성 측면에서 추가적인 연구가 필요하다.

참 고 문 헌

[1] M. J. Freedman, K. Nissim, and B. Pinkas. "Efficient private matching and set intersection," in Advances in Cryptology-EUROCRYPT 2004, ser. Lecture Notes in comput. Sci., vol. 3027. Springer-Verlag, 2004, pp. 1-19.

[2] L. Kissner and D. Song, "Privacy-preserving set operations," in Advances in Cryptology-CRYPTO 2005, ser. Lecture Notes in Comput. Sci., vol. 3621. Springer-Verlag, 22005, pp. 241-257.

[3] R. Agrawal, A. Evfimievski, and R. Srikant, "Information sharing across private databases," in Proc. ACM SIGMOD 2003, San Diego, CA, Jun. 2003, pp. 86-97.

[4] Y. Li, J. D. Tygar, and J. M. Hellerstein, "Private matching," Intel Research, Tech. Rep. IRB-TR-04-005, Feb. 2004.

[5] P. Fouque, G. Poupard, and J. Stern, "Sharing decryption in the context of voting of lotteries," in Proc. International Conference on Financial Cryptography (FC '00), ser. Lecture Notes in Comput. Sci., vol. 1962. Springer-Verlag, 2000, pp. 90-104.

[6] M. K. Franklin and M. K. Reiter, "Fair exchange with a semi-trusted third party," in Proc. ACM Conference on Computer and Communications Security (CCS '97), Zurish, Switzerland, Apr. 1997, pp. 1-5.

[7] F. Bao, R. H. Deng, and W. Mao, "Efficient and practical fair exchange protocols with off-line TTP," in Proc. IEEE Symposium on Security and Privacy, Oakland, CA, May 1998, pp. 77-85.

[8] Goldreich, Foundations of Cryptography. Cambridge University Press, 2004, vol. 2.

[9] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in Advances in Cryptology-EUROCRYPTO '99, ser. Lecture Notes in Comput. Sci., vol. 1592. Springer-Verlag, 1999, pp. 223-238.