fedora™ magazine -LATAM

Una revista para Latinoamericanos
hecha por latinoamericanos

Nº 0007
Año 01

STORAGE BARATO

y fácil de usar

LVM
Administrando
Volúmenes
lógicos

Programando
en Fedora con
NETBEANS

SCREEN
Un multiplexor
de terminales

RUBY
Capítulo I:El
primer
contacto
(Continuación)

fedora

PREGUNTA

RESPONDE

COMPARTE

Interactúa con la comunidad Fedora

Visita el canal IRC #fedora-latam

http://proyectofedora.org/chat

E D I T O R I A L

El Proyecto Fedora ya lleva varios años con iniciativas que intentan de alguna manera aportar a la comunidad de usuarios de esta distro con mejoras, organizaciones, eventos, soporte y por que no fomentando buenas relaciones lo cual es muy loable para cada quien que realice tal tarea.

Sin embargo, los actores pocas veces cambian teniendo que hacer hasta lo imposible por cumplir con su compromiso como siempre lo han hecho.... siempre. Y es aquí en donde quiero detenerme y preguntar ¿no han habido cambios suficientes como para traer mentes mas frescas? Si bien la respuesta es obvia no es concordante con el escenario actual.

Es por eso que me dirijo a ti lector, a ti que tienes las ganas, a ti que te sientes capaz de tomar el desafío, para decirte: únete! Únete a este gran proyecto que es Fedora LATAM y así poder difundir tu labor por el mundo, que todos sepan que eres tu quien da un paso mas por hacer un cambio en esta ruleta tan rutinaria, que eres tu quien quiere dejar una huella en la historia.

Y no lo digo solo por el hecho de aparecer en una revista o algo parecido, sino por que se siente bien de verdad participar de un proyecto que esta haciendo futuro día a día con el solo hecho de organizarse. Y esperamos que ese futuro sea un futuro libre y colaborativo, en el cual las sociedades tomen conciencia del aporte comunitario y la filosofía del software libre.

Es por esto y por muchas razones mas que tenemos por objetivo como proyecto convertirnos en una herramienta de difusión de noticias, desarrollos de nuevas tecnologías, eventos y todo lo que compete a Fedora para así hacer partícipes a todos nuestros usuarios de este, por que no decirlo, estilo de vida que significa implicarse dentro de este mundo basado en la colaboración y en la libertad del conocimiento.

Alguien comenzó con esto en un tiempo con el fin de que usuarios como tu lo usen un dia.... ¿Quieres ser tú el que lo continúe?

Mauricio Moreno R.
Fedora Ambassador (Chile)
http://fedoraproject.org/wiki/User:V1k1n60

El primer contacto (Continuación)

Por Guillermo Gómez Savino
http://fedoraproject.org/wiki/User:gomix

Recapitulación rápida, los objetos y sus métodos, ¿documentación?

Teniendo cualquier objeto de cualquier clase a la mano, para acceder a su métodos podemos usar la sintaxis objeto.método. En el ejemplo abajo llamamos al popular método to_s que nos ofrece una representación en String del objeto en cuestión.

```
1 >> hash.to_s
2 => "1hoytemperatura75colorrojo"
3 >> arreglo.to_s
4 => "a1b2"
5 >> 8.to_s
6 => "8"
```

Ahora bien, ¿dónde consigo la documentación de dichas clases y métodos? Demos la bienvenida a ri para ayuda local en línea de comandos, y por supuesto, en la web a http://www.ruby-doc.org/ . Francamente, en nuestros días la primera fuente de información es la Web, y en segundo lugar, nuestros recursos locales como ri. Entonces, y para efectos de esta serie de artículos, vamos a usar inicialmente Ruby 1.8.7.

http://www.ruby-doc.org/core-1.8.7/
http://www.ruby-doc.org/stdlib-1.8.7/

Ejemplo de salida ri (extracto).

```
$ ri Fixnum
-------------------------------------------------
Class: Fixnum < Integer
         A +Fixnum+ holds +Integer+ values that can be represented in a
         native machine word (minus 1 bit). If any operation on a +Fixnum+
         exceeds this range, the value is automatically converted to a
         +Bignum+.
    ...
    Instance methods:
    -----------------
         %, &, *, **, +, -, -@, /, <, <<, <=, <=>, ==, >, >=, >>, [], ^,
         __serialize__, abs, dclone, div, divmod, html_safe?, id2name,
         modulo, power!, quo, rdiv, rpower, size, to_f, to_s, to_sym, xchr,
         zero?, |, ~
```

Ruby es dinámico ¿ 2 + 2 = 4 ?

Uno de los aspectos notables de Ruby es su dinamismo, una forma de visualizarlo es hacer uso del hecho que todas las clases están "abiertas" y es posible redefinir sus métodos, por ejemplo:

```
fixnum_mod.rb
#!/usr/bin/ruby
#
class Fixnum
  def +(otro)
    100
  end
end
puts (2+2).to_s
```

Y ahora ejecutamos nuestro programa:

```
$ ruby fixnum_mod.rb
100
```

Horror, hemos echado a perder el método sumar de Fixnum, por ello algunos consideran peligroso los lenguajes dinámicos, sin embargo existe la forma de protegernos de este tipo de modificaciones en el caso de que ello no sea deseable. El ejemplo sin embargo demuestra que toda clase puede redefinir cualquiera de sus métodos en tiempo de ejecución, en cualquier momento, este dinamismo le da gran poder a Ruby, piense en una clase u objeto que evoluciona y gana funcionalidad en el tiempo de existencia del programa, o que la pierde, su funcionalidad puede mutar, cambiar. No puede devolver el cambio, no sin que le enseñe cómo preservar el código sobrescrito, probablemente en la próxima edición de esta columna.

Estructuras de control

Por supuesto que ningún lenguaje está completo si no tiene la capacidad de ejecución de código condicionada, es decir, evaluar alguna condición o estado, y proceder en consecuencia de distintas maneras. Abajo le resumimos las estructuras más comunes.

```
1 # Evaluacion máxima 20
2 if evaluacion < 10
3       puts "Usted reprobó la asignatura."
4 elsif evaluacion > 16
5       puts "Usted obtuvo un grado sobresaliente."
6 else
7   puts "Usted aprobó la materia."
8 end
```

```
1 unless unaCancion.duracion > 180 then
2   costo = .25
3 else
4   costo = .35
5 end
```

```
1 case forma
2   when Cuadrado, Rectangulo
3     # ...
4   when Circulo
5     # ...
6   when Triangulo
7     # ...
8   else
9     # ...
10 end
```

Lazos e iteradores

Un iterador en Ruby es simplemente un método que puede invocar un bloque de código. Note como se pasa la referencia del bloque de código y este a su vez ejecutado por medio de la llamada yield.

Ruby

```
1 def tres_veces
2   yield
3   yield
4   yield
5 end
6
7 tres_veces { puts "Hola Mundo" }
```

Salida

```
Hola Mundo
Hola Mundo
Hola Mundo
```

Algunos iteradores son muy comunes en muchas clases Ruby para representar colecciones, por ejemplo each en un arreglo simple. Note que each además de iterar por cada uno de los elementos del arreglo, pasa un argumento al bloque de código ha ser ejecutado, en este caso pasa el contenido correspondiente en el arreglo.

Ruby

```
1 [1,3,5,7,9].each { |i| puts i }
```

Salida

```
1
3
5
7
9
```

Lazos con while y until.

Ruby	Salida
1 peso = 5	10
2 while peso < 100	20
3 peso = peso * 2	40
4 puts peso	80
5 end	160

Ruby	Salida
1 peso = 5	10
2 until peso > 100	20
3 peso = peso * 2	40
4 puts peso	80
5 end	160

Lazo con for y loop:

Ruby	Salida
1 for i in 1..8 do 2 puts i 3 end	1 2 3 4 5 6 7 8

Ruby	Salida
1 for i in 1..8 do 2 puts i 3 end	1 2 3 4 5 6 7 8

De salida, por supuesto hay mucho más acerca de Ruby y su mundo, usted podrá hacer desde pequeños guiones (scripts) hasta poderosas aplicaciones de escritorio o su último desarrollo web, no se detenga aquí y espero que nos leamos en la próxima entrega de esta publicación, envíeme sus comentarios a <gomix@fedoraproject.org>.

Nuestro trabajo es resolver problemas concretos, no alimentar al compilador con cucharilla, nos gustan los lenguajes dinámicos que se adapten a nosotros sin reglas rígidas que seguir.

Gomix_1


Somos Fedora
www.proyectofedora.org

iSCSI : Storage barato y fácil de usar

Antonio Sebastián Sallés M.
http://fedoraproject.org/wiki/User:asalles

El problema radica en que para realizar dicha tarea necesitamos un servicio adicional (ej, NFS, Samba, entre otros) el cual genera una capa de abstracción que se encarga de traducir las peticiones y orquestar los accesos a disco, perdiendo muchísimo performance y desperdiciando recursos a destajo. Hagan la prueba: escriban sobre un recurso NFS compartido montado y se darán cuenta que la velocidad es muchísimo mas lenta si la comparamos a estar escribiendo directamente en una partición montada. Peor aún si tienen hardware viejo, no van a llegar a unos cuantos megas por segundo de escritura.

> Comúnmente compartimos directorios con el fin de que uno o varios clientes puedan escribir y leer de forma paralela, y es a mi parecer una solución práctica.

Arquitectura básica

Como supongo que ya saben, SCSI (Small Computer System Interface, http://www.ietf.org/rfc/rfc3720.txt) es la interfaz estándar para transferencia de datos entre periféricos usando el bus del computador, osea es por ejemplo la forma de transmitir datos entre el disco duro y la placa madre. Ahora, imagínense transmitir dicho protocolo por internet, podríamos compartir directamente una partición (o logical volume) que se encuentre en nuestra máquina y permitir que otra máquina remota pueda verlo como un simple dispositivo de bloques, utilizando la red para transferir la data y eliminando la capa de abstracción que generaría un servicio de por medio. La escalabilidad de acceso al dispositivo es tan amplia que incluso podríamos darle acceso a la máquina remota para que pueda formatear la partición en cuestión. A nivel de arquitectura, el driver iSCSI provee las mismas funciones que un canal de fibra, pero con la diferencia de que en vez de utilizar una HBA, utilizamos una simple tarjeta Ethernet. Claramente si necesitamos una solución de alta disponibilidad y queremos eliminar todos los puntos de fallo sólo necesitaremos realizar un bonding de las interfaces de red y listo.

El mercado

El driver iSCSI apareció hace algunos años como alternativa a las soluciones de storage existentes en el mercado, que tienen como principal diferencia utilizar fibra óptica y además ser excesivamente caras, incluso inaccesibles para Pymes o personas comunes. De esta forma iSCSI es la solución perfecta para quienes necesiten por ejemplo adosar un storage compartido a un cluster, recurso que si además lo formateamos en GFS podríamos permitir que los nodos participantes puedan realizar lecto-escritura simultánea sin tener un servicio de por medio y con una excelente performance.

Tutorial iSCSI

A continuación les explicaré como configurar un servidor iSCSI, como configurar un cliente que use un recurso iSCSI compartido y algunos tips de por medio.

El famoso target

Lo primero que haremos es armar el target, que es el server (1.1.1.1) que va a compartir el recurso, que en este caso es un logical volume.

1. Creamos un logical volume llamado iscsi, de 10GB, el cual posteriormente será exportado.

```
[ root@server ]# lvcreate -L 10G -n iscsi
vg_local
```

2. Instalamos scsi-target-utils

```
[ root@server ]# yum install -y scsi-
target-utils
```

3. Ahora tenemos que modificar /etc/tgt/targets.conf para crear la LUN a exportar.

Dentro de la configuración agregaremos el LV que acabamos de crear (/dev/vg_local/iscsi), el IQN (iSCSI Qualified Name) el cual le llamaremos 'iqn.2011-04.com.example.storage:iscsi ' y agregarmos que sólo el cliente 1.1.1.2 podrá utilizarlo.

```
[ root@server ]# cat <<FIN >>
/etc/tgt/targets.conf<target iqn.2011-
04.com.example.storage:iscsi>backing-store
/dev/vg_local/iscsi initiator-address 1.1.1.2
</target>
FIN
```

4. Iniciamos el servicio llamado tgt, el cual que se encargará de entregar el recurso. Obviamente es de suma importancia dejarlo activo al inicio para evitar que los clientes se cuelguen y/o arrojen un error de acceso al dispositivo.

```
[ root@server ]# service tgtd start
[ root@server ]# chkconfig tgtd on
```

5. Para comprobar que está todo funcionando bien podríamos utilizar tgt-admin. Cabe recalcar que por cada LUN que exportemos el sistema creará otra LUN adicional, de 0MB que será utilizada por el servicio para fines de control.

```
[ root@server ]# tgt-admin -s
Target 1:  iqn.2011-
04.com.example.storage:iscsi
  System information:
    Driver: iscsi
    State: ready
  I_T nexus information:
  LUN information:
  LUN: 0
    Type: controller
    SCSI ID: deadbeaf1:0
    SCSI SN: beaf10
    Size: 0 MB
    Online: Yes
    Removable media: No
    Backing store: No backing store
  LUN: 1
    Type: disk
    SCSI ID: deadbeaf1:1
    SCSI SN: beaf11
    Size: 10240 MB
    Online: Yes
    Removable media: No
    Backing store: /dev/vg_local/iscsi
  Account information:
    ACL information:
      1.1.1.2
```

El cliente

1. En el cliente debemos primero

instalar el paquete 'iscsi-initiator-utils ', que nos entregará las herramientas para acceder al target.

```
[ root@client ]# yum install -y iscsi-
initiator-utils
```

2. Luego debemos configurar un alias en el archivo /etc/iscsi/initiatorname.iscsi con el fin de que el servicio tgt pueda conversar de forma correcta.

```
[ root@client ]# echo
"InitiatorAlias=cliente1" >>
/etc/iscsi/initiatorname.iscsi
```

3. Iniciamos el servicio iSCSI y lo dejamos activo.

```
[ root@client ]# chkconfig iscsi on ;
service iscsi start
```

4. Si no encontramos error alguno podemos empezar a buscar los targets que acabamos de configurar.

```
[ root@client ]# iscsiadm -m discovery -t
sendtargets -p 1.1.1.1
1.1.1.1.:3260,1 iqn.2011-
04.com.example.storage:iscsi
```

La salida de del comando debería mostrar los dispositivos de bloques disponibles en la siguiente estructura: <target_IP:port> <target_iqn_name>. En este caso ve sin problemas el target configurado y sólo tenemos que adosarlo como un disco duro más.

5. Para realizar la conección al recurso desde el cliente ejecutamos el comando iscsiadm agregando el parámetro de login al final (-l) activando la opción de realizar un adosamiento del dispositivo de bloques compartido.

```
[ root@client ]# iscsiadm -m node -T
iqn.2011-04.com.example.storage:iscsi
-p 1.1.1.1 -l
```

6. Listo. Ahora tenemos el dispositivo adosado y ya podemos empezar a utilizarlo. Para ello podemos analizar si existe con el comando 'fdisk -l' y con el mismo comando empezar a crear las particiones, para luego formatearlo con mkfs.

```
[ root@client ]# fdisk -l | grep GB
...
Disk /dev/sdb: 10 GB, 10001569 bytes
...
[ root@client ]# fdisk /dev/sdb
...
[ root@client ]# partprobe
[ root@client ]# mkfs.ext3 /dev/sdb1
...
```

7. Finalmente si queremos agregarlo a fstab para que el dispositivo se monte durante el reinicio del sistema tenemos que tener en cuenta que hay que agregarle el parámetro '_netdev'.

```
[ root@client ]# cat <<FIN >> /etc/fstab
/dev/sdb1    /punto/de/montaje    ext3
_netdev    0 0
FIN
```

8. Una vez realizado el adosamiento del dispositivo, éste queda activo incluso despues de un reinicio. Si no lo queremos usar más entonces podemos desactivarlo agregando la opción '-u' al comando iscsiadm de la siguiente forma:

```
[ root@client ]# iscsiadm -m node -T
iqn.2011-04.com.example.storage:iscsi
-p 1.1.1.1 -u
```

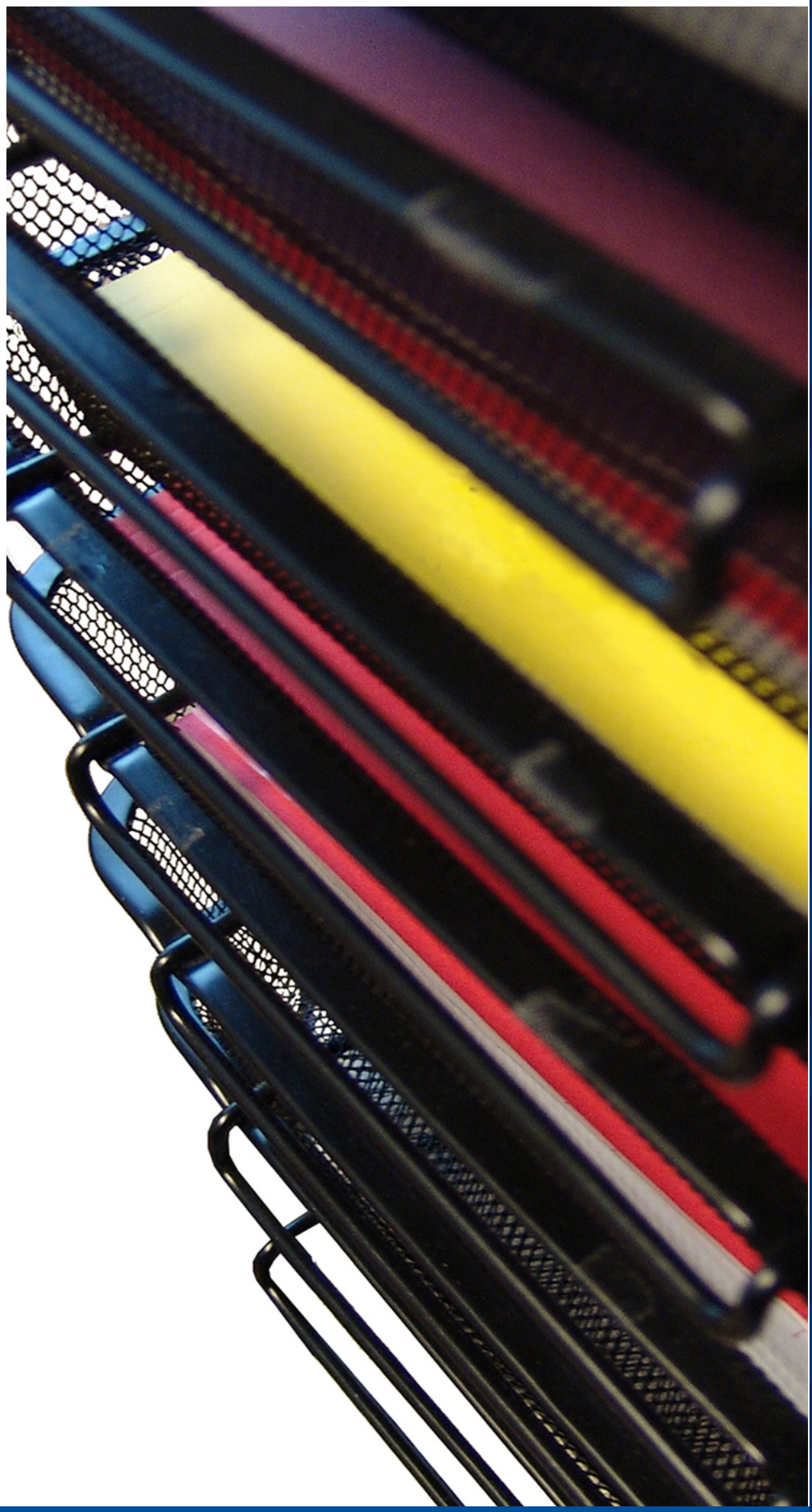
Síntesis

El driver iSCSI es muy util si queremos montar dispositivos y acceder a ellos a gran velocidad utilizando de por medio la tarjeta de red. Obviamente si tenemos una NIC lenta tendremos limitaciones, pero en caso de utilizar tarjetas de 1GB o superior podríamos tener tasas de transferencias altísimas.

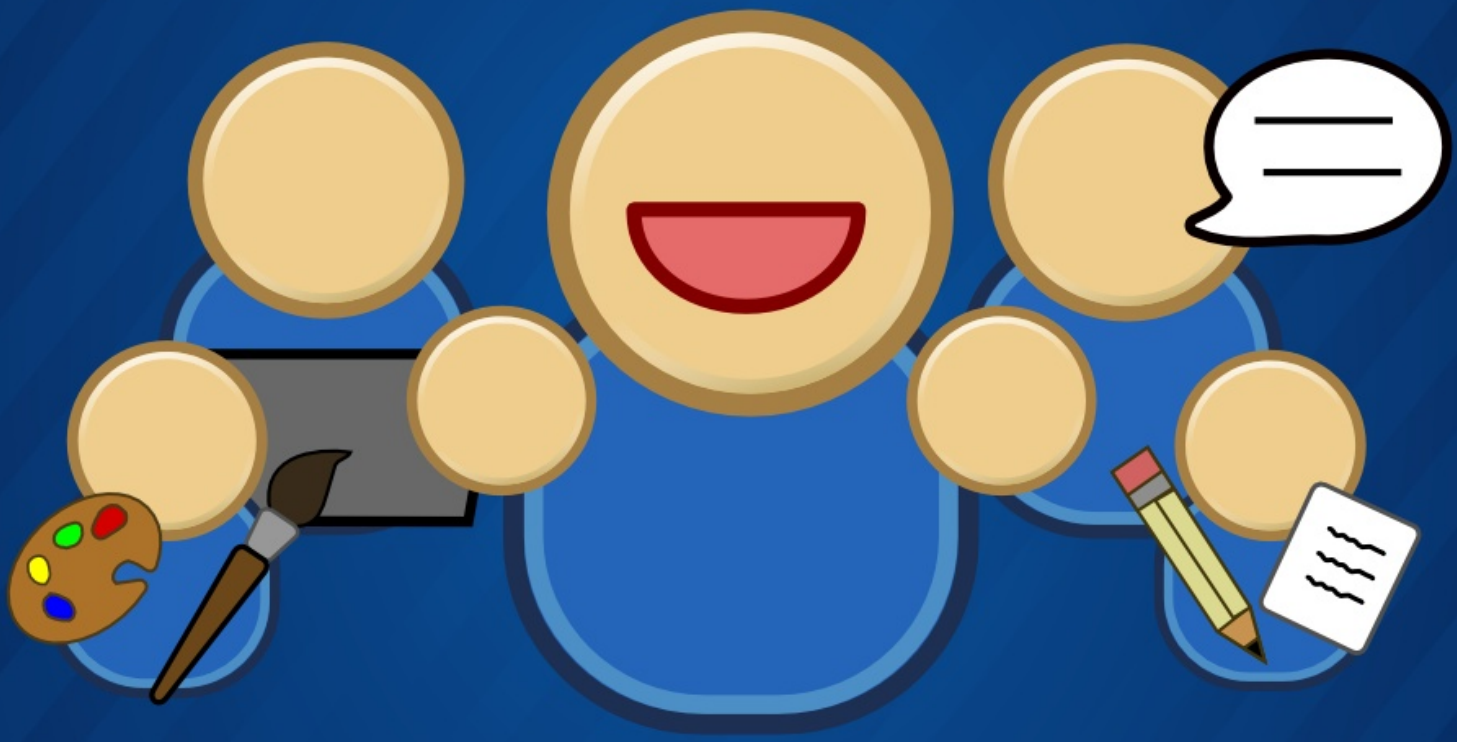
El escenario se complica si tenemos muchos nodos que quieren escribir de forma simultánea, ya que no podremos utilizar ext3 o ext4, sino que tendremos que formatear el dispositivo en GFS y para ello necesitamos configurar un ambiente clusterizado (escenario que tampoco es dificil de armar). De todas formas es una solución fácil de implementar y que da muy buenos resultados a la hora de armar soluciones baratas y sin tener la necesidad de comprar hardware caro.

Saludos.

Antonio Sebastián Sallés M.
Fedora Ambassador, Chile
Technical Account Manager
para Red Hat Latam
asalles@redhat.com



¿Quienes son fedora?



Ven y conocenos en proyectofedora.org

LVM: Administrando Volumenes Logicos

Henry Anchante
http://fedoraproject.org/wiki/User:Hacataka

¿Cuando nos funciona LVM para nuestro Linux?

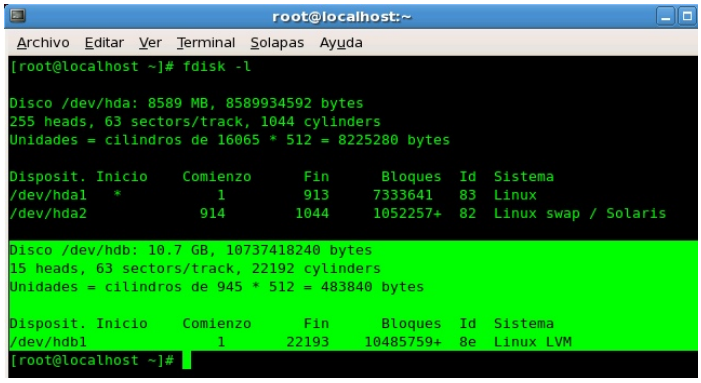
Implementación de LVM cuando el sistema ya este funcionando, agregando o aumento mas unidades Lógicas, al sistemas sin necesidad de modifica toda la estructura.

Ejemplo: tenemos un servidor File Server con 160 GB ya copados la unidad /home con 60 user y con espacio en disco de 2 GB, nuestro servidor esta en producción y solo podemos parar una noche cualquiera para poder aumentar el la unidad /home.

Agregaremos un disco duro de 160 GB, y aumentaremos la unidad /home

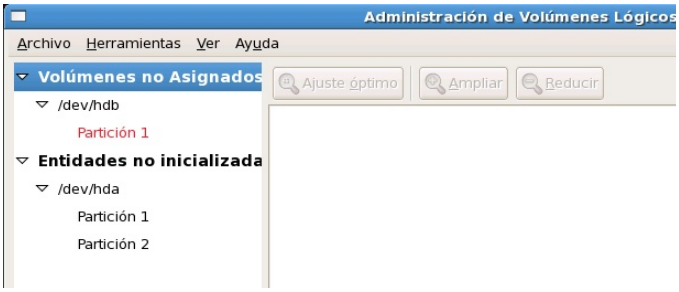
1.- con la siguiente línea de comando vamos a visualizar la unidad física agregada.

```
#fdisk -l
```



Las líneas sombreadas de color verde indica, tamaño 10.7 GB, nombre de la unidad, dispositivo iniciado en LVM.

Unas vez iniciadas abrimos el LVM que viene por defecto en nuestro Linux Redhat, Sistemas / Administracion / Administracion de volúmenes lógicos.



Luego asignaremos un nombre y espacio al nuevo volumen lógico, nombre del grupo del volumen puede ser

CARACTERISTICAS DE LVM

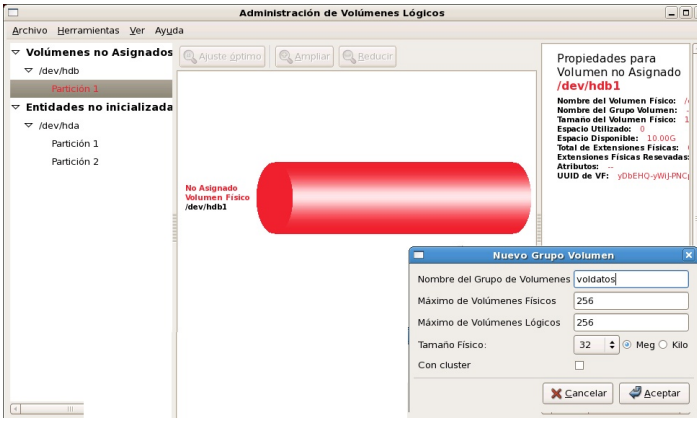
* LVM, es una implementación de un administrador de volúmenes lógicos para el Kernel de Linux.

* Proporciona una vista de alto nivel sobre el almacenamiento, en vez de la tradicional vista de discos y particiones.

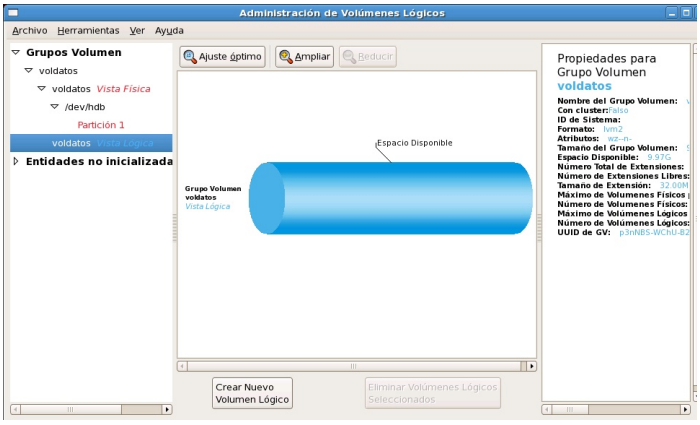
* Los volúmenes son llamados, por ejemplo, “ventas” o “desarrollo”, en vez de nombres de dispositivos físico, como “sda” o “sdb”.

* LVM no implementa RAID1 o RAID5.

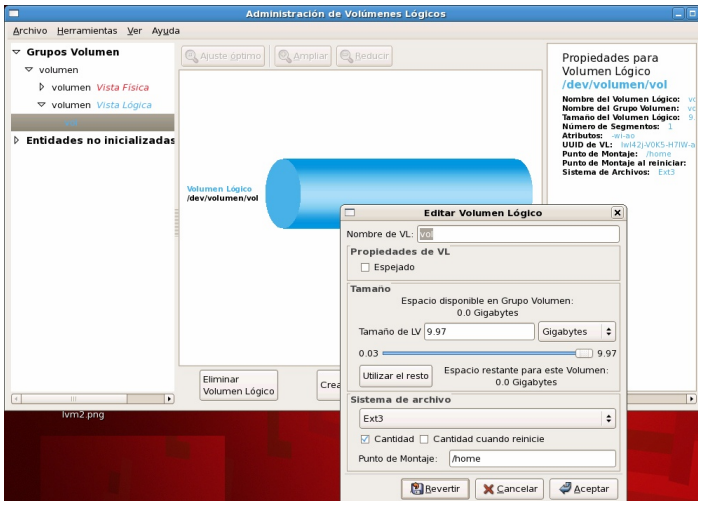
cualquiera, las cantidades deben ser como indica la imagen.



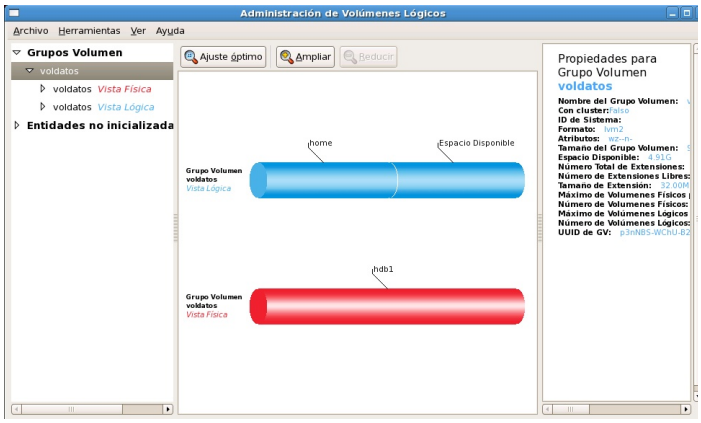
En la siguiente imagen veremos nuestra imagen lógica ya creado



Luego crearemos un volumen lógico dentro del volumen lógico.



En esta imagen indicaremos, que unidad física o lógica ya creada vamos aumentar la cantidad en GB, puedes indicar con la barra desde 1 GB hasta la capacidad total del volumen agregado.



Observamos en la imagen la unidad física en rojo, agregada al volumen (voldatos), en la imagen celeste podemos visualizar la unidad lógica (voldatos) y aun nos queda espacio disponible por asignar.


Fedora 15
Codename: LoveLock

Programando en Fedora con NetBeans

Joel Porras

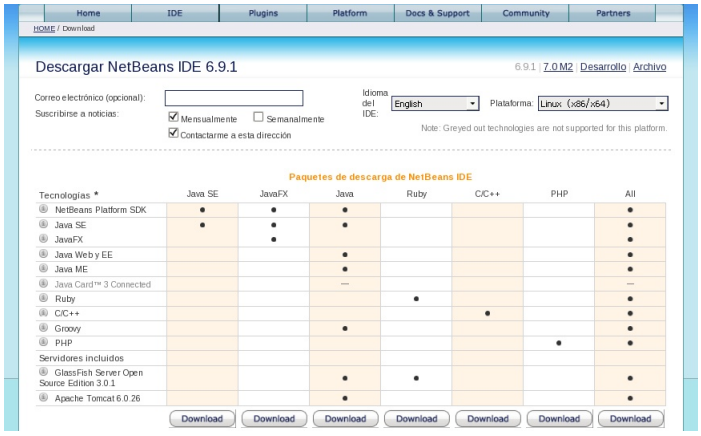
Desde su version Alpha esta distro ya ha venido con bastante buena estabilidad y a estado siendo muy robusto, Fedora no solo esta orientado a los SysAdmin sino tambien tiene espacio para la gente que se dedica al desarrollo.

En esta oportunidad instalaremos NetBeans

NetBeans es un editor muy estable y compatible con varios lenguajes de Programación, personalmente la he usado para programar en Java y Php con resultados bastante buenos.

1. Para obtener el programa nos vamos a la pagina web / descargamos la versión para Linux

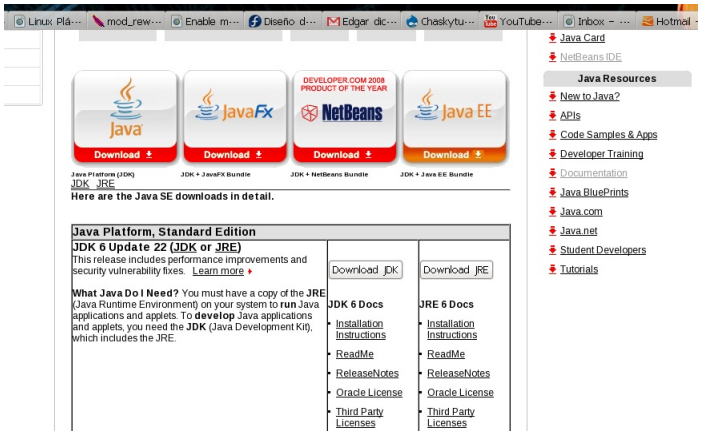
http://netbeans.org/downloads/index.html



No debemos olvidar que si queremos practicar Java debemos tener instalado JDK (Java Developement Kit) que lo vamos a encontrar en la siguiente url.

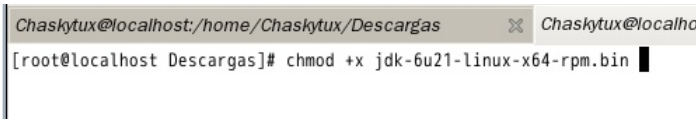
http://www.oracle.com/technetwork/java/javase/downloads/index.html

http://cds.sun.com/is-bin/INTERSHOP.enfinity/WFS/CDS-CDS_Developer-Site/en_US/-/USD/VerifyItem-Start/jdk-6u22-linux-x64-rpm.bin?BundledLineItemUUID=mIeJ_hCwj2IAAAEsL9EAHFIR&OrderID=YkKJ_hCwMD4AAAEsFNEAHFIR&ProductID=w9aJ_hCw9PAAAAErNFJuIQy3&FileName=/jdk-6u22-linux-x64-rpm.bin



2. Una vez descargado Jdk y NetBeans ingresamos como root a la carpeta donde los tenemos y le damos permisos de ejecución

chmod +x jdk-6u21-linux-x64-rpm.bin



chmod +x netbeans-6.9.1-ml-linux.sh

```
Chaskytux@localhost:/home/Chaskytux/Descargas

[root@localhost Descargas]# chmod +x jdk-6u21-linux-x64-rpm.bin
[root@localhost Descargas]# chmod +x netbeans-6.9.1-ml-linux.sh
[root@localhost Descargas]#


[root@localhost Descargas]#
```

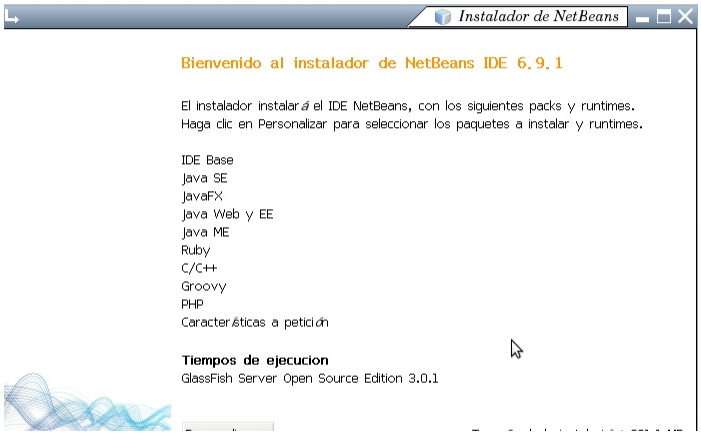
3. Primero instalamos jdk:
./jdk-6u21-linux-x64-rpm.bin

```
Chaskytux@localhost:/home/Chaskytux/Descargas
[root@localhost Descargas]# chmod +x jdk-6u21-linux-x64-rpm.bin
[root@localhost Descargas]# chmod +x netbeans-6.9.1-ml-linux.sh
[root@localhost Descargas]# ./jdk-6u21-linux-x64-rpm.bin
Unpacking...
Checksumming...
Extracting...
UnZipSFX 5.50 of 17 February 2002, by Info-ZIP (Zip-Bugs@lists.wku.edu).
replace jdk-6u21-linux-amd64.rpm? [y]es, [n]o, [A]ll, [N]one, [r]ename:
```

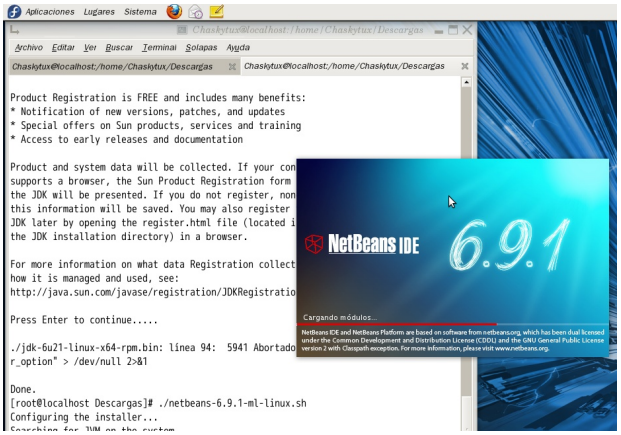
4. Una vez instalado Jdk instalaremos NetBeans:
./ netbeans-6.9.1-ml-linux.sh

```
Done.
[root@localhost Descargas]# ./netbeans-6.9.1-ml-linux.sh
Configuring the installer...
Searching for JVM on the system...
Extracting installation data...
Running the installer wizard...
```

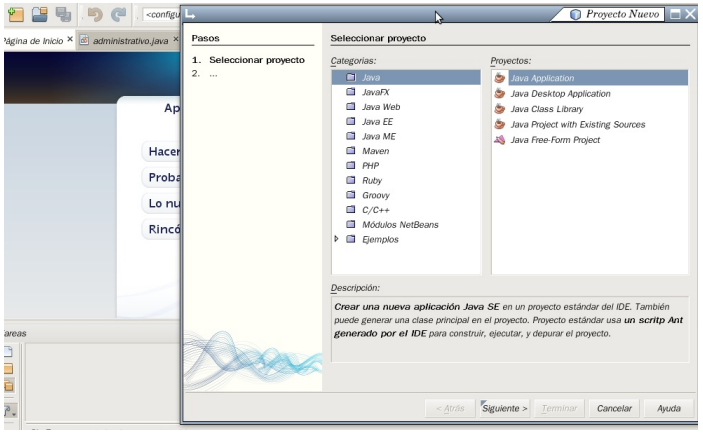
5. Se inicia la instalación



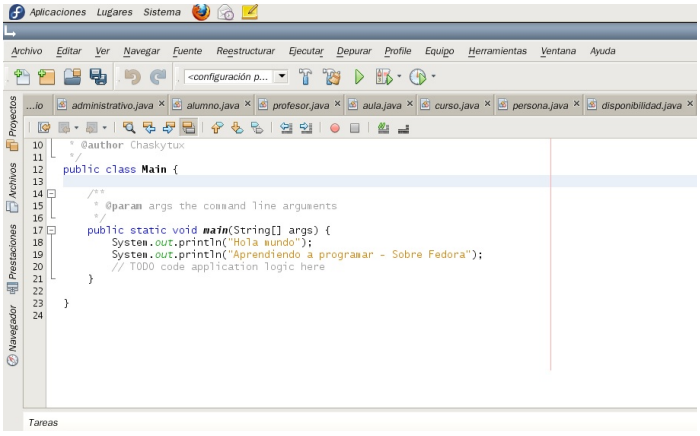
6. Una vez finalizada la instalación nos vamos Aplicaciones / Programación / NetBeans



7. Como lo había comentado al principio aqui podemos observar los lenguajes que se pueden utilizar en este editor.



En mi caso escogí Java
Aquí va mi primer “Hola mundo”




fedora

SCREEN: Un multiplexor de terminales

Francisco Diaz
http://fedoraproject.org/wiki/User:fcodiaz

Screen se puede decir que es un multiplexor de terminales, mediante su uso, se puede ejecutar cualquier número de aplicaciones interactivas como shells, sesiones a host remotos, compilaciones, aplicaciones basadas en ncurses, etc. y lo mejor de todo, desde una sola terminal. Esta capacidad es la que está atrayendo a más y más personas a usar screen como una herramienta necesaria en la administración/monitoreo de sistemas.

Otras de las características importantes de screen es su habilidad de desconectarse de las diferentes sesiones, lo cual permite en casos extremos que cualquier aplicación, aún ejecutandose en alguna de ellas, continue funcionando en caso que accidentalmente cerremos la ventana de la terminal donde estamos trabajando. de esta forma, podemos regresar y conectarnos a todas las ventanas (programas, conecciones remotas, compilación, etc) que hayamos tenido activas, si es que estas no han concluido todavía, seguirán ejecutandose normalmente.

Para instalar el programa, simplemente hagamos una búsqueda del paquete screen en nuestro programa/administrador de paquetes favorito y acorde a la distribución, para el caso de fedora tenemos:

```
[fdiaz@fcoaaone ~]$ yum search screen
========== Matched: screen ==========
..../// parte de la salida ha sido eliminada con propositos de legibilidad
screen.i686 : A screen manager that supports multiple logins on one terminal
....
[fdiaz@fcoaaone ~]$
```

Luego lo instalamos de la siguiente forma:

```
[fdiaz@fcoaaone ~]$ sudo yum install screen
[sudo] password for fdiaz:
Loaded plugins: presto, refresh-packagekit
Setting up Install Process
Resolving Dependencies
--> Running transaction check
---> Package screen.i686 0:4.0.3-15.fc12 set to be installed
--> Finished Dependency Resolution
Dependencies Resolved
..../// Parte de la salida ha sido truncada para para facilitar la lectura.
Transaction Test Succeeded
Running Transaction
  Installing     : screen-4.0.3-15.fc12.i686                                1/1
Installed:
  screen.i686 0:4.0.3-15.fc12
Complete!
```

> Screen tiene la capacidad de desconectarse de las ventanas activas y las aplicaciones o conecciones existentes en ellas persistirán.

Finalemente, iniciamos screen desde una terminal, ejecutando el siguiente comando:

```
[fdiaz@fcoaaone ~]$ screen
```

Posiblemente un mensaje de ingreso a screen sea presentado, el título de la terminal también pueda que cambie para identificar que screen está activa; cada programa ejecutándose bajo screen se asocia a una ventana diferente y cada ventana es identificada por un número mayor o igual que cero, ejecutemos algún comando en esta ventana para poder identificarla después cuando nos estemos moviendo entre las diferentes ventanas.

Creemos otra ventana, la número 1, para ello presionamos la combinación en el teclado ctrl+a-c para explicar un poco, esto significa que presionamos al mismo tiempo la tecla ctrl y la tecla a (minúscula) y luego presionamos la tecla c (del inglés create).

Ahora tenemos dos ventanas, el contenido de la ventana anterior desapareció, puesto que ahora estamos en la ventana #1, para regresar a la ventana anterior tenemos las siguientes opciones:

- Podemos usar la combinación ctrl+a-p (del inglés previous)
- la combinación ctrl+a-n (next) hasta regresar a la ventana 0
- ctrl+a-ctrl+a nos regresa a la ventana donde nos encontrabamos anteriormente
- ctrl+a-# donde # es el número de la ventana a donde queremos movernos
- ctrl+a-" (comillas dobles) esta combinacion nos mostrará un listado de las ventanas existentes, en la cual podremos movernos haciendo uso de las teclas de flechas y seleccionando mediante la tecla Enter la ventana que deseamos utilizar.

Las ventanas que tenemos activas las podemos re-nombrar para identificar que acción/proceso se desarrolla en cada una de ellas, para eso podemos hacer uso de la combinación de teclas ctrl+a-A (en este caso es A mayúscula), una línea con el nombre actual de la ventana aparecerá en la parte inferior de la terminal, donde podremos especificar nuestro propio nombre para esa ventana.

En el caso que alguna de las sesiones se quede congelada o la aplicación deja de responder, dicha sesión puede cerrarse mediante la combinación ctrl+a-K

Como mencionamos anteriormente screen tiene la capacidad de desconectarse de las ventanas activas y las aplicaciones o conecciones existentes en ellas persistirán; para desconectarse de screen usamos la combinacion ctrl+a-d. La desconección se puede realizar de igual forma cerrando la terminal donde estamos ejecutando screen. Ninguna de las aplicaciones en las diferentes ventanas debería verse afectada por esta desconección, para confirmar esto,

abramos otra terminal, si es que cerramos la terminal que estabamos usando, ahora ejecutamos el comando: screen -r, veremos como las ventanas se restablecen, si ejecutamos la combinacion ctrl+a-" confirmamos que todas se encuentran activas y los programas en ellas aún se encuentran funcionando, como si nunca nos hubiesemos desconectado.

Para finalizar, screen no es fácil de eliminar o cerrar, como ya se habrán dado cuenta, para salir de screen hay que cerrar cada una de las ventanas que se tengan abiertas, ya sea cerrando el shell en cada una de ellas o la aplicación que se esté ejecutando.

Screen es una herramienta muy útil para mantener el control de diferentes aplicaciones o conecciones a sistemas remotos, monitoreo, etc.; la aplicación tiene más opciones, así que les dejo la curiosidad para que ustedes mismos las descubran, usando la combinación ctrl+a-? screen nos mostrará todas las opciones y combinaciones disponibles; por último no olvidemos que el manual siempre está a la mano mediante el comando: man screen.

Enlaces:
http://www.gnu.org/software/screen/
http://en.wikipedia.org/wiki/GNU_Screen

Francisco Diaz

fudcon 2011 PANAMA

FEDORA USERS & DEVELOPERS CONFERENCE

26 al 28 de Mayo, Ciudad del Saber

https://fedoraproject.org/wiki/FUDCon:Panama_2011

Patrocinan:

redhat

Ciudad del Saber
PANAMA
City of Knowledge

CIDETYS
Centro Internacional de Desarrollo Tecnológico y Software Libre