NAME

bhyve - run a guest operating system inside a virtual machine

SYNOPSIS

bhyve [**-AaCDeHhPSuWwxY**] [**-c** [[**cpus=**]*numcpus*][**,sockets=***n*][**,cores=***n*][**,threads=***n*]] [**-f** *name***,**[**string**|**file**]=*data*] [**-G** [*w*][*bind_address***:**]*port*] [**-k** *config_file*] [**-K** *layout*] [**-l** *lpcdev*[**,***conf*]] [**-m** *memsize*[**K**|**k**|**M**|**m**|**G**|**g**|**T**|**t**]] [**-o** *var*=*value*] [**-p** *vcpu***:***hostcpu*] [**-r** *file*] [**-s** *slot***,***emulation*[**,***conf*]] [**-U** *uuid*] *vmname*

bhyve -l help

bhyve -s help

DESCRIPTION

bhyve is a hypervisor that runs guest operating systems inside a virtual machine.

Parameters such as the number of virtual CPUs, amount of guest memory, and I/O connectivity can be specified with command-line parameters.

If not using a boot ROM, the guest operating system must be loaded with bhyveload(8) or a similar boot loader before running **bhyve**, otherwise, it is enough to run **bhyve** with a boot ROM of choice.

bhyve runs until the guest operating system reboots or an unhandled hypervisor exit is detected.

OPTIONS

-A Generate ACPI tables. Required for FreeBSD/amd64 guests.

-a The guest's local APIC is configured in xAPIC mode. The xAPIC mode is the default setting so this option is redundant. It will be deprecated in a future version.

-C Include guest memory in core file.

-c [*setting* ...]

Number of guest virtual CPUs and/or the CPU topology. The default value for each of *numcpus*, *sockets*, *cores*, and *threads* is 1. If *numcpus* is not specified then it will be calculated from the other arguments. The topology must be consistent in that the *numcpus* must equal the product of *sockets*, *cores*, and *threads*. If a *setting* is specified more than once the last one has precedence.

The maximum number of virtual CPUs defaults to the number of active physical CPUs in the system available via the *hw.vmm.maxcpu* sysctl(8) variable. The limit can be adjusted via the *hw.vmm.maxcpu* loader tunable.

-D Destroy the VM on guest initiated power-off.

-e Force **bhyve** to exit when a guest issues an access to an I/O port that is not emulated. This is intended for debug purposes.

-f *name*,[**string**|**file**]=*data*

Add a fw_cfg file *name* to the fw_cfg interface. If a **string** is specified, the fw_cfg file contains the string as data. If a **file** is specified, bhyve reads the file and adds the file content as fw_cfg data.

-G [*w*][*bind_address*:]*port*

Start a debug server that uses the GDB protocol to export guest state to a debugger. An IPv4 TCP socket will be bound to the supplied *bind_address* and *port* to listen for debugger connections. Only a single debugger may be attached to the debug server at a time. If the option begins with 'w', **bhyve** will pause execution at the first instruction waiting for a debugger to attach.

-H Yield the virtual CPU thread when a HLT instruction is detected. If this option is not specified, virtual CPUs will use 100% of a host CPU.

-h Print help message and exit.

-k *config_file*

Set configuration variables from a simple, key-value config file. Each line of the config file is expected to consist of a config variable name, an equals sign

-K *layout* Specify the keyboard layout. The value that can be specified sets the file name in */usr/share/bhyve/kbdlayout*. This specification only works when loaded with UEFI mode for VNC. When using a VNC client that supports QEMU Extended Key Event Message (e.g. TigerVNC), this option isn't needed. When using a VNC client that doesn't support QEMU Extended Key Event Message (e.g. tightVNC), the layout defaults to the US keyboard unless specified otherwise. ('='), and a value. No spaces are permitted between the variable name, equals sign, or value. Blank lines and lines starting with '#' are ignored. See bhyve_config(5) for more details.

-l help Print a list of supported LPC devices.

-l *lpcdev*[**,***conf*]

Allow devices behind the LPC PCI-ISA bridge to be configured. The only supported devices are the TTY-class devices **com1**, **com2**, **com3**, and **com4**, the TPM module **tpm**,

the boot ROM device **bootrom**, the **fwcfg** type and the debug/test device **pc-testdev**.

The possible values for the *conf* argument are listed in the **-s** flag description.

-m *memsize*[**K**|**k**|**M**|**m**|**G**|**g**|**T**|**t**]
Set the guest physical memory size This must be the same size that was given to bhyveload(8).

The size argument may be suffixed with one of **K**, **M**, **G** or **T** (either upper or lower case) to indicate a multiple of kilobytes, megabytes, gigabytes, or terabytes. If no suffix is given, the value is assumed to be in megabytes.

The default is 256M.

-o *var*=*value*
Set the configuration variable *var* to *value*.

-P Force the guest virtual CPU to exit when a PAUSE instruction is detected.

-p *vcpu*:*hostcpu*
Pin guest's virtual CPU *vcpu* to *hostcpu*.

-r *file* Resume a guest from a snapshot. The guest memory contents are restored from *file*, and the guest device and vCPU state are restored from the file "*file*.kern".

Note that the current snapshot file format requires that the configuration of devices in the new VM match the VM from which the snapshot was taken by specifying the same **-s** and **-l** options. The count of vCPUs and memory configuration are read from the snapshot.

-S Wire guest memory.

-s help Print a list of supported PCI devices.

-s *slot*,*emulation*[,*conf*]
Configure a virtual PCI slot and function.

bhyve provides PCI bus emulation and virtual devices that can be attached to slots on the bus. There are 32 available slots, with the option of providing up to 8 functions per slot.

The *slot* can be specified in one of the following formats:

- *pcislot*
- *pcislot*:*function*
- *bus*:*pcislot*:*function*

The *pcislot* value is 0 to 31. The optional *function* value is 0 to 7. The optional *bus* value is 0 to 255. If not specified, the *function* value defaults to 0. If not specified, the *bus* value defaults to 0.

The *emulation* argument can be one of the following:

hostbridge A simple host bridge. This is usually configured at slot 0, and is required by most guest operating systems.

amd_hostbridge Emulation identical to **hostbridge** using a PCI vendor ID of AMD.

passthru PCI pass-through device.

virtio-net Virtio network interface.

virtio-blk Virtio block storage interface.

virtio-scsi Virtio SCSI interface.

virtio-9p Virtio 9p (VirtFS) interface.

virtio-rnd Virtio RNG interface.

virtio-console Virtio console interface, which exposes multiple ports to the guest in the form of simple char devices for simple IO between the guest and host userspaces.

virtio-input Virtio input interface.

ahci AHCI controller attached to arbitrary devices.

ahci-cd AHCI controller attached to an ATAPI CD/DVD.

ahci-hd AHCI controller attached to a SATA hard drive.

e1000 Intel e82545 network interface.

uart PCI 16550 serial device.

lpc LPC PCI-ISA bridge with COM1, COM2, COM3, and COM4 16550 serial ports, a boot ROM, and, optionally, a fwcfg type and the debug/test device. The LPC bridge emulation can only be configured on bus 0.

fbuf Raw framebuffer device attached to VNC server.

xhci eXtensible Host Controller Interface (xHCI) USB controller.

nvme NVM Express (NVMe) controller.

hda High Definition Audio Controller.

The optional parameter *conf* describes the backend for device emulations. If *conf* is not specified, the device emulation has no backend and can be considered unconnected.

Network device backends:

- **tap***N*[**,mac=***xx:xx:xx:xx:xx:xx*][**,mtu=***N*]
- **vmnet***N*[**,mac=***xx:xx:xx:xx:xx:xx*][**,mtu=***N*]
- **netgraph,path=***ADDRESS***,peerhook=***HOOK*[**,socket=***NAME*][**,hook=***HOOK*][**,mac=***xx:xx:xx:xx:xx:x*

If **mac** is not specified, the MAC address is derived from a fixed OUI and the remaining bytes from an MD5 hash of the slot and function numbers and the device name.

The MAC address is an ASCII string in ethers(5) format.

With **virtio-net** devices, the **mtu** parameter can be specified to inform the guest about the largest MTU that should be allowed, expressed in bytes.

With **netgraph** backend, the **path** and **peerhook** parameters must be specified to set the destination node and corresponding hook. The optional parameters **socket** and **hook** may be used to set the ng_socket(4) node name and source hook. The *ADDRESS*, *HOOK*, and *NAME* must comply with netgraph(4) addressing rules.

Block storage device backends:

- */filename*[**,***block-device-options*]
- */dev/xxx*[**,***block-device-options*]

The *block-device-options* are:

nocache Open the file with O_DIRECT.

direct Open the file using O_SYNC.

ro Force the file to be opened read-only.

sectorsize=*logical*[/*physical*]
Specify the logical and physical sector sizes of the emulated disk. The physical sector size is optional and is equal to the logical sector size if not explicitly specified.

nodelete Disable emulation of guest trim requests via DIOCGDELETE requests.

bootindex=*index*
Add the device to the bootorder at *index*. A fwcfg file is used to specify the bootorder. The guest firmware may ignore or doesn't support this fwcfg file. In that case, this feature doesn't work as expected.

SCSI device backends:

- */dev/cam/ctl*[*pp***.***vp*][**,***scsi-device-options*]

The *scsi-device-options* are:

iid=*IID* Initiator ID to use when sending requests to specified CTL port. The default value is 0.

bootindex=*index*
Add the device to the bootorder at *index*. A fwcfg file is used to specify the bootorder. The guest firmware may ignore or doesn't support this fwcfg file. In that case, this feature doesn't work as expected.

9P device backends:

- *sharename=/path/to/share*[**,***9p-device-options*]

The *9p-device-options* are:

ro Expose the share in read-only mode.

TTY device backends:

stdio Connect the serial port to the standard input and output of the **bhyve** process.

/dev/xxx Use the host TTY device for serial port I/O.

TPM device backends:

type,*path*[**,***tpm-device-options*]
Emulate a TPM device.

The *tpm-device-options* are:

version=*version*
Version of the TPM device according to the TCG specification. Defaults to **2.0**

Boot ROM device backends:

romfile[**,***varfile*]
Map *romfile* in the guest address space reserved for boot firmware. If *varfile* is provided, that file is also mapped in the boot firmware guest address space, and any modifications the guest makes will be saved to that file.

Fwcfg types:

fwcfg The fwcfg interface is used to pass information such as the CPU count or ACPI tables to the guest firmware. Supported values are 'bhyve' and 'qemu'. Due to backward compatibility reasons, 'bhyve' is the default option. When 'bhyve' is used, bhyve's fwctl interface is used. It currently reports only the CPU count to the guest firmware. The 'qemu' option uses QEMU's fwcfg interface. This interface is widely used and allows user-defined information to be passed to the guest. It is used for passing the CPU count, ACPI tables, a boot order and many other things to the guest. Some operating systems such as Fedora CoreOS can be configured by qemu's fwcfg interface as well.

Pass-through device backends:

- **ppt***N*[,*passthru-device-options*]
- *bus*/*slot*/*function*[,*passthru-device-options*]
- **pci***bus*:*slot*:*function*[,*passthru-device-options*]

Connect to a PCI device on the host either named ppt *N* or at the selector described by *slot*, *bus*, and *function* numbers.

The *passthru-device-options* are:

rom=*romfile*
: Add *romfile* as option ROM to the PCI device. The ROM will be loaded by firmware and should be capable of initializing the device.

bootindex=*index*
: Add the device to the bootorder at *index*. A fwcfg file is used to specify the bootorder. The guest firmware may ignore or doesn't support this fwcfg file. In that case, this feature doesn't work as expected.

Guest memory must be wired using the **-S** option when a pass-through device is configured.

The host device must have been reserved at boot-time using the *pptdevs* loader variable as described in vmm(4).

TPM devices:

type Specifies the type of the TPM device.

Supported types:

passthru

version=*version*
: The *version* of the emulated TPM device according to the TCG specification.

Supported versions:

2.0

Virtio console device backends:

- **port1**=*/path/to/port1.sock*[**,port***N*=*/path/to/port2.sock* ...]

A maximum of 16 ports per device can be created. Every port is named and corresponds to a Unix domain socket created by **bhyve**. **bhyve** accepts at most one connection per port at a time.

Limitations:

- Due to lack of destructors in **bhyve**, sockets on the filesystem must be cleaned up manually after **bhyve** exits.
- There is no way to use the "console port" feature, nor the console port resize at present.
- Emergency write is advertised, but no-op at present.

Virtio input device backends:

/dev/input/eventX
: Send input events of */dev/input/eventX* to guest by VirtIO Input Interface.

Framebuffer devices backends:

- [**rfb**=*ip-and-port*][**,w**=*width*][**,h**=*height*][**,vga**=*vgaconf*][**,wait**][**,password**=*password*]

Configuration options are defined as follows:

rfb=*ip-and-port* (or **tcp**=*ip-and-port*)
: An IP address and a port VNC should listen on. There are two formats:

  - [*IPv4***:**]*port*
  - **[***IPv6%zone***]:***port*

  The default is to listen on localhost IPv4 address and default VNC port 5900. An IPv6 address must be enclosed in square brackets and may contain an optional zone identifier.

w=*width* and **h**=*height*
: A display resolution, width and height, respectively. If not specified, a default resolution

of 1024x768 pixels will be used. Minimal supported resolution is 640x480 pixels, and maximum is 1920x1200 pixels.

vga=*vgaconf*

Possible values for this option are **io** (default), **on** , and **off**. PCI graphics cards have a dual personality in that they are standard PCI devices with BAR addressing, but may also implicitly decode legacy VGA I/O space (*0x3c0-3df*) and memory space (64KB at *0xA0000*). The default **io** option should be used for guests that attempt to issue BIOS calls which result in I/O port queries, and fail to boot if I/O decode is disabled.

The **on** option should be used along with the CSM BIOS capability in UEFI to boot traditional BIOS guests that require the legacy VGA I/O and memory regions to be available.

The **off** option should be used for the UEFI guests that assume that VGA adapter is present if they detect the I/O ports. An example of such a guest is OpenBSD in UEFI mode.

Please refer to the **bhyve** FreeBSD wiki page (**https://wiki.freebsd.org/bhyve**) for configuration notes of particular guests.

wait Instruct **bhyve** to only boot upon the initiation of a VNC connection, simplifying the installation of operating systems that require immediate keyboard input. This can be removed for post-installation use.

password=*password*

This type of authentication is known to be cryptographically weak and is not intended for use on untrusted networks. Many implementations will want to use stronger security, such as running the session over an encrypted channel provided by IPsec or SSH.

xHCI USB device backends:

tablet A USB tablet device which provides precise cursor synchronization when using VNC.

NVMe device backends:

- *devpath*[**,maxq=**#][**,qsz=**#][**,ioslots=**#][**,sectsz=**#][**,ser=**#][**,eui64=**#][**,dsm=***opt*]

Configuration options are defined as follows:

devpath Accepted device paths are: */dev/blockdev* or */path/to/image* or **ram=***size_in_MiB*.

maxq Max number of queues.

qsz Max elements in each queue.

ioslots Max number of concurrent I/O requests.

sectsz Sector size (defaults to blockif sector size).

ser Serial number with maximum 20 characters.

eui64 IEEE Extended Unique Identifier (8 byte value).

dsm DataSet Management support. Supported values are: **auto**, **enable**, and **disable**.

AHCI device backends:

- [[**hd:**|**cd:**]*path*][**,nmrr=***nmrr*][**,ser=***#*][**,rev=***#*][**,model=***#*]

Configuration options are defined as follows:

nmrr Nominal Media Rotation Rate, known as RPM. Value 1 will indicate device as Solid State Disk. Default value is 0, not report.

ser Serial Number with maximum 20 characters.

rev Revision Number with maximum 8 characters.

model Model Number with maximum 40 characters.

HD Audio device backends:

- [**play=***playback*][**,rec=***recording*]

Configuration options are defined as follows:

play Playback device, typically */dev/dsp0*.

rec Recording device, typically */dev/dsp0*.

Set the universally unique identifier (UUID) in the guest's System Management BIOS System Information structure. By default a UUID is generated from the host's hostname and *vmname*.

RTC keeps UTC time.
Force virtio PCI device emulations to use MSI interrupts instead of MSI-X interrupts.
Ignore accesses to unimplemented Model Specific Registers (MSRs). This is intended for debug purposes.
The guest's local APIC is configured in x2APIC mode.
Disable MPtable generation.
Alphanumeric name of the guest. This should be the same as that created by bhyveload(8).

CONFIGURATION VARIABLES

bhyve uses an internal tree of configuration variables to describe global and per-device settings. When **bhyve** starts, it parses command line options (including config files) in the order given on the command line. Each command line option sets one or more configuration variables. For example, the **-s** option creates a new tree node for a PCI device and sets one or more variables under that node including the device model and device model-specific variables. Variables may be set multiple times during this parsing stage with the final value overriding previous values.

Once all of the command line options have been processed, the configuration values are frozen. **bhyve** then uses the value of configuration values to initialize device models and global settings.

More details on configuration variables can be found in bhyve_config(5).

DEBUG SERVER

The current debug server provides limited support for debuggers.

Registers

Each virtual CPU is exposed to the debugger as a thread.

General purpose registers can be queried for each virtual CPU, but other registers such as floating-point and system registers cannot be queried.

Memory

Memory (including memory mapped I/O regions) can be read and written by the debugger. Memory operations use virtual addresses that are resolved to physical addresses via the current virtual CPU's active address translation.

Control

The running guest can be interrupted by the debugger at any time (for example, by pressing Ctrl-C in the debugger).

Single stepping is only supported on Intel CPUs supporting the MTRAP VM exit.

Breakpoints are supported on Intel CPUs that support single stepping. Note that continuing from a breakpoint while interrupts are enabled in the guest may not work as expected due to timer interrupts firing while single stepping over the breakpoint.

SIGNAL HANDLING

bhyve deals with the following signals:

SIGTERM Trigger ACPI poweroff for a VM

EXIT STATUS

Exit status indicates how the VM was terminated:

0 rebooted
1 powered off
2 halted
3 triple fault
4 exited due to an error

EXAMPLES

If not using a boot ROM, the guest operating system must have been loaded with bhyveload(8) or a similar boot loader before bhyve(4) can be run. Otherwise, the boot loader is not needed.

To run a virtual machine with 1GB of memory, two virtual CPUs, a virtio block device backed by the */my/image* filesystem image, and a serial port for the console:

```
bhyve -c 2 -s 0,hostbridge -s 1,lpc -s 2,virtio-blk,/my/image \
  -l com1,stdio -A -H -P -m 1G vm1
```

Run a 24GB single-CPU virtual machine with three network ports, one of which has a MAC address specified:

```
bhyve -s 0,hostbridge -s 1,lpc -s 2:0,virtio-net,tap0 \
  -s 2:1,virtio-net,tap1 \
  -s 2:2,virtio-net,tap2,mac=00:be:fa:76:45:00 \
  -s 3,virtio-blk,/my/image -l com1,stdio \
  -A -H -P -m 24G bigvm
```

Run an 8GB quad-CPU virtual machine with 8 AHCI SATA disks, an AHCI ATAPI CD-ROM, a single virtio network port, an AMD hostbridge, and the console port connected to an nmdm(4) null-modem device.

```
bhyve -c 4 \
  -s 0,amd_hostbridge -s 1,lpc \
  -s 1:0,ahci,hd:/images/disk.1,hd:/images/disk.2,\
hd:/images/disk.3,hd:/images/disk.4,\
hd:/images/disk.5,hd:/images/disk.6,\
hd:/images/disk.7,hd:/images/disk.8,\
cd:/images/install.iso \
  -s 3,virtio-net,tap0 \
  -l com1,/dev/nmdm0A \
  -A -H -P -m 8G
```

Run a UEFI virtual machine with a display resolution of 800 by 600 pixels that can be accessed via VNC at: 0.0.0.0:5900.

```
bhyve -c 2 -m 4G -w -H \
  -s 0,hostbridge \
  -s 3,ahci-cd,/path/to/uefi-OS-install.iso \
  -s 4,ahci-hd,disk.img \
  -s 5,virtio-net,tap0 \
  -s 29,fbuf,tcp=0.0.0.0:5900,w=800,h=600,wait \
  -s 30,xhci,tablet \
  -s 31,lpc -l com1,stdio \
  -l bootrom,/usr/local/share/uefi-firmware/BHYVE_UEFI.fd \
   uefivm
```

Run a UEFI virtual machine with a VNC display that is bound to all IPv6 addresses on port 5900.

```
bhyve -c 2 -m 4G -w -H \
  -s 0,hostbridge \
  -s 4,ahci-hd,disk.img \
  -s 5,virtio-net,tap0 \
  -s 29,fbuf,tcp=[::]:5900,w=800,h=600 \
  -s 30,xhci,tablet \
  -s 31,lpc -l com1,stdio \
  -l bootrom,/usr/local/share/uefi-firmware/BHYVE_UEFI.fd \
   uefivm
```

Run a UEFI virtual machine with a VARS file to save EFI variables. Note that **bhyve** will write guest modifications to the given VARS file. Be sure to create a per-guest copy of the template VARS file from */usr*.

```
bhyve -c 2 -m 4g -w -H \
  -s 0,hostbridge \
  -s 31,lpc -l com1,stdio \
  -l bootrom,/usr/local/share/uefi-firmware/BHYVE_UEFI_CODE.fd,BHYVE_UEFI_VARS.fd
   uefivm
```

SEE ALSO

bhyve(4), netgraph(4), ng_socket(4), nmdm(4), vmm(4), bhyve_config(5), ethers(5), bhyvectl(8), bhyveload(8)

Intel, *64 and IA-32 Architectures Software Developer's Manual*, Volume 3.

HISTORY

bhyve first appeared in FreeBSD 10.0.

AUTHORS

Neel Natu <*neel@freebsd.org*>
Peter Grehan <*grehan@freebsd.org*>