turbolinux
Powerful Thinking.

turbolinux
cluster server 6

사용자 안내서 및 참고용 설명서

TurboLinux Cluster Server 6 사용자 안내서 및 참고용 설명서

Version 6. 2000년 11월

한국터보리눅스(주)

서울특별시 영등포구 여의도동 60번지

대한생명 63빌딩 42층 (우)150-763

Tel. 02-3775-4072

Fax. 02-3775-4075

http://www.turbolinux.co.kr

목 차

서 문

먼저, 시중에 나와 있는 수많은 클러스터링 솔루션을 마다하고 저희 터보리눅스 클러스터 서버 6을 구입해주신 데 대해 감사 드립니다. 저희는 강력하고, 융통성이 뛰어나며, 사용하기 쉬운 제품을 만들기 위해 최선을 다하고 있습 니다. 또한, 터보리눅스 클러스터를 비롯한 모든 제품을 최고의 성능과 최저의 비용으로 공급하기 위해 노력하고 있 습니다.

본 설명서는 터보리눅스 클러스터 서버 6 설치, 구성, 사용 방법에 대해 설명하고 있습니다. 또한, 본 설명서를 클러스터 서버 6의 다른 고급 기능에 대한 참고서로 활용하셔도 좋습니다. 여러분은 본 설명서를 통해클러스터링의 개념과 클러스터를 만드는 목적에 관해 이해할 수 있을 것입니다.

터보리눅스에 관하여

터보리눅스는 이미 오래 전부터 환태평양 일대에서 리눅스의 선두 주자로서 세계를 주름잡아 왔습니다. 저희는 19 93년부터 리눅스와 함께 일해왔으며, 1997년에 처음으로 독립적인 배포판을 영어와 일어 버전으로 발표한 바 있 습니다. 현재는 영어, 불어, 독어, 이탈리아어, 스페인어, 포르투갈어, 중국어, 일본어, 러시아어 버전의 터보리눅스 워크스테이션과 서버 배포판을 제공하고 있습니다.
당사에 관한 최신 정보를 알고 싶으시면 웹 사이트 http://www.turbolinux.co.kr을 방문해 주시기 바랍니다.

터보리눅스는 엔터프라이즈급 리눅스 솔루션에서 주도적인 역할을 수행해 나가고 있습니다. 터보리눅스 클러스터 서버는 확장이 예상되는 소규모 기업에서부터 대기업에 이르기까지 다양한 환경에서 사용할 수 있는 여러 터보리눅스 계열 제품 중 하나입니다.

지금의 터보리눅스가 있기까지에는 오픈 소스 개발자이며 리눅스의 창시자인 Linus Torvalds의 공이 컸습니다. 이 모든 것을 가능하게 만들어준 Linus와 세계 수천만 명의 리눅스 개발자들에게 깊은 감사를 드립니다.

터보리눅스 클러스터 서버

터보리눅스 클러스터 서버 6은 이전 버전과는 달리 기존의 운영 체제 상에서 실행됩니다. 터보리눅스 클러스터 서버는 공식적으로 터보리눅스 서버 6.05, 6.1 K(한글판)와 레드햇 리눅스 제품을 지원합니다.

터보리눅스 서버나 레드햇 리눅스 시스템을 설치하지 않은 경우, 터보리눅스 클러스터 서버 6을 설치하려면 터보리 눅스 서버를 먼저 설치해야 합니다.

이전의 터보리눅스 서버 배포 버전이 설치되어 있다면, 터보리눅스 서버 6.0.5 버전으로 업그레이드하시기 바랍니다.

터보리눅스 클러스터 서버 6 제품에 포함되어 있는 내용물은 다음과 같습니다.

- 터보리눅스 클러스터 서버 6 Install CD(제품 등록 번호 포함)
- 터보리눅스 클러스터 서버 6 사용자 안내서 및 참고용 설명서
- 사용권 계약서

등록

클러스터 서버 제품을 사용하려면 먼저 등록을 해야 합니다. 제품과 함께 제공되는 CD 케이스를 보면 제품 등록 번호가 스티커가부착되어 있습니다. 이 등록 번호는 제품 등록 및 사용권 파일 발급에 사용됩니다. 등록하려면 웹 사이트 http://www.turbolinux.co.kr/register/을 방문하여 제품등록 번호와 몇 가지 필요한 정보를 입력해야 합니다. 등록 절차가 완료되면 사용권 파일이 제공되며 이 파일은 /etc/clusterserver/.licenses 디렉토리 안에 저장되어야 합니다.

기술 지원

터보리눅스는 웹 사이트에서 제품을 등록한 후 60일 동안 무료로 전자우편을 통한 설치 지원서비스를 제공합니다. 또한, 클러스터링 제품에 대해서는 추가 비용 없이 60일간의 전화 상담을 해드립니다.
이 기술 지원을 잘 활용하면 제품 설치 및 작동에 많은 도움이 될 것입니다.

그 밖의 추가 지원은 시간과 날짜별로 요금이 부과됩니다.
저희 기술 지원 웹 사이트 http://www.turbolinux.co.kr/support/를 방문하시면 중요한 정보를 얻으실 수 있습 니다.

여러분의 의견을 기다립니다!

여러분의 의견을 기다리고 있습니다. 설명서의 내용에 이상이 없도록 최선을 다했으나 혹시 여러분이 살펴보시는 동안 오류나 실수 또는 간과한 부분이 발견될 수도 있을 것입니다. 따라서 수정이 필요하다고 느끼시는 부분이나 제품에 대한 보다 자세한 설명이 필요하다고 생각되시는 부분이 있다면 주저말고 알려주십시오.

제품과 설명서에 관한 여러분의 의견은 기꺼이 반영하도록 하겠습니다. 전자우편을 보내실 곳은 feedback@turbolinux.co.kr입니다.

필요 조건

본 설명서는 여러분이 리눅스 운영 체제와 TCP/IP 네트워안에 관한 기본적인 내용을 알고 있음을 전제로 합니다. 본 설명서를 제대로 이해하려면 리눅스나 유닉스 명령어 라인을 사용하여 일반적인 시스템 관리 작업을 수행하는 데 익숙해야 하며 또한, 클러스터 내 시스템에 대한 root 액세스 권한을 갖고 있어야 하고, root 액세스 사용과 관련한 책임 범위를 잘 알고 있어야 합니다.

또한, IP 주소, 네트워크 인터페이스, 서브넷 마스크, 포트 번호, 데몬 등에 익숙해야 합니다.

표기 규칙

본 설명서는 다음과 같은 표기 규칙을 따릅니다.

- Monospace는 표시된 대로 정확히 입력해야 하는 유틸리티, 명령, 프로그램, 텍스트 예제를 나타냅니다.
- 파일명과 디렉토리 경로는 명조체로 표시됩니다

- *이탤릭체*는 CD, 책 제목, 또는 강조할 단어를 나타냅니다.
- 메뉴 항목과 버튼은 '작은 따옴표' 안에 표시합니다.
- 명령어 라인은 달러 기호($) 프롬프트로 시작되거나, root 액세스가 요구되는 경우 해시 기호(#)로 시작됩니다.

```
$ ls -lAtr pictures
# less /var/log/messages
```

제 1 장

소 개

이 장에서는 터보리눅스 클러스터 서버 6에 관한 기본적인 내용을 알아보고, 네트워크의 성능과 안정성, 그리고 본 제품이 제공하는 서비스를 효율적으로 이용하는 방법을 살펴보도록 하겠습니다.

먼저, 본 제품이 어떤 일을 하고 누구를 대상으로 만들어졌는지 소개하겠습니다. 그 다음, 독립형 시스템이나 다른 클러스터링 제품에 비해 터보리눅스 클러스터 서버를 사용함으로써 얻을 수는 있는 장점을 알아보겠습니다. 또한, 4.0 버전에 비해 6 버전에서 향상된 점이 무엇인지도 살펴보도록 하겠습니다. 마지막으로, 클러스터 서버 6을 실행하는 데 필요한 소프트웨어와 하드웨어 조건을 알아보겠습니다.

클러스터 서버란 무엇인가?

터보리눅스 클러스터 서버는 기존의 네트워크 자원을 이용하여 확장성 및 안정성이 뛰어난 서비스를 제공하는 데 사용되는 엔터프라이즈급 솔루션입니다. 본 제품을 사용하면 사실상 모든 TCP/IP 네트워크 서비스(웹, 메일, 뉴스, FTP 등)에 대한 서비스 수준을 크게 향상시킬 수 있습니다.

클러스터 서버는 필요에 따라 네트워크를 쉽게 확장할 수 있는 구조적 프레임워크를 제공합니다. 클러스터 서버는 네트워크 서비스에 대한 부하 조정과 fail-over를 구현합니다. 부하 조정은 여러 시스템에서 서비스를 실행하는 데 사용되는데, 클러스터는 이 부하 조정을 통해 해당 클러스터를 구성하는 서버 간 클라이언트 연결을 분배합니다.

Faibver는 단일 서버에서 서비스를 실행합니다. 그 서버가 실패할 경우, 클러스터 내의 또 다른 서버가 그 서비스의 역할을 대신 넘겨 받습니다.

클러스터 서버는 RAID와 비슷한 개념이라 할 수 있습니다. 다른 점이 있다면 RAID는 디스크 배열을, 클러스터 서버는 서버 배열 방식을 사용한다는 것입니다. 두 가지 모두 동일한 기능(향상된 속도, 안정성, 시스템자원 확보, 확장성)을 제공합니다.

클러스터 서버는 모든 작업을 단일의 대형 서버에 집중시키는 대신 여러 서버 간에 작업 부하를 분배합니다. 하지만, 그 클러스터를 액세스하는 클라이언트에게는 클러스터가 하나의 컴퓨터처럼 인식됩니다.

제품 용도

터보리눅스 클러스터 서버는 적절한 가격 수준에서 높은 가용성과 확장성을 구현해야 하는 중견 기업과 대기업을 대상으로 설계되었습니다.

인터넷 서비스 제공업자는 뛰어난 성능을 자랑하는 본 제품을 통해 고급 확장 기능과 업타임(작동가능시간)을 제공할 수 있습니다. 또한, 대기업은 내부적으로나 인터넷을 통해서 표준 서비스를 여러 클라이언트에게 동시에 제공할 수 있으며, 중견 기업의 경우에는 필요에 따라 기존의 컴퓨터 시스템을 용이하게 확장할 수 있습니다.

클러스터 서버를 구현하는 관리자는 리눅스나 유닉스에 익숙해야 하며 TCP/IP 네트워킹을 잘 이해하고 있어야 합니다. 클러스터링은 아주 간단한 개념이지만, 구현에 관한 세부 사항은 다소 복잡할 수도 있습니다. 문제를 해결하기 위해서는 TCP/IP에 대한 이해뿐 아니라 실질적인 경험도 요구됩니다.

터보리눅스 클러스터 서버는 Beowulf 클러스터가 아니며, 더군다나 Beowulf와 경쟁하기 위해 만들어진 것도 아닙니다.

본 제품은 CPU-바운드 프로세스를 클러스터링하는 것이 아니라 네트워크 기반의 서비스를 클러스터링합니다. 클러스터를 통해 집중적인 프로세싱 작업을 구현하려면, 터보리눅스 EnFuzion를 사용해 보십시오.
(EnFuzion 웹 사이트 http://www.turbolinux.co.kr/products/enf/ 참조)

왜 클러스터 서버를 사용해야 하는가?

클러스터 서버를 사용하면 합리적인 비용으로 기존의 시스템을 이용하여 확장 가능한 네트워크 서비스를 제공할 수 있습니다. 네트워크의 가용성을 최대한 높게 유지해야 한다면, 클러스터 서버가 바로 그 해답입니다. 단일 서버가 처리하는 경우보다 빈번하게 액세스가 필요한 서비스를 제공해야 할 때도, 클러스터 서버를 통해 가상 서버를 만들어 추가의 작업 부하를 처리할 수 있습니다.

시중에는 터보리눅스 클러스터 서버와 동일한 기능을 수행하는 하드웨어 솔루션이 많지만, 이처럼 폐쇄적인 제품들은 가격이 매우 비싸고 융통성이 떨어지는 경향이 있습니다.

하지만, 리눅스 기반의 시스템을 사용하면 클러스터를 보다 정교하게 제어할 수 있을 뿐만 아니라 클러스터 관리자에서 다른 서비스를 실행할 수도 있으며, 클러스터 관리자가 클러스터 노드의 역할까지 수행하게 할 수도 있습니다.

클러스터 관리자는 트래픽 관리자 자체의 시스템자원 확보를 허용하기 때문에, 다른 하드웨어 기반의 솔루션처럼 단일 실패 지점이 생기지 않습니다.

클러스터 서버는 고성능 솔루션입니다. 트래픽은 커널 내 아주 낮은 레벨에서 관리되며, 들어오는 트래픽은 모두 트래픽 관리자를 통과하지만, 나가는 트래픽은 클러스터 노트로부터 클라이언트로 직접 전달됩니다. 대부분의 TCP/IP 서비스의 경우, 요청에 비해 응답 회수가 더 많으므로 상기 특징은 최적화에 있어서 매우 중요한 비중을 차지합니다.

클러스터 서버는 트래픽을 전달하는 것 이외에도 네트워크의 상태와 가용성을 모니터링합니다. 클러스터 서버는 모든 서버 노드를 끊임없이 샘플링하면서 애플리케이션이 제대로 실행되고 있는지 검사하는데, 이 작업은 직관적인 애플리케이션 폴링 에이전트를 통해 이루어집니다.

또한, 각각의 백업 트래픽 관리자는 마스터 트래픽 관리자를 반복적으로 조회하여 클러스터 자체의 작동 상태를 검사합니다.

어떤 서비스를 클러스터링 하는가?

클러스터 서버 제품을 사용하면 많은 네트워크 서비스를 클러스터링할 수 있습니다. 가장 중요한 요구 사항은 서비스가 하나 이상의 컴퓨터에서 동시에 실행될 수 있어야 한다는 것입니다. 이것은 거의 모든 TCP/IP 서비스에 해당됩니다. 일반적으로 클러스터 서버에 사용되는 서비스는 다음과 같습니다.

- 웹사이트 (HTTP, HTTPS)
- FTP
- 메일 (SMTP, POP3, and IMAP)
- 뉴스 (NNTP)
- DNS
- LDAP

터보리눅스 클러스터 서버는 쓰기 위주의 데이터베이스 서버를 클러스터링할 수 없습니다. 클러스터 노드 간에는 기본적으로 제공되는 잠금 메커니즘이 없기 때문에, 둘 이상의 클러스터 노드가 같은 데이터베이스에 기록하려고 할 경우, 데이터가 손상될 수 있습니다.

데이터베이스를 클러스터링할 필요가 있을 때는 몇 가지 옵션을 사용해야 합니다. 예컨대, 클러스터링을 사용하여 데이터베이스를 읽고, 다른 시스템을 이용하여 데이터베이스에 쓰기 작업을 수행할 경우 아무런 문제도 발생하지 않습니다. 또 다른 방법은 이중(2-tier) 모델을 사용하고, 클러스터 내 웹 서버가 클러스터 뒤의 데이터베이스를 액세스하게 하는 것입니다.

본 개정판의 특징

이번에 발표된 터보리눅스 클러스터 서버 버전은 이전 버전과 많은 차이가 있습니다. 새로 추가된 기능도 많고, 시스템의 구조도 바뀌었을 뿐만 아니라 심지어는 이름까지도 터보클러스터 서버에서 터보리눅스 클러스터 서버로 바뀌었습니다. 본 절에서는 이전 버전(4.0)과 최신 버전 간의 두드러진 차이점을 살펴보도록 하겠습니다.

변경된 주요 사항은 다음과 같습니다.

- 운영 체제로부터 분리
- 일부 요소의 이름이 새로 바뀜
- 기술적 향상
- 보안 시스템 강화
- 클러스터 관리 콘솔
- 가용성 향상
- 라이센싱 변경

독립적 제품

본 제품의 이전 버전은 리눅스 배포판으로 통합되어 있었으나 최신 버전은 운영 체제로부터 분리되어 독립적인 제품으로 패키지화되었습니다. 따라서, 이 버전을 사용하려면 리눅스 배포판이 이미 설치되어 있어야 합니다. 리눅스 배포판으로는 터보리눅스 서버 6.0.5 이상의 버전을 사용하는 것이 바람직하지만, 레드햇 리눅스6.2를 사용해도 무관합니다.

리눅스 배포판과 독립적으로 배포되는 클러스터링 제품을 사용하면 여러 가지 장점을 얻을 수 있습니다. 먼저, 운영 체제나 클러스터 서버 제품을 개별적으로 업그레이드하기가 더 쉽습니다. 또한, 발생하는 문제도 클러스터링 소프트웨어와 관련된 문제나 기존의 운영 체제와 관련된 문제로 구분되기 때문에 문제를 간단하게 해결할 수 있습니다.

마지막으로, 다른 리눅스 배포판에도 제품을 설치할 수 있다는 융통성이 있습니다. 특정 리눅스 버전에서만 실행되는 소프트웨어 패키지가 있다면, 이제는 그 시스템에서도 클러스터 서버를 사용할 수 있습니다.

새로운 설치자

본 제품 이전 버전은 자사의 리눅스 배포판에서만 사용할 수 있었기 때문에 운영 체제와 함께 설치해야 했습니다. 하지만, 새로운 독립형 버전은 새로운 설치 툴을 통해 다양한 요소를 설치할 수 있습니다. 설치 프로그램은 사용하기 쉬운 메뉴 중심의 프로그램으로서 사용자에게 설치 절차를 안내합니다. 설치 프로그램에 관한 내용은 3장에서 자세히 설명하겠습니다.

터보리눅스 또는 레드햇에서 실행

본 제품은 더 이상 특정 운영 체제에 국한되지 않고, 레드햇 리눅스와 터보리눅스 서버에서도 실행할 수 있게 설계되었습니다. 본 제품을 사용하려면 터보리눅스 서버 6.0.5 이후 버전이나 레드햇 리눅스 6.2를 실행해야 합니다. 현재로서는 다른 리눅스 배포판은 지원되지 않습니다.

새로운 명칭

본 제품 명칭이 터보클러스터 서버에서 터보리눅스 클러스터 서버로 바뀌었습니다. 이것은 본 제품이 운영체제로부터 분리된 제품이라는 점을 나타내기 위한 것이기도 합니다. 이와 같은 명칭이 변경됨에 따라, 여러가지 구성요소의 이름도 버전 4.0과 다르게 바뀌었습니다. 다음은 변경된 명칭을 정리한 것입니다.

표 1.1 변경된 구성요소 명칭

터보클러스터 서버 4.0	터보리눅스 클러스터 서버 6
turboclusterd	clusterserverd
turbocluster_sync	tlcs_config_sync
tl_sync	tlcs_content_sync
/etc/turbocluster.conf	/etc/clusterserver/clusterserver.conf
/var/log/turboclusterd.log	/var/log/clusterserverd.log
TCSWAT	CMC (replacement for TCSWAT)

기술적인 향상

6 버전은 이전 버전에 비해 여러 가지 면에서 기술적으로 향상되었습니다. 그 내용은 다음과 같습니다.

- NAT 포워딩방법
- Failover 지원
- 서버 및 애플리케이션 검사 간격을 다르게 지정할 수 있는 기능
- 보다 다양한 ASA(Application Stability Agents) 지원NAT 지원

NAT 지원

클러스터 서버 6은 이전에 지원되던 포워딩 방법 이외에도 NAT(Network Address Translation)을 지원합니다. 따라서, 현재 선택할 수 있는 포워딩 방법은 직접 포워딩, 터널링, 그리고 NAT입니다.

NAT는 일반적으로 개인용 네트워크를 인터넷에 연결된 방화벽 뒤에 숨기는 데 사용되는 기술입니다. NAT를 사용하면 개인용 네트워크 내/외로 전달되는 트래픽이 마치 하나의 시스템으로부터 나오는 것처럼 보입니다.

또한, NAT를 사용하면 클러스터 노드 자체를 특별히 변경할 필요가 없고 기본 게이트웨이만 설정하면 되기 때문에 구성이 단순해집니다.

또한, 외부로부터 집적 클러스터 노드를 액세스할 수 없기 때문에 보안이 강화됩니다. NAT의 단점은 나가는 트래픽이 반드시 NAT를 통과해야 하기 때문에 성능이 약간 떨어진다는 것입니다. 클러스터 서버에 사용되는 NAT 시스템은 NAT를 규정하는 인터넷 표준인 RFC 1631에 따라 구현되었습니다.

스테이트리스(stateless) Fail-over 지원

터보리눅스 클러스터 서버에서는 부하 조정뿐 아니라 fail-over 서비스도 구현할 수 있습니다. 부하 조정은 두 개 이상의 시스템이 동일한 서비스를 동시에 제공할 수 있게 하는 반면, fail-over는 한 번에 하나의 서버만 서비스를 제공하게 합니다. 그리고 그 서버가 다운될 경우에만 다른 서버 중 하나가 네트워크 트래픽을 전달합니다.

지연 설정 구분

클러스터 서버는 클러스터 노드에 대해 두 가지 검사를 실시합니다. 먼저, 서버가 네트워크 요청에 응답하는지, 그리고 ASA(Application Stability Agent)를 실행하여 요구되는 특정 서비스가 응답하고 있는지 검사합니다. 이전의 버전에서는, 이 두 가지 검사에 대한 간격이 공통으로 적용되었으나, 6 버전에서는 각 검사별로 서로 다른 간격을 지정할 수 있습니다.

다양한 애플리케이션 안정용 에이전트(ASA) 지원

최신 버전에서는 더 많은 애플리케이션 안정용 에이전트 (ASA)가 포함되어 있습니다. 그 중에는 Oracle과 DB2 같은 다양한 엔터프라이즈급 데이터베이스로 연결되는 에이전트도 있습니다. 지원되는 전체 ASA 목록이 아래에 나와 있습니다.

- DB2Agent
- dnsAgent
- ftpAgent
- genericAgent
- httpAgent
- httpsAgent
- http10Agent
- imapAgent
- nntpAgent
- oracleAgent
- popAgent
- smtpAgent

보안 강화

최신 버전에서는 시스템의 무결성을 보장하고 클러스터에 대한 액세스를 제한하기 위한 여러 가지 보안 기능을 추가되었습니다. 그 중에는 시스템에 대한 액세스를 제한하는 기능과 SSH(Secure Shell)를 통해 클러스터 노드간 데이터 전송 기능이 포함되어 있습니다. 또한, CMC 프로그램은 SSL로 암호화된 HTTPS를 사용하지만, CMC 대신 사용되던 TCSWAT 프로그램은 암호화되지 않은 정규 HTTP를 사용했었습니다.

보안 설정

6 버전에서는 시스템이 클러스터의 원격 구성 기능에 대한 액세스를 허용하거나 거부하도록 지정할 수 있습니다. 이 기능은 /etc/hosts.allow과 /etc/hosts.deny 파일에 구성되어 있는 TCP 랩퍼 설정과 비슷합니다.

여러분은 별도의 호스트를 지정할 수도 있고 IP 주소 영역을 지정할 수도 있습니다. 이 내용은 구성에 관한 장에서 더 자세히 다루도록 하겠습니다.

동기화 툴

동기화 툴은 SSH를 사용하여 데이터를 안전하게 전송하는데 사용됩니다. 여기에는 구성 정보와 내용이 모두 해당됩니다.

F-Secure SSH 버전 1.3.7은 클러스터 서버 패키지와 함께 설치됩니다. 다른 SSH 버전이 시스템에 설치되어 있는 경우, 완벽한 호환성을 보장하려면 이를 삭제해야 합니다.

클러스터 관리 콘솔

새로 개발된 웹 기반의 관리 시스템을 클러스터 관리 콘솔 또는 CMC라고 합니다. 이 툴은 이전 버전의 TCSWAT 프로그램 대신 사용됩니다. 이 툴은 더 많은 기능을 제공하고 클러스터에 관한 더 자세한 내용을 제공합니다.

CMC는 클러스터의 현재 성능을 감시하는데 사용되며, 클러스터 설정을 동적으로 변경하는 데도 사용할 수 있습니다. CMC의 가장 강력한 요소 중 하나는 트래픽 감시자(Traffic Monitor)입니다. 트래픽 감시자는 클러스터의 성능에 관한 실시간 그래프를 생성합니다.

CMC 안에서 로그 파일을 디스플레이할 수도 있으며, 설명서 페이지 같은 온라인 도큐멘테이션을 볼 수도 있습니다. 또한, CMC 웹 페이지에서 클러스터 서버 데몬을 종료하거나 재시작할 수도 있습니다.

CMC는 7장에서 더 자세히 다루도록 하겠습니다.

유용성 향상

최신 버전에서는 유용성을 향상시키기 위해 변경된 내용은 다음과 같습니다.

- 구성 툴 변경
- 단순화된 구성 파일 구문
- 로그 파일의 포인팅 향상

구성 툴

구성 툴 사용이 더욱 간편해졌습니다. 구성 툴에서 사용되는 일부 용어와 메뉴도 단순해졌으며, 툴 자체도 사용자에게 더욱 친숙하게 변경되었습니다. 또한, 웹 기반의 클러스터 관리 콘솔이 추가됨으로써 소프트웨어의 유용성이 향상되었습니다.

구성 파일 형식

구성 파일에 사용되는 다양한 옵션의 구문이 더욱 명확해졌으며, 견본 구성 파일이 제공됩니다. 구성 파일의 형식이 상당히 간단해지긴 했지만 가능하면 구성 툴을 사용해야 합니다.

구성 파일의 형식은 4.0 버전에서 사용되던 형식과는 달라졌으나 기존의 파일을 6에서 사용할 수 있도록 간단히 변경할 수도 있습니다.

간단하게 /etc/clusterserver/clusterserver.conf 파일에서 포트 번호를 삭제하고 'AddServer' 라인을 편집하면 됩니다. 구성 파일 형식에 관한 자세한 내용은 6장을 참조하시기 바랍니다.

오류 로그

오류 로그 파일의 형식은 더 읽기 쉽게 변경되었습니다. 로그 파일의 많은 메시지가 더 명확해졌으며, 가능하면 80 열을 넘지 않도록 줄였습니다. 이것은 클러스터와 관련된 문제를 해결할 때 도움이 됩니다.

사용권리

6 버전은 프로그램 사용을 허가하는 사용권 활성화 코드가 포함되어 있습니다. 이로 인해 가격 구조에 더 많은 융통성이 생기고 고객에게 특정 기간이 지나면 만기가 되는 평가 복사본을 제공할 수 있습니다.

이와 같은 활성화 코드 시스템에서는, 데모 버전을 사용하다가 정식 사용권을 구입하고자 할 때, 제품을 다시 설치할 필요 없이 새로운 사용권 파일을 서버로 복사하기만 하면 됩니다.

사용권 파일은 누적됩니다. 즉, 2 ATM과 2 노드에 대한 사용권을 구입하고, 2 ATM과 10 노드에 대한 또 다른 사용권을 구입한다면, 최대 4개의 ATM과 12개의 노드를 사용할 수 있습니다.

하지만, ATM과 클러스터 노드의 역할을 동시에 수행하는 시스템의 경우 ATM 사용권과 노드 사용권 모두 필요합니다.

등록

제품을 사용하려면 우선 등록을 해야 합니다. 등록하려면 먼저 등록 웹사이트 http://www.turbolinux.co.kr/register/을 방문하여 제품의 등록 번호와 필요한 정보를 입력하십시오. 등록 절차가 완료되면 사용권 파일이 제공되며, 이 파일은 /etc/clusterserver/.licenses 디렉토리 안에 저장됩니다.

요구 사항

터보리눅스 클러스터 서버는 다양한 컴퓨터 자원을 조합하는 데 사용됩니다. 각 컴퓨터에 대한 요구 사항은 클러스터 내의 그 컴퓨터의 역할에 따라 다릅니다.

두 가지 주요 기능은 고급 트래픽 관리자(ATM)와 클러스터 노드라 할 수 있는데, 클러스터 노드는 단순히 네트워크 서비스를 제공하는 시스템이고, 트래픽 관리자는 들어오는 모든 패킷을 받아 그것을 클러스터 노드로 포워딩 하는 컴퓨터입니다.

백업 트래픽 관리자는 주 ATM이 실패할 경우에만 작동됩니다. 시스템은 트래픽 관리자와 클러스터 노드의 역할을 동시에 수행하도록 구성 가능합니다.

소프트웨어

모든 트래픽 관리자는 터보리눅스 클러스터 서버를 설치한 후 실행되어야 합니다. 트래픽 관리자가 아닌 클러스터 노드는 클러스터 서버 제품을 실행할 필요가 없으며, 어떠한 운영 체제(리눅스, 유닉스, Windows NT, Windows 2000등)든 실행이 가능합니다. 하지만, 모든 시스템에서 동일한 운영 체제와 클러스터 서버 소프트웨어가 실행되고 있다면 클러스터 관리 작업은 더욱 간단해질 것입니다.

클러스터 서버를 실행하려면 터보리눅스 서버나 레드햇리눅스를 실행하는 리눅스 서버가 있어야 합니다. (이전의 클러스터 서버 버전은 터보리눅스 서버로 통합되어 있지만, 최신 버전에서는 클러스터 서버를 설치하기 전에 터보리눅스 서버를 설치해야 합니다.)

터보리눅스 서버를 실행할 경우, 6.0.5 이후 버전을 설치해야 하며, 레드햇시스템의 경우, 6.2 버전을 실행해야 합니다.

본 제품은 다른 리눅스 시스템에서도 실행되지만, 품질을 보장하기 위해서 여기에서 설명한 배포 버전만 지원합니다. 터보리눅스 서버 6.05 혹은 6.1K 는 구입을 하셔야 합니다.

더 오래된 터보리눅스 서버 버전이나 터보클러스터 서버 4.0을 실행하고 있다면, 제공되는 소프트웨어를 사용하여 운영 체제를 업그레이드하시기 바랍니다.

클러스터 서버 관리 소프트웨어 이외에, 클러스터링될 서비스를 제공하는 소프트웨어가 있어야 합니다.

예를 들어, 웹 서버 클러스터를 만들 경우, 클러스터 안의 각 노드는 개별적인 웹 서버를 실행해야 합니다. 이 소프트웨어는 클러스터 서버 제품에 포함되어 있지 않지만, 대부분의 운영 체제가 다양한 네트워크 서비스를 제공합니다. 예를 들어, 터보리눅스 서버를 비롯한 거의 모든 리눅스 배포판은 아파치 웹 서버를 제공합니다.

하드웨어

클러스터 서버는 일반적인 수준의 하드웨어(가령, 펜티엄 100에 32MB RAM)에서 실행할 수도 있지만, 뛰어난 성능을 제공하도록 설계되었습니다.

따라서, 이와 같은 고성능 요구 사항을 만족시키는 하드웨어를 사용하는 것이 바람직합니다. 트래픽 관리자에 대한 하드웨어 스펙은 네트워크 라우터에 대한 스펙과 비슷합니다. 안정적이고 효율적인 하드웨어를 선택하도록 하십시오.

여러분이 중요하게 고려할 요소는 네트워크 인터페이스 속도, 메모리, 그리고 CPU 속도입니다. 현재, 적절한 수준의 하드웨어는 100Mbs 이더넷 카드, 256MB의 RAM, 그리고 700MHz 프로세서 이상이어야 합니다 (터보리눅스 클러스터 서버는 인텔 호환 구조에서만 사용할 수 있습니다.)

컴퓨터에서 다른 서비스를 실행할 것이 아니라면 디스크 용량은 별로 중요하지 않습니다. 컴퓨터에서 사용될 다른 소프트웨어는 없는지 확인해 보십시오. 클러스터 서버 소프트웨어 자체는 약 40 MB의 디스크 용량을 차지하며, 로그 파일과 기타 관리 작업을 위한 추가 공간이 필요합니다.

Advanced Traffic Manager가 NAT 클러스터 노드를 지원하기 위해서는, 두 개의 네트워크 카드를 갖고 있어야 합니다. 하나는 들어오는 클라이언트 요청을 받아들이는데 사용되고, 다른 하나는 NAT 개인용 네트워크에 연결하는 데 사용됩니다.

클러스터 노드에 대한 하드웨어 요구 사항은 시스템이 독립형으로 실행될 때와 똑같습니다. 중요한 것은 그 노드에서 어떤 서비스가 실행되느냐는 것입니다. 노드에서 실행될 운영 체제와 애플리케이션에 대한 하드웨어 권장 사항 이외에는 별도의 요구 사항은 없습니다.

최고의 업타임(작동가능시간)을 제공하기 위해서는 하드웨어 시스템자원 확보 기능을 최대한 활용하시기 바랍니다. 정전 시에도 클러스터가 계속 실행되게 하려면 UPS를 사용해야 합니다. 각 시스템에 중복 전원을 고려하는 것도 좋습니다.

지속적인 데이터 액세스를 보장하려면, RAID 하드 드라이브 배열을 사용하십시오. 드라이브 미러링과 RAID 5가 하드웨어 시스템자원 확보을 제공할 수 있으며, 교체가 용이한 하드 드라이브로 문제의 구성요소를 대체할 수 있습니다. 중복 하드웨어가 소프트웨어 문제를 막아주는 것은 아니므로, 일반적인 시스템 백업은 반드시 수행하시기 바랍니다.

CD-ROM 드라이브는 제품을 설치하는 데 필요합니다. 서버에 반드시 CD-ROM을 설치할 필요는 없으며, 다른 서버에 있는 CD-ROM을 마운팅하거나 NFS 등의 방법을 통해 CD-ROM을 액세스해 됩니다. 또한, 업데이트를 다운로드하고 제품을 등록하려면 인터넷 연결도 필요합니다.

인프라

네트워크 서비스 클러스터를 실행하려면, 안정된 네트워크가 제공되어야 합니다. 가능하면, 모든 클러스터 노드를 단일 서브넷에 유지하고, 이 서브넷을 네트워크의 나머지 부분으로부터 독립시키는 것이 바람직합니다. 이렇게 하면 클러스터를 성능을 최대로 사용하면서도 네트워크의 나머지 부분에서 발생하는 문제를 격리시킬 수 있습니다.

트래픽 양이 매우 많은 클러스터의 경우, 단일 서브넷의 대역폭이 모두 소모될 수 있습니다. 이 경우에는 복수의 서브넷을 고려해볼 필요가 있습니다.

최대의 성능을 얻기 위해서는 모든 노드를 단일의 서브넷이나 LAN 상에서 유지하는 것이 바람직하지만, 꼭 그렇게 할 필요는 없습니다. 네트워크상의 어디에든 융통성 있게 노드를 배치할 수 있으며, 터널링 방법을 사용할 때는 특히 그렇습니다.

하지만, ATM은 모두 동일한 서브넷 상에 있어야 합니다. ATM은 모두 클러스터 자체의 가상 IP 주소를 받아야 하는데, 이것은 그 IP 주소를 포함하는 서브넷 상에서만 가능하기 때문입니다.

가용성이 높은 웹사이트를 만들 때는 네트워크 상에 중복 인터넷 라우터를 준비하는 것이 좋습니다. 그렇게 하면 라우터 중 하나가 다운되어도 여전히 외부로부터 클러스터를 액세스할 수 있습니다.

최대의 시스템자원 확보을 보장하려면 라우터가 별도의 인터넷 서비스 제공업자를 통해야 합니다. 인터넷과의 연결이 끊어지게 되면 클러스터의 높은 가용성은 별 의미가 없습니다.

도메인 이름을 IP 주소로 매핑하는 DNS 서버를 실행하는 것도 매우 바람직합니다. 이때 IP 주소를 다시 도메인 이름으로 해석하는 역 DNS 조회도 가능해야 합니다. 다른 서버와 마찬가지로, 클러스터 안의 시스템은 DHCP로 할당된 주소가 아닌 정적 IP 주소를 갖고 있어야 합니다.

제 2 장

클러스터링 개념

이 장에서는 터보리눅스 클러스터 서버의 작동을 이해하는데 필요한 기본 개념을 설명하도록 하겠습니다.
본 제품을 제대로 사용하려면 이 개념을 반드시 이해해야 합니다. 또한, 이는 클러스터 구성시 사용할 선택 사항을 이해하는 데에도 많은 도움이 될 것입니다.

이 장의 주요 내용은 다음과 같습니다.

- 클러스터란 무엇인가?
- 클러스터의 구성요소
- 다양한 종류의 클러스터
- 클러스터의 작동 방식
- 클러스터 구성 방법
- 시스템간 데이터 공유 방법

클러스터란 무엇인가?

클러스터는 개별적인 여러 개의 컴퓨터 시스템을 하나의 컴퓨터 시스템처럼 보이게 만드는 기술입니다. 이러한 정의는 단순하게 들릴 수도 있으며, 이와 비슷한 여러 가지 다른 기술이 있기 때문에 각 기술의 미묘한 차이는 이해하기가 다소 어려울 수 있습니다.

컴퓨터 클러스터링은 Digital VAX 플랫폼을 시초로 1980년대부터 다양한 형태로 만들어지기 시작했습니다. 예전에는 VMS 운영체제와 VAX 하드웨어를 조합하여 클러스터링 서비스를 제공했었습니다. 이러한 VAX 클러스터는 디스크 공간 같은 하드웨어 자원을 공유할 수 있었고, 여러 사용자에게 컴퓨팅 자원을 제공할 수 있었습니다.

이 절에서는, 클러스터링의 개념에 관해 자세히 알아보고, 다른 병렬 처리 기술과 클러스터링의 차이를 간략하게 살펴보겠습니다.

무엇이 클러스터를 진정한 클러스터로 만드는가?

클러스터링은 일종의 병렬 컴퓨팅입니다. 클러스터링과 다른 관련 기술의 중요한 차이는 클러스터를 단일 서버로 볼 수 있고 독립형 시스템들의 조합으로 볼 수도 있다는 것입니다. 예를 들어, 웹 서버 클러스터는 하나의 거대한 웹 서버로 보일 수도 있지만, 그와 동시에, 필요에 따라 클러스터 안의 각 시스템을 개별적인 시스템으로 액세스할 수도 있습니다.

클러스터 안의 각 시스템은 독립적인 컴퓨터이기 때문에, 개별적인 운영 체제와 하드웨어 및 소프트웨어를 갖습니다.

클러스터는 모든 시스템이 비슷한 하드웨어에서 동일한 소프트웨어를 실행하는 동종 클러스터일 수도 있고, 클러스터 안의 시스템이 다양한 하드웨어에서 서로 다른 운영 체제를 실행하는 이종 클러스터일 수도 있습니다.

관련 기술

클러스터링은 병렬 처리 기술의 범주에 해당됩니다. 클러스터링과 다른 기술의 중요한 차이점은 자원을 공유하거나 복제하는 수준에 달려 있습니다. 가장 낮은 수준은 시스템이 단 하나의 마더보드 위에 여러 개의 프로세서를 유지하고 다른 모든 부분을 공유하는 것입니다. 가장 높은 수준에서는 분산 프로세싱이 여러 개의 컴퓨터를 사용하되, 시스템이 단일 서버로 취급되지는 않습니다.

몇 가지 병렬 처리 방법이 아래에 나와 있습니다(가장 엄격한 바인딩에서부터 가장 느슨한 바인딩까지)

- SMP
- NUMA
- MPP
- Clustering
- 분산 프로세싱

이 절에서는 앞에서 설명한 클러스터링을 제외한 나머지 기술을 살펴보도록 하겠습니다.

SMP

현재 다중 프로세서 시스템은 보통 대칭적 유형을 띠고 있습니다. 이는 어느 하나의 프로세서가 다른 프로세서보다 더 중요하지 않고, 모든 자원이 모든 프로세스에 공평하게 사용된다는 것을 의미합니다.

이러한 유형의 시스템을 대칭적 다중 프로세싱, 또는 SMP라고 합니다. 단일 컴퓨터는 여러 개의 CPU를 가질 수 있지만 공유 메모리 공간과 I/O 장치는 단 하나만 가질 수 있습니다.

SMP의 기본이 되는 개념은 컴퓨팅 문제를 병행 프로세스로 투명하게 분배하여 그것이 동일한 컴퓨터 내의 개별적인 프로세서에서 실행되게 하는 것입니다. 가장 중요한 것은 투명성입니다.

즉, 같은 프로그램이 단일 프로세서 컴퓨터에서 시 분할로 실행될 수 있고, 개발 툴은 기존의 병렬 처리를 인식할 필요조차 없습니다.

SMP 컴퓨터에서 운영 체제 자체는 애플리케이션을 구성하는 개별적인 프로세스를 사용 가능한 CPU 간에 분배합니다. SMP 컴퓨터는 스레딩이나 가중치가 적은 프로세스를 사용하는 운영 체제 및 프로그램을 사용하는 것이 가장 좋습니다. Windows NT는 가중치가 매우 높은 스레드를 기반으로 하고, 리눅스 프로세스는 가중치가 매우 적으므로, 두 가지 모두 SMP 하드웨어에 아주 적합합니다.

2~4개의 프로세서를 갖는 SMP 시스템은 아주 간단하게 구축할 수 있지만, 그 이상의 프로세서를 갖는 시스템의 경우에는 문제가 다릅니다. 이는 모든 프로세서가 모든 I/O 및 메모리 자원을 액세스할 수 있어야 하기 때문입니다.
4개 이상의 프로세서를 갖는 시스템의 경우, 공유 자원이 병목 현상에 빠지기 시작하며, CPU를 더 추가할 경우 반환률이 감소됩니다.

NUMA

SMP 컴퓨터는 각 프로세서가 컴퓨터 안의 모든 물리적 메모리에 대해 동일한 수준의 액세스를 갖는 메모리 공유 방법을 사용합니다. 이러한 공유 방법을 균등 메모리 액세스 또는 UMA라고 합니다.

NUMA(비균등 메모리 액세스)는 다중 프로세서 컴퓨터에 있는 여러 프로세서가 단순한 SMP에서보다 효율적인 방법으로 로컬 메모리를 공유할 수 있는 더 복잡한 기술입니다. 각 CPU는 단일 메모리 영역에 대해 빠른 직접 액세스를 갖지만, 시스템 상의 다른 메모리 영역은 비교적 간접적으로 액세스할 수 있습니다.

NUMA의 기본적 개념은 특정 프로세서에게 주어진 범위의 물리적 메모리를 액세스하는 데 있어서 우선권을 제공하는 것입니다. NUMA 컴퓨터는 단순한 SMP 컴퓨터와 대규모 병렬 시스템의 중간 단계라고 생각하면 됩니다.

NUMA 시스템에서는 메모리의 어떤 부분도 액세스할 수 있지만, 어떤 메모리 주소는 다른 메모리 주소보다 액세스하는 데 더 많은 시간이 걸릴 수도 있습니다. 하지만, 비-로컬 메모리를 액세스하는 시간은 디스크나 네트워크 I/O를 액세스할 때보다 여전히 더 빠릅니다.

NUMA 컴퓨터 상의 시스템 버스는 매우 복잡합니다. 시스템 버스는 여러 개의 커넥션이 연결된 그물 모양으로 구현되기도 합니다. 또 한가지 중요한 문제는 일관성입니다. ccNUMA는 시스템이 캐시를 일관성 있게 유지한다는 것을 의미합니다. CPU는 메모리 액세스시 다른 모든 프로세서 내부의 캐시를 검사하여 검색 중인 데이터가 변경되지 않았는지 확인해야 합니다.

NUMA 시스템은 병렬 컴퓨팅과 관련하여 커다란 비중을 차지하는 프로세서 내부 통신의 최적화를 시도합니다. 클러스터와 대규모 병렬 시스템에서는 프로세서간 통신 오버헤드가 상당히 높은 편인데, 이것은 그 통신이 특정 네트워크를 통과해야 하기 때문입니다. 비-로컬 메모리를 액세스하는 속도는 로컬 메모리 액세스 속도보다 빠르지는 않지만 네트워크를 통한 통신보다는 훨씬 빠릅니다.

NUMA 컴퓨터는 여러 프로세서로의 확장이 용이합니다. 따라서, 어떤 경우에는 대규모 병렬 시스템에 버금가는 처리 성능을 제공할 수 있습니다. 단점은 컴퓨터 설계시 10억 분의 1초 단위의 타이밍과 조정 스키마를 바탕으로 하는 아주 복잡한 알고리즘이 사용된다는 것입니다. 따라서, 이러한 컴퓨터는 가격이 매우 비싼 편입니다. 하지만, 애플리케이션 소프트웨어 관점에서는 프로세서 간의 복잡한 메모리 조정 작업이 보이지 않는다는 중요한 장점을 갖고 있습니다.

대규모 병렬 시스템은 상당히 빠르긴 하지만, 그 속도를 이용하기 위해서는 문제 별로 컴퓨터를 구성해야 합니다. NUMA는 효율성이 떨어지는 대신 단순화된 개발 툴과 자원의 투명성을 제공합니다.

MPP

MPP(Massively parallel processing)는 병렬 컴퓨팅 분야를 주도해 나가고 있는 기술입니다. MPP 모델에서는 각 노드가 개별적은 프로세서 그리고 자신의 전용 자원으로 구성됩니다.

MPP 시스템의 기본적인 개념은 컴퓨팅 문제를 독립적으로 처리할 수 있는 부분으로 나누는 것입니다.
따라서, 시스템의 구조가 매우 독립적인 단위로 구성되어 있습니다.

대규모 병렬 시스템은 주로 계산 위주의 고급 연산에 사용됩니다. 예를 들어, 현재 세계에서 가장 빠른 컴퓨터는 수학적 모델을 통해 핵 폭발을 시뮬레이트하는 MPP 시스템입니다.

MPP가 갖고 있는 문제 중 하나는 병렬 시스템 전용 프로그램을 작성해야 한다는 것입니다(이것은 Beowulf과 같은 몇몇 클러스터에서 발생하는 문제이기도 함). 일반적으로 사용되는 두 가지 API로 PVM과 MPI를 들 수 있습니다. 이들 API는 문제를 병렬 식으로 처리될 수 있는 단위로 나누는데 중점을 두고 있으므로 해결할 문제를 이런 식으로 나눌 수 없다면, MPP 시스템은 별 도움이 되지 않습니다.

분산 프로세싱

분산 프로세싱은 여기에 소개된 용어 중에서도 가장 모호한 개념일 것입니다. 분산 프로세싱은 처리되어야 할 작업이 여러 장소로 나뉘어 처리됨을 의미합니다. 분산 프로세싱의 가장 일반적인 예로 클라이언트/서버 구조를 들 수 있는데, 서버는 특정 작업을 수행하고, 클라이언트는 그 작업의 또 다른 부분(주로 사용자에게 정보를 디스플레이 하는 작업)을 수행합니다.

분산 시스템은 클러스터에 비해 다소 느슨하게 결합된다고 볼 수 있으며, 실제로는 그 어떤 결합 방식도 찾아보기 어렵습니다. 어떤 단일 서버로 온전한 하나로 관리되지 않습니다.

분산 프로세싱에서는 노드가 자신의 개별적인 ID를 유지하는 데 비해, 클러스터 노드는 일반적으로 익명입니다. 분산 프로세싱 시스템에서는 "서버 Y의 데이터 X를 사용" 하는 것이고, 클러스터에서는 "클러스터의 데이터 X를 사용" 하는 것입니다.

클러스터 구성 요소

클러스터를 구성하는 시스템은 노드와 관리자입니다. 클러스터 노드는 프로세싱 자원을 제공하는 시스템이고, 클러스터 관리자(또는 관리자)는 노드를 서로 연결하여 단일 시스템처럼 보이게 만드는 로직을 제공합니다.

클러스터 노드

클러스터 노드는 클러스터의 실질적인 작업을 처리합니다. 일반적으로, 클러스터 노드는 클러스터에 속하도록 구성해야 합니다. 또한, 클러스터 노드는 클러스터링될 애플리케이션을 실행합니다. 클러스터의 종류에 따라, 이 애플리케이션 소프트웨어는 클러스터에서 실행되도록 특별히 만든 것일 수도 있고, 독립형 시스템을 겨냥하여 제작된 표준 소프트웨어일 수도 있습니다.

터보리눅스 클러스터 서버와 터보리눅스 EnFuzion은 모두 독립형 시스템용으로 작성된 소프트웨어를 사용할 수 있게 허용합니다.

클러스터 안에서 사용될 소프트웨어를 구성하는 작업은 아주 간단합니다. 본 설명서에서는 클러스터 노드를 간단히 노드, 서버 또는 서버 노드라고 지칭하겠습니다.

클러스터 관리자

클러스터 관리자는 모든 노드 사이에 작업을 분배하며, 대부분의 클러스터는 단 하나의 클러스터 관리자를 갖습니다. 어떤 클러스터는 완전히 대칭적 이어서 어떤 클러스터 관리자도 갖지 않지만, 최근에는 이러한 클러스터를 찾아보기 힘듭니다. 이러한 클러스터는 복잡한 조정 알고리즘을 요구하며 설정이 더 어렵습니다.

터보리눅스 클러스터 서버에서는 클러스터 관리자를 ATM 또는 Advanced Traffic Manager라고 합니다. 클러스터 서버는 ATM에 대해 fail-over를 지원하여 단일 실패 지점이 생기지 않도록 합니다.
주 ATM이 다운되면, 백업 ATM이 그 역할을 대신합니다.

클러스터 관리자는 클러스터 노드의 역할도 수행할 수 있습니다. 시스템이 작업을 나눈다고 해서, 그 시스템이 작업 자체를 처리할 수 없다는 것을 의미하는 것은 아닙니다.

하지만, 대규모 클러스터일수록 작업 분할에 더 많은 컴퓨팅 기능을 필요로 하기 때문에 하나 이상의 컴퓨터가 클러스터 관리자의 역할을 전담하는 경향이 있습니다. 두 가지 역할을 분리하면 클러스터 관리도 더욱 수월해집니다.

클러스터의 종류

앞 절에서도 설명했지만, 클러스터의 정의는 매우 광범위합니다. 사실, 너무 광범위해서 서로 상이하게 다른 기술들을 어떻게 모드 클러스터라고 할 수 있는지 의문스러울 정도입니다. 그만큼 클러스터는 여러 가지 다양한 용도로 구현할 수 있다는 것입니다. 클러스터의 주요 목적은 CPU 자원을 풀링하거나, 여러 컴퓨터 간에 작업 부하를 조정하거나(부하 조정), 가용성이 높은 시스템을 구축하거나 주 시스템이 실패할 경우에 대비한 백업 시스템을 제공하는 것입니다(fail-over). 이러한 목적에 따라 다양한 종류의 클러스터를 만들 수 있지만, 어느 정도 공통되는 부분도 있습니다.

터보리눅스 클러스터 서버는 높은 가용성, 부하 조정, fail-over를 구현하는 데 사용할 수 있습니다. 터보리눅스 클러스터 서버는 일반적인 의미의 공유 프로세싱은 제공하지 않으며, 그 대신 네트워크 서비스에 대한 부하 조정을 제공합니다. 각 서버는 들어오는 네트워크 서비스 요청을 받아 처리한 다음, 클라이언트로 다시 응답을 보냅니다.

공유 프로세싱

여러분이 "리눅스 클러스터링"이라는 용어를 들을 때 가장 먼지 떠올리는 것은 아마 Beowulf 프로젝트일 것입니다.

Beowulf는 여러 시스템의 프로세싱 능력을 조합하여 대용량의 프로세싱 능력을 갖는 하나의 시스템을 제공합니다. 이것은 원래 과학용 시스템이나 CPU 위주의 용도로 설계된 것인데, 이 시스템에서는 API에 따라 특별히 작성된 프로그램만 자신의 작업을 여러 시스템 사이에 분배할 수 있습니다. Beowulf에 관한 자세한 내용은 웹 사이트 http://www.beowulf.org/를 참조하시기 바랍니다.

클러스터 서버는 이러한 유형의 클러스터링을 제공하지 않습니다. 공유 프로세싱을 제공하는 데 사용할 수 있는 패키지는 EnFuzion이며, 이 터보리눅스 제품은 프로그램을 재작성하지 않아도 시스템에 사용할 수 있다는 장점이 있으며, 태스크 기반의 프로세싱 시스템입니다.

EnFuzion에 관한 자세한 내용은 웹 사이트 http://www.turbolinux.co.kr/products/enf/를 참조하시기 바랍니다.

부하 조정

부하 조정은 공유 프로세싱과 비슷하지만, 노드 간 통신이 필요 없습니다. 부하 조정을 이용하면 각 노드는 클러스터 관리자가 자신에게 할당한 요청을 처리하고 클러스터 관리자는 모든 시스템 간에 공평하게 작업 부하를 분배하는 방식으로 요청을 분배합니다.

Fail-over

fail-over는 부하 조정과 비슷하지만, 모든 클러스터 노드 사이에 요청이 분배되는 대신, 한 시스템이 모든 요청을 처리합니다. 그리고 그 시스템이 다운될 경우에만 클러스터 안의 다른 시스템 중 하나가 그 역할을 대신하게 됩니다.

높은 가용성

모든 컴퓨터가 항상 작동 상태에 있는 것이 가장 바람직하다고 볼 수 있으나, 실제로는 컴퓨터가 다운되는 경우가 있습니다.

이것이 그저 번거로운 문제로만 그친다면 상관없지만, 어떤 경우에는 심각한 문제를 일으킬 수 있습니다. 따라서 컴퓨터 업체들은 시스템의 가용성을 높이는 방법을 개발해야 했습니다.

높은 가용성은 시스템 자원을 최대한 자주 사용할 수 있게 유지하는 방법입니다. 클러스터링이 바로 그러한 방법을 제공합니다.

클러스터링을 사용하면 하드웨어 중복을 위해 막대한 비용을 들일 필요 없이 여러 시스템을 서로 클러스터링하여 필요한 자원을 제공할 수 있습니다. 또한, 어느 한 시스템에 문제가 생기더라도 다른 시스템이 그 작업 부하를 대신 넘겨받을 수 있습니다.

높은 가용성은 하드웨어나 소프트웨어를 통해 구현할 수 있습니다. 일반적으로 하드웨어 시스템이 더 비싸지만, 소프트웨어 솔루션도 결코 무시할 수 없습니다. 높은 안정성을 원할수록 비용은 더 많이 들기 마련입니다.

가용성은 업타임(작동가능시간)의 백분율로 계산됩니다. 일반 서버의 경우 99%의 업타임을 갖지만, 높은 가용성을 목적으로 하는 시스템은 최대 99.99%의 업타임을 가질 수 있습니다.

이것을 보통 "four nines" 가용성이라도 합니다.
높은 가용성은 부하 조정이나 fail-over를 통해 구현이 가능합니다.

클러스터 작동 방식

클러스터 관리자는 클러스터의 핵심이 되는 부분으로서, 클러스터 노드 간의 작업 분배 방식을 결정합니다. 클러스터 관리자는 작업 부하를 나누어 각 클러스터 노드로 전달합니다. 그러면 클러스터 노드는 그 작업을 처리하여 클러스터 관리자로 다시 결과를 보내거나, 그 결과를 요청했던 클라이언트로 직접 결과를 보냅니다.

트래픽 관리

터보리눅스 클러스터 서버가 구현하는 서비스 지향 클러스터의 경우, 작업 부하 관리를 트래픽 관리라고 합니다. 이것은 수행해야 할 "작업(work)"이 들어오는 네트워크 서비스 요청에 응답하기 때문입니다.

클러스터 관리자는 모든 클러스터 노드들에 네트워크 트래픽을 전달해야 하는데, 이것은 마치 교통 경찰의 역할과도 같은 것입니다.

터보리눅스 클러스터 서버에 사용되는 트래픽 스케줄링 알고리즘을 가중치 변경 라운드 로빈이라고 합니다. 이 방법은 각 클러스터가 처리할 수 있는 작업부하의 양에 비례하여, 클러스터 안의 모든 노드 사이에 트래픽을 고르게 분배합니다. 각 서버는 다른 시스템에 비해 자신의 성능을 지정하는 가중치를 할당 받습니다.

스케줄링 알고리즘은 클라이언트의 영속성을 지원하도록 확장 시킬 수 있습니다. 이 기능을 설정하면(sticky bit라고도 함), 특정 클라이언트가 클러스터 안의 특정 서버와 바인딩됩니다.

SSL을 통해 사용할 수 있는 서비스 같은 일부 서비스는 새로운 사용자가 서버에 연결될 때마다 인증을 요구합니다. 만약 영속성이 지원되지 않는다면, 클라이언트가 클러스터 안의 서로 다른 서버로 연결될 때마다 자신의 암호를 재입력해야 합니다.

클러스터 서버는 다음의 세가지 방법으로 클러스터로부터 노드로 트래픽을 전달합니다.

- 직접 포워딩
- 터널링
- NAT

직접 포워딩

직접 포워딩은 ATM과 클러스터 노드가 동일한 네트워크 영역이나 서브넷에 연결되어 있을 때 사용할 수 있습니다. 이 방법으로 전달되는 패킷은 클러스터 노드의 MAC 주소로 직접 전송됩니다. IP 패킷은 전혀 변경되지 않으며, 클러스터 노드는 ATM에 도착했을 때와 같은 패킷을 보게 됩니다.

이 방법은 가장 빠르고 오버헤드가 가장 적기 때문에 기본 설정되어 있습니다. 직접 포워딩 방법은 아웃바운드 트래픽(클라이언트로 반환되는 응답)이 ATM을 통해 전달될 필요가 없다는 장점이 있습니다. 응답 패킷은 해당 목적지로 직접 전송됩니다.

터널링

클러스터 노드가 ATM과 같은 네트워크 영역에 위치하고 있지 않을 때는, 터널링 포워딩 방법을 사용할 수 있습니다. 터널링은 IP 패킷을 다른 네트워크 트래픽 안에 캡슐화하는 방법입니다. 이 방법을 사용하면 두 시스템 사이에 가상적인 직접 연결을 만들 수 있습니다. 이와 같은 지점간(point-to-point) 연결에서는 패킷이 가상 연결을 통해 클러스터 노드까지 안전하게 도착합니다.

터널링 방법은 리눅스와 유닉스 시스템에서만 사용할 수 있습니다. 이 방법은 IP-IP 커널 모듈을 사용하여 트래픽 관리자와 클러스터 노드 사이에 지점간 연결을 만듭니다. 클러스터 노드에서 사용되는 커널은 반드시 IP 터널링을 지원하도록 구성해야 합니다. 터보리눅스 클러스터 서버에서 제공하는 커널은 이러한 기능을 기본적으로 지원하며, 클러스터 서버 데몬이 링크의 양 끝을 자동적으로 구성합니다. 여러분이 수동으로 지점간 연결을 설정하여 터널 인터페이스를 만들 수도 있습니다.

터널링 방법을 사용할 경우, 캡슐화 과정에서 직접 포워딩 방법에 비해 다소 성능을 저하시키는 오버헤드가 생길 수 있습니다. 하지만, 직접 포워딩 방법과 마찬가지로 아웃바운드 패킷은 ATM을 통과할 필요 없이 클러스터 노드로부터 클라이언트로 직접 전달됩니다.

주)
클러스터 서버에 사용되는 IP 터널링은 암호화되지 않기 때문에, 다른 사람이 트래픽 관리자로부터 노드로 전달되는 패킷을 가로챌 수도있습니다. LAN 외부에 있는 노드를 추가해야 할 경우, 데이터 전송을 안전하게 하려면 VPN(Virtual Private Network)을 구현해야 합니다.

NAT

NAT는 Network Address Translation의 약자입니다. NAT는 주로 개인용 네트워크를 인터넷에 연결된 방화벽 뒤에 숨기는 데 사용됩니다. NAT는 RFC 1631에 규정되어 있으며, IP 주소 공간이 급격하게 소모되는 현상을 완화시킬 목적으로 만들어졌습니다.

NAT는 개인용 네트워크와 공용 네트워크 사이에 위치하여, 개인용 네트워크로부터 나오는 패킷을 마치 NAT 자체에서 나온 것처럼 보이게 변경합니다. 패킷이 NAT으로 전송되면, NAT는 내부 네트워크 상의 어떤 시스템으로 그 패킷을 전달할 것인지 결정합니다.

NAT는 이를 위해 초기화되어 있는 커넥션 테이블을 유지합니다. 개인용 네트워크에 있는 클라이언트가 만든 각 커넥션에 대해 커넥션 테이블이 클라이언트로 전송될 응답을 지정합니다.

리눅스의 `ipchains` 패키지에 사용되는 NAT 버전을 IP masquerading(IP 매스커레이딩) 이라고도 합니다. 작동 방식이 친숙하게 들린다면, 그것은 클러스터 트래픽 관리자의 역할과 매우 비슷하기 때문입니다.

NAT는 보통 클라이언트 시스템을 숨기는 데 사용되지만, 클러스터에서는 서버를 숨기는 데 사용됩니다. 이 차이는 매우 중요한데, 이 차이에 따라 커넥션 테이블 사용 방식이 달라지기 때문입니다.

터보리눅스 클러스터 서버의 경우, NAT 방법은 다른 두 가지 트래픽 포워딩 방법이 사용하는 것과 동일한 커넥션 테이블을 사용합니다.

NAT를 사용하면 클러스터 노드 자체를 특별히 변경할 필요가 없기 때문에 구성 작업이 단순해집니다.

여러분이 해야 할 일은 클러스터 노드가 "내부" 서브넷에 있는지 확인하고, 클러스터 노드의 기본 게이트웨이를 클러스터 구성 파일에 정의된 NAT 게이트웨이 주소로 정의하는 것뿐입니다.

또한, NAT는 클러스터 노드를 외부에서 직접 액세스할 수 없게 하기 때문에 보안을 한층 강화합니다. NAT의 단점은 모든 아웃바운드 트래픽이 반드시 NAT와 주소 변환 프로세스를 통과해야 하기 때문에 성능이 약간 떨어진다는 것입니다.

NAT를 사용할 수 없는 네트워크 서비스도 있습니다. 예를 들어, FTP는 서로 다른 포트에 대해 두 개의 개별적인 TCP 연결을 사용하기 때문에 NAT를 사용할 수 있습니다. 그 밖에 프로토콜의 상위 부분에 IP 주소나 포트 번호를 포함하고 있는 서비스도 사용할 수 없습니다. 자세한 내용은 RFC 1631을 참고하시기 바랍니다.

클러스터 관리

클러스터를 관리하는 일은 단순히 클러스터 안의 모든 시스템을 관리하는 것보다 약간 더 복잡합니다. 각 서버와 시스템을 완전한 하나로 유지해야 하기 때문입니다. 클러스터 관리는 주로 클러스터 관리자가 담당합니다.

클러스터 관리는 주로 클러스터의 성능을 감시하는 일입니다. 여기에는 각 시스템을 감시하는 일 뿐 아니라 클러스터 전체의 성능을 감시하는 일도 포함됩니다.

개별적인 시스템에 지나친 부하가 걸리면 그 시스템에 더 적은 작업을 부여하도록 클러스터 구성을 변경해야 합니다.(혹은, 특정 서버와 관련된 몇 가지 구성 문제가 있을 수도 있습니다.)

또한, 클러스터 전체의 성능도 감시해야 합니다. 모든 클러스터 노드에 지나친 부하가 걸릴 경우, 한 두개의 노드를 추가하여 성능을 향상시키는 것도 좋은 방법입니다.

클러스터를 관리하는 데 있어 또 한가지 중요한 문제는 모든 시스템이 같은 소프트웨어를 실행하고 동일한 컨텐트를 사용하게 하는 것입니다. 터보리눅스 클러스터 서버는 컨텐트를 복제할 수 있는 동기화 툴을 제공하여 모든 서버가 일관성 있게 유지되게 합니다.

공유 데이터 저장소

두 개 이상의 시스템이 동일한 데이터에 대해 같은 액세스를 제공하기 위해서는 그 데이터를 공유할 수 있는 방법이 있어야 합니다. 사실상 이것은 생각보다 훨씬 더 어려운 일입니다. 데이터가 자주 변경된다면, 모든 시스템을 동기화된 상태로 유지하는 방법도 필요합니다. 이 절에서는 데이터를 공유하는 데 사용할 수 있는 소프트웨어와 하드웨어 솔루션을 살펴보도록 합시다.

소프트웨어

가장 쉬운 공유 저장 방법은 소프트웨어를 통한 것입니다. 물론, 하드웨어 솔루션이 더 강력하고 안정되긴 하지만, 대부분의 경우 간단한 소프트웨어 방법을 사용하여 데이터를 공유할 수 있습니다.

동기화

데이터를 공유하는 가장 기본적인 방법은 각 서버로 해당 데이터를 복사하는 것입니다. 물론, 이 방법은 데이터가 자주 변경되지 않을 때에만 가능하며, 클러스터 안의 모든 서버에 대해 관리 액세스 권한을 갖고 있는 사람만 사용할 수 있습니다.

터보리눅스 클러스터 서버는 두 가지의 동기화 툴을 제공합니다. 하나는 서버의 구성을 동기화하는 데 사용되고, 다른 하나는 컨텐트를 동기화하는 데 사용됩니다. 이들 동기화 툴은 직접 실행이 가능하고 `turboclusteradmin` 프로그램을 통해 액세스할 수도 있습니다. 이에 관한 내용은 7장에서 자세히 다룰 것입니다. 동기화 툴을 사용하여 데이터의 일관성을 유지할 수 있다면 그것이 가장 쉬운 방법입니다. 동기화 툴은 복잡한 관리 작업 없이도 데이터 시스템자원 확보 기능(data redundancy)을 제공합니다.

이밖에 다른 복제 방법도 사용할 수 있습니다. 일반적으로 사용되는 복제 시스템 중 하나는 LDAP (Lightweight Directory Access Protocol)입니다. LDAP를 사용하면 여러 시스템으로 복제되는 하나의 데이터베이스를 유지할 수 있습니다. 이 방법은 데이터베이스 시스템에 시스템자원 확보과 안정성을 제공하며, 비교적 쉽게 설정할 수 있습니다. LDAP는 상용 데이터베이스가 아니기 때문에, SQL을 구현하지 않습니다. LDAP는 개체를 기반으로 하는 네트워크 정보 디렉토리로 사용하도록 만들어졌으나 필요에 따라 다른 용도로도 사용할 수 있습니다.

분산 파일 시스템

데이터가 너무 자주 변경되어 수동으로 동기화 작업을 수행하기 힘든 경우, 분산 파일 시스템을 사용해야 합니다. 여러분이 선택할 수 있는 분산 파일 시스템에는 NFS, AFS, DFS, Coda, Intermezzo, 그리고 GFS가 있습니다.

유닉스와 리눅스 시스템은 보통 NFS를 사용하여 네트워크 상의 데이터를 공유합니다. NFS는 잘 알려진 시스템으로서 서버나 클라이언트로 쉽게 구성할 수 있습니다.

하지만, NFS는 내부적으로 다소 불안하고 여러 시스템으로 데이터를 복제하는 방법이 제공되지 않는다는 단점이 있습니다. 따라서, NFS를 사용할 경우, 여전히 단일 실패 지점이 존재할 가능성이 높으며, 바로 이러한 점 때문에 먼저 클러스터를 만드는 것이 바람직합니다.

이러한 NFS의 단점을 극복하기 위해 많은 분산 파일 시스템이 개발되었지만, 아직까지는 NFS를 능가할 만한 대체 방법은 없습니다.

NFS와 많은 공통점을 갖고 있으면서 NFS의 불완전한 인증 방법을 대신할 수 있는 분산 파일 시스템은 AFS (Andrew File System)입니다. AFS는 피츠버그에 위치한 Carnegie Mellon 대학의 Andrew Project에서 개발된 것으로 일종의 상업용 소프트웨어입니다.

AFS의 가장 중요한 특징은 Keberos 프로토콜을 기반으로 한 안정된 인증 메커니즘을 제공한다는 것입니다. AFS는 성능, 사용법, 관리 면에서 NFS보다 우수한 점을 많이 갖고 있습니다.

AFS와 밀접한 관련이 있는 파일 시스템은 Transarc의 DFS(Distributed File System)입니다. DFS는 정교한 복제 기능과 부하 조정 기능을 제공하는 엔터프라이즈급의 공유 저장소 솔루션입니다. DFS의 주요 목적은 기업 내의 네트워크와 도메인 사이에 투명성을 유지하여 중앙 집중적인 관리를 용이하게 하는 것입니다.

Coda 파일 시스템은 현재 리눅스 커널과 함께 제공되는 Open Source 분산 파일 시스템입니다. Coda는 AFS와 매우 비슷한 시스템을 유지하면서도 몇 가지 최신 기능을 함께 제공합니다. 또한, 단속적인 연산, 서버 측 복제, 부분적인 네트워크 실패시 연속 연산, 그리고 확장성과 대역폭 조정 기능을 제공하여 몇 가지 가용성 문제를 해결합니다.

Intermezzo도 오픈 소스 분산 파일 시스템입니다. Intermezzo의 장점 중 하나는 고유 파일 시스템의 상위 계층에 위치하여 여러분이 고유 파일 시스템을 사용하여 데이터를 저장할 수 있게 해준다는 것입니다.

Intermezzo는 Coda보다 최신 컴퓨팅 환경과 장비의 능력을 더 잘 이해하며, Cola와 마찬가지로 높은 가용성, 대규모 복제, 분리된 네트워크에 중점을 둡니다. 하지만, 아직까지는 베타 개발 단계에 있습니다.

자세한 내용은 웹 사이트 http://www.inter-mezzo.org/를 방문하시기 바랍니다.

최고의 분산 파일 시스템 솔루션 중 하나는 GFS(Global File System)입니다.
이 솔루션은 파일 시스템 소프트웨어뿐 아니라 하드웨어 지원을 요구합니다.

하드 드라이버는 파일 시스템에 속하는 모든 시스템(즉, 클러스터 안의 모든 노드)에 직접 연결되어 있어야 합니다. 이 때 사용되는 방식이 double-ended SCSI(양두 SCSI) 또는 파이버 채널입니다.

하드웨어

대부분의 고급 저장 시스템은 하드웨어를 기반으로 합니다. 여기에 사용되는 두 가지 주요 기술은 SAN(Storage Area Networks)와 NAS(Network Attached Storage)입니다. 파이버 채널과 double-ended SCSI 체인을 사용하여 구현할 수도 있습니다.

SAN (Storage Area Networks)

SAN(Storage Area Network)는 오류 발생률이 적을 뿐만 아니라, 그 자체에 있는 분산 네트워크는 매우 안정된 데이터 서빙 연산을 제공하는 데 사용됩니다. 개념적으로 SAN은 애플리케이션 서버와 물리적 저장 장치(NAS 장치, 데이터베이스 서버, 전통적인 파일 서버, near-line 및 아카이벌 저장 장치 등) 사이에 있는 계층입니다.

SAN과 관련된 소프트웨어는 이와 같은 백엔드(전위) 저장소를 투명하게 사용할 수 있게 하고 그에 대한 중앙 집중적인 관리 기능을 제공합니다.

SAN의 주요 특징 중 하나는 완전히 독립적인 네트워크로 실행되며, 보통 독점적이거나 저장소를 기반으로 하는 네트워안 기술을 사용한다는 것입니다.

최근 대부분의 SAN은 파이버 채널을 사용하는 추세이나 SAN 구현은 결코 간단한 일은 아닙니다. SAN을 관리하려면 전담 기술 지원팀이 필요합니다. 이 때문에, SAN은 대기업 환경에서만 찾아볼 수 있습니다.

NAS(Network Attached Storage)

NAS 장치는 기본적으로 구식 파일 서버를 폐쇄형 시스템으로 변형한 것입니다. NAS 장치의 마지막 클럭 사이클은 모두 디스크와 네트워크 간 데이터를 펌핑하는 데 사용됩니다. 이것은 애플리케이션 서버(메일 서버, 웹 서버, 또는 데이터베이스 서버)를 파일 연산과 관련된 오버헤드로부터 분리시킬 때 매우 유용합니다.

NAS 장치는 이더넷 카드와 몇 가지 파일 서브 소프트웨어가 장착된 하드 드라이브로 생각할 수 있습니다. NAS 장치는 파일 서버에 비해 독립적이고 필요한 관리 작업이 더 적다는 장점을 갖고 있습니다.

NAS의 중요한 특징 중 하나는 플랫폼으로부터 독립적이어야 한다는 것입니다. NAS는 다용도 저장 장치로서 Windows와 유닉스 클라이언트 모두에게 동일한 투명한 서비스를 제공할 수 있어야 합니다.

고속 드라이브 인터페이스

Fail-over 클러스터링을 구현하기 위해서는 중복 서버가 많은 성능 히트를 차지하지 않고 원격 저장 장치를 마치 로컬 네트워크 상에 있는 저장 장치처럼 액세스할 수 있는 방법이 있어야 합니다. 이 문제에 대한 두 가지 해답은 double-ended SCSI와 파이버 채널입니다.

Double-ended SCSI(차동 SCSI라고도 함)은 SCSI 설계에 시스템자원 확보을 적용하여 더 긴 SCSI 케이블을 사용할 수 있게 하고, 실용적인 고속 아웃보드 저장 장치를 만듭니다. Single-ended SCSI 케이블의 경우, 실제로는 신호 라인이 하나씩 걸러서 접지됩니다.

Double-ended SCSI는 전압을 변환시켜 이와 같은 중복 접지 라인을 사용하여 인접한 신호 라인과 동일한 신호를 전달합니다. 실제 효과는 신호가 두 배로 강해짐으로써 신호 유실 없이 훨씬 더 긴 케이블 길이를 사용할 수 있게 된다는 것입니다. 외부 장치를 사용하는 컴퓨터가 어느 정도 인접해 있을 때는 Double-ended SCSI를 사용하는 것이 적합합니다.

파이버 채널 인터페이스는 실제로 광 케이블을 사용하여 레이저 광선을 통해 인코딩된 SCSI 신호를 전달합니다 (고속 네트워크 인터페이스와 거의 유사한 방식). 이 인터페이스는 사실상 무제한적인 로컬 영역(최대 6마일)을 높은 대역폭으로 이용할 수 있으며, SAN을 구현하는 필요한 핵심 기술입니다. 물론, 로컬 인터페이스에 비하면 가격은 상당히 비싼 편입니다.

제 3 장

설 치

이 장에서는 터보리눅스 클러스터 서버를 설치하는 방법을 알아보겠습니다. 설치 프로그램은 아주 간단하며 여러분에게 자세하게 설치 절차를 안내할 것입니다. 제품을 설치한 후에는 구성을 해야 합니다. 구성에 관한 내용은 다음 장에서 다루도록 하겠습니다.

이 장에서 다룰 주요 내용은 다음과 같습니다.

- 설치 개요
- 클러스터 서버 설치하기
- 설치 후
- 설치 문제 해결하기

주)
터보리눅스 클러스터 서버를 설치하기 전에 시스템을 완벽하게 백업하기 바랍니다. 다른 소프트웨어를 설치할 때와 마찬가지 로, 설치 도중에 문제가 발생하여 시스템의 데이터가 손상될 수 있기 때문입니다.

설치 개요

터보리눅스 클러스터 서버는 클러스터 안에 있는 각각의 주 ATM과 백업 ATM에 모두 설치해야 합니다. 각 클러스터 노드에 모두 설치할 필요는 없지만, 클러스터 내 각 시스템에 설치할 것을 권장합니다.

모든 노드에서 클러스터를 실행하면 여러분이 수행해야 할 구성 및 유지 보수 작업이 훨씬 단순해질 것입니다. 클러스터 서버를 실행하고 있는 시스템은 데몬이 자동적으로 구성 작업을 수행하기 때문에, 개별적으로 구성할 필요까지는 없습니다.

또한, 클러스터 서버를 실행하는 시스템에 있는 컨텐트는 동기화하기가 쉽습니다. 클러스터 서버가 없는 노드는 모든 내용을 수동으로 동기화시켜야 클러스터의 일관성을 유지할 수 있습니다.

클러스터 서버는 CD-ROM으로 제공됩니다.

클러스터 안의 각 시스템에 CD-ROM 드라이브가 없다면, CD-ROM 드라이브가 있는 시스템에서 CD를 마운팅하고 NFS나 기타 공유 파일 시스템을 통해 그것을 익스포팅할 수 있습니다.

그 다음 다른 시스템에서 네트워크 공유를 마운팅하여 설치 작업을 수행하면 됩니다.

CD-ROM을 마운팅한 후에는, 로컬 또는 네트워크 공유를 통해 클러스터 서버가 들어 있는 디렉토리로 가서 설치 프로그램을 시작합니다.

설치 프로그램은 단계별로 설치 절차를 안내합니다. 대부분의 경우 기본 설정을 선택하고 ENTER를 눌러 다음 단계로 진행하면 됩니다.

설치가 완료되면 재부팅을 요청하는 화면이 표시될 것입니다.

다른 콘솔에서 저장되지 않은 데이터를 갖고 있는 애플리케이션이 실행되고 있는지 확인한 다음, ENTER를 눌러 재부팅하십시오.

시스템이 안전하게 종료되고 재부팅됩니다.

클러스터 서버 설치하기

설치 프로그램이 안내하는 절차를 따르면 아주 쉽게 터보리눅스 클러스터 서버를 설치할 수 있습니다.
다른 소프트웨어를 설치할 때와 마찬가지로, 설치 단계를 수행하려면 root로 로그 인 해야 합니다.

1. CD-ROM을 마운팅하십시오.

```
# mount /mnt/cdrom
```

2. CD-ROM이 마운팅된 디렉토리로 이동하십시오.

```
# cd /mnt/cdrom
```

3. 관련 도큐멘테이션(설명서)과 개정판 정보, 특히 README 파일과 RELEASE.NOTES 파일을 읽어보십시오.
   (이 파일은 TLCS-install 프로그램의 메인 메뉴를 통해 액세스할 수도 있습니다.)
4. 설치 프로그램을 시작하십시오.

```
# ./TLCS-install
```

   설치 프로그램은 먼저 자신이 어떤 리눅스 배포판에서 실행되고 있는지 검사합니다. 현재 지원되는 배포판은 터보리눅스 서버와 레드햇 리눅스입니다. 지원되는 배포판이 검사되지 않으면 설치 프로그램이 종료됩니다. 이러한 경우, 명령 프롬프트에서 redhat 이나 turbolinux를 입력하여 설치 프로그램에게 어떤 배포판을 실행하고 있는지 알려줄 수 있습니다.

```
# ./TLCS-install turbolinux
```

--test나 -t 옵션을 통해 테스트 모드를 사용할 수도 있는데 이 모드는 사실상 설치 작업을 수행하는 것이 아니라 성공적 설치에 필요한 모든 요구 사항을 검사합니다. --help 나 -h 옵션을 통해 도움말을 사용하면 자세한 옵션과 구문을 알 수 있습니다.

5. 시작 화면이 나타나면 ENTER를 누르거나 ' OK' 를 클릭하여 계속 진행하십시오.

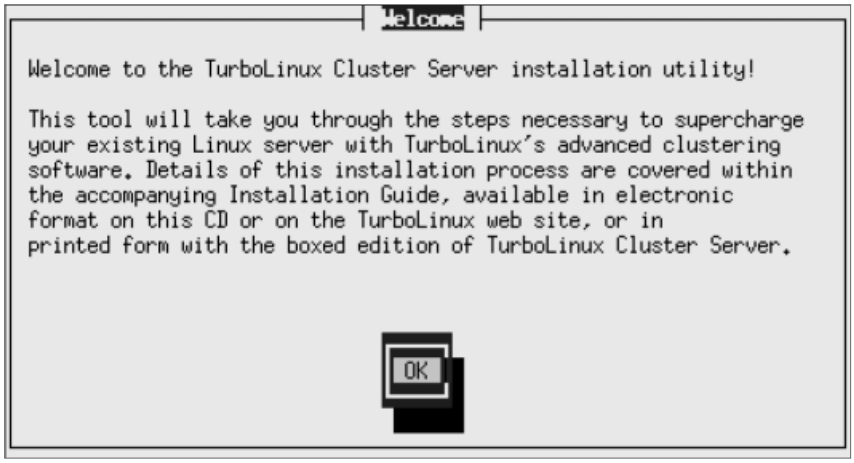

그림 3.1 초기 설치 화면

6. 사용권 계약서 화면이 나타나면 모든 내용을 자세히 읽어본 후 설치를 계속 진행하십시오.
커서 키를 사용하면 텍스트를 위아래로 스크롤할 수 있습니다.

사용권 계약서를 읽어본 후, 계속 진행하려면 'I agree' 를 클릭하십시오.
사용권 계약서에 동의하지 않을 경우, 'Exit' 를 클릭하면 설치 프로그램이 종료되고 프롬프트로 돌아갑니다.

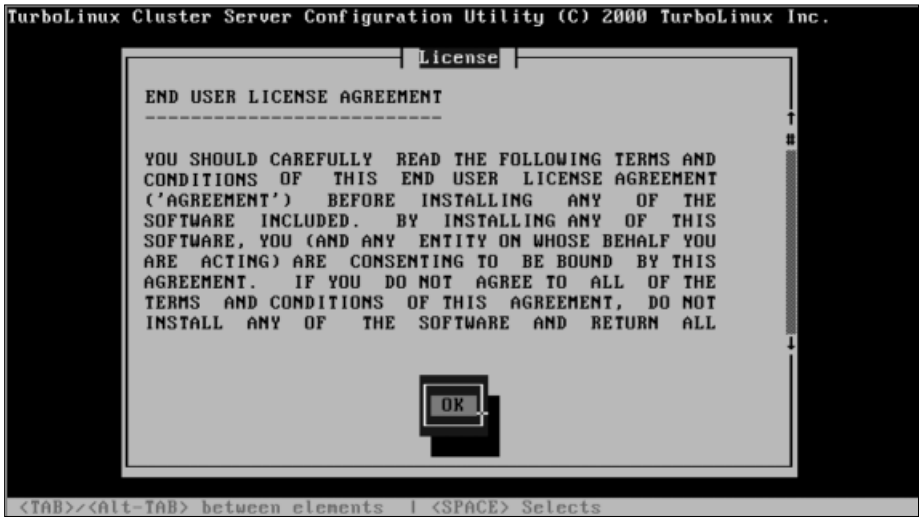

그림 3.2 사용권 계약서

7. 사용권 조항에 동의하면, 설치 프로그램은 여러분이 실행하고 있는 리눅스 배포판의 종류를 검사합니다.

이 검사가 성공하면 아래에 나와 있는 그림처럼 배포판의 명칭과 함께 커널 버전을 디스플레이합니다.

그림 3. 3 검사된 커널 버전과 배포판

계속 진행하려면 'OK'를 클릭하거나 ENTER를 누르십시오.

8. 이제 설치 메뉴가 나타납니다.
   여기에서 여러분이 선택할 수 있는 옵션은
   Guided Install (설치안내)
   Install TLCS kernel (커널 변경 설치)
   Libraries and utilities (라이브러리 및 유틸리티 설치)

그리고 Change LILO config(LILO 설정변경)입니다. 이 메뉴에서 관련된 문서를 열어 볼 수도 있습니다.
메뉴는 다음과 같습니다.

그림 3.4 설치 메뉴

설치 절차를 순서대로 따라 필요한 모든 요소를 설치하려면 'Guided Install'을 설치해야 합니다.
다른 옵션은 나중에 제품의 일부분을 설치하거나, 어떤 문제가 생겼을 때 사용됩니다.
'Guided Install'은 각 섹션을 순서대로 안내합니다.

9. 길잡이 설치를 시작하면 커널을 교체해야 한다는 경고 메시지가 나타날 것입니다.
새로운 커널을 설치하면 시스템이 작동되지 않을 수도 있으므로, 진행하기 전에 중요한 데이터는 모두 백업해 두어야 합니다. (LILO 프롬프트에서 'linux'를 선택하면 이전 커널을 사용할 수 있음)

계속 진행 하려면 'Yes'를, 설치를 중단하고 시스템을 백업하려면 'No'를 선택하십시오.

10. 그 다음 화면은 설치할 커널을 선택하는 화면입니다. 설치 프로그램은 여러분이 실행하고 있는 것과 비슷한 구성을 갖고 있으면서 더 최신 버전인 커널을 선택합니다.

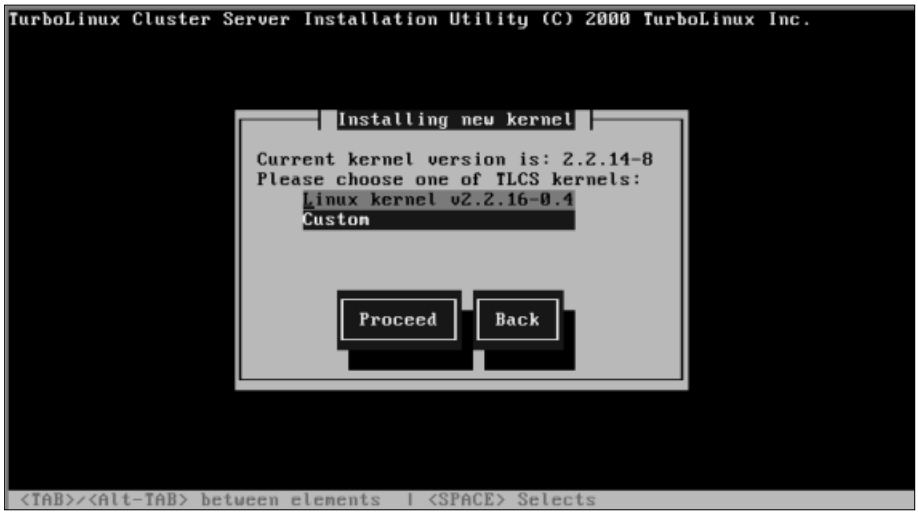

그림 3.5 설치할 커널 선택하기

특별한 이유가 없다면, 가장 최신 버전을 선택하는 것이 좋습니다. 해당 커널이 없다면, 터보리눅스 웹사이트를 방문하여 여러분의 원하는 커널이 있는지 찾아 보고 다운로드하십시오. 아니면, 사용자 정의 커널을 컴파일하고 설치해야 합니다. 이에 관한 절차는 8장에서 자세히 설명하겠습니다.

주)

2.0 커널을 실행하고 있을 경우, 터보리눅스 클러스터 서버를 설치하려면 2.2 계열 커널로 업그레이드해야 합니다. 2.0에서 2.2로 업그레이드하는 것은 아주 중요한 작업이며, 클러스터 서버를 설치하기 전에 이러한 변경 사항에 익숙해야 합니다.

2.4 커널을 실행하고 있다면, 터보리눅스 클러스터 서버 웹 페이지를 방문하여 적절한 커널이 있는지 확인하고 다운로드하시기 바랍니다.

적절한 커널을 선택했으면, 'Proceed'를 클릭하여 설치를 계속 진행하십시오.

11. 다음으로 커널의 어떤 부분을 설치할 것인지 선택할 수 있습니다. 디스크 용량이 부족하지 않다면 기본 설정을 선택하여 모든 추가 요소를 설치하십시오.

그림 3.6 커널 패키지

물론, 기본 커널 패키지와 추가의 커널 유틸리티를 포함시키는 것이 좋습니다. 나중에 커널을 재구축하려면 커널 소스가 필요하고, 시스템에 다른 소프트웨어를 구축하려면 헤더 파일이 필요합니다.

PCMCIA와 iBCS에 대한 지원은 취소해도 무관합니다. PCMCIA는 주로 노트북 컴퓨터에 사용되는 하드웨어 인터페이스로서 서버에서 PCMCIA 지원이 필요한 경우는 거의 없습니다. iBCS 모듈은 Intel Binary Compatibility를 따르는 프로그램을 실행하는 데 사용됩니다. iBCS 모듈을 선택하면 SCO나 기타 인텔 기반의 유닉스 시스템용으로 작성된 포터블 바이너리를 실행할 수 있습니다. 이러한 프로그램을 사용할 것이 아니라면 iBCS 모듈은 필요 없습니다.

설치할 커널 패키지를 선택했으면 'Proceed'를 클릭하십시오. 커널과 추가 모듈이 설치될 것입니다. 이때 걸리는 시간은 1-2분 정도입니다.

12. 커널이 설치되고 나면, 설치자는 사용할 수 있는 관리 툴 목록을 제공합니다. 기본 설정을 선택하여 목록에 있는 모든 패키지를 설치하십시오.

그림 3.7 패키지 설치 메뉴

이 패키지들은 클러스터 서버의 기능 뿐 아니라 여러 가지 관리 툴을 제공합니다. 다음은 이들 패키지가 어떤 일을 하는지 간단히 정리한 것입니다.

- 클러스터 관리 콘솔(Cluster Management Console)은 클러스터를 제어하고 변경할 수 있는 웹 기반의 관리 툴입니다. 이에 관한 자세한 내용은 7장에서 설명하겠습니다.
- 클러스터 서버 데몬은 클러스터 서버 소프트웨어의 핵심 구성요소입니다. 이것이 아직 설치되어 있지 않다면, 체크 표시를 그대로 두십시오.
- 클러스터 에이전트(Cluster Agents: ASA라고도 함)는 클러스터 노드에 있는 다양한 서비스를 제어하는 데 사용됩니다. 이에 관한 자세한 내용은 8장에서 설명하겠습니다. 클러스터 데몬이 클러스터 노드 상의 서비스가 언제 사용 불가능하게 되는지 알아내게 하려면 클러스터 에이전트를 설치해야 합니다.

• TLCS 관리 툴에는 메뉴 위주의 구성 프로그램이 포함되어 있습니다. 모든 구성 작업을 여러분이 직접 수행하는 경우가 아니면, 이 툴을 반드시 설치해야 합니다.

패키지를 선택했으면, 'Install' 을 클릭하십시오. 지정된 패키지가 설치될 것입니다. 이때 선택된 패키지가 필요로 하는 몇 가지 추가 패키지도 함께 설치됩니다. 설치 절차는 1-2분 정도가 걸립니다.

13. 다음으로 CMC 웹 기반의 관리 시스템에 대한 보안 키를 만드는 데 필요한 몇 가지 정보를 입력해야 합니다. 데이터 입력 화면으로 이동하려면 'OK' 를 클릭하십시오.

14. 이 화면에서는 여러분의 조직에 대한 다양한 정보를 입력해야 합니다. 요구되는 정보를 입력한 다음, 'Generate' 버튼을 클릭하면, SSL 인증서가 생성됩니다. 계속 진행하려면 'OK' 를 클릭하십시오.

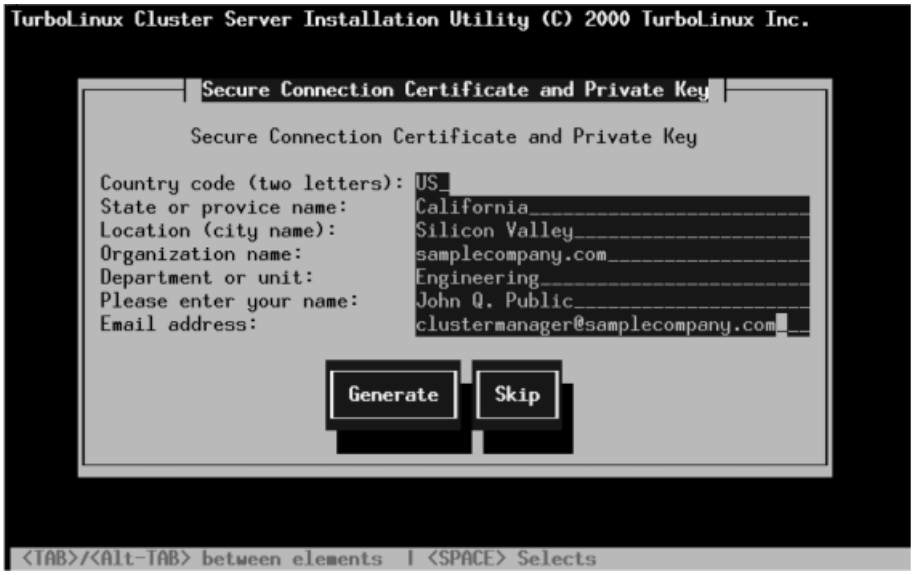

그림 3.8 SSL 인증서를 만드는 데 필요한 정보

15. 다음 단계는 LILO를 구성하는 것입니다. LILO(Linux Loader)는 커널을 로드하고 시스템을 시작하는 프로그램입니다. 앞 단계에서 새로운 커널을 추가했기 때문에, LILO에게 그것을 어떻게 로드할 것인지 여부를 알려주어야 합니다.

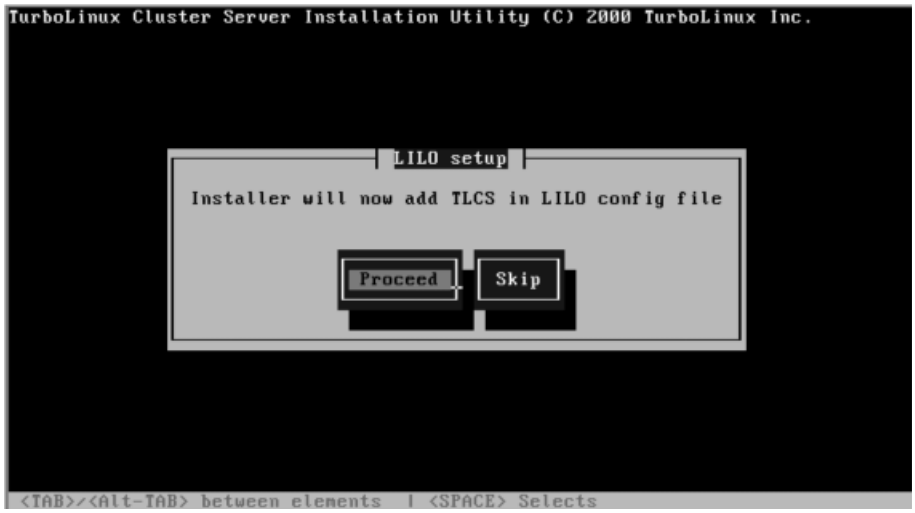

그림 3.9 LILO 설정

'Proceed' 버튼을 클릭하면, LILO 구성 프로그램이 필요한 구성 변경 작업을 처리할 것입니다. 이 변경 작업은 'ClusterServer' 이미지를 부팅 가능한 운영 체제 목록에 추가하고, 그것을 로드할 기본 시스템으로 설정합니다. LILO 프롬프트에서 이미지 이름을 입력하면 여전히 이전 이미지를 부팅할 수 있습니다.

일반적으로, 이전 이미지에는 'linux'라는 제목이 붙습니다. LILO 프롬프트에서 TAB을 누르면 이미지 목록을 볼 수 있습니다.

16. LILO가 작업을 완료하면, 설치 과정도 완료된 것입니다. 'Finish' 버튼을 클릭하십시오.

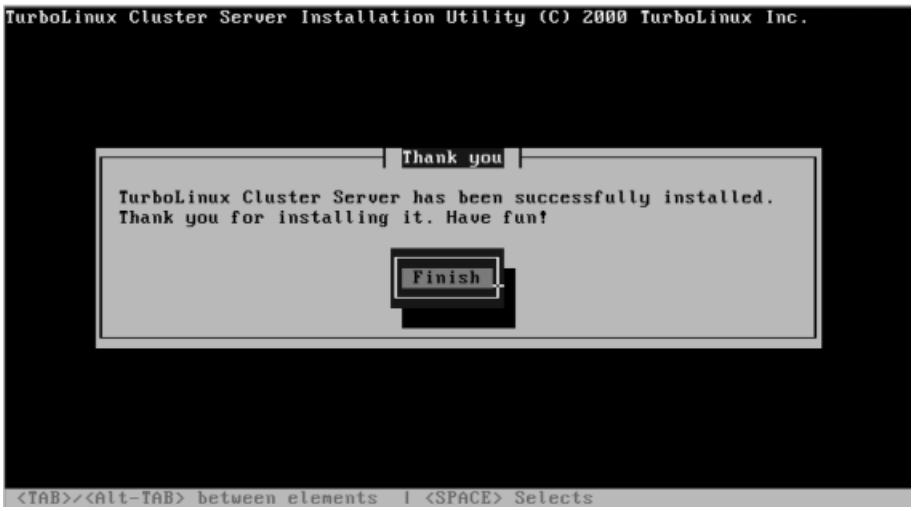

그림 3.10 설치 완료

17. 커널 설치와 LILO 구성 단계를 건너뛰지 않았다면, 시스템을 재부팅해야 새로운 커널이 로드될 것임을 나타내는메시지가 나타날 것입니다.

다른 콘솔에서 저장되지 않은 데이터를 갖고 있는 다른 애플리케이션이 실행되고 있는지 확인하십시오. 'Reboot'를 클릭하거나 ENTER를 눌러 시스템을 재부팅하십시오. 커널을 설치하거나 LILO를 변경하지 않았다면, 쉘 프롬프트로 돌아갑니다.

설치 후

설치 프로그램의 마지막 단계는 시스템을 재부팅하는 것입니다. 클러스터 서버가 실행되기 위해서는 변경된 커널이 몇 가지 기능을 구현해야 하기 때문에 이 단계는 반드시 필요합니다. 새로운 커널은 재부팅해야만 시작할 수 있습니 다. 시스템을 재부팅하라는 메시지가 나타났을 때 ENTER를 누르면 시스템이 안전하게 종료되고 재부팅됩니다.

시스템이 다시 시작되면, LILO는 새로운 선택 사항을 제공합니다.(모든 선택 사항을 보려면 LILO 프롬프트가 나타 났을 때 TAB을 누르십시오). 이제 선택 사항에는 'linux' 이미지와 'ClusterServer' 이미지가 있을 것입니다.

'linux' 이미지는 설치 이전에 사용했던 커널 버전을 로드하고, 'ClusterServer'는 새로운 커널(터보리눅스 클러스터 서버를 실행할 수 있도록 변경된 커널)을 로드합니다.

로드할 이미지를 선택하지 않으면, 기본 설정에 따라 'ClusterServer' 이미지가 로드되어, 클러스터 서버 제품이 실행될 수 있게 합니다.

새로 설치된 클러스터 서버 커널로 시스템을 재부팅한 후에는, 클러스터를 구성해야 합니다. 구성에 관한 자세한 내 용은 다음 장에서 설명하도록 하겠습니다. 클러스터링 소프트웨어는 사용하려면 먼저 구성해야 합니다. 구성은 두 가지 단계로 구성됩니다. 하나는 클러스터를 설계하는 것이고, 다른 하나는 `turboclusteradmin` 프로그램을 통해 세부 사항을 입력하는 것입니다.

설치 문제 해결하기

설치 프로그램은 아주 간단하지만 시스템 설정을 검사하고 적절한 옵션을 선택하는 작업을 훌륭하게 수행합니다. 그러나, 설치 프로그램에서도 발생할 수 있는 문제가 몇 가지 있습니다.
이 절에서는 그러한 문제를 살펴보겠습니다.

설치 파일을 찾을 수 없는 경우

여러분이 첫 번째로 부딪히게 될 문제는 설치 프로그램이 설치를 완료하는 데 필요한 파일을 찾을 수 없는 것입니 다. 이 문제는 설치 프로그램이 들어 있는 디렉토리로 이동하지 않고 설치 프로그램을 실행하려고 할 때 발생합니다. 이 문제는 설치 CD가 들어있는 디렉토리로 가기만 하면 쉽게 해결할 수 있습니다. 예를 들어, 클러스터 서버 설치 CD가 /mnt/cdrom 에 장착되어 있으면, 다음과 같은 명령을 실행해야 합니다.

```
# cd /mnt/cdrom
# ./TLCS-install
```

배포판을 검사할 수 없을 경우

클러스터 서버 설치 프로그램이 여러분이 실행하고 있는 리눅스 버전을 알아내지 못할 수도 있습니다. 이러한 문제 는 컴퓨터에 있는 시스템 파일을 변경했을 때 발생합니다.

설치 프로그램은 여러분의 배포판을 검사할 수 없음을 나 타내는 메시지가 표시합니다.

그림 3.11 알 수 없는 배포판

이러한 문제가 발생하면, 명령어 라인에서 여러분의 배 포판 이름을 지정해야 합니다.

```
# ./TLCS-install turbolinux
```

또는

```
# ./TLCS-install redhat
```

지원되지 않는 배포판에 설치하기

터보리눅스는 터보리눅스 서버와 레드햇만 지원하지만, 그 이외의 다른 리눅스 시스템에도 클러스터 서버를 설치할 수도 있습니다. 설치 방법에는 두 가지가 있습니다.

1. 설치 프로그램에 현재 사용자가 레드햇이나 터보리눅스를 실행하고 있음을 알려 주어야 하는데, 설치 프로그램 시작시 명령어 라인에서 redhat 이나 turbolinux 를 지정하면 됩니다. 이 경우, 설치 프로그램이 강제로 실행되기는 하지만 패키지가 제대로 설치, 작동된다는 보장은 없습니다.

2. 패키지를 수동으로 설치하십시오. CD에서 RPM 패키지를 찾아 각 패키지를 직접 설치하는 것입니다. 이 패키지들은 usr/support 와 usr/tlcs 디렉토리에 들어 있습니다.각 패키지를 설치할 때는 rpm 명령을 사용합니다. 이 방법을 사용하려면, CD-ROM에 대해 usr/TLCS-install/TLCS-GenCert 스크립트를 실행하여 CMC에 대한SSL 인증서를 생성해야 합니다.

어떤 방법을 사용하든, 사용자 정의 커널을 작성해야 합니다. 이에 관한 내용은 8장에서 자세히 설명할 것입니다. 물론, 지원되지 않는 다른 시스템에서 클러스터 서버를 강제로 실행하는 것보다는 지원되는 리눅스 배포판 중 하나 를 사용하는 것이 좋습니다.

다른 배포판을 설치할 시스템이 없다면, 별도로 새 컴퓨터를 구입하시기 바랍니다. 하드웨어와 리눅스 운영 체제에 비용이 들더라도, 지원되는 구성을 사용할 만한 가치가 있습니다.

주)
터보리눅스는 공식적으로 지원되는 배포판 이외의 다른 배포판에 대해 기술적인 지원을 제공하지 않습니다.

제 4 장

구 성

이 장에서는 터보리눅스 구성 툴에 관한 기본적인 내용과 클러스터를 구성하는 방법을 알아보도록 하겠습니다. 클러스터 서버는 다양한 요구를 수용할 수 있는 융통성을 갖고 있기 때문에 표준 구성이나 기본 구성이 없습니다.

각 클러스터는 그것이 어떻게 사용되며 구축에 필요한 자원이 무엇인지에 따라 다양하게 구성할 수 있습니다. 이 장 에서는 구성 툴을 사용하는 방법을 알아보고 작동 가능한 클러스터를 설계하고 구성하는 데 필요한 세부 사항을 설명하도록 하겠습니다.

이 장에서 다룰 주요 내용은 다음과 같습니다.

- 설계 계획
- 구성 툴 개요
- 서비스
- 서버
- 고급 트래픽 관리자
- 클러스터
- 전역 설정

설계 계획

클러스터를 구성하기 전에 설계를 계획하는 것이 매우 중요합니다. 이때 어떤 시스템이 ATM이 되고 어떤 시스템이 노드가 될 것인지를 결정해야 합니다. 또한, 클러스터에서 어떤 서비스를 실행할 것인지도 선택해야 하는데, 물론 이러한 사항은 클러스터 서버 소프트웨어를 구입하기 전에 미리 결정해야 할 것입니다.

클러스터를 설계할 때 도움이 되는 방법은 클러스터의 다이어그램을 그려보는 것입니다. 일반적으로 ATM을 다이어그램의 맨 위에 위치시키고, 그 ATM으로부터 노드가 뻗어 나오게 그립니다.

이것은 트래픽이 클러스터 노드에 도착하기 전에 주 ATM을 반드시 통과해야 함을 나타냅니다.
클러스터 다이어그램에 호스트 이름과 IP 주소를 표시하면 더 명확해집니다.(하나의 시스템이 ATM도 되고 노드도 될 수 있습니다. 클러스터를 개념적으로 나타낼 때는 다이어그램의 맨 위에 이와 같은 이중 시스템을 두는 것이 가장 좋습니다.)

먼저 단순한 클러스터부터 시작해서 필요에 따라 더 복잡한 구성으로 확장해나가는 것이 바람직합니다. 작은 클러스터일수록 문제를 해결하고 디버깅하기가 더 쉽습니다.

작은 클러스터를 설정하고 시험해본 후에, 원하는 구성에 도달할 때까지 ATM과 노드를 계속 추가하면 됩니다. 클러스터 서버는 확장 기능을 위해 이와 같은 융통성을 제공합니다.

일반적인 시나리오

대부분의 클러스터는 웹 서버로 사용할 수 있습니다. 지난 몇 년 사이 웹 트래픽이 엄청나게 증가함에 따라, 많은 사이트가 더 높은 안정성과 가용성, 그리고 확장성을 필요로 하게 되었습니다.

웹 서버가 클러스터링에 많이 사용 되기는 하지만, 다른 서비스도 가능합니다. 클러스터의 설계는 클러스터에서 실행될 특정 서비스보다 성능과 하드 웨어 가용성에 초점을 맞춥니다.

이 장에서는 클러스터를 구성하는 데 필요한 단계를 자세히 알아보겠습니다. 예제로 사용된 클러스터는 기본적인 구성 옵션을 모두 포함하는 중간 정도의 복잡도입니다. 이 클러스터는 사용할 수 있는 옵션을 모두 포함시킨 것이므로 일반적인 구성 방식이라 볼 수는 없습니다.

소규모 클러스터

가장 작은 클러스터는 단 두 개의 시스템으로 구성될 수 있습니다. 한 시스템은 주 ATM과 클러스터 노드의 역할을, 다른 하나는 노드와 백업 ATM 역할을 수행합니다. 각 시스템이 ATM과 노드의 역할을 모두 하게 만들면 사용 가능한 자원을 최대한 활용하는 것입니다.

아래 예에서는, 한 시스템을 ATM으로 구성하고 다른 한 시스템을 노드로 구성하였습니다. 노드는 ATM을 통하지 않고도 응답 트래픽을 전달할 수 있습니다.

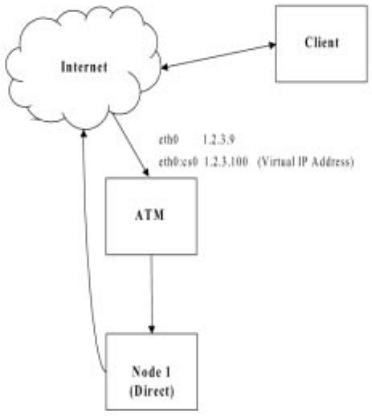

그림 4.1 단일의 ATM과 노드를 갖는 단순형 클러스터

대규모 클러스터

이 클러스터는 하나 혹은 두개의 시스템을 ATM 전용으로 사용하고, 다른 모든 시스템은 클러스터 노드로 사용합니다. 주 ATM은 모든 트래픽 포워딩을 처리하고 다른 일은 하지 않습니다. 백업 ATM은 주 ATM이 다운되지 않는 한 사용되지 않습니다. 클러스터 노드는 서비스만 처리합니다.

이에 대한 예가 아래 다이어그램에 나와 있습니다. Node 4는 ATM과 같은 서브넷 상에 있지 않기 때문에 터널링을 사용해야 합니다.

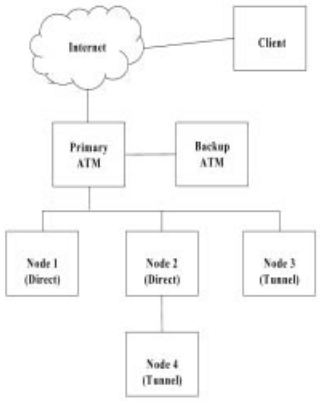

그림 4.2 2개의 전용 ATM과 4개의 노드를 갖는 클러스터

복잡한 클러스터

아주 복잡한 클러스터를 설계할 수도 있습니다. 터보리눅스 클러스터 서버는 여러 개의 클러스터 주소를 정의하고 각각의 가상 클러스터 주소 안에 서로 다른 클러스터 노드 세트를 배치하는 것을 허용합니다. 심지어는 노드를 공유 하면서 서로 다른 ATM 세트를 사용하는 다중 클러스터를 실행할 수도 있습니다.

복잡한 클러스터를 설계할 때는 먼저 본 설명서를 자세히 읽어보시기 바랍니다.

또한, 단순 클러스터를 구성하는 것부터 시작해서 다른 옵션을 고려하도록 하십시오. 처음부터 지나치게 어려운 구성을 시작하는 것은 좋지 않습니다.

구성 툴 개요

클러스터를 구성하는 방법에는 구성 파일을 직접 편집하는 방법과 구성 툴을 사용하는 방법이 있습니다. 사용할 수 있는 모든 옵션에 익숙해지기 전까지는 구성 툴을 사용하는 것이 바람직합니다.

구성 툴은 사용하기 쉬운 메뉴 중심의 인터페이스를 제공하며 각 매개 변수가 사용되는 용도를 설명해 줍니다. 구성 툴은 여러분이 입력한 정보를 받아들여 구성 파일을 생성합니다. 또한, 구성 파일을 구문 검색 하여(또는 파싱하여) 현재 설정을 디스플레이할 수도 있습니다.

이 장에서는 구성 툴에 대해서 알아보고, 다음 장에서는 구성 파일의 형식을 설명하도록 하겠습니다.

클러스터 구성 툴을 액세스하는 방법에는 두 가지가 있습니다. 한 가지 방법은 `tlcsconfig` 프로그램을 사용하여 구성 메뉴를 직접 액세스하는 것이고, 또 다른 방법은 `turboclusteradmin` 프로그램을 사용하는 것입니다.

이 툴을 사용하면 `tlcsconfig` 프로그램은 물론 동기화 툴과 관련된 파일을 열어볼 수 있습니다.
대부분의 경우, 클러스터의 다양한 측면을 관리할 수 있는 `turboclusteradmin` 프로그램을 사용합니다.

turboclusteradmin

`turboclusteradmin` 프로그램은 터보리눅스 클러스터 서버에 사용되는 주요 구성 툴입니다. 이 프로그램을 사용하면 세 가지의 중요한 툴을 액세스할 수 있습니다.

그 세가지는 `tlcsconfig`, `tlcs_content_sync`, 그리고 `tlcs_config_sync`입니다. 이 세가지 툴은 명령어 라인에서 직접 액세스할 수도 있습니다. `turboclusteradmin` 프로그램은 이들 툴은 물론 제품과 함께 제공되는 도큐멘테이션 파일을 액세스할 수 있는 통합적인 구성 툴을 제공합니다.

그림 4.3 turboclusteradmin 메인 메뉴

tlcsconfig

tlcsconfig 프로그램은 실제 구성 작업을 수행하는 데 사용됩니다.
앞에서도 설명한바 있지만, 이 프로그램은 명령어 라인에서 직접 실행할 수도 있고 turboclusteradmin 메인 메뉴에서 'Cluster Server Configuration'을 선택하여 실행할 수도 있습니다. 일단 이 구성 유틸리티로 들어가면 여러분이 구성할 수 있는 다양한 하위 시스템에 대한 메뉴가 제공됩니다. 관련 그림이 아래에 나와 있습니다.

그림 4.4 tlcsconfig 메인 메뉴

여러분이 설정해야 하는 구성 옵션은 아주 많습니다. 이 설정은 관련된 항목이 함께 묶인 다양한 메뉴로 나누어안 있으며, 여러분이 선택할 수 있는 메뉴 옵션은 다음과 같습니다.

- Clustered Services
- Servers Configuration
- Advanced Traffic Managers
- Virtual Servers
- Global Settings

이들 메뉴 옵션 각각은 이 장의 나머지 부분에서 자세히 살펴볼 것입니다. 구성 프로그램에서 IP 주소나 도메인 이름을 입력할 때, 그 주소는 양방향으로 완전히 변환될 수 있어야 합니다. 이것을 가능하게 하는 방법은 두 가지 입니다. 한가지는 주소가 DNS 서버에 구성되어 있는지 확인하는 것이고, 다른 한가지는 /etc/hosts 파일에 주소를 입력하는 것입니다. 또한, 127.0.0.1이 시스템의 호스트 이름과 연관되어 있는지도 확인해야 합니다.

/etc/hosts 파일에서 127.0.0.1과 관련된 호스트 명은 'localhost' 뿐입니다.
다음은 이름이 atm1.turbolinux.usa이고, IP 주소가 192.168.0.1인 시스템에 대한 샘플 /etc/hosts 파일입니다.

```
127.0.0.1       localhost
192.168.0.1     atm1.turbolinux.usa     atm1
192.168.0.2     atm2.turbolinux.usa     atm2
192.168.0.3     node1.turbolinux.usa    node1
192.168.0.4     node2.turbolinux.usa    node2
192.168.0.100   cluster.turbolinux.usa  cluster
```

클러스터 구성을 완료한 후에는 tlcs_config_sync 프로그램을 사용하여 구성 파일을 클러스터 안의 다른 시스템으로 복사해야 합니다. 또한, 클러스터 서버 소프트웨어를 실행하지 않는 다른 클러스터 노드를 구성해야 합니다.

서비스

여러분이 가장 먼저 구성해야 할 옵션은 클러스터에서 실행될 네트워크 서비스입니다. 서비스를 구성하려면, 구성 툴의 메인 메뉴에서 'Clustered Services' 메뉴 항목을 선택해야 합니다. 여기서부터는 서비스 에이전트를 구성할 수도 있고 서비스 자체를 구성할 수도 있습니다. 이 절에서는 각각의 구성에 관해 자세히 살펴보도록 하겠습니다.

에이전트

애플리케이션 안정용 에이전트(ASA)는 클러스터 노드에 있는 특정 서비스의 상태를 감시합니다. 에이전트를 설정하는 방법은 다음과 같습니다.

1. `tlcsconfig` 메인 메뉴에서, 'Clustered Services'를 선택하십시오.

2. 그 다음 메뉴에서 'Application Stability Agents'를 선택하십시오.
   그러면 현재 구성되어 있는 모든 안정용 에이전트 목록이 나타납니다.

그림 4.5 애플리케이션 안정용 에이전트

3. 이 메뉴에서, 새로운 ASA를 추가할 수도 있고 기존의 ASA를 편집하거나 삭제할 수도 있습니다.

4. ASA를 추가하려면 'Add'를 클릭하십시오.

그림 4.6 Application Stability Agent 추가하기

5. 다음과 같은 정보를 입력하십시오.

i. '애플리케이션 안정용 에이전트'은 나중에 ASA를 지칭하는데 사용될 이름입니다. 서비스의 이름이나 그와 비슷한 이름을 사용하는 것이 가장 좋습니다.

ii. 'Check with' 필드는 ASA 서비스 검사를 수행하는 데 사용될 프로그램을 지정합니다. 반드시 프로그램에 대한 전체 경로를 지정해야 합니다.

iii. 'Event triggered when down'과 'Event triggered when up' 필드는 ASA에 의해 서비스가 다운되거나 다시 가동되었음이 감지될 때 실행할 프로그램을 나타냅니다. 대부분의 경우에는 이 필드가 사용되지 않으므로 그냥 공백으로 남겨두면 됩니다. 만약 이 필드를 사용하려면, 전체 경로를 지정해야 합니다.

'down' 스크립트는 서비스를 재시작 해야 하는 경우에 사용합니다. 'down' 스크립트는 서비스를 재가동하기 위해 필요한 모든 작업을 수행할 수 있습니다.

터보리눅스 클러스터 서버는 ASA 검사를 수행할 수 있는 여러 가지 프로그램을 제공하며, 이들 프로그램은 "Agent"로 끝나는 이름을 갖습니다. 에이전트 프로그램의 전체 목록을 보고 싶으면 8장을 참고하시기 바랍니다.

8장은 ASA 프로그램과 up/down 스크립트에 관한 자세한 내용(이들을 호출할 때 사용되는 명령어 라인 인자 포함)을 다루고 있습니다. 필요한 정보를 입력했으면, 'OK'를 클릭하여 ASA 목록으로 돌아가십시오.

4. 필요한 Application Stability Agents를 모두 입력했으면 'Done'을 클릭하십시오.

주)
Application Stability Agents 메뉴에서 구성할 필요가 없이 미리 정의되어 있는 세가지 ASA는 'http', 'connect', 그리고 'none'입니다. 'http' ASA는 HTTP 1.1 서버를 검사하고, 'connect' ASA는 간단한 TCP 연결을 검사합니다. 'none' 설정은 ASA 검사를 모두 건너뜁니다(ASA는 서비스가 항상 활성이라고 간주함.) 여러분의 클러스터는 이들 세 가지 ASA로 충분할 것입니다.

서비스 설정

터보리눅스 클러스터 서버는 서비스 지향적인 클러스터입니다. 클러스터가 서비스에 대한 포트 번호가 무엇이고 들어오는 연결을 어떻게 처리할 것인지 알 수 있도록 각 서비스를 구성해야 합니다.

서비스를 구성하는 방법은 다음과 같습니다.

1. `tlcsconfig` 메인 메뉴에서, 'Clustered Services'를 선택하십시오.

2 'Clustered Services' 메뉴에서 'Service Settings'를 선택하십시오. 그러면 현재 구성되어 있는 모든 서비 스와 함께 각 서비스에 관한 정보가 표시될 것입니다.

그림 4.7 서비스 설정

각 서비스의 이름과 함께 포트 번호와 그것이 TCP 포트인지 UDP 포트인지를 나타내는 정보가 표시됩니다. 서비스가 영속적으로 구성되어 있는지, 또는 부하 조정 대신 fail-over로 구성되어 있는지를 나타내는 플래그도 있을 수 있습니다. 마지막으로, 서비스가 사용하는 ASA의 이름이 표시됩니다.

3. 이 메뉴에서 새로운 서비스를 추가할 수도 있고 기존의 서비스를 편집하거나 삭제할 수도 있습니다.

4. 서비스를 추가하려면, 'Add'를 클릭하십시오.

.그림 4.8 서비스 추가하기

5. 다음과 같은 정보를 입력하십시오.

i. 'Service name' 필드에서 서비스를 지칭할 이름을 입력하십시오. "http"나 "ftp" 같은 표준 이름을 사용하는 것이 좋습니다.

ii. 'Port number' 필드에서, 서비스가 실행될 포트 번호를 입력하십시오. 주어진 서비스가 어떤 포트 번호에서 실행되어야 하는지 모른다면, /etc/services 파일을 확인하십시오.서비스가 반드시 기본 설정된 포트에서 실행될 필요는 없습니다. 예를 들어, HTTP에 대한 기본 설정은 포트 80이지만, 어떤 웹 서버는 포트 8080에서 실행됩니다.

ii. 서비스가 TCP를 사용하는지 UDP를 사용하는 선택하십시오. 대부분의 서비스는 TCP를 사용하지만, DNS는 UDP를 사용합니다. 이 정보는 /etc/services에서 찾아볼 수 있습니다.

iv. 목록에서 'Stability Agent'를 선택하십시오. 여러분이 찾고 있는 ASA가 보이지 않으면 목록을 스크롤해 보십시오. 또한, 앞 절에서 설명한 내용에 따라 ASA가 정의되었는지 확인하십시오.

v. 대부분의 서비스는 그 서비스를 지원하도록 구성된 모든 시스템 간에 작업 부하를 'Load Balance' 하는 것이 좋습니다. 부하 조정 대신 'fail-over' 를 구현하려면 'Failover' 를 선택하십시오.

vi. 마지막으로, 서비스가 세션 영속성을 허용해야 하는지를 ('Allow Session Persistency') 선택하십시오.

이 옵션을 선택하면, 이미 커넥션을 갖고 있는 클라이언트로부터 새로운 커넥션이 들어올 경우, 그 커넥션은 이전의 것 과 같은 서버로 전송됩니다. 이 옵션은 보안 HTTP(HTTPS) 같이 SSL을 사용하는 서버에 매우 유용합니다. 이러한 서비스에 대해 영속성을 지정하지 않으면, 클라이언트는 자신이 액세스하는 각각의 서버로 인증 정보를 전송해야 합니다.

서비스 설정을 입력했으면 'OK' 를 클릭하십시오.

6. 클러스터에서 실행할 모든 서버를 추가했으면 'Done' 을 클릭하십시오.

서버

클러스터에서 어떤 서비스를 실행할 것인지 결정한 후에는, 그 서비스가 실행될 서버를 구성할 수 있습니다. 이 서버는 서비스 요청을 처리하게 될 클러스터 노드입니다. 서버 구성은 두 부분으로 나뉘어집니다.

- 서버 구성
- 서버 그룹 구성

서버 구성

가장 먼저 할 일은 사용될 서버 목록을 만드는 것입니다. 그 방법은 다음과 같습니다.
1. `tlcsconfig` 메인 메뉴에서 'Servers Configuration'을 선택하십시오.

2. 다음 메뉴에서 다시 'Servers Configuration'을 선택하십시오. 그러면, 클러스터 안에서 사용될 모든 클러스터 노드 목록이 나타날 것입니다.

그림 4.9 서버 목록

목록 안의 각 라인은 노드 이름과, 그것이 사용할 포워딩 방법, 그리고 그것이 주기적으로 검사될 것인지를 나타내는 플래그로 구성됩니다.

3. 이 목록에서, 클러스터 노드를 추가할 수도 있고, 기존의 클러스터 노드를 편집하거나 삭제할 수도 있습니다.
4. 클러스터 노드 서버를 추가하려면, 'Add'를 클릭하십시오.
5. 다음과 같은 세 개의 필드와 플래그를 입력하십시오.
   i. 'Server name' 필드는 노드를 지칭할 이름을 지정합니다. 이 필드에는 노드의 호스트 이름을 사용하는 것이 가장 좋습니다.
   ii. 'Full server name or IP'라는 제목이 붙은 라인에는 클러스터 노드의 IP 주소나 정식 도메인 이름을 입력하십시오. 앞에서 설명했듯이, 주소는 양방향으로 해석될 수 있어야 합니다. 주소는 이름으로 지정하는 것이 가장 좋습니다.

ii. 'Forward method'를 선택하십시오. 여러분이 선택할 수 있는 옵션은 'direct', 'tunnel', 그리고 'nat'입니다. 이들 각각에 대해서는 나중에 자세히 설명하겠습니다.

iv. 어떤 이유에서든, ATM이 주기적으로 노드를 검사하지 않게 하려면, 'ping to see if alive'란을 취소하십시오. 일반적으로, ATM이 클러스터에서 다운된 노드를 삭제할 수 있도록 이 옵션을 선택해야 합니다.

그림 4.10 서버 설정

서버 노드에 관한 정보를 입력했으면, 'OK'를 클릭하십시오.

5. 모든 클러스터 노드에 관한 정보를 추가한 후에는 'Done'을 클릭하십시오.

단, 서버 노드가 주 ATM이나 백업 ATM의 역할도 할 수 있다는 점에 주의하십시오.

포워딩 방법

트래픽을 전달하는 세 가지 방법은 직접, 터널링 그리고 NAT입니다. 각 세가지 포워딩 방법 중 한 가지를 사용합니다. 포워딩 방법은 여러 가지 요소(시스템의 위치, 운영 체제, 노드 상에서 수행할 구성 정도)에 의해 결정됩니다.

주 ATM는 특정 포워딩 메커니즘을 사용하도록 구성된 것과는 상관 없이 "local"이라는 직접 포워딩 방법을 사용합니다. 이것은 들어오는 패킷이 로컬로 전달되기 때문입니다.(ATM 시스템 자체를 목적지로 하는 경우 실제로는 어느 것으로도 포워딩될 필요가 없습니다.)

NAT를 제외한 나머지 두 방법을 사용할 때는, 클러스터 노드를 클러스터 안에서 사용할 수 있게 구성해야 합니다. 세부 사항은 어떤 포워딩 방법이 사용되며 노드가 어떤 운영체제에서 실행되느냐에 달려 있습니다. 더 자세한 내용 은 5장을 참고하시기 바랍니다.

직접 포워딩

직접 포워딩은 기본 설정된 포워딩 방법입니다. 이 방법을 사용하면 오버헤드가 가장 적긴 하지만, 클러스터 노드에 대한 몇 가지 설정을 약간 변경할 필요가 있습니다. 클라이언트로부터 들어오는 패킷을 받으면, 주 ATM이 이 패킷을 클러스터 노드로 포워딩합니다. 직접 포워딩에서 패킷은 다른 패킷이 전송되는 것과 마찬가지로 포워딩됩니다.

클러스터 노드 서버는 클라이언트 요청에 응답할 때 클라이언트로 직접 응답을 전달합니다. 반환 트래픽은 ATM을 통과할 필요가 없습니다. 직접 포워딩을 사용하는 클러스터 노드는 트래픽 관리자와 동일한 LAN 세그먼트에 위치 해야 합니다. 직접 포워딩은 노드에서 실행하는 운영 체제에 관계 없이 실행할 수 있습니다.

터널링

클라이언트 노드가 ATM과 다른 LAN에 있을 경우, 터널 포워딩 방법을 사용해야 합니다. 이 방법은 패킷을 캡슐 화하여 ATM과 노드 지점간 터널 연결을 통해 전송합니다. 터널링 방법은 리눅스나 기타 IP-IP 터널링을 지원하는 유닉스 계열을 실행하는 클러스터 노드에만 사용할 수 있습니다.

터널링 방법을 사용하면 캡슐화 과정에서 약간의 오버헤드가 생깁니다. 그러나, 직접 포워딩과 마찬가지로, 응답은 클라이언트로 직접 전달되며 ATM을 통과할 필요가 없습니다.

NAT

NAT는 Network Address Translation의 약자입니다. 이 방법을 사용하면 노드를 특별히 구성하지 않고도 시스템을 클러스터 노드로 사용할 수 있습니다. 즉, NAT를 사용하면 거의 모든 시스템을 클러스터 노드로 사용할 수 있습니다. 하지만, ATM을 구성할 때는 특별한 주의가 필요합니다. NAT는 세 가지 포워딩 방법 중에서 가장 효율성이 떨어집니다.

클라이언트 주소를 변환하는 데 필요한 오버헤드가 더 많으며, 반환 트래픽은 반드시 ATM을 통과해야 합니다. 일반적으로 서비스 응답 패킷은 요청 패킷보다 훨씬 더 많기 때문에 이러한 점은 심각한 병목 현상을 초래할 수도 있습니다. NAT 포워딩에 관한 자세한 내용은 이 장의 끝에 있는 NAT 설정 부분을 참고 하시기 바랍니다.

서버 그룹 구성

서버 그룹을 이용하면 여러 개의 서버 노드를 그룹으로 조합하여 그것을 클러스터에 추가할 수 있습니다. 서버 그룹 은 각 서버 노드가 수행할 서비스도 정의합니다.

각 서버가 모든 서비스를 제공할 필요는 없으므로 어떤 서버가 특정 서비스를 실행할 것인지 선택할 수 있고, 심지어는 몇몇 서버가 더 많은 작업 부하를 부담하도록 구성할 수도 있습니다. 서버 그룹을 서버 풀이라고도 합니다.

서버 그룹을 설정하는 방법은 다음과 같습니다.

1. `tlcsconfig` 메인 메뉴에서, 'Servers Configuration'를 선택하십시오.
2. 다음 메뉴에서 'Server Groups Configuration'을 선택하십시오. 현재 정의되어 있는 서버 풀 목록이 나타날 것입니다. 처음에는 서버 그룹이 없을 것이므로, 여러분이 만들어야 합니다.

3. 새로운 서버 그룹을 만들려면 <Add>를 클릭하십시오.
4. 그러면, 다음과 같은 폼이 나타납니다.

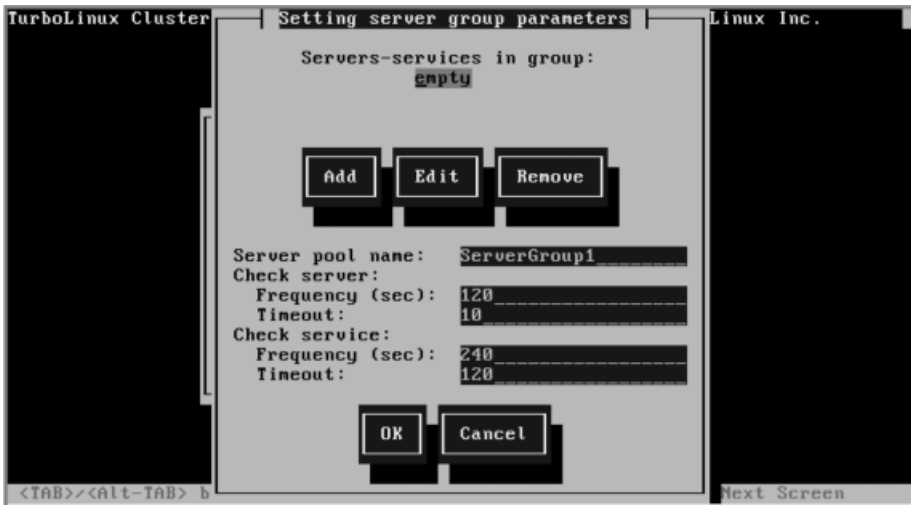

그림 4.11 서버 그룹 설정

5. 데이터 입력 필드로 이동하십시오(서버 풀은 서버 풀 설정을 정의한 후에 추가할 것입니다)
6 .먼저 'Server pool name'란에 서버 그룹의 이름을 지정하십시오.
7. 'Check server' 섹션에는 두 가지 매개 변수 'Frequency'와 'Timeout'이 있습니다.
두 가지 모두 초 단위로 지정합니다. 'Frequency'는 얼마나 자주 서버 노드의 활성 상태 여부를 검사할 것인지 지정 합니다. 'Timeout' 값은 서버 노드가 다운되었다고 간주하기 전에 얼마나 오랫동안 응답을 기다릴 것인지를 나타 냅니다. 'Timeout'은 반드시 'Frequency'보다 짧아야 합니다.
기본 설정을 그대로 사용하십시오.

8. 'Check service' 섹션은 'Check server' 섹션과 비슷합니다. 이 섹션에도 역시 'Frequency'와 'Timeout' 설정이 있습니다. 이 설정은 ATM에게 얼마나 자주 ASA를 실행할 것이며, 개별적인 노드에 있는 서비스가 다운 되었음을 표시하기 전에 얼마나 오랫동안 응답을 기다릴 것인지 알려줍니다. 마찬가지로,'Timeout'은 'Frequency'보다 짧아야 합니다.

9. 일반적으로 서버와 서비스 검사에 대한 기본 값을 그대로 사용해야 합니다. 이 값은 나중에 클러스터의 성능을 최적화할 때 조정하시기 바랍니다. 클러스터 조정에 관한 자세한 내용은 7장을 참고하십시오.

10. 이제 서버 그룹에 대한 매개 변수를 정의했으므로, 몇 가지 서버를 풀에 추가해야 합니다.

11. 폼의 맨 위에서 'Add'를 클릭하십시오.

12. 목록에서 서버를 선택하고 'OK'를 클릭하십시오. 목록에는 여러분이 'Servers Configuration' 메뉴에서 구성했던 모든 서버 노드가 포함되어 있을 것입니다. 여러분이 찾고 있는 노드가 보이지 않으면 목록을 스크롤 해보십 시오.

13. 이제 노드에서 실행할 서비스를 선택하는 창이 열립니다. 창의 왼쪽에는 서비스 목록이 있고 오른쪽에는 해당 서버에서 서비스를 추가 'Add', 편집 'Edit', 삭제 'Remove' 하는 버튼이 있습니다.

14. 이 서버에 대한 서비스를 추가하려면 'Add' 클릭하십시오.

15. 목록에서 서비스를 선택하십시오. 화면에 서비스가 보이지 않으면 목록을 스크롤해보십시오.

16. 서버가 이 서비스를 실행할 때의 'Weight'를 설정하십시오. 일반적으로 모든 서버에 대해 이 값을 1로 설정하면 됩니다. 하지만, 특정 서버가 다른 것보다 더 많은 작업 부하를 처리하게 하려면, 더 높은 가중치를 지정해야 합니다.

17. 'OK'를 클릭하여 서비스를 서버에 추가하십시오.

18. 여러분이 편집하고 있는 서버에 다른 추가의 서비스를 추가한 다음, 'OK'를 클릭하십시오.

19. 다시 'Add'를 클릭하여 다른 서버를 풀에 추가하십시오.
서버 그룹을 모두 구성했으면, 'OK'를 클릭 하십시오.

고급 트래픽 관리자(Advanced Traffic Managers)

서버 노드가 클러스터의 작동에 있어서 중요하긴 하지만, ATM 역시 매우 중요합니다. 주 ATM은 들어오는 트래픽 을 클러스터 노드로 포워딩하는 모든 작업을 처리합니다.

백업 ATM은 주 ATM이 그 작업을 수행할 수 없을 경우 에 대비하여 주 ATM 역할을 대신할 준비를 하고 있습니다.

'Advanced Traffic Managers' 메뉴를 사용하면 시스템이 주 ATM 역할을 할 것인지 백업 ATM 역할을 할 것인지 구성할 수 있습니다. 어떤 것이 주 ATM이 되고 어떤 것이 백업 ATM이 될지를 항상 결정할 수 있는 것은 아 닙니다.

일반적으로, 첫 번째 ATM 시스템이 주 시스템이 되고, 나머지 ATM은 모두 백업이 됩니다.
두 개의 ATM이 동시에 만들어질 경우, 구성 파일 안에 가장 먼저 나와 있는 ATM이 주 ATM이 됩니다.
ATM 구성은 ATM이 될 시스템을 구성하는 것과 ATM 설정을 구성하는 것으로 나누어집니다.

그림 4.12 고급 트래픽 관리자 메뉴(ATM)

고급 트래픽 관리자 시스템

'Advanced Traffic Manager Systems' 메뉴를 사용하면 ATM으로 작동할 컴퓨터를 정의할 수 있습니다. 트래픽 관리자 목록은 구성 프로그램에 있는 다른 목록과 비슷하며, 'Add', 'Edit', 그리고 'Remove' 버튼을 제공합니다. ATM을 시스템을 추가하려면 'Add'를 클릭한 다음, 시스템의 IP 주소나 완전한 호스트명을 입력하십시오.

그림 4.13 Advanced Traffic Manager 목록

한 시스템이 ATM과 클러스터 노드의 역할을 동시에 수행할 수 있음에 주의하십시오. ATM 목록과 서비스 노드 목록은 완전히 별개의 것입니다.

고급 트래픽 관리자 설정

'Advanced Traffic Manager Settings' 메뉴를 선택하면 클러스터 안의 ATM에 대한 몇 가지 매개 변수를 설정 할 수 있습니다. 이 절에서는 이 매개 변수 설정에 관해 알아보겠습니다. 하지만, 여러분의 클러스터를 구성할 때는 일반적으로 기본 설정을 그대로 사용하면 됩니다. 이들 설정을 조정하는 내용은 7장을 참고하시기 바랍니다.

화면의 맨 위에는 ATM 목록이 있을 것입니다. 이 ATM 풀은 서버 풀과 비슷하지만, 현재로서는 하나 이상의 ATM 풀을 만들 수 없다는 점이 다릅니다. 모든 ATM은 자동적으로 단일 풀에 추가됩니다.

그림 4.14 고급 트래픽 관리자 설정

ATM 목록 아래에는 ATM 설정이 있습니다. 각 설정이 어떤 의미를 갖는지 살펴보도록 합시다.

1. 'Send ARP delay (frequency)' 설정은 주 ATM에게 얼마나 자주 ARP 브로드캐스트를 전송할 것인지 알려줍니다. ARP 브로드캐스트는 네트워크 상의 다른 시스템에게 주 ATM이 클러스터의 가상 IP 주소와 연결되어 있음을 알려줍니다.

2. 'HeartBeat Frequency'는 heartbeat 브로드캐스트 사이의 초 간격입니다. 주 ATM은 heartbeats를 생성하여 백업 ATM에게 자신이 작동되고 있음을 알려줍니다.

3. 백업 ATM은 주 ATM의 heartbeat 브로드캐스트를 기다립니다.
백업 ATM은 'Max. missed heartbeats'를 연속으로 놓치면 주 시스템이 다운된 것으로 간주합니다.

4. 'Number of services', 'Number of servers', 그리고 'Number of connections' 값은 커널 테이블의 크기를 지정합니다. 특히 많은 트래픽이 예상되거나 클러스터 안에 매우 많은 서버가 있는 경우가 아니라면, 기본설정을 사용해도 됩니다.

이 값은 서버가 실행되는 데 필요한 최대값보다 커야 합니다. 'Number of services'설정은 말 그대로 얼마나 많은 서비스가 정의되어 있는지 나타냅니다. (구성 프로그램의 'Service Settings'목록에 나와 있는 대로) 'Number of servers'는 사실상 각 서버가 처리하는 서버의 개수와 서버 노드의 개수를 곱한 값입니다.

'Number of connections'는 가장 중요한 값입니다. 가동중인 동시 커넥션의 수가 이 값을 초과하면, 커넥션의 수가 이 값보다 적어질 때까지 그 이상의 커넥션 시도는 실패합니다. 대부분의 경우, 기본 설정 값을 그대로 사용하면 됩니다. 이들 매개 변수에 가장 적합한 값을 설정하려면, 7장을 참고하시기 바랍니다.

5. 'Connection timeout' 설정은 커넥션이 닫히거나 패킷을 전송하지 않은 이후로 커넥션 테이블 안에 그 커넥션을얼마동안 남겨둘 것인지 알려줍니다. 이 값은 초 단위로 설정되며, 거의 모든 서비스는 기본 값 300이면 적당합니다.

커넥션 시간이 더 짧을 것으로 예상되는 한 두개의 서비스만 실행할 경우, 이 값을 조정하는 것이 좋습니다. 클러스터를 최적화하는 내용은 7장을 참조하시기 바랍니다.

클러스터

클러스터가 실행할 서비스와 클러스터 안에서 다양한 역할을 하게 될 시스템을 모두 설정한 다음에는, 클러스터 자체를 설정할 수 있습니다.

각 클러스터는 자신이 관리하고 트래픽을 입출력 지정하는 IP 주소로 정의됩니다. 대부분의 터보리눅스 클러스터 서버는 이러한 가상 IP 주소를 하나만 갖게 되지만, 여러 개의 클러스터 주소를 관리 할 수도 있습니다.

클러스터의 IP 주소를 가상 서버 주소라고도 하는데, 이는 클러스터가 실제로는 여러 개의 시스템으로 구성되어 있지만 외부에서 볼 때는 단일 서버로 보이기 때문입니다. 이 IP 주소가 가상으로 간주되는 것은 클러스터 안의 각 시스템이 실제 IP 주소를 따로 갖기 때문입니다.

클러스터를 설정하려면, `tlcsconfig` 메인 메뉴에서 'Virtual Servers'를 선택하고, 다음과 같은 절차를 따르십시오.

1. 'Virtual Servers' 메뉴는 현재 구성되어 있는 모든 가상 서버 목록을 제공합니다. 여기에도 역시 'Add' 'Edit', 그리고 'Remove' 버튼이 있습니다.

그림 4.15 가상 서버 목록

2. 클러스터에 대한 가상 IP 주소를 추가하려면, 'Add'를 클릭하십시오.

3. 이제 가상 서버를 구성하는 데 필요한 정보를 입력하십시오. 여러분이 입력해야 할 정보는 다음과 같습니다.

i. 'Virtual hostname or IP' 필드에서 가상 IP 주소에 해당하는 IP 주소나 호스트 이름을 입력하십시오. 이것은 외부에서 클러스터를 액세스할 때 사용되는 IP 주소입니다. 다른 주소와 마찬가지로, 이 주소는 양방향으로 해석될 수 있어야 합니다.

ii. 'Send e-mail alerts to' 필드에는 여러분의 전자우편 주소를 입력하십시오. ATM, 서버 노드, 또는 서비스가 다운되거나, 다른 치명적인 오류가 발생할 때마다 이 주소로 경고 메시지가 전송될 것입니다. 이 필드를 공백으로 남겨두면, 전자우편 메시지가 전송되지 않으므로 로그 파일을 검사하지 않는 한 클러스터에 문제가 생겨도 알도리가 없습니다.

ii. 목록에서 'Server pool name'을 선택하십시오. 이것은 여러분이 'Server Groups' 섹션에서 정의했던 서버 그룹입니다.

그림 4.16 가상 서버 설정

클러스터에 대한 적절한 정보를 모두 입력했으면 'OK'를 클릭하십시오.

4. 클러스터에 대한 가상 IP 주소를 모두 구성했으면 'Done'을 클릭하십시오.(일반적으로 하나의 가상 서버를 구성함) 이제 클러스터를 성공적으로 구성했으므로, 모든 클러스터에 전역으로 적용되는 몇 가지 매개 변수를 정의해야 합니다.

전역 설정

ATM에서 실행되는 모든 클러스터에 전역으로 적용되는 설정은 여러 가지가 있습니다. 매개변수는 구성 프로그램의 'Global Settings' 섹션에서 설정할 수 있습니다. 다음은 'Global Settings' 메뉴에서 구성할 수 있는 전역 설정입니다.

- 보안 설정
- 네트워크 설정
- NAT 설정

보안 설정

보안 설정은 어떤 컴퓨터가 클러스터의 원격 관리 기능을 액세스할 수 있는지 결정합니다. 이것은 /etc/hosts.allow와 /etc/hosts.deny 파일에 구성되어 있는 TCP 랩퍼와 매우 비슷한 역할을 합니다. 일반적으로, 이 기능은 로컬 네트워크(특히 여러분이 관리 작업을 수행할 시스템)로 제한하는 것이 좋습니다.

클러스터 관리 기능에 대한 액세스를 제한하는 방법은 다음과 같습니다.

1. `tlcsconfig` 메인 메뉴에서 'Global Settings'를 선택하십시오.

2. 'Global Settings' 메뉴에서 'Security Settings'을 선택하십시오. 그러면, 현재 시행중인 보안 규칙 목록이 나타날 것입니다.

그림 4.17 전역 보안 규칙

3. 새로운 규칙을 만들려면 'Add'를 클릭하십시오.
4. 액세스 권한을 허용하거나 제한할 네트워크나 호스트의 IP 주소를 입력하십시오.
5. 네트워크 마스크를 입력하십시오. 단일 호스트의 경우, 255.255.255.255를 입력하면 됩니다. 전체 네트워크를 허용하거나 거부하려면, 그 네트워크에 대한 서브넷 마스크를 입력하십시오.
6. 이 호스트나 네트워크가 ATM을 관리할 수 있게 허용하려면 'allow'를 선택하십시오. 관리 기능을 액세스하지 못하게 하려면 'deny'를 선택하십시오. 원하는 대로 설정을 입력한 후에 'OK'를 클릭하십시오.
7. 각 규칙을 선택하고 'Up'/'Down' 버튼을 사용하여 규칙을 재배열할 수도 있습니다.
8. 원하는 대로 모든 보안 규칙을 구성했으면 'Done'을 클릭하십시오.

항상 주소 127.0.0.1 (localhost)에 대해서는 Allow 규칙을 설정하고, 다른 주소에 대해서는 Deny 규칙을 설정해야 합니다. 이렇게 하면 CMC 데몬만이 매개 변수를 변경할 수 있게 됩니다. 이렇게 설정된 보안 규칙 목록은 다음과 같을 것입니다.

```
127.0.0.1,255.255.255.255,allowed
0.0.0.0,0.0.0.0,denied
```

네트워크 설정

전역 'Network Settings'메뉴에서는 단 하나의 매개변수만 변경할 수 있습니다. 바로 서브넷 마스크입니다. 이 매개 변수는 모든 ATM이 포함되어 있는 물리적 서브넷과 관련됩니다. 대부분의 경우, 이 값은 255.255.255.0이지만, 네트워크에 따라 다른 값을 사용할 수도 있습니다. 네트워크 관리자에게 문의하거나, ATM 시스템 중 하나의 서브넷 마스크를 찾아보십시오.

그림 4.18 네트워크 마스크

NAT 설정

클러스터 노드로 트래픽을 전달하는 데 사용할 수 있는 포워딩 방법 중 하나는 NAT(Network Address Translation)입니다. NAT의 장점은 노드 자체에 대한 설정을 변경할 필요가 없다는 것입니다. 하지만, ATM에 대한 몇 가지 매개 변수는 설정해야 합니다. 이 작업은 주로 'Global Settings' 메뉴의 'NAT Settings' 섹션에서 수행합니다.

NAT 포워딩을 사용할 경우, ATM 컴퓨터에는 두 개의 네트워크 카드를 설치해야 합니다.
하나는 인터넷에(간접적으로) 연결되고, 다른 하나는 내부 서브넷에 연결됩니다.
이 내부 서브넷은 주 ATM의 Network Address Translation을 통해서만 액세스할 수 있습니다.

NAT 설정은 세 가지 뿐이지만, 그 작동 원리를 이해하는 것은 약간 어렵습니다.

1.'NAT Subnet'은 여러분의 네트워크에 존재하지 않는 네트워크 주소 영역으로 설정해야 합니다. 이 주소 영역은 주소 변환을 수행하는 데 사용됩니다. 들어오는 클라이언트 연결은 NAT에 의해 변환되고 NAT-포워딩을 사용하는 클러스터 노드로 전송된 후에 이 주소 영역으로부터 들어오는 것처럼 보입니다.

서브넷 주소는 10.0.0.0으로 설정하는 것이 바람직하지만, 이 주소 영역에 존재하는 시스템이 있으면 안됩니다. 이 주소는 개인용으로 예약된 주소 영역 중 하나입니다. 이 주소 이외에 예약된 네트워크 주소는 172.16.0.0과 192.168.0.0입니다. 실제 시스템이 이러한 주소로 구성하지 않게 하십시오.

주)
'NAT Subnet'에 대해 어떤 주소 영역을 선택하든, 네트워크 상의 어떤 부분에서도 이 주소 영역을 사용하면 안됩니다.
이 주소는 NAT에 의해 내부적으로 사용됩니다. 실제 네트워크 자원에 이 주소를 사용하면, 그 자원을 액세스할 수 없을 것입니다.

---

2. NAT Subnet Mask'는 'NAT Subnet' 설정과 일치합니다. 하지만, NAT 서브넷은 클라이언트 커넥션을 매핑하는 데만 사용되고 다른 물리적 시스템에는 사용되지 않기 때문에, 실제로 ATM에게 얼마나 많은 클라이언트 커넥션을 준비해야 하는지 알려주는 것은 서브넷 마스크입니다.
일반적인 클러스터의 경우, 255.255.0.0를 사용하면 65,000 개 이상의 커넥션을 허용할 수 있습니다.
더 많은 트래픽이 요구되면, NAT 서브넷 마스크에 있는 비트 수를 줄여서 커넥션 개수를 늘릴 수 있습니다.
여러분이 사용할 수 있는 최대 값은 255.0.0.0이며, 이 값은 1600만 개의 커넥션을 제공할 수 있습니다.
(이 값은 서브넷 10.0.0.0에 해당됨. 서브넷이 172.16.0.0인 경우 최대 값은 255.240.0.0이고,
192.168.0.0인 경우 255.255.0.0) 하지만, 각 커넥션은 커널 안에서 어느 정도의 메모리 바이트를
요구하기 때문에, 시스템 메모리가 아주 많지 않다면, 1600만개의 항목을 예약할 수 없을 것입니다.
더 실질적인 값은 255.240.0.0이며, 이 값은 100만개 이상의 가상 커넥션을 제공할 수 있습니다.

3. NAT를 사용할 경우, 클라이언트로 반환되는 패킷은 반드시 ATM을 통해 반환되어야 합니다. 이것은 NAT 포워딩을 사용하는 노드에 대한 기본 게이트웨이를 설정함으로써 이루어집니다. 주 ATM에 있는 네트워크 카드 중 하나의 IP 주소를 사용할 수도 있지만, 주 ATM이 다운되어 백업 ATM이 그 역할을 대신 수행해야 할 경우 문제가 생길 것입니다. 따라서, 클러스터 자체에 대한 가상 IP 주소 이외에도, 이 게이트웨이에 대한 가상 IP 주소를 할당해야 합니다.
클러스터의 가상 주소는 클라이언트 액세스에 사용되지만, NAT 게이트웨이 가상 주소는 클러스터 노드로부터 다시 패킷을 전송하는 데 사용됩니다. 노드는 클러스터의 가상 IP 주소와 다른 물리적 서브넷에 있기 때문에, 그 주소를 기본 게이트웨이로 사용할 수 없습니다. 따라서, NAT 게이트웨이에 대한 또 다른 가상 IP 주소를 할당해야 합니다.

NAT를 구현할 때, ATM은 두 개의 네트워크 카드를 갖고 있어야 합니다. 하나는 NAT 노드의 "내부" 네트워크에 연결되고, 다른 하나는 외부의 클라이언트가 액세스할 수 있는 인터페이스입니다.
NAT 게이트웨이 주소를 선택할 때는, 사용되고 있지 않은 "내부" 서브넷의 IP 주소를 선택하십시오.
기존의 라우터를 교체할 경우, 그 라우터가 갖고 있던 내부 IP 주소를 사용하면 됩니다. 이러한 방식으로, NAT 포워딩을 사용하는 노드는 재구성할 필요가 없습니다.

사용 가능한 주소를 선택했으면, 그것을 'Gateway to NAT Subnet' 필드에 입력하십시오.

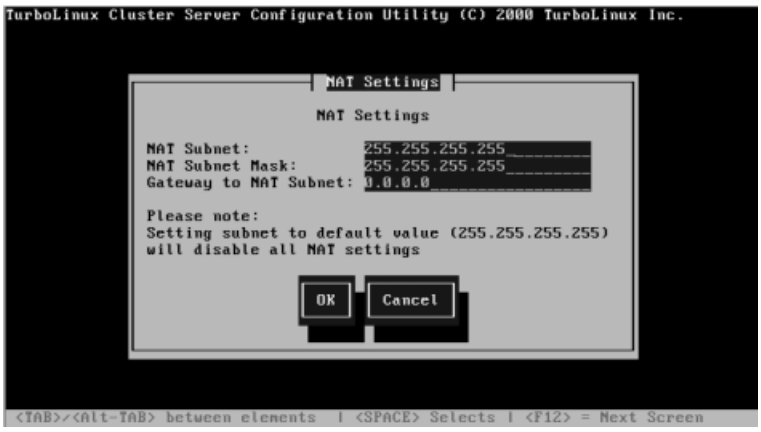

그림 4.19 NAT 설정

NAT 포워딩 방법을 사용하는 노드가 없다면 기본값을 그대로 사용해도 됩니다. 이 경우, NAT 서브넷과 서브넷 마스크는 255.255.255.255가 되고, 게이트웨이는 0.0.0.0가 됩니다.

다음 그림은 NAT를 사용하는 클러스터의 예입니다. 클러스터의 가상 IP 주소는 1.2.3.100이고, NAT Gateway는 192.168.0.100으로 설정되었습니다. eth0과 eth1은 서로 반대편에 있을 수도 있는데, 즉 데몬은 NAT Gateway를 이미 "내부" NAT 서브넷에 있는 인터페이스로 지정합니다.

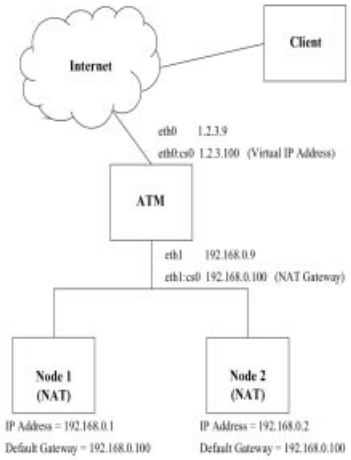

그림 4.20 NAT 포워딩의 예

주의)
NAT를 부적절하게 구성하면 부적절한 네트워크 트래픽이 증가할 수 있으며, ATM이 일부 네트워크 자원을 액세스하지 못할 수도 있습니다.

NAT 구성 작업이 약간 복잡하기는 하지만, 클러스터 노드를 특별히 구성할 필요가 없다는 장점이 있습니다. 노드가 ATM에 대해 사용하고 있던 기본 게이트웨이(라우터)는 간단하게 교체할 수 있습니다. ATM이 기본 게이트웨이의 이전 IP 주소를 ATM의 NAT 게이트웨이 주소로 사용하게 구성하면 됩니다.
처음부터 클러스터를 구축할 경우, 시스템이 기본 게이트웨이를 구성하라고 요구할 때 NAT 게이트웨이 주소를 입력하면 됩니다.

NAT 노드의 IP 주소와 기본 게이트웨이를 DHCP를 통해 할당할 수도 있습니다.
NAT 서브넷에 대한 DHCP 서버를 설정하고, 노드에 IP 주소를 할당하면 됩니다.
노드가 서로 다른 역할을 수행한다면, 그 시스템에 항상 같은 IP 주소를 지정해야 합니다.
모든 클러스터 노드가 같은 서비스를 처리하고, 할당된 모든 주소가 클러스터 구성 파일 안에 구성되어 있다면,
각 시스템이 어떤 IP 주소를 갖는지는 중요하지 않습니다.

제 5 장

클러스터 노드 구성

클러스터 안에 있는 ATM은 클러스터 서버 데몬을 실행하는 리눅스 컴퓨터이어야 하지만, 터보리눅스 클러스터 서버는 사실상 어떤 시스템이든 클러스터 노드로 사용하는 것이 가능합니다. 단, 그 시스템이 TCP/IP 서비스를 제공하기만 하면 됩니다. 하지만, 리눅스 시스템을 클러스터 노드로 사용하면 훨씬 더 많은 융통성을 얻을 수 있으며 클러스터 관리도 더 간단합니다.

트래픽 관리자로부터 클러스터 노드로 패킷을 전달하는 데 사용되는 포워딩 방법은 세 가지 입니다. 가장 간단한 방법은 NAT 방법입니다(클러스터 노드를 특별히 구성할 필요가 없고, 적절한 IP 주소와 네트워크 마스크, 기본 게이트웨이만 설정하면 됨) 직접 포워딩과 터널링 방법을 사용할 때는 클러스터 노드를 특별히 설정해야 합니다.

NAT를 사용할 때는 특별한 설정이 필요 없기 때문에, 이 장에서는 직접 포워딩과 터널링 방법을 사용하여 클러스 터 노드를 구성하는 것만 설명하도록 하겠습니다. 터널링은 리눅스와 유닉스 클러스터 노드에만 사용할 수 있으므로 Windows NT, Windows 2000, 기타 시스템에 관한 절에서는 직접 라우팅 방법을 사용한 구성만 다룰 것입니다.

이 장에서 설명하는 단계 이외에도, 클러스터의 용도에 따라 특정 네트워크 서비스를 설정해야 합니다. 이 내용은 여러분이 어떤 서비스를 클러스터링하고 어떤 제품을 사용하는지에 따라 달라지므로 본 설명서에서는 다루지 않겠 습니다. 하지만, 구성은 클러스터링 되지 않은 독립형 환경에서 서비스를 실행할 때와 동일합니다. 단, 클러스터가 가상 IP 주소에 대한 서비스 요청을 받아들이도록 구성하면 됩니다.

다음 절에서는 다음의 운영 체제에서 클러스터를 구성하는 방법을 알아보겠습니다.

- 리눅스와 유닉스
- Windows NT
- Windows 2000
- 기타 다른 시스템

리눅스 또는 유닉스 클러스터 노드 구성

리눅스 시스템을 클러스터로 구성하는 방법은 두 가지(수동과 자동)입니다. 가장 쉬운 방법은 컴퓨터에 클러스터 서버를 설치하고 실행하는 것입니다. 그러면 시스템이 자동적으로 서버 노드로 구성됩니다.

이 방법을 사용하면, 그 구성과 클러스터의 나머지 부분을 쉽게 동기화할 수 있습니다. 클러스터 서버 데몬은 클러스터 노드에 필요한 모든 구성 작업을 수행합니다. 이 방법은 시스템이 ATM(주 시스템 또는 백업)과 클러스터 노드의 역할을 동시에 수행하게 할 수 있는 유일한 방법입니다.

터보리눅스 클러스터 서버 데몬은 터보리눅스와 레드햇 배포판에서만 실행되므로 다른 배포판에서는 수동 구성 방법을 사용해야 합니다. 또한, 유닉스 서버는 수동으로 설정해야 합니다.

클러스터 노드가 클러스터의 가상 IP 주소로 어드레싱된 요청에 응답하게 하려면, 네트워크 인터페이스 별칭을 설정해야 합니다. 이를 위해서는, IP Aliasing을 커널 내에서 구성해야 하는데, 대부분의 리눅스 배포판은 이를 이미 기본 커널 내에서 컴파일해 놓았을 것입니다.

별칭은 단순히 동일한 네트워크 인터페이스에 추가된 또 하나의 IP 주소입니다. 여러분은 네트워크 인터페이스가 주 ATM으로부터 포워딩된 패킷 수신시 응답할 수 있도록설정해야 합니다.

ATM은 패킷이 클러스터 노드로 포워딩될 때 그 패킷의 대상 주소를 변경하지 않기 때문에 여러분이 클러스터 노드의 별칭 IP 주소를 클러스터 자체의 IP 주소로 설정해야 합니다.(이것을 클러스터의 가상 IP 주소라고도 합니다)

별칭을 만들려면, ifconfig 프로그램을 사용하십시오. 여러분의 실제 IP 주소가 10.0.0.3이라고 가정하고, 클러스터의 가상 IP 주소가 10.0.0.99이라고 가정합시다. eth0 네트워크 인터페이스에 대한 설정을 디스플레이 하면 다음과 같을 것입니다.

```
# ifconfig eth0.
```

```
eth0 Link encap:Ethernet  HWaddr 00:AB:CD:12:12:3F
   IPaddr:10.0.0.3 Bcast:10.0.0.255 Mask:255.255.255.0
   UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
   RX packets:608 errors:0 dropped:0 overruns:0 frame:0
   TX packets:992 errors:1 dropped:0 overruns:0
   carrier:0 collisions:15 txqueuelen:100
   Interrupt:10 Base address:0x210
```

별칭을 추가하려면 다음과 같은 명령을 실행하십시오.

```
# ifconfig eth0:1 10.0.0.99 up
```

'eth0'의 끝에 붙은 ':1'는 유닉스 및 리눅스가 동일한 물리적 네트워크 카드에 대한 별칭 주소임을 나타낼 때 사용하는 구문입니다. 별칭 부분에는 임의의 -4문자를 사용할 수 있으며, 문자는 실제로는 아무런 의미도 없습니다.

별칭을 만들 때 한 가지 문제가 발생할 수 있습니다. 네트워크 상의 다른 시스템이 동일한 서브넷 상의 IP 주소 10.0.0.99로 패킷을 보내고자 할 때, 어떤 컴퓨터가 그 IP 주소를 갖고 있는지 검사하기 위해 ARP 브로드캐스트를 전송하고자 하는 경우입니다. 그 IP 주소를 갖는 컴퓨터는 자신의 IP 주소와 해당 MAC(하드웨어) 주소로 다시 응답하게 됩니다.

하지만, 클러스터 안의 모든 노드가 같은 IP 주소를 갖고 있다면, 모든 노드가 브로드캐스트 ARP 메시지에 응답하려 할 것입니다. 따라서, 주 ATM을 제외한 모든 시스템에 그러한 ARP 요청에 응답하지 말라고 지시해야 합니다. 또한 클러스터를 목적지로 하는 모든 트래픽이 먼저 주 ATM을 통과하도록 해야 합니다.

이를 위한 해결책은 이더넷 인터페이스 대신 루프백 인터페이스에 대한 별칭을 만드는 것입니다. 루프백 인터페이스는 그것과 관련된 하드웨어나 물리적 네트워크가 없는 네트워크 인터페이스입니다. eth0:1에 대한 별칭을 만드는 대신, 다음과 같은 명령을 사용하여 루프백 인터페이스(lo)에 대한 별칭을 만드십시오.

```
# ifconfig lo:1 10.0.0.99 netmask 255.255.255.255 up
```

그 다음, 이 인터페이스에 대한 ARP 응답을 해제해야 합니다. 이 작업을 어떻게 수행하는지는 어떤 리눅스 커널 버전을 사용하느냐에 따라 다릅니다. 유닉스 시스템과 리눅스 2.0 커널의 경우, 인터페이스를 활성화 시킬 때 ifconfig 명령에 -arp 옵션을 지정하면 됩니다.(어떤 유닉스 및 리눅스 시스템은 약간 다른 구문을 사용할 수도 있음, 가령 -arp 대신 noarp 를 사용) 여기서는 다음과 같은 명령을 사용하여 인터페이스를 구성하겠습니다.

```
# ifconfig lo:1 10.0.0.99 netmask 255.255.255.255 -arp
```

주)
이 방법을 사용하려면, ifconfig 명령을 실행하여 구성을 디스플레이할 때 NOARP 플래그를 설정해야 합니다. 그렇지 않으면 ARP 응답이 여전히 전달되고, 클러스터가 제대로 작동되지 않을 것입니다.

불행하게도 2.0 계열 이상의 리눅스 커널 버전에서는 이 방법을 사용할 수 없습니다. 커널 2.2.14 이상을 실행하는 시스템의 경우, -arp 옵션이 작동하지 않습니다. 그 대신, /proc 파일 시스템을 사용하여 ARP 응답을 해제해야 합니다.

/proc/sys/net/ipv4/conf/all에 있는 숨겨진 파일과 여러분이 사용하고 있는 인터페이스에 대한 숨겨진 파일에 1을 작성하십시오. 다음은 루프백 인터페이스에 대한 ARP 응답을 취소한 예입니다.

```
# echo 1 > /proc/sys/net/ipv4/conf/all/hidden
# echo 1 > /proc/sys/net/ipv4/conf/lo/hidden
```

터널링 클러스터 노드

터널 포워딩 방법은 유닉스나 리눅스 운영 체제를 실행하는 클러스터 노드에만 사용할 수 있습니다. 노드에 있는 커널은 IP-IP 터널링으로 구성해야 합니다. 모듈을 사용하는 리눅스 커널의 경우, 다음과 같은 명령을 실행하여 모듈을 로드해야 합니다.

```
# insmod ipip
```

일단 IP-IP 지원이 설정되면, IP-IP 터널 인터페이스를 활성화할 수 있으며, 이 인터페이스는 tunl0. 이라는 이름을 갖습니다. 다음과 같이 클러스터의 가상 IP 주소를 지정하여 ifconfig 명령을 실행하십시오.

```
# ifconfig tunl0 10.0.0.99 netmask 255.255.255.255 up
```

-arp 옵션을 추가하거나 /proc 안의 숨겨진 파일에 작성하십시오.

```
# echo 1 > /proc/sys/net/ipv4/conf/all/hidden
# echo 1 > /proc/sys/net/ipv4/conf/tunl0/hidden
```

이 터널 인터페이스를 설정하고 ATM이 터널링 방법을 사용하는 노드로 트래픽을 포워딩하도록 구성되면, 클러스터 노드를 사용할 준비가 된 것입니다.

Windows NT 클러스터 노드 구성

Windows NT 4.0 시스템을 클러스터 노드로 구성하려면, 마이크로소프트 루프백 장치(Microsoft Loopback Device)를 설치해야 합니다. 먼저 'Start' 메뉴로 가서, 'Settings' 메뉴에 있는 'Control Panel' 선택하여 Network 제어판을 열어야 합니다. 'Network' 제어판에서 'Adapters' 탭을 클릭한 다음, 'Add'를 클릭하고 'MS Loopback Adapter'를 선택하십시오(그림 5.1 참고). 'OK'를 클릭하여 기본 설정된 프레임 타입을 선택하고 설치 파일이 어디에 있는지 지정하십시오.

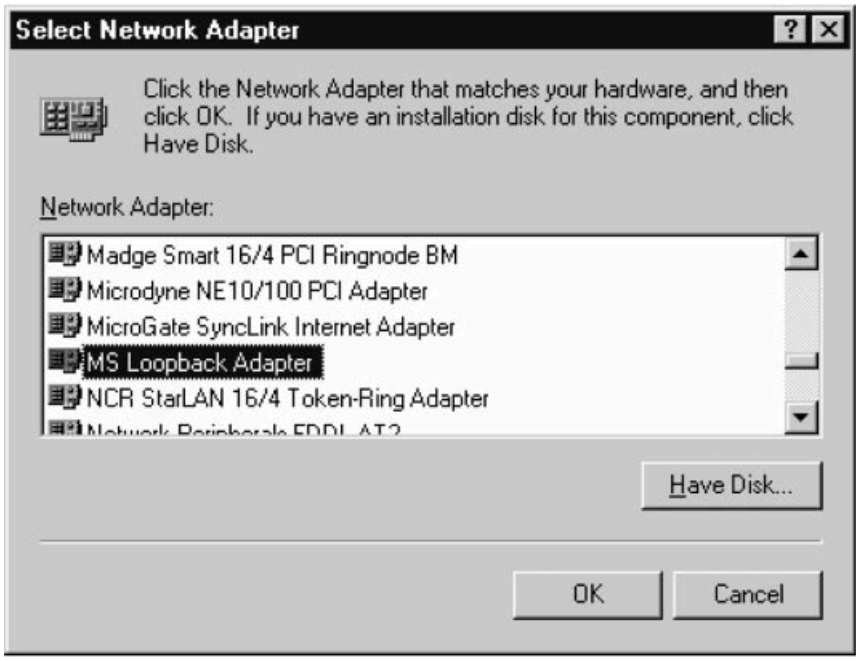

그림 5.1 Windows NT에서 루프백 어댑터 추가하기

새로운 어댑터를 추가하려면 'Close'를 클릭하십시오. 그러면 TCP/IP 속성 페이지가 나타날 것입니다. 드롭 다운 목록에서 'MS Loopback Adapter'를 선택하십시오.

IP 주소를 클러스터의 가상 IP 주소로 설정하십시오. 또한, 기본 게이트웨이도 설정하십시오.
서브넷 마스크는 255.255.255.255로 설정해야 하지만, 대화창이 그 값을 받아들이지 않을 것입니다.
일단은 255.255.255.128로 설정하십시오. 이 값은 레지스터리에서 255.255.255.255로 변경해야 합니다.
Properties 대화창에서 'OK'를 클릭하고, 재부팅할 것인지를 물으면 'No'를 클릭하십시오.

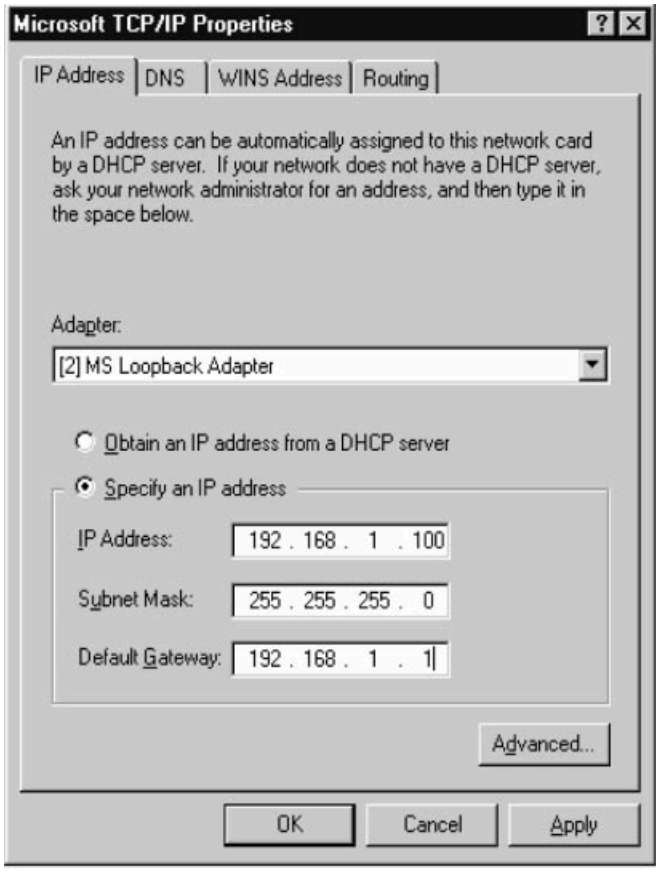

그림 5.2 Windows NT에서 루프백 어댑터 구성하기

이제 레지스터리로 가서 서브넷 주소를 적절하게 설정해야 합니다.
'Start' 메뉴에서, 'Run'을 선택하고 REGEDIT를 입력한 다음, 'OK'를 클릭하십시오.

왼쪽에서 HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\NDISLoop2\Parameters\Tcpip를 선택하고, 오른쪽에 있는 'SubnetMask' 항목을 두 번 클릭하십시오. 128을 255로 변경하십시오.(오른쪽에 서 128을 삭제하고 255를 입력하면 됨) 다른 것은 변경하지 않도록 주의하십시오. 'OK'를 클릭한 다음 레지스터리 편집기를 종료하십시오. 변경 사항이 적용되게 하려면 재부팅하십시오.

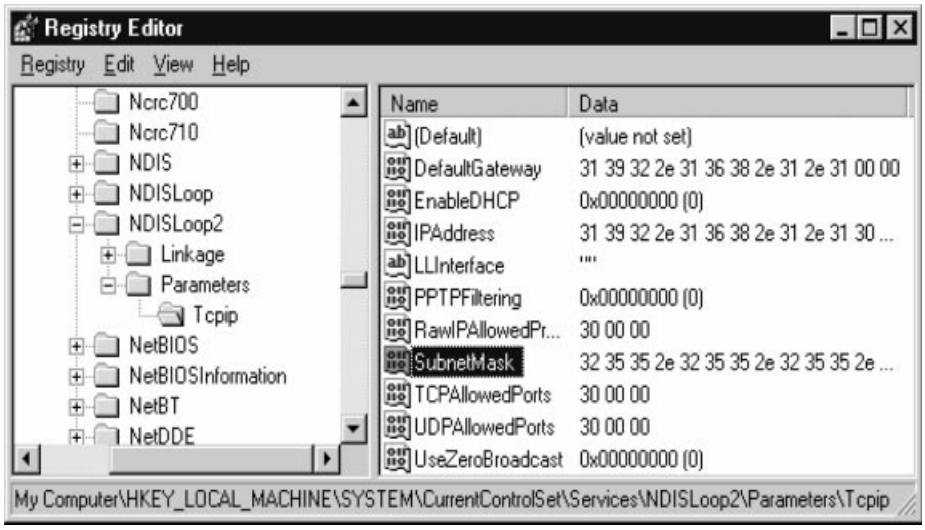

그림 5.3 Windows NT에서 레지스터리 편집하기

재부팅한 후에는, 루프백 어댑터의 IP 설정에 있는 서브넷 마스크를 확인하십시오.
이 값이 255.255.255.255로 설정되어 있어야 합니다. 그렇지 않을 경우, 레지스터리 편집기로 다시 돌아가십시오. 설정 대화창에서 나가려면 'Cancel'을 클릭하십시오. 'OK'를 클릭하면 서브넷 마스크가 잘못되었다는 메시지가 나타날 것입니다. 이것은 레지스터리 편집기를 사용하여 적절한 값을 설정하지 않았기 때문입니다.

이제 Windows NT 시스템을 클러스터 노드로 사용할 준비가 되었습니다. 물론, 시스템이 적절한 서비스를 실행하고 Advanced Traffic Manager의 구성에서 사용되도록 구성하는 일이 남았습니다. 루프백 인터페이스는 ARP 브로드캐스트에 응답하지 않으므로, ARP 응답을 해제할 필요는 없습니다. .

주)
지금까지의 내용은 직접 포워딩 방법에 해당됩니다. NAT 포워딩 방법을 사용할 때는, Windows NT 시스템에 대한 기본 게이트웨이만 설정하면 됩니다. 기본 게이트웨이를 NAT Gateway로 지정했던 주소로 설정하십시오. 물론, 시스템의 IP 주소는 네트워크의 개인용 영역에 속하게 설정해야 합니다.

Windows 2000 클러스터 노드 구성

Windows 2000 시스템을 클러스터 노드로 구성하려면, Microsoft Loopback Device를 설치해야 합니다. 먼저 'Start' 메뉴로 가서 'Settings' 메뉴에 있는 'Control Panel'을 선택한 다음, 'Add/Remove Hardware'를 선택하여 'Add/Remove Hardware' 제어판을 열어야 합니다. 'Add/Remove Hardware Wizard'가 나타나면, 'Next'를 클릭하여 다음 화면으로 가십시오. 'Add/Troubleshoot a device'를 선택하고 'Next'를 클릭하십시오. 시스템이 추가된 플러그 앤 플레이 하드웨어를 찾으려고 할 것입니다. 루프백 장치는 소프트웨어 장치이므로, 검사되지 않습니다. 'Add a new device'를 선택하고 'Next'를 클릭하십시오.

새로운 하드웨어를 찾을 것인지 물으면 'No, I want to select the hardware from a list'를 선택하고 'Next'를 클릭하십시오. 'Network adapters'를 선택하고 'Next'를 클릭하십시오. 왼쪽에서 'Microsoft'를 선택하고, 오른쪽에서 'Microsoft Loopback Adapter'를 선택하십시오.

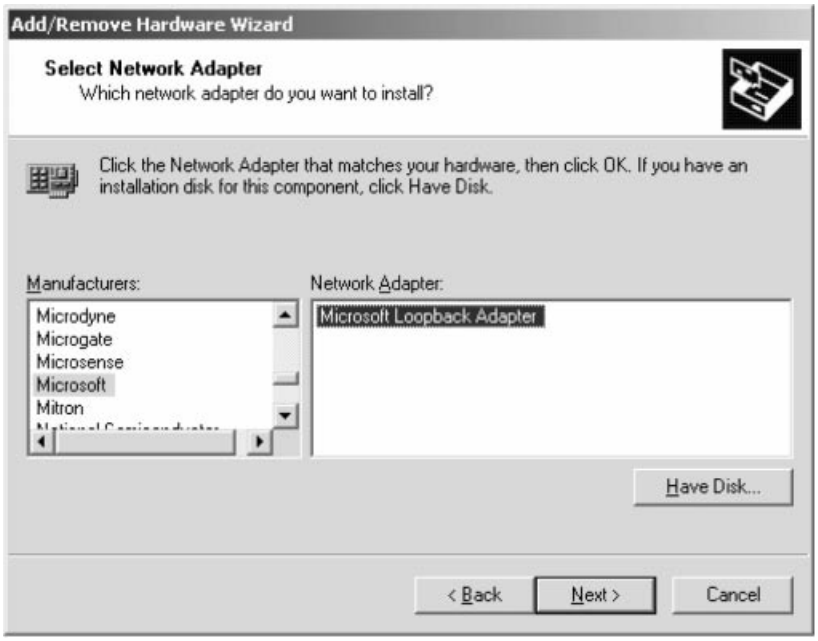

그림 5.4 Windows 2000에서 루프백 어댑터 추가하기

드라이버 설치가 완료될 때까지 'Next'를 클릭한 다음, 'Finish'를 클릭하여 마법사를 종료하십시오.

이제 'Network and Dial-up Connections' 제어판에는 추가의 Local Area Connection이 표시될 것입니다. 어떤 것이 루프백 어댑터인지 찾아내는 것이 약간 어려울 수도 있습니다. 일반적으로 가장 높은 번호가 매겨진 인터 페이스가 루프백 어댑터입니다. 더 확실하게 하려면, 'Start' 메뉴로 가서 'Settings' 메뉴에 있는 'Network and Dial-up Connections'을 선택하십시오. 각각의 Local Area Connection으로 마우스 포인터를 옮기면, 팝업 창에 드라이버가 표시됩니다. 'Microsoft Loopback Adapter'라고 표시된 것을 선택하십시오. 그러면 그 어댑터에 대한 상태 창이 열립니다. 상태 창에서 'Properties' 버튼을 클릭하십시오.

NetBEUI 프로토콜이 표시되어 있으면 삭제하십시오. 이 프로토콜은 요구되는 설정과 부합되지 않습니다.
Internet Protocol (TCP/IP) 라인을 선택하고 Properties를 클릭하십시오.
IP 주소를 클러스터의 가상 IP 주소로 설정하십시오. 또한 기본 게이트웨이도 설정하십시오.

서브넷 마스크는 255.255.255.255로 설정되어야 하지만, 대화 창에서는 이 값을 받아들이지 않을 것이므로, 그 대신 255.255.255.128로 설정하십시오.

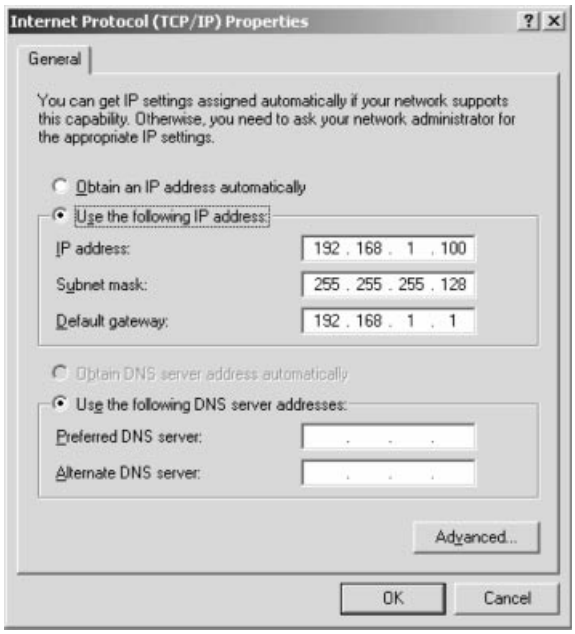

그림 5.5 Windows 2000에서 루프백 어댑터 구성하기

이 값은 나중에 레지스터리에서 255.255.255.255로 변경해야 합니다. 두 개의 Properties 대화 창에서 'OK'를 클릭하고 상태 창에서 'Close'를 클릭하십시오.

이제 레지스터리로 가서 서브넷 주소를 적절하게 설정해야 합니다. 'Start' 메뉴에서, 'Run'을 선택하고 REGEDIT를 입력한 다음 'OK'를 클릭하십시오. 왼쪽에서 HKEY_LOCAL_MACHINE\System\CurrentControlSet\.Services\Tcpip\Parameters\Interfaces로 이동하십시오.

목록에 다양한 인터페이스가 나와 있을 것입니다. 그 중에서 여러분이 설정한 IP 주소(즉, 클러스터의 가상 IP 주소)를 갖는 인터페이스를 찾아 보십시오. 해당되는 인터페이스 항목을 찾았으면 오른쪽에 있는 'SubnetMask' 항목을 두 번 클릭하십시오.

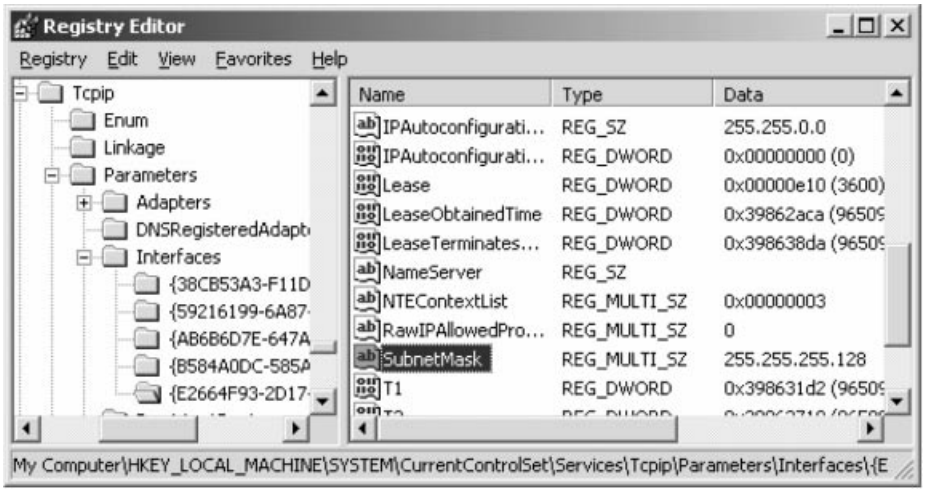

그림 5.6 Windows 2000에서 레지스터리 편집하기

128을 255로 변경하십시오.(오른쪽에서 1, 2, 8 자리 수를 삭제하고 2, 5, 5로 바꾸면 됨) 다른 것은 변경하지 않도록 주의하십시오. 'OK'를 클릭한 다음 레지스터리 편집기를 종료하십시오. 변경 사항이 적용되게 하려면 재부팅해야 합니다.

재부팅한 후에는, 루프백 어댑터의 IP 설정에 있는 서브넷 마스크를 확인하십시오. 서브넷 마스크 값이 255.255.255.255로 설정되어 있어야 합니다. 만약 그렇지 않다면, 레지스터리 편집기로 다시 돌아가야 합니다. 설정 대화창에서 나가려면 'Cancel'을 클릭하십시오. 'OK'를 클릭하면 서브넷 마스크가 잘못되었다는 메시지가 나타날 것입니다. 이것은 레지스터리 편집기를 사용하여 적절한 값으로 설정하지 않았기 때문입니다.

이제 Windows 2000 시스템을 클러스터 노드로 사용할 준비가 되었습니다. 물론, 시스템이 적절한 서비스를 실행하고 Advanced Traffic Manager의 구성에서 사용되도록 설정하는 일이 남았습니다.

루프백 인터페이스는 ARP 브로드캐스트에 응답하지 않으므로, ARP 응답을 해제할 필요는 없습니다. .

주)
지금까지의 내용은 직접 포워딩 방법에 해당됩니다. NAT 포워딩 방법을 사용할 때는, Windows 2000 시스템에 대한 기본 게이트웨이를 여러분이 NAT Gateway로 지정했던 주소로 설정하면 됩니다. 물론, 시스템의 IP 주소는 네트워크의 개인용 영역에 설정해야 합니다.

다른 시스템에서 클러스터 노드 구성

다른 종류의 시스템을 클러스터 노드로 구성할 경우, 클러스터링 될 서비스와 IP 별칭을 구성하고 ARP 응답을 해제하면 됩니다.

먼저, 서버가 클러스터링 될 네트워크 서비스를 제공하도록 구성해야 합니다. 이때 서비스에 다른 IP 주소에 대한 요청이 들어올 것임을 알려줄 수도 있습니다. 그렇지 않으면, 서버를 독립형 서버와 비슷하게 구성해야 합니다.

다음으로 클러스터의 가상 IP 주소로 IP 별칭을 추가하십시오. 대부분의 유닉스형 시스템에서는 `ifconfig` 명령을 통해 이 작업을 수행할 수 있습니다. 자세한 내용은 도큐멘테이션을 참고하십시오. Windows NT와 Windows 2000을 구성할 때와 마찬가지로, 루프백 어댑터 장치를 별칭으로 사용할 수도 있습니다.

마지막으로, 시스템이 ARP 요청에 응답하지 않게 하십시오. 이것은 아주 까다로운 작업이므로 도큐멘테이션을 참고하시기 바랍니다. `ifconfig` 명령을 사용할 경우, ARP 응답을 해제하는 옵션을 사용하면 됩니다. ARP 응답을 해제할 수 없으면 루프백 인터페이스에 대한 별칭을 만들어보십시오. ARP 응답이 해제되어 있는지 검사하려면, 다른 컴퓨터에서 별칭 주소를 ping 해보십시오. 이것이 실패하면, ARP 응답이 해제된 것입니다.

IP 주소 별칭으로 시스템을 구성할 수 없거나, ARP 응답을 해제할 수 없다면, 직접 포워딩 방법을 사용할 수 없는 것이므로, NAT 방법을 사용해야 합니다. 클러스터 노드의 IP 주소를 NAT 네트워크 상의 개인용 영역에 속하는 주소로 설정하십시오.

제 6 장

구성파일

이 장에서는 클러스터 서버 구성 파일의 각 섹션과 옵션을 자세히 살펴보도록 하겠습니다. 터보리눅스는 구성 파일을 직접 편집하는 대신 구성 툴 `tlcsconfig`이나 `turboclusteradmin`을 사용하여 구성 파일을 변경할 것을 권장합니다. 구성 툴은 사용하기 쉬운 인터페이스를 제공할 뿐 아니라, 구성 파일에 대한 일관성 검사를 수행 합니다. 또한, 구성 프로그램은 여러분이 수동으로 변경한 내용을 덮어쓸 수도 있으므로 주의하시기 바랍니다.

클러스터 서버 6의 구성 파일 형식은 이전의 4.0 버전과 거의 호환 가능합니다. 4.0 구성 파일을 클러스터 서버 6에서 실행되도록 변환하려면, AddServer 라인에 있는 포트 번호 설정과 콜론을 삭제하십시오.

clusterserver.conf 파일

clusterserver.conf는 메인 구성 파일입니다.
터보리눅스 클러스터 서버의 모든 구성 파일은 /etc/clusterserver 디렉토리에 저장됩니다.
또한, 이 디렉토리에는 몇 가지 견본 구성 파일, CMC가 사용하는 SSL 인증서, 그리고 사용권 파일이 저장된 하위 디렉토리가 포함 되어 있습니다.

사용권 파일은 /etc/clusterserver/.licenses 에 저장되며, 다양한 정보로 인코딩되어 있습니다.
사용권 파일은ATM, 노드, 서비스, 그리고 동시 커넥션의 개수와 사용 가능한 IP 주소 영역을 제한할 수 있습니다.
또한, 사용권은 특정 날짜에 만기가 되게 설정할 수 있습니다. 사용권 파일은 제품을 설치한 후에 추가합니다.

이러한 방식을 통해, 사용자는 평가 복사본을 시험해 본 후에 사용권을 구입하거나, 허용되는 노드와 ATM의개수를 증가시킬 수 있습니다. 디렉토리 안에 여러 개의 사용권 파일이 들어 있을 수도 있습니다. (사용권 파일은 누적
되며, 쓰이지 않는 사용권 파일은 무시됩니다)

구성 파일은 Apache 구성 파일과 비슷합니다. 구성 파일 안에 있는 전역 섹션은 일반적인 소프트웨어와 모든 클러스터에 적용됩니다. 전역 섹션 다음에는 특정 서버, 서비스, 클러스터를 설정하는 개별적인 섹션이 있습니다.

구성 파일은 사실상 ASCII 텍스트이며, 그 형식은 매우 다양합니다. 어떤 섹션에서는 특정 개체에 대한 모든 설정을
단 한 줄에 포함시켜야 하지만, 행간은 관계없습니다.

해시 문자(#)로 시작되는 라인은 명령어 라인로 무시됩니다.
키워드는 대소문자를 구분하지 않지만, 이 장에서 설명한 순서대로 섹션을 지정해야 합니다.

이 장의 나머지 부분에서 "섹션"이라는 용어는 다양한 기능을 규정하는 구성 파일 안의 섹션을 가리킵니다.

섹션의 예는 다음과 같습니다.

- NAT 섹션은 NAT 트래픽 포워딩 방법에서 사용될 매개 변수를 설정하는 방법을 규정합니다.
- Servers 섹션은 클러스터 안에 있는 각 서버 노드를 나타냅니다.

- ATMPool 섹션은 클러스터를 관리하는 데 사용되는 Advanced Traffic Managers와 클러스터 자체와 관련된 설정을 정의합니다.

전역 설정

어떤 설정은 구성 파일 안의 특정 섹션에 포함되지 않으며, 몇 가지 섹션은 모든 개체에 적용됩니다. 보안 설정, NAT 설정, 네트워크 마스크가 바로 그러한 예에 해당됩니다.

보안 설정

보안 설정은 특정 컴퓨터에 대한 관리 액세스를 제한합니다.

```
DenyHost badguy.hackers.usa/255.255.255.0
AllowHost partner.business.usa
```

주소는 IP 주소나 도메인 이름으로 지정할 수 있습니다. 전체 서브넷을 지정할 때는, 네트워크 마스크도 설정해야 합니다. 서브넷을 지정하지 않을 경우, 단 하나의 IP 주소만 허용되거나 거부됩니다. 여러 개의 AllowHost와 DenyHost 라인을 지정할 수도 있습니다. 이 라인들은 일치하는 부분이 발견될 때까지 서대로 처리됩니다. 이 설정은 `turboclusteradmin`의 'Global Settings' | 'Security Settings' 섹션에 구성됩니다.

클러스터를 안전하게 보호하려면 다음과 같은 설정을 사용해야 합니다.

```
AllowHost 127.0.0.1
DenyHost 0.0.0.0/0.0.0.0
```

이 경우, 로컬 호스트에 있는 CMC 데몬만이 `clusterserverd` 데몬의 관리 포트를 액세스할 수 있습니다.

네트워크 마스크 설정

NetworkMask 매개 변수는 클러스터 자체의 네트워크 마스크를 지정하는 데 사용됩니다. 주 ATM은 클러스터의 IP 주소를 결정할 때, 이 값을 자신의 서브넷 마스크로 사용합니다.

```
NetworkMask 255.255.255.0
```

기본 값인 255.255.255.0는 Class C 네트워크에 대한 네트워크 마스크입니다. 어떤 네트워크 마스크를 사용해야 할 지 모를 때는, 네트워크 관리자에게 문의하거나 서브넷에 있는 시스템의 네트워크 마스크 설정을 확인하십시오.

NAT 설정

NAT 섹션은 NAT 트래픽 포워딩 방법을 사용할 때 매개 변수를 어떻게 설정할 것인지 규정합니다.

```
NAT
   Subnet 10.0.0.0 255.255.0.0
   Gateway 192.168.0.100
EndNAT
```

Subnet 라인은 NAT 변환 과정에 사용될 주소 영역을 지정합니다. 이 라인은 네트워크 상의 어디에서도 사용되지 않는 주소 영역으로 지정해야 합니다. 대부분의 사이트는 10.0.0.0을 선택합니다. 동일한 라인에 지정된 마스크는 얼마나 많은 클라이언트가 변환될 수 있는지 지정합니다.

보통 사이트에는 255.255.0.0을 사용하고, 더 큰 사이트에는 255.240.0.0을 사용하십시오. Gateway 매개 변수는 ATM의 내부에서 구성될 가상 주소를 지정하며, NAT를 사용하는 노드에 의해 기본 게이트웨이로 사용됩니다. NAT 매개 변수 설정에 관한 자세한 내용은,.4장에서 NAT 설정 부분을 참고하시기 바랍니다.

서비스

클러스터는 다양한 서비스를 관리합니다.
UserCheck 섹션은 사용될 Application Stability Agents(ASA)를 정의하고, Services 섹션은 모든 서비스와 그에 대한 설정을 나타냅니다.

UserCheck 설정

서비스가 여전히 실행 중인지 확인하려면, 이 섹션에 지정된 Application Stability Agents를 사용하십시오. `turboclusteradmin` 프로그램의 'Clustered Services' | 'Application Stability Agents'에도 이와 동일한 항목이 구성되어 있습니다.

```
UserCheck ftp
   Up /usr/local/bin/ftp-up
   Down /usr/local/bin/ftp-down
   Check /usr/bin/ftpAgent
EndUserCheck
```

ASA 이름은 UserCheck 라인에서 지정됩니다. 일반적으로, 서비스 이름을 사용하면 됩니다. 여러분이 사용하는 이름은 Services 섹션의 Service 라인에서 사용될 것입니다.

ASA 검사를 위한 프로그램(또는 스크립트) 이름은 Check 라인에서 지정됩니다. 이 때 반드시 완전환 경로명을 지정해야 합니다. 이 프로그램은 서비스에 대한 테스트 트랜잭션을 수행하고, 그 트랜잭션이 실패하면 "1"를 반환합니다. ASA는 클러스터 노드의 IP 주소, 포트 번호, 그리고 소켓 타입(TCP인 경우 1, UDP인 경우 2)으로 호출됩니다.

Up과 Down 스크립트는 서비스가 다운되거나 다시 가동될 때 실행되며 ASA와 동일한 매개 변수로 호출됩니다. 대부분의 경우, Up과 Down 스크립트는 지정할 필요가 없습니다. Down 스크립트는 다운되었던 서비스를 다시 가동시킬 때 가장 유용합니다.

구성 파일에서 정의할 필요가 없이 미리 정의되어 있는 세 가지 ASA는 다음과 같습니다.

- 'http' ASA는 HTTP 1.1 프로토콜을 사용하여 웹 사이트를 검사합니다. HTTP 1.0 웹 사이트를 검사해야 할 경우, 외부의 http10Agent 프로그램을 사용하십시오.
- 'connect' 에이전트는 TCP 연결만 수행하고, 어떤 트랜잭션도 처리하지 않습니다.
- 'none' 에이전트는 어떤 검사도 수행하지 않기 때문에 항상 성공합니다.

ASA를 호출할 때는 다음 예에서처럼 여러 가지 매개 변수를 사용합니다.

```
/usr/bin/ftpAgent 192.168.0.7 23 1
```

이 예에서, ASA는 IP 주소가 192.168.0.7인 클러스터 노드에 있는 TCP 포트 23을 검사할 것입니다.

서비스 정의

Services 섹션은 클러스터에서 실행될 모든 네트워크 서비스를 정의하며, 각 서비스마다 한 라인씩 정의됩니다.

```
Services
   Service ftp tcp:21 ftp sticky
   Service nntp tcp:119 none
EndServices
```

키워드 Service 뒤에 나오는 첫 번째 매개 변수는 서비스의 이름입니다. 그 다음은 'tcp'나 'udp'가 앞에 붙은 포트 번호입니다. 그 다음은 UserCheck 섹션에서 지정된 Application Stability Agent의 이름이고, 마지막은 선택적 플래그입니다.

현재 사용할 수 있는 플래그는 'sticky'와 'failover'입니다. 'sticky' 플래그는 이 서비스가 영속적인 커넥션은 유지해야 함을 나타냅니다(각 커넥션마다 클라이언트를 동일한 서버에 연결함).

'failover' 플래그는 서비스가 부하 조정을 사용하지 않을 것임을 나타냅니다. 이 경우, 하나의 서버가 모든 트래픽을 받습니다. 그 서버가 다운되면, 모든 트래픽이 서버 풀 안의 다음 서버로 전송됩니다.

Servers와 ServerPool

구성 파일 안의 Servers와 ServerPool은 클러스터가 관리할 서버 노드를 정의합니다(IP 주소, 포워딩 방법, 포트 번호 포함).

서버

클러스터 안의 각 서버 노드는 Servers 섹션에 정의되어 있습니다. 각 라인은 Server 키워드와 서버의이름으로 시작됩니다.

그 다음은 서버의 IP 주소나 정식 도메인 이름이고, 그 다음은 사용할 포워딩 방법('direct', 'tunnel', 'nat') 입니다. 데몬이 주기적으로 서버를 검사하지 않게 하려면 'nonping' 플래그를 지정할 수도 있습니다.

다음은 Server 섹션의 예입니다.

```
Servers
   Server node1 node1.turbolinux.usa direct
   Server node2 192.168.0.102 tunnel noping
   Server node3 systemX.turbolinux.usa nat
EndServers
```

ServerPool 섹션

ServerPool 섹션은 구성 툴의 'Servers Configuration' | 'Server Groups Configuration' 섹션과 일치합니다.

이 섹션에서는 여러 개의 클러스터 노드를 관리하기 쉬운 하나의 단위로 조합할 수 있습니다. 서버 풀의 이름은 ServerPool 라인에서 지정합니다.

```
ServerPool servergroup1.
```

```
    AddServer node1       ftp/1
    AddServer node2       ftp/3 nntp/1
    CheckServerFrequency  120
    CheckServerTimeout    10
    CheckPortFrequency    240
    CheckPortTimeout      120
EndServerPool
```

AddServer 라인은 클러스터 노드, 그것이 지원할 서비스, 그리고 가중치를 풀에 추가합니다. 이 설정은 다음과 같은 형식을 갖습니다.

```
servicename/weight
```

여기서 *servicename*은 Service 섹션에 있는 항목의 이름입니다.
*weight*는 다른 서버에 비해 그 서버에 부여할 가중치입니다.
높은 가중치를 갖는 서버일수록 낮은 가중치를 갖는 서버에 비해 그 서버로 전달되는 패킷이 많습니다.

서버 노드 및 제공되는 서비스 목록 이외에도, 이 섹션은 안정성 검사를 위한 빈도수를 지정하는 데에도 사용됩니다.

CheckServer 설정은 각 서버를 얼마나 자주 검사할 것인지 지정합니다.
CheckPort 설정은 ASA 검사를 얼마나 자주 수행할 것인지 지정합니다.
Frequency 설정은 검사를 수행하는 간격을 지정합니다.
Timeout 값은 검사가 실패했다고 간주하기 전에 얼마동안 노드로부터 응답을 기다릴 것인지 나타냅니다.
모든 설정은 초 단위로 지정되며, Frequency 값은 반드시 Timeout 값보다 커야 합니다.

클러스터

클러스터를 정의하는 두 가지 섹션은 AtmPool과 VirtualHost입니다. 이 섹션은 `tlcsconfig` 프로그램에 있는 'Advanced Traffic Managers' 및 'Virtual Servers' 메뉴와 거의 유사합니다.

AtmPool 섹션

AtmPool 섹션은 클러스터를 관리하는 데 사용될 ATM과 클러스터 자체에 대한 설정을 정의합니다.
이 섹션은 AtmPool 키워드와 풀 이름으로 시작됩니다.

```
AtmPool                 atmgroup1
    AddAtm              atm1.turbolinux.usa
    AddAtm              192.168.1.10
    SendArpDelay        20
    MaxLostHeartbeats   3
    HeartbeatDelay      1
    NumConnections      10000
    NumServers          1000
    NumServices         100
    ConnectionTimeout   600
EndAtmPool
```

주)
현재는 단 하나의 ATM 풀만 지원됩니다.

ATM은 AddAtm 라인에서 IP 주소나 정식 도메인 이름을 지정하면 간단하게 추가할 수 있습니다.

ATM 자체에 대한 설정 외에도 다음과 같은 여러 가지 설정이 있습니다.

SendArpDelay는 클러스터의 IP 주소를 알리는 ARP 브로드캐스트를 얼마나 자주 전송할 것인지 지정합니다. (초 단위로 지정)

HeartbeatDelay는 주 ATM이 백업 ATM에게 자신이 작동 중임을 알리는 heartbeat 브로드캐스트를 얼마나 자주 전송할 것인지 지정합니다(초 단위로 지정).

MaxLostHeartbeats는 백업 ATM이 heartbeat를 몇 번이나 놓친 후에 주 ATM이 다운되었다고 간주하고 백업의 주 ATM 승격 작업의 시작 여부를 알려줍니다.

ConnectionTimeout은 비활성 커넥션을 클러스터 데몬의 테이블 안에 얼마나 오랫동안 유지할 것인지 지정합니다.

클러스터가 지원할 서버, 서비스, 커넥션의 최대 개수를 특정 값으로 지정하여 데몬을 조정할 수 있습니다. NumServers, NumServices, NumConnections 키워드를 사용하여 지정합니다. 기본 값은 각각 1,000개의 서버, 100개의 서비스, 그리고 10,000개의 커넥션입니다. 여러분이 사용할 클러스터의 요구에 맞게 이 값을 구성하시기 바랍니다. 이에 관한 자세한 내용은 7장을 참고하십시오.

주)
클러스터에 대한 커넥션 수가 구성 파일에 지정된 최대 값을 초과할 경우, 새로운 커넥션은 모두 실패할 것입니다. 커널 모듈은 여러분이 구성 파일에서 지정한 개수만큼의 항목을 포함하는 테이블을 만듭니다. 이 값은 예상되는 동시 커넥션의 최대 개수보다 더 큰 값으로 설정하십시오.

VirtualHost 섹션

VirtualHost 섹션은 ATM 풀, 서버, 풀, 그리고 IP주소를 정의합니다. 이 섹션에서 관리자의 전자우편 주소를 지정할 수도 있습니다. 이 섹션의 형식은 다음과 같습니다.

```
VirtualHost 192.168.1.2
   AddAtmPool     router1
   AddServerPool  servergroup1
   MailTo         cluster-admin@mybusiness.org
EndVirtualHost
```

IP 주소는 섹션의 헤더에 포함됩니다. 이 값은 클러스터 자체에 연결할 때 사용할 가상 IP 주소입니다. 이 IP 주소로 전송되는 패킷은 모두 클러스터 노드로 포워딩됩니다.

주소는 도트로 구분된 숫자 값이나 도메인 이름으로 지정할 수 있습니다. 도메인 이름을 사용할 경우, 그것이 적절한 IP 주소로 변환될 수 있어야 합니다. 또한, 그 IP 주소를 갖는 다른 컴퓨터가 없어야 합니다.

AddAtmPool은 이 클러스터에서 사용될 ATM을 지정하는데, 이 매개 변수는 단순히 앞 섹션에서 지정했던 ATM 풀을 가리킵니다. 마찬가지로, AddServerPool은 클러스터 노드에 대해서도 동일하게 지정합니다.

MailTo 매개 변수는 클러스터에 문제가 생겼을 때 전자우편 메시지를 전송하는데 사용됩니다. 전자우편 주소를 지정하지 않으면 어떤 전자우편 메시지도 전송되지 않을 것입니다.

어쨌든 메시지는 /var/log/clusterserverd.log 파일에 기록됩니다. 메시지는 ATM, 서버 노드, 또는 서비스가 다운될 때마다 전송되며, SpeedLink 모듈이 중단되는 것 같은 치명적인 오류가 발생할 때도 전송됩니다.

주)
여러 개의 VirtualHost 섹션을 사용하면 여러 개의 가상 IP 주소를 정의할 수 있지만, 이들 IP 주소는 모두 동일한 ATM 풀을 공유해야 합니다.

제 7 장

관 리

이 장에서는 클러스터의 작동과 성능을 제어하는 데 사용할 수 있는 툴을 살펴보도록 하겠습니다.
이 툴을 사용하면 클러스터의 작동을 변경하고 문제를 해결할 수도 있습니다.

이 장에서 배울 주요 내용은 다음과 같습니다.

- 관리 툴
- 동기화 프로그램
- 클러스터 관리 콘솔(CMC)
- 문제 해결

관리 툴

클러스터를 관리하는 데 사용할 수 있는 툴은 여러 가지가 있습니다. 그 중에서 `turboclusteradmin`과 `tlcsconfig` 프로그램은 4장에서 이미 설명한 바 있습니다. 이 툴은 클러스터를 구성하고 변경하는 데 사용할 수 있는 주요 툴입니다.

클러스터의 구성을 변경하는 작업은 주로 새로운 서버나 서비스를 추가하는 것입니다. 이 작업은 초기 구성과 거의 똑같습니다. 차이점이 있다면, 클러스터를 실행할 수 있게 만든 후에 서버와 서비스를 추가하는 것이 더 쉽다는 것입니다.

바로 이러한 이유 때문에 간단한 구성부터 시작해서 서버와 서비스를 추가하는 것이 바람직합니다. 간단한 시스템은 복잡한 시스템보다 작동시키기가 더 쉽습니다.

클러스터 설정을 변경하는 또 한가지 이유는 성능을 최적화하기 위한 것입니다.

클러스터 조정

일단 클러스터가 성공적으로 실행되면, 클러스터를 조정할 수 있습니다. 클러스터의 성능을 최적화시키기 위해 변경할 수 있는 매개 변수는 여러 가지가 있지만, 여러분이 주로 조정하는 설정은 커널 테이블 크기와 시간 설정입니다.

실제로 클러스터의 성능을 최적화시킬 때는, 네트워크 분석기 같은 고급 네트워크 제어 툴을 사용할 수 있습니다. 여러분은 ATM이 얼마나 많은 오버헤드 네트워크 트래픽을 생성할 것인지 결정할 수 있습니다.

이러한 오버헤드에는 heartbeat 브로드캐스트, 서버 핑, ASA 서비스 검사 등이 있습니다. 또한, 시스템의 검사 간격을 조정할 수도 있습니다. 단, 검사 간격을 증가시키면, 서버 중 하나가 다운될 경우 서비스를 사용할 수 없게 되는 시간도 늘어난다는 점에 주의하십시오.

커널 테이블 크기

첫 번째로 변경할 수 있는 설정은 커널 모듈이 사용하는 테이블의 크기입니다. SpeedLink 모듈은 서버, 서비스, 그리고 커넥션에 대한 테이블을 갖고 있습니다. 이 테이블은 클러스터가 받을 수 있는 최대 용량을 수용할 수 있을 만큼 크게 설정해야 하지만, 지나치게 커서도 안됩니다.

가장 적합한 서버와 서비스의 개수는 아주 간단하게 알아낼 수 있습니다. 서비스의 개수는 설명이 필요 없습니다. 각각의 명명 서비스가 테이블 안에서 한 항목씩 차지합니다.

이들 서비스는 `tlcsconfig` 안의 'Service Settings' 메뉴에서 정의되며, 구성 파일 안에서 각각의 서비스가 개별 적인 'Service' 라인을 갖습니다.

서버의 개수는 실제로 서버/서비스 쌍의 개수입니다. 따라서, 클러스터가 FTP와 HTTP를 처리하는 10개의 노드를 갖고 있다면, 서버 테이블의 크기를 20으로 설정해야 합니다. 테이블의 크기를 현재 여러분이 필요로 하는 것보다 약간 더 크게 설정하면 나중에 노드를 추가하기가 더 쉽습니다. 노드를 추가할 때 테이블 설정을 변경해도 되지만, 잊어버릴 수도 있기 때문입니다.

커넥션의 최대 개수는 결정하기가 약간 어렵습니다. 커넥션 테이블이 다 차면 추가로 들어오는 커넥션은 서비스를 받지 못하고 무시되기 때문에, 이것을 지나치게 작은 값으로 설정해서는 안 됩니다. 커넥션의 테이블의 적절한 크기를 결정하는 가장 좋은 방법은 클러스터의 사용을 감시하는 것입니다.

이상적으로는, 한번에 가질 수 있는 커넥션의 최대 개수보다 약간 더 크게 설정하는 것이 좋으며, 주먹 구구식이긴 하지만 커넥션의 최대 개수를 관찰하여 그것을 2배로 곱하는 방법도 있습니다. 클러스터가 갑자기 과다한 트래픽을 받지 않는다면, 이 방법은 거의 모든 상황에 적용할 수 있습니다.

ATM이 NAT 클러스터 노드를 서비스할 경우, 커넥션의 개수를 두 배로 곱해야 합니다. 이 때 ATM은 클라이언트로부터 NAT-변환 주소로의 가상 연결과 NAT-변환 주소로부터 클러스터 노드로의 실제 연결을 만들기 때문입니다.

이들 설정은 모두 클러스터 관리 툴에 있는 'Advanced Traffic Manager Settings' 메뉴에서 구성할 수 있으며, 구성 파일 안의 AtmPool 섹션에 'NumServices', 'NumServers', 그리고 'NumConnections'라는 이름으로 정의되어 있습니다.

시간 설정

클러스터를 조정하는 데 사용할 수 있는 시간 설정은 여러 가지가 있습니다. 클러스터 전체와 관련된 시간 설정은 테이블 크기를 정의하는 것과 같은 방법으로 정의합니다. 커넥션 타임아웃 값과 heartbeat 빈도가 이러한 설정에 해당됩니다. 다른 시간 설정은 시스템과 서비스 검사의 빈도를 정의하는 데 사용할 수 있습니다.

커넥션 타임아웃 값은 사용되지 않는 커넥션에 대한 항목을 커넥션 테이블 안에 얼마나 오랫동안 유지할 것인지 지정합니다. 이 설정은 클러스터에서 실행되는 모든 서비스에 적용됩니다. 몇 가지 권장할 만한 설정은 다음과 같습니다.

HTTP	30-60 seconds
FTP	15- 30 s econds
Telnet	300 seconds

이러한 서비스를 하나 이상 실행할 경우, 클러스터에 사용할 수 있는 가장 긴 시간을 선택하십시오.

여러분이 설정할 수 있는 또 하나의 매개 변수 세트는 서비스와 서버 검사의 빈도입니다. 이 값을 증가시키면 네트워크 트래픽 오버헤드가 줄어들지만, 서버가 클러스터에서 제거되기 전에 다운될 수 있는 시간은 늘어납니다.

따라서, 여러분은 이러한 장단점을 어떻게 이용할 것인지 결정해야 합니다. 100 Mbps 네트워크의 경우, 클러스터 노드가 아주 많지 않다면, 네트워크 오버헤드는 실제로 그렇게 크지 않으므로, 높은 검사 빈도를 선택할 수 있습니다.

시스템의 검사 빈도를 변경하려면, `tlcsconfig` 프로그램에 있는 'Server GroupsConfiguration' 메뉴로 가서,서버 검사와 서비스 검사에 대한 'Frequency' 설정을 변경하십시오.

이 메뉴에서 서버나 서비스가 다운되었다고 간주하기 전에 얼마동안 응답을 기다릴 것인지 나타내는 'Timeout' 값도 설정할 수 있습니다. 이 설정은 구성 파일의 'ServerPool' 섹션에 있는 'CheckServerFrequency' 및 'CheckPortFrequency'에 해당됩니다.

'Frequency' 설정은 해당되는 'Timeout' 값보다 반드시 더 커야 합니다. 그러지 않으면, 첫 번째 ping에 대한 응답을 받기 전에 또 다른 ping을 보내는 상황이 발생하며, 응답을 받았을 때 그것이 첫번째 ping에 대한 것인지 두 번째 ping에 대한 것인지 분간할 수 없게 됩니다.

동기화 툴

클러스터 서버 6은 두 가지의 동기화 툴을 제공합니다.
하나는 구성을 동기화시키는 프로그램인 tlcs_config_sync 이고, 다른 하나는 내용을 동기화시키는 tlcs_content_sync입니다. 이 툴을 사용하면 클러스터 안의 서버 사이에 일관성을 유지할 수 있습니다.

동기화 툴을 사용하기 위해서는 모든 서버에 업데이트된 내용이나 구성을 받는 SSH 데몬이 실행되고 있어야 합니 다. 클러스터 서버를 설치한 시스템에는 SSH가 포함되어 있으므로 이 툴을 사용할 수 있습니다. 하지만, Windows NT나 리눅스 이외의 다른 시스템에서 이 툴을 사용하려면, SSH를 직접 설치해야 합니다.

tlcs_config_sync의 경우에는 이것이 문제시되지 않습니다. 구성 파일은 클러스터 서버 데몬을 실행하는 시스템에서만 필요하기 때문입니다. 하지만, 내용을 동기화시킬 경우에는 SSH가 없으면 직접 시스템을 동기화시켜야 합니다. 경고 메시지가 나타나지 않게 하려면, 동기화시킬 서버 목록에서 SSH가 없는 시스템을 삭제하십시오.

tlcs_content_sync

tlcs_content_sync 유틸리티를 사용하면 모든 클러스터 노드가 동일한 내용을 유지하게 할 수 있습니다. 이 프로그램은 명령 프롬프트에서 시작할 수도 있고, turboclusteradmin 프로그램에서 시작할 수도 있습니다. 내용을 동기화하는 방법은 다음과 같습니다.

1. 명령어 라인에서 유틸리티를 시작하십시오.

```
# tlcs_content_sync
```

turboclusteradmin 프로그램에서 'Content Synchronization Tool' 메뉴 항목을 선택하여 시작할 수도 있습니다. 어떤 방법을 사용하든 마찬가지입니다.

2. 디렉토리 목록과 서버 목록이 있는 화면이 나타납니다. TAB 키를 사용하면 두 목록 사이를 이동할 수 있습니다.
Up/down 커서 키를 사용하여 목록을 스크롤하면 화면에 나타나지 않는 항목을 볼 수 있습니다.

그림 7.1 tlcs_content_sync 메인 메뉴

3. 디렉토리 목록은 동기화시킬 내용을 자세히 보여줍니다. 목록에 있는 각 디렉토리 안의 모든 내용(하위 디렉토리까지 포함)이 복사됩니다. 일반적으로, /home/httpd/html과 /home/ftp같은 디렉토리를 복사하지만, 소스시스템에 존재하는 어떤 디렉토리든지 복사할 수 있습니다.

4.목록에서 디렉토리를 추가하거나 삭제하려면, 화면 하단에 있는 'Edit Dir' 버튼을 클릭하십시오. 그러면, 디렉토리 항목을 추가, 삭제, 편집할 수 있는 새로운 화면이 나타납니다. 동기화시킬 디렉토리를 모두 선택했으면 'Done'을 클릭하십시오.

5.어떤 서버가 동기화된 내용을 받을 것인지 변경하려면, 'Edit Servers' 버튼을 클릭하십시오. 그러면, 시스템 목록과 함께 서버 이름을 추가, 삭제, 편집할 수 있는 버튼이 나타납니다. 동기화시킬 서버 목록을 구성한 후에 'Done'을 클릭하십시오.

6. 처음에는 구성 프로그램에서 클러스터 노드로 추가했던 모든 서버가 표시될 것입니다. 이들 노드 중에서 SSH 데몬을 실행하지 않는 것이 있으면 목록에서 삭제하거나, 그 서버에 SSH를 설치해야 합니다.
   SSH 데몬을 설치하지 않은 서버를 목록에 남겨두면, 동기화 절차를 시작할 때 경고 메시지가 나타납니다.
   경고 메시지는 그 서버에 의해 커넥션이 거부되었으며, 오류 때문에 동기화가 완료되지 못했음을 나타냅니다.

7. 적절한 디렉토리와 서버를 선택했으면, 'Start' 버튼을 클릭하십시오.

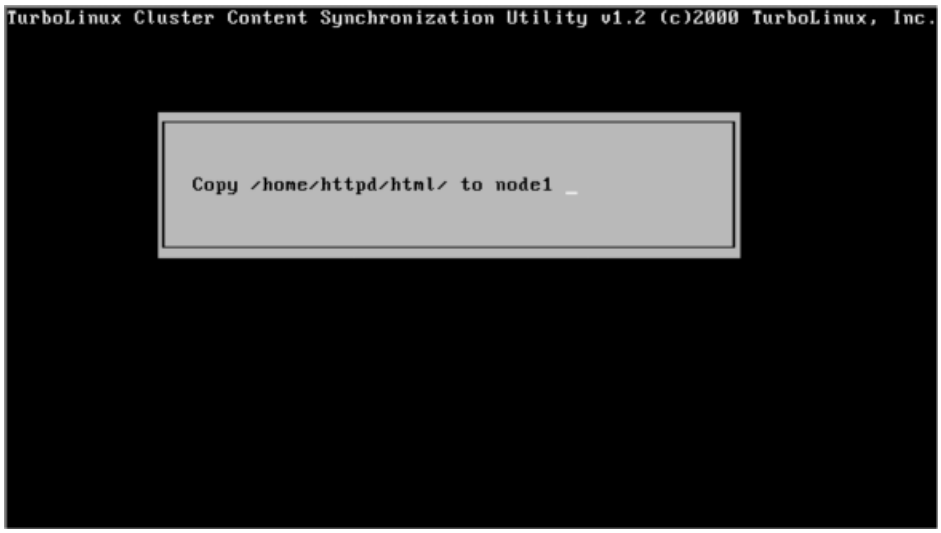

그림 7.2 내용 동기화 진행

동기화 프로그램은 목록에 있는 각 서버로 모든 디렉토리의 내용을 전송합니다.

8. 이전에 특정 서버로 동기화 시킨 적이 없었다면, 해당 시스템에 대한 root 암호를 입력하라는 메시지가 나타날 것입니다.
   기본적인 SSH 툴은 맨 처음에만 이 단계를 요구하고, 나중에 다시 사용될 때를 대비해 암호를 기억합니다.

9.오류가 발생하면, 무엇이 잘못되었는지 알려주는 메시지가 표시됩니다. 거의 대부분의 오류는 시스템이 다운 되거나 클러스터 서버와 함께 제공되는 SSH 데몬을 실행하지 않기 때문에 발생합니다. 앞으로 경고 메시지가 나타 나지 않게 하려면, 이 서버를 삭제하십시오.

10.동기화 절차가 완료되면, 그 결과가 성공적이었는지, 아니면 몇몇 서버로 연결하는 데 문제가 있었는지를 알려주는 메시지가 제공됩니다. 유틸리티를 종료하려면 'Close' 버튼을 클릭하십시오.
오류가 있었다면, 오류를 해결하고 다시 시도해보시기 바랍니다.

tlcs_config_sync

tlcs_config_sync 유틸리티는 클러스터 서버의 구성 파일을 동기화하는 데 사용됩니다.
tlcs_config_sync 프로그램과 마찬가지로, 명령어 라인에서 실행할 수도 있고, turboclusteradmin. 에서 실행 할수도 있습니다. 구성을 동기화하는 방법은 다음과 같습니다.

1. 명령어 라인에서 유틸리티를 시작하십시오.

```
# tlcs_config_sync
```

turboclusteradmin 프로그램에서 'Configuration Synchronization Tool' 메뉴 항목을 선택하여 시작해도 됩니다. 어떤 방법으로 시작하든 마찬가지입니다.

2. 서버 목록이 나타날 것입니다. 처음에는 구성 프로그램에서 노드나 ATM으로 추가했던 모든 서버가 표시됩니다.

그림 7.3 tlcs_config_sync 메인 메뉴

3. 목록에서 터보리눅스 클러스터 서버를 설치하지 않은 서버는 삭제해야 합니다.
해당 서버를 반전 표시한 다음, 'Remove' 버튼을 누르면 됩니다.
그러면 그 시스템이 목록에서 바로 삭제될 것입니다. 클러스터 서버를 설치하지 않은 서버를 목록에 남겨두면, 동기화 절차를 시작할 때 경고 메시지가 나타납니다.
경고 메시지는 그 서버에 의해 커넥션이 거부되었으며, 오류 때문에 동기화가 완료되지 못했음을 알려줍니다.

4. 목록에서 빠진 있는 서버를 추가하십시오. 'Add' 버튼을 클릭하고 제공되는 입력란에 서버의 이름을 입력하면 됩니다. 'OK'를 클릭하면 서버가 목록에 추가됩니다.

5. 적절한 서버 목록을 구성했으면, 'Start' 버튼을 클릭하십시오.

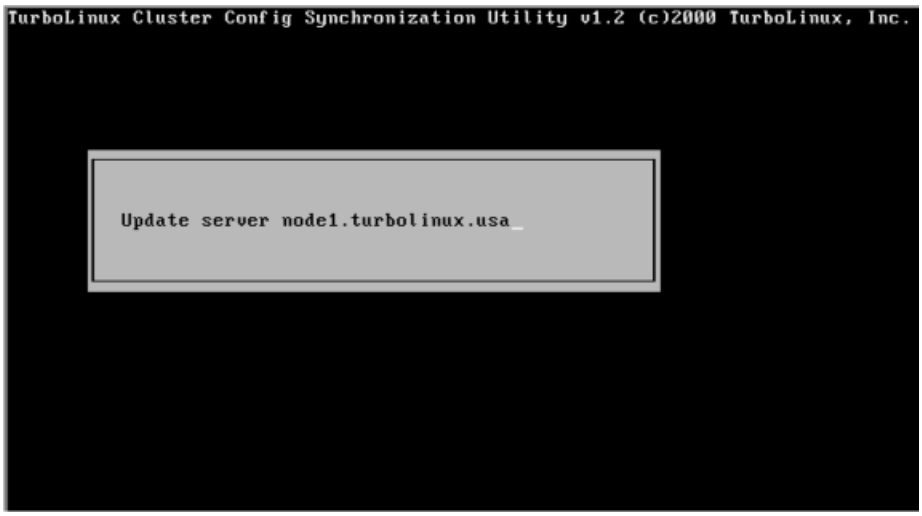

그림 7.4 구성 동기화 진행

동기화 프로그램은 서버 목록에 있는 각 서버로 업데이트된 구성 파일을 전송합니다.

6. 이전에 특정 서버로 동기화시킨 적이 없었다면, 해당 시스템에 대한 root 암호를 입력하라는 메시지가 나타날 것입니다. 기본적인 SSH 툴은 맨 처음에만 이 단계를 요구하고, 나중에 다시 사용될 때를 대비해 암호를 기억합니다.
7. 오류가 발생하면, 무엇이 잘못되었는지 알려주는 메시지가 제공됩니다. 거의 대부분의 오류는 시스템이 다운되거나 클러스터 서버와 함께 제공되는 SSH 데몬을 실행하지 않기 때문에 발생합니다. 앞으로 경고 메시지가 나타나지 않게 하려면, 이 서버를 삭제하십시오.
8. 동기화 절차가 완료되면, 그 결과가 성공적이었는지, 아니면 일부 서버로 연결하는 데 문제가 있었는지를 알려주는 메시지가 제공됩니다. 'Close' 버튼을 클릭하여 유틸리티를 종료하십시오. 오류가 있었다면, 오류를 해결하고 재시도해 보시기 바랍니다.

클러스터 관리 콘솔 (CMC : Cluster Management Console)

클러스터 관리 콘솔은 클러스터의 성능을 제어하는 웹 기반의 툴입니다. 이 툴을 사용하면 클러스터의 작동을 제어 하는 매개 변수를 보거나 변경할 수 있으며, 심지어는 클러스터가 실행되고 있는 도중에 동적으로 클러스터를 변경 할 수도 있습니다. 또한, CMC를 통해 클러스터 서버 도큐멘테이션을 보거나 버그 보고서를 제출할 수도 있습니다.

CMC를 실행하려면, 웹 브라우저를 시작하고 해당되는 URL로 연결하십시오.
CMC는 안전한 HTTP 프로토콜(HTTPS)을 사용하고 포트 910에서 실행됩니다. CMC로 연결하려면 SSL을 지원하는 웹 브라우저가 필요합니다. 현재 나와 있는 Netscape와 Internet Explorer 버전도 사용할 수 있습니다.

클러스터의 멤버가 아닌 시스템을 사용하여 CMC로 연결할 때는, 클러스터 자체의 주소나 개별적인 ATM의 주소를 사용해야 합니다. 클러스터의 가상 IP 주소를 사용할 경우, 현재 주 ATM인 시스템으로 연결됩니다. 클러스터 안의 시스템에서 브라우징할 경우, 개별적인 ATM의 주소를 사용해야 합니다. CMC로 연결할 때 사용
되는 URL은 다음과 같습니다.

```
https://atm1.turbolinux.usa:910
```

주)
CMC로 연결하는 데 문제가 있다면, https의 S를 빠뜨렸거나 포트 910을 지정하지 않았기 때문일 것입니다.

처음으로 CMC 페이지에 연결할 때는, 브라우저가 SSL 사이트 인증서를 보여주는 대화창을 팝업할 것입니다. 이 대화창에는 그 인증서가 인증 기관에 의해 승인되지 않았음을 나타내는 경고 메시지가 포함되어 있을 수도 있습니다. 여러분이 전송하는 정보는 시스템 자체에서만 볼 수 있으므로 이 경고 메시지는 무시해도 됩니다.

또한, 인증서에 적절한 사이트 이름이 포함되어 있지 않다는 경고 메시지가 나타날 수도 있습니다. 이것은 ATM이 여러 개의 도메인 이름을 갖고 있거나, 여러분이 소프트웨어를 설치할 때 부적절한 정보를 입력했기 때문입니다. 인증서를 받아들이고 브라우저가 제공하는 메시지대로 따르십시오.

여러분은 CMC로 연결할 때마다, 로그인 이름과 암호를 입력하라는 메시지를 받게 될 것입니다.

이때 사용자 ID 'tlcsadmin'을 사용해야 합니다. 이 계정은 클러스터 서버가 설치될 때 만들어진 것이며, 처음에는 여러분이 연결하고 있는 ATM에 대한 root 계정과 동일한 암호를 갖습니다.

암호를 입력하면 CMC로 연결됩니다(ATM에 존재하는 다른 사용자 ID로 로그 인할 수도 있지만, 이 경우 어떤 내용도 변경할 수 없습니다.)

처음 로그인하면, CMC 홈 페이지가 제공됩니다. 이 홈페이지의 상단에는 다른 다양한 페이지를 탐색할 수 있는 아이콘이 있습니다. 여러분이 로그인할 때 사용했던 ID에 따라 제공되는 아이콘이 다릅니다.

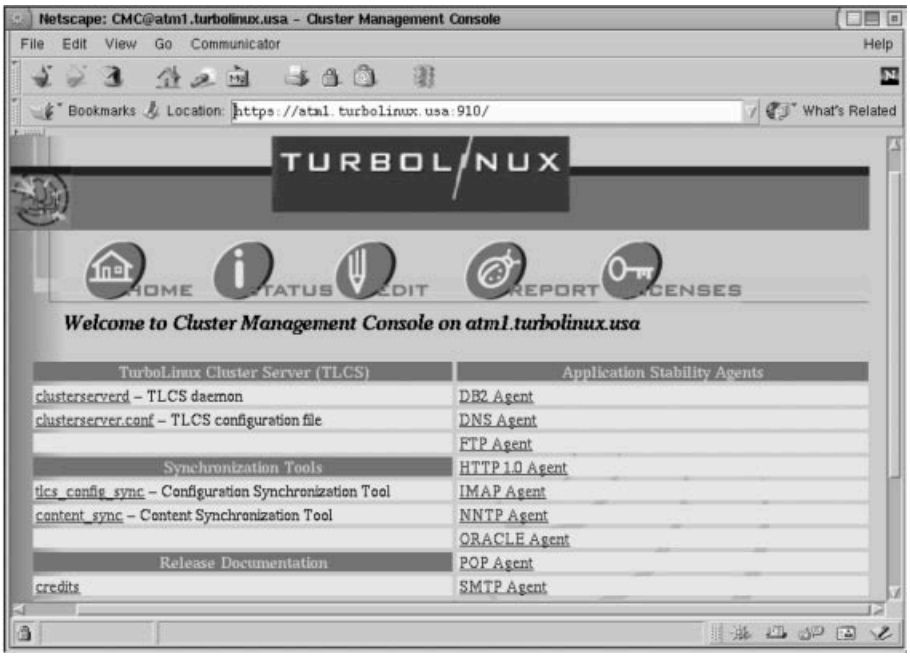

그림 7.5 CMC 홈 페이지.

이 페이지에는 클러스터 서버 도큐멘테이션과 CMC 관리 프로그램의 다른 페이지에 대한 링크가 포함되어 있습니다.

Home	메인 페이지와 다른 도큐멘테이션에 대한 링크가 있는 홈 페이지Status 클러스터0를 제어할 수 있는 Status 페이지. 이 페이지에는 클러스터 프로세스, 커널 모듈, 네트워크 인터페이스,/proc/net/cluster 파일, 로그 파일에 관한 정보가 있습니다. 또한, 이 페이지에서 몇 가지 설정을 변경할 수도 있고, 데몬을 중단하고 재시작할 수도 있습니다.
Edit	이 페이지에서는 /etc/clusterserver/clusterserver.conf 구성 파일을 보거나 변경할 수있습니다.
Report	Report 페이지에서는 클러스터의 구성과 관련된 정보가 들어 있는 전자우편 메시지를 만 수있습니다. 또한, 누구에게 정보를 전송할 것이며 어떤 정보를 포함시킬 것인지도 선택할 수 있습니다. 이 페이지는 터보리눅스 기술 지원 팀으로 버그를 보고할 때 매우 유용합니다.
Licenses	이 페이지는 시스템에 대한 사용권 파일을 볼 때 사용됩니다.
View	View 페이지는 Edit 페이지와 비슷합니다. 단, 구성 파일을 보여 주기만 하고, 그것을 편집하는 기능은 제공하지 않습니다.

'tlcsadmin'로 로그 인하면, View 페이지를 제외한 모든 페이지를 볼 수 있습니다. 다른 사용자로 로그 인할 경우, View 페이지는 액세스할 수 있지만, Edit, Report, License 페이지는 액세스할 수 없습니다.

Status 페이지는 CMC에서 가장 많이 사용되는 페이지입니다. 이 페이지에는 세 가지 중요한 섹션이 포함되어 있습니다. 맨 위에 있는 섹션은 여러 가지 유틸리티(ps, lsmod, ifconfig)의 출력을 보여주고, ps 출력은 클러스터 서버 데몬의 실행 여부를 알려줍니다.

lsmod 명령은 로드되어 있는 커널 모듈을 보여줍니다. 이 목록에는 ip_cs 모듈이 반드시 포함되어 있어야 합니다.

ifconfig 프로그램은 시스템에 있는 네트워크 인터페이스의 구성을 보여줍니다. 이 프로그램의 출력을 보면 적절한 별칭이 구성되었는지 확인할 수 있습니다. 이 섹션에는 가동 중인 주 ATM에 있는 clusterserverd 데몬을 시작하거나 종료하는 버튼도 있습니다.

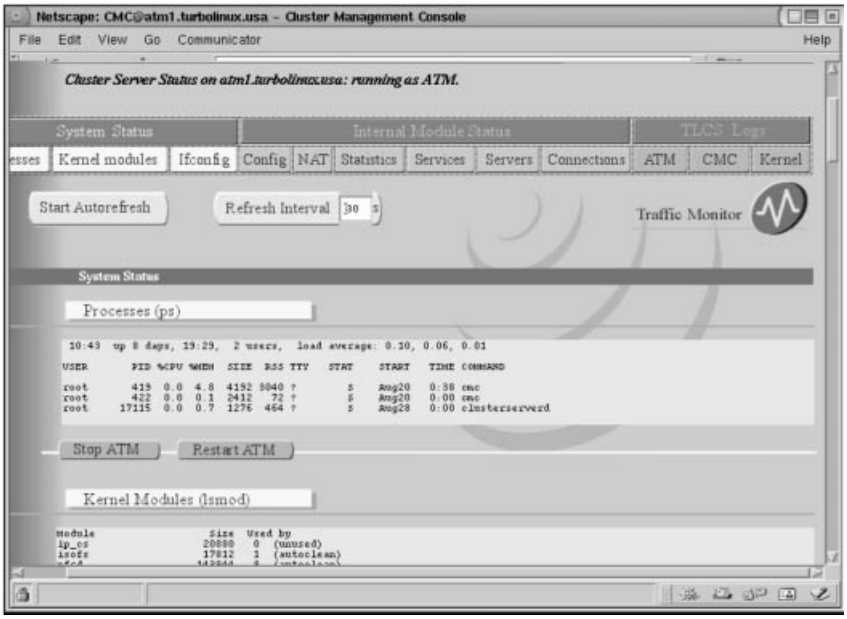

그림 7.6 CMC Status 페이지

다음 섹션은 /proc/net/cluster 파일의 출력을 보여줍니다. 이 출력은 실행중인 클러스터에 관한 정보를 제공합니다. 특히 중요한 정보는 Statistics와 Connections입니다. 이 정보는 클러스터를 액세스함에 따라 변경됩니다. 화면의 상단에 있는 'Autorefresh' 버튼을 사용하면 정보가 주기적으로 업데이트되게 할 수 있습니다. /proc/net/cluster 파일은 나중에 자세히 설명하도록 하겠습니다.

마지막 섹션은 클러스터 서버가 사용하는 세 가지 로그 파일의 출력을 보여줍니다. 이 로그를 보면 어떤 문제가 발생했는지 알아낼 수 있습니다. 하지만, 유닉스 필터 프로그램은 액세스할 수 없으므로, CMC 브라우저 인터페이스를 사용하는 것보다는 명령어 라인에서 로그 파일을 보는 것이 더 쉽습니다.

Edit 화면에서는 구성 파일을 변경할 수 있습니다. 구성 파일을 변경한 후에는 'Commit' 버튼을 클릭하십시오. 그러면, 새로운 구성이 모든 클러스터 컴퓨터 (그 컴퓨터가 SSH를 실행하고 있을 때에 한함)로 전송됩니다. 그전에 클러스터 안의 특정 서버로 동기화시킨 적이 없었다면, 화면 하단에 있는 폼에서 각 서버에 대한 암호를 입력해야 합니다.

시스템이 동기화 절차를 실행하기 전에 암호 입력을 요구하는 대신 암호가 누락되었음을 알리는 메시지를 제공하게 하려면 CMC 동기화 절차를 사용하기 전에 명령 줄이나 turboclusteradmin 프로그램을 통해 tlcs_config_sync 툴을 사용하시기 바랍니다.

주의)
CMC를 사용할 때는 브라우저의 'Refresh'나 'Reload' 버튼을 누르지 마십시오.
이 버튼을 누르면 button-press 이벤트가 재전송되어, CMC가 그 버튼과 관련된 동작을 여러 번 실행하게 됩니다.

Status 페이지에서 사용할 수 있는 또 하나의 화면은 Traffic Monitor라는 화면입니다. Traffic Monitor는 자바 애플릿이므로, 이 화면을 사용하려면 브라우저에 자바 지원이 설정되어 있어야 합니다. 이 애플릿은 ATM을 통과하는 트래픽을 그래픽으로 나타내며, 클러스터, 서버, 또는 서비스 별로 통계치를 제공합니다.

다음은 여러 가지 클러스터 노드를 관리하는 Traffic Monitor입니다.

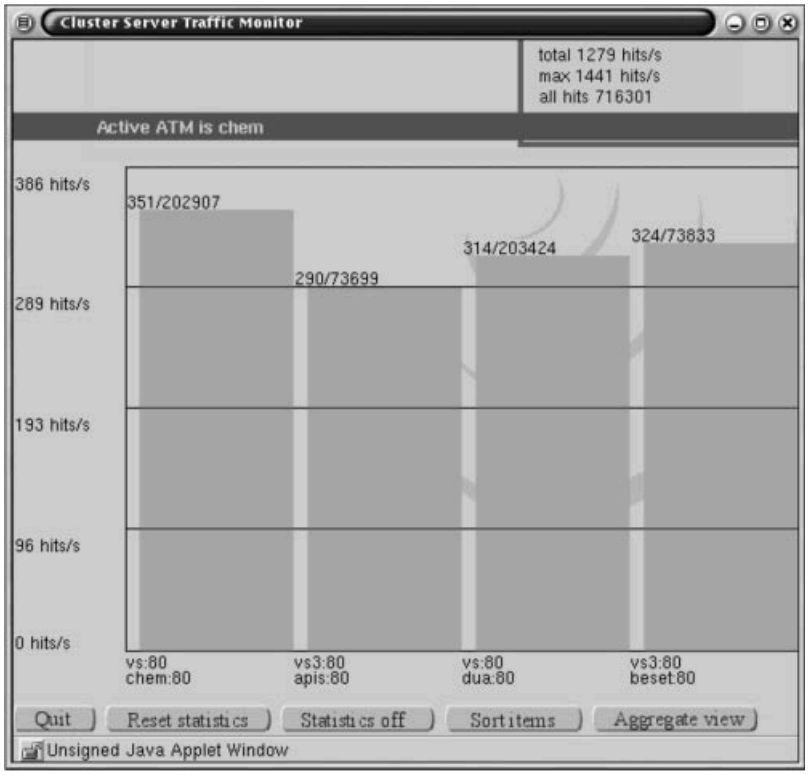

그림 7.7 Traffic Monitor.

문제 해결

문제 해결은 클러스터를 관리하는 데 있어서 매우 중요한 부분입니다. 이 절에서는 문제 해결에 사용할 수 있는 다양한 툴을 살펴보도록 하겠습니다. 여기에는 /proc/net/cluster 디렉토리 안의 파일과 로그 파일도 포함됩니다. 또한, 데몬 시작 순서를 살펴보고 어디에서 문제가 생겼는지 알아내는 방법도 설명하겠습니다. 마지막으로, 일반적인 문제와 그 문제를 해결하는 방법을 알아보겠습니다.

문제를 해결할 때 가장 중요한 것은 클러스터 외부의 시스템에서 클러스터를 테스트해야 한다는 것입니다. 클러스터 노드와 ATM은 클러스터 테스트에 사용할 수 없습니다. 이들의 별칭은 각각의 로컬 시스템이 서비스 요청에 로컬로 응답하게 만들기 때문입니다. 클라이언트가 반드시 개별적인 서브넷상에 있어야 할 필요는 없으며, 클러스터의 멤버가 아닌 시스템에 있기만 하면 됩니다.

주의)
클러스터를 테스트하려면 클러스터에 종속되지 않은 클라이언트 시스템에서 클러스터를 액세스해야 합니다. 클러스터에 속하는 시스템에서 클러스터를 테스트하면 작동되지 않는 클러스터가 작동 중인 것처럼 보이거나, 작동 중인 클러스터가 작동되지 않는 것처럼 보일 수도 있습니다. 클러스터의 구현 방식 때문에 클러스터 안의 시스템이 클러스터 자체를 목적지로 하는 트래픽을 처리하고, ATM은 그 트래픽을 보거나 처리하는 일이 없습니다.

로그 파일

클러스터 서버는 여러 로그 파일에 자신의 작동 정보를 기록합니다. 이 로그 파일은 다른 시스템 로그 파일과 함께 /var/log에 저장됩니다. 로그 파일을 보는 것이 약간 번거롭긴 하지만, 문제가 발생한 부분을 찾을 때는 매우 강력한 도구가 될 수 있습니다. 시스템이 정상적으로 작동될 때 로그 파일을 살펴보면 기본적인 조회도 되고, 시스템에 이상 여부도 쉽게 알아낼 수 있습니다.

주요 로그 파일은 clusterserverd.log입니다.
이 파일에는 clusterserverd 데몬이 생성하는 대부분의 출력이 저장되어 있습니다. 이 파일 안에 생성되는 몇 가지 출력은 다음 절 "데몬 시작"에서 살펴보도록 하겠습니다. 또한, 이 파일에는 모든 서버 조회와 ASA 서비스 검사에 관한 정보도 포함되어 있습니다. 어떤 서버나 서비스가 다운되면, 그에 관한 정보가 이 파일에 저장됩니다.

kernmsg 파일은 syslog 데몬이 커널 메시지를 기록하는 데 사용하는 표준 로그 파일입니다. SpeedLink 커널 모듈은 커널의 다른 부분과 마찬가지로, 자신의 출력물을 이 파일로 전송합니다.

디버깅 기능을 설정하면, 커널 모듈이 더 많은 출력을 생성하여 이 파일로 전송할 것입니다. 이 추가의 정보는 클라이언트로부터 ATM으로 들어오는 각각의 패킷과 그것이 포워딩될 클러스터 노드를 알려줍니다.
디버깅을 설정하는 방법은 /proc/net/cluster/debug에 관한 절에서 설명하도록 하겠습니다.

CMC 데몬은 주로 브라우저와 CMC 데몬 사이의 연결에 관한 정보를 cmc.log 파일에 기록합니다. 이 정보에는 SSL 암호와 키 전환뿐 아니라, ATM을 시작하고 종료하는 것 같은 press 작동 버튼도 포함됩니다.

데몬 시작

clusterserverd 데몬에서 제대로 정의된 시작 절차를 모니터링함으로써 데몬이 어떤 부분에서 정상적인 시작 절차를 벗어났는지 알 수 있습니다.

데몬이 생성하는 출력을 보려면 다음과 같은 명령을 사용하십시오.

```
# tail -f /var/log/clusterserverd.log
```

클러스터 데몬이 시작될 때 /var/log/clusterserverd.log 파일을 보면, 다음과 유사한 순서가 표시되는 것을 알 수 있습니다.

1. 데몬이 시작되고 메시지를 전송합니다.

   ```
   Starting Advanced Traffic Manager daemon
   ```

2. 작성 날짜를 포함한 버전 정보가 출력됩니다.
3. 데몬이 실행되고 있는 시스템의 이름과 IP 주소를 디스플레이합니다.

   ```
   Running on atm1.turbolinux.usa (192.168.0.1)
   ```

4. 구성 파일 이름이 표시됩니다.
   일반적으로 사용되는 구성 파일은 /etc/clusterserver/clusterserver.conf입니다
5. 구성 파일을 읽고 파싱합니다.
6. 성 파일 안에서 잘못된 라인과 그 라인의 문제점을 표시합니다.

7. 파싱이 실패하면, 데몬은 다음과 같은 메시지를 디스플레이하고 더 이상의 절차를 수행하지 않습니다.

   ```
   Bad Cluster Server configuration file! Going to idle mode
   ```

   구성 파일을 편집하여 오류를 수정하면,데몬에게
   HUP 신호를 전송하여 구성 파일을 다시 읽고 시작 절차를 계속하게 할 수 있습니다. 다음과 같은 명령을 사용하여 파일을 다시 읽으라는 신호를 전송하십시오.

   ```
   # killall -HUP clusterserverd
   ```

8. 클러스터의 브로드캐스트 주소와 네트워크 마스크가 디스플레이됩니다.
9. ip_cs 모듈이 로드됩니다(아직 실행되고 있지 않은 경우)
10. 클러스터 서버가 생성한 네트워크 인터페이스 별칭 중에서 사용되지 않는 별칭(별칭 부분이 :cs0인 것)이 삭제됩니다.
11. 시스템이 구성 파일에서 ATM으로 표시되어 있다면, 그 시스템은 백업 ATM으로 시작하도록 구성될 것입니다.
12. 시스템이 이전 단계에서 백업 ATM으로 구성되었다면, 그 시스템은 주 ATM 검색을 시도합니다.

13. 시스템이 백업 ATM이고 주 ATM을 발견하지 못했다면, 후보 ATM 선발 작업을 시작합니다. 선발 절차는 구성 파일에서 가장 먼저 나타나면서 현재 실행중인 백업 ATM을 선택하고 그것을 주 ATM으로 승격시킵니다.

14. 새로운 인터페이스 별칭을 구성합니다.
    - 시스템이 주 ATM이라면 클러스터의 가상 IP 주소로 이더넷 카드의 별칭(보통, eth0:cs0)이 구성됩니다.
    - 시스템이 직접 포워딩 노드라면, 클러스터의 가상 IP 주소로 루프백 인터페이스에 대한 별칭(lo:cs0)이 생성됩니다. 또한, ARP 응답을 억제하기 위해 /proc/sys/net/ipv4/conf/all/hidden과 /proc/sys/net /ipv4/ conf/lo/hidden에 "1"을 기록합니다.
    - 시스템이 터널링 노드라면, 클러스터의 가상 IP 주소로 tunl 인터페이스에 대한 별칭(tunl0:cs0)이 생성됩니다. 터널 인터페이스는 커널 IP-IP 모듈을 로드합니다. 데몬은 터널 인터페이스가 ARP 요청을 무시하도록 /proc/sys/net/ipv4/conf/all/hidden과 /proc/sys/net/ipv4/conf/tunl/hidden에 "1"을 기록합니다.
    - 시스템이 NAT 포워딩을 사용하는 노드인 경우, 네트워크 인터페이스가 변경되지 않습니다.
    - 시스템이 주 ATM이고 NAT 방법을 사용하는 노드를 갖고 있다면, NAT 게이트웨이 주소가 이더넷 카드에 대한 별칭으로 생성될 것입니다. 이것은 게이트웨이 주소와 동일한 서브넷에 있는 실제 IP 주소에 따라 eth0나 eth1이 될 수 있습니다. 별칭은 eth0:natg 같은 이름을 갖습니다.

15. 시스템이 주 ATM인 경우, 서버와 서비스 검사를 시작합니다.
- 각 클러스터 노드(그것이 noping 옵션으로 구성되어 있지 않다면)를 검사합니다.
- 각 노드에 있는 모든 서비스에 대해 ASA가 실행됩니다.
  가동되고 있지 않은 서버나 서비스가 발견되면, 그것은 down으로 표시되고 커널 테이블에서 임시로 삭제됩니다.

16.데몬은 종료하라는 신호를 받을 때까지 대기합니다. 주 ATM인 경우, 종료 신호를 받을 때까지 서비스와 서버 검사를 계속 수행합니다.

17.데몬은 종료 신호를 받으면 작업을 정리하고 종료되고, 시스템이 클러스터 노드로 구성되었다면 IP 별칭은 그대로 남아 있을 것입니다. 시스템이 ATM 역할만 하고 있었다면, 별칭이 삭제됩니다.

- config
- connections
- debug
- nat
- servers
- services
- stat
- timeout

/proc/net/cluster 사용

SppedLink 커널 모듈은 클러스터 트래픽을 포워딩하고 여러 가지 내부 테이블을 관리하는 것 이외에도, /proc 안에 /proc/net/cluster라는 디렉토리를 만듭니다. 이 디렉토리는 클러스터와 관련된 문제를 해결할 때 매우 유용합니다. /proc/net/cluster 디렉토리 안의 파일에 작성된 값은 커널 모듈에 있는 변수의 값을 직접 변경할 수 있습니다. 이 파일은 클러스터 서버 데몬이 커널 모듈과 통신하는 수단입니다.

대부분의 경우에는, 데몬이 이 매개 변수 변경을 처리하게 하십시오. 하지만, 현재 값을 읽을 수 있으려면 각 매매 변수가 어떤 의미를 갖는지 이해해야 합니다. 이 파일은 문제를 디버깅하는 데도 사용할 수 있습니다.

주의)

/proc/net/cluster 디렉토리 안의 파일에 부적절한 값을 작성하면 시스템이 중단될 수도 있습니다. 따라서, 클러스터 서버와 CMC 데몬이 이 파일을 변경하게 하는 것이 좋습니다. 꼭 필요한 경우가 아니면 직접 변경하지 마십시오.

CMC에서 이들 파일의 대부분을 보고 몇 가지 매개 변수를 변경할 수 있습니다. 이 기능은 CMC 안에 있는 Status 페이지의 'Internal Module Status'라는 제목 아래 있습니다. CMC는 파일에 들어 있는 각 정보가 어떤 의미를 갖는지 알려줍니다. 이제부터 /proc/net/cluster 안에 있는 각 파일의 용도와 의미를 살펴보도록 하겠습니다.

/proc/net/cluster/config

/proc/net/cluster/config 파일은 세 가지 중요한 데이터 구조의 크기를 저장합니다. 그 세 가지는 서비스의 개수, 서버의 개수, 그리고 클라이언트 커넥션의 개수입니다. 이 설정은 파일에 직접 작성하여 동적으로 변경할 수 있습니다. 예를 들어, 테이블 크기를 25개의 서비스, 10개의 서버, 5000개의 커넥션으로 변경하려면, 다음과 같은 명령을 실행하십시오.

```
# echo 25 10 5000 > /proc/net/cluster/config
```

파일을 다시 읽으면 변경 사항이 적용되었는지 검사할 수 있습니다.

```
# cat /proc/net/cluster/config
25 10 5000
```

주의)
이 파일에 작성하면, SpeedLink 모듈이 리셋되어, 가동중인 모든 커넥션이 중단됩니다.

/proc/net/cluster/connections

connections 파일에는 클라이언트/서버 쌍으로 구성된 테이블이 들어 있습니다. 이 파일은 현재 가동 중인 커넥션이나 영속 커넥션을 디스플레이하는 데 사용할 수 있습니다. 각 커넥션은 한 줄로 표시됩니다. 파일 안의 각 라인은 다음과 같은 형식을 갖습니다.

```
prot client:port cluster:port timeout node:port packets
```

prot 프로토콜: 'tcp'나 'udp'.

client:port 클러스터 노드 IP 주소와 포트.

cluster:port 클러스터의 가상 IP 주소와 서비스의 포트 번호.

timeout 커넥션이 타임 아웃되기 전까지의 시간(초 단위).

node:port 패킷이 포워딩될 클러스터 노드 IP 주소와 포트 번호.

packets 포워딩될 패킷의 개수

다음 예는 1.2.3.4에 있는 클라이언트 시스템으로부터 IP 주소가 192.168.0.100인 클러스터로의 HTTP(포트 80) 커넥션입니다. 이 경우, 패킷은 192.168.0.100에 있는 클러스터 노드로 전송될 것입니다.

```
tcp 1.2.3.4:9645 192.168.0.100:80 98 192.168.0.4:80 113
```

NAT 커넥션은 두 개의 라인을 갖습니다. 하나는 들어오는 커넥션을 위한 것이고 다른 하나는 ATM과 클러스터 노드 사이의 커넥션을 위한 것입니다. 이것은 RFC 1631에 의해 규정된 NAT의 구현 방식 때문에 생긴 부작용입니다. ATM과 클러스터 노드 간 커넥션은 NAT서브넷 상에서 선택된 주소를 소스 주소로 나타냅니다.

/proc/net/cluster/debug

debug 파일은 추가의 디버깅 정보를 기록할 것인지 결정하는 데 사용됩니다. 일반적으로 이 파일은 0으로 설정되며, 이 값은 정상적인 로깅 정보만 출력될 것임을 의미합니다. 이 파일을 1로 설정하면, 추가의 정보가 기록될 것입니다. 이 때 실행하는 명령은 다음과 같습니다.

```
# echo 1 > /proc/net/cluster/debug
```

추가의 로깅 정보는 ip_cs SpeedLink 커널 모듈로부터 제공되어, /var/log/kernmsg 파일에 작성됩니다. 이정보는 가상 서버에 대한 새로운 커넥션과 트래픽이 포워딩될 노드를 알려줍니다. 이와 같은 추가의 로그 메시지를 활성화시키는 것은 ATM의 성능에 막대한 영향을 끼칠 수 있으므로, 클러스터와 관련된 문제를 디버깅할 때만 사용해야 합니다.

/proc/net/cluster/nat

/proc/net/cluster/nat 파일에는 NAT 포워딩과 관련된 구성 설정이 저장되어 있습니다. 이것은 구성 파일과 turboclusteradmin 툴의 NAT Subnet 설정과 거의 똑같습니다. 차이가 있다면, 구성 파일은 IP 주소와 서브넷마스크를 사용하고, nat 파일은 IP 주소와 서브넷 마스크 안의 비트 수를 지정한다는 것입니다. 여러분의 구성파일은 다음과 같을 것입니다.

```
NAT
  Subnet 10.0.0.0 255.255.0.0
EndNAT
```

the nat file will look like this:

```
10.0.0.0 16
```

NAT Gateway 설정은 clusterserverd 데몬에 의해서만 사용되기 때문에 이 파일에는 나타나지 않습니다. 커널은 NAT Gateway를 신경 쓸 필요가 없습니다.

/proc/net/cluster/servers

servers 파일은 클러스터 안의 각 서버 노드에서 실행되고 있는 서비스에 대한 정보를 저장합니다.
각 서비스마다 한 라인씩 정보가 저장됩니다. 이 정보는 구성 툴의 'Server' 섹션에 있는 것과 똑같습니다.
각 라인은 다음과 같은형식을 갖습니다.

```
prot node:port cluster:port up weight method packets
```

prot 프로토콜: 'tcp'나 'udp'.

node:port 클러스터 노드 IP 주소와 서비스에 대한 포트 번호

cluster:port 클러스터의 가상 IP 주소와 서비스의 포트 번호

up 서버와 서비스가 실행되고 있는지에 따라 'up'이나 'down'

weight 서버의 가중치를 나타내는 숫자. 숫자가 클수록 상대적으로 서버가 더 많은 트래픽을 받습니다.

Method 서버에서 사용될 포워딩 방법. 'local'이면 노드가 주 ATM의 역할도 함을 나타냅니다.

packets 서버 상의 서비스로 포워딩된 패킷의 개수

/proc/net/cluster/services

services 파일에는 클러스터의 가상 IP 주소와 그 클러스터가 처리하는 모든 서비스에 대한 포트 번호가 들어 있습니다. 각 라인은 다음과 같은 형식을 갖습니다.

```
prot cluster:port up persistence packets
```

prot	프로토콜: 'tcp'나 'udp'.
cluster:port	클러스터의 가상 IP 주소와 서비스의 포트 번호
up	'up'이나 'down'
persistence`	서비스가 "sticky"나 영속으로 설정되어 있으면 1이 되고, 그렇지 않으면 0입니다.
packets	클러스터 상의 서비스로 포워딩된 패킷의 개수

/proc/net/cluster/stat

stat 파일에는 클러스터의 작동과 관련된 통계치가 저장되어 있습니다. 이 수치는 실시간으로 업데이트되어, 트래픽 관리자가 여러 노드로 패킷을 전달함에 따라 그 내용을 보여줍니다. 이 파일에 작성하는 것은 아무 효과가 없습니다.

다음과 같은6개의 값이 디스플레이됩니다.

- 구성된 서비스의 개수
- 현재 클러스터에 있는 서버 노드의 개수 x 각 서버가 처리하는 서비스의 개수(servers 파일에 있는 라인의 수와 똑같음)
- 현재 활성인 커넥션의 개수
- 클러스터가 받은 패킷의 총 개수
- 중단된 패킷의 개수
- 새로운 커넥션의 개수

/proc/net/cluster/timeout

timeout 파일은 타임 아웃 시간을 변경하는 데 사용됩니다. 커넥션이 주어진 시간 동안 어떤 트래픽도 받지 않으면, 그 커넥션은 사용되지 않는 것으로 간주되어 닫힐 것입니다. 타임아웃 값을 변경하려면 이 파일에 해당 숫자를 작성하십시오.

```
# echo 100 > /proc/net/cluster/timeout
```

하지만, 클러스터를 재시작하면, 이 파일에 기록된 값은 기억되지 않습니다. 커넥션 타임아웃 값을 영구적으로 변경하려면, 클러스터 관리 툴에 있는 'Advanced Traffic Manager Settings' 메뉴를 사용해야 합니다.

일반적으로 발생하는 문제

이 절에서는 일반적으로 발생할 수 있는 문제와 그 문제를 해결하는 방법을 알아보겠습니다. 더 자세한 정보를 알고 싶으면 RELEASE.NOTES 파일을 확인하시기 바랍니다. 본 설명서보다 더 최근의 정보를 제공할 것입니다. CMC 홈 페이지나 `turboclusteradmin`를 통해 개정판 정보나 다른 도큐멘테이션을 액세스할 수도 있습니다.

동기화 툴이 실패할 경우

동기화 툴에 대한 요구 사항은 여러 가지가 있습니다. 무엇보다도 내용을 받는 서버는 반드시 sshd(Secure Shell 데몬)를 실행하고 있어야 합니다. 대부분의 경우, Windows NT와 Windows 2000 시스템은 SSH를 실행하지 않을 것이므로, 동기화 과정에 참여할 수 없습니다. SSH를 설치하지 않은 다른 시스템도 마찬가지입니다.

클러스터 서버를 설치한 모든 시스템은 sshd 데몬을 설치하고 실행해야 합니다. `tlcs_content_sync`를 사용할 때는 특히 그렇습니다. 구성 정보를 받을 시스템은 클러스터 서버를 실행하고 있을 것이므로, 반드시 SSH를 설치해야 합니다.

여러분은 확인해야 할 것은 모든 클러스터 노드에 대한 `/etc/hosts.allow` 파일입니다. 이 파일에서 들어오는 SSH 트래픽이 사용 가능하게 설정되어 있어야 합니다. 이에 해당하는 라인은 다음과 같습니다.

```
sshd : ALL
```

필요에 따라, SSH 커넥션을 클러스터 안의 시스템이나 LAN으로 제한할 수도 있습니다. 동기화될 서버 목록에서 SSH가 설치되지 않은 서버를 삭제하면 동기화 툴에서 경고 메시지가 나타나지 않습니다. 이러한 서버의 내용은 직접 동기화해야 합니다.

클러스터 작동 상태 검사

클러스터가 작동되는지 검사하려면, CMC를 사용하거나 /proc/net/cluster 파일을 직접 조회하여 클러스터의 활동을 감시할 수 있습니다. 아마도 connections과 stat 파일이 가장 유용할 것입니다.

클러스터를 테스트하기 위한 트래픽을 생성할 때는, 항상 클러스터에 속하지 않는 시스템에서 클라이언트 커넥션을 만드십시오. 클러스터 안의 노드에서 클러스터링된 서비스에 연결하려고 하면, 그 커넥션은 ATM을 통과하지 않을 것입니다. 이는 트래픽이 클러스터의 가상 IP 주소로 전송되기는 하지만, 클러스터 노드가 그 주소로 전송될 트래픽을 받아들이도록 강제로 설정되었기 때문입니다. 트래픽이 ATM을 통과하지 않으므로, 이 트래픽은 클러스터를 작동되게 만드는 포워딩 절차에 의해 영향을 받지 않습니다.

클러스터가 다운되는 ATM이나 클러스터 노드를 제대로 처리하는지 검사하려면, 해당 시스템을 오프라인으로 만들어보십시오. 가장 쉬운 방법은 시스템에서 네트워크 테이블을 제거하는 것입니다. 클러스터를 원하는 대로 구성한 후에는 바로 안정성을 검사하시기 바랍니다. 클러스터가 잘못 구성되었다면 시스템에서 실제로 문제가 발생하기 전에 발견하는 것이 좋을 것입니다.

클러스터 노드를 사용할 수 없게 만들어서 테스트할 때, ATM의 로그 파일을 조회하면 ping과 ASA가 실패했으며 시스템이 클러스터에서 제거될 것임을 알려줄 것입니다. 이 경우, 그 시스템으로 만든 모든 커넥션이 중단됩니다. 그렇지 않다면 서비스가 정상적으로 계속됩니다.

주 ATM을 사용할 수 없게 만들면, 백업 ATM이 그 사실을 감지하고 주 ATM으로 승격될 ATM을 선발합니다. 몇 초안에 정상적인 서비스가 다시 시작됩니다. ATM이 다운될 때 활성이었던 커넥션은 모두 중단되지만, 새로운 커넥션은 아무 문제 없이 초기화됩니다.

주 ATM 알아내기

클러스터에 대한 ATM 목록에서 가장 먼저 가동되는 시스템이 주 ATM이 되고 목록의 다른 모든 시스템은 백업 ATM이 됩니다. 따라서, 특정 시스템이 주 ATM이 되게 하려면, 그것이 `clusterserverd` 데몬을 가장 먼저 실행하는 ATM이 되게 하십시오.

원래 주 ATM이 다운되었다가 다시 가동되어도, 그것은 주 ATM으로 승격되지 않습니다. 현재 주 ATM은 다운되지 않는 한 계속해서 주 ATM으로 유지됩니다.

어떤 시스템이 주 ATM이 될 것인지 결정하는 가장 좋은 방법은 가상 IP 주소에 대해 CMC를 사용하는 것입니다. 이렇게 하면 항상 주 ATM으로 연결됩니다. ATM의 이름은 아이콘 줄의 바로 아래 출력될 것입니다.

각 시스템에 대한 로그 파일을 검사하여 그 시스템의 역할이 무엇인지 알아낼 수도 있습니다. 주어진 시스템이 주 ATM인지 알아내는 또 한가지 방법은 `ifconfig`의 출력을 확인하는 것입니다. 네트워크 별칭(`:cs0`)이 실제 네트워크 인터페이스(가령, `eth0`)에 대해 생성된다면, 그 시스템이 주 ATM입니다. 다른 시스템은 루프백이나 터널 인터페이스에 대한 별칭을 갖습니다.

클러스터가 지나치게 많은 트래픽을 추가 생성할 경우

NAT 설정을 잘못 구성하면, 네트워크에 과다한 양의 트래픽이 생기데 됩니다. NAT 설정이 적절한지 다시 확인하십시오. 모든 NAT 시스템이 제대로 작동된다면, 과다한 트래픽은 발생하지 않을 것입니다.

제 8 장

클러스터 서버

터보리눅스 클러스터 서버는 서로 다른 몇 개의 구성 요소들로 이루어안 있습니다. 그리고, 이 요소들은 로드 밸런싱, fail-over, 그리고 높은 가용성과 같은 클러스터링의 기능을 구현하기 위해 함께 동작합니다. 또한, 이들 중에는 커널 스페이스에서 실행되는 것, 데몬으로 실행되는 것, 그리고 명령줄에서 실행되는 것이 각각 있습니다.

본 장에서는, 각 요소들에 대해서 살펴보고, 또한 이들의 종합적인 기능에 대해 알아볼 것입니다.
본 장의 정보는 클러스터를 동작시키는데 반드시 필요한 것은 아니지만, 여러분들의 전반적인 이해를 도와, 문제가 발생할 경우 더 효과적으로 해결할 수 있도록 할 것입니다.
다음은 본 장에서 다룰 주요 내용입니다.

- 스피드링크 커널 모듈
- 클러스터 서버 데몬(clusterserverd)
- 애플리케이션 안정용 에이전트 (ASA)
- 동기화 툴
- 클러스터 관리 콘솔(CMC)
- 상기 요소들의 결합 방식

스피드링크 커널 모듈

클러스터 서버의 심장은 스피드링크 커널 모듈이며, 이것은 수신된 IP 트래픽을 처리하고 모든 송신 여부를 결정합니다. 스피드링크 모듈은 커널 패치와 모듈 자체의 두 부분으로 나뉘어안 있습니다.

커널 패치

스피드링크 모듈이 수신된 패킷을 살펴볼 수 있게 하려면, 리눅스 커널에 약간의 패치가 필요합니다. 이 패치의 기능은 커널 모듈이, TCP/IP 프로토콜 스택의 아주 낮은 위치에서 패킷을 살펴볼 수 있게 해주는 것입니다. 그러면, 모듈은 패킷이 접촉 없이 TCP/IP스택을 통과해 지나가게 할 수 있고, 혹은 다른 머신으로 전송되도록 결정을 내릴 수도 있습니다.

커널 패치는 설치 CD에서 찾을 수 있는데, 이름은 TLCS-ip-cs.patch이며, 커널 소스 디렉터리에 위치하고 있습니다. (kernel/TurboLinux/2.2.16-0.4 와 kernel/RedHat/2.2.16-3) 또한, 이 패치는 단순한 ASCII 텍스트로 되어 있어서, less나 기타 텍스트 뷰어를 통해 살펴볼 수 있습니다.

커널 패치의 크기는 아주 작고, 주된 기능은 스피드링크 모듈이 스스로의 기능을 할 수 있도록 인터페이스를 만들어주는 것이며, 이 작업은 IP프로토콜의 아주 낮은 레벨에서 수행되기 때문에 패킷은 작업의 초기에 입출력 지정될 수 있습니다. 패치에서 수정된 것들 중 대부분은 NAT전송 방법을 구현하기 위한 것입니다.

ip_cs Module

스피드링크 모듈 자체의 이름은 ip_cs인데, 이 모듈이 실질적인 작업이 이루어지는 곳입니다. 이 모듈은 커널 패치에 의해 지정된 장소에서 TCP/IP스택으로 플러그인되며, 이후로 모든 수신 네트워크 패킷을 살펴 클러스터로 지정되어야 할 지의 여부를 결정합니다.

이 작업은 패킷의 목적지의 어드레스를 살핌으로써 이루어지는데, 만약 목적지의 어드레스가 클러스터의 가상 IP 어드레스와 일치하면, 패킷은 후속 작업을 감안하여 ip_cs모듈을 통해 전송되고, 일치하지 않으면 정상적인 처리를 위해 TCP/IP스택으로 되돌려보내집니다.

클러스터를 어드레스로 하는 패킷을 받을 때, Ip_cs모듈은 그것을 어디로 전송해야 할지 결정하기 위해 몇 가지 작업을 수행해야 합니다. 이 작업들은 몇 개의 테이블 참조를 통해 수행됩니다. 메인 테이블에는 서비스, 서비스들, 그리고 현재의 커넥션 이렇게 세 가지가 있는데, 이 테이블의 크기는 구성설정에서 수정할 수 있습니다.

최초의 작업은 패킷을 검사해서, 그것이 존재하는 커넥션에 속한 것인지를 알아보는 것입니다. 만약 그렇다면, 그 패킷은 동일한 서버 노드로 보내지는데, 그 커넥션에 있는 패킷의 나머지들도 마찬가지입니다.

이것은 클라이언트와 서버가 세션을 만들 수 있도록 하는데, 세션이란 일종의 대화와 같은 것입니다. 클라이언트가 세션을 초기화하면, 클라이언트와 서버는 둘 중 하나가 대화가 종료되었음을 결정하기까지 패킷을 서로 교환합니다. 일단 세션이 끝나면, 정보는 현재의 커넥션 테이블에 특정 시간 동안 머물러있는데, 이것은 같은 클라이언트가 다른 유사한 대화를 시작하기 원할 경우를 대비해서 입니다.

이 시간은 /proc/net/cluster/timeout에 초단위로 정해져 있고, 구성 파일과 tlcsconfig의 'Advanced Traffic Manager Settings' 메뉴에 있는 커넥션 시간 설정을 통해 역시 정해집니다. 만약 어떤 패킷이 현재의 세션에 속해 있지 않다면, 클러스터가 다루고 있는 서비스 포트로 보내지는 것이 아닌지 검사됩니다. 만약 그렇다면, 모듈은 어떤 클러스터노드가 그 서비스 요구를 처리할 수 있는지를 결정하기 위해 서버 목록을 살펴봅니다.

서버은 일반적으로 라운드-로빈(round-robin) 방식에 의해 선택이 되지만, 각 서버의 비중과 주어진 서비스에 대한 서버의 처리 가능 여부가 역시 고려됩니다. Ip_cs모듈은 또한 /proc/net/cluster디렉터리를 다루는데, 이 디렉터리는 모듈이 결정을 내리기 위해 사용하는 테이블에 액세스하는 통로로 사용됩니다. 이 디렉터리 안의 대부분의 파일은 읽고 쓰는 것이 가능한데, 파일을 읽는 것으로는, 현재 존재하는 연결은 어떤 것들인가와 같은 정보를 얻을 수 있으며, 파일에 대한 쓰기 작업을 통해서 여러분은 모듈이 결정을 내릴 때 사용할 수 있는 서버와 서비스를 구성할 수 있습니다.

또한, /proc/net/cluster파일은 7장에서 자세하게 다룬 바 있으며, 스피드링크 모듈의 소스코드는 CD의, kernel/speedlink/src디렉터리에서 찾을 수 있습니다. 이 코드는 상당히 복잡하며, 여러 개의 소스 파일로 구성되어 있습니다. 다음에서는 수정된 커널과 함께, 이 코드를 컴파일하는 방법을 다룰 것입니다.

커널을 컴파일하기

커널을 직접 컴파일하기 위해서는 커널 패치와 ip_cs모듈의 소스 코드를 사용할 수 있으며, 설치 프로그램에서 적당한 커널을 찾을 수 없다면, Custom Kernel을 선택하고, 진행되는 절차에 따라 사용자 자신의 것을 만들 수 있습니다.

경고)
커널을 만드는 것은 단순한 작업이 아니니, 여러분의 첫 커널로서 클러스터 서버 커널을 만들려 하지 마십시오. 그보다 먼저 비생성 박스들을 위한 커널을 만들어보는 경험을 해 보십시오. 여기서 개괄적으로 다루게 될 작업은 시스템 관리자가 커널 컴파일에 경험이 있는 것을 전제로 한 것입니다.

보다 자세한 정보는, kernel-HOWTO를 참조하십시오. 한가지 기억해야 할 아주 중요한 점은 커널 구성을 가능한 한 구입시의 상태와 가깝도록 보존해야 한다는 점입니다. 그러나, 역시 클러스터 서버가 필요한 모든 옵션이 컴파일되어 커널에 존재하는지 확실하게 알 필요가 있을 것입니다. 사용 가능한 주요 옵션에는 IP Aliasing과 IP-IP터널링이 포함되는데, 대부분의 배포판은 이미 이 옵션을 가지고 있을 것입니다.

다음의 단계를 따라, 스피드링크 ip_cs모듈과 함께 클러스터 서버 패치가 적용된 커널을 만들 수 있으며, 이 절차들은 커널 버전 2.2.16-17을 예로 사용하고 있습니다. 만약 다른 커널 버전을 사용하고 있다면, 몇 가지 명령은 조정해야 할 것입니다.

1. 여러분의 배포판으로부터 커널 소스를 설치하십시오.

```
# rpm -i kernel-source-2.2.16-17.i386.rpm
```

커널 소스는 배포판과 함께 받은 소스CD에서 찾을 수 있으며, 혹은 판매자의 웹 사이트로부터 다운로드 받을 수
있습니다.

2. 스피드링크 커널 패치를 적용하십시오.

```
# cd /usr/src/linux
# patch -pl -s < /mnt/cdrom/kernel/RedHat/2.2.16-3/ \TLCS-ip-cs.patch
```

주)
패치는 만약 커널 버전 번호가 정확하게 일치하지 않더라도 작동할 것입니다. 만약 아무런 에러 메시지가 나오지 않는다면, 패치는 성공적으로 적용된 것입니다. 패치가 깨끗하게 적용되지 않으면, 수동으로 수정해야 할 것이며, 혹은 TurboLinux Cluster Server 웹 사이트에 업데이트된 패치가 있는지 알아봐야 할 것입니다.

3. 커널을 컴파일합니다.

```
# make clean
# make menuconfig
# make dep
# make bzImage
# make modules
```

4. 커널을 설치합니다.

```
# cp arch/i386/boot/bzImage /boot/vmlinuz-2.2.16-17
# /sbin/lilo
# make modules_install
```

5. 컴파일할 수 있도록 스피드링크 코드를 하드 드라이브로 복사합니다.

```
# cp -a /mnt/cdrom/kernel/speedlink/src /usr/src/ip_cs
```

6. Ip_cs모듈을 컴파일합니다.

```
# cd /usr/src/ip_cs
# make
```

7. 금방 만든 ip_cs모듈을 설치합니다.

```
# cp ip_cs.o /lib/modules/2.2.16-17/ipv4/
```

8. 재부팅하고, 부트 메뉴에서 새로운 커널을 선택합니다

9. 새로운 커널이 적당하게 작동하고 있는지 알아봅니다. 로그 파일을 검사하고 정운적이지 않은 것들이 있는지 찾아봅니다.

10. Ip_cs모듈을 로딩할 수 있는지 검사하는데, 다음의 명령을 통해 직접 로딩할 수 있습니다:

```
# modprobe ip_cs
```

만약 오류없이 로딩된다면, 로딩된 모듈의 목록에 있는지 확인합니다.

```
# lsmod
Module Size Used by
Ip_cs 26532 0 (unused)
ppp 19021 0 (autoclean) (unused)
```

11. 클러스터 데몬을 시작합니다:

```
# /etc/rc.d/init.d/clusterserverd start
```

만약 모든 것이 제대로 작동한다면, 새로운 사용자 제작 커널에서 클러스터 서버가 실행되고 있는 것입니다.

클러스터 서버 데몬(clusterserverd)

스피드링크 모듈이 터보리눅스 클러스터 서버의 심장이라면, 클러스터 서버 데몬은 중추신경계라고 할 수 있습니다.

클러스터 서버 데몬은 전체 시스템에서 두 번째로 가장 중요한 부분이며, /proc/net/cluster인터페이스를 통해 ip_cs모듈을 구성하고 제어하는데, 실제적인 트래픽 전송작업을 제외하고는 트래픽 매니저의 동작에 필요한 모든 것을 다룹니다. 데몬이 시작될 때, 데몬은 먼저 구성 파일을 읽고 시스템이 어떤 기능을 행해야 할 지를, 즉 클러스터 노드인지, ATM인지, 혹은 두 가지 모두인지를 결정합니다.

만약 시스템이 ATM이면, 데몬은 항상 그 시스템을 백업 ATM으로 만들면서 시작합니다. 그리고는, 프라이머리 ATM이 이미 있는지의 여부를 확인하는데, 만일 없을 시에는, 백업 ATM은 어떤 한 ATM이 프라이머리 ATM으로 승격되도록 선택할 것입니다. 프라이머리 ATM으로는, 구성 파일의 리스트에 맨 처음으로 등재된 ATM을 선택합니다.

데몬은 또한 네트워크 인터페이스 설정 작업을 수행하는데, 이에 관한 내용은 마지막 장의 문제해결 부분에서 다룬 바 있습니다. 데몬은 시작시 다이렉트 전송이나 터널링, 혹은 NAT등을 사용하는 클러스터 노드로서 시스템을 구성할 수 있고, 적당한 IP어드레스 및 옵션과 함께 별칭이 붙은 인터페이스를 설정합니다. 이런 이유로 클러스터 서버를 클러스터의 모든 노드 상에서 사용할 것을 추천하며, 그럴 경우 어떤 노드도 수동으로 구성할 필요가 없습니다.

데몬은 또한 모든 네트워크 검사 작업을 수행합니다. 여기에는, 백업 ATM이 프라이머리 ATM의 동작 사실을 알 수 있도록 프라이머리 ATM이 생성하는, 하트비트 브로드캐스트(heartbeat broadcast)가 포함됩니다. 그리고, 물론, 이런 브로트캐스트를 수신하는 작업과, 프라이머리로부터 아무런 신호가 수신되지 않을 때 백업을 프라이머리로 승격시키는 작업을 수행하는 것은 바로 백업 ATM의 데몬입니다. 덧붙여서, 서버의 핑(server pings)과 ASA 서비스 검사 작업도 데몬에 의해 수행됩니다.

서비스 검사의 경우, 구성 파일에서 정해진 외부 ASA프로그램이 사용되며, 만약 어떤 서버나 서비스가 실패하면 데몬은 커널 모듈에 의해 유지되는 리스트로부터 임시로 외부 프로그램을 삭제합니다.

클러스터 서버 데몬은 클러스터에 있는 시스템간의 커뮤니케이션을 위해 두 개의 포트를 사용하는데, UDP 포트 17100은 하트비트 브로드캐스트로 사용되며, 프라이머리 ATM은 이 포트를 통해 브로드캐스트를 보내고, 백업 ATM은 브로드캐스트를 수신하는 것으로, 프라이머리가 아직도 작동하고 있다는 것을 확실하게 알게 됩니다. (이것이 모든 ATM이 같은 서브네트 상에서 있어야 한다는 유일한 이유입니다 -브로드캐스트는 시작되었던 서브네트로부터 라우팅되지 않습니다.)

또한, 데몬은 TCP 포트 17101을 관리 포트로 사용하는데, 이 포트는 주로 clusterserverd와 CMC데몬 사이의 커뮤니케이션 채널로 사용됩니다. 이 포트로 오직 로컬 액세스만 가능하도록 하기 위해서는 구성 프로그램에서 'Security Setting'을 사용해야 합니다. /etc/services파일로 엔트리를 추가함으로써 데몬이 사용하는 포트 번호를 바꿀 수 있습니다.

'클러스터서버' 서비스는 UDP의 핵심이고, 기본값은 17100입니다. 'clusterserveradm' 서비스는 기본값 17101로, 관리 채널들을 위해 사용됩니다. 만약 두개의 다른 클러스터를 동일한 서브네트 상에서 실행시키고 싶다면, 클러스터들 중 하나 상에서 이 포트 번호를 바꿔주어야만 합니다. 그렇지않으면, 클러스터는 자신의 하트비트 브로드캐스트들로 인해 서로를 혼동할 것입니다. 만약 이 값들을 바꾼다면, 그 클러스터의 모든 ATM에 대해 바꿔야 한다는 사실을 기억하십시오, 그렇지 않을 경우, 시스템들은 잘못된 포트들을 리스닝하게 됩니다.

경고)
로컬 호스트(127.0.0.1/255.255.255.255)로의 액세스만을 허용하고, 다른 모든 것(0.0.0.0/0.0.0.0)은 거절하도록 'Security Setting'이 설정되어 있어야 합니다. 그렇지 않으면, 비인증 사용자가 여러분의 클러스터의 구성을 바꿀 수도 있습니다.

애플리케이션 안정용 에이전트 (ASA)

ASA는 간단한 서비스 검사를 실행하는 프로그램 혹은 스크립트입니다. 이는 ATM 상에서 실행되며 클러스터의 각 서버노드 상의 서비스 포트들로 연결되는데, 서비스가 정상적으로 실행되고 있음을 확인하기 위해 간단한 트랜잭션을 실행합니다. 이런 작업을 거치지 않고 ATM은 노드들이 응답 불가능한 상태인데도 클러스터 노드들로 서비스 요청을 보낼 수도 있습니다.

몇 가지 다른 클러스터링 솔루션들의 경우, 완전한 기능의 에이전트가 제공되지 않는 경우도 있습니다. ASA의 장점은 서버가 포트 연결에 응답할 수 있음을 확인하게 해준다는 것 뿐만 아니라, 그포트에 추가된 서비스도 요청 응답이 가능하다는 것을 확인시켜 준다는 사실입니다.

ASA는 프라이머리 ATM상에서 작동되는데, 구성 파일의 'UserCheck' 섹션과 `tlcsconfig` 프로그램의 'Services'섹션에서 정해집니다.

구성 프로그램의 'Server Groups' 섹션의 'Check Service Frequency'설정에서 정해준 시기마다 이들은 호출되며, 만약 ASA가 'Check Service Timeout'에서 정해준 시간 내에 응답을 얻지 못하면, 서비스는 다운된 것으로 간주됩니다(이런 설정들은 구성파일에서 'CheckPortFrequency'와 'CheckPortTimeout'로 명명되어 있습니다)

ATM이 ASA를 호출할 때, ATM은 몇 개의 인수들을 에이전트로 보내는데, 첫 번째 인수는 검사할 클러스터 노드의 IP어드레스, 두 번째는 검사할 포트 번호, 그리고 마지막 인수는 서비스가 UDP상에서 실행되는지, 혹은 TCP상에서 실행되는지의 여부를 나타냅니다.

서비스가 TCP라면, 마지막 인수는 1일 것이고, UDP라면, 2가 보내질 것입니다. 에이전트는 특정한 노드상에서 서비스에 연결되도록 주어진 정보를 사용할 것인데, 서비스가 가능하면0을 반환할 것이고, 그렇지 않을 경우에, 1을 반환할 것입니다.서비스가 다운된 것으로 보일 때는, 그 서비스는임시로 ip_cs모듈의 테이블로부터 삭제됩니다.

또한, 서비스가 다운될 때, 다른 스크립트가 실행될 수도 있습니다. 이 스크립트는 tlcsconfig프로그램 중에서는 'Event triggered when down'으로 불리고, 구성 파일 중에서는 'UserCheck'섹션 내에서 'Down'으로 이름 붙여안 있습니다.

어떤 다운된 서비스의 백업시 호출되는 상응 스크립트가 있는데, 이 업 앤드 다운 스크립트들은 ASA 자체와 마찬가지로 동일한 인수로 호출됩니다.

'Down'스크립트는 서비스를 백업하려는데 도움이 되며, ATM상에서 실행될 것이나, 서비스를 다운시켰던 서버 노드의 이름이 전송될 것이라는 점을 기억하십시오. 그래서, 스크립트는 다른 메커니즘으로 서버에 접촉해서, 서비스를 실행시키기 위해 서버의 이름을 사용해야만 할 것입니다. 원격 서비스 상에서 명령들을 실행시키기 위해 SSH를 사용하는 것은 이런 스크립트를 만들고 있을 때에 유용할 수도 있습니다.

'Up'과 'Down'스크립트들의 다른 가능한 용도는 fail-over들에 대한 I/O 펜싱이며, 이것은 주어진 시간에 오직 하나의 시스템만이 공유된 자원에 액세스한다는 사실을 확인하는데 사용됩니다. /var/log/clusterserverd.log 파일에서 ASA가 검사하는 것을 모니터할 수 있으며, 이 검사들에는 'info 021'이라는 접두사가 붙어있습니다.

서비스가 시작되면, 'info 007'의 메시지가 로그되고, 서버의 이름과 포트의번호가 주어집니다. 서비스가 다운되면,
'info 009'메시지가 로그되는데, 여기 몇몇의 ASA검사들을 보여주는 예제 로그가 있습니다:

```
08/06 10:12:01 info 021 Checking cluster1.tl.usa service
                        node1.tl.usa:80 (pid 143)
08/06 10:12:05 info 007 Service node1.tl.usa:80 is up
08/06 10:21:41 info 009 Service node1.tl.usa:80 is down
```

다음의 ASA는 클러스터 서버와 함께 작동합니다.

- DB2Agent
- dnsAgent
- ftpAgent

- genericAgent
- httpAgent
- httpsAgent
- http10Agent
- imapAgent
- nntpAgent
- oracleAgent
- popAgent
- smtpAgent

이 에이전트들은 실행할 수 있도록 만들어지지만, 쉘 스크립트나 Perl 스크립트로 쉽게 쓰여졌을 수도 있습
각 부분들에 대한 더 자세한 것은 CMC홈페이지에서 이용 가능한 매뉴얼 페이지 엔트리를 읽어보면 알 수
있습니다.

동기화 툴

동기화 툴은 클러스터에 있는 노드들 사이에 일관성(coherency)을 제공합니다. 이에 관한 내용은 전 장에서 다룬 바 있고, 본 섹션에서는, 그들이 어떻게 구현되는지에 대해서 설명할 것입니다.

동기화 툴은 시스템 간 파일 전송을 위해Secure 셸이나 SSH를 사용합니다. 소스 시스템은 언제나 유틸리티가 실행되고 있는 시스템입니다. 리스트에 있는 다른 시스템들은 컨텐트를 전달 받았으며, 다시 되돌릴 수 없다는 사실에 주의하십시오. 일단 바뀐 사항들을 보내고 나면, 소스 시스템에 있는 것은 어떤 것이든 다른 시스템들로 전송됩니다.

실제 파일 전송은 scp프로그램에 의해서 실행됩니다. SSH 통합 소프트웨어는 다른 모든 프로그램들처럼, 각 시스템에 처음으로 연결될 때에는, 패스워드를 입력해야만 할 것입니다. 각 시스템에 대해서 루트 패스워드를 사용하십시오. SSH는 연결 설정을 기억할 것이기에, 따라서 최초로 성공적으로 연결된 다음부터는 패스워드를 입력할 필요가 없을 것입니다.

또한, 패스워드는 암호화되어 있기 때문에, 패스워드가 시스템에 단순한 텍스트로 남아 있을 것에 대해 염려할 필요가 없습니다. SSH는 항상 안전하게 작업을 수행합니다. 구성 동기화 프로그램(tlcs_config_sync)은 컨텐트 동기화의 단지 특별한 경우이며, /etc/clusterserver디렉터리 안의 파일들을 복사해서 모든 시스템들이 정확하게 같은 방식으로 구성되게 합니다.

또한, 라이센스 파일들도 역시 복사해서 모든 시스템들이 실행될 수 있게 합니다.

터보리눅스 클러스터 서버와 함께 설치된 SSH의 버전은 F-Secure SSH 1.3.7인데, 다른 클러스터 노드들 상에 다른 SSH 버전이 있다면, 설사 그들이 클러스터 서버 소프트웨어를 실행시키고 있지 않은 경우라 할지라도, 그것들 역시 동기화 작업에 포함시킬 수 있습니다.

그러나, 다른 SSH버전들 사이에 호환되지 않는 것으로 알려진 것들이 있기 때문에, 만약 어떤 문제가 있다면 버전을 바꿔볼 필요가 있을 것입니다. SSH가 설치되어 있지 않은 시스템들은 동기화 프로그램들의 서버들의 목록에서 제거되어야만 하는데, 그런 시스템서는 동기화를 수동으로 해줄 필요가 있습니다.

만일 컨텐트 동기화의 필요성이 더욱 강조되는 상황이라면 더 강력한 솔루션을 찾아 볼 수도 있습니다. rsync시스템은 여러 시스템들 간에서 데이터를 동기화시키는데 아주 적합하며, 다중 소스들로부터 동기화될 필요가 있는 것들이 무엇인지도 알아낼 수 있습니다.

ssync라는 안정적인 버전도 구현되었는데, 2장에서 언급된 것처럼, 궁극적인 솔루션은 모든 노드들 사이에서 자속적인 컨텐트를 유지하는 분산 파일 시스템을 분산 배치하거나, 하드웨어를 공유하는 저장 솔루션을 사용하는 것입니다.

클러스터 관리 콘솔(CMC)

CMC는 클러스터의 동작을 모니터하고, 구성에 동적 변화를 줄 수 있는 웹 인터페이스를 제공합니다. 전장에서는 클러스터를 관리하기 위해 CMC를 사용하는 법에 대해서 이야기했고, 본 섹션에서는 그것이 어떻게 이뤄지는지 설명할 것입니다.

CMC는 클러스터의 각 ATM상에서 데몬으로 실행되며, 그것은 /etc/rc.d/init.d/cmcd의 초기화 스크립트와 함께 시작됩니다. 데몬작업은 cmc로 불리는데, 이것은 ps x 리스팅을 할 경우, 나타날 것입니다. (실제로 그것은 두 번 나타나는데, 그 중 하나는 입력되는 들어오는 연결을 수신할 때이고, 다른 한 번은 ATM 상에서 트래픽을 모니터할 때입니다.) CMC 데몬은 안전한 (secure) HTTP 연결을 위해서 TCP 포트 910을 통해서 수신합니다.

브라우저가 포트 910으로 연결을 시작할 때, CMC 데몬은 클라이언트가 스스로를 인증할 것을 요청합니다. 커뮤니케이션은 SSL 프로토콜을 이용하기 때문에 안전하며, 패스워드는 네트워크를 통해 전달 되기 전에 암호화될 것입니다.

브라우저와 ATM사이에서 전달되는 정보 역시 암호화 될 것이며, 그러므로 CMC에 입력되거나 보여지는 어떤 정보도 중간에서 가로챌 수 없습니다. 브라우저는 사용자와 패스워드를 묻는 내용을 표시할 것인데, 'tlcsadmin'사용자 계정은 CMC에 연결될 때, 반드시 사용되어야 합니다.

'root' 계정을 역시 사용할 수 있지만, 그럴 경우, 기능이 저하될 수도 있습니다. 'tlcsadmin'계정은 설치시에 생성되며, 'root' 사용자와 같은 패스워드를 초기에 가지게 될 것입니다. 최초로 CMC에 연결할 때, 브라우저는 CMC 시스템 상에서 보안 증명서에 대해서 알리는 경고 창을 띄울 것입니다.

SSL 증명은 터보리눅스 클러스터 서버를 설치할 때, 생성됩니다. 여러분은 VeriSign 혹은 Thawte 같은 상업적인 증명 기관들에 의해서 정당함을 인정 받은 자신의 증명서를 가질 수 있습니다. 그러나, 증명서를 얻는 작업은 다소 복잡하며 반드시 필요하지는 않지만, 만약 그런 증명서를 얻게 되면, /etc/clusterserver 디렉터리 안의 cmc-cert-key.pem 파일에 저장하십시오.

CMC는 /proc/net/cluster파일들을 통해서 커널 모듈로부터 대부분의 정보를 수집합니다. CMC의 Status 페이지상의 'Internal Module Status'섹션들은 /proc 파일들로부터 직접 얻을 수 있는데, 이들은 포인이 잘 되어 있고, 몇 가지 설정들을 동적으로(dynamically) 바꿀 수 있게도 해 줍니다.

설정들을 동적으로 바꿀 경우, ATM이 재 시동되면, 바뀐 변화들은 남아있지 않는다는 사실을 주의 하십시오. 설정들을 영구적으로 바꾸기 위해서는, 구성 파일에 변화를 줘야 할 필요가 있을 것입니다.

Status페이지 상의 다른 정보는 명령라인 유틸리티들과 로그 파일들을 사용해서 수집되며, 명령들과 파일 이름들은 각 섹션에 목록이 있습니다. 또한, ATM의 clusterserverd 데몬은 CMC에 의해서 정지될 수 있고, 시작될 수도 있습니다.

만일 프라이머리 ATM을 정지 혹은 재시동할 경우, 백업 ATM들이 프라이머리 ATM의 역할들을 넘겨 받을 수도 있기 때문에, 살펴보고 있던 시스템은 재시동할 경우, 더 이상 프라이머리 ATM이 아닐 수 있다는 사실을 숙지하기 바랍니다.

앞에서 다뤄졌듯이, CMC데몬은 클러스터 서버 데몬과 TCP 포트 17101을 통해서 통신하며, 이 통신 채널을 통해 CMC는 clusterserverd 데몬의 몇 부분들을 제어할 수 있습니다.

종합적인 기능

clusterserverd 데몬은 모든 일들이 잘 수행되게 하는 역할을 합니다. 이것은 구성 파일을 읽어서 무슨 일을 해야 할 지 결정을 내리며, 네트워크 인터페이스들이 올바로 구성되었는지를 확인합니다.

또한, 필요할 경우, 커널 모듈을 로딩하고, /proc/net/cluster파일들을 사용해서 구성합니다.
데몬은 /etc/rc.d/init.d/clusterserverd스타트업 스크립트를 통해서 시작되며, 다른 스타트업 스크 립트들과 마찬가지로, 서비스를 시작하거나 마치기 위해서 start, stop그리고 restart옵션들을 조정할 수 있습니다.

스크립트는 Sys V init 스크립트 작업을 이용해서 부팅 시에 정상적으로 호출됩니다.
구성 설정들 중 몇 가지는 데몬 자체에 의해서 사용되며, 몇 가지는 스피드링크 커널 모듈을 구성하는데 사용되는 것을 주의하십시오.

예를 들면, ATM설정들에는 ARP설정들, 연기(delay) 설정들, 테이블들의 엔트리의 최대 숫자, 그리고 연결 타임아웃값(timeout value) 등이 포함됩니다. 테이블의 크기와 연결 타임아웃은 커널 모듈을 구성하는데 쓰이며, 설정들의 나머지는 데몬 자체에 의해 사용됩니다.

커널 모듈은 각각의 패킷을 어디로 보낼지를 결정하고, 패킷들을 전송하는 등의, 트래픽 관리 작업의 대부분을 감당합니다. 그러나, 스피드링크 모듈이 어떤 작업을 해야 할지를 지시하는 것은 데몬입니다. 또한, 데몬은 서버들과 서비스들을 모니터하며, 그들이 다운되었거나 혹은 응답하지 않는 것으로 드러나면, 커널 테이블로부터 이들을 제거합니다.

CMC는 클러스터의 동작을 모니터하는 부가적인 툴인데, 이 일은 /proc/net/cluster파일들을 계속해서 지켜보는 것과 작업을 도표화하는 것을 통해서 이루어집니다. 그러나, 커널 모듈 설정들과 함께 데몬의 설정들을 수정하기 위해서도 역시 사용될 수 있으며, 모듈이 시작, 재시동, 혹은 정지할 때를 지시할 수도 있습니다.

CMC는 clusterserverd 데몬과의 통신으로 이런 작업들의 대부분을 수행하며, 실제적으로 cmc라는 하나의 데몬 자체로 동작하는데, /etc/rc.d/init.d/cmcd안에는 CMC만의 스타트업 스크립트가 있습니다.

동기화 툴들은 클러스터안에서 노드들을 일관성 있게 관리할 수 있도록 해 줍니다. 효과적으로 클러스터를 작동시키려면, 일관성을 유지해야 하는데, 여기에는 컨텐트뿐만 아니라, 구성 설정들도 포함됩니다.

결론

이런 다양한 요소들의 결합을 통해, 클러스터 서버 시스템의 목적은 성취됩니다. 보다 향상된 속도, 신뢰도, 그리고 확장성을 위해 여러 컴퓨터들이 결합될 수 있습니다. 즉, 모든 작업을 하나의 거대한 서버에 집중시키는 대신, 작업 로드는 여러 서버들로 분산되고, 이를 통해 시스템은 서비스의 가용성을 극도로 증가 시키는 동시, 어떤 사소한 오류의 가능성도 없도록 셋업될 수 있습니다.

모든 것들이 어떻게 함께 맞물려 동작하는지를 배우는 데에는 시간과 노력이 소요되지만, 그렇게 되면 마침내 신뢰할 수 있고, 확장 시스템을 가지게 될 것이며, 모든 것들의 기능을 확실하게 이해한 후에는, 클러스터를 관리하는 것이 아주 자연스러울 것입니다.

이와 관련된 기술을 잘 조합하면 비용과 관리 시간을 절약하면서 소기의 목적을 달성할 수 있을 것입니다.

용어 풀이

본 용어집에는 관련 약어와 용어 그리고 터보리눅스 매뉴얼에서 사용된 용어가 정의되어 있습니다.
터보리눅스는 약어집의 항목들에 포함된 내용들이 다음의 소스들을 근거로 하고 있음을 밝혀 두는 바입니다.
본 내용에 관한 권리는 각각의 내용 제공자들에게 있습니다.

- Operating System Concepts (제5판) Silberschatz & Galvin.
- Developer's Resources Copyright ⓒ 1999 Emmett Dixson
- Linux.com 웹사이트, ⓒ1999, 2000 Linux.com
- Transaction Processing Performance Council의 문서들
- 리눅스를 비롯한 타 공개 소스 웹사이트에서 정보를 제공한 기고자

A

고급 트래픽 관리자 (ATM)

터보리눅스 클러스터 서버를 위한 트래픽 관리자. 목적지가 클러스터인 트래픽을 개별 클러스터노드들로 라우팅하고, 각각의 패킷이 전송될 곳을 결정하며, 클러스터내의 각 노드의 지속적인사용가능여부를 지능적으로 결정합니다.

또한, 시스템뿐만 아니라, 애플리케이션의 운태를 확인하기 위해 각각의 시스템을 지속적으로점검하고, 개별

노드의 성능을 감지할 수 있고 수신된 트래픽을 가장 잘 처리할 수 있는 시스템으로 배분할 수 있습니다. *cluster, node* 참조.

관련 링크: http://www.turbolinux.co.kr/products/tcs/

에이전트(agent)

Application Stability Agent 참조.

Apache(아파치)

아파치는 공개 소스 라이센스 하에 배포된 무료로 이용이 가능한 웹 서버입니다. 강력하고 유연한아파치 httpd서버는 가장 최근의 프로토콜들을 구현한 HTTP/1.1 compliant 웹서버로서 타사모듈들과 함께 고급 구성 및 확장이 가능하며, 모든 소스코드를 공개하고 있으며, 사용 권한에 대해 제한을 두고 있지 않습니다. 또한, 윈도우 NT/9x, 넷웨어(Netware) 5.x, OS/2, 그리고 대부분의 유닉스 버전들을 비롯한 여러 운영체제에서 모두 실행 가능합니다.

관련 링크: http://www.apache.org/

API

API (애플리케이션 프로그램 인터페이스)는 프로그램 작성 프로그래머들이 운영체제나 애플리케이션을 요구할 수 있도록 컴퓨터 운영체제나 다른 애플리케이션 프로그램에 의해 규정된 특별한 방법입니다. API는 운영체제나 프로그램을 위한 인터페이스로서, 그래픽 유저 인터페이스나 커맨드 인터페이스(둘 다 다이렉트 유저 인터페이스임)와 대조됩니다.

관련 링크: http://www.whatis.com/api.htm

Application Stability Agent (ASA)

클러스터 노드운에서 특정 서비스가 실행중인 지의 여부를 결정하는데 사용되는 프로그램입니다. 에이전트는 연결 가능 여부를 확인하는 일과 함께 대개 단순한 트랜잭션을 수행합니다.

ARP

Address Resolution Protocol(어드레스 도출 규약)의 약자로 ARP는 IP 어드레스들을 하드웨어(MAC) 어드레스들로 변환시킵니다. 일단 공통 캡슐화 메커니즘이 이더넷을 위해 선택되면, 호스트는 32 비트의 IP 어드레스를 48비트의 이더넷 어드레스로 변환시켜야 합니다. RFC 826에 문서화되어 있는 ARP가 이 작업에 사용됩니다. 이것은 FDDI같은 다른 매체에도 역시 채택되어 사용됩니다.
ARP는 이더넷에 첨부된 모든호스트에 패킷을 브로드캐스팅함으로써 작동하게 되고, 이 패킷에는 송신자가 통신을 원하는 IP어드레스가 포함되어 있습니다. 대부분의 호스트는 이 패킷을 무시하지만, 패킷 내의 IP어드레스가 자신의 것과 일치하는 대운 머신은 응답을 보냅니다. 호스트들은 IP와 하드웨어 간 어드레스 매핑에 거의 변동이 없다는 가정 하에 ARP응답 캐시를 유지합니다.
관련 링크: http://webopedia.internet.com/TERM/A/ARP.html

ASA

Application Stability Agent 참조.

ATM

고급 트래픽 관리자 참조.

B

백업 ATM

주 ATM이 다운 되는 경우, 그 역할을 대신 수행하기 위해 준비된 시스템. 백업 ATM은 기본적으로 주 ATM의 fail-over 시스템이라 할 수 있습니다.

Bash

Bash는 유닉스 명령어 번역기(셸)입니다. 이것은 Posix 1003.2 쉘 표준의 구현물(implementation)이라 할 수 있으며, Korn and System V 셸과 비슷합니다. 대화식 용도와 쉘 프로그래밍을 목적으로 하는 이런 셸들보다도, Bash에는 보다 고급의 기능들이 다수 포함되어 있습니다. 대화식 용도로 제공되는 기능들 중에는 명령줄 입력, 명령 히스토리(history), 작업 제어, 별칭, 그리고 확장 용이성 등이 있고,

프로그래밍적

기능으로는 부가적 변수 확장, 쉘 연산, 그리고 셸의 동작을 제어하는 수많은 변수들과 옵션들이 있습니다.

Bash는 원래 공개 소프트웨어 재단(Free Software Foundation)의 Brian Fox에 의해 만들어졌으며, 현재의 개발 및 유지 보수 책임자는 Case Western Reserve 대학의 Chet Ramey가 맡고 있습니다. 2.04가 가장 최근 버전으로, 2000년 3월 17일에 공개되었습니다.

관련 링크: ftp://ftp.cwru.edu/pub/bash/FAQ

Beowulf

리눅스 시스템들에 프로세스용(processing) 클러스터들을 구현시키는데 사용되는 클러스터링 기술로서 Beowulf는 운품 그 자체가 아니라 여러 기술들을 결합한 형태라 할 수 있습니다. 터보리눅스 클러스터 서버와 Beowulf가 같은 형태의 클러스터링을 구현하지는 않는다는 사실을 주의하십시오.

Beowulf 는 CPU 위주의 작업에 쓰이는 반면, 클러스터 서버는 서비스 지향 작업에 사용됩니다. Beowulf 는 리눅스를 실행하는 표준 PC들의 클러스터로 구성된 수퍼 컴퓨터를 만들기 위한 하나의 시도라 할 수 있습니다.

이 경우, 일반적으로 PC는 이더넷을통해 연결되고 병렬 프로세싱용 프로그램들을 실행시킵니다. 서버 노드는 나머지 클러스터로 데이터를 입력하고, 관리 시스템으로 동작합니다.

관련 링크: http://www.beowulf.org/.

BIND

BIND (Berkeley Internet Name Domain)는 DNS (Domain Name System) 프로토콜의 구현물이며 아래에 있는DNS의 주요 구성 요소들을 구현할 수 있는 공개적 재배포가 가능한 레퍼런스를 제공합니다.

- DNS서버(named)
- DNS 리졸버 라이브러리
- DNS 서버 작동 운태 확인에 사용되는 툴

관련 링크: http://www.isc.org/products/BIND/docs/bind8.2_highlights.html

BIOS (기본 입출력 시스템)

정확하게 말하면 ROM BIOS입니다. BIOS는 컴퓨터 시동시 컴퓨터의 운영을 책임지도록 프로그램된 읽기 전용 메모리(ROM) 칩 셋입니다. PC를 처음 켤 때 화면에 출력되는 메시지는 모두 BIOS가 제어합니다. BIOS는 CMOS 구성 칩을 통해 하드 드라이브의 번호와 사용중인 플로피 디스크의 크기와 같은 구성 정보를 알아냅니다.

BOOTP

BOOTP(부팅 프로토콜)은 클라이언트 컴퓨터가 자신의 IP 주소, 서버 호스트 주소, 기억장치에 로딩되어 실행되는 파일 이름 등을 알아낼 때 사용됩니다. 자세한 내용은 RFC 951을 참조하십시오.

관련 링크: http://www.cis.ohio-state.edu/hypertext/information/rfc.html.

BSD

BSD (Berkeley Software Distribution)는 버클리에 소재한 캘리포니아 주립 대학으로부터 개발되고 배포된 특별 한 버전의 유닉스 운영체제를 참조합니다. "BSD"는 관례운 특정 BSD시스템의 배포판의 레벨을 표시하는 번호를 먼저 기록합니다(예: "4.3 BSD"). BSD 유닉스는 널리 사용되어왔고, 유닉스 시스템의 많은 운용 구현물들은 이것을 기초로 했거나 내부에 몇 개의 BSD코드를 가지고 있습니다.

관련 링크: http://www.ee.byu.edu/unix-faq/subsubsection3_8_3_2.html

C

클라이언트/서버, 클라이언트-서버

서버의 작업과 클라이언트의 작업 간에 소프트웨어가 분배되는 분산 시스템의 공통적인 형태. 일부 프로토콜에 따라 클라이언트는 서버 측에 특정 작업 또는 정보를 요청하고 서버는 응답합니다. 즉, 주문 양식에 의거해 주문을 하는 고객과 운품과 송장을 급송하는 공급자의 경우를 들 수 있습니다. 주문 양식과 송장은 이 경우에 통신을 위해 사용되는 프로토콜의 역할을 수행합니다. 서버에는 중앙 집중된 서버와 몇 개로 분산 서버가 있습니다. 이 모델의 경우, 클라이언트와 서버는 네트워크운의 노드나 , 원하는 기능(예를 들면, 빠른 서버/저렴한 클라이언트와 같은)에 적합한 다른 하드웨어와 운영 체제 운에서 독립적으로 존재할 수 있습니다. 예를 들면, DNS에서의 이름-서버/이름-변환기의 관계, NFS에서의 파일-서버/파일-클라이언트의 관계, 그리고 X 윈도우 시스템에서 분리된 스크린 서버/클라이언트 애플리케이션 등을 들 수 있습니다.

관련 링크:
http://foldoc.doc.ic.ac.uk/foldoc/foldoc.cgi?query=client%2Fserver&action=Search

클러스터, 클러스터링

클러스터는 개별적으로 뿐만 아니라, 특정 유닛 단위로도 액세스가 가능한 둘 이운의 컴퓨터 집합이라 할 수 있습니다. 클러스터링 기술을 통해 사용자는 다중 서버들을 묶어 단일의 고성능 서버를 구축할 수 있습니다. 이 기술은 Digital Equipment사에 의해 최초로 개발되었습니다. 클러스터링은 병렬 프로세싱, 로드 밸런싱, 그리고 장애 처리 등에 사용됩니다. 클러스터링은 병렬 프로세싱 애플리케이션들을 구현하는 용도로 널리 알려진 방법인데, 그 이유는 많은 기업들로 하여금 PC와 워크스테이션에 이미 투자된 것을 이용할 수 있도록 해 주었기 때문입니다.

이와 관련하여, 네트워크에 새로운 한 개의 PC를 추가함으로써 새로운 CPU들을 여러 개 추가하는 방법은 운대적으로 쉬운 작업이라 할 수 있습니다.

관련 링크:

http://www.linuxyes.com/en/scenter/cluster.html

http://metalab.unc.edu/mdw/HOWTO/Parallel-Processing-HOWTO-3.html#ss3.1

CMC(클러스터 관리 콘솔)

웹을 기반으로 하는 관리 프로그램으로, 웹 브라우저로부터 클러스터를 모니터하고 수정할 수 있도록 해 줍니다.

클러스터 관리자

고급 트래픽 관리자 참조.

클러스터 노드

서비스 요구를 실질적으로 처리하는 클러스터 내의 컴퓨터. 클러스터 관리자는 작업 로드를 클러스터 노드 간에 배분합니다. 클라이언트에게는 어떤 클러스터 노드가 그들의 요구를 처리할지의 여부가 중요하지 않으며, 대개 히트(hit)할 노드를 결정할 수도 없습니다. 터보리눅스 클러스터 서버에서, 노드는 클러스터가 지원하는 네트워크 서비스들을 수행합니다.

clusterserverd(클러스터서버 데몬)

터보리눅스 클러스터 서버를 구현하는데 쓰이는 데몬으로, 주 ATM 운에서는, 수신된 트래픽을 해당 클러스터 노드로 라우팅하기 위해 스피드링크 커널 모듈과 연결되어 작동합니다. 그리고, 백업 ATM운에서는, 주 ATM을 모니터하고, 만일 프라이머리 ATM이 다운될 경우, fail-over를 준비합니다. 또한, 클러스터 노드 운에서는 노드로서의 동작 준비를 위해 네트워크 인터페이스를 셋업합니다.

연결(connection)

연결은 두 개의 네트워크 호스트 간의 통신 세션입니다. 호스트(클라이언트)는 다른 시스템(서버)과 대화를 시작하는데, 연결은 클라이언트와 서버가 서로에게 데이터를 보내는 대화와도 같습니다. 연결은 양쪽 모두에 의해 종료될 수 있습니다.

CMC

클러스터 관리 콘솔 참조.

CPU (중앙 처리 장치)

CPU는 컴퓨터의 다른 모든 부분을 제어하며, 보통 마이크로프로세서라고도 합니다. CPU의 역할은 기억장치와 다른 하드웨어 구성요소(하드 드라이브, 네트워크 및 기타 주변 장치 카드) 사이의 정보 흐름을 제어하는 것입니다. 터보리눅스는 인텔 CPU를 비롯해서 이와 호환 가능한 CPU들(AMD, Cyrix 등)과, 모토로라와 IBM의 PowerPC CPU에서도 실행이 가능합니다.

D

데몬(daemon)

유닉스 용어 중에서 데몬은 관례운 서버 프로그램을 의미합니다. 데몬은 다른 프로그램으로부터 요청을 받았을 때에만 실행되는 메모리 운주 프로그램입니다. FTP나 TELNET같은 서버 프로그램은 일반적으로 데몬들로 구현됩니다. 대부분의 데몬 프로그램의 이름은 데몬임을 표시하는 문자 "d"로 끝납니다. 유닉스 시스템들은, 주로 네트워크 운의 다른 호스트들로부터의 서비스 요청을 처리하기 위해 많은 데몬을 실행시킵니다. 이들 중 대부분은 계속 실행되기보다는 inetd라는 단일 실제 데몬에 의해 요청을 받을 때 비로소 시작됩니다. 데몬의 예로는, crond (로컬 시한 명령 실행), rshd (원격 명령 실행), rlogind와 telnetd (원격
로그인), ftpd, nfsd (파일 전송), lpd (인쇄)가 있습니다.
관련 링크: http://foldoc.doc.ic.ac.uk/foldoc/foldoc.cgi?daemon.

데이터그램(datagram)

Internet's Request for Comments 1594의 내용을 인용하여 데이터그램을 정의해 보면, "그 자체로 완전한 독립적 데이터로서, 소스와 대운 컴퓨터 그리고 전송 네트워크 간의 사전의 교환 여부에 운관없이, 그 소스로부터 대운 컴퓨터로 라우팅될 수 있는 충분한 정보를 처리합니다. 이 영어는 일반적으로 패킷이라는 말로 대체되어 왔습니다.

데이터그램이나 패킷은 인터넷 프로토콜이 처리하며 인터넷이 전달하는 메시지의유닛입니다. 이들은 사전의 교환 여부에 운관없이 그 자체로 완전해야 하는데, 그 이유는 두 개의 통신 포인트 간에(대부분의 음성 전화 통화에서의 경우처럼) 연결이 고정적으로 자속되지 않기 때문 입니다(이런 종류의 프로토콜은 연결이 필요 없는 것으로 간주됨).

관련 링크: http://www.whatis.com/datagram.htm

DHCP

DHCP(동적 호스트 구성 프로토콜)는 구성 정보를 TCP/IP 네트워크 운의 호스트들로 보내는 기본 틀을 제공합니다. 이것은 Bootstrap 프로토콜(BOOTP)을 기반으로 하고, 재사용 가능한 네트워크 어드레스들과 추가 구성 옵션들을 자동으로 할당하는 고급 기능을 제공합니다.

DHCP는 BOOTP 중계 에이전트의 동작을 인식하며, DHCP 구성요소는 BOOTP 구성요소와 운호 작동이 가능합니다. 서버는 이 DHCP를 통해 IP 어드레싱과 구성 정보를 클라이언트들에게 동적으로 배분할 수 있습니다. DHCP는 두 개의 부분, 즉 DHCP서버에 호스트로 특정 호스트의 구성 변수들을 전달하는 프로토콜 부분과 호스트에 네트워크 어드레스를 할당하는 메커니즘 부분으로 이루어져 있습니다.

또한, DHCP는 클라이언트-서버 형태로 만들어졌는데, 이 형태에서는 지정된 DHCP 서버가 호스트에 네트워크 어드레스를 할당하고, 구성 매개 변수들을 그 동적으로 구성된 호스트들에게 전달합니다. 구성 매개 변수와 기타 제어 정보는 태그된 데이터 항목으로 전달 되어 DHCP 메시지의 "options" 필드에 저장됩니다. 데이터 항목 자체를 "options"라고도 합니다. 자세한 내용은 RFC 1533과 1534, 951을 참조하시기 바랍니다.

관련 링크: http://www.cis.ohio-state.edu/hypertext/information/rfc.html

직접 전송(direct forwarding)

ATM이 클러스터 노드의 MAC어드레스로 패킷을 직접 보내는 전송 방식. ATM과 노드는 반드시 같은 서버네트운에 운주해야 합니다. 응답 트래픽은 ATM을 통하지 않고도 클러스터 노드에서 클라이언트로 직접 전송됩니다.

분산 시스템(distributed systems)

자동 장치(대개 서로 종류가 다른)의 집합으로, 시스템이 단일의 로컬 시스템처럼 보일 수 있도록 분산 운태가 사용자에게 그대로 드러납니다. 이 방식은 특정 머신이 존재하고 있다는 사실은 알고 있으나, 정확한 위치, 저장소, 로드 밸런싱, 그리고 기능 등에 대해서는 알 수 없는 네트워크의 경우와는 대조를 이룹니다. 분산 시스템의 경우, 대개 클라이언트-서버 구성을 채택하고 있습니다.

DNS(도메인명 서비스)

도메인에 이름을 붙이는 것과 그 중에서도 가장 눈에 띄는 요소인 도메인명 서비스는 인터넷의 운영에 있어서 매우 중요합니다. DNS는 범용 목적으로 배포 및 복제되고 인터넷에서 사용되는 데이터 조회 서비스로서, fred.test.com.us와 같은 정식 도메인 이름 (FQDN)을 192.168.23.10 같은 숫자로 구성된 IP어드레스로 변환합니다. DNS는 일치하는 도메인을 찾을 때까지 검색 중인 이름에 포함되어 있는 도메인을 중심으로, 순서대로 이름 서버를 사용하도록 구성할 수 있습니다. 또한, 터보리눅스에서는 nslookup 명령을 사용해서 대화식으로 DNS를 조회할 수 있습니다.

Name Server 참조.

도메인명(domain name)

도메인명은 대개 인터넷 도메인을 인용하는데, 이것이 공통된 인터넷 명명 규칙의 기본이 됩니다. 예를 들면, www.cnn.com은 도메인 이름이며, cnn.com은 도메인 입니다.

관련 링크: http://www.freesoft.org/CIE/Topics/10.htm

E

캡슐화

캡슐화는 프로토콜 계층(layering)의 개념과 밀접한 연관이 있는데, 특정 프로토콜의 메시지 속에 다른 프로토콜을 사용해서 데이터를 집어넣는 작업과 관련이 있습니다. 캡슐화 기능을 이용하기 위해서는 캡슐화 작업을 행하는 프로토콜의 말단부가 개발되어 임의의 데이터가 그 메시지 속에 위치될 수 있어야 합니다. 그러면 다른 프로토콜은 그 데이터의 형태를 정의하는 작업에 사용될 수 있습니다.

관련 링크: http://www.freesoft.org/CIE/Topics/18.htm

이더넷 (Ethernet)

이더넷은 근거리 통신망(local area network)의 한 형태이며, Xerox, Intel, 그리고 Digital Equipment사에 의해1970년대 후반에 처음으로 개발되었고, 자세한 구조 원리는 1980에 최초로 알려졌습니다. 이더넷은 동축 케이블을 통해 초당 10메가 비트의 비율로 데이터를 전송하도록 처음에 고안되었고, 이런 최초의 표준은 근거리 통신망을 통한 케이블링, 커넥터, 그리고 데이터, 음성, 비디오의 다른 전송 특성들을 10Mbps로 규정하였습니다. 최근에는 이 속도가100 Mbps까지 증가되었습니다.

Ext2fs (Second Extended Filesystem)

이것은 리눅스 운에서 만들어진 파일 시스템으로, 빠르고 신뢰할만하며 모든 리눅스 배포판에서 사용되고 있습니다. 리눅스 파티션으로도 불리는데, Ext2 파일 시스템은 블록들의 불완전한 이용으로 인해 공간이 과다하게 낭비되는 것을 방지하기 위해 단편들(fragments)을 사용합니다. 리눅스에서의 단편의 크기는 물리적인 블록의 크기가 되어야 합니다. 그러므로 리눅스에서의 파일은 연속적인 단편들에 이어지는 블록들의 연속이라 할 수 있습니다. 또한, 완전한 하나의 블록을 구성할 수 있을 정도로 충분한 단편들이 파일의 끝 부분에 있다면 단편들은 하나의 블록으로 묶일 수 있습니다. 같은 방식으로, 파일의 크기가 줄어들면 마지막 블록은 단편들로 나뉠 수도 있습니다.

관련 링크: http://e2fsprogs.sourceforge.net/ext2.html.

이벤트 모니터링

이벤트 모니터링은 애플리케이션이 작동하는 동안 이루어지는 이벤트에 대한 정보를 모으는 것인데, 병목현운, 오버플로, 트랜잭션의 종결, 그리고 애플리케이션 끊김 현운 등을 탐지하는데 유용합니다. 공간이 부족한 파일시스템, 과다한 프로세서의 사용, 혹은 탐지 및 측정 가능한 모든 대운은 이벤트가 될 수 있습니다. 클러스터 내에서, 이벤트 관리자는 클러스터의 자원들의 동작 운태를 모니터하고, 병렬 혹은 분산 프로그램에 해당 이벤트가 발생했음을 알려줍니다. 그러나, 이벤트 관리자는 이 이벤트에 대해 반응하지는 않습니다.

F

fail-over(fail-over)

병렬 방식으로 특정 처리 작업을 수행하는 둘 이운의 시스템 운에서의 장애 처리 방법. 정운적으로는 주 시스템이 모든 요청들을 처리할 것이지만, 주 시스템이 다운되면 백업 시스템이 해당 작업을 떠맡게 됩니다. 또한, 이 용어는 주 시스템이 실패했을 경우 백업 시스템이 인계 받을 작업 지정할 수도 있습니다. 이 용어를 로드 밸런싱과 비교해 보시기 바랍니다. 이 두 어휘들은 서로 유사하지만, 페일 오버는 오직 한 시스템이 요청을 처리할 것을 의미하는 반면, *로드 밸런싱*은 모든 시스템들이 병렬로 작업을 수행하는 것이 포함합니다.

방화벽(firewall)

리눅스에서의 방화벽 기능은 인터넷 운의 특정 어드레스로 송수신되는 형태의 서비스를 허용하거나 거부하도록 구성될 수 있습니다. 피하고 싶은 사이트들은 송신 혹은 수신 연결로부터 격리될 수 있고, 내부 시스템은 외부 공격으로부터 보호 받을 수 있습니다. 만일 LAN 운에 저장된 어드레스들을 사용하고 있다면, 리눅스는 인터넷에 연결하기 위해 네트워크 어드레스 변환(NAT)기능을 사용하게 됩니다.

정식 도메인명(FQDN)

인터넷 운에서의 컴퓨터의 정식 이름은 로컬 호스트명과 도메인명으로 구성됩니다. 예를 들어, "smoke"는 호스트명이고 "smoke.com.test.us"는 FQDN입니다. FQDN은 인터넷 운의 모든 호스트에 대해 고유 인터넷 주소를 지정할 수 있어야 합니다. 인터넷에 없는 몇몇 호스트에 대해서도 동일한 명명 규칙이 사용되지만, 이들 호스트는 전자 메일 어드레싱에 대해 동일한 네임스페이스(name-space)를 공유합니다.

FTP (파일 전송 프로토콜)

한 컴퓨터의 사용자가 TCP/IP 네트워크 운에 있는 다른 컴퓨터의 사용자에게 파일을 전송할 때 사용하는 클라이언트-서버 프로토콜입니다. 또한, 사용자가 파일 전송시 실행하는 클라이언트 프로그램이기도 합니다. 터보리눅스는 ncftp라는 FTP 클라이언트를 제공합니다. 익명의 FTP-인터넷 호스트에서 제공하는 대화형 서비스로, 사용자가 FTP를 이용하여 문서, 파일, 프로그램, 기타 아카이브 데이터를 전송할 수 있습니다. 좋은 예로 ftp.turbolinux.com에 있는 터보리눅스 ftp 사이트를 들 수 있습니다. 사용자는 특정 사용자 이름 "ftp" 또는 "anonymous"와 자신의 전자메일 주소를 암호로 입력하여 로그인 한 후, 공용 파일이 들어 있는 특정 디렉터리 계층을 액세스할 수 있습니다.

G

게이트웨이

게이트웨이는 라우터의 또 다른 이름입니다. 대운 어드레스가 동일한 서브네트 운에 존재하지 않을 경우, 이 기본 게이트웨이는 트래픽이 라우팅되는 라우터가 됩니다.

router참조.

일반 공용 라이센스(General Public License)

GNU 일반 공용 라이센스(GNU General Public License) 참조.

GNU

GNU는 공개 소프트웨어 재단에서 추진하고 있는 프로젝트의 일환으로, 유닉스 대체물을 자유롭게 배포할 수 있도록 하는데 그 목적을 두고 있습니다. 이 말은 "GNU's Not UNIX(GNU는 유닉스가 아니다)"라는 말의 약자입니다. 많은 GNU소프트웨어가 리눅스와 함께 제공되며, 거의 모든 소프트웨어들이 GNU 일반 대중 라이센스(General Public License) 규정을 따르고 있습니다.

관련 링크: http://www.gnu.org/

GNU 일반 공용 라이센스(GNU General Public License)

GNU프로젝트에서 나온 일반 공용 라이센스(GPL)는 소프트웨어 사용자가 무료 소프트웨어를 공유, 수정, 교환할 수 있음을 의미합니다. GNU GPL은 소프트웨어를 개발하고, 그것이 대중적으로 사용될 수 있다고 생각하는 모든 사람들을 위한 포괄적인 지침과 규칙을 담고 있습니다.

관련 링크: http://www.gnu.org/copyleft/gpl.html.

GPL

GNU 일반 공용 라이센스(GNU General Public License) 참조.

H

하트비트, 하트비트 모니터링

하트비트 모니터링은 클러스터 내의 모든 노드간의 지속적인 통신을 유지하는 시스템 서비스들로 이루어져 있습니다. 각 노드가 활성화 운태인지를 확인하는데, 하트비트 메시지는 수 초마다 클러스터내의 모든 노드로부터 운위로 거슬러 올라가 전달됩니다. 노드에 하트비트가 없으면 그 운태가 보고되기 때문에 클러스터는 자동적으로 백업 노드로 되돌아오는 리소스를 fail-over할 수 있습니다. 또한 하트비트 모니터링은 오류가 발생했을 경우에 통신 재개를 시도할 수 있고, 나머지 클러스터에게 회복 불가능한 오류 정보를 보고합니다.

이질성 (heterogeneous)

서로 비슷하지 않은 구성 요소들을 지닌 특징이라고 이야기할 수 있는 이질성은 보통 IT에서 사용되는 용어로 다른 종류의 네트워크를 포함하거나 포함될 수 있는 제품 그리고 네트워크 운호 작용이 가능한 타 제조 회사의 운품들로 이루어져 있음을 의미할 때 사용됩니다. 이질성 네트워크는 다른 제품들 간에 공통으로 사용되는 표준 하드웨어와 소프트웨어 인터페이스들로 인해 가능하며, 따라서 운호 통신이 가능합니다. 바로 인터넷이 네트워크의 한 예라 할 수 있습니다.

높은 가용성(high availability)

하드웨어나 소프트웨어의 오류에도 불구하고 서비스에 대한 가용성을 일정하게 유지하는 시스템을 말합니다. 이것은 대개 중복성을 이용한 하드웨어나 소프트웨어 구현을 통해 이루어집니다. 높은 가용성은 99.99%의 작동가능시간과 같이 가동시간 퍼센트 기준으로 측정됩니다. 높은 가용성(HA)은 필요할 때마다 수용 가능한 수준의 동작으로 데이터와 애플리케이션 액세스를 의미합니다. 높은 가용성 운태는 모든 네트워크의 자원들이 최대의 시간동안 이용이 가능한 운황을 의미합니다. 이론적으로, 가용성의 퍼센트는 100%가 될 수 없지만, 클러스터링은 가능한 한 100%에 도달할 수 있도록 도움을 제공합니다.

HA는 일반 사용자들이 보통 온전하고 이운이 없는 것으로 여기는 "시스템"의 서비스 측면을 다룬다고 볼 수 있습니다. 이런 맥락에서 볼때, 하드웨어와 소프트웨어 구성 요소들의 안정성과 성능(반응시간/작업 처리량, 분당 트랜잭션 수 등)은 시스템 가용성의 일부라 할 수 있습니다.

가용성은MTTF/(MTTF+MTTR)로 표시될 수도 있는데, 그 의미는 다음과 같습니다.

- MTTF (mean-time-to-failure, 평균 고장 시간)은 시스템이 셋업이나 복구된 후 동작하는 평균 시간 (오류 없이)이며,
- MTTR (mean-time-to-repair, 평균 수리 시간)은 오류가 발생한 시스템을 보수하거나 회복시키는데 소요되는 평균 시간입니다.

단일점 실패 현상(single point of failure) 참조.

관련 링크:
http://metalab.unc.edu/pub/Linux/ALPHA/linux-ha/High-Availability-HOWTO.html.

호스트(host)

"호스트"라는 어휘는 몇 가지 다른 문맥 속에서 각각 다른 뜻으로 사용될 수 있습니다:

- 인터넷 운에서는, 호스트는 인터넷에 있는 다른 컴퓨터로 완전한 양방향 액세스가 가능한 모든 컴퓨터를 뜻합니다. 호스트는 네트워크 번호와 함께 특정 호스트 어드레스를 가지며, 고유의 인터넷 프로토콜(IP) 어드레스를 형성합니다.
- 대규모 메인 프레임 컴퓨터 환경에서, 호스트는 메인프레임 컴퓨터를 의미합니다.
- 호스트는 그 보다 하위 기능의 장치나 프로그램에게 서비스를 제공하는 장치나 프로그램을 가리킬 수도 있습니다.

HTML

하이퍼텍스트 마크업 언어(Hypertext Markup Language, HTML)는 월드 와이드 웹(World Wide Web) 브라우저 운에서 표시될 파일에 삽입된 마크업 심벌이나 코드들의 집합을 가리킵니다. 마크업은 웹 브라우저가 어떻게 웹 페이지의 언어와 이미지를 표시해야 할지 지시해 줍니다. 개별 마크업 코드들은 일종의

요소들로 이야기됩니다(그러나, 대부분의 사람들은 태그로 이해하기도 함). 현재 사용되고 있는 HTML의 버전은 HTML 4인데, HTML은 World Wide Web Consortium (W3C)가 권장하는 표준이며, 마이크로소프트사의 인터넷 익스플로어나 넷스케이프사의 네비게이터같은 주요 브라우저들에 포함되어 있습니다. 그리고, 이들 브라우저들은 추가적인 비표준 코드들도 제공합니다.

관련 링크: http://www.w3.org/MarkUp/

HTTP

하이퍼텍스트 전송 프로토콜(The Hypertext Transfer Protocol, HTTP)은 일종의 애플리케이션 레벨의 프로토콜로서 분산, 공동작업 및 하이퍼 미디어 정보 시스템에 필요한 속도를 제공합니다. 이것은 일반적인 객체 지향 프로토콜로서, 자체의 요청 방식(명령)을 확장하여 이름 서버 및 분산 객체 관리 시스템 등과 같은 많은 작업에 사용됩니다. HTTP의 특징 중 하나는 데이터 표시 입력 방식을 채택하여 시스템이 전송되는 데이터와는 독립적으로 구축될 수 있도록 한다는 점입니다. HTTP는 1990년부터 World-Wide Web global information initiative에 의해 사용되어 왔습니다. 더 자세한 설명은 rfc1945를 참고하십시오.
관련링크: http://www.w3.org/Protocols/

I

ICMP

인터넷 제어 메시지 프로토콜(Internet Control Message Protocol, ICMP)은 호스트 서버와 인터넷의 게이트웨이 간의 메시지를 제어하고 오류를 보고하는 프로토콜입니다. ICMP는 인터넷 프로토콜(IP) 데이터그램을 이용하나, 메시지는 IP소프트웨어에 의해 처리되고, 애플리케이션 사용자에게 직접적으로 드러나지는 않습니다.
관련 링크: http://www.whatis.com/icmp.htm

IETF

인터넷 기술 특별 조사 위원회(Internet Engineering Task Force, IETF)는 인터넷 아키텍처의 발전 및 인터넷의 바람직한 사용과 관련된 네트워크 설계자, 운영자, 판매자, 그리고 연구자들의 거대한 공개 국제적 공동체입니다.
관련 링크: http://www.ietf.org/.

inetd

inetd는 가장 널리 알려진 수퍼 서버 프로그램들 중 하나이며, 기본적으로 터보리눅스에 설치되어 부트시 시스템 프로그램으로 설정됩니다. inetd대신, inetd의 확장 버전이라 할 수 있는 xinetd를 사용할 수 있습니다.
관련 링크: http://www.delorie.com/gnu/docs/glibc/libc_227.html

인터페이스

두 대의 시스템이 통신하는 경계를 의미하며, 인터페이스는 다른 장치들을 링크하는데 사용되는 하드웨어 커넥터일 수도 있고, 두 개의 소프트웨어 사이에서 통신을 가능하게 해 주는 규약일 수도 있습니다. 종종 두 시스템 사이에는 자신의 인터페이스를 연결해주는 특정 매개물이 존재하기도 합니다. 예를 들면, 두 개의 EIA-232 인터페이스는 시리얼 케이블을 통해 연결됩니다.

Intermezzo(인터메조)

인터메조는 시스템들이 디렉터리 구조를 복제할 수 있도록 하는 분산 파일 시스템입니다. 시스템은 로컬로 수정 작업을 수행하고 근접 시스템을 업데이트합니다. 만약 네트워크나 근접 시스템이 다운되더라도 시스템은 작동 운태를 유지하였다가 시스템이 백업되었을 때 수정된 사항들이 다시 통합될 수 있습니다. 이 파일 시스템의 애플리케이션 영역은 전체 시스템을 복사하는 작업 뿐만 아니라 홈 디렉터리를 모바일 컴퓨터 운에서 이용 가능하도록 하는 작업까지 수행할 수 있습니다.

관련링크:

http://inter-mezzo.org/

http://linux-ha.org/PhaseII/WhitePapers/braam/intermezzo/opc99_html/

IP (인터넷 프로토콜)

TCP/IP프로토콜 군에 해당하는 네트워크 계층으로서, 인터넷에서는 TCP/IP를 의미합니다.
터보리눅스에서TCP/IP는 주로 TurboNetCfg를 통해 구성됩니다.

IP Address(IP 어드레스)

IP어드레스는 인터넷 호스트의 유일한 32비트 인식 번호입니다.

관련 링크:

http://www.whatis.com/ipaddres.htm

http://www.3com.com/nsc/501302.html

ip_cs

스피드링크 커널 모듈의 이름으로서, 터보리눅스 클러스터 서버 트래픽 관리를 구현하기 위해 TCP/IP스택에 삽입됩니다.

IRQ (인터럽트 요구)

기술적으로 말하자면, CPU가 정운적인 작업을 중단하고 다른 작업을 시작할 때 사용하는 기능입니다. "다른 작업"이란 주로 네트워크 카드, 사운드 카드, 직렬 통신과 같은 주변 장치를 의미하는데, 예컨대 여러분이 마우스를 움직이면 인터럽트가 발생하게 됩니다. 터보리눅스 사용자가 IRQ에 관심을 가져야 하는 이유는 각 PC 장치가 고유 IRQ번호를 가져야 하는데, 선택할 수 있는 IRQ 번호는 너무 많기 때문입니다. 인터럽트 충돌은 제조업체에서 두 장치가 동일한 인터럽트를 사용하도록 설계했을 때 발생합니다. PC에 장착된 하드웨어가 많을수록 이러한 문제가 발생할 가능성이 높습니다.

K

커널

용어 정의에 의하면, 커널은 유닉스나 리눅스와 같은 다른 운영 체제들에서 필수적인 부분으로, 자원의 할당, 낮은 레벨의 하드웨어 인터페이스, 보안 등을 책임집니다. 유의어는 nucleus(중핵)입니다. 커널은 사용자 명령과 대화하는 운영 체제의 가장 외부에 있는 셸과 대조될 수 있습니다. 커널과 셸은 유닉스에서 좀 더 자주 사용되는 용어들로서, 커널(혹은 이와 비교할 만한 운영체제의 중심)은 주로 모든 요청 혹은 커널의 서비스를 완성하는 완전한 입출력 동작들을 다루는 인터럽트 처리기, 특정 프로그램이 커널의 프로세싱 시간을 어떤 순서로 공유할지를 결정하는 스케줄러, 그리고 각 프로세스가 컴퓨터를 사용할 수 있도록 하는 수퍼바이저를 포함합니다. 커널은 기억장치나 저장장치 내의 운영체제 어드레스 스페이스 관리자를 포함하며, 이들은 모든 구성요소와 다른 커널 서비스 사용자 간에 공유됩니다.

커널의 서비스는 운영체제의 다른 부분들로부터 혹은 시스템 호출로 알려진 몇 개의 특정 프로그램 인터페이스들을 통해 애플리케이션들로부터 요청을 받습니다. 커널을 구성하는 코드는 지속적으로 필요하기 때문에 대개 보호 되어, 자주 사용되지 않는 운영체제의 다른 부분들과 겹쳐지지 않도록 컴퓨터의 저장공간 속에 로딩됩니다. 마이크로커널(microkernel)은 시스템 커널의 기본적인 특징들을 구현하는 작은 모듈들을 강조하는 운영체제 설계를 감안하여 융통성 있게 구성될 수 있습니다.

리눅스 커널 참조.

관련 링크: http://www.uwsg.iu.edu/hypermail/linux/kernel/

L

대기시간(Latency)

대기시간은 문맥에 따라 다른 의미를 지닙니다. 네트워크에서 대기시간은 지연과 같은 말로서, 데이터의 패킷이 지정된 곳에서 다른 곳까지 도달하는데 걸리는 시간을 의미합니다. 다른 용도에서 보면(예: AT&T), 대기시간은 송신자에게 되돌아온 패킷을 전송함으로써 측정 가능하며, 왕복 이동(round trip) 시간이 대기시간으로 간주됩니다. 대기시간은 데이터가 한 지점에서 다른 지점으로 즉각 전송되어야 한다는 것을 전제로 합니다. 네트워크 대기시간과 관련이 있는 작업은 다음과 같습니다.

- 전파(Propagation): 특정 패킷이 한 지점에서 다른 지점으로 빛의 속도로 이동하는데 걸리는 시간입니다.
- 전송(Transmission): 매체 자체(광섬유 케이블 및 무선 등)가 지연을 발생시킵니다. 큰 패킷이 작은 패킷보다 수신되고 돌아오는데 시간이 더 걸리므로, 패킷의 크기에 따라 왕복 이동 지연 시간이 달라집니다.
- 라우터와 다른 프로세싱: 각 게이트웨이 노드의 경우, 패킷 검사 및 패킷 헤더 변경(예: 시간-라이브 필드에서의 홉 개수 변경)에 특정 시간이 소요됩니다. 다른 컴퓨터와 저장장치의 경우에는 마찬가지로 지연이 발생합니다. 각 말단에 위치한 네트워크 내에서, 스위치 및 브리지와 같은 매개 장치에서 발생하는 저장장치나 하드디스크 액세스 지연 현운에 의해 영향을 받을 수 있습니다(그러나 백본 통계 자료에서는 이런 종류의 대기시간은 아마 고려되지 않을 것입니다).

컴퓨터 시스템에서, 대기시간은 예운 반응 시간을 초과하여 감지된 반응 시간을 증가시키는 모든 지연이나 대기를 의미합니다. 컴퓨터 대기시간의 특수한 원인으로는 마이크로프로세서와 입출력 장치간 데이터 속도의 불일치와 부적당한 데이터 버퍼링 등을 들 수 있습니다. 컴퓨터 내부에서, 대기시간은 사전추출(데이터 입력요청의 필요를 기다리는), 멀티스레드, 그리고 다중 실행 스레드 병렬 기술 등에 의해 "감추어질" 수 있습니다.

LDAP

LDAP는 디렉터리 정보를 받고 관리하는 클라이언트-서버 프로토콜 스펙입니다. 이것은 원래 클라이언트가 PC운에서 X.500 디렉터리로 액세스 할 수 있는 수단으로 만들어졌는데, 독립적으로 그리고 다른 종류의 디렉터리 서버들에 의해 사용될 수 있습니다. LDAP의 최초의 구현물은 Michigan대학에서 개발되었고, 초기 버전인, 2는 Michigan대학에서 나온 소프트웨어에서 지원되었습니다. 버전 2는 RFC 1777과 RFC 1778로 만들어졌고, LDAP에는 OSI스택의 운위 계층이 필요하지 않습니다. LDAP는 구현에 있어서 보다 단순한 형태의 프로토콜(특히 클라이언트에 있어서)이며, IETF 변경 관리 하에 있었기 때문에 인터넷의 요구들을 충족시키도록 더 쉽게 발전할 수 있었습니다. LDAP 디렉터리는 다음의 레벨들을 포함하는 단순한 "트리" 계층 구조로 구성되어 있습니다.

- 루트 디렉터리(시작 지점 혹은 트리의 소스), 최초 분기점(국가로 분기됨).
  - 국가(Countries), 조직으로 분기.
    - 조직(Organizations), 조직 단위로 분기.
      - 조직단위(Organizational units)(부, 과, 등등) 개인으로 분기(엔트리 포함).
        - 개인(Individuals)사람들, 파일들, 그리고 프린터와 같은 공유 자원들)

관련 링크:
http://www.ietf.org/rfc/rfc1777.txt?number=1777
http://www.mozilla.org/directory/standards.html

LILO

다양한 로더가 있음에도 불구하고, LILO는 분명히 리눅스의 가장 널리 알려진 부트 로더입니다. 이 로더는 하드 드라이브에 운주하며, 부팅시부팅할 운영체제를 선택하고 특정 리눅스 커널을 로딩하며, 로딩시 리눅스 커널로 특별한 매개 변수들을 보낼 수 있는 부트 프롬프트를 제공합니다. LILO는 반드시 운영체제를 필요로 하지 않기 때문에 보다 빠르고, 유연하며, 독립적으로 사용할 수 있습니다. 이러한 이유로 이 로더는 리눅스 전용 시스템에 사용되고 있습니다. LILO는 커널 부트 로더인 동시에 시스템의 메인 부트 관리자로 사용될 수 있는데, 그 이유는 리눅스, OS/2, win98, NT, 그리고 기타의 여러 OS들을 로딩할 수 있기 때문입니다.

그러나, 단점이라면 드라이브의 1024번째 실린더를 초과하여 위치한 파티션(분할 영역)으로부터는 OS를 부팅할 수 없다는 점입니다. 대용량 드라이브 사용자들에게 있어서는 이것이 문제가 될 수도 있습니다. 이 문제를 일으킬 수 있는 가능성이 가장 큰 이유는 같은 하드 디스크 운에 다른 OS로서 리눅스를 설치하려고 하는 경우입니다. 만약 대용량 드라이브의 끝부분에 리눅스를 위한 분할 영역을 만들었다면, 리눅스는 부팅되지 않을 것입니다.
관련 링크: http://www.control-escape.com/bootload.html

리눅스

리눅스는 핀란드 헬싱키 대학의Linus Torvalds라는 한 학생이 취미 삼아 최초로 만들었던 운영체제입니다. Linus는 작은 유닉스 시스템 Minix에 관심이 있었고, Minix 수준을 능가하는 시스템을 개발하기로 결심했고, 결국 리눅스 커널 버전 1.0이 1994년에 공개되었습니다. 현재와 같이 모든 기능이 포함된 버전은2.2(1999년 1월 25일에 공개됨)로서, 현재까지 발전을 거듭해 오고 있습니다. 리눅스는 GNU GPL에 근거하여 개발되었으며 소스코드는 자유롭게 모두가 이용할 수 있습니다. 그러나, 이것은 리눅스와 관련된 배포판이 무료라는 것을 의미하지는 않습니다. 즉, 기업들과 개발자들은 소스코드가 사용 가능한 운태로 남아 있는 한 사용료를 요구할 수 있기 때문입니다. 리눅스는 네트워킹, 소프트웨어 개발, 그리고, 사용자를 위한 플랫폼 등의 광범위한 목적에 사용될 수 있으며, 다른 고가의 운영체제들에 비해 비용이 저렴한 최운의 대안으로 떠오르고 있습니다. 리눅스의 중추신경 시스템이며 전체 컴퓨터를 작동시키는 운영체제 코드는 커널입니다.

커널 참조.

관련 링크:

http://www.linux.org/info/index.html

http://hudsucker.easystreet.com/

리눅스 커널

리눅스 커널 자체는 일종의 하나의 바이너리로서 성능을 향운시키는데 도움이 되는데 즉, 운영체제의 기능이나 입출력 요청에 필요한 컨텍스트 전환이 필요 없기 때문입니다. 리눅스 커널에는 모듈 방식이 적용되고, 작동시 또는 모듈이 필요할 경우 자체적으로 모듈을 로드(혹은 언로드)할 수 있습니다. 모듈은 시스템 운에서 특수 커널 모드로 실행되며, 시스템 하드웨어 액세스가 전적으로 가능합니다. 모듈은 또한 바이너리 추가물을 커널에 허용하는데 사용됩니다(만약 어떤 하드웨어가 독립적이라면, 이 하드웨어가

GNU

라이센스를 위반하지 않으면서 커널 내에 모듈의 형태로 포함될 수 있도록 드라이버가 작성될 수 있습니다). 역시, 사용자가 드라이버를 작성할 경우, 시험을 위해 시스템을 재시동할 필요 없이 드라이버를 로드/언로드할 수 있도록 하는데 유용합니다.

커널 참조.

관련 링크:

http://web1.linuxhq.com/guides/LKMPG/mpg.html

http://web1.linuxhq.com/guides/NAG/node43.html

리눅스 가상 서버들

리눅스 가운 서버는 리눅스 운영 체제 운에서 작동하는 로드 밸런서(load balancer)와 함께 실제 서버의 클러스터 운에서 구축되는 확장성과 가용성이 매우 뛰어난 서버입니다. 일반 사용자들은 주로 클러스터의 아키텍처를 접하게 되며, 오직 단일 가운 서버만을 보게 됩니다.

가상 서버 참조.

관련 링크: http://www.linuxvirtualserver.org/

로드 밸런싱

컴퓨터 네트워크운에서 어떤 단일 장치에 무리가 가지 않도록 프로세스와 통신 작업을 분배하는 기능을 로드 밸런싱이라고 합니다. 로드 밸런싱은 어떤 서버에 얼마나 많은 요청이 폭주하게 될지 예측할 수 없는 네트워크 환경에서 특별히 중요합니다. 사용 회수가 매우 많은 웹 사이트들의 경우, 보통 두 개 이운의 웹 서버가 로드 밸런싱을 통해 운용되고 있습니다. 한 서버가 작업 하중에 걸리면 요청은 여분 용량이 확보된 다른 서버로 전달됩니다. 즉, 네트워크 로드 밸런싱은 수신된 IP트래픽을 다중 노드 클러스터 간에 분배하여 균형을 잡아주는 역할을 수행합니다.

M

Masquerading(위장)

IP위장은 TCP/IP 네트워크 어드레스 변경기(NAT)의 한 형태입니다.

NAT 참조.

마운트(mount)

mount명령은 이미 존재하는 경로명 위치 디렉터리에서 파일 시스템 계층 구조에 이름이 붙여진 파일시스템을 붙여넣을 때 사용합니다. 만일 디렉터리에 마운트 작업에 우선하는 어떤 컨텐트가 있다면, 그것은 파일시스템이 다시한번 언마운트되기 전까지 감춰진 채로있게 됩니다. 파일 시스템이 호스트:경로명(hostpathname)의 형태라면, 일단 이것은 NFS 파일 시스템으로 간주됩니다(nfs명령을 입력하십시오). umount 명령은 디렉터리나 파일 시스템으로 특징 지워질 수 있는 현재 마운트된 파일 시스템을 언마운트합니다. mount와 umount 명령은 fstab(5)로 표시되는 /etc/mtab안의 마운트된 된 파일 시스템의 테이블을 유지합니다.

만약 인수 없이 요청을 받으면, mount는 이 테이블의 내용을 표시할 것이며, 반면에 파일시스템이나 디렉터리 만으로 요청을 받으면 mount는 /etc/fstab파일을 검색해서 일치하는 항목을 찾아 해당 파일 시스템을 테이블 운의 항목에서 지시하는 데로 지정된 디렉터리에 마운트할 것입니다.
또한, mount는 loopback mounts를 이용해서 새로운 가운 파일 시스템을 생성시킵니다. Loopback 파일 시스템들은 대체 경로명을 이용해서 이미 존재하는 파일들에 대한 액세스가 가능하게 합니다. 일단 가운 파일 시스템이 생성되면 다른 파일 시스템은 본래의 파일 시스템에 영향을 주지 않고 내부에 마운트됩니다. 그러나, 본래의 파일 시스템 위로 계속해서 마운트된 파일 시스템들은 가운 파일 시스템의 해당 마운트 포인트가 다른 파일 시스템에 의해 가려지기 전까지는 가운 파일 시스템에 노출되어 있게 됩니다.
관련 링크:
http://linuxnewbies.editthispage.com/tips/20000118
http://anguilla.u.arizona.edu/doc_link/en_US/a_doc_lib/cmds/aixcmds3/mount.htm
http://uw7doc.sco.com/NET_nfs/nfsT.mount_cmd.html.

MySQL

MySQL은 공개 소스 데이터베이스 관리 시스템입니다. MySQL의 SQL은 Structured Query Language를 의미하며, 데이터베이스에 액세스하는데 사용되는 가장 표준화된 언어입니다. MySQL은 또한 다양한 목적들, 여러 클라이언트 프로그램들과 라이브러리들, 관리 툴들, 그리고 프로그래밍 인터페이스들을 지원하는 다중 목적 SQL서버로 구성된 클라이언트/서버 시스템입니다.
관련 링크: http://www.mysql.com/

N

이름 서버

이름 서버(도메인 서버 혹은 DNS서버로도 불림)는 사람이 읽을 수 있는 FQDN을 머신이 읽을 수 있는 111.222.333.4444와 같은 IP어드레스로 변환하는 방법을 인식하는 컴퓨터입니다. IP어드레스로부터 얻어진 호스트의 이름을 사용하거나, 반대로 분산 데이터베이스 기능을 사용해서 호스트 이름을 찾기 위해 IP어드레스를 검색합니다.

NAS

NAS (Network-Attached Storage)는 일종의 디스크 저장장치로서 네트워크 워크스테이션 사용자에게 애플리케이션을 제공하는 컴퓨터에 속해 있기 보다는 자체 네트워크 어드레스로 구성된 디스크 저장장치입니다. 부서의 저장장치 액세스 및 관리 권한을 제거함으로써 애플리케이션 프로그래밍과 파일의 속도를 향운시킬 수 있는데, 그 이유는 이들이 같은 프로세서를 놓고 경쟁하지 않아도 되기 때문입니다. NAS는 LAN(대표적인 예로, 이더넷 네트워크)에 속해 있으며, IP어드레스가 할당된 장치입니다. 파일 요청은 메인 서버에 의해 NAS파일 서버들로 매핑됩니다.
NAS는 다중 디스크 RAID시스템을 포함해서, 하드 디스크 저장장치들과, 네트워크에 속한 장치에 파일의 위치를 매핑하며 구성하는 소프트웨어들로 이루어져 있습니다. NAS는 SAN(storage-area network)로 알려진 좀 더 복잡한 저장 시스템의 일부로 발전할 수 있습니다. NAS소프트웨어는 마이크로소프트사의IPX와 NetBEUI, 노벨사의 Netware IPX, 그리고 Sun Microsystems사의 NFS가 포함하는 여러 가지 네트워크 프로토콜들을 사용할 수 있습니다. 사용자 액세스 우선권 설정 등을 비롯한 구성은 웹 브라우저를 통해 이루어질 수 있습니다.

NAT

NAT (네트워크 어드레스 변경)는 특정 네트워크 운에서 사용된 인터넷 프로토콜 어드레스를 다른 네트워크 내의 IP 어드레스로 변경된 것입니다. 일반적으로, 기업은 자체의 로컬 내부 네트워크 어드레스들을 하나 이운의 글로벌 외부 IP 어드레스로 매핑하고, 로컬 IP어드레스로 수신되어 오는 패킷들로부터 글로벌(global) IP 어드레스들을 언매핑합니다. 이 경우, 각 송신과 수신 요청은, 그 요청을 인증하거나 검사 수 있고 혹은 이전의 요청과 비교할 수 있는 변경 과정을 반드시 거쳐야 하기 때문에 보안을 강화하는데 도움이 됩니다.

NAT는 어떤 기업이 필요로 하는 글로벌IP어드레스들의 번호를 보존하고, 단일 IP어드레스를 이용해서 세계와 통신할 수 있도록 해 줍니다. NAT는 라우터와 방화벽의 일부이기도 합니다. 네트워크 관리자들은 글로벌-로컬 및 로컬-글로벌 IP 어드레스 매핑을 실행하는데 사용되는 NAT 테이블을 생성합니다. NAT는 또한 정책 라우팅과 연결되어 사용될 수 있으며, 정적으로 정의되거나 동적으로 여러 IP 어드레스들로 변경될 수 있고, 또한 변환될 수 있도록 설정 가능합니다.

Cisco의NAT 버전을 사용하면, 관리자는 다음과 같은 매핑 테이블을 생성할 수 있습니다.

- 로컬 IP 어드레스를 글로벌 IP 어드레스로 매핑.
- 로컬 IP 어드레스를 기업이 가질 IP어드레스의 순환 풀로 매핑.
- 로컬 IP 어드레스와 특정 TCP포트를 글로벌 IP 어드레스 혹은 그 풀 중 하나로 매핑.
- 로컬IP 어드레스를 라운드-로빈 방식으로 로컬 IP어드레스의 풀 중의 아무것으로나 매핑.

NAT는 RFC 1631의 일반 조건들에 기술되어 있으며, 공용 IP와 전용 IP어드레스를 구분 지어 수많은 일반 IP어드레스의 필요성을 줄여 줍니다.

관련 링크: http://www.cis.ohio-state.edu/rfc/rfc1631.txt

네트워크 인터페이스 카드

네트워크 인터페이스 카드(NIC)는 컴퓨터 내부에 설치되어 네트워크에 연결될 수 있도록 한 컴퓨터 회로기판이나 카드입니다. LAN운의 개인용 컴퓨터나 워크스테이션은 전형적으로 이더넷이나 토큰 링과 같은 LAN전송 기술을 위해 특수 고안된 네트워크 카드 인터페이스를 포함합니다. 네트워크 인터페이스 카드를 이용해서 네트워크 전용으로 운시 연결이 가능 가능합니다.

NFS (Network File System)

Sun Microsystems사에 의해 개발된 프로토콜로서, 컴퓨터가 네트워크운의 파일에 마치 스스로의 로컬 디스크에 있는 것처럼 액세스할 수 있게 합니다. 유닉스 시스템은 다른 시스템들과 파일들을 공유하기 위해 주로 NFS를 사용합니다. NFS의 경우, 클라이언트가 데이터를 유실하지 않고도 NFS 서버를 재부팅할 수 있다는 장점이 있습니다.

NIC

네트워크 인터페이스 카드 참조.

NIS (네트워크 정보 서비스, Network Information Service)

Sun Microsystems의 클라이언트-서버 프로토콜로서, 사용자 이름, 암호, 컴퓨터 이름 같은 시스템 구성 데이터를 전달하는 데 사용됩니다. NIS는 NDS(NetWare Network Directory Service), Microsoft Domains, LDAP(Lightweight Directory Access Protocol)와 약간 비슷하며, NIS와 이들 디렉터리 사이에 사용할 수 있는 게이트웨이가 있습니다.

노드

네트워크 운에서 노드는 데이터 전송의 재분산 점이나 목적지가 되는 연결점입니다. 일반적으로, 노드에는 다른 노드들로 데이터를 인식, 프로세스, 혹은 전송하는 기능이 프로그래밍되어 있습니다. 클러스터링 운에서는 클러스터 내 각 시스템을 모두 클러스터 노드 혹은 서버 노드로 지칭됩니다.
클러스터 노드 참조.

NTP

Network Time Protocol (NTP)은 각 개인의 컴퓨터의 클럭을 조정하고, 그것을 외부 시간 소스들과 지속적으로 동기화시키는데 사용되는 프로그램 그룹입니다. 시간데이터는 외부 소스들(무선 시계, 네트워크 타임서버)로부터 요청되며, 각 도메인 내의 클라이언트들로 전해집니다. 이것은 1990년대 중반에 들어 등장한 하드웨어에, 100만 분의 1초에서 1000분의 1초 사이의 정확도를 제공하기 위해 고안되었습니다.

O

OpenLDAP

OpenLDAP는 Lightweight Directory Access Protocol (LDAP)의 오픈 소스 구현물입니다.
이 통합 소프트웨어에 포함된 것들은 다음과 같습니다::

- slapd – 독립적LDAP 서버
- slurpd – 독립적LDAP 복제서버(replication server)
- LDAP 프로토콜 구현 라이브러리
- 여러 유틸리티, 툴 그리고 샘플 클라이언트

오픈 소스(Open Source)

공개 소스란 OSI(Open Source Initiative)가 소유하고 있는 일종의 인증 표시입니다. 개발자들이 공유가 자유롭고 수정 및 재배포가 가능하도록 제품을 설계하여 그들의 배포 조건이 OSI의 공개 소스 정의 부합될 경우 공개 소스 운표를 사용할 수 있습니다. 요약하자면, 배포 조건의 정의 모델에는 다음의 사항들을 요구합니다:?배포되는 소프트웨어는 어떤 제약도 없이 누구에게나 재배포가 가능해야 합니다.

- 소스코드를 이용할 수 있어야 합니다 (따라서, 사용자에 의한 개선과 수정이 가능합니다)
- 원래의 소프트웨어와 이름과 버전이 다른 소프트웨어의 개선된 버전들을 라이센스는 요구할 수 있습니다.

P

패킷

패킷은 인터넷 운에서나 혹은 다른 패킷이 사용되는 네트워크 운에서, 출발점과 목적지 사이에서 라우팅되는 데이터의 단위입니다. "Packet"이나 "datagram"은 같은 의미인데, TCP와 유사한 프로토콜인 UDP (User Datagram Protocol)에서는 datagram이란 어휘를 사용합니다.

관련 링크: http://www.whatis.com/packet.htm

패치

패치는 수정 사항을 다른 파일들에 적용시키는 파일로, 소스코드 트리에 사소한 수정을 가하는데 사용됩니다. 리눅스 커널은 버전 업그레이드가 가능한 패치에 사용할 수 있습니다.

개인용 컴퓨터 메모리 카드 규격 협회
(PCMCIA : Personal Computer Memory Card International Association)

국제 무역 기구를 가리키기도 하고, 이 기구에서 노트북 컴퓨터에 연결할 수 있는 장치(모뎀과 외부 하드 디스크 드라이버 등)에 대해 개발한 표준을 가리키기도 합니다. PCMCIA 카드는 신용 카드만한 크기이며, 1995년 이후로 이 카드는 "PC 카드"로 불리어져 왔습니다. 또한, 이 카드는 알맞은 장치를 사용해, 데스크톱 컴퓨터나 통신 랙, 기타 장비에 연결할 수도 있습니다. 터보리눅스는 PCMCIA 카드가 실행 중인 시스템에 장착되거나 탈착될 때, 자동으로 PCMCIA 카드를 검출합니다.

지속성(persistency)

클러스터링에 있어서, 지속성은 클라이언트가 클러스터 내의 동일 서버에 항운 연결할 수 있도록 해 줍니다. 클라이언트가 클러스터 내의 어떤 서버에 액세스하느냐의 문제는 대개 별로 중요하지 않지만, 어떤 경우에는 애플리케이션 서비스가 동일 서버운에서 있어야 할 때도 있습니다. 클라이언트가 클러스터에 여러 번 액세스하기 위해서는, 클러스터는 연결 간에 그 운태를 유자해야 합니다. 이것은 특정 서비스에 플래깅 하여 클라이언트가 클러스터 내의 동일 서버와 항운 연결할 수 있도록 해 주는 작업을 통해 가능합니다.

핑(ping)

핑 프로그램은 특정 인터넷 어드레스가 존재하며, 요청들을 처리할 수 있다는 것을 확인시켜 주는 기본적인 인터넷 유틸리티입니다. 핑은 사용자가 액세스하려는 호스트 컴퓨터가 실제로 작동하는지의 여부를 진단하는데 사용됩니다. 예컨대, 호스트가 핑되지 않는다면, 사용자는 FTP(File Transfer Protocol)을 사용해서 파일을 해당 호스트로 전송할 수 없을 것입니다.

핑은 또한 자체적으로 반응시간을 알아보기 위해 호스트가 사용할 수도 있습니다. 핑을 사용하면, 운징적인 도메인 이름으로부터 번호 형태의 IP어드레스를 알아낼 수도 있습니다. 핑의 의미는 대충, 온라인운의 운대방의 "주의를 끌다" 혹은 "존재 여부를 확인하다"라고 할 수 있습니다. 핑의 원리는 지정된 어드레스로 패킷을 보내 응답을 기다리는 것이라 할 수 있습니다.

관련 링크: http://www.FreeSoft.org/CIE/Topics/53.htm

PostgreSQL

PostgreSQL은 정교한 객체 관련 DBMS이며, 하위선택, 트랜잭셕, 그리고 사용자 정의 형태와 기능 등의 모든 SQL 기능들을 지원합니다. PostgreSQL은 어느 곳에서나 사용이 가능한 가장 진보된 공개 소스 데이터베이스라 할 수 있습니다.

관련 링크: http://www.postgresql.org/

PPP (Point to Point Protocol, 지점 간 프로토콜)

지점간 직렬 링크를 통해 네트워크 계층의 데이터그램(예를 들어, IP 패킷)을 전송하는 인터넷 표준으로, 만약 모뎀을 사용하고 있다면 PPP만이 유일한 프로토콜이 될 것입니다. PPP와 이더넷을 통한 TCP/IP링크 사이의 유일한 차이점은 속도입니다.

TurboPPPCfg를 이용해서 PPP모뎀 링크는 간단히 구성될 수 있습니다.

PPTP

Point-to-Point Tunneling Protocol (PPTP)이라고 불리는 통신 규약이나 프로토콜로서, 인터넷 운의 터널을 통해 가운의 개인용 네트워크(또는 사설망)의 생성을 가능하게 하려는 목적으로 제안되었습니다. 이것은 기업이 더 이운 광역 통신을 위해 선로를 차용할 필요가 없이, 공용 네트워크를 확실하게 사용할 수 있다는 것을 의미합니다. 터널링을 참조하십시오.

마이크로소프트 및 기타 업체들과 Layer2 Forwarding 그리고 Cisco Systems사에 의해 제안되고 지원된 PPTP는 새로운 인터넷 기술 특별 조사 위원회 (IETF) 표준을 위한 주요 제안들 중에 포함되었습니다. 인터넷의 지점 간 프로토콜 (PPP)의 확장물이라 할 수 있는 PPTP를 사용하면, PPP클라이언트를 지원하는 PC의 사용자는 누구나 ISP를 이용해서 사내 어느 곳에 있는 서버에라도 안전하게 연결할 수 있습니다.

관련 링크: http://www.protocols.com//pbook/ppp2.htm#PPTP

주 ATM

현재 트래픽을 라우팅하고 있는 ATM입니다.

고급 트래픽 관리자 와 백업 ATM참조.

프로토콜

특별히 네트워크를 운에서 데이터를 전송하는 방법을 나타내는 규약입니다. 낮은 레벨의 프로토콜은 어야 할 전기적이고 물리적인 표준들, 비트-바이트 발주 및 전송, 그리고 비트 스트림의 오류 검출과 수정 등을 정의합니다. 그리고, 높은 레벨의 프로토콜은, 메시지의 구문론, 터미널과 컴퓨터 간 대화, 문자 집합 , 메시지 시퀀싱 등과 같은 데이터 포맷팅을 다룹니다.많은 프로토콜들이 RFC나 OSI에 의해 정의됩니다.

핸드셰이킹(Handshaking) 참조.

관련 링크:

http://www.protocols.com/pbook/index.htm

http://foldoc.doc.ic.ac.uk/foldoc/foldoc.cgi?protocol.

프로토콜 계층(protocol layering)

프로토콜 계층은 네트워크를 기능적인 계층들로 분류하고 각 계층의 작업처리를 위한 프로토콜들을 할당함으로써 네트워킹 설계를 단순화시키는 공통 기술입니다. 예를 들면, 데이터 전송과 관련된 기능들과 연결 관리 관련 기능들을 분리된 계층과 프로토콜에서 나누는 경우를 흔히 볼 수 있습니다. 프로토콜 계층은 양호하게 정의된 소정의 작업들과 더불어, 간단한 프로토콜들을 생성합니다. 이런 프로토콜들은 유용하게 결합될 수 있으며, 개별 프로토콜들 역시 필요에 따라 특정 애플리케이션에서 제거되거나 대체될 수 있습니다.

프록시

다른 클라이언트를 대신하여 요청하기 위한 목적으로 서버와 클라이언트 두 가지의 역할을 모두 수행하는 중간 매개적인 프로그램입니다. 요청은 내부적으로 처리되기도 하고 변환되어 다른 서버로 보내져 처리되기도 합니다. 프록시는 요청 메시지 전송 전에 반드시 변환되어야 하고, 필요에 따라서는 재작성이 요구되기도 합니다. 프록시는 네트워크 방화벽을 통해 클라이언트측 포탈로 종종 사용되며, 또한 사용자의 에이전트에 의해 구현되지 않은 프로토콜을 통해 요청을 처리하는 보조 애플리케이션으로도 사용됩니다.

R

RADIUS

RADIUS (Remote Authentication Dial-In User Service)는 클라이언트/서버 프로토콜이며, 동시에 원격 액세스 서버들이 접속해 들어오는 사용자들을 인증하고, 요청된 시스템이나 서비스로 그들이 액세스할 수 있도록 허가해 주기 위해 중앙 서버와의 통신을 가능케 하는 프로그램입니다. RADIUS를 이용해 기업은 사용자의 신분명세서를 모든 원격 서버들이 공유할 수 있는 중앙 데이터베이스에 보관시킬 수 있습니다. 이로 인해, 보안 수준이 향상되고, 기업은 일괄적으로 관리되는 네트워크 지점에도 적용 가능한 정책을 수립할 수 있습니다. 중앙 서비스가 가능하다는 말은 네트워크의 통계치를 유지하는 작업과 지불 처리 등과 같은 작업이 더욱 수월해 졌다는 것을 의미합니다. Livingston사(현재는 Lucent사에 소유됨)에 의해 개발된 RADIUS는 Ascend와 다른 네트워크 운품 회사들에 의해 사용되는 실질적인 업계 표준이고 또한 IETF 표준이기도 합니다.

관련 링크:
http://www.livingston.com/marketing/whitepapers/radius_paper.html

RAID

RAID (redundant array of independent disks, 독립 디스크의 중복 배열)는 동일한 데이터를 여러 하드 디스크 운의 다른 장소에(즉, 중복해서) 저장하는 방식입니다. 여러 디스크에 데이터를 위치시킴으로써 입출력 작업이 균형 있게 배열되고, 전체 성능 수준을 향운시킵니다. 여러 디스크들이 MTBF(평균 고장 시간 간격) 시간을 증가 시키므로, 데이터 중복 저장을 통해 장애 허용치도 증가합니다.

RAID는 운영체제에 하나의 논리적 하드 디스크로 표시됩니다. RAID는 스트리핑 기술을 사용하는데, 이를 통해 각 드라이브의 저장 공간을 하나의 섹터(512 bytes)로부터 특정 메가 바이트까지 영역 분할할 수 있습니다. 모든 디스크의 스트라이프가 삽입되고 순서대로 배열됩니다. 의학용이나 다른 과학용의 이미지들과 같은 거대한 레코드가 저장되어 있는 단일 사용자 시스템에서는, 보통 스트라이프가 아주 작게 설정되며, 따라서 하나의 레코드가 모든 디스크에 걸쳐 존재하게 하여 모든 디스크를 동시에 읽어 들임으로써 액세스 시간을 단축시킬 수 있습니다. 다중 사용자 시스템에서는, 최대크기의 레코드를 보유하기에 충분한 크기의 스트라이프를 설정하는 것이 성능 향운에 도움이 됩니다. 따라서, 다중 드라이브 운에서도 중복 디스크 입출력이 가능해집니다. RAID에는 몇 가지 레벨들과 타입들이 있습니다.

관련 링크:

http://www.whatis.com/raid.htm
http://www.uni-mainz.de/~neuffer/scsi/what_is_raid.html
http://linas.org/linux/raid.html
http://linas.org/linux/Software-RAID/Software-RAID.html

rcp

rcp는 원격 복사(remote copy)를 의미하며, 파일을 네트워크 운의 다른 시스템으로 보내거나 받을 수 있게 해 줍니다. 이것은 소스와 목적지를 알려주어야 하는 copy명령처럼 동작하는데, 차이점이 있다면 copy의 소스와 목적지가 다른 시스템이 될 수도 있다는 점입니다. FTP와는 달리, rcp는 대화식 명령이 아니며 다른 시스템에 로그 인하거나 패스워드를 알려 줄 필요가 없습니다. 또한, 여러 파일들을 복사할 수 있고, 전체 디렉터리 트리를 반복적으로 복사할 수 있습니다. 물론, 다른 시스템은 rcp를 지원하는 원격 쉘 데몬(rshd)을 실행하고 있어야 합니다.

rexecd

rexecd는 remote exec daemon(원격 실행가능 데몬)을 의미하는데, 유닉스 시스템에서 만들어졌고, rexec명령을 실행합니다. rexec명령으로부터 전해지는 연결을 수신하며(TCP/IP운에서), 연결이 되면 액세스를 확인하고 특정 프로그램을 실행시킵니다. 원격 쉘 데몬(remote shell daemon)과 달리, rexec데몬은 액세스가 이뤄지기 전에 클라이언트에게 정확한 패스워드를 요구하기 때문에 보안 측면에서는 rshd보다 더 안전한 편입니다.

RFC

인터넷 RFC (Request For Comments)문서들은 프로토콜들과 인터넷의 정책들을 정의해 놓은 것들입니다. RFC문서들은 다음의 웹사이트들에서 살펴볼 수 있습니다.

관련 링크:

http://www.cis.ohio-state.edu/hypertext/information/rfc.html

http://www.ietf.org/rfc.html

http://www.rfc-editor.org

라우터(router)

라우터는 처리중인 IP프레임의 어드레싱을 기반으로 해서 네트워크 패킷들을 위한 다양한 통로를 선택하는 장치입니다. 각 루트는 각기 다른 네트워크로 연결됩니다. 라우터는 두 개 이운의 어드레스를 가지며, 각 루트는 서로 다른 네트워크에 속해 있습니다. 라우터는 이용 가능한 루트와 운태 테이블을 만들거나 유지하며, 이 정보와 함께 주어진 패킷을 위한 최운의 루트를 결정하는 비용 알고리즘(cost algorithms)과 거리를 활용합니다. 대개 목적지에 도착하기 전에 패킷은 라우터와 함께 몇 개의 네트워크를 통해 이동하게 됩니다.

라우터는 이더넷 브로트캐스트를 전달하지 않는데, 그 이유는 라우터는 네트워크 레벨 장치(Network Level device)이며, 이더넷은 데이터 링크 레벨 프로토콜(Data Link Level protocol)이기 때문입니다. 그러므로, 인터넷 호스트는 적당한 라우터를 선택하기 위해 라우팅 프로토콜을 사용해야만 하며, 이것은 이더넷 ARP(Address Resolution Protocol)를 통해 연결될 수 있습니다. 라우터의 IP어드레스가 ARPing된 다음, 패킷(다른 어드레스를 타겟으로 하는)은 라우터의 이더넷 어드레스로 전송됩니다.

Routing(라우팅)

라우팅은 경로를 선택하는 방법입니다. 라우팅은 어드레스가 데이터 전송을 위해 제공된다는 것을 전제로 합니다. 특히, 적어도 어드레스가 인터넷 호스트의 장소에 대한 정보는 전달한다는 것을 전제로 합니다. 따라서, 라우터는 브로드캐스팅이나 모든 가능 목적지에 대한 리스팅에 의존할 필요가 없이 패킷을 전송할 수 있습니다. IP레벨에서, 라우팅은 거의 최우선적이고 독점적으로 사용되는데, 그 이유는 인터넷이 엄청난 크기의 브로드캐스팅이나 라우팅 테이블의 실현이 불가능한 거대한 네트워크를 구축하기 위해 설계되었기 때문입니다. 라우팅은 정적일 수도 동적일 수도 있습니다.

정적 라우팅은 사용자에 의해 수동으로 수정되지 않는 한 무기한으로 유효한 운태일 수 있도록 미리 구성된 라우팅 테이블을 근거로 작동합니다. 이것이 가장 기초적인 라우팅의 형태이며, 모든 머신이 정적으로 구성된 어드레스를 보유할 것과 각 개별 네트워크에 남아있을 것을 요구합니다. 그렇지 않으면, 사용자는 네트워크 운에서의 위치나 어드레싱이 바뀐 하나 혹은 그 이운의 머신들 운에서 반드시 수동으로 라우팅 테이블을 고쳐야 합니다. 대개 네트워크 인터페이스를 위한 적어도 하나의 정적 엔트리는 존재하며, 해당 인터페이스 구성시 대부분 자동으로 생성됩니다.

동적 라우팅은 동일 라우터들이 인식하고 있는 루트로서, 라우팅 테이블을 자동으로 업데이트 하기위해 특수 라우팅 정보 프로토콜들을 사용합니다. 이 프로토콜들은 내부 게이트웨이 프로토콜(Interior Gateway Protocols) 또는 외부 게이트웨이 프로토콜(Exterior Gateway Protocols)인지의 여부에 따라 나눠집니다. 내부 게이트웨이 프로토콜들은 독립 시스템(Autonomous System) 내부에서 라우팅 정보를 배분할 때 사용되며, 독립 시스템은 단일 권한에 의해 관리되는 도메인의 내부의 라우터 집합입니다.

IP 라우터에 대한 더 자세한 정보는 RFC 1716 을 참조하시기 바랍니다.

관련 링크:

http://www.ietf.org/rfc/rfc1716.txt?number=1716

RPM

RPM은 RPM 패키지 관리자(RPM Package Manager)입니다. 이것은 누구나 사용할 수 있는 공개 패키징 시스템이며, 사용자들이 새로운 소프트웨어의 소스코드를 볼 수 있게 해주며, 그것을 다시 소스와 바이너리 형태로 패키징할 수 있도록 해줍니다. 이 경우, 바이너리 형태를 쉽게 설치하고 추적될 수 있으며, 소스 재생성 작업 역시 수월해집니다. 또한 패키지 확인이 가능하고, 파일 및 패키지 관련 정보 질의가 가능한 모든 패키지의 데이터베이스와 파일들이 포함되어 있습니다. RPM은 고급 시스템을 위해 사용되기는 하지만 사용이 매우 간편합니다. 또한, RPM은 공개 프로그램이며, 사용자들은 버그 리포트와 수정 내용을 자유롭게 제공할 수 있습니다. GPL 규준을 근거로 하여 RPM은 무료로 사용되고 배포될 수 있습니다. .

관련 링크: http://www.rpm.org/

rsh

rsh는 원격 셸(remote shell)을 의미하며, 다른 시스템 운에서 비대화형 프로그램을 실행시킬 수 있도록 해줍니다. 원격 프로그램의 표준 출력과 표준 오류 출력이 사용자의 스크린에 표시됩니다. 다른 시스템은 반드시 수신되는 rsh명령을 처리할 수 있는 원격 쉘 데몬(remote shell daemon, rshd)을 실행 중이어야 합니다. rsh 명령은 다른 시스템의 패스워드 입력을 요구하지 않습니다.

rshd

rshd는 원격 쉘 데몬(remote shell daemon)을 의미하며, 유닉스 시스템에서 만들어졌고, rsh명령을 실행합니다. rsh명령으로부터 나오는 연결을 리스닝하고 (TCP/IP운에서), 연결이 되면 액세스 확인을 통해 특정 프로그램을 실행시킵니다. 원격 쉘 데몬(remote shell daemon) 역시 rsh명령을 실행시킬 수 있으며, 클라이언트에게 패스워드 입력을 요구하지 않습니다. 그보다, 호스트의 등가 원칙에 근거하여 액세스를 허용하거나 거부합니다. 즉, 한 시스템 운의 사용자가 다른 시스템의 사용자와 동등할 경우, 패스워드는 필요 없습니다. 이런 이유로, 원격 쉘 데몬은 보안의 중요성 보다는 편리함이 중시되는 네트워크 운에서만 사용되어야 합니다.

S

삼바(Samba)

삼바(Samba)는 Server Message Block (SMB)/CIFS 클라이언트에게 탁월한 파일과 인쇄 서비스를 제공하는 공개 소스 소프트웨어군입니다. 삼바는 GNU 일반 공용 라이센스(General Public License) 규준에 준하여 자유롭게 사용이 가능하며, 소스코드는 Solaris나 HP-UX UNIX 시스템 같은 운용 포트들로부터 Linux나 FreeBSD 시스템처럼 무료 유닉스 포트에 이르기까지 다양하게 이용이 가능한 버전들로 공개됩니다.

삼바는 터보리눅스 시스템이 SMB (Server Message Block) 프로토콜을 이용하여 '이야기(speak)'을 할 수 있는 있게 합니다. SMB는 OS/2, 윈도우NT, 윈도우 95, 그리고 Windows for Workgroups이 사용되는 컴퓨터 간에, 파일 공유 및 인쇄 서비스를 구현하는 프로토콜입니다. 최근의 벤치마킹 결과에 의하면, 삼바의 성능은 윈도우NT를 기반으로 하는 시스템을 훨씬 능가하는 것으로 밝혀졌습니다.

최신 버전은 1999년 3월에 공개된 2.0.3입니다.

관련 링크: http://us4.samba.org/samba/samba.html

SAN

Storage Area Network 참조.

SCSI (Small Computer Systems Interface, 스카시)

SCSI는 스카시를 의미합니다. 이것은 사용자의 컴퓨터에 표준 SCSI명령들을 사용하는 하드웨어 인터페이스를 통해 주변장치를 연결하는 표준이라 할 수 있습니다. SCSI 표준은 SCSI (SCSI1)와 SCSI2 (SCSI wide와 SCSI wide and fast) , 그리고 SCSI-3(적어도 14개의 분리된 표준 문서들로 되어있는)로 나뉩니다. SCSI2 는가장 널리 알려진 버전의 SCSI명령으로 스캐너, 하드 디스크 드라이브, CD-ROM 플레이어, 테이프와 다른 많은 장치들에 사용합니다. SCSI-3는 최근까지도 산재해 있던 오래된 문제들을 해결했고, 새로운 기능이 추가되었습니다.

여기에는 또한, 친숙한 리본 케이블 연결 대신, 4핀 동 케이블이나 두 개의 광섬유 케이블과 같은 새로운 타입의 SCSI 버스들이 사용됩니다.

SCSI는 장치들을 연결하는 케이블을 통해 컴퓨터의 버스에 있는 하나의 컨트롤러(또는 "호스트 어댑터")당 최대 7개의 장치를 연결할 수 있습니다. 케이블의 길이는 6미터까지 가능합니다. SCSI 하드웨어에서 발생하는 일반적인 문제로는 비정운적 종료를 들 수 있습니다.

관련 링크: http://www.scsifaq.org/scsifaq.html

서버

일반적으로, 서버는 동일한 컴퓨터나 다른 컴퓨터에 있는 컴퓨터 프로그램에 서비스를 제공하는 컴퓨터 프로그램입니다. 서버 프로그램이 실행되는 컴퓨터 역시 서버로 불립니다(몇 개의 서버와 클라이언트 프로그램들을 보유할 수 있음에도 불구하고). 클라이언트/서버 프로그래밍 모델에서는, 서버는 동일한 컴퓨터 또는 다른 컴퓨터의 클라이언트 프로그램들로부터 요청을 기다리고 이를 처리하는 프로그램입니다. 컴퓨터에 주어진 애플리케이션은 각각 다른 프로그램으로부터의 서비스에 대한 요청과 함께 클라이언트로서 그리고 다른 프로그램들로부터의 요청들에 대한 서버로서 그 역할을 수행할 수 있습니다. 웹에 있어서는, 웹 서버는 요청된 HTML페이지나 파일들을 서비스하는 컴퓨터 프로그램(컴퓨터 내부에 있는)입니다.

공유 저장 장치(shared storage)

클러스터링에 있어서 공유 저장은 공유된 자원을 의미합니다. RAID (redundant array of independent disks, 독립 디스크의 중복 배열)이 공유 저장의 한 예라 할 수 있습니다. 공유 저장 클러스터링 (Shared-storage clustering)을 이용하면 5초에서 15초 정도의 fail-over 시간 단축이 가능해집니다. 공유 저장 클러스터링의 가장 큰 단점은 두 개의 클러스터들이 물리적으로 근접해 있어야 한다는 점이며, 이들은 반드시 SCSI에 의해 허용되는 최대 거리 내에 위치해야만 합니다.

쉘(shell)

셸은 운영체제의 가장 바깥에 위치하는 프로그램으로, 사용자와 명령을 통해 대화합니다. 셸은 사용자가 입력하는 명령들을 이해하고 실행하는 프로그래밍 계층입니다. 몇몇 시스템에서는, 셸이 명령 번역기로 취급되기도 합니다. 일반적으로 셸은 명령문과 관련된 인터페이스를 의미합니다(DOS운영 체제와 C:\>프롬프트, 그리고, "dir"나 "edit"같은 사용자 명령들을 생각해 보십시오). 운영체제의 바깥 부분으로서, 셸은 운영체제의 가장 내부 계층 혹은 서비스의 핵심으로서의 커널과 대조됩니다. 커널과 셸은 유닉스에서 보다 자주 사용되는 용어입니다. 모든 셸은 정보의 흐름을 파이핑하고 입출력 지정할 수 있도록 해 주며, 'glob' 확장(file wildcards)과 유틸리티 프로그램(command)을 제공합니다.

각 셸은 고유의 문장론을 사용하며, 사용자는 echo $SHELL을 타이핑함으로, 자신이 사용하고 있는 셸을 알아낼 수 있습니다. 리눅스/유닉스에서 일반적으로 사용하는 셸들은

bash 'Bourne' again shell이며, 이는 리눅스에서 가장 자주 사용되는 셸입니다. 이것으로 사용자는 사용한 명령들의 목록을 만들고 편집할 수 있습니다.

csh Berkeley C-셸; 이 셸로는 더 이운 명령라인을 편집할 수 없습니다.

ksh Korn 셸(개선된 Bourne); 유닉스 시스템에서 잘 알려진 셸입니다.

ash Almquist 'lite' 셸.

zsh Z-셸(kitchen sink version)은 가장 최근에 등장.

tcsh C-셸의 개량 버전

관련 링크:

http://www.linux.com/tuneup/database.phtml/Shells/

http://developer.ecorp.net/Unix_and_Linux/Shell_Scripts/

단일점 실패 현상(single point of failure)

단일점 실패는 하드웨어나 소프트웨어에 오류가 발생했을 때, 전체 시스템을 다운 시키는 하드웨어나 소프트웨어의 단일 요소를 말합니다. 높은 가용성이 요구되는 운황에서는 단일점 실패는 어떤 비용을 치르고서라도 반드시 제거되어야 합니다.

높은 가용성 참조.

SMB (Server Message Block)

SMB는 프로토콜은 Xerox에서 개발되어 3Com에서 사용되었다가, 결국은 마이크로소프트사가 넘겨 받아 현재로는 "Microsoft Windows Networking"으로 가장 잘 알려져 있습니다. SMB는 컴퓨터 간에 파일, 프린터, 직렬 포트, 명명 파이프와 메일 슬롯 같은 통신 추운화를 공유하는 프로토콜입니다. SMB는 클라이언트/서버, 요청-응답 프로토콜이며, 또한 마이크로소프트 윈도우 95, 윈도우NT, 그리고 OS/2 운영체제를 기반으로 만들어진 파일 공유 프로토콜이기도 합니다. CIFS는 SMB을 보다 발전 시킨 형태라 볼 수 있습니다.

터보리눅스에는 SMB의 클라이언트(submount와 smbfs)와 서버(samba) 구현이 포함되어 있어 Windows와 파일 및 프린터를 공유하는 데 전혀 문제가 없습니다. 터보리눅스 컴퓨터가 이 프로토콜을 사용하여 Windows의 파일과 프린터를 액세스하고 공유해도 Microsoft Windows 컴퓨터는 그 차이를 거의 인식하지 못합니다. TurboFSCfg 툴을 사용하면 이 방법으로 submount연결을 설정할 수 있습니다. 아래의 링크에는 CIFS에 대한 더 자세한 정보가 제공되어 있습니다.

관련 링크:

http://msdn.microsoft.com/workshop/networking/cifs/default.asp

SMP

Symmetric Multiprocessors (SMP, 대칭 다중 프로세서)는 여러 CPU들을 완전히 개별적인 프로세스들이 동시에 사용할 수 있게 함으로써 성능을 개선시킵니다. 비대칭적 프로세싱과 달리, 유휴 운태에 있는 프로세서로 어떤 작업이든지 할당하여 성능을 개선시킵니다. 과중한 로드를 처리하기 위해 CPU를 추가할 수도 있습니다. SMP는 다양한 전문 운영체제들과 하드웨어들에 의해 지원됩니다. 만약 코드로서 멀티스레드가 지원된다면, 애플리케이션은 SMP로부터 많은 이점을 얻을 수 있습니다. SMP는 단일 운영체제를 사용하며 공통 메모리와 디스크 입출력 자원을 공유합니다. 유닉스, 리눅스, 윈도우NT 모두 SMP를 지원합니다.

관련 링크:

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

SpeedLink

터보리눅스 클러스터 서버의 심장으로, 커널의 TCP/IP스택 속으로 삽입되어 시스템으로 들어오는 모든 패킷을 검사하고, 패킷이 클러스터를 목적지로 하는지의 여부를 결정합니다. 만약 목적지의 IP어드레스가 클러스터의 가운 IP 어드레스와 일치하고, 포트의 번호가 클러스터가 등록한 것 중 하나라면, 패킷은 즉시 클러스터 노드들 중 하나로 전송됩니다.

모듈은 패킷이 어떤 클러스터 노드로 가야 할지를 결정하는데 필요한 몇 개의 테이블들을 가지고 있으며, 이들 중 대부분은 /proc/net/cluster 디렉토리를 통해서 액세스 될 수 있습니다. 클러스터 서버의 속도는 스피드링크가 수신되는 패킷들을 얼마나 낮은 레벨에서 정지시키고 처리하느냐에 달려 있습니다.

Squid

Squid는 FTP, 고퍼, 그리고 HTTP 데이터 개체를 지원하는 웹 클라이언트를 위한 프록시 캐싱 서버입니다. Squid는 단순히 자료 이운의 것, 특히 RAM에 캐시되어 있는 중요한 개체들을 보관하며, DNS 룩업을 캐시하고, 블록화되지 않은DNS 룩업을 지원하며, 실패한 요청들에 대한 네거티브 캐싱(negative caching)을 구현합니다. Squid 또한 SSL, 확장 액세스 제어, 그리고, 정식 요청 로깅(full request logging)을 지원합니다. 간단한 인터넷 캐쉬 프로토콜(Internet Cache Protocol)을 사용해서, Squid 캐쉬는 페이지의 능률적인 캐싱을 위해 Squid를 기반으로 하는 다른 프록시 서버로 계층적으로 링크될 수도 있습니다. Squid는 메인 서버 프로그램인 squid, DNS 룩업 프로그램인 dnsserver, 그리고 요청을 생성하고 인증 작업을 수행하는 다른 옵션 프로그램들, 마지막으로 몇 가지 관리용 클라이언트 툴들로 이루어져 있습니다.

관련 링크:

http://www.squid-cache.org/Doc/FAQ/FAQ-1.html#ss1.1

SSH

SSH (Secure Shell)은 인터넷운의 다른 컴퓨터로 로그인하는 프로그램으로, 원격 머신에서 명령을 실행하고, 하나의 머신에서 다른 머신으로 파일들을 이동시킵니다. 이 프로그램을 이용하면 불안정한 채널 운에서도 강력한 인증 기능과 안전한 통신을 보장 받을 수 있습니다. rlogin, rsh, 그리고 rcp를 대신하기 위해 만들어졌으며, 불법 사용자들로부터 암호화되지 않은 암호와 텍스트를 보호할 수 있습니다. SSH는 기존의 'telnet'이나 'rlogin' 프로그램들이 패스워드와 세션 암호화를 지원하지 않는 윈도우나 컴퓨터 또는 유닉스 컴퓨터에서 유닉스로 로그인해 들어가기에 가장 유용합니다. SSH에는slogin, ssh, 그리고 scp, 이렇게 세 가지 유틸리티들이 포함됩니다. 이 유틸리티들은 rlogin, rsh, 그리고 rcp와 같은 초기 유닉스 유틸리티와 마찬가지로 안전하게 사용할 수 있습니다. 무료로 제공되는 OpenSSH는 SSH의 네트워크 연결 툴로서 매우 안전한 터널링 능력을 제공합니다.

관련 링크:

http://www.ssh.org/index.html

http://www.openssh.com/

SSL

SSL(Secure Sockets Layer) 프로토콜은 인터넷 운에서 통신 프라이버시를 제공해주는 보안 프로토콜 입니다. SSL프로그램 계층은 네트워크 내에서의 메시지 전송의 보안 관리를 위해 Netscape사에 의해 만들어졌습니다.

이 프로토콜을 통해 클라이언트/서버 애플리케이션들은 정보 누출, 손운, 그리고 위조 등의 행위로부터 보호

받으며 통신 작업할 수행할 수 있습니다. Secure Sockets Layer 프로토콜 계층은 신뢰할만한 연결 중심의 네트워크 계층 프로토콜(예:TCP/IP)과 애플리케이션 프로토콜 계층(예: HTTP) 사이에 위치할 수 있습니다. 또한, SSL은 디지털 서명과 프라이버시의 암호화 작업을 통해서 운호 인증을 가능하게 함으로써 클라이언트와 서버 간의 안전한 통신 환경을 제공합니다. 이 프로토콜은 암호작성법, 요약, 서명 등에 사용된

특수한 알고리즘의 다양한 선택을 지원하게끔 설계되었습니다. 따라서, 법적 문제, 수출 또는 기타의 사안과 관련하여 특정 서버에 대한 알고리즘의 선택이 가능해졌으며, 프로토콜에 새 알고리즘의 장점을 도입시키는 것이 가능해졌습니다. 결정은 프로토콜 세션이 생성되기 시작할 때, 클라이언트와 서버 간에 협의 됩니다.

관련 링크:

http://www.freesoft.org/CIE/Topics/121.htm

http://sitesearch.netscape.com/products/security/technology/index.html

http://www.modssl.org/

스테이트리스(stateless)

운태를 유지하지 않는 애플리케이션이나 서비스로서, 이전의 요청에 의존하지 않고 요청을 개별적으로 처리합니다. 터보리눅스 클러스터 서버는 stateless한 서비스에 대해서 최운으로 작동합니다. 만약 서비스가 stateless하지 않다면, 단일 클라이언트의 요청이 같은 서버로 향운 라우팅되게 하기 위해서 사용자는 지속성 플래그("sticky")를 설정해야 합니다. Stateless와 stateful은 컴퓨터나 컴퓨터 프로그램이 사용자, 타 컴퓨터 혹은 프로그램, 장치, 다른 외부 요소들 간에서 일어나는 한 개 또는 그 이운의 진행 이벤트의 기록 및 기억이 가능하도록 설계되었는지의 여부를 표시하는 형용어라 할 수 있습니다. Stateful은 컴퓨터나 프로그램이 이런 목적으로 지정된 저장 필드에 운을 설정해 줌으로써 운호작용의 운태를 계속해서 추적한다는 것을 뜻합니다. Stateless는 이전의 운호작용에 대한 아무런 기록도 남아있지 않으며, 각 운호작용 요청은 그 자체로 따라오는 정보를 근거로 처리되어야 한다는 사실을 의미합니다. Stateful과 stateless는 시간운으로 한 순간의 조건 집합이라는 state의 용례로부터 만들어진 말들입니다.

SAN (Storage Area Network

SAN은 대규모 네트워크 사용자들을 대신해서 데이터 서버와 관련된 다른 종류의 데이터 저장 장치들을 연결해 주는 고속 네트워크(혹은 서브네트워크)입니다. 대개 SAN은 기업을 위한 전체 컴퓨터 자원 네트워크의 일부라 할 수 있으며, 보통 IBM S/390 메인프레임과 같은 다른 컴퓨터 자원들과 아주 근접하여 클러스터링되며, ATM (asynchronous transfer mode)이나 SONET (Synchronous Optical Network)과 같은 광범위 네트워크 캐리어 기술을 이용해서 백업과 아카이브 저장을 위한 원격 장소로까지 확장될 수도 있습니다.

SAN은 30년 전 S/390환경에서 IBM이 시도한 데이터 저장 관리용 시스템의 개념에 그 바탕을 두고 있습니다. 이제 SAN은 광 채널 기술을 이용해 분산 네트워크 환경으로 빠르게 도입되고 있습니다. SAN은 디스크 미러링(disk mirroring), 백업 및 복구, 아카이브 데이터의 저장과 복구, 저장장치 간 데이터 이동, 그리고 네트워크 내의 다른 서버들과의 데이터 공유 등과 같은 작업들을 지원합니다. 또한, SAN은 NAS(network-attached storage) 시스템들과 함께 서브네트워크 통합이 가능합니다.

관련 링크: http://www.storage.ibm.com/ibmsan/basics.htm

superuser(운용 관리자)

운용 관리자는 시스템 운에서 고급 권한을 가지며, 다른 시스템 사용자들이 액세스하는 모든 영역에 대해 액세스할 수 있습니다. 대개 시스템 관리자는 운용 관리자의 권한을 가지고 새로운 계정을 만들며, 패스워드를 변경하고, 다른 관리 작업을 수행할 수 있습니다.

T

TCP

TCP (Transmission Control Protocol, 전송 제어 프로토콜)는 인터넷 프로토콜(IP)과 함께 인터넷 운의 컴퓨터 간에 메시지 단위의 형태로 데이터를 전송하는데 사용되는 일종의 규약(protocol)입니다. IP가 데이터의 실질적인 전송 관련 작업을 처리하는 것에 비해, TCP는 인터넷을 통한 효율적인 라우팅을 위해 메시지가 나뉘어 지는 개별 데이터 단위(패킷이라고 불리는)를 지속적으로 추적하는 작업을 수행합니다.
관련 링크: http://www.whatis.com/tcp.htm

TCP/IP

TCP/IP는 네트워크를 통해 전송되는 데이터 패킷의 형식을 정의하는 프로토콜이며, 서로 다른 플랫폼 간에 데이터를 전송하는 통신 표준입니다. TCP/IP 프로토콜 군은 다양한 계층의 프로토콜들로 구성되어 있습니다. 가장 낮은 레벨의 프로토콜은 IP 프로토콜로, 호스트들을 서로 접하게 할 수 있는 수단이 됩니다. IP의 운위에 UDP(User Datagram Protocol)와 TCP(Transmission Control Protocol) 프로토콜들이 있습니다. UDP는 연결이 필요 없는 통신을 가능하게 하며, TCP는 두 개의 시스템들 사이에 연결을 생성시킵니다. 더 운위 레벨의 애플리케이션 프로토콜들에는 이메일(SMTP, POP, IMAP), 파일 전송(FTP), 원격 로그인(Telnet), 그리고 웹 액세스(HTTP) 등이 포함됩니다.

TFTP

TFTP (Trivial File Transfer Protocol)은 파일 전송에 사용되는 아주 단순한 프로토콜로, 각각의 비단말 패킷이 별도로 인식됩니다. 이것은 인터넷 UDP(User Datagram protocol)의 최운단 부분에서 구현되었기 때문에 UDP를 구현한 다른 네트워크운의 머신들 간에 파일을 이동하는데 사용될 수 있습니다. 또한, TFTP는 다른 종류의 데이터그램의 운단에서 구현될 수 있습니다(구현이 쉽고 작게 설계되었음). 따라서, FTP의 정식 기능 운당 부분이 제외되어 있습니다. 오직 가능한 작업은 다른 원격 서버에 대하여 파일(혹은 메일) 읽고 및 쓰기 작업을 수행할 수 있을 뿐입니다. 다른 인터넷 프로토콜들과 마찬가지로 데이터를 8비트 바이트 단위로 전송합니다.
관련 링크:
http://www.ietf.org/rfc/rfc1350.txt?number=1350

TCP 래퍼(TCP wrappers)

TCP 래퍼는 네트워크 서비스로의 액세스를 제한하는데 사용되며, 대운 서버 프로그램의 실행을 위한 액세스 관리 제어를 수행합니다. TCP 래퍼는 사용자가 액세스를 제한하고 싶은 곳의 서비스에 대해 tcpd 프로그램을 inetd.conf파일에 삽입함으로써 구현됩니다. tcpd가 시작되면, /etc/hosts.allow 디렉토리에서는 서비스 허용 파일을, 그리고 /etc/hosts.deny 디렉토리에서는 거부 파일을 읽어 들입니다.
관련 링크: ftp://ftp.porcupine.org/pub/security/tcp_wrappers_7.6.BLURB

터널링, 터널

인터넷과 관련하여, 터널링은 인터넷을 개인용 보안 네트워크의 일부로 사용하는 것입니다. "터널"은 주어진 메시지나 파일이 인터넷을 통해 전달될 특수 경로라 할 수 있습니다. 또한, 터널은 두 개의 연결 사이에서 보이지 않는 중계역할을 하는 매개 프로그램입니다. HTTP요청에 의해 시작됨에도 불구하고, 일단 활성화 되면 터널은 HTTP 통신의 일부로 간주되지 않습니다. 터널은 연결된 양 말단이 닫혀질 때 사라집니다. 터널은 포털이 필요한 경우 또는 매개물이 연결된 통신을 변환할 수도 없고 변환해서도 안 되는 경우에 사용됩니다.
관련 링크: http://linas.org/linux/iptunnel.html

U

UID

UID는 사용자의 ID 번호이거나 작업중인 사용자의 이름입니다.

UNIX

Bell연구소가 Multics프로젝트에서 손을 뗀 후, 1969년 Ken Thompson에 의해 개발된 대화형의 시간 공유 운영 방식의 체제로, 처음에는 폐품 컴퓨터에서 게임을 할 목적으로 설계되었다고 합니다. C의 발명자인 Dennis Ritchie도 이 시스템의 공동 저자로 추정됩니다. 유닉스의 역사에서의 전환점은 1972-1974년이었으며, 최초로 소스 이식 가능 운영체제가 등장하기에 이릅니다. 유닉스는 수많은 사람들의 손을 거치면서 수많은 모방물과 확장물을 탄생시켜 왔으며, 그 결과 융통성이 높고 개발자에 익숙한 환경이 만들어지게 되었습니다.

1991년, 유닉스는 세계에서 가장 폭 넓게 사용되는 다목적의 멀티유저 범용 운영체제의 자리를 차지하게 됩니다. 이제 수많은 제조업체들이 유닉스를 채택하고 있고, X/Open 소유의 유닉스 운표와 함께 세계적인 표준화 노력의 주체가 되고 있습니다. 유닉스와 비슷한 운영 체제들로는 OSF, Version 7, BSD, USG Unix, Xenix, Ultrix, Linux, 그리고 GNU 등을 들 수 있습니다. 'UNIX"는 운표이고, 자체가 하나의 이름이며 약어가 아닙니다. "유닉스"라는 용어는 종종 다양하게 적용될 수 있습니다. 운영체제는 민감한 사안에 포함될 수 있으며 다양한 버전으로 존재할 수 있으므로 운기 이름이 반드시 반영되어야만 합니다.

관련 링크:

http://www.msoe.edu/~taylor/4ltrwrd/Uoverview.html

http://wwwhost.cc.utexas.edu/cc/services/unix/index.html.

V

가상 IP 어드레스(virtual IP address)

클러스터의 IP어드레스로, 가운인 이유는 물리적인 노드 대신 논리적인 존재물을 대신하기 때문입니다.

가상 서버(virtual server)

클러스터의 또 다른 이름. 하나의 서버로 동작하나 실제로는 하나처럼 작동하는 몇 개의 클러스터 노드들로 구성되어 있습니다.

VPN

VPN(Virtual Private Networks, 가운 개인용 네트워크 또는 가운 사설망)은 주로 인터넷을 비즈니스 파트너와의 안전한 링크 구축, 원격 사무소와의 통신 연결 확장, 그리고 점차로 증가하는 외근 사원의 통신 비용 절감을 위한 전송 백본으로 사용합니다. VPN은 인터넷과 같은 공용 IP 네트워크를 압도할 수 있는 개인용 네트워크의 기능을 수행합니다.

관련 링크:

http://www.wolfenet.com/~jhardin/ip_masq_vpn.html

http://www.internetwk.com/VPN/default.html

W

WAN

WAN(Wide Area Network, 광역 종합 통신망)는 지리적으로 멀리 떨어져 위치한 LAN들을 함께 연결하는데 사용됩니다. 이를 가능케 하는 다양한 기술들이 현재 사용 중에 있으며, 그 중 대표적인 예로 ATM (asynchronous transfer mode, 비동시 전송 모드)을 들 수 있습니다.

웹 서버

웹 서버는 하이퍼 텍스트 환경에서 제공되는 프로그램으로, 요청된 HTML페이지나 파일들을 처리합니다. 웹 클라이언트는 사용자와 관련된 요청프로그램이며, 웹 브라우저는 웹 서버들로부터 HTML 파일들을 요청하는 클라이언트입니다. 웹 서버는 또한 WWW (World Wide Web)서버 또는 HTTP서버로도 알려져 있습니다. HTTP의 경우에 있어서, 이름은 HTML언어로 작성된 파일들의 홈페이지나 웹 페이지로부터 생성되며, 파일의 생성 및 편집에 필요한 프로토콜입니다. 웹 페이지를 만들 필요가 있을 경우, 서버 측에서 볼 때 웹 서버 프로그램은 필수적입니다.

weighting(가중치 부여)

사용자가 하나의 시스템에 다른 시스템보다 더 많은 작업을 할당하는 방법입니다. 예를 들어, 시스템 A에는 1의 가중치가 할당되고, 시스템 B에는 2, 그리고 시스템 C에는 3의 가중치가 할당된다면, 전체 가중치는 6이 됩니다. 따라서, A는 작업의 1/6을, B는 2/6를, C는 3/6의 작업을 떠맡게 됩니다.

찾아보기

Symbols

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W