主題

차세대 인터넷 IP 프로토콜 기술

숭실대학교 정보통신전자공학부 김영한, 박일균

차 례

I. 서 론

1990년대에 들어 인터넷의 규모가 급속도로 커지고 있다. 이러한 갑작스러운 성장은 초기 인터넷 설계자들의 예상을 빗나가는 것이었으며, 32비트 인터넷 주소 길이로 인한 주소 할당 공간의 부족이 가장 큰 이슈로 부상하였다. 이 문제를 해결하기 위해 IETF(Internet Engineering Task Force)의 IPng WG(working group)을 중심으로 차세대 인터넷 프로토콜의 표준화가 진행되어 새로운 인터넷 프로토콜 IPv6가 개발되었다. IPv6 및 관련 프로토콜의 표준화는 IPng WG에서 진행하고, 동시에 현재의 IPv4에서 IPv6로의 이전 및 활용 요건 조성 등은 ngtrans WG에서 담당하고 가상망인 6bone을 구축하여 실험하고 있다. 각 운영 체제 상에 IPv6 코드는 이미 구현되어 있으며, 상호 운용성도 검증되었고 단지 자연스로운 이전만을 기다리고 있는 상황이다.

본고에서는 이러한 IPv6 프로토콜 및 관련 주요 프로토콜들을 살펴보고 주소 체계 및 자동 설정 기능 등에 관해 알아본다. 또한 구현 현황과 이전 기법 및 6bone에 대한 현황을 살펴본다.

II. 차세대 인터넷 IP 프로토콜

1. 프로토콜 표준화 진행 현황

IPv6 프로토콜은 IPv6 이외에 많은 관련 프로토콜로 구성된다. 이들 프로토콜은 1995년도에 표준화가 진행된 이후에 1998년 말에 전체적인 프로토콜들에 대해 다시 한번 전체적인 개정이 이루어졌다. 따라서 현재 각 프로토콜의 RFC 번호가 새롭게 바뀌었거나 드래프트 상태에 있는 중이다. 표 1에서도 볼 수 있듯이, 대부분의 프로토콜들이 표준화 완료 상태에 있다.

문서 종류	제 목
RFC 2460	Internet Protocol, Version 6 (IPv6) Specification
RFC 2463	Internet Control Message Protocol(ICMPv6) for the Internet Protocol Version 6 (IPv6)
RFC 2461	Neighbor Discovery for IP Version 6 (IPv6)
RFC 2373	IP Version 6 Addressing Architecture
RFC 2450	Proposed TLA and NLA Assignment Rules
RFC 2375	IPv6 Multicast Address Assignments
RFC 2374	An IPv6 Aggregatable Global Unicast Address Format
RFC 2471	IPv6 Testing Address Allocation
RFC 2462	IPv6 Stateless Address Autoconfiguration
RFC 1886	DNS Extentions to support IP version 6
RFC 2080	RIPng for IPv6
RFC 2283	Multiprotocol Extensions for BGP-4
Draft	Basic Socket Interface Extensions for IPv6
RFC 2292	Advanced Sockets API for IPv6
Draft	Transition Mechanisms for IPv6 Hosts and Routers
Draft	Network Address Translation - Protocol Translation(NAT-PT)
Draft	Stateless IP/ICMP Translator(SIIT)
RFC 1981	Path MTU Discovery for IP version 6
RFC 2472	IP Version 6 over PPP
RFC 2470	Transmission of IPv6 Packets over Token Ring Networks
RFC 2467	Transmission of IPv6 Packets over FDDI Networks
RFC 2464	Transmission of IPv6 Packets over Ethernet Networks
RFC 2491	IPv6 over Non-Broadcast Multiple Access (NBMA) networks
RFC 2492	IPv6 over ATM Networks
RFC 2497	Transmission of IPv6 Packets over ARCnet Networks
Draft	Transmission of IPv6 Packets over Frame Relay Networks Specification
RFC 2465	Management Information Base for IP Version 6: Textual Conventions and General Group
RFC 2466	Management Information Base for IP Version 6: ICMPv6 Group
RFC 2454	IP Version 6 Management Information Base for the User Datagram Protocol
RFC 2452	IP Version 6 Management Information Base for the Transmission Control Protocol

표 1. 표준화 현황

2. Internet Protocol version 6 (IPv6)[1]

그림 1에 IPv4 및 IPv6의 헤더 구조를 나타냈다. IPv4 헤더에서 IPv6 헤더로 넘어갈 때 기능이 없어진 영역은 음영으로 표시하였다. 128비트 주소 영역 때문에 헤더의 길이는 두배로 확장되었지만 영역의 수는 12개에서 8개로 감소하여 처리 과정을 단순화한다. IPv6 헤더에서 version은 인터넷 프로토콜의 버전을 말하며, 여기서는 6의 값을 가진다. traffic class는 트래픽 클래스를 명시해주는 영역이며, 현재 diffserv WG에서 이 영역의 활용을 고려하고 있다. flow label은 해당 IPv6 패킷이 속하는 플로우에 대한 특성을 나타내주며 이 영역에 대한 사용은 표준화되어 있지 않다. payload length 영역은 IPv6 헤더 다음에 붙는 데이터들의 길이를 표시하며, next header 영역은 IPv6 헤더 다음의 헤더의 종류를 표시한다. hop limit영역은 IPv4에서의 time to live(TTL) 영역과 같은 역할을 하며 거쳐갈 수 있는 라우터의 최대 수를 명시한다. 그 후로는 각각 128비트의 소스 주소와 목적지 주소가 위치한다. 이 헤더 구조 상의 차이와 같이 나타나는 IPv6의 특징은 다음과 같이 요약할 수 있다.

- 주소 공간 확장
- 간략화된 헤더 포맷
- 확장 헤더의 이용
- 플로우 레이블을 이용한 QoS 지원
- 보안용 확장 헤더를 통한 IPv6 계층에서의 보안 기능 지원

IPv6는 기존 IPv4에서의 32비트 주소 길이를 128비트로 확장함으로써 현재의 주소 부족 문제를 해소하며, 다양한 주소 할당 방식을 취할 수 있게 된다. 또한 IPv4의 헤더에서 조각화/재조립 등 잘 사용되지 않는 영역들은 삭제되거나 확장헤더로 옮겨져서 필요할 때만 사용되게끔 하였다.

확장 헤더는 패킷 조각화/재조립 또는 소스 라우팅과 같이 IPv4에서 잘 이용되지 않았던 영역이나 옵션들을 위한 것으로 필요할 때에만 사용하게끔 함으로써 기본 헤더를 간략하게 하여 일반 처리 과정의 효율을 높이게 하였다. 현재 IPv6의 확장 헤더로는 다음이 정의되어 있다.

- hop-by-hop options header
- destination options header
- routing header

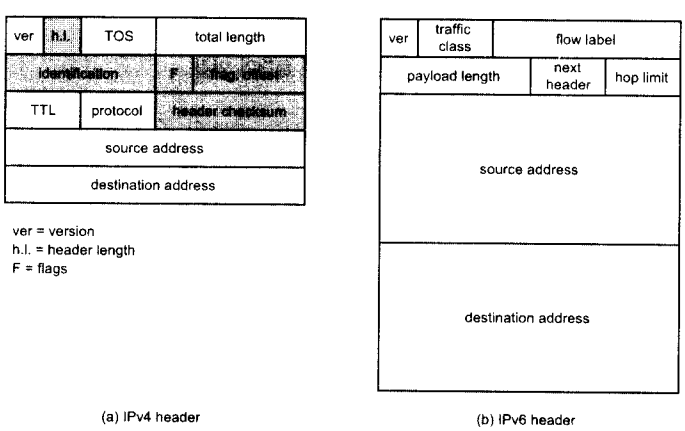

그림 1. IPv4 및 IPv6 헤더 구조

- fragmentation header
- authentication header
- encapsulating security payload(ESP) header

옵션 헤더는 지나가는 홉(hop)마다 처리할 옵션을 포함하는 hop-by-hop 옵션 헤더와 IPv6 헤더의 목적지 주소에 명시된 노드에서만 처리하는 옵션을 포함하는 목적지 옵션 헤더로 구성된다. Routing 헤더는 소스 라우팅을 위한 헤더로서 여러가지 타입을 가질 수 있지만 현재는 중간 라우터 주소들의 리스트만 포함하는 제일 간단한 구성의 타입 0만 선언되어 있다. 조각화 헤더는 조각화/재조립을 위한 정보를 포함하는 헤더이며, 인증 헤더 및 캡슐화 헤더는 보안을 위한 헤더이다.

IPv6의 각 헤더들은 바로 다음에 어떤 종류의 헤더가 있는지 명시해주는 next header 영역을 가지고 있다. 만약 IPv6 헤더의 next header 값이 0이면 다음 헤더가 hop-by-hop 헤더임을 알려준다.

로 사용할 생각이었으나 구체적인 사용법에 대해서 표준화는 아직 진행되어 있지 않고 다른 QoS 관련 WG의 활동 결과가 추후 반영될 것으로 보인다. QoS 지원을 받지 않는 플로우일 때는 해당 플로우 레이블을 0으로 채운다. 만약 QoS 기능을 지원하지 않는 호스트나 라우터에서는 패킷 생성 시 이 영역을 0으로 채워서 내보내고 값을 바꾸지 않고 전달하며 수신지에서는 무시한다.

마지막으로는 IPv6에서의 확장 헤더 중에 인증 헤더와 캡슐화 헤더 등이 있어서 네트워크 계층 단위의 보안 기능을 기본으로 지원하도록 하였다. 이 헤더 내에 포함되는 인증 기법 및 암호화 기법은 ipsec WG에서 표준화를 진행하고 있다.

3. IPv6 주소 체계 [4]

IPv6에서 할당할 수 있는 주소 공간이 넓어짐에 따라 다양한 주소 할당 방식을 사용할 수 있으며, 여러 종류의 주소들을 포함할 수 있게 되었다. IPv6 주

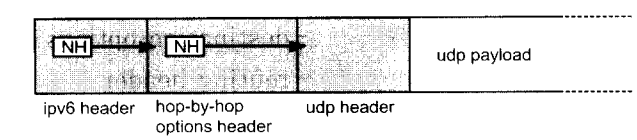

그림 2. Next hedaer 영역을 이용한 헤더의 연결

그림 2는 next header 영역을 이용하여 ipv6 헤더 hop-by-hop 옵션 헤더 udp 헤더의 연결을 표시해주고 있는 예이다. 각 확장 헤더들은 자신을 나타내는 next header 값을 가지고 있으며, UDP나 TCP 등의 상위 계층 프로토콜들에 대한 NH 값은 IPv4에서 protocol 영역에서 사용되던 값들을 사용한다.

IPv4에서는 없었던 플로우 레이블 영역은 길이가 20비트이며, QoS 지원을 위해 IP 패킷의 연속적인 흐름을 플로우로 정의하고, 이를 식별하기 위한 값으

소에서는 주소의 종류 및 서브넷을 판별할 때 사용하는 prefix와, 네트워크에 연결되어 있는 각 인터페이스들을 구별해 주는 interface ID로 구성된다. IPv6 망에서 주소를 할당하는 기본 단위는 호스트나 라우터가 아닌 인터페이스 중심이다. 즉 같은 호스트에 동일 링크와 연결되는 여러 개의 인터페이스들이 있을 경우 각 인터페이스마다 다른 주소를 할당받게 된다. 그러나 보통 한 호스트에 하나의 인터페이스로 링크에 연결되므로, interface ID로 호스트를 구분하게 된다. 또한 기존의 IPv4 주소 체계에서의

unicast 및 multicast 주소와 다른 새로운 anycast라는 주소의 개념을 도입하였다. 한 라우터가 anycast 주소를 목적지 주소로 가지는 패킷을 수신했을 경우, 라우팅 프로토콜에 따라 같은 anycast 주소를 가지는 여러 노드들 중 가장 가까운 노드로만 패킷을 보내는 방식으로 이용한다.

IPv6 주소의 표기는 8개의 16진수 4자리 숫자를 콜론(:)으로 구분하여 표기한다.

3ffe:2e01:1:0:0:60:9791:b839
3ffe:2e01:1::60:9791:b839

주소의 길이가 길기 때문에, 0이 연속되어 있는 구간에 대해서는 한번에 한하여 두개의 콜론(::)으로 표시할 수 있게 된다. 위의 주소 예에서, 두 개의 주소는 같은 것이다. 위의 예에서는 어느 부분까지가 prefix인지 판별할 수 없기 때문에 prefix에 대한 정보를 표시하고자 할 때에는 주소를 써주고 "/" 다음에 prefix의 길이를 명시해준다. 만약 위의 주소에서 prefix의 길이가 48비트일 경우

3ffe:2e01:1::/48

로 표시한다.

표 2에 IPv6의 각 주소 영역과 이를 식별하는 prefix 값을 나타냈다.

0000 0000 prefix 영역은 특수 목적 주소 영역으로 ::1 은 loopback 주소를 나타내며 또한 IPv4 주소를 그대로 IPv6 주소로 이용하고자 할 때 사용하는 IPv4-compatible (예: ::203.253.3.254) 주소나, IPv6 스택을 갖추지 못한 호스트의 IPv4 주소를 표시할 때 사용하는 IPv4-mapped(예: ::ffff:203.253.3.254) 주소 등이 이 부분에 속한다. 두번째와 세번째의 예약 영역은 TCP/IP 스택이 아닌 ISO 및 IPX 계열의 프로토콜들을 위해 예약된 곳이다. Aggregatable global unicast address영역은 일반적인 IPv6 주소 영역이다. 이 영역에서의 주소의 할당은 계층 구조를 가지게 된다. Link-local 주소 및 site-local 주소는 모두 해당 영역 안에서만 의미를 가지는 주소이다. 이 때 interface ID로 사용되는 값들은 link나 site에서 고유한 값임을 보장받아야 한다. 따라서 링크 계층 주소들이 이들 prefix 뒤에 사용될 수 있다. 마지막 영역은 멀티캐스트 주소를 위한 prefix이다. 나머지 표시하지 않은 영역들은 아직 할당되지 않은 부분들이다.

그림 3은 표 2의 주소 영역 중 aggretable global unicast 주소의 계층적인 구조를 보여주고 있다. 여기서 TLA ID는 IANA(Internet Assigned Numbers Authority) 기관에서 정의한다. TLA ID를 할당받은 기관은 하부 기관에

Address allocation	Prefix
Reserved	0000 0000
Reserved for NSAP	0000 001
Reserved for IPX	0000 010
Aggregatable global unicast address	001
Link-local unicast address	1111 1110 10
Site-local unicast address	1111 1110 11
Multicast address	1111 1111

표 2. IPv6 주소 할당

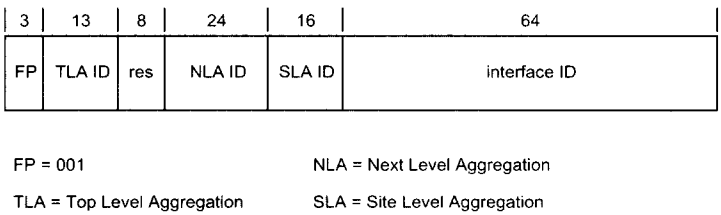

그림 3. Aggregatable global unicast address allocation

NLA ID를 주게 되고, 하부 기관은 그 밑의 하위 서브넷에 SLA ID를 할당해주게 된다. 현재 6bone에서의 테스트용 주소 할당은 위의 방법을 토대로 하고 있다. 현재 한국에서는 6bone으로부터 3ffe:2e00::/24의 TLA 주소를 할당받았다. 그 다음부터는 NLA 영역을 NLA1, NLA2로 나누어 할당받아 사용하고 있다.

4. Internet Control Message Protocol version 6 (ICMPv6) [2]

ICMPv6는 기존의 ICMP 헤더 중 에러 메시지에 IPv6 헤더를 포함할 수 있도록 수정되어 있으며, 그밖에 메시지 타입의 종류가 조정되었다. ICMPv6에서의 메시지 종류는 표 3과 같다.

위와 같이 IPv4에서의 source quench 메시지 및 timestamp, information, address mask에 대한 요구/응답 메시지가 삭제되었으며, redirect 메시지와 router advertisement/solicitation 메시지는 NDP로 옮겼다. packet too big 메시지가 새로 추가되었으며 이는 IPv4와는 달리 중간 라우터에서 MTU 감소로 인하여 전달 중이던 패킷을 조각화 할 수 없기 때문이다. 그 밖에 메시지가 그대로 존재하여도 해당 메시지에 대한 내부 코드는 많이 수정되어있다.

5. Neighbor Discovery Protocol(NDP) [3]

NDP는 기존 IPv4에서 ICMP의 router discovery, redirection 기능 및 ARP의 link-layer address resolution 기능을 포함하며 다음의 다섯 가지의 메시지로 구분된다.

- router solicitation/advertisement 메시지

router solicitation 메시지는 일반 호스트나 라우터가 동일 링크 상의 라우터에 대한 정보를 알고 싶을 때 보내진다. router advertisement 메시지는 각 라우터가 자신에 대한 정보를 주기적으로 브로드캐스팅할 때 사용하는 메시지이지만 router solici-

ICMPv6 에러 메시지	Destination Unreachable
	Packet Too Big
	Time Exceeded
	Parameter Problem
ICMPv6 정보 메시지	Echo Request
	Echo Reply

표 3. ICMPv6 메시지 종류

tation 메시지를 수신했을 때는 주기가 되지 않았어도 바로 보낸다.

- neighbor solicitation/advertisement 메시지

neighbor solicitation 메시지는 동일 링크 상에 있는 다른 호스트의 링크 주소를 알기 원할 때 보내지는 것이며, neighbor advertisement 메시지는 solicitation 메시지에 응답하여 보내어지는 메시지이다.

- redirection 메시지

패킷을 보낸 호스트에게 해당 목적지로 보내는데 더 좋은 경로를 가지는 라우터가 있음을 알려주는 메시지이다. 이를 수신한 호스트는 라우팅 테이블을 수정하여 그 다음부터는 패킷을 수정된 라우터로 보내게 된다.

NDP 메시지들은 해당 NDP 헤더 다음에 하나 이상의 추가 정보가 연속될 수 있는데 이들 정보에는 source link-layer 주소, target link-layer 주소, redirected 헤더, prefix 정보, MTU 값 등이 있다.

이상의 메시지를 이용한 NDP의 주요 기능들은 다음과 같다.

- Router 및 prefix discovery

동일 링크 상에 붙어있는 라우터를 찾아내며, 동시에 라우터로부터 prefix에 대한 정보를 얻어온다. 호스트의 라우터 인식은 라우터가 보낸 router advertisement 메시지를 수신함으로서 가능하다. 라우터는 이 메시지를 두 가지 방식으로 보낸다. 즉 주기적으로 계속 router advertisement 메시지를 보내거나 또는 router solicitation 메시지를 수신했을 때 응답으로서 바로 메시지를 보낼 수 있다. Router advertisement 메시지를 보낼 때 NDP 헤더 다음의 옵션 영역을 통해 router의 링크 계층 주소, 해당 링크의 MTU 값, prefix 정보 등이 전달될 수 있다. 이 메시지를 수신함으로써 해당 호스트는 router 및 prefix 정보를 취할 수 있다.

-address resolution 및 neighbor unreachability 탐지

동일 링크 상에 연결되어 있는 호스트나 라우터의 링크 계층 주소를 얻어낸다. Neighbor solicitation/advertisement 메시지를 이용하여 ARP에서처럼 해당 IPv6 주소에 대한 링크 계층 주소를 구하며, 이에 덧붙여 같은 IPv6 주소의 중복 여부를 판별하거나 최종 목적지가 동일 링크 상에 있는지 여부를 판별하게 된다.

-redirection 기능

라우터가 해당 목적지에 대한 보다 나은 경로 상의 첫번째 라우터를 알고 있을 경우 라우팅 정보를 수정하도록 해당 호스트에게로 redirect 메시지를 보낸다.

6. 자동 설정(auto configuration) 기능 [5]

자동 설정 기능은 사용자의 수동 설정 없이도 호스트 및 라우터에서 자체로 주소를 설정하는 일종의 플러그 앤 플레이(plug-and-play) 기능이다. 자동 설정 동작에서 주소 설정 시 인터페이스 ID는 링크 계층 주소를 이용한다. 동일 링크 상에 라우터가 없을 때에는 fe80::〈링크 계층 주소〉의 형태로 link-local 주소가 설정되어 최소한 동일 링크 상에서의 통신이 가능하게끔 한다. 라우터가 존재할 때에는 router solicitation 메시지에 대한 응답으로 router advertisement 메시지를 수신하게 되며, 이로써 디폴트 라우터의 설정이 link-local 주소 설정 위에 추가될 수 있다. 또한 이 메시지에는 자동 설정을 하려는 호스트가 사용할 수 있는 prefix 정보가

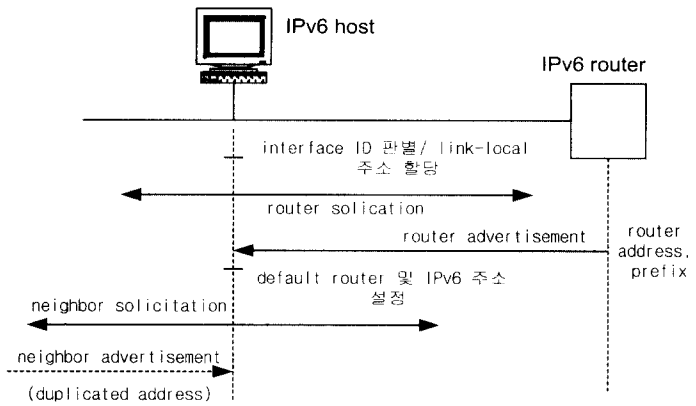

그림 4. 자동 설정 과정

포함되어 있다. 호스트는 이 prefix에 자신의 링크 계층 주소를 결합하여 IPv6 주소가 추가될 수 있다.

또한 자신과 중복된 주소가 사용되는지에 대한 여부를 알기 위해 neighbor solicitation 메시지를 보내게 된다. 이 때 응답이 없을 경우 설정된 주소를 사용하게 된다. 그림 4는 자동 설정 기능을 사용했을 때의 동작 절차를 보여준다.

III. IPv6로의 이전

1. IPv4/IPv6 이전 기술[7]

현재의 인터넷의 많은 IPv4 노드들을 한꺼번에 IPv6로 대치하는 것은 사실상 불가능하며, 따라서 IPv4와 IPv6를 같이 사용하는 방법을 모색하여 점

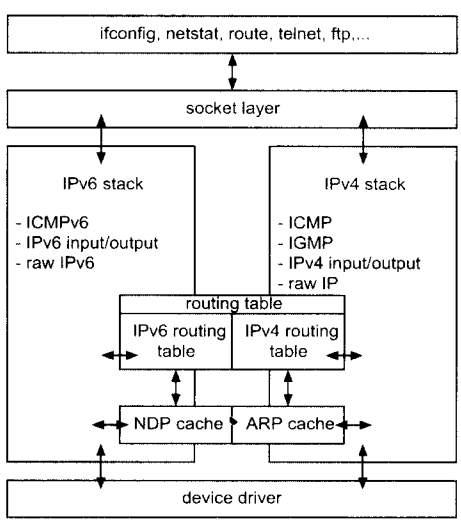

그림 5. IPv4/IPv6 이중 구조

진적으로 IPv6 인터넷 환경으로 넘어가는 것이 IPv6로의 이전의 최선책이다. 이를 위해 필요한 사항은 다음과 같다.

- IPv6 스택을 갖춘 호스트 및 라우터들은 IPv4 스택도 함께 갖추게끔 하여 기존 IPv4 인터넷과의 통신을 가능하게 하고, IPv4 인터넷 상에서의 IPv6를 통한 통신에서 보조 역할을 담당하게 한다. 그림 5는 IPv4/IPv6 이중(dual) 스택 구조를 보여준다.
- 각 호스트 및 라우터들은 IPv4 주소가 IPv6 주소의 일부로 포함되는 IPv4-compatible IPv6 주소를 가진다. IPv4/IPv6 이중 구조를 취하는 노드에서는 IPv4의 주소를 사용할 수 있다.
- IPv4 망 건너편에 있는 IPv6 노드와의 통신을 위한 터널링 기능을 제공한다. 터널링 기능은 상대방의 IPv6 사이트에 도달할 때까지 IPv6 패킷을 IPv4 헤더에 씌워 IPv4 패킷처럼 보내는 기능을 말한다.
- IPv4 전용 노드와 IPv6 전용 노드 간의 통신을 위해서 IPv4와 IPv6 간의 변환 라우터 등을 이용한다.

본 장에서는 이들 기술을 살펴본다.

2. IPv4/IPv6 터널링[7]

터널링 기능은 IPv6 패킷을 IPv6로 통신이 가능한 두 지점의 끝점을 IPv4 헤더로 캡슐화하여 실제로 IPv4 인터넷 환경 상에서 IPv6 패킷의 전달을 가능하게 하는 역할을 담당한다. 터널링은 설정 터널링(configured tunneling)과 자동 터널링(automatic tunneling)으로 구분된다. 설정 터널링은 미리 설정된 터널링의 끝점의 IPv4 주소를 IPv4 헤더의 목적지 주소로 사용하여 보내는 방식이다. 자동 터널은 IPv6 주소 내에 포함된 IPv4 주소를 IPv4 헤더의 목적지 주소에 이용하여 다른 설정 없이 자동으로 터널을 생성하여 패킷을 전송하는 방식이다. 여기에 사용될 수 있는 주소는 IPv4-compatible IPv6 주소가 해당된다.

터널링을 끝점에 해당하는 노드에 따라 네가지로 분류된다. 라우터에서 라우터로 터널링을 하거나, 호스트에서 라우터로 터널링을 할 때에는 해당 라우터들이 IPv6 헤더의 목적지 주소에 명시된 최종 목적지가 아닌 경우가 많다. 이 경우에는 터널링의 양 끝점의 주소를 미리 알고 있어야 하며, 따라서 설정 터널링을 사용한다. 반대로 라우터에서 호스트로 터널링을 하거나, 호스트에서 상대방 호스트로 터널링을 할 때에는 끝점에 해당하는 호스트가 Ipv6 패킷의 최종 목적지인 경우이며, 이 때에는 IPv6 목적지 주소로 터널링용 주소를 설정할 수 있으므로 자동 터널링을 사용한다.

IPv6 패킷이 터널링을 통해 전송될 때 터널 양 끝점간의 MTU 값은 실제 링크 MTU 값에서 IPv4 헤더의 크기 20바이트를 뺀 값이 된다. 또한, IPv4 헤더의 DF(don't fragment) 플랙을 1로 셋팅하여 링크 MTU 값이 감소로 인해 조각화 기능이 불가피하게 되면 패킷을 삭제하도록 하고 "packet too big" ICMPv6 에러 메시지를 전송하게 한다.

3. IPv4/IPv6 변환

IPv6 이전 기술 외에 현재의 IPv4 환경의 인터넷에서 IPv6를 이용한 통신을 하기 위한 방법 중 하나가 IPv6/IPv4 변환(translation)이다. IPv6 이전 기술은 IPv4-IPv6 스택을 모두 가진 호스트들끼리 통신을 하는 방법에 관한 것이며, 터널을 통한 통신 과정이 숨겨지지 않고 그대로 드러나게 된다. 이와 반대로 IPv4/IPv6 변환은 IPv4 전용 호스트와 IPv6 전용 호스트 간의 통신을 위한 기술이며, 주소 및 헤더의 변환 과정이 감추어져 있게 된다. 그림 6은 IPv4/IPv6 변환을 위한 네트워크 상의 요소들을 보여준다.

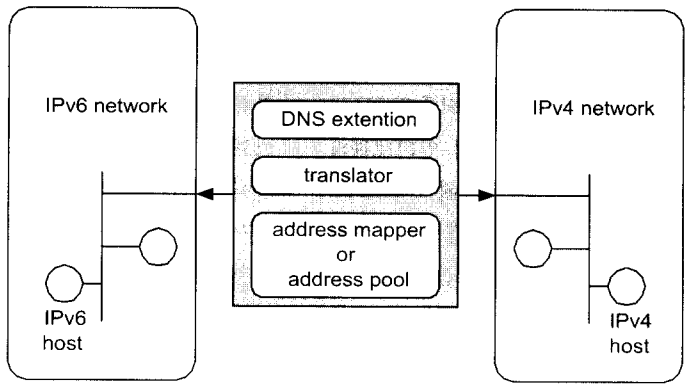

그림 6. IPv4/IPv6 변환을 위한 구성 요소 [10]

위 구성 요소 중 translator는 실제 IPv4 패킷과 IPv6 패킷의 변환을 담당하는 부분이다. DNS extention 요소에서는 AAAA 레코드 및 A 레코드를 변환해주며 각 변환된 레코드에 맞게 IPv4 또는 IPv6 주소를 반환한다. 또한 address mapper는 IPv4 및 IPv6 주소의 맵핑을 담당하며 맵핑 시 address pool에서 주소를 하나 가져온다.

IPv4/IPv6 변환에는 두가지 방식이 존재한다. 첫번째는 헤더 변환 방식으로, 현재 ngtrans WG에서는 NAT-PT(network address translation-protocol translation) 및 SIIT(stateless IP/ICMP translation)를 중심으로 표준화가 진행 중에 있다. 다른 하나는 프록시 서버(proxy server) 방식으로, SOCKS를 중심으로 하여 구현 작업이 진행 중에 있다.

그림 6의 모델과 비교했을 때 NAT-PT/SIIT에서는 NAT가 address mapper로서 동작하며, PT는 SIIT를 보완하는 형식을 취하여 실제적인 헤더 변환 작업을 수행하는 translator가 된다. DNS이나 FTP 등 응용 계층 차원에서 주소를 다루는 경우에는 ALG(application level gateway)를 사용하게 된다. SOCKS에서는 SOCKS 서버가 실질적인 translator로서 동작하며, address mapper는 따로 가지고 있게 된다.

NAT-PT에서는 먼저 DNS ALG에서 AAAA 레코드와 A 레코드의 변환 및 v4-DNS와 v6-DNS 간의 주소 정보 교환을 가능하게 한다. 그 다음에 IPv6 호스트에서 IPv4 호스트로 패킷을 전송할 경우에는 소스 주소를 IPv4 호스트에서 목적지 주소로 이용할 수 있게끔 NAT-PT로부터 할당받고 목적지 주소를 v4 주소 앞에 NAT-PT가 인식할 수 있는 prefix를 삽입하여 IPv6 패킷으로 전송한다. NAT-PT에서는 목적지 주소로부터 IPv4 주소를 인식하고, 패킷 헤더를 IPv4로 변환하여 보내게 된다. 그 반대 방향에 대해서는 IPv4 패킷의 목적지 주소에는 DNS에 의해 NAT-PT로부터 할당받은 IPv4주소를 사용하게 된다[9].

SOCKS는 원래 방화벽 용의 서버로 구현된 것이다. 여기에 IPv4/IPv6 변환을 위한 확장 기능을 추가함으로써 같은 기능을 제공할 수 있다. 대신 SOCKS에서는 호스트 API의 수정을 필요로 하며 중간의 SOCKS 서버가 변환을 담당한다. 버전이 다른 호스트끼리 통신을 원할 경우에서는 호스트의 응용 계층과 API 계층 사이의 SOCKS API 클라이언트가 서버에 접속하여 변환 기능을 수행할 수 있게끔 해야 한다[11].

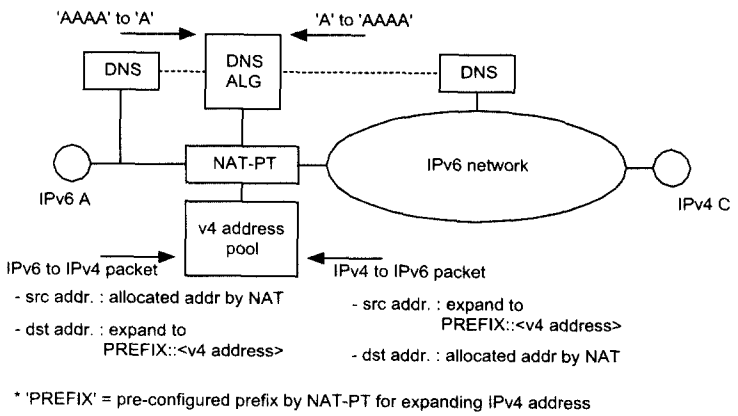

그림 7. NAT-PT 동작 [9]

IV. IPv6 구현 현황

90년대 중반부터 IPv6의 표준화가 이루어지면서, 여러 가지 다양한 플랫폼에서 IPv6 스택을 구현하는 작업이 이루어져왔다. 현재도 IPv6 표준의 개정이 계속되고 있으며, 이에 따라 구현 코드도 수정을 같이 하고 있다.

IPv6 구현 코드는 크게 호스트 부분과 라우터 부분으로 나누어 볼 수 있다. 호스트 부분으로는 각 운영체제 별로 다양한 구현 코드가 존재한다. linux나 FreeBSD, NetBSD, BSDI, BSD4.4/lite 등의 BSD 계열처럼 코드가 개방되어 있는 운영체제로부터 선 시스템, SCO, AIX(IBM), apple , SGI 등에서도 테스트용 구현 코드를 내놓거나 구현 중에 있다. 최근에는 마이크로소프트 사에서도 자사의 win NT 기반의 테스트 구현 코드를 내놓았다[13].

라우터 부분으로는 3COM, CISCO, BAY Network 등 라우터 장비 개발 회사에서 구현 코드의 개발이 진행 중에 있다. MERIT에서는 gated 및 MRT 소프트웨어를 통해 RIPng 및 BGP4+ 등 IPv6 주소를 이용한 라우팅 프로토콜을 구현하고 있다. Gated 및 MRT는 특히 BSD 계열의 운영체제 상에서 운용되고 있어 RIPng까지가 한계인 호스트 부분의 다른 구현 코드들에 비해 강점을 가지고 있다[13][15].

이 중 BSD 계열에서 NetBSD 및 FreeBSD를 기반으로 국내를 포함하여 여러 곳에서 IPv6 스택 구현 코드의 개발이 진행되고 있다. 프랑스의 INRIA Rocquencourt 사는 FreeBSD 기반의 IPv6 코드 개발을 지속적으로 진행해 나가고 있는 곳 중의 하나이다. 현재 FreeBSD 3.0 기반의 코드에까지 구현이 진행되었며, 응용프로그램은 웹 서버 및 브라우저까지 진척된 상태이다. 일본의 WIDE 프로젝트에서 개발 중이던 코드를 기반으로 지속적인 코드 구현을 하고 있는 KAME 역시 FreeBSD 3.0까지 구현되어 있다. 웹 서버 및 브라우저 외에도 IPv6 용 SMTP/POP가 구현되어 있으며, 앞으로의 QoS 지원을 고려하여 ALTQ가 결합되어 있다. 국내에서는 숭실대와 ETRI가 공동으로 DCN 코드 1차 버전이 구현되어 있으며 현재 FreeBSD 2.2.7까지 버전 업이 이루어진 상태이다. DCN 코드는 기본적인 응용프로그램이 구현되어 있으며 6bone과의 상호 연동성 시험을 성공적으로 수행하였다. 표 4는 FreeBSD를 기반으로 한 각 구현 코드들을 비교한 것이다. 이 중 IPv6 core는 IPv6, ICMPv6, NDP 뿐만 아니라 autoconfiguration, trasi-

	DCN	INRIA	KAME
IPv6 core	Yes	Yes	Yes
Routing Protocol	static/RIPng/ BGP4+ (테스트 중)	static/RIPng/ BGP4+	static/RIPng/ BGP4+
Advanced Protocol (www client / server)	99년 1/4분기	apache/mmm	apache/mozilla
Security	Porting 중	Yes	Yes

표 4. FreeBSD에 기반한 각 구현 코드들 간의 비교 [19]

tion mechanism 구현 상태, 'AAAA' 레코드에 의한 호스트 주소 찾기 및 ftp, telnet 등의 기본 응용 프로그램을 포함한다[15][16].

현재 DCN 코드는 BGP4+ 테스트 중에 있으며, 99년도 상반기 내에 웹 서버/클라이언트 등 좀더 진보된 응용 프로그램 포팅될 예정이다. 또한 RSVP 및 Diffserv 등을 IPv6에 포함시킴으로써 QoS에 대한 지원을 할 예정에 있다.

V. 6bone 현황

1. 6bone

6bone은 'IPv6 backbone'의 약자로서, IPv6

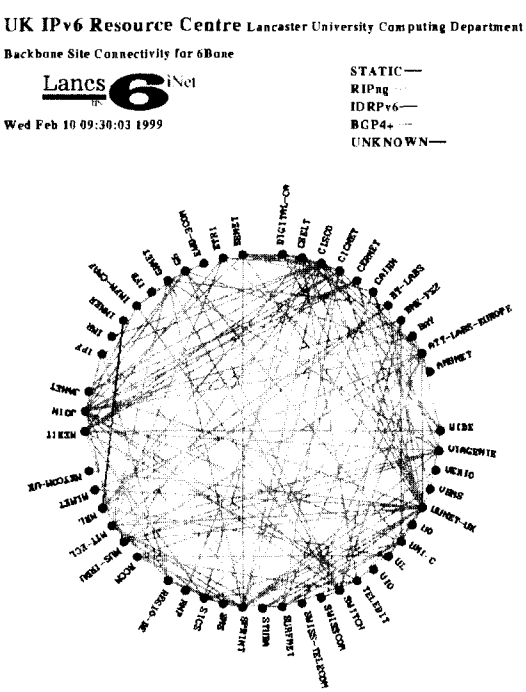

그림 8. 6bone 백본 망 연결도 [17]

NLA 종류	사이트 명	할당 prefix
NLA1 (3ffe:2e0x::/32)	6bone KR	3ffe:2e01::/32
	미리내	3ffe:2e02::/32
	event 용	3ffe:2e03::/32
NLA2 (3ffe:2e01:yyyy::/48)	ETRI	3ffe:2e01:1::/48
	SSU	3ffe:2e01:2::/48
	KAIST	3ffe:2e01:3::/48
	HYU	3ffe:2e01:4::/48
	INET	3ffe:2e01:5::/48(reserved)
	KT	3ffe:2e01:6::/48
	ICU	3ffe:2e01:7::/48
	DGU	3ffe:2e01:8::/48

표 5. 6bone KR prefix 할당 현황 [18]

호스트와라우터로 구성된 가상망을 말한다. 현재 인터넷은 거의 대부분이 IPv4 환경으로 구성되어 있기 때문에, 한꺼번에 IPv6 환경으로 전환하는 것도 힘들 뿐만 아니라 구현된 코드들의 테스트에도 여러 가지 장애가 된다. IPv6 스택을 구현하는 곳에서 실험실 수준으로 망을 구성하여 테스트를 수행할 수 있겠지만 거대한 인터넷에서의 다양한 네트워크 환경들을 모두 테스트하기에는 역부족이다. 따라서 여러 군데에 흩어져 있는 IPv6 구축 환경들을 터널을 통하여 연결함으로써 현재 구현된 코드들에 대해 다양한 테스트를 수행할 수 있으며, 이외에도 IPv6에 대한 마인드를 확산시키는 역할도 하게 된다.

각 사이트 간의 연결은 static 라우팅 정보의 설정에 따른 연결, RIPng에 의한 연결, 그리고 BGP4+에 의한 연결이 가능하다. 그러나 6bone에서의 백본 라우터가 되기 위해서는 BGP4+가 동작이 가능해야 하고, 현재 대부분의 연결은 BGP4+로 되어 있다. 현재 55개 IPv6 사이트가 pTLA prefix를 할당받았으며, 총 29개국 300개 이상의 사이트가 6bone에 연결되어 있다. 그림 8은 영국 Lancaster 대에서 계속 갱신하는 백본 망 구성도로, 현재 6bone에서의 pTLA를 사용하는 백본 망의 연결 상태를 보여준다.

2. 6bone KR

6bone KR은 국내의 6bone 활동을 위한 모임으로써 98년 초 ETRI에서 테스트용 pTLA prefix를 6bone으로부터 할당 받음으로써 본격적인 활동이 시작되었다. 국내 주소 할당을 담당하고 있으며 NLA 부분을 NLA1, NLA2로 나누어 NLA1은 국가 규모의 프로젝트에 할당하고, NLA2는 각 학교나 단체에 할당해주게 된다. 현재 약 8 개의 NLA2 주소가 할당되어 있다.(표 5) 앞으로 다양한 망과 연동할 예정에 있다.

그림 9는 현재 6bone KR의 연결 상태를 보여준다.

VI. 결 론

현재 IPv4 환경의 인터넷에서는 부족한 주소 공간으로 인한 문제를 안고 있다. 이를 해결하기 위해서 주소 공간을 대폭 확장한 IPv6가 표준화되었으며, 주소 공간 확장과 더불어 효율적인 IPv6 헤더 처

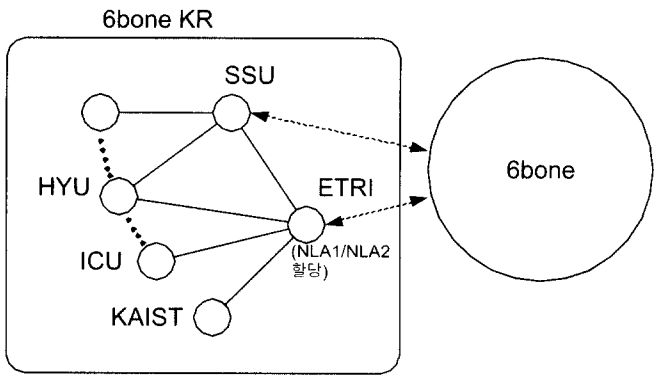

그림 9. 6bone KR 구성도

리를 위한 수정이 가해졌다. 현재 전세계적으로 6bone이라는 가상망을 구성하여 이 새로운 프로토콜의 구현 및 테스트를 통해 앞으로의 실질적인 IPv6 운용을 위한 기술을 쌓고 있다. 국내에서도 6bone으로부터 pTLA 주소를 할당 받아 6bone KR이라는 가상망을 구성하였다. 주소공간 확충이라는 문제로 인하여 IPv6로의 이전은 조만간 이루어질 것이며, 따라서 차세대 IP 계층 기술에 대한 더 많은 관심과 연구가 필요하다.

※ 참 고 문 헌

[1] RFC 2460, "Internet Protocol, Version 6 (IPv6) Specification"

[2] RFC 2463, "Internet Control Message Protocol(ICMPv6) for the Internet Protocol Version 6 (IPv6)"

[3] RFC 2461, "Neighbor Discovery for IP Version 6 (IPv6)"

[4] RFC 2373, "IP Version 6 Addressing Architecture"

[5] RFC 2462, "IPv6 Stateless Address Autoconfiguration"

[6] Internet draft, "DNS Extentions to support IP version 6," draft-ietf-ipngwg- dns-lookups-03.txt

[7] Internet draft, "Transition Mechanisms for IPv6 Hosts and Routers," draft-ietf-ngtrans-mech-01.txt

[8] Internet draft, "Stateless IP/ICMP Translator(SIIT)," draft-ietf-ngtrans-siit-05.txt

[9] Internet draft, "Network Address Translation-Protocol Translation(NAT-PT)," draft-ietf-ngtrans-natpt- 04.txt

[10] Internet draft, "A Communication Mechanism between IPv4 and IPv6," draft-tsuchiya-ipv4-ipv6-translator-00.txt

[11] Internet draft, "A SOCKS-based IPv6/IPv4 Translator Architecture," draft-kitamura-socks-ipv6-trans-arch-00.txt

[12] "Deployment and Experiences of WIDE 6bone, "http://www.v6.wide.

ad.jp/Papers/yamamoto/
[13] "SOCKS Proxy Protocol," http:// www.socks.nec.com/
[14] "IPng Implementations," http:// playground.sun.com/pub/ipng/ html/ipng-implementations.html
[15] "gated," http://www.gated.edu
[16] "KAME Project," http://www.kame.net
[17] "6bone Backbone Diagram," http:// www.6bone.net/6bone-backbone.html
[18] 신명기, "6bone KR status report," WIO 5th.
[19] 박일균, "FreeBSD 상의 IPv6 스택 구현," WIO 5th
[20] 김영한 외 4인, "IPv6 호스트의 구현과 성능 분석," 정보과학회, 논문지 1997.12.
[21] 김영한 외 3인 "인터넷에서의 실시간 서비스를 위한 스케쥴링 알고리즘의 구현 및 성능 평가," 정보과학회, 논문지 1997.1.

김 영 한

1984. 2 서울대학교 전자공학과 학사
1986. 2 KAIST 전기 및 전자공학과 석사
1990. 8 KAIST 전기 및 전자공학과 박사
1987. 1~1994. 8 디지콤 정보통신 연구소 부장
1994. 9 ~ 현재 숭실대학교 정보통신전자공학과 조교수

박 일 균

1997. 2 숭실대학교 정보통신공학과 학사
1999. 2 숭실대학교대학원 정보통신공학과 공학석사
1999. 3~현재 숭실대학교 대학원 전자공학과 박사과정