효율적인 오용 방지 다자간 서명 프로토콜

주홍돈° 장직현
서강대학교 컴퓨터학과
{narziss°,jchang}@sogang.ac.kr

An Efficient Abuse-free Multi-party Contract Signing Protocol

Hongdon Joo° J. Chang
Dept. of Computer Sciences and Engineering, Sogang University

요 약

다자간의 계약서 서명 프로토콜은 Asokan등[1]에 의하여 제안되었다. 비동기 네트워크에서 다자간의 계약서 서명 프로토콜은 동기 네트워크에 비하여 효율성이 떨어져 그에 대한 연구가 많이 이루어졌다[2,3,4]. 지금까지 알려진 비동기 환경에서 가장 효율적인 다자간의 계약서 서명 프로토콜은 Waidner[4]에 의하여 제시되었다. 본 논문에서는 Waidner[4]가 제시한 프로토콜을 기본으로 하여 라운드 수를 줄인 프로토콜을 제시한다.

1. 서 론

다자간 공정 계약서 서명 프로토콜은 공정 교환 프로토콜의 단순화된 프로토콜로 여러 가지 효율적인 프로토콜이 제시되었다[1,2,3,4]. 다자간 계약서 서명 프로토콜을 간단하게 정리하면, n명이 계약서에 서명을 하고자 할 때, 정직한 참가자들은 모두 서명된 계약서를 얻거나 또는 아무도 서명이 된 계약서를 얻지 못하게 하는 프로토콜이다. 여기에서 부정직한 참가자들이란 프로토콜에서 주어진 명세대로 행동을 하지 않은 참가자들을 의미한다. Asokan 등[1]에 의하여 2개의 페이즈를 가지는 프로토콜을 제시하였다. 하지만 Asokan 등[1]에 의하여 제시된 프로토콜은 동기 네트워크에서 동작하는 프로토콜이다. 최초의 비동기 방식 프로토콜은 J.Garay[2]등에 의하여 제시되었는데, $O(n^2)$의 라운드를 요구한다. Waidner[3]은 O(t)(t는 부정직한 참여자의수)의 라운드를 가지는 프로토콜을 제시하였다. 그 후에 Waidner[4]는 이를 발전시켜서 좀 더 라운드의 수를 줄인 프로토콜을 제시하였다. 본 논문에서는 Waidner[4]에서 제시한 프로토콜을 기반으로 하여 라운드 수를 감소시킨 프로토콜을 제시한다.

2. 모델 및 정의

본 논문에서 사용하고자 하는 모델은 Waidner[3]에서 제시된 모델과 유사하게 정의된다.

참가자 : $P_1,\ldots P_n$ 은 다자간의 계약서 서명에 참가하는 참가자이다. TTP는 신뢰기관이다.

공격자 모델: n 명이 서로 서명을 하는 다자간의 계약서프로토콜에서 최대 t(0<t<n)명의 참가자가 정직하지 않을 수 있다. 정직하지 못한 참가자들은 서로 협력할 수 있다.

네트워크 : 신뢰기관 TTP에서부터 각 사용자로 보내지는 메시지들과 신뢰기관 TTP로 보내는 모든 메시지는 정확하게 배송된다. 참여자들 간의 메시지들은 배송이 되지 않을 수도 있고, 또는 순서대로 배달이 되지 않을 수도 있다. 공격자는 모든 채널의 메시지를 읽을 수도 있고, 경우에 따라서 자신이 메시지를 추가할 수도 있다.

라운드 : 동기 방식에서의 페이즈(Phase)는 일정한 시간 안에 완료가 되는 것을 의미하지만, 비동기 네트워크에서의 라운드는 동기화가 이루어져있지 않다. 다만, 현재 라운드에 사용하는 모든 메시지들은 이전 라운드에서 보내거나 받은 메시지들 또는 로컬로 직접 만들 수 있는 것들을 이용한다.

정의 1 : 비동기 환경에서 다자간 계약서 서명 프로토콜 : 다음의 2가지 프로토콜로 구성된다.

서명 (sign[$P_1,..,P_n$]) : 다자간의 계약서에 서명을 하려고 하는 참가자 $P_1,..,P_n$의 서명으로 구성됨. 경우에 따라서 신뢰할 수 있는 제삼자인 TTP가 참여할 수 있다.

검증 (verify[P_i,V_j]) : 참가자 P_i는 자신의 계약서를 임의의 V_j, $j \in \{1,..n\}$에게 보일 수 있다. 이 경우에는 TTP는 참여하지 않는 양자간의 프로토콜이다.

참가자 P_i는 로컬 입력 (tid_i, terms, contr)으로부터 프로토콜을 시작한다. tid_i는 계약서 서명을 하는 동안에 식별자로 사용되며, 서명의 수행 중에 유일하게 존재한다. terms는 현재 수행하는 프로토콜에 대한 정보와 참가자들의 수와 식별자에 대한 정보를 포함하고 있다. contr는 서명을 하고자 하는 문서를 의미한다. 또, 참가자 P_i는 임의의 시점에서라도 resolve 메시지를 전송하고 TTP로부터 결정이 올 때까지 기다릴 수 있다. 비동기 환경에서 다자간 계약서 서명 프로토콜은 다음의 요구사항들을 반드시 만족하여야 한다.

정확한 수행 : 모든 참여자가 정직하게 프로토콜을 수행한다면, 공정 계약서 교환 프로토콜은 정상적으로 완료된다.

위조 불가능 : 정직한 참여자 P_i가 서명을 다른 참가자들에게 전송하지 않는 한, 누구도 임의의 P_j로부터 다자간의 서명이 완료된 메시지를 가질 수 없다.

유효한 계약서에 대한 검증 : 참가자들은 향후에 계약서에 대한 검증을 할 수 있어야 한다.

유효하지 않은 계약 : 만약에 정직한 참가자 P_i가 취소 메시지를 받는다면, 어떠한 참가자도 P_j도 유효한 계약서를 가지지 않는다.

적절한 시간이내에 완료 : 정직한 참가자 P_i는 서명된 계약서 또는 취소 메시지를 적절한 시간 이내에 받을 수 있다.

즉, 부정직한 참가자의 의도에 따라 조절이 불가능하다.

정의 2 (낙관적인 프로토콜) : 모든 참가자들이 정직하게 프로토콜을 따라간다면, TTP의 참가 없이 프로토콜이 완료된다.

오용자유에 대한 개념은 J.Garay 등[5]이 제시하였다.
정의 3 (오용 자유(Abuse freeness)) :프로토콜 진행 중에는 외부에 프로토콜을 완료 또는 취소시킬 수 있는 능력이 있음을 증명할 수 없다.

3. 프로토콜

Waidner[4]에서는 다자간의 계약서 서명 프로토콜과 오용방지 기능을 추가하는 프로토콜을 순차적으로 적용하여, 오용방지가 가능한 다자간의 계약서 서명 프로토콜을 다자간의 계약서 서명 프로토콜에 R라운드 그리고 오용방지 기능을 추가하는 프로토콜에 2라운드, 총 R+2라운드에 완료하였다. 본 논문에서는 Waidner[4]에서 제시한 프로토콜을 기반으로 하여 두개의 다자간의 계약서 서명 프로토콜과 오용방지 기능을 추가하는 프로토콜을 혼합하여서 R+1 라운드에 완료되는 프로토콜을 제시하였다. 여기에서 R은 Waider[4]에서 제시한 것과 같으며 다음과 같이 정의된다.

$$\begin{cases} 3 & \text{if } n \geq 2t+1 \\ 2\lfloor (t+1)/(n-t) \rfloor + 1 + \min(2, t+1 \bmod (n-t)) & \text{if } t+2 < n \leq 2t \\ t+2 & \text{if } n = t+1, t+2 \end{cases}$$

다자간의 계약서 서명 프로토콜

서명 프로토콜 : P_i

c_i = (tid_i, terms, contr); c = (terms, contr);
r = 1;
//sign은 P_i의 비밀키 sk_i로 생성; 공개키 pk_i에 의하여 검증
//SIGN은 비밀키 SK_i로 생성;PK_i에 의하여 검증;
//ECERT = ($ecert_1$,...,$ecert_n$),(PK_1,..,PK_n)
//CERT = ($cert_1$,...,$cert_n$),(PK_1,..,PK_n)
라운드 1 :
* 단기간 사용하는 비대칭키 (PK_i,SK_i)를 생성;
* 다음을 계산 $cert_i = sign_i(PK_i)$, $ecert_i = E_T(cert_i; r_i)$
* 모든 참가자에게 $m_{1,i}$=($SIGN_i(c_i,1,ok)$,($ecert_i$,PK_i)) 전송
* 다른 모든 참가자들로부터 $m_{1,j}$를 받으면 M_1=($SIGN_1(c_1,1,OK)$,...,$SIGN_n(c_n,1,OK)$); $X_1 = M_1$;ECERT를 설정
* 라운드를 증가

라운드 r : $1<r\leq R$
* P_i는 $m_{r,i}$=($SIGN_i(M_{r-1},r,OK)$, $SIGN_i(c_i,r,OK)$)를 모든 참가자 P_j에게 전송
* 모든 참가자들로부터 $m_{r,j}$를 받으면 M_r=($SIGN_1(c_i,r,OK)$, ...,$SIGN_n(c_n,r,OK)$와 X_r=($SIGN_1(M_{r-1},r,OK)$,..,$SIGN_n(M_{r-1},r,OK)$)을 설정
* r를 1씩 증가

라운드 R+1:
* r_i를 모든 참가자들에게 전송
* 다른 모든 참가자들의 $ecert_j$를 r_j를 이용하여 $cert_j$ = $sign_j(PK_j)$인지 검사
  -C = (c,CERT,M_r)로 설정
  -아니면 TTP로 reserve 메시지 전송

P_i는 언제든지 프로토콜을 중단하고 TTP에게 resolve 메시지 (ECERT,$resolve_{r,i}$)를 전송할 수 있다.

$$resolve_{r,i} = \begin{cases} (c_i, r, i, sign_i(M_R), M_R) & R+1 \\ (c_i, r, i, SIGN_i(X_{r-1}), X_{r-1}, M_{r-2}) & r \geq 3 \\ (c_i, 2, i, SIGN_i(X_1), X_1) & r = 2 \\ (c_i, 1, i, SIGN_i(c_i)) & r = 1 \end{cases}$$

Resolve 메시지를 TTP에게 전송한 후에는 TTP로부터 메시지가 올 때까지 기다린다. TTP는 다음의 2가지 중에 1개의 메시지를 전달한다.
sign = (CERT,($resolve_{r,i}$, $sign_T$(c,r,sign)))
abort = ($resolve_{r,i}$, $sign_T$(c,r,abort))

서명 프로토콜 : TTP

초기화 : 처음으로 resolve 메시지를 TTP가 받은 경우
- RQ = {}; RDC = {};
- RMAC = {}; Round = 2;
- Result = MayABORT RMRC = {};

규칙 1 : 유효한 메시지인지 검사
a. 유효한 tid_i로부터 온 $resolve_{r,i}$인지 검사
b. $P_i \in$ RDC ∪ RMAC ∪ RMRC 이면 종료
c. r≠ 1이면 CERT가 유효한지 검사
  - ($resolve_{r,j}$, $sign_T$(c,r,abort)) 전송 후 종료
d. RQ = RQ ∪ {$resolve_{r,i}$} ;

규칙 2 : Result = MayABORT인 경우
a. r < Round -1이면 RDC = RDC ∪ {P_i}
b. r = Round -1이면 RMAC = RMAC ∪ {P_i}
c. r = Round 이면
  - |RMAC| = 0 이면 Result = SIGN
    - P_i에게 (CERT,$resolve_{r,i}$, $sign_T$(c,r,sign)) 전송
  - 아니면 RMRC = RMRC ∪ {P_i}
d. r = Round +1 이면
  - Round = r;
  - |RMAC| + |RMRC| < n-t 이면 Result = SIGN
    - P_i에게 (CERT,$resolve_{r,i}$, $sign_T$(c,r,sign)) 전송
    - RMAC ∪ RMRC 에 있는 모든 P_j에게
      - ($resolve_{r,j}$, $sign_T$(c,r,sign))
  - |RMAC| + |RMRC| ≥ n-t이면
    - RMRC 에 있는 모든 P_j에게
      - ($resolve_{r,j}$, $sign_T$(c,r,abort)) //최종 결과 아님
    - R = 2이면 RMAC 에 있는 모든 P_j에게도
      - ($resolve_{r,j}$, $sign_T$(c,r,abort)) //최종 결과 아님
    - RDC = RDC ∪ RMAC
    - RMAC = RMRC
    - RMRC = {}
e. r >Round 이면
  - Round = r;
  - P_i에게 (CERT,$resolve_{r,i}$, $sign_T$(c,r,Sign)) 전송
  - Result = SIGN
f. |RMAC ∪ RMRC| ≥ t+1 - |RDC| 이고 Result=MayABORT이면
  - Result = ABORT
  - P_i에게 ($resolve_{r,i}$, $sign_T$(c,r,abort)) 전송

규칙 3 : Result = SIGN인 경우
a. P_i에게 (CERT,$resolve_{r,i}$, $sign_T$(c,r,sign)) 전송

규칙 4 : Result = ABORT인 경우

a. P_i에게 ($resolve_{r,i}$, $sign_T(c,r,abort)$) 전송

검증(Verify) : 다자간의 계약서 서명은 2가지의 형식으로 구성될 수 있다.
규칙 1 : $(c,CERT,SIGN_1(c_1,R,OK),...,SIGN_n(c_n,R,OK)))$
규칙 2 : $(CERT,c,resolve_{r,j},sign_T(c,r,sign))$

4. 안전성

정리 1 : 다자간의 공정한 계약서 서명 프로토콜은 모든 참가자들이 정직하게 행동을 한다면 R+1라운드에 완료된다. 그리고 프로토콜이 진행 중에 외부에 현재 프로토콜을 완료 또는 취소시킬 수 있는 능력이 있음을 증명할 수 없다.
증명) 모든 사용자들이 프로토콜을 따라서 수행하면 위의 다자간의 계약서 서명 프로토콜이 완료되는 것과 검증 규칙1 또는 2가 쉽게 검증 가능한 것은 자명하다. 그리고 모든 서명은 참가자의 비밀키 sk_i 또는 SK_i 로 서명하므로, 디지털 서명이 안전하다면 다른 참가자들이 서명을 위조할 수 없다. 그리고 TTP로 보내는 모든 resolve 메시지에는 참가자들끼리 교환하는 메시지와는 다른 메시지에 서명을 하여 전송하므로, 어떠한 참가자도 TTP로 보내는 resolve 메시지 또한 위조할 수 없다.
공정성에 대하여 살펴보자. 먼저, CERT가 유효한 값이 아니라고 한다면 프로토콜이 무조건 취소되므로, 의미가 없다. 그래서 CERT가 유효한 값을 가지는 경우에 대하여 살펴본다. 정직한 참가자 P_i가 라운드 r에 TTP로부터 취소 메시지($resolve_{r,i}$, $sign_T(c,r,abort)$)를 받았다면, 다른 모든 정직한 참가자들도 r 또는 r+1 라운드에 취소메시지를 TTP로부터 받는다. 이는 P_i는 r+1라운드에는 더 이상 프로토콜을 처리하지 않기 때문이다. 그런데, 정직한 참가자 P_i가 라운드 R에 TTP로부터 취소 메시지를 받았다면, 일부의 참가자들은 P_i로부터 서명된 메시지를 받아서 프로토콜을 완료할 수 있다. 하지만, R의 정의에 의하여 임의의 정직한 참가자 P_i가 라운드 R에서 취소 메시지를 받게 되려면, 이전 모든 연속되는 2개의 라운드들은 n-t명 이상이 resolve 메시지를 TTP로 전송해야 하는데, $R= 2\lfloor (t+1)/(n-t) \rfloor + \min(2,(t+1) \bmod (n-t)) +1$ 로 구성되어 있으므로 매 연속되는 2라운드 마다 n-t의 resolve 메시지를 전송한다고 하면 부정직한 참가자 모두가 TTP에게 resolve 메시지를 보낸 시점이후에도 2개의 라운드가 남게 된다. 그러므로 정직한 참가자는 R-1 라운드 이전에 취소 메시지를 받게 된다. 또, n=t+1인 경우에는 R=t+2이므로 하나의 라운드라도 resolve를 보내지 않는 적이 없다면 t라운드에 모든 부정직한 참가자의 resolve 메시지가 끝이 나고 2개의 라운드가 남게 되므로 취소메시지를 받는 최대 라운드는 R-1이 된다. 그리고 정직한 참가자들이 취소메시지를 받았다면 다음 라운드에서는 정직한 참가자들이 서명한 메시지가 없으므로 프로토콜을 진행할 수 없다. 그래서 다른 참가자들은 서명된 계약서를 얻을 수 없다.
이제 비동기 네트워크에서는 중요한 기능인, 부정직한 참가자들에 의한 시간 조절이 가능한지에 대하여 살펴보자. 만약 부정직한 참가자가 자신이 원하는 시간까지 프로토콜을 완료 또는 취소 결정을 연장하려면, 단 한명의 정직한 참가자도 resolve 메시지를 R-3 라운드 이전에 TTP로 보내면 안 된다. 왜냐하면, 정직한 참가자 P_i가 resolve 메시지를 r라운드에서 보냈다면, 이전 라운드의 상황에 따라서 r-1 라운드 이전 한 라운드라도 연속된 2개의 라운드가 n-t이하의 resolve 메시지를 TTP가 받았다면 다자간의 계약서 서명 프로토콜은 완료될 것이고, 아니라면 r+1 라운드에서 취소되기 때문이다. 그리고 R-2라운드에 프로토콜의 취소 또는 완료가 결정이 유보되려면, R-2 라운드시작 전에 최대로 남아 있는 부정직한 참가자의 수는 t'는 $n \geq 2t'+1$을 만족한다. 그리고 R-2와 R-1라운드에 n-t'의 참가자가 resolve를 보내지 않으면 프로토콜이 완료되는데, 이는 정직한 참가자가 resolve 메시지를 TTP에 보내야만 한다. 그런데 정직한 참가자가 resolve메시지를 TTP에 전송했다면 R-1라운드에서 프로토콜이 취소된다. 그러므로 부정직한 참가자들이 시간 조절하는 것은 불가능하다.
마지막으로, 오용 방지 기능이 있는지 살펴보자. 참가자 P_i가 자신이 프로토콜을 취소 또는 완료 시킬 수 있는 능력을 외부에 보여주려고 한다면, 다른 참가자들로부터 받은 PK_i가 정당함을 증명해야하는데, 이는 $cert_i$가 요구된다. 그런데 $cert_i$는 TTP에 의하여 암호화 되어 $ecert_i$로 구성되어 있기 때문에 암호화에 사용된 랜덤 변수 r_i를 P_i로부터 받기 전에는 외부에서 뿐만 아니라 참가자 또한 확실하게 알 수가 없다. 그러므로 R+1 이전에는 증명이 불가능하다. 그리고 단 하나의 참가자라도 R+1라운드에 있다면 나머지 참가자들은 R라운드 이상에 있으며, CERT가 정상적이라면 프로토콜의 취소는 불가능하고 완료만 가능하므로 외부에 증명하는 것이 의미가 없다. □

4. 결론 및 향후 연구 방향

본 논문에서는 Waidner[4]에서 제시한 프로토콜을 변형하여 조금 더 적은 수의 라운드를 가지는 프로토콜을 제시하였다. 다음의 표는 본 논문에서 제시한 프로토콜로서 향상이 되는 정도를 나타낸 것이다. t와 관계없이 1개의 라운드가 감소하였다.

	Waidner[4]	Our Scheme
n=t-1,t-2	t+4	t+3
$2t \geq n > t+2$	$2\lfloor (t+1)/(n-t) \rfloor + 3 + \min(2,(t+1) \bmod (n-t))$	$2\lfloor (t+1)/(n-t) \rfloor + 2 + \min(2,(t+1) \bmod (n-t))$
$n \geq 2t+1$	5	4

향후에 Waidner[4]에서 제시한 프로토콜보다 작은 수의 라운드를 가지는 프로토콜의 개발 및 하한에 대한 연구가 필요하다. 뿐만 아니라, 오용 방지 기능을 추가하기 위하여 Waidner[3]이외에도 J.Garay 등[5]이 제시한 방식을 이용하면 라운드 수를 늘리지 않지만 통신상에 많은 메시지가 요구되므로, 라운드의 수를 줄이면서도 메시지의 양이 크게 증가하지 않는 방법에 대한 연구도 필요하다.

참고 문헌

[1] N. Asokan, B. Baum-Waidner, M. Schunter, M. Waidner : Optimistic Synchronous Multi-Party Contract Signing; IBM Research Reports RZ 3089(#93125), Zurish, December 1998

[2] J.Garay, P.MacKenzie: Abuse-free Multi-party Contract Signing:intern. Sympon Distr. Comput (DISC '99) LNCS 1693, Springer-Verlag, Berlin 1999, 151-165

[3] B Baum-Waidner, M.Waidner : Round-optimal and Abuse-free Optimistic Multi-Party Contract Signing : ICALP 2000, June 2000, 524-535

[4] B. Baum-Waidner : Optimistic Asynchronous Multi-Party Contract Signing with Reduced Number of Rounds ICALP 2001, July 2001, 898-911

[5] J.Garay, M. Jakobsson, P.MacKenzi :Abuse-free Optimistic Contract Signing : Crypto '99, LNCS1666, Springer-Verlag, Berline 1999, 449-466