

Universidad de San Carlos de Guatemala  
Facultad de Ingeniería  
Escuela de Ciencias y Sistemas  
Sistemas Operativos 1  
Laboratorio

**Proyecto Fase 1**  
Virtualización, Kernel y Módulos.

Luis Aroldo Morales Noriega  
201020944  
08/10/2016

# Manual de Configuraciones

## Sistema Operativo:

El sistema operativo donde se instalara KVM será Ubuntu donde se escogió instalar el sistema Debían con el cual se trabajara el proyecto.

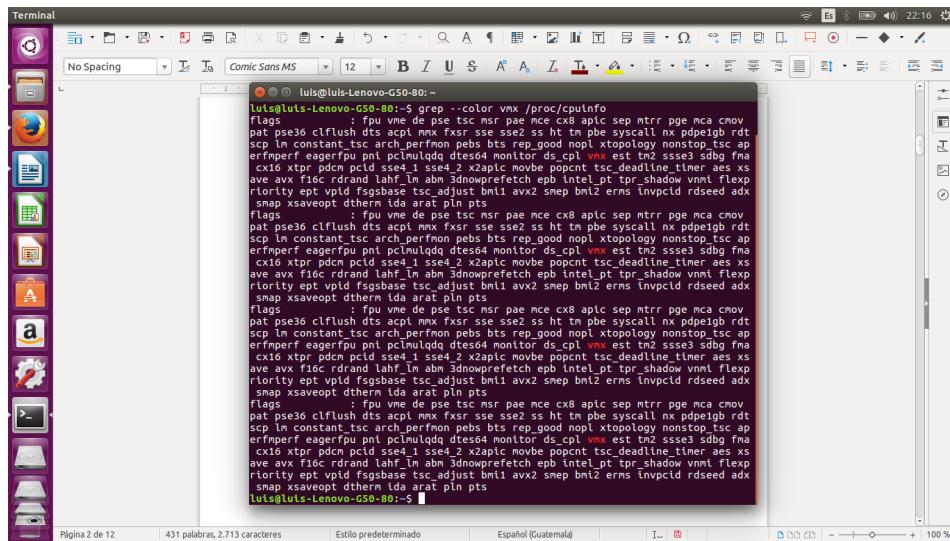
## Instalación de KVM en Ubuntu:

Es importante conocer si nuestra computadora tiene soporte para KVM, para ello debemos buscar entre las banderas del procesador, en el archivo /proc/cpuinfo, las siguientes palabras: vmx para procesador Intel ó svm para procesadores AMD. Puede utilizar el siguiente comando para verificarlo, para el caso de procesadores Intel:

---

```
grep --color vmx /proc/cpuinfo
```

---



```
luis@luis-Lenovo-G50-80:~ luis@luis-Lenovo-G50-80:~$ grep --color vmx /proc/cpuinfo
flags           : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov
pat pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe syscall nx pdpe1gb rdt
scp lm constant_tsc arch_perfmon pebs bts rep_good nopl xttopology nonstop_tsc ap
erfmprefr eagerfpn pni pclmulqdq dtes64 monitor ds_cpl vmx est tm2 sse3 sub8 rma
cx16 xtr pdcm pcid sse4_1 sse4_2 x2apic movebe popcnt tsc_deadline_timer aes xs
ave avx f16c rdrand lahf_lm abm 3dnowprefetch epb intel_pt tpr_shadow vnmi flexp
riority ept vpid fsgsbase tsc_adjust bm11 avx2 smp bm12 erms invpcid rdseed adx
snap xsaveopt dtherm ida arat pln pts
flags           : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov
pat pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe syscall nx pdpe1gb rdt
scp lm constant_tsc arch_perfmon pebs bts rep_good nopl xttopology nonstop_tsc ap
erfmprefr eagerfpn pni pclmulqdq dtes64 monitor ds_cpl vmx est tm2 sse3 sub8 rma
cx16 xtr pdcm pcid sse4_1 sse4_2 x2apic movebe popcnt tsc_deadline_timer aes xs
ave avx f16c rdrand lahf_lm abm 3dnowprefetch epb intel_pt tpr_shadow vnmi flexp
riority ept vpid fsgsbase tsc_adjust bm11 avx2 smp bm12 erms invpcid rdseed adx
snap xsaveopt dtherm ida arat pln pts
flags           : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov
pat pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe syscall nx pdpe1gb rdt
scp lm constant_tsc arch_perfmon pebs bts rep_good nopl xttopology nonstop_tsc ap
erfmprefr eagerfpn pni pclmulqdq dtes64 monitor ds_cpl vmx est tm2 sse3 sub8 rma
cx16 xtr pdcm pcid sse4_1 sse4_2 x2apic movebe popcnt tsc_deadline_timer aes xs
ave avx f16c rdrand lahf_lm abm 3dnowprefetch epb intel_pt tpr_shadow vnmi flexp
riority ept vpid fsgsbase tsc_adjust bm11 avx2 smp bm12 erms invpcid rdseed adx
snap xsaveopt dtherm ida arat pln pts
luis@luis-Lenovo-G50-80:~$
```

Y para el caso de procesadores AMD:

---

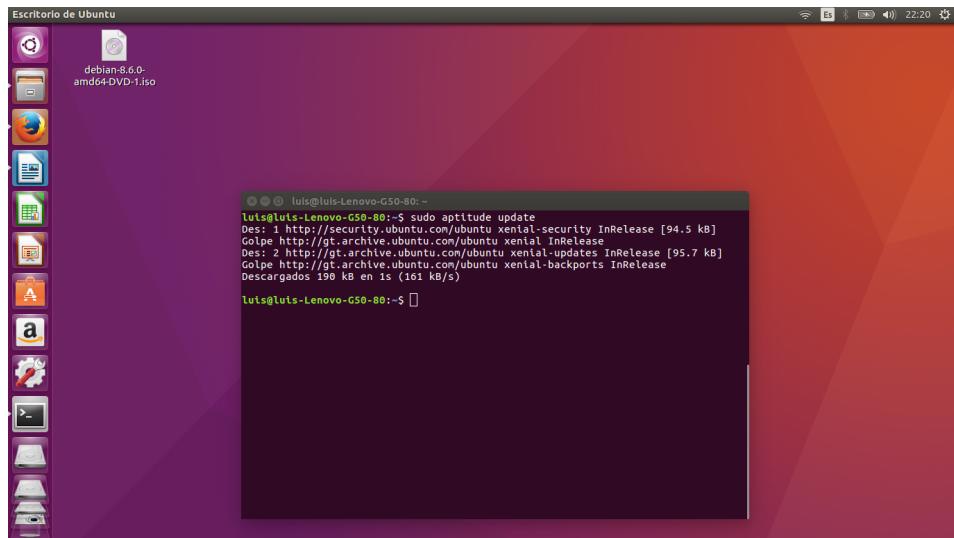
```
grep --color svm /proc/cpuinfo
```

---

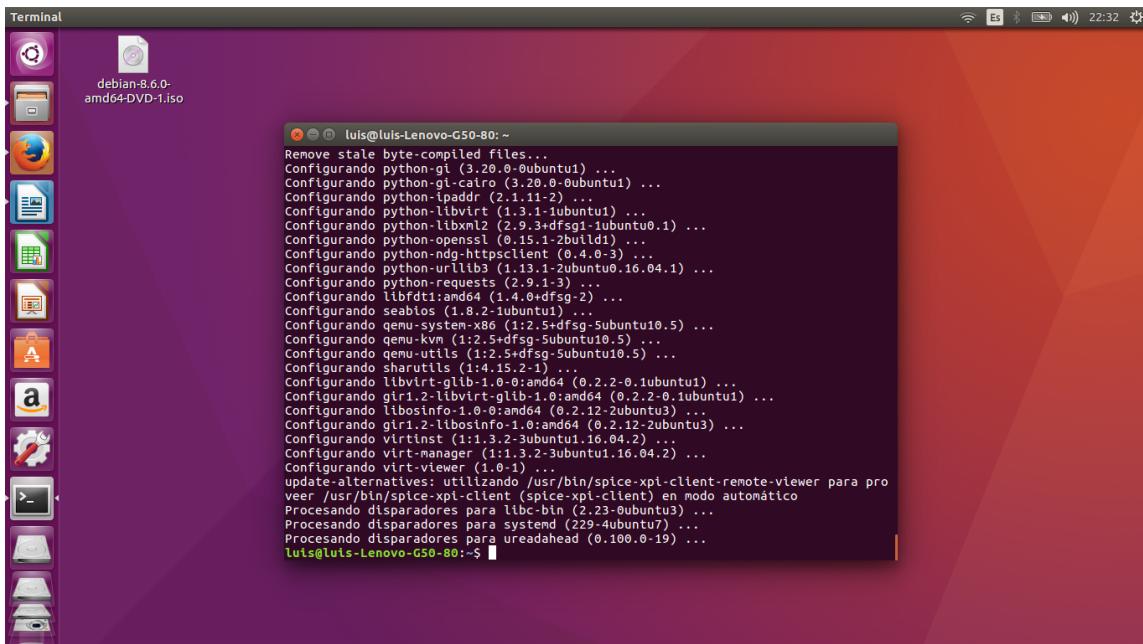
Debe aparecer en rojo la palabra que buscamos, si no fuera así, por favor verifique de nuevo haber escrito bien el comando o esto hace indicar que su procesador no tiene posibilidad de soportar KVM.

Para instalar KVM (Kernel Based Virtual Machine) escribimos los siguientes comandos en consola de Debían.

*sudo aptitude update*



```
sudo apt-get install qemu-kvm libvirt-bin bridge-utils virt-manager
```

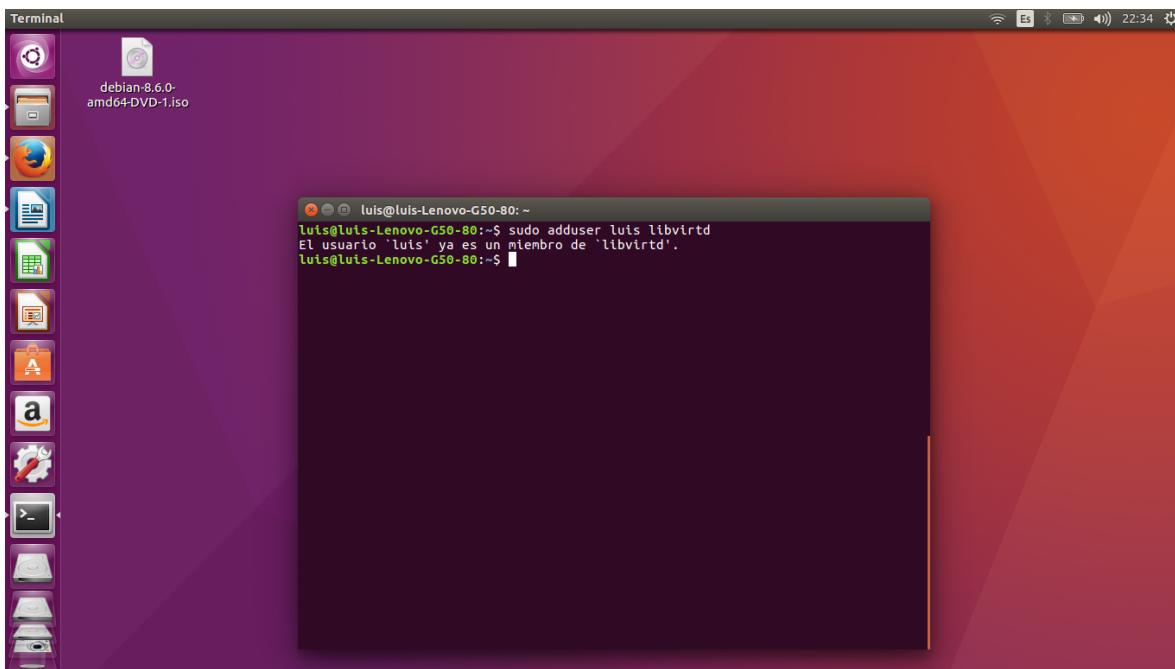


Para agregar al usuario que queremos que use la virtualización. En este caso usamos luis, pero pueden elegir cualquiera. Escribimos en la consola:

---

*sudo adduser luis libvirt*

---



Cargamos módulo por seguridad o evitar algún problema:  
Para AMD, escribimos en la consola:

---

*sudo modprobe kvm-amd*

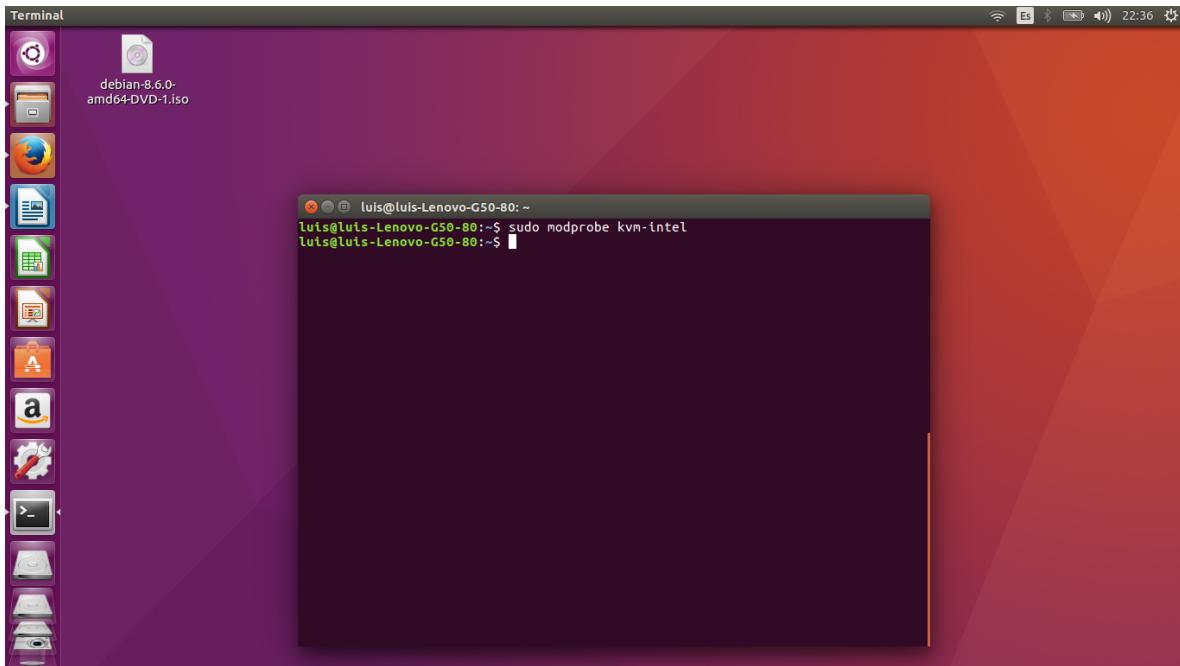
---

Para Intel, escribimos en la consola:

---

*sudo modprobe kvm-intel*

---



Preparar la configuración de red para que los futuros guests tengan conectividad entre el host y los guest.

Entramos en la consola y escribimos:

---

*sudo nano /etc/network/interfaces*

---

Les podría salir algo similar a lo siguiente:

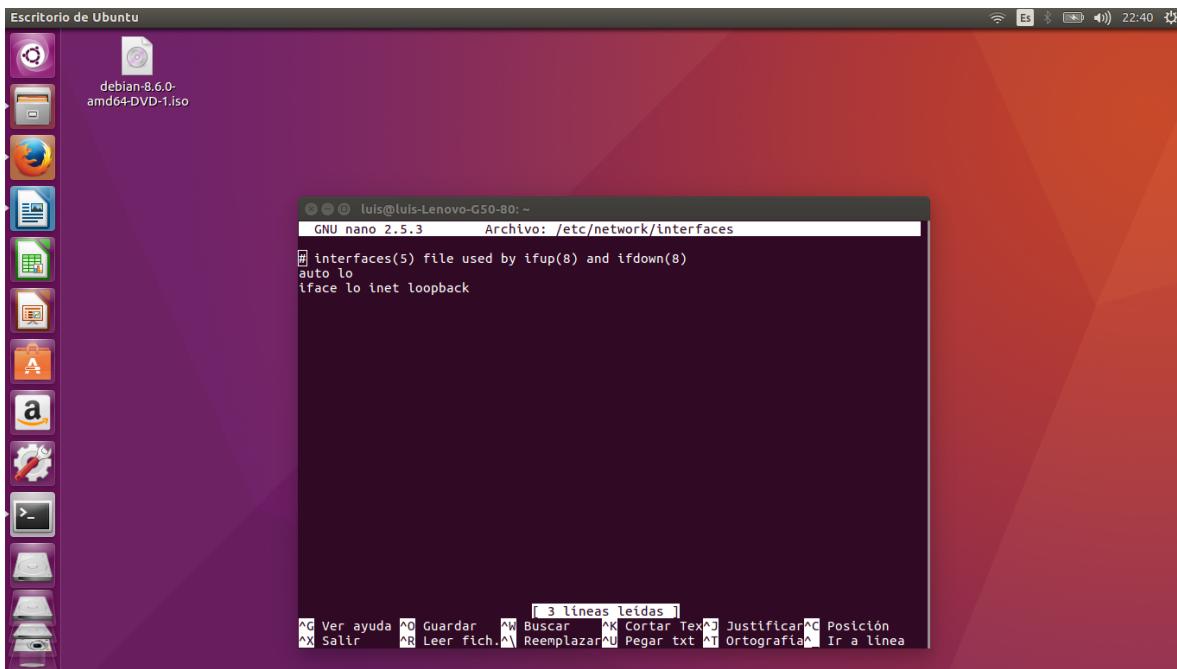
---

*auto eth0*

---

*iface eth0 inet dhcp*

---



Deberían de llegar a algo como lo siguiente:

---

*auto br0*

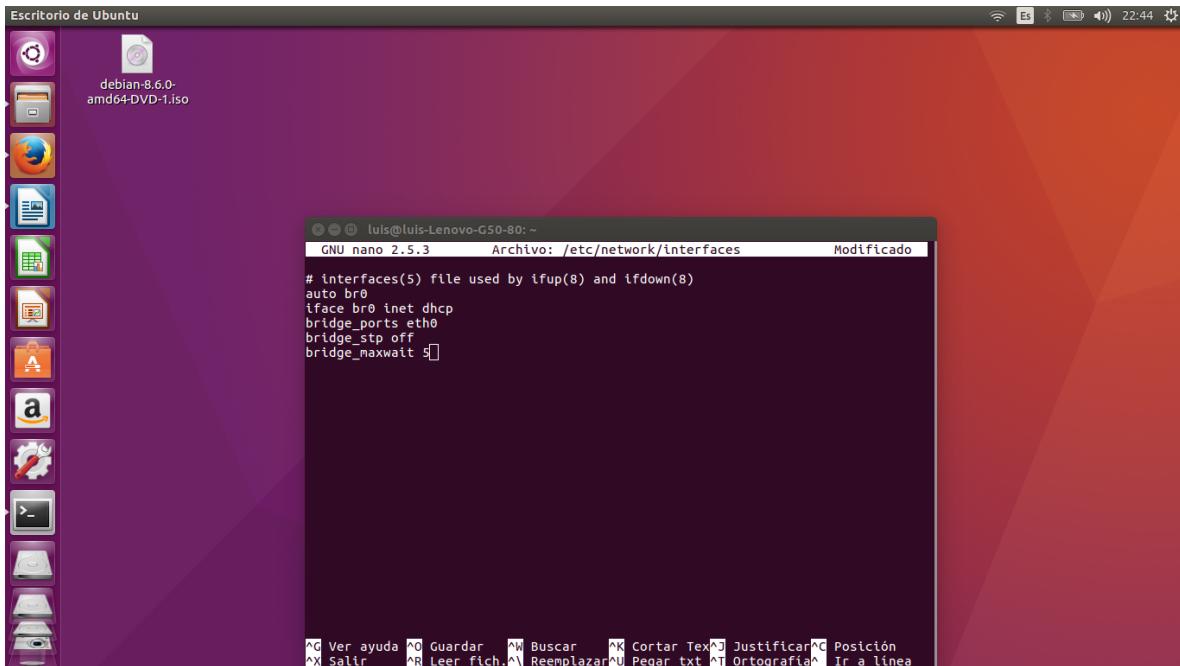
*iface br0 inet dhcp*

*bridge\_ports eth0*

*bridge\_stp off*

---

*bridge\_maxwait 5*

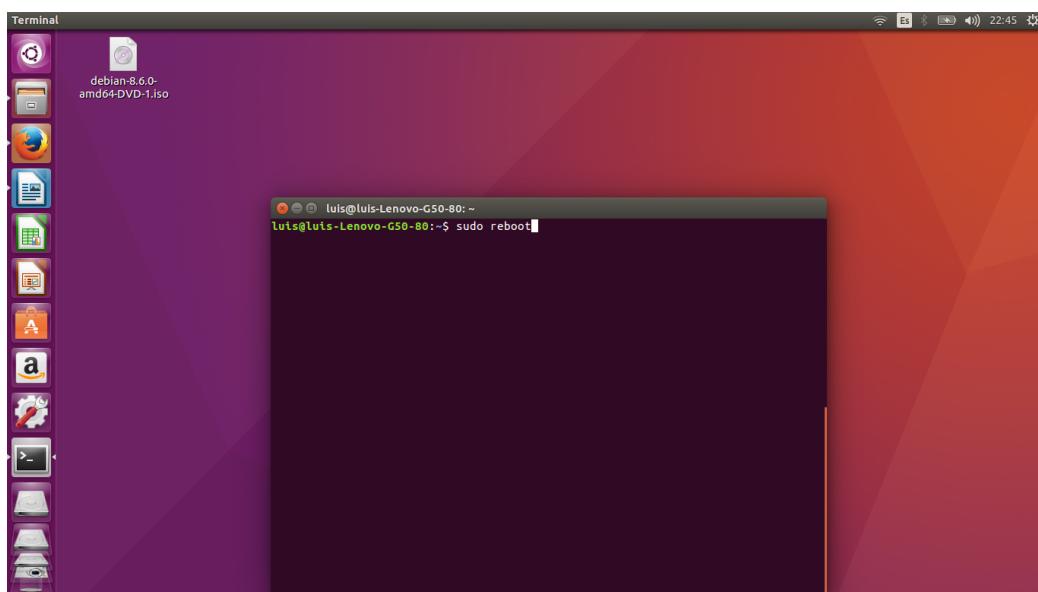


se guardan los cambios con control + o, antes de iniciar una máquina virtual se debe de reiniciar el sistema, escribimos en consola

---

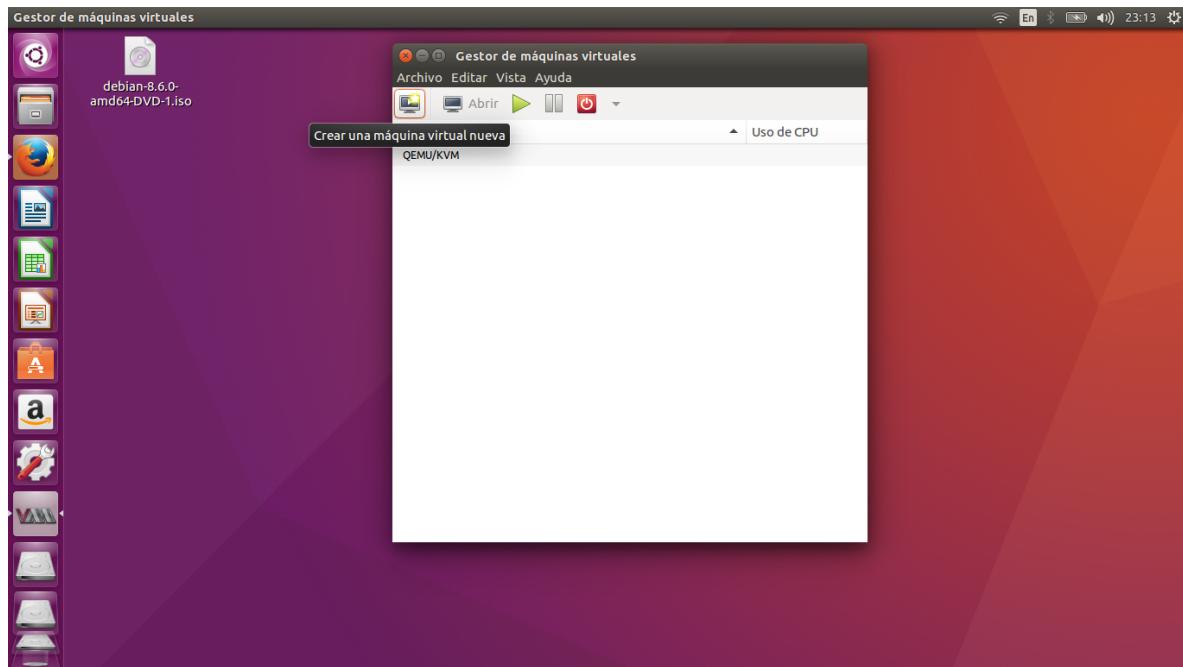
*sudo reboot*

---

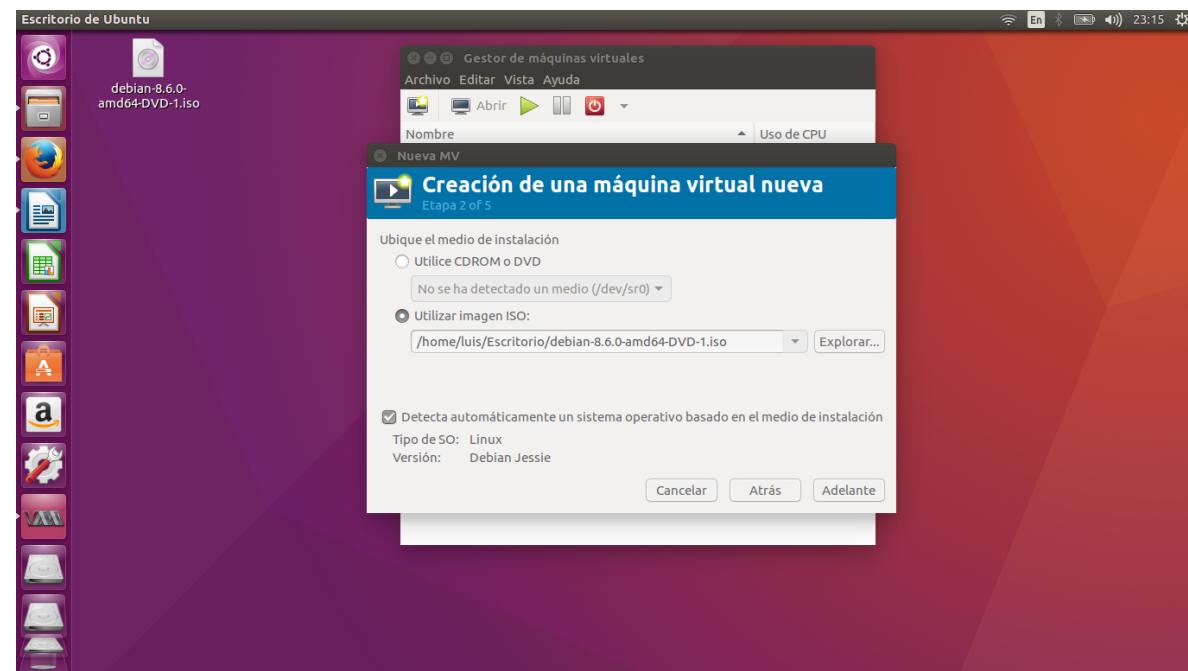


## Instalación de Debian 8 en KVM:

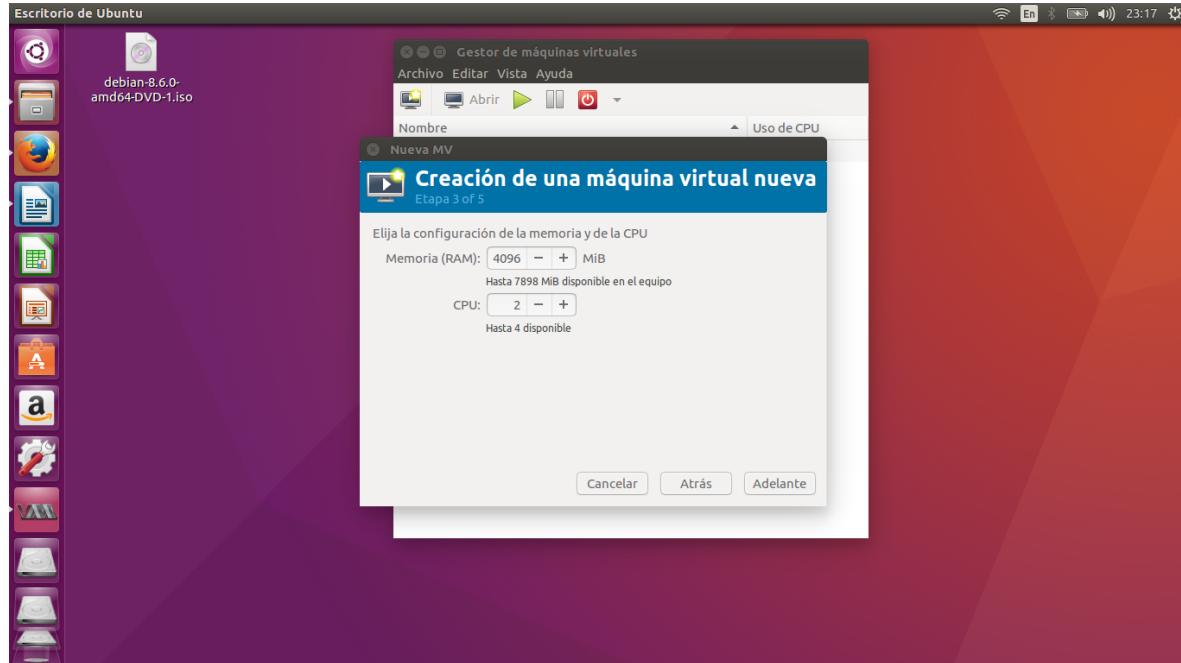
Ya que tenemos instalado el KVM vamos al gestor de máquinas virtuales y le damos al botón crear nueva máquina virtual.



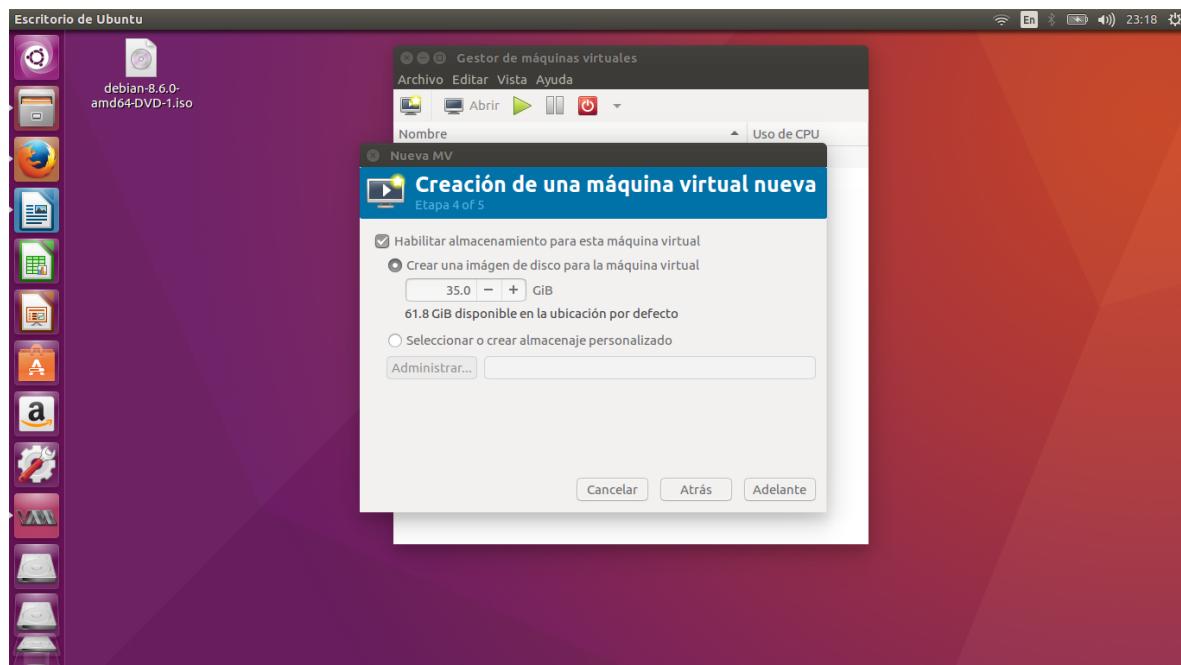
Creamos una nueva máquina virtual seleccionando un iso.



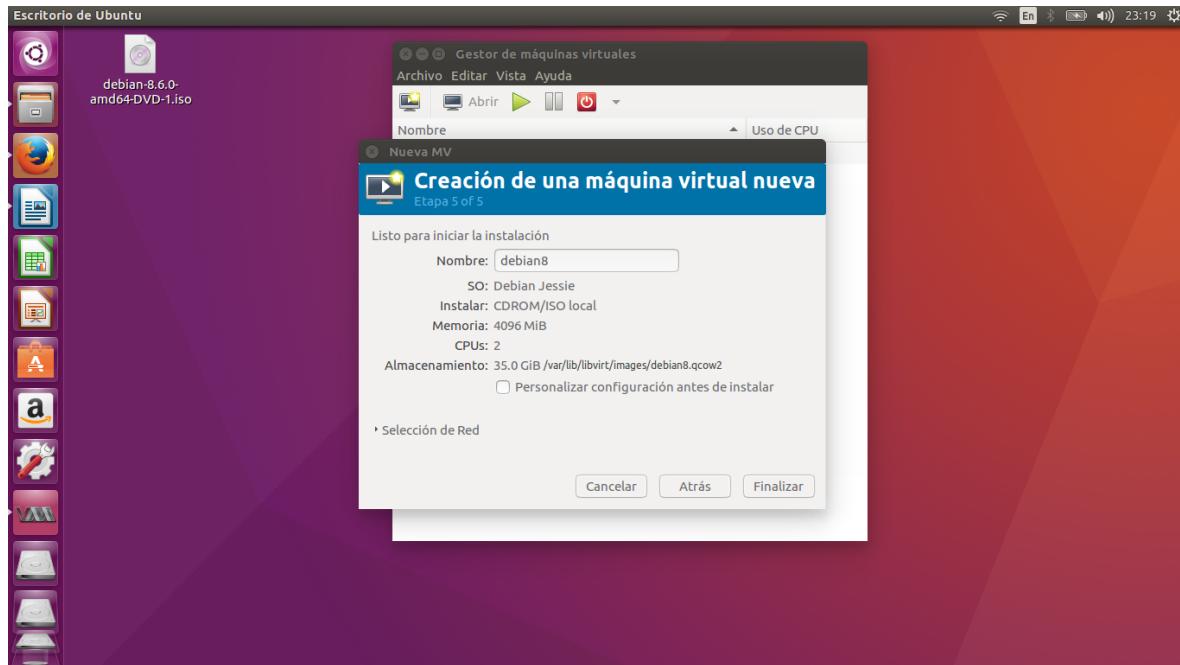
Seleccionamos capacidad de ram y cpu en este caso 2048Mb y 1 cpu



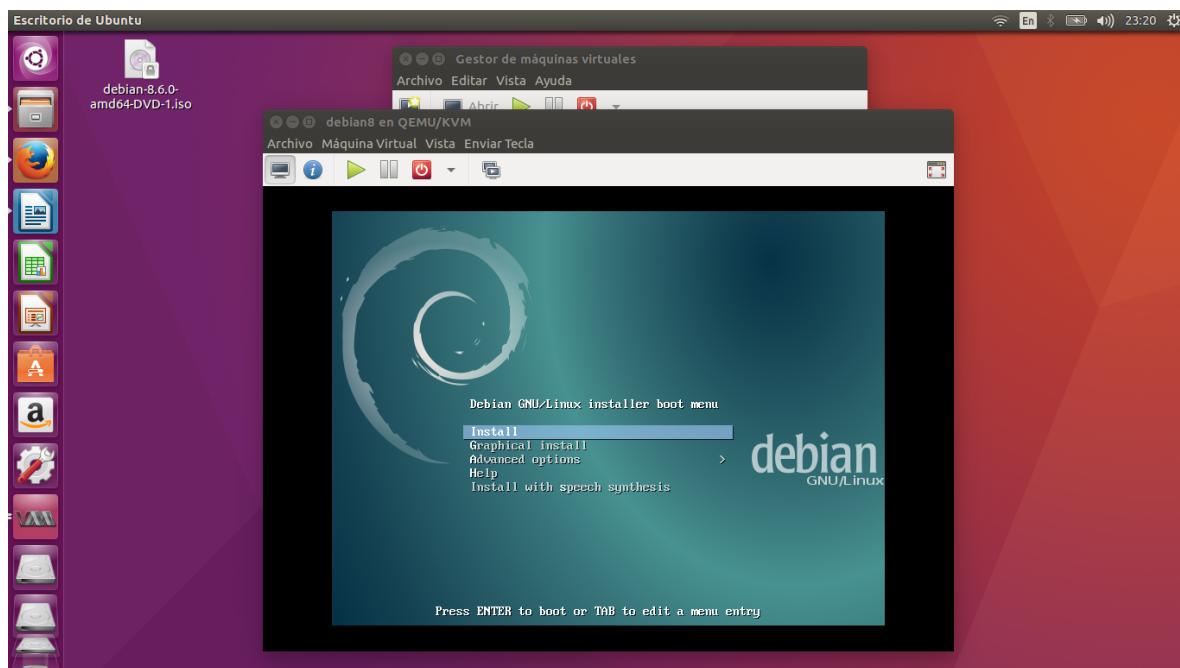
Seleccionamos capacidad de disco duro en este caso 35.0 GB



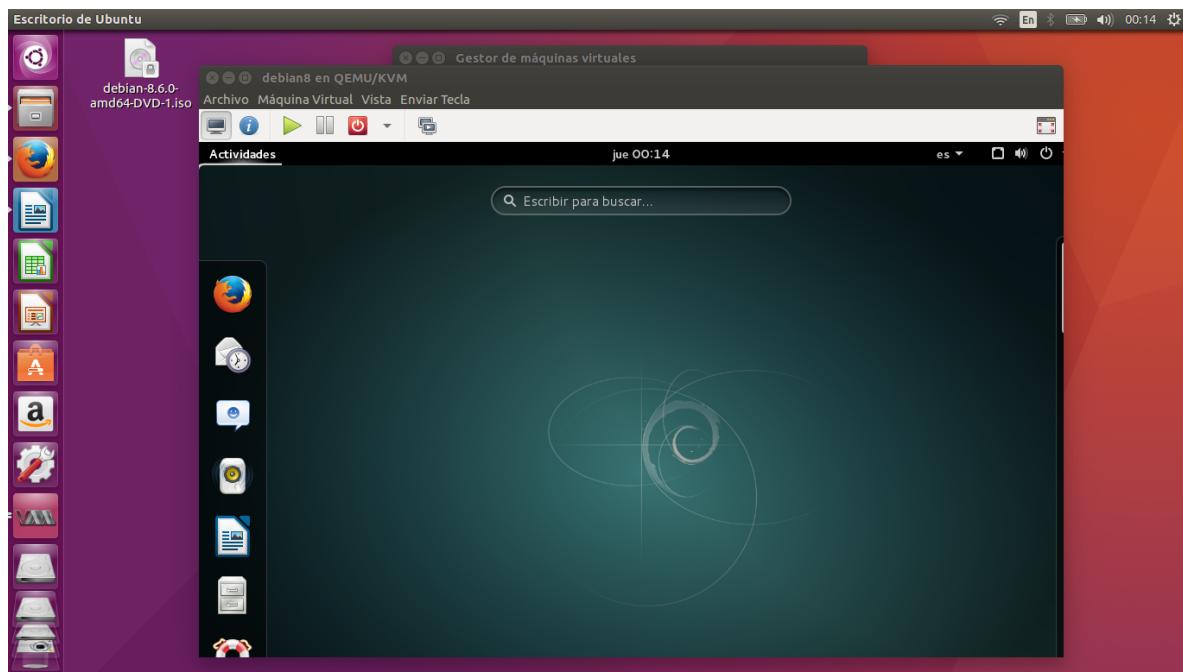
Por ultimo nos muestra el resumen de nuestra máquina e iniciamos la instalación de nuestro sistema operativo con el botón de finalizar.



Instalamos nuestro sistema operativo.

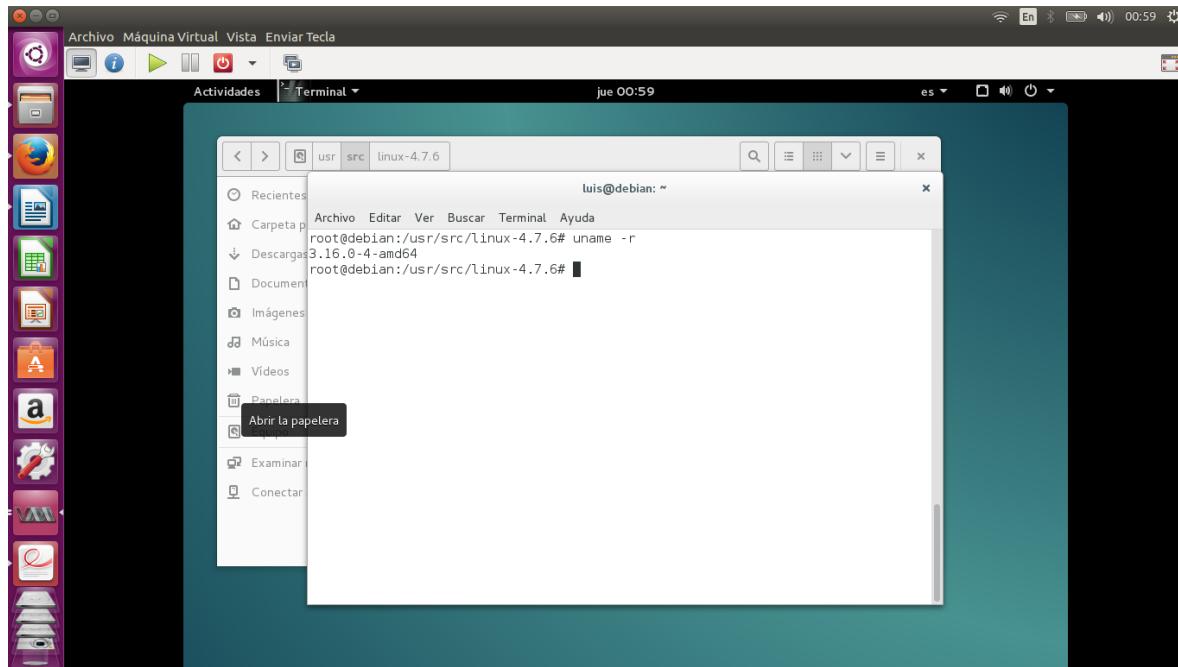


Al finalizar tendremos nuestra máquina virtual con el sistema operativo debían instalado y listo para la siguiente fase del proyecto.

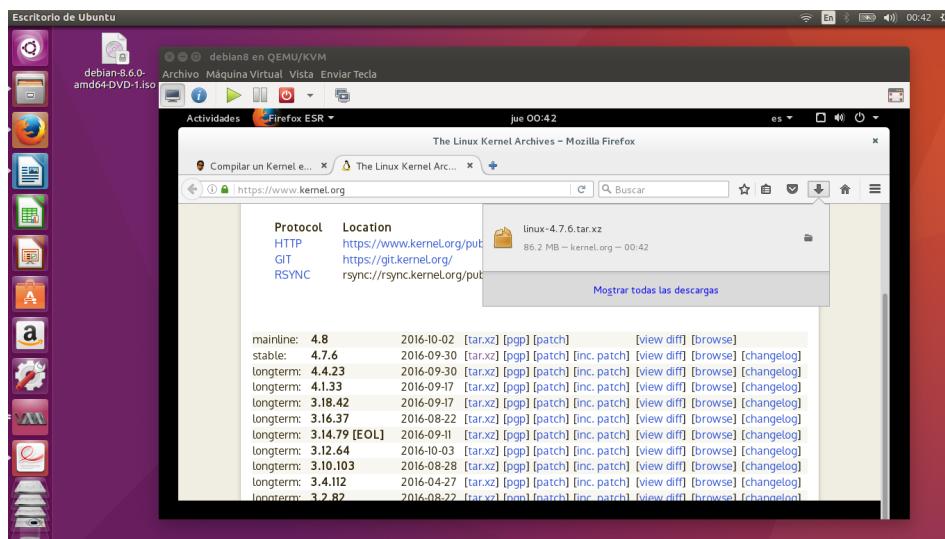


## Compilación de Kernel de Debian 8:

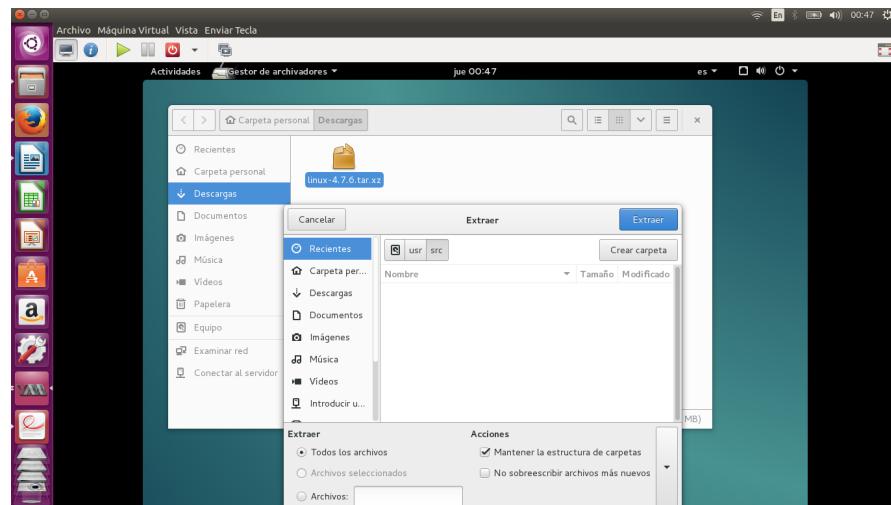
Nuestro kernel actual de Debian 8 es 3.16 se muestra con el comando “uname -r”



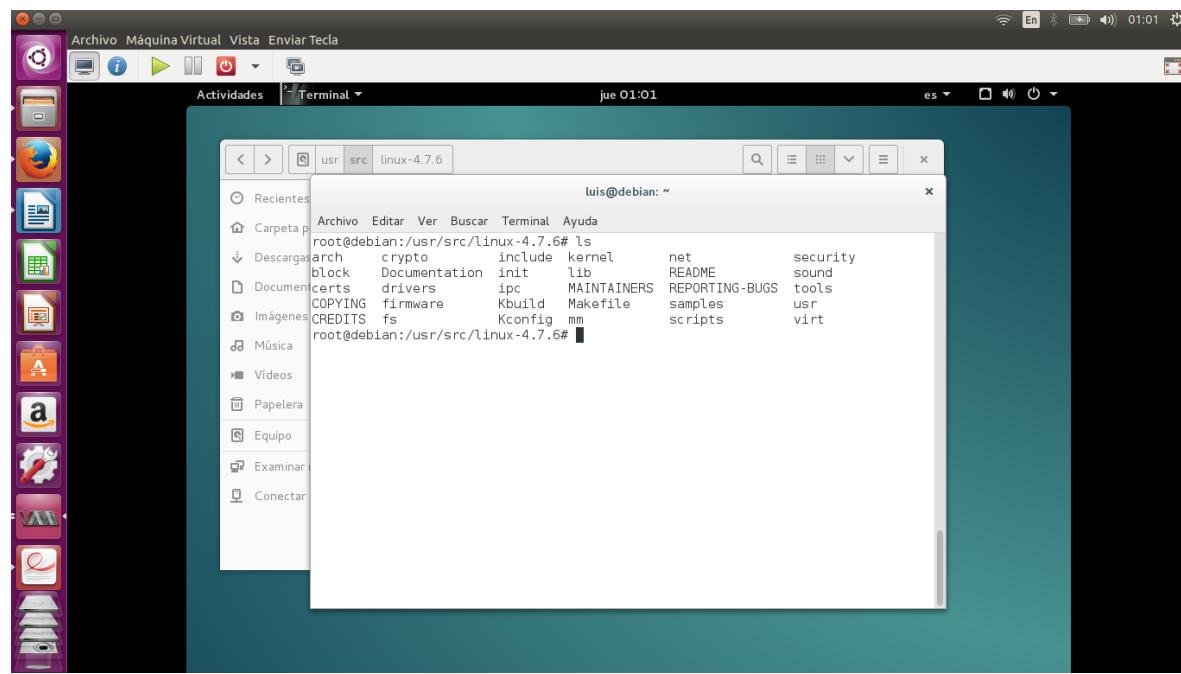
Lo primero que debemos hacer para compilar nuestro nuevo kernel es obtener la versión que deseamos instalar de la página oficial de linux <https://www.kernel.org/> donde en nuestro caso se escoge la versión 4.7.6 ya que es estable y tiene las configuraciones que necesitamos para nuestro proyecto.



Una vez descargado es recomendable descomprimirlo en el directorio /usr/src/ solo para tener control y orden sin embargo se puede poner donde se deseé.



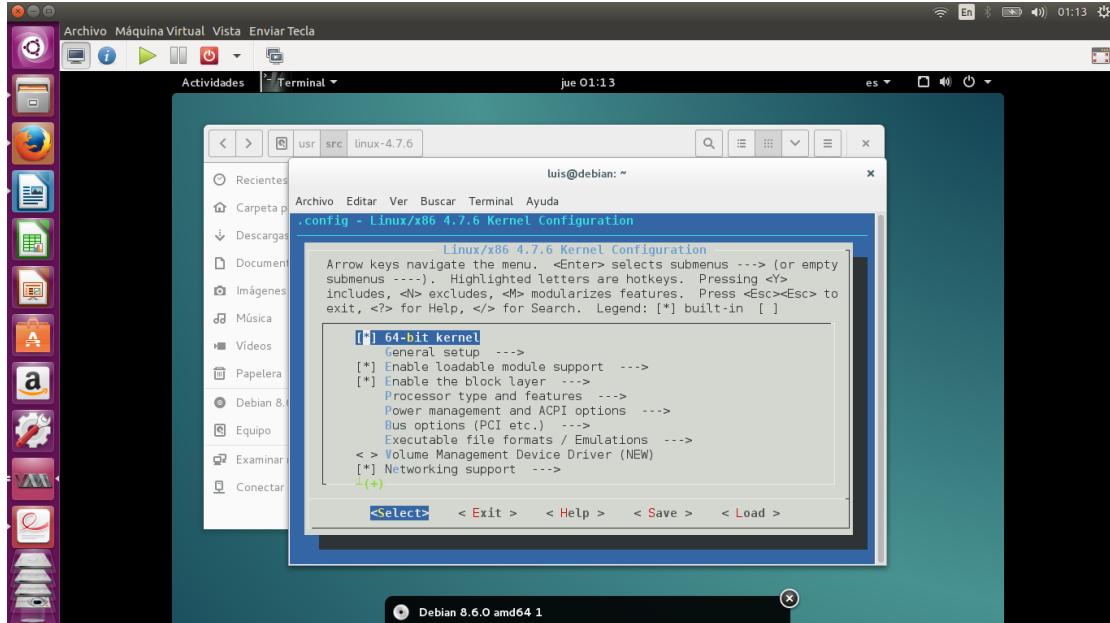
Ahora entramos a la terminal al directorio para poder configurar nuestro nuevo kernel



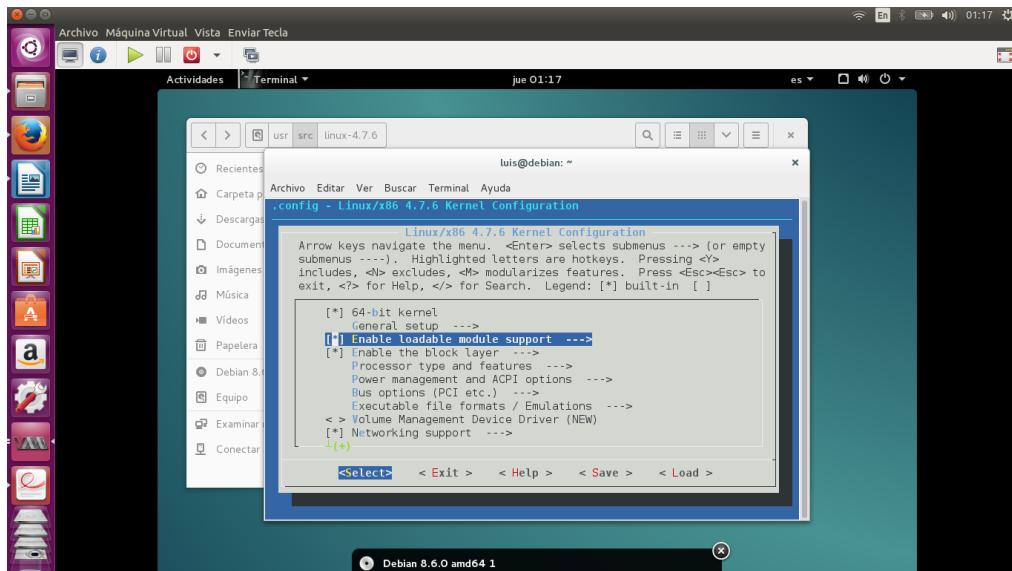
Para configurar nuestro kernel utilizaremos el comando

*make menuconfig*

Este comando nos mostrara el menú siguiente donde configuraremos todo lo necesario para nuestro nuevo kernel



Como podemos ver en la siguiente imagen en nuestra configuración tenemos la carga dinámica de módulos por lo tanto marcamos la opción si hubiera alguna otra podemos ir buscando con la ayuda de este menú.



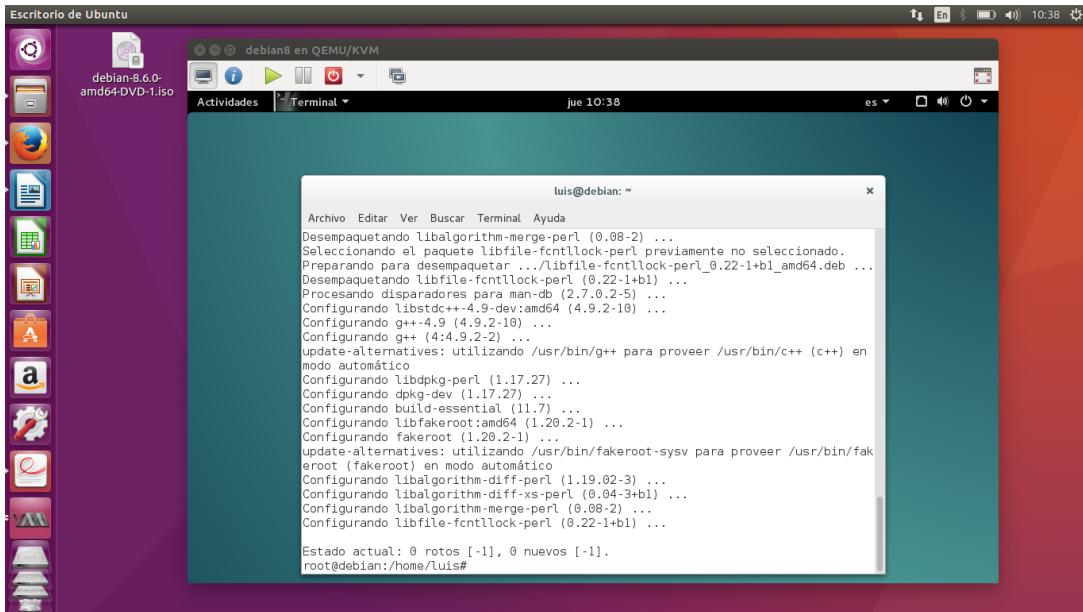
Una vez terminada la configuracion guardamos los cambios y vamos podemos empezar a compilar nuestro kernel para esto se necesita un paquete llamado kernel-package, el cual suministra las herramientas necesarias para poder compilar y generar el paquete con nuestro nuevo kernel.

Para instalarlo usaremos el comando

---

*aptitude install kernel-package*

---

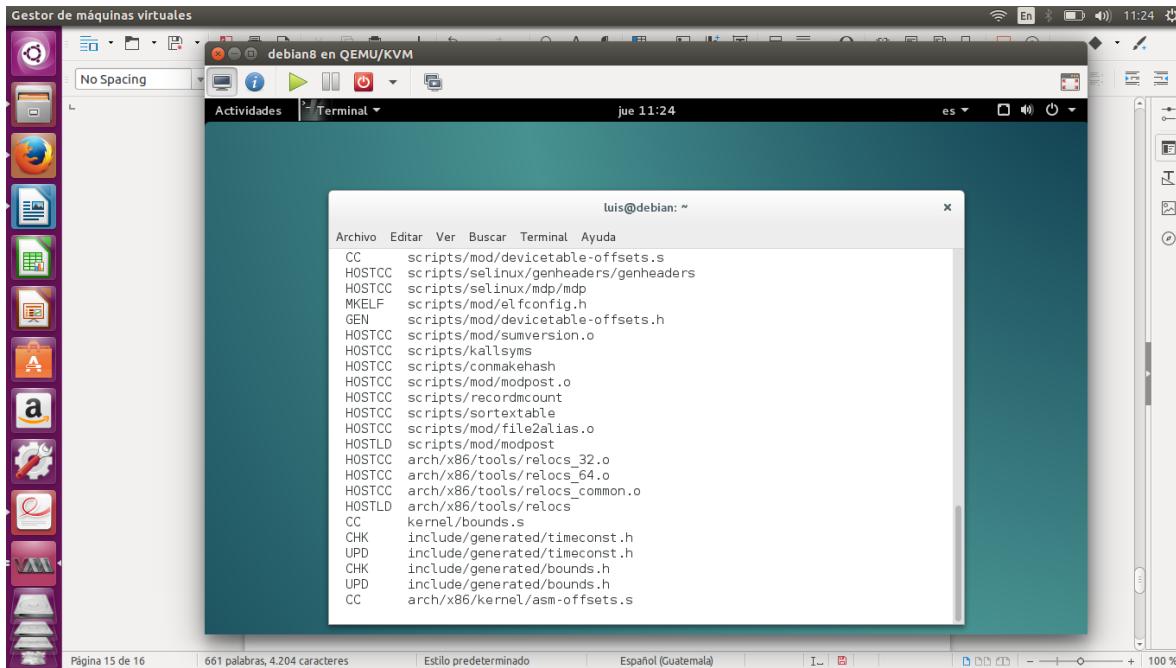


Una vez instalado el kernel-package pasamos a limpiar el sistema para poder compilar el kernel sin problemas con el siguiente comandos

---

*sudo make -j2*

---

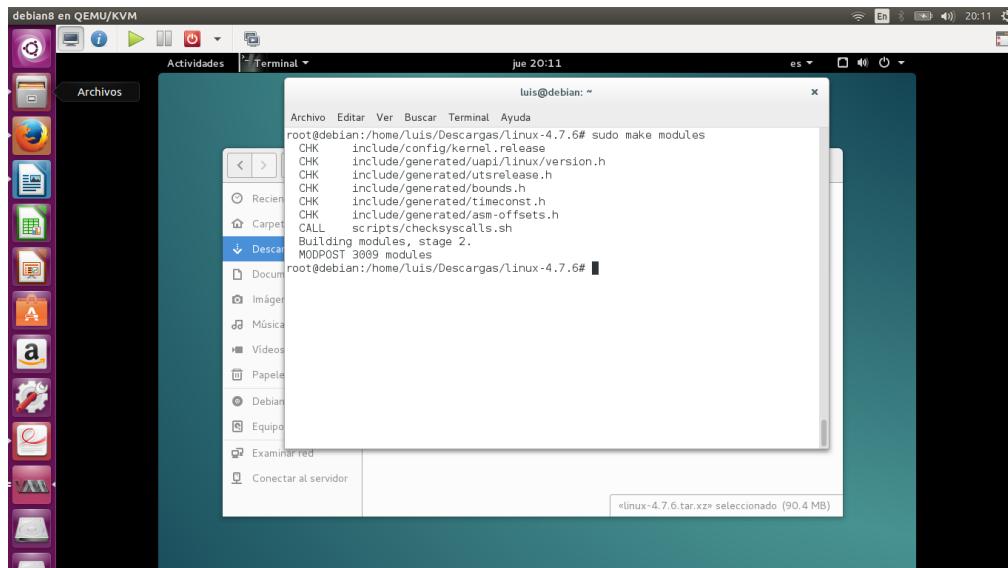


Como siguiente paso tenemos que compilar los módulos con:

---

*sudo make modules*

---

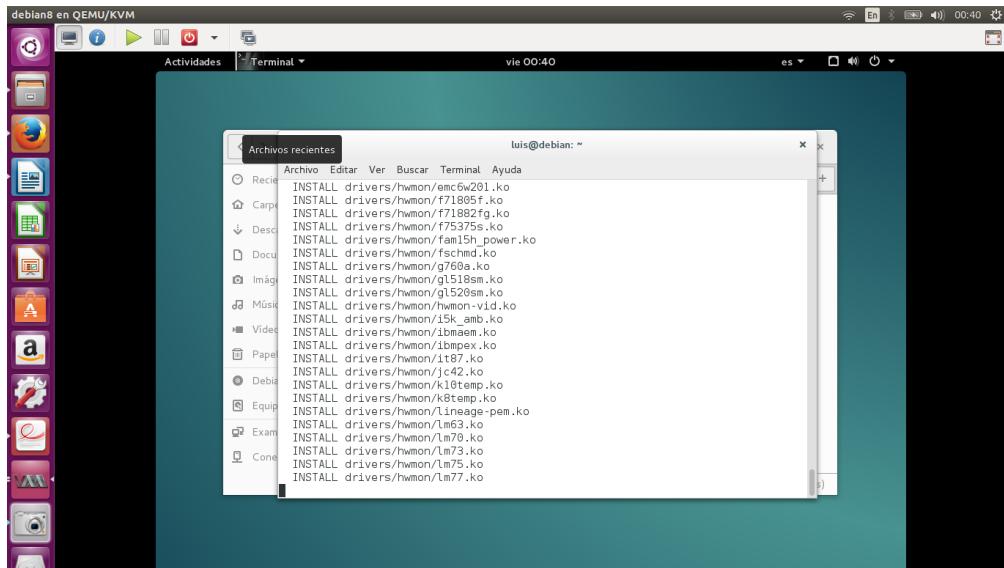


Al terminar instalamos los módulos con:

---

*sudo make module\_install*

---

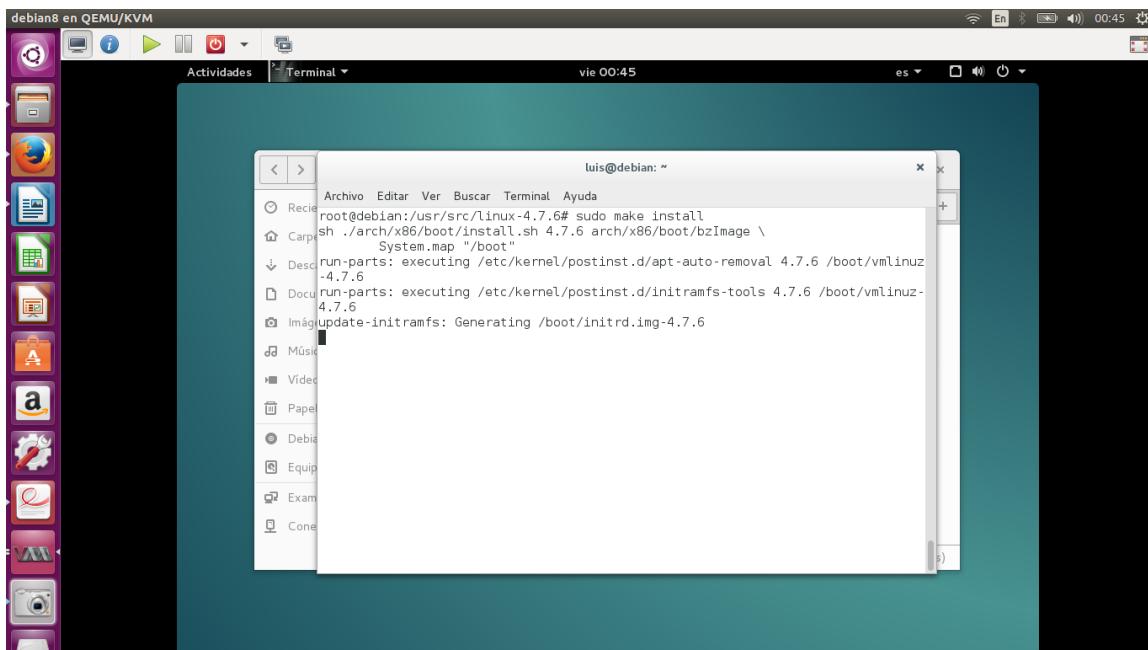


Ahora si podemos instalar nuestro nuevo kernel con:

---

*sudo make install*

---



Al finalizar la instalación reiniciamos la maquina y podemos ver nuestro nuevo kernel con:

---

*uname -r*

---

