NASA2022 Homework6

B08902149 資工三 徐晨祐

Network Administration

SSID/BSSID

(1.)
SSID (service set identifier)，為AP所設定的字串格式(最多32個字元組成)，方便使用者看得懂，也就是我們在連接網路服務時所看到的名稱，一個SSID可能由一個或多個AP所組成的集合所代表。
BSSID (Basic Service Set Identifier)，是AP的MAC address。
Reference:
https://www.juniper.net/documentation/en_US/junos-space-apps/network-director3.7/topics/concept/wireless-ssid-bssid-essid.html

(2.)
a. 同一個AP可以有多個SSID，例如系館可以設定某個SSID公開給訪客使用，IP可已和系上人員使用的不同網段。
b. 如果一個AP可以同時支援多個SSID的話，則AP會分配不同的BSSID來對應這些SSID。所以一個AP有可能同時有多個BSSID。
Reference:
https://www.sidmartinbio.org/can-access-point-have-multiple-ssid/
https://b8807053.pixnet.net/blog/post/349906417-wifi相關基礎概念

(3.)
a. 攻擊者偽裝成合法的AP，和某些真的和法的AP有著相同或類似的SSID，用以欺騙使用者。當使用者連上這個假的AP，攻擊者就可以對使用者進行一些攻擊，例如將某些網頁換成假的網站，造成使用者經濟上的損失或資安上的危險。假扮的基地台不易追蹤，因為駭客只要將電腦或裝置關閉，走人即可。
b. 不讓你的裝置可以自動連線wifi，避免你的裝置自動連上一個惡意但有相同SSID的wifi。
Reference:
https://techgenix.com/protect-against-evil-twin-attack/

PSK/EAP/PEAP

1. **PSK:** 指的是預共享金鑰。是一個客戶驗證的方式，使用64個16進位數字形成的字串，或是由8到63個ASCII character所形成之passphrase，來為每個client生成獨一無二的加密金鑰。欲連接AP者，只要輸入大家共用的密碼即可連接。
   EAP: 是一個認證的framework，常用在無線網路或點對點的連接。EAP protocol提供許多的認證機制，而不需要預先協商要使用特定一種。
   PEAP: 也是EAP的一種，最大的差別在使用了數位憑證，認證之過程會被公開鑰匙加密算法所保護，而私鑰只有Radius Server有，用以解開使用者的認證封包。透過安全通道增強破解的難度以保護機密的資料。
   Reference:
   https://helpcenter.engeniustech.com/hc/en-us/articles/4406436754587-What-is-a-pre-shared-key-PSK-
   https://www.cc.ntu.edu.tw/chinese/epaper/0006/20080920_6003.htm

2. PSK更適合用在personal network，事實上它是被設計給負擔不起802.1X驗證的個人用戶，因為用在企業內不適合把密碼設成同一個，舉個例子，一旦企業內部有人離職，那企業內部的wifi密碼可能就洩露了。
   Reference: https://www.globalsign.com/en/blog/wpa2-personal-or-enterprise

WiFi Certificate

1. WiFi Certificate是一種憑證，用來驗證所連接的裝置或服務確實是屬於使用者認為他們自己所到的組織；此外certificate也會負責加密使用者的裝置和wifi hosting device之間的連線。它可以提供安全的network access並增加對公用熱點的信任。
2. CA是憑證的簽發機構，其負責簽發、認證憑證，並且管理已頒發憑證的機關。CA必須制定具體的政策和步驟來驗證使用者身份並對使用者憑證進行簽章，來保證憑證之持有者的身份以及公鑰的所有權。

   Reference:
   https://zh.wikipedia.org/zh-tw/证书颁发机构
   https://www.quora.com/What-is-a-WIFI-certificate

Connect to WiFi with terminal

我用的是macOS，不確定其他OS是否適用以下指令。

```
networksetup -setairportpower en0 on #開啟wifi
networksetup -setairportnetwork en0 <SSID> <password> #連接wifi
```

```
(base) chriscyhxx@kelisixiaoxudeMacBook-Pro ~ % networksetup -setairportpower en0 on
(base) chriscyhxx@kelisixiaoxudeMacBook-Pro ~ % networksetup -setairportnetwork en0 Qoo 12345678
(base) chriscyhxx@kelisixiaoxudeMacBook-Pro ~ % ping google.com
PING google.com (142.251.42.238): 56 data bytes
64 bytes from 142.251.42.238: icmp_seq=0 ttl=113 time=19.780 ms
64 bytes from 142.251.42.238: icmp_seq=1 ttl=113 time=25.905 ms
64 bytes from 142.251.42.238: icmp_seq=2 ttl=113 time=19.487 ms
64 bytes from 142.251.42.238: icmp_seq=3 ttl=113 time=17.070 ms
64 bytes from 142.251.42.238: icmp_seq=4 ttl=113 time=20.638 ms
^C
--- google.com ping statistics ---
5 packets transmitted, 5 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 17.070/20.576/25.905/2.916 ms
(base) chriscyhxx@kelisixiaoxudeMacBook-Pro ~ %
```

Reference:
https://www.mattcrampton.com/blog/managing_wifi_connections_using_the_mac_osx_terminal_command_line/

AP channel

我們應該要避免使用較多人使用的channel，才能避免干擾，增加速度和穩定度。因此我們必須讓3台AP都去使用較少人用的channel，且盡量不要讓這三台都設在同個channel。
Reference: https://www.techbang.com/posts/9700-24ghz-wireless-router-channel-test

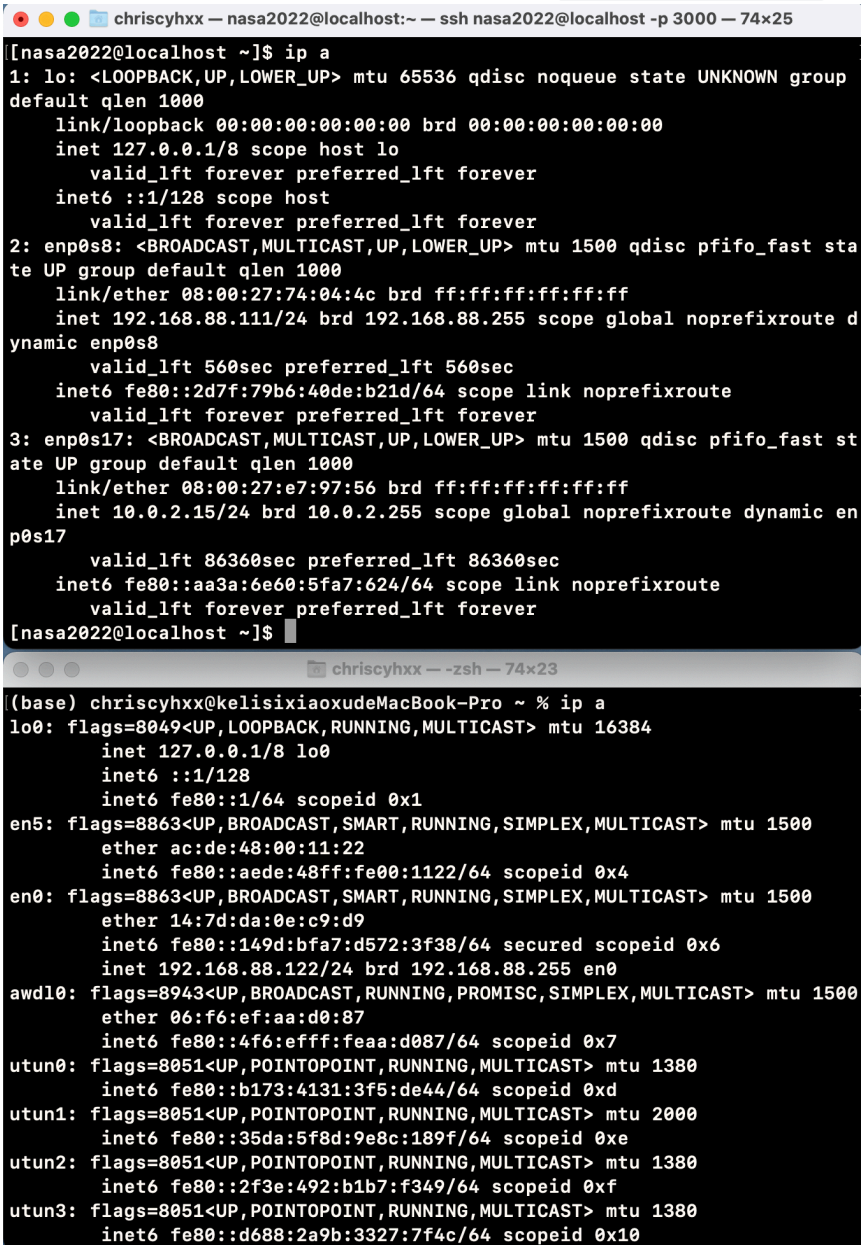
System Administration

Web Terminology

1. 第一，Nginx是用single-threaded architecture，多個client request由一個thread處理，而Apache是multi-threaded architecture，一個request就create一個process。
   第二，NGINX 能夠很好地處理靜態內容，比 Apache 快上2.5倍。 而Apache在Web服務器本身內處理動態內容，NGINX則不能在內部處理動態內容，並且依賴於外部的process來執行。
   第三，NGINX可以支援幾乎所有Unix作業系統，但只能支援部分的windows作業系統。而Apache可以支援所有Unix-like的作業系統和windows作業系統。
   Reference: https://www.interviewbit.com/blog/nginx-vs-apache/
2. Static web server只會提供事先寫好的檔案，也就是說它提供的檔案都是在被請求的當下就available的。而Dynamic web server則是基於Static web server再附加一些軟體，通常是application server和database，一個瀏覽器最終生成的頁面，是透過application server從database中讀取資料的HTML模板來填補出來的；舉個Dynamic web server的應用情境，像維基百科，有著成千上萬的網頁，但他們並非每個都是真的HTML文件，而是透過少數的HTML模板還有龐大的資料庫生成的，這樣做使得要維護並傳送資料，都會變得很容易。
   Reference:
   https://developer.mozilla.org/zh-TW/docs/Learn/Common_questions/What_is_a_web_server
3. Proxy是提供給客戶端的一種代理服務，可以允許客戶端透過這個服務和另一個伺服器進行非直接的連接。常見的三項功能如下：
   第一，代理伺服器通常有一個較大的緩衝區，可以將常用的網路資訊儲存在這個緩衝區中，當有客戶端試著透過該Proxy向外連線取得資訊時，Proxy會先看這個東西是否被儲存下來了，如果有的話就可以直接回傳給客戶端，提升存取速度。
   第二，可以透過Proxy隱藏真實的IP。
   第三，有些國家有網路審查機制，透過Proxy可以突破言論審查，存取被過濾的網站。
   Reference: https://zh.wikipedia.org/zh-tw/代理服务器
4. Reverse proxy是提供給伺服器端的一種代理服務，當客戶端連上某個服務時，實際上是連接到一個代理伺服器，Reverse proxy會根據客戶端的需求，向其後端的伺服器取得資源，並將這些資源回傳給客戶端，客戶端只會知道Reverse proxy的IP，而不會知道其後端還有很多負責處理不同服務的伺服器。使用Reverse proxy常見的三個功能如下：
   第一，可以做到load balancing，當某個伺服器負荷較高，可以透過Reverse proxy請負載較低且提供相同福的伺服器進行支援。
   第二，對客戶端隱藏Reverse proxy後端實際提供服務的server IP，可以更加安全。
   第三，有caching的功能，可以將某些資訊儲存下來，當有人再存取已經記錄下來的資訊，就可以直接回傳，而不用再去請後端的server服務。
   Reference: https://www.cloudflare.com/zh-tw/learning/cdn/glossary/reverse-proxy/

Web Server Configurations

1. **Basic Setups**
   首先我們在vm的網路設定中先多加一個Brdiged Adapter，開機後透過`ip a`指令檢查，可以看到此時VM和本機在相同網域。如下所示，會發現兩者都在`192.168.88.111/24`這個sunbet底下。



   接著我們安裝Apache server，我們先設定一下防火牆：

```
sudo systemctl start firewalld
sudo systemctl enable firewalld
sudo firewall-cmd --zone=public --add-service=http --permanent
sudo firewall-cmd --zone=public --add-service=https --permanent
sudo firewall-cmd --reload
```

   安裝Apache並啟動：

```
sudo yum install -y httpd
sudo systemctl start httpd
sudo systemctl enable httpd
```

Reference: 無

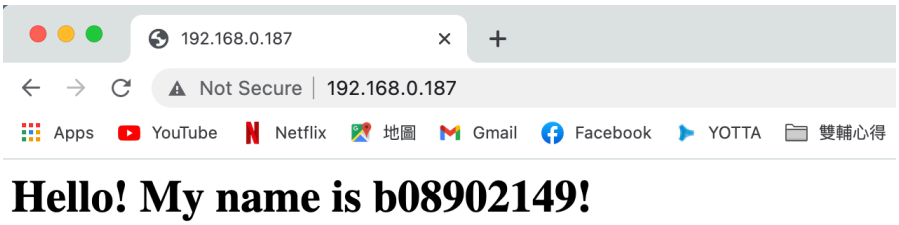
2. **Firewall Settings**

```
sudo firewall-cmd --add-port=80/tcp --permanent
sudo firewall-cmd --add-port=80/udp --permanent
```

Reference: 無

3. **The Main Page**
在`/var/www/html`裡面新增一個`index.html`，將內容設定為：

```
<h1>Hello! My name is b08902149!</h1>
```



Hello! My name is b08902149!

Reference: 無

4. **User Directory**
先把`/etc/httpd/conf.d/userdir.conf`改成以下：

```
<IfModule mod_userdir.c>
    UserDir enabled
    UserDir /home/*/public_html
</IfModule>

<Directory "/home/*/public_html">
    AllowOverride FileInfo AuthConfig Limit Indexes
    Options MultiViews Indexes SymLinksIfOwnerMatch IncludesNoExec
    Require method GET POST OPTIONS
</Directory>
```

然後執行`sudo systemctl restart httpd`。
接著，我們先建立`/home/b08902149/public_html`，其中`public_html`就類似工作站的`htdocs`。

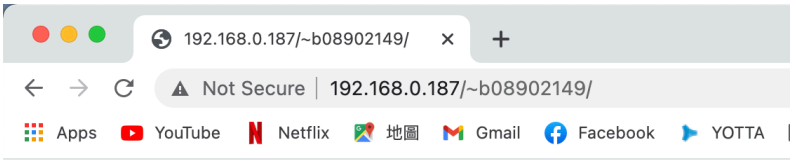
```
sudo mkdir -p -m 755 /home/b08902149/public_html
```

我們在`/home/b08902149/public_html`新增`index.html`，內容為：

```
<h1>Hello! My name is b08902149!</h1>
```

再執行以下指令：

```
sudo setsebool -P httpd_enable_homedirs true
sudo chcon -R -t httpd_sys_content_t /home/b08902149/public_html
```



Hello! My name is b08902149!

Reference:
userdir使用：https://www.tecmint.com/enable-apache-userdir-module-on-rhel-centos-fedora/
SELinux介紹：https://notes.wadeism.net/post/learning-selinux/

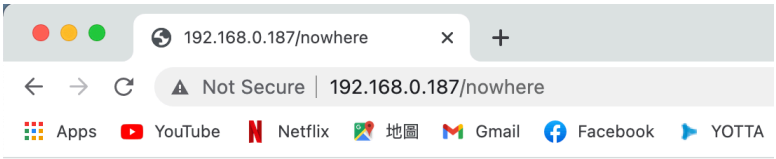
5. **404 Not Found**
   我們可以在`/var/www/html`裡，新增一個檔`error_404.html`當作404 error發生時的頁面，內容如下：

```
<h1>Sorry, the page doesn't exist...</h1>
```

接著我們在`/etc/httpd/conf.d`新增`custom_errors.conf，內容如下：

```
ErrorDocument 404 /error_404.html
```

其中`ErrorDocument`指令是用來讓apache知道每個error發生時所對應到的是哪個頁面。



Sorry, the page doesn't exist...

Reference:
自定義Error Pages: https://www.digitalocean.com/community/tutorials/how-to-configure-apache-to-use-custom-error-pages-on-centos-7

6. **Secret**
   我們先在`/var/www/html`新增`secret.html`，接著我們在`/etc/httpd/conf/httpd.conf`新增下列內容：

```
<Files "secret.html">
    Order allow,deny
    Allow from 192.168.28.0/24
</Files>
```

就能讓只有`192.168.28.0/24`內的ip可以透過`http://{your_vm_ip}/secret.html`來連上`secret.html`。

Reference:
https://www.cyberciti.biz/faq/apache-restrict-access-based-on-ip-address-to-selected-directories/

Reverse Proxy

我是開3個centos 7的vm當作Proxy、host A、host B。
先在reverse proxy上，在`/etc/yum.repos.d/nginx.repo`編寫Nginx套件位置，內容為：

```
[nginx]
name=nginx repo
baseurl=http://nginx.org/packages/centos/$releasever/$basearch/
gpgcheck=0
enabled=1
```

接著透過`sudo yum install -y nginx`安裝。
以下列指令設定防火牆：

```
sudo firewall-cmd --permanent --zone=public --add-service=http
sudo firewall-cmd --permanent --zone=public --add-service=https
sudo firewall-cmd --reload
```

以下列指令啟動Niginx：

```
sudo systemctl enable nginx
sudo systemctl start nginx
```

在此，我讓host A和host B分別開一個web server，其`index.html`有分別寫著host A和host B，以利檢查。
接著修改Proxy的`/etc/nginx/conf.d/default.conf`為以下內容：

```
server {
    listen       80 default_server;
    listen   [::]:80 default_server;
    server_name  _;

    location / {
        root   /usr/share/nginx/html;
        index  index.html index.htm;
    }
    location /hostA {
        proxy_pass http://10.217.44.28/;
    }
    location /hostB {
        proxy_pass http://10.217.44.6/;
    }
}
```

最後執行`sudo systemctl restart nginx`來重啟服務即可。
此時就可以透過`http://{your_public_ip}/hostA`來access到host A，透過`http://{your_public_ip}/hostB`來access到host B。
討論對象：`b09505014 王聖文`