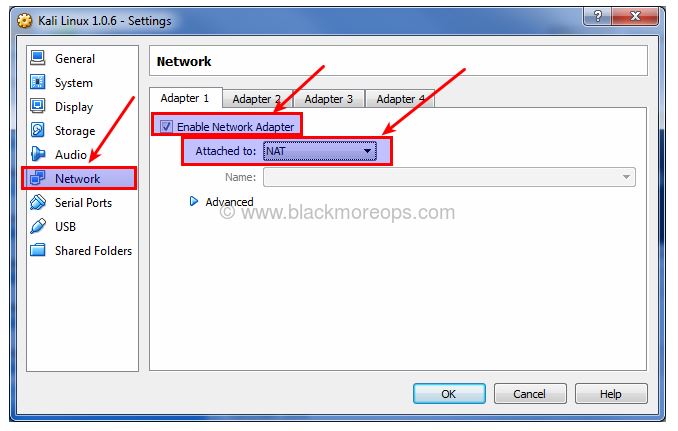
2017 / 3 / 31

Author: Simon Wang

開始學習 Linux Internals 和 Device Driver Programming 的第一篇文章，我的動機是想要在 Linux 系統上開始寫 Loadable kernel module ，本來依照著這篇 “Lesson 4 Writing and running your first kernel module” 文章的指示，我以為只要有 Linux Kernel Source Code 的 Kernel Library 的 Include 檔就可以寫 Kernel Module了，於是我就想辦法要從網路上下載 Linux Kernel Source Code 的 Header 檔的壓縮檔 -- kernel-headers-3.10.0-229.el7.x86\_64.rpm。

這時我用的是 Cent OS 7 + VMware Workstation Player 12.5.3(免費版) ，我碰到的第一個問題是我始終無法設定讓 Cent OS 連上網，我的網路環境是家用的ADSL上網、沒有任何的分享器，我嘗試了網路上的文章中的許多辦法都還是無解，由於現在要下載 Linux Kernel Source Code 幾乎都是利用 wget 或 apt-get 這樣的工具來直接下載，所以我原本的拷貝一份 Linux Kernel Source Code 的壓縮檔到 USB 硬碟、再用 USB 硬碟接上虛擬機 OS 來拷貝進去的想法是行不通了，所以我還是先得解決連線的問題。

經過多種嘗試後，我意識到這個連線的問題可能是出在 VMware 上，所以我改用了完全免費的 VirtualBox 5.1.14 ，同時作業系統我也改用了更新的 Kali Linux 2016.2 ，在按照了 “A detailed guide on installing Kali Linux on VirtualBox” 這篇文章的指示來安裝 VirtualBox ，來到下面這個畫面:



直接設定為 NAT 、其他什麼都不用動。

再繼續照著 “A detailed guide on installing Kali Linux on VirtualBox” 這篇文章的只是安裝完 Kali Linux 2016.2 ，在虛擬機內重新開機進去 Kali Linux 後，用:

ping 168.95.1.1 -c 3 (ping 這個中華電信的DNS伺服器)

nslookup [www.yahoo.com.tw](http://www.yahoo.com.tw) (查詢 Yahoo 的DNS名稱)

來測試網路，結果就通了! 不用在 Kali Linux 上有任何額外的設定，真的比 Cent OS 好用多了，自此可以下載我們的 Linux Kernel Source Code 。

實際開始要 Compile 新的 Kernel 時，發現了許多預期之外的障礙問題，對新手來說真的可以說是困難重重，所以我這篇文章側重的是在所出現問題的排解、比較重的篇幅跟圖示是在這，其他按照網路上的 Tutorials 可以順利通過的地方，就簡單帶過。

按照 “How to Compile and Install Linux Kernel v4.9.11 Source On a Debian / Ubuntu Linux” 這篇文章的指示，我們先把在 Debian 系列(Kali Linux 也是屬於 Debian 系列)下 Compile Kernel 所需的套件安裝完成:

$ sudo apt-get install git fakeroot build-essential ncurses-dev xz-utils libssl-dev bc

接著再安裝 kernel-package 這個套件:

$ sudo apt-get install kernel-package

再來就是下載 Linux Kernel Source Code 和 Kernel Source Code 的 Verify Sign 檔案(驗證 Kernel Source Code 檔案的完整性用)，此時最新的 Linux Kernel Source Code 版本是 4.9.13 :

$ wget https://cdn.kernel.org/pub/linux/kernel/v4.x/linux-4.9.13.tar.sign  
$ wget wget https://cdn.kernel.org/pub/linux/kernel/v4.x/linux-4.9.13.tar.xz

下載完後，解壓縮並使用 GnuPG 來驗證 Kernel Signatures :

$ unxz linux-4.9.13.tar.xz  
$ gpg --verify linux-4.9.13.tar.sign

第一次驗證時會出現錯誤訊息，因為沒有它的 Public Key ，所以要先從錯誤訊息中查看它的 RSA Key ID ，再用以下指令去 PGP Keyserver 把它的 Public Key 抓回來:

$ gpg --keyserver hkp://keys.gnupg.net --recv-keys 00411886

(這裡的 00411886 是這篇文章中的例子，實際要看你的機器上的錯誤訊息顯示的是多少)

取得 Public Key 之後，再驗證一次:

$ gpg --verify linux-4.9.13.tar.sign

驗證成功之後，解開剩下的壓縮並進入這個 Kernel Source Code 的目錄:

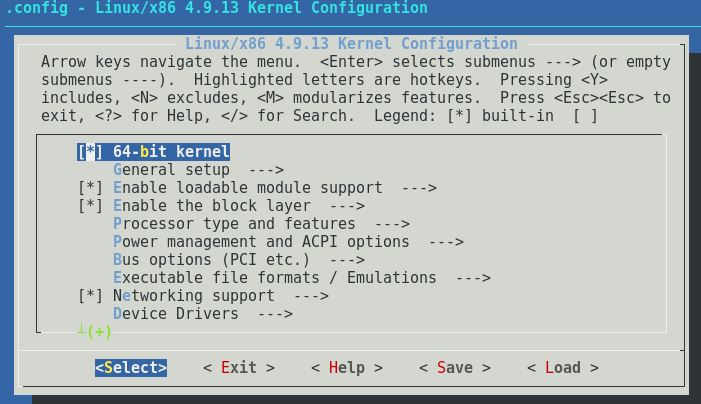
$ tar xvf linux-4.9.13.tar  
$ cd linux-4.9.13

由於我們目前的系統是正常且穩定的，所以第一次 Compile 新的 Kernel 實為了方便設定且不出錯起見，我們把舊的(目前系統正在跑的) Kernel 的 Configuration 檔案複製過來:

$ cp -v /boot/config-$(uname -r) .config

然後輸入重要的以視窗格式來設定 Kernel Configuration :

$ make menuconfig

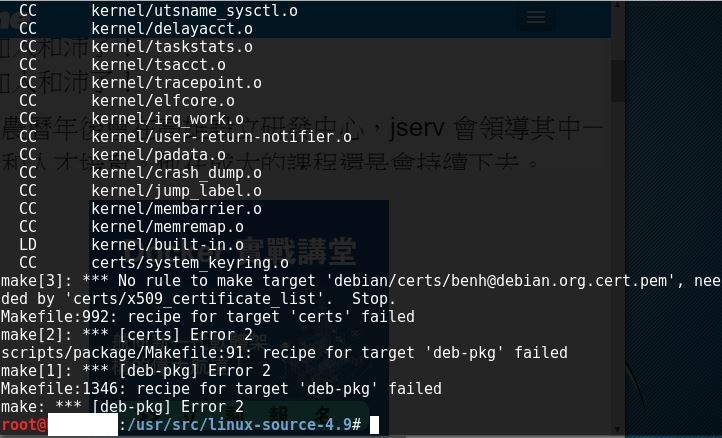


由於我們的目的只是想要得到一個可以開發 Kernel Module 的新 Kernel ，所以在確認了這個 Kernel 預設支援了 “Enable loadable module support” 之後，我們就不用再修改其它設定，按下 Exit 儲存退出。

接著清理 kernel-package 的 parameters :

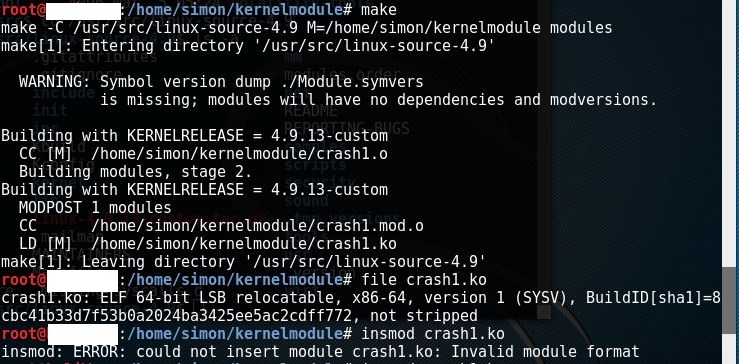
$ make-kpkg clean

然後我們以最基本的方式來 Compile 這個新的 Kernel ，就只要輸入 make 。結果跑了很久之後居然停在這裡，出現了以下的錯誤訊息:

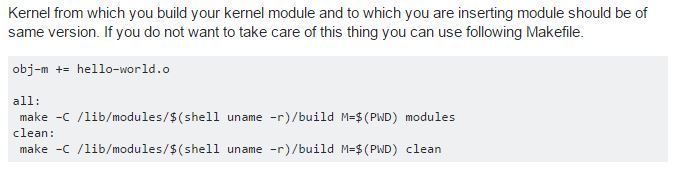


這時我有了一個想法，既然我已經得到完整的 Kernel Source Code 、包含其中的 Kernel Libraries 的 Include 檔了，也許我可以直接套用這個 4.9.13 的 Kernel 的 Include 檔來寫我的第一個 Kernel Module ?

所以在參考了 “Lesson 4 Writing and running your first kernel module” 這篇文章之後，我完成了 crash1.c 的程式，並且把 Makefile 中的 KERNELDIR ?= 調整為 /usr/src/linux-source-4.9 ，再執行 make ，通過了。我們用 file 指令來查看一下這個 .ko 檔，然後再執行 insmod crash1.ko ，卻出現以下錯誤:

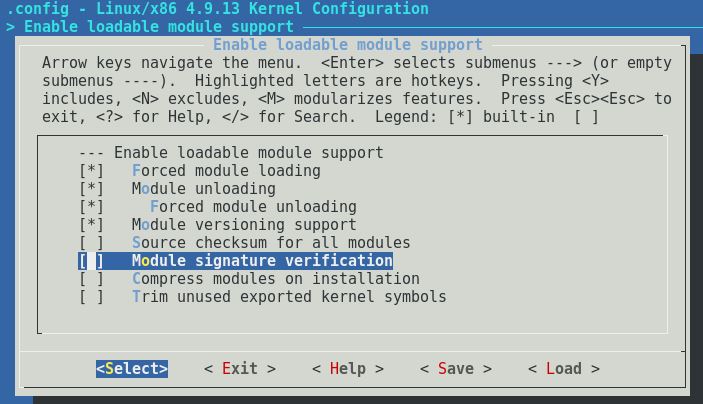


上網找答案後，發現問題是出在這裡:

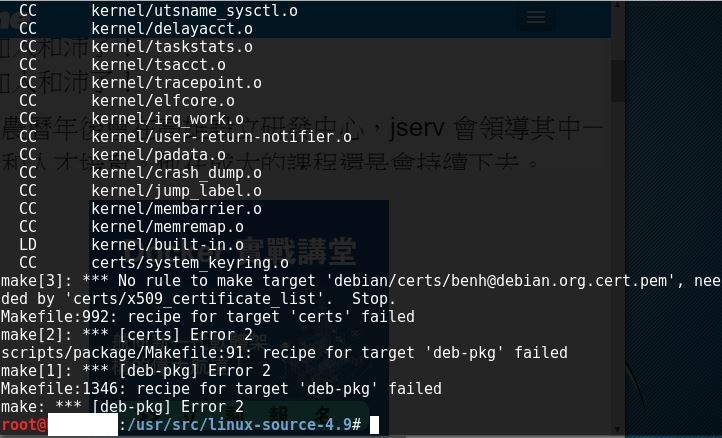


意思就是: 用來 Compile 這個 Kernel Module 的 Kernel 版本，必須要和它所要安裝上去的 Kernel 是相同版本，否則就會出現這樣的錯誤訊息。

所以我們還是得先解決這個 4.9.13 的 Kernel 它 Compile 不過的問題，我先是依網路上的文章、嘗試了在 make menuconfig 中、把這個 “Module signature verification” 功能給註解掉:



然後再 make 一次，結果還是出現相同的錯誤訊息:



困擾許久之後，我從這篇 “signed kernel modules using an auxiliary key” 文章中得到想法: 原來我們必須要先自行產生一個 x509 key configuration file 為 x509.genkey 之後，再以:

openssl req -new -nodes -utf8 -sha256 -days 36500 -batch -x509 \

-config x509.genkey -outform PEM -out kernel\_key.pem \

-keyout kernel\_key.pem

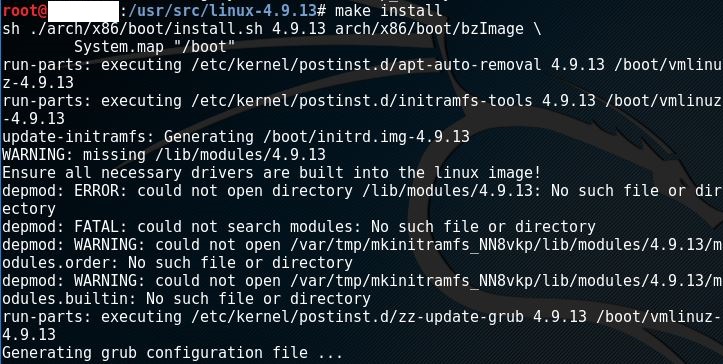
指令產生 key x509 keypairs 之後，才能在 Kernel Configuration 之中勾選 signed kernel modules 的相關選項。

短期要避免掉這個困擾的辦法，就只有在 Kernel Configuration 中，把 Module signature verification (CONFIG\_MODULE\_SIG) 這個選項不勾選之後，然後在 root of the Linux kernel sources 目錄裡:

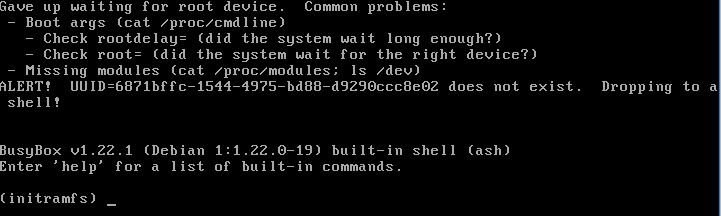
C:\Users\kdbot\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Compile Linux Kernel 4.9 on Kali Linux 時出現的錯誤 問題出在這裡 1.jpg

把這兩行選項都同時註解掉 # ，缺一不可，然後再次 Compile Kernel ，最後它還是會很囉唆的問你一次、要不要產生一個 Trust Key ? 這時候要記得輸入 N : 不要， Compile 才會繼續進行、且順利下去沒有錯誤了。

接著嘗試用 make install 來安裝新的 Kernel :



有點出現這樣的錯誤訊息，但還是完成了。保守起見，還是重新回到 make modules ，完成後再一次 make install && make modules\_install，結果完成了(但過程中還是會顯示上面的訊息)。接著用 cat /boot/grub/grub.cfg 來檢查，好像新的 Kernel 的開機選項已經有寫入了，就重開機進 Advanced Mode 並選擇新的 Kernel 來進入，結果等了很久，出現以下:

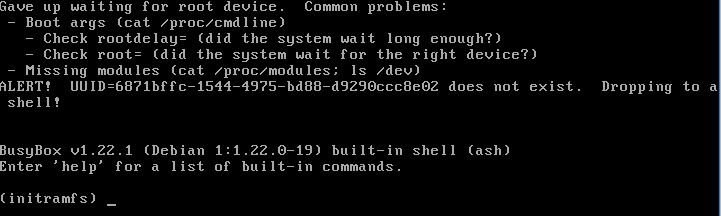


在找到 “After installing a new Kernel in Ubuntu 14.04, rebooting takes me to busybox initramfs. How do I mount a filesystem after that?” 這篇文章後，發現問題可能是我在 make modules、make install && make modules\_install 的順序出了問題，所以上頁第一個錯誤訊息顯示的是找不到相關的 modules 和 drivers ，沒有把 modules\_install 安裝上去、意即問題在於: 沒有把 Disk controller 的 Kernel Module 安裝上去。

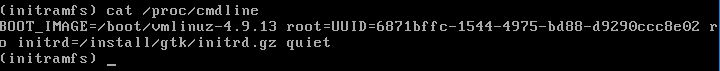
正確的順序應該是: 用 make 來 Compile Kernel 之後，先 make modules 、make modules\_install ，最後再 make install 。

不過既然很難得的碰到了這個狀況、進入了 initramfs: 這個介面，我不妨就來練習一下其他人怎麼來檢測硬碟故障的狀況。

我們開機後來到這個畫面:



我們照著它的指示，來看看是甚麼問題，輸入 cat /proc/cmdline :



顯示了新的 Kernel 版本號碼和 /root Partition 的UUID號碼，由於之前判斷是因為沒有 Disk controller 的 Kernel Module ，所以我們照著 “Gave up waiting for root device” 這篇文章的指示來測試硬碟方面的問題，先輸入 fdisk -l - 列出系統上的 Drives 和相關的 partitions :

C:\Users\kdbot\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Compile Kernel 重開機錯誤後嘗試檢查硬碟問題 3.jpg

居然找不到這個指令?? 我們再嘗試 blkid – 列出所有裝置的UUID :

C:\Users\kdbot\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Compile Kernel 重開機錯誤後嘗試檢查硬碟問題 4.jpg