Netfilter를 이용한 IP 주소 공유 기술 구현

김명선°, 황선엽, 원영암, 이광희, 최훈
충남대학교 컴퓨터공학과
{mskim05°, syhwang, yawon, leekwanghee, hc}@cnu.ac.kr

The Implementation of IP Address Sharing Method using Netfilter

Myoungsun Kim°, Sunyeop Hwang, Youngam Won, Kwanghee Lee, Hoon Choi
Mobile Distributed Computing Lab, Department of Computer Engineering,
Chungnam National University, KOREA

요 약

IP 주소 공유 기술인 FSL3/4는 IP 패킷의 헤더나 페이로드의 수정없이 단지 패킷 헤더 참조에 의해 로컬 네트워크의 호스트들에게 인터넷 풀 액세스 및 종단간 IPSEC 세션을 지원한다. 그러나 FSL3/4는 사용자 인터페이스를 제공하지 않으며 커널에 적재할 수 있는 모듈의 형태로만 존재하기 때문에 사용자 접근이 용이하지 않고, 추가적인 기능을 위해서는 커널 소스를 직접 수정해야 하는 불편함이 있다. 본 논문에서는 이런 문제점을 보완하고 FSL3/4 기능 확장을 쉽게 하는 Netfilter를 설계하고 구현하였다.

1. 서론

최근 WWW(World Wide Web) 서비스를 비롯한 다양한 인터넷 응용의 등장과 비효율적인 IP 주소 할당에 의하여 IP 주소 부족 현상이 발생하고 있다. IP 주소 부족 현상은 앞으로 다가올 유비쿼터스 컴퓨팅 시대에 심각한 위협이 되고 있다. IP 주소 부족 현상을 해결하기 위한 방법은 크게 두 가지가 있다. 장기적인 관점에서의 해결책으로 IPv4 주소 체계에서 IPv6 주소 체계로의 전환이 있지만 이는 현재 구축되어 있는 IPv4의 모든 네트워크 장비와 호스트를 전환해야 하므로 많은 시간과 비용이 요구된다. 이에 따라 IP 공유 기술이 단기적인 해결책으로 주목을 받고 있다.

IP 공유 기술에는 NAT(Network Address Translation), NAPT(Network Address Port Translation), NAT-FS(Network Address Translation by Flow Separation) 등의 네트워크 주소 변환 기술이 있다. 그러나 네트워크 주소 변환 기술은 IP 네트워크의 통신 특성인 종단간 연결성(end-to-end connectivity)을 제공하지 못하므로 종단간의 연결성을 요구하는 여러가지 세션을 지원하지 못한다[1]. 이에 반해 FSL3/4(Flow Separation by Layer 3/4)는 패킷의 변환없이 단지 참조를 통하여 데이터 플로우를 식별함으로써 종단간 연결성을 지원하고 보안 프로토콜 등 여러 가지 세션을 지원할 수 있다[2].

기존의 FSL3/4 기술은 사용자 인터페이스없이 커널에 적재할 수 있는 모듈의 형태로 개발되었기 때문에 사용자의 접근이 용이하지가 않다[3]. 또한 추가적인 기능을 제공하기 위해서는 소스(source)를 직접 수정해야 하는 불편함이 있다. 이에 따라 본 논문에서는 리눅스 환경에서 인터넷 패킷 맹글링(mangling)을 위하여 가장 대중적으로 사용되고 있는 Netfilter, Iptables에 기존의 FSL3/4 기능을 연동하는 기술을 제안하고자 한다.

본 논문의 구성은 2장에서 기존 FSL3/4에 대한 연구 동향을 살펴보고, 3장에서 FSL3/4 기능을 지원하는 Netfilter에 대한 구조에 대해 기술한다. 4장에서 현재 구현된 FSL3/4 기능을 지원하는 Netfilter에 대한 실험 결과에 대하여 기술하고, 끝으로는 결론 및 향후 연구 방향에 대해 제시한다.

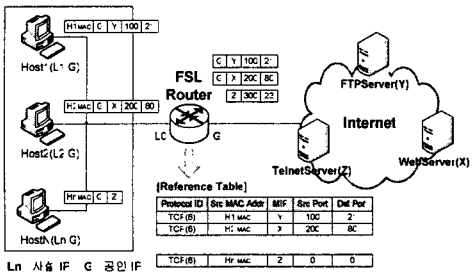
2. FSL3/4 구조

FSL3/4는 기존의 네트워크 주소 변환 기술에서 지원하지 못했던 종단간 연결성을 지원하기 위해 패킷에 대한 변환 작업없이 단지 패킷 헤더 정보의 참조(reading)에 의해 로컬 네트워크와 글로벌 네트워크의 통신을 지원한다.

FSL 라우터(router)에 연결된 로컬 네트워크의 호스트들은 각각 사설 IP를 갖고 있고, FSL 라우터가 갖고 있는 하나의

* 본 논문은 한국과학재단이 지정한 지역협력연구센터(RRC)인 충남대학교 소프트웨어연구센터의 지원으로 수행되었음

공인 IP 주소를 공유한다[3]. 로컬 네트워크 내의 통신은 사설 IP을 이용하여 통신하고, 글로벌 네트워크에 있는 인터넷 서버나 FTP 서버 등과 통신할 때는 공인 IP를 이용하여 패킷을 생성하고 FSL 라우터에게 전송한다.



[그림 1] FSL 3/4 구조

예를 들어, [그림 1]에서와 같이 로컬 네트워크에 호스트 1(Host1)이 글로벌 네트워크의 FTP 서버(Y)와 통신하려고 하면 패킷을 FSL 라우터로 전송한다. FSL 라우터는 로컬 네트워크로부터 패킷을 수신하면, 패킷을 전송한 로컬 호스트의 MAC(Media Access Control) 주소에 관한 참조 테이블을 생성하고, 일반 라우팅(routing)을 수행하여 패킷을 글로벌 네트워크에 있는 목적지 호스트로 전송한다. 서버로부터 이에 대한 응답 패킷을 받으면 참조 테이블을 검색하여 해당 엔트리를 찾아 MAC 주소를 이용하여 응답 패킷을 수신할 호스트 찾아 응답 패킷 전달한다[3].

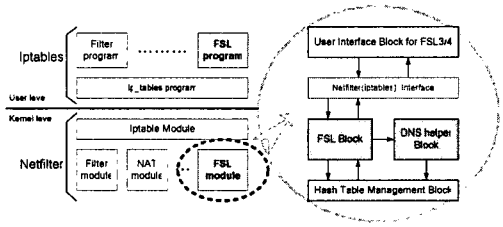
3. FSL3/4 기능을 지원하는 Netfilter 설계 및 구현

3.1. Netfilter

Netfilter는 인터넷 패킷 맹글링에 대한 프레임워크로서, 패킷을 필터링할 뿐만 아니라, NAT 기능 및 FSL3/4 기능을 갖는 리눅스 시스템을 운영할 수 있다. Netfilter는 훅(hook)을 정의하여 사용하며, 훅은 패킷이 지나가는 경로의 포인터이다. 훅에는 PREROUTING, POSTROUTING, INPUT, OUTPUT, FORWARD가 있다. 각 훅 포인터에서 패킷을 후킹(hooking)하여 패킷을 필터링하고, 맹글링할 수 있다. 또한, 커널 레벨 프로그램인 Netfilter 위에는 Iptables라는 사용자 레벨 프로그램이 있어 사용자가 인터넷 패킷을 필터링 및 맹글링할 수 있는 사용자 인터페이스를 제공한다[4].

3.2. FSL 기능을 지원하는 Netfilter 구조

FSL3/4 기능을 제공하는 Netfilter는 기존에 Netfilter와 Iptables가 제공하는 사용자 인터페이스를 이용한다. Netfilter에 FSL3/4 기능을 추가하기 위해 Netfilter의 필터링 모듈 및 NAT 모듈과 같은 레벨에 FSL 모듈을 추가하고, Iptables 프로그램에 FSL 프로그램을 추가한다. Netfilter에 추가된 FSL 모듈은 FSL3/4 기능 및 DNS helper 기능을 수행하고, Iptables에 추가된 프로그램은 사용자 인터페이스를 제공한다.

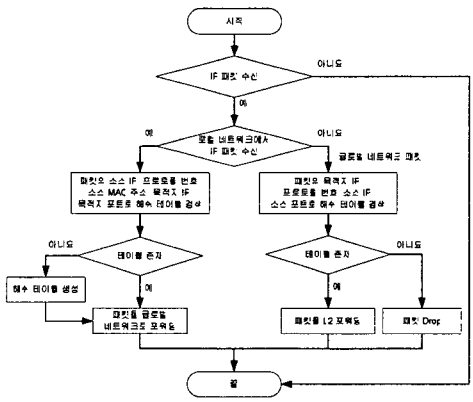


[그림 2] FSL3/4 기능을 지원하는 Netfilter의 구조도

[그림 2]와 같이 FSL 모듈은 FSL 블록(block)과 DNS helper 블록, HTM(Hash Table Management) 블록으로 구성된다. HTM 블록은 해쉬 테이블을 관리해주는 블록이다. HTM의 해쉬 테이블은 글로벌 IP, 프로토콜 번호, 소스 MAC 주소, 목적지 IP, 목적지 포트, 소스 포트, 타이머(timer)로 구성된다.

3.2.1. FSL 블록

FSL 블록은 FSL3/4의 라우팅 기능과 L2 포워딩(Layer 2 forwarding) 기능을 수행한다. [그림 3]은 FSL 블록의 기능에 대한 흐름도이다.



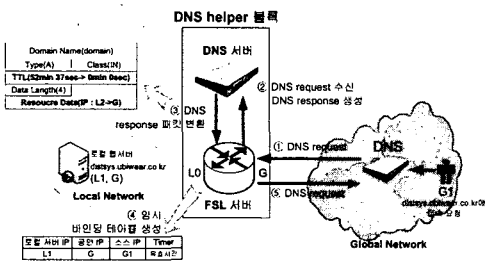
[그림 3] FSL Block 흐름도

3.2.2. DNS helper 블록

DNS helper 블록은 로컬 네트워크와 글로벌 네트워크 간에 양방향 서비스가 가능하도록 지원한다. 로컬 네트워크에 웹 서버가 존재하고, 글로벌 네트워크에 있는 단말이 웹 서버에 접속할 때 DNS helper 블록의 기능이 필요하다. DSN helper 기능을 제공하기 위해 FSL 서버 안에 DNS 서버가 존재해야 한다.

[그림 4]에서 (1)글로벌 네트워크의 DNS 서버로 부터 DNS request 패킷을 수신한 DNS 서버(FSL 서버)는 (2)로컬 네트워크에 있는 웹 서버의 IP를 담은 DNS response 패킷을 생성하

여 (5)글로벌 네트워크에 전송하게 된다. 이때 (3)DNS helper는 DNS response 패킷 중 Answer RR(Resource Recode)의 IP 부분을 FSL 서버에서 사용하는 공인 IP로 변경한다. 또한, 글로벌 네트워크의 단말이 DNS 정보를 캐시(cache)하지 않고 항상 질의하도록 Answer RR의 TTL(Time To Live)값을 0으로 변경한다. 이때 (4)로컬 네트워크의 사설 IP와 MAC 주소 값을 이용하여 임시 바인딩 테이블(binding table)을 구성하여 DNS 요청을 한 단말과 로컬 네트워크에 서버가 양방향 통신을 제공한다.



[그림 4] DNS helper 블록

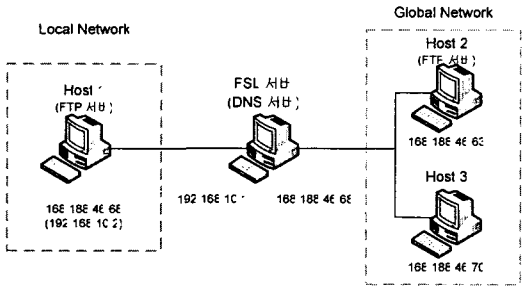
3.2.3. User Interface 블록

FSL3/4를 위한 사용자 인터페이스로서 기존 Iptables의 명령어를 확장하여 기존 Iptables 사용자들이 쉽게 명령어에 적응할 수 있도록 하였다. FSL3/4를 위해 nedia 테이블과 FSL 타겟을 사용한다. nedia 테이블 사용을 위해 -t nedia 옵션을 FSL 타겟 사용을 위해 -j FSL 옵션을 사용한다. 그 외 Iptables 명령어와 매칭 옵션들은 기존의 Iptables 명령어와 동일하게 사용할 수 있다. FSL 타겟을 위한 새로운 옵션들은 -j FSL 뒤에 사용한다. 사용자가 Iptables 명령어를 이용하여 작성한 명령어는 Netfilter의 nedia 테이블과 FSL 타겟에 전달되어 해당 작업을 처리한다.

4. 테스트

4.1. 테스트 환경

FSL3/4를 지원하는 Netfilter를 테스트하기 위한 환경 구축은 다음 [그림 5]와 같다.



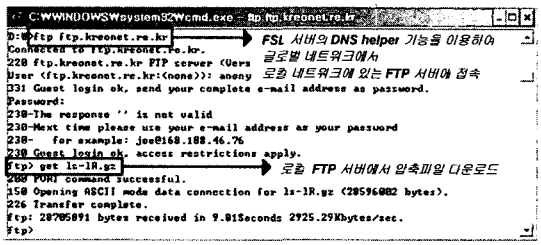
[그림 5] FSL3/4 기능을 지원하는 Netfilter 테스트 환경

FSL 서버의 운영체제는 리눅스이며, 로컬 네트워크와 글로벌 네트워크의 호스트들은 Windows XP 이다.

4.2. 테스트 및 테스트 결과

FSL 서버에서 FSL 명령어를 사용하여 FSL 모듈을 커널에 적재한다. FSL 블록의 기능을 테스트를 위해 로컬 네트워크에 있는 호스트 1에서 글로벌 네트워크의 FTP 서버인 호스트 2에 접속하여 테스트함으로써 FTP의 정상 작동을 확인하였다.

DNS helper 블록의 기능 테스트를 위해 글로벌 네트워크의 호스트 3에서 로컬 네트워크 FTP 서버인 호스트 1에 접속하여 FTP의 정상 작동 여부를 테스트하였다. [그림 6]은 DNS helper 블록 테스트 과정이다.



[그림 6] 테스트 환경에서의 FSL 동작 과정

5. 결론

기존의 FSL은 커널 기반의 소스 프로그램이기 때문에 소스를 수정할 경우 커널에 프로그램 수정사항을 반영하기 위하여 커널 컴파일 후 재시작을 해야하는 번거로움이 있었다.

논문에서 제시하는 FSL3/4를 지원하는 Netfilter는 모듈로 구성하여 소스 수정 시 모듈만 수정하여 컴파일함으로써 커널 컴파일 및 시스템 재시작에 대한 번거로움을 제거하였다.

향후 연구로서 FSL을 지원하는 Netfilter에서 패킷 후킹 및 처리 등에서 지연되는 시간을 감소시켜 네트워크 환경 이외의 지연 속도를 최소화시킬 수 있는 방안을 연구하고, 보다 간편하고 자동화된 툴을 개발할 계획이다.

6. 참고문헌

[1] P. Srisuresh and M. Holredge, "IP Network Translator (NAT) Terminology and Considerations," RFC 2663, IETF, August 1999.

[2] 이광희, 오명환, 최훈, "호스트 라우팅을 이용한 공인 IP 주소 공유 기법", 한국정보과학회 추계학술발표 논문집 제 30권 2호(3), pp. 352-354, 2003.

[3] K.wang-Hee Lee, Hoon Choi, "FSL3/4 on NEDIA (Flow Separation by Layer 3/4 on Network Environment using Dual IP Address," LNCS 3090, pp. 1015-1024, 2004.

[4] http://www.netfilter.org/documentation/HOWTO//netfilter-hacking-HOWTO.html