Arch Linux BootSplash-HowTo

Thu Aug 7 08:57:49 2003

Abstract

This short guide explains to the interested user how to setup his system to display a high resolution boot splash image and use custom images as a background for the framebuffer consoles. Instructions how to patch the kernel are elaborated as necessary as well.

Contents

1 Preparations

1.1 What do I need?

To use the nifty boot splashscreen feature, you will need:

- a kernel patch for your kernel version
- a JPG image of an appropriate resolution to use as a splash screen (Duh!)
- the splash screen utilities
- a graphic card supporting the VESA framebuffer kernel extensions
- some time to kill, as usual

1.2 Where do I get all this?

Unfortunately the official bootsplash webpage at http://www.bootsplash.org does not (yet) offer a patch to the latest stable 2.4.21 kernel sources, which Arch Linux has been using for some time. Fear not, however, as you can download this patch and everything else you'll need from

http://archlinux.veloxis.de/howtos/bootsplash/downloads/.

There's even a package with some exemplary 1024x768 images you can use. If you want higher resolutions than this, I suggest you check out e1904's homepage at http://www.geocities.com/e1904/, which is in fact where I got these pictures from originally. Major kudos to e1904 not only for his (IMHO) very catchy pictures, but also for his help which made this HowTo possible in the first place! However, e1904 does not seem to offer a ready-made theme package for his higher resolutions, so I'd strongly suggest you try to get this stuff working with the 1024x768 package first, and customize later where you see fit. It's much easier that way.

2 Kernel Settings

2.1 How do I apply the patch to the kernel?

Since you are using Arch Linux' fabulous ABS, you can quite easily build your own kernel package to integrate seamlessly with the package management. Follow these simple steps and see below for detailed instructions:

1. Run *abs* as root to update the ABS tree

2. Copy */usr/abs/kernels/kernel* (or */kernel-scsi* if you need SCSI support) to */usr/abs/local/kernel* to have a good base to work with. You may of course re-use your existing custom kernel PKGBUILD

3. Download the kernel patch file and put it into the local kernel directory you just created, right next to all other included patches

4. Edit the PKGBUILD file to add the name of the bootsplash patch file to the "source" array and add a correct "patch" line right below the existing GCC patch. The PKGBUILD file should read like this:

```
[...]
backup=('boot/kconfig')
source=(ftp://ftp.kernel.org/pub/linux/kernel/v2.4/linux-$pkgver.tar.bz2 \
  linux-gcc3-$pkgver.patch bootsplash-3.0.7-$pkgver-vanilla-1.patch config

build() {
  rm -rf $startdir/pkg/*
  cd $startdir/src/linux-$pkgver
  patch -Np1 -i ../linux-gcc3-$pkgver.patch
  patch -Np1 -i ../bootsplash-3.0.7-$pkgver-vanilla-1.patch
  cp ../config ./.config
[...]
```

   You should also change the "pkgrel" value to "custom1" or something similar to prevent a mixup with the official kernel packages.

5. Now create a new kernel configuration file including the bootsplash option. Novice users may simply want to add these lines to the "config" file in the ABS kernel build directory:

```
CONFIG_EXPERIMENTAL=y
CONFIG_BLK_DEV_RAM=y
CONFIG_BLK_DEV_RAM_SIZE=4096
CONFIG_BLK_DEV_INITRD=y
```

```
CONFIG_BLK_DEV_LOOP=y
CONFIG_FB=y
CONFIG_FB_VESA=y
CONFIG_FBCON_SPLASHSCREEN=y
```

All but the last option is included in the stock kernel already, so unless you are working with a customized copy, only add the "FBCON" line somewhere to the file, the specific place does not matter.

Those folks who want to compile a totally custom kernel while they're at it should consider a different way, namely downloading the used kernel sources manually and untarring them somewhere in a temporary directory. Then patch these sources by copying the bootsplash patch into the kernel sources directory and running *patch -Np1 -i ./bootsplash** from within this dir. The patch should run flawlessly, so that you may then start *make menuconfig* to configure your kernel according to your wishes. First off you should load the configuration options from */boot/kconfig* and work from there to make your life easier. This file contains the configuration options used by the currently installed kernel package. If you know that you don't want to use these as a base, you probably know what you're doing anyway. Once you're ready to configure your kernel, make sure you activate the following options (choose "yes" not "module" where possible):

```
Code maturity / Prompt
Block devices / Loopback device support
Block devices / RAM disk support
Block devices / (4096) Default RAM disk size
Block devices / Initial RAM disk (initrd) support
Console / Frame-buffer / Support for frame buffer
Console / Frame-buffer / VESA VGA graphics console
Console / Frame-buffer / Use splash screen instead of boot logo
```

Again, the stock kernel has already everything enabled by default except for the last option. If the splash screen option is nowhere to be found, make sure that the code maturity option is activated to show you this experimental option! Also you should check whether the patch

has been applied correctly. Do not use any other framebuffer support but VESA, it will most likely not work! Keeping them as modules is okay, although you might want to remove them anyway to conserve space. Do not forget to save your new configuration into some file, and copy this new configuration into the ABS directory as the "config" file to use.

6. Now you should have edited your PKGBUILD to patch the sources, and modified the supplied "config" file in one way or another to include the boot splash option. All that's left to be done now is running "makepkg" and waiting until your kernel compilation completes.

7. Install the new kernel package and re-run LILO if need be.

2.2 Choosing a resolution

With frame buffer support compiled in, you just need to tell the kernel which screen resolution to use. As you might have guessed this is done with the help of a kernel boot parameter that should be added to your *lilo.conf* or your GRUB kernel line. This parameter is "vga=791" for a resolution of 1024x768x16. Please note that the bootsplash needs a color depth of 16bit, nothing else will work!

3 Userspace

3.1 Creating a theme

You can retrieve more in-depth information on the config file formats for the bootsplash themes on the official page at http://www.bootsplash.org. The explanations there are a little sparse, but at least complete. This document will only explain how the arch-theme does it's magic, how to go on with fancier stuff like progress meters and little animations is on the ToDo list for a new version of this HowTo.

That being said, you should first create the */etc/bootsplash/themes* directory and extract the arch-theme.tar.gz there, thus creating a new subdirectory called "arch" that contains the theme files used in this example.

Some background information to clear things up a bit; The "splash" utility included in the splashutils package (we will come to that later) expects a config file to parse. This config file must include such vital information like what picture(s) to use, where text output shall be placed and so on. A

'theme' therefore really is only a collection of config files for the splash utility and all pictures and other files that you may need.

You may now have a look at the configuration file for the boot splash screen which is located at *arch/config/arch-1024-boot.cfg*. This simple file does not contain many options; In fact it only makes clear that it's a version 3 config file, that the picture should actually be displayed, what color to use for the text and it's background, and in what area to write any text output. Not to forget it also sports a path to the actual file to display, one for the verbose and one for silent mode. Silent mode suppresses the display of boot messages in favor of an optional progress meter which involves some more setup with a wrapper script. That's why I left it disabled for now. ;)

Basing on this file, you could for example edit the referenced image to your liking and fiddle with the text color a bit. For now, leave everything alone and continue with the setup.

3.2 Installing the userspace tools

This example only makes use of the "splash" utility. To install it along with a few other helpful programs, download the binary package or the ABS PKGBUILD depending on your preference and build/install the splashutils package. You'll end up with the "splash" binary and it's friends getting installed to */usr/sbin*.

3.3 Installing a theme

Since the kernel does not have access to your harddrive at the early booting stage the splash screen is supposed to be shown, you need to make use of an initrd. Fortunately you don't need to create one yourself; This is where the splash utility comes in handy, as it creates an initrd automatically, based on the config file (=theme) you choose. All you need to do is running "splash -s -f /etc/bootsplash/themes/arch/config/arch-1024-boot.cfg ¿ /boot/initrd.splash" to create a new initrd. Add this initrd to your lilo.conf or GRUB configuration and don't forget to re-run LILO. Again.

At this point you may want to reboot to cackle with glee over your new way cool boot splash screen. If it doesn't work, check the dmesg output for any errors and fix them.

4 Backgrounds for virtual consoles

4.1 Configuration files

Another cool thing to do with the "splash" utility is setting the background of your virtual consoles to a custom picture. The arch-theme package includes half a dozen config files to do exactly that. These config files are basically identical copies of the boot config, but refer to different pictures. That's it, everything else is untouched. I chose to use a different picture for each virtual terminal to distinguish them easily.

4.2 Activating the backgrounds

Actually that's something like a crude hack, but it works just fine. To set the background image of a VC, one can use the -u option of the splash utility. Putting a handful of these commands into your */etc/rc.local* will set the background images right after bootup, and all is dandy. My rc.local got these entries:

```
# Set background images on consoles
/usr/sbin/splash -s -u 0 /etc/bootsplash/themes/arch/config/arch-1024-vt1.cfg
/usr/sbin/splash -s -u 1 /etc/bootsplash/themes/arch/config/arch-1024-vt2.cfg
/usr/sbin/splash -s -u 2 /etc/bootsplash/themes/arch/config/arch-1024-vt3.cfg
/usr/sbin/splash -s -u 3 /etc/bootsplash/themes/arch/config/arch-1024-vt4.cfg
/usr/sbin/splash -s -u 4 /etc/bootsplash/themes/arch/config/arch-1024-vt5.cfg
/usr/sbin/splash -s -u 5 /etc/bootsplash/themes/arch/config/arch-1024-vt6.cfg
```

You can of course add more VCs as you see fit, just modify the supplied images and config files, and add these to the rc.local. I am pretty sure you get the idea how it generally works.

5 Feedback

Feedback is of course appreciated, even if it's only a short "Works for me!" message. If you have the time and inclination to elaborate on setting up the advanced features of the bootsplash config like progress meters, boxes, animations and the whole wrapper issue, you're cordially invited to do so! Please send any feedback to <dennis@archlinux.org>.