VmgumpConfig

This page contains notes taken while installing a new Linux virtual machine in October 2016 - the new machine was named vmgump3 and later renamed to vmgump (which was the name of the old machine as well).

Puppet

Most of the configuration happens via Puppet, in particular via the module gump_server https://github.com/apache/infrastructure-puppet/tree/deployment/modules/gump_server - unfortunately a few manual steps are still required.

Installed packages

Currently installed packages are stored inside a svn repository that's only accessible to Gump PMC members. In the past we've checked it out manually and this has been done for now as well. A different option would be a role user but then we'd need to find a way to store the user's credentials securely.

```
~$ cd /srv/gump
/srv/gump$ sudo svn co https://svn.apache.org/repos/private/pmc/gump/packages
```

Set up database

Set a mysql password for root

```
mysqladmin -u root -p password "NEW-PASSWORD"
```

the initial password after installation is empty.

Create a gump database and user

```
~$ mysql -u root -p
mysql> CREATE DATABASE gump_public;
mysql> GRANT ALL PRIVILEGES ON gump_public.* TO 'gump'@'localhost' IDENTIFIED BY 'othernewpwd';
mysql> FLUSH PRIVILEGES;
mysql> EXIT;
~$ mysql -u gump -p gump_public
```

```
~$ sudo vi /root/.mysqlpass
```

dump old database on vmgump

Since we are migrating an installation, the database is created by restoring a database dump from the "old" vmgump.

```
~$ mkdir dump
~$ export currdate=`date +%Y%m%d`
~$ cd dump/
~/dump$ mysqldump -u root -p gump_public --add-drop-table --create-options > $currdate-gump_public.mysqldump
```

restore database on new vmgump2

copy over the dump

```
~$ mysql --user=root -p gump_public < $currdate-gump_public.mysqldump
```

---

everything below is obsolete but kept so the Puppet setup may explain itself better

This page contains notes taken while installing a new Linux virtual machine in August 2010 - the new machine was named vmgump2 and later renamed to vmgump (which was the name of the old machine as well).

vmgump runs Ubuntu Linux 10.4

Debian packages installed via apt-get

```
subversion cvs mercurial bzr git-core darcs openjdk-6-jdk nant autoconf automake curl unzip apache2 libtool
mysql-server mysql-client python-mysqldb mono-mcs g++ mailutils libexpat1-dev
```

This pulls in lots and lots of dependencies including X and Mono.

```
~$ java -version
java version "1.6.0_18"
OpenJDK Runtime Environment (IcedTea6 1.8) (6b18-1.8-4ubuntu3)
OpenJDK 64-Bit Server VM (build 16.0-b13, mixed mode)
```

Other dependencies

Build mvn repository proxy from http://svn.apache.org/repos/asf/gump/mvnrepo/tags/0.5/ using "ant zip" and copy the resulting repoproxy.zip to /tmp

```
~$ sudo mkdir -p /opt/__versions__
~$ cd /tmp
/tmp$ curl -O http://archive.apache.org/dist/maven/binaries/maven-1.0.2.tar.bz2
/tmp$ curl -O http://archive.apache.org/dist/maven/binaries/apache-maven-2.2.1-bin.tar.bz2
/tmp$ curl -O http://archive.apache.org/dist/maven/binaries/apache-maven-3.0.3-bin.tar.gz
/tmp$ cd /opt/__versions__
/opt/__versions__$ sudo tar xjf /tmp/maven-1.0.2.tar.bz2
/opt/__versions__$ sudo tar xjf /tmp/apache-maven-2.2.1-bin.tar.bz2
/opt/__versions__$ sudo tar xzf /tmp/apache-maven-3.0.3-bin.tar.gz
/opt/__versions__$ sudo unzip /tmp/repoproxy.zip
/opt/__versions__$ sudo mv lib repoproxy-0.5
/opt/__versions__$ rm /tmp/*.bz2
/opt/__versions__$ rm /tmp/*.zip
/opt/__versions__$ cd ..
/opt$ sudo ln -s __versions__/maven-1.0.2/ maven
/opt$ sudo ln -s __versions__/apache-maven-2.2.1/ maven2
/opt$ sudo ln -s __versions__/apache-maven-3.0.3/ maven3
/opt$ sudo ln -s __versions__/repoproxy-0.5/ repoproxy
```

create the Gump user

```
~$ sudo adduser gump
~$ sudo mkdir /home/gump
~$ sudo echo 'general@gump.apache.org' > ~gump/.forward
~$ sudo chown -R gump:gump /home/gump
~$ sudo chsh -s /bin/bash gump
```

create the Gump directory structure

```
~$ sudo mkdir -p /srv/gump/public
~$ sudo chown -R gump:gump /srv/gump/public
~$ cd /srv/gump
/srv/gump$ sudo svn co https://svn.apache.org/repos/private/pmc/gump/packages
/srv/gump$ cd public
/srv/gump/public$ sudo -u gump svn co https://svn.apache.org/repos/asf/gump/live/ gump
/srv/gump/public$ sudo -u gump mkdir -p workspace/log
```

create a testbed workspace

The testbase workspace runs all supported builders and scm updaters and can finish in less than half an hour, it is a good way to verify the installation. All files are created by user gump.

Create /srv/gump/public/gump/metadata/vmgump.xml

```
<?xml version="1.0" ?>
<workspace basedir="/srv/gump/public/workspace"
           pkgdir="/srv/gump/packages"
           version="0.4" name="vmgump-test">
  <sysproperty name="build.sysclasspath" value="only"/>
  <sysproperty name="java.awt.headless" value="true"/>
  <profile href="testbase/profile.xml"/>
</workspace>
```

Create /srv/gump/public/gump/cron/local-env-vmgump2.sh

here vmgump2 must match `hostname -s`

```
export GUMP_WORKSPACE=/srv/gump/public/gump/metadata/vmgump
export JAVA_HOME=/usr/lib/jvm/java-6-openjdk
export MAVEN_HOME=/opt/maven
export M2_HOME=/opt/maven2
export M3_HOME=/opt/maven3
export MVN_PROXY_HOME=/opt/repoproxy

export LANG=en_US.utf8
export PATH=$PATH:$MAVEN_HOME/bin:$M2_HOME/bin
```

run the test build

```
gump@vmgump:/srv/gump/public/gump/cron$ ./gump.sh
```

make the results world-visible

configure /etc/apache2/sites-available/vmgump.apache.org

```
NameVirtualHost *
<VirtualHost *>

        ServerAdmin private@gump.apache.org
        ServerName vmgump2.apache.org

        DocumentRoot /var/www/vmgump.apache.org

        <Directory />
                Options FollowSymLinks
                AllowOverride None
        </Directory>
        <Directory /var/www/vmgump.apache.org>
                Options Indexes FollowSymLinks MultiViews
                AllowOverride None
                Order allow,deny
                allow from all
        </Directory>

        ErrorLog /var/log/apache2/vmgump.apache.org.error.log

        # Possible values include: debug, info, notice, warn, error, crit,
        # alert, emerg.
        LogLevel warn

        CustomLog /var/log/apache2/vmgump.apache.org.access.log combined
        ServerSignature On

        Alias /gump/public/ /srv/gump/public/workspace/log/

        <Directory /srv/gump/>
            HeaderName /disclaimer.html
            Options Indexes MultiViews FollowSymLinks
            AllowOverride None
            Order allow,deny
            Allow from all
        </Directory>

        <Location /gump/public/workspace_defn.html>
            Order deny,allow
            Deny from all
        </Location>
</VirtualHost>
```

```
~$ sudo mkdir /var/www/vmgump.apache.org
~$ sudo chown gump:gump /var/www/vmgump.apache.org
~$ sudo a2ensite vmgump.apache.org
~$ sudo a2dissite default
~$ sudo /etc/init.d/apache2 reload
```

Set up database

A root password has been set during installation, the Ubuntu packages don't install anonymous users or test dbs, so http://dev.mysql.com/doc/refman/5.1/en/default-privileges.html is satisfied.

Create a gump database and user

```
~$ mysql -u root -p
mysql> CREATE DATABASE gump_public;
mysql> GRANT ALL PRIVILEGES ON gump_public.* TO 'gump'@'localhost' IDENTIFIED BY PASSWORD('othernewpwd');
mysql> FLUSH PRIVILEGES;
mysql> EXIT;
~$ mysql -u gump -p gump_public
```

```
~$ sudo vi /root/.mysqlpass
```

Since we are migrating an installation, the database is created by restoring a database dump from the "old" vmgump.

dump old database on vmgump

```
~# mkdir dump
~# umask 277
~# export currdate=`date +%Y%m%d`
~# cd dump/
~/dump# mysqldump -u root -p gump_public --add-drop-table --create-options > $currdate-gump_public.mysqldump
```

restore database on new vmgump2

copy over the dump

```
~$ mysql --user=root -p gump_public < $currdate-gump_public.mysqldump
```

Alternatively: create database from scratch

This is for a new machine rather than a migration

```
$ cd /srv/gump/public/gump/mysql
$ mysql -u gump -p gump_public < gump.sql
```

re-run testbed

add a database element with the gump user password to `/srv/gump/public/gump/metadata/vmgump.xml`

```
gump@vmgump:~$ cd /srv/gump/public/gump/cron/
gump@vmgump:/srv/gump/public/gump/cron$ ./gump.sh
```

make it a real Gump instance

switch to the gump profile

edit `/srv/gump/public/gump/metadata/vmgump.xml` so the `profile` element now reads

```
  <profile href="profile/gump.xml"/>
```

set up crontab for user gump

```
MAILTO=""

# Public - these are subruns of public that don't send email but update the web site
0 0,6,12,18 * * *    cd /srv/gump/public/gump/cron; /bin/bash gump.sh all

# Clean up older artifacts
0 0 * * * /usr/bin/find /srv/gump/public/workspace/repo -type f -ctime +6 | /usr/bin/xargs -r /bin/rm > /dev
/null 2>&1

#Clean up after POI and other tests
0 0 * * * /usr/bin/find /tmp -type f -ctime +6 | /usr/bin/xargs -r /bin/rm > /dev/null 2>&1
```

wait for the next run to start.

help Maven 1.x

some Maven 1.x plugins need to be downloaded and won't because Gump runs maven with the --offline switch. After the first Gump run has finished, do as user gump:

```
$ export JAVA_HOME=/usr/lib/jvm/java-6-openjdk
$ cd /srv/gump/public/workspace/village/
/srv/gump/public/workspace/village$  /opt/maven/bin/maven jar
$ cd /srv/gump/public/workspace/commons-dormant/test/
/srv/gump/public/workspace/commons-dormant/test$ /opt/maven/bin/maven jar
```

help CVS

sometimes Gump fails to populate ~/.cvspass properly. As user gump:

```
~$ cvs -q -z3 -d :pserver:anonymous@cvs.extreme.indiana.edu:/l/extreme/cvspub login
Logging in to :pserver:anonymous@cvs.extreme.indiana.edu:2401/l/extreme/cvspub
CVS password: cvsanon
~$ cvs -q -z3 -d :pserver:anonymous@cvs.jdom.org:/home/cvspublic login
Logging in to :pserver:anonymous@cvs.jdom.org:2401/home/cvspublic
CVS password: anonymous
$ cvs -q -z3 -d :pserver:anonymous@www.jacorb.org:/web/www.jacorb.org/cvs/jacorb login
Logging in to :pserver:anonymous@www.jacorb.org:2401/web/www.jacorb.org/cvs/jacorb
CVS password: anonymous
$ cvs -q -z3 -d :pserver:anonymous@invicta.cvs.sourceforge.net:/cvsroot/invicta login
Logging in to :pserver:anonymous@invicta.cvs.sourceforge.net:2401/cvsroot/invicta
CVS password:
```

help git

github doesn't like anonymous users so as user "gump" run

gump@vmgump:$ git config --global user.email "general@gump.apache.org" gump@vmgump:$ git config --global user.name "Apache Gump"

Clean up after Surefire

At least with the versions we use sometimes Surefire leaves java processes around when it times out tests. The included post-run script kills them once Gump is done. It is not the most sophisticated script but does what it is supposed to do, as long as there is only one Gump instance running at a time.

Create `/srv/gump/public/gump/cron/local-post-run-vmgump.sh`

```
#!/bin/sh

free
ps -C java -f -w -w | fgrep surefire
for i in `ps -C java -o pid,command -w -w | fgrep surefire | cut -c 1-5`; do
    echo "Killing $i"
    kill -9 $i
done
free
```

and make it executable.

make Gump send out mails

Only one machine should be sending nag mails. As of 2010-08-09 this is vmgump2

get ready to send email

as root

```
~# echo 'private@gump.apache.org' > ~root/.forward
~# mail yourself@somewhere
```

assuming this worked and a reply to the testmail reached the PMC list, email should be working (at least for @apache.org addresses).

enable nagging but send everything to yourself

edit `/srv/gump/public/gump/metadata/vmgump.xml` modify the <workspace> element and and add <nag> element

```
<workspace basedir="/srv/gump/public/workspace"
           pkgdir="/srv/gump/packages"
           version="0.4"

           name="vmgump"

           logurl="http://vmgump2.apache.org/gump/public"
           mailserver="localhost"
           administrator="general@gump.apache.org"
           email="gump@vmgump2.apache.org"
           >
  ...
  <nag to="yourself@somewhere"/>
```

modify crontab so the gump runs are "official" and send mails. Later only one run a day will be marked official but for now - nagging goes to yourself - it can be all of them

```
0 0,6,12,18 * * *    cd /srv/gump/public/gump/cron; /bin/bash gump.sh all --official
```

make the new machine the one sending out official nags

Once satisfied with the result, you can remove the *to* attribute from the nag element in `/srv/gump/public/gump/metadata/vmgump.xml`. If there is any other machine sending out nag mails, you should now remove the nag element fro m its workspace definition completely.

As of August 2010 a complete Gump run takes between 9 and 13 hours, this means two subsequent build times like the ones startung at 0am and 6am will never both be run at the same time, but at least one of the two should be running each day. Mark both of them as *official*, where official Gump builds are those that send reminder nag mails.

Edit gump's crontab

```
# Public and official (send reminder nag mails)  only one of them will really run each day since a full run
takes more than six hours
0 0,6 * * * cd /srv/gump/public/gump/cron; /bin/bash gump.sh all --official

# Public - these are subruns of public that don't send email but update the web site
0 12,18 * * *    cd /srv/gump/public/gump/cron; /bin/bash gump.sh all
```

Rename vmgump2 -> vmgump

```
# hostname vmgump
# mv /srv/gump/public/gump/cron/local-env-vmgump2.sh /srv/gump/public/gump/cron/local-env-vmgump.sh
```

edit `/etc/apache2/sites-available/vmgump.apache.org` and change ServerName to vmgump.apache.org

```
$ sudo /etc/init.d/apache2 reload
```

edit `/srv/gump/public/gump/metadata/vmgump.xml` and change <workspace>'s *logurl* and *email* attributes.

edit `/etc/postfix/main.cf` and change myhostname and mydestination

```
$ sudo newaliases
$ sudo /etc/init.d/postfix reload
```

everything below this is old information

gump3 setup

```
useradd -d /home/gump3 -s /bin/bash gump3
addgroup gump3
adduser gump3 gump3
adduser gump3 staff
mkdir /home/gump3
chown -Rf gump3:gump3 /home/gump3
passwd gump3
$EDITOR ~gump3/.passwd # save password
chmod 600 ~gump3/.passwd
chown gump3:gump3 ~gump3/.passwd
su - gump3 # do the below as "gump3" user in /home/gump3...
svn co https://svn.apache.org/repos/asf/gump/branches/Gump3
echo 'general@gump.apache.org' > ~gump3/.forward
```

- disable auth caching for svn, editing ~gump3/.subversion/config so that

```
### Section for authentication and authorization customizations.
[auth]
### Set store-passwords to 'no' to avoid storing passwords in the
### auth/ area of your config directory.  It defaults to 'yes'.
### Note that this option only prevents saving of *new* passwords;
### it doesn't invalidate existing passwords.  (To do that, remove
### the cache files by hand as described in the Subversion book.)
store-passwords = no
### Set store-auth-creds to 'no' to avoid storing any subversion
### credentials in the auth/ area of your config directory.
### It defaults to 'yes'.  Note that this option only prevents
### saving of *new* credentials;  it doesn't invalidate existing
### caches.  (To do that, remove the cache files by hand.)
store-auth-creds = no
```

- [VmgumpConfig/Gump3BashConfig]
- install gump3 prereqs

```
exit # fromp gump3, back into su mode
apt-get install python2.4 python2.4-mysqldb python-pmock
# no python2.4-pmock...
cp /usr/lib/python2.3/site-packages/pmock.py /usr/lib/python2.4/site-packages
python2.4 -OO -c "from pmock import *; import pmock; import sys; sys.exit()"
```

- test it

```
su - gump3
GUMP_TEST_NO_MYSQL=yes gump test
```

Done!

Get to business...

- su into gump, and run some tests:
  {`cd /srv/gump/public/gump/python export PYTHONPATH={{pwd`
  python gump/check.py -w ../brutus.xml all --debug
  }}}
- make sure we trust svn.apache.org `cd /srv/gump/public/gump; svn up` (hit 'p' to store cert)
- now kick of an "actual run" manually:

`cd /srv/gump/public/gump/cron; /bin/bash gump.sh all`

Gump Farm Layout Details

Current structure is:

File System:

```
    /srv/gump -- root
    /srv/gump/packages -- shared packages

    /srv/gump/{flavour} -- e.g. public or jdk15 or test or ...
    /srv/gump/{flavour}/gump -- Installation of Gump
    /srv/gump/{flavour}/workspace -- working area
    /srv/gump/{flavour}/results -- WWW site
    /srv/gump/{flavour}/jars -- Artifact Repository

    /opt -- prereqs not installed using apt
    /var/www/vmgump.apache.org -- website
```

HTTP:

```
    http://vmgump.apache.org/gump/{flavour} ->
        /srv/gump/{flavour}/results -- WWW site

    http://vmgump.apache.org/gump/{flavour}-jars/ ->
        /srv/gump/{flavour}/jars -- Artifact Repository
```

Currently deployed flavours:

```
    public -- against SVN /live, does e-mail notification, automatic runs.

    test -- against SVN /trunk, manual runs. Often cleaned out to save disk space.
```