Title	**Manual - vtty**
No.	

Rev.	Date	By	History
1.0.0	2003-12-4	UIS	Initial draft
1.1.0	2005-07-05	JOJ	Added the Linux kernel 2.6 support
1.1.2	2005-07-18	JOJ	Changes in vtty v1.1.2 are added

1 Introduction

vtty consists of **sena_vpty**, **ttyd** and **vtty_manager**.
sena_vpty is a device driver module that is dynamically loaded into the Linux kernel. **ttyd** is a terminal server application. **vtty_manager** is a wrap-up application that provides an in-hand way of using the driver module, pseudo-tty devices and terminal server application.

2 Compile and installation

Get and copy vtty-1.1.2.tar.gz to somewhere in your Linux machine.

```
#tar –xvzf vtty-1.1.2.tar.gz
#cd vtty
#make
#make install
```

Note:

1. All the binaries are installed in “/usr/local/bin” and the compiled kernel module is copied to “/usr/local/vpty” at the install time. An initialization script “vtty init” is appended to the end of “/etc/rc.local“ as well.
2. Some Linux distribution does not run /etc/rc.local at boot time.
   For this case, “vtty init” should be run by manual at every system reboot.
   You can confirm whether your machine run rc.local by checking following command command just after system booting,
   # lsmod | grep sena_vpty
3. To use vtty under custom 2.4.x kernel, you should have the corresponding kernel source that is running on your machine and also /usr/src/linux-2.4 should be linked to the top directory of kernel source.
4. To use vtty under Linux kernel 2.6.x, you should have the configured kernel source that is running on your machine.
   For the configuration and compilation of kernel, please refer to the related documents from the each Linux distributions.
5. Default Linux kernel from some Linux distributions (such as Fedora) does not include Legacy PTY support so you need to recompile your kernel with following two options in kernel configuration enabled.

   - Device Drivers->Character devices->(*) Legacy (BSD) PTY support
   - Device Drivers->Character devices->(256) Maximum number of legacy PTY in use

   You can check whether your kernel includes the Legacy PTY support by checking /proc/tty/drivers. If there are two drivers which use ttyp and pty node, then you have the kernel with Legacy PTY support.
   # cat /proc/tty/drivers
   …
   pty_slave /dev/ttyp 3 0-255 pty:slave

pty_master /dev/pty 2 0-255 pty:master
…

3 Kernel driver module

After installation, with first execution of “vtty init“ or “vtty add {arguments..}”, the driver module for pseudo-tty devices is loaded into the kernel with its name “sena_vpty”. The major numbers of devices are 226 and 227 respectively for master and slave node each. Users can add up to 255 devices and the device files are managed by “**vtty_manager**”.

Note:

If you want to install new kernel driver module, you should run “rmmod sena_vpty” command (unloading old driver) before run “vtty init“ command.

4 Device files

Once users add a node using “vtty add [SUFFIX]” command, there generated a pair of master and slave devices: /dev/vpty[SUFFIX] for master and /dev/tty[SUFFIX] for slave. The “**vtty_manager**“ finds the lowest available minor number ranging from 0 to 255 and assign it for the devices.

5 Database

“**vtty_manager**” utilizes a local database for its data management including nodes(device files), serial options for terminal server side and node status. The database is generated as “/var/vtty/vtty.db”.

6 Syntax

```
vtty { add | addnt } TTYSUFFIX ip_address tcp_port [serial_option]
vtty { enable |enablent | disable | remove } TTYSUFFIX
vtty status [TTYSUFFIX]
```

Note:

The TTYSUFFX is a suffix for the device file name and at the same time, it becomes a management keyword for the vtty database.

```
serial_option: -b <baudrate> -p <port_setting>

        <baudrate>: one of the following

        75, 150, 200, 300, 600, 1200, 1800, 2400, 4800,
        9600, 19200, 38400, 57600, 115200, 230400

        <port_setting>: specified by a string of one of more of the
        following concatenated together with no intervening spaces

        8: 8 bits/character
        7: 7 bits/character
```

6: 6 bits/character
5: 5 bits/character
N: No parity
E: Even parity
O: Odd parity
C0: No hardware flow control
C1: Hardware flow control
S0: No software flow control
S1: Software flow control

If the serial option is not specified, the serial option defaults to : baudrate = 9600, port setting = 8NC0

Example:

```
vtty add mysuffix 192.168.1.1 6001 –b 9600 –p 8NC0
vtty enable mysuffix
vtty status
```

Add(Addnt):

```
vtty add mysuffix 192.168.1.1 6001 –b 9600 –p 8NC0
```

Adds a node information on the local database and makes two device files: master(/dev/vptymysuffix) and slave(/dev/ttymysuffix). And then, invokes a terminal server process "ttyd" and the through-connection to the remote host is ready.

Users might want to use slave device for their application.

The status of the newly added device is "enabled".

If there happens any problem listed in the following in running the vtty, the command is discarded.

- Device add limit exceeded (No more minor number for the new device is available)
- The suffix already exists in the local database
- Failed in loading driver module
- A device file with the same name already exists (Either of master or slave device file)
- Any failure in making device file
- "ttyd" process invocation failure

Note:

1. If there, in the kernel, is not loaded driver module, "add" or "init" command checks the module and loads it into the kernel.
2. If you add a new node with "add" command, some Telnet options are send to the device at the beginning of the connection. Super series, STS series and new Pro series (PS110/410/810) have an option to enable Telnet support. But if you do not want to use the Telnet protocol, please use "addnt" command to disable the Telnet options offering.

Remove:

```
vtty remove mysuffix
```

Removes the node information from the local database and deletes two device files: master(/dev/vptymysuffix) and slave(/dev/ttymysuffix). And then, kills the relevant "ttyd" process running.

Enable(Enablent):

```
vtty enable mysuffix
```

If the state of the node is "disabled", then the command sets the node status "enabled" and invokes a "ttyd" process to be ready to connect. Otherwise, this command has no effects.

Note:

If you enable a node with “enable” command, some Telnet options are send to the device at the beginning of the connection. Super series, STS series and new Pro series (PS110/410/810) have an option to enable Telnet support. But if you do not want to use the Telnet protocol, please use “enablent” command to disable the Telnet options offering.

Disable:

```
vtty disable mysuffix
```

If the state of the node is “enabled”, then the command sets the node status “disabled” and kills relevant “ttyd” process. Otherwise, the command has no effects.

Note: The “disable” command does not delete device files.

Status:

```
vtty status [TTYSUFFIX]
```

Displays the status of the node(s). If TTYSUFFIX is given, it displays the status of the specified node. Else, it shows status of all nodes registered in the local database.

Note: STATUS field in the result is categorized into CONNECTED, DISCONNECTED and DISABLED.

Init:

```
vtty init
```

Initializes the vtty by loading driver module and running “ttyd” processes marked “enabled” in the local database. This command does not have effect twice or more.

Note:

1. This command is appended to the end of “/ect/rc.local” file so to run at the boot-up time of the machine.
2. Some Linux distribution does not run /etc/rc.local at boot time.
   For this case, “vtty init” should be run by manual at every system reboot.
   You can confirm whether your machine run rc.local by checking following command just after system booting,
   # lsmod | grep sena_vpty