**3\_Services04分离解析RAID进程管理日志system运行级别**

案例:

虚拟机:A构建DNS服务器,实现www.sina.com的解析结果为10.20.30.40

虚拟机A:

**1 修改主配置文件:**

vim /etc/named.conf

options {

directory "/var/named";

};

zone "tedu.cn" IN {

type master;

file "tedu.cn.zone";

};

zone "qq.com" IN {

type master;

file "qq.com.zone";

};

**zone "sina.com" IN {**

**type master;**

**file "sina.com.zone";**

**};**

**2 创建地址库文件:**

cd /var/named

cp **-p** named.localhost sina.com.zone

vim sina.com.zone

$TTL 1D

@ IN SOA @ rname.invalid. (

0 ; serial

1D ; refresh

1H ; retry

1W ; expire

3H ) ; minimum

**sina.com. NS svr7**

**svr7 A 192.168.4.7**

**www A 10.20.30.40**

systemctl restart named #重启服务

**3 验证**

虚拟机B:

[root@pc207 ~]# nslookup www.sina.com

Server: 192.168.4.7

Address: 192.168.4.7#53

Name: www.sina.com

Address: 10.20.30.40

**一 Split 分离解析**

**1.1定义:**当收到客户机的DNS查询请求的时候,能够区分客户机的来源地址,为不同类别的客户机提供不同的解析结果(IP地址)

**不同的客户端解析同一个域名,解析结果不同**

**1.2 典型使用场景:**

访问压力大的网站,购买CDN提供的内容分发服务[CND:content delivery network 内容分发网络]

在全国各地/不同网络内部署大量镜像服务节点

针对不同的客户机就近提供服务器

**1.3 BIND的view视图**

根据源地址集合将客户机分类,不同客户机获得不同结果(待遇有差别)

view “nsd” {

match-clients {来源地址192.168.1.1;... ..;}; #匹配客户机来源地址

zone “12306.cn” IN {

......地址库文件12306.zone;

};};

view “abc” {

match-clients {来源地址192.168.2.1;... ..;}; #匹配客户机来源地址

zone “12306.cn” IN {

......地址库文件12306.abc;

};};

view “other” {

match-clients {来源地址any;... ..;};

#匹配客户机来源地址

zone “12306.cn” IN {

......地址库文件12306.other;

};};

**匹配原则:由上到下,匹配即停止,所有的客户端都要找到自己的分类,所有的zone都要在view中.**

**1.4案例:**

环境及需求

权威DNS:svr7.tedu.cn 192.1168.4.7

负责区域:tedu.cn

A记录分离解析----以www为例

客户机来自 解析结果

192.168.4.207 192.168.4.100

其他地址 1.2.3.4

修改主配置文件

cd /etc/

vim /etc/named.conf

options {

directory "/var/named";

};

**view "nsd" { #分类名称**

**match-clients { 192.168.4.207; }; #匹配客户机来源地址**

**zone "tedu.cn" IN {**

**type master;**

**file "tedu.cn.zone"; #解析结果写192.168.4.100**

**};**

**};**

**view "other" {**

**match-clients { any; };**

**zone "tedu.cn" IN {**

**type master;**

**file "tedu.cn.other"; #解析结果写1.2.3.4**

**};**

**};**

建立地址库文件tedu.cn.zone与tedu.cn.other,写入不同的解析结果

cd /etc/named

vim /var/named.tedu.zone

$TTL 1D

@ IN SOA @ rname.invalid. (

0 ; serial

1D ; refresh

1H ; retry

1W ; expire

3H ) ; minimum

tedu.cn. NS svr7.tedu.cn.

svr7.tedu.cn. A 192.168.4.7

**www.tedu.cn. A 192.168.4.100**

ftp.tedu.cn. A 2.2.2.2

\* A 1.2.3.4

mail.tedu.cn. CNAME ftp.tedu.cn.

$GENERATE 1-50 pc$.tedu.cn. A 192.168.10.$

增加other的地址库文件并修改

cp -p tedu.cn.zone tedu.cn.other

vim tedu.cn.other

$TTL 1D

@ IN SOA @ rname.invalid. (

0 ; serial

1D ; refresh

1H ; retry

1W ; expire

3H ) ; minimum

tedu.cn. NS svr7.tedu.cn.

svr7.tedu.cn. A 192.168.4.7

**www.tedu.cn. A 1.2.3.4**

ftp.tedu.cn. A 2.2.2.2

\* A 2.3.4.5

mail.tedu.cn. CNAME ftp.tedu.cn.

$GENERATE 1-50 pc$.tedu.cn. A 192.168.10.$

Systemctl restart named #重启服务

验证:

虚拟机B

[root@pc207 ~]# nslookup www.tedu.cn 192.168.4.7

Server: 192.168.4.7

Address: 192.168.4.7#53

Name: www.tedu.cn

Address: 192.168.4.100

真机:

[student@room9pc01 ~]$ nslookup www.tedu.cn 192.168.4.7

Server: 192.168.4.7

Address: 192.168.4.7#53

Name: www.tedu.cn

Address: 1.2.3.4

**1.5 多个域名的分离解析:每个view中zone个数保持一致**

A记录分离解析--以www.tedu.cn为例

A记录分离解析--以www.sina.com为例

客户机来自 解析结果

192.168.4.207 192.168.4.100

其他地址 1.2.3.4

1.5.1修改主配置文件,增加sina.com的zone

vim /etc/named.conf

options {

directory "/var/named";

};

view "nsd" {

match-clients { 192.168.4.207; };

zone "tedu.cn" IN {

type master;

file "tedu.cn.zone";

};

**zone "sina.com" IN {**

**type master;**

**file "sina.com.zone";**

**};**

};

view "other" {

match-clients { any; };

zone "tedu.cn" IN {

type master;

file "tedu.cn.other";

};

**zone "sina.com" IN {**

**type master;**

**file "sina.com.other";**

**};**

};

1.5.2 增加sina.com的other的地址库文件并修改

cd /var/named

cp -p tedu.cn.other sina.com.other

vim sina.com.other

$TTL 1D

@ IN SOA @ rname.invalid. (

0 ; serial

1D ; refresh

1H ; retry

1W ; expire

3H ) ; minimum

**sina.com. NS svr7**

svr7 A 192.168.4.7

**www A 1.2.3.4**

ftp A 2.2.2.2

\* A 2.3.4.5

Mail CNAME ftp.tedu.cn.

$GENERATE 1-50 pc$.tedu.cn. A 192.168.10.$

systemctl restart named #重启服务

1.5.3 验证

虚拟机B

[root@pc207 ~]# nslookup www.sina.com 192.168.4.7

Server: 192.168.4.7

Address: 192.168.4.7#53

Name: www.sina.com

Address: 192.168.4.100

真机

[student@room9pc01 ~]$ nslookup www.sina.com 192.168.4.7

Server: 192.168.4.7

Address: 192.168.4.7#53

Name: www.sina.com

Address: 1.2.3.4

options {

directory "/var/named";

};

view "nsd" {

match-clients { 192.168.4.207; };

zone "tedu.cn" IN {

type master;

file "tedu.cn.zone";

};

zone "sina.com" IN {

type master;

file "sina.com.other";

};

};

view "abc" {

match-clients { 192.168.4.7; };

zone "tedu.cn" IN {

type master;

file "tedu.cn.other";

};

zone "sina.com" IN {

type master;

file "sina.com.zone";

};

};

view "other" {

match-clients { any; };

zone "tedu.cn" IN {

type master;

file "tedu.cn.other";

};

zone "sina.com" IN {

type master;

file "sina.com.other";

};

};

**1.6 ACL地址列表**

acl地址列表,类似于变量作用

acl “test” { 192.168.4.207;192.168.1.1;192.168.2.1;192.168.3.1;192.168.7.0/24 }

**二 RAID磁盘阵列**

**2.1 RAID:redundant arrays of inexpensive disks**

通过硬件/软件技术,将多个较小/低速的磁盘整合成一个大磁盘

阵列的价值:提升I/O效率 硬件级别的数据冗余

不同的RAID级别的功能 特性各不相同

**2.2 RAID0:条带模式**

同一个文档分散存放在不同的磁盘,并行写入以提高效率,至少需要两块磁盘组 成;无容错功能.

**2.3 RAID1:镜像模式**

一份文档复制成多份,分别写入不同磁盘,多份拷贝提高了可靠性,至少需要两块 磁盘; 效率无提升.

**2.4 RAID5:高性价比模式**

相当于RAID0和RAID1的折中方案,需要至少一块磁盘的容量来存放校验数据,至少 需要三块磁盘.

**2.5RAID6:高性价比模式/可靠模式**

相当于扩展的RAID5阵列,提供2份独立校验方案.需要至少两块磁盘的容量来存 放校验数据;至少需要四块磁盘组成.

**2.6 RAID0+1 RAID1+0**

整合RAID0 RAID1的优势

RAID0+1:两块磁盘先组成RAID0,再两组RAID0组成RAID1.

RAID1+0:两块磁盘先组成RAID1,再两组RAID1组成RAID0.

**2.7 RAID各级别特点对比**

对比项 RAID0 RAID1 RAID10 RAID5 RAID6

磁盘数 >=2 >=2 >=4 >=3 >=4

校验盘 无 无 无 1 2

容错性 无 有 有 有 有

IO性能 高 低 中 较高 较高

**三 进程管理**

**3.1 查看进程树**

程序:静态的代码,占用磁盘空间

进程:动态执行的代码,占用CPU 内存资源

一个程序会产生底多个进程

父进程/子进程 示例:微信/输入法 树型结构

开启时先启动父进程,再由父进程启动子进程

关闭时先关闭子进程,再关闭父进程

杀死父进程时,会联动杀死对应子进程

其他进程:僵尸进程 孤儿进程

进程唯一标识:PID 进程的编号,编号越小,代表进程越重要,优先运行.

3.1.1命令:pstree Processes Tree

格式 pstree [选项] [PID或用户名]

-a :显示完整的命令行

-p:列出对应的PID编号

Systemd:上帝进程,所有进程的父进程

[root@svr7 ~]# useradd lisi

[lisi@svr7 ~]$ vim 1.txt

[root@svr7 ~]# pstree lisi

bash───vim

[root@svr7 ~]# pstree -a lisi

bash

└─vim 1.txt

[root@svr7 ~]# pstree -ap lisi

bash,904

└─vim,929 1.txt

3.1.2 命令:ps Processes Snapshot

格式 ps [选项]

ps aux 操作:列出正在运行的所有进程

USER PID %CPU %MEM VSZ RSS TTY STAT TART TIME COMMAND

用户 PID

命令: ps -elf 操作:列出正在运行的所有进程,PPID表示该进程的父进程ID

F S UID PID PPID C PRI NI ADDR SZ WCHAN STIME TTY TIME CMD

案例:统计正在运行的进程有多少?

ps aux | wc -l 或 ps -elf | wc -l

3.1.3 命令:top 交互式工具**[动态查看命令]**

格式 :top [-d 刷新秒数] [-u 用户名]

按P 以CUP占用率排序

按M 以内存占用率排序

load average: 0.00, 0.01, 0.04 #CPU负载量 1,5,15分钟内的平均负载量

3.1.4 命令:pgrep process grep

格式 pgrep [选项]...查询条件

-l:输出进程名,而不仅仅是PID

-U:检索制定用户的进程, [U必须离用户名最近]

-x:精确匹配完整的进程名

[root@svr7 ~]# pgrep -l a #检索所有名称带a的进程

[root@svr7 ~]# pgrep -lU lisi #检索用户lisi的所有进程

904 bash

[root@svr7 ~]# pstree -ap lisi #检索用户lisi的所有进程

bash,904

**3.2 进程的前后台调度**

3.2.1 后台启动

在命令行末尾加 & :将进程放到后台运行

ctrl+z 组合键:暂停正在运行的进程

jobs:查看后台运行的进程

bj 后台进程编号:继续运行暂停的进程

fg 后台进程编号:将后台运行的进程放到前台

[root@svr7 ~]# sleep 9000 & #正在运行的进程放入后台

[1] 1040

[root@svr7 ~]# jobs #查看后台运行的进程

[1]+ 运行中 sleep 9000 &

[root@svr7 ~]# sleep 8000

^Z #输入ctrl+z,暂停进程并放入后台

[2]+ 已停止 sleep 8000

[root@svr7 ~]# jobs

[1]- 运行中 sleep 9000 &

[2]+ 已停止 sleep 8000

[root@svr7 ~]# bg 2 #将后台编号为2的进程继续运行

[2]+ sleep 8000 &

[root@svr7 ~]# jobs

[1]- 运行中 sleep 9000 &

[2]+ 运行中 sleep 8000 &

[root@svr7 ~]# fg 1 #将后台编号为1的进程恢复到前台

sleep 9000

^C #输入ctrl+c 结束

[root@svr7 ~]# jobs

[2]+ 运行中 sleep 8000 &

[root@svr7 ~]# fg 2

sleep 8000

^C #输入ctrl+c 结束

**3.3 进程的查杀**

ctrl+c 组合键,中断当前命令程序

kill [-9] PID... 或 kill [-9] %后台任务编号 #[-9]表示强制,能不加就不加,可对 僵尸进程使用

killall [-9] 进程名...

pkill 查找条件

[root@svr7 ~]# sleep 1000 &

[1] 1061

[root@svr7 ~]# jobs

[1]+ 运行中 sleep 1000 &

[root@svr7 ~]# jobs -l

[1]+ 1061 运行中 sleep 1000 &

[root@svr7 ~]# kill 1061

[1]+ 已终止 sleep 1000

[root@svr7 ~]# jobs

[root@svr7 ~]# sleep 1000 &

[1] 1063

[root@svr7 ~]# jobs -l

[1]+ 1063 运行中 sleep 1000 &

[root@svr7 ~]# kill -9 1063

[1]+ 已杀死 sleep 1000

[root@svr7 ~]# sleep 1000 &

[1] 1064

[root@svr7 ~]# sleep 1000 &

[2] 1065

[root@svr7 ~]# sleep 1000 &

[3] 1066

[root@svr7 ~]# jobs

[1] 运行中 sleep 1000 &

[2]- 运行中 sleep 1000 &

[3]+ 运行中 sleep 1000 &

[root@svr7 ~]# killall sleep

[1] 已终止 sleep 1000

[2]- 已终止 sleep 1000

[3]+ 已终止 sleep 1000

[root@svr7 ~]# sleep 1000 &

[1] 1069

[root@svr7 ~]# sleep 1000 &

[2] 1070

[root@svr7 ~]# sleep 1000 &

[3] 1071

[root@svr7 ~]# jobs

[1] 运行中 sleep 1000 &

[2]- 运行中 sleep 1000 &

[3]+ 运行中 sleep 1000 &

[root@svr7 ~]# pkill sleep

[1] 已终止 sleep 1000

[2]- 已终止 sleep 1000

[3]+ 已终止 sleep 1000

杀死一个用户开启的所有进程(强制踢出一个用户) [只有root有权限执行命令]

killall -u 用户名

[lisi@svr7 ~]$ vim 1.txt

[root@svr7 ~]# killall -u lisi

Vim: Caught deadly signal TERM

Vim: Finished.

Terminated

[lisi@svr7 ~]$

killall -9 -u 用户名

[lisi@svr7 ~]$ vim 1.txt

[root@svr7 ~]# killall -9 -u lisi

已杀死

~ [root@svr7 ~]#

**四 日志管理**

**4.1 日志的功能**

记录系统\程序运行中发生的各种事件

通过查看日志,了解及排除故障

信息安全控制的”依据”

**4.2 内核及系统日志**

由系统服务rsyslog统一记录/管理

日志消息采用文本格式

主要记录事件发生的时间\主机\进程\内容

常见的日志文件

日志文件 主要用途

/var/log/messages 记录内核消息\各种服务的公共消息

/var/log/dmesg 记录系统启动过程中的各种消息

/var/log/cron 记录与cron计划任务相关的消息

/var/log/maillog 记录邮件收发相关的消息

/var/log/secure 记录与访问限制相关的安全消息

**4.3 用户日志**

由登录程序负责记录\管理

日志消息采用二进制格式

记录登录用户的事件\来源\执行的命令等信息

**4.4 日志分析**

4.4.1 查看文本日志消息

通用分析工具

tail tailf less grep等文本浏览/检索命令

tailf:实时跟踪[新开一个终端, tail /路径/文件名]

awk sed等格式化过滤工具

专用分析工具

Webmin 系统管理套件

Webalizer\AWStats等日志统计套件

4.4.2 用户登录分析

users who w 命令

查看已登录的用户信息,详细度不同w最详细,who最常用

[lisi@svr7 ~]$ who

root pts/0 2019-07-18 14:57 (192.168.4.254)

root pts/1 2019-07-18 16:58 (192.168.4.254)

pts:图形命行令终端 /0 /1 /2 表示第1 2 3 个图形命令行终端

last lastb 命令: 查看最近登录成功/失败的用户信息

[lisi@svr7 ~]$ last -n 2

root pts/1 192.168.4.254 Thu Jul 18 16:58 still logged in

root pts/2 192.168.4.254 Thu Jul 18 15:29 - 15:29 (00:00)

[student@room9pc01 ~]$ ssh -X haha@192.168.4.7

[student@room9pc01 ~]$ ssh -X [hehe@192.168.4.7](mailto:hehe@192.168.4.7)

[root@svr7 ~]# lastb -n 2

hehe ssh:notty 192.168.4.254 Thu Jul 18 17:07 - 17:07 (00:00)

hehe ssh:notty 192.168.4.254 Thu Jul 18 17:07 - 17:07 (00:00)

4.4.3 日志消息的优先级

Linux内核定义的事件紧急程度

分为0-7 共8种优先级,数值越小,表示对应事件越紧急/重要

0 EMERG(紧急) 会导致主机系统不可用的情况

1 ALERT(警告) 必须马上采取措施解决的问题

2 CRIT(严重)

3

4

5

6

7

4.4.4 使用journalctl工具

提取由systemd-journal服务收集的日志

主要包括内核/系统日志\服务日志

常见用法

journalctl | grep 关键词

journalctl -xe

**journalctl -u 服务名 [-p 优先级]**

journalctl -n 消息条数

journalctl --since=”yyyy-mm-dd HH:MMSS” --until=”yyyy-mm-dd HH:MMSS”

**五 systemd介绍**

由内核引导后加载的第一个进程(PID=1)

负责掌控整个Linux的运行/服务资源组合

一个更高效的系统&服务管理器

开机服务并行启动,各系统服务间的精确因爱

配置目录:/etc/systemd/system/

服务目录:/lib/systemd/system/

主要管理工具:systemctl

对于服务的管理

**systemctl restart** 服务名

systemctl start 服务名

systemctl stop 服务名

systemctl status 服务名

**systemctl enable** 服务名

systemctl disable 服务名

systemctl is-enabled 服务名 #查看是否为开机自启

**六 RHEL6 运行级别**

不同级别,开启的服务不同

0:关机

1:单用模式(基本功能的实现,破解Linux密码)

2:多用户字符界面(不支持网络)

3:多用户字符界面(支持网络)**服务器默认的运行级别**

4:未定义

5:图形界面

6:重启

切换运行级别:init 级别代码

**七 RHEL7 运行模式**

字符模式:multi-user.target

图形模式:graphical.target

直接切换到字符模式 systemctl isolate multi-user.target = init 3

直接切换到图形模式 systemctl isolate graphical.target = init 5

查看默认级别 systemctl get-default

设置永久策略,每次开机自动进入该模式

systemctl set-defalut multi-user.target

systemctl set-default graphical.target