오버레이 링을 이용한 이동 에이전트 간의 안전한 그룹 통신 기법

정 용 우[†] · 최 정 환[††] · 고 광 선[†††] · 김 구 수[††††] · 엄 영 익[†††††]

요 약

일반적인 멀티에이전트 시스템에서 에이전트들 간의 통신 기능을 지원하기 위한 다양한 모델들이 제안되어 왔으며, 특히, 그룹 통신 기능을 지원하기 위한 다양한 기법들이 제시되어 있다. 그러나, 이동 에이전트 환경에서는 에이전트의 이주로 인해 발생하는 토폴로지의 변화를 고려한 새로운 그룹 통신 기법을 필요로 하며, 더불어 이러한 그룹 통신 기능의 안전성을 보장하는 기법이 필요하게 된다. 본 논문에서는 계층적 오버레이 링을 이용한 이동 에이전트들 간의 안전한 그룹 통신 기법을 제안하며, 이 기법은 이동 에이전트의 이주에 따른 토폴로지의 변화에 유연하게 대응하기 위해 링 채널을 이용한다. 여기서 링 채널은 이동 에이전트들 간의 링 토폴로지를 구성하기 위한 객체로서 이동 에이전트 플랫폼에 의해서만 관리되며, 이에 따라 이동 에이전트들은 링 채널을 직접 관리할 필요가 없게 되고, 이주에 의해 발생하는 링 토폴로지 변화에 대한 고려 없이 그룹 통신을 진행할 수 있게 된다.

키워드 : 이동 에이전트, 그룹 통신, 링 토폴로지, 그룹 공유키

A Secure Group Communication Scheme for Mobile Agents using the Hierarchical Overlay Ring

Youngwoo Jung[†] · Jung Hwan Choi[††] · Kwang Sun Ko[†††] · Gu Su Kim[††††] · Young Ik Eom[†††††]

ABSTRACT

In multi agent systems, various inter agent communication models have been proposed, and, especially, there are several group communication schemes proposed so far, where some schemes guarantees transparent communication among the agents. However, in mobile agent environments, we require new group communication schemes that consider topology changes caused by mobile agent migrations. Also, these group communication schemes should be secure in order for them to be practical. In this paper, we propose a secure group communication scheme using the hierarchical overlay ring structure of mobile agents. The proposed scheme uses the ring channel in order to cope adaptively with the change of ring topology. The ring channel has basic information for construction of the ring and is managed only by the mobile agent platforms. Therefore, each mobile agent need not directly handle the ring channel and it can perform group communication without any consideration on the change of the ring topology.

Key Words : Mobile agents, Group communication, Ring topology, Group shared key

1. 서 론

이동 에이전트는 컴퓨터 네트워크 상에서 노드와 노드 사이를 자율적으로 이동하며 사용자를 대신하여 특정 연산을 수행하는 프로그램이다[1]. 이동 에이전트는 네트워크의 환경 변화에 동적이고 유연하게 적응하며, 비동기적으로 작업을 수행함으로써 대역폭이 낮은 네트워크 환경에서도 원활한 작업이 가능하다. 이러한 특징은 정보 검색, 네트워크 관리, 전자상거래 등의 다양한 분산 환경에서 유용하게 활용될 것으로 기대된다[2-4].

에이전트 간의 통신은 멀티에이전트 시스템들 사이에서 가장 근본적인 문제로써 이를 해결하기 위한 다양한 통신 모델들이 제안되었다[5, 6]. 특히, HFTRP(Hierarchical Fault Tolerant Ring Protocol)[7]와 SSRP(Secure Synchronous Ring Protocol)[8] 등의 그룹 통신 기법은 가장 널리 이용되는 모델로써 그룹을 구성하는 멤버들에게 투명한 메시지 전송을 보장한다. 그러나, 이러한 그룹 통신을 이동 에이전트

※본 연구는 21세기 프론티어 연구개발사업의 일환으로 추진되고 있는 정보통신부의 유비쿼터스컴퓨팅및네트워크원천기술개발사업의 지원에 의한 것임 (2007-0266-0100).
※본 연구는 정보통신부 및 정보통신연구진흥원의 대학 IT연구센터 지원사업의 연구결과로 수행되었음 (IITA-2007-(C1090 0701-0046))
† 준 회 원 : 성균관대학교 전자전기컴퓨터공학과 석사과정
†† 준 회 원 : 성균관대학교 휴대폰학과 석사과정
††† 정 회 원 : 성균관대학교 이동통신공학과 연구교수
†††† 정 회 원 : 동양대학교 정보통신공학부 교수
††††† 종신회원 : 성균관대학교 정보통신공학부 교수
논문접수 : 2007년 7월 13일, 심사완료 : 2007년 10월 17일

환경에서 적용하기 위해서는 이동 에이전트의 이주에 의해서 발생되는 토폴로지의 변화와 인증에 필요한 키 관리의 문제가 반드시 해결되어야 한다.

본 논문에서는 오버레이 링을 이용한 이동 에이전트간의 안전한 그룹 통신 기법을 제안한다. 본 기법은 이동 에이전트의 이주에 따른 토폴로지의 변화에 유연하게 대처하기 위해 링 채널을 이용한다. 링 채널은 이동 에이전트간의 링 토폴로지를 구성하기 위한 객체로써 이동 에이전트 플랫폼에 의해 관리된다. 플랫폼은 링 채널을 이용하여 링 구성에 필요한 멤버 인증, 그룹 공유키 생성 및 분배, 링 토폴로지 재구성 등을 담당한다. 따라서 본 기법에서 이동 에이전트는 링 구성에 필요한 세부 메커니즘과는 상관없이 그룹 통신을 할 수 있다.

본 논문의 구성은 다음과 같다. 2장에서는 그룹 통신을 지원하는 기존 프로토콜들을 살펴보고 3장에서는 본 논문에서 제안하는 오버레이 링을 이용한 이동 에이전트간의 안전한 그룹 통신을 위한 시스템 구조에 대해서 설명한다. 4장에서는 링 토폴로지 관리에 필요한 메커니즘을 살펴보고 5장에서는 본 기법의 인증 메커니즘과 키 분배 메커니즘에 대해서 설명한다. 6장에서는 본 기법의 안전성을 분석하고 7장을 통해 결론을 맺는다.

2. 그룹 통신 프로토콜

본 장에서는 분산 환경에 적합하게 설계된 대표적인 그룹 통신 프로토콜인 HFTRP(Hierarchical Fault Tolerant Ring Protocol)[7]와 SSRP(Secure Synchronous Ring Protocol)[8]를 설명한다. HFTRP는 분산 실시간 시스템을 위해 고안된 계층적인 통신 프로토콜이다. (그림 1)은 HFTRP를 나타낸다.

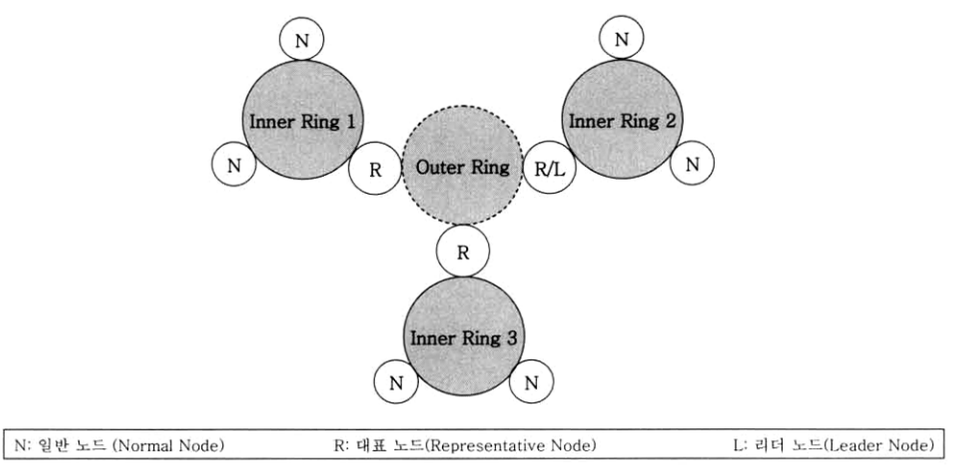
본 프로토콜에서 링은 일반 노드(N), 대표자 노드(R), 리더 노드(L)로 구성되며, 이들은 각자의 역할을 바탕으로 링의 계층적 관계를 유지한다. 각각의 링은 하나의 대표자 노드와 다수의 일반 노드들로 구성되며, 대표자 노드는 자신이 속한 내부 링의 일반 노드들의 효율적인 관리 및 내부 링을 한 단계의 상위 링인 외부 링과 연결하는 역할을 수행한다.

대표자 노드는 매 시간마다 빈 토큰을 생성하고, 이를 링을 통해 전송한다. 빈 토큰은 각 노드들을 거치면서 요청한 정보들을 받아 토큰 내부에 저장하고, 최종적으로 대표자 노드로 반환한다. 대표자 노드로 반환된 결과 토큰은 다시 상위 레벨의 링으로 보내지고, 상위 레벨에서도 이와 같은 작업을 반복적으로 수행함으로써 최종 결과 토큰을 리더 노드로 전송하게 된다. 노드들에 기반한 본 프로토콜 구조는 그룹의 크기, 통신 비용, 그리고 연산 오버헤드 등을 고려하여 계층의 깊이를 유연하게 결정할 수 있다.

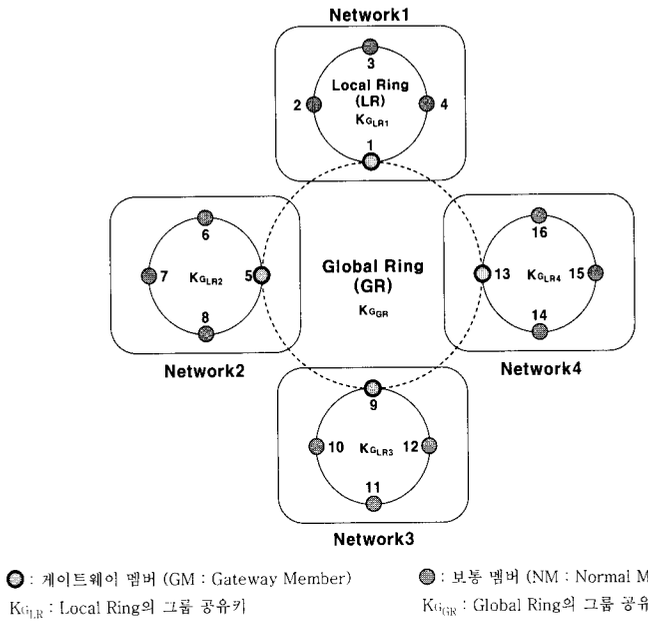
SSRP(Secure Synchronous Ring Protocol)은 동기적인 방식으로 동작하는 오류에 대한 내구성을 지닌 그룹 통신 프로토콜이다. 본 프로토콜은 계층적 관계를 갖는 그룹 통신 프로토콜로써, 안전한 그룹 통신을 위하여 Cliques의 GDH(Generalized Diffie-Hellman) 키 관리 프로토콜[9]과 Diffie-Hellman의 암호화 알고리즘을 사용한다. SSRP는 보안 기능을 기존의 그룹 통신 계층에 통합 시킴으로써, 그룹의 메시지 보호뿐만 아니라 그룹 통신 계층 내의 관련된 다른 정보들을 외부 공격으로부터 보호한다.

SSRP는 키 관리 프로토콜로 그룹 멤버십 수행 시 최소의 비용을 요구하는 GDH를 사용한다. GDH의 관리 구조는 링의 형태와 비슷하며, 링에서와 마찬가지로 대표 노드를 통한 그룹 키 관리를 수행한다. 또한 SSRP는 멤버의 등록 및 해제, 그룹의 등록 및 해제와 같은 멤버십 수행 시 독립적으로 각 링에 대한 그룹키를 생성하여 보다 뛰어난 안전성을 보장한다.

그러나 위에서 언급한 HFTRP와 SSRP 그룹 통신 프로토콜들은 뛰어난 성능을 보임에도 불구하고 이동 에이전트간의 그룹 통신을 지원하기에는 한계점을 가진다. 특히, 이동 에이전트의 이주에 의해 발생되는 토폴로지의 변화는 이동 에이전트간의 그룹 통신에 필요한 인증, 그룹 공유키 생성 및 분배, 토폴로지의 재구성 등의 문제를 야기시킴으로



(그림 1) HFTRP(Hierarchical Fault Tolerant Ring Protocol)의 시스템 구조



(그림 2) 계층적 오버레이 링에 기반한 제안 시스템 구조

써 그룹 통신을 어렵게 만든다. 따라서 본 논문에서는 계층적 오버레이 링을 통해 이동 에이전트간의 안전한 통신을 지원하는 그룹 통신 기법을 제안한다.

3. 계층적 오버레이 링에 기반한 제안 시스템 구조

본 장에서는 이동 에이전트간의 그룹 통신을 위한 오버레이 링 토폴로지의 구조를 살펴보고, 링 형성에 필요한 정보를 저장하는 링 채널에 대하여 설명한다.

3.1 링 토폴로지

본 기법은 계층적 구조를 가진 링을 이용하여 그룹 통신을 지원한다. 각 그룹의 모든 멤버는 이동 에이전트이며, 각각의 이동 에이전트는 자신이 이주할 수 있는 네트워크 영역을 인지하고 있다고 가정한다. 각 그룹을 구성하는 이동 에이전트는 논리적으로 링의 형태를 취한다. 링은 LR(Local Ring)과 GR(Global Ring)로 구성되며 이 두 가지의 링은 계층적 관계를 유지한다. LR은 동일한 네트워크 안에 존재하는 이동 에이전트 그룹 멤버를 연결하는 논리적인 링으로써 링에 연결되는 그룹 멤버는 일반멤버(NM: Normal Member)와 GM(Gateway Member)으로 구분된다. GR은 각 네트워크 마다 구성된 LR을 연결하기 위한 전역적인 링으로써 각 LR의 GM을 멤버로 취한다. GM은 LR을 구성하는 그룹 멤버 목록을 관리하며, GR과의 통신을 담당한다. (그림 2)는 본 논문에서 제안한 그룹 통신을 위한 시스템 구조를 나타낸다.

3.2 링 채널

각 링의 멤버들은 링 채널(RC: Ring Channel)이라는 객체를 통하여 논리적인 링을 형성한다. 링 채널은 링을 형성하기 위해 필요한 정보를 담고 있으며, 이동 에이전트 플랫폼에 의해 관리된다. 따라서, 이동 에이전트가 그룹 통신에 참여하고자 요청하면, 이동 에이전트 플랫폼은 해당 이동 에이전트에 대한 링 채널을 생성하고 이를 네트워크에 존재하는 LR에 연결함으로써 이동 에이전트의 그룹 통신을 지원한다. 링 채널은 이동 에이전트 플랫폼에 의해 관리되고 이동 에이전트가 다른 호스트로 이주 할때 같이 전송됨으로써, 링의 멤버는 링 토폴로지의 생성 및 소멸, 이주에 따른 링 토폴로지의 재구성, 그리고 그룹 공유키의 변경 등에 필요한 작업들을 고려하지 않고 그룹 통신을 수행할 수 있다. (그림 3)은 링 채널의 구조를 나타낸다.

$$RC = \{ RCID \mid PID \mid MAID \mid RCType \mid GroupID \mid GSK \mid P\_Link \mid N\_Link \}$$

RCID	: 링 채널의 ID
PID	: 현재의 링 채널을 관리하는 이동 에이전트 플랫폼의 ID
MAID	: 현재의 링 채널을 사용하는 이동 에이전트의 ID
RCType	: LR과 GR을 구분하기 위한 링의 타입
GroupID	: 현재의 링 채널이 참여하는 링의 ID
GSK	: 현재의 링에서 사용하는 그룹 공유키
P_Link	: 앞에 이웃한 멤버를 가리키기 위한 링크
N_Link	: 뒤에 이웃한 멤버를 가리키기 위한 링크

(그림 3) 링 채널 구조

*P_Link*와 *N_Link*는 이웃한 멤버를 가리키기 위한 링크로써 *PID*와 *RCID*를 필드로 갖는 구조체이다. 이웃 멤버는 같은 호스트에 존재하거나 다른 호스트에 있을 수 있으므로 PID 링크를 통해 이를 확인할 수 있다.

4. 링 관리 메커니즘

본 장에서는 링의 구성에 필요한 메커니즘을 살펴보고, 이동 에이전트의 이주에 따른 링 토폴로지의 재구성 기법에 대하여 설명한다.

4.1 이동 에이전트의 그룹 등록

이동 에이전트는 그룹에 참여하기 위해 플랫폼에게 그룹 등록을 요청한다. (그림 4)는 플랫폼에서 수행되는 그룹 등록 알고리즘을 나타낸다.

이동 에이전트는 그룹 ID를 이용하여 플랫폼에게 그룹 등록 요청을 한다. 플랫폼은 등록을 요청한 이동 에이전트의 링 채널을 생성한다. 우선, 플랫폼은 자신의 플랫폼 상에서 동작하고 있는 이동 에이전트 중 동일 그룹에 참여하고 있는 이동 에이전트가 존재하는지 확인한다. 만일 멤버가 존재한다면, 플랫폼은 SRC (Simple RC Connection) 연결을 수행하여 새로운 링 채널을 기존 멤버의 링 채널들과 연결함으로써 그룹 등록을 완료한다.

만일, 자신의 플랫폼 상에 요청한 그룹의 멤버가 존재하지 않는다면, 플랫폼은 LR탐색을 통한 그룹 등록을 수행한다. LR탐색은 동일한 네트워크에 형성된 LR의 GM을 확인한다. LR탐색을 위해 플랫폼은 *LR_DISCOVERY* 메시지를 로컬 네트워크로 브로드캐스트한다. 다음은 *LR_DISCOVERY* 메시지의 구조를 나타낸다.

< *LR_DISCOVERY, PlatformID, MobileAgentID* >

플랫폼이 *LR_DISCOVERY* 메시지를 브로드캐스트했을 때, 동일 네트워크 상에 LR이 존재한다면, LR의 GM은 *LR_DISCOVERY* 메시지를 전송 받는다. GM은 그룹 등록을 요청한 이동 에이전트에 대해 멤버 인증을 수행한다. 인증에 성공하면, GM은 이동 에이전트에게 그룹 공유키(K_G)를 분배한다. 그룹 공유키를 분배 받은 이동 에이전트는 링크 인증을 통해 그룹 멤버간의 링크를 형성한다. 멤버 인증과 링크 인증은 5장에서 설명한다.

LR탐색에 실패하면, 플랫폼은 그룹 등록을 요청한 이동 에이전트를 새로운 LR의 GM으로 설정한다. GM의 설정이 완료되면, 플랫폼은 *GR_DISCOVERY* 메시지를 이용한 GR 탐색을 실행한다. GR탐색은 LR탐색과 동일하게 수행된다. GR탐색이 완료되면 GM은 GR의 멤버로 등록된다.

```
void join(MobileAgent ma, int groupID){
  RingChannel new_lrc = makeRingChannel(ma, groupID);
  bexist = Search_LR(groupID);
  if(bexist == true) {
    RingChannel rc = getRingChannel(groupID);
    Simple_RC_Connection(rc, new_lrc);
  } else {
    msg = discoveryLR(groupID);
    if(msg.bfind == true) {
      joinLR(new_lrc, msg.previousLink, msg.nextLink);
    } else {
      makeGateway(ma, groupID);
      msg = discoveryGR(groupID);
      if(msg.bfind == true) {
        RingChannel new_grc = makeRingChannel(ma, groupID, GR);
        joinGR(new_grcm msg.previousLink, msg.nextLink);
      }
    }
  }
}
```

(그림 4) 그룹 등록 알고리즘

4.2 그룹 해제

멤버가 그룹으로부터 해제를 원할 경우, 멤버는 플랫폼에게 그룹해제를 요청한다. 플랫폼 P_i에 위치한 이동 에이전트 α가 그룹으로부터 해제를 원할 경우, 플랫폼 P_i는 α의 *P_Link*와 *N_Link*로 연결된 멤버 α_P와 α_N에게 *LINK_REFRESH* 메시지를 각각 전송하고 α의 링 채널을 삭제함으로써 그룹 해제를 완료한다. *LINK_REFRESH* 메시지는 다음과 같이 구성된다.

< *LINK_REFRESH, Target_Link*>

LINK_REFRESH 메시지의 *Target_Link*는 새롭게 연결할 링 채널의 ID와 그 채널이 있는 플랫폼 ID로 구성된다. 예를 들어, α_P에게 전송되는 메시지는 <*LINK_REFRESH, N_Link of RC for a*>와 같이 구성된다. *LINK_REFRESH* 메시지를 받은 α_P와 α_N의 플랫폼은 해당 이동 에이전트의 링 채널 값을 변경함으로써 링 토폴로지를 재구성하고 그룹 해제를 마친다.

그룹 해제를 요구한 이동 에이전트 α가 GM일 경우, 플랫폼 P_i는 α의 *P_Link*와 *N_Link*로 연결된 멤버 α_P와 α_N에게 *LINK_REFRESH* 메시지를 각각 전송하고, GM의 역할을 α_N에게 위임한다. α_N을 관리하는 플랫폼 P_N은 플랫폼 P_i로부터 LR을 구성하는 그룹 멤버 목록과 GR에 참여하기 위해 필요한 링 채널을 전송 받는다. 플랫폼 P_N은 전송 받은 링 채널을 이용하여 α_N을 GR의 멤버로 등록함으로써 GM 위임 작업을 완료한다. GM 위임이 완료되면, P_i는 α의 링 채널을 삭제함으로써 그룹 해제를 마무리한다.

4.3 이동 에이전트의 이주에 따른 링 토폴로지의 재구성

그룹에 참여하던 이동 에이전트가 다른 호스트로 이주할 경우, LR 및 GR의 링 토폴로지는 변경될 수 있다. 본 장에서는 동일 네트워크 내에서의 이동 에이전트의 이주와 다른 네트워크로의 이동 에이전트 이주에 따른 링 토폴로지의 변화와 재구성 기법을 살펴본다.

동일 네트워크 내에서의 이동 에이전트의 이주는 오버레이 링의 토폴로지에 변화를 발생시키지 않는다. 동일 네트워크에 있는 플랫폼 P_i에서 플랫폼 P_j로 이동 에이전트가 이주할 경우, 플랫폼 P_i는 이동 에이전트의 링 채널을 P_j로 전송한다. 이동 에이전트의 이주가 완료되면, P_j는 링 채널의 PID의 값을 P_j의 PID로 수정하고 P_Link와 N_Link의 해당 플랫폼에게 이주한 플랫폼에게 이주한 플랫폼의 PID 수정을 요청한 뒤, 이동 에이전트를 재 실행함으로써 그룹 통신이 가능하다.

이동 에이전트 α가 현재 플랫폼 P_i에서 다른 네트워크에 있는 플랫폼 P_j로 이주하는 경우, 우선 α는 플랫폼 P_i에게 그룹 해지를 요청한다. P_i는 그룹 해지에 필요한 작업을 수행한다. 그룹 해지가 완료되면, α는 P_j로 이주한 뒤, P_j에게 그룹 등록을 요청한다. P_j는 α를 LR_j의 일반 멤버로 등록함으로써 토폴로지 재구성을 완료한다. 그룹 해지와 그룹 등록은 위에서 설명한 그룹 해지 알고리즘과 그룹 등록 알고

리즘을 통하여 수행된다.

본 기법에서 이동 에이전트의 이주에 따른 링 토폴로지의 재구성은 이동 에이전트 플랫폼이 담당한다. 따라서, 본 기법에서 이동 에이전트는 링 토폴로지의 형성에 필요한 링 채널을 직접적으로 다루지 않기 때문에 이주에 의해 발생하는 링 토폴로지의 변화를 고려하지 않고 그룹 통신이 가능하다.

5. 인증 및 키 관리

본 장에서는 새 그룹 멤버에 대한 인증기법과 안전한 그룹 통신을 위한 키 관리 기법에 대해서 설명한다. 본 논문에서 제안한 인증 기법은 멤버 인증과 링크 인증으로 구성된다. 멤버 인증은 그룹에 등록을 원하는 새 멤버에 대한 인증으로 인증에 성공한 이동 에이전트에게 그룹 공유키를 전송한다. 링크 인증은 그룹을 구성하고 있는 이동 에이전트간의 링크에 관한 인증으로써 링 토폴로지가 구성될 때 수행된다.

5.1 멤버 인증 및 키 분배

본 인증 기법에서의 이동 에이전트는 안전한 채널(Secure Channel)을 통하여 이주를 수행하며 모든 이동 에이전트는 그룹 프로파일(Group Profile)을 통하여 그룹의 정보를 소유한다고 가정한다.

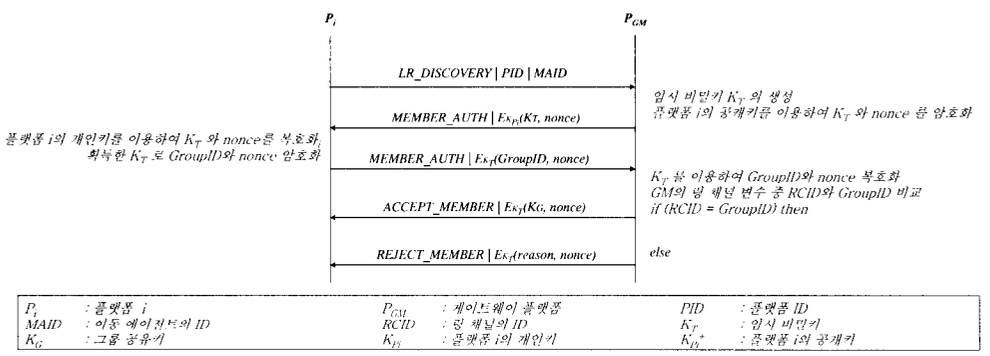
이동 에이전트가 플랫폼에게 그룹 등록을 요청하면, 플랫폼은 멤버 인증을 수행한다. SRC연결을 통한 그룹 등록이 수행될 경우, 플랫폼은 등록을 요청한 이동 에이전트의 그룹 ID를 플랫폼 상에 존재하는 이동 에이전트의 링 채널 GroupID 필드와 비교하여 인증을 수행한다.

LR탐색을 통한 그룹 등록이 수행될 경우, 그룹의 GM을 관리하는 플랫폼 P_{GM}은 새 멤버에 대한 인증을 수행하고, 그룹 공유키를 배포한다. (그림 5)는 P_{GM}에서 수행되는 인증 프로토콜을 보인다.

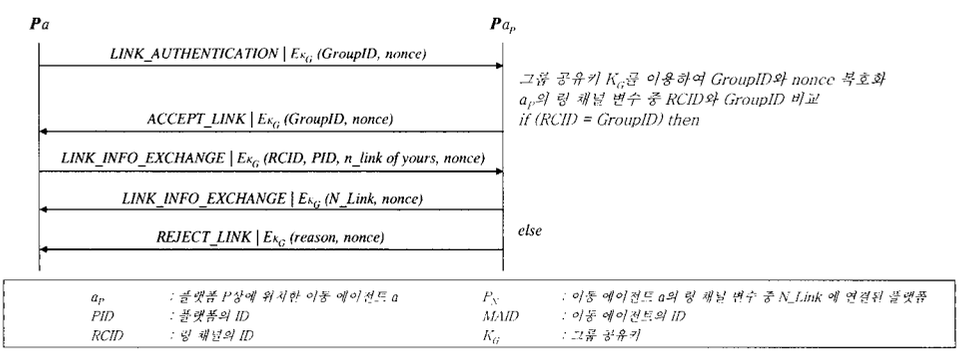
P_i는 *LR_DISCOVERY* 메시지를 P_{GM}에게 전송한다. *LR_DISCOVERY* 메시지를 수신한 P_{GM}은 임시 비밀 키 K_T를 생성하고 P_i의 공유키 K_{Pi}^+로 암호화 하여 P_i에게 전송한다. P_i는 개인키 K_{Pi}로 복호화하여 K_T를 추출한 뒤, GroupID를 K_T로 암호화하여 P_{GM}에게 전송한다. P_{GM}은 K_T로 복호화하여 GroupID를 추출하고 이 값을 GM의 링 채널 GroupID와 비교한다. 두 값이 동일하다면, P_{GM}은 그룹 공유키 K_G를 K_T로 암호화 하여 P_i에게 전송함으로써 인증을 완료한다.

5.2 링크 인증

인증에 성공한 이동 에이전트는 그룹의 다른 멤버들과 연결을 통해 그룹의 멤버가 된다. 이동 에이전트 a는 그룹의 멤버 a_P와 a_N에게 연결을 요구한다. a_P와 a_N는 링크 인증을 통해 a와의 연결을 수락한다. 링크 인증은 *LINK_AUTHENTICATION* 메시지를 이용하여 그룹의 멤버와 새 멤버간의 연결에 대한 유효성을 판단한다. (그림 6)은



(그림 5) 멤버 인증 프로토콜



(그림 6) 링크 인증 프로토콜

LINK_AUTHENTICATION 메시지를 이용한 링크 인증 프로토콜을 보인다.

(그림 6)은 이동 에이전트 α와 α_P간의 링크 인증을 나타낸다. 플랫폼 Pα는 *LINK_AUTHENTICATION* 메시지를 Pa_P에게 전송한다. *LINK_AUTHENTICATION* 메시지는 다음과 같이 구성된다.

< *LINK_AUTHENTICATION*, EK_G (*GroupID, nonce*) >

Pa_P는 K_G를 이용하여 GroupID를 추출하고 이것을 α_P의 링 채널 GroupID와 비교한다. 두 값이 동일하면, Pa_P는 Pα에게 링크를 허가하고 링크에 필요한 정보를 서로 교환함으로써 링크인증을 완료한다. .

6. 안전성 검증

본 장에서는 본 논문에서 제안한 오버레이 링을 이용한 이동 에이전트간의 안전한 그룹 통신 기법을 검증한다. 검증에 앞서 본 기법에서 사용한 암호화된 메시지는 공격자의 위변조 공격으로부터 충분히 안전하다고 가정한다. <표 1>은 제안 기법의 검증을 위한 표기법을 나타낸다.

본 기법에서 제안한 그룹 통신을 위한 링 토폴로지는 다음과 같다.

$$\text{Network } N = \{LR_1, LR_2, \cdots, LR_n\}$$
$$LR_i = R(nm_1, nm_2, \cdots, nm_n, gm_i)$$
$$GR = R(gm_1, gm_2, \ldots, gm_n)$$

〈표 1〉 검증에 사용되는 표기법

표 기	설 명
$Enc_{K_{Pi}}(m)$	메시지 m을 홈 플랫폼 i의 공개키 K_U로 암호화
$Dec_{K_{Pi}}(m)$	메시지 m을 홈 플랫폼 i의 개인키 K_P로 복호화
K_G	그룹 공유키
K_T	One-Time 비밀 키
P_{MA}	이동 에이전트 MA가 동작하는 플랫폼
$R(a_1, a_2, \ldots, a_n)$	멤버 $a_1, \ldots, a_n$으로 구성된 링 토폴로지
$M(m_1, m_2, \ldots, m_n)$	$m_1, m_2, \ldots, m_n$으로 구성된 메시지 생성
$Ext(m, a_i)$	메시지 m으로부터 원소 a_i를 추출
$Adv(m)$	메시지 m에 대한 공격자의 공격
$A \rightarrow B:m$	A로부터 B로의 메시지 m 전송

(정리 1) 본 기법의 인증 프로토콜은 안전한 그룹 공유키의 분배를 보장한다.

증명:

P_i : $m_0 = M(LR\_DISCOVERY, PID, MAID)$

$P_i \rightarrow P_{GM}$: m_0

P_{GM} : $m_A = Enc_{K_{Pi}}(K_T, nonce)$

$P_{GM} \rightarrow P_i$: m_A

P_i : $Ext(Dec_{K_{Pi}}(m_A), K_T)$

: $m_{A'} = Enc_{K_T}(GroupID, nonce)$

$P_i \rightarrow P_{GM}$: $m_{A'}$

P_{GM} : $COMPARE(Ext(Dec_{K_T}(m_{A'}), GroupID), GroupID\ of\ RC\ for\ GM)$

: $m_G = Enc_{K_T}(K_G, nonce)$

$P_{GM} \rightarrow P_i$: m_G

P_i : $Ext(Dec_{K_T}(m_G), K_G)$

if $m_{adv} = Adv(m_G)$ *then*

$P_{GM} \rightarrow P_i$: m_{adv}

P_i : $Ext(Dec_{K_T}(m_{adv}), K_G)$

따라서, 공격자에 의한 K_G의 변형을 확인하고 인증 프로토콜을 반복 수행함으로써 안전한 그룹 공유키의 분배를 보장한다. ■

(정리 2) 본 인증 프로토콜은 메시지 재전송 공격으로부터 안전하다.

증명:

Adversary : $m_{captured} = Adv(m_A)\ or\ Adv(m_G)$

Adversary $\rightarrow P_{GM}$: $m_{captured}$

P_{GM} : $COMPARE(Ext(Dec_{K_T}(m_{captured}), nonce), TIME)$

따라서, P_{GM}은 메시지가 재전송 되었음을 확인하고 인증을 거부한다. ■

(정리 3) 본 인증 프로토콜은 메시지 송수신 부인 공격으로부터 안전하다.

증명:

P_i : $m_0 = M(LR\_DISCOVERY, PID, MAID)$

$P_i \rightarrow P_{GM}$: m_0

P_{GM} : $m_A = Enc_{K_{Pi}}(K_T, nonce)$

$P_{GM} \rightarrow P_i$: m_A

P_i : $Ext(Dec_{K_{Pi}}(m_A), K_T)$

$Ext(Dec_{K_{Pi}}(m_A), K_T) = Ext(Dec_{K_{Pi}}(Enc_{K_{Pi}}(K_T, nonce)), K_T) = K_T$

따라서, 공개키 암호화 알고리즘의 일반적인 성질에 의해 P_{GM}와 P_i는 각각 송, 수신을 부인할 수 없다. ■

(정리 4) 링크 인증 프로토콜은 멤버간의 안전한 링크를 보장한다.

증명:

$P_i \rightarrow P_{iN}$: $m_0 = M(LINK\_AUTHENTICATION, Enc_{K_G}(GroupID, nonce))$

P_{iN} : $COMPARE((Dec_{K_G}(M_0), GroupID), GroupID\ of\ RC\ for\ P_{iN})$

: $m_{Accept} = Enc_{K_G}(GroupID, nonce)$

$P_N \rightarrow P_i$: m_{Accept}

P_i : $m_{Link} = Enc_{K_G}(RCID, PID, n\_link\ of\ yours, nonce)$

$P_i \rightarrow P_{iN}$: $Ext(Dec_{K_T}(m_{Link}), n\_link\ of\ yours)$

: $m_{Link} = Enc_{K_G}(N\_Link, nonce)$

$P_{iN} \rightarrow P_i$: m_{Link}

if $m_{adv} = Adv(m_0)$ *then*

P_{iN} : $COMPARE((Dec_{K_G}(M_{adv}), GroupID), GroupID\ of\ RC\ for\ P_{iN})$

However, P_{iN} *cannot decrypt* m_{adv} *with* K_T *so* P_{iN} *can detect message transformation.*

Therefore, P_{iN} *requires* m_0 *to* P_i *by next process*

P_{iN} : $m_{Reject} = Enc_{K_G}(m_{adv}, nonce)$

$P_{iN} \rightarrow P_i$: m_{Reject}

P_{iN} : $COMPARE(Ext(m_{Reject}, m_{adv}), m_0)$

: $m_0 = M(LINK\_AUTHENTICATION, Enc_{K_G}(GroupID, nonce))$

따라서, Pi_N는 공격자에 의한 m_0의 변형을 확인하고 메시지 재송신을 요청함으로써 멤버간의 안전한 링크를 보장한다. ■

(정리 5) 링크 인증 프로토콜은 메시지 재전송 공격으로부터 안전하다.

증명:

$Adversary$: $m_{captured} = Adv(m_0)$ or $Adv(m_{Ink})$

$Adversary \rightarrow P_{iN}$: $m_{captured}$

P_{iN} : $COMPARE(Ext(Dec_{K_G}(m_{captured}), nonce), TIME)$

따라서, Pi_N은 메시지가 재전송 되었음을 확인하고 멤버의 링크를 거부한다. ■

(정리 6) 오버레이 링을 이용한 이동 에이전트간의 메시지 전송은 안전하다.

증명:

(1) 링을 구성하고 있는 모든 멤버는 정리 1, 정리 2, 정리 3을 통해 안전하게 그룹 공유키를 한다.

(2) 링을 구성하고 있는 모든 멤버는 정리 4, 정리 5를 통해 멤버간의 안전한 링크를 보장한다.

(3) 가정에 의해 그룹 공유키에 의해 암호화된 메시지는 공격자의 위변조 공격으로 충분히 안전하다

따라서, 그룹 공유키 K_G에 의해 암호화된 메시지 MK_G는 멤버간의 안전한 링크를 통해 링을 구성하고 있는 모든 에이전트에게 안전하게 전송된다. ■

7. 결 론

본 논문에서는 계층적 오버레이 링을 이용한 이동 에이전트 간의 안전한 그룹 통신 기법을 제안하였다. 본 기법은 계층적인 오버레이 링을 사용함으로써 서로 다른 네트워크에 존재하는 이동 에이전트간의 투명한 메시지 전송이 가능하다. 또한, 본 기법은 이동 에이전트의 이주에 따른 링 토폴로지의 변화에 유연하게 대처하기 위해 링 채널을 사용하였다. 링 채널은 논리적인 링을 형성하기 위한 객체로서 플랫폼에 의해서 관리된다. 따라서 이동 에이전트는 링 토폴로지의 형성에 필요한 링 채널을 직접적으로 다루지 않기 때문에 이주에 의해 발생하는 링 토폴로지의 변화를 고려하지 않고 그룹 통신을 할 수 있다.

본 논문에서는 제안한 기법에 대한 안전성을 평가하였다. 본 논문에서는 본 기법의 인증 프로토콜이 안전한 그룹 공유키의 분배를 보장하며, 메시지 재전송공격과 송수신 부인 공격으로부터 안전함을 보였다. 또한 링크 인증 프로토콜 역시 메시지 재전송 공격으로부터 안전하며 멤버간의 안전한 링크를 보장함을 보였으며 이를 통해 오버레이 링을 이용한 이동 에이전트간의 그룹 통신은 충분히 안전함을 증명하였다.

참 고 문 헌

[1] K. Cho, H. Hayashi, M. Hattori, A. Ohsuga, and S. Honiden, "picoPlangent: An Intelligent Mobile Agent System for Ubiquitous Computing," LNAI 3371, Springer-Verlag, 2005.

[2] A. Aneiba and S. J. Rees, "Mobile Agents Technology and Mobility," Proc. of the 5th Annual Postgraduate Symposium on the Convergence of Telecommunications, Networking, and Broadcasting, Jun. 2004.

[3] K. Takashio, G. Soeda, and H. Tokuda "A Mobile Agent Framework for Follow-Me Applications in Ubiquitous Computing Environment," Proc. 21st International Conference on Distributed Computing System Workshop, Apr. 2001.

[4] T. Ledoux and N. Bouraqadi-Saadani, "Adaptability in Mobile Agent Systems using Reflection," Proc. of the Workshop on Reflective Middleware (RM2000), New York, USA, Apr. 2000.

[5] Q. H. Mahmoud, Understanding Network Class Loaders, Developer Technical Articles & Tips, Oct. 2004.

[6] P. Mihailescu, W. Binder, and E. Kendall, "MAE: Mobile Agent Environment for Resource Limited Devices," Proc. of the 8th ECOOP Workshop on Mobile Object System: Agent Applications and New Frontiers, Jun. 2002.

[7] T. Tunal, K. Erciye, and Z. Soysert, "A Hierarchical Fault-Tolerant Ring Protocol for Distributed Real-Time Systems," Proc. of Special issue of Parallel and Distributed Computing Practices on Parallel and Distributed Real-Time Systems, 2000.

[8] O. Saglam, M. Dalkilic, and K. Erciyes, "Design and Implementation of a Secure Group Communication Protocol on a Fault Tolerant Ring," Proc. of Computer and Information Sciences (ISCIS 2003), 2003.

[9] M. Steiner, G. Tsudik, and M. Waidner, "Key Agreement in Dynamic Peer Group," IEEE Trans. on Parallel and Distributed Systems. Vol. 11(8), pp. 769-781, 2000.

정 용 우

e-mail : withdubu@ece.skku.ac.kr

2006년 성균관대학교 정보통신공학부 (학사)

2006년~현재 성균관대학교 전자전기 컴퓨터공학과 석사과정

관심분야 : 이동 에이전트, 이동 에이전트 시스템 보안, 유비쿼터스 컴퓨팅 등

최 정 환

e-mail : themars@ece.skku.ac.kr
2007년 성균관대학교 컴퓨터공학과 (학사)
2007년~현재 성균관대학교 휴대폰학과 석사과정
관심분야 : 이동 에이전트, 이동 에이전트 시스템 보안, HCI 컴퓨팅 등

김 구 수

e-mail : gusukim@dyu.ac.kr
1994년 성균관대학교 정보공학과 (학사)
1996년 성균관대학교 정보공학과 (공학석사)
2006년 성균관대학교 정보통신공학과 (공학박사)
2007년~현재 동양대학교 정보통신공학부 교수
관심분야 : 분산 컴퓨팅, 이동 에이전트 등

고 광 선

e-mail : rilla91@ece.skku.ac.kr
1998년 성균관대학교 정보공학과 (학사)
2004년 성균관대학교 전기전자 및 컴퓨터 공학부(공학석사)
2007년 성균관대학교 전자전기컴퓨터 공학과(공학박사)
2007년~현재 성균관대학교 이동통신공학과 연구교수
관심분야 : 정보보호, 리눅스, 네트워크 등

엄 영 익

e-mail : yieom@ece.skku.ac.kr
1983년 서울대학교 계산통계학과 (학사)
1985년 서울대학교 전산과학과 (이학석사)
1991년 서울대학교 전산과학과 (이학박사)
2000년~2001년 Dept. of Info. and Comm. Science at UCI 방문교수
2005년 한국정보처리학회 학회지 편집위원장
1993년~현재 성균관대학교 정보통신공학부 교수
관심분야 : 분산 컴퓨팅, 이동 컴퓨팅, 이동 에이전트, 시스템 보안, 운영체제, 내장형 시스템 등