DNS 服务器

## DNS 起源与背景

### 1.1、 Internet 简史

internel网的祖先Advanced Research Projects Agrency Network(ARPANET)在1996年建立，其建立的初衷是使美国政府研究中心能够更快的共享信息。随后越来越多的研究中心链接到ARPANET中，到20世纪未，共有320台电脑连接到这个网络中。

他的发展看起来很有希望，但是有一个缺陷，缺少扩展性。ARPANEET使用一个名为HOSTS.TXT（1）的文件就能容纳所有了解的主机信息，它包含所有连接到ARPANET的主机名字到地址的映射(name-to-address mapping)。如果有更多的计算机连接到ARPANET，它变的很难去维护与变更，而且还会出现很多问题（流量负载，名字冲突，一致性）。

HOST.TXT文件是由SRI的网络信息中心（network Information Center, NIC）负责维护，并且从一台主机SRI-NIC上分发到整个网络

### 1.2、 DNS历史

早在ARPANET工作之前，Paul Mockapetris就早已经开发出了高效的计算机文件系统，不过因为在MIT学院仅可以使用小机器，工作受到限制，后来在ARPANET他借此机会奖自己的难住转化为现实，在Jon Poste和 Zaw-Sing Su的帮助下，他在1984年提出DNS管理命名规范，包括下面两点

名字， 比如 ibm

分类， 比如 .com

过了几年，这个分类或者叫gTLDs ( gerneric Top-Level Domains) 被建立起来，包括了一些大家熟悉的扩展名比如： .com, .net, .org, .gov 。 第一个注册的域名为Symbolics.com，至今仍然在用。

随着互联网扩展到欧洲，南美洲，亚洲，更多的用户开始采用DNS，美国政府先聘请网络解决公司（NSI）免费分通用顶级域名（gTLD），但是后来授予该公司独家授权的权限，到了1995年，域名不再免费，注册一个两年期限的域名需要交100美元。这个价格是严重高估的，因而呼起规范与监控全球的DNS。在这环境下成立了互联网名称与数据地址分配机构（Internet Corparation for Assigned Names and Numbers, ICANN）的非营利组织，它向其他公司开放域名注册。

DNS （Domain Name System） 中包含了用来按照一种分层结构定义了Internet上使用的主机名字的语法，还有名字的授权规则，定义域名和IP地址对应等一些设置。 实际上，DNS 是一个分布式数据库。它允许对整个数据库的各个部分进行本地控制；同时整个网络也能通过客户……服务器方式访问每个部分的数据，借助备份和缓存机制， DNS 将更强壮和足够的性能。使用的是TCP或者UDP传输数据，战胜端口53。

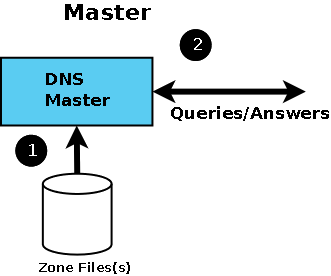
## DNS 分类

大多数DNS服务器是分散的，他们可以是主服务器（需要授权），也可以是从服务器，或者缓存服务器。需要注意两点

1. DNS服务器支持递归（resursive）查询是一个非常坏的想法，可能会受到DDOS攻击造成缓存急速增加的风险。这里可以对一个区域中的IP来允许递归查询
2. 以最小化为原则，一个DNS服务器只提供一个服务，比如只授权（authoritative）,只缓存(caching only)。不能把两个服务同时放在同一个服务器中，这是对的并且理想的状态，但是并不是那么容易达到，因为在大的组织中经常会使用混合的DNS服务，因为下层服务会特别灵活而增加风险。

### 2.1 Master (Primary) Name Servers

一个主DNS服务定义了一个或多个区文件（zone files），这台DNS是权威的，该区域已经被委派（通过NS资源记录）到这个DNS。



主服务器状态是在区域文件中声明，

// example.com fragment from named.conf

// defines this server as a zone master

zone "example.com" in{

type master;

file "pri.example.com";

};

这里需要注意的是

1. 在 Windows TCP/IP属性中的主次DNS条目并不意味着什么，它们可能反应主从名称服务器，也可能不会，这根据操作决定，而不是BIND配置
2. 最主要的是主DNS服务器是通过本地获取zone 数据。但是对应的从DNS服务器获取zone数据是从网络中获取（主要是从master获取，也有可能不是）

当主DNS接收到其权威的区域查询时，它将作为权威进行响应（在查询响应中设置AA位），当收到 不属于主服务器或者从服务器的区域查询，那么它将通过配置来进行操作

1. 如果允许缓存行为，并且允许递归查询，刚服务器将完全回答请求或者返回错误
2. 如果允许缓存行为，并且允许迭代查询，刚服务器可以完整回答请求（如果已经在缓存中）进行响应，或者返回错误
3. 如果不允许缓存（仅限权威DNS服务器）则服务器将返回引用错误。

一个主DNS服务可以通过设置 NOTIFY zone来设定从服务。NOTIFY 消息是为了保证zone修改后迅速传到从服务器，而不是从周期性的请求主是否有变动。BIND 默认会通知有该区域中有NS 记录服务器，除了自身。

一个区域服务也可以隐藏（仅有几个slave知道他是存在的），这个配置中主服务器不需要出现在域的NS RR中，唯一的要求是多个name sserver 支持该区域。

// example.com fragment from named.conf

// defines this server as a zone master

// 192.168.0.2 is a stealth server NOT listed in a NS record

zone "example.com" in{

type master;

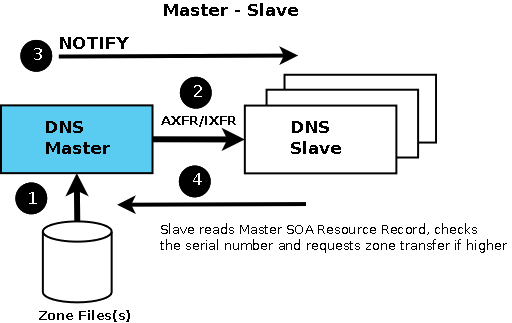
also-notify {192.168.0.2;};

file "pri/pri.example.com";

};

### 2.2 Secondary Name servers

从DNS服务器是从来自主服务器获取区域数据。并且对定义为从的区域做权威响应。



从服务器的角色也是在域名中声明的

// example.com fragment from named.conf

// defines this server as a zone slave

zone "example.com" in{

type slave;

file "sec/sec.example.com";

masters {192.168.23.17;};

};

主服务器在区域中的SOA记录的expiry参数定义了从服务器多久刷新一个域名记录。如果当expiry时间到了后主服务器不能响应，那么从服务器也不会再响应区域的请求，它不会使用过期的数据。

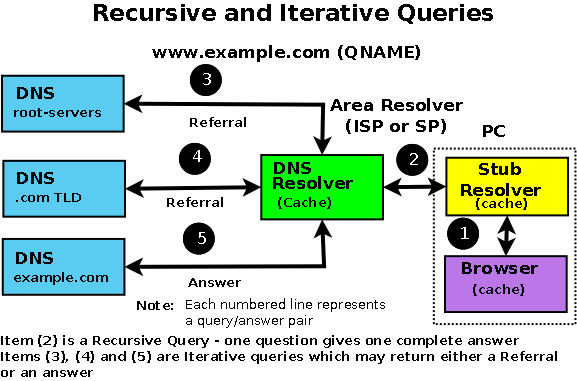
file参数是可选的，这是允许从把数据缓存到本地，当从服务器重启后在时间未过期的情况下会使用本地数据，这样会节省网络带宽。

从服务也可以变为主，有些出于安全考虑，需要把主服务器隐藏，而把多个从服务器置于防火墙之外，多个从可以是从－从配置，

### 2.3 Caching Name Servers

DNS缓存服务器也会被叫做 Resolver ，从区域中的主服务器获取数据并缓存到本地以供主机查询，直到TTL过期，然后重新从区域主机主获取数据。如果一个缓存服务器直接从主DNS服务器获取数据，那么他的请求便是权威的，如果数据是从缓存中提供的，那么便是非权威的。

缓存数据器需要同时开始递归查询和迭代查询，递归查询需要通过 ‘type hint’ 语句提供根服务器 。 他们的区别见下图。



// options section fragment of named.conf

// recursion yes is the default and may be omitted

options {

directory "/var/named";

version "not currently available";

recursion yes;

};

// zone section

....

// the DOT indicates the root domain = all domains

zone "." IN {

type hint;

file "root.servers";

};

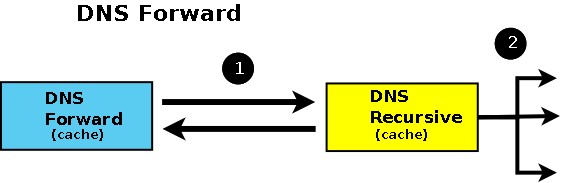
需要注意几点

1. Bind默认是支持递归查询的，在配置中的 recursion 参数就是控制这个行为的
2. 区域“.” 是根服务器的缩写，意思是任何domain包括master或者slave
3. 重启bind后缓存失效

### 3.4 Forwarding(PROXY) Name Servers ( a.k.a Proxy)

一个转发服务器仅是把请求转发到另一台DNS或者缓存DNS服务器，以下情况下可以使用代理

1. 访问外部网络非常慢，或者非常昂贵
   1. 本地DNS缓存－ 结果会缓存在代理服务器上，所以请求的damain会非常快速的得到结果
   2. 远程（转发到）DNS服务器提供递归查询支持，从而减少流量拥塞
2. 转发服务器还可用于通过提供可以管理远程名称服务器的更改的单个点来减轻本地管理的负担，而不必更新所有主机。 因此，特定网络部分或区域中的所有主机可以被配置为指向固定转发DNS，其可以被配置为根据需要流式传输DNS流量，并且以最小的努力随时间而改变。
3. 给交通消毒，特别的在大的私有网络，
4. 转发也可以用作外围防御的拆分服务器配置的一部分。



在BIND中的GLOBAL中或者在每个zone里配置中的forward forwarders 两个参数用来配置转发服务器，

全局转发所有请求

// options section fragment of named.conf

// forwarders can have multiple choices

options {

directory "/var/named";

version "not currently available";

forwarders {10.0.0.1; 10.0.0.2;};

forward only;

};

// zone file sections

....

每个Domain 转发

// zone section fragment of named.conf

zone "example.com" IN {

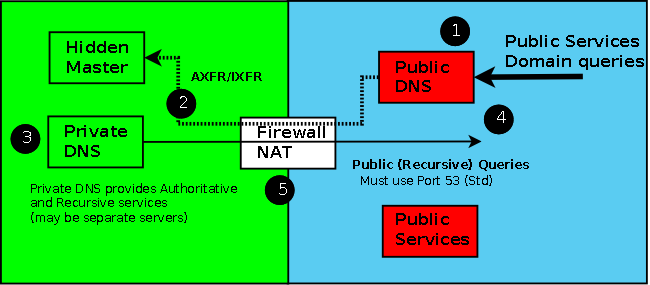
type forward;

forwarders {10.0.0.1; 10.0.0.2;};

};

### 3.5 Stealth Name Server （a.k.a. DMZ or Hidden Master）

隐藏服务器的定义是在domain中一个域名服务器的NS 记录在公有网络中不会出现。



外部网络配置为Authoritative Only 响应和不缓存。针对这些服务器的区域文件(zone file)必须惟一，包括 ONLY 这些系统或者服务在可以公开可见e.g. SOA, 公开的域名服务器的NS记录， MX，www , ftp.记录。zone 传输可以在公有的服务之间传输。但不得转发或者接受来自隐身服务器的传输，

一个公开的zone file

; public zone master file

; provides minimal public visibility of external services

example.com. IN SOA ns.example.com. root.example.com. (

2003080800 ; se = serial number

3h ; ref = refresh

15m ; ret = update retry

3w ; ex = expiry

3h ; min = minimum

)

IN NS ns1.example.com.

IN NS ns2.example.com.

IN MX 10 mail.example.com.

ns1 IN A 192.168.254.1

ns2 IN A 192.168.254.2

mail IN A 192.168.254.3

www IN A 192.168.254.4

ftp IN A 192.168.254.5

内部的或者隐藏的域名服务器。可以提供可见的内部和外部服务器，提供递归查询，或者其它服务，

; private zone master file used by stealth server(s)

; provides public and private services and hosts

example.com. IN SOA ns.example.com. root.example.com. (

2003080800 ; se = serial number

3h ; ref = refresh

15m ; ret = update retry

3w ; ex = expiry

3h ; min = minimum

)

IN NS ns1.example.com.

IN NS ns2.example.com.

IN MX 10 mail.example.com.

; public hosts

ns1 IN A 192.168.254.1

ns2 IN A 192.168.254.2

mail IN A 192.168.254.3

www IN A 192.168.254.4

ftp IN A 192.168.254.5

; private hosts

joe IN A 192.168.254.6

bill IN A 192.168.254.7

fred IN A 192.168.254.8

....

accounting IN A 192.168.254.28

payroll IN A 192.168.254.29

参见文档 PRO DNS and BIND

## Zone file and DNS Resource Record(RRS)

一个Zone file中包含资源记录 （Resource Records (RRs)）和 指令（Directives.）

; zone file for example.com

$TTL 2d ; 172800 secs default TTL for zone

$ORIGIN example.com.

@ IN SOA ns1.example.com. hostmaster.example.com. (

2003080800 ; se = serial number

12h ; ref = refresh

15m ; ret = update retry

3w ; ex = expiry

3h ; min = minimum

)

IN NS ns1.example.com.

IN MX 10 mail.example.net.

joe IN A 192.168.254.3

www IN CNAME joe

这里有几个规范需要注意下

1. $TTL 需要出现在文件的首行 （RFC2308）
2. 第一个RRS必须是SOA（start of Authority）

每个RRS的格式是这样子的

owner-name ttl class type type-specific-data

|  |  |
| --- | --- |
| owner-name | 该记录所属区域文件中的节点的所有者名称（标签）， |
| ttl | 生存时间（秒）指定RR可能被缓存多长时间 |
| class | 用于定义协议族或者协议实例， 正常值是IN＝internel 其他值是HS和CH 都是历史MIT协议 |
| [types](http://www.zytrax.com/books/dns/ch8/#types) | 确定类型特定数据字段的资源记录类型。类型采用下面会介绍 |
| type-specific-data | 每个记录的数据内容由类型和类值定义 |

记录的类型有

A记录：地址记录，用来指定域名的IPv4地址（如：8.8.8.8），如果需要将域名指向一个IP地址，就需要添加A记录。

CNAME： 如果需要将域名指向另一个域名，再由另一个域名提供ip地址，就需要添加CNAME记录。

TXT：在这里可以填写任何东西，长度限制255。绝大多数的TXT记录是用来做SPF记录（反垃圾邮件）。

NS：域名服务器记录，如果需要把子域名交给其他DNS服务商解析，就需要添加NS记录。

AAAA：用来指定主机名（或域名）对应的IPv6地址（例如：ff06:0:0:0:0:0:0:c3）记录。

MX：如果需要设置邮箱，让邮箱能收到邮件，就需要添加MX记录。

显性URL：从一个地址301重定向到另一个地址的时候，就需要添加显性URL记录（注：DNSPod目前只支持301重定向）。

隐性URL：类似于显性URL，区别在于隐性URL不会改变地址栏中的域名。

SRV：记录了哪台计算机提供了哪个服务。格式为：服务的名字、点、协议的类型，例如：\_xmpp-server.\_tcp。

zone file指令以$开头，有三个指令

|  |  |
| --- | --- |
| Directive | Description |
| [$INCLUDE](http://www.zytrax.com/books/dns/ch8/include.html) | Includes the defined file in-line. |
| [$ORIGIN](http://www.zytrax.com/books/dns/ch8/origin.html) | Defines the base name (aka label) to be used for 'unqualified' name substitution. |
| [$TTL](http://www.zytrax.com/books/dns/apa/ttl.html) | Defines the default Resource Record TTL value, used if no TTL is defined in a resource record. |

## BIND 服务器搭建

环境准备

ns1为主DNS域名服务器，ns2为从域名服务器，host1,host2用于测试

| **Host** | **Role** | **Private FQDN** | **Private IP Address** |
| --- | --- | --- | --- |
| host1 | Generic Host 1 | host1.www.gsandow.com | 192.168.56.100 |
| host2 | Generic Host 2 | host2.www.gsandow.com | 192.168.56.101 |
| ns1 | Primary DNS Server | ns1.www.gsandow.com | 192.168.56.111 |
| ns2 | Secondary DNS Server | ns2.www.gsandow.com | 192.168.56.112 |

### 4.1 软件安装及配置

直接使用base源就OK，现在在ns1,ns2上都执行下面命令

yum install bind-utils bind bind-devel bind-chroot

**在ns1,ns2上都修改下面文件**

**编辑 /etc/named.com**

options {

version "1.1.1";

listen-on port 53 {any;};

directory "/var/named/chroot/etc/";

pid-file "/var/named/chroot/var/run/named/named.pid";

allow-query { any; };

Dump-file "/var/named/chroot/var/log/binddump.db";

Statistics-file "/var/named/chroot/var/log/named\_stats";

zone-statistics yes;

memstatistics-file "log/mem\_stats";

empty-zones-enable no;

forwarders {202.106.196.115;8.8.8.8; };

};

key "rndc-key" {

algorithm hmac-md5;

secret "Eqw4hClGExUWeDkKBX/pBg==";

};

controls {

inet 127.0.0.1 port 953

allow { 127.0.0.1; } keys { "rndc-key"; };

};

logging {

channel warning {

file "/var/named/chroot/var/log/dns\_warning" versions 10 size 10m;

severity warning;

print-category yes;

print-severity yes;

print-time yes;

};

channel general\_dns {

file "/var/named/chroot/var/log/dns\_log" versions 10 size 100m;

severity info;

print-category yes;

print-severity yes;

print-time yes;

};

category default {

warning;

};

category queries {

general\_dns;

};

};

include "/var/named/chroot/etc/view.conf";

**编辑/etc/rndc.key， 这一步是没必要的**

key "rndc-key" {

algorithm hmac-md5;

secret "Eqw4hClGExUWeDkKBX/pBg==";

};

**编辑/etc/rndc.conf**

key "rndc-key" {

algorithm hmac-md5;

secret "Eqw4hClGExUWeDkKBX/pBg==";

};

options {

default-key "rndc-key";

default-server 127.0.0.1;

default-port 953;

};

**在ns1上编辑/var/named/chroot/etc/view.conf**

view "View" {

zone "www.gsandow.com" {

type master;

file "www.gsandow.com.zone";

allow-transfer {

192.168.56.112;

};

notify yes;

also-notify {

192.168.56.112;

};

};

zone "168.192.in-addr.arpa" {

type master;

file "192.168.zone";

allow-transfer {

192.168.56.112;

};

notify yes;

also-notify {

192.168.56.112;

};

};

};

**在ns1上编辑www.gsandow.com.zone**

$ORIGIN .

$TTL 3600 ; 1 hour

www.gsandow.com IN SOA ns1.www.gsandow.com. admin.www.gsandow.com. (

2001 ; serial

900 ; refresh (15 minutes)

600 ; retry (10 minutes)

86400 ; expire (1 day)

3600 ; minimum (1 hour)

)

;name servers - NS records

NS ns1.www.gsandow.com.

NS ns2.www.gsandow.com.

$ORIGIN www.gsandow.com.

ns1 A 192.168.56.111

ns2 A 192.168.56.112

host1 A 192.168.56.100

host2 A 192.168.56.101

**在NS1上编辑192.168.zone**

$TTL 604800

@ IN SOA ns1.www.gsandow.com. admin.www.gsandow.com. (

3

60480

86400

2419200

604800)

; name server

NS ns1.www.gsandow.com.

NS ns2.www.gsandow.com.

; PTR recodes

111.56 IN PTR ns1.www.gsandow.com.

112.56 IN PTR ns2.www.gsandow.com.

100.56 IN PTR host1.www.gsandow.com.

101.56 IN PTR host2.www.gsandow.com.

**在ns2上编辑/var/named/chroot/etc/view.conf**

view "SlaveView" {

zone "www.gsandow.com" {

type slave;

masters {192.168.56.111; };

file "ns2.www.gsandow.com.zone";

};

zone "168.192.in-addr.arpa" {

type slave;

masters {192.168.56.111; };

file "ns2.192.168.zone";

};

};

**在ns1,ns2上同时操作**

cd /var && chown -R named.named named/

/etc/init.d/named start

chkconfig named on

如果需要重要加载配置可以使用下面命令

rndc reload

这时候主从DNS就已经搭建完结。

### 4.2 增加记录

所有的记录都是编辑文件/var/named/chroot/etc/www.gsandow.com.zone，在最后增加

所有的PTR记录都是编辑/var/named/chroot/etc/192.168.zone， 在最后增加

**增加A记录**

编辑master节点在文件末尾添加一条记录

a A 192.168.56.101

将serial值+1

执行rndc reload命令

分别对master、slave主机下执行：host a.www.gsandow.com 127.0.0.1 结果应该是192.168.56.101

**CNAME记录**

cname CNAME a.www.gsandow.com.

将serial值+1

执行rndc reload命令

分别对master、slave解析下：host cname.www.gsandow.com 127.0.0.1结果应该是192.168.56.101

**MX记录**

mx MX 5 192.168.122.101

mx MX 10 192.168.123.101

将serial值+1

执行rndc reload命令

分别对master、slave解析下：host mx.lnh.com 127.0.0.1 结果应该是192.168.122.101

### 4.3 故障案例

**4.3.1. 运维同学突然告诉我，你的DNS不能用了**

记得那时我刚来公司还没转正的时候。有个哥们问我，dns现在没什么事吧？卧槽，dns？什么dns？

记得那会线上跑着多少dns，每台dns配置一不一样，没人说得清楚。于是我说兄弟，你什么dns，怎么了？他告诉我一个ip，说用的这个dns，现在解析不了了。我说我看下，你先改个别的dns，这种时候一定要先恢复业务，然后再排查、解决问题。

我尝试用那台dns解析域名，确实不行了，登上去看，dns进程还在，想来是有什么瓶颈了，看日志吧，在日志中发现有报netfilter满了的日志，怎么tm会报这个呢，难道机器起着iptables？卧槽，还真是起着呢，这不蛋逼呢么。赶紧把iptables停掉，问题恢复了。

在iptable运行时，即便是没有任何规则，所有的报文也要过netfilter表，而dns这种高并发的业务很容易把netfilter表填满，从而影响服务。

后续的工作就是整理线上dns，同一配置，做好监控，做好高可用和负载均衡

**4.3.2 突然收到报警，提示DNS虚地址不能正常解析了**

那天带孩子体检，请了一天假，还在医院的时候，收到的报警，赶紧电话到公司，询问怎么回事，原来是有同事将zabbix的一个域名删除了，而这个域名恰恰是我lvs健康检测dns服务是否可以提供解析的域名，域名都没了，那肯定解析失败，于是lvs把所有dns都剔除了。

虽然恢复起来很快，但是这也提醒我，系统之间，一定要尽量减小藕合性，避免雪崩。

**4.3.3 开发来找，说调用微信支付接口有超时的现象，错误日志中报解析微信域名失败**

接到这个case，首先是确认问题是否出现在域名解析的环节，我登录到服务器上，用nslookup解析对应的域名，连续解析1000次均没问题，但开发小妹依然喋喋不休，我按耐住tmd心情。突然想起也许她php中调用的解析函数与nslookup不同，于是我用host解析，这时发现问题了。每次解析微信支付这个域名的时候，都要等10s左右才出结果。这时，我又找其他机器做测试，发现只有这个微信支付的域名有这种现象，这不由得让我对大qq产生了膜拜的情绪。

出现这种情况让人很头大，而旁边开发小妹还在殷切的看着你，你背后还有领导不时的喘着粗气，而恰恰此时你对这问题一点头绪也没有，怎么办？

越是这种时候，越要淡定，不然怎么办？！这玩意不是谁教的，是你多遇到几次这种情况，每次都要提醒自己，冷静、冷静、tmd冷静

既然只有解析微信这个接口有问题，那我想可能是微信对dns这块有什么限制，那我可不可以换个forward的公网dns试下呢?分两头，一头是我去测试，另外一头让同事联系微信，给提供一个qq那边的权威dnsip，最终我将qq权威dns的ip加到我dns的forward列表中解决了这个case。

## DNS 高级技术

### 5.1 配置DNS视图（智能DNS）

编辑master节点vim /var/named/chroot/etc/named.conf，在include上面添加

acl group1 {

10.5.35.14;

};

acl group2 {

10.5.35.15;

};

编辑master节点vim /var/named/chroot/etc/view.conf为

view "GROUP1" {

match-clients { group1; };

zone "viewlnh.com" {

type master;

file "group1.viewlnh.com.zone";

};

};

view "GROUP2" {

match-clients { group2; };

zone "viewlnh.com" {

type master;

file "group2.viewlnh.com.zone";

};

};

编辑master节点vim /var/named/chroot/etc/group1.viewlnh.com.zone

$ORIGIN .

$TTL 3600 ; 1 hour

viewlnh.com IN SOA op.viewlnh.com. dns.viewlnh.com. (

2005 ; serial

900 ; refresh (15 minutes)

600 ; retry (10 minutes)

86400 ; expire (1 day)

3600 ; minimum (1 hour)

)

NS op.viewlnh.com.

$ORIGIN viewlnh.com.

op A 192.168.122.1

view A 192.168.122.1

编辑master节点vim /var/named/chroot/etc/group2.viewlnh.com.zone

$ORIGIN .

$TTL 3600 ; 1 hour

viewlnh.com IN SOA op.viewlnh.com. dns.viewlnh.com. (

2005 ; serial

900 ; refresh (15 minutes)

600 ; retry (10 minutes)

86400 ; expire (1 day)

3600 ; minimum (1 hour)

)

NS op.viewlnh.com.

$ORIGIN viewlnh.com.

op A 192.168.122.2

view A 192.168.122.2

修改文件所属，加载配置

chown named.named /var/named/chroot/etc/group\*.zone

rndc reload

分别在主机A与B上做view.viewlnh.com的域名解析测试

结果应为主机A上是192.168.122.1主机B上是192.168.122.2

## DNS 诊断工具

bind提供了一些实用的工具来测试配置文件，和诊断问题，比如 named-checkconf 用来检查配置语法是否正确，rndc 用来重新加载zone（但如果没有正确配置，可能会有安全隐患）， nsupdate可以允许动态更新dns（需要注意的是，如果配置不发，可能把所有集群暴露到公网内），这里只介绍三个非常实用的工具 dig, nslookup, host,在工作会经常用到。使用下面命令来安装工具包。

yum install bind-utils

### 6.1 host

获取A或者PTR记录

[root@ns1:~]#host 192.168.56.112

112.56.168.192.in-addr.arpa domain name pointer ns2.www.gsandow.com.

### 6.2 nslookup

nslookup通过返回A或者PTR记录，但是也可以使用特别的参数来覆盖默认值。他有交互和命令行两种方式

查看A记录，如果需要使用特定的dns服务器可以 在域名后面加dns服务器IP，可以指定type

A：查看主机的IPv4地址

AAAA：查看主机的IPv6地址

ANY：查看关于主机域的所有信息

CNAME：查找与别名对应的正式名字

HINFO：查找主机的CPU与操作系统类型

MINFO：查找邮箱信息

MX：查找邮件交换信息

NS：查找主机域的域名服务器

PTR：查找与给定IP地址匹配的主机名

RP：查找域负责人记录

SOA：查找域内的SOA地址

UINFO：查找用户信息

[root@ns1:~]#nslookup host1.www.gsandow.com

Server: 192.168.56.111

Address: 192.168.56.111#53

Name: host1.www.gsandow.com

Address: 192.168.56.100

查看MX NS 记录

[root@ns1:~]#nslookup -type=ANY pdmi.cn

Server: 192.168.56.111

Address: 192.168.56.111#53

Non-authoritative answer:

pdmi.cn nameserver = ns2.dns.com.cn.

pdmi.cn nameserver = ns1.dns.com.cn.

pdmi.cn mail exchanger = 10 mxbiz2.qq.com.

pdmi.cn mail exchanger = 5 mxbiz1.qq.com.

pdmi.cn text = "v=spf1" "include:spf.mail.qq.com" "~all"

pdmi.cn

origin = dns.bizcn.com

mail addr = abuse.bizcn.com

serial = 79

refresh = 28800

retry = 14400

expire = 14400

minimum = 600

Authoritative answers can be found from:

pdmi.cn nameserver = ns1.dns.com.cn.

pdmi.cn nameserver = ns2.dns.com.cn.

获取 SOA记录

[root@ns1:~]#nslookup -all -type=SOA pdmi.cn

Set options:

novc nodebug nod2

search recurse

timeout = 0 retry = 3 port = 53

querytype = A class = IN

srchlist =

Server: 192.168.56.111

Address: 192.168.56.111#53

Non-authoritative answer:

pdmi.cn

origin = dns.bizcn.com

mail addr = abuse.bizcn.com

serial = 79

refresh = 28800

retry = 14400

expire = 14400

minimum = 600

Authoritative answers can be found from:

pdmi.cn nameserver = ns1.dns.com.cn.

pdmi.cn nameserver = ns2.dns.com.cn.

### 6.3 dig 命令

dig 全称为 Domain information Groper, dig 的命令格式为

dig @dnsserver name querytype

如果你设置的dnsserver是一个域名，那么dig会首先通过默认的上连DNS服务器去查询对应的IP地址，然后再以设置的dnsserver为上连DNS服务器。如果你没有设置@dnsserver，那么dig就会依次使用/etc/resolv.conf里的地址作为上连DNS服务器。

而对于querytype，上一节nslookup命令的讲解，你可以设置A/AAAA/PTR/MX/ANY等值，默认是查询A记录。

一些选项：

1 -c选项，可以设置协议类型（class），包括IN(默认)、CH和HS。

2 -f选项，dig支持从一个文件里读取内容进行批量查询，这个非常体贴和方便。文件的内容要求一行为一个查询请求。来个实际例子吧

3 -4和-6两个选项，用于设置仅适用哪一种作为查询包传输协议，分别对应着IPv4和IPv6。

4 -t选项，用来设置查询类型，默认情况下是A，也可以设置MX等类型，

5 -x选项，是逆向查询选项。可以查询IP地址到域名的映射关系。

dig总共有42个查询选项，涉及到DNS信息的方方面面，下面举几个

1 dig非常著名的一个查询选项就是+trace，当使用这个查询选项后，dig会从根域查询一直跟踪直到查询到最终结果，并将整个过程信息输出出来。

2 使用+nocmd的话，可以节省输出dig版本信息。

3 使用+short的话，仅会输出最精简的CNAME信息和A记录，其他都不会输出。就像这样：

$ cat querylist

www.baidu.com

www.sohu.com

$ dig -f querylist -c IN -t A

**[root@ns1:~]#dig pdmi.cn**

;; QUESTION SECTION:

;pdmi.cn. IN A

;; AUTHORITY SECTION:

pdmi.cn. 60 IN SOA dns.bizcn.com. abuse.bizcn.com. 79 28800 14400 14400 600

[root@ns1:~]#dig pdmi.cn mx

;; QUESTION SECTION:

;pdmi.cn. IN MX

;; ANSWER SECTION:

pdmi.cn. 2388 IN MX 5 mxbiz1.qq.com.

pdmi.cn. 2388 IN MX 10 mxbiz2.qq.com.

获取所有记录

dig pdmi.cn axfr

## DNS 安全