

# ctys-setupVDE(1)

manages on-the-fly network configuration

March 10, 2011

## Contents

<b>1</b>	<b>NAME</b>	<b>2</b>
<b>2</b>	<b>SYNTAX</b>	<b>2</b>
<b>3</b>	<b>DESCRIPTION</b>	<b>2</b>
<b>4</b>	<b>OPTIONS</b>	<b>6</b>
<b>5</b>	<b>ARGUMENTS</b>	<b>7</b>
<b>6</b>	<b>EXIT-VALUES</b>	<b>7</b>
<b>7</b>	<b>SEE ALSO</b>	<b>8</b>
<b>8</b>	<b>AUTHOR</b>	<b>8</b>
<b>9</b>	<b>COPYRIGHT</b>	<b>8</b>

## 1 NAME

**ctys-setupVDE** - manages on-the-fly network configuration

## 2 SYNTAX

**ctys-setupVDE**

```
[-b <virtual-bridge>]
[-d <level>]
[-f]
[-g <sbit-group>*]
[-h]
[-H <help-options>]
[-i <interface>]
[-l <remote-user>]
[-r <remote-hosts>]
[-s <ALTERNATE-QEMUSOCK>]
[-S <ALTERNATE-QEMUMGMT>]
[-u <non-privileged-user>[.<group>]]
[-n]
[-V]
[-X]
[-Z <set-sudo-ksu>]
(cancel|check|create|info|ports|list|listall)
```

## 3 DESCRIPTION

**ctys-setupVDE** creates and manages the complete set of Network-Devices required for interconnection by KVM and QEMU. The creation and deletion is performed by just one call. Therefore **ctys-setupVDE** encapsulates and combines the subset of functionality of required tools supporting the TAP/TUN devices by VDE.

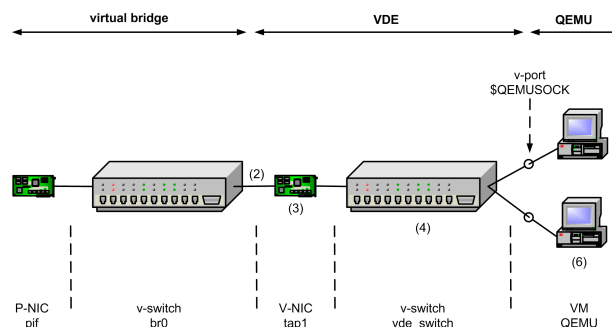


Figure 1: QEMU/KVM-Network Interconnection

The utility could be performed locally or remotely by full support of remote ctys-addressing, including context specific target-options, MACROS and GROUPS. E.g. the required system permissions could be preconfigured for specific users by "ksu" and/or "sudo", for additional information refer to 'VDE Remote Configuration'.

- **ATTENTION:** The remote-execution includes some inherent pit-falls to be considered thoroughly! This is the case, when this utility has to be executed on a remote site, where not yet a bridge (the only supported networking device for now) exists. During the creation of the required bridge - the so called

'main virtual-bridge' the network is disconnected for a short time, so any access to NFS or any other networked file system is interrupted temporarily, which eventually leads to missing of additional tools required for call-completion, e.g. for reconnecting to the network.

The same is true for authentication, when kerberos based "ksu" or "sudo", or any other network centric authentication is used in a non-cached environment, so for non-root users the access to system resources is frequently rejected. Particularly the reconnection of the network device.

Thus remote execution is not approved for users with a mounted remote-home, even though it might work under specific conditions. Local-only users with "sudo" control by complete locally configured environments are verified to work stable.

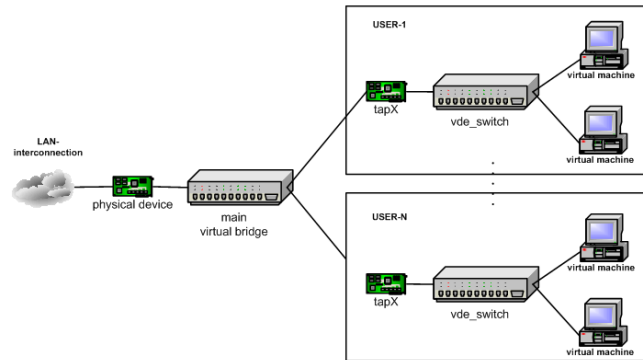


Figure 2: QEMU/KVM-Network Interconnection - Components

A specific behaviour of the current version is applied to the created main-bridges. These will get the same IP and MAC addresses as the logical interface, anyhow it works perfectly, as long as you can cope with multiple interfaces with same address information within applied tools. For the functionality of the UnifiedSessionsManager this is handled by a "sort -u" on resulting enumeration IF-lists. The current debian setup works the same way, where even the name of the first interface is reused as the name of the created bridge.

One reason for "doing" the bridge allocation this way within the UnifiedSessionsManager is the minimized risk of detaching the remotely handled VMs for too long from the network services, which might make them unusable from then on. This aspect has to be emphasized due to the intention of frequent on-the-fly creation of networking devices. This naming and address-assignment concept will probably be modified slightly in future versions. Anyhow, the remote usage of "ctys-setupVDE", once the authentication is configured properly and security facilities are setup thoroughly, offers a simple interface for centralized setup of VM stacks. This is particularly true in combination of remote usage of GENMCONF and PLUGINS.

The usage of ctys-setupVDE assures the appropriate environment for the used of the wrappers "vdeq" and "vdeqemu" of the package VDE-SOURCEFORGE, which is the recommended tool when TAPTUN-byVDE has to be created. This utility could be used in any comparable case too, but fit particularly for QEMU setup.

The configuration files if QEMU are shared here, thus the consistency of QEMUSOCK is assured. The variable QEMUSOCK is based on the variable CTYS\_SOCKETBASE, which is the default base directory, where UNIX domain sockets are created. This should be used for eventual additional UNIX domain sockets, such as tcp based serial ports or monitoring devices, too. For additional applicability refer to the user manual of QEMU and to the templates provided by UnifiedSessionsManager.

The following tools are combined within this script:

- vde\_tunctl
- vde\_switch

- unixterm
- nc
- brctl
- ifconfig
- /etc/init.d/network

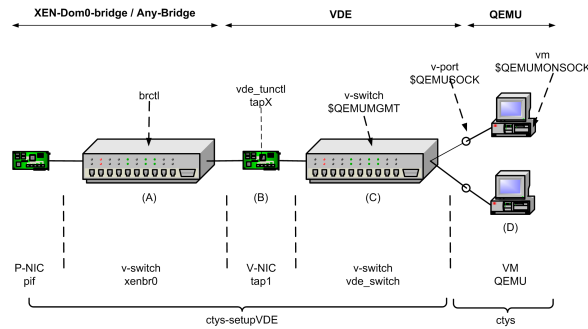


Figure 3: QEMU/KVM-Network Interconnection - Details

Two types of virtual bridges/switches( see figure:NestedProtocolStacks are managed by ctys-setupVDE.

- "main virtual-bridge"
  - The switch to be used for interconnecting the "external" interface, which is in case of the hosting machine itself a physical NIC.
  - This switch is created if not yet present, but has to be deleted manually by the user.
- "vde\_switch"
  - The switch to be used for attachment of VMs. This switch is completely managed by ctys-setupVDE.

TAPTUNbyVDE prepares a TAP device with a attached new bridge, therefore it requires the VirtualDistributedEthernet - sourceforgeVde package. Additional information within a Wiki containing some helpful tutorials for virtualsquare-basicnet working could be found at the website of VirtualSquare.

In current implementation some assumptions are made in order to ease design and implementation. Anyhow, for practical application these constraints might not be an important matter.

- one TAP for each vde\_switch
- each user has one switch which communicates by default via `"/var/tmp/vde_switch0.$USER"`.
- the management interface for each switch is by default `"/var/tmp/vde_mgmt0.$USER"`.
- appropriate access permissions are provided by sudo or ksu, for automatic detection the ctys framework is used

#### ATTENTION:

The default bridge used is the first found in alphabetical order. For some default installations this might not be the intended. When DHCP is used the first to check in case of errors is the actual used bridge. The bridge to be used could be forced by the **-b** option.

The following steps are performed by ctys-setupVDE:

1. Creation of a TAP device.

```
"vde_tunctl -u <user-without-root-permission>"
```

e.g.

```
vde_tunctl -u acue
```

Returns a line like:

```
Set 'tap3' persistent and owned by uid 4711
```

2. Use the returned 'tapX' for networking.

```
ifconfig $1 0.0.0.0 up  
brctl addif $2 $1
```

Does the same as:

```
/etc/xen/qemu-ifup tap3 xenbr0
```

Which brings up the newly created interface 'tap3' and adds an interface to the virtual Xen bridge connecting it to the world outside.

The results could be verified with:

- `ifconfig tap3`  
should list an interface 'tap3'
- `brctl show`  
should contain an interface 'tap3'

3. Connect the device.

Now this interface will be connected to another virtual switch, the `vde_switch` in order to provide an internal multiplexer for multiple QEMU instances to be connected to the external interfaces e.g. via a present Xen-bridge.

```
QEMUSOCK=/var/tmp/vde_switch0.$USER  
QEMUMGMT=/var/tmp/vde_mgmt0.$USER
```

```
vde_switch -d \  
            -tap tap3 \  
            -s ${QEMUSOCK} \  
            -M ${QEMUMGMT}
```

```
chown -R <userX.groupX> ${QEMUSOCK}  
chown -R <userX.groupX> ${QEMUMGMT}
```

The state could be verified with:

```
QEMUMGMT=/var/tmp/vde_mgmt0.$USER
```

```
unixterm ${QEMUMGMT}
```

For additional information refer to examples of the manual.

## 4 OPTIONS

### -b <virtual-bridge>

The virtual bridge connected to the external network to be attached by TAP device. Default is to use the first bridge detected by brctl. If none is present, tha by default a new one is created with the name "ctysbr0", and the first found interface is added to the bridge.

When an interface is provided by "-i" option and a new bridge has to be created, this will be used instead of the first valid.

### -d <level>

Sets debug.

### -f

Forces execution even when processing seems to be critical.

- Forces call of "kill <PID>", when here-script with

```
"unixterm ... shutdown"
```

fails. For current version this seems to be frequently the case on i386 architecture, whereas x86\_64 works.

- Creates a new bridge, even when connected via a network session. This could interrupt the current calling session permanently, even lead to it's hang-up due to a required short-time disconnect. So this should preferrably proceeded from within a local session.

### -g <sbit-group>

Sets the s-bit for the group, this has to be the same as the resulting owner's group.

If not set, the resulting permissions for QEMUSOCK are

```
"rwx-----"
```

else

```
"rwx--S---"
```

### -h

Print help, refer to "-H" for additional information.

### -H <help-option>

The extended help option is based on system interfaces for display of manpages, PDF and HTML documents. This comprises the man pages and installed manuals.

For additional help refer to the documents or type **ctys -H help**.

### -i <interface>

The interface to be added to a newly created bridge, see "-b" option.

### -l <remote-user>

Refer to "ctys" generic options for additional information.

### -r <remote-hosts>

List of remote hosts for execution. Either a list of valid hostnames, ipaddresses, or EMail-Format hostnames.

### -s <ALTERNATE-QEMUSOCK>

A file-socket to be used for communications peer via virtual switch. Default is set by common QEMUSOCK configuration.

### -S <ALTERNATE-QEMUMGMT>

A file-socket to be used for management console of virtual switch. Default is set by common QEMUMGMT configuration.

Could be used with "unixterm \$QEMUMGMT" of VDE.

- u** `<non-privileged-user>[.<group>]`  
Owner of the created TAP device. Default is current user.
- V**  
Version.
- X**  
See ctys, terse for machine output.
- Z**  
See ctys.

## 5 ARGUMENTS

- `<command>`

- **cancel** removes the switch and it's attached TAP device. In case of partial present resources these will be cleared as present, thus remaining parts of partly execution could be reset.
- **check** performs basic check for the accesibility of the virtual switch etup for selected USER. Therefore a simple "ctys-setupVDE PORTS" call is analysed for the occurance of at least one "tap" device and one UNIX-Domain socket, which are verified by their existence. In case of erroneous state basic information for further analysis is displayed. Anyhow, still malfunction could occur, but if check fails, it will definetly.
- **create** a new virtual switch, this comprises a new TAP device and an attached virtual switch. When no bridge is present a virtual bridge is created too, and the tap-device is attached.  
The CREATE call just checks whether a functional switch is already present, if not it just creates a new one. Therefore the current defined socket for the management interface is utilized. Thus a new call on a present, but erroneous switch leads to reuse of the sockets, but creates a new tap-device and starts a new instance of a vde-switch-process. Present tap-devices are not reused, and just kept untouched.
- **info** shows vde\_switch information. This is the default behaviour.
- **ports** lists ports of vde\_switch.
- **list** lists present vde\_switch-es. The base-switch entries are displayed only.
- **listall** lists present vde\_switch-es. Any entry is displayed, this includes the dynamic created port specific sockets.  
Due to some minor difficulties for now these are not removed, when the client disappears, thus "listall" could be used to check the dengling entries from time to time.

## 6 EXIT-VALUES

- 0: OK:** Result is valid.
- 1: NOK:** Erroneous parameters.
- 2: NOK:** Missing an environment element like files or databases.

## 7 SEE ALSO

**ctys use-cases** *ctys-QEMU(7)*

**ctys plugins PMs** *ctys-PM(1)*

**VMs** *ctys-QEMU(1)*

**HOSTS** *ctys-CLI(1)*, *ctys-PM(7)*, *ctys-VNC(7)*, *ctys-X11(7)*

**ctys executables** *ctys-distribute(1)*, *ctys-extractARPlst(1)*, *ctys-extractMAClst(1)*, *ctys-genmconf(1)*, *ctys-vping(1)*, *ctys-plugins(1)*, *ctys-vhost(1)*, *ctys-wakeup(1)*

**system executables** *vde\_tunctl*, *vde\_switch*, *unixterm*, *ifconfig(8)*, *brctl(8)*, *ether-tool(8)*, *nc(1)* <a.k.a. *netcat*>

## 8 AUTHOR

Maintenance: <[acue\\_sf1@users.sourceforge.net](mailto:acue_sf1@users.sourceforge.net)>  
Homepage: <<http://www.UnifiedSessionsManager.org>>  
Sourceforge.net: <<http://sourceforge.net/projects/ctys>>  
Berlios.de: <<http://ctys.berlios.de>>  
Commercial: <<http://www.i4p.com>>



## 9 COPYRIGHT

Copyright (C) 2008, 2009, 2010, 2011 Ingenieurbuero Arno-Can Uestuenseoz  
For BASE package following licenses apply,

- for software see GPL3 for license conditions,
- for documents see GFDL-1.3 with invariant sections for license conditions,

This document is part of the **DOC package**,

- for documents and contents from DOC package see  
'**Creative-Common-Licence-3.0 - Attrib: Non-Commercial, Non-Deriv**'  
with optional extensions for license conditions.

For additional information refer to enclosed Releasenotes and License files.

