Firewalling and Proxy Server HOWTO

David Rudder, *drig@execpc.com* <mailto:drig@execpc.com> v0.2, 17 July 1995

번역: 조용준, *sudoer@nownuri.net* <mailto:sudoer@nownuri.net>

이 문서는 PC에 기초한 리눅스에서 방화벽 설정의 기본을 가르치기 위해 설계 되었다. 또한 방화벽의 벽을 넘어 인터넷으로의 접근을 도와주는 프락시 서버 의 설치와 사용법도 커버하고 있다.

목 차

제 1 절 소개

방화벽은 인터넷 보안의 궁극적인 형태로서 최근 많은 명성을 얻어왔다. 명성 을 얻는 많은것들이 그러하듯 이 것 또한 오해를 일으켜 왔다. 이 하우투 문서는 방화벽이란 무엇인지, 그것을 어떻게 설정하는지, 또 프락시 서버란 무엇인지, 그것을 어떻게 설정하는지에 대한 기본지식을 잘 다룰 것이다. 또 보안 분야가 아닌 다른곳에서 이 기술을 사용한 응용 프로그램에 대해서도 다룰 것이다.

1.1 피드백

어떤 피드백도 환영한다. 매킨토시에 대한 정보가 부족하기 때문에 특히 매킨토 시 사용자로부터의 피드백을 바라고 있다. 이 문서에 정확하지 않은 곳이 있다 면 부디 나에게 알려주기 바란다!! 나도 사람이므로 실수를 할 수도 있다. 당신 이 실수를 발견했을 경우 그것들을 고치는 것이 내 최고의 즐거움이다. 나는 모 든 전자우편에 대해 답을 하려고 노력할 것이나, 나도 바쁘므로 혹시 답을 하지 못했다고 해서 욕을 하지는 말기 바란다. 나의 전자우편 주소는 *drig@execpc.com* `<mailto:drig@execpc.com>`이다.

1.2 선언

이 문서는 방화벽과 프락시 서버가 어떻게 작동하는가에 대한 소개자로서의 의미를 지닌다. 나는 보안의 전문가가 아니며 또 전문가인체 하지도 않는다. 단 지 다른 살마보다 컴퓨터를 좋아하고 좀 더 많은 책을 읽은 사람에 지나지 않는 다. 나는 이 문서에 근거한 행위에서 기인한 어떠한 손상에 대해서도 책임을 지지 않는다. 많은 사람들이 이 주제에 대해 친근함을 가지도록 도와주기 위해 이글을 쓰는 것 뿐이다. 또한 이 문서의 정확성에 내 인생을 묶어두고 싶지도 않다.

1.3 저작권

다른 언급이 없다면, 리눅스 하우투의 저작권는 존경받는 제작자에게 주어진 다. 리눅스 하우투문서는 모든 복사본에 이 저작권가 유지되는 한, 전체 또 는 부분적으로 복제와 배포가 허용되며, 물리적 또는 전자적인 어떤 수단으로 배포되어도 상관없다. 상업적인 배포도 허용·격려된다. 그러나 어떤 배포본에서 도 제작자는 밝혀져야 한다.

리눅스 문서와 관계된 모든 번역이라든지 원본을 변경하는 작업들 또는, 그것을 결합한 작업들은 저작권가 알려진 상태하에 이루어 져야한다. 다시 말해, 원 본 문서를 훼손해서는 안되며 그것의 배포시에 그것을 속이고 팔아서도 안된다. 이 룰의 예외는 특정한 조건내에서만 받아들여진다; 아래 주소의 리눅스 코디네 이터와 접촉을 시도하라.

정리하면, 가능한 많은 채널을 통해서 이 정보의 보급이 활발해지기를 우리는 희망한다. 그러나, 하우투 문서상에 저작권가 남아있기를 바라며, 하우투 문 서의 재배포에 관한 어떠한 계획이라도 우리에게 알려지기를 바란다. 질문이 있다면, David Rudder와 접촉을 하라. <*drig@execpc.com* `<mailto:drig@execpc.com>`>.

1.4 이 글을 쓴 이유

지난 몇 년간 comp.os.linux.*에서는 수많은 토론과 방화벽에 대한 도움요청이 계속되었다. 내게는 마치 아무도 대답을 해주려하지 않는 것 처럼 보였다. 그 이 유가 아무도 어떻게 해야하는지를 몰랐기 때문이라고 추측한다. 그러므로, 방화벽을 가지고 놀기위해 잠시 시간을 내고, 공부를 했다. 이 문서는 그런 모 든 요청들에 대한 응답이다.

1.5 해야 할 일

¥ 매킨토시 상에서 이것이 어떻게 되는가를 공부한다
¥ 다른종류의 windows tcp/ip 패키지를 공부한다
¥ 리눅스와 호환되는 훌륭한 UDP 프락시 서버를 찾는다

1.6 더 읽어야 할 것

¥ tis 방화벽 툴킷에 관한 문서
¥ net-2 하우투 문서
¥ ppp 하우투 문서
¥ 이서네트 하우투 문서
¥ multiple 이서네트 하우투 문서
¥ 리눅스에서의 네트워킹
¥ O'Reilly and Associates사가 출판한 TCP/IP Network Administrator's Guide

tis의 방화벽 툴킷에는 방화벽과 관련 구조물에 대한 가장 좋은 문서들이 모아져 있다. 방화벽 툴킷에 대한 더 많은 정보는 방화벽 소프트웨어에 대한 문서를 찾아 보아라.

제 2 절 방화벽의 이해

방화벽이라는 것은 자동차의 한 부분을 일컫기 위해 사용되는 용어이다. 자동 차에서 방화벽이라는 것은 객실로부터 엔진 블록을 분리하기 위한 물리적 장치 이다. 그것들은 자동차 폭발시에 승객을 보호하기 위한 의도로 사용되었다. 컴퓨터에서의 방화벽은 대중으로부터 사적인 네트워크를 보호하는 논리적 장치 이다.

방화벽의 작동법:

1. (리눅스 박스 처럼) 라우팅 능력을 지닌 컴퓨터를 갖는다.
2. 2개의 인터페이스를 집어넣는다. (시리얼 포트s, 이서네트, Token Ring 등등..)
3. ip forwarding을 중지한다.
4. 하나의 인터페이스로 인터넷에 접속한다.
5. 다른 인터페이스로 보호중인 네트워크에 접속한다.

이제, 하나의 컴퓨터를 공유하는 2개의 별개의 네트워크를 가지게 된다. 방화벽 컴퓨터는 , 앞으로는 그냥 "방화벽"이라 한다, 보호받는 네트워크와 인터넷 두 개 모두에 접근이 가능하다. 보호중인 네트워크는 인터넷에 접근이 안되며, 인터 넷 또한 보호중인 네트워크에 접근할 수 없다.

누군가가 보호중인 네트워크 내부로부터 인터넷으로 접근을 하려면, telnet으로 방화벽에 접속한 뒤 그곳으로부터 인터넷을 사용해야한다. 따라서, 보호받는 네 트워크에 도달하기 위해서는 반드시 방화벽을 거쳐야만 한다.

이것은 인터넷으로부터의 공격에 대해 뛰어난 보안기능을 제공해 준다. 보호받는 네트워크에 대해 협공을 가하려 한다면, 반드시 방화벽을 거쳐서 해 야하며, 이것은 한 스텝을 늦춰주고 공격을 더욱 어렵게 만든다. 또 보호받는 네트워크에 대해 폭탄 메일 또는 "인터넷 worm" 따위의 일반적인 방법의 공격 을 가한다면, 그들은 보호받는 네트워크에 도달하지 못 할 것이다. 이것은 뛰어 난 보안을 해준다.

2.1 방화벽의 결점

방화벽의 가장 큰 문제점은 방화벽내부로부터의 인터넷 접속을 방해한다는 것 이다. 기본적으로, 그들은 다이얼업 shell 계정를 경유한 인터넷의 사용을 줄 여버린다. 넷스케이프처럼 인터넷으로의 직접 접속을 요구하는 프로그램들은 방화벽의 뒤에서는 작동하지 않을 것이다. 이 문제에 대한 해결책은 프락시 서버를 갖는 것이다.

2.2 프락시 서버

프락시 서버는 방화벽을 넘어서 인터넷으로의 직접접속을 허용하는 구조물이 다. 그들은 서버에 있는 소켓을 열고 그 소켓을 통해 인터넷으로의 communication을 허용한다. 예를들면, 내 컴퓨터의 경우, drig(작가의 아이디 입 니다)는 보호받는 네트웍의 안쪽에 있으며, 넷스케이프를 사용하여 웹 검색을 하 고싶다면 나는 방화벽상에서 프락시 서버을 설정한다. 프락시 서버는 내 컴퓨 터로부터의 요청을 허락하도록 환경설정이 될것이며, 포트 1080에 접속하기 위 해, 포트 80을 사용할 것이다. 그리고 모든 요청들을 적당한 장소로 다시 보낼 것이다.

TIA나 TERM을 사용해본 사람이라면 이런 개념을 본적이 있을 것이다. 이들 두 프로그램을 사용하면 당신은 한 포트를 재전송 할 수 있다. 친구 하나가 누 구든 192.251.139.21 포트 4024를 사용하면 누구든 그의 웹 서버에 접속을 허 락하는 TIA 셋업을 하고 있다. 프락시 서버는 이렇게, 그러나 backward 작업으 로 작동한다. 누구라도 포트 80에 접속하기 위해서는 포트 1080을 (또는 자신이 설정해놓은) 사용해야 한다.

프락시 서버의 좋은점은 올바르게 환경설정이 됐다면, 완벽하게 안전하다는 것 이다. 프락시 서버는 누구라도 그들을 통해 안으로 들어오도록 하지 않을 것이 다.

제 3 절 Setting This All Up

3.1 하드웨어 요구사항

우리의 예를 들어보면, 컴퓨터는 486-dx66, 메모리는 8메가, 500M의 리눅스 파 티션, 14400 모뎀을 통한 인터넷 제공회사와의 ppp 연결 등이 있다. 이것 이 기본적인 리눅스 박스이다. 여기에 방화벽을 만들기 위해서 NE2000 이서네트 card를 하나 추가한다. 그리고 그것은 windows 3.1을 가동하는 3대의 pc와 SunOS 4.1을 돌리는 2대의 sun에 연결된다. 이런 설정은 선택한 것은 이것이 매우 일반적인 것들이고 두 개의 플랫폼이 모두 나에게 친숙한 것이기 때문이 다. 나는 지금 이야기하는 것들이 매킨토시에서도 가능할것이라고 상상하지만 Mac을 그리 자주 사용하지 않으므로 장담할 수는 없다.

3.2 software의 setting

이제 당신은 14.4 ppp 라인을 통해 네트워크에 접속된 리눅스 박스를 갖게 되었다. 그리고 리눅스와 다른 컴퓨터들로 연결된 이서네트 네트웍을 지녔다. 우선은 적당한 옵션을 주고 리눅스 커널을 다시 컴파일해야 한다. 여기서 나는 Kernel- 하우투, Ethernet-하우투, NET-2 하우투 등을 살펴보고, "make config"을 하였다.

1. 네트워킹 Sup포트을 켠다
2. TCP/IP 네트워킹을 켠다

3. IP Forwarding을 끈다
4. IP Firewalling을 켠다
5. IP 계정ing을 켠다. 보안 수단을 설정하고 있으므로 신중하게 보인다
6. 네트워킹 Device Support를 켠다
7. 우리는 PPP and 이서네트 support를 켰지만, 이것은 당신의 interface에 달 린 것이다.

그런 뒤 우리는 커널을 다시 컴파일하고, 설치한 후 리부트 했다. 부트 과정에서 인터페이스가 나타날 것이고, 우리는 잘 되었다. 정상적으로 되지 않는다면, 다른 하우투를 참고하며 될 때까지 해보아라.

3.3 네트웍 주소의 조정

이것은 매우 재미있는 부분이다. 우리는 접근를 하기위해 인터넷을 원하지는 않으므로, 실제 주소를 사용할 필요가 없다. 어떤 훌륭한 C class는 192.168.2.xxx라는 것을 사용하는데, 이것은 견본의 test doamin으로 설정된 것 이다. 이와같이, 아무도 그것을 사용하지 않으며, 외부로부터의 어떠한 요청과도 충돌을 일으키지 않을 것이다. 그러므로, 이 설정에서는, 오직 하나의 실제 IP 주소가 필요할 뿐이다. 다른 것은 하던 말던 자유이며, 네트웍에 전혀 영향 을 미치지 않는다.

ppp를 위해 사용되는 시리얼 포트에 실제 IP를 부여하라. 방화벽의 이서네트 카드에 192.168.2.1 을 부여하라. 그리고 보호받는 네트웍의 다른 모든 머신 에 그 도메인의 숫자들을 부여해라.

3.4 테스트하기

첫째로, 방화벽에서 인터넷에 ping을 해보아라. 나는 테스트 지점으로 nic.ddn.mil을 사용하곤 했다. ping은 여전히 훌륭한 테스트 방법이긴 하지만, 내가 기대했던만 큼 믿을만 하지는 않았다. 처음에 제대로 작동하지 않는다면, 당신의 lan에 연결 되지 않은 다른 장소로 시도해 보아라. 그래도 작동하지 않는다면, ppp 설정이 잘못된 것이다. net-2 하우투를 다시한번 읽고, 다시 시도해 보아라. 이제, 보호받는 네트웍 내의 다른 호스트간에 ping을 해봐라. 모든 컴퓨터들은 서로 ping이 가능해야 한다. 그렇지 않다면, net-2 하우투를 다시읽고 네트웍 상에서 좀더 일을 해봐라.

그러면 보호받는 네트웍 내의 모든 컴퓨더에서 방화벽로 ping이 가능하다. 그 렇지 않다면, 되돌아 와라. ppp 주소가 아니라 192.168.2.1로 ping가 가능해야 한다는 것을 기억하라. 그리고, 보호받는 네트웍의 내부로부터 방화벽의 ppp 주소로 ping을 시도 해 보라. 그것이 된다면, IP forwarding을 끄지 않은 것이며, 커널을 다시 컴파 일해야만 한다. 보호받는 네트웍에 192.168.2.1 이라는 도메인을 부여하는 것은 어떤 패킷이라도 이 네트웍으로 라우팅되지 않게 한다는 것을 의미한다. 그러 나 IP forwarding을 꺼놓는 것이 어쨌든 더욱 안전하다. 이것은 콘트롤을 ppp 제공회사가 아닌 당신의 손에 남겨놓는다.

마지막으로, 방화벽로부터 보호받는 네트웍내의 모든 머신에게 ping을 해보 라. 이때 까지 아무런 문제도 없어야 한다. 이제 방화벽의 기본적인 설정이 되 었다.

3.5 방화벽 보호

방화벽은 공격에 대해 많이 개방되어 있다면 아무런 효과가 없다. 우선 /etc/inetd.conf를 살펴보라. 이 파일이 "super 서버"라 불리는 그것이다. 그것 은 요청이 있을 때 한무더기의 서버 데몬을 가동한다.

```
예:
 + Telnet
 + Talk
 + FTP
 + Daytime
```

필요치 않은 것은 모두 꺼놓아라. 특히 netstat, systat, tftp, bootp, finger 등은 확실히 끄도록 해라. 또 telnet은 꺼놓고 rlogin 또는 그 반대만을 허용할수도 있다. 그러면, kill -HUP <pid>을 입력하여 해당 프로세스에 SIG-HUP 메시지 를 보낸다. 이것은 inetd로 하여금 config 파일을 다시읽고 재시작 하게 만들 것 이다. 방화벽의 포트 15로 텔넷하여 그것을 테스트해라. 포트 15는 netstat의 포트이다. netstat의 출력이 있다면, 제대로 재시작을 하지 못한 것이다.

제 4 절 Firewalling Software

4.1 사용가능한 패키지

완전한 방화벽은 리눅스 커널과 기본적인 네트워킹 패키지(inetd, telnetd와 telnet, ftpd와 ftp) 등을 제외한 어떠한 소프트웨어도 필요로하지 않는다.

그러나, 이러한 방화벽은 매우 제한적이며, 그다지 유용하지도 않다. 그래서, 소프트웨어 패키지들은 방화벽을 좀더 유용하게 하도록 만들어져야 한다. 내가 가장 상세히 시험해보고 싶은 것은 "socks"라 불리는 방화벽 패키지이다. 이것외에도 당신이 기억해 둘만한 프로그램들이 2개 더있다. 이제 그것들에 대 한 간단히 살펴보도록 하겠다.

4.2 The TIS Firewall 툴킷

TIS는 방화벽을 수월하게 만들도록 설계된 많은 프로그램의 모음을 내놓았다. 그 프로그램들은 기본적으로 socks 패키지와 같은역할을 하지만, 조금 다른 전 략을 가지고 설계되었다. socks는 모든 인터넷 처리를 담당하는 하나 의 프로그램을 가지고 있는 반면에, TIS는 하나의 프로그램을 방화벽을 사용해 야하는 각각의 유틸리티들로 나누어 놓았다.

이 두가지를 좀더 잘 비교하기위해 www과 telnet 접근를 예로 들어보자. socks를 가지고는, 하나의 config 파일과 하나의 데몬을 설정한다. 이 파일과 데몬을 통해, telnet과 www 두가지가 모두 가능하다. 물론 불가능하도록 해 놓지 않은 다른 서비스들도 마찬가지이다.

TIS 툴킷을 가지고는, 설정 파일 뿐만아니라 각각의 www과 telnet을 위하여 하나의 데몬을 설정해준다. 이것을 마친 뒤, 명시적인 set up이 있을 때 까지 다른 인터넷 접근는 금지된다. (talk 같은) 데몬이 제공되지 않았다면, "plug-in" 데몬을 사용할수 있지만, 다른 tool처럼 유연하지도 또 설정하기 쉽 지도 않다.

이것은 조금 달라보일 뿐이지만, 매우 커다란 차이가 된다. socks는 당신을 부주 의하게 만들 수 있다. 부주의하게 설정된 socks 서버는, 내부의 누군가가 최초 에 의도된 것 보다 많은 인터넷 접근를 얻을 수 있도록 만든다. TIS 툴킷 을 사용하면, 내부의 사람들은 시스템 관리자가 허용한 접근만 할 수 있다. socks는 설정하기 쉽고, 컴파일하기 쉬우며, 대단히 유연하다. 보호받는 네트웍 내부의 사람들을 통제하고 싶다면, TIS가 더 안전하다. 두가지 모두 외부로 부 터는 절대적인 보호를 제공한다.

4.3 TCP Wrapper

tcp wrapper는 방화벽 유틸리티는 아니지만, 상당히 비슷한 효과를 내준다. tcp wrapper를 사용하면 log를 기록할수 있을뿐만 아니라, 누가 당신의 머신과 서비스에 접근를 가질 것 인가를 제어할수 있다. 또한 기본적인 위조를 발견 해 준다.

tcp wrapper는 몇가지 이유로 이곳에서 더 이상 상세하게 다루어지지는 않는 다.

¥ tcp wrapper는 실제 방화벽은 아니다. ¥ 그것을 사용하려면 인터넷에 연결되어 있어야 하며, 이처럼 IP 주소를 가 져야만 한다. ¥ 오직 자신이 설치되어있는 머신만을 제어하며, 네트웍에서는 효과적이지 않 다. 방화벽은 모든 아키텍처의 머신을 보호할 수 있지만, TCP wrapper은 매킨토시와 MS 윈도에서는 작동하지 않는다.

4.4 Ipfw and Ipfw Admin

제 5 절 프락시 서버

5.1 프락시 서버의 설정

프락시 서버는 부가의 software를 필요로 한다. 그것들을 다음의 주소에서 얻 을 수 있다:

sunsite.unc.edu/pub/linux/system/network/misc/socks-linux-src.tgz `<ftp://sunsite.unc.edu/pub/linux/system/network/misc/socks-linux-src.tgz>` 이 디렉토리에는 "socks-conf"라 불리는 예제 config 파일도 들어있다. 이 파일을 당신 머신의 디렉토리에서 압축을 풀고, make하는 방법에 대한 지시를 따르라. 내가 그것을 make 할 때는 몇가지문제가 있었다. Makefile이 올바른지를 확인하 라. 어떤 것은 바르고, 어떤 것은 그렇지 않다.

프락시 서버가 /etc/inetd.conf에 추가되야 한다는 중요한 사실을 잊지말라. 당 신은 다음의 라인을 추가해야한다: socks stream tcp nowait nobody /usr/local/etc/sockd sockd

5.2 프락시 서버의 환경설정

socks 프로그램에는 2개의 분리된 설정 파일이 필요하다. 하나는 접근가 허용 되었음을 알리기 위한것이며, 다른 것은 요청된 것을 적당한 프락시 서버로 routing하기 위한 것이다. 접근 파일은 서버에 놓여 있어야 한다. routing 파일 은 모든 유닉스 머신에 놓여있어야 한다. 도스와 매킨토시는 짐작컨대, 자신의 routing이 있을 것이다.

5.2.1 the 접근 파일

socks 4.2 beta에서 접근 파일은 "sockd.conf"로 불린다. 그것은 permit과 deny 두 개의 라인을 포함해야 한다. 각각의 라인은 세 개의 엔트리를 가질 것이다:

```
+ The Identifier (permit/deny)
+ The IP 주소
+ The 주소 modifier
```

IP 주소는 전형적인 dot 표기법으로 이루어진 4바이트의 주소를 지닌다. 즉, 192.168.2.0 따위의 것이다. 주소 modifier 또한 4바이트의 전형적인 IP 주소이다. 그것은 마치 netmask처럼 동작한다. 이 숫자가 32bit이 되도록 상상해보라(1s 또는 0s). 그 bit이 1이면, 그것이 체크하는 응답 bit이 IP 주소so의 응답 bit와 들어맞아야 한다.

예를 들어, line이 이렇다면: permit 192.168.2.23 255.255.255.255 그것은 예를들어 192.168.2.3과 같이 192.168.2.23내의 모든 bit와 들어맞는 IP

주소만을 허용할 것이다. 이 line의 경우: permit 192.168.2.0 255.255.255.0 192.168.2.255를 통해 그룹 192.168.2.0내의 모든 번호들을 허용할 것이다. 당신은 이 라인을 가져서는 안된다: permit 192.168.2.0 0.0.0.0 왜냐하면 이것은 아무 주소나 상관없이 허용할 것이기 때문이다.

그러므로, 우선 원하는 모든 주소를 허용하고, 그런 뒤 나머지를 불허하라. domain 192.168.2.xxx 내의 모든사람을 허용하고 싶다면: permit 192.168.2.0 255.255.255.0 deny 0.0.0.0 0.0.0.0 이렇게 하면 잘 된다. deny line의 첫 번째 "0.0.0.0" 을 주목해라. "0.0.0.0"의 modifier를 가지면, IP 주소는 문제가 되지않는다. 입력하기 쉬우므로 모두 '0' 을 치는 것이 표준이다.

각각에 하나이상의 엔트리도 허용된다. 특정한 사용자만 접근을 허가 또는 불허 할수도 있다. 이것은 ident 인증을 통해서 이루어진다. 모든 시스템이 idnet를 지원하는 것은 아니므로, 트럼펫 윈삭을 포함하여, 그것을 여기서 상세히 이야기 하지는 않겠다. socks에 딸려있는 문서에 이 주제에 대해 자세히 나와있다.

5.2.2 The Routing 파일

routing 파일은 기분나쁘게도 "socks.conf"라고 불린다. 내가 "기분이 나쁘다"라고 이야기 한 것은 이것이 접근 파일의 이름과 매우 비슷하여 이 두 파일을 혼동하 기 쉽기 때문이다.

routing 파일은 socks 클라이언트에게 언제 socks를 사용할것인지, 또 언제 사용하지 않을 것인지를 알려주기위해 존재한다. 예를들어, 우리의 네트웍 내에서는, 192.168.2.3은 방화벽인 192.168.2.1과 통신하기위해 socks를 사용할 필요는 없 다. 그것은 이서네트을 경유한 직접 연결을 가지고 있기 때문이다. 그것 은 127.0.0.1의 loopback를 자동으로 정의한다. 물론 자신에게 이야기하기 위해서 도 socks 가 필요없다. 여기에는 3개의 엔트리가 있다.

```
+ deny
+ direct
+ sockd
```

deny는 socks에게 언제 요청을 거부할지를 알려준다. 이 엔트리 는 sockd.conf에 서와 같이 indentifier, 주소, modifier의 세 field 를 지닌다. 일반적으로, 이것들도 sockd.conf, 접근 파일에 의해 다뤄지므로, modifier field 는 0.0.0.0으로 설정된다. 만약 어느곳으로 부터의 호출도 불가능하게 하고싶다 면, 그것을 여기서 할 수 있다. direct 엔트리는 어느 주소가 socks 를 사용하 지 않을것인가를 알려준다. 이것들은 모두 프락시 서버 없이 접근 가능한 주소들이다. 이것도 identifier, 주소, modifier의 세 field 를 지닌다. 우리의 예를 들어 보면 이렇다.

direct 192.168.2.0 255.255.255.0 Thus going direct for any on our 보호받는 네트웍. sockd 엔트리는 컴퓨터에게 어느것이 socks 서버 데몬을 가지고 있는지 알 려준다. 명령의 형식은 이렇다: sockd @=<서버list> <IP 주소> <modifier> @= 엔트리를 주목해라. 이것은 당신이 프락시 서버 list의 주소를 설정하도 록 해준다. 그러나 잘 되지 않을 경우, 지나치게 많은 부담을 주거나 쓸데없이 남아돌게 할 수 있 다.

IP 주소와 modifier field는 다른 예에서와 마찬가지로 작동한다. 이것으로 어 느 주소가 어느 장소로 가도록 명시해준다.

5.2.3 방화변 뒤어서의 DNS

방화벽의 뒤에서 도메인 네임서버를 설정하는 것은 상대적으로 쉬운일이 다. 단순히 방화벽 머신에서 dns 만 설정해주면 된다. 그리고나서 방화벽 뒤의 각각의 머신이 이 dns를 사용하도록 설정한다.

5.3 Working With a Proxy Server

5.3.1 Unix

당신의 어플리케이션이 프락시 서버와 동작하게 하기위해선, 그것들이 "sockified"될 필요가 있다. 당신은 직접 연결하기 위한, 프락시 서버 를 경유한 통신을 위한, 서로다른 두 개의 telnet이 필요할 것이다.

socks 에는 어떻게 프로그램을 sockify 하는지에 대한 지시가 딸려온다. 어디론 가 직접 가기위해서 sockify된 버전을 사용한다면, socks는 자동적으로 direct version으로 전환해 줄 것이다. 이 때문에, 우리는 보호받는 네트웍 상의 모든 프로그램의 이름을 바꾸어주고 sokify된 프로그램들로 대체하길 원한다. "finger" 는 finger.orig"가 되고, "telnet"은 "telnet.orig"이 될 것이다. 당신은 이들 각각이 include/socks.h 파일을 경유한다는 것을 알려야한다. 어떤 프 로그램은 routing과 sockifying 자체를 다루기도 한다. 넷스케이프이 그런 프로그램 중 하나이다. 당신의 프락시하의 socks field에 있는 서버의 주소를 입력함 으로써, 넷스케이프 상에서 프락시 서버을 사용할 수있다. 각각의 응용 프로그램은 그것이 프락시 서버를 어떻게 다루느냐에 상관없이, 최소한 약간의 간섭이 필 요하다.

5.3.2 MS Windows with Trumpet Winsock

trumpet winsock은 프락시 서버의 능력을 갖춘채로 나온다. setup 메뉴에서 서버의 IP 주소와 직접 접근이 가능한 모든 컴퓨터의 주소를 입력하라. 그러면, trumpet winsock이 모든 dutgoing 패킷를 다룰 것이다.

5.4 UDP 패킷과 함께 작업하는 프락시 서버 얻기

socks 패키지는 오직 TCP 패킷와 동작하며, UDP와는 동작하지 않는다. 이 것이 socks의 유용함에 약간의 흠집을 남긴다. talk나 archie 같은 많은 유용한 프로그램들이 UDP를 이용한다. Tom Fitzgerald <*fitz@wang.com* `<mailto:fitz@wang.com>`>에 의해 만들 어진 UDPrelay라는 UDP 패킷을 위한 프락시 서버로 사용되기위해 설계된 패키지도 있다. 공교롭게도, 이 글을 쓰는 순간에 그것은 리눅스와 호환이 되지 않았다.

5.5 프락시 서버의 단점

프락시 서버는 무엇보다도 보안을 위한 장치이다. 제한된 IP 주소를 가지고 인터넷 접근를 증가시키기 위해 프락시를 사용하려면 많은 단점들이 있을 것 이다. 프락시 서버는 보호받는 네트웍 내부에서 바깥으로의 더욱 나아진 접근를 허용할 것이지만, 외부의 접근으로부터 그 내부는 완벽하게 지켜낼 것 이다. 이것은 talk나 archie 접속과 같은 어떠한 서버라든지, 내부 컴퓨터로 의 직접 메일 보내는 것이 불가능함을 의미한다. 이런 결점들은 작아보일지 모르나, 이 렇게 생각해보라:

¥ 방화벽로 보호중인 네트웍상에 작성하던 리포트를 남겨놓았다. 당신은 집 에 있으며, 그것(방화벽)을 넘어가고 싶다고 생각한다. 그러나 할 수 없다. 컴퓨터가 방화벽의 뒤쪽에 있다는 이유로 당신은 접근할 수 없다. 우선 당 신은 방화벽에 으로그인 하려고 노력할테지만, 모든 사람이 프락시 서버 접근를 가지고 있으므로, 아무도 당신을 위해 계정을 설정해 놓지 않았을 것이다.

¥ 당신의 여자친구가 다른 대학에 다닌다. 그녀에게 전자우편을 보내고 싶다. 당신은 사적으로 하고싶은 이야기가 있으며, 메일이 당신의 머신으로 직접 보내지기를 원한다. 당신은 systems administrator 완전히 믿고 있지만, 이것은 어디까지나 사적인 메일이다.

¥ UDP 패킷을 사용할수 없는 것이 프락시 서버의 가장 큰 단점이다. 곧 UDP도 지원되리라 생각한다.

FTP는 프락시 서버에 다른 문제를 일으킨다. ls 명령을 내릴 때, ftp 서버는 client 머신의 소켓을 열고 그곳을 통해 정보를 보낸다. 프락시 서버는 이것을 허용하지 않으므로, ftp는 면밀하게 동작하지 않는다. 또 프락시 서버는 느리게 동작한다. 막대한 오버헤드 때문에, 이것을 얻는 대개 의 다른 방법들은 훨씬 빠를 것이다.

기본적으로, IP 주소를 가지고 있고 보안에 대해 걱정하지 않는다면, 방화벽 과/또는 프락시 서버 를 사용하지 말아라. IP 주소를 갖지않고, 보안에 대해 걱정하지 않지만, term이나 slirp, TIA 등을 사용하여 조사를 하고싶을지 모른 다. term은 <>