
MCVirt Documentation

Release 4.0.1

I.T. Dev Ltd

Feb 16, 2019

Contents

1	MCVirt - Managed Consistent Virtualisation	1
2	Description	3
3	MASTER BRANCH WARNING	5
4	Getting started	7
5	LICENSE	9
6	COPYRIGHT	11
7	API Documentation	25
8	Indices and tables	35

CHAPTER 1

MCVirt - Managed Consistent Virtualisation

MCVirt (`em-see-vert`) - Command line virtual machine management utility.

CHAPTER 2

Description

MCVirt is a utility for managing virtual machines, supporting the following technologies:

- [Ubuntu 14.04 LTS](#).
- [KVM virtualisation](#).
- Clustering with optional [DRBD](#) support.

MCVirt is implemented in Python, using the [libvirt virtualisation library](#).

MASTER BRANCH WARNING

The master branch generally continues the current development of an upcoming release.

Use the releases to obtain the latest stable release

CHAPTER 4

Getting started

MCVirt must currently be built from source into a deb package, using the build script. The package and dependencies can then be installed:

```
$ ./build.sh
$ sudo dpkg -i mcvirt_*_all.deb
$ sudo apt-get -f install
```

See the [installation guide](#) for other dependencies and system configuration.

Start the MCVirt nameserver and daemon by running:

```
$ sudo service mcvirt-ns start
$ sudo service mcvirt-d start
```

Most commands require a username and password to the MCVirt daemon. During installation a superuser is created with username `mjc` and password `pass` - see [permissions](#) for information on creating new users.

Configure the volume group that MCVirt will use to store virtual machine data:

```
$ mcvirt node --set-vm-vg <Volume Group>
```

See the [configuration guide](#) for further node configuration steps.

Create a VM:

```
$ mcvirt create --cpu-count 1 --memory 512 --disk-size 8000 test-vm
```

See the [create/remove VMs](#), [cluster](#), [permissions](#) and [modifying VMs](#) guides for further administrative instructions.

Start the VM:

```
$ mcvirt start test-vm
```

See the [controlling VMs guide](#) for further user instructions.

Note: Username and password can be provided on the command line with the `--username` and `--password` options to instead of typing them in after every command.

For information on developing on MCVirt, see the [development documentation](#).

See the [guide index](#) for a full list of manuals

For more information, please feel free to [contact us](#)

CHAPTER 5

LICENSE

MCVirt is licensed under GPL v2. For more information, please see [LICENSE](#)

Copyright © 2015 - I.T. Dev Ltd

6.1 Clustering

Nodes running MCVirt can be joined together in a cluster - this allows the synchronization of VM/global configurations.

6.1.1 Viewing the status of a cluster

To view the status of the cluster, run the following on an MCVirt node:

```
mcvirt info
```

This will show the cluster nodes, IP addresses, and status.

6.1.2 Adding a new node

It is best to join a blank node (containing a default configuration without any VMs) to a cluster.

When a new node is connected to a cluster, the configuration from the present nodes in the cluster (e.g. users, permissions, networks etc.) are pushed to the new node and any existing configuration is replaced.

Note: Always run the `mcvirt cluster add` command from the source machine, containing VMs, connecting to a remote node that is blank.

The new node must be configured on separate network/VLAN for MCVirt cluster communication.

The IP address that MCVirt clustering/DRBD communications will be performed over must be configured by performing the following on both nodes:

```
mcvirt node --set-ip-address <Node cluster IP address>
```

This configuration can be retrieved by running `mcvirt info`.

Joining the node to the cluster

Note: The following can only be performed by a superuser.

1. From the remote node, run:

```
mcvirt cluster get-connect-string
```

The connect string will be displayed

2. From the source node, run:

```
mcvirt cluster add-node --connect-string <connect string>
```

where `<connect string>` is the string printed out in step 1.

3. The local node will connect to the remote node, ensure it is suitable as a remote node, setup authentication between the nodes and copy the local permissions/network/virtual machine configurations to the remote node.

Note: All existing data on the remote node will be removed.

6.1.3 Removing a node from the cluster

Note: The following can only be performed by a superuser.

To the remove a node from the cluster, run:

```
mcvirt cluster remove-node --node <Remote Node Name>
```

6.1.4 Get Cluster information

- In order to view status information about the cluster, use the 'info' parameter for MCVirt, without specifying a VM name:

```
mcvirt info
```

6.1.5 Virtual machine migration

- VMs that use DRBD-based storage can be migrated to the other node in the cluster, whilst the VM is powered off, using:

```
mcvirt migrate --node <Destination node> <VM Name>
```

- Additional parameters are available to aid the migration and minimise downtime:
 - `--wait-for-shutdown`, which will cause the migration command to poll the running state of the VM and migrate once the VM is in a powered off state, allowing the user to shutdown the VM from within the guest operating system.
 - `--start-after-migration`, which starts the VM immediately after the migration has finished

- `--online`, which will perform online migration. Note: these cannot be used with either of the previous arguments.

6.2 DRBD

DRBD is used by MCVirt to use replicate storage across a 2-node cluster.

Once DRBD is configured and the node is in a cluster, ‘DRBD’ can be specified as the storage type when creating a VM, which allows the VM to be migrated between nodes.

6.2.1 Configuring DRBD

1. Ensure the package `drbd8-utils` is installed on both of the nodes in the cluster
2. DRBD data will be transmitted over the ‘cluster’ address. Ensure that this has been set and that the network is segmented from other network traffic (e.g. by using VLANs).
3. Perform the following MCVirt command to configure DRBD:

```
mcvirt drbd enable
```

6.2.2 DRBD verification

MCVirt has the ability to start/monitor DRBD verifications (See the [DRBD documentation](#)).

The verification can be performed by using:

```
mcvirt verify <--all>|<VM Name>
```

This will perform a verification of the specified VM (or all of the DRBD-backed VMs, if ‘--all’ is specified). Once the verification is complete, an exception is thrown if any of the verifications fail.

The status of the latest verification is captured and will stop users from starting/migrating the VM.

If the verification fails:

- The DRBD volume can be resynced using `resync`:

```
mcvirt resync --source-node=<Node>|--auto-determine <VM Name>
```

6.3 Troubleshooting

6.3.1 Failures during VM creation/deletion

When a VM is created, the following order is performed:

1. The VM is created, configured with the name, memory allocation and number of CPU cores
2. The VM is then created on the remote node
3. The VM is then registered with LibVirt on the local node
4. The hard drive for the VM is created. (For DRBD-backed storage, the storage is created on both nodes and synced)

5. Any network adapters are added to the VM

If a failure of occurs during steps 4/5, the VM will still exist after the failure. The user should be able to see the VM, using `mcvirt list`.

The user can re-create the disks/network adapters as necessary, using the `mcvirt update` command, using `mcvirt info <VM Name>` to monitor the virtual hardware that is attached to the VM.

6.4 Configuration

6.4.1 Configure Network

Remove default network

- By default, libvirt configures a default network, 'default'.
- The 'default' network is attached to a private network, which provides NAT routing through the node's physical interfaces.
- If you wish to use bridging, the default network can be removed by performing the following:

```
mcvirt network delete default
```

Creating/Removing Networks

- Networks provide bridges between physical/bridge interfaces and virtual machines.
- To create a bridge network on the node, an additional network interface will need to be created on the node
- This will generally be placed in `/etc/network/interfaces`

The following example should help with creating this interface:

```
auto vmbr0
iface vmbr0 inet manual
    bridge_ports <Physical interface>
    bridge_stp off
    bridge_fd 0
```

Where *<Physical interface>* is the name of the interface that the bridge should be bridged with, e.g. 'eth0'

- To create a network on the node, perform the following as a superuser:

```
mcvirt network create <Network name> --interface <Bridge interface>
```

- Assuming that there are not any VMs connected to a network, they can be removed using:

```
mcvirt network delete <Network name>
```

6.4.2 Configure MCVirt

- The first time MCVirt is run, it creates a configuration file for itself, found in `/var/lib/mcvirt/<Hostname>/config.json`.
- The volume group, in which VM data will be stored as logical volumes, must be setup using:

```
mcvirt node --set-vm-vg <Volume Group>
```

- The cluster IP address must be configured if the node will be used in a cluster (See the [Cluster documentation](#)):

```
mcvirt node --set-ip-address <Cluster IP Address>
```

- In order for the MCVirt client to connect to the daemon, the hosts file at `/etc/hosts` must be edited by changing the line:

```
127.0.0.1    <hostname>
```

to:

```
<Cluster IP Address>    <hostname>
```

6.5 Controlling VMs

All commands must be performed on the MCVirt node, which can be accessed via SSH.

6.5.1 Start VM

- Use the MCVirt utility to start VMs:

```
mcvirt start <VM name>
```

6.5.2 Stop VM

- Use the MCVirt utility to stop VMs:

```
mcvirt stop <VM name>
```

6.5.3 Reset VM

- Use the MCVirt utility to reset VMs:

```
mcvirt reset <VM Name>
```

- Only a super user can reset a VM. Normal users can stop and start the VM.

For information about attaching ISOs, see [Modifying VMs](#)

6.5.4 Get VM information

- In order to view information about a VM, use the ‘info’ parameter for MCVirt:

```
mcvirt info <VM Name>
```

- Example output:

```
<Username>@node:~# mcvirt info test-vm
Name | test-vm
CPU Cores | 1
Memory Allocation | 512MB
State | Running
ISO location | /var/lib/mcvirt/iso/ubuntu-12.04-server-amd64.iso
-- Disk ID -- | -- Disk Size --
1 | 8GB
-- MAC Address -- | -- Network --
52:54:00:2b:8a:a1 | Production
-- Group -- | -- Users --
owner | mc
user | nd
```

6.5.5 Listing virtual machines

- In order to list the virtual machines on a node, run the following:

```
mcvirt list
```

- This will provide the names of the virtual machines and their current state (running/stopped)

6.5.6 Connect to VNC

- By default, VMs are started with a VNC console, for which the port is automatically generated.
- The default listening IP address is 127.0.0.1, meaning that it can only be accessed from the node itself.
- To manually gain access to a VNC console, ssh to the node, forwarding the port:

1. Determine the port that the VM is listening on:

```
mcvirt info <VM Name> --vnc-port
5904
```

2. SSH onto the node, forwarding the port provided in the previous step (5904 in this case)

- The local port can be any available port. In this example, 1232 is used:

```
ssh <Username>@<Node> -L 1232:127.0.0.1:5904
```

- For putty, use the tunnels configuration under **Connection -> SSH -> Tunnels**, where the source port is the local port and the destination is 127.0.0.1:<VNC Port>

3. Use an VNC client to connect to 127.0.0.1:1232 on your local PC

6.5.7 Removing VNC display

- By disabling the VNC display, a greater VM performance may be achieved.
- Power off the VM
- Perform:

```
virsh edit <VM Name>
```

- Remove the `<graphics type='vnc'...>...</graphics>` lines from the configuration.
- Save the configuration and start the VM
- This can only be performed by root

6.5.8 Monitoring Resources

- To monitor resources, the following commands are available that can be run from an SSH console:
 - `top` - monitor CPU/memory usages by processes
 - `iftop` - monitor network usage
 - `iostat` - monitor disk usages

6.5.9 Back up VM

MCVirt can provide access to snapshots of the raw volumes of VM disks, allowing a superuser to backup the data

1. To create a snapshot, perform the following:

```
mcvirt backup create-snapshot --disk-id <Disk ID> <VM Name>
```

2. The returned path provides access to the disk at the time that the snapshot was created

Warning: The snapshot is 500MB in size, meaning that once the VM has changed 500MB of space on the disk, the VM will no longer be able to write to its disk

3. Once the data has been backed up, the snapshot can be removed by performing:

```
mcvirt backup delete-snapshot --disk-id <Disk ID> <VM Name>
```

- This can only be performed by a superuser

6.6 Create/Remove VMs

- All commands must be performed on the MCVirt node, which can be accessed via SSH using LDAP credentials.
- You must be a superuser to create and remove VMs

6.6.1 Create VM

- Use the MCVirt utility to create VMs:

```
mcvirt create '<VM Name>'
```

- The following parameters are available:
 - **–memory** - Amount of memory to allocate to the VM (MB) (required)
 - **–cpu-count** - Number of vCPUs to be allocated to the VM (required)
 - **–disk-size** - Size of initial disk to be added to the VM (MB) (optional)
 - **–network** - Provide the name of a network to be attached to the VM. (optional)
 - * This can be called as multiple times.

- * A separate network interface is added to the VM for each network.
- * A network can be specified multiple times to create multiple adapters connected to the same network.
- **–storage-type** - Storage backing type - either `Local` or `DRBD`.
- **–nodes** - Specifies the nodes that the VM will be hosted on, if a `DRBD` storage-type is specified and there are more than 2 nodes in the cluster.
- **–driver** - The virtual disk driver to use. If this is not specified then MCVirt will select the most appropriate driver (optional)

6.6.2 Cloning a VM

Cloning/duplicating a VM will create an identical replica of the VM.

Although both cloning and duplicating initially may appear to provide the same functionality, there are core differences, based on how they work, which should be noted to decide which function to use.

Both cloning and duplicating a VM can be performed by an **owner** of a VM.

6.6.3 Cloning

- The hard disk for the VM is **snapshotted**, which means the VM is cloned very quickly
- Cloning VMs is not support for `DRBD`-backed VMs
- Some restrictions are imposed on both the parent and clone, due to the way that the storage is cloned:
 - Parent VMs cannot be:
 - * Started
 - * Resize (HDDs)
 - * Deleted
 - VM Clones cannot be:
 - * Resized
 - * Cloned
 - **Note:** All restrictions are lifted once all VM clones have been removed.

A VM can be cloned by performing the following:

```
mcvirt clone --template <Source VM Name> <Target VM Name>
```

6.6.4 Duplicating

- Duplicating produces a new VM that is a completely separate entity to the source, meaning that no restrictions are imposed on either VM
- Duplicating a VM will copy the entire VM hard drive, which takes longer than cloning a VM

A VM can be duplicated by performing the following:

```
mcvirt duplicate --template <Source VM Name> <Target VM Name>
```

6.6.5 Removing VM

- Ensure that the VM is stopped.
- Use the MCVirt utility to remove the VM:

```
mcvirt delete <VM Name>
```

- Without any parameters, the VM will simply be ‘unregistered’ from the node.
- To delete all data associated with the VM, supply the parameter **-delete-data**
- Only a superuser can delete a VM

6.7 Development

6.7.1 Coding Standards

The MCVirt code base follows the [python PEP 8 coding standards](#), with a line length limit of 100 characters.

All code changes must comply with this coding standard and are checked by continuous integration.

The PEP 8 code checker can be installed using:

```
sudo apt-get install pep8
```

Run the checks using:

```
pep8
```

6.7.2 Automated Tests

There is a collection of unit tests for MCVirt, which can be run as follows:

```
python /usr/lib/mcvirt/test/run_tests.py
```

Before running the tests ensure that the `mcvirt-ns` service is running on all nodes in the cluster, and that `mcvirtd` is running on all nodes except the one the tests are being run on (since `mcvirtd` is started when the tests are run)

6.7.3 Manual Test Procedure

This test procedure is designed to compliment the automated unit tests and should be performed prior to making a new release.

- Make sure the `mcvirt-ns` and `mcvirtd` daemons are started
- Create a VM called ‘test-vm’
- Run `mcvirt list` and check that ‘test-vm’ is shown in the list, and that its state is ‘STOPPED’
 - If the node is part of a cluster, run `mcvirt list` on another node in the cluster, and check that ‘test-vm’ is listed
- Start ‘test-vm’. Run `mcvirt list` again and check that its state is now ‘RUNNING’
 - Run `mcvirt list` on a remote node to check the state of ‘test-vm’ if the node is part of a cluster

- Try to delete ‘test-vm’ and check that an error is shown saying ‘Can’t delete a running VM’
- Stop ‘test-vm’, and try to delete it again. Check that it is no longer shown in the output of `mcvirt list`
 - If the node is part of a cluster, confirm ‘test-vm’ has been deleted on a remote node too
- If the node is part of a cluster:
 - Make sure DRBD is enabled by running `mcvirt drbd enable`
 - Create a new VM called ‘cluster-vm’, specifying the storage type as ‘Drbd’
 - Start ‘cluster-vm’
 - Test online migration of VMs by running `mcvirt migrate --online --node <remote node> cluster-vm`
 - Run `mcvirt list` on the local and remote nodes to check that ‘cluster-vm’ is now registered on the remote node

6.8 Installation

6.8.1 Install Operating System

- MCVirt is currently built to support Ubuntu 14.04 with native versions of dependencies.
- When installing the operating system, create the following logical volumes:
 - Root - Create a 50GB partition using ext4. This is used for the operating system, MCVirt configurations and ISO images
 - SWAP - leave the suggested SWAP volume unaltered
- Virtual machine storage will be created as additional volumes in the volume group.

6.8.2 Building the package

- Ensure the build dependencies are installed: `dpkg`, `python-docutils`
- Clone the repository with: `git clone https://github.com/ITDevLtd/MCVirt`
- From within the root of the working copy, run `build.sh`

6.8.3 Installing Package

To install the package, run:

```
$ sudo dpkg -i mcvirt_X.XX_all.deb
$ sudo apt-get -f install
```

MCVirt uses a customised version of `Pyro`, which can be installed by running:

```
$ git clone https://github.com/MatthewJohn/Pyro4
$ cd Pyro4
$ sudo pip install .
```

You may need to install `pip` by running `sudo apt-get install python-pip`.

6.9 User Guides

6.9.1 Installation

- [Installation](#) - Procedure for building the MCVirt package, setting up a node with MCVirt and performing initial configuration

6.9.2 Configuration

- [Configuration](#) - Procedures for performing initial configurations of an MCVirt installation

6.9.3 Administration

- [Permissions](#) - Procedures for configuring permissions within MCVirt
- [Cluster](#) - Procedures for configuring an MCVirt Cluster

6.9.4 Usage

- [Create/Remove VMs](#) - Procedures for creating and deleting virtual machines
- [Controlling VMs](#) - Instructions for using the MCVirt script and controlling virtual machines
- [Modifying VMs](#) - Procedures for making changes to virtual machine configurations

6.9.5 Development

- [Development](#) - Information about performing development on MCVirt

6.10 Modifying VMs

6.10.1 Attaching ISO

- ISO images can be attached to the cdrom drive of a VM whilst booting the VM
- Use the MCVirt utility to start the VM, using the ‘-iso’ parameter to define the ISO image to be attached to the VM:

```
mcvirt start <VM Name> --iso <Name of ISO file>
```

6.10.2 ISO Management

ISOs are stored on a per-node basis.

The ISOs present on the local node can be determined by:

```
mcvirt iso list
```

ISOs can be added using either of the following:

```
mcpvrt iso add --from-url http://example.com/example.iso
mcpvrt iso add --from-path ./local.iso
```

6.10.3 Increase Disk Size

- Power off the VM
- Use MCVirt to increase the size of the disk - you will need to find the disk ID, which can be found by looking at the VM configuration (in most cases where a VM has one disk attached to it, it should be 1):

```
mcpvrt update --increase-disk <Amount to increase (MB)> --disk-id <Disk Id> <VM_
↪Name>
```

6.10.4 Change Memory/CPU Allocation

- Update the VM memory allocation and virtual CPU count using the following:

```
mcpvrt update --memory <New Memory Allocation (MB)> <VM Name>
mcpvrt update --cpu-count <New CPU count> <VM Name>
```

- The changes will take affect the next time the VM is booted. If the VM is running, it will need to be powered off and started again.

6.10.5 Add Additional Disk

- Use the following MCVirt command to add an additional disk to a VM:

```
mcpvrt update --add-disk <Size of disk (MB)> <VM Name>
```

- The device will be attached to the VM the next time it's booted. If the VM is running, it will need to be powered off and started again.

6.10.6 Add/Remove Network Adapter

- Use the following MCVirt command to add/remove network adapters to/from a VM
- Add an adapter:

```
mcpvrt update --add-network <Network Name> <VM Name>
```

- Remove an adapter:

```
mcpvrt update --remove-network '<NIC MAC Address>' <VM Name>
```

- Use the formatting '00:11:22:33:44:55' for the MAC address
- The device will altered the next time the VM is booted. If the VM is running, it will need to be powered off and started again.

VM Locking

VMs can be locked by superusers, which stops them from being started, stopped or migrated

- To lock a VM:

```
mcvirt lock --lock <VM Name>
```

- To unlock a VM:

```
mcvirt lock --unlock <VM Name>
```

- Users can check the lock status of a VM by running:

```
mcvirt lock --check-lock <VM Name>
```

6.11 Permissions

6.11.1 Superusers

- To run MCVirt commands as a superuser you must either:
 - Have your username included in the superusers section in the configuration file.
- Superusers can be added/removed using the following:

```
mcvirt permission --add-superuser=<username>
mcvirt permission --delete-superuser=<username>
```

6.11.2 Managing users

- To create a new user, perform the following as a superuser:

```
mcvirt user create <new username>
```

The password for the new user can be provided interactively, passed on the command line with `--user-password <new password>`, or generated automatically with `--generate-password`. The generated password will be displayed when the user is created.

- To delete a user, perform the following as a superuser:

```
mcvirt user delete <user>
```

- To change your password, perform the following:

```
mcvirt user change-password
```

The new password can be provided interactively or on the command line with `--new-password <new password>`. **Note:** Superusers can change the password of any other user by running `mcvirt user change-password --target-user <other user>`.

- In MCVirt, ‘users’ are able to start/stop VMs
- To view the current permissions on a VM, including users and owners of a VM, run:

```
mcvirt info <VM Name>
```

- To add a user to VM, perform the following:

```
mcvirt permission --add-user <Username> <VM Name>
```

- To remove a user, perform the following:

```
mcvirt permission --delete-user <Username> <VM Name>
```

- **Owners** of a VM are able to manage the **users** of a VM.

6.11.3 Managing owners

- VM owners have the same permissions as users, except they are also able to manage the users of the VM
- To add an owner to VM, perform the following:

```
mcvirt permission --add-owner <Username> <VM Name>
```

- To remove an owner, perform the following:

```
mcvirt permission --delete-owner <Username> <VM Name>
```

- Only superusers are able to manage the **owners** of a VM.

Contents:

7.1 mcvirt package

7.1.1 Subpackages

`mcvirt.agent` package

Submodules

`mcvirt.agent.daemon` module

`mcvirt.agent.host_connection` module

Module contents

`mcvirt.auth` package

Subpackages

`mcvirt.auth.group` package

Submodules

`mcvirt.auth.group.factory` module

`mcvirt.auth.group.group` module

Module contents

`mcvirt.auth.user_types` package

Submodules

`mcvirt.auth.user_types.cluster_user` module

`mcvirt.auth.user_types.connection_user` module

`mcvirt.auth.user_types.drbd_hook_user` module

`mcvirt.auth.user_types.ldap_user` module

`mcvirt.auth.user_types.local_user` module

`mcvirt.auth.user_types.user_base` module

Module contents

Submodules

`mcvirt.auth.auth` module

`mcvirt.auth.factory` module

`mcvirt.auth.permissions` module

`mcvirt.auth.session` module

Module contents

`mcvirt.client` package

Submodules

`mcvirt.client.rpc` module

Module contents

`mcvirt.cluster` package

Submodules

`mcvirt.cluster.cluster` module

`mcvirt.cluster.remote` module

Module contents

`mcvirt.iso` package

Submodules

`mcvirt.iso.factory` module

`mcvirt.iso.iso` module

Module contents

`mcvirt.node` package

Subpackages

`mcvirt.node.network` package

Submodules

`mcvirt.node.network.factory` module

`mcvirt.node.network.network` module

Module contents

Submodules

`mcvirt.node.drbd` module

`mcvirt.node.ldap_factory` module

`mcvirt.node.libvirt_config` module

`mcvirt.node.node` module

Module contents

`mcvirt.parser_modules` package

Subpackages

`mcvirt.parser_modules.virtual_machine` package

Submodules

`mcvirt.parser_modules.virtual_machine.backup_parser` module

`mcvirt.parser_modules.virtual_machine.clone_parser` module

`mcvirt.parser_modules.virtual_machine.create_parser` module

`mcvirt.parser_modules.virtual_machine.delete_parser` module

`mcvirt.parser_modules.virtual_machine.duplicate_parser` module

`mcvirt.parser_modules.virtual_machine.info_parser` module

`mcvirt.parser_modules.virtual_machine.list_parser` module

`mcvirt.parser_modules.virtual_machine.lock_parser` module

`mcvirt.parser_modules.virtual_machine.migrate_parser` module

`mcvirt.parser_modules.virtual_machine.move_parser` module

`mcvirt.parser_modules.virtual_machine.register_parser` module

`mcvirt.parser_modules.virtual_machine.reset_parser` module

`mcvirt.parser_modules.virtual_machine.shutdown_parser` module

`mcvirt.parser_modules.virtual_machine.start_parser` module

`mcvirt.parser_modules.virtual_machine.stop_parser` module

`mcvirt.parser_modules.virtual_machine.unregister_parser` module

`mcvirt.parser_modules.virtual_machine.update_parser` module

Module contents

Submodules

`mcvirt.parser_modules.clear_method_lock_parser` module

`mcvirt.parser_modules.cluster_parser` module

`mcvirt.parser_modules.drbd_parser` module

`mcvirt.parser_modules.group_parser` module

`mcvirt.parser_modules.iso_parser` module

`mcvirt.parser_modules.network_parser` module

`mcvirt.parser_modules.node_parser` module

`mcvirt.parser_modules.permission_parser` module

`mcvirt.parser_modules.resync_parser` module

`mcvirt.parser_modules.storage_parser` module

`mcvirt.parser_modules.user_parser` module

`mcvirt.parser_modules.verify_parser` module

`mcvirt.parser_modules.watchdog_parser` module

Module contents

`mcvirt.rpc` package

Submodules

`mcvirt.rpc.certificate_generator` module

`mcvirt.rpc.certificate_generator_factory` module

`mcvirt.rpc.constants` module

`mcvirt.rpc.daemon_lock` module

`mcvirt.rpc.expose_method` module

`mcvirt.rpc.lock` module

`mcvirt.rpc.name_server` module

`mcvirt.rpc.pyro_object` module

`mcvirt.rpc.rpc_daemon` module

`mcvirt.rpc.ssl_socket` module

Module contents

`mcvirt.storage` package

Submodules

`mcvirt.storage.base` module

`mcvirt.storage.factory` module

`mcvirt.storage.file` module

`mcvirt.storage.lvm` module

Module contents

`mcvirt.test` package

Subpackages

`mcvirt.test.lock` package

Submodules

`mcvirt.test.lock.lock_tests` module

Module contents

`mcvirt.test.node` package

Submodules

`mcvirt.test.node.network_tests` module

`mcvirt.test.node.node_tests` module

Module contents

`mcvirt.test.virtual_machine` package

Subpackages

`mcvirt.test.virtual_machine.hard_drive` package

Submodules

`mcvirt.test.virtual_machine.hard_drive.drbd_tests` module

Module contents

Submodules

`mcvirt.test.virtual_machine.online_migrate_tests` module

`mcvirt.test.virtual_machine.virtual_machine_tests` module

Module contents

Submodules

`mcvirt.test.auth_tests` module

`mcvirt.test.ldap_tests` module

`mcvirt.test.run_tests` module

`mcvirt.test.test_base` module

`mcvirt.test.unit_test_bootstrap` module

`mcvirt.test.update_tests` module

`mcvirt.test.validation_tests` module

Module contents

`mcvirt.thread` package

Submodules

`mcvirt.thread.auto_start_watchdog` module

`mcvirt.thread.repeat_timer` module

`mcvirt.thread.watchdog` module

Module contents

`mcvirt.virtual_machine` package

Subpackages

`mcvirt.virtual_machine.hard_drive` package

Submodules

`mcvirt.virtual_machine.hard_drive.base` module

`mcvirt.virtual_machine.hard_drive.drbd` module

`mcvirt.virtual_machine.hard_drive.factory` module

`mcvirt.virtual_machine.hard_drive.local` module

Module contents

`mcvirt.virtual_machine.network_adapter` package

Submodules

`mcvirt.virtual_machine.network_adapter.factory` module

`mcvirt.virtual_machine.network_adapter.network_adapter` module

Module contents

Submodules

`mcvirt.virtual_machine.agent_connection` module

`mcvirt.virtual_machine.disk_drive` module

`mcvirt.virtual_machine.factory` module

`mcvirt.virtual_machine.usb_device` module

`mcvirt.virtual_machine.virtual_machine` module

`mcvirt.virtual_machine.virtual_machine_config` module

Module contents

7.1.2 Submodules

7.1.3 mcvirt.argument_validator module

7.1.4 mcvirt.bash-complete module

7.1.5 mcvirt.build_man module

7.1.6 mcvirt.config_file module

7.1.7 mcvirt.constants module

7.1.8 mcvirt.exceptions module

7.1.9 mcvirt.libvirt_connector module

7.1.10 mcvirt.logger module

7.1.11 mcvirt.mcvirt-drbd module

7.1.12 mcvirt.mcvirt_config module

7.1.13 mcvirt.parser module

7.1.14 mcvirt.syslogger module

7.1.15 mcvirt.system module

7.1.16 mcvirt.utils module

7.1.17 mcvirt.version module

7.1.18 Module contents

CHAPTER 8

Indices and tables

- `genindex`
- `modindex`
- `search`