How to simply recover a lost inittab file

How to recover a lost /etc/inittab file on Centos or RHEL?

If you’ve lost your */etc/inittab* in a RHEL or Centos environement, recovering it is quite easy:

1. Boot to single user mode (eg: add "single" at the end of your grub options)
2. Mount your cdrom or iso to /mnt (*mount -r /dev/cdrom /mnt*)
3. Find which package provides inittab (*yum -q --whatprovides /etc/inittab*)
4. Go to /mnt and find your rpm package which is actually "initscripts" (*find /mnt -name '\*initscripts\*'*)
5. Reinstall your the corresponding package which, for centos 6.5 is located in the *Packages* subdirectory if the cd. (*rpm -Uv --noscripts --force /mnt/Packages/initscripts-\*.rpm*)
6. **Reinstall a corrupted or destroyed boot partition.**
7. Lately I have had a couple of systems where the boot partition got corrupted or it's contents  
   deleted mistakenly. Even though I could not boot the system, I did not want to reinstall the whole  
   system and recover the data from backups.  
      
   So here is a quick solution that worked both on a SUSE 9.3 and a CentOS 5.4 system.
8. **1.** Boot the computer into rescue mode with the relevant CD/DVD by typing "linux rescue" at the boot prompt.
9. **2.** Mount the / partition of the system. On CentOS this is automatically done and the / is mounted on /mnt/sysimage.
10. Otherwise just do (assuming the root partition is on /dev/sda2)

|  |
| --- |
| mount /dev/sda2 /mnt/sysimage |

1. **3.** Chroot into the old root partition since that is where we actually want to make all the changes

|  |
| --- |
| chroot /mnt/sysimage |

1. **4.** If the filesystem labels got deleted up for some reason or they don't agree with what is described in /etc/fstab we should fix that (Assuming the boot partition is /dev/sda1):

|  |
| --- |
| e2label /dev/sda1 /boot |

1. **5.** Make sure the boot partition is mounted:

|  |
| --- |
| mount /dev/sda1 /boot |

1. **6.** Mount the CD-ROM we used to boot the machine

|  |
| --- |
| mkdir /media/cdrom mount -o loop,ro /dev/hdc /media/cdrom |

1. **7.** Now we need to install the relevant rpms. Make sure you pick the correct kernel for your configuration system (PAE etc..):

|  |
| --- |
| cd /media/cdrom/CentOS *(this will vary depending on the distro)* rpm -ivh --force kernel-2.6.18-164.el5.i686.rpm rpm -Uvh --force grub-0.97-13.5.i386.rpm rpm -Uvh --force redhat-logos-4.9.99-11.el5.centos.noarch.rpm |

1. **8.** Install GRUB:

|  |
| --- |
| grub-install /dev/sda1 *(or whichever partition boot is mounted on)* |

1. **9.** If /boot/grub/grub.conf is also lost, you need to create it manually.

|  |
| --- |
| vim /boot/grub/grub.conf  default=0 timeout=5 splashimage=(hd0,0)/grub/splash.xpm.gz hiddenmenu title CentOS         root (hd0,0)         kernel /vmlinuz-6.18-164.el5 ro root=/dev/VolGroup00/LogVol00         initrd /initrd-2.6.18-164.el5.img |

1. 10. Lastly make a soft link to grub.conf:

|  |
| --- |
| cd /boot/grub ln -s grub.conf menu.lst |

1. Reboot the system and if everything goes well it should boot normally. Run immediately a yum upgrade incase you had to install an older version of the kernel from the CD-ROM.

### [Missing GRUB Config in Linux](http://troysunix.blogspot.in/2011/01/missing-grub-config-in-linux.html)

Having recently written up how to [restore GRUB to a missing / corrupted  
master boot record (MBR)](http://troysunix.blogspot.com/2011/01/grub-corrupted-mbr-and-linux.html), it seemed appropriate to follow up with  
resolving a missing GRUB config file.  Our host details for this  
situation are:

        HOST:           tux

        PROMPTS:        [grub> |tux [0] ]

        OS:             CentOS 5.4 Linux

        DISKS:          [sda (hd0|disk1)|sdb (hd1|disk2)]

        ROOT PARTITION: sdb1

While the host has two viable disks, we'll only focus on disk 2.  Also,  
the following details a Red Hat variant of Linux, though may additionally  
be usable with other distros or other UNIX variants using GRUB.  
  
After powering on the host, we tell the BIOS to boot disk 2.  Once through  
POST, the BIOS runs the boot code from the MBR of disk 2, only to display  
the following:

            GNU GRUB  version 0.97  (639K lower / 1047488K upper memory)

         [ Minimal BASH-like line editing is supported.  For the first word, TAB

           lists possible command completions.  Anywhere else TAB lists the possible

           completions of a device/filename.]

        grub> \_

Well, GRUB is installed on the MBR, otherwise we wouldn't have seen  
the above.  The reason that we see it is likely due to a missing GRUB  
configuration file (grub.conf or menu.lst).  Proceeding from that  
assumption, we use 'find' to locate the "stage1" boot file on the  
available disks, subsequently setting the root disk via 'root':

        grub> find /boot/grub/stage1

         (hd0,0)

         (hd1,0)

        grub> root (hd1,0)

         Filesystem type is ext2fs, partition type 0x83

The 'find' returns two disks.  As previously stated, we want the  
correlative to our root disk on "sdb1" so we set 'root' to "hd1,0".  
(We are actually setting the device that contains our /boot directory.)  
The value breaks down to hd1 = disk 2, hd1,0 = disk2, first partition.  
Now we need to tell grub where our kernel is.  In the first 'kernel'  
example, I've only typed out the beginning of the kernel file and hit  
[TAB] for autocompletion, resulting in the second 'kernel' command.  
The second kernel command has additional options of "ro" to tell the  
kernel to initially mount the root FS read only, while "root=/dev/sdb1"  
identifies our root partition.  After 'kernel', we specify an initial  
ramdisk to use via 'initrd' ([TAB] autocompletion can be used here  
as well):

        grub> kernel /boot/vml[TAB]

        grub> kernel /boot/vmlinuz-2.6.18-164.el5 ro root=/dev/sdb1

           [Linux-bzImage, setup=0x1e00, size=0x1c31b4]

        grub> initrd /boot/initrd-2.6.18-164.el5.img

           [Linux-initrd @ 0x37d73000, 0x27c402 bytes]

        grub>

As an aside, if either the kernel or ramdisk cannot be found, grub will  
return the following:

        grub> kernel /boot/trash[TAB]

        Error 15: File not found

        grub> kernel /boot/trashfile

        Error 15: File not found

In this case, if the file is your kernel and you are typing out the  
correct path, you will need to either re-install your kernel or recover it  
from backups.  The same applies for the ramdisk, though you may also have  
the additional option of using 'mkinitrd' from an alternate boot disk.  
After setting the root disk, the kernel file, and the ramdisk, we can  
tell GRUB to boot the system which will bring us to our login prompt:

        grub> boot

        # system boots up

        CentOS release 5.4 (Final

        Kernel 2.6.18-164.el5 on an i686

        tux login: \_

With the system up, we verify that neither grub.conf nor menu.lst exist,  
check if there is a volume label on our root disk, and verify the file  
names of our ramdisk and kernel:

        tux [0] /bin/ls -l /boot/grub/grub.conf /boot/grub/menu.lst

        /bin/ls: /boot/grub/grub.conf: No such file or directory

        /bin/ls: /boot/grub/menu.lst: No such file or directory

        tux [2] /sbin/tune2fs -l /dev/sdb1 | /bin/grep name:

        Filesystem volume name:   /2

        tux [0] /bin/ls /boot | /bin/egrep 'vmlinuz|initrd'

        initrd-2.6.18-164.el5.img

        vmlinuz-2.6.18-164.el5

        tux [0]

At this point, we need to create a new configuration file to boot the  
host without futher intervention.  Using your favorite text editor  
(vi?), create "grub.conf" with the following, or similar, tailoring it  
to your needs:

        tux [0] /bin/cat /boot/grub/grub.conf

        default=0

        timeout=5

        title CentOS

                root (hd1,0)

                kernel /boot/vmlinuz-2.6.18-164.el5 ro root=LABEL=/2

                initrd /boot/initrd-2.6.18-164.el5.img

Of note, rather than specifying the exact disk for root as we did from  
the GRUB prompt, we've configured "grub.conf" with root using the volume  
label (/2) that was returned from 'tune2fs -l'.  Since we are using  
CentOS (a Red Hat clone), "menu.lst" is a symlink back to "grub.conf".  
Use 'ln -s' to recreate the symlink and verify it:

        tux [0] /bin/ln -s /boot/grub/grub.conf /boot/grub/menu.lst

        tux [0] /bin/ls -l /boot/grub/grub.conf /boot/grub/menu.lst

        -rw-r--r-- 1 root root 141 Jan 17 17:58 /boot/grub/grub.conf

        lrwxrwxrwx 1 root root  20 Jan 17 17:59 /boot/grub/menu.lst -> /boot/grub/grub.conf

With our work complete, the only thing left to do is reboot and verify  
our work.  Aside from telling the BIOS to boot disk 2, no further  
interaction occurs:

        tux [0] /sbin/reboot

    Output after telling the BIOS to use disk 2:

         GNU GRUB  version 0.97  (639K lower / 1047488K upper memory)

        CentOS

           Use the <up> and <down> keys to select which entry is highlighed.

           Press enter to boot the selected OS, 'e' to edit the

           commands before booting, 'a' to modify the kernel arguements

           before booting, or 'c' for a command-line.

        The highlighted entry will be booted automatically in 5 seconds.

    After timeout, the "CentOS" option is automatically booted:

          Booting 'CentOS'

        root (hd1,0)

         Filesystem type is ext2fs, partition type 0x83

        kernel /boot/vmlinuz-2.6.18-164.el5 ro root=LABEL=/2

           [Linux-bzImage, setup=0x1e00, size=0x1c31b4]

        initrd /boot/initrd-2.6.18-164.el5.img

           [Linux-initrd @ 0x37d73000, 0x27c402 bytes]

        <snip...>

        CentOS release 5.4 (Final

        Kernel 2.6.18-164.el5 on an i686

        tux login: \_

### [GRUB, a Corrupted MBR, and Linux](http://troysunix.blogspot.in/2011/01/grub-corrupted-mbr-and-linux.html)

Recently, after [cloning a root disk in Linux](http://troysunix.blogspot.com/2011/01/disk-cloning-in-linux.html), I ran into an issue because  
I failed to setup the master boot record (MBR) on the alternate disk.  
Everything else was configured, including the boot image files and grub  
configuration under /boot/grub, but the MBR wasn't setup.  The following  
details one solution to this using the following details:

        HOST:           tux

        PROMPTS:        [boot: |sh-3.2# |grub> |tux [0] ]

        OS:             CentOS 5.4 Linux

        DISKS:          [sda (hd0|disk 1)|sdb (hd1|disk 2)]

        MEDIA:          disk 1 of Linux install CDs / DVD

To start, after cloning the root disk, I attempted to boot the alternate  
disk (disk 2) from the BIOS, only to see the following:

        FATAL: No bootable medium found! System halted.

As one of my colleagues kindly pointed out to me recently, I like working  
with "islands", as in removing potentially easy options and making things  
seemingly more difficult.  (This is because easy options aren't always  
available.)  So, rather than booting back to the primary disk to resolve  
this, we'll proceed assuming that the primary disk isn't usable for our  
recovery purposes.  Reset the machine and boot from your Linux install  
media, disk 1.  You should see something similar to that below, so type  
'linux rescue' at 'boot:':

        -  To install or upgrade in graphical mode, press the <ENTER> key.

        -  To install or upgrade in text mod, type: linux text <ENTER>.

        -  Use the function keys listed below for more information.

        [F1-Main] [F2-Options] [F3-General] [F4-Kernel] [F5-Rescue]

        boot: linux rescue

In the following screens select Language option, keyboard type, network  
startup [Yes | No] (select No), whether to mount root rw or ro [Continue  
| Read-Only | Skip] (select Skip), etc.  Once you get to a shell, enter  
into 'grub':

        sh-3.2# grub

After the screen clears / refreshes, you should see something akin to  
the following:

            GNU GRUB  version 0.97  (640K lower / 3072K upper memory)

         [ Minimal BASH-like line editing is supported.  For the first work, TAB

           lists possible command completions.  Anywhere else TAB lists the possible

           completions of a device/filename.]

        grub>

While I know that my /boot directory is on the root partition at sdb1,  
we can let grub search it out and display possible candidates:

        grub> find /boot/grub/stage1

         (hd0,0)

         (hd1,0)

The naming convention above of hd0,0 and hd1,0 references sda1 and sdb1,  
respectively.  (hd0 => first disk, hd0,0 => first disk, first partition;  
hd1 = second disk, hd1,0 = second disk, first partition; etc.)  Of the  
next two commands to grub, 'root' sets the current root device to that  
selected (hd1,0 (sdb1)), and 'setup' installs 'grub' to the MBR on the  
disk (hd1 (sdb)).

        grub> root (hd1,0)

         Filesystem type is ext2fs, partition type 0x83

        grub> setup (hd1)

         Checking if "/boot/grub/stage1" exists... yes

         Checking if "/boot/grub/stage2" exists... yes

         Checking if "/boot/grub/e2fs\_stage1\_5" exists... yes

         Running "embed /boot/grub/e2fs\_stage1\_5 (hd1)"... 15 sectores are embedded.

        succeeded

         Running "install /boot/grub/stage1 (hd1) (hd1)1+15 p (hd1,0)/boot/grub/stage2

        /boot/grub/grub.conf"... succeeded

        Done.

        grub> quit

During 'setup' execution, we see that various files on the disk are  
checked in order to proceed.  If any of these files don't exist, 'setup'  
will fail.  After 'setup' completes, we've finished setting up the MBR,  
exited out of 'grub', and rebooted the host as seen below (don't forget  
to remove the CD / DVD):

        sh-3.2# reboot

## Updating Your Kernel To Kernel 3.3.1

Yo Yo Boyz. Today I have come with a really nice tutorial for you guys. Here we will upgrade our kernel to the latest stable version of Linux kernel. I am using RHEL6 with kernel 2.6 and now I have upgraded it to 3.3.1. So it’s time to tell you how to do this on your machine. So before we begin our recipe, download kernel 3.3.1 source package from [**http://boot.kernel.org/**](http://boot.kernel.org/). Make sure you have enough hard disk space on your system as on my system the final size of the package after compilation has become 3.5 GB. Now it’s showtime so get ready and do as I say.

* Swith to /usr/src directory using **cd /usr/src** command.
* Copy the downloaded linux-3.3.1.tar.bz2 package to this directory.
* Now extract the contents of the package using   **tar xvjf linux-3.3.1.tar.bz2**command**.** Now you’ll get a directory named linux-3.3.1.
* Switched to that directory using **cd linux-3.3.1** command.
* Now make sure that kernel-headers kernel-devel and gcc packages are installed on Linux.In case they are not installed install them using **yum install kernel-headers kernel-devel gcc** command**.**
* Now run**cp /usr/src/kernels/2.6.32-71.el6.i686/.config .**command.This will copy the contents to the current directory.Don’t forget the . at the end of the command.
* Now it’s time to run**make oldconfig**command**.**Here you’ll  be asked a lot of yes no type questions. I just kept on pressing enter key except one place where you’ll be asked the processor type. So select the appropriate processor you are using.For rest of the questions you can simply press enter to accept the defaults like I did**.**
* Now issue**make dep**

**make clean**

**make bzImage**

This will take a long time. So you can have a walk or take a cup of coffee.

* Now issue**cp arch/x86/boot/bzImage /boot/vmlinuz3**

**chmod a+x /boot/vmlinuz3**

**cp System.map /boot/System.map3**

**make modules.**

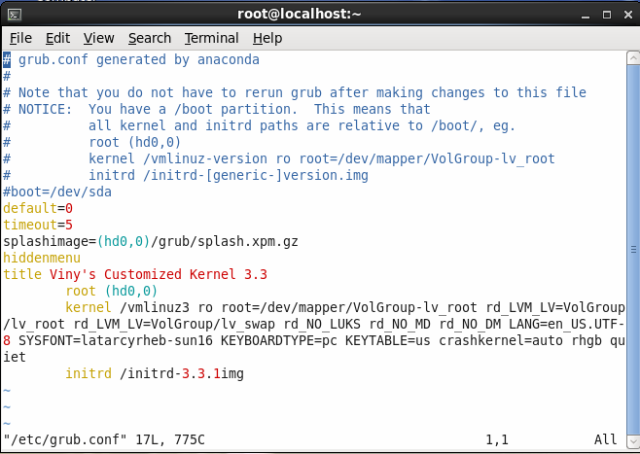
Again it will take some time. So another cup of coffee can be a good choice to cut the time**.**

**make modules\_install**

**mkinitrd /boot/initrd-3.3.1.img 3.3.1**

Please note that there is a space between initrd-3.3.1.img and 3.3.1. In case there is a mismatch in the name it will show an error and you can check /lib/modules for appropriate name.

* Now it’s time to update grub.So open**/boot/grub/grub.conf**and change the values of kernel and initrd parameters as **kernel /vmlinuz3**and **initrd /initrd-3.3.1img**as shown in the figure. Now save this file and reboot your machine. Now you can verify that your kernel has been upgraded successfully by issuing uname -r command.If it is showing 3.3.1 then the task is completed successfully. Time taking task but all is well that ends well.

**[](https://hackstips.files.wordpress.com/2012/04/boot.png)**