Bandwidth Limiting HOWTO

Tomasz Chmielewski

tch@writemail.com

도운이

서정룡

s_ryong@hotmail.com

이 문서는 다운로드 대역폭 또는 들어오는 트래픽을 제한할 수 있는 리눅스 서버 설정 방법과 인터넷 링크의 효율적 사용 방법을 기술한다.

최종수정일 **: 2001**년 **6**월 **9**일

고친 과정

고침 0.22001-05-20고친이 tc

1. 소개

이 지침의 목적은 들어오는 트래픽을 제한하기 위한 손쉬운 해결책을 제공하는 것으로 따라서 랜 사용자들이 인터넷 링크의 모든 대역폭 (bandwidth) 을 소비하는 것을 예방할 수 있다.

이는 인터넷 링크가 느리거나 랜 사용자들이 다량의 mp3 파일들 및 최신 리눅스 배포판 이미지 *.iso 파일들을 다운로드할 때 유용하다.

1.1. 이 문서의 새로운 버전

이 문서의 최신 버전은 http://www.linuxdoc.org 웹 사이트에서 늘 볼 수 있다.

이 문서의 새로운 버전은 http://www.linuxdoc.org의 LDP 홈페이지를 포함해 다수의 리눅스 WWW 및 FTP 사이트에 또한 업로드될 것이다.

1.2. Disclaimer

Neither the author nor the distributors, or any other contributor of this HOWTO are in any way responsible for physical, financial, moral or any other type of damage incurred by following the suggestions in this text.

1.3. Copyright and License

This document is copyright 2001 by Tomasz Chmielewski, and is released under the terms of the GNU Free Documentation License, which is hereby incorporated by reference.

1.4. 피드백과 수정

이 문서에 대한 질문 또는 의견이 있다면 Tomasz Chmielewsk 에게 *tch@writemail.com* (mailto:tch@writemail.com) 로 자유롭게 메일을 보내주기 바란다. 어떠한 제안 또는 비평도 환영한다. 이 문서에서 잘못된 내용 또는 오타 (저자의 모국어가 영어가 아니기때문에 많이 발견할 수 있다) 를 발견하면 다음 버전에서 이를 수정할 수 있도록 저자에게 알려주길 바란다.

1.5. 감사의 글

저자는 이 HOWTO 문서를 SGML 포맷으로 변환하고 잘못된 내용을 바로잡는데 도움을 준 Ami M. Echeverri lula@pollywog.com 에 감사드리고 싶다. 또한 유용한 제안을 해준 Ryszard Prosowicz prosowicz@poczta.fm 에게도 또한 감사드린다.

2. 시작하기 전에

다음 상황이라고 가정해보자:

- 115,2 kbits/s ppp (modem) 인터넷 링크를 갖고 있다 (115,2/10 = 11,5 kbytes/s); 물론 그러한 유형의 연결은 있다! eth 연결의 경우는 115,2 를 8 로 나눌 것이다; ppp 연결의 경우는 시작/중지 비트때문에 (8 + 1 + 1 = 10) 10으로 나눈다.
- 어떤 LAN 스테이션을 갖고 있으며 사용자들이 언제든지 다량의 다운로드를 하고 있다.
- 얼마나 많은 다운로드를 하는 지에 상관없이 웹 페이지가 빨리 오픈되길 원한다.
- 인터넷 인터페이스는 **ppp0**이다.
- 랜 인터페이스는 **eth0**이다.
- 네트워크는 192.168.1.0/24 이다.

2.1. 무엇을 해야 하는가?

믿거나 말거나 들어오는 트래픽을 결정하는 것은 쉬운 일이며 라우팅 또는 큐잉 (queuing) 알고리듬에 관해 많은 책을 읽을 필요는 없다.

이를 작동시키기 위해서 적어도 스퀴드 프록시 (Squid proxy) 가 필요하며 이를 미세 조정하고 싶다면 ipchains 또는 iptables 및 CBQ 에 익숙해질 필요가 있을 것이다.

테스트를 위해 IPTraf 를 설치할 수 있다.

2.2. 어떻게 작동하는가?

스퀴드는 아마 리눅스에서 사용할 수 있는 가장 진보된 HTTP 프록시 서버로 다음 두가지 방식으로 대역폭을 보호할 수 있게 해준다:

- 첫번째는 프록시 서버의 주된 특징이다 -- 프록시 서버는 메모리 또는 디스크상에 다운로드 받은 웹 페이지, 그림과 다른 객체들을 계속 간직할 수 있으며 따라서 두 사람이 동일한 웹 페이지를 요청하고 있다면 인터넷이 아닌 로컬 프록시에서 다운로드되는 것이다.
- 통상적인 캐싱 (caching) 과는 별도로 스퀴드는 지연풀 (delay pool) 이라 불리는 특별한 특징을 갖고 있는데 이러한 지연풀 덕택에 URL 에 존재하는 소위 'magic words' 에 따라 합리적인 방식으로 인터넷 트래픽을 제한하는 것이 가능하다. 예를 들어 magic word 는 '.mp3', '.exe' or '.avi' 등일 수 있으며 .avi 와 같은 URL 의 모든 구별되는 부분도 magic word 로 정의될 수 있다.

지연풀 특징을 이용해 스퀴드에게 특정 속도 (이 문서에서는 대략 5kbytes/s) 로 이러한 유형의 파일들을 다운로드하라고 말할 수 있다. 랜 사용자들이 동시에 파일들을 다운로드한다면 이들은 전체적으로 5kbytes/s 속도로 다운로드될 것이며 웹 페이지, 이메일, 뉴스, irc 등에게 남아있는 대역폭을 남겨줄 것이다.

물론 인터넷이 웹 페이지 (http 또는 ftp) 를 통해 파일들을 다운로드하는데만 사용되지는 않는데 추후 냅스터, 리얼오디오 및 다른 가능한 것들에 대해 대역폭을 제한하는 방법을 다룰 것이다.

3. 필요한 소프트웨어 설치 및 설정하기

이 절에서는 대역폭 사용을 제한 및 검사할 수 있도록 필요한 소프트웨어의 설치 방법을 설명할 것이다.

3.1. 지연풀 특징을 갖는 스퀴드 설치하기

앞에 언급했듯이 스퀴드는 다운로드 대역폭을 제어할 수 있도록 하는 지연풀이라는 특징을 갖고 있다. 그러나 불행히도 대부분의 배포판에서 스퀴드는 이 특징없이 설치된다.

그래서 스퀴드가 이미 설치되어 있다면 이를 설치제거한 후 밑에 설명된 방식으로 지연풀 기능이 작동하도록 다시 설치해야 한다.

1. 스퀴드 프록시에서 최대 성능을 얻기 위해서는 /cache/ 라는 캐시 전용 파티션을 생성하는 것이 가장 좋은데 그 크기는 필요에 따라 대략 300 MB 정도여야 한다.

   독립된 파티션 생성 방법을 모른다면 주 파티션에 /cache/ 디렉토리를 생성할 수 있지만 스퀴드 성능은 약간 저하될 수 있다.

2. 안전한 'squid' 사용자를 추가한다:

   ```
   # useradd -d /cache/ -r -s /dev/null squid >/dev/null 2>&1
   ```

   루트를 포함해 어느 누구도 squid 로 로그인할 수 없다.

3. 스퀴드 소스를 http://www.squid-cache.org 에서 다운로드받는다.

   이 하우투 문서 작성할 때의 최신 버전은 Squid 2.4 stable 1 이었다:

   http://www.squid-cache.org/Versions/v2/2.4/squid-2.4.STABLE1-src.tar.gz
   (http://www.squid-cache.org/Versions/v2/2.4/squid-2.4.STABLE1-1rc.tar.gz)

4. `/var/tmp` 디렉토리에 압축을 푼다:

5. `# tar xzpf squid-2.4.STABLE1-src.tar.gz`

6. 스퀴드 컴파일 (모든 옵션은 동일 라인) 및 설치를 한다:

   ```
    # ./configure --prefix=/opt/squid --exec-prefix=/opt/squid
   --enable-delay-pools --enable-cache-digests --enable-poll
   --disable-ident-lookups --enable-truncate --enable-removal-policies
   ```

   ```
   # make all
   ```

   ```
   # make install
   ```

3.2. 지연풀 특징을 사용할 수 있도록 스퀴드 설정하기

1. squid.conf 파일을 설정해라 (/opt/squid/etc/ 디렉토리 (또는 다른 디렉토리) 에 위치할 수 있다):

   ```
   #squid.conf
   #이 파일의 각 옵션은 원본 squid.conf 파일 및
   #http://www.visolve.com/squidman/Configuration%20Guide.html 에
   #상세히 설명되어 있다

   #
   #스퀴드가 요청을 듣는 포트
   http_port 8080
   icp_port 3130
   ```

```
#cgi-bins 은 캐시되지 않을 것이다
acl QUERY urlpath_regex cgi-bin \?
no_cache deny QUERY
#스퀴드가 사용할 메모리. 그렇지만 스퀴드는 이 이상을 사용할 것이다
cache_mem 16 MB
#250 은 스퀴드가 디스크 공간 중 250 MB 사용할 것임을 의미한다
cache_dir ufs /cache 250 16 256
redirect_rewrites_host_header off
cache_replacement_policy GDSF
acl localnet src 192.168.1.0/255.255.255.0
acl localhost src 127.0.0.1/255.255.255.255
acl Safe_ports port 80 443 210 119 70 20 21 1025-65535
acl CONNECT method CONNECT
acl all src 0.0.0.0/0.0.0.0
http_access allow localnet
http_access allow localhost
http_access deny !Safe_ports
http_access deny CONNECT
http_access deny all
maximum_object_size 3000 KB
store_avg_object_size 50 KB

#랜 사용자들은 모두 리눅스에서 Mozilla 를 사용하는 것처럼 외부 서버들에 보여질 것
이다
anonymize_headers deny User-Agent
fake_user_agent Mozilla/5.0 (X11; U; Linux 2.4.4 i686)

#연결을 더욱 더 빠르게 하기 위해 밑의 라인과 비슷한 라인을 놓는다.
#가장 가까이 있는 서버를 변경하는 것을 잊지 마라.
#ping, traceroute 등을 비교 평가해라
#cache_peer w3cache.icm.edu.pl parent 8080 3130 no-digest default

#이는 캐시 관리자를 사용하길 원할 때 유용하다
#cachemgr.cgi 를 www 서버의 cgi-bin 디렉토리로 복사해라
cache_mgr your@email
cachemgr_passwd secret_password all

#스퀴드는 다음의 사용자 이름으로 작동한다
cache_effective_user squid
cache_effective_group squid

log_icp_queries off
buffered_logs on


#####DELAY POOLS
#이는 스퀴드를 이용해 들어오는 트래픽을 결정하는 가장 중요한 부분이다
#세부적인 설명은 squid.conf 파일 또는 http://www.squid-cache.org 의 문서들
을 보라
```

```
#로컬 네트워크에서 다운로드를 제한하길 원하지 않는다
acl magic_words1 url_regex -i 192.168

#다음 유형 파일들의 다운로드를 제한하길 원한다
#한 라인에 모두 열거한다
acl magic_words2 url_regex -
i ftp .exe .mp3 .vqf .tar.gz .gz .rpm .zip .rar .avi .mpeg .mpe .mpg .qt .ram .rm .iso .
#.html, .gif, .jpg 와 유사한 파일들은 일반적으로 대역폭을 많이 소비하지 않기 때
문에
#다운로드에 제한을 두지 않는다

#2개의 다른 지연풀을 갖는다
delay_pools 2

#첫번째 지연풀
#로컬 트래픽을 지연하길 원하지 않는다
#3가지 풀 클래스가 있다; 여기서는 단지 두번째 클래스만 다룰 것이다
delay_class 1 2

#-1/-1 은 제한이 없음을 의미한다
delay_parameters 1 -1/-1 -1/-1

#magic_words1: 192.168
delay_access 1 allow magic_words1

#두번째 지연풀
#magic_words2 에 언급된 파일들의 다운로드를 제한하길 원한다
delay_class 2 2

#숫자들은 바이트 단위인데 스퀴드가 시작/중지 비트를 고려하지 않음을 유념해야 한다
#5000/150000 은 전체 네트워크에 대한 값들이다
#5000/120000 단일 IP 대한 값들이다
#다운로드된 파일들이 150000 바이트를 초과 (두배 또는 세배 이상이라도) 한 후에는
#대략 5000 bytes/s 속도로 계속해서 다운로드할 것이다

delay_parameters 2 5000/150000 5000/120000
delay_access 2 allow magic_words2

#EOF
```

모든 것을 설정한 후 `/opt/squid` 와 `/cache` 디렉토리내의 모든 것들이 사용자 'squid' 에 속하는지 확인해야 한다.

```
# chown -R squid:squid /opt/squid/
```

```
# chown -R squid:squid /cache/
```

or

```
# chown -R squid.squid /opt/squid/
```

```
# chown -R squid.squid /cache/
```

이제 스쿼드를 작동할 준비가 되어 있는데 처음 이를 구동할 때는 캐시 디렉토리를 생성해야 한다:

```
# /opt/squid/usr/bin/squid -z
```

스쿼드를 작동해서 모든 것이 작동하는지 검사하는데 이를 위한 좋은 도구는 IPTraf 이다; 이는 http://freshmeat.net 에서 찾을 수 있다. 웹 브라우저에 적절한 프록시를 설정했는지 확인해라 (예, 192.168.1.1, port 8080):

```
# /opt/squid/usr/bin/squid
```

모든 것이 작동하고 있다면 `/opt/squid/usr/bin/squid` 라인을 초기화 스크립트의 마지막에 추가하는데 보통 이 스크립트는 `/etc/rc.d/rc.local` 일 것이다.

스쿼드에서 다른 유용한 옵션들로는 다음이 있을 수 있다:

 # /opt/squid/usr/bin/squid -k reconfigure (이는 squid.conf 파일을 변경한 경우 스쿼드를 재설정한다)

 # /opt/squid/usr/bin/squid -help :) 자명하다

`cachemgr.cgi` 을 www 웹 서버의 cgi-bin 디렉토리에 복사할 수 있다.

3.3. 남아있는 문제 해결

스쿼드를 설치해 지연풀을 사용하도록 설정하였다. 틀림없이 어느 누구도 특히 영리한 랜 사용자들은 제한받기를 원하지 않는데 그들은 자신들이 좋아하는 mp3 파일들을 약간 더 빠르게 다운로드하기 위해 제한을 피할려고 할 것이다 (따라서 난처하게 한다).

저자는 사용자들이 IRC, ICQ, 이메일 등을 사용할 수 있도록 랜에 IP-매스쿼레이드를 사용하고 있다고 가정한다. 좋다 그러나 랜 사용자들이 웹 페이지에 접속해 `ftp` 를 사용하기 위해서 지연풀 기능이 있는 스쿼드를 사용할 것인지를 확신해야 한다.

이러한 문제점들의 대부분은 리눅스 2.2.x 커널의 `ipchains` 또는 리눅스 2.4.x 커널의 `iptables` 을 사용해 해결할 수 있다.

3.3.1. 리눅스 2.2.x 커널 (ipchains)

어느 누구도 프록시 서버를 속여 이를 사용하려고 하지 않을 것임을 확신해야 하는데 공개 프록시는 보통 3128 과 8080 포트에서 작동한다:

```
/sbin/ipchains -A input -s ! 192.168.1.1 -d ! 192.168.1.1 3128 -p TCP -j REJECT
/sbin/ipchains -A input -s ! 192.168.1.1 -d ! 192.168.1.1 8080 -p TCP -j REJECT
```

또한 어느 누구도 프록시 서버를 속여 웹 페이지를 다운로드하기 위해 직접적으로 인터넷 (IP-masquerade) 에 연결하려고 하지 않을 것임을 확신해야 한다:

```
/sbin/ipchains -A input -s ! 192.168.1.1 -d ! 192.168.1.1 http -p TCP -j REDIRECT 8080
/sbin/ipchains -A input -s ! 192.168.1.1 -d ! 192.168.1.1 https -p TCP -j REDIRECT 8080
```

모든 것이 작동하고 있다면 이 라인들을 초기화 스크립트의 마지막에 추가하는데 보통 이 스크립트는 `/etc/rc.d/rc.local` 일 것이다.

랜 사용자들에게 스퀴드 사용을 강요하기 위해 `ftp` 트래픽 (포트 20 과 21) 을 막아야 한다고 생각할 수 있지만 적어도 다음 두가지 이유로 인해 좋은 생각은 아니다:

- 스퀴드는 실제 `ftp` 프록시가 아닌 `ftp` 를 지원하는 http 프록시로 `ftp` 에서 다운로드하거나 `ftp` 에 업로드할 수 있지만 원격 `ftp` 서버상의 파일들을 제거하거나 이름을 변경할 수 없다.

  포트 20 과 21 을 막는 경우 원격 `ftp` 서버상의 파일들을 제거하거나 이름을 변경할 수 없다.

- IE5.5 는 버그를 갖고 있다 (적어도 저자의 것은 그렇다) -- `ftp` 디렉토리를 검색하기 위해 프록시를 사용하지 않으며 대신 IP 매스커레이드를 통해 직접적으로 연결한다.

  포트 20 과 21 을 막는 경우 IE5.5 를 사용해 `ftp` 디렉토리를 브라우징할 수 없다.

이 때문에 다른 방법을 사용하여 과도한 `ftp` 다운로드를 막을 것인데 이는 4 장에서 다룰 것이다.

3.3.2. 리눅스 2.4.x 커널 (iptables)

어느 누구도 프록시 서버를 속여 이를 사용하려고 하지 않을 것임을 확신해야 하는데 공개 프록시는 보통 3128 과 8080 포트에서 작동한다:

/sbin/iptables -A FORWARD -s ! 192.168.1.1 -d ! 192.168.1.1 --dport 3128 -p TCP -j DROP

/sbin/iptables -A FORWARD -s ! 192.168.1.1 -d ! 192.168.1.1 --dport 8080 -p TCP -j DROP

또한 어느 누구도 프록시 서버를 속여 웹 페이지를 다운로드하기 위해 직접적으로 인터넷 (IP-masquerade) 에 연결하려고 하지 않을 것임을 확신해야 한다:

/sbin/iptables -A FORWARD -s ! 192.168.1.1 -d ! 192.168.1.1 --dport 80 -p TCP -j DROP

/sbin/iptables -A FORWARD -s ! 192.168.1.1 -d ! 192.168.1.1 --dport 443 -p TCP -j DROP

모든 것이 작동하고 있다면 이 라인들을 초기화 스크립트의 마지막에 추가하는데 보통 이 스크립트는 `/etc/rc.d/rc.local` 일 것이다.

랜 사용자들에게 스퀴드 사용을 강요하기 위해 `ftp` 트래픽 (포트 20 과 21) 을 막아야 한다고 생각할 수 있지만 적어도 다음 두가지 이유로 인해 좋은 생각은 아니다:

- 스퀴드는 실제 `ftp` 프록시가 아닌 `ftp` 를 지원하는 http 프록시로 `ftp` 에서 다운로드하거나 `ftp` 에 업로드할 수 있지만 원격 `ftp` 서버상의 파일들을 제거하거나 이름을 변경할 수 없다.

  포트 20 과 21 을 막는 경우 원격 `ftp` 서버상의 파일들을 제거하거나 이름을 변경할 수 없다.

- IE5.5 는 버그를 갖고 있다 (적어도 저자의 것은 그렇다) -- `ftp` 디렉토리를 검색하기 위해 프록시를 사용하지 않으며 대신 IP 매스쿼레이드를 통해 직접적으로 연결한다.

  포트 20 과 21 을 막는 경우 IE5.5 를 사용해 `ftp` 디렉토리를 브라우징할 수 없다.

이 때문에 다른 방법을 사용하여 과도한 `ftp` 다운로드를 막을 것인데 이는 4 장에서 다룰 것이다.

4. CBQ 를 이용해 대역폭을 소비하는 다른 프로토콜들 다루기

랜 사용자가 냅스터 또는 리얼오디오를 사용한다면 3 장에 따라 행한 작업을 손상시킬 수 있음을 유념해야 한다. 또한 3.3 절에서 `ftp` 트래픽을 막지 않았음을 유념해야 한다.

이는 다운로딩을 직접적이 아닌 간접적으로 제한함으로써 달성할 것인데 인터넷 디바이스와 랜 디바이스가 각각 `ppp0` 와 `eth0` 라면 `eth0` 인터페이스에서 나가는 트래픽을 제한함으로써 `ppp0` 로 들어오는 트래픽을 제한할 것이다.

이를 행하기 위해서는 CBQ 와 `cbq.init` 스크립트에 익숙해져야 할 것이다. 이는 ftp://ftp.equinox.gu.net/pub/linux/cbq/ 에서 얻을 수 있는데`cbq.init-v0.6.2` 를 다운로드받아 이를 `/etc/rc.d/` 디렉토리내에 놓아라.

`iproute2` 를 설치할 필요가 있는데 이는 모든 리눅스 배포판에 포함되어 있다.

`/etc/sysconfig/cbq/` 디렉토리를 살펴보면 `cbq.init` 과 함께 작동해야 하는 파일들의 예가 있는데 없다면 아마도 이를 커널내에 컴파일하지 않은 것이다.

4.1. FTP

3 장에서 업로드를 할 수 있도록 그리고 버그가 있는 IE5.5 사용자들이 `ftp` 디렉토리를 브라우징할 수 있도록 하기 위한 두가지 이유로 인해 ftp 를 막지 않았다. 결국 웹 브라우저와 `ftp` 프로그램들은 스퀴드 프록시를 통해 다운로드를 해야 하며 `ftp` 업로드/이름바꾸기/삭제는 IP 매스쿼레이드를 통해 행해져야 한다.

`/etc/sysconfig/cbq/` 디렉토리에 `cbq-10.ftp-network` 라 불리는 파일을 생성한다:

 # touch /etc/sysconfig/cbq/cbq-10.ftp-network

다음의 라인들을 파일에 추가한다:

```
DEVICE=eth0,10Mbit,1Mbit
RATE=10Kbit
WEIGHT=1Kbit
PRIO=5
RULE=:20,192.168.1.0/24
RULE=:21,192.168.1.0/24
```

`cbq.init-v0.6.2` 파일내에서 이러한 라인들의 설명을 볼 수 있다.

`/etc/rc.d/cbq.init-v.0.6.2` 스크립트를 구동할 때 이 스크립트는 `/etc/sysconfig/cbq/` 에 있는 설정을 읽어들일 것이다:

 # /etc/rc.d/cbq.init-v.0.6.2 start

모든 것이 작동하고 있다면 `/etc/rc.d/cbq.init-v.0.6.2 start` 를 초기화 스크립트에 추가하는데 보통 이는 `/etc/rc.d/rc.local` 일 것이다.

이 명령때문에 서버는 `ftp` 데이타를 10kbits/s 보다 빠른 `eth0` 를 통해 전송하지 않을 것이며 따라서 `ftp` 데이타를 10kbits/s 보다 빠르게 다운로드하지 않을 것이다. 랜 사용자들은 `ftp` 다운로드에는 스퀴드 프록시를 사용하는 것이 더욱 효율적임을 알게 될 것이며 또한 버그가 있는 IE5.5 를 사용해 `ftp` 디렉토리를 브라우징할 수 있을 것이다.

IE5.5 에는 또한 다른 버그가 있다 - `ftp` 디렉토리에서 파일에 마우스 오른쪽 버튼을 클릭해 '폴더로 복사하기' 를 선택할 때 파일은 프록시가 아닌 IP 매스쿼레이드를 통해 직접적으로 다운로드되어 지연풀 기능이 있는 스퀴드를 생략한다는 것이다.

4.2. 냅스터, 리얼오디오, 윈도우미디어 및 other issues

개념은 `ftp` 에서와 동일한데 단지 다른 포트를 추가해 다른 속도를 설정하는 것이다.

`/etc/sysconfig/cbq/` 디렉토리에 `cbq-50.napster-network` 을 생성한다:

 # touch /etc/sysconfig/cbq/cbq-50.napsterandlive

다음의 라인들을 파일에 추가한다:

```
DEVICE=eth0,10Mbit,1Mbit
RATE=50Kbit
WEIGHT=5Kbit
PRIO=5
#윈도우 미디어 플레이어
RULE=:1755,192.168.1.0/24
#리얼 플레이어는 TCP 포트 554 를 사용하며 UDP 의 경우는 다른 포트를 사용한다
RULE=:554,192.168.1.0/24
RULE=:7070,192.169.1.0/24
#냅스터는 포트 6699 와 6700 또는 아마 다른 포트를 사용한다?
RULE=:6699,192.168.1.0/24
RULE=:6700,192.168.1.0/24
#오디오갤럭시는 41000 근처의 포트를 사용하며 이 포트들은 많이 있다
#따라서 모든 포트 목록을 열거하지 않음을 명심해라
RULE=:41060,192.168.1.0/24
RULE=:41133,192.168.1.0/24
#어떤 영리한 사용자들은 냅스터, 오디오갤럭시 등을 사용할 때 SOCKS 서버에
#연결할 수 있는데 자신의 SOCKS 프록시를 작동할 때는 좋은 생각이다
RULE=:1080,192.168.1.0/24
#원하는 다른 포트를 추가해라; IPTraf 를 사용해 프로그램들이 사용하는 포트를 쉽게
#검사할 수 있다
#RULE=:port,192.168.1.0/24
```

5. 자주 묻는 질문들

5.1. 지연풀을 이용해 각 사용자별로 대역폭을 제한하는 것이 가능한가**?**

가능하다. `squid.conf` 원본 파일의 내부를 보고 http://www.squid-cache.org 에 있는 스퀴드 문서를 조사해라.

5.2. cbq.init 스크립트를 이용해 각 사용자별로 대역폭을 제한하는 것이 가능한가**?**

가능하다. 이 스크립트의 내부를 보라 많은 예들이 있다

5.3. CBQ 가 때때로 아무 이유없이 작동하지 않는다**.**

일반적으로 작동해야 한다. 그러나 냅스터 또는 오디오갤럭시가 사용하는 모든 포트를 막았다 하더라도 때때로 다량의 다운로드를 볼 수 있는데 다량의 다운로드를 위해 아마도 오픈된 포트가 분명히 있을 수 있다. 이 오픈된 포트를 찾기 위해 IPTraf 를 사용할 수 있지만 아마도 수천개의 그러한 포트가 있을 수 있기 때문에 이는 실제로 어려운 일일 수 있다. 이를 더욱 쉽게 하기 위해서는 자신의 SOCKS 프록시를 작동하는 것을 고려할 수 있다 - 냅스터, 오디오갤럭시 및 많은 프로그램들은 SOCKS 프록시를 사용할 수 있는데 따라서 가능한 수천개의 포트보다는 단지 한개의 포트 (1080) 만을 다루는 것이 더욱 쉽다. 트래픽을 위한 모든 포트를 닫고 25 와 119 (SMTP 와 POP3) 와 같은 포트 및 유용하다고 생각할 수 있는 다른 포트들을 열어두는 것을 잊지 마라. 이 하우투의 마지막에 NEC SOCKS 프록시에 대한 링크를 찾을 수 있는데 이를 사용하려면 이 링크 페이지에서 찾을 수 있는 패치를 적용하는 것을 잊지 마라.

5.4. 지연풀은 멍청하다**;** 네트워크를 혼자만 사용하고 있을 때 왜 최대한의 속도로 무언가를 다운로드할 수 없는가**?**

불행히도 이를 가능하게 하는 방법은 거의 없다.

가능한 방법은 **cron** 을 사용해서 예를 들어 새벽 1시에는 스퀴드가 지연풀을 사용하지 못하도록 재설정하고 다시 아침 7시 30분에 지연풀을 사용하도록 재설정하는 것이다.

5.5. CBQ 는 멍청하다**;** 네트워크를 혼자만 사용하고 있을 대 왜 최대한의 속도로 무언가를 다운로드 할 수 없는가**?**

운좋게도 이는 가능하다!

이는 두가지 방법으로 가능하다.

첫번째는 쉬운 것으로 스퀴드와 관련된 해결책과 유사한데 `/etc/sysconfig/cbq/` 내에 있는 CBQ config 파일들에 밑의 라인과 유사한 라인을 추가한다:

```
TIME=15:00-16:00;110Kbit/11Kbit
TIME=00:00-08:00;110Kbit/11Kbit
```

CBQ config 파일에 여러개의 TIME 매개변수를 가질 수 있다.

두번째 방법은 더욱 어렵지만 더욱 영리한 것이다. 이와 관련된 내용은 리눅스 2.4 Advanced Routing 하우투 문서에서 찾을 수 있다.

6. 그 외의 다른 사항들

6.1. 유용한 자료들

Squid Web Proxy Cache

http://www.squid-cache.org

Squid 2.4 Stable 1 Configuration manual

http://www.visolve.com/squidman/Configuration%20Guide.html

http://www.visolve.com/squidman/Delaypool%20parameters.htm

Squid FAQ

http://www.squid-cache.org/Doc/FAQ/FAQ-19.html#ss19.8

cbq-init script

ftp://ftp.equinox.gu.net/pub/linux/cbq/

Linux 2.4 Advanced Routing HOWTO

http://www.linuxdoc.org/HOWTO/Adv-Routing-HOWTO.html

Traffic control (in Polish)

http://ceti.pl/~kravietz/cbq/

Securing and Optimizing Linux Red Hat Edition - A Hands on Guide

http://www.linuxdoc.org/guides.html

IPTraf

http://cebu.mozcom.com/riker/iptraf/

IPCHAINS

http://www.linuxdoc.org/HOWTO/IPCHAINS-HOWTO.html

NEC SOCKS proxy server

http://www.socks.nec.com/cgi-bin/download.pl (http://www.socks.nec.com/c/s.dll/download.pl)