# 통합 방화벽에서 터미널 호스트 추적 기법의 설계 및 구현 *

이승원○ 조유근
서울대학교 컴퓨터공학과

# Design and Implemenatation of Terminal Host Tracing Method on Hybrid Firewall System

Seungwon Lee Yookun Cho
Dept. of Computer Engineering, Seoul National University

요 약

터미널 호스트(Terminal Host)란 사용자가 네트워크를 통하지 않고 직접 사용하고 있는 호스트를 말한다. 사용자가 네트워크를 통해 특정 호스트에 로그인할 경우, 특정 호스트를 지키던 방화벽은 해당 사용자의 터미널 호스트를 인증하는 것이 아니라 해당 사용자가 네트워크을 통해 직접 특정 호스트에 연결시킨 호스트만을 인증하게 된다. 만약 사용자의 터미널 호스트가 보안상 위험에 노출되어 있다면 특정 호스트는 방화벽으로 보호를 하고 있음에도 불구하고 역시 똑같은 위험에 노출되게 되어 보안상의 커다란 위협이 된다. 본 논문에서는 이러한 위협을 없애기 위해 통합 방화벽에 적용가능한 효율적인 터미널 호스트 추적 기능을 제안하고, 터미널 호스트 추적 기능을 갖춘 FreeBSD기반의 통합 방화벽의 설계 및 구현 내용을 기술한다.

## 1. 개요

방화벽은 외부 네트워크의 호스트와 내부 네트워크의 호스트 사이의 연결들을 관리자가 정의한 규칙에 따라 제어하는 네트워크 보안도구이다 [2]. 방화벽의 기능은 외부 네트워크의 호스트들 중에서 보안의 위협이 될 호스트들의 연결을 차단하는 것이다. 모든 방화벽들은 IP 패킷 헤더에 적혀있는 목적지 호스트의 IP주소와 출발지 호스트의 IP주소만을 기준으로 연결을 허락할 것인가 거부할 것인가를 결정한다. 방화벽을 통과할 시점에서 연결을 요청한 호스트와 연결의 대상이 될 호스트의 IP주소만으로 연결의 여부를 결정하는데 방화벽이 이렇게 판단하는 이유는 연결을 통해 내부 네트워크의 호스트를 사용하는 사용자가 연결을 요청한 호스트에서 사용하고 있다는 가정에서 출발한다. 그러나, telnet이나 rlogin같은 연결중심(connetion-oriented)의 서비스의 경우에는 연결을 요청한 호스트와 실제 사용자가 있는 터미널 호스트가 일치하지 않는 경우가 많이 있다. 이런 경우에 방화벽은 터미널 호스트에 대한 아무런 정보를 가지고 있지 않으므로 터미널 호스트에 대한 어떠한 인증 작업도 할 수 없게 된다. 따라서, 연결된 내부 네트워크의 호스트가 보안상의 위협에 완전히 노출된 것이나 마찬가지의 상황에 처하게 된다.

이러한 상황을 막기위해 방화벽이 사용자의 터미널 호스트를 알수 있는 터미널 호스트 추적 기능이 필요하다. 터미널 호스트 추적 기능을 구현하기 위해서는 같은 사용자가 telnet이나 rlogin 등으로 연결한 호스트들의 연결 상황을 알아야 한다. 이러한 연결 상황을 회선체인(connetcion chain) 이라고 부른다. 즉, 통합 방화벽에서 효율적인 회선체인 구성 기법이 효율적인 터미널 호스트 추적 기법이 되는 것이다. 본 논문에서는 통합 방화벽에 적용가능한 효율적인 터미널 호스트 추적 기법을 제안하고, 터미널 호스트 추적 기능을 갖춘 FreeBSD기반의 통합 방화벽의 설계 및 구현 내용을 기술한다.

본 논문은 다음과 같이 구성되어 있다. 2.장에서는 기존의 회선체인 구성 기법들에 대해 기술하며 방화벽들의 종류와 각 장단점에 대해 기술하고 3.장에서는 UNIX계열의 공용 운영체제인 FreeBSD 상에서 통합 방화벽의 구현 방식과 효율적인 터미널 호스트 추적 기법의 구현에 대해 기술한다.

*본 연구는 과학기술부의 특정연구개발사업인 "Internet을 위한 방화벽 및 네트워크 통신 보호 시스템 개발(과제번호 sw-09-01)"의 지원에 의해 이루어졌다

## 2. 배경지식

### 2.1 회선체인 구성기법

한 사람(혹은 하나의 프로그램)이 한 호스트에 로그인 한 후 모뎀이나 네트워크 회선을 통해 또 다른 호스트들에 계속해서 로그인을 했을 경우 이 두 호스트들은 한 사용자에 의해 telnet이나 rlogin으로 연결되게 된다. 이렇게 한 사용자에 의해 telnet이나 rlogin으로 연결된 호스트들의 집합을 회선체인(connection chain)이라 부른다 [1] 기존의 제안된 회선체인 구성하는 방식은 크게 세 가지가 있을 수 있다

첫 번째 방식은 네트워크에 있는 모든 사용자를 다 추적하고 각 사용자에게 전체 네트워크에서 식별이 가능한 식별자를 주어 모든 사용자의 행동을 기록하여 회선체인을 만드는 방법이다. UC Davis에서 개발한 DIDS는 하나의 LAN에서 이러한 방식을 구현한 예이다 [4]. 이 방식으로 구현된 시스템은 중앙관리시스템이 있는 광역 네트워크에서는 효율적이겠지만 인터넷같이 중앙관리시스템이 없이 분산되어 관리되는 광역 네트워트에서는 별로 효과를 보기 힘들다.

두 번째 방식은 역추적 방식이다. 이 방식은 전체 네트워크의 모든 사용자를 추적하거나 기록을 남길 필요가 없는 대신 네트워크에 있는 모든 호스트가 추적시스템을 가지고 있어야 한다. 추적시스템은 회선체인에서 다음 호스트가 어디인지를 나타낼 수 있는 능력을 가지고 있는 시스템이다. 따라서 회선체인을 만들려고 하는 호스트는 회선체인에 속한 호스트와 통신을 통해 회선체인의 다음 호스트가 어디인지를 알아서 역추적하게 하는 방식이다. 이 방식으로 구현된 회선체인 구성시스템의 예로는 CIS(Caller Identification System)이 있다 [5]. 이 시스템은 네트워크에 있는 각 호스트의 사용자가 어느 호스트에서 로그인을 하였는지를 알 수 있는 로그정보를 기록할 수 있도록 각 호스트에 추적시스템을 설치한다. 이 방식의 문제점은 회선체인의 중간 호스트가 추적시스템을 가지고 있지 않을 경우 회선체인의 구성이 실패한다는 것이다.

세 번째 방식은 Thumbprint기법을 이용한 회선체인 구성기법이다. Thumbprint란 네트워크 회선의 특정부분을 효과적으로 요약하는 식별자를 의미한다 [3]. 이 기법은 동일한 시간에서 볼때 회선체인을 구성하는 어떤 회선이든지 회선을 지나가는 내용은 모두 동일하다는 사실에 기반을 두고 있다. 따라서 회선체인 구성시스템이 동일한 시간대에 모든 회선이 갖는 내용에 대한 Thumbprint를 만들수 있다면 그 Thumbprint들을 가지고 어떤 회선들이 같은 Thumbprint를 가지고 있는지를 판단하여 회선체인을 구성할 수 있게 된다 Thumbprint를 이용한 회선체인 구성기법은 앞에서 언급한 두 가지 방식과는 달리 모든 호스트가 회선체인 시스템을 설치할 필요도 없고 모든 사용자의 모든 행동에 대한 정보를 기록할 필요가 없다. 또한 회선체인의 중간호스트를 알아낼 수 없다고 하여도 회선체인 시작호스트를 알아낼 수 있는 장점을 가지고 있다. 그러나 이 기법은 만약 관계없는 두 회선의 내용이 동일하다면 같은 회선체인으로 여길수 밖에 없는 단점을 가지고 있다. 이러한 단점을 방화벽에서 없앤 회선체인 구성 기법이 [6]에서 제안되었다.

본 논문에서는 [6]에서 제안한 기법을 이용한 터미널 호스트 추적 기법을 통합 방화벽에서 구현하였다.

### 2.2 통합 방화벽

방화벽의 형태는 크게 세 가지로 나뉘어 진다. 첫 번째 형태는 라우터 방식 방화벽으로 내부 네트워크를 외부 네트워크와 물리적으로 구분한 다음 그 사이에 리우터를 설치하여 라우터를 통해서 외부 네트워크와 내부 네트워크가 연결될 수 있도록 한 후 라우터에 팻킷 필터링(packet filtering)기능을 추가하는 형태이다.

두 번째 형태는 응용프로그램 게이트웨이 방식 방화벽으로 리우터를 이용하거나 물리적 구성을 통해 외부 네트워크와 내부 네트워크를 같이 연결할 수 있는 호스트를 지정한 다음 그 호스트에 응용 프로그램(telnet, ftp 등)별로 proxy를 두어 내부 네트워크로 연결을 하려는 외부 호스트는 직접 내부 네트워크의 호스트로 연결하는 것이 아니라 proxy가 있는 호스트를 통해 내부 네트워크 연결하도록 하는 형태이다. 이때, proxy는 내부 네트워크와 외부 네트워크의 응용프로그램의 사이에서 데이터를 전달하게 함으로써 외부 네트워크와 내부 네트워크를 연결되도록 한다.

세 번째 형태는 위의 두 가지 형태의 방화벽들을 통합하여 만든 통합 방화벽이다. 통합 방화벽 시스템은 크게 두 부분으로 구성되는데 하나는 네트워크 인터페이스 카드간에 패킷을 전달하고 패킷에서 필요한 정보를 얻어내는 역할을 하는 필터 모듈이고 다른 하나는 필터 모듈에서 전달해 준 정보와 네트워크 관리자가 만든 접속 제어 규칙을 가지고 회선의 접속을 허락하거나 감시하는 것을 결정하는 방화벽 모듈이다.

필터 모듈은 라우터 방식 방화벽 시스템의 라우터와 같은 기능을 수행하기 때문에 통합 방화벽은 게이트웨이 방식 방화벽 과는 달리 사용자에게 투명성을 보장해 줄 수 있고 대부분의 패킷이 네트워크 인터페이스 카드에서 다른 인터페이스 카드로 직접 전달되기 때문에 사용자 데이터를 패킷에서 뽑아낸 다음 다시 패킷을 만들어 전달하는 게이트웨이 방식 방화벽보다 훨씬 좋은 성능을 보이고 있다. 또한 방화벽 모듈이 네트워크 서비스별로 자세한 접근 제어를 할 수 있기 때문에 게이트웨이 방식 방화벽처럼 자세한 접근 제어가 가능하다.

## 3. 구현

### 3.1 개요

본 논문에서 제안하는 터미널 호스트 추적 기법은 그림 1와 같다. 그림 1에서 터미널 호스트는 C이고 사용자는 호스트 B와 A를 telnet을 통하여 연결한 후 방화벽을 통해 내부 네트워크로 연결하려고 한다고 가정하자. 방화벽은 내부 네트워크로 연결을 시도하는 호스트 A는 신뢰하는 호스트 †이므로 일단 연결을 허락한 후 내부 네트워크

†그림 1에서 짙은 색깔의 타원은 신뢰하는 호스트를 나타낸다.

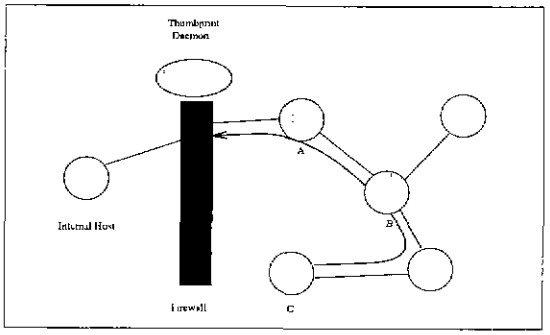


그림 1: 터미널 호스트 추적 기법 개요

에서 외부 네트워크로 나가는 패킷의 사용자 데이타부분에 미리 정한 특수문자열을 첨가시킨다. 이때 Thumbprint데몬은 외부 네트워크의 회선들을 감시하며 앞에 설명한 특수문자열이 지나간 회선들을 가려낸 다음 회선체인 구성 알고리즘 [6]을 이용해 터미널 호스트를 추적해 방화벽에게 알려준다. 터미널 호스트를 알아낸 방화벽은 호스트 C가 신뢰하는 호스트가 아니므로 연결을 끊어버리게 된다.

이러한 터미널 호스트 추적 기법을 구현하기 위해서는 두 개의 모듈이 필요하다. 내부 네트워크와 외부 네트워크를 연결하고 특정 연결에는 미리 정한 특수문자열을 첨가시키는 필터 모듈과 외부 네트워크를 감시하고 각 연결들의 Thumbprint를 계산하여 동일한 Thumbprint를 갖는 연결들로 회선체인을 구성하는 Thumbprint 데몬이다. 본 장에서는 필터 모듈과 Thumbprint데몬의 구현에 대해 기술한다.

### 3.2 TCP 프로토콜

TCP(Transmission Control Protocol)은 네트워크 프로토콜로 다음과 같은 특징을 가지고 있다. 먼저, TCP는 클라이언트와 서버 사이에 연결(connection)을 제공한다. TCP클라이언트는 주어진 서버와 연결을 생성하고 생성한 연결을 통해서 서버와 데이터를 주고받은 후에 그 연결을 종결시킨다.

TCP는 신뢰성(reliability)을 제공한다. TCP에서 데이터를 상대편으로 보냈을 경우, TCP는 응답으로는 답신(acknowledgement)을 요구한다. 만약 답신을 받지 못하면, TCP는 자동으로 데이터를 다시 보내고 좀더 오랜 시간동안 답신을 기다린다. 몇 번의 재전송 이후, 4분이나 10분 정도의 시간의 지나면 TCP는 데이터 전송을 포기한다.

TCP는 TCP가 보내는 모든 데이터를 순서(sequence)번호와 연관시켜 데이터의 순서를 유지한다. 예를 들어, 어떤 응용프로그램이 2048비이트를 TCP소켓에 write했으면, TCP는 두 개의 세그먼트를 보낸다. 첫 번째는 순서번호가 1-1024까지 연관된 데이터를 가진 세그먼트이고 두 번째는 순서번호가 1025-2048까지 연관된 데이터를 가진 세그먼트이다. 만약 세그먼트들이 순서가 바뀌어 도착하게 되면 데이터를 받는 응용프로그램에게 전달하기 전에 수신자 측의 TCP가 순서번호에 따라 세그먼트들을 재 정렬한다.

### 3.3 FreeBSD의 IP 패킷 처리과정

FreeBSD의 IP 패킷 처리 과정은 다음과 같다.

먼저, 네트워크 인터페이스 카드에 IP 패킷이 도착하면 네트워크 인터페이스 카드는 인터럽트를 발생시키게 된다. 네트워크 인터페이스 카드의 인터럽트는 ipintr()라는 함수를 호출한다. 이 함수는 인터럽트 핸들러로 인터럽트가 발생할 때마다 호출된다. ipintr()을 새로운 패킷이 저장된 버퍼의 주소를 ip_input()함수에게 전달하고 ip_input()을 호출한다. ip_input()은 IP수준의 옵션을 처리하고, IP헤더의 정보를 처리하고, 단편화된 IP 패킷도 하나도 통합한

후 프로토콜 계층상 상위계층인 tcp_input()함수에게 처리된 패킷의 버퍼 주소를 전달하고 tcp_input()을 호출한다. tcp_input()은 TCP프로토콜이 제공하는 여러 가지 기능을 위한 과정을 처리한 후 사용자 데이터만 뽑아서 데이터를 기다리고 있는 프로세스의 버퍼에 복사한 후 해당 프로세스에게 시그널을 보내준다

사용자 데이터가 네트워크 인터페이스 카드를 통해 다른 호스트로 전달되기 위한 과정은 역으로 진행된다. 사용자가 소켓을 통해 보내고 싶은 데이터를 전달하면 소켓시스템호출은 tcp_output()을 호출한다. tcp_output()은 사용자 데이터에 tcp 메시지 헤더를 첨가하고 헤더의 각 필드를 적당한 값으로 채운 후 해당 버퍼의 주소를 ip_output()에 전달하고 ip_output()을 호출한다. ip_output()은 IP 패킷 헤더를 첨가하고 필요한 필드의 값을 계산하여 채운 후 네트워크 인터페이스 카드를 나타내는 디바이스 파일을 통해 네트워크 인터페이스 카드에 데이터를 전달한다.

### 3.4 필터 모듈

필터 모듈은 정해진 규칙에 따라 패킷을 내부 네트워크와 외부 네트워크 사이에서 전달하는 역할을 담당하는 부분으로 FreeBSD 커널 내부에 구현되었다. 터미널 호스트 추적 기능을 통합 방화벽에서 구현하기 위해서는 필터 모듈이 특정 TCP연결을 지나가는 패킷에 미리 정한 특수 문자열을 첨가하여야 한다. 즉, 서버에서 클라이언트로 보내지는 패킷에 필터 모듈이 새로운 데이터를 첨가해야 하는 것이다. 서버가 보내는 원래 패킷에 새로운 데이터를 첨가하기 위해서는 데이터영역에 새로운 데이터를 첨가하는 것 외에도 몇 가지 문제를 해결해야 한다. 첫 번째로 IP 패킷헤더의 있는 패킷 크기를 나타내는 필드를 맞게 고쳐야 하고 IP 패킷헤더 checksum값을 새로 계산해서 고쳐줘야 한다. 두 번째로 TCP 메시지헤더를 고쳐야 하는데 이 부분은 조금 복잡하다. TCP는 신뢰성을 제공하기 위해 메시지의 모든 데이터는 하나의 순서번호와 연관를 맺고 있다. 따라서 새로운 데이터를 첨가하기 위해 새로운 순서번호를 추가시켜야 한다. 새로 추가된 순서번호는 실제로 메시지를 보낸 서버에서는 모르기 때문에 클라이언트가 응답번호를 보내는 패킷에서 새로 추가된 순서번호에 대한 응답번호는 제거해야 한다. 물론 이러한 작업을 한 TCP 메시지는 모두 다시 TCP checksum을 다시 만들어야 한다.

### 3.5 Thumbprint 데몬

Thumbprint 데몬은 libpcap-1 7이라는 라이브러리를 이용하여 구현하였다. libpcap은 UC Berkeley에서 개발한 팻킷획득(packet capture)을 효과적으로 할 수 있게 만든 공용 라이브러리이다. libpcap은 팻킷을 네트워크 인터페이스로부터 획득하여 사용자가 접근할 수 있는 사용자 메모리에 복사한 다음 사용자가 지정한 함수를 메모리에 복사된 팻킷에 수행하도록 한다. Thumbprint 데몬은 libpcap을 이용하여 팻킷을 획득한 다음 그 팻킷의 데이타를 기반으로 회선의 Thumbprint를 만든다. Thumbprint 데몬은 회선의 Thumbprint값을 기반으로 회선체인을 구성한 후 터미널 호스트를 찾아내 필터 모듈에게 알려준다. 방화벽의 필터 모듈이 Thumbprint 데몬에게 회선체인을 구성하도록 명령하도록 구현되었다.

### 3.6 필터모듈과 Thumbprint데몬의 통신

Thumbprint 데몬이나 로그 데몬은 사용자 영역의 응용프로그램이고 필터 모듈은 커널영역에 존재한다. 따라서 Thumbprint 데몬과 로그 데몬은 FreeBSD 커널과 통신을 할 수 있어야 한다. 개발된 통합 방화벽에서는 Thumbprint 데몬이 커널의 필터 모듈에게 명령을 보낼때는 시스템 호출(ipfw)을 이용하고 반대로 커널의 필터 모듈에게 명령을 내릴 경우에는 시그널(SIGIO)을 이용하여 명령을 전달했다.

## 4. 결론

본 논문에서는 통합 방화벽에서의 효율적인 터미널 호스트 추적 기능을 설계하고 FreeBSD를 기반으로 통합 방화벽을 구현하고 터미널 호스트 추적 기능의 구현 내용을 기술하였다 구현된 통합 방화벽은 인트라넷과 같은 가상 사설망(Virtual Private Network)이 필요한 환경에서 기존의 방화벽에 비해 더 적합하다.

## 참고 문헌


[1] Stuart Staniford-Che and L Todd Hebeilein, Holding Intruders Accountable on the Internet, *Proceeding of 1995 IEEE Symposium on Security and Privacy*, pp39-49, 1995

[2] Marcus J. Ranum, Thinking About Firewall, *Proceeding of Second International Conference on System and Network Security and Management(SANS-II)*, pp1-14, 1993

[3] L.T. Heberlein, K Levitt and B. Mukherjee, Internetwork Security Monitor. An Intrusion-Detection System for Large-Scale Networks, *Proceeding of 15th National Computer Security Conference*, pp262-271, 1992

[4] S Snapp, DIDS(Distributed Intrusion Detection System) - Motivation, Architecture, and An Early Prototype, *Proceeding of 14th National Computer Security Conference*. 1991

[5] H.T. Jung, Caller Identification System in the Internet Environment, *Proceeding of 4th USENIX Security Symposium*, 1993

[6] 이 승원,조 유근 이행적 신뢰 방지를 위한 방화벽시스템의 설계 및 구현, *1996*년도 한국정보과학회 가을 학술발표논문집 *Vol 23. No 2*, pp1435-1438,1996