Main

Home
Projects
About

Projects

TG Install uClibc chroot
Basic initramfs image
VAIO Brightness
Basic cryptsetup
Install CD -> USB
Gentoo stages
NiTi research project

Howto create an initramfs image

Initramfs is the new way of doing stuff before the root partition is mounted. This is useful if you need to do something special to get your root partition visible to the kernel. For example, if it is an lvm partition, encrypted or on USB.
You can also stay in the initramfs. This can be very useful for debugging, or if you just want a very fast booting system.

Creating the folder structure

Before we get to the fun bit, you'll need to create a few folders.
Alternatively you can use ***this*** tarball, which also contains the /bin/busybox binary and the /init script.
The empty mdev.conf is created so that newer versions of busybox would not complain.

```
$ mkdir -p work/initramfs/{bin,sbin,etc,proc,sys,newroot}
$ cd work
$ touch initramfs/etc/mdev.conf
```

Getting BusyBox

BusyBox combines most common UNIX utilities into a single binary. If it's compiled statically, it is usually the only binary file you will need in the initramfs image. Instead you could *ldd* the dynamicly linked BusyBox binary, and copy all the necessary libraries to /lib in the initramfs, but it is a lot easier to go with a static binary.
BusyBox can be compiled using glibc or uClibc. Both work just the same, but uClibc produces quite a lot smaller binaries.

```
$ wget http://jootamam.net/initramfs-files/busybox-1.10.1-static.bz2 -O -
| bunzip2 > initramfs/bin/busybox
$ chmod +x initramfs/bin/busybox
$ ln -s busybox initramfs/bin/sh
```

To use your own BusyBox, but make sure it's static. You can use *file /bin/{bb,busybox} | grep static* to check if your current system has a static version of busybox

/init

If an initramfs is built into the kernel or passed to it, the kernel will try to execute /init in the initramfs. Usually /init is a shell script that will find and mount the root partition,

then switch_root onto the root partiton and execute /sbin/init.

```
$ touch initramfs/init
$ chmod +x initramfs/init
```

Then edit *initramfs/init*

```
#!/bin/sh

#Mount things needed by this script
mount -t proc proc /proc
mount -t sysfs sysfs /sys

#Disable kernel messages from popping onto the screen
echo 0 > /proc/sys/kernel/printk

#Clear the screen
clear

#Create all the symlinks to /bin/busybox
busybox --install -s

#Create device nodes
mknod /dev/null c 1 3
mknod /dev/tty c 5 0
mdev -s

#Function for parsing command line options with "=" in them
# get_opt("init=/sbin/init") will return "/sbin/init"
get_opt() {
        echo "$@" | cut -d "=" -f 2
}

#Defaults
init="/sbin/init"
root="/dev/hda1"

#Process command line options
for i in $(cat /proc/cmdline); do
        case $i in
                root\=*)
                        root=$(get_opt $i)
                        ;;
                init\=*)
                        init=$(get_opt $i)
                        ;;
        esac
done

#Mount the root device
mount "${root}" /newroot

#Check if $init exists and is executable
```

```
if [[ -x "/newroot/${init}" ]] ; then
        #Unmount all other mounts so that the ram used by
        #the initramfs can be cleared after switch_root
        umount /sys /proc

        #Switch to the new root and execute init
        exec switch_root /newroot "${init}"
fi

#This will only be run if the exec above failed
echo "Failed to switch_root, dropping to a shell"
exec sh
```

This /init script doesnt actually do anything that you couldn't do without an initramfs, but it can very easily be extended.

Creating the .cpio and .igz

When passing an initramfs to the kernel using your bootloader (the initrd option in Grub), the kernel is expecting a gzipped cpio archive (.igz). When building the initramfs image into the kernel, one of the easiest ways is to create a cpio archive (.cpio).
We will be creating both files.

```
$ cd initramfs
$ find . | cpio -H newc -o > ../initramfs.cpio
$ cd ..
$ cat initramfs.cpio | gzip > initramfs.igz
```

You might want to put these commands in a shell script file so you can easily re-create the initramfs images when you make changes to the content of work/initramfs/

Kernel configuration

The only thing you will need in the kernel is CONFIG_BLK_DEV_INITRD=y
If you want to build your initramfs image into the kernel. This way you won't need to pass an initrd option to your bootloader, but when you update the initramfs image, you will also need to rebuild the kernel. If you do want to do this, give the path to the initramfs.cpio file in *Initramfs source file(s)*.

```
    General Setup --->
       [*] Initial RAM filesystem and RAM disk (initramfs/initrd) support
       ()    Initramfs source file(s)
```

Grub configuration

To pass the initramfs.igz to the kernel using Grub you will need the *initrd* option in menu.lst (grub.conf for Gentoo users).

```
title Gentoo
kernel /vmlinuz
initrd /initramfs.igz
```

As we have set default root partition and init in our *work/initramfs/init*, we do not need to specify them on the kernel line. You still can specify those options if you want to or

need to by appending *root=/dev/hda1 init=/sbin/init*.

Images by ***Mrs A. Linnapuomi***