Xen and the linux console

or: why xencons={tty,ttyS,xvc} will go away.

by Gerd Hoffmann <kraxel@redhat.com>

So, what is the Linux console?

Well, there isn't a simple answer to that question. Which is the reason for this paper in the first place. For most users it probably is the screen they are sitting in front of. Which is correct, but it also isn't the full story. Especially there are a bunch of CONFIG_*_CONSOLE kernel options referring to two different (but related) subsystems of the kernel.

Introducing virtual terminals (CONFIG_VT)

Well, every linux user knows them: Virtual terminals. Using Alt-Fx you can switch between different terminals. Each terminal has its own device, namely /dev/tty<nr>. Most Linux distributions have a getty ready for text login on /dev/tty{1-6} and the X-Server for the graphical login on /dev/tty7. /dev/tty0 is a special case: It referes to the terminal which is visible at the moment. The VT subsystem doesn't draw the characters itself though, it has hardware specific drivers for that. The most frequently used ones are:

CONFIG_VGA_CONSOLE

> VGA text console driver, this one will drive your VGA card if you boot the machine in VGA text mode.

CONFIG_FRAMEBUFFER_CONSOLE

> Provides text screens on top of a graphical display. The graphical display in turn is driven by yet another driver. On x86 this very often is vesafb. Other platforms have generic drivers too. There are also a bunch of drivers for specific hardware, such as rivafb for nvidia cards. The virtual framebuffer driver for xenified linux kernels is yet another one.

CONFIG_DUMMY_CONSOLE

> This one is funny. The VT subsystem is quite unhappy in case it has no output driver attached. Here comes the dummy driver into play. It will jump in and fill the hole in case there is nobody else. Often the driver is active for a short time at boot only. The VT subsystem initializes quite early at boot, whereas the framebuffer comes much later. As soon as the framebuffer driver is initialized and ready the dummy driver hands over to the framebuffer console:

```
zweiblum kraxel ~# dmesg | grep Console
Console: colour dummy device 80x25
Console: switching to colour frame buffer device 156x60
```

Introducing /dev/console

Here starts the confusion. /dev/console is the linux console device. In many cases it is identical to /dev/tty0. But that is not necessarily the case. Especially /dev/console has nothing to do with the VT subsystem, except that the virtual terminals are one possible console device.

The kernel will start /sbin/init with /dev/console as terminal, thus the boot messages will show up on that device. Also the kernel messages printed by the kernel itself will go to the console device. There are two possible kinds of console devices: The ones which can just output messages and nothing else, and the ones which are associated with a full-featured terminal device.

Here is an incomplete list of console drivers:

CONFIG_VT_CONSOLE
: Linux console on virtual terminals. Most popular one.

CONFIG_SERIAL_*_CONSOLE
: Very often used for headless systems: Linux console on a serial line.

CONFIG_LP_CONSOLE
: Print your kernel messages on paper (print only).

CONFIG_NETCONSOLE
: Dump the kernel messages using NAPI to a log server on the network (print only, patches for input?).

CONFIG_HVC_CONSOLE
: Hypervisor virtual console. To be exact some fancy infrastructure where you just hook in two simple functions to send and receive characters and get a full-featured terminal device. Used by powerpc folks only until recently. I'll come back to that one later.

There are alot more drivers, most of them are architecture-specific.

Oh, and btw: The earlyprintk code is just a simple console too. Output-only, also quite limited otherwise so the implementation doesn't need allocate memory and thus is able to operate very early at boot time.

The XenLinux console, sparse tree

How does the console driver in xenified linux kernels work today?

It is a terminal driver. It registers as console (aka /dev/console) device. That is fine.

It can hijack the console (aka VT subsystem) major/minor number range, and does that by default on guest domains (xencons=tty). That is a dirty hack. It is somewhat convenient as you'll get a getty for login on the console without any configuration work. Trouble is you can't do that and use virtual terminals at the same time. Which is a problem now as the VT subsystem is needed for a console on the virtual framebuffer device we got recently. The xen console code also has to deal with some problems causes by the hijacking, such as providing some dummy terminal devices for /dev/tty{2..8}.

It can also hijack the serial line major/minor numbers, and does that by default on the control domain (xencons=ttyS). Dirty hack too, makes it impossible for the dom0 kernel to use the serial line directly.

Recently (3.0.4 and newer) the console driver got the ability to use the major/minor number assigned for the xen console (xencons=xvc). A step into the right direction. The xen console finally becomes a device of its own, just like any other console device. Needs some configuration work (see below) as long as distributions are not adapted yet, but saves a lot of trouble in the long run.

The XenLinux console, paravirt_ops queue

Remember CONFIG_HVM_CONSOLE mentioned above? The xen console code in the paravirt_ops patch queue goes one step further. It uses the existing infrastructure for hypervisor console devices, brougth to us by the powerpc guys. It handles all the interfacing with the tty layer and the console subsystem. The only thing needed to get a fully-functional console device are two functions: One for writing characters. One for reading characters. And some glue code to register the console. xencons= is gone. The console device is /dev/hvc0, unconditionally.

Linux console configuration guide

Some more background information first. The linux kernel can have multiple console devices compiled in, and usually they actually have at least two compiled in: For the console on top of the virtual terminals, and for a serial console.

The console device which registers first will be used by the linux kernel. Initialization happens in link order, thus essentially the directory ordering in drivers/Makefile decides which console driver will be used by default. It is possible to overwrite the default device on the kernel command line. It is even possible to specify multiple console devices, the kernel boot messages will be printed on all devices then. For /dev/console only one device can be used, the kernel will pick the last one specified on the kernel command line.

The syntax to configure the console device is simply console=name,options. Here are some examples:

console=ttyS0,115200n8
: Classic way to configure a serial console: /dev/ttyS0 is the console device, the serial port will be configured at 115200 baud, no parity, 8bit.

xencons=xvc console=xvc0
: Recommended way to configure the xen console with 3.0.4+ sparse kernels, without virtual framebuffer. /dev/xvc0 becomes the console device.

xencons=xvc console=xvc0 console=tty1
: Recommended way to configure the xen console with 3.0.4+ sparse kernels, when using the framebuffer. /dev/tty1 becomes the console device. Not hijacked but owned by the VT subsystem, thus visible on the virtual framebuffer. The kernel messages are printed to both /dev/tty1 (framebuffer) and /dev/xvc0 (xen console).

console=hvc0
: Recommended way for paravirt_ops kernels. /dev/hvc0 becomes the console device.

It is also very handy to have a getty running on /dev/hvc0 (or /dev/xvc0), so you can login after connecting to the domain using "xm console". Most distributions already have a commented example line in /etc/inittab for a getty on the serial line. Copy, uncomment and adapt it, it should look like this then:

```
h0:12345:respawn:/sbin/agetty -L 9600 hvc0 screen
```

9600 is the line speed, it doesn't matter what you specify here. "hvc0" is the device. "screen" is the terminal type, you might want to use "xterm" or "linux" instead, depending on your favorite terminal application.

If you want be able to login as root via "xm console" you have to add hvc0 to /etc/securetty.

Note that the /dev/hvc0 terminal device is present unconditionally. If you configure something else as console, no problem, it is still there. You'll get neither kernel nor init messages of course. But getty's friendly login prompt will show up once the machine finished booting.

That's it. Enjoy.