**Intel Encryption and Measurement Patches for Openstack**

**( XEN + KVM )**

**and**

**Openstack Patch Tool**

Ver. 2.0

***Overview :***

Openstack is modified for integration with RPCore enabling encryption and measurement of virtual images. A particular image can be uploaded to Openstack ( glance ) as an encrypted or plain image. Encrypted image also contains the Key uploaded in the KMS. For both types of images, the manifest.xml file is also uploaded over glance. The id of manifest file is updated as an the extended-attribute (MANIFEST\_UUID) for a particular image. Once the image and manifest are uploaded, the VM is ready to instantiate.

When user instantiates a measured and encrypted VM image, the nova compute downloads the image in instances/\_base dir. Once download is complete, the “mhagent” is called to retrieve the Disk Encryption Key from KMS. The decrypted image in kept in \_base dir.

The image in \_base dir is then copied into #INSTANCE# dir and then boot sequence is initiated. Once the request is sent to hypervisor ( KVM ), rp\_proxy intercepts the call and decides if this is a vm launch call. If yes, then it initiates the measurement and launches the VM if measurement is successful, otherwise, VM launch fails.

Following are the component list of changes included into patch :

1. Dashboard
   1. File : create\_instance.py
      * Added the Measured UI flags to be shown over the launch\_instance view
      * Passed it to other components to be shared across.
   2. File : nova.py
      * Added an is\_measured flag to server\_create method
2. python-nova :
   1. File : nova/virt/libvirt/driver.py
      * It retrieves the following parameter for a launch “mh\_encrypted”, “mh\_checksum”, “mh\_dek\_url” for a particular image and determine if the image is encrypted or plain.
      * Depending upon encryption / plain, it passes on parameters download the image from glance and decrypt it, if required.
      * It also download the manifest file after obtaining the image attributed under “manifest\_uuid”
      * The decrypted image resides in \_base with the image id along with the manifest file.
      * Further launches are passed from nova after using the decrypted image residing in \_base.
   2. File : nova/virt/libvirt/Utils.py
      * Utils.py contains modifications related to decryption of image using mhagent. As described in point 2.1.2 above.
   3. File : nova/compute/api.py
      * This file contains the base\_options for an image. Added is\_measured to these options. The base options are validated against the incoming request and hence is\_measured needs to be added to this data structure.
      * This file also contains a call for create and create\_instance which consumes the is\_measured flag to pass it ahead.
   4. File : nova/compute/manager.py
      * Added handling for measured launch exceptions
   5. File : nova/conductor/manager.py
      * Added is\_measured as an allowed value to be updated.
   6. File : nova/scheduler/manager.py
      * Added the MeasuredLaunch exception in the list of expected Exceptions.
   7. File : nova/api/openstack/compute/servers.py
      * Contains methods like create, extract related to servers. Added is\_measured flag to such methods.
   8. File : nova/db/sqlalchemy/models.py
      * Added the is\_measured column to Instance class.
   9. File : 235\_add\_is\_measured\_to\_instance.py
      * This file is used in migration to add is\_measured flag inside the database tables.
   10. File : nova/exception.py
       * Defines MeasuredLaunch Exception
   11. File : nova/virt/xenapi/image/glance.py
       * Added methods to download the manifest file and update “mh\_encrypted” and “mh\_checksum” instance variables of Instance class.
   12. File : nova/virt/xenapi/vmops.py
       * Added handling for RPCORE denied message while starting a particular instance.
   13. File : nova/virt/xenapi/driver.py
       * Added handling to connect to xenapi proxy instead of xen-api. Also updated the exception handling to manage MeasuredLaunch Exception
   14. File : nova/virt/xenapi/vm\_utils.py
       * Added “is\_measured” flag to create method for handling measurement option.
3. Python-novaclient
   1. File : servers.py
      * This python module contains a server manager for users for nova using version v1\_1. It creates instances for Servers. We added an “is\_measured” flag during the creation of VM instances.

*Note: base.py changes are required in Havana but not in Icehouse as the \_boot method is moved from base.py to servers.py.*

***Patch tool:***

The patch tool resides in git at location <TBD>. This tool will check the version of current installation of openstack and applies patches over it accordingly. If the version is Icehouse and the current machine role is “controller” it will patch all the files required over the controller and restart the services to take effect.

If the script finds a mismatch with existing Openstack installation and current available patch versions, the script will exit giving a mismatch message.

Otherwise, it will apply the patch for provided argument.

*Note : This tool takes a backup of the existing python modules and copies the new files. Hence, this is not a traditional patch.*

e.g :

Following will apply the patches for controller

bash ./Openstack\_applyPatches.sh --controller

Following will apply the patches for compute node

bash ./Openstack\_applyPatches.sh --compute

***How to add patches for newer Openstack versions :***

The script gains information the release version of Openstack and the structure of files in patch dir. Once the python modules are copied, it resyncs the database and restarts the respective services.

For e.g:

Suppose we need to add a patch for /usr/lib/python2.7/dist-packages/nova/exception.py, for version Icehouse ( 1:2014 )

Create the dir with same structure under "patch" for version

$ mkdir 1:2014

cd 1:2014

mkdir -p usr/lib/python2.7/dist-packages/nova/

cp <your\_code\_location>/exception.py usr/lib/python2.7/dist-packages/nova/.

Once the file is copied, the script picks up the files and replaces it over the patched system

It removes all the corresponding \*.pyc files

resyncs the DB

And restarts the respective services.