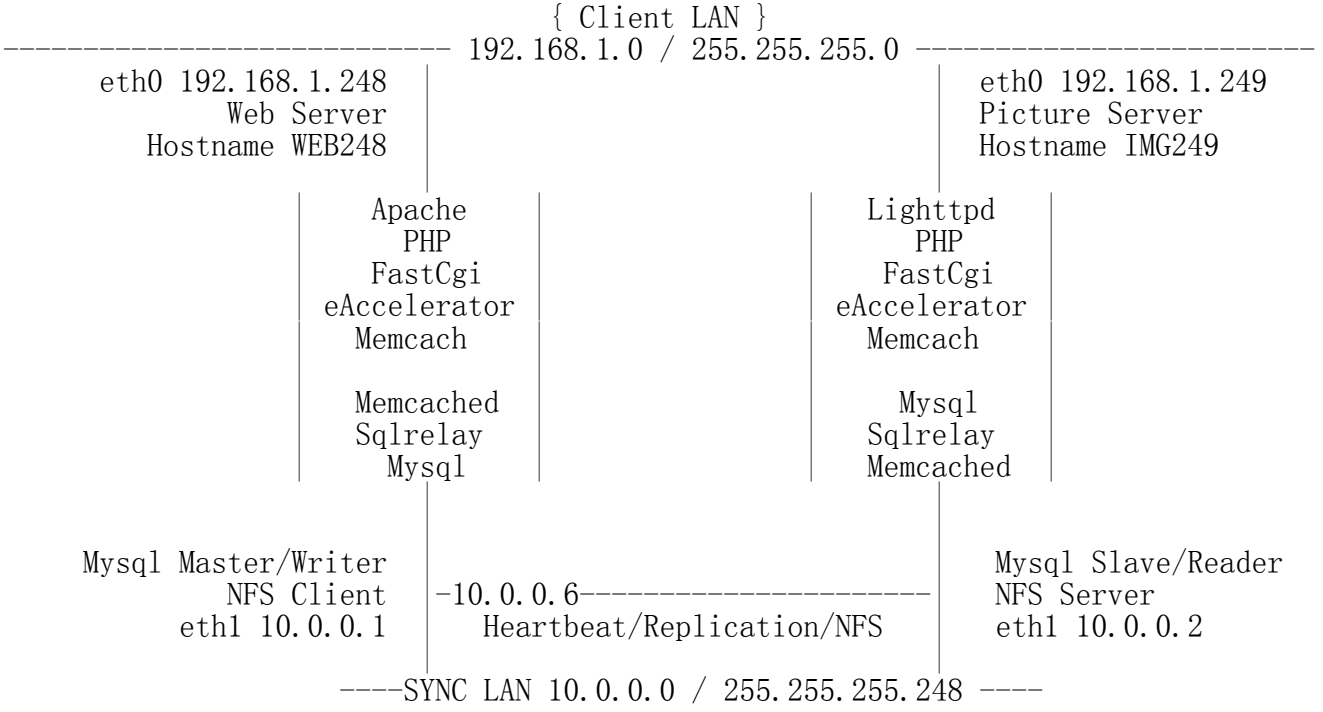
```
################################################
# File name   : PhaseI.fkoo.website.testing
# Description : fkoo一期网站系统架构虚拟实现
# Requirement : Vmware workstation *1 (含 VMware-Tools Linux.iso)
#               Windows XP PC *1
#               RHEL5.1 ISO光盘文件
#
# Copyright(C), fkoo, 2009, All Rights Reserved.
#
# Author: Far Young Chen / fkoo (fkoo.com@gmail.com)
# URL: http://www.fkoo.net
#
################################################

#################################################################################################
#                                        Network Layout
#################################################################################################
```




```
#-------------------- 初始化安装，VMware虚机名为：WEB248 --------------------
```

```
# 缩短启动选择菜单的等待时间为 1 秒
# (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
timeout=1
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"
service vmware-tools start
chkconfig vmware-tools on

# 查看系统时间
# date

#--------------------标记安装步骤: WEB248 init --------------------
# 安装mysql5.1.30 稳定版
cd /tmp
tar -zxvf  /mnt/hgfs/share/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
# vi /etc/profile
export PATH="$PATH:/usr/local/mysql/bin"
```

```
# 安装libunwind (64位需要安装,32位不用)
cd /tmp/
tar zxvf /mnt/hgfs/share/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/hgfs/share/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

# 定时校正服务器时钟，定时与中国国家授时中心授时服务器同步
# crontab -e
加入一行：
15 3 * * * /usr/sbin/ntpdate 210.72.145.44 > /dev/null 2>&1

# 安装openssl
tar xvfz /mnt/hgfs/share/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 ..
rm -rf openssl-0.9.8i*

# 安装 zlib
cd /tmp
tar zxvf /mnt/hgfs/share/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/hgfs/share/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*

# 安装 libjpeg
echo -n "Install libjpeg ..."
cd /tmp
tar zxvf /mnt/hgfs/share/jpegsrc.v6b.tar.gz
cd jpeg-6b
# 建立必须的目录 #
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/
# 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
```

```
echo -n "Install freetype ..."
cd /tmp
tar zxvf /mnt/hgfs/share/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/hgfs/share/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/hgfs/share/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/hgfs/share/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/hgfs/share/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*

# 添加 主机名-IP 的本地解析
# vi /etc/hosts
192.168.1.248           WEB248
10.0.0.1                WEB248
192.168.1.249           IMG249
10.0.0.2                IMG249

# -------------------- 标记安装步骤: WEB248 U1 --------------------

## 安装 apache #
cd /tmp
tar jxvf /mnt/hgfs/share/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
# 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/hgfs/share/httpd.apache /etc/rc.d/init.d/httpd
chmod 755 /etc/rc.d/init.d/httpd
# 添加 httpd 服务, 设定默认为关闭 ###
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

# -------------------- 标记安装步骤: WEB248 U2 --------------------

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

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

# 安装 php
cd /tmp
tar -jxvf /mnt/hgfs/share/php-5.2.8.tar.bz2
cd php-5.2.8
patch -p1 < /mnt/hgfs/share/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 # ########################
./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

# 生产环境需要加载的编译参数:
# --disable-debug  \

make
make install
cp php.ini-dist /usr/local/php-fcgi/etc/php.ini
cp /mnt/hgfs/share/phpinfo.php /mnt/hgfs/web/
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

# 安装 CGI/FastCGI - mod_fcgid 模块支持
cp /mnt/hgfs/share/mod_fcgid.2.2.gz /tmp/
cd /tmp
tar zxvf mod_fcgid.2.2.gz
cd mod_fcgid.2.2
make top_dir=/usr/local/apache
make install top_dir=/usr/local/apache
cd ..
rm -rf mod_fcgid.2.2*

# 编辑apache配置文档,支持 mod_fcgid
# vi /etc/httpd/httpd.conf
LoadModule fcgid_module modules/mod_fcgid.so

# 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>

<IfModule mod_fcgid.c>
        AddHandler fcgid-script .php .py .pl .fcgi
        SocketPath /tmp/fcgid.sock
        IdleTimeout 600
        ProcessLifeTime 3600
        MaxProcessCount 8
        DefaultMinClassProcessCount 3
        DefaultMaxClassProcessCount 3
        IPCConnectTimeout 20
        IPCCommTimeout 48
</IfModule>

<Directory "/usr/local/httpd/htdocs">
    # 注销掉
    #Options Indexes FollowSymLinks
    AllowOverride None
    Order allow,deny
    # 新插入下面3行
    FCGIWrapper /usr/local/php-fcgi/bin/php-cgi .php
    FCGIWrapper /usr/local/php-fcgi/bin/php-cgi .php5
    Options ExecCGI SymLinksIfOwnerMatch
    Allow from all
</Directory>

# 修改 httpd 主目录
DocumentRoot "/mnt/hgfs/web"
<Directory "/mnt/hgfs/web">

# 安装 libevent
cd /tmp/
tar vxzf /mnt/hgfs/share/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/hgfs/share/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 10.0.0.1 -d -p 62880 -u nobody -m 20
# 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 20MB 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/hgfs/share/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=10.0.0.1
        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/hgfs/share/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/hgfs/share/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"

# 安装SQL Relay
#安装Rudiments:
cd /tmp/
tar vxzf  /mnt/hgfs/share/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/hgfs/share/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

# 修改IP地址和路由转发参数
# rm -rf /etc/sysconfig/network-scripts/ifcfg-eth2
# vi /etc/sysctl.conf
net.ipv4.ip_forward = 0
# sysctl -p

# 修改防火墙设置
vi /etc/sysconfig/system-config-securitylevel
--enabled
--trust=eth1
--port=22:tcp
--port=80:tcp
```

```
--port=443:tcp

more /etc/sysconfig/iptables
more /etc/sysconfig/selinux

# -------------------- 标记安装步骤: WEB248 U3 --------------------

# 由 安装步骤标记: WEB248 U1 克隆生成 IMG249

# 修改主机名为 IMG249
# vi /etc/sysconfig/network
HOSTNAME=IMG249

# 修改IP地址和路由转发参数
# vi /etc/sysconfig/network-scripts/ifcfg-eth0
IPADDR=192.168.1.249
# vi /etc/sysconfig/network-scripts/ifcfg-eth1
IPADDR=10.0.0.2
# rm -rf /etc/sysconfig/network-scripts/ifcfg-eth2
# vi /etc/sysctl.conf
net.ipv4.ip_forward = 0
# sysctl -p

# 修改防火墙设置
vi /etc/sysconfig/system-config-securitylevel
--enabled
--trust=eth1
--port=22:tcp
--port=80:tcp
--port=443:tcp

more /etc/sysconfig/iptables

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

# -------------------- 标记安装步骤: IMG249 U1 --------------------

# 安装pcre库 (支持 lighttpd 或 nginx 的 rewrite 模块)
cd /tmp
tar zxvf /mnt/hgfs/share/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/hgfs/share/php-5.2.8.tar.bz2
cd php-5.2.8
patch -p1 < /mnt/hgfs/share/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/hgfs/share/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

# 安装 libevent
cd /tmp/
tar vxzf /mnt/hgfs/share/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/hgfs/share/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 10.0.0.2 -d -p 62880 -u nobody -m 20
# 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 20MB 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/hgfs/share/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/hgfs/share/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/hgfs/share/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"

# -------------------- 标记安装步骤: IMG249 U3 --------------------

# !!! 这段是废弃的配置, 仅作为存档 !!!
# 安装 lighttpd
tar jxvf /mnt/hgfs/share/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 \
--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

# 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 %u %t \"%r\" %s %b \"%{User-Agent}i\" \"%{Referer}i\""
# 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

# -------------------- 标记安装步骤: IMG249 U2 --------------------

# 在 安装步骤标记: IMG249 U3 基础上,
```

```
# 32位系统安装 gamin-devel
rpm -ivh /mnt/hgfs/share/gamin-devel-0.1.7-8.el5.i386.rpm

# 安装 lighttpd
tar jxvf /mnt/hgfs/share/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

# 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 %u %t \"%r\" %s %b \"%{User-Agent}i\" \"%{Referer}i\""
# 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"
        )
      )
    )
 )

# 使 php-cgi 能正常使用 SCRIPT_FILENAME 这个变量
# vi /usr/local/php-fcgi/etc/php.ini
cgi.fix_pathinfo=1

# 安装日志回滚
cd /tmp/
tar xvf /mnt/hgfs/share/cronolog-1.6.2.tar.tar
cd cronolog-1.6.2
./configure
make && make install
cd ..
rm -rf cronolog-1.6.2
# LogFormat "%h %l %u %t \"%r\" %>s %b" common

# vi /etc/lighttpd/lighttpd.conf
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"

# 优化 php-fpm
# 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>

# 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

# 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

# -------------------- 标记安装步骤: IMG249 U4 --------------------

# 在 安装步骤标记: WEB248 U3 的基础上,继续配置 mysql master/slave

# 建立数据库复制帐号 fkoocopy 和mon的监控用户 fkoo_monitor
# 复制和监控都是通过内网网卡的IP地址
# mysql -p
```

```
use mysql
grant replication slave on *.* to 'fkoocopy'@'10.0.0.2' identified by 'fkoopasswd';
grant select on *.* to 'fkoo_monitor'@'10.0.0.2' identified by 'FkooMonitor';
# 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='WEB248';

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

cp /etc/my.cnf /etc/my.cnf.bak
vi /etc/my.cnf
bind-address    = 10.0.0.1
# log-bin=mysql-bin
# server-id       = 1
server-id       = 2
log-bin=WEB248-bin
binlog-do-db=fkoodb
binlog-ignore-db = mysql
binlog-ignore-db = test
auto-increment-increment = 2
auto-increment-offset = 2

replicate-same-server-id = 0
master-host=10.0.0.2
master-user=fkoocopy
master-password=fkoopasswd
master-port=3306
master-connect-retry=60
report-host=WEB248
replicate-do-db=fkoodb
log-slave-updates
expire_logs_days = 10
max_binlog_size = 500M

service mysql restart
```

```
mysql -p

# 修正同步参数的步骤
# 根据 master status 修正 MASTER_LOG_FILE 和 MASTER_LOG_POS;  (以下数值仅供参考)
mysql -p
FLUSH TABLES WITH READ LOCK;
STOP SLAVE;
show master status;
unlock tables;
CHANGE MASTER TO
  MASTER_HOST='10.0.0.2',
  MASTER_USER='fkoocopy',
  MASTER_PASSWORD='fkoopasswd',
  MASTER_PORT=3306,
  MASTER_LOG_FILE='IMG249-bin.000001',
  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;
STOP SLAVE;
RESET MASTER;
RESET SLAVE;
SLAVE START;
show master status;
show slave status\G;

# -------------------- 标记安装步骤: WEB248 U4 --------------------

# 在 安装步骤标记:  IMG249 U4 的基础上,继续配置 mysql master/slave

# 建立数据库复制帐号 fkoocopy 和mon的监控用户 fkoo_monitor
# 复制和监控都是通过内网网卡的IP地址
# mysql -p
use mysql
grant replication slave on *.* to 'fkoocopy'@'10.0.0.1' identified by 'fkoopasswd';
grant select on *.* to 'fkoo_monitor'@'10.0.0.1' identified by 'FkooMonitor';
# 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='WEB248';

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

cp /etc/my.cnf /etc/my.cnf.bak
vi /etc/my.cnf
bind-address       = 10.0.0.2
log-bin=IMG249-bin
server-id       = 1

binlog-do-db=fkoodb
binlog-ignore-db = mysql
binlog-ignore-db = test
auto-increment-increment = 2
auto-increment-offset = 1

replicate-same-server-id = 0
master-host=10.0.0.1
master-user=fkoocopy
master-password=fkoopasswd
master-port=3306
master-connect-retry=60
report-host=IMG249
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
# 根据 同步关系对方的 master status 修正 MASTER_LOG_FILE 和 MASTER_LOG_POS; (以下数值仅供参考)
mysql -p
FLUSH TABLES WITH READ LOCK;
STOP SLAVE;
show master status;
unlock tables;
CHANGE MASTER TO
  MASTER_HOST='10.0.0.1',
```

```
  MASTER_USER='fkoocopy',
  MASTER_PASSWORD='fkoopasswd',
  MASTER_PORT=3306,
  MASTER_LOG_FILE='WEB248-bin.000002',
  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;

# -------------------- 标记安装步骤: IMG249 U5 --------------------

# 在 安装步骤标记: WEB248 U4 的基础上 (已建立好Replication),
# 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

# 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
# mysql -p
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;

# --------------------步骤标签:  WEB248 U4_1 --------------------

# 安装Mon
rpm -ivh /mnt/hgfs/share/perl-Time-Period-1.20-2.el5.rf.noarch.rpm
rpm -ivh /mnt/hgfs/share/perl-Net-SNPP-1.17-1.2.el5.rf.noarch.rpm
```

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

rpm -i /mnt/hgfs/share/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/hgfs/share/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 IMG249 的私网 IP;
# vi /etc/mon/mon.cf
### group definitions (hostnames or IP addresses)
hostgroup MasterDB 10.0.0.2

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 "service mysql start" >> /usr/lib/mon/alert.d/test.alert
echo "/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/hgfs/share/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
-------------------------------------------------------------------------------------------------------
  GROUP           SERVICE      STATUS      LAST       NEXT       ALERTS SUMMARY
R MasterDB        mysql        -           1s         3s         none

  GROUP           SERVICE      STATUS      LAST       NEXT       ALERTS SUMMARY
R MasterDB        mysql        FAIL        0s         0s         1      10.0.0.2

# 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/hgfs/share/perl-TimeDate-1.16-5.el5.noarch.rpm
rpm -ivh /mnt/hgfs/share/heartbeat-pils-2.1.4-2.1.i386.rpm
rpm -ivh /mnt/hgfs/share/heartbeat-stonith-2.1.4-2.1.i386.rpm
```

```
rpm -ivh /mnt/hgfs/share/heartbeat-2.1.4-2.1.i386.rpm
rpm -ivh /mnt/hgfs/share/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 "auth 1" >> /etc/ha.d/authkeys
echo "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
auto_failback off
node    IMG249
node    WEB248

# 配置 heartbeat 服务启动/关闭 的资源
# 设置 WEB248 为漂移地址 10.0.0.6 所在的默认的 master
echo "WEB248 10.0.0.6 " >> /etc/ha.d/haresources
tail /etc/ha.d/haresources

# 扩大 eth1 的掩码, 以便可以设置漂移地址
# vi /etc/sysconfig/network-scripts/ifcfg-eth1
BROADCAST=10.0.0.7
IPADDR=10.0.0.1
NETMASK=255.255.255.248
NETWORK=10.0.0.0
# 重启网络
```

```
service network restart

# 启动 heartbeat 服务
service heartbeat start

# -------------------- 标记安装步骤: WEB248 U5 --------------------

# 在 安装步骤标记: IMG249 U5 的基础上, 执行与 步骤标签:  WEB248 U4_1 相同的操作, 但需要做调整如下:

# 将 MasterDB 的 IP, 修改为 WEB248 的私网 IP;
# 说明: 在 hostgroup 和 watch 两行之间需要空一行;
# vi /etc/mon/mon.cf
### group definitions (hostnames or IP addresses)
hostgroup MasterDB 10.0.0.1

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

# 将 IPADDR 改为 IMG249 的私网IP
vi /etc/sysconfig/network-scripts/ifcfg-eth1
BROADCAST=10.0.0.7
IPADDR=10.0.0.2
NETMASK=255.255.255.248
NETWORK=10.0.0.0

# -------------------- 标记安装步骤: IMG249 U6 --------------------

# 在 安装步骤标记: IMG249 U6 的基础上

# 安装SQL Relay
#安装Rudiments:
cd /tmp/
tar vxzf  /mnt/hgfs/share/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/hgfs/share/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

# 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
# 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 10.0.0.0/30(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

# 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

# -------------------- 标记安装步骤: IMG249 U7 --------------------

# 在 安装步骤标记: WEB248 U5 的基础上

# NFS 客户端, 新建 mount点, 加载 NFS 服务端的目录;
#
mkdir /img
mount -t nfs -o timeo=3,vers=3,udp,hard,bg,rsize=16384,wsize=16384 10.0.0.2:/nfs /img

# 查看 nfs 信息
mount | grep nfs
nfsstat -v

# 卸载 nfs 目录
umount /img

# 在系统启动脚本 rc.local 中加入自动 mout 命令; 前面加上等待 10 秒,让系统网卡先连接上
echo -ne "
```

```
sleep 10
mount -t nfs -o timeo=3,vers=3,udp,hard,bg,rsize=16384,wsize=16384 10.0.0.2:/nfs /img
" >> /etc/rc.d/rc.local
cat /etc/rc.d/rc.local

# -------------------- 标记安装步骤: WEB248 U6 --------------------

# ----- 在 VMware 快照 IMG249 U7 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 WEB248 U6 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;

# ----- 继续在 IMG249 上,  安装 MogileFS

# 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*

# 向数据库安装 mogilefs_fk 数据库表单
# dbhost=10.0.0.6, 是 mysql M-H-S 结构中的 漂移地址
mogdbsetup --dbhost=10.0.0.6 --dbname=mogilefs_fk --dbuser=mogile_fk --dbpass=mogile_pw

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

# -------------------- 标记安装步骤: IMG249 U8 --------------------

# 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
```

```
# 生成存储节点配置文件
# server = lighttpd, 表示 mogstored 启用 lighttpd 作为 webdav
# httplisten = 10.0.0.2:7500, 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 = 10.0.0.2:7500
mgmtlisten = 10.0.0.2:7501
docroot = /var/mogdata
" >> /etc/mogilefs/mogstored.conf
cat /etc/mogilefs/mogstored.conf

# 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

# 生成 Tracker 配置文件
# listen = 10.0.0.2, 跟踪器使用 IMG249 的内网 IP
# db_dsn = 10.0.0.6, 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:10.0.0.6
db_user = mogile_fk
db_pass = mogile_pw
listen = 0.0.0.0:6001
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

# 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

# 取消执行 sudo 命令时需要终端的限制
vi /etc/sudoers
# Defaults    requiretty

# 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

# mogilefsd  与 syslog 有依存关系, 启动syslog服务
chkconfig syslog on
service syslog start

# 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

# 生成 mogadm 配置文件
echo -ne "trackers = 10.0.0.2:6001
" >> /etc/mogilefs/mogilefs.conf
cat /etc/mogilefs/mogilefs.conf

# 生成 mogtool 配置文件
echo -ne "trackers = 10.0.0.2:6001
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

# 用 mogadm 添加存储节点 Mog_IMG249
```

```
mogadm host add Mog_IMG249 --ip=10.0.0.2 --port=7500 --status=alive
# 列出存储节点
mogadm host list
# 向存储节点 Mog_IMG249 中添加编号为 1 的设备
mogadm device add Mog_IMG249 1
mogadm device add Mog_IMG249 2
mogadm device add Mog_IMG249 3
# 列出存储设备
mogadm device list
# 为存储设备 1 添加工作目录
mkdir -p /var/mogdata/dev1
mkdir -p /var/mogdata/dev2
mkdir -p /var/mogdata/dev3
# 监测存储系统
mogadm check
# 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=>['10.0.0.2:6001'], 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://10.0.0.2:7500/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'} = "10.0.0.6";
$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>

# -------------------- 标记安装步骤: IMG249 U9 --------------------

# 安装mogilefs客户端模块
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)

# -------------------- 标记安装步骤: IMG249 U10 --------------------

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

# 添加开机自启 sqlr-start 实例
# vi /etc/rc.local
/usr/local/sqlrelay/bin/sqlr-start -id usersdata
/usr/local/sqlrelay/bin/sqlr-start -id logsdata

# -------------------- 标记安装步骤: IMG249 U11 --------------------

# 在 WEB248 上安装 Mogilefs
# 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*

# 新建配置文件目录
mkdir /etc/mogilefs

# 生成 Tracker 配置文件
# listen = 127.0.0.1, 跟踪器使用 WEB248 的本地 IP
# db_dsn = 10.0.0.6, 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:10.0.0.6
db_user = mogile_fk
db_pass = mogile_pw
listen = 127.0.0.1:6001
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

# 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

# 取消执行 sudo 命令时需要终端的限制
vi /etc/sudoers
# Defaults    requiretty

# 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

# mogilefsd  与 syslog 有依存关系, 启动syslog服务
chkconfig syslog on
service syslog start

# 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

# 生成 mogadm 配置文件
echo -ne "trackers = 127.0.0.1:6001
" >> /etc/mogilefs/mogilefs.conf
```

```
cat /etc/mogilefs/mogilefs.conf

# 列出存储节点
mogadm host list
# 列出存储设备
mogadm device list
# 监测存储系统
mogadm check

# 安装mogilefs客户端模块
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)

# -------------------- 标记安装步骤: WEB248 U10 --------------------
[root@WEB248 ~]# cat /usr/local/sqlrelay/bin/sqlr-on
```

```
#!/bin/sh

export PATH=$PATH:/usr/local/sqlrelay/bin

/usr/local/sqlrelay/bin/sqlr-start -id usersdata
/usr/local/sqlrelay/bin/sqlr-start -id logsdata
/usr/local/sqlrelay/bin/sqlr-start -id albumsdata

[root@WEB248 ~]# chmod 777 /usr/local/sqlrelay/bin/sqlr-on

[root@WEB248 ~]# vi /etc/rc.local
sleep 120
service mogilefsd restart

sleep 2
# /usr/local/sqlrelay/bin/sqlr-start.sh
/usr/local/sqlrelay/bin/sqlr-on
```