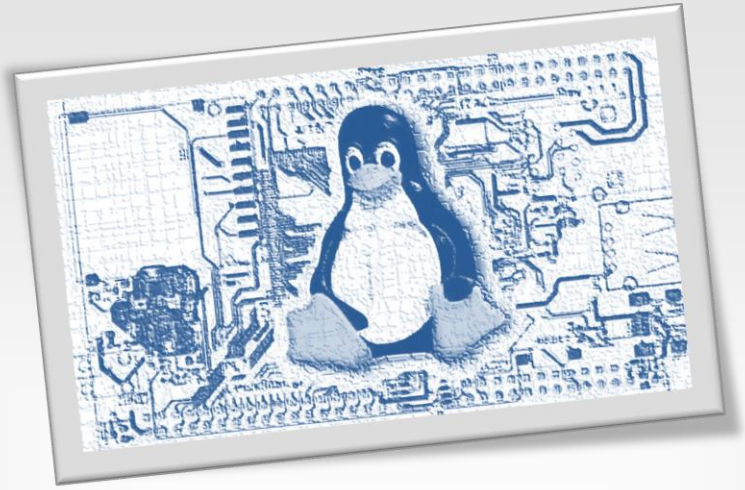
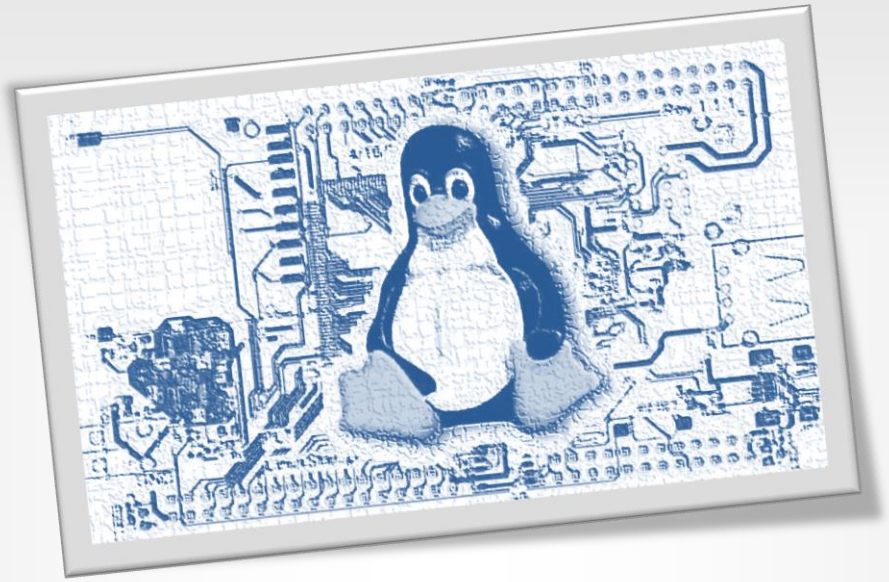


Embedded Linux
MiniOS for X86

Information Technology Institute (ITI)

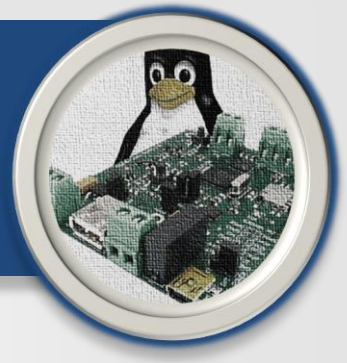
Sherif Mousa

The expert in anything was once … a beginner



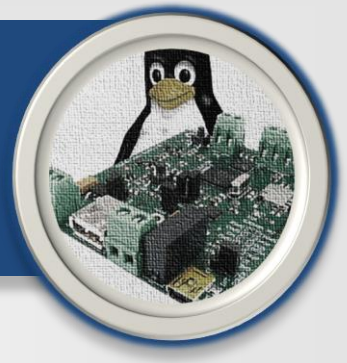
Introduction

Introduction



- This tutorial will show you how to create your own "Minimal Linux Distribution" on a USB Flash memory.
- You will learn about:
  - The FHS (Filesystem Hierarchy Standard).
  - Linux minimal common commands that make the system start and operate well
  - How to deal with Grub bootloader (install, edit conf)
  - How to use the Linux kernel
  - Dealing with some root commands (mount, mkfs.ext2, grub-install)

Prerequisites



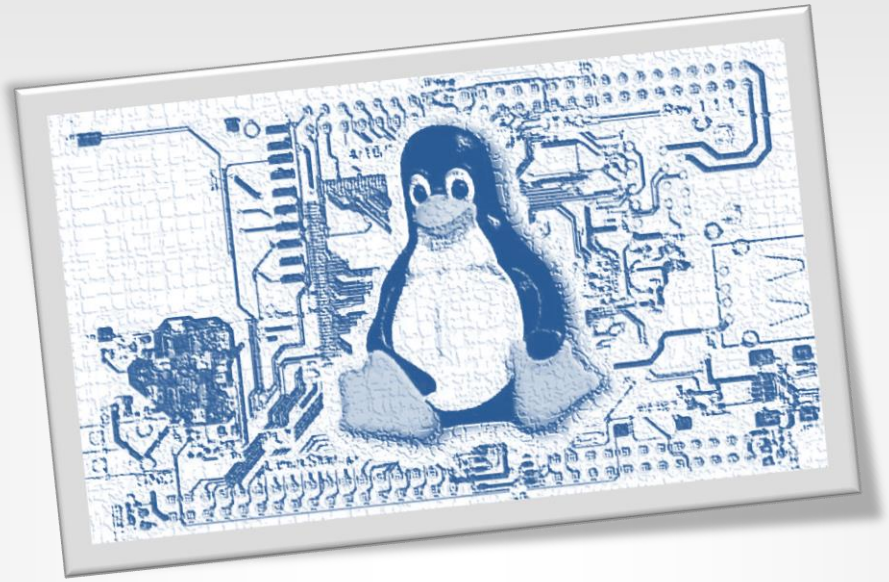
- You will need to install these packages first on Ubuntu

```
sudo apt-get update
sudo apt-get install ncurses-dev bison texinfo
flex vim
```

- You should be good in working with at least one Linux distribution like UBUNTU, and can understand the command you write and its options.

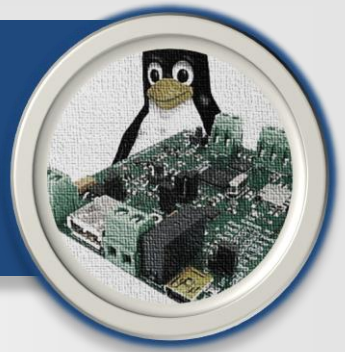
Basic Linux Structure

- Boot Loader to start the system from its storage media.
- Linux Kernel image with or without initram filesystem (initrd.img).
- Root Filesystem (main directories and sub directories).
- Configuration files and startup scripts in the (/etc) directory.
- Essential Linux commands needed for system to operate well.
- C & GCC Shared Libraries needed for any C or C++ application to run.
- GUI system or not.
- Extra Programs.



Root FileSystem /

Creating Root Filesystem /

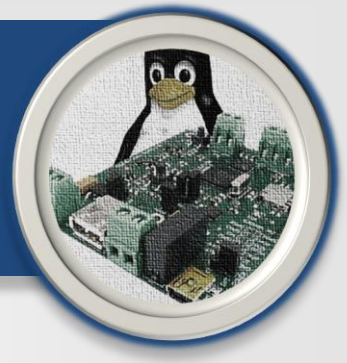


- Open a new Terminal, you will be right now inside your home directory. Assume that the user name currently running is (shatrix), you will be in: **/home/shatrix**
- Create any directory to contain the whole OS files and enter it

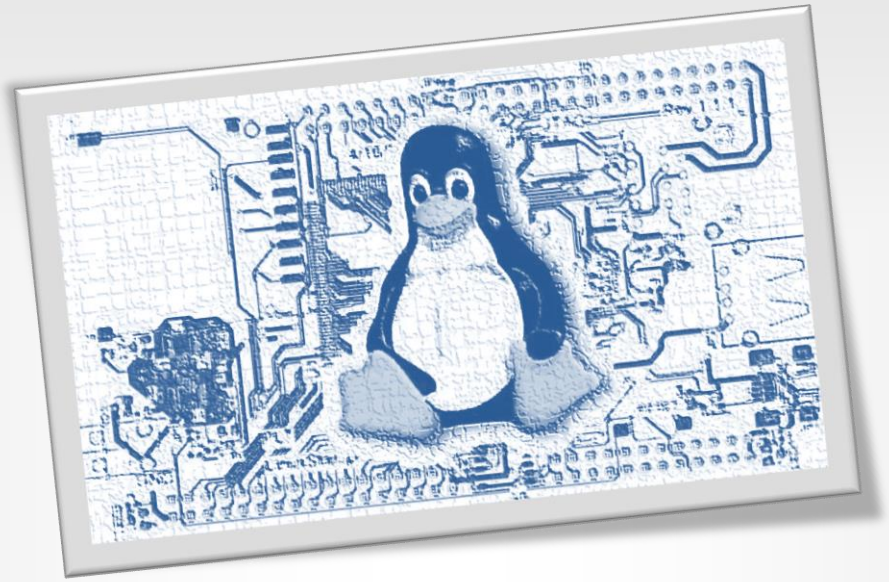
```
mkdir minios
cd minios
mkdir tools system
cd system
```

- Now you are in:
  - **/home/shatrix/minios/system**

Creating Root Filesystem /



- Execute these commands:
  - `mkdir bin boot dev etc lib proc root sbin media sys tmp var usr`
  - `mkdir usr/bin usr/sbin usr/lib`
  - `chmod 1777 tmp`
- Now you have the main filesystem directories to contain the whole system.



Configuration Files

Conf Files & Startup Scripts

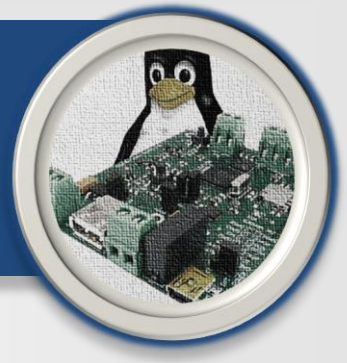
- Inside (/etc), create these files & fill them with these contents:
- **gedit group**
  - copy and paste this line inside it:

    ```
    root:x:0:
    ```
- **gedit passwd**
  - copy and paste this line inside it:

    ```
    root:x:0:0:root:/root:/bin/sh
    ```
- **gedit shadow**
  - copy and paste this line inside it:

    ```
    root::10:0:0:0:::
    ```

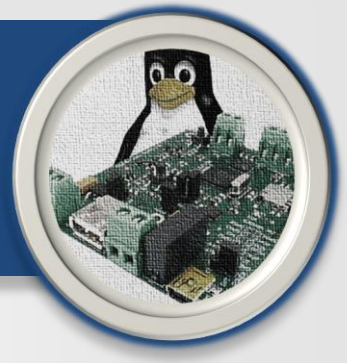
Conf Files & Startup Scripts



- **gedit inittab**

```
null::sysinit:/bin/mount -t proc proc /proc
null::sysinit:/bin/mkdir -p /dev/pts
null::sysinit:/bin/mount -a
null::sysinit:/sbin/ifconfig lo 127.0.0.1 up
::sysinit:/etc/rc
tty1::respawn:/sbin/getty 38400 tty1
null::shutdown:/bin/umount -a -r
```

Conf Files & Startup Scripts

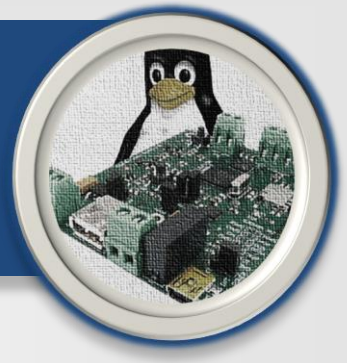


- **gedit rc**

```
#!/bin/sh
PATH=/sbin:/bin:/usr/sbin:/usr/bin
export PATH
/sbin/hwclock -s
/bin/hostname EmbeddedLinuxMiniOS
/bin/echo " "
/bin/echo " Y A T T A "
/bin/echo " My First Embedded Linux "
/bin/echo " "
```

- **chmod +x rc**

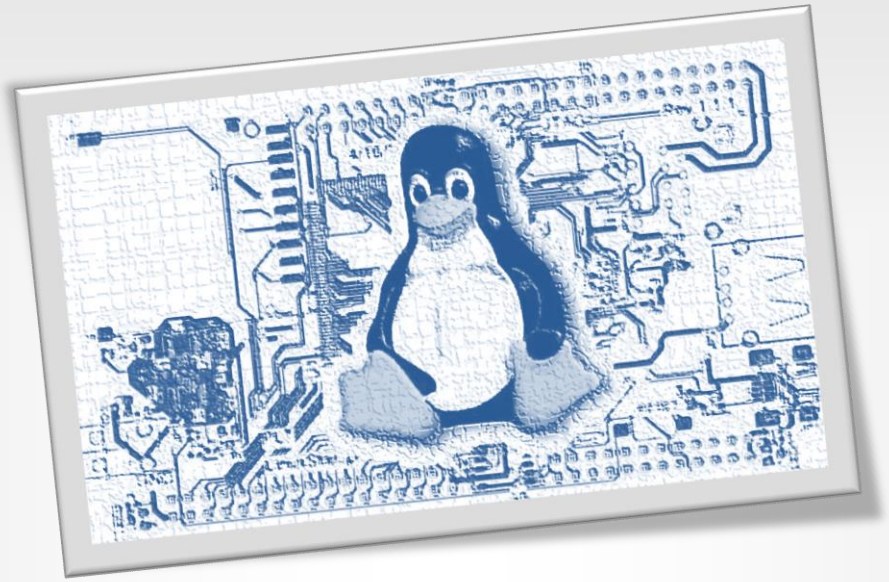
Conf Files & Startup Scripts



- **gedit profile**

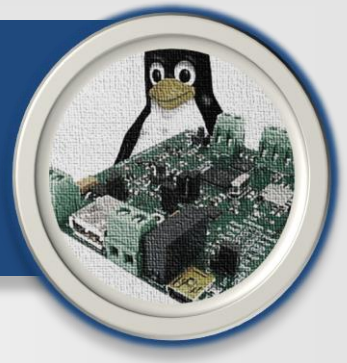
```
export PS1='EmbeddedLinuxMiniOS #'
export USER=`id -un`
export HISTSIZE=1000
echo "ya welcome ya welcome ^__^"
echo " "
```

- After Finishing back to the main dir:
  - **/home/shatrix/minios/system**



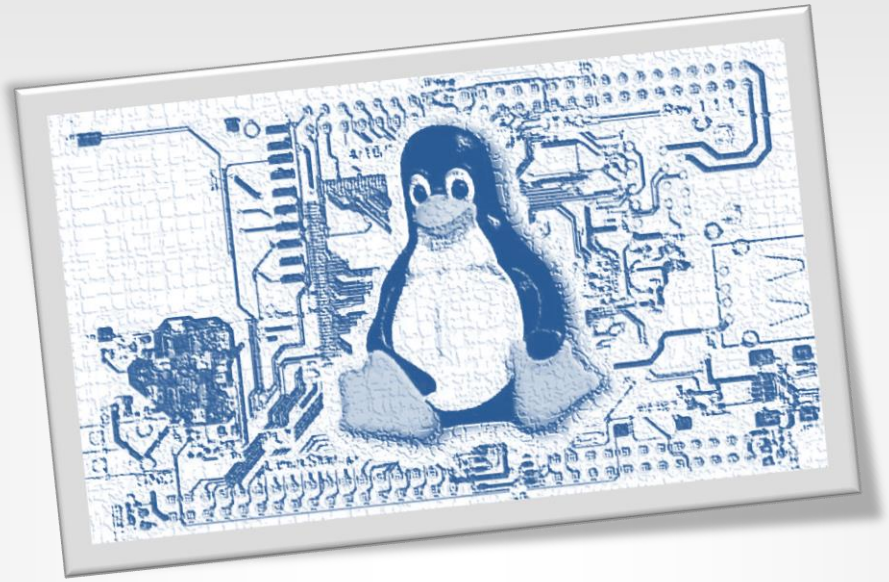
Linux Kernel

Linux Kernel



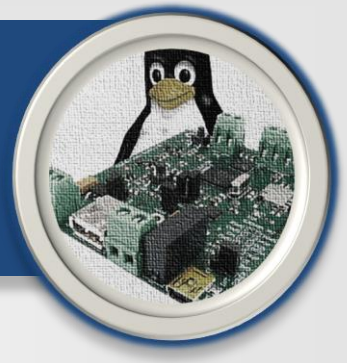
- To make your own Linux kernel is not easy like 1 2 3…
- Now, we can just use the Linux kernel already exists in our Ubuntu distribution.
- So, copy the kernel & initrd files from /boot:

```
cp /boot/vmlinuz-3.2.0-57-generic-pae boot/
cp /boot/initrd.img-3.2.0-57-generic-pae boot/
```



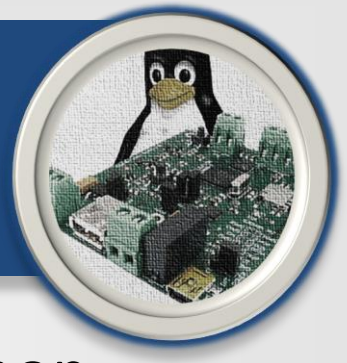
Linux Basic Commands

Linux Commands



- Any Linux distribution need some basic commands to operate correctly, like (mount, ls, pwd, ifconfig, cd, mkdir, touch, file, rm, vi) and more. We need these commands, so there are two ways to get them:
  - Getting the source code for each command needed, compile it separately and copy it to any bin directory inside the distribution.
  - Using BusyBox.

Linux Commands (BusyBox)



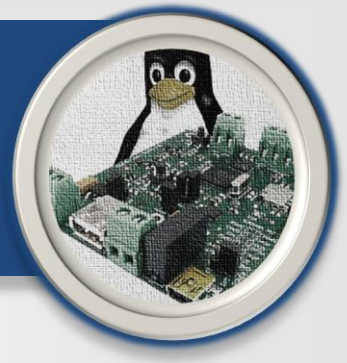
- BusyBox combines tiny versions of many common UNIX utilities into a single small executable. It provides replacements for most of the utilities you usually find in GNU fileutils, shellutils, etc. The utilities in BusyBox generally have fewer options than their full-featured GNU cousins; however, the options that are included provide the expected functionality and behave very much like their GNU counterparts. BusyBox provides a fairly complete environment for any small or embedded system.
- Get Busybox:

https://launchpad.net/ubuntu/precise/+source/busybox/1:1.18.5-1ubuntu4.1/+files/busybox_1.18.5.orig.tar.bz2

BusyBox

- Put the tar file in your **/home/shatrix/minios/tools** dir and open a new Terminal and go to this dir:
  - **`tar xvf busybox_1.18.5.orig.tar.bz2`**
  - **`cd busybox_1.18.5/`**
- Configure Busybox as required:
  - **`make defconfig`**
  - **`make menuconfig`**
- The command (make menuconfig) will open a GUI terminal-based configuration for the BusyBox program, you can choose what to be installed and what not to be installed, or you can just choose to exit this configuration and remember to save your configuration settings on exit.

BusyBox Configuration



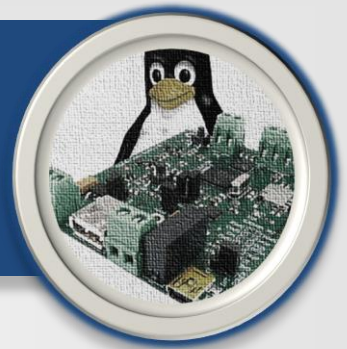
- menuconfig

```
BusyBox 1.17.2 Configuration
                    Busybox Configuration
  Arrow keys navigate the menu.  <Enter> selects submenus --->.
  Highlighted letters are hotkeys.  Pressing <Y> includes, <N> excludes,
  <M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help, </>
  for Search.  Legend: [*] built-in  [ ] excluded  <M> module  < >

     Busybox Settings  --->
  --- Applets
      Archival Utilities  --->
      Coreutils  --->
      Console Utilities  --->
      Debian Utilities  --->
      Editors  --->
      Finding Utilities  --->
      Init Utilities  --->
      Login/Password Management Utilities  --->
  v(+)

                <Select>    < Exit >    < Help >
```

BusyBox Build



- Then execute this command to compile and install BusyBox inside our system:

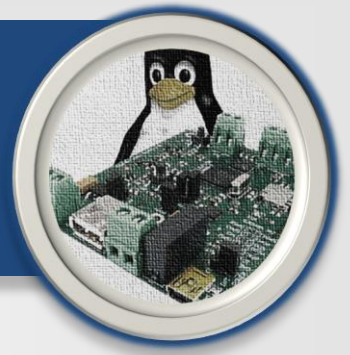
```
make CONFIG_PREFIX=/home/shatrix/minios/system install
```

- After finishing, close this Terminal and get back to the main Terminal and execute these commands

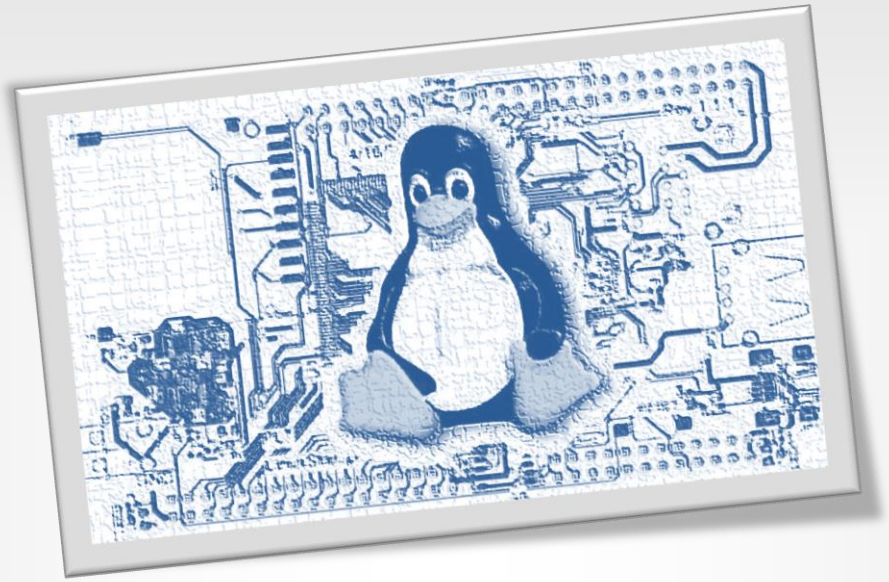
```
ls bin
ls sbin
```

- You will see that the commands have been installed inside the binary directories of our system. Also you need to know that all these commands are linked to one executable file named busybox inside /bin and all these commands shown are just symbolic links to that command.

C & GCC Libraries

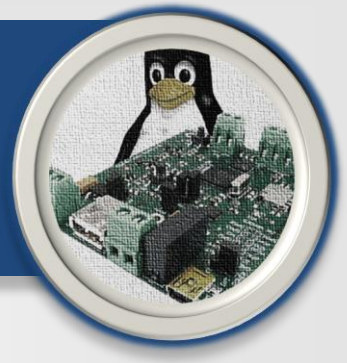


```
cp -d /lib/i386-linux-gnu/ld-* /lib/i386-
linux-gnu/libc-2* /lib/i386-linux-
gnu/libcrypt* /lib/i386-linux-gnu/libc.so*
/lib/i386-linux-gnu/libdl* /lib/i386-linux-
gnu/libgcc_s* /lib/i386-linux-gnu/libm.so*
/lib/i386-linux-gnu/libm-2* /lib/i386-linux-
gnu/libpthread* /lib/i386-linux-
gnu/libresolv* /lib/i386-linux-gnu/librt*
/lib/i386-linux-gnu/libutil* .
```



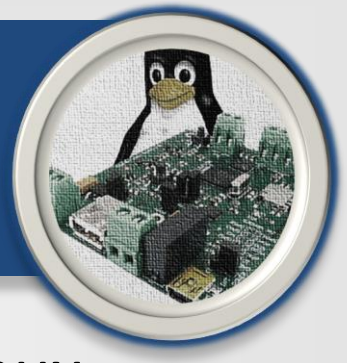
Boot Loader

Boot Loader



- Till now this system can run on any x86 cpu, and it can be copied to any storage (Flash memory, partition, CD) to be a stand alone operating system.
- Now, we just need a bootloader to start the system.
- We can use the bootloader already installed on UBUNTU (grub).

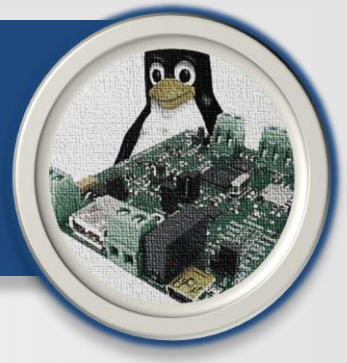
Boot Loader (Grub)



- We will use the grub already installed in our Linux distribution, we just need to write the grub configuration file manually.
- **grub.cfg:**
  - This is a text file contains the menu entries or the list of operating systems that grub should list them when the computer starts. We will create it manually.
  - Go to the PATH of our system
  - Create (boot/grub) dir
  - Create (boot/grub/grub.cfg) file

```
cd  /home/shatrix/minios/system
mkdir boot/grub
gedit boot/grub/grub.cfg
```

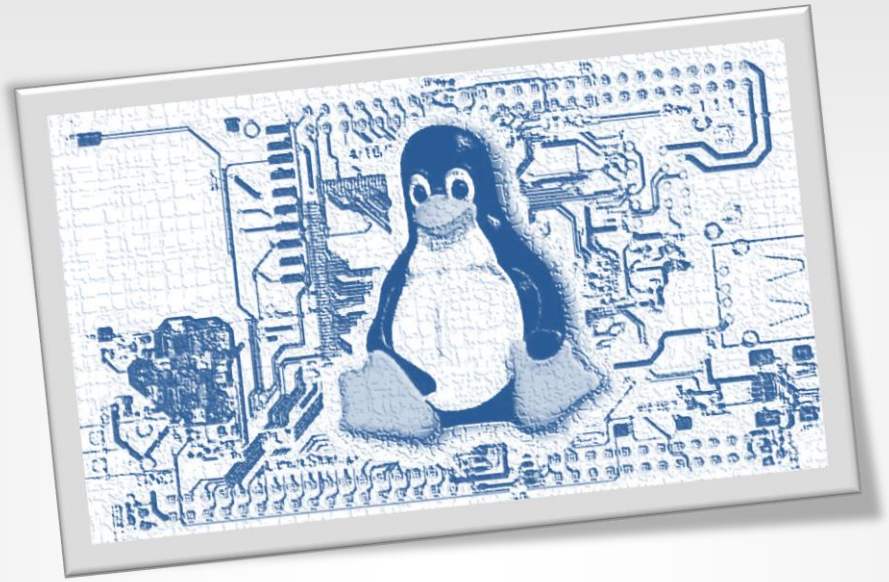
Boot Loader (Grub)



- copy and paste these lines in the grub.cfg file

```
set menu_color_normal=yellow/red
set menu_color_highlight=black/light-gray
menuentry 'LINUX MiniOS FOR X86' {
set root=(hd0,1)
linux /boot/vmlinuz-3.2.0-57-generic-pae
root=UUID=<disk-uuid> rw
initrd /boot/initrd.img-3.2.0-57-generic-pae
}
```

- The red text <disk-uuid> will be replaced with a value after that, don’t forget.

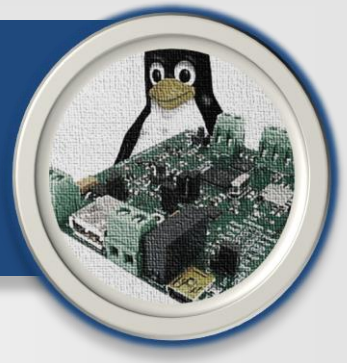


Install The Final MiniOS

Putting System on Flash

- Plug the USB Flash in, It will take a name like that sdx.
- First we need to format and partition it, from Terminal run these commands only if you are sure about the flash device name:
  - `sudo umount /media/flash-disk-mounted-folder`
  - `sudo mkfs.ext2 /dev/sdx1`
- If you have one hard disk and you have connected only one flash disk, it will take the name sdb, the command should be:
  - `sudo mkfs.ext2 /dev/sdb1`
- Then mount the flash back again
  - `sudo mkdir /media/flash`
  - `sudo mount /dev/sdb1 /media/flash`

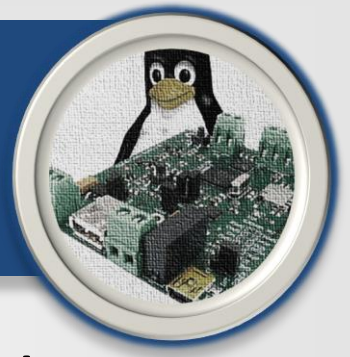
Putting System on Flash



- Copy all the OS files from the **</home/shatrix/minios/system>** to the mounted partition.
  - `cd  /home/shatrix/minios/system`
  - `cp -a * /media/flash`
- Finally install grub to the Flash device:

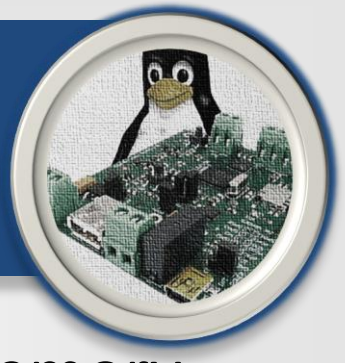
  `sudo  grub-install --root-directory=/media/flash /dev/sdb`
- Remember that the sdb will be the Flash disk device name in /dev, and /media/flash is the directory where the flash disk is mounted

Editing grub.cfg (UUID)



- To get the UUID for any partition or Flash:
  - `sudo blkid`
- copy the UUID for the required device and paste it in its correct place in the grub.cfg file on the Flash memory.
  - `sudo gedit /media/flash/boot/grub/grub.cfg`
- In the kernel line (`root=UUID=<disk-uuid>`)
- make it (`root=UUID=the-result-uuid-from-blkid`)

Congratulations ^__^



- If you make all these steps as it is, your flash memory can now boot with the new Linux OS.
- Restart your computer, and select to boot from the USB flash from your BIOS settings.
- The Grub from the Flash memory should start now, with one selection:
  - (LINUX Minimal OS FOR X86)
- Press Enter to start the system, and enjoy.
- When the system starts, go to the first console by typing Ctrl+Alt+F1 ,it will ask for login, type (root) and press enter, the system will start without asking for password.

eng.sherif.mosa@gmail.com

http://about.me/shatrix