通信情報保護學會誌
第5卷 第1號, 1995. 3

멀티파티 프로토콜에 대한 소고

양 형 규**, 박 성 준*, 원 동 호*

1. 서 론

분산 시스템의 빠른 발전은 특히 폴트가 발생했을 때 통신참가자에 의해서 수행될 수 있는 업무가 무엇인지에 대한 자연스런 의문이 제기되었다. 통신참가자들의 계산 능력이 무한한지 혹은 제한됐는 지에 의존하면서, 이러한 의문에 대한 해결책으로 두 가지의 접근이 있었다.

Diffie와 Hellman에 의한 암호학적 접근은 통신참가자들이 계산적으로 제한되었다는 것과 어떤 (one-way) 함수의 존재를 가정하는 것이었다[1]. 이러한 간단한 가정은 두 명의 통신참가자사이에 안전하게 메세지를 교환하는 업무를 실행할 수 있도록 하기 위해서 Diffie와 Hellman에 의해서 고려되었으며, 광범위하게 사용되었다.

최근 몇년사이에 위의 가정을 기본으로 하는 프로토콜이 contract signing, secret exchange, joint coin flipping, voting, 그리고 playing Poker와 같은 좀더 복잡한 업무들을 실행할 수 있도록 제안되었다. 이러한 결과들은 영지식 증명을 이용해서 NP-complete 문제의 존재를 이자간 (two party) 계산과 멀티파티 계산에 대한 완전성 이론으로 변형시켰다. Goldreich 및 Micali와 Wigderson은 만약 non-uniform 일방향 함수가 존재한다면, n개의 입력을 가진 모든 확률적 함수는 다음과 같은 방식으로 계산 능력이 제한된 n명의 통신참가자에 의해서 계산된다는 것을 보였다[2].

(1) 만약 어떠한 폴트도 발생하지 않는다면, 통신참가자의 어떠한 부분 집합도 임의의 다른 추가적인 정보를 계산할 수 없다.

(2) 비록 Byzantine 폴트가 발생할지라도 크기가 $t < n/2$인 통신참가자의 집합은 함수의 계산을 방해할 수 없으며, 어떠한 정보도 얻지 못한다.

본 고에서는 멀티파티 구성시 기본이 되는 비밀 분산 프로토콜 혹은 임계치 방식에 대해 고찰하고 본 고에서 사용할 용어에 대한 정의와 멀티파티 프로토콜의 구성 요소에 대해 고찰하고 정의한다[3]. 그리고 Beaver가 제안한 resilience 개념을 이용하여 안전성과 신뢰성 즉, 멀티파티 프로토콜의 안전한 계산에 대해 고찰한다.

2. 비밀 분산 프로토콜

비밀 분산 프로토콜은 하나의 통신참가자 즉, 분배자(dealer)가 비밀 정보 s를 네트워크상의 모든 통신참가자에게 분배할 수 있게 하는 방식으로, 임의의 통신참가자들의 연합만이 s를 결정할 수 있다. 폴트 혹은 의심스런 통신참가자의 수가 t로 제한됐을 때, s를 재구성할 수 있는 조건은 $t +$

* 성균관대학교 정보공학과
** 강남대학교 전자계산학과

1 혹은 그 이상의 통신참가자가 있어야 하며, t 이하의 통신참가자는 재구성할 수 없다.

1979년 Shamir는 Lagrange의 다항식 보간법를 이용한 간단한 비밀 분산 방식을 제안하였다[4]. 이 방식을 간단히 설명하면 다음과 같다. 통신참가자의 수보다 큰 유한체를 E라 표시하자. 비밀 정보 $s \in E$를 소유한 분배자 D는 상수항이 s이고 E내에서 t개의 랜덤 계수 $a_1, \ldots, a_t$을 선택하여, 이를 계수로 갖는 t차 다항식 $p(u)$를 생성한다. 이때, $\alpha_1, \ldots, \alpha_n$을 선택하여, D는 분할정보 $PIECE_i(s)(=p(\alpha_i))$를 생성하여 비밀리에 통신참가자에게 전달한다. 이 방식의 절차는 그림 1.1과 같이 표시할 수 있다.

임의의 $t + 1$ 명의 통신참가자들은 서로의 분할정보를 교환하여 간단하게 s를 구할 수 있다. 왜냐하면 $t + 1$ 개의 점들이 t차인 다항식을 결정할 수 있기 때문이다. 그들 자신들의 점들을 통과하는 t차 다항식을 보간하므로써, 충분히 많은 통신참가자들의 집합은 $p(0) = s$를 재구성할 수 있다. 에러 혹은 분실된 분할정보들의 수가 $n - t$ 보다 적으면, 즉 $t + 1$ 개의 타당한 분할정보들이 남아 있는 한 s의 재구성은 가능하다. 그러므로 $n - t \geq t + 1$ 이 만족되어야 한다. 결과적으로, 프로토콜이 진행되는 시간동안에 믿을 수 있는 $t + 1$ 이상의 통신참가자들이 보유하고 있는 분할정보들로부터 비밀 정보 s를 재구성하는 것은 용이하다. 이 방식은 그림 1.2와 같다.

Share (s)

단계 1. 유한체 E, $|E| > n$, 그리고 원소들 $\alpha_1, \ldots, \alpha_n \neq 0$을 선택

단계 2. D는 랜덤하게 $a_t, \ldots, a_1 \in E$을 선택

단계 3. D는 t차인 다항식 $p(u) = a_t u^t + \ldots + a_1 u + s$를 계산

단계 4. D는 $PIECE_i(s) \leftarrow p(\alpha_i)$을 계산하고 $PIECE_i(s)$을 통신참가자 i에게 비밀리 송신

그림 1.1 D가 s ∈ E을 비밀리에 분배하는 프로토콜

Reconstruct Secret($PIECE_{i_1}(s), \ldots, PIECE_{i_{t+1}}(s)$)

단계 1. 개개의 통신참가자 i는 $PIECE_i(s)$를 통신참가자 j에게 송신

단계 2. 통신참가자 j는 자신이 받은 분할정보들 중에서 $t+1$개의 분할 정보를 선택 그리고 j는 $(i_1, PIECE_{i_1}(s)), \ldots, (i_{t+1}, PIECE_{i_{t+1}}(s))$을 통과하는 t차 다항식 $p(u)$를 보간법을 이용하여 계산

그림 1.2 통신참가자 j가 비밀 정보 s를 재구성하는 프로토콜

개념적으로 설명하면, t차 다항식에서 t 혹은 t 보다 적은 임의의 계산점들은 t차다항식을 계산할 수 없다. 그러므로 ADV는 t 혹은 t 보다 적은 통신참가자들의 협력에 의해서 얻어진 정보 즉, 분할정보들을 이용해서 비밀 정보에 대한 어떠한 정보도 얻을 수 없다. 이와 같은 Shamir의 방식을 (n, t)-비밀 분산 프로토콜이라 한다.

3. 멀티파티 프로토콜 구성 요소

멀티파티 프로토콜이란 상호간 통신할 수 있는 통신참가자들로 구성된 네트워크가 그들이 소유한 비밀 정보를 이용하여 임의의 함수를 안전하게 계산할 수 있다는 것을 의미하며, 다른 용어로 안전한 계산이라 한다. 다시 말하면, 임의의 ADV가 임의의 계산을 방해하려고 노력하더라도 통신참가자들은 자신들이 소유한 입력 값 즉, 비밀 정보에 대한 어떠한 정보도 누설하지 않고 임의의 함수에 대한 계산을 올바르게 계산할 수 있다. 멀티파티 프로토콜을 이용하여 해결할려고 시도한 기존의 문제를 살펴보면, [5]의 백만장자 문제, [6]의 디지탈 선거 문제, [7]의 정당성 문제, [8]의 동전 던지기 문제, [9]의 Oblivious transfer 문제, 그리고 [10]의 Mental Poker 문제 등이 있다. 위의 열거된 문제들의 공통된 문제점은 둘이상의 통신참가자가 참여한 계산이 올바르고 안전해야 한다는 점이다.

멀티파티 프로토콜과 관련한 기존의 정의들이 보편 타당하지 않으므로 안전한 프로토콜을 개발하는데에 많은 문제점이 야기됐다. 멀티파티 프로토콜에 대한 올바른 정의를 내려야지만 안전한 프로토콜로부터 얻고자 하는 특성에 대한 정확한 이해를 할 수 있으며 이러한 정의에 의존하여 안전한 프로토콜을 개발할 수 있다. 또한 제안된 프로토콜과 기존의 프로토콜과의 안전성을 비교할 수 있다.

본 고에서는 기존의 연구를 바탕으로 하여 멀티파티 프로토콜 구성시 필요한 구성 요소들에 대한 정의를 내리고자 한다[11]. 즉, 통신 참가자와 ADV의 계산 능력, 이용 가능한 통신 채널, 안전성을 얻기 위해서 만든 가정, ADV의 특성, 정확성, 비밀성에 대한 형식적인 정의를 내리고자 한다. 그리고 완전히 안전한 프로토콜(perfect secure protocol) 즉, 이상적 프로토콜(Ideal protocol)을 형식화하고자 한다. 이상적 프로토콜이란 n명의 통신참가자들이 각각 하나의 비밀 정보 x_i를 소유한 상태에서, 신뢰성있는 센타가 비밀리에 통신참가자들의 비밀 정보를 받아서 함수의 결과치를 제외한 어떠한 정보도 누설시키지 않고 유한하고 다중 값을 출력하는 확률적 함수 $F(x_1, \ldots, x_n)$를 계산할 수 있는 프로토콜을 말한다. 또한, 정직한 통신 참가자만이 참여하는 신뢰성있는 프로토콜, 수동 ADV가 존재할 때의 프로토콜, 그리고 비잔틴 ADV가 존재할 때의 프로토콜에 대한 실행 단계를 각각 형식화한다.

3.1 통신참가자

완전한 안전성을 얻기 위한 정보 이론적 모델을 사용하기 위해서 통신참가자와 ADV의 계산 능력을 제한하지 않는다. 따라서 일반성을 위해서 통신참가자는 유한 혹은 무한의 상태 집합을 가지는 오토마타로서 간주된다.

멀티파티 프로토콜에서의 통신참가자는 다음과 같이 정의할 수 있다.

■ **정의 3.1** 통신참가자는 튜플 (Q, X, Z, q^0, δ)로 구성된 상호 튜링 기계이다.

단, Q : 상태 집합

X : 표준 입력 집합, $X \in \Sigma^*$

Z : 임의의 입력 집합

$q^0 : X \times Z \times N \rightarrow Q$

$\delta : \delta : Q \times H \rightarrow \mathrm{dist}(Q \times H)$

dist(U)는 집합 U상에서의 모든 분포들의 집합을 의미하고, 확률 전이 함수(probabilistic transition function) δ는 현재 상태와 수신된 메세지 H를 입력으로 하여 임의의 분포를 갖는 새로운 상태와 새로운 메세지를 생성한다. 다시 말하면, 전이 함수는 확률적 튜링 기계의 계산 결과와 유사하게 임의의 분포를 갖는 출력을 생성한다. 임의의 입력 Z는 여러 명의 통신참가자가 상호간 소유하고 있는 정보를 의미하며, 그것의 일반적인 사용 목적은 단순히 프로토콜의 일부분으로서가 아니라 이전의 계산과 상호 통신에 의해서 얻어진 정보를 기록하기 위한 것이다. 주어진 프로토콜과 관련하여, 통신참가자가 항상 프로토콜에 따라서 계산한다면, 정직하다 라고 한다. 폴트란 통신참가자가 정직하지 않을 때를 말한다. 폴트가 된 통신참가자들은 시스템을 방해하기 위해서 그들의 정보를 공유할 수 있다.

라운드란 완전한 네트워크(complete network) 상에서 동작하는 동기 프로토콜에서 개개의 기계 P_i가 능동 상태에 들어가서, 임의의 계산을 하고 개개의 테이프 W_{ij}가 R_{ji}로 복사된 후 비능동 상태로 되돌아 가는 과정을 말한다.

기계 P_i의 계산 시간은 그것이 능동 상태에 있는 동안 행한 계산 시간의 합이다. 프로토콜의 메세지 복잡도는 상호간 교환한 모든 메세지의 총 길이이다.

3.2 프로토콜

3.2.1 네트워크와 통신 채널

채널이란 메세지 m을 입력으로 하여 분배 가능한 메세지들의 집합에 대한 임의의 분포를 생성하는 확률 함수 $C : \Sigma^* \rightarrow N \times N \times \Sigma^*$를 말한다. 분배 가능한 메세지는 (s, r, m)의 형태로 표시되며, $1 \leq s, r \leq n$이고 $m \in \Sigma^*$이다. 모든 채널 함수가 적용되고 그 결과 생성된 분포들의 집합은 분배되는 메세지들의 집합 M을 형성한다. $\{(s, r, m) \mid (s, r, m) \in M, 1 \leq r \leq n\}$으로 정의되는 서브집합 M_s는 통신참가자 r의 입력 테이프상에 위치한다. 만약 통신참가자 s가 자신의 통신 테이프상에 (m, C_{sr})을 쓰면, 채널 C_{sr}은 m을 입력으로 해서 (s, r, m)을 출력한다. 이 때, (s, r, m)은 통신참가자 r의 통신테이프상에 위치하게 된다.

통신참가자 s와 r을 연결하는 비밀 채널 C_{sr}는 입력 m에 대해서 $\{(s, r, m)\}$을 생성하는 함수이다. 이 비밀 채널은 오직 통신참가자 s와 r만이 사용하는 채널로서 제삼자는 송·수신되는 메세지의 내용을 알 수 없다. 광역 채널 C_s란 입력 m에 대해서 $\{(s, 1, m), (s, 2, m), \ldots, (s, n, m)\}$을 생성하는 함수이다. 광역 채널을 통해서 수신된 메세지는 모든 통신참가자들이 동시에 알 수 있다. 즉, 모든 통신참가자들이 볼 수 있는 채널을 의미한다.

■ **정의 3.2** 비밀 채널 C_{sr}이란 통신참가자 s와 r을 연결하는 채널로서 입력 m에 대해서 $\{(s, r, m)\}$을 생성하는 함수이다.

■ **정의 3.3** 광역 채널 C_s이란 입력 m에 대해서 $\{(s, 1, m), (s, 2, m), \ldots, (s, n, m)\}$을 생성하는 함수이다.

■ **정의 3.4** 네트워크는 $P_1, \ldots, P_n$인 통신참가자가 통신 채널상에서 메세지 m을 송수신하므로써 상호간 통신하는 채널들의 집합을 의미한다.

3.2.2 멀티파티 프로토콜

n명의 통신참가자로 구성된 멀티파티 프로토콜은 통신참가자, 채널, 출력 함수 그리고 출력의 집합으로 구성되며 다음과 같이 표시한다.

$$\langle\ \Pi\ \rangle = \{(P_1, C_1, O_1, Y_1), \ldots, (P_n, C_n, O_n, Y_n)\},$$

단, $O_i : Q_i \rightarrow Y_i$는 통신참가자 i의 마지막 상태를 Y_i의 출력 값으로 사상시키는 출력 함수이다.

이것은 프로토콜의 마지막에서 work 테이프 혹은 출력 테이프에 기록되는 내용으로 간주할 수 있다. 출력 집합은 임의의 비트 스트링의 집합으로 간주한다. 통신참가자 i는 $P_i = (Q_i, X_i, Z_i, q^0_i, \delta_i)$로 표시한다. C_i는 통신참가자 P_i가 자신의 메세지를 보내는데 사용하는 채널의 집합 $\{C^j\}_{j\in N}$이다.

3.2.3 정직한 통신참가자만이 참여한 프로토콜의 실행

모든 통신참가자가 정직할 때 프로토콜 실행동안 발생되는 일련의 실행 단계들을 표현하고자 한다. 여기서의 목적은 형식적인 표현보다는 프로토콜과 프로토콜의 실행에 대한 특징을 상술하는데 있다. 언급한 것처럼 프로토콜의 각 라운드동안 통신참가자는 계산을 수행하고 채널을 통해서 메세지를 송신한다.

먼저, 사용할 표기에 대해 설명하면, $data^{out}(U,V,r)$은 라운드 r동안 집합 U에 있는 통신참가자가 집합 V에 있는 통신참가자에게 보내는 메세지의 집합을 의미하며, $data(U,V,r)$는 집합 U에 속해있는 통신참가자로 부터 집합 V에 속해 있는 통신참가자에게 실질적으로 송신될 메세지 즉, $data^{out}(U,V,r)$을 수행하는 여러 채널들에 따라서 생성되는 메세지를 말한다. $D(U,V,r) = data(U,V,r) \cup data(V,U,r)$는 U에서 V 그리고 V에서 U로 수신된 메세지를 의미한다. $data(i,j,r;x)$는 라운드 r 동안 i가 j에게 변수 x와 관련된 송신할 메세지를 나타낸다. $pass(\{m_1, \ldots, m_k\}, \{C_1, \ldots, C_k\})$는 각각의 메세지 m_i를 채널 C_i에게 입력하므로써 생성되는 메세지의 집합을 의미한다.

입력 $\vec{x}$와 임의의 입력 $\vec{a}$를 가지는 동기식 $R(n,m,k)$-라운드 프로토콜 $\Pi(n,m,k)$의 실행 결과는 다음과 같은 실험으로부터 얻어지는 일련의 상태와 메세지들로 구성된다. $\vec{x}$와 $\vec{a}$에 따라서 모든 상태를 초기화한다. 각 라운드에서 각 통신참가자의 전이 함수를 사용해서 새로운 상태와 송신될 메세지를 선택하고, 그 메세지를 채널에 적용시켜서 실제적으로 송신할 메세지를 생성한다. 마지막으로 라운드 $R = R(n,m,k)$ 후에 최종 계산을 한다. 통신참가자 i의 출력은 자신의 마지막 상태를 입력으로 하는 함수의 값이다. 여기서 사용되는 네트워크는 비밀 채널과 광역 채널이다. 비밀 채널을 통해서 보내진 메세지는 $data_{priv}(U,V,r)$ 그리고 광역 채널을 통해서 보내진 메세지는 $data_{broad}(U,[n],r)$으로 표기한다. C_{priv}와 C_{broad}는 비밀 채널과 광역 채널을 의미한다. 이상의 설명은 그림 3.1과 같이 표시할 수 있다.

$\tilde{Q}$는 임의의 통신참가자의 상태 벡터의 집합이라 하고 다음과 같이 표시한다.

$$\tilde{Q} = \{\vec{q}_B \mid \vec{q} \in Q_1 \times \cdots \times Q_n,\ B \subseteq [n]\}.$$

유사하게 통신참가자들의 임의의 부분집합인 입력과 임의의 입력 집합을 표기하기 위해서 다음과 같이 표기한다.

$$\tilde{X} = \{(\vec{x}_B, \vec{a}_B) \mid \vec{x} \in X_1 \times \cdots \times X_n,\ \vec{a} \in Z_1 \times \cdots \times Z_n,\ B \subseteq [n]\},$$

그리고 통신참가자의 임의의 부분집합에 대한 출력 집합 $\vec{y}$를 표기하기 위해서 위의 표기법을 사용한다.

라운드 r 동안에 통신참가자의 부분집합 B에 의해서 기록된 상태 벡타와 메세지를 view라고 하고 다음과 같이 표기한다.

$$\text{view}^r_B = (\vec{Q}_B, M(B,[n],r))$$

모든 가능한 view의 집합은 $\tilde{V} = \tilde{Q} \times \tilde{H}$이다.

프로토콜의 기록(history) 혹은 실행은 입력, 임의의 입력, 초기 상태, view, 그리고 최종 상태의 집합인 exec로 표시할 수 있다.

$$\text{exec} = \tilde{X} \times \tilde{V} \times \tilde{Q} \times (\tilde{V})^R \times \tilde{Q}$$

프로토콜은 그림 3.1에서 표시한 실행에 따른 임의의 분포를 생성한다. 일반적으로 프로토콜은 입력을 받아서 exec를 출력하는 확률 함수로 간주할 수 있다.

For $i = 1..n$ **do** /* 상태 초기화 */

$q_i^0 \leftarrow q_i^0(x_i, a_i, k)$

For $r = 1..R$ **do for** $i = 1..n$ **do** /* 메세지 계산과 송신 */

$(q_i^r,\ data_{priv}^{out}(i,[n],r),\ data_{broad}^{out}(i,[n],r)) \leftarrow \delta_i(q_i^{r-1}, data([n],i,r-1))$

$(data_{priv}(i,[n],r),\ data_{broad}(i,[n],r) \leftarrow pass(data_{priv}^{out}(i,[n],r) \cup data_{broad}^{out}(i,[n],r),$
$C_{priv} \cup C_{broad})$

For $i = 1..n$ **do** /* 최종 계산 */

$q_i^f \leftarrow \delta_i(q_i^R, data([n],i,R))$

그림 3.1 비밀 채널과 광역 채널을 갖는 동기 네트워크 상에서 정직한 통신참가자만이 참여한 프로토콜 〔Π〕의 실행

3.3 Adversaries와 폴트 모델

Adversaries와 폴트 모델을 정의하는 목적은 여러 종류의 폴트 모델하에서 멀티파티 프로토콜의 안전성과 신뢰성을 검사하는데 있다. 폴트는 프로토콜에 따라서 계산된 메세지를 새로운 메세지로 대체하기 위한 방법을 임의로 선택할 수 있는 ADV에 따라서 분류할 수 있다. 이러한 선택은 ADV의 출력 스트링에 대한 해석으로 간주할 수 있다. ADV는 임의의 전이 함수와 자신의 현재 상태를 기본으로 해서 출력 스트링을 계산하는 악의의 통신참가자이다.

정적(static) ADV란 프로토콜이 실행되기 전에 자신이 오염(corrupt)시킬 통신참가자의 집합을 선택하는 ADV이다. 동적(dynamic) ADV란 라운드의 마지막에서 자신이 다음 라운드동안에 오염시킬 새로운 통신참가자의 집합을 선택하는 ADV를 말한다. 동적 ADV는 통신되는 메세지의 내용을 변경시킬 수 있다. 즉, 정직한 통신참가자가 그들의 메세지를 보낼 때까지 기다렸다가 ADV는 자신이 접근한 메세지를 본 다음 오염된 통신참가자를 통해서 메세지를 그 라운드동안에 출력한다. 앞으로 이것을 러쉬(rush)라 하고, 오염된 통신참가자를 폴트 통신참가자라 한다.

멀티파티 프로토콜의 안전성은 ADV의 능력에 의해 분류된다. 암호 이론적으로 안전성을 갖는 전형적인 예는 ADV가 다항식 계산 능력을 갖는 경우이다. 더욱 강력한 능력을 갖는 ADV는 무한한 계산 능력을 갖는 기계로서 이러한 ADV와 관련된 안전성은 정보 이론적으로 안전하다고 한다.

ADV와 관련한 중요한 파라메타는 폴트를 일으키고 있는 통신참가자의 수이다. 현재까지 멀티파티 프로토콜에서 허용되고 있는 폴트 통신참가자의 수는 $t < n/3$, $n/3 \leq t \leq n/2$, 그리고 $n/2 \leq t \leq n-1$이다.

폴트의 종류는 다음과 같이 분류할 수 있다.

(1) 수동(Passive) : ADV는 정직한 통신참가자의 메세지를 다른 임의의 메세지로 대체시킬 수 없으며, 자신이 폴트를 유발시킨 통신참가자의 테이프와 상태만 접근 할 수 있다.

(2) 실패-멈춤(Fail-Stop) : 폴트 통신참가자들은 부적절한 메세지를 사용할 수 없으며, 임의의 라운드에서 멈춘 후, 오직 널(∧)메세지만을 보낸다.

(3) 생략(Omission) : 폴트 통신참가자들은 부적절한 메세지를 사용할 수 없지만 임의의 메

세지를 주기적으로 생략할 수 있다. 즉, 메세지를 ∧로 대체시킨다. 이러한 모델은 폴트가 발생한 통신 선을 검사하는데 유용하다.

(4) Byzantine : ADV는 자신의 선택에 따라서 프로토콜에 따른 메세지를 계산할 것인지 아니면, 부적절한 메세지를 계산할 것인지를 결정할 수 있으며 위와 마찬가지로 메세지를 ∧로 대체시킬 수 있다.

3.3.1 수동 ADV

폴트 클래스 Γ는 폴트가 허용된 연합의 집합 즉, $[n]$의 부분집합들의 집합이다. 만약 $T \in \Gamma$이고 $U \subseteq T$이면 $U \in \Gamma$라 하자. 일반적인 폴트 클래스는 t-폴트 클래스 $\Gamma = \{T \subseteq [n] \mid |T| \leq t\}$이며, t 보다 적은 임의의 집합 T만이 오염되는 것을 허락한다.

수동 ADV는 튜플 $(Q_{\langle A_I, A_D\rangle}, Z_{\langle A_I, A_D\rangle}, q^0_{A_D}, \delta_{A_D}, T)$로 구성된다. Q_A는 상태의 집합이고, $q^0{}_A$는 Z_A를 입력으로 Q_A로 사상시키는 함수이며, δ_A는 전이함수, 그리고 T는 Q_A를 $[n]$으로 사상시키는 함수이며, ADV가 폴트를 일으키기 위해서 선택한 현재의 연합을 의미한다. 수동 ADV는 메세지를 생성하지는 않지만 자신이 유발시킨 폴트 통신참가자의 현재의 상태와 현재까지 통신된 내용을 이용해서 새로운 상태와 새로운 연합을 결정한다.

$$\delta_{A_D} : Q_A \times \widetilde{V} \rightarrow \mathrm{dist}(Q_A)$$

수동 ADV가 만약 T가 상수로 프로토콜 시작전에 고정되어 있으면, 정적이며, T가 ADV의 상태에 따라서 변화하면 동적이다. 연합을 구성할 수 있는 경우는 ${}_nC_t$이므로 랜덤하게 그들 중의 하나를

For $i = 1..n$ **do** /* 초기 상태 */

$q_i^0 \leftarrow q_i^0(x_i, a_i, k)$

Set

$q_A \leftarrow q_A^0(a_A, n, m, k)$

For $r = 1..R$ **do**

For $i = 1..n$ **do** /* 메세지 계산과 송신 */

$(q_i', data^{snt}(i, [n], r)) \leftarrow \delta_i(q_i^r, data([n], i, r-1))$

$data(i, [n], r) \leftarrow pass(data^{snt}(i, [n], r), C_1 \cup \ldots \cup C_n)$

Set

$q_A' \leftarrow \delta_{A_D}(q_A', \mathrm{view}^{r,1}_{T(q_A')})$

For $i = 1..n$ **do** /* 최종 계산 */

$q_i^f \leftarrow \delta_i(q_i^R, data([n], i, R))$

Set

$q_A^f \leftarrow \delta_{A_D}(q_A', \mathrm{view}^{R}_{T(q_A')})$

그림 3.2 정적인 수동 ADV가 참여한 프로토콜 〔A, Π〕의 실행

선택할 확률은 $c_n = 1/{}_nC_c$이다. 임의의 동적 ADV는 최소한 확률 c_n을 가지고 연합 T를 선택한다. 만약 n이 증가하면, 특정 연합 T의 확률은 급속히 감소하기 때문에 수동 ADV와 능동 ADV사이에는 큰 차이점이 있다.

폴트 클래스 T인 수동 T-ADV는 다음의 조건을 만족하는 수동 ADV이다.

(1) 모든 $q_A \in Q_A$에 대해서 $T(q_A) \in T$
(2) 이것은 항상 현재 연합의 superset인 새로운 연합을 선택한다.

조건 (2)를 다시 설명하면, 모든 $q_A \in Q_A$, 모든 부분적인 view인 $\mathbf{view}^{T(q_A)}$, 그리고 $\delta_{A_D}(q_A, \mathbf{view}^{T(q_A)})$의 출력인 모든 상태 q'_A에 대해서 $T(q_A) \subseteq T(q'_A)$이 성립한다.

그림 3.2는 R-라운드 프로토콜이 수동 ADV의 존재하에서 어떻게 동기가 맞게 수행되는 가를 보여준다.

3.3.2 Byzantine ADV

Byzantine ADV $(Q_{(A_1,A_D)}, Z_{(A_1,A_D)}, q^0{}_{A_D}, \delta_{A_D}, T)$는 자신이 유발시킨 폴트 통신참가자들에 의해서 송신되는 메세지를 고쳐 쓸 수 있는 추가적인 능력을 갖고 있다. 따라서 Byzantine ADV가 악의적인 메세지를 생성하기 위해서는 전이 함수를 다음과 같이 수정해야 한다.

$$\delta_{A_D} : Q_A \times \tilde{V} \rightarrow \mathrm{dist}(Q_A \times H)$$

수동 ADV와 마찬가지로, Byzantine ADV도 T가 상수이면 정적이고, T가 상태에 의존하면 동적이다. Byzantine t-ADV는 폴트 클래스 $\Gamma = \{T \mid |T| \leq t\}$를 의미한다.

가장 심각한 Byzantine ADV의 종류는 메세지를 러쉬하는 ADV로 Byzantine ADV는 다른 메세지들이 송신되기 전에 정직한 통신참가자들로부터 보내진 메세지를 이용하여 새로 폴트를 일으킬 통신참가자를 결정한다.

입력 $\vec{x}$, 임의의 입력 $\vec{a}$, ADV의 임의의 입력 a_A, 그리고 러쉬 가능한 Byzantine ADV를 가지는 프로토콜의 실행은 그림 3.3에서 보인다.

프로토콜이 끝난 후 ADV와 통신참가자는 출력 스트링 $Y_A = O_A(q'_A)$과 $Y_i = O_i(q'_i)$을 출력 테이프에 기록한다. 폴트 통신참가자의 테이프는 $\wedge$를 포함한다.

ADV A가 임의의 입력 a를 소유한다고 하자. 입력 $\vec{x} = (x_1, \ldots, x_n)$와 임의의 입력 $\vec{a} = (a_1, \ldots, a_n)$ 그리고 안전 파라메타 k에 대한 프로토콜 $[A, \Pi]$의 실행은 $(\Sigma^*)^{2n+2}$ 즉, A의 출력과 view와 통신참가자의 출력과 view에 대한 분포를 생성하며 다음과 같다.

$$[A, \Pi]^{hist}(n,m)(\vec{x} \circ \vec{a} \circ a, k) = (Y_A, Y_1, Y_2, \ldots, Y_n, \mathrm{view}_A, \mathrm{view}_1, \mathrm{view}_2, \ldots, \mathrm{view}_n).$$

프로토콜을 실행하므로써 생성된 ADV의 출력 즉, 분포는 다음과 같다.

$$[A, \Pi]^{Y_A}(\vec{x} \circ \vec{a} \circ a, k) = Y_A.$$

통신참가자의 출력에 대한 표기 역시 유사하다.

$$[A, \Pi]^{Y}(\vec{x} \circ \vec{a} \circ a, k) = (Y_1, Y_2, \ldots, Y_n).$$

비밀성은 $[A, \Pi]^{Y_A}$에 관계되고 정확성은 $[A, \Pi]^{Y}$과 관계된다.

4. 안전한 계산

비밀성과 정확성은 안전하고 신뢰성있는 계산에서 요구되는 가장 중요한 특성이다. 프로토콜은 만약 ADV의 view가 자신에게 부여된 제한된 정보로 부터 생성된 것일지라도 이 view에는 어떠한 정보도 포함되서는 안된다. 이것은 비밀성에 대한 일반적인 정의이다. 만약 통신참가자의 출력이 통신참가자의 입력을 변수로 하는 함수 $F(x_1, \ldots,$

(B1.1) **For** $i = 1..n$ **do** /* 초기 상태 */

$q_i^0 \leftarrow q_i^0(x_i, a_i, k)$

(B1.2) **Set**

$q_A \leftarrow q_A^0(a_A, n, m, k)$

$T \leftarrow T(q_A)$

$T_{old} \leftarrow \phi$

$mesg([n],[n],0) \leftarrow \phi$

(B2.1) **For** $r = 1..R$ **do**

(B2.1.1) **For** $i \in \bar{T}$ **do** /* 올바른 메세지 */

$(q_i^r, data^{out}(i,[n],r)) \leftarrow \delta_i(q_i^{r-1}, data([n],i,r-1))$

(B2.1.2) **Repeat** /* 메세지 러쉬와 새로운 폴트 */

$T_{new} \leftarrow T - T_{old}$

$T_{old} \leftarrow T$

$data(T, T_{new}, r) \leftarrow pass(data^{out}(\bar{T}, T_{new}, r), C_1 \cup \ldots \cup C_n)$

/* ADV 계산 그러나 메세지가 러쉬될 때까지 메세지 송신 안함 */

$(q_A, data(\bar{T}, T, r)) \leftarrow \delta_A(q_A, data(\bar{T}, T_{new}, r))$

$T \leftarrow T(q_A)$

until $T_{new} = \phi$ /* 새로운 폴트 미발생 */

(B2.1.3) **Set** /* 모든 지연된 메세지를 송신 */

$data(T, \bar{T}, r) \leftarrow pass(data^{out}(\bar{T}, T, r))$

$data(\bar{T}, \bar{T}, r) \leftarrow pass(data^{out}(\bar{T}, \bar{T}, r))$

(B3.1) **For** $i \in \bar{T}$ **do** /* 최종 계산 */

$q_i^f \leftarrow \delta_i^R(q_i, data([n], i, R))$

$q_A^f \leftarrow \delta_A(q_A, data([n], T, R))$

그림 3.3 러쉬 비잔틴 ADV를 포함하는 프로토콜 (A, II)의 실행

x_n)의 값이라면 프로토콜은 정확하다. 이것은 정확성에 대한 일반적인 정의이다.

비밀성에 대한 정의는 종종 입력 값에 대한 비밀에 주로 관심을 둠으로써 새어나갈지도 모르는 다른 정보에 대해서는 거의 무관하게 다루었다[12,13]. 만약 비밀성에 대한 정의가 정확하지 않다면, 각

각의 프로토콜은 자신의 입력은 숨길지 모르나 다른 프로토콜의 비밀 정보를 누출시킬런지 모른다. 기존의 프로토콜들은 $F(x_1, \ldots, x_n)$를 제외한 어떠한 정보도 누출시키지 않는다는 강력한 요구 조건을 만족했기 때문에 이러한 관점은 간과되었다.

어떤 특성을 얻기 위해서 기존의 연구들은 특별한 가정을 만들었다. 예로서 통신참가자에게 모든 입력이 암호화된 값으로 제공된다든가 혹은 모든 입력이 비밀리에 분산되었다 등이 있다. 입력에 대한 commital의 개념은 프로토콜에서 중요한 역할을 한다. 프로토콜의 실행이 끝난 후, 정확성은 commit된 입력과 관련하여 정의 내릴 수 있지만 이러한 접근 방식은 안전성을 얻기 위한 특정한 방법으로서 이것은 너무 특정해서 안전성에 대한 일반적인 정의로 간주할 수 없다.

따라서 본 고에서는 commital과 같은 특정한 면에 관점을 두지 않고 정확성을 이해하고 정의하기 위해서 일반적인 도구에 대해서 논의할 것이다. 또한 이러한 도구를 이용하여 비밀성에 대한 정의도 내릴 것이다.

일반적으로 프로토콜의 안전성과 신뢰성을 증명하거나 비교하기 위해서 신뢰성 센터가 함수를 계산하는 이상적인 프로토콜을 정의한다[2]. 실제의 프로토콜은 최대 t명의 부분 집합으로 구성되는 폴트 통신참가자와 임의의 통신 채널의 집합을 가지는 프로토콜이다. 예로서 실제의 프로토콜은 통신참가자 상호간에 광역 채널과 비밀 채널이 제공되며, 중요한 점으로서 어떠한 통신참가자가 오염되도 상관이 없다는 점이다. 반면에 이상적인 프로토콜은 절대적으로 신뢰성있고 안전한 신뢰성 센터가 존재한다. 이러한 가정은 실제의 상황에서는 신중하지 않고 효율적이지도 못하며, 신뢰성 센터에 의한 계산 역시 ADV의 존재를 고려해야만 한다. 이상적인 경우에서 ADV의 제한된 능력이 명확하게 묘사되며, 이것은 절대적인 안전성과 신뢰성에 대한 주장에 정당성을 주기 위해서 필요하다.

그림 4.1은 함수 F에 대한 이상적인 프로토콜을 묘사한 것이다. 통신참가자들은 신뢰성 센터와 함께 $n + 1$ 명이 된다. 각각의 통신참가자는 신뢰성 센터와 일대일 비밀 채널로 연결되어 있기 때문에 통신참가자간의 통신에 의해서 새어나가는 정보를 고려할 필요가 없다.

ADV에게 이상적인 프로토콜을 공격하는 것을 허락할 때, 그것의 능력은 통신참가자의 입력과 입력의 영향에 대해 엄격히 제한된다. 즉, 신뢰성 센터는 폴트에 대해 면역이 되어 있다. 이상적 t-폴트 클래스 T^{t}_{ideal}는 크기가 최대한 t인 $\{1, \ldots, n\}$

이상적 프로토콜 〈 ID(F) 〉

(I1) 각각의 통신참가자 i $(1 \le i \le n)$는 입력 (x_i, a_i)으로 시작한다.
신뢰성 센터는 어떠한 입력도 소유치 않으며, 각각의 통신참가자 i는 x_i를 신뢰성 센터에 송신

(I2) 신뢰성 센터는 영역 X_i에 있지 않는 모든 메세지에 대해 $X_i^* = \wedge$로 설정하고 그렇지않으면 메세지를 X_i^*로 설정
신뢰성 센터는 $(y_1, \ldots, y_n) \leftarrow F(x_1^*, \ldots, X_n^*)$를 계산
신뢰성 센터는 통신참가자 i에게 y_i를 송신 (단, $1 \le i \le n$)

그림 4.1 신뢰성 센터와 정직한 통신참가자로 구성된 이상적인 프로토콜

의 부분집합으로 구성된다. 즉, 이상적 t-ADV 클래스 A^{t}_{ideal}는 오직 이상적 t-폴트 클래스 내에서 연합을 요구하는 모든 ADV로 구성된다. 동적인 Byzantine ADV가 있는 이상적인 프로토콜의 실행은 그림 4.2에 묘사되어 있다.

안전한 프로토콜은 함수를 계산하기 위해서 그림 4.1과 그림 4.2의 이상적인 프로토콜에 의한 함수 F의 계산을 모방해야한다. 안전성에 대한 형식은 임의의 프로토콜에 의한 함수 F의 계산은 이상적인 프로토콜에 의한 함수 F의 계산과 같다는 것을 보증하는 절차에 의해서 형식화된다.

본 고에서는 안전성과 신뢰성에 대한 개념을 형식화하기 위해서 이상적인 프로토콜의 계산과 임의의 프로토콜의 계산이 같다는 것을 Beaver가 제안한 방식 즉, resilience를 이용한다.

Resilience를 정의하기 위해서 프로토콜의 안전성과 신뢰성에 의해서 한 프로토콜을 다른 프로토콜과 비교할 수 있는 수단인 인터페이스 S를 설명한다. 안전성과 신뢰성의 조합을 resilience라 부른다. 인터페이스 S를 이용하여 프로토콜의 실행동안 도청 ADV가 수집한 정보를 측정하고, 방해 ADV가 출력에 미치는 영향을 측정하고자 한다. 신뢰성 센터를 공격할 수 없는 ADV는 오염된 통신참가자들이 센터에게 보낼 입력들을 선택할 수 있으므로 계산에 대해 제한된 영향을 미칠 수 있다. 따라서 도청 ADV가 배우는 정보와 방해 ADV가 계산에 영향을 미치는 것과 관련하여 ADV의 공격을 조사해야 한다.

두 개의 멀티파티 프로토콜 α와 β가 있다고 하자. 각각은 허용된 ADV의 클래스 A_{α}와 A_{β}를 갖고

ADV가 있는 이상적 프로토콜 〈 A, ID(F) 〉

IA1.1 각각의 통신참가자 i($1 \le i \le n$)는 입력 (x_i, a_i)로 시작한다.
신뢰성 센터는 어떠한 입력도 소유치 않음.

IA1.2 ADV는 오염시킬 통신참가자의 허락된 부분집합 $T \subseteq \{1, \ldots, n\}$을 선택한다. 이것은 이러한 통신참가자의 입력 $(\vec{x_T}, \vec{a_T})$를 얻음.
동적 ADV는 이러한 단계를 반복.

IA1.3 ADV는 연합 T에 대한 효과적인 대체 입력 집합 x_i'을 선택하고, 이것을 신뢰성 센터에게 송신.

IA1.4 정직한 통신참가자는 자신의 입력 x_i을 신뢰성 센터에 송신.

IA2.1 신뢰성 센터는 영역외에 있는 메세지 x_i^*를 $\wedge$로 설정하고 그렇지 않으면 x_i^*를 자신이 받은 메세지로 간주.
신뢰성 센터는 $(y_1, \ldots, y_n) \leftarrow F(x_1^*, \ldots, x_n^*)$를 계산.
단, $x_i^* : x_i, \wedge$, 혹은 가능한 임의의 오염된 메세지.

IA2.2 신뢰성 센터는 통신참가자 i에게 y_i를 송신.
ADV는 출력 $\vec{y_T}$을 수신.

IA2.3 ADV는 오염시킬 더많은 통신참가자를 선택해서 그들의 입력 (x_i, a_i)과 그들의 출력 값 $\vec{y_T}$을 얻음.

그림 4.2 신뢰성 센터, 동적 ADV, 그리고 정직한 통신참가자로 구성된 이상적인 프로토콜

있다. ADV $A \in A_\alpha$에 대한 프로토콜 α의 resilience와 프로토콜 β의 resilience를 비교하기 위해서 A가 프로토콜 β를 공격하는 것을 허락한다. 그러나 α와 β는 근본적으로 다른 프로토콜일런지 모른다. 따라서 ADV A를 이용하여 프로토콜 β를 실행할 수 없다. 대신에 인터페이스 S를 이용하여 A를 둘러 싸보자. 인터페이스 S는 A가 프로토콜 α에 참여하고 있다고 확신시킬 수 있는 A를 위한 환경을 생성한다. 반면에 S는 프로토콜 β를 공격하는 것이 허락된다. 다시 설명하면, S 내부에 있는 A와의 조합 $S(A,\cdot)$는 프로토콜 β의 실행에 대한 ADV로서 사용된다. 표기 $S(A,\cdot)$는 S가 두 개의 통신 라인을 갖고 있다는 것을 의미한다. 하나는 A와의 통신을 위한 것이고 다른 하나는 프로토콜 실행에 포함된 ADV의 입출력 선으로서 사용된다. $S(A,\cdot)$는 A_β에 속해 있는 것이 허용된 ADV임에 틀림없다.

예로서 오염이 불가능한 센터가 하나의 메세지를 오염된 통신참가자에게 송신하는 특별하고 간단한 프로토콜 β를 생각해 보자. 프로토콜 β에 대한 ADV는 오직 이러한 정보만 얻을 수 있다. 인터페이스 S는 이러한 메세지를 받아서 A를 위한 적절한 환경을 제공해야 한다. 이러한 방식에서 상호 통신은 영지식 상호 증명 방식의 특별한 경우에 해당되고 인터페이스는 시뮬레이터로서만 사용된다.

정의 4.1 인터페이스 $S(\cdot,\cdot)$는 두 개의 통신 라인을 갖는 상호 튜링 기계이다. 첫번째 라인은 환경 시뮬레이션 라인이고 두번째는 ADV 라인이다. 만약 α가 ADV 클래스 A_α를 인정하고 β가 ADV 클래스 A_β를 인정하면, 다음의 조건을 만족한다면 S를 α와 β를 연결하는 인터페이스라 한다[11].

조건) 모든 $A \in A_\alpha$에 대해서, $S(A,\cdot) \in A_\beta$

앞으로 인터페이스를 $S(\cdot,\cdot)$로 표기한다. $S(\cdot,\cdot)$가 단하나의 통신 라인을 가지는 임의의 입력을 갖는 상호 기계 ADV A와 연결됐을 때, $S(A(a),\cdot)$를 자신의 오른쪽에 있는 ADV로써 간주한다. 이것은 단하나의 통신 테이프 즉, ADV 테이프를 갖는다. S가 자신의 ADV 라인에다 쓴 오염 요구와 오염된 메세지를 사용해서, 마치 프로토콜 β에서 발생한 것처럼 요구된 메세지와 β에 있는 오염된 통신참가자에 의해서 소유된 정보를 S에게 제공할 수 있기 때문에 $S(A(a),\cdot)$을 β에 대한 ADV로써 취급한다.

오염에 대한 요구와 오염된 메세지들을 전달할 수 있는 인터페이스가 주어진 상태에서 ADV A가 프로토콜 α를 오염시킬 수 없으면 그것이 프로토콜 β 역시 오염시킬 수 없다는 것을 인터페이스 S를 이용하여 형식화하여 보이겠다. 만약 이러한 경우가 발생한다면 α는 β만큼 resilience하다 라고 말할 수 있다. 즉, ADV가 α를 공격하든지 인터페이스의 도움으로 β를 공격하던지 간에 정직한 통신참가자의 상태에 대한 ADV의 영향은 같다는 것을 의미한다. ADV는 그것이 α를 공격했을 때 어떤 상태에 도달한 다음 공격을 중단했을 것이다. 이때, 최종 상태에서의 분포는 그것이 α를 공격했을 때와 S에 의해서 시뮬레이트된 환경에서 공격했든지 간에 같다. 이것은 α에서 어떠한 정보도 얻지 못했으면 β에서도 역시 어떠한 정보도 얻지 못한다는 것을 의미한다.

분포 $[A,\alpha](n,m)(\vec{x}\circ\vec{a}\circ a,\ k)$와 $[A,S,\beta](n,m)(\vec{x}\circ\vec{a}\circ a,\ k)$를 살펴보자. 전자는 ADV $A(a)$와 입력 그리고 임의의 입력을 갖는 프로토콜 α를 실행하므로써 생성된 분포이다. 후자는 ADV $S(A(a),\cdot)$와 같은 입력 그리고 임의의 입력을 갖는 프로토콜 β를 실행하므로써 생성된 분포이다. n, m, $\vec{x}$, $\vec{a}$ 그리고 a가 변화한다면, 두 개의 프로토콜은 각각 앙상블 $[A,\alpha]$와 $[A,S,\beta]$을 생성한다. 이러한 두 개의 앙상블은 방해 ADV의 영향을 반영하는 통신참가자의 최종 상태와 도청 ADV가 얻은 정보를 반영하는 ADV의 최종 상태를 묘사한 것이다.

정의 4.2 다음의 조건을 만족한다면 프로토콜 α는 프로토콜 β만큼 (A_α, A_β)-resilient 하다. 즉, $\alpha \geq (A_\alpha, A_\beta)\beta$.

조건) 모든 ADV $A \in A_t$에 대해서 $[A,\alpha] \asymp [A, S,\beta]$를 만족하는 α와 β를 연결하는 인터페이스 S가 존재한다.
단, 기호 $\asymp$는 두개의 앙상블이 같다는 의미이다.

멀티파티 프로토콜의 정확성과 비밀성 그리고 장애 허용에 대한 정의를 내리고 이상적인 프로토콜을 사용하여 다음과 같이 정의를 한다.

■ 정의 4.3 폴트 클래스 T를 가진 프로토콜 Π이 다음의 조건을 만족한다면 T-resilient F이다.

조건) $\Pi^{(A_t,A_D)} \geq {}^{(T,A^I_{Shell})}\mathbf{ID(F)}$.

■ 정의 4.4 다음은 정확성과 비밀성에 대한 정의이다.

정확성 : $[\mathbf{A},\Pi]^{(Y,A_D)} \asymp [\mathbf{A},\mathbf{S},\mathbf{ID(F)}]^{(Y,A_D)}$를 만족하는 인터페이스 S가 존재한다면 프로토콜 Π는 t-correct F이다.

비밀성 : $[\mathbf{A},\Pi]^{Y,A_I} \asymp [\mathbf{A},\mathbf{S},\mathbf{ID(F)}]^{Y,A_I}$를 만족하는 인터페이스 S가 존재한다면 프로토콜 Π는 t-private F이다.

참 고 문 헌

[1] W.Diffie and M. Hellman, "New Directions in Cryptography," IEEE Transactions of Information Theory IT-22, pp. 644-654, Nov. 1976.

[2] O. Goldreich, S. Micali, and A. Wigderson, "How to Play Any Mental Game, or A Completeness Theorem for Protocols with Honest Majority," The 19th STOC ACM, pp.218-229, 1987.

[3] 양 형규, 원 동호외 6인, "분산 시스템에서 영지식을 이용한 multiparty 프로토콜에 관한 연구," 한국전자통신연구소 최종보고서, 1993.

[4] A. Shamir, "How to Share a Secret," Communications of the ACM, 22, pp. 612-613, 1979.

[5] A.C. Yao, "Protocols for Secure Computations," The 23th IEEE FOCS, pp.160-164, 1982.

[6] D. Chaum, "Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms," Comm. of the ACM 24 (2), pp.84-88, 1981.

[7] B. Chor, S. Goldwasser, and B. Awerbuch, "Verifiable Secret Sharing and Achieving Simultaneity in the Presence of Faults," The 26th IEEE FOCS, pp. 383-395, 1985.

[8] M. Blum, "Coin Flipping by Telephone," IEEE COMPCON, pp.133-137, 1981.

[9] M. Rabin, "How to Exchange Secrets by Oblivious Transfer," Tecnical Memo TR-81, Aiken Computation Lab, Harvard University, 1981.

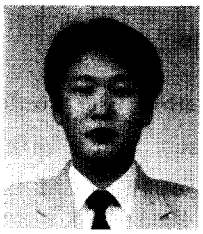
[10] A. Shamir, R. Rivest, and L. Adleman, "Mental Poker," In Mathmatical Gardner, d.e.Klarner, ed., International, pp.37-43, 1981.

[11] D. Beaver, "Distributed Computations Tolerating a Faulty Minority, and Multiparty Zero-Knowledge Proof System," J. Cryptology, 1990

[12] S. Haber, "Multi-party Cryptographic Computation: Techniques and Applications," PhD Thesis, Columbia University, 1988.

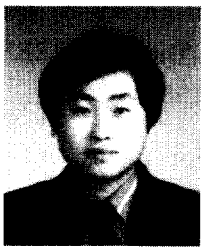
[13] M. Abadi, J. Feigenbaum, and J. Kilian, "On Hiding Information from an Oracle," J.Comput. System Sci. 39 pp.21-50, 1989.

□ 著者紹介



양 형 규(정회원)

1959년 2월 15일생
1983년 성균관대학교 전자공학과 졸업(공학사)
1985년 성균관대학교 대학원 전자공학과 졸업(공학석사)
1991년 ~ 1994년 성균관대학교 대학원 정보공학과 졸업(공학박사)
1985년 ~ 1991년 삼성전자 컴퓨터부문 선임연구원
1995년 ~ 현재 강남대학교 전자계산학과 전임강사



박 성 준 (朴 性 俊, Sung Jun Park) 정회원

1960년 10월 29일생
1983년 2월 한양대학교 수학과 졸업(이학사)
1985년 2월 한양대학교 대학원 수학과 졸업(이학석사)
1985년 1월 ~ 1994년 3월 한국전자통신연구소 부호기술부 선임연구원
1992년 3월 ~ 현재 성균관대학교 대학원 정보공학과 박사과정

※ 주관심분야 : 암호이론, 계산이론, 정보이론

원 동 호 (元 東 豪, Dong-Ho Won) 종신회원

1949년 9월 23일생
1976년 2월 성균관 대학교 전자공학과 졸업 (공학사)
1978년 2월 성균관 대학교 대학원 전자공학과 졸업 (공학석사)
1988년 2월 성균관 대학교 대학원 전자공학과 졸업 (공학박사)
1978년 4월 - 1980년 3월 한국전자통신연구소 연구원
1985년 9월 - 1986년 8월 일본 동경공대 객원연구원
1982년 3월 - 현재 성균관대학교 공과대학 정보공학과 교수
1991년 - 현재 한국통신정보보호학회 편집이사

※ 주관심분야 : 암호이론, 정보이론