nova-zvm-virt-driver Documentation

Release 2015.1

OpenStack Foundation

Aug 24, 2017

Contents

Welcome to nova-zvm driver's documentation!

System z is a family name used by IBM for all of its mainframe computers. IBM System z are the direct descendants of System/360, announced in 1964, and the System/370 from 1970s, and now includes the IBM System z9, the IBM System z10 and the newer IBM zEnterprise. System z is famous for its high availability and used in government, financial services, retail, manufacturing, and just about every other industry.

z/VM is a hypervisor for the IBM System z platform that provides a highly flexible test and production environment. z/VM offers a base for customers who want to exploit IBM virtualization technology on one of the industry's best-of-breed server environments, the IBM System z family.

zVM drivers, consist of a set of drivers/plugins for different OpenStack components, enables OpenStack to communicate with z/VM hypervisor to manage z/VM system and virtual machines running on the system.

xCAT is an open source scalable distributed computing management and provisioning tool that provides a unified interface for hardware control, discovery, and OS diskful/diskfree deployment. For more info, please refer to http://xcat.org/ and https://github.com/xcat2/xcat-core.

Overview

Topology

Generic concepts and components

Above picture shows a conceptual view of the relationship between any OpenStack solution and z/VM.

An OpenStack solution is free to run its components wherever it wishes; its options range from running all components on z/VM, to running some on z/VM and others elsewhere, to running all components on other platform(s). The solution is also free to source its components wherever it wishes, either using z/VM. OpenStack enablement components or not.

z/VM ships a set of servers that provide local system management APIs. These servers consist of request servers that accept local connections, receive the data, and then call one of a set of worker servers to process the request. These servers are known collectively as SMAPI. The worker servers can interact with the z/VM hypervisor (CP) or with a directory manager. A directory manager is required for this environment.

Beginning with z/VM version 6.3, additional functionality is provided by integrated xCAT services. xCAT is an Open Source scalable distributed computing management and provisioning tool that provides a unified interface for hardware control, discovery, and deployment, including remote access to the SMAPI APIs. It can be used for the deployment and administration of Linux servers that OpenStack wants to manipulate. The z/VM drivers in the OpenStack services communicate with xCAT services via REST APIs to manage the virtual servers.

xCAT is composed of two main services: the xCAT management node (xCAT MN) and ZHCP. Both the xCAT MN server and the ZHCP server run within the same virtual machine, called the OPNCLOUD virtual machine. The xCAT MN coordinates creating, deleting and updating virtual servers. The management node uses a z/VM hardware control point (ZHCP) to communicate with SMAPI to implement changes on a z/VM host. Only one instance of the xCAT MN is necessary to support multiple z/VM hosts. Each z/VM host runs one instance of ZHCP. xCAT MN supports both a GUI for human interaction and REST APIs for use by programs (for example, OpenStack).

Overall architecture
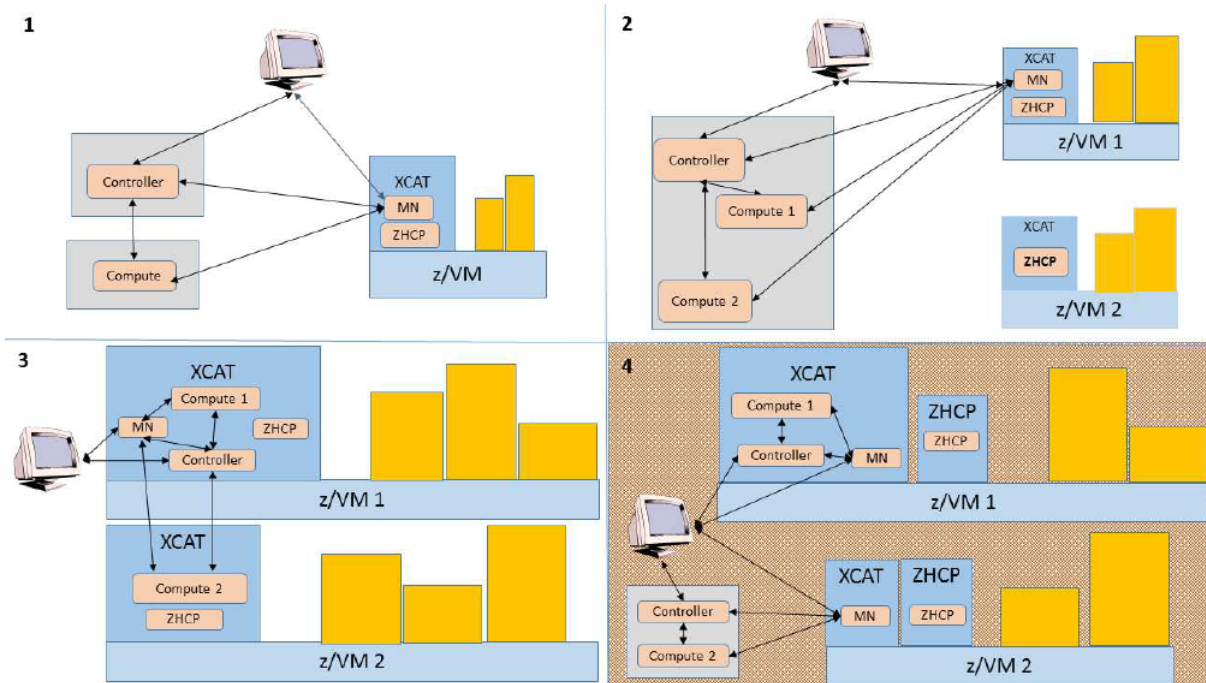
Feature Support Matrix

> **Warning:** Please note, while this document is still being maintained, this is slowly being updated to re-group and classify features.

When considering which capabilities should be marked as mandatory the following general guiding principles were applied

- **Inclusivity** - people have shown ability to make effective use of a wide range of virtualization technologies with broadly varying featuresets. Aiming to keep the requirements as inclusive as possible, avoids second-guessing what a user may wish to use the cloud compute service for.

- **Bootstrapping** - a practical use case test is to consider that starting point for the compute deploy is an empty data center with new machines and network connectivity. The look at what are the minimum features required of a compute service, in order to get user instances running and processing work over the network.

- **Competition** - an early leader in the cloud compute service space was Amazon EC2. A sanity check for whether a feature should be mandatory is to consider whether it was available in the first public release of EC2. This had quite a narrow featureset, but none the less found very high usage in many use cases. So it serves to illustrate that many features need not be considered mandatory in order to get useful work done.

- **Reality** - there are many virt drivers currently shipped with Nova, each with their own supported feature set. Any feature which is missing in at least one virt driver that is already in-tree, must by inference be considered optional until all in-tree drivers support it. This does not rule out the possibility of a currently optional feature becoming mandatory at a later date, based on other principles above.

Summary

Feature	Status	IBM z/VM
Attach block volume to instance	optional	
Detach block volume from instance	optional	
Attach virtual network interface to instance	optional	
Detach virtual network interface from instance	optional	
Set the host in a maintenance mode	optional	
Evacuate instances from a host	optional	
Rebuild instance	optional	
Guest instance status	mandatory	
Guest host status	optional	
Live migrate instance across hosts	optional	
Force live migration to complete	optional	
Launch instance	mandatory	
Stop instance CPUs (pause)	optional	
Reboot instance	optional	
Rescue instance	optional	
Resize instance	optional	
Restore instance	optional	
Service control	optional	
Set instance admin password	optional	
Save snapshot of instance disk	optional	
Suspend instance	optional	
Swap block volumes	optional	
Shutdown instance	mandatory	
Trigger crash dump	optional	
Resume instance CPUs (unpause)	optional	
Auto configure disk	optional	
Instance disk I/O limits	optional	
Config drive support	choice	
Inject files into disk image	optional	
Inject guest networking config	optional	
Remote desktop over RDP	choice	
View serial console logs	choice	
Remote interactive serial console	choice	
Remote desktop over SPICE	choice	
Remote desktop over VNC	choice	
Block storage support	optional	
Block storage over fibre channel	optional	
Block storage over iSCSI	condition	
CHAP authentication for iSCSI	optional	
Image storage support	mandatory	
Network firewall rules	optional	
Network routing	optional	
Network security groups	optional	
Flat networking	choice	
VLAN networking	choice	
uefi boot	optional	

Details

- **Attach block volume to instance Status: optional.** The attach volume operation provides a means to hotplug additional block storage to a running instance. This allows storage capabilities to be expanded without interrup-

tion of service. In a cloud model it would be more typical to just spin up a new instance with large storage, so the ability to hotplug extra storage is for those cases where the instance is considered to be more of a pet than cattle. Therefore this operation is not considered to be mandatory to support.

CLI commands:

– `nova volume-attach <server> <volume>`

drivers:

– **IBM z/VM:** `complete`

• **Detach block volume from instance Status: optional.** See notes for attach volume operation.

CLI commands:

– `nova volume-detach <server> <volume>`

drivers:

– **IBM z/VM:** `complete`

• **Attach virtual network interface to instance Status: optional.** The attach interface operation provides a means to hotplug additional interfaces to a running instance. Hotplug support varies between guest OSes and some guests require a reboot for new interfaces to be detected. This operation allows interface capabilities to be expanded without interruption of service. In a cloud model it would be more typical to just spin up a new instance with more interfaces.

CLI commands:

– `nova interface-attach <server>`

drivers:

– **IBM z/VM:** `missing`

• **Detach virtual network interface from instance Status: optional.** See notes for attach-interface operation.

CLI commands:

– `nova interface-detach <server> <port_id>`

drivers:

– **IBM z/VM:** `missing`

• **Set the host in a maintenance mode Status: optional.** This operation allows a host to be placed into maintenance mode, automatically triggering migration of any running instances to an alternative host and preventing new instances from being launched. This is not considered to be a mandatory operation to support. The driver methods to implement are "host_maintenance_mode" and "set_host_enabled".

CLI commands:

– `nova host-update <host>`

drivers:

– **IBM z/VM:** `missing`

• **Evacuate instances from a host Status: optional.** A possible failure scenario in a cloud environment is the outage of one of the compute nodes. In such a case the instances of the down host can be evacuated to another host. It is assumed that the old host is unlikely ever to be powered back on, otherwise the evacuation attempt will be rejected. When the instances get moved to the new host, their volumes get re-attached and the locally stored data is dropped. That happens in the same way as a rebuild. This is not considered to be a mandatory operation to support.

CLI commands:

– `nova evacuate <server>`

– `nova host-evacuate <host>`

drivers:

– **IBM z/VM:** `missing`

- **Rebuild instance Status: optional.** A possible use case is additional attributes need to be set to the instance, nova will purge all existing data from the system and remakes the VM with given information such as 'metadata' and 'personalities'. Though this is not considered to be a mandatory operation to support.

  CLI commands:

  – `nova rebuild <server> <image>`

  drivers:

  – **IBM z/VM:** `missing`

- **Guest instance status Status: mandatory.** Provides a quick report on information about the guest instance, including the power state, memory allocation, CPU allocation, number of vCPUs and cummulative CPU execution time. As well as being informational, the power state is used by the compute manager for tracking changes in guests. Therefore this operation is considered mandatory to support.

  drivers:

  – **IBM z/VM:** `complete`

- **Guest host status Status: optional.** Unclear what this refers to

  drivers:

  – **IBM z/VM:** `complete`

- **Live migrate instance across hosts Status: optional.** Live migration provides a way to move an instance off one compute host, to another compute host. Administrators may use this to evacuate instances from a host that needs to undergo maintenance tasks, though of course this may not help if the host is already suffering a failure. In general instances are considered cattle rather than pets, so it is expected that an instance is liable to be killed if host maintenance is required. It is technically challenging for some hypervisors to provide support for the live migration operation, particularly those built on the container based virtualization. Therefore this operation is not considered mandatory to support.

  CLI commands:

  – `nova live-migration <server>`

  – `nova host-evacuate-live <host>`

  drivers:

  – **IBM z/VM:** `complete`

- **Force live migration to complete Status: optional.** Live migration provides a way to move a running instance to another compute host. But it can sometimes fail to complete if an instance has a high rate of memory or disk page access. This operation provides the user with an option to assist the progress of the live migration. The mechanism used to complete the live migration depends on the underlying virtualization subsystem capabilities. If libvirt/qemu is used and the post-copy feature is available and enabled then the force complete operation will cause a switch to post-copy mode. Otherwise the instance will be suspended until the migration is completed or aborted.

  CLI commands:

  – `nova live-migration-force-complete <server> <migration>`

  drivers:

– **IBM z/VM:** `missing`

- **Launch instance Status: mandatory.** Importing pre-existing running virtual machines on a host is considered out of scope of the cloud paradigm. Therefore this operation is mandatory to support in drivers.

  drivers:

  – **IBM z/VM:** `complete`

- **Stop instance CPUs (pause) Status: optional.** Stopping an instances CPUs can be thought of as roughly equivalent to suspend-to-RAM. The instance is still present in memory, but execution has stopped. The problem, however, is that there is no mechanism to inform the guest OS that this takes place, so upon unpausing, its clocks will no longer report correct time. For this reason hypervisor vendors generally discourage use of this feature and some do not even implement it. Therefore this operation is considered optional to support in drivers.

  CLI commands:

  – `nova pause <server>`

  drivers:

  – **IBM z/VM:** `complete`

- **Reboot instance Status: optional.** It is reasonable for a guest OS administrator to trigger a graceful reboot from inside the instance. A host initiated graceful reboot requires guest co-operation and a non-graceful reboot can be achieved by a combination of stop+start. Therefore this operation is considered optional.

  CLI commands:

  – `nova reboot <server>`

  drivers:

  – **IBM z/VM:** `complete`

- **Rescue instance Status: optional.** The rescue operation starts an instance in a special configuration whereby it is booted from an special root disk image. The goal is to allow an administrator to recover the state of a broken virtual machine. In general the cloud model considers instances to be cattle, so if an instance breaks the general expectation is that it be thrown away and a new instance created. Therefore this operation is considered optional to support in drivers.

  CLI commands:

  – `nova rescue <server>`

  drivers:

  – **IBM z/VM:** `missing`

- **Resize instance Status: optional.** The resize operation allows the user to change a running instance to match the size of a different flavor from the one it was initially launched with. There are many different flavor attributes that potentially need to be updated. In general it is technically challenging for a hypervisor to support the alteration of all relevant config settings for a running instance. Therefore this operation is considered optional to support in drivers.

  CLI commands:

  – `nova resize <server> <flavor>`

  drivers:

  – **IBM z/VM:** `complete`

- **Restore instance Status: optional.** See notes for the suspend operation

  CLI commands:

– `nova resume <server>`

drivers:

– **IBM z/VM:** `missing`

• **Service control Status: optional.** Something something, dark side, something something. Hard to claim this is mandatory when no one seems to know what "Service control" refers to in the context of virt drivers.

drivers:

– **IBM z/VM:** `missing`

• **Set instance admin password Status: optional.** Provides a mechanism to (re)set the password of the administrator account inside the instance operating system. This requires that the hypervisor has a way to communicate with the running guest operating system. Given the wide range of operating systems in existence it is unreasonable to expect this to be practical in the general case. The configdrive and metadata service both provide a mechanism for setting the administrator password at initial boot time. In the case where this operation were not available, the administrator would simply have to login to the guest and change the password in the normal manner, so this is just a convenient optimization. Therefore this operation is not considered mandatory for drivers to support.

CLI commands:

– `nova set-password <server>`

drivers:

– **IBM z/VM:** `complete`

• **Save snapshot of instance disk Status: optional.** The snapshot operation allows the current state of the instance root disk to be saved and uploaded back into the glance image repository. The instance can later be booted again using this saved image. This is in effect making the ephemeral instance root disk into a semi-persistent storage, in so much as it is preserved even though the guest is no longer running. In general though, the expectation is that the root disks are ephemeral so the ability to take a snapshot cannot be assumed. Therefore this operation is not considered mandatory to support.

CLI commands:

– `nova image-create <server> <name>`

drivers:

– **IBM z/VM:** `complete`

• **Suspend instance Status: optional.** Suspending an instance can be thought of as roughly equivalent to suspend-to-disk. The instance no longer consumes any RAM or CPUs, with its live running state having been preserved in a file on disk. It can later be restored, at which point it should continue execution where it left off. As with stopping instance CPUs, it suffers from the fact that the guest OS will typically be left with a clock that is no longer telling correct time. For container based virtualization solutions, this operation is particularly technically challenging to implement and is an area of active research. This operation tends to make more sense when thinking of instances as pets, rather than cattle, since with cattle it would be simpler to just terminate the instance instead of suspending. Therefore this operation is considered optional to support.

CLI commands:

– `nova suspend <server>`

drivers:

– **IBM z/VM:** `missing`

• **Swap block volumes Status: optional.** The swap volume operation is a mechanism for changing a running instance so that its attached volume(s) are backed by different storage in the host. An alternative to this would

be to simply terminate the existing instance and spawn a new instance with the new storage. In other words this operation is primarily targeted towards the pet use case rather than cattle, however, it is required for volume migration to work in the volume service. This is considered optional to support.

CLI commands:

- `nova volume-update <server> <attachment> <volume>`

drivers:

- **IBM z/VM:** `missing`

- **Shutdown instance Status: mandatory.** The ability to terminate a virtual machine is required in order for a cloud user to stop utilizing resources and thus avoid indefinitely ongoing billing. Therefore this operation is mandatory to support in drivers.

  CLI commands:

  - `nova delete <server>`

  drivers:

  - **IBM z/VM:** `complete`

- **Trigger crash dump Status: optional.** The trigger crash dump operation is a mechanism for triggering a crash dump in an instance. The feature is typically implemented by injecting an NMI (Non-maskable Interrupt) into the instance. It provides a means to dump the production memory image as a dump file which is useful for users. Therefore this operation is considered optional to support.

  drivers:

  - **IBM z/VM:** `missing`

- **Resume instance CPUs (unpause) Status: optional.** See notes for the "Stop instance CPUs" operation

  CLI commands:

  - `nova unpause <server>`

  drivers:

  - **IBM z/VM:** `complete`

- **Auto configure disk Status: optional.** something something, dark side, something something. Unclear just what this is about.

  drivers:

  - **IBM z/VM:** `missing`

- **Instance disk I/O limits Status: optional.** The ability to set rate limits on virtual disks allows for greater performance isolation between instances running on the same host storage. It is valid to delegate scheduling of I/O operations to the hypervisor with its default settings, instead of doing fine grained tuning. Therefore this is not considered to be an mandatory configuration to support.

  CLI commands:

  - `nova limits`

  drivers:

  - **IBM z/VM:** `missing`

- **Config drive support Status: choice(guest.setup).** The config drive provides an information channel into the guest operating system, to enable configuration of the administrator password, file injection, registration of SSH keys, etc. Since cloud images typically ship with all login methods locked, a mechanism to set the administrator

password of keys is required to get login access. Alternatives include the metadata service and disk injection. At least one of the guest setup mechanisms is required to be supported by drivers, in order to enable login access.

drivers:

  – **IBM z/VM:** `complete`

• **Inject files into disk image Status: optional.** This allows for the end user to provide data for multiple files to be injected into the root filesystem before an instance is booted. This requires that the compute node understand the format of the filesystem and any partitioning scheme it might use on the block device. This is a non-trivial problem considering the vast number of filesystems in existence. The problem of injecting files to a guest OS is better solved by obtaining via the metadata service or config drive. Therefore this operation is considered optional to support.

drivers:

  – **IBM z/VM:** `complete`

• **Inject guest networking config Status: optional.** This allows for static networking configuration (IP address, netmask, gateway and routes) to be injected directly into the root filesystem before an instance is booted. This requires that the compute node understand how networking is configured in the guest OS which is a non-trivial problem considering the vast number of operating system types. The problem of configuring networking is better solved by DHCP or by obtaining static config via config drive. Therefore this operation is considered optional to support.

drivers:

  – **IBM z/VM:** `complete`

• **Remote desktop over RDP Status: choice(console).** This allows the administrator to interact with the graphical console of the guest OS via RDP. This provides a way to see boot up messages and login to the instance when networking configuration has failed, thus preventing a network based login. Some operating systems may prefer to emit messages via the serial console for easier consumption. Therefore support for this operation is not mandatory, however, a driver is required to support at least one of the listed console access operations.

CLI commands:

  – `nova get-rdp-console <server> <console-type>`

drivers:

  – **IBM z/VM:** `missing`

• **View serial console logs Status: choice(console).** This allows the administrator to query the logs of data emitted by the guest OS on its virtualized serial port. For UNIX guests this typically includes all boot up messages and so is useful for diagnosing problems when an instance fails to successfully boot. Not all guest operating systems will be able to emit boot information on a serial console, others may only support graphical consoles. Therefore support for this operation is not mandatory, however, a driver is required to support at least one of the listed console access operations.

drivers:

  – **IBM z/VM:** `complete`

• **Remote interactive serial console Status: choice(console).** This allows the administrator to interact with the serial console of the guest OS. This provides a way to see boot up messages and login to the instance when networking configuration has failed, thus preventing a network based login. Not all guest operating systems will be able to emit boot information on a serial console, others may only support graphical consoles. Therefore support for this operation is not mandatory, however, a driver is required to support at least one of the listed console access operations. This feature was introduced in the Juno release with blueprint https://blueprints.launchpad.net/nova/+spec/serial-ports

CLI commands:

> **–** `nova get-serial-console <server>`

drivers:

> **– IBM z/VM:** `missing`

- **Remote desktop over SPICE Status: choice(console).** This allows the administrator to interact with the graphical console of the guest OS via SPICE. This provides a way to see boot up messages and login to the instance when networking configuration has failed, thus preventing a network based login. Some operating systems may prefer to emit messages via the serial console for easier consumption. Therefore support for this operation is not mandatory, however, a driver is required to support at least one of the listed console access operations.

  CLI commands:

  > **–** `nova get-spice-console <server> <console-type>`

  drivers:

  > **– IBM z/VM:** `missing`

- **Remote desktop over VNC Status: choice(console).** This allows the administrator to interact with the graphical console of the guest OS via VNC. This provides a way to see boot up messages and login to the instance when networking configuration has failed, thus preventing a network based login. Some operating systems may prefer to emit messages via the serial console for easier consumption. Therefore support for this operation is not mandatory, however, a driver is required to support at least one of the listed console access operations.

  CLI commands:

  > **–** `nova get-vnc-console <server> <console-type>`

  drivers:

  > **– IBM z/VM:** `missing`

- **Block storage support Status: optional.** Block storage provides instances with direct attached virtual disks that can be used for persistent storage of data. As an alternative to direct attached disks, an instance may choose to use network based persistent storage. OpenStack provides object storage via the Swift service, or a traditional filesystem such as NFS/GlusterFS may be used. Some types of instances may not require persistent storage at all, being simple transaction processing systems reading requests & sending results to and from the network. Therefore support for this configuration is not considered mandatory for drivers to support.

  drivers:

  > **– IBM z/VM:** `complete`

- **Block storage over fibre channel Status: optional.** To maximise performance of the block storage, it may be desirable to directly access fibre channel LUNs from the underlying storage technology on the compute hosts. Since this is just a performance optimization of the I/O path it is not considered mandatory to support.

  drivers:

  > **– IBM z/VM:** `complete`

- **Block storage over iSCSI Status: condition(storage.block==missing).** If the driver wishes to support block storage, it is common to provide an iSCSI based backend to access the storage from cinder. This isolates the compute layer for knowledge of the specific storage technology used by Cinder, albeit at a potential performance cost due to the longer I/O path involved. If the driver chooses to support block storage, then this is considered mandatory to support, otherwise it is considered optional.

  drivers:

  > **– IBM z/VM:** `missing`

- **CHAP authentication for iSCSI Status: optional.** If accessing the cinder iSCSI service over an untrusted LAN it is desirable to be able to enable authentication for the iSCSI protocol. CHAP is the commonly used authentication protocol for iSCSI. This is not considered mandatory to support. (?)

  drivers:

  – **IBM z/VM:** `missing`

- **Image storage support Status: mandatory.** This refers to the ability to boot an instance from an image stored in the glance image repository. Without this feature it would not be possible to bootstrap from a clean environment, since there would be no way to get block volumes populated and reliance on external PXE servers is out of scope. Therefore this is considered a mandatory storage feature to support.

  CLI commands:

  – `nova boot --image <image> <name>`

  drivers:

  – **IBM z/VM:** `complete`

- **Network firewall rules Status: optional.** Unclear how this is different from security groups

  drivers:

  – **IBM z/VM:** `missing`

- **Network routing Status: optional.** Unclear what this refers to

  drivers:

  – **IBM z/VM:** `missing`

- **Network security groups Status: optional.** The security groups feature provides a way to define rules to isolate the network traffic of different instances running on a compute host. This would prevent actions such as MAC and IP address spoofing, or the ability to setup rogue DHCP servers. In a private cloud environment this may be considered to be a superfluous requirement. Therefore this is considered to be an optional configuration to support.

  drivers:

  – **IBM z/VM:** `missing`

- **Flat networking Status: choice(networking.topology).** Provide network connectivity to guests using a flat topology across all compute nodes. At least one of the networking configurations is mandatory to support in the drivers.

  drivers:

  – **IBM z/VM:** `complete`

- **VLAN networking Status: choice(networking.topology).** Provide network connectivity to guests using VLANs to define the topology. At least one of the networking configurations is mandatory to support in the drivers.

  drivers:

  – **IBM z/VM:** `complete`

- **uefi boot Status: optional.** This allows users to boot a guest with uefi firmware.

  drivers:

  – **IBM z/VM:** `missing`

Notes

Using the driver

Planning and Requirements

This dicussed the requirements related to the z/VM system, disk storage etc.

z/VM System Requirements

- A supported version of z/VM 6.4.

- In order to use live migration, the z/VM system must be configured in a Single System Image (SSI) configuration, and must have been created using the IBM-provided installation instructions for SSI configurations.

- The appropriate APARs installed, the current list of which can be found at z/VM xCAT Maintenance (http://www.vm.ibm.com/sysman/xcmntlvl.html) and z/VM OpenStack Cloud Information (http://www.vm.ibm.com/sysman/osmntlvl.html).

Note: IBM z Systems hardware requirements are based on both the applications and the load on the system. Please consult your IBM Sales Representative or Business Partner for assistance in determining the specific hardware requirements for your environment.

Configuration

The following is a sample `nova_zvm.conf` configuration file for the nova-zvm driver, for adaptation and use.

It is auto-generated from the nova-zvm-virt-driver project when this documentation is built, so if you are having issues with an option, please compare your version of the `nova-zvm-virt-driver` Python package with the version of this documentation.

The sample configuration can also be viewed in file form.

Creating zVM Images

Image guide

This guideline will describe the requirements, steps to make images, configurations about the image that can be used in z/VM solution.

Image Requirements

- supported Linux distribution (for deploy). The following are supported: RHEL 6.2, 6.3, 6.4, 6.5, 6.6, and 6.7 RHEL 7.0, 7.1 and 7.2 SLES 11.2, 11.3, and 11.4 SLES 12 and SLES 12.1 Ubuntu 16.04

- A supported root disk type for snapshot/spawn. The following are supported: FBA ECKD

- Images created with the previous version of the OpenStack support should be recaptured with an updated xcat-conf4z installed, version 2.0 or later. see below chapter for more detail on xcatconf4z.

- An image deployed on a compute node must match the disk type supported by that compute node, as configured by the zvm_diskpool_type property in the nova.conf configuration file. A compute node supports deployment on either an ECKD or FBA image, but not both at the same time. If you wish to switch image types, you need to change the zvm_diskpool_type and zvm_diskpool properties in the nova.conf file, accordingly. Then restart the nova-compute service to make the changes take effect.

- If you deploy an instance with an ephemeral disk, both the root disk and the ephemeral disk will be created with the disk type that was specified by zvm_diskpool_type property in the nova.conf file. That property can specify either ECKD or FBA.

- When resizing, remember that you can only resize an instance to the same type of disk. For example, if an instance is built on an FBA type disk, you can resize it to a larger FBA disk, but not to an ECKD disk.

- For glance image-create, it is strongly suggested that you capture an instance with a root disk size no greater than 5GB. If you really want to capture a larger root device, you will need to logon xCAT MN and modify the timeout value in for httpd service to make image-create work as expected.

- For nova boot, it is recommended that you deploy an instance with a root disk size no greater than 5GB. If you really want to deploy a larger root device, you will need to logon xCAT MN and modify the timeout value in for httpd service to make boot work as expected.

- For nova resize operation, we suggest that you resize an instance with a root disk size no greater than 5GB.

- The network interfaces must be IPv4 interfaces.

- Image names should be restricted to the UTF-8 subset, which corresponds to the ASCII character set. In addition, special characters such as /, , $, %, @ should not be used. For the FBA disk type "vm", capture and deploy is supported only for an FBA disk with a single partition. Capture and deploy is not supported for the FBA disk type "vm" on a CMS formatted FBA disk.

- The virtual server/Linux instance used as the source of the new image should meet the following criteria: 1. The root filesystem must not be on a logical volume. 2. The minidisk on which the root filesystem resides should be a minidisk of the same type as

    desired for a subsequent deploy (for example, an ECKD disk image should be captured for a subsequent deploy to an ECKD disk),

    3. not be a full-pack minidisk, since cylinder 0 on full-pack minidisks is reserved, and be defined with virtual address 0100.

    4. The root disk should have a single partition.

    5. The image being captured should support SSH access using keys instead of specifying a password. The subsequent steps to capture the image will perform a key exchange to allow xCAT to access the server.

    6. The image being captured should not have any network interface cards (NICs) defined below virtual address 1100.

In addition to the specified criteria, the following recommendations allow for efficient use of the image:

- The minidisk on which the root filesystem resides should be defined as a multiple of full gigabytes in

size (for example, 1GB or 2GB). OpenStack specifies disk sizes in full gigabyte values, whereas z/VM handles disk sizes in other ways (cylinders for ECKD disks, blocks for FBA disks, and so on). See the appropriate online information if you need to convert cylinders or blocks to gigabytes; for example: http://www.mvsforums.com/helpboards/viewtopic.php?t=8316.

- During subsequent deploys of the image, the OpenStack code will ensure that a disk image is not

copied to a disk smaller than the source disk, as this would result in loss of data. The disk specified in the flavor should therefore be equal to or slightly larger than the source virtual machine's root disk. IBM recommends specifying the disk size as 0 in the flavor, which will cause the virtual machine to be created with the same disk size as the source disk.

Active Engine Guide

Active engine is used as initial configuration and management tool during deployed machine startup. Currently z/VM driver use xcatconf4z and cloud-init as 2 stage active engine.

Installation and Configuration of xcatconf4z

xCAT supports initiating changes to a Linux on z Systems virtual machine while Linux is shut down or the virtual machine is logged off. The changes to Linux are implemented using an activation engine (AE) that is run when Linux is booted the next time. The xCAT activation engine, xcatconf4z, handles changes initiated by xCAT. The script/service must be installed in the Linux on z Systems virtual server so it can process change request files transmitted by the

xCAT ZHCP service to the reader of the virtual machine as a class X file. The script is xcatconf4z and is located at /opt/xcat/share/xcat/scripts in the xCAT MN machine.

The xcatconf4z script should be installed in a machine that can be managed while it is logged off. This includes a Linux on z Systems that will be captured for netboot or sysclone deploys.

Note: An additional activation engine, cloud-init, should be installed to handle OpenStack related tailoring of the system. The cloud-init AE relies on tailoring performed by the xCAT AE, xcatconf4z.

Installation and Configuration of cloud-init

OpenStack uses cloud-init as its activation engine. Some distributions include cloud-init either already installed or available to be installed. If your distribution does not include cloud-init, you can download the code from https://launchpad.net/cloud-init/+download. After installation, if you issue the following shell command and no errors occur, cloud-init is installed correctly:

```
cloud-init init --local
```

Installation and configuration of cloud-init differs among different Linux distributions, and cloud-init source code may change. This section provides general information, but you may have to tailor cloud-init to meet the needs of your Linux distribution. You can find a community-maintained list of dependencies at http://ibm.biz/cloudinitLoZ.

The z/VM OpenStack support has been tested with cloud-init 0.7.4 and 0.7.5 for RHEL6.x and SLES11.x, 0.7.6 for RHEL7.x and SLES12.x, and 0.7.8 for Ubuntu 16.04. If you are using a different version of cloud-init, you should change your specification of the indicated commands accordingly.

During cloud-init installation, some dependency packages may be required. You can use zypper and python setuptools to easily resolve these dependencies. See https://pypi.python.org/pypi/setuptools for more information.

Contributing to the project

Contributing

If you would like to contribute to the development of the nova-zvm project, you must follow the rules for OpenStack contributions described in the "If you're a developer, start here" section of this page:

> http://wiki.openstack.org/HowToContribute

Once those steps have been completed, changes to the nova-zvm project should be submitted for review via the Gerrit tool, following the workflow documented at:

> http://wiki.openstack.org/GerritWorkflow

Pull requests submitted through GitHub will be ignored.

The Git repository for the nova-zvm project is here:

> http://git.openstack.org/cgit/openstack/nova-zvm-virt-driver

Bugs against the nova-zvm project should be filed on Launchpad (not on GitHub):

> https://bugs.launchpad.net/nova-zvm-virt-driver

Pending changes for the nova-zvm project can be seen on its Gerrit page:

> https://review.openstack.org/#/q/project:openstack/nova-zvm-virt-driver

Links

- Source: http://git.openstack.org/cgit/openstack/nova-zvm-virt-driver
- Github shadow: https://github.com/openstack/nova-zvm-virt-driver
- Bugs: http://bugs.launchpad.net/nova-zvm-virt-driver
- Gerrit: https://review.openstack.org/#/q/project:openstack/nova-zvm-virt-driver