Subject: Re: Howto create a desired distro template?
Posted by Gregor Mosheh on Tue, 20 Nov 2007 14:55:43 GMT
View Forum Message <> Reply to Message

Hi, Donatas.
I was faced with this same ordeal, as I am a OS creator/maintainer myself. Attached is our procedure, though you'll find that large parts of it don't apply to you (e.e. removing inetd and using xinetd instead, installing Nagios). Still, with this documentation and VMWare you should be on a good start as far as trimming down a VMWare machine running a OS, into a template cache.

Cheers.

--
Gregor Mosheh / Greg Allensworth
System Administrator, HostGIS cartographic development & hosting services
http://www.HostGIS.com/

"Remember that no one cares if you can back up,
only if you can restore." - AMANDA

Creating a new "Host Template Cache" for HostGIS Linux

This document describes how to use VMWare to create a new VM, install HostGIS Linux (HGL) on it and tweak the system into shape, and then create a Host Template Cache (a compressed VE image) for use in OpenVZ.

***** CREATING THE VM

Start by creating a new VM in VMWare. The stats can be minimal, and there is no need to create the entire disk at once during the setup.
* Create the disk as SCSI.

Then install HGL.
* Create a small partition at the end of the disk for swap. Some swap is technically necessary, but since you'll never in fact be using it, a few MB should be fine.
* Do set the timezone properly. The internal clock does not use UTC/GMT.
* Select the default mouse, but do NOT enable GPM at startup.
* Hostname: template Domain: internal.lan
* IP config: as appropriate for your LAN
* Nameserver: no

Reboot into your new HGL install.

Now we want to tweak it into a usable template.
Go ahead and login to the VM.

***** REPLACE INETD WITH XINETD

```
removepkg inetd
rm -f /etc/inetd.conf* /etc/rc.d/rc.inetd

cd /tmp
wget --header="Host: xinetd.org" http://204.152.188.37/xinetd-2.3.14.tar.gz
tar zxvf xinetd*.gz
cd xinetd*
./configure --prefix=/usr --sysconfdir=/etc
make && make install
mkdir /etc/xinetd.d
cat >> /etc/rc.d/rc.local <<EOF

# xinetd
/usr/sbin/xinetd
EOF
cat > /etc/xinetd.conf <<EOF
defaults
{
        log_type        = SYSLOG daemon notice
        log_on_success  = HOST EXIT DURATION
        log_on_failure  = HOST ATTEMPT
        instances       = 30
        cps             = 50 10
}
includedir /etc/xinetd.d
EOF
```

***** BASIC FILE SECURITY SETTINGS

```
# clear out old/dummy SSL certificates
mv /etc/ssl/openssl.cnf /tmp ; rm -r /etc/ssl/* ; mv /tmp/openssl.cnf /etc/ssl

# fix file permissions
find / -mount -nouser -exec chown root {} \; &
find / -mount -nogroup -exec chgrp root {} \; &
for i in \
   /bin/ping /bin/mount /bin/ping6 /bin/umount /usr/bin/rcp /usr/bin/rsh /usr/bin/chfn \
   /usr/bin/chsh /usr/bin/crontab /usr/bin/chage /usr/bin/traceroute6 /usr/bin/traceroute \
```

```
  /usr/bin/expiry /usr/bin/newgrp /usr/bin/passwd /usr/bin/gpasswd /usr/bin/rlogin \
  /usr/libexec/ssh-keysign /usr/libexec/pt_chown /usr/local/nagios/libexec/check_dhcp \
  /usr/local/nagios/libexec/check_icmp /usr/bin/wall /usr/bin/write
  do chmod u-s $i ; done

# fix Apache's configuration:
# add ServerTokens prod
# go to the htdocs Directory definition and change Indexes to -Indexes
# delete the entries for phpmyadmin and phppgadmin and tilecache
vi /etc/apache/httpd.conf

# keep FTP users chrooted:
echo "" >> /etc/proftpd.conf
echo "# keep all users chrooted to their homedir" >> /etc/proftpd.conf
echo "DefaultRoot ~" >> /etc/proftpd.conf

# allow the mailq to be checked by anybody:
chgrp smmsp /var/spool/mqueue
chmod g+rx /var/spool/mqueue

# disable the root and user accounts
# by changing the password for root and user to a ! character.
vi /etc/shadow
```

***** UPGRADES AND SECURITY PATCHES

The default HGL you used may require some software to be reinstalled, since new versions and critical bugfixes may have been released since that version of HGL was released. Follow these instructions, and also update them aas necessary for the appropriate versions and to remove paragraphs when a revision of HGL comes out that no longer requires them.

```
/etc/rc.d/rc.pgsql stop
cd /tmp
wget --passive-ftp
ftp://ftp.us.postgresql.org/pub/mirrors/postgresql/source/v8.2.4/postgresql-8.2.4.tar.bz2
tar jxvf postgresql-8.2.4.tar.bz2
cd postgresql-8.2.4
LDFLAGS=-lstdc++ ./configure --prefix=/usr --sysconfdir=/etc --localstatedir=/var \
      --with-perl --with-python --with-openssl \
      --enable-thread-safety --enable-integer-datetimes
make && make install

vi /var/lib/pgsql/postmaster.conf     # set stats_row_collector=on
```

***** NAGIOS: THE HEALTH-MONITORING SYSTEM

```
groupadd nagios
useradd -g nagios -d /usr/local/nagios -m nagios
echo "nrpe            5666/tcp  # Nagios NRPE" >> /etc/services

cd /tmp
wget http://superb-east.dl.sourceforge.net/sourceforge/nagiosplug/nagios-plugins-1.4.6.tar.gz
tar zxvf nagios-plugins-*.tar.gz ; cd nagios-plugins-*
./configure && make all && make install
cd /tmp
wget http://umn.dl.sourceforge.net/sourceforge/nagios/nrpe-2.6.tar.gz
tar zxvf nrpe-2.6.tar.gz ; cd nrpe-2.6
./configure && make && cp src/nrpe /usr/local/nagios/nrpe

for plugin in \
   check_wave check_users check_ups check_time check_tcp check_swap check_ssh
check_ssmtp \
   check_spop check_simap check_smtp check_sensors check_rpc check_real check_pop
check_ping \
   check_overcr check_oracle check_nwstat check_nt check_nntps check_nntp check_nagios \
   check_mysql_query check_mrtgtraf check_mrtg check_log check_jabber check_ircd \
   check_imap check_ifstatus check_ifoperstatus check_icmp check_http check_ftp check_flexlm \
   check_file_age check_dummy check_disk_smb check_dig check_dhcp check_clamd
check_by_ssh \
   check_breeze check_apt check_udp
do rm -f /usr/local/nagios/libexec/$plugin ; done

cat > /usr/local/nagios/nrpe.cfg <<EOF
# NRPE Config File
pid_file=/var/run/nrpe.pid
debug=0
command_timeout=60
connection_timeout=300

# And now the list of allowed check-commands:
command[check_disk]=/usr/local/nagios/libexec/check_disk -w 20% -c 10% -m /
command[check_dns]=/usr/local/nagios/libexec/check_dns www.google.com
command[check_load]=/usr/local/nagios/libexec/check_load -w 5,5,5 -c 8,8,8
command[check_mailq]=/usr/local/nagios/libexec/check_mailq -w 10 -c 20
command[check_mysql]=/usr/local/nagios/libexec/check_mysql -d gisdata -H localhost -u gisdata
-p password
command[check_pgsql]=/usr/local/nagios/libexec/check_pgsql -d gisdata -H localhost -l gisdata -p
password
command[check_ntp]=/usr/local/nagios/libexec/check_ntp -H pool.ntp.org
command[check_crond]=/usr/local/nagios/libexec/check_procs -u root -c 1: --command=crond
```

```
command[check_syslogd]=/usr/local/nagios/libexec/check_procs -u root -c 1:1
--command=syslogd
command[check_xinetd]=/usr/local/nagios/libexec/check_procs -u root -c 1:1 --command=xinetd
EOF

cat > /etc/xinetd.d/nrpe <<EOF
# description: NRPE for Nagios
service nrpe
{
   socket_type     = stream
   protocol        = tcp
   wait            = no
   user            = nagios
   server          = /usr/local/nagios/nrpe
   server_args     = -c /usr/local/nagios/nrpe.cfg --inetd
   only_from       = __HOSTIP__
}
EOF

chown -R nagios:nagios /usr/local/nagios
chmod -R o-rwx /usr/local/nagios
chmod go-rwx /etc/xinetd.d




***** OTHER UNNECESSARY STUFF

rm -rf /lib/modules /boot /dev/.udev /usr/doc /usr/info /media

cd /var/log/packages
for pkg in \
   hotplug-* hdparm-* devmapper-* udev-* usbutils-* pciutils-* module-init-tools-* \
   mdadm-* floppy-* lvm2-* phpMyAdmin-* phppgAdmin-* raidtools-* reiserfsprogs-* \
   smartmontools-* sysfsutils-* syslinux-* wireless_tools.* quota-* iptables-*
do removepkg $pkg ; done
# slackpkg doesn't work on 64-bit systems
removepkg slackpkg
rm -rf /etc/slackpkg

# prune init's getty
edit /etc/inittab and delete everything after entry l6 (runlevel 6)
init q

# clean out the fstab and mtab files
( cd /etc ; rm -f fstab mtab ; ln -s ../proc/mounts mtab )
echo "proc     /proc      proc     defaults    0  0" >> /etc/fstab
echo "devpts   /dev/pts   devpts   mode=0620   0  0" >> /etc/fstab
```

```
# the startup sequence and services, even the firewall
cd /etc/rc.d
rm -f rc.gpm-sample rc.hotplug rc.ip_forward rc.modules rc.scanluns  rc.serial rc.udev rc.sysvinit
rc.firewall
vi rc.syslog    # delete all mentions of klogd
vi rc.local     # delete smartd
vi rc.M         # delete the setterm entry
vi rc.S         # delete the MOTD clobbering

# blow away the network configuration with dummy strings for later replacement
#    replace the IP address with __IPADDRESS_
#    replace the netmask with __NETMASK__
#    replace the GATEWAY with __GATEWAY__
vi /etc/rc.d/rc.inet1.conf
```

***** REBOOTING

A VE cannot actually reboot, since there's no power switch to power-cycle the machine
after the VE has been shut down. OpenVZ emulates this effect with an external cronjob
called vpsreboot (see /etc/cron.d/vz). In order to reboot a VE that has been shut down
and which is expecting a reboot, the shutdown sequence must create a file named /reboot
in the VE's filesystem.

Also, the /etc/mtab file should point to /proc/mounts soo it can detect the / filesystem.

vi /etc/rc.d/rc.6
And add these two lines near the start:
```
# create the reboot flag so we get rebooted automatically
touch /reboot
```

vi /etc/rc.d/rc.M
And add these two lines near the start:
```
# replace the mtab file with a link to /proc/mounts so OpenVZ can find the / filesystem
rm -f /etc/mtab ;  ln -s /proc/mounts /etc/mtab
```

***** DELETING AND BLANKING SETTINGS

Lastly, you'll want to delete or blank out a bunch of files so they start fresh
when the VE is booted for its first time.

```
# stop all services
```

```
apachectl stop
killall syslogd klogd udevd crond
/etc/rc.d/rc.sendmail stop
/etc/webmin/stop
/etc/rc.d/rc.pgsql stop
/etc/rc.d/rc.mysqld stop
killall named proftpd
killall xinetd

# refresh the 'locate' cache
/etc/cron.daily/slocate

# blank out the system logfiles
for logfile in \
    /var/log/messages /var/log/syslog /var/log/debug /var/log/secure \
    /var/log/maillog /var/log/spooler /var/log/proftpd.log /var/log/xinetd.log \
    /var/log/dmesg /var/log/faillog /var/log/lastlog /var/log/wtmp \
    /var/log/apache/access_log /var/log/apache/error_log \
    /var/log/webmin/miniserv.error /var/log/webmin/miniserv.pid
do cp /dev/null $logfile ; done
rmdir /var/log/sa

# clear the SSH host key
rm -f /etc/ssh/ssh_host_*

# database server logfiles
rm -f /var/lib/mysql/*.err /var/lib/pgsql/logfile

# delete vi backup files, bash_history files, and so on
unset HISTFILE
find / -name '*~' -o -name .bash_history -o -name .gnupg -o -name .lesshst -o -name .viminfo -o
-name .rnd  -delete

# the junk under /tmp
rm -rf /tmp/*
```

***** CREATING THE VE CACHE IMAGE

A VE cache is just a tar.gz file of the entire filesystem. So creating them is simple!

```
tar zcvf /tmp/HostGIS_Linux_4.0_64bit.tar.gz --exclude='/sys/*' --exclude='/proc/*'
--exclude='/tmp/*' /
```

Ta-da! That's your new VE template cache. Just SFTP it to the VE server and you're all set!