DNS HOWTO

Nicolai Langfeldt janl@math.uio.no v2.1.1, 12 November 1998

이승규 hanuel@edunet.kmec.net, 이운억wulee@nownuri.net v.2.0 1998년 3월 13일

이 HOWTO는 시간을 적게 투자하여 편하게 DNS를 관리하는 방법을 설명한다.

목 차

제 1 절 머리말

검색어: DNS, bind, bind-4, bind-8, named, dialup, ppp, slip, isdn, Internet, domain, name, hosts, resolving

1.1 Legal stuff

1.2 도움에 감사드리며..., 도움을 부탁하며...

이 문서 초안을 수도 없이 읽어 주고 많은 제안을 해준 Arnt Gulbrandsen 씨에게 감사의 말을 전하는 것이 순서에 맞을 것 같다. 또한 e-mail로 의견과 유용한 내용을 보낸 준 사람들에게도 감사의 말을 전한다.

이 문서는 완결된 문서가 아니다. DNS를 설정하여 사용할 때 발생하는 문제점이나 그에 따른 해결책들이 있을 것이다. 그러한 내용들은 e-mail로 보내 준다면, 다음 번에는 더 좋은 DNS-HOWTO가 나올 수 있을 것이다. money나 의견 또는 의문점은 janl@math.uio.no 앞으로 보내 주길 바란다. e-mail을 보내기 전에 *반드시* 자신의 e-mail 주소가 올바른지 확인하도록 한다. 그래야 답신을 받을 수 있다는 것은 당연한 이야기 일 것이다. 또한 메일을 보내기 전에 8(질문과 답) 절을 읽어 보기 바란다.

이 HOWTO를 번역하고자 한다면, 나에게 알려 주기 바란다. 그러면, 어떤 언어로 번역이 되었는지 정리해 둘 수 있을 것이며 이 HOWTO가 개정될 때 알려 줄 수 있을 것이다.

역자의 말 : 한글 판에 문제가 있거나 오역이 있으면 haneul@edunet.kmec.net으로 메일을 보내주길 바랍니다.

1.3 Anne Line Norheim Langfeldt에 바치며

이 HOWTO를 Anne Line Norheim Langfeldt에 바친다. 그녀는 이 문서를 읽어 보지도 않겠지만, 그녀는 정말로 특별한 여자이다.

제 2 절 소개

DNS에 대하여.

DNS는 Domain Name System의 약자이다. 즉 DNS는 컴퓨터 이름과 IP 주소를 상호 변환시킨다. 즉, 이름을 주소로, 주소를 이름으로 변환한다. 이 HOWTO에서는 이름과 주소 사이의 그러한 매핑을 리눅스에서 정의하여 서비스하는 방법을 설명한다. 매핑이란 아주 단순한 것으로 이름과 주소를 서로 결합시켜 둔 것에 지나지 않는다. 즉, ftp.linux.org와 같은 이름과 199.249.150.4와 같이 숫자로 된 IP를 연결시키는 것이다.

초보자(여러분 ;-)에게 DNS는 네트워크 관리 영역 중에서도 꽤 모호한 편에 속한다. 이 HOWTO에서는 DNS에 관한 몇 가지 주제를 명확하게 전달할 것이다. 즉, *simple* DNS를 설정하는 방법들을 설명할 것이다. 우선 캐시 전용 서버(Caching Only Server) 설정 방법을 설명한 후에 1차 DNS(Primary DNS)를 설정하여 특정 도메인의 Name Resolving 서비스를 제공하는 방법을 설명할 것이다. 보다 자세한 설명이 필요한 경우에는 이 문서에서 8(질문과 답) 절을 참고하기 바란다. 8(질문과 답) 절에 필요한 내용이 없으면, *참고 문헌*을 읽어 보도록 한다. 참고 문헌은 이 문서의 9(마지막 절)에 수록해 두었다.

우선 컴퓨터를 설정하여 Telnet으로 접속 가능하게 하고, 필요한 네트워크 응용 프로그램들이 올바로 작동할 수 있도록 컴퓨터의 네트워크 환경을 설정한 다음, `telnet 127.0.0.1` 명령을 입력하여 자기 자신에게 접속이 되는지 확인한다.(당장 확인!!) 그리고 /etc/nsswitch.conf (또는 /etc/host.conf), /etc/resolv.conf, /etc/hosts 파일이 올바르게 설정되어 있어야 한다. 이러한 파일들을 설정한 적이 없다면, NET-3-HOWTO와 PPP-HOWTO에 이 파일들을 설정하는 방법이 설명되어 있으니, 이러한 파일들을 설정한 적이 없다면, NET-3-HOWTO와 PPP-HOWTO를 읽어 보도록 한다.

아무런 수식어 없이 ‘컴퓨터’라고 이야기를 할 때는 DNS로 사용할 컴퓨터를 뜻하는 것이다. 기타 다른 컴퓨터를 뜻하는 말이 아니므로 혼동하지 않도록 한다.

이 문서에서는 기본적으로 컴퓨터가 방화벽 바깥쪽에 있기 때문에 방화벽에 관련된 문제가 발생하지 않는다고 가정한다. 방화벽 관련 설정이 필요한 경우에는 8(질문과 답) 절을 참고한다.

Unix에서 DNS 프로그램은 named라고 하는 프로그램이다. 이 프로그램은 Internet Software Consortium의 멤버인 Paul Vixie 씨가 만든 bind 패키지의 일부분이다. 리눅스 배포판에는 대부분 Named가 포함되어 있으며 /usr/sbin/named라는 이름으로 설치된다. 현재 컴퓨터에 named가 있으면, 그냥 사용하면 되지만, 컴퓨터에 named가 없는 경우에는 리눅스 ftp 사이트에서 바이너리를 구해서 사용할 수 있다. named의 소스는 ftp.isc.org:/isc/bind/src/cur/bind-8/에서 구할 수 있다. 이 HOWTO는 bind 버전 8을 사용하는 것을 전제로 하고 있다. bind 4에 대한 이전 버전 HOWTO는 http://www.math.uio.no/\~{}janl/DNS/에서 구할 수 있다. named 맨페이지에서 named.conf 파일을 언급하면, bind 8이다. 그렇지 않고 named.boot에 대해서 언급하면 bind 4이다. bind 4인 경우에는 보안 문제가 있으므로 bind 8로 업그레이드하기 바란다.

DNS는 네트워크 전반에 넓게 분산된 데이터베이스다. 그러므로 새로운 항목을 추가할 때는 신중해야 한다. 엉터리로 된 항목을 추가하면 그 도메인에 접속하는 모든 사용자들이 엉터리 주소를 사용하게 된다. DNS를 잘 정돈하고 일관성 있게 운영하면 좋은 결과를 얻을 것이다. 사용 방법, 관리 방법, 디버그 방법을 배워라. 그러면 네트워크를 잘못된 관리 때문에 발생할 수 있는 오버로드를 미연에 방지하여 네트워크를 훌륭히 관리할 수 있을 것이다.

이 문서에서 완전한 사실이 아닌 사항도 사실인 것처럼 이야기 한다. (적어도 반 정도는 사실이다). 단순하게 설명하기 위해서이다. 이 문서에서 말하는 것들을 믿는다면 (아마도 ;-) 모든 것은 제대로 될 것이다.

Tip: 편집해야 하는 파일들을 모두 백업해 두도록 한다. 그래야만 제대로 작동하지 않을 때 원래대로 복구하기가 용이하다.

제 3 절 캐시 전용 네임 서버(Caching only name server)

DNS 설정의 첫 단계로 다이얼업 사용자에게 매우 유용하다.

캐시 전용 네임 서버(Caching only name server)는 네임 쿼리의 응답을 찾은 후 기억해 두었다가 다음 번에 필요할 때 곧 바로 응답한다. 특히, 접속 회선이 느린 경우에는 기다리는 시간을 상당히 줄여 줄 것이다.

우선 /etc/named.conf 파일이 필요하다. named가 시작하면서 이 파일을 읽어 들인다. 당장은 단순히 아래와 같이 편집하도록 하자.

```
// Config file for caching only name server

options {
        directory "/var/named";

        // Uncommenting this might help if you have to go through a
        // firewall and things are not working out:

        // query-source port 53;
};

zone "." {
        type hint;
        file "root.hints";
};

zone "0.0.127.in-addr.arpa" {
```

```
        type master;
        file "pz/127.0.0";
};
```

'directory'는 파일을 어디에서 찾아야 하는지 named에게 알려 준다. 이후 나오는 파일들은 모두 이 디렉토리에 대한 상대 경로이다. 그러므로 pz는 /var/named 디렉토리의 하위 디렉토리이다. 즉, /var/named/pz이다. /var/named는 *Linux File system Standard*에 명시된 디렉토리이다.

/var/named/root.hints라는 파일의 이름을 여기에 적어 준다. /var/named/root.hints 파일의 내용은 다음과 같다.

```
.                        6D IN NS        G.ROOT-SERVERS.NET.
.                        6D IN NS        J.ROOT-SERVERS.NET.
.                        6D IN NS        K.ROOT-SERVERS.NET.
.                        6D IN NS        L.ROOT-SERVERS.NET.
.                        6D IN NS        M.ROOT-SERVERS.NET.
.                        6D IN NS        A.ROOT-SERVERS.NET.
.                        6D IN NS        H.ROOT-SERVERS.NET.
.                        6D IN NS        B.ROOT-SERVERS.NET.
.                        6D IN NS        C.ROOT-SERVERS.NET.
.                        6D IN NS        D.ROOT-SERVERS.NET.
.                        6D IN NS        E.ROOT-SERVERS.NET.
.                        6D IN NS        I.ROOT-SERVERS.NET.
.                        6D IN NS        F.ROOT-SERVERS.NET.

G.ROOT-SERVERS.NET.      5w6d16h IN A    192.112.36.4
J.ROOT-SERVERS.NET.      5w6d16h IN A    198.41.0.10
K.ROOT-SERVERS.NET.      5w6d16h IN A    193.0.14.129
L.ROOT-SERVERS.NET.      5w6d16h IN A    198.32.64.12
M.ROOT-SERVERS.NET.      5w6d16h IN A    202.12.27.33
A.ROOT-SERVERS.NET.      5w6d16h IN A    198.41.0.4
H.ROOT-SERVERS.NET.      5w6d16h IN A    128.63.2.53
B.ROOT-SERVERS.NET.      5w6d16h IN A    128.9.0.107
C.ROOT-SERVERS.NET.      5w6d16h IN A    192.33.4.12
D.ROOT-SERVERS.NET.      5w6d16h IN A    128.8.10.90
E.ROOT-SERVERS.NET.      5w6d16h IN A    192.203.230.10
I.ROOT-SERVERS.NET.      5w6d16h IN A    192.36.148.17
F.ROOT-SERVERS.NET.      5w6d16h IN A    192.5.5.241
```

이 파일은 인터넷의 루트 네임 서버들을 지정하고 있다. 바뀌는 경우가 있으므로 *잘 관리하여야 한다*. 최신으로 유지하는 방법은 6(유지 보수 절)를 참고한다.

그 다음은 이 파일의 마지막 존(zone)이다. 사용법은 다음 장에서 설명하기로 하고 지금은 그냥 pz 디렉토리에 127.0.0 파일을 만든다.

```
@               IN      SOA     ns.linux.bogus. hostmaster.linux.bogus. (
                                1       ; Serial
                                8H      ; Refresh
                                2H      ; Retry
                                1W      ; Expire
                                1D)     ; Minimum TTL
```

```
                         NS      ns.linux.bogus.
1                        PTR     localhost.
```

/etc/resolv.conf 파일을 다음과 같이 편집한다.

```
search subdomain.your-domain.edu your-domain.edu
nameserver 127.0.0.1
```

'search'는 사용자가 호스트 명만 입력한 경우에 검색할 도메인을 지정한다. 'nameserver'는 사용할 네임 서버를 나타낸다. 이 경우에는 네임서버를 직접 운영하므로 사용자 컴퓨터의 IP 주소를 적는다. (127.0.0.1을 적어 주면 문제가 없다. 사용자 컴퓨터에 다른 IP 주소가 없는 경우에도 문제없이 작동한다.) 네임 서버를 여러 개 사용하려는 경우에는 'nameserver' 라인을 여러 라인 두면 된다. (참고:Named는 이 파일을 읽지 않는다. named를 이용하는 resolver가 이 파일을 사용한다.)

이 파일의 작동 방식: 클라이언트가 foo를 조회하는 경우 먼저 foo.subdomain.your-domain.edu를 찾는다. 다음으로 foo.your-fomain.edu를 찾고 마지막으로 foo를 찾는다. 클라이언트가 sunsite.unc.edu를 조회하는 경우에는 먼저 sunsite.unc.edu.subdomain.your-domain.edu을 찾는다. (물론 멍청하긴 하지만 이렇게 동작한다.) 다음으로 sunsite.unc.edu.your-domain.edu를 찾고 마지막으로 sunsite.unc.edu를 찾는다. search 라인에 도메인이 너무 많은면 검색 시간이 꽤 길어지게 되므로 search에 도메인을 많이 두지 않는 것이 좋다.

이 예에서 사용자가 속한 도메인이 subdomain.your-domain.edu라고 가정한다. 그러면 사용자의 컴퓨터는 your-machine.subdomain.your-domain.edu가 될 것이다. 'search' 라인에 자신이 속한 도메인의 TLD(Top Level Domain, 여기서는 'edu')이 포함되지 않도록 주의한다. 다른 도메인에 속한 호스트에 자주 접속을 한다면 다음처럼 'search'에 계속해서 추가하는 것도 나쁘지는 않다.

```
search subdomain.your-domain.edu your-domain.edu other-domain.com
```

예와 똑같이 설정하지 말고 각자 실제로 사용하는 도메인 명을 명시한다. 도메인 명의 끝에 점(period)이 없다는 것에도 유의한다.

다음 단계는 libc의 버전에 따라 약간 달라지는데, /etc/nsswitch.conf 또는 /etc/host.conf 파일을 편집한다. 복잡하게 생각할 것 없이 현재 컴퓨터에 nsswitch.conf라는 파일이 있으면 그 파일을 편집하고, 없으면 host.conf 파일을 편집한다.

/etc/nsswitch.conf

이 파일은 약간 긴 파일로 어떤 파일이나 데이타베이스로부터 여러 종류의 정보(호스트 명, 암호, 쉐도우 암호, 그룹 정보, 알리아스 정보 등)를 얻어 와서 사용할 것인지를 지정한다. 보통 파일 시작 부분에 도움말이 있으므로 읽어 보면 편집하는 데에 도움이 된다. 지금 당장 읽어 보기 바란다. 'hosts:로 시작하는 라인을 찾아 보자. 다음과 같은 라인이 있으면 정상이다.

```
hosts:      files dns
```

'hosts:'로 시작하는 라인이 없는 경우에는 위와 같이 추가하도록 한다. 프로그램이 주소를 조회할 때 먼저 /etc/hosts 파일을 검사하고 그 파일에서 찾지 못하는 경우 resolv.conf에 명시된 DNS에서 주소를 찾는다.

/etc/host.conf

이 파일은 보통 여러 라인으로 구성되는데, order로 시작하는 라인이 있어야 한다. 일반적으로는 아래와 같다면 정상이다.

```
order hosts,bind
```

'order'로 시작하는 라인이 없는 경우에는 위의 라인을 삽입한다. 먼저 /etc/hosts 파일을 찾아 보고 없으면 네임 서버(resolv.conf 파일에서 127.0.0.1로 지정하였다)에서 주소를 찾는다는 의미이다. 리눅스 배포판에는 대부분 이 두 파일을 resolv(8) 맨페이지('man 8 resolv'를 실행해 본다.)에서 설명하고 있다. That man page is IMHO readable, and everyone, especially DNS admins, should read it. Do it now, if you say to yourself "I'll do it later" you'll never get around to it.

3.1 named 실행

이제 named를 싱행하면 된다. 전화 접속 사용자인 경우에는 우선 전화를 걸어 접속을 하도록 한다. 'ndc start'를 입력하고 엔터를 누른다. 다른 옵션은 필요 없다. 잘 안돼면 '/usr/sbin/ndc start'를 실행한다. 그래도 이상하다면 8() 절을 참고한다. 이제 정상적으로 작동하는지 시험해 보자. named가 시작하는 동안 message 파일의 내용을 살펴 보자. 보통 messages 파일은 /var/adm/messages 파일이지만, 경로가 /var/log인 경우가 있으며, 파일명이 syslog인 경우도 있다. tail -f /var/log/messages 명령으로 내용을 확인할 수 있는데, 확인 결과가 아래와 같다면 정상이다.

('\'는 다음줄과 연결 되었음을 뜻한다.)

```
Feb 15 01:26:17 roke named[6091]: starting.  named 8.1.1 Sat Feb 14 \
  00:18:20 MET 1998 ^Ijanl@roke.uio.no:/var/tmp/bind-8.1.1/src/bin/named
Feb 15 01:26:17 roke named[6091]: cache zone "" (IN) loaded (serial 0)
Feb 15 01:26:17 roke named[6091]: master zone "0.0.127.in-addr.arpa" \
  (IN) loaded (serial 1)
Feb 15 01:26:17 roke named[6091]: listening [127.0.0.1].53 (lo)
Feb 15 01:26:17 roke named[6091]: listening [129.240.230.92].53 (ippp0)
Feb 15 01:26:17 roke named[6091]: Forwarding source address is [0.0.0.0].1040
Feb 15 01:26:17 roke named[6092]: Ready to answer queries.
```

에러 메시지가 보이면 중간에 뭔가 실수가 있다는 뜻이다. 설정할 때 실수한 파일명(named.conf나 root.hints일 것이다)을 named가 보여 줄 것이다. named를 죽이고 그 파일을 점검한다.

이제 nslookup으로 named가 정상적으로 작동하는지 점검할 차례이다.

```
$ nslookup
Default Server:  localhost
Address:  127.0.0.1

>
```

위와 같다면 제대로 된 것이다. 그러기를 바란다. 그렇지 않다면 처음부터 다시 검사한다. named.conf를 수정할 때마다 ndc restart 명령으로 named를 재시작시켜야 한다.

이제 쿼리를 입력할 수 있다. 근처에 있는 컴퓨터를 찾아 보자. Oslo 대학에 있는 pat.uio.no가 저자에게는 pat.uio.no가 가깝다.

```
> pat.uio.no
Server:  localhost
Address:  127.0.0.1

Name:    pat.uio.no
Address:  129.240.130.16
```

nslookup이 여러분이 설정한 named에게 `pat.uio.no` 컴퓨터를 찾도록 요청했다. 그래서 named는 `root.hints` 파일에 있는 네임 서버 중 하나에 접속한 후 그 응답을 받았다. `/etc/resolv.conf`에 써 넣은 도메인들을 모두 검색하기 때문에 그만큼 시간이 걸릴 것이다.

똑같은 요청을 다시 한다면 다음 처럼 보일 것이다.

```
> pat.uio.no
Server:  localhost
Address:  127.0.0.1

Non-authoritative answer:
Name:    pat.uio.no
Address:  129.240.2.50
```

'`Non-authoritative answer:`' 라인에 유의하자. 이 라인은 외부로 나가지 않고 대신 캐시를 검사하여 찾아왔음을 뜻한다. 그러나 캐시에 남아 있는 정보는 *오래되어* 실제로는 변경된 경우도 있다. 그래서 경고의 뜻으로 '`Non-authorative answer:`'를 보여 준다. 어떤 호스트에 대해 두 번째 질의했을 때 `nslookup`이 이 메시지를 보여 준다면, named가 정보를 캐시에 저장하였다가 사용한다는 뜻이다. 즉, 정상적으로 작동한다는 뜻이다. '`exit`을 입력하여 `nslookup`을 종료한다.

이제 캐시 전용 DNS(Caching Only DNS) 설정 방법을 알았다. 자축하는 뜻으로 맥주나 우유를 한 잔하는 건 어떨까?

제 4 절 도메인을 한번 설정해 보자.

도메인을 설정하는 간단한 방법

4.1 먼저 알아야 하는 것들

이 절을 시작하기 전에 DNS가 어떻게 동작하는지 약간의 이론을 설명하겠다. 읽어 두면 많은 도움이 되기 때문에 계속 읽는 것이 좋다. 읽고 싶지 않더라도 대충 훑어 보기는 해야 한다. 어쨌든 `named.conf`에 관한 내용부터는 자세히 읽어서 완전히 이해해야 한다.

DNS는 계층적인 시스템이다. 최상위 계층은 '`.`'으로 적고 '루트'로 발음한다. '.' 아래로 ORG, COM, EDU, NET 같은 최상위 도메인(TLDs: Top Level Domains)이 있다.

어떤 컴퓨터를 찾을 때, 쿼리는 최상위 계층에서부터 시작하여 하위 계층으로 찾아 내려 간다. `prep.ai.mit.edu`를 찾는 경우 사용자의 네임 서버는 edu 도메인을 담당하는 네임 서버를 찾아야 한다. 그래서 `.` 서버에서 질의하게 되고 그러면 `.` 서버는 edu 도메인 담당 서버들의 목록을 넘겨 준다. (이미 `.` 서버에 대해서는 사용자의 네임 서버가 알고 있다. 왜냐하면, `root.hints` 파일에서 명시해 주었기 때문이다.)

```
$ nslookup
Default Server:  localhost
Address:  127.0.0.1
```

루트 서버에 질의 시작

```
> server c.root-servers.net.
Default Server:  c.root-servers.net
Address:  192.33.4.12
```

쿼리 유형을 NS로 설정 (name server records):

```
> set q=ns
```

edu에 관해 질의

```
> edu.
```

여기에서 마지막 칸의 .이 중요한데, . 아래에 있는 edu 도메인에 대해 질의하고 있음을 뜻한다. (이렇게 함으로써 검색 범위를 축소한다.)

```
edu     nameserver = A.ROOT-SERVERS.NET
edu     nameserver = H.ROOT-SERVERS.NET
edu     nameserver = B.ROOT-SERVERS.NET
edu     nameserver = C.ROOT-SERVERS.NET
edu     nameserver = D.ROOT-SERVERS.NET
edu     nameserver = E.ROOT-SERVERS.NET
edu     nameserver = I.ROOT-SERVERS.NET
edu     nameserver = F.ROOT-SERVERS.NET
edu     nameserver = G.ROOT-SERVERS.NET
A.ROOT-SERVERS.NET      internet address = 198.41.0.4
H.ROOT-SERVERS.NET      internet address = 128.63.2.53
B.ROOT-SERVERS.NET      internet address = 128.9.0.107
C.ROOT-SERVERS.NET      internet address = 192.33.4.12
D.ROOT-SERVERS.NET      internet address = 128.8.10.90
E.ROOT-SERVERS.NET      internet address = 192.203.230.10
I.ROOT-SERVERS.NET      internet address = 192.36.148.17
F.ROOT-SERVERS.NET      internet address = 192.5.5.241
G.ROOT-SERVERS.NET      internet address = 192.112.36.4
```

위의 결과로 `*.root-servers.net`이 `edu.` 담당 서버임을 알 수 있다. 이제 계속해서 c 서버에게 질의할 수 있다. 이번에는 어느 서버가 `mit.edu.` 도메인을 담당하는지 알아 보자. 계속해서 아래와 같이 `mit.edu.`을 입력한다.

```
> mit.edu.
Server:  c.root-servers.net
Address:  192.33.4.12

Non-authoritative answer:
mit.edu nameserver = W20NS.mit.edu
mit.edu nameserver = BITSY.mit.edu
mit.edu nameserver = STRAWB.mit.edu

Authoritative answers can be found from:
W20NS.mit.edu   internet address = 18.70.0.160
BITSY.mit.edu   internet address = 18.72.0.3
STRAWB.mit.edu  internet address = 18.71.0.151
```

`steawb`, `w20ns`와 `bitsy` 서버가 `mit`를 담당한다. 그 중 하나를 선택하여 `ai.mit.edu`에 대해 질의해 보자.

```
> server W20NS.mit.edu.
```

호스트명은 대소문자를 구별하지는 않는다. 다만 마우스로 화면을 긁어 붙여서 이렇게 보인다.

```
Server:  W20NS.mit.edu
Address:  18.70.0.160

> ai.mit.edu.
Server:  W20NS.mit.edu
Address:  18.70.0.160

Non-authoritative answer:
ai.mit.edu      nameserver = ALPHA-BITS.AI.MIT.EDU
ai.mit.edu      nameserver = GRAPE-NUTS.AI.MIT.EDU
ai.mit.edu      nameserver = TRIX.AI.MIT.EDU
ai.mit.edu      nameserver = MUESLI.AI.MIT.EDU
ai.mit.edu      nameserver = LIFE.AI.MIT.EDU
ai.mit.edu      nameserver = BEET-CHEX.AI.MIT.EDU
ai.mit.edu      nameserver = MINI-WHEATS.AI.MIT.EDU
ai.mit.edu      nameserver = COUNT-CHOCULA.AI.MIT.EDU
ai.mit.edu      nameserver = MINTAKA.LCS.MIT.EDU

Authoritative answers can be found from:
AI.MIT.EDU      nameserver = ALPHA-BITS.AI.MIT.EDU
AI.MIT.EDU      nameserver = GRAPE-NUTS.AI.MIT.EDU
AI.MIT.EDU      nameserver = TRIX.AI.MIT.EDU
AI.MIT.EDU      nameserver = MUESLI.AI.MIT.EDU
AI.MIT.EDU      nameserver = LIFE.AI.MIT.EDU
AI.MIT.EDU      nameserver = BEET-CHEX.AI.MIT.EDU
AI.MIT.EDU      nameserver = MINI-WHEATS.AI.MIT.EDU
AI.MIT.EDU      nameserver = COUNT-CHOCULA.AI.MIT.EDU
AI.MIT.EDU      nameserver = MINTAKA.LCS.MIT.EDU
ALPHA-BITS.AI.MIT.EDU   internet address = 128.52.32.5
GRAPE-NUTS.AI.MIT.EDU   internet address = 128.52.36.4
TRIX.AI.MIT.EDU internet address = 128.52.37.6
MUESLI.AI.MIT.EDU       internet address = 128.52.39.7
LIFE.AI.MIT.EDU internet address = 128.52.32.80
BEET-CHEX.AI.MIT.EDU    internet address = 128.52.32.22
MINI-WHEATS.AI.MIT.EDU  internet address = 128.52.54.11
COUNT-CHOCULA.AI.MIT.EDU        internet address = 128.52.38.22
MINTAKA.LCS.MIT.EDU     internet address = 18.26.0.36
```

위의 결과에서 museli.ai.mit.edu가 ai.mit.edu 담당 네임 서버 중 하나임을 알 수 있다. 마지막으로 아래와 같이 질의해 보자.

```
> server MUESLI.AI.MIT.EDU
Default Server:  MUESLI.AI.MIT.EDU
Address:  128.52.39.7
```

네임 서버를 찾았으므로 이제 쿼리 유형을 바꿔서 prep.ai.mit.edu에 관한 모든 사항을 질의해 보자.

```
> set q=any
> prep.ai.mit.edu.
Server:  MUESLI.AI.MIT.EDU
Address:  128.52.39.7

prep.ai.mit.edu CPU = dec/decstation-5000.25    OS = unix
prep.ai.mit.edu
```

```
        inet address = 18.159.0.42, protocol = tcp
          ftp  telnet  smtp  finger
prep.ai.mit.edu preference = 1, mail exchanger = gnu-life.ai.mit.edu
prep.ai.mit.edu internet address = 18.159.0.42
ai.mit.edu      nameserver = beet-chex.ai.mit.edu
ai.mit.edu      nameserver = alpha-bits.ai.mit.edu
ai.mit.edu      nameserver = mini-wheats.ai.mit.edu
ai.mit.edu      nameserver = trix.ai.mit.edu
ai.mit.edu      nameserver = muesli.ai.mit.edu
ai.mit.edu      nameserver = count-chocula.ai.mit.edu
ai.mit.edu      nameserver = mintaka.lcs.mit.edu
ai.mit.edu      nameserver = life.ai.mit.edu
gnu-life.ai.mit.edu     internet address = 128.52.32.60
beet-chex.ai.mit.edu    internet address = 128.52.32.22
alpha-bits.ai.mit.edu   internet address = 128.52.32.5
mini-wheats.ai.mit.edu  internet address = 128.52.54.11
trix.ai.mit.edu internet address = 128.52.37.6
muesli.ai.mit.edu       internet address = 128.52.39.7
count-chocula.ai.mit.edu        internet address = 128.52.38.22
mintaka.lcs.mit.edu     internet address = 18.26.0.36
life.ai.mit.edu internet address = 128.52.32.80
```

이렇게 해서 .로부터 시작해서 도메인 네임을 담당하는 전단계의 네임 서버들을 성공적으로 찾았다. 다른 서버를 사용하지 않고 여러분의 DNS 서버를 사용했다면 여러분의 named는 당연히 그 모든 정보들을 보관해 두었을 것이다. 그리고 당분간은 같은 질의를 하지 않을 것이다.

많이 거론되지는 않지만 중요한 도메인이 in-addr.arpa 이다. 이 도메인 역시 정상 도메인을 구성한다. in-addr.arpa는 호스트의 주소를 알고 있을 때 그 이름을 알려준다. 여기서 주의해야 할 점은 in-addr.arpa 도메인에서는 ip 숫자들이 역순으로 사용한다는 것이다. 192.128.52.43 컴퓨터의 주소를 알고 있는 경우, prep.ai.mit.edu의 예처럼 'named'는 arpa. 서버를 찾는다. 그 다음으로 in-addr.arpa. 서버를 찾고, 192.in-addr.arpa. 서버를 찾은 다음, 128.192.in-addr.arpa. 서버를 찾아서 52.128.192.in-addr.arpa. 서버를 찾는다. 그리고는 43.52.128.192.in-addr.arpa.에 해당하는 항목을 찾는다. 똑똑하죠?(그렇다고 말하길...) 숫자를 역순으로 사용하는 것은 처음 2년 정도는 혼란스러울 수 있다.

사실 필자는 지금까지 거짓말을 했다. DNS는 저자가 이야기한 글자 그대로 작동하지는 않는다. 그렇지만 그 의미는 충분하다.

4.2 도메인을 설정해 보자.

이제는 간단하게라도 한번 도메인을 설정해 보자. *linux.bogus*라는 도메인을 정하고 그기에 속한 컴퓨터들의 이름을 정의할 것이다. 어느 누구도 혼동하지 않도록 실제로는 존재하지도 않는 도메인을 사용할 것이다.

시작하기 전에 하나 더: 모든 문자를 호스트명으로 사용할 수 있는 것은 아니다. 영문자: a-z, 숫자: 0-9, 그리고 '-' (dash) 문자만 사용할 수 있다. 이 문자들을 명심하라. 대문자나 소문자나 DNS에게는 마찬가지다. 그래서 *pat.uio.no*은 *Pat.UiO.No*와 동일하다.

named.conf에서 아래 부분 설정에 대한 것은 이미 설명했다.

```
zone "0.0.127.in-addr.arpa" {
        type master;
        file "pz/127.0.0";
};
```

이 파일에서 도메인 네임의 마지막에 '.'이 없음에 유의하자. 위의 설정 중 첫번째 라인은 `0.0.127.in-addr.arpa` 존(zone)에 대한 정의임을 뜻하고, 두번째 라인인 이 서버가 `0.0.127.in-addr.arpa` 존의 마스터 서버임을 뜻하며, 마지막 라인은 호스트명과 IP 주소 사이의 매핑 정보가 ps/127.0.0 파일에 저장되어 있음을 뜻한다. ps/127.0.0 파일에 대해서는 이미 설명하였다. 내용은 다음과 같다.

```
@               IN      SOA     ns.linux.bogus. hostmaster.linux.bogus. (
                                1       ; Serial
                                8H      ; Refresh
                                2H      ; Retry
                                1W      ; Expire
                                1D)     ; Minimum TTL
                        NS      ns.linux.bogus.
1                       PTR     localhost.
```

위의 named.conf 파일과는 대조적으로 이 파일에서는 완전한 도메인명(full domain name)의 끝에는 모두 '`.`'이 있음에 유의하자. Some people like to start each zone file with a `$ORIGIN` directive, but this is superfluous. The origin (where in the DNS hierarchy it belongs) of a zone file is specified on the zone section of the `named.conf` file, in this case it's `0.0.127.in-addr.arpa`.

이 '존(zone) 파일'에는 'resource records' (RRs)가 3개 있다. SOA, NS, 그리고 PTR이다. SOA는 'Start Of Authority'의 축약어이다. '@'은 origin을 뜻하는 특수문자이다. 이 파일에 대한 '도메인' 항목이 0.0.127.in-addr.arpa 이므로 첫줄의 의미는 다음과 같다.

```
0.0.127.in-addr.arpa.   IN      SOA ...
```

NS는 네임 서버 RR이다. 이 줄에는 처음에 '@' 문자가 없다. 바로 위에서 '@' 문자로 시작한 줄이 있으므로 이를 암묵적으로 따른다. 타수도 줄일겸.. 그러므로 NS 줄은 다음과 같다.

```
0.0.127.in-addr.arpa.   IN      NS      ns.linux.bogus
```

`0.0.127.in-addr.arpa` 도메인의 네임 서버가 `ns.linux.bogus`임을 다른 DNS들에게 알려 준다. 'ns'가 네임 서버의 이름으로 관례처럼 쓰인다. 그러나 웹서버의 이름이 관례적으로 www.something이듯 다른 이름을 사용하는 것도 무방하다.

마지막으로 PTR 항목은 `0.0.127.in-addr.arpa` 서브넷에서 주소가 1인 호스트, 즉 127.0.0,1의 이름이 `localhost`임을 뜻한다.

SOA 항목은 존 파일의 머리말로 각 존 파일마다 꼭 하나씩, 첫줄에 *반드시* 있어야만 한다. 이 항목은 현재 설정하고 있는 Primary 네임 서버의 이름은 무엇인지 (`ns.linux.bogus`), 관리자는 누구인지, (`hostmaster@linux.bogus`), 존 파일은 버전이 어떻게 되는지 (serial: 1), 캐시 설정과 secondary DNS 서버에 관한 내용을 설정한다. 남은 항목들은 refresh, retry, expire, 그리고 minimum인데, 이 문서와 동일한 값으로 설정하면, 크게 신경쓰지 않아도 잘 작동할 것이다.

이제 `ndc restart` 명령으로 named를 재시작하고 nslookup으로 지금까지 설정한 것을 시험해 보자.

```
$ nslookup

Default Server:  localhost
Address:  127.0.0.1

> 127.0.0.1
Server:  localhost
Address:  127.0.0.1
```

```
Name:    localhost
Address:  127.0.0.1
```

위에서 IP 127.0.0.1에 매핑된 호스트명 `localhost`를 찾는데 성공했다. 굿 이제 `named.conf`에 존(zone)을 새로 추가하여 우리의 주목적인 linux.bogus 도메인을 설정해 보자.

```
zone "linux.bogus" {
        notify no;
        type master;
        file "pz/linux.bogus";
};
```

`named.conf` 파일에서 도메인 네임 마지막에 '`.`'이 없다는 것에 주의하도록 한다.

linux.bogus 존 파일에 100% 가상 데이타를 삽입할 것이다.

```
;
; Zone file for linux.bogus
;
; The full zone file
;
@       IN      SOA     ns.linux.bogus. hostmaster.linux.bogus. (
                        199802151       ; serial, todays date + todays serial #
                        8H              ; refresh, seconds
                        2H              ; retry, seconds
                        1W              ; expire, seconds
                        1D )            ; minimum, seconds
;
                NS      ns              ; Inet Address of name server
                MX      10 mail.linux.bogus     ; Primary Mail Exchanger
                MX      20 mail.friend.bogus.   ; Secondary Mail Exchanger
;
localhost       A       127.0.0.1
ns              A       192.168.196.2
mail            A       192.168.196.4
```

SOA 항목에서 2가지를 주의해야 한다. ns.linux.bogus는 A 레코드가 있는 *실제 컴퓨터이어야 한다.* SOA 레코드에서 언급된 컴퓨터를 다른 컴퓨터로 알리아싱하는 CNAME 레코드가 있으면 규칙에 어긋난다. 이름이 'ns'일 필요는 없으며 다른 호스트명을 사용해도 무방하다. 다음으로, hostmaster.linux.bogus는 hostmaster@linux.bogus로 읽으면 된다. DNS 관리자의 메일 알리아스나 메일박스를 설정하는 곳이다. 도메인 관련 메일은 모두 이 주소로 배달된다. 이름이 'hostmaster'일 필요는 없다. 전자우편 주소라면 어떤 것을 사용해도 상관없지만, 'hostmaster'를 사용하는 것도 나쁘지 않다.

이 파일에는 MX(Mail eXchanger)라는 새로운 RR 유형이 있다. `someone@linux.bogus`의 주소로 들어오는 메일을 처리할 메일 시스템을 지정한다. 위의 예에서 `someone@linux.bogus` 주소로 수신되는 메일은 `mail.linux.bogus` 또는 `mail.friend.bogus`로 보낸다. 호스트명 바로 앞에 있는 숫자는 MX 항목의 우선순위을 뜻한다. 메일은 이 숫자가 가장 낮은(여기서는 10) RR에 메일을 보낸다. 여기서 실패하면 숫자가 그 다음으로 낮은 두번째 메일 서버 즉, 우선 순위가 20일 `mail.friend.bogus`으로 보낼 것이다.

`ndc restart`로 'named'를 재시작한 다음 nslookup으로 결과를 확인하자.

```
$ nslookup
> set q=any
> linux.bogus
Server:  localhost
Address:  127.0.0.1

linux.bogus
        origin = ns.linux.bogus
        mail addr = hostmaster.linux.bogus
        serial = 199802151
        refresh = 28800 (8 hours)
        retry   = 7200 (2 hours)
        expire  = 604800 (7 days)
        minimum ttl = 86400 (1 day)
linux.bogus     nameserver = ns.linux.bogus
linux.bogus     preference = 10, mail exchanger = mail.linux.bogus.linux.bogus
linux.bogus     preference = 20, mail exchanger = mail.friend.bogus
linux.bogus     nameserver = ns.linux.bogus
ns.linux.bogus  internet address = 192.168.196.2
mail.linux.bogus        internet address = 192.168.196.4
```

위 결과를 잘 살펴보면 버그를 찾을 수 있을 것이다.

```
linux.bogus     preference = 10, mail exchanger = mail.linux.bogus.linux.bogus
```

위 라인은 틀렸다. 다음과 같이 출력되어야 정상이다.

```
linux.bogus     preference = 10, mail exchanger = mail.linux.bogus
```

여러분이 좀 더 잘 이해할 수 있도록 이 부분에 실수를 일부러 넣어 두었다. ;-) 존 파일에서 다음 라인을 찾도록 하자.

```
                    MX      10 mail.linux.bogus     ; Primary Mail Exchanger
```

마지막에 점이 빠졌다. 고치지 않으면 'linux.bogus'가 붙어 나오게 된다. 존 파일에서 호스트명이 점으로 끝나지 않으면 `linux.bogus.linux.bogus`처럼 origin이 첨부된다. 그러므로

```
                MX      10 mail.linux.bogus.    ; Primary Mail Exchanger
```

또는

```
                MX      10 mail                 ; Primary Mail Exchanger
```

로 설정하는 것이 올바르다. 저자는 타이핑 수가 적은 후자를 더 좋아한다. bind를 잘 아는 사용자들 중에는 여기에 동의하지 않는 사람도 있고, 동의하는 사람도 있다. 지역 파일에서 도메인은 '`.`'으로 끝나게 완전히 적거나 아니면 디폴트인 origin에 해당하는 부분을 포함하지 말아야 한다.

강조하건데 named.conf 파일에서는 도메인 네임의 끝에 '`.`'이 *없어야* 한다. '.'이 있고 없음이 얼마나 일을 꼬이게 만들고 사람들을 혼란스럽게 만드는지 상상도 못할 것이다.

여기 저자의 견해가 반영된 새로운 존 파일이 있다. 자료가 약간 더해졌다.

```
;
; Zone file for linux.bogus
;
; The full zone file
;
@       IN      SOA     ns.linux.bogus. hostmaster.linux.bogus. (
                        199802151       ; serial, todays date + todays serial #
                        8H              ; refresh, seconds
                        2H              ; retry, seconds
                        1W              ; expire, seconds
                        1D )            ; minimum, seconds
;
                TXT     "Linux.Bogus, your DNS consultants"
                NS      ns              ; Inet Address of name server
                NS      ns.friend.bogus.
                MX      10 mail         ; Primary Mail Exchanger
                MX      20 mail.friend.bogus. ; Secondary Mail Exchanger

localhost       A       127.0.0.1

gw              A       192.168.196.1
                HINFO   "Cisco" "IOS"
                TXT     "The router"

ns              A       192.168.196.2
                MX      10 mail
                MX      20 mail.friend.bogus.
                HINFO   "Pentium" "Linux 2.0"
www             CNAME   ns

donald          A       192.168.196.3
                MX      10 mail
                MX      20 mail.friend.bogus.
                HINFO   "i486"  "Linux 2.0"
                TXT     "DEK"

mail            A       192.168.196.4
                MX      10 mail
                MX      20 mail.friend.bogus.
                HINFO   "386sx" "Linux 1.2"

ftp             A       192.168.196.5
                MX      10 mail
                MX      20 mail.friend.bogus.
                HINFO   "P6" "Linux 2.1.86"
```

위에 새로운 RR이 꽤 많이 있다. HINFO(Host INFOrmation)은 두 부분으로 이로어져 있는데 각각을 큰따옴표로 둘러 싸는 것이 좋다. 앞부분은 컴퓨터 하드웨어 또는 CPU 정보이다. 두번째 부분은 소프트웨어 또는 OS 정보이다. 'ns' 컴퓨터는 Pentium CPU에 Linux 2.0을 사용한다. CNAME(Canonical NAME)은 컴퓨터 하나에 이름을 여러 개 부여하는 방법이다. 그러므로 www은 ns에 대한 알리아스이다.

CNAME 레코드의 용법은 약간 논쟁의 여지가 있다. 그러나 다음 규칙을 따르면 안전하다. MX, CNAME, SOA 항목은 CNAME 레코드와는 *절대*로 연결하지 말아야 하고, A 항목이 있는 가진 다른것과 연결하여야 한다. 즉, 다음은 잘못 설정한 것이다.

```
foobar          CNAME   www                     ; NO!
```

아래와 같이 설정하는 것이 올바르다.

```
foobar          CNAME   ns                      ; Yes!
```

또한 CNAME은 전자우편 주소로 바람직한 호스트명이 아니라고 가정하는 것이 안전하다. 즉, `webmaster@www.linux.bogus`는 규정에 어긋난 전자우편 주소이다. 이 가정을 따르지 않으면 비록 동작은 하겠지만 메일 관리가 상당히 어려워진다. 이를 막으려면 A 레코드(또는 MX 같은 레코드)를 대신 사용한다.

```
www             A       192.168.196.2
```

많은 bind 전문가들은 CNAME을 사용하지 말 것을 권한다. 그러므로 사용하지 않는 것에 대해 *아주 신중하게* 검토해 보라.

그러나 여러분도 알듯, 이 하우투도 그렇고 많은 사이트가 이 규칙을 따르지는 않는다.

`ndc reload`로 데이터베이스를 새로 읽어 들이자. `ndc reload`를 실행하면 named는 파일들을 다시 읽는다.

```
$ nslookup
Default Server:  localhost
Address:  127.0.0.1

> ls -d linux.bogus
```

이는 모든 레코드가 출력되어야 함을 뜻한다 . 결과는 다음과 같다.

```
[localhost]
$ORIGIN linux.bogus.
@                       1D IN SOA       ns hostmaster (
                                        199802151       ; serial
                                        8H              ; refresh
                                        2H              ; retry
                                        1W              ; expiry
                                        1D )            ; minimum

                        1D IN NS        ns
                        1D IN NS        ns.friend.bogus.
                        1D IN TXT       "Linux.Bogus, your DNS consultants"
                        1D IN MX        10 mail
                        1D IN MX        20 mail.friend.bogus.
gw                      1D IN A         192.168.196.1
                        1D IN HINFO     "Cisco" "IOS"
                        1D IN TXT       "The router"
mail                    1D IN A         192.168.196.4
                        1D IN MX        10 mail
                        1D IN MX        20 mail.friend.bogus.
                        1D IN HINFO     "386sx" "Linux 1.0.9"
localhost               1D IN A         127.0.0.1
```

```
www                     1D IN CNAME     ns
donald                  1D IN A         192.168.196.3
                        1D IN MX        10 mail
                        1D IN MX        20 mail.friend.bogus.
                        1D IN HINFO     "i486" "Linux 1.2"
                        1D IN TXT       "DEK"
ftp                     1D IN A         192.168.196.5
                        1D IN MX        10 mail
                        1D IN MX        20 mail.friend.bogus.
                        1D IN HINFO     "P6" "Linux 1.3.59"
ns                      1D IN A         192.168.196.2
                        1D IN MX        10 mail
                        1D IN MX        20 mail.friend.bogus.
                        1D IN HINFO     "Pentium" "Linux 1.2"
@                       1D IN SOA       ns hostmaster (
                                        199802151       ; serial
                                        8H              ; refresh
                                        2H              ; retry
                                        1W              ; expiry
                                        1D )            ; minimum
```

결과가 위와 같다면 정상이다. 위 결과는 바로 존 파일과 비슷하게 보인다. www에 대해서는 무어라 말하는지 확인해 보자.

```
> set q=any
> www.linux.bogus.
Server:  localhost
Address:  127.0.0.1

www.linux.bogus canonical name = ns.linux.bogus
linux.bogus     nameserver = ns.linux.bogus
linux.bogus     nameserver = ns.friend.bogus
ns.linux.bogus  internet address = 192.168.196.2
```

달리 표현하자면, `www.linux.bogus`의 실제 이름은 `ns.linux.bogus`이다. 도한 ns에 대한 정보도 함께 반환해 주기 때문에 프로그램은 이 정보를 이용하여 ns(www이기도 함)에 접속할 수 있다.

이제 한 반 정도를 설명하였다.

4.3 역변환 존(The reverse zone)

이제 클라이언트 프로그램들이 linux.bogu 도메인 호스트들의 이름을 주소로 변환하여 원하는 컴퓨터에 접속할 수 있다. 그렇지만, 역변환 존이 설정되어야 DNS가 주소를 이름으로 변환할 수 있다. FTP, IRC, WWW 등 다양한 서버가 여러분의 컴퓨터와 통신을 허용할 것인지, 허용한다면 어떤 우선 순위를 줄 것인지 결정하는 데 바로 호스트명을 사용한다. 그러므로 역변환 존이 설정되어 있어야만 해당 도메인의 컴퓨터가 모든 인터넷 서비스를 완전하게 사용할 수 있다.

아래 내용을 `named.conf` 파일에 삽입하자.

```
zone "196.168.192.in-addr.arpa" {
        notify no;
        type master;
        file "pz/192.168.196";
};
```

0.0.127.in-addr.arpa과 동일하다. 내용도 비슷하다.

```
@       IN      SOA     ns.linux.bogus. hostmaster.linux.bogus. (
                        199802151 ; Serial, todays date + todays serial
                        8H      ; Refresh
                        2H      ; Retry
                        1W      ; Expire
                        1D)     ; Minimum TTL
                NS      ns.linux.bogus.

1               PTR     gw.linux.bogus.
2               PTR     ns.linux.bogus.
3               PTR     donald.linux.bogus.
4               PTR     mail.linux.bogus.
5               PTR     ftp.linux.bogus.
```

이제 당신의 named를 재시작(`ndc restart`)하고 nslookup으로 지금까지 설정한 내용을 확인해 보자.

```
> 192.168.196.4
Server:  localhost
Address:  127.0.0.1

Name:    mail.linux.bogus
Address:  192.168.196.4
```

위와 같이 제대로 보이면, 확인삼아 전체를 덤프시켜 보자.

```
> ls -d 196.168.192.in-addr.arpa
[localhost]
$ORIGIN 196.168.192.in-addr.arpa.
@                       1D IN SOA       ns.linux.bogus. hostmaster.linux.bogus. (
                                        199802151       ; serial
                                        8H              ; refresh
                                        2H              ; retry
                                        1W              ; expiry
                                        1D )            ; minimum

                        1D IN NS        ns.linux.bogus.
1                       1D IN PTR       gw.linux.bogus.
2                       1D IN PTR       ns.linux.bogus.
3                       1D IN PTR       donald.linux.bogus.
4                       1D IN PTR       mail.linux.bogus.
5                       1D IN PTR       ftp.linux.bogus.
@                       1D IN SOA       ns.linux.bogus. hostmaster.linux.bogus. (
                                        199802151       ; serial
                                        8H              ; refresh
                                        2H              ; retry
                                        1W              ; expiry
                                        1D )            ; minimum
```

와우, 성공이다!

여기에 보충해야 할 것이 약간 있다. 위의 예에서 사용된 IP 숫자들은 'private nets' 블럭중에서 하나를 택한 것이다. 그러므로 인터넷에 물려서 공식적으로 사용되어서는 안된다. 그래서 HOWTO에서 예제로 사용하는 것은 안전하다. 두번째는 `notify no;` 줄이다. 이것은 'named'가 그 지역 파일들 중에서 하나가 갱신되었을 때 secondary(slave) 서버에게 알리지 않도록 한다. bind-8에서는 지역 파일이 갱신되었을 때 지역 파일에 나열된 NS 레코드의 서버에게 'named'가 알려줄 수 있다. 이 기능은 DNS를 실제로 운영할 때는 편리하지만 사적인 연습에는 이 기능을 꺼야할 것이다. 우리의 연습으로 인터넷을 오염시킬 수는 없지 않은가?

지금까지 사용한 도메인도 완전히 가상이고, 그 주소들도 실제로 사용하는 주소가 아니다. 실제 도메인의 예는 다음 절을 참조하라.

제 5 절 도메인 설정의 실제 예

여기에서 *실제* 존 파일 몇 개를 다룰 것이다.

사용자들이 교육적인 예와 함께 실제로 사용되고 있는 도메인의 예를 포함해 줄 것을 제안했다.

LAND-5의 David Bullock 씨의 허락하에 아래 예들을 사용한다. 이 파일들은 1996년 9월 24일에 만들어졌다. bind-8 조건에 맞게 수정하였고 저자가 좀 더 확장하여 사용하였다. 그러므로 현재의 LAND-5 네임 서버에 쿼리를 한다면 여기서와는 조금은 다른 결과를 얻게 될 것이다.

5.1 /etc/named.conf (또는 /var/named/named.conf)

여기서 127.0.0 과 LAND-5의 206.6.177 서브넷에 필요한 역변환 존에 대한 마스터 존 섹션을 살펴보자. 그리고 lang-5.com 존을 살펴 보자. 이 하우투에서 저자는 `pz` 라는 디렉토리에 파일들을 두었지만 그는 `zone`zone 이라는 디렉토리에 두고 있음에 주의하자.

```
// Boot file for LAND-5 name server

options {
        directory "/var/named";
};

zone "." {
        type hint;
        file "root.hints";
};

zone "0.0.127.in-addr.arpa" {
        type master;
        file "zone/127.0.0";
};

zone "land-5.com" {
        type master;
        file "zone/land-5.com";
};

zone "177.6.206.in-addr.arpa" {
```

```
        type master;
        file "zone/206.6.177";
};
```

위의 내용을 실제로 named.conf에 넣어서 테스트할 경우에는, 사고가 일어나지 않도록 lang-5 존과 역변환 존 두 곳에 `notify no;` 라인을 *반드시* 넣어라.

5.2 /var/named/root.hints

이 파일은 유동적임을 명심하라. 그러므로 여기 나열된 정보는 예전 것이다. 이전에 설명되었던 dig로 산출된 현재의 것을 사용하는 것이 훨씬 나을 것이다.

```
; <<>> DiG 8.1 <<>> @A.ROOT-SERVERS.NET.
; (1 server found)
;; res options: init recurs defnam dnsrch
;; got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 10
;; flags: qr aa rd; QUERY: 1, ANSWER: 13, AUTHORITY: 0, ADDITIONAL: 13
;; QUERY SECTION:
;;      ., type = NS, class = IN

;; ANSWER SECTION:
.                       6D IN NS        G.ROOT-SERVERS.NET.
.                       6D IN NS        J.ROOT-SERVERS.NET.
.                       6D IN NS        K.ROOT-SERVERS.NET.
.                       6D IN NS        L.ROOT-SERVERS.NET.
.                       6D IN NS        M.ROOT-SERVERS.NET.
.                       6D IN NS        A.ROOT-SERVERS.NET.
.                       6D IN NS        H.ROOT-SERVERS.NET.
.                       6D IN NS        B.ROOT-SERVERS.NET.
.                       6D IN NS        C.ROOT-SERVERS.NET.
.                       6D IN NS        D.ROOT-SERVERS.NET.
.                       6D IN NS        E.ROOT-SERVERS.NET.
.                       6D IN NS        I.ROOT-SERVERS.NET.
.                       6D IN NS        F.ROOT-SERVERS.NET.

;; ADDITIONAL SECTION:
G.ROOT-SERVERS.NET.     5w6d16h IN A    192.112.36.4
J.ROOT-SERVERS.NET.     5w6d16h IN A    198.41.0.10
K.ROOT-SERVERS.NET.     5w6d16h IN A    193.0.14.129
L.ROOT-SERVERS.NET.     5w6d16h IN A    198.32.64.12
M.ROOT-SERVERS.NET.     5w6d16h IN A    202.12.27.33
A.ROOT-SERVERS.NET.     5w6d16h IN A    198.41.0.4
H.ROOT-SERVERS.NET.     5w6d16h IN A    128.63.2.53
B.ROOT-SERVERS.NET.     5w6d16h IN A    128.9.0.107
C.ROOT-SERVERS.NET.     5w6d16h IN A    192.33.4.12
D.ROOT-SERVERS.NET.     5w6d16h IN A    128.8.10.90
E.ROOT-SERVERS.NET.     5w6d16h IN A    192.203.230.10
I.ROOT-SERVERS.NET.     5w6d16h IN A    192.36.148.17
F.ROOT-SERVERS.NET.     5w6d16h IN A    192.5.5.241
```

```
;; Total query time: 215 msec
;; FROM: roke.uio.no to SERVER: A.ROOT-SERVERS.NET.  198.41.0.4
;; WHEN: Sun Feb 15 01:22:51 1998
;; MSG SIZE  sent: 17  rcvd: 436
```

5.3 /var/named/zone/127.0.0

기본적으로 필수 레코드인 SOA 레코드가 필요하며, 127.0.0.1을 `localhost`로 매핑해 주는 레코드가 필요하다. 그 외의 것들이 이 파일에 있어서는 안된다. 네임서버가 바뀌거나 hostmaster 메일 주소가 바뀌지 않는 한 이 파일은 갱신할 필요가 없다.

```
@               IN      SOA     land-5.com. root.land-5.com. (
                                199609203       ; Serial
                                28800   ; Refresh
                                7200    ; Retry
                                604800  ; Expire
                                86400)  ; Minimum TTL
                        NS      land-5.com.

1                       PTR     localhost.
```

5.4 /var/named/zone/land-5.com

필수 레코드인 SOA 레코드가 필요하며, NS 레코드도 필요하다. secondary 네임 서버로 ns2.psi.net이 있음을 알수 있다. 이 서버는 백업용으로 *항상* 사이트 밖에 있어야 한다. 또한 다양한 인터넷 서비스를 담당하는 마스터 호스트로 lang-5를 두었고, 그러한 처리를 CNAME으로 해결하고 있음을 알 수 있다. (A 레코드를 사용한 방법도 있다.)

SOA 항목에서 알 수 있듯이, 존 파일은 origin이 land-5.com이며 관리자는 `root@land-5.com`이다. `hostmaster`는 관리자의 주소로 자주 사용되는 것이다. 시리얼 넘버는 의례적으로 yyyymmdd 형식에 그날의 시리얼 넘버를 덧붙인다. 아래서 보면 아마 이 지역 파일은 1996년 9월 20일에 6번째 수정한 버전일 것이다. 시리얼 넘버는 *한방향으로만 증가해야* 함을 명심하자. 여기서는 그날의 시리얼 넘버가 *한자리*다. 그러므로 9번을 편집하고 나서 또 편집하려면 내일을 기다려야 할 것이다. 두 자리수 사용을 고려하자.

```
@       IN      SOA     land-5.com. root.land-5.com. (
                        199609206       ; serial, todays date + todays serial #
                        8H              ; refresh, seconds
                        2H              ; retry, seconds
                        1W              ; expire, seconds
                        1D )            ; minimum, seconds
                NS      land-5.com.
                NS      ns2.psi.net.
                MX      10 land-5.com.  ; Primary Mail Exchanger

localhost       A       127.0.0.1

router          A       206.6.177.1
```

```
land-5.com.     A       206.6.177.2
ns              A       206.6.177.3
www             A       207.159.141.192

ftp             CNAME   land-5.com.
mail            CNAME   land-5.com.
news            CNAME   land-5.com.

funn            A       206.6.177.2
@               TXT     "LAND-5 Corporation"

;
;       Workstations
;
ws-177200       A       206.6.177.200
                MX      10 land-5.com.   ; Primary Mail Host
ws-177201       A       206.6.177.201
                MX      10 land-5.com.   ; Primary Mail Host
ws-177202       A       206.6.177.202
                MX      10 land-5.com.   ; Primary Mail Host
ws-177203       A       206.6.177.203
                MX      10 land-5.com.   ; Primary Mail Host
ws-177204       A       206.6.177.204
                MX      10 land-5.com.   ; Primary Mail Host
ws-177205       A       206.6.177.205
                MX      10 land-5.com.   ; Primary Mail Host
; {Many repetitive definitions deleted - SNIP}
ws-177250       A       206.6.177.250
                MX      10 land-5.com.   ; Primary Mail Host
ws-177251       A       206.6.177.251
                MX      10 land-5.com.   ; Primary Mail Host
ws-177252       A       206.6.177.252
                MX      10 land-5.com.   ; Primary Mail Host
ws-177253       A       206.6.177.253
                MX      10 land-5.com.   ; Primary Mail Host
ws-177254       A       206.6.177.254
                MX      10 land-5.com.   ; Primary Mail Host
```

---

land-5의 네임 서버를 확인해 보면 알겠지만 호스트명이 ws_*number*의 형식으로 되어 있다. 예전의 bind 4 버전에서는 named가 시작할 때, 호스트명으로 사용할 수 있는 문자 제한을 강제로 준수하였다. 그러나 bind-8에서는 작동하지 않으므로 '_'(underline) 대신 '-'(dash)로 바꿨다.

또 하나 주목할 사항은 웍스테이션들은 개개의 이름이 없고 IP 숫자의 끝 두부분을 이름으로 사용한다는 점이다. 이런 관례는 유지 보수를 상당히 단순화할 수 있다. 대신 조금은 비인간적이라 고객들 사이에 불만의 요인이 될수 있다.

또한 funn.land-5.com이 land-5.com에 대한 알리아스임을 알 수 있다. 그러나 CNAME 항목이 아닌 A 항목을 사용한다.

5.5 /var/named/zone/206.6.177

이 파일에 대해서는 잠시 후에 설명할 것이다.

```
@               IN      SOA     land-5.com. root.land-5.com. (
                                199609206       ; Serial
                                28800   ; Refresh
                                7200    ; Retry
                                604800  ; Expire
                                86400)  ; Minimum TTL
                        NS      land-5.com.
                        NS      ns2.psi.net.
;
;       Servers
;
1       PTR     router.land-5.com.
2       PTR     land-5.com.
2       PTR     funn.land-5.com.
;
;       Workstations
;
200     PTR     ws-177200.land-5.com.
201     PTR     ws-177201.land-5.com.
202     PTR     ws-177202.land-5.com.
203     PTR     ws-177203.land-5.com.
204     PTR     ws-177204.land-5.com.
205     PTR     ws-177205.land-5.com.
; {Many repetitive definitions deleted - SNIP}
250     PTR     ws-177250.land-5.com.
251     PTR     ws-177251.land-5.com.
252     PTR     ws-177252.land-5.com.
253     PTR     ws-177253.land-5.com.
254     PTR     ws-177254.land-5.com.
```

역변환 존은 재난의 대부분을 재난을 일으키는 설정 부분으로 보인다. 역변환 존은 컴퓨터의 IP 주소를 알때 그 호스트명을 찾는데 사용된다. 예: 여러분의 컴퓨터가 IRC 서버이고 IRC 클라이언트의 접속을 허용한다. 그러나 그 컴퓨터는 노르웨이 언어 전용의 IRC 서버로 노르웨이와 다른 스칸다나비아 반도 국가에 있는 클라이언트의 접속 만을 허락하고 싶다. 클라이언트로부터 접속이 있을때 C 라이브러리는 접속하는 클라이언트 컴퓨터의 IP 주소를 알려줄 수 있다. 네트웍을 지나는 모든 패킷에 클라이언트 컴퓨의 IP 주소가 포함되어 있기 때문이다. 이제 여러분의 IRC 서버는 주어진 IP 주소로 호스트명을 찾는 gethostbyaddr 함수를 호출할 수 있다. Gethostbyaddr 함수는 DNS 서버를 찾을 것이다. 그리고는 컴퓨터를 찾는 항해를 한다. ws-177200.land-5.com에서 클라이언트가 접속했다고 가정하면 C 라이브러리가 IRC 서버에게 건네는 IP 주소는 206.6.177.200이다. 이 컴퓨터의 호스트명을 찾으려면 200.177.6.206.in-addr.arpa을 찾아야 한다. DNS 서버는 먼저 arpa. 서버를 찾는다. 그런 다음 in-addr.arpa. 서버를, 그 다음에는 206을, 그 다음에는 6을, 마지막으로 land-5에서 177.6.206.in-addr.arpa zone을 담당하는 서버를 찾는다. 거기서 마침내 200.177.6.206.in-addr.arpa라는 주소에 'PTR ws-177200.land-5.com'이라는 레코드가 매핑되어 있다는 응답을 얻을 수 있다. 그 의미는 206.6.177.200의 호스트명이 ws-177200.land-5.com이라는 것을 뜻한다. prep.ai.mit.edu의 설명에서와 마찬가지로 이 설명은 허구에 가깝다.

IRC 서버의 예로 돌아가자. 위의 IRC 서버는 *.no, *.se, *.dk와 같은 스칸다나비아 반도 주변국에서의 접속 만을 허용하고자 한다. ws-177200.land-5.com는 해당 사항이 없으므로 접속을 거부할 것이다. in-addr.arpa

존에 206.2.177.200의 역변환 매핑(reverse mapping)이 *없다면* 서버는 이름을 알수 없을 것이고 결국은 206.2.177.200라는 숫자를 *.no, *.se, *.dk와 비교하게 될 것이다.

역변환 매핑(reverse lookup mapping)이 서버한테만 중요하다고 하는 이도 이으며, 전혀 중요하지 않다고 말하는 이도 있다. 그러나 사실은 매우 중요한다. 많은 ftp, news, IRC, 심지어 http(WWW) 서버도 몇몇은 클라이언트 컴퓨터의 이름을 찾을 수 없다면 접속을 불허할 것이다. 그러므로 컴퓨터의 역변환 매핑은 반드시 필요하다.

제 6 절 유지 보수

항상 올바른 작동을 위해 (Keeping it working).

named가 실행되도록 유지하는 것 외에 항상 유념해야 하는 것이 있다. `root.hints` 파일을 최신의 것으로 유지하는 것이다. 제일 쉬운 방법은 dig를 사용하는 것이다. 먼저 아무런 아규먼트 없이 dig를 실행한다. 그러면 바로 서버에 따라서 약간은 다른 root.hints를 얻을 것이다. 그런 다음 `dig @rootserver`로 나열된 루트 서버 중 한곳에 요청한다. `root.hints`와 유사한 끔찍한 결과를 얻게 될 것이다. 결과를 파일로 저장하고(`dig @e.root-servers.net . ns >root.hints.new`) 예전의 root.hints와 대체시킨다.

캐쉬 파일을 대체한 후에는 반드시 named를 재시작하도록 하자.

Al Longyear씨가 `root.hints`를 자동으로 갱신할 수 있는 아래 스크립트를 보내 주었다. crontab에 넣어서 한달에 한번꼴로 실행되도록 해두면 잊어도 된다. 이 스크립트에서는 여러분의 메일이 작동하고 있고 메일 알리아스 'hostmaster'가 정의되어 있다고 가정한다. 여러분에게 맞게 고쳐야 한다.

```
#!/bin/sh
#
# Update the nameserver cache information file once per month.
# This is run automatically by a cron entry.
#
# Original by Al Longyear
# Updated for bind 8 by Nicolai Langfeldt
# SERVFAIL problem discovered by David A. Ranch
#
(
 echo "To: hostmaster <hostmaster>"
 echo "From: system <root>"
 echo "Subject: Automatic update of the named.conf file"
 echo

 export PATH=/sbin:/usr/sbin:/bin:/usr/bin:
 cd /var/named

 dig @rs.internic.net . ns >root.hints.new

 case `cat root.hints.new` in
   *SERVFAIL*)
        echo "The named.conf file update has FAILED."
        echo "This is the error that DIG reported:"
        echo
        cat root.hints.new
        exit 0
 esac
```

```
 echo "The named.conf file has been updated to contain the following
information:"
 echo
 cat root.hints.new

 chown root.root root.hints.new
 chmod 444 root.hints.new
 rm -f root.hints.old
 mv root.hints root.hints.old
 mv root.hints.new root.hints
 ndc restart
 echo
 echo "The nameserver has been restarted to ensure that the update is complete."
 echo "The previous root.hints file is now called
/var/named/root.hints.old."
) 2>&1 | /usr/lib/sendmail -t
exit 0
```

여러분 중 몇몇은 ftp로 Internic에서 root.hints 파일을 가져올 수 있다고 꼬집어 말할지도 모른다. ftp로 root.hints를 갱신 하지 말라. 위의 방법이 네트웍에 더욱 친근하다.

제 7 절 버전 4에서 버전 8로의 마이그레이션

이 부분은 David E. Smith (dave@bureau42.ml.org)씨가 쓴 'using bind 8'에 있던 절이다. 새로운 절의 이름에 맞도록 약간 편집을 가했다.

별로 해야할 것은 없다. named.boot대신 named.conf를 사용하는 점 말고는 모든 것이 동일하다. bind8은 펄 스크립트로 옛 형식의 파일들을 새로운 형식에 맞게 변환한다. 다음은 옛형식으로 된 캐시 전용 네임 서버의 예이다.

```
directory /var/named
cache   .                                        root.hints
primary 0.0.127.IN-ADDR.ARPA                     127.0.0.zone
primary localhost                                localhost.zone
```

On the command line, in the bind8/src/bin/named directory (*this assumes you got a source distribution. If you got a binary package the script is probably around, I'm not sure where it would be though. -ed.*), type:

bind8/src/bin/named 디렉토리(*여러분에게 소스가 있다고 가정한다. 만약 바이너리 패키지를 가지고 있더라도 이 스크립트는 분명 어딘가에 있을 것이다. 어디에 있을지는 확신할수 없다.*)에서 다음 명령을 입력하자.

```
./named-bootconf.pl < named.boot > named.conf
```

그러면 named.conf가 만들어 진다.

```
// generated by named-bootconf.pl

options {
```

```
        directory "/var/named";
};

zone "." {
        type hint;
        file "root.hints";
};

zone "0.0.127.IN-ADDR.ARPA" {
        type master;
        file "127.0.0.zone";
};

zone "localhost" {
        type master;
        file "localhost.zone";
};
```

named.conf 파일에 들어갈 수 있을 만큼 모든 것이 작동하기는 하지만 bind8이 지원하는 새롭게 향상된 기능이나 설정 옵션들은 추가되지 않는다. 여기에 똑같은 일을 하지만 좀더 효과적인 더욱 완전한 named.conf가 있다.

```
// This is a configuration file for named (from BIND 8.1 or later).
// It would normally be installed as /etc/named.conf.
// The only change made from the ‘stock’ named.conf (aside from this
// comment :) is that the directory line was uncommented, since I
// already had the zone files in /var/named.

options {
        directory "/var/named";
        datasize 20M;
};

zone "localhost" IN {
        type master;
        file "localhost.zone";
};

zone "0.0.127.in-addr.arpa" IN {
        type master;
        file "127.0.0.zone";
};

zone "." IN {
        type hint;
        file "root.hints";
};
```

bind8/src/bin/named/test에 위의 예와 함께 바로 가져다 쓸 수 있는 존 파일 복사본이 많이 있다.

존 파일과 root.hints 파일을 업데이트하는 명령이 동일하듯이, 존 파일과 root.hints 파일의 형식도 동일하다.

제 8 절 질문과 답

필자에게 메일을 보내기 전에 아래 내용을 읽어 주길 바란다.

1. named가 named.boot 파일을 요구한다.

   여러분은 어뚱한 HOWTO를 읽고 있다. bind-4에 관한 HOWTO는 `http://www.math.uio.no/\~{}janl/DNS/`에서 찾을 수 있다.

2. 방화벽 내부에서는 DNS를 어떻게 사용하는가?

   힌트: 'forwarders', 'slave', 그리고 이 HOWTO의 마지막에 있는 참고 문헌들을 살펴 보기 바란다. 3(캐시 전용 네임 서버) 절의 예에서 제안한 것처럼 `named.conf` 파일의 옵션 부분에 아래 코드가 필요한 경우도 있다.

```
query-source port 53;
```

3. 어떤 서비스를 제공할 때 이 서비스를 제공하는 컴퓨터들의 주소를 DNS가 순서대로 차례 차례 답하도록 하여 트래픽을 효과적으로 분산시킬 수 있는가? 예를 들며, www.busy.site

   www.busy.site와 주소를 매핑하는 **A** 레코드를 여러 개 만든다. 그리고 bind는 4.9.3 또는 그 이후 버전을 사용해야 한다. 그러면 bind가 알아서 www.busy.site에 매핑된 주소를 하나씩 차례로 응답할 것이다. 그 이전 버전의 bind에서는 이렇게 작동하지 않을 것이다.

4. (외부와 연결이 안된) 인트라넷에 DNS를 설정하고 싶다. 어떻게 하나?

   root.hints 파일은 빼고 존 파일만 사용한다. 이것은 또한 항상 새로운 hint 파일을 가져올 필요가 없음을 뜻한다.

5. secondary (slave) 네임 서버는 어떻게 설정하는가?

   만약 primary 서버의 주소가 127.0.0.1이라면 secondary 서버의 named.conf 파일에 다음과 같이 입력한다.

```
zone "linux.bogus" {
      type slave;
      file "sz/linux.bogus";
      masters { 127.0.0.1; };
};
```

   여기에 마스터 서버 여러 개를 ';' (세미콜론)으로 분리하여 나열하면, 여러 서버로부터 linux.bogus의 설정을 복사하게 된다. 물론 여기에 나열하는 마스터 서버에는 linux.bogus가 설정되어 있어야 한다.

6. 네트워크 접속이 끊어질 때 bind를 가동하고 싶다.

   이 주제에 관한 해답(설명)은 두 가지가 있다.

   - Ian Clark <ic@deakin.edu.au> 씨로부터 그가 사용하는 방법을 설명한 메일을 받았다.

```
나는 'Masquerading'을 사용하는 컴퓨터에서 named를 운영한다. 나는
root.hints 파일을 두 개 사용한다. 실제 루트 네임 서버의 이름들을 가진
root.hints.real과 아래와 같은 내용의 root.hints.fake를 사용한다.

----
; root.hints.fake
; this file contains no information
```

```
----

네트웍과 연결이 끊어질 때 root.hints.fake 파일을 root.hints로 복사하고 named를
재시작한다.

네트웍과 연결될 때는 root.hints.real 파일을 root.hints로 복사하고 named
를재시작한다.

ip-down과 ip-up이라는 스크립트를 각각 만들어서 사용한다.

네트웍과 단절되었을 때 named에 상세한 정보가 없는 도메인
네임에 관해 쿼리를 보내면 messages 파일에 같은 내용을 기록한다.

Jan 28 20:10:11 hazchem named[10147]: No root nameserver for class IN

이상이 내가 사용하는 것이다.

확실히 나에게는 제대로 작동하는 것 같다. 나는 네트웍과 단절되었을 때도 지역
컴퓨터을 위한 네임서버를 외부의 도메인 네임으로 인한 타임아웃 지연 없이
사용할 수 있다. 그리고 네트웍과 연결된 동안에는 일반적인 외부 도메인에 대한
쿼리를 실행할 수 있다.
```

- 네트워크와 연결되지 않는 컴퓨터에서 bind가 NFS 및 포트매퍼(portmapper)와 함께 운영하는 방법에 대한 설명을 Karl-Max Wanger 씨가 보내 주었다.

```
가끔 모뎀으로 인터넷에 접속하는 모든 컴퓨터에 named를
운영하고 있다. 네임 서버는 캐시 전용 서버로만 작동하며,
인증 영역이 없어서 모든 쿼리를 root.cache 파일에 명시된 네임 서버(들)에게
질의한다. 그리고 named는 nfsd와 mountd가 기동하기 전에 시작하는데, 이 방식은
슬랙웨어에서는 일반적이다.

LAN에 연결된 다른 컴퓨터가 내 컴퓨터 중 하나(Libretto 30 노트북)를 가끔
마운트하지 못하는 문제가 있었다. 그런데 그 가끔이 실제로는 대부분이었다.
이러한 현상은 PLIP, PCMCIA 랜카드, 시리얼 인터페이스를 통한 PPP 모두에서
일어나는 공통적인 현상이 었다.

몇 시간 동안 생각하고 실험을 거친 후에, 부팅될 때 named가 nfsd와 mountd의
등록 과정과 뒤죽박죽이 되어서 포트매퍼에 등록되었기 때문에 이런 문제가
생긴다는 것을 알았다. (나는 보통 이 데몬들을 부팅할 때 실행한다.)
nfsd와 mountd를 먼저 실행한 다음 named를 실행하니 이러한 문제가 없어졌다.

부팅 순서를 위와 같이 바꾸어도 그로 인한 아무런 문제가 생기지 않으니, 모두들
이렇게 바꾸어서 잠재적인 문제점을 해결해 두는 것이 좋을 것 같다.
```

7. 캐쉬 전용 네임 서버는 그 캐쉬 정보를 어디에 저장하나? 캐쉬 크기를 제어할수 있는 방법은 없는가?

   캐시된 정보는 모두 메모리에 저장된다. 디스크에는 기록되지 *않는다*. named를 죽일 때마다 캐시는 사라진다. 캐시 정보는 어떤 방법으로든 제어할 수 *없다*. named는 어떤 간단한 규칙에 따라 캐시를 다루는데 다음과 같다. 어떤 목적으로든 캐시 정보나 캐시 크기를 제어할 방법은 없다. 그러고 싶다면 named를 해킹해서 수정하면 된다. 그러나 권하지는 않는다.

8. named가 재시작되는 동안은 캐시를 저장하는가? 저장하도록 할 수 있는가?

   없다. named는 멈출 때 캐시를 저장하지 *않는다*. 즉, named가 멈추었다가 다시 시작할 때마다 캐시는 새로 만들어 진다. named로 하여금 캐시를 파일로 저장하게 할 수는 *없다*. 그러고 싶다면 named를 해킹해서 수정하면 된다. 그러나 권하지는 않는다.

제 9 절 많은 시간을 할애하여 DNS를 관리해야 한다면.

문서와 도구

통신과 출판물로 유용한 문서가 있다. 간단히 도메인을 관리하는 정도가 아니라 많은 시간을 들여서 복잡한 도메인을 관리해야 한다면 이 문서 중 몇 가지는 반드시 읽어야 한다. 출판물 중 대표적인것은 C. Liu 와 P. Albitz가 쓴 *DNS and BIND*라는 책으로 O'Reilly & Associates에서 출판하였다. 필자도 읽어 보았는데 아주 훌륭하다. Craig Hunt가 집필하여 역시 O'Reilly & Associates에서 출판한 *TCP/IP Network Administration*의 DNS 절도 읽어 볼 만 하다. DNS 관리에 좋은(혹은 문제 해결에 좋은) 책으로는 *Robert M. Prisig*이 쓴 *Zen and the Art of Motorcycle Maintenance*이 있다. :-) ISBN은 0688052304이다. 그외 유용한 것들이 있다.

인터넷에서는 `<http://www.dns.net/dnsrd/>`, `<http://www.isc.org/bind.html>`에서 관련 내용을 찾을 수 있다. FAQ, 레퍼런스 매뉴얼(BOG; Bind Operations Guide), 기사, 프로토콜 정의, DNS 해킹 (전부는 아니지만, 이 문서와 rfcs 대부분이 bind 배포본에 포함되어 있다.) 필자는 이들 대부분을 읽지 보지 않았다. 어째든 필자는 전문적으로 DNS를 관리하는 관리자는 아니다. 반면 Arnt Gulbrandsen은 BOG를 읽었고 그 사실에 황홀해 한다. :-) `comp.protocols.tcp-ip.domains`가 DNS 관련 뉴스그룹이다. 또한, DNS에 관한 RFC가 많이 있다. 아마도 가장 중요한 것은 이것 들일 것이다.

RFC 2052

A. Gulbrandsen, P. Vixie, *A DNS RR for specifying the location of services (DNS SRV)*, October 1996

RFC 1918

Y. Rekhter, R. Moskowitz, D. Karrenberg, G. de Groot, E. Lear, *Address Allocation for Private Internets*, 02/29/1996.

RFC 1912

D. Barr, *Common DNS Operational and Configuration Errors*, 02/28/1996.

RFC 1912 Errors

B. Barr *Errors in RFC 1912*, this is available at `<http://www.cis.ohio-state.edu/~barr/rfc1912-errors.html>`

RFC 1713

A. Romao, *Tools for DNS debugging*, 11/03/1994.

RFC 1712

C. Farrell, M. Schulze, S. Pleitner, D. Baldoni, *DNS Encoding of Geographical Location*, 11/01/1994.

RFC 1183

R. Ullmann, P. Mockapetris, L. Mamakos, C. Everhart, *New DNS RR Definitions*, 10/08/1990.

RFC 1035

P. Mockapetris, *Domain names - implementation and specification*, 11/01/1987.

RFC 1034

P. Mockapetris, *Domain names - concepts and facilities*, 11/01/1987.

RFC 1033

M. Lottor, *Domain administrators operations guide*, 11/01/1987.

RFC 1032

M. Stahl, *Domain administrators guide*, 11/01/1987.

RFC 974

C. Partridge, *Mail routing and the domain system*, 01/01/1986.