Linux 常用企业服务汇总 v1.0

Apprentice 制作
2014.4.6

Update:

1.0 Linux 学习企业服务总结,实验本人均以实测,但不排除存在 Bug,本手册会继续更新,记录点点滴滴.欢迎移步指教一二,不甚感激.
新浪微博 @丶 Apprentice
51cto: http://apprentice.blog.51cto.com/

I. 实验: DHCP + tftp + xinetd + apache 完成 PXE 网络引导安装

服务名称	安装名称版本	服务端口信息	服务描述
DHCP	dhcp-4.1.1 dhcp-common	Server udp 67 Client udp 68	为客户端提供自动 ip 分配服务
tftp	Tftp-0.49 Tftp-Server-0.49	Udp 69	小型 ftp 服务器 无安全机制
Xinetd	Xinted-2.3.14		超级守护进程为不常用服务提供托管调度
Syslinux	Syslinux-4.02-8		为 pxe 方式安装 linux 提供引导或安全服务
Apache	Apache-2.4.9	Tcp 80	为 pxe 方式安装 linux 提供内核引导文件下载途径

配置文件:

dhcp: /etc/dhcpd/dhcpd.conf

实验思路:

自动安装

1,制作 iso 镜像

i) 根据当前模板机的 anaconda-ks.cfg 下载所有 rpm 包

ii) 下载引导内核与 initfs

iii) 制作光盘镜像文件

2,基于网络 pxe

i) 基于 dhcp 服务使客户端完成自动 ip 获取

ii) 客户端拿到 ip 后被指向系统安装引导内核

iii) 引导内核中包含了网络 ks 文件

实验步骤:

1. 光盘制作过程

a) 以当前最微系统的安装过程为蓝本 cp ~/anaconda-ks.cfg ~/ks.cfg
b) 修正 ks 参数 , 以及安装需要的所有工具
#详细参见 Red_Hat_Enterprise_Linux-5-Installation_Guide-en-US.pdf

```
# Kickstart file automatically generated by anaconda.
#version=DEVEL
text
install
cdrom
lang en_US.UTF-8
keyboard us
network --onboot yes --device eth0 --bootproto dhcp --noipv6
rootpw --
iscrypted$6$pcYzFDFOq/kCTHmk$r/4JaQm8qav2X1OfAh5S/uTH.V0JIxFZoef5F
hU10Ox53TBsH1IXeelSc5xIM/0gfWYD7OWDdGjttFOlBvy400
firewall --service=ssh
```

```
        authconfig --enableshadow --passalgo=sha512
        selinux --enforcing
        timezone --utc Asia/Shanghai
bootloader --location=mbr --driveorder=sda --append="crashkernel=auto rhgb
quiet"
        zerombr
        clearpart --all

        volgroup VolGroup --pesize=4096 pv.008002
logvol / --fstype=ext4 --name=lv_root --vgname=VolGroup --grow --size=1024 --
maxsize=51200
logvol swap --name=lv_swap --vgname=VolGroup --grow --size=992 --
maxsize=992

        part /boot --fstype=ext4 --size=500
        part pv.008002 --grow --size=1

        repo --name="CentOS"  --baseurl=cdrom:sr0 --cost=100

        %packages --nobase
        @core
        @Core
        %end
```

c) 制作光盘 .sh 脚本

```
        #!/bin/bash
        # trap "rm -rf /root/iso/Packages/*; exit 0;" INT

        [ -e "/root/myiso/Packages" ] || mkdir -p /root/myiso/Packages
        # 下载 install.log 文件中每一个安装包
        cd /root/myiso/Packages
        for i in `grep "^Installing" /root/install.log | awk '{print $2}'`; do
                if [ ! -e $centospath$i.rpm ]; then
                        wget -qnc $centospath$i.rpm -P $path
                else
                        wget -qnc $eplepath$i.rpm -P $path
                fi
        done

        cd /root
        createrepo /root/myiso/Packages
        #mv /root/myiso/Packages/repodata/ /root/myiso/
        # 准备安装过程中需要的 boot 与 fs 文件
        [ -e "/root/myiso/isolinux" ] || mkdir -p /root/myiso/isolinux
        cd /root/myiso/isolinux
        echo "mirror" | lftp "http://172.16.0.1/cobbler/ks_mirror/centos-6.5-
x86_64/isolinux"

        cp /root/local.cfg /root/myiso/local.cfg
```

```
        sed -i "s/\(^[[:space:]]\{2\}append.*img$\)/\\1 ks=cdrom:\/local.cfg/"
/root/myiso/isolinux/isolinux.cfg

        # 制作镜像
        cd /root/myiso
        mkisofs -R -J -T -v --no-emul-boot --boot-load-size 4 --boot-info-table -V
"CentOS 6.5 x86_64 boot" -b isolinux/isolinux.bin -c isolinux/boot.cat -o
/root/centos6.5.iso  ~/myiso
```

2. 网络设备 PXE

```
基于 DHCP 与 TFTP + Web 服务器
a) 建立 dhcp 引导网络中的设备   etc/dhcp/dhcpd.conf
        subnet 172.16.43.0 netmark 255.255.255.0 {
                range 172.16.43.1 172.16.43.50;
        }

b) 在动态为客户端分配 IP 后  指向 nextserver 为 tftp isolinux 所在路径
        /var/lib/tftpboot/下的 pxelinux.0
        subnet 172.16.43.0 netmark 255.255.255.0 {
                range 172.16.43.1 172.16.43.50;
                next-server 10.0.0.100
                filename "pxelinux.0"
        }

c) 安装启动 xinetd tftp-server tftp 并 chkconfig 服务

d) 安装 httpd 将安装树绑定到 httpd 的目录下
        先挂载在绑定  mount --bind /media /var/www/html/centos6

e) 安装 syslinux

f) 拷贝如下文件
        # 需要准备文件的说明
        # boot.msg  init 文件初始化文件系统  dhcp 引导安装文件
        # 引导配置文件  背景图      vesamenu.c32    内核
        cp /media/cdrom/images/pxeboot/{vmlinuz,initrd.img}
/var/lib/tftpboot/
        cp /media/cdrom/isolinux/{boot.msg,vesamenu.c32,splash.jpg}
/var/lib/tftpboot/
        cp /usr/share/syslinux/pxelinux.0  /var/lib/tftpboot/
        mkdir /var/lib/tftpboot/pxelinux.cfg
        cp /media/cdrom/isolinux/isolinux.cfg
/var/lib/tftpboot/pxelinux.cfg/default

g) 网络引导 pxelinux.cfg 中加入
        ks=http://HTTP_SERVER_IP/ks.cfg
```

问题总结:

制作 iso 镜像时注意在被制作文件的目录下

II. 实验：SSH 客户端登陆

服务名称	安装名称版本	服务端口信息	服务描述
sshd	Openssh-5.3p1-94 Openssh-Client Openssh-Server	Tcp 22	为客户端提供安全会话服务
Sftp	Sshd 服务器定义即可开启安全传输会话 Subsystem sftp /usr/libexec/openssh/sftp-server		
Dropbear(编译)	Dropbear-0.58 Dbclient Dropbearkey Scp	Tcp 22	与 sshd 雷同,但更加轻量级

配置文件:

配置文件：/etc/ssh/sshd_config

服务脚本：/etc/rc.d/init.d/sshd

脚本配置文件：/etc/sysconfig/sshd

实验思路:

实验步骤:

i) openssh 配置

```
vim /etc/ssh/sshd_config
Port: 22
ListenAddress: 172.16.43.100
Protocol V2
PermitRootLogin  # 限制 root 登陆
AllowUsers user1,user2,...: #用户白名单
AllowGroups:
DenyUsers : #用户黑名单
```

ii) dropbear 配置

```
安装配置 dropbear ：
1、编译安装
        # tar xf  dropbear-2013.58.tar.bz2
        # cd  dropbear-2013.58
        # ./configure
        # make PROGRAMS="dropbear dbclient dropbearkey scp"
        # make PROGRAMS="dropbear dbclient dropbearkey scp" install

2、服务脚本/etc/rc.d/init.d/dropbear
#!/bin/bash
#
# description: dropbear ssh daemon
# chkconfig: 2345 66 33
#
dsskey=/etc/dropbear/dropbear_dss_host_key
rsakey=/etc/dropbear/dropbear_rsa_host_key
lockfile=/var/lock/subsys/dropbear
pidfile=/var/run/dropbear.pid
dropbear=/usr/local/sbin/dropbear
```

```
dropbearkey=/usr/local/bin/dropbearkey

[ -r /etc/rc.d/init.d/functions ] && . /etc/rc.d/init.d/functions

[ -r /etc/sysconfig/dropbear ] && . /etc/sysconfig/dropbear

keysize=${keysize:-1024}
port=${port:-22}

gendsskey() {
        [ -d /etc/dropbear ] || mkdir /etc/dropbear
        echo -n "Starting generate the dss key: "
        $dropbearkey -t dss -f $dsskey &> /dev/null
        RETVAL=$?
        if [ $RETVAL -eq 0 ]; then
                success
                echo
                return 0
        else
                failure
                echo
                return 1
        fi
}

genrsakey() {
        [ -d /etc/dropbear ] || mkdir /etc/dropbear
        echo -n "Starting generate the rsa key: "
        $dropbearkey -t rsa -s $keysize -f $rsakey &> /dev/null
        RETVAL=$?
        if [ $RETVAL -eq 0 ]; then
                success
                echo
                return 0
        else
                failure
                echo
                return 1
        fi
}

start() {
        [ -e $dsskey ] || gendsskey
        [ -e $rsakey ] || genrsakey

        if [ -e $lockfile ]; then
                echo -n "dropbear daemon is already running: "
                success
                echo
```

```
                exit 0
        fi

        echo -n "Starting dropbear: "
        daemon --pidfile="$pidfile" $dropbear -p $port -d $dsskey -r $rsakey
        RETVAL=$?
        echo

        if [ $RETVAL -eq 0 ]; then
                touch $lockfile
                return 0
        else
                rm -f $lockfile $pidfile
                return 1
        fi
}

stop() {
        if [ ! -e $lockfile ]; then
                echo -n "dropbear service is stopped: "
                success
                echo
                exit 1
        fi

        echo -n "Stopping dropbear daemon: "
        killproc dropbear
        RETVAL=$?
        echo

        if [ $RETVAL -eq 0 ]; then
                rm -f $lockfile $pidfile
                return 0
        else
                return 1
        fi
}

status() {
        if [ -e $lockfile ]; then
                echo "dropbear is running..."
        else
                echo "dropbear is stopped..."
        fi
}

usage() {
        echo "Usage: dropbear {start|stop|restart|status|gendsskey|genrsakey}"
}
```

```
case $1 in
start)
        start ;;
stop)
        stop ;;
restart)
        stop
        start
        ;;
status)
        status
        ;;
gendsskey)
        gendsskey
        ;;
genrsakey)
        genrsakey
        ;;

*)
        usage
        ;;
esac
```

3、脚本配置文件/etc/sysconfig/dropbear

```
keysize=2048
port=22022
```

4、后配置

```
# chmod +x /etc/rc.d/init.d/dropbear
# chkconfig --add dropbear
```

III. 实验：openssl 实验企业自签证书认证

服务名称	安装名称版本	服务端口信息	服务描述
Openssl	Openssl-1.0.1		加密服务命令行客户端

配置文件:

/etc/pki/tls/openssl.cnf

实验思路:

1. 企业自签证书需要密钥对
2. 根据密钥队授权自己的自签证书
3. 客户端申请证书验证

实验步骤:

用 openssl 实现私有 CA：

生成密钥对：

(umask 077; openssl genrsa -out private/cakey.pem 2048)

如果想查看公钥：

openssl rsa -in private/cakey.pem -pubout -text -noout

生成自签证书：

openssl req -new -x509 -key private/cakey.pem -out cacert.pem -days 3655

创建需要的文件：

touch index.txt serial crlnumber

echo 01 > serial

用 openssl 实现证书申请：

在主机上生成密钥，保存至应用此证书的服务的配置文件目录下, 例如：

mkdir /etc/httpd/ssl

cd /etc/httpd/ssl

(umask 077; openssl genrsa -out httpd.key 1024)

生成证书签署请求：

openssl req -new -key httpd.key -out httpd.csr

将请求文件发往 CA；

CA 签署证书：

签署：

openssl ca -in /path/to/somefile.csr -out /path/to/somefile.crt -days DAYS

将证书传回请求者

吊销证书：

openssl ca -revoke /path/to/somefile.crt

IV. 实验: 正解反解区域传送子域授权,rdnc 管理

服务名称	安装名称版本	服务端口信息	服务描述
named	Bind-9.8.2	Tcp 23	为客户端提供 dns 解析服务
Rndc	随 bind 发行		Rndc 为管理 bind 提供方便途径

配置文件:

/etc/named/*.conf

实验思路:

实验步骤:

配置文件一览:

```
options {
   listen-on port 53 { 127.0.0.1; };  # 只监听在 127 的 53 端口
   directory      "/var/named";   # named 运行目录
   dump-file      "/var/named/data/cache_dump.db";
   statistics-file "/var/named/data/named_stats.txt";
   memstatistics-file "/var/named/data/named_mem_stats.txt";
   allow-query    { localhost; };
   #recursion yes;  此服务器允许递归
   #iteration yes;  此服务器允许迭代

   dnssec-enable yes;
   dnssec-validation yes;
   dnssec-lookaside auto;
};
```

做区域转发

```
zone "example.com" IN {
      type forward;
      forward only;
      forwarders { 172.16.0.1; };
};
```

1.递归查询:

一般客户机和服务器之间属递归查询，即当客户机向 DNS 服务器发出请求后,若 DNS 服务器本身不能解析,则会向另外的 DNS 服务器发出查询请求，得到结果后转交给客户机；

2.迭代查询(反复查询):

一般 DNS 服务器之间属迭代查询，如：若 DNS2 不能响应 DNS1 的请求，则它会将 DNS3 的 IP 给 DNS2，以便其再向 DNS3 发出请求；

举例：比如学生问老师一个问题，王老师告诉他答案这之间的叫递归查询。这期间也许王老师也不会，这时王老师问张老师，这之间的查询叫迭代查询！

```
cache 正向
        option
                可以递归 recursion yes

        rfc1912
                zone "magedu.com" IN {
                        type master;
                        file "magedu.com.zone"
                }

        在 var/named/ 建立 magedu.com.zone 文件

        $TTL 600
        @       IN      SOA dns.magedu.com.         dsadmin.magedu.com. (
                                                    2014040401
                                                    1H
                                                    5M
                                                    3D
                                                    12H)
        magedu.com. IN      NS      dns.magedu.com.
                            IN      NS      dns
        @                   IN      NS      dns
                            IN      MX 10 mail
        dns             IN      A       172.16.100.7
        www                 IN  A   172.16.100.1
        mail            IN      A       172.16.100.2
        ftp                 IN      CNAME       www
        pop                 IN  CNAME   mail

        chown root:named magedu.com.zone
        chmod 640 magedu.com.zone
        killall -1 named
        dig -t A www.magedu.com @172.16.100.7


cache 反向
        rfc1912
                zone "100.16.172.in-addr.apra" IN {
                        type master;
                        file "172.16.100.zone";
                };

        在 var/named/ 建立 172.16.100.zone 文件
                $TTL 600
                @       IN      SOA dns.magedu.com.         dsadmin.magedu.com. (
                                                            2014040401
                                                            1H
                                                            5M
```

```
                                        3D
                                        12H)
                        IN      NS      dns.magedu.com.
        7               IN  PTR         dns.magedu.com.
        1       IN  PTR www.magedu.com.
        2               IN      PTR     mail.magedu.com.
        //  ///  /  / / / // / / / / / / / /
        dns     IN      A       172.16.100.7
        www             IN  A   172.16.100.1
        mail    IN      A       172.16.100.2

        反向查询 dig -t PTR -x 172.16.43.2

        模拟区域传送 dig -t axfr magedu.com @172.16.100.7


主从同步:
    -1 获得上级授权
    1 在主服务器中添加从的位置信息 NS 对应 A 记录
        $TTL 600
        @       IN      SOA dns.magedu.com.         dsadmin.magedu.com. (
                                        2014040401
                                        1H
                                        5M
                                        3D
                                        12H)
                        IN      NS      dns
                        IN      NS      ns2   // 子的 NS
        ns2             IN  A   172.168.100.8  // 子的 A
                        IN      MX 10 mail
        dns     IN      A       172.16.100.7
        www             IN  A   172.16.100.1
        mail    IN      A       172.16.100.2
        ftp             IN      CNAME       www
        pop             IN  CNAME   mail

    2 在从服务器上 rfc1912 编辑区域
        zone "magedu.com" IN {
                type slave;
                masters { 172.16.100.7; };
                file "slave/magedu.linux.zone";
        };
        明确写明主要通知谁!!!
        允许传输 allow-transfer { ip }


子域授权:
    父域设置
```

```
[root@King100 ~]# cat /var/named/king.com.zone
$TTL 600
@       IN      SOA     dns.king.com. admin.king.com. (
                        2014040402
                        1H
                        5M
                        3D
                        12H )
        IN      NS      dns
tech.king.com.          IN      NS      dns.tech.king.com.
        IN      MX  10  mail
dns.tech.king.com.      IN      A       172.16.43.200
dns     IN      A       172.16.43.100
mail    IN      A       172.16.43.2
www     IN      A       172.16.43.1
img     IN      A       172.16.43.88
ftp     IN      CNAME   www

[root@King100 ~]# cat /var/named/172.16.43.zone
$TTL 600
@       IN    SOA    dns.king.com.   admin.king.com. (
                2014040402
                1H
                5M
                3D
                12H )
     IN     NS     dns.king.com.
tech.king.com.  IN     NS     dns.tech.king.com.
200     IN     PTR    dns.tech.king.com.
100     IN      PTR    dns.king.com.
2       IN      PTR    mail.king.com.
1       IN      PTR    www.king.com.
88      IN      PTR    img.king.com.
```

子域的设置

```
[root@King200 ~]# cat /etc/named.rfc1912.zones
zone "tech.king.com" IN {
        type master;
        file "tech.king.com.zone";
};

zone "43.16.172.in-addr.apra" IN {
        type master;
        file "172.16.43.zone";
};

[root@King200 ~]# cat /var/named/tech.king.com.zone
```

```
$TTL 600
@       IN      SOA     dns.tech.king.com.    admin.tech.king.com. (
                        2014010401
                        1H
                        5M
                        3D
                        12H )
        IN      NS      dns
dns     IN      A       172.16.43.200
www     IN      A       172.16.43.201
img     IN      A       172.16.43.202

[root@King200 ~]# cat /var/named/172.16.43.zone
$TTL 600
@       IN      SOA     dns.tech.king.com.    admin.tech.king.com. (
                        2014010401
                        1H
                        5M
                        3D
                        12H )
        IN      NS      dns.tech.king.com.
200     IN      PTR     dns.tech.king.com.
201     IN      PTR     www.tech.king.com.
202     IN      PTR     img.tech.king.com.
```

rndc 配置文件 /etc/rndc.conf

rndc-confgen -r /dev/urandom

生成 key 文件　后一半放置到 named.conf 文件中

视图功能: 在 dns 主机上

```
[root@King100 ~]# cat /etc/named.conf
// 注意此处所有区域都应该放入到视图中

acl intranet { 192.168.100.0/24;

};

acl internet { ! 192.168.100.0/24; any; };

view "lan" {
        match-clients { "intranet"; };

        zone "." IN {
                type hint;
                file "named.ca";
```

```
        };

        zone "intranet.king.com" IN {
                type master;
                file "intranet.king.com.zone";
                //  www.intranet.king.com
        };
};

view "wan" {
        match-clients { "internet"; };

        zone "." IN {
                type hint;
                file "named.ca";
        };

        zone "internet.king.com" IN {
                type master;
                file "internet.king.com.zone";
                // www.internet.king.com
        };
}
```

安全传输: TSIG Key

```
// 生成 dns key
dnssec-keygen -a hmac-sha256 -b 128 -n HOST host1-host2.

// 在服务器要添加
key host1-host2. {
 algorithm hmac-sha256;
 secret "La/E5CjG9O+os1jq0a2jdA==";
};

// 在要传输的客户端添加
server 10.1.2.3 {
 keys { host1-host2. ;};
};

allow-update { key host1-host2. ;};
```

智能 DNS 方案:

方案描述将区域信息全部存入到 db 库中,在 dns 每次查询时到数据中查询

在 bind-9.8.5 之后,编译安装 --with-dlz-mysql=/usr/local/mysql

添加 acl 访问列表

网络获取存放于 /etc/named/

在区域文件中添加数据库的访问

database "mysqldb smartdns CNC localhost root 123456"; #这里定义的数据库相关的配置，字段分别为：SQLtype DataBase Table Host User Password

V. 实验: LAMP 架构组合及优化

服务名称	安装名称版本	服务端口信息	服务描述
Httpd	Httpd-2.4.9	Tcp 80	为客户端提供 http 服务静态内容
Mysqld,mysql	Mysql-5.5.33 Mysql-Server-5.5.33	Tcp 3306	为客户端提供数据存储服务
Php	Php-5.4.26 Php-common Php-zts		为客户端提供 http 动态网页内容 模块化 httpd 方式
Php-fpm	Php-fpm	Tcp 9000	独立进程 php
Xcache	Xcache-3.0.10		为 php 提供 opcode 级别速度优化(HHVM)

配置文件:

httpd: /etc/httpd/*.conf
/etc/httpd/extra/*.conf
mysql: /etc/my.cnf
php: /etc/php.ini

实验思路:

实验步骤:

1. httpd 基本操作

```
1. httpd-apache.2.4.9 安装过程
编译需要依赖 pcre-devel-7.8-6.el6.bz2  apr-1.5.0.bz2  apr-util-1.5.3.bz2
按需进行依次编译安装
# tar xf pcre-devel-7.8-6.el6.bz2
# cd pcre-devel-7.8-6
# ./configure --prefix=/usr/local/pcre
# make && make install

# tar xf apr-1.5.0.bz2
# cd apr-1.5.0
# ./configure --prefix=/usr/local/apr
# make && make install

# tar xf apr-util-1.5.3.bz2
# cd apr-util-1.5.3
# ./configure --prefix=/usr/local/apr-util --with-apr=/usr/local/apr
# make && make install

# 安装之前请确保系统之前预装的 httpd 已被卸载
# tar xf httpd-2.4.9.bz2
# cd httpd-2.4.9
# 参数依次是: httpd 安装路径  httpd 配置文件存放路径  启用模块化方式  启用 ssl 安全连接
```

```
# 启用 cgi 脚本功能 启用 url 重写 启用服务器压缩 启用正则表达式支持  apr 安装路径
# apr util 安装路径  启用常用模块以确保 apache 正常工作  将多进程模型非静态化
# 启用事件异步模型
# ./configure --prefix=/usr/local/apache --sysconfdir=/etc/httpd --enable-so --enable-ssl --enable-cgi --enable-rewrite --with-zlib --with-pcre=/usr/local/pcre --with-apr=/usr/local/apr --with-apr-util=/usr/local/apr-util/ --enable-modules=most --enable-mpms-shared=all --with-mpm=event
# make && make install

2 .实战: 配置 CGI、虚拟主机、https、mod_deflate、mod_status ,
A) 配置 CGI、(在部分业务中常见,例如邮箱)

# 在 /etc/named/named.conf 文件中找到 alias_module 模块
# 替换 cgi-bin 的输出路径
<IfModule alias_module>
        # 在此路径下可编写 shell 脚本实现 cgi 方式输出 html 内容
        ScriptAlias /cgi-bin/ "/usr/local/apache/cgi-bin/"
</IfModule>

# cgi-bin 示例
#!/bin/bash
#
userinfo=`cat /etc/passwd | grep ^root`

cat << EOF
Content-Type: text/html

<pre>
The hostname is: `hostname`.
The time is: `date`.

The User Information: ${userinfo}.
</pre>

EOF

B) 虚拟主机: (提高主机生产力)
        一机多站点成为可能,需要取消中心主机在 2.4.x + 的版本上配置虚拟主机已经不需要支持 NameVirtualHost

具体配置如下:
# 在 httpd.conf 中开启 vhost 子配置文件
# Include /etc/httpd/extra/httpd-vhosts.conf
# vim /etc/httpd/extra/httpd-vhosts.conf
# 在此要注意关闭之前的主机配置也就是 httpd.conf
```

```
# 中的<Directory>节点

<VirtualHost *:80>
   DocumentRoot "/var/www/html/a.com"
   ServerName www.a.com
   ServerAlias www.aa.com
   <Directory />
      AllowOverride none
      Require all granted
   </Directory>
   ErrorLog "logs/dummy-host.example.com-error_log"
   CustomLog "logs/dummy-host.example.com-access_log" common
</VirtualHost>

<VirtualHost *:80>
   DocumentRoot "/var/www/html/b.org"
   ServerName www.b.org
   ErrorLog "logs/dummy-host2.example.com-error_log"
   <Directory />
      AllowOverride none
      Require all granted
   </Directory>
   CustomLog "logs/dummy-host2.example.com-access_log" common
</VirtualHost>

C) mod_deflate (节约带宽成本)
        启用压缩模块,gzip,deflate

# 在 httpd.conf 中开启压缩模块
# LoadModule deflate_module modules/mod_deflate.so

# 在 httpd.conf 中编写如下配置
<IfModule deflate_module>
AddOutputFilterByType DEFLATE text/plain
AddOutputFilterByType DEFLATE text/html
AddOutputFilterByType DEFLATE application/xhtml+xml
AddOutputFilterByType DEFLATE text/xml
AddOutputFilterByType DEFLATE application/xml
AddOutputFilterByType DEFLATE application/x-javascript
AddOutputFilterByType DEFLATE text/javascript
AddOutputFilterByType DEFLATE text/css

# 压缩等级(0-9),数值越大 CPU 压力也就越大
DeflateCompressionLevel 9

# 非 Mozilla4 标准浏览器仅开启 gzip 压缩
BrowserMatch ^Mozilla/4 gzip-only-text/html
```

```
# Mozilla4.06,4.07,4.08 版本不开启压缩
BrowserMatch ^Mozilla/4\.0[678] no-gzip

# IE 浏览器不开启压缩
BrowserMatch \bMSI[E] !no-gzip !gzip-only-text/html
</IfModule>
```

D) setHandler (apache 自带探测功能)

可访问服务器信息,为服务器监控提供便利

```
# 需要注意: SetHandler 中的内容必要要经过用户认证
# htpasswd -c /etc/httpd/conf/.htpasswd -M tom
# -c 生成 .htpasswd 文件 -M md5 加密用户密码
# 在 httpd.conf 中编写如下配置

<Location /server-status>
   SetHandler server-status
   AuthType Basic
   AuthName "Server Status"
   AuthUserFile "/etc/httpd/conf/.htpasswd"
   Require valid-user
   Order deny,allow
   Allow from all
</Location>

# 其他 SetHandler 可设置的值及对应模块
default-handler: 使用 default_handler() 发送文件，它是用来处理静态内容的处理器(核心)。
send-as-is: 直接发送，不增加 HTTP 头(mod_asis)。
cgi-script: 按 CGI 脚本处理(mod_cgi)。
imap-file: 按 imagemap 规则处理(mod_imagemap)。
server-info: 取得服务器配置信息(mod_info)。
server-status: 取得服务器状态报告(mod_status)。
type-map: 用于内容协商，按类型映射文件处理(mod_negotiation)。
```

https (为私密信息保驾护航)

https 链接采用证书认证流程,客户端发送链接请求与服务器进行握手,协商加密方式,

服务器发送带有证书签发者与自身信息的证书给客户端验证,客户端验证完毕即建立

https 安全链接通信.

1) 证书服务器自签证书阶段(需要密钥对一组,自签证书一张)

```
# cd /etc/pki/CA
```

```
# openssl genrsa -out private/cakey.pem 2048
# openssl req -new -x509 -key private/cakey.pem -out cacert.pem -days 3650
# touch index.txt serial calnumber
# echo 01 > serial
2) Web 服务器安装 mod_ssl 模块
# yum -y install mod_ssl

3) Web 服务器申请证书阶段
# cd /var/www/html
# openssl genrsa -out httpd.key 1024
# openssl req -new -key httpd.key -out httpd.csr

4) 证书服务器签署证书
# openssl ca -in /var/www/html/httpd.csr -out /var/www/html/httpd.crt -days
365

5) 启用 ssl 模块与配置文件
# 在 httpd.conf 中开启 ssl_module 模块和 ssl 的配置文件
# LoadModule ssl_module modules/mod_ssl.so
# Include /etc/httpd/extra/httpd-ssl.conf
# vim /etc/httpd/extra/httpd-ssl.conf
<VirtualHost *:443>
DocumentRoot "/var/www/html"
ServerName www.king.com:443
ErrorLog "/usr/local/apache/logs/error_log"
TransferLog "/usr/local/apache/logs/access_log"
SSLEngine on
SSLCertificateFile "/var/www/html/httpd.crt"
SSLCertificateKeyFile "/var/www/html/httpd.key"

6) 重启 apachectl 服务
        # apachectl restart

7) 测试
```

2. httpd+php-fpm+cache+nfs

```
分析如下:
        step1 (172.16.43.1) DNS 用途: 为客户端请求做解析服务,将请求定向到
web1 或者 web2 中
        step2 (172.16.43.2,172.16.43.3) 安装 apache 服务,实现对 dns 交来请求
的处理
        step3 (172.16.43.4) 安装 nfs 为 apache 提供静态内容信息,php-fpm 处理
动态内容
        step4 (172.16.43.5) 安装 mysql 为 php 的数据请求提供服务
        step5 (172.16.43.4) 安装 xcache,为 php 提供 opcode 级别的加速服务
step1 dns
a) 安装 bind(dns)服务器
```

```
yum -y install bind

b) 配置 cache only 的 dns 服务器
i) 在 vim /etc/named/named.conf 中输入以下内容
        options {
          directory      "/var/named";
          dump-file      "/var/named/data/cache_dump.db";
          statistics-file "/var/named/data/named_stats.txt";\
          allow-query    { any; };  # 允许任意客户端查询
          recursion yes;
          rrset order { random; };   # dns 轮询解析
        }

        logging {
          channel default_debug {
                file "data/named.run";
                severity dynamic;
          };
        };

        zone "." IN {
              type hint;
              file "named.ca";
        };

        include "/etc/named.rfc1912.zones";

                ii) 在 vim /etc/named.rfc1912.zones 中输入以下内容
        zone "king.com" IN {
              type master;
              file "king.com.zone";   #正向解析配置文件
        };

        zone "43.16.172.in-addr.arpa" IN {
              type master;
              file "172.16.43.zone";  #反向解析配置文件
        };

iii) 在 vim /var/named/king.com.zone 中输入以下内容
        $TTL 600
        @      IN     SOA    dns.king.com.   admin.king.com. (
                        2014032401
                        1H
                        5M
                        3D
                        12H )
              IN     NS     dns
        dns    IN     A      172.16.43.1
```

```
        www    IN     A      172.16.43.2
        www    IN     A      172.16.43.3
```

iv) 在 vim /var/named/172.16.43.zone 中输入以下内容

```
        $TTL 600
        @      IN     SOA    dns.king.com.   admin.king.com. (
                              2014032401
                              1H
                              5M
                              3D
                              12H )
             IN     NS     dns.king.com.
        1     IN     PTR    dns.king.com.
        2     IN     PTR    www.king.com.
        3     IN     PTR    www.king.com.
```

v) 更改 iii,iv 步的权限及属主属组

```
        chown root:named /var/named/*.zone
        chmod 640 /var/named/*.zone
        service named start
```

vi) 测试:

```
        # vim /etc/resolv.conf 更改 DNS1=172.16.43.1
        dig -t A www.king.com
        客户端查询
        dig -t PTR -x 172.16.43.2
```

step2 apache

a) 为安装准备环境

```
        rpm -e httpd --nodeps
        rm -rf /etc/httpd
        rm -rf /usr/lib64/httpd
        yum -y install pcre-devel
```

b) 其余手动安装配置

```
        i) 配置安装 apr-1.5.0
        tar xf apr-1.5.0.tar.bz2
        cd apr-1.5.0
        ./configure --prefix=/usr/local/apr
        make && make install

        ii) 配置安装 apr-util-1.5.3
        tar xf apr-util-1.5.3.tar.bz2
        cd apr-util-1.5.3
        ./configure --prefix=/usr/local/apr-util --with-apr=/usr/local/apr
        make && make install

        iii) 配置安装 httpd-2.4.9
```

```
        tar xf httpd-2.4.9.tar.bz2
        cd httpd-2.4.9
        ./configure --prefix=/usr/local/apache --sysconfdir=/etc/httpd --enable-
so --enable-ssl --enable-cgi --enable-rewrite --with-zlib --with-pcre --with-
apr=/usr/local/apr --with-apr-util=/usr/local/apr-util --enable-modules=most -
-enable-mpms-shared=all --with-mpm=event
        make && make install

c) 配置开机服务及环境变量
        在 vim /etc/rc.d/init.d/httpd 编辑此项
        请参考 http://apprentice.blog.51cto.com/2214645/1380490
        在 vim /etc/httpd/httpd.conf 添加此项
        PidFile "/var/run/httpd.pid"
        # 导出环境变量
        echo "PATH=/usr/local/apache/bin:$PATH" > /etc/profile.d/httpd.sh
        # 重载环境变量
        . /etc/profile.d/httpd.sh
        # 启动/添加开机服务
        chkconfig --add httpd
        chkconfig httpd on
        service httpd start

d) 测试前 3 台机器是否工作正常


step3 nfs,php-fpm
a) 安装 nfs 工具
        # 如果机器内核没有编译 nfs 则需要安装
        # yum -y install nfs-utils
        mkdir /nfsshared
        在 vim /etc/exports 编辑此项
        # 为 172.16.43.0/24 此段开放加载 nfs 网络文件功能
        /nfsshared     172.16.43.0/24(rw,async,no_root_squash)
        # 添加客户端准备操作的账户,防止 nobody 属组文件出现
        # useradd -r -s /sbin/nologin daemon
        # 导出所有挂载分区
        exportfs -a
        service nfs restart

b) 客户端测试挂载
        showmount -e 172.16.43.4
        mkdir /share
        # 挂载网络文件系统
        mount -t nfs 172.16.43.4:/nfsshared /share
        # 开机挂载可编辑 /etc/fstab
        172.16.43.4:/nfsshared /share  ext4
        # 测试 nfs, 在/nfsshared 中 touch 文件,观察 2 个客户端(web1,web2)
```

c) 安装 php-fpm

i) 解决 php 安装依赖

```
yum -y groupinstall "Desktop Platform Development"
yum -y install libmcrypt-devel
yum -y install bzip2-devel
```

ii) 安装 php with fpm

```
tar xf php-5.4.19.tar.bz2
cd php-5.4.19
./configure --prefix=/usr/local/php --with-mysql=mysqlnd --with-pdo-mysql=mysqlnd --with-mysqli=mysqlnd --with-openssl --enable-mbstring --with-freetype-dir --with-jpeg-dir --with-png-dir --with-zlib --with-libxml-dir=/usr --enable-xml --enable-sockets --enable-fpm --with-mcrypt --with-config-file-path=/etc --with-config-file-scan-dir=/etc/php.d --with-bz2
make && make install
```

iii) 为 php 提供配置文件：

```
cp php.ini-production /etc/php.ini
```

iv) 配置 php-fpm

为 php-fpm 提供 SysV init 脚本，并将其添加至服务列表：

```
cp sapi/fpm/init.d.php-fpm  /etc/rc.d/init.d/php-fpm
chmod +x /etc/rc.d/init.d/php-fpm
chkconfig --add php-fpm
chkconfig php-fpm on
```

为 php-fpm 提供配置文件：

```
cp /usr/local/php/etc/php-fpm.conf.default /usr/local/php/etc/php-fpm.conf
```

编辑 php-fpm 的配置文件：vim /usr/local/php/etc/php-fpm.conf

配置 fpm 的相关选项为你所需要的值，并启用 pid 文件（如下最后一行）：

```
pid = /usr/local/php/var/run/php-fpm.pid
listen = 172.16.43.4:9000
```

vi) 编辑 apache 为运行 php 后端程序做准备

```
# 分别编辑 172.16.43.2/3 中的 vim /etc/httpd/httpd.conf
DocumentRoot "/share"
<Directory "/share">
# 开启 fcgi 转发
LoadModule proxy_module modules/mod_proxy.so
LoadModule proxy_fcgi_module modules/mod_proxy_fcgi.so
# 禁止 apache 正向代理,转而实现反向代理后端的 php-fpm
ProxyRequests Off
ProxyPassMatch ^/(.*\.php)$ fcgi://172.16.43.4:9000/nfsshared
# 添加 apache 识别 php 文件
AddType application/x-httpd-php  .php
```

AddType application/x-httpd-php-source .phps
定位至 DirectoryIndex
DirectoryIndex index.php index.html

d) 安装配置 Discuz 在 /nfsshared
cd /nfsshared
下载 Discuz 最新版本到此目录
unzip Discuz_X3.1_SC_UTF8.zip
rm -rf Discuz_X3.1_SC_UTF8.zip readme/ utility/
mv ./upload/* ./
rm -rf upload/

测试 php 页面解析
图:

安装 discuz 需要将 config , data , uc_client , uc_server 的权限
设置为 777
图:

step4 mysql
i) 创建 mysql 的数据目录
mkdir /data
groupadd -r mysql
useradd -g mysql -r -s /sbin/nologin -M -d /data mysql
chown -R mysql:mysql /data

ii) 安装二进制 mysql
tar xf mysql-5.5.33-linux2.6-x86_64.tar.gz -C /usr/local
cd /usr/local
ln -sv mysql-5.5.33-linux2.6-x86_64 mysql
cd mysql
chown -R mysql:mysql .
mysql/scripts/mysql_install_db --user=mysql --datadir=/data
chown -R root .
提供 mysql 的配置文件
cp support-files/my-large.cnf /etc/my.cnf
# 需要添加如下行指定 mysql 数据文件的存放位置：
datadir = /mydata/data

iii) 为 mysql 提供 sysv 服务脚本：
cd /usr/local/mysql
cp support-files/mysql.server /etc/rc.d/init.d/mysqld
chmod +x /etc/rc.d/init.d/mysqld
添加至服务列表：
chkconfig --add mysqld
chkconfig mysqld on

```
        echo "export PATH=/usr/local/mysql/bin:$PATH" >
/etc/profile.d/mysql.sh
        . /etc/profile.d/mysql.sh

        iv) 启动服务并授权 php 服务器账号访问
        service mysqld restart
        mysql
        grant all on *.* to 'root'@'172.16.43.4' identified by '123456';
        flush privileges;

step5 xache
        i) 安装 xcache
        tar xf xcache-3.0.3.tar.bz2
        cd xcache-3.0.3
        /usr/local/php/bin/phpize
        ./configure --enable-xcache --with-php-config=/usr/local/php/bin/php-
config
        make && make install
        # 安装结束时，会出现类似如下行, 将后半句复制
        Installing shared extensions:    /usr/local/php/lib/php/extensions/no-
debug-non-zts-20100525/

        ii) 编辑 php.ini，整合 php 和 xcache：
        # 首先将 xcache 提供的样例配置导入 php.ini
        mkdir /etc/php.d
        # xcache.ini 文件在 xcache 的源码目录中。
        cp xcache.ini /etc/php.d
        # 接下来编辑/etc/php.d/xcache.ini 修改为如下：
        extension = /usr/local/php/lib/php/extensions/no-debug-non-zts-
20100525/xcache.so

        service php-fpm restart

        在 /nfsshared 中建立 testxcache.php 文件测试 xcache 的启用情况
```

VI. 实验：配置完成 vsftpd 共享与用户身份校验

服务名称	安装名称版本	服务端口信息	服务描述
vsftpd	vsftpd-2.2.2	Tcp 21	为客户端提供文件共享服务
Mysql(略)			
Pam_mysql	Pam_mysql-0.7		连接 vsftpd 与 mysql 实现 pam 认证框架的拓展

配置文件:

vsftpd: /etc/vsftpd.conf

实验思路:

安装启动服务与 pam.d 认证框架结合 mysql 实现对用户认证过程

实验步骤:

一、安装所需要程序

1、安装 mysql 和 pam_mysql

```
# yum -y install vsftpd mysql-server mysql-devel pam_mysql
```

注意：pam_mysql 由 epel 源提供。

二、创建虚拟用户账号

1.准备数据库及相关表

首先请确保 mysql 服务已经正常启动。而后，按需要建立存储虚拟用户的数据库即可，这里将其创建为 vsftpd 数据库。

```
mysql> create database vsftpd;

mysql> grant select on vsftpd.* to vsftpd@localhost identified by
'www.magedu.com';
mysql> grant select on vsftpd.* to vsftpd@127.0.0.1 identified by
'www.magedu.com';
mysql> flush privileges;

mysql> use vsftpd;
mysql> create table users (
   -> id int AUTO_INCREMENT NOT NULL,
   -> name char(20) binary NOT NULL,
   -> password char(48) binary NOT NULL,
   -> primary key(id)
   -> );
```

2、添加测试的虚拟用户

根据需要添加所需要的用户，需要说明的是，这里将其密码为了安全起见应该使用 PASSWORD 函数加密后存储。

```
mysql> insert into users(name,password) values('tom',password('magedu'));
mysql> insert into users(name,password) values('jerry',password('magedu'));
```

三、配置 vsftpd
1.建立 pam 认证所需文件
#vim /etc/pam.d/vsftpd.mysql
添加如下两行
auth required /lib64/security/pam_mysql.so user=vsftpd
passwd=www.magedu.com host=localhost db=vsftpd table=users
usercolumn=name passwdcolumn=password crypt=2
account required /lib64/security/pam_mysql.so user=vsftpd
passwd=www.magedu.com host=localhost db=vsftpd table=users
usercolumn=name passwdcolumn=password crypt=2
注意：由于 mysql 的安装方式不同，pam_mysql.so 基于 unix sock 连接 mysql 服务器时可能会出问题，此时，建议授权一个可远程连接的 mysql 并访问 vsftpd 数据库的用户。

2.修改 vsftpd 的配置文件，使其适应 mysql 认证
建立虚拟用户映射的系统用户及对应的目录
useradd -s /sbin/nologin -d /var/ftproot vuser
chmod go+rx /var/ftproot
请确保/etc/vsftpd/vsftpd.conf 中已经启用了以下选项
anonymous_enable=YES
local_enable=YES
write_enable=YES
anon_upload_enable=NO
anon_mkdir_write_enable=NO
chroot_local_user=YES
而后添加以下选项
guest_enable=YES
guest_username=vuser
并确保 pam_service_name 选项的值如下所示
pam_service_name=vsftpd.mysql

四、启动 vsftpd 服务

service vsftpd start
chkconfig vsftpd on
查看端口开启情况
netstat -tnlp |grep :21

使用虚拟用户登录,验正配置结果，以下为本机的命令方式测试，你也可以在其它 Win Box 上用 IE 或者 FTP 客户端工具登录验正
ftp localhost

五、配置虚拟用户具有不同的访问权限

vsftpd 可以在配置文件目录中为每个用户提供单独的配置文件以定义其 ftp 服务访问权限，每个虚拟用户的配置文件名同虚拟用户的用户名。配置文件目录可以是任意未使用目录，只需要在 vsftpd.conf 指定其路径及名称即可。

1、配置 vsftpd 为虚拟用户使用配置文件目录
vim /etc/vsftpd/vsftpd.conf
添加如下选项
user_config_dir=/etc/vsftpd/vusers_config

2、创建所需要目录，并为虚拟用户提供配置文件
mkdir /etc/vsftpd/vusers_config/
cd /etc/vsftpd/vusers_config/
touch tom jerry

3、配置虚拟用户的访问权限
虚拟用户对 vsftpd 服务的访问权限是通过匿名用户的相关指令进行的。比如，如果需要让 tom 用户具有上传文件的权限，可以修改/etc/vsftpd/vusers/tom 文件，在里面添加如下选项即可。
anon_upload_enable={YES|NO}
anon_mkdir_write_enable={YES|NO}
anon_other_write_enable={YES|NO}

VII. 实验:完成文件同步与安全策略配置

服务名称	安装名称版本	服务端口信息	服务描述
samba	Samba4-4.0.0 Samba4-Common Samba4-Client Samba4-Winbind Samba4-swat	Udp 137,138 Tcp 139,445	为客户端提供文件同步
NFS	Nfs-util-1.2.3	Tcp,Udp 2049 Tcp,Udp 111	雷同 samba 使用略显生涩

配置文件:

samba: /etc/samba/smb.conf

实验思路:

samba 基于 smb 协议进行局域网共享文件服务,与基于 cifs 的 win32 进行共享.设置独立密码和共享模块

实验步骤:

```
smbclient
        检查服务器上的共享：
                smbclient -L Server -U username
        以交互式模式连入服务器的某共享：
                smbclient //Server/Shared -U username

# 添加 samba 账户
        useradd samba
# 安装 samba
        yum –y install samba4
# 新建一个共享文件系统
        vim + /etc/samba/smb.conf

        [myshred]
        comment = this is samba share
        # 此 path 路径需要让 samba 账户可写可读
        path = /sambaShared
        read only = yes
        writable = no
        browseable = yes
        public = yes  # 普通*nix 访问
        guest ok = yes  # 来宾用户访问 win32 共享使用
        write list =
                # 其他人只读列出列表,mygroup 只写
                @mygroup

测试配置文件语法误，并显示最终生效的配置：
        # testparm

*nix 挂载 cifs 文件系统
        # mount -t cifs //172.16.43.1/myshared  /data -o username=smbuser
```

使用 GUI 方式配置 samba

samba-swat：

```
vim /etc/xinetd.d/swap
disable = no
# 图形化终端只允许内部访问
only_from = 172.16.0.0/16
```

VIII. 实验: iptables 配置 netfilter, dnat,snat,layer7 功能

服务名称	安装名称版本	服务端口信息	服务描述
iptables	Iptables-1.4.20		内核级 tcp/ip 协议栈,提供本主机流控限制
Layer7	l7-protocols-2009-05-28 netfilter-layer7-v2.23		为 iptables 附加全七层协议栈支持
Kernel	kernel-2.6.32-431.5.1.el6.src.rpm		附加七层协议栈承载内核

配置文件:

iptables: iptables-save > /path/to/file
iptables-restore < /path/to/file
/proc/sys/net*

实验思路:

内核级 tcp/ip 协议栈原本没有提供 7 层协议限制,需要先为内核打补丁,往 iptables 中加入过滤协议从新安装.在 iptables 中加入 snat,masquerade 来代理客户端上互联网

实验步骤:

0.准备工作

l7-protocols-2009-05-28.tar.gz
iptables-1.4.20.tar.bz2
kernel-2.6.32-431.5.1.el6.src.rpm
netfilter-layer7-v2.23.tar.bz2

1. 安装编译内核源码

useradd mockbuild
rpm -ivh kernel-2.6.32-431.5.1.el6.src.rpm
此时用户根目录下会生成 rpmbuild 目录
cd rpmbuild/SOURCES
tar xf linux-2.6.32-431.5.1.el6.tar.bz2 -C /usr/src
cp /boot/config-2.6.32-431.el6.x86_64 ./.config
patch -p1 < /root/netfilter-layer7-v2.23/kernel-2.6.32-layer7-2.23.patch
make menuconfig -> / 搜索 layer7 即可
make -j 4
make moduals_install
make install
vim /etc/grub.conf
init 6

2. 安装新 iptables 与 l7layer 模块

i) 卸载旧的 iptables

cp /etc/sysconfig/iptables-config ~
cp /etc/rc.d/init.d/iptables ~
rpm -e iptables

ii) 安装新的 iptables

libxt_layer7.c libxt_layer7.man

cp netfilter-layer7-v2.23/iptables-1.4.3forward-for-kernel-2.6.20forward/* ~/iptables-1.4.20/extensions/

cd iptables-1.4.20

.configure --prefix=/usr/local/iptables --with-ksource=/usr/src/linux

make && make install

cp iptables-config /etc/sysconfig/

cp iptables /etc/rc.d/init.d/

iii) 安装 l7layer 模块

modprope nf_conntrack

vim /etc/sysctl.conf

-> ip_forward = 1

-> nf_conntrack_acct = 1

sysctl -p # 重新加载

cd l7-protocols-2009-05-28

make install

3). 测试环境

注意此时带有 l7layer 的机器可以理解为一台路由器

那么客户端的网关需要指定这台机器

客户端 带有 l7layer 的机器 外网

10.10 10.10 172.10 172.1

iptables -t nat -A POSTROUTING -s 10.0.0.0/8 -j SNAT --to-source 172.16.0.10

iptables -t nat -A POSTROUTING -s 10.0.0.0/8 -j MARQUERADE

iptables -A FORWARD -i eth0 -m layer7 --L7proto qq -j REJECT

IX. 实验: rsync+inofied 配置完成文件服务器时时推送

服务名称	安装名称版本	服务端口信息	服务描述
rsync	3.0.6-9.el6_4.1	Tcp: 873 Udp: 873	为本机建立共享模块. 利用此模块可完成 pull,push 操作
xinetd	2:2.3.14-39.el6_4		为 iptables 附加全七层协议栈支持
Inotifytool	3.14-1.el6		附加七层协议栈承载内核

配置文件:

rsync: /etc/rsyncd.conf

inotify: /proc/sys/fs/inotify/*

实验思路:

利用 inotify 接口对文件进行监控,在发生变化后将变化文件增量到远程目录

实验步骤:

i. 配置 rsync 文件 包括全局和共享模块

```
yum -y install xinetd rsync
vim /etc/xinetd.d/rsync  disabled = no
新建 vim /etc/rsyncd.conf

# Global Settings
uid = nobody
gid = nobody
use chroot = no
max connections = 10
strict modes = yes
pid file = /var/run/rsyncd.pid
log file = /var/log/rsyncd.log

# Directory to be synced
[synced_name]
path = /path/to/some_dir
ignore errors = yes
read only = no
write only = no
hosts allow = white_list_ip/net
hosts deny = *
list = false
uid = root
gid = root
auth users = username
secrets file = /etc/rsyncd.passwd
```

ii) 配置权限文件 要求访问权限为 600

```
vim /etc/rsyncd.passwd   -> Tom:123456
```

```
        chmod 600 /etc/rsyncd.passwd

iii) 启动 xinetd  测试服务
        chkconfig rsync on
        service xinetd start
        ss -tunl | grep "873"

iv) 安装 inotify 模块
        yum -y install inotify
        inotifywait -mrq -e modify,create,move,delete /var/www/html
        rsync -azH --delete /var/www/html 远程主机

        # 注意这里的 dst 也要建立一个 rsync 的共享模块区域
        #!/bin/bash
        #

        host=172.16.43.100
        src=/var/www/html
        dst=/Users/King/test
        user=tom
        /usr/bin/inotifywait -mrq -e modify,delete,create,attrib
/var/www/html/ | while read file
        do
        /usr/bin/rsync -vzrtopg --delete --progress  --password-
file=/etc/rsyncd.passwd $src $user@$host::$dst
        echo "${file} was rsynced" >> /tmp/rsync.log 2>&1
        done
```

X. 实验：rsyslog 与 mysql 实现日志收集与分析

服务名称	安装名称版本	服务端口信息	服务描述
Rsyslog-mysql	5.8.10-8.el6		此模块可将系统日志通过远程传送到 mysql
Mysql(略)			
loganalyzer	loganalyzer-3.6.4.tar.gz		在 mysql 中的数据进行统计与分析工具

配置文件:

/etc/rsyslog.conf

实验思路:

将本主机所产生任意规则的日志全部转移到远程(日志服务器),在日志服务器中使用 loganalyzer 处理分析日志

实验步骤:

i) 安装 rsyslog-mysql,mysql 导入数据,配置权限

```
yum –y install rsyslog-mysql mysql
mysql < /usr/share/doc/rsyslog-mysql/createDB.sql
```

ii) 配置加载 ModLoad 模块 记录数据类型

```
vim /etc/rsyslog.conf
# 以下必须编辑在配置文件的 ****modules**** 处
$ModLoad ommmysql
$ModLoad imudp.so
$UDPServerRun 514
# 以下必须编辑在配置文件的 ****rules**** 处
*.* :ommysql:172.16.100.1,Syslog,rsysloguser,rsyslogp@ss
```

iii) 安装 rsyslog

```
tar xvzf loganalyzer-3.6.4.tar.gz
cd loganalyzer-3.6.4
mv src/* /usr/local/apache/htdocs/syslog/
mv contrib/* /usr/local/apache/htdocs/syslog/
chmod u+x /usr/local/apache/htdocs/syslog/*.sh
./configure.sh
./secure.sh
chmod 666 config.php
chown -R daemon.daemon *
```

iv) 网页登陆并测试