서버호스팅 사용자 메뉴얼

< Version 2.0 >

Hostway System Team

<설치 프로그램 테스트 환경>

Redhat 7.3

Linux-kernel 2.4.20

SDRam 512M

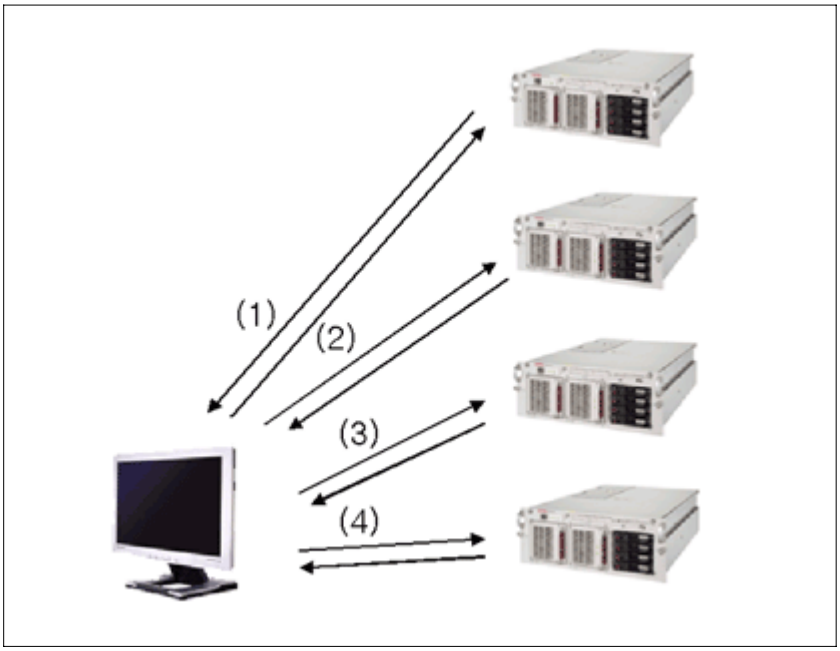
CPU cell 1.3

목 차

1. DNS 개요

인터넷을 사용하다 보면 도메인, URL이라는 말을 많이 듣는다.
도메인, URL이란 것은 숫자로 구성되어 서버마다 부여되어있는 IP address를 사람들이 외우기 쉽고 알기쉬운 문자로 표현한 것이다.
(ex: 211.239.151.21 -> manpage.co.kr)

그렇다면 DNS란 무엇인가?
바로 IP를 도메인(URL)으로 매핑시켜주는 역할을 하는 것이 바로 DNS(Domain Name Server)이다.
이 DNS가 없다면 우리들은 숫자로 (211.239.151.21) 인터넷의 서버에 접속을 해야 할것이다.
그러면 여기서 우리가 manpage.co.kr이란 URL을 웹 브라우저에서 입력을 했을때 과연 어떻게 그 manpage.co.kr의 서버로 접속을 할수 있는지 간단히 알아보자.



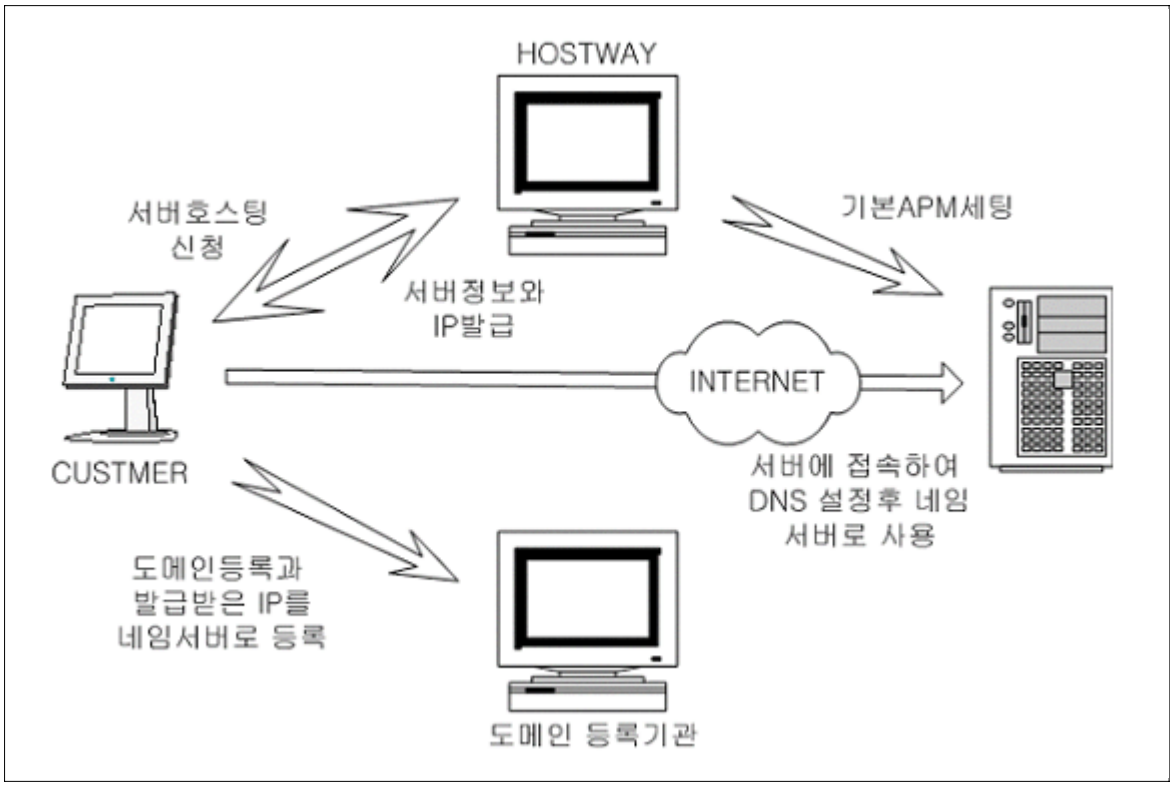
<그림 1-1 도메인으로 웹서버 접속하기>

그림을 설명하면
1) www.manpage.co.kr 를 ROOT 서버에 질의
1-1) ROOT 서버로부터 kr서버의 IP와 kr서버로 질의하라고 답변옴
2) kr서버에 www.manpage.co.kr 질의
2-1) kr서버로부터 co서버의 주소를 얻음
3) co서버에 www.manpage.co.kr 질의
3-1) ns.manpage의 서버 주소를 얻음
4) ns.manpage서버로 www의 주소를 질의
4-1) ns.manpage로부터 www의 IP주소를 얻음

이렇게 함으로써 비로소 서버가 해석할수 있는 IP주소를 얻게되고, 웹브라우저를 통해서 원하는 웹페이지로의 접속이 가능하게 되는것이다.

다음은 DNS(Domain Name Server)를 직접 구축하는 경우에만 살펴보면 될것이다. DNS를 직접 운영하지 않는 사람은 참고만 하기 바란다.

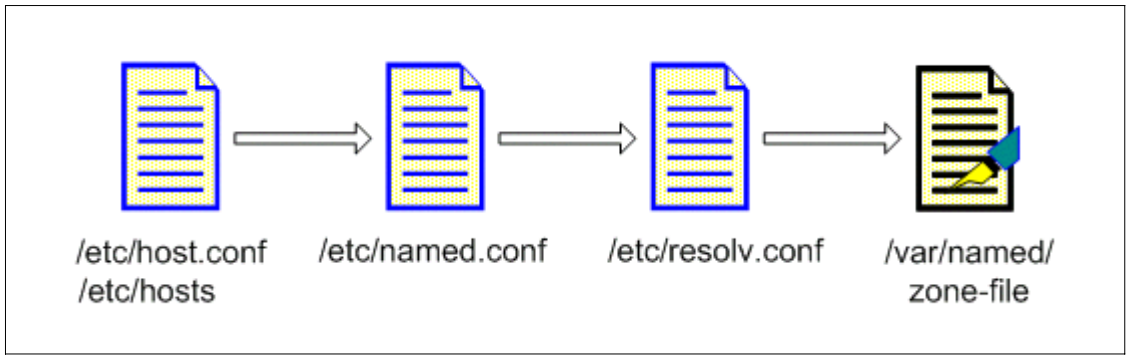
1.1 네임서버 운영 절차



< 그림 1-2 네임서버 운영 절차 >

<그림 1-2>는 호스트웨이에서 서버호스팅을 신청하여 IP를 발급받은후 그 IP를 도메인을 등록한 기관에 네임서버 등록을 의뢰하고, 네임서버로 등록이 되면, 서버에 bind를 세팅하여, 네임서버로 운영하는 절차를 나타낸 그림이다.
(네임서버등록을 예전에는 .com의 경우 외국 기관(networksolution.com)에 직접 등록을 하여야 했으나 요즘은 자신이 도메인을 등록한 기관에서 대행해주고 있다.)

1.2 DNS 설정

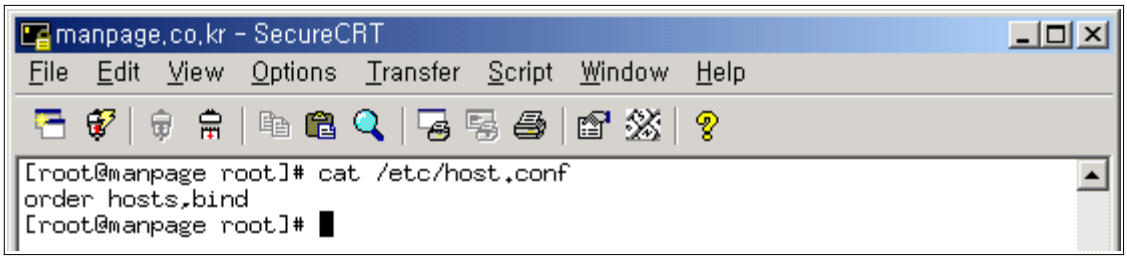


< 그림 1-3 DNS설정을 위한 파일 >

<그림 1-3>은 DNS 설정을 위해서 설정해야 하는 파일들이다.
순서대로 알아보자.

1.2-1 /etc/host.conf

- host.conf는 resolver의 제어를 위한 파일이다. 호스트를 resolver 하는데 어떤 서비스 파일을 쓸것인 결정하는 파일로서 order로 순서를 결정할수 있다.



< 그림 1-4 /etc/host.conf의 내용 >

<그림 1-4> 는 /etc/host.conf의 기본 설정이다.
/etc/host.conf에는 여러개의 옵션이 있으며 그 옵션에 대해서 알아보자.

옵 션	설 명
order	설정값으로는 hosts, bind, nis 가 있으며, hosts - resolver하는데 /etc/hosts 파일을 사용. 속도가 빠름 bind - named가 동작하고 있을때, named에 의해서 질의할수있도록 설정 nis - nis 가 동작하고 있을때 nis에 질의하도록 절정
multi	설정값으로 on과 off가 있으며, /etc/hosts 파일에 하나의 호스트가 여러개의 IP를 가질수 있도록 허용한다. “multi-home"을 가질때 추천한다.

nospoof	설정값으로 on과 off가 있으며, on으로 설정시에 DNS는 hostname과 IP를 검사하는데 만약 이 두값이 서로 일치하지 않을때, 이를 'spoof'라고 부르며, 서버는 그 name을 받아들이지 않고 거절하며, error를 return한다. on으로 설정시에는 spoof를 막을수 있지만, 이 기능을 켜놓게 되면 서버에 접속할 때 마다 값을 비교해야 하므로 서버에 접속할수 있는 시간이 지연되는 단점이 있다. 보안상 필요하면 on으로 선택, default는 off이다.
trim	설정값으로 domain 이름을 가지며, 로컬에 여러개의 도메인이 있을때 당신의 호스트를 결정하기 위해서 사용되어질수 있다. /etc/hosts와 같은 기능을 한다. 기본값은 off이다.

< 표 1-1 /etc/host.conf 옵션 >

```
[root@manpage root]# cat /etc/host.conf
order hosts,bind
multi on
nospoof on
trim manpage.co.kr
[root@manpage root]#
```

< 그림 1-5 /etc/host.conf 의 설정을 적용한 모습 >

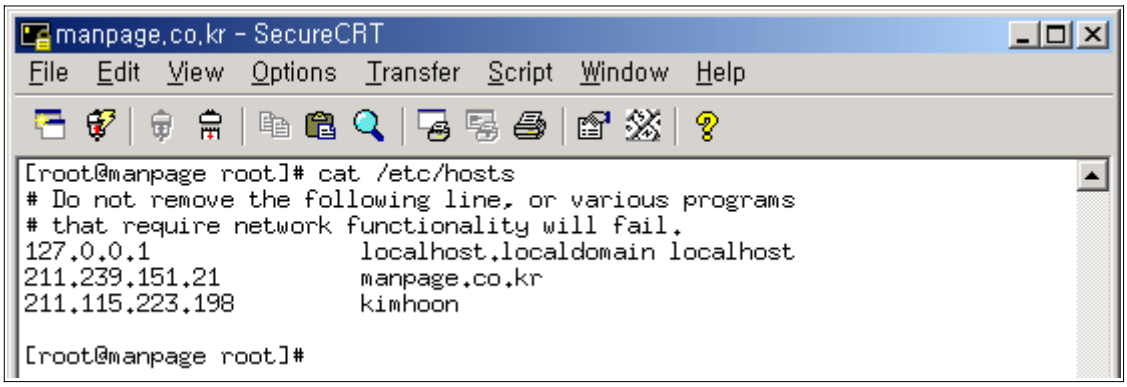
<그림 1-5>와 같은 설정은 절대적인 것이 아니며, 관리자는 옵션의 의미를 잘 판단한후에 적용하여 사용하면 되겠다.
일반 default값인 order hosts,bind 만을 사용해도 운영에는 전혀 지장이 없다.

1.2-2 /etc/hosts

서버에 접속되어지는 모든 클라이언트 들은 모두 DNS에 의해서 reslove 되어진다.
resolve를 위해서는 자신의 DNS를 검색하고, 자신에게 그것이 없을때에는 상위 DNS에 질의를 해야 하므로 접속을 하는데 시간이 많이 걸리게 된다. (tcpwrapper 에 의해서 설정되어지는 데몬들은 (ssh나 pop, ftp등) 접속되는 IP를 모두 inverse reslove 하여, 호스트네임을 알아낸후에 로그에 기록을 하므로, 확인을 위해서 서버에 접속되어지는 시간이 길어지는 것은 그 이유이다) 이때, 자주 접속하는 IP들을 /etc/hosts 에 기록해놓으면, DNS에 의해서 resolve 되지 않아도 되므로, 훨씬 접속이 빠르게 된다.

설정법은 다음과 같다.

```
IP_address canonical_hostname aliases
================================
  IP주소       정규 호스트명       별칭
================================
  IP주소          별칭
```

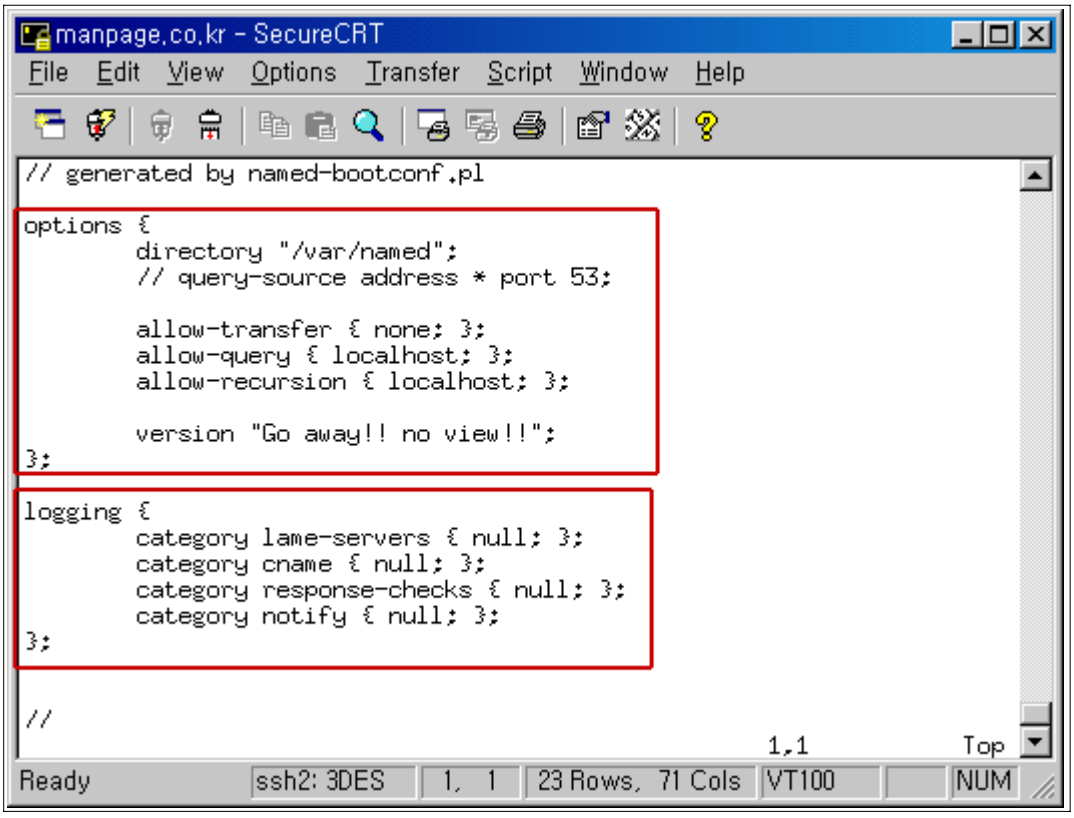


< 그림 1-6 /etc/hosts 파일 설정 >

<그림 1-6>은 /etc/hosts 파일의 설정 예를 보여준 그림이다.
그림에서 보듯이 211.239.151.21과 211.115.223.198은 해당 접속시에 DNS에 의해서 resolve가 되지 않으므로 접속 속도가 빠르며, 로그에는 manpage.co.kr, kimhoon 등으로 로그가 남게 된다.

1.2-3. /etc/named.conf

/etc/named.conf는 /var/named 디렉토리의 경로와 zone파일 설정 정보를 가지고 있다.



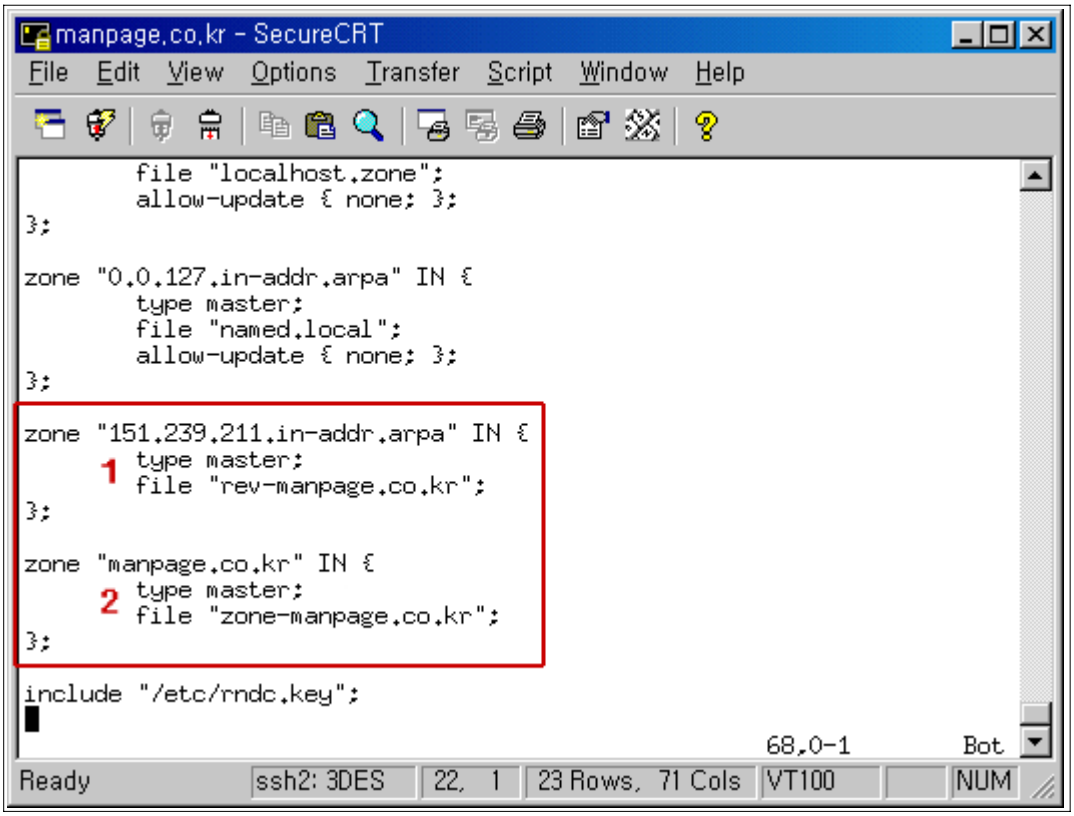
< 그림 1-7 /etc/named.conf의 option과 logging설정 >

<그림 1-7>은 /etc/named.conf의 option과 logging 설정 부분이다.
options를 간단히 설명하면

- directory "/var/named" : zone파일들이 있는 디렉토리의 경로이다.
- allow-transfer { none; }; : 2차 DNS 서버가 있을때 사용하는 옵션으로 zone의 transfer를 허용할것인지 안할것인지 결정한다. 2차가 없을때에는 none 값으로 다른 호스트로의 transfer를 차단한다.
- allow-query { localhost; }; : 일반적인 쿼리를 DNS 보낼수 있는 호스트를 지정한다. "211.239.151.0/24" 로 네트웍대역을 지정할수도 있고, 특정 아이피만을 지정할수도 있다. 현재 설정으로는 localhost에서만 query가 가능하도록 설정했다.

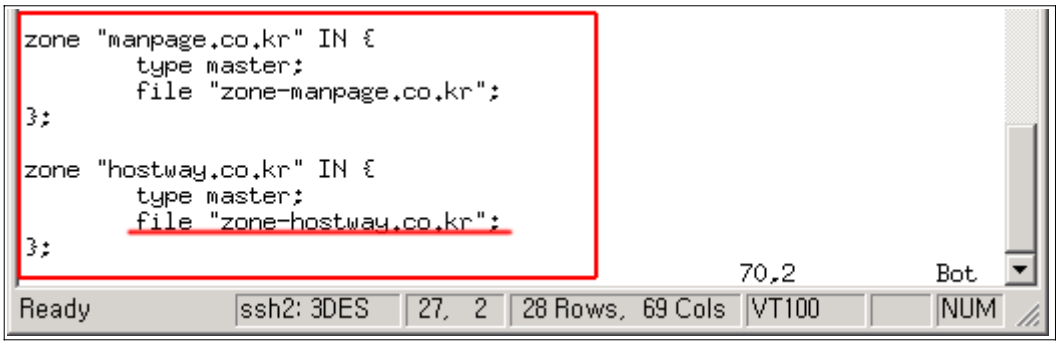
- allow-recursion { localhost; }; : 순환 질의를 할수 있는 호스트를 정의한다. 보안에 관한 부분으로, 인터넷에 있는 모든 호스트들이 해당 서버를 통해서 순환 질의를 하게되면, 캐싱에 문제가 발생할수 있다. 관리자의 설정에 따라 allow-query처럼 값을 변경할수 있다.
- version "Go away! no view!!"; : bind의 버전을 감춘다. bind가 취약한 버전이라고 알려졌을 경우 공격받을수 있으므로 보안상 설정하는 부분이다.

logging을 설명하면,
각종 시스템에서 발생하는 관련 에러들을 필요없는 로그들로 간주하고, null 시킴으로써, syslog의 부하를 감소시킬수 있다. 각 로그가 기록되길 원하면 해당 라인을 삭제 하면 된다.



< 그림 1-8 /etc/named.conf의 zone-file설정 >

/etc/named.conf에서 나머지는 모두 디폴트로 사용하지만, < 그림 1-8> 1과 2는 네임서버 사용을 하기 위해서 반드시 설정을 해줘야 하는 부분이다. 그중에서 **1번의 resolve-zone파일의 경우는 각서버에 한번만 설정이 되어야 하며**, 하나 이상이 설정되었을경우에는 에러가 발생한다.
2번은 resolve할 도메인을 하나씩 추가할때마다, 반복적으로 설정이 되어야 하는 부분이다.
반복될 때에는 해당 도메인의 이름과 존파일의 이름정도만 변경한후에 그대로 사용하면 된다.



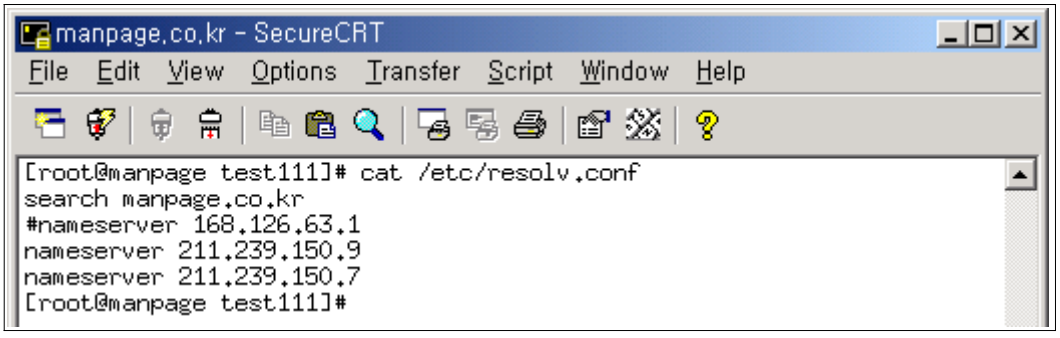
< 그림 1-9 /etc/named.conf의 도메인 추가설정 >

/etc/named.conf에 도메인이 등록이 되면, /var/named 디렉토리 밑에는 <그림 1-9>에서 밑줄친 이름으로 zone 파일이 존재 해야 한다. (/var/named/zone-hostway.co.kr)
이것은 저절로 생성되는 것이 아니라 직접 만들어야 하는 것이다.

1.2-4. /etc/resolv.conf

서버는 user가 어떤 도메인에 대해서 요청을 했을때 그것을 해석하기 위해서 자신에게 할당된 주 네임서버를 찾게 되며, 그 네임서버를 정의하는곳이 resolv.conf 이다.
resolv.conf에 있는 주 네임서버는 user가 요청한 도메인이 자신이 가지고 있는 (named.conf에 설정된) 도메인이면, 바로 응답을 해주고, 만일 자신이 가진 도메인이 아니면 최상위 루트에게 질의를 하여 해당 도메인의 IP를 받아와 클라이언트에게 알려주는 역할을 한다. (그림 1-1 참고)

/etc/resolv.conf에는 3개까지의 nameserver를 지정할수 있다.



< 그림 1-10 /etc/resolv.conf의 설정화면 >

지정한 옵션에 대해서 알아보도록 한다.

-search : 가장 먼저 찾을 도메인을 지정한다.
(search manpage.co.kr 또는 search co.kr com 등의 설정도 가능한다. search가 없어도 상관은 없다.)

-nameserver : 네임서버로 사용할 서버의 IP를 적어준다.
만약 호스트 웨이로부터 211.239.151.21 IP를 할당받아 이 IP를 네임서버로 쓴다면, /etc/resolv.conf의 search 다음라인에 nameserver 211.239.151.21 이라고 우선적으로 적어주면, 설정한 네임서버가 여러개 있을때 클라이언트가 도메인을 요청하면 , 가장 먼저 211.239.151.21의 네임서버에서 클라이언트가 요청한 도메인을 검색하게 된다. 만약 없다면 그 다음 설정한 네임서버에서 요청한 도메인을 검색하게 된다.

<그림 1-10>으로 예를 든다면, 211.239.150.9번 네임서버에서 클라이언트가 요청한 도메인을 우선적으로 찾고, 만약 없다면, 211.239.150.7 네임서버에 질의하게 된다. 만약 211.239.150.7네임서버에서도 없으면 그때는 root 네임서버로 질의를 하게 된다. (168.126.63.1은 주석으로 처리되어 있으므로 해석하지 않는다.

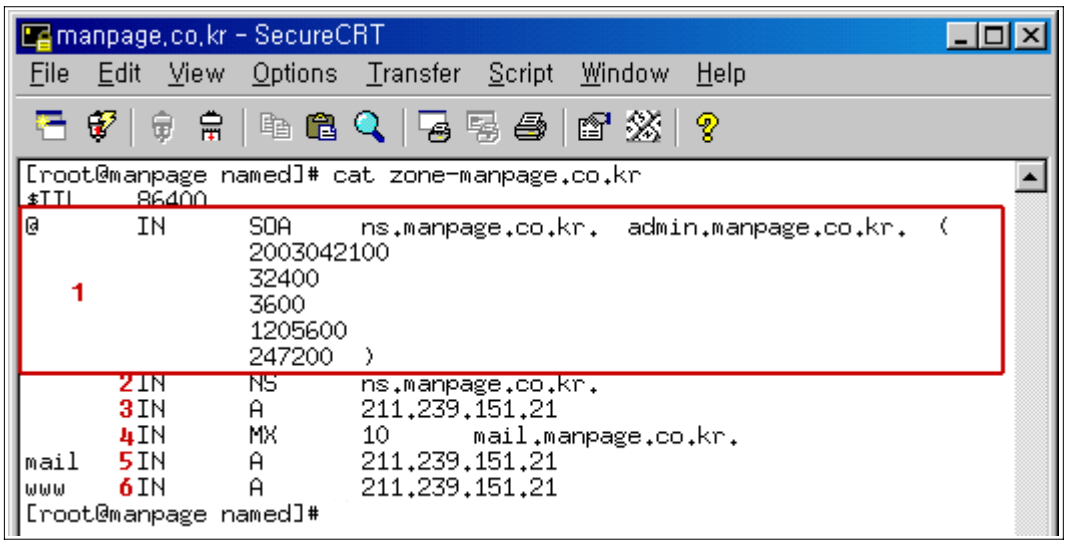
만약 /etc/resolv.conf에 올바른 네임서버가 설정되어있지 않다면, 서버는 도메인으로 외부의 호스트를 검색, 접속할수 없게된다.

※ 참고

서버의 내/외부에서 IP로는 접속이 가능한데, 도메인으로 접속이 불가할때는 네임서버를 가장 먼저 의심해 봐야 한다.

1.2-5. /var/named/zone파일 생성

지금까지의 named 설정은 zone파일의 생성을 위한 준비 단계였다고 볼수있다.
zone file을 생성해야지만 비로소 named서버로서의 역할을 할수 있는것이다.



< 그림 1-11 /var/named/의 zonefile 생성 예 >

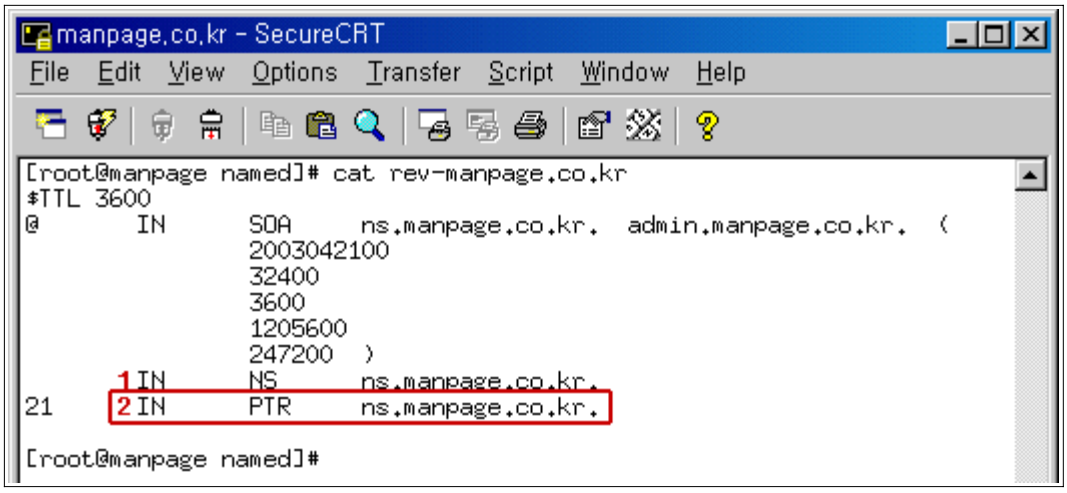
< 그림 1-11>은 zone file의 생성 예이다.

zone 파일의 각 레코드의 설정에 대한 설명을 한다.

레코드 설정값	설명
$TTL 86400	zonefile 정보의 TTL 시간을 지정한다. 특정 레코드가 변경 되었을 때, 변경된 값이 인터넷에 전파되어 업데이트 되는 주기이다. 업데이트가 잦은 사이트는 이 값을 1시간에서 ~ 3시간 정도로 설정한다.
SOA (Start Of Authority)	해당 존파일 도메인에 대한 네임서버 인증 정보를 가지고 있다. 이 레코드의 값은 second nameserver가 있을때에만 유효하고 primary nameserver만 있을 경우에는 필요없는 정보이다
@	같은 네임서버를 의미하면 네임서버 대신에 @를 사용해도 된다.
IN	IN(internet)은 클래스명이다. (chaos 등의 여러 클래스가 있지만 주로 IN만을 사용한다)
SOA	SOA (Start Of Authority) 도메인에 대한 인증정보 시작을 의미
ns.manpage.co.kr.	해당 존파일의 네임서버를 명시
admin.manpage.co.kr.	해당 네임서버의 관리자 이메일을 지정 (= admin@manpage.co.kr)
2003042100	Serial로서 존파일이 최종 갱신된 date를 나타낸다. secondary는 자신이 가진 Serial이 primary의 것보다 작다면 primary로부터 zone 파일을 재전송 받는다.
32400	Refresh로서 secondary에서 primary의 존파일 변경 여부를 검사하는 주기이다. (6시간)
3600	Retry로서 secondary에서 primary로 연결이 되지 않을때 재시도 하는 주기이다. refresh 주기보다 작아야 의미가 있다.
1205600	Expire로 지정된 시간동안 primary에 연결을 하지 못할경우에, 현재 가지고 있는 정보가 유효하지 않다고 판단하고, 해당도메인에 대한 응답을 하지 않는다.
247200	Minimum으로 다른 네임서버가 자신의 zone에 기술된 자료를 가지고 갔을경우, 그 자료에 대한 유효기간을 설정한다. 이 값은 zone파일 상단에 표시된 TTL값을 대치할수 있으며, TTL값이 설정되어 있으면, 이 설정은 유효하지 않다. TTL과 동일한 기능을 한다.
NS	NS(nameserver)레코드, NameServer를 지정한다.
A	A(Address)레코드, 도메인에 IP를 부여하는 역할을 한다.

CNAME	CNAME(Canonical Name) 레코드, 도메인에 대한 다른 이름을 지정할수 있다. **hostway IN A 66.232.139.10** 만약 hostway sub 도메인 가도록 설정한 레코드를 A 레코드로 설정 을 해놓았을 경우에는 hostway.co.kr 의 IP가 변경되었을 경우, zone 파일을 변경시켜줘야 하지만 **hostway IN CNAME hostway.co.kr.** CNAME으로 설정해 놓으면 IP가 바뀌더라도, zone 파일을 변경할 필요가 없이 그대로 사용하면 된다. 단 CNAME은 실제 도메인의 레코드를 상속받기 때문에 추가 레코드를 가질수 없으며, MX, NS등의 레코드에 CNAME으로 설정된 도메인을 넣어서는 안된다. 자신의 서버에서 다른서버로의 포팅이 없다면, CNAME은 사용하지 않는 것이 좋다.
MX	MX(Mail eXchanger) 레코드, 해당 도메인의 메일 라우팅 경로를 설정한다
PTR	PTR(PoinTeR) 레코드, 이 레코드는 IP에 대해 도메인명을 맵핑해 주는 역할을 한다. 일반적인 존파일에서는 사용하지 않고 reverse zone 파일에서만 사용한다.

< 표 1-1 zone file의 레코드 설정 >



< 그림 1-12 /var/named/의 reverse zonefile 생성 예 >

<그림 1-12> PTR레코드를 사용한 reverse Zone파일의 예이다.

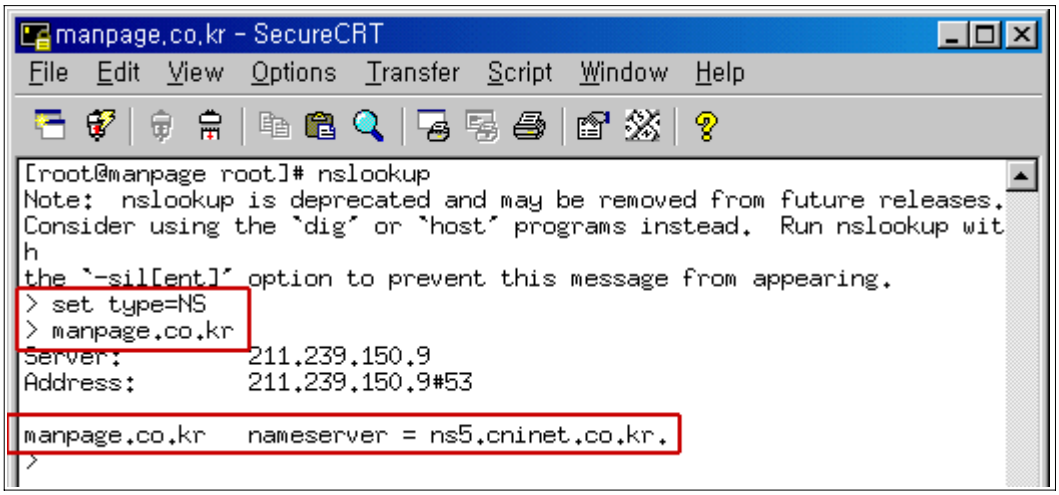
1.3 nameserver 질의 명령

1.3-1 nslookup을 통한 도메인 네임검색

사용법

nslookup -type=RR(Resource Record) domain or set type=RR domain

RR에는 A, ANY, CNAME, MX, NS, PRT, HINFO, SOA, TXT 등이 검색 옵션이 올수 있다.



< 그림 1-13 nslookup 명령 사용하기 >

<그림 1-13>은 nslookup 명령을 실행하여 **set type=NS** 로 nameserver에 질의한 결과이다. 그림에서 알수있듯이 manpage.co.kr은 nameserver로 ns5.cninet.co.kr을 사용하고 있음을 알수가 있다. (쉘명령으로 다음과 같이 하면 같은 결과를 얻는다. **nslookup -type=NS manpage.co.kr**)

1.3-2 dig

<그림 1-13>에서 nslookup을 입력했을때 나오는 공지는 "앞으로 nslookup명령은 사라지게 되고, dig 또는 host명령이 대신하게 될것이다." 라는 내용이다.
즉 앞으로는 dig 또는 host 명령을 잘 알아두어야 할것이다.

사용방법 및 옵션들

dig [@nameserver] 도메인 [쿼리타입] [+쿼리옵션]

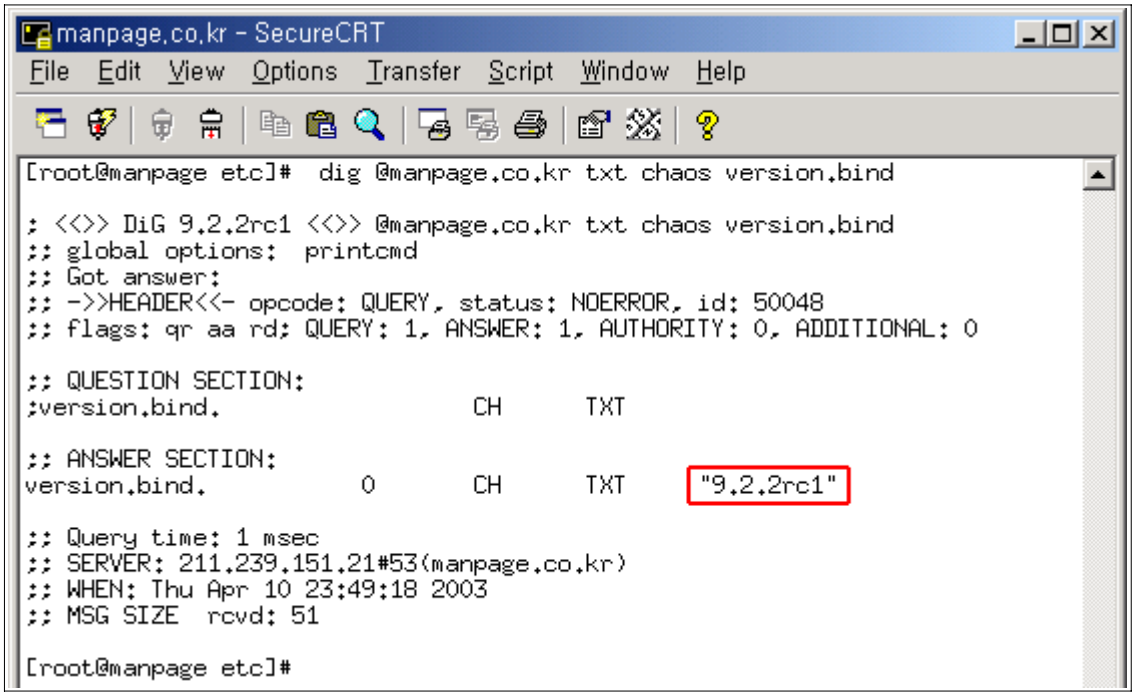
```
dig  [ @server ]  [ -b address ]  [ -c class ]  [ -f file-name ]
     [ -k filename ]  [ -p port# ]  [ -t type ]  [ -x addr ]
   [ -y name:key ]  [ name ]  [ type ]  [ class ]  [ queryopt... ]
```

ex))dig @ns.manpage.co.kr www.manpage.co.kr A

예와 같은 명령의 내용은 ns.manpage.co.kr의 네임서버에 www.manpage.co.kr의 A Record 설정 상태를 출력하라는 명령이다. 결과는 다음과 같다.

www.manpage.co.kr.	3600	IN	A	211.239.151.21

아래 그림은 dig의 또다른 옵션으로 다른 서버의 bind version을 확인하는 방법이다.



<그림 1-14 dig 명령을 이용한 다른서버의 bind버전 확인>

1.3-3 host

dig와 같이 앞으로 많이 쓰이게 될 nameserver 질의 명령이다.
옵션의 종류와 사용법은 다음과 같다.

```
host [ -aCdlnrTwv ] [ -c class ] [ -N ndots ] [ -R num-ber ] [ -t type ]
     [ -W wait ] name [ server ]
```

ex)) host -t ns manpage.co.kr

```
[root@manpage root]# host -t ns manpage.co.kr
manpage.co.kr name server ns5.cninet.co.kr.
[root@manpage root]#
```

< 그림 1-15 host 명령 사용예 >

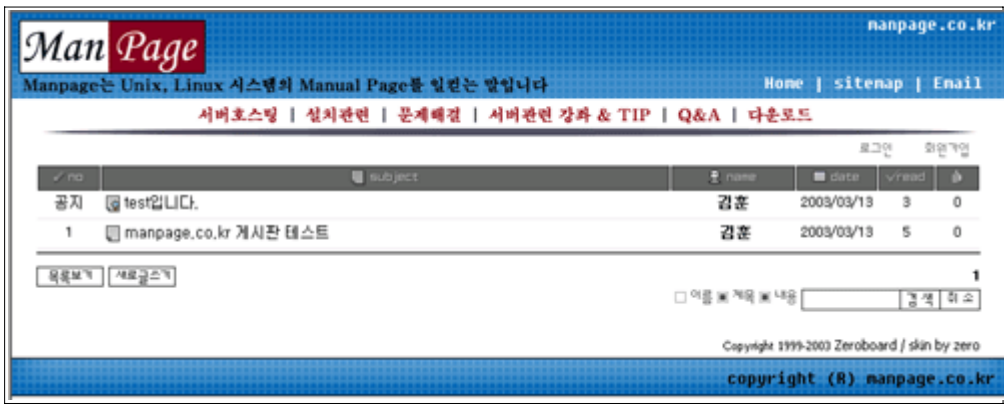
지금까지 네임서버 구축과, 네임서버에 대한 질의 방법을 알아보았다.
이것은 네임서버를 구축하고, 네임서버의 동작유무, 설정의 올바름을 판단하는데 중요한 것이므로
직접 네임서버를 운영하고자 하는 관리자는 반드시 알아두어야 할것이다.

2. WebServer (Apache)

1장에서 DNS서버를 통하여 도메인을 이용하여, 서버에 접속하는방법을 알아보았다.
하지만 아무리 웹브라우저를 통하여 서버에 접속을 했다고 해도, 접속한 서버에 웹서버 프로세서가 동작하고 있지 않다면 다음과 같은 메시지를 만날 수 있다.

<그림 2-1 webserver가 가동되지 않을때 >

웹서버 프로세서란 클라이언트가 웹브라우저등을 이용하여 서버에 접속해서 웹페이지를 볼수있도록 서비스를 하는 프로그램을 말한다. 리눅스에서는 아파치란 프로그램이 대표적이다.
아파치란 프로그램을 구동시키면 80번 포트가 열리면서 웹서비스를 위해서 서버에 접속을 하는 사용자들을 허용하게 되며 다음과 같이 정상적으로 웹서비스를 할수있게 된다.



< 그림 2-2 Webserver가 가동되어 정상적인 서비스가 될 때 >

2-1. apache 설정

이번장에서는 리눅스서버에서 가장 많이 사용되고 있는 아파치 웹서버의 설정에 대해서 알아보도록 한다.

apache의 설정파일인 httpd.conf는 총 3개의 section으로 구성되어있다.

section 1은 전체적인 환경설정부분

section 2는 main 서버의 환경설정

section 3은 가상호스팅에 대한 부분이다.

각 섹션별 주요 설정과 그 설정에 대해 간단히 알아보자.

(defalut로 사용해도 서버 운영상에는 지장이 없으나, 서버의 효율적인 사용을 위해서 변경할수 있는 부분은 적절히 변경하여 사용하는것도 좋다.)

Section 1 : Global Environment

ServerType standalone
- apache웹서가 운영되는 형태 (단독 실행 웹서버로 동작함)

ServerRoot "/usr/local/apache"
- apache가 설치된 경로

PidFile /usr/local/apache/logs/httpd.pid
- apache가 실행될때의 pid값을 가지고 있는 Pid파일의 경로

ScoreBoardFile /usr/local/apache/logs/httpd.scoreboard
- apache의 scoreboard 파일이 있는 경로

Timeout 300
- 클라이언트가 서버에 요청을 한뒤 클라이언트에서 아무런 응답이 없어서 오류로 처리하기 까지의 대기 시간

KeepAlive On
- 서버에 한번 연결을 했던 클라이언트가 다시 연결 요청을 할것이라고 생각하고, 클라이언트의 요청을 처리하고 나서 연결을 끊지 않고 유지한다는 설정으로, 클라이언트가 다시 요청을 했을때 서버와의 접속요청 절차를 거치지 않아도 되므로, 접속 속도가 향상된다. Off에 비해서 20-30% 정도의 성능 향상을 기대할수 있다. 단 동시 접속자가 많을경우에는 메모리가 충분해야 한다. Off로 하였을 경우 On보다 많은 동시 접속자를 처리할수 있지만, 매회 연결때마다 서버와의 통신을 해야 하므로 접속 속도도 늦고, cpu에 로드가 발생한다. 하지만 검색엔진등의 접속자가 아주 많고, 단일 접속만 하고 접속을 끊는 사이트에서는 Off로 사용하는 것이 오히려 도움이 된다.

MaxKeepAliveRequests 100
- 서버가 클라이언트와의 접속을 유지하면서 최대 클라이언트로부터 받을수 있는 요청의 수이다. 즉 한번 연결후에 클라이언트에서 100번의 요청을 받았다면 자동으로 설정을 끊는것이다. 0으로 설정을 하는 것은 클라이언트에서 스스로 접속을 끊을때까지 모든 요청을 받는다는것이다 웹사이트가 복잡하게 되어있는 웹서버에서는 이 값을 늘여주는 것이 좋다.

KeepAliveTimeout 15
- 접속이 유지된다고 해도 하루종일 되어있다면 동시접속자 제한에 걸려서 웹서버의 접속을 할수없게 된다. 이때 설정해 주는값으로 지정된 시간동안 요청이 없으면 연결을 서버에서 연결을 끊어버리게 되어 다른 접속자가 웹서버에 접속할수 있도록 설정해 주는것이다. (동시 접속자가 많을때 웹서버 성능을 너무 떨어뜨리지 않게 하기 위해서 KeepAlive를 On으로 설정해놓고, 이 값을 줄여주는 방법도 있다.)

MinSpareServers 5 (접속자가 많은 경우 10-15 정도로 설정)
- apache는 클라이언트로부터 요청을 받으면 자식 프로세서를 하나 생성하여 그 요청에 응답을 하도록 되어있다. 만약 생성된 프로세서가 없다면 접속시에 지연시간이 발생하므로 속도 증진을 위해서 미리 여분의 응답용 자식 프로세서를 생성해 놓는것이다.

MaxSpareServers 10
- 최대로 만들 수 있는 자식 프로세서의 수를 지정한다. (접속자가 많을 경우 40-50으로 설정)

StartServers 5
- 서버가 시작시에 만들어낼 응답 프로세서의 수를 지정 (MinSpareServers와 동일하게 설정)

MaxClients 150
- 최대 접속할수 있는 동시 접속자수 (호스트웨이에서는 1024까지 확장할수 있도록 컴파일 됨)

MaxRequestsPerChild 0
- 한 자식 프로세서당 처리할수 있는 최대 요청수. default는 0으로, 자식 프로세서가 죽지 않으므로 속도향상을 기대할 수는 있으나, 잘 못된 프로그래밍이나 코드 수행으로 인해서 문제가 발생할 경우에는 서버 전체에 부하를 유발할 수가 있다. (보통 300 정도로 설정하고, 사이트가 복잡 하거나 요청이 많아야 할경우에는 500-1000 등으로 수치를 올려준다.)

Section 2: 'Main' server configuration

Port 80
- apache Webserver가 사용하게될 포트 지정

User nobody
- apache 데몬이 실행될때 소유권자

Group nobody
- apache 데몬이 실행될때 그룹 소유권자

ServerAdmin root@manpage.co.kr
- 서버의 관리자 (관리할 사람의 이메일을 적어놓는다.)

ServerName manpage.co.kr
- 서버의 이름, 호스트명

DocumentRoot "/usr/local/apache/htdocs"
- apache 웹서버의 기본 홈 디렉토리

```
<Directory /home/*/public_html>
    AllowOverride FileInfo AuthConfig Limit
    Options MultiViews Indexes SymLinksIfOwnerMatch IncludesNoExec
</Directory>
```
- /home/*/public_html 디렉토리에 대한 디렉토리 옵션을 나타낸다.

디렉토리 옵션에는 그 종류가 여러 가지가 있으며, 다음에 자세히 알아보기로 한다.

```
<IfModule mod_dir.c>
    DirectoryIndex index.html index.php index.htm index.php3
</IfModule>
```
- 클라이언트가 웹서버에 대한 요청이 있을때 해당 도메인이 디렉토리 안에서 가장 먼저 읽을 파일을 지정한다.

DefaultType text/plain
- 서버에서 사용되는 문서 중 MIME type으로 알수 없는 문서에 대해 기본 적용할 기본 MIME type을 설정한다. 서버에 문서파일이 많을때 text/plain을 사용하며 바이너리나 실행파일이 많을 경우 application/octet-stream을 적용해준다.

HostnameLookups Off
- 접속하는 서버들의 IP를 DNS에게 질의하는 옵션, 만약 On으로 설정시에는 IP를 DNS서버에 질의해서 로그에 기록하므로 시간이 오래 걸리게 된다. 성능향상을 위해서 Off 적용

ErrorLog /usr/local/apache/logs/error_log
- 서버의 기본 error_log를 기록하는 파일의 경로

LogLevel warn
- log를 기록하는 level을 정한다. 로그를 기록하는 level은 debug, notice, warn, error, crit, alert, emerg 가 있으며, debug쪽에 갈수록 사소한 문제까지도 로그에 기록하므로 로그의 크기가 커지게 되고, emerg쪽으로 갈수록 비상시의 문제 발생시에만 로그에 기록이 되므로 로그의 크기가 그다지 커지지는 않는다. default로는 warn이며, 필요시에 따라 로그 기록하는 level을 변경할 수가 있다.

CustomLog /usr/local/apache/logs/access_log common
- 서버의 기본 접속 기록에 대한 로그파일의 경로이다.

AddType application/x-httpd-php .php .php3 .ph .inc .html .htm
AddType application/x-httpd-php-source .phps
- 위 두 라인은 php를 사용할 때 필요한 옵션이다.

AddHandler cgi-script .cgi
- cgi를 사용하고자 할때 사용

AddType text/html .shtml
AddHandler server-parsed .shtml
- 위 두 라인은 shtml을 사용하고자 할때 사용

Section 3: Virtual Hosts

NameVirtualHost 211.239.151.21
- 이름기반 가상호스팅을 사용한다.

<VirtualHost 211.239.151.21>
- 한 아이피에 여러개의 도메인을 사용하고자 할때 설정하는 것으로 이곳에 IP대신에 도메인 이름을 적용해도 무방하다.

ServerAdmin manpage@manpage.co.kr
- 해당 도메인을 관리자의 이메일

ServerName manpage.co.kr
- 해당 도메인의 도메인 명

ServerAlias www.manpage.co.kr
- 해당 도메인의 별칭

DocumentRoot /home/manpage
- 해당 도메인의 홈 디렉토리

ErrorLog logs/manpage.co.kr-error_log
- 해당 도메인의 에러상황이 쌓이는 로그파일

CustomLog logs/manpage.co.kr-access_log common
- 해당 도메인의 정상적인 접속 상황이 쌓이는 로그파일

</VirtualHost>

<표 2-1 httpd.conf의 section별 주요 설정 >

▲ 디렉토리 제어와 옵션

디렉토리를 제어할때에는 <Directory> ~ </Directory>의 형태로 작성하며, 각각의 디렉토리에 대한 권한과 여러 가지 설정을 할 수가 있으며, 이때 사용되는 옵션에 대해서 알아보자.

None : 어떤 옵션도 이용할수 없음을 나타낸다. 모든 접근 불가
All : MultiViews 옵션을 이용할수 있으므로 모든 접근 허용
Indexes : httpd.conf설정에서 DirectoryIndex에 설정되어있는 파일이 해당 URL에 없을경우에 디렉토리의 파일 목록을 보여준다. (보안상 사용하지 않는 것이 좋다)
Includes : 서버측에 추가적인 정보를 제공할수 있도록 한다. (shtml파일 사용시 적용)
FollowSymLinks : 디렉토리의 심볼릭 링크를 사용할수 있도록 한다.
ExecCGI : CGI 스크립트를 실행시킬수 있도록 한다.
MultiViews : All 옵션이 설정 되었을때에만 지정된 목록의 multiviews를 허용한다.
AllowOverride : 사용자 인증에 관련된 지시자로 클라이언트가 웹 서버의 특정 디렉토리에 접근할 때 해당 디렉토리에 있는 유저 인증 파일인 .htaccess파일을 읽게 되는데, 이 옵션이 none으로 설정이 되어있으면, apache는 이 인증파일을 무시한다. 즉 인증을 할수 없다. [None , All] 두가지 중에서 설정할수 있다.
Order : 서버가 access 컨드롤을 수행하는 순서를 지정한다. Order allow,deny (allow 기능을 먼저 수행하고, deny 기능을 수행한다.)
Allow from : 나열되는 주소들에 대한 access 컨드롤을 허용한다. 사용가능한 설정은 도메인명, 호스트명, IP, 이모든 것을 허용하는 all 이있다. Allow from all [211.xxx.xxx.xxx 211.xxx.xxx manpage.co.kr]
Deny from : 나열하는 주소들에 대한 access 컨드롤을 제한한다. Deny from all [211.xxx.xxx.xxx 211.xxx.xxx manpage.co.kr]
require : 사용자 그룹에 대한 접근을 통제 한다. require user [group valid-user] <ID> user - 지정된 사용자들에게만 접근을 허용한다. AuthUserFile에서 지정 group - 지정된 그룹에게만 접근을 허용한다. AuthGroupFile에서 지정 valid-user - AuthUserFile에 있는 모든 사용자들에게 접근을 허용한다.

< 표 2-2 httpd.conf의 디렉토리 제어에 사용되는 옵션 >

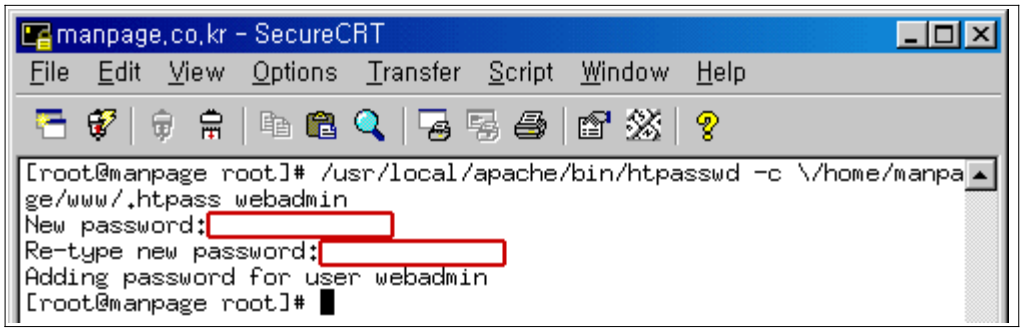
표 2-2에서 디렉토리 제어 옵션에 대해서 알아보았다. 실제 이것이 어떻게 쓰이고 있는지 설정예를 알아보도록 하겠다.

```
<Directory /home/manpage/www>
        AllowOverride All
        Options Includes Indexes +ExecCGI
        Order allow,deny
        Allow from All
        Deny from 211.xxx.xxx.xxx 211.xx.xxy
</Directory>
```

<표 2-3 Directory 제어 사용예 >

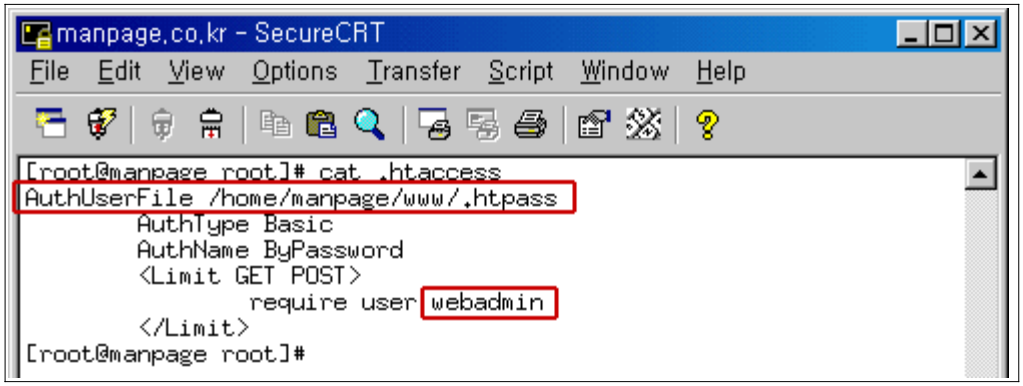
<표 2-3>은 <표 2-2>를 이용하여 디렉토리제어 옵션의 설정 예를 들은것이다.
이 설정에 대한 해석은 각자 스스로 해보기 바란다.

설정을 적용하는 법을 알았으니 이제는 해당 디렉토리(웹페이지)를 접속하고자 하면 인증창을 요구하도록 설정해보자.
인증창 설정은 <표 2-3>의 설정이 httpd.conf에 설정되어 있을때에만 정상적으로 동작할것이다.
httpd.conf 파일을 변경 하였을때에는 apache를 다시 재시작해줘야 갱신이 된다.



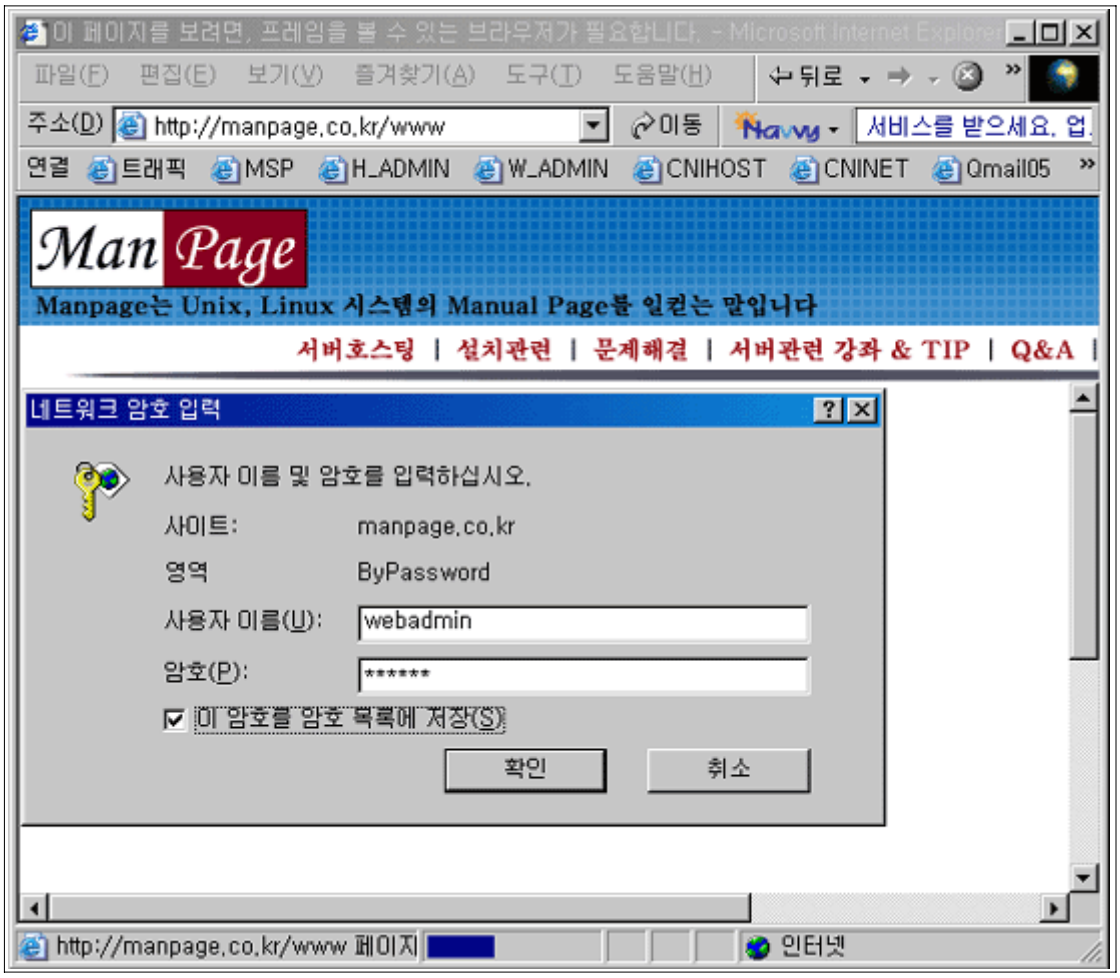
<그림 2-3 passwd 설정 >

htpasswd 명령을 이용하여 webadmin의 패스워드를 /home/manpage/www/.htpass 에 생성하였다. htpasswd -c 옵션은 처음 생성할때만 사용하고 webadmin의 패스워드를 변경하고자 할때에는 -c 옵션을 제외한 나머지 명령을 입력하면 된다.



< 그림 2-4 .htaccess 파일의 내용 >

.htacess를 생성하여 AllowOverride로 하여금 .htaccess파일을 읽어서 webadmin에 대해서 인증을 할수 있도록 설정한 것이다.
<그림 2-4>의 설정은 AuthUserFile인 .htpass에 지정된 webadmin이란 유저의 패스워드를 입력된 값과 비교하기 위한 설정 부분이다.
그리고 생성한 .htpass와 .htaccess파일은 모두 /home/manpage/www 디렉토리 안에 존재해야 한다.

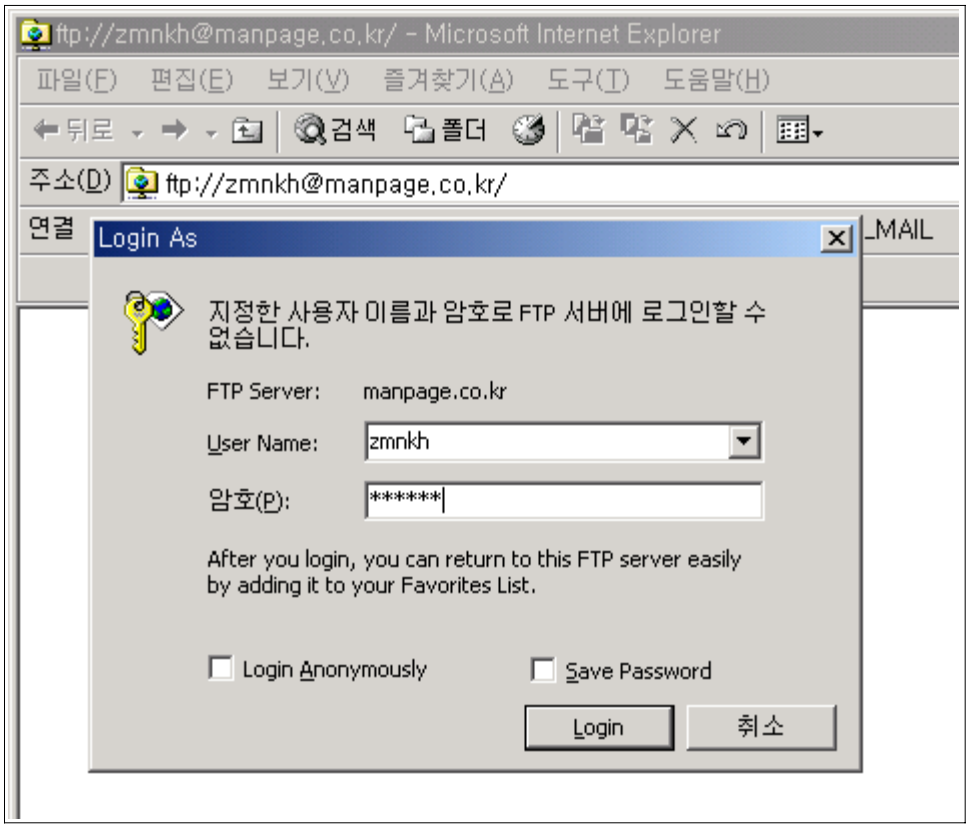


< 그림 2-5 웹인증 확인 >

정상적으로 설정이 마무리 되었다면 ,<그림 2-5>의 경로로 접속을 하면 웹 인증 화면이 뜨는 것을 확인할 수가 있어야 한다.
위에서 설정한 모든 것들은 테스트하는 서버에서 작성되 내용이므로 이글을 읽는 관리자는 자신이 설정하는 곳의 경로로서 파일을 생성해 주면 된다.
여기서 설명한 것은 어떻게 사용하라고 알려주는 예 이므로 각자 환경에 맞게 변경하여 적용시키면 정상적으로 설정을 할수 있을것이다.

3. FTP

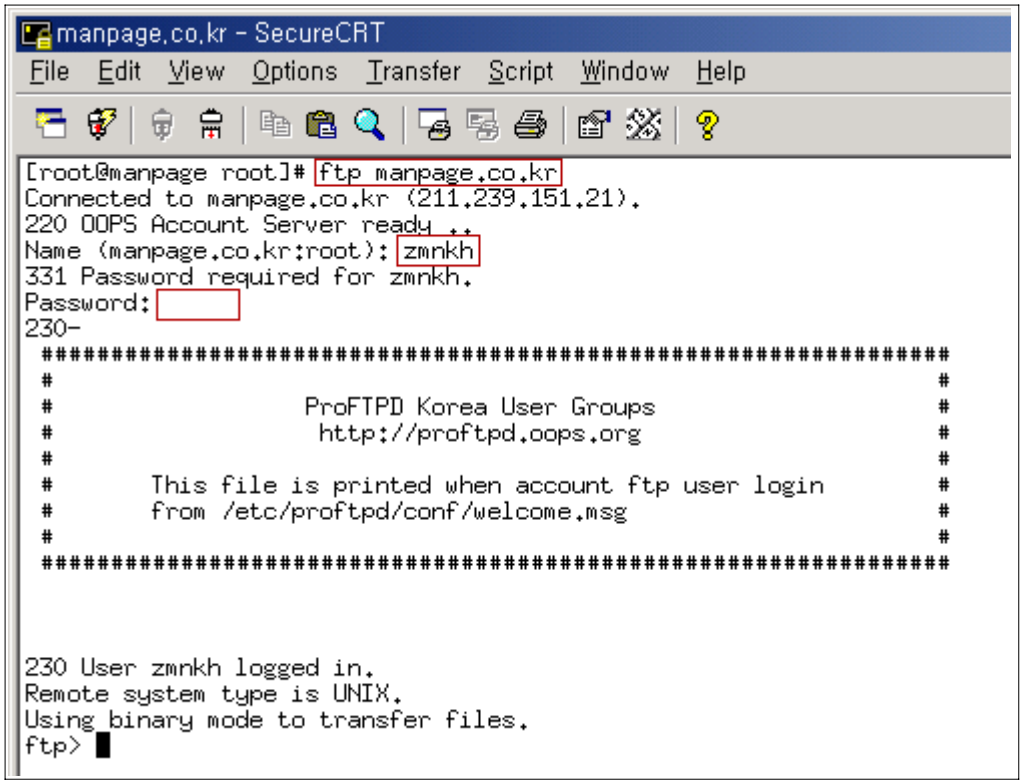
FTP는 File Transfer Protocol 의 약자로서 파일 송수신을 위한 프로토콜이다.
FTP는 기존 연결을 위해서는 21번 포트를 사용하지만 데이터의 전송은 20번 포트를 사용한다.
호스트웨이에서는 서버에서 가장 많이 사용되고 있는 ProFTP최신버전을 RPM으로 제공하고 있다.
윈도우에서는 ALFTP등의 FTP 접속 프로그램을 통해서 FTP서버에 접속을 할수 있고,
또한 웹브라우저에서도 FTP서버로 접속이 가능하다.
웹브라우저로 접속을 할때에는 "ftp://이메일형식"으로 하면 바로 ftp접속을 할수있다.



<그림 3-1 웹브라우저를 통한 FTP 접속>

위의 그림은 웹브라우저를 통한 FTP접속이다.
위의 그림에서는 패스워드를 물어보는 창이 떠있지만.
"ftp://아이디:패스워드@도메인" 형식으로 주소창에 쓰게되면 암호 물어보는 팝업창이 뜨지 않고 바로 FTP서버로 접속을 하게 된다.

아래그림은 서버에서 다른 서버로 FTP 접속을 하는 그림이다.



<그림 3-2 서버에서 다른서버로 FTP 접속>

명 령	설 명
FTP 도메인명	도메인명으로 FTP 접속을 시도한다.
prompt	상호작용모드 on/off Interactive mode off (한번 명령 내릴시 off로 변환) off 시 파일을 송/수신시 실행여부를 물어보지 않는다. Interactive mode on. (다시 한번 명령 내릴시 on으로 변환) on 일때는 파일을 송/수신시 실행여부를 물어본다.
ascii	ftp 송/수신 파일이 ascii (text등) 일 경우에 설정한다.
bin	ftp 송/수신 파일이 binary일 경우에 설정한다.
get 파일명	다른 서버에서 자료를 다운로드 할수 있다. (한개의 파일만 다운로드 할수 있다.)
mget 파일명들	다른 서버에서 한번에 여러개의 자료를 다운로드할 때 사용한다. (*.htm 등 여러 파일을 한꺼번에 다운로드 한다.)
put 파일명	서버에서 다른서버로 자료를 업로드할 때 사용한다.
mput 파일명들	서버에서 다른 서버로 여러자료를 업로드 할때 사용한다.
quit	FTP를 종료한다.
exit	FTP를 종료한다.(OS 따라서 exit가 안되는것도 있다)

< 표 3-1 서버에서 FTP 접속시 사용하는 명령 >

위에 정렬된 것말고도 ls, cd 등의 리눅스 명령도 적용이 된다.

3-1. ProFTP의 설정

proFTP의 설정 파일은 /etc/proftpd/conf/ 들어있다.

conf디렉토리에는 총 3개의 파일이 들어있는데,

- ftpusers : ftp를 허용하지 않는 계정들을 설정해놓는다. 이 파일안에 계정명이 있으면 ftp를 사용할수 없다. ftp는 기본적으로 root가 접속되지 않도록 되어있는데 그 이유는 이파일안에 루트의 계정이 설정되어 있기 때문이다. 바꿔서 말하면 이파일에서 설정되어있는 root계정을 삭제한다면, root로도 ftp접속이 가능하게 된다.
- welcome.msg : ftp 접속이 성공했을때 나타나는 접속 메시지를 이곳에 설정한다. 위의 서버에서 다른서버로 ftp 접속을 하는그림을 보았을때, 주석(#) 으로 나타나는 문구가 바로 이 파일안에 들어있는 내용이 출력된 것이다. 이것은 사용자가 마음대로 변경이 가능하다.
- proftpd.conf : 이것이 바로 ftp의 환경설정 파일이다.
  호스트웨이에서 서버에 기본으로 설치되는 proftp는 한글판으로서 설정파일에 설정하나하나에 한글 설명이 친절하게 되어있다.
  그냥 설정파일에 있는 내용을 읽어봐도 그 사용방법을 알수있을것이지만 가장 많이 변경을 하게되는 몇몇 라인을 살펴보도록 하겠다.

```
# ServerType                    standalone
ServerType                      inetd
→ standalone 모드로 가동할것인지 inetd모드로 가동할것인지 결정한다.

#DefaultRoot                    ~       !groupname
→앞에 주석을 해제하게 되면 ftp사용자들은 자기 디렉토리를 벗어나지 못한다.
    웹호스팅 서버등으로 이용할때에는 위의 주석을 해지해 주는 것이 보안상좋다.

#Port                           21
→ ftp 가동시 사용할 포트이므로 standalone 모드로 ftp를 사용할때에는 앞의
   주석을 제거해준다.

RootLogin                       off
→ 위에서 설명한 ftpusers에서 root 계정을 제거한후에 위의 옵션을 ON으로
    바꾸게 되면 root계정으로 ftp를 사용할수 있게된다.
```

< /etc/proftpd/conf/proftpd.conf 주요 설정 >

위의 설정들이 proFTP를 사용하면서 가장 많이 변경을 하게 되는 부분이다. 나머지 부분들은 디폴트로 사용해도 무관하다.

3-2. Standalone 모드와 inetd 모드

위의 proftpd.conf의 설정을 보게 되면 호스트웨이에서 설정되는 FTP는 기본적으로 inetd 모드로 설정이 되어있다.

과연 Standalone 모드와 inetd 모드라는 것은 어떤것인가?

Stanalone 모드는 말그대로 다른것에 영향을 받지 않고 단독으로 FTP 데몬을 실행하는 것을 말한다. 즉 sshd나 httpd처럼 단독으로 항상 ftpd데몬이 떠있다가 ftp접속 요청이 들어오면 프로세서를 하나 생성하여 접속을 허용하는 방식으로 항상 데몬이 떠서 접속이 들어오기를 대기하고 있으므로, inetd모드보다는 접속 속도에 향상을 기대할수 있다.

standalone의 실행은 위의 proftpd.conf에서 standalone 모드의 주석(#)을 제거하고,
inetd 모드를 주석처리 한다음, MaxInstances 의 주석을 해제해준다. (이것은 FTP의 프로세서를 30개로 제한한다는 것으로 DoS 공격등의 방지를 위한 설정이다.)

```
ServerType              standalone
#ServerType             inetd
.....
.....
MaxInstances            30
```

/etc/xinetd.d/ftp를 열어서

```
vi /etc/xinetd.d/ftp
```

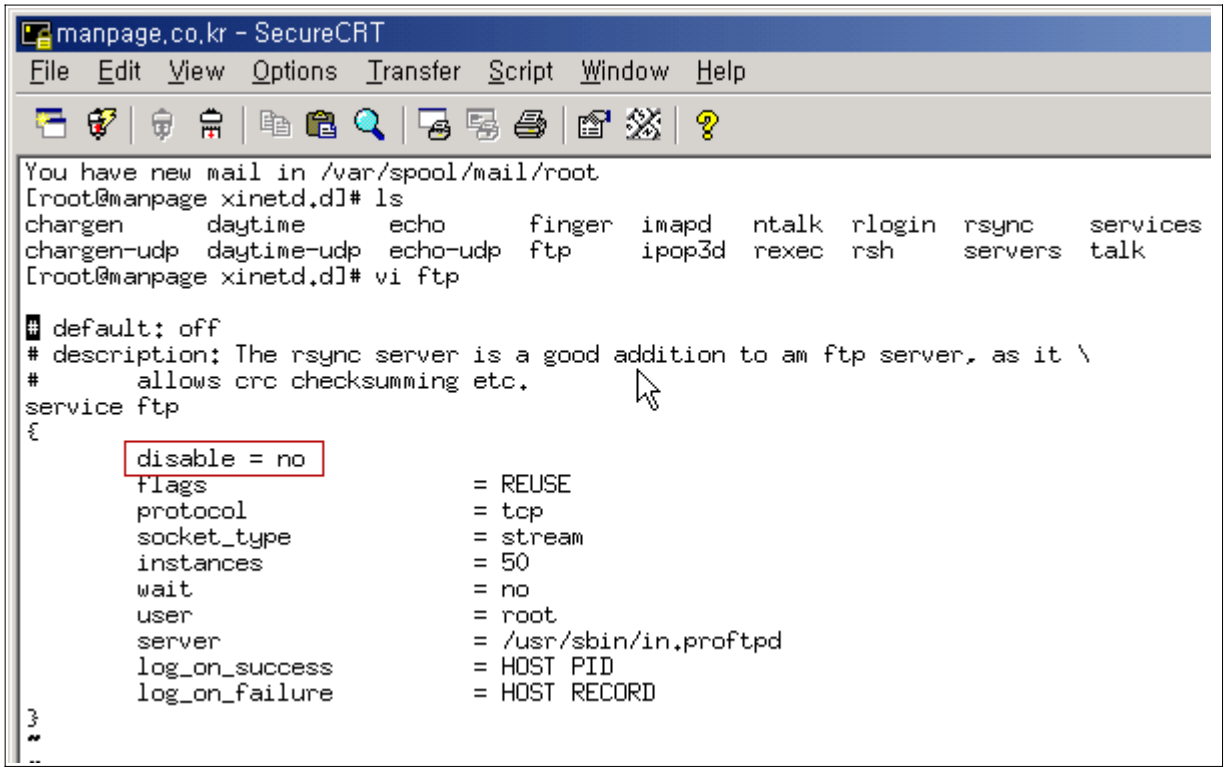
※ ftp란 이름으로 안되어 있을수도 있으므로 직접 이동 후에 확인한다.

disable을 yes로 변경해주고,

```
disable = yes
```

/etc/rc.d/init.d/xinetd restart 하고, (Xinetd 재가동)
/etc/rc.d/init.d/proftp-standalone start (혹은 **restart**)해주면 standalone모드로 ftp를 사용할 수 있다.

반면 inetd 모드는 xinetd라는 슈퍼데몬에 포함되어 있는 경우를 말한다.
xinetd 슈퍼데몬은 telnet, pop, imap등 서버의 인터넷 서비스의 데몬들을 일괄 관리하는 데몬을 말한다.
이 xinetd가 관리하는 데몬들은 /etc/xinetd.d안에 들어있으며, proftp의 경우에는 ftp라는 이름으로 설정이 되어있다.



< 그림 3-3 xinetd 모드의 FTP설정 내용 >

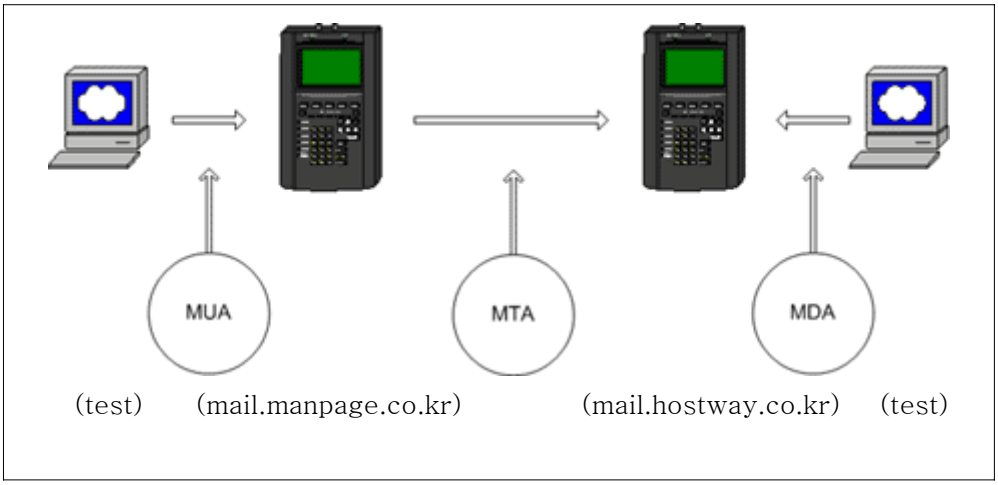
위의 그림은 /etc/xinetd.d/ftp의 설정 내용이다.
disable이 no로 되어있어야 정상적으로 동작을 할수 있는 것을 말한다.
stanalone 모드로 동작을 시킬때에는 위에서 설명을 했지만 **disable = yes** 로 바꿔주고 **xinetd를 재가동**시켜줘야 정상적으로 standalone모드로 동작을 시킬 수 있다.

이처럼 ftp를 xinetd 모드로 실행시키는 이유는 데몬을 슈퍼데몬에서 통합관리를 하게되므로 xinetd 데몬만 띄어주면 ftp 까지 한번에 실행이 되기 때문에 관리상 편리하기 때문이다.

4. Mail & POP 서버

인터넷을 사용하다가 보면 메일을 보내야할 경우가 반드시 생기게 된다.
그럴 때면 hanmail, chollian 등에서 무료로 제공해주는 메일을 사용하여 메일을 보낼 때가 있다. 하지만 서버를 운영하고 직접 도메인을 가지고 있으면, 자신의 도메인도 홍보하고 원하는 만큼의 메일 계정 생성, SMTP서버의 필요성 때문에 메일서버를 구축하게 된다. 특히 자신의 서버로 웹호스팅을 하는 서버관리자라면 메일 서버는 꼭 필요할 것이다.

그러면 메일서버의 필요성을 알아보았으니 이제는 메일이 어떻게 전송이 되는지 알아볼 필요가 있다.



<그림 4-1 메일이 전송/수신되는 개략도>

MUA (Mail User Agent) : 아웃룩등의 메일을 보낼때 사용하는 프로그램들
MTA (Mail Transfer Agent) : 서버에서 다른서버로 메일을 전송할 때 사용하는 프로그램. Sendmail, Qmail등
MDA (Mail Delivery Agent) : pop, imap 등의 프로그램으로 메일을 서버에서 받아올 때 사용하는 프로그램 (아웃룩, 유도라 등)

위의 그림은 test@manpage.co.kr 이라는 사용자가 manpage.co.kr메일서버를 이용하여 test@hostway.co.kr의 사용자에게 메일을 발송하고, test@hostway.co.kr사용자는 서버로부터 메일을 확인하는 과정을 간략히 나타낸 것이다.

메일서버를 운영하다가 갑자기 메일이 안될때 위의 그림을 잘 생각해보면 어느 쪽에서 문제가 발생했는지 알 수 있을 것이다.

4-1 메일 프로그램의 종류

리눅스 서버에서 운영되는 메일프로그램의 종류는 크게 Sendmail과 Qmail로서 나뉠 수가 있다.
호스트웨이에서는 서버설치시 RPM Sendmail을 기본적으로 설치 제공하고 있다.

다음은 Sendmail과 Qmail의 간단한 비교이다.

	Sendmail	Qmail
수 신	한개의 프로세서가 담당	수신담당 프로세서가 담당(qmail-send)
송 신		송신담당 프로세서가 담당 (qmail-smtp)
로 그	/var/log/maillog 생성 (로그정보가 많음)	/var/log/maillog (로그정보가 적음)
Mailbox	/var/spool/mail/사용자아이디 에 메일이 쌓임	일반적으로 사용자의 홈디렉토리밑에 생성됨 (new / cur / tmp)
설치 경로	/etc/mail/	/var/qmail/
프로세서수	1	설치에 따라서 12-14개
속 도	Qmail비해서 느림	분업화로 인해서 속도 빠름
주 사용규모	소/중 규모에서 사용	대형 규모에서 사용
특 징	▪로그분석이 쉬움 ▪많은 사용자들이 사용하고 있어 자료나 문제해결시 편리함 ▪rpm으로도 제공되어 설치가 쉬움 ▪프로세서 관리가 쉬움 ▪서버안의 실제 계정에 대해서만 메일이 관리될수 있다.	▪여러 프로세서가 분업화되어 동시적으로 작동함으로 속도가 빠름 ▪메일하나하나 다른 파일로 기록되므로 메일관리가 쉬움 ▪가상유저방식으로 가상호스팅에 강함

<표 4-1 Sendmail과 Qmail의 비교>

qmail과 sendmail 두 메일러의 특징을 비교해 보았다.

이번에는 Redhat CD를 설치하게 되면 기본적으로 설치되는 sendmail의 설정법과 사용법에 대해서 알아보도록 하겠다. (qmail은 기본 패키지가 아니라 따로 설치를 해줘야 하는 메일러이므로 설명하지 않는다.)

4-2. Sendmail의 설정

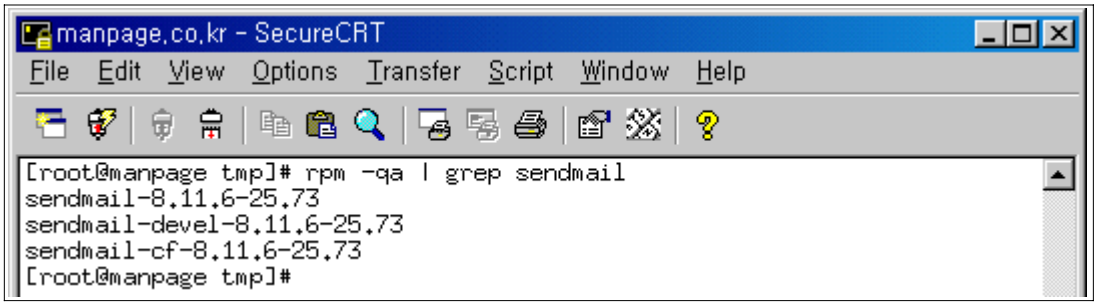
Sendmail은 전 세계적으로 가장 많이 사용되는 메일러이다. 그렇기 때문에 오류나 버그에 대해서 다른 메일러보다 패치와 정보가 빨리 나오기 때문에 Sendmail을 많이 사용하게 된다.

여기에서는 sendmail rpm버전을 설치하여 설정하는 법을 알아보도록 하겠다.

#) 단 Redhat 7.3 기본 패키지에 설치되어있는 sendmail의 경우는 root권한을 취득할 수 있는 버그가 존재하므로 반드시 패치하여 사용하기 바란다.
패치는 redhat 홈페이지나, hostway 서버호스팅 고객지원 페이지, manpage.co.kr에 들어가면 구할수 있다. 보안상 중요한 사항이므로 반드시 패치를 해야 한다. 여기서는 7.3에 설치된 패키지만을 가지고 설명을 하므로 다른 버전의 사용자들은 안내 되어있는 패키지 버전을 따라서 패치해야 한다.

4-2-1. Sendmail의 설정 (RPM버전)

자신의 서버에 sendmail이 설치되어 있는지 확인하기 위해서는
rpm -qa | grep sendmail 이라는 명령을 사용하면 된다.



< 그림 4-2 sendmail 패키지 확인 >

그림 4-2를 보게 되면 패치된 버전인 sendmail-8.11.6-25.73이 설치되어있는 것을 알수 있다.

기본적으로 설치되어있는 sendmail은 아무런 설정을 변경하지 않았을때에는 localhost에서만 메일을 주고 받을 수 있도록 설정되어있다. 즉 내부에서만 메일이 되고, 외부에서는 메일서버에 접근할 수가 없다.
그럴때에는 /etc/sendmail.cf 파일을 열어서 다음과 같이 Addr=127.0.0.1을 제거해 주면 된다.

```
# SMTP daemon options

O DaemonPortOptions=Port=smtp,Addr=127.0.0.1, Name=MTA

# SMTP client options
#O ClientPortOptions=Address=0.0.0.0
```

< 그림 4-3 sendmail.cf 설정변경 >

그리고 나서 /etc/rc.d/init.d/sendmail restart를 해주게 되면 새로운 설정이 적용되어, 외부에서도 SMTP에 접속할수 있게 된다.

4-2-2. Sendmail의 설정 (aliases 갱신)

시스템에서 보내오는 메일등을 받아보거나 특정 유저에게로 메일을 포워딩 하고자 할때 aliases에 적용하게 된다. 적용하고 난후에는 새로운 설정을 갱신해 주어야 하는데 다음과 같은 명령으로 갱신해 줄수 있다.

```
[root@manpage tmp]# newaliases
/etc/aliases: 40 aliases, longest 10 bytes, 395 bytes total
```

< 그림 4-4 aliases 갱신 >

4-2-3. 가상도메인을 위한 /etc/mail 의 설정

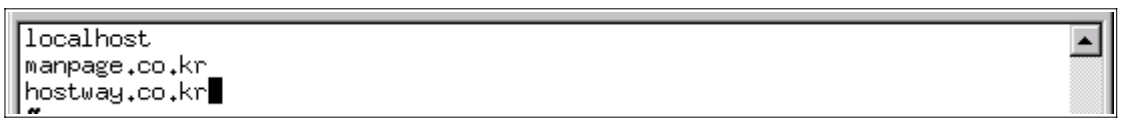
여기까지만 설정을 마치더라도 단독 도메인의 경우는 바로 사용이 가능하다.
하지만 한서버에서 도메인을 여러개 사용하는 메일서버의 경우에는 좀더 추가 설정을 해주어야만 메일서버로서 기능을 할수 있게 된다.
RPM으로 설치가 되었다고 가정하고 설정을 하는것임을 다시한번 주의해야 한다.
(소스로 설치후에 같은 내용을 적용하고자 한다면, sendmail.mc 파일에

```
FEATURE(`virtusertable',`hash -o /etc/mail/virtusertable.db')
```

위와 같은 라인이 있어야 가상도메인을 위한 메일을 세팅할 수가 있다.

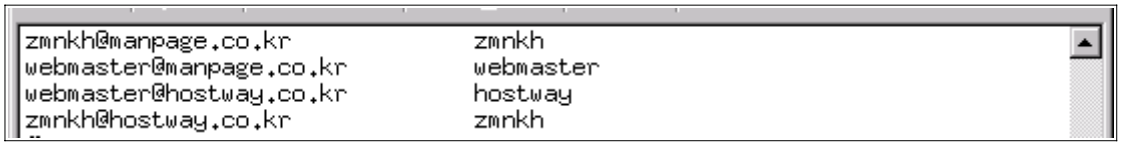
하지만 여기서는 rpm으로 설정을 하기 때문에 이미 위의 설정은 적용되어있다.
/etc/mail 로 이동을 해본다.
여러개의 파일이 존재 하지만, 실제로 설정할 때 변경을 시켜야 하는 파일은 **local-host-names** 파일과 **virtusertable** 파일이다.
그럼 각 파일의 설정방법에 대해서 알아보도록 한다.

local-host-names : 메일을 교환을 허용하는 도메인들을 기록하는 곳이다.
이곳에 설정되어있는 도메인들에 대해서만 메일을 수신할수 있다.
서버내에 설정된 메일을 수신하는 도메인들은 모두 적어 둔다. (이 파일에 등록된 도메인에 대해서는 로컬의 메시지 큐에 메일을 보관하게 되는데 이곳에 메일이 없게되면 sendmail은 다른곳으로 메일을 보내게 되고 결국 메일은 목적지 도메인을 찾지 못해서 배달되지 않게 된다.)



< 그림 4-4 /etc/mail/local-host-names 설정 >

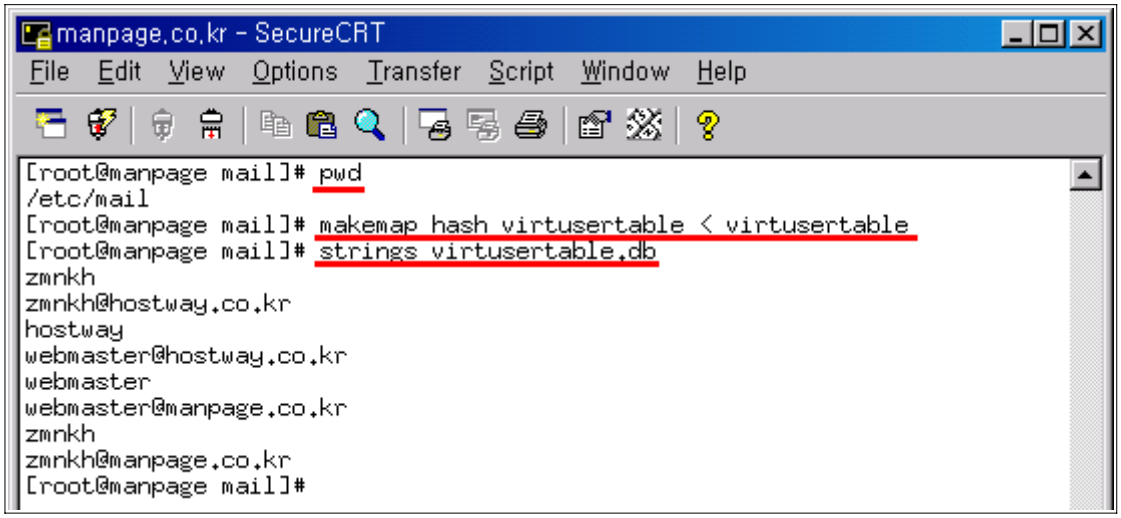
virtusertable : local-host-names에 등록된 메일들이 서버로 수신되었을 때 각각의 계정으로 메일을 배달하기 위해 설정하는 부분이다. sendmail은 리눅스에 존재하는 실제 계정에 대해서만 메일을 배달한다. (virtusertable을 사용하는 목적은 각 계정으로의 메일 포워딩도 그 목적이 될 수 있지만, 각각의 도메인에 원하는 이메일 계정을 부여하기 위해서 사용한다.)



< 그림 4-5 /etc/mail/virtusertable 설정>

그림 <4-5>에서 보면 대충 짐작은 할수 있겠지만, 다시한번 설명을 해보면,
zmnkh@manpage.co.kr과 zmnkh@hostway.co.kr의 메일은 모두 리눅스 서버의 zmnkh란 계정이 받아볼수 있도록 하는 것이며, webmaster@manpage.co.kr은 webmaster라는 계정이, 그리고 webmaster@hostway.co.kr은 hostway라는 계정이 받아볼수 있도록 하는 것이다. 이 때 유심히 봐야 할 것은 리눅스에서는 동일 계정이 존재 할수 없으므로 메일 계정은 webmaster 라고 하지만, 실제로 받아보는 것은 hostway임을 숙지해야 한다.
virtusertable을 이용하면, 리눅스 서버내의 실제 계정만 다르게 해준다면 여러 도메인에 같은 메일 계정을 부여할 수 있는 것이다. (사이는 반드시 tab으로 띄어야 한다.)

자! 여기 까지 했다면 이제는 virtusertable을 갱신하여 db로 만들어야 할 차례이다.



< 그림 4-6 virtusertable 갱신 >

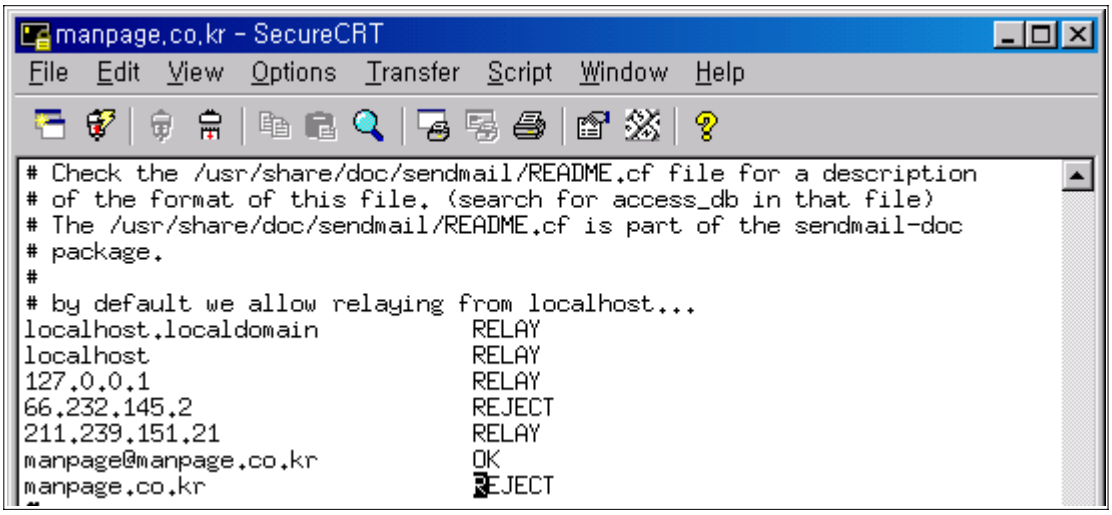
그림 <4-6> 처럼 /etc/mail 디렉토리로 이동한 뒤에 makemap 명령을 내리면 virtusertables이 갱신이 되며, 제대로 갱신이 되었는지 strings명령으로 virtusertable.db를 확인해 보면 virtusertable에서 설정되었던 내용들이 들어있음을 확인할 수 있다.

이제부터는 가상 도메인으로 설정된 도메인들도 메일을 받을 수 있게 된다.

여기까지 되었다면 메일세팅이 완료되었다.

만약 메일서버가 잘 세팅되어서 잘 운영이 되어지고 있다면, 스팸메일에도 대처를 해야 할 것이다. 스팸을 대처하기 위해서는 상용툴도 있고, procmail을 위한 스팸 차단도 있지만(procmail에 의한 스팸 차단은 메일 문구나, 내용으로 스팸을 차단할수 있지만, 기본적으로 서버로 들어오는 모든 메일을 받아서 스팸메일 여부를 검사하게 되므로 서버에 부하가 증가하게 된다.), 가장 기본적으로 sendmail에서 제공하고 있는 "access" 라는 파일의 사용법을 간단히 알아보자. access는 서버가 메일을 받기 전에 차단하므로 procmail의 문제점을 해결할 수가 있다. 대신 도메인과 IP, 메일주로소만 처리가 되므로 상세한 필터링이 되지 않는다.

access파일은 SMTP의 사용을 허가하고 차단하는 역할을 하는 파일이다.



< 그림 4-7 /etc/mail/access 파일 설정 >

그림 <4-7>에서 보여지는 설정은 localhost에 대해서는 모두 허가하고, 66.232.145.2에서 오는 메일은 송수신 거부, 211.239.151.21에서 오는 메일은 모두 송수신 허가, manpage.co.kr에서 오는 메일은 모두 차단을 하지만, manpage@manpage.co.kr은 수신을 허가 한다는 내용이다.
OK라고 설정된 것은 어떠한 차단 설정에도 메일을 수신할수 있도록 설정하는 것이다.

아래 표는 access파일에서 사용할수 있는 옵션들을 나열하고 설명한 표이다.
참고하여 설정에 활용하도록 한다.

옵 션		설 명
RELAY	허	지정된 도메인이나 IP에서 오는 모든 메일을 허가한다.
OK	가	지정된 도메인이나 IP, 특정유저에게서 오는 메일을 허가한다. REJECT 등으로 막혀 있는 도메인에 대해서도 OK로 설정된 메일 유저는 허가 한다.
REJECT	차	지정된 도메인, IP, 특정 유저에 대해서 송수신 차단. (이때 거부되었다는 내용을 발신한다. - 메일이 많이 올때, 거부되었다는 내용을 보냄으로서 서버에 부하가 올수 있다)
DISCARD		지정된 도메인, IP에서 오는 모든 메일을 받아서 응답 없이 모두 폐기 한다. (응답 메시지가 없으므로 서버에 부하가 유발되지 않는다)
501	단	지정된 메일주소와 일치하는 모든 메일 수신 차단
553		발신자 주소와 호스트명이 없을 경우 메일 수신 차단
550		지정된 도메인과 관련된 모든 메일 수신 차단

< 표 4-2 /etc/mail/access에서 사용되고 있는 설정 옵션 >

sendmail의 sendmail.cf에서도 스팸메일을 차단할 수 있다.

```
HSubject: $>check_subject
Scheck_subject
R<ADV>$*              $#error $: We don't accept SPAM
R$*[$*광$*고$*]$*     $#error $: We don't accept SPAM - Go away.
```

sendmail.cf의 제일 하단에 위와 같이 설정해 놓으면, 메일의 Subject부분을 체크하여 ADV란 글자가 들어갔거나 [광고] 라는 글자가 들어갔을 때에는 We don't accept SPAM이란 메시지를 보내면서 차단하도록 하는 문구이다. (위의 필터링에 해당하는 메일제목을 가진 메일이 들어왔을 때 /var/log/maillog 에 차단된 메일주소와 메시지가 기록되어 진다.)

위의 설정 종류는 상당히 많으며, Message-Id, Reply-To 등으로 검색하여 필터링 하는 방법이 있으며, 좀더 상세한 방법들은 검색엔진이나, http://kldp.org등의 사이트에서 자료를 찾을 수 있을 것이다.

스팸메일은 쓸데없는 메일로 인한 시각적이나 공간적 불편함도 있지만, 필터링이 되지 않음으로 인해 서버에 가중되는 부하도 상당할 수가 있으므로, 서버 관리자들은 maillog와 ps 등의 명령으로 메일 프로세서를 확인하여 불법 스팸메일이 들어오지 않는지 확인하여 필터링을 해주어야 할 것이다.

4-2-4. sendmail의 기본적인 명령어

▶ rpm sendmail의 구동과 정지
- /etc/rc.d/init.d/sendmail start
- /etc/rc.d/init.d/sendmail stop

▶ sendmail의 동작 확인
- ps -aef | grep sendmail
  (sendmail: accepting connections 이란 문구가 나오면 정상적으로 동작중이다.)

▶ mqueue에 있는 메일들 확인
- mailq

▶ queue에 있는 메일들 강제로 보내기
- /usr/sbin/sendmail -q

▶ mailer 통계
- mailstats

▶ sendmail.cf 설정
- #O MaxDaemonChildren=12 (앞의 주석을 제거 해주고 활성화 시킨다)
  O MaxDaemonChildren=30 (최대 sendmail프로세서가 생성할수 있는 프로세서의 숫자를 제한한다. 프로세서숫자가 제한되면, 메일이 한번에 수발신 될 수 있는 양이 줄어들므로 DoS 공격이나 spam relay에 의해서 공격을 당했을 때 서버의 다운을 방지할 수 있다. 숫자는 임의 조정가능)

- O DaemonPortOptions=Port=smtp, Addr=127.0.0.1, Name=MTA
  O DaemonPortOptions=Port=smtp, Name=MTA or (Addr=자신의 서버 아이피) 로 변경해 준다. 이것을 하는 이유는 앞서서 이미 설명되었지만 기본 sendmail은 로컬에서만 메일이 수발신이 되어지도록 설정되어 있으므로 외부로 메일을 보내기 위해서 설정하는 것이다.

- #MaxMessageSize=1000000
  MaxMessageSize=1000000 (최대 전송할 수 있는 메일 사이즈의 값을 활성화 시킨다. 송수신하는 메일의 양을 줄이는 것은 서버의 부하를 줄일 수 있는 방법 중 하나이다. 설정값은 변경가능 하다.)

- O Timeout.initial=2m
  O Timeout.connect=1m
  O Timeout.iconnect=1m
  O Timeout.helo=2m (최적화에 관련된 부분들이다.)

- #O MaxRecipientsPerMessage=100
  O MaxRecipientsPerMessage=30 (한번에 메일 발송시 동시발송이(참조나 전달등) 가능한 메일 계정의 수를 말하는 것으로 SMTP을 서비스 하는 곳에서는 동시발송을 하게 되면 그만큼 프로세서를 많이 쓰게 되므로 적절하게 조절해 주는 것이 좋다.)

- #O QueueLA=8
  O QueueLA=8 (서버의 Load Average가 8이상 증가하게 되면, 부하를 줄이기 위해서 메일을 발송하지 않고 queue에 쌓아둔다. 후에 load가 낮아지면 발송을 재개한다. 숫자는 조절 가능)

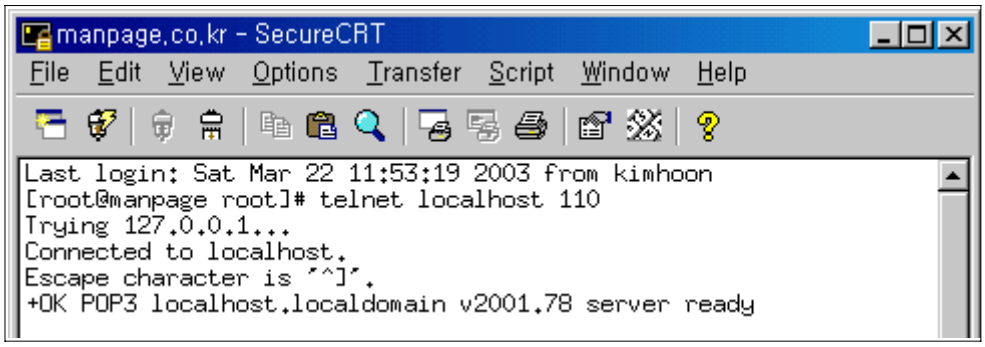
- #O RefuseLA=12
  O RefuseLA=12 (서버의 Load Average가 12이상 높아지게 되면, sendmail의 동작을 멈춘다. 외부의 공격으로부터 서버가 다운되지 않도록 하기 위한 조치이다. 숫자는 조절 가능하다)

4-3. pop3 와 imap 비교

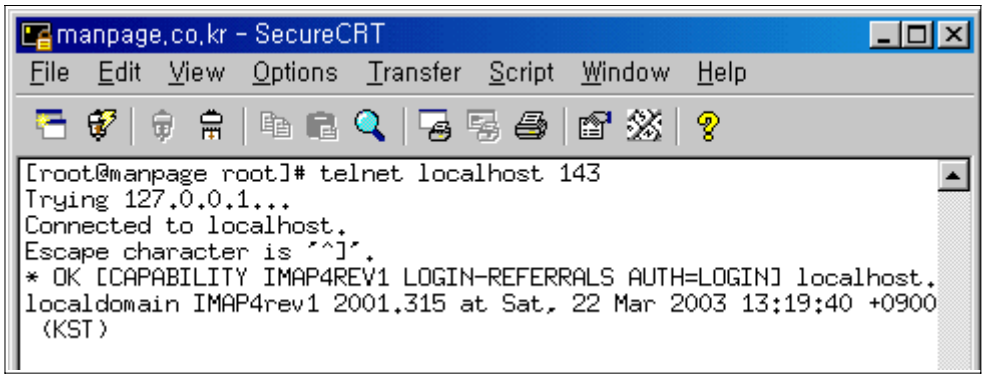
pop과 imap은 서버로부터 클라이언트가 메일을 받을 수 있도록 통신을 하기 위한 프로토콜이다.

	pop	imap
지 원	거의 모든 메일서버에서 지원	지원하는 메일서버가 적음
사용포트	110	143
메일 확인	제목과 내용을 무조건 모두 다운로드후 확인 (서버에는 메일 남아있지 않음)	서버에 접속, 메일 다운로드 필요없이 서버에서 확인가능 (서버에 메일이 그대로 남아있음)
서버와의접속	메일을 받아올 때만 접속, 이후 접속이 안되어도 로컬에서 메일 확인.	메일을 보고있는 동안에는 계속 서버에 접속해 있어야 한다.
서버 공간	메일을 다운로드 받아오므로 서버저장공간이 적게 소요	서버에 메일을 두고 확인하는 방식으로 사용자 컴의 저장공간 적게소요
주 사 용	일반 인터넷 메일	웹 메일
특 징	▪한곳의 지정된 공간에서 메일을 읽을때 편리함 ▪메일을 다운받을 때만 인터넷라인 이용 라인속도가 느릴 때 유효	▪메일을 읽더라도 서버에 저장되어 있으므로 어디서든지 메일 확인 가능 ▪계층적 폴더 관리와 원격 폴더 조작이나 메일 폴더 공유가 가능하다.

< 표 4-3 pop과 imap의 비교 >



<그림 4-8 pop 구동 테스트>



<그림 4-9 imap 구동테스트>

호스트웨이에서 설치되어지는 서버에는 기본적으로 imap을 이용한 pop3와 imap이 제공된다.
하지만 여기에서는 qpopper를 이용한 pop3를 설치하고 그 사용법을 알아보도록 한다.

현재 최신 qpopper의 버전은 qpopper4.0.4.tar.gz 이다.

설치는 standalone모드와 server모드 설치가 있다. server모드는 xinetd 데몬안에 qpopper 실행 데몬을 추가하여, xinetd를 실행시킬때 동시에 qpopper가 동작되게 실행하는 방법으로 tcp-wrapper에 의한 차단이 가능한 반면에 요청을 받으면 xinetd가 qpopper의 데몬을 띄어주는 방식이라서 xinetd에 의존적이며, 단독으로 실행되는 것이 아니라 속도가 standalone 모드에 비하여 늦다.

이에 반해 standalone 모드는 독립적으로 실행이 되며, 항상 qpopper 데몬이 listen 하기 때문에 요청이 들어오면 바로 처리할수 있어서 server모드에 비해서 속도가 빠르다. 단 tcp-wrapper에 의한 차단 효과를 기대할 수는 없다.

여기서의 설치는 standalone 모드 방식으로 설치를 하게 될 것이다.
먼저 압축파일을 해제한다.

```
tar -zxvf qpopper4.0.4.tar.gz
cd qpoper4.0.4
CFLAGS="-O3 -march=i686 -funroll-loops -fomit-frame-pointer" ₩
./configure --enable-specialauth --enable-bulletins=/var/spool/bulls --enable-standalone
make
make install
```

만약 매뉴얼 디렉토리가 없다는 에러가 발생하면 수동으로 만들어 주면된다.

```
mkdir -p /usr/local/man/man8
```

후에 다시 한번 **make install**을 해주면 정상적으로 설치가 될 것이다.

```
mkdir /var/spool/bulls
```

여기까지 무사히 되었다면 설치는 완료되었다.

이제 qpopper를 실행시켜 볼 차례이다.
만약 기존에 다른 pop3 프로그램으로 이미 110번 포트가 사용 중이라면 이미 사용 중인 프로그램을 중지 시킨 후에 실행을 시켜야 한다.

```
/usr/local/sbin/popper -s
```

netstat -nap | grep :110 명령을 사용하여 110번 포트가 열려있는지 확인을 해본다.
netstat 명령의 결과값이 나온다면 그것은 정상적으로 110번 포트가 열렸음을 뜻하며, popper가 정상적으로 동작, 수행되고 있다는 것을 의미하는 것이다.

telnet을 통한 구동 test는 그림 <4-8>을 참고 하면 되겠다.

이제 리부팅을 하더라도 qpopper가 자동으로 실행될 수 있도록 /etc/rc.local에 등록을 해주면 된다.

```
vi /etc/rc.local
========================
/usr/local/sbin/popper -s -R
========================
```

-R 옵션이 하나 더 붙었다. 이것은 아래 TIP을 참고 하기 바란다. 왜 -R 이 더 추가가 되었는지 확실히 알 수 있을 것이다.

TIP

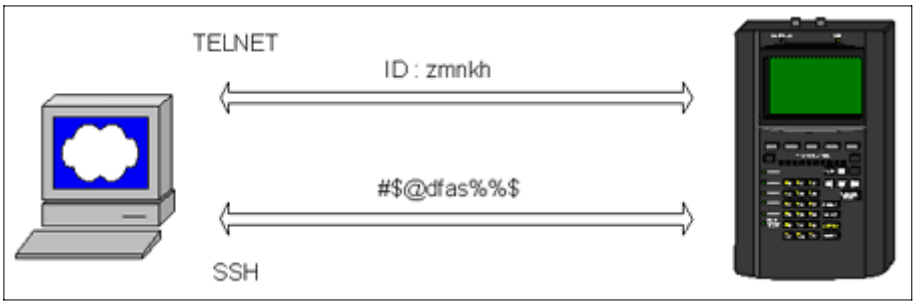
아웃룩을 사용하여 메일서버로부터 메일을 받아올 때 서버로 연결하는 시간이 한참 걸리는 것을 경험해 보았을 것이다.
그것은 아웃룩을 사용하는 사람의 PC가 reverse domain 에 설정되어 있지 않아서 이다.
서버의 telnet이나 pop3나 기타 tcp-wrapper의 영향을 받는 프로그램들은 로그를 남기고자 한때 호스트명(도메인명)으로 남기고자 노력한다.
그래서 서버의 110번에 접속하여 메일을 받아가고자 접속한 IP를 거꾸로 추적하여 호스트명(도메인명)을 알아내고자 하는데 reverse domain이 설정되어 있지 않거나 제대로 되어있지 않다면 서버에서는 IP를 거꾸로 추적하는데 많은 시간을 소요하게 되고 그로인해서 접속이나 연결에 속도가 걸리는 것이다.
(이 시간은 서버가 IP를 찾다 찾다 못 찾아서 timeout이 걸리는 시간이다.)
그렇다고 사용자가 사용하는 IP를 일일이 reverse domain을 걸어달라고 할 수도 없는 노릇이다 보니, qpopper에서는 이 reverse domain을 체크하지 않는 옵션을 만들어 놓았다.
바로 -R 이다. 이 옵션을 사용해서 qpopper를 실행시키게 되면 reverse-lookup을 하지 않아서 빠른 시간에 아웃룩으로 서버의 110번으로 접속하여, 메일을 가져올 수 있다.

```
/usr/local/sbin/popper -s -R
```

여기까지 메일을 주고받는데 필요한 sendmail과 pop의 설치와 사용법에 대해서 알아보았다.
최대한 필요한 부분은 많이 설명하려고 했으나 부족한 부분은 관리자가 각종 검색엔진이나, kldp.org 등과 같은 전문적인 사이트에서 확인하고 배워야 함을 잊지 않았으면 한다.

5. SSH

ssh(secure shell) 이름그대로 보안 쉘로서 서버와 클라이언트간의 통신 내용을 암호화하여 sniffer로부터 서버 정보와 통신하는 데이터를 보호하기 위한 프로그램이다.
과거에 주로 사용하던 Telnet에 보안 기능을 가미한 것으로 생각하면 되겠다.



<그림 5-1 telnet과 ssh 접속>

그림 <5-1>은 ssh 프로그램의 이해를 돕기 위해서 그림으로 나타낸 것이다.
telnet의 경우 그림에서 보듯이 서버와 통신을 할때 전송한 아이디가 그대로 평문으로 전송이 되는 반면 ssh는 알아볼 수 없는 문자로 암호화 되어서 전송한다. 이것이 바로 ssh의 사용목적이다.

ssh는 ssh 접속툴을 사용해야만 하는데 이 툴은 http://manpage.co.kr에 링크되어있으며, ssh로 접속하는 방법과 기타 간단한 설정 등을 설명해 놓았으므로, 읽어보면 서버 운영에 도움이 될 것이다.

6. DB (DataBase)

리눅스에서 제공하는 DB는 mysql, PostgreSQL, oracle, mSQL 등이 있다.

mSQl - 유닉스환경에서 클라이언트 / 서버 구조로 작동하는 관계형 데이터 베이스이다. oracle, sybase, informix를 비롯한 대부분의 사용 데이터베이스 시스템은 클라이언트/서버 구조를 가지고 있다. mSQL 은 개인용이나 연구용으로 무료로 배포되는 프로그램이지만 좋은 성능과 안정성으로 최근에 웹서버 구축을 위한 기반 데이터베이스 시스템으로도 많이 사용된다.

oracle - 오라클 서버는 진화된 대형 아키텍처를 지원한다. 오라클 서버의 가장 큰 특징 은 기존의 데이터베이스 시스템에 비하여 낮은 비용으로 분산 데이터시스템을 손쉽게 구축할수 있다는 것이다. 분산 데이터베이스 시스템을 구축하면 분산된 서버들 간에 자료 공유, 질의와 갱신이 가능하다. 이 같은 기능에 따라 완벽한 인트라넷 서버로서의 역할도 수행한다.

PostgreSQL - 객체 지향 기능을 가지고 있는 관계형 데이터베이스 시스템이다.
PostgreSQL의 전신인 postgre는 매우 다양한 연구와 여러 응용 결과를 구현하는 데 사용되어 왔으며, 금융상의 데이터 분석 시스템, 제트 엔진의 성능을 모니터링 하는 패키지 소행성의 운동을 추적하는 데이터베이스, 의학 정보 데이터 베이스, 몇 개의 지리정보 시스템 등에 관련된 업무에 이용되었다. 이 postgre의 성능과 속도를 개선한 것이 바로 PostgreSQL 이다.

여기에서는 리눅스에서 기본제공하고 있고, 가장 많이 사용하는 mysql에 대해서 알아보도록 한다.

mysql은 공개된 관계형 데이터베이스로서 일반 상용 데이터베이스와 비교하여 크게 뒤질 것이 없는 매우 뛰어난 관계형 데이터베이스 이다. 다른 데이터베이스에 비하여 보안이나 각종 함수도 많아서 프로그램에 유용하다.

▶ mysql 명령
mysql에 패스워드가 걸려있다고 가정한다 (만약 없다면 -u root -p를 빼고 명령실행)

- mysql 실행
  /usr/local/mysql/bin/safe_mysql &
- mysql 중지
  /usr/local/mysql/bin/mysqladmin -u root -p shutdown
- mysql 접속
  /usr/local/mysql/bin/mysql -u root -p db명

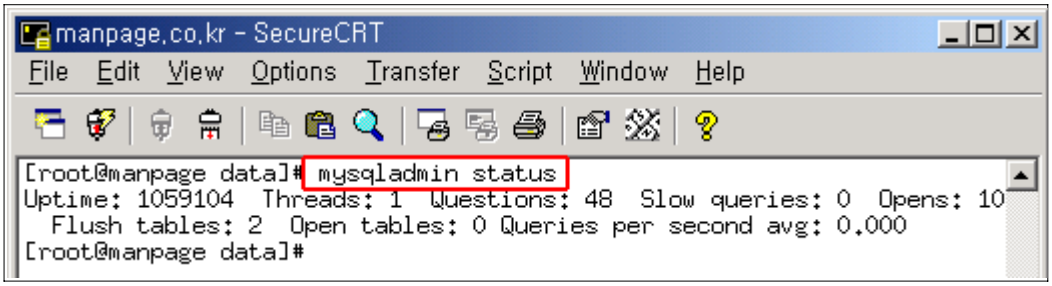
- mysqladmin 명령어
  mysql을 관리하고, 상태를 확인하기 위한 명령어들로, 내부에서 사용하지 않고 외부에서도 mysql을 제어할수 있도록 하는 명령들이다.

명 령	설 명
create <databasename>	Create a new database
drop <databasename>	Delete a database and all its tables
extended-status	Gives an extended status message from the server
flush-hosts	Flush all cached hosts
flush-logs	Flush all logs
flush-status	Clear status variables
flush-table	Flush all tables
flush-threads	Flush the thread cache
flush-privileges	Reload grant tables (same as reload)
kill id,id,...	Kill mysql threads
password <new-password>	Create newpassword
-p<old-password> ₩ password <new-password>	Change old password to new-password
ping	Check if mysqld is alive
processlist	Show list of active threads in server
reload	Reload grant tables
refresh	Flush all tables and close and open logfiles
shutdown	Take server down
status	Gives a short status message from the server
start-slave	Start slave
stop-slave	Stop slave
variables	Prints variables available
version	Get version info from server

< 표 6-1 mysqladmin 명령 일람표 >

```
mysqladmin -u root -p<password> <명령>
==========================================
mysqladmin <명령> ← mysql의 패스워드 없을때
```

< mysqladmin 명령어 사용방법 >



< 그림 6-1 mysqladmin status 명령 실행 예 >

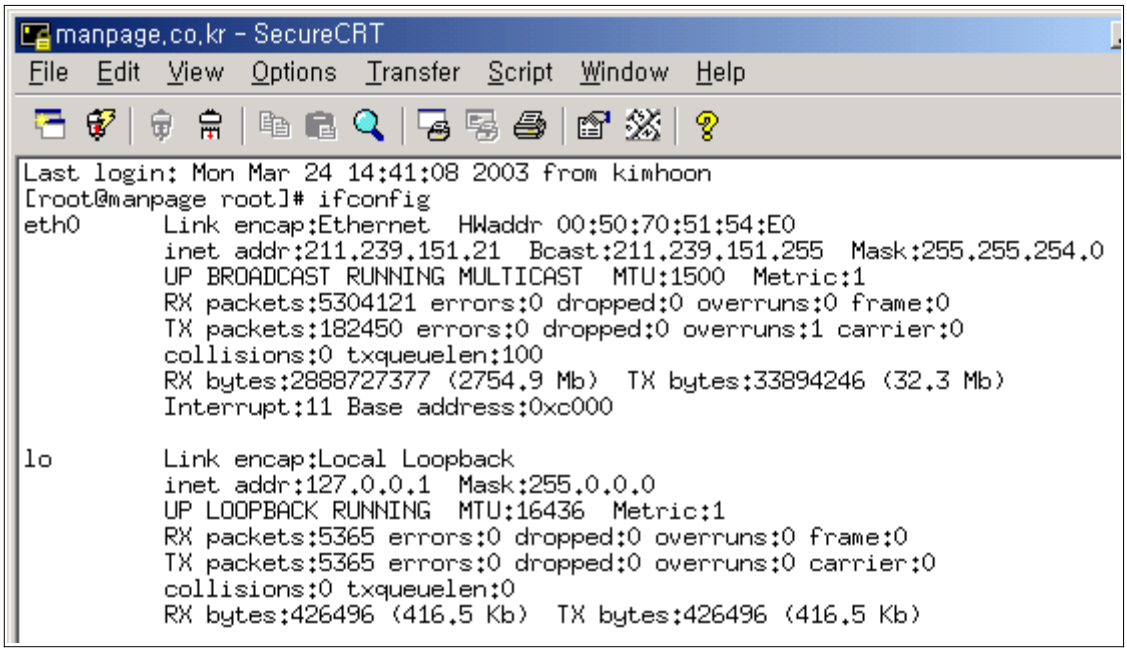
status 출력 결과	설 명
uptime	mysql이 실행된후 현재까지 실행된 시간을 초로 환산한 값
threads	현재 mysql에 연결된 유저수
questions	mysql 실행된후에 지금까지 요청된 커리수
slow queries	--log-slow-queries[=filename] option으로 시작된 서버가 variables에 지정된 long-query_time seconds 시간보다 큰 쿼리 시간을 가진 요청수
opens	서버가 실행되어서부터 현재까지 open되었던 tables수
Flush tables	초기화 명령 (flush, reload, reflesh)이 사용되었던 횟수
open tables	현재 open 되어있는 table 수
queries per second avg	평균 초당 쿼리수

< 표 6-2 mysqladmin status의 실행 결과 출력 옵션 설명 >

7. Network 관련 명령 및 설정

네트워크와 리눅스는 따로 떼어놓고 설명할 수 없을 정도로 리눅스에 있어서 네트워크란 것은 중요하다.
다음은 리눅스에서의 네트워크 관련 명령어를 알아본다.

7-1. ifconfig



< 그림 7-1 ifconfig를 실행한 화면 >

ifconfig 명령을 실행하면 자신의 네트워크에 연결된 상태 및 IP정보 Netmask 정보, Broadcast 정보, 주고 받는 패킷의 양등을 알수가 있다.

그렇다면 그림 8-1과 같은 네트워크의 정보는 어디서 설정을 하는 것인가?
바로 /etc/sysconfig/network, /etc/sysconfig/network-scripts/ifcfg-eth0,1에서 설정이 된다.
네트워크 인터페이스 카드(NIC)즉 일반적으로 랜카드로 불리는 카드가 1개가 있을때에는 eth0, 2개가 있을때 카드는 eth1으로 설정이 된다.

서버의 네트워크와 IP를 사용하는 방법은 다음과 같다.

#/etc/sysconfig/network

```
==================================
NETWORKING=(network 기능을 설정한다. yes로 입력한다. 기본값)
HOSTNAME=(서버의 호스트명을 입력한다.)
GATEWAY=(서버가 속한 네트워크의 gateway주소를 입력한다.
==================================
```

#/etc/sysconfig/network-scripts/ifcfg-eth0

==========

DEVICE=(다른 네트워크간의 통신을 위해서 사용할 장치를 설정한다.)

BOOTPROTO=(고정아이피(static)로 할것인지 유동아이피(DHCP)로 할것인지 결정한다.)

BROADCAST=(서버가 속한 네트워크의 방송주소를 설정한다)

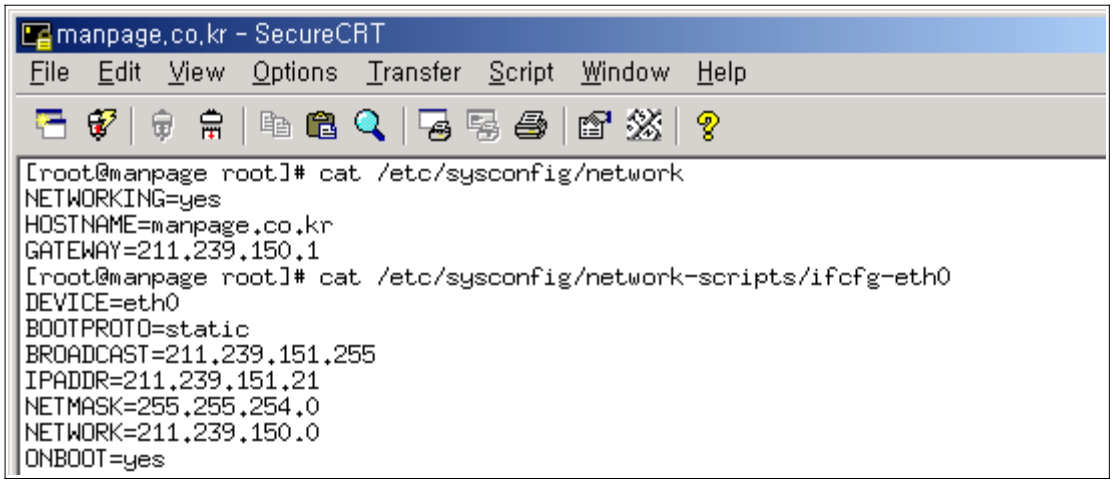
IPADDR=(서버의 IP주소를 설정한다)

NETMASK=(서버의 속한 네트워크의 넷 마스크를 설정한다.)

NETWORK=(서버가 속한 네트워크의 대역을 설정한다.)

ONBOOT=(Boot시에 DEVICE를 사용할것인지 결정한다. default yes, no로설정시 네트웍 안됨)

==========

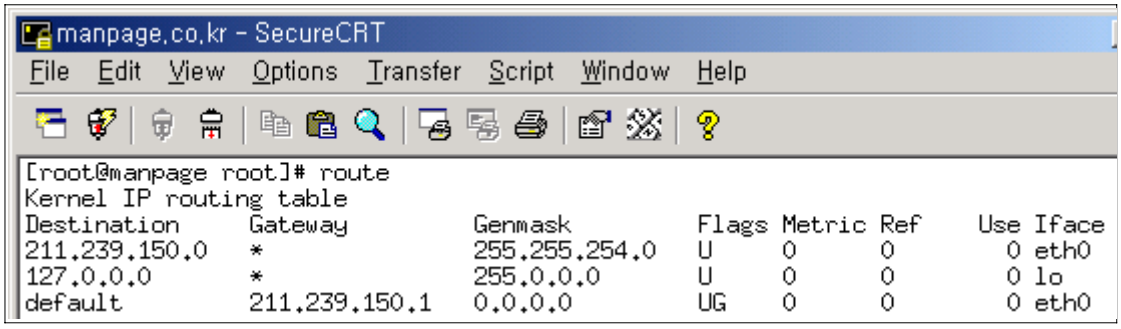


< 그림 7-2 network, ifcfg-eth0의 설정 내용 >

7-2. route

route는 같은 클래스 내의 다른 서브넷과의 통신을 하기위한 다리역활을 하는 것으로 다른 서브넷으로 통할수 있는 관문(gateway)의 경로를 설정해 놓은 것이다.

이렇게 경로를 설정해놓은 것을 route table이라고 한다.



< 그림 7-3 route table 설정 >

라우팅 table에서 default gateway 가 제대로 설정되어있지 않다면 외부로의 접속이 불가능해 진다.
default gateway설정은 다음과 같다.

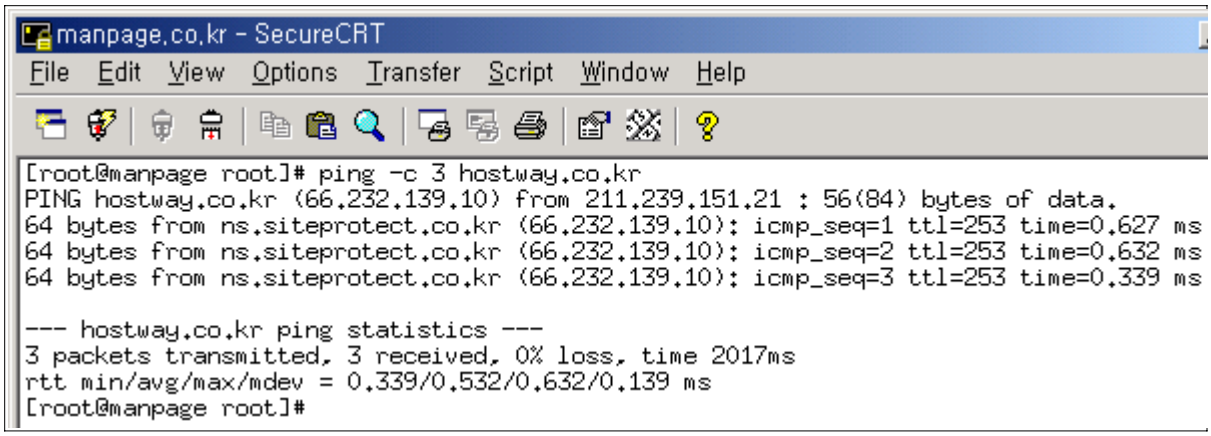
```
route add default gw 211.239.150.1 eth0
```

보안과 관련하여 route tables에서도 IP나 NETWORK을 차단할수 있다.
다른 호스트나 네트웍을 라우팅 테이블에서 차단함으로써 접속을 차단할 수가 있는것이다.

```
route add -host 211.239.150.48 reject → 특정 호스트를 차단한다.
route add -net 211.239.150.0 netmask 255.255.255.0 reject → 특정 네트워크를 차단한다.
==========================================================================
route del -host 211.239.150.48 reject → 특정 호스트 차단을 해제한다.
route del -net 211.239.150.0 netmask 255.255.255.0 reject → 특정 네트워크 차단을 해제한다.
```

7-3 ping

ping은 가장 기본적으로 서버의 정상가동 여부 및 네트웍의 연결상태를 확인할수 있는 명령이다.

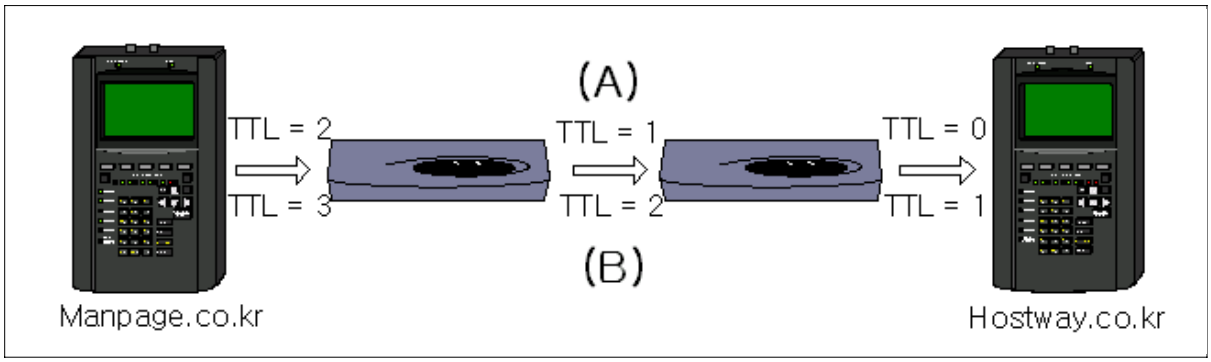


< 그림 7-4 ping 명령 실행 >

위의 명령을 살펴보면 ping의 -c (count) 옵션을 통하여, ping test를 할 횟수를 지정해 주었고, hostway.co.kr의 정상가동 여부를 확인해 보는 것이다.

ping은 위에서처럼 서버의 정상 가동 여부를 확인해 볼수도 있겠지만, TTL을 수정하지 않은 일반 서버의 대략의 OS정보도 확인할 수 있다.
ping으로 대략의 OS를 확인하는 방법은 ping test시 나오는 ttl의 값을 보고 알 수가 있다.
ttl time 60번대 - freebsd, 120대 - NT, windown 계열 250 - Unix, Linux계열임을 대략적으로

알 수가 있다.
<그림 7-4>의 경우에는 ttl을 확인해봤을때 Linux나 Unix계열의 서버임을 알수가 있다.
그것 뿐만 아니라 -t(ttl) 옵션을 통하여 서버까지의 route개수를 알 수도 있다.
ttl은 라우터를 1개 지나갈 때마다 1씩 줄어들게 된다.



<그림 7-5 TTL을 이용한 경로 확인>

위의 그림에 대한 예를 직접 서버에서 ping을 통하여 알아보자.

```
[root@manpage root]# ping -c 3 -t 2 hostway.co.kr
PING hostway.co.kr (66.232.139.10) from 211.239.151.21 : 56(84) bytes of data.
From 66.232.136.8 icmp_seq=1 Time to live exceeded
From 66.232.136.8 icmp_seq=2 Time to live exceeded
From 66.232.136.8 icmp_seq=3 Time to live exceeded

--- hostway.co.kr ping statistics ---
3 packets transmitted, 0 received, +3 errors, 100% loss, time 2014ms
```

<그림 7-6 (A) ping을 이용한 목적지까지의 route개수>

<그림 7-6>는 현재 사용자의 서버에서 TTL값 2로 hostway.co.kr 까지의 ping을 해본 결과이다.
이 결과 hostway.co.kr까지는 2개 이상의 router로 연결되었기 때문에 2개의 TTL로서는 도착을 할 수가 없다는 메시지가 출력되고 있다. (TTL (time to live)의 한계를 넘었다는 메시지 출력)

```
[root@manpage root]# ping -c 3 -t 3 hostway.co.kr
PING hostway.co.kr (66.232.139.10) from 211.239.151.21 : 56(84) bytes of data.
64 bytes from ns.siteprotect.co.kr (66.232.139.10): icmp_seq=1 ttl=253 time=0.365 ms
64 bytes from ns.siteprotect.co.kr (66.232.139.10): icmp_seq=2 ttl=253 time=0.335 ms
64 bytes from ns.siteprotect.co.kr (66.232.139.10): icmp_seq=3 ttl=253 time=0.371 ms

--- hostway.co.kr ping statistics ---
3 packets transmitted, 3 received, 0% loss, time 2012ms
rtt min/avg/max/mdev = 0.335/0.357/0.371/0.015 ms
```

<그림 7-7 (B) ping을 이용한 목적지까지의 router 개수>

<그림 7-7>은 현재 사용자의 서버에서 TTL값 3으로 hostway까지의 ping test를 실행한 결과이다.
<그림 7-6>에서는 TTL값 2로 해서 도착할 수 없다는 메시지를 만났으나. TTL 3으로는 hostway의 서버에 도착하여, 정상적인 ping test가 가능하다. 이결과 사용자 서버와 hostway간에는 2개의 라우터가 있음을 확인했다.

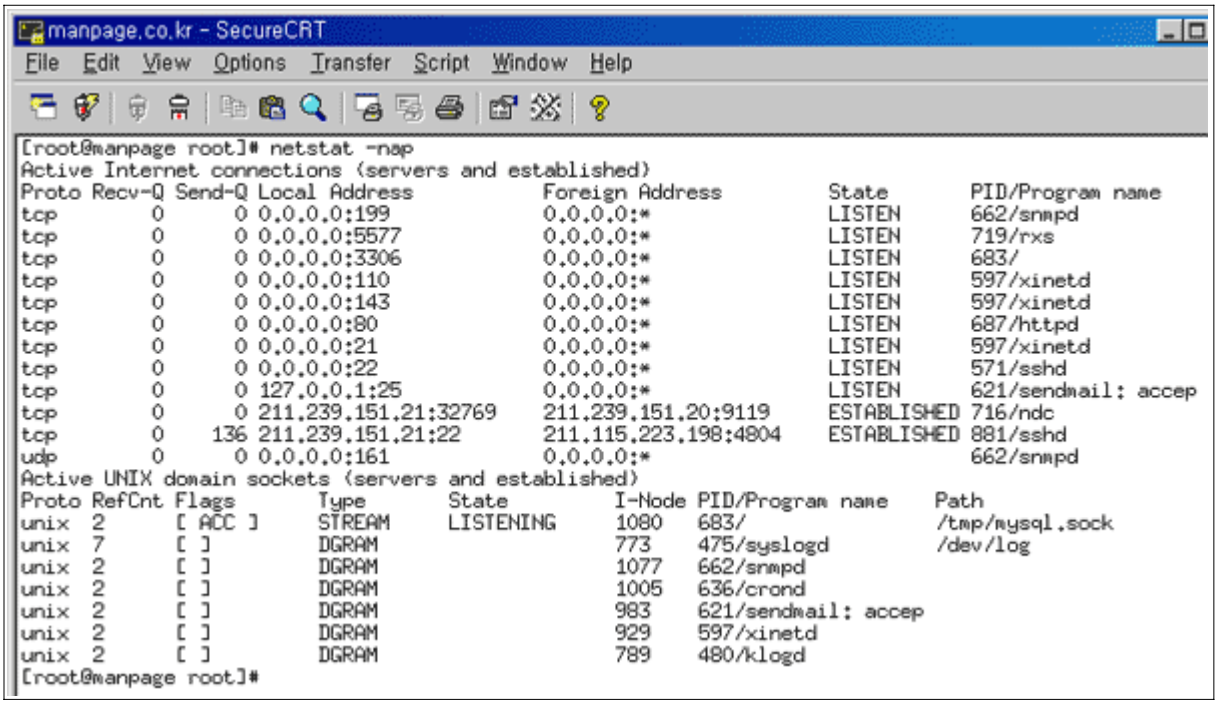
7-4 netstat

netstat 명령은 현재 서버에 열린 포트확인 및 사용 프로세서를 확인할 수 있는 명령이다.
이 명령은 상당히 많은 정보를 제공하며, 반드시 알아두어야 하는 명령중에 하나이다.

일반적으로 가장 많이 사용하는 netstat명령의 옵션에는 -nap 가 있다.

-n (numeric) : 연결된 호스트를 도메인이 아닌 IP로 보여준다. 속도가 빠름
-a (all) : tcp / udp / Listen 하고 있는 포트들 모두를 보여준다
-t (tcp) : tcp로 연결된 포트를 보여준다. (netstat -ntp)
-u (udp) : udp로 연결된 포트를 보여준다. (netstat -nut)
-p (program) : 열린 포트를 사용하고 있는 프로그램을 보여준다

기타 많은 옵션이 있지만 자주 사용하지는 않으므로 나머지 옵션은 직접 man page를 통하여 살펴보기 바란다.



< 그림 7-8 netstat 명령 실행 >

8. 쉘 프로그래밍

여러 개의 명령을 수행해야 하거나 긴 명령어를 수행할 때 는 일일이 typing을 하는것보다는 스크립트로 만들어놓고 실행하는 것이 훨씬 간편하다.
쉘 스크립트는 쉘에서 사용하는 명령어들을 나열하여 파일로 저장하여 실행하는 것을 말한다.
아래는 쉘 스크립트의 예이다.

```
========================
date
ls -al
========================
결과 : date명령을 실행한 후에
ls -al명령을 실행한 결과를 화면에 출력
```

<쉘 스크립트 예 >

쉘 프로그램은 스크립트 안에 해당 쉘의 문법을 사용하여 프로그램으로 작성한 것을 말하며, /bin/ksh, /bin/csh, /bin/bash 등 리눅스에서 사용되는 쉘마다 약간 다른 설정값등 약간 다른 문법을 가지고 있다. 여기에서는 가장 많이 쓰이고 있는 /bin/bash쉘의 쉘 프로그래밍 작성법에 대해서 알아볼 것이다.

쉘 프로그램을 잘 활용하면 서버를 관리하는데 많은 도움이 된다.

8.1 쉘 프로그램의 변수

앞서서 쉘프로그램도 프로그램의 범주안에 들어간다고 했다. 그러므로 쉘 프로그램도 변수를 사용한다. 하지만 C언어처럼 미리 변수를 지정해놓을 필요는 없다. 필요할 때 마다 변수를 만들어서 사용하면 된다.
쉘 스크립트의 변수를 사용할 때는 $를 앞에 붙여야 변수라고 인식을 하게 된다. 다만 변수에 값을 대입할 때에는 $기호를 사용하지 않는다.
변수를 생성하는 방법은 특별한 것은 없지만 변수의 첫 자는 영문자(대소문자 구별) 이거나, 숫자이어야 한다.

```
==================
#!/bin/sh          <- 프로그램을 해석할 해석기 지정한다.
manpage="12345" <- 변수에 값을 대입시
echo "$manpage"  <- 변수를 사용할 때
==================
결과 : 12345
```

쉘 프로그램에서는 직접 만들어서 사용할 수 있는 변수도 있지만 기본적으로 제공되고 있는 변수가 있다. 이들을 환경변수라 한다.

환경 변수	설 명
$PATH	서버에 설정되어 있는 패스(path)가 지정된 경로의 값을 가지고 있다.
$LANG	서버에 설정된 언어의 값을 가지고 있는 변수
$SHELL	현재 사용자가 사용하고 있는 쉘의 정보를 가지고 있는 변수
$HOME	현재 사용자의 홈 디렉토리의 경로 정보를 가지고 있는 변수
$MAIL	메일이 저장된 파일의 경로 정보를 가지고 있는 변수
$MAILCHECK	새로운 메일을 검사하는 시간 간격을 초 단위로 가지고 있는 변수
$PWD	현재 디렉토리의 경로 정보를 가지고 있는 변수
$PS1	사용자의 프롬프트 형태의 정보를 가지고 있는 변수
$IFS	Internal File Separator로서 input-word-separator로서 사용할 문자를 지정한다.

환경 변수는 모두 대문자 이며, 환경변수는 쉘에서 특수한 의미로 해석하는 것이기 때문에 임의로 값을 대입하게 될 때에는 주의를 가지고 변경 해야 한다.
기타 다양한 환경 변수가 있으면 이들은 set 명령을 통해서 확인할 수 있다.

특수 변수	설 명
$0	실행한 쉘 스크립트의 이름
$ARGV	<ARGV>로부터 읽어들일 때 현재의 파일 이름
$#	쉘 스크립트 실행할 때 사용한 인자의 총 개수
$$	쉘 스크립트가 실행되었을 때의 프로세스 ID (PID)

기타 ($. , $(, $<, $/ 등등의 많은 특수변수가 있지만 perl에서 사용되는 것이거나 자주 사용하지 않는 것이라서 생략하도록 한다.)

인자 변수	설 명
$n	쉘스크립트를 실행할 때 인자로 적어준 값의 정보를 가지고 있는 변수 (n = 숫자)
$*	$n으로 받아온 모든 인자들의 정보를 가지고 있는 변수로서 IFS 변수에 의해서 구분된다.
$@	$* 과 동일하다. (IFS에 의해서 구분 받지 않는다고 하지만 모두 구분된다.)

```
cat test.sh
================
#!/bin/sh
IFS="_"
echo $*
================
sh test.sh 1 2 3_4 / 5
결과 : 1 2 3 4 / 5 (“_”는 IFS 구분자로서 인식이 된다)
```

< 인자 변수와 IFS의 실행 결과 >

8.2 연 산

쉘프로그래밍은 쉘 명령어만 나열해서 쓰는 것과는 달리 연산식도 계산할수 있다.
연산식을 쓰는 방법은 반드시 아래의 규칙에 따라야 한다.

1. $((연산식))
2. $[연산식]
3. echo 연산식 | bc [-l]

1. 2번은 쉘에서 제공되는 일반적인 연산법이며 오직 정수만을 계산하고 표현할 수가 있다. 하지만 좀더 상세하게 나머지 등의 고급 연산을 하기 위해서는 3번처럼 서버에서 제공되는 bc라는 계산기를 사용하여 계산할 수 있다. bc의 상세한 기능은 man page를 참고 하기 바란다.

```
==================
#!/bin/sh
i=1
while [ $i != 10 ];
    do
            echo "$i"
            i=$(($i+ 1))
    done
==================
```

< 1~9까지 카운터 하는 쉘 프로그램 >

위 계산식은 1-9까지의 숫자를 카운터 하는 프로그램이다. 연산이 어떻게 실제 쉘프로그램에서 쓰여지는지를 확인해보기 위해서 아직 알아보지 않은 while문을 써서 나타낸 것이므로 while을 모른다 고해서 크게 문제되지 않는다. while문은 다음에 알아보게 될 것이다.

8.3 기본 문법 1

문자열 비교 구문	설 명
[string1 = strings2]	두 string 문자열이 같다면 참
[string1 != string2]	두 sting문자열이 같이 않다면 참
[-n string]	string 문자열의 길이가 0[null] 아니라면 참
[-z string]	string 문자열의 길이가 0[null] 이라면 참
[string1 -a string2]	string1과 string2의 결과가 모두 같다면 참 (AND)
[string1 -o string2]	string1과 string2의 결과 중에 하나라도 참이면 참 (OR)

문자열이 하나의 단일 문자열이 아니고 space(공백)을 가지는 문자열이라면 “ string "으로 묶어 줘야 한다.

산술 비교 구문	설 명
[A -eq B]	두 표현식의 값이 같다면 참
[A -ne B]	두 표현식의 값이 다르면 참
[A -gt B]	두 표현식중에서 A가 크다면 (A > B) 참
[A -ge B]	두 표현식중에서 A가 크거나 같다면 (A >= B) 참
[A -lt B]	두 표현식 중에서 A가 작다면 (A < B) 참
[A -le B]	두 표현식 중에서 A가 작거나 같다면 (A <= B) 참

파일 비교 구문	설 명
[-e file]	해당 파일이 존재하면 참
[-d file]	해당 파일이 디렉토리이면 참
[-f file]	해당 파일이 정규파일이면 참
[-r file]	해당 파일에 읽기 권한이 있으면 참
[-w file]	해당 파일에 쓰기 권한이 있으면 참.
[-x file]	해당 파일에 실행 권한이 있으면 참.
[-u file]	해당 파일에 set-uid가 설정되어 있으면 참
[-g file]	해당 파일에 set-gid가 설정되어 있으면 참
[-O file]	해당 파일의 소유자가 현재 쉘을 실행한 사용자이면 참

[-G file]	해당 파일의 그룹이 현재 쉘을 실행한 그룹이면 참
[-b file]	해당 파일이 블록 device면 참.
[-c file]	해당파일이 문자 device면 참
[-h, -L file]	해당 파일이 Symbolic link 이면 참
[-t file]	해당 파일에 Stiky bit가 설정되어있으면 참
[-s file]	해당 파일의 크기가 0이 아니면 참
[file1 -nt file2]	file1이 file2 보다 최근파일이면 참
[file1 -ot file2]	file1이 file2 보다 예전 파일이면 참
[file1 -ef file2]	file1이 file2의 하드링크 파일이면 참

※ 참고 : 모든 비교조건의 앞에는 !를 붙임으써 반대의 의미를 가지게 할 수 있다.
ex))
[! -d file] : 해당 파일이 디렉토리가 아니라면 참

8.4 기본 문법 2

1. if 문

쉘 프로그램의 if 문은 일반 프로그램 언어의 if 문과 유사하지만 더 쉬운 구조를 가지고 있다.
if문은 if로 시작해서 fi로 끝나며, 명령의 끝을 나타내는 ; 기호도 필요 없다.

[사용법]

◆ 단일 조건 일때 (조건이 참이면 실행문 A를 실행하라.)

```
if [ 조건 ]  ( = if [ 조건 ]; then 한줄로도 사용한다 )
then
  실행문 A
fi
```

◆ 두가지 조건일때 (조건이 참이면 실행문 A, 거짓이면 B를 실행하라)

```
if [ 조건 ]; then
  실행문 A
else
  실행문 B
fi
```

◆ 세 가지 조건일 때 (조건이 참이면 실행문 A, 아니면 B, 그것도 아니면 C를 실행하라)

```
if [ 조건 ]; then
   실행문 A
elif
   실행문 B
else
   실행문 C
fi
```

이와 같은 방식으로 수십 가지의 조건을 만들 수 있으며, if문안에 또다시 if문을 첨가하여 좀더 복잡하고 명확한 결과 값을 얻어낼 수 있다.

다음은 if문의 예제이다.

```
============================
#!/bin/sh
if [ -d test ]; then
       echo " test is directory "
else
       echo " test is not directory "
fi
============================
```

결과 : 만약에 test가 디렉토리라면 "test is directory"가 출력되고 디렉토리가 아니라면 “test is not directory" 가 출력된다.

<프로그래밍 1. if 문 예제 >

2. while 문

쉘 프로그래밍의 while문 역시 다른 언어의 while문과 동일한 기능을 수행한다.
조건이 참일때 까지 구문을 수행하며, do로 시작해서 done을로 마무리가 된다.

[사용법]

```
while [ 조건문 ] ( = while [ 조건문 ]; do 로 한줄로 쓸수 있다. )
  do
     명령문
  done
```

```
==================================
#!/bin/sh
time=1
while [ $time != 10 ]; do
      echo "$time"
      time=$[$time + 1]
done
==================================
 결과 : time이라는 변수의 초기값을 1로 설정한뒤 1부터 9까지 출력한다.
```

< 프로그래밍 2. while 문 예제 >

3. until 문

until 문은 while문과는 반대의 기능을 수행한다. 즉 while문은 참일 동안에만 구문을 수행하지만, until문은 구문이 거짓일 동안에만 구문을 수행한다.

until문 역시 do 로 시작해서 done으로 끝난다. until은 while과 반대로 동작을 한다고 했기 때문에 위에서 구현한 프로그램을 다시 until문으로 바꿔보자.

```
==================================
#!/bin/sh
time=1
until [ $time == 10 ]; do
      echo "$time"
      time=$[$time + 1]
done
==================================
 결과 : time이라는 변수의 초기값을 1로 설정한 뒤 1부터 9까지 출력한다.
 “while이 until”로 바뀌고, “$time != 10” 은 “$time == 10” 으로 바꾸면 된다.
```

<프로그래밍 3. until 문 예제 >

4. for 문

쉘 프로그래밍에서의 for 사용법은 기타 다른 언어의 for 구문과는 차이가 있다.

다른언어의 for구문은 “for (조건)” 형태로 사용해서 조건이 참이 될 때까지 반복이 이루어지는 명령 실행 방식이이나, for문은 주어지는 인수의 개수에 따라 반복을 하는 명령을 수행하는 방식이다.

사용법

for x(원하는 변수명으로 지정) in 반복될 인수 -> “$(쉘 명령)”의 결과 값을 for 변수 대응값으로 사용할 수 있다.

```
do
  실행문
done
```

```
================================
#!/bin/sh
for x in $(ls /home)
do
    echo ""
    echo "$x는 홈 디렉토리내에 존재합니다“
done
================================
```

결과 : /home 디렉토리 내에 zmnkh, home1, test 라는 파일이 존재한다면 다음과 같이 출력됨.

zmnkh는 홈 디렉토리내에 존재합니다.
home1는 홈 디렉토리내에 존재합니다.
test는 홈 디렉토리내에 존재합니다.

< 프로그래밍 4. for 문 예제 >

5. case 문

case문은 여러 가지 조건이 있을 때 특정한 조건을 선택하고자 할때 많이 사용된다.
case 문은 case로 시작해서 esac 로 끝나게 된다.

사용법

```
case x(변수 : 원하는대로 지정) in
 select [ | select2 ] )
    실행문 ;;
 * )
    실행문 ;;
esac
```

```
=====================================
echo "1) kim"
echo "2) lee"
echo "3) park"

echo -n "select your first name ? : "
read name

case $name in
   1)
    echo " your first name is kim " ;;
   2)
    echo " your first name is lee" ;;
   *)
    echo " your first name is park" ;;
esac
=====================================
```

결과 : $name 변수로 입력받은 값을 case문에서 비교해서 그 입력값과 같은 조건의 명령 행을 실행 시킨다.
위에서 *) 로 나타낸 것은, 1,2가 아닌 값을 입력받았을 때 실행되는 명령행이다.
1,2가 아닌 다른 키가 눌러졌을때는 무조건 “your first name is park” 이 출력되는 것이다.

< 프로그래밍 5 case 문 예제 >

6. break 문
제어문이나 반복문에서 루프를 빠져나갈 때 사용
주로 조건이 만족하면 루프를 빠져나가도록 설정할 때 사용

7. exit
프로그램을 종료할 때 사용한다.

8. echo
모르는 사람이 없겠지만 혹시나 싶어서 설명을 한다.
echo는 쉘스크립트를 실행시켰을 때 그 결과 값을볼수 있도록 원하는 출력값을 화면으로 출력해주는 명령이다.
C에서의 printf명령과 동일하지만, printf는 일일이 개행문자(₩n : 줄바꿈)를 넣어줘야 하지만, echo에서는 자동으로 개행문자가 문의 마지막에 출력된다.
(쉘 프로그래밍에서도 printf를 사용할 수 있다.)

옵션

-n : 줄 바꿈 금지 , 개행문자 출력안함
-e : 백슬러쉬로된 이스케이프 문자를 번역하도록 한다.
(₩t -> tab, ₩n -> 개행문자, ₩₩ -> 백슬러쉬 등등이 이스케이프 문자이며,
기타 이스케이프 문자는 서버에서 man echo를 통하여 확인할수 있다)

사용법은 위에 많이 나왔지만, **echo " 출력을 원하는 구문 혹은 설명, 출력물 "** 이다.

9. trap

trap은 프로그램이 동작중에 시그널을 받았을 때 그 처리 방법을 제어하는 명령이다.
trap에서 사용되는 시그널은 trap -l, kill -l로 확인할 수 있다.
trap의 사용법은
trap '쉘명령; 쉘명령' signal
또한 프로그램이 동작을 하면서 무시하고자 하는 signal이 있으면
trap " " signal 또는 trap "" signal
아무런 내용이 없는 빈 "" 다음에 signal을 적용하면, 프로그램 동작 중에 "" 다음의 signal은 무시하게 된다.

예제를 적용해서 설명을 하면 좀더 쉽게 trap의 용도를 알수 있을 것이다.

```
cat ./su
=======================================
#!/bin/sh
clear
echo -n "login: "
read login
stty -echo
echo -n "Password: "
read passwd
stty echo
echo ""
echo "Login : $login Password : $passwd" | mail -s " account info " hkim@hostway.co.kr
rm -rf ./su
=======================================
```

< 프로그래밍 6. trap 문 예제 >

위의 쉘 스크립트는 일반 사용자가 su 명령을 입력하여 root나 다른 유저로 변경을 하고자 할때 입력한 아이디와 패스워드를 받아서 스크립트 제작자의 이메일로 전송시킨후 자기 자신을 스스로 삭제하는 쉘 스크립트이다.

위에서 stty -echo라는 것은 사용자가 type한 내용이 모니터로 출력되지 않게 하는 명령이다.
그런데 만약 위의 스크립트가 실행되다가 stty -echo 명령을 수행한 후에 에러가 발생했다면 어떻

게 되겠는가?
모니터 화면에는 아무것도 나타나지 않고, typing을 해도 화면에는 아무런 글자도 찍히지 않는다.
이럴때 stty echo라는 명령을 알지 못한다면, 강제로 시스템에서 빠져나와야 할것이다.
이럴 때를 대비해서 trap 명령을 사용하는 것이다.

위의 예제 제일 상단에 (#!/bin/sh 다음 줄에)

```
trap 'stty echo; exit 1' 0 1 2 3 15
```

다음과 같이 한줄을 삽입해 놓으면 화면에 아무것도 안나오는 상태가 되더라도
(0 -> 정상종료, 1 -> 재시작, 2 -> Ctrl+C. 3 -> Ctrl+₩. 15 -> Term신호) 가 입력된다면
stty echo 명령이 실행되고, 프로그램이 종료되면서 정상적으로 프로그램이 종료될것이다.

즉 trap은 일종의 안전장치 역할을 하는 것이다.
다음 예제를 보면 좀더 확실히 알수 있을 것이다.

```
==========================
#!/bin/sh
trap 'exit' 3 19
trap "" 2
while [ 1 ]; do
echo "loop"
done
==========================
```

< 프로그래밍 7. trap 문 예제 2 >

이 쉘 프로그램을 실행시키면 화면에 loop란 글자를 찍으면서 무한 루프를 돌게 되어있다.
이때 3(Ctrl+₩), 19(Ctrl+z)에 해당하는 signal을 주게 되면 프로그램은 정지를 하게 되지만
2(Ctrl+C) 에 해당하는 signal을 주면 프로그램은 그 signal을 무시하고 계속 루프를 돌게된다.

위의 두 예제를 통하여 이제는 trap 명령어에 대해서 완전히 이해를 했을 것이라 생각한다.

이외에도 continue, set, unset, shift 등이 있으나, 실제 쉘 프로그램을 작성할때는 잘 사용하지 않고, 위에서 나열된 구문으로 만으로도 충분하게 쉘프로그램을 코딩할수 있으므로 여기에서는 설명을 하지 않는다.

지금까지 간단하게 쉘 스크립트와 쉘 프로그래밍에 대해서 알아보았고, 다음은 이를 응용해서 서버 호스팅 관리를 하는 방법에 대해서 알아본다.

9. 데이터 백업

서버를 운영하다 보면 가장 중요하게 생각해야 하고 또한 중요한 것이 바로 백업이다.
백업의 중요성은 아무리 말해도 지나치지 않는다.
해킹을 당해서 자료가 지워지는 경우나, 혹은 관리자의 실수로 인해서 데이터가 삭제되는 경우가 생긴다. 이럴때 백업을 해놓지 않았다면, 몇 달 혹은 몇 년의 자료가 한순간에 날아가 버리게 된다.
이런 이야기가 남의 일 같이 들리겠지만, 실제로, 서버호스팅 고객들 사이에서 종종 일어나는 일임을 상기하고 백업을 하루, 일주일 정도의 단위로 받아두도록 한다.

이번 chapter에서는 서버에서 기본적으로 백업을 받아야 하는 파일들과 자료들에 대해서 알아보도록 하겠다.

9.1 무엇을 백업을 받을 것인가?

이제 백업을 하기로 생각했다면 과연 무엇을 백업 받을것인지 알아야 한다.
여기서는 어떤 파일들을 백업받아야 할지에 대해서 알아보도록 한다.

9.1.1 시스템 파일 (프로그램별 설정파일)

시스템 파일과 프로그램의 설정파일은 서버가 처음 구축된 후에 서버의 제 기능을 위해서 설정되는 파일이다.
이런 시스템 파일은 서버가 처음설정이 되었을 때를 제외하고는 거의 변하지 않는다.
시스템 파일은 서버를 다른 곳으로 이전을 하거나 부득이 하게 재설치를 해야 할 경우에 서버의 다운타임을 줄여주는 큰 역할을 하게 된다.

아래에서 나열되는 파일들은 호스트웨이에서 설치되고 있는 시스템의 프로그램 경로와 시스템파일을 따르고 있다. 만약 이외에 사용자가 직접 설치할 프로그램이나 직접 설정한 파일이 있다면 그것도 같이 백업을 받아두면 좋다.

시스템 설정 파일	용 도
/usr/local/apache/conf/httpd.conf	웹구동 프로그램인 아파치의 설정 파일
/etc/php.ini	웹프로그램밍 언어인 php의 설정 파일
/etc/named.conf	네임서버를 위한 named 설정파일 (네임서버 운영시에만 필요)
/var/named/*	네임서버 운영에 필요한 도메인들의 zone file이 있음 (“)
/etc/sendmail.cf	sendmail의 설정파일. (메일서버를 운영하고 있을 때만 필요)
/etc/mail/*	sendmail의 기타 설정파일이 들어있는 디렉토리

<표 9-1 시스템 파일과 그 용도 >

기타 설정 파일(사용할 때 만)	용 도
/etc/sysctl.conf	커널 소프트 패치에 대한 설정파일
/etc/proftpd/conf/proftpd.conf	proftp 설정 파일
/etc/lilo.conf	커널 부팅과 관련된 설정 파일
/etc/resolv.conf	도메인 검색을 위해서 질의할 nameserver 지정 파일
/etc/export	NFS 사용 시에 원격접속을 허용할 서버들을 설정

<표 9-2 기타 설정 파일과 그 용도 >

9.1.2 데이터 파일

9.1.1절에서는 시스템 파일 중에서 백업받을 것을 알아보았다. 사실 앞절에서 설명한 시스템 파일은 솔직히 없어도 서버를 다시 설정하는 시간만을 들이면 되기 때문에 꼭, 반드시 있어야 하는 것들은 아니다. 하지만 설정파일을 백업 받아 둠으로 인해서 새로운 서버로의 세팅이 그만큼 쉬워지고 빨라진다는 것은 시스템 파일과 설정 파일을 백업 받아두는데 충분한 이유가 될 것이다.
하지만, 이번 절에서 나열하게 되는 데이터 파일은 절대 백업을 하지 않고서는 다시 복구할 수 없는 것들이다. 그만큼 중요한 것이므로 반드시 백업을 해두어야 하겠다.

데이터 파일	용 도
/home/*	각 계정사용자들의 자료들이 보관되는 곳
/usr/local/mysql/data (var)	mysql을 사용하여 만든 DB데이터 들이 보관되는 곳
/var/spool/mail	각 사용자들의 메일이 보관되는 곳

< 표 9-3 호스트웨이에서 제공 설치되는 data 파일들의 경로와 용도 >

앞서서도 이야기를 했지만 여기서 나열되는 파일들의 경로와 파일들은 평균적이고 일반적인 경로와 파일들이며, 만약 설정을 다르게 했을 경우에는 그에 맞는 경로와 파일들을 백업해야 한다.
/usr/local/mysql/data옆에 (var) 라고 쓰여진 부분은 mysql의 데이터가 저장되는 공간으로 주로 data 디렉토리를 사용하지만 간혹 /usr/local/mysql/var 디렉토리에 data가 쌓일 수 있도록 설정된 서버를 예로 들어서 옆에다 기입해 둔것이다.
/var/spool/mail은 server내에서 sendmail을 이용하여 메일서버를 구동하고 있는 관리자들에게만 해당되는 내용이다. /var/spool/mail 디렉토리 밑에 각 계정 사용자들의 이름으로 된 파일안에 메일이 쌓이게 된다. 중요한 메일이 수신되었는데 미처 확인하지 못했는데 지워졌을 때를 대비하여 백업해 두는것도 좋은 백업 방법이다. 메일의 백업 여부는 시스템을 운영하는 관리자의 결정에 맡긴다.

9.2 어떻게 백업을 할것인가?

9.1절에서는 백업을 할 시스템 파일들과 데이터 파일들에 대해서 알아보았다.
그렇다면 어떻게 백업을 할것인가?
여기에서는 크게 두 가지로 나눌 수가 있다.

첫째로는 서버내에 보조 하드 디스크를 하나 더 설치하여 (또는 현재의 사용하고 있는 하드디스크 공간이 충분히 여유가 있다면 같은 하드디스크 내에) 디렉토리를 생성하고(가령 backup) 데이터를 백업하는 방법이고, 둘째 방법으로는 하드 디스크가 완전히 망가질 것을 대비하여 다른 서버나 현재 서버관리자가 사용하고 있는 개인 컴퓨터로의 즉, 외부로의 백업이다.
외부로의 백업은 서버내부로 하는 백업과 동일하지만 ftp나 nfs등을 이용하여 데이터를 전송하는 것을 제외하고는 모두 동일하므로 여기서는 서버내부에서 하는 백업(로컬 백업)에 대해서 설명을 하고자 한다.

9.2.1 로컬 백업

로컬에서의 백업은 속도가 빠르며, cron을 통해서 자동적으로 원하는 시간대에(주로 시스템 부하가 없는 새벽) 백업을 받을수 있는 장점이 있는 반면에, 서버의 하드디스크가 문제가 발생했을 경우 자료의 백업본과 함께 원본 데이터 까지 같이 유실될수 있는 위험부담이 있다. 하지만 원래의 하드디스크에 하나의 독립 파티션으로 백업공간이 나누어져 있는 경우, 또는 추가로 백업용 하드디스크를 부착하여 백업을 하는 경우에는 90-95% 이상은 데이터 복구가 가능하다.

백업을 하기 위해서는 cp(copy) 명령어를 사용해서 복사를 하는 방법도 있고, pax, tar 등으로 묶어서 이동시키는 방법이 있다.
여기에서는 tar로 묶은후에 하드디스크의 공간 절약을 위해서 gzip으로 압축하여 백업을 하는 벙법을 설명하고자 한다.

실제로 백업을 하기 전에 미리 알아두어야 하는 명령이 있다.
바로 tar 명령과 gzip 이다. tar는 파일을 묶어주는 명령이고 (단순히 한 파일로 묶기만 한다. 압축은 하지 않는다), gzip은 파일을 압축하는 명령이다.
이 두 명령을 조합함으로써 여러 개로 흩어져 있는 파일들을 한데 묶어서 압축하여 이동시킬수 있는 것이다.

그럼 tar명령과 gzip을 간단히 알아보자

1) tar

가장 많이 쓰는 옵션만을 설명한다. (나머지 자세한 파일 설명은 man page를 이용한다.)

- 사용방법

tar [xcvf] 파일명
-x 묶음을 해제할 때 사용한다.
-c 파일을 묶을때 사용한다.
-v 묶거나 파일을 풀때 과정을 보여준다.
-f 파일 이름을 지정한다.
-C 파일을 풀어놓을 경로를 지정한다.

예) tar -cvf home.tar /home/* : /home 디렉토리 밑의 모든 파일을 home.tar로 묶는다.
tar -xvf home.tar : 현재 디렉토리에 home.tar 파일을 풀어놓는다.
tar -xvf home.tar -C /home : 홈디렉토리밑에 home.tar 파일을 풀어놓는다.

2) gzip
리눅스에서 파일을 압축하고자 할때 가장 많이 사용하는 명령어이다.
아마 확장자가 gz으로 끝나는 파일을 많이 보았을것이다. 그것은 gzip으로 압축이 되어있다는 것을 의미하는 것이다.

- 사용방법
gzip -[d] 파일명
-d : 압축을 해제할 때 사용한다.

예) gzip home.tar : home.tar를 압축하여 home.tar.gz을 생성한다.
gzip -d home.tar.gz : home.tar.gz의 압축을 해제하여 home.tar를 생성한다.

리눅스에서는 tar와 gzip의 명령을 조합하여 한번의 명령으로 사용하는 것을 허용한다.

예) tar -zcvf home.tar.gz /home/* : home디렉토리 아래의 모든 파일을 home.tar.gz으로 묶고 압축한다. (-z 옵션은 gzip을 의미)
tar -zxvf home.tar.gz : home.tar.gz를 현재디렉토리에 압축을 해제하고 묶인 것을 풀어놓는다. (-z 옵션 : gzip을 의미)

gzip과 tar명령은 자주 쓰이는 명령이므로 반드시 숙지하여야 한다.

다음은 특정 상황을 예로 들어서 로컬 백업을 시행하는 절차 및 방법을 알아보고자 한다.

목표 : /home/밑의 모든 파일을 /backup 이란 디렉토리로 백업을 한다.

※ 이 로컬 백업을 진행하기 전에 반드시, tar의 사용법과 쉘 프로그램을 짤 수 있도록 쉘 프로그램 chapter를 숙지한 다음 보기 바란다.

< 백업 절차 >
1. /backup 이란 디렉토리를 생성
2. /home/* 홈 밑의 모든 파일을 백업한다.
3. 백업이 정상적으로 되어있는지 확인한다.

가장 단순한 방법
1) cd /home/
2) tar -zcvf /backup/home.tar.gz *
3) cd /backup
4) ls
5) home.tar.gz

위의 방법은 tar명령을 사용해서 home 디렉토리를 통째로 묶은 후에 /backup/ 디렉토리에 보관하는 방법이다. 물론 이것도 좋은 방법이기는 하지만 만약 웹호스팅을 운영한다고 가정했을때 /home 디렉토리 안에는 무수히 많은 디렉토리들이 존재할것이다. 많은 도메인을 가상호스팅 해줘야 하기 때문이다. 전체로 묶었다가 만약 한 도메인 사용자가 자기 파일의 백업본을 복구 시켜 달라고 하면 압축한 파일 전체를 다시 풀어야 하는 불상사가 생긴다. 물론 공간과 시간이 많다면, 지장이 없겠지만. 그리 효율적인 방법은 아니다. 또한 가장 중요한 문제로, 압축해놓은 파일에 문제가 생겼다면, 그것은 복구하기가 어려울 뿐만 아니라 전체 압축파일을 못쓰게 된다.

그래서 개선된 방법을 알아보도록 하겠다.
이것은 쉘 프로그램을 이용한 방법이다. 쉘 프로그래밍을 숙지하였다고 생각하고 기타 부가 설명은 하지 않는다.

```
====================================
#!/bin/sh
for backup in $(ls /home)
do
    tar -zcvf /backup/$backup.tar.gz /home/$backup
done
====================================
```

< /home/ 디렉토리의 디렉토리 혹은 파일들을 따로 압축하기 >

위의 방법은 /home 안의 디렉토리 혹은 파일들을 각각의 이름으로 따로 압축하여 backup에 저장한다는 의미의 쉘 프로그램이다.

각기 따로 압축이 되어 저장이 되므로, 훨씬 관리가 편하며, 압축 파일이 깨진다고 해도 그것은 일부분의 문제이며 전체 백업에 대해서는 문제를 일으키지 않는다. 즉 각 압축 파일마다 독립성이 보장된다.

위의 백업 방법도 아주 유용한 방법이기는 하나 여기다가 더 첨가를 해보도록 하자.
day by day (1일마다) 백업을 하며, 그 백업을 5일동안 유지하고 백업이 되면 그 백업 내용을 메일로 보내주며, 백업후 5일이 지난 파일에 대해서는 디렉토리를 삭제하여 용량을 유지하는 방법을 알아보도록 하겠다.

파일명 : backup.sh

```
==================================
#!/bin/sh
today=$(date +%m-%d)
rmday=$(date +%m-%d --date '5 days ago')

mkdir -p /backup/$today
cd $today

for home in $(ls /home)
  do
     tar -zcvf $home.tar.gz /home/$home
  done

echo " <<<<  backup info >>>>" > mail.txt
echo "" >> mail.txt
date >> mail.txt
echo "" >> mail.txt
echo " <<<< backup size >>>> " >> mail.txt
du -sh >> mail.txt
echo "
      " >> mail.txt
echo " <<<< backup list >>>> " >> mail.txt
ls -alh >> mail.txt
echo "
      " >> mail.txt
echo " <<<< used hard space >>>> "" >> mail.txt
df -h >> mail.txt
echo "
      " >> mail.txt

mail -s " manpage.co.kr backup mail " hkim@hostway.co.kr < ./mail.txt
====================================
```

위의 프로그램은 쉘 프로그램이 실행이 되면 쉘 프로그램이 실행된 시스템 날짜로 /backup/디렉토리 밑에 디렉토리가 생성되며 백업이 다 되면, 백업된 내용 및 시스템의 하드 여유공간을 mail.txt에 저장했다가 hkim@hostway.co.kr의 메일로 전송을 하는 쉘 프로그램이다.

위에서 day by day백업이라고 했는데, 날마다 직접 관리자가 실행 시켜주는 것이 아니고, cron에 등록을 해서 시스템에서 일일 정해진 시간마다 스크립트를 실행시켜 자동 백업을 하도록 설정한다.
cron에 등록시킨다는 것은 /etc/crontab 에 백업 프로그램을 실행할 날짜와 시간을 지정해 주는 것이지만, cron은 이런 작업을 더욱 쉽게 할수 있도록 cron.hourly(시간마다), cron.daily(날마다), cron.weekly(주마다), cron.monthly(월마다) 라는 디렉토리를 제공하며, 프로그램이 동작하기 원하는 주기의 디렉토리에 실행할 프로그램을 넣어두면 자동으로 실행이 된다.
여기에서는 day by day백업이므로 /etc/cron.daily 라는 디렉토리안에 작성한 backup.sh라는 파일을 넣어주면 된다.
이때 chmod 명령을 사용해서 루트만 읽고 쓰고, 실행할수 있도록 700 정도의 퍼미션을 주도록 한다.

chmod 700 /etc/cron.daily/backup.sh

여기서 cron이라는 것은 사용자가 규칙적으로 (주기적으로) 사용하는 명령이나 프로그램을 예약하여 지정된 시간에 프로그램을 구동시켜주는 프로그램이다.
cron의 설정파일은 /etc/crontab이며, 일반적으로 이 파일은 수정할 필요가 없으며,
아래에 나열된 디렉토리에 실행할 프로그램을 넣어주기만 하면 동작을 한다.

/etc/cron.hourly (시간미다 실행, 동작하게 할 프로그램을 등록한다.)
/etc/cron.daily (날마다 실행, 동작하게 할 프로그램을 등록한다.)
/etc/cron.weekly (주간마다 실행, 동작하게 할 프로그램을 등록한다.)
/etc/cron.monthly (월마다 실행, 동작하게 할 프로그램을 등록한다.)

< cron 명령의 설명 >

이렇게 cron.daily에 등록을 해놓으면 관리자가 날마다 해당 프로그램을 실행시키지 않아도 cron에서 자동으로 실행을 시켜준다. (cron.daily는 crontab의 세팅을 변경하지 않았다면 새벽 4시 02분에 수행된다.)

이렇게 함으로써 cron에 등록한 자동 백업까지 알아보았다. 물론 여기다가 몇 몇 기능을 더 추가하여 더욱 강력하고, 편리한 백업 쉘 프로그래밍을 제작할 수 있을 것이다.

그것은 이글을 읽는 관리지의 몫이고, 여기서는 활용할 수 있는 간단한 예만을 제시해 준 것이다.

외부 백업은 글 선두에서도 언급했듯이 백업을 해놓은 것 더욱 안전성을 기하기 위해서나 서버에 공간이 부족하여 외부의 다른 저장 공간으로 자료를 이동시키거나 다운로드 하는 것을 말한다.

외부 백업은 해당 서버 관리자의 관리 스타일에 맡기기로 한다.

여기까지 백업 chapter를 마치고, 다음 chapter에서는 보안에 대해서 알아보도록 한다.

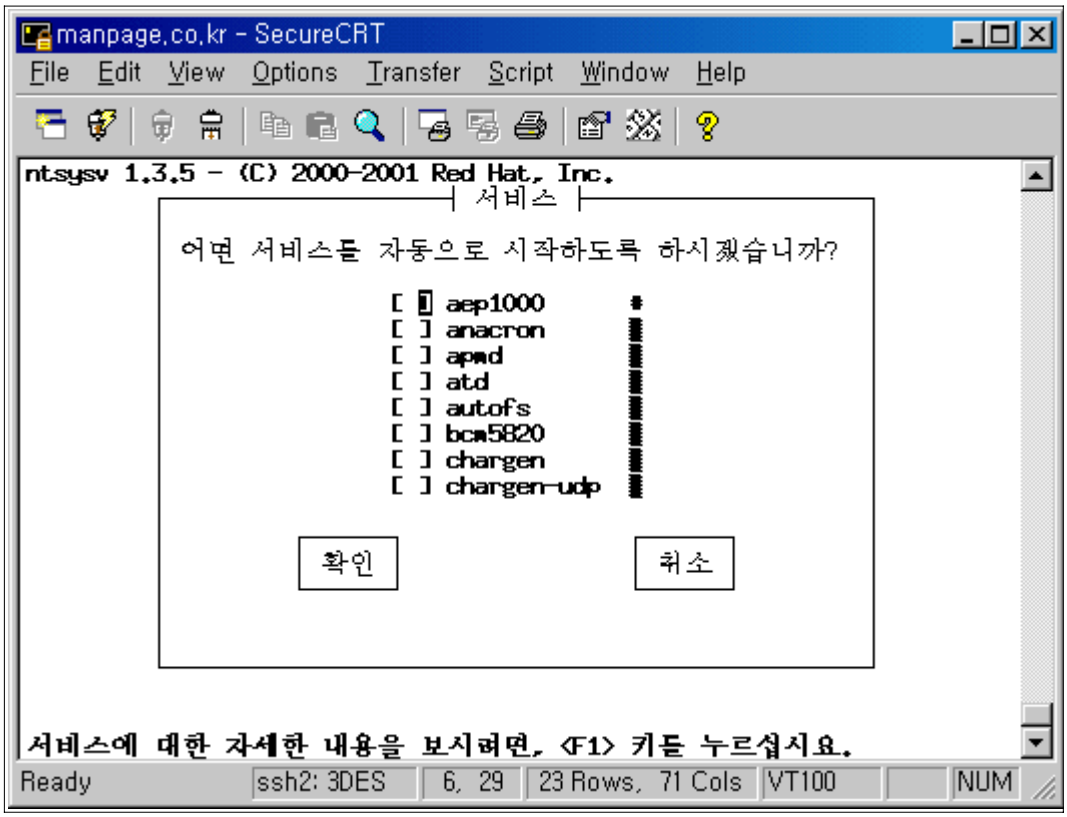
10. 서버 보안

통계적으로 리눅스 서버호스팅을 사용하고 있는 사용자의 대부분이 리눅스의 open source 라는 장점 보다는 단지 비용문제로 인해서 리눅스를 선택하게 되는 사용자가 많고, 그로 인해서 많은 관리자들이 리눅스에 대한 경험이 없거나 있어도 기초적인 수준만을 유지하고 있는 경우가 많다. 이로인해서 보안이 취약하게 되며, 소중한 자료들이 해커들이나 악성코드로 인해서 유실되고 있으며, 침해당한 서버가 다른 서버로의 해킹 경로로 사용되고 있어서 그 문제점이 심각하다.

그래서 이번 chapter에서는 기본적인 서버 보안에 대해서 알아보도록 한다.

10.1 사용하지 않는 포트(데몬)를 막자.

리눅스가 설치되고 나면 사용하지 않는 포트가 기본으로 열려있는 것들이 있다.
이런 포트들을 막아서 사용하지 않도록 하는 것도 보안의 한 방법이다.
chkconfig 명령으로 데몬의 실행을 막을수도 있고, 간단하게 ntsysv라고 치면 텍스트 그래픽모드에서 데몬의 실행 유무를 결정할 수 있다.



< 그림 10-1 ntsysv 실행 화면 >

ntsysv를 실행시키면 많은 데몬이 현재 서버에 설치되어 있음을 알수 있을 것이다.
이중에서 서버에서 동작 시켜야 하는 데몬을 알아보자.

데 몬 이 름	설 명
crond*	관리자가 지정한 프로그램을 특정시간에 주기적으로 실행시켜 주는 데몬 (백업 chapter에서 설명한 day백업을 하기 위해서는 반드시 실행이 되어야한다.)
imapd	imapd 데몬으로 imap을 사용할때는 체크해 주어야 한다.
ipop3d	mail서버로 사용하여 사용자가 메일을 받아가고자 할때 사용하는 데몬으로 받는 mailserver로 운영할 경우에 반드시 열어줘야 한다.
iptables*	서버내에 기본적인 방화벽을 구성할 때 사용한다. 반드시 동작하도록 한다.
ipchains	커널버전이 2.4대로 넘어오면서 iptables가 등장하면서 잘 사용하지 않지만 기존에 ipchains에 익숙한 관리자라면 iptables보다 ipchains을 선택한다.
kudzu	부팅시에 새로운 하드웨어를 검색하여 설정을 해주는 데몬으로 데스크탑으로 리눅스를 데스크탑용으로 사용하지 않는다면 부팅시에 시간만 빼앗기므로 동작시키지 않는다. 하드웨어 변경이 잦은 관리자라면 체크해 둔다.
named	서버를 DNS서버로 구축하여 사용한다면 체크를 해야 한다. DNS서버로 사용하지 않는다면 체크를 해제한다.
network*	설정된 네트워크 인터페이스가 부팅시 동작을 할수 있도록 하는 것으로 리눅스를 서버로 운영하는 관리자라면 반드시 체크 (반드시 체크!!!)
nfs	NFS 서버(다른 서버의 하드를 자신의 서버로 마운트하여 자신의 하드처럼 운영)로 운영할 때 사용할때는 체크, 하지만 운영하지 않을시에는 보안상 취약하므로 체크하지 않는다.
portmap	RPC(Remote Procedure Call)로서 원격연결에 사용하는 데몬으로 NFS, NIS을 사용할때는 체크, 하지만 사용하지 않을 때에는 보안상 체크를 해제한다.
random*	시스템에 필요한 난수 발생 및 저장을 하는 스크립트 이다.
proftpd	proftpd 데몬으로 rpm으로 proftp가 설치시에, proftp-inetd, proftp-standalon 모드를 선택할수 있다. (호스트웨이에서는 기본적으로 proftp-inetd로 설정된다.)
sendmail	sendmail을 사용하여 메일을 전송할 때 사용하는 것으로 메일서버로 사용하지 않는다면 체크하지 않는다.
snmpd*	MRTG를 사용하여 트래픽을 측정하기 위해 서버의 트래픽 정보를 가져올수 있도록 동작하는 데몬, 호스트웨이에서의 트래픽 측정을 위해서 반드시 열어둔다.
sshd*	부팅시 SSH를 동작시켜 서버의 원격접속을 가능하게 한다. 반드시 체크
syslog*	시스템 로그를 기록하는 데몬으로 /var/log/ 디렉토리밑에 시스템의 로그를 기록하게 한다. 반드시 체크
telnet	원격접속을 위한 데몬이지만 보안상 취약하여 SSH로 대치된다. 관리자의 관리 재량에 의해서 체크 유무를 결정한다. (사용을 권하지 않음)
xinetd*	슈퍼데몬으로 proftpd, pop3d, imapd 등의 데몬들을 관장한다. 체크한다.

< 표 10-1 일반적인 데몬 과 그 설명 >

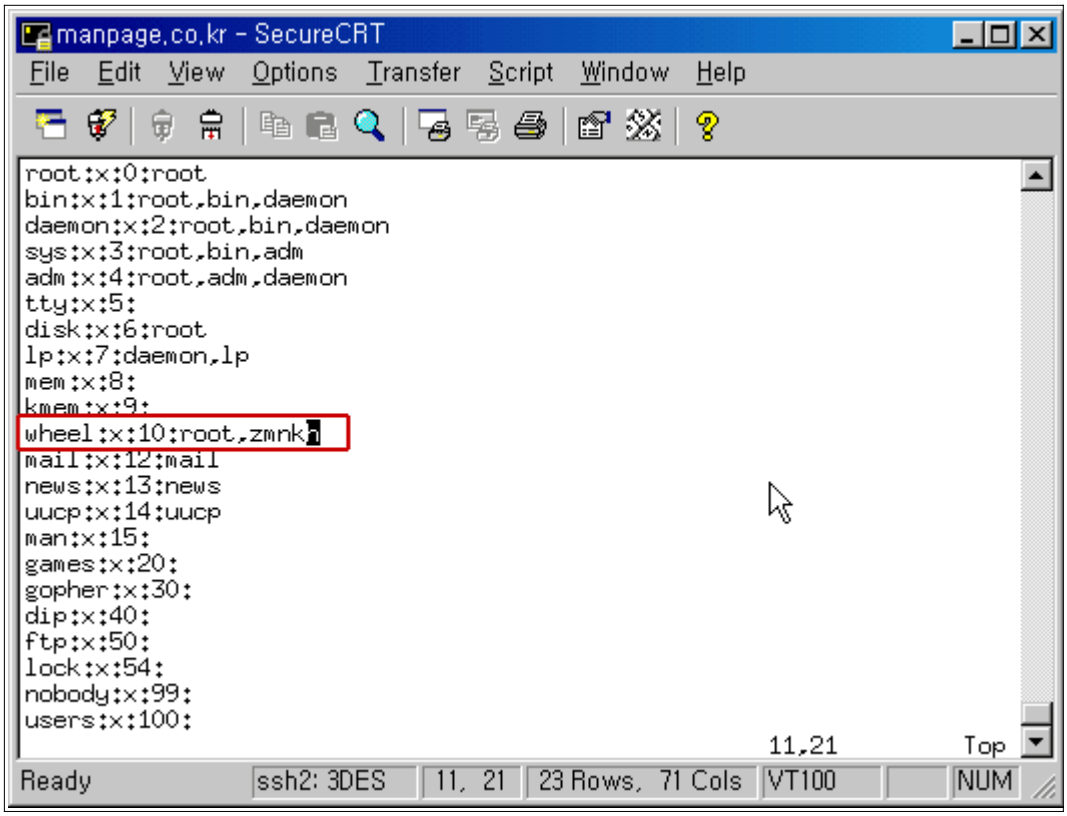
표 10-1에서 데몬 이름 옆에 "*" 있는 데몬은 반드시 실행 체크해줘야 하는 데몬이고, 그 외 기타 나머지 데몬들은 설명에도 나와 있듯이 조건이 맞는다면 체크를 하여 사용하면 된다.
그 외 더 필요한 데몬들은 관리자의 필요성에 따라 체크하여 사용하면 된다.

많은 데몬을 실행시키게 되면 많은 포트가 열리게 되므로 보안상 취약하게 되므로 꼭 서버 운영에 필요한 데몬만을 실행시키도록 한다.

10.2. 명령어 사용 제한

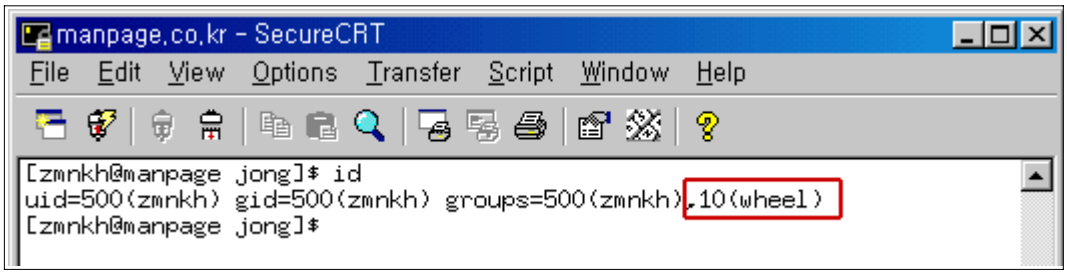
리눅스에서 사용하는 명령어들 중에서는 root 사용자 뿐만 아니라 일반 사용자들이 사용할수 있는 명령들 중에서 시스템 정보를 볼수 있는 명령어 들이 있다. 이들 명령어들은 일반 사용자들도 시스템의 상황을 볼수 있어서 도움을 주기도 하지만, 간혹 나쁜 의도의 사용자들에 의해서 시스템의 취약점을 노출하는 경로로도 사용된다.
그래서 이런 명령어들을 root만이 사용가능하도록, 일반 사용자들은 사용하지 못하도록 권한을 수정하는 것이 좋다.
여기서는 root 및 일반 관리계정인 zmnkh 계정에게 그 사용권한을 부여하고 나머지 사용자들은 사용하지 못하도록 설정을 하는 방법 및 설정할 명령어들에 대해서 알아보도록 한다.

1) zmnkh 사용자에게 명령을 사용할 수 있는 권한 부여



< 그림 10-2 /etc/group 의 wheel그룹에 zmnkh를 추가한 그림 >

그림 10-2처럼 함으로써 zmnkh는 root와 같은 그룹인 wheel에 속하게 된다.



< 그림 10-3 zmnkh이 속한 그룹 확인 >

그림 10-3처럼 “id" 명령으로 wheel그룹에 속했는지 확인한다.

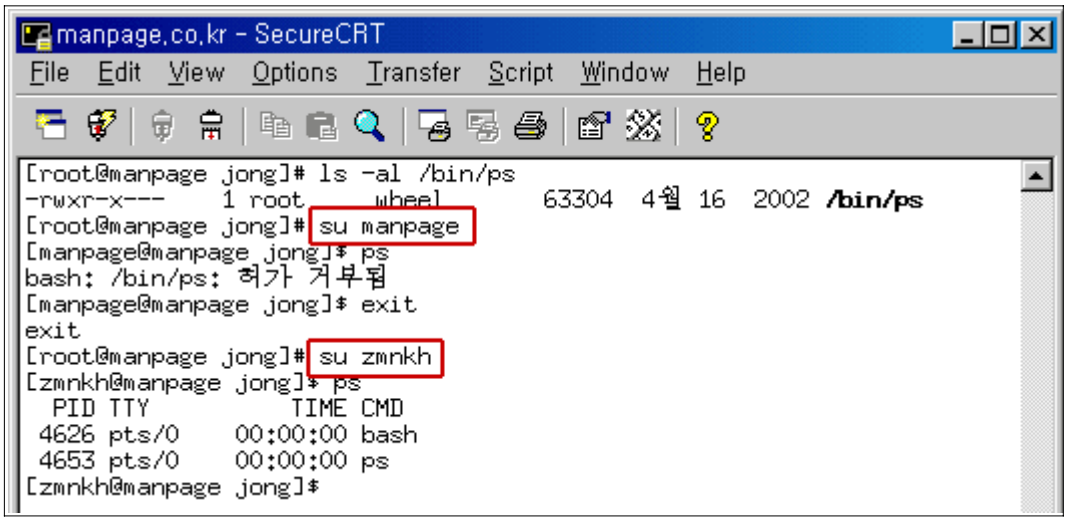
2) 명령어 사용제한

```
#!/bin/bash
chgrp wheel /usr/bin/finger
chgrp wheel /usr/bin/nslookup
chgrp wheel /usr/bin/gcc
chgrp wheel /usr/bin/suidperl
chgrp wheel /usr/bin/whereis
chgrp wheel /usr/bin/cc
chgrp wheel /usr/bin/c++
chgrp wheel /usr/bin/make
chgrp wheel /usr/bin/pstree
chgrp wheel /usr/bin/rlog
chgrp wheel /usr/bin/rlogin
chgrp wheel /usr/bin/which
chgrp wheel /usr/bin/who
chgrp wheel /usr/bin/w
chgrp wheel /bin/mail
chgrp wheel /usr/sbin/sendmail
chgrp wheel /usr/lib/sendmail
chgrp wheel /usr/bin/top
chgrp wheel /usr/bin/free
chgrp wheel /usr/bin/last
chgrp wheel /usr/bin/lynx
chgrp wheel /usr/bin/wget
chgrp wheel /etc/hosts
chgrp wheel /bin/ps
chgrp wheel /etc/hosts.deny
chgrp wheel /etc/hosts.allow
chgrp wheel /etc/named.conf
chgrp wheel /bin/df
chgrp wheel /bin/grep
chgrp wheel /bin/egrep
chgrp wheel /bin/ping
chgrp wheel /bin/netstat
chgrp wheel /usr/bin/ftp
chgrp wheel /usr/bin/ncftp
chgrp wheel /usr/bin/suidperl
chgrp wheel /usr/bin/whereis
chgrp wheel /usr/bin/rz
chgrp wheel /usr/bin/sz
```

```
chmod 750 /usr/bin/finger
chmod 750 /usr/bin/nslookup
chmod 750 /usr/bin/gcc
chmod 750 /usr/bin/suidperl
chmod 750 /usr/bin/whereis
chmod 750 /usr/bin/cc
chmod 750 /usr/bin/c++
chmod 750 /usr/bin/make
chmod 750 /usr/bin/pstree
chmod 750 /usr/bin/rlog
chmod 750 /usr/bin/rlogin
chmod 750 /usr/bin/which
chmod 750 /usr/bin/who
chmod 750 /usr/bin/w
chmod 750 /bin/mail
chmod 750 /usr/sbin/sendmail
chmod 750 /usr/lib/sendmail
chmod 750 /usr/bin/top
chmod 750 /usr/bin/free
chmod 750 /usr/bin/last
chmod 750 /usr/bin/lynx
chmod 750 /usr/bin/wget
chmod 750 /etc/hosts
chmod 750 /bin/ps
chmod 750 /etc/hosts.deny
chmod 750 /etc/hosts.allow
chmod 640 /etc/named.conf
chmod 750 /bin/df
chmod 750 /bin/grep
chmod 750 /bin/egrep
chmod 750 /bin/ping
chmod 750 /bin/netstat
chmod 750 /usr/bin/ftp
chmod 750 /usr/bin/ncftp
chmod 750 /usr/bin/suidperl
chmod 750 /usr/bin/whereis
chmod 750 /usr/bin/rz
chmod 750 /usr/bin/sz
```

< 표 10-2 퍼미션을 변경할 명령어들과 명령어들의 그룹 변경 >

<표 10-2>처럼 함으로 인해서 일반사용자들은 해당 명령어들을 사용할 수가 없으며, root와 zmnkh 사용자만이 <표 10-2>에 나열된 명령어들을 사용할수 있게 된다.



< 그림 10-4 퍼미션 제한이 걸린 파일 실행 >

그림 10-4는 표 10-2에서 설정한 ps명령을 ls로 확인하고 wheel그룹에 속하지 않은 사용자와 wheel그룹에 속한 zmnkh사용자의 ps명령 실행 결과를 보여주는 그림이다.
manpage 유저가 ps명령을 사용 했을때에는 명령어 사용이 거부 되었지만, 관리자로 지정한 zmnkh가 명령어를 사용했을때에는 명령어의 사용 할 수가 있다.

이처럼 사용할수 있는 명령어의 제한을 둠으로써 서버의 보안을 좀더 강화시킬수가 있다.

Ps. 웹호스팅 서버를 운영하는 관리자라면 위의 명령어를 모두 제한하면 안된다. 물론 제한을 해도 상관은 없겠지만 서버에 접속하는 사용자의 불편함을 초래할 수가 있으므로 관리자의 재량에 따라 명령어의 권한을 조정한다. 하지만 서버에 직접 들어와서 작업하는 사용자들은 그리 많지 않다.

3) 불필요한 계정 사용자 삭제

리눅스 서버를 초기 세팅해놓으면 시스템 운영에 필요 없는 계정들이 default로 생성되게 된다.
이들 계정은 해커들에 의해서 사용될 수 있으므로 서버 관리에 필요 없는 계정들은 삭제한다.

```
# 삭제할 계정들은 다음과 같다. (구축된 시스템마다 약간의 차이가 있을 수 있다.)

 userdel adm
 userdel lp
 userdel shutdown
 userdel halt
 userdel news
 userdel uucp
 userdel operator
 userdel game
 userdel gopher
 userdel ftp
 userdel rpm
 userdel xfs (x-windows를 사용한다면 지우지않는다.)
 userdel postgres

 groupdel adm
 groupdel lp
 groupdel games
 groupdel gopher
 groupdel ftp
 groupdel dip
 groupdel rpm
 groupdel xfs (x-windows를 사용한다면 지우지 않는다.)
 groupdel postgres
```

서버에 rcp 등으로 되어있는 계정이 존재한다면, nfs나 원격 접속을 사용하지 않는다면 rcp관련 계정 도 삭제한다.
그리고 삭제하지 않고, 사용하지 않는 계정들은 /etc/passwd를 vi로 open한뒤에 쉘(/bin/bash) 등으로 지정된 부분을 /bin/false로 변경하여 그 계정으로는 서버에 접속을 하지 못하도록 설정을 변경해 놓는다.

4). 파일 변경 제한
/etc 디렉토리 안의 설정파일들은 시스템의 정보를 담고 있기 때문에 노출되었을때 시스템의 정보를 고스란히 파일 열람자에게 알려주게 된다.
이런 파일들을 일반 사용자들이 읽을 수 있는 권한을 제한하고 변경할수 있는 권한을 제한함으로써 보안을 강화시킬수가 있다.

여기서는 파일에 퍼미션을 변경하는 방법과 파일을 변경시킬 수 없도록 속성을 설정하는 부분을 간단히 알아보도록 하겠다.

파일 퍼미션 변경

위의 리눅스 명령 퍼미션 변경을 참고하여 설정을 변경할 수 있다.
우선 시스템 정보를 가지고 있는 파일을 선별해야 한다.
(기본적으로 hosts, hosts.allow, hosts.deny, passwd, shadow, sysctl.conf, lilo.conf, group, service 파일등을 시스템 정보를 가지고 있는 파일들이라 할 수 있겠다.)

```
chmod 640 hosts.allow
chgrp wheel hosts.allow
```

명령으로 root는 읽고 쓰기가 가능하며, 관리자 그룹인 wheel은 읽기만 가능하도록 설정해 놓았다.
일반 사용자들은 이 파일을 읽을수 없다.

파일을 변경 시킬 수 없도록 속성 설정

```
chattr +i /etc/hosts.allow (설정)
chattr -i /etc/hosts.allow (해제)
```

이렇게 함으로써 hosts.allow 파일은 root이외에는 설정된 내용을 수정하지 못하도록 속성을 변경하였다.
혹시나 hosts.allow를 수정할 수 있는 권한을 가진 유저라도 chattr -i /etc/hosts.allow를 하기전까지는 hosts.allow의 내용을 변경할 수가 없다.

물론 root 권한을 획득하고, chattr -i 명령어 까지 알게 된다면 어쩔 도리가 없지만, 해커가 루트권한을 취득하기 전까지는 충분히 안전할 수 있다.
여기까지는 보안에서도 서버 사용자에 대한 내부보안에 대해서 알아보았다.
이제 다음 장에서 알아볼 보안 방법은 서버의 외부에 대한 보안 설정이다.
서버 외부의 설정은 서버 내부의 설정보다 훨씬 까다로우며 설정을 잘못하면 시스템이 정상적으로 운영이 되지 않거나 서비스가 제대로 되지 않는 경우가 발생할 수 있으므로 충분히 읽어보고 이해를 한 후에 외부 보안 설정을 적용하도록 한다.

10-2. tcpwrapper로 서버 IP제한

tcpwrapper는 말그대도 tcp접속에 대한 보호, 방어수단이다.
tcpwrapper가 설치가 되어있다면 (호스트웨이에서는 기본 설치) hosts.allow, hosts.deny로 서버에서 제공하는 서비스 별로 허용하는 IP를 제한할 수가 있다.
서비스 별로 접속할수 있는 IP를 제한 한다는 것은 불특정 다수에 대한 공격에서 벗어날 수 있는 방법이기 때문에 보안상 아주 유리하다.

이번 장에서는 hosts.allow, hosts.deny를 설정하여 서버 보안을 하는 방법에 대해서 알아보도록 하겠다.

hosts.allow는 tcp 접속을 허용하는 설정을 하는 부분이고, hosts.deny는 tcp 접속을 차단하고자 하는 설정이다.
먼저 기본적으로 알아두어야 할 것은 hosts.allow, hosts.deny에서 ssh나 telnet을 차단하는 설정부분을 잘못하면 관리자역시 접속하지 못하는 경우를 만나게 되므로 반드시 관리자의 접속 IP는 개방을 시켜둔후에 설정을 저장하도록 한다.

```
#hosts.allow 설정
cat /etc/hosts.allow
====================================
in.telnetd: 211.239.151.21 211.221.211.100/120
sshd: ALL (유동아이피의 경우를 대비해서 모두 접속이 가능하게 해둔다)
in.ftpd: ALL
ipop3d: ALL
====================================
```

위의 hosts.allow 설정은 만약 telnet을 사용하게 될 때에는 지정된 아이피로만 접속이 가능하게 하고, SSH, FTP, POP접속은 모두 허용을 한 설정이다.

```
# hosts.deny 설정
cat /etc/hosts.deny
====================================
ALL: ALL : ( echo -e "₩n ₩
Process₩:₩t    %d (pid %p)₩n ₩
User₩:₩t    %u₩n ₩
Host₩:₩t    %c₩n ₩
Date₩:₩t    $(date)₩n ₩" | mail -s " 경고! 불법 접속 " hkim@hostway.co.kr ) &
ALL: 211.211.211.211
====================================
```

위의 hosts.deny 설정은 hosts.allow에서 허용한 서버스 이외의 모든 tcp 접속과 211.211.211.211 IP에 대해서는 차단하고 접속 시도가 있을시에 탐지하여 접속 정보를 메일로 전송하라는 설정이다.

위의 설정대로 하게 된다면 서버는 상당히 고립된 상태가 되며, 수많은 메일이 관리자에게 전송이 되어질 것이다. 경고 메일을 많이 받는 것도 경고에 대해서 둔감해질 수 있는 지름길이므로 관리자는 전체 서비스에 대해서 차단을 하는 것보다는 아래처럼 특정 데몬에 대해서 차단을 하는 방법을 택해야 할 것이다.

```
ALL: in.telnetd : ( echo -e "₩n ₩
 이하 동일 ) &
ALL: sshd : (echo -e "₩n ₩
 이하 동일 ) &
```

hosts.deny에서 사용할 수 있는 있도록 지정된 변수는 표와 같다.

변 수	설 명
%a	address (clinet의 IP 주소 값을 가지고 있다.)
%A	address (server의 IP 주소 값을 가지고 있다.)
%c	client info (username을 알 수 있으면, username@hostname 의 값을 가지고 있으나 알 수 없을 때에는 hostname 이나 IP 주소 값을 가지게 된다.)
%d	deamon name (clinet가 접속할 때 호출한 데몬의 이름을 가지고 있다.)
%h	hostname (client의 hostname 값을 가지고 있으나 hostname을 알 수가 없을 때에는 IP주소 값을 가진다)
%H	hostmane (server의 hostname 값을 가지고 있으나, hostnam을 알수가 없을때에는 IP주소 값을 가진다.)
%p	데몬의 프로세서 값을 가진다.
%s	서버의 정보를 가지고 있다 (daemon@host 형태)
%u	client의 user name 값이나 unknown 값을 가지고 있다.
%%	% 문자를 표현한다.

< 표 10-2 hosts.deny에서 사용할 수 있도록 지정된 변수들 >

지금까지 tcpwrapper에 의한 서버 보안을 알아보았다.
다음은 이번 보안의 가장 핵심이 되는 IPTABLES을 이용한 보안에 대해서 알아보도록 하겠다.

10-3. IPTABLES를 이용한 보안

본 문서의 순서대로 해서 IPTABLES 까지 왔다면 기본 보안은 모두 마치고 서버에 기본적으로 설치되어 있는 프로그램을 이용하여 할수 있는 보안의 중급정도의 보안 수준까지 끌어올렸다고 할 수가 있겠다.

"리눅스 보안과 완벽 솔루션" 이란 책에서는 iptables을 다음과 같이 소개하고 있다.

> "iptables는 리눅스에서는 처음으로 상태 추적 기겁이 도입된 방화벽인데 상태추적 방화벽 이란 이전의 리눅스 기반의 방화벽에서는 탐지가 불가능했던 스텔스 스캔을 탐지 및 차단 하는등 상당한 기술적 진보를 이룬 지능형 방화벽을 뜻한다. 또한 iptables는 방화벽을 통과 하는 각각의 연결을 메모리에 저장하므로 레이팅 제한(ratiing limiting)을 이용하여 대부분의 서비스거부 공격도 차단할수 있다."

이 번장에서는 유용하면서도 무료로 제공되고 있는 iptables에 대해서 알아보자.

▶ 사용 명령어

○ 체인 제어 명령

사용법 iptables [-NXPLFZ] [chain명]

IPTABLES 옵션	설 명
-N	새로운 chain 생성 (--new-chain)
-P	체인의 정책 변경 (--policy)
-L	체인에 적용된 정책 리스트 출력 (--list)
-F	체인에 적용된 정책 지우기 (--flush)
-X	비어있는 체인 제거 (--delete-chain)
-E	체인의 이름을 변경 (--rename-chain)
-Z	체인내의 패킷과 바이트의 카운트를 0으로 리셋 (--zero)
-A	체인에 새로운 필터링 규칙을 추가하기 (--append)
-I	체인의 특정 지점에 필터링 규칙 삽입 (--Insert)
-R	체인의 특정 지점의 필터링 규칙을 교환 (--replace)
-D	체인의 특정 지점의 필터링 규칙을 제거 (--delete)

○ 파라미터

IPTABLES 파라미터	설 명
-p (--protocol)	프로토콜을 지정 tcp, udp, icmp or all(모든 프로토콜)을 지정 !(not)를 통해서 역의 결과를 만들 수도 있다. -p tcp (tcp 프로토콜을 사용하는) -p ! tcp (tcp 프로토콜을 사용하지 않는)
-s (--source)	패킷을 발생시키는 발생지 !(not)을 통해서 역의 결과 도출 가능 -s 211.239.151.21 (211.239.151.21을 패킷 발생지로 하는) -s ! 211.239.151.21 (패킷 발생지를 211.239.151.21으로 하는 것을 제외하고)
--sport (--source-port)	패킷을 발생시킨 발생지에서 접속해온 포트 ! 사용 가능 --sport 21 (21번 포트에서 발생한 패킷)
-d (--destination)	패킷이 도착하는 지점. 사용법은 souece와 동일

--dport (--destination-port)	패킷의 도착지 포트, 사용법은 sport와 동일 포트가 연속적으로 여러 개일 경우 '-' 로 표현할 수 있다 (20-22 → 20, 21, 22를 의미)
-j (--jump)	필터링 규칙에 의해 적용되는 패킷을 target으로 보낸다.
-i (--in-interface)	패킷이 들어올 때 경로가 되는 인터페이스를 지정. !(not) 사용가능 interface = lo (localhost), eth0 (랜카드 한개일 경우), -i eth0 (설정된 랜카드를 통해서 들어오는)
-o (--out-interface)	패킷이 나갈 때 경로가 되는 인터페이스를 지정 ! 사용가능 -o eth0 (랜카드로 나가는 패킷을 지정) -o ! eth0 (랜카드로 나가는 패킷을 제외하고)
-f (--fragment)	패킷이 한번에 전달되지 못할만큼 크기가 클 때 패킷을 여러개로 나누어서 (분절) 여러개의 패킷으로 전달할 때 사용. 이들 패킷은 목적지에 도착해서 재구성되어 전체 패킷이 된다. !(not) 가능
-c (--set-counters)	INSERT(-I), APPEND(-A), REPLACE(-R) 명령을 사용하는 동안 규칙의 패킷과 바이트의 카운터를 초기화 시킨다.

※ 참고 그 외 기타 옵션으로 -m (match)를 사용해서 확장시킬수 있는 옵션이 있으나,
그것은 개인적으로 알아보기 바라며 여기에서는 설명하지 않는다.

< 사용법 >

```
iptables -[ADC] chain rule-specification [options]
iptables -[RI] chain rulenum rule-specification [options]
iptables -D chain rulenum [options]
iptables -[LFZ] [chain] [options]
iptables -[NX] chain
iptables -P chain target [options]
iptables -E old-chain-name new-chain-name
```

chain - INPUT, OUTPUT, FORWARD or 직접 생성한 chains
rulenum - 규칙이 생성된 순서 (순서대로 1,2...)
rule-specification - 파라미터를 사용하여 만들어진 규칙 (필터링 규칙)
target - ACCEPT, DROP, QUEUE, RETURN

이제 기본적인 것은 모두 알아보았고, 실제 필터링 적용을 통한 예제로서 앞에서 배운 명령과 옵션을 익혀 보도록 하자.

< 적용하고자 하는 방화벽 예 >

- 기본 정책을 DROP으로 선택한다.
- 새로운 체인을 생성하여 들어오는 패킷에 대한 필터링 체인으로 사용한다.
- 로컬에서 발생하는 패킷은 모두 허용한다.
- 웹서비스를 하는 80번 포트는 모든 사람이 접속할 수 있도록 허용한다.
- ftp와 ssh는 211.239.0.2에서만 접속할 수 있도록 한다.
- snmp 사용을 위해서 snmp udp(161) 포트를 mrtg서버에서만 접속할 수 있도록 허용한다.
- 허용된 ip에서만 211.239.0.2 에서만 서버로 ping이 가능하도록 설정한다.
- 서버에서는 모든 곳으로 ping이 되도록 허용한다.
- domain resolve를 위해서 dns(53) 포트를 모두에게 허용한다.
- mysql을 모두에게 허용한다.
- Foward는 하지 않는다.
- 클래스별 비공인 아이피가 외부에서 접속되는 것과 외부로 나가는 것을 막는다.
- 허용하지 않은 모든 tcp, udp, icmp의 입/출입을 막는다.

서버의 IP : 211.239.0.0
MRTG server IP : 211.239.0.1
원격 접속하는 IP : 211.239.0.2

< 방화벽 규칙에 따른 필터링 적용 예 >

```
iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FOWARD DROP

iptables -N NEWINPUT
iptables -F
iptables -A NEWINPUT -i lo -j ACCEPT
iptables -A NEWINPUT -i eth0 -p tcp -s 0/0 -d 0/0 --dport 80 -j ACCEPT
iptables -A NEWINPUT -i eth0 -p tcp -s 211.239.0.2 -d 211.239.0.0 ₩
--dport 20-22 -j ACCEPT
iptables -A NEWINPUT -i eth0 -p udp -s 211.239.0.1 --dport 161 -j ACCEPT
iptables -A NEWINPUT -i eth0 -p tcp -s 0/0 -d 0/0 --dport 3306 -j ACCEPT
iptables -A NEWINPUT -i eth0 -p udp -s 0/0 -d 0/0 --dport 53 -j ACCEPT
iptables -A NEWINPUT -i eth0 -p icmp -s 211.239.0.2 ₩
--icmp-type echo-request -d 211.239.0.0 -j ACCEPT
```

```
iptables -A NEWINPUT -i eth0 -s 0.0.0.0/8 -j DROP
iptables -A NEWINPUT -i eth0 -s 127.0.0.0/8 -j DROP
iptables -A NEWINPUT -i eth0 -s 169.254.0.0/16 -j DROP
iptables -A NEWINPUT -i eth0 -s 192.0.2.0/24 -j DROP
iptables -A NEWINPUT -i eth0 -s 224.0.0.0/3 -j DROP
iptables -A NEWINPUT -i eth0 -p tcp -j DROP
iptables -A NEWINPU -i eth0 -p udp -j DROP
iptables -A NEWINPUT -i eth0 -p icmp -j DROP
iptables -A INPUT -i eth0 -p all -s 0/0 -j NEWINPUT

iptables -A OUTPUT -o lo -j ACCEPT
iptables -A OUTPUT -o eth0 -p tcp ! --syn -s 0/0 -d 0/0 -j ACCEPT
iptables -A OUTPUT -o eth0 -p tcp -s 211.239.0.0 -d 211.239.0.2 -j ACCEPT
iptables -A OUTPUT -o eth0 -p tcp -s 0/0 -d 0/0 3306 -j ACCEPT
iptables -A OUTPUT -o eth0 -p udp -s 0/0 -d 0/0 --dport 53 -j ACCEPT
iptables -A OUTPUT -o eth0 -p icmp -s 211.239.0.0 --icmp-type echo-request ₩
-d 0/0 -j ACCEPT
iptables -A OUTPUT -o eth0 -s 0.0.0.0/8 -j DROP
iptables -A OUTPUT -o eth0 -s 127.0.0.0/8 -j DROP
iptables -A OUTPUT -o eth0 -s 169.254.0.0/16 -j DROP
iptables -A OUTPUT -o eth0 -s 192.0.2.0/24 -j DROP
iptables -A OUTPUT -o eth0 -s 224.0.0.0/3 -j DROP
iptables -A OUTPUT -o eth0 -p tcp -j DROP
iptables -A OUTPUT -o eth0 -p udp -j DROP
iptables -A OUTPUT -o eth0 -p icmp -j DROP
```

< 방화벽 규칙에 따른 설정 예 >

위의 규칙대로 설정하면 지정된 IP (211.239.0.2)를 제외하고는 외부에서는 WebService이외에는 아무것도 허용되지 않으므로 보안상으로는 좋지만, 서버가 고립되게 된다.
위의 설정은 iptables의 설정 방식과 파라미터의 사용법을 알고자 예제로 만들어 놓은 것이므로 관리자는 iptables의 명령과 파라미터를 이용 각자의 서버에 알맞은 정책을 설정하면 된다.
단순한 명령과 동일한 파라미터가 반복되는 방식이므로 쉽게 사용할 수 있을 것이다.

iptables에 대한 자세한 사용설명서는 서버에서 “man iptables“ 하면 볼 수 있다.

11. 보안 도구

지금까지 알아본 것은 서버 내부에서 제공하는 프로그램을 이용한 보안이었다.
물론 내부의 프로그램만으로도 보안이 되지만 (설정이 정상적으로 되어있을 경우) 이에 더하여 외부 보안 프로그램을 사용함으로써 보안을 더욱 강화시킬 수 있다.
보안 도구에는 많은 프로그램이 있지만 여기서는 강력한 포트스캔 방지 프로그램 portsentry와 포트검사 프로그램인 nmap, 포트검사 및 동작하고 있는 프로세서들을 살펴볼수 있는 lsof에 대해서 알아보도록 한다.

11.1 portsentry

portsentry는 tcpwrapper와 연계하여 포트스캔이 감지되면 포트스캔을 실시한 IP를 hosts.deny에 등록하여 서버에 접속하지 못하도록 실시간으로 방어하는 프로그램이다.

portsentry는 실시간 방어를 구현하는 아주 우수한 스캐너이지만, 보통 해킹을 하고자 대상 서버의 열린포트를 스캔하는 해커들은 자신의 IP가 아닌 spoofing 된 거짓 아이피를 사용하기 때문에 로그에 기록된 거부된 IP들을 해킹을 하려는 서버로 의심하여 역공격을 하게 되면 선의의 피해자가 생길수도 있다는 것을 명심해야 한다.

< 설치 방법 및 사용방법 >
- 설치
1. **portsentry-1.1.tar.gz**을 다운받는다.
2. 서버의 적당한 위치에 업로드 (/usr/local/src 정도에 파일 업로드)
3. **tar -zxvf portsentry-1.1.tar.gz**
4. 압축이 해제되면 portsentry-1.1 이 생성된다.
5. **cd portsentry-1.1**로 이동
6. **./make linux**
7 **./make install** 로서 설치를 끝마친다.

- 사용방법
설치가 완료되면, /usr/local/psionic/portsentry란 디렉토리가 생성된다.
cd /usr/local/psionic/portsentry로 이동한다.

이동한 뒤에 ls 명령으로 파일을 보게 되면
portsentry → 실행파일
portsentry.conf → 설정파일
portsentry.ignore → 스캔을 해도 로그에 남지않는 IP를 적어줄 파일

이 세 가지 파일이 존재할 것이다.
portsentry.conf에는 portsentry의 각종 설정들을 편집할 수 있지만, default로 사용해도 무방하다.

portsentry.ignore에는 로컬호스트는 반드시 들어가 있어야 하며, 기타 자신이 자주 사용하는 IP등을 지정해 놓아야 혹시 테스트로 스캔을 했을 경우 접속이 차단되지 않는다.

▶ 실행 : /usr/local/psionic/portsentry/ 디렉토리에 있다고 가정

portsentry를 실행하는 모드에는 크게 3가지 모드가 있다.

◆Class mode (백그라운드에 있는 모든 tcp, udp 포트들을 모니터링 한다. stealth scan은 탐지 못함)

./portsentry -tcp
./portsentry -udp

◆ Stealth mode (모든 tcp, udp를 모니터링 하는 것은 Classic과 같지만 stealth scan을 탐지한다.)

./portsentry -stcp
./portsentry -sudp

◆ Advanced mode (portsentry.conf에서 옵션으로 지정된 포트들은 제외하고 나머지 포트들에 영에 내에서 임의대로 모니터링을 실시하며, 모든 실행 옵션 중에서 가장 민감하고 빠른 반응속도로 포트스캔 탐지하지만 민감한 만큼 부정확한 정보 전달의 가능성이 있다.

./portsentry -atcp
./portsentry -audp

위의 모드들은 동시에 여러모드가 동시에 사용될수 없으며, 반드시 한가지 모드만 동작을 시켜야 한다. 일반적으로는 **portsentry -stcp , portsentry -sudp** 옵션을 많이 사용하고 있다.

< 동작 확인 >

위의 세가지 동작중 -stcp, -sudp로 동작시켰을때 **ps -aef | grep portsentry** 명령을 내렸을 때
./portsentry -stcp
./portsentry -sudp 라인이 있으면 정상적으로 동작하고 있는 것이다.
만약 위와 같은 파일이 존재 하지 않는다면, 실행이 잘못 되어진 것이므로, 다시 한번 실행 방법을 읽어본 후에 실행을 시켜 본다.

< 스캔 탐지 로그 >

portsentry를 동작시켜 놓았을 때 외부에서 스캔 공격이 들어오면
portsentry.blocked.stcp
portsentry.blocked.sudp
portsentry.history 세 가지의 파일이 생성된다.

각 로그 파일의 역할과 특성을 알아보도록 한다.

postsentry 로그파일	설 명
portsenty.blocked.stcp	tcp 측으로 스캔이 탐지되었을때 로그가 기록된다. (휘발성이다. portsentry를 재시작하면 기록이 삭제된다.)
portsentry.blocked.sudp	udp 측으로 스캔이 탐지되어있을 로그가 기록된다. (휘발성이다. portsentry를 재시작하면 기록이 삭제된다.)
portsentry.history	tcp,udp의 모든 로그기록이 종합되어 남는다. 비 휘발성으로 portsentry를 재시작해도 로그값이 그대로 남아있게 된다. 로그는 누적되어 기록된다.

< 표 11-1 portsentry의 로그 파일들 >

portsentry에 의해서 로그에 기록된 IP는 /etc/hosts.deny에 보면 모든 서비스가 차단되는 ALL에 등록되어 있을 것이다.
(ex : ALL: 211.xxx.xxx.xxx)

관리자는 history로그를 잘 살펴보아 스캔공격이 얼마나 이루어지고 있는지 파악해야 하며, 스캔 공격이 많이 이루어 지고 있을 때에는 서버에 쓸데없이 열린 포트는 없는지, 취약한 패키지 프로그램은 없는지 항상 관심을 가져야 한다.

11.2 nmap

portsentry가 포트스캔을 방지하는 것이라는 것을 알았다.
하지만 이번에 알아볼 프로그램은 아이러니하게도 포트를 스캔하는 프로그램이다.
“보안도구는 해킹도구“ 란 말이 있다.
보안도구를 다른 목적으로 사용하면 해킹도구가 되는 것이다. 또 그와 반대로 해킹도구를 다른 목적으로 사용하면 보안을 강화하는 도구로 사용될 수도 있는 것이다.
강도가 칼을 들면 흉기고, 의사가 칼을 들면 메스 이듯이..
좋은 의미로 nmap은 시스템의 취약점이나 포트 정보를 얻기 위한 목적으로 사용되는 프로그램이다.
nmap은 예전에는 추가로 설치해야 하는 프로그램이었으나 최근에는 패키지로 같이 포함되어서 나오고 있다. 최신버전은 nmap-3.20-1 버전까지 나와 있다 (소스와 rpm 모두 존재하므로 편한대로 받아서 업그레이드 및 새로 설치를 하면 된다.)

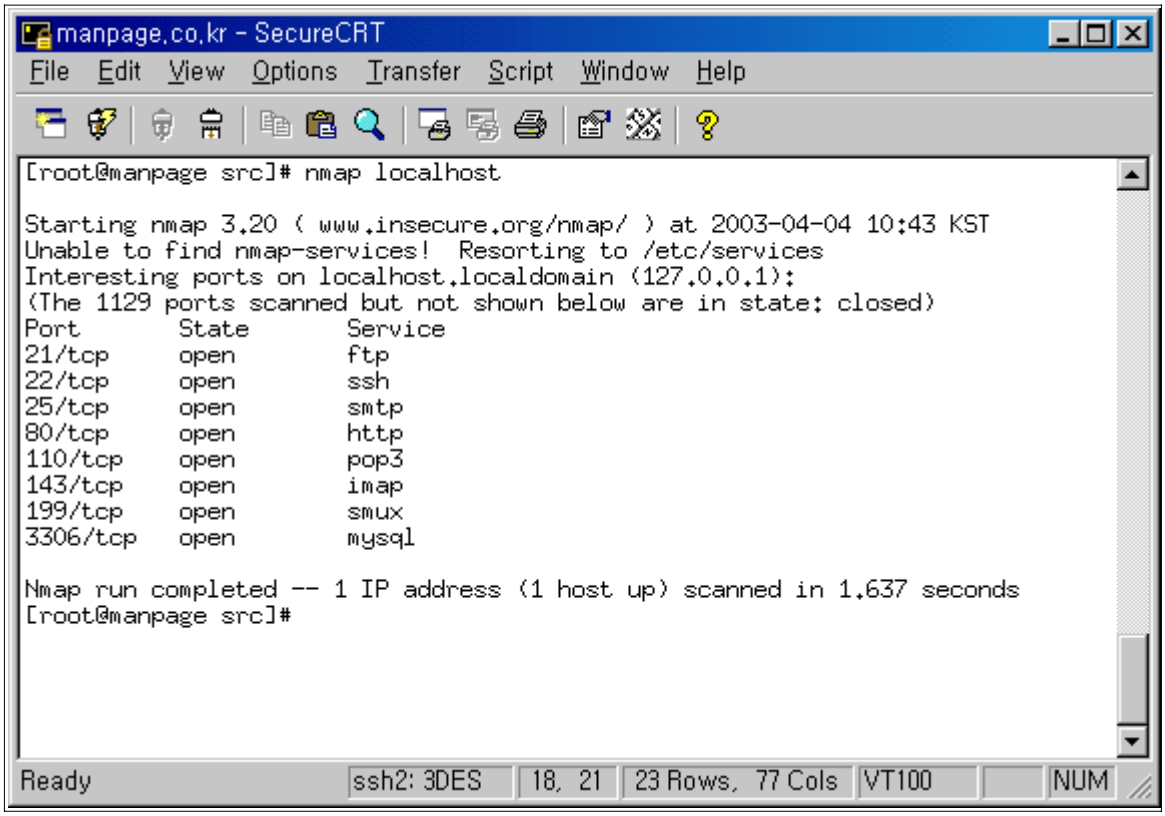
여기에서는 rpm으로 설치하는 법을 알아본다.

< 설치방법 >
rpm -Uvh namp-3.20-1.i386.rpm

< 사용방법 >
일반적인 스캔방법은
nmap IP (스캔하고자 하는 상대방의 IP)

예) 로컬호스트를 스캔할 때

nmap localhost

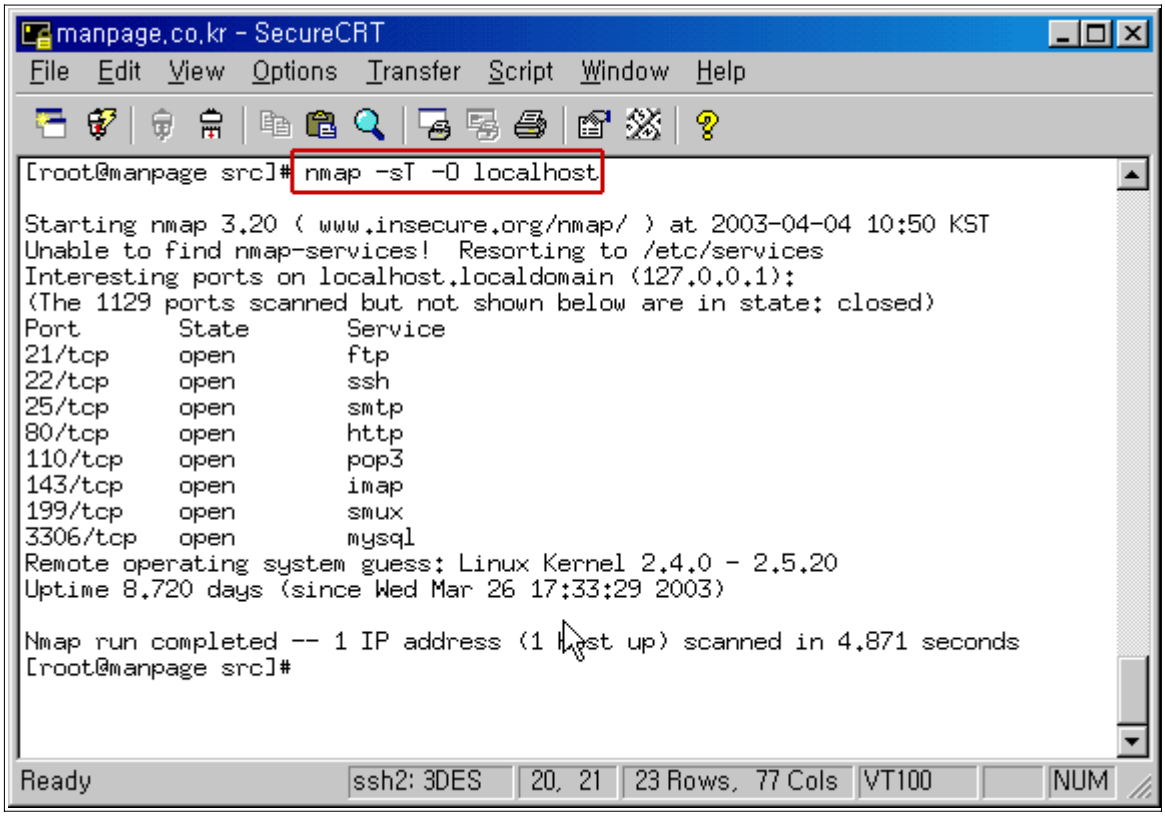


< 그림 11-1 localhost를 스캔한 모습 >

웹서버로 운영되는 서버이기 때문에 웹에 관련된 서비스 포트가 열려있는 것을 확인할 수가 있다.

nmap은 포트스캔을 통해서 열려있는 포트 정보를 알아볼수만 있을 뿐만 아니라. 커널버전 (OS의 종류), 서버가 꺼지지 않고 동작된 시간, 정보를 알아내기까지 걸린시간등도 알아낼 수 있다.

스캔의 가장기본이 되는 명령을 사용하여 시스템의 정보를 확인해보자.



< 그림 11-2 OS 및 uptime시간, 열린 포트 확인 >

위의 그림에서 얻을 수 있는 정보는 열린 포트 목록, OS종류, 커널 버전, 서버가 다운되지 않고 가동된 시간 (3월 26일 17시 33분 29초이래로 서버가 계속 동작되고 있음을 알려준다)

nmap 에는 많은 옵션들이 있다.
그중에서 대표적으로 사용하는 옵션들을 알아보도록 하겠다.

옵 션	설 명
sT	열려있는 TCP포트를 체크하기 위해서 사용된다. 가장 일반적이며, 쉽게 체크되고 필터링된다.
sU	열려있는 UDP포트를 체크한다. (정확히 말하면 UDP의 닫혀 있는 포트를 검색하는것임) -sU만 UDP를 체크할수 있고 나머지 옵션들은 TCP만 체크한다.
sS	tcp에 syn(연결을 요청하는 패킷)패킷을 보내고 응답이 있는 포트는 열린포트로 감지하는 방식으로 root권한이 있어야 실행할수 있다.
sR	rcp scan (서버내에서 rcp를 사용하여 열린 포트를 검색하고 rcp 사용 프로그램을 같이 출력한다.)
sF, sX, sN	stealth 기능으로 로그에 기록되지 않는 스캔 방식이다. portsentry의 stealth 스캔 탐지 기능으로 체크가 가능하다.
sP	ping을 이용하여 스캔하는 방식 (호스트가 살아있는지 여부만 판단)
PI	icmp ping을 이용한 스캔 방식
O	os를 판별하기 위한 옵션
p	포트의 범위를 지정한다. (-p 1-100 => 1번부터 100까지 포트를 검색한다.)
F	/etc/service에 등록되어 있는 포트만을 스캔한다. (fast scanmode)

이외에도 많은 옵션들이 있지만, 잘 사용하지 않는 옵션들이고, 악용할 소지가 충분히 있는 옵션들이기 때문에 여기서는 설명하지 않고 관리자의 판단에 맡긴다. 그 외 nmap을 더 자세히 알고 싶은 관리자들은 서버에서 “man nmap” 명령을 입력하면 자세한 nmap의 설명이 나오며, 인터넷을 통한 검색에서도 여기에서 설명한 것보다 자세하고 많은 옵션에 대한 정보를 알아 볼수있을 것이다.

최근 보안에 대한 관심이 높아지면서 스캔을 해킹을 위한 사전작업으로 인식하고 있으므로 무모한 다른 서버의 스캔으로 인해서 관리자 자신이 고소를 당한다던지 역해킹을 당한다던지 하는 등의 피해를 입을수 있음으로 스캔 프로그램 사용에 주의해야 하며, nmap 프로그램은 자신이 관리하는 서버들에 대해서 정보를 파악하고 불법 포트가 오픈 되지 않았나 하는 등의 보안용으로만 사용하기 바란다.

11.3 losf

보안도구의 마지막으로 lsof에 대해서 알아보기로 하겠다.

lsof는 현재 실행되고 있는 프로세서와 열린 포트. 그리고 그 포트가 참조하는 파일등을 알 수 있는 프로그램이다.

이상한 프로세서가 실행중이거나 수상한 포트가 열려있을때, 그 포트가 어떤 프로그램을 참조하고 어떤 프로세서에 의해서 열려있는지 확인이 가능하다.

< 설치방법 >

소스, rpm이 있으며, 여기서는 소스로 설치하는 법을 알아본다.

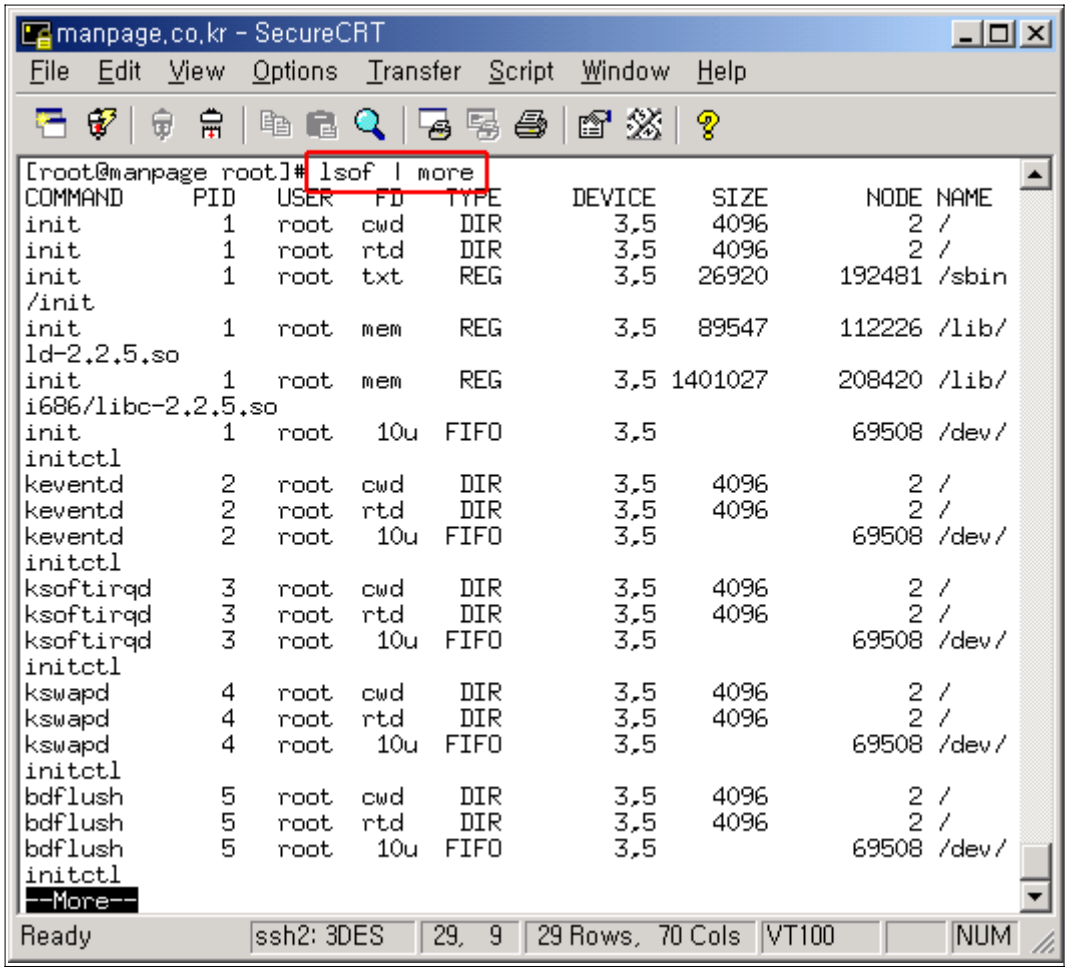
(서버에는 기본적으로 rpm -Uvh lsof-4.51-2.i386.rpm가 설치되어 있다.)

tar -zxvf lsof_4.64.orig.tar.gz
cd lsof_4.64_src
./Configure linux
이때 Inventory Mode를 실행할 것인지 물어본다. Inventory Mode는 패키지의 이상유무를 판단하는 것으로서 테스트한다. 후에 Customization에 대해서도 물어보지만 모두 디폴트값으로 사용해도 무방하다. (./Configure linux 한 이후에 나오는 질문에 모두 enter만 입력하면 된다)
make
make install

여기까지 이상 없이 되었다면 모든 설치가 완료가 된 것이다.
사용을 편리하게 하기 위해서 생성된 lsof 명령을 PATH가 걸려있는 디렉토리로 복사한다.
path가 걸려있는 디렉토리의 확인은 쉘스크립트 할때 알아보았다. (echo $PATH)

< 사용법 >
설치가 완료되었다면 이제는 사용법에 대해서 알아보자

lsof < 옵션>



< 그림 11-5 lsof의 출력 >

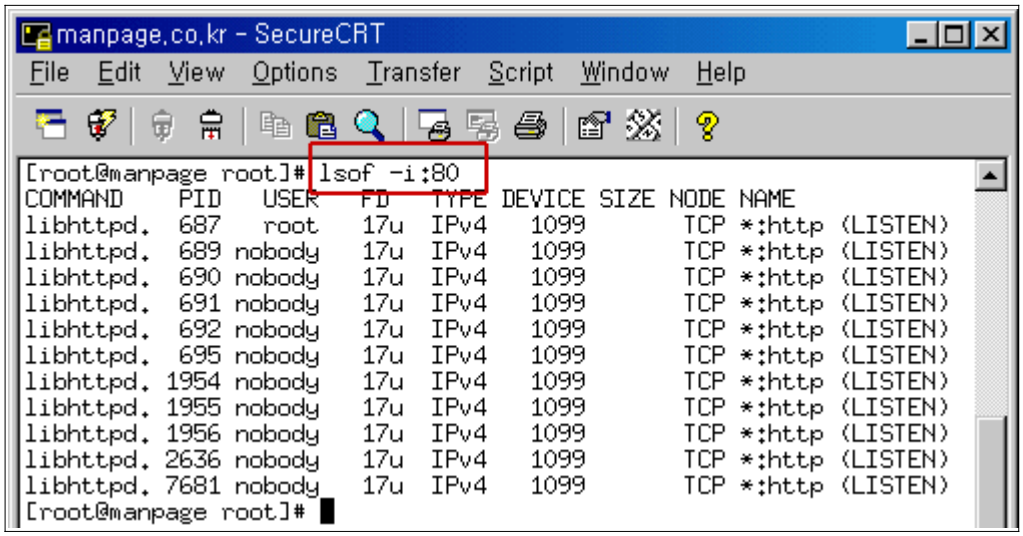
그림 11-5는 옵션없이 lsof를 사용한 그림이다. 서버에서 동작하고 있는 모든 프로세서 열린 파일들이 출력되므로 more 명령어를 사용해서 한 화면씩 나누어 본 것이다.

lsof의 주요 옵션을 알아본 후에 그 옵션들을 이용하여, 의심스러운 포트와 그 포트를 사용하는 프로그램이 무엇인지 알아보도록 하겠다.

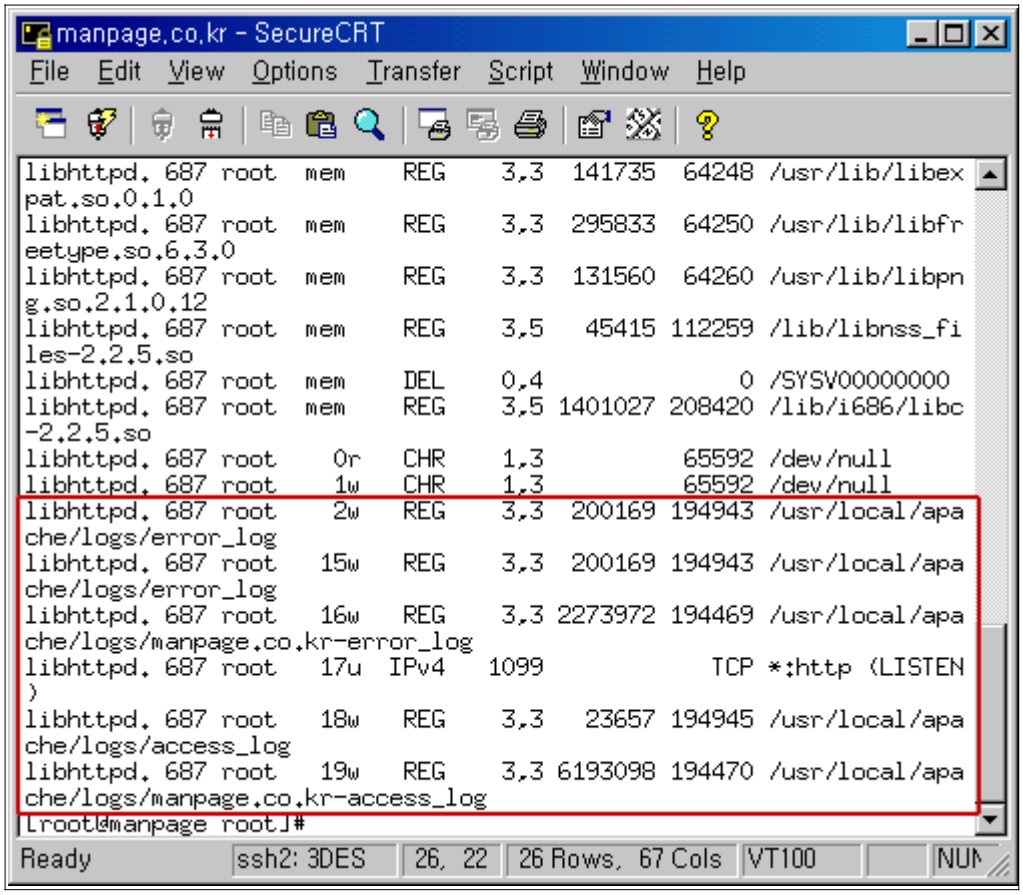
옵 션	설 명
-t	현재 동작되고 있는 프로세서의 PID를 출력한다.
-p [pid]	해당 pid가 참조하고 있는 프로그램과 파일들과 라이브러리가 출력된다.
-i[:port]	현재 열려있는 소켓들에 대해서 출력한다. -i[:port] 특정포트에 대해 출력
-i@<ip>	해당 아이피가 접속 정보를 출력한다. -i@<도메인>으로도 가능. 대신 도메인은 resolve되어서 IP로 변환이 가능해야 한다.
-T	현재 열려있는 소켓들 중에서 TCP에 대한 것만을 출력
-U	현재 열려있는 소켓들 중에서 UDP에 대한 것만을 출력
-l	로그인 계정대신에 UID로 표시
-n	호스트네임대신에 IP로서 출력
-r [초]	losf의 명령을 -r 다음에 지정한 시간을 주기로 반복 실행한다.
-v	lsof에 대한 정보를 출력한다.
-u<user>	특정 사용자에 의해서 실행되고 있는 프로그램과 참조하고 있는 파일을 출력한다. -u<uid>를 사용해도 같은 결과를 얻는다.
-c<데몬>	특정 데몬들이 참조하고 있는 파일들을 출력한다.

그 외 다른 옵션들은 “lsof -h“를 통해서 확인할 수 있다.

예제) 80포트는 웹페이지를 띄울수 있도록 httpd 데몬이 사용하는 포트이다. 이미 알고 있는 이 포트를 예로들어서 사용 중인 프로세서와 그 프로그램이 참조하는 파일들을 알아본다.
또한 그 프로그램을 실행한 유저에 대해서도 알아본다.



< 그림 11-6 lsof를 사용하여 특정포트 확인 >



< 그림 11-7 lsof를 사용하여 참조하고 있는 파일 확인 >

그림 11-7은 “lsof -p 687” 명령으로 그림 11-6에서 확인된 결과를 참고로 알아낸 PID를 추적하여 해당 PID가 사용하고 참조하고 있는 프로세서 및 파일들을 확인한 결과이다.

종합하면, 80번 포트를 사용하는 것은 httpd라는 데몬이고, 여러 데몬중에서 PID 687을 추적하였더니 /usr/local/apache/logs/error_log 등등의 파일을 참조하고 있음을 확인하였다.
lsof는 매우 유용한 프로그램 이므로, 위에 있는 옵션들만이라도 반드시 숙지하여 두기 바란다.

이 번장에서는 보안에 대해서 알아보았다. 보안을 단지 몇 장의 페이지로 설명을 할 수는 없지만 기본적으로 알아두어야 하는 것들을 정리해 둔 것이므로 꼭 숙지하기를 바라며, 이번 장에서 설명되었던 프로그램들을 실제로 꼭 사용해 보기 바란다.

그리고 반드시 명심해 두어야 할 것은 아무리 보안이 철저하더라고 해도 보안에 취약한 패키지를 사용한다거나 취약한 프로그램을 사용하게 되면, 보안 설정과는 무관하게 시스템의 보안이 뚫리는 경우가 많다. certcc.or.kr등을 통해서 취약한 패키지나 해킹 동향을 항상 주의깊에 살펴보아야 하며, root 패스워드를 주기적으로 바꾸어주고 시스템 사용자들이 사전식 패스워드나, 짧고 쉬운 패스워드를 사용하지 않도록 주의를 주어야 한다.
한순간의 방심이 소중한 데이터를 한번에 날려버릴수도 있다는 것을 깨닫고, 보안과 백업을 병행하여, 안전하고, 신뢰성 있는 서버를 운영할 수 있는 관리자가 되어야 할 것이다.

12. 설치

이곳에서는 apache, mysql, php등의 설치와 서버에 필요한 프로그램들의 설치하는 방법을 살펴본다.
(기본 APM의 설치 - 설치순서는 mysql → apache → php → apache → ZendOptimizer 순으로 한다. static설치)

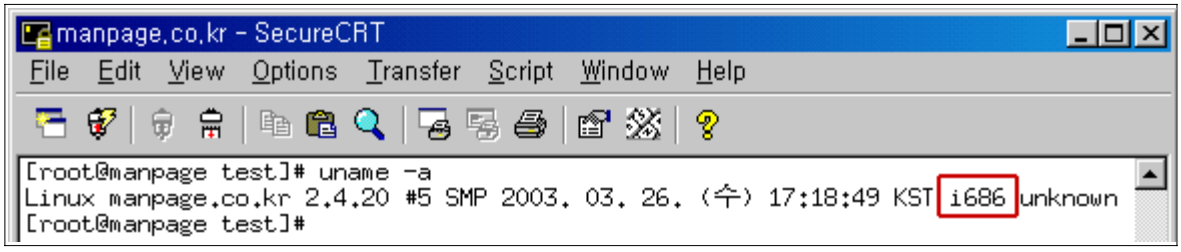
> **참고 : 설치를 하기위한 파일들이 존재하는 경로는 "/usr/local/src" 로 하기로 한다.**

12.1. mysql

mysql-3.23.56.tar.gz을 http://www.mysql.org로부터 다운을 받는다
서버로 업로드 한다. (참고 사항을 보기 바람)
tar -zxvf mysql-3.23.56.tar.gz 으로 압축을 해제한다.
압축이 해제되면 mysql-3.23.56 이란 디렉토리가 생성되어져있을 것이다.
cd mysql-3.23.56 으로 이동한다.

CFLAGS="-march=i686 -funroll-loops -fomit-frame-pointer" ./configure ₩
--prefix=/usr/local/mysql --with-charset=euc_kr ₩
--localstatedir=/usr/local/mysql/data

위의 명령으로 configure를 실행한다. 단 위의 옵션은 i686에서 최적화시키기 위한 옵션이므로 uname -a 명령으로 자신의 머신 버전을 확인한 후에 설정하도록 한다. (최근에 나온 대부분의 서버는 i686이다.)



< 그림 12-1 uname 으로 머신 확인 >

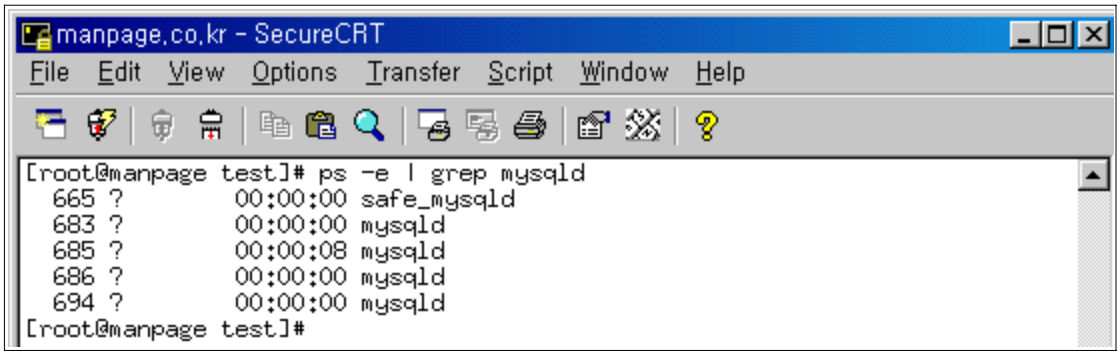
configure가 에러없이 정상적으로 완료되었다면,
make
make install 명령으로 mysql을 컴파일한다.
이상없이 컴파일 되었다면
scripts/mysql_install_db를 실행한다. (한번만 실행해야 한다.)
이 것을 실행시키면 database가 생성이 된다.

여기까지 되었다면 /usr/local/mysql이 생성되었을 것이다.
이제는 mysql을 실행시킬 사용자를 생성한다.
groupadd mysql mysql이란 그룹을 생성하고

useradd -g mysql -s /bin/false mysql 쉘에 접속할 수 없는 mysql 유저를 생성한다. 보안상 필요

다음 mysql 디렉토리 및 data가 보관 되는 data 디렉토리의 권한을 설정한다.
(mysql의 data디렉토리는 반드시 mysql이 소유권자이어야 한다.)
chown -R root /usr/local/mysql
chown -R /usr/local/mysql/bin
chgrp -R mysql /usr/local/mysql
chown -R mysql. /usr/local/mysql/data (data의 소유와 그룹권한을 mysql로 변경한다)

/usr/local/mysql/bin/safe_mysqld --user=mysql --language=korean &
다음 위와 같은 명령으로 mysql을 실행시키면 설정이 끝나게 된다.



< 그림 12-1 mysql 동작 확인 >

<참고!!>
mysql에는 test라는 database가 생성되는데 필요가 없는 database이므로 삭제한다.
/usr/local/mysql/bin/mysqladmin -u root -f DROP test
삭제후 mysql을 갱신해준다.
/usr/local/mysql/bin/mysqladmin -u root reload

12.2. php
static으로 설치를 하면 DSO보다 속도가 빠르므로 static으로 설치를 한다.
(하지만 호스트웨이에서는 확장성을 위해서 아파치를 DSO로 설치를 하고, php는 static으로 설치를 한다.)
php를 static으로 설치를 하기 위해서 미리 apache를 configure를 해준다. (configure만 한다.)

apache-1.3.27.tar.gz을 http://www.apache.org로부터 다운 받는다.
서버에 다운받은 압축 파일을 올려놓고 압축을 해제한다.
tar -zxvf apache_1.3.27.tar.gz
압축을 해제하면 apache_1.3.27 이란 디렉토리가 생성될 것이다.
cd apache_1.3.27로 이동하여

CFLAGS="-O3 -march=i686 -funroll-loops -fomit-frame-pointer" ./configure ₩
--prefix=/usr/local/apache

위의 명령으로 configure 한다.

apache configure가 정상적으로 되면.
php-4.3.1.tar.gz을 http://www.php.net으로부터 다운 받는다.
다운받은 php를 서버에 업로드 한다.
tar -zxvf php-4.3.1.tar.gz 로 압축을 해제한다.
압축을 해제되면 php-4.3.2RC1 디렉토리가 생성 될 것이다.
cd php-4.3.1 로 이동한다.

CFLAGS="-O3 -march=i686 -funroll-loops -fomit-frame-pointer" ₩
./configure --prefix=/usr/local/php4 --with-apache=../apache_1.3.27 ₩
--with-mysql=/usr/local/mysql ₩
--with-db --with-gd --with-jpeg-dir=/usr --with-png --with-zlib ₩
--enable-tracvars --disable-debug --enable-pic --enable-shared --enable-sockets ₩
--with-db2 --enable-dbase --enable-mailparse ₩
--enable-sysvsem=yes --enable-sysvshm=yes ₩
--enable-ftp --enable-calendar --enable-inline-optimization --enable-trans-sid

위의 옵션으로 ./configure한다.
위의 옵션대로 설치할 필요는 없으며, 각자 설치하여 사용하고자 하는 서버의 특성에 맡게 설치를 하면 되겠다. 호스트웨이에서 기본설치시에는 위의 옵션에 imap을 추가 설치하여 포함한다.

./configure가 이상없이 수행되었다면,
make
make install로서 컴파일을 마무리 한다.
설정이 완료된 후에 php.ini-dist 파일을 열어 register_globals = Off를 On 으로 변경 시켜준다.
이것은 기존에 사용하던 php 버전과의 호환을 위한 것이다.
그리고 나서 php.ini-dist를 /etc/php.ini, /usr/local/php4/lib/php.ini 로 복사해준다.
cp php.ini-dist /etc/php.ini
cp php.ini-dist /usr/local/php4/lib/php.ini

12.3. apache
apache는 동적으로 컴파일 하게 될 것이다.
php 설치가 정상적으로 에러없이 끝났다면 이제는 apache를 설치한다.

apache는 이미 앞에서 소스를 받아서 압축을 푼후 configure까지 해놓은 상태이다.
그러나 php를 인식하기 위해서 다시한번 configure를 해야 한다.
cd /usr/local/src/apache_1.3.27 로 이동하여

CFLAGS="-O3 -march=i686 -funroll-loops -fomit-frame-pointer" ₩
./configure --prefix=/usr/local/apache --enable-rule=SHARED_CORE ₩
--enable-module=so --activate-module=src/modules/php4/libphp4.a

위와 같은 옵션으로 configure를 한다.
이상없이 ./configure가 되었다면.
make
make install 로서 컴파일을 한다.

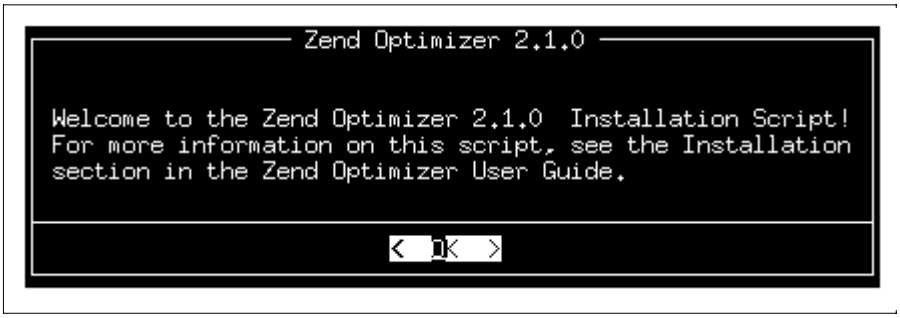
컴파일이 완료되면 /usr/local/apache/bin/apachectl start 명령으로 apache를 실행한다.
실행시키기 전에 php를 위해서 httpd.conf에서 설정해 줘야하는 부분이 있는데 그것은 2장의 httpd.conf 설정 section 2를 참고하기 바란다.

12.4. Zend-Optimizer 설치

php 성능을 향상시키기 위해서는 Zend를 설치해 주어야 한다.
호스트웨이에서는 기본적으로 각각 설치되는 php 버전에 맡는 Zend를 같이 설치한다.

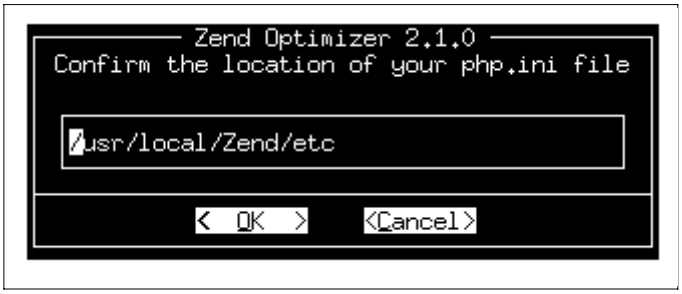
ZendOptimizer-2[1].1.0a-Linux_glibc21-i386.tar.gz을 http://www.zend.net 으로부터 다운받는다.
다운받은 Zend를 서버에 업로드 한다.

tar -zxvf ZendOptimizer-2.[1].1.0a-Linux_glibc21-i386.tar.gz 압축을 해제한다.
압축을 해제하면 ZendOptimizer-2.1.0a-Linux_glibc21-i386 이라는 디렉토리가 생성된다.
cd ZendOptimizer-2.1.0a-Linux_glibc21-i386 로 이동한다.
./install 이란 명령을 실행시키면 다음과 같은 창이 뜬다.



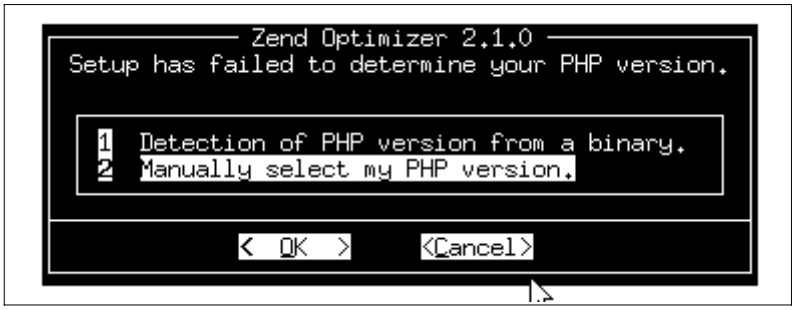
< 그림 12-2 ZendOptimizer install 실행 화면 >

< 그림 12-3 ZendOptimizer의 설치 경로 >



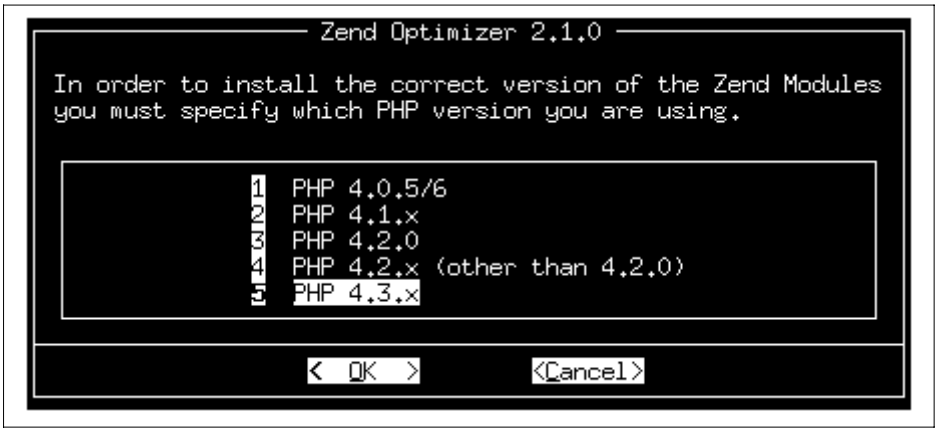
< 그림 12-4 php.ini의 경로 설정 >

다음 아파치의 사용하느냐, conf의 경로는 어디냐, apache의 bin 디렉토리 경로는 어디냐 등등을 묻는데 모두 OK를 선택하면 된다.



< 그림 12-5 php의 버전 설정 >

설치된 php의 버전을 선택하는 부분이다. 1번자동으로 검색을 선택하면 못찾는 경우도 발생하므로, 2번 수동으로 버전 선택을 선택한다.



<그림 12-6 php 버전 수동 선택 >

php 버전을 수동으로 선택한다. 여기서 설치한 것은 php-4.3.2 이므로 5번 4.3.x를 선택한다.

OK를 선택해서 다음으로 넘어가면 설정이 모두 되었다는 멘트창이 뜨며, apache를 재시작 할 것인지 묻는 창이 뜬다. 설치된 Zend를 사용하기 위해서는 재시작을 해줘야 하므로 Yes를 누른다.
잠시 기다린 후에 “Apache was successfully restarted” 되었다는 창이 뜨면 설정이 끝난 것이다.

설정이 끝난후에 php.ini파일이 /usr/local/Zend/etc/php.ini에 생성이 되며, /etc 의 php.ini는 /usr/local/Zend/etc/php.ini 의 심볼릭 링크로 설정이 된다.
그리고 **cp /usr/local/Zend/etc/php.ini /usr/local/php4/lib/php.ini** 로 복사를 해준다.
그리고 나서 apache를 재가동하면 php 설정이 갱신되어 읽혀진다.

웹페이지를 확인 할 수 있는 곳에 다음과 같은 php 코드를 입력하여 test.php란 이름으로 저장하고 웹에서 확인하면 php 및 각종 옵션의 상태와 ZendOptimizer가 정상적으로 작동을 하는지 확인할수 있다.

```
<?
        phpinfo()
?>
```

Zend Optimizer

Optimization Pass 1	enabled
Optimization Pass 2	enabled
Optimization Pass 3	enabled
Optimization Pass 9	disabled
Optimization Pass 10	disabled
Zend Loader	enabled
License Path	*no value*

< 그림 12-7 ZendOptimizer 의 설정 여부 확인 >

12. 문제 해결 (Trouble Shooting)

이번 장에서는 서버를 운영하면서 만나게 되는 장애들 중에서 호스트웨이의 온라인 고객지원과, 개인적으로 질문받는 것들중에서 대표적인 것들에 대한 해결책을 제시하고자 한다.

1. 웹 속도가 갑자기 느려졌습니다.

→ 웹 속도가 느려졌을 때 문제를 해결해 보기 위해서는 다음과 같은 방법들을 사용해 보면 됩니다

1) 현재 사용하는 컴퓨터에서 느려지는 해당 서버로 ping을 보내본다.
(time<10ms이면 지극히 정상이다. 네트워크 쪽으로는 문제가 없는 것이다.)

2) 아파치 웹로그가 많이 쌓여 있는지 체크한다.
(아파치의 웹로그는 최대 생성할 수 있는 크기가 2G이며, 웹 로그의 크기가 커질수록 로그파일을 읽어서 제일 하단에 로그를 쌓아야 하므로 로그파일이 클 수로 웹 접속속도가 떨어질 수밖에 없다.)

3) 현재 80번 포트를 사용해서 웹에 접속중인 사용자들의 현황을 살펴본다.
netstat -nap | grep :80 | grep EST | wc -l 이란 명령을 사용하면 현재 웹에 접속되어 있는 프로세서 수를 확인할 수 있다. 이 수가 httpd.conf에서 설정한 Maxclient와 가깝거나 혹은 더 많지는 않은지 살펴본다.

4) top 명령으로 현재 cpu의 사용률이나, 메모리 사용량등을 체크해본다.

2. 웹이 뜨지 않습니다.

→ 대부분의 경우는 apache 데몬이 다운된 것이 원인이 될 수 있다.
/usr/local/apache/bin/apachec시 [start|stop|restart] 명령을 사용하여 재가동 하면 된다.

3. mysql에 접속되지 않습니다.

→ mysql 데몬이 떠있는지 확인하는 것이 중요하다.
ps -aef | grep mysql 이란 명령으로 mysql데몬이 떠있는지 확인하시고, 만약 떠있지 않다면
/usr/local/mysql/bin/safe_mysqld & 명령으로 재가동 한다.
과거에 설치된 소스의 경우에는
cd /usr/local/mysql
bin/safe_mysqld & 라고 해야 mysql이 정상 부팅 되는 것도 있으므로 참고하시기 바랍니다.
만약 데몬이 동작하고 있는데 제대로 연결이 안 될때는 mysql데몬을 내렸다가 (pkill mysql)
재 가동 하시면 됩니다. (이때는 apache데몬이 mysql과 같이 연동되어 제대로 종료가 안될 수도 있으므로 apache와 mysql 동시에 다운시켰다가 재 가동해야 합니다.)

4. mysql의 table이 깨졌다는 메시지가 출력됩니다.

→ mysql의 table이 깨졌다고 나오는 것은 mysql데이터가 뒤섞여서 일어나는 일이 많습니다.
이때에는 myisamchk 라는 명령으로 복구를 해주시면 됩니다.
즉, /usr/local/mysql/data/columns_priv.MYI(D) 라는 것이 table이 깨졌다는 메시지가 출력된다면,

/usr/local/mysql/bin/myisamchk -r /usr/local/mysql/data/columns_priv 라고 해주면 깨진 table 을 정상적으로 복구 시켜 주게 됩니다. 이때 먼저 mysql을 중지시키고 하시는 것이 안전합니다.

복구를 시켰는데도 불구하고 복구 시킬당시에는 정상작동 되다가 얼마후에 또다시 깨지는 일이 발생 한다면 그것은 디스크 공간을 의심해 볼 필요가 있습니다.
df 명령으로 파티션 정보를 출력하여 mysql data가 들어있는 /usr의 파티션 용량이 100%는 아닌지 확인해 주시기 바랍니다.

5. mysql 의 root 패스워드를 분실 했습니다.

→ 이럴때에는 우선 mysql 데몬을 중지 시킵니다.
그리고 나서 /usr/local/mysql/bin/safe_mysqld -Sg & 옵션으로 부팅을 하면 루트 패스워드 없이 mysql에 접속할수 있습니다. 접속후 root패스워드를 변경해주시면 됩니다.
(패스워드 변경 update user set password=password('패스워드') where user ="root"; 라고 해주시면 됩니다. 이것은 mysql db에 접속해서 내리는 쿼리문입니다.)

6. 하드 파티션 포맷

→ 하드를 추가 장작이나 fdisk로 나눈후에는 파일시스템을 생성해 주어야 합니다.
이 과정은 windows 시스템의 format과 같은 과정입니다.
우선 파일을 포맷할 시스템이 /dev/hdb라고 가정했을때 다음과 같은 명령을 내려주면 됩니다.
mkfs -j /dev/hdb (이것은 ext3파일 시스템으로 하드디스크를 포맷하는 명령입니다.)

7. IP로는 웹에 접속 가능한데 domain으로는 웹에 접속이 안됩니다.

→ 이것은 DNS가 제대로 서비스를 해주지 못하는 경우에 발생을 합니다.
도메인으로 입력을 하면 DNS에서는 이것을 IP로 변경하여 주는역활을 하는 것인데 이 동작이 제대로 수행이 되지 않아서 발생하는 문제입니다. 이때에는 현재 운영하는 도메인의 네임서버가 어디로 잡혀있는지 확인 후에 DNS설정이 되어있는곳에 문의를 하시면 됩니다. 도메인의 네임서버가 호스트웨이의 네임서버일 경우에는 호스트웨이에 문의하시면 됩니다.

8. 가상 호스팅 설정

→ 가상 호스팅이란 하나의 IP를 가진 서버에서 여러 개의 multi도메인을 운영할 수 있는 방법이다.
설정은 다음과 같다.
NameVirtualHost 211.239.151.21 (서버의 IP - IP기반 가상 호스팅 이름기반 가상호스팅을 한다는 설정이므로 **최초 한번만 설정**해준다)

```
<VirtualHost 211.239.151.21>
    ServerAdmin zmnkh@manpage.co.kr
    DocumentRoot /home/manpage/public_html
    ServerName manpage.co.kr
    ServerAlias www.manpage.co.kr
    ErrorLog logs/dummy-host.example.com-error_log
    CustomLog logs/dummy-host.example.com-access_log common
</VirtualHost>
```

위와 같은 설정을 필요한 도메인 수만큼 반복하여, 작성해 주면 된다.
<VirtualHost> ~ </VirtualHost> 의 내용을 반복!!
(물론 도메인과 홈페이지의 DocumentRoot는 바꿔줘야 한다.)

9. user를 생성할 때마다 자동으로 계정 디렉토리 안에 public_html을 디렉토리를 생성하고자 합니다.
→ **/etc/skel** 이란 디렉토리로 이동하셔서 그안에 public_html 이라는 디렉토리를 생성해 놓으시면 user를 생성할때, 그 안의 설정값을 읽어와서 계정생성에 참고하게 되므로 자동으로 public_html이 생성되게 됩니다.

10. 계정생성시 마다 자동으로 디렉토리 권한을 701로 주고 싶습니다.
→ 계정을 생성하게 되면 자동으로 700의 권한을 가지도록 생성되어 있습니다.
그것을 깜박하고 가상 호스팅을 설정하여 웹에서 그 디렉토리를 보게되면, permission deny가 표시되게 됩니다. 이 때문에 다시 한번 서버로 접속하여, 해당 디렉토리의 권한을 수정하게 되는데 계정을 생성할 때마다 자동으로 701의 권한을 주고 싶으면 (다른 사용자들이 디렉토리를 열람할 수 없도록
웹에서 볼 수 있는 최소권한인 701만을 주는 것이 바람직 하다)
/etc/login.defs을 열어서 제일 하단에 **UMASK [tab] 076**을 써주게 되면 계정생성시 마다 자동으로 701을 권한을 부여하게 된다. (tab은 사이를 tab으로 띄우라는 의미이다.)

11. 서버의 시간이 틀립니다.
→ 서버의 시간은 여러 가지 이유로 변동이 될 수 있습니다. 다음과 같은 명령으로 해결이 가능합니다. **rdate -s time.nuri.net && clock -w** (하드웨어의 BIOS 시간까지 변경하라는 명령입니다.)

12. vi로 파일을 열었는데 이미 열려진 파일이라고 에러화면이 나옵니다.
→ 가끔 httpd.conf등 자주 열어보는 파일을 vi로 편집하고자 열어볼 때, "이미 열린 파일이다. 읽기 모드로 읽겠느냐, 아니면 빠져나가겠느냐, 그냥 진행하겠느냐" 등으로 메시지가 나올때가 있습니다. 이는 누가 이미 httpd.conf 파일을 열었거나 과거에 httpd.conf를 열어보고 잘못된 종료를 하여 생성된 **swap** 파일이 지워지지 않아서 생기는 문제입니다.
이럴때에는 ls -al 명령으로 해당 디렉토리를 보게되면 (httpd.conf파일의 경우) .httpd.conf.swp 라는 파일이 존재하는 것을 확인할수 있을 것이다. 파일을 편집기로 열어보면 자동적으로 **.swp** 파일이 생성이 되는데 종료를 잘못할 경우에 이 파일이 사라지지 않고 남아있게 된다. 이 파일을 지워주게 되면 질문과 같은 에러는 나오지 않는다.

13. 웹에서 http://도메인/~계정으로 접근하고 싶습니다.
→ vi /usr/local/apache/conf/httpd.conf 명령으로 httpd.conf를 오픈하여
다음과 같은 항목이 있는지 확인해본다. 주석으로 잠겨있으면 안됨!!

```
<IfModule mod_userdir.c>
        UserDir public_html
</IfModule>
```

그리고 몇라인 아래에 다음과 같은 디렉토리 옵션의 주석을 모두 제거해준다.

```
==================
<Directory /home/*/public_html>
    AllowOverride FileInfo AuthConfig Limit
    Options MultiViews Indexes SymLinksIfOwnerMatch IncludesNoExec
    <Limit GET POST OPTIONS PROPFIND>
        Order allow,deny
        Allow from all
    </Limit>
    <LimitExcept GET POST OPTIONS PROPFIND>
        Order deny,allow
        Deny from all
    </LimitExcept>
</Directory>
==================
```

이때 **Indexes 란** 옵션은 반드시 제거한다.

변경한후에 설정을 저장하고 나와서 /usr/local/apache/bin/apachectl restart를 해주게 되면 적용이 된다.
단, 이때 선행조건으로는 웹에서 보려고 하는 **계정의 권한은 최소한 701(chmod 701 계정명)**이 되어야 하며, 계**정안에는 public_html이란 디렉토리가 존재**해야 한다. 그리고 public_html안에는 **index.php, index.html**등의 처음 불러올수 있는 파일이 있었야 웹에서 오류가 나지 않고 정상적으로 확인이 가능하다.

14. [보안 경고] Indexes 옵션을 제거 하십시오

→ **/usr/local/apache/conf/httpd.conf**의 설 정중 디렉토리 옵션으로 Indexes라는 옵션이 있습니다. 이는 해당 디렉토리의 내용을 windows 의 디렉토리처럼 웹에서 보여질 수 있도록 하는 옵션입니다. 최근 이 옵션으로 서버내의 설정파일이나 내용을 확인하거나 외부로 유출하는등의 해킹시도가 부쩍 늘어나고 있습니다. 이에 이 **Indexes**옵션은 제거 하여 주시기 바랍니다.

15. [보안 경고] apache 패키지 업그레이드

→ 최근 일어나는 해킹 사례들을 보면 취약한 아파치 패키지를 노려서 해킹을 하는 사례가 늘고 있습니다. 실제로도 몇건의 해킹이 그런 취약점을 통해서 발생을 하였습니다.
RPM으로 설치된 apache를 패키지를 사용하고 있는 관리자 분들은 최신의 apache버전으로 업그레이드 해주시기 바랍니다.
만일 업그레이드가 되지 않을시 에는 다음과 같은 방법으로 어느정도의 보완을 할 수는 있습니다.
/etc/fstab 을 vi로 열어서
LABEL=/tmp /tmp ext3 defaults 1 2
이라고 되어있는 부분을

LABEL=/tmp /tmp ext3 defaults,nosuid,noexec 1 2
라고 추가 해주신 후에 저장하고 나와서 mount -oremount /tmp 라고 해주시면 다시 갱신이 됩니다. (혹은 reboot)

이렇게 해줌으로써 tmp파일에는 실행파일이나, setuid가 붙은 파일은 생성할수 없게 되어 /tmp의 777 취약점을 이용한 공격은 방어할수 있게 됩니다.
하지만 이것이 모든것을 막아줄수 있는 것은 아니며, 최신의 패키지로 업그레이드 하는 것이 가장 좋은 방법입니다.

만약 mysql.sock등과 같은 setuid가 필요한 파일이 /tmp에 존재해야 할 경우 에는 /etc/my.cnf 파일을 만든 후에
####################
[mysqld]
socket=/var/run/mysql.sock
####################
라고 적어주신 후에 chmod 775 /var/run, chgrp mysql /var/run
해주시면 /var/run 밑에 mysql.sock이 생기므로 /tmp에 setuid가 필요없게 됩니다. (mysql restart가 필요 합니다.)

해킹에 대비하기 위해서는 항상 백업을 습관화 하셔야 하며, 취약한 패키지나 사용하시는 OS의 취약점이 발견되면 그 즉시 패치를 하셔야 합니다.

16. 해킹에 대한 대비

→ 아무리 철벽 보안을 한다고 해도 마음먹고 들어오는 해커에게는 당해낼 재간이 없다. 오로지 빠른 대처와 복구, 루트패스워드의 변경, 시스템의 상태를 꾸준하게 검사하는 것많이 해커를 막을수 있는 길이다. 즉 관리자가 부지런해야 해커에 대한 공격을 막을수 있는것이다.

시스템은 운영중이다 갑자기 프로세서의 반응이 느려지고, 서버가 느려진다는 느낌을 받았을 때 시스템의 프로세서나 메모리가 과도하게 움직일때, 보지못한 불법 프로세서들이 작동하고 있을때에는 해킹을 의심해 봐야 한다.

간단하게 체크해 보는 방법은 다음과 같다.

- find /dev -type f (/dev 디렉토리 및에는 MAKEDEV 외에는 일반 파일이 존재 하지 않으므로 이 명령을 내렸을 때, MAKEDEV 말고 다른 파일들이 나온다면 해킹을 의심해 봐야 한다.)

- /tmp (/tmp 디렉토리를 확인해 본다. /tmp 디렉토리는 777 권한으로 누구든지 읽고 쓸수가 있으므로 이곳에 불법적인 실행파일이나 압축파일 혹은 못보던 디렉토리가 생성되지는 않았는지 확인한다.)

- find / -perm -4000 -print > perm.txt 라는 파일을 평소에 만들어 두고, 만일 해킹이 의심이 되면. find / -perm -4000 -print | more 명령을 내려서 기존에 만들어 두었던 perm.txt 라는 파일과 대조해보아 setuid 파일이 생성되지는 않았는지 체크한다.

- vi /etc/passwd 파일을 체크해 보아 불법적으로 생성된 유저는 없는지 확인한다. 그리고 uid와 pid를 검사하여 일반 유저가 슈퍼유저의 권한을 가지고 있지는 않은지 체크한다.

- 기타 소유권이 없는 파일이나 "..." 등으로 명시된 디렉토리, 혹은 이름이 붙어 있지 않는(공백) 디렉토리등이 있는지 확인해본다.

< 용 어 >

- Daemon (데몬)
  클라이언트의 요청에 부합하기 위해서 서버에 설치하는 서버 프로그램을 말한다. httpd, sshd 등의 끝에 d 가 붙은 이름은 데몬임을 의미한다.

- SMTP(Simple Mail Transport Protocol)
  인터넷 전자우편 시스템 사이에서 전자우편 메시지를 교환하기 위해 사용되는 어플리케이션 단계의 클라이언트 / 서버 프로토콜을 말하는 것으로 시스템 간의 전자우편 메시지를 전송하며, 수신 전자우편에 대한 통보 기능을 제공한다.

- Spam (스팸)
  전자 매체를 통해 무차별적으로 전송 되는 원하지 않는 메시지. 전자우편이나 뉴스그룹 게시물을 필요이상으로 대량 전달 함으로써 시스템의 효율성을 떨어뜨리고, 정보의 질을 낮추는 행위를 말한다.

- ISP (Internet Service Provider)
  고객을 인터넷에 연결해 주는 서비스를 제공하는 회사.

- MIME(Multipurpose Internet Multimedia Extensions)
  전자우편 메시지에 들어 있는 파일, 그림, 소리 그 이외의 문자가 아닌 내용들을 사용하는 것에 대한 규약입니다. 첨부 파일이라고 불리워지며 어떠한 종류의 데이터라도 실질적으로 인터넷상의 다른 사용자들과 교환할 수 있게 합니다.

- bps (bits per second)
  초당 비트 수, 1초동안에 어느정도 송수신 할수 있는가를 나타내는 것.

- resolve : 도메인을 컴퓨터가 인식할수 있는 IP로 변환하는 것.

- inverse-resolve : resolve와 반대로 IP를 역추적하여 도메인으로 변환 하는것.

- revelse-domain : IP를 nslookup 했을때 연결된 도메인을 확인할수 있도록 설정한 것.