```
################################################
# File name   : PhaseII.fkoo.website.testing
# Description : fkoo二期网站系统架构虚拟实现
# Requirement : Vmware workstation *1 (含 VMware-Tools Linux.iso)
#               Windows XP PC *1
#               RHEL5.1 ISO光盘文件
#
# Copyright(C), fkoo, 2009, All Rights Reserved.
#
# Author: Far Young Chen / fkoo (fkoo.com@gmail.com)
# URL: http://www.fkoo.net
#
#################################################

#***************************************************************************************************
#                                     Network Layout
#***************************************************************************************************


                                         { Client LAN }
   ------------------------------- 192.168.1.0 / 255.255.255.0 -------------------------------------
         |     :     |            |           |           |            |           |           |
<Master> |..[ VIP ]..|  <Passive>  |           |           |            |           |           |
      \  |           |  /         |           |           |            |           |           |
      LVS201      LVS202       WEB203      WEB204      WEB205       IMG206      IMG207      IMG208
         |           |            |           |           |            |           |           |
         |.heartbeat.|            |           |           |            |           |           |
         |           |            |           |           |            |           |           |
     ------------------------DB LAN 10.0.0.0 / 255.255.255.0 ---------------------------------------
                                |     :     |            |
                     <Master>   |..[ VIP ]..|   <Slave>  |
                            \   |           |  /      \  |
                             SQL210      SQL211       SQL212
                                |           |            |
                                |.heartbeat.|            |
                                |           |            |
          ------------SYNC LAN 192.168.1.0 / 255.255.255.0 -----------------------------------------

#-------------------- 初始化安装，VMware 虚机名为 LVS201 --------------------

# 由 WEB248 init 快照，完整克隆(为以后可独立拷贝操作)生成 LVS201
# 此为完成步骤 ″ 制作 VMware 快照 IMG208 U1″ 修正之后的hosts
rm -rf /etc/hosts
echo -ne ″
# Do not remove the following line, or various programs
# that require network functionality will fail.
127.0.0.1               localhost.localdomain localhost

# IP address            # Hostname      # ethN
# ----------------------------------------------
192.168.1.200           HTTPLVS         # eth0:1
192.168.1.201           LVS201          # eth0
10.0.0.201              LVS201          # eth1
192.168.1.202           LVS202          # eth0
10.0.0.202              LVS202          # eth1
# ----------------------------------------------
192.168.1.203           WEB203          # eth0
10.0.0.203              WEB203          # eth1
192.168.1.204           WEB204          # eth0
10.0.0.204              WEB204          # eth1
192.168.1.205           WEB205          # eth0
10.0.0.205              WEB205          # eth1
# ----------------------------------------------
192.168.1.206           IMG206          # eth0
10.0.0.206              IMG206          # eth1
192.168.1.207           IMG207          # eth0
10.0.0.207              IMG207          # eth1
192.168.1.208           IMG208          # eth0
10.0.0.208              IMG208          # eth1
# ----------------------------------------------
10.0.0.209              SQLM-H-S        # eth0:0
192.168.1.209           SQLVRRP         # eth1:0
10.0.0.210              SQL210          # eth0
192.168.1.210           SQL210          # eth1
10.0.0.211              SQL211          # eth0
192.168.1.211           SQL211          # eth1
10.0.0.212              SQL212          # eth0
192.168.1.212           SQL212          # eth1
″ >> /etc/hosts
cat /etc/hosts

# 关闭 SELinux
# more /etc/sysconfig/selinux
SELINUX=disabled
```

```
# 关闭防火墙
# 特别说明: 因为此测试环境是用VMware的桥接功能来同时链接所有网段
# cat /etc/sysconfig/system-config-securitylevel
--disabled
--port=22:tcp

# 实际生产环境中防火墙配置应该为
vi /etc/sysconfig/system-config-securitylevel
--enabled
--trust=eth1
--port=22:tcp
--port=80:tcp
--port=443:tcp
# 查看 iptables
more /etc/sysconfig/iptables

# 修改主机名, IP地址
# vi /etc/sysconfig/network
HOSTNAME=LVS201
# vi /etc/sysconfig/network-scripts/ifcfg-eth0
IPADDR=192.168.1.201
# vi /etc/sysconfig/network-scripts/ifcfg-eth1
BROADCAST=10.0.0.255
IPADDR=10.0.0.201
NETMASK=255.255.255.0
# 删除由 WEB248 init 快照 带来的 ifcfg-eth2; 重启网络服务
rm -rf /etc/sysconfig/network-scripts/ifcfg-eth2
service network restart

# (单CPU,32位系统环境下) 添加 clock=pit nosmp noapic nolapic ; 解决Vmware下linux时间跑快及跑慢的问题
# vi /etc/grub.conf
kernel /vmlinuz-2.6.18-53.el5 ... rhgb quiet clock=pit nosmp noapic nolapic

# 开启VMware客户机与主机(寄主)之间的时间同步
# 启动 vmware-tools 服务, 并设置为默认启动
vmware-guestd --cmd "vmx.set_option synctime 0 1"
chkconfig vmware-tools on

# 查看系统时间,与主机时间对比,确认为同步一致; 关机, 制作快照
date
poweroff
```

```
# 初始化安装，开启服务最小化的情况
# chkconfig --list |grep 3:on
acpid           0:off   1:off   2:off   3:on    4:on    5:on    6:off
network         0:off   1:off   2:on    3:on    4:on    5:on    6:off
sshd            0:off   1:off   2:on    3:on    4:on    5:on    6:off
syslog          0:off   1:off   2:on    3:on    4:on    5:on    6:off
vmware-tools    0:off   1:off   2:on    3:on    4:on    5:on    6:off

# (若需要) 在VMware Workstation中，按如下操作步骤，建立虚拟共享目录
"虚拟机" -> "设置" -> "选项" -> "共享文件夹" ->
"总是启用" -> "添加" -> "下一步" -> 名称: "share" ->
主机文件夹: "E:\shares" -> "下一步" -> "完成"

#--------------------制作 VMware 快照 LVS201 U1 --------------------

# 新建 VMware 虚拟机分组 PhaseII.fkoo; 将 LVS201 虚机添加进分组;
# 将 LVS201 虚机的 "以太网" 和 "以太网 2" 都设置为 "桥接"
# 将 LVS201 虚机的内存设置为 64 MB

# 在VMware Workstation中，按如下操作步骤，建立虚拟共享目录 web 和 img
"虚拟机" -> "设置" -> "选项" -> "共享文件夹" ->
"总是启用" -> "添加" -> "下一步" -> 名称: "web" ->
主机文件夹: "E:\PhaseI.fkoo\web" -> "下一步" -> "完成"

"虚拟机" -> "设置" -> "选项" -> "共享文件夹" ->
"总是启用" -> "添加" -> "下一步" -> 名称: "img" ->
主机文件夹: "E:\PhaseI.fkoo\img" -> "下一步" -> "完成"

# 挂载上软件代码光盘包 "PhaseII.fkoo"
mkdir /mnt/cdrom
mount /dev/cdrom /mnt/cdrom
ls /mnt/cdrom

# 安装openssl
cd /tmp/
tar xvfz /mnt/cdrom/openssl-0.9.8i.tar.tar
tar xvfz /mnt/cdrom/openssl-0.9.8i.tar.tar
cd openssl-0.9.8i
./config
make
make install
cd /usr/local/bin
ln -s /usr/local/ssl/bin/openssl openssl
```

```
cd /tmp/
rm -rf openssl-0.9.8i*

# 安装 zlib
cd /tmp
tar xvfz /mnt/cdrom/zlib-1.2.3.tar.gz
cd  zlib-1.2.3
./configure
make
make install
cd ..
rm -rf   zlib-1.2.3*

# 安装 libpng
cd /tmp
tar -zxvf /mnt/cdrom/libpng-1.2.34.tar.gz
cd libpng-1.2.34
cp scripts/makefile.std makefile
make
make install
cd ..
rm -rf  libpng-1.2.34*

# 建立安装 libjpeg 必须的目录 #
mkdir /usr/local/jpeg
mkdir /usr/local/jpeg/include
mkdir /usr/local/jpeg/lib
mkdir /usr/local/jpeg/bin
mkdir /usr/local/jpeg/man/
mkdir /usr/local/jpeg/man/man1/
# 开始安装 libjpeg
cd /tmp
tar zxvf /mnt/cdrom/jpegsrc.v6b.tar.gz
cd jpeg-6b
# 编译安装，设定安装目录为 /usr/local/jpeg #
./configure --prefix=/usr/local/jpeg --enable-shared --enable-static
make
make install
cd ..
rm -rf jpeg*

# 安装 freetype
cd /tmp
```

```
tar zxvf /mnt/cdrom/freetype-2.3.7.tar.gz
cd freetype-2.3.7
# 编译安装，设定安装目录为 /usr/local/freetype #
./configure --prefix=/usr/local/freetype
make
make install
cd ..
rm -rf freetype-2.3.7*

# 安装 libxml2
cd /tmp
tar xzvf /mnt/cdrom/libxml2-2.7.2.tar.gz
cd libxml2-2.7.2
# 编译安装，设定安装目录为 /usr/local/libxml2 #
./configure -prefix=/usr/local/libxml2
make
make install
cd ..
rm -rf libxml2-2.7.2*

# 安装 libmcrypt
cd /tmp
tar zxvf /mnt/cdrom/libmcrypt-2.5.8.tar.gz
cd libmcrypt-2.5.8
# 编译安装，设定安装目录为 /usr/local/libmcrypt2 #
./configure --prefix=/usr/local/libmcrypt2
make
make install
# install libltdl
cd libltdl
./configure --enable-ltdl-install
make
make install
cd ../..
rm -rf  libmcrypt-2.5.8*

# 安装 fontconfig
cd /tmp
tar zxvf /mnt/cdrom/fontconfig-2.6.0.tar.gz
cd fontconfig-2.6.0
# 编译安装，设定安装目录为 /usr/local/fontconfig；指定 freetype 的实际安装目录 /usr/local/freetype/bin/freetype-config#
./configure --prefix=/usr/local/fontconfig \
--with-freetype-config=/usr/local/freetype/bin/freetype-config
```

```
make
make install
cd ..
rm -rf  fontconfig-2.6.0*

# 安装 gd
cd /tmp
tar jxvf /mnt/cdrom/gd-2.0.36RC1.tar.bz2
cd gd-2.0.36RC1
# 编译安装, 设定安装目录为 /usr/local/gd; 指定 png, jpeg, freetype, zlib, fontconfig 安装路径为实际安装目录. 如前#
./configure -prefix=/usr/local/gd \
--with-png=/usr/local/lib/ \
--with-jpeg=/usr/local/jpeg/ \
--with-freetype=/usr/local/freetype/ \
--with-zlib \
--with-fontconfig=/usr/local/fontconfig
make
make install
cd ..
rm -rf  gd-2.0.36RC1*

#--------------------制作 VMware 快照 LVS201 U2 --------------------

# 将虚拟机内存调整为 256 MB

# 挂载上软件代码光盘包 "PhaseII.fkoo"
mount /dev/cdrom /mnt/cdrom

# 安装mysql5.1.30 稳定版
cd /tmp
tar -zxvf  /mnt/cdrom/mysql-5.1.30-linux-i686-icc-glibc23.tar.gz
groupadd mysql
useradd -g mysql -s /sbin/nologin mysql
mv mysql-5.1.30-linux-i686-icc-glibc23 /usr/local/mysql
cd /usr/local/mysql
chown -R root  .
chown -R mysql data
chgrp -R mysql .
scripts/mysql_install_db --user=mysql
cp /usr/local/mysql/support-files/mysql.server /etc/rc.d/init.d/mysql
#cp /usr/local/mysql/support-files/my-innodb-heavy-4G.cnf /etc/my.cnf
cp /usr/local/mysql/support-files/my-huge.cnf /etc/my.cnf
chmod +x /etc/rc.d/init.d/mysql
```

```
chkconfig --del mysql
chkconfig --add mysql
chkconfig mysql on
# /usr/local/mysql/bin/mysqld_safe --user=mysql &
service mysql start
/usr/local/mysql/bin/mysqladmin -u root password 'rvdgi,jl'

# 给 mysql 命令增加系统环境变量 /usr/local/mysql/bin
# 同时给后面的 php-fpm 命令增加系统环境变量 /usr/local/php-fcgi/sbin/
# vi /etc/profile
export PATH="$PATH:/usr/local/mysql/bin:/usr/local/php-fcgi/sbin/"
# 重新登录以生效
su -

# 安装libunwind (64位需要安装,32位不用)
cd /tmp/
tar zxvf /mnt/cdrom/libunwind-snap-070410.tar.gz
cd libunwind-snap-070410/
./configure
make && make install
cd ..

# 安装TCMalloc（Thread-Caching Malloc）,提高MySQL服务器在高并发情况下的性能.
cd /tmp/
tar zxvf /mnt/cdrom/google-perftools-1.0rc2.tar.gz
cd google-perftools-1.0rc2/
./configure
make && make install
cd ..
rm -rf google-perftools-1.0rc2

# 修改MySQL启动脚本（根据你的MySQL安装位置而定）：
vi /usr/local/mysql/bin/mysqld_safe
# 在# executing mysqld_safe的下一行，加上：
export LD_PRELOAD=/usr/local/lib/libtcmalloc.so
# 保存后退出，然后重启MySQL服务器。
service mysql restart

# 使用 lsof 命令查看tcmalloc是否起效：
/usr/sbin/lsof -n | grep tcmalloc
如果发现以下信息，说明tcmalloc已经起效：
mysqld    10847   mysql  mem       REG        8,5  1203756   20484960 /usr/local/lib/libtcmalloc.so.0.0.0
```

```
# 关闭 mysql
service mysql stop
chkconfig mysql off

#--------------------制作 VMware 快照 LVS201 U3 --------------------

# 挂载上软件代码光盘包 "PhaseII.fkoo"
mount /dev/cdrom /mnt/cdrom

## 安装 apache #
cd /tmp
tar jxvf /mnt/cdrom/httpd-2.2.11.tar.bz2
cd httpd-2.2.11
./configure --prefix=/usr/local/apache \
--with-mpm=worker \
--enable-rewrite \
--enable-so \
--enable-ssl --with-ssl=/usr/local/ssl/ \
--enable-cgi \
--enable-cache \
--enable-disk-cache \
--enable-mem-cache \
--enable-file-cache \
--enable-expires \
--enable-proxy \
--enable-proxy-http \
--disable-ipv6 \
--sysconfdir=/etc/httpd
make
make install

# 加载 mod_rewrit 模块 ###
cd modules/mappers/
/usr/local/apache/bin/apxs -c mod_rewrite.c -lgdbm
gcc -shared -o mod_rewrite.so mod_rewrite.o -lgdbm
/usr/local/apache/bin/apxs -i -A -n mod_rewrite mod_rewrite.so

# 增加 Mod_vhost_alias.so 模块
/usr/local/apache/bin/apxs -c mod_vhost_alias.c
gcc -shared -o mod_vhost_alias.so mod_vhost_alias.o
/usr/local/apache/bin/apxs -i -A -n vhost_alias mod_vhost_alias.so

# 删除安装源文件 ###
```

```
cd ../../..
rm -rf httpd-2.2.11*
# 从主机共享目录复制 httpd 服务启动脚本到客户机的服务启动目录，设置为可执行 ###
cp /mnt/cdrom/httpd.apache /etc/rc.d/init.d/httpd
chmod 755 /etc/rc.d/init.d/httpd
# 添加 httpd 服务，设定默认为开启 ###
chkconfig --add httpd
chkconfig httpd on
service httpd start
# 测试 httpd 服务是否安装正常 ###
/usr/local/apache/bin/httpd -t
# 查看 apache 加载的模块；版本 ###
/usr/local/apache/bin/httpd -l
/usr/local/apache/bin/httpd -v

# 安装日志回滚
cd /tmp/
tar xvf /mnt/cdrom/cronolog-1.6.2.tar.tar
cd cronolog-1.6.2
./configure
make && make install
cd ..
rm -rf cronolog-1.6.2

# 注销掉原有的日志格式；修改日志格式
vi /etc/httpd/httpd.conf
    # CustomLog "logs/access_log" common
    CustomLog "|/usr/local/sbin/cronolog /usr/local/apache/logs/access_log.%Y%m%d%H" combined

# 暂停 httpd 服务，删掉旧日志，重启 httpd
# 强调：不删除 access_log 原文件，可能不会出现新日志格式；即便是将 access_log 改名
service httpd stop
rm -rf /usr/local/apache/logs/access_log
service httpd start
ls /usr/local/apache/logs/access_log*

# 安装pcre库（支持 lighttpd 或 nginx 的 rewrite 模块）
cd /tmp
tar zxvf /mnt/cdrom/pcre-7.7.tar.gz
cd pcre-7.7
./configure
make && make install
cd ..
```

```
rm -rf  pcre-7.7*

#--------------------制作 VMware 快照 LVS201 U4 --------------------

# 挂载上软件代码光盘包 "PhaseII.fkoo"
mount /dev/cdrom /mnt/cdrom

# 安装 php
cd /tmp
tar -jxvf /mnt/cdrom/php-5.2.8.tar.bz2
cd php-5.2.8
# 编译安装, 设定安装目录为 /usr/local/php; 指定各支持包的安装路径为实际安装目录. 如前
# 注意: piranha-gui 依赖安装了 apache 和 php (apxs2 的 apache2handler 模式) 运行环境
# 编译安装 apache 必须有  --with-apxs2, 否则 /usr/local/apache/modules/ 目录下没有 libphp5.so
./configure \
--prefix=/usr/local/php \
--with-mysql=/usr/local/mysql \
--with-pdo-mysql=/usr/local/mysql/bin/mysql_config \
--with-config-file-path=/usr/local/php/etc \
--with-zlib \
--with-zlib-dir \
--with-png-dir=/usr/local/lib \
--with-jpeg-dir=/usr/local/jpeg \
--with-freetype-dir=/usr/local/freetype \
--with-gd=/usr/local/gd \
--with-ttf \
--enable-gd-native-ttf \
--enable-gd-jis-conv \
--with-libxml-dir=/usr/local/libxml2 \
--with-mcrypt=/usr/local/libmcrypt2 \
 --with-iconv \
--with-openssl \
--enable-mbstring \
--enable-pdo \
--without-pdo-sqlite \
--without-sqlite \
--with-curl \
--with-curlwrappers \
--enable-xml \
--with-pear \
--enable-magic-quotes \
--enable-ftp \
--with-bz2 \
--enable-sysvsem \
--enable-exif \
--with-pcre-dir \
--with-apxs2=/usr/local/apache/bin/apxs \
--disable-ipv6

make
make install
cp php.ini-dist /usr/local/php/etc/php.ini
cp /mnt/cdrom/phpinfo.php /mnt/hgfs/web/
cd ..
rm -rf  php-5.2.8*

# 编辑 apache 配置文档, 支持 php
vi /etc/httpd/httpd.conf
        # 在httpd.conf 中添加 worker 模块参数
        # ServerLimit乘以ThreadsPerChild必须大于等于MaxClients。而且MaxClients必须是ThreadsPerChild的整数倍。
        # 实例为一个每秒并发量在3000－4000左右的网站的设置：ServerLimit乘以ThreadsPerChild正好等于MaxClients
<IfModule worker.c>
        StartServers 10
        MaxClients 4096
        ServerLimit 128
        MinSpareThreads 32
        MaxSpareThreads 64
        ThreadLimit 1024
        ThreadsPerChild 32
        MaxRequestsPerChild 0
</IfModule>

        # 修改 httpd 主目录;  -Indexes 不列出目录索引
DocumentRoot "/mnt/hgfs/web"
<Directory "/mnt/hgfs/web">
    Options -Indexes FollowSymLinks
        # 增加 php 文件类型
    AddType application/x-httpd-php .php
        # 增加 php 默认首页 index.php
    DirectoryIndex index.php index.html index.htm

# 安装 memcached 服务器端之前, 先要安装 libevent 支持
cd /tmp/
tar vxzf /mnt/cdrom/libevent-1.4.9-stable.tar.gz
cd libevent-1.4.9-stable/
./configure
```

```
make
make install
#建立一个符号连接：
ln -s /usr/local/lib/libevent-1.4.so.2 /usr/lib
cd ..
rm -rf libevent-1.4.9-stable

# 安装 memcached 服务器端
cd /tmp/
tar vxzf /mnt/cdrom/memcached-1.2.6.tar.gz
cd memcached-1.2.6/
./configure --prefix=/usr/local/memcached \
--with-libevent=/usr
make
make install
cd ..
rm -rf memcached-1.2.6/

# memcached 启动命令
/usr/local/memcached/bin/memcached -l 10.0.0.201 -d -p 62880 -u nobody -m 2
# 表示用 daemon 的方式启动 memcached，监听在 10.0.0.201 的 62880 端口上，运行用户为 nobody，为其分配 2MB 的内存。

# 查看 memcached 选项
# /usr/local/memcached/bin/memcached -h
        -t <num> number of threads to use, default 4

# 添加 memcached 为服务
cp /mnt/cdrom/memcached.init /etc/rc.d/init.d/memcached
chmod 755 /etc/rc.d/init.d/memcached
# 编辑 memcached 启动命令文件
vi /etc/rc.d/init.d/memcached
        PORT1=62880
        USER=nobody
        MAXCONN=1024
CACHESIZE=2
IP_ADDR=10.0.0.201
        OPTIONS="-t 8"
# 说明: 添加了绑定IP的选项 -l $IP_ADDR
  daemon $MEMDAEMON -d -p $PORT1 -u $USER -m $CACHESIZE -c $MAXCONN -l $IP_ADDR $OPTIONS

chkconfig  --add memcached
chkconfig memcached  on
chkconfig  --list | grep mem
```

```
service memcached restart
ps aux | grep memcached

# 安装memcache php客户端
cd /tmp/
tar xvfz /mnt/cdrom/memcache-2.2.4.tgz
cd memcache-2.2.4
/usr/local/php/bin/phpize
./configure \
--enable-memcache \
--with-php-config=/usr/local/php/bin/php-config \
--with-zlib-dir
make && make install
# Installing shared extensions:     /usr/local/php/lib/php/extensions/no-debug-zts-20060613/
cd ..
rm -rf memcache-2.2.4*

# 配置 php.ini 支持扩展
# vi /usr/local/php/etc/php.ini
extension_dir = "/usr/local/php/lib/php/extensions/no-debug-zts-20060613/"
extension=memcache.so

# 安装 eAccelerator PHP加速器
cd /tmp/
tar -xvf  /mnt/cdrom/eaccelerator-0.9.5.3.tar.tar
cd eaccelerator-0.9.5.3/
/usr/local/php/bin/phpize
./configure --enable-eaccelerator=shared \
--with-php-config=/usr/local/php/bin/php-config
make
make install
        # Installing shared extensions:     /usr/local/php/lib/php/extensions/no-debug-zts-20060613/
cd ..
rm -rf eaccelerator-0.9.5.3/
mkdir /tmp/eaccelerator && chmod 777 /tmp/eaccelerator && touch /var/log/eaccelerator_log
# 编辑php.ini , 将 eAccelerator 作为 PHP Extension 添加
# vi /usr/local/php/etc/php.ini
# 加上:
extension="eaccelerator.so"
eaccelerator.shm_size="16"
eaccelerator.cache_dir="/tmp/eaccelerator"
eaccelerator.enable="1"
eaccelerator.optimizer="1"
eaccelerator.check_mtime="1"
eaccelerator.debug="0"
eaccelerator.log_file = "/var/log/eaccelerator_log "
eaccelerator.filter=""
eaccelerator.shm_max="0"
eaccelerator.shm_ttl="0"
eaccelerator.shm_prune_period="0"
eaccelerator.shm_only="0"
eaccelerator.compress="1"
eaccelerator.compress_level="9"

# 安装SQL Relay 前，先安装Rudiments:
cd /tmp/
tar vxzf  /mnt/cdrom/rudiments-0.31.tar.gz
cd rudiments-0.31
./configure --prefix=/usr/local/rudiments
make
make install
cd ..
rm -rf rudiments-0.31
# 安装SQL Relay:
cd /tmp/
tar vxzf /mnt/cdrom/sqlrelay-0.39.4.tar.gz
cd sqlrelay-0.39.4
./configure --prefix=/usr/local/sqlrelay --with-rudiments-prefix=/usr/local/rudiments \
--with-mysql-prefix=/usr/local/mysql \
--with-php-prefix=/usr/local/php
make
make install
cd ..
rm -rf sqlrelay-0.39.4

# 修改 php.ini 文件
# vi /usr/local/php-fcgi/etc/php.ini
extension_dir = "/usr/local/php/lib/php/extensions/no-debug-zts-20060613/"
extension=sql_relay.so
# 修改 SQL Relay 的配置文件
cp /usr/local/sqlrelay/etc/sqlrelay.conf.example /usr/local/sqlrelay/etc/sqlrelay.conf

# 修改最大打开文件数
echo -ne "
* soft nofile 65536
* hard nofile 65536
```

```
" >>/etc/security/limits.conf

cat /proc/sys/fs/file-max
cat /proc/sys/fs/file-nr

# tcpip调优
echo -ne "
net.ipv4.ip_local_port_range = 1024 65536
net.core.rmem_max = 16777216
net.core.wmem_max = 16777216
net.ipv4.tcp_rmem = 4096 87380 16777216
net.ipv4.tcp_wmem = 4096 65536 16777216
net.ipv4.tcp_fin_timeout = 3
net.ipv4.tcp_tw_recycle = 1
net.core.netdev_max_backlog = 30000
net.ipv4.tcp_no_metrics_save = 1
net.core.somaxconn = 262144
net.ipv4.tcp_syncookies = 0
net.ipv4.tcp_max_orphans = 262144
net.ipv4.tcp_max_syn_backlog = 262144
net.ipv4.tcp_synack_retries = 2
net.ipv4.tcp_syn_retries = 2
fs.file-max = 65536
" >> /etc/sysctl.conf

# vi /etc/sysctl.conf
## net.ipv4.tcp_syncookies = 1

# sysctl -p /etc/sysctl.conf
#--------------------制作 VMware 快照 LVS201 U5 --------------------

# 挂载上软件代码光盘包 "PhaseII.fkoo"
mount /dev/cdrom /mnt/cdrom

# 以 LVS / DR 方式搭建负载均衡

# 安装 ipvsadm( LVS 管理软件), 编译 piranha( LVS 功能软件) 的 RPM 源码包
rpm -ivh /mnt/cdrom/ipvsadm-1.24-11.i386.rpm
rpm -i /mnt/cdrom/piranha-0.8.4-9.3.el5.0.1.src.rpm
cd /usr/src/redhat/SPECS
rpmbuild -bp piranha.spec
cd /usr/src/redhat/BUILD/piranha
```

```
# 备份 Makefile 源码; 从软件包中复制 Makefile 修改版
mv /usr/src/redhat/BUILD/piranha/Makefile /usr/src/redhat/BUILD/piranha/Makefile.bak
cp /mnt/cdrom/Makefile.piranha-gui /usr/src/redhat/BUILD/piranha/Makefile

{
# 或者, 定制修改 Makefile
vi Makefile
        # 修正默认 $(LIBDIR) 为 /usr/local
        # 说明: 因前面安装php后 libphp5.so 模块文件所在位置为 /usr/local/apache/modules/
DEFAULT_LIBDIR = /usr/local
        # 修正 libphp5.so 的 modules 路径; httpd 命令路径; 注意 ln 前面的缩进是 Tab 键, 而不能是空格
                ln -sf $(LIBDIR)/apache/modules $(HADIR)/modules
                ln -sf /usr/local/apache/bin/httpd $(SBIN)/piranha_gui
}

make && make install
cd ../..
rm -rf BUILD/piranha
rm -rf SPECS/piranha.spec
rm -rf SOURCES/piranha*

# 备份 piranha-gui httpd.conf 源码; 从软件包中复制 piranha-gui httpd.conf 修改版
mv /etc/sysconfig/ha/conf/httpd.conf /etc/sysconfig/ha/conf/httpd.conf.bak
cp /mnt/cdrom/httpd.conf.piranha-gui /etc/sysconfig/ha/conf/httpd.conf

{
# 或者, 定制修改 piranha-gui httpd.conf
vi /etc/sysconfig/ha/conf/httpd.conf
        # 注释掉下面参数, 否则启动
#MinSpareServers 1
#MaxSpareServers 1
#MaxClients 4

        # 删除所有 LoadModule 项, 仅保留 php5_module 项即可
LoadModule php5_module        modules/libphp5.so
        # 修改监听端口号 3636 为自定义的 6666; 修改 用户名 / 用户组 为上述自定义的 fkoo / fkoogroup
Listen 6666
User fkoo
Group fkoogroup
        # 修改 Options 为 -Indexes 从而关闭 web 目录的目录树浏览
        # 设定此目录允许访问IP段为: 192.168.1.0/255.255.255.0
<Directory /etc/sysconfig/ha/web>
```

```
    # Allow from all
    Deny from all
    Allow from 192.168.1.0/255.255.255.0
</Directory>
        # 设定此目录允许访问IP段为: 192.168.1.0/255.255.255.0
<Directory /etc/sysconfig/ha/web/secure>
    AllowOverride All
    Order deny,allow
    # Allow from all
    Deny from all
    Allow from 192.168.1.0/255.255.255.0
        # 修改 <Limit GET> 下 require user piranha 为 上述预置设定的用户名 fkoo
    <Limit GET>
        require user fkoo
}

# 备份 piranha.passwd 源码; 从软件包中复制 piranha-passwd.piranha-gui 修改版
mv /usr/sbin/piranha-passwd /usr/sbin/piranha-passwd.bak
cp /mnt/cdrom/piranha-passwd.piranha-gui /usr/sbin/piranha-passwd

{
# 或者, 定制修改 piranha.passwd
vi /usr/sbin/piranha-passwd
        # 修改 piranha-gui 的默认登录用户名 piranha 为预置设定的用户名 fkoo
        # 同时须修正 htpasswd 命令的实际路径为 /usr/local/apache/bin/htpasswd
        # 同时修正 piranha.passwd 所属的 用户名.用户组 为预置设定的 fkoo.fkoogroup
        /usr/local/apache/bin/htpasswd -b $DEST/piranha.passwd fkoo "$password"
        /usr/local/apache/bin/htpasswd -c -b $DEST/piranha.passwd fkoo "$password"
chown fkoo.fkoogroup $DEST/piranha.passwd
}

# 备份 passwd 源码; 从软件包中复制 passwd.piranha-gui 修改版
# 备份 shadow 源码; 从软件包中复制 shadow.piranha-gui 修改版
# 备份 group 源码; 从软件包中复制 group.piranha-gui 修改版
mv /etc/passwd /etc/passwd.bak
cp /mnt/cdrom/passwd.piranha-gui /etc/passwd
mv /etc/shadow /etc/shadow.bak
cp /mnt/cdrom/shadow.piranha-gui /etc/shadow
mv /etc/group /etc/group.bak
cp /mnt/cdrom/group.piranha-gui /etc/group

{
# 或者, 定制修改 piranha.passwd
# 将默认的 piranha-gui 服务登录用户名由 piranha 改为预置设定的 fkoo , 组名改为 fkoogroup
# vi /etc/passwd
fkoo:x:60:60::/etc/sysconfig/ha:/dev/null
# vi /etc/shadow
fkoo:!!:14198::::::
# vi /etc/group
fkoogroup:x:60:
}

# 安装成功则可以正常启动 piranha-gui 服务, 并设置密码
        # 说明: piranha-gui 服务不依赖 apache 服务, 无需同时开启 httpd 服务
service piranha-gui start

# 修改预置设定用户 fkoo 的登录密码, 需要重复确认输入
# 说明: 设定密码成功后显示 Updating password for user fkoo
# piranha-passwd
Adding password for user fkoo
# 说明: 查看设定的密码, 显示为已经过加密的格式
# cat /etc/sysconfig/ha/conf/piranha.passwd
fkoo:qeVL81nMXL0dI

# 编辑 LVS 的配置文件
# vi /etc/sysconfig/ha/lvs.cf
serial_no = 1
primary = 192.168.1.201
primary_private = 10.0.0.201
service = lvs
backup_active = 1
backup = 192.168.1.202
backup_private = 10.0.0.202
heartbeat = 1
heartbeat_port = 539
keepalive = 6
deadtime = 18
network = direct
debug_level = NONE
monitor_links = 1
virtual HTTPVS {
     active = 1
     address = 192.168.1.200 eth0:1
     vip_nmask = 255.255.255.255
     port = 80
     send = "GET / HTTP/1.0\r\n\r\n"
     expect = "HTTP"
     use_regex = 0
     load_monitor = none
     scheduler = wrr
     protocol = tcp
     timeout = 6
     reentry = 15
     quiesce_server = 0
     server WEB203 {
         address = 192.168.1.203
         active = 1
         weight = 1
     }
     server WEB204 {
         address = 192.168.1.204
         active = 1
         weight = 1
     }
     server WEB205 {
         address = 192.168.1.205
         active = 1
         weight = 1
     }
}

# 重载 pulse 服务, 使 lvs.cf 配置生效;
service pulse reload

# 查看负载均衡状态; 保存负载均衡表, 并显示
ipvsadm -Ln
service ipvsadm save
cat /etc/sysconfig/ipvsadm

# 添加 pulse, piranha-gui 服务
# 设置 pulse 服务为默认开机启动
# 启动 pulse, piranha-gui 服务
        # 说明: piranha-gui 服务提供 LVS 的 WEB 管理界面; pulse 为 LVS 的后台运行程序
        # 出于安全考虑, 设置 piranha-gui 服务为默认开机不启动 (需要通过 WEB 设置 LVS 时可手工启动服务)
chkconfig --add piranha-gui
chkconfig --add pulse
chkconfig pulse on
chkconfig piranha-gui on
service pulse start
service piranha-gui restart

# 检查 ip_forward 路由转发功能是否开启; 0 表示关闭, 1 表示开启
# cat /proc/sys/net/ipv4/ip_forward
1
# 若 ip_forward 为 0 关闭; 修改并激活设置为开启
# vi /etc/sysctl.conf
net.ipv4.ip_forward = 1
# 使 sysctl.conf 配置立即激活生效
sysctl -p
或者
echo "1"> /proc/sys/net/ipv4/ip_forward

#--------------------制作 VMware 快照 LVS201 U6 --------------------

# 编辑 LVS 的配置文件, 添加 persistent = 300
# 说明: 设置 persistent (持久) 参数, 让同一个来源始终连同一台 real server, 以防止连到不同的 real server 而造成 Session 丢失.
# 注意: 必须同时设置 quiesce_server = 0, 以使 real server 宕机时, 从列表中删除. persistent 链接会转发给正常的 real server.
# vi /etc/sysconfig/ha/lvs.cf
     persistent = 300

# ipvsadm 实用命令
ipvsadm -Ln                                             \\ 查看LVS的连接情况
ipvsadm -Ln --persistent-conn           \\ 查看持久链接
ipvsadm -Ln --rate                              \\ 查看LVS的吞吐量情况
ipvsadm -Ln --stats                             \\ 查看LVS的统计信息
watch ipvsadm -Ln                               \\ 实时查看LVS连接状态变化

#--------------------制作 VMware 快照 LVS201 U7 --------------------

# 由 VMware 快照 LVS201 U7 克隆链接生成 LVS202

# 修改 hostname 和 IP
# vi /etc/sysconfig/network
HOSTNAME=LVS202
# vi /etc/sysconfig/network-scripts/ifcfg-eth0
IPADDR=192.168.1.202
# vi /etc/sysconfig/network-scripts/ifcfg-eth1
IPADDR=10.0.0.202

# 修改 LVS 的配置文件
# vi /etc/sysconfig/ha/lvs.cf
primary = 192.168.1.202
primary_private = 10.0.0.202
backup = 192.168.1.201
backup_private = 10.0.0.201

# 修改 memcached 服务启动脚本
# vi /etc/rc.d/init.d/memcached
IP_ADDR=10.0.0.202

# 调试命令
# 在 LVS active 上切换到 standby
service pulse stop
# 查看是否切换成功
watch ifconfig

#--------------------制作 VMware 快照 LVS202 U1 --------------------

# 由 VMware 快照 LVS201 U5 克隆链接生成 WEB203

# 挂载上软件代码光盘包 "PhaseII.fkoo"
mount /dev/cdrom /mnt/cdrom

# 修改 hostname 和 IP
# vi /etc/sysconfig/network
HOSTNAME=WEB203
# vi /etc/sysconfig/network-scripts/ifcfg-eth0
IPADDR=192.168.1.203
# vi /etc/sysconfig/network-scripts/ifcfg-eth1
IPADDR=10.0.0.203

# 关闭 ip forwarding 路由
# vi /etc/sysctl.conf
net.ipv4.ip_forward = 0

# 创建哑设备 dummy0, 并将 LVS VIP 192.168.1.200 绑定其上
# vi /etc/sysconfig/network-scripts/ifcfg-dummy0
DEVICE=dummy0
BROADCAST=192.168.1.200
IPADDR=192.168.1.200
NETMASK=255.255.255.255
ONBOOT=yes
```

```
# 添加到 LVS VIP 192.168.1.200 的路由到哑设备 dummy0
# vi /etc/sysconfig/network-scripts/route-dummy0
192.168.1.200/32 via 0.0.0.0 dev dummy0

# 重启 network 服务，使 dummy0 设备生效
service network restart

{
# 或者用命令行配置（ 重启后失效 ）
ifconfig dummy0 192.168.1.200 broadcast 192.168.1.200 netmask 255.255.255.255 up
route add -host 192.168.1.200 dev dummy0
}

# 关闭 Realserver 的被动 ARP广播响应，使之生效
echo -ne ″
net.ipv4.conf.dummy0.arp_ignore = 1
net.ipv4.conf.dummy0.arp_announce = 2
net.ipv4.conf.all.arp_ignore = 1
net.ipv4.conf.all.arp_announce = 2
″ >> /etc/sysctl.conf
sysctl -p
tail /etc/sysctl.conf

{
# 或者用命令行配置（ 重启后失效 ）
echo 1 > /proc/sys/net/ipv4/conf/dummy0/arp_ignore
echo 2 > /proc/sys/net/ipv4/conf/dummy0/arp_announce
echo 1 > /proc/sys/net/ipv4/conf/all/arp_ignore
echo 2 > /proc/sys/net/ipv4/conf/all/arp_announce
}

# 安装配置 arptables_jf  直接路由服务
# 说明：使得 Realserver 将 LVS DR 转发来的请求，以其自身 IP 返回给客户端
rpm -ivh /mnt/cdrom/arptables_jf-0.0.8-13.fc10.i386.rpm
chkconfig arptables_jf on

# 清空所有的链；丢弃目的地址为 VIP 192.168.1.200 的包
# 将返回给 VIP 192.168.1.200 的数据包源地址改为本 Realserver 的 IP，直接返回给客户端
# 保存 arptables
arptables --flush
arptables -A IN -d 192.168.1.200 -j DROP
arptables -A OUT  -d 192.168.1.200 -j mangle --mangle-ip-s 192.168.1.203
service arptables_jf save
```

```
# 列出当前活动的 arptables; 显示已保存的 arptables 配置文件
arptables --list
cat /etc/sysconfig/arptables

# 在 apache 的主目录下定制测试页面
#   vi /mnt/hgfs/web/WEB203.html
<html>
 <head>
  <title> WEB203 </title>
  <meta http-equiv="refresh" content="10">
 </head>
 <body>
WEB203
 </body>
</html>

# 给 apache 添加本机的默认首页
# vi /etc/httpd/httpd.conf
    DirectoryIndex index.php index.html index.htm WEB203.html

# 修改 memcached 服务启动脚本
# vi /etc/rc.d/init.d/memcached
IP_ADDR=10.0.0.203

#--------------------制作 VMware 快照 WEB203 U1 --------------------

# 由 VMware 快照 WEB203 U1 克隆链接生成 WEB204

# 修改 hostname 和 IP
# vi /etc/sysconfig/network
HOSTNAME=WEB204
# vi /etc/sysconfig/network-scripts/ifcfg-eth0
IPADDR=192.168.1.204
# vi /etc/sysconfig/network-scripts/ifcfg-eth1
IPADDR=10.0.0.204

# 清空所有的链; 丢弃目的地址为 VIP 192.168.1.200 的包
# 将返回给 VIP 192.168.1.200 的数据包源地址改为本 Realserver 的 IP, 直接返回给客户端
# 保存 arptables
arptables --flush
arptables -A IN -d 192.168.1.200 -j DROP
arptables -A OUT  -d 192.168.1.200 -j mangle --mangle-ip-s 192.168.1.204
```

```
service arptables_jf save

# 在 apache 的主目录下定制测试页面
#   vi /mnt/hgfs/web/WEB204.html
<html>
 <head>
  <title> WEB204 </title>
  <meta http-equiv="refresh" content="10">
 </head>
 <body>
WEB204
 </body>
</html>

# 给 apache 添加本机的默认首页
# vi /etc/httpd/httpd.conf
    DirectoryIndex index.php index.html index.htm WEB204.html

# 修改 memcached 服务启动脚本
# vi /etc/rc.d/init.d/memcached
IP_ADDR=10.0.0.204

#--------------------制作 VMware 快照 WEB204 U1 --------------------

# 与 VMware 快照 WEB204 U1 同理的操作, 生成 WEB205

# 测试注意: 在有线网卡上测试通过
# 因为 VMware 的无线网卡 BUG, 无线网卡做桥接不轮询转发 LVS

#--------------------制作 VMware 快照 WEB205 U1 --------------------

# 由 VMware 快照 LVS201 U5 克隆链接生成 SQL210

# 修改 hostname 和 IP
# vi /etc/sysconfig/network
HOSTNAME=SQL210
# vi /etc/sysconfig/network-scripts/ifcfg-eth0
BROADCAST=10.0.0.255
IPADDR=10.0.0.210
NETWORK=10.0.0.0

# 本测试环境是以同一个物理环境的网卡桥接所有网段
# 仅仅是为了远程控制方便, 设置 eth1 与调试客户端在同一个 IP 子网
# vi /etc/sysconfig/network-scripts/ifcfg-eth1
BROADCAST=192.168.1.255
IPADDR=192.168.1.210
NETWORK=192.168.1.0

# 修改 memcached 服务启动脚本
# vi /etc/rc.d/init.d/memcached
IP_ADDR=10.0.0.210

# 打开 mysql
service mysql start
chkconfig mysql on

# 建立数据库复制帐号 fkoocopy 并允许来自 192.168.1.211 ( 同步复制的对方) 的IP
# 建立 mon 监控数据库服务的帐号 fkoo_monitor 并允许来自192.168.1.211 ( 同步复制的对方) 的IP
# 仅仅为了测试方便, 授权来自客户端网段的 root 用户有完全权限
mysql -p
进入 mysql>
use mysql
grant replication slave on *.* to 'fkoocopy'@'192.168.1.211' identified by 'fkoopasswd';
grant select on *.* to 'fkoo_monitor'@'192.168.1.211' identified by 'FkooMonitor';
grant all privileges on *.* to 'root'@'192.168.1.%' identified by 'rvdgi,jl';
# 存档: 删除用户的命令
# delete from user where user='fkoocopy';
# delete from user where user='fkoo_monitor';
# 为安全考虑, 删除默认生成的不用的帐号和权限
delete from user where user='';
delete from user where user='root' and host='%';
delete from user where user='root' and host='127.0.0.1';
delete from user where user='root' and host='LVS201';

use mysql
flush privileges;
select * from user;
quit;

# 配置 mysql 同步
        # 仅仅为了测试方便, 关闭 bind-address 项和 skip-networking; 生产环境下需要开启
        # 本案中的 M-ha-S-Slaves 结构, 实际上同时只有1个Active, 不是多master结构.
        # 说明: auto-increment-increment 和 auto-increment-offset是用于多主 ( multi-master ) 数据库的复制.
                # 能够让多个主服务器产生不同的字增值,从而不会产生冲突.auto-increment-increment 选项的值必须大于服务器的总
数,并且每个服务器的值必须唯一.
```

```
cp /etc/my.cnf /etc/my.cnf.bak
vi /etc/my.cnf

#skip-networking
# bind-address    = 10.0.0.210
log-bin=SQL210-bin
server-id       = 1

binlog-do-db=fkoodb
binlog-ignore-db = mysql
binlog-ignore-db = test
#auto-increment-increment = 20
#auto-increment-offset = 1

replicate-same-server-id = 0
master-host=192.168.1.211
master-user=fkoocopy
master-password=fkoopasswd
master-port=3306
master-connect-retry=60
report-host=SQL210
replicate-do-db=fkoodb
log-slave-updates
expire_logs_days = 10
max_binlog_size = 500M

service mysql restart

# 修正同步参数的步骤
# 1.  master 上:
show master status;
# 2.  slave 上:
STOP SLAVE;
CHANGE MASTER TO
  MASTER_HOST='192.168.1.211',
  MASTER_USER='fkoocopy',
  MASTER_PASSWORD='fkoopasswd',
  MASTER_PORT=3306,
  MASTER_LOG_FILE='SQL211-bin.000003',
  MASTER_LOG_POS=106,
  MASTER_CONNECT_RETRY=60;
SLAVE START;
show slave status\G;
```

```
# 重置同步日志; mysql 会删除 *-bin.00000* ; 从 *-bin.000001重新开始记录;
# 测试: 此时不能 FLUSH TABLES WITH READ LOCK;  锁定表, 否则 Master_Log_File 不更新
STOP SLAVE;
RESET MASTER;
RESET SLAVE;
SLAVE START;
show master status;
show slave status\G;

# 已建立好的Replication, show slave status\G; 时,在master和slave上应该显示:
mysql> show slave status\G;
             Slave_IO_Running: Yes
             Slave_SQL_Running: Yes

# 关闭 ip_forward 路由
# vi /etc/sysctl.conf
net.ipv4.ip_forward = 0

#--------------------制作 VMware 快照 SQL210 U1 --------------------

# 由 VMware 快照 LVS201 U5 克隆链接生成 SQL211

# 修改 hostname 和 IP
# vi /etc/sysconfig/network
HOSTNAME=SQL211
# vi /etc/sysconfig/network-scripts/ifcfg-eth0
BROADCAST=10.0.0.255
IPADDR=10.0.0.211
NETWORK=10.0.0.0

# 本测试环境是以同一个物理环境的网卡桥接所有网段
# 仅仅是为了远程控制方便, 设置 eth1 与调试客户端在同一个 IP 子网
# vi /etc/sysconfig/network-scripts/ifcfg-eth1
BROADCAST=192.168.1.255
IPADDR=192.168.1.211
NETWORK=192.168.1.0

# 修改 memcached 服务启动脚本
# vi /etc/rc.d/init.d/memcached
IP_ADDR=10.0.0.211

# 打开 mysql
```

```
service mysql start
chkconfig mysql on

# 建立数据库复制帐号 fkoocopy 并允许来自 192.168.1.210 ( 同步复制的对方) 的IP
# 建立 mon 监控数据库服务的帐号 fkoo_monitor 并允许来自192.168.1.210 ( 同步复制的对方) 的IP
# 仅仅为了测试方便, 授权来自客户端网段的 root 用户有完全权限
mysql -p
进入 mysql>
use mysql
grant replication slave on *.* to 'fkoocopy'@'192.168.1.210' identified by 'fkoopasswd';
grant select on *.* to 'fkoo_monitor'@'192.168.1.210' identified by 'FkooMonitor';
grant all privileges on *.* to 'root'@'192.168.1.%' identified by 'rvdgi,jl';
# 存档: 删除用户的命令
# delete from user where user='fkoocopy';
# delete from user where user='fkoo_monitor';
# 为安全考虑, 删除默认生成的不用的帐号和权限
delete from user where user='';
delete from user where user='root' and host='%';
delete from user where user='root' and host='127.0.0.1';
delete from user where user='root' and host='LVS201';

use mysql
flush privileges;
select * from user;
quit;

# 配置 mysql 同步
        # 仅仅为了测试方便, 关闭 bind-address 项和 skip-networking; 生产环境下需要开启
        # 本案中的 M-ha-S-Slaves 结构, 实际上同时只有1个Active, 不是多master结构.
        # 说明: auto-increment-increment 和 auto-increment-offset是用于多主 ( multi-master ) 数据库的复制.
                # 能够让多个主服务器产生不同的字增值,从而不会产生冲突.auto-increment-increment 选项的值必须大于服务器的总
数,并且每个服务器的值必须唯一.

cp /etc/my.cnf /etc/my.cnf.bak
vi /etc/my.cnf

#skip-networking
# bind-address    = 10.0.0.211
log-bin=SQL211-bin
server-id       = 2

binlog-do-db=fkoodb
binlog-ignore-db = mysql
```

```
binlog-ignore-db = test
#auto-increment-increment = 20
#auto-increment-offset = 2

replicate-same-server-id = 0
master-host=192.168.1.210
master-user=fkoocopy
master-password=fkoopasswd
master-port=3306
master-connect-retry=60
report-host=SQL211
replicate-do-db=fkoodb
log-slave-updates
expire_logs_days = 10
max_binlog_size = 500M

service mysql restart

# 修正同步参数的步骤
# 1.  master 上:
show master status;
# 2.  slave 上:
STOP SLAVE;
CHANGE MASTER TO
  MASTER_HOST='192.168.1.211',
  MASTER_USER='fkoocopy',
  MASTER_PASSWORD='fkoopasswd',
  MASTER_PORT=3306,
  MASTER_LOG_FILE='SQL211-bin.000003',
  MASTER_LOG_POS=106,
  MASTER_CONNECT_RETRY=60;
SLAVE START;
show slave status\G;

# 重置同步日志; mysql 会删除 *-bin.00000* ; 从 *-bin.000001重新开始记录;
# 测试: 此时不能 FLUSH TABLES WITH READ LOCK;  锁定表, 否则 Master_Log_File 不更新
STOP SLAVE;
RESET MASTER;
RESET SLAVE;
SLAVE START;
show master status;
show slave status\G;
```

```
# 已建立好的Replication, show slave status\G; 时,在master和slave上应该显示:
mysql> show slave status\G;
             Slave_IO_Running: Yes
             Slave_SQL_Running: Yes

# 关闭 ip_forward 路由
# vi /etc/sysctl.conf
net.ipv4.ip_forward = 0

#--------------------制作 VMware 快照 SQL211 U1 --------------------

# 在 VMware 快照 SQL210 U1的基础上 (已建立好Replication),

# 接着新建将被监控的库和表,再安装mon,heartbeat
# 说明: 表单必须建立,否则后面配置的 mon 监测不到数据库表单而触发误动作

# 同时将  SQL210 U1 和 SQL211 U1 开机

#在任一台上
mysql -p

mysql>
show databases;
create database fkoodb;
use fkoodb
CREATE TABLE mytable (name VARCHAR(20), sex CHAR(1), \
birth DATE, birthaddr VARCHAR(20));
show tables;
DESCRIBE mytable;
select * from mytable;

# 在另一台上, 应该同时自动同步了新数据库和表单
mysql -p

mysql>
show databases;
use fkoodb
show tables;
DESCRIBE mytable;
select * from mytable;

# 在 SQL210 挂载上软件代码光盘包 "PhaseII.fkoo"
mount /dev/cdrom /mnt/cdrom
```

```
# 安装Mon
rpm -ivh /mnt/cdrom/perl-Time-Period-1.20-2.el5.rf.noarch.rpm
rpm -ivh /mnt/cdrom/perl-Net-SNPP-1.17-1.2.el5.rf.noarch.rpm
rpm -ivh /mnt/cdrom/perl-Math-TrulyRandom-1.0-1.2.el5.rf.i386.rpm
rpm -ivh /mnt/cdrom/perl-Convert-BER-1.3101-1.el5.rf.noarch.rpm
rpm -ivh /mnt/cdrom/perl-Mon-0.11-2.2.el5.rf.noarch.rpm
rpm -ivh /mnt/cdrom/perl-AOL-TOC-0.340-1.el5.rf.noarch.rpm
rpm -ivh /mnt/cdrom/perl-Authen-PAM-0.16-1.2.el5.rf.i386.rpm
rpm -ivh /mnt/cdrom/perl-UNIVERSAL-can-1.12-1.el5.rf.noarch.rpm
rpm -ivh /mnt/cdrom/perl-UNIVERSAL-isa-0.06-1.el5.rf.noarch.rpm
rpm -ivh /mnt/cdrom/perl-Test-MockObject-1.08-1.el5.rf.noarch.rpm
rpm -ivh /mnt/cdrom/perl-Test-Mock-LWP-0.05-1.el5.rf.noarch.rpm
rpm -ivh /mnt/cdrom/perl-HTML-Tagset-3.20-1.el5.rf.noarch.rpm
rpm -ivh /mnt/cdrom/perl-HTML-Parser-3.56-1.el5.rf.i386.rpm
rpm -ivh /mnt/cdrom/libghttp-1.0.9-10.99_2.0.el5.i386.rpm
rpm -ivh /mnt/cdrom/libghttp-devel-1.0.9-10.99_2.0.el5.i386.rpm
rpm -ivh /mnt/cdrom/perl-HTTP-GHTTP-1.07-1.el5.rf.i386.rpm
rpm -ivh /mnt/cdrom/perl-libwww-perl-5.803-2_6.0.el5.noarch.rpm
rpm -ivh /mnt/cdrom/perl-Net-Daemon-0.43-1.el5.rf.noarch.rpm
rpm -ivh /mnt/cdrom/perl-PlRPC-0.2020-1.el5.rf.noarch.rpm
rpm -ivh /mnt/cdrom/perl-DBI-1.602-1.el5.rf.i386.rpm
rpm -ivh /mnt/cdrom/mysqlclient15-5.0.45-1.el5.remi.i386.rpm
rpm -ivh /mnt/cdrom/perl-DBD-mysql-4.006-1.el5.rf.i386.rpm

rpm -i /mnt/cdrom/perl-Time-HiRes-1.9712-1.rf.src.rpm
cd /usr/src/redhat/SPECS
rpmbuild -bp perl-Time-HiRes.spec
cd /usr/src/redhat/BUILD/Time-HiRes-1.9712/
perl Makefile.PL
make
make install
cd ../..
rm -rf BUILD/Time-HiRes-1.9712*
rm -rf SOURCES/Time-HiRes-1.9712.tar.gz
rm -rf SPECS/perl-Time-HiRes.spec

rpm -ivh /mnt/cdrom/mon-1.2.0-1.el5.rf.i386.rpm
cp /etc/mon/mon.cf /etc/mon/mon.cf.bak

# hostgroup 与 watch 之间必须空一行
# hostgroup MasterDB 的 IP, 是 mysql replication 同步复制关系的对方 192.168.1.211  的私网 IP;
# vi /etc/mon/mon.cf
```

```
### group definitions (hostnames or IP addresses)
hostgroup MasterDB 192.168.1.211

watch MasterDB
    service mysql
        interval 5s
        monitor msql-mysql.monitor --mode mysql --username=fkoo_monitor \
        --password=FkooMonitor --database=fkoodb
        period wd {Mon-Sun}
        alert test.alert
                #alert mail.alert fkoo.com@gmail.com
                #upalert mail.alert fkoo.com@gmail.com
                alertevery 600s
                alertafter 3

# 编辑 mon 监测到 mysql 服务失败后的触发脚本
# 保证 mysql 服务为 start; 接管 heartbeat 的 漂移IP 和 主服务
chmod 755 /usr/lib/mon/alert.d/test.alert

echo -ne "
service mysql start
/usr/lib/heartbeat/hb_takeover
" >> /usr/lib/mon/alert.d/test.alert

tail  /usr/lib/mon/alert.d/test.alert

# 复制 msql-mysql.monitor 监控脚本给 mon 服务; 修改权限为可执行
cp /mnt/cdrom/msql-mysql.monitor /usr/lib/mon/mon.d/
chmod 755 /usr/lib/mon/mon.d/msql-mysql.monitor

# 重启/启动 mon 服务; 添加 mon 为自启动服务; 查看 mon 的监测状态
service mon restart
chkconfig mon on
chkconfig --list |grep mon
monshow --full
# 正常情况下:
--------------------------------------------------------------------------------------------------------------
  GROUP            SERVICE      STATUS      LAST       NEXT       ALERTS SUMMARY
R MasterDB         mysql        -           1s         3s         none

# 故障情况下:
--------------------------------------------------------------------------------------------------------------
  GROUP            SERVICE      STATUS      LAST       NEXT       ALERTS SUMMARY
```

```
R MasterDB        mysql        FAIL        0s          0s          1      192.168.1.211

# 确定 mysql 服务为自启动服务
chkconfig mysql on
service mysql start

# 安装 heartbeat 服务
useradd -g haclient hacluster
rpm -ivh /mnt/cdrom/perl-TimeDate-1.16-5.el5.noarch.rpm
rpm -ivh /mnt/cdrom/heartbeat-pils-2.1.4-2.1.i386.rpm
rpm -ivh /mnt/cdrom/heartbeat-stonith-2.1.4-2.1.i386.rpm
rpm -ivh /mnt/cdrom/heartbeat-2.1.4-2.1.i386.rpm
rpm -ivh /mnt/cdrom/libnet-1.1.2.1-2.1.i386.rpm

cp /usr/share/doc/packages/heartbeat/ha.cf  /etc/ha.d/
cp /usr/share/doc/packages/heartbeat/authkeys /etc/ha.d/
cp /usr/share/doc/packages/heartbeat/haresources /etc/ha.d/

chkconfig --add heartbeat
chkconfig  heartbeat on
chkconfig --list |grep heartbeat

# 设置 heartbeat 密钥格式
echo -ne "
auth 1
1 crc
" >> /etc/ha.d/authkeys

tail /etc/ha.d/authkeys
chmod 600 /etc/ha.d/authkeys

# 配置 heartbeat 服务参数 (仅列出需要修改的地方)
# vi /etc/ha.d/ha.cf
debugfile /var/log/ha-debug
logfile        /var/log/ha-log
keepalive 2
deadtime 30
warntime 10
initdead 120
udpport        694
bcast   eth1
ucast eth0 192.168.1.211
auto_failback off
```

```
node    SQL210
node    SQL211

# 配置 heartbeat 服务启动/关闭 的资源
# 设置 SQL210 为漂移地址 10.0.0.209 所在的默认的 master
echo "SQL210 10.0.0.209 " >> /etc/ha.d/haresources
tail /etc/ha.d/haresources

# 启动 heartbeat 服务
service heartbeat start

# 修改数据库复制帐号 fkoocopy 并允许来自 192.168.1.% ( slaves 所在网段 ) 的IP
# 建立 mon 监控数据库服务的帐号 fkoo_monitor 并允许来自192.168.1.% ( slaves 所在网段) 的IP
mysql -p
进入 mysql>
use mysql
delete from user where user='fkoocopy' and host='192.168.1.211';
delete from user where user='fkoo_monitor' and host='192.168.1.211';
grant replication slave on *.* to 'fkoocopy'@'192.168.1.%' identified by 'fkoopasswd';
grant select on *.* to 'fkoo_monitor'@'192.168.1.%' identified by 'FkooMonitor';
flush privileges;
select * from user;
quit;

#--------------------制作 VMware 快照 SQL210 U2 --------------------

# 在 VMware 快照 SQL211 U1的基础上 (已建立好Replication),

# 挂载上软件代码光盘包 "PhaseII.fkoo"
mount /dev/cdrom /mnt/cdrom

# 安装Mon
rpm -ivh /mnt/cdrom/perl-Time-Period-1.20-2.el5.rf.noarch.rpm
rpm -ivh /mnt/cdrom/perl-Net-SNPP-1.17-1.2.el5.rf.noarch.rpm
rpm -ivh /mnt/cdrom/perl-Math-TrulyRandom-1.0-1.2.el5.rf.i386.rpm
rpm -ivh /mnt/cdrom/perl-Convert-BER-1.3101-1.el5.rf.noarch.rpm
rpm -ivh /mnt/cdrom/perl-Mon-0.11-2.2.el5.rf.noarch.rpm
rpm -ivh /mnt/cdrom/perl-AOL-TOC-0.340-1.el5.rf.noarch.rpm
rpm -ivh /mnt/cdrom/perl-Authen-PAM-0.16-1.2.el5.rf.i386.rpm
rpm -ivh /mnt/cdrom/perl-UNIVERSAL-can-1.12-1.el5.rf.noarch.rpm
rpm -ivh /mnt/cdrom/perl-UNIVERSAL-isa-0.06-1.el5.rf.noarch.rpm
rpm -ivh /mnt/cdrom/perl-Test-MockObject-1.08-1.el5.rf.noarch.rpm
rpm -ivh /mnt/cdrom/perl-Test-Mock-LWP-0.05-1.el5.rf.noarch.rpm
rpm -ivh /mnt/cdrom/perl-HTML-Tagset-3.20-1.el5.rf.noarch.rpm
rpm -ivh /mnt/cdrom/perl-HTML-Parser-3.56-1.el5.rf.i386.rpm
rpm -ivh /mnt/cdrom/libghttp-1.0.9-10.99_2.0.el5.i386.rpm
rpm -ivh /mnt/cdrom/libghttp-devel-1.0.9-10.99_2.0.el5.i386.rpm
rpm -ivh /mnt/cdrom/perl-HTTP-GHTTP-1.07-1.el5.rf.i386.rpm
rpm -ivh /mnt/cdrom/perl-libwww-perl-5.803-2_6.0.el5.noarch.rpm
rpm -ivh /mnt/cdrom/perl-Net-Daemon-0.43-1.el5.rf.noarch.rpm
rpm -ivh /mnt/cdrom/perl-PlRPC-0.2020-1.el5.rf.noarch.rpm
rpm -ivh /mnt/cdrom/perl-DBI-1.602-1.el5.rf.i386.rpm
rpm -ivh /mnt/cdrom/mysqlclient15-5.0.45-1.el5.remi.i386.rpm
rpm -ivh /mnt/cdrom/perl-DBD-mysql-4.006-1.el5.rf.i386.rpm

rpm -i /mnt/cdrom/perl-Time-HiRes-1.9712-1.rf.src.rpm
cd /usr/src/redhat/SPECS
rpmbuild -bp perl-Time-HiRes.spec
cd /usr/src/redhat/BUILD/Time-HiRes-1.9712/
perl Makefile.PL
make
make install
cd ../..
rm -rf BUILD/Time-HiRes-1.9712*
rm -rf SOURCES/Time-HiRes-1.9712.tar.gz
rm -rf SPECS/perl-Time-HiRes.spec

rpm -ivh /mnt/cdrom/mon-1.2.0-1.el5.rf.i386.rpm
cp /etc/mon/mon.cf /etc/mon/mon.cf.bak

# hostgroup 与 watch 之间必须空一行
# hostgroup MasterDB 的 IP, 是 mysql replication 同步复制关系的对方 192.168.1.210  的私网 IP;
# vi /etc/mon/mon.cf
### group definitions (hostnames or IP addresses)
hostgroup MasterDB 192.168.1.210

watch MasterDB
    service mysql
        interval 5s
        monitor msql-mysql.monitor --mode mysql --username=fkoo_monitor \
        --password=FkooMonitor --database=fkoodb
        period wd {Mon-Sun}
        alert test.alert
                #alert mail.alert fkoo.com@gmail.com
                #upalert mail.alert fkoo.com@gmail.com
                alertevery 600s
                alertafter 3

# 编辑 mon 监测到 mysql 服务失败后的触发脚本
# 保证 mysql 服务为 start; 接管 heartbeat 的 漂移IP 和 主服务
chmod 755 /usr/lib/mon/alert.d/test.alert

echo -ne ″
service mysql start
/usr/lib/heartbeat/hb_takeover
″ >> /usr/lib/mon/alert.d/test.alert

tail  /usr/lib/mon/alert.d/test.alert

# 复制 msql-mysql.monitor 监控脚本给 mon 服务; 修改权限为可执行
cp /mnt/cdrom/msql-mysql.monitor /usr/lib/mon/mon.d/
chmod 755 /usr/lib/mon/mon.d/msql-mysql.monitor

# 重启/启动 mon 服务; 添加 mon 为自启动服务; 查看 mon 的监测状态
service mon restart
chkconfig mon on
chkconfig --list |grep mon
monshow --full
# 正常情况下:
--------------------------------------------------------------------------------------------------------------
  GROUP            SERVICE      STATUS      LAST        NEXT        ALERTS SUMMARY
R MasterDB         mysql        -           1s          3s          none

# 故障情况下:
--------------------------------------------------------------------------------------------------------------
  GROUP            SERVICE      STATUS      LAST        NEXT        ALERTS SUMMARY
R MasterDB         mysql        FAIL        0s          0s          1      192.168.1.210

# 确定 mysql 服务为自启动服务
chkconfig mysql on
service mysql start

# 安装 heartbeat 服务
useradd -g haclient hacluster
rpm -ivh /mnt/cdrom/perl-TimeDate-1.16-5.el5.noarch.rpm
rpm -ivh /mnt/cdrom/heartbeat-pils-2.1.4-2.1.i386.rpm
rpm -ivh /mnt/cdrom/heartbeat-stonith-2.1.4-2.1.i386.rpm
rpm -ivh /mnt/cdrom/heartbeat-2.1.4-2.1.i386.rpm
rpm -ivh /mnt/cdrom/libnet-1.1.2.1-2.1.i386.rpm
```

```
cp /usr/share/doc/packages/heartbeat/ha.cf  /etc/ha.d/
cp /usr/share/doc/packages/heartbeat/authkeys /etc/ha.d/
cp /usr/share/doc/packages/heartbeat/haresources /etc/ha.d/

chkconfig --add heartbeat
chkconfig  heartbeat on
chkconfig --list |grep heartbeat

# 设置 heartbeat 密钥格式
echo -ne "
auth 1
1 crc
" >> /etc/ha.d/authkeys

tail /etc/ha.d/authkeys
chmod 600 /etc/ha.d/authkeys

# 配置 heartbeat 服务参数 (仅列出需要修改的地方)
# vi /etc/ha.d/ha.cf
debugfile /var/log/ha-debug
logfile        /var/log/ha-log
keepalive 2
deadtime 30
warntime 10
initdead 120
udpport        694
bcast   eth1
ucast eth0 192.168.1.210
auto_failback off
node    SQL210
node    SQL211

# 配置 heartbeat 服务启动/关闭 的资源
# 设置 SQL210 为漂移地址 10.0.0.209 所在的默认的 master
echo "SQL210 10.0.0.209 " >> /etc/ha.d/haresources
tail /etc/ha.d/haresources

# 启动 heartbeat 服务
service heartbeat start

# 修改数据库复制帐号 fkoocopy 并允许来自 192.168.1.% ( slaves 所在网段 ) 的IP
# 建立 mon 监控数据库服务的帐号 fkoo_monitor 并允许来自192.168.1.% ( slaves 所在网段) 的IP
```

```
mysql -p
进入 mysql>
use mysql
delete from user where user='fkoocopy' and host='192.168.1.210';
delete from user where user='fkoo_monitor' and host='192.168.1.210';
grant replication slave on *.* to 'fkoocopy'@'192.168.1.%' identified by 'fkoopasswd';
grant select on *.* to 'fkoo_monitor'@'192.168.1.%' identified by 'FkooMonitor';
flush privileges;
select * from user;
quit;

#--------------------制作 VMware 快照 SQL211 U2 --------------------

# 由 VMware 快照 LVS211 U2 克隆链接生成 SQL212

# 修改 hostname 和 IP
# vi /etc/sysconfig/network
HOSTNAME=SQL212
# vi /etc/sysconfig/network-scripts/ifcfg-eth0
IPADDR=10.0.0.212

# 本测试环境是以同一个物理环境的网卡桥接所有网段
# 仅仅是为了远程控制方便, 设置 eth1 与调试客户端在同一个 IP 子网
# vi /etc/sysconfig/network-scripts/ifcfg-eth1
IPADDR=192.168.1.212

# 修改 memcached 服务启动脚本
# vi /etc/rc.d/init.d/memcached
IP_ADDR=10.0.0.212

vi /etc/my.cnf
# 修改 mysql 同步 ( 仅列出修改部分 )
        # 说明: 因为是 slaves, 不用启动 log-bin; 同时注释掉 log-slave-updates
#log-bin=SQL212-bin
server-id       = 3
master-host=10.0.0.209
report-host=SQL212
#log-slave-updates

# 重启 mysql
service mysql restart

# 修正同步参数的步骤
# 1.  master 上:
show master status;
# 2.  slave 上:
STOP SLAVE;
CHANGE MASTER TO
  MASTER_HOST='192.168.1.209',
  MASTER_USER='fkoocopy',
  MASTER_PASSWORD='fkoopasswd',
  MASTER_PORT=3306,
  MASTER_LOG_FILE='SQL21?-bin.000003',
  MASTER_LOG_POS=106,
  MASTER_CONNECT_RETRY=60;
SLAVE START;
show slave status\G;

# 重置同步日志; mysql 会删除 *-bin.00000* ; 从 *-bin.000001重新开始记录;
# 测试: 此时不能 FLUSH TABLES WITH READ LOCK;  锁定表, 否则 Master_Log_File 不更新
STOP SLAVE;
RESET MASTER;
RESET SLAVE;
SLAVE START;
show master status;
show slave status\G;

# 已建立好的Replication, show slave status\G; 时,在master和slave上应该显示:
mysql> show slave status\G;
             Slave_IO_Running: Yes
             Slave_SQL_Running: Yes

vi /etc/mon/mon.cf
# 配置 mon 监控和触发脚本
        # hostgroup 与 watch 之间必须空一行
        # hostgroup MasterDB 的 IP, 是 mysql M/S 漂移 VIP 192.168.1.209;
        # 说明: 设计是同时监控 master 和 slave 的 mysql 服务, 其中任一个故障, 本地的 mysql 服务, 以使
### group definitions (hostnames or IP addresses)
hostgroup MasterDB 192.168.1.210
hostgroup SlaveDB 192.168.1.211

watch MasterDB
    service mysql
        interval 5s
        monitor msql-mysql.monitor --mode mysql --username=fkoo_monitor \
        --password=FkooMonitor --database=fkoodb
        period wd {Mon-Sun}
        alert test.alert
                #alert mail.alert fkoo.com@gmail.com
                #upalert mail.alert fkoo.com@gmail.com
                alertevery 600s
                alertafter 3

watch SlaveDB
    service mysql
        interval 5s
        monitor msql-mysql.monitor --mode mysql --username=fkoo_monitor \
        --password=FkooMonitor --database=fkoodb
        period wd {Mon-Sun}
        alert test.alert
                #alert mail.alert fkoo.com@gmail.com
                #upalert mail.alert fkoo.com@gmail.com
                alertevery 600s
                alertafter 3

# 编辑 mon 监测到 mysql 服务失败后的触发脚本
# 保证 mysql 服务为 start; 接管 heartbeat 的 漂移IP 和 主服务
# vi /usr/lib/mon/alert.d/test.alert
# service mysql restart \\ 注释掉

chkconfig heartbeat off

#--------------------制作 VMware 快照 SQL212 U1 --------------------

chkconfig mon off

vi /etc/my.cnf
# 修改 mysql 同步为M-H-S 中的 slave
master-host=192.168.1.211


# 重启 mysql
service mysql restart

#--------------------制作 VMware 快照 SQL212 U2 --------------------

# 由 VMware 快照 LVS201 U3 克隆链接生成 IMG206

# 挂载上软件代码光盘包 "PhaseII.fkoo"
```

```
mount /dev/cdrom /mnt/cdrom

# 关闭 ip_forward 路由
# vi /etc/sysctl.conf
net.ipv4.ip_forward = 0

# 修改 hostname 和 IP
# vi /etc/sysconfig/network
HOSTNAME=IMG206
# vi /etc/sysconfig/network-scripts/ifcfg-eth0
IPADDR=192.168.1.206
# vi /etc/sysconfig/network-scripts/ifcfg-eth1
IPADDR=10.0.0.206

# 安装pcre库 (支持 lighttpd 或 nginx 的 rewrite 模块)
cd /tmp
tar zxvf /mnt/cdrom/pcre-7.7.tar.gz
cd pcre-7.7
./configure
make && make install
cd ..
rm -rf  pcre-7.7*

# 安装 php
cd /tmp
tar -jxvf /mnt/cdrom/php-5.2.8.tar.bz2
cd php-5.2.8
patch -p1 < /mnt/cdrom/php-5.2.8-fpm-0.5.10.diff
# 编译安装, 设定安装目录为 /usr/local/php-fcgi; 指定各支持包的安装路径为实际安装目录. 如前
#php支持 CGI/FastCGI需要 php-cgi 命令工具, 因此编译安装不能加 --disable-cli ; 不能添加为 apache2handler 支持的
--with-apxs2=/usr/local/apache/bin/apxs #
# 生产环境需要加载的编译参数:##########
# --disable-debug  \
./configure \
--prefix=/usr/local/php-fcgi \
--with-mysql=/usr/local/mysql \
--with-pdo-mysql=/usr/local/mysql/bin/mysql_config \
--enable-fastcgi \
--enable-force-cgi-redirect \
--with-config-file-path=/usr/local/php-fcgi/etc \
--with-zlib \
--with-zlib-dir \
--with-png-dir=/usr/local/lib \
--with-jpeg-dir=/usr/local/jpeg \
--with-freetype-dir=/usr/local/freetype \
--with-gd=/usr/local/gd \
--with-ttf \
--enable-gd-native-ttf \
--enable-gd-jis-conv \
--with-libxml-dir=/usr/local/libxml2 \
--with-mcrypt=/usr/local/libmcrypt2 \
 --with-iconv \
--with-openssl \
--enable-mbstring \
--enable-pdo \
--without-pdo-sqlite \
--without-sqlite \
--with-curl \
--with-curlwrappers \
--enable-xml \
--with-pear \
--enable-magic-quotes \
--enable-fpm \
--enable-ftp \
--with-bz2 \
--enable-sysvsem \
--enable-exif \
--with-pcre-dir \
--disable-ipv6

make
make install
cp php.ini-dist /usr/local/php-fcgi/etc/php.ini
cp /mnt/cdrom/phpinfo.php /mnt/hgfs/img/
cd ..
rm -rf  php-5.2.8*

# 查看 php-fpm 配置
# vi /usr/local/php-fcgi/etc/php-fpm.conf
# 这个表示php的fastcgi进程监听的ip地址以及端口
<value name="listen_address">127.0.0.1:9000</value>
# 表示php的fastcgi进程以什么用户以及用户组来运行
# 需要手工去掉注释符   <!-- *** -->
<value name="user">nobody</value>
<value name="group">nobody</value>
# 是否显示php错误信息
```

```
<value name="display_errors">0</value>
# 最大的子进程数目
<value name="max_children">5</value>

# 下面运行php-fpm；现在php的fastcgi进程就已经在后台运行，并监听127.0.0.1的9000端口。
/usr/local/php-fcgi/bin/php-cgi --fpm

# 可以用ps和netstat来看看结果：
ps aux | grep php-cgi
netstat -tpnl | grep php-cgi

# php-fpm 管理程序
/usr/local/php-fcgi/sbin/php-fpm
# 该程序有如下参数：
        start 启动php的fastcgi进程
        stop 强制终止php的fastcgi进程
        quit 平滑终止php的fastcgi进程
        restart 重启php的fastcgi进程
        reload 重新加载php的php.ini
        logrotate 重新启用log文件

        也就是说，在修改了php.ini之后，我们可以使用
        /usr/local/php-fcgi/sbin/php-fpm reload
        这样，就保持了在php的fastcgi进程持续运行的状态下，又重新加载了php.ini。

# 给 php-fpm 命令增加系统环境变量 /usr/local/php-fcgi/sbin/
# vi /etc/profile
export PATH="$PATH:/usr/local/php-fcgi/sbin/"
# 重新登录
# su -

# 将  php-fpm 加入开机启动项
echo "/usr/local/php-fcgi/sbin/php-fpm start" >> /etc/rc.local
cat /etc/rc.local

# 优化 php-fpm ( 未配置, 需要在生产环境测试 )
# vi /usr/local/php-fcgi/etc/php-fpm.conf
<value name="max_children">128</value>
<value name="MaxSpareServers">250</value>
<value name="rlimit_files">51200</value>
<value name="max_requests">51200</value>

# 安装 libevent
```

```
cd /tmp/
tar vxzf /mnt/cdrom/libevent-1.4.9-stable.tar.gz
cd libevent-1.4.9-stable/
./configure
make
make install
# 建立一个符号连接 ###
ln -s /usr/local/lib/libevent-1.4.so.2 /usr/lib
cd ..
rm -rf libevent-1.4.9-stable

# 安装 memcached 服务器端
cd /tmp/
tar vxzf /mnt/cdrom/memcached-1.2.6.tar.gz
cd memcached-1.2.6/
./configure --prefix=/usr/local/memcached \
--with-libevent=/usr
make
make install
cd ..
rm -rf memcached-1.2.6/

# memcached 启动命令
/usr/local/memcached/bin/memcached -l 10.0.0.206 -d -p 62880 -u nobody -m 2
# 表示用 daemon 的方式启动 memcached，监听在 10.0.0.1 的 62880 端口上，运行用户为 nobody，为其分配 2MB 的内存。

# 查看 memcached 选项
# /usr/local/memcached/bin/memcached -h
        -t <num> number of threads to use, default 4

# 添加 memcached 为服务
cp /mnt/cdrom/memcached.init /etc/rc.d/init.d/memcached
chmod 755 /etc/rc.d/init.d/memcached
# vi /etc/rc.d/init.d/memcached
        PORT1=62880
        USER=nobody
        MAXCONN=1024
        CACHESIZE=20
        IP_ADDR=10.0.0.2
        OPTIONS="-t 8"
# 添加绑定IP的选项 -l $IP_ADDR
  daemon $MEMDAEMON -d -p $PORT1 -u $USER -m $CACHESIZE -c $MAXCONN -l $IP_ADDR $OPTIONS
```

```
chkconfig  --add memcached
chkconfig memcached  on
chkconfig  --list | grep mem
service memcached restart
ps aux | grep mem

# 安装memcache php客户端
cd /tmp/
tar xvfz /mnt/cdrom/memcache-2.2.4.tgz
cd memcache-2.2.4
/usr/local/php-fcgi/bin/phpize
./configure \
--enable-memcache \
--with-php-config=/usr/local/php-fcgi/bin/php-config \
--with-zlib-dir
make && make install
        # Installing shared extensions:     /usr/local/php-fcgi/lib/php/extensions/no-debug-non-zts-20060613/
cd ..
rm -rf memcache-2.2.4*

# vi /usr/local/php-fcgi/etc/php.ini
extension_dir = "/usr/local/php-fcgi/lib/php/extensions/no-debug-non-zts-20060613/"
extension=memcache.so

# 安装 eAccelerator PHP 加速器
cd /tmp/
tar -xvf  /mnt/cdrom/eaccelerator-0.9.5.3.tar.tar
cd eaccelerator-0.9.5.3/
/usr/local/php-fcgi/bin/phpize
./configure --enable-eaccelerator=shared \
--with-php-config=/usr/local/php-fcgi/bin/php-config
make
make install
        # Installing shared extensions:     /usr/local/php-fcgi/lib/php/extensions/no-debug-non-zts-20060613/
cd ..
rm -rf eaccelerator-0.9.5.3/
mkdir /tmp/eaccelerator && chmod 777 /tmp/eaccelerator && touch /var/log/eaccelerator_log
# 编辑php.ini , 将 eAccelerator 作为 PHP Extension 添加
# vi /usr/local/php-fcgi/etc/php.ini
# 加上:
extension="eaccelerator.so"
eaccelerator.shm_size="16"
eaccelerator.cache_dir="/tmp/eaccelerator"
eaccelerator.enable="1"
eaccelerator.optimizer="1"
eaccelerator.check_mtime="1"
eaccelerator.debug="0"
eaccelerator.log_file = "/var/log/eaccelerator_log "
eaccelerator.filter=""
eaccelerator.shm_max="0"
eaccelerator.shm_ttl="0"
eaccelerator.shm_prune_period="0"
eaccelerator.shm_only="0"
eaccelerator.compress="1"
eaccelerator.compress_level="9"

# 先安装 Rudiments 于安装SQL Relay之前
cd /tmp/
tar vxzf  /mnt/cdrom/rudiments-0.31.tar.gz
cd rudiments-0.31
./configure --prefix=/usr/local/rudiments
make
make install
cd ..
rm -rf rudiments-0.31
# 安装SQL Relay:
cd /tmp/
tar vxzf /mnt/cdrom/sqlrelay-0.39.4.tar.gz
cd sqlrelay-0.39.4
./configure --prefix=/usr/local/sqlrelay --with-rudiments-prefix=/usr/local/rudiments \
--with-mysql-prefix=/usr/local/mysql \
--with-php-prefix=/usr/local/php-fcgi
make
make install
cd ..
rm -rf sqlrelay-0.39.4

# 修改 php.ini 文件
# vi /usr/local/php-fcgi/etc/php.ini
extension_dir = "/usr/local/php-fcgi/lib/php/extensions/no-debug-non-zts-20060613/"
extension=sql_relay.so
# 修改 SQL Relay 的配置文件
cp /usr/local/sqlrelay/etc/sqlrelay.conf.example /usr/local/sqlrelay/etc/sqlrelay.conf

# 安装 gamin-devel ( 32位系统版 ), 以支持 -with-fam
rpm -ivh /mnt/cdrom/gamin-devel-0.1.7-8.el5.i386.rpm
```

```
# 安装 lighttpd
tar jxvf /mnt/cdrom/lighttpd-1.4.20.tar.bz2
cd lighttpd-1.4.20/
./configure \
--prefix=/usr/local/lighttpd \
--with-webdav-props \
--with-webdav-locks \
--with-pcre \
--with-gdbm \
--with-memcache \
--with-linux-aio \
--with-bzip2 \
--enable-lfs \
--with-fam \
--disable-ipv6
make
make install

groupadd lighttpd
useradd -g lighttpd -s /sbin/nologin -d /dev/null lighttpd
mkdir /etc/lighttpd/
mkdir /var/log/lighttpd
chown -R lighttpd.lighttpd /var/log/lighttpd/
chmod 750 /var/log/lighttpd/
cp ./doc/lighttpd.conf /etc/lighttpd/
cp ./doc/rc.lighttpd.redhat /etc/init.d/lighttpd
cp ./doc/sysconfig.lighttpd /etc/sysconfig/lighttpd
chmod 755 /etc/init.d/lighttpd
cd ..
rm -rf lighttpd-1.4.20/
chkconfig --add lighttpd
chkconfig lighttpd on

# 安装日志回滚
cd /tmp/
tar xvf /mnt/cdrom/cronolog-1.6.2.tar.tar
cd cronolog-1.6.2
./configure
make && make install
cd ..
rm -rf cronolog-1.6.2
```

```
# 修正 lighttpd 程序所在的目录
vi /etc/init.d/lighttpd
lighttpd="/usr/local/lighttpd/sbin/lighttpd"

# 编辑 lighttpd.conf , 打开如下的模块
# vi /etc/lighttpd/lighttpd.conf
server.modules              = (
                               "mod_rewrite",
                                "mod_access", \\ 默认为打开
                               "mod_fastcgi",
                               "mod_compress",
                               "mod_accesslog" ) \\ 默认为打开

# 修改 lighttpd 相关目录
server.document-root        = "/mnt/hgfs/img/"
# 访问日志, 以及日志格式 (combined), 使用X-Forwarded-For可越过代理读取真实ip
accesslog.format = "%{X-Forwarded-For}i %v %h %l %u %t \"%r\" %>s %b"
accesslog.filename = "|/usr/local/sbin/cronolog /var/log/lighttpd/access.log.%Y-%m-%d-%H"
# 设置禁止访问的文件扩展名
url.access-deny = ( "~", ".inc", ".tpl" )
# 服务监听端口
server.port = 80
# virtual directory listings 如果没有找到index文件就列出目录。建议disable。
dir-listing.activate = "disable"
# 服务运行使用的用户及用户组
server.username = "lighttpd"
server.groupname = "lighttpd"
# 设定文件过期时间
expire.url = (
"/css/" => "access 2 hours",
"/js/" => "access 2 hours",
)
# gzip压缩存放的目录以及需要压缩的文件类型
        # 可以指定某些静态资源类型使用压缩方式传输, 节省带宽,
        # 对于大量AJAX应用来说, 可以极大提高页面加载速度。
compress.cache-dir = "/tmp/lighttpd/cache/compress/"
compress.filetype = ("text/plain", "text/html","text/javascript","text/css")

# 配置 fastcgi
server.modules += ("mod_fastcgi")
fastcgi.server = ( ".php" =>
   ( "localhost" =>
     (
        "host"     => "127.0.0.1",
        "port"     => 1026,
        #"socket" => "/tmp/php-fastcgi.socket",
        "bin-path" => "/usr/local/php-fcgi/bin/php-cgi",
        "idle-timeout" => 20,
        "max-procs" => 4,
        "bin-environment" => (
        "PHP_FCGI_CHILDREN" => "8",
        "PHP_FCGI_MAX_REQUESTS" => "500"
        )
      )
    )
  )

# vi /usr/local/php-fcgi/etc/php.ini
cgi.fix_pathinfo=1

# 优化 lighttpd
# 说明: server.network-backend = "linux-sendfile" # lighttpd1.4 适用 sendfile 已经非常好了
#       server.network-backend = "linux-aio-sendfile" # lighttpd1.5 适用 但不是纯粹的AIO 大部分的还是sendfile #
echo -ne "
server.max-keep-alive-requests = 0
server.max-keep-alive-idle = 30
server.max-read-idle = 60
server.max-write-idle = 360
server.max-fds = 40240
server.event-handler = \"linux-sysepoll\"
server.stat-cache-engine = \"fam\"
server.network-backend = \"linux-sendfile\"
" >> /etc/lighttpd/lighttpd.conf
tail /etc/lighttpd/lighttpd.conf

# 配置 NFS 服务器端, 并将 lighttpd 主目录 export 出来
# 设置 NFS 服务为自启动
chkconfig portmap on
chkconfig nfs on
service portmap start
service nfs start

# 新建 /nfs 目录, 更改 lighttpd 主目录为 /nfs
 mkdir /nfs
 chmod 777 /nfs
 ls -al /nfs/
```

```
# vi /etc/lighttpd/lighttpd.conf
server.document-root        = "/nfs"

# 设置 NFS 服务器端将 lighttpd 主目录 /nfs 导出给 DB LAN 10.0.0.0/24
# 设定为 rw (可读写);
# sync (将数据同步写入内存缓冲区与磁盘中，效率低，但可以保证数据的一致性);
# no_wdelay (若有写操作则立即执行，应与sync配合使用)
echo -ne "
/nfs 10.0.0.0/24(rw,sync,no_wdelay)
" >> /etc/exports
cat /etc/exports

# 重新导出 或 重载 NFS 服务
exportfs -rv
service nfs reload

# 查看导出列表
# showmount  -e
        Export list for IMG249:
        /nfs 10.0.0.0/30

# 修改最大打开文件数
echo -ne "
* soft nofile 65536
* hard nofile 65536
" >>/etc/security/limits.conf

cat /proc/sys/fs/file-max
cat /proc/sys/fs/file-nr

# tcpip调优
echo -ne "
net.ipv4.ip_local_port_range = 1024 65536
net.core.rmem_max = 16777216
net.core.wmem_max = 16777216
net.ipv4.tcp_rmem = 4096 87380 16777216
net.ipv4.tcp_wmem = 4096 65536 16777216
net.ipv4.tcp_fin_timeout = 3
net.ipv4.tcp_tw_recycle = 1
net.core.netdev_max_backlog = 30000
net.ipv4.tcp_no_metrics_save = 1
net.core.somaxconn = 262144
# net.ipv4.tcp_syncookies = 0
```

```
net.ipv4.tcp_max_orphans = 262144
net.ipv4.tcp_max_syn_backlog = 262144
net.ipv4.tcp_synack_retries = 2
net.ipv4.tcp_syn_retries = 2
fs.file-max = 65536
" >> /etc/sysctl.conf
tail -20 /etc/sysctl.conf

# vi /etc/sysctl.conf
## net.ipv4.tcp_syncookies = 1

# sysctl -p /etc/sysctl.conf

# 测试 lighttpd+PHP
cp /mnt/hgfs/img/phpinfo.php /nfs
service lighttpd restart

# 此项还未配置, 未测试
# 修改/etc/hosts.allow和/etc/hosts.deny达到限制CLIENT的目的
echo -ne "
portmap: 10.0.0.1/255.255.255.255 : allow
" >> /etc/hosts.allow
cat /etc/hosts.allow

echo -ne "
portmap: ALL : deny
" >> /etc/hosts.deny
cat /etc/hosts.deny

# 关闭 ip_forward 路由
# vi /etc/sysctl.conf
net.ipv4.ip_forward = 0

#--------------------制作 VMware 快照 IMG206 U1 --------------------

# 由 VMware 快照 IMG206 U1 克隆链接生成 IMG207

# 修改 hostname 和 IP
# vi /etc/sysconfig/network
HOSTNAME=IMG206
# vi /etc/sysconfig/network-scripts/ifcfg-eth0
IPADDR=192.168.1.206
# vi /etc/sysconfig/network-scripts/ifcfg-eth1
```

```
IPADDR=10.0.0.206

# 添加 memcached 为服务
# vi /etc/rc.d/init.d/memcached
        IP_ADDR=10.0.0.207

#-------------------- 制作 VMware 快照 IMG207 U1 --------------------

# 由 VMware 快照 IMG206 U1 克隆链接生成 IMG208

# 修改 hostname 和 IP
# vi /etc/sysconfig/network
HOSTNAME=IMG206
# vi /etc/sysconfig/network-scripts/ifcfg-eth0
IPADDR=192.168.1.206
# vi /etc/sysconfig/network-scripts/ifcfg-eth1
IPADDR=10.0.0.206

# 添加 memcached 为服务
# vi /etc/rc.d/init.d/memcached
        IP_ADDR=10.0.0.207

#-------------------- 制作 VMware 快照 IMG208 U1 --------------------

# 修正 /etc/hosts, 并生成快照:
LVS201 U6_1     ( president=0 )
LVS201 U7_1     ( president=300 )
WEB203 U1_1
WEB204 U1_1
WEB205 U1_1
SQL210 U2_1
SQL211 U2_1
SQL212 U1_1

#-------------------- 安装 MogileFS ------------------------------------------

# ----- 在 VMware 快照 SQL210 U2_1 的基础上,安装 MogileFS 数据库

# 创建 MogileFS 数据库
        # 一些扩展库不支持 mysql 的 new passwords; 因此这里用 “OLD_PASSWORD”
        # 在更改密码前, 请确定比本例中的写法更好
mysql -p
# 进入 mysql>
```

```
CREATE DATABASE mogilefs_fk;
GRANT ALL ON mogilefs_fk.* TO 'mogile_fk'@'%';
SET PASSWORD FOR 'mogile_fk'@'%' = OLD_PASSWORD( 'mogile_pw' );
FLUSH PRIVILEGES;
use mysql;
select * from user;
show databases;
quit

# 配置 mysql, 同步 mogilefs_fk 数据库
# vi /etc/my.cnf
binlog-do-db = mogilefs_fk
replicate-do-db = mogilefs_fk

# 重启 mysql 服务
service mysql restart

# 检查同步状态
mysql -p
# 进入 mysql>
show master status;
show slave status\G;

stop slave;
reset slave;
start slave;
show slave status\G;

# ----- 在 VMware 快照 SQL211 U2_1 的基础上,安装 MogileFS 数据库

# 创建 MogileFS 数据库
        # 一些扩展库不支持 mysql 的 new passwords; 因此这里用 “OLD_PASSWORD”
        # 在更改密码前, 请确定比本例中的写法更好
mysql -p
# 进入 mysql>
CREATE DATABASE mogilefs_fk;
GRANT ALL ON mogilefs_fk.* TO 'mogile_fk'@'%';
SET PASSWORD FOR 'mogile_fk'@'%' = OLD_PASSWORD( 'mogile_pw' );
FLUSH PRIVILEGES;
use mysql;
select * from user;
show databases;
quit
```

```
# 配置 mysql, 同步 mogilefs_fk 数据库
# vi /etc/my.cnf
binlog-do-db = mogilefs_fk
replicate-do-db = mogilefs_fk

# 重启 mysql 服务
service mysql restart

# 检查同步状态
mysql -p
# 进入 mysql>
show master status;
show slave status\G;

# ----- 在 VMware 快照 SQL212 U1_1 的基础上,安装 MogileFS 数据库

# 创建 MogileFS 数据库
        # 一些扩展库不支持 mysql 的 new passwords; 因此这里用 “OLD_PASSWORD”
        # 在更改密码前, 请确定比本例中的写法更好
mysql -p
# 进入 mysql>
CREATE DATABASE mogilefs_fk;
GRANT ALL ON mogilefs_fk.* TO 'mogile_fk'@'%';
SET PASSWORD FOR 'mogile_fk'@'%' = OLD_PASSWORD( 'mogile_pw' );
FLUSH PRIVILEGES;
use mysql;
select * from user;
show databases;
quit

# 配置 mysql, 同步 mogilefs_fk 数据库
# vi /etc/my.cnf
replicate-do-db = mogilefs_fk

# 重启 mysql 服务
service mysql restart

# 检查同步状态
mysql -p
# 进入 mysql>
show master status;
show slave status\G;
```

```
# ----- 继续在 IMG206 / IMG207 / IMG208 上， 安装 MogileFS

# 在 VMware 快照 IMG206 U1 / IMG207 U1 / IMG208 U1 的基础上, 安装 MogileFS Storage Node 存储节点

# 挂载上软件代码光盘包 "PhaseII.fkoo"
mount /dev/cdrom /mnt/cdrom

# 安装 mogilefs 服务器
rpm -ivh /mnt/cdrom/perl-IO-stringy-2.110-1.2.el5.rf.noarch.rpm
rpm -ivh /mnt/cdrom/perl-HTML-Tagset-3.20-1.el5.rf.noarch.rpm
rpm -ivh /mnt/cdrom/perl-HTML-Parser-3.56-1.el5.rf.i386.rpm
rpm -ivh /mnt/cdrom/libghttp-1.0.9-10.99_2.0.el5.i386.rpm
rpm -ivh /mnt/cdrom/perl-HTTP-GHTTP-1.07-1.el5.rf.i386.rpm
rpm -ivh /mnt/cdrom/perl-libwww-perl-5.803-2_6.0.el5.noarch.rpm
rpm -ivh /mnt/cdrom/perl-MogileFS-Client-1.08-1.fc8.noarch.rpm

rpm -ivh /mnt/cdrom/perl-Compress-Raw-Zlib-2.008-1.el5.rf.i386.rpm
rpm -ivh /mnt/cdrom/perl-IO-Compress-Base-2.008-1.el5.rf.noarch.rpm
rpm -ivh /mnt/cdrom/perl-IO-Compress-Zlib-2.008-1.el5.rf.noarch.rpm
rpm -ivh /mnt/cdrom/perl-Compress-Zlib-2.008-1.el5.rf.noarch.rpm
rpm -ivh /mnt/cdrom/perl-MogileFS-Utils-2.12-1.el5.noarch.rpm

rpm -ivh /mnt/cdrom/perl-Net-Netmask-1.9015-1.el5.rf.noarch.rpm
rpm -ivh /mnt/cdrom/perl-Net-Daemon-0.43-1.el5.rf.noarch.rpm
rpm -ivh /mnt/cdrom/perl-PlRPC-0.2020-1.el5.rf.noarch.rpm
rpm -ivh /mnt/cdrom/perl-DBI-1.602-1.el5.rf.i386.rpm
rpm -ivh /mnt/cdrom/mysqlclient15-5.0.45-1.el5.remi.i386.rpm
rpm -ivh /mnt/cdrom/perl-DBD-mysql-4.006-1.el5.rf.i386.rpm
rpm -ivh /mnt/cdrom/perl-Gearman-1.09-1.el5.noarch.rpm
rpm -ivh /mnt/cdrom/perl-Sys-Syscall-0.22-1.el5.rf.noarch.rpm
rpm -ivh /mnt/cdrom/perl-Danga-Socket-1.58-1.el5.rf.noarch.rpm
rpm -ivh /mnt/cdrom/perl-Gearman-Client-Async-0.94-3.el5.noarch.rpm
rpm -ivh /mnt/cdrom/perl-Gearman-Server-1.09-1.el5.noarch.rpm
rpm -ivh /mnt/cdrom/perl-BSD-Resource-1.2901-1.el5.rf.i386.rpm
rpm -ivh /mnt/cdrom/perl-IO-AIO-2.51-1.el5.rf.i386.rpm
rpm -ivh /mnt/cdrom/Perlbal-1.59-1.el5.noarch.rpm

rpm -i /mnt/cdrom/perl-mogilefs-server-2.20-4.el5.src.rpm
cd /usr/src/redhat/SPECS
rpmbuild -bp perl-mogilefs-server.spec
cd /usr/src/redhat/BUILD/mogilefs-server-2.20/
perl Makefile.PL
```

```
make
make install
cd ../..
rm -rf BUILD/mogilefs-server-2.20
rm -rf SPECS/perl-mogilefs-server.spec
rm -rf SOURCES/mog*

# 在任意一个 IMG206 / IMG207 / IMG208 上，向数据库节点安装 mogilefs_fk 数据库表单
# dbhost=10.0.0.209，是 mysql M-H-S 结构中的 漂移地址
mogdbsetup --dbhost=10.0.0.209 --dbname=mogilefs_fk --dbuser=mogile_fk --dbpass=mogile_pw

# 检查 mogilefs_fk 数据库同步
use mogilefs_fk;
show tables;

# 新建存储目录，生产环境应该是单独的分区
# 新建配置文件目录
mkdir /var/mogdata
mkdir /etc/mogilefs

# 在 IMG206 生成存储节点配置文件
# server = lighttpd，表示 mogstored 启用 lighttpd 作为 webdav
# httplisten = 10.0.0.206:7500，lighttpd 绑定监听内网的 7500 端口
echo -ne "server = lighttpd
serverbin = /usr/local/lighttpd/sbin/lighttpd
daemonize = 1
maxconns = 10000
httplisten = 10.0.0.206:7500
mgmtlisten = 10.0.0.206:7501
docroot = /var/mogdata
" >> /etc/mogilefs/mogstored.conf
cat /etc/mogilefs/mogstored.conf

# 在 IMG207 生成存储节点配置文件
# server = lighttpd，表示 mogstored 启用 lighttpd 作为 webdav
# httplisten = 10.0.0.207:7500，lighttpd 绑定监听内网的 7500 端口
echo -ne "server = lighttpd
serverbin = /usr/local/lighttpd/sbin/lighttpd
daemonize = 1
maxconns = 10000
httplisten = 10.0.0.207:7500
mgmtlisten = 10.0.0.207:7501
```

```
docroot = /var/mogdata
" >> /etc/mogilefs/mogstored.conf
cat /etc/mogilefs/mogstored.conf

# 在 IMG208 生成存储节点配置文件
# server = lighttpd, 表示 mogstored 启用 lighttpd 作为 webdav
# httplisten = 10.0.0.208:7500, lighttpd 绑定监听内网的 7500 端口
echo -ne "server = lighttpd
serverbin = /usr/local/lighttpd/sbin/lighttpd
daemonize = 1
maxconns = 10000
httplisten = 10.0.0.208:7500
mgmtlisten = 10.0.0.208:7501
docroot = /var/mogdata
" >> /etc/mogilefs/mogstored.conf
cat /etc/mogilefs/mogstored.conf

# 在 IMG206 / IMG207 / IMG208 复制并设定 mogstored 存储启动文件
cp /mnt/cdrom/mogstored.init /etc/init.d/mogstored
chmod 755 /etc/rc.d/init.d/mogstored
chkconfig --add mogstored
chkconfig mogstored on
service mogstored restart
ps -ef | grep mogstored

# 在 IMG206 生成 Tracker 配置文件
# listen = 10.0.0.206, 跟踪器使用 IMG206 的内网 IP
# db_dsn = 10.0.0.209, 是 MogileFS 数据库的 IP
# default_mindevcoun 是默认备份在多少个 device 上备份;即备份多少份
echo -ne "daemonize = 1
db_dsn = DBI:mysql:mogilefs_fk:10.0.0.209
db_user = mogile_fk
db_pass = mogile_pw
listen = 10.0.0.206:6001
conf_port = 6001
query_jobs = 2
listener_jobs = 10
delete_jobs = 1
replicate_jobs = 5
reaper_jobs = 1
default_mindevcount = 2
" >> /etc/mogilefs/mogilefsd.conf
cat /etc/mogilefs/mogilefsd.conf
```

```
# 在 IMG207 生成 Tracker 配置文件
# listen = 10.0.0.207, 跟踪器使用 IMG207 的内网 IP
# db_dsn = 10.0.0.209, 是 MogileFS 数据库的 IP
# default_mindevcoun 是默认备份在多少个 device 上备份;即备份多少份
echo -ne "daemonize = 1
db_dsn = DBI:mysql:mogilefs_fk:10.0.0.209
db_user = mogile_fk
db_pass = mogile_pw
listen = 10.0.0.207:6001
conf_port = 6001
query_jobs = 2
listener_jobs = 10
delete_jobs = 1
replicate_jobs = 5
reaper_jobs = 1
default_mindevcount = 2
" >> /etc/mogilefs/mogilefsd.conf
cat /etc/mogilefs/mogilefsd.conf

# 在 IMG208 生成 Tracker 配置文件
# listen = 10.0.0.208, 跟踪器使用 IMG208 的内网 IP
# db_dsn = 10.0.0.209, 是 MogileFS 数据库的 IP
# default_mindevcoun 是默认备份在多少个 device 上备份;即备份多少份
echo -ne "daemonize = 1
db_dsn = DBI:mysql:mogilefs_fk:10.0.0.209
db_user = mogile_fk
db_pass = mogile_pw
listen = 10.0.0.208:6001
conf_port = 6001
query_jobs = 2
listener_jobs = 10
delete_jobs = 1
replicate_jobs = 5
reaper_jobs = 1
default_mindevcount = 2
" >> /etc/mogilefs/mogilefsd.conf
cat /etc/mogilefs/mogilefsd.conf

# 在 IMG206 / IMG207 / IMG208 复制并设定 mogilefsd Tracker服务启动文件
cp /mnt/cdrom/mogilefsd.init /etc/init.d/mogilefsd
# 显示启动 mogilefsd 服务的用户名
# cat /etc/init.d/mogilefsd |grep dbUser
# 在 IMG206 / IMG207 / IMG208 取消执行 sudo 命令时需要终端的限制
vi /etc/sudoers
# Defaults    requiretty

# 在 IMG206 / IMG207 / IMG208 为运行 mogilefsd 添加运行用户 mogile_fk
# 此用户与 dbUser / mogilefsd.conf 和 dbUser / mogilefsd 的相同
adduser mogile_fk

# 在 IMG206 / IMG207 / IMG208 mogilefsd  与 syslog 有依存关系，启动syslog服务
chkconfig syslog on
service syslog start

# 在 IMG206 / IMG207 / IMG208 设置 mogilefsd Tracker服务开启状态
chmod 755 /etc/rc.d/init.d/mogilefsd
chkconfig --add mogilefsd
chkconfig mogilefsd on
service mogilefsd restart
ps -ef |grep mogilefsd

# 在 IMG206 生成 mogadm 配置文件
echo -ne "trackers = 10.0.0.206:6001
" >> /etc/mogilefs/mogilefs.conf
cat /etc/mogilefs/mogilefs.conf

# 在 IMG207 生成 mogadm 配置文件
echo -ne "trackers = 10.0.0.207:6001
" >> /etc/mogilefs/mogilefs.conf
cat /etc/mogilefs/mogilefs.conf

# 在 IMG208 生成 mogadm 配置文件
echo -ne "trackers = 10.0.0.208:6001
" >> /etc/mogilefs/mogilefs.conf
cat /etc/mogilefs/mogilefs.conf

# 在 IMG206 生成 mogtool 配置文件
echo -ne "trackers = 10.0.0.206:6001
domain = IMG_Domain
class = IMG_Class01
lib = /usr/lib/perl5/vendor_perl/5.8.8/
gzip = 1
big = 1
overwrite = 1
```

```
chunksize = 32M
receipt = admin@fkoo.com
verify = 1
concurrent = 3
" >> /etc/mogilefs/mogtool.conf
cat /etc/mogilefs/mogtool.conf

# 在 IMG207 生成 mogtool 配置文件
echo -ne "trackers = 10.0.0.207:6001
domain = IMG_Domain
class = IMG_Class01
lib = /usr/lib/perl5/vendor_perl/5.8.8/
gzip = 1
big = 1
overwrite = 1
chunksize = 32M
receipt = admin@fkoo.com
verify = 1
concurrent = 3
" >> /etc/mogilefs/mogtool.conf
cat /etc/mogilefs/mogtool.conf

# 在 IMG208 生成 mogtool 配置文件
echo -ne "trackers = 10.0.0.208:6001
domain = IMG_Domain
class = IMG_Class01
lib = /usr/lib/perl5/vendor_perl/5.8.8/
gzip = 1
big = 1
overwrite = 1
chunksize = 32M
receipt = admin@fkoo.com
verify = 1
concurrent = 3
" >> /etc/mogilefs/mogtool.conf
cat /etc/mogilefs/mogtool.conf

# 在任一 IMG206 / IMG207 / IMG208 上用 mogadm 添加存储节点
mogadm host add Mog_IMG206 --ip=10.0.0.206 --port=7500 --status=alive
mogadm host add Mog_IMG207 --ip=10.0.0.207 --port=7500 --status=alive
mogadm host add Mog_IMG208 --ip=10.0.0.208 --port=7500 --status=alive
# 在任一 IMG206 / IMG207 / IMG208 列出存储节点
mogadm host list
```

```
# 在任一 IMG206 / IMG207 / IMG208 向存储节点 Mog_IMG206 中添加编号为 1 的设备; 其余依次同理.
mogadm device add Mog_IMG206 1
mogadm device add Mog_IMG207 2
mogadm device add Mog_IMG208 3
# 在任一 IMG206 / IMG207 / IMG208 列出存储设备
mogadm device list

# 在 IMG206 为存储设备 dev1 添加工作目录
mkdir -p /var/mogdata/dev1
# 在 IMG207 为存储设备 dev1 添加工作目录
mkdir -p /var/mogdata/dev2
# 在 IMG208 为存储设备 dev1 添加工作目录
mkdir -p /var/mogdata/dev3

# 在任一 IMG206 / IMG207 / IMG208 监测存储系统
mogadm check
# 在任一 IMG206 / IMG207 / IMG208 添加存储域 IMG_Domain; 向存储域 IMG_Domain 中添加存储类别 IMG_Class01
mogadm domain add IMG_Domain
mogadm class add IMG_Domain IMG_Class01

vi test.pl
# 生成 MogileFS 存储系统测试文件
#=============test.pl=====================
use MogileFS::Client;
my $mogfs = MogileFS::Client->new(domain=>'IMG_Domain', hosts=>['10.0.0.208:6001'], root=>'/var/mogdata',);
my $fh = $mogfs->new_file("file_key", "IMG_Class01");
die $fh unless $fh->print($mogfs->readonly);
my $content = "file.txt";
@num = $mogfs->store_content("file_key","IMG_Class01",$content);
print "@num \n";
my $file_contents = $mogfs->get_file_data("file_key");
print "$file_contents \n";
#$mogfs->delete("file_key");
$fh->print($file_contents);
@urls = $mogfs->get_paths("file_key");
print "@urls \n";
#=============EOF========================

# 执行测试
# perl test.pl
8
SCALAR(0x8e68b74)
http://10.0.0.206:7500/dev1/0/000/000/0000000014.fid
```

```
vi dbtest.pl
# 生成 MogileFS 数据库连接测试文件
#============dbtest.pl====================
#!/usr/bin/perl
# DBI is perl module used to connect to the database
use DBI;

# hostname or ip of server (for local testing, localhost should work)
$config{'dbServer'} = "10.0.0.209";
$config{'dbUser'} = "mogile_fk";
$config{'dbPass'} = "mogile_pw";
$config{'dbName'} = "mogilefs_fk";
$config{'dataSource'} = "DBI:mysql:$config{'dbName'}:$config{'dbServer'}";

# Connect to MySQL
my $dbh = DBI->connect($config{'dataSource'},$config{'dbUser'},$config{'dbPass'}) or
die "Can't connect to $config{'dataSource'}<br>$DBI::errstr";
 print "Connected successfully<br>";
$dbh->disconnect();
#==============EOF=======================

# 执行测试
# perl dbtest.pl
Connected successfully<br>

# 在 IMG206 / IMG207 / IMG208 上均安装 mogilefs php extension 扩展库
mount /dev/cdrom /mnt/cdrom
cd /tmp/
tar xvf /mnt/cdrom/neon-0.28.3.tar.tar
cd neon-0.28.3/
./configure
make
make install
cd ..
rm -rf neon-0.28.3

cd /tmp/
tar jxvf /mnt/cdrom/mogilefs-0.7.5b3.tar.tar
cd mogilefs-0.7.5b3/
/usr/local/php-fcgi/bin/phpize
make clean
./configure --with-php-config=/usr/local/php-fcgi/bin/php-config
```

```
make
make install
cd ..
rm -rf mogilefs-0.7.5b3
# Installing shared extensions:     /usr/local/php-fcgi/lib/php/extensions/no-debug-non-zts-20060613/

# 修改 php.ini 配置文件, 添加 mogilefs.so 扩展库
# 确认 extension_dir = "/usr/local/php-fcgi/lib/php/extensions/no-debug-non-zts-20060613/"
# vi /usr/local/php-fcgi/etc/php.ini
extension=mogilefs.so

# 测试 mogilefs.so 扩展库是否安装成功
/usr/local/php-fcgi/bin/php -r "var_dump(extension_loaded('mogilefs'));"
# 如果成功,应该显示为:
bool(true)

#-------------------- 制作 VMware 快照 IMG206 U2 --------------------
```

理解:
1. 在 mysql 服务器上创建MogileFS 数据库; 为防止swap颠簸,Vmware RAM至少为256M;
2. 在 storage上安装 mogilefs server;
3. 在 traker 或者 storage 上安装 管理软件 mogilefs utils;为防止swap颠簸,Vmware RAM至少为128M;
4. 在 IMG or? web 上 安装 mogilefs client
5. 最好在 traker上 运行 mogdbsetup 创数据库表单;
6. 在 traker或storage上运行 mogadm , 通过traker,将storage 加进节点和数据库

```
# HOWTO: linux mount ISO
mount -o loop /mnt/hgfs/share/PhaseII.fkoo.ISO /mnt/cdrom
mount -t iso9660 /mnt/hgfs/share/PhaseII.fkoo.ISO /mnt/cdrom -o loop
umount /mnt/cdrom/
```
2. 但在添加程序时系统可能提示不能安装，会出现“无法访问磁盘”的提示。
这时要执行以下步骤:
2.1 进入/dev/，删除cdrom，（最好先 #ls -l cdrom，记下当前/dev/cdrom的属性，可能是指向/dev/hda）
2.2 运行 #ln -s /dev/loop7 /dev/cdrom
2.3 运行 #losetup /dev/loop7 /****.iso
2.4 运行 #mount /mnt/cdrom
这样就可以通过 “添加 /删除 程序”来添加包，如安装内核源码，装vsftp等。
另外，假如要换盘，就执行#losetup -d /dev/loop7,然后重复2.3 和 2.4。
假如在装完相关包后，以后不再频繁需要iso，最好把/dev/cdrom改回原来的属性，（即刚开始ls -l的结果）