Network Administration/System Administration Homework #6

B10202012 劉仲楷

1 Short Answer

1. (a) Static web server 直接回傳靜態檔案，如 HTML、圖片等，這種速度快、架構簡單。而 Dynamic web server 還會執行後端程式，如 PHP、Python 動態生成頁面。這種則可以更彈性，但負載也較高。

   (b) Forward proxy 由 client 端設定，用來代表 client 對外發送請求。Server 端不知道實際 client IP。而 Reverse proxy 則相反，由 server 端設定，用來代表 server 接收 client 的 request，再 forward 到內部伺服器。優點是

   - 匿名性、安全性提升：隱藏 client/server 真實 IP，讓 attacker 不能輕易知道內部的 topology。
   - 效能提升：可以用 cache 等功能在 forward proxy，這樣就可以減少對外的 request 以及減少 RTT。
   - 伺服器架構更清晰，更好維護：如果有多台伺服器，則可以用 reverse proxy 依據特定 policy forward 到各 server。想增加 server 或 server 故障、下線都更好管理。

   (c) Load balancing 是將多個 client request 依據 workload 合理分配到各個 server 上的過程。希望能夠避免單一伺服器負擔太大。其中一種方法是 round robin，就是依序分配給各個 server。假設 k 個伺服器，則第 n 個 request 就會被分配到第 $n \mod k$-th server。

2. (a) Apache 每個連線都有個別對應的 process 或 thread，因此處理不同連線之間的 context switch 成本很高。而 Nginx 則是用 async 和 event loop 的方式，只需要少數 worker 就能處理很多連線。（ref:Apache vs Nginx）

   (b) Master process 負責載入設定檔、bind to network port、管理 worker process（包含動、監控、重開）等；Worker process 則實際負責處理的 requests，各個 worker process 都獨立運作，且使用 async I/O 和 event loop 的方式。（ref: Nginx Master-Worker Architecture）

   (c) Master process 是 root；Worker process 是 www-data。這個設計也讓系統更安全。（ref: Nginx Master-Worker Architecture）

3. (a) PKI 是一套用來管理用 public key 認證的系統。我們知道非對稱加密可以用來做 certificate，以達到身份認證（也能順便保證 integrity 和 confidentiality），那問題就在於我們怎麼拿到對方的 certificate。因此 PKI 建立了 chain of trust 的概念。如果有上層 CA（Certificate Authority）認證下層的 certificate，則我們就信任這個 certificate，直到最上層（Root CA）通常是內建於作業系統或瀏覽器內的（user 起碼應該相信作業系統或瀏覽器）。TLS 則是能用憑証來在握手階段驗證對方身份。

(b) ACME（Automatic Certificate Management Environment）是 Let's Encrypt 等 CA 所使用的自動化憑證發放協定，可以讓伺服器自動完成申請、驗證與安裝 SSL 憑證。在 client 像 CA 提出要簽發 certificate 後，CA 會發出一些 challenges。例如在 client domain 下特定路徑放指定的 HTTP 文件，或是在該 domain 的 DNS 記錄放置驗證 token。目的是證明這個網域真的是這個 client 所有。（ref: Let's Encrypt 的運作原理）

(c) 一個 domain 可以對應到很多 IP，且 IP 很容易變動。這會導致需要驗證的內容變多，以及需要常常註銷、重新驗證 certificate。此外，用 domain name 的好處是使用者也比較直覺能夠驗證（畢竟沒人會記 IP 對吧）。

(d) SSL termination 是指 reverse proxy 會負責解密 https 流量，再以 http 的方式發送到內部伺服器。加上

```
proxy_set_header X-Forwarded-Proto https;
proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
proxy_set_header X-Real-IP $remote_addr;
```

可以讓內部伺服器從 header 判斷原始來源是 https 流量。（ref: SSL Termination for TCP Upstream Servers）

2 Web Server Configurations

1. 用以下指令安裝 Nginx 和防火牆（ref: man ufw）

```
sudo apt install nginx ufw
```

用以下指令設定防火牆

```
sudo ufw allow 22/tcp
sudo ufw allow 80/tcp
```

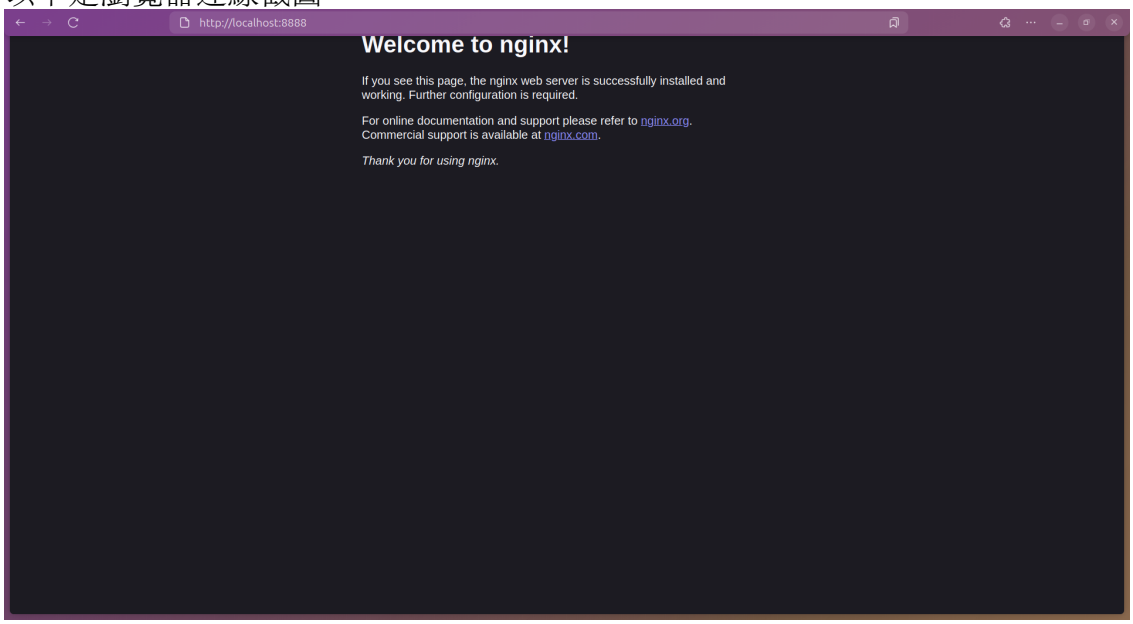
以下是 `sudo ufw status verbose` 的結果

```
b10202012@b10202012:~$ sudo ufw status verbose
Status: active
Logging: on (low)
Default: deny (incoming), allow (outgoing), disabled (routed)
New profiles: skip

To                         Action      From
--                         ------      ----
22/tcp                     ALLOW IN    Anywhere
80/tcp                     ALLOW IN    Anywhere
22/tcp (v6)                ALLOW IN    Anywhere (v6)
80/tcp (v6)                ALLOW IN    Anywhere (v6)
```

此外，也用 `curl` 檢查防火牆正確設定。

```
b10202012@b10202012:~$ curl -I http://localhost:80
HTTP/1.1 200 OK
Server: nginx/1.22.1
Date: Sun, 26 Oct 2025 08:22:50 GMT
Content-Type: text/html
Content-Length: 615
Last-Modified: Sat, 25 Oct 2025 10:08:40 GMT
Connection: keep-alive
ETag: "68fca1a8-267"
Accept-Ranges: bytes

b10202012@b10202012:~$ curl -I http://localhost:443
curl: (7) Failed to connect to localhost port 443 after 0 ms: Couldn't connect to server
```

以下是瀏覽器連線截圖。



2. 在 `/etc/nginx/sites-enabled/default` 內找到 `index.html` 位置，在 `/var/www/html/index.html` 裡寫入 `b10202012: Default`。結果如下：

3. 新增一個 `/var/www/html/403.html`，並寫入 `b10202012: 403 Forbidden`。新增一個 forbidden 資料夾，並更改權限至不可讀。

```
sudo mkdir /var/www/html/forbidden
sudo chmod 000 /var/www/html/forbidden
```

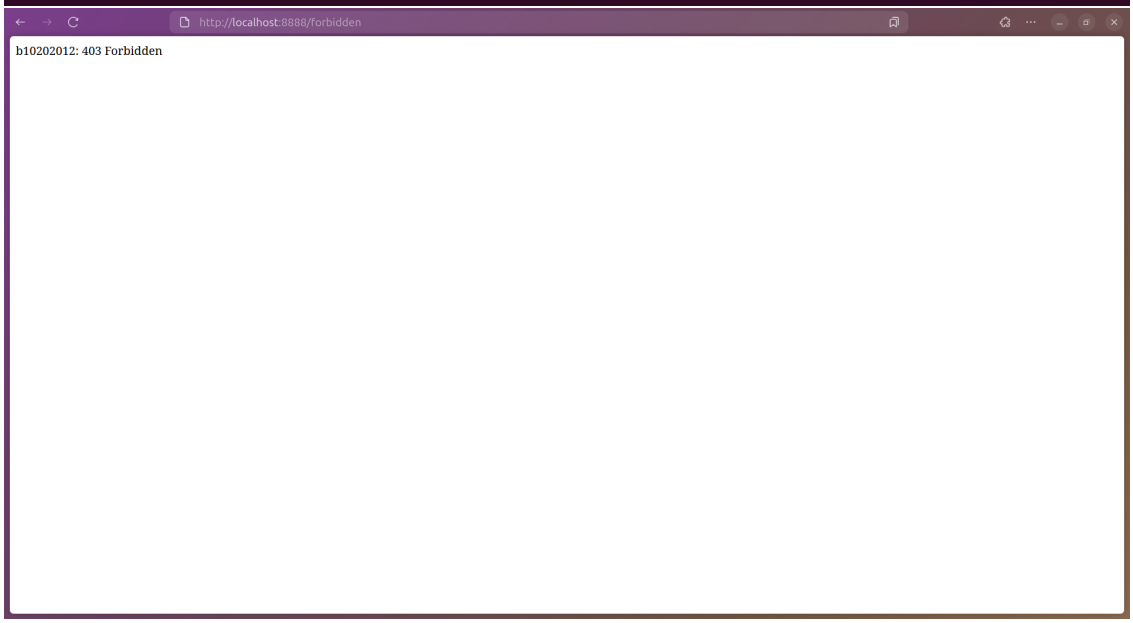
在 `/etc/nginx/sites-enabled/default` 的 `server{...}` 中加入

```
error_page 403 /403.html;
location = /403.html {
    root /var/www/html;
```

```
    internal;
}
```

使用 `curl -I http://localhost/forbidden` 測試 Status code。Status code 和瀏覽器結果如下：

```
b10202012@b10202012:/var/www/html$ curl -I http://localhost/forbidden
HTTP/1.1 403 Forbidden
Server: nginx/1.22.1
Date: Sun, 26 Oct 2025 15:43:07 GMT
Content-Type: text/html
Content-Length: 25
Connection: keep-alive
ETag: "68fdde7f-19"
```



4. 用以下指令讓所有新增使用者的假目錄都有 htdocs，並更改權限讓 nginx 抓的到。

```
sudo mkdir -p /etc/skel/htdocs
sudo chmod 711 /etc/skel
sudo chmod 755 /etc/skel/htdocs
```

接在 /etc/nginx/sites-enabled/default 裡的 `server{...}` 中加入

```
location ~ ^/~([a-zA-Z0-9_-]+)(/.*)?$ {
    alias /home/$1/htdocs$2;
    index index.html;
}
```

最後，新增 user1。

```
sudo adduser user1
sudo mkdir -p /home/user1/htdocs
sudo chown -R user1:user1 /home/user1/htdocs
```

可能會看到一些要輸入的內容，可以隨便輸。這時候可以看到家目錄底下已經有 htdocs 了。接在 /home/user1/htdocs 下建立 index.html，內容要有 user1: UserDir。要注意的是可能需要 `chmod 644` nginx 才抓得到。以下是結果：

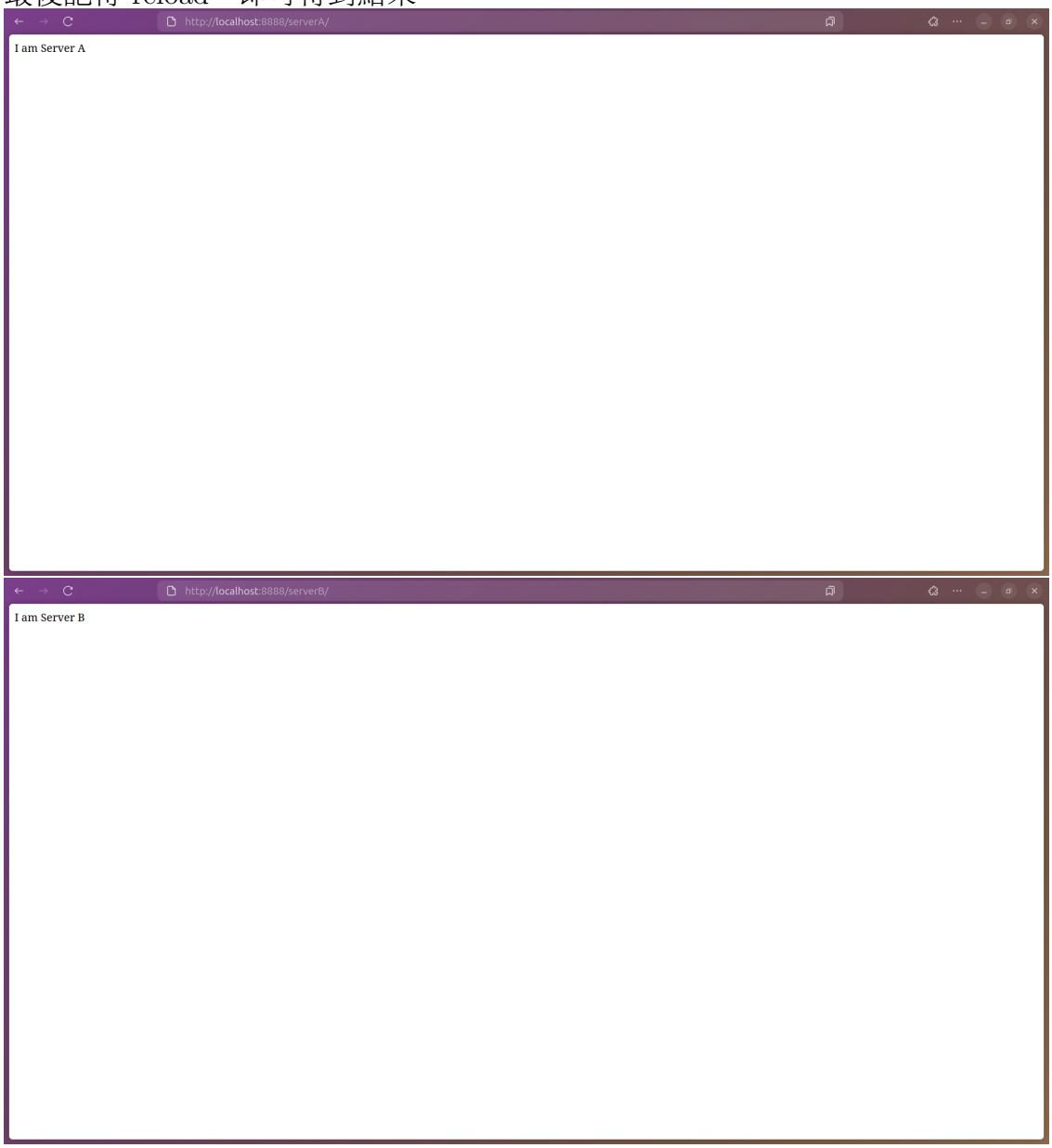
5. ref: 使用 Nginx 提供 http 反向代理
寫到這裡我才發現需要用 port 8888，我前面把 VM 的 port 80 forward 出來成 8888 了，但這不影響，只是避免誤會記錄以下。我用 python 開兩個 server：

```
sudo mkdir -p /srv/serverA
echo "I am Server A" | sudo tee /srv/serverA/index.html
cd /srv/serverA
python3 -m http.server 7777
sudo mkdir -p /srv/serverB
echo "I am Server B" | sudo tee /srv/serverB/index.html
cd /srv/serverB
python3 -m http.server 8888
```

並在 /etc/nginx/sites-enabled/default 裡的 `server{...}` 中加入

```
location /serverA/ {
    proxy_pass http://127.0.0.1:7777/;
    proxy_set_header Host $host;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header X-Forwarded-Proto $scheme;
}

location /serverB/ {
    proxy_pass http://127.0.0.1:8888/;
    proxy_set_header Host $host;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header X-Forwarded-Proto $scheme;
}
```

最後記得 reload。即可得到結果。



6. ref: Using nginx as HTTP load balancer
   同樣先新增 Server C：

```
sudo mkdir -p /srv/serverC
echo "I am Server C" | sudo tee /srv/serverC/index.html
cd /srv/serverC
python3 -m http.server 9999
```

   在 /etc/nginx/sites-enabled/default 裡的 `server{...}` 中加入

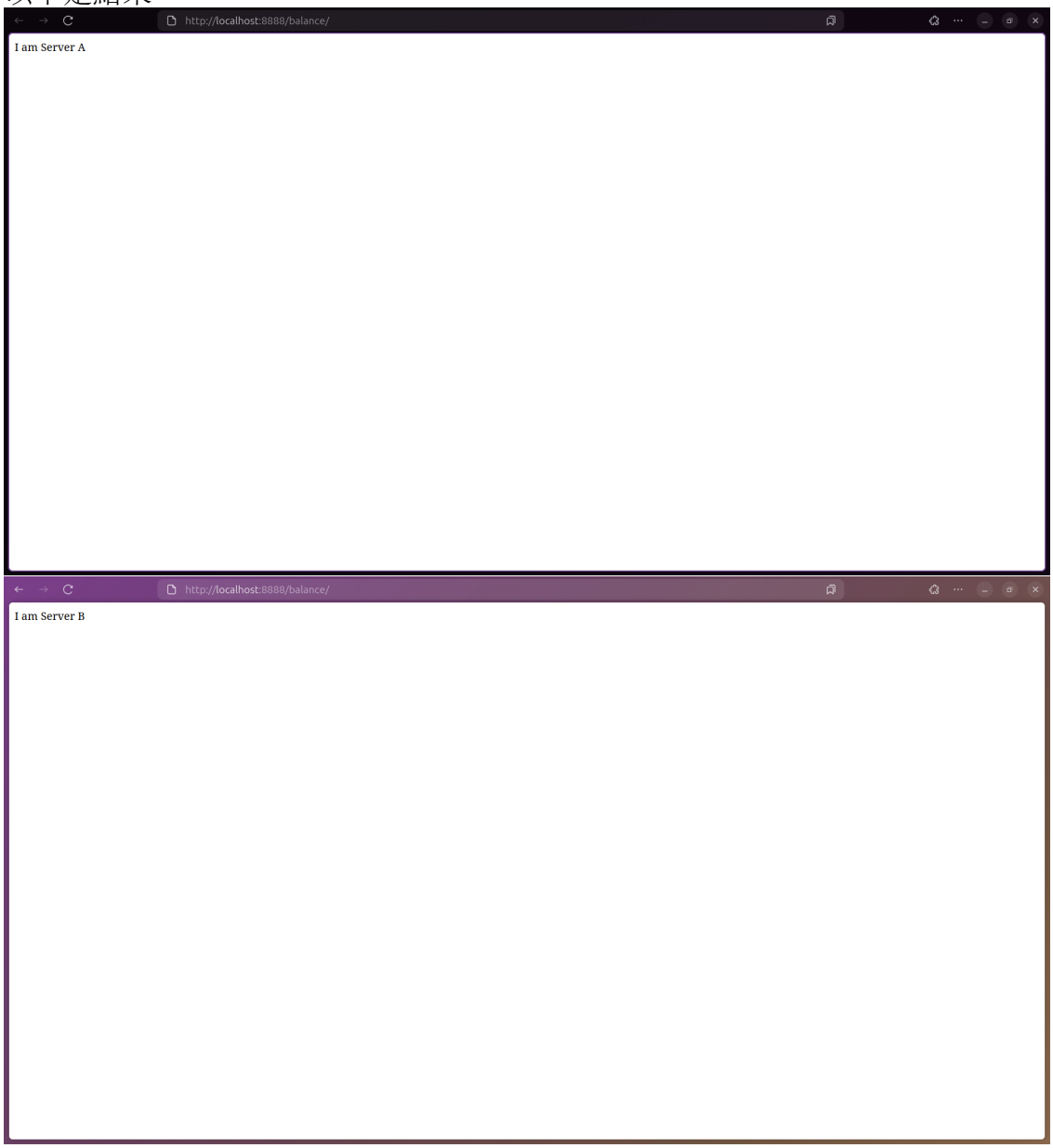
```
location /balance/ {
    proxy_pass http://backend_cluster/;
    proxy_set_header Host $host;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
```

```
    proxy_set_header X-Forwarded-Proto $scheme;
}
```

在 `server{...}` 外加入

```
upstream backend_cluster {
    server 127.0.0.1:7777 max_fails=3 fail_timeout=30s;
    server 127.0.0.1:8888 max_fails=3 fail_timeout=30s;
    server 127.0.0.1:9999 backup;
}
```

以下是結果



有時候網頁會被 cache 住，要開其他頁面才能得到。這是直接用 `curl` 的結果，可以更清楚看到 round robin：

```
joshua@joshua-Aspire-A514-55:~/.ssh$ curl http://localhost:8888/balance/
I am Server A
joshua@joshua-Aspire-A514-55:~/.ssh$ curl http://localhost:8888/balance/
I am Server B
joshua@joshua-Aspire-A514-55:~/.ssh$ curl http://localhost:8888/balance/
I am Server A
joshua@joshua-Aspire-A514-55:~/.ssh$ curl http://localhost:8888/balance/
I am Server B
joshua@joshua-Aspire-A514-55:~/.ssh$ curl http://localhost:8888/balance/
I am Server A
joshua@joshua-Aspire-A514-55:~/.ssh$ curl http://localhost:8888/balance/
I am Server B
```

把 ServerA、ServerB 都關掉後，可以看到連線到 ServerC：