Puppet Procedure to run Fermi2AWS Export

1. For a Paravirtual image on AWS: (Copy these for other images and rename them)

- vi /etc/puppet/modules/awsexport/manifests/gcso_sl6_pv.pp
- change parameters to your credentials.
- *Note: Obtain HVM worker instance id from aws console (caws_worker_instance_id => 'i-7788c97c',)

- #contains parameters for the gcso_sl6 PV image conversion
- awsexport::awsexport_params {'gcso_sl6_pv':
- cvm_file_location => '/opt/gcso/awsexport',
- cvm_image_location => 'oneadmin@fcl008:/var/lib/one/local/images/55c42a4cc7f87ea3390bc2bef14212c5',
- ckernel_ver => '2.6.32-431.23.3.el6.x86_64',
- cvm_number => '103',
- cvm_name => 'gcso_sl6',
- cvm_owner => 'your Fermi username',
- caws_image_name => 'GCSO_SL6_PV',
- caws_image_owner => 'gcso',
- caws_instance => 'm3.medium',
- caws_key => 'add aws owner key here',
- caws_secret_key => 'add aws secret owner key here',
- caws_pem_name => 'gcso.pem',
- caws_worker_instance_id => 'i-7788c97c',
- caws_owner_keypair_name => 'gcso',
- caws_eph_mount => '/ ephemeral_mount_dir or none',
- }

2. For a HVM image on AWS: (Copy these for other images and rename them)

- vi /etc/puppet/modules/awsexport/manifests/gcso_sl6_hvm.pp
- change parameters to your credentials.
- *Note: DO NOT CHANGE (caws_worker_instance_id => 'hvm',) leave as 'hvm' to create a worker vm on aws

- #contains parameters for the gcso_sl6 HVM image conversion
- awsexport::awsexport_params {'gcso_sl6_hvm':
- cvm_file_location => '/opt/gcso/awsexport',
- cvm_image_location => 'oneadmin@fcl008:/var/lib/one/local/images/55c42a4cc7f87ea3390bc2bef14212c5',
- ckernel_ver => '2.6.32-431.23.3.el6.x86_64',
- cvm_number => '103',
- cvm_name => 'gcso_sl6',
- cvm_owner => 'your Fermi username ',
- caws_image_name => 'GCSO_SL6_HVM',
- caws_image_owner => 'gcso',
- caws_instance => 'm3.medium',
- caws_key => 'add aws owner key here',
- caws_secret_key => 'add aws secret owner key here',
- caws_pem_name => 'gcso.pem',
- caws_worker_instance_id => 'hvm',
- caws_owner_keypair_name => 'gcso',
- caws_eph_mount => '/ephemeral_mount_dir or none',
- }

3. To setup Crontab jobs:

- vi /etc/puppet/modules/awsexport/manifests/awsexport_params.pp

- Change MAILTO=username@fnal.gov to your email address to receive cron job completion emails.

- Change the runtime schedule to what you want:

- cron {'awsexport':

  minute => '55',

  hour => '10',

  monthday => '14',

  month => '8',

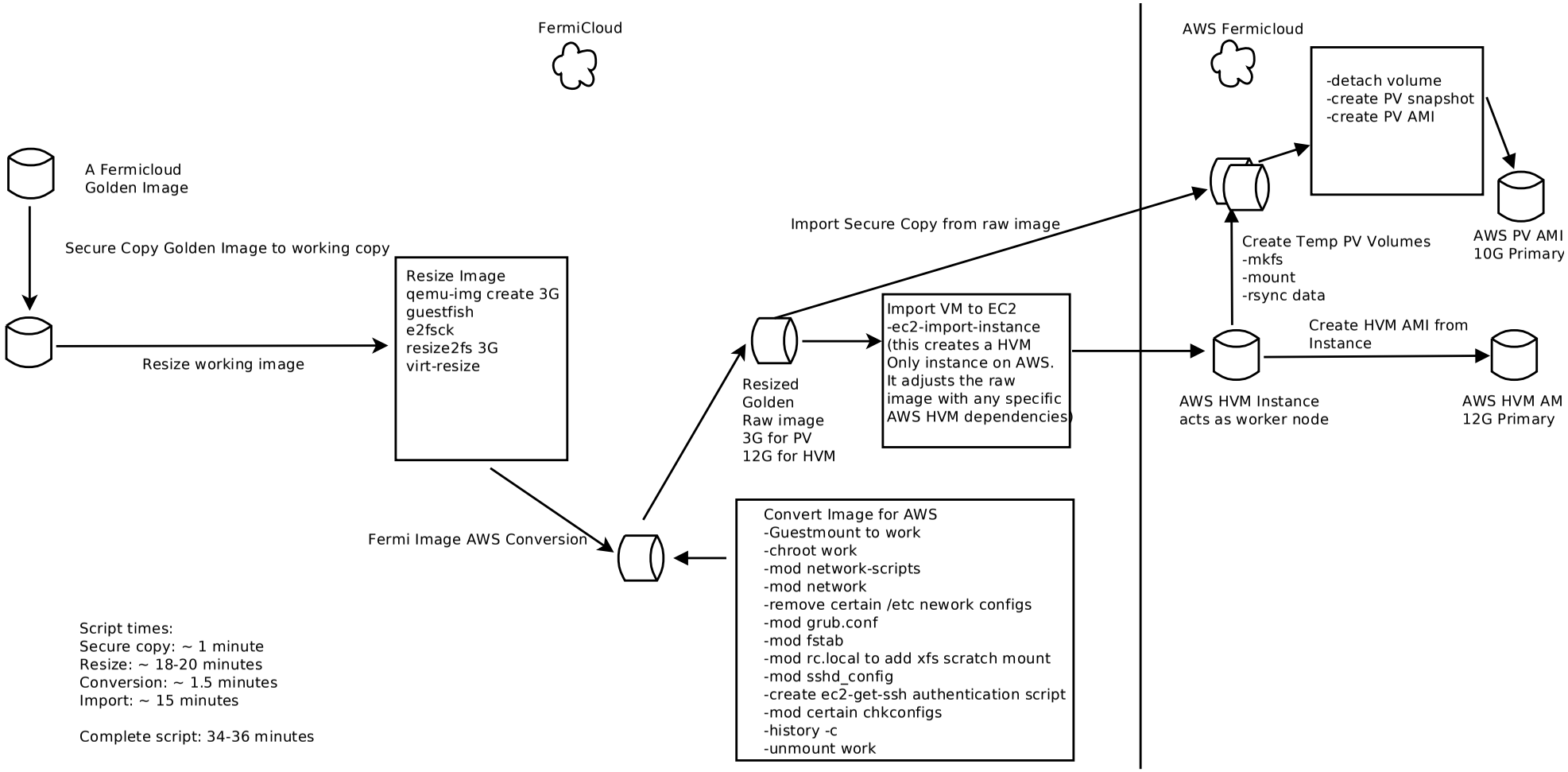
  weekday => '*',

4. Run Puppet apply to set crontab job for AWS HVM worker:

- run 'puppet apply /etc/puppet/modules/awsexport/manifests/gcso_sl6_hvm.pp'
- wait for cron completion email (in about 1.5 hours for a HVM Conversion)
- detail job log is located at: /opt/gcso/awsexport/aws_image_convert.log
- obtain aws console HVM worker node 'instance id' for subsequent PV conversion runs.
- *Note: this HVM worker node is needed only once. It can be run again for other images, if you want to provide HVM AMI's and instances on AWS.

5. Run Puppet apply to set crontab job for AWS PV Images:

- run 'puppet apply /etc/puppet/modules/awsexport/manifests/gcso_sl6_pv.pp'
- wait for cron completion email (in about 55 minutes for a PV Conversion)
- detail job log is located at: /opt/gcso/awsexport/aws_image_convert.log
- check aws console to see PV AMI's and instances.
- *Note: this job can run, as needed, to obtain a latest AWS image. The AWS AMI's and instances are time-stamped to identify the latest version.




Script times:
Secure copy: ~ 1 minute
Resize: ~ 18-20 minutes
Conversion: ~ 1.5 minutes
Import: ~ 15 minutes

Complete script: 34-36 minutes

```
File: puppetrepo-awsexport/files/opt/gcso/awsexport/Convert.py
#!/usr/bin/env python
# encoding: utf-8
'''
FermiCloud.AWS.Images -- Convert FermiCloud KVM Images to AWS Xen Images

@author:     khs

@copyright:  2014 Fermilab. All rights reserved.

@license:    license

@contact:    kirkshal@fnal.gov
'''

import sys
import os
#import random
#import string
import subprocess
import logging
import textwrap
#import socket
import datetime
import time
#import httplib


from argparse import ArgumentParser
from argparse import RawDescriptionHelpFormatter

__all__ = []
__version__ = 0.1
__date__ = '2014-07-15'
__updated__ = '2014-07-15'

DEBUG = 1
TESTRUN = 0
PROFILE = 0

class CLIError(Exception):
    '''Generic exception to raise and log different fatal errors.'''
    def __init__(self, msg):
        super(CLIError).__init__(type(self))
        self.msg = "E: %s" % msg
    def __str__(self):
        return self.msg
    def __unicode__(self):
```

```
        return self.msg

class Timer:
    def __enter__(self):
        self.start = time.time()
        return self

    def __exit__(self, *args):
        self.end = time.time()
        self.interval = self.end - self.start

def get_help():
    """This function outputs the general usage of the script and uses
       the text wrapper class to  specify 80 columns for display.
    """
    wrapper = textwrap.TextWrapper(width=80,
    initial_indent=" " * 2,
    subsequent_indent=" " * 2,
    break_long_words=False,
    replace_whitespace=False,
    expand_tabs=True,
    break_on_hyphens=False)
    help_text= ("(Run Fermi's Kerberos Initialize before running this script: kinit userid). This script takes
fifteen (15) arguments: "
                " <script dir location>, <vm image location>, <vm kernel version>, <fermicloud worker vm number>,
<fermicloud vm name>, <fermicloud vm owner>, <aws image name>,"
                " <aws image owner>, <aws instance type>, <aws key>, <aws secret key>, <aws pem name>,"
                " <aws worker instance id>, <aws owner keypair name> and </ephemeral_mount_dir or none>."
                " A pre-requisite is to have a Fermicloud worker vm setup ahead of time
(root@fermicloudnnn.fnal.gov) with a /data directory created to store files."
                " The script will accept a VM image
(oneadmin@fcl008:/var/lib/one/local/images/55c42a4cc7f87ea3390bc2bef14212c5),"
                " with kernel (2.6.32-431.23.3.el6.x86_64) and will make a worker copy of the image ready for AWS
conversion,"
                " then convert the image (fermi cleanse and resize) and import to AWS by the specified <aws image
name>."
                " The script requires 5 parameters to be obtained ahead of time from the AWS Console. They are:"
                " <aws owner security pem name> <aws instance id of the HVM worker image, use 'hvm' here to
create HVM worker before creating PV's.> <aws owner keypair name>"
                " <aws key> and <aws secret key> which are obtained from the AWS console under EC2 instances and
the IAM security tab prior to running this script."
                " For example, to convert a Fermicloud VM named gcso_slf6 with owner oneadmin,"
                " and to create a AWS image named SLF6Vanilla, with owner oneadmin, and a instance type of
m3.medium and with a worker"
                " vm of fermicloud103.fnal.gov you would run the following script:\n"
                "  **\n"
```

```
                " /opt/gcso/awsexport/Convert.py /opt/gcso/awsexport
oneadmin@fcl008:/var/lib/one/local/images/55c42a4cc7f87ea3390bc2bef14212c5 2.6.32-431.23.3.el6.x86_64 103
gcso_slf6 oneadmin SLF6Vanilla gcso m3.medium awskey awssecretkey gcso.pem i-hvminstanceid gcso /scratch\n"
                "  **\n"
                " ...Available.AWS.Instance.Types.listed.below...\n"
                "  Type......vCPU..ECU..Memory.(GiB)..Instance Storage (GB)..Cost per hour\n"
                " General.Purpose.-.Current.Generation\n"
                " t2.micro....1....Variable....1....EBSOnly......$0.013 per Hour\n"
                " t2.small....1....Variable....2....EBSOnly......$0.026 per Hour\n"
                " t2.medium...2....Variable....4....EBSOnly......$0.052 per Hour\n"
                " m3.medium...1.......3......3.75...1.x.4.SSD....$0.070 per Hour\n"
                " m3.large....2......6.5.....7.5....1.x.32.SSD...$0.140 per Hour\n"
                " m3.xlarge...4......13.......15....2.x.40.SSD...$0.280 per Hour\n"
                " m3.2xlarge..8......26.......30....2.x.80.SSD...$0.560 per Hour\n"
                " Compute.Optimized.-.Current.Generation\n"
                " c3.large....2.......7......3.75...2.x.16.SSD...$0.105 per Hour\n"
                " c3.xlarge...4......14......7.5....2.x.40.SSD...$0.210 per Hour\n"
                " c3.2xlarge..8......28.......15....2.x.80.SSD...$0.420 per Hour\n"
                " c3.4xlarge..16.....55.......30....2.x.160.SSD..$0.840 per Hour\n"
                " c3.8xlarge..32....108.......60....2.x.320.SSD..$1.680 per Hour\n"
                " GPU.Instances.-.Current.Generation\n"
                " g2.2xlarge..8......26.......15........60.SSD...$0.650 per Hour\n"
                " Memory.Optimized.-.Current.Generation\n"
                " r3.large....2......6.5......15....1.x.32.SSD...$0.175 per Hour\n"
                " r3.xlarge...4.....13.......30.5...1.x.80.SSD...$0.350 per Hour\n"
                " r3.2xlarge..8.....26........61....1.x.160.SSD..$0.700 per Hour\n"
                " r3.4xlarge..16....52.......122....1.x.320.SSD..$1.400 per Hour\n"
                " r3.8xlarge..32....104......244....2.x.320.SSD..$2.800 per Hour\n"
                " Storage.Optimized.-.Current.Generation\n"
                " i2.xlarge...4.....14.......30.5...1.x.800.SSD..$0.853 per Hour\n"
                " i2.2xlarge..8.....27.......61.....2.x.800.SSD..$1.705 per Hour\n"
                " i2.4xlarge..16....53.......122....4.x.800.SSD..$3.410 per Hour\n"
                " i2.8xlarge..32....104......244....8.x.800.SSD..$6.820 per Hour\n"
                " hs1.8xlarge.16....35.......117...24.x.2048.....$4.600 per Hour\n")
    print '\n', wrapper.fill(help_text)
    print
#    sys.exit()

def copy_to_image_location(vm_script_location, vm_number, vm_name, vm_owner, aws_pem_name, vm_image_location):
    """ This function copies the selected Fermicloud VM image to a worker VM image.
    """
    get_date = datetime.date.today()
    time_differential = datetime.timedelta(days=7)
    delete_date = str(get_date - time_differential)
    logging.info("Start: Copying the selected Fermicloud VM image. Function copy_to_image_location.")
    """ Change to location of your scripts below.
    """
    cp_command = (
```

```
                  "cp -f {v_script_location}/Fermi_AWS_Modifications.sh /data"
                  " && cp -f {v_script_location}/{v_aws_pem_name} /data"
                  " && cp -f {v_script_location}/Fermi_AWS_Resize.sh /data"
                  " && cp -f {v_script_location}/ec2-get-ssh /data"
                  " && cp -f {v_script_location}/cloud.cfg /data"
                  " && cp -f {v_script_location}/Fermi_AWS_CLI_Setup.sh /root"
                  " && cp -f {v_script_location}/Fermi_AWS_Import.sh /root"
                          " && kinit -k -t /var/adm/krb5/cloudadminpp.keytab
cloudadmin/cron/fermicloudpp.fnal.gov@FNAL.GOV"
                          " && scp {v_image_location} /data/{v_name}.qcow2tmp"
                      #   The 2 lines above need to be modified for production to accept a parameter driven image
the one above is pre-production
                      #  " && scp oneadmin@fcl008:/var/lib/one/local/images/55c42a4cc7f87ea3390bc2bef14212c5
/data/{v_name}.qcow2tmp"
                  #   " && ssh -o StrictHostKeyChecking=no -l {v_owner} fermicloud.fnal.gov 'scp -Cq
{v_owner}@fermicloud.fnal.gov:{v_name}.qcow2 root@fermicloud{v_number}.fnal.gov:/data/{v_name}.qcow2tmp"
                #   " && rm -f {v_owner}@fermicloud.fnal.gov:{v_name}.qcow2.{remove} && exit'"
                          ).format (v_owner=vm_owner, today=str(get_date), v_name=vm_name, v_number=vm_number,
v_script_location=vm_script_location, v_aws_pem_name=aws_pem_name, v_image_location=vm_image_location,
remove=delete_date)
    with open('/opt/gcso/awsexport/aws_image_convert.log', 'a') as my_log:

        try:
            with Timer() as t:
                subprocess.check_call(cp_command, shell=True, stdout=my_log, stderr=my_log)
                logging.info("Stop: Completed Copying the selected Fermicloud VM image. Function
copy_to_image_location.")
        except:
            logging.error("Couldn't copy image to temp area")
            raise Exception("Couldn't copy image to temp area! Aborting.")
            sys.exit(1)
        finally:
            min_interval = t.interval / 60
            print('Copying took %.03f minutes.' % min_interval)

def convert_image(vm_number, vm_name, vm_owner, aws_worker_instance_id, aws_instance, kernel_ver, eph_mount):
    """ This function converts the worker Fermicloud VM image for AWS specifics.
    """
    get_date = datetime.date.today()
    logging.info("Start: Converting the worker Fermicloud VM image. Function convert_image.")
    convert_command = (
                          #         "ssh root@fermicloud{v_number}.fnal.gov 'mkdir -p /data/work"
                                    "mkdir -p /data/work"
                      " && guestmount -a /data/{v_name}.raw -m /dev/sda1 /data/work"
                      " && mv /data/Fermi_AWS_Modifications.sh /data/work"
                      " && mv /data/ec2-get-ssh /data/work"
                      " && mv /data/cloud.cfg /data/work"
```

```
                        " && /usr/sbin/chroot /data/work ./Fermi_AWS_Modifications.sh {v_aws_worker_instance_id}
{v_aws_instance} {v_kernel_ver} {v_eph_mount}"
                        " && sleep 10"
                        " && rm -f /data/work/Fermi_AWS_Modifications.sh && rm -f /data/work/ec2-get-ssh && rm -f
/data/work/cloud.cfg"
                        " && rm -f /data/Fermi_AWS_Modifications.sh && rm -f /data/ec2-get-ssh && rm -f
/data/cloud.cfg"
                        " && fusermount -uz /data/work && rmdir /data/work"
                          #        " && exit'"
                        ).format (v_owner=vm_owner, today=str(get_date), v_name=vm_name, v_number=vm_number,
v_aws_worker_instance_id=aws_worker_instance_id, v_aws_instance=aws_instance, v_kernel_ver=kernel_ver,
v_eph_mount=eph_mount)
    with open('/opt/gcso/awsexport/aws_image_convert.log', 'a') as my_log:

        try:
            with Timer() as t:
                subprocess.check_call(convert_command, shell=True, stdout=my_log, stderr=my_log)
                logging.info("Stop: Completed Converting the worker Fermicloud VM image. Function
convert_image.")
        except:
            logging.error("Couldn't convert image in temp area")
            raise Exception("Couldn't convert image in temp area! Aborting.")
            sys.exit(1)
        finally:
            min_interval = t.interval / 60
            print('Converting took %.03f minutes.' % min_interval)

def resize_image(vm_number, vm_name, vm_owner, aws_worker_instance_id):
    """ This function resizes the worker Fermicloud VM image for AWS image import.
    """
    get_date = datetime.date.today()
    logging.info("Start: Resizing the worker Fermicloud VM image. Function resize_image.")
    resize_command = (
                          #        "ssh root@fermicloud{v_number}.fnal.gov 'cd /data"
                                 "cd /data"
                        " && ./Fermi_AWS_Resize.sh {v_name} {v_aws_worker_instance_id}"
                        " && rm -f /data/Fermi_AWS_Resize.sh"
                  #     " && exit'"
                      ).format (v_owner=vm_owner, today=str(get_date), v_name=vm_name, v_number=vm_number,
v_aws_worker_instance_id=aws_worker_instance_id)
    with open('/opt/gcso/awsexport/aws_image_convert.log', 'a') as my_log:

        try:
            with Timer() as t:
                subprocess.check_call(resize_command, shell=True, stdout=my_log, stderr=my_log)
                logging.info("Stop: Completed Resizing the worker Fermicloud VM image. Function resize_image.")
        except:
            logging.error("Couldn't resize image in temp area")
```

```
            raise Exception("Couldn't resize image in temp area! Aborting.")
            sys.exit(1)
        finally:
            min_interval = t.interval / 60
            print('Resizing took %.03f minutes.' % min_interval)

def import_image(vm_number, vm_name, vm_owner, aws_instance, aws_key, aws_secret_key, aws_pem_name,
aws_worker_instance_id, aws_image_owner, aws_image_name, aws_owner_keypair_name, my_env):
    """ This function sets up the AWS CLI tools and imports the resized raw worker Fermicloud VM image to a AWS
HVM Instance.
    """
    get_date = datetime.date.today()
    logging.info("Start: Importing the worker Fermicloud VM image. Function import_image.")
    import_command = (
                     #              "ssh root@fermicloud{v_number}.fnal.gov 'cd /root && chmod +x
Fermi_AWS_CLI_Setup.sh"
                                    "cd /root && chmod +x Fermi_AWS_CLI_Setup.sh"
                        " && ./Fermi_AWS_CLI_Setup.sh {v_aws_key} {v_aws_secret_key}"
                        " && rm -f /root/Fermi_AWS_CLI_Setup.sh && cd /root && chmod +x Fermi_AWS_Import.sh"
                        " && ./Fermi_AWS_Import.sh {v_aws_key} {v_aws_secret_key} {v_name} {v_aws_instance}
{v_aws_pem_name} {v_aws_worker_instance_id} {v_aws_image_owner} {v_aws_image_name} {v_aws_owner_keypair_name}"
                        " && rm -f /root/Fermi_AWS_Import.sh"
            #           " && exit'"
                                ).format (v_owner=vm_owner, today=str(get_date), v_name=vm_name,
v_number=vm_number, v_aws_instance=aws_instance, v_aws_key=aws_key,
                                          v_aws_secret_key=aws_secret_key, v_aws_pem_name=aws_pem_name,
v_aws_worker_instance_id=aws_worker_instance_id,
                                          v_aws_image_owner=aws_image_owner, v_aws_image_name=aws_image_name,
v_aws_owner_keypair_name=aws_owner_keypair_name)
    with open('/opt/gcso/awsexport/aws_image_convert.log', 'a') as my_log:

        try:
            with Timer() as t:
                subprocess.check_call(import_command, env=my_env, shell=True, stdout=my_log, stderr=my_log)
                logging.info("Stop: Completed Importing the worker Fermicloud VM image. Function import_image.")
        except:
            logging.error("Couldn't import image to AWS")
            raise Exception("Couldn't import image to AWS! Aborting.")
            sys.exit(1)
        finally:
            min_interval = t.interval / 60
            print('Importing took %.03f minutes.' % min_interval)

def main(argv=None): # IGNORE:C0111
    """ Change location of Log file.
    """
    logging.basicConfig(level=logging.DEBUG,
                        format='%(asctime)s %(levelname)-8s %(message)s',
```

```
                        datefmt='%b %d %H:%M:%S',
                        filename='/opt/gcso/awsexport/aws_image_convert.log')

    my_env = os.environ.copy()

    '''Command line options.'''

    if argv is None:
        argv = sys.argv
    else:
        sys.argv.extend(argv)

    get_help()
    program_name = os.path.basename(sys.argv[0])
    program_version = "v%s" % __version__
    program_build_date = str(__updated__)
    program_version_message = '%%(prog)s %s (%s)' % (program_version, program_build_date)
    program_shortdesc = __import__('__main__').__doc__.split("\n")[1]
    program_license = '''%s

  Created by khs on %s.
  Copyright 2014 Fermilab. All rights reserved.

  Licensed under the Apache License 2.0
  http://www.apache.org/licenses/LICENSE-2.0

  Distributed on an "AS IS" basis without warranties
  or conditions of any kind, either express or implied.

USAGE

''' % (program_shortdesc, str(__date__))
    try:
        with Timer() as t:
            # Setup argument parser
            parser = ArgumentParser(description=program_license, formatter_class=RawDescriptionHelpFormatter)
            parser.add_argument('-V', '--version', action='version', version=program_version_message)
            parser.add_argument(dest="cvm_script_location", help="Location of all scripts",
metavar="cvm_script_location")
            parser.add_argument(dest="cvm_image_location", help="Location of Fermi Image",
metavar="cvm_image_location")
            parser.add_argument(dest="ckernel_ver", help="Kernel version of Fermi Image", metavar="ckernel_ver")
            parser.add_argument(dest="cvm_number", help="Fermicloud Work VM number", metavar="cvm_number")
            parser.add_argument(dest="cvm_name", help="Fermicloud VM name", metavar="cvm_name")
            parser.add_argument(dest="cvm_owner", help="Fermicloud Owner name", metavar="cvm_owner")
            parser.add_argument(dest="caws_image_name", help="AWS AMI VM name", metavar="caws_image_name")
            parser.add_argument(dest="caws_image_owner", help="AWS AMI Owner name", metavar="caws_image_owner")
            parser.add_argument(dest="caws_instance", help="AWS AMI Instance Type", metavar="caws_instance")
```

```
            parser.add_argument(dest="caws_key", help="AWS key", metavar="caws_key")
            parser.add_argument(dest="caws_secret_key", help="AWS Secret Key", metavar="caws_secret_key")
            parser.add_argument(dest="caws_pem_name", help="AWS PEM Name", metavar="caws_pem_name")
            parser.add_argument(dest="caws_worker_instance_id", help="AWS Worker Instance ID from AWS Console",
metavar="caws_worker_instance_id")
            parser.add_argument(dest="caws_owner_keypair_name", help="AWS Owner Keypair Name",
metavar="caws_owner_keypair_name")
            parser.add_argument(dest="caws_eph_mount", help="AWS ephemeral mount dir (/ephemeral_mount_dir or
none)", metavar="caws_eph_mount")

            # Process arguments
            args = parser.parse_args()
            cvm_script_location = args.cvm_script_location
            cvm_image_location = args.cvm_image_location
            ckernel_ver = args.ckernel_ver
            cvm_number = args.cvm_number
            cvm_name = args.cvm_name
            cvm_owner = args.cvm_owner
            caws_image_name = args.caws_image_name
            caws_image_owner = args.caws_image_owner
            caws_instance = args.caws_instance
            caws_key = args.caws_key
            caws_secret_key = args.caws_secret_key
            caws_pem_name = args.caws_pem_name
            caws_worker_instance_id = args.caws_worker_instance_id
            caws_owner_keypair_name = args.caws_owner_keypair_name
            caws_eph_mount = args.caws_eph_mount

            logging.info('Begin script run')
            print("Arguments supplied:")
            print("location of all files and scripts->", cvm_script_location)
            print("location of fermicloud vm image->", cvm_image_location)
            print("kernel version of fermicloud vm image->", ckernel_ver)
            print("fermicloud worker vm number->", cvm_number)
            print("fermicloud vm image name->", cvm_name)
            print("fermicloud vm owner->", cvm_owner)
            print("aws vm image name->", caws_image_name)
            print("aws vm image owner->", caws_image_owner)
            print("aws instance type->", caws_instance)
            print("aws owner key->", caws_key)
            print("aws owner secret key->", caws_secret_key)
            print("aws owner pem name->", caws_pem_name)
            print("aws hvm worker instance id from aws console for PV or 'hvm' for HVM->",
caws_worker_instance_id)
            print("aws owner keypair name->", caws_owner_keypair_name)
            print("aws ephemeral mount->", caws_eph_mount)
            print("Starting AWS VM conversion. This job can take up to 60 minutes for PV and 84 for HVM...")
            print("Copying the Golden Fermicloud VM image takes under 1 minute...")
```

```
            copy_to_image_location(cvm_script_location, cvm_number, cvm_name, cvm_owner, caws_pem_name,
cvm_image_location)
            print("Resizing the worker Fermicloud VM image takes up to 20 minutes for PV and 10 for HVM...")
            resize_image(cvm_number, cvm_name, cvm_owner, caws_worker_instance_id)
            print("Converting the worker Fermicloud VM image takes over 24 minutes...")
            convert_image(cvm_number, cvm_name, cvm_owner, caws_worker_instance_id, caws_instance, ckernel_ver,
caws_eph_mount)
            print("Importing the raw image to AWS takes up to 15 minutes for PV and 49 for HVM...")
            import_image(cvm_number, cvm_name, cvm_owner, caws_instance, caws_key, caws_secret_key,
caws_pem_name, caws_worker_instance_id, caws_image_owner, caws_image_name, caws_owner_keypair_name, my_env)
            print("Completed AWS VM conversion.")
            logging.info('End script run')
            return 0
    except KeyboardInterrupt:
        ### handle keyboard interrupt ###
        return 0
    except Exception, e:
        if DEBUG or TESTRUN:
            raise(e)
        indent = len(program_name) * " "
        sys.stderr.write(program_name + ": " + repr(e) + "\n")
        sys.stderr.write(indent + "  for help use --help")
        return 2
    finally:
            min_interval = t.interval / 60
            print('Job took %.03f minutes. Thank you.' % min_interval)


if __name__ == "__main__":
    sys.exit(main())
-----------
-----------
-----------


File: puppetrepo-awsexport/files/opt/gcso/awsexport/Fermi_AWS_CLI_Setup.sh
#!/bin/bash

# Use the following steps to install the Amazon API Tools and the Amazon AMI Tools on the Linux platform. 2
positional parameters are needed; You will need your AWS_ACCESS_KEY as the 1st parameter and AWS_SECRET_KEY as
the second parameter.
# Remember to 'chmod +x Fermi_AWS_CLI_Setup.sh' after copying this script to your root directory
# Syntax Format is ./Fermi_AWS_CLI_Setup.sh <AWS_ACCESS_KEY> <AWS_SECRET_KEY>

# Shell Login Script

#  Add the following environment variables to your shell login script (i.e. /root/.bashrc). Make any necessary
changes for your specific environment by replacing AWS_ACCOUNT_NUMBER, AWS_ACCESS_KEY_ID, and
```

```
AWS_SECRET_ACCESS_KEY with your AWS account number and security credentials. Make certain to remove the < and >
characters when providing your values.

cp /root/.bashrc /root/.bashrc.backup
cd /root
rm -f .bashrc

echo '# .bashrc' >> .bashrc
echo ' ' >> .bashrc
echo '# User specific aliases and functions' >> .bashrc
echo ' ' >> .bashrc
echo "alias rm='rm -i'" >> .bashrc
echo "alias cp='cp -i'" >> .bashrc
echo "alias mv='mv -i'" >> .bashrc
echo ' ' >> .bashrc
echo '# Source global definitions' >> .bashrc
echo 'if [ -f /etc/bashrc ]; then' >> .bashrc
echo '        . /etc/bashrc' >> .bashrc
echo 'fi' >> .bashrc
echo ' ' >> .bashrc
echo 'PATH=$PATH:$HOME/bin' >> .bashrc
echo ' ' >> .bashrc
echo 'export PATH' >> .bashrc
echo ' ' >> .bashrc
echo 'export EC2_BASE=/usr/local/ec2' >> .bashrc
echo 'export EC2_HOME=$EC2_BASE/ec2-api-tools-1.7.1.0' >> .bashrc
echo 'export EC2_PRIVATE_KEY=$EC2_BASE/certificates/ec2-pk.pem' >> .bashrc
echo 'export EC2_CERT=$EC2_BASE/certificates/ec2-cert.pem' >> .bashrc
echo 'export EC2_URL=https://ec2.us-west-2.amazonaws.com' >> .bashrc
echo 'export AWS_ACCOUNT_NUMBER=159067897602' >> .bashrc
echo 'export AWS_ACCESS_KEY_ID='$1 >> .bashrc
echo 'export AWS_ACCESS_KEY='$1 >> .bashrc
echo 'export AWS_SECRET_ACCESS_KEY='$2 >> .bashrc
echo 'export AWS_SECRET_KEY='$2 >> .bashrc
echo 'export PATH=$PATH:/usr/local/bin:/usr/local/sbin:/usr/bin:/usr/sbin:/bin:/sbin:$EC2_HOME/bin' >> .bashrc
echo 'export JAVA_HOME=/usr/lib/jvm/jre-1.6.0-openjdk.x86_64' >> .bashrc

source ~/.bashrc

cp /root/.bash_profile /root/.bash_profile.backup
cd /root
rm -f .bash_profile

echo '# Get the aliases and functions' >> .bash_profile
echo 'if [ -f ~/.bashrc ]; then' >> .bash_profile
echo '        . ~/.bashrc' >> .bash_profile
echo 'fi' >> .bash_profile
echo ' ' >> .bash_profile
```

```
echo '# User specific environment and startup programs' >> .bash_profile
echo ' ' >> .bash_profile
echo 'PATH=$PATH:$HOME/bin' >> .bash_profile
echo ' ' >> .bash_profile
echo 'export PATH' >> .bash_profile
echo ' ' >> .bash_profile
echo 'export EC2_BASE=/usr/local/ec2' >> .bash_profile
echo 'export EC2_HOME=$EC2_BASE/ec2-api-tools-1.7.1.0' >> .bash_profile
echo 'export EC2_PRIVATE_KEY=$EC2_BASE/certificates/ec2-pk.pem' >> .bash_profile
echo 'export EC2_CERT=$EC2_BASE/certificates/ec2-cert.pem' >> .bash_profile
echo 'export EC2_URL=https://ec2.us-west-2.amazonaws.com' >> .bash_profile
echo 'export AWS_ACCOUNT_NUMBER=159067897602' >> .bash_profile
echo 'export AWS_ACCESS_KEY_ID='$1 >> .bash_profile
echo 'export AWS_ACCESS_KEY='$1 >> .bash_profile
echo 'export AWS_SECRET_ACCESS_KEY='$2 >> .bash_profile
echo 'export AWS_SECRET_KEY='$2 >> .bash_profile
echo 'export PATH=$PATH:/usr/local/bin:/usr/local/sbin:/usr/bin:/usr/sbin:/bin:/sbin:$EC2_HOME/bin' >>
.bash_profile
echo 'export JAVA_HOME=/usr/lib/jvm/jre-1.6.0-openjdk.x86_64' >> .bash_profile

source ~/.bash_profile

# Install Java

# The EC2 API Tools and Amazon EC2 AMI Tools are Java based. If you don't already have a version of Java
installed, do so now.

yum -y install java-1.6.0-openjdk

# The JAVA_HOME environment variable should be set to the appropriate home directory in your shell login script
(i.e. /root/.bashrc) which was handled in the previous step. Verify the JAVA_HOME environment variable is set for
the current shell and confirm that Java is installed correctly.

echo $JAVA_HOME

# java -version

# Install the Amazon EC2 Tools

# Download the Amazon EC2 API Tools.

mkdir -p $EC2_HOME

curl -o /tmp/ec2-api-tools.zip http://s3.amazonaws.com/ec2-downloads/ec2-api-tools.zip

unzip -qq -o /tmp/ec2-api-tools.zip -d /tmp

cp -r /tmp/ec2-api-tools-*/* $EC2_HOME
```

```
# Download the Amazon EC2 AMI Tools to the EC2 image.

curl -o /tmp/ec2-ami-tools.zip http://s3.amazonaws.com/ec2-downloads/ec2-ami-tools.zip

unzip -qq -o /tmp/ec2-ami-tools.zip -d /tmp

cp -rf /tmp/ec2-ami-tools-*/* $EC2_HOME

# EC2 Private Certificate Key File and EC2 Certificate File

# If used (Certificates are being depracated), Copy your X.509 Certificate (private key file and certificate
file) to appropriate directory. For the purpose of this example, I will be renaming my private key file from pk-
2L7LZYRTNEAC4KGZMPPZWAOZ4KYCTCA4.pem to ec2-pk.pem and my certificate file from cert-
2L7LZYRTNEAC4KGZMPPZWAOZ4KYCTCA4.pem to ec2-cert.pem.

mkdir -p $EC2_BASE/certificates

# cp pk-2L7LZYRTNEAC4KGZMPPZWAOZ4KYCTCA4.pem $EC2_BASE/certificates/ec2-pk.pem

# cp cert-2L7LZYRTNEAC4KGZMPPZWAOZ4KYCTCA4.pem $EC2_BASE/certificates/ec2-cert.pem

# Verify Amazon EC2 Tools

# Verify that the Amazon EC2 Tools have been installed correctly.

# Test the ec2-describe-regions script which is found in the EC2 API Tools to list the regions you have access
to.

# ec2-describe-regions

# Install Expect used for interaction 'yes' prompts throughout script

yum -y install tcl-devel tk-devel expect-devel expectk

# Package updates completed
sleep 1
echo 'AWS CLI Tools Setup Completed.'
-----------
-----------
-----------


File: puppetrepo-awsexport/files/opt/gcso/awsexport/Fermi_AWS_Import.sh
#!/bin/bash

# Use the following steps to perform a VM import to create a HVM Instance on AWS.
# Remember to 'chmod +x Fermi_AWS_Import.sh' after copying this script to your root directory
```

```
# Syntax Format is ./Fermi_AWS_Import.sh <AWS_ACCESS_KEY> <AWS_SECRET_KEY> <Fermicloud VM Name> <AWS Instance
Type> <PEM Name> <AWS Worker Instance ID> <AWS image owner> <AWS image name> <AWS owner keypair name>

cd /data
dte=`date +%Y%m%d%H%M%S`
chmod 400 $5

if [ "$6" == "hvm" ]; then
# VM Import used to create AWS HVM Worker Image that is used for building Paravirtual Images
# Note: That the size of the primary boot partition must stay at 12G to prevent freeze at 53% conversion
import=`/usr/local/ec2/ec2-api-tools-1.7.1.0/bin/ec2iin -f RAW -t $4 -a x86_64 -b fcloudimport$dte -o $1 -O $1 -w
$2 -W $2 -p Linux -g sg-a6a010c3 -s 13 -d $8_$dte -K $9 --region us-west-2 -z us-west-2a $3.raw | grep
IMPORTINSTANCE | awk {'print($4)'}`
sleep 5
# poll every 60 seconds to see if import is completed
status=$(/usr/local/ec2/ec2-api-tools-1.7.1.0/bin/ec2-describe-conversion-tasks -O $1 -W $2 $import | awk
'/IMPORTINSTANCE/{print $8}')
while [ "$status" != "completed" ] ; do
sleep 60
status=$(/usr/local/ec2/ec2-api-tools-1.7.1.0/bin/ec2-describe-conversion-tasks -O $1 -W $2 $import | awk
'/IMPORTINSTANCE/{print $8}')
done
sleep 5

volimp=`/usr/local/ec2/ec2-api-tools-1.7.1.0/bin/ec2-describe-conversion-tasks -O $1 -W $2 $import | grep
DISKIMAGE | awk {'print($7)'}`
sleep 2
vmimpinstance=`/usr/local/ec2/ec2-api-tools-1.7.1.0/bin/ec2-describe-conversion-tasks -O $1 -W $2 $import | grep
IMPORTINSTANCE | awk {'print($10)'}`
sleep 2
# Create Snapshot to be used to create HVM AMI
snap=`/usr/local/ec2/ec2-api-tools-1.7.1.0/bin/ec2-create-snapshot -O $1 -W $2 --region us-west-2 -d "$8_$dte HVM
Snapshot" $volimp | grep SNAPSHOT | awk {'print($2)'}`
sleep 5
# poll every 10 seconds to see if snapshot is completed
status=$(/usr/local/ec2/ec2-api-tools-1.7.1.0/bin/ec2-describe-snapshots -O $1 -W $2 $snap | awk
'/SNAPSHOT/{print $4}')
while [ "$status" != "completed" ] ; do
sleep 10
status=$(/usr/local/ec2/ec2-api-tools-1.7.1.0/bin/ec2-describe-snapshots -O $1 -W $2 $snap | awk
'/SNAPSHOT/{print $4}')
done
sleep 5

# Create HVM AMI
ami=`/usr/local/ec2/ec2-api-tools-1.7.1.0/bin/ec2-register -O $1 -W $2 -n "$8_$dte" -d "$8_$dte" -b
"/dev/sda1=$snap:13:true:standard" -b "/dev/sdb=ephemeral0" --architecture x86_64 --region us-west-2 --
virtualization-type hvm | grep IMAGE | awk {'print($2)'}`
```

```
sleep 30

# Create and run Instance from HVM AMI (Comment out for golive)
instance=`/usr/local/ec2/ec2-api-tools-1.7.1.0/bin/ec2-run-instances -O $1 -W $2 --region us-west-2 -z us-west-2a
$ami -g sg-a6a010c3 -n 1 -t $4 -k $9 | grep INSTANCE | awk {'print($2)'}`
sleep 5
# poll every 10 seconds to see if instance status is running
status=$(/usr/local/ec2/ec2-api-tools-1.7.1.0/bin/ec2-describe-instances -O $1 -W $2 $instance | awk
'/INSTANCE/{print $6}')
while [ "$status" != "running" ] ; do
sleep 10
status=$(/usr/local/ec2/ec2-api-tools-1.7.1.0/bin/ec2-describe-instances -O $1 -W $2 $instance | awk
'/INSTANCE/{print $6}')
done
sleep 5

# Get Volume for create tags
vol2=`/usr/local/ec2/ec2-api-tools-1.7.1.0/bin/ec2-describe-volumes -O $1 -W $2 | grep $instance | awk
{'print($2)'}`
sleep 10

# Create and attach a work volume for ongoing PV conversions
vol3=`/usr/local/ec2/ec2-api-tools-1.7.1.0/bin/ec2-create-volume -O $1 -W $2 --size 20 --region us-west-2 -z us-
west-2a --type standard | grep VOLUME | awk {'print($2)'}`
sleep 10

/usr/local/ec2/ec2-api-tools-1.7.1.0/bin/ec2-attach-volume $vol3 -O $1 -W $2 --instance $instance --device
/dev/sdf

# Create tags for all resources
/usr/local/ec2/ec2-api-tools-1.7.1.0/bin/ec2-create-tags -O $1 -W $2 $vmimpinstance $volimp $snap $ami $instance
$vol2 $vol3 --tag "Name=$8_$dte" --tag "User=$7"
sleep 10

# Stop the AWS HVM Worker and delete temp work volume
/usr/local/ec2/ec2-api-tools-1.7.1.0/bin/ec2-stop-instances $instance -O $1 -W $2
/usr/local/ec2/ec2-api-tools-1.7.1.0/bin/ec2-terminate-instances $vmimpinstance -O $1 -W $2
sleep 90
/usr/local/ec2/ec2-api-tools-1.7.1.0/bin/ec2-delete-volume $volimp -O $1 -W $2
pip install --upgrade awscli
aws s3 rb s3://fcloudimport$dte --force

echo 'AWS HVM VM Import Completed.'
exit
fi

# Launch new HVM AMI Instance (AWS Instance ID is needed as a parameter to the script)
/usr/local/ec2/ec2-api-tools-1.7.1.0/bin/ec2-start-instances $6 -O $1 -W $2
```

```
# poll every 10 seconds to see if instance status is running
status=$(/usr/local/ec2/ec2-api-tools-1.7.1.0/bin/ec2-describe-instances -O $1 -W $2 $6 | awk '/INSTANCE/{print
$6}')
while [ "$status" != "running" ] ; do
sleep 10
status=$(/usr/local/ec2/ec2-api-tools-1.7.1.0/bin/ec2-describe-instances -O $1 -W $2 $6 | awk '/INSTANCE/{print
$6}')
done
sleep 5

# Wait for instance to launch and get public ip address and setup staging area
ip=`/usr/local/ec2/ec2-api-tools-1.7.1.0/bin/ec2-describe-instances -O $1 -W $2 $6 | grep NICASSOCIATION | awk
{'print($2)'}`
sleep 10

# Create new volume on AWS Worker to be used for PV snapshot
vol=`/usr/local/ec2/ec2-api-tools-1.7.1.0/bin/ec2-create-volume -O $1 -W $2 --size 10 --region us-west-2 -z us-
west-2a | grep VOLUME | awk {'print($2)'}`
sleep 10

# Attach volume on AWS Worker and Wait for attachment to complete and setup mount for data
/usr/local/ec2/ec2-api-tools-1.7.1.0/bin/ec2-attach-volume -O $1 -W $2 --instance $6 --device /dev/sdg $vol --
region us-west-2
sleep 10

# build the expect script in bash

    expect1_sh=$(expect -c "
    spawn ssh -i $5 root@$ip
    expect \"yes/no\"
    send \"yes\r\"
    expect \"#\"
    send \"mkfs -t ext4 /dev/xvdf\r\"
      expect \"#\"
    send \"mount -t ext4 /dev/xvdf /mnt\r\"
      expect \"#\"
    send \"cd /mnt\r\"
      expect \"#\"
    send \"mkdir -p images\r\"
    expect \"#\"
    send \"mkfs -t ext4 /dev/xvdg\r\"
      expect \"#\"
    send \"mkdir -p /opt/ec2/mnt\r\"
      expect \"#\"
    send \"mount -t ext4 /dev/xvdg /opt/ec2/mnt\r\"
    expect \"#\"
    send \"exit\r\"
```

```
      expect eof
      puts \"spawned process completed...\"
      exit
    ")

    # run the expect script and then exit it
    echo "$expect1_sh"

echo "Back to script. Please wait..."
sleep 5

# Upload Fermicloud prepaed raw image to AWS Worker stage area (about 3 minutes)
# rsync or scp can be used, rsync requies root password to be passed (so default to scp)
echo "Copying (SCP) Raw image to AWS. Please wait..."
# rsync -S -z -q -e ssh -i $5 /data/$3.raw root@$ip:/mnt/images
scp -Cq -i $5 /data/$3.raw root@$ip:/mnt/images

sleep 5

# Prepare raw image to become extracted boot volume on AWS Worker

# build the expect script in bash

    expect3_sh=$(expect -c "
    spawn ssh -i $5 root@$ip
    expect \"#\"
    send \"cd /mnt/images\r\"
      expect \"#\"
    send \"mkdir -p raw\r\"
      expect \"#\"
    send \"kpartx -a $3.raw\r\"
      expect \"#\"
    send \"mount /dev/mapper/loop0p1 /mnt/images/raw\r\"
      expect \"#\"
    send \"cd /mnt/images/raw\r\"
      expect \"#\"
    send \"rsync -aqHx /mnt/images/raw/ /opt/ec2/mnt\r\"
      sleep 120
      expect \"#\"
    send \"rsync -aqHx /mnt/images/raw/dev /opt/ec2/mnt\r\"
      expect \"#\"
    send \"cd /opt/ec2/mnt\r\"
      expect \"#\"
    send \"tune2fs -L \'/\' /dev/xvdg\r\"
      expect \"#\"
    send \"sync;sync;sync;sync\r\"
      expect \"#\"
    send \"cd /mnt/images\r\"
```

```
      expect \"#\"
    send \"umount /mnt/images/raw\r\"
      expect \"#\"
    send \"kpartx -d $3.raw\r\"
    expect \"#\"
    send \"exit\r\"
      expect eof
      puts \"spawned process completed...\"
      exit
    ")

    # run the expect script and then exit it
    echo "$expect3_sh"

echo "Back to script. Please wait..."
sleep 5

# Detach temp worker volume
/usr/local/ec2/ec2-api-tools-1.7.1.0/bin/ec2-detach-volume $vol -O $1 -W $2
sleep 60

# Create Snapshot to be used to create PV AMI
snap=`/usr/local/ec2/ec2-api-tools-1.7.1.0/bin/ec2-create-snapshot -O $1 -W $2 --region us-west-2 -d "$8_$dte PV
Snapshot" $vol | grep SNAPSHOT | awk {'print($2)'}`
sleep 5
# poll every 10 seconds to see if snapshot is completed
status=$(/usr/local/ec2/ec2-api-tools-1.7.1.0/bin/ec2-describe-snapshots -O $1 -W $2 $snap | awk
'/SNAPSHOT/{print $4}')
while [ "$status" != "completed" ] ; do
sleep 10
status=$(/usr/local/ec2/ec2-api-tools-1.7.1.0/bin/ec2-describe-snapshots -O $1 -W $2 $snap | awk
'/SNAPSHOT/{print $4}')
done
sleep 5

# Create PV AMI
ami=`/usr/local/ec2/ec2-api-tools-1.7.1.0/bin/ec2-register -O $1 -W $2 -n "$8_$dte" -d "$8_$dte" -b
"/dev/sda1=$snap:10:true:standard" -b "/dev/sdb=ephemeral0" --architecture x86_64 --kernel aki-fc8f11cc --region
us-west-2 | grep IMAGE | awk {'print($2)'}`
sleep 30

# Create and run Instance from PV AMI (Comment out for golive)
instance=`/usr/local/ec2/ec2-api-tools-1.7.1.0/bin/ec2-run-instances -O $1 -W $2 --region us-west-2 -z us-west-2a
$ami -g sg-a6a010c3 -n 1 --kernel aki-fc8f11cc -t $4 -k $9 | grep INSTANCE | awk {'print($2)'}`
sleep 5
# poll every 10 seconds to see if instance status is running
status=$(/usr/local/ec2/ec2-api-tools-1.7.1.0/bin/ec2-describe-instances -O $1 -W $2 $instance | awk
'/INSTANCE/{print $6}')
```

```
while [ "$status" != "running" ] ; do
sleep 10
status=$(/usr/local/ec2/ec2-api-tools-1.7.1.0/bin/ec2-describe-instances -O $1 -W $2 $instance | awk
'/INSTANCE/{print $6}')
done
sleep 5

# Get Volume for create tags (Comment out for golive)
vol2=`/usr/local/ec2/ec2-api-tools-1.7.1.0/bin/ec2-describe-volumes -O $1 -W $2 | grep $instance | awk
{'print($2)'}`
sleep 10

# Create tags for all resources
/usr/local/ec2/ec2-api-tools-1.7.1.0/bin/ec2-create-tags -O $1 -W $2 $vol $snap $ami $instance $vol2 --tag
"Name=$8_$dte" --tag "User=$7"
sleep 10

# Stop the AWS HVM Worker and new PV instances and delete temp HVM work volume
/usr/local/ec2/ec2-api-tools-1.7.1.0/bin/ec2-stop-instances $6 -O $1 -W $2
/usr/local/ec2/ec2-api-tools-1.7.1.0/bin/ec2-stop-instances $instance -O $1 -W $2
sleep 5
# poll every 10 seconds to see if instance status is stopped
status=$(/usr/local/ec2/ec2-api-tools-1.7.1.0/bin/ec2-describe-instances -O $1 -W $2 $instance | awk
'/INSTANCE/{print $5}')
while [ "$status" != "stopped" ] ; do
sleep 10
status=$(/usr/local/ec2/ec2-api-tools-1.7.1.0/bin/ec2-describe-instances -O $1 -W $2 $instance | awk
'/INSTANCE/{print $5}')
done
sleep 5
/usr/local/ec2/ec2-api-tools-1.7.1.0/bin/ec2-delete-volume $vol -O $1 -W $2

echo 'AWS VM Import Completed.'

-----------
-----------
-----------


File: puppetrepo-awsexport/files/opt/gcso/awsexport/Fermi_AWS_Modifications.sh
#!/bin/bash
# This is a script to add or modify several OS networking files required for the AWS image conversion

# Remove glideinwms-vm-one (OpenNebula), if exists, and Install glideinwms-vm-ec2 (AWS)

yum -y remove glideinwms-vm-one
yum -y install glideinwms-vm-core
yum -y install glideinwms-vm-ec2
```

```
# Install cloud-init for aws metadata user data and scripts

rpm -Uvh http://download.fedoraproject.org/pub/epel/6/x86_64/epel-release-6-8.noarch.rpm
yum -y install cloud-init
yum -y install cloud-utils
yum -y install http://repos.fedorapeople.org/repos/openstack/openstack-havana/epel-6/python-backports-1.0-
4.el6.x86_64.rpm
cd /
mv -f cloud.cfg /etc/cloud

# Modify ifcfg-eth0
cd /etc/sysconfig/network-scripts
rm -f ifcfg-eth0
echo DEVICE=eth0 >> ifcfg-eth0
echo BOOTPROTO=dhcp >> ifcfg-eth0
echo ONBOOT=yes >> ifcfg-eth0
echo TYPE=Ethernet >> ifcfg-eth0

# Modify network
cd /etc/sysconfig
rm -f network
echo NETWORKING=yes >> network

# Remove fermi network resolv and hosts that AWS will not use
cd /etc
rm -f /etc/resolv.conf
rm -f /etc/hosts
rm -f /etc/hosts.allow
rm -f /etc/hosts.deny

# Remove fermi specifics that AWS will not use
rm -f /etc/yp.conf
rm -f /etc/auto.master
rm -f /etc/auto.misc
rm -f /etc/rc.d/rc3.d/K99.credentials
rm -f /etc/rc.d/rc3.d/S09one-context
rm -f /etc/rc.d/rc3.d/K84one-context
rm -f /etc/init.d/one-context
rm -f /etc/init.d/.credentials*

# Create grub.conf for AWS paravitual
cd /boot/grub
rm -f grub.conf
echo default=0 >> grub.conf
echo timeout=0 >> grub.conf
echo 'title Scientific Linux Fermi ('$3')' >> grub.conf
if [ "$1" == "hvm" ]; then
```

```
echo 'root (hd0,0)' >> grub.conf
echo 'kernel /boot/vmlinuz-'$3' ro root=/dev/xvda1 rd_NO_PLYMOUTH' >> grub.conf
else
echo 'root (hd0)' >> grub.conf
echo 'kernel /boot/vmlinuz-'$3' ro root=/dev/xvde1 rd_NO_PLYMOUTH' >> grub.conf
fi
echo 'initrd /boot/initramfs-'$3'.img' >> grub.conf

# Create symbolic link for boot
cd /boot; ln -s . boot

# Create fstab for AWS paravirtual
cd /etc
rm -f fstab

if [ "$1" == "hvm" ]; then
  echo '/dev/xvda1              /                       ext3    defaults        1 1' >> fstab
  if [ "$4" != "none" ] && [ "$2" != "t2.micro" ] && [ "$2" != "t2.small" ] && [ "$2" != "t2.medium" ]; then
    echo '/dev/xvdb                '$4'                xfs     defaults        0 0' >> fstab
  fi
else
  echo '/dev/xvde1              /                       ext4    defaults        1 1' >> fstab
  if [ "$4" != "none" ] && [ "$2" != "t2.micro" ] && [ "$2" != "t2.small" ] && [ "$2" != "t2.medium" ]; then
    echo '/dev/xvdf                '$4'                xfs     defaults        0 0' >> fstab
  fi
fi

echo 'tmpfs                   /dev/shm                tmpfs   defaults        0 0' >> fstab
echo 'devpts                  /dev/pts                devpts  gid=5,mode=620  0 0' >> fstab
echo 'sysfs                   /sys                    sysfs   defaults        0 0' >> fstab
echo 'proc                    /proc                   proc    defaults        0 0' >> fstab


# Modify rc.local final init script to add additional ephemeral drive and mount scratch there
if [ "$4" != "none" ]; then
cd /etc/rc.d
echo ' ' >>rc.local
echo '# Create additional Scratch directory' >>rc.local
echo 'mkdir -p '$4'' >>rc.local
echo ' ' >>rc.local
echo '# Mount scratch as XFS' >>rc.local
 if [ "$1" == "hvm" ]; then
  echo 'mkfs.xfs -f /dev/xvdb' >>rc.local
  if [ "$2" != "t2.micro" ] && [ "$2" != "t2.small" ] && [ "$2" != "t2.medium" ]; then
    echo 'mount /dev/xvdb '$4'' >>rc.local
  fi
 else
  echo 'mkfs.xfs -f /dev/xvdf' >>rc.local
```

```
  if [ "$2" != "t2.micro" ] && [ "$2" != "t2.small" ] && [ "$2" != "t2.medium" ]; then
    echo 'mount /dev/xvdf '$4'' >>rc.local
  fi
 fi
fi

# Modify sshd_config
cd /etc/ssh
rm -f sshd_config
echo 'SyslogFacility AUTHPRIV' >> sshd_config
echo 'RSAAuthentication no' >> sshd_config
echo 'PubkeyAuthentication yes' >> sshd_config
echo 'AuthorizedKeysFile   .ssh/authorized_keys' >> sshd_config
echo 'PasswordAuthentication yes' >> sshd_config
echo 'KerberosAuthentication no' >> sshd_config
echo 'KerberosOrLocalPasswd no' >> sshd_config
echo 'KerberosTicketCleanup no' >> sshd_config
echo 'GSSAPIAuthentication no' >> sshd_config
echo 'GSSAPICleanupCredentials no' >> sshd_config
echo 'UsePAM no' >> sshd_config
echo 'AllowTcpForwarding yes' >> sshd_config
echo 'X11Forwarding yes' >> sshd_config
echo 'UseLogin no' >> sshd_config
echo 'UseDNS no' >> sshd_config

# Create ec2-get-ssh authenication script for public keypair
cd /
mv ec2-get-ssh /etc/init.d
/bin/chmod +x /etc/init.d/ec2-get-ssh

# Modify /sbin/chkconfig services
/sbin/chkconfig ec2-get-ssh on
/sbin/chkconfig rpcbind off
/sbin/chkconfig postfix off
# /sbin/chkconfig autofs off
/sbin/chkconfig vmcontext off
# Prevent 10 minute automatic vm shutdown for testing (**turn back on after testing**)
# /sbin/chkconfig glideinwms-pilot off

# Clear history
history -c

# exit chroot
exit
-----------
-----------
-----------
```

```
File: puppetrepo-awsexport/files/opt/gcso/awsexport/Fermi_AWS_Resize.sh
#!/bin/bash
# This is a script to resize the converted image (accepts param $1=vmimage name $2="hvm") from a QCOW2 256G image
down to a 12G primary partiton required for the AWS HVM image upload or a 3G primary partiton for a AWS PV image
upload

# Remove previous raw image, if exists, and formats new raw image
cd /data
rm -f $1.raw

# Use guestfish to delete 2nd and 3rd partitions, if present, clean up count errors and resize primary partiton
content (about 8.5 minutes)
      guestfish -a $1.qcow2tmp <<_EOF1_
      run
      part-del /dev/sda 2
      part-del /dev/sda 3
_EOF1_
sleep 2
      guestfish -a $1.qcow2tmp <<_EOF2_
      run
      e2fsck-f /dev/sda1
_EOF2_
sleep 2

if [ "$2" == "hvm" ]; then

      guestfish -a $1.qcow2tmp <<_EOF3_
      run
      resize2fs-size /dev/sda1 12G
_EOF3_
sleep 2
# Resize boot partiton takes about 3.5 minutes
      qemu-img create -f raw $1.raw 12291M
      virt-resize --resize /dev/sda1=12G $1.qcow2tmp $1.raw

else

      guestfish -a $1.qcow2tmp <<_EOF4_
      run
      resize2fs-size /dev/sda1 3G
_EOF4_
sleep 2
# Resize boot partiton takes about 3.5 minutes
      qemu-img create -f raw $1.raw 3075M
      virt-resize --resize /dev/sda1=3G $1.qcow2tmp $1.raw
fi
```

```
# Status of converted raw file
qemu-img info $1.raw
-----------
-----------
-----------
```

File: puppetrepo-awsexport/files/opt/gcso/awsexport/cloud.cfg

```
#Bare-bone cloud.cfg, add parameters as needed for FermiCloud

user: root

#If this is not explicitly false, cloud-init will change things so that root
#login via ssh is disabled. Set it false to allow root login via ssh keypair.

disable_root: false

#add additional cloud-init output logging

output: {all: '| tee -a /var/log/cloud-init-output.log'}

#Since cloud-init runs at multiple stages of boot, this needs to be set so
#it can log in all of them to /var/log/cloud-init.

syslog_fix_perms: null

#This is the piece that makes userdata work. You need this to have userdata
#scripts be run by cloud-init.

datasource_list: [Ec2]
datasource:
  Ec2:
    metadata_urls: ['http://169.254.169.254']

#modules that run early in boot

cloud_init_modules:
 - bootcmd  #for running commands during boot. Commands can be defined in cloud-config userdata.

#modules that run after boot

cloud_config_modules:
 - runcmd  #like bootcmd, but runs after boot. Use this instead of bootcmd for after boot processing.

#modules that run at some point after config is finished

cloud_final_modules:
 - scripts-per-once  #all of these run scripts at specific events. Like bootcmd, can be defined in cloud-config.
 - scripts-per-boot
 - scripts-per-instance
 - scripts-user
 - phone-home  #if defined, can make a post request to a specified url when done booting
 - final-message  #if defined, can write a specified message to the log
 - power-state-change  #if defined, can trigger stuff based on power state changes

system_info:
  distro: rhel

# vim:syntax=yaml
-----------
-----------
-----------
```

File: puppetrepo-awsexport/files/opt/gcso/awsexport/docs/AWS Convert files Inventory.txt

Fermicloud AWS Convert documentation and script files:

1. AWS Convert files Inventory.txt -- This document inventory file

2. Fermi2AWS Readme 1st.txt -- Read this file 1st to get overview of conversion process

3. fermi2aws.pdf -- Diagram of script process flow

4. fermi_VM_Convert.txt -- Manual working notes to identify script process flow

5. keypair.pem -- AWS Private key pair user file (Retrieved from AWS EC2 Console under key pairs - when created it will ask you to save, save it to this directory)

6. Convert.py -- Python script to run conversion (you can modify log output location otherwise will default to this directory)

7. ec2-get-ssh -- bash script to setup ec2 ssh authentication (loads public key pair for user on server)

8. cloud.cfg -- Cloud Init package bare-bone config file

9. Fermi_AWS_Modifications.sh -- bash script to perform aws conversion modifications to fermi golden image

10. Fermi_AWS_Resize.sh -- bash script to perform resizing of 256G qcow2 image to 12G for HVM and 3G for PV images

11. Fermi_AWS_CLI_Setup.sh -- bash script to load ec2 api tools, setup ec2 environment variables and expect package

12. Fermi_AWS_Import.sh -- bash script to perform import of raw image and ec2 steps to create instances, ami, snapshot and volumes used in conversion process

13. aws_image_convert_hvm.log -- example output log file of actual run for HVM

14. aws_image_convert_pv.log -- example output log file of actual run for PV

15. HVM Run.txt -- capture of terminal console during HVM script run

16. PV Run.txt -- capture of terminal console during PV script run

17. create_samplemime.txt -- example userdata (metadata) that creates a multipart mime txt file for cloud init

18. samplemime.txt -- example userdata (metadata) that executes on AWS vm boot up

19. Puppet Procedure to run Fermi2AWS Export.pdf -- Puppet procedures to run cron jobs

---
---
---

File: puppetrepo-awsexport/files/opt/gcso/awsexport/docs/AWS2Fermi PreProduction/AWS2Fermi Manual Procedures.txt
Importing an Amazon Web Services (AWS) PV VM into FermiCloud

Suppose to have two running VMs on Amazon EC2, both of them in the same region (e.g. us-west-2c):
one of these (e.g. i-a966649d) is an Amazon Linux image, with has got all the EC2 tools (CLI and AMI) by default;
the other (e.g. i-f61f83c2) is the Red Hat instance that you want to import on FermiCloud.
The EC2 tools are assumed well configured. This means that you have set up the environment with the variables
AWS_ACCESS_KEY, AWS_SECRET_KEY and EC2_URL.

To begin you have to connect to the first one, using for example SSH. So, from your computer's terminal, run:

```
$ ssh -i foo gcso.pem root@nn.nn.nn.nn
```

Then obtain root access using:

```
$ sudo su
```

Now you can get a raw image of the second VM with the following instructions:

```
# cd /tmp/
# ec2-stop-instances i-f61f83c2
# ec2-detach-volume vol-5f6fbe36
# ec2-attach-volume vol-5f6fbe36 -i i-a966649d -d /dev/sda2

# dd if=/dev/xvdg | cp --sparse=always /dev/stdin gcso_pv.img

# chmod 644 gcso_pv.img
# exit
$ exit
```

Then, you will find your PC's terminal. So get the image with the command:

```
$ rsync -S -z -v --progress -e "ssh -i gcso.pem"  root@54.191.161.121:/tmp/gcso_pv.img .
```

After that you can begin the conversion procedure. So:

```
$ su
# dd if=/dev/zero of=newimage.img bs=1M seek=10240 count=0
# losetup -fv newimage.img
```

Take now the path of the loop device you have created (e.g. /dev/loop1) and use it in the next instruction:

```
cfdisk /dev/loop1
```

Go ahead and create a New Primary Bootable partition. Then Write the partition table and Quit. Then find the
partition beginning, ending, number of blocks, number of cylinders, and block size using:

```
fdisk -l -u /dev/loop1
```

In our example we have:

```
Disk /dev/loop1: 10.7 GB, 10737418240 bytes
255 heads, 63 sectors/track, 1305 cylinders, total 20971520 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x00000000

Device       Boot     Start     End         Blocks         Id   System
/dev/loop1p1   *          63    20971519    10485728+  83  Linux
```

Then create a new loop device (e.g. /dev/loop2), using the beginning of the image and the block size:

```
# losetup -fv -o $((512*63)) newimage.img
```

After that create the filesystem on the image. Use the end, the beginning and the block size again:

```
# mkfs.ext3 -b 4096 /dev/loop2 $(((20971519 - 63)*512/4096))
```

Now fill the image:
If you have boot problem with the VM, maybe you have to change your kernel with a KVM compatible one. To do that
substitute all your /boot/ folder with a SLF /boot/ folder.

```
# mkdir /mnt/gcso
# mount gcso_pv.img /mnt/gcso/ -o loop
# mkdir /mnt/loop
# mkdir /mnt/loop/2
```

```
# mount /dev/loop2 /mnt/loop/2/
# cp -a /mnt/gcso/* /mnt/loop/2/
# vim /mnt/loop/2/boot/grub/menu.lst
```

At this point use this sample configuration file:

```
default=0
timeout=5
hiddenmenu
title Scientific Linux Fermi (2.6.32-431.23.3.el6.x86_64)
root (hd0,0)
kernel /boot/vmlinuz-2.6.32-431.23.3.el6.x86_64 ro root=/dev/sda1 selinux=0 enforcing=0
initrd /boot/initramfs-2.6.32-431.23.3.el6.x86_64.img

default=0
timeout=0
title Scientific Linux Fermi (2.6.32-431.23.3.el6.x86_64)
root (hd0,0)
kernel /boot/vmlinuz-2.6.32-431.23.3.el6.x86_64 ro root=UUID=17898259-6979-4bb3-9d73-26ae917e8ed9
rd_NO_LUKS rd_NO_LVM LANG=en_US.UTF-8 rd_NO_MD SYSFONT=latarcyrheb-sun16 crashkernel=auto  KEYBOARDTYPE=pc
KEYTABLE=us rd_NO_DM rhgb quiet edd=off
initrd /boot/initramfs-2.6.32-431.23.3.el6.x86_64.img
```

Then you have to edit another file:

```
# vim /mnt/loop/2/etc/fstab
```

Use for example:

```
/dev/sda1               /                       ext3    defaults        1 1
tmpfs                   /dev/shm                tmpfs   defaults        0 0
devpts                  /dev/pts                devpts  gid=5,mode=620  0 0
sysfs                   /sys                    sysfs   defaults        0 0
proc                    /proc                   proc    defaults        0 0

UUID=17898259-6979-4bb3-9d73-26ae917e8ed9 /            ext3    defaults        1 1
tmpfs  /dev/shm     tmpfs  defaults     0 0
devpts /dev/pts     devpts gid=5,mode=620     0 0
sysfs  /sys   sysfs  defaults     0 0
proc   /proc  proc   defaults     0 0
```

Now you have to configure the image to be executed on OpenNebula. Remove the Amazon context and install OpenNebula context:

```
# # echo 0 > /selinux/enforce (did not do)
# cp /data/one-context_4.6.0.rpm /mnt/loop/2/tmp/
# chroot /mnt/loop/2
# mv /etc/rc.local /etc/rc.local.old
```

```
# passwd root
# rpm -e cloud-init
# rpm -e rh-amazon-rhui-client
# cd /tmp/
# rpm -i one-context_4.6.0.rpm
# After this command you may get this error: /var/tmp/rpm-tmp.UdnuNi: line 26: cd: /etc/rcunknown.d: No
such file or directory. To fix that simply copy /S99vmcontext in the right runlevel folder (e.g. /etc/rc3.d/).
# exit
# # echo 1 > /selinux/enforce (did not do)
# umount /mnt/gcso/
# umount /mnt/loop/2
# losetup -d /dev/loop1
# losetup -d /dev/loop2
```

Then install the boot loader on the image. Using again those parameters grabbed with fdisk (cylinders, heads, sectors) you can state:

```
# grub --device-map=/dev/null
> device (hd0) /data/newimage.img
> geometry (hd0) 1305 255 63
> root (hd0,0)
> setup (hd0)
> quit
```

Make a qcow2 from the raw image:

```
$ qemu-img convert newimage.img -O qcow2 gcso_sl6_pv.qcow2
```

>>>>>>>>>>>>> Perform AWS2Fermi export procedures (see that document aws2ferm_export.txt start at >>>>>>
guestmount ... <<<<<<<<<<<<  )

** At this point a manual image setup in OpenNebula (see 'opennebtemplate.tx' for template setup) and launch creates a Kernel Panic with the following error: **

```
Kernel Panic - not syncing: Attempted to kill init!
Pid: 1 comm: init Not tainted 2.6.32-431.23.3.el6.x86_64 #1
Call Trace:
[] ? panic+0xs7/0x16f
[] ? do_exit+0x862/0x870
[] ? fput+0x25/0x30
[] ? do_group_exit+0x58/0xd0
[] ? sys_exit_group+0x17/0x20
[] ? system_call_fastpath+0x16/0x1b
```

Googling shows there are issues with grub under rhel 6.4 (http://bugs.centos.org/view.php?id=6327)

Contract ends 8/26/14 not able to finish debugging this.

---

>>> Below procedures are for OpenNebula command line launch <<<<<<
Now the image is ready to be deployed in FermiCloud with OpenNebula 3.2. Let's use the EC2 interface:

```
$ rsync -S -z -v --progress -e ssh gcso_sl6_pv.qcow2 fcl316.fnal.gov:~
$ ssh fcl316.fnal.gov
$ export ONE_LOCATION=/opt/one32
$ export PATH=$PATH:/opt/one32/bin
$ export ONE_AUTH=/cloud/login/user/.one/one_x509
$ econe-upload --url https://fermicloudpp.fnal.gov:8444/ --access-key /tmp/x509up_u502  --secret-key
/tmp/x509up_u502 ~/gcso_sl6_pv.qcow2
```

The last command returns the AMI ID (e.g.  ami-00000106). With this you can finally launch your instance:

```
$ econe-run-instances --url https://fermicloudpp.fnal.gov:8444/ --access-key /tmp/x509up_u502  --secret-key
/tmp/x509up_u502 ami-00000106 --user-data abc -t m1.small.francesco
```

The template used is m1.small.francesco. To define it you need to edit /cloud/app/one/3.2/etc/econe.conf. It has to look like:

```
# Configuration for the image repository
# IMAGE_DIR will store the Cloud images, check space!
IMAGE_DIR=/var/lib/one/local/pub_scratch

# OpenNebula sever contact information
:one_xmlrpc: http://fermicloudpp.fnal.gov:2633/RPC2

# Host and port where econe server will run
:server: localhost
:port: 4567

# SSL proxy that serves the API (set if is being used)
:ssl_server: https://fermicloudpp.fnal.gov:8444/

# Authentication driver for incomming requests
#   ec2, default Acess key and Secret key scheme
#   x509, for x509 certificates based authentication
#:auth: ec2
:auth: x509

# Authentication driver to communicate with OpenNebula core
#   cipher, for symmetric cipher encryption of tokens
#   x509, for x509 certificate encryption of tokens
#:core_auth: cipher
:core_auth: x509

# VM types allowed and its template file (inside templates directory)
```

```
:instance_types:
      :m1.small:
            :template: m1.small.erb
      :m1.small.francesco:
            :template: m1.small.francesco.erb
```

Then you have to edit the file /cloud/app/one/3.2/etc/ec2query_templates/m1.small.francesco.erb. For example you can use:

```
NAME   = eco-vm
CPU    = 1
VCPU    = 1
MEMORY = 1024
OS = [ ARCH = x86_64 ]

DISK = [ IMAGE_ID   = <%= erb_vm_info[:img_id] %>]

DISK  = [
  type     = swap,
  size     = 5120]


#NIC=[NETWORK_ID=<EC2-VNET-ID>]
NIC=[NETWORK_ID=0,
     MODEL = virtio]

#IMAGE_ID      = <%= erb_vm_info[:ec2_img_id]    %>
#INSTANCE_TYPE = <%= erb_vm_info[:instance_type] %>

FEATURES=[ acpi="yes" ]

GRAPHICS = [
  type    = "vnc",
  listen  = "127.0.0.1",
  port    = "-1",
  autoport = "yes",
  keymap="en-us" ]


#<% if erb_vm_info[:user_data] %>
#CONTEXT = [
#       EC2_USER_DATA="<%= erb_vm_info[:user_data] %>",
#       TARGET="hdc"
#       ]
#<% end %>

<% if erb_vm_info[:user_data] %>
CONTEXT=[
```

```
        CTX_USER="$USER[TEMPLATE]",
        ETH0_DNS="$NETWORK[DNS, NETWORK_ID=0]",
        ETH0_GATEWAY="$NETWORK[GATEWAY, NETWORK_ID=0]",
        ETH0_IP="$NIC[IP, NETWORK_ID=0]",
        ETH0_MASK="$NETWORK[NETWORK_MASK, NETWORK_ID=0]",
        FILES=/opt/one32/share/init.sh,
        IB=false,
        ROOT_PUBKEY=id_dsa.pub,
        SSH_PUBLIC_KEY=id_dsa.pub,
        TARGET=hdc,
        USERNAME=opennebula,
        EC2_USER_DATA="<%= erb_vm_info[:user_data] %>"
      ]
      <% else %>
      CONTEXT=[
        CTX_USER="$USER[TEMPLATE]",
        ETH0_DNS="$NETWORK[DNS, NETWORK_ID=0]",
        ETH0_GATEWAY="$NETWORK[GATEWAY, NETWORK_ID=0]",
        ETH0_IP="$NIC[IP, NETWORK_ID=0]",
        ETH0_MASK="$NETWORK[NETWORK_MASK, NETWORK_ID=0]",
        FILES=/opt/one32/share/init.sh,
        IB=false,
        ROOT_PUBKEY=id_dsa.pub,
        SSH_PUBLIC_KEY=id_dsa.pub,
        TARGET=hdc,
        USERNAME=opennebula
      ]
      <% end %>

      REQUIREMENTS = "HYPERVISOR=\"kvm\""
      RANK = "FREEMEMORY"


-----------
-----------
-----------


File: puppetrepo-awsexport/files/opt/gcso/awsexport/docs/AWS2Fermi PreProduction/AWS2Fermi_Export.txt
# Export HVM image from AWS back to Fermi

# the following original Fermicloud files should be copied from a current working Opennebula vm and saved to a
temp work area (/data/readd) for resuse
[root@fermicloud103 readd]# ls
.credentials auto.master  auto.misc  fstab  grub.conf  hosts  hosts.allow  hosts.deny  network  one-context.prod
resolv.conf  sshd_config  vmcontext.pp  vmcontext.prod  yp.conf

# AWS ec2 api and cli tools need to be installed on worker vm
```

```
ec2-create-instance-export-task i-eb5218e0 -e Citrix -f VHD -b fcloudexport

ec2-describe-export-tasks

# wait until status completed then download VHD image

cd /data
aws s3 sync s3://fcloudexport .

# Delete temp S3 bucket and VHD image

aws s3 rb s3://fcloudexport --force

# convert downloaded vhd to qcow2

qemu-img convert export-i-fgpsizu8.vhd -O qcow2 gcso_sl6_hvm.qcow2

# Access the image to add fermicloud dependencies

guestmount -a gcso_sl6_hvm.qcow2 -m /dev/sda1 /mnt

>>>>>>>>>>>>>> Start up from here for PV image convert <<<<<<<<<<<<<<<<<<<

guestmount -a gcso_sl6_pv.qcow2 -m /dev/sda1 /mnt


mkdir /mnt/data
mkdir /mnt/data/readd
cp /data/readd/* /mnt/data/readd
cp /data/readd/.credentials /mnt/data/readd

# chroot to mnt image
chroot /mnt

# reverse aws modifications made back to fermicloud dependencies ...

# if home.orig exists
mv home home.aws
mv home.orig home

# For Glidein Images do
yum -y remove glideinwms-vm-ec2
yum -y install glideinwms-vm-one

# Modify network
cd /etc/sysconfig
cp -f /data/readd/network .
```

```
# Add back fermi network resolv and hosts
cd /etc
cp -f /data/readd/resolv.conf .
cp -f /data/readd/hosts .
cp -f /data/readd/hosts.allow .
cp -f /data/readd/hosts.deny .

# Add back fermi specifics
cp -f /data/readd/yp.conf .
cp -f /data/readd/auto.master .
cp -f /data/readd//auto.misc .

cd /etc/init.d
cp -f /data/readd/.credentials .
# for production only
cp -f /data/readd/one-context.prod one-context

# Recreate symbolic links
ln -s /etc/init.d/.credentials /etc/rc.d/rc3.d/K99.credentials
# for production only
ln -s /etc/init.d/one-context /etc/rc.d/rc3.d/S09one-context
ln -s /etc/init.d/one-context /etc/rc.d/rc3.d/K84one-context

# copy vmcontext from pp for pp testing and prod for prod testing
cd /etc/init.d

# for preproduction only
cp -f /data/readd/vmcontext.pp vmcontext

# for production only
cp -f /data/readd/vmcontext.prod vmcontext

# Delete ifcfg-eth0
cd /etc/sysconfig/network-scripts
rm -f ifcfg-eth0

# Modify sshd_config
cd /etc/ssh
cp -f /data/readd/sshd_config .

# modify grub.conf using uuid

cd /boot/grub
cp -f /data/readd/grub.conf .

# default=0
# timeout=0
```

```
# title Scientific Linux Fermi (2.6.32-431.23.3.el6.x86_64)
# root (hd0,0)
# kernel /boot/vmlinuz-2.6.32-431.23.3.el6.x86_64 ro root=UUID=17898259-6979-4bb3-9d73-26ae917e8ed9 rd_NO_LUKS
rd_NO_LVM LANG=en_US.UTF-8 rd_NO_MD SYSFONT=latarcyrheb-sun16 crashkernel=auto  KEYBOARDTYPE=pc KEYTABLE=us
rd_NO_DM rhgb quiet edd=off
# initrd /boot/initramfs-2.6.32-431.23.3.el6.x86_64.img

# modify fstab using uuid

cd /etc
cp -f /data/readd/fstab .

# UUID=17898259-6979-4bb3-9d73-26ae917e8ed9 /            ext3    defaults        1 1
# tmpfs      /dev/shm    tmpfs  defaults     0 0
# devpts     /dev/pts    devpts gid=5,mode=620     0 0
# sysfs      /sys  sysfs  defaults     0 0
# proc /proc  proc  defaults     0 0

# Modify /sbin/chkconfig services
/sbin/chkconfig ec2-get-ssh off
/sbin/chkconfig rpcbind on
/sbin/chkconfig postfix on
# /sbin/chkconfig autofs off
/sbin/chkconfig vmcontext on
# Prevent 10 minute automatic vm shutdown for testing (**turn back on after testing**)
# /sbin/chkconfig glideinwms-pilot off

# Clear history
history -c

# exit chroot
exit

# Unmount image
cd /data
fusermount -u /mnt
-----------
-----------
-----------

File: puppetrepo-awsexport/files/opt/gcso/awsexport/docs/AWS2Fermi PreProduction/opennebtemplate.txt
Update image properties

BUS=virtio
DEV_PREFIX=hd
DRIVER=qcow2

------------------------------------
```

```
Update template properties (change IMAGE_ID= and TEMPLATE_ID= when creating new images under OpenNebula)

CONTEXT=[
  CTX_USER="$USER[TEMPLATE]",
  ETH0_DNS="$NETWORK[DNS, NETWORK=\"DynamicIP\"]",
  ETH0_GATEWAY="$NETWORK[GATEWAY, NETWORK=\"DynamicIP\"]",
  ETH0_IP="$NIC[IP, NETWORK=\"DynamicIP\"]",
  ETH0_MASK="$NETWORK[NETWORK_MASK, NETWORK=\"DynamicIP\"]",
  ETH0_NETWORK="$NETWORK[NETWORK_ADDRESS, NETWORK=\"DynamicIP\"]",
  FILES="/cloud/images/OpenNebula/scripts/one3.2/contextualization/init.sh
/cloud/images/OpenNebula/scripts/one3.2/contextualization/credentials.sh
/cloud/images/OpenNebula/scripts/one3.2/contextualization/kerberos.sh",
  GATEWAY="$NETWORK[GATEWAY, NETWORK=\"DynamicIP\"]",
  INIT_SCRIPTS="init.sh credentials.sh kerberos.sh",
  IP_PUBLIC="$NIC[IP, NETWORK=\"DynamicIP\"]",
  NETMASK="$NETWORK[NETWORK_MASK, NETWORK=\"DynamicIP\"]",
  NETWORK=YES,
  ROOT_PUBKEY=id_dsa.pub,
  TARGET=hdc,
  USERNAME=opennebula,
  USER_PUBKEY=id_dsa.pub ]
CPU=0.5
DISK=[
  IMAGE_ID=223,
  IMAGE_UNAME=oneadmin,
  TARGET=vda ]
DISK=[
  SIZE=2048,
  TARGET=vdb,
  TYPE=swap ]
FEATURES=[
  ACPI=yes ]
GRAPHICS=[
  AUTOPORT=yes,
  KEYMAP=en-us,
  LISTEN=127.0.0.1,
  PORT=-1,
  TYPE=vnc ]
MEMORY=2048
NAME="aws hvm export"
NIC=[
  MODEL=virtio,
  NETWORK=DynamicIP,
  NETWORK_UNAME=oneadmin ]
NPTYPE=NPERNLM
OS=[
  ARCH=x86_64,
```

```
  BOOT=hd ]
RANK=FREEMEMORY
RAW=[
  DATA="
                <devices>
                <serial type='pty'>
                        <target port='0'/>
                </serial>
                <console type='pty'>
                <target type='serial' port='0'/>
                </console>
                </devices>",
  TYPE=kvm ]
REQUIREMENTS="HYPERVISOR=\"kvm\" & HOSTNAME=\"fcl010.fnal.gov\""
TEMPLATE_ID=139
VCPU=1
-----------
-----------
-----------
```

File: puppetrepo-awsexport/files/opt/gcso/awsexport/docs/Fermi2AWS Readme 1st.txt
Date: August 14, 2014

Steve/Gerard/Nick,

I have successfully implemented a conversion of a Fermicloud Golden image to a launched AWS Para virtual image using a python script with several subprocess bash scripts and puppet modules. The entire process takes about 60 to 84 minutes for a PV and HVM conversion. The actual processing time is less than that, but asynchronous conditional sleeps on the amazon side were implemented to make sure those ec2 steps completed before going on to the next step.

If you would like to test this script, you can run the puppet apply procedures. (See 'Puppet Procedure to run Fermi2AWS Export.pdf'). Change the parameters as described in that document, prior to running the script.

**Note: the following 4 steps are a pre-requisite prior to running this puppet module:

1. kinit username

2. Obtain a fermicloud worker vm instance from the OpenNebula Fermicloud. This is used as a interim work area for the conversions. Identify the 3 digit assigned number to the worker vm as in fermiicloud(nnn).fnal.gov.

3. This script also requires 5 parameters and files to be obtained ahead of time from the AWS Console. They are:
   - <aws owner security pem name> (keypair_name.pem) (Download this file from the AWS console to your terminal root directory. *Important* You must run 'chmod 400 keypair_name.pem' to allow a ssh session to connect to a aws instance.)
   - <aws instance id of the HVM worker image> (use 'hvm' here to create AWS HVM worker before creating AWS PV's, Otherwise view ec2 hvm under aws ec2 console and get i-xxxxxxx for hvm worker)

```
   - <aws owner keypair name> (create in aws console ec2-key pairs - create key pair)
   - <aws key> (create in aws console IAM Users - Security Credentials tab and download)
   - <aws secret key> (create in aws console IAM Users - Security Credentials tab and download)
   (all of these are obtained from the AWS console under EC2 instances and the IAM Users-Security Credentials tab
prior to running this script).

4. Run the puppet procedure with 15 positional parameters and adjust your crontab start time (as documented in
'Puppet Procedure to run Fermi2AWS Export.pdf').

A log file (aws_image_convert.log) is created for each run in the /opt/gcso/awsexport directory for review. The
Python module can be modified to point that log file to another writeable directory that GCSO's monitoring system
runs from, if you wish to have it monitored there.

To support the use of userdata (metadata), to install 3rd party packages, you will need to create a multipart
mime txt file to be used in advanced userdata when launching a AWS instance from a AMI. (see
create_samplemime.txt and samplemime.txt files)

Kirk
-----------
-----------
-----------


File: puppetrepo-awsexport/files/opt/gcso/awsexport/docs/HVM Run.txt
(Run Fermi's Kerberos Initialize before running this script: kinit userid).
  This script takes twelve (12) arguments:  <script dir location>, <fermicloud
  worker vm number>, <fermicloud vm name>, <fermicloud vm owner>, <aws image
  name>, <aws image owner>, <aws instance type>, <aws key>, <aws secret key>,
  <aws pem name>, <aws worker instance id>, and <aws owner keypair name>. A
  pre-requisite is to have a Fermicloud worker vm setup ahead of time
  (root@fermicloudnnn.fnal.gov) with a /data directory created to store files.
  The script will accept a VM image that is pre-loaded on your
  username@fermicloud.fnal.gov root directory, (it must have a qcow2 extension),
  and will make a worker copy of the image ready for AWS conversion, then
  convert the image (fermi cleanse and resize) and import to AWS by the
  specified <aws image name>. The script requires 5 parameters to be obtained
  ahead of time from the AWS Console. They are: <aws owner security pem name>
  <aws instance id of the HVM worker image, use 'hvm' here to create HVM worker
  before creating PV's.> <aws owner keypair name> <aws key> and <aws secret key>
  which are obtained from the AWS console under EC2 instances and the IAM
  security tab prior to running this script. For example, to convert a
  Fermicloud VM named gcso_slf6 with owner oneadmin, and to create a AWS image
  named SLF6Vanilla, with owner oneadmin, and a instance type of m3.medium and
  with a worker vm of fermicloud103.fnal.gov you would run the following script:
  **
 ./Convert.py /Users/terminalroot 103 gcso_slf6 oneadmin SLF6Vanilla
  oneadmin m3.medium awskey awssecretkey awsowner.pem i-hvminstanceid
  aws_owner_keypair_name
```

```
  **
 ...Available.AWS.Instance.Types.listed.below...
  Type......vCPU..ECU..Memory.(GiB)..Instance Storage (GB)..Cost per hour
  General.Purpose.-.Current.Generation
  t2.micro....1....Variable....1....EBSOnly......$0.013 per Hour
  t2.small....1....Variable....2....EBSOnly......$0.026 per Hour
  t2.medium...2....Variable....4....EBSOnly......$0.052 per Hour
  m3.medium...1.......3......3.75...1.x.4.SSD....$0.070 per Hour
  m3.large....2......6.5.....7.5....1.x.32.SSD...$0.140 per Hour
  m3.xlarge...4......13.......15....2.x.40.SSD...$0.280 per Hour
  m3.2xlarge..8......26.......30....2.x.80.SSD...$0.560 per Hour
  Compute.Optimized.-.Current.Generation
  c3.large....2.......7......3.75...2.x.16.SSD...$0.105 per Hour
  c3.xlarge...4......14......7.5....2.x.40.SSD...$0.210 per Hour
  c3.2xlarge..8......28.......15....2.x.80.SSD...$0.420 per Hour
  c3.4xlarge..16.....55.......30....2.x.160.SSD..$0.840 per Hour
  c3.8xlarge..32....108.......60....2.x.320.SSD..$1.680 per Hour
  GPU.Instances.-.Current.Generation
  g2.2xlarge..8......26.......15........60.SSD...$0.650 per Hour
  Memory.Optimized.-.Current.Generation
  r3.large....2......6.5......15....1.x.32.SSD...$0.175 per Hour
  r3.xlarge...4.....13.......30.5...1.x.80.SSD...$0.350 per Hour
  r3.2xlarge..8.....26........61....1.x.160.SSD..$0.700 per Hour
  r3.4xlarge..16....52.......122....1.x.320.SSD..$1.400 per Hour
  r3.8xlarge..32....104......244....2.x.320.SSD..$2.800 per Hour
  Storage.Optimized.-.Current.Generation
  i2.xlarge...4.....14.......30.5...1.x.800.SSD..$0.853 per Hour
  i2.2xlarge..8.....27.......61.....2.x.800.SSD..$1.705 per Hour
  i2.4xlarge..16....53.......122....4.x.800.SSD..$3.410 per Hour
  i2.8xlarge..32....104......244....8.x.800.SSD..$6.820 per Hour
  hs1.8xlarge.16....35.......117...24.x.2048.....$4.600 per Hour

Arguments supplied:
('location of all scripts->', '/Users/kshallcross')
('fermicloud work vm number->', '103')
('fermicloud vm name->', 'gcso_sl6')
('fermicloud vm owner->', 'kirkshal')
('aws image name->', 'GCSO_SL6_HVM')
('aws image owner->', 'kirkshal')
('aws instance type->', 'm3.medium')
('aws key->', 'xxxxx')
('aws secret key->', 'xxxxxx')
('aws owner pem name->', 'khs-fermi.pem')
('aws owner keypair name->', 'khs-fermi')
('aws worker instance id from aws console->', 'hvm')
Starting AWS VM conversion. This job can take up to 52 minutes for PV and 77 for HVM...
Copying the Golden Fermicloud VM image takes under 1 minute...
Copying took 0.908 minutes.
```

```
Resizing the worker Fermicloud VM image takes up to 20 minutes for PV and 10 for HVM...
Resizing took 9.787 minutes.
Converting the worker Fermicloud VM image takes over 18 minutes...
Converting took 19.033 minutes.
Importing the raw image to AWS takes up to 13 minutes for PV and 48 for HVM...
Importing took 44.233 minutes.
Completed AWS VM conversion.
Job took 73.961 minutes. Thank you.
-----------
-----------
-----------


File: puppetrepo-awsexport/files/opt/gcso/awsexport/docs/PV Run.txt
(Run Fermi's Kerberos Initialize before running this script: kinit userid).
  This script takes twelve (12) arguments:  <script dir location>, <fermicloud
  worker vm number>, <fermicloud vm name>, <fermicloud vm owner>, <aws image
  name>, <aws image owner>, <aws instance type>, <aws key>, <aws secret key>,
  <aws pem name>, <aws worker instance id>, and <aws owner keypair name>. A
  pre-requisite is to have a Fermicloud worker vm setup ahead of time
  (root@fermicloudnnn.fnal.gov) with a /data directory created to store files.
  The script will accept a VM image that is pre-loaded on your
  username@fermicloud.fnal.gov root directory, (it must have a qcow2 extension),
  and will make a worker copy of the image ready for AWS conversion, then
  convert the image (fermi cleanse and resize) and import to AWS by the
  specified <aws image name>. The script requires 5 parameters to be obtained
  ahead of time from the AWS Console. They are: <aws owner security pem name>
  <aws instance id of the HVM worker image, use 'hvm' here to create HVM worker
  before creating PV's.> <aws owner keypair name> <aws key> and <aws secret key>
  which are obtained from the AWS console under EC2 instances and the IAM
  security tab prior to running this script. For example, to convert a
  Fermicloud VM named gcso_slf6 with owner oneadmin, and to create a AWS image
  named SLF6Vanilla, with owner oneadmin, and a instance type of m3.medium and
  with a worker vm of fermicloud103.fnal.gov you would run the following script:
  **
 ./Convert.py /Users/terminalroot 103 gcso_slf6 oneadmin SLF6Vanilla
  oneadmin m3.medium awskey awssecretkey awsowner.pem i-hvminstanceid
  aws_owner_keypair_name
  **
 ...Available.AWS.Instance.Types.listed.below...
  Type......vCPU..ECU..Memory.(GiB)..Instance Storage (GB)..Cost per hour
  General.Purpose.-.Current.Generation
  t2.micro....1....Variable....1....EBSOnly......$0.013 per Hour
  t2.small....1....Variable....2....EBSOnly......$0.026 per Hour
  t2.medium...2....Variable....4....EBSOnly......$0.052 per Hour
  m3.medium...1.......3......3.75...1.x.4.SSD....$0.070 per Hour
  m3.large....2......6.5.....7.5....1.x.32.SSD...$0.140 per Hour
  m3.xlarge...4......13.......15....2.x.40.SSD...$0.280 per Hour
```

```
  m3.2xlarge..8......26.......30....2.x.80.SSD...$0.560 per Hour
  Compute.Optimized.-.Current.Generation
  c3.large....2.......7......3.75...2.x.16.SSD...$0.105 per Hour
  c3.xlarge...4......14......7.5....2.x.40.SSD...$0.210 per Hour
  c3.2xlarge..8......28.......15....2.x.80.SSD...$0.420 per Hour
  c3.4xlarge..16.....55.......30....2.x.160.SSD..$0.840 per Hour
  c3.8xlarge..32....108.......60....2.x.320.SSD..$1.680 per Hour
  GPU.Instances.-.Current.Generation
  g2.2xlarge..8......26.......15........60.SSD...$0.650 per Hour
  Memory.Optimized.-.Current.Generation
  r3.large....2......6.5......15....1.x.32.SSD...$0.175 per Hour
  r3.xlarge...4.....13.......30.5...1.x.80.SSD...$0.350 per Hour
  r3.2xlarge..8.....26........61....1.x.160.SSD..$0.700 per Hour
  r3.4xlarge..16....52.......122....1.x.320.SSD..$1.400 per Hour
  r3.8xlarge..32....104......244....2.x.320.SSD..$2.800 per Hour
  Storage.Optimized.-.Current.Generation
  i2.xlarge...4.....14.......30.5...1.x.800.SSD..$0.853 per Hour
  i2.2xlarge..8.....27.......61.....2.x.800.SSD..$1.705 per Hour
  i2.4xlarge..16....53.......122....4.x.800.SSD..$3.410 per Hour
  i2.8xlarge..32....104......244....8.x.800.SSD..$6.820 per Hour
  hs1.8xlarge.16....35.......117...24.x.2048.....$4.600 per Hour

Arguments supplied:
('location of all scripts->', '/Users/kshallcross')
('fermicloud work vm number->', '103')
('fermicloud vm name->', 'gcso_sl6')
('fermicloud vm owner->', 'kirkshal')
('aws image name->', 'GCSO_SL6_PV')
('aws image owner->', 'kirkshal')
('aws instance type->', 'm3.medium')
('aws key->', 'xxx')
('aws secret key->', 'xxxxx')
('aws owner pem name->', 'khs-fermi.pem')
('aws owner keypair name->', 'khs-fermi')
('aws worker instance id from aws console->', 'i-94fe889f')
Starting AWS VM conversion. This job can take up to 53 minutes for PV and 75 for HVM...
Copying the Golden Fermicloud VM image takes under 1 minute...
Copying took 0.906 minutes.
Resizing the worker Fermicloud VM image takes up to 20 minutes for PV and 10 for HVM...
Resizing took 19.422 minutes.
Converting the worker Fermicloud VM image takes over 19 minutes...
Converting took 20.174 minutes.
Importing the raw image to AWS takes up to 13 minutes for PV and 45 for HVM...
Importing took 12.136 minutes.
Completed AWS VM conversion.
Job took 52.638 minutes. Thank you.
-----------
-----------
```

```
-----------

File: puppetrepo-awsexport/files/opt/gcso/awsexport/docs/aws_image_convert_hvm.log
Aug 02 14:52:36 INFO     Begin script run
Aug 02 14:52:36 INFO     Start: Copying the selected Fermicloud VM image. Function copy_to_image_location.
Aug 02 14:53:31 INFO     Stop: Completed Copying the selected Fermicloud VM image. Function
copy_to_image_location.
Aug 02 14:53:31 INFO     Start: Resizing the worker Fermicloud VM image. Function resize_image.
*stdin*:2: libguestfs: error: e2fsck_f: /dev/sda1: 94023/786432 files (0.3% non-contiguous), 550839/3145728
blocks
Formatting 'gcso_sl6.raw', fmt=raw size=12888047616
Examining gcso_sl6.qcow2tmp ...
**********

Summary of changes:

/dev/sda1: This partition will be left alone.

There is a surplus of 864.0K.  The surplus space is not large enough
for an extra partition to be created and so it will just be ignored.

**********
Setting up initial partition table on gcso_sl6.raw ...
Copying /dev/sda1 ...

Resize operation completed with no errors.  Before deleting the old
disk, carefully check that the resized disk boots and works correctly.
image: gcso_sl6.raw
file format: raw
virtual size: 12G (12888047616 bytes)
disk size: 12G
Aug 02 15:03:18 INFO     Stop: Completed Resizing the worker Fermicloud VM image. Function resize_image.
Aug 02 15:03:18 INFO     Start: Converting the worker Fermicloud VM image. Function convert_image.
warning: /var/tmp/rpm-tmp.UDfdjD: Header V3 RSA/SHA256 Signature, key ID 0608b895: NOKEY
Retrieving http://download.fedoraproject.org/pub/epel/6/x86_64/epel-release-6-8.noarch.rpm
Preparing...                ##################################################
epel-release                ##################################################
Loaded plugins: security
Setting up Install Process
Resolving Dependencies
--> Running transaction check
---> Package cloud-init.noarch 0:0.7.4-2.el6 will be installed
--> Processing Dependency: python-boto >= 2.6.0 for package: cloud-init-0.7.4-2.el6.noarch
--> Processing Dependency: python-requests for package: cloud-init-0.7.4-2.el6.noarch
--> Processing Dependency: python-prettytable for package: cloud-init-0.7.4-2.el6.noarch
--> Processing Dependency: python-jsonpatch for package: cloud-init-0.7.4-2.el6.noarch
--> Processing Dependency: python-configobj for package: cloud-init-0.7.4-2.el6.noarch
--> Processing Dependency: python-cheetah for package: cloud-init-0.7.4-2.el6.noarch
```

```
--> Processing Dependency: python-argparse for package: cloud-init-0.7.4-2.el6.noarch
--> Processing Dependency: policycoreutils-python for package: cloud-init-0.7.4-2.el6.noarch
--> Processing Dependency: libselinux-python for package: cloud-init-0.7.4-2.el6.noarch
--> Processing Dependency: PyYAML for package: cloud-init-0.7.4-2.el6.noarch
--> Running transaction check
---> Package PyYAML.x86_64 0:3.10-3.el6 will be installed
--> Processing Dependency: libyaml-0.so.2()(64bit) for package: PyYAML-3.10-3.el6.x86_64
---> Package libselinux-python.x86_64 0:2.0.94-5.3.el6 will be installed
---> Package policycoreutils-python.x86_64 0:2.0.83-19.30.el6 will be installed
--> Processing Dependency: libsemanage-python >= 2.0.43-4 for package: policycoreutils-python-2.0.83-
19.30.el6.x86_64
--> Processing Dependency: audit-libs-python >= 1.4.2-1 for package: policycoreutils-python-2.0.83-
19.30.el6.x86_64
--> Processing Dependency: setools-libs-python for package: policycoreutils-python-2.0.83-19.30.el6.x86_64
--> Processing Dependency: libcgroup for package: policycoreutils-python-2.0.83-19.30.el6.x86_64
---> Package python-argparse.noarch 0:1.2.1-2.el6 will be installed
---> Package python-boto.noarch 0:2.27.0-1.el6 will be installed
---> Package python-cheetah.x86_64 0:2.4.1-1.el6 will be installed
--> Processing Dependency: python-pygments for package: python-cheetah-2.4.1-1.el6.x86_64
--> Processing Dependency: python-markdown for package: python-cheetah-2.4.1-1.el6.x86_64
---> Package python-configobj.noarch 0:4.6.0-3.el6 will be installed
---> Package python-jsonpatch.noarch 0:1.2-2.el6 will be installed
--> Processing Dependency: python-jsonpointer for package: python-jsonpatch-1.2-2.el6.noarch
---> Package python-prettytable.noarch 0:0.7.2-1.el6 will be installed
---> Package python-requests.noarch 0:1.1.0-4.el6 will be installed
--> Processing Dependency: python-urllib3 for package: python-requests-1.1.0-4.el6.noarch
--> Processing Dependency: python-ordereddict for package: python-requests-1.1.0-4.el6.noarch
--> Processing Dependency: python-chardet for package: python-requests-1.1.0-4.el6.noarch
--> Running transaction check
---> Package audit-libs-python.x86_64 0:2.2-2.el6 will be installed
---> Package libcgroup.x86_64 0:0.37-7.el6 will be installed
---> Package libsemanage-python.x86_64 0:2.0.43-4.2.el6 will be installed
---> Package libyaml.x86_64 0:0.1.6-1.el6 will be installed
---> Package python-chardet.noarch 0:2.0.1-1.el6 will be installed
---> Package python-jsonpointer.noarch 0:1.0-3.el6 will be installed
---> Package python-markdown.noarch 0:2.0.1-3.1.el6 will be installed
---> Package python-ordereddict.noarch 0:1.1-2.el6 will be installed
---> Package python-pygments.noarch 0:1.1.1-1.el6 will be installed
--> Processing Dependency: python-setuptools for package: python-pygments-1.1.1-1.el6.noarch
---> Package python-urllib3.noarch 0:1.5-7.el6 will be installed
--> Processing Dependency: python-six for package: python-urllib3-1.5-7.el6.noarch
--> Processing Dependency: python-backports-ssl_match_hostname for package: python-urllib3-1.5-7.el6.noarch
---> Package setools-libs-python.x86_64 0:3.3.7-4.el6 will be installed
--> Processing Dependency: setools-libs = 3.3.7-4.el6 for package: setools-libs-python-3.3.7-4.el6.x86_64
--> Processing Dependency: libqpol.so.1(VERS_1.3)(64bit) for package: setools-libs-python-3.3.7-4.el6.x86_64
--> Processing Dependency: libpoldiff.so.1(VERS_1.3)(64bit) for package: setools-libs-python-3.3.7-4.el6.x86_64
--> Processing Dependency: libapol.so.4(VERS_4.1)(64bit) for package: setools-libs-python-3.3.7-4.el6.x86_64
--> Processing Dependency: libqpol.so.1(VERS_1.2)(64bit) for package: setools-libs-python-3.3.7-4.el6.x86_64
```

```
--> Processing Dependency: libqpol.so.1(VERS_1.4)(64bit) for package: setools-libs-python-3.3.7-4.el6.x86_64
--> Processing Dependency: libseaudit.so.4(VERS_4.2)(64bit) for package: setools-libs-python-3.3.7-4.el6.x86_64
--> Processing Dependency: libpoldiff.so.1(VERS_1.2)(64bit) for package: setools-libs-python-3.3.7-4.el6.x86_64
--> Processing Dependency: libsefs.so.4(VERS_4.0)(64bit) for package: setools-libs-python-3.3.7-4.el6.x86_64
--> Processing Dependency: libapol.so.4(VERS_4.0)(64bit) for package: setools-libs-python-3.3.7-4.el6.x86_64
--> Processing Dependency: libseaudit.so.4(VERS_4.1)(64bit) for package: setools-libs-python-3.3.7-4.el6.x86_64
--> Processing Dependency: libseaudit.so.4()(64bit) for package: setools-libs-python-3.3.7-4.el6.x86_64
--> Processing Dependency: libqpol.so.1()(64bit) for package: setools-libs-python-3.3.7-4.el6.x86_64
--> Processing Dependency: libapol.so.4()(64bit) for package: setools-libs-python-3.3.7-4.el6.x86_64
--> Processing Dependency: libpoldiff.so.1()(64bit) for package: setools-libs-python-3.3.7-4.el6.x86_64
--> Processing Dependency: libsefs.so.4()(64bit) for package: setools-libs-python-3.3.7-4.el6.x86_64
--> Running transaction check
---> Package python-backports-ssl_match_hostname.noarch 0:3.4.0.2-1.el6 will be installed
--> Processing Dependency: python-backports for package: python-backports-ssl_match_hostname-3.4.0.2-1.el6.noarch
---> Package python-setuptools.noarch 0:0.6.10-3.el6 will be installed
---> Package python-six.noarch 0:1.6.1-1.el6 will be installed
---> Package setools-libs.x86_64 0:3.3.7-4.el6 will be installed
--> Running transaction check
---> Package python-backports.x86_64 0:1.0-3.el6 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package                                Arch     Version            Repository
                                                                           Size
================================================================================
Installing:
 cloud-init                             noarch   0.7.4-2.el6        epel   487 k
Installing for dependencies:
 PyYAML                                 x86_64   3.10-3.el6         epel   157 k
 audit-libs-python                      x86_64   2.2-2.el6          slf     58 k
 libcgroup                              x86_64   0.37-7.el6         slf    110 k
 libselinux-python                      x86_64   2.0.94-5.3.el6     slf    201 k
 libsemanage-python                     x86_64   2.0.43-4.2.el6     slf     80 k
 libyaml                                x86_64   0.1.6-1.el6        epel    52 k
 policycoreutils-python                 x86_64   2.0.83-19.30.el6   slf    341 k
 python-argparse                        noarch   1.2.1-2.el6        epel    48 k
 python-backports                       x86_64   1.0-3.el6          epel   5.3 k
 python-backports-ssl_match_hostname    noarch   3.4.0.2-1.el6      epel    12 k
 python-boto                            noarch   2.27.0-1.el6       epel   1.6 M
 python-chardet                         noarch   2.0.1-1.el6        epel   225 k
 python-cheetah                         x86_64   2.4.1-1.el6        slf    364 k
 python-configobj                       noarch   4.6.0-3.el6        slf    181 k
 python-jsonpatch                       noarch   1.2-2.el6          epel    14 k
 python-jsonpointer                     noarch   1.0-3.el6          epel   9.2 k
 python-markdown                        noarch   2.0.1-3.1.el6      slf    117 k
 python-ordereddict                     noarch   1.1-2.el6          epel   7.6 k
```

```
 python-prettytable                     noarch   0.7.2-1.el6        epel    37 k
 python-pygments                        noarch   1.1.1-1.el6        slf    561 k
 python-requests                        noarch   1.1.0-4.el6        epel    71 k
 python-setuptools                      noarch   0.6.10-3.el6       slf    335 k
 python-six                             noarch   1.6.1-1.el6        epel    25 k
 python-urllib3                         noarch   1.5-7.el6          epel    41 k
 setools-libs                           x86_64   3.3.7-4.el6        slf    399 k
 setools-libs-python                    x86_64   3.3.7-4.el6        slf    221 k

Transaction Summary
================================================================================
Install      27 Package(s)

Total download size: 5.6 M
Installed size: 25 M
Downloading Packages:
--------------------------------------------------------------------------------
Total                                               562 kB/s | 5.6 MB     00:10
Retrieving key from file:///etc/pki/rpm-gpg/RPM-GPG-KEY-EPEL-6
warning: rpmts_HdrFromFdno: Header V3 RSA/SHA256 Signature, key ID 0608b895: NOKEY
Importing GPG key 0x0608B895:
 Userid : EPEL (6) <epel@fedoraproject.org>
 Package: slf-release-6.4-1.x86_64 (@anaconda-ScientificLinuxFermi-201304091346.x86_64/6)
 From   : /etc/pki/rpm-gpg/RPM-GPG-KEY-EPEL-6
Running rpm_check_debug
Running Transaction Test
Transaction Test Succeeded
Running Transaction
Warning: RPMDB altered outside of yum.

  Installing : python-ordereddict-1.1-2.el6.noarch                          1/27

  Installing : libselinux-python-2.0.94-5.3.el6.x86_64                      2/27

  Installing : python-argparse-1.2.1-2.el6.noarch                           3/27

  Installing : libsemanage-python-2.0.43-4.2.el6.x86_64                     4/27

  Installing : python-boto-2.27.0-1.el6.noarch                              5/27

  Installing : python-six-1.6.1-1.el6.noarch                                6/27

  Installing : python-configobj-4.6.0-3.el6.noarch                          7/27

  Installing : python-setuptools-0.6.10-3.el6.noarch                        8/27

  Installing : python-pygments-1.1.1-1.el6.noarch                           9/27
```

```
  Installing : python-markdown-2.0.1-3.1.el6.noarch                      10/27
  Installing : python-cheetah-2.4.1-1.el6.x86_64                         11/27
  Installing : python-chardet-2.0.1-1.el6.noarch                         12/27
  Installing : python-prettytable-0.7.2-1.el6.noarch                     13/27
  Installing : libcgroup-0.37-7.el6.x86_64                               14/27
  Installing : setools-libs-3.3.7-4.el6.x86_64                           15/27
  Installing : setools-libs-python-3.3.7-4.el6.x86_64                    16/27
  Installing : libyaml-0.1.6-1.el6.x86_64                                17/27
  Installing : PyYAML-3.10-3.el6.x86_64                                  18/27
  Installing : audit-libs-python-2.2-2.el6.x86_64                        19/27
  Installing : policycoreutils-python-2.0.83-19.30.el6.x86_64            20/27
  Installing : python-backports-1.0-3.el6.x86_64                         21/27
  Installing : python-backports-ssl_match_hostname-3.4.0.2-1.el6.noarch  22/27
  Installing : python-urllib3-1.5-7.el6.noarch                           23/27
  Installing : python-requests-1.1.0-4.el6.noarch                        24/27
  Installing : python-jsonpointer-1.0-3.el6.noarch                       25/27
  Installing : python-jsonpatch-1.2-2.el6.noarch                         26/27
  Installing : cloud-init-0.7.4-2.el6.noarch                             27/27
  Verifying  : python-jsonpointer-1.0-3.el6.noarch                        1/27
  Verifying  : python-backports-1.0-3.el6.x86_64                          2/27
  Verifying  : libselinux-python-2.0.94-5.3.el6.x86_64                    3/27
  Verifying  : audit-libs-python-2.2-2.el6.x86_64                         4/27
  Verifying  : setools-libs-python-3.3.7-4.el6.x86_64                     5/27
  Verifying  : libyaml-0.1.6-1.el6.x86_64                                 6/27
```

```
  Verifying  : setools-libs-3.3.7-4.el6.x86_64                            7/27
  Verifying  : libcgroup-0.37-7.el6.x86_64                                8/27
  Verifying  : python-jsonpatch-1.2-2.el6.noarch                          9/27
  Verifying  : python-pygments-1.1.1-1.el6.noarch                        10/27
  Verifying  : python-prettytable-0.7.2-1.el6.noarch                     11/27
  Verifying  : python-backports-ssl_match_hostname-3.4.0.2-1.el6.noarch  12/27
  Verifying  : python-urllib3-1.5-7.el6.noarch                           13/27
  Verifying  : python-chardet-2.0.1-1.el6.noarch                         14/27
  Verifying  : python-requests-1.1.0-4.el6.noarch                        15/27
  Verifying  : python-cheetah-2.4.1-1.el6.x86_64                         16/27
  Verifying  : policycoreutils-python-2.0.83-19.30.el6.x86_64            17/27
  Verifying  : cloud-init-0.7.4-2.el6.noarch                             18/27
  Verifying  : python-markdown-2.0.1-3.1.el6.noarch                      19/27
  Verifying  : python-setuptools-0.6.10-3.el6.noarch                     20/27
  Verifying  : python-configobj-4.6.0-3.el6.noarch                       21/27
  Verifying  : python-six-1.6.1-1.el6.noarch                             22/27
  Verifying  : PyYAML-3.10-3.el6.x86_64                                  23/27
  Verifying  : python-boto-2.27.0-1.el6.noarch                           24/27
  Verifying  : python-ordereddict-1.1-2.el6.noarch                       25/27
  Verifying  : libsemanage-python-2.0.43-4.2.el6.x86_64                  26/27
  Verifying  : python-argparse-1.2.1-2.el6.noarch                        27/27

Installed:
  cloud-init.noarch 0:0.7.4-2.el6

Dependency Installed:
  PyYAML.x86_64 0:3.10-3.el6
  audit-libs-python.x86_64 0:2.2-2.el6
```

```
  libcgroup.x86_64 0:0.37-7.el6
  libselinux-python.x86_64 0:2.0.94-5.3.el6
  libsemanage-python.x86_64 0:2.0.43-4.2.el6
  libyaml.x86_64 0:0.1.6-1.el6
  policycoreutils-python.x86_64 0:2.0.83-19.30.el6
  python-argparse.noarch 0:1.2.1-2.el6
  python-backports.x86_64 0:1.0-3.el6
  python-backports-ssl_match_hostname.noarch 0:3.4.0.2-1.el6
  python-boto.noarch 0:2.27.0-1.el6
  python-chardet.noarch 0:2.0.1-1.el6
  python-cheetah.x86_64 0:2.4.1-1.el6
  python-configobj.noarch 0:4.6.0-3.el6
  python-jsonpatch.noarch 0:1.2-2.el6
  python-jsonpointer.noarch 0:1.0-3.el6
  python-markdown.noarch 0:2.0.1-3.1.el6
  python-ordereddict.noarch 0:1.1-2.el6
  python-prettytable.noarch 0:0.7.2-1.el6
  python-pygments.noarch 0:1.1.1-1.el6
  python-requests.noarch 0:1.1.0-4.el6
  python-setuptools.noarch 0:0.6.10-3.el6
  python-six.noarch 0:1.6.1-1.el6
  python-urllib3.noarch 0:1.5-7.el6
  setools-libs.x86_64 0:3.3.7-4.el6
  setools-libs-python.x86_64 0:3.3.7-4.el6

Complete!
Loaded plugins: security
Setting up Install Process
Resolving Dependencies
--> Running transaction check
---> Package cloud-utils.x86_64 0:0.27-10.el6 will be installed
--> Processing Dependency: qemu-img for package: cloud-utils-0.27-10.el6.x86_64
--> Processing Dependency: python-paramiko for package: cloud-utils-0.27-10.el6.x86_64
--> Processing Dependency: euca2ools for package: cloud-utils-0.27-10.el6.x86_64
--> Processing Dependency: cloud-utils-growpart for package: cloud-utils-0.27-10.el6.x86_64
--> Running transaction check
---> Package cloud-utils-growpart.x86_64 0:0.27-10.el6 will be installed
---> Package euca2ools.noarch 0:2.1.4-1.el6 will be installed
--> Processing Dependency: m2crypto for package: euca2ools-2.1.4-1.el6.noarch
---> Package python-paramiko.noarch 0:1.7.5-2.1.el6 will be installed
--> Processing Dependency: python-crypto >= 1.9 for package: python-paramiko-1.7.5-2.1.el6.noarch
---> Package qemu-img.x86_64 2:0.12.1.2-2.415.el6_5.10 will be installed
--> Processing Dependency: libusbredirparser.so.1()(64bit) for package: 2:qemu-img-0.12.1.2-2.415.el6_5.10.x86_64
--> Processing Dependency: libgfxdr.so.0()(64bit) for package: 2:qemu-img-0.12.1.2-2.415.el6_5.10.x86_64
--> Processing Dependency: libgfrpc.so.0()(64bit) for package: 2:qemu-img-0.12.1.2-2.415.el6_5.10.x86_64
--> Processing Dependency: libgfapi.so.0()(64bit) for package: 2:qemu-img-0.12.1.2-2.415.el6_5.10.x86_64
--> Running transaction check
---> Package glusterfs-api.x86_64 0:3.4.0.36rhs-1.el6 will be installed
```

```
---> Package glusterfs-libs.x86_64 0:3.4.0.36rhs-1.el6 will be installed
---> Package m2crypto.x86_64 0:0.20.2-9.el6 will be installed
---> Package python-crypto.x86_64 0:2.0.1-22.el6 will be installed
---> Package usbredir.x86_64 0:0.5.1-1.el6 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package               Arch    Version                       Repository     Size
================================================================================
Installing:
 cloud-utils           x86_64  0.27-10.el6                   epel           43 k
Installing for dependencies:
 cloud-utils-growpart  x86_64  0.27-10.el6                   epel           25 k
 euca2ools             noarch  2.1.4-1.el6                   epel          326 k
 glusterfs-api         x86_64  3.4.0.36rhs-1.el6              slf-security   44 k
 glusterfs-libs        x86_64  3.4.0.36rhs-1.el6              slf-security  225 k
 m2crypto              x86_64  0.20.2-9.el6                  slf           470 k
 python-crypto         x86_64  2.0.1-22.el6                  slf           157 k
 python-paramiko       noarch  1.7.5-2.1.el6                 slf           727 k
 qemu-img              x86_64  2:0.12.1.2-2.415.el6_5.10      slf-security  595 k
 usbredir              x86_64  0.5.1-1.el6                   slf            39 k

Transaction Summary
================================================================================
Install      10 Package(s)

Total download size: 2.6 M
Installed size: 13 M
Downloading Packages:
--------------------------------------------------------------------------------
Total                                            589 kB/s | 2.6 MB     00:04
Running rpm_check_debug
Running Transaction Test
Transaction Test Succeeded
Running Transaction

  Installing : glusterfs-libs-3.4.0.36rhs-1.el6.x86_64                      1/10

  Installing : glusterfs-api-3.4.0.36rhs-1.el6.x86_64                       2/10

  Installing : usbredir-0.5.1-1.el6.x86_64                                  3/10

  Installing : 2:qemu-img-0.12.1.2-2.415.el6_5.10.x86_64                    4/10

  Installing : python-crypto-2.0.1-22.el6.x86_64                            5/10
```

```
  Installing : python-paramiko-1.7.5-2.1.el6.noarch                        6/10

  Installing : cloud-utils-growpart-0.27-10.el6.x86_64                     7/10

  Installing : m2crypto-0.20.2-9.el6.x86_64                                8/10

  Installing : euca2ools-2.1.4-1.el6.noarch                                9/10

  Installing : cloud-utils-0.27-10.el6.x86_64                             10/10

  Verifying  : 2:qemu-img-0.12.1.2-2.415.el6_5.10.x86_64                   1/10

  Verifying  : euca2ools-2.1.4-1.el6.noarch                                2/10

  Verifying  : m2crypto-0.20.2-9.el6.x86_64                                3/10

  Verifying  : cloud-utils-growpart-0.27-10.el6.x86_64                     4/10

  Verifying  : glusterfs-libs-3.4.0.36rhs-1.el6.x86_64                     5/10

  Verifying  : python-crypto-2.0.1-22.el6.x86_64                           6/10

  Verifying  : python-paramiko-1.7.5-2.1.el6.noarch                        7/10

  Verifying  : cloud-utils-0.27-10.el6.x86_64                              8/10

  Verifying  : usbredir-0.5.1-1.el6.x86_64                                 9/10

  Verifying  : glusterfs-api-3.4.0.36rhs-1.el6.x86_64                     10/10

Installed:
  cloud-utils.x86_64 0:0.27-10.el6

Dependency Installed:
  cloud-utils-growpart.x86_64 0:0.27-10.el6
  euca2ools.noarch 0:2.1.4-1.el6
  glusterfs-api.x86_64 0:3.4.0.36rhs-1.el6
  glusterfs-libs.x86_64 0:3.4.0.36rhs-1.el6
  m2crypto.x86_64 0:0.20.2-9.el6
  python-crypto.x86_64 0:2.0.1-22.el6
  python-paramiko.noarch 0:1.7.5-2.1.el6
  qemu-img.x86_64 2:0.12.1.2-2.415.el6_5.10
  usbredir.x86_64 0:0.5.1-1.el6

Complete!
Loaded plugins: security
Setting up Install Process
Examining /var/tmp/yum-root-bFavOG/python-backports-1.0-4.el6.x86_64.rpm: python-backports-1.0-4.el6.x86_64
```

```
Marking /var/tmp/yum-root-bFavOG/python-backports-1.0-4.el6.x86_64.rpm as an update to python-backports-1.0-
3.el6.x86_64
Resolving Dependencies
--> Running transaction check
---> Package python-backports.x86_64 0:1.0-3.el6 will be updated
---> Package python-backports.x86_64 0:1.0-4.el6 will be an update
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package           Arch    Version    Repository                            Size
================================================================================
Updating:
 python-backports  x86_64  1.0-4.el6  /python-backports-1.0-4.el6.x86_64  318

Transaction Summary
================================================================================
Upgrade       1 Package(s)

Total size: 318
Downloading Packages:
Running rpm_check_debug
Running Transaction Test
Transaction Test Succeeded
Running Transaction

  Updating   : python-backports-1.0-4.el6.x86_64                            1/2

  Cleanup    : python-backports-1.0-3.el6.x86_64                            2/2

  Verifying  : python-backports-1.0-4.el6.x86_64                            1/2

  Verifying  : python-backports-1.0-3.el6.x86_64                            2/2

Updated:
  python-backports.x86_64 0:1.0-4.el6

Complete!
Aug 02 15:22:20 INFO     Stop: Completed Converting the worker Fermicloud VM image. Function convert_image.
Aug 02 15:22:20 INFO     Start: Importing the worker Fermicloud VM image. Function import_image.
Loaded plugins: refresh-packagekit, security
Setting up Install Process
Package 1:java-1.6.0-openjdk-1.6.0.0-6.1.13.4.el6_5.x86_64 already installed and latest version
Nothing to do
/usr/lib/jvm/jre-1.6.0-openjdk.x86_64
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
```

```
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0
  0 14.2M    0 13021    0     0   121k      0  0:01:59 --:--:--  0:01:59  153k
 96 14.2M   96 13.7M    0     0  12.5M      0  0:00:01  0:00:01 --:--:-- 12.8M
100 14.2M  100 14.2M    0     0  12.7M      0  0:00:01  0:00:01 --:--:-- 13.0M
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed

  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0
100  152k  100  152k    0     0   718k      0 --:--:-- --:--:-- --:--:--  807k
Loaded plugins: refresh-packagekit, security
Setting up Install Process
Package 1:tcl-devel-8.5.7-6.el6.x86_64 already installed and latest version
Package 1:tk-devel-8.5.7-5.el6.x86_64 already installed and latest version
Package expect-devel-5.44.1.15-2.el6.x86_64 already installed and latest version
Package expectk-5.44.1.15-2.el6.x86_64 already installed and latest version
Nothing to do
AWS CLI Tools Setup Completed.
ATTACHMENT   vol-5f6ff95e i-94fe889f   /dev/sdf     attaching    2014-08-02T21:04:24+0000
TAG   instance     i-32f68039   Name  GCSO_SL6_HVM_20140802152232
TAG   instance     i-32f68039   User  kirkshal
TAG   volume vol-2b63f52a Name  GCSO_SL6_HVM_20140802152232
TAG   volume vol-2b63f52a User  kirkshal
TAG   snapshot     snap-69d4759e      Name  GCSO_SL6_HVM_20140802152232
TAG   snapshot     snap-69d4759e      User  kirkshal
TAG   image  ami-21d2a911 Name  GCSO_SL6_HVM_20140802152232
TAG   image  ami-21d2a911 User  kirkshal
TAG   instance     i-94fe889f   Name  GCSO_SL6_HVM_20140802152232
TAG   instance     i-94fe889f   User  kirkshal
TAG   volume vol-5c6ff95d Name  GCSO_SL6_HVM_20140802152232
TAG   volume vol-5c6ff95d User  kirkshal
TAG   volume vol-5f6ff95e Name  GCSO_SL6_HVM_20140802152232
TAG   volume vol-5f6ff95e User  kirkshal
INSTANCE     i-94fe889f   running      stopping
INSTANCE     i-32f68039   stopped      terminated
VOLUME vol-2b63f52a
Requirement already up-to-date: awscli in /usr/local/lib/python2.7/site-packages
Requirement already up-to-date: botocore>=0.59.0,<0.60.0 in /usr/local/lib/python2.7/site-packages (from awscli)
Requirement already up-to-date: bcdoc>=0.12.0,<0.13.0 in /usr/local/lib/python2.7/site-packages (from awscli)
Requirement already up-to-date: six>=1.1.0 in /usr/local/lib/python2.7/site-packages (from awscli)
Requirement already up-to-date: colorama==0.2.5 in /usr/local/lib/python2.7/site-packages (from awscli)
Requirement already up-to-date: docutils>=0.10 in /usr/local/lib/python2.7/site-packages (from awscli)
Requirement already up-to-date: rsa==3.1.2 in /usr/local/lib/python2.7/site-packages (from awscli)
Requirement already up-to-date: jmespath==0.4.1 in /usr/local/lib/python2.7/site-packages (from
botocore>=0.59.0,<0.60.0->awscli)
Requirement already up-to-date: python-dateutil>=2.1 in /usr/local/lib/python2.7/site-packages (from
botocore>=0.59.0,<0.60.0->awscli)
Requirement already up-to-date: pyasn1>=0.1.3 in /usr/local/lib/python2.7/site-packages (from rsa==3.1.2->awscli)
```

```
Cleaning up...
delete: s3://fcloudimport20140802152232/0b585d2b-3609-4ac8-b3f4-42a754be0589/gcso_sl6.raw.part0
Completed 1 part(s) with ... file(s) remaining
delete: s3://fcloudimport20140802152232/0b585d2b-3609-4ac8-b3f4-42a754be0589/gcso_sl6.raw.part1006
Completed 2 part(s) with ... file(s) remaining
delete: s3://fcloudimport20140802152232/0b585d2b-3609-4ac8-b3f4-42a754be0589/gcso_sl6.raw.part1002
Completed 3 part(s) with ... file(s) remaining
delete: s3://fcloudimport20140802152232/0b585d2b-3609-4ac8-b3f4-42a754be0589/gcso_sl6.raw.part1008
Completed 4 part(s) with ... file(s) remaining

[SNIP]

delete: s3://fcloudimport20140802152232/0b585d2b-3609-4ac8-b3f4-42a754be0589/gcso_sl6.raw.part998
Completed 1230 of 1231 part(s) with 1 file(s) remaining
delete: s3://fcloudimport20140802152232/0b585d2b-3609-4ac8-b3f4-42a754be0589/gcso_sl6.rawmanifest.xml
remove_bucket: s3://fcloudimport20140802152232/
AWS HVM VM Import Completed.
Aug 02 16:06:34 INFO     Stop: Completed Importing the worker Fermicloud VM image. Function import_image.
Aug 02 16:06:34 INFO     End script run

-----------
-----------
-----------


File: puppetrepo-awsexport/files/opt/gcso/awsexport/docs/aws_image_convert_pv.log
Aug 02 16:14:59 INFO     Begin script run
Aug 02 16:14:59 INFO     Start: Copying the selected Fermicloud VM image. Function copy_to_image_location.
Aug 02 16:15:54 INFO     Stop: Completed Copying the selected Fermicloud VM image. Function
copy_to_image_location.
Aug 02 16:15:54 INFO     Start: Resizing the worker Fermicloud VM image. Function resize_image.
*stdin*:2: libguestfs: error: e2fsck_f: /dev/sda1: 94023/786432 files (0.3% non-contiguous), 550839/3145728
blocks
Formatting 'gcso_sl6.raw', fmt=raw size=3224371200
Examining gcso_sl6.qcow2tmp ...
**********

Summary of changes:

/dev/sda1: This partition will be resized from 12.0G to 3.0G.  The
    filesystem ext3 on /dev/sda1 will be expanded using the 'resize2fs'
    method.

There is a surplus of 864.0K.  The surplus space is not large enough
for an extra partition to be created and so it will just be ignored.

**********
Setting up initial partition table on gcso_sl6.raw ...
```

```
Copying /dev/sda1 ...
Expanding /dev/sda1 using the 'resize2fs' method ...

Resize operation completed with no errors.  Before deleting the old
disk, carefully check that the resized disk boots and works correctly.
image: gcso_sl6.raw
file format: raw
virtual size: 3.0G (3224371200 bytes)
disk size: 3.0G
Aug 02 16:35:19 INFO     Stop: Completed Resizing the worker Fermicloud VM image. Function resize_image.
Aug 02 16:35:19 INFO     Start: Converting the worker Fermicloud VM image. Function convert_image.
warning: /var/tmp/rpm-tmp.6lH0MT: Header V3 RSA/SHA256 Signature, key ID 0608b895: NOKEY
Retrieving http://download.fedoraproject.org/pub/epel/6/x86_64/epel-release-6-8.noarch.rpm
Preparing...                ##################################################
epel-release                ##################################################
Loaded plugins: security
Setting up Install Process
Resolving Dependencies
--> Running transaction check
---> Package cloud-init.noarch 0:0.7.4-2.el6 will be installed
--> Processing Dependency: python-boto >= 2.6.0 for package: cloud-init-0.7.4-2.el6.noarch
--> Processing Dependency: python-requests for package: cloud-init-0.7.4-2.el6.noarch
--> Processing Dependency: python-prettytable for package: cloud-init-0.7.4-2.el6.noarch
--> Processing Dependency: python-jsonpatch for package: cloud-init-0.7.4-2.el6.noarch
--> Processing Dependency: python-configobj for package: cloud-init-0.7.4-2.el6.noarch
--> Processing Dependency: python-cheetah for package: cloud-init-0.7.4-2.el6.noarch
--> Processing Dependency: python-argparse for package: cloud-init-0.7.4-2.el6.noarch
--> Processing Dependency: policycoreutils-python for package: cloud-init-0.7.4-2.el6.noarch
--> Processing Dependency: libselinux-python for package: cloud-init-0.7.4-2.el6.noarch
--> Processing Dependency: PyYAML for package: cloud-init-0.7.4-2.el6.noarch
--> Running transaction check
---> Package PyYAML.x86_64 0:3.10-3.el6 will be installed
--> Processing Dependency: libyaml-0.so.2()(64bit) for package: PyYAML-3.10-3.el6.x86_64
---> Package libselinux-python.x86_64 0:2.0.94-5.3.el6 will be installed
---> Package policycoreutils-python.x86_64 0:2.0.83-19.30.el6 will be installed
--> Processing Dependency: libsemanage-python >= 2.0.43-4 for package: policycoreutils-python-2.0.83-
19.30.el6.x86_64
--> Processing Dependency: audit-libs-python >= 1.4.2-1 for package: policycoreutils-python-2.0.83-
19.30.el6.x86_64
--> Processing Dependency: setools-libs-python for package: policycoreutils-python-2.0.83-19.30.el6.x86_64
--> Processing Dependency: libcgroup for package: policycoreutils-python-2.0.83-19.30.el6.x86_64
---> Package python-argparse.noarch 0:1.2.1-2.el6 will be installed
---> Package python-boto.noarch 0:2.27.0-1.el6 will be installed
---> Package python-cheetah.x86_64 0:2.4.1-1.el6 will be installed
--> Processing Dependency: python-pygments for package: python-cheetah-2.4.1-1.el6.x86_64
--> Processing Dependency: python-markdown for package: python-cheetah-2.4.1-1.el6.x86_64
---> Package python-configobj.noarch 0:4.6.0-3.el6 will be installed
---> Package python-jsonpatch.noarch 0:1.2-2.el6 will be installed
```

```
--> Processing Dependency: python-jsonpointer for package: python-jsonpatch-1.2-2.el6.noarch
---> Package python-prettytable.noarch 0:0.7.2-1.el6 will be installed
---> Package python-requests.noarch 0:1.1.0-4.el6 will be installed
--> Processing Dependency: python-urllib3 for package: python-requests-1.1.0-4.el6.noarch
--> Processing Dependency: python-ordereddict for package: python-requests-1.1.0-4.el6.noarch
--> Processing Dependency: python-chardet for package: python-requests-1.1.0-4.el6.noarch
--> Running transaction check
---> Package audit-libs-python.x86_64 0:2.2-2.el6 will be installed
---> Package libcgroup.x86_64 0:0.37-7.el6 will be installed
---> Package libsemanage-python.x86_64 0:2.0.43-4.2.el6 will be installed
---> Package libyaml.x86_64 0:0.1.6-1.el6 will be installed
---> Package python-chardet.noarch 0:2.0.1-1.el6 will be installed
---> Package python-jsonpointer.noarch 0:1.0-3.el6 will be installed
---> Package python-markdown.noarch 0:2.0.1-3.1.el6 will be installed
---> Package python-ordereddict.noarch 0:1.1-2.el6 will be installed
---> Package python-pygments.noarch 0:1.1.1-1.el6 will be installed
--> Processing Dependency: python-setuptools for package: python-pygments-1.1.1-1.el6.noarch
---> Package python-urllib3.noarch 0:1.5-7.el6 will be installed
--> Processing Dependency: python-six for package: python-urllib3-1.5-7.el6.noarch
--> Processing Dependency: python-backports-ssl_match_hostname for package: python-urllib3-1.5-7.el6.noarch
---> Package setools-libs-python.x86_64 0:3.3.7-4.el6 will be installed
--> Processing Dependency: setools-libs = 3.3.7-4.el6 for package: setools-libs-python-3.3.7-4.el6.x86_64
--> Processing Dependency: libqpol.so.1(VERS_1.3)(64bit) for package: setools-libs-python-3.3.7-4.el6.x86_64
--> Processing Dependency: libpoldiff.so.1(VERS_1.3)(64bit) for package: setools-libs-python-3.3.7-4.el6.x86_64
--> Processing Dependency: libapol.so.4(VERS_4.1)(64bit) for package: setools-libs-python-3.3.7-4.el6.x86_64
--> Processing Dependency: libqpol.so.1(VERS_1.2)(64bit) for package: setools-libs-python-3.3.7-4.el6.x86_64
--> Processing Dependency: libqpol.so.1(VERS_1.4)(64bit) for package: setools-libs-python-3.3.7-4.el6.x86_64
--> Processing Dependency: libseaudit.so.4(VERS_4.2)(64bit) for package: setools-libs-python-3.3.7-4.el6.x86_64
--> Processing Dependency: libpoldiff.so.1(VERS_1.2)(64bit) for package: setools-libs-python-3.3.7-4.el6.x86_64
--> Processing Dependency: libsefs.so.4(VERS_4.0)(64bit) for package: setools-libs-python-3.3.7-4.el6.x86_64
--> Processing Dependency: libapol.so.4(VERS_4.0)(64bit) for package: setools-libs-python-3.3.7-4.el6.x86_64
--> Processing Dependency: libseaudit.so.4(VERS_4.1)(64bit) for package: setools-libs-python-3.3.7-4.el6.x86_64
--> Processing Dependency: libseaudit.so.4()(64bit) for package: setools-libs-python-3.3.7-4.el6.x86_64
--> Processing Dependency: libqpol.so.1()(64bit) for package: setools-libs-python-3.3.7-4.el6.x86_64
--> Processing Dependency: libapol.so.4()(64bit) for package: setools-libs-python-3.3.7-4.el6.x86_64
--> Processing Dependency: libpoldiff.so.1()(64bit) for package: setools-libs-python-3.3.7-4.el6.x86_64
--> Processing Dependency: libsefs.so.4()(64bit) for package: setools-libs-python-3.3.7-4.el6.x86_64
--> Running transaction check
---> Package python-backports-ssl_match_hostname.noarch 0:3.4.0.2-1.el6 will be installed
--> Processing Dependency: python-backports for package: python-backports-ssl_match_hostname-3.4.0.2-1.el6.noarch
---> Package python-setuptools.noarch 0:0.6.10-3.el6 will be installed
---> Package python-six.noarch 0:1.6.1-1.el6 will be installed
---> Package setools-libs.x86_64 0:3.3.7-4.el6 will be installed
--> Running transaction check
---> Package python-backports.x86_64 0:1.0-3.el6 will be installed
--> Finished Dependency Resolution

Dependencies Resolved
```

```
================================================================================
 Package                                Arch     Version            Repository
                                                                           Size
================================================================================
Installing:
 cloud-init                             noarch   0.7.4-2.el6        epel   487 k
Installing for dependencies:
 PyYAML                                 x86_64   3.10-3.el6         epel   157 k
 audit-libs-python                      x86_64   2.2-2.el6          slf     58 k
 libcgroup                              x86_64   0.37-7.el6         slf    110 k
 libselinux-python                      x86_64   2.0.94-5.3.el6     slf    201 k
 libsemanage-python                     x86_64   2.0.43-4.2.el6     slf     80 k
 libyaml                                x86_64   0.1.6-1.el6        epel    52 k
 policycoreutils-python                 x86_64   2.0.83-19.30.el6   slf    341 k
 python-argparse                        noarch   1.2.1-2.el6        epel    48 k
 python-backports                       x86_64   1.0-3.el6          epel   5.3 k
 python-backports-ssl_match_hostname    noarch   3.4.0.2-1.el6      epel    12 k
 python-boto                            noarch   2.27.0-1.el6       epel   1.6 M
 python-chardet                         noarch   2.0.1-1.el6        epel   225 k
 python-cheetah                         x86_64   2.4.1-1.el6        slf    364 k
 python-configobj                       noarch   4.6.0-3.el6        slf    181 k
 python-jsonpatch                       noarch   1.2-2.el6          epel    14 k
 python-jsonpointer                     noarch   1.0-3.el6          epel   9.2 k
 python-markdown                        noarch   2.0.1-3.1.el6      slf    117 k
 python-ordereddict                     noarch   1.1-2.el6          epel   7.6 k
 python-prettytable                     noarch   0.7.2-1.el6        epel    37 k
 python-pygments                        noarch   1.1.1-1.el6        slf    561 k
 python-requests                        noarch   1.1.0-4.el6        epel    71 k
 python-setuptools                      noarch   0.6.10-3.el6       slf    335 k
 python-six                             noarch   1.6.1-1.el6        epel    25 k
 python-urllib3                         noarch   1.5-7.el6          epel    41 k
 setools-libs                           x86_64   3.3.7-4.el6        slf    399 k
 setools-libs-python                    x86_64   3.3.7-4.el6        slf    221 k

Transaction Summary
================================================================================
Install      27 Package(s)

Total download size: 5.6 M
Installed size: 25 M
Downloading Packages:
--------------------------------------------------------------------------------
Total                                           532 kB/s | 5.6 MB     00:10
Retrieving key from file:///etc/pki/rpm-gpg/RPM-GPG-KEY-EPEL-6
warning: rpmts_HdrFromFdno: Header V3 RSA/SHA256 Signature, key ID 0608b895: NOKEY
Importing GPG key 0x0608B895:
 Userid : EPEL (6) <epel@fedoraproject.org>
```

```
 Package: slf-release-6.4-1.x86_64 (@anaconda-ScientificLinuxFermi-201304091346.x86_64/6)
 From   : /etc/pki/rpm-gpg/RPM-GPG-KEY-EPEL-6
Running rpm_check_debug
Running Transaction Test
Transaction Test Succeeded
Running Transaction
Warning: RPMDB altered outside of yum.

  Installing : python-ordereddict-1.1-2.el6.noarch                          1/27

  Installing : libselinux-python-2.0.94-5.3.el6.x86_64                      2/27

  Installing : python-argparse-1.2.1-2.el6.noarch                           3/27

  Installing : libsemanage-python-2.0.43-4.2.el6.x86_64                     4/27

  Installing : python-boto-2.27.0-1.el6.noarch                              5/27

  Installing : python-six-1.6.1-1.el6.noarch                                6/27

  Installing : python-configobj-4.6.0-3.el6.noarch                          7/27

  Installing : python-setuptools-0.6.10-3.el6.noarch                        8/27

  Installing : python-pygments-1.1.1-1.el6.noarch                           9/27

  Installing : python-markdown-2.0.1-3.1.el6.noarch                        10/27

  Installing : python-cheetah-2.4.1-1.el6.x86_64                           11/27

  Installing : python-chardet-2.0.1-1.el6.noarch                           12/27

  Installing : python-prettytable-0.7.2-1.el6.noarch                       13/27

  Installing : libcgroup-0.37-7.el6.x86_64                                 14/27

  Installing : setools-libs-3.3.7-4.el6.x86_64                             15/27

  Installing : setools-libs-python-3.3.7-4.el6.x86_64                      16/27

  Installing : libyaml-0.1.6-1.el6.x86_64                                  17/27

  Installing : PyYAML-3.10-3.el6.x86_64                                    18/27

  Installing : audit-libs-python-2.2-2.el6.x86_64                          19/27

  Installing : policycoreutils-python-2.0.83-19.30.el6.x86_64              20/27
```

```
  Installing : python-backports-1.0-3.el6.x86_64                          21/27
  Installing : python-backports-ssl_match_hostname-3.4.0.2-1.el6.noarch   22/27
  Installing : python-urllib3-1.5-7.el6.noarch                            23/27
  Installing : python-requests-1.1.0-4.el6.noarch                         24/27
  Installing : python-jsonpointer-1.0-3.el6.noarch                        25/27
  Installing : python-jsonpatch-1.2-2.el6.noarch                          26/27
  Installing : cloud-init-0.7.4-2.el6.noarch                              27/27
  Verifying  : python-jsonpointer-1.0-3.el6.noarch                         1/27
  Verifying  : python-backports-1.0-3.el6.x86_64                           2/27
  Verifying  : libselinux-python-2.0.94-5.3.el6.x86_64                     3/27
  Verifying  : audit-libs-python-2.2-2.el6.x86_64                          4/27
  Verifying  : setools-libs-python-3.3.7-4.el6.x86_64                      5/27
  Verifying  : libyaml-0.1.6-1.el6.x86_64                                  6/27
  Verifying  : setools-libs-3.3.7-4.el6.x86_64                             7/27
  Verifying  : libcgroup-0.37-7.el6.x86_64                                 8/27
  Verifying  : python-jsonpatch-1.2-2.el6.noarch                           9/27
  Verifying  : python-pygments-1.1.1-1.el6.noarch                         10/27
  Verifying  : python-prettytable-0.7.2-1.el6.noarch                      11/27
  Verifying  : python-backports-ssl_match_hostname-3.4.0.2-1.el6.noarch   12/27
  Verifying  : python-urllib3-1.5-7.el6.noarch                            13/27
  Verifying  : python-chardet-2.0.1-1.el6.noarch                          14/27
  Verifying  : python-requests-1.1.0-4.el6.noarch                         15/27
  Verifying  : python-cheetah-2.4.1-1.el6.x86_64                          16/27
  Verifying  : policycoreutils-python-2.0.83-19.30.el6.x86_64             17/27
```

```
  Verifying  : cloud-init-0.7.4-2.el6.noarch                              18/27
  Verifying  : python-markdown-2.0.1-3.1.el6.noarch                       19/27
  Verifying  : python-setuptools-0.6.10-3.el6.noarch                      20/27
  Verifying  : python-configobj-4.6.0-3.el6.noarch                        21/27
  Verifying  : python-six-1.6.1-1.el6.noarch                              22/27
  Verifying  : PyYAML-3.10-3.el6.x86_64                                   23/27
  Verifying  : python-boto-2.27.0-1.el6.noarch                            24/27
  Verifying  : python-ordereddict-1.1-2.el6.noarch                        25/27
  Verifying  : libsemanage-python-2.0.43-4.2.el6.x86_64                   26/27
  Verifying  : python-argparse-1.2.1-2.el6.noarch                         27/27

Installed:
  cloud-init.noarch 0:0.7.4-2.el6

Dependency Installed:
  PyYAML.x86_64 0:3.10-3.el6
  audit-libs-python.x86_64 0:2.2-2.el6
  libcgroup.x86_64 0:0.37-7.el6
  libselinux-python.x86_64 0:2.0.94-5.3.el6
  libsemanage-python.x86_64 0:2.0.43-4.2.el6
  libyaml.x86_64 0:0.1.6-1.el6
  policycoreutils-python.x86_64 0:2.0.83-19.30.el6
  python-argparse.noarch 0:1.2.1-2.el6
  python-backports.x86_64 0:1.0-3.el6
  python-backports-ssl_match_hostname.noarch 0:3.4.0.2-1.el6
  python-boto.noarch 0:2.27.0-1.el6
  python-chardet.noarch 0:2.0.1-1.el6
  python-cheetah.x86_64 0:2.4.1-1.el6
  python-configobj.noarch 0:4.6.0-3.el6
  python-jsonpatch.noarch 0:1.2-2.el6
  python-jsonpointer.noarch 0:1.0-3.el6
  python-markdown.noarch 0:2.0.1-3.1.el6
  python-ordereddict.noarch 0:1.1-2.el6
  python-prettytable.noarch 0:0.7.2-1.el6
  python-pygments.noarch 0:1.1.1-1.el6
  python-requests.noarch 0:1.1.0-4.el6
  python-setuptools.noarch 0:0.6.10-3.el6
  python-six.noarch 0:1.6.1-1.el6
  python-urllib3.noarch 0:1.5-7.el6
```

```
  setools-libs.x86_64 0:3.3.7-4.el6
  setools-libs-python.x86_64 0:3.3.7-4.el6

Complete!
Loaded plugins: security
Setting up Install Process
Resolving Dependencies
--> Running transaction check
---> Package cloud-utils.x86_64 0:0.27-10.el6 will be installed
--> Processing Dependency: qemu-img for package: cloud-utils-0.27-10.el6.x86_64
--> Processing Dependency: python-paramiko for package: cloud-utils-0.27-10.el6.x86_64
--> Processing Dependency: euca2ools for package: cloud-utils-0.27-10.el6.x86_64
--> Processing Dependency: cloud-utils-growpart for package: cloud-utils-0.27-10.el6.x86_64
--> Running transaction check
---> Package cloud-utils-growpart.x86_64 0:0.27-10.el6 will be installed
---> Package euca2ools.noarch 0:2.1.4-1.el6 will be installed
--> Processing Dependency: m2crypto for package: euca2ools-2.1.4-1.el6.noarch
---> Package python-paramiko.noarch 0:1.7.5-2.1.el6 will be installed
--> Processing Dependency: python-crypto >= 1.9 for package: python-paramiko-1.7.5-2.1.el6.noarch
---> Package qemu-img.x86_64 2:0.12.1.2-2.415.el6_5.10 will be installed
--> Processing Dependency: libusbredirparser.so.1()(64bit) for package: 2:qemu-img-0.12.1.2-2.415.el6_5.10.x86_64
--> Processing Dependency: libgfxdr.so.0()(64bit) for package: 2:qemu-img-0.12.1.2-2.415.el6_5.10.x86_64
--> Processing Dependency: libgfrpc.so.0()(64bit) for package: 2:qemu-img-0.12.1.2-2.415.el6_5.10.x86_64
--> Processing Dependency: libgfapi.so.0()(64bit) for package: 2:qemu-img-0.12.1.2-2.415.el6_5.10.x86_64
--> Running transaction check
---> Package glusterfs-api.x86_64 0:3.4.0.36rhs-1.el6 will be installed
---> Package glusterfs-libs.x86_64 0:3.4.0.36rhs-1.el6 will be installed
---> Package m2crypto.x86_64 0:0.20.2-9.el6 will be installed
---> Package python-crypto.x86_64 0:2.0.1-22.el6 will be installed
---> Package usbredir.x86_64 0:0.5.1-1.el6 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package               Arch    Version                       Repository     Size
================================================================================
Installing:
 cloud-utils           x86_64  0.27-10.el6                   epel           43 k
Installing for dependencies:
 cloud-utils-growpart  x86_64  0.27-10.el6                   epel           25 k
 euca2ools             noarch  2.1.4-1.el6                   epel          326 k
 glusterfs-api         x86_64  3.4.0.36rhs-1.el6              slf-security   44 k
 glusterfs-libs        x86_64  3.4.0.36rhs-1.el6              slf-security  225 k
 m2crypto              x86_64  0.20.2-9.el6                  slf           470 k
 python-crypto         x86_64  2.0.1-22.el6                  slf           157 k
 python-paramiko       noarch  1.7.5-2.1.el6                 slf           727 k
 qemu-img              x86_64  2:0.12.1.2-2.415.el6_5.10     slf-security  595 k
```

```
 usbredir              x86_64  0.5.1-1.el6                   slf            39 k

Transaction Summary
================================================================================
Install      10 Package(s)

Total download size: 2.6 M
Installed size: 13 M
Downloading Packages:
--------------------------------------------------------------------------------
Total                                              570 kB/s | 2.6 MB     00:04
Running rpm_check_debug
Running Transaction Test
Transaction Test Succeeded
Running Transaction

  Installing : glusterfs-libs-3.4.0.36rhs-1.el6.x86_64                      1/10

  Installing : glusterfs-api-3.4.0.36rhs-1.el6.x86_64                       2/10

  Installing : usbredir-0.5.1-1.el6.x86_64                                  3/10

  Installing : 2:qemu-img-0.12.1.2-2.415.el6_5.10.x86_64                    4/10

  Installing : python-crypto-2.0.1-22.el6.x86_64                            5/10

  Installing : python-paramiko-1.7.5-2.1.el6.noarch                         6/10

  Installing : cloud-utils-growpart-0.27-10.el6.x86_64                      7/10

  Installing : m2crypto-0.20.2-9.el6.x86_64                                 8/10

  Installing : euca2ools-2.1.4-1.el6.noarch                                 9/10

  Installing : cloud-utils-0.27-10.el6.x86_64                              10/10

  Verifying  : 2:qemu-img-0.12.1.2-2.415.el6_5.10.x86_64                    1/10

  Verifying  : euca2ools-2.1.4-1.el6.noarch                                 2/10

  Verifying  : m2crypto-0.20.2-9.el6.x86_64                                 3/10

  Verifying  : cloud-utils-growpart-0.27-10.el6.x86_64                      4/10

  Verifying  : glusterfs-libs-3.4.0.36rhs-1.el6.x86_64                      5/10

  Verifying  : python-crypto-2.0.1-22.el6.x86_64                            6/10
```

```
  Verifying  : python-paramiko-1.7.5-2.1.el6.noarch                        7/10

  Verifying  : cloud-utils-0.27-10.el6.x86_64                              8/10

  Verifying  : usbredir-0.5.1-1.el6.x86_64                                 9/10

  Verifying  : glusterfs-api-3.4.0.36rhs-1.el6.x86_64                     10/10

Installed:
  cloud-utils.x86_64 0:0.27-10.el6

Dependency Installed:
  cloud-utils-growpart.x86_64 0:0.27-10.el6
  euca2ools.noarch 0:2.1.4-1.el6
  glusterfs-api.x86_64 0:3.4.0.36rhs-1.el6
  glusterfs-libs.x86_64 0:3.4.0.36rhs-1.el6
  m2crypto.x86_64 0:0.20.2-9.el6
  python-crypto.x86_64 0:2.0.1-22.el6
  python-paramiko.noarch 0:1.7.5-2.1.el6
  qemu-img.x86_64 2:0.12.1.2-2.415.el6_5.10
  usbredir.x86_64 0:0.5.1-1.el6

Complete!
Loaded plugins: security
Setting up Install Process
Examining /var/tmp/yum-root-DElfP9/python-backports-1.0-4.el6.x86_64.rpm: python-backports-1.0-4.el6.x86_64
Marking /var/tmp/yum-root-DElfP9/python-backports-1.0-4.el6.x86_64.rpm as an update to python-backports-1.0-
3.el6.x86_64
Resolving Dependencies
--> Running transaction check
---> Package python-backports.x86_64 0:1.0-3.el6 will be updated
---> Package python-backports.x86_64 0:1.0-4.el6 will be an update
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package           Arch    Version    Repository                            Size
================================================================================
Updating:
 python-backports  x86_64  1.0-4.el6  /python-backports-1.0-4.el6.x86_64  318

Transaction Summary
================================================================================
Upgrade       1 Package(s)

Total size: 318
Downloading Packages:
```

```
Running rpm_check_debug
Running Transaction Test
Transaction Test Succeeded
Running Transaction

  Updating   : python-backports-1.0-4.el6.x86_64                           1/2

  Cleanup    : python-backports-1.0-3.el6.x86_64                           2/2

  Verifying  : python-backports-1.0-4.el6.x86_64                           1/2

  Verifying  : python-backports-1.0-3.el6.x86_64                           2/2

Updated:
  python-backports.x86_64 0:1.0-4.el6

Complete!
Aug 02 16:55:29 INFO     Stop: Completed Converting the worker Fermicloud VM image. Function convert_image.
Aug 02 16:55:29 INFO     Start: Importing the worker Fermicloud VM image. Function import_image.
Loaded plugins: refresh-packagekit, security
Setting up Install Process
Package 1:java-1.6.0-openjdk-1.6.0.0-6.1.13.4.el6_5.x86_64 already installed and latest version
Nothing to do
/usr/lib/jvm/jre-1.6.0-openjdk.x86_64
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed

  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0
  4 14.2M    4  708k    0     0   890k      0  0:00:16 --:--:--  0:00:16  915k
100 14.2M  100 14.2M    0     0  8873k      0  0:00:01  0:00:01 --:--:-- 8997k
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed

  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0
100  152k  100  152k    0     0   575k      0 --:--:-- --:--:-- --:--:--Loaded plugins: refresh-packagekit,
security
Setting up Install Process
  630k
Package 1:tcl-devel-8.5.7-6.el6.x86_64 already installed and latest version
Package 1:tk-devel-8.5.7-5.el6.x86_64 already installed and latest version
Package expect-devel-5.44.1.15-2.el6.x86_64 already installed and latest version
Package expectk-5.44.1.15-2.el6.x86_64 already installed and latest version
Nothing to do
AWS CLI Tools Setup Completed.
INSTANCE    i-94fe889f   stopped      pending
ATTACHMENT   vol-af1680ae i-94fe889f   /dev/sdg     attaching    2014-08-02T21:56:45+0000
spawn ssh -i khs-fermi.pem root@54.187.12.76
The authenticity of host '54.187.12.76 (54.187.12.76)' can't be established.
```

```
RSA key fingerprint is ab:90:b1:c4:ef:65:f4:5f:c9:8f:db:df:91:f6:6a:9c.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '54.187.12.76' (RSA) to the list of known hosts.

Last login: Thu Jul 17 16:43:33 2014 from 131.225.170.210

                              NOTICE TO USERS

       This  is a Federal computer (and/or it is directly connected to a
       Fermilab local network system) that is the property of the United
       States Government.  It is for authorized use only.  Users (autho-
       rized or unauthorized) have no explicit or  implicit  expectation
       of privacy.

       Any  or  all uses of this system and all files on this system may
       be intercepted, monitored, recorded,  copied, audited, inspected,
       and  disclosed  to authorized site, Department of Energy  and law
       enforcement personnel, as  well as authorized officials of  other
       agencies,  both  domestic and foreign.  By using this system, the
       user consents to such interception, monitoring, recording,  copy-
       ing,  auditing,  inspection,  and disclosure at the discretion of
       authorized site or Department of Energy personnel.

       Unauthorized or improper use of this system may result in  admin-
       istrative  disciplinary  action and civil and criminal penalties.
       By continuing to use this system you indicate your  awareness  of
       and  consent to these terms and conditions of use.  LOG OFF IMME-
       DIATELY if you do not agree to  the  conditions  stated  in  this
       warning.

       Fermilab  policy  and  rules for computing, including appropriate
       use, may be found at http://www.fnal.gov/cd/main/cpolicy.html
/usr/bin/xauth:  creating new authority file /root/.Xauthority
[root@ip-172-31-38-72 ~]# mkfs -t ext4 /dev/xvdf
mke2fs 1.41.12 (17-May-2010)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
Stride=0 blocks, Stripe width=0 blocks
1310720 inodes, 5242880 blocks
262144 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=4294967296
160 block groups
32768 blocks per group, 32768 fragments per group
8192 inodes per group
Superblock backups stored on blocks:
```

```
       32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632, 2654208,
       4096000

Writing inode tables:
0/1601/1602/1603/1604/1605/1606/1607/1608/1609/16010/16011/16012/16013/16014/16015/16016/16017/16018/16019/16020/
16021/16022/16023/16024/16025/16026/16027/16028/16029/16030/16031/16032/16033/16034/16035/16036/16037/16038/16039
/16040/16041/16042/16043/16044/16045/16046/16047/16048/16049/16050/16051/16052/16053/16054/16055/16056/16057/1605
8/16059/16060/16061/16062/16063/16064/16065/16066/16067/16068/16069/16070/16071/16072/16073/16074/16075/16076/160
77/16078/16079/16080/16081/16082/16083/16084/16085/16086/16087/16088/16089/16090/16091/16092/16093/16094/16095/16
096/16097/16098/16099/160100/160101/160102/160103/160104/160105/160106/160107/160108/160109/160110/160111/160112/
160113/160114/160115/160116/160117/160118/160119/160120/160121/160122/160123/160124/160125/160126/160127/160128/1
60129/160130/160131/160132/160133/160134/160135/160136/160137/160138/160139/160140/160141/160142/160143/160144/16
0145/160146/160147/160148/160149/160150/160151/160152/160153/160154/160155/160156/160157/160158/160159/160done
Creating journal (32768 blocks): done
Writing superblocks and filesystem accounting information: mount -t ext4 /dev/xvdf /mnt
done

This filesystem will be automatically checked every 27 mounts or
180 days, whichever comes first.  Use tune2fs -c or -i to override.
[root@ip-172-31-38-72 ~]# mount -t ext4 /dev/xvdf /mnt
cd /mnt
[root@ip-172-31-38-72 ~]# cd /mnt
[root@ip-172-31-38-72 mnt]# mkdir -p images
[root@ip-172-31-38-72 mnt]# mkfs -t ext4 /dev/xvdg
mke2fs 1.41.12 (17-May-2010)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
Stride=0 blocks, Stripe width=0 blocks
655360 inodes, 2621440 blocks
131072 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=2684354560
80 block groups
32768 blocks per group, 32768 fragments per group
8192 inodes per group
Superblock backups stored on blocks:
       32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632

Writing inode tables:
0/801/802/803/804/805/806/807/808/809/8010/8011/8012/8013/8014/8015/8016/8017/8018/8019/8020/8021/8022/8023/8024/
8025/8026/8027/8028/8029/8030/8031/8032/8033/8034/8035/8036/8037/8038/80mkdir -p /opt/ec2/mnt
39/8040/8041/8042/8043/8044/8045/8046/8047/8048/8049/8050/8051/8052/8053/8054/8055/8056/8057/8058/8059/8060/8061/
8062/8063/8064/8065/8066/8067/8068/8069/8070/8071/8072/8073/8074/8075/8076/8077/8078/8079/80done
Creating journal (32768 blocks): done
Writing superblocks and filesystem accounting information: mount -t ext4 /dev/xvdg /opt/ec2/mnt
done
```

```
This filesystem will be automatically checked every 24 mounts or
180 days, whichever comes first.  Use tune2fs -c or -i to override.
[root@ip-172-31-38-72 mnt]# mkdir -p /opt/ec2/mnt
[root@ip-172-31-38-72 mnt]# mount -t ext4 /dev/xvdg /opt/ec2/mnt
[root@ip-172-31-38-72 mnt]# exit
logout
Connection to 54.187.12.76 closed.

spawned process completed...
Back to script. Please wait...
Copying (SCP) Raw image to AWS. Please wait...
spawn ssh -i khs-fermi.pem root@54.187.12.76
Last login: Sat Aug  2 16:56:57 2014 from 131.225.155.50

                              NOTICE TO USERS

       This  is a Federal computer (and/or it is directly connected to a
       Fermilab local network system) that is the property of the United
       States Government.  It is for authorized use only.  Users (autho-
       rized or unauthorized) have no explicit or  implicit  expectation
       of privacy.

       Any  or  all uses of this system and all files on this system may
       be intercepted, monitored, recorded,  copied, audited, inspected,
       and  disclosed  to authorized site, Department of Energy  and law
       enforcement personnel, as  well as authorized officials of  other
       agencies,  both  domestic and foreign.  By using this system, the
       user consents to such interception, monitoring, recording,  copy-
       ing,  auditing,  inspection,  and disclosure at the discretion of
       authorized site or Department of Energy personnel.

       Unauthorized or improper use of this system may result in  admin-
       istrative  disciplinary  action and civil and criminal penalties.
       By continuing to use this system you indicate your  awareness  of
       and  consent to these terms and conditions of use.  LOG OFF IMME-
       DIATELY if you do not agree to  the  conditions  stated  in  this
       warning.

       Fermilab  policy  and  rules for computing, including appropriate
       use, may be found at http://www.fnal.gov/cd/main/cpolicy.html
[root@ip-172-31-38-72 ~]# cd /mnt/images
[root@ip-172-31-38-72 images]# mkdir -p raw
[root@ip-172-31-38-72 images]# kpartx -a gcso_sl6.raw
[root@ip-172-31-38-72 images]# mount /dev/mapper/loop0p1 /mnt/images/raw
[root@ip-172-31-38-72 images]# cd /mnt/images/raw
[root@ip-172-31-38-72 raw]# rsync -aqHx /mnt/images/raw/ /opt/ec2/mnt
[root@ip-172-31-38-72 raw]# rsync -aqHx /mnt/images/raw/dev /opt/ec2/mnt
```

```
[root@ip-172-31-38-72 raw]# cd /opt/ec2/mnt
[root@ip-172-31-38-72 mnt]# tune2fs -L '/' /dev/xvdg
tune2fs 1.41.12 (17-May-2010)
[root@ip-172-31-38-72 mnt]# sync;sync;sync;sync
[root@ip-172-31-38-72 mnt]# cd /mnt/images
[root@ip-172-31-38-72 images]# umount /mnt/images/raw
[root@ip-172-31-38-72 images]# kpartx -d gcso_sl6.raw
loop deleted : /dev/loop0
[root@ip-172-31-38-72 images]# exit
logout
Connection to 54.187.12.76 closed.

spawned process completed...
Back to script. Please wait...
ATTACHMENT   vol-af1680ae i-94fe889f   /dev/sdg     detaching    2014-08-02T21:56:45+0000
TAG    volume vol-af1680ae Name   GCSO_SL6_PV_20140802165545
TAG    volume vol-af1680ae User   kirkshal
TAG    snapshot     snap-117fdee6      Name   GCSO_SL6_PV_20140802165545
TAG    snapshot     snap-117fdee6      User   kirkshal
TAG    image  ami-9fd2a9af Name   GCSO_SL6_PV_20140802165545
TAG    image  ami-9fd2a9af User   kirkshal
TAG    instance     i-abe593a0   Name   GCSO_SL6_PV_20140802165545
TAG    instance     i-abe593a0   User   kirkshal
TAG    volume vol-79178178 Name   GCSO_SL6_PV_20140802165545
TAG    volume vol-79178178 User   kirkshal
INSTANCE     i-94fe889f   running      stopping
INSTANCE     i-abe593a0   running      stopping
VOLUME vol-af1680ae
AWS VM Import Completed.
Aug 02 17:07:38 INFO     Stop: Completed Importing the worker Fermicloud VM image. Function import_image.
Aug 02 17:07:38 INFO     End script run

-----------
-----------
-----------
```

File: puppetrepo-awsexport/files/opt/gcso/awsexport/docs/create_samplemime.txt

```
$ vi my-user-script.txt
#!/bin/bash
echo "=========Hello Fermi=========" > /root/fermihello1.txt
:wq

$ vi my-cloudconfig.txt
#cloud-config
cloud_final_modules:
 - rightscale_userdata
```

```
 - scripts-per-once
 - scripts-per-boot
 - scripts-per-instance
 - [scripts-user, always]
 - ssh-authkey-fingerprints
 - keys-to-console
 - phone-home
 - final-message

runcmd:

 - [ sh, -c, echo "=========Hello Fermi=========" > /root/fermihello2.txt ]
:wq

$ write-mime-multipart --output=samplemime.txt \
   my-user-script.txt:text/x-shellscript \
   my-cloudconfig.txt
-----------
-----------
-----------


File: puppetrepo-awsexport/files/opt/gcso/awsexport/docs/fermi_VM_Convert.txt
# This file contains working steps to manually create a AWS working instance. From this I was able to create the
automated
# scripts. Note: Not every command was used in the automated scripts these are working notes only.

# Copy golden image from fermicloud.fnal.gov to data area on fermicloud vm (~2 minutes)
scp kirkshal@fermicloud.fnal.gov:/var/lib/one/local/pub_scratch/gcso_sl6.img
root@fermicloud103.fnal.gov:/data/gcso_sl6.qcow2
scp kirkshal@fermicloud.fnal.gov:/var/lib/one/local/images/fa5219771174d0a16dd4ba0b0cf9e012
root@fermicloud103.fnal.gov:/data/gcso_sl6.qcow2
scp kirkshal@fermicloud.fnal.gov:/var/lib/one/local/images/21022925deb7edd64800ebb3bf97f102
root@fermicloud103.fnal.gov:/data/gcso_sl6.qcow2

# Clean Up Image for conversion to aws
ssh root@fermicloud103.fnal.gov
virt-rescue /data/gcso_sl6.qcow2
fdisk -l
fdisk /dev/sda
p
d
2
w

#Exit virt-rescue
exit
```

```
cd /data
mkdir work
guestmount -a gcso_sl6.qcow2 -m /dev/sda1 work

# cp /grid/data/parag/gwms-cloudvm-rpms/glideinwms-vm-core-0.4-0.2.rc2.el6.noarch.rpm work
# cp /grid/data/parag/gwms-cloudvm-rpms/glideinwms-vm-ec2-0.4-0.2.rc2.el6.noarch.rpm work

chroot work

cd /etc/sysconfig/network-scripts
vi ifcfg-eth0
DEVICE=eth0
BOOTPROTO=dhcp
ONBOOT=yes
TYPE=Ethernet

:wq


cd /etc/sysconfig
vi /etc/sysconfig/network

from:

NETWORKING=yes
HOSTNAME=localhost.localdomain
GATEWAY=131.225.154.1

to:

NETWORKING=yes

:wq

cd /etc
rm /etc/resolv.conf
y
rm /etc/hosts
y
rm /etc/hosts.allow
y
rm /etc/hosts.deny
y

vi /boot/grub/grub.conf

from: (for Xen HVM Only)
```

```
# grub.conf generated by anaconda
#
# Note that you do not have to rerun grub after making changes to this file
# NOTICE:  You do not have a /boot partition.  This means that
#          all kernel and initrd paths are relative to /, eg.
#          root (hd0,0)
#          kernel /boot/vmlinuz-version ro root=/dev/vda1
#          initrd /boot/initrd-[generic-]version.img
#boot=/dev/vda
default=0
timeout=5
splashimage=(hd0,0)/boot/grub/splash.xpm.gz
hiddenmenu
title Scientific Linux Fermi (2.6.32-431.11.2.el6.x86_64)
        root (hd0,0)
        kernel /boot/vmlinuz-2.6.32-431.11.2.el6.x86_64 ro root=UUID=17898259-6979-4bb3-9d73-26ae917e8ed9
rd_NO_LUKS rd_NO_LVM LANG=en_US.UTF-8 rd_NO_MD SYSFONT=latarcyrheb-sun16 crashkernel=auto  KEYBOARDTYPE=pc
KEYTABLE=us rd_NO_DM rhgb quiet
        initrd /boot/initramfs-2.6.32-431.11.2.el6.x86_64.img

to: (for Xen PV Only)

default=0
timeout=0
title Scientific Linux Fermi (2.6.32-431.11.2.el6.x86_64)
 root (hd0)
 kernel /boot/vmlinuz-2.6.32-431.11.2.el6.x86_64 ro root=/dev/xvde1 rd_NO_PLYMOUTH
 initrd /boot/initramfs-2.6.32-431.11.2.el6.x86_64.img

:wq

cd /boot; ln -s . boot

vi /etc/fstab

from: (for Xen HVM Only)

#
# /etc/fstab
# Created by anaconda on Fri May  2 13:00:03 2014
#
# Accessible filesystems, by reference, are maintained under '/dev/disk'
# See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info
#
UUID=17898259-6979-4bb3-9d73-26ae917e8ed9 /                       ext3    defaults        1 1
UUID=2c7666fb-e233-4793-899e-c783e3e4f328 swap                    swap    defaults        0 0
UUID=b0dcfd19-a2b1-41a1-8ff6-168732f909dd swap                    swap    defaults        0 0
```

```
tmpfs                   /dev/shm                tmpfs   defaults        0 0
devpts                  /dev/pts                devpts  gid=5,mode=620  0 0
sysfs                   /sys                    sysfs   defaults        0 0
proc                    /proc                   proc    defaults        0 0

to: (for Xen PV Only)

/dev/xvde1              /                       ext4    defaults        1 1
/dev/vdb                swap                    swap    defaults        0 0
tmpfs                   /dev/shm                tmpfs   defaults        0 0
devpts                  /dev/pts                devpts  gid=5,mode=620  0 0
sysfs                   /sys                    sysfs   defaults        0 0
proc                    /proc                   proc    defaults        0 0
/dev/xvdf               /scratch                xfs     defaults        0 0

:wq


vi /etc/ssh/sshd_config

from:

Protocol 2
RSAAuthentication no
PubkeyAuthentication no
PasswordAuthentication no
ChallengeResponseAuthentication no
UsePAM yes
KerberosAuthentication yes
KerberosOrLocalPasswd no
KerberosTicketCleanup yes
GSSAPIAuthentication yes
GSSAPIKeyExchange yes
GSSAPICleanupCredentials yes
X11Forwarding yes
AllowTcpForwarding yes
UsePrivilegeSeparation yes
UseDNS no
PermitRootLogin yes

to:

SyslogFacility AUTHPRIV
RSAAuthentication no
PubkeyAuthentication yes
AuthorizedKeysFile   .ssh/authorized_keys
PasswordAuthentication yes
KerberosAuthentication no
```

```
KerberosOrLocalPasswd no
KerberosTicketCleanup no
GSSAPIAuthentication no
GSSAPICleanupCredentials no
UsePAM no
AllowTcpForwarding yes
X11Forwarding yes
UseLogin no
UseDNS no

:wq

# Create a script that captures the public key credentials for your root login
vi /etc/init.d/ec2-get-ssh

#!/bin/bash
# chkconfig: 2345 95 20
# processname: ec2-get-ssh
# description: Capture AWS public key credentials for EC2 user

# Source function library
. /etc/rc.d/init.d/functions

# Source networking configuration
[ -r /etc/sysconfig/network ] && . /etc/sysconfig/network

# Replace the following environment variables for your system
export PATH=:/usr/local/bin:/usr/local/sbin:/usr/bin:/usr/sbin:/bin:/sbin

# Check that networking is configured
if [ "${NETWORKING}" = "no" ]; then
  echo "Networking is not configured."
  exit 1
fi

start() {
  if [ ! -d /root/.ssh ]; then
    mkdir -p /root/.ssh
    chmod 700 /root/.ssh
  fi
  # Retrieve public key from metadata server using HTTP
  curl -f http://169.254.169.254/latest/meta-data/public-keys/0/openssh-key > /tmp/my-public-key
  if [ $? -eq 0 ]; then
    echo "EC2: Retrieve public key from metadata server using HTTP."
    cat /tmp/my-public-key >> /root/.ssh/authorized_keys
    chmod 600 /root/.ssh/authorized_keys
    rm /tmp/my-public-key
  fi
```

```
}

stop() {
  echo "Nothing to do here"
}

restart() {
  stop
  start
}

# See how we were called.
case "$1" in
  start)
    start
    ;;
  stop)
    stop
    ;;
  restart)
    restart
    ;;
  *)
    echo $"Usage: $0 {start|stop|restart}"
    exit 1
esac

exit $?
:wq

# Update the runlevel information for the new system service on the image.


/bin/chmod +x /etc/init.d/ec2-get-ssh

chkconfig --level 34 ec2-get-ssh on

chkconfig postfix off
chkconfig autofs off
chkconfig vmcontext off


# rpm -ivh /glideinwms-vm-core-0.4-0.2.rc2.el6.noarch.rpm
# rpm -ivh /glideinwms-vm-ec2-0.4-0.2.rc2.el6.noarch.rpm

# Prevent 10 minute vm shutdown for testing (turn back on after testing)
# chkconfig  glideinwms-pilot off
```

```
# make sure /etc/puppet/modules directory is present if not create it
# download Puppet Module for CVMFS
# puppet module install desalvo-cvmfs

# install cvmfs
# vi /etc/puppet/modules/cvmfs/manifests/client.pp
# add "include cvmfs::client" to the end
# :wq
# puppet apply /etc/puppet/modules/cvmfs/manifests/client.pp

history -c
exit
fusermount -u work
rmdir /data/work


# Resize Image from 256G to 12.1G
# qemu-img create [-f format] [-o options] filename [size]

rm gcso_sl6.raw
y
qemu-img create -f raw gcso_sl6.raw 3075M

# guestfish
cd /data

guestfish -a gcso_sl6.qcow2tmp
e2fsck-f /dev/sda1
resize2fs-size /dev/sda1 3G
# virt-rescue -a gcso_sl6.qcow2tmp
# e2fsck -f /dev/sda1
# resize2fs /dev/sda1 3G
# fdisk -l
# blkid -c /dev/null
exit

# Resize takes about 7 minutes ?
virt-resize --delete /dev/sda2 --resize /dev/sda1=3G gcso_sl6.qcow2tmp gcso_sl6.raw

# Convert Image from QCOW2 to RAW for AWS VM Import (Not used as Virt-Resize replaces this)
# qemu-img convert -p -f qcow2 -O raw gcso_sl6.qcow2 ./gcso_sl6.raw

# Status of converted 3G raw file
qemu-img info gcso_sl6.raw


# Install AWS EC2 API Tools (ec2-api-tools.zip)
# http://docs.aws.amazon.com/AWSEC2/latest/CommandLineReference/set-up-ec2-cli-linux.html
```

```
# Import VM for HVM Snapshot
ec2iin -f RAW -t m3.medium -a x86_64 -b fcloudimport -o AKIAIDJ22XYOSDF3BRPQ -w
SwLUaC5Bi06Bj7KqVKEDW/AOY1qLcLgJ82THqc3C -p Linux --region us-west-2 -z us-west-2a gcso_sl6.raw

# Average speed was 22.738 MBps to upload import (~ 10 minutes)
# Approx. 20 minutes to convert to AWS Instance after upload
# Total time approx 30 minutes.

# Status of import
ec2-describe-conversion-tasks

# Create AMI from Imported Instance
# Snapshot auto created

# Launch new HVM AMI Instance
# Manual
ssh -i khs-fermi.pem root@nn.nn.nn.nn
# Auto
# ec2-start-instances instance_id [instance_id...]
ec2-start-instances i-650b6b6e
ip=`ec2-describe-instances i-181e8913 | grep NICASSOCIATION | awk {'print($2)'}`
echo $ip
ssh -i khs-fermi.pem root@$ip

# Below only used if pem auth not set up
# Accept key
# yes
# Use root pw  (get from admin if needed)
# mkdir .ssh
# cd .ssh
# vi authorized_keys
# insert your public key
# :wq
# chmod 700 ~/.ssh
# chmod 600 ~/.ssh/authorized_keys
# exit
# ssh -i khs-fermi.pem root@nn.nn.nn.nn

# Create staged VM Image on AWS HVM 30G Worker Instance
fdisk -l
mkfs -t ext4 /dev/xvdf
mount -t ext4 /dev/xvdf /mnt
cd /mnt
mkdir -p /mnt/images

# Create new volume on AWS Worker from fermicloud103
vol=`ec2-create-volume --size 10 --region us-west-2 -z us-west-2a | grep VOLUME | awk {'print($2)'}`
```

```
# Attach volume on AWS Worker from fermicloud103
ec2-attach-volume --instance i-181e8913 --device /dev/sdg vol-010a8600 --region us-west-2

# on AWS Worker
mkfs -t ext4 /dev/xvdg

mkdir -p /opt/ec2/mnt
mount -t ext4 /dev/xvdg /opt/ec2/mnt

# Import VM for PV Snapshot

# Copy .pem and metadata script from Desktop terminal to FermiCloud103 worker VM
scp -Cq /Users/kshallcross/khs-fermi.pem root@fermicloud103.fnal.gov:/data
scp -Cq -i khs-fermi.pem /Users/kshallcross/Desktop/ec2-metadata root@54.191.225.130:/root
# ON AWS Worker
chmod +x ec2-metadata
#List IP addresses
./ec2-metadata -o -v

# From FermiCloud103
cd /data
#rsync -S -z -v --progress -e "ssh -i khs-fermi.pem" /data/gcso_sl6.raw root@nn.nn.nn.nn:/mnt/images
rsync -S -z -q -e "ssh -i khs-fermi.pem" /data/gcso_sl6.raw root@54.191.225.130:/mnt/images
yes to auth

# Rename grub.conf.pv and fstab.pv to grub.conf and fstab on fermicloud103

cd /mnt/images
mkdir raw
kpartx -a gcso_sl6.raw
mount /dev/mapper/loop0p1 /mnt/images/raw
cd /mnt/images/raw
cd etc
mv fstab fstab.hvm
mv fstab.pv fstab
cd ..
cd boot/grub
mv grub.conf grub.conf.hvm
mv grub.conf.pv grub.conf
cd /mnt/images

# Move image to new volume
rsync -aqHx /mnt/images/raw/ /opt/ec2/mnt

rsync -aqHx /mnt/images/raw/dev /opt/ec2/mnt

# Label the disk.
cd /opt/ec2/mnt
```

```
tune2fs -L '/' /dev/xvdf

# Flush all writes and unmount the volume.
sync;sync;sync;sync

cd /mnt/images
umount /mnt/images/raw
kpartx -d gcso_sl6.raw
# dmsetup remove /dev/mapper/loop0p1
# losetup -d /dev/loop0p1


# Move image to new volume (Not used, kpartx rsync above does this)
# sudo dd if=/opt/ec2/mnt/gcso_sl6.raw of=/dev/xvdg bs=10M (not needed)
# scp root@fermicloud103.fnal.gov:/data/gcso_sl6.raw root@54.187.254.70:/dev/xvdh (not needed)

# Detach Volume from fermicloud103
ec2-detach-volume vol-010a8600

# Create Snapshot to be used to create PV AMI
ec2-create-snapshot --region us-west-2 -d "gcso_sl6 pv snap" vol-010a8600

# Create PV AMI
# ec2-register -n "AMI-HK-SLF582" -d "SLF58-Test2" --root-device-name /dev/sda2 -b /dev/sda=snap-fdc5fa09:12:true
--architecture x86_64 --region us-west-2
ec2-register -n "AMI-GCSO-SL6-PV test" -d "AMI-GCSO-SL6-PV test" -b "/dev/sda1=snap-fdc5fa09:10:true:gp2" -b
"/dev/sdb=ephemeral0" --architecture x86_64 --kernel aki-fc8f11cc --region us-west-2

# Create and run Instance from PV AMI
ec2-run-instances --region us-west-2 -z us-west-2a ami-d3bec6e3 -g sg-a6a010c3 -n 1 --kernel aki-fc8f11cc -t
m3.medium -k khs-fermi

# Create tags for all resources
ec2-create-tags vol-010a8600 snap-fdc5fa09 ami-d3bec6e3 i-6fe08164 vol-c525a9c4 --tag "Name=Fermi GCSO_SL6 PV" --
tag "User=kirkshal"

# On new instance mount Scratch on ephemeral0 SSD storage
# Modify (append) /etc/rc.d/rc.local
# mkfs.xfs /dev/xvdf (already ext3)

mkdir -p /scratch
mount /dev/xvdf /scratch


-------------------------------------------------------

#
#
```

```
# Supportive Documentation below...
# Prepare the EBS PV Volume

#
#
# Below are details that support above statements
# Log in to the EC2 instance HVM AMI and create an ext4 filesystem type on the partitionless EBS volume.


/bin/egrep '[xvsh]d[a-z].*$' /proc/partitions
# 202       65   10485760 xvda1
# 202       66  156352512 xvda2
# 202       67     917504 xvda3
# 202      144   10485760 xvdf

mkfs.ext4 /dev/xvdf


# Note that while we specified to attach the EBS volume as device /dev/sdg in the previous step, Amazon Linux
uses the Xen virtual disk notation /dev/xvdn. The volume in my case was attached as /dev/xvdg.

# Create a mount point directory and mount the EBS volume.

mkdir -p /opt/ec2/mnt

mount -t ext4 /dev/xvdf /opt/ec2/mnt

mount

# /dev/xvda1 on / type ext4 (rw)
# none on /proc type proc (rw)
# none on /sys type sysfs (rw)
# none on /dev/pts type devpts (rw,gid=5,mode=620)
# none on /dev/shm type tmpfs (rw)
# /dev/xvda2 on /mnt/vol1 type ext4 (rw)
# none on /proc/sys/fs/binfmt_misc type binfmt_misc (rw)
# /dev/xvdf on /opt/ec2/mnt type ext4 (rw)

# Remove any local instance storage entries from /etc/fstab if they exist. Booting from an EBS volume does not
use local instance storage by default. If you followed this guide to create the instance store-backed AMI, remove
the local instance storage entry in /etc/fstab.

cat /etc/fstab | grep -v mnt > /tmp/fstab

mv /etc/fstab /etc/fstab.bak

mv /tmp/fstab /etc/fstab
```

```
# Sync the root and dev file systems to the EBS volume.

rsync -S -z -v --progress -e "ssh -i khs-fermi.pem" /data/gcso_sl6.raw root@54.187.254.70:/opt/ec2/mnt

rsync -aqHx / /opt/ec2/mnt

rsync -aqHx /dev /opt/ec2/mnt

# Label the disk.

tune2fs -L '/' /dev/xvdg

# Flush all writes and unmount the volume.

sync;sync;sync;sync

umount /opt/ec2/mnt

# Detach Volume

# Create Snapshot from Volume to be used to create PV AMI

# Create PV AMI

# Create Instance from PV AMI

# Modify (append) /etc/rc.d/rc.local
# Make a Scratch file system
mkdir -p /scratch

# Mount and Unmount Paravirtual
mkfs.xfs -f /dev/xvdf
mount /dev/xvdf /scratch
umount /scratch

# Mount and Unmount HVM
mkfs.xfs -f /dev/xvdb
mount /dev/xvdb /scratch
umount /scratch

# display fs info
fdisk -l
df -h
blkid -c /dev/null

# Linux Release info
cat /etc/*release
uname -a
```

```
# Find files
find / -name xxx* 2> /dev/null

#Setup Expect
yum install tcl-devel tk-devel expect-devel expectk

-----------
-----------
-----------


File: puppetrepo-awsexport/files/opt/gcso/awsexport/docs/samplemime.txt
Content-Type: multipart/mixed; boundary="===============3302562582348490243==="
MIME-Version: 1.0

--================3302562582348490243==
Content-Type: text/x-shellscript; charset="us-ascii"
MIME-Version: 1.0
Content-Transfer-Encoding: 7bit
Content-Disposition: attachment; filename="my-user-script.txt"

#!/bin/bash
echo "=========Hello Fermi=========" > /root/fermihello1.txt

--================3302562582348490243==
Content-Type: text/cloud-config; charset="us-ascii"
MIME-Version: 1.0
Content-Transfer-Encoding: 7bit
Content-Disposition: attachment; filename="my-cloudconfig.txt"

#cloud-config
cloud_final_modules:
 - rightscale_userdata
 - scripts-per-once
 - scripts-per-boot
 - scripts-per-instance
 - [scripts-user, always]
 - ssh-authkey-fingerprints
 - keys-to-console
 - phone-home
 - final-message

runcmd:

 - [ sh, -c, echo "=========Hello Fermi=========" > /root/fermihello2.txt ]

--================3302562582348490243==--
```

```
-----------
-----------
-----------


File: puppetrepo-awsexport/files/opt/gcso/awsexport/ec2-get-ssh
#!/bin/bash
# chkconfig: 2345 95 20
# processname: ec2-get-ssh
# description: Capture AWS public key credentials for EC2 user

# Source function library
. /etc/rc.d/init.d/functions

# Source networking configuration
[ -r /etc/sysconfig/network ] && . /etc/sysconfig/network

# Replace the following environment variables for your system
export PATH=:/usr/local/bin:/usr/local/sbin:/usr/bin:/usr/sbin:/bin:/sbin

# Check that networking is configured
if [ "${NETWORKING}" = "no" ]; then
  echo "Networking is not configured."
  exit 1
fi

start() {
  if [ ! -d /root/.ssh ]; then
    mkdir -p /root/.ssh
    chmod 700 /root/.ssh
  fi
  # Retrieve public key from metadata server using HTTP
  curl -f http://169.254.169.254/latest/meta-data/public-keys/0/openssh-key > /tmp/my-public-key
  if [ $? -eq 0 ]; then
    echo "EC2: Retrieve public key from metadata server using HTTP."
    cat /tmp/my-public-key >> /root/.ssh/authorized_keys
    chmod 600 /root/.ssh/authorized_keys
    rm /tmp/my-public-key
  fi
}

stop() {
  echo "Nothing to do here"
}

restart() {
  stop
  start
```

```
}

# See how we were called.
case "$1" in
  start)
    start
    ;;
  stop)
    stop
    ;;
  restart)
    restart
    ;;
  *)
    echo $"Usage: $0 {start|stop|restart}"
    exit 1
esac

exit $?
-----------
-----------
-----------


File: puppetrepo-awsexport/files/opt/gcso/awsexport/gcso.pem
-----BEGIN RSA PRIVATE KEY-----

[SNIP]

-----END RSA PRIVATE KEY-----
-----------
-----------
-----------


File: puppetrepo-awsexport/manifests/awsexport.pp
# include awsexport
class { 'awsexport': }
-----------
-----------
-----------


File: puppetrepo-awsexport/manifests/awsexport_params.pp
#contains parameters for the awsexport::awsexport class

define awsexport::awsexport_params (
      $cvm_file_location,
```

```
    $cvm_image_location,
    $ckernel_ver,
    $cvm_number,
    $cvm_name,
    $cvm_owner,
    $caws_image_name,
    $caws_image_owner,
    $caws_instance,
    $caws_key,
    $caws_secret_key,
    $caws_pem_name,
    $caws_worker_instance_id,
    $caws_owner_keypair_name,
    $caws_eph_mount
)
{
include awsexport

# create a cron job

$command = "/opt/gcso/awsexport/Convert.py $cvm_file_location $cvm_image_location $ckernel_ver $cvm_number
$cvm_name $cvm_owner $caws_image_name $caws_image_owner $caws_instance $caws_key $caws_secret_key $caws_pem_name
$caws_worker_instance_id $caws_owner_keypair_name $caws_eph_mount"

$environmentv = [ 'MAILTO=kirkshal@fnal.gov', 'JAVA_HOME=/usr/lib/jvm/jre-1.6.0-openjdk.x86_64',
'EC2_BASE=/usr/local/ec2', 'EC2_HOME=/usr/local/ec2/ec2-api-tools-1.7.1.0', 'EC2_URL=https://ec2.us-west-
2.amazonaws.com', 'PATH=/usr/lib64/qt-
3.3/bin:/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin:/root/bin:/usr/local/ec2/ec2-api-tools-
1.7.1.0/bin', 'AWS_ACCOUNT_NUMBER=159067897602', "AWS_ACCESS_KEY_ID=${caws_key}",
"AWS_SECRET_ACCESS_KEY=${caws_secret_key}" ]

  cron {'awsexport':

    minute      => '15',

    hour        => '00',

    monthday    => '26',

    month       => '8',

    weekday     => '*',

    user        => 'root',

    command     => $command,

    environment => $environmentv;
```

```
  }

}
-----------
-----------
-----------


File: puppetrepo-awsexport/manifests/gcso_sl6_giwms_pv.pp
#contains parameters for the gcso_sl6 GIWMS PV image conversion
    awsexport::awsexport_params {'gcso_sl6_giwms_pv':
     cvm_file_location => '/opt/gcso/awsexport',
       cvm_image_location => 'oneadmin@fcl008:/var/lib/one/local/images/6a50ad27120ad62979d75ba2dbfc8e98',
     ckernel_ver => '2.6.32-431.23.3.el6.x86_64',
     cvm_number => '103',
     cvm_name => 'gcso_sl6_giwms',
     cvm_owner => 'kirkshal',
     caws_image_name => 'GCSO_SL6_GIWMS_PV',
     caws_image_owner => 'gcso',
     caws_instance => 'm3.medium',
     caws_key => 'AKIAJX75FCYK6SAOW4DQ',
     caws_secret_key => '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',
     caws_pem_name => 'gcso.pem',
     caws_worker_instance_id => 'i-3eb6e435',
     caws_owner_keypair_name => 'gcso',
     caws_eph_mount => 'none',
    }
-----------
-----------
-----------


File: puppetrepo-awsexport/manifests/gcso_sl6_hvm.pp
#contains parameters for the gcso_sl6 HVM image conversion
    awsexport::awsexport_params {'gcso_sl6_hvm':
     cvm_file_location => '/opt/gcso/awsexport',
       cvm_image_location => 'oneadmin@fcl008:/var/lib/one/local/images/55c42a4cc7f87ea3390bc2bef14212c5',
     ckernel_ver => '2.6.32-431.23.3.el6.x86_64',
     cvm_number => '103',
     cvm_name => 'gcso_sl6',
     cvm_owner => 'kirkshal',
     caws_image_name => 'GCSO_SL6_HVM',
     caws_image_owner => 'gcso',
     caws_instance => 'm3.medium',
     caws_key => 'AKIAJX75FCYK6SAOW4DQ',
     caws_secret_key => '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',
     caws_pem_name => 'gcso.pem',
```

```
     caws_worker_instance_id => 'hvm',
     caws_owner_keypair_name => 'gcso',
     caws_eph_mount => '/scratch',
    }
-----------
-----------
-----------


File: puppetrepo-awsexport/manifests/gcso_sl6_pv.pp
#contains parameters for the gcso_sl6 PV image conversion
    awsexport::awsexport_params {'gcso_sl6_pv':
     cvm_file_location => '/opt/gcso/awsexport',
       cvm_image_location => 'oneadmin@fcl008:/var/lib/one/local/images/55c42a4cc7f87ea3390bc2bef14212c5',
     ckernel_ver => '2.6.32-431.23.3.el6.x86_64',
     cvm_number => '103',
     cvm_name => 'gcso_sl6',
     cvm_owner => 'kirkshal',
     caws_image_name => 'GCSO_SL6_PV',
     caws_image_owner => 'gcso',
     caws_instance => 'm3.medium',
     caws_key => 'AKIAJX75FCYK6SAOW4DQ',
     caws_secret_key => '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',
     caws_pem_name => 'gcso.pem',
     caws_worker_instance_id => 'i-3eb6e435',
     caws_owner_keypair_name => 'gcso',
     caws_eph_mount => '/scratch',
    }
-----------
-----------
-----------


File: puppetrepo-awsexport/manifests/init.pp
# This class deploys all the files for the awsexport scripts that Kirk has developed (Aug 2014).
include cron
class awsexport {

# create awsexport source directory
  file {'/opt/gcso/awsexport':
    ensure => 'directory',
    owner  => 'root',
    group  => 'root',
    recurse => true,
    source => 'puppet:///files/opt/gcso/awsexport',
  }

# create data directory
```

```
  file { "/data":
    ensure => "directory",
      owner  => 'root',
    group  => 'root',
    recurse => true,
  }

# Setup by path properties for the different puppet resources

  Exec {
    path => ['/usr/lib64/qt-
3.3/bin','/usr/krb5/bin','/bin','/usr/local/bin','/usr/bin','/usr/local/sbin','/usr/sbin','/sbin','/root/bin','/u
sr/local/ec2/ec2-api-tools-1.7.1.0/bin'],
  }

}
-----------
-----------
-----------
```