NAME

virsh − management user interface

SYNOPSIS

virsh [*OPTION*]... [*COMMAND_STRING*]

virsh [*OPTION*]... *COMMAND* [*ARG*]...

DESCRIPTION

The **virsh** program is the main interface for managing virsh guest domains. The program can be used to create, pause, and shutdown domains. It can also be used to list current domains. Libvirt is a C toolkit to interact with the virtualization capabilities of recent versions of Linux (and other OSes). It is free software available under the GNU Lesser General Public License. Virtualization of the Linux Operating System means the ability to run multiple instances of Operating Systems concurrently on a single hardware system where the basic resources are driven by a Linux instance. The library aims at providing a long term stable C API. It currently supports Xen, QEMU, KVM, LXC, OpenVZ, VirtualBox and VMware ESX.

The basic structure of most virsh usage is:

```
virsh [OPTION]... <command> <domain> [ARG]...
```

Where *command* is one of the commands listed below; *domain* is the numeric domain id, or the domain name, or the domain UUID; and *ARGS* are command specific options. There are a few exceptions to this rule in the cases where the command in question acts on all domains, the entire machine, or directly on the xen hypervisor. Those exceptions will be clear for each of those commands. Note: it is permissible to give numeric names to domains, however, doing so will result in a domain that can only be identified by domain id. In other words, if a numeric value is supplied it will be interpreted as a domain id, not as a name. Any *command* starting with **#** is treated as a comment and silently ignored, all other unrecognized *commands* are diagnosed.

The **virsh** program can be used either to run one *COMMAND* by giving the command and its arguments on the shell command line, or a *COMMAND_STRING* which is a single shell argument consisting of multiple *COMMAND* actions and their arguments joined with whitespace and separated by semicolons or newlines between commands, where unquoted backslash−newline pairs are elided. Within *COMMAND_STRING*, virsh understands the same single, double, and backslash escapes as the shell, although you must add another layer of shell escaping in creating the single shell argument, and any word starting with unquoted **#** begins a comment that ends at newline. If no command is given in the command line, **virsh** will then start a minimal interpreter waiting for your commands, and the **quit** command will then exit the program.

The **virsh** program understands the following *OPTIONS*.

−c, **−−connect** *URI*

Connect to the specified *URI*, as if by the **connect** command, instead of the default connection.

−d, **−−debug** *LEVEL*

Enable debug messages at integer *LEVEL* and above. *LEVEL* can range from 0 to 4 (default). See the documentation of **VIRSH_DEBUG** environment variable below for the description of each *LEVEL*.

- **−e**, **−−escape** *string*

Set alternative escape sequence for *console* command. By default, telnet's **^]** is used. Allowed characters when using hat notation are: alphabetic character, @, [,], , ^, _.

- **−h**, **−−help**

Ignore all other arguments, and behave as if the **help** command were given instead.

- **−k**, **−−keepalive−interval** *INTERVAL*

Set an *INTERVAL* (in seconds) for sending keepalive messages to check whether connection to the server is still alive. Setting the interval to 0 disables client keepalive mechanism.

- **−K**, **−−keepalive−count** *COUNT*

Set a number of times keepalive message can be sent without getting an answer from the server without marking the connection dead. There is no effect to this setting in case the *INTERVAL* is set to 0.

- **−l**, **−−log** *FILE*

Output logging details to *FILE*.

- **−q**, **−−quiet**

Avoid extra informational messages.

- **−r**, **−−readonly**

Make the initial connection read−only, as if by the *−−readonly* option of the **connect** command.

- **−t**, **−−timing**

Output elapsed time information for each command.

- **−v**, **−−version[=short]**

Ignore all other arguments, and prints the version of the libvirt library virsh is coming from

- **−V**, **−−version=long**

Ignore all other arguments, and prints the version of the libvirt library virsh is coming from and which options and driver are compiled in.

NOTES

Most **virsh** operations rely upon the libvirt library being able to connect to an already running libvirtd service. This can usually be done using the command**ser vice libvirtd start**.

Most **virsh** commands require root privileges to run due to the communications channels used to talk to the hypervisor. Running as non root will return an error.

Most **virsh** commands act synchronously, except maybe shutdown, setvcpus and setmem. In those cases the fact that the **virsh** program returned, may not mean the action is complete and you must poll periodically to detect that the guest completed the operation.

virsh strives for backward compatibility. Although the**help** command only lists the preferred usage of a command, if an older version of **virsh** supported an alternate spelling of a command or option (such as *−−tunnelled* instead of *−−tunneled*), then scripts using that older spelling will continue to work.

Several **virsh** commands take an optionally scaled integer; if no scale is provided, then the default is listed in the command (for historical reasons, some commands default to bytes, while other commands default to kibibytes). The following case−insensitive suffixes can be used to select a specific scale:

```
b, byte  byte      1
KB       kilobyte  1,000
k, KiB   kibibyte  1,024
MB       megabyte  1,000,000
```

```
M, MiB   mebibyte  1,048,576
GB       gigabyte  1,000,000,000
G, GiB   gibibyte  1,073,741,824
TB       terabyte  1,000,000,000,000
T, TiB   tebibyte  1,099,511,627,776
PB       petabyte  1,000,000,000,000,000
P, PiB   pebibyte  1,125,899,906,842,624
EB       exabyte   1,000,000,000,000,000,000
E, EiB   exbibyte  1,152,921,504,606,846,976
```

GENERIC COMMANDS

The following commands are generic i.e. not specific to a domain.

help

Syntax:

```
help [command-or-group]
```

This lists each of the virsh commands. When used without options, all commands are listed, one per line, grouped into related categories, displaying the keyword for each group.

To display only commands for a specific group, give the keyword for that group as an option. For example:

Example 1:

```
virsh # help host

Host and Hypervisor (help keyword 'host'):
    capabilities                   capabilities
    cpu-models                     show the CPU models for an architecture
    connect                        (re)connect to hypervisor
    freecell                       NUMA free memory
    hostname                       print the hypervisor hostname
    qemu-attach                    Attach to existing QEMU process
    qemu-monitor-command           QEMU Monitor Command
    qemu-agent-command             QEMU Guest Agent Command
    sysinfo                        print the hypervisor sysinfo
    uri                            print the hypervisor canonical URI
```

To display detailed information for a specific command, give its name as the option instead. For example:

Example 2:

```
virsh # help list
  NAME
    list - list domains

  SYNOPSIS
    list [--inactive] [--all]

  DESCRIPTION
    Returns list of domains.

  OPTIONS
    --inactive       list inactive domains
    --all            list inactive & active domains
```

quit, exit

Syntax:

```
quit
exit
```

quit this interactive terminal

version

Syntax:

```
version [--daemon]
```

Will print out the major version info about what this built from. If *--daemon* is specified then the version of the libvirt daemon is included in the output.

Example:

```
$ virsh version
Compiled against library: libvirt 1.2.3
Using library: libvirt 1.2.3
Using API: QEMU 1.2.3
Running hypervisor: QEMU 2.0.50

$ virsh version --daemon
Compiled against library: libvirt 1.2.3
Using library: libvirt 1.2.3
Using API: QEMU 1.2.3
Running hypervisor: QEMU 2.0.50
Running against daemon: 1.2.6
```

cd

Syntax:

```
cd [directory]
```

Will change current directory to *directory*. The default directory for the **cd** command is the home directory or, if there is no *HOME* variable in the environment, the root directory.

This command is only available in interactive mode.

pwd

Syntax:

```
pwd
```

Will print the current directory.

connect

Syntax:

```
connect [URI] [--readonly]
```

(Re)−Connect to the hypervisor. When the shell is first started, this is automatically run with the *URI* parameter requested by the **−c** option on the command line. The *URI* parameter specifies how to connect to the hypervisor. The URI docs *https://libvirt.org/uri.html* list the values supported, but the most common are:

- xen:///system

  this is used to connect to the local Xen hypervisor

- qemu:///system

  connect locally as root to the daemon supervising QEMU and KVM domains

- qemu:///session

  connect locally as a normal user to his own set of QEMU and KVM domains

- lxc:///system

  connect to a local linux container

To find the currently used URI, check the *uri* command documented below.

For remote access see the URI docs *https://libvirt.org/uri.html* on how to make URIs. The −−*readonly* option allows for read−only connection

uri

Syntax:

```
uri
```

Prints the hypervisor canonical URI, can be useful in shell mode.

hostname

Syntax:

```
hostname
```

Print the hypervisor hostname.

sysinfo

Syntax:

```
sysinfo
```

Print the XML representation of the hypervisor sysinfo, if available.

nodeinfo

Syntax:

```
nodeinfo
```

Returns basic information about the node, like number and type of CPU, and size of the physical memory. The output corresponds to virNodeInfo structure. Specifically, the "CPU socket(s)" field means number of CPU sockets per NUMA cell. The information libvirt displays is dependent upon what each architecture may provide.

nodecpumap

Syntax:

```
nodecpumap [−−pretty]
```

Displays the node's total number of CPUs, the number of online CPUs and the list of online CPUs.

With *−−pretty* the online CPUs are printed as a range instead of a list.

nodecpustats
Syntax:

```
nodecpustats [cpu] [−−percent]
```

Returns cpu stats of the node. If *cpu* is specified, this will print the specified cpu statistics only. If *−−percent* is specified, this will print the percentage of each kind of cpu statistics during 1 second.

nodememstats
Syntax:

```
nodememstats [cell]
```

Returns memory stats of the node. If *cell* is specified, this will print the specified cell statistics only.

nodesevinfo
Syntax:

```
nodesevinfo
```

Reports information about the AMD SEV launch security features for the node, if any. Some of this information is also reported in the domain capabilities XML document.

nodesuspend
Syntax:

```
nodesuspend [target] [duration]
```

Puts the node (host machine) into a system−wide sleep state and schedule the node's Real−Time−Clock interrupt to resume the node after the time duration specified by *duration* is out. *target* specifies the state to which the host will be suspended to, it can be "mem" (suspend to RAM), "disk" (suspend to disk), or "hybrid" (suspend to both RAM and disk). *dur ation* specifies the time duration in seconds for which the host has to be suspended, it should be at least 60 seconds.

node−memory−tune
Syntax:

```
node−memory−tune [shm−pages−to−scan] [shm−sleep−millisecs] [shm−merge−across−no
```

Allows you to display or set the node memory parameters. *shm−pa ges−to−scan* can be used to set the number of pages to scan before the shared memory service goes to sleep; *shm−sleep−millisecs* can be used to set the number of millisecs the shared memory service should sleep before next scan; *shm−merge−across−nodes* specifies if pages from different numa nodes can be merged. When set to 0, only pages which physically reside in the memory area of same NUMA node can be merged. When set to 1, pages from all nodes can be merged. Default to 1.

Note: Currently the "shared memory service" only means KSM (Kernel Samepage Merging).

capabilities
Syntax:

```
capabilities
```

Print an XML document describing the capabilities of the hypervisor we are currently connected to. This includes a section on the host capabilities in terms of CPU and features, and a set of description for each kind of guest which can be virtualized. For a more complete description see:

https://libvirt.org/formatcaps.html

The XML also show the NUMA topology information if available.

domcapabilities

Syntax:

```
domcapabilities [virttype] [emulatorbin] [arch] [machine]
```

Print an XML document describing the domain capabilities for the hypervisor we are connected to using information either sourced from an existing domain or taken from the **virsh capabilities** output. This may be useful if you intend to create a new domain and are curious if for instance it could make use of VFIO by creating a domain for the hypervisor with a specific emulator and architecture.

Each hypervisor will have different requirements regarding which options are required and which are optional. A hypervisor can support providing a default value for any of the options.

The *virttype* option specifies the virtualization type used. The value to be used is either from the 'type' attribute of the <domain/> top level element from the domain XML or the 'type' attribute found within each <guest/> element from the **virsh capabilities** output. The *emulatorbin* option specifies the path to the emulator. The value to be used is either the <emulator> element in the domain XML or the **virsh capabilities** output. The *arch* option specifies the architecture to be used for the domain. The value to be used is either the "arch" attribute from the domain's XML <os/> element and <type/> subelement or the "name" attribute of an <arch/> element from the **virsh capabililites** output. The *machine* specifies the machine type for the emulator. The value to be used is either the "machine" attribute from the domain's XML <os/> element and <type/> subelement or one from a list of machines from the **virsh capabilities** output for a specific architecture and domain type.

For the QEMU hypervisor, a *virttype* of either 'qemu' or 'kvm' must be supplied along with either the *emulatorbin* or *arch* in order to generate output for the default *machine*. Supplying a*mac hine* value will generate output for the specific machine.

pool−capabilities

Syntax:

```
pool−capabilities
```

Print an XML document describing the storage pool capabilities for the connected storage driver. This may be useful if you intend to create a new storage pool and need to know the available pool types and supported storage pool source and target volume formats as well as the required source elements to create the pool.

inject−nmi

Syntax:

```
inject−nmi domain
```

Inject NMI to the guest.

list

Syntax:

```
list [−−inactive | −−all]
     [−−managed−save] [−−title]
     { [−−table] | −−name | −−uuid | −−id }
     [−−persistent] [−−transient]
     [−−with−managed−save] [−−without−managed−save]
```

```
[--autostart] [--no-autostart]
[--with-snapshot] [--without-snapshot]
[--with-checkpoint] [--without-checkpoint]
[--state-running] [--state-paused]
[--state-shutoff] [--state-other]
```

Prints information about existing domains. If no options are specified it prints out information about running domains.

Example 1:

An example format for the list is as follows:

```
``virsh`` list
  Id    Name                           State
----------------------------------------------------
  0     Domain-0                       running
  2     fedora                         paused
```

Name is the name of the domain. ID the domain numeric id. State is the run state (see below).

STATES

The State field lists what state each domain is currently in. A domain can be in one of the following possible states:

- **running**

  The domain is currently running on a CPU

- **idle**

  The domain is idle, and not running or runnable. This can be caused because the domain is waiting on IO (a traditional wait state) or has gone to sleep because there was nothing else for it to do.

- **paused**

  The domain has been paused, usually occurring through the administrator running **virsh suspend**. When in a paused state the domain will still consume allocated resources like memory, but will not be eligible for scheduling by the hypervisor.

- **in shutdown**

  The domain is in the process of shutting down, i.e. the guest operating system has been notified and should be in the process of stopping its operations gracefully.

- **shut off**

  The domain is not running. Usually this indicates the domain has been shut down completely, or has not been started.

- **crashed**

  The domain has crashed, which is always a violent ending. Usually this state can only occur if the domain has been configured not to restart on crash.

- **pmsuspended**

  The domain has been suspended by guest power management, e.g. entered into s3 state.

Normally only active domains are listed. To list inactive domains specify *--inactive* or *--all* to list both active and inactive domains.

Filtering

To further filter the list of domains you may specify one or more of filtering flags supported by the **list** command. These flags are grouped by function. Specifying one or more flags from a group enables the filter group. Note that some combinations of flags may yield no results. Supported filtering flags and groups:

Persistence

Flag *--persistent* is used to include persistent guests in the returned list. To include transient guests specify *--transient*.

Existence of managed save image

To list domains having a managed save image specify flag *--with-managed-save*. For domains that don't have a managed save image specify *--without-managed-save*.

Domain state

The following filter flags select a domain by its state: *--state-running* for running domains, *--state-paused* for paused domains, *--state-shutoff* for turned off domains and *--state-other* for all other states as a fallback.

Autostarting domains

To list autostarting domains use the flag *--autostart*. To list domains with this feature disabled use *--no-autostart*.

Snapshot existence

Domains that have snapshot images can be listed using flag *--with-snapshot*, domains without a snapshot *--without-snapshot*.

Checkpoint existence

Domains that have checkpoints can be listed using flag *--with-checkpoint*, domains without a checkpoint *--without-checkpoint*.

When talking to older servers, this command is forced to use a series of API calls with an inherent race, where a domain might not be listed or might appear more than once if it changed state between calls while the list was being collected. Newer servers do not have this problem.

If *--managed-save* is specified, then domains that have managed save state (only possible if they are in the **shut off** state, so you need to specify *--inactive* or *--all* to actually list them) will instead show as **saved** in the listing. This flag is usable only with the default *--table* output. Note that this flag does not filter the list of domains.

If *--name* is specified, domain names are printed instead of the table formatted one per line. If *--uuid* is specified domain's UUID's are printed instead of names. If *--id* is specified then domain's ID's are printed indead of names. However, it is possible to combine *--name*, *--uuid* and *--id* to select only desired fields for printing. Flag *--table* specifies that the legacy table-formatted output should be used, but it is mutually exclusive with *--name*, *--uuid* and *--id*. This is the default and will be used if neither of *--name*, *--uuid* or *--id* is specified. If neither *--name* nor *--uuid* is specified, but *--id* is, then only active domains are listed, even with the *--all* parameter as otherwise the output would just contain bunch of lines with just *-1*.

If *--title* is specified, then the short domain description (title) is printed in an extra column. This flag is usable only with the default *--table* output.

Example 2:

```
$ virsh list --title
```

```
 Id    Name        State      Title
----------------------------------------
 0     Domain-0    running    Mailserver 1
 2     fedora      paused
```

freecell

Syntax:

```
freecell [{ [--cellno] cellno | --all }]
```

Prints the available amount of memory on the machine or within a NUMA cell. The freecell command can provide one of three different displays of available memory on the machine depending on the options specified. With no options, it displays the total free memory on the machine. With the --all option, it displays the free memory in each cell and the total free memory on the machine. Finally, with a numeric argument or with --cellno plus a cell number it will display the free memory for the specified cell only.

freepages

Syntax:

```
freepages [{ [--cellno] cellno [--pagesize] pagesize |     --all }]
```

Prints the available amount of pages within a NUMA cell. *cellno* refers to the NUMA cell you're interested in. *pagesize* is a scaled integer (see **NOTES** above). Alternatively, if *--all* is used, info on each possible combination of NUMA cell and page size is printed out.

allocpages

Syntax:

```
allocpages [--pagesize] pagesize [--pagecount] pagecount [[--cellno] cellno] [
```

Change the size of pages pool of *pagesize* on the host. If *--add* is specified, then *pagecount* pages are added into the pool. However, if *--add* wasn't specified, then the *pagecount* is taken as the new absolute size of the pool (this may be used to free some pages and size the pool down). The *cellno* modifier can be used to narrow the modification down to a single host NUMA cell. On the other end of spectrum lies *--all* which executes the modification on all NUMA cells.

cpu-baseline

Syntax:

```
cpu-baseline FILE [--features] [--migratable]
```

Compute baseline CPU which will be supported by all host CPUs given in <file>. (See **hypervisor-cpu-baseline** command to get a CPU which can be provided by a specific hypervisor.) The list of host CPUs is built by extracting all <cpu> elements from the <file>. Thus, the <file> can contain either a set of <cpu> elements separated by new lines or even a set of complete <capabilities> elements printed by **capabilities** command. If *--features* is specified, then the resulting XML description will explicitly include all features that make up the CPU, without this option features that are part of the CPU model will not be listed in the XML description. If *--migratable* is specified, features that block migration will not be included in the resulting CPU.

cpu-compare

Syntax:

```
cpu-compare FILE [--error] [--validate]
```

Compare CPU definition from XML <file> with host CPU. (See **hypervisor-cpu-compare** command for comparing the CPU definition with the CPU which a specific hypervisor is able to provide on the host.) The

XML <file> may contain either host or guest CPU definition. The host CPU definition is the <cpu> element and its contents as printed by **capabilities** command. The guest CPU definition is the <cpu> element and its contents from domain XML definition or the CPU definition created from the host CPU model found in domain capabilities XML (printed by **domcapabilities** command). In addition to the <cpu> element itself, this command accepts full domain XML, capabilities XML, or domain capabilities XML containing the CPU definition. For more information on guest CPU definition see: *https://libvirt.org/formatdomain.html#elementsCPU*. If *−−error* is specified, the command will return an error when the given CPU is incompatible with host CPU and a message providing more details about the incompatibility will be printed out. If *−−validate* is specified, validates the format of the XML document against an internal RNG schema.

cpu−models
Syntax:

```
cpu−models arch
```

Print the list of CPU models known by libvirt for the specified architecture. Whether a specific hypervisor is able to create a domain which uses any of the printed CPU models is a separate question which can be answered by looking at the domain capabilities XML returned by **domcapabilities** command. Moreover, for some architectures libvirt does not know any CPU models and the usable CPU models are only limited by the hypervisor. This command will print that all CPU models are accepted for these architectures and the actual list of supported CPU models can be checked in the domain capabilities XML.

hypervisor−cpu−compare
Syntax:

```
hypervisor−cpu−compare FILE [virttype] [emulator] [arch] [machine] [−−error] [
```

Compare CPU definition from XML <file> with the CPU the hypervisor is able to provide on the host. (This is different from **cpu−compare** which compares the CPU definition with the host CPU without considering any specific hypervisor and its abilities.)

The XML *FILE* may contain either a host or guest CPU definition. The host CPU definition is the <cpu> element and its contents as printed by the **capabilities** command. The guest CPU definition is the <cpu> element and its contents from the domain XML definition or the CPU definition created from the host CPU model found in the domain capabilities XML (printed by the **domcapabilities** command). In addition to the <cpu> element itself, this command accepts full domain XML, capabilities XML, or domain capabilities XML containing the CPU definition. For more information on guest CPU definition see: *https://libvirt.org/formatdomain.html#elementsCPU*.

The *virttype* option specifies the virtualization type (usable in the 'type' attribute of the <domain> top level element from the domain XML). *emulator* specifies the path to the emulator, *arch* specifies the CPU architecture, and *machine* specifies the machine type. If *−−error* is specified, the command will return an error when the given CPU is incompatible with the host CPU and a message providing more details about the incompatibility will be printed out. If *−−validate* is specified, validates the format of the XML document against an internal RNG schema.

hypervisor−cpu−baseline
Syntax:

```
hypervisor−cpu−baseline FILE [virttype] [emulator] [arch] [machine] [−−features
```

Compute a baseline CPU which will be compatible with all CPUs defined in an XML *file* and with the CPU the hypervisor is able to provide on the host. (This is different from **cpu−baseline** which does not consider any hypervisor abilities when computing the baseline CPU.)

The XML *FILE* may contain either host or guest CPU definitions describing the host CPU model. The host CPU definition is the <cpu> element and its contents as printed by **capabilities** command. The guest CPU definition may be created from the host CPU model found in domain capabilities XML (printed by **domcapabilities** command). In addition to the <cpu> elements, this command accepts full capabilities XMLs, or domain capabilities XMLs containing the CPU definitions. It is recommended to use only the CPU definitions from domain capabilities, as on some architectures using the host CPU definition may either fail or provide unexpected results.

When *FILE* contains only a single CPU definition, the command will print the same CPU with restrictions imposed by the capabilities of the hypervisor. Specifically, running th **virsh hypervisor−cpu−baseline** command with no additional options on the result of **virsh domcapabilities** will transform the host CPU model from domain capabilities XML to a form directly usable in domain XML.

The *virttype* option specifies the virtualization type (usable in the 'type' attribute of the <domain> top level element from the domain XML). *emulator* specifies the path to the emulator, *arch* specifies the CPU architecture, and *machine* specifies the machine type. If −−*features* is specified, then the resulting XML description will explicitly include all features that make up the CPU, without this option features that are part of the CPU model will not be listed in the XML description. If −−*migratable* is specified, features that block migration will not be included in the resulting CPU.

DOMAIN COMMANDS

The following commands manipulate domains directly, as stated previously most commands take domain as the first parameter. The *domain* can be specified as a short integer, a name or a full UUID.

autostart

Syntax:

```
autostart [−−disable] domain
```

Configure a domain to be automatically started at boot.

The option −−*disable* disables autostarting.

blkdeviotune

Syntax:

```
blkdeviotune domain device [[−−config] [−−live] | [−−current]]
   [[total−bytes−sec] | [read−bytes−sec] [write−bytes−sec]]
   [[total−iops−sec] | [read−iops−sec] [write−iops−sec]]
   [[total−bytes−sec−max] | [read−bytes−sec−max] [write−bytes−sec−max]]
   [[total−iops−sec−max] | [read−iops−sec−max] [write−iops−sec−max]]
   [[total−bytes−sec−max−length] |
    [read−bytes−sec−max−length] [write−bytes−sec−max−length]]
   [[total−iops−sec−max−length] |
    [read−iops−sec−max−length] [write−iops−sec−max−length]]
   [size−iops−sec] [group−name]
```

Set or query the block disk io parameters for a block device of *domain*. *device* specifies a unique target name (<target dev='name'/>) or source file (<source file='name'/>) for one of the disk devices attached to *domain* (see also **domblklist** for listing these names).

If no limit is specified, it will query current I/O limits setting. Otherwise, alter the limits with these flags: −−*total−bytes−sec* specifies total throughput limit as a scaled integer, the default being bytes per second if no suffix is specified. −−*r ead−bytes−sec* specifies read throughput limit as a scaled integer, the default being bytes per second if no suffix is specified. −−*write−bytes−sec* specifies write throughput limit as a scaled integer, the default being bytes per second if no suffix is specified. −−*total−iops−sec* specifies total

I/O operations limit per second. *−−read−iops−sec* specifies read I/O operations limit per second. *−−write−iops−sec* specifies write I/O operations limit per second. *−−total−bytes−sec−max* specifies maximum total throughput limit as a scaled integer, the default being bytes per second if no suffix is specified *−−read−bytes−sec−max* specifies maximum read throughput limit as a scaled integer, the default being bytes per second if no suffix is specified. *−−write−bytes−sec−max* specifies maximum write throughput limit as a scaled integer, the default being bytes per second if no suffix is specified. *−−total−iops−sec−max* specifies maximum total I/O operations limit per second. *−−read−iops−sec−max* specifies maximum read I/O operations limit per second. *−−write−iops−sec−max* specifies maximum write I/O operations limit per second. *−−total−bytes−sec−max−length* specifies duration in seconds to allow maximum total throughput limit. *−−read−bytes−sec−max−length* specifies duration in seconds to allow maximum read throughput limit. *−−write−bytes−sec−max−length* specifies duration in seconds to allow maximum write throughput limit. *−−total−iops−sec−max−length* specifies duration in seconds to allow maximum total I/O operations limit. *−−read−iops−sec−max−length* specifies duration in seconds to allow maximum read I/O operations limit. *−−write−iops−sec−max−length* specifies duration in seconds to allow maximum write I/O operations limit. *−−size−iops−sec* specifies size I/O operations limit per second. *−−group−name* specifies group name to share I/O quota between multiple drives. For a QEMU domain, if no name is provided, then the default is to have a single group for each *device*.

Older versions of virsh only accepted these options with underscore instead of dash, as in *−−total_bytes_sec*.

Bytes and iops values are independent, but setting only one value (such as −−read−bytes−sec) resets the other two in that category to unlimited. An explicit 0 also clears any limit. A non−zero value for a given total cannot be mixed with non−zero values for read or write.

It is up to the hypervisor to determine how to handle the length values. For the QEMU hypervisor, if an I/O limit value or maximum value is set, then the default value of 1 second will be displayed. Supplying a 0 will reset the value back to the default.

If *−−live* is specified, affect a running guest. If *−−config* is specified, affect the next start of a persistent guest. If*−−curr ent* is specified, it is equivalent to either *−−live* or *−−config*, depending on the current state of the guest. When setting the disk io parameters both *−−live* and *−−config* flags may be given, but *−−current* is exclusive. For querying only one of *−−live*, *−−config* or *−−current* can be specified. If no flag is specified, behavior is different depending on hypervisor.

blkiotune

Syntax:

```
blkiotune domain [--weight weight] [--device-weights device-weights]
   [--device-read-iops-sec device-read-iops-sec]
   [--device-write-iops-sec device-write-iops-sec]
   [--device-read-bytes-sec device-read-bytes-sec]
   [--device-write-bytes-sec device-write-bytes-sec]
   [[--config] [--live] | [--current]]
```

Display or set the blkio parameters. QEMU/KVM supports *−−weight*. *−−weight* is in range [100, 1000]. After kernel 2.6.39, the value could be in the range [10, 1000].

device−weights is a single string listing one or more device/weight pairs, in the format of /path/to/device,weight,/path/to/device,weight. Each weight is in the range [100, 1000], [10, 1000] after kernel 2.6.39, or the value 0 to remove that device from per−device listings. Only the devices listed in the string are modified; any existing per−device weights for other devices remain unchanged.

device−read−iops−sec is a single string listing one or more device/read_iops_sec pairs, int the format of /path/to/device,read_iops_sec,/path/to/device,read_iops_sec. Each read_iops_sec is a number which type is

unsigned int, value 0 to remove that device from per−device listing. Only the devices listed in the string are modified; any existing per−device read_iops_sec for other devices remain unchanged.

device−write−iops−sec is a single string listing one or more device/write_iops_sec pairs, int the format of /path/to/device,write_iops_sec,/path/to/device,write_iops_sec. Each write_iops_sec is a number which type is unsigned int, value 0 to remove that device from per−device listing. Only the devices listed in the string are modified; any existing per−device write_iops_sec for other devices remain unchanged.

device−read−bytes−sec is a single string listing one or more device/read_bytes_sec pairs, int the format of /path/to/device,read_bytes_sec,/path/to/device,read_bytes_sec. Each read_bytes_sec is a number which type is unsigned long long, value 0 to remove that device from per−device listing. Only the devices listed in the string are modified; any existing per−device read_bytes_sec for other devices remain unchanged.

device−write−bytes−sec is a single string listing one or more device/write_bytes_sec pairs, int the format of /path/to/device,write_bytes_sec,/path/to/device,write_bytes_sec. Each write_bytes_sec is a number which type is unsigned long long, value 0 to remove that device from per−device listing. Only the devices listed in the string are modified; any existing per−device write_bytes_sec for other devices remain unchanged.

If *−−live* is specified, affect a running guest. If *−−config* is specified, affect the next start of a persistent guest. If*−−curr ent* is specified, it is equivalent to either *−−live* or *−−config*, depending on the current state of the guest. Both *−−live* and *−−config* flags may be given, but *−−current* is exclusive. If no flag is specified, behavior is different depending on hypervisor.

blockcommit

Syntax:

```
blockcommit domain path [bandwidth] [--bytes] [base]
   [--shallow] [top] [--delete] [--keep-relative]
   [--wait [--async] [--verbose]] [--timeout seconds]
   [--active] [{--pivot | --keep-overlay}]
```

Reduce the length of a backing image chain, by committing changes at the top of the chain (snapshot or delta files) into backing images. By default, this command attempts to flatten the entire chain. If *base* and/or *top* are specified as files within the backing chain, then the operation is constrained to committing just that portion of the chain; *−−shallow* can be used instead of *base* to specify the immediate backing file of the resulting top image to be committed. The files being committed are rendered invalid, possibly as soon as the operation starts; using the *−−delete* flag will attempt to remove these invalidated files at the successful completion of the commit operation. When the *−−keep−relative* flag is used, the backing file paths will be kept relative.

When *top* is omitted or specified as the active image, it is also possible to specify *−−active* to trigger a two−phase active commit. In the first phase, *top* is copied into *base* and the job can only be canceled, with top still containing data not yet in base. In the second phase, *top* and *base* remain identical until a call to **blockjob** with the *−−abort* flag (keeping top as the active image that tracks changes from that point in time) or the *−−pivot* flag (making base the new active image and invalidating top).

By default, this command returns as soon as possible, and data for the entire disk is committed in the background; the progress of the operation can be checked with **blockjob**. However, if *−−wait* is specified, then this command will block until the operation completes (or for *−−active*, enters the second phase), or until the operation is canceled because the optional *timeout* in seconds elapses or SIGINT is sent (usually with **Ctrl−C**). Using*−−verbose* along with *−−wait* will produce periodic status updates. If job cancellation is triggered, *−−async* will return control to the user as fast as possible, otherwise the command may continue to block a little while longer until the job is done cleaning up. Using *−−pivot* is shorthand for combining *−−active −−wait* with an automatic **blockjob** *−−pivot*; and using *−−keep−overlay* is shorthand for

combining *--active --wait* with an automatic **blockjob** *--abort*.

path specifies fully-qualified path of the disk; it corresponds to a unique target name (<target dev='name'/>) or source file (<source file='name'/>) for one of the disk devices attached to *domain* (see also **domblklist** for listing these names). *bandwidth* specifies copying bandwidth limit in MiB/s, although for QEMU, it may be non-zero only for an online domain. For further information on the *bandwidth* argument see the corresponding section for the **blockjob** command.

blockcopy

Syntax:

```
blockcopy domain path { dest [format] [--blockdev] | --xml file }
   [--shallow] [--reuse-external] [bandwidth]
   [--wait [--async] [--verbose]] [{--pivot | --finish}]
   [--timeout seconds] [granularity] [buf-size] [--bytes]
   [--transient-job] [--synchronous-writes]
```

Copy a disk backing image chain to a destination. Either *dest* as the destination file name, or *--xml* with the name of an XML file containing a top-level <disk> element describing the destination, must be present. Additionally, if *dest* is given, *format* should be specified to declare the format of the destination (if *format* is omitted, then libvirt will reuse the format of the source, or with *--reuse-external* will be forced to probe the destination format, which could be a potential security hole). The command supports *--raw* as a boolean flag synonym for *--format=raw*. When using*dest*, the destination is treated as a re gular file unless *--blockdev* is used to signal that it is a block device. By default, this command flattens the entire chain; but if *--shallow* is specified, the copy shares the backing chain.

If *--reuse-external* is specified, then the destination must exist and have sufficient space to hold the copy. If *--shallow* is used in conjunction with *--reuse-external* then the pre-created image must have guest visible contents identical to guest visible contents of the backing file of the original image. This may be used to modify the backing file names on the destination.

By default, the copy job runs in the background, and consists of two phases. Initially, the job must copy all data from the source, and during this phase, the job can only be canceled to revert back to the source disk, with no guarantees about the destination. After this phase completes, both the source and the destination remain mirrored until a call to **blockjob** with the *--abort* and *--pivot* flags pivots over to the copy, or a call without *--pivot* leaves the destination as a faithful copy of that point in time. However, if *--wait* is specified, then this command will block until the mirroring phase begins, or cancel the operation if the optional *timeout* in seconds elapses or SIGINT is sent (usually with **Ctrl-C**). Using*--verbose* along with *--wait* will produce periodic status updates. Using *--pivot* (similar to **blockjob** *--pivot*) or *--finish* (similar to **blockjob** *--abort*) implies *--wait*, and will additionally end the job cleanly rather than leaving things in the mirroring phase. If job cancellation is triggered by timeout or by *--finish*, *--async* will return control to the user as fast as possible, otherwise the command may continue to block a little while longer until the job has actually cancelled.

path specifies fully-qualified path of the disk. *bandwidth* specifies copying bandwidth limit in MiB/s. Specifying a negative value is interpreted as an unsigned long long value that might be essentially unlimited, but more likely would overflow; it is safer to use 0 for that purpose. For further information on the *bandwidth* argument see the corresponding section for the **blockjob** command. Specifying *granularity* allows fine-tuning of the granularity that will be copied when a dirty region is detected; larger values trigger less I/O overhead but may end up copying more data overall (the default value is usually correct); hypervisors may restrict this to be a power of two or fall within a certain range. Specifying *buf-size* will control how much data can be simultaneously in-flight during the copy; larger values use more memory but may allow faster completion (the default value is usually correct).

--transient-job allows specifying that the user does not require the job to be recovered if the VM crashes

or is turned off before the job completes. This flag removes the restriction of copy jobs to transient domains if that restriction is applied by the hypervisor.

If *--synchronous-writes* is specified the block job will wait for guest writes to be propagated both to the original image and to the destination of the copy so that it's guaranteed that the job converges if the destination storage is slower. This may impact performance of writes while the blockjob is running.

blockjob

Syntax:

```
blockjob domain path { [--abort] [--async] [--pivot] |
   [--info] [--raw] [--bytes] | [bandwidth] }
```

Manage active block operations. There are three mutually-exclusive modes: *--info*, *bandwidth*, and *--abort*. *--async* and *--pivot* imply abort mode; *--raw* implies info mode; and if no mode was given, *--info* mode is assumed.

path specifies fully-qualified path of the disk; it corresponds to a unique target name (<target dev='name'/>) or source file (<source file='name'/>) for one of the disk devices attached to *domain* (see also **domblklist** for listing these names).

In *--abort* mode, the active job on the specified disk will be aborted. If *--async* is also specified, this command will return immediately, rather than waiting for the cancellation to complete. If *--pivot* is specified, this requests that an active copy or active commit job be pivoted over to the new image.

In *--info* mode, the active job information on the specified disk will be printed. By default, the output is a single human-readable summary line; this format may change in future versions. Adding*--r aw* lists each field of the struct, in a stable format. If the *--bytes* flag is set, then the command errors out if the server could not supply bytes/s resolution; when omitting the flag, raw output is listed in MiB/s and human-readable output automatically selects the best resolution supported by the server.

bandwidth can be used to set bandwidth limit for the active job in MiB/s. If *--bytes* is specified then the bandwidth value is interpreted in bytes/s. Specifying a negative value is interpreted as an unsigned long value or essentially unlimited. The hypervisor can choose whether to reject the value or convert it to the maximum value allowed. Optionally a scaled positive number may be used as bandwidth (see **NOTES** above). Using *--bytes* with a scaled value permits a finer granularity to be selected. A scaled value used without *--bytes* will be rounded down to MiB/s. Note that the *--bytes* may be unsupported by the hypervisor.

Note that the progress reported for blockjobs corresponding to a pull-mode backup don't report progress of the backup but rather usage of temporary space required for the backup.

blockpull

Syntax:

```
blockpull domain path [bandwidth] [--bytes] [base]
   [--wait [--verbose] [--timeout seconds] [--async]]
   [--keep-relative]
```

Populate a disk from its backing image chain. By default, this command flattens the entire chain; but if *base* is specified, containing the name of one of the backing files in the chain, then that file becomes the new backing file and only the intermediate portion of the chain is pulled. Once all requested data from the backing image chain has been pulled, the disk no longer depends on that portion of the backing chain.

By default, this command returns as soon as possible, and data for the entire disk is pulled in the background; the progress of the operation can be checked with **blockjob**. However, if *--wait* is specified, then

this command will block until the operation completes, or cancel the operation if the optional *timeout* in seconds elapses or SIGINT is sent (usually with **Ctrl−C**). Using−−*verbose* along with −−*wait* will produce periodic status updates. If job cancellation is triggered, −−*async* will return control to the user as fast as possible, otherwise the command may continue to block a little while longer until the job is done cleaning up.

Using the −−*keep−relative* flag will keep the backing chain names relative.

path specifies fully−qualified path of the disk; it corresponds to a unique target name (<target dev='name'/>) or source file (<source file='name'/>) for one of the disk devices attached to *domain* (see also **domblklist** for listing these names). *bandwidth* specifies copying bandwidth limit in MiB/s. For further information on the *bandwidth* argument see the corresponding section for the **blockjob** command.

blockresize

Syntax:

```
blockresize domain path size
```

Resize a block device of domain while the domain is running, *path* specifies the absolute path of the block device; it corresponds to a unique target name (<target dev='name'/>) or source file (<source file='name'/>) for one of the disk devices attached to *domain* (see also **domblklist** for listing these names).

size is a scaled integer (see **NOTES** above) which defaults to KiB (blocks of 1024 bytes) if there is no suffix. You must use a suffix of "B" to get bytes (note that for historical reasons, this differs from **vol−resize** which defaults to bytes without a suffix).

console

Syntax:

```
console domain [devname] [--safe] [--force]
```

Connect the virtual serial console for the guest. The optional *devname* parameter refers to the device alias of an alternate console, serial or parallel device configured for the guest. If omitted, the primary console will be opened.

If the flag −−*safe* is specified, the connection is only attempted if the driver supports safe console handling. This flag specifies that the server has to ensure exclusive access to console devices. Optionally the −−*force* flag may be specified, requesting to disconnect any existing sessions, such as in a case of a broken connection.

cpu−stats

Syntax:

```
cpu-stats domain [--total] [start] [count]
```

Provide cpu statistics information of a domain. The domain should be running. Default it shows stats for all CPUs, and a total. Use −−*total* for only the total stats, *start* for only the per−cpu stats of the CPUs from *start*, *count* for only *count* CPUs' stats.

create

Syntax:

```
create FILE [--console] [--paused] [--autodestroy]
   [--pass-fds N,M,...] [--validate]
```

Create a domain from an XML <file>. Optionally, −−*validate* option can be passed to validate the format of the input XML file against an internal RNG schema (identical to using virt−xml−validate(1) tool).

Domains created using this command are going to be either transient (temporary ones that will vanish once destroyed) or existing persistent guests that will run with one−time use configuration, leaving the persistent XML untouched (this can come handy during an automated testing of various configurations all based on the original XML). See the example below for usage demonstration.

The domain will be paused if the *−−paused* option is used and supported by the driver; otherwise it will be running. If *−−console* is requested, attach to the console after creation. If *−−autodestroy* is requested, then the guest will be automatically destroyed when virsh closes its connection to libvirt, or otherwise exits.

If *−−pass−fds* is specified, the argument is a comma separated list of open file descriptors which should be pass on into the guest. The file descriptors will be re−numbered in the guest, starting from 3. This is only supported with container based virtualization.

Example:

1. prepare a template from an existing domain (skip directly to 3a if writing one from scratch)

```
# virsh dumpxml <domain> > domain.xml
```

2. edit the template using an editor of your choice and:
   a. DO CHANGE! <name> and <uuid> (<uuid> can also be removed), or
   b. DON'T CHANGE! either <name> or <uuid>

```
# $EDITOR domain.xml
```

3. create a domain from domain.xml, depending on whether following 2a or 2b respectively:
   a. the domain is going to be transient
   b. an existing persistent guest will run with a modified one−time configuration

```
# virsh create domain.xml
```

define

Syntax:

```
define FILE [−−validate]
```

Define a domain from an XML <file>. Optionally, the format of the input XML file can be validated against an internal RNG schema with *−−validate* (identical to using virt−xml−validate(1) tool). The domain definition is registered but not started. If domain is already running, the changes will take effect on the next boot.

desc

Syntax:

```
desc domain [[−−live] [−−config] |
   [−−current]] [−−title] [−−edit] [−−new−desc
   New description or title message]
```

Show or modify description and title of a domain. These values are user fields that allow storing arbitrary textual data to allow easy identification of domains. Title should be short, although it's not enforced. (See also **metadata** that works with XML based domain metadata.)

Flags *−−live* or *−−config* select whether this command works on live or persistent definitions of the domain. If both *−−live* and *−−config* are specified, the *−−config* option takes precedence on getting the current description and both live configuration and config are updated while setting the description. *−−current* is exclusive and implied if none of these was specified.

Flag *--edit* specifies that an editor with the contents of current description or title should be opened and the contents saved back afterwards.

Flag *--title* selects operation on the title field instead of description.

If neither of *--edit* and *--new-desc* are specified the note or description is displayed instead of being modified.

destroy

Syntax:

```
destroy domain [--graceful]
```

Immediately terminate the domain *domain*. This doesn't give the domain OS any chance to react, and it's the equivalent of ripping the power cord out on a physical machine. In most cases you will want to use the **shutdown** command instead. However, this does not delete any storage volumes used by the guest, and if the domain is persistent, it can be restarted later.

If *domain* is transient, then the metadata of any snapshots will be lost once the guest stops running, but the snapshot contents still exist, and a new domain with the same name and UUID can restore the snapshot metadata with **snapshot-create**. Similarly, the metadata of any checkpoints will be lost, but can be restored with **checkpoint-create**.

If *--graceful* is specified, don't resort to extreme measures (e.g. SIGKILL) when the guest doesn't stop after a reasonable timeout; return an error instead.

domblkerror

Syntax:

```
domblkerror domain
```

Show errors on block devices. This command usually comes handy when**domstate** command says that a domain was paused due to I/O error. The**domblk error** command lists all block devices in error state and the error seen on each of them.

domblkinfo

Syntax:

```
domblkinfo domain [block-device --all] [--human]
```

Get block device size info for a domain. A *block-device* corresponds to a unique target name (<target dev='name'/>) or source file (<source file='name'/>) for one of the disk devices attached to *domain* (see also **domblklist** for listing these names). If *--human* is set, the output will have a human readable output. If *--all* is set, the output will be a table showing all block devices size info associated with *domain*. The *--all* option takes precedence of the others.

domblklist

Syntax:

```
domblklist domain [--inactive] [--details]
```

Print a table showing the brief information of all block devices associated with *domain*. If *--inactive* is specified, query the block devices that will be used on the next boot, rather than those currently in use by a running domain. If *--details* is specified, disk type and device value will also be printed. Other contexts that require a block device name (such as *domblkinfo* or *snapshot-create* for disk snapshots) will accept either target or unique source names printed by this command.

domblkstat
Syntax:

```
domblkstat domain [block-device] [--human]
```

Get device block stats for a running domain. A *block-device* corresponds to a unique target name (<target dev='name'/>) or source file (<source file='name'/>) for one of the disk devices attached to *domain* (see also **domblklist** for listing these names). On a LXC or QEMU domain, omitting the *block-device* yields device block stats summarily for the entire domain.

Use *--human* for a more human readable output.

Availability of these fields depends on hypervisor. Unsupported fields are missing from the output. Other fields may appear if communicating with a newer version of libvirtd.

Explanation of fields (fields appear in the following order):

- rd_req - count of read operations
- rd_bytes - count of read bytes
- wr_req - count of write operations
- wr_bytes - count of written bytes
- errs - error count
- flush_operations - count of flush operations
- rd_total_times - total time read operations took (ns)
- wr_total_times - total time write operations took (ns)
- flush_total_times - total time flush operations took (ns)
- <-- other fields provided by hypervisor -->

domblkthreshold
Syntax:

```
domblkthreshold domain dev threshold
```

Set the threshold value for delivering the block-threshold event. *dev* specifies the disk device target or backing chain element of given device using the 'target[1]' syntax.*thr eshold* is a scaled value of the offset. If the block device should write beyond that offset the event will be delivered.

domcontrol
Syntax:

```
domcontrol domain
```

Returns state of an interface to VMM used to control a domain. For states other than "ok" or "error" the command also prints number of seconds elapsed since the control interface entered its current state.

domdirtyrate-calc
Syntax:

```
domdirtyrate-calc <domain> [--seconds <sec>]
```

Calculate an active domain's memory dirty rate which may be expected by user in order to decide whether it's proper to be migrated out or not. The **seconds** parameter can be used to calculate dirty rate in a specific time which allows 60s at most now and would be default to 1s if missing. The calculated dirty rate

information is available by calling 'domstats −−dirtyrate'.

domdisplay
Syntax:

```
domdisplay domain [−−include−password] [[−−type] type] [−−all]
```

Output a URI which can be used to connect to the graphical display of the domain via VNC, SPICE or RDP. The particular graphical display type can be selected using the**type** parameter (e.g. "vnc", "spice", "rdp"). If*−−include−passwor d* is specified, the SPICE channel password will be included in the URI. If *−−all* is specified, then all show all possible graphical displays, for a VM could have more than one graphical displays.

domfsfreeze
Syntax:

```
domfsfreeze domain [[−−mountpoint] mountpoint...]
```

Freeze mounted filesystems within a running domain to prepare for consistent snapshots.

The *−−mountpoint* option takes a parameter **mountpoint**, which is a mount point path of the filesystem to be frozen. This option can occur multiple times. If this is not specified, every mounted filesystem is frozen.

Note: **snapshot−create** command has a *−−quiesce* option to freeze and thaw the filesystems automatically to keep snapshots consistent. **domfsfr eeze** command is only needed when a user wants to utilize the native snapshot features of storage devices not supported by libvirt.

domfsinfo
Syntax:

```
domfsinfo domain
```

Show a list of mounted filesystems within the running domain. The list contains mountpoints, names of a mounted device in the guest, filesystem types, and unique target names used in the domain XML (<target dev='name'/>).

Note that this command requires a guest agent configured and running in the domain's guest OS.

domfsthaw
Syntax:

```
domfsthaw domain [[−−mountpoint] mountpoint...]
```

Thaw mounted filesystems within a running domain, which have been frozen by domfsfreeze command.

The *−−mountpoint* option takes a parameter **mountpoint**, which is a mount point path of the filesystem to be thawed. This option can occur multiple times. If this is not specified, every mounted filesystem is thawed.

domfstrim
Syntax:

```
domfstrim domain [−−minimum bytes] [−−mountpoint mountPoint]
```

Issue a fstrim command on all mounted filesystems within a running domain. It discards blocks which are not in use by the filesystem. If *−−minimum* **bytes** is specified, it tells guest kernel length of contiguous free range. Smaller than this may be ignored (this is a hint and the guest may not respect it). By increasing this

value, the fstrim operation will complete more quickly for filesystems with badly fragmented free space, although not all blocks will be discarded. The default value is zero, meaning "discard every free block". Moreover, if a user wants to trim only one mount point, it can be specified via optional *--mountpoint* parameter.

domhostname

Syntax:

```
domhostname domain [--source lease|agent]
```

Returns the hostname of a domain, if the hypervisor makes it available.

The *--source* argument specifies what data source to use for the hostnames, currently 'lease' to read DHCP leases or 'agent' to query the guest OS via an agent. If unspecified, driver returns the default method available (some drivers support only one type of source).

domid

Syntax:

```
domid domain-name-or-uuid
```

Convert a domain name (or UUID) to a domain id

domif-getlink

Syntax:

```
domif-getlink domain interface-device [--config]
```

Query link state of the domain's virtual interface. If *--config* is specified, query the persistent configuration, for compatibility purposes, *--persistent* is alias of *--config*.

interface-device can be the interface's target name or the MAC address.

domif-setlink

Syntax:

```
domif-setlink domain interface-device state [--config]
```

Modify link state of the domain's virtual interface. Possible values for state are "up" and "down". If *--config* is specified, only the persistent configuration of the domain is modified, for compatibility purposes, *--persistent* is alias of *--config*. *interface-device* can be the interface's target name or the MAC address.

domifaddr

Syntax:

```
domifaddr domain [interface] [--full]
   [--source lease|agent|arp]
```

Get a list of interfaces of a running domain along with their IP and MAC addresses, or limited output just for one interface if *interface* is specified. Note that *interface* can be driver dependent, it can be the name within guest OS or the name you would see in domain XML. Moreover, the whole command may require a guest agent to be configured for the queried domain under some hypervisors, notably QEMU.

If *--full* is specified, the interface name and MAC address is always displayed when the interface has multiple IP addresses or aliases; otherwise, only the interface name and MAC address is displayed for the first name and MAC address with "-" for the others using the same name and MAC address.

The *--source* argument specifies what data source to use for the addresses, currently 'lease' to read DHCP leases, 'agent' to query the guest OS via an agent, or 'arp' to get IP from host's arp tables. If unspecified, 'lease' is the default.

backup-begin
Syntax:

```
backup-begin domain [backupxml] [checkpointxml] [--reuse-external]
```

Begin a new backup job. If *backupxml* is omitted, this defaults to a full backup using a push model to filenames generated by libvirt; supplying XML allows fine-tuning such as requesting an incremental backup relative to an earlier checkpoint, controlling which disks participate or which filenames are involved, or requesting the use of a pull model backup. The *backup-dumpxml* command shows any resulting values assigned by libvirt. For more information on backup XML, see: *https://libvirt.org/formatbackup.html*

If *--reuse-external* is used it instructs libvirt to reuse temporary and output files provided by the user in *backupxml*.

If *checkpointxml* is specified, a second file with a top-level element of *domaincheckpoint* is used to create a simultaneous checkpoint, for doing a later incremental backup relative to the time the backup was created. See *checkpoint-create* for more details on checkpoints.

This command returns as soon as possible, and the backup job runs in the background; the progress of a push model backup can be checked with *domjobinfo* or by waiting for an event with *event* (the progress of a pull model backup is under the control of whatever third party connects to the NBD export). The job is ended with *domjobabort*.

backup-dumpxml
Syntax:

```
backup-dumpxml domain
```

Output XML describing the current backup job.

domiflist
Syntax:

```
domiflist domain [--inactive]
```

Print a table showing the brief information of all virtual interfaces associated with *domain*. If *--inactive* is specified, query the virtual interfaces that will be used on the next boot, rather than those currently in use by a running domain. Other contexts that require a MAC address of virtual interface (such as *detach-interface* or *domif-setlink*) will accept the MAC address printed by this command.

domifstat
Syntax:

```
domifstat domain interface-device
```

Get network interface stats for a running domain. The network interface stats are only available for interfaces that have a physical source interface. This does not include, for example, a 'user' interface type since it is a virtual LAN with NAT to the outside world. *interface-device* can be the interface target by name or MAC address.

domiftune
Syntax:

```
domiftune domain interface-device [[--config] [--live] | [--current]]
   [*--inbound average,peak,burst,floor*]
   [*--outbound average,peak,burst*]
```

Set or query the domain's network interface's bandwidth parameters. *interface-device* can be the interface's target name (<target dev='name'/>), or the MAC address.

If no *--inbound* or *--outbound* is specified, this command will query and show the bandwidth settings. Otherwise, it will set the inbound or outbound bandwidth. *average,peak,burst,floor* is the same as in command *attach-interface*. Values for *average*, *peak* and *floor* are expressed in kilobytes per second, while *burst* is expressed in kilobytes in a single burst at *peak* speed as described in the Network XML documentation at *https://libvirt.org/formatnetwork.html#elementQoS*.

To clear inbound or outbound settings, use *--inbound* or *--outbound* respectfully with average value of zero.

If *--live* is specified, affect a running guest. If *--config* is specified, affect the next start of a persistent guest. If*--curr ent* is specified, it is equivalent to either *--live* or *--config*, depending on the current state of the guest. Both *--live* and *--config* flags may be given, but *--current* is exclusive. If no flag is specified, behavior is different depending on hypervisor.

dominfo

Syntax:

```
dominfo domain
```

Returns basic information about the domain.

domjobabort

Syntax:

```
domjobabort domain
```

Abort the currently running domain job.

domjobinfo

Syntax:

```
domjobinfo domain [--completed [--keep-completed]] [--anystats] [--rawstats]
```

Returns information about jobs running on a domain. *--completed* tells virsh to return information about a recently finished job. Statistics of a completed job are automatically destroyed once read (unless *--keep-completed* is used) or when libvirtd is restarted.

Normally only statistics for running and successful completed jobs are printed. *--anystats* can be used to also display statistics for failed jobs.

In case *--rawstats* is used, all fields are printed as received from the server without any attempts to interpret the data. The "Job type:" field is special, since it's reported by the API and not part of stats.

Note that time information returned for completed migrations may be completely irrelevant unless both source and destination hosts have synchronized time (i.e., NTP daemon is running on both of them).

domlaunchsecinfo

Syntax:

```
domlaunchsecinfo domain
```

Returns information about the launch security parameters associated with a running domain.

The set of parameters reported will vary depending on which type of launch security protection is active. If none is active, no parameters will be reported.

domsetlaunchsecstate
Syntax:

```
domsetlaunchsecstate domain --secrethdr hdr-filename
    --secret secret-filename [--set-address address]
```

Set a launch security secret in the guest's memory. The guest must have a launchSecurity type enabled in its configuration and be in a paused state. On success, the guest can be transitioned to a running state. On failure, the guest should be destroyed.

--secrethdr specifies a filename containing the base64-encoded secret header. The header includes artifacts needed by the hypervisor firmware to recover the plain text of the launch secret. *--secret* specifies the filename containing the base64-encoded encrypted launch secret.

The *--set-address* option can be used to specify a physical address within the guest's memory to set the secret. If not specified, the address will be determined by the hypervisor.

dommemstat
Syntax:

```
dommemstat domain [--period seconds] [[--config] [--live] | [--current]]
```

Get memory stats for a running domain.

Availability of these fields depends on hypervisor. Unsupported fields are missing from the output. Other fields may appear if communicating with a newer version of libvirtd.

Explanation of fields:

- **swap_in** - The amount of data read from swap space (in KiB)
- **swap_out** - The amount of memory written out to swap space (in KiB)
- **major_fault** - The number of page faults where disk IO was required
- **minor_fault** - The number of other page faults
- **unused** - The amount of memory left unused by the system (in KiB)
- **available** - The amount of usable memory as seen by the domain (in KiB)
- **actual** - Current balloon value (in KiB)
- **rss** - Resident Set Size of the running domain's process (in KiB)
- **usable** - The amount of memory which can be reclaimed by balloon without causing host swapping (in KiB)
- **last-update** - Timestamp of the last update of statistics (in seconds)
- **disk_caches** - The amount of memory that can be reclaimed without additional I/O, typically disk caches (in KiB)
- **hugetlb_pgalloc** - The number of successful huge page allocations initiated from within the domain
- **hugetlb_pgfail** - The number of failed huge page allocations initiated from within the domain

For QEMU/KVM with a memory balloon, setting the optional *--period* to a value larger than 0 in seconds

will allow the balloon driver to return additional statistics which will be displayed by subsequent **dommemstat** commands. Setting the *−−period* to 0 will stop the balloon driver collection, but does not clear the statistics in the balloon driver. Requires at least QEMU/KVM 1.5 to be running on the host.

The *−−live*, *−−config*, and *−−current* flags are only valid when using the *−−period* option in order to set the collection period for the balloon driver. If *−−live* is specified, only the running guest collection period is affected. If *−−config* is specified, affect the next start of a persistent guest. If *−−current* is specified, it is equivalent to either *−−live* or *−−config*, depending on the current state of the guest.

Both *−−live* and *−−config* flags may be given, but *−−current* is exclusive. If no flag is specified, behavior is different depending on the guest state.

domname

Syntax:

```
domname domain-id-or-uuid
```

Convert a domain Id (or UUID) to domain name

dompmsuspend

Syntax:

```
dompmsuspend domain target [--duration]
```

Suspend a running domain into one of these states (possible *target* values):

- **mem** − equivalent of S3 ACPI state
- **disk** − equivalent of S4 ACPI state
- **hybrid** − RAM is saved to disk but not powered off

The *−−duration* argument specifies number of seconds before the domain is woken up after it was suspended (see also **dompmwakeup**). Default is 0 for unlimited suspend time. (This feature isn't currently supported by any hypervisor driver and 0 should be used.).

Note that this command requires a guest agent configured and running in the domain's guest OS.

Beware that at least for QEMU, the domain's process will be terminated when target disk is used and a new process will be launched when libvirt is asked to wake up the domain. As a result of this, any runtime changes, such as device hotplug or memory settings, are lost unless such changes were made with *−−config* flag.

dompmwakeup

Syntax:

```
dompmwakeup domain
```

Wakeup a domain from pmsuspended state (either suspended by dompmsuspend or from the guest itself). Injects a wakeup into the guest that is in pmsuspended state, rather than waiting for the previously requested duration (if any) to elapse. This operation does not necessarily fail if the domain is running.

domrename

Syntax:

```
domrename domain new-name
```

Rename a domain. This command changes current domain name to the new name specified in the second argument.

Note: Domain must be inactive.

domstate

Syntax:

```
domstate domain [--reason]
```

Returns state about a domain. *--reason* tells virsh to also print reason for the state.

domstats

Syntax:

```
domstats [--raw] [--enforce] [--backing] [--nowait] [--state]
   [--cpu-total] [--balloon] [--vcpu] [--interface]
   [--block] [--perf] [--iothread] [--memory] [--dirtyrate]
   [[--list-active] [--list-inactive]
    [--list-persistent] [--list-transient] [--list-running]y
    [--list-paused] [--list-shutoff] [--list-other]] | [domain ...]
```

Get statistics for multiple or all domains. Without any argument this command prints all available statistics for all domains.

The list of domains to gather stats for can be either limited by listing the domains as a space separated list, or by specifying one of the filtering flags *--list-NNN*. (The approaches can't be combined.)

By default some of the returned fields may be converted to more human friendly values by a set of pretty-printers. To suppress this behavior use the *--raw* flag.

The individual statistics groups are selectable via specific flags. By default all supported statistics groups are returned. Supported statistics groups flags are: *--state*, *--cpu-total*, *--balloon*, *--vcpu*, *--interface*, *--block*, *--perf*, *--iothread*, *--memory*, *--dirtyrate*.

Note that - depending on the hypervisor type and version or the domain state - not all of the following statistics may be returned.

When selecting the *--state* group the following fields are returned:

- **state.state** - state of the VM, returned as number from virDomainState enum
- **state.reason** - reason for entering given state, returned as int from virDomain*Reason enum corresponding to given state

--cpu-total returns:

- **cpu.time** - total cpu time spent for this domain in nanoseconds
- **cpu.user** - user cpu time spent in nanoseconds
- **cpu.system** - system cpu time spent in nanoseconds
- **cpu.haltpoll.success.time** - cpu halt polling success time spent in nanoseconds
- **cpu.haltpoll.fail.time** - cpu halt polling fail time spent in nanoseconds
- **cpu.cache.monitor.count** - the number of cache monitors for this domain
- **cpu.cache.monitor.<num>.name** - the name of cache monitor <num>
- **cpu.cache.monitor.<num>.vcpus** - vcpu list of cache monitor <num>
- **cpu.cache.monitor.<num>.bank.count** - the number of cache banks in cache monitor <num>

- **cpu.cache.monitor.<num>.bank.<index>.id** − host allocated cache id for bank <index> in cache monitor <num>
- **cpu.cache.monitor.<num>.bank.<index>.bytes** − the number of bytes of last level cache that the domain is using on cache bank <index>

−−*balloon* returns:

- **balloon.current** − the memory in KiB currently used
- **balloon.maximum** − the maximum memory in KiB allowed
- **balloon.swap_in** − the amount of data read from swap space (in KiB)
- **balloon.swap_out** − the amount of memory written out to swap space (in KiB)
- **balloon.major_fault** − the number of page faults when disk IO was required
- **balloon.minor_fault** − the number of other page faults
- **balloon.unused** − the amount of memory left unused by the system (in KiB)
- **balloon.available** − the amount of usable memory as seen by the domain (in KiB)
- **balloon.rss** − Resident Set Size of running domain's process (in KiB)
- **balloon.usable** − the amount of memory which can be reclaimed by balloon without causing host swapping (in KiB)
- **balloon.last−update** − timestamp of the last update of statistics (in seconds)
- **balloon.disk_caches** − the amount of memory that can be reclaimed without additional I/O, typically disk (in KiB)
- **balloon.hugetlb_pgalloc** − the number of successful huge page allocations from inside the domain via virtio balloon
- **balloon.hugetlb_pgfail** − the number of failed huge page allocations from inside the domain via virtio balloon

−−*vcpu* returns:

- **vcpu.current** − current number of online virtual CPUs
- **vcpu.maximum** − maximum number of online virtual CPUs
- **vcpu.<num>.state** − state of the virtual CPU <num>, as number from virVcpuState enum
- **vcpu.<num>.time** − virtual cpu time spent by virtual CPU <num> (in microseconds)
- **vcpu.<num>.wait** − virtual cpu time spent by virtual CPU <num> waiting on I/O (in microseconds)
- **vcpu.<num>.halted** − virtual CPU <num> is halted: yes or no (may indicate the processor is idle or even disabled, depending on the architecture)
- **vcpu.<num>.delay** − time the vCPU <num> thread was enqueued by the host scheduler, but was waiting in the queue instead of running. Exposed to the VM as a steal time.

−−*interface* returns:

- **net.count** − number of network interfaces on this domain
- **net.<num>.name** − name of the interface <num>
- **net.<num>.rx.bytes** − number of bytes received
- **net.<num>.rx.pkts** − number of packets received
- **net.<num>.rx.errs** − number of receive errors

- **net.<num>.rx.drop** − number of receive packets dropped
- **net.<num>.tx.bytes** − number of bytes transmitted
- **net.<num>.tx.pkts** − number of packets transmitted
- **net.<num>.tx.errs** − number of transmission errors
- **net.<num>.tx.drop** − number of transmit packets dropped

−−*perf* returns the statistics of all enabled perf events:

- **perf.cmt** − the cache usage in Byte currently used
- **perf.mbmt** − total system bandwidth from one level of cache
- **perf.mbml** − bandwidth of memory traffic for a memory controller
- **perf.cpu_cycles** − the count of cpu cycles (total/elapsed)
- **perf.instructions** − the count of instructions
- **perf.cache_references** − the count of cache hits
- **perf.cache_misses** − the count of caches misses
- **perf.branch_instructions** − the count of branch instructions
- **perf.branch_misses** − the count of branch misses
- **perf.bus_cycles** − the count of bus cycles
- **perf.stalled_cycles_frontend** − the count of stalled frontend cpu cycles
- **perf.stalled_cycles_backend** − the count of stalled backend cpu cycles
- **perf.ref_cpu_cycles** − the count of ref cpu cycles
- **perf.cpu_clock** − the count of cpu clock time
- **perf.task_clock** − the count of task clock time
- **perf.page_faults** − the count of page faults
- **perf.context_switches** − the count of context switches
- **perf.cpu_migrations** − the count of cpu migrations
- **perf.page_faults_min** − the count of minor page faults
- **perf.page_faults_maj** − the count of major page faults
- **perf.alignment_faults** − the count of alignment faults
- **perf.emulation_faults** − the count of emulation faults

See the **perf** command for more details about each event.

−−*block* returns information about disks associated with each domain. Using the −−*backing* flag extends this information to cover all resources in the backing chain, rather than the default of limiting information to the active layer for each guest disk. Information listed includes:

- **block.count** − number of block devices being listed
- **block.<num>.name** − name of the target of the block device <num> (the same name for multiple entries if −−*backing* is present)
- **block.<num>.backingIndex** − when −−*backing* is present, matches up with the <backingStore> index listed in domain XML for backing files
- **block.<num>.path** − file source of block device <num>, if it is a local file or block device

- **block.<num>.rd.reqs** − number of read requests
- **block.<num>.rd.bytes** − number of read bytes
- **block.<num>.rd.times** − total time (ns) spent on reads
- **block.<num>.wr.reqs** − number of write requests
- **block.<num>.wr.bytes** − number of written bytes
- **block.<num>.wr.times** − total time (ns) spent on writes
- **block.<num>.fl.reqs** − total flush requests
- **block.<num>.fl.times** − total time (ns) spent on cache flushing
- **block.<num>.errors** − Xen only: the 'oo_req' value
- **block.<num>.allocation** − offset of highest written sector in bytes
- **block.<num>.capacity** − logical size of source file in bytes
- **block.<num>.physical** − physical size of source file in bytes
- **block.<num>.threshold** − threshold (in bytes) for delivering the VIR_DOMAIN_EVENT_ID_BLOCK_THRESHOLD event. See domblkthreshold.

−−*iothread* returns information about IOThreads on the running guest if supported by the hypervisor.

The "poll−max−ns" for each thread is the maximum nanoseconds to allow each polling interval to occur. A polling interval is a period of time allowed for a thread to process data before being the guest gives up its CPU quantum back to the host. A value set too small will not allow the IOThread to run long enough on a CPU to process data. A value set too high will consume too much CPU time per IOThread failing to allow other threads running on the CPU to get time. The polling interval is not available for statistical purposes.

- **iothread.count − maximum number of IOThreads in the subsequent list**
  as unsigned int. Each IOThread in the list will will use it's iothread_id value as the <id>. There may be fewer <id> entries than the iothread.count value if the polling values are not supported.
- **iothread.<id>.poll−max−ns** − maximum polling time in nanoseconds used by the <id> IOThread. A value of 0 (zero) indicates polling is disabled.
- **iothread.<id>.poll−grow** − polling time grow value. A value of 0 (zero) growth is managed by the hypervisor.
- **iothread.<id>.poll−shrink** − polling time shrink value. A value of (zero) indicates shrink is managed by hypervisor.

−−*memory* returns:

- **memory.bandwidth.monitor.count** − the number of memory bandwidth monitors for this domain
- **memory.bandwidth.monitor.<num>.name** − the name of monitor <num>
- **memory.bandwidth.monitor.<num>.vcpus** − the vcpu list of monitor <num>
- **memory.bandwidth.monitor.<num>.node.count − the number of memory**
  controller in monitor <num>
- **memory.bandwidth.monitor.<num>.node.<index>.id** − host allocated memory controller id for controller <index> of monitor <num>
- **memory.bandwidth.monitor.<num>.node.<index>.bytes.local** − the accumulative bytes consumed by @vcpus that passing through the memory controller in the same processor that the scheduled host CPU belongs to.

- **memory.bandwidth.monitor.<num>.node.<index>.bytes.total** − the total bytes consumed by @vcpus that passing through all memory controllers, either local or remote controller.

−−*dirtyrate* returns:

- **dirtyrate.calc_status** − the status of last memory dirty rate calculation, returned as number from virDomainDirtyRateStatus enum.
- **dirtyrate.calc_start_time** − the start time of last memory dirty rate calculation.
- **dirtyrate.calc_period** − the period of last memory dirty rate calculation.
- **dirtyrate.megabytes_per_second** − the calculated memory dirty rate in MiB/s.

Selecting a specific statistics groups doesn't guarantee that the daemon supports the selected group of stats. Flag −−*enforce* forces the command to fail if the daemon doesn't support the selected group.

When collecting stats libvirtd may wait for some time if there's already another job running on given domain for it to finish. This may cause unnecessary delay in delivering stats. Using −−*nowait* suppresses this behaviour. On the other hand some statistics might be missing for such domain.

domtime

Syntax:

```
domtime domain { [--now] [--pretty] [--sync] [--time time] }
```

Gets or sets the domain's system time. When run without any arguments (but *domain*), the current domain's system time is printed out. The −−*pretty* modifier can be used to print the time in more human readable form.

When −−*time* **time** is specified, the domain's time is not gotten but set instead. The −−*now* modifier acts like if it was an alias for −−*time* **$now**, which means it sets the time that is currently on the host virsh is running at. In both cases (setting and getting), time is in seconds relative to Epoch of 1970−01−01 in UTC. The −−*sync* modifies the set behavior a bit: The time passed is ignored, but the time to set is read from domain's RTC instead. Please note, that some hypervisors may require a guest agent to be configured in order to get or set the guest time.

domuuid

Syntax:

```
domuuid domain-name-or-id
```

Convert a domain name or id to domain UUID

domxml−from−native

Syntax:

```
domxml-from-native format config
```

Convert the file *config* in the native guest configuration format named by *format* to a domain XML format. For QEMU/KVM hypervisor, the *format* argument must be **qemu−argv**. For Xen hypervisor, the *format* argument may be **xen−xm**, **xen−xl**, or **xen−sxpr**. For LXC hypervisor, the *format* argument must be **lxc−tools**. For VMware/ESX hypervisor, the *format* argument must be **vmware−vmx**. For the Bhyve hypervisor, the *format* argument must be **bhyve−argv**.

domxml−to−native

Syntax:

```
domxml-to-native format { [--xml] xml | --domain domain-name-or-id-or-uuid }
```

Convert the file *xml* into domain XML format or convert an existing *--domain* to the native guest configuration format named by *format*. The*xml* and *--domain* ar guments are mutually exclusive. For the types of *format* argument, refer to **domxml-from-native**.

dump

Syntax:

```
dump domain corefilepath [--bypass-cache]
   { [--live] | [--crash] | [--reset] }
   [--verbose] [--memory-only] [--format string]
```

Dumps the core of a domain to a file for analysis. If *--live* is specified, the domain continues to run until the core dump is complete, rather than pausing up front. If *--crash* is specified, the domain is halted with a crashed status, rather than merely left in a paused state. If *--reset* is specified, the domain is reset after successful dump. Note, these three switches are mutually exclusive. If*--bypass-cac he* is specified, the save will avoid the file system cache, although this may slow down the operation. If *--memory-only* is specified, the file is elf file, and will only include domain's memory and cpu common register value. It is very useful if the domain uses host devices directly. *--format string* is used to specify the format of 'memory-only' dump, and*string* can be one of: elf, kdump-zlib(kdump-compressed format with zlib-compressed), kdump-lzo(kdump-compressed format with lzo-compressed), kdump-snappy(kdump-compressed format with snappy-compressed), win-dmp(Windows full crashdump format).

The progress may be monitored using **domjobinfo** virsh command and canceled with **domjobabort** command (sent by another virsh instance). Another option is to send SIGINT (usually with **Ctrl-C**) to the virsh process running **dump** command. *--verbose* displays the progress of dump.

NOTE: Some hypervisors may require the user to manually ensure proper permissions on file and path specified by argument *corefilepath*.

NOTE: Crash dump in a old kvmdump format is being obsolete and cannot be loaded and processed by crash utility since its version 6.1.0. A --memory-only option is required in order to produce valid ELF file which can be later processed by the crash utility.

dumpxml

Syntax:

```
dumpxml domain [--inactive] [--security-info] [--update-cpu] [--migratable]
```

Output the domain information as an XML dump to stdout, this format can be used by the **create** command. Additional options affecting the XML dump may be used. *--inactive* tells virsh to dump domain configuration that will be used on next start of the domain as opposed to the current domain configuration. Using *--security-info* will also include security sensitive information in the XML dump. *--update-cpu* updates domain CPU requirements according to host CPU. With *--migratable* one can request an XML that is suitable for migrations, i.e., compatible with older libvirt releases and possibly amended with internal run-time options. This option may automatically enable other options (*--update-cpu*, *--security-info*, ...) as necessary.

edit

Syntax:

```
edit domain
```

Edit the XML configuration file for a domain, which will affect the next boot of the guest.

This is equivalent to:

```
virsh dumpxml --inactive --security-info domain > domain.xml
vi domain.xml (or make changes with your other text editor)
virsh define domain.xml
```

except that it does some error checking.

The editor used can be supplied by the **$VISUAL** or **$EDITOR** environment variables, and defaults to **vi**.

emulatorpin

Syntax:

```
emulatorpin domain [cpulist] [[--live] [--config]  | [--current]]
```

Query or change the pinning of domain's emulator threads to host physical CPUs.

See **vcpupin** for *cpulist*.

If *--live* is specified, affect a running guest. If *--config* is specified, affect the next start of a persistent guest. If*--curr ent* is specified, it is equivalent to either *--live* or *--config*, depending on the current state of the guest. Both *--live* and *--config* flags may be given if *cpulist* is present, but *--current* is exclusive. If no flag is specified, behavior is different depending on hypervisor.

event

Syntax:

```
event {[domain] { event | --all } [--loop] [--timeout seconds] [--timestamp] |
```

Wait for a class of domain events to occur, and print appropriate details of events as they happen. The events can optionally be filtered by *domain*. Using*--list* as the only ar gument will provide a list of possible *event* values known by this client, although the connection might not allow registering for all these events. It is also possible to use*--all* instead of *e vent* to register for all possible event types at once.

By default, this command is one-shot, and returns success once an event occurs; you can send SIGINT (usually via **Ctrl-C**) to quit immediately. If*--timeout* is specified, the command gi ves up waiting for events after *seconds* have elapsed. With *--loop*, the command prints all events until a timeout or interrupt key.

When *--timestamp* is used, a human-readable timestamp will be printed before the event.

get-user-sshkeys

Syntax:

```
get-user-sshkeys domain user
```

Print SSH authorized keys for given *user* in the guest *domain*. Please note, that an entry in the file has internal structure as defined by *sshd(8)* and virsh/libvirt does handle keys as opaque strings, i.e. does not interpret them.

guest-agent-timeout

Syntax:

```
guest-agent-timeout domain [--timeout value]
```

Set how long to wait for a response from guest agent commands. By default, agent commands block forever waiting for a response. **value** must be a positive value (wait for given amount of seconds) or one of the following values:

- −2 − block forever waiting for a result (used when −−timeout is omitted),
- −1 − reset timeout to the default value (currently defined as 5 seconds in libvirt daemon),
- 0 − do not wait at all,

guestinfo

Syntax:

```
guestinfo domain [--user] [--os] [--timezone] [--hostname] [--filesystem]
   [--disk] [--interface]
```

Print information about the guest from the point of view of the guest agent. Note that this command requires a guest agent to be configured and running in the domain's guest OS.

When run without any arguments, this command prints all information types that are supported by the guest agent. You can limit the types of information that are returned by specifying one or more flags. If a requested information type is not supported, the processes will provide an exit code of 1. Available information types flags are −−*user*, −−*os*, −−*timezone*, −−*hostname*, −−*filesystem*, −−*disk* and −−*interface*.

Note that depending on the hypervisor type and the version of the guest agent running within the domain, not all of the following information may be returned.

When selecting the −−*user* information type, the following fields may be returned:

- **user.count** − the number of active users on this domain
- **user.<num>.name** − username of user <num>
- **user.<num>.domain** − domain of the user <num> (may only be present on certain guets types)
- **user.<num>.login−time** − the login time of user <num> in milliseconds since the epoch

−−*os* returns:

- **os.id** − a string identifying the operating system
- **os.name** − the name of the operating system
- **os.pretty−name** − a pretty name for the operating system
- **os.version** − the version of the operating system
- **os.version−id** − the version id of the operating system
- **os.kernel−release** − the release of the operating system kernel
- **os.kernel−version** − the version of the operating system kernel
- **os.machine** − the machine hardware name
- **os.variant** − a specific variant or edition of the operating system
- **os.variant−id** − the id for a specific variant or edition of the operating system

−−*timezone* returns:

- **timezone.name** − the name of the timezone
- **timezone.offset** − the offset to UTC in seconds

−−*hostname* returns:

- **hostname** − the hostname of the domain

−−*filesystem* returns:

- **fs.count** − the number of filesystems defined on this domain
- **fs.<num>.mountpoint** − the path to the mount point for filesystem <num>
- **fs.<num>.name** − device name in the guest (e.g. **sda1**) for filesystem <num>
- **fs.<num>.fstype** − the type of filesystem <num>
- **fs.<num>.total−bytes** − the total size of filesystem <num>
- **fs.<num>.used−bytes** − the number of bytes used in filesystem <num>
- **fs.<num>.disk.count** − the number of disks targeted by filesystem <num>
- **fs.<num>.disk.<num>.alias** − the device alias of disk <num> (e.g. sda)
- **fs.<num>.disk.<num>.serial** − the serial number of disk <num>
- **fs.<num>.disk.<num>.device** − the device node of disk <num>

−−*disk* returns:

- **disk.count** − the number of disks defined on this domain
- **disk.<num>.name** − device node (Linux) or device UNC (Windows)
- **disk.<num>.partition** − whether this is a partition or disk
- **disk.<num>.dependency.count** − the number of device dependencies
- **disk.<num>.dependency.<num>.name** − a dependency name
- **disk.<num>.serial** − optional disk serial number
- **disk.<num>.alias** − the device alias of the disk (e.g. sda)
- **disk.<num>.guest_alias** − optional alias assigned to the disk

−−*interface* returns: * **if.count** − the number of interfaces defined on this domain * **if.<num>.name** − name in the guest (e.g. **eth0**) for interface <num> * **if.<num>.hwaddr** − hardware address in the guest for interface <num> * **if.<num>.addr.count** − the number of IP addresses of interface <num> * **if.<num>.addr.<num1>.type** − the IP address type of addr <num1> (e.g. ipv4) * **if.<num>.addr.<num1>.addr** − the IP address of addr <num1> * **if.<num>.addr.<num1>.prefix** − the prefix of IP address of addr <num1>

guestvcpus

Syntax:

```
guestvcpus domain [[--enable] | [--disable]] [cpulist]
```

Query or change state of vCPUs from guest's point of view using the guest agent. When invoked without *cpulist* the guest is queried for available guest vCPUs, their state and possibility to be offlined.

If *cpulist* is provided then one of −−*enable* or −−*disable* must be provided too. The desired operation is then executed on the domain.

See **vcpupin** for information on *cpulist*.

iothreadadd

Syntax:

```
iothreadadd domain iothread_id [[--config] [--live] | [--current]]
```

Add a new IOThread to the domain using the specified *iothread_id*. If the*iothr ead_id* already exists, the command will fail. The *iothread_id* must be greater than zero.

If *--live* is specified, affect a running guest. If the guest is not running an error is returned. If *--config* is specified, affect the next start of a persistent guest. If *--current* is specified, it is equivalent to either *--live* or *--config*, depending on the current state of the guest.

iothreaddel

Syntax:

```
iothreaddel domain iothread_id [[--config] [--live] | [--current]]
```

Delete an IOThread from the domain using the specified *iothread_id*. If an IOThread is currently assigned to a disk resource such as via the **attach-disk** command, then the attempt to remove the IOThread will fail. If the *iothread_id* does not exist an error will occur.

If *--live* is specified, affect a running guest. If the guest is not running an error is returned. If *--config* is specified, affect the next start of a persistent guest. If *--current* is specified, it is equivalent to either *--live* or *--config*, depending on the current state of the guest.

iothreadinfo

Syntax:

```
iothreadinfo domain [[--live] [--config] | [--current]]
```

Display basic domain IOThreads information including the IOThread ID and the CPU Affinity for each IOThread.

If *--live* is specified, get the IOThreads data from the running guest. If the guest is not running, an error is returned. If*--config* is specified, get the IO Threads data from the next start of a persistent guest. If *--current* is specified or *--live* and *--config* are not specified, then get the IOThread data based on the current guest state, which can either be live or offline.

iothreadpin

Syntax:

```
iothreadpin domain iothread cpulist [[--live] [--config] | [--current]]
```

Change the pinning of a domain IOThread to host physical CPUs. In order to retrieve a list of all IOThreads, use **iothreadinfo**. To pin an *iothread* specify the *cpulist* desired for the IOThread ID as listed in the **iothreadinfo** output.

cpulist is a list of physical CPU numbers. Its syntax is a comma separated list and a special markup using '-' and '^' (ex. '0-4', '0-3,^2') can also be allowed. The '-' denotes the range and the '^' denotes exclusive. If you want to reset iothreadpin setting, that is, to pin an *iothread* to all physical cpus, simply specify 'r' as a *cpulist*.

If *--live* is specified, affect a running guest. If the guest is not running, an error is returned. If *--config* is specified, affect the next start of a persistent guest. If *--current* is specified, it is equivalent to either *--live* or *--config*, depending on the current state of the guest. Both *--live* and *--config* flags may be given if *cpulist* is present, but *--current* is exclusive. If no flag is specified, behavior is different depending on hypervisor.

Note: The expression is sequentially evaluated, so "0-15,^8" is identical to "9-14,0-7,15" but not identical to "^8,0-15".

iothreadset

Syntax:

```
iothreadset domain iothread_id [[--poll-max-ns ns] [--poll-grow factor]
```

```
[--poll-shrink divisor]]
[[--config] [--live] | [--current]]
```

Modifies an existing iothread of the domain using the specified *iothread_id*. The *--poll-max-ns* provides the maximum polling interval to be allowed for an IOThread in ns. If a 0 (zero) is provided, then polling for the IOThread is disabled. The *--poll-grow* is the factor by which the current polling time will be adjusted in order to reach the maximum polling time. If a 0 (zero) is provided, then the default factor will be used. The *--poll-shrink* is the quotient by which the current polling time will be reduced in order to get below the maximum polling interval. If a 0 (zero) is provided, then the default quotient will be used. The polling values are purely dynamic for a running guest. Saving, destroying, stopping, etc. the guest will result in the polling values returning to hypervisor defaults at the next start, restore, etc.

If *--live* is specified, affect a running guest. If the guest is not running an error is returned. If *--current* is specified or *--live* is not specified, then handle as if *--live* was specified. (Where "current" here means whatever the present guest state is: live or offline.)

managedsave
Syntax:

```
managedsave domain [--bypass-cache] [{--running | --paused}] [--verbose]
```

Save and destroy (stop) a running domain, so it can be restarted from the same state at a later time. When the virsh **start** command is next run for the domain, it will automatically be started from this saved state. If *--bypass-cache* is specified, the save will avoid the file system cache, although this may slow down the operation.

The progress may be monitored using **domjobinfo** virsh command and canceled with **domjobabort** command (sent by another virsh instance). Another option is to send SIGINT (usually with **Ctrl-C**) to the virsh process running **managedsave** command. *--verbose* displays the progress of save.

Normally, starting a managed save will decide between running or paused based on the state the domain was in when the save was done; passing either the *--running* or *--paused* flag will allow overriding which state the **start** should use.

The **dominfo** command can be used to query whether a domain currently has any managed save image.

managedsave-define
Syntax:

```
managedsave-define domain xml [{--running | --paused}]
```

Update the domain XML that will be used when *domain* is later started. The *xml* argument must be a file name containing the alternative XML, with changes only in the host-specific portions of the domain XML. For example, it can be used to change disk file paths.

The managed save image records whether the domain should be started to a running or paused state. Normally, this command does not alter the recorded state; passing either the *--running* or *--paused* flag will allow overriding which state the **start** should use.

managedsave-dumpxml
Syntax:

```
managedsave-dumpxml domain [--security-info]
```

Extract the domain XML that was in effect at the time the saved state file *file* was created with the **managedsave** command. Using *--security-info* will also include security sensitive information.

managedsave−edit
Syntax:

```
managedsave−edit domain [{−−running | −−paused}]
```

Edit the XML configuration associated with a saved state file of a *domain* was created by the **managedsave** command.

The managed save image records whether the domain should be started to a running or paused state. Normally, this command does not alter the recorded state; passing either the *−−running* or *−−paused* flag will allow overriding which state the **restore** should use.

This is equivalent to:

```
virsh managedsave−dumpxml domain−name > state−file.xml
vi state−file.xml (or make changes with your other text editor)
virsh managedsave−define domain−name state−file−xml
```

except that it does some error checking.

The editor used can be supplied by the **$VISUAL** or **$EDITOR** environment variables, and defaults to **vi**.

managedsave−remove
Syntax:

```
managedsave−remove domain
```

Remove the **managedsave** state file for a domain, if it exists. This ensures the domain will do a full boot the next time it is started.

maxvcpus
Syntax:

```
maxvcpus [type]
```

Provide the maximum number of virtual CPUs supported for a guest VM on this connection. If provided, the *type* parameter must be a valid type attribute for the <domain> element of XML.

memtune
Syntax:

```
memtune domain [−−hard−limit size] [−−soft−limit size] [−−swap−hard−limit size]
   [−−min−guarantee size] [[−−config] [−−live] | [−−current]]
```

Allows you to display or set the domain memory parameters. Without flags, the current settings are displayed; with a flag, the appropriate limit is adjusted if supported by the hypervisor. LXC and QEMU/KVM support *−−hard−limit*, *−−soft−limit*, and *−−swap−hard−limit*. *−−min−guarantee* is supported only by ESX hypervisor. Each of these limits are scaled integers (see **NOTES** above), with a default of kibibytes (blocks of 1024 bytes) if no suffix is present. Libvirt rounds up to the nearest kibibyte. Some hypervisors require a larger granularity than KiB, and requests that are not an even multiple will be rounded up. For example, vSphere/ESX rounds the parameter up to mebibytes (1024 kibibytes).

If *−−live* is specified, affect a running guest. If *−−config* is specified, affect the next start of a persistent guest. If*−−curr ent* is specified, it is equivalent to either *−−live* or *−−config*, depending on the current state of the guest. Both *−−live* and *−−config* flags may be given, but *−−current* is exclusive. If no flag is specified, behavior is different depending on hypervisor.

For QEMU/KVM, the parameters are applied to the QEMU process as a whole. Thus, when counting them, one needs to add up guest RAM, guest video RAM, and some memory overhead of QEMU itself. The last piece is hard to determine so one needs guess and try.

For LXC, the displayed hard_limit value is the current memory setting from the XML or the results from a **virsh setmem** command.

- *−−hard−limit*

  The maximum memory the guest can use.

- *−−soft−limit*

  The memory limit to enforce during memory contention.

- *−−swap−hard−limit*

  The maximum memory plus swap the guest can use. This has to be more than hard−limit value provided.

- *−−min−guarantee*

  The guaranteed minimum memory allocation for the guest.

Specifying −1 as a value for these limits is interpreted as unlimited.

metadata

Syntax:

```
metadata domain [[--live] [--config] | [--current]]
   [--edit] [uri] [key] [set] [--remove]
```

Show or modify custom XML metadata of a domain. The metadata is a user defined XML that allows storing arbitrary XML data in the domain definition. Multiple separate custom metadata pieces can be stored in the domain XML. The pieces are identified by a private XML namespace provided via the *uri* argument. (See also **desc** that works with textual metadata of a domain.)

Flags *−−live* or *−−config* select whether this command works on live or persistent definitions of the domain. If both *−−live* and *−−config* are specified, the *−−config* option takes precedence on getting the current description and both live configuration and config are updated while setting the description. *−−current* is exclusive and implied if none of these was specified.

Flag *−−remove* specifies that the metadata element specified by the *uri* argument should be removed rather than updated.

Flag *−−edit* specifies that an editor with the metadata identified by the *uri* argument should be opened and the contents saved back afterwards. Otherwise the new contents can be provided via the *set* argument.

When setting metadata via *−−edit* or *set* the *key* argument must be specified and is used to prefix the custom elements to bind them to the private namespace.

If neither of *−−edit* and *set* are specified the XML metadata corresponding to the *uri* namespace is displayed instead of being modified.

migrate

Syntax:

```
migrate [--live] [--offline] [--direct] [--p2p [--tunnelled]]
```

```
[--persistent] [--undefinesource] [--suspend] [--copy-storage-all]
[--copy-storage-inc] [--change-protection] [--unsafe] [--verbose]
[--rdma-pin-all] [--abort-on-error] [--postcopy] [--postcopy-after-precopy]
domain desturi [migrateuri] [graphicsuri] [listen-address] [dname]
[--timeout seconds [--timeout-suspend | --timeout-postcopy]]
[--xml file] [--migrate-disks disk-list] [--disks-port port]
[--compressed] [--comp-methods method-list]
[--comp-mt-level] [--comp-mt-threads] [--comp-mt-dthreads]
[--comp-xbzrle-cache] [--auto-converge] [auto-converge-initial]
[auto-converge-increment] [--persistent-xml file] [--tls]
[--postcopy-bandwidth bandwidth]
[--parallel [--parallel-connections connections]]
[--bandwidth bandwidth] [--tls-destination hostname]
[--disks-uri URI] [--copy-storage-synchronous-writes]
```

Migrate domain to another host. Add *--live* for live migration; <--p2p> for peer-2-peer migration; *--direct* for direct migration; or *--tunnelled* for tunnelled migration. *--offline* migrates domain definition without starting the domain on destination and without stopping it on source host. Offline migration may be used with inactive domains and it must be used with *--persistent* option.

--persistent leaves the domain persistent on destination host, *--undefinesource* undefines the domain on the source host, and *--suspend* leaves the domain paused on the destination host.

--copy-storage-all indicates migration with non-shared storage with full disk copy, *--copy-storage-inc* indicates migration with non-shared storage with incremental copy (same base image shared between source and destination). In both cases the disk images have to exist on destination host, the *--copy-storage-*... options only tell libvirt to transfer data from the images on source host to the images found at the same place on the destination host. By default only non-shared non-readonly images are transferred. Use *--migrate-disks* to explicitly specify a list of disk targets to transfer via the comma separated **disk-list** argument. With *--copy-storage-synchronous-writes* flag used the disk data migration will synchronously handle guest disk writes to both the original source and the destination to ensure that the disk migration converges at the price of possibly decreased burst performance.

--change-protection enforces that no incompatible configuration changes will be made to the domain while the migration is underway; this flag is implicitly enabled when supported by the hypervisor, but can be explicitly used to reject the migration if the hypervisor lacks change protection support.

--verbose displays the progress of migration.

--abort-on-error cancels the migration if a soft error (for example I/O error) happens during the migration.

--postcopy enables post-copy logic in migration, but does not actually start post-copy, i.e., migration is started in pre-copy mode. Once migration is running, the user may switch to post-copy using the **migrate-postcopy** command sent from another virsh instance or use *--postcopy-after-precopy* along with *--postcopy* to let libvirt automatically switch to post-copy after the first pass of pre-copy is finished. The maximum bandwidth consumed during the post-copy phase may be limited using *--postcopy-bandwidth*. The maximum bandwidth consumed during the pre-copy phase may be limited using *--bandwidth*.

--auto-converge forces convergence during live migration. The initial guest CPU throttling rate can be set with *auto-converge-initial*. If the initial throttling rate is not enough to ensure convergence, the rate is periodically increased by *auto-converge-increment*.

--rdma-pin-all can be used with RDMA migration (i.e., when *migrateuri* starts with rdma://) to tell the

hypervisor to pin all domain's memory at once before migration starts rather than letting it pin memory pages as needed. For QEMU/KVM this requires hard_limit memory tuning element (in the domain XML) to be used and set to the maximum memory configured for the domain plus any memory consumed by the QEMU process itself. Beware of setting the memory limit too high (and thus allowing the domain to lock most of the host's memory). Doing so may be dangerous to both the domain and the host itself since the host's kernel may run out of memory.

Note: Individual hypervisors usually do not support all possible types of migration. For example, QEMU does not support direct migration.

In some cases libvirt may refuse to migrate the domain because doing so may lead to potential problems such as data corruption, and thus the migration is considered unsafe. For QEMU domain, this may happen if the domain uses disks without explicitly setting cache mode to "none". Migrating such domains is unsafe unless the disk images are stored on coherent clustered filesystem, such as GFS2 or GPFS. If you are sure the migration is safe or you just do not care, use *−−unsafe* to force the migration.

dname is used for renaming the domain to new name during migration, which also usually can be omitted. Likewise, *−−xml* **file** is usually omitted, but can be used to supply an alternative XML file for use on the destination to supply a larger set of changes to any host−specific portions of the domain XML, such as accounting for naming differences between source and destination in accessing underlying storage. If *−−persistent* is enabled, *−−persistent−xml* **file** can be used to supply an alternative XML file which will be used as the persistent guest definition on the destination host.

−−timeout **seconds** tells virsh to run a specified action when live migration exceeds that many seconds. It can only be used with *−−live*. If*−−timeout−suspend* is specified, the domain will be suspended after the timeout and the migration will complete offline; this is the default if no *−−timeout−\`\` option is specified on the command line. When *−−timeout−postcopy*is used, virsh will switch migration from pre−cop y to post−copy upon timeout; migration has to be started with *−−postcopy* option for this to work.

−−compressed activates compression, the compression method is chosen with *−−comp−methods*. Supported methods are "mt" and "xbzrle" and can be used in any combination. When no methods are specified, a hypervisor default methods will be used. QEMU defaults to "xbzrle". Compression methods can be tuned further. *−−comp−mt−level* sets compression level. Values are in range from 0 to 9, where 1 is maximum speed and 9 is maximum compression. *−−comp−mt−threads* and *−−comp−mt−dthreads* set the number of compress threads on source and the number of decompress threads on target respectively. *−−comp−xbzrle−cache* sets size of page cache in bytes.

Providing *−−tls* causes the migration to use the host configured TLS setup (see migrate_tls_x509_cert_dir in /etc/libvirt/qemu.conf) in order to perform the migration of the domain. Usage requires proper TLS setup for both source and target. Normally the TLS certificate from the destination host must match the host's name for TLS verification to succeed. When the certificate does not match the destination hostname and the expected certificate's hostname is known, *−−tls−destination* can be used to pass the expected *hostname* when starting the migration.

−−parallel option will cause migration data to be sent over multiple parallel connections. The number of such connections can be set using *−−parallel−connections*. Parallel connections may help with saturating the network link between the source and the target and thus speeding up the migration.

Running migration can be canceled by interrupting virsh (usually using **Ctrl−C**) or by **domjobabort** command sent from another virsh instance.

The *desturi* and *migrateuri* parameters can be used to control which destination the migration uses. *desturi* is important for managed migration, but unused for direct migration; *migrateuri* is required for direct migration, but can usually be automatically determined for managed migration.

Note: The *desturi* parameter for normal migration and peer2peer migration has different semantics:

- normal migration: the *desturi* is an address of the target host as seen from the client machine.
- peer2peer migration: the *desturi* is an address of the target host as seen from the source machine.

In a special circumstance where you require a complete control of the connection and/or libvirt does not have network access to the remote side you can use a UNIX transport in the URI and specify a socket path in the query, for example with the qemu driver you could use this:

```
qemu+unix:///system?socket=/path/to/socket
```

When *migrateuri* is not specified, libvirt will automatically determine the hypervisor specific URI. Some hypervisors, including QEMU, have an optional "migration_host" configuration parameter (useful when the host has multiple network interfaces). If this is unspecified, libvirt determines a name by looking up the target host's configured hostname.

There are a few scenarios where specifying *migrateuri* may help:

- The configured hostname is incorrect, or DNS is broken. If a host has a hostname which will not resolve to match one of its public IP addresses, then libvirt will generate an incorrect URI. In this case *migrateuri* should be explicitly specified, using an IP address, or a correct hostname.
- The host has multiple network interfaces. If a host has multiple network interfaces, it might be desirable for the migration data stream to be sent over a specific interface for either security or performance reasons. In this case*migr ateuri* should be explicitly specified, using an IP address associated with the network to be used.
- The firewall restricts what ports are available. When libvirt generates a migration URI, it will pick a port number using hypervisor specific rules. Some hypervisors only require a single port to be open in the firewalls, while others require a whole range of port numbers. In the latter case *migrateuri* might be specified to choose a specific port number outside the default range in order to comply with local firewall policies.
- The *desturi* uses UNIX transport method. In this advanced case libvirt should not guess a *migrateuri* and it should be specified using UNIX socket path URI:

  ```
  unix:///path/to/socket
  ```

See *https://libvirt.org/migration.html#uris* for more details on migration URIs.

Optional *graphicsuri* overrides connection parameters used for automatically reconnecting a graphical clients at the end of migration. If omitted, libvirt will compute the parameters based on target host IP address. In case the client does not have a direct access to the network virtualization hosts are connected to and needs to connect through a proxy, *graphicsuri* may be used to specify the address the client should connect to. The URI is formed as follows:

```
protocol://hostname[:port]/[?parameters]
```

where protocol is either "spice" or "vnc" and parameters is a list of protocol specific parameters separated by '&'. Currently recognized parameters are "tlsPort" and "tlsSubject". For example,

```
spice://target.host.com:1234/?tlsPort=4567
```

Optional *listen−address* sets the listen address that hypervisor on the destination side should bind to for incoming migration. Both IPv4 and IPv6 addresses are accepted as well as hostnames (the resolving is done on destination). Some hypervisors do not support specifying the listen address and will return an error if this parameter is used. This parameter cannot be used if *desturi* uses UNIX transport method.

Optional *disks−port* sets the port that hypervisor on destination side should bind to for incoming disks traffic. Currently it is supported only by QEMU.

Optional *disks−uri* can also be specified (mutually exclusive with *disks−port*) to specify what the remote hypervisor should bind/connect to when migrating disks. This can be *tcp://address:port* to specify a listen address (which overrides *−−migrate−uri* and *−−listen−address* for the disk migration) and a port or *unix:///path/to/socket* in case you need the disk migration to happen over a UNIX socket with that specified path. In this case you need to make sure the same socket path is accessible to both source and destination hypervisors and connecting to the socket on the source (after hypervisor creates it on the destination) will actually connect to the destination. If you are using SELinux (at least on the source host) you need to make sure the socket on the source is accessible to libvirtd/QEMU for connection. Libvirt cannot change the context of the existing socket because it is different from the file representation of the socket and the context is chosen by its creator (usually by using *setsockcreatecon{,_raw}()* functions).

migrate−compcache
Syntax:

```
migrate−compcache domain [−−size bytes]
```

Sets and/or gets size of the cache (in bytes) used for compressing repeatedly transferred memory pages during live migration. When called without *size*, the command just prints current size of the compression cache. When *size* is specified, the hypervisor is asked to change compression cache to *size* bytes and then the current size is printed (the result may differ from the requested size due to rounding done by the hypervisor). The *size* option is supposed to be used while the domain is being live−migrated as a reaction to migration progress and increasing number of compression cache misses obtained from domjobinfo.

migrate−getmaxdowntime
Syntax:

```
migrate−getmaxdowntime domain
```

Get the maximum tolerable downtime for a domain which is being live−migrated to another host. This is the number of milliseconds the guest is allowed to be down at the end of live migration.

migrate−getspeed
Syntax:

```
migrate−getspeed domain [−−postcopy]
```

Get the maximum migration bandwidth (in MiB/s) for a domain. If the *−−postcopy* option is specified, the command will get the maximum bandwidth allowed during a post−copy migration phase.

migrate−postcopy
Syntax:

```
migrate−postcopy domain
```

Switch the current migration from pre−copy to post−copy. This is only supported for a migration started with *−−postcopy* option.

migrate−setmaxdowntime
Syntax:

```
migrate−setmaxdowntime domain downtime
```

Set maximum tolerable downtime for a domain which is being live−migrated to another host. The *downtime* is a number of milliseconds the guest is allowed to be down at the end of live migration.

migrate−setspeed

Syntax:

```
migrate−setspeed domain bandwidth [−−postcopy]
```

Set the maximum migration bandwidth (in MiB/s) for a domain which is being migrated to another host. *bandwidth* is interpreted as an unsigned long long value. Specifying a negative value results in an essentially unlimited value being provided to the hypervisor. The hypervisor can choose whether to reject the value or convert it to the maximum value allowed. If the *−−postcopy* option is specified, the command will set the maximum bandwidth allowed during a post−copy migration phase.

numatune

Syntax:

```
numatune domain [−−mode mode] [−−nodeset nodeset]
   [[−−config] [−−live] | [−−current]]
```

Set or get a domain's numa parameters, corresponding to the <numatune> element of domain XML. Without flags, the current settings are displayed.

mode can be one of `strict', `interleave', `preferred' and 'restrictive' or any valid number from the virDomainNumatuneMemMode enum in case the daemon supports it. For a running domain, the mode can't be changed, and the nodeset can be changed only if the domain was started with `restrictive' mode.

nodeset is a list of numa nodes used by the host for running the domain. Its syntax is a comma separated list, with '−' for ranges and '^' for excluding a node.

If *−−live* is specified, set scheduler information of a running guest. If *−−config* is specified, affect the next start of a persistent guest. If *−−current* is specified, it is equivalent to either *−−live* or *−−config*, depending on the current state of the guest.

For running guests in Linux hosts, the changes made in the domain's numa parameters does not imply that the guest memory will be moved to a different nodeset immediately. The memory migration depends on the guest activity, and the memory of an idle guest will remain in its previous nodeset for longer. The presence of VFIO devices will also lock parts of the guest memory in the same nodeset used to start the guest, regardless of nodeset changes.

perf

Syntax:

```
perf domain [−−enable eventSpec] [−−disable eventSpec]
   [[−−config] [−−live] | [−−current]]
```

Get the current perf events setting or enable/disable specific perf events for a guest domain.

Perf is a performance analyzing tool in Linux, and it can instrument CPU performance counters, tracepoints, kprobes, and uprobes (dynamic tracing). Perf supports a list of measurable events, and can measure events coming from different sources. For instance, some event are pure kernel counters, in this case they are called software events, including context−switches, minor−faults, etc.. Now dozens of events from different sources can be supported by perf.

Currently only QEMU/KVM supports this command. The *−−enable* and *−−disable* option combined with **eventSpec** can be used to enable or disable specific performance event. **eventSpec** is a string list of one or more events separated by commas. Valid event names are as follows:

Valid perf event names

- **cmt** − A PQos (Platform Qos) feature to monitor the usage of cache by applications running on the platform.
- **mbmt** − Provides a way to monitor the total system memory bandwidth between one level of cache and another.
- **mbml** − Provides a way to limit the amount of data (bytes/s) send through the memory controller on the socket.
- **cache_misses** − Provides the count of cache misses by applications running on the platform.
- **cache_references** − Provides the count of cache hits by applications running on th e platform.
- **instructions** − Provides the count of instructions executed by applications running on the platform.
- **cpu_cycles** − Provides the count of cpu cycles (total/elapsed). May be used with instructions in order to get a cycles per instruction.
- **branch_instructions** − Provides the count of branch instructions executed by applications running on the platform.
- **branch_misses** − Provides the count of branch misses executed by applications running on the platform.
- **bus_cycles** − Provides the count of bus cycles executed by applications running on the platform.
- **stalled_cycles_frontend** − Provides the count of stalled cpu cycles in the frontend of the instruction processor pipeline by applications running on the platform.
- **stalled_cycles_backend** − Provides the count of stalled cpu cycles in the backend of the instruction processor pipeline by applications running on the platform.
- **ref_cpu_cycles** − Provides the count of total cpu cycles not affected by CPU frequency scaling by applications running on the platform.
- **cpu_clock** − Provides the cpu clock time consumed by applications running on the platform.
- **task_clock** − Provides the task clock time consumed by applications running on the platform.
- **page_faults** − Provides the count of page faults by applications running on the platform.
- **context_switches** − Provides the count of context switches by applications running on the platform.
- **cpu_migrations** − Provides the count cpu migrations by applications running on the platform.
- **page_faults_min** − Provides the count minor page faults by applications running on the platform.
- **page_faults_maj** − Provides the count major page faults by applications running on the platform.
- **alignment_faults** − Provides the count alignment faults by applications running on the platform.
- **emulation_faults** − Provides the count emulation faults by applications running on the platform.

Note: The statistics can be retrieved using the **domstats** command using the −−*perf* flag.

If −−*live* is specified, affect a running guest. If −−*config* is specified, affect the next start of a persistent guest. If−−*curr ent* is specified, it is equivalent to either −−*live* or −−*config*, depending on the current state of the guest. Both −−*live* and −−*config* flags may be given, but −−*current* is exclusive. If no flag is specified, behavior is different depending on hypervisor.

reboot

Syntax:

```
reboot domain [−−mode MODE−LIST]
```

Reboot a domain. This acts just as if the domain had the **reboot** command run from the console. The command returns as soon as it has executed the reboot action, which may be significantly before the domain actually reboots.

The exact behavior of a domain when it reboots is set by the *on_reboot* parameter in the domain's XML definition.

By default the hypervisor will try to pick a suitable shutdown method. To specify an alternative method, the *--mode* parameter can specify a comma separated list which includes **acpi**, **agent**, **initctl**, **signal** and **paravirt**. The order in which drivers will try each mode is undefined, and not related to the order specified to virsh. For strict control over ordering, use a single mode at a time and repeat the command.

reset

Syntax:

```
reset domain
```

Reset a domain immediately without any guest shutdown. **reset** emulates the power reset button on a machine, where all guest hardware sees the RST line set and reinitializes internal state.

Note: Reset without any guest OS shutdown risks data loss.

restore

Syntax:

```
restore state-file [--bypass-cache] [--xml file]
   [{--running | --paused}]
```

Restores a domain from a **virsh save** state file. See *save* for more info.

If *--bypass-cache* is specified, the restore will avoid the file system cache, although this may slow down the operation.

--xml **file** is usually omitted, but can be used to supply an alternative XML file for use on the restored guest with changes only in the host-specific portions of the domain XML. For example, it can be used to account for file naming differences in underlying storage due to disk snapshots taken after the guest was saved.

Normally, restoring a saved image will use the state recorded in the save image to decide between running or paused; passing either the *--running* or *--paused* flag will allow overriding which state the domain should be started in.

Note: To avoid corrupting file system contents within the domain, you should not reuse the saved state file for a second **restore** unless you have also reverted all storage volumes back to the same contents as when the state file was created.

resume

Syntax:

```
resume domain
```

Moves a domain out of the suspended state. This will allow a previously suspended domain to now be eligible for scheduling by the underlying hypervisor.

save

Syntax:

```
save domain state-file [--bypass-cache] [--xml file]
   [{--running | --paused}] [--verbose]
```

Saves a running domain (RAM, but not disk state) to a state file so that it can be restored later. Once saved,

the domain will no longer be running on the system, thus the memory allocated for the domain will be free for other domains to use. **virsh r estore** restores from this state file. If *−−bypass−cache* is specified, the save will avoid the file system cache, although this may slow down the operation.

The progress may be monitored using **domjobinfo** virsh command and canceled with **domjobabort** command (sent by another virsh instance). Another option is to send SIGINT (usually with **Ctrl−C**) to the virsh process running **save** command. *−−verbose* displays the progress of save.

This is roughly equivalent to doing a hibernate on a running computer, with all the same limitations. Open network connections may be severed upon restore, as TCP timeouts may have expired.

−−xml **file** is usually omitted, but can be used to supply an alternative XML file for use on the restored guest with changes only in the host−specific portions of the domain XML. For example, it can be used to account for file naming differences that are planned to be made via disk snapshots of underlying storage after the guest is saved.

Normally, restoring a saved image will decide between running or paused based on the state the domain was in when the save was done; passing either the *−−running* or *−−paused* flag will allow overriding which state the **restore** should use.

Domain saved state files assume that disk images will be unchanged between the creation and restore point. For a more complete system restore point, where the disk state is saved alongside the memory state, see the **snapshot** family of commands.

save−image−define
Syntax:

```
save−image−define file xml [{−−running | −−paused}]
```

Update the domain XML that will be used when *file* is later used in the **restore** command. The *xml* argument must be a file name containing the alternative XML, with changes only in the host−specific portions of the domain XML. For example, it can be used to account for file naming differences resulting from creating disk snapshots of underlying storage after the guest was saved.

The save image records whether the domain should be restored to a running or paused state. Normally, this command does not alter the recorded state; passing either the *−−running* or *−−paused* flag will allow overriding which state the **restore** should use.

save−image−dumpxml
Syntax:

```
save−image−dumpxml file [−−security−info]
```

Extract the domain XML that was in effect at the time the saved state file *file* was created with the **save** command. Using*−−security−info* will also include security sensiti ve information.

save−image−edit
Syntax:

```
save−image−edit file [{−−running | −−paused}]
```

Edit the XML configuration associated with a saved state file *file* created by the **save** command.

The save image records whether the domain should be restored to a running or paused state. Normally, this command does not alter the recorded state; passing either the *−−running* or *−−paused* flag will allow overriding which state the **restore** should use.

This is equivalent to:

```
virsh save-image-dumpxml state-file > state-file.xml
vi state-file.xml (or make changes with your other text editor)
virsh save-image-define state-file state-file-xml
```

except that it does some error checking.

The editor used can be supplied by the **$VISUAL** or **$EDITOR** environment variables, and defaults to **vi**.

schedinfo

Syntax:

```
schedinfo domain [[--config] [--live] | [--current]] [[--set] parameter=value]
schedinfo [--weight number] [--cap number] domain
```

Allows you to show (and set) the domain scheduler parameters. The parameters available for each hypervisor are:

LXC (posix scheduler) : cpu_shares, vcpu_period, vcpu_quota

QEMU/KVM (posix scheduler): cpu_shares, vcpu_period, vcpu_quota, emulator_period, emulator_quota, global_period, global_quota, iothread_period, iothread_quota

Xen (credit scheduler): weight, cap

ESX (allocation scheduler): reservation, limit, shares

If *--live* is specified, set scheduler information of a running guest. If *--config* is specified, affect the next start of a persistent guest. If *--current* is specified, it is equivalent to either *--live* or *--config*, depending on the current state of the guest.

Note: The cpu_shares parameter has a valid value range of 2-262144.

Note: The weight and cap parameters are defined only for the XEN_CREDIT scheduler.

Note: The vcpu_period, emulator_period, and iothread_period parameters have a valid value range of 1000-1000000 or 0, and the vcpu_quota, emulator_quota, and iothread_quota parameters have a valid value range of 1000-17592186044415 or less than 0. The value 0 for either parameter is the same as not specifying that parameter.

screenshot

Syntax:

```
screenshot domain [imagefilepath] [--screen screenID]
```

Takes a screenshot of a current domain console and stores it into a file. Optionally, if the hypervisor supports more displays for a domain, *screenID* allows specifying which screen will be captured. It is the sequential number of screen. In case of multiple graphics cards, heads are enumerated before devices, e.g. having two graphics cards, both with four heads, screen ID 5 addresses the second head on the second card.

send-key

Syntax:

```
send-key domain [--codeset codeset] [--holdtime holdtime] keycode...
```

Parse the *keycode* sequence as keystrokes to send to *domain*. Each*k eycode* can either be a numeric value or a symbolic name from the corresponding codeset. If −−*holdtime* is given, each keystroke will be held for that many milliseconds. The default codeset is **linux**, but use of the −−*codeset* option allows other codesets to be chosen.

If multiple keycodes are specified, they are all sent simultaneously to the guest, and they may be received in random order. If you need distinct keypresses, you must use multiple send−key invocations.

- **linux**

  The numeric values are those defined by the Linux generic input event subsystem. The symbolic names match the corresponding Linux key constant macro names.

  See virkeycode−linux(7) and virkeyname−linux(7)

- **xt**

  The numeric values are those defined by the original XT keyboard controller. No symbolic names are provided

  See virkeycode−xt(7)

- **atset1**

  The numeric values are those defined by the AT keyboard controller, set 1 (aka XT compatible set). Extended keycoes from **atset1** may differ from extended keycodes in the **xt** codeset. No symbolic names are provided

  See virkeycode−atset1(7)

- **atset2**

  The numeric values are those defined by the AT keyboard controller, set 2. No symbolic names are provided

  See virkeycode−atset2(7)

- **atset3**

  The numeric values are those defined by the AT keyboard controller, set 3 (aka PS/2 compatible set). No symbolic names are provided

  See virkeycode−atset3(7)

- **os_x**

  The numeric values are those defined by the macOS keyboard input subsystem. The symbolic names match the corresponding macOS key constant macro names

  See virkeycode−osx(7) and virkeyname−osx(7)

- **xt_kbd**

  The numeric values are those defined by the Linux KBD device. These are a variant on the original XT codeset, but often with different encoding for extended keycodes. No symbolic names are provided.

  See virkeycode−xtkbd(7)

- **win32**

  The numeric values are those defined by the Win32 keyboard input subsystem. The symbolic names match the corresponding Win32 key constant macro names

  See virkeycode−win32(7) and virkeyname−win32(7)

- **usb**

  The numeric values are those defined by the USB HID specification for keyboard input. No symbolic names are provided

  See virkeycode−usb(7)

- **qnum**

  The numeric values are those defined by the QNUM extension for sending raw keycodes. These are a variant on the XT codeset, but extended keycodes have the low bit of the second byte set, instead of the high bit of the first byte. No symbolic names are provided.

  See virkeycode−qnum(7)

Examples:

```
# send three strokes 'k', 'e', 'y', using xt codeset. these
# are all pressed simultaneously and may be received by the guest
# in random order
virsh send−key dom −−codeset xt 37 18 21

# send one stroke 'right−ctrl+C'
virsh send−key dom KEY_RIGHTCTRL KEY_C

# send a tab, held for 1 second
virsh send−key −−holdtime 1000 0xf
```

send−process−signal

Syntax:

```
send−process−signal domain−id pid signame
```

Send a signal *signame* to the process identified by *pid* running in the virtual domain *domain−id*. The *pid* is a process ID in the virtual domain namespace.

The *signame* argument may be either an integer signal constant number, or one of the symbolic names:

```
"nop", "hup", "int", "quit", "ill",
"trap", "abrt", "bus", "fpe", "kill",
"usr1", "segv", "usr2", "pipe", "alrm",
"term", "stkflt", "chld", "cont", "stop",
"tstp", "ttin", "ttou", "urg", "xcpu",
"xfsz", "vtalrm", "prof", "winch", "poll",
"pwr", "sys", "rt0", "rt1", "rt2", "rt3",
"rt4", "rt5", "rt6", "rt7", "rt8", "rt9",
"rt10", "rt11", "rt12", "rt13", "rt14", "rt15",
"rt16", "rt17", "rt18", "rt19", "rt20", "rt21",
"rt22", "rt23", "rt24", "rt25", "rt26", "rt27",
```

```
"rt28", "rt29", "rt30", "rt31", "rt32"
```

The symbol name may optionally be prefixed with **sig** or **sig_** and may be in uppercase or lowercase.

Examples:

```
virsh send-process-signal myguest 1 15
virsh send-process-signal myguest 1 term
virsh send-process-signal myguest 1 sigterm
virsh send-process-signal myguest 1 SIG_HUP
```

set-lifecycle-action

Syntax:

```
set-lifecycle-action domain type action
   [[--config] [--live] | [--current]]
```

Set the lifecycle *action* for specified lifecycle *type*. The valid types are "poweroff", "reboot" and "crash", and for each of them valid *action* is one of "destroy", "restart", "rename-restart", "preserve". For *type* "crash", additional actions "coredump-destroy" and "coredump-restart" are supported.

set-user-password

Syntax:

```
set-user-password domain user password [--encrypted]
```

Set the password for the *user* account in the guest domain.

If *--encrypted* is specified, the password is assumed to be already encrypted by the method required by the guest OS.

For QEMU/KVM, this requires the guest agent to be configured and running.

set-user-sshkeys

Syntax:

```
set-user-sshkeys domain user [--file FILE] [{--reset | --remove}]
```

Append keys read from *FILE* into *user*'s SSH authorized keys file in the guest *domain*. In the*FILE* k eys must be on separate lines and each line must follow authorized keys format as defined by *sshd(8)*.

If *--reset* is specified, then the guest authorized keys file content is removed before appending new keys. As a special case, if *--reset* is provided and no *FILE* was provided then no new keys are added and the authorized keys file is cleared out.

If *--remove* is specified, then instead of adding any new keys then keys read from *FILE* are removed from the authorized keys file. It is not considered an error if the key does not exist in the file.

setmaxmem

Syntax:

```
setmaxmem domain size [[--config] [--live] | [--current]]
```

Change the maximum memory allocation limit for a guest domain. If *--live* is specified, affect a running guest. If*--config* is specified, af fect the next start of a persistent guest. If *--current* is specified, it is equivalent to either *--live* or *--config*, depending on the current state of the guest. Both *--live* and *--config* flags may be given, but *--current* is exclusive. If no flag is specified, behavior is different

depending on hypervisor.

Some hypervisors such as QEMU/KVM don't support live changes (especially increasing) of the maximum memory limit. Even persistent configuration changes might not be performed with some hypervisors/configuration (e.g. on NUMA enabled domains on QEMU). For complex configuration changes use command **edit** instead).

size is a scaled integer (see **NOTES** above); it defaults to kibibytes (blocks of 1024 bytes) unless you provide a suffix (and the older option name *--kilobytes* is available as a deprecated synonym) . Libvirt rounds up to the nearest kibibyte. Some hypervisors require a larger granularity than KiB, and requests that are not an even multiple will be rounded up. For example, vSphere/ESX rounds the parameter up to mebibytes (1024 kibibytes).

setmem

Syntax:

```
setmem domain size [[--config] [--live] | [--current]]
```

Change the memory allocation for a guest domain. If *--live* is specified, perform a memory balloon of a running guest. If *--config* is specified, affect the next start of a persistent guest. If *--current* is specified, it is equivalent to either *--live* or *--config*, depending on the current state of the guest. Both *--live* and *--config* flags may be given, but *--current* is exclusive. If no flag is specified, behavior is different depending on hypervisor.

size is a scaled integer (see **NOTES** above); it defaults to kibibytes (blocks of 1024 bytes) unless you provide a suffix (and the older option name *--kilobytes* is available as a deprecated synonym) . Libvirt rounds up to the nearest kibibyte. Some hypervisors require a larger granularity than KiB, and requests that are not an even multiple will be rounded up. For example, vSphere/ESX rounds the parameter up to mebibytes (1024 kibibytes).

For Xen, you can only adjust the memory of a running domain if the domain is paravirtualized or running the PV balloon driver.

For LXC, the value being set is the cgroups value for limit_in_bytes or the maximum amount of user memory (including file cache). When viewing memory inside the container, this is the /proc/meminfo "MemTotal" value. When viewing the value from the host, use the **virsh memtune** command. In order to view the current memory in use and the maximum value allowed to set memory, use the **virsh dominfo** command.

setvcpus

Syntax:

```
setvcpus domain count [--maximum] [[--config] [--live] | [--current]] [--guest
```

Change the number of virtual CPUs active in a guest domain. By default, this command works on active guest domains. To change the settings for an inactive guest domain, use the *--config* flag.

The *count* value may be limited by host, hypervisor, or a limit coming from the original description of the guest domain. For Xen, you can only adjust the virtual CPUs of a running domain if the domain is paravirtualized.

If the *--config* flag is specified, the change is made to the stored XML configuration for the guest domain, and will only take effect when the guest domain is next started.

If *--live* is specified, the guest domain must be active, and the change takes place immediately. Both the *--config* and *--live* flags may be specified together if supported by the hypervisor. If this command is run before the guest has finished booting, the guest may fail to process the change.

If *--current* is specified, it is equivalent to either *--live* or *--config*, depending on the current state of the guest.

When no flags are given, the *--live* flag is assumed and the guest domain must be active. In this situation it is up to the hypervisor whether the *--config* flag is also assumed, and therefore whether the XML configuration is adjusted to make the change persistent.

If *--guest* is specified, then the count of cpus is modified in the guest instead of the hypervisor. This flag is usable only for live domains and may require guest agent to be configured in the guest.

To allow adding vcpus to persistent definitions that can be later hotunplugged after the domain is booted it is necessary to specify the *--hotpluggable* flag. Vcpus added to live domains supporting vcpu unplug are automatically marked as hotpluggable.

The *--maximum* flag controls the maximum number of virtual cpus that can be hot-plugged the next time the domain is booted. As such, it must only be used with the *--config* flag, and not with the *--live* or the *--current* flag. Note that it may not be possible to change the maximum vcpu count if the processor topology is specified for the guest.

setvcpu

Syntax:

```
setvcpu domain vcpulist [--enable] | [--disable]
   [[--live] [--config] | [--current]]
```

Change state of individual vCPUs using hot(un)plug mechanism.

See **vcpupin** for information on format of *vcpulist*. Hypervisor drivers may require that *vcpulist* contains exactly vCPUs belonging to one hotpluggable entity. This is usually just a single vCPU but certain architectures such as ppc64 require a full core to be specified at once.

Note that hypervisors may refuse to disable certain vcpus such as vcpu 0 or others.

If *--live* is specified, affect a running domain. If *--config* is specified, affect the next startup of a persistent guest. If *--current* is specified, it is equivalent to either *--live* or *--config*, depending on the current state of the guest. This is the default. Both *--live* and *--config* flags may be given, but *--current* is exclusive.

shutdown

Syntax:

```
shutdown domain [--mode MODE-LIST]
```

Gracefully shuts down a domain. This coordinates with the domain OS to perform graceful shutdown, so there is no guarantee that it will succeed, and may take a variable length of time depending on what services must be shutdown in the domain.

The exact behavior of a domain when it shuts down is set by the *on_poweroff* parameter in the domain's XML definition.

If *domain* is transient, then the metadata of any snapshots and checkpoints will be lost once the guest stops running, but the underlying contents still exist, and a new domain with the same name and UUID can restore the snapshot metadata with **snapshot-create**, and the checkpoint metadata with **checkpoint-create**.

By default the hypervisor will try to pick a suitable shutdown method. To specify an alternative method, the *--mode* parameter can specify a comma separated list which includes **acpi**, **agent**, **initctl**, **signal** and

paravirt. The order in which drivers will try each mode is undefined, and not related to the order specified to virsh. For strict control over ordering, use a single mode at a time and repeat the command.

start

Syntax:

```
start domain-name-or-uuid [--console] [--paused]
   [--autodestroy] [--bypass-cache] [--force-boot]
   [--pass-fds N,M,...]
```

Start a (previously defined) inactive domain, either from the last **managedsave** state, or via a fresh boot if no managedsave state is present. The domain will be paused if the *--paused* option is used and supported by the driver; otherwise it will be running. If *--console* is requested, attach to the console after creation. If *--autodestroy* is requested, then the guest will be automatically destroyed when virsh closes its connection to libvirt, or otherwise exits. If*--bypass-cac he* is specified, and managedsave state exists, the restore will avoid the file system cache, although this may slow down the operation. If *--force-boot* is specified, then any managedsave state is discarded and a fresh boot occurs.

If *--pass-fds* is specified, the argument is a comma separated list of open file descriptors which should be pass on into the guest. The file descriptors will be re-numbered in the guest, starting from 3. This is only supported with container based virtualization.

suspend

Syntax:

```
suspend domain
```

Suspend a running domain. It is kept in memory but won't be scheduled anymore.

ttyconsole

Syntax:

```
ttyconsole domain
```

Output the device used for the TTY console of the domain. If the information is not available the processes will provide an exit code of 1.

undefine

Syntax:

```
undefine domain [--managed-save] [--snapshots-metadata]
   [--checkpoints-metadata] [--nvram] [--keep-nvram]
   [ {--storage volumes | --remove-all-storage
      [--delete-storage-volume-snapshots]} --wipe-storage]
```

Undefine a domain. If the domain is running, this converts it to a transient domain, without stopping it. If the domain is inactive, the domain configuration is removed.

The *--managed-save* flag guarantees that any managed save image (see the **managedsave** command) is also cleaned up. Without the flag, attempts to undefine a domain with a managed save image will fail.

The *--snapshots-metadata* flag guarantees that any snapshots (see the **snapshot-list** command) are also cleaned up when undefining an inactive domain. Without the flag, attempts to undefine an inactive domain with snapshot metadata will fail. If the domain is active, this flag is ignored.

The *--checkpoints-metadata* flag guarantees that any checkpoints (see the **checkpoint-list** command) are also cleaned up when undefining an inactive domain. Without the flag, attempts to undefine an inactive

domain with checkpoint metadata will fail. If the domain is active, this flag is ignored.

--nvram and *--keep-nvram* specify accordingly to delete or keep nvram (/domain/os/nvram/) file. If the domain has an nvram file and the flags are omitted, the undefine will fail.

The *--storage* flag takes a parameter **volumes**, which is a comma separated list of volume target names or source paths of storage volumes to be removed along with the undefined domain. Volumes can be undefined and thus removed only on inactive domains. Volume deletion is only attempted after the domain is undefined; if not all of the requested volumes could be deleted, the error message indicates what still remains behind. If a volume path is not found in the domain definition, it's treated as if the volume was successfully deleted. Only volumes managed by libvirt in storage pools can be removed this way. (See**domblklist** for list of target names associated to a domain). Example: --storage vda,/path/to/storage.img

The *--remove-all-storage* flag specifies that all of the domain's storage volumes should be deleted.

The *--delete-storage-volume-snapshots* (previously *--delete-snapshots*) flag specifies that any snapshots associated with the storage volume should be deleted as well. Requires the *--remove-all-storage* flag to be provided. Not all storage drivers support this option, presently only rbd. Using this when also removing volumes handled by a storage driver which does not support the flag will result in failure.

The flag *--wipe-storage* specifies that the storage volumes should be wiped before removal.

NOTE: For an inactive domain, the domain name or UUID must be used as the *domain*.

vcpucount

Syntax:

```
vcpucount domain  [{--maximum | --active}
   {--config | --live | --current}] [--guest]
```

Print information about the virtual cpu counts of the given *domain*. If no flags are specified, all possible counts are listed in a table; otherwise, the output is limited to just the numeric value requested. For historical reasons, the table lists the label "current" on the rows that can be queried in isolation via the *--active* flag, rather than relating to the *--current* flag.

--maximum requests information on the maximum cap of vcpus that a domain can add via **setvcpus**, while *--active* shows the current usage; these two flags cannot both be specified. *--config* requires a persistent guest and requests information regarding the next time the domain will be booted, *--live* requires a running domain and lists current values, and *--current* queries according to the current state of the domain (corresponding to *--live* if running, or *--config* if inactive); these three flags are mutually exclusive.

If *--guest* is specified, then the count of cpus is reported from the perspective of the guest. This flag is usable only for live domains and may require guest agent to be configured in the guest.

vcpuinfo

Syntax:

```
vcpuinfo domain [--pretty]
```

Returns basic information about the domain virtual CPUs, like the number of vCPUs, the running time, the affinity to physical processors.

With *--pretty*, cpu affinities are shown as ranges.

Example:

```
$ virsh vcpuinfo fedora
VCPU:           0
CPU:            0
State:          running
CPU time:       7,0s
CPU Affinity:   yyyy

VCPU:           1
CPU:            1
State:          running
CPU time:       0,7s
CPU Affinity:   yyyy
```

STATES

The State field displays the current operating state of a virtual CPU

- **offline**

  The virtual CPU is offline and not usable by the domain. This state is not supported by all hypervisors.

- **running**

  The virtual CPU is available to the domain and is operating.

- **blocked**

  The virtual CPU is available to the domain but is waiting for a resource. This state is not supported by all hypervisors, in which case *running* may be reported instead.

- **no state**

  The virtual CPU state could not be determined. This could happen if the hypervisor is newer than virsh.

- **N/A**

  There's no information about the virtual CPU state available. This can be the case if the domain is not running or the hypervisor does not report the virtual CPU state.

vcpupin

Syntax:

```
vcpupin domain [vcpu] [cpulist] [[--live] [--config] | [--current]]
```

Query or change the pinning of domain VCPUs to host physical CPUs. To pin a single *vcpu*, specify *cpulist*; otherwise, you can query one *vcpu* or omit *vcpu* to list all at once.

cpulist is a list of physical CPU numbers. Its syntax is a comma separated list and a special markup using '−' and '^' (ex. '0−4', '0−3,^2') can also be allowed. The '−' denotes the range and the '^' denotes exclusive. For pinning the *vcpu* to all physical cpus specify 'r' as a *cpulist*. If−−*live* is specified, af fect a running guest. If−−*config* is specified, af fect the next start of a persistent guest. If −−*current* is specified, it is equivalent to either −−*live* or −−*config*, depending on the current state of the guest. Both −−*live* and −−*config* flags may be given if *cpulist* is present, but −−*current* is exclusive. If no flag is specified, behavior is different depending on hypervisor.

Note: The expression is sequentially evaluated, so "0−15,^8" is identical to "9−14,0−7,15" but not identical to "^8,0−15".

vncdisplay
Syntax:

```
vncdisplay domain
```

Output the IP address and port number for the VNC display. If the information is not available the processes will provide an exit code of 1.

DEVICE COMMANDS

The following commands manipulate devices associated to domains. The *domain* can be specified as a short integer, a name or a full UUID. To better understand the values allowed as options for the command reading the documentation at *https://libvirt.org/formatdomain.html* on the format of the device sections to get the most accurate set of accepted values.

attach−device
Syntax:

```
attach−device domain FILE [[[−−live] [−−config] | [−−current]] | [−−persistent]
```

Attach a device to the domain, using a device definition in an XML file using a device definition element such as <disk> or <interface> as the top−level element. See the documentation at *https://libvirt.org/formatdomain.html#elementsDevices* to learn about libvirt XML format for a device. If *−−config* is specified the command alters the persistent guest configuration with the device attach taking effect the next time libvirt starts the domain. For cdrom and floppy devices, this command only replaces the media within an existing device; consider using **update−device** for this usage. For passthrough host devices, see also **nodedev−detach**, needed if the PCI device does not use managed mode.

If *−−live* is specified, affect a running domain. If *−−config* is specified, affect the next startup of a persistent guest. If *−−current* is specified, it is equivalent to either *−−live* or *−−config*, depending on the current state of the guest. Both *−−live* and *−−config* flags may be given, but *−−current* is exclusive. When no flag is specified legacy API is used whose behavior depends on the hypervisor driver.

For compatibility purposes, *−−persistent* behaves like *−−config* for an offline domain, and like *−−live −−config* for a running domain.

Note: using of partial device definition XML files may lead to unexpected results as some fields may be autogenerated and thus match devices other than expected.

attach−disk
Syntax:

```
attach−disk domain source target [[[−−live] [−−config] |
   [−−current]] | [−−persistent]] [−−targetbus bus]
   [−−driver driver] [−−subdriver subdriver] [−−iothread iothread]
   [−−cache cache] [−−io io] [−−type type] [−−alias alias]
   [−−mode mode] [−−sourcetype sourcetype]
   [−−source−protocol protocol] [−−source−host−name hostname:port]
   [−−source−host−transport transport] [−−source−host−socket socket]
   [−−serial serial] [−−wwn wwn] [−−rawio] [−−address address]
   [−−multifunction] [−−print−xml]
```

Attach a new disk device to the domain. *sour ce* is path for the files and devices unless *−−source−protocol* is specified, in which case *source* is the name of a network disk. *tar get* controls the bus or device under which the disk is exposed to the guest OS. It indicates the "logical" device name; the optional *targetbus* attribute specifies the type of disk device to emulate; possible values are driver specific, with typical values being *ide*, *scsi*, *virtio*, *xen*, *usb*, *sata*, or *sd*, if omitted, the bus type is inferred from the style of the device

name (e.g. a device named 'sda' will typically be exported using a SCSI bus). *driver* can be *file*, *tap* or *phy* for the Xen hypervisor depending on the kind of access; or *qemu* for the QEMU emulator. Further details to the driver can be passed using *subdriver*. For Xen *subdriver* can be *aio*, while for QEMU subdriver should match the format of the disk source, such as *raw* or *qcow2*. Hypervisor default will be used if *subdriver* is not specified. However, the default may not be correct, esp. for QEMU as for security reasons it is configured not to detect disk formats. *type* can indicate *lun*, *cdrom* or *floppy* as alternative to the disk default, although this use only replaces the media within the existing virtual cdrom or floppy device; consider using **update−device** for this usage instead. *alias* can set user supplied alias. *mode* can specify the two specific mode *readonly* or *shareable*. *sourcetype* can indicate the type of source (block|file|network) *cache* can be one of "default", "none", "writethrough", "writeback", "directsync" or "unsafe". *io* controls specific policies on I/O; QEMU guests support "threads", "native" and "io_uring". *iothread* is the number within the range of domain IOThreads to which this disk may be attached (QEMU only). *serial* is the serial of disk device. *wwn* is the wwn of disk device. *rawio* indicates the disk needs rawio capability. *address* is the address of disk device in the form of pci:domain.bus.slot.function, scsi:controller.bus.unit, ide:controller.bus.unit, usb:bus.port, sata:controller.bus.unit or ccw:cssid.ssid.devno. Virtio−ccw devices must have their cssid set to 0xfe. *multifunction* indicates specified pci address is a multifunction pci device address.

There is also support for using a network disk. As specified, the user can provide a *−−source−protocol* in which case the *source* parameter will be interpreted as the source name. *−−source−protocol* must be provided if the user intends to provide a network disk or host information. Host information can be provided using the tags *−−source−host−name*, *−−source−host−transport*, and *−−source−host−socket*, which respectively denote the name of the host, the host's transport method, and the socket that the host uses. *−−source−host−socket* and *−−source−host−name* cannot both be provided, and the user must provide a *−−source−host−transport* if they want to provide a *−−source−host−socket*. The*−−source−host−name* parameter supports host:port syntax if the user wants to provide a port as well.

If *−−print−xml* is specified, then the XML of the disk that would be attached is printed instead.

If *−−live* is specified, affect a running domain. If *−−config* is specified, affect the next startup of a persistent guest. If *−−current* is specified, it is equivalent to either *−−live* or *−−config*, depending on the current state of the guest. Both *−−live* and *−−config* flags may be given, but *−−current* is exclusive. When no flag is specified legacy API is used whose behavior depends on the hypervisor driver.

For compatibility purposes, *−−persistent* behaves like *−−config* for an offline domain, and like *−−live −−config* for a running domain. Likewise, *−−shareable* is an alias for *−−mode shareable*.

attach−interface

Syntax:

```
attach−interface domain type source [[[−−live]
   [−−config] | [−−current]] | [−−persistent]]
   [−−target target] [−−mac mac] [−−script script] [−−model model]
   [−−inbound average,peak,burst,floor] [−−outbound average,peak,burst]
   [−−alias alias] [−−managed] [−−print−xml]
   [−−source−mode mode]
```

Attach a new network interface to the domain.

type can be one of the:

network to indicate connection via a libvirt virtual network,

bridge to indicate connection via a bridge device on the host,

direct to indicate connection directly to one of the host's network interfaces or bridges,

hostdev to indicate connection using a passthrough of PCI device on the host,

vhostuser to indicate connection using a virtio transport protocol.

source indicates the source of the connection. The source depends on the type of the interface:

network name of the virtual network,

bridge the name of the bridge device,

direct the name of the host's interface or bridge,

hostdev the PCI address of the host's interface formatted as domain:bus:slot.function.

vhostuser the path to UNIX socket (control plane)

−−**target** is used to specify the tap/macvtap device to be used to connect the domain to the source. Names starting with 'vnet' are considered as auto−generated and are blanked out/regenerated each time the interface is attached.

−−**mac** specifies the MAC address of the network interface; if a MAC address is not given, a new address will be automatically generated (and stored in the persistent configuration if "−−config" is given on the command line).

−−**script** is used to specify a path to a custom script to be called while attaching to a bridge − this will be called instead of the default script not in addition to it. This is valid only for interfaces of *bridge* type and only for Xen domains.

−−**model** specifies the network device model to be presented to the domain.

alias can set user supplied alias.

−−**inbound** and −−**outbound** control the bandwidth of the interface. At least one from the*aver age*, *floor* pair must be specified. The other two *peak* and *burst* are optional, so "average,peak", "average,,burst", "average,,,floor", "average" and ",,,floor" are also legal. Values for *average*, *floor* and *peak* are expressed in kilobytes per second, while *burst* is expressed in kilobytes in a single burst at *peak* speed as described in the Network XML documentation at *https://libvirt.org/formatnetwork.html#elementQoS*.

−−**managed** is usable only for *hostdev* type and tells libvirt that the interface should be managed, which means detached and reattached from/to the host by libvirt.

−−**source−mode** is mandatory for *vhostuser* interface and accepts values *server* and *client* that control whether hypervisor waits for the other process to connect, or initiates connection, respectively.

If −−**print−xml** is specified, then the XML of the interface that would be attached is printed instead.

If −−**live** is specified, affect a running domain. If −−**config** is specified, affect the next startup of a persistent guest. If −−**current** is specified, affect the current domain state, which can either be live or offline. Both −−**live** and −−**config** flags may be given, but −−**current** is exclusive. When no flag is specified legacy API is used whose behavior depends on the hypervisor driver.

For compatibility purposes, −−**persistent** behaves like −−**config** for an offline domain, and like −−**live** −−**config** for a running domain.

Note: the optional target value is the name of a device to be created as the back−end on the node. If not provided a device named "vnetN" or "vifN" will be created automatically.

detach−device
Syntax:

```
detach−device domain FILE [[[−−live] [−−config] |
   [−−current]] | [−−persistent]]
```

Detach a device from the domain, takes the same kind of XML descriptions as command **attach−device**. For passthrough host devices, see also **nodedev−reattach**, needed if the device does not use managed mode.

Note: The supplied XML description of the device should be as specific as its definition in the domain XML. The set of attributes used to match the device are internal to the drivers. Using a partial definition, or attempting to detach a device that is not present in the domain XML, but shares some specific attributes with one that is present, may lead to unexpected results.

Quirk: Device unplug is asynchronous in most cases and requires guest cooperation. This means that it's up to the discretion of the guest to disallow or delay the unplug arbitrarily. As the libvirt API used in this command was designed as synchronous it returns success after some timeout even if the device was not unplugged yet to allow further interactions with the domain e.g. if the guest is unresponsive. Callers which need to make sure that the device was unplugged can use libvirt events (see virsh event) to be notified when the device is removed. Note that the event may arrive before the command returns.

If *−−live* is specified, affect a running domain. If *−−config* is specified, affect the next startup of a persistent guest. If *−−current* is specified, it is equivalent to either *−−live* or *−−config*, depending on the current state of the guest. Both *−−live* and *−−config* flags may be given, but *−−current* is exclusive. When no flag is specified legacy API is used whose behavior depends on the hypervisor driver.

For compatibility purposes, *−−persistent* behaves like *−−config* for an offline domain, and like *−−live −−config* for a running domain.

Note that older versions of virsh used *−−config* as an alias for *−−persistent*.

detach−device−alias
Syntax:

```
detach−device−alias domain alias [[[−−live] [−−config] | [−−current]]]
```

Detach a device with given *alias* from the *domain*. This command returns successfully after the unplug request was sent to the hypervisor. The actual removal of the device is notified asynchronously via libvirt events (see virsh event).

If *−−live* is specified, affect a running domain. If *−−config* is specified, affect the next startup of a persistent guest. If *−−current* is specified, it is equivalent to either *−−live* or *−−config*, depending on the current state of the guest. Both *−−live* and *−−config* flags may be given, but *−−current* is exclusive.

detach−disk
Syntax:

```
detach−disk domain target [[[−−live] [−−config] |
   [−−current]] | [−−persistent]] [−−print−xml]
```

Detach a disk device from a domain. The *target* is the device as seen from the domain.

If *--live* is specified, affect a running domain. If *--config* is specified, affect the next startup of a persistent guest. If *--current* is specified, it is equivalent to either *--live* or *--config*, depending on the current state of the guest. Both *--live* and *--config* flags may be given, but *--current* is exclusive. When no flag is specified legacy API is used whose behavior depends on the hypervisor driver.

For compatibility purposes, *--persistent* behaves like *--config* for an offline domain, and like *--live --config* for a running domain.

Note that older versions of virsh used *--config* as an alias for *--persistent*.

If **--print-xml** is specified, then the XML which would be used to detach the disk is printed instead.

Please see documentation for **detach-device** for known quirks.

detach-interface

Syntax:

```
detach-interface domain type [--mac mac]
   [[[--live] [--config] | [--current]] | [--persistent]]
```

Detach a network interface from a domain. *type* can be either *network* to indicate a ph ysical network device or *bridge* to indicate a bridge to a device. It is recommended to use the *mac* option to distinguish between the interfaces if more than one are present on the domain.

If *--live* is specified, affect a running domain. If *--config* is specified, affect the next startup of a persistent guest. If *--current* is specified, it is equivalent to either *--live* or *--config*, depending on the current state of the guest. Both *--live* and *--config* flags may be given, but *--current* is exclusive. When no flag is specified legacy API is used whose behavior depends on the hypervisor driver.

For compatibility purposes, *--persistent* behaves like *--config* for an offline domain, and like *--live --config* for a running domain.

Note that older versions of virsh used *--config* as an alias for *--persistent*.

Please see documentation for **detach-device** for known quirks.

update-device

Syntax:

```
update-device domain file [--force] [[[--live]
   [--config] | [--current]] | [--persistent]]
```

Update the characteristics of a device associated with *domain*, based on the device definition in an XML *file*. The*--for ce* option can be used to force device update, e.g., to eject a CD-ROM even if it is locked/mounted in the domain. See the documentation at *https://libvirt.org/formatdomain.html#elementsDevices* to learn about libvirt XML format for a device.

If *--live* is specified, affect a running domain. If *--config* is specified, affect the next startup of a persistent guest. If *--current* is specified, it is equivalent to either *--live* or *--config*, depending on the current state of the guest. Both *--live* and *--config* flags may be given, but *--current* is exclusive. Not specifying any flag is the same as specifying *--current*.

For compatibility purposes, *--persistent* behaves like *--config* for an offline domain, and like *--live --config* for a running domain.

Note that older versions of virsh used *--config* as an alias for *--persistent*.

Note: using of partial device definition XML files may lead to unexpected results as some fields may be autogenerated and thus match devices other than expected.

update−memory−device
Syntax:

```
update−memory−device domain [−−print−xml] [[−−alias alias] | [−−node node]]
  [[−−live] [−−config] | [−−current]]
  [−−requested−size size]
```

This command finds **<memory/>** device inside given *domain*, changes requested values and passes updated device XML to daemon. If *−−print−xml* is specified then the device is not changed, but the updated device XML is printed to stdout. If there are more than one **<memory/>** devices in *domain* use *−−alias* or *−−node* to select the desired one.

If *−−live* is specified, affect a running domain. If *−−config* is specified, affect the next startup of a persistent guest. If *−−current* is specified, it is equivalent to either *−−live* or *−−config*, depending on the current state of the guest. Both *−−live* and *−−config* flags may be given, but *−−current* is exclusive. Not specifying any flag is the same as specifying *−−current*.

If *−−requested−size* is specified then **<requested/>** under memory target is changed to requested *size* (as scaled integer, see **NOTES** above). It defaults to kibibytes if no suffix is provided. The option is valid only for **virtio−mem** memory device model.

change−media
Syntax:

```
change−media domain path [−−eject] [−−insert]
   [−−update] [source] [−−force] [[−−live] [−−config] |
   [−−current]] [−−print−xml] [−−block]
```

Change media of CDROM or floppy drive. *path* can be the fully−qualified path or the unique target name (<target dev='hdc'>) of the disk device. *source* specifies the path of the media to be inserted or updated. The *−−block* flag allows setting the backing type in case a block device is used as media for the CDROM or floppy drive instead of a file.

−−eject indicates the media will be ejected. *−−insert* indicates the media will be inserted. *source* must be specified. If the device has source (e.g. <source file='media'>), and *source* is not specified, *−−update* is equal to *−−eject*. If the device has no source, and *source* is specified, *−−update* is equal to *−−insert*. If the device has source, and *source* is specified, *−−update* behaves like combination of *−−eject* and *−−insert*. If none of *−−eject*, *−−insert*, and *−−update* is specified, *−−update* is used by default. The*−−for ce* option can be used to force media changing. If *−−live* is specified, alter live configuration of running guest. If *−−config* is specified, alter persistent configuration, effect observed on next startup of the guest. *−−curr ent* can be either or both of *live* and *config*, depends on the hypervisor's implementation. Both *−−live* and *−−config* flags may be given, but *−−current* is exclusive. If no flag is specified, behavior is different depending on hypervisor. If*−−print−xml* is specified, the XML that w ould be used to change media is printed instead of changing the media.

NODEDEV COMMANDS

The following commands manipulate host devices that are intended to be passed through to guest domains via <hostdev> elements in a domain's <devices> section. A node device key is generally specified by the bus name followed by its address, using underscores between all components, such as pci_0000_00_02_1, usb_1_5_3, or net_eth1_00_27_13_6a_fe_00. The **nodedev−list** gives the full list of host devices that are known to libvirt, although this includes devices that cannot be assigned to a guest (for example, attempting to detach the PCI device that controls the host's hard disk controller where the guest's disk images live could cause the host system to lock up or reboot).

For more information on node device definition see: *https://libvirt.org/formatnode.html*.

Passthrough devices cannot be simultaneously used by the host and its guest domains, nor by multiple active guests at once. If the <hostdev> description of a PCI device includes the attribute **managed='yes'**, and the hypervisor driver supports it, then the device is in managed mode, and attempts to use that passthrough device in an active guest will automatically behave as if **nodedev−detach** (guest start, device hot−plug) and **nodedev−reattach** (guest stop, device hot−unplug) were called at the right points. If a PCI device is not marked as managed, then it must manually be detached before guests can use it, and manually reattached to be returned to the host. Also, if a device is manually detached, then the host does not regain control of the device without a matching reattach, even if the guests use the device in managed mode.

nodedev−create
Syntax:

```
nodedev−create FILE
```

Create a device on the host node that can then be assigned to virtual machines. Normally, libvirt is able to automatically determine which host nodes are available for use, but this allows registration of host hardware that libvirt did not automatically detect. *file* contains xml for a top−le vel <device> description of a node device.

nodedev−destroy
Syntax:

```
nodedev−destroy device
```

Destroy (stop) a device on the host. *device* can be either device name or wwn pair in "wwnn,wwpn" format (only works for vHBA currently). Note that this makes libvirt quit managing a host device, and may even make that device unusable by the rest of the physical host until a reboot.

nodedev−define
Syntax:

```
nodedev−define FILE
```

Define an inactive persistent device or modify an existing persistent one from the XML *FILE*.

nodedev−undefine
Syntax:

```
nodedev−undefine device
```

Undefine the configuration for a persistent device. If the device is active, make it transient.

nodedev−start
Syntax:

```
nodedev−start device
```

Start a (previously defined) inactive device.

nodedev−detach
Syntax:

```
nodedev−detach nodedev [−−driver backend_driver]
```

Detach *nodedev* from the host, so that it can safely be used by guests via <hostdev> passthrough. This is reversed with **nodedev−reattach**, and is done automatically for managed devices.

Different backend drivers expect the device to be bound to different dummy devices. For example, QEMU's "kvm" backend driver (the default) expects the device to be bound to pci−stub, but its "vfio" backend driver expects the device to be bound to vfio−pci. The −−*driver* parameter can be used to specify the desired backend driver.

nodedev−dumpxml
Syntax:

```
nodedev−dumpxml device
```

Dump a <device> XML representation for the given node device, including such information as the device name, which bus owns the device, the vendor and product id, and any capabilities of the device usable by libvirt (such as whether device reset is supported). *device* can be either device name or wwn pair in "wwnn,wwpn" format (only works for HBA).

nodedev−info
Syntax:

```
nodedev−info device
```

Returns basic information about the *device* object.

nodedev−list
Syntax:

```
nodedev−list [−−cap capability] [−−tree] [−−inactive | −−all]
```

List all of the devices available on the node that are known by libvirt. *cap* is used to filter the list by capability types, the types must be separated by comma, e.g. −−cap pci,scsi. Valid capability types include 'system', 'pci', 'usb_device', 'usb', 'net', 'scsi_host', 'scsi_target', 'scsi', 'storage', 'fc_host', 'vports', 'scsi_generic', 'drm', 'mdev', 'mdev_types', 'ccw', 'css', 'ap_card', 'ap_queue', 'ap_matrix'. By default, only active devices are listed. −−*inactive* is used to list only inactive devices, and −*all* is used to list both active and inactive devices. If−−*tr ee* is used, the output is formatted in a tree representing parents of each node. −−*tree* is mutually exclusive with all other options.

nodedev−reattach
Syntax:

```
nodedev−reattach nodedev
```

Declare that *nodedev* is no longer in use by any guests, and that the host can resume normal use of the device. This is done automatically for PCI devices in managed mode and USB devices, but must be done explicitly to match any explicit **nodedev−detach**.

nodedev−reset
Syntax:

```
nodedev−reset nodedev
```

Trigger a device reset for *nodedev*, useful prior to transferring a node device between guest passthrough or the host. Libvirt will often do this action implicitly when required, but this command allows an explicit reset when needed.

nodedev−event
Syntax:

```
nodedev−event {[nodedev] event [−−loop] [−−timeout seconds] [−−timestamp] | −−
```

Wait for a class of node device events to occur, and print appropriate details of events as they happen. The events can optionally be filtered by *nodedev*. Using−−*list* as the only ar gument will provide a list of possible *event* values known by this client, although the connection might not allow registering for all these events.

By default, this command is one−shot, and returns success once an event occurs; you can send SIGINT (usually via **Ctrl−C**) to quit immediately. If−−*timeout* is specified, the command gi ves up waiting for events after *seconds* have elapsed. With −−*loop*, the command prints all events until a timeout or interrupt key.

When −−*timestamp* is used, a human−readable timestamp will be printed before the event.

nodedev−autostart

Syntax:

```
nodedev−autostart [−−disable] device
```

Configure a device to be automatically started when the host machine boots or the parent device becomes available. With −−*disable*, the device will be set to manual mode and will no longer be automatically started by the host. This command is only supported for persistently−defined mediated devices.

VIRTUAL NETWORK COMMANDS

The following commands manipulate networks. Libvirt has the capability to define virtual networks which can then be used by domains and linked to actual network devices. For more detailed information about this feature see the documentation at *https://libvirt.org/formatnetwork.html* . Many of the commands for virtual networks are similar to the ones used for domains, but the way to name a virtual network is either by its name or UUID.

net−autostart

Syntax:

```
net−autostart network [−−disable]
```

Configure a virtual network to be automatically started at boot. The −−*disable* option disable autostarting.

net−create

Syntax:

```
net−create file [−−validate]
```

Create a transient (temporary) virtual network from an XML *file* and instantiate (start) the network. See the documentation at *https://libvirt.org/formatnetwork.html* to get a description of the XML network format used by libvirt.

Optionally, the format of the input XML file can be validated against an internal RNG schema with −−*validate*.

net−define

Syntax:

```
net−define file [−−validate]
```

Define an inactive persistent virtual network or modify an existing persistent one from the XML *file*. Optionally, the format of the input XML file can be validated against an internal RNG schema with −−*validate*.

net−destroy
Syntax:

```
net−destroy network
```

Destroy (stop) a given transient or persistent virtual network specified by its name or UUID. This takes effect immediately.

net−dumpxml
Syntax:

```
net−dumpxml network [−−inactive]
```

Output the virtual network information as an XML dump to stdout. If *−−inactive* is specified, then physical functions are not expanded into their associated virtual functions.

net−edit
Syntax:

```
net−edit network
```

Edit the XML configuration file for a network.

This is equivalent to:

```
virsh net−dumpxml −−inactive network > network.xml
vi network.xml (or make changes with your other text editor)
virsh net−define network.xml
```

except that it does some error checking.

The editor used can be supplied by the **$VISUAL** or **$EDITOR** environment variables, and defaults to **vi**.

net−event
Syntax:

```
net−event {[network] event [−−loop] [−−timeout seconds] [−−timestamp] | −−list
```

Wait for a class of network events to occur, and print appropriate details of events as they happen. The events can optionally be filtered by *network*. Using*−−list* as the only ar gument will provide a list of possible *event* values known by this client, although the connection might not allow registering for all these events.

By default, this command is one−shot, and returns success once an event occurs; you can send SIGINT (usually via **Ctrl−C**) to quit immediately. If*−−timeout* is specified, the command gi ves up waiting for events after *seconds* have elapsed. With *−−loop*, the command prints all events until a timeout or interrupt key.

When *−−timestamp* is used, a human−readable timestamp will be printed before the event.

net−info
Syntax:

```
net−info network
```

Returns basic information about the *network* object.

net−list
Syntax:

```
net−list [−−inactive | −−all]
   { [−−table] | −−name | −−uuid }
   [−−persistent] [<−−transient>]
   [−−autostart] [<−−no−autostart>]
```

Returns the list of active networks, if *−−all* is specified this will also include defined but inactive networks, if *−−inactive* is specified only the inactive ones will be listed. You may also want to filter the returned networks by *−−persistent* to list the persistent ones, *−−transient* to list the transient ones, *−−autostart* to list the ones with autostart enabled, and *−−no−autostart* to list the ones with autostart disabled.

If *−−name* is specified, network names are printed instead of the table formatted one per line. If *−−uuid* is specified network's UUID's are printed instead of names. Flag *−−table* specifies that the legacy table−formatted output should be used. This is the default. All of these are mutually exclusive.

NOTE: When talking to older servers, this command is forced to use a series of API calls with an inherent race, where a pool might not be listed or might appear more than once if it changed state between calls while the list was being collected. Newer servers do not have this problem.

net−name
Syntax:

```
net−name network−UUID
```

Convert a network UUID to network name.

net−start
Syntax:

```
net−start network
```

Start a (previously defined) inactive network.

net−undefine
Syntax:

```
net−undefine network
```

Undefine the configuration for a persistent network. If the network is active, make it transient.

net−uuid
Syntax:

```
net−uuid network−name
```

Convert a network name to network UUID.

net−update
Syntax:

```
net−update network command section xml
   [−−parent−index index] [[−−live] [−−config] | [−−current]]
```

Update the given section of an existing network definition, with the changes optionally taking effect immediately, without needing to destroy and re−start the network.

command is one of "add−first", "add−last", "add" (a synonym for add−last), "delete", or "modify".

section is one of "bridge", "domain", "ip", "ip−dhcp−host", "ip−dhcp−range", "forward", "forward−interface", "forward−pf", "portgroup", "dns−host", "dns−txt", or "dns−srv", each section being named by a concatenation of the xml element hierarchy leading to the element being changed. For example, "ip−dhcp−host" will change a <host> element that is contained inside a <dhcp> element inside an <ip> element of the network.

xml is either the text of a complete xml element of the type being changed (e.g. "<host mac="00:11:22:33:44:55' ip='1.2.3.4'/>", or the name of a file that contains a complete xml element. Disambiguation is done by looking at the first character of the provided text − if the first character is "<", it is xml text, if the first character is not "<", it is the name of a file that contains the xml text to be used.

The −−*parent−index* option is used to specify which of several parent elements the requested element is in (0−based). For example, a dhcp <host> element could be in any one of multiple <ip> elements in the network; if a parent−index isn't provided, the "most appropriate" <ip> element will be selected (usually the only one that already has a <dhcp> element), but if −−*parent−index* is given, that particular instance of <ip> will get the modification.

If −−*live* is specified, affect a running network. If−−*config* is specified, af fect the next startup of a persistent network. If−−*curr ent* is specified, it is equivalent to either −−*live* or −−*config*, depending on the current state of the guest. Both −−*live* and −−*config* flags may be given, but −−*current* is exclusive. Not specifying any flag is the same as specifying −−*current*.

net−dhcp−leases

Syntax:

```
net−dhcp−leases network [mac]
```

Get a list of dhcp leases for all network interfaces connected to the given virtual *network* or limited output just for one interface if *mac* is specified.

NETWORK PORT COMMANDS

The following commands manipulate network ports. Libvirt virtual networks have ports created when a virtual machine has a virtual network interface added. In general there should be no need to use any of the commands here, since the hypervisor drivers run these commands are the right point in a virtual machine's lifecycle. They can be useful for debugging problems and / or recovering from bugs / stale state.

net−port−list

Syntax:

```
net−port−list { [−−table] | −−uuid } network
```

List all network ports recorded against the network.

If −−*uuid* is specified network ports' UUID's are printed instead of a table. Flag −−*table* specifies that the legacy table−formatted output should be used. This is the default. All of these are mutually exclusive.

net−port−create

Syntax:

```
net−port−create network file [−−validate]
```

Allocate a new network port reserving resources based on the port description. Optionally, the format of the input XML file can be validated against an internal RNG schema with −−*validate*.

net−port−dumpxml
Syntax:

```
net−port−dumpxml network port
```

Output the network port information as an XML dump to stdout.

net−port−delete
Syntax:

```
net−port−delete network port
```

Delete record of the network port and release its resources

INTERFACE COMMANDS

The following commands manipulate host interfaces. Often, these host interfaces can then be used by name within domain <interface> elements (such as a system−created bridge interface), but there is no requirement that host interfaces be tied to any particular guest configuration XML at all.

Many of the commands for host interfaces are similar to the ones used for domains, and the way to name an interface is either by its name or its MAC address. However, using a MAC address for an *iface* argument only works when that address is unique (if an interface and a bridge share the same MAC address, which is often the case, then using that MAC address results in an error due to ambiguity, and you must resort to a name instead).

iface−bridge
Syntax:

```
iface−bridge interface bridge [−−no−stp] [delay] [−−no−start]
```

Create a bridge device named *bridge*, and attach the existing network device *interface* to the new bridge. The new bridge defaults to starting immediately, with STP enabled and a delay of 0; these settings can be altered with *−−no−stp*, *−−no−start*, and an integer number of seconds for *delay*. All IP address configuration of *interface* will be moved to the new bridge device.

See also **iface−unbridge** for undoing this operation.

iface−define
Syntax:

```
iface−define file [−−validate]
```

Define an inactive persistent physical host interface or modify an existing persistent one from the XML *file*. Optionally, the format of the input XML file can be validated against an internal RNG schema with *−−validate*.

iface−destroy
Syntax:

```
iface−destroy interface
```

Destroy (stop) a given host interface, such as by running "if−down" to disable that interface from active use. This takes effect immediately.

iface−dumpxml
Syntax:

```
iface−dumpxml interface [−−inactive]
```

Output the host interface information as an XML dump to stdout. If *--inactive* is specified, then the output reflects the persistent state of the interface that will be used the next time it is started.

iface-edit
Syntax:

```
iface-edit interface
```

Edit the XML configuration file for a host interface.

This is equivalent to:

```
virsh iface-dumpxml iface > iface.xml
vi iface.xml (or make changes with your other text editor)
virsh iface-define iface.xml
```

except that it does some error checking.

The editor used can be supplied by the **$VISUAL** or **$EDITOR** environment variables, and defaults to **vi**.

iface-list
Syntax:

```
iface-list [--inactive | --all]
```

Returns the list of active host interfaces. If*--all* is specified this will also include defined b ut inactive interfaces. If*--inactive* is specified only the inacti ve ones will be listed.

iface-name
Syntax:

```
iface-name interface
```

Convert a host interface MAC to interface name, if the MAC address is unique among the host's interfaces.

interface specifies the interface MAC address.

iface-mac
Syntax:

```
iface-mac interface
```

Convert a host interface name to MAC address.

interface specifies the interface name.

iface-start
Syntax:

```
iface-start interface
```

Start a (previously defined) host interface, such as by running "if-up".

iface-unbridge
Syntax:

```
iface-unbridge bridge [--no-start]
```

Tear down a bridge device named *bridge*, releasing its underlying interface back to normal usage, and moving all IP address configuration from the bridge device to the underlying device. The underlying interface is restarted unless *−−no−start* is present; this flag is present for symmetry, but generally not recommended.

See also **iface−bridge** for creating a bridge.

iface−undefine
Syntax:

```
iface−undefine interface
```

Undefine the configuration for an inactive host interface.

iface−begin
Syntax:

```
iface−begin
```

Create a snapshot of current host interface settings, which can later be committed (*iface−commit*) or restored (*iface−rollback*). If a snapshot already exists, then this command will fail until the previous snapshot has been committed or restored. Undefined behavior results if any external changes are made to host interfaces outside of the libvirt API between the beginning of a snapshot and its eventual commit or rollback.

iface−commit
Syntax:

```
iface−commit
```

Declare all changes since the last *iface−begin* as working, and delete the rollback point. If no interface snapshot has already been started, then this command will fail.

iface−rollback
Syntax:

```
iface−rollback
```

Revert all host interface settings back to the state recorded in the last *iface−begin*. If no interface snapshot has already been started, then this command will fail. Rebooting the host also serves as an implicit rollback point.

STORAGE POOL COMMANDS

The following commands manipulate storage pools. Libvirt has the capability to manage various storage solutions, including files, raw partitions, and domain−specific formats, used to provide the storage volumes visible as devices within virtual machines. For more detailed information about this feature, see the documentation at *https://libvirt.org/formatstorage.html* . Many of the commands for pools are similar to the ones used for domains.

find−storage−pool−sources
Syntax:

```
find−storage−pool−sources type [srcSpec]
```

Returns XML describing all possible available storage pool sources that could be used to create or define a storage pool of a given *type*. If *srcSpec* is provided, it is a file that contains XML to further restrict the query for pools.

Not all storage pools support discovery in this manner. Furthermore, for those that do support discovery,

only specific XML elements are required in order to return valid data, while other elements and even attributes of some elements are ignored since they are not necessary to find the pool based on the search criteria. The following lists the supported *type* options and the expected minimal XML elements used to perform the search.

For a "netfs" or "gluster" pool, the minimal expected XML required is the <host> element with a "name" attribute describing the IP address or hostname to be used to find the pool. The "port" attribute will be ignored as will any other provided XML elements in *srcSpec*.

For a "logical" pool, the contents of the *srcSpec* file are ignored, although if provided the file must at least exist.

For an "iscsi" or "iscsi−direct" pool, the minimal expect XML required is the <host> element with a "name" attribute describing the IP address or hostname to be used to find the pool (the iSCSI server address). Optionally, the "port" attribute may be provided, although it will default to 3260. Optionally, an <initiator> XML element with a "name" attribute may be provided to further restrict the iSCSI target search to a specific initiator for multi−iqn iSCSI storage pools.

find−pool−sources−as
Syntax:

```
find−storage−pool−sources−as type [host] [port] [initiator]
```

Rather than providing *srcSpec* XML file for **find−storage−pool−sources** use this command option in order to have virsh generate the query XML file using the optional arguments. The command will return the same output XML as **find−storage−pool−sources**.

Use *host* to describe a specific host to use for networked storage, such as netfs, gluster, and iscsi *type* pools.

Use *port* to further restrict which networked port to utilize for the connection if required by the specific storage backend, such as iscsi.

Use *initiator* to further restrict the iscsi *type* pool searches to specific target initiators.

pool−autostart
Syntax:

```
pool−autostart pool−or−uuid [−−disable]
```

Configure whether *pool* should automatically start at boot.

pool−build
Syntax:

```
pool−build pool−or−uuid [−−overwrite] [−−no−overwrite]
```

Build a given pool.

Options −−*overwrite* and −−*no−overwrite* can only be used for **pool−build** a filesystem, disk, or logical pool.

For a file system pool if neither flag is specified, then **pool−build** just makes the target path directory and no attempt to run mkfs on the target volume device. If −−*no−overwrite* is specified, it probes to determine if a filesystem already exists on the target device, returning an error if one exists or using mkfs to format the target device if not. If −−*overwrite* is specified, mkfs is always executed and any existing data on the target device is overwritten unconditionally.

For a disk pool, if neither of them is specified or *--no-overwrite* is specified, **pool-build** will check the target volume device for existing filesystems or partitions before attempting to write a new label on the target volume device. If the target volume device already has a label, the command will fail. If *--overwrite* is specified, then no check will be made on the target volume device prior to writing a new label. Writing of the label uses the pool source format type or "dos" if not specified.

For a logical pool, if neither of them is specified or *--no-overwrite* is specified, **pool-build** will check the target volume devices for existing filesystems or partitions before attempting to initialize and format each device for usage by the logical pool. If any target volume device already has a label, the command will fail. If *--overwrite* is specified, then no check will be made on the target volume devices prior to initializing and formatting each device. Once all the target volume devices are properly formatted via pvcreate, the volume group will be created using all the devices.

pool-create
Syntax:

```
pool-create file [--build] [[--overwrite] | [--no-overwrite]]
```

Create and start a pool object from the XML *file*.

[*--build*] [[*--overwrite*] | [*--no-overwrite*]] perform a **pool-build** after creation in order to remove the need for a follow-up command to build the pool. The *--overwrite* and *--no-overwrite* flags follow the same rules as **pool-build**. If just *--build* is provided, then **pool-build** is called with no flags.

pool-create-as
Syntax:

```
pool-create-as name type
   [--source-host hostname] [--source-path path] [--source-dev path]
   [--source-name name] [--target path] [--source-format format]
   [--source-initiator initiator-iqn]
   [--auth-type authtype --auth-username username
   [--secret-usage usage | --secret-uuid uuid]]
   [--source-protocol-ver ver]
   [[--adapter-name name] | [--adapter-wwnn wwnn --adapter-wwpn wwpn]
   [--adapter-parent parent |
   --adapter-parent-wwnn parent_wwnn adapter-parent-wwpn parent_wwpn |
   --adapter-parent-fabric-wwn parent_fabric_wwn]]
   [--build] [[--overwrite] | [--no-overwrite]] [--print-xml]
```

Create and start a pool object *name* from the raw parameters. If *--print-xml* is specified, then print the XML of the pool object without creating the pool. Otherwise, the pool has the specified *type*. When using **pool-create-as** for a pool of *type* "disk", the existing partitions found on the *--source-dev path* will be used to populate the disk pool. Therefore, it is suggested to use **pool-define-as** and **pool-build** with the *--overwrite* in order to properly initialize the disk pool.

[*--source-host hostname*] provides the source hostname for pools backed by storage from a remote server (pool types netfs, iscsi, rbd, sheepdog, gluster).

[*--source-path path*] provides the source directory path for pools backed by directories (pool type dir).

[*--source-dev path*] provides the source path for pools backed by physical devices (pool types fs, logical, disk, iscsi, zfs).

[*--source-name name*] provides the source name for pools backed by storage from a named element (pool types logical, rbd, sheepdog, gluster).

[*--target path*] is the path for the mapping of the storage pool into the host file system.

[*--source-format format*] provides information about the format of the pool (pool types fs, netfs, disk, logical).

[*--source-initiator initiator-iqn*] provides the initiator iqn for iscsi connection of the pool (pool type iscsi-direct).

[*--auth-type authtype --auth-username username* [*--secret-usage usage* | *--secret-uuid uuid*]] provides the elements required to generate authentication credentials for the storage pool. The *authtype* is either chap for iscsi *type* pools or ceph for rbd *type* pools. Either the secret *usage* or *uuid* value may be provided, but not both.

[*--source-protocol-ver ver*] provides the NFS protocol version number used to contact the server's NFS service via nfs mount option 'nfsvers=n'. It is expect the *ver* value is an unsigned integer.

[*--adapter-name name*] defines the scsi_hostN adapter name to be used for the scsi_host adapter type pool.

[*--adapter-wwnn wwnn --adapter-wwpn wwpn* [*--adapter-parent parent* | *--adapter-parent-wwnn parent_wwnn adapter-parent-wwpn parent_wwpn* | *--adapter-parent-fabric-wwn parent_fabric_wwn*]] defines the wwnn and wwpn to be used for the fc_host adapter type pool. Optionally provide the parent scsi_hostN node device to be used for the vHBA either by parent name, parent_wwnn and parent_wwpn, or parent_fabric_wwn. The parent name could change between reboots if the hardware environment changes, so providing the parent_wwnn and parent_wwpn ensure usage of the same physical HBA even if the scsi_hostN node device changes. Usage of the parent_fabric_wwn allows a bit more flexibility to choose an HBA on the same storage fabric in order to define the pool.

[*--build*] [[*--overwrite*] | [*--no-overwrite*]] perform a **pool-build** after creation in order to remove the need for a follow-up command to build the pool. The *--overwrite* and *--no-overwrite* flags follow the same rules as **pool-build**. If just *--build* is provided, then **pool-build** is called with no flags.

For a "logical" pool only [*--name*] needs to be provided. The [*--source-name*] if provided must match the Volume Group name. If not provided, one will be generated using the [*--name*]. If provided the [*--target*] is ignored and a target source is generated using the [*--source-name*] (or as generated from the [*--name*]).

pool-define
Syntax:

```
pool-define file [--validate]
```

Define an inactive persistent storage pool or modify an existing persistent one from the XML *file*. Optionally, the format of the input XML file can be validated against an internal RNG schema with *--validate*.

pool-define-as
Syntax:

```
pool-define-as name type
   [--source-host hostname] [--source-path path] [--source-dev path]
   [*--source-name name*] [*--target path*] [*--source-format format*]
   [--source-initiator initiator-iqn]
   [*--auth-type authtype* *--auth-username username*
   [*--secret-usage usage* | *--secret-uuid uuid*]]
   [*--source-protocol-ver ver*]
   [[*--adapter-name name*] | [*--adapter-wwnn* *--adapter-wwpn*]
```

```
[*--adapter-parent parent*]] [*--print-xml*]
```

Create, but do not start, a pool object *name* from the raw parameters. If *--print-xml* is specified, then print the XML of the pool object without defining the pool. Otherwise, the pool has the specified *type*.

Use the same arguments as **pool-create-as**, except for the *--build*, *--overwrite*, and *--no-overwrite* options.

pool-destroy
Syntax:

```
pool-destroy pool-or-uuid
```

Destroy (stop) a given *pool* object. Libvirt will no longer manage the storage described by the pool object, but the raw data contained in the pool is not changed, and can be later recovered with **pool-create**.

pool-delete
Syntax:

```
pool-delete pool-or-uuid
```

Destroy the resources used by a given *pool* object. This operation is non-recoverable. The*pool* object will still exist after this command, ready for the creation of new storage volumes.

pool-dumpxml
Syntax:

```
pool-dumpxml [--inactive] pool-or-uuid
```

Returns the XML information about the *pool* object. *--inactive* tells virsh to dump pool configuration that will be used on next start of the pool as opposed to the current pool configuration.

pool-edit
Syntax:

```
pool-edit pool-or-uuid
```

Edit the XML configuration file for a storage pool.

This is equivalent to:

```
virsh pool-dumpxml pool > pool.xml
vi pool.xml (or make changes with your other text editor)
virsh pool-define pool.xml
```

except that it does some error checking.

The editor used can be supplied by the **$VISUAL** or **$EDITOR** environment variables, and defaults to **vi**.

pool-info
Syntax:

```
pool-info [--bytes] pool-or-uuid
```

Returns basic information about the *pool* object. If *--bytes* is specified the sizes of basic info are not converted to human friendly units.

pool−list
Syntax:

```
pool−list [−−inactive] [−−all]
   [−−persistent] [−−transient]
   [−−autostart] [−−no−autostart]
   [[−−details] [−−uuid]
   [−−name] [<type>]
```

List pool objects known to libvirt. By default, only active pools are listed; −−*inactive* lists just the inactive pools, and −−*all* lists all pools.

In addition, there are several sets of filtering flags. −−*persistent* is to list the persistent pools, −−*transient* is to list the transient pools. −−*autostart* lists the autostarting pools, −−*no−autostart* lists the pools with autostarting disabled. If −−*uuid* is specified only pool's UUIDs are printed. If −−*name* is specified only pool's names are printed. If both −−*name* and −−*uuid* are specified, pool's UUID and names are printed side by side without any header. Option −−*details* is mutually exclusive with options −−*uuid* and −−*name*.

You may also want to list pools with specified types using *type*, the pool types must be separated by comma, e.g. −−type dir,disk. The valid pool types include 'dir', 'fs', 'netfs', 'logical', 'disk', 'iscsi', 'scsi', 'mpath', 'rbd', 'sheepdog', 'gluster', 'zfs', 'vstorage' and 'iscsi−direct'.

The −−*details* option instructs virsh to additionally display pool persistence and capacity related information where available.

NOTE: When talking to older servers, this command is forced to use a series of API calls with an inherent race, where a pool might not be listed or might appear more than once if it changed state between calls while the list was being collected. Newer servers do not have this problem.

pool−name
Syntax:

```
pool−name uuid
```

Convert the *uuid* to a pool name.

pool−refresh
Syntax:

```
pool−refresh pool−or−uuid
```

Refresh the list of volumes contained in *pool*.

pool−start
Syntax:

```
pool−start pool−or−uuid [−−build] [[−−overwrite] | [−−no−overwrite]]
```

Start the storage *pool*, which is previously defined but inactive.

[−−*build*] [[−−*overwrite*] | [−−*no−overwrite*]] perform a **pool−build** prior to **pool−start** to ensure the pool environment is in an expected state rather than needing to run the build command prior to startup. The −−*overwrite* and −−*no−overwrite* flags follow the same rules as **pool−build**. If just −−*build* is provided, then **pool−build** is called with no flags.

Note: A storage pool that relies on remote resources such as an "iscsi" or a (v)HBA backed "scsi" pool may

need to be refreshed multiple times in order to have all the volumes detected (see **pool−refresh**). This is because the corresponding volume devices may not be present in the host's filesystem during the initial pool startup or the current refresh attempt. The number of refresh retries is dependent upon the network connection and the time the host takes to export the corresponding devices.

pool−undefine
Syntax:

```
pool−undefine pool−or−uuid
```

Undefine the configuration for an inactive *pool*.

pool−uuid
Syntax:

```
pool−uuid pool
```

Returns the UUID of the named *pool*.

pool−event
Syntax:

```
pool−event {[pool] event [−−loop] [−−timeout seconds] [−−timestamp] | −−list}
```

Wait for a class of storage pool events to occur, and print appropriate details of events as they happen. The events can optionally be filtered by *pool*. Using−−*list* as the only ar gument will provide a list of possible *event* values known by this client, although the connection might not allow registering for all these events.

By default, this command is one−shot, and returns success once an event occurs; you can send SIGINT (usually via **Ctrl−C**) to quit immediately. If−−*timeout* is specified, the command gi ves up waiting for events after *seconds* have elapsed. With −−*loop*, the command prints all events until a timeout or interrupt key.

When −−*timestamp* is used, a human−readable timestamp will be printed before the event.

VOLUME COMMANDS

vol−create
Syntax:

```
vol−create pool−or−uuid FILE [−−prealloc−metadata]
```

Create a volume from an XML <file>.

pool−or−uuid is the name or UUID of the storage pool to create the volume in.

FILE is the XML <file> with the volume definition. An easy way to create the XML <file> is to use the **vol−dumpxml** command to obtain the definition of a pre−existing volume.

[−−*prealloc−metadata*] preallocate metadata (for qcow2 images which don't support full allocation). This option creates a sparse image file with metadata, resulting in higher performance compared to images with no preallocation and only slightly higher initial disk space usage.

Example:

```
virsh vol−dumpxml −−pool storagepool1 appvolume1 > newvolume.xml
vi newvolume.xml (or make changes with your other text editor)
virsh vol−create differentstoragepool newvolume.xml
```

vol−create−from
Syntax:

```
vol−create−from pool−or−uuid FILE vol−name−or−key−or−path
   [−−inputpool pool−or−uuid]  [−−prealloc−metadata] [−−reflink]
```

Create a volume, using another volume as input.

pool−or−uuid is the name or UUID of the storage pool to create the volume in.

FILE is the XML <file> with the volume definition.

vol−name−or−key−or−path is the name or key or path of the source volume.

−−inputpool pool−or−uuid is the name or uuid of the storage pool the source volume is in.

[*−−prealloc−metadata*] preallocate metadata (for qcow2 images which don't support full allocation). This option creates a sparse image file with metadata, resulting in higher performance compared to images with no preallocation and only slightly higher initial disk space usage.

When *−−reflink* is specified, perform a COW lightweight copy, where the data blocks are copied only when modified. If this is not possible, the copy fails.

vol−create−as
Syntax:

```
vol−create−as pool−or−uuid name capacity [−−allocation size] [−−format string]
   [−−backing−vol vol−name−or−key−or−path]
   [−−backing−vol−format string] [−−prealloc−metadata] [−−print−xml]
```

Create a volume from a set of arguments unless *−−print−xml* is specified, in which case just the XML of the volume object is printed out without any actual object creation.

pool−or−uuid is the name or UUID of the storage pool to create the volume in.

name is the name of the new volume. For a disk pool, this must match the partition name as determined from the pool's source device path and the next available partition. For example, a source device path of /dev/sdb and there are no partitions on the disk, then the name must be sdb1 with the next name being sdb2 and so on.

capacity is the size of the volume to be created, as a scaled integer (see **NOTES** above), defaulting to bytes if there is no suffix.

−−allocation size is the initial size to be allocated in the volume, also as a scaled integer defaulting to bytes.

−−format string is used in file based storage pools to specify the volume file format to use; raw, bochs, qcow, qcow2, vmdk, qed. Use extended for disk storage pools in order to create an extended partition (other values are validity checked but not preserved when libvirtd is restarted or the pool is refreshed).

−−backing−vol vol−name−or−key−or−path is the source backing volume to be used if taking a snapshot of an existing volume.

−−backing−vol−format string is the format of the snapshot backing volume; raw, bochs, qcow, qcow2, qed, vmdk, host_device. These are, however, meant for file based storage pools.

[*--prealloc-metadata*] preallocate metadata (for qcow2 images which don't support full allocation). This option creates a sparse image file with metadata, resulting in higher performance compared to images with no preallocation and only slightly higher initial disk space usage.

vol-clone

Syntax:

```
vol-clone vol-name-or-key-or-path name
   [--pool pool-or-uuid] [--prealloc-metadata] [--reflink]
```

Clone an existing volume within the parent pool. Less powerful, but easier to type, version of **vol-create-from**.

vol-name-or-key-or-path is the name or key or path of the source volume.

name is the name of the new volume.

--pool pool-or-uuid is the name or UUID of the storage pool that contains the source volume and will contain the new volume. If the source volume name is provided instead of the key or path, then providing the pool is necessary to find the volume to be cloned; otherwise, the first volume found by the key or path will be used.

[*--prealloc-metadata*] preallocate metadata (for qcow2 images which don't support full allocation). This option creates a sparse image file with metadata, resulting in higher performance compared to images with no preallocation and only slightly higher initial disk space usage.

When *--reflink* is specified, perform a COW lightweight copy, where the data blocks are copied only when modified. If this is not possible, the copy fails.

vol-delete

Syntax:

```
vol-delete vol-name-or-key-or-path [--pool pool-or-uuid] [--delete-snapshots]
```

Delete a given volume.

vol-name-or-key-or-path is the volume name or key or path of the volume to delete.

[*--pool pool-or-uuid*] is the name or UUID of the storage pool the volume is in. If the volume name is provided instead of the key or path, then providing the pool is necessary to find the volume to be deleted; otherwise, the first volume found by the key or path will be used.

The *--delete-snapshots* flag specifies that any snapshots associated with the storage volume should be deleted as well. Not all storage drivers support this option, presently only rbd.

vol-upload

Syntax:

```
vol-upload vol-name-or-key-or-path local-file
   [--pool pool-or-uuid] [--offset bytes]
   [--length bytes] [--sparse]
```

Upload the contents of *local-file* to a storage volume.

vol-name-or-key-or-path is the name or key or path of the volume where the *local-file* will be uploaded.

−−pool pool−or−uuid is the name or UUID of the storage pool the volume is in. If the volume name is provided instead of the key or path, then providing the pool is necessary to find the volume to be uploaded into; otherwise, the first volume found by the key or path will be used.

−−offset is the position in the storage volume at which to start writing the data. The value must be 0 or larger.

−−length is an upper bound of the amount of data to be uploaded. A negative value is interpreted as an unsigned long long value to essentially include everything from the offset to the end of the volume.

If *−−sparse* is specified, this command will preserve volume sparseness.

An error will occur if the *local−file* is greater than the specified *length*.

See the description for the libvirt virStorageVolUpload API for details regarding possible target volume and pool changes as a result of the pool refresh when the upload is attempted.

vol−download
Syntax:

```
vol-download vol-name-or-key-or-path local-file
   [--pool pool-or-uuid] [--offset bytes] [--length bytes]
   [--sparse]
```

Download the contents of a storage volume to *local−file*.

vol−name−or−key−or−path is the name or key or path of the volume to download into *local−file*.

−−pool pool−or−uuid is the name or UUID of the storage pool the volume is in. If the volume name is provided instead of the key or path, then providing the pool is necessary to find the volume to be uploaded into; otherwise, the first volume found by the key or path will be used.

−−offset is the position in the storage volume at which to start reading the data. The value must be 0 or larger.

−−length is an upper bound of the amount of data to be downloaded. A negative value is interpreted as an unsigned long long value to essentially include everything from the offset to the end of the volume.

If *−−sparse* is specified, this command will preserve volume sparseness.

vol−wipe
Syntax:

```
vol-wipe vol-name-or-key-or-path [--pool pool-or-uuid] [--algorithm algorithm]
```

Wipe a volume, ensure data previously on the volume is not accessible to future reads.

vol−name−or−key−or−path is the name or key or path of the volume to wipe. It is possible to choose different wiping algorithms instead of re−writing volume with zeroes.

−−pool pool−or−uuid is the name or UUID of the storage pool the volume is in. If the volume name is provided instead of the key or path, then providing the pool is necessary to find the volume to be wiped; otherwise, the first volume found by the key or path will be used.

Use the *−−algorithm* switch choosing from the list of the following algorithms in order to define which algorithm to use for the wipe.

Supported algorithms

- zero − 1−pass all zeroes
- nnsa − 4−pass NNSA Policy Letter NAP−14.1−C (XVI−8) for sanitizing removable and non−removable hard disks: random x2, 0x00, verify.
- dod − 4−pass DoD 5220.22−M section 8−306 procedure for sanitizing removable and non−removable rigid disks: random, 0x00, 0xff, verify.
- bsi − 9−pass method recommended by the German Center of Security in Information Technologies (*https://www.bsi.bund.de*): 0xff, 0xfe, 0xfd, 0xfb, 0xf7, 0xef, 0xdf, 0xbf, 0x7f.
- gutmann − The canonical 35−pass sequence described in Gutmann's paper.
- schneier − 7−pass method described by Bruce Schneier in "Applied Cryptography" (1996): 0x00, 0xff, random x5.
- pfitzner7 − Roy Pfitzner's 7−random−pass method: random x7.
- pfitzner33 − Roy Pfitzner's 33−random−pass method: random x33.
- random − 1−pass pattern: random.
- trim − 1−pass trimming the volume using TRIM or DISCARD

Note: The **scrub** binary will be used to handle the 'nnsa', 'dod', 'bsi', 'gutmann', 'schneier', 'pfitzner7' and 'pfitzner33' algorithms. The availability of the algorithms may be limited by the version of the **scrub** binary installed on the host. The 'zero' algorithm will write zeroes to the entire volume. For some volumes, such as sparse or rbd volumes, this may result in completely filling the volume with zeroes making it appear to be completely full. As an alternative, the 'trim' algorithm does not overwrite all the data in a volume, rather it expects the storage driver to be able to discard all bytes in a volume. It is up to the storage driver to handle how the discarding occurs. Not all storage drivers or volume types can support 'trim'.

vol−dumpxml

Syntax:

```
vol−dumpxml vol−name−or−key−or−path [−−pool pool−or−uuid]
```

Output the volume information as an XML dump to stdout.

vol−name−or−key−or−path is the name or key or path of the volume to output the XML.

−−pool pool−or−uuid is the name or UUID of the storage pool the volume is in. If the volume name is provided instead of the key or path, then providing the pool is necessary to find the volume to be uploaded into; otherwise, the first volume found by the key or path will be used.

vol−info

Syntax:

```
vol−info vol−name−or−key−or−path [−−pool pool−or−uuid] [−−bytes] [−−physical]
```

Returns basic information about the given storage volume.

vol−name−or−key−or−path is the name or key or path of the volume to return information for.

−−pool pool−or−uuid is the name or UUID of the storage pool the volume is in. If the volume name is provided instead of the key or path, then providing the pool is necessary to find the volume to be uploaded into; otherwise, the first volume found by the key or path will be used.

If *−−bytes* is specified the sizes are not converted to human friendly units.

If *--physical* is specified, then the host physical size is returned and displayed instead of the allocation value. The physical value for some file types, such as qcow2 may have a different (larger) physical value than is shown for allocation. Additionally sparse files will have different physical and allocation values.

vol-list
Syntax:

```
vol-list [--pool pool-or-uuid] [--details]
```

Return the list of volumes in the given storage pool.

--pool pool-or-uuid is the name or UUID of the storage pool.

The *--details* option instructs virsh to additionally display volume type and capacity related information where available.

vol-pool
Syntax:

```
vol-pool vol-key-or-path [--uuid]
```

Return the pool name or UUID for a given volume. By default, the pool name is returned.

vol-key-or-path is the key or path of the volume to return the pool information.

If the *--uuid* option is given, the pool UUID is returned instead.

vol-path
Syntax:

```
vol-path vol-name-or-key [--pool pool-or-uuid]
```

Return the path for a given volume.

vol-name-or-key is the name or key of the volume to return the path.

--pool pool-or-uuid is the name or UUID of the storage pool the volume is in. If the volume name is provided instead of the key, then providing the pool is necessary to find the volume to be uploaded into; otherwise, the first volume found by the key will be used.

vol-name
Syntax:

```
vol-name vol-key-or-path
```

Return the name for a given volume.

vol-key-or-path is the key or path of the volume to return the name.

vol-key
Syntax:

```
vol-key vol-name-or-path [--pool pool-or-uuid]
```

Return the volume key for a given volume.

vol-name-or-path is the name or path of the volume to return the volume key.

−−pool pool−or−uuid is the name or UUID of the storage pool the volume is in. If the volume name is provided instead of the path, then providing the pool is necessary to find the volume to be uploaded into; otherwise, the first volume found by the path will be used.

vol−resize

Syntax:

```
vol−resize vol−name−or−path capacity [−−pool pool−or−uuid] [−−allocate] [−−del
```

Resize the capacity of the given volume, in bytes.

vol−name−or−key−or−path is the name or key or path of the volume to resize.

capacity is a scaled integer (see **NOTES** above) for the volume, which defaults to bytes if there is no suffix.

−−pool pool−or−uuid is the name or UUID of the storage pool the volume is in. If the volume name is provided instead of the key or path, then providing the pool is necessary to find the volume to be uploaded into; otherwise, the first volume found by the key or path will be used.

The new *capacity* might be sparse unless *−−allocate* is specified.

Normally, *capacity* is the new size, but if *−−delta* is present, then it is added to the existing size.

Attempts to shrink the volume will fail unless *−−shrink* is present. The *capacity* cannot be negative unless *−−shrink* is provided, but a negative sign is not necessary.

This command is only safe for storage volumes not in use by an active guest; see also **blockresize** for live resizing.

SECRET COMMANDS

The following commands manipulate "secrets" (e.g. passwords, passphrases and encryption keys). Libvirt can store secrets independently from their use, and other objects (e.g. volumes or domains) can refer to the secrets for encryption or possibly other uses. Secrets are identified using a UUID. See *https://libvirt.org/formatsecret.html* for documentation of the XML format used to represent properties of secrets.

secret−define

Syntax:

```
secret−define file [−−validate]
```

Create a secret with the properties specified in *file*, with no associated secret value. If*file* does not specify a UUID, choose one automatically. If*file* specifies a UUID of an e xisting secret, replace its properties by properties defined in *file*, without affecting the secret value.

Optionally, the format of the input XML file can be validated against an internal RNG schema with *−−validate*.

secret−dumpxml

Syntax:

```
secret−dumpxml secret
```

Output properties of *secret* (specified by its UUID) as an XML dump to stdout.

secret−event
Syntax:

```
secret−event {[secret] event [−−loop] [−−timeout seconds] [−−timestamp] | −−lis
```

Wait for a class of secret events to occur, and print appropriate details of events as they happen. The events can optionally be filtered by *secret*. Using−−*list* as the only ar gument will provide a list of possible *event* values known by this client, although the connection might not allow registering for all these events.

By default, this command is one−shot, and returns success once an event occurs; you can send SIGINT (usually via **Ctrl−C**) to quit immediately. If−−*timeout* is specified, the command gi ves up waiting for events after *seconds* have elapsed. With −−*loop*, the command prints all events until a timeout or interrupt key.

When −−*timestamp* is used, a human−readable timestamp will be printed before the event.

secret−set−value
Syntax:

```
secret−set−value secret (−−file filename [−−plain] | −−interactive | base64)
```

Set the value associated with *secret* (specified by its UUID) to the value Base64−encoded value *base64* or Base−64−encoded contents of file named *filename*. Using the −−*plain* flag is together with −−*file* allows one to use the file contents directly as the secret value.

If −−*interactive* flag is used the secret value is read as a password from the terminal.

Note that −−*file*, −−*interactive* and *base64* options are mutually exclusive.

Passing secrets via the *base64* option on command line is INSECURE and deprecated. Use the −−*file* option instead.

secret−get−value
Syntax:

```
secret−get−value [−−plain] secret
```

Output the value associated with *secret* (specified by its UUID) to stdout, encoded using Base64.

If the −−*plain* flag is used the value is not base64 encoded, but rather printed raw. Note that unless virsh is started in quiet mode (*virsh −q*) it prints a newline at the end of the command. This newline is not part of the secret.

secret−undefine
Syntax:

```
secret−undefine secret
```

Delete a *secret* (specified by its UUID), including the associated value, if any.

secret−list
Syntax:

```
secret−list [−−ephemeral] [−−no−ephemeral]
   [−−private] [−−no−private]
```

Returns the list of secrets. You may also want to filter the returned secrets by −−*ephemeral* to list the

ephemeral ones, *−−no−ephemeral* to list the non−ephemeral ones, *−−private* to list the private ones, and *−−no−private* to list the non−private ones.

SNAPSHOT COMMANDS

The following commands manipulate domain snapshots. Snapshots take the disk, memory, and device state of a domain at a point−of−time, and save it for future use. They have many uses, from saving a "clean" copy of an OS image to saving a domain's state before a potentially destructive operation. Snapshots are identified with a unique name. See *https://libvirt.org/formatsnapshot.html* for documentation of the XML format used to represent properties of snapshots.

snapshot−create

Syntax:

```
snapshot−create domain [xmlfile] {[−−redefine [−−current]] |
   [−−no−metadata] [−−halt] [−−disk−only] [−−reuse−external]
   [−−quiesce] [−−atomic] [−−live]} [−−validate]
```

Create a snapshot for domain *domain* with the properties specified in *xmlfile*. Optionally, the *−−validate* option can be passed to validate the format of the input XML file against an internal RNG schema (identical to using the virt−xml−validate(1) tool). Normally, the only properties settable for a domain snapshot are the <name> and <description> elements, as well as <disks> if *−−disk−only* is given; the rest of the fields are ignored, and automatically filled in by libvirt. If*xmlfile* is completely omitted, then lib virt will choose a value for all fields. The new snapshot will become current, as listed by **snapshot−current**.

If *−−halt* is specified, the domain will be left in an inactive state after the snapshot is created.

If *−−disk−only* is specified, the snapshot will only include disk content rather than the usual full system snapshot with vm state. Disk snapshots are captured faster than full system snapshots, but reverting to a disk snapshot may require fsck or journal replays, since it is like the disk state at the point when the power cord is abruptly pulled; and mixing *−−halt* and *−−disk−only* loses any data that was not flushed to disk at the time.

If *−−redefine* is specified, then all XML elements produced by **snapshot−dumpxml** are valid; this can be used to migrate snapshot hierarchy from one machine to another, to recreate hierarchy for the case of a transient domain that goes away and is later recreated with the same name and UUID, or to make slight alterations in the snapshot metadata (such as host−specific aspects of the domain XML embedded in the snapshot). When this flag is supplied, the*xmlfile* ar gument is mandatory, and the domain's current snapshot will not be altered unless the *−−current* flag is also given.

If *−−no−metadata* is specified, then the snapshot data is created, but any metadata is immediately discarded (that is, libvirt does not treat the snapshot as current, and cannot revert to the snapshot unless *−−redefine* is later used to teach libvirt about the metadata again).

If *−−reuse−external* is specified, and the snapshot XML requests an external snapshot with a destination of an existing file, then the destination must exist and be pre−created with correct format and metadata. The file is then reused; otherwise, a snapshot is refused to avoid losing contents of the existing files.

If *−−quiesce* is specified, libvirt will try to use guest agent to freeze and unfreeze domain's mounted file systems. However, if domain has no guest agent, snapshot creation will fail. Currently, this requires *−−disk−only* to be passed as well.

If *−−atomic* is specified, libvirt will guarantee that the snapshot either succeeds, or fails with no changes; not all hypervisors support this. If this flag is not specified, then some hypervisors may fail after partially performing the action, and **dumpxml** must be used to see whether any partial changes occurred.

If *−−live* is specified, libvirt takes the snapshot while the guest is running. Both disk snapshot and domain

memory snapshot are taken. This increases the size of the memory image of the external snapshot. This is currently supported only for full system external snapshots.

Existence of snapshot metadata will prevent attempts to **undefine** a persistent guest. However, for transient domains, snapshot metadata is silently lost when the domain quits running (whether by command such as **destroy** or by internal guest action).

For now, it is not possible to create snapshots in a domain that has checkpoints, although this restriction will be lifted in a future release.

snapshot−create−as

Syntax:

```
snapshot−create−as domain {[−−print−xml] [−−no−metadata]
   [−−halt] [−−reuse−external]} [name]
   [description] [−−disk−only [−−quiesce]] [−−atomic] [−−validate]
   [[−−live] [−−memspec memspec]] [−−diskspec] diskspec]...
```

Create a snapshot for domain *domain* with the given <name> and <description>; if either value is omitted, libvirt will choose a value. If*−−print−xml* is specified, then XML appropriate for *snapshot−cr eate* is output, rather than actually creating a snapshot. Otherwise, if *−−halt* is specified, the domain will be left in an inactive state after the snapshot is created, and if *−−disk−only* is specified, the snapshot will not include vm state.

The *−−memspec* option can be used to control whether a full system snapshot is internal or external. The *−−memspec* flag is mandatory, followed by a **memspec** of the form **[file=]name[,snapshot=type]**, where type can be **no**, **internal**, or **external**. To include a literal comma in **file=name**, escape it with a second comma. *−−memspec* cannot be used together with *−−disk−only*.

The *−−diskspec* option can be used to control how *−−disk−only* and external full system snapshots create external files. This option can occur multiple times, according to the number of <disk> elements in the domain xml. Each <diskspec> is in the form **disk[,snapshot=type][,driver=type][,stype=type][,file=name]**. A *diskspec* must be provided for disks backed by block devices as libvirt doesn't auto−generate file names for those. The optional **stype** parameter allows one to control the type of the source file. Supported values are 'file' (default) and 'block'. To exclude a disk from an external snapshot use **−−diskspec disk,snapshot=no**.

To include a literal comma in **disk** or in **file=name**, escape it with a second comma. A literal *−−diskspec* must precede each **diskspec** unless all three of *domain*, *name*, and *description* are also present. For example, a diskspec of "vda,snapshot=external,file=/path/to,,new" results in the following XML:

```
<disk name='vda' snapshot='external'>
  <source file='/path/to,new'/>
</disk>
```

If *−−reuse−external* is specified, and the domain XML or *diskspec* option requests an external snapshot with a destination of an existing file, then the destination must exist and be pre−created with correct format and metadata. The file is then reused; otherwise, a snapshot is refused to avoid losing contents of the existing files.

If *−−quiesce* is specified, libvirt will try to use guest agent to freeze and unfreeze domain's mounted file systems. However, if domain has no guest agent, snapshot creation will fail. Currently, this requires *−−disk−only* to be passed as well.

If *−−no−metadata* is specified, then the snapshot data is created, but any metadata is immediately

discarded (that is, libvirt does not treat the snapshot as current, and cannot revert to the snapshot unless **snapshot−create** is later used to teach libvirt about the metadata again).

If *−−atomic* is specified, libvirt will guarantee that the snapshot either succeeds, or fails with no changes; not all hypervisors support this. If this flag is not specified, then some hypervisors may fail after partially performing the action, and **dumpxml** must be used to see whether any partial changes occurred.

If *−−live* is specified, libvirt takes the snapshot while the guest is running. This increases the size of the memory image of the external snapshot. This is currently supported only for external full system snapshots.

For now, it is not possible to create snapshots in a domain that has checkpoints, although this restriction will be lifted in a future release.

Optionally, the *−−validate* option can be passed to validate XML document which is internally generated by this command against the internal RNG schema.

snapshot−current
Syntax:

```
snapshot−current domain {[−−name] | [−−security−info] | [snapshotname]}
```

Without *snapshotname*, this will output the snapshot XML for the domain's current snapshot (if any). If *−−name* is specified, just the current snapshot name instead of the full xml. Otherwise, using *−−security−info* will also include security sensitive information in the XML.

With *snapshotname*, this is a request to make the existing named snapshot become the current snapshot, without reverting the domain.

snapshot−edit
Syntax:

```
snapshot−edit domain [snapshotname] [−−current] {[−−rename] | [−−clone]}
```

Edit the XML configuration file for *snapshotname* of a domain. If both *snapshotname* and *−−current* are specified, also force the edited snapshot to become the current snapshot. If *snapshotname* is omitted, then *−−current* must be supplied, to edit the current snapshot.

This is equivalent to:

```
virsh snapshot−dumpxml dom name > snapshot.xml
vi snapshot.xml (or make changes with your other text editor)
virsh snapshot−create dom snapshot.xml −−redefine [−−current]
```

except that it does some error checking.

The editor used can be supplied by the **$VISUAL** or **$EDITOR** environment variables, and defaults to **vi**.

If *−−rename* is specified, then the edits can change the snapshot name. If *−−clone* is specified, then changing the snapshot name will create a clone of the snapshot metadata. If neither is specified, then the edits must not change the snapshot name. Note that changing a snapshot name must be done with care, since the contents of some snapshots, such as internal snapshots within a single qcow2 file, are accessible only from the original name.

snapshot−info
Syntax:

```
snapshot-info domain {snapshot | --current}
```

Output basic information about a named <snapshot>, or the current snapshot with *--current*.

snapshot-list

Syntax:

```
snapshot-list domain [--metadata] [--no-metadata]
   [{--parent | --roots | [{--tree | --name}]}] [--topological]
   [{[--from] snapshot | --current} [--descendants]]
   [--leaves] [--no-leaves] [--inactive] [--active]
   [--disk-only] [--internal] [--external]
```

List all of the available snapshots for the given domain, defaulting to show columns for the snapshot name, creation time, and domain state.

Normally, table form output is sorted by snapshot name; using *--topological* instead sorts so that no child is listed before its ancestors (although there may be more than one possible ordering with this property).

If *--parent* is specified, add a column to the output table giving the name of the parent of each snapshot. If *--roots* is specified, the list will be filtered to just snapshots that have no parents. If *--tree* is specified, the output will be in a tree format, listing just snapshot names. These three options are mutually exclusive. If *--name* is specified only the snapshot name is printed. This option is mutually exclusive with *--tree*.

If *--from* is provided, filter the list to snapshots which are children of the given **snapshot**; or if *--current* is provided, start at the current snapshot. When used in isolation or with *--parent*, the list is limited to direct children unless *--descendants* is also present. When used with *--tree*, the use of *--descendants* is implied. This option is not compatible with*--r oots*. Note that the starting point of*--fr om* or *--current* is not included in the list unless the *--tree* option is also present.

If *--leaves* is specified, the list will be filtered to just snapshots that have no children. Likewise, if *--no-leaves* is specified, the list will be filtered to just snapshots with children. (Note that omitting both options does no filtering, while providing both options will either produce the same list or error out depending on whether the server recognizes the flags). Filtering options are not compatible with *--tree*.

If *--metadata* is specified, the list will be filtered to just snapshots that involve libvirt metadata, and thus would prevent **undefine** of a persistent guest, or be lost on **destroy** of a transient domain. Likewise, if *--no-metadata* is specified, the list will be filtered to just snapshots that exist without the need for libvirt metadata.

If *--inactive* is specified, the list will be filtered to snapshots that were taken when the domain was shut off. If*--active* is specified, the list will be filtered to snapshots that were tak en when the domain was running, and where the snapshot includes the memory state to revert to that running state. If *--disk-only* is specified, the list will be filtered to snapshots that were taken when the domain was running, but where the snapshot includes only disk state.

If *--internal* is specified, the list will be filtered to snapshots that use internal storage of existing disk images. If*--e xternal* is specified, the list will be filtered to snapshots that use external files for disk images or memory state.

snapshot-dumpxml

Syntax:

```
snapshot-dumpxml domain snapshot [--security-info]
```

Output the snapshot XML for the domain's snapshot named *snapshot*. Using*--security-info* will also

include security sensitive information. Use **snapshot−current** to easily access the XML of the current snapshot.

snapshot−parent
Syntax:

```
snapshot−parent domain {snapshot | −−current}
```

Output the name of the parent snapshot, if any, for the given *snapshot*, or for the current snapshot with *−−current*.

snapshot−revert
Syntax:

```
snapshot−revert domain {snapshot | −−current} [{−−running | −−paused}] [−−forc
```

Revert the given domain to the snapshot specified by *snapshot*, or to the current snapshot with *−−current*. Be aware that this is a destructive action; any changes in the domain since the last snapshot was taken will be lost. Also note that the state of the domain after snapshot−revert is complete will be the state of the domain at the time the original snapshot was taken.

Normally, reverting to a snapshot leaves the domain in the state it was at the time the snapshot was created, except that a disk snapshot with no vm state leaves the domain in an inactive state. Passing either the *−−running* or *−−paused* flag will perform additional state changes (such as booting an inactive domain, or pausing a running domain). Since transient domains cannot be inactive, it is required to use one of these flags when reverting to a disk snapshot of a transient domain.

Since libvirt 7.10.0 the VM process is always restarted so the following paragraph is no longer valid. If the snapshot metadata lacks the full VM XML it's no longer possible to revert to such snapshot.

There are a number of cases where a snapshot revert involves extra risk, which requires the use of *−−force* to proceed:

- One is the case of a snapshot that lacks full domain information for reverting configuration (such as snapshots created prior to libvirt 0.9.5); since libvirt cannot prove that the current configuration matches what was in use at the time of the snapshot, supplying *−−force* assures libvirt that the snapshot is compatible with the current configuration (and if it is not, the domain will likely fail to run).
- Another is the case of reverting from a running domain to an active state where a new hypervisor has to be created rather than reusing the existing hypervisor, because it implies drawbacks such as breaking any existing VNC or Spice connections; this condition happens with an active snapshot that uses a provably incompatible configuration, as well as with an inactive snapshot that is combined with the *−−start* or *−−pause* flag.
- Also, libvirt will refuse to restore snapshots of inactive QEMU domains while there is managed saved state. This is because those snapshots do not contain memory state and will therefore not replace the existing memory state. This ends up switching a disk underneath a running system and will likely cause extensive filesystem corruption or crashes due to swap content mismatches when run.

snapshot−delete
Syntax:

```
snapshot−delete domain {snapshot | −−current}
   [−−metadata] [{−−children | −−children−only}]
```

Delete the snapshot for the domain named *snapshot*, or the current snapshot with *−−current*. If this snapshot has child snapshots, changes from this snapshot will be merged into the children. If *−−children* is passed, then delete this snapshot and any children of this snapshot. If *−−children−only* is passed, then

delete any children of this snapshot, but leave this snapshot intact. These two flags are mutually exclusive.

If *--metadata* is specified, then only delete the snapshot metadata maintained by libvirt, while leaving the snapshot contents intact for access by external tools; otherwise deleting a snapshot also removes the data contents from that point in time.

CHECKPOINT COMMANDS

The following commands manipulate domain checkpoints. Checkpoints serve as a point in time to identify which portions of a guest's disks have changed after that time, making it possible to perform incremental and differential backups. Checkpoints are identified with a unique name. See *https://libvirt.org/formatcheckpoint.html* for documentation of the XML format used to represent properties of checkpoints.

checkpoint-create

Syntax:

```
checkpoint-create domain [xmlfile] { --redefine [--redefine-validate] | [--qui
```

Create a checkpoint for domain *domain* with the properties specified in *xmlfile* describing a <domaincheckpoint> top-level element. The format of the input XML file will be validated against an internal RNG schema (identical to using the virt-xml-validate(1) tool). If *xmlfile* is completely omitted, then libvirt will create a checkpoint with a name based on the current time.

If *--redefine* is specified, then all XML elements produced by **checkpoint-dumpxml** are valid; this can be used to migrate checkpoint hierarchy from one machine to another, to recreate hierarchy for the case of a transient domain that goes away and is later recreated with the same name and UUID, or to make slight alterations in the checkpoint metadata (such as host-specific aspects of the domain XML embedded in the checkpoint). When this flag is supplied, the*xmlfile* ar gument is mandatory.

If *--redefine-validate* is specified along with *--redefine* the hypervisor performs validation of metadata associated with the checkpoint stored in places besides the checkpoint XML. Note that some hypervisors may require that the domain is running to perform validation.

If *--quiesce* is specified, libvirt will try to use guest agent to freeze and unfreeze domain's mounted file systems. However, if domain has no guest agent, checkpoint creation will fail.

Existence of checkpoint metadata will prevent attempts to **undefine** a persistent guest. However, for transient domains, checkpoint metadata is silently lost when the domain quits running (whether by command such as **destroy** or by internal guest action).

For now, it is not possible to create checkpoints in a domain that has snapshots, although this restriction will be lifted in a future release.

checkpoint-create-as

Syntax:

```
checkpoint-create-as domain [--print-xml] [name]
   [description] [--quiesce] [--diskspec] diskspec]...
```

Create a checkpoint for domain *domain* with the given <name> and <description>; if either value is omitted, libvirt will choose a value. If*--print-xml* is specified, then XML appropriate for *c heckpoint-create* is output, rather than actually creating a checkpoint.

The *--diskspec* option can be used to control which guest disks participate in the checkpoint. This option can occur multiple times, according to the number of <disk> elements in the domain xml. Each <diskspec> is in the form **disk[,checkpoint=type][,bitmap=name]**. A literal *--diskspec* must precede each **diskspec**

unless all three of *domain*, *name*, and *description* are also present. For example, a diskspec of "vda,checkpoint=bitmap,bitmap=map1" results in the following XML:

```
<disk name='vda' checkpoint='bitmap' bitmap='map1'/>
```

If *--quiesce* is specified, libvirt will try to use guest agent to freeze and unfreeze domain's mounted file systems. However, if domain has no guest agent, checkpoint creation will fail.

For now, it is not possible to create checkpoints in a domain that has snapshots, although this restriction will be lifted in a future release.

checkpoint-edit

Syntax:

```
checkpoint-edit domain checkpointname
```

Edit the XML configuration file for *checkpointname* of a domain.

This is equivalent to:

```
virsh checkpoint-dumpxml dom name > checkpoint.xml
vi checkpoint.xml (or make changes with your other text editor)
virsh checkpoint-create dom checkpoint.xml --redefine
```

except that it does some error checking, including that the edits should not attempt to change the checkpoint name.

The editor used can be supplied by the **$VISUAL** or **$EDITOR** environment variables, and defaults to **vi**.

checkpoint-info

Syntax:

```
checkpoint-info domain checkpoint
```

Output basic information about a named <checkpoint>.

checkpoint-list

Syntax:

```
checkpoint-list domain [{--parent | --roots |
   [{--tree | --name}]}] [--topological]
   [[--from] checkpoint | [--descendants]]
   [--leaves] [--no-leaves]
```

List all of the available checkpoints for the given domain, defaulting to show columns for the checkpoint name and creation time.

Normally, table form output is sorted by checkpoint name; using *--topological* instead sorts so that no child is listed before its ancestors (although there may be more than one possible ordering with this property).

If *--parent* is specified, add a column to the output table giving the name of the parent of each checkpoint. If *--roots* is specified, the list will be filtered to just checkpoints that have no parents. If *--tree* is specified, the output will be in a tree format, listing just checkpoint names. These three options are mutually exclusive. If *--name* is specified only the checkpoint name is printed. This option is mutually exclusive with *--tree*.

If *--from* is provided, filter the list to checkpoints which are children of the given **checkpoint**. When used in isolation or with *--parent*, the list is limited to direct children unless *--descendants* is also present. When used with *--tree*, the use of *--descendants* is implied. This option is not compatible with *--roots*. Note that the starting point of *--from* is not included in the list unless the *--tree* option is also present.

If *--leaves* is specified, the list will be filtered to just checkpoints that have no children. Likewise, if *--no-leaves* is specified, the list will be filtered to just checkpoints with children. (Note that omitting both options does no filtering, while providing both options will either produce the same list or error out depending on whether the server recognizes the flags). Filtering options are not compatible with *--tree*.

checkpoint-dumpxml

Syntax:

```
checkpoint-dumpxml domain checkpoint [--security-info] [--no-domain] [--size]
```

Output the checkpoint XML for the domain's checkpoint named *checkpoint*. Using*--security-info* will also include security sensitive information.

Using *--size* will add XML indicating the current size in bytes of guest data that has changed since the checkpoint was created (although remember that guest activity between a size check and actually creating a backup can result in the backup needing slightly more space). Note that some hypervisors may require that *domain* is running when *--size* is used.

Using *--no-domain* will omit the <domain> element from the output for a more compact view.

checkpoint-parent

Syntax:

```
checkpoint-parent domain checkpoint
```

Output the name of the parent checkpoint, if any, for the given *checkpoint*.

checkpoint-delete

Syntax:

```
checkpoint-delete domain checkpoint
   [--metadata] [{--children | --children-only}]
```

Delete the checkpoint for the domain named *checkpoint*. The record of which portions of the disk changed since the checkpoint are merged into the parent checkpoint (if any). If *--children* is passed, then delete this checkpoint and any children of this checkpoint. If *--children-only* is passed, then delete any children of this checkpoint, but leave this checkpoint intact. These two flags are mutually exclusive.

If *--metadata* is specified, then only delete the checkpoint metadata maintained by libvirt, while leaving the checkpoint contents intact for access by external tools; otherwise deleting a checkpoint also removes the ability to perform an incremental backup from that point in time.

NWFILTER COMMANDS

The following commands manipulate network filters. Network filters allow filtering of the network traffic coming from and going to virtual machines. Individual network traffic filters are written in XML and may contain references to other network filters, describe traffic filtering rules, or contain both. Network filters are referenced by virtual machines from within their interface description. A network filter may be referenced by multiple virtual machines' interfaces.

nwfilter-define

Syntax:

```
nwfilter-define xmlfile [--validate]
```

Make a new network filter known to libvirt. If a network filter with the same name already exists, it will be replaced with the new XML. Any running virtual machine referencing this network filter will have its network traffic rules adapted. If for any reason the network traffic filtering rules cannot be instantiated by any of the running virtual machines, then the new XML will be rejected.

Optionally, the format of the input XML file can be validated against an internal RNG schema with *--validate*.

nwfilter-undefine
Syntax:

```
nwfilter-undefine nwfilter-name
```

Delete a network filter. The deletion will fail if any running virtual machine is currently using this network filter.

nwfilter-list
Syntax:

```
nwfilter-list
```

List all of the available network filters.

nwfilter-dumpxml
Syntax:

```
nwfilter-dumpxml nwfilter-name
```

Output the network filter XML.

nwfilter-edit
Syntax:

```
nwfilter-edit nwfilter-name
```

Edit the XML of a network filter.

This is equivalent to:

```
virsh nwfilter-dumpxml myfilter > myfilter.xml
vi myfilter.xml (or make changes with your other text editor)
virsh nwfilter-define myfilter.xml
```

except that it does some error checking. The new network filter may be rejected due to the same reason as mentioned in *nwfilter-define*.

The editor used can be supplied by the **$VISUAL** or **$EDITOR** environment variables, and defaults to **vi**.

NWFILTER BINDING COMMANDS

The following commands manipulate network filter bindings. Network filter bindings track the association between a network port and a network filter. Generally the bindings are managed automatically by the hypervisor drivers when adding/removing NICs on a guest.

If an admin is creating/deleting TAP devices for non-guest usage, however, the network filter binding commands provide a way to make use of the network filters directly.

nwfilter−binding−create
Syntax:

```
nwfilter−binding−create xmlfile [−−validate]
```

Associate a network port with a network filter. The network filter backend will immediately attempt to instantiate the filter rules on the port. This command may be used to associate a filter with a currently running guest that does not have a filter defined for a specific network port. Since the bindings are generally automatically managed by the hypervisor, using this command to define a filter for a network port and then starting the guest afterwards may prevent the guest from starting if it attempts to use the network port and finds a filter already defined.

Optionally, the format of the input XML file can be validated against an internal RNG schema with *−−validate*.

nwfilter−binding−delete
Syntax:

```
nwfilter−binding−delete port−name
```

Disassociate a network port from a network filter. The network filter backend will immediately tear down the filter rules that exist on the port. This command may be used to remove the network port binding for a filter currently in use for the guest while the guest is running without needing to restart the guest. Restoring the network port binding filter for the running guest would be accomplished by using *nwfilter−binding−create*.

nwfilter−binding−list
Syntax:

```
nwfilter−binding−list
```

List all of the network ports which have filters associated with them.

nwfilter−binding−dumpxml
Syntax:

```
nwfilter−binding−dumpxml port−name
```

Output the network filter binding XML for the network device called **port−name**.

HYPERVISOR-SPECIFIC COMMANDS

NOTE: Use of the following commands is **strongly** discouraged. They can cause libvirt to become confused and do the wrong thing on subsequent operations. Once you have used these commands, please do not report problems to the libvirt developers; the reports will be ignored. If you find that these commands are the only way to accomplish something, then it is better to request that the feature be added as a first−class citizen in the regular libvirt library.

qemu−attach
Syntax:

```
qemu−attach pid
```

Attach an externally launched QEMU process to the libvirt QEMU driver. The QEMU process must have been created with a monitor connection using the UNIX driver. Ideally the process will also have had the '−name' argument specified.

```
$ qemu−kvm −cdrom ~/demo.iso \
```

```
    -monitor unix:/tmp/demo,server,nowait \
    -name foo \
    -uuid cece4f9f-dff0-575d-0e8e-01fe380f12ea  &
$ QEMUPID=$!
$ virsh qemu-attach $QEMUPID
```

Not all functions of libvirt are expected to work reliably after attaching to an externally launched QEMU process. There may be issues with the guest ABI changing upon migration and device hotplug or hotunplug may not work. The attached environment should be considered primarily read-only.

qemu-monitor-command

Syntax:

```
qemu-monitor-command domain { [--hmp] | [--pretty] [--return-value] } command.
```

Send an arbitrary monitor command *command* to domain *domain* through the QEMU monitor. The results of the command will be printed on stdout.

If more than one argument is provided for *command*, they are concatenated with a space in between before passing the single command to the monitor.

Note that libvirt uses the QMP to talk to qemu so *command* must be valid JSON in QMP format to work properly. If *command* is not a JSON object libvirt tries to wrap it as a JSON object to provide convenient interface such as the groups of commands with identical handling:

```
# simple command
$ virsh qemu-monitor-command VM commandname
$ virsh qemu-monitor-command VM '{"execute":"commandname"}'

# with arguments
$ virsh qemu-monitor-command VM commandname '"arg1":123' '"arg2":"test"'
$ virsh qemu-monitor-command VM commandname '{"arg1":123,"arg2":"test"}'
$ virsh qemu-monitor-command VM '{"execute":"commandname", "arguments":{"arg1"
```

If *--pretty* is given the QMP reply is pretty-printed.

If *--return-value* is given the 'return' key of the QMP response object is extracted rather than passing through the full reply from QEMU.

If *--hmp* is passed, the command is considered to be a human monitor command and libvirt will automatically convert it into QMP and convert the result back.

qemu-agent-command

Syntax:

```
qemu-agent-command domain [--timeout seconds | --async | --block] command...
```

Send an arbitrary guest agent command *command* to domain *domain* through QEMU agent. *--timeout*, *--async* and *--block* options are exclusive. *--timeout* requires timeout seconds *seconds* and it must be positive. When*--aysnc* is gi ven, the command waits for timeout whether success or failed. And when *--block* is given, the command waits forever with blocking timeout.

qemu-monitor-event

Syntax:

```
qemu-monitor-event [domain] [--event event-name]
```

```
[--loop] [--timeout seconds] [--pretty] [--regex] [--no-case]
[--timestamp]
```

Wait for arbitrary QEMU monitor events to occur, and print out the details of events as they happen. The events can optionally be filtered by *domain* or *event-name*. The 'query-events' QMP command can be used via *qemu-monitor-command* to learn what events are supported. If *--regex* is used, *event-name* is a basic regular expression instead of a literal string. If *--no-case* is used, *event-name* will match case-insensitively.

By default, this command is one-shot, and returns success once an event occurs; you can send SIGINT (usually via **Ctrl-C**) to quit immediately. If*--timeout* is specified, the command gi ves up waiting for events after *seconds* have elapsed. With *--loop*, the command prints all events until a timeout or interrupt key. If*--pr etty* is specified, any JSON event details are pretty-printed for better legibility.

When *--timestamp* is used, a human-readable timestamp will be printed before the event, and the timing information provided by QEMU will be omitted.

lxc-enter-namespace

Syntax:

```
lxc-enter-namespace domain [--noseclabel] --
   /path/to/binary [arg1, [arg2, ...]]
```

Enter the namespace of *domain* and execute the command **/path/to/binary** passing the requested args. The binary path is relative to the container root filesystem, not the host root filesystem. The binary will inherit the environment variables / console visible to virsh. The command will be run with the same sVirt context and cgroups placement as processes within the container. This command only works when connected to the LXC hypervisor driver. This command succeeds only if**/path/to/binary** has 0 e xit status.

By default the new process will run with the security label of the new parent container. Use the *--noseclabel* option to instead have the process keep the same security label as **virsh**.

ENVIRONMENT

The following environment variables can be set to alter the behaviour of **virsh**

- VIRSH_DEBUG=<0 to 4>

  Turn on verbose debugging of virsh commands. Valid levels are

  - VIRSH_DEBUG=0

    DEBUG - Messages at ALL levels get logged

  - VIRSH_DEBUG=1

    INFO - Logs messages at levels INFO, NOTICE, WARNING and ERROR

  - VIRSH_DEBUG=2

    NOTICE - Logs messages at levels NOTICE, WARNING and ERROR

  - VIRSH_DEBUG=3

    WARNING - Logs messages at levels WARNING and ERROR

  - VIRSH_DEBUG=4

    ERROR - Messages at only ERROR level gets logged.

- VIRSH_LOG_FILE=``LOGFILE``

  The file to log virsh debug messages.

- VIRSH_DEFAULT_CONNECT_URI

  The hypervisor to connect to by default. Set this to a URI, in the same format as accepted by the **connect** option. This environment variable is deprecated in favour of the global **LIBVIRT_DEFAULT_URI** variable which serves the same purpose.

- LIBVIRT_DEFAULT_URI

  The hypervisor to connect to by default. Set this to a URI, in the same format as accepted by the **connect** option. This overrides the default URI set in any client config file and prevents libvirt from probing for drivers.

- VISUAL

  The editor to use by the **edit** and related options.

- EDITOR

  The editor to use by the **edit** and related options, if **VISUAL** is not set.

- VIRSH_HISTSIZE

  The number of commands to remember in the command history. The default value is 500.

- LIBVIRT_DEBUG=LEVEL

  Turn on verbose debugging of all libvirt API calls. Valid levels are

  - LIBVIRT_DEBUG=1

    Messages at level DEBUG or above

  - LIBVIRT_DEBUG=2

    Messages at level INFO or above

  - LIBVIRT_DEBUG=3

    Messages at level WARNING or above

  - LIBVIRT_DEBUG=4

    Messages at level ERROR

For further information about debugging options consult *https://libvirt.org/logging.html*

BUGS

Please report all bugs you discover. This should be done via either:

1. the mailing list

   https://libvirt.org/contact.html

2. the bug tracker

   https://libvirt.org/bugs.html

Alternatively, you may report bugs to your software distributor / vendor.

AUTHORS

Please refer to the AUTHORS file distributed with libvirt.

COPYRIGHT

Copyright (C) 2005, 2007−2015 Red Hat, Inc., and the authors listed in the libvirt AUTHORS file.

LICENSE

virsh is distributed under the terms of the GNU LGPL v2+. This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE

SEE ALSO

virt−install(1), virt−xml−validate(1), virt−top(1), virt−df(1), *https://libvirt.org/*