VPN을 위한 리눅스 상에서의 IPSec 설계 및 구현

김정범 박남섭 김태윤
고려대학교 컴퓨터학과
{qston, nspark, tykim}@netlab.korea.ac.kr

IPSec design and implement under Linux for VPN

Jeong-Beom Kim, Nam-Sup Park, Tai-Yun Kim
Dept. of Computer Science and Engineering, Korea University

요 약

최근 리눅스에 대한 사용이 빠른 속도로 증가하고 있다. 하지만 리눅스의 오픈 소스 정책에 따른 리눅스 보안의 필요성이 대두되어 리눅스 기반의 효과적인 암호 개발이 급속히 확산되고 있다. 이러한 것을 보완하기 위한 해결책으로써 리눅스 상에서 IP 계층의 보안 프로토콜인 IPSec이 개발되었다. 하지만 이렇게 구현된 IPSec은 사용자가 설치 및 운영이 매우 어렵다. 이에 본 논문은 새로운 커널 컴파일 없이 ethertap을 이용한 IPSec 구조의 설계 및 구현한다.

1. 서론

네트워크를 통해 전세계의 컴퓨터들이 네트워크로 연결되면서 해킹이나 바이러스와 같은 침해 사고들이 빈번하게 발생하여 많은 피해가 발생하고 있다. 이에 따라 시스템의 안전에 대한 인식도 증가하여 시스템 환경에서 보안 기능을 추가하려는 노력도 증가하여 왔다. 이러한 시스템들 중 리눅스가 무료로 보급되면서 리눅스의 취약성을 이용한 해킹 사례가 빈번하게 보고되고 있는 실정이다. 리눅스는 소스 코드의 공개와 관련문서의 풍부한 제공으로 쉽게 수정이 가능하여 많은 사람들의 사용이 확산되고 있다. 이러한 시스템을 가지고 사람들의 재택 근무가 활발해지고 기업 외부와도 네트워크 구성이 필요하게 되는 등의 기업 네트워크가 점차 확대되어감에 따라 막대한 시설 투자가 필요하게 되었다. 네트워크의 확대와 함께 네트워크에 연결된 서로간에 안전한 통신을 하기 위해 사용해 오던 전용망에 투자해야 하는 비용과 그에 따른 운영과 관리가 커다란 문제가 되고 있다. VPN(Virtual Private Network)이란 이런 문제들의 해결 방안으로서 기업의 네트워크를 구성할 때 전용 임대회선을 사용하는 것이 아니라 공용망인 인터넷망을 이용하는 연결망이다. VPN은 터널링(Tunneling)이라는 기법을 사용하여 일대일 연결과 같은 터널을 형성하며 데이터 패킷들은 터널을 통하여 안전하게 전달된다. 이러한 터널링을 구현하는 기술 중 리눅스에서는 IETF(Internet Engineering Task Force)에 의해서 IP 계층 보안을 위해 설계된 IPSec(Internet Protocol Security)이 개발되었으나 커널 컴파일과 사용자의 이용이 어려워 많이 쓰이지 않았다. 본 논문에서는 이러한 어려움을 해결하고자 리눅스 상에서 ethertap을 이용한 IPSec을 구현한다.

본 논문의 구성은 다음과 같다. 2장에서는 관련 연구로 IPSec에 대해 분석한다. 3장에서는 IPSec 구현과 이를 이용한 VPN 구현에 대해 설명하고 4장에서는 결론 및 향후 연구과제를 제시한다.

2. 관련연구

2.1 IPSec(Internet Protocol Security)

IPSec은 다음과 같은 보안 서비스를 제공한다. 이 서비스는 선택적이며 일반적으로 로컬 보안 정책은 이 서비스 중에서 하나나 그 이상의 서비스를 선택한다.

- 데이터의 비밀성 - IPSec의 송신자는 네트워크로 전송되기 전에 데이터를 암호화하여 보낸다.
- 데이터의 무결성 - IPSec 수신자가 보내는 데이터는 받는 사람의 데이터와 같다.
- 데이터의 인증 - IPSec의 수신자는 패킷을 보낸 송신자의 인증을 확인할 수 있다.
- 재사용성 방지 - IPSec 수신자는 재사용 공격을 방지할 수 있다.

IPSec 동작에는 세 가지 기본 구성요소가 필요하다. 즉, Security Association(SA), Authentication Header(AH), Encapsulating Security Payload(ESP)가 IPSec 동작에 중요한 역할을 한다.

● Security Association (SA)

데이터 송수신자간에 비밀데이터(인증되었거나, 암호화된 데이터)를 교환할 때 사전에 암호 알고리즘, 키 교환방법, 키 교환 주기 등에 대한 합의가 이루어져야 한다. 데이터 교환 전에 통일되어야 할 이러한 요소들을 IPSec에서는 SA로 정의한다. 데이터 송수신자간의 안전한 통신을 위해서는 적어도 하나의 SA가 필요하다. 그러므로 패킷 인증과 암호를 위해서는 SA가 선행되어야 한다. 동일한 알고리즘이 사용되어도 서로 다른 두 개의 키가 요구될 경우에는 두 개의 SA가 필요하다. SA를 대인간 또는 그룹간, 네트워크간의 네트워크 보안

채널로 생각할 수도 있다. SA는 다중 VPN 구성에 유리하다. 다수의 상대방이 있을 경우 개인별로 SA를 별도로 정의함으로써 서로 다른 VPN을 구성할 수 있다. 또한 SA는 일방향 전송도 가능하다. 하나의 SA가 정의될 경우 송신자는 수신자에게 데이터를 전송할 수 있으나 동일한 SA로 데이터의 수신은 불가능하다. 따라서 양방향 통신을 위해서는 수신자가 전송자에게 보내는 SA가 별도로 정의되어야 한다.

● Authentication Header(AH)

AH는 IPSec에서 IP 데이터에 대한 인증 서비스와 데이터 무결성 서비스를 제공한다. AH는 패킷의 체크섬 결과를 포함하고 있다. AH는 1)next header field, 2)payload length, 3)security parameter index(SPI), 4)sequence number, 5)authentication data 다섯 가지 필드를 포함한다. 다섯 가지 필드 중에서 SPI는 송신자가 사용하는 보안 프로토콜에 대한 정보를 알려주며, authentication data는 SPI에서 정의된 암호알고리즘으로 패킷의 payload를 암호화하여 얻어진다. IPSec은 인증 기능의 향상을 위해 패킷 체크섬 계산 방법을 기존 MD5 방식 대신 HMAC(Hash-based Message Authentication Code)-MD5와 HMAC-SHA 방식 중 하나를 사용하고 있다.

● Encapsulating Security Payload(ESP)

ESP는 패킷의 암호화 서비스를 제공한다. AH처럼 패킷 처리에 필요한 SA 정보를 수신자에게 알려주기 위해 SPI를 포함하고 있다. ESP는 다양한 암호화 알고리즘을 지원하며 사용자는 상대방에 따라 서로 다른 암호 알고리즘을 사용할 수 있다. ESP에서도 인증 서비스를 제공할 수 있으나 이 경우 AH와 달리 IP 헤더에 대한 인증 서비스는 제공하지 못한다.

이외에도 IPSec에서는 키 관리가 중요하다. 키 관리는 IPSec과는 독립적으로 구현되므로 여러 가지 키 관리 프로토콜 중에서 선택하여 사용할 수 있다. IPSec에서 사용 가능한 대표적인 키 관리 프로토콜로는 SKIP, Photuris, ISAKMP 등이 있다. 키 관리 시에는 네트워크 상에서 키의 안전한 교환, 키의 주기적인 변경, 송, 수신자간의 키 협상 등을 고려하여야 한다.

3. IPSec 설계 및 구현

본 장에서는 IPSec 구현에 필요한 커널 모듈과 IPSec 구현에 대해 설명한다. 구현 환경으로는 리눅스 커널 버전 2.2.16을 이용하였다.

3.1 ethertap을 이용한 IPSec 설계

먼저 Ethertap은 사용자 공간에서 패킷을 송수신하는 역할을 한다. 이것은 네트워크 상에서 패킷을 수신하는 더넷 디바이스의 기능을 수행하며 그것은 사용자 공간으로부터 패킷을 받는다.

Ethertap 사용자 공간으로부터 네트워크 스택을 쉽게 실행할 수 있다. 유저 공간 프로그램들이 이더넷 프레임들을 그 파일을 통해 읽고 쓸 수 있을 것이다. tap0는 ifconfig와 route 명령에 의해 다른 이더넷 장치들처럼 설정될 수 있지만 어떤 물리적 LAN에도 연결되지 않고 사용자가 /dev/tap0에 쓰는 모든 것이 마치 LAN으로부터 tap0 장치에 온 것처럼 커널에 의해 다뤄진다. 커널이 tap0 장치 너머로 보내려 하는 모든 것들이 사용자에게는 /dev/tap0로부터 읽혀진다. ethertap을 활성화시키고 ifconfig /dev/tap* 192.168.1.1으로 설정한다. (replace 192.168.1.1 with the -proper IP number for your situation.)
그리고 나서 Netlink를 설정한다.

동작과정은 다음과 같다. VLAN (virtual LAN) 은 두 개의 물리적인 네트워크가 하나의 네트워크로 구성된 것 처럼 구성되어 있다. 게이트웨이 gw1 과 gw2 로 구성된 2개의 분리된 네트워크를 구성요소로 한다.

mknod 명령으로 메이저 넘버 36 마이너 넘버 16번인 캐릭터형 특수 파일 /dev/tap0을 만들었다. 유저 공간 프로그램들이 이더넷 프레임들을 그 파일을 통해 읽고 쓸 수 있을 것이며, 유저 공간에서 만들어진 파일들을 커널에 가져와서 네트워크를 통해 가져온 것처럼 읽혀진다. 따라서 사용자가 /dev/tap0에 쓰는 모든 것이 마치 LAN으로부터 tap0 장치에 온 것처럼 커널에 의해 다뤄진다. 커널이 tap0 장치 너머로 보내려 하는 모든 것들이 사용자에게는 /dev/tap0로부터 읽혀지므로 결국은 사용자 공간 프로그램들과 커널과의 파일의 이동이라고 볼 수 있다.

또 다른 모듈인 Netlink는 linux kernel 2.0까지는 원시적인 device 기반의 인터페이스를 제공하였으나, kernel 2.2부터 socket 형태의 사용하기 편리한 인터페이스로 바뀌게 되었다. netlink의 주요특징은 다음과 같다.

- Netlink는 NETLINK_ROUTE, NETLINK_FIREWALL, NETLINK_ARPD, NETLINK_ROUTE6, NETLINK_IP6_FW, NETLINK_TAP, NETLINK_SKIP,NETLINK_USERSOCK 이라는 family를 이루고 있는데, 주로 네트워크 정보를 읽거나 지정하는 기능에 사용된다.
- netlink는 Linux kernel space와 user space간의 정보 전달을 위한 interface protocol을 말한다.
- Linux kernel source에서 net/netlink에 소스 코드가 있으며, Linux2.0에서의 primitive device 기반의 형태에서 발전하여 Linux 2.2에서는 socket 형태로 일반화된 interface를 제공한다.

그림 19는 netlink의 구조를 보여주고 있다.

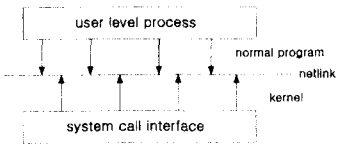

그림 2 netlink 구조

3.2 IPSec 구현

먼저 구현 환경을 설명하면 운영체제는 리눅스이고 커널 버전은 2.2.16이다. IPSec에서 쓰일 암호화 방법은 OpenSSL을 이용하였다.

그리고 위에서 설명한 각각의 모듈을 사용하는 IPSec 프로그램의 함수 다이어그램은 다음과 같다.

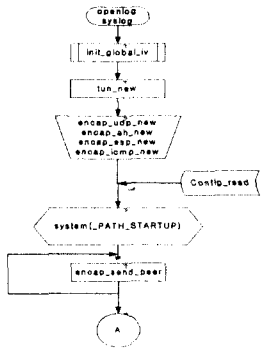

그림 2 나가는 패킷 처리

그림을 설명하면, 먼저 시스템 로그 파일을 연후에 프로그램 시작 로그를 남긴다. 동시에 IV(Initialization Vector)를 초기화시키고 ethertap 자료구조를 저장할 수 있는 영역을 할당하여 초기화한다. UDP, AH, ESP, ICMP 처리 루틴을 초기화하고 설정 파일을 읽어들여서 메모리에 적재한다. 그러고나서 초기화 파일을 실행시키면 설정 파일에서 peer로 설정된 호스트로 패킷을 보낸다.

들어오는 패킷 처리는 다음과 같다.

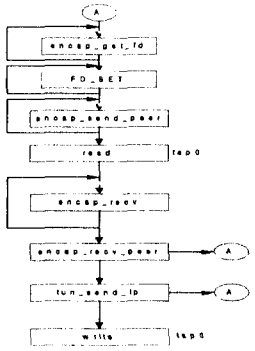

그림 3 들어오는 패킷 처리

그림을 보면, 먼저 tap0~15로 들어오는 패킷이 있는지 polling한 후들어오는 패킷이 있는 경우 FD(File Descriptor)를 설정하여 설정 파일에서 peer로 설정된 호스트로 IPSec 루틴을 거친 패킷을 보낸다. tap0~15로 들어오는 실제 패킷을 버퍼로 읽어들이며 들어오는 패킷의 종류에 따른 수신 루틴을 실행한다. 패킷의 종류에 따라 IPSec 루틴을 거친 후 실제 패킷을 얻어낸다. 해당 어드레스로 실제 패킷을 보낸다.

이렇게 구현된 IPSec을 이용한 전체 VPN 구조는 다음과 같다.

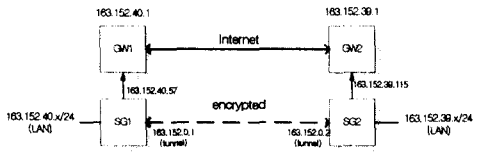

그림 4 실험 환경

그림과 같이 163.152.40.x/24 라인으로 연결된 LAN A와 163.152.39.x/24 라인으로 연결된 LAN B를 중간 노드로 SG(Security Gateway) A와 SG B를 이용해서 구현하였다. SG A와 SG Ben 노드는 PPP(Point-to-Point)방식을 이용하여 연결한다. 이를 위해서 앞에서 언급한 ethertap과 netlink 모듈을 사용한다. 또한 tap0와 실제 패킷을 전송하는 eth0와의 연동은 startup이라는 설정 파일에서 해주게 되는데 이것의 역할은 tap0의 가상 IP를 통하여 SG1과 SG2 사이를 가상 링크로 연계시켜주는 역할을 한다. 터널링에 따른 AH나 ESP 중 IPSec 터널에서 어떠한 방식을 이용할 것이며 암호화 방식을 구별해주는 SPI는 무엇인지 등의 SA와 관련된 내용은 config 파일에서 설정한다.

4. 결론 및 향후 연구 과제

본 연구에서는 VPN을 구성하는데 있어 핵심 기술이라 할 수 있는 IPSec을 새로운 커널 컴파일 없이 사용할 수 있게 구현하였다. IPSec의 개발 시 여러 가지 구현 방법이 있지만 본 연구에서는 서브넷 간을 연결하는데 중점을 두었다. 서브넷의 임의의 호스트에 개발 모듈을 설치하고 그 호스트를 SG로 설정하였고 같은 방식으로 다른 서브넷에서도 이러한 환경을 제공하였다. 결구 이러한 SG 간에서 일어나는 가상적인 터널을 통해 두 서브넷은 가상적인 하나의 서브넷을 구성할 수 있었다.

향후 연구과제로는 제안된 모델에서는 클라이언트들 간이 트랜스포트 모드를 지원하지 않고 있으며 사용되는 암호 알고리즘을 openSSl을 사용하고 있어 향후 보안상의 공격가능성이 있다고 할 것이다. 향후 트랜스포트 모드 지원을 통해 호스트간의 터널링 역시 지원될 수 있도록 연구가 지속되어야 하고 독자적으로 개발된 암호 알고리즘을 사용하여 보안에 있어서 좀 더 깊은 연구가 진행되어야 할 것이다. 또한 현재 사용한 리눅스 커널 버전이 2.2.x 버전이며 이러한 커널의 모듈 중 ethertap과 netlink 모듈을 이용했는데 커널이 2,4.x에서는 이러한 모듈을 통합한 형태인 netfilter 모듈을 지원한다. 따라서 이를 이용한 연구가 계속 진행되어야 할 것이다.

5. 참고 문헌

[1] Atkinson, R., "Security Architecture for the Internet Protocol", RFC 2401, NRL,

[2] Atkinson, R., "IP Authentication Header", RFC 2401 August 1998.

[3] Atkinson, R., "IP Encapsulating Security Payload", RFC 1827, NRL, August 1995.

[4] Internet Security Association and Key ManagementProtocol(ISAKMP),Douglas Maughan, Mark Schertler, Mark Schneider, Jeff Tunner, INTERNET-DRAFT draft-ietf-ipsec-isakmp-08.txt,.ps,July26, 1997.

[5] Tailer, James S. "IPSec Virtual Private Networks"

[6] Dan harkins, "The New Security Standard for the Internet Intranets, and Virtual Private Networks"