第 3 章　域名系统——DNS&BIND

域名系统(Domain Name System,DNS)是 Internet 基础架构服务,在 Internet 上当一台主机要访问另外一台主机时,必须首先获知其地址,TCP/IP 中的 IP 地址是由 4 段以.分开的数字组成,记起来总是不如名字那么方便,所以,就采用了域名系统来管理名字和 IP 的对应关系。因此,DNS 是因特网的一项核心服务,它作为可以将域名和 IP 地址相互映射的一个分布式数据库,能够让用户更方便地访问互联网,而不用去记住能够被机器直接读取的 IP 数串。

实现 DNS 服务的软件包有多种,本章主要介绍 Linux 环境下 DNS 软件包——BIND(Berkeley Internet Name Domain)的安装、配置与管理。

3.1　BIND 概述

BIND 是 Linux 系统中实现 DNS 服务的软件包。几乎所有 Linux 发行版都包含 BIND,它的功能有了很大的改善和提高,已成为 Internet 上使用最多的 DNS 服务器软件。由于负责开发与维护开源 BIND 的互联网系统协会(Internet Systems Consortium,ISC)没有足够的资源维持项目的开发,所以在 2013 年发布了 BIND 的最后一个版本 10.1.2。

3.1.1　BIND 的安装

在 BIND 的发行版本中 BIND 9 的普及度最高,所以本书介绍在 CentOS 中 BIND 9 的安装。在 CentOS 中 BIND 的安装很容易,可以在系统安装阶段中选中 DNS 软件,也可以在系统安装完毕后再单独安装 BIND 软件包。CentOS 的软件包一般都采用 rpm 软件包的形式,可以从安装光盘中找到 BIND 软件包,也可以直接借助网络安装。本书采用 yum 命令直接下载安装 BIND,步骤如下。

安装之前首先使用 rpm 命令检测一下系统中是否已安装了 BIND。

```
[root@abc1 ~]#rpm -qa | grep BIND
samba-winBIND-3.6.23-12.el6.x86_64
samba-winBIND-clients-3.6.23-12.el6.x86_64
```

返回结果显示当前系统中没有安装 BIND 相关的组件。接下来我们借助网络安装的方式。

```
[root@abc1 ~]#yum -y install BIND BIND-chroot BIND-utils
```

安装过程如图 3-1 所示。

3.1.2　配置文件

在成功安装上述软件包后,还需要配置一些文件来实现域名解析。与域名解析相关

```
Loaded plugins: fastestmirror, security
Setting up Install Process
Determining fastest mirrors
 * base: mirrors.btte.net
 * extras: mirrors.nwsuaf.edu.cn
 * updates: mirrors.btte.net
base                                                     | 3.7 kB     00:00
extras                                                   | 3.4 kB     00:00
extras/primary_db                                        |  29 kB     00:00
updates                                                  | 3.4 kB     00:00
updates/primary_db                                       | 4.7 MB     00:06
Resolving Dependencies
--> Running transaction check
---> Package bind.x86_64 32:9.8.2-0.62.rc1.el6_9.4 will be installed
--> Processing Dependency: bind-libs = 32:9.8.2-0.62.rc1.el6_9.4 for package: 32
:bind-9.8.2-0.62.rc1.el6_9.4.x86_64
--> Processing Dependency: liblwres.so.80()(64bit) for package: 32:bind-9.8.2-0.
62.rc1.el6_9.4.x86_64
--> Processing Dependency: libisccfg.so.82()(64bit) for package: 32:bind-9.8.2-0
.62.rc1.el6_9.4.x86_64
--> Processing Dependency: libisccc.so.80()(64bit) for package: 32:bind-9.8.2-0.
62.rc1.el6_9.4.x86_64
--> Processing Dependency: libisc.so.83()(64bit) for package: 32:bind-9.8.2-0.62
.rc1.el6_9.4.x86_64
--> Processing Dependency: libdns.so.81()(64bit) for package: 32:bind-9.8.2-0.62
.rc1.el6_9.4.x86_64
--> Processing Dependency: libbind9.so.80()(64bit) for package: 32:bind-9.8.2-0.
62.rc1.el6_9.4.x86_64
---> Package bind-chroot.x86_64 32:9.8.2-0.62.rc1.el6_9.4 will be installed
---> Package bind-utils.x86_64 32:9.8.2-0.62.rc1.el6_9.4 will be installed
--> Running transaction check
---> Package bind-libs.x86_64 32:9.8.2-0.62.rc1.el6_9.4 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package          Arch        Version                          Repository    Size
================================================================================
Installing:
 bind             x86_64      32:9.8.2-0.62.rc1.el6_9.4        updates      4.0 M
 bind-chroot      x86_64      32:9.8.2-0.62.rc1.el6_9.4        updates       77 k
 bind-utils       x86_64      32:9.8.2-0.62.rc1.el6_9.4        updates      189 k
Installing for dependencies:
 bind-libs        x86_64      32:9.8.2-0.62.rc1.el6_9.4        updates      892 k

Transaction Summary
 ================================= ============================================
Install       4 Package(s)
Total download size: 5.1 M
Installed size: 10 M
Downloading Packages:
(1/4): bind-9.8.2-0.62.rc1.el6_9.4.x86_64.rpm            | 4.0 MB     00:05
(2/4): bind-chroot-9.8.2-0.62.rc1.el6_9.4.x86_64.rpm     |  77 kB     00:00
(3/4): bind-libs-9.8.2-0.62.rc1.el6_9.4.x86_64.rpm       | 892 kB     00:01
(4/4): bind-utils-9.8.2-0.62.rc1.el6_9.4.x86_64.rpm      | 189 kB     00:00
--------------------------------------------------------------------------------
Total                                          619 kB/s | 5.1 MB     00:08
Running rpm_check_debug
Running Transaction Test
Transaction Test Succeeded
Running Transaction
  Installing : 32:bind-libs-9.8.2-0.62.rc1.el6_9.4.x86_64                   1/4
  Installing : 32:bind-9.8.2-0.62.rc1.el6_9.4.x86_64                        2/4
  Installing : 32:bind-chroot-9.8.2-0.62.rc1.el6_9.4.x86_64                 3/4
  Installing : 32:bind-utils-9.8.2-0.62.rc1.el6_9.4.x86_64                  4/4
  Verifying  : 32:bind-libs-9.8.2-0.62.rc1.el6_9.4.x86_64                   1/4
  Verifying  : 32:bind-9.8.2-0.62.rc1.el6_9.4.x86_64                        2/4
  Verifying  : 32:bind-utils-9.8.2-0.62.rc1.el6_9.4.x86_64                  3/4
  Verifying  : 32:bind-chroot-9.8.2-0.62.rc1.el6_9.4.x86_64                 4/4

Installed:
  bind.x86_64 32:9.8.2-0.62.rc1.el6_9.4
  bind-chroot.x86_64 32:9.8.2-0.62.rc1.el6_9.4
  bind-utils.x86_64 32:9.8.2-0.62.rc1.el6_9.4

Dependency Installed:
  bind-libs.x86_64 32:9.8.2-0.62.rc1.el6_9.4

Complete!
```

图 3-1　BIND 安装过程

的文件有/etc/hosts、/etc/host.conf 和/etc/resolv.conf。下面分别介绍每个文件的主要功能。

1. /etc/hosts

hosts 文件是 Linux 系统中一个负责 IP 地址与域名快速解析的文件，以 ASCII 格式保存在/etc 目录下。hosts 文件包含了 IP 地址和主机名之间的映射，还包括主机名的别名。在没有域名服务器的情况下，系统上的所有网络程序都通过查询该文件来解析对应于某个主机名的 IP 地址，否则就需要使用 DNS 服务程序来解决。通常可以将常用的域名和 IP 地址映射加入到 hosts 文件中，实现快速方便的访问。利用 hosts 文件通信的主机都应该包含相同的记录，即所有主机都应该包含相同的 hosts 文件。随着网络中主机数量的不断增加，维护起来非常烦琐，因此，hosts 文件只适合于主机数量很少的网络。目前，虽然在系统中仍然可以看到 hosts 文件，但一般也仅限于本机解析环回地址使用，或少数几台主机通信。下面是 hosts 文件的主要内容：

IP 地址	本机默认域名	别名
127.0.0.1	localhost.localdomain	localhost

一般情况下，hosts 文件中只包含这一条记录。hosts 文件的每行为一个主机，每行由网络 IP 地址、主机名/域名、主机名别名 3 部分组成，每个部分由空格隔开。主机名通常在局域网内使用，通过 hosts 文件主机名就被解析到对应 IP；域名通常在 Internet 上使用，但如果本机不想使用 Internet 上的域名解析，就可以更改 hosts 文件，加入自己的域名解析。每行也可以是两部分，即主机 IP 地址和主机名，例如 192.168.1.100 test100。

2. /etc/host.conf

/etc/host.conf 是解析器配置文件，用来指定如何解析主机名。Linux 通过解析库(resolver library)来获得主机名对应的 IP 地址。

```
order hosts,BIND    #关键字 order 用来指明主机查询的顺序，先在 hosts 文件中查询，如
                     果未找到，再利用 DNS 查询
```

3. /etc/resolv.conf

/etc/resolv.conf 是 DNS 客户端的配置文件，用于设置 DNS 服务器的 IP 地址及 DNS 域名，还包含了主机的域名搜索顺序。该文件是由域名解析器(resolver，一个根据主机名解析 IP 地址的库)使用的配置文件。它的格式很简单，每行以一个关键字开头，后接一个或多个由空格隔开的参数。

```
domain test.com                 #定义本地域名
search test.com                 #定义域名的搜索列表
nameserver 192.168.1.111        #服务器 IP 地址
nameserver 192.168.1.112        #服务器备用 IP 地址
```

3.2　BIND 主配置文件

BIND 的主配置文件 named. conf 和/etc/named. rfc1912. zones 用来实现域名服务的关键配置文件，包括 DNS 服务器的类型、特性、区域的名称等。

3.2.1　named. conf 的默认配置

named. conf 是 DNS 服务器的全局配置文件，由一个个子句组成，每个子句都有一个头跟一对大括号组成，大括号里面是该子句中的因子和值。下面的代码是 named. conf 的一个默认配置，其中//为注释符号。

```
[root@abc1 ~]#cat /etc/named.conf
//
// named.conf
//
// Provided by Red Hat BIND package to configure the ISC BIND named(8) DNS
// server as a caching only nameserver (as a localhost DNS resolver only).
//
// See /usr/share/doc/BIND*/sample/ for example named configuration files.
//

options {
listen-on port 53 { 127.0.0.1; };
listen-on-v6 port 53 { ::1; };
directory   "/var/named";                       //指定区域数据库文件存放的位置
dump-file   "/var/named/data/cache_dump.db"; //指定转储文件存放的位置及文件名
       statistics-file "/var/named/data/named_stats.txt";
       memstatistics-file "/var/named/data/named_mem_stats.txt";
allow-query     { localhost; };                 //允许哪些主机可以查询本服务器
recursion yes;                                  //开启递归解析

dnssec-enable yes;                              //开启 DNS 安全特性
dnssec-validation yes;                          //开启 DNS 安全合法性检验

/* Path to ISC DLV key */
BINDkeys-file "/etc/named.iscdlv.key";

managed-keys-directory "/var/named/dynamic";
};

logging {
           channel default_debug {
```

```
            file "data/named.run";
            severity dynamic;
        };
};

zone "." IN {                          //关键字 zone 用来定义区域,此处定义根"."区域
  type hint;                           //定义区域类型为提示类型
  file "named.ca";                     //指定该区域的数据库文件为 named.ca
};

include "/etc/named.rfc1912.zones"; //将区域定义文件包含进来,关键文件
include "/etc/named.root.key";
```

3.2.2 修改 named.conf

1. 修改监听的端口号和 IP

```
listen-on port 53 { any; };              //监听来自任意 IP 地址上 53 号端口的数据包
listen-on port 53 { 192.168.233.128; }; //监听来自 IP192.168.233.128 上 53 号端口
                                           的数据包
```

2. 查询本服务器的主机列表

```
allow-query     { any; };                //允许任意主机进行 DNS 查询
```

3.2.3 默认区域配置文件/etc/named.rfc1912.zones

/etc/named.rfc1912.zones 是 DNS 服务器的区域配置文件,默认配置代码如下所示。

```
// named.rfc1912.zones:
//
// Provided by Red Hat caching-nameserver package
//
// ISC BIND named zone configuration for zones recommended by
// RFC 1912 section 4.1 : localhost TLDs and address zones
// and afts/draft-ietf-dnsop-default-local-zones-02.txt
// (c)2007 R W Franks
//
// See /usr/share/doc/BIND*/sample/ for example named configuration files.
//

zone "localhost.localdomain" IN {  //定义区域 localhost.localdomain
type master;                       //定义区域类型为主要类型
file "named.localhost";            //指定该区域的数据库文件为 named.localhost
allow-update { none; };            //不允许更新
```

```
};

zone "localhost" IN {                    //定义区域 localhost
type master;                             //定义区域类型为主要类型
file "named.localhost";                  //指定该区域的数据库文件为 named.localhost
allow-update { none; };                  //不允许更新
};

zone "1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.ip6.
arpa" IN {
//定义反向区域,该区域为 IPv6 的地址表示方法
type master;                             //定义区域类型为主要类型
file "named.loopback";                   //指定该区域的数据库文件为 named.loopback
allow-update { none; };                  //不允许更新
};

zone "1.0.0.127.in-addr.arpa" IN {       //定义反向区域 1.0.0.127.in-addr.arpa
type master;                             //定义区域类型为主要类型
file "named.loopback";                   //指定该区域的数据库文件为 named.loopback
allow-update { none; };                  //不允许更新
};

zone "0.in-addr.arpa" IN {               //定义反向区域 0.in-addr.arpa
type master;                             //定义区域类型为主要类型
file "named.empty";                      //指定该区域的数据库文件为 named.empty
allow-update { none; };                  //不允许更新
};
```

3.2.4　自定义区域

DNS 服务器是以区域为单位来进行管理的,用 zone 关键字来定义区域。一个区域是一个连续的域名空间,区域名称一般用双引号引起来。下面自定义一个区域:区域名为 test.com,区域类型为 master 类型,区域数据库文件名为 test.com.zone。zone 段配置代码如下。

```
zone "test.com" IN {
        type master;
        file "test.com.zone";
};
```

区域 test.com 为正向区域,同样的方法,再来定义一个用来实现 IP 地址到域名翻译的反向区域。反向区域的名称是网络地址的“逆序”,即若反向区域的 IP 是 192.168.233.128,那么反向区域的名称应该是 233.168.192.in-addr.arpa,其中.in-addr.arpa 必

须加上，arpa 表示反向域名空间的顶级域名，in-addr 表示 arpa 的下一级域名。下面自定义一个反向区域：区域名 233.168.192.in-addr.arpa，区域类型为 master 类型，区域数据库文件名为 test.com.rezone。zone 段配置代码如下。

```
zone "233.168.192.in-addr.arpa" IN {
    type master;
    file "test.com.rev";
};
```

3.3 正向区域数据库文件

上面定义了区域 test.com，接下来在/var/named 目录下需要定义区域数据库的文件 test.com.zone，配置代码如下。

```
$TTL 86400
@  IN  SOA dns.test.com. admin.test.com.(
  201810101
       1800
       3600
       604800
       86400
)
         IN  NS  dns.test.com.
dns IN   A   192.168.233.128
www      IN  A  192.168.233.127
mail IN  A   192.168.233.126
ftp  IN  CNAME  www
```

文件中首先定义了 SOA 类型的资源记录。一般情况下，SOA(Start of Authority)起始授权记录是区域数据库文件中第一条资源记录，用来表示某区域的授权服务器是哪台主机及相关参数。SOA 记录的含义如表 3-1 所示。

表 3-1 SOA 记录的含义

标　识	含　义
@	区域名称
IN	Internet 类
SOA	起始授权类型，将某区域授权给某台服务器
serial	区域数据库文件的版本号
refresh	刷新间隔，即辅 DNS 与主 DNS 服务器的同步时间间隔
retry	重试间隔
expiry	过期时间
minmum	最小生存期

文件后面还定义了几种典型的正向资源记录，具体标识含义如表 3-2 所示。

表 3-2　zheng'xiang 记录的含义

标　　识	含　　义
NS 记录	名字服务器类型的记录
A 记录	主机类型的记录
CNAME 记录	别名记录

使用 named-checkconf 和 named-checkzone 命令，来检查配置文件的语法是否有错误。

```
[root@abc1 ~]named-checkconf /etc/named.conf
[root@abc1 ~]#named-checkzone "test.com.zone" /var/named/test.com.zone
zone test.com.zone/IN: loaded serial 201810101
OK
```

3.4　反向区域数据库文件

正向区域数据库文件定义好后，接下来建立反向区域文件。

```
[root@abc1 ~]#vi /var/named/test.com.rev
$ TTL 86400
@  IN   SOA dns.test.com. admin.test.com.(
            201810101
            1800
            3600
            604800
            86400
)
IN    NS  dns.test.com.
128   IN  PTR dns.test.com.
127   IN  PTR www.test.com.
126   IN  PTR mail.test.com.
ftp   IN  CNAME  www
```

一般情况下，正、反向区域数据库文件的 SOA 与 NS 记录是相同的，所以只需用 PTR 类型来定义由 IP 地址到域名翻译的记录即可。

3.5　测试 DNS 服务

DNS 服务器端配置好后，接下来开始运行及测试工作。

1. 重启 DNS 服务

```
[root@abc1 ~]#service named restart
Stopping named:                                [ OK ]
```

```
Starting named:                                [ OK ]
```

2. 测试 DNS 服务

常用的测试 DNS 服务命令有 3 种，分别是 nslookup、host 和 dig。

1）nslookup 命令

nslookup 命令后跟上要解析的域名或 IP 地址。

```
[root@abc1 ~]#nslookup dns.test.com
Server:        192.168.233.128
Address:  192.168.233.128#53

Name:    dns.test.com
Address: 192.168.233.128
```

2）host 命令

host 命令后跟上要解析的域名或 IP 地址，也可加上-a 参数，列出详细信息。

```
[root@abc1 ~]#host www.test.com
www.test.com has address 192.168.233.127
[root@abc1 ~]#host -a www.test.com
Trying "www.test.com"
;; -> > HEADER<<-opcode: QUERY, status: NOERROR, id: 63233
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 1

;; QUESTION SECTION:
;www.test.com.      IN  ANY

;; ANSWER SECTION:
www.test.com.    86400  IN  A  192.168.233.127

;; AUTHORITY SECTION:
test.com.    86400  IN  NS  dns.test.com.

;; ADDITIONAL SECTION:
dns.test.com.    86400  IN  A  192.168.233.128

Received 80 bytes from 192.168.233.128#53 in 0 ms
```

3）dig 命令

dig 命令后跟上要解析的域名，来显示从受请求的域名服务器返回的答复。如要进行反向查询需要加上-x 参数。

```
[root@abc1 ~]#dig mail.test.com

; <<>>DiG 9.8.2rc1-RedHat-9.8.2-0.62.rc1.el6_9.4 <<>>mail.test.com
```

```
;; global options: + cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 47984
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 1

;; QUESTION SECTION:
;mail.test.com.        IN  A

;; ANSWER SECTION:
mail.test.com.    86400  IN  A  192.168.233.126

;; AUTHORITY SECTION:
test.com.    86400  IN  NS  dns.test.com.

;; ADDITIONAL SECTION:
dns.test.com.    86400  IN  A  192.168.233.128

;; Query time: 0 msec
;; SERVER: 192.168.233.128#53(192.168.233.128)
;; WHEN: Sat Feb 3 21:07:33 2018
;; MSG SIZE rcvd: 81

[root@abc1 ~]#dig -x 192.168.233.128

; <<>>DiG 9.8.2rc1-RedHat-9.8.2-0.62.rc1.el6_9.4 <<>>-x 192.168.233.128
;; global options: + cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 18857
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 1

;; QUESTION SECTION:
;128.233.168.192.in-addr.arpa.  IN  PTR

;; ANSWER SECTION:
128.233.168.192.in-addr.arpa. 86400 IN  PTR  dns.test.com.

;; AUTHORITY SECTION:
233.168.192.in-addr.arpa. 86400  IN  NS  dns.test.com.

;; ADDITIONAL SECTION:
dns.test.com.    86400  IN  A  192.168.233.128

;; Query time: 0 msec
;; SERVER: 192.168.233.128#53(192.168.233.128)
```

```
;; WHEN: Sat Feb 3 21:09:06 2018
;; MSG SIZE rcvd: 102
```

3.6 辅 DNS

主 DNS 服务器(master DNS)保存的是某个区域的数据,并对此区域数据是可读可写的,即可以在该服务器上直接修改区域数据库的内容。而辅 DNS 服务器(slave DNS)保存的是某个区域的辅助版本(只读版本),它只能提供查询服务而不能在该服务器上修改该区域的内容。一般来说,辅 DNS 服务器一是作为主 DNS 服务器的备份,二是分担主 DNS 服务器的负载。

3.6.1 辅 DNS 的配置

辅 DNS 服务器的区域数据库文件是从主 DNS 服务器复制过来的,所以配置辅 DNS 服务器只需编辑 DNS 的区域配置文件即可,无须建立区域数据库文件。

1. 配置辅 DNS

配置辅 DNS 的区域配置文件,zone 段代码如下。

```
[root@abc2 ~]#vi /etc/named.rfc1912.zones
zone "test.com" IN {
type slave;
file "slaves/test.com.zone.slave";
masters {192.168.233.128;};
};

zone "233.168.192.in-addr.arpa" IN {
type slave;
file "slaves/test.com.rev.slave";
masters {192.168.233.128;};
};
```

注意,这两个区域的名称应与主 DNS 服务器中区域的名称一致。

2. 配置主 DNS

首先在主 DNS 的区域配置文件里加上辅 DNS 的信息,zone 段代码如下。

```
[root@abc1 ~]#vi /etc/named.rfc1912.zones
zone "test.com" IN {
type master;
file "test.com.zone";
allow-transfer {192.168.233.129;};
};
```

```
zone "233.168.192.in-addr.arpa" IN {
type master;
file "test.com.rev";
allow-transfer {192.168.233.129;};
};
```

然后分别在正、反向区域数据库文件里加上辅 DNS 的 NS 记录，代码如下。

```
[root@abc1 ~]#vi /var/named/test.com.zone
$TTL 86400
@  IN  SOA dns.test.com. admin.test.com.(
       201810101
       1800
       3600
       604800
       86400
)
         IN  NS  dns.test.com.
         IN  NS  dns2.test.com.
dns IN   A   192.168.233.128
www      IN  A  192.168.233.127
mail IN  A   192.168.233.126
dns2 IN  A   192.168.233.129
ftp  IN  CNAME  www
[root@abc1 ~]#vi /var/named/test.com.rev
$TTL 86400
@  IN  SOA dns.test.com. admin.test.com.(
           201810101
           1800
           3600
           604800
           86400
)
IN  NS  dns.test.com.
IN  NS  dns2.test.com.
128  IN  PTR  dns.test.com.
129  IN  PTR  dns2.test.com.
127  IN  PTR  www.test.com.
126  IN  PTR  mail.test.com.
ftp  IN  CNAME  www
```

修改好后，重新运行 DNS 服务。

```
[root@abc1 ~]#service named restart
Stopping named:                            [ OK ]
Starting named:                            [ OK ]
```

3. 启动辅 DNS 服务

```
[root@abc2 ~]#service named start
Starting named:                                    [ OK ]
```

此时,辅 DNS 服务器已从主 DNS 服务器上将正、反区域数据库文件复制到了/var/named/slaves/目录下。

```
[root@abc2 ~]#ls /var/named/slaves
test.com.rev.slave test.com.zone.slave
```

3.6.2 测试辅 DNS

在辅 DNS 上,利用 dig 命令正反向查询来测试辅 DNS 服务器。

```
[root@abc2 ~]#dig -t A www.test.com @192.168.233.128

; <<>>DiG 9.8.2rc1-RedHat-9.8.2-0.62.rc1.el6_9.4 <<>>-t A www.test.com @192.
168.233.128
;; global options: + cmd
;; Got answer:
;; ->>HEADER<<-opcode: QUERY, status: NOERROR, id: 45445
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 2

;; QUESTION SECTION:
;www.test.com.      IN  A

;; ANSWER SECTION:
www.test.com.    86400  IN  A  192.168.233.127

;; AUTHORITY SECTION:
test.com.    86400  IN  NS  dns2.test.com.
test.com.    86400  IN  NS  dns.test.com.

;; ADDITIONAL SECTION:
dns.test.com.    86400  IN  A  192.168.233.128
dns2.test.com.   86400  IN  A  192.168.233.129

;; Query time: 0 msec
;; SERVER: 192.168.233.128#53(192.168.233.128)
;; WHEN: Sat Feb 3 23:49:26 2018
;; MSG SIZE rcvd: 115

[root@abc2 ~]#dig -x 192.168.233.129 @192.168.233.128
```

```
; <<>>DiG 9.8.2rc1-RedHat-9.8.2-0.62.rc1.el6_9.4 <<>>-x 192.168.233.129 @
192.168.233.128
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<-opcode: QUERY, status: NOERROR, id: 27363
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 2

;; QUESTION SECTION:
;129.233.168.192.in-addr.arpa.  IN  PTR

;; ANSWER SECTION:
129.233.168.192.in-addr.arpa. 86400 IN  PTR  dns2.test.com.

;; AUTHORITY SECTION:
233.168.192.in-addr.arpa. 86400  IN  NS  dns.test.com.
233.168.192.in-addr.arpa. 86400  IN  NS  dns2.test.com.

;; ADDITIONAL SECTION:
dns.test.com.    86400  IN  A  192.168.233.128
dns2.test.com.   86400  IN  A  192.168.233.129

;; Query time: 0 msec
;; SERVER: 192.168.233.128#53(192.168.233.128)
;; WHEN: Sat Feb 3 23:51:09 2018
;; MSG SIZE rcvd: 137
```

从测试结果可以看出，辅 DNS 已经正常工作。

3.7　子　　域

子域(Subdomain)是域名层次结构中的一个术语，是对某一个域进行细分时的下一级域，从而实现 DNS 的层次化和分布式。例如，test.com 是一个顶级域名，可以把 sub.test.com 配置成是它的一个子域。配置子域可以有两种方式，一种是把子域配置放在另一台 DNS 服务器上，即子域和父域在不同的 DNS 服务器上。还有一种是子域配置与父域配置放在一起，即子域和父域在同一台 DNS 服务器上。

3.7.1　父子域在同一台 DNS 服务器上

父子域在同一台 DNS 服务器上的情况又称为虚拟子域，配置方法如下。

1. 配置父域的正向区域数据库

在父域的正向区域数据库中增加子域的记录，配置代码如下。

```
$TTL 86400
```

```
@  IN  SOA dns.test.com. admin.test.com.(
201810101
       1800
       3600
       604800
       86400
)
         IN  NS  dns.test.com.
         IN  NS  dns2.test.com.
dns IN   A   192.168.233.128
www      IN  A  192.168.233.127
mail IN  A   192.168.233.126
dns2 IN  A   192.168.233.129
dns.sub1.test.com.  IN  A  192.168.233.130  //以下两行代码为直接增加的子域记录
sub1.test.com.  IN  CNAME  dns.sub1.test.com.
ftp  IN  CNAME  www
```

2. 配置父域的反向区域数据库

在父域的反向区域数据库中增加子域的 PRT 记录,配置代码如下。

```
$TTL 86400
@  IN  SOA dns.test.com. admin.test.com.(
           201810101
           1800
           3600
           604800
           86400
)
IN  NS  dns.test.com.
IN  NS  dns2.test.com.
128 IN  PTR dns.test.com.
129 IN  PTR dns2.test.com.
127 IN  PTR www.test.com.
126 IN  PTR mail.test.com.
130 IN  PTR dns.sub1.test.com.      //以下两行代码为直接增加子域的 PRT 记录
130 IN  PTR sub1.test.com.
ftp IN  CNAME www
```

3. 测试子域

重新启动 named 进程,进行测试。

```
[root@abc1 ~]# service named restart
Stopping named:                            [ OK ]
```

```
Starting named:                                [ OK ]
[root@abc1 ~]#nslookup
>sub.test.com
Server:        192.168.233.128
Address:  192.168.233.128#53

sub.test.com  canonical name =dns.sub.test.com.
Name:  dns.sub.test.com
Address: 192.168.233.130
>192.168.233.130
Server:    192.168.233.128
Address:  192.168.233.128#53

130.233.168.192.in-addr.arpa  name =sub.test.com.
130.233.168.192.in-addr.arpa  name =dns.sub.test.com.
>exit
```

从测试结果可以看出，子域建立成功。

3.7.2　父子域在不同 DNS 服务器上

父子域在不同的 DNS 服务器上更能体现出分布式的特点，这种情况又称为区域委派，即父域将子域委派给另一台机器来管理。一般情况下，只需要做正向区域委派，很少进行反向区域委派，所以下面介绍一下正向区域配置方法。

区域委派的实现分为两部分，一是在父域 DNS 服务器的区域数据库文件中设置指向子域的记录；二是在子域 DNS 服务器上建立该子域的数据库文件。其中第二部分就是建立子域的主 DNS 服务器。

例如，建立区域 test.com 的子域 sub2.test.com，其中父域 DNS 服务器的 IP 地址为 192.168.233.128，子域的 DNS 服务器为 192.168.233.131。配置方法如下。

1. 配置父域的正向区域数据库

在父域的正向区域数据库中增加指向子域 DNS 的记录，配置代码如下。

```
[root@abc1 /]#vi /var/named/test.com.zone
$TTL 86400
@  IN  SOA dns.test.com. admin.test.com.(
201810101
       1800
       3600
       604800
       86400
)
IN  NS   dns.test.com.
IN  NS   dns2.test.com.
```

```
dns IN   A  192.168.233.128
www  IN  A  192.168.233.127
mail IN  A  192.168.233.126
dns2 IN  A  192.168.233.129
dns.sub1.test.com. IN  A  192.168.233.130
sub1.test.com.  IN  CNAME  dns.sub1.test.com.
sub2.test.com.  IN  NS  dns.sub2.test.com.   //定义子域的 DNS 服务器是
                                                dns.sub2.test.com.
dns.sub2.test.com.  IN  A  192.168.233.131   //增加子域的 DNS 记录
ftp  IN  CNAME  www
```

2. 配置子域的 DNS 服务器

在子域的 DNS 服务器(192.168.233.131)上编辑主配置文件,定义区域. sub2. test. com,配置代码如下。

```
zone "sub2.test.com" IN {
type master;
file "sub2.test.com.zone";
};
```

3. 在子域的 DNS 服务器上创建子域的正向区域数据库文件 sub2. test. com. zone

配置代码如下。

```
[root@abc3 named]#vi /var/named/sub2.test.com.zone
$TTL 86400
@  IN  SOA dns.sub2.test.com. admin.sub2.test.com.(
201810105
       1800
       3600
       604800
       86400
)
IN  NS  dns.sub2.test.com.
dns.sub2.test.com.  IN  A  192.168.233.131
www  IN  A  192.168.233.131
```

4. 测试子域

子域重新启动 named 进程。

```
[root@abc3 ~]#service named restart
Stopping named:                          [ OK ]
Starting named:                          [ OK ]
```

父域重新启动 named 进程。

```
[root@abc1 ~]#serv: ice named restart
Stopping named:                            [ OK ]
Starting named:                            [ OK ]
```

在父域 DNS 服务器上进行测试。

```
[root@abc1 /]#dig -t A www.sub2.test.com @192.168.233.128

; <<>>DiG 9.8.2rc1-RedHat-9.8.2-0.62.rc1.el6_9.5 <<>>-t A www.sub2.test.com
@192.168.233.128
;; global options: + cmd
;; Got answer:
;; ->>HEADER<<-opcode: QUERY, status: NOERROR, id: 5250
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 1

;; QUESTION SECTION:
;www.sub2.test.com.      IN  A

;; ANSWER SECTION:
www.sub2.test.com.  86400  IN  A  192.168.233.131

;; AUTHORITY SECTION:
sub2.test.com.    86400  IN  NS  dns.sub2.test.com.

;; ADDITIONAL SECTION:
dns.sub2.test.com.  86400  IN  A  192.168.233.131

;; Query time: 1 msec
;; SERVER: 192.168.233.128#53(192.168.233.128)
;; WHEN: Thu Apr 12 01:32:29 2018
;; MSG SIZE rcvd: 85
```

通过以上步骤实现了正向区域委派，从测试结果可以看出子域建立成功。

3.8　高 级 配 置

前面讨论了 DNS 服务器的基本配置内容，下面将介绍几个典型的 BIND 高级配置。

3.8.1　DNS 转发

当客户提出查询请求时，首先向自己首选 DNS 发出解析请求，如果该首选 DNS 被配置为 forwarding DNS server，那么 forwarding DNS server 会先在缓存和本地区域数据库中进行查询，若未找到相应记录，则将请求转发给 forwarder DNS server，让 forwarder DNS server 替 forwarding DNS server 进行解析。如果 forwarder DNS server 能够查到结果，则将结果返还给 forwarding DNS server，并且自己缓存一份；接着，由 forwarding

DNS server 将得到的结果应答给客户，并且自己也缓存一份，解析成功完成。如果 forwarder DNS server 没有查到结果，则将“未找到”返还给 forwarding DNS server；接着，forwarding DNS server 自己再进行一次解析，这次不再利用 forwarder DNS server，如果还不能找到，则 forwarder DNS server 返还给客户“未找到”。

显然，forwarding DNS server 是要利用 forwarder DNS server 进行解析的，这样可以在很大程度上减少 forwarding DNS server 的工作量。配置使用转发器 forwarder，实际上就是把自己变成 forwarding DNS server。

上一小节中，区域委派时只能在父域中查找到子域的记录，而子域中找不到父域的记录。如果想让子域解析父域，可以在子域的配置文件(/etc/namd. conf)中设置转发区域指向父域，这样就可以解析到父域中的主机了。主配置文件中原始内容不变，只增加一个区域文件即可，因为转发不需要数据文件。

在子域的 DNS 服务器(192. 168. 233. 131)上编辑文件/etc/named. conf，增加 zone 段代码配置如下。

```
zone "test.com" IN {
        type forward;
        forwarders{192.168.233.128;};
};
```

子域和父域重新启动 named 进程后，在子域上测试。

```
[root@abc3 named]#dig -t A www.test.com @192.168.233.128

; <<>>DiG 9.8.2rc1-RedHat-9.8.2-0.62.rc1.el6_9.5 <<>>-t A www.test.com @192.
168.233.128
;; global options: + cmd
;; Got answer:
;; ->>HEADER<<-opcode: QUERY, status: NOERROR, id: 53245
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 2

;; QUESTION SECTION:
;www.test.com.          IN  A

;; ANSWER SECTION:
www.test.com.    86400  IN  A  192.168.233.127

;; AUTHORITY SECTION:
test.com.    86400  IN  NS  dns.test.com.
test.com.    86400  IN  NS  dns2.test.com.

;; ADDITIONAL SECTION:
dns.test.com.    86400  IN  A  192.168.233.128
dns2.test.com.    86400  IN  A  192.168.233.129
```

```
;; Query time: 0 msec
;; SERVER: 192.168.233.128#53(192.168.233.128)
;; WHEN: Mon Apr 16 02:15:26 2018
;; MSG SIZE rcvd: 115
```

从以上结果可以看出，将父域 DNS 服务器作为子域 DNS 服务器的 forwarder DNS server 后，就可以实现在子域 DNS 服务器上“成功解析”父域 DNS 服务器中的记录了。

3.8.2　负载均衡

随着网络的规模越来越大，网络服务器的负担也变得越来越重。一台服务器要同时应付成千上万用户的并发访问，必然会导致服务器过度繁忙，甚至运行不稳定。为了解决这个问题，可以在 DNS 服务器上配置负载均衡功能。

DNS 负载均衡是在 DNS 服务器中为同一个域名配置多个 IP 地址（为一个主机名设置多条 A 资源记录），在应答 DNS 查询时，DNS 服务器对每个查询将以 DNS 文件中主机记录的 IP 地址按顺序返回不同的解析结果，将客户端的访问引导到不同的机器上去，使得不同的客户端访问不同的服务器，从而达到负载均衡的目的。这就是实现负载均衡的最简单的方法——轮询。下面编辑正向区域数据库文件，为域 www. test. com 增加两个 IP 地址，配置代码如下。

```
[root@abc1 named]#vi /var/named/test.com.zone
$TTL 86400
@  IN  SOA dns.test.com. admin.test.com.(
201810101
       1800
       3600
       604800
       86400
)
IN  NS  dns.test.com.
IN  NS  dns2.test.com.
dns IN  A  192.168.233.128
www IN  A  192.168.233.127    //以下两行为域 www.test.com 增加两个 IP 地址
    IN  A  192.168.233.132
    IN  A  192.168.233.133
mailIN  A  192.168.233.126
dns2IN  A  192.168.233.129
dns.sub1.test.com.  IN  A  192.168.233.130
sub1.test.com.  IN  CNAME  dns.sub1.test.com.
sub2.test.com.  IN  NS  dns.sub2.test.com.
dns.sub2.test.com.  IN  A  192.168.233.131
ftp  IN  CNAME  www
```

重新启动 named 进程后，测试。