

# Linux

## Administration

5 مارس 2005

الاصدار الاول

## بسم الله الرحمن الرحيم

الحمد لله رب العالمين والصلاه والسلام على محمدآ وعلى آله وصحبه وسلم , اما بعد

يعتبر هذا العمل محاوله منى لايجاد كتاب يتحدث عن اداره لينوكس .  
والدافع الى هذه المحاوله ، هو عدم وجود اى كتاب باللغة العربيه فى هذا المجال ، مما يشكل صعوبه للراغبين فى تعلم اداره لينوكس وبالتالي انتشاره فى بلادنا العربيه .  
فقد قابلتني شخصيا هذه المشكله فى بدايه تعلمى للنظام ، وبحث كثيرا عن اى كتاب باللغة العربيه لكى يسهل عمليه فهم النظام وادارته ، ولم اجد .  
وانا متأكد ان هذه المساله سوف تتكرر مع العديد من الراغبين فى تعلم لينوكس . لهذا -استعنت بالله- وقررت ان ابدأ فى كتابه كتاب يشرح اداره لينوكس .

وهذا الكتاب الذى بين يديك الان عباره عن اصدار يشمل العديد من المواضيع الهامه للراغب فى تعلم اداره النظام .  
وبالاطلاع على هذه المواضيع سيجد القارى نفسه قد استطاع الوقوف على ارضيه ثابتة ، تمكنه مؤقتا فى شق الطريق لتعلم لينوكس ، لحين اكمال هذا الكتاب ، والذى انوى جعله مرجع باللغة العربيه .  
مع الاخذ فى الاعتبار ، انه حتى بعد اكتمال هذا الكتاب -باذن الله- فان الراغب فى تعلم لينوكس لا يستطيع الاستغناء عن اللغه الانجليزيه ، لانها لغه النظام فى الاساس ، ولان المنات من المراجع مكتوبه بها .

ولعل السبب فى عدم تمكنى من اكمال هذا الكتاب الان ، هو اننى لم اعتاد على كتابه المذكرات والتلاخيص .  
فلا اتذكر مره واحده اكملت فيها ملخصا لاي ماده اثناء دراستى -سواء الثانويه او الجامعيه- فقد كنت اصاب بالملل من الكتابه والتلخيص ، ولو حتى لصفحات معدوده لايتعدى عددها اصابع اليد الواحده .  
ولهذا فان اتمام هذا العدد من الصفحات يُعد -بالنسبه لى- ليس امرا هينا .

ولكن كان الدافع لاستمرارى فى الكتابه ، هو ابتغاء الثواب من الله -عز وجل- باعتبار هذا الكتاب ، علما ينتفع به هذا من ناحيه ومن ناحيه اخرى لمساعدته اخواننا لتعلم علم قد يساهم -بشكل او باخر- فى احداث تغيير فى مجتمعاتنا

اذا فان هذا العمل الذى بين يديك يعتبر

• اصدار سوف يتبعه عدد من الاصدارات فى فترات متزامنه (من ثلاث الى سته اشهر) لحين اكتماله فى شكل كتاب كامل .

• محاوله لتشجيع الكثير منا لمحاوله الكتابه ، لانه لا يعقل وجود الكثير من المتحمسين والمحبين ل لينوكس ، ولا يحاول البعض منهم ازاله العوائق الموجوده فى سبيل انتشاره ، ولو حتى فى شكل المساهمه بكتابه اى عدد -ولو كان بسيطا- من الصفحات .

• يركز بشكل اساسى على مواضيع اداره لينوكس ، ولهذا فان هذا الاصدار -على الاقل- يتطلب معرفه بسيطه ب لينوكس ، لاننى لم اتعرض لمسائل مثل تنزيل النظام ونحوها ، معتمدا فى ذلك على وجود عدد لا باس به من المقالات تشرح مثل هذه المسائل بالاضافه الى كتاب الاخ فيصل يوسف .

• المواضيع الموجوده فى هذا الكتاب تعتمد فى تطبيقاتها وامثلتها على ال cmd line .

• تمت الاستعانه بنسخه ردهات 9.2 فى هذا الكتاب .

• اذا كان للقارى رأى او ملاحظه او تصحيح على هذا الكتاب فيمكن ارساله الى العنوان الالكتروني الوارد بالاسفل .

شكر خاص للاخ احمد ابو بكر ، الذى ساهم فى اخراج هذا الكتاب بهذا الشكل .



## الفهرس

- تمهيد
- تاريخ اليونكس
- ظهور اليونكس
- ميزه خاصه فى اليونكس
- unix و لغه البرمجه C
- unix بعد ذلك
- مشروع GNU
- بدايه لينوكس
- نظره على minix

### **. عالم لينوكس**

- كيف ينطق لينوكس
- ما هو لينوكس بالضبط
- اداره النظام
- تعدد المهام وتعدد المستخدمين
- مفتوح المصدر open source
- الملفات التنفيذيه فى لينوكس
- البرامج والاوامر
- اصدارات الكرنل
- اصدارات لينوكس
- انواع المستخدمين فى لينوكس
- رخصه GPL
- لماذا لينوكس

### **shell (نظره عامه)**

- كيف تعمل ال shell
- اهم مفاتيح التعامل مع ال shell
- ال shell لاول مره

### **basic command**

#### **الاوامر الاساسيه**

- التعامل مع الملفات
- التعامل مع المجلدات
- نسخ الملفات والمجلدات
- نقل الملفات والمجلدات
- عرض الملفات والمجلدات
- اوامر عامه
- ملخص الاوامر

## the linux documentation.

### وثائق لينوكس

- تمهيد
- النوع الاول من الوثائق on line help
- man page
- مفاتيح التعامل مع ال man page
- info page
- مفاتيح التعامل مع ال info page
- النوع الثاني من المساعدة
- الوثائق الموجوده على النظام
- النوع الثالث من المساعدة
- الوثائق الموجوده على الانترنت
- النوع الاول : tldp
- النوع الثاني : قوائم البريد و مجموعات الاخبار

## system startup and shutdown .

### عمليات فتح واغلاق النظام

- الخطوة الاولى Bios
- ال MBR
- الخطوة الثانيه boot loader
- مميزات grub
- ملف تهيئه grub
- الخطوة الثالثه kernel booting
- الخطوة الرابعه init
- ال runlevel
- كيفيه التحول من ال runlevel
- ملف inittab
- ملاحظات على الملف inittab
- ماذا بعدالتعديل فى الملف inittab
- نظره اخيره على init
- شكل المجلد rc\*.d من الداخل
- كيفيه حذف ال services
- اغلاق النظام : الامر shutdown

## ال shell للمره الثانيه (نظره اعمق)

- الخصائص العامه لل shell
- انواع ال shell
- كيف تعمل ال bash
- الفرق بين المتغيرات الكليه والجزئيه
- من خصائص ال bash
- الخاصيه الاولى : الامر history
- الامر export وتعديل ال variable
- لماذا ننشأ variable
- تعديل ال variable بطريقه دائمه
- الخاصيه الثانيه : الامر alias

## **the regular expression.**

### **العلامات والرموز**

- استخدامات ال metacharacter
- تنفيذ عدة اوامر فى الوقت الواحد
- ال stdin وال stdout
- ال piping و redirection

## **the hard ware .**

### **الهاردوير (نظره عامه)**

- اللوحه الام mother board
- الاجزاء المكونه لل mother board
- الاجزاء الاخرى فى ال mother board
- كيف يتعامل الهارد وير مع بعضه البعض

## **the text editing .**

### **المحرر vi**

- البدايه
- الحاله الاولى command mode
- الحاله الثانيه insert mode
- الحاله الثالثه ex mode

## **the file permission .**

### **تصاريح الملفات**

- فهم ال permission
- الاختلاف بين الملف والمجلد
- ال permission بطريقه اخرى
- تعديل ال permission
  - التعديل بالارقام
  - التعديل بالاحرف
- اوامر متعلقه بال permission
  - الامر umask
  - ضبط ال permission بالامر umask
  - الامر chown
  - الامر chgrp

## the filesystem & mount . نظام الملفات وال mount

- شكل ال filesystem
- عمليه ال Mount
- الملف fstab
- الامر mount
- خيارات الامر Mount
- النوع الاول : cmd line option
- النوع الثانى : mount option
- انواع ال filesystem
- مثال عملى للامر mount
- الامر umout

## the user adminstration . اداره المستخدمين والمجموعات

- تمهيد
- التعامل مع المستخدمين
- الامر useradd
- تعديل القيم الافتراضيه
- الامر userdel
- الامر usermod
- ملفات المستخدم على النظام
- الملف passwd
- ملاحظات على الملف passwd
- الملف shadow
- ملاحظات على الملف shadow
- كيفيه عمل disable للمستخدم
- التعامل مع المجموعات
- الامر groupadd
- الامر groupdel
- الامر groupmod
- ملفات المجموعات على النظام
- الملف group
- الملف gshadow
- اوامر متعلقه باداره المستخدمين
- نقل المستخدمين من نظام لآخر
- عرض بياناتك
- التحول الى مجموعه اخرى
- وضع ال passwd للمجموعه

## Filesystem Hierarchy Standard . الترتيب الهرمى للملفات FHS • ملخص FHS



## تمهيد

لينوكس linux هو نظام تشغيل قوى مستقر ثابت ومفتوح المصدر .  
اسسه " لينوس تورفالدس Linus Torvalds " طالب دراسات الكمبيوتر بجامعة هيلسينكى  
بفنلندا .

وكان هدفه من انشاء " لينوكس linux " هو تشغيل نظام " يونكس Unix " القوى والشهير على  
جهازه الشخصى فى المنزل . ( يونكس وقتها كان لا يعمل على معالجات intel الخاصه بالاجهزه الشخصيه  
Personal Computer بل انه صمم ليعمل على اجهزه من نوع mainframe على الاقل ، وليس على  
الاجهزه الشخصيه )

كان ذلك فى عام 1991 . وهو نفس العام الذى شهد اول اصدار من linux .

ولكن لكى نستطيع ان نفهم تاريخ لينوكس linux ان نتعرض اولا لليونكس Unix وتاريخه ثم بعد  
ذلك نتكلم بتفصيل اكثر عن لينوكس linux .

## تاريخ اليونكس UNIX .

يونكس Unix هو الاب الروحى لانظمه التشغيل الحالى .

وهو ايضا اقوى نظام تشغيل واكثرهم ثباتا حتى الان ، والمقصود بالثبات انه يمكنه العمل بقوه وكفاءة لفترات طويله قد تستمر (بدون مبالغه) لسنوات بدون اغلاق او حتى اعاده تشغيل restart . اما بالنسبه لقوته فترجع الى الاسباب التاريخيه لنشئته وتطويره ، وهو ما سنتعرف عليه .

فى منتصف الستينيات 1965 تكونت مجموعه من المتخصصين لانشاء نظام تشغيل يخدم الباحثين وهذه المجموعه مكونه من

- 1- جنرال اليكتريك General Electric .
- 2- معامل بيل Bell Laboratories والتي اصبح اسمها بعد ذلك AT&T .
- 3- معهد ماساشوستس للتكنولوجيا MIT .

وخرجت هذه المجموعه فى عام 1968 بنظام تشغيل يسمى MULTICS اختصارا  
**multiplexed Information Computing System**

وهذا النظام مدمج فيه مفاهيم تعدد المهام multi-tasking و تعدد المستخدمين multi-users واداره ملفات متعدده المستويات multi-level file management وواجهه المستخدم التفاعليه user interaction (يتفاعل ويتعامل بها المستخدم مع النظام ، وهى طبعا ليست كالواجهات ال graphical المعتاده اليوم ولكنها واجهه معتمده على سطر الاوامر command line)

\* يلاحظ هنا ان السبب فى وجود الخاصيتين الشهيرتين والمميزتين لانظمه Unix-like عموما وهما تعدد المهام و المستخدمين سببه تاريخى ، وهو الحاجه لنظام تشغيل يعمل على خدمه الباحثين كما اسلفنا .

## ظهور Unix .

هذا ، وان كانت بدايه مشروع MULTICS لثلاث مجموعات عمل كما ذكرنا ، الا ان معامل بيل bell lab لم تكمل هذا المشروع للنهايه وسحبت من هذا المشروع مجموعه العمل الخاصه بها والتي كانت بقياده " كين تومسون Ken Thompson "



كين تومسون

الا ان " تومسون " قام فى عام 1969 هو وعدد من المبرمجين الذين عملوا فى مشروع MULTICS باستئناف العمل فى معامل bell لانتاج نظام تشغيل خاص بهم دامجين فيه العديد من خصائص MULTICS ، وكان هدفهم ان يعمل هذا النظام على اجهزه من نوع minicomputer .

اما بالنسبه للمشاركين فى هذا المشروع فمنهم

- دينس ريتشى Dennis Ritchie

- رود كانادى Rudd Canaday

- دوج ماكلورى Doug McIlroy

وصمم نظام التشغيل هذا (Unix) وعمل فى البدايه على جهاز من نوع PDP-7 (والذى كانت تنتجه شركه DEC) مع القليل من البرامج utility معه .

- وبالنسبه لتسميته Unix فقد أطلق عليه هذا الاسم بواسطه احد افراد المجموعه وهو " بريان كرنيجهان Brian Kernighan " وذلك تشبها بنظام MULTICS .  
(كان قد اطلق عليه unics فى البدايه ثم تم تعديله ليصبح Unix)

- واصبح 1 يناير 1970 هو ال time zero للنظام حيث تم تشغيل اول اصدارات اليونكس .

- وفى العام التالى 1971 تم تشغيل Unix على جهاز PDP-11 ، ولك ان تتخيل مواصفات هذا الجهاز التى تعد بدائيه جدا هذه الايام .

فقد كان يحوى ذاكره عشوائيه RAM ذات 16 كيلو بايت (16K bytes) .  
وهارد ديسك حجمه 512 K bytes . (اى ما يعادل نصف حجم قرص مرن floppy disk من الموجود الان)

- قامت معامل بيل -فى البدايه- بالسماح للجامعات باستخدام (كرنل) Unix على انه منتج بحثى research product ، فباستطاعتهم ان يستخدموه فى ابحاثهم ويطوروا فيه على ان تظل الملكيه لمعامل بيل ، وتم توزيع اول نسخه منه مجانا .

(وليس معنى هذا ان يونكس مجانا او ان استخدام النظام مسموح به لاي شخص (مثل Linux) ، بل كانت معامل بيل تعطى تراخيص لاستخدامه ولانتاجه باسماء اخرى دون استخدام اسم يونكس)

- وكان ممن حصلوا على نسخه Unix ، اداره علوم الكمبيوتر بجامعة كاليفورنيا "بيركلى" computer science department .

وقام الباحثون بجامعة بيركلى بدراسه وتطوير Unix و اضافوا اليه الكثير من الخصائص وهذه الخصائص كانت من الاهميه بمكان حتى انه عند الكلام بايجاز عن من انشأ يونكس فيقال معامل bell وجامعة بيركلى .

- وأخرجت جامعه بيركلى فى عام 1977 الى النور نظام التشغيل المعروف BSD  
Berkeley Software Distribution

هذا النظام الخاص بجامعه بيركلى والمعتمد على يونكس اصبح بعد ذلك بقليل من اشد الاصدارات منافسه لليونكس Unix الذى تنتجه معامل bell

## ميزه خاصه فى Unix .

\* يلاحظ انه بالاضافه لمميزات نظام التشغيل يونكس العديده إلا انه كان يتمتع بميزه ، وهى كون الكرنل -وهو لب النظام- مفتوح المصدر وقتها ، وهذه الميزه ساعدت فى اكسابه هذه القوه والشهره العريضه ، وبالاعتماد على هذه الميزه استطاعت العديد من مراكز الابحاث والجامعات بل وايضا الشركات كتابه نظم تشغيل مماثله ل يونكس .  
وهذا بالفعل ما حدث مع جامعه بيركلى وايضا مع العديد والعديد من الشركات الكبيره والمعروفه ، فكل شركه من الشركات العالميه تقريبا اصدار خاص بها من اليونكس .

- فشركه sun microsystems الشهيره لها اصدار من اليونكس وهو Solaris الذى يعد من اكثر انظمه اليونكس شهرة .

- ايضا شركه IBM لها اصدار خاص بها من اليونكس يعمل على اجهزه ال work station التى تنتجها ، ويسمى هذا الاصدار AIX .

- وشركه Macintosh ماكنتوش (العتيقه) الشهيره بنظامها Apple Macintosh ، لها اصدار من اليونكس يدعى AUX . (يعرف ايضا ب AU/X)

- واصدار شركه نوفل Novell من اليونكس يسمى Unix ware .

- ولشركه hp (Hewlett Packard) اصدار يسمى HP-UX .

- ولشركه Compaq اصدار يسمى Digital Unix .

\* بل إن شركه مايكروسوفت MicroSoft كانت قد انتجت يونكس (بترخيص من معامل بيل AT&T) ايضا وذلك اواخر عام 1979 وكان اصدارها يسمى Xenix .

ويلاحظ ان هذا الامر قد القى بظلاله بعد ذلك على اصدارها MS-DOS والذى خرج الى النور فى يوليو 1981.

(شركه مايكروسوفت قد تأسست فى اغسطس عام 1975 على يد كلا من "ويليام هنرى جيتس" المعروف ب بيل جيتس و "بول ألين " )

هذا بالاضافه الى العشرات والعشرات من الاصدارات التى تأسست بناء على مفاهيم قواعد يونكس (Unix-Architecture) .

ولهذا فانك ستصادف كثيرا المصطلح Unix-like OS وايضا المصطلح Unices (كجمع طريق Unix) واللذان يشيران الى الاصدارات المتعدده و المعتمده على يونكس .

\* من هذا المنطلق فان لينوكس linux يُعد -وفى الحقيقه يُعرف بأنه- احد اصدارات Unix الخاصه بالعمل على الاجهزه الشخصيه Personal Computer .

## الـ Unix و C language .

ليس هذا كل شى فبعد إنشاء يونكس Unix (عام 1969) بثلاث سنوات وفى عام 1972 تمت اعاده كتابه الكرنل الخاص به بلغه برمجه جديده ، هذه اللغه (والتي انشئت من اجل كتابه Unix) هى لغه C الشهيره والقويه والمستعمله حتى الان . والتي قام بإنشائها > دينس ريتشى و بريان كرنيجهان < .



دينس ريتشى



بريان كرنيجهان

وبفضل هذه الخطوه الهامه (انشاء لغه البرمجه C) وعوامل اخرى ، استطاع Unix ان يصبح اقوى نظام تشغيل حتى الان واكثرهم استقرارا .

ذلك لان وجود هذه اللغه الجديده دفع العديد من الاكاديميين وطلبة الكمبيوتر فى الجامعات والمعاهد بل وايضا الهواه والمحترفين -hackers- الى تعلم هذه اللغه القويه ، وبالتالي رجعت كل هذه الجهود على يونكس Unix ، متمثله فى التطوير الذى لحق به على مدار سنوات .

بذلك يتضح لنا كيف استطاع نظام التشغيل Unix ان يصل الى هذه القوه .

ايضا تطورت لغه C بهذه الطريقه ، فقد كانت متاحه للجميع وعلى مستوى العالم ان يتعلموها وكانوا كلما اكتشفوا ثغره او عيب بها فانهم يرسلون بهذه الملاحظات الى " ريتشى " ويقوم هو ببناء على ذلك بسد الثغرات وتلافي العيوب ويدمج كل هذا فى الكود الخاص بها .

## Unix بعد ذلك .

قامت معامل bell بتطوير العديد من اصدارات يونكس Unix حتى كان عام 1983 وهو العام الذى طرحت فيه اول نسخه تجاريه غير مفتوحه المصدر سميت system3 ثم اتبعت بالنسخه systemV .

وبناء على هذه الاصدارات منعت معامل bell تداول كرنل يونكس المفتوح المصدر ، واصدرت تراخيص جديده غير التى كانت ساريه لسنوات مضت .

فكانت هذه الخطوه الغير مسبوقة سببا لغضب الكثير من المبرمجين والمهتمين ب Unix والذين ساهموا فى تطويره على مدار سنوات مما دفع بعضهم بقياده احد المبرمجين العاملين بمعهد ماساشوستس للتكنولوجيا MIT وهو " ريتشارد ستولمن Richard Stallman " الى الخروج بفكره انشاء انظمه تشغيل مفتوحه المصدر .

وكانت هذه الخطوه هى بدايه مشروع GNU .

## مشروع GNU .

عندما استلم " ستولمن " نسخه system3 عام 1983 ولم يجد الكود المصدر source code الذى اعتاد ان يحصل عليه مع الاصدارات السابقة من Unix (وذلك لكى يقوم باجراء التغييرات التى كانت تناسبه فى عمله وتناسب العاملين معه فى MIT) حاول ان يحصل عليه ولكنه اكتشف انه لا يستطيع الحصول عليه الا اذا وقع (sign) على اتفقيه عدم ' كشف ' الكود NDA Non-Disclosure Agreement الامر الذى يجعل من حصوله على الكود عديم القيمه .

وعلى اثر ذلك قام " ريتشارد ستولمن " بالاعلان عن مشروع GNU عام 1984 وبمعاونه العديد من المبرمجين والباحثين الاخرين الذين ايدوا الفكره .



ريتشارد ستولمن

\* وGNU تعد اختصار لـ **Gnu is Not Unix** والتى تعنى انظمه تشغيل مفتوحه المصدر تعمل بنفس قواعد يونكس لكنها لا تسمى يونكس . وتتنطق (جنو guh-noo) البعض -وانا منهم- ينطقها (جى إن يو) ، لكن النطق الصحيح لها طبقا لـ [www.gnu.org](http://www.gnu.org) هو الاول .

والسبب طبعا فى جعل الانظمه التى سينتجها هذا المشروع متوافقه مع يونكس من حيث البناء (الكرنل و نظام الملفات وتقريبا كل اوامر يونكس) وليست مختلفه معه وبالتالي يكون لها نظام وبناء اخر ، هو ان القائمين على اتمام هذا المشروع هم فى الاساس محترفى يونكس وتأقلموا على العمل معه لسنوات .

وايضا - وهو الاهم - ان نظام يونكس يمتاز بالعديد من المميزات ، فبناء نظام الملفات filesystem Architecture بالاضافه الى انسجام العمل بين برامجه programs ومنافعه utilitys يعطيه مميزات عديده منها مثلا عدم قابليته (للتعليق) وهو ما يسمى hang وغيرها من المميزات ، وبالتالي فالتنازل عن هذه الخصائص والمميزات التى اكتسبها على مدار سنوات من تطوير الالاف له وذلك لمجرد المخالفه فقط لا يعد من الحكمه .

وبالفعل قام المشاركون فى هذا المشروع بكتابه وانجاز الكثير من البرامج الضروريه لانشاء انظمه جديده ، ولكن العقبه التى واجهتهم هى كتابه برامج كبيره ومهمه مثل ال C compiler (والذى يتمثل دوره فى الوساطه بين لغه الجهاز الاسمبلى وبين لغه البرمجه C) .

ايضا ما يسمى بالمكتبات libraries ودورها مثل دور المكتبات بالمسمى المعروف لدينا ، فهى تحوى فهارس ومراجع وتعليمات عامه للاوامر موجود بها ما تحتاجه الاوامر عموما من معلومات وتوجيهات ضروريه (لكى لا تتكرر هذه المعلومات مع كل برنامج) لهذه الاوامر عندما يتم استدعاؤها ، ولهذا فهى تسمى system calls .

وبرامج اساسيه مثل محرر النصوص emacs ، وواجهه المستخدم bash .

هذا ولا ننسى اهم الاجزاء الا وهو الكرنل kernel .

كل هذه البرامج وغيرها كان لا بد لها ان تُنجز لكي يصبح هناك بالفعل انظمه تشغيل . الامر الذى استغرق عده سنوات للانتهاء من كل هذه البرامج .  
وبالفعل تم انجاز هذه الاعمال فى بدايه التسعينات . وتم اتمام المشروع واصبح له موقع على الانترنت هو <http://www.gnu.org> .

ايضا تم انشاء موقع على الانترنت لكي يكون بمثابة المستودع لهذه البرامج وهذا الموقع هو <http://www.fsf.org> ، و fsf اختصار ل free software foundation .

وتزامن الوقت الذى تم فيه الانتهاء من المشروع ، بالوقت الذى كتب فيه لينوس تورفالدز اول كرنل له (كان لا يتعدى فى حجمه 63 كيلو بايت )

فتزامن انجاز لينوس للكرنل الخاص به انتهاء مشروع GNU من كتابه كافه برامجه ولم يتبقى الا الكرنل الخاص بمشروعهم والذى يسمى Hurd . ( يشار له فى ال doc وصفحات ال man وال info ب GNU/hurd )  
فاصبح كرنل لينوس هو المتمم لمشروع GNU .

#### ويلاحظ هنا

انهم بالفعل انتهوا من كتابه الكرنل Hurd ولكن تم ذلك بعد اشتهار الكرنل المسمى linux الخاص ب لينوس. اى انه موجود ويتم العمل به بالفعل .

ايضا يلاحظ ان مشروع GNU هدفه هو اخراج انظمه تشغيل وليس نظام تشغيل وحيد و منفرد . وهذا ما يتم الان بالفعل ولكن شهره linux طغت على غيره . حتى وصل الامر بان اطلاق اسم linux الان اصبح يعبر عن الانظمه مفتوحة المصدر بصفه عامه .

## بدایه لینوکس .

لینوکس نظام التشغيل الشهير ، قام بإنشائه طالب (لاحظ انه طالب فقط) دراسات الكمبيوتر بجامعة هيلسينكي واسمه

“ لينوس بندكت تورفالدس Linus Benedict Torvalds ”

وقام بذلك في بدايات التسعينات .

\* ولكن الذى تجدر الاشاره اليه ، انه فى ذلك الوقت كان مشروع GNU تقريبا يعد منتهى ، وهذا المشروع كما اسلفنا سينتج انظمه تشغيل كامله ، الامر الذى لا داعى معه الى هذه المحاولات الفرديه التى لا تثمر مثل العمل فى مجموعات (كالGNU) ولهذا كانت بدايه لينوكس مختلفه .

كانت البدايه مع مشروع بحثى research كان يقوم به لينوس . linus



لينوس تورفالدز

وهذا المشروع معتمد على فكره بسيطه ولكنها هامه .

فقد كان لينوس يعمل على معالج (processor) من نوع intel 80386 (والذى كانت تنتجه شركه انتل قبل انتاج المعالجات pentium)

وكانت الفكره هى محاوله اكتشاف وتطبيق خاصيه جديده لم تكن معروفه انذاك تدعى (MMU Memory Management Unit) وهذه الخاصيه تجعل كل عمليه (process) تقوم بها الذاكره منفصله عن بقيه العمليات التى يتم تنفيذها الان ، مما يساعد على اجراء عده عمليات متزامنه فى وقت واحد . وايضا يكون لكل عمليه جزء segment خاص بها فى الذاكره (وليس كل العمليات مشتركه مع بعضها البعض) . وبذلك فى حاله فشل اى عمليه فلا يؤثر ذلك على بقيه العمليات . وقد كان الوضع قبل ذلك يتمثل فى ان اى عمليه (process) تقفل (crash) فى اداء مهمتها فانها تقوم بعمل انهيار للنظام ككل system down .

وكانت انظمه التشغيل التى تعمل على المعالج 80386 وقتها هى Windows الخاص بال work-group و Minix . والذان لم يكونا موفران لهذه الخاصيه بعد .

فبدأ بكتابه ثلاث برامج

\* الاول برنامج بسيط جدا ، وظيفته كتابه الحرف A باستمرار على الشاشة .

\* والثانى ايضا مثل الاول لكنه يكتب الحرف B .

\* اما الثالث فكان اكبر نسبيا من الاول والثانى ، ووظيفته ابقاء ال processor فى وضع آمن protected mode ، وايضا وضع البرنامج الاول والثانى فى جدول schedule ثم دمج الناتج منهما معا و اخرجهم على الشاشة .

وعندما قام بتجربه هذه البرامج ووجد ان الناتج الذى يظهر على الشاشة ABABAB.... علم ان الفكره نجحت وان باستطاعته ان ينشأ نواه kernel تقوم بمهام متعدده (multi tsking) فى نفس الوقت .

استطاع لينوس ان يكتب اول كرنل له ونجح فى تشغيل C compiler و Bash الخاصين ب GNU تحت هذا الكرنل .

بدأ ذلك فى شهر ابريل من عام 1991 وانتهى منه فى اغسطس ثم بعث برسالة على الانترنت (والذى كان وقتها عبارة عن شبكة داخلية كبيرة تخدم الجامعة/الجامعات) الى مجموعه الاخبار الخاصة ب Minix (سيتم التحدث عنه لاحقا) يخبرهم فيها بما توصل اليه ويدعوهم الى المشاركة معه.

وهذا هو نص الرسالة.

```
From: torvalds@klaava.Helsinki.FI (Linus Benedict Torvalds)
Newsgroups: comp.os.minix
Subject: What would you like to see most in minix?
Summary: small poll for my new operating system
Message-ID: <1991Aug25.205708.9541@klaava.Helsinki.FI>
Date: 25 Aug 91 20:57:08 GMT
Organization: University of Helsinki
Copyright ©2000 SYBEX , Inc., Alameda, CA www.sybex.com
The Linux Kernel
111
Hello everybody out there using minix -
I'm doing a (free) operating system (just a hobby, won't be big
and professional like gnu) for 386(486) AT clones. This has been
brewing since april, and is starting to get ready. I'd like any
feedback on things people like/dislike in minix, as my OS
resembles it somewhat(same physical layout of the file-system (due
to practical reasons)among other things).
I've currently ported bash(1.08) and gcc(1.40), and things seem to
work. This implies that I'll get something practical within a few
months, and I'd like to know what features most people would want.
Any suggestions are welcome, but I won't promise I'll implement
them :-)
```

Linus (torvalds@kruuna.helsinki.fi)

PS. Yes - it's free of any minix code, and it has a multi-threaded fs.  
It is NOT protable [sic](uses 386 task switching etc), and it  
probably never will support anything other than AT-harddisks, as  
that's all I have :-).

## نظرة على Minix .

سبق ذكر كلمه مينكس Minix مرتين فى الصفحات القليله السابقيه .

الاولى عند الكلام عن الاجهزه التى يعمل عليها المعالج intel .  
والثانيه فى رساله لينوس الى مجموعه الاخبار news group .

وMinix هذا عباره عن نظام تشغيل مفتوح المصدر .  
كتبه " أندرو تانينباوم Andrew Tanenbaum " استاذ علوم الكمبيوتر بالجامعه المفتوحه  
بأمرستردام Free University in Amsterdam .

فقد كان هذا البروفيسور -الالمانى الاصل- يدرس لطلابه بطريقه عمليه كيف يكتب نظام التشغيل  
وذلك على احد نسخ يونكس وهى النسخه المفتوحه المصدر من BSD .  
ولكن بعد تغيير نظام رخصه الBSD فى الثمانيات وعدم تمكنه من الحصول على الكود ، قرر  
تانينباوم ان يكتب بنفسه نظام تشغيل موافق ل يونكس (Unix-like) ليحل محل النسخه العمليه التى  
كان يستخدمها .



أندرو تانينباوم

### ويلاحظ

- \* ان تانينباوم عندما كتب نظامه كان هدفه جعل النظام صغيرا قدر الامكان وذلك لكى يتمكن من استخدامه فى عمليه الشرح لطلابه .
- \* ولهذا اطلق عليه Minix اختصارا لـ MINI uniX .
- \* وكان يتكون من 12 الف سطر، ولهذا لم يكن Minix غنيا فى حد ذاته كغيره من انظمه اليونكس
- \* كان الهدف ان يعمل هذا النظام على معالجات intel \*286 .
- \* كان متاحا لاي شخص على الانترنت ولكن للاستخدامات التعليميه .  
( وهذا يفسر كيف كان يتدارسه "لينوس" وهو فى فنلندا بينما كان "أندرو" فى هولندا ) .

يلاحظ ايضا ان لينوس كان على اتصال ب تانينباوم وذلك لمحاوله جعل Minix يعمل على 386\*  
وايضا لدعم اكبر للهاردوير وغيرها من البناء الداخلى . ولكن فى النهايه لم يتفقا على العمل معا .

المحادثات والحوارات التى دارت بين لينوس وتانينباوم موجوده على الموقع التالى  
[http://www.dina.dk/~abraham/Linus\\_vs\\_Tanenbaum.html](http://www.dina.dk/~abraham/Linus_vs_Tanenbaum.html)



## عالم لينوكس .

سيتم التحدث في النقاط التالية على بعض ملامح وصفات لينوكس .

### كيف ينطق لينوكس .

شغلت هذه المسألة تقريبا كل المهتمين بلينوكس . وليست المشكلة عندنا نحن العرب فقط بل هي موجودة ايضا عند الاوربيين والامريكان وغيرهم . والسبب في عدم الاتفاق على نطق لينوكس بطريقة واحدة في العالم كله هو ذاته الاختلاف في نطق كلمات مثل hello من بلد لآخر ومن لغة لآخرى .

فتقريبا كلمة hello موجودة في معظم لغات اوربا ، ولكن لكل بلد طريقته في نطقها ، وكذلك لينوكس . الامر الذي جعل لينوس يسجل بنفسه كيف ينطق لينوكس لكي يحل هذه المشكلة .

فعلى الموقع <ftp://ftp.funet.fi/pub/linux/PEOPLE/Linus/SillySounds> يوجد ملف صوتي (حجمه 40kb) يمكنك تنزيله والاستماع اليه (ستجده باللغة الانجليزية وآخر بالسويدية) وفيه يقول لينوس بصوته ،

hello this is linus torvalds , i pronounce linux as linux  
أهلا , هذا هو (أنا) لينوس تورفالدز , انا انطق لينوكس ك لينوكس .

ولكن كيف نطقها نحن كعرب . اذا اردنا نطقها بطريقة صحيحة فلا بد من نطقها مثل نطق lee nox (اي نكسر اللام ونضم النون لينوكس )

\* حقيقة يوجد العديد من خبراء لينوكس ينطقونها lee nex (لينكس) ، وما زلت انا ايضا انطقها في بعض الاحيان lee nex ، ..... اذا المهم ان تكون على معرفه بكيفية نطق linux ثم بعد ذلك انطقها كما تعودت ، فالمستمع لك سيفهم ما تقصده .

### ما هو لينوكس بالضبط .

لينوكس ، يطلق ويقصد به عموما نظام التشغيل ككل . ولكن بالنسبة للمختصين او لدارسي linux فلا بد ان يعرفوا ان linux هو الكرنل بالتحديد .

والكرنل هذا هو المسئول عن اداره النظام كله . فهو الذي يدير الهارد والسوفت وير . \* وهذا الكرنل ليس ملكيه شائعه للجميع بل هو مملوك Copyright ل لينوس ، الذي يسمح لاي فرد كان ان يستخدمه في اي غرض .

\* اذا مما يتكون نظام التشغيل .

يتكون نظام التشغيل من عدة اجزاء

- . **kernel** الكرنل
- . **Environment** وهى البيئات التى يتعامل معها المستخدم ليتصل بالنظام ، وهى اما ان تكون graphical مثل gnome او kde ، او تكون متمثلة فى سطر الاوامر cmd line .
- . **File structure** المسئول عن اداره نظام الملفات .
- . **Programs package** وهى البرامج المختلفه سواء من انتاج fsf او من انتاج الشركات التى تنتج لينوكس .

## إدارة النظام .

لاداره النظام فى linux نوعان

الاولى الاداره عن طريق ال cmd line (سطر الاوامر) . وميزه هذه الاداره انها توفر للمدير التحكم الكامل فى النظام . فكل شى يمكن ان ينفذ بال cmd line .  
ذلك لان linux او Unix والانظمه المتوافقه معه فى الاساس انشئت بهذا الاسلوب من البدايه فهذا النوع من الاداره هو الاقدم بل والاكفأ .

الثانيه وهى الاداره عن طريق الgui اى الواجهه الرسوميه Graphical User Interface  
وهى الطريقه الاحداث ، ولكن هذه الطريقه لها بعض العيوب منها

- . الاعتماد عليها فى الاداره لا يمكن المدير من الاستفادة بكل مميزات النظام .  
ذلك لانها تجعل المدير ملزم بما توفره له من امكانيات فلا يستطيع تجاوزها (فالزر tab  
الفلانى يفعل كذا وهذا يفعل كذا وبالتالي فالمدير محصور بهذه الادوات فى ادارته ، بعكس ال  
cmd line ، فبه تستطيع ان تفعل بالنظام ما تشاء انت وليس ما يريده هو بك)

. ال gui تستخدم وتستهلك الكثير من موارد النظام ، فلتشغيل ال gui او ال X window تحتاج  
على الاقل الى 128 RAM فى حين ان النظام يستطيع العمل بكفأه وعن طريق ال cmd line  
ب 64 RAM واقل ، مما يعنى استهلاك موارد الجهاز فى عمليات بامكاننا الاستغناء عنها .

. توقع دائما امكانيه عدم استجابته ال X window للعمل عند محاوله تشغيل النظام start up  
وفى هذه الحاله ستجد نفسك وجها لوجه مع ال cmd line الذى كنت تتهرب منه . فلا بد من  
الاستعداد لهذا اليوم .

(من المعروف عن لينوكس انه نظام التشغيل المفضل للمحترفين hackers وايضا لمخترقي الانظمه crackers  
لاختراق الشبكات على الانترنت ذلك لانهم يستخدمون ال cmd line وقدراتها ، والتى توفر لهم كل ما يحتاجونه  
لاداء هذه المهام)

## تعدد المهام و تعدد المستخدمين .

يتميز لينوكس بخاصيه مميزه جدا ، بل ان التحدث عن مميزات لينوكس لا بد ان يتصدره هاتين الميزتين ، وهما ، تعدد المهام وتعدد المستخدمين

فتعدد المهام يعنى ان النظام باستطاعته اداء اكثر من مهمه فى نفس الوقت

وقد يتمثل الفرق بين لينوكس وبين ويندوز على سبيل المثال -اذا اعتبرنا ان هذا الاخير متعدد المهام ايضا- فى الوقت الذى يستطيع فيه نظام التشغيل الاستمرار فى اداء مهامه .  
فنظام ويندوز ان اردت ان تختبره ، فلتجعله يعمل يوما كاملا مثلا بدون اغلاق ، ثم بعد ذلك حاول ان تختبره ، وستجد ان النظام قد فقد الكثير من قدراته

اما تعدد المستخدمين ، فيعنى ان اكثر من مستخدم باستطاعته الاتصال بالنظام فى نفس الوقت ، وبالتالي اداء المهام الذى يريدونها

وقد نلاحظ ان الميزتين السابقتين متداخلتان.فمعنى اتصال اكثر من مستخدم بالنظام هو اداء اكثر من مهمه فى نفس الوقت

ويلاحظ ان معنى تعدد المستخدمين بالنظام ، لا يعنى وجود مثلا خمس او حتى عشره مستخدمين يستعملون الkey board و mouse فى نفس الوقت ، فاتصال المستخدمين بالنظام له صور عديده ، منها على سبيل المثال -لا الحصر- الاتصال بسيرفر فى وقت واحد لاكثر من فرد

## مفتوح المصدر open source .

يعنى مصطلح مفتوح المصدر او open source ، ان الكود المكتوب به نظام التشغيل متاح لاي فرد يريد ان يرى كيف كتب نظام التشغيل لينوكس .

وهذه المساله قد تكون فى نظر البعض قليله الاهميه ، الا ان الحقيقه ان هذه المساله هامه جدا من عده نواح .

. فمن ناحيه الفرد . يمكن لاي شخص عنده الرغبه فى تعلم كيفيه كتابه انظمه التشغيل ان يستعين بهذا " الكود المفتوح " ليرى كيف تعمل وتكتب انظمه التشغيل .  
فيعتبر الكود المفتوح فى هذه الحاله ، مثال عملى على كتابه البرامج والانظمه .

. ومن ناحيه الدول . تعد هذه المساله هامه جدا ، ذلك لان الدول تمتلك العديد من الاجهزه الحساسه جدا ، مثل الوزارات والهيئات المختلفه . وبالتالي فان نظام تشغيل مفتوح المصدر يعد الحل الامثل لهذه الحالات ، لانه يمنع شفرات التجسس ، التى لا يمكن التعرف عليها بسهوله مع الانظمه والبرامج مغلقة المصدر .  
واليك بعض الامثله على اهميه هذه المساله

. تستخدم وزاره الدفاع الامريكى لينوكس ، بسبب هذه الميزه ، وعندها خبراء يعرفون ما الذى يعنيه كل سطر فى نظام لينوكس .

. طلبت روسيا من شركة مايكروسوفت ال source code لنظامها ويندوز ، لكي تسمح باستخدامه في هيئاتها المختلفه . وكذلك فعلت اسرائيل من قبل .

### الملفات التنفيذي في لينوكس .

من الاسئله التي تدور في اذهان المستخدمين الجدد للنظام ، لماذا لا توجد البرامج ذات الامتداد exe في لينوكس .

والاجابه بسيطه وهى ، الملفات في لينوكس غيرها في ويندوز ، فالقاعده العامه في لينوكس انه لا يوجد لملفاته اى امتداد .

بل ان الذى يحدد نوعيه الملف هو التصريح الممنوح لهذا الملف .  
فبهذا التصريح يكون الملف doc اذا كان التصريح الممنوح له r او w .  
وهذا الملف ايضا قد يصبح برنامج اذا تحول التصريح الممنوح له الى x .

وبهذا الكتاب فصل كامل يشرح هذه التصاريح ، وما تعنيه بالضبط ، كيفيه تغييرها والتحكم فيها .

### البرامج والاوامر .

يتم التعامل مع لينوكس من خلال الاوامر .  
وهذه الاوامر تسمى ايضا في لينوكس بالبرامج ، فان سمعت احدا يقول ، ان برنامج ls يقوم بعرض محتويات المجلد باكثر من صيغه . فلا تندهش لذلك .

ولعل سبب وجود هاتين التسميتين يتلخص في ان "الامر" قد يُعرف بانه ما يكتب على سطر الاوامر cmd line ، اما "البرنامج" فهو اسم هذا "الامر" عند تشغيله (run) .

## اصدارات الكرنل .

احدى الصفات التى تميز الكرنل هى ارقام اصدارات هذا الكرنل .

فالرقم الذى يخرج به الكرنل له معنى ابعد من كونه رقم مسلسل ، ولهذا الرقم اصداران

- . اصدار زوجى مثل 2.2 و 2.4 و 2.6 .
- . اصدار فردى ويكون مثل 2.3 و 2.5 وهكذا .

فان كان هذا الرقم " فردى " فمعنى هذا ان هذا الكرنل تحت التطوير والتجربه ، وبالتالي فان المتعاملين معه هم الباحثين والمطورين الذين يختبرون هذا الاصدار من الكرنل وبالتالي يتم تصحيح الكود ان كان فيه خطأ .

وان كان هذا الرقم " زوجى " فمعنى هذا ان هذا الكرنل تم اختباره وبالتالي يمكن الاعتماد عليه . وهذا بالطبع الكرنل الذى نتعامل معه نحن المستخدمين .

ويسمى الكرنل ذو الرقم الفردى ب **development release** او الاصدار الذى تحت التطوير .  
ويسمى الكرنل ذو الرقم الزوجى ب **stable release** او الاصدار المستقر .

ايضا رقم الكرنل هذا عبارته عن ثلاثه اجزاء

- . **major** ويعنى التغييرات الكبيره التى تطرأ على الكرنل
- . **minor** والتى تعنى التغييرات الصغيره فى الكرنل
- . **revision number** والتى بدورها تعنى التعديلات التى يتم ادماجها فى الكرنل

وكمثال لهذه الارقام فان الكرنل يكون له الشكل التالى 2.4.22

فرقم 2 دليل على ال **major** ومثاله ، عندما تم اصدار الكرنل 2 هذا -عام 1996- فانه اصبح كرنل يعمل بنظام ال **modules** (قدره الكرنل على تحميل اجزاء من الكود الخاص به اثناء عمله)  
والرقم 4 دليل على ال **minor** ، والتى تعنى بمعنى بسيط ، عمل **up grade** للخصائص التى يوفرها الكرنل ، وهى بالفعل موجوده فيه .  
اما الرقم 22 فهو دليل على ال **revision** والتى تعنى ، عندما يتم اكتشاف **bugs** فى الكرنل ويتم اصلاحه فانه ياخذ رقم **revision** جديد (ولا فرق ان يكون فردى او زوجى ، بعكس الاثنان السابقان)

وقد تضيف الاصدارات المختلفه مثل ردهات و غيرها ، رقم اضافى الى هولاء الثلاثه ، ويكون بالتالى رقم الكرنل بهذا الشكل 2.4.22-8 . وهذا الرقم الاضافى بمثابة **patch** خاص بالاصداره ذات نفسها .

## اصدارات لينوكس .

من الاوامر الشائعة والمعروفة عن لينوكس ، انه يوجد العديد والعديد من اصداراته . وقد يسبب هذا الامر سوء فهم عند البعض -خاصه الجدد- ، بسبب التعود على ان نظام التشغيل لا بد ان يكون مملوك لشركة "تجاريه" مثلما هو الحال مع ويندوز ومايكروسوفت . فلا يستطيع استيعاب كون نظام تشغيل له اكثر من خمسين شكلا .

والسبب في وجود هذا العدد الكبير من اصدارات لينوكس هو كون هذا النظام مفتوح المصدر ومجاني (لا يسمح لينوس ببيع الكرنل على الاطلاق) مما يجعل العديد من الشركات والجامعات والمنظمات تستخدمه لاغراضها المختلفه .

فالشركات تستخدمه بهدف الربح . والجامعات تقوم بتعليمه الى طلابها . والمنظمات -الغير ربحيه- مثل debian تستخدمه لاغراض البحث والتطوير وخلافه .

وكل هؤلاء -مع تعدد دوافعهم لاستخدام لينوكس- يشاركون في تطوير هذا النظام في شكل الاصدارات التي تخرج الينا .

فكما حدث مع Unix من قبل ، يحدث الان مع لينوكس وبصوره اكثر سرعه واتساعا .

وعلى الموقع [www.linux-iso.org](http://www.linux-iso.org) يوجد الكثير من اصدارات لينوكس ، وعددهم قريب من الخمسين اصداره .

## انواع المستخدمين في لينوكس .

يوجد في لينوكس نوعان من المستخدمين

- 1- مدير النظام ويسمى Administrator او root
- 2- المستخدم العادي ، وهو اى مستخدم اخر موجود بالنظام غير ال root .

والمدير او ال root ، يستطيع فعل اى شى وكل شى في النظام ، في حين ان المستخدم العادي مقيد في تعامله مع النظام بوظائفه فقط ، ولهذا فهو لا يستطيع اتلاف شى .

ولهذا فان استخدام ال root في العمليات العاديه ( كقراء الملفات وغيرها ) خطير جدا ولا ينصح به ابدا ، الا اذا كنت تنوى الانضمام الى اولئك الذين (يخربون) انظمتهم بغير قصد . والسبب كما اسلفنا هو ان النظام لا يعارضه -ابدا- في اى وظيفه يقوم بها .

## رخصة GPL .

المصطلح GPL سيقابلك كثيرا وانت تتعامل مع لينوكس وبرامجه . وهى تعنى رخصة gnu العامه ، او **gnu general public licence** .

وبمعرفة تاريخ ال unix ، والتطور الذى مرت به رخصه ، يتضح لنا معنى هذه الرخصة .  
فرخصة gpl ظهرت بظهور مشروع gnu والذى ظهر بدوره كنتيجة لرخصة unix الصادره عام 84 والتى جعلت unix مغلق المصدر .

وتهدف رخصة gpl الى اتاحة البرامج ليس فقط فى صورتها الاخير (binary) بل ايضا مع ال (source code) الخاص بها .

(يكتب البرنامج اولا بلغة برمجه -وهذه هى نسخه ال source code- ثم بعد ذلك يحدث له compile اى تجميع -وهذه هى نسخه ال binary- وما نحصل عليه نحن كمستخدمين ، هى النسخه ال binary)

وهذا هو معنى كلمه free الملازمه لمشروع gnu ، اى متاحه بالكود المصدر ، وقد تكون بعد ذلك بمقابل مادي او حتى مجانيه .

وبذلك تعطى رخصة gpl للمستخدم عده حقوق

- . فهى تتيح له الكود المصدر للاغراض التعليميه
- . وتسمح له بان ينسخ البرامج
- . وايضا بان يعدل فيها
- . وبامكانه اعاده بيعها ، ولكن كل هذا بشرط توفير الكود المصدر لما قام به من تعديلات لغيره .

ومثال على ذلك ، الشركات التى تنتج لينوكس ، فهذه الشركات تحصل على الكود المصدر للبرامج المختلفه من اماكن متعدده ، ثم تقوم باجراء التعديلات التى تراها مناسبه ، ثم تقوم ببيع ما قامت بانتاجه .

وبعد هذا كله فهى ملزمه باتاحه ال source code الناتج عن كل العمليات التى قامت بها فى تعديل البرامج .

## لماذا لينوكس .

قد يتسأل البعض ، لماذا البدايه من جديد مع نظام تشغيل لم نتعود عليه مثل windows ، ويحتاج لوقت اطول من غيره لكي نتعلمه .  
ايضا لماذا هذا العناء في حفظ كل هذه الاوامر والتعامل مع الشاشة السوداء console .  
وغيرها من التساؤلات وعلامات الاستفهام .

وللرد على من تدور في رأسه هذه الاسئله ، يجب ان

- . اولا النظر من زاويه الفرد وما يجنيه من تعلم لينوكس
- . ثانيا النظر من زاويه اصحاب العمل
- . ثالثا النظر من زاويه بلادنا ومصالحها

## فاولا من ناحيه الفرد .

المسأله ليست كما يتصور البعض . ان ما يفعله لينوكس يفعله ايضا ويندوز ، فهذه النظرة قاصره جدا .

ف لينوكس كما هو معروف عنه صنعه المبرمجون بانفسهم ولانفسهم .  
وبالتالي فلا تتوقع بعد مرور عده سنوات ان قدراتك ومعلوماتك الشخصيه ستتساوى في الحالتين (مع لينوكس ومع ويندوز).

فالتعامل مع لينوكس وعالمه مختلف ، ذلك لان تاريخ النظام وطريقه نشئته و تطويره لا بد ان تلقى بظلالها على تفكيرك ونظرتك الى عالم ال Information Technology- IT- عموما .

اذا انت مع الوقت وبسهولة يمكنك ان تكون مطور (developer) لهذه الانظمه او مبرمج (programer) او صانع لبروتوكول او حتى لنظام تشغيل جديد .

اما مع مايكروسوفت فانت في معزل عن كل هذا ، وانت فقط الخطوه الاخيريه في عمليه الانتاج العملاقه التي تدر عليها المليارات . لهذا فليس من الغريب ، القول انك مع مايكروسوفت قد تحولت من مجرد user الى super user ، فانت تحفظ فقط ما الذي تفعله هذه الاداه وكيف .

ليس هذا فقط ، بل فحتى بعد دراسه مناهج مايكروسوفت لا تستطيع تطبيق هذه الدراسات -بشكل قانوني- الا بعد دفع التراخيص .اذا فانت تعرف ولا تستطيع ان تنفذ وتطبق .  
فعدم امتلاك تكاليف هذه التراخيص (وهو ما يهم مايكروسوفت) يعنى ان دراستك لمناهجها قد تعتبر اكتساب معلومات عامه جديده وممتازة ولكن بمبلغ كبير .

ان كان هذا هو الوضع مع مايكروسوفت ، فالوضع في لينوكس يختلف  
فان كنت لا تعلم . فقط اتصل بالانترنت واحصل (وبدون مبالغه) على المئات من الكتب .  
ليس هذا فحسب ، تريد ايضا ان تطبق ما تعلمته على مشروعك الصغير ذو الميزانيه المحدوده ، طبق ايضا وبدون مقابل ولا تراخيص .

## ثانيا النظر من زاوية صاحب العمل .

هل سألت نفسك يوما لماذا تتركنا شركه مايكروسوفت ننسخ برامجها التى تكبدها بلا شك خسائر كبيره ظاهريا ؟

مع انها فى استطاعتها -عن طريق مبرمجيتها- ان تجعل برامجها غير قابله حتى لفكره النسخ !!

والاجابه : هذه هى الطريقه الجديده والغير تقليديه فى البيع وكسب العميل .او بمعنى اخر اترك للعميل قرش اليوم واحصل غدا منه على مائه قرش .

فمعظمنا عندما بدأ تعلم الكمبيوتر كانت بدايته مع Windows و Microsoft Office ، والعديد منا متقن العمل بهما .

بعد ذلك هل تعتقد ان اى صاحب عمل عندما يريد تحديث عمله ، هل سيأتى مثلا بال Office الخاص بشركه Novell او Macintosh مثلا .

الاجابه ، طبعا سيأتى بما هو معروف للجميع وما تعلمه الجميع ، وبذلك فهو الان لا يملك القرار .

اى ان مايكروسوفت تتركنا الان ننسخ برامجها لنكون نحن غدا ورقه الضغط(التى لا تضغط بها هى) على اصحاب العمل .

ولكن الامر مع لينوكس مختلف نوعا ما بالنسبه لصاحب العمل . فالان يوجد نظام تشغيل اقوى من ويندوز و...،...،... ومجانى . فلماذا يدفع صاحب العمل الان الالاف للحصول على نظام يوجد الاقوى منه والارخص .

هذا بالاضافه الى كون البرامج المكتبية التى تستخدم غالبا (مثل ال Office) والخاصه بـ لينوكس مشابهه لتلك التى تنتجها مايكروسوفت .

(فى الحقيقه تم كتابه هذا الكتاب بواسطه open office وهو الاوفيس الرئيسى مع معظم اصدارات لينوكس ، بل اننى استخدم النسخه التى تعمل تحت ويندوز !! ويمكنك تنزيل هذا الاوفيس مجانا من [www.openoffice.org](http://www.openoffice.org) ) وحجمه اكبر بقليل من 60 ميجا بايت .

ولا يتصور البعض ان الكثير سيعزف عن تعلم لينوكس وبالتالي سيضطر اصحاب العمل الى الرجوع ثانياه الى ويندوز .

بل على العكس ، ينتشر لينوكس وبسرعه وخاصه بين المتخصصين والخبراء فى مجال ال IT . وهذه عن تجربه وملاحظه شخصيه .

فبعد دخول الشخص الى مجال ال IT ويتعرف على امكانيات لينوكس واحتياجات السوق ، فتراه قد تحول بعد ذلك الى linux .

وهذه بعض الاسعار للبرامج التجارية . التي قد يستعاض عنها ب لينوكس .  
لاحظ اسعار windows server والتي تبدأ ب 849 دولار.

استعرتها من الكتاب الجيد (linux for dummies)  
والذي قد استعارها هو ايضا من موقع amazon.com بتاريخ 2001/2/3 .

(ملاحظه : موقع امازون هذا ، هو اول موقع تم انشاءه لبيع الكتب على شبكة الانترنت online bookstore  
والذي تأسس في يوليو 1995 بمدينة سياتل الامريكية وعندما تحول الى لينوكس عام 2001 استطاع توفير حوالى  
27 مليون دولار سنويا)

To get a flavour of the value of Linux, here are some prices for commercial software as listed at  
www.amazon.com. All prices are in \$USA, as listed on 2001-02-03, with discounts. Roughly equivalent  
Linux software is included on almost any Linux CD (but with no restrictions on the number of clients). In  
addition, the hardware for Linux is MUCH cheaper, since Linux can run all services on a single server.  
Microsoft Windows 2000 Server (5-client)--\$848.99; Microsoft Exchange 2000 Server (5-client)--\$1,279.99;  
Microsoft Outlook 2000 (1-client)--\$94.99; Systems Management Server 2.0 (10-Cals)--\$994.99; Proxy Server  
2.0--\$886.99; Microsoft SQL Server 2000 Standard Edition (5-client)--\$1,229.99; Microsoft SQL Server 2000  
Standard Edition (1-user License)--\$4,443.99; Microsoft BackOffice Small Business Server 4.5 NT (Add-On  
5-CAL)--\$264.99; Windows NT Server Prod Upgrade From BackOffice SBS Small Bus Server  
(25-client)--\$558.99; Microsoft Windows 2000 Advanced Server Upgrade (25-client)--\$3,121.99; Microsoft  
FrontPage 2000--\$129.99; Microsoft Internet Security and Acceleration Server --\$664.99; Site Server  
Commerce 3.0 (25-client)--\$4,092.99; Visual C++ 6.0 Professional Edition with Plus Pack--\$525.99;  
Microsoft Visual Basic Enterprise 6.0 with Plus Pack--\$1,128.99; Microsoft Visual Sourcesafe 6.0  
CD--\$469.99; Microsoft Office 2000 Standard (1-client)--\$384.99; Adobe Photoshop 6.0--\$551.99; Microsoft  
Plus Game Pack--\$19.99

### ثالثا النظر من زاوية بلادنا .

تعاملنا مع الشركات المحتكره العالميه ، يعنى ببساطه استنزاف بلادنا اكثر واكثر .

وما نحصل عليه ليس الا خدمات ، وهذه الخدمات مثل غيرها ، قد نحصل عليها مجانا ، وقد ندفع  
فى سبيلها مليارات .

وهذا بالفعل ما يحدث ، فبلد مثل السعوديه يمكنها توفير 7 مليارات ريال اذا اعتمدت على نظام  
لينوكس فى العديد من الاعمال .

المصدر من مقالة الدكتور د. توفيق الربيعه والتي سبق نشرها في جريدة "الرياض" بعنوان "لينكس هو الحل" التي تحدث بها عن  
امكانية توفير مبلغ يصل الى 7 مليارات ريال سعودي خلال 10 سنوات في حالة اعتماد نظام لينكس بديلا عن نظام الويندوز.

ويكفيها فقط هذا المثل .

(يلاحظ ان هذا المبلغ الذى سيتم توفيره ناتج عن تراخيص استخدام ويندوز ، وايضا المئات من البرامج التى تعمل  
تحتة).



## ال shell ( نظرة عامه ) .

ال shell هي الشاشة السوداء التي نتعامل من خلالها مع النظام فمن خلالها نمرر له الاوامر ومن خلالها ايضا نستقبل منه نتيجة هذه الاوامر ، ولهذا فهي تسمى ايضا ال **command line** .  
وينظره اكثر دقه فهي الواجهه التي نتعامل بها مع الكرنل ، المسئول الرئيسي عن اداره ال hardware و ال software بالنظام.

### كيف نتعامل معها

صيغه الاوامر الذي تدخل الى ال shell متعدد

**1** فابسطها كتابه الامر منفردا مثل الامر **ls** (لعرض المحتويات) ثم ضغط زر **enter**

**2** وقد ياتي مع الامر حرف ، الهدف منه تعديل الصيغه الاساسيه لهذا الامر فالامر **ls** اذا اتى معه حرف **L** مثلا فانه يغير شكل الناتج من هذا الامر ، وهذا الحرف يسمى "خيار" او **option** ويكون مسبقا في الغالب بالعلامه "-" "dash" ، مثل **ls -l** او **ls -L** .

**3** وقد ياتي بالاضافه الى هذا ال **option** ملف (يكون بمثابة الهدف الذي سينفذ عليه الامر) ويسمى **argument**

اذا فعندنا ثلاث صيغ للاوامر التي تدخل الى ال shell

|            |                            |
|------------|----------------------------|
| ls         | الامر وحده                 |
| ls -l      | الامر مع option            |
| ls -L file | الامر مع option و argument |

وطبعا بعد كل صيغه لا بد من الضغط **enter** .

ليس هذا كل شئ ، فال shell او ال **command line** لها قواعد لا بد ان تراعى من هذه القواعد

**المسافات** فبين الامر و ال **option** وبين هذا الاخير وال **argument** لا بد من وضع مسافات (كما هو واضح بالمثال)

**حاله الاحرف** او ما يسمى **case sensitive** ، فالحرف ال **capital** له معنى غير الحرف ال **small** واغلب التعامل مع ال shell بالاحرف **small** ، فالامر (**ls**) هو امر تعرفه ال shell ومن ثم عندما يدخل لها فانها تنفذه ، اما الامر (**LS**) فهي لا تعرفه ولن تتعامل معه.

**الخيارات options** الخيار هو ما يلي الامر كما ذكرنا ، ولا بد ان يسبقه علامه (-) **dash** في اغلب الاحيان .

## اهم المفاتيح للتعامل مع ال shell .

|                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|----------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| space          | به يتم الفصل بين الامر والخيار الذى يتبعه .                                                                                                                                                                                                                                                                                                                                                                                                                         |
| enter          | بعد كتابه الامر هذا المفتاح هو الذى يخبر ال shell انك انتهيت لكى تبدأ فى التعامل مع الامر .                                                                                                                                                                                                                                                                                                                                                                         |
| back space     | عند كتابه امر او خيار خطأ يتم به مسح الحروف الخاطئه .                                                                                                                                                                                                                                                                                                                                                                                                               |
| Tab            | يعد هذا المفتاح من اهم المفاتيح، فهو يظهر خاصيه من اهم خصائص ال shell الا وهى اكمال الامر .<br>مثلا اذا اردت الامر mkdir فما عليك الا كتابه الاحرف mkd ومن ثم ستكمل لك ال shell بقيه الامر اذا ضغطت tab مرتين .<br>اما ان كان هناك اكثر من امر يبدأ ب mkd (مثلا mkdir و mkdev فسيتم عرضهم امامك لكى تختار الامر المناسب .<br>ويلاحظ<br>انه اذا تم الضغط على ال tab بدون كتابه اى احرف فسيعرض عليك النظام عرض كل الاوامر الموجوده به ، (يمكنك تجربته هذه الملاحظه) . |
| Shift+pageup   | عند امتلاء الشاشة امامك وارادت استرجاع الجزء الذى اختفى بالاعلى فيمكنك بهذين المفتاحين رؤيتها.                                                                                                                                                                                                                                                                                                                                                                      |
| shift+pagedown | عكس الذى قبله .                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| up or down     | الاسهم فوق وتحت ، هذه ايضا تظهر خاصيه من الخصائص الهامه لل shell فال up arrow ياتى لك ثانيه بالامر السابق دون الحاجه لاعاده كتابته ثانيه (خاصه عندما يكون الامر طويل) .                                                                                                                                                                                                                                                                                             |
| alt+ctrl+F1    | اذا كنت تعمل على الشاشة ال graphic واردت الانتقال الى ال consol اى الشاشة السوداء، فبهذه المفاتيح الثلاثه يمكن الانتقال اليها .<br>(هناك 6 شاشات consol تبدأ من F1 وحتى F6 ، اما F7 فترجع بك ثانيه الى ال graphic)                                                                                                                                                                                                                                                  |

## ال shell لأول مره .

بدايه وعندما تدخل الى ال shell سوف تجد شكل المحث prompt كالتالى

`[username@localhost userhome] $`

• فمن اليسار - اسم المستخدم الذى دخلت به الى النظام (وعاده يتم تسجيل هذا المستخدم بعد اتمام عملية التنزيل)  
• ثم يليه اسم النظام وهو هنا يسمى localhost ، وهذه التسميه هى ال default عند التنزيل ،  
ويلاحظ انه يفصل بينهما علامه @

• ثم بعد ذلك اسم المجلد الذى يتم العمل فيه الان اى ال working directory ، وهو هنا ال home الخاص بالمستخدم ، ولا يوجد بين اسم النظام localhost وبين مكان العمل اى علامه ، فقط (مسافه white space)  
• وهذه البيانات الثلاثه محصوره داخل قوسين [ ]

• ثم بعد ذلك علامه الدولار \$ والتي تعنى ان المتعامل مع النظام هو المستخدم العادى وليس مدير النظام Administrator

هذه ببساطه اول ما سيقابلك عندما تتعامل مع ال shell  
وبدايه الاوامر التى سنتعلمها هى كيفيه تغير هذه البيانات

فالامر **su** وظيفته تغيير او تبديل المستخدمين ، انت قد دخلت للنظام (login) كمستخدم عادى  
ولكى تقوم باداء مهمه من مهام المدير (Administrator) فلا بد ان تتحول الى المستخدم root  
ويتم هذا التحول عن طريق الامر su  
طبق الامر الان كما هو موضح

`[username@localhost userhome] $ su`

سوف تجد المحث prompt قد ظهر فى السطر الثانى يسالك عن كلمه السر الخاصه بالمدير root (والتي سبق تحديدها اثناء التنزيل)  
وما عليك الا ادخالها ، ويلاحظ انها لن تكتب على الشاشة ، وهذا من اجراءات الامان بحيث لو نظر شخص الى الشاشة فلن يعرف مما تتكون كلمه السر.

\*بعد ذلك ما ستلاحظه ان اسم المستخدم الموجود على اليسار قد تحول الى root ، ولكن ما زال يعمل فى ال home directory الخاص بالمستخدم . ولكى تتحول الى ال home directory  
اى بيت ال root ستضيف بعد الامر su علامه ( - ) اى ال dash ، فيكون شكل الامر -su .  
(طبعا اذا ضغطت على زر ال tab مرتين ستلاحظ ان عدد الاوامر المتاحة الان لل root اكثر من التى كانت متاحه للمستخدم)  
\*ستلاحظ ايضا ان علامه \$ قد تحولت الى علامه # ، دليل ان المستخدم الحالى هو المدير root

ثانى هذه الاوامر هو الامر **hostname** وهذا الامر له استعمالان  
الاول اذا كتبتة منفردا اى hostname فيخبرك باسم النظام (وال default هو localhost.localdomain)

الثاني ، وهو ما نريده ، كتابه الامر يتبعه الاسم الجديد للنظام فمثلا `hostname elnajeeb` فما سيحدث هو ان اسم النظام سيتغير من `localhost` الى `elnajeeb` (ويلاحظ كما هو موضح بالمثال لا بد من كونك `root` <#> لان مدير النظام هو فقط من يملك تغيير اسم النظام)

\*ويلاحظ هنا ان الاسم لن يتغير في الحال (لان ال `shell` لها ملفات تهيئه تقرأها قبل الدخول للنظام) ولهذا لا بد من الخروج من ال `shell` ويتم ذلك بالامر `logout` او ببساطه بضغط `ctrl+d`

ثالث هذه الاوامر هو الامر `cd` ، وهذا الامر اختصار ل `change directory` اي اريد تغيير مكان العمل.

وعندما تدخل للنظام فانه يدخل بك الى ال `home dir` اي مجلد البيت للمستخدم الذي دخلت به (طبعاً الكلام هنا لل `root` لان المستخدم العادي ليس له الا ان يدخل فقط باسمه وبالتالي سيدخل الى منزله هو `user home dir`) ولاي سبب ان اردت تغيير مكان العمل ما عليك الا كتابه الامر يتبعه المكان الذي تريد الذهاب اليه ، مثل

`cd /home` ففي هذا المثال البسيط ستنتقل من `/home/user` الى `/home` اي سنقف على المكان الذي يحوى كل مجلدات المستخدمين، طبعاً هذا الامر هام لل `root` لانه يمكن انجاز مهمه داخل مجلد مستخدم معين ثم الذهاب الى مجلد مستخدم اخر لانجاز مهمه اخرى هناك.

(يمكنك تجربه هذا الامر لاكتشاف النظام ومجلداته المختلفه)

\*وهذا الامر `cd` له رمزان يستخدمان معه هم `{ .. ، ~ }`

فال `double dot` تعنى المجلد الذى يعلو المجلد الحالى (اي التحرك خطوه لاعلى) ويسمى `parent dir` ، مثال `cd ..`

اما العلامه `[~]` فانها تعنى ال `home dir` للمستخدم ،اي المجلد ال `default` الذى يقف عليه المستخدم بعد دخوله للنظام ، مثال على ذلك اذا ذهبت الى مكان بالنظام ووقفت عليه ولنقل مثلاً

`cd /var/mail`

فلكى ترجع ثانيه الى ال `home` الخاص بك فبدلاً من ان تكتب `cd /home/your home` فقط اكتب `cd ~` ستجد نفسك قد رجعت الى مجلدك ال `home` .(يمكن ايضا بكتابه الامر `cd` وحده الذهاب الى ال `home` )

## ملخص

لتغيير المستخدم سواء من مستخدم لمدير، او من مستخدم لمستخدم اخر.  
لعرض اسم النظام او تغييره .  
للتنقل بين مجلدات النظام .  
للخروج من ال shell .

su  
hostname  
cd  
logout

## basic command

بعد ذلك نتعرض للاوامر الاساسيه التى يجب ان تكون ملما بها للتعامل مع النظام

### أولا اوامر التعامل مع الملفات .

#### touch -1

صيغته touch file  
touch [option] file  
هذا الامر خاص بانشاء الملفات ، مثال touch filename سينشأ الملف المسمى filename

\*(الامر touch فى الاساس يغير الوقت الخاص بانشاء الملف ، فان انشأته امس فيمكن تعديله ليظهر انه قد تم انشائه اليوم) مثال touch existfile فان الملف existfile المفترض وجوده سلفا سيتغير تاريخ انشأه الى الوقت الحالى .

#### rm -2

صيغته rm file  
وظيفته إلغاء الملف rm filename سيلغى الملف filename .

#### cat -3

صيغته cat file  
cat file1 file2 file3....  
cat > file

لهذا الامر عده وظائف  
\* فالامر cat يستخدم لقراءة محتوى ملف واحد فقط على الشاشة (يعرض على الشاشة دون ان تتمكن من التغيير فيه) cat filename .

\* ولكن وظيفته الاساسيه هى سلسله الملفات concatenate اى دمج الملف الاول مع الثانى مع الثالث .... واطهار محتوى الثلاثه معا على الشاشة ،  
فلو عندك ثلاث ملفات او اكثر وتريد ان تعرضهم كلهم مره واحده على الشاشة ، دون ان يتأثر محتوى اى واحدا منهم بالآخر اربطهم معا فى سلسله عن طريق الامر cat ، مثال  
cat file1 file2 file3

\* ايضا يستخدم للكتابه داخل الملف وذلك باضافه الرمز > او المسمى redirection ، مثال  
cat > file فتجد المحث prompt قد ظهر فى السطر الثانى ينتظر منك ادخال ما تريد كتابته ، وبعد الكتابه فقط اضغط enter ثم ctrl+d لكى يحفظ ما كتبت داخل الملف ويخرج الى ال shell ثانيه . (يلاحظ ان الملف ان لم يكن موجود من الاساس فبعد هذا المثال السابق ستجد ان

الملف قد أنشأ، اى ان **cat** تنشأ الملفات ايضا بهذه الطريقه)

\* يستخدم ايضا للاضافه فى ملف موجود ومكتوب فيه من قبل ، ولكن فى هذه الحاله سيكون الرمز هكذا >> اى مضاعف ويسمى **append**. مثال  
الملف الذى انشأناه من قليل والمسمى **file** يفترض اننا قد كتبنا به بعض البيانات ، ولكى لا نمسح هذه البيانات نستخدم >> ، لاننا ان جعلناها > فقط فسيتم مسح البيانات الموجوده واحلال البيانات الجديده مكانها .

#### **-4 tac**

كما هو واضح فهو عكس **cat** اى انه يعرض الملف او الملفات ولكن بطريقه عكسيه ، اى من اخره لاوله (يبدأ من اخر سطر)

## ثانياً التعامل مع المجلدات .

### mkdir -1

صيعته mkdir dir  
mkdir -p dir/dir  
mkdir dir1 dir2 dir3

وظيفته إنشاء مجلد directory او مجلدات (بجانب بعضهم او داخل بعضهم) مثال  
mkdir newdir سينشأ المجلد newdir  
اما اذا اردنا إنشاء عدة مجلدات فى نفس الوقت فيكون شكل الامر mkdir new1 new2  
new3 اى نضع بينهم مسافات، وعند استخدام الoption الخيار -p mkdir سينشأ مجلد  
داخل مجلد.... وهكذا ، مثال  
mkdir new1/new2/new3 (لاحظ الفرق فى استخدام المسافات جيداً)

### rm -2

صيعته rmdir dir  
rmdir -p dir/dir  
وظيفته الغاء المجلدات الفارغه (الغير محتويه على ملفات داخلها)  
واذا جاء بالخيار -p معه فيلغى المجلدات التى تحوى داخلها مجلدات ولكن بشرط كون كل  
المجلدات فارغه .

### rm -3

(هو الخاص بالغاء الملفات) ولكنه هنا يلغى المجلدات بشرط اقترانه بالخيارين -r f فىكون  
المثال كالتالى rm -r f dir1/dir2/dir3 (يلاحظ ان الامر rm ك default يلغى الملفات  
فقط ، ولكن عند استخدام الخيارات معه فيمكنه إلغاء المجلدات التى تحوى ملفات ، ومعنى الخيار r  
هو recursive اى ابدأ الالغاء من اسفل لاعلى (يبدأ بالغاء الملفات ثم المجلد الذى يحويهم ثم  
المجلد parent) ومعنى الخيار f هو force أى الغهم بالقوه.

صيغته cp [options] source target

يقوم الامر **copy** بهذه المهمة ، ولا بد من تحديد (من اين ستنسخ والى اين ستنسخ) ، نفترض مثلاً ان فى الhome الخاص بك، ملف و مجلد ، وتريد نسخ الملف لداخل المجلد فيكون المثال كالتالى `/cp myfile mydir` (ويلاحظ ان ال slash / قد تكتب وقد لا تكتب ) هذا بالنسبة للنسخ داخل الhome dir اما بالنسبة للنسخ لاي مكان بالنظام فلا بد من كتابه المسار path سواء من المنسوخ منه او اليه ، فنفترض انك الroot وتريد نسخ الملف 11 من الhome الخاص بالمستخدم user1 الى الhome الخاص بالمستخدم user2 فيكون المثال كالتالى

`cp /home/user1/file /home/user2`

\* والامر **cp** يستخدم العلامتين (النقطه والنقطتين) **dot - double dot** (الdot علامه على الcurrent dir اى المجلد الحالى، اما الdouble dot فعلامه على المجلد الاعلى بدرجه او بمعنى اخر المجلد الاصل للمجلد الحالى) فالامر `mkdir -p parent/child` سينشأ مجلد داخله مجلد ، الان تحرك الى المجلد **parent** بالامر `cd parent` وانشأ داخله ملف بالامر `file touch` ، بعد ذلك تحرك الى المجلد **child** ثم نفذ هذا الامر `cp ../file` . فما سيحدث هو ان الامر **cp** سيذهب الى المجلد **parent** (..) وينقل الملف المسمى **file** الى المجلد **child** (.)

\*الخيارات التى تاتى مع **cp**

- f والذى يعنى **force** اى انقل الملف او المجلد بدون نقاش
- r or -R الاثنين بنفس المعنى اى **recursive** ويستخدم لنقل مجلد داخله مجلدات او ملفات (لانه يبدأ من الاسفل الى الاعلى)
- i وتعنى تفاعل معى وانت تنقل، وستجد الامر يسالك عن كل ملف هل ينقله ام لا(هذا بالنسبه لنقل عدد من الملفات)
- p وتعنى انقل الملف او المجلد مع الاحتفاظ **preserve** بكل بياناته (المالك ، التصاريح ....)

## نقل الملفات والمجلدات .

mv

صيغته mv [options] source destination

بالامر **move** يمكن نقل الملفات او المجلدات ، ويتطلب هذا الامر ايضا (مصدر و مكان ينقل اليه) source and destination ،مثل

mv dir/file /dir2

يستخدم هذا الامر ايضا لاعاده التسميه (ويتم ذلك بدون تحديد مكان ينقل اليه) مثل mv filename newname

\*الخيارات (options)

f - مثل الامر cp

i - مثل الامر cp

## عرض الملفات والمجلدات .

ls

صيغته ls [options]

ls [options] directory

الصيغه الاولى لعرض محتويات المجلد (ملفات و مجلدات) الذى اقف عليه وغالبا يستخدم الخيارين [-a -l] فالخيار L يعنى اعرض بالتفصيل ، اما الخيار a فيعنى اعرض الملفات ال hidden وهى ملفات يتعامل معها النظام ، وتبدأ بـ dot مثل .bash\_profile.

اما الصيغه الثانيه فلعرض محتوى مجلد بعينه مثل ls -al dir

\* الخيارات options

-a تعنى اعرض كل المحتويات بما فيها ملفات النظام (المخفيه)

-L اعرض بصيغه long format

-i اعرض ال inode

dir

مثل الامر ls فانه يعرض الملفات والمجلدات ، ويأخذ نفس خياراته .

## اوامر عامه

### man

صيغته man comand

وظيفته استدعاء صفحات المساعدة المتاحة على الـ system ، مثل man dir فيظهر لك معلومات عن الامر وكيف تستخدمه .

### pwd

وظيفته هذا الامر هو الاخبار عن مكان العمل الحالي او present working directory ، ويستخدم هذا الامر عند الانتقال من مكان لآخر في النظام لعدة مرات ولا تكون على درايه اين انت تماما،

### whoami

بهذا الامر تسأل النظام من انا؟ (في حاله كونك root وتنقلت بين عدة مستخدمين ولا تعرف من انت الان)

### passwd

صيغته كمستخدم passwd  
صيغته كـ root passwd username

لتغيير الـ password  
في الصيغه الاولى اذا كنت تريد تغيير كلمه السر الخاصه بك (وانت user) ، فيكتب الامر منفردا ، فيرد عليك النظام ويسألك عن كلمه السر الحاليه (ليؤكد انك انت المستخدم الفعلي وليس مستخدم اخر)  
في الصيغه الثانيه ، وعندما يريد الـ root تغيير الـ password لاحد المستخدمين فانه يدخل اسم الـ user المراد بعد الامر (اما لو كتب الامر فقط بدون user فيفترض النظام انه يريد تغيير password الخاصه به ، ولانه root فلا يسأله النظام عن الـ password القديمه)

### history

وظيفته عرض كل الاوامر التي ادخلها الـ user الى النظام (وهو يحتفظ باخر 1000 امر ادخل اليه)

clear

وظيفته مسح الشاشة وارجاع ال prompt فقط ، ايضا المفتاحان ctrl+L يقوموا نفس الوظيفة .

date

صيغته date  
صيغته MMDDHHmmCCYY date

وظيفته عرض الساعه واليوم والشهر والسنة  
ويستخدم ايضا لضبط البيانات السابقه(لا بد ان تكون root) وتكون صيغته (شهر-يوم-ساعه-  
دقيقه-سنة- يلاحظ ان كل قيمه لا بد ان تتكون من رقمين) مثل date 100114002004  
فاول رقمين لشهر 10 ثم يوم 1 (لابد من ال 0 قبل 1) ثم الساعه 2 ظهرا والدقيقه 00 والسنة  
2004

cal

صيغته cal  
صيغته cal month year

لعرض التقويم calender ، النتيجة الشهريه .  
الصيغه الاولى لعرض الشهر الحالى ، اما الصيغه الثانيه فلعرض شهر معين بسنة معينه ،  
فلعرض شهر يناير سنة 2000 مثلا نكتب cal 1 2000  
واذا كتبنا cal 2000 فقط فسيعرض كل شهور سنة 2000 .

## ملخص

### لانشاء الملفات

|                          |       |
|--------------------------|-------|
| لانشاء ملف               | touch |
| لalgاء ملف               | rm    |
| لقراءة ملف من اعلى لاسفل | cat   |
| لقراءة ملف من اسفل لاعلى | tac   |

### لانشاء المجلدات

|             |       |
|-------------|-------|
| لانشاء مجلد | mkdir |
| لalgاء مجلد | rmdir |
| لalgاء مجلد | rm    |

### لنقل الملفات والمجلدات

|                  |    |
|------------------|----|
| لنقل ملف او مجلد | mv |
|------------------|----|

### لعرض المحتويات

|                   |     |
|-------------------|-----|
| لعرض محتويات مجلد | ls  |
| لعرض محتويات مجلد | dir |

### اوامر عامه

|                            |         |
|----------------------------|---------|
| استدعاء صفحه المساعده      | man     |
| عرض مكان العمل الحالى      | pwd     |
| الاخبار من المستخدم الحالى | whoami  |
| لتغيير كلمه السر           | passwd  |
| لعرض الاوامر السابقه       | history |
| لمسح الشاشة                | clear   |



## linux documentation

فى هذا الفصل سنتعرف على

**on line help** المساعدة الفوريه اثناء العمل مع النظام  
المستندات والوثائق الموجوده بالنظام  
المستندات الموجوده على الانترنت

### أولا المساعدة الفوريه **on line help**

المساعدة الفوريه هى التى يلجأ اليها المستخدم اثناء العمل مع النظام ، وسميت فوريه لان المستخدم لا يحتاج للرجوع الى كتب خاصه يملكها ولكنه يستدعيها وهو جالس على الجهاز ، بمعنى اخر هى مساعده موجوده اساسا فى النظام تنزل تلقائيا مع الاوامر و البرامج التى يتكون منها النظام .

والسبب فى وجود مثل هذه المساعدة هو كما رأينا ان كل امر له خيارات **options** عديده قد تصل الى العشرات الامر الذى يصعب معه بالتاكيد حفظ كل هذه الخيارات . هذا بالاضافه الى عدد الاوامر الكبير ايضا .

و بالنظام نوعين من المساعدة الاولى هى صفحات ال **manual**  
والثانيه هى صفحات ال **information**

وصفحات ال **man** هى فى الاساس من نظام اليونكس UNIX .  
اما صفحات ال **info** فهى من انشاء مشروع GNU .

وتعدد الانواع هنا سببه ان مشروع GNU يهدف منذ نشأته الى بناء انظمه تشغيل(معتمده على يونكس) كامله بذاتها لا تعتمد على انظمه اخرى فكان لا بد لهذه الانظمه من صفحات مساعده مستقله (ويظهر ذلك ليس فقط فى صفحات المساعدة ولكن ايضا فى برامج وادوات تهيئه وغيرها)

اما سبب ادماج صفحات ال **man** هنا مع وجود بديل له وهو (info) فهو ،  
- اثرء نظام التشغيل بادوات وبرامج مستقره بسبب تطويرها على مدار سنوات .  
- وايضا لا ننسى ان كل المشاركين فى مشروع GNU هم فى الاساس مطوروا UNIX القدامى ، وهذه الادوات (كالمساعدة وغيرها) قد ألفوها وتعاملوا معها على مدار سنوات .

ومع ان النوعين يوفران مساعده قد تكون متشابهه فى الشكل والمضمون الا إننا احيانا نجد اختلاف او اتفاق بين النوعين ، وهذا ما سنذكره

### أوجه الاختلاف بين النوعين .

\* صفحات ال **man** مكونه من صفحه واحده ويتم الانتقال فيها لاعلى واسفل فقط ، اما صفحات ال **info** فهى متشعبه يتم الانتقال فيها لاعلى واسفل (داخل الصفحه) وللأمام والخلف (الصفحات السابقه والاحقه) .

\* بالنسبة للاوامر والبرامج التى انتجها مشروع GNU فغالبا المساعدة المتوفرة لها فى صيغه ال info فقط دون ال man ، فقد حاول البحث عن امر معين فى ال man ولا تجده وعندما تبحث عليه فى ال info تجده موجود .

\* الشرح الذى تقدمه صفحات ال info قد يكون اكثر تفصيلا من صفحات ال man .(وفى بعض الاحيان تجده نسخه من ال man ) .

.صفحات ال man لها ترقيم تصنف به الاوامر ، فالرقم 5 مثلا خاص بملفات تهيئه النظام و8 خاص باوامر اداره النظام ، ومن ثم فانت تعرف مع اى نوعيه من صفحات المساعدة تتعامل ، (ملفات ,اوامر , برامج) اما مع صفحات ال info فالامر ليس كذلك .

اما بالنسبه لوجه الاتفاق بينهم فعيده منها

\* طريقه استدعاء المساعدة والخروج منها واحده ، فلاستدعاء ال info تكتب info command ولصفحات ال man تكتب man command ولكى تخرج من النوعين تكتب حرف q .

\* للغالبية العظمى من الاوامر صفحات مساعده عند ال man وعند info (اى ان للامر صفحتان واحده عند ال man والاخرى عند ال info) والاقليه كما اسلفنا موجوده فقط عند ال info .  
\* ان كانت طريقه التنقل داخل الصفحتين مختلفتان نوعا ما ، الا ان شكل الصفحتين متفقتان من حيث الشكل والترتيب (فاذا كنت معتاد على قراءة صفحات ال man فقط واردت قراءة صفحات ال Info فلن تجد اى مشكله على الاطلاق)

## man page

صفحات ال man هي الاقدم في ال unix-like os ، فقد كانت هي وسيله المساعدة الفوريه التي كان يتيحها UNIX .

وال man page (وايضا ال info ) ليست وسيله تعليميه تتعلم منها النظام ، فهي موجوده لكي تذكرك بكيفيه استخدام الامر وبالخيارات المتوفره معه مع شرح موجز لنتائج كل هذا ، بالاضافه الى توجيهك الى الاوامر المتعلقة بنفس الموضوع ، اى انها تفترض انك من الاساس تعرف كيف تتعامل مع النظام .  
بمعنى اخر لا تحاول فتحها وان تعتقد انك ستتعلم لينوكس من خلالها ، لان هذه وظيفه وثائق documentation اخرى كما سنرى .

وال man page مكتوبه بطريقه مختصره جدا ، والهدف من ذلك جمع اكبر قدر ممكن من المعلومات في اقل مساحه ممكنه (لاحظ الخيارات وما يليها من شرح موجز لدورها) ، ( فبعض الاوامر وخياراتها تحتاج في بعض الاحيان لشرح اكبر بكثير مما هو موجود بال man page )

\* شكل ال man page

نفذ هذا الامر man ls ولاحظ شكل صفحه ال man

```
LS(1) 1                                FSF                                LS(1)
NAME 2
ls - list directory contents

SYNOPSIS 3
ls [OPTION]... [FILE]...

DESCRIPTION 4
List information about the FILES (the current directory by default).
Sort entries alphabetically if none of -cftuSUX nor --sort.

Mandatory arguments to long options are mandatory for short options
too.

-a, --all
    do not hide entries starting with .

-A, --almost-all
    do not list implied . and ..

--author
    print the author of each file

:
```

فنجذ الناتج من الامر

1 نجد اسم الامر يتبعه رقم ، وهذا الرقم يدل على نوع من الاوامر تتعامل

- فرقم 1 هي الاوامر التقليديه التى يمكن لاي مستخدم تنفيذها user command
- 2 الاوامر الخاصه بالكرنل kernel function (system call)
- 3 الاوامر الخاصه بالمكتبات library function
- 4 الاوامر الخاصه بالdevices مثل الهارد مثلا
- 5 الاوامر الخاصه بملفات التهيئه configuration files
- 6 الاوامر الخاصه بال games
- 7 اوامر متفرقه miscellaneous
- 8 الاوامر الخاصه باداره النظام Adminstration comand

ولكن المهم معرفته من هؤلاء الارقام هم 1 ، 2 ، 5 ، 8

وهكذا ، فالمثال الذى امامنا الان للامر ls يبين لنا ان هذا الامر متاح للجميع سواء كانوا مستخدمين او مديرين لانه مرقم برقم 1 ، فالمستخدم يمكنه تنفيذ هذا الامر على المجلدات التى يملك حق استعراضها .

والاوامر التى تحت القسم (2) هي الاوامر التى تستدعى النظام ليقوم بعملية معينه مثل الامر kill .

اما القسم (5) فهو يشرح لنا ملفات النظام و كيفية التعامل معها، من حيث تهيئتها وما هي وظائفها ونحو ذلك .

واخيرا القسم (8) الخاص بالاوامر المتعلقة بالمدير root اى التى يسمح له فقط بتنفيذها مثل الامر fdisk على سبيل المثال .

فهؤلاء الاقسام الاربعه هم فقط ما تحتاج معرفتهم الان .

ولكن قد يكون للامر الواحد اكثر من رقم (اى موجود باكثر من قسم) مثل الامر kill فهو موجود بالقسم 1 وايضا بالقسم 2 ، وهذا يعنى ان هذا الامر يستطيع المستخدم استدعاؤه وتنفيذه مباشره بنفسه (ارسال signal الى الprocess المعينه) وهو فى ذات الوقت وظيفه من وظائف النظام (يوكل المستخدم النظام فى ارسال هذه الsignal)

هذا من ناحيه التقسيم ولكن هناك مسأله اخرى مهمه الا وهى كيفية استدعاء ال man page الخاصه بكلا منهما ، ذلك لانه عند تنفيذ الامر man kill فان الامر man يذهب ويبحث فى قاعده بياناته ويأتى باول نتيجة تصادفه ، اى انه دائما سيأتى لك ب 1 kill لانه اول ما يجده ، ولتفادى هذه العقبه فلا بد من تحديد الامر بذاته لكى يتجنب الخلط وذلك عن طريق رقمه، فتكتب له kill 2 man (يجب مراعاة المسافات) حينئذ سيعود لك بالنتيجة التى تريدها .

## NAME 2

في هذا ال section نجد بسهولة اسم الامر تتبعه وظيفته ، فالامر ls وظيفته عرض محتويات المجلد/المجلدات ، ولانه يصف الامر بتلخيص فاننا نجده مكون من سطر واحد ، ويمكنك اذا اردت فقط معرفه وظيفه الامر (اي استعراض هذا السطر) دون اراده الدخول لل man page الخاص بهذا الامر ، ان تستخدم الامر **what is** يتبعه الامر، ايضا الامر **man -f** يقوم بنفس هذه الوظيفة .

## SYNOPSIS 3

هذا ال section يبين لنا كيف نستخدم الامر (صيغه الامر حين تطبيقه) فنجد انه مكتوب

`ls [option] ... [file]...`

فالامر يكتب اولاً ، ثم يتبعه ال option سواء كان واحداً او اكثر ، ثم بعد ذلك ال argument او الملف (غالبا) الذي يطبق عليه الامر.

### ملاحظات هامه

لفهم صيغه كتابه الاوامر اهميه كبرى في تعاملك مع النظام . ولهذا لا بد ان تضع في اعتبارك عدة نقاط هامه وهي

\*1 عندما تجد الامر قائما بذاته (اي ليس داخل اقواس او ما شابه) يكتب كما هو موضح ، وكل نص قائم بذاته يجب كتابته كما هو ، مثل الاوامر كلها (فهى تكتب كما هى) .

\*2 القوسان [ ] يدلان على ان ما ياتي بينهما هو امر اختياري ، قد تكتبهم وقد لا تفعل ، ونرى في المثال انه كتب بينهما options فيمكن ان يكون الامر هكذا `ls -la` او هكذا `ls` فقط او حتى هكذا `ls -l dir` (وهذا وفق احتياجاتك من الامر) وسينجح الامر في كل الحالات ، ذلك لان ال option وال file امر خيارى وليس اجبارى كتابتهما .

\*3 القوسان < > يدلان على انه لا بد من كتابه ما يقع بينهما مع ابداله بالنص المناسب ، فلو كان المثال يقول `< file >` فما عليك الا ابدال الكلمه `file` بالملف الذى ستعامل معه .

\*4 القوسان { } يدلان على انك لا بد ان تكتب على الاقل واحد (او اكثر من واحد) ما يقع بينهما (تختار ما يناسبك)، مثال لو افترضنا ان الامر `ls` مكتوب بالصيغه الاتيه `ls {acdfgh}` فمعنى هذا ان الامر `ls` لن يعمل الا اذا اخذ احد هؤلاء ال options ، قد يفصل بين ال options بعلامه | piping .

\*5 علامه الثلاث نقاط (...) دليل على كلمه (وهكذا او الخ) ففي مثال `ls` نجد مكتوبا `... [option]` اي خيار واحد او اكثر ، وكذلك مع `... [file]` اي ملف واحد او اكثر .

\*6 قد يتم دمج الاقواس مع بعضهم ، فقد ترى مثلاً `ls [< file>]` اي انك مخير بين ادخال القيمه التى بين < > او لا تدخلها .

(يلاحظ ان الرموز المستخدمه هناهى قواعد يتبعها كل من يتعامل مع لينوكس فى شرح مراده ، سواء كان هذا المتعامل مولفا لكتاب او كاتبا لman page او غيره ، اى انها قواعد توقع ان تجدها باكثر من مكان فى مجتمع اللينوكس)

#### DESCRIPTION 4

فى هذا ال section نجد شرح للامر وما يقوم به (وطبعا بإيجاز) فى عده سطور ، وفى مثال ال ls الموضح باعلى نجده يقول  
( عرض معلومات عن الملف او الملفات ويفتح قوس ويقول انه اذا استخدم منفردا سيعرض المجلد الحالى current dir ، ثم يكمل ، والامر يصنف الذى سيعرض بطريقه ابجديه اذا لم يستخدم اى من هذه الخيارات ثم يورد الخيارات )  
ثم بعد ذلك يبدأ فى عرض الخيارات مع شرح مبسط لكل خيار .

\*ونلاحظ ان الخيارات options اما ان تكون حرف واحد مسبوق ب one dash او عباره عن كلمه مسبوقه ب double dash .  
ويرجع هذا الاختلاف الى ان ال UNIX فى الاساس كان يستخدم الخيار المكون من حرف واحد ، وعندما تاسس مشروع GNU ادخلوا على الانظمه التى ينتجونها الخيارات المميزه لهم وهى ذات الكلمه وال double dash ولذلك ستجد ان الامر ls -a يخرج نفس نتيجه الامر ls --all .

\*يلاحظ ايضا ان الخيارات options اما ان تكون capital او small (بعكس الاوامر فغالبيتها العظمى small)

\* يلاحظ ايضا ان خيارات ال GNU ذات الكلمه و ال double dot قد ياتى بعدها WORD= (علامه يساوى وكلمه capital word ) وهذا يعنى انك قد تكتب علامه = وتدخل بعدها قيمه التى تريدها مكان كلمه WORD .

#### AUTHOR 5

ثم بعد ذلك تجد فى هذا ال section اسم المبرمج الذى كتب هذا البرنامج والman page ، وقد يكتب بريده الاليكترونى لتسهيل عمليه الاتصال به.(بالنسبه للامر ls نجد ان كاتبه هوريتشارد ستولمن مؤسس مشروع GNU)

#### BUGS 6

فى هذا ال section نجد بريد اليكترونى يرسل عليه على سبيل المثال المشاكل التى واجهتك اثناء تنفيذ البرنامج .

#### COPYRIGHT 7

يذكر به معلومات حقوق الملكيه لهذا البرنامج .

## 8 SEE ALSO

يعد هذا ال section من افيد التقسيمات الموجوده بصفحه ال man page اذ انه يدلك على الاوامر المتعلقة بنفس الموضوع او على الاقل التى بينها وبين هذا الامر علاقه من نوع ما . فنرى بذلك انه تكفى معرفه امر واحد ثم بعد الاطلاع على صفحه المساعدة الخاصه به ستخرج منها بطرف خيط يقوم بعمل المرشد لك (يلاحظ ايضا ان بعض الاوامر ليس لها علاقه مع اوامر اخرى)

## 9 FILES

نجد هذا ال section فى الاوامر التى لها ملفات تهيئه ، انظر مثلا man mount .

## 10 HISTORY

يذكر هذا ال section تاريخ الامر اذا كان يستعمل مثلا فى انظمه تشغيل اخرى مثل UNIX او BSD.

## مفاتيح التعامل مع صفحات الman

لا بد وانت تتعامل مع الman page من معرفه المفاتيح التى تتيح لك التفاعل معها .  
وهذه المفاتيح هى

up & down arrow لاستعراض الصفحه لاعلى واسفل ، ايضا يقوم بهذا الدور المفاتيح f  
space (اسفل) وال b (اعلى)

|                                                                 |           |
|-----------------------------------------------------------------|-----------|
| للخروج من الصفحه                                                | Q         |
| للبحث داخل صفحه الman عن كلمه معينه من اعلى لاسفل               | /         |
| ~~~~~ من اسفل لاعلى                                             | ?         |
| للنزول مباشره الى اخرالصفحه(اذا اردت الذهاب مباشره الى النهايه) | shift + g |
| فقط ، تصعد بك الى قمه الصفحه(اذا كنت فى منتصفها او اخرها)       | g         |

(يلاحظ انه مع كثره تعاملك وتمرسك بالنظام يمكنك ان تعتمد نسبيا على صفحات الman page  
او حتى الinfo page فى فهم الاوامر التى تتعامل معها لاول مره ، وان كانت القاعده ان هذه  
الصفحات للمساعده الفوريه ، الا ان كثره اطلاعك عليها وفهم لغتها المختصره جدا يكسبك خبره  
فى فهم الاوامر الجديده)

```
File: info.info, Node: Getting Started, Next: Expert Info, Prev: Top, Up: Top
Getting Started
*****

This first part of the Info manual describes how to get around inside
of Info. The second part of the manual describes various advanced Info
commands, and how to write an Info as distinct from a Texinfo file.
The third part briefly explains how to generate Info files from Texinfo
files.

* Menu:

* Help-Small-Screen:: Starting Info on a Small Screen
* Help:: How to use Info
* Help-P:: Returning to the Previous node
* Help-^L:: The Space, DEL, B and ^L commands.
* Help-M:: Menus
* Help-Xref:: Following cross-references
* Help-Int:: Some intermediate Info commands
* Help-Q:: Quitting Info

--zz-Info: (info.info.gz)Getting Started, 23 lines --All-----
```

لاستدعاء معلومات عن ال info page نفذ الامر

info info

وكما هو ملاحظ ان اسم الصفحة التى تقف عليها الان تسمى node

واسم الصفحة اللاحقه يوجد بجانب كلمه Next

والصفحه السابقه اسمها بجانب Prev

ال info page هي النوع الاخر من المساعدة الفوريه التي يتيحها النظام ، ولصفحات ال info خصائص تفوق تلك المتوفره لل man page ، باعتبار انها الاحدث من هذه الاخيريه .

من هذه الخصائص امكانيه طبعتها ، ايضا طبيعته بنائها فهي ذات طبيعته تشعبيه ، بمعنى انك تستطيع التنقل داخلها ذهابا وإيابا .

طريقه استدعائها هي نفس طريقه استدعاء ال man page ، ولكن الاختلاف في نتيجته هذا الاستدعاء ، فعندما لا يجد البرنامج man الامر الذي يبحث عنه فانه يجيبك انه -no manual entry for this command- ، اما ال info فعندما لا تجد الامر الذي تبحث عنه فانها تفتح لك الصفحه الرئيسيه لها .

وفي اعلى صفحه ال info سطر يحدد لك على اى امر او عنوان تقف ويسمى node ، ايضا يخبرك ما العنوان الذي يسبق هذه الصفحه وما الذي يليها اى next و prev ( رأسيا ) ، وما الذي يعلوها ( افقيا ) .

### مفاتيح التعامل مع ال info page

التنقل في الصفحه

|                |                                  |
|----------------|----------------------------------|
| e              | للذهاب لنهايه الصفحه end مباشره  |
| space          | للذهاب لآخر الصفحه خطوه خطوه     |
| b              | للرجوع لاول الصفحه begain مباشره |
| del او b-space | للرجوع لاول الصفحه خطوه خطوه     |

التنقل بين الصفحات

|     |                                                                                                             |
|-----|-------------------------------------------------------------------------------------------------------------|
| (*) | هذه العلامه في اول السطر دليل على وجود صفحه جديده ، تحرك بالاسهم وقف عليها واضغط enter للدخول لهذه الصفحه . |
| N   | للتحرك للامام ( رأسيا ) الصفحه المقبله                                                                      |
| P   | للتحرك للخلف ~~~ الصفحه السابقيه                                                                            |
| U   | للتحرك لاعلى ( افقيا )                                                                                      |
| D   | عند التعمق داخل الصفحات فهذا المفتاح يرجعك الى الصفحه الرئيسيه لل info                                      |

للبحث داخل الصفحه

|   |                                            |
|---|--------------------------------------------|
| / | بحث من اعلى لاسفل ، مثل صفحات ال man       |
| s | للبحث بدون مراعاة حاله الحروف              |
| S | للبحث مع مراعاة حاله الحروف case-sensitive |
| Q | للخروج من ال info                          |

حاول تجربه المفتاح ؟ ولسوف تجد عدد كبير جدا من المفاتيح التي يمكنك التعامل مع ال info بها.

ايضا نفذ الامر info info لكي ترى معلومات مفصله عن ال info وكيفيه التعامل مع صفحاتها.

## النوع الثانى من المساعدة

الوثائق الموجودة مع النظام

هذا النوع من المساعدة المقصود به الوثائق التى تاتى مع الاوامر والبرامج ، وهذ النوع من المساعدة هو الذى يمكن الاعتماد عليه فى تعلم اللينوكس او النظام .

والمكان الرئيسى لهذه الوثائق هو المسار `/usr/share/doc/xxxxx`

فيمكنك الان استعراض الكتيبات الموجوده (اذا كنت تتعامل مع النظام من خلال ال GUI فلاحظ انه عن فتح مستعرض الانترنت -mozilla مثلا- ستجد ان الصفحة ال default له هى مكان ال documentantion الاساسيه بالنظام وهو المسار السابق ذكره) .

اما ان كنت تتعامل مع النظام من خلال ال console او ال terminal ، مثل ما افعل ، فيمكنك استعراضها بعده اوامر مثل ( cat او less او more او حتى vi ) فقط نفذ الامر التالى  
`cat /usr/share/doc`  
وطبعا ستضغط double tab لى تجعل النظام يعرض عليك سرد كل ما يحويه هذا المجلد امامك لتختار منها ،  
اما اذا كنت تعلم ما الذى تريده بالضبط فما عليك الا كتابه الاسم كله او على الاقل جزء منه مثل bas لل bash وذلك عند اراده البحث عن معلومات او كتيبات عن ال bash.

## ما الذى تقرأه بالضبط

تتكون الوثائق من عدة ملفات ويتحدد ما الذى ستقرأه على حسب ما الذى تريده .

فعندما تحاول تنزيل برنامج جديد program package للنظام او حتى تحديثه up grade ، فالمناسب هنا هو قراءة ملف INSTALL ، الذى يشرح لك كيفيه انجاز مثل هذه العمليه .

اما ان كنت تبحث عن كاتب هذا البرنامج وبريده الاليكترونى فملف AUTHORS هو الانسب فى هذه الحاله.

ايضا من الملفات الهامه ملف README الذى يشرح لك ما هذه ال package وما الذى تؤديه بالاضافه الى نبذه مفيده عنها.

FAQ او كما هو معروف ، اختصار ل frequently asked questions ايضا هو من الملفات المفيده والذى يجيب عن الاسئله التى من المفترض انها ستدور فى ذهن المستخدم الجديد . وهناك ملفات اخرى ذات قيمه ، حاول استعراضها لى تتعرف اكثر واكثر على هذا النوع من المساعدة .

## النوع الثالث من المساعدة.

الوثائق والمستندات على الانترنت

هذا القسم من الوثائق هو المتعلق بالمساعدة الموجوده على شبكة الانترنت ، سواء كانت هذه الوثائق فى شكل مجموعات الاخبار او ال mailing list او المجلات المتعلقة ب لينوكس او غيرها .

فيمكننا تصنيف المساعدة الموجوده على الانترنت الى نوعين

الاول هو المتعلق بالوثائق المكتوبه او ال documentation .  
الثانى وهو المتعلق بالمساعدة المتخصصة.

## النوع الاول

وهذا النوع من الوثائق doc له عدة صور

الكتب بشكلها المعروف (guides) فهناك العديد والعديد من الكتب التى تشرح لينوكس ، بدايه من الكتب التى تشرح مواضيع لينوكس المختلفه، الى الكتب التى تركز على موضوعات بعينها اما لاهميتها او بسبب تخصص كاتبها فيها، مثل الامان security .

واذا حاولت البحث عن لينوكس فى احد محركات البحث (search engine) مثل google او altavista على سبيل المثال فستجد كم هائل من الكتب التى تتحدث عن لينوكس ، ولكن لتوفير الوقت والجهد فهناك عدة مواقع (sites) هدفها جمع الوثائق (doc) المهمه والمفيده للمهتمين ب لينوكس، بدايه من المستخدم العادى الى المستخدم المتخصص ، من هذه المواقع

LDP اختصارا ل the Linux Documentation Project وموقعه على الانترنت هو  
<http://www.tldp.org>

يعد هذا الموقع من اهم المواقع التى توفر ال doc لمستخدم لينوكس ، اما بالنسبه لانواع ال doc التى توفرها فهى

### **1- the guides**

اى كما اسلفنا الكتب التى تقوم بشرح الموضوعات المختلفه فى لينوكس.

### **2- HOWTO**

هذا النوع من ال doc يركز على مهمه واحده معينه ويشرحها خطوه بخطوه حتى يتم انجاز هذه المهمه ، مثلا ان وجدت وثيقه تسمى linux installation HOWTO اذا فهى تركز على مهمه تنزيل النظام من الالف الى الياء .

ملاحظه

(عالم لينوكس غنى بهذا النوع من الوثائق، فستجد العشرات والعشرات من هذه النوعيه، وطبعاً ليس مطلوب منك ان تقرأها كلها، ولكن ثق، فعندما تحتاج الى مساعده لاداء مهمه معينه، فحينئذ هذا هو دور ال HOWTO)

وهذا النوع من الوثائق (HOWTO) له شكلان

الاول هو ال HOWTO فقط ، اى normal HOWTO  
اما الثانى فهو ال mini HOWTO ويتميز عن الاول بصغر حجمه .  
(ولكن يلاحظ انه بعد تاريخ 24/7/2003 قد تم دمج النوع الثانى mini فى الاول normal  
 واصبح هناك صيغه واحده للثتان هى HOWTO فقط)

وال HOWTO متوافره فى صيغ كثيره فتجدها pdf او plain text او html .  
اما بالنسبه لعدد صفحاتها فهو متباين

فقد تجد وثيقه 4 صفحات مثل linux+win+grub HOWTO  
وقد تجدها 23 صفحه مثل installation HOWTO  
اما بالنسبه ل hardware HOWTO فانك تجدها 183 صفحه – يتحدث كاتبها Steven  
Pritchard و Patrick Reijnen عن كل انواع ال devices، التى تتوافق والتى لا تتوافق مع  
لينوكس ، بدايه من المعالجات التى تخدم السيرفرات الى كروت الصوت والمودم.

ملحوظه  
(قد تنزل ال HOWTO مع ال documentation التى تنزل بالنظام ك default ،  
هذا اذا كانت النسخه التى لديك من لينوكس تدعم هذا النوع من ال doc، حينئذ سيكون مكانها هو  
</usr/share/doc/HOWTO> ) .

اما فى حاله عدم وجودها فى هذا المكان فيمكنك تنزيلها بنفسك يدويا ، وذلك بانشاء مجلد يسمى  
howto ونسخ هذه ال doc بداخله .

### 3- man page

يوفر LDP ايضا صفحات المساعده man page والتى يمكنك ان تنزلها download من  
الموقع وتفق ضغطها ثم تنزلها install على نظامك (هذا ان لم تكن قد نزلت install مع النظام ك  
default كما هى العاده ، او اذا اردت ان تنزلها بلغه اخرى غير الانجليزيه - فانها تتوافر فى 8  
لغات بجانب الانجليزيه)

### 4- FAQ او Frequently Asked Questions

قد تجد فى هذا الموقع عدد لا باس به من ال faq والتى تشرح موضوع معين عن طريق طرح  
الاسئله البديهيه حول هذا الموضوع ، فتجدها عباره عن سؤال واجابته ، وذلك ليمهد وايضا ليجيب  
على الاسئله التى تدور فى ذهن المستخدم الجديد عن موضوع بعينه ، فمثلا linux faq يسال  
اسئله مثل ما هو لينوكس ، هل لينوكس هو يونكس ، كيف ينطق لينوكس وهكذا ثم يجيب عنها.

### 5- linux focus و linux gazette

يوفر LPD ايضا مجلتان online عن لينوكس واسرارها، هما لينوكس فوكس و لينوكس جازيت .

وهناك ايضا نوع اخر من ال doc ولكنه ليس موجود فى موقع LDP، هذا النوع هو

## RFC -6

الذى يعد اختصار ل request for comments او (طلب للتعقيب عليه)، وهذا النوع من ال doc يعد (معياري على شبكة الانترنت) internet standard الهدف منه تعريف كل شى عن اتصالات الانترنت او عن موضوع بعينه، بمعنى اخر اذا وجدت مثلاً فى صفحات ال man هذه الجملة (RFC1290) فهذا يعنى الاتى ( لكى تكون ملماً بتفاصيل هذا الموضوع فيستحسن الرجوع لهذه الوثيقة ذات الرقم الفلانى لكى تقف على التفاصيل الدقيقة لهذا الموضوع )

وكما اسلفت فان موقع LDP ليس به هذا النوع من الوثائق ولكنها توجد فى موقع

<ftp://metalab.unc.edu/pub/docs/rfc>

**الثانى** هو المتعلق بالمساعدة المتخصصة .  
ولتوضيح هذا النوع اكثر، نقول هو المساعدة التى تتم عن طريق قوائم البريد mailing list وعن طريق مجموعات الاخبار news groups .

ولكن بدايه ما هى قوائم البريد ومجموعات الاخبار.

**قوائم البريد**، هى ببساطه قائمه (طويله او قصيره) من عناوين البريد الالكترونى، لها عنوان رئيسى وعندما يرسل اى شخص رساله الى العنوان الرئيسى لهذه القائمه فان هذه الرساله تصل الى كل عناوين الاشخاص التى تحويها هذه القائمه.

وقد كانت هذه الطريقه فيما مضى من اهم مميزات الانترنت والطريقه الوحيديه للحصول على المعلومات، ولكنها الان قد استبدلت بطريقه جديده فى نقل الاخبار الا وهى مجموعات الاخبار (ما زالت قوائم البريد هذه قيد الاستخدام ولكنها فى نطاق ضيق نظراً لانها تتعامل مع مجموعات مغلقه)

**مجموعات الاخبار** هى الطريقه الاحداث فى نقل الاخبار والمعلومات، فيمكن اعتبارها كلوحه اعلانات عالميه .

وتعمل هذه المجموعات بطريقه مختلفه عن قوائم البريد، ففى قوائم البريد ما عليك الا فتح ال mail الخاص بك واستعراض الرسائل التى ارسلت الى المجموعه (القائمه) الموجود بها عنوانك البريدى اما بالنسبه لمجموعات الاخبار news groups فانك تقوم بزياره المكان الذى به هذه الرسائل والذى يسمى news server (خادم الاخبار).

هذا من ناحيتك انت كقارى لهذه الاخبار ولكن كيف تسير الامور قبل قراءتك للرسائل؟

اولاً مرسل الرساله ما عليه الا ارسالها الى ال news server ، وهذا الاخير يقوم بنشرها على بقية ال news servers المنتشرين حول العالم والمشاركين معه فى خدمه نفس المجموعه الاخباريه، اى انه هناك مجموعه اخبار news group واحده ، والعديد من ال news servers منتشره حول العالم.

ولكن لماذا هذا التعدد من جهة خوادم الاخبار news servers ؟

الامر ببساطه هو لتقليل الحركه traffic الغير ضروريه على الانترنت والتي بدورها ستسبب فى حدوث high traffic او ازدحام شديد ليس له داع على شبكه الانترنت .  
والمفترض ان كل ISP (موفر خدمات للانترنت) يجب عليه توفير هذا ال news server ولكن طبعا لان ال news server وظيفته الرئيسيه هى تخزين وحفظ الرسائل لكى يستطيع القارى قراتها فيما بعد ، فهو من وجهه نظر ال ISP (وهى شركات تجاريه فى المقام الاول) تكلفه فى الامكان الاستغناء عنها . ولكن مع ذلك هناك العديد من ال ISP التى توفر خادم للاخبار.

كيف تستطيع قراءة هذه news groups ؟

يتم قراءة هذه الرسائل بعده طرق ،  
فاولا ان كان (موفر خدمات الانترنت) الذى تتعامل معه يوفر خدمه ال news groups فيمكنك الاتصال با news server الخاص به ومن ثم استعراض الرسائل التى يحتفظ بها .

ثانيا -وهى الطريقه الاسهل- استخدام خدمه قراءة الرسائل التى توفرها مواقع مثل google على العنوان التالى

<http://groups.google.com>

ايضا التى يوفرها yahoo على العنوان <http://groups.yahoo.com>

ثالثا عن طريق ال web browser مثل mozilla او netscape ،  
فكلاهما يوفران news & mail reader application

#### ملاحظه

(يلاحظ انه فى العالم الان اكثر من 16الف مجموعه اخباريه تغطى كل نواحى الحياه التى تتصورها والتى ايضا لا تتصورها .  
فهى شبكه عملاقه جدا يعدها الخبراء مرادفه للانترنت وتسير بمحازاته ، ويطلق عليها ايضا المصطلح USENET .)



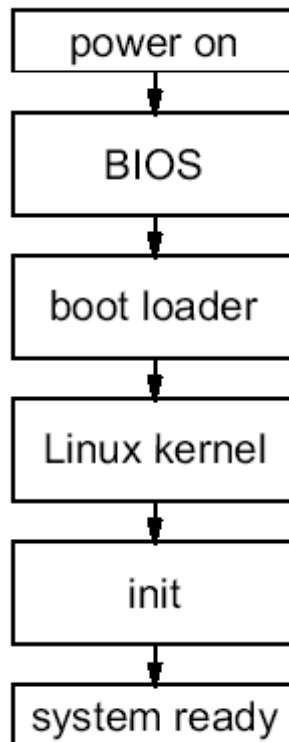
## عمليات ال shutdown و startup .

في هذا الفصل سنتعرض لمسالة تشغيل النظام واغلاقه .  
فمسالة تشغيل النظام تشمل

- . ال Bios وما يجريه من عمليات
- . ال boot loader
- . تحميل الكرنل في ال Ram

ثم بعد ذلك يصبح النظام مستعد للعمل

وقد يصور الشكل التالي هذه العمليات



## الخطوة الاولى : ال Bios .

ال Bios هذا عبارته عن برنامج مخزن في ذاكره Rom ووظيفته القيام بوظائف بسيطه ومحدده . ومن هذه الوظائف

- . فحص الذاكرة العشوائيه (Ram)
- . اداء عدد من الوظائف والخيارات ، على سبيل المثال ، تحديد وترتيب ال device الذى سيبدأ منه نظام التشغيل عمله او ما يسمى the order of boot device ، وغيرها من الوظائف والتي يمكن تعديلها عن طريق الضغط على المفتاح Del اثناء عمليه اختبار الذاكرة .
- . اختبار ال device الذى سيبدأ منه نظام التشغيل ، مثل ال hard disk او ال cdrom .
- . تحميل البرنامج الموجود في ال MBR .

وبهذه الخطوات يكون ال Bios قد ادى وظائفه ، واعطى البرنامج الموجود في ال MBR اشارته البدء لكي يقوم هذا الاخير باعماله .

## **ما هو ال MBR .**

- . اول ال mbr اختصارا ل Master Boot Record .
- . وهو عبارته عن اول sector موجود في ال hard disk ، وال sector كما هو معروف عبارته عن 512byte (لاحظ انه بالبايت وليس بالكيلو بايت) اي انه جزء صغير جدا .
- . وهذا ال mbr يوجد قبل كل ال partition الموجوده على ال hard disk ، بمعنى انه ليس جزء من اول partition .
- . ويطلق عليه ايضا ال boot sector .

ويوجد بهذا ال mbr شيئان  
اولا ، برنامج ال boot loader وهو اما lilo (وهو الاقدم) او grub (وهو الاحدث والمستخدم حاليا)  
ثانيا ، ما يسمى بال partition table ، ويوجد به معلومات عن ال hard disk .

## الخطوة الثانية : ال boot loader .

وهذا ال boot loader عبارته عن برنامج ، وظيفته الاساسيه هي استلام البيانات والمعلومات من ال Bios ، والبحث عن نظام التشغيل ، وتحميل هذا الاخير فى ال Ram كخطوة اولى لبدايه تشغيله .

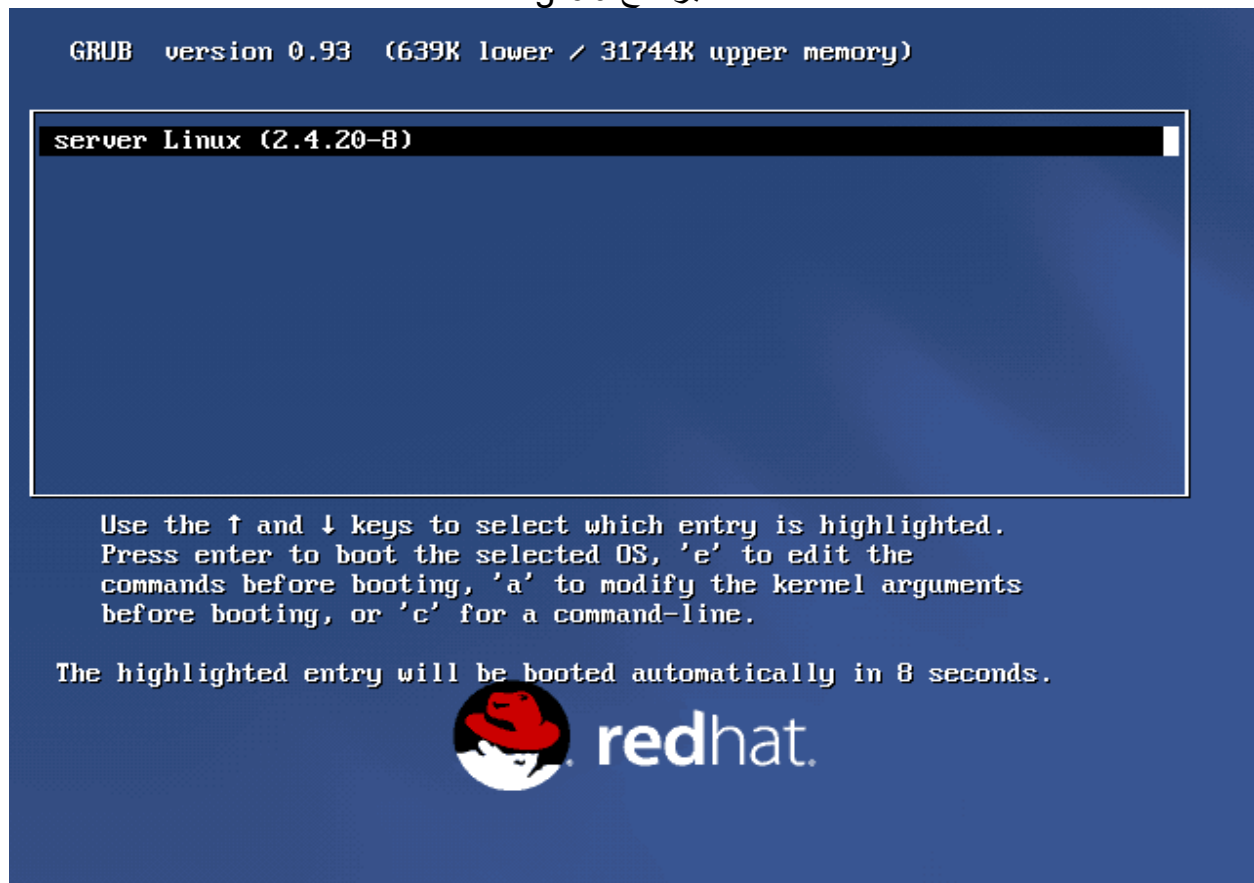
ويوجد اكثر من برنامج يقوم بوظيفه ال boot loader . ولكن اشهرهم اثنان ، وهم

برنامج lilo وهو اختصار ل **linux loader** .  
برنامج grub وهو اختصار ل **grand unified boot loader** .

والمستخدم حاليا هو ال grub ، فهو الاحدث والافضل بسبب عدد من المميزات الموجوده به .

وبرنامج ال grub هذا ، هو ببساطه الشاشة الزرقاء التى تراها عند تشغيل الجهاز

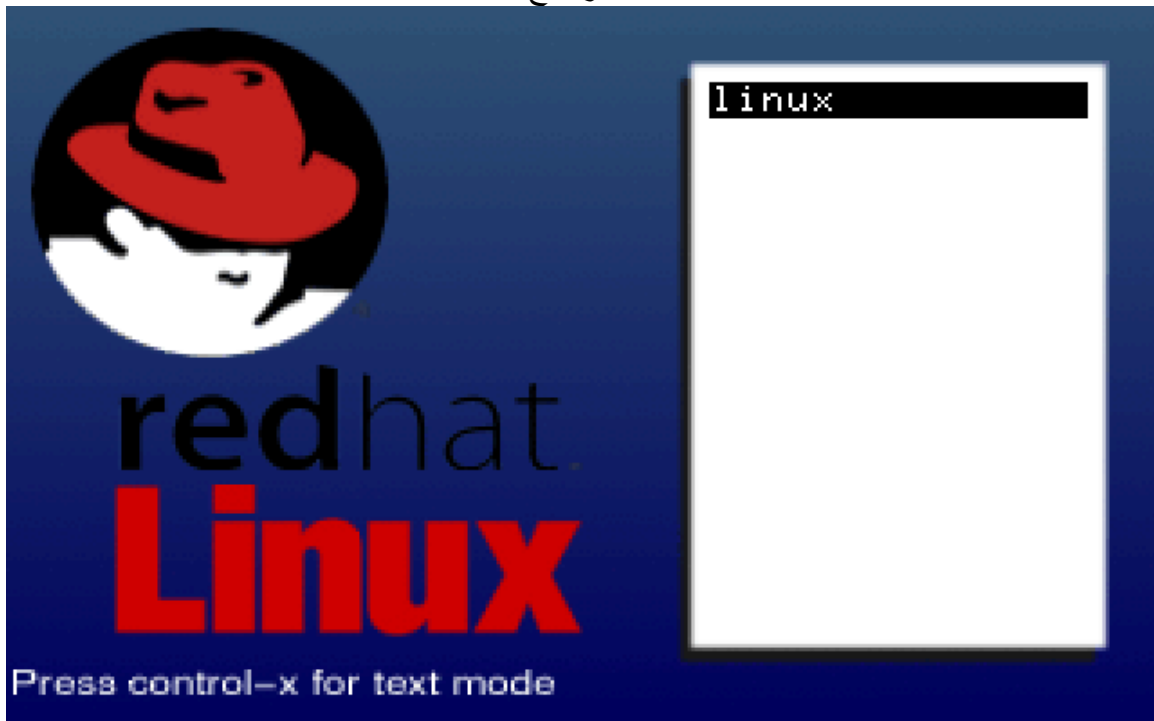
شكل برنامج grub



## ومن مميزات grub

- انه برنامج اكثر مرونة وسهولة في التعامل من lilo .
- لبرنامج grub ، minimal shell خاصه به ،شبيه بال shell التى نتعامل مع النظام بها ، والمسماه bash .
- برنامج grub يوفر خاصيه التشفير (encryption) لكلمات السر الموجوده بملف تهيئته .
- يستخدم برنامج grub مع العديد من انظمه unix الاخرى ، فهو يستعمل مع GNU/hurd و مع BSD و مع غيرهم ايضا ، اما lilo فهو خاص ب لينوكس فقط .
- بالاضافه الى العديد من المميزات الاخرى .

شكل برنامج lilo



ومن الملاحظ ان المساحه الموجوده فى ال mbr (512byte) تعد صغيره ومحدوده جدا لوضع اى برنامج بها ، ولهذا فان البرنامج الذى يوضع فى ال mbr سواء كان lilo او grub ، يجزأ الى اكثر من مرحله (stage) ، فيوضع بال mbr جزء ويسمى stage 1 ، ويوضع جزء اخر فى ال hard disk وبالضبط فى /boot/ ويسمى stage 2 .

بالنسبه للبرنامج grub فان له ثلاث مراحل

- |           |                                              |
|-----------|----------------------------------------------|
| stage 1   | وهى الموجوده بال mbr                         |
| stage 1.5 | وهى الموجوده بالمجلد /boot/                  |
| stage 2   | وهى عباره عن الشاشة الزرقاء التى سبق عرضها . |

## ملف تهيئه ال grub .

يوجد لبرنامج grub ملف تهيئه موجود فى المسار /etc ويسمى grub.conf .  
وشكل هذا الملف كالاتى

```
# grub.conf generated by anaconda
#
# Note that you do not have to rerun grub after making changes to this file
# NOTICE: You have a /boot partition. This means that
#           all kernel and initrd paths are relative to /boot/, eg.
#           root (hd0,0)
#           kernel /vmlinuz-version ro root=/dev/hda5
#           initrd /initrd-version.img
#boot=/dev/hda
default=0
timeout=10
splashimage=(hd0,0)/grub/splash.xpm.gz
title server Linux (2.4.20-8)
    root (hd0,0)
    kernel /vmlinuz-2.4.20-8 ro root=LABEL=/
    initrd /initrd-2.4.20-8.img
~
~
~
```

فكما نرى ، ان هذا الملف عبارته عن جزئين ، الجزء الاول هو ذو اللون اللبني ، والجزء الثانى عبارته عن اللون الابيض العادى .

فالجزء الاول عبارته عن comment (تعليق) اى انه يقرأ ولكن لا ينفذ من قبل النظام ، بمعنى اخر هو بمثابة التعليمات للمتعامل مع الملف .  
اما الجزء الثانى فهو الذى نتعامل معه ، وهو ايضا ما يقرأه النظام وينفذه .

(ال comment بصفه عامه -فى كل ملفات النظام- هو السطر الذى يبدأ بالعلامه # ، فلو ازيلت هذه العلامه لتغير لون السطر واصبح من الاسطر التى يقرأها النظام وينفذها)

ونلاحظ ايضا ان هذا الجزء الثانى (ذو اللون الابيض) يقسم الى جزئين  
اول جزء هو فقط الثلاث اسطر الاولى منه ، وهم default و timeout و splashimage .  
وثانى جزء هو بقية الاسطر الموجوده لنهايه هذا الملف ، حتى لو كانت عشرات الاسطر .

فالجزء الاول بمثابة المعلومات العامه لبرنامج grub ومعانيهم هى

**default** تعنى ، نظام التشغيل الذى سيقوم النظام بتشغيله مباشره اذا لم يتم الاختيار بين الانظمه الموجوده -وبتوضيح ذلك اكثر فى حاله وجود اكثر من نظام على الجهاز- فالرقم 0 للنظام الاول والرقم 1 للثانى وهكذا .

**timeout** وتعنى كم من الوقت سينتظر لى يبدء تحميل النظام المعين ، ونرى ان الوقت محدد ب 10 ثوانى .

**splashimage** وهى صورته الشاشة الزرقاء التى تظهر عند تشغيل الجهاز .

ثم بعد ذلك نرى الجزء الثانى ، وهو الذى يبدأ بكلمه **title** وبجانب هذه الكلمه يوجد الاسم الذى تسمى به نظامك .

ملاحظه : اذا كان بالجهاز اكثر من نظام تشغيل ، فاننا سنرى لكل نظام عدد من الاسطر بمثابه معلومات عنه ، وتبدأ هذه الاسطر بالكلمه **title** . بمعنى اخر ، فى حاله وجود ويندوز على جهازك فسنرى معلومات عنه وهذه المعلومات ستبدأ بالكلمه **title** .

ثم بعد ذلك كلمه **root** وهو ليس ال **root** الخاص بالنظام ، بل هو المجلد الذى يحوى كل المعلومات السابق الاشاره اليها ، وهو **/boot** .

ثم بعد ذلك مكان ال **kernel** ، وهو ما يسمى بال **kernel image** ثم يتبعه بقيه ال **option** التى تحدد عمله ،

ونراهم عباره عن (**ro**) والتى تعنى ، عمل **mount** لل **root** هذا بصيغه **read only** وايضا (**root=/LABEL=**) وهذا يعنى ان ال **root** الخاص بالنظام هو ما يسمى **LABEL** بالملف **fstab** .

مصطلح **kernel image** يطلق على نسخه مصغره ومضغوطة من الكرنل موجوده بالمجلد **/boot** ، وهى اول ما يتم تحميله فى ال **Ram** وبناء عليه يتم تحميل بقيه النظام ، وهذه ال **image** تحوى بالكاد اقل ما يحتاجه النظام من ال **modules** و **drivers** لكى يبدأ عمله .

ثم بعد ذلك مكان ال **initrd** وهى عباره عن **initial root disk** او ال **root ( / )** المعروف لنا اى تحميل ال **/** هذا فى ال **Ram** مع نسخه الكرنل السابق الاشاره اليها .

ملاحظه هامه : عند عمل **boot disk** فان النظام يضع على ال **floppy** ال **kernel image** وال **initrd** السابق الاشاره اليهم .

تعد المعلومات السابقه اقل تهيئه ممكنه قد تكون موجوده فى الملف **grub.conf** . ويمكن الرجوع لصفحات المساعده **info grub** للاطلاع على كافه الامكانيات المتاحة لملف التهيئه .

## الخطوة الثالثة : kernel booting .

وتتمثل هذه الخطوة في تحميل ال kernel image الى ال Ram ، وهذه ال image -كما اسلفنا- مضغوطة ، فيقوم الكرنل بفك (uncompress) نفسه داخل ال Ram ويبدأ عدة خطوات

- . قرأه ال option المحدده له فى ملف grub.conf . (والتي سبق شرحهم)
- . اكتشاف او اختبار كل ال hard ware الموجود على الجهاز مثل ال hard disk و network adapter وال mouse وال key board والكروت المختلفه .
- . التحول الى multi tasking وال multi user .
- . عمل mount لل /
- . تشغيل اول process فى النظام ، وهى init والتي تاخذ رقم 1 فى ال process التي تعمل على النظام ككل ، وهذه ال init تقوم بتشغيل بقيه ال process التي ستعمل على النظام

كل هذه الخطوات يقوم الكرنل بادائهام بسرعه كبيره ، حتى انك لا تكاد تستطيع قرأه اى من الرسائل التي تظهر امامك على الشاشة ، ولهذا فهناك امر ، وظيفته عرض كل هذه الاجراءات التي يقوم الكرنل بادائها -والتي بالطبع تسجل بالنظام- وهذا الامر هو

**dmesg**

ولان الناتج من هذا الامر سطور كثيره فيمكن ادخال هذا الناتج للامر less لقرأه المحتوى صفحه بصفحه ، وذلك بالاتي less | dmesg .

## الخطوة الرابعة : init .

بوصول النظام الى هذه الخطوة ، فانه يكون قد تجاوز مراحل الاعداد والكشف على مكونات الجهاز ، وبدء بالفعل فى تشغيل ال process المختلفه .

وتقوم init بقرأه الملف inittab الموجود فى المسار /etc ، وبناء على المعلومات الموجوده بهذا الملف فانها تقوم بتشغيل ال process المختلفه .

ومن ضمن المعلومات التى يعرفها init من الملف inittab ، ال runlevel المحدد الذى سيشغل فيه النظام .

## ال runlevel

وال runlevel هذا هو الحاله التى سيعمل عليها النظام .  
فمثلا عندما يعمل مدير النظام منفردا -بدون وجود اى مستخدمين- فهذا runlevel ، وعندما يعمل متصلا بالشبكة فهذا runlevel اخر وهكذا .

ولتبسيط ال runlevel هذا ،فاننا نقول انه قد يكون شبيه بال mode الموجود بويندوز مثل safe mode ونحوه

ولل runlevel هذه عدده حالات لكل منها معنى معين ، ولكل حاله رقم يميزها عن بقية ال runlevel الاخرى .

ويتم استخدام الامر runlevel لمعرفة فى اى level يعمل النظام الان ، وصيغته هى

## runlevel

فبعد كتابه هذا الامر -منفردا- تظهر لنا نتيجته مشابهه لهذه

```
[root@elna.jeb root]# runlevel
N 3
[root@elna.jeb root]# _
```

فنرى انه قد ظهر الحرف N ثم بعده الرقم 3 .

فالحرف N يعنى ان النظام (لم يكن فى اى runlevel من قبل)  
اما الرقم 3 فيعنى ان النظام يعمل الان فى ال runlevel رقم 3 .

ويتم التحول من runlevel الى runlevel اخر عن طريق الامر telinit ، والذى سيتم شرحه بعد معرفه معانى ارقام ال runlevel .

فمعانى ارقام هذه ال runlevel (اجمالا) هى

- 0 . halt يعنى
- 1 . single user mode يعنى
- 2 . multiuser without NFS يعنى
- 3 . multiuser with NFS يعنى
- 4 . unused
- 5 . multiuser with graphical تعنى
- 6 . reboot تعنى

ومعانى هذه الارقام بالتفصيل هى

**0** يعنى ، ان النظام سيحدث له halt (بمعنى shutdown) اذا تم تحويله الى هذا ال runlevel

**1** يعنى ان النظام سيعمل فى ال single user mode اى فى وضع المستخدم الواحد ، وهذا المستخدم الواحد هو بالطبع ال root ، وهذا ال runlevel مخصص لعمليات الصيانه وتنزيل ال software المختلفه للنظام .

**2** تعنى وجود مستخدمين مع ال root على النظام ، ولكن ليس بينهم اتصال عن طريق الشبكه ، وهذا ما يعنيه without NFS ، فال NFS اختصارا للشبكه .

**3** تعنى وجود مستخدمين مع ال root وبينهم اتصال عن طريق الشبكه .

**4** غير مستخدمه

**5** مثل رقم 3 ولكن مع وجود ال graphical login اى تشغيل ال x window فى هذا ال runlevel .

**6** عند التحول الى هذا ال runlevel ، فان النظام سيحدث له reboot .

### كيفية التحول من ال runlevel

يتم التحول من runlevel الى runlevel اخر عن طريق الامر **telinit** .

والامر telinit هذا ، عبارة عن link للامر init ، ومن الافضل استخدام الاول ، لان اى خطأ فى استخدام الامر init قد يسبب مشكله فى النظام ، باعتبار ان init هذه process حساسه ولها وضع معين بالنظام .

وصيغه هذا الامر هى

telinit <number>

ويوضع مكان <number> رقم ال runlevel التى تريد التحول اليها .

لاحظ انه اذا تم تحديد رقم 0 لهذا الامر فان النظام سيغلق (shutdown) ، واذا تم تحديد الرقم 6 فان النظام سيجدث له reboot .

وسبب التحول من runlevel الى اخر ، ان النظام -بعد تشغيله- يدخل الى ال runlevel المحدد له بالملف **inittab** -ويكون غالبا 3 او 5- فعندما يريد المدير (root) ان يقوم باى عمل من اعمال الصيانة للنظام او تنزيل ال software المختلفه ، فانه من الافضل التحول الى ال runlevel رقم 1 لانه انسب mode لاداء هذه الوظائف .  
ويلاحظ انه من الممكن -ايضا- اداء هذه الوظائف من 3 او 5 الا انه من الافضل التحول الى 1 .

## ملف inittab .

ملف **inittab** موجود بالمسار **etc** ، اى ان اسمه بالكامل **/etc/inittab** .  
ومسجل بهذا الملف المعلومات التى يقرأها البرنامج **init** .

وكما نرى فى الشكل الذى امامنا ، فهناك ما يقرب من 7 مجموعات من التعليمات والاوامر ، يقوم بتنفيذها البرنامج **init** .

(وقد وضعت بجانب كل مجموعه رقم ، لتسهيل عمليه التمييز بينهم )

```
# Default runlevel 1
id:3:initdefault:

# System initialization. 2
si::sysinit:/etc/rc.d/rc.sysinit

l0:0:wait:/etc/rc.d/rc 0 3
l1:1:wait:/etc/rc.d/rc 1
l2:2:wait:/etc/rc.d/rc 2
l3:3:wait:/etc/rc.d/rc 3
l4:4:wait:/etc/rc.d/rc 4
l5:5:wait:/etc/rc.d/rc 5
l6:6:wait:/etc/rc.d/rc 6

# Things to run in every runlevel. 4
ud::once:/sbin/update

# Trap CTRL-ALT-DELETE 5
ca::ctrlaltdel:/sbin/shutdown -t3 -r now

# Run gettys in standard runlevels 6
1:2345:respawn:/sbin/mingetty tty1
2:2345:respawn:/sbin/mingetty tty2
3:2345:respawn:/sbin/mingetty tty3
4:2345:respawn:/sbin/mingetty tty4
5:2345:respawn:/sbin/mingetty tty5
6:2345:respawn:/sbin/mingetty tty6

# Run xdm in runlevel 5 7
x:5:respawn:/etc/X11/prefdm -nodaemon
```

1. فاؤل الملف به السطر التالى

```
id:3:initdefault
```

وهذا يعنى انه ال runlevel التقليدى (default) الذى سيعمل النظام به كلما دخلنا للنظام .

2. ثم بعد ذلك ال system initialization ، وهذا يعنى اول شى سيعمل بالنظام عند تشغيله ونرى انه تم تحديد ال script الرئيسى للنظام ، والمسمى rc.sysinit .

3. ثم 7 اسطر ، وفيهم يتحدد ما الذى سيعمل (run) فى كل runlevel على حده .  
وبآخر كل سطر يوجد rc ثم رقم ، فهذا ال rc عبارته عن مجلد (ومساره مكتوب بالملف) وكل مجلد يحتوى البرامج التى ستعمل فى كل runlevel ، بناء على الرقم المحدد ، فرقم 0 مثلا -وهو الخاص ب halt- لا يوجد به ايه برامج لتعمل ، لانه مخصص لاغلاق النظام ، اما رقم 2 فيه البرامج التى ستعمل فى ال runlevel رقم 2 ، وهكذا .

4. ثم بعد ذلك البرنامج update ، وهذا البرنامج -كما هو مكتوب- سيعمل عندما يتم تغيير ال runlevel ، ووظيفته هذا البرنامج (daemon) هى كتابته الموجود فى ال buffer الى ال hard disk ويتم ذلك بواسطه الامر sync (انظر info sync) .

5. ثم بعد ذلك ctrlaltdel والتى تعنى ctrl + alt + del ، وبالضغط على هذه المفاتيح الثلاثه سيتم عمل reboot للنظام . (وهذا هو معنى الامر shutdown والخيارات التى معه)

6. بعد ذلك تشغيل البرنامج getty على ال terminal فى كل ال runlevel .

7. وهذه هى المسئوله عن تشغيل ال x window ، فنراه قد كتب x:5 اى تشغيل ال x فى ال runlevel 5 .

ما يجب ملاحظته .

اولا هذه الاقسام السبعه هي اهم الاجزاء الموجوده بالملف `inittab` ، ومعنى هذا انه قد يوجد تفاصيل وبيانات اخرى غير التى وردت هنا (ولكن الطبيعى ان ياتى ما تم ذكره هنا فقط)

ثانيا ، يفصل بين الخانات المختلفه (field) بعلامه `colon` .

اذا تم تغيير ال `id` الموجود باول الملف ، فلا يتم تغييره على الاطلاق الى ارقام 0 او 6 ، لان معنى هذا ان النظام كلما حاول الدخول سيحدث له اما `reboot` او `halt` ، وذلك لانه يتبع ال `id` (بتاعه)

المجموعه (التى اعطيتها رقم 3) والتى نجدها مكتوبه كالاتى `10:0wait:/etc/rc.d/rc` تعنى اولاً رقم 10 هذا هو عبارته عن رقم تسلسلى ، لا يحدث فرق كبير عند تغييره  
ثانيا الرقم 0 هذا الرقم يحدد ال `runlevel` ولهذا فاننا نجد 7 اسطر تحت هذا السطر بهم كل ال `runlevel`

ثالثاً ال `action` التى ستعمل فى كل `runlevel` ، ونراها هنا انها `wait` لكل الاسطر ، والتى تعنى ان على ال `init` ان تنتظر (`wait`) حتى تشتغل (`run`) كل `process` ثم بعد ذلك تكمل `init` التتابع الخاص ببدايه عملها .

رابعاً المسار الذى ستنتظره `init` لى ينهى عملياته ، وهو كما نراه `/etc/rc.d/rc` ثم يتبعه رقم ال `runlevel` الذى سيعمل (`rc` هذه اختصار ل `run command`)

ثم بعد ذلك المجموعه 6 والخاصه بالبرنامج `getty`  
فهذا البرنامج هو الذى يوفر الدخول للنظام ، بمعنى ، ان كلمه `username` و `password` التى تجدها دائماً عندما تدخل الى النظام ، عبارته عن هذا البرنامج `getty` .  
فلو لم يوجد هذا البرنامج ، لما استطاع المستخدم الدخول الى النظام ، فهو بمثابة البوابه للنظام

هذا اولاً ، ثم بعد ذلك سته اسطر هى بمثابة ال `terminal` المتاحه (وهذه ال `terminal` هى التى يتم التبديل بينها بالمفاتيح `ctrl+alt+f1....f6`) ولكن لكل سطر من هذه الاسطر سته معنى

اولاً شكل السطر يكون كالتالى `1:2345respawn:/sbin/mingetty tty1`

- فمن اليسار ، رقم 1 هو لتحديد ال `terminal` رقم 1 (ونجدها مكتوبه باخر السطر `tty1`)

- والرقم 2345 يعنى ال `runlevel` من 2 الى 5

- وكلمه `respawn` لها معنى هام ، وهو ، كلما حدث لل `terminal` اغلاق (`kill`) اعد فتح هذه ال `terminal` ثانيه . ولهذا فعندما تخرج من ال `terminal` فانك تجده ان كلمه `username` قد ظهرت ثانيه .

- ثم بعد هذا نجد المسار الخاص للبرنامج `getty` وهو هنا يسمى `mingetty`

(فى الحقيقه فان لبرنامج `getty` عدده تسميات فهنا تجده يسمى `mingetty` و بصفحات ال `man` يشار اليه ب `agetty` (انظر `man agetty`) ، اذا فالمهم ان تعرف ان الاساس يسمى `getty` ويضاف اليه -لاسباب لا اعلمها- عدده مقاطع باوله)

- ثم رقم ال `terminal` والمشار اليها ب `tty1`

ملاحظه هامه ، عند اراده اضافه عدد من ال terminal الى نظامك ، فما عليك الا اضافه عدد من الاسطر الى هذه الاسطر الستة ، وصيغه هذه الاسطر الجديده ، هي ذاتها صيغه الاسطر الستة ولكن لا بد من تغيير اول رقم بالسطر -ليوافق التسلسل- وايضا تغيير رقم ال tty ليوافق الرقم الجديد. حينئذ ستجد ان نظامك قد اضيف اليه terminal جديده ، تصل اليها بالمفاتيح alt,ctrl,f7

.المجموعه 7 وهي الخاصه بتشغيل البرنامج x display manager .  
ونرى ان ال default لتشغيل xdm هو ال 5 runlevel .  
فعند اراده تشغيل ال x window فى اى runlevel اخر ، فما عليك الا اضافه رقم ال runlevel المحدد بجانب الرقم 5 .

للاطلاع على كل ال action مثل (respawn و wait) وغيرهم مما تم ذكرهم سابقا ، يرجى الرجوع الى صفحه المساعدة man inittab .

### ماذا بعد التعديل فى الملف inittab .

الملف inittab تقرأه ال init مره واحده فقط وذلك عند دخولها للنظام .  
وذلك يعنى ان اى تعديل يحدث فى هذا الملف لن يتم قرأته الا فى المره القادمه لتشغيل النظام .  
ولان بعض الانظمه تستمر فتره طويله قبل ان يتم عمل reboot لها ، فان الامر telinit ياتى معه option وظيفته جعل النظام يعيد قرأه الملف ثانيه .

وهذا الخيار هو q ، فيكون صيغه الامر كالاتى

**telinit q**

ويكتب الحرف q اما capital او small ، ولا تسبقه علامه ( - ) مثل باقى ال option .

## نظرة اخيره على init .

فى هذه النظرة ، سنقترب اكثر من عمليات init ، ونرى بالضبط ما الذى تفعله init .  
وبالتالى سنتعرض للملفات التى تقوم init بتشغيلها .

وبالتعرض لهذه الملفات ، فاننا سنفهم اكثر ، معانى (المسارات) التى وردت فى الملف `inittab` .

ومبدئيا ، فان ال `services` التى تشغلها init موجوده بمجلد يسمى `init.d` وهذا الملف موجود فى المسار `/etc` .

وهذا المجلد لا يتم التعامل معه مباشرة ، لسببين

- لانه يعتبر ال `data base` لكل الملفات التى تشغلها init ، وبالتالى فان اى حذف لمحتوياته ، يعنى ان الخدمة (`services`) التى تم حذفها ، لا يمكن استعادتها .
- لان بالنظام اكثر من `runlevel` ، فان اردت حذف خدمه من `runlevel` معين دون الاخر ، فانك ستتعامل مع ال `runlevel` المقصود ، وليس مع (المستودع) ذاته .

ولهذا فاننا نترك المجلد `init.d` ونتعامل مع روابط (`links`) موجوده لكل `runlevel` .

اذا يوجد `data base` لكل ال `services` التى تشغلها init .  
وتوجد روابط لهذه ال `services` (الموجوده فى ال `data base`) .  
وتعاملنا يكون مع الروابط ، وليس مع ال `data base` .

فكل ملفات ومجلدات init موجوده تحت `/etc` ، وبمجلد يسمى `rc.d` (وهو الذى تم ذكره بالملف `inittab`) اذا فالمسار الكامل لمجلد `init.d` هو `/etc/rc.d/init.d` .  
والمسار الكامل للملفات -التي هى فى نفس الوقت روابط ل `init.d`- هو `/etc/rc.d/rc*.d` .

فالعلامه \* بديله عن ال `runlevel` المختلفه .

اذا ما الذى يعنيه هذا الكلام ؟

يعنى هذا الكلام ، انه لو كان ال `runlevel` المحدد لدخول النظام هو `runlevel 5` ، فان الملفات الموجوده بالمسار `/etc/rc.d/rc5.d` هى التى ستشتغل .  
وان كان ال `runlevel` -المحدد بالملف `inittab`- هو 3 مثلا ، فان الملفات الموجوده بالمجلد `/etc/rc.d/rc3.d` هى التى ستشتغل .

## شكل ملفات rc\*.d من الداخل .

الملفات الموجودة بالمجلد rc\*.d ، لها شكل غريب -بعض الشيء- .

فهذا الشكل الذى امامنا يوضح شكل المجلد rc3.d ، بعد الدخول اليه ببرنامج vi

```
" Press ? for keyboard shortcuts
" Sorted by name (.bak,~, .o, .h, .info, .swp, .obj at end of list)
"= /etc/rc.d/rc3.d/
../
K05innd
K05sasauthd
K20nfs
K24irda
K25squid
K34dhcrelay
K34ypasswdd
K35dhcpcd
K35winbind
K45named
K50tux
K50vsftpd
K74ypserv
K74ypxfrd
S05kudzu
S08ipchains
S08iptables
S10network
S12syslog
S13portmap
"/etc/rc.d/rc3.d/" Illegal file name 1,1 Top
```

فالسطر الاول (المكتوب بالاحمر) يخبرك انه بالضغط على ( ؟ ) فانه سيتم عرض ال help لك .  
والسطر الثانى ، يخبرك ان ترتيب عرض هذا الملف معتمد على اِسماء ال services .  
والسطر الثالث ، يخبرك باسم المجلد الذى تقف فيه ، وهو بالطبع rc3.d .  
ثم بعد ذلك العلامة ( ../ ) والتى تعنى ال parent directory للموجود حاليا ، بمعنى اخر ، هو  
المجلد الذى يعلو المجلد الحالى ( اى (/etc/rc.d ) .

ثم بعد ذلك -وهو المهم- شكل ال services .

فتراها مكتوبه مثل k05innd ومثل k24irda . وهذا الشكل قد نقسمه الى ثلاث اقسام

- الاول ، هو الحرف ، وهو مثل حرف k باول الملف ، او s الموجود باخر الملف . فحرف k اختصار ل kill اى ان هذه ال services سيتم اغلاقها عندما يدخل النظام هذا ال runlevel ، وعكسها الحرف s والتى تعنى ان ال services التى تبدأ بهذا الحرف سوف يتم تشغيلها (start) فى هذا ال runlevel .
- الثانى ، وهو عبارته عن الرقم ، مثل 05 او 24 ، ومهمه هذا الرقم ، ترتيب فتح ال services المختلفه ، فلو انك لاحظت بشده ، ستجد ان هذا الترتيب لل services هو ذاته الذى تراه عندما يشتغل النظام .
- الثالث ، اسم ال service وهو مثل innd و irda وغيرهم .

ملاحظه مفیده ، يمكنك التعرف على وظيفة كل service او process موجوده باى runlevel عن طريق الدخول للمجلد rc\*.d ، ثم الوقوف على اسم ال service المعينه، ثم الضغط enter . فما سيحدث هو انك ستدخل فى ال script الذى يشغل هذه ال service ، وبأول هذا ال script يوجد سطرين او ثلاثه تشرح وظيفة كل service .

## كيفية حذف ال services .

ال process او ال service قد نعتبرها برامج تعمل فى runlevel معين ، وعند اراده الغاء هذا البرنامج من هذا ال runlevel المعين ، فاننا ببساطه نحذف الرابط (link) المخصص له بهذا ال runlevel . (وهنا تظهر فائده وجود الروابط) .

ويتم حذف هذا الرابط بعده طرق ، الا اننى ساقوم بشرح طريقتين -بسيطتين- من هذه الطرق .

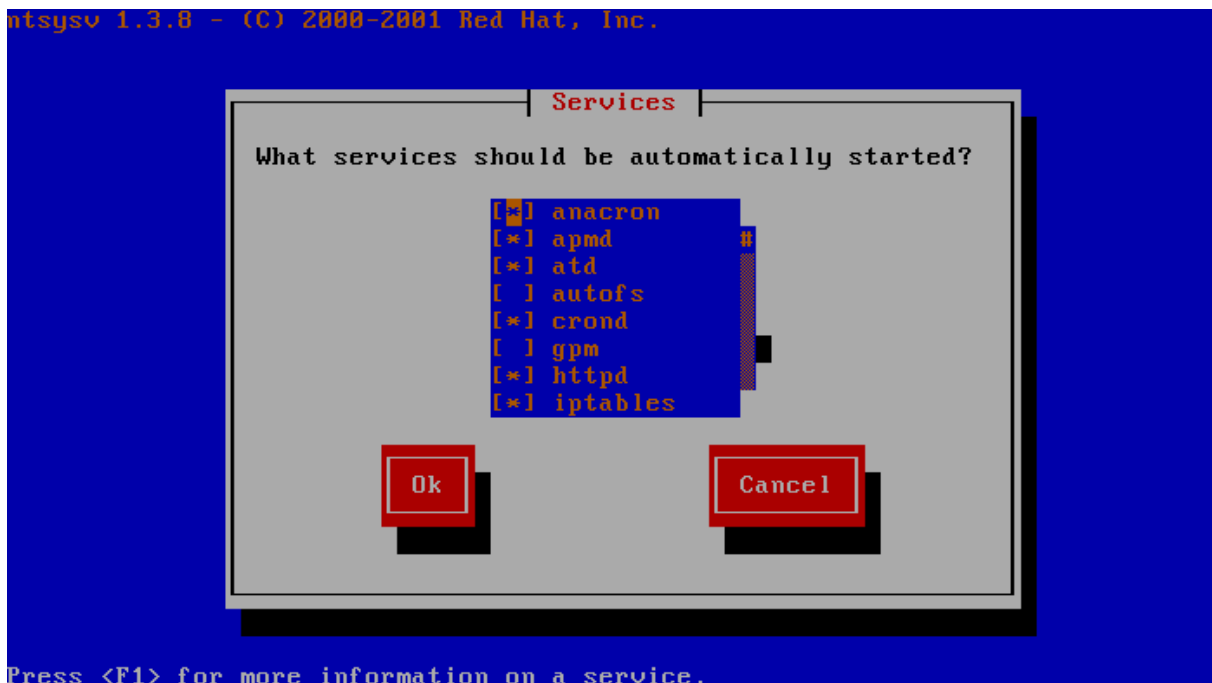
الطريقه الاولى ، هى حذف هذا الرابط يدويا . ويتم ذلك بالدخول للمجلد rc\*.d ، ثم الوقوف على اسم ال service المراد حذفها ، ثم الضغط على المفتاح D (بصيغه capital) .

الطريقه الثانيه ، تتم عن طريق امر يسمى ntsysv ، وصيغته هى

**ntsysv**

ولا يكتب معه اى option ، فقط هو وحده .

ونتيجه هذا الامر عبارته عن برنامج يعمل على ال termina ، وشكله كالاتى



والتعامل مع هذا البرنامج سهل للغاية .

- فما عليك الا استخدام الاسهم up و down للتحرك لافى واسفل .
- واستخدام ال space bar للتأشير على ال service ، سواء لحذفها ، او لتشغيلها .
- (عند تشغيلها تجد العلامة \* ، وبحذف هذه العلامة يتم الالغاء)
- والمفتاح tab يتيح لك التحرك الى الازرار ok و cancel ، للموافقة والخروج او للالغاء .

ملاحظه ، البرامج التى تعمل على ال termina مثل ntsysv ، تسمى consol based .  
اما تلك التى تعمل على ال gui مثل mozilla وغيره ، فتسمى gui based .

## اغلاق النظام .

هناك عدة اوامر تقوم باغلاق النظام مثل **poweroff** او **init 0** .  
الا ان هذه النوعيه من الاوامر تقوم باغلاق النظام بشكل مفاجى للمستخدمين .  
ولهذا فان الامر **shutdown** هو المسئول عن اغلاق النظام بشكل سليم .

والسبب هو ، ان الامر shutdown يقوم بتحذير المستخدمين بان النظام سيتم غلقه ، بعكس الاوامر الاخرى ،  
فانها تقوم باغلاق النظام مباشره ، .  
(واذا لم يتحدد option مع الامر

وصيغته هى

`shutdown [-akrhfnc] [-t secs] time [warning message]`

- يكتب الامر shutdown
- يتبعه اى من هذه الخيارات `[akrhfnc]` مسبوقة بالعلامة ( - )
- وقد يتبعه t- ثم عدد معين من الثوانى
- وتبدل كلمه time بالوقت المحدد التى تريد اغلاق النظام فيه
- ثم الرساله التى ستظهر الى المستخدمين

ولكن لاحظ ان كل الخيارات المكتوبه بين القوسين [ ] هى خيارات غير ملزمه ، بمعنى انك غير مضطر بان تكتبهم . ولهذا فان الحد الادنى الذى لا بد ان تكتبه هو

- الامر shutdown
- الوقت المحدد (وهذا الوقت قد يكتب بارقام مثل 5 او مثل +5 ، وقد تكتب كلمه now للاغلاق الان)

وذلك مثل

`shutdown 9`

فما سيحدث هو ان النظام سيقلق بعد تسع دقائق . ولكن بدون ارسال اى رساله تحذير الى المستخدمين . ويمكن تحديد هذه الرساله كالاتى

`shutdown +9 "system will go down"`



## ال shell للمره الثانيه(نظره اعمق)

تكلما فيما سبق عن ال shell كنظره عامه، اما الان فسنعرض لها ولكن بنظره اكثر تعمقا.

ال shell هذه هي القشره التى نتعامل من خلالها مع النظام ،وبكلام اكثر دقه نتعامل من خلالها مع لب النظام وهو الكرنل kernel .

ولل shell هذه اكثر من نوع ،  
وكل نوع له اسم وايضا له مميزات وخصائص.

### ويمكننا ذكر بعض الخصائص العامه لل shell .

- 1- ال shell هذه عباره عن برنامج مستقل يتعامل مع الكرنل وليس جزء منه.
- 2- اى نوع من انواع ال shell يمكنك تعديله (اى تعديل قواعده) ليتوافق مع الحاجه ، فلل shell ما يسمى ال variables والتي تعد كقواعد تتحكم فى اداء ال shell ، فمثلا ال bash تخزن اخر 1000 امر ادخل اليها ، وهذا هو ال variable ، ومن ثم فيمكنك تعديلها لتخزن مثلا 2000 امر.
- 3- يمكن تشغيل shell من داخل shell اخرى ، وهذا ما يحدث فى حاله ال script ، ومن ثم تكون ال shell الثانيه عباره عن process للاولى تعمل بنفس قواعدها var. اى ال var الخاص بالاصليه هو ايضا ال var الخاص بالفرعيه .
- 4- ال shell تستخدم عده ملفات تهيئه لها ، وتقرأهم كلما بدأت العمل .
- 5- ال shell لها لغه برمجيه خاصه بها ، فيمكنك كتابه ملف بلغه برمجيتها shell script ، او حتى كتابه هذه اللغه على ال command line ذات نفسه (كانك تكتب امر) .

### \*اما بالنسبه لانواعها فعيده

فهناك shell تسمى sh وهى الاقدم نوعا من بقيه الانواع ، وتقريبا تجدها فى كل انظمه unix او المتوافقه معه (unix-like os) ولها اصدارات متطوره ، نذكر منها bourne shell والتي طورت بمعامل bell .  
وهناك ايضا C shell (تختصر csh) والتي طورت لتعمل مع BSD .  
وهناك ايضا Tcsh المتطوره من Cshell ، وايضا Zsh و Ksh وغيرهم من الانواع .

ولينوكس يدعم كل هذه الانواع ، فيمكنك مثلا العمل مع Cshell (القريبه الشبه من لغه البرمجيه (c

ولكن ال default فى لينوكس والتى عندما تنزل النظام ستجدها امامك هى ال bash shell.  
وال bash هذه هى النسخه المطوره من bourne shell والتى سميت كذلك نسبة لمطورها  
steve bourne .  
وال bash من اصدار GNU ، بمعنى انهم هم من قاموا بتطويرها .

## كيف تعمل ال bash .

لل bash قواعد خاصه بها ، مهمه هذه القواعد هي تعديل عمل ال bash .  
وليس معنى ذلك انه عندما تبدأ العمل مع ال bash ان تذهب مباشره لتعديل هذه القواعد، لانها فى وضع standard يلائم المستخدم الجديد .  
ولكن ان اردت تعديل هذه القواعد لتلائم حاجاتك فبسهوله تستطيع ذلك .

وهذه القواعد التى تحكم عمل ال bash تسمى environment variables .  
وهى تعنى نفس معناها الحرفى ، اى متغيرات البيئه .

فال bash هذه هي البيئه ، وهذه البيئه لها متغيرات (عندما تتغير هذه المتغيرات تغير شكل الناتج من عمل ال bash) .

وهناك نوعان من هذه المتغيرات

- 1- متغيرات كليه global variables
- 2- متغيرات جزئيه local variables

### الفرق بين النوعين.

\*المتغيرات الكليه هي المتغيرات التى تحكم عمل هذه ال shell وايضا تحكم عمل ال shell التى قد تتولد منها .

بمعنى اخر عندما تبدأ انت العمل مع ال shell الرئيسيه فانها تستخدم هذه ال variable ، وان تولدت shell من هذه الرئيسيه فان هذه ال child shell تأخذ نفس قواعد ال var ال shell الرئيسيه.

\*اما المتغيرات الجزئيه (local) فعندما تتولد shell اخرى فلا يتم تصدير المتغيرات (var) لها .

(child shell او ال shell المتوالده ، هي ال shell التى تنشأ لتقوم بعمل معين ومحدد ، وتظهر بوضوح مع برمجته ال shell بلغه الاسكربت script . فبعد كتابته ملف به وظائف معينه لتقوم ال bash بتنفيذها > وهذا هو ال script < فعند تنفيذ هذه العمليه تتوالد ال child shell الى ان تنتهى من تنفيذ هذه العمليه ومن ثم ترجع ثانيه الى ال shell الاساسيه).

وانت لن تلاحظها ولن ترى اى تغيير يظهر على ال shell الاساسيه التى تعمل عليها ، ولكن كل هذه العمليات من اختصاص النظام فقط يكفيك ان تعرف

ان ال var المخصصه لل shell الرئيسيه سوف تكون هي ايضا المخصصه لل shell الفرعيه .

الان يمكن معرفه ال global environment (المتغيرات الكليه) عن طريق تنفيذ الامر **env**

```

[ahmed1@elna.jeeb ahmed1]$ env
HOSTNAME=elna.jeeb
TERM=linux
SHELL=/bin/bash
HISTSIZE=1000
USER=ahmed1
LS_COLORS=no=00:fi=00:di=01:34:ln=01:36:pi=40:33:so=01:35:bd=40:33:01:cd=40:33:01:or=01:05:37:41:mi=01:05:37:41:ex=01:32:*.cmd=01:32:*.exe=01:32:*.com=01:32:*.bat=01:32:*.sh=01:32:*.csh=01:32:*.tar=01:31:*.tgz=01:31:*.arj=01:31:*.taz=01:31:*.lzh=01:31:*.zip=01:31:*.z=01:31:*.Z=01:31:*.gz=01:31:*.bz2=01:31:*.bz=01:31:*.tz=01:31:*.rpm=01:31:*.cpio=01:31:*.jpg=01:35:*.gif=01:35:*.bmp=01:35:*.xpm=01:35:*.xpm=01:35:*.png=01:35:*.tif=01:35:
MAIL=/var/spool/mail/ahmed1
PATH=/usr/local/bin:/bin:/usr/bin:/usr/X11R6/bin:/home/ahmed/bin
INPUTRC=/etc/inputrc
PWD=/home/ahmed
LANG=en_US.UTF-8
SSH_ASKPASS=/usr/libexec/openssh/gnome-ssh-askpass
SHLVL=1
HOME=/home/ahmed
LOGNAME=ahmed1
LESSOPEN=|/usr/bin/lesspipe.sh %s
G_BROKEN_FILENAMES=1
_=/bin/env
[ahmed1@elna.jeeb ahmed1]$

```

فى المثال السابق وبعد تنفيذ الامر env يمكننا ملاحظه الاتى

ان المتغيره هى الموجود على اليسار capital ، اما الموجود على اليمين small فهى قيمه هذه المتغيره، يفصل بينهم علامه =

فمثلا HOSTNAME هى المتغيره variable اما elna.jeeb فهى قيمه هذه المتغيره ، اذا فالنظام هنا يسمى ب elna.jeeb .

\*) (طبعا هذه المتغيرات تختلف من نظام لنظام وليس فقط فى اسم النظام ولكن ايضا فى عدد هذه المتغيرات، يمكن الرجوع فى ذلك الى صفحه المساعدة (man bash)

ايضا يظهر فى المثال ال variable المسماه SHELL والتي تعنى نوع ال shell المستخدمه ، وهنا نجدها ال bash طبعا .

اما المتغيره HISTSIZE فهى المسئوله عن عدد الاوامر التى تستطيع ال bash تخزينها .

فى هذا المثال الثانى سنتعرض للمتغيرات الجزئيه (التي لا تصدر لغيرها) ويمكن عرض هذه المتغيرات عن طريق الامر **set**

```

BASH=/bin/bash
BASH_VERSINFO=( [0]="2" [1]="05b" [2]="0" [3]="1" [4]="release" [5]="i386-
linux-gnu")
BASH_VERSION='2.05b.0(1)-release'
COLORS=/etc/DIR_COLORS
COLUMNS=80
DIRSTACK=()
EUID=500
GROUPS=()
G_BROKEN_FILENAMES=1
HISTFILE=/home/ahmed/.bash_history
HISTFILESIZE=1000
HISTSIZE=1000
HOME=/home/ahmed
HOSTNAME=elna.jeeb
HOSTTYPE=i386
IFS=$' \t\n'
INPUTRC=/etc/inputrc
LANG=en_US.UTF-8
LESSOPEN='! /usr/bin/lesspipe.sh %s'
LINES=25
LOGNAME=ahmed1
LS_COLORS='no=00:fi=00:di=01;34:ln=01;36:pi=40;33:so=01;35:bd=40;33;01:cd
01:or=01;05;37;41:mi=01;05;37;41:ex=01;32:*.cmd=01;32:*.exe=01;32:*.com=0
:
```

وقبل الكلام عن المتغيرات المعروضة بهذا الامر يجب علينا ملاحظه ان

\*نتيجه الامر ين تكد ان تكون متشابهه ولكن عدد المتغيرات التى يعرضها الامر set هو اكثر من تلك التى يعرضها الامر env .

\*ايضا الناتج من الامر set نراه يعرض تفاصيل اكثر عن النظام من تلك الذى ينتجه الامر env .

فمثلا نجده يعرض ثلاث متغيرات خاصه بحجم الاوامر التى تستطيع ال bash الاحتفاظ بها فى ذاكرتها (والذاكره هنا عباره عن ملف) وهذه المتغيرات هى .

**HISTFILE** والذى يوضح اسم الملف الذى يحوى الاوامر التى تحتفظ بها ال bash ، وهو هنا الملف **bash\_history** . ، الموجود فى مجلد ال **home** الخاص بالمستخدم (كل مستخدم له ملف بنفس الاسم ولذا ان الوظيفه) ولا ننسى ال **dot** التى قبل الاسم والتى تعنى ان الملف مخفى (hidden) .

وال **variable** الثانى هو **HISTFILESIZE** والذى يعنى عدد الاوامر التى يستطيع الملف تخزينها والاحتفاظ بها.

اما ال **variable** الثالث و المسمى **HISTSIZE** فوظيفته تحديد عدد الاوامر التى ستعرض على ال bash امام المستخدم عندما يتم استدعاء الاوامر التى فى الذاكره . (انظر الامر **history** باسفل)

## من خصائص ال bash

كما اسلفنا فان ال bash قد تم تطويرها من قبل مشروع GNU او ما يعرف (fsf) ومعنى تطويرها انهم قد اخذوا قواعد ال shell التي كانت تدعى sh - واصداراتها- وقاموا بانشاء shell اخرى تتمتع بصفات sh مضيفين اليها عدد من الملامح الجديده والتي لم تكن موجوده في سابقتها .

من هذه الخصائص امران

history

و

alias

## الخاصيه الاولى

### history

يعد من خصائص ال bash قدرتها على تخزين الاوامر التي ادخلت اليها من قبل، وهذه الخاصيه تكون مفيده جدا خاصه مع الاوامر الطويله ، فبدلا من ادخال الامر عدة مرات فيكفى فقط استدعاءه من ال history (ذاكره ال bash)

كل ما عليك عمله هو فقط كتابه الامر history  
ومن ثم ستجد امامك قائمه طويله بالاوامر التي قد كتبتها من قبل.

### ولكن كيف يعمل الامر history ؟

كما ذكرنا فان لل bash قواعد تحكم عملها ، فعندما يكتب هذا الامر لل bash فانها تذهب لتبحث عن كيفية التعامل معه  
وطبعا ستجد الثلاث متغيرات variables المتعلقين بالذاكره (history) الذي سبق الكلام عنهم عند الامر set .

فبيدا الامر history في قراءة ملف ال history المسمى bash\_history . ، ولا يعرض كل محتوياته بل يسال ال variable المسئول عن عدد الاوامر التي ستعرض وهو HISTSIZE .

وكما هو واضح بالمثال السابق فان عدد الاوامر التي سيعرضها 1000 امر ، بمعنى اخر ستجد الشاشة امامك قد امتلات بقائمه من الاوامر وكل امر بجانبه رقمه المسلسل .

اما اذا اردت فقط اخر 20 امر مثلا فاما ان تضيف option للامر history

او ان تعدل قيمه ال HISTSIZE variable وهذا هو ما سنناقشه.

## الامر export وتعديل ال variable .

هناك طريقتان لتعديل ال variable  
الاولى مؤقتة وتنتهى بالخروج من ال shell  
اما الثانيه فهي تدوم حتى بعد خروجك من ال shell.

وهذه الطريقه المؤقتة تستخدم لانجاز مهمه عابره لا تستدعى تغيير ال variable تغييرا دائما ،  
ويقوم باداء هذا التغيير المؤقت الامر **export** .

والامر export يقوم بتغيير ال variable الكليه والجزئيه ، اى التى تصدر الى ال child shell  
(والتي تستعرض بالامر env) وايضا التى لاتصدر (والتي تستعرض بالامر set) .

\* ليس هذا فحسب ولكنه ايضا يستخدم لانشاء variable جديد كما سنرى .

صيغه الامر

`export VARIABLE=value`

وسنقوم بتنفيذ هذا الامر export على ال HISTSIZE var .

`export HISTSIZE=20`

فما قمنا به هو اننا اخبرنا ال bash ان العدد الجديد للوامر التى ستعرض للامر history هو  
20 بدلا من 1000.

الان نفذ الامر history مره ثانيه وانظر للناتج ، سوف تجده يعرض 20 امر فقط (لاحظ مسلسل  
الارقام)

هذا مثال بسيط لكيفيه تغيير قيم ال var بواسطه الامر export .

ولكنك فى بعض الحالات تحتاج لانشاء variable جديد ، تستخدمها لانجاز مهمه مؤقتة ايضا .  
وتستطيع انشاء هذه ال var بنفس الطريقه السابقه ولكن هذه المره ستدخل المتغيره وقيمتها وليس  
القيمه فقط كما سبق .

على سبيل المثال تستطيع ان تجعل اسمك هو ال variable الجديد

`export AHMED=adminstrator`

(لا بد لل var ان تكون capital لكى تفهم ال bash ان هذه variable)

## لماذا ننشأ variable ؟

نفترض مثلا انك كتبت script (ملف به عدة وظائف مكتوب بلغه تفهمها ال bash وتنفذها) لاداء مهمه معينه ، وفى هذه المهمه يوجد اسم شخص ، فطبعا ال bash لن تعرف هذا الاسم وبالتالي ستقشل ال bash فى اداء مهمتها، ولكى لا تقشل ال bash فى تنفيذ هذا ال script (لانيها لا تعرف اسم هذا الشخص) سوف نعرفها على هذا الشخص ، وذلك عن طريق 1- وصفه 2- ووضعه فى المكان التى ترجع اليه ال bash لتراجع قواعدها التى تعمل بها.

فالمثال الذى كتبناه يعرف الشخص (احمد) لل bash على انه المدير.  
ووضعناه فى قائمه القواعد التى ترجع لها عن طريق الامر export.

كل هذه الخطوات السابقة تجعل ال variable معروفه لل bash بطريقه مؤقتة ولكن ان اردت جعلها دائمة فالمسألة بسيطة وهذا هو ما سيتم شرحه لاحقا .

### ملاحظه

يمكن ايضا تغيير ال variable عن طريق تعريف ال var فقط لل bash وذلك بان تكتب

AHMED=adminstrator

ولكن هذه القيمه لا تصدر الى global environment بل تظل تعمل مع ال shell الاساسيه فقط ، فان تولدت shell من هذه الاساسيه فهذه المتولده لن تتعرف على هذه ال variable الجديده.

## تغيير ال variable بطريقة دائمة.

هذه الطريقة فى تغيير ال var اسهل من الطريقة السابقة  
فبدلاً من كتابه امر وتعريف variable جديده فيكفيك فقط التعديل فى الملف الرئيسى الذى تقرأه  
ال bash قبل ان يبدأ النظام فى العمل .

\* وهذا الامر سيقودنا الى الكلام عن الملفات التى تتعامل معها ال bash او ال shell عموماً ، وهو  
الامر الذى سنتعرض له بعد قليل .

الملف الرئيسى الذى تقرأه ال bash عند الدخول للنظام (log in) هو الملف المسمى profile  
والموجود فى المسار /etc ، اى ان اسمه بالكامل هو /etc/profile .

يلاحظ ان هذا الملف هو الملف الرئيسى الذى تقرأه ال bash عند الدخول للنظام والتغيير فيه  
سيشمل كل المستخدمين ، اما تغيير ال var لمستخدم معين فستذكر عن التحديث عن ملفات bash

ولكى نستطيع التعديل فيه فلا بد من استخدام محرر نصوص ( وهو عبارته عن برنامج يستطيع  
التعامل مع الملفات -انشاء ، تعديل- ولكن عن طريق ال shell)

واشهر محرر نصوص فى عالم ال Unix هو المحرر vi (ينطق vee eye)  
وسنتكلم عنه عندما يحين الكلام عن محررات النصوص ، ولكننا الان سنستخدمه للتعديل فى  
الملف سابق الذكر .

ولكى نتمكن من التعديل فى الملف /etc/profile فلا بد اولاً من كونك المدير root .  
ثانياً لابد من تحميل هذا الملف داخل برنامج vi -نفتحه من خلاله- ويتم ذلك عن طريق الامر الاتى

vi /etc/profile

الان سنجد shell اخرى قد فتحت امامنا وشكلها كالاتى

```
# /etc/profile

# System wide environment and startup programs, for login setup
# Functions and aliases go in /etc/bashrc

pathmunge () {
    if ! echo $PATH | /bin/egrep -q "^(!:$)$(!:$)" ; then
        if [ "$2" = "after" ] ; then
            PATH=$PATH:$1
        else
            PATH=$1:$PATH
        fi
    fi
}

# Path manipulation
if [ `id -u` = 0 ]; then
    pathmunge /sbin
    pathmunge /usr/sbin
    pathmunge /usr/local/sbin
fi

pathmunge /usr/X11R6/bin after
```

الان انت داخل ملف `/etc/profile` ولكى تعدل فيه ما عليك الا التحرك بالسهم down arrow لكى تتحرك لاسفل .

ولنفترض انك ستغير ال variable الخاصة HISTSIZE ، فستتحرك حتى تصل الى السطر التالى

```
HOSTNAME=`/bin/hostname`
HISTSIZE=1000
```

والان سنبدأ بالتعامل مع الملف بالمحرر vi

تحرك بالسهم يمينا حتى تقف على 1000 عندئذ اضغط المفتاح del لتمسح القيمة السابقة بعد ذلك اضغط المفتاح i (والذى يعنى insert) اكتب 20 مكان القيمة السابقة 1000

```
HOSTNAME=`/bin/hostname`
HISTSIZE=20
```

بعد ذلك اضغط المفتاح Esc (وستلاحظ اختفاء كلمة ----- INSERT ---- من اسفل الشاشة) ثم اضغط الثلاث مفاتيح الاتيه على التوالى wq: -بدون اى مسافات بينهم- وستلاحظ انه قد كتب باسفل الشاشة ما يدل على ان التغييرات قد حفظت فى الملف .  
بهذه الطريقة يتحول ال variable الى قيمة ثابتة لا تتغير بالخروج من ال bash .

### alias

الخاصيه الثانيه من خصائص ال bash والتي سنناقشها الان هي alias وتتخلص وظيفه alias فى امكانيته (صنع) اوامر جديده ، او تعديل الاوامر الثابته.

ومعنى صنع امر جديد ، اى دمج عده اوامر معا ، او حتى دمج امر مع عده خيارات له ولكن بطريقه ال alias .  
والتي تعنى انه عند كتابه الامر (المعد سلفا بطريقه alias) فسيظهر الناتج ليس فقط للامر وحده ولكن سيظهر كما لو انك كتبت الخيارات معه .

وبالمثال يتضح المقال.

الان اكتب على ال bash

```
alias ahmed='ls -la'
```

فما فعلته انا فى المثال السابق هو جعل ahmed يساوى الامر ls -al ، (اصبح ahmed امر الان)  
فان كتبت لل bash كلمه ahmed فان الناتج سيكون مثل نتيجه الامر ls -al .

طبعا فى المثال السابق اتيت باسم شخص لجعل المثال اوضح ، ولكن الحال دائما هو ان يكون هذا الاسم (اسم الشخص) امر .

مثال اخر

```
alias rm='rm -i'
```

فى هذا المثال فان الامر rm (والخاص بالغاء الملفات) عند كتابته وحده -بعد عمل alias له- فان ال bash ستعتبر انه قد كتب بالخيار -i- معه ، وكلما كتبتة وحده فقط بدون خيار معه فسوف تسالك ال bash (هل انت متأكد من الغاء الملف) وكأنه قد كتب بالخيار -i- معه .

اذا اصبح الامر rm وحده يساوى الامر rm -i ولكن بدون كتابه الخيار معه كل مره .

اذا فمعنى alias هو اختصار الاوامر .

ولماذا نختصر الاوامر ؟

السبب هو بدلا من كتابه الاوامر الطويله كلما احتجنا اليها ، فاننا نستطيع اختصارها فى امر قصير او حتى فى حرف واحد فقط  
هذا من ناحيه ، ومن ناحيه اخرى فهو مفيد كاجراء امان تضمن به انك لن تلغى ملفات ذات اهميه بدون قصد منك خاصه ان كنت root ، كما هو واضح فى مثال الامر rm .

كل ما سبق قد نعتبره تمهيد للامر alias (ولا اريد ان اصنع انطباع ان الامر واستخدامه صعب بل على العكس ف alias امر بسيط جدا ومفيد ايضا وهذا ما سنراه)

الان اكتب على ال bash الامر alias منفردا ، وانظر الى النتائج التى ستظهر لك .

عندما كتبته انا ، اخرج لى هذه النتائج

```
[ahmed1@elnajeeb ahmed]$ alias
alias ahmed='ls -al'
alias l.='ls -d .* --color=tty'
alias ll='ls -l --color=tty'
alias ls='ls --color=tty'
alias which='alias | /usr/bin/which --tty-only --read-alias --show-dot --show-ti
lde'
[ahmed1@elnajeeb ahmed]$ _
```

ايضا النتائج التى ستظهر لك ستكون مشابهه لتلك التى فى المثال السابق .

دعونا الان نناقش النتائج التى فى المثال السابق

السطر الاول كما هو واضح هو ال prompt وفيه مكتوب الامر alias فكانت نتيجته خمس اسطر كل منها يبدأ بكلمه alias

فأول سطر يظهر فيه ان الامر ahmed يعد اختصار للامر ls -al وثانى سطر يوضح ان الامر ( . | ) حرف dot+ L هو اختصار للامر ls بالاضافه للخيار (-d. \*) والذى يعنى اعرض كل الملفات التى تبدأ ب dot (المختفيه) .

وبقيه الاسطر على هذا المنوال (الامر الذى على اليسار يعد اختصار لما هو مكتوب على اليمين)

المهم من هذا المثال ، وما يجب التركيز عليه هو شكل الامر المختصر (الذى على اليمين) فنلاحظ انه بين قوسين صغيرين (كوتة qouta) وهذا بالطبع ذو معنى خاص عند ال bash .

اذا يمكننا الان معرفه كيف يمكن كتابه اختصار للامر باستخدام alias

اولا تكتب alias ثم تترك مسافه  
ثانيا تكتب الامر المراد عمل اختصار له  
ثالثا تضع علامه = (بدون ترك مسافه)  
رابعا تضع علامه ( ' ) qouta وتكتب الامر الاصلى الذى سينفذ عند كتابه اختصاره السابق ، ثم  
تتهى الامر ب qouta اخرى .

ملاحظه

(يلاحظ انك عند كتابه الامر alias وحده – وانت root – ستجد نتيجه مختلفه نسبيا عن اذا ما كتبتَه وانت مستخدم عادى، فسوف تجد الاوامر rm و mv و cp لا بد ان يكون معمول لهم alias باضافه الخيار -i معهم،والذى يعنى تنبيه الroot عندما يحاول الغاء او نقل او نسخ اى من الملفات،لان النظام لا يعارض الroot بالجمله المعروفه (permission denied)

يلاحظ ايضا ان الامر alias تاثيره لا يمتد الى الchild shell فهو خاص فقط بالshell الاساسيه ولا يتعدها لغيرها .

وهذا التغيير الذى يحدث للامر عن طريق alias انما هو تغيير مؤقت ينتهى بالخروج من الbash ، فيرجع الامر كما كان قبل alias .

يمكن ايضا ارجاع الامر لما كان عليه قبل alias عن طريق الامر unalias ، مثل

**unalias rm**

فما ستكتبه هو فقط اختصار الامر(اى ما يكتب على اليسار فقط)

اما لجعل التغيير ثابت لا ينتهى بالخروج من الbash ،فيتم بنفس الطريقه التى تعاملنا بها مع الvariable وهى بالتغيير داخل الملفات .

والملف المسئول عن الalias فى النظام كله هو الملف bashrc الموجود ايضا بالمسار /etc اى ان اسمه كاملا سيكون /etc/bashrc .

يلاحظ ايضا ان التغيير فى هذا الملف سيؤثر على كل المستخدمين اما التغيير لمستخدم بعينه ، فسنذكره عند التحدث عن الملفات التى تتعامل معها الbash .

```
# /etc/bashrc

# System wide functions and aliases
# Environment stuff goes in /etc/profile

# by default, we want this to get set.
# Even for non-interactive, non-login shells.
if [ "`id -gn`" = "`id -un`" -a `id -u` -gt 99 ]; then
    umask 002
else
    umask 022
fi
```

شكل مقدمه الملف /etc/bashrc

ولكى نعدل فيه سنستخدم ايضا المحرر vi (ولا بد من كونك root)

- 1 vi /etc/bashrc
- 2 تحرك بالاسهم
- 3 اضغط i
- 4 ابدأ بالكتابة فى المكان المحدد بالصورة الاتيه
- 5 بعد الانتهاء من الكتابة اضغط (الثلاث مفاتيح) wq: لكى يتم حفظ ما كتبته .

```
# /etc/bashrc

# System wide functions and aliases
# Environment stuff goes in /etc/profile
alias 1='touch'
alias 2='rm'
alias 3='cat'

# by default, we want this to get set.
# Even for non-interactive, non-login shells.
if [ "`id -gn`" = "`id -un`" -a `id -u` -gt 99 ]; then
    umask 002
else
    umask 022
fi
```

مقدمه الملف /etc/bashrc بعد التعديل فيه

يلاحظ اننى قد انشئت ثلاث اوامر جديده ، او بمعنى اخر اختصرت ثلاث اوامر فى ثلاث ارقام .

- 1 فرق عند استخدامه سيعطى نفس نتيجة الامر touch (لانشاء ملف)
- 2 ورقم 2 بديل للامر rm (لإلغاء ملف)
- 3 اما رقم 3 فيمكن به قراءة الملفات مثل الامر cat .

الان اذا كتبت 1 على ال shell فستترجمه ال shell على انه الامر touch ، وهكذا مع بقية الاختصارات aliases .

وطبعا لان الملف /etc/bashrc من ملفات النظام التى تقرأه ال bash عند الدخول للنظام ، فان هذه التغييرات لن تنفذ الا فى دخولك لل bash المره القادمه .



## Regular Expression

المقصود بالregular expression هي هذه العلامات والرموز notations الموجودة على  
الkey board من امثال

| & ' " < > ` ( ) \* ^ \$ ~ !

وهذه العلامات لها معاني خاصه عند الbash ، وذلك لان الbash وان كانت هي واجهه  
المستخدم التى يتعامل مع النظام من خلالها ، الا انها ايضا تعد واجهه تفاعليه (يمكن التعامل  
والتفاعل معها بلغات البرمجه مباشره) interactive interface وبالتالى فهى تتفاعل مع  
المستخدم عند ادخال اى من هذه الرموز اليها وبالتالى تفهمها .

فالbash بمعنى اخر هي بيئه مناسبه جدا لعمل المبرمجين ، فالرموز المستخدمه فى لغات  
البرمجه (بما فيهم الshell script) يمكن كتابتها مباشره على الbash ، ولهذا فهي تعد  
command language interpreter ، لغه برمجه تكتب على سطر الاوامر .

وهذه الرموز تسمى metacharacters او wildcards .

والفرق بين التسميتين يتمثل فى كون الmetacharacter اعم واشمل من الwildcards فكل  
الرموز والعلامات بشكل عام تسمى metacharacter اما التى تتعامل مع الملفات فتسمى  
wildcards (بخاصه العلامات المستخدمه فى البحث عن الملفات واجراء عملياته المتطابق بينهم  
من امثله [ ] , [ - ] , ? , \* )

### استخدامات الmetacharacter .

يمكننا تصنيف استخدامات الmetacharacter على النحو التالى

1- تستخدم ؟ للدلاله على حرف واحد فقط ، فعلى سبيل المثال اذا كان عندك مجلد يسمى  
dir1 واخر يسمى dir10 (اى ان الفرق بينهم هذا الصفر) فعندما تكتب الامر

ls dir?

فسيتم عرض المجلد dir1 (تم ابدال العلامه؟ بالرقم 1) ولا فرق بين رقم او حرف المهم ان يكون  
one character

اما اذا كتبت dir?? بدلا من dir? فسيتم عرض المجلد dir10 .

**2-** تستخدم \* (علامة النجمة) asterisk للدلالة على حرف او اكثر ، فمثل المثال السابق اذا كتبت \* ls فسيتم عرض كل المجلدات والملفات الموجودة في الـ working dir الحالي ،

واذا كتبت ls dir\* فسيعرض المجلدان dir1 و dir10 (لان \* تعني حرف فما اكثر)

**3-** القوسين [ ] ولهما عدة استخدامات

\* اولا [abc...] في هذه الحالة فانت تخبر الـ shell بان تستخدم اى من هذه الحروف الموجودة بين القوسين ، فان كان هناك عدة ملفات تسمى filec و fileb و filea ... الخ ، فما عليك الا كتابته الامر (نفترض cp) يتبعه فقط كلمه file ثم تفتح قوسين وتكتب بينهما abc ،

cp file[abc] onther place

عندئذ سوف تقوم الـ shell بنقل الثلاث ملفات (فما اكثر) الى المكان الجديد، وذلك بدلا من كتابته اسماء الثلاث ملفات كامله ثلاث مرات.

\* ثانيا [a-m] في هذه الحالة بدلا من كتابته الحروف من a الى m ، يكفي فقط ان تضع علامه dash والتي تعني ( من هذا الحرف الى ذاك ) .  
ومثل المثال السابق ، فبدلا من كتابته اسماء الملفات المراد نقلها كامله ( وهى هنا الملفات التى تنتهى بالحرف a وحتى الحرف m اى 13 ملف) فبدلا من كتابتها 13 مره فاننا فقط سنكتب الامر

cp file[a-m] onther place

\* ثالثا [!abc] وتعني العلامه ! الموجوده قبل الاحرف abc ان تتجاهل هذه الحروف .

امثله على metacharacter .

لفهم الـ metacharacter بصورة افضل يمكننا تطبيق عدة امثله ،

اولا سننشأ 5 ملفات اسمائهم filea ,fileb ,filec ,filed ,filee  
ثانيا سننشأ ايضا 5 ملفات file1 ,file2 ,file3 ,file4 ,file5 ،

عن طريق الامر ls (لعرض الملفات) نكتب

- ls file\* (بذلك نستطيع ان نعرض الـ 10 ملفات) .

- ls file[abc] (سيتم عرض ثلاث ملفات فقط) .

- `ls file* [!12345]` ( بذلك سيحاول الامر `ls` عرض كل الملفات التي تبدأ بكلمه `file` ثم يتجاهل الملفات التي تنتهى بالارقام من 1 الى 5 ، ومن ثم سيعرض فقط الملفات التي تنتهى بالحروف) .

### تنفيذ عدة اوامر فى نفس الوقت .

من خصائص ال `bash` انك تستطيع دمج عدة اوامر معا ، ومن ثم تقوم هى بتنفيذهم واحدا تلو الاخر ، دون الحاجه الى كتابه كل امر على حده ، وأحد هذه الخصائص هى

4 - علامه semi colone ( ; ) > تجدها بجانب المفتاح `enter` < فبهذه علامه يمكننا دمج امرين او اكثر معا

الصيغه `command ; command`  
مثل

`touch newfile ; ls`

فما سيحدث هو انشاء الملف `newfile` ثم عرض كل المحتويات الموجوده فى المجلد الحالى او ما يسمى `working dir` .

ايضا يمكننا تطبيق المثال التالى

`mkdir dir ; cd dir`

فأنشأنا المجلد `dir` ثم بعد ذلك سننتقل اليه (اي سنقف عليه ويكون هو ال `working dir`) .

## الstdin و stdout .

كلنا طبعا نعرف ان الkey board و الmouse تعد من الاجزاء المسماة input device ، وايضا ان الطابعة والسماعات تعد من الاجزاء المسماة output device .

وفى لينوكس ، -وبلفظ ادق- فى الshell هناك ايضا ادوات للدخال وللإخراج من وإلى الshell (ادخال الاوامر واخراج النتائج) ولكن تسميتهم ليست input و output ولكنها تسمى stdin و stdout . (اختصارا لكلمتى standard in و standard out) .

فالstdin هى الkey board المسئولة الاساسيه عن ادخال البيانات والاوامر الى الshell .

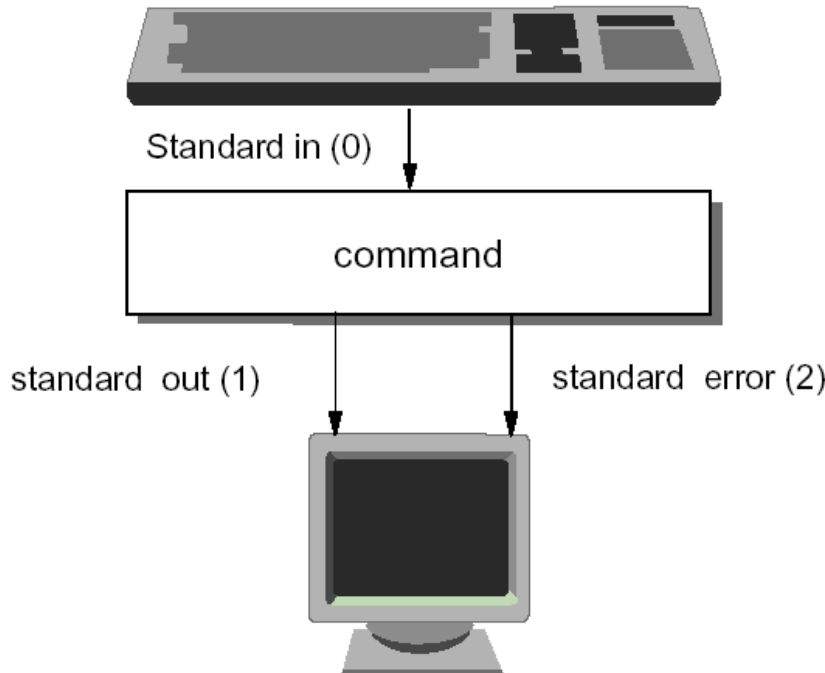
اما الstdout فهى بشكل عام النتيجة التى تخرج من الامر، وهى فى الغالب الشاشة terminal والتى يظهر عليها ناتج الامر الذى نفذته الshell .

ليس هذا كل شى فإلنتاج او النتيجة التى تخرج من تنفيذ الامر ليست غالبا صحيحة فقد يكون الامر مكتوب بطريقة خاطئه مثلا ، فى هذه الحالة لن يساوى النظام هذه النتيجة الخاطئه مع النتيجة التى تخرج صحيحة ،

ولهذا فان النظام يطلق على هذه النتيجة الخاطئه تعبير stderr اختصارا لstandard error .

\* اذا فعندنا الان ثلاث صيغ للتعامل مع الshell

- الاولى هى للمسئول عن ادخال البيانات للshell ويسمى stdin .
- الثانية هى للتعبير عن النتيجة الصحيحة للامر وتسمى stdout .
- الثالثة هى للتعبير عن النتيجة الخاطئه للامر وتسمى stderr .



يلاحظ ان رسائل الخطأ التي تظهر نتيجة لتنفيذ أي امر نوعان

\* الاول عند كتابه الامر بطريقه خاطئه ، فتخبرك ال bash ان الامر خاطى .

```

[ahmed@elnajeeb ahmed]$ lsw
-bash: lsw: command not found
[ahmed@elnajeeb ahmed]$ ls -al
total 6
drwx----- 2 ahmed ahmed 1024 Nov 20 22:15 .
drwxr-xr-x 4 root root 1024 Nov 20 22:08 ..
-rw----- 1 ahmed ahmed 36 Nov 20 22:15 .bash_history
-rw-r--r-- 1 ahmed ahmed 24 Nov 20 22:08 .bash_logout
-rw-r--r-- 1 ahmed ahmed 191 Nov 20 22:08 .bash_profile
-rw-r--r-- 1 ahmed ahmed 124 Nov 20 22:08 .bashrc

```

فما نلاحظه فى المثال الذى امامنا

اولا عندما كتبت الامر lsw وهو طبعا ليس بأمر(ولكنى تعمدت كتابه امر خاطى وذلك باضافه w مع الامر ls) فردت على ال bash انه لا يوجد امر اسمه lsw .  
ثانيا بعد ذلك عندما كتبت الامر ls -al حينئذ قامت ال bash بتنفيذه ، لانها تعرفه، وطبعا نجدها قد عرضت الملفات المخفيه (hidden) والتي تبدأ بال dot .

\* الثانى عند كتابه الامر بطريقه صحيحه ولكن الخطأ اما فى ال option الذى ياتى مع الامر او مع ال argument الذى ينفذ عليه الامر(وغالبا يكون ملف) ففى هذه الحاله يخبرك الامر بنفسه -وليس ال bash- بوجود خطأ .

```

[ahmed@elnajeeb ahmed]$ ls -al najeeb
ls: najeeb: No such file or directory
[ahmed@elnajeeb ahmed]$ _

```

وكما نرى فى المثال الذى امامنا فالملف المسمى najeeb ليس موجود ولهذا فان الذى يحدثنى الان هو الامر ls (لاننى كتبت بطريقه صحيحه) ولكن الخطأ فى عدم وجود الملف .

## ال piping و redirection .

علامة ال piping ( | ) > ستجدها غالبا بجانب ال back space < وشكلها كما هو واضح شرطتين بشكل طولى فوق بعضهما .

الصيغة command | command

وتستخدم ال piping لاستقبال الناتج stdout من الامر الاول ثم ادخاله الى الامر الثانى كstdin وهى من خصائص ال bash التى تستخدم لتنفيذ امرين على التوالى يكون فيهما الامر الاول ذو تاثير على الامر الثانى ، و تستخدم ايضا خاصية ال piping فى فلترة الناتج من الامر .

فلنفترض على سبيل المثال انه يوجد فى ال home dir الخاص بك حوالى 50 او 100 ملف ، ففى هذه الحالة لو نفذت الامر ls -l فستجد الشاشة امامك قد ملئت بالملفات .

اما ال redirection فهى عبارته عن العلامة ( > )

وتستخدم ال redirection لتحويل ال stdout الى ملف ، بدلا من عرضه على الشاشة ، مثل

cat file > other file

فما سيحدث هنا ، هو تحويل الناتج من الامر cat والمتمثل فى عرض الملف file ، الى الملف other file ، مع ان الوضع الطبيعى ان يتم عرض هذا الملف على ال stdout والتى هى الشاشة فى معظم الاحيان .

وقد يتم (عكس) علامة ال redirection ، فقد تكتب هكذا ( < ) وتؤدى نفس وظيفه العلامة الاولى ولكن بالطبع لا بد من تبديل وضع الملفات ، مثل

cat other file < file

وقد تاتى العلامة ( > ) فى الصورة المركبه ، اى ان تكون شكلها كالاتى ( >> ) .

والفرق بين الشكل البسيط والمركب ، يتمثل فى ان

- الشكل البسيط (>) يكتب للملف ، بصرف النظر ، هل بداخله بيانات ام لا ، فان كان فارغا ، فانه يكتب داخل الملف ، وان كان به بيانات فانه لا يبالى ، فيمسحها ويكتب من جديد .
- اما الشكل المركب (>>) فانه يقوم بالكتابه فى الملف ، ولكن بعد المكتوب به ، اى انه لا يمسح المكتوب من قبل .



## الهارد وير Hard Ware

فى هذا الفصل سنتحدث عن الهاردوير بنظرة عامه. هذا ، وان كان لينوكس يعمل على ال hard ware المختلف والخاص بالعديد من السيرفرات server ، والمختلف عن هذا المتعلق بالاجهزه الشخصيه (pc) الا ان هذا الفصل سيتعرض فقط لل hard ware الخاص بالاجهزه الشخصيه .

### **Mother Board**

ال M.board هى اللوحه الاساسيه والرئيسيه لجهاز الكمبيوتر ، بل هى اهم جزء فيه ، فبدونها لا يمكن الاستفادة من بقيه الاجزاء .  
وهى عباره عن لوحه كهربيه كبيره -نوعا ما- يثبت بها العديد من محتويات الجهاز peripheral device ، مثل الهارد ديسك و ال floppy وغيرهم .  
وتقوم البيانات التى تنتقل بين هذه الاجزاء بالمروور من خلال ال mother board الى هذه المحتويات المختلفه .  
وتستمد هذه المحتويات الكهرباء الخاصه بها من هذه ال mother board . والتى بدورها تستمد الكهرباء الخاصه بها من ما يسمى power supply .

(المقصود بال peripheral هى الاجزاء (device) التى تؤدى دور مساعد (Auxiliary) فى النظام وليس دور رئيسى  
\*ايضا تعرف بانها اى شى فى الجهاز ليس ال M.board او ال power supply او الاجزاء الجوهرية purely mechanical (مثل RAM او processor .....).

وتنقسم اللوحه الام mother board من حيث التكامل (integration) الى نوعان

- 1 المتكامله (integrated) وهو هذا النوع الذى يحوى عدد من الاجزاء المبنيه فيه والذى يطلق عليه built-in مثل كارت الشبكه او الصوت .
  - 2 الغير متكامله (Non-integrated) وهى بالتالى عكس الاولى ، فهى لا تحوى اى نوع من الاجزاء بها. بل تضاف اليها هذه الاجزاء عن طريق فتحات التوسعه المسماه slots .
- (المقصود بالتكامل ، تكامل المكونات الاساسيه التى تتركب على اللوحه الام والتى تعد ضروريه لعمل الجهاز)

## الاجزاء المكونه لل Mother board .

ال mother board كما اسلفنا عباره عن لوحه كهربائيه circuit board ولكنها مكونه من العديد من الاجزاء الهامه والتي منها

. ال **data buses** والمقصود بها هي القنوات التي تسير فيها البيانات .  
وهذه ال buses لها سرعات مختلفه ، وكلما زادت سرعه هذه ال bus زادت بالتالى السرعه بين الاجزاء المختلفه الموجوده على ال mother board .

. ال **jumbers** وهى عباره عن اجزاء معدنيه مغلفه بالبلاستيك ، ووظيفتها تتمثل فى توصيل الكهرباء الى (الاسنان) الموجوده بال mother board او الموجوده بالاجزاء المختلفه فى اوضاع معينه ، وكل وضع بالطبع له وظيفه معينه .  
وتفصيل هذه الاوضاع ووظائفها دائما يذكر بالكتيب المرفق لكل جزء من ال hard ware .

. **chipset** وهى الرقاقه المعدنيه المثبتة على اللوحه الام .  
وال chipset عموما هى دائره كهربيه او مجموعه من الدوائر الكهربيه المتكامله والمصغره جدا والموجوده داخل شريحه من السيلكون فيما يسمى IC ، ويوجد منها العديد على اللوحه الام .  
ولكن عدد ال chipset الاساسيه على ال M.board اثنتان . وظيفتهما تحديد خصائص اللوحه الام . من حيث اقصى سرعه وسعه يمكن ان تصل لها الذاكره RAM ، ونوع المعالج processor الذى سيعمل على هذه ال M.board وغيرها من خصائص ووظائف ال M.board .

. ال **slot** وهى هذه الاجزاء التى تشبه الفتحات والتى يوضع بها العديد من الاجزاء device التى يمكن اضافتهم مثل كروت الصوت والمودم والشاشه . ولهذا فهى تسمى فتحات التوسعه .

ويوجد من هذه ال slot عدده انواع

ISA.

(eye-sah) وهى النوع الاقدم من ال slots وتستخدم الان على مستوى ضيق وبالاخص فى الاجهزه القديمه.

وهى اختصار ل Industry Standard Architecture

PCI.

(pee-see-eye) وهى النوع الحديث والمستعمل الان ، والفرق بين ISA و PCI يتمثل فى سرعه نقل البيانات الى ال CPU .

وهى اختصار ل Peripheral Component Interface

AGP .

وهى نوعيه فتحات slot احدث من ال PCI وغالبية استخدامها لكروت الشاشه .

وهى اختصار ل Accelerated Graphic Processors

## الاجزاء الاخرى بال M.board .

بال M.board اجزاء اخرى ولكنها اكثر خصوصيه واستقلاليه من الاجزاء السابقه ومنها

### **CPU -1**

وهو من اكثر الاجزاء اهميه بالجهاز ، لدرجه ان الجهاز ككل قد يسمى باسم ال cpu فيقال جهاز intel 286 او pentium او غير ذلك .  
وهو اختصار ل **Central Processing Unit** ، او وحده المعالجه المركزيه .

ويتكون المعالج من ثلاثه اقسام

- 1- وحده المعالجه المركزيه RAM (سيتم الكلام عنها لاحقا)
- 2- وحده الحساب والمنطق Arithmetic And Logic Unit  
ويتم في هذه الوحده كل الحسابات والمقارنات والقرارات المنطقيه
- 3- وحده التحكم Control Unit  
والغرض من هذه الوحده هو تنظيم سير العمليات داخل وحده المعالجه المركزيه وتنظيم تدفق البيانات بين وحده المعالجه المركزيه ووحدات الادخال والاخراج .

والمعالج عباره عن شريحه صغيره -نوعا ما- من السيلكون بها الملايين من الدوائر الكهربيه الصغيره (الترانزستور)  
وطريقه المعالج في العمل تتمثل في انه يحتوى على timer او ساعه تضبط عمله، وهذا timer يقوم باداء مئات الملايين من العمليات في كل ثانيه ، وكل عمليه من هذه العمليات تسمى (هرتز hertz) ولهذا فالمعالج الذى يطلق عليه 800 MHz يعنى ان هذا المعالج قادر على اداء 800 مليون عمليه في كل ثانيه . (وهذه العمليات ترسل اليه من الذاكره RAM)

### **2- الذاكره Memory**

هناك عدده انواع من الذاكره موجوده بالجهاز

\* اولاً الذاكره العشوائيه وهى ما يطلق عليها ال RAM .  
وهى وحده التخزين الرئيسيه بالنظام (يظن البعض ان وحده التخزين الرئيسيه بالنظام هو الهارد ديسك ، ولكن الصحيح انه وحده التخزين الثانويه بعد ال RAM) .  
ووظيفه ال RAM تتمثل في

التخزين المؤقت للبيانات التى تحول اليها من وحدات الادخال حتى يتم التعامل مع هذه البيانات .  
مكان للتعامل مع البيانات يتم فيه إجراء بعض العمليات الحسابيه والمنطقيه للوصول الى النتائج المتوسطه  
التخزين المؤقت للنتائج المتوسطه لحين ارسالها الى المكان المحدد لها .  
هذا بالاضافه الى مساحه مخصصه لتخزين البرامج التى تقوم باجراء العمليات السابقه على عمليه المعالجه

وتفقد الـ RAM كل محتوياتها بمجرد انقطاع التيار الكهربائي عنها .  
وهي اختصار لـ **Random Access Memory** .

\* ذاكره القراءة فقط . او ما يطلق عليها ROM  
وسميت كذلك لانها فعلا تستخدم لقرأه محتوياتها فقط ، وبالتالي لا يكتب اليها اثناء عمل الجهاز .  
وهذه الذاكره تستخدمها المصانع لكتابه التعليمات على اجزاء الهاردوير والتي يقرأها الجهاز كلما بدأ العمل **start up** .  
واشهر انواع هذه الذاكره هو البرنامج BIOS .  
وتعد اختصار لـ **Read Only Memory** .

\* الذاكره المخبأه cash memory  
سميت كذلك لانها موجوده بداخل الـ **cpu** .  
ووظيفه هذه الذاكره هي تسريع عمليه المعالجه التي يقوم بها الـ **processor** .  
وتقوم بهذه العمليه عن طريق نسخ بعض البيانات من البرامج التي تقوم الذاكره الاساسيه RAM  
بفتحها ، وبذلك توفر الوقت الذي سيستغرقه البرنامج (الموجود فى RAM) لارسال هذه البيانات  
الى المعالج وذلك لانها موجوده بالفعل فى المعالج .  
وحجمها مقارنة مع RAM يعد صغير جدا ، فهي غالبا اقل من 1mega مع الاجهزه الشخصيه .

## كيف يتعامل الهاردوير .

السؤال الآن -بعد هذا الموجز- هو كيف يتعامل الهاردوير مع بعضه البعض .

اولا لا بد ان نعرف ان اهم الاجزاء فى عمليه المعالجه هو ال CPU (والذى يشمل RAM كما اسلفنا) وبالتالي فان كل ال devices لا بد وان تتعامل معه ، ويتم التعامل مع المعالج عن طريق ارسال اشارات بواسطه ال data bus الموجوده بال M.board .

وعندما تصل هذه الاشارات الى المعالج فانه لا بد ان يميز من اين اتت له ، لهذا سنتعرض الى ثلاث طرق فى التعامل مع ال CPU وهم

- I/O -1
- IRQ -2
- DMA -3

### I/O

المقصود بال i/o هي منافذ ال input , output . وهذه المنافذ (ports) موجوده بالذاكره RAM . وتستخدم فى الكتابه لل peripheral المختلفه . وبالنسبه لعددتها فيتجاوز 65000 (خمسه وستون الف) وبالتاكيد لا يستعمل كل هذا العدد بل العشرات منه فقط .

### IRQ

ال irq هي عباره عن عناوين address محدده لكل device، فكارث الصوت له رقم او عنوان والمودم والطابعه كذلك . ومعنى irq هو Interrupts ReQuests والذى يعنى طلبات المقاطعه . فعندما يريد ال device ان يرسل بيانات الى ال cpu فان هذه البيانات تحمل معها رقم هذا ال device لكى يتعرف ال cpu على مرسل هذه البيانات . وبعد تعرف ال cpu على رقم ال device فانه يترك ما يفعله ، يبدأ فى التعامل مع البيانات الجديده ، لان ال devices التى تحمل ارقام irq لها اولويه فى التعامل مع ال cpu .

### DMA

هي اختصار ل direct memory access . وقد يقوم اسمها بتفسير وظيفتها ، فهي تتعامل مع ال memory مباشره ، دون وساطه ال cpu . بمعنى انه لو ان ال irq يتعامل مع ال cpu مباشره ، فان ال dma هذه تتعامل مع ال memory مباشره . ايضا اذا كان بالجهاز 16 رقم محدد لل irq . فان لل dma -باكثر تقدير- حوالى اربعه مسارات الى ال ram .

وتستخدم هذه المسارات لتسريع العمل لبعض ال devices ،مثل ال hard disk .



## التعديل فى الملفات .

للتعديل فى الملفات طريقتان

احدهما عن طريق ال shell ،  
والاخرى عن طريق ال gui .  
ولكن التركيز الان سيكون على التعامل مع الملفات عن طريق ال shell .

واشهر محرر للنصوص text editor فى عالم Unix يعمل على ال shell هو المحرر VI ،  
ينطق vee eye .

وبالنسبه عندنا فى لينوكس فيوجد محرر آخر شهير ، وهو emacs هذا المحرر الذى كتبه  
(ريتشارد ستولمن - مؤسس مشروع gnu) ويعد هذا المحرر هو ال default بعد ال vi . وال  
default عموما للانظمه التى تنتجها gnu . (ذلك لان vi فى الاساس من اليونكس) .

( للتعامل مع الملفات عن طريق ال gui فيمكننا ذلك عن طريق المحرر editor المسمى kedit )

والملفات فى لينوكس اما ان تكون text file اى ملف يمكن قرأته وبالتالي التعديل فيه ، وهذه هى  
غالبية الملفات التى يتعامل معها المستخدم .  
وملفات لا يمكن للشخص ان يقرأها لانها -على سبيل المثال- تتعامل مع النظام .

فبالنسبه للنوع الاول ، وهو الذى يمكن قرأته ، فهذا هو الذى نتعامل معه بالمحرر vi .  
اما بالنسبه للنوع الثانى فإما ان يتعامل معه بالمحرر editor الذى انشئه (وهو غالبا ليس متاح  
للمستخدمين) او ببرامج من نوعه hex editor حيث تقوم بدور هذا المحرر الخاص

(يلاحظ انه قد يكون فى نسخه linux التى تستخدمها ، برنامج اسمه hex editor وهو ذاته الذى نشير اليه) .

\* ولمعرفه نوع الملف الذى نتعامل معه فاننا نستخدم الامر file الذى يخبرنا هل هذا ملف نصى  
text file او ليس كذلك .

وصيغته

file filename

او

file /etc/passwd اذا كان الملف له امتداد .

والامر file له قاعده بيانات data base بها كل انواع الملفات ، ووظيفته تتمثل فى انه يقرأ بدايه  
الملف ثم يقارنها بقاعده البيانات الموجوده عنده ، ثم يخبرك بنوع هذا الملف .

والتركيز على ال vi على وجه الخصوص -سواء فى هذا الكتاب او فى غيره من الكتب- له عده اسباب  
منها

. ان هذا المحرر editor هو الاقدم فى عالم ال Unix على الاطلاق ، ولهذا فلا تتدهش عندما  
تجده فى اى توزيعه من توزيعات لينوكس او يونكس .

. ايضا فى حالات الطوارئ emergency (حاله عدم قيام النظام بالعمل) ففى هذه الحاله وبعدالدخول للنظام فان المحرر ال deafult وقتها هو vi .

واذا تم مقارنة هذا المحرر editor بغيره من المحررات editors الحديثه (مثل تلك المعتمده على الgui) والتي تمتاز -طبعا- بسهوله الاستخدام فاننا سنلاحظ انه صعب فى التعليم ، ولكن كما اسلفنا فلا بد من معرفته ، ثم بعد ذلك فيمكنك التعامل مع الملفات اثناء عمالك اليومى باى محرر ترتاح فى العمل معه.

## البدايه

اسم المحرر vi قادم من Visual Interpreter (البعض يقول انه اختصار Visual editor) . وعند انشاء هذا ال editor كان هناك محررين اخرين editors ولكنهم كانوا يتعاملون مع الملف سطر بسطر او ما يسمى line based ، ولكن من مميزات vi انه يتعامل مع الملف بنظام ال full screen اى انك بإمكانك التحرك فى طول وعرض الشاشة التى امامك .

وللتعامل مع الملف بواسطه vi ما عليك الا كتابه الامر vi على ال shell وتكتب بعد ذلك الملف الذى ستتعامل معه ك argument لهذا الامر ، مثل

```
[ahmed@elnajeeb ahmed]$ vi /etc/passwd_
```

بعد ذلك ستجد ان هناك shell اخرى (child) قد فتحت امامك -مكان الاساسيه- وبها الملف الذى كتبت اسمه سابقا.

ملحوظه

(تكلما سابقا عن الchild shell وانها توصف بذلك لاسباب تتعلق بالتعامل مع النظام ، اما انت كمستخدم فلن تلاحظ ابدا ان الshell التى امامك قد تغيرت)

هذا بالنسبه لفتح الملف عن طريق vi ، اما الان فسنعرف كيف نتعامل مع الملف داخل vi .

## الحاله الاولى command mode .

الان وبعد دخولك الى الملف فانت في اول(مرحلة) mode تقابلها في التعامل مع المحرر vi .  
وهذا المرحلة تسمى command mode .

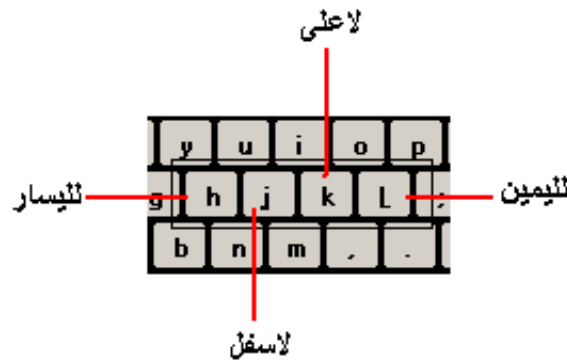
ملحوظه : ساطق على ال mode اسم مرجله او حاله .

\* وسمى هذا ال mode بالاسم command mode لانه بالضبط على اى حرف فان ال vi لن يتعامل معه كحرف ومن ثم يكتبه داخل الملف edit ، بل سيقوم بالتعامل معه على انه امر ذو وظيفه محدده .

بمعنى اخر، هذا ال mode لا يسمح لك مباشره بالتحرير edit داخل الملف ، بل ان التحرير والتعديل داخل الملف هي من وظيفه المرحله الثانيه في التعامل مع المحرر والمسماه insert mode .

## التحرك داخل الملف

**1-** هناك اربعه مفاتيح للاربعه اتجاهات ، وهذه الحروف هي **hjkl**



فالحرف **h** للتحرك ناحيه اليسار  
والحرف **j** للتحرك لاسفل  
والحرف **k** للتحرك لاعلى  
والحرف **l** للتحرك لليمين

وهذه الحروف يمكن بسهوله استبدالها بمفاتيح الاتجاهات



ولكن العله في استخدام المفاتيح **hjkl** هي انه ليس كل انواع ال Unices تدعم مفاتيح الاتجاهات هذه .

**2-** يمكننا الان التحرك بطريقه اخرى داخل الملف .  
هذه الطريقه معتمده على القفز فوق الحروف وليس السير حرف حرف .

ويتم ذلك بالمفاتيح \$ و 0,^ (zero).  
فعلامه الدولار \$ (بالضغط على shift+4) تذهب بك الى نهايه السطر .  
وللرجوع ثانيه الى اول السطر استخدم العلامه ^ (بالضغط على shift+6) او بالضغط مباشره على المفتاح (0) zero .

### **3-** التحرك صفحه لاعلى او لاسفل

بالمفاتيح page up و page down يمكنك تحريك الصفحه باكملها الى اعلى او الى اسفل ،  
ولكن هناك ايضا مفاتيح فى ال main keyboard تقوم بهذه الوظيفه وهى

**f**^ (بالضغط على ctrl+f) تذهب الى اسفل الملف forward صفحه بصفحه .  
**b**^ (بالضغط على ctrl+b) ترجع الى اعلى الصفحه ثانيه backward .

### **4-** التحرك كلمه كلمه

يتم ذلك بالضغط على المفتاح w او e  
فالمفتاح w ينقلك الى اول حرف فى الكلمه القادمه .  
اما المفتاح e فيقف بك على آخر حرف فى الكلمه القادمه .

ستلاحظ ان ال cursor عندما يتحرك كلمه كلمه فانه يقف على العلامات مثل , ( ) - : ويعتبرهم كلمات مستقله ،  
نعم هذه خاصيه فى البرنامج ، فهو يعتبر ان المسافات هى التى تفصل بين الكلمات .  
ومن هذا المفهوم ايضا فان برنامج vi بامكانه فهم الكلمات المتصله ببعضها البعض بواسطه هذه العلامات على  
انها كلمه واحده كبيره bigword مثل you,and.me-are:linux'users . ولجعله يتفاعل معهم هكذا سنستخدم  
نفس المفاتيح السابقه W و E ولكن بصيغه ال capital . فتراه سيقف على اول حرف فى هذه ال bigword وهو  
حرف y ، وعند الضغط W ثانيه سيقفز فوق كل هذه الكلمات ويقف على اول الكلمه التى تلى users .  
حاول تجربه هذه الملاحظه .

### **5-** اول الصفحه واخرها

يمكن الذهاب مباشره الى بدايه الصفحه عن طريق الضغط على gg .  
ونفس المفتاح للذهاب مباشره الى اسفل الصفحه ولكن باضافه الضغط على shift اى G .

## النسخ والقص واللصق .

يمكنك ايضا فى ال cmd mode (وهو ال mode او المرحلة التى لا يمكنك فيها تحرير edit الملف) ان تقوم بنسخ copy سطر او اسطر ، كذلك يمكنك قصهم cut واعاده لصقهم paste فى اماكن اخرى بنفس الملف .

فعند الضغط dd يقوم البرنامج بقص السطر الحالى والاحتفاظ به فى الذاكرة الخاصه المسماه . buffer

ايضا عند الضغط yy فيقوم البرنامج بنسخ السطر الحالى لحين تحديد اين سيتم لصقه paste .

وبالضغط على p يتم لصق السطر الذى تم قصه او نسخه . وذلك فى المكان الذى يقف به ال . cursor

هذا بالنسبه لنسخ او قص سطر واحد ، اما بالنسبه فى حاله وجود عدده اسطر فالامر بسيط ايضا ، وما عليك الا كتابه عدد الاسطر التى تريد نسخها اوصلها قبل dd او yy .

## مثال

امامك الان جزء من ملف /etc/inittab ، ونلاحظ ان ال cursor بجانب كلمه RUN .

كل ما عليك عمله الان هو ان تكتب 7dd (ولاحظ ان ما تكتبه لن يظهر على الشاشة) فما سيفعله برنامج vi هو قص cut السبع اسطر ( بدايه من السطر الذى يقف عليه ال cursor ) الى الذاكرة buffer

```
# Run gettys in standard runlevels
1:2345:respawn:/sbin/mingetty tty1
2:2345:respawn:/sbin/mingetty tty2
3:2345:respawn:/sbin/mingetty tty3
4:2345:respawn:/sbin/mingetty tty4
5:2345:respawn:/sbin/mingetty tty5
6:2345:respawn:/sbin/mingetty tty6
```

بعد ذلك يمكنك الذهاب الى اى جزء بالملف والضغط على المفتاح p ، حينئذ سيقوم البرنامج بوضعهم بالسطر الذى يلى ال cursor .  
اى اذا وقفت على السطر الذى يبدأ بالرقم 6 فى المثال السابق وضغطت p فسيتم لصقهم فى السطر التالى لسطر الرقم 6 .

ونفس الخطوات فى حاله النسخ .

## الغاء كلمه او حرف .

بعد ان تعرضنا لالغاء او حذف سطر عن طريق dd فسنعرض الان لحذف كلمه او حرف .  
فبدايه يمكن الغاء اى حرف عن طريق del .

ولكن كما اسلفنا فان ال vi يعمل دائما باللوحه الام main keyboard ويقوم باداء الوظائف العده  
عن طريق مفاتيح الحروف .

ولهذا فعند اراده الغاء حرف يقع تحت ال cursor (اى ان ال cursor يقف عليه) ما عليك الا الضغط  
على المفتاح X .

ولالغاء حرف يقع قبل ال cursor (اى كانه يرجع للخلف) فبنفس المفتاح ولكن capital اى X .

هذا بالنسبه لما يتعلق بالاحرف اما بالنسبه للكلمات  
فيمكن الغاء اى كلمه عن طريق dw

فان اردت الغاء اكثر من كلمه ، فقط اكتب عدد الكلمات المراد الغائهم مثل 3dw ، لالغاء ثلاث  
كلمات .

## اببدال الاحرف .

يتم ابدال الحروف بالضغط على المفتاح r والذى يقوم بابدال replace حرف واحد فقط  
اما اذا اردت ابدال السطر كله مثلا ، فلتكتب R . فبذلك يتم الابدال لكل الكلمات .

يلاحظ انه بعد الضغط R ما عليك الا كتابه ما تريد وستجد امامك على الشاشة ان الحرف الذى  
يقف عليه ال cursor يتغير بهذا الذى تكتبه على ال keyboard .

## . Undo and Repeat

المقصود ب undo هو عدم فعل اخر خطوه والغائها  
والمقصود ب repeat اى تكرار اخر خطوه

ويمكن تفعيل ال undo عن طريق كتابه الحرف **U** ، حيث يقوم بعدم تطبيق اخر خطوه .  
اما اذا كتبت **U** اى مثل الاولى ولكنها capital فانها تقوم بعدم تطبيق كل الخطوات السابقه .

اما لعمل repeat لآخر خطوه قمت بها ، فيتم ذلك عن طريق ال **> . dot** .

## الحاله الثانيه insert mode .

هذه المرحله mode هي الحاله الثانيه ، وهى التى يمكنك فيها ان تحرر فى الملف (edit) .  
وهذه المرحله لا تحوى اوامر كثيره ، ذلك لانها ببساطه مرحله التعديل داخل الملف ، اى ان معظم ما يُفعل فى هذه المرحله هو ، ادخال بيانات (data) او التعديل فى البيانات الموجوده .

اذا فكل ما يفعله البرنامج vi هو ارشادك الى كيفيه الدخول لهذه المرحله (mode) ، وايضا كيفيه الخروج منها ، مخولا لك ما يحدث بين هاتين العمليتين .

### كيف تدخل الى حاله التحرير insert mode

لا بد لكى تدخل هذه الحاله mode ان تمر بالمرحله الاولى cmd mode .

ويمكن التحول من المرحله الاولى الى ال insert mode بعده طرق ، اسهلهم ان تضغط على المفتاح **i** الذى يعنى insert .

ستلاحظ انه باسفل الصفحه قد ظهر سطر ----- insert ----- والذى ينبهك الى انك فى مرحله التحرير داخل الملف .

بعد هذه الخطوه ، فان المفاتيح كلها ستقوم بوظيفتها التقليديه ، فان ضغطت **h** فستكتب **h** ولن تجعلك تتحرك يسارا .

ولعل السبب فى ان البرنامج ينبهك بالكلمه insert انك فى مرحله التحرير ، هو تعدد الحروف وبالتالي الطرق التى تحولك الى هذه المرحله . لكى لا تعتقد ان حرف **i** هو فقط المسئول عن هذا، فان ضغطت على حرف من هذه الحروف بطريق الخطأ ، نبهك البرنامج الى انك فى مرحله التحرير (editing) .

\* يمكن ايضا استخدام **i** ولكن **capital** . والتى بدورها ستحولك الى ال insert mode ، لكن الفرق بين الاثنين بسيط وهو .

الضغط على **i** small يجعلك تحرر edit قبل ال cursor ، فكلّمه word ان كانت مكتوبه فى الملف وال cursor تحت الحرف **d** ، فحينئذ عند الضغط **i** ستبدأ بالكتابه بعد الحرف **r** وقبل الحرف **d** .

اما فى حال الضغط على نفس الحرف ولكن بصيغه **capital** فحينئذ ستبدأ الكتابه فى بدايه السطر

بعد معرفتنا بالحرف **i** فى حالتيه ال small وال **capital** .  
نتعرف الان على الحرف **a** فى حالتيه ايضا ال small وال **capital** .

فعند الضغط على الحرف **a** ستبدأ عمليه التحرير ولكن بعد ال cursor اى **append** اما بالضغط عليه **A** فستبدأ بالكتابه فى نهايه السطر .

إذا الحرف **i** يكتب قبل ال cursor  
والحرف **a** بعد ال cursor  
أما الحرف **I** فيجعلك تبدأ الكتابة في بدايه السطر  
والحرف **A** في نهايه السطر

وهناك ايضا المزيد من الحالات مثل  
حرف **o** لبدايه سطر جديد قبل ال cursor  
ونفس الحرف **O** اى capital لبدايه سطر جديد بعد ال cursor

والحرف **s** لاببدال الحرف الحالى  
أما الحرف **S** فلابدال السطر الحالى كله

الان تستطيع ان تقوم بالمهام والتعديلات التى تريدها فى الملف وبعد ذلك سيأتى دور الخروج من  
حاله التحرير هذه (insert mode) .

## الخروج من ال insert mode .

يقوم بهذه الوظيفه المفتاح **Esc** .

فبعد الانتهاء من التحرير (editing) وتريد الخروج من ال insert mode والعوده ثانيه الى  
cmd mode ، ما عليك الا الضغط على هذا المفتاح (وستلاحظ بالتأكيد اختفاء كلمه --- insert --- من  
اسفل الشاشة)

الان وقد رجعت الى الحاله الاولى cmd mode يمكنك ان تحفظ ما كتبتّه وتخرج من البرنامج .

وهذه هى وظيفه الحاله الثالثه ال ex mode .

## الحاله الثالثه ex mode .

\* ال ex mode تعد اختصار ل execute mode .

ذكرنا في مقدمه هذا الفصل ان المحرر vi هو الاول والاقدم في عالم ال Unix . وذكرنا ايضا انه كان هناك عده محررين (editors) ولكنهم معتمدين على التعامل مع الاسطر (line based) .

ملاحظه : معنى كونه الاول ، اى اول محرر يمكن الاعتماد عليه وحده بسبب مميزاته .

وهذه الانواع من المحررين تم دمجهم في المحرر vi .  
ولانهم كما اسلفنا معتمدين على السطر الواحد ، فاننا نقوم من خلالهم باداء اعمال لا تتطلب منهم امكانيات عاليه ، مثل الحفظ (save) للبيانات التى كتبت والخروج من البرنامج quit ونحو ذلك .

وقبل الحديث عن هذا ال mode يجب ان نلفت النظر الى

- 1- بالخروج من ال insert mod (الحاله الثانيه) فاننا رجعنا تلقائيا الى الحاله الاولى cmd mode . ولم نذهب الى الحاله الثالثه .
- 2- هذه الحاله وان كانت الثالثه الا انها تستدعى من خلال الحاله الثانيه والاولى ايضا .

## **استدعاء الحاله الثالثه ex mode**

يتم استدعاؤها او بمعنى اخر الدخول اليها عن طريق امر بسيط وهو

كتابه > : < colon .

فبعد كتابه هذا ال colon ستلاحظ انه (ال colon) قد ظهر باسفل الصفحه ، فيمكنك بعد ذلك ان تكتب له الامر الذى تريد منه ان ينفذه .

وغالبية الاستخدام لهذه الحاله هى الاوامر الخاصه بحفظ ما تم كتابته او للخروج من البرنامج vi .  
(اذ انه يعد الحاله الوحيد المتاحه للخروج بطريقه صحيحه من البرنامج) .

ملاحظه ، يمكن التعرف على اوامر الحاله الثالثه هذه عن طريق ادخال اى حرف الى البرنامج بعد كتابه ال colon ثم الضغط على المفتاح tab ، حينها سيقوم البرنامج بعرض الاوامر التى تبدأ بهذا الحرف .

## كيفية الحفظ والخروج من برنامج vi .

لكي تستطيع حفظ ما تم تعديله في الملف ستستخدم الحرف **w** دلالة على write اي تخبر البرنامج ان يكتب هذه البيانات .  
وستلاحظ ان الحرف **w** سيكتب في اسفل الشاشة بجانب ال colon فيكون شكله **w:**

اما لكي تحفظ وتخرج من البرنامج مباشرة ، فستضيف الحرف **q** دلالة على quit ، فيكون **wq:** ثم بعد ذلك ستضغط على المفتاح enter .

وفي بعض الحالات لا يسمح لك البرنامج -لأسباب- ان تحفظ ما كتبتة (save) . ايضا في اوقات اخرى قد لا تستطيع الخروج من البرنامج مع انك لم تامره بان يحفظ (save) ما كتبتة -لسبب وجيه من وجهه نظره- يتمثل في انه احس بوجود تغيير في شكل الملف ، مع انك لم تحفظ هذا التغيير بل رجعت فيه وبالتالي رجع الملف الى شكله حين تم فتحه ، ولكن يظل احساس البرنامج انه قد حدث تغيير.

لهذه الاسباب وغيرها قد تضطر الى إجباره على حفظ ما قمت بتعديله save . وايضا اجباره على الخروج من البرنامج دون حفظ (un save) .

ويتم ذلك عن طريق وضع علامه التعجب ! بعد الحرف w او الحرف q . فيكون الشكل كالآتي

**w!** للحفظ      ويكون **q!** للخروج دون حفظ

(يفترض طبعا ان يكون كل حرف منفرد لانهم لو اجتمعوا معا - حرف w و q - اذا فلن تكون هناك مشكله ، فسيحفظ البرنامج التعديلات وسيخرج)



## file permission ال

تعد من خصائص انظمه التشغيل التى توفر خاصيه تعدد المستخدمين خاصيه ال permission  
والتي تتحدد على الملفات والمجلدات .

والpermission تعنى نفس معناها الحرفى وهو التصريح اوالتصاريح المسموح بها .  
فكل ملف او مجلد ينشأ فى النظام يتحدد له تصريح يحدد الصلاحيه الممنوحه لكل انواع  
المستخدمين > منشأ الملف، المجموعه التى ينتمى اليها ، بقيه المستخدمين الموجودين فى النظام<

ويتحكم فى تحديد هذه الصلاحيات مالك الملف -اى من قام بانشائه- وايضا مدير النظام وهو ال  
. root

ويمكن استعراض هذه التصاريح بالامر ls -l

```
ahmed@elnajeeb ahmed]$ ls -l
total 1
drwxr-xr-x  2 ahmed  ahmed    1024 Jan 23 12:55 ahmed
-rw-r--r--  1 ahmed  ahmed       0 Jan 23 12:55 ahmed1
```

فنجد ان الناتج من الامر عبارته عن مجلد <باللون الازرق> وملف <باللون الابيض> ، ولكن ما  
يهتمنا فى هذا المثال هى هذه الحروف المكتوبه باول السطر ، فهى التى تبين التصاريح الممنوحه  
لكلا من المجلد والملف .

فنجد فى بدايه سطر المجلد ahmed ذو اللون الازرق قد كتب حرف d والذى يوضح ان هذا  
عباره عن مجلد directory ، <وحرف d هذا ليس جزءاً من التصاريح permission> ثم  
تتبعه هذه الحروف rwx ثم يتكرر الحرفان r-x فقط بدون الحرف w والذى تم وضع مكانه  
العلامه <-> ثم يعاد تكرار الحرفان r وx لثالث مره .  
يلاحظ انه سيتم شرح معانى هذه الحروف وسبب تكرارها فى السطور القادمه فلا تدعها تزعجك الان.

اما السطر الثانى فيحوى احرف اقل من السطر الاول ، فنراه بدايه لم يكتب حرف d بل وضع  
مكانه العلامه - والتي تعنى ان هذا السطر خاص بملف ،  
ثم بعد ذلك نراه قد كتب -rw ثم اعاد الحرف r فقط لمرتان مع وضع علامه ال dash او الشرطه  
مكان الاحرف التى لم يكتبها .

هذا باختصار شكل ال permission او التصاريح . والان سنبدأ فى فهم معنى هذه الحروف .

## فهم ال permission .

لكي يتم فهم ال permission بطريقة بسيطة يمكننا ان نقسمه الى قسمين  
• الشكل البسيط  
• والشكل المركب

1- الشكل البسيط لل permission وهو مكون من ثلاثة احرف مثل rwx وتفصيلهم كالآتي  
• حرف r للدلالة على read اى امكانيه قراءه الملف  
• وحرف w للدلالة على write اى امكانيه الكتابه فى الملف  
• وحرف x للدلالة على execute اى تنفيذ هذا الملف وكأنه برنامج

2- اما الشكل المركب لل permission فهو تكرر هذا الشكل البسيط ثلاث مرات ، مثل  
rwx rwx rwx

والسبب فى هذا التكرار هو ان النظام يصنف المستخدمين عنده الى ثلاث فئات  
- مالك الملف او المجلد  
- المستخدمين المشتركين مع هذا المالك فى مجموعه واحده  
- بقيه المستخدمين بالنظام

ثم بعد ذلك يحدد النظام التصاريح المحدده لكل فئة .  
بمعنى اخر ، ان كل فئة لها تصاريح خاصه بها ، فمالك الملف له rwx خاصه به تحدد صلاحياته  
على الملف/المجلد ، ومجموعته لها rwx تحدد ايضا الصلاحيات المسموح لهم بها ، وبقية  
المستخدمين بالنظام لهم ايضا rwx تحدد صلاحياتهم .

فاذا نظرت لكل فئة على حده فهذا هو الشكل البسيط لل permission ، واذا نظرت الى الثلاث  
فئات مره واحده فهذا هو الشكل المركب لل permission والذي تم عرضه بالامر ls سالف  
الذكر .

## الاختلاف بين الملف والمجلد فى التصاريح .

يختلف `rwX` اى التصريح بالقراءة والكتابة والتنفيذ من الملف الى المجلد .  
ذلك لان طبيعه المجلد مختلفه عن طبيعه الملف ، فبينما تستطيع الكتابة (`edit`) فى ملف الا انك لا تستطيع ان تكتب (`edit`) فى مجلد . (لانه و كما هو معروف ، فان الملف يحوى بيانات بينما المجلد يحوى ملفات ومجلدات وليس بيانات)

لهذا فان التصريح بالكتابة فى المجلد له معنى مختلف عنه فى الملف وهكذا الحال مع القراءة والتنفيذ (`write` و `execute`) وهذه هى معانى التصاريح بالنسبه للمجلدات .

- \* فالتصريح بالقراءة او `r` تعنى عرض محتويات المجلد
- \* والتصريح بالكتابة او `w` تعنى انشاء عناصر `items` <ملفات ومجلدات> داخل المجلد
- \* اما التصريح بالتنفيذ او `x` فيعنى التحرك والوقوف عليه بامر مثل `cd`

يتضمن ايضا التصريح ب `x` عرض محتويات المجلد ، ولكن الفرق بينه وبين `r` يتمثل فى عند ازاله `r` من التصريح فان المجلد لا يعرض محتوياته على الاطلاق ، اما عند ازاله `x` فان المجلد يعرض محتوياته ولكن بدون تفصيل . (اى تراه قد عرض لك الملفات والمجلدات الموجوده داخل المجلد دون ذكر اى معلومات عنها).

## ال permission بنظره اخرى .

يمكن تغيير ال permission وتعديله بحسب الحاجه الى ذلك ، فقد يمنع مالك الملف مثلا اى شخص من ان يقرأ او يعدل فى الملف الخاص به ، ويقتصر امر التعديل والقراءة عليه هو فقط ، ومن ثم سيكون التصريح اوال permission على الصورة التاليه --- rwx .

واذا اراد ان يعطى المستخدمين المشتركين معه فى نفس المجموعه صلاحيات فسيتحول شكل ال permission الى الشكل التالي --- rwx rwx . (على فرض ان التصريح لهم سيشمل القراءة والكتابة والتنفيذ للملف)

ونفس الامر مع المستخدمين الاخرين الباقين فى النظام .

فمالك الملف او المجلد بالاضافه الى ال root لهما الحق فى وضع الصلاحيات المختلفه .

ولكن قبل الكلام عن كيفية تعديل ال permission فلا بد من التعرض للامر **stat** .

فالامر **stat** هو امر مهم لانه يعرض تفاصيل عن المجلد او الملف اكثر مما يعرضه الامر **ls** ، هذا من ناحيه ، ومن ناحيه اخرى – وهى الاهم- انه يعرض ال permission بطريقة الارقام و الحروف وليس الحروف فقط مثل الامر **ls** ، وطريقه الارقام هذه قد تكون اسهل فى فهم ال permission من طريقه الحروف .

والان سنعطى مثالا للامر **stat** ثم نتحدث بعد ذلك عن ناتجه

```
[ahmed@elnajeeb ahmed]$ ls -l
total 1
drwxr-xr-x  2 ahmed  ahmed    1024 Jan 23 13:54 ahmed
-rw-r--r--  1 ahmed  ahmed      0 Jan 23 12:55 ahmed1
[ahmed@elnajeeb ahmed]$ stat ahmed
  File: 'ahmed'
  Size: 1024          Blocks: 2          IO Block: 4096   Directory
Device: 303h/771d    Inode: 7777          Links: 2
Access: (0755/drwxr-xr-x)  Uid: ( 500/  ahmed)  Gid: ( 500/  ahmed)
Access: 2005-01-23 13:54:15.000000000 +0200
Modify: 2005-01-23 13:54:15.000000000 +0200
Change: 2005-01-23 13:54:15.000000000 +0200

[ahmed@elnajeeb ahmed]$ stat ahmed1
  File: 'ahmed1'
  Size: 0             Blocks: 0          IO Block: 4096   Regular File
Device: 303h/771d    Inode: 3896        Links: 1
Access: (0644/-rw-r--r--) Uid: ( 500/  ahmed)  Gid: ( 500/  ahmed)
Access: 2005-01-23 12:55:29.000000000 +0200
Modify: 2005-01-23 12:55:29.000000000 +0200
Change: 2005-01-23 12:55:29.000000000 +0200
```

فالاولا تم تطبيق الامر **ls -l** وناتجه مثل ما نرى عبارته عن مجلد وملف ، ثانيا تم تطبيق الامر **stat** على المجلد **ahmed** ، فاخرج لنا كما نرى العديد من المعلومات ،

ولكن ما يهمنا الان من هذه المعلومات هو ال permission والموجود بجانب Access على اليسار ، فنراه شبيها لذلك المذكور مع الامر -l ls إلا انه مضاف اليه رقم على يساره وهو 0755 ونفس المساله مع المثال الذى يليه والمطبق على الملف ahmed1 .

فماذا يعنى هذا الرقم ؟  
هذا الرقم هو عبارته عن ال permission ولكن فى صورته رقم .

فامامنا الرقم 0755 (ال0 الموجود على اليسار له معنى -بعكس ما تعلمناه فى المدارس- ولكن لن اتعرض له الان لان استخدامه قليل)  
هذا الرقم (755) عبارته عن الثلاث انواع من المستخدمين الموجودين بالنظام ، فمن اليسار الى اليمين (مالك الملف ، المجموعه ، الاخرين) ، ولكن تم تبديل القيمه التى تمثلها الحروف rwx الى رقم .

كيف يتم ذلك ؟  
يتم ذلك باعطاء الحرف r رقم 4  
واعطاء الحرف w رقم 2  
واعطاء الحرف x رقم 1  
اذا فان مجموع rwx > وهو التصريح الخاص بالمالك فقط هو مجموع 4+2+1 اى 7 وهو الرقم الموجود على اليسار فى المثال السابق .

مره اخرى .  
الثلاث ارقام المذكوره وهى 755 هى بديله عن r-x r-x rwx اى ان الثلاث ارقام تعد الصوره المركبه لل permission والتى سبق التحدث عنها . اما الصوره البسيطه له فهى

- 1- بالنسبه لمالك الملف/المجلد فان المسموح له هو القراءه r ، والكتابه w ، والتنفيذ x وبتحويل هذه الاحرف الى ارقام فان ال permission المسموح له به سيكون 4 و 2 و 1 اى 7 .
- 2- بالنسبه لمجموعته فالمسموح لهم به هو القراءه r والتنفيذ x فقط اى 4 و 1 فالمجموع 5 .
- 3- بالنسبه للآخرين فهو نفس الشئ مع المجموعه .

واستخدام هذه الارقام يعد اسهل -بعد التعود عليها- من استخدام الاحرف وبالاخص عند تعديل ال permission كما سنرى .

ولكن قبل الانتقال الى كيفية تعديل ال permission فهناك معلومه مفيده ، الا وهى ، ان هذه الارقام لا يمكن ان يحدث بينها تعارض .  
فعند جمع اى رقمين من هؤلاء الثلاثه لا يمكن ان ينتج عنه قيمه الرقم الثالث ابدا ، فاذا كان ال permission قرأه وتنفيذ فسيكون الرقم 5 .  
وان كان كتابه وتنفيذ فسيكون الرقم 3 وهكذا . وهذه قائمه باحتمالات الارقام كلها

- . الرقم 1 للدلاله على التنفيذ فقط اى x
- . الرقم 2 للدلاله على الكتابه فقط w
- . الرقم 3 للدلاله على w و x
- . الرقم 4 للدلاله على القراءه فقط r
- . الرقم 5 للدلاله على r و x
- . الرقم 6 للدلاله على r و w
- . الرقم 7 للدلاله على r و w و x

## تعديل ال permission .

يتم تعديل وتغيير ال permission اى الصلاحيات بالامر **chmod** . وصيغته كالاتى

`chmod <mode> file`

وال `mode` المقصود فى هذا المثال هو ال permission .

ويمكن للامر `chmod` ان يتعامل مع ال `mode` فى صورته الحروف او فى صورته الارقام

وقبل ضرب مثال على ذلك لا بد من معرفه معلومه هامه جدا وهى

ان الملف عند انشائه لا يتعدى ال permission الخاص به الرقم 6 سواء كان للمالك اولغيره ، اى ان ال `default` له انه ليس `executable` . فان اراد مالكة او ال `root` جعله `exe` اى جعل الملف يعمل كبرنامج فانه يقوم بتعديل ال permission بنفسه بالامر `chmod` .

### 1- تعديل ال permission بالارقام .

نفترض اننى املك ملف يسمى `perm` ، وهو ذو permission عبارته عن 644 (`rw- r-- r--`) واريد تعديل ال permission لجعل هذا الملف `executable` اى اضافته `x` للمالك ، ولاننى مالكة فببساطه سانفذ الامر التالى

`chmod 744 perm`

فببساطه اعدت صياغه ال permission من جديد ولكن بعد اضافته الرقم 1 (وهو البديل للحرف `x`) الى تصريح المالك الذى كان فى الاساس 6 .

خطوه اخرى ، وفيها سيتم منح المجموعه تصريح الكتابه اى التعديل فى الملف والمتمثل فى `w` ، فالمجموعه لها تصريح قراءه فقط اى 4 وما اريد فعله هو تعليقه هذا التصريح الى 6 ليكون قراءه وكتابه وبالتالي سيكون شكل ال `mode` الجديد هو

`chmod 764 perm`

ويمكن تنفيذ الامر `ls -l` او `stat` قبل وبعد عمليه تغيير ال permission لملاحظه الفارق .

## 2- تعديل ال permission بالحروف .

صيغه هذا التعديل مثل الصيغه الاولى ، الا انه بدلا من كتابه ارقام فسيتم كتابه الاحرف الداله على ال permission وهى r او w او x .  
ولانه يوجد اكثر من نوعيه مستخدم بالنظام (3 فئات كما ذكرنا) فلا بد من تحديد الفئة التى سينفذ عليها هذا ال permission .

اذا ، عند كتابه ال mode فانه يكتب بمثل هذه الصيغه  
u=rwx هذا بالنسبه للمالك اى ال user  
والمجموعه يكتب g=rwx  
ولبقية المستخدمين يكتب o=rwx ، اختصارا ل other

فيكون شكل الامر كالتالى

chmod u=rx perm

وبذلك نكون قد حددنا صلاحيه المالك بالقراء والتنفيذ ، ونفس القاعده فى التعامل مع الفئتين الاخرين .

\* اما اذا اردت تحديد ال permission للجميع (الثلاث فئات) بشكل متساوى ، اى ان ياخذ كل المستخدمين نفس ال permission ، ففى هذه الحاله ستستخدم الحرف a للدلاله على all ويكون شكل الامر كالاتى

chmod a=rx perm

\* بقى ايضا شى اخير وهو متعلق بالعلامه الفاصله بين نوعيه المستخدم وال permission ، فليس فى كل الحالات نقوم باستخدام العلامه (=) بل يتم ايضا استخدام (+ او -) فان اردنا زياده ال permission لمستخدم معين فاننا نستخدم علامه + ، اما اذا اردنا انقاص ال permission فنستخدم العلامه - وهكذا .

## اوامر متعلقه بال permission .

- يوجد ثلاث اوامر متعلقه بال permission وهم
- 1- umask وذلك لتحديد ال permission لكل الملفات عند انشائهم من البدايه .
  - 2- chown وذلك لتغيير ملكيه الملفات لمستخدم او لمجموعه اخرى .
  - 3- chgrp لتغيير ملكيه الملف لمجموعه اخرى .

### **umask** . اول الامر

لعلنا لاحظنا عند الكلام عن تعديل ال permission وذلك باستخدام الامر chmod (الذي يعد اختصار ل **change mode**) ان هذا الامر يغير ال mode الخاص بال permission ، مما يعنى ان ال permission فى الاساس موجود .

ولعل الامر umask يوضح لنا هذه النقطه، فبكتابه وتنفيذ الامر منفردا يظهر لنا ناتج ، هذا الناتج عباره عن رقم ، وهذا الرقم يوضح لنا ال permission الذى يحدد تصاريح الملفات الجديده التى سوف تُنشأ .

بمعنى اخر ، الملفات الجديده التى ستنشأها لا داعى معها ان تعدل ال permission الخاص بها بعد انشائها اذا كنت قد ضبطت هذا ال permission من البدايه بالامر umask .

دعنا الان نرى الرقم الذى سيظهر ومعناه ، ثم بعد ذلك نرى كيف يمكننا ضبط ال permission بالنسبه للملفات التى ستنشأ فى المستقبل .

فقط اكتب الامر umask وسترى ان الناتج منه مثل هذا الرقم

0022

هذا الرقم كما اسلفت يوضح لنا ال permission الحالى ولكن بطريقه عكسيه . اى ان ال permission الحالى ليس 022 ، ولكنه 755.

فالامر umask يعمل عن طريق طرح هذا الرقم (22) من الرقم ال default لل permission والرقم ال default لل permission هو 777 للمجلدات ، و 666 للملفات . وذلك يعنى ان ال permission الحالى هو 755 للمجلدات ، و 644 للملفات .

فلو كان ناتج الامر umask على نظامك هو نفس الناتج المذكور هنا ، فيمكنك الان انشاء ملف جديد ثم عرض ال permission الخاص به ومن ثم ستجده r-- r-- rw- اى 644 .

## ضبط ال permission

يمكننا ايضا بالامر umask تهيئه ال permission للمجلدات والملفات الجديده بحيث تكون مقترنه بال permission المطلوب وقت انشاءها ومن ثم فلا داعى لاجراء عمليه chmod على كل ملف على حده بعد انشاءه.

يتم ذلك عن طريق اضافه ال permission الى الامر umask ولكن كما رأينا بطريقه عكسيه.

فان اردت تصريح قيمته 733 مثلا فيكون شكل الامر كالتالى

umask 0044

وان كان التصريح الذى تريده 422 فسيكون الامر بالصيغه التاليه

umask 355

الان يمكنك ضبط ال permission الذى تريده على نظامك للملفات التى ستنشأ فى المستقبل ، ولا تجعل الامر umask وطريقته المعكوسه تقلقك الان ، فالمسأله مساله وقت ، ثم ستجد نفسك قد ألفت طريقته فى العمل .

## ثانيا الامر **chown** .

وظيفه الامر chown هى تغيير ملكيه الملف الى مستخدم اخر او ايضا الى مجموعه اخرى .

وصيغه الامر هى

`chown <newuser> file...`

`chown <newuser:newgroup> file...`

فالصيغه الاولى تقوم بتغيير ملكيه الملف/الملفات فقط الى مستخدم جديد .  
اما الصيغه الثانيه فتقوم بتغيير ملكيه الملف والمجموعه ايضا .

والفرق بين الصيغه الاولى والثانيه هو ،الصيغه الاولى عندما تنتقل ملكيه الملف فان مجموعه الملف الاولى تبقى كما هى لا تتغير . فيتغير المالك ولا يتغير الملف الجديد الى مجموعته بل يظل فى المجموعه السابقه .

اما الصيغه الثانيه فتنتقل الملكيه وايضا العضويه فى المجموعه . وقد تكون هذه المجموعه الجديده هى ذاتها نفس مجموعه المالك الجديد او تكون مجموعه مختلفه عنه تماما .  
ويفصل فى هذه الحاله بين المالك الجديد والمجموعه الجديده بال **colon** او فقط بال **dot** .

## ثالثا الامر **chgrp**

يعد الامر **chgrp** مرادف للامر **chown** عندما يستخدم هذا الاخير فى تغيير المجموعه ، فيمكن الاستعانه به فى تغيير المجموعه للملف/الملفات المعينه ، او فقط الاكتفاء باستخدام الامر **chown**

وصيغته هى

`chgrp <newgroup> file...`



## ال fs وال mount .

### شكل ال file system

لعل من الخواص المميزه ل لينوكس -او لانظمه يونكس عموما- هي طريقه عمل ال file-system الخاص به والتي نعرفها بال mount .

فال يونكس او ال لينوكس له طريقه عمل مختلفه تماما عن التي اعتدناها مع ويندوز بالنسبه لل file system ، وتتمثل في

للنظام ككل شكل مميز ، شبيه بالشجره وافرعها ، فكما ان للشجره جذر رئيسي وتتفرع منه بقية الافرع ، ايضا ال file system الخاص ب يونكس (والانظمه المتوافقه معه ك لينوكس) له جذر رئيسي ويتفرع منه بقية الاجزاء المكونه للنظام .

فالجذر الرئيسي في اليونكس هو ال / ولهذا يسمى ال root ، وبقية الاجزاء تتدلى من هذا الجذر وملحقه به مثل /etc او /boot وغيرهم ، بمعنى ان ، هذا الفرع له اسم ولكن يسبقه -كما نرى- ال / .

والمسأله ليست جذر وفرع واحد فقط ، بل ان هذا الفرع يتدلى منه ايضا افرع اخرى ولكن يختلف عددها من (فرع) لآخر .

### عملية ال mount

يتمتع نظام يونكس بخاصيه اخرى متعلقه ايضا بال file system وهي عملية ال mount . فالنظام ككل (اي الشجره بفروعها) ليس كتله او قطعه واحده ، بل انه عباره عن اجزاء ، وكل هذه الاجزاء متصله بالجذر الرئيسي للنظام وهو ال / .

وعملية الاتصال او بلفظ ادق الالتحاق بال / (root) ، تسمى عملية ال mount .  
وايضا عملية انهاء هذا الالتحاق بال / (root) ، تسمى unmount .

هذا من ناحيه اسم العمليه ووظيفتها ، ولكن كيف تتم هذه العمليه ؟

يمكننا تقسيم النظام -من جهه اتمام هذه العمليه- الى قسمين

1- قسم تتم له عملية ال mount او اللاحق بال root تلقائيا ، اي يقوم النظام بهذه العمليه بنفسه  
واثناء التحميل booting ، ويتم هذا مع كل ال file system الذي تم تخصيص له partition  
اثناء تنزيل النظام مثل ال var و tmp و usr وغيرهم .

2- وقسم تتم له عملية ال mount بواسطة المستخدم او المتعامل مع النظام ، ويتم هذا غالبا مع  
الاجزاء التي تتركب وتنزع من النظام مثل ال cd وال floppy وما شابههم ويطلق عليهم  
removable device .

فالقسم الاول محدد له معلومات في ملف معين ، يذهب النظام لقرأته وتنفيذ الامر mount علي  
كل من فيه كلما قمنا بتشغيل النظام .

وهذا الملف ينشأ مع الملفات التي يقوم النظام بانثائها اثناء التنزيل ويسمى **fstab** ، اختصارا ل **file system table** .

والقسم الثانى المذكور ايضا فى الملف **fstab** ولكن لانه يركب وينزع من النظام باستمرار فان له تعامل خاص يتمثل فى ان المستخدم يتولى عمليه الحاقه بنفسه بالنظام وذلك ايضا بالامر **mount** .

السبب فى ان النظام لا يلحق هذا القسم الثانى بال / (مع انه مذكور فى الملف **fstab** ) هو كما ذكرنا ان هذا القسم ينزع ويركب ، فلو حاول النظام الحاقه مع بقية ال **fs** ولم يجد اى **media** داخله فسوف تحدث مشكله للنظام ، تتمثل فى

1- يتوجب على النظام الحاقه نظريا ،لانه مذكور فى الملف **fstab**  
2- ومن ناحيه اخرى عدم امكانيه الحاقه عمليا بسبب عدم وجود **media** بداخل ال **cdrom** او ال **floppy** او غيرهم من ال **removable device** > فالوضع الطبيعى لهذه الاجزاء خلوها اثناء قيام النظام من اى **media** < ولتجنب مثل هذه المشاكل فان النظام (اوكل) او فوض المستخدم فى اجراء عمليه ال **mount** هذه بالنسبه لهذه الاجزاء القابله للنزع **removable device** ، فان اراد المستخدم استعمالها وكانت هناك **media** موجوده بالفعل فى هذه الاجزاء القابله للنزع فانه يقوم بربطها بال **fs** . وان لم توجد **media** فلا يحدث للنظام اى مشكله عند بدايه التشغيل .

اذا فعندنا الان **fs** متعدد يتم ربطه والحاقه بالجذر الرئيسى .  
ويتم اللاحاق بعمليه - او بمعنى اخر بامر- ال **mount** .  
وهناك ملف به معلومات تحدد طريقه عمل الامر **mount** واسم الملف هو **fstab** .

وقبل بدايه شرح الامر **mount** والملف **fstab** ، بقيت جزئيه اخيره ، وهى شكل هذا ال **fs** بعد عمليه ال **mount** .

وبالاجابه على هذا التساؤل سنجد انفسنا امام مصطلح اخر الا وهو ال **mount point** .

فال **mount** هو الجزء المحدد فى ال **hard ware** مثل ال **partition** .  
اما ال **mount point** فهو اسم هذا ال **partition** والذى تتعامل انت معه مثل **var** , **etc** , وغيرهم . وهو عباره عن مجلد .  
وهو بالضبط المجلد الذى تراه وتتعامل معه من خلال ال الشاشة ال **graphical** او **gui** .

اذا هناك **mount** وهناك **mount point** .  
الاول هو ال **device** والثانى عباره عن مجلد .  
ويقوم الامر **mount** بالربط بين هذين الجزئين .

## الملف fstab .

ملف fstab يوجد تحت المسار /etc/ أى ان مساره بالكامل هو /etc/fstab وشكله كالتالى

| 1           | 2           | 3       | 4                   | 5 | 6 |
|-------------|-------------|---------|---------------------|---|---|
| LABEL=/     | /           | ext3    | defaults            | 1 | 1 |
| LABEL=/boot | /boot       | ext3    | defaults            | 1 | 2 |
| none        | /dev/pts    | devpts  | gid=5,mode=620      | 0 | 0 |
| LABEL=/home | /home       | ext3    | defaults            | 1 | 2 |
| none        | /proc       | proc    | defaults            | 0 | 0 |
| none        | /dev/shm    | tmpfs   | defaults            | 0 | 0 |
| /dev/hda2   | swap        | swap    | defaults            | 0 | 0 |
| /dev/fd0    | /mnt/floppy | auto    | noauto,owner,kudzu  | 0 | 0 |
| /dev/cdrom  | /mnt/cdrom  | iso9660 | auto,owner,kudzu,ro | 0 | 0 |
| ~           |             |         |                     |   |   |
| ~           |             |         |                     |   |   |
| ~           |             |         |                     |   |   |

وكما نرى فان الملف عبارته عن ستة اعمده ، وترتيب هذه الاعمده اجمالا هو

- 1- اسم ال device
- 2- اسم المجلد المربوط بال device
- 3- نوع ال file system
- 4- الخيارات او options
- 5- معلومات خاصه لل dump
- 6- معلومات خاصه لل fsck

اما ما تعنيه هذه الاعمده تفصيلا فهو كالاتى

1- اسم ال device هو الجزء المحدد فى ال hard ware مثل ال partition المختلفه .  
وهذا ال device يسمى ايضا بال mount .

2- اسم المجلد وهو بمعنى اخر ، النافذه التى من خلالها نستطيع ان نرى ونتعامل مع ال device ، فال partition على سبيل المثال -ايا كان نوعه- يحوى data او بيانات ، ولكى نستطيع التعامل مع هذه ال data الموجوده بالفعل على الهارد ، فلا بد من تحديد نافذه لها .  
ويسمى هذا المجلد -او النافذه- بال mount point .

والمكان الافتراضى default لل device التى تتركب وتنزع باستمرار (removable) هو /mnt ولهذا فاننا نرى ان ال cdrom و floppy موجودان تحته .

3- نوع الملفات او file system type وهو كما نرى فى المثال السابق مقاطع مثل , ext3  
swap , proc , udf وغيرهم ، وسيتم التحدث عنهم عند الكلام على خيارات الامر mount .

#### 4- الخيارات او ال options

ويوضع بهذا العمود الخيارات التي تريد ان تحدد لها ال device ليعمل على اساسها .  
وسيتم شرح هذه الخيارات عند الحديث على الامر mount (وبالاخص عند الحديث عن النوع الثاني  
من الخيارات التي تأتي معه) .

#### 5- العمود الخاص بالامر dump

<dump هو البرنامج المسئول عن عمل ال backup فى لينوكس ويونكس ايضا>

ومعنى هذا ان الامر dump عند استدعاؤه لعمل backup على ال file system فانه يذهب  
الى ملف fstab ويعرف منه كيفية التعامل مع هذا ال file system ، ولهذا فاننا نراه ياخذ قيم  
عباره عن ارقام ( رقم 1 او رقم 0 )

فالرقم 1 دليل على ان هذا ال file system خاص ب لينوكس وبالتالي عمل backup له .  
اما الرقم 0 فيدل على ان هذا ال file system ليس لينوكس وبالتالي لا يحتاج اجراء backup  
عليه .

#### 6- العمود الخاص بالامر fsck

<الامر fsck هو المسئول عن عمل file system check بعد كل عمليه reboot>

فيعرف النظام من خلال ملف fstab ترتيب ال file system فى عمليه ال check ، وتحدد  
ارقام ايضا فى هذا العمود ( 1 و 2 و 0 )

فالرقم 1 يتحدد فقط لل / او ال root partition

والرقم 2 يتحدد لبقية ال partition الخاصة ب لينوكس مثل boot و home

اما الرقم 0 فلبقية ال partition والتي لا تحتاج لعمليه fsck

اذا فعندما يقوم النظام بعمل fsck فانه يقوم باجراء هذا الامر اولا على ال root partition ثم  
يتبعه ببقية ال partition .

## الامر mount .

وظيفة الامر mount هي ربط الجزء المحدد من ال hard ware الى مجلد .  
وله ثلاث صيغ ، وهم

- mount [option] directory -1
- mount [option] device -2
- mount [option] device directory -3

فالصيغه الاولى والثانيه تستخدم اذا كان اسم ال device موجود فى ملف fstab .  
اما الصيغه الثالثه فتستخدم اذا لم يكن هناك اى ذكر لل device فى ملف fstab .  
وسنعطى امثله للثلاث حالات .

وهذا هو شكل ملف fstab

| 1           | 2           | 3       | 4                   | 5 | 6 |
|-------------|-------------|---------|---------------------|---|---|
| LABEL=/     | /           | ext3    | defaults            | 1 | 1 |
| LABEL=/boot | /boot       | ext3    | defaults            | 1 | 2 |
| none        | /dev/pts    | devpts  | gid=5,mode=620      | 0 | 0 |
| LABEL=/home | /home       | ext3    | defaults            | 1 | 2 |
| none        | /proc       | proc    | defaults            | 0 | 0 |
| none        | /dev/shm    | tmpfs   | defaults            | 0 | 0 |
| /dev/hda2   | swap        | swap    | defaults            | 0 | 0 |
| /dev/fd0    | /mnt/floppy | auto    | noauto,owner,kudzu  | 0 | 0 |
| /dev/cdrom  | /mnt/cdrom  | iso9660 | auto,owner,kudzu,ro | 0 | 0 |

فالعمود الاول (كما سبق ان ذكرنا) موجود به ال device ونراه مكتوب /dev/cdrom .  
والعمود الثانى موجود به ال directory ولهذا فاننا نراه تحت المجلد /mnt

ولتطبيق الامر mount على ال directory (الصيغه الاولى) ، نكتب الى ال shell هذا الامر

mount /mnt/cdrom

فما تفعله ال shell هو انها تذهب الى الملف fstab وتسأله معلومات عن هذا ال directory  
وبالتالى سيخبرها ان هذا ال directory مختص بال device المسمى /dev/cdrom .

ونفس الوضع مع الصيغه الثانيه والتي تكتب كالتالى

mount /dev/cdrom

فتحدث نفس العمليه الاولى ولكن مع ابدال المعلومه التى ستسأل عنها ال shell الملف fstab .

اما الصيغه الثالثه فتستخدم فى حاله عدم وجود اى معلومات عن ال device ولا عن ال directory او بلفظ اكثر دقه ال mount و mount point .

ولهذا فاننا ندخل الى النظام كلتا المعلومتين .

ويمكن اعطاء مثال على هذه الحاله ، بال partition الخاصه باى نظام تشغيل اخر مشترك مع لينوكس فى نفس الهارد ديسك ، مثل ويندوز على سبيل المثال .

(فالوضع الطبيعى وال default عند اشتراك لينوكس مع اى نظام تشغيل اخر على هارد ديسك واحد ، ان لا يستطيع لينوكس ان يرى ال partition الخاصه بهذه الانظمه الاخرى الا بعد عمل mount لهذه ال partition)

فالذى سنقوم به الان عبارته عن خطوتان الاولى سنستعرض كل ال partition الموجوده على نظامك (لينوكس و ويندوز وغيرهم اذا وجد) اما الثانيه فسنقوم فيها بتنفيذ الامر mount على واحد من هذه ال partition الموجوده .

ولرؤيه ال partition كلها الموجوده على نظامك فاننا سنستخدم الامر fdisk بالاضافه الى الخيار -l (اى small L) .  
الان تحول الى مدير النظام root ونفذ هذا الامر

`fdisk -l /dev/hda`

(الامر fdisk يتطلب كونك root لتنفيذه ، لانه من اوامرداره النظام التى تتطلب صلاحيات مدير النظام)  
لاحظ ايضا انه قد تختلف الصيغه hda فى نظامك فقد تكون hdb او hdc او حتى hde بناء على كون الهارد master او slave .

وسترى نتيجه مشابهه بتلك المعروضه امامنا

```
[root@elnajeeb root]# fdisk -l /dev/hda

Disk /dev/hda: 2621 MB, 2621251584 bytes
128 heads, 63 sectors/track, 634 cylinders
Units = cylinders of 8064 * 512 = 4128768 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/hda1    *           1          19       76576+   83  Linux
/dev/hda2                20          44      100800   82  Linux swap
/dev/hda3                45          69      100800   83  Linux
/dev/hda4                70         634     2278080    5  Extended
/dev/hda5                70         634     2278048+   83  Linux
[root@elnajeeb root]# _
```

فنرى على اقصى اليسار ال device ورقمه ، اما على اقصى اليمين فيوجد نوع file system.

وما نريده من هذا المثال هو ذلك الذى على اليسار ، اى /dev/hdaN . مع ابدال الحرف N بالرقم المناسب عندك .

\*اذا كان بنظامك اكثر من partition للويندوز ، فيمكنك اختيار احدهم لتنفيذ هذه العملية عليه .

هذا ببساطه هو ال device ، وما نحتاجه الان لاكمال عملية ال mount هو ال directory ، وما عليك الان فعله هو فقط انشاء مجلد فارغ - من اى محتويات- ليتم ربط هذا ال device به . ولنفترض اننا سننشئه تحت /mnt وسنسميه windows ليسهل علينا تذكره .

الان نفذ هذا الامر

```
mkdir /mnt/windows
```

#### ملاحظه

يفضل انشاءه فى ال home directory الخاص باى مستخدم عادى -ولنفترض انه المستخدم الذى تعتاد الدخول باسمه للنظام- ، والسبب فى ذلك هو ، تسهيل الدخول لهذا ال directory واستعراض محتوياته ، لكى لا تضطر كل مره تريد الدخول فيها اليه ان تتحول الى المستخدم root .

بذلك تكون الخطوه الاولى قد تمت ، ولم يتبق الا الخطوه الثانيه وهى عملية ال mount . وهى تتم بهذا الامر

```
mount /dev/hdaN /mnt/windows
```

الان يمكنك الانتقال الى المجلد /mnt/windows سواء بالامر cd او حتى عن طريق ال gui وستجد ان بداخل هذا المجلد كل الملفات والبيانات الموجوده فى هذا ال partition بالويندوز .

---

#### ملاحظه هامه

يلاحظ ان الامر mount الذى اجريناه على هذا ال windows partition والذى قام بعمل الحاق لهذا ال partition واستطعنا بعده ان نتعامل مع البيانات الموجوده فى هذا ال partition عباره عن الحاق مؤقت ، اى بمجرد اغلاق النظام فان هذا ال partition سيحدث له umount ولن يحدث له mount فى المره المقبله من تشغيل النظام . ولجعل هذا ال mount او اللاحق دائم -لا ينتهى باغلاق النظام- لا بد من تسجيله فى ملف fstab ، وهذا الاجراء بسيط وسأشرحه بعد الانتهاء من الامر mount .

يلاحظ ايضا اننا يمكن استخدام الامر mount منفردا ، او مع الخيار -i- وذلك لعرض ال device المتصل والملتحق بالنظام الان .

## خيارات الامر mount .

ياتى مع الامر mount نوعان من الخيارات او options

- 1 cmd line option وهى الخيارات التى اعتدنا كتابتها مع الاوامر مثل a- و h- وغيرهم .
- 2 mount option وهذا النوع عبارته عن مقطع من كلمه (اى اختصارا لها) مثل ro , user , noauto وغيرهم ، وتحدد هذه الخيارات- معلومات عن كيفية ربط هذا ال device .

النوع الاول ، واشهر الخيارات فيه هى

\* **a-** وتعنى اربط كل ال partition الموجود فى ملف fstab ، ما عدا اولئك المحددين بالخيار noauto ، وهذا الخيار هو الذى يستخدمه النظام حين يبدأ العمل والموجود فى ال script الرئيسى للنظام <sysinit>

\* **o-** ويحدد بعد هذا الحرف اى خيار من خيارات النوع الثانى ، والذى سنذكرهم لاحقا (يلاحظ ان حرف o اختصار لكلمه option)

\* **r-** وتعنى read only اى ربط هذا ال device للقراءة فقط .

\* **w-** وتعنى read and write اى ربط ال device كقراءة وكتابة .

\* **t-** وبعد هذا الخيار نكتب نوع الملفات لهذا ال device ويتم استعمال هذا الخيار عندما لا يكون fs محدد لل device بالعمود الثالث فى ملف fstab . وسيتم سرد انواع ال file system فى السطور المقبلة .

مثال على كيفية استخدام هذه الخيارات

```
mount -t iso9660 /mnt/cdrom
```

## النوع الثانى

وهذا النوع من الخيارات اما ان يوضع فى العمود الرابع بملف fstab واما ان يكتب على ال shell ولكن لا بد ان يسبقه الحرف o- .

وهذا النوع من الخيارات عبارته عن كلمه او مقطع منها ، وليس حرف كالنوع الاول . واشهر هذه الخيارات هى

\* **defaults** وهذا هو اشهر الخيارات على الاطلاق ، ذلك لان هذا الخيار يعنى فى ذاته 7 خيارات اخرى (اى انك لو حددت هذا الخيار لل device فكانك قد كتبت 7 خيارات) وهم ,nouser , auto, exec, dev, suid ,rw . وهذا الخيار مفيد جدا اذا كنت جديد فى التعامل مع ملف fstab فبسهولة يمكنك تحديده لل device الجديد .

\* **auto** والتى تعنى ربط ال device تلقائيا ، فلا يحتاج المستخدم ان يربطه بال device

بالامر mount .

\* **noauto** وهذا الخيار عكس السابق ، فاذا تحدد لل device فلا بد ان يربطه المستخدم بنفسه عن طريق الامر mount .

\* **dev** ويعنى هذا الخيار ان ال fs يستطيع ان يفهم ويتعامل مع انواع ال device المعينه ، مثل ال block device مثل الهارد ديسك ، وال character device مثل ال terminal وغيرهم

\* **exec** ويتحدد هذا الخيار اذا كان بال partition برامج executable فانه يتم عمل exec لهم بهذا الخيار .

\* **noexec** ويعنى عدم تشغيل اى برامج executable موجوده فى هذا ال partition ، ويعد هذا الخيار اجراء امنى هام اى security measure .

\* **nosuid** منع تأثير ال suid و sgid على الملفات ال executable .

\* **nouser** وتعنى منع اى مستخدم غير root من عمل mount و umount لل device .

\* **ro** وتعنى read only وتتساوى مع الخيار -r (من النوع الاول)

\* **rw** وتعنى read and write وايضا تتساوى مع الخيار -w (من النوع الاول)

\* **suid** وتعنى تطبيق خاصيه set user id وايضا set group id على الملفات ال exec .

\* **user** وهذا الخيار مهم جدا ، ويعنى ان المستخدم الذى عمل mount لل device هو فقط من يستطيع عمل unmount له .

\* **users** وهذا الخيار مثل الذى قبله ، الا ان الفرق هو ، ان اى مستخدم يستطيع عمل umount لل device وليس الذى عمل ال mount تحديدا .

مثال على كتابه هذه الخيارات مع الامر mount

```
mount -o ro /usr
```

يمكن الاطلاع على بقيه الخيارات وذلك بصفحه ال man page الخاصه بالامر mount وايضا الخاصه بالملف . fstab

ولكن ضع فى اعتبارك دائما هذه الخيارات defaults و nouser و user و users لاهميتها.

## انواع ال file system .

هذه الانواع من ال file system هي التي تكتب في العمود الثالث بملف fstab ، وايضا يمكن كتابتها على سطر الاوامر shell ولكن لا بد ان يسبقها الحرف t- .

واشهر هذه الانواع هي

\* **ext2 و ext3** وهذه بالطبع هي الانواع ال default لل linux file system ، فنجدهم دائما مع كل ال partition الخاصه بالنظام ، مثل / و boot و home و etc و use وغيرهم من ال partition المختلفه ل لينوكس . (باستثناء swap و proc)

\* **vfat** وهذا ال fs يتحدد ل windows partition .

\* **iso9660** وهذا هو ال fs الخاص بال cdrom ، وله ايضا fs اخر وهو udf ولكن الاول هو الاشهر وال default ايضا .

\* **nfs** ويتحدد هذا ال fs اذا لم يكن ال partition موجود على ال local hard ولكنه على جهاز اخر في نفس الشبكه .  
ومنذ زمن ليس بالبعيد كان يوضع هذا ال fs لل /usr ، عندما كان يوضع هذا ال partition على ال server ليخدم كل المستخدمين ، وذلك بسبب كبر مساحه ال partition من ناحيه ومن ناحيه اخرى بسبب صغر حجم ال hard disk للمستخدمين وقت ذاك .

\* **swap** ويتحدد هذا ال fs لل swap partition فقط .

\* **proc** وهذا النوع من ال fs خاص فقط بال proc .

وهذا النوع -proc- لا يعد fs في حد ذاته ، بل هو اشبه ب virtual file system ، ولهذا فان له format خاص به .  
ويلاحظ ايضا -ولاجل هذا السبب- انه ليس له device خاص يربط به ، ولهذا فان في عمود(خانه) ال device الخاص به نجد انه لا يتحدد له device بل يكتب none .

## مثال عملي للامر mount .

المقصود بهذا المثال العملي هو كيفية التعامل مع الملف `fstab` ، ولكن لان المتعامل مع هذا الملف كان لا بد ان يكون عالما بالامر `mount` ، لهذا فاني وضعت هذا المثال بعدما شرحت الامر `mount` .

ويمكن استعاره المثال السابق الذكر والذي قمنا فيه بعمل `mount` ل `partition` من الويندوز .

فاستخدام الامر `mount` في عمل الحاق ل `device` معين للنظام هو اجراء مؤقت ينتهي باغلاق النظام . اما الاجراء الدائم الذي يحفظ عمليه ال `mount` هذه ويجريها كلما اشتغل النظام ، فهو بتسجيل هذا ال `partition` ببياناته في ملف `fstab` .

وهذه الاجراءات هي

- 1- فتح الملف `fstab` عن طريق اى `text editor` وساستعمل انا ال `vi editor` وذلك بالامر `vi /etc/fstab`
- 2- تحرك بالاسهم لاسفل الملف ثم اضغط على حرف `o` ليبدأ بالتسجيل ولكن من السطر التالي
- 3- اكتب اولاً وفي خانة ال `device` رقم ال `partition` على الهارد عندك ، وستكون صيغته كالتالي `/dev/hd?3` ، مع ملاحظه ابدال ؟ بالحرف المناسب على نظامك (غالبا `a` وقد يكون `b` او `c` او `d`) وابدال الرقم 3 بالرقم المناسب لل `windows partition` عندك
- 4- في خانة المجلد ستكتب اسم المجلد ومساره ، وهذا المجلد هو الذى ستتعامل من خلاله مع ال `partition` .
- 5- في العمود الثالث والخاص بال `fs` ستكتب `vfat` لان هذا ال `partition` نوعه ويندوز .
- 6- وفي العمود الرابع ستكتب الخيار `defaults` والذي بكتابته سيغنيك عن عده خيارات اخرى .
- 7- في العمود الخامس والخاص بالبرنامج `dump` ستكتب 0 لان هذا ال `fs` ليس لينوكس .
- 8- في العمود السادس والخاص بالبرنامج `fsck` ستكتب اما رقم 2 لعمل `fsck` ، او ستكتب 0 لكى لا يقوم البرنامج بعمل `check` عليه .

بعد ذلك ستحفظ ما كتبته وستخرج من الملف وذلك بالخطوات التاليه

اضغط على المفتاح `Esc`

اكتب الرمز `colon` `>:` وذلك بالضغط على `shift` والحرف `<ك>`

اكتب الحرفين `wq` ، وان صادفك انه لا يستطيع الخروج من الملف ، فاضف الحرف `> !`

يلاحظ ان الملف `fstab` يقوم النظام بقرأته مره واحده وذلك عندما يبدأ التشغيل . وبهذه الخطوات سيقوم النظام بعمل `mount` لهذا ال `partition` كلما بدء التشغيل .

## الامر umount .

يعد الامر **umount** عكس الامر **mount** .  
ووظيفته تتمثل فى فصل الربط او اللاحق **attach** الذى قام به الامر **mount** .

صيغته

وهو مثل الامر **mount** ولكن له صيغتان فقط وهم

**umount [option] device**  
**umount [option] directory**

ويمكن اعطاء ال **cdrom** كمثال عليه  
فبعد الانتهاء من ال **cdrom** واراده فصله عن النظام ، لا تعتقد انك ببساطه ستضغط على الزر **eject** وسيخرج ال **cd** من ال **cdrom** كما يحدث فى الويندوز ، فنظام ال **file system** هناك غير نظامه هنا فى لينوكس .

فكما راينا ، النظام ككل عباره عن شجره كبيره ، وال **cdrom** فيها كفرع من افرعها ، فان اردت فصل هذا الفرع عن الشجره ، فيتم ذلك بالامر **umount** .

ومثال الصيغه الاولى هو

**umount /dev/cdrom**

ومثال الصيغه الثانيه هو

**umount /mnt/cdrom** او ببساطه **umount /cdrom**

يمكنك بعد ذلك اجراء الامر **mount** فقط بدون اى **option** لتتأكد ان عمليه اللاحق لهذا ال **device** قد انتهت ، ومن ناحيه اخرى لرؤيه بقية ال **device** المعمول لهم **attach** .



# The User Administration

المقصود ب User Administration هو اداره حساب المستخدم/المجموعه .  
واداره حساب المستخدم user account تشمل انشاء حسابه ، وتعديل هذا الحساب ، وايضا  
إلغائه من النظام ، هذا بالاضافه الى اجراء نفس العمليات الثلاثه على المجموعه group .

نظام لينوكس يعتمد فى الدخول اليه -log in- على حساب سابق للمستخدم بالاضافه الى كلمه سر ،  
والحساب او ال account هذا ، عبارته عن اسم هذا المستخدم فى معظم الاحيان .

هذا بالنسبه لحساب المستخدم قبل الدخول ، اما بعد الدخول فان المستخدم لا بد وان ينتمى الى  
مجموعه من مجموعات النظام .

ومهمه مدير النظام Administrator او ال root ، هى انشاء هذا ال account للمستخدم  
الجديد ، او تهيئته(بالتعديل او الالغاء) اذا كان موجود من قبل ، هذا بالاضافه الى تهيئته المجموعه  
التي ينتمى اليها .

## التعامل مع المستخدم .

للمستخدم فى انظمه يونكس ولينوكس اهميه كبرى ، ويعتمد عليه فى جانب كبير فى حمايه النظام  
ولهذا فان للمستخدم فى النظام (رقم) بالاضافه الى اسمه العادى الذى يدخل به الى النظام ،  
<المستخدم يستعمل اسمه فى تعامله مع النظام ، والنظام يستعمل رقم هذا المستخدم فى عملياته> وهذا الرقم  
فريد اى unique بمعنى انه لا يمكن ان يتكرر لاي مستخدم اخر فى النظام .

وكل عمليه يقوم بها هذا المستخدم فى النظام تكون بالاعتماد على رقمه المنفرد هذا ، فالبرامج  
والاوامر لا تعرف الاسماء .

والمستخدم لا بد وان ينتمى الى مجموعه .

وعند انشاء اى ملف او مجلد ، فان هذا الملف/المجلد بالاضافه الى كونه مملوك لمستخدم ، الا انه  
لا بد ان يتحدد له مجموعته ، والتي غالبا تكون المجموعه ال default للمستخدم .

والامر ls (او حتى ll ) يعرض لنا المستخدم والمجموعه ، التابع لها الملف المعين ، مثل

```
[ahmed@elnajeeb ahmed]$ ll
total 1
drwxr-xr-x  2 ahmed  ahmed  1024 Jan 23 13:54 ahmed
-rw-r--r--  1 ahmed  ahmed    0 Jan 23 12:55 ahmed1
-rw-r--r--  1 ahmed1  root     0 Jan 23 16:02 newown
-rw-r--r--  1 root    root     0 Jan 23 15:27 perm
[ahmed@elnajeeb ahmed]$
```

فنرى نتيجة هذا الامر عبارته عن اربعة اسطر ، وسنركز فقط على السطر الاول والثالث

فنجذ فى السطر الاول وبعد ال permission، كلمه ahmed مكرره مرتان ، فالاولى داله على المستخدم ، والثانيه داله على المجموعه . (وهنا المجموعه لها نفس اسم المستخدم) .

اما فى السطر الثالث -والخاص بالملف newown- فنجذ ان مالكه مستخدم يسمى ahmed1 وهذا المستخدم مشترك فى مجموعته مدير النظام ال root ، ولهذا فان المجموعه تسمى root .

وهكذا ، فالمستخدم فى المثالين هو منشأ الملفات ، وبعد انشاء هذه الملفات فالمجموعه التى ينتمى اليها المستخدم حق فى هذه الملفات ، وهذا الحق هو الذى يحدده المستخدم -وايضا ال root- وتمثل فى التصاريح الممنوحه لها .

### التعامل مع المجموعه .

\* ال group هى عبارته عن مجموعته من المستخدمين يجمع بينهم شى مشترك غالبا .  
\* والمجموعه لها رقم واسم منفردين unique ، فلا يتكرر اسمها او رقمها مع اى مجموعته اخرى

\* والمجموعه قد تحوى مستخدم واحد فقط ، وقد تحوى العشرات من المستخدمين .

\* وكل مستخدم لا بد وان ينتمى الى مجموعته واحده على الاقل ، وقد ينتمى لعدده مجموعات فى نفس الوقت .

\* والمجموعه الاساسيه للمستخدم تسمى primary group .

\* والمجموعه الثانيه للمستخدم تسمى secondry group .

\* وتستعرض المعلومات عن المجموعه بواسطه اوامر مثل ls -l وايضا stat .

وتتم ادارته حساب المستخدم/المجموعه عن طريق عده اوامر وايضا عده ملفات .

فالوامر بالنسبه للتعامل مع المستخدم هي

|         |                             |
|---------|-----------------------------|
| useradd | لاضافه حساب لمستخدم جديد .  |
| userdel | لإلغاء حساب موجود بالفعل .  |
| usermod | لتعديل حساب لمستخدم موجود . |

اما الملفات (والموجوده في المسار /etc ) فهي

|        |                                                                       |
|--------|-----------------------------------------------------------------------|
| passwd | وهو الملف الاساسي في التعامل مع المستخدم ، ويحوى معلومات عن كل مستخدم |
| shadow | وهو ملف يحوى معلومات خاصه ، وسريه ، ولا يتعامل معه الا ال root .      |

وبالنسبه للمجموعات ، فامر ها شبيه بالمستخدم .

فالوامر التى نتعامل بها مع المجموعات هي

|          |                                  |
|----------|----------------------------------|
| groupadd | ويستخدم لاضافه مجموعه جديده .    |
| groupdel | ويستخدم لإلغاء مجموعه موجوده .   |
| groupmod | وهو لتعديل مجموعه قائمه بالفعل . |

والملفات الخاصه بالمجموعات (والموجوده ايضا بالمسار /etc ) فهي

|         |                                               |
|---------|-----------------------------------------------|
| group   | وتحوى معلومات عن المجموعات الموجوده بالنظام . |
| gshadow | وهو مثل الملف shadow المذكور باعلى .          |

## اولا : التعامل مع المستخدمين .

### الامر **useradd** .

ميزه اوامر اداره المستخدمين والمجموعات ، انها ذات صيغه واحده .

فكل الاوامر الاساسيه فى اداره المستخدم تبدأ بالكلمه **user** ثم يضاف اليها المقطع المناسب لها ، مثل **add** للاضافه او **del** للازاله او **mod** للتعديل .  
ونفس الشئ مع اوامر المجموعات .

والامر **useradd** قد يستعمل منفردا بالاضافه الى اسم المستخدم ك **argument** ، مثل

**useradd NAME**

وهذه هي الصيغه الاشهر فى انشاء المستخدمين الجدد .  
وهذا الامر السابق ينشأ القيم الافتراضيه او ال **default** ، مثل انشاء ال **home directory** الخاص بهذا المستخدم تحت المسار **/home** ، وانشاء مجموعه بنفس اسم المستخدم ، وغيرها من القيم ال **default** .

ولكن عند اراده تغيير هذه القيم ، فيتم اضافته **option** الى الامر **useradd** .  
ومن اشهر ال **option** التى تاتى مع الامر **useradd** هي

\* **-g** وهذا ال **option** وظيفته تغيير ال **group** الافتراضيه **default** التى ينشأها النظام تلقائيا  
بنفس اسم المستخدم الجديد ، مثال

**useradd -g <group> <newuser>**

فى هذا المثال تم الحاق المستخدم الجديد (والذى يسمى **newuser**) بالمجموعه الموجوده بالفعل  
والمسماه -على سبيل المثال- **group** .  
فلو لم نحدد المجموعه للمستخدم الجديد بالخيار **-g** فان النظام سينشئ (تلقائيا) مجموعته لها نفس  
اسم المستخدم .

\* **-G** ووظيفه هذا ال **option** هو اضافته المستخدم الى اكثر من **group** فى نفس الوقت ،  
. ويمكن ان ياتى هذا ال **option** فى صورتين .

الاولى : منفردا ، ومثاله

**useradd -G root <user>**

فالناتج من هذا المثال شيئين  
الاول هو انشاء مستخدم جديد بالاضافه الى انشاء مجموعته لها نفس اسمه (اى اننا سنجد مستخدم  
يسمى **<user>** وايضا سنجد له مجموعته لها نفس الاسم **<user>**  
الثانى ، سيضاف هذا المستخدم الجديد **<user>** الى مجموعته ال **root** .

الثانيه : مع الخيار -g مثل

```
useradd -g <primary> -G <secondry> <user>
```

وبمعرفه الفرق بين الصورتين سنعرف نتيجته هذا الامر الاخير .

فالفرق بينهم بسيط -مع ان شكل الامر يخيف- وهذا الفرق هو اننا فى المثال الثانى اضعنا المستخدم الى مجموعه موجوده بالفعل (سيكتب اسمها مكان primary) وهذا هو وظيفه الخيار -g- ثم بعد ذلك اضعنا هذا المستخدم الى مجموعه ثانيه (واسمها سيكون مكان secondry) وهذا هو وظيفه الخيار -G- .

اى ان الفرق بين الصورتين متعلق بالgroup التى سيشارك فيها للمستخدم ، فبالصوره الاولى (-g) سيضيف النظام المستخدم الجديد الى مجموعه جديده ، اما الصوره الثانيه فنحن نحدد بانفسنا المجموعتين الاولى والثانيه التى سيشارك فيهما المستخدم الجديد.

لكن يلاحظ انه لا بد من وجود المجموعتين بالفعل على النظام . ويكون المستخدم جديد ، لانه لو كان موجود من قبل فان طريقه اضافته الى المجموعات لا تتم بهذه الطريقه، بل ستم بالامر chgrp .

**\* -d** وهذا ال option وظيفته تغيير ال home dir للمستخدم الجديد ، مثل

```
useradd -d /var/newuser newuser
```

ففى هذا المثال تم تغيير ال home dir للمستخدم newuser الى /var/newuser (والمكان الطبيعى لانشاء ال home dir للمستخدمين الجدد هو /home )

**\* -c** وبهذا ال option سيتم وضع comment للمستخدم ، بمعنى اخر ، معلومات اضافيه كرقم التليفون ، عنوانه وهكذا . (ويمكن رؤيه هذه التعليقات بالامر finger)

```
useradd -c <tel 010xxxxx> <user>
```

وهناك العديد من ال option موجوده بال man page للامر useradd يمكن الاطلاع عليها، ولكن المهم الان هو ال option المسمى -D- وهو ما سنناقشه الان .

يلاحظ انه لا بد من تنفيذ الامر passwd على المستخدم الجديد بعد انشاء حساب له ، ذلك لانه بعدم تنفيذ مثل هذا الامر لن يكون للمستخدم passwd على الاطلاق ، وبالتالي لن يستطيع هو نفسه من الدخول الى النظام .

## تعديل القيم ال default .

يمكن بال option المسمى -D تعديل القيم الاساسيه ال default التى ستنشأ لكل المستخدمين الجدد .

فاى مستخدم جديد عند انشاء account له بالنظام -وذلك بالامر useradd فقط بدون اى option- فانه ياخذ القيم default المحدده فى النظام من قبل .  
ومن هذه القيم مثلاً ، كون ال home directory الخاص به موجود تحت /home ، و نوع ال shell الخاصه به ، وغيرها .

والفرق بين هذا التعديل ، وبين ادخال option الى الامر useradd هو  
فى حال ادخال option مع الامر useradd فان هذا التعديل سينطبق فقط على هذا المستخدم الذى تم تطبيق هذا ال option عليه .  
اما مع حاله التعديل بالخيار -D فان هذا التعديل سيغير فى القيم الاساسيه التى يضعها النظام لكل المستخدمين الجدد .

فمثلاً ، بكتابه الامر useradd user فان النظام سينشأ لهذا المستخدم الجديد مجلد تحت /home ليكون هو ال home dir له ، والسؤال هنا لماذا ينشأ النظام هذا المجلد فى هذا المكان؟

والاجابه هى ، انه يقرأ من ملفات معينه ، تخبره هذه الملفات بانه لا بد ان ينشأ هذا المجلد تحت /home .

ووظيفه الخيار -D هو التعديل فى هذه الملفات . فمثلاً اذا كتبنا الامر التالى

```
useradd -D -b /var
```

فان نتيجه هذا الامر هو تحويل ال home directory لاي مستخدم جديد فى النظام الى المسار /var بدلاً من كونه فى المسار /home .  
فاى مستخدم جديد سيتم انشاء account له بالنظام بالامر useradd فقط بدون اى option سيجد ان ال home dir الخاص به موجود تحت /var .

ويوجد مع الخيار -D عده خيارات (مثل -b سالف الذكر) وتغير هذه الخيارات القيم ال default للنظام .  
ويمكن الاطلاع عليها من ال man page .

\* ايضا يمكن استعراض هذه القيم ال default (والتي يعطيها النظام لاي مستخدم جديد) بالامر

```
useradd -D
```

فقط وبدون اى اضافات ، فترى ناتجه عبارته عن عده سطور ، تبين هذه السطور القيم ال default على نظامك .  
\* والملف المسجل به هذه القيم موجود فى [/etc/default/useradd](#) .

## الامر userdel

يستخدم الامر userdel لالغاء حساب (account) لمستخدم موجود بالفعل على النظام .

وصيغه هذا الامر بسيطه جدا وهى

```
userdel <user>
```

فبذلك يتم الغاء هذا الحساب من النظام .

ويأتى مع هذا الامر خيار واحد فقط ، وهو **-r** وهذا ال option وظيفته الغاء كل الملفات والمجلدات الموجوده فى ال home directory للمستخدم ، مثل

```
userdel -r <userName>
```

ويلاحظ انه اذا لم يستخدم هذا ال option فان النظام سيقوم بالغاء حساب المستخدم فقط ، مع الابقاء على ال home directory الخاص به .

## الامر usermod

وظيفة هذا الامر هو تعديل البيانات الخاصة بالمستخدم ، ولهذا فهو اختصار ل **modification** .

والخيارات التى تاتى مع هذا الامر تقريبا نفس الخيارات التى تاتى مع الامر **useradd** ، مثل

- g** . لل primary group
- G** . لل secondry group
- c** . للمعلومات الاضافيه comment
- e** . لعدد الايام الباقيه للمستخدم او expire date
- s** . لتعديل ال shell الافتراضيه للمستخدم default
- u** . لتغيير ال UID او بمعنى اخر ، رقم المستخدم فى النظام

وغيرهم من الخيارات الاخرى ، ولكن الذى يجب التركيز عليه والانتباه له خياران وهم

**-L** . ويقوم هذا ال option بعمل lock لكلمه المرور password الخاصه بالمستخدم ، فلا يستطيع المستخدم الدخول للنظام -لان ال password مغلقه- وايضا لا يستطيع المستخدم ان يغير ال password الا بعد ان يزيل ال root هذه الخاصيه .  
ويضيف هذا ال option علامه < ! > امام كلمه السر للمستخدم فى الملف shadow (والذى سنتحدث عنه بعد قليل)

**-U** . ويتم بهذا ال option عمل unlock للمستخدم ، ومن ثم تمكينه من تغيير password اى ان هذا ال option عكس الذى قبله .

Usermod -L <userName>

فبهذا الامر فاننا منعنا المستخدم من امكانيه تغيير ال password ، وللسماع له ، ننفذ الامر

usermod -U <userName>

## ملفات المستخدم على النظام .

وهذه الملفات هي passwd و shadow والموجودان بالمسار /etc .

فالملف /etc/passwd مهمته حفظ معلومات عن المستخدم ، وهذه المعلومات -التي سنتعرف عليها الان- تعد معلومات عامه عن المستخدم .  
اما الملف /etc/shadow فهو ايضا لحفظ معلومات عن المستخدم ، ولكن هذه المعلومات اكثر اهميه من الموجوده بالملف passwd .

ويكفي ان نعرف ان الملف shadow له permission خاص جدا ، قيمته 400 اى انه يسمح فيه بالقرأه فقط لمالكه وهو بالطبع ال root .  
فى حين ان ال permission الخاص بالملف passwd هو 644 اى، قرأه وكتابه للمالك root وقرأه فقط لمجموعه ال root وللباقيين .

والملاحظ هنا شيئان ، هما

.اولا ، ان كان ال permission للملف shadow هو قرأه فقط لل root فهذا لا يمنع على الاطلاق امكانيه التعديل فيه ، لان ال root له كافه الصلاحيات بالنظام ، فعند محاوله التعديل فى الملف سيظهر لك باسفل الملف سطر يوضح ان الملف قرأه فقط ، وستضطر بناءً على ذلك ان تضيف علامه > ! < بعد الحروف wq: لكى تضطره الى قبول التغيير الذى حدث .

. ثانيا ، السبب فى جعل ال permission بهذه الصوره للملف shadow وليس كذلك للملف passwd -مع ان كليهما من ملفات النظام- ، هو ان الملف passwd يستخدم بواسطه برامج تعتمد عليه فى عملها ، فالوامر التى لا بد ان تتعرف على ال shell الخاصه بالمستخدم او حتى التى لا بد لها من معرفه ال UID وال GID للمستخدم (وهى من ضمن المعلومات التى فى الملف) لا بد لها ان تقرأ وتطلع على هذا الملف .

ولهذا فان المعلومات الخاصه بالمستخدمين موجوده فى ملفان ، وهذا هو ما سنتعرف عليه الان .

## ملف passwd .

شكل هذا الملف كالاتي

```
1      2      3      4      5      6      7
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:/sbin/nologin
news:x:9:13:news:/etc/news:
uucp:x:10:14:uucp:/var/spool/uucp:/sbin/nologin
operator:x:11:0:operator:/root:/sbin/nologin
games:x:12:100:games:/usr/games:/sbin/nologin
gopher:x:13:30:gopher:/var/gopher:/sbin/nologin
ftp:x:14:50:FTP User:/var/ftp:/sbin/nologin
rpm:x:37:37::/var/lib/rpm:/bin/bash
mailnull:x:47:47::/var/spool/mqueue:/sbin/nologin
smmsp:x:51:51::/var/spool/mqueue:/sbin/nologin
pcap:x:77:77::/var/arpwatch:/sbin/nologin
ahmed:x:500:500::/home/ahmed:/bin/bash
apache:x:48:48:Apache:/var/www:/sbin/nologin
mohamed:x:501:500::/home/mohamed:/bin/bash
```

فنجذ شكل الملف passwd عبارته عن 7 اعمده او خانات ، يفصل بينهم بعلامه colon > : < ومعاني هذه الاعمده هي

1- **username** وبه يتم تسجيل اسم المستخدم ، وهذا هو الاسم الذي يدخل به المستخدم للنظام

2- **password** ونجد في هذه الخانه العلامه x والتي تعني ان الpassword مشفره وموجوده بالملف shadow .

3- **UID** وهو الرقم المحدد لكل مستخدم ، وهو كما اسلفنا ، رقم منفرد unique .

4- **GID** وهو رقم المجموعه التي ينتمى اليها المستخدم .

5- **comment** وبهذه الخانه توضع المعلومات الاضافيه للمستخدم ، مثل التليفون او العنوان ...

6- **home dir** وبه يتحدد ال home directory الخاص بالمستخدم .

7- **shell** ويكون بهذه الخانه ال shell المحدده لكل مستخدم على حده .

## ما يجب ملاحظته

يلاحظ على الملف passwd ما يلي

- \* كل مستخدم بالنظام له سطر بهذا الملف ، تتحدد به معلومات هذا المستخدم .
- \* ان المستخدمين الموجودين بالملف هم ثلاث انواع

- 1- super user وهو ال root .
- 2- special user وهم من نوعيه bin و daemon و adm و lp وغيرهم .
- 3- regular user وهم مثل ahmed و mohamed الموجودين باسفل الملف .

فال root او مدير النظام له رقم خاص به ، وهو الرقم 0 سواء لل UID او GID .

اما الارقام من 1 الى 99 فهي ارقام خاصه بمستخدمين من نوع معين (يطلق على هذه النوعيه system account) ، من امثله lp او ftp او rpm ، وهؤلاء المستخدمين لهم تعامل خاص مع النظام ، ويتمثل هذا التعامل تقريباً في ان النظام عندما ينفذ امر خاص مثلاً بال ftp server فانه يستدعى هذا المستخدم المسمى ftp ليقوم بتنفيذ مثل هذه الاوامر ، وكذلك الحال مع lp وبقيه المستخدمين .

اما الارقام من 500 فما فوق فهي مخصصه للمستخدمين الذين سيتم اضافتهم الى النظام عن طريق المدير root ، ولهذا فان المستخدمين من امثال ahmed و mohamed نجد ارقامهم تبدأ من 500 .

يلاحظ ايضا

ان خانه password لا يكتب بها ال password ، بل يوجد مكانها الحرف x ، وهذا يعنى ان ال password موجوده بالملف shadow (وذلك لاجراءات امنيه)

\* خانه ال comment نجد انها مع بعض المستخدمين فارغه ومع البعض الاخر بها تعليق او معلومات ، وهذه الخانه ليست من الاهميه بمكان ، فان اردت استخدامها فيمكنك ذلك ، وان لم ترد فليس في ذلك شئ .

\* يلاحظ ان كل المستخدمين ال regular لهم shell من نوعيه bash ، وهى ال default للنظام ، فان لم تتحدد هذه الخانه ، او تركت فارغه ، فان النظام سيعطى المستخدم shell من نوعيه sh .

\* فى النهايه ، لا بد من اعطاء اهميه لل root ورقمه 0 ، ذلك لان ال crackers عندما يحاولون الدخول للنظام ، فانهم يدخلون عن طريق كتابه record - اى سطر - بالملف passwd باى اسم ، ثم يضعون لهذا الاسم الرقم 0 وبذلك يصبح لهم صلاحيات ال root ، فكن على حذر من هذه الخدعه .

## ملف shadow

وشكل هذا الملف كالاتي

```
1      2      3      4 5      6 7 8 9
root:$1$WuzwFX7J$0d0zdb32WUCZqX001.nt1/:12742:0:99999:7:::
bin:!:12742:0:99999:7:::
daemon:!:12742:0:99999:7:::
adm:!:12742:0:99999:7:::
lp:!:12742:0:99999:7:::
shutdown:!:12742:0:99999:7:::
halt:!:12742:0:99999:7:::
mail:!:12742:0:99999:7:::
news:!:12742:0:99999:7:::
operator:!:12742:0:99999:7:::
games:!:12742:0:99999:7:::
gopher:!:12742:0:99999:7:::
ftp:!:12742:0:99999:7:::
rpm:!:12742:0:99999:7:::
sshd:!:12742:0:99999:7:::
rpc:!:12742:0:99999:7:::
nfsnobody:!:12742:0:99999:7:::
mailnull:!:12742:0:99999:7:::
smb:!:12742:0:99999:7:::
ahmed:$1$K7sm2yCc$yHnmxiKMOa92pqtRPChSe.:12824:0:99999:7:::
apache:!:12782:::
mohamed:!:12827:0:99999:7:::
```

وهذا الملف كما نراه شبيه بالملف passwd الا انه يزيد عن ملف passwd بخانتين او عمودين ، فهذا الملف 9 خانات . ومعانيهم هي

1- **username** وهو اسم الدخول للمستخدم ، وهو ذاته نفس الاسم الموجود بملف passwd .

2- **password** ونراها هنا <مشفرة> encrypted ، وهي عبارة عن رموز .

3- **last passwd chang** والرقم الموجود فيها-مثل 12742- هو رقم standard يبدأ من 1 يناير 1970 . ويعني اخر مره تم تغيير ال password فيها .

4- **days untill change allow** وهو الوقت الذي ينتظره المستخدم لكي يستطيع تغيير ال password ، ويحسب من اخر مره تم فيها تغيير ال password .

5- **days before change required** وذلك يعني امهال المستخدم فتره ، لا بد بعدها من تغيير ال password ، وهي كما نرى انها standard ومحدده ب 99999 .

6- **warning days before passwd expire** وتوضع بهذه الخانه عدد الايام التي لا بد للمستخدم ان يغير ال password فيها قبل انتهاء حسابه ، ونراها محدده ب 7 ايام . فان كان حساب المستخدم محدد ب 99999 فقبل انتهاء هذه الايام ب 7 ايام لا بد ان يتغير ال passwd .

7- **days between expiration and deactivation** وبه يتحدد الايام بين انتهاء حساب المستخدم ، وبين اغلاق الحساب كلياً ، والفرق بينهما ، ان الاولى متعلقه بتعامل المستخدم مع النظام ، اما الثانيه فمتعلقه بسياسه ادارته النظام .

-8 **account expire** وتعنى ، ان الحساب سيغلق بعد هذه الفتره .

-9 **special flag** هذه الخانه غير مستخدمه حاليا ، ومحجوزه للاستخدام فى المستقبل .

## ما يجب ملاحظته

يلاحظ على الملف shadow ما يلي

\* اننا لا نتعامل مع كل هذه الخانات او الاعمده ، بل ان تعاملنا -فى الغالب- يقتصر فقط على ثلاث او اربع خانات ، ولهذا فلا تجعل عدد هذه الاعمده ومعانيها يرهبك .

\* خانه username لمعرفة المستخدم الذى نتعامل معه ، اى على من سيتم التعديل المراد .

\* خانه ال password ونجد ان لها عدة اشكال فى هذا الملف

. الشكل الاول ، وهو < !! > ، والذى يعنى ان هذا المستخدم لم توضع له password حتى الان .

. الشكل الثانى وهو < \* > والذى يعنى ان هذا المستخدم لا توضع له password (ونرى هذه العلامة دائما مع ال system account من امثال daemon و bin وغيرهم .  
. الشكل الثالث وهو كما نراه عبارته عن احرف ورموز ، وهذا الشكل يوضع للمستخدمين الذين تم تحديد password لهم ، مثل root و ahmed . وهو هنا بصيغته مشفرة .

## \* كيفية عمل disable للمستخدم

اولا ، معنى disable هو اغلاق حساب المستخدم على النظام .  
اى ان المستخدم لا يستطيع الدخول الى النظام ، وفى نفس الوقت فان كل ملفاته و العمليات المتعلقة بحسابه تظل موجوده على النظام .

والفرق بين ال disable وال delete لحساب المستخدم -مع انهما متساويين بالنسبة للمستخدم- ان الاولى لا تلغى ملفات وعمليات المستخدم ، فى حين ان الثانىه تلغيهم .

ويتم عمل disable عن طريق خطوه بسيطه جدا ، متمثله فى كتابه الرقم الموجود فى خانه 3 بعد انقاص (يوم) من قيمته ، الى خانه 8 .  
فلو كان الرقم الموجود فى خانه 3 مثلا 12824 فيكتب 12823 فى خانه 8 .

والعله فى هذا هى .

ان خانه 3 يعتبرها النظام بدايه حساب المستخدم ، (عدد الايام الموجوده بها منذ 1970 وهو ال time zero لتشغيل يونكس) اما خانه 8 فتخبر النظام متى سينتهى هذا الحساب (account) فلو انقصنا العدد الموجود بالخانه 8 عن العدد الموجود بالخانه 3 فسيفهم النظام ان حساب المستخدم هذا قد انتهى .

ولا فرق عند النظام ان كان هذا العدد قد تم تحديده امس او من سنه او حتى من 10 سنوات ، بل المهم عنده هو ان الرقم 8 قد طابق الرقم 3 .

\*يوجد بعد ذلك عدة خانات قد يتم التعامل معهم وقد يتركوا كما هم ، ومن هذه الخانات

. الخانه 4 وبها يتم تقييد المستخدم بعدد معين من الايام قبل محاوله تغيير ال password من جديد ، فلا يستطيع تغيير ال passwd الا بعد انتهاء الايام الموجوده بهذه الخانه .

. الخانه 7 (والخاصه بعدد الايام بين انتهاء عمل ال passwd وبين اغلاق الحساب) وتوضع بها قيمتان ،  $< 0 >$  او  $< -1 >$  ، فالاول اى 0 سيقوم بعمل disable للحساب مباشره بعد انتهاء الوقت المحدد لل passwd ، اما -1- فانها توقف هذا ال disable التلقائى للحساب.

. الخانه 8 تستخدم للحسابات المؤقته (اسبوع او شهر) ويتم كتابه الايام المراد ضبط الحساب عليها ، فلو كانت شهر يكتب 30 وان كانت اسبوع يكتب 7 وهكذا .

## ثانيا : التعامل مع المجموعات .

يتم التعامل مع المجموعات بنفس اسلوب التعامل مع المستخدمين .

وهذا الاسلوب يسهل عمليه اداره المجموعات ، لانك بسهولة تستطيع حفظ هذه الاوامر ، فكما انه هناك ثلاثه اوامر لاداره المستخدمين وهم useradd و userdel و usermod .  
فهناك ايضا ثلاث اوامر لاداره المجموعات وهم

|          |                      |
|----------|----------------------|
| groupadd | لاضافه مجموعه جديده  |
| groupdel | لإلغاء مجموعه موجوده |
| groupmod | لتعديل مجموعه موجوده |

ونفس الشئ كذلك مع ملفات النظام الخاصه بالمجموعات ، فهناك ايضا ملفان رئيسيان ، هما

/etc/group وهو مثل الملف passwd  
/etc/gshadow وهو مثل ملف shadow

ملاحظه هامه

اصداره redhat لها طريقه معينه فى انشاء المجموعات ، بعكس الاصدارات الاخرى وهذه الطريقه تتمثل فى

عند انشاء حساب لمستخدم جديد على النظام فان النظام ينشئ له مجموعه بنفس اسمه ، وبالتالي فان كل مستخدم له مجموعه الخاصه ، بعكس الاصدارات الاخرى ، فان هذه الاخيريه تضيف كل المستخدمين الى مجموعه واحده كبيره -وغالبا تسمى users- .

والفرق بين الطريقتين يظهر فى ال permission .  
فمع ردهات فان ال permission للملفات الجديده يكون 002 -قرأه وكتابه للمالك ومجموعته وقرأه فقط للآخرين-  
اما مع الاصدارات الاخرى فان ال permission يكون 022 -قرأه وكتابه للمالك فقط ، وقرأه فقط للباقيين .

وتسمى طريقه redhat هذه بال user private group .

اما الطريقه الاخرى فتسمى shared-group .

## الامر groupadd

هذا الامر هو المسئول عن اضافه مجموعات جديده للنظام ، وصيغته بسيطه وهى

`groupadd <groupname>`

ويأتى معه ثلاث خيارات فقط

**-g** والتي تحدد رقم ال group ، وذلك عند اراده تحديد رقم معين للمجموعه ، وعدم تمكين النظام من وضع الرقم التسلسلى للمجموعات ، ويتبع هذا الخيار الرقم المراد ، ثم اسم المجموعه .  
مثل

`groupadd -g 510 <grpname>`

**-r** وذلك عند اراده جعل هذه المجموعه من مجموعات النظام ، والتي تاخذ ارقام اقل من 500 .

**-f** ويتم الاستعانه بهذا الخيار لعمل force او انشاء المجموعه (بالقوه) وذلك لمنع النظام من اخراج رسائل الخطأ اذا كانت المجموعه مثبتة بالفعل .

## الامر groupdel

وظيفه هذا الامر هو الغاء المجموعه الموجوده بالفعل ، ولا يأتى معه اى خيار ، فقط الامر واسم المجموعه مثل

`groupdel <groupname>`

ويلاحظ انه عند اراده الغاء المجموعه الاساسيه (primary group) للمستخدم الموجود بالفعل على النظام فانه يتم الغاء حساب المستخدم اولا ثم بعد ذلك الغاء مجموعته الاساسيه .

## الامر groupmod

وظيفه هذا الامر هو تعديل المجموعه الموجوده على النظام ، وصيغته هي

`groupmod <groupname>`

ويأتى مع هذا الامر خياران ، وهما

**-g** ويقوم هذا الخيار بتعديل رقم المجموعه المعينه ، ولهذا فانه يأتى بعده الرقم الجديد ثم اسم المجموعه ، وذلك مثل

`groupmod -g 530 <group>`

فما سيحدث هو ان المجموعه المحدده سيتم تغيير رقمها الى هذا الرقم المحدد بال cmd line وهو كما فى هذا المثال سيصبح 530 .

وان كانت القاعده الاساسيه تخبرنا انه لا بد ان يكون هذا الرقم الجديد منفردا ، اى ليس موجود من قبل بالنظام .

فان النظام يكسر هذه القاعده ويتيح للمدير ان يضع رقم غير منفرد -اى ان هذا الرقم موجود بالفعل لمجموعه اخرى- وبالتالي سيكون عندك مجموعتين بنفس الرقم ، وهذه الامكانيه تاتى بالخيار **-o** وصيغته هي

`groupmod -g 502 <group> -o`

فان كان بنظامك مجموعه موجوده من قبل لها رقم 502 ، فانك ستجد انك الان تمتلك مجموعتين لهما نفس الرقم .

ولكن يلاحظ ان تغيير رقم المجموعه امر خطير جدا ، لان كل الملفات -السابقه للتغيير- ستظل محتفظه برقمها الاول ، مما يعنى وجود ملفات لمجموعه ليست موجوده ، ولذلك فعند اراده تغيير رقم المجموعه او حتى اسمها ، فلا بد ان تكون على درايه بما سوف يترتب على هذا التغيير .

**-n** ويتم بهذا الخيار تغيير اسم المجموعه مثل

`groupmod -n <new> <old>`

فيوضع الاسم الجديد بعد الخيار -n ثم بعد ذلك الاسم القديم للمجموعه .

## ملفات المجموعات على النظام .

كما سبق ان ذكرنا ، فان للمجموعات ملفان ، مثلهما مثل ملفات المستخدمين .  
وهذه الملفات هي group و gshadow الموجودان تحت المسار /etc .

### الملف group .

شكل هذا الملف بسيط ، فهو يتكون من اربعة اعمده فقط كما نرى

```
1  2  3  4
root:x:0:root
bin:x:1:root,bin,daemon
daemon:x:2:root,bin,daemon
sys:x:3:root,bin,adm
adm:x:4:root,adm,daemon
tty:x:5:
disk:x:6:root
lp:x:7:daemon,lp
mem:x:8:
kmem:x:9:
wheel:x:10:root
mail:x:12:mail
news:x:13:news
uucp:x:14:uucp
man:x:15:
games:x:20:
gopher:x:30:
dip:x:40:
ftp:x:50:
lock:x:54:
nobody:x:99:
users:x:100:
```

12,1

فاذا ضربنا مثال بالمجموعه الخاصه بالمستخدم root (وهو اول سطر بالملف) فسنجد ان السطر الخاص بهذه المجموعه به فقط اربعة اعمده ، يفصل بينهما بعلامه colon .

1- **groupname** الخانه الاولى خاصه باسم المجموعه .

2- **password** وهو خاص بال passwd لهذه المجموعه .

3- **GID** يتحدد به رقم هذه المجموعه .

4- **users** وهو يوضح المستخدمين المنتمين الى المجموعه .

## الملف gshadow .

هذا الملف شبيه بوظيفه الملف shadow ، ولهذا فانه له نفس اسمه مع اضافه حرف g فى البدايه

ولكن هذا الملف ليس -شديد السريه- مثل الملف shadow .  
ولهذا فاننا نرى ان ال permission الخاص به -وبالملف group ايضا- هو 644 .

وشكل الملف gshadow تقريبا هو شكل الملف group تماما .  
ولكن يتغير شكله بعض الشئ عند وضع password للمجموعه ، فيتحول الحرف x الى ال password المشفره .

وهذا هو شكل الملف gshadow

```
root:x:0:root
bin:x:1:root,bin,daemon
daemon:x:2:root,bin,daemon
sys:x:3:root,bin,adm
adm:x:4:root,adm,daemon
tty:x:5:
disk:x:6:root
lp:x:7:daemon,lp
mem:x:8:
kmem:x:9:
wheel:x:10:root
mail:x:12:mail
news:x:13:news
uucp:x:14:uucp
man:x:15:
games:x:20:
gopher:x:30:
dip:x:40:
ftp:x:50:
lock:x:54:
nobody:x:99:
users:x:100:
```

فهو يعد نسخه طبق الاصل من الملف group .

ومعاني الخانات الاربع التى يتكون منها هذا الملف ، هى ذاتها نفس المعانى لملف group فلا داعى لتكرارها ثانيه .

لاحظ ان الوضع الdefault لجميع المجموعات الموجوده بهذا الملف انه لا يوجد لها passwd .  
سواء كانت مجموعات النظام مثل root و bin وغيرها ، او المجموعات التى يقوم مدير النظام باضافتهم ، فلا تحاول وضع passwd الا اذا كنت تعلم ما الذى تفعله .



## اوامر متعلقه باداره المستخدمين .

يعد الجزء السابق الذى تحدثنا فيه عن اداره المستخدمين واداره المجموعات على النظام كافيا ليستطيع مدير النظام القيام بمهمه اداره المستخدمين . بل ان الكثير من الكتب عندما تتحدث عن اداره المستخدمين ، فانها تقتصر فقط على ما تم ذكره من تفاصيل .

ولكن هناك عده اوامر متعلقه باداره المستخدمين على النظام ، ورأيت لكى تتم الفائدة للقارى ان اكمل هذا الفصل بهذه الاوامر ولو حتى بشكل موجز .

## نقل المستخدمين من نظام الى نظام اخر .

قد تحتاج -لاى سبب كان- ان تنتقل المستخدمين الموجودين على نظامك الى نظام اخر ، كأن قام مثلا المسؤولين عن النظام بتغيير الاصداره التى كنت تعمل عليها . او قد تريد -فى حالات اخرى- الاعتماد على ملف واحد فى اداره المستخدمين بدلا من ملفين .

وايما كان السبب ، فان النظام يتيح لك استخدام الملف passwd فقط دون الملف shadow . وايضا يتيح لك استخدام الملف group فقط دون الملف gshadow .

ويتم ذلك عن طريق عده اوامر ، هى

**pwunconv**

فهذا الامر يكتب منفردا ، بدون اى option او argument ، ويتمثل الناتج منه فى انه يحول ال password الموجوده بالملف shadow الى الملف passwd . وبالتالي سيتم الغاء الملف shadow ، والاعتماد فقط على الملف passwd .

ويتم ارجاع الوضع كما كان عن طريق الامر

**pwconv**

فتجد ان الملف shadow قد تم كتابته ثانيه ، وتم ايضا تحويل (convert) ال password اليه

هذا بالنسبه للمستخدمين وملفاتهم ، ونفس الاجراء يتم للمجموعات و باوامر مشابهه ، وهى

**grpunconv** وبتنفيذه يتم الاعتماد فقط على الملف group .

وايضا

**grpconv** لارجاع الملف gshadow .

## التعرف على بياناتك .

هناك امران يتم التعرف بهما على بياناتك وهما

### id

ويكتب هذا الامر اما منفردا ، او باضافه اسم مستخدم بعده .  
والناتج من هذا الامر يعرض

- اسم المستخدم ورقمه
- اسم مجموعته ورقمها
- المجموعات التى يشترك فيها المستخدم

وكما نرى فى المثال

```
[root@elnajeeb root]# id
uid=0(root) gid=0(root) groups=0(root),1(bin),2(daemon),3(sys),4(adm),6(disk),10(wheel)
[root@elnajeeb root]# su - ahmed

[ahmed@elnajeeb ahmed]$ id
uid=500(ahmed) gid=500(ahmed) groups=500(ahmed)
[ahmed@elnajeeb ahmed]$ _
```

فعند تطبيق هذا الامر مع المستخدم root ، نراه قد عرض المعلومات الثلاثه وهى uid و gid و groups (وهى المجموعات التى يشترك فيها المستخدم root)

ثم بعد ذلك تحولت الى المستخدم ahmed وطبقت نفس الامر عليه ، فكان الناتج متماثل الا فى المجموعات التى يشترك فيها المستخدم ahmed ، لانه غير مشترك فى ايه مجموعات -باستثناء مجموعته طبعا-

والامر الثانى هو

### groups

وناتجه مثل المعلومه الثالثه التى يظهرها الامر id ، بمعنى انه يظهر المجموعات التى يكون المستخدم مشترك فيها . هذا اذا تم كتابته منفردا .  
اما فى حاله اضافته اسم مستخدم بعده ، فانه يعرض المجموعات لهذا المستخدم .

والامر id قد ياخذ العديد من ال option والتى تعدل فى شكل ناتجه ، ويمكن الاطلاع عليهم من صفحات المساعدة .

اما الامر groups فلا ياخذ اى option ، فقط ياخذ argument متمثل فى اسم المستخدم .

## التحول الى مجموعه اخرى .

عندما يدخل المستخدم الى النظام فانه يدخل تلقائيا الى المجموعه ال default له ، او بمعنى اخر ال primary group .  
وعندما يكون هذا المستخدم له اشتراك فى اكثر من مجموعه ، فان النظام يسمح له بالتحول الى هذه المجموعه عن طريق الامر

### newgrp

فبكتابه هذا الامر وإتباعه باسم المجموعه التى يريد المستخدم التحول اليها ، فان النظام يحوله الى هذه المجموعه .

والملاحظ هنا ان المستخدم يتحول فقط الى المجموعه الاخرى وبالتالى فان كافه بياناته تظل كما هى فيما عدا العمليات التى يجريها ، مثل انشاء الملفات ونحوه ، فانها تكون فى هذه الحاله مملوكه للمجموعه التى تغير اليها .

ويمكن للمستخدم ان يرجع ثانيه الى مجموعته الاولى عن طريق الامر exit او عن طريق الامر newgrp فقط بدون اى اسم بعده .

## وضع password للمجموعه .

يتم وضع password للمجموعه عن طريق الامر

### **gpaswd**

وهو كما نرى ، مشابهه للامر passwd -الخاص بالمستخدمين- مع اضافته حرف g فى البدايه .

ويكتب اسم المجموعه المراد عمل passwd لها بعد هذا الامر ، مثل

`gpaswd <group>`

ويتم التنبيه للمره الثانيه ان المجموعات بطبعها ال default لا يتم وضع password لها ، ولكن عند اراده وضع password لاي مجموعه ، فلا بد ان تكون على درايه بما تفعل .

والامر gpaswd لا يستخدم فقط لوضع password للمجموعه ، بل ان له استخدام اخر اكثر اهميه ، وهو انه يمكن بهذا الامر اضافته مستخدم الى مجموعته بعينها . بالاضافه الى عدده وظائف اخرى لهذا الامر وهى كما يلى

`gpaswd -a <user> <group>` لاضافه مستخدم الى مجموعته

`gpaswd -d <user> <group>` لازاله مستخدم من مجموعته

`gpaswd -A <user> <group>` وضع Adminsrator للمجموعه

`gpaswd -M <user> <group>` اضافته Member للمجموعه

`gpaswd -r <group>` لازاله ال password الخاصه بالمجموعه

`gpaswd -R <group>` لمنع الدخول للمجموعه بالامر newgrp

ولعل الملاحظ ان اوامر المجموعات التى سبق شرحها فى الصفحات السابقه مثل groupmod (واخوتها) ، لا تقوم بهذه الوظيفه (وهى الاضافه الى المجموعه) بل يقوم بهذا الامر gpaswd .



## الترتيب الهرمي لنظام الملفات . File system Hierarchy Standard

يعد الترتيب الهرمي لنظام الملفات في لينوكس من الاشياء "المحيره" للمتعامل الجديد مع لينوكس ، وهذا بالطبع للمستخدم الذى اعتاد العمل مع نظام التشغيل windows .

ويطلق على هذا الشكل file system hierarchy standard او اختصارا FHS .

وهذا الشكل (الترتيب الهرمي) شبيه بالشجره .  
فللنظام جذر رئيسى ، ويتفرع من هذا الجذر بقيه الافرع المكمله للنظام .

واذا اردنا مثالا يوضح معنى الجذر وافرعه ، فاننا يمكن الاستعانه بالعنوان الذى يكتب فى ال address bar الموجود فى نظام windows .

فلنفترض ان بداخل احد ال partition -ولنفترض انه ال C partition- يوجد مجلد يسمى directory وداخل هذا المجلد ملف يسمى file .

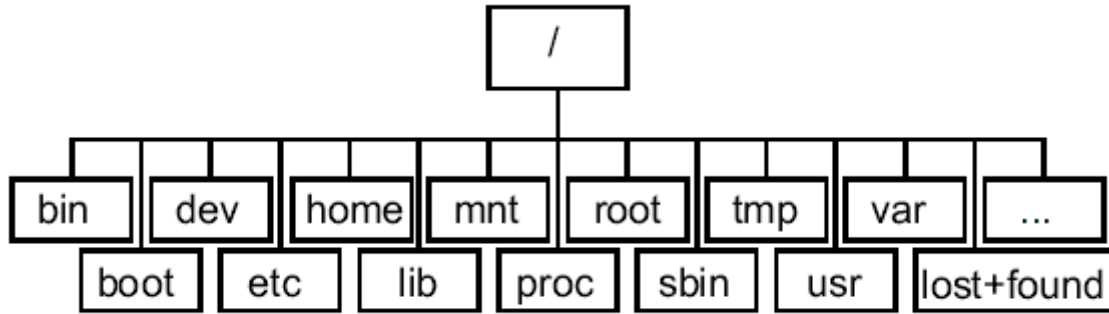
ولكى تصل الى هذا الملف ستستخدم بالطبع ال mouse -كطريقه سهله فى التعامل مع الويندوز- وستقوم بالضغط double click على my computer اولا ، ثم بعد ذلك على C partition ثم تتبع ذلك بالضغط على المجلد directory ، ومن ثم ستجد نفسك امام الملف file .

كل هذه الخطوات تراها قد ظهرت فى ال address bar بهذا الشكل C:\directory\file .

وهذا بالضبط معنى الجذر والافرع ، فالجذر الرئيسى فى لينوكس قد نعتبره هذا ال C partition وكل فرع تحت هذا الجذر -مثل etc و lib وغيرهم فى لينوكس- شبيه بهذا المجلد وبهذا الملف .

اذا فان معنى المصطلح /etc ، هو ان المجلد etc موجود تحت / .  
ومعنى /etc/passwd هو ان الملف passwd موجود تحت etc وهو بدوره موجود تحت / وهكذا . وهذا هو ما يسمى بال path اى المسار.

وفى السطور القادمه سنتعرض لنظام الملفات هذا بصورة مختصره وما يعنيه كل فرع من افرعه . وسيتم عرض هذه الاجزاء بالاعتماد على الترتيب الابدئى .



1- / وهذا هو الجذر الرئيسى للنظام ولبقية الافرع الاخرى ، ويطلق على هذا الجذر عدة اسماء فيطلق عليه ال root باعتباره جذر النظام .  
ويطلق عليه slash تمييزا له بالاعتماد على شكله .  
\* وهذا ال / من الاجزاء الضرورية (mandatory) للتكوين الهرمى للنظام ، بل انه يعد اهم جزء فى النظام .

2- /bin ويحتوى هذا المجلد على البرامج والاورام-الهامة- التى يحتاج اليها مدير النظام فى ال single user mode وفى حاله ال rescue (الطوارئ) ايضا .  
وهذا المجلد لا يحوى مجلدات فرعية تحته (subdirectory) بل يحوى الاوامر فقط وهى ما تسمى ب binary .  
\* وهذا المجلد من الاجزاء الضرورية للنظام (mandatory)

3- /boot ويحتوى هذا المجلد على جميع الملفات التى يحتاجها النظام لكى يستطيع ان يبدأ العمل فهو يحتوى على نسخه من الكرنل ، وهى التى يتم تحميلها فى ال Ram وهى الخطوه الاولى والبسيطة لبدايه نظام التشغيل ، ثم يتبعها النظام بتحميل كل ملفاته ، وذلك طبقا لما هو موجود بالملفات fstab و inittab .  
ونسخه الكرنل الموجوده فى هذا المجلد تسمى vmlinuz ويتبعها بقيه رقم الكرنل .  
وهى نسخه مصغره من الكرنل ، وظيفتها فقط هى تحميل هذا الكرنل -الصغير- فى الذاكرة Ram ثم يقوم بعد تحميل نفسه بالذاكره ، بارشاد الكرنل الاساسى للنظام لكى يبدأ عمله .  
\* وهذا المجلد يعد ضرورى للنظام (mandatory)

4- /dev ويحتوى هذا المجلد على اجزاء الهاردوير التى تسمى devices ، وهى عباره عن ملفات -لان كل شى بالنظام هو عباره عن ملف- ، وتشمل كل انواع ال devices الموجوده بالنظام ، من امثال hard disk و floppy disk و terminals وغيرهم من ال devices .  
ويعبر عنهم باختصارات من امثال hd او fd وهكذا .  
\* وهذا المجلد ضرورى (mandatory)

5- /etc ويحوى هذا المجلد كل ملفات التهيئه الخاصه بالنظام . وتشمل كل ملفات التهيئه ، سواء كانت لل boot او لل x window او الشبكات والسيرفرات .  
وهذا المجلد يحوى ملفات و مجلدات . وينطق (اى تى سى) .  
\* وهذا المجلد ضرورى (mandatory)

**6- /home** وهذا المجلد هو ال home directory لجميع المستخدمين على النظام .  
وهم مقسمون الى مجلدات فرعيه تحت هذا المجلد ، مثل /home/user1 فلكل مستخدم مجلد خاص به ، ويتميز كل مجلد عن الاخر باسم المستخدم على النظام .  
وغالبا ما يكون هذا المجلد ، الاكبر حجما على النظام ، لانه من المتوقع ان ينمو باستمرار .  
\* وهذا المجلد اختياري (optional)

**7- /lib** ويحتوى هذا المجلد على المكتبات التى تعتمد عليها الاوامر والبرامج فى عملها ، ولهذا السبب فان هذه المكتبات تسمى shared libraries .  
وفكره هذه ال shared libraries فى انها تحتوى على الاجزاء التى تتكرر فى الكود لهذه البرامج ، مما يجعل المبرمجين يتفادون كتابه هذه الاجزاء مع كل برنامج على حده .  
وهذه ال libraries هى التى يحتاجها النظام فى حالات مثل ال single user mode وحالات الطوارئ ، اما هذه التى يحتاجها النظام فى الاعمال الاقل اهميه فتوجد تحت /usr .  
\* وهذا المجلد ضرورى (mandatory)

**8- /mnt** ويحتوى هذا المجلد على ال temprory mount او الاجزاء التى تلحق بالنظام بشكل مؤقت ، مثل floppy و cdrom .  
وايضا قد يشمل ال partition لانظمه التشغيل الاخرى التى تتعامل مع النظام ، مثل ويندوز .  
\* وهذا المجلد اختياري (optional)

**9- /opt** وهذا المجلد مخصص للبرامج التى تضاف للنظام والتى تسمى بال third party اى التى ليست من انتاج الشركه التى قامت باخراج النظام .  
\* وهذا المجلد اختياري (optional)

**10- /proc** يحوى هذا المجلد نظام ملفات -غير حقيقى- وهو ما يسمى بال virtual filesystem . والسبب فى كونه virtual هو انه لا يوجد بالفعل على ال hard disk .  
بل انه يحوى process اى عمليات ، ومعلومات الكرنل التى يعمل الان على النظام .  
وتعد هذه المعلومات هامه جدا فى اوقات ال trouble shotting ، خاصه المتعلقه بالهاردوير ، واهم المعلومات التى قد تحتاج اليها هى interrupts و devices و I/O ports .  
\* وهذا المجلد اختياري (optional)

**11- /root** وهذا المجلد هو ال home directory لمدير النظام .  
فيوجد به العديد من ملفات التهيئه التى يقوم المدير بانشائها لاداره نظامه .  
والسبب فى انشاء هذا ال home لل root -مع ان النظام باكملة تحت ادارته- هو منع امتلاء ال / الاساسى للنظام بملفات التهيئه التى ينشأها المدير للمهام المختلفه .  
\* وهذا المجلد اختياري (optional)

**12- /sbin** وهذا المجلد مثل المجلد bin ، الا ان هذا يحتوى على الاوامر والملفات التى تعد من ادوات اداره النظام ، من امثال mkfs و shutdown و qoutaon وغيرها من اوامر النظام .  
\* وهذا المجلد ضرورى (mandatory)

**13- /tmp** وهذا المجلد خاص بالملفات المؤقتة التي تنشأها البرامج والوامر المختلفه اثناء ادائها لوظائفها .  
\* وهذا المجلد ضرورى (mandatory)

**14- /usr** يعد هذا المجلد من المجلدات الهامه والتي -لا بد ان- يخصص لها مساحه كبيره .  
ذلك لان هذا المجلد توجد به البرامج التي يحتاجها النظام فى اعماله الاعتياديه ، بمعنى انها لا يحتاج اليها النظام اثناء عمليه ال start up ولا ال emergency .  
ولهذا المجلد شكل هرمى مشابه للموجود تحت / ذات نفسه .  
ولان هذا المجلد لا يحتاجه النظام فى عمليه ال start up ، فانه - فى الانظمه ذات المساحات المحدوده فى ال hard disk- يتم ربطه (mount) بالنظام من خلال الشبكه .(اى انه يوجد على السيرفر الرئيسى ليخدم بقيه الاجهزه بالشبكه ، بدلا من تكرار وضعه على كل جهاز على حده)  
ويتم ايضا ربطه (mount) كقراء فقط . read only .  
\* وهذا المجلد ضرورى (mandatory)

**15- /var** وهذا المجلد يحتوى على الملفات والمجلدات التي يتغير حجمها وبياناتها باستمرار .  
ملفات ال login -والتي تتغير كلما قام النظام بعملية login جديد- يتم تخزينها فى هذا المجلد .  
بالاضافه الى ملفات ال printer -التي تتغير بياناتها باستمرار- وغيرها من الملفات .  
اما بالنسبه للمجلدات ، فيوجد مجلدات مختلفه تخدم العديد السيرفرات ، مثل ال ftp server وهو الذى يخدم سيرفر ftp ، ايضا سيرفر الاباتشى والذى توضع ملفاته فى المجلد www وغيرهم .  
\* وهذا المجلد ضرورى (mandatory)

**16- lost+found** يوجد هذا المجلد فى كل partition موجود على النظام ، فان كان فى نظامك -على سبيل المثال- 7 partition فانك ستجد هذا المجلد 7 مرات .  
ووظيفه هذا المجلد هى ، عندما يقوم البرنامج fsck بعمل check على ال filesystem بعد عمليه ال system crash فان من وظائفه -fsck- ان يقوم بارجاع كل الملفات الى اماكنها الطبيعیه فى النظام ، فان صادف ولم يتعرف على ملفات بعينها ، فانه يضعها فى هذا المجلد ، تاركا الامر الى مدير النظام ليقوم هو بنفسه بارجاعها ، او استبدالها اذا لزم الامر .

كلمه المجلد التي تم ذكرها مع كل ال filesystem قد تعتبر مرادفه لكلمه partition ، ولكن كلمه مجلد اعم فى استخدامها ، والسبب هو ، ان كل ما هو موجود تحت ال ( / ) عباره عن مجلد ، والبعض منه فقط partition .  
بمعنى اخر، اذا كان للمجلد etc partition على الهارد ، اذا فان هذا ال partition هو ال mount والمجلد etc هو ال mount point . وان لم يوجد له partition فهو عباره عن مجلد تابع لل ( / )

## ملخص .

- / .** هو الجذر الرئيسى للنظام .
- /bin .** به جميع الاوامر العاديه التى يحتاجها النظام فى حاله الطوارئ .
- /boot .** يحتوى على كل الملفات المطلوبه وقت ال booting .
- /dev .** كل ال devices موجوده تحته .
- /etc .** به جميع ملفات التهيئه .
- /home .** مجلدات المستخدمين .
- /lib .** يحوى جميع المكتبات المشتركه .
- /mnt .** خاص بال mount المؤقت (مثل cd و floppy) .
- /opt .** للبرامج الغير تابعه للاصداره التى تعمل عليها .
- /proc .** تكتب به المعلومات الخاصه بالكرنل الذى يعمل حاليا .
- /root .** ال home directory الخاص بمدير النظام .
- /sbin .** مثل /bin ولكنه خاص باوامر إداره النظام .
- /tmp .** الملفات المؤقته او ال temprory .
- /usr .** به جميع البرامج التى يستخدمها النظام فى عمله العادى .
- /var .** يوجد به الملفات والمجلدات التى تتغير باستمرار .
- /lost+found .** للملفات الناتجه من برنامج fsck .

ملاحظه .

الفرق بين مجلدات مثل /bin و /sbin و /lib وغيرهم من المجلدات التى توجد تحت ال / وايضا تحت /usr ، هو عندما يحدث للنظام اى مشكله ، ولنفتراض مثلا انه لا (يبوت -booting) فان ما نقوم به هو اما جعل النظام ييبوت من start up floppy ، او من الاسطوانه ولكن ك rescue mode .  
وايا كانت الطريقه ، فهذا لا يهم ، بل ان المهم هو ، شكل النظام بعد الدخول اليه بهذه الطريقه ؟  
فانت ستدخل الى النظام ولكن ليس كالدخول العادى الذى يحدث فى الاوقات العاديه ، بل ان هذا الدخول يسمى دخول ولكن فى حاله الطوارئ .  
وما يحدثه هذا الدخول هو عمل mount للجذر الرئيسى للنظام ( / ) فقط ، وبالتالي عمل mount لكل المجلدات الموجوده تحته . وفى هذه الحاله فانت ستستخدم الاوامر الموجوده فى المجلدات /bin و /sbin وغيرهم .  
وبعد حل المشكله ، وعمل reboot ، والدخول للنظام فى حالته الطبيعيه ، ففى هذه الحاله سنقوم باستخدام الاوامر الموجوده فى المجلدات سالفه الذكر وايضا اولئك الموجوده تحت /usr .  
\* <اذا فان الاوامر الموجوده فى /bin وغيرها ، هى الاوامر التى بالكاد ، تحتاجها فقط فى حالات الطوارئ>

