Network Administration and System Administration

Midterm Examination

Time: 2021/4/19 (Mon.) 09:10 - 12:10

Instructions and Announcements

- 考試時間共三小時，三人一組考試。
- 考試期間禁止使用任何通訊軟體與外界交流，也禁止組與組之間一切討論與合作，如被發現視為作弊行為，**期中考 0 分**。
- 作答過程中請自行斟酌備份，避免電腦發生意外，損失過多進度，如使用 204 電腦請自行準備隨身碟備份作答進度。
- 為避免發生重大意外，請自行注意 VM 用量，同時開啟過多 VM 可能導致電腦當機，我們恕不負責。
- 助教將會在給定的 Docker container 中測試 shell script 題目，請確保程式能正確地在上面執行。
- 完成題目時請至 https://forms.gle/hZRniL9xAEdfS5EPA 上傳作答內容，每組每題 **最多上傳 3 次**。
- 組與組間**禁止討論**。考試期間禁止使用手機、電話、任何通訊軟體等與同組成員外任何人聯繫。如被發現將視為作弊行為，**期中考以 0 分計**，並依校規懲處。
- 各題後面黑色星號數目代表我們估計的難度。請參考，可用來決定解題順序。
- 滿分為 205 pts。

Shell Scripting

在作答第四題「By the way, I use Arch」時，請確保你的腳本能在我們提供的 Arch Linux ISO 中執行。

對第五題「Fediverse」和第六題「ZIP Checker」而言，我們會提供一個 Docker image，而你的腳本僅可以使用該映像檔中的工具。另外，任何非必要的網路連線（也就是除了和第五題的伺服器溝通所需以外），將會在測試時被阻擋，還請注意。

此 Docker image 可以在給定的壓縮檔中找到，並能夠（在 `amd64` 平臺上）使用以下的指令匯入：

```
docker image load -i nasa-image.tar
```

sha1sum:

- `14f60487e14a80152613f1ac60612c0700c373ea` Dockerfile
- `79ba2b1e0020f303a30b80fd8240df46c1fad193` nasa-image.tar

1 Wireshark (22 points)

Resources

- Problem 1.1: p1-1b.pcapng (sha1sum: `1f0397b2cc51bcebcbaafc094d45a8e329df1c48`), p1-1c.pcapng (sha1sum: `b5c30d7ecd2c8f751e6bff616f6fb04b2a10bd39`)
- Problem 1.2: p1-2.pcapng (sha1sum: `e0156453aa2659c748489e9357fd8c51e408b077`)
- Problem 1.3: p1-3.pcapng (sha1sum: `701d217a73c6b2c8323650305730f9dd69ec9ab0`)

1.1 Robert messed up everything ★★☆☆☆ (6 points)

(a) 請紀錄你在 https://google.com 上搜尋 `teamXX` (XX 為你的 team id) 的過程，並存入檔案 `p1-1a.pcapng`。上傳時請將 `p1-a.pcapng` 和對應的 pre-master secret 一併壓縮上傳。建議在錄製的時候開一個新的無痕視窗避免瀏覽器 cache 的問題，可以的話還請先在本機確認有沒有錄製成功，避免浪費上傳次數。

(b) Robert 有天很粗心的把他的 pre-master secret 外洩出去了！請從 `p1-1b.pcapng` 找出他的 pre-master secret。
(hint: tshark)

(c) 現在你有了 Robert 的 pre-master secret (上一題)，請使用這個 key 去 decrypt `p1-1c.pcapng` 並撈出 Robert 用來登入 `nasa.giversostrong.ninja` 的帳號和密碼。

(hint: search "wireshark, https, pre-master secret")

1.2 Who is "DDoS" ing? ★★☆☆☆ + 0.5★ (6 points)

請從 `p1-2.pcapng` 裡找出所有向 `127.0.0.53` 發出超過 1 次的 DNS request 的 IP 們。(hint: tshark)

1.3 Who is tracerouting? ★★★☆☆ (10 points)

請從 `p1-3.pcapng` 找出 traceroute 的 source 和 destination。Source 請以 ip 回答，destination 則是以 FDQN 回答。(hint: how does traceroute work?)

Submission

- Problem 1.1(a): 把壓縮檔 (命名為 `p1-1a.zip`) 透過 Google 表單上傳。
- Problem 1.1(b): 把 pre-master secret 存成文字檔 (命名為 `p1-1b.txt`) 透過 Google 表單上傳。
- Problem 1.1(c): 把帳號 / 密碼透過 Google 表單上傳。
- Problem 1.2: 把那些 IP 存成文字檔 (命名為 `p1-2.txt`) 透過 Google 表單上傳。
- Problem 1.3: 把 source(IP) / destination(FDQN) 透過 Google 表單上傳。

2 Cisco Packet Tracer ★☆☆☆☆ (15 points)

Resources

- **請下載 2021-mid.pkt，並完成設定後回傳給助教**
  (sha1sum: `366be1b2ef6096e3a16a732380bc9bc00d01a543 2021-mid.pkt`)
- Packet Tracer account: nasa.cisco.pt@yopmail.com (if needed)
- Packet Tracer password: Nasa.Cisco.Pt.217
- **這次的檔案不提供 Activity Check，會比較接近現實狀況，請自行檢查設定結果。(使用 host Desktop > Command Prompt 等)**
- **以下題目敘述中，`font` 這種字體代表一台在 2021-mid.pkt 的機器。**

2021-mid.pkt 網路架構：

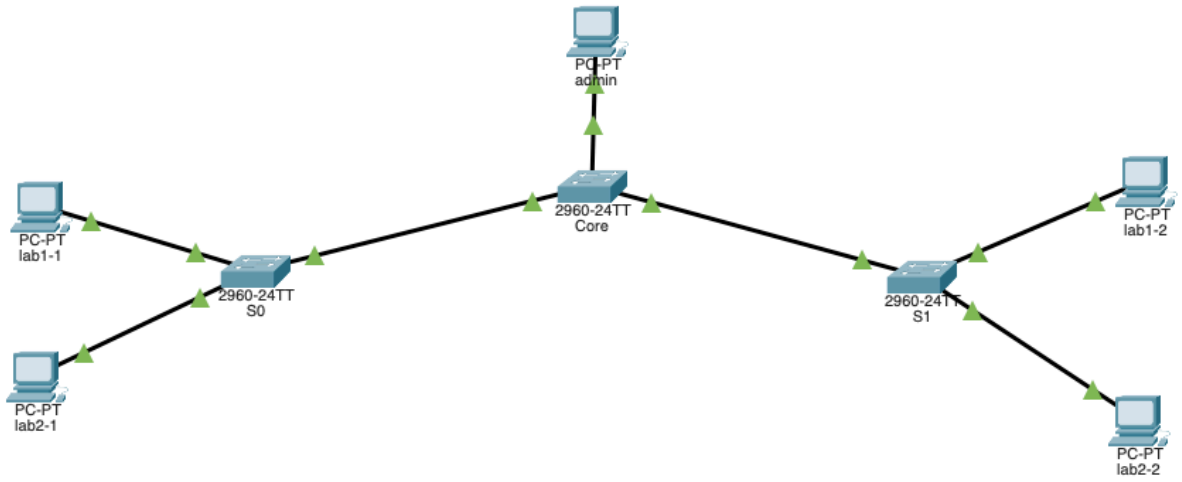


Figure 1: Problem2 網路架構

Host IP

- `admin`: 192.168.99.11/24
- `lab1-1`: 192.168.100.11/24
- `lab1-2`: 192.168.100.12/24
- `lab2-1`: 192.168.200.11/24
- `lab2-1`: 192.168.200.12/24

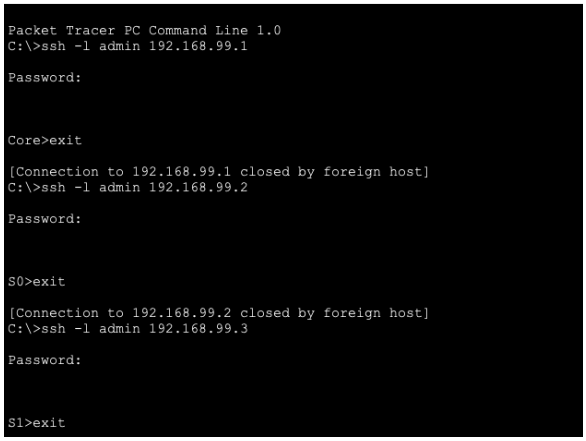
設定目標

1. 更改 switch 名字、enable 密碼 (3 points)
   - **請將 `Core` 的 hostname 設為 Core，密碼也設為 Core (不可明碼顯示在 running-config)。**

- 請將 `S0` 的 hostname 設為 S0，密碼也設為 S0（不可明碼顯示在 running-config）。
- 請將 `S1` 的 hostname 設為 S1，密碼也設為 S1（不可明碼顯示在 running-config）。

2. VLAN 設定 (4 points)
   - `admin` 在 VLAN 99，`lab1-1` 和 `lab1-2` 在 VLAN 100，`lab2-1` 和 `lab2-2` 在 VLAN 200。
   - 相同 VLAN 可以傳輸訊息，不同 VLAN 無法看見彼此的訊息。
3. VTY 設定 (8 points)
   - 在 `Core` 設定 vty 0 到 4 使得 host `admin` 可以用 username: admin/ password: admin，以 ssh -l admin 192.168.99.1 的方式連到 `Core`。不能使用 telnet 192.168.99.1 的方式連到 `Core`。（如下圖）
   - 在 `S0` 設定 vty 0 到 4 使得 host `admin` 可以用 username: admin/ password: admin，以 ssh -l admin 192.168.99.2 的方式連到 `S0`。不能使用 telnet 192.168.99.2 的方式連到 `S0`。（如下圖）
   - 在 `S1` 設定 vty 0 到 4 使得 host `admin` 可以用 username: admin/ password: admin，以 ssh -l admin 192.168.99.3 的方式連到 `S1`。不能使用 telnet 192.168.99.3 的方式連到 `S1`。（如下圖）
   - 以上密碼皆不可明碼顯示在 running-config
   - `lab1-1`、`lab1-2`、`lab2-1`、`lab2-2` 都不可以用任何方式 (ssh 或 telnet) 連到任何一台 switch。

   設定細節：
   - 設定 `Core`、`S0`、`S1` 的 domain name 為 nasa.com
   - 設定 `Core`、`S0`、`S1` 使用 ssh version 2
   - 設定 `Core`、`S0`、`S1` 的 ssh timeout 為 90 秒



(a) `admin` ssh 結果

(b) `admin` telnet 結果

Figure 2: Problem2 VTY 設定

Hint

- SSH 連線要先確認可以 ping 對吧:)
- 建議加 no shutdown 的指令。如果你覺得有下了正確的指令，但沒有任何作用的時候，請先稍等一下，packet tracer 的指令生效會比較慢。如果還是不行，有可能是 interface state down。
- 請繳交完成設定的 pkt 檔案。如果你覺得你的設定正確，是 packet tracer 的問題請附註後繳交。

Submission Format

- 請上傳設定完成後的 pkt 檔案。如果有其他問題，可以在最後一頁附註。

3 pfSense ★★☆☆☆ ~ ★★★★☆ (30 points)

Description

在這個大題，你需要建立一台 pfSense 虛擬機與一台內網的虛擬機 (以下用 Alpine 來代稱，但不限定使用 Alpine Linux)。下載 工作站的 mirror 可能會比官網快。請使用 204 電腦來完成此大題。

- pfSense 的虛擬機需要兩張網卡，一張設定 bridge (請不要使用 NAT)，另一張設定內部網路。在 pfSense 中，請在 LAN interface 設定 VLAN 5，並開啟 DHCP server。
- Alpine 的虛擬機需要一張網卡，設定為內部網路 (與 pfSense 的內部網路相同)。Alpine 需要透過 pfSense 的 DHCP 拿到 VLAN 5 的 IP 位址。

我們強烈建議你將 pfSense 的 admin 密碼改掉，以避免未經許可的第三方登入。再次提醒，請使用 204 電腦來完成此大題。

Tasks

1. (3 points) 請在 Alpine 上執行 `ip a; ip r` 指令並上傳螢幕截圖。請注意，除了 loopback 網路介面之外，Alpine 只有 VLAN 5 的介面能擁有 IP 位址。
2. (5 points) 請設定 pfSense 的 DNS Resolver 服務，讓 `cloudflare.dns` 這個 domain 對應到 `1.1.1.1` 和 `1.0.0.1` 這兩個 IP 位址。請在 Alpine 上執行 `nslookup cloudflare.dns` 並上傳螢幕截圖。
3. (7 points) 請設定 pfSense，只讓你的 host machine 和助教的電腦 (192.168.204.xxx，確切 IP 將寫在白板上) 可以 ssh/https 到 pfSense。在 pfSense 的 WAN 介面，請不要設定「allow all traffic」這種規則，請只設定必要的規則。請提供 pfSense WAN 的 IP，並提供 pfSense 的 admin 帳號的密碼。助教將會登入你的 pfSense 來檢查你的設定是否合理。
   - 如助教只能使用 ssh 或 https 其中一種方式連上 pfSense，則只能拿到 2 分。
   - 如助教可使用 ssh/https 連上 pfSense，但無法登入或登入後發現防火牆規則不合理，則只能拿到 4 分。
4. (7 points) 在 這個連結 有許多的 URL，請只允許這裡面的 URL 可以 ping 到 pfSense 的 WAN IP。請提供 pfSense WAN 的 IP，助教會嘗試從不同的 IP 位址 ping 你的 pfSense 來檢查你的設定是否合理。
5. (8 points) 請在 Alpine 上設定 openssh server，並對 pfsense 做必要的設定，使助教可以從第 2. 小題的 IP 位址 (192.168.204.xxx) ssh 到你的 Alpine。請提供 ssh 指令和密碼，讓助教可以使用該指令與密碼來 ssh 到 Alpine。ssh 指令的格式為 `ssh -p <port> <user>@<ip>`。助教會用 ssh 登入你的 Alpine，然後在你的 Alpine 上執行 `traceroute`，並檢查輸出是否合理。
   - 如助教可 ssh 到 Alpine，但無法登入或登入後發現防火牆規則不合理，則只能拿到 6 分。

Hint

- 請使用 204 電腦來完成此大題。
- 正確地使用 `setup-alpine` 可以避免資料在 Alpine Linux 重開機時遺失。
- Lab 投影片會是你的好朋友。
- pfSense 官方文件 會是你的好朋友。
- pfSense 的 Aliases 設定中有很多不同的選項...
- 什麼是 port forwarding?

Submission

- Problem 3.1: 請透過 Google Forms 上傳截圖。
- Problem 3.2: 請透過 Google Forms 上傳截圖。
- Problem 3.3: 請透過 Google Forms 提供 WAN 的 IP、帳號、密碼。
- Problem 3.4: 請透過 Google Forms 提供 WAN 的 IP。
- Problem 3.5: 請透過 Google Forms 提供指令、密碼。

4 By the way, I use Arch ★★☆☆☆ ~ ★★★★☆ (25 points)

Arch Linux 是一個廣為人知的 linux distribution 之一，而它的安裝介面只有一個純文字的 shell 介面，更沒有自動化的安裝功能（雖然在今年愚人節所推出的最新版 arch linux ISO 新增了個自動安裝的 archinstall 指令...），因此本題將要求你寫一個 arch linux 的自動化安裝腳本。在本題中一共有三個小題，請寫一個 shell script 腳本，在一用給定的 archiso 開機一台新的 VM 後，執行該腳本就能完成題目要求。在本題的 shell script 中，你能使用任何該 archiso 中有提供的指令與程式，但是不能使用需要額外下載的套件或指令。
請使用 `archlinux-2021.04.18-x86_64.iso` 這個 archiso 映像檔來安裝你的系統。
(sha1sum: `9b7937656ea2db3044e4096348d2ed006a9b6bda`)

4.1 Preparation (10 points)

請建立一個 VM 包含三個 4G 的硬碟 (/dev/sd[abc])，並將硬碟 (以 GPT 格式 (gdisk 指令)) 分割成：

- /dev/sda1: 1M bios boot partition
- /dev/sda2: 1G
- /dev/sda3: 3G (所有剩餘空間)
- /dev/sdb: 不用切割
- /dev/sdc: 不用切割

並將 /dev/sda3 與 /dev/sdb 合組成一個 vg (name: nasa) 並再分出兩個 lv ((name, size): (root, 5G), (home, 剩餘空間 (約 2G)))，然後再將 nasa-root 這個 lv 與 /dev/sdc 用 mdadm 指令合組成一個 raid0 的磁碟陣列 (name: md0)，並依照下表作格式化與掛載：

- `/dev/sda1`: bios boot partition
- `/dev/sda2`: [SWAP]
- `/dev/md0`: ext4 /mnt
- `/dev/nasa/home`: xfs /mnt/root

若輸入 `lsblk; df -h; lvs; vgs`; 確認硬碟切割正確且都有掛載成功，且 lvm 容量/剩餘容量 (0) 皆正確。

4.2 Installation (5 points)

請接續寫你的腳本，使其能支援自動安裝好 arch linux 系統。建議參考 arch linux 的官方文件 Installation Guide 來了解安裝步驟。其中 boot loader 請選擇安裝 grub 。本題在成功安裝後，只要你能在重新開機時順利啟動 grub 並進入 arch linux 系統即可，就算系統本身發生意外無法順利開啟也沒關係。

4.3 Boot the System (10 points)

請嘗試成功讓系統可以順利開機。請注意你所做的修改也需要加進 shell script 中使其能自動完成，也就是將你的 shell script 放到一台新的機器中也可以成功安裝一台可以開機的 arch linux。

Submission

- 請透過 Google Forms 上傳你的 script，若有需要助教可能會到座位上觀看同學 demo，並以 demo 結果為主。

5 Fediverse ★★★☆☆ (20 points)

> The Fediverse (a portmanteau of "federation" and "universe") is an ensemble of federated (i.e. interconnected) servers that are used for web publishing (i.e. social networking, microblogging, blogging, or websites) and file hosting, but which, while independently hosted, can communicate with each other. On different servers (instances), users can create so-called identities. These identities are able to communicate over the boundaries of the instances because the software running on the servers supports one or more communication protocols which follow an open standard. As an identity on the fediverse, users are able to post text and other media, or to follow posts by other identities.
>
> – Wikipedia

你有聽過 Fediverse 嗎？在 Fediverse 中，最有名的應用大概是基於 ActivityPub 協定的 Mastodon 吧。Mastodon，中文有時稱為長毛象，是一個去中心化的，類似 Twitter 的 microblogging 服務。而除了 Mastodon 以外，也存在許多其他基於 ActivityPub 的應用，像是 Pleorma (另一個 microblogging 服務) 或是 Peertube (影音平臺)。

你發現你的偶像 The December Drive (為了簡潔起見，以下就簡稱為 TDD 吧！) 經常在 ActivityPub 上發文。(並沒有。) 你想要追蹤 TDD 的動向，但是直接 follow 他們或是用 RSS 的話對你來說太簡單了。(不要瞎掰好嗎？) 因此，你便想寫個 shell script 來爬下 TDD 的貼文。

確切地說，你需要實作一支 shell script，使其能以以下的方式執行：

```
./fedi.sh [-n NUM_POSTS] [-a] ACCOUNT_ID
```

其中 ACCOUNT_ID 是你想要印出的帳號，而 -n 是個可選的參數，表示你需要輸出最新的 NUM_POSTS 則訊息。(預設為 10 篇。)

至於 -a 這個選項，表示需不需要印出 type 為 Announce 的訊息。(訊息類別會在下文中闡述。) 如果使用者有指定這個選項，你需要同時印出 Announce 和 Create 兩種訊息。反之，你只需要印出 Create 類的訊息。

注意選項可以以任意順序出現。並且為了方便起見，可以假設 argument 是合法的，也不需要處理和伺服器溝通可能出現的錯誤 (error)。

爬取 ActivityPub 的方式可以參考 https://www.gkbrk.com/2018/06/fetching-activitypub-feeds/。

腳本的輸出格式定義如下：

```
DATE
CONTENT
===
DATE
CONTENT
===
...
```

換言之，訊息間應以一行 === (三個 ASCII 0x3D) 分隔，而每則訊息的格式是一行日期，接著是訊息的內容。

日期的格式應為 RFC 5322，並以本機的時區輸出，如 Mon, 14 Aug 2006 02:34:56 +0800。

你應該支援兩種訊息格式：type 為 Create 的貼文 (包含 Replies) 和 type 為 Announce 的轉推。對於前者，你應該使用 w3m -dump 將貼文的 HTML 內容轉為純文字。(這裡假設貼文的 object type 均為 Note，也就是可以依循上面連結的文章讀取 content 這個欄位。) 而對於後者，你只要輸出一行轉貼的貼文連結。

Hints

以下的工具可能會有幫助：

- curl
- date
- jq
- w3m

Sample Input/Output

注意以下僅作為格式的參考。實際的輸出內容有可能會因為新發文而異。

```
./fedi.sh -a -n 3 @hntooter@mastodon.social
```

範例輸出如下：

```
Wed, 14 Apr 2021 22:07:36 +0800
```

```
Bernie Madoff, mastermind of the nation's biggest investment fraud, dies at 82
https://www.cnbc.com/2021/04/14/bernie-mad
off-dies-mastermind-of-the-nations-biggest-investment-fraud-was-82.html

(submitted by daolf)

===
Wed, 14 Apr 2021 21:54:01 +0800
1986 Sat Nav
https://www.bbc.co.uk/archive/sat-nav-cd-top-gear/zkj3hbk

(submitted by AlfGalf2)

===
Wed, 14 Apr 2021 21:46:13 +0800
Latino Programming Language
https://www.lenguajelatino.org

(submitted by chespinoza)
```

該使用者的訊息可以使用瀏覽器在 https://mastodon.social/@hntooter 看到。另外，以下也提供幾個其他的使用者作為測試用途：

- @freemo@qoto.org
- @lowvolume@iddqd.social
- @catpostingbot@pawoo.net

Submission

- 請透過 Google Forms 上傳你的 script，助教會為你評分後寫在 COMMENTS 欄。

6 ZIP Checker (25 points)

請用 shell script 實作一個 zip checker，檢查輸入的 zip 檔案是否符合輸入的格式規範。執行時第一個參數會是 zip 檔案、第二個會是一個描述格式規範的 JSON 檔。也就是說，如果你的 script 叫做 `checker.sh` ，那麼它會以如下的方式被執行：

```
./checker.sh input.zip format.json
```

你的程式必須判斷 `input.zip` 是否符合 `format.json` 的要求，如果是的話請回傳 0、否則回傳一個非 0 的結果。`format.json` 的格式可以用下面的遞迴關係來描述[1]：

```
<spec> ::= { <directory-spec> }

<directory-spec> ::=
    "<directory-name>": [ <file-specs> ] |
    "<directory-name>": { <directory-fields> }
```

[1] 請參考 https://en.wikipedia.org/wiki/Backus%E2%80%93Naur_form

```
<directory-fields> ::=
    <directory-fields>, <directory-spec-or-files> |
    <empty>

<directory-spec-or-files> ::=
    <directory-spec> |
    "_files": [ <file-specs> ]

<file-specs> ::=
    <file-specs>, "<file-name>" |
    <empty>
```

其中 `file-name` 以及 `directory-name` 皆是僅包含大小寫英文字母（`[a-zA-Z]`）、數字（`[0-9]`）以及點（`.`）的 ASCII 字串，而 `<empty>` 則是代表空字串。以下是一個合法的格式規範檔：

```
{
    "b09902000": {
        "code": {
            "sh": ["p1.sh", "p2.sh"],
            "cpp": ["p3.cpp"],
            "py": [],
            "_files": ["Makefile"]
        },
        "report": ["report.pdf"],
        "_files": ["^.gitignore", "README.md"]
    }
}
```

簡單來說，對於給定的 `format.json` ，一個輸入的 zip 檔案 `input.zip` 是符合規則的若下面的條件皆滿足：

- `input.zip` 為單一資料夾的壓縮檔。也就是說，所有的檔案以及子資料夾都必須直接或間接的被最上層的資料夾包含。這樣的 zip 檔案通常可以用 `zip -r input.zip directory` 這樣的指令製造。
- 從 `input.zip` 中的最上層資料夾 D 開始，每個子資料夾都必須遞迴的滿足 `format.json` 中相對應的規則。首先，D 的名字必須與 `format.json` 中最外層唯一的 key K（在上述範例中是 `b09902000`）完全相符。再來，考慮以下的幾個情形：
  - K 對應到的 field 是一個 array [2]：此時 D 必須不包含任何子資料夾，且 D 包含的檔案必須滿足該 array 的規範。這個部分稍後會作說明。
  - K 對應到的 field 是一個 object：此時考慮該 object 中的每一個 key K'。若 K' 為 `"_files"`，則 K' 對應到的 field 一定是一個 array，此時代表 D 中直接包含的檔案必須滿足該 array 的規範。否則，K' 以及其對應的 field F 描述了 D 的一個子資料夾，代表 D 必須有一個名為 K' 的子資料夾，且該子資料夾必須遞迴地滿足 F 的限制。

  注意到 D 中的每一個子資料夾都必須在 `format.json` 中有對應的描述，這部分詳細可以參考後面的範例。

[2] 關於 JSON 檔中 array 與 object 的介紹，可以參考 https://en.wikipedia.org/wiki/JSON

對於一個資料夾 D 以及一個陣列（例如一個 `"_files"` 所對應的陣列），如果對於陣列裡的每個字串 `f`，下面的條件都滿足的話：

- 如果 `f = ^ g`，則 D 不包含檔名為 `g` 的檔案
- 不然，代表 D 包含檔名為 `f` 的檔案

那麼 D 就算是符合這個檔案陣列的規定。同樣注意到檔案與資料夾規則的不同：對於檔案來說，沒有被描述到的檔案沒有任何的規範，而對於資料夾來說，若存在沒有被描述到的子資料夾，則視為格式不正確。

為了方便說明，以下舉一個較簡單的 `format.json` 範例：

```
{
    "b09902000": {
        "subdir1": ["test.txt"],
        "subdir2": {
            "test": ["test2.txt"],
            "_files": ["ok"]
        },
        "_files": ["^forbidden"]
    }
}
```

以下為一些合法與不合法的 zip 檔案範例：

- 合法：

```
b09902000/
b09902000/subdir1/
b09902000/subdir1/test.txt
b09902000/subdir2/
b09902000/subdir2/test/
b09902000/subdir2/test/test2.txt
b09902000/subdir2/ok
```

- 不合法（包含 `forbidden`）：

```
b09902000/
b09902000/subdir1/
b09902000/subdir1/test.txt
b09902000/subdir2/
b09902000/subdir2/test/
b09902000/subdir2/test/test2.txt
b09902000/subdir2/ok
b09902000/forbidden
```

- 合法（就算包含 `ok2`）：

```
b09902000/
b09902000/subdir1/
b09902000/subdir1/test.txt
b09902000/subdir2/
b09902000/subdir2/test/
b09902000/subdir2/test/test2.txt
b09902000/subdir2/ok
b09902000/subdir2/ok2
```

- 不合法（包含 `subdir3`）：

```
b09902000/
b09902000/subdir1/
b09902000/subdir1/test.txt
b09902000/subdir2/
b09902000/subdir2/test/
b09902000/subdir2/test/test2.txt
b09902000/subdir2/ok
b09902000/subdir3/
b09902000/subdir3/ok
```

6.1 ★★☆☆☆ (10 points)

在這個子題中，輸入的 format.json 中最上層唯一一個 field 一定是 array。也就是說，只需要處理一層的資料夾並檢查它所包含的檔案即可。更具體地，只會出現形式同下的 format.json：

```
{
    "b09902000": ["file1.txt", "file2.txt", "^file3.txt"]
}
```

6.2 ★★★☆☆ (12 points)

放寬上一個子題的限制，可允許任意層數的資料夾。

6.3 ★★★★☆ (3 points)

上面的定義中 directory-name 以及 file-name 只可以為簡單的 ASCII 字串，在這個子題中我們放寬這個限制，容許 regular expression (PCRE) 的出現來達到更全面的格式檢查。具體來說，現在除了 ASCII 字串以外，每當在 directory-name 以及 file-name 中遇到兩個方括號 ([[ ) 還需要處理下列兩種可能：

- [[NAME:REGEX]]：此為 regular expression 的定義，代表從目前讀到的位置開始必須匹配一個符合 REGEX 限制的子字串，並且將這個子字串的值存進 NAME 這個變數中。保證整個 format.json 中只會有一個 NAME 的定義、且 file-name 中不會出現這種情況。NAME 只會包含大小寫英文字母以及數字。
- [[NAME]]：代表將這組方括號替換為 NAME 這個變數所代表的字串。保證 NAME 這個變數會在目前位置的上層資料夾們中被定義過。

為了避免歧義，所有的 regular expression 都不會包含連續的方括號 ([[或]]) 以及冒號 (:)。以下是一個範例的 format.json：

```
{
    "[[id:b[0-9]{8}]]": {
        "report": ["[[id]].pdf"],
    }
}
```

在這個 format.json 的規範下，以下的 zip 檔案是合法的：

```
b09902000/
b09902000/report/
b09902000/report/b09902000.pdf
```

而以下的則皆不是：

- id 為 b09902001 而非 b09902002：

```
b09902001/
b09902001/report/
b09902001/report/b09902002.pdf
```

- `b0990200` 不符合 regular expression `b[0-9]{8}`：

```
b0990200/
b0990200/report/
b0990200/report/b0990200.pdf
```

Notes

- 可以使用 `jq` [3] 這個工具來處理輸入的 `JSON` 檔。
- 請在**不將輸入完全解壓縮**的情況下完成檢查。
- 為了避免不必要的麻煩，輸入的 zip 壓縮檔中所有資料夾以及檔案名稱一樣只包含大小寫英文、數字以及點。
- 對於 Subtask 3 ，假設某個資料夾或是檔案的名稱含有 REGEX 的定義，那麼在所有同層的資料夾以及檔案中，至多只有一個符合該 REGEX 。

Hints

- 對應到 Notes 的第二項，要如何在沒有完全解壓縮的情況下看到 zip 檔的結構？

Submission

- 請透過 Google Forms 上傳你的 script，並勾選要上傳的小題數，助教為你評分後寫在 COMMENTS 欄。

7 Partition and Filesystem (CTF) ★★☆☆☆ ~ ★★★★☆ (22 points)

Resources

- VM Image : `NasaMidtermArch.ova` (sha1sum: `86680031d466b49bf2eb501636df7d8cad333f6d`)
- OS : Arch Linux
- Username: `root`
- Password: `nasa`
- Flag format: `FLAG{54mpl3_fl46}`
- 你可以對 VM 做任何修改/新增硬體設備等等，只要能拿到 flag 或達成要求即可
- 如果你使用 VirtualBox 開啟的話，請記得要在 VM 的 OS 選擇 Arch Linux (64-bit) 才能正常開機

[3] https://stedolan.github.io/jq/

7.1 Task 1 - Snapshot and Backup (8 points) ★★★☆☆

在 VM 中家目錄底下的 data 資料夾 (/root/data) 是一個 LVM 分區，裡面有一個執行檔 check 以及一個資料檔 secret ，其中 check 程式會讀取 secret 的內容，必須要成功通過驗證才會將秘密解密並告訴你。為了避免未來對 secret 的修改產生問題，該 LV 在每天的一開始都會建立一個 snapshot ，並在每天的結束前會將 snapshot 分區的資料整個打包壓縮出來存放在 /root/backup/$id.snap 檔案中 (其中 $id 是他的工作日編號，從 0 開始計)，然後再移除該 snapshot 分區。某天突然發現 secret 有問題無法成功通過驗證，因此想將 secret 還原至 32 天前的狀態 (倒退 32 個版本)，請利用每日備份的 snapshot 檔案將 secret 還原並成功通過驗證。

7.2 Task 2 - Shrinking LVM (8 points) ★★★★☆

在這台 VM 中原本一共有兩顆各 4GB 的硬碟 (GPT 格式) ，現在想將第一顆硬碟 (/dev/sda) 尾端清出 512MB 的空間來做為 swap 使用，不幸的是，目前硬碟已經沒有多餘的空間了，因此你打算把最尾端的 /dev/sda3 分區 (用於 LVM) 縮減 512MB。然而，該 VG (name: nasa) 的所有空間也都正在使用中，因此你要先將該 VG 中的一個 LV (name: home) 縮減 512MB ，如此一來該 VG 就會有足夠的空間可以將前述硬碟分區縮減，並建立新的 512MB 分區 `/dev/sda4` 作為 swap 分區。(由於一些空間消耗，只要你的 swap 有至少 500MB 即可。) 請在做完上述動作後要確保可以該 VM 可以成功重新開機，並且 LV: nasa-docs 和新設的 swap 分區都會自動 mount 上去。此外，docs 中的檔案也應該維持不變。若你能達成上述目標，你將會在重新開機時拿到本題的 flag 。其中 task2 這個執行檔可以用來檢查當前的狀態是否滿足要求 (但你需要在重新開機後仍能滿足要求才會拿到 flag)。

7.3 Task 3 - Luks (6 points) ★★☆☆☆

為了增加資料的安全性，現在打算將 mount 在 /root 資料夾的 LV 加密 (將加密的裝置名稱取作 encryptedhome ，密碼則隨意)，請使用 Luks 套件將其加密並且會在開機自動 mount 。請在做完此動作後將 VM 重開確保會看到在開機時 luks 會要求輸入密碼的畫面。此外，你可能會需要在建立 luks 分區前將該分區底下的所有資料備份，並於建立完重新開機後還原回去。

Submission

- Problem 7.1: 透過 Google Forms 上傳 flag。
- Problem 7.2: 透過 Google Forms 上傳 flag。
- Problem 7.3: 透過 Google Forms 發 Demo Request，助教會來檢查。

8 Docker ★★☆☆☆ (18 points)

我們公司使用 Go 架設網站，現在想要一些 Go 的 Docker Image 讓同事方便開發、佈署，你可以幫我們嗎？

你可能會用到的 Go 知識

1. 直接執行 (含編譯)：go run main.go
2. 編譯：go build main.go
3. 確認版本：go version

8.1 建立 Docker Image （2 分）

- 請使用 Alpine 當 Base OS。
- 我們想要內建 curl, wget, vim, git 以方便在 container 裏面 debug。
- 請安裝 go 1.13 版本。
- 開啟 container 時，它會執行 /src/main.go。
- 不能把程式碼放在 image 裏面，請用 volume 的方式外掛

請用這個 main.go 來測試 docker image。

8.2 優化 Docker Image 的大小（4 分）

做得好！但我們希望 docker image 可以再更小一點，你可以幫我們把它的大小減少到不超過 30 MB 嗎？(不需要內建 curl, wget, vim, git 了，請用 tag:deploy 和之前 build 的 image 分開)

8.3 Debug Docker （4 分）

我們發現離職的同事很早就寫好了 Docker Image，並以.tar 檔儲存，但我們 pull 下來發現它有一些問題，而且 Dockerfile 已經遺失了，可以協助我們找出 bug 嗎？
請下載這個 image，找出 bug 並以這個 image 為 base，提供改善版本的 Dockerfile。

8.4 自架 Registry （5 分）

請架一個 docker registry，將第 1 小題 and/or 第 2、3 小題的 image push 到該 registry。請確定其他電腦或 VM 也能 pull/push 該 registry 的 image。(只有測試 localhost 並不能拿到分數)

8.5 Docker-compose (3 分)

請寫一個 docker-compose.yml 將第 1 小題和第 2 小題的 image 分別佈署在 8000 與 8080 port，程式碼仍使用外掛 volume 的方式，source 必須來自第 4 小題自己架設的 registry 且非 localhost，否則只得 1 分。

Submission

- 請透過 Google Forms 發 demo request，並勾選你要 demo 的小題 (可以一次 demo 多個)。

9 Docker Network ★★★★☆ (28 points)

在這個題目中，你會有一個 VM，VM 當中有一些 docker 服務，你需要把 VM 的網路修好，並且將裡面的 docker 服務升起來。

Resources

- VM 需要至少 10GB 的空間，Memory 1024MB，以及 1 CPU。
- 請使用 VMware 開啟 ova 檔，避免 Virtual box 發生無法預期的錯誤。

- 你可以在本機端使用 ssh 進入 VM 當中，這會讓你更好地操作以下題目。
- user: nasamid, password: nasamid(有 sudo 權限)
- docker image 中，nasamidweb 是一個網頁服務，nasamiddb 是這個 web 服務的 db。
- ova 的 shasum: 55b664427706d90efcf49dcefd1e1b67c2351e06

Subtasks & Scores

(1) (4 pts) 請在 VMware 上將 ova 檔安裝好，然後修好 VM 的網路，修好以後，請截圖 `ping 8.8.8.8` 的結果。

(2) (8 pts) VM 當中有兩個 docker image：nasamiddb, nasamidweb，請分別執行以下動作：

  (a) (4 pts) 當前的 nasamiddb 是壞掉的狀態，請將它重新升起並截圖執行 `sudo docker container ls -a` 的結果。
    * hint: 升起 image 時請記得讓他跑在背景，若沒跑在背景可能會當機。
    * hint: 以上可以用一個指令完成。

  (b) (4 pts) 請將 nasamidweb 的 image 升起來，nasamidweb 中會跑一個服務在 docker 的 3000 port 上，請讓 VM 的 3000 port 能 access 到這個服務，成功後請在 VM 上執行 `curl 127.0.0.1:3000` 並截圖結果。
    * hint: 以上可以用一個指令完成。

(3) (6 pts) nasamidweb 以及 nasamiddb 目前的網路目前並不互通，請建立兩者之間的網路，建好以後，請進到 nasamidweb 裡面執行 `ping nasamiddb`，進到 nasamiddb 裡面執行 `ping nasamidweb`，並截圖結果。

(4) (10 pts) 請在本機端（你的筆電或 204 桌機）上，打開 nasamidweb 的服務，你會發現那個頁面要你輸入 db hostname、username、password。

  (a) (3 pts) 已知有一個 username=nasamid、password=nasapass，請將此 username、password 以及你的 db hostname（不用加上 port）輸入在網頁的表單當中，若成功連結的話你會看到一句話，請將那句話截圖上傳。

  (b) (7 pts) 你能找到 nasamiddb 的 root 密碼嗎？請在網頁的表單中輸入 username=root、password= 你找到的密碼、以及你的 db hostname，若你找到的是正確的密碼，你會獲得另外一句話，請將這句話截圖上傳。

Submission

- Problem 9.1: 請透過 Google Forms 上傳螢幕截圖。
- Problem 9.2 (a): 請透過 Google Forms 上傳螢幕截圖。
- Problem 9.2 (b): 請透過 Google Forms 上傳螢幕截圖。
- Problem 9.3: 請透過 Google Forms 上傳螢幕截圖。
- Problem 9.4 (a): 請透過 Google Forms 上傳螢幕截圖。
- Problem 9.4 (b): 請透過 Google Forms 上傳螢幕截圖。