KVM 簡介：

KVM，全名為 Kernel-based Virtual Machine，自 2006 年 12 月起，就屬於 Linux 核心架構下的一部分，簡單來說，就是以後 Linux 系統核心，就具備支援虛擬化的功能，並且是以核心模組化的方式載入執行，只要硬體搭配得宜（此指 CPU 支援虛擬化技術，也就是 Intel VT 或 AMD-V），再安裝 KVM 這一套虛擬系統，就可以直接在現有的 Linux 系統架構之下，建構裸機式（Bare-Metal）虛擬系統平台。

首先必須知道手上的電腦/伺服器的 CPU 是否可執行 KVM，也就是是否支援 Intel VT 或 AMD-V，如果沒有 CPU 的相關資訊（如型號或是說明文件），該如何知道 CPU 有沒有 Intel VT 或 AMD-V 呢？有以下幾種方式可以知道：

#所謂的〝可否執行 KVM〞，並不代表若 CPU 不支援 Intel VT 或 AMD-V 就不可以使用 KVM，而是指有沒有核心模組可以來釋放 KVM 全部的效能，若 CPU 不支援虛擬化技術，還是可以安裝與使用 KVM，不過實際在運作的就不是 KVM，而是 QEMU。若 CPU 沒有支援虛擬化技術，還是可以安裝 KVM 套件，只是系統會以 QEMU 來執行虛擬主機(速度很慢)。

可至 CPU 硬體廠商的官方網站，查詢 CPU 相關訊息即可得知。

Intel：http://www.intel.com/products/processor_number/

AMD：http://www.amd.com/tw/products/Pages/processors.aspx

1. 開啟 CPU 虛擬化設定

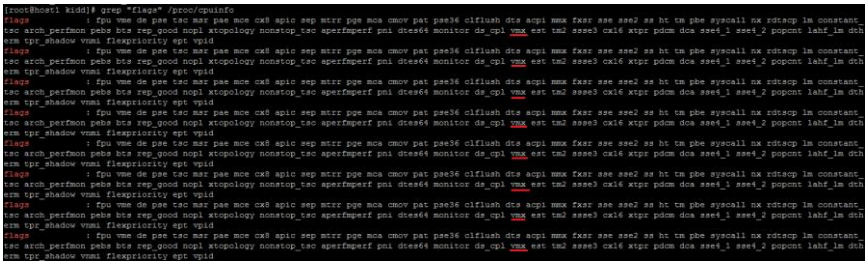
   安裝 KVM 之主機必須支援 CPU 虛擬化技術，如 AMD 的 AMD-V 或是 Intel 的 VT-x。

   安裝前，需先進入 BIOS 畫面選擇 Advanced-->CPU Configuration/Processer Configuration--> AMD-V(AMD Virtualization)/Intel-VT(Intel® Virtualization Technology) -->Enable。(各家主機板的選項名稱不一，請依個人設備規格而設定)

2. Linux 下檢查 CPU 是否支援 KVM

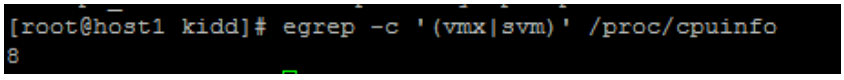
   若系統為 RedHat 或是 CentOS，必須注意 SELinux 是開啟的狀態，否則之後一些動作（vitr-install）將不會正常運作，請利用〝system-config-securitylevel〞指令來確認 SELinux 的狀態。

   透過〝egrep '(vmx | svm)' --color=always /proc/cpuinfo〞或〝egrep '(vmx | svm)' /proc/cpuinfo〞，來查看 CPU 相關訊息，如果什麼資訊都沒有，那就表示此電腦的 CPU 不支援虛擬化技術了。



指令：grep "flags" /proc/cpuinfo

如果輸出的信息中有 vmx，說明 Intel 處理器支持完全虛擬化。如果顯示 svm，說明是 AMD 的處理器支持虛擬化。



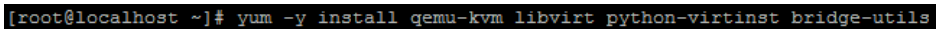
指令：egrep -c '(vmx|svm)' /proc/cpuinfo

若輸出為 0 則代表不支援；反之若輸出為 1 以上則代表主機支援 CPU 虛擬化技術。

3. 安裝 KVM 套件

套件名稱介紹：

套件名程	用途
qemu	一套完整和獨立的處理器模擬的自由軟體，由 Fabrice Bellard 所撰寫，Qemun 負責所有週邊硬體的模擬工作
qemu-kvm	使用者層級(user-level)的 KVM 模擬器
libvirt	是一套管理平台虛擬化的開放原始碼的程式介面、背景服務(libvirtd)和管理工具。可用來管理不同的虛擬化軟體，如 KVM、VMware、ESXi、Qemu、Hyper-V。libvirt 的背景服務程式 libvirtd 則負責處理程式呼叫並管理虛擬機器和控制 Hypervisor，提供 Hypervisor 與主機系統溝通之程式庫。
virt-install	提供建立虛擬機器的 virt-install 命令
libvirt-python	支援程式以 Python 語言呼叫 libvirt 的程式介面
bridge-utils	設置網絡網卡橋接



指令：yum -y install qemu-kvm libvirt python-virtinst bridge-utils

4. 啟動服務

啟動 libvirtd 及 messagebus 的服務，並設定成開機就啟動

```
[root@localhost ~]# /etc/rc.d/init.d/libvirtd start
Starting libvirtd daemon:                                  [  OK  ]
[root@localhost ~]# /etc/rc.d/init.d/messagebus start
Starting system message bus:                               [  OK  ]
[root@localhost ~]#  chkconfig libvirtd on
[root@localhost ~]#  chkconfig messagebus on
```

指令：/etc/rc.d/init.d/libvirtd start

/etc/rc.d/init.d/messagebus start

chkconfig libvirtd on

chkconfig mesagebus on

設定網路的橋接功能

1. 建立橋接網卡

進入/etc/sysconfig/network-scripts/目錄

```
[root@localhost ~]# cd /etc/sysconfig/network-scripts/
```

指令：cd /etc/sysconfig/network-scripts

2. 複製 ifcfg-eth0 成 ifcfg-br0

```
[root@localhost network-scripts]# mv ifcfg-eth0 ifcfg-br0
```

指令：mv ifcfg-eth0 ifcfg-br0

3. 編輯 ifcfg-br0 檔案

```
[root@localhost network-scripts]# vi ifcfg-br0
```

指令：vi ifcfg-br0

4. 修改 ifcfg-br0 中的 device、type 及 NM_CONTROLLED 的資料

```
DEVICE=br0
TYPE=Bridge
UUID=b0d87df5-7563-4937-80aa-7a79fb502767
ONBOOT=yes
NM_CONTROLLED=no
BOOTPROTO=none
HWADDR=52:54:00:42:E0:EC
IPADDR=103.198.82.5
PREFIX=24
GATEWAY=103.198.82.254
DNS1=8.8.8.8
DEFROUTE=yes
IPV4_FAILURE_FATAL=yes
IPV6INIT=no
NAME="System eth0"
```

NM_CONTROLLED 最好在網卡一設定時就改為”NO”，因為重啟 network 服務時，網卡內的 GW 設定可能會起不來；或是做 Bridge 的 MAC Address 會自動產生一個虛擬的 MAC Address。

5. 修改 ifcfg-eth0 的資料

```
[root@localhost network-scripts]# vi ifcfg-eth0
```

指令：vi ifcfg-eth0

6. 產生一個新的 eth0 檔案，並將它指向 br0

```
# create new
DEVICE=eth0
TYPE=Ethernet
ONBOOT=yes
BRIDGE=br0
NM_CONTROLLED=no
```

指令：

```
# create new
DEVICE=eth0
TYPE=Ethernet
ONBOOT=yes
BRIDGE=br0
NM_CONTROLLED=no
```

7. 編輯 sysctl.conf

```
[root@localhost ~]# vim /etc/sysctl.conf
```

指令：vi /etc/sysctl.conf

8. 開啟 IP forwarding，讓 guest 可以和 host 相通
   0 是”關閉”，1 是”開啟”

```
# Controls IP packet forwarding
net.ipv4.ip_forward = 1
```

設定 net.ipv4.ip_forward = 1

9. 重新啟動網路服務

```
[root@localhost network-scripts]# /etc/init.d/network restart
```

指令：/etc/init.d/network restart

10. Br0 已成功啟動

```
[root@localhost network-scripts]# /etc/init.d/network restart
Shutting down interface eth0:  bridge br0 does not exist!
                                                           [  OK  ]
Shutting down loopback interface:                          [  OK  ]
Bringing up loopback interface:                            [  OK  ]
Bringing up interface eth0:                                [  OK  ]
Bringing up interface br0:  Determining if ip address 103.198.82.5 is already in use for device br0...
                                                           [  OK  ]
```

11. 檢查網卡資料
    檢查 br0 的 Mac Address 是否為原本的 Physical Address，並非是重新產生的位址。

```
[root@localhost network-scripts]# ifconfig
br0       Link encap:Ethernet  HWaddr 52:54:00:42:E0:EC
          inet addr:103.198.82.5  Bcast:103.198.82.255  Mask:255.255.255.0
          inet6 addr: fe80::5054:ff:fe42:e0ec/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:15817 errors:0 dropped:0 overruns:0 frame:0
          TX packets:280 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:571560 (558.1 KiB)  TX bytes:22855 (22.3 KiB)

eth0      Link encap:Ethernet  HWaddr 52:54:00:42:E0:EC
          inet6 addr: fe80::5054:ff:fe42:e0ec/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:6077056 errors:0 dropped:0 overruns:0 frame:0
          TX packets:300080 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:501028168 (477.8 MiB)  TX bytes:29540029 (28.1 MiB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:16 errors:0 dropped:0 overruns:0 frame:0
          TX packets:16 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:1392 (1.3 KiB)  TX bytes:1392 (1.3 KiB)

virbr0    Link encap:Ethernet  HWaddr 52:54:00:58:39:42
          inet addr:192.168.122.1  Bcast:192.168.122.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
```

指令： ifconfig

12. 檢查網卡資料

當使用 ifconfig 無法實際或正確呈現網卡資料時，就可使用 ip addr 來檢視

```
[root@localhost network-scripts]# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether 52:54:00:42:e0:ec brd ff:ff:ff:ff:ff:ff
    inet6 fe80::5054:ff:fe42:e0ec/64 scope link
       valid_lft forever preferred_lft forever
3: virbr0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UNKNOWN
    link/ether 52:54:00:58:39:42 brd ff:ff:ff:ff:ff:ff
    inet 192.168.122.1/24 brd 192.168.122.255 scope global virbr0
4: virbr0-nic: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN qlen 500
    link/ether 52:54:00:58:39:42 brd ff:ff:ff:ff:ff:ff
6: br0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UNKNOWN
    link/ether 52:54:00:42:e0:ec brd ff:ff:ff:ff:ff:ff
    inet 103.198.82.5/24 brd 103.198.82.255 scope global br0
    inet6 fe80::5054:ff:fe42:e0ec/64 scope link
       valid_lft forever preferred_lft forever
```

指令：ip addr

參考資料：

CentOS 6 - KVM - Install : Server World
http://www.server-world.info/en/note?os=CentOS_6&p=kvm

傲笑紅塵路: 架設 Linux KVM 虛擬化主機(Set up Linux KVM virtualization host
http://www.lijyyh.com/2015/12/linux-kvm-set-up-linux-kvm.html

CentOS 6 install 安裝 kvm 虛擬機器- Linux 系列討論- ADJ 網路實驗室
http://dz.adj.idv.tw/thread-105185-1-1.html