

Cold-standby 방식의 고가용 웹 서버 시스템 구현*

김용희⁰, 성영락[†], 이철훈[†]
충남대학교 컴퓨터공학과, [†] 국민대학교 전자정보통신공학부
{yonghee, chlee}@ce.cnu.ac.kr; [†] yeong@mail.kookmin.ac.kr

Implementation of High Available Web-Servers using Cold-standby

Yong-Hee Kim⁰, Yeong Rak Seong[†], and Cheol-Hoon Lee
Dept. of Computer Engineering, Chungnam National Univ.
[†] School of Electrical Engineering, Kookmin Univ.

요 약

본 논문은 서버를 구성할 수 있는 방식 중의 하나인 Cold-standby 방식을 이용하여 리눅스를 기반으로 한 고가용의 웹 서버 설계 및 구현에 대해 그 방안을 제시하고자 한다. Cold-standby 방식은 프라이머리 서버와 필요시 그 기능을 대신하는 백업 서버로 구성되며, 서비스를 제공하는 프라이머리 서버에 결함이 발생하여 서비스를 제공할 수 없을 경우 백업 서버가 그 역할을 대신하는 동작 방식이다. 프라이머리 서버에서 백업 서버로의 스위칭은 리눅스에서 오픈 소스로 제공하는 heartbeat 기법을 이용하여 프라이머리 서버의 결함 발생시 백업 서버로의 스위칭이 가능하며, 프라이머리 서버의 과중한 트래픽으로 인해 발생할 수 있는 오버헤드를 줄이기 위해 실제 서비스를 처리하는 리얼 서버들을 클러스터링하였다. 또한 mon 기법을 이용하여 리얼 서버들의 상태를 모니터링함으로써 리얼 서버들의 추가 및 삭제가 용이하며, 웹 서버를 위한 하드웨어 및 소프트웨어 기법의 고가용을 제공함으로써 클라이언트에 대하여 안정성 및 신뢰성을 보장한 고가용의 서비스를 제공하는 웹 서버를 구현하였다.

1. 서 론

인터넷 사용의 일반화 및 생활화로 인해 그 응용범위 및 기능은 매우 확대 및 고급화되고 있으며, 인간의 활동범위와도 밀접한 관계를 가지게 되었다. 특히, IT 산업에 있어서 60%이상의 서버 시스템들이 Mission critical 또는 Business critical 한 작업에 투입되고 있으며, 네트워크망을 통한 제어 시스템의 활용도 또한 날로 증대되고 있는 추세이다. 이러한 시스템에서는 웹 서버의 안정성 및 가용성은 물론 서버의 고비용 및 운용(관리) 등의 문제가 큰 이슈(issue)로 대두되고 있으며, 최근 과중한 트래픽으로 인한 웹 서버의 다운으로 시간적/경제적인 막대한 손실을 가져온 사건은 이러한 문제점들을 더욱더 확연히 증명하는 계기가 되었다. 또한 서버에 결함이 발생하였을 경우 그에 따른 신속한 조치에 대한 클라이언트들의 기대 역시 커졌으며, 빠른 시간 내에 시스템의 결함을 찾아내고 필요한 조치를 신속하게 처리하기 위한 기능이 강력히 요구되어지고 있다.

이에 따라 본 논문에서는 리눅스에서 오픈소스로 제공하는 heartbeat 과 mon 기법을 이용하여 Cold-standby 방식의 고가용성 및 신뢰성을 보장하는 웹 서버를 구현하고자 한다. 또한 실제 서비스를 처리하는 리얼 서버들을 클러스터링하여 로드밸런싱함으로써 서버에 대한 과중한 트래픽으로 인해 발생할 수 있는 오버헤드를 줄일 수 있으며, 외부의 클라이언트에게는 가상 IP를 통해 오직 하나의 IP만을 보여줌으로써 안정성 및 고신뢰성을 지향하는 고가용의 서비스를 제공하는 웹 서버를 구현하고자 하는데 그 목적이 있다.

*본 연구는 산업자원부 중기거점과제 연구비에서 지원을 받았음.

본 논문은, 2 장에서는 고가용의 웹 서버를 구현하기 위하여 하드웨어적인 Cold-standby 및 Hot-standby 방식에 대해 기술하며, 소프트웨어적으로는 리눅스의 heartbeat 과 mon 기법에 대한 관련 연구를 기술한다. 3 장에서는 실제 구현한 Cold-standby 방식의 웹 서버 시스템의 구성도 및 동작방식에 대해 설명하고, 4 장에서는 실험결과 및 문제점들에 대해 기술하며, 마지막으로 5 장에서는 결론 및 향후 연구과제를 기술한다.

2. 관련 연구

2.1 Hot-standby 및 Cold-standby 방식

이중의 서버 즉, 프라이머리 서버와 백업 서버를 갖춘 서버 시스템이라면 어느 한 서버에 결함이 발생하더라도 중단하지 않고 대기시켜 놓은 다른 서버로 바로 스위칭이 가능하다.

Hot-standby 방식은 프라이머리 서버와 백업 서버가 존재하며, 동시에 두 서버가 모두 서비스를 처리하는 동작 방식이다. 프라이머리 서버가 실제 서비스를 처리하며, 백업 서버는 리얼서버로서 동작을 하거나 다른 애플리케이션을 수행하게 된다. 이때 프라이머리 서버에 결함이 발생하게 되면 즉시 백업 서버로 스위칭되어 서비스를 계속할 수 있게 된다. 그래서 각 서버의 구성이 매우 중요한데, 프라이머리 서버에 결함이 발생하여 백업 서버로 스위칭하였을 때, 백업 서버가 이미 50%이상의 자원을 사용하고 있어 오버헤드로 인해 결국 백업 서버마저 다운되는 최악의 상태가 된다. 그러므로 각 서버의 자원 사용율은 50%미만으로 구성하거나 프라이머리 서버보다 더 우수한 성능의 시스템으로 구성하

여야 한다[1].

Cold-standby 방식은 프라이머리 서버와 백업 서버가 존재하며, 프라이머리 서버가 서비스를 제공하다가 결함이 발생하면 비로소 대기하고 있던 백업 서버가 스위칭되어 프라이머리 서버의 기능을 대신하게 되는 동작 방식이다.

[표 1] Hot-standby 과 Cold-standby 방식의 특징

	Hot-standby 방식	Cold-standby 방식
장점	- 시스템 사용의 효율성 - 스위칭 시간이 짧음	- 비용이 적게 들 - 오버헤드로 인한 고장발생률 적음 - 구현 용이
단점	- 비용이 많이 들 - 구현 복잡 - 오버헤드로 인한 고장발생률 높음	- 스위칭 시간이 오래 걸림 - 성능대비 가격 높음

[표 1]과 같이 Hot-standby 방식이 스위칭 시간이 짧은 것과 시스템 사용이 효율적이라는 장점이 있지만, 오버헤드로 인한 고장발생률이 높기 때문에, 이보다는 클라이언트에게 안정적이고 신뢰성있는 서비스를 제공하는 것이 웹 서버로서의 그 기능을 충분히 제공하는 것이라고 볼 수 있다[1].

2.2 Heartbeat 및 mon 기법

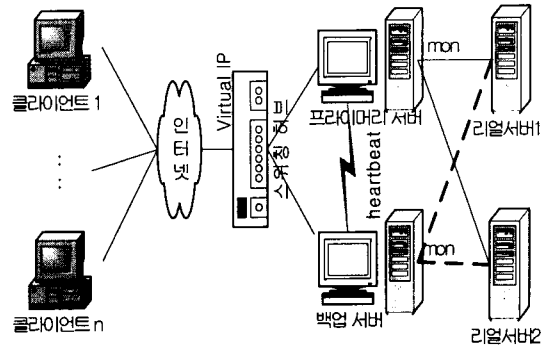
Heartbeat 과 mon 은 리눅스에서 오픈소스로 제공하는 고가용을 위한 소프트웨어 기법으로, heartbeat은 이더넷 카드(Ethernet card) 또는 시리얼 케이블(Serial cable)로 연결된 프라이머리 서버와 백업 서버가 heartbeat 이라는 프로토콜을 사용하여 주기적으로 메시지를 주고 받으며 시스템의 생존의 유무를 알린다. 일정 시간동안 heartbeat 메시지를 받지 못하면 프라이머리 서버에 결함이 발생했다고 판단하고 백업 서버는 가상 IP 를 포함한 자원(resource)을 인계(takeover)받게 된다. 프라이머리 서버에서 백업 서버로의 스위칭은 heartbeat 에 포함되어 있는 fake 기능으로, ARP(Address Resolution Protocol) spoofing 기법을 사용하여 프라이머리 서버에 aliasing 되어 있던 가상 IP 를 백업 서버에 aliasing 하는 기능을 한다[2][3][4].

mon 은 서버에서 제공하는 네트워크 서비스의 가용성과 리얼 서버들의 발생가능한 문제점을 모니터링하는데 사용하는 범용 자원 모니터링 시스템이다. mon 은 stand-alone 프로그램으로서 새로운 서비스의 모니터링이 필요하거나 새로운 기능이 요구되었을 때 추가 및 삭제가 용이하기 때문에 쉽게 확장할 수 있다[5][6].

3. Cold-standby 방식의 웹 서버 구현

3.1 서버 시스템의 구성

[그림 1]과 같이 서버 시스템을 구성하고 있는 프라이머리 서버와 백업 서버에는 각각 소프트웨어 기법인 heartbeat 과 mon 을 포팅하였으며, 프라이머리 서버와 백업 서버는 Cold-standby 방식으로 설정되어 있다. 외부의 클라이언트에게는 가상 IP 만이 존재하며, 실제 클라이언트에 대한 서비스를 처리하는 리얼 서버 2 대를 클러스터링하여 처리하도록 하였다.



[그림 1] 웹 서버 시스템의 구성도

가. heartbeat 설정

프라이머리 서버와 백업 서버간에 생존여부를 알리는 heartbeat 을 ha.cf 에 설정하며, 그 기능은 아래와 같다.

```
debugfile /var/log/ha-debug
logfile /var/log/ha-log
keepalive 2
deadtime 5
node seven.ce.cnu.ac.kr
node mazelut.ce.cnu.ac.kr
```

- keepalive : 각 heartbeat 신호사이의 경과시간
- deadtime : 프라이머리 서버의 결함판단을 위한 시간
- node : Heartbeat 으로 연결된 두 서버

위에서 언급한 heartbeat 의 주기기능을 설정한 파일의 내용에 의거하여 서버의 위치를 인식하게 되며, 프라이머리/백업 서버는 각각 시스템 자신의 상태를 서로에게 알리게 된다.

나. mon 설정

```
#!/etc/mon/mon.cf
# group definitions (hostname or IP addresses)
hostgroup cluster1 192.168.2.3
hostgroup cluster2 192.168.2.4
# Real server monitoring
watch cluster1
service http
monitor http.monitor
period wd {Sun-Sat}
alert lvs.alert -P tcp -V 168.188.46.175:80 -R 192.168.2.3 -W 1 -F m -X down
upalert lvs.alert -P tcp -V 168.188.46.175:80 -R 192.168.2.3 -W 1 -F m
watch cluster2
service http
Interval 5s
monitor http.monitor
period wd {Sun-Sat}
alert lvs.alert -P tcp -V 168.188.46.175:80 -R 192.168.2.4 -W 1 -F m -X down
upalert lvs.alert -P tcp -V 168.188.46.175:80 -R 192.168.2.4 -W 1 -F m
```

[그림 2] 서버들의 가용성 모니터링 설정

mon 은 프라이머리 서버와 백업 서버에 각각 포팅하였으며, 서버 및 리얼 서버들의 웹 서비스(http) 가용성을 모니터링하도록 mon.cf 구성파일에 설정하였다.

다. 로드밸런싱 설정

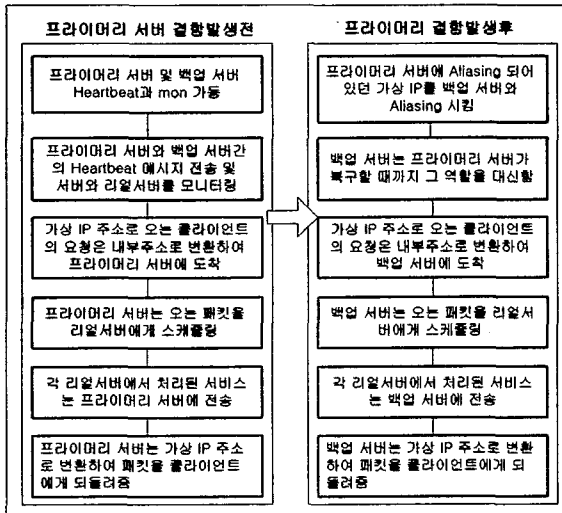
클라이언트로부터 오는 요청(requests)을 실제로 서비스하는 리얼서버들에 대한 로드밸런싱 프로그램으로, 가상 IP 주소로 오는 요청을 NAT(Network Address Translation) 방식에 의해 가상 주소를 내부 주소로 변환하여 라운드-로빈 스케줄링에 의해 차례대로 리얼서버에게 라우팅하도록 설정한다.

```
# Load Balance daemon Scripts

$IIPVSADM -A -t 168.188.46.175:80 -s rr
$IIPVSADM -a -t 168.188.46.175:80 -R 192.168.2.3 -m
$IIPVSADM -a -t 168.188.46.175:80 -R 192.168.2.4 -m
echo -n " Started daemon loadbalance: "
daemon $MON -f -c /etc/mon/mon.cf
RETVAL = $?
echo
$RETVAL = 0 ] && touch /var/lock/subsys/loadbal
echo
```

[그림 3] NAT 방식의 라운드-로빈 로드밸런싱

3.2 서버 시스템의 동작방식



[그림 4] 프라이머리 서버 결합 발생 전/후

4. 실험 환경 및 결과

4.1 실험환경

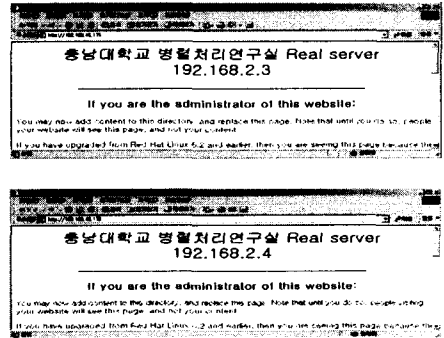
[표 2] 실험을 위한 고가용 웹 서버 시스템 관련 환경

	프라이머리/백업 서버	리얼 서버 2대
CPU	Pentium 400MHz	Pentium 200 MHz
RAM	128MB	64MB
HDD	4GB/4GB	4GB
LAN card	100/10Mbps Ethernet	100/10Mbps Ethernet
OS	Redhat 7.2 (2.4.17)	Redhat 7.2 (2.4.17)

본 논문에서 연구를 위한 실험환경은 [표 2]에서와 같이 구성하였으며, 클라이언트 2 대와 아파치(1.x)로 구성된 Cold-standby 방식을 적용한 고가용 웹 서버이다.

4.2 실험결과

본 논문에서의 실험결과로는 프라이머리 서버와 백업 서버를 서로 연결하고 있는 heartbeat의 기능을 확인하였으며, [그림 5]에서와 같이 프라이머리 서버가 클라이언트로부터의 요청을 리얼 서버에게 라우팅하여 스케줄링하는 것을 확인할 수 있었다.



[그림 5] 리얼서버들의 로드밸런싱 수행

그러나 프라이머리 서버의 결합 발생 후, 프라이머리 서버에서 백업서버로 스위칭하여 로드밸런싱하는 결과는 아직 도출하지는 못하였다. 충분한 실험을 한 후 결과를 도출할 수 있을 것이라 사료된다.

5. 결론 및 향후 연구과제

본 논문에서는 이미 언급된 바와 같이 하드웨어적 기법인 Cold-standby 방식과 리눅스의 고가용 소프트웨어 기법인 heartbeat 과 mon 을 적용하여, 가상 IP 를 두고 NAT 방식 을 사용하여 고가용 웹 서버를 구현하였다. Cold-standby 방식의 저비용 및 안정성을 고려한 구성으로, 프라이머리 서버의 결합 발생시 백업 서버로 스위칭함으로써 클라이언트에 대한 고신뢰 및 고가용의 서비스를 제공할 수 있다.

향후 연구과제로는 프라이머리 서버에서 백업 서버로의 스위칭과 함께 백업서버의 로드밸런싱에 관한 실험결과를 도출하는 것과 차후 개선점에 대해 연구할 계획이다.

참고문헌

- [1] D.K. Pradhan and N.H. Vaidya, *Fault Tolerant Computer System Design*, pp.160-192, 1996.
- [2] Linux Kernel 2.4 네트워크 트래픽 분산, " <http://www.kldp.org> " .
- [3] 리눅스 2.4 NAT HowTo : NAT 조절하기, " <http://www.kldp.org> " .
- [4] 리눅스 2.4 패킷 필터링 하우투, " <http://www.kldp.org> " .
- [5] High-Availability Linux Project, " <http://www.linux-ha.org> " .
- [6] LinuxOne High Availability Web Server, " <http://alpha-cluster.linuxone.co.kr> " .