Bisoños Usuarios de GNU/Linux de Mallorca y Alredededores | Bergantells Usuaris de GNU/Linux de Mallorca i Afegitons

Graphic boot with bootsplash and progress bar in Debian (70588 lectures)

Per **Pedro Jurado Maqueda**, ***melenas*** (http://www.kdehispano.org)

Creado el 06/07/2003 12:21 modificado el 06/07/2003 12:21

Whoever has installed the SuSE and/or Mandrake last versions, will have seen the graphic boots with progress bar. What is more, contrary to lpp, which needs to compile the image inside the kernel, this system that uses these distributions named bootsplash, let change the background image not recompiling the kernel, giving us the chance of "showing" the boot just pressing F2.

In this article, I explain how to get a graphic boot with Debian, including a progress bar.

You can found this article in Spanish here[1]

Puedes encontrar este artículo en español aquí[1]

INSTALATION

The first thing to do is to download the bootsplash patch. It can be found at the Con Kolivas's kernel patchs page[2] together with many more, but as of now we are only interested in bootsplash.

I won't explain here how to compile and patch a kernel, but I'll still give you a couple of tips:

Provided you have the kernel sources correctly installed in /usr/src/linux, copy the kernel patch bootsplash wherever you want (e.g. /usr/src) and decompress typing:

```
bunzip2 patch-1090_BS_0306020027_2.4.21-ck2.bz2
```

Then apply the patch from the /usr/src/linux directory:

```
patch -p1 <../1090_BS_0306020027_2.4.21-ck2
```

Once the patch has been applied, proceed to configure the kernel with any of the four methods avaliable. You should activate the following settings (verbatim copy from www.bootsplash.org[3])

- Go to "Console drivers" -> "Frame-Buffer support" and select "VESA VGA graphics console" (or possibly another framebuffer driver, if one is available for your graphics hardware) and select "Use splash screen instead of boot logo"
- Be sure to enable "Initial Ramdisk support" in "Block Devices"

Once done, all you have to do is to compile and install (or not yet) the new kernel.

Next step is to download a program named "splash", which will help us to create a initrd image which contains the necessary images and config files in order to boot properly with bootsplash, as obviously those can not be taken from hard disk at boot time.

Download it from ftp://ftp.suse.com/pub/people/stepan/bootsplash/rpm-sources/bootsplash/splashutils.tar.bz2[4]. Decompress wherever you want (why not /usr/src again ;-) like this:

```
tar xvfj splashutils.tar.bz2
```

This will create a new directory named splashutils. cd into it and type:

```
make
```

A warning during the progress compilation will be shown, but it can simply be ignored. At this point we will have four binaries named fbresolution, getkey, progress and splash. These must be copied to /usr/local/bin so they can be found in the path, though we are only interested in the progress and splash files.

Now it's time to choose a theme for our splashscreen. I would recommend Theme-Linux[5], as this will be the used one as the example for the rest of the article. Nevertheless you can choose any other that fits your taste. This theme has a progress bar and two modes:

- Verbose mode: All boot messages will be shown but with a translucid background image. Here you have a screenshot[6]
- Silent mode: No message is shown, just an image with a progress bar. Be here an image[7]

The best and easiest way to install a theme is to decompress in /etc/bootsplash/themes so we don't need to change anything in the config file.

At this point we have the following files in our system:

- */etc/bootsplash/themes/Linux/config/bootsplash-1024x768.cfg:* It contains the boot configuration. A detailed description of it can be found here[8]. We won't need to change anything in this file and the default values should be ok.
- */etc/bootsplash/themes/Linux/images/bootsplash-1024x768.jpg:* this is the jpg image to be used in verbose mode.
- */etc/bootsplash/themes/Linux/images/silent-1024x768.jpg:* this is the jpg image to be used in silent mode.

CONFIGURATION

Now it's time to build the initrd image that contains the config files and images showed on boot. Issue the following command:

```
splash -s -f /etc/bootsplash/themes/Linux/config/bootsplash-1024x768.cfg >> /boot/initrd.splash
```

Why bootsplash.cfg and not the images? Because this file will hold where the images will be loaded from.

Right now we are read to modify our favorite's boot manager config:

In Grub

```
kernel (hd0,3)/boot/vmlinuz-2.4.21-melenas root=/dev/hda4 vga=791 splash=silent
```

```
initrd (hd0,3)/boot/initrd.splash
```

In Lilo we add and/or modify the following lines:

```
vga=791
 initrd=/boot/initrd.splash
 append="splash=silent"
```

Let's see what's what:

- **vga=791:** It will show the boot splash at 1024x768 with 65000 colors (16 bits per pixel). The number will depend on the resolution and color depth of the theme we are using. Below is a brief table with all the available codes.

Colors	640x480	800x600	1024x768	1280x1024

256	769	771	773	775
32000	784	787	790	793
65000	785	788	791	794
16.7 Mill.	786	789	792	795

- **splash=silent:** Show boot splash in silent mode. If you prefer vebose mode, just delete this option. Verbose mode is the default option.
- **initrd=/boot/initrd.splash:** In this line the file initrd.splash (where the images are stored) is loaded

---

GETTING THE PROGRESS BAR TO WORK

If you followed all the instructions correctly, you now can restart and enjoy your new and fresh boot splash. But, what wrong?, there wasn't any progress bar! Documents found in Internet explain how to do it in SuSE, Mandrake, Gentoo and LFS, but not in Debian. I tried it with many examples and animations with mng, but none of them works. Eventually, I got to do it, but this method was not tested intensely as I have only tried it in my Debian/Sarge. It should work fine in Woody as well, but do it at your own risk.

First we need to know how to use the progress utility (remember it was compiled and copied together with splash). Progress just draws a quadrilateral in any framebuffer device (/dev/fb0 by default). If you run it without any argument you will get a list with all the available options. The only parameters we are interested in are:

- **x:** The quadrilateral origin for the x coordenate
- **y:** The quadrilateral origin for the y coordenate
- **dx:** Quadrilateral width
- **dy:** Quadrilateral height
- **color:** Quadrilateral color. This is a hexadecimal number. This colors work the same as for web pages, in case you are familiar with them.

For example:

```
progress 0 0 512 384 FFFFFF
```

will draw a white quadrilateral with the origin on the superior left corner of the screen, and it will cover a fourth of it.

```
progress 256 192 521 384 FCD192
```

will draw a yellow quadrilateral right in the middle of the screen.

This examples must be tried in a tty (preferely in tty1) and not in a Konsole or xterm window.

Ok, now we need to put the drawing program and the boot together. I studied Mandrake's boot and realized that its boot scripts use this program to draw a quadrilateral in a given position and with a given size every time a new service is started (xfs, cron, samba or any other it has installed). In Mandrake, this is achieved through the /etc/rc.d/rc script, but in Debian there's two: /etc/init.d/rcS is executed in first place and then /etc/init.d/rc (I won't explain how it works for every other distro as it's not the goal of this article).

Now it's time to take a look at the /etc/init.d/rcS code. Note that this script calls all the scripts on every runlevel in order, we insert our code **marked as bold** for drawing our progress bar. It should look like this:

```
#      Initialize j variable
#        j is the progress bar width
j=20

for i in /etc/rcS.d/S??*
do
        # Ignore dangling symlinks for now.
        [ ! -f "$i" ] && continue

#        Start progress bar
#
```

```
#       This is for not escape from established bar progress width

       <6f6)()(j
                then
#       With this, we call progress to draw a bar wider than
#       the previous one, exactly 15 pixels wider (j=dx)
#                                                 x   y  dx dy color

                /usr/local/bin/progress 204 667 $j 21 FCD123
                let j=j+15
        fi
#       End progress bar

        case "$i" in
                *.sh)
                        # Source shell script for speed.
                        (
                                trap - INT QUIT TSTP
                                set start
                                . $i
                        )
                        ;;
                *)
                        # No sh extension, so fork subprocess.
                        $i start
                        ;;
        esac
done
```

This code will draw the first part of the bar, but now we have to draw the second part with /etc/init.d/rc. In order to do this, we need to know exactly in what position did the drawing process stop so we can go on with the second smoothing.

```
# Explained below

danger_var=395
j=danger_var

   for i in /etc/rc$runlevel.d/S*
   do
        [ ! -f $i ] && continue

            if [ $previous != N ] && [ $previous != S ]
            then
                #
                # Find start script in previous runlevel and
                # stop script in this runlevel.
                #
                suffix=${i#/etc/rc$runlevel.d/S[0-9][0-9]}
                stop=/etc/rc$runlevel.d/K[0-9][0-9]$suffix
                previous_start=/etc/rc$previous.d/S[0-9][0-9]$suffix
                #
                # If there is a start script in the previous level
                # and _no_ stop script in this level, we don't
                # have to re-start the service.
                #
                [ -f $previous_start ] && [ ! -f $stop ] && continue


            fi

                #Continue progress bar
                #This is to not escape from established bar progress width
                    if ((j <616))
                    then

                    /usr/local/bin/progress 204 666 $j 21 FCD123
                    let j=j+15
                    fi
                #End progress bar

                 case "$runlevel" in
                        0|6)
                                startup $i stop
                                ;;
                        *)
                                startup $i start
                                ;;
                esac
done
```

The code is clear. Proceeding the same way, it draws the progress bar. But what is danger_var?. We'll go little by little. I said before that we need to know in what posicion did the first part of the bar drawing process stop so it could continue with the second smoothing. This posicion is signaled by danger_var. As I can't pass it like a local variable with export or environment variable, I had to get creative to do it. It's coarse and even dangerous if it isn't done right. The following code must be inserted at the end of /etc/init.d/rcS :

```
sed 's/danger_var=[0-9]*/danger_var='$j'/g' /etc/init.d/rc > /tmp/boots
mv /tmp/boots /etc/init.d/rc
chmod 755 /etc/init.d/rc
```

It couldn't be any easier: get j value, search the string *danger_var=any_number* in rc file and modify like this: *danger_var=last_value_of_j*. Save as /tmp/boots, rename then in /etc/init.d/rc and change its permissions. If you choose the name danger_var, you're not very likely to have name coincidence problems with any of the other boot scripts..

If you restart the computer now, you will be able to enjoy a beautiful progress bar, but you'll realize that when you halt or reboot the computer, the splash screen won't have a progress bar like it did before. This is solved with the following code (in bold), which must be added in /etc/init.d/rc

```
# First, run the KILL scripts.
 if [ $previous != N ]
 then

  k=20

  for i in /etc/rc$runlevel.d/K[0-9][0-9]*
  do
  # Check if the script is there.
  [ ! -f $i ] && continue

     # Stop the service.

        #This is to not escape from established bar progress width

          if ((k <616))
          then

          /usr/local/bin/progress 204 666 $k 21 FCD123
          let k=k+25
          fi
        # Fin barra proceso

        startup $i stop
   done
 fi
```

I don't need to explain what does the k variable do and why is it there, do I? ;-)

Now we can enjoy our graphical boot with progress bar and graphical halt.

EXTRAS AND FINAL NOTES

As an extra, I'll tell you that if you append this:

```
/usr/local/bin/splash -n -s -u 1 /etc/bootsplash/themes/Linux/config/bootsplash-1024x768.cfg
/usr/local/bin/splash -n -s -u 2 /etc/bootsplash/themes/Linux/config/bootsplash-1024x768.cfg
/usr/local/bin/splash -n -s -u 3 /etc/bootsplash/themes/Linux/config/bootsplash-1024x768.cfg
/usr/local/bin/splash -n -s -u 4 /etc/bootsplash/themes/Linux/config/bootsplash-1024x768.cfg
/usr/local/bin/splash -n -s -u 5 /etc/bootsplash/themes/Linux/config/bootsplash-1024x768.cfg
```

in the /etc/init.d/rcS file, you will realize that the different tty (from 2 to 6) have the same background image as tty1 in verbose mode. What's more, you can, after having downloaded other themes, have a different theme for every tty. You only need to modify the config files like in this example:

```
/usr/local/bin/splash -n -s -u 1 /etc/bootsplash/themes/OfficeDesktop/config/bootsplash-1024x768.cfg
```

which is a SuSE based theme with a box text inside.

Notes

1. Be careful when modifying /etc/init.d/rcS and /etc/init.d/rc. If you make a mistake, the machine might not boot properly. It's happened to me a few times, but I got around it booting from another Linux distro and then solving the errors. A Knoppix distro can be used for this purpose. I'll say it one last time: be careful with the modifications you make on both files.
2. Any upgrade in the sysinit package can happen to overwrite your rc and rcS modifications, so made a backup of your files.
3. If you switch to verbose mode with F2, you will notice the bar progress is still drawing in the screen. I still don't know how to solve this.

Recommended links

- Bootsplash official page[3]
- Bootsplash howto for SuSE[9]
- Very large howto in Gentoo[10]
- LFS bootsplash howto with good ideas extrapolateable to other distros[11]
- Con Koliva's kernel path page[2]
- Gorka Olaizola's howto with other scripts and information about animations[12]

Thanks to Javier Malonda[13] for helping me in this translation.

Lista de enlaces de este artículo:

1. http://bulma.net/body.phtml?nIdNoticia=1807
2. http://members.optusnet.com.au/ckolivas/kernel/
3. http://www.bootsplash.org
4. ftp://ftp.suse.com/pub/people/stepan/bootsplash/rpm-sources/bootsplash/splashuti
5. ftp://ftp.suse.com/pub/people/stepan/bootsplash/themes/Theme-Linux.tar.bz2
6. http://www.bootsplash.org/verbose-mode.jpg
7. http://www.bootsplash.org/silent-mode.jpg
8. http://www.bootsplash.org/config.html
9. http://sdb.suse.de/en/sdb/html/jkoeke_splashscreen.html
10. http://forums.gentoo.org/viewtopic.php?t=49036
11. http://hints.linuxfromscratch.org/hints/bootsplash.txt
12. http://wiki.escomposlinux.org/Escomposlinux/DoLiCaBootsplashEnDebian
13. http://comic.escomposlinux.org/

E-mail del autor: pjmelenas _ARROBA_ biwemail.com

Podrás encontrar este artículo e información adicional en: http://bulma.net/body.phtml?nIdNoticia=1812