Windows 2000에서 Mobile IPv4를 위한 Mobile Node의 설계와 구현

서영주, 박진(ETRI), 장희진[0], 임채태
포항공과 대학교 컴퓨터 공학과
{yjsuh, spider, arche[0], chtim}@postech.ac.kr

The Design and Implementation of Mobile Node for Mobile IPv4 in Windows 2000

Young-Joo Suh, Jin Park, Hee-Jin Jang, Chae-Tae Im
Department of Computer Science & Engineering, Pohang University of Science & Technology

요 약

컴퓨터의 이동성을 지원하기 위해 1996년 IETF (Internet Engineering Task Force)에서 mobile IPv4를 제정하였고 이에 따른 구현 사례가 등장하였다. 그러나 현재까지 국내외 대학에서 발표된 구현 사례는 주로 유닉스 기반의 플랫폼(리눅스, BSD)에서 개발되었으며 가장 보편적으로 사용되고 있는 운영체제인 윈도우에서는, 라우팅 정보가 변경되면 재부팅을 해야 하는 결함때문에 개발에 어려움이 있었다. NT 기반의 윈도우즈가 출시됨에따라 윈도우 플랫폼에서 Mobile IP의 구현이 가능하게 되었으며 본 연구실에서는 윈도우즈 2000 플랫폼에서 Mobile IPv4를 위한 mobile node를 설계하여 POSTECH MIP mobile node를 개발하였다. POSTECH MIP mobile node는 확장성과 효율성을 목표로 설계되어 커널 스페이스(kernel space)와 유저 스페이스 (user space)에서 동작하도록 구현되었다.

1. 서 론

최근 하드웨어와 통신기술의 발달로 인해 음성뿐만 아니라 데이터의 전송까지 요구되는 이동 컴퓨팅 시대로 접어들게 되었다. IETF (Internet Engineering Task Force)는 이러한 요구에 맞춰 컴퓨터의 이동성을 지원할 수 있는 새로운 인터넷 프로토콜인 Mobile IP를 제정하여 1996년에 Mobile IP의 표준[1]을 발표하였다. 이에따라 Mobile IP의 표준에 따른 Mobile IP의 구현사례도 등장했는데 외국의 경우, 스탠포드 대학과 싱가폴 국립대학교에서는 리눅스 상에서[2,3], 카네기멜론 대학교에서는 FreeBSD상에서[4] 구현한 예가 있었다. 당시 가장 보편화된 운영체제인 윈도우에서는 루마니아의 부카레스트 대학이 윈도우 NT에서 Mobile IP를 개발하였다[5]. 그러나 라우팅 정보의 변경 시 재부팅을 요하는 운영체제의 결함 때문에 Mobile IP의 구현에 문제가 있었으며 이러한 운영체제의 문제점을 극복한 윈도우즈 2000이 출시되면서 윈도우에서의 Mobile IP의 구현이 가능해 졌다. 본 논문에서는 네트워크 코드가 한층 안정화 되어있는 윈도우즈 2000 플랫폼에서 개발된 POSTECH MIP mobile node의 설계 및 구현에 대해 기술한다.

2. 배경 지식

NDIS (Network Driver Interface Specification)는 네트워크 카드와의 인터페이스를 위한 표준 API (Application Program Interface)를 정의해 놓은 것이다. NDIS는 네트워크 디바이스 드라이버에서 하드웨어 작동과 프로토콜 작업을 분리하여 두 가지 타입 (miniport driver, protocol driver)에 대한 표준 API를 제공하여 각 네트워크 벤더나 플랫폼에 의존하지 않고 독립적으로 드라이버(driver)를 쉽게 개발할 수 있는 환경을 제공한다. NDIS 드라이버는 프로토콜 드라이버, 중계(intermediate) 드라이버, 미니포트(miniport) 드라이버로 나뉘는데 POSTECH MIP mobile node의 커널 스페이스의 구현 부분은 중계 드라이버에서 구현하였다. 중계 드라이버를 채택한 이유는 프로토콜 드라이버를 사용할 경우 패킷을 완전히 후킹할 수 없다는 단점 때문이다. 기타 중계 드라이버와 NDIS 함수에 대한 설명은 참조된 문헌[6,7]과 DDK (Driver Developers Kit)[8]에서 제공되는 문서를 참조하도록 한다. 또한 SDK(Software Development Kit)[9]에서 제공되는 함수를 이용하여 ARP (Address Resolution Protocol) 캐쉬와 라우팅 테이블의 정보를 변경하였다.

3. 설계 및 구현

3.1. 설계 목표

POSTECH Windows MIP mobile node는 효율성, 유연성, 확장성을 고려하여 유저 스페이스 (application)와 커널 스페이스 (driver)에서 동작하도록 설계되었다. 국내외 대학의 구현 사례를 살펴보면 커널 스페이스 (NUS Mobile IP)[3] 혹은 유저 스페이스 (Binghamton Mobile IP)[10]에서만 구현한 사례가 있다. 전자의 경우에는 등록(registration)을 포함한 모든 처리가 커널 스페이스에서 이뤄지므로 유저 스페이스에서 구현을 한 경우보다 처리속도가 빠르다. 그러나 기존의 TCP/IP 코드를 많이 수정해야 하고 multicast나 QoS (Quality of Service)등을 위한 확장에 어려움이 있었다. 또한 GUI (Graphic User Interface)를 제공할 수 없으며 보안에 관련된 암호 관리 시스템의 변경 및 추가 시에 드라이버를 재컴파일/재설치해야 하므로 확장성과 유연성이 떨어지는 단점이 있다. 반면 유저 스페이스에서 구현하는 경우 확장성은 뛰어나지만 처리 속도가 떨어지며 ARP나 라우팅 등의 동작을 직접 제어하지 못하는 문제점을 가진다. 따라서 POSTECH MIP mobile node는 Mobile IP의 모듈을 각각 분리하여 커널에서 직접 제어를 요하는 부분은 커널 스페이스에서 구현하고 확장이나 변경이 이루어 질 수 있는 부분은 유저 스페이스에서 구현함으로써 확장성과 유연성을 추구하면서도 효율적으로 동작할 수 있도록 설계되었다. 또한 GUI를 통하여 사용자가 편리하게 사용할 수 있도록 하였다.

3.2 설계 및 구현

POSTECH MIP mobile node는 C++ 언어를 사용하여 구현되었으며 전체적으로 그림 1과 같이 설계되었다. 본 절에서는 커널 스페이스와 유저 스페이스로 나누어 기술하고자 한다.

3.2.1 POSTECH MIP mobile node의 유저 스페이스

유저 스페이스에서는 에이전트 광고를 수신하여 핸드오프 결정을 수행하는 모듈, 드라이버로부터 에이전트 광고를 수신했는지 질의(polling)하는 스레드 모듈 (ADVT스레드), 등록을 요청하고 이

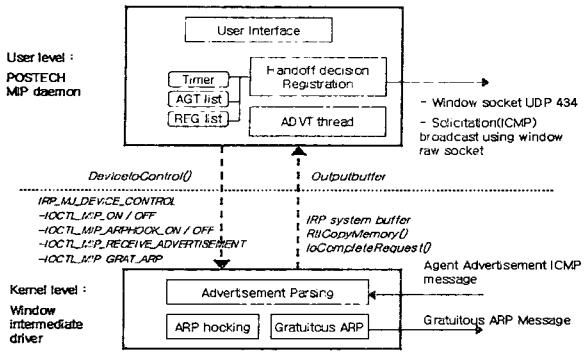

그림 1. POSTECH Windows Mobile Node 설계

에 대한 응답를 분석하여 처리하는 모듈, 사용자와 인터페이스하는 GUI 모듈, 타이머 핸들러 모듈로 이루어져 있다. ADVT 스레드는 드라이버에 에이전트 광고가 수신되었는지를 폴링하여 핸드오프 결정 모듈로 에이전트 광고를 전달하는 역할을 한다. 또한 유저 스페이스에서는 IOCTL (Input Output ConTroL)을 통해 유저 스페이스가 커널 스페이스의 동작을 제어할 수 있도록 구현하였다. 유저 스페이스에서 각 모듈을 구현하기 위한 클래스 설계는 그림 2와 같다. 우선 CRegList, CAgentList, CKeyList들은 각각 binding 정보, 암호 정보, 에이전트 광고를 전송한 에이전트 정보

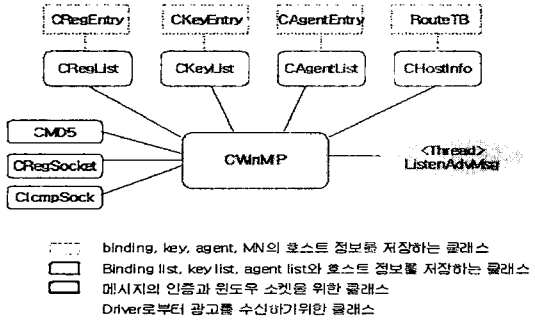

그림 2. 유저 스페이스의 클래스 설계

를 저장하고 있는 리스트이며 CHostInfo는 mobile node의 여러 가지 정보를 저장하고 있는 구조체이다. CWinMIP 클래스는 4개의 주요 클래스를 바탕으로 핸드오프 결정과 등록을 수행한다. 각 타이머 핸들러들은 전역으로 선언된 콜백 함수로 구현하였다. 그 외에도 MD5[11]는 메시지의 인증에 필요한 클래스이며 CRegSocket과 CIcmpSocket은 각각 등록 메시지의 송수신을 위한 윈도우즈 소켓과 광고 요청 메시지 (solicitation) 전송을 위한 윈도우즈 raw 소켓을 이용하기 위한 클래스이다. 유저 스페이스에서 mobile node는 휴지 상태, 등록 요청 상태, 등록 상태, 등록 연장 상태 중 하나에 속해 있는데 광고 메시지나 등록 요청에 대한 응답을 받거나 타이머가 만료되면 상태를 전이한다. 상태 전이는 그림 3과 같으며 각 상태의 의미는 다음과 같다.

◇ 휴지 상태 (Idle state) - 등록되지 않은 상태
◇ 등록 요청 상태 (Request state) -등록 요청 메시지는 전송했으나 응답을 수신하지 못한 상태
◇ 등록 상태 (Registered state) - 등록된 상태
◇ 등록 연장 상태 (Extend state) - 등록 연장을 요청하였으나 응답을 수신하지 못한 상태

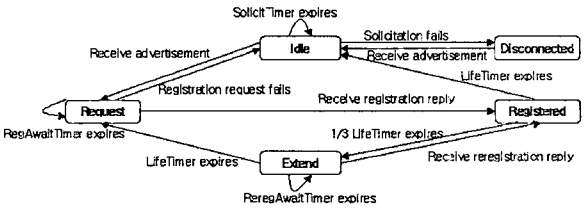

그림 3. mobile node의 상태 전이도

유저 스페이스에서 핸드오프 결정 알고리즘은 그림 4와 같다. 우선 에이전트 광고가 홈 네트워크에서 수신된 것이면 존재하는 모든 등록을 해제한다. 그렇지 않고 외부 네트워크로부터 광고를 수신하였다면 새로운 에이전트로부터 수신한 광고인지를 검사한다. 새로운 에이전트로부터 수신했을 경우, 핸드오프로 간주하여 등록 요청을 하고 그렇지 않은 경우 에이전트의 유효기간만 갱신한다.

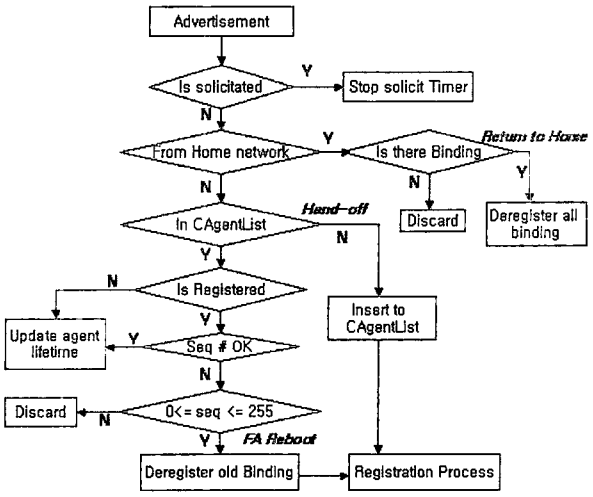

그림 4. Mobile Node의 핸드오프 결정 알고리즘

3.2.2 POSTECH MIP mobile node의 커널 스페이스

커널 스페이스에서는 유저 스페이스와 통신하며 핸드오프 시 ARP 패킷을 후킹(hooking)하고 에이전트 광고를 수신하여 유저 스페이스로 전달하는 작업을 수행한다.

◆ ARP hooking

mobile node가 외부 네트워크에 머물고 있을 때는 외부 에이전트로 향하는 ARP 응답 패킷을 제외한 ARP 패킷의 전송을 허용하지 않아야 한다. 외부 네트워크에서 ARP 후킹을 구현하기 위해 중계 드라이버의 코드 중 하위 레이어로 전달하는 함수에서 하위 레이어로 전송되는 모든 패킷의 헤더 부분을 검사하여 외부 에이전트로 향하는 ARP 응답 패킷이 아닌 ARP 패킷은 전송되지 않도록 구현하였다.

◆ 유저 스페이스로의 에이전트 광고 전달

커널 스페이스에서는 에이전트 광고를 수신할 경우 핸드오프 결정을 수행하는 유저 스페이스로 전달해야 한다. 이를 구현하기 위해 미니포트 드라이버로부터 패킷을 수신하는 함수에서 모든 패킷의 헤더를 분석하여, 수신한 패킷이 에이전트 광고인 경우 별도로 메모리를 할당하여 저장해 두었다가 유저 스페이스에서 질의시 전달하도록 구현하였다.

4. 실험

성능 평가를 위한 네트워크 환경은 그림 5와 같으며 에이전트들은 리눅스 커널 2.0.36에서 구현된 POSTECH MIP mobile 에이전트를 사용하였으며 각각의 에이전트는 access point와 직접 연결되어 base station의 역할을 한다. mobile node는 무선 랜카드를 가지고 있는 노트북을 사용하였으며 응용 프로그램은 웹 서비스, 실시간 라디오 방송, 실시간 MP3 음악 파일 재생, 3가지를 대상으로 하였다. 그림 6은 사용자가 POSTECH MIP mobile node를 동작시키고 환경을 설정하기 위한 GUI이며 그림 7은 자체적으로 개발한 분석 프로그램을 이용해 전송 중인 등록 요청 메시지를 분석한 것이다. 실험 결과, 핸드오프로 인한 지연(*hand-off latency*)이 중요하지 않은 웹 응용프로그램에서는 만족할 만한 성능을 보였으며, Mobile IP의 실행으로 인해 시스템에 미치는 영향은 무시할 정도로 적었다. 실시간 음악 파일 재생의 경우, 실시간 음악 혹은 비디오 같은 대부분의 응용 프로그램이 내부적으로 데이터를 버퍼링하여 재생하기 때문에, 핸드오프로 인한 서비스 중단 현상은 발생하지 않았다. 즉 핸드오프로 인한 지연 시간 동안 응용 프로그램의 내부 버퍼에 저장된 데이터가 계속 재생되어 사용자는 핸드오프를 인식하지 못하였다. 따라서 대부분의 멀티미디어 서비스가 내부적으로 버퍼링을 지원하므로 핸드오프로 인한 서비스 중단은 상당부분 해결할 수 있을 것으로 생각된다. 실시간 라디오 방송의 경우, 핸드오프 지연 시간동안 서비스가 잠시 중단되었으나, 핸드오프 완료 후 계속하여 정상적인 서비스를 받을 수 있었다.

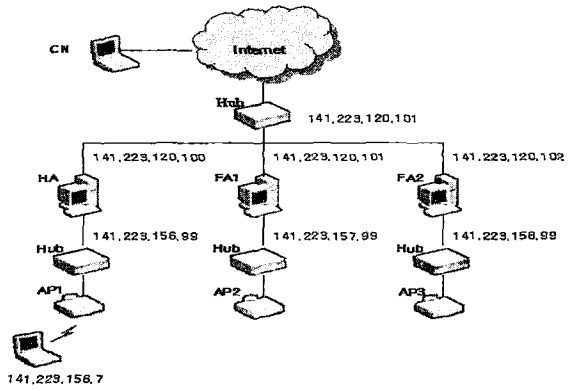

그림 5. 테스트 베드 구성도

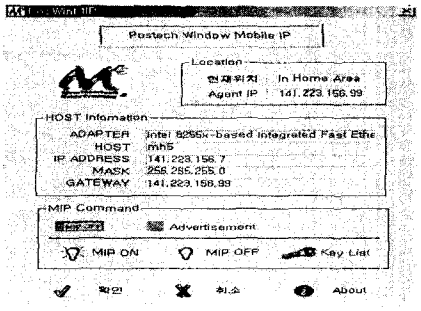

그림 6. mobile node의 사용을 위한 GUI

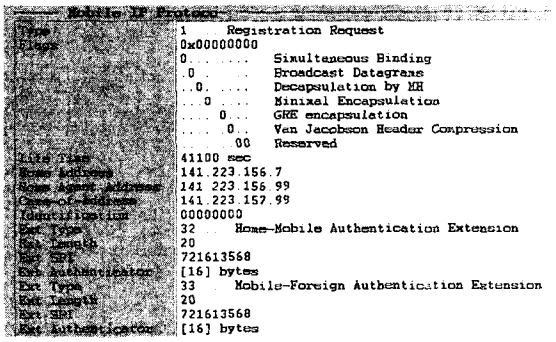

그림 7. 전송 중인 등록 요청 메시지의 분석

5. 결론

통신 기술이 발달함에 따라 컴퓨터 사용자들은 노트북, 랩탑(Lap Top), PDA (*Personal Digital Assistance*), HPC (Handheld PC) 등의 휴대 가능한 컴퓨터를 사용해서 장소에 구애받지 않고 원하는 정보를 얻기를 기대하게 되었으며 이동 컴퓨팅에 대한 수요가 급발전하여 이제는 이동 컴퓨팅이 곧 경쟁력으로 여겨지는 이동 컴퓨팅 시대로 접어들게 되었다. 본 연구실에서는 시대적 요구에 맞춰 이동 컴퓨팅을 지원할 수 있는 *mobile node*를 윈도우즈 2000 플랫폼에서 설계하여 POSTECH MIP mobile node를 개발하였다. POSTECH MIP mobile node는 Mobile IP 프로토콜을 분리하여 커널 스페이스와 유저 스페이스에서 설계함으로써 확장성과 효율성을 추구하였다. 또한 편리한 윈도우즈의 GUI를 통해 사용자가 쉽게 이동 서비스를 이용할 수 있도록 구현하였다. 실험 결과, 불가피한 최소한의 핸드오프 지연이 발생하기는 했지만 사용자에게 투명한 핸드오프를 제공할 수 있었다. 특히 대부분의 응용 프로그램이 버퍼링을 이용하여 핸드오프 지연 시간동안 서비스를 지원함으로써 핸드오프로 인한 서비스 중단 현상은 발생하지 않았다. 본 연구실에서는 현재까지 Mobile IP를 윈도우 2000 및 리눅스 환경에서 구현하였다. 차후에 이동성 관리 프로토콜들을 비교 분석하기 위해, Mobile IP 경로 최적화 (Route Optimization), 계층구조의 Mobile IP와 같은 프로토콜들을 구현 할 예정이다.

6. 참고 문헌

[1] C.Perkins, "Ipv4 Mobility Support", RFC 2002, October 1996
[2] A Project of MosquitoNet Mobile Computing Group http://mosquitonet.stanford.edu/mip
[3] Information on Mobile-IP implementation on Linux, National University of Singapore, http://mip.ee.nus.sg
[4] IETF Mobile IPv4 for 4.4BSD-based Unix systems, http://www.monarch.cs.cmu.edu/mobile_ipv4.html
[5] Mobile IPon Windows NT, http://mip-nt.aii.pub.ro
[6] Peter G.Viscarola, R&D Books, "Windows NT-Device Driver Development"
[7] Chris Cant, Macmillan Tecnnical Publishing, "Writing Windows WDM Device Drivers"
[8] Microsoft Windows 2000 Driver Development Kit
[9] Microsoft platform Software Development Kit
[10] Information on SUNY Binghamton Mobile-IP, http://anchor.cs.binghamton.edu/ ~mobileip
[11] Rivest, Ronald L, The MD5 Message-Digest Algorithm, RFC 1321, April 1992