NASA2022 Homework0

Network Administration

討論對象：
b05504066 李旻翰
b09505014 王聖文

1. True / False

1. **False**，雖然4G、5G有速度上的差異，但G所指的是generation的意思。

   (Ref: https://zh.wikipedia.org/zh-tw/5G)

2. **False**，每張網路介面卡都有獨一無二的識別碼，這個識別碼是由六組16進位數字組成的physical address，也稱MAC address，如果一個裝置有多個網路介面卡，則可能有多個MAC address。

   (Ref: https://tsowen.pixnet.net/blog/post/272632658)

3. **True**，因為NAT server的作用便是將不同的內部網路ip轉換成一個ip，假設私有網路的電腦要傳送封包到網際網路，NAT server會將封包表頭的ip位址改成一個外部網路的網址。(如果有多個公有ip就Flase，因為有可能不一樣)

   (Ref: https://www.stockfeel.com.tw/nat/)

4. **False**，VPN會多包一個外部封包於原資料外，所以所在位置會以連上VPN伺服器的IP位址來顯示。

   (Ref: https://nordvpn.com/zh-tw/what-is-a-vpn/)

5. **False**。Intranet的基本思想是在企業內部網路上採用TCP/IP協議，利用Internet的Web概念與技術為標準平臺，通過防火牆把內部網路和Internet隔開。因此可知Intranet內部是使用同個protocol，不需要使用gateway，因為gateway是用來連接兩種不同協定網路的裝置。

   (Ref: https://www.itread01.com/p/1421476.html)

6. **False**，用戶端電腦會向DNS伺服器查詢域造訪之網站的IP位址，接著DNS伺服器會查詢自己的記憶體，如果曾經被詢問過就會有記錄，如果第一次被詢問則沒有記錄，於是轉向根網域 DNS 伺服器查詢。所以並不會查詢不到結果就回傳"no corresponding domain"。

   (Ref: https://www.stockfeel.com.tw/dns/)

7. **False**。DHCP server是用來節省子網路中IP位址使用量的指揮中心，負責動態分配 IP 位址，當網路中有任何一台電腦要連線時，才向 DHCP 伺服器要求一個 IP 位址；NAT server是負責把將所有內部網路的IP改為一個真實IP的工具；DNS server則是將網域轉換為IP位址，若已知IP則不必透過DNS server查找。我們不需要他們也可以連線，只要我們有伺服器的IP位址即可。

   (Ref: https://www.stockfeel.com.tw/dhcp/)

8. **False**。DHCP server會發放一個可用的IP位址給客戶端，這個IP位址是有使用期限的，然而在這個期限內，客戶端都是使用這個IP位址向外連線，無需再透過DHCP server，直到期限到期，才需自行向DHCP server要求。

   (Ref: https://www.netadmin.com.tw/netadmin/zh-tw/technology/DD0EC17A9FF84D1D9AA6FEE7140B3803?page=2)

9. **True**，HTTP的資料傳遞過程皆是明文，所以收到封包的人皆可以看到其內容。

   (Ref: https://tw.alphacamp.co/blog/http-https-difference)

10. **True**，由於目前仍沒有找到一個多項式時間的演算法來因數分解，選擇短的密鑰可能可以強力破解，但只要長度越長，所需破解的成本會越高也越不可能破解。

    (Ref: https://zh.wikipedia.org/zh-tw/RSA加密演算法)

2. Short Answer

1. (a) **MAC address**: 每一個網路介面卡上街有一個獨一無二的識別碼，這個識別碼是由六組十六進位數字所組成的物理位置，也稱作MAC address。

   (b) **Switch**: 交換器是一個負責Network bridging的網路硬體設備，會讀取網路介面卡之MAC address將資量準確送達目的地。

   (c) **broadcast storm**: 網路廣播風暴，其起因為廣播與多波訊號之累積，佔用大量的網路頻寬導致正常的網路訊號無法流通。

   (Ref: https://zh.wikipedia.org/zh-tw/廣播風暴

   https://www.tp-link.com/tw/blog/119/交換器是什麼-3種常見的交換器接法-應用場景及功能介紹/)

2. subnet mask用來分割子網路與區分哪些ip是同個網段。我們首先計算出subnet mask，23代表subnet mask所佔位元數，因此可得：

$$\begin{aligned} &11111111.11111111.11111110.00000000 \\ =\ &255.255.254.0 \end{aligned}$$

接下來只要將每個ip位址和255.255.254.0做`&`位元運算計算即可得知是否在同個子網路：

$$\begin{aligned} 192.168.0.1\ \&\ 255.255.254.0 &= 192.168.0.0 \\ (a)\ 192.167.0.1\ \&\ 255.255.254.0 &= 192.167.0.0 \\ (b)\ 192.168.0.0\ \&\ 255.255.254.0 &= 192.168.0.0 \\ (c)\ 192.168.1.0\ \&\ 255.255.254.0 &= 192.168.0.0 \\ (d)\ 193.168.0.1\ \&\ 255.255.254.0 &= 193.168.0.0 \end{aligned}$$

可以發現(b)、(c)和題目給的IP在同個子網路。

(Ref: https://dotblogs.com.tw/chris0920/2010/11/02/18730)

3. (1) Five layers分別為：Application Layer/ Transport Layer/ Network Layer/ Link Layer/ Physical Layer。

   (2)

   - **Application layer**是負責應用層負責運行在兩個不同end system上的應用程式之間的通信，這些應用程式包括browser、email client等等。
   - **Transport layer**在傳輸端是負責搜集application layer的message，然後送到network layer去透過network傳輸；在接收端則是接收network layer傳來的message，然後送到相關的socket讓application layer 得以access那些message。
   - **Network layer**負責透過網路將一個系統欲傳輸的資料傳至另一個系統。
   - **Link layer**負責一個裝置與其鄰居之間的溝通，在一個packet傳輸的過程中，兩個不同的end system之間會有許多的中介裝置，這些中介裝置可能是router、switch或其他電腦，兩個鄰居裝置之間的溝通就是由link layer負責。
   - **Physical layer**負責將資料架構分解成bits，讓它們轉化成一個可以在physical communication line上傳輸的形式。

   (3)

   - **Application layer**: SMTP、FTP、HTTP皆是應用層提供的protocol。
   - **Transport layer**: 會在application layer傳來的message上加上一個header，裡面會有transport layer的一些資訊，TCP和UDP皆是傳輸層提供的protocol。

- **Network layer**: 提供Internet Protocol，這個protocol是使用IP address來辨認連接到網路的不同系統。
- **Link layer**: 最常被應用在network adapter/network interface card (NIC)，乙太網、Wi-Fi皆與link layer有關。
- **Physical layer**: 實體層設備有網路線、網路卡等等。

(Ref: https://www.educative.io/edpresso/what-is-the-five-layer-internet-protocol-stack)

4. (1) **TCP**是通訊控制協定，**UDP**則是用戶資料包協定。TCP與UDP皆屬transport layer的通訊協定，它們能確保網際網路資料傳輸的快速和完整性。它們都會將資料分隔成更小的單位(也就是封包)進行傳輸。

   (2)

   - **TCP**: 為每一個封包發放一個唯一的識別碼和序號。當接收端收到封包且順序正確，會向發送端傳送確認信號，如果封包遺失或順序錯誤，接收端會保持沈默，這就表示發送端需要重新傳送封包。這樣使TCP可以有效的流量控制和解決資料壅塞的問題，但同時也因為發送端與接收端之間有大量通訊導致建立連線與交換訊號需要更多時間。
   - **UDP**: 不需唯一的識別碼和序號即能完成工作，以串流之方式傳遞資料，發送端會不斷的發送封包資料而不去等接收端確認信號，這導致UDP幾乎沒有錯誤修的功能，也不會理會封包的遺失，因此易出錯，但傳輸速度會比TCP快。

   (3)

   - 選擇**TCP**而不選UDP的情況：需要可靠的傳輸服務，例如電子郵件、網頁瀏覽、檔案傳輸。
   - 選擇**UDP**而不選TCP的情況： 需要即時的服務，例如串流媒體、網路電話、網路遊戲。

   (Ref: https://nordvpn.com/zh-tw/blog/tcp-udp-bijiao/)

5. (a) **DoS**: 透過傳送大量且不合法的request到網路伺服器，因為伺服器需要應對這些大量的請求導致真正需要被處理的請求無法被處理。防範方法包括：儘早使用安全性修補程式、定期使用IDS(入侵偵測系統)以及IPS(入侵防禦系統)檢查、過濾localhost或是其他私有以及無法路由的IP address、阻擋所有進入自家網路的ICMP流量、關閉所有非必需的TCP\UDP服務。

   (Ref: https://ithelp.ithome.com.tw/articles/10277033)

(b) **DDoS**: 建立殭屍網路，使殭屍網路中的每個機器人都向目標的IP位址發送請求，使得該伺服器或網路不堪重負、對正常的流量拒絕服務。防範方法包括：讓防火牆和路由器設置為丟棄傳入的ICMP封包或阻擋來自網路外部的DNS回應、準備足夠的頻寬來處理可能由網路攻擊所引起的流量湧升。

(Ref: https://www.digicentre.com.tw/industry_detail.php?id=56)

(c) **Man in the middle attacks**: 中間人攻擊是指攻擊者在和正要通訊的兩端分別建立聯絡，使得兩端都認為他們正透過一個私密的連線與對方溝通，但整個談話皆被攻擊者掌握。防範方法包括：建立相互驗證的機制、延遲測試(例如透過複雜的hash function進行計算以造成延遲，如果雙方在正常溝通的情況下各需花20秒計算，當整個通訊過程花了60秒以上就代表有中間人)、第二通道校驗。

(Ref: https://zh.wikipedia.org/zh-tw/中间人攻击)

3. Command Line Utilities

1. (a) 140.112.8.116

   (b) 140.112.30.26

   透過`nslookup <domain name>`即可查到對應的IP address。

   (c) linux1.csie.ntu.edu.tw

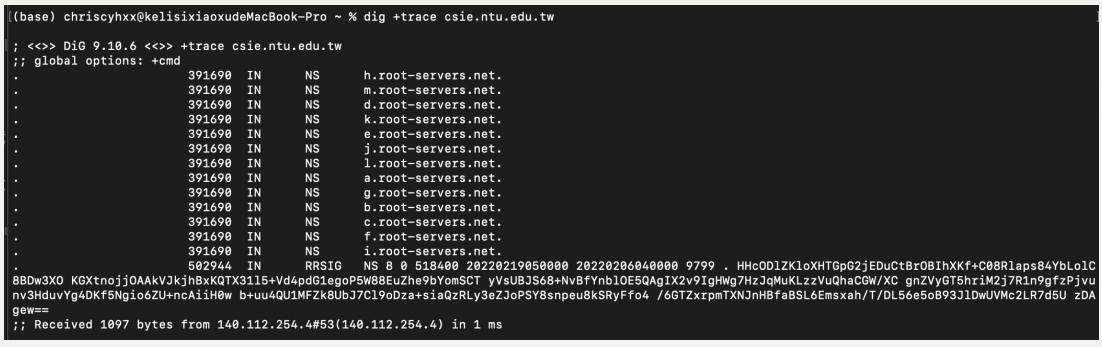
   (d) ceiba.ntu.edu.tw

   透過`nslookup <IP address>`即可查到對應的domain name。

2. (a) 140.112.71.61，透過指令`ip a`即可查詢。

   (b)

   - 140.112.254.4，透過`nslookup csie.ntu.edu.tw`可得DNS server。
   - 透過`dig +trace csie.ntu.edu.tw`可得delegation path。



   (c)

   - 172.20.10.1
   - 使用VPN之後會使原本的IP被屏蔽，變成140.112.71.61，由於所使用的

VPN有提供DNS server，所以會優先使用VPN所提供的DNS server。當我切斷VPN，就使用本來我的ISP所提供的DNS server。

(Ref: https://security.stackexchange.com/questions/13900/if-i-use-a-vpn-who-will-resolve-my-dns-requests)

(d) 透過`traceroute 172.20.10.1`即可查到：

```
(base) chriscyhxx@kelisixiaoxudeMacBook-Pro ~ % traceroute 172.20.10.1
traceroute to 172.20.10.1 (172.20.10.1), 64 hops max, 52 byte packets
 1  172.20.10.1 (172.20.10.1)  9.812 ms  3.080 ms  2.458 ms
```

System Administration

討論對象：
b05504066 李旻翰
b09505014 王聖文

Flag:

- **1-1**: `NASA{P1-1_H3r3_i5_the_f1a6_from_SSH}`
- **1-2**: `NASA{P1-2_D0nt_p3ek_my_s3cret_inf0}`
- **1-3**: `NASA{P1-3_I_f0und_wher3_!_am!!}`
- **2**: `NASA{P2_H@v3_y0u_p1ayed_P4cman_83fore?By_th3_w4y,I_u5e_Arch!}`
- **3**:`NASA{P3_3nj0y_y0ur_exqu1si73_p4ck4geXD}`
- **4**:`NASA{P4_6R3p_Th3_REgex_0f_tH3_fLaG}`
- **5-1**:`NASA{P5-1_find_ex3c_sav3$_us_fr0m_Wind0w5}`
- **5-2**:`NASA{P5-2_#ow_pow3rful_find_ex3c_is}`
- **6-1**:`NASA{P6-1_Red1rec7ion_is_v3ry_imp0rtan7}`
- **6-2**:`NASA{P6-2_H4v3_y0u_h3@rd_0f_std!n_$tdout_std3rr_4nd_file_d3$criptor?}`
- **6-3**:`NASA{P6-3_Can_y0u_d0n3_it_w1t#ou7_syntax_$u6ar_&>?}`
- **6-4**:`NASA{P6-4_P!pe_1s_4lso_v3ry_1mport@nt!B7W_7he_ord3r_of_redir3ct1on$_ma77er!}`
- **7**:`NASA{P7_9reping_and_7ransla7in9_in_W0nderl@nd!}`
- **8-1**:`NASA{P8-1_Be_careful_wi7#_5p4c!@l_c#arac7er5}`
- **8-2**:`NASA{P8-2_strac3_i5_a_g00d_t0ol_to_d38ug_with_sy$tem_c@1ls}`
- **9-1**:`NASA{P9-1_du_c#ecks_home_qu0ta_on_C$13_w0rkst47i0n}`
- **9-2**:`NASA{P9-2_.you_.f0und_.#idden_.f11es_.in_.11nux!}`
- **9-3**:`NASA{P9-3_Did_you_7h!nk_the_fi1esy$7em_is_s7r4ng3XD}`
- **10-1**:`NASA{P10-1_5ym801!c_link_7ricks_me}`
- **10-2**:`NASA{P10-2_Have_you_impl3mented_sm@rt_po1nter_8efore(X}`
- **10-3**:`NASA{P10-3_How_stran9e_emp7y_sp@rse_fil3s_are!}`
- **11-1**:`NASA{P11-1_chr00t_provid3s_i$olati0n}`
- **11-2**:`NASA{P11-2_sp3ci41_c#aract3r_file$_1n_man4}`
- **11-3**:`NASA{P11-3_1_10ve_7#at_co10rful_tr@in!!!}`
- **12-1**:`NASA{P12-1_SI9INT_and_SIGQUIT_ar3_5!gn@ls!}`

- **12-2**:`NASA{P12-2_ctrl+z_suspend_7#e_m0n5ter!}`
- **12-3**:`NASA{P12-3_k1ll_mon$ter_w1th_more_p3ople!}`
- **12-4**:`NASA{P12-4_k!llall_ki11_a11}`
- **13-1**:`NASA{P13-1_tcpdump_c@n_@1s0_dump_udp}`
- **13-2**:`NASA{P13-2_nc4t_1s_pow3rful_i$n'7_i7?}`
- **13-3**:`NASA{P13-3_nc@t_3x3c_#elp_cr34te_simpl3_socket}`
- **13-4**:`NASA{P13-4_simpl3_port_f0rwarding_w!th_nc47!}`
- **14-1**:`NASA{P14-1_sp117_d!ff_pa7ch_cat_in_a_r0w}`
- **14-2**:`NASA{P14-2_ext3n$i0n5_are_ac7u@11y_me@n1n91ess}`
- **14-3**:`NASA{P14-3_$y$t3mctl_man@ge_systemd_s3rvice5}`
- **14-4**:`NASA{P14-4_0p3nssl_can_man4g3_T1S_c3rtificat35!}`
- **15**:`NASA{P15_C0ngra7ulati0ns!!!Y0u_w0n_th3_pri2es_@nd_b3c0me_@_CSIE_m11liona!re!!!}`

---

1. 1-1直接登入可得：

   ```
   Hi, here is the flag: NASA{P1-1_H3r3_i5_the_f1a6_from_SSH}
   Enjoy your journey in NASA hw0!
   Last login: Sun Jan 30 06:05:34 2022
   ```

   得到自身簡介：`getent passwd nasa`

   ```
   [nasa@nasahw0 ~]$ getent passwd nasa
   nasa:x:1000:1001:NASA{P1-2_D0nt_p3ek_my_s3cret_inf0}:/home/nasa:/bin/bash
   ```

   以`pwd`可查詢所在目錄

   ```
   [nasa@nasahw0 ~]$ pwd
   NASA{P1-3_I_f0und_wher3_!_am!!}
   ```

2. 查詢pacman指令的作用：`man pacman`

   ```
   NASA{P2_H@v3_y0u_p1ayed_P4cman_83fore?By_th3_w4y,I_u5e_Arch!}
   ```

3. 以`cp ./beginner/package.tar.gz .`複製一份package到~

   再以`tar zxvf package.tar.gz`解壓縮後，發現存在一份`flag.txt`

   將其以`cat flag.txt`印出即可得：

```
[nasa@nasahw0 ~]$ cat flag.txt
 ________________________________________
< NASA{P3_3nj0y_y0ur_exqu1si73_p4ck4geXD} >
 ----------------------------------------
        \   ^__^
         \  (oo)\_______
            (__)\       )\/\
                ||----w |
                ||     ||
```

4. 使用`grep -o -E "NASA{P4_[a-zA-Zo-9_]{26}} -r ."`去遞迴的搜尋符合該表示式的flag，會加`-o`代表只印出match的部分，`-E`是因為有用到`{26}`，`-r`是代表遞迴的從指定目錄往下搜尋。

```
[nasa@nasahw0 p4]$ grep -o -E "NASA{P4_[a-zA-Z0-9_]{26}}" -r .
./8/7/1/8/0/zwwrd:NASA{P4_6R3p_Th3_REgex_0f_tH3_fLaG}
```

5. 先用`unzip windows.zip`解壓縮

   再用`find . -type d -exec chmod 0755 {} \;`將directories的權限改成755，以及`find . -type f -exec chmod 0644 {} \;`將一般file的權限改644，得第一個flag：

```
[nasa@nasahw0 p5]$ ./check
Thank you for your help, here is the flag: NASA{P5-1_find_ex3c_sav3$_us_fr0m_Wind0w5}
```

   以`find . -type f -exec file {} \; | grep 'ELF' | cut -d : -f 1 | xargs chmod a+x`先把所有file找出來並且透過`grep`挑選哪些屬於`ELF`，並把檔名用`cut`切出來，最後用`chmod a+x`將所有人的權限改成可執行。

```
Here is the bonus flag: NASA{P5-2_#ow_pow3rful_find_ex3c_is}
```

6. 以`./prayer | ./kamisama`直接將prayer想說的傳給kamisama，得到第一個flag：

```
NASA{P6-1_Red1rec7ion_is_v3ry_imp0rtan7}
```

   `./prayer 2>&1 | ./kamisama`，把error message寫到stdout就能將所有東西透過pipe傳給kamisama：

```
NASA{P6-2_H4v3_y0u_h3@rd_0f_std!n_$tdout_std3rr_4nd_file_d3$criptor?}
```

   先`./prayer 2>err.txt`將error message都輸出至`err.txt`，並且透過`./kamisama <err.txt`將內容傳給kamisama：

```
No wonder he comes to talk to me. NASA{P6-3_Can_y0u_d0n3_it_w1t#ou7_syntax_$u6ar_&>?}
```

   `./prayer 2>&1 1>out.txt | ./kamisama`，透過`2>&1`將error都寫到stdout，再將原本從stdout寫出的資料寫到其他地方，就能透過pipe將error傳給kamisama：

```
NASA{P6-4_P!pe_1s_4lso_v3ry_1mport@nt!B7W_7he_ord3r_of_redir3ct1on$_ma77er!}
```

7. 先用`sed -i 's/[^a-zA-Z0-9]//g' story.txt`將非英文數字的字元刪除

接著用`tr -d '\n' < story.txt > tmp.txt`將`story.txt`中的換行字元刪除，最後再用`tr`

```
tr -s
'qHyUMf6QXlOR18SbjsIk7PWYoC3chpgx94wirt0uenNKm5AdDTvEFZ2BJzGLV
a,'
'xfP7Xng_paSF@}izoJIMhQbVYyH8KL4m\!G01RwA6uNBed{qWrlsC3U5TtDEk
v9' <story.txt | grep -o 'NASA{.*}'
```

將加密結果改回來並且以`grep`找出flag：

```
[[nasa@nasahw0 p7]$ grep -o 'NASA{.*}' tmp.txt
NASA{P7_9reping_and_7ransla7in9_in_W0nderl@nd!}
```

8. 第一個flag直接將`./bad_kid`的輸出結果用`grep`搜尋即可：

```
[[nasa@nasahw0 p8]$ ./bad_kid | grep -o 'NASA{.*}'
NASA{P8-1_Be_careful_wi7#_5p4c!@l_c#arac7er5}
```

第二個flag透過`strace`追蹤`./bad_kid`執行期間使用到的system call，指令為：`strace -e write -s 859 ./bad_kid`，透過`-s`增加顯示的資訊量，去看`.\bad_kid`寫了什麼到`/dev/null`。

```
QQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQ
NASA{P8-2_strac3_i5_a_g00d_t0ol_to_d38ug_with_sy$tem_c@1ls}
QQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQ
```

9. 透過`du -a | sort -r -n | head -n 40`將前40大的檔案都印出來，看哪些是file，透過`du -a`可以把隱藏檔一併列入，再透過`rm`將那些檔案刪除，可得前兩個flag：

```
You delete the first redundant file: NASA{P9-1_du_c#ecks_home_qu0ta_on_C$13_w0rkst47i0n}
You delete the second redundant file: NASA{P9-2_.you_.f0und_.#idden_.f11es_.in_.11nux!}
```

我們要先在`mnt/p9`透過`sudo umount ./3/3/2`將`p9/3/3/2`還原到和`mnt`同一個file system，再使用`du -a | sort -r -n | head -n 40`去看有哪些大的檔案，會發現剛剛未顯示的`./3/3/2/xdxyc`很大，此時將它刪除即可得第三個flag：

```
Here is the bonus flag: NASA{P9-3_Did_you_7h!nk_the_fi1esy$7em_is_s7r4ng3XD}
```

10. 可以透過`dd if=/dev/zero of=./tmp bs=1M count=1`先製造一個1mb的大檔案，然後透過symbolic link `ln -s tmp 0`，將0-9都連`tmp`檔，可以得到第一個flag：

```
here is the first flag: NASA{P10-1_5ym801!c_link_7ricks_me}
```

可以透過`dd if=/dev/zero of=./tmp bs=1M count=1`先製造一個1mb的大檔案，然後透過hard link `ln tmp 0`，將0-9都連到`tmp`檔，可以得到第二個flag：

```
They are actually the same files!
Here is the second flag: NASA{P10-2_Have_you_impl3mented_sm@rt_po1nter_8efore(X}
```

透過`dd of=./0 bs=1 count=0 seek=1M`可以製造1mb的空白檔，以此方法創造0-9就可以得到第三個flag：

```
Here is the third flag: NASA{P10-3_How_stran9e_emp7y_sp@rse_fil3s_are!}
You should let the files occupy the same amount of storage to get the other two flags.
```

11. 透過在`p11/cage`裡執行`sudo chroot .`，得到第一個flag：

```
[[nasa@nasahw0 cage]$ sudo chroot .
Welcome to the chroot cage! NASA{P11-1_chr00t_provid3s_i$olati0n}
```

透過以下指令，建造所需的檔案，再用`chmod`去更改權限

```
mknod /dev/console c 5 1
mknod /dev/null c 1 3
mknod /dev/zero c 1 5
mknod /dev/ptmx c 5 2
mknod /dev/tty c 5 0
mknod /dev/random c 1 8
mknod /dev/urandom c 1 9
```

即可獲得第二的flag：

```
Here is your flag: NASA{P11-2_sp3ci41_c#aract3r_file$_1n_man4}
```

先離開chroot狀態，執行以下指令：

```
mount -t proc /proc proc/
mount -t sysfs /sys sys/
mount --rbind /dev dev/
sed -i 's/CheckSpace/#CheckSpace/1' pacman.conf
cp /etc/resolv.conf etc/resolv.conf
sudo chroot .
pacman -S sl
pacman -Syu lolcat
```

接著進到`root`執行`./train`可以得到第三個flag：

```
[(chroot) [root@nasahw0 ~]# ./train
Ya! I see the train! NASA{P11-3_1_10ve_7#at_co10rful_tr@in!!!}
```

12. 使用`ctrl+\`將monster殺死，得到第一個flag：

```
I didn't expect you will use ctrl+\ qq. NASA{P12-1_SI9INT_and_SIGQUIT_ar3_5!gn@ls!}
```

第二個方法是先用`ctrl+z`暫停process，然後透過`ps`查詢pid，再以`kill`殺死monster，就得到第二個flag：

```
[(chroot) [root@nasahw0 /]# ./monster
No one can stop me! Hahaha!
No one can stop me! Hahaha!
No one can stop me! Hahaha!
^ZIt's ctrl+z!? However, it can only "stop" me but cannot "terminate" me. I will come back soon!

[1]+  Stopped                 ./monster
[(chroot) [root@nasahw0 /]# ps
    PID TTY          TIME CMD
  57020 pts/1    00:00:00 sudo
  57021 pts/1    00:00:00 bash
  57072 pts/1    00:00:00 monster
  57073 pts/1    00:00:00 ps
[(chroot) [root@nasahw0 /]# kill 57072
[(chroot) [root@nasahw0 /]# fg
./monster
Oh no!!!! I will take revenge on you soon. NASA{P12-2_ctrl+z_suspend_7#e_m0n5ter!}
```

使用`./monster_revenge 2>&1 >flag.txt &`讓process在background跑，之後使用`kill -14 pid`送`SIGALRM`給process再用`kill -9 pid`殺死他，可以得到第三個flag：

```
Oh no!!!! I hate this sound! I'll give you the flag and don't annoy me anymore!
NASA{P12-3_k1ll_mon$ter_w1th_more_p3ople!}
```

先執行`./clone_monster > flag.txt &`，接著使用`killall -ALRM clone_monster`將所有process都殺掉，拿到第四個flag：

```
How can you kill all of my clones!!? I surrender. NASA{P12-4_k!llall_ki11_a11}
```

13. 執行`./Alice`和`./Bob`後，透過`lsof -P | grep Alice`去看他們是以哪個port溝通，發現是localhost:30000，便以`sudo tcpdump -i any -X 'port 30000'`去看傳輸內容，得到第一個flag：

    ```
    NASA{P13-1_tcpdump_c@n_@1s0_dump_udp}
    ```

    執行`./Carlos`後，使用指令：`ncat -l -k -p 30001 >out`將傳輸內容傳入`out`，再透過：`sed -i 's/Hi, thank you for comming! This is a present for you: //g'`以及`tr -d '\n' < out`即可看到第二個flag：

    ```
    NASA{P13-2_nc4t_1s_pow3rful_i$n'7_i7?}
    ```

    透過`lsof -P | grep Charlie`知道目前是使用localhost:38787，接著便以`ncat localhost 38787 -e ./Eve`得到第三個flag：

    ```
    [[nasa@nasahw0 p13]$ ncat localhost 38787 -e ./Eve
     flag: NASA{P13-3_nc@t_3x3c_#elp_cr34te_simpl3_socket}
    ```

    先使用`sudo iptables -t nat -A OUTPUT -p tcp -d localhost --dport 40002 -j REDIRECT --to-ports 30002`進行連接埠轉送，並且以`sudo sysctl -w net.ipv4.ip_forward=1`讓系統允許port forwarding，然後執行`./Dave`和`./Faythe`並以`sudo tcpdump -i any -X 'port 40002`去看傳輸資料，即可得

到第四個flag：

```
NASA{P13-4_simpl3_port_f0rwarding_w!th_nc47!}
```

14. 先以`split -C 831 -d ./mails`，再寫shell script：

```
#!/bin/sh
rm mails
for i in $(ls)
do
  if [ "${i:0:1}" = "x" ]
  then
    sed -i '8i MIME-Version: 1.0' $i
    sed -i '9i Content-Type: text/plain; charset=ascii'  $i
    sed -i '4c Received: by csie.ntu.edu.tw' $i
    cat $i >> mails
  fi
done
```

執行完之後，`./check`就會得到第一個flag：

```
Thank you for help. NASA{P14-1_sp117_d!ff_pa7ch_cat_in_a_r0w}
```

先以`mv flag.jpg flag.gz`改檔名，再用`gunzip flag.gz`解壓縮，接著再用`mv flag.zip flag.tiff`改檔名，然後用本機透過`scp`來傳遞圖檔即可得第二個flag：

```
NASA{P14-2_ext3n$i0n5_are_ac7u@11y_me@n1n91ess}
```

先以`sudo systemctl restart postfix.service`開啟postfix，再用`sudo systemctl enable postfix`將其設成自動開啟，可以拿到第三個flag：

```
Here is the flag you want: NASA{P14-3_$y$t3mctl_man@ge_systemd_s3rvice5}
```

15. 第1題：如果在使用`ln -s`建立symbolic link時，打的是相對路徑的話，那就會以相對路徑指向某個檔，因此在`cp`之後，在`/home/nasa/`內的`symlnk`會指到`../A`，但不存在這個檔，所以導致`ENOENT`。而hard link裡存的是某個檔的i-node資料，會直接指向該檔案，所以在哪個資料夾開啟`hdlnk`都能正確指到`/home/nasa/A`。

第2題：在使用`ln -s`建立symbolic link時，打的是相對路徑的話，那就會以相對路徑指向某個檔，就會以cwd相對該檔案的相對路徑去創造symbolic link，所以進到`~/D/`後，`lnk1`指到的檔就是`~/D/D/A`，而`lnk2`指到的就是`~/A`。

第3題：swap space存於硬碟，透過`swapon -s`可以就可以看到swap space的資訊。

第4題：一般 tmpfs 都是使用virtual memory代替傳統的永久保存媒體存放檔案。

第5題：`/etc`內有configuration file，而libraries預設是存在`/usr/local/lib, /usr/local/lib64, /usr/lib, /usr/lib64`，Linux log files可以在`/var/log`找到，所以選BADC。

第6題：其實由字面上的意思可以推測，process information應存於`/proc`；kernel modules and configurations與系統有關，應存於`/sys`；character special file又稱character device，block special file又稱block device他們應存於`/dev`。

第7題：只要對某資料夾有`x`的權限，且對內部的某個檔有r權限即可讀該檔，因此file1可讀，而file2和file3不可。

第8題：要對某個資料夾有`wx`權限才可移除其內部檔案。

第9題：因為要對某個資料夾有`wx`權限才可移除其內部檔案，所以沒辦法移除file1，但對`/home/alice`有`x`權限，所以可以往下面的資料夾繼續進行remove。因爲對`/home/alice/public`有`wx`所以可以刪file2，但不可刪掉`/home/alice/public/data`這個資料夾，因為沒辦法刪除裡面的file3導致該資料夾非空故無法刪除。

第10題：只要alice的某個dir有開sticky bit ，就只有alice或su或擁有他們的權限的process可刪那裡頭的檔。

第11題：因為`rm3, rm5`皆有other執行的權限，且都有setuid，所以他們可以去刪除`/tmp`底下的檔案。

第12題：進到`/usr/bin`之後，可以用`ls -l`來看他們的權限即可知。

第13題：進到`/home`之前都有`x`的權限，所以可以在`/home` share。同時也可知在`/tmp`以及`/tmp2`是可以的。

第14題：可以透過`ulimit -a`查詢，發現process number有誤。

第15題：透過`type time`發現其為shell keyword，選B；透過`type echo`發現其為shell builtin，選B；透過`which ls`發現會執行`/usr/bin/ls`，選E；透過`which pwd`發現會執行`/home/nasa/.bin/pwd` ，選G。

NASA{P15_C0ngra7ulati0ns!!!Y0u_w0n_th3_pri2es_@nd_b3c0me_@_CSIE_m11liona!re!!!}