Standard Server Build Script

**Scope**: Concept and notes on the development of the standard server platform and scripts

**Audience**: Development staff

**Short link:**  <http://goo.gl/FR1iA>

**Document Owner**: Bill

**Document started** by Bill, Aug 2012

**Last update** see File->See Revision Hx

[Related documents](#_43q41mle53qq)

[Goals of build script](#_7oj2dy9qh57t)

[Use cases](#_62ddwfrymtp2)

[Instructions to Gifford... Aug 9](#_pkm4kbvjz38u)

[Initial focus on Step 2 - Aug 15](#_6feyvpvvzj8)

[Four steps to build a server - Aug 15](#_sdx0bk1jjbxn)

[Server build script repository](#_4snhqnie6w8l)

[Instructions](#_f5kxyhoibei6)

[Status](#_1wosarl888q)

[To Do](#_d64nqrftgk3c)

[Java](#_nltvv2xh0p1u)

[Gifford’s Notes](#_xw30on9uttym)

[Bill’s Priorities/ToDo items - Aug 23](#_7i9kqlcjmpvu)

[Use and dissemination](#_oc2vblt557hp)

[Script structure and function](#_2fd9qmdjt18)

[Test Run - creating an instance for Darius - Aug 23](#_o3riq45qlyfa)

[Gifford’s in-progress version of the puppet openmrs script](#_gup4keyde0i7)

[Meeting 2012-08-23](#_66z32k1sxfg6)

[Version string](#_qo5b9ywy49jp)

[Goal is 2012.1  
  
Darius’s Notes from 9/2](#_ksxg8snsmra9)

[Anikate and Darius upgrading the 157 server with a new release](#_wgvma5ehlyvx)

[MySQL puppet issues](#_ngizcz85ooci)

[KenyaEMR documentation of process (Bill, Anikate, Oct 28)](#_vyi4u57maxwa)

[Step 1](#_tk1s8vvajnfi)

[Step 2](#_r3cbqdrw98az)

[Step 3 (Application install)](#_hdclwpi1x1xp)

[Step 4 (Configuration)](#_naestxtexy7i)

[SWIM LANE](#_7kpawf3h3vp)

## Related documents

High level discussion of what’s in a server -

<https://docs.google.com/document/d/1QIVMYaCuXyTupwkm8_2_lQMxVZQcAwxgK_WRe9F9H0Y/edit>

Public documentation of server build process -

<https://docs.google.com/document/d/1_BfcLhVfYHsv0DWtBYk9WzOYFAgWcPAj3XtRWEmZnrE/edit>

Roadmap for build script - <https://docs.google.com/document/d/1jwWrXA3IUeXsUE7WbBsVZf09auzpg8dNfV67siO3V04/edit#>

Server confgiurations - discussion of dev, nightly build/ci, demo, test, and prod needs -

<https://docs.google.com/document/d/1ehT1cxg3sCUFD2T3-HjdIvQFydz26W7T6myjBR3Kixc/edit>

## Goals of build script

The initial goal of this process is a script that will address step two, below, by installing basic system components on a core distribution. The core distribution is specified in step one.

the goal of having such a script is that it will be modified to create successive, versioned, variations on the build process to create a standardized server deployment platform for virtual machines are native installs.

the first version of the script should install Apache, mod JK, and tomcat.

a successive build might also install open VPN and scripts or configuration to create a VPN connection to one of our servers. another reason for successive builds may be that either the versions of specific system components may change, or the versions of the underlying core may change and require different system components or configuration.

an example of a variation might be a parameter controlled option to install mySQL, postgreSQL, or both, depending on the intended application. I don’t know if this is the case with system components like Apache or tomcat, but it may be that some system components would care about the number of CPUs on the target system, whether native or virtual. in that case the script may need to either discover some characteristics about the “hardware ” or accept those characteristics as arguments

puppet modules need to log at a high level all puppet installed components. this puppet log should always be available on a link at https://<VM\_IP\_address>:443/ - namely the root context. All other applications should live in a named context, or if necessary (for historical reasons), behind a virtual host name, not the raw IP address.

## Use cases

the intended users of this script are system administrators or developers who need to create a standardized build for one of a variety of different appliance applications. That may be openELIS, openMRS./Kenya EMR, iSante, gossamer, etc.

the intended use is for a user like to Svend, Jim, Paul V., Bill, or Sam Kanga to create a machine for themselves by obtaining an iso or vm export of turnkey core 11.3, and either importing the image as a new VM or installing the iso as either a native or virtual machine install.

if a developer creates this machine, or if thestep three install scripts for specific applications gets better, then it’s likely the machine would be created in quantity one, and the created instance would become the target instance for an application install. This is likely the case for demo and test instances. However, most commonly, I would anticipate that we would create a new platform VM when there is something substantial that changes in the script, export that platform VM to an OVA or OVF file, and import that image to create new individual instances. this would be the case for prod instances.

## Instructions to Gifford... Aug 9

We need a standard linux server configuration, buildable by a script, on which to install our applications (whether nightly build, demo, test, production, or whatever). Eventually, I’d like this standard server to have a range of features, as described here: <https://docs.google.com/document/d/1QIVMYaCuXyTupwkm8_2_lQMxVZQcAwxgK_WRe9F9H0Y/edit#heading=h.b94leftcnl6q>

This script should be something that can be run on a basic installed linux system, whether a native install or a vm install.

I think the following steps are the correct ones with which to start:

1. Check w/ Svend to see if he already has a build script that will work
2. Start w/ Turnkey Core VM <http://www.turnkeylinux.org/core>. Use release 11.3.
3. Ask Darius what versions of Apache, mod\_jk, and Tomcat to install (also MySQL, java, etc...)
   1. I found this comment on the Tomcat page of the install instructions: There are issues with versions of Tomcat later than 6.0.29 that have yet to be resolved. Installation through a package manager is not recommended as this will likely install a later version (Turnkey Core 11.3 is based on Ubuntu 10.04, which includes tomcat 6.0.24, so it should be OK. - Svend)
   2. The instructions are at <https://wiki.openmrs.org/display/docs/Installing+OpenMRS>
4. Install above software configured such that OpenMRS can be then installed on that server. I don’t think OpenMRS needs anything more - Darius would know. As you look at versions, consider whether those versions will also work with OpenELIS.
5. Document what you're doing in the script

That's it for now.

## Initial focus on Step 2 - Aug 15

Gifford and I talked a bit this evening, and I asked him to first concentrate on the System Components step (Step 2) of the four step process.

I think the first step should be “manual” – the installer deploys a TK Core 11.3 image using whatever method they prefer: vm, native install, amazon cloud, etc.

The second step should first be implemented as a script that can be run manually by the root user, who will obviously also have to get that script to the new machine (scp, paste into a “cat” command, something more clever – who cares – it’s the user’s choice how they want to do that).

Svend said he might later roll that script into Puppet to facilitate widespread deployment, but 1) we have to develop and test the sequence of steps for System Components first, and 2) one good use for that script would be to add system components to a TK core vm, after which the next two steps (KenyaEMR Install, Purpose Configuration) could be done, and at that point, the vm could be exported as a new image to be duplicated to create production servers. So, even a script that was only executed on a single vm image could be very useful.

## Four steps to build a server - Aug 15

I think separating those four steps, and describing the interface between each will help us separate responsibilities and ease development and maintenance.

for instance, the interface between System Components and KenyaEMR Install is addressed by the following question:

What does System Components look like when done? Ans: Specific versions of packaged distributions are installed and configured. Modifications are made to TurnKey core configuration. Nothing else is installed.)

Step 1 O/S base install (manual process)

=============

Start w/ TK Core 11.3

Deploy as best suits the situation (VM, native install, Amazon Cloud, CIRG KVM, etc...)

Get a shell prompt as root...

Step 2 System Components (script)

=====================

(need to decide where to store the “version number” for the built system – likely the version number of the script plus any specific configuration params passed to it, like “MySQL” or “postgres”

(configure turnkey core installed items, like blue screen/console, webmin, etc.) - ignore this for now.

Install "standard" components, presumably from a package manager, specifying versions

Example:

> Package: apache2

> Version: 2.2.14-5ubuntu8.8

> Package: tomcat6

> Version: 6.0.24-2ubuntu1.10 (use 6.0.29)

> Package: libapache2-mod-jk

> Version: 1:1.2.28-2

Also, Java if not installed as part of core (prior to tomcat), SSL, PHPMyAdmin (or PostGres equiv), etc

**Start w/ the minimum set up components we need for KenyaEMR, and eventually for other apps.**

[Kenya required Packages document](https://docs.google.com/document/d/1SG5Ux89vWiFfaZwhtOeCMi2YYjsITZBNA_iId33eJBU/edit)

(eventually add OpenVPN, etc., in subsequent revisions

Any required configuration steps for

Step 3 KenyaEMR Install

==============

Darius knows best what this means, but what I think it means is

OMRS 1.9.something

A specific set of modules, in some cases versions that may not be easily available from the respository

The KenyaEMR module

The MetadataDistributionModule

Run the MDM to get the concept dictionary, forms, and other stuff

Step 4 Purpose Configuration

==================

Depending on the specific purpose, install a demo, or test, or clean database

Possibly install scripts to refresh the database at some interval (nightly?)

Possibly convert existing data from another installation

## Server build script repository

<https://github.com/cirg/appliance-setup>

### Instructions

1. Follow instructions in README.md to setup the appliance.

2. Open <http://localhost:8080/openmrs> in a web browser

3. Run the OpenMRS setup. The MySQL root account password is UP9aeKas.

### Status

* Installs firefox, java, tomcat, mysql, apache2
* (REMOVED) OpenMRS is automatically downloaded and deployed under Tomcat. OpenMRS URL is <http://localhost:8080/openmrs>

### To Do

* Set up Apache/Tomcat proxy
* Set up Apache SSL

### Java

Ubuntu 10.04 has removed sun-java6 from its repositories. Several people have had success with the openjdk package, which is the default in Ubuntu:

<http://listarchives.openmrs.org/OpenJDK-and-OpenMRS-td7236251.html>

## **Gifford’s** Notes

Tuesday 8/21

- Ran Svend’s puppet script

- Suggestion: Make the script more verbose when running puppet

- Suggestion: Make the script more interactive (allowing dry runs, confirmation checks)

- Suggestion: install “locate” to make it easier for (me) to find my way around the operating system

- Problem [resolved]: openmrs wants access to /usr/share/tomcat6 should this be chowned and chgrped to tomcat6? (For now: solve this by letting the puppet script install openMRS and manage the chown/chgrp instead of by doing it manually according to the openmrs directions. The issue is that any apt-get update will reappropriate the ownership of everything else in the directory to root. For more information: <https://wiki.openmrs.org/display/docs/Application+Data+Directory> )

- Downloaded openmrs.war file from <http://openmrs.org/download/>

- Copied war file to:/var/lib/tomcat6/webapps

NOT: /usr/share/tomcat6/webapps

- wait for about 30 seconds for tomcat to auto deploy the war file (this can be confirmed by checking to see if an openmrs directory has been created in webapps)

- configure openmrs via: <http://xx.xx.xx.xx:8080/openmrs/>

-Tuesday 8/21: trying the “simple installation”

- using the mysql root password inside opt/appliance-setup/puppet/site.pp

(e.g. UP9aeKas )

- admin/Admin123 is the username/password for the demo

- + demodata yes

## Bill’s Priorities/ToDo items - Aug 23

These are combined from Aug 22 notes, Aug 22 late night email, and Aug 23 chat w/ Svend

### Use and dissemination

1. **how can Bill (as a test case) run the puppet process** to make a VM instance of TK core 11.3 w/ these system components?
2. Darius needs a new VM for Kenya asap (today?)
3. Create a public document explaining the process, how to access the scripts through git, how to use system. target audience: OpenMRS developer community. Location: <https://docs.google.com/document/d/1_BfcLhVfYHsv0DWtBYk9WzOYFAgWcPAj3XtRWEmZnrE/edit>

### Script structure and function

* Ensure there’s a clean split between scripts to run Step 2 and scripts to run Step 3. The link between these two processes should not (yet) be automated.
* (Eventually) supply arguments to produce (for instance) a MySQL version and a PostgreSQL version and a “both” version.
* Store a “version string” in the root directory of the created instance as part of the Step 2 process. If you think it should be stored somewhere else, that’s fine - it should have permissions set so it’s pretty easily readable. The “string” can be a small text file with multiple lines. The format of the string should be the following items, perhaps with one line for the first four bullets and one line per bullet for the other
  + A short name based on our version scheme
    - YYYY.planned release.unplanned release [options summary]
    - e.g. 2012.1 [MySQL]
  + Date/time instance was created (date/time script was run)
  + Version of script used (svend says this is the version control tag from git)
  + Any arguments passed to the script to control it’s function
  + The names and versions of any components installed as part of step 2
  + Brief descriptions of any server configuration actions taken as part of step 2
  + (these last two items may correspond to individual Puppet modules, in which case perhaps each module can append a “description line” of what it did to the version string file, after that module finishes its work).
* For the first Darius instance, you can create this file by hand once the instance is installed

## Test Run - creating an instance for Darius - Aug 23

(Svend and Gifford)

[9:35:34 AM] Bill Lober: A good use case for the puppet script just came up. We need a new VM instance (not a "template" VM that we'd use to create an OVA for rapid duplication/distribution) but a single instance.

[9:36:23 AM] Svend Sorensen: OK

[9:36:42 AM] Bill Lober: Yesterday, I asked Gifford to do a couple things 1) make sure I could use the script to create an instance 2) figure out a good scheme to create a version string for the instance and 3) make sure the script was separable between step 2 and step 3

[9:38:31 AM] Svend Sorensen: Most projects use a version control tag for the version string.

[9:38:40 AM] Bill Lober: I added another item in the email, I think - create a public page outlining what we're trying todo, linking to github, and providing some basic instructions. The idea would be to explain and share this process more broadly w/ OpenMRS developers group

I've combined all the requests into the "Bill’s Priorities/ToDo items - Aug 23" section of this document.

[10:27:33 AM] Bill Lober: Svend - darius needs a new VM for KE EMR - not PHI. I'd like to use this new process to build it. Where would you want to host it? (KVM machine, Vbox host (mac, laptop, other?) )

[10:29:30 AM] Svend Sorensen: Cumulus (KVM) would be the easiest place for me to set it up. Amazon EC2 would be another option.

[10:31:21 AM] Svend Sorensen: What is KE EMR, and what are the requirements?

[10:36:13 AM] Bill Lober: what I want is a base install, as per the puppet process, on TK core 11.3 w/ only the step 2 components installed.

[10:36:22 AM] Bill Lober: that's what Gifford did yesterday, right?

[10:37:45 AM] Gifford Cheung: KE EMR == Kenya EMR

[10:38:17 AM] Svend Sorensen: That's right.

[10:38:41 AM] Bill Lober: Cumulus sounds better than EC2.

[10:41:39 AM] Bill Lober: Between you and/or gifford - create the instance using Puppet process and document what you did here: <https://docs.google.com/document/d/1K6qeYgU28ys13oAqNXjUFtDQWdqYluWtq5gbA23jK6w/edit#heading=h.jo1j5fkchptu>

This is **Version 2012.1**This is CIRG Appliance Setup version [f93cccfc42](https://github.com/cirg/appliance-setup/tree/f93cccfc42f246ac7d4ace5e9b4750819c2519d3)

Steps to use this process on Cumulus to get to the end of Step 2 were:

======

Note, this first part (“Step 1”) will differ depending on where you are deploying, but the goal is to get to a root prompt on a newly installed, unmodified, TK Core 11.3 system

======

1. Download Turnkey Core 11.3, (VMDK file)
2. Copy vmdk file to /var/lib/libvirt/images (on cumulus)
3. Convert VMDK to a RAW format
   1. qemu-img convert -O raw \  
      turnkey-core-11.3-lucid-x86.vmdk \  
      turnkey-core-11.3-lucid-x86.raw
4. Copy raw image file to /var/lib/libvirt/images
   1. sudo cp turnkey-core-11.3-lucid-x86.raw \  
      /var/lib/libvirt/images/kenya-emr.img
5. Instantiate VM
   1. virt-install --connect qemu:///system \  
      --name kenya-emr --ram 512 --import \  
      --disk /var/lib/libvirt/images/kenya-emr.img,size=10 \  
      --vnc
   2. A console window will pop up and display the VM as it boots up (X-forwarding required)
6. Going through Turnkey first-boot setup
   1. From VM console: when prompted, set root password (aYee3nez) (root password has since been changed)
   2. Skip Hub services setup
   3. Install security updates (this step takes 7 minutes)
   4. Select Advanced Menu
   5. Select Network Configuration to display IP address (69.91.227.157)

======

End of “Step 1”

“Step 2” is about installing system components

======

1. SSH into VM as root
   1. ssh root@69.91.227.157  
      (Svend copied SSH keys for Svend, Gifford, and Darius to the VM.)
2. Download the latest puppet script into the /opt/appliance-setup directory  
   For reference: <https://github.com/cirg/appliance-setup>
   1. Get the script  
      git clone --recursive \  
       https://github.com/cirg/appliance-setup.git \  
       /opt/appliance-setup
   2. Run the puppet script  
      /opt/appliance-setup/bin/appliance-setup apply  
      This can take as long as 10 minutes.   
      Wait for the text: “Finished catalog run in XXX seconds”
   3. This default script will install apache, java, mysql (mysql root pwd: ‘UP9aeKas’), tomcat, and firefox. It is now ready for OpenMRS.

Future plans include a puppet script for installing openmrs (see /opt/applicance-setup/puppet/modules/openmrs/manifests/init.pp for a working version of this) In this version of CIRG Appliance Setup #([f93cccfc42](https://github.com/cirg/appliance-setup/tree/f93cccfc42f246ac7d4ace5e9b4750819c2519d3)), we have not run the openmrs puppet script. This can be done via the command:   
/opt/appliance-setup/bin/appliance-setup puppet -e “include openmrs”

Administrative to dos::

1. Modify Puppet to generate a random pwd for applications that need passwords in Step 2 (just like it should do any other needed configuration), and to write those passwords into a “keys” file, stored in the same location as the “server configuration” file, but with root-only access permissions (if Svend thinks this is appropriate security)
2. Regarding root pwd: items 1-7 above are all “step 1” - there are lots of different ways to get there (install from an iso, use an EC2 instance, etc.), so I don’t think we can have a standard way of dealing w/ the system root pwd. It is the responsibility of the person who set up the system to convey that pwd to whoever else needs it. In the event of the “export template server to an image, copy, and deploy multiple times”, Step 2 might install a “first run” script that either requires the user to reset the root pwd, or that generates a new root pwd and displays it on the TK console.

========================

After Darius was given the system, we then had to do the following:

1. Download OpenMRS 1.9.1 Enterprise WAR (from link on <http://openmrs.org/download/),> and uploaded this to /var/lib/tomcat6/webapps
2. mkdir /usr/share/tomcat6/.OpenMRS
3. chown tomcat6.tomcat6 /usr/share/tomcat6/.OpenMRS
4. Went through OpenMRS initial setup wizard (via browser at <http://69.91.227.157:8080/openmrs>)
   1. replacement for this is if to manually create a database, and an openmrs-runtime.properties file at /usr/share/tomcat6/.OpenMRS/openmrs-runtime.properties before uploading the WAR to tomcat
      1. we can choose another location if we don’t like that one
   2. Non-default Setup options:
      1. Advanced
      2. No (to create DB), password: UP9aeKas
      3. No (for do you have db user), password: UP9aeKas
      4. Admin password: Admin123
   3. Scripting phase 3
      1. Script step 1: creates the .OpenMRS directory
      2. Script step 2: downloads openmrs.war from source forge
      3. Script step X: unzips war file into webapps directory
      4. Script step 3: Create openmrs database account (openmrs)
      5. Script step 4: Create openmrs database (openmrs)
      6. Import database schema/tables
         1. Script step 5: git clone git://github.com/openmrs/openmrs-core.git
         2. cd openmrs-core
         3. Script step 6: git checkout 1.9.0 (should be 1.9.1, but that’s missing at the moment)
         4. run the liquibase script at api/src/main/resources/liquibase-schema-only.xml
         5. run the liquibase script at api/src/main/resources/liquibase-core-data.xml
            1. both of these need to specify the non-standard:

databaseChangeLogTableName = liquibasechangelog

databaseChangeLogLockTableName = liquibasechangeloglock

* + 1. Download MVP/CIEL dictionary from Andy’s dropbox (there should be only one file matching openmrs\_concepts\_1.9.\*.sql.zip) and source this in mysql (this will overwrite some tables in the openmrs database)
    2. Create .OpenMRS/openmrs-runtime.properties (see sample openmrs-runtime.properties below)
    3. Fetch the “Module Distro” OpenMRS module from <https://modules.openmrs.org/modules/download/moduledistro/moduledistro-1.2.omod> and put it in .OpenMRS/modules
    4. Get Kenya EMR, build the right version, and unzip the built distribution
       1. git clone git://github.com/djazayeri/openmrs-module-kenyaemr.git
       2. cd openmrs-module-kenyaemr
       3. git checkout $VERSION\_TAG
       4. mvn package -DbuildDistro=true
       5. unzip distro/target/kenyaemr-distro\*.zip {somewhere}/.OpenMRS
    5. Now is the time to move openmrs.war to tomcat/webapps, or if we did that earlier, restart tomcat.
  1. Scripting phase 4
     1. Import test-database specific content
     2. Administrator username and password

1. Download latest MVP/CIEL dictionary from Andy’s dropbox (openmrs\_concepts\_1.9.0\_20120727.sql.zip), upload, unzip, and source this in mysql
2. Download latest “Module Distro” module from <https://modules.openmrs.org/modules/view.jsp?module=moduledistro> and uploaded to /usr/share/tomcat6/.OpenMRS/modules
   1. then restart tomcat. Better to move this step earlier

Darius Notes:

* mod\_jk hasn’t been set up (at least <https://69.91.227.157/openmrs/> and <http://69.91.227.157/openmrs/> do not work, but <http://69.91.227.157:8080/openmrs/> does)
* mod\_ssl hasn’t been set up (at least [https://69.91.227.157/](https://69.91.227.157/openmrs/) gives CONNECTION\_REFUSED)

Svend: I have set up Apache/SSL and the Apache/Tomcat proxy.

## Gifford’s in-progress version of the puppet openmrs script

(last update 9/5/2012), assumes that maven is installed prior to run

class openmrs {

package { "mysql-server": ensure => installed }

# Chaining the Notifications to control the order of the installation steps.

Notify["OpenMRS-1"] -> # Download war file

Notify["OpenMRS-2"] -> # .OpenMRS directory and .OpenMRS/modules subdirectory

Notify["OpenMRS-3"] -> # Create Mysql user

Notify["OpenMRS-4"] -> # Create Mysql database

Notify["OpenMRS-5"] -> # Git clone and fetch openmrs module kenyaemr

Notify["OpenMRS-6"] -> # Git checkout tags/2012.2-dev

Notify["OpenMRS-7"] -> # maven install

Notify["OpenMRS-8"] # unzip into modules dir

notify {"OpenMRS-1":

message=> "Step 1. openmrs.war has been downloaded from sourceforge and saved to /usr/src/openmrs.war",

}

exec { 'download-openmrs':

cwd => '/usr/src',

creates => '/usr/src/openmrs.war',

command => '/usr/bin/wget \'http://iweb.dl.sourceforge.net/project/openmrs/releases/OpenMRS\_1.9.1/openmrs.war\'',

require => Notify['OpenMRS-1'],

}

notify {"OpenMRS-2":

message=>"Step 2. .OpenMRS directory and modules subdirector are created/confirmed to exist.",

}

file { '/usr/share/tomcat6/.OpenMRS':,

ensure => directory,

group => 'tomcat6',

mode => '0775',

require => Notify['OpenMRS-2'],

}

file { '/usr/share/tomcat6/.OpenMRS/modules':,

ensure => directory,

group => 'tomcat6',

mode => '0775',

require => File['/usr/share/tomcat6/.OpenMRS'],

}

/\*

# no longer in use

exec { 'unzip-openmrs':

command => '/usr/bin/unzip /usr/src/openmrs.war -d /var/lib/tomcat6/webapps/openmrs',

creates => '/var/lib/tomcat6/webapps/openmrs',

require => Exec['download-openmrs'],

}

\*/

notify {"OpenMRS-3":

message=> "Step 3. mysql user openmrs@localhost with temp password \'temp\_openmrs\' exists.",

}

database\_user{ 'openmrs@localhost':

ensure => present,

password\_hash => mysql\_password('temp\_openmrs'),

require => Notify['OpenMRS-3'],

}

database\_grant{'openmrs@localhost':

privileges => [all],

require => Database\_user['openmrs@localhost'],

}

notify {"OpenMRS-4":

message=> "Step 4. database openmrs created",

}

database{ 'openmrs':

ensure => present,

charset => 'utf8',

require => Notify['OpenMRS-4'],

}

notify {"OpenMRS-5":

message=> "Step 5. Clone and fetch a copy of openmrs-module-kenyaemr.git to /usr/src/openmrs-module-kenyaemr",

}

exec{ 'openmrs-module-kenyaemr-git-clone':

command => '/usr/bin/git clone --depth 1 git://github.com/djazayeri/openmrs-module-kenyaemr.git /usr/src/openmrs-module-kenyaemr',

creates => '/usr/src/openmrs-module-kenyaemr',

require => Notify['OpenMRS-5'],

}

exec{ 'openmrs-module-kenyaemr-git-fetch':

cwd => '/usr/src/openmrs-module-kenyaemr',

command => '/usr/bin/git fetch --depth 1 ',

require => Exec['openmrs-module-kenyaemr-git-clone'],

}

notify {"OpenMRS-6":

message=> "Step 6. git checkout tags/2012.2-dev",

}

exec{ 'openmrs-module-kenyaemr-git-checkout':

cwd => '/usr/src/openmrs-module-kenyaemr',

command => '/usr/bin/git checkout tags/2012.2-dev',

require => Notify['OpenMRS-6'],

}

/\* Manually acquiring liquibase (no longer used)

notify {"OpenMRS-7":

message=> "Step 7. Confirm that liquibase.jar is in /opt/liquibase",

}

file{ "/opt/liquibase":

ensure => "directory",

require => Notify['OpenMRS-7'],

}

exec{ "download-liquibase":

cwd => '/opt/liquibase',

command => '/usr/bin/wget \'https://github.com/downloads/liquibase/liquibase/liquibase-2.0.5-bin.zip\'',

creates => '/opt/liquibase/liquibase-2.0.5-bin.zip',

require => File['/opt/liquibase'],

}

\*/

notify {"OpenMRS-7":

message=> "Step 7. Run maven install to create distro.zip",

}

exec{ "maven-install":

cwd => '/usr/src/openmrs-module-kenyaemr',

command => '/usr/bin/mvn install -DbuildDistro=true -DsetupDatabase=true',

require => Notify['OpenMRS-7'],

}

notify {"OpenMRS-8":

message=> "Step 8. Unzip distro into tomcat6 modules directory TODO, control chaining properly",

}

}

Latest: <https://github.com/gief/puppet-openmrs/blob/f404013b71daad675533467bb5e58f81f10b5395/manifests/init.pp>

## Meeting 2012-08-23

1. Change/versioned

2. Explain

3. Variety

### Version string

2012.major.minor-<dev|rc|final>

## Goal is 2012.1 Darius’s Notes from 9/2

Step 1. wget OpenMRS 1.9.1 WAR

Step 2. a. create /usr/share/tomcat6/.OpenMRS and set ownership and permissions

b. mkdir /usr/share/tomcat6/.OpenMRS/modules

mysql

use mysql;

create user openmrs@localhost identified by 'temp\_openmrs';

create database openmrs default character set 'UTF8';

grant all on openmrs to openmrs@localhost;

apt-get install maven2

*this will (1) download supporting modules, and build kenyaemr.omod, and (2) run liquibase scripts from the relevant OpenMRS version to build an empty database with only core data*

cd /usr/src

git clone git://github.com/I-TECH/openmrs-module-kenyaemr.git

cd openmrs-module-kenyaemr

git checkout tags/2012.2-dev (should be a parameter, not always the vm version)

mvn install -DbuildDistro=true -DsetupDatabase=true

unzip distro/target/kenyaemr-distro-\*-distro.zip /usr/share/tomcat6/.OpenMRS/modules

***Whole process, only partly done above:***

* create an empty mysql database called "openmrs" and accessible by the openmrs@localhost user with password temp\_openmrs
* do the mvn install step
* unzip the zip file that step will have built.
* get the MVP dictionary, and load it
* move the OpenMRS 1.9.1 war file to where it's supposed to go in tomcat
  + but particularly do this \*after\* unzipping the modules that kenyaemr put in the distro.zip
* (if a demo installation) Create the demo user in the openmrs.users table

TODOs:

* Include default roles and privileges
* Include demo user

What I did to 157 server on Sept 10-11:

* manually uploaded mvp dictionary (7/27 version)
* source that into openmrs database
  + Seems like I did this before creating the schema. Whoops!
  + drop database openmrs; create database openmrs;
* mysql> set password for 'openmrs'@'localhost' = PASSWORD('temp\_openmrs');
* mysql> grant all on openmrs.\* to 'openmrs'@'localhost';
* cd /usr/src/openmrs-module-kenyaemr
* git checkout master
* git pull
* mvn install -DsetupDatabase=true
* mvn install -DbuildDistro=true
* cd /usr/share/tomcat6/.OpenMRS/modules
* unzip -j /usr/src/openmrs-module-kenyaemr/distro/target/kenyaemr-distro-\*-distro.zip
* wget <https://modules.openmrs.org/modules/download/moduledistro/moduledistro-1.2.omod>
* wget https://modules.openmrs.org/modules/download/logic/logic-0.5.2.omod
* source the mvp dictionary again
* cat > /usr/share/tomcat6/.OpenMRS/openmrs-runtime.properties
* connection.username=openmrs
* connection.password=temp\_openmrs
* connection.url=jdbc:mysql://localhost:3306/openmrs?autoReconnect=true&sessionVariables=storage\_engine=InnoDB&useUnicode=true&characterEncoding=UTF-8
* module.allow\_web\_admin=false
* auto\_update\_database=false
* (control-D)
* Added these lines to /etc/init.d/tomcat6
  + OPENMRS\_RUNTIME\_PROPERTIES\_FILE=/usr/share/tomcat6/.OpenMRS/openmrs-runtime.properties
  + export OPENMRS\_RUNTIME\_PROPERTIES\_FILE
  + JAVA\_OPTS="$JAVA\_OPTS -Xmx512m -Xms512m -XX:PermSize=256m -XX:MaxPermSize=256m -XX:NewSize=128m"
* cp /usr/src/openmrs.war /var/lib/tomcat6/webapps
* Visited <https://69.91.227.157/openmrs> and had to enter maintenance mode to apply one DB update. (Because kenyaemr is still referencing OpenMRS 1.9.0, and this needs updating)
* Things seem to hang, with catalina.out ending with this, and no indication of why
  + DEBUG - HtmlFormSubstitutionUtils.performSubstitution(82) |2012-09-12 00:53:49,223| substitution pattern: <encounterProviderAndRole(?:\s|\s[^>]\*\s)encounterRole
  + DEBUG - HtmlFormSubstitutionUtils.performSubstitution(82) |2012-09-12 00:53:49,223| substitution pattern: <encounterProviderAndRole(?:\s|\s[^>]\*\s)encounterRole
* Actually, after some time I see
  + WARN - ThreadPoolAsynchronousRunner$DeadlockDetector.run(608) |2012-09-12 00:55:47,796| com.mchange.v2.async.ThreadPoolAsynchronousRunner$DeadlockDetector@87b852 -- APPARENT DEADLOCK!!! Creating emergency threads for unassigned pending tasks!
  + WARN - ThreadPoolAsynchronousRunner$DeadlockDetector.run(624) |2012-09-12 00:55:57,404| com.mchange.v2.async.ThreadPoolAsynchronousRunner$DeadlockDetector@87b852 -- APPARENT DEADLOCK!!! Complete Status:
  + Managed Threads: 3
  + Active Threads: 3
  + Active Tasks:
  + com.mchange.v2.resourcepool.BasicResourcePool$AcquireTask@b4cbd (com.mchange.v2.async.ThreadPoolAsynchronousRunner$PoolThread-#0)
  + I don’t know what’s causing this and I’ve never seen it before, so I just did a dump of my local laptop OpenMRS database, scp’d it to /usr/src, and sourced it, then restarted tomcat
    - (later note) this was probably due to lack of memory.
  + somehow the ‘role’ and ‘privilege’ tables weren’t created, so I manually ran those lines from my dump...

## Anikate and Darius upgrading the 157 server with a new release

Where we pick up, the machine is already set up with a demo of an earlier version of the code (including database, openmrs-runtime.properties file, etc.)

**How to Tag a Release, as a Developer**

*On a developer machine, with the code cloned directly from the I-TECH github repository*

git clone https://github.com/I-TECH/openmrs-module-kenyaemr.git

(Make sure that our user has github permissions to push to this repository)

git push

(enter username and password, make sure you don’t get a 403 back. Assuming not, continue.)

mvn install

check to make sure we get SUCCESS for all modules

mvn release:prepare

(you should see: What is the release version for ”Kenya OpenMRS EMR Module”?)

2012.1.1

2012.1.1

2012.1.1

2012.1.1

2012.1.1

(type this 5 times, for root, API, OMOD, Distribution, Installation)

2012.1.1

(type it one more time, for the tag)

2012.1.2-SNAPSHOT

2012.1.2-SNAPSHOT

2012.1.2-SNAPSHOT

2012.1.2-SNAPSHOT

2012.1.2-SNAPSHOT

(type this 5 times, for root, API, OMOD, Distribution, Installation)

(it will build for a while, then prompt for your github username/password)

If the build fails at this point, after having pushed commits to github, then you need to do “mvn release:rollback”

(At this point, there is a “2012.1.1” tag in the code)

TODO: consider doing a backup at this point (of database and maybe configuration)

**How to Update a Server to a new release of the Kenya EMR Distribution**

*On the dev-demo server (i.e. how to update a server to the latest kenyaemr distribution)*

cd /usr/src/openmrs-module-kenyaemr

git checkout master

git pull

git checkout tags/2012.1.1

(it will tell you you are in ‘detached HEAD’ state: this is okay, and expected)

mvn clean install -Pdistribution

(it should report SUCCESS for all 5 maven modules)

cd /usr/share/tomcat6/.OpenMRS

rm -rf modules-backup

(delete last time’s module folder backup, if there is one)

mv modules modules-backup

mkdir modules

cd modules

cp /usr/src/openmrs-module-kenyaemr/distro/target/distro/\*

*Sometimes (but we have no good way of signaling when) you also need to update to the latest version of the MVP concept dictionary...TODO: figure out how to automate and version this, and insert that into the process now...*

(temporarily, this time only, we need to manually keep the logic module. Next release will include the logic module so we don’t need to automate this step)

cp modules-backup/logic-0.5.2.omod modules

service tomcat6 restart

(if you’re curious, you can see progress: tail -f /var/log/tomcat6/catalina.out )

Using a browser, navigate to the demo server <https://69.91.227.157/openmrs>

Verify that the header has the correct version in it: v2012.1.1

Click around a bit and make sure things are running

## MySQL puppet issues

Fri 10/26/2012 6:10 PM

I tried what you mentioned but puppet continued to throw access denied errors for mySQL.

I did some digging and found the below issues -

<https://github.com/puppetlabs/puppetlabs-mysql/pull/92>

<http://comments.gmane.org/gmane.comp.sysutils.puppet.user/44947>

There are some problems with Puppet reading the /root/.my.cnf file for mySQL root password. The

resolution (temporary) mentioned was to move the my,cnf file to /etc.

After moving it to /etc/my.cnf, the db user and db creation step went through successfully.

I will move on to the next step now. Wanted to let you know that for now we might have to store the my.cnf file in /etc which

is where Puppet is looking for it. The problem is that this is not secure, so Puppet must provide some fix.

## KenyaEMR documentation of process (Bill, Anikate, Oct 28)

(link to this document section is at <http://goo.gl/jhecy>)

Bill’s experience with this process is documented here: <http://goo.gl/o0hET> Anikate will add to that.

### Step 1

Default TK 12.0 Core build

### Step 2

Puppet Modules

Apache

Tomcat

MySQL

### Step 3 (Application install)

Puppet Module

KenyaEMR - placeholder - replace w/ these two commands

/opt/appliance-setup/bin/appliance-setup puppet -e “include openmrs”

/opt/appliance-setup/bin/appliance-setup puppet -e “include kenyaemr”

create users in DB

create OMRS db

git clone of KenyaEMR module code/tagged checkout (“2012.1.2”)

git merge ?

maven install -

get’s other dependent modules

liquibase applied

output is distro.zip

unzip into OMRS modules folder in Apache

wget concept dictionary from dropbox

import concept dictionary into MySQL OMRS db

restart Tomcat

### Step 4 (Configuration)

/opt/appliance-setup/bin/appliance-setup puppet -e “include kenyaemr-demo”

kenyaemr-ci, kenyaemr-test, kenyaemr-prod, etc.

create OMRS users...

what varies between ci, test, demo, for instance? <http://goo.gl/8QKvl>

test server might have real data and unique pwds

demo server might have demo data, simple pwds, reset db nightly

prod might have a single test patient, unique pwds,

## SWIM LANE

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Step 1  Base System | Step 2  Sys Components | Step 3  App Components | Step 4  Application Config | Step 5  Interop Config |
| iSante | TK 12.0 | Apache, TK, mod\_proxy\_  maven  java  mysql | iSante  PHP  Jquery  Ext | CI  test  demo  prod | configure interface |
| OE | TK 12.0 | A,TK, modproxy,  Java  MySql | -MySql  +Postgres  OE | ci  test  demo  prod | configure interface |
| KenyaEMR | TK 12.0 | A,T, modproxy, Maven, Java, MySQL, Monitoring | OMRS 1.9.1  KenyaEMR module | ci  test  demo  prod  db vers/refresh/app config  auth config |  |
| DHAIR/cPRO |  |  |  |  |  |
| Gossamer | TK 12.0 |  |  |  |  |

LDAP

Python

cURL

PostgresSQL extension for OE

How to tell version in the repo -- uses the latest supported by Debian, so check packages.debian.org

What about if you need an older version of something not installed by Debian --

- can install from the source

- then you would have to manage everything yourself

What about incompatibilities between versions of components?

**Gaps/Next Steps**

What is missing that is needed to support the projects

- definition of system components (debian repo and installed in standard way) vs. application components (anywhere but from the debian repo)

- iSante - Steve/Svend

- OE - Paul S

- KenyaEMR - Anikate

- DHAIR/cPRO - Ivan

- Gossamer - on hold

- DOH - Paul B

- Github repo / docs - Svend

- system components

- application components

Svend/Paul B will write up a quick virtualbox/vm setup

Update from everyone next **Mondays Informatics Call 10am**