3.終端機基本操作

勞動部 產業人才投資計畫
中國文化大學 推廣教育部
張耀鴻 副教授
2022年 暑期班

綱要

- Bash環境
- 管線與重定向
- 文字搜尋和操弄
- 從命令列編輯檔案
- 檔案比對
- 管理程序
- 檔案和命令監控
- 下載檔案
- 客製化Bash環境

Bash環境

- 用echo查看環境變數

```
┌──(kali㉿kali)-[~]
└─$ echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/local/games
:/usr/games

┌──(kali㉿kali)-[~]
└─$ echo $USER
kali

┌──(kali㉿kali)-[~]
└─$ echo $PWD
/home/kali

┌──(kali㉿kali)-[~]
└─$ echo $HOME
/home/kali
```

export

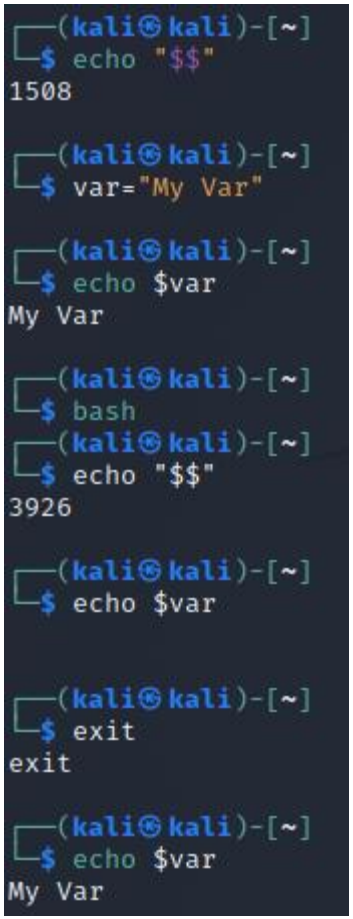
- 使用export命令定義環境變數
  - 例如，如果我們正在掃描一個目標，並且不想重複輸入系統的IP位址，我們可以快速為其分配一個環境變數，並使用該變數：

```
┌──(kali㉿kali)-[~]
└─$ export b=10.0.2.15

┌──(kali㉿kali)-[~]
└─$ ping -c 2 $b
PING 10.0.2.15 (10.0.2.15) 56(84) bytes of data.
64 bytes from 10.0.2.15: icmp_seq=1 ttl=64 time=1.92 ms
64 bytes from 10.0.2.15: icmp_seq=2 ttl=64 time=0.066 ms

--- 10.0.2.15 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1006ms
rtt min/avg/max/mdev = 0.066/0.993/1.920/0.927 ms
```

  - 如果我們設定環境變數時沒有使用export，只能在目前shell中使用該環境變數

$$

- $$變數顯示目前shell的process ID，以確保我們的確在兩個不同的shell中發出命令：



用export定義全域環境變數

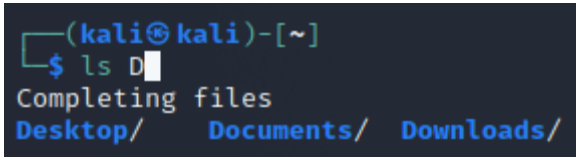
env

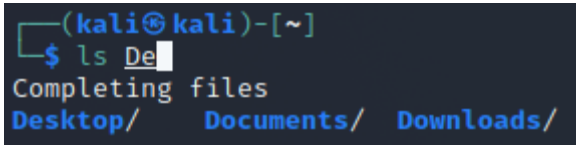
- 在Kali Linux中，預設情況下還定義了許多其他環境變數。我們可以在命令列上執行env來查看這些內容：

```
┌──(kali㉿kali)-[~]
└─$ env
COLORFGBG=15;0
COLORTERM=truecolor
COMMAND_NOT_FOUND_INSTALL_PROMPT=1
DBUS_SESSION_BUS_ADDRESS=unix:path=/run/user/1000/bus
DESKTOP_SESSION=lightdm-xsession
DISPLAY=:0.0
DOTNET_CLI_TELEMETRY_OPTOUT=1
GDMSESSION=lightdm-xsession
GDM_LANG=en_US.utf8
GTK_MODULES=gail:atk-bridge
HOME=/home/kali
LANG=en_US.UTF-8
LANGUAGE=
LOGNAME=kali
PANEL_GDK_CORE_DEVICE_EVENTS=0
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/local/
games:/usr/games
POWERSHELL_TELEMETRY_OPTOUT=1
POWERSHELL_UPDATECHECK=Off
PWD=/home/kali
QT_ACCESSIBILITY=1
QT_AUTO_SCREEN_SCALE_FACTOR=0
QT_QPA_PLATFORMTHEME=qt5ct
SESSION_MANAGER=local/kali:@/tmp/.ICE-unix/936,unix/kali:/tmp/.ICE-unix/936
```

Tab自動完成命令

- Bash shell自動完成功能允許我們使用Tab鍵完成檔案名和目錄路徑。這個特性大大加快了shell的使用速度





history

- 在進行滲透測試時。記錄輸入到shell中的命令很重要。幸運的是，Bash維護了已輸入命令的歷史記錄，可以隨history命令一起顯示

```
┌──(kali㉿kali)-[~]
└─$ history
    1
    2  gcc
    3  su
    4  sudo passwd su
    5  sudo passwd root
    6  su
    7  cd Downloads
    8  ls
    9  sudo apt install ./code_1.63.2-1639562499_amd64.deb
   10  vscode
```

```
  125  ls Desktop
  126  history
  127  cat /etc/lsb-release
  128  ls /etc/lsb*
  129  ls /etc

┌──(kali㉿kali)-[~]
└─$ !125

┌──(kali㉿kali)-[~]
└─$ ls Desktop
```

!!

- !!會重覆在terminal上執行的最後一個命令：

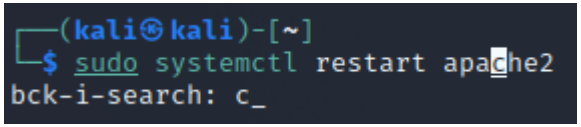
```
┌──(kali㉿kali)-[~]
└─$ sudo systemctl restart apache2
[sudo] password for kali:

┌──(kali㉿kali)-[~]
└─$ !!

┌──(kali㉿kali)-[~]
└─$ sudo systemctl restart apache2
```

HISTSIZE和HISTFILESIZE

- 命令歷史記錄保存在使用者主目錄中的.bash_history檔案中
- HISTSIZE控制當前會話儲存在記憶體中的命令數目
- HISTFILESIZE配置歷史檔中保留的命令數目
- 這些變數可以根據需要進行編輯，並保存到Bash設定檔(.bashrc)中

再次執行history中的命令

- 上鍵和下鍵可選擇要再次執行的命令
- CTRL+R再按一個字母可反向搜尋(reverse-i-search)history中包括該字母的命令



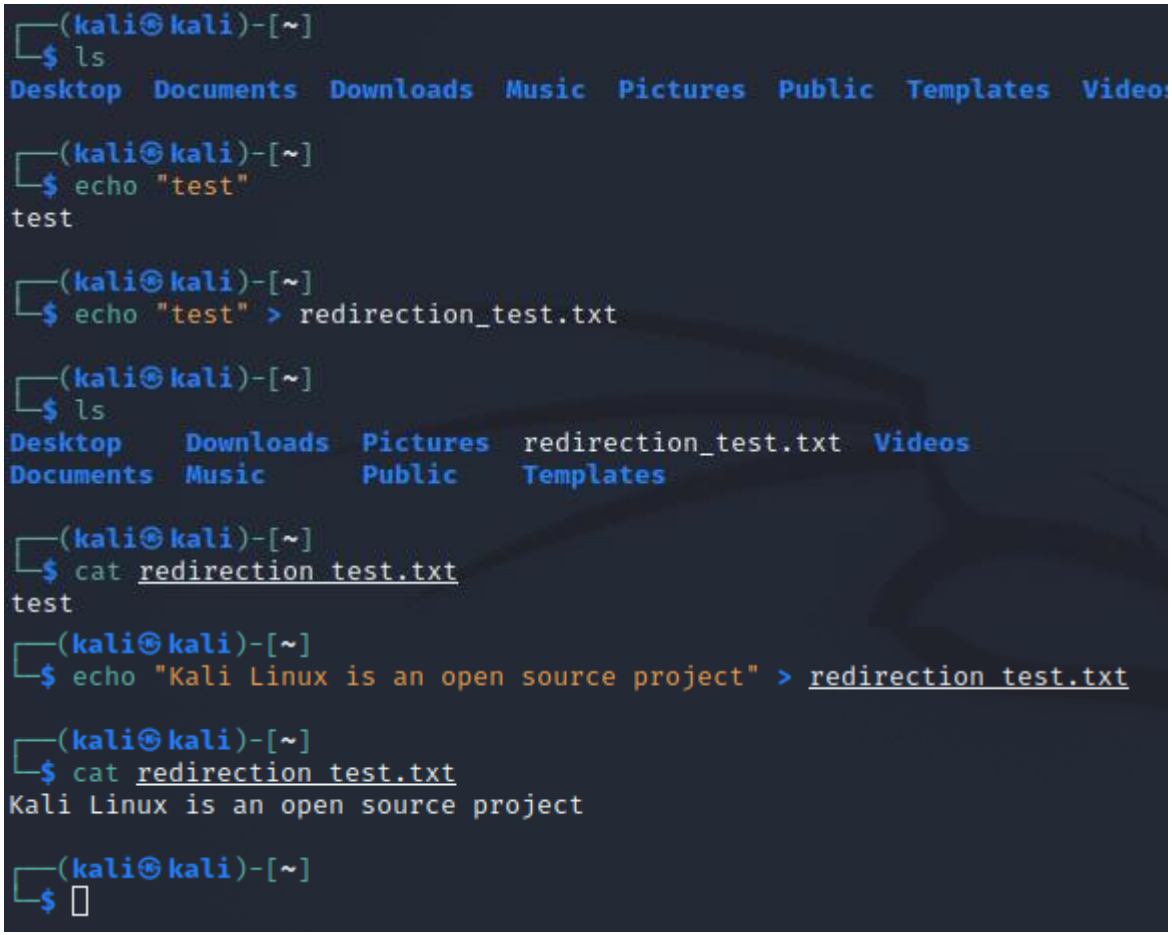
Exercise 3-1

1. 檢視bash歷史記錄，並用!指定要重新執行的命令
2. 透過reverse-i-search反向搜尋曾經執行的命令

管線與重定向

- 從命令列執行的每個程式都有三個連接到它的資料串流，作為與外部環境的溝通通道。這些串流如下：
  - ✓ STDIN: 標準輸入
  - ✓ STDOUT: 標準輸出
  - ✓ STDERR: 錯誤訊息輸出

重定向到新檔


┌──(kali㉿kali)-[~]
└─$ ls
Desktop Documents Downloads Music Pictures Public Templates Video
┌──(kali㉿kali)-[~]
└─$ echo "test"
test
┌──(kali㉿kali)-[~]
└─$ echo "test" > redirection_test.txt
┌──(kali㉿kali)-[~]
└─$ ls
Desktop Downloads Pictures redirection_test.txt Videos
Documents Music Public Templates
┌──(kali㉿kali)-[~]
└─$ cat redirection test.txt
test
┌──(kali㉿kali)-[~]
└─$ echo "Kali Linux is an open source project" > redirection test.txt
┌──(kali㉿kali)-[~]
└─$ cat redirection test.txt
Kali Linux is an open source project
┌──(kali㉿kali)-[~]
└─$

其他重定向命令

- 重定向到舊檔

```
┌──(kali㉿kali)-[~]
└─$ echo "that is maintained and funded by Offensive Security" >>  redirectio
n test.txt
```

```
┌──(kali㉿kali)-[~]
└─$ cat redirection test.txt                                          130 ×
Kali Linux is an open source project
that is maintained and funded by Offensive Security
```

- 從檔案重定向

```
┌──(kali㉿kali)-[~]
└─$ wc -m < redirection test.txt
89
```

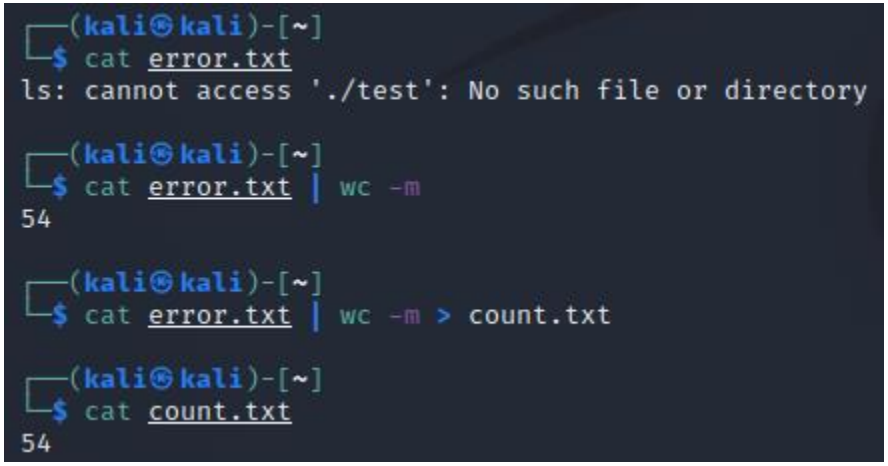
重定向STDERR

- 根據POSIX規範，STDIN、STDOUT和STDERR的檔案描述符號分別為0、1和2。這些數字很重要，因為它們可以用於在執行或連接不同命令時操縱來自命令列的相對應資料串流

```
┌──(kali㉿kali)-[~]
└─$ ls .
Desktop    Downloads  Music     Public                Templates
Documents  error.txt  Pictures  redirection_test.txt  Videos

┌──(kali㉿kali)-[~]
└─$ ls ./test
ls: cannot access './test': No such file or directory

┌──(kali㉿kali)-[~]
└─$ ls ./test 2>error.txt

┌──(kali㉿kali)-[~]
└─$ cat error.txt
ls: cannot access './test': No such file or directory
```

管線

- 用管線字元（|）可將一個命令的輸出重定向到另一個命令的輸入
- 這個概念可能看似不重要，但將不同的命令連接在一起是處理各種資料的強大工具



Exercise 3-2

1. 在Kali Linux系統上，結合cat命令和sort命令把/etc/passwd檔的內容重新排序
2. 將上一個練習的輸出重定向到家目錄中的檔案

文字搜尋和操弄

- 文字處理主要藉由grep, sed, cut, 和 awk命令
- 需對正規表示法有一定程度的認識
- 可參考以下網站資料
  - http://www.rexegg.com/
  - http://www.regular-expressions.info/

grep

- 簡而言之，grep在文字檔中搜尋給定正規表示法的樣式，並將符合的結果輸出到標準輸出
- 命令列選項:
  - ✓ -r: 遞迴搜尋
  - ✓ -i: 忽略大小寫

```
┌──(kali㉿kali)-[~]
└─$ ls -la /usr/bin | grep zip
-rwxr-xr-x  3 root root         38984 Jul 20  2020 bunzip2
-rwxr-xr-x  3 root root         38984 Jul 20  2020 bzip2
-rwxr-xr-x  1 root root         18424 Jul 20  2020 bzip2recover
-rwxr-xr-x  1 root root         22944 Jan 10  2021 funzip
-rwxr-xr-x  1 root root          3516 Apr 22  2021 gpg-zip
-rwxr-xr-x  2 root root          2346 Mar  2  2021 gunzip
```

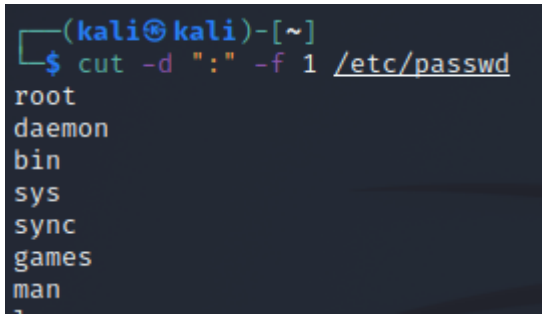
sed

- sed是一個強大的串流編輯器。它也是非常複雜的，這裡只簡單地描述一下
- sed可以高階的方式對文本串流（一組特定檔案或標準輸出）進行編輯:

```
┌──(kali㉿kali)-[~]
└─$ echo "I need to try hard" | sed 's/hard/harder/'
I need to try harder
```

cut

- cut命令很簡單，但通常很方便。cut用於從一行中提取一段文字，並將其輸出到STDOUT
- 一些最常用的選項包括-f 用於我們要裁切的欄位編號, -d用於欄位分隔符號



awk

- awk是一種為文字處理而設計的程式語言，通常用作資料提取和報告的工具
- awk也非常強大，並且可以非常複雜，在這裡只觸及一點皮毛
- 一個常用的選項是欄位分隔符號(-F)和輸出結果文字的print命令

```
┌──(kali㉿kali)-[~]
└─$ echo "hello::there::friend" | awk -F "::" '{print $1, $3}'
hello friend
```

綜合演練

- 假設我們有一個Apache HTTP伺服器日誌(http://www.offensive-security.com/pwk-files/access_log.txt.gz)，裡面有攻擊的證據
- 我們的任務是利用Bash命令檢查檔案並發現各種資訊，例如攻擊者是誰以及伺服器上到底發生了什麼事
- 首先，我們將用head和wc命令快速查看日誌檔，以瞭解其結構。head命令顯示檔案中的前10行，wc -l 命令顯示檔案中的總共有幾行

```
┌──(kali㉿kali)-[~]
└─$ wget http://www.offensive-security.com/pwk-files/access_log.txt.gz   1 ×
URL transformed to HTTPS due to an HSTS policy
--2022-03-11 08:18:02--  https://www.offensive-security.com/pwk-files/access_
log.txt.gz
Resolving www.offensive-security.com (www.offensive-security.com)... 192.124.
249.5
Connecting to www.offensive-security.com (www.offensive-security.com)|192.124
.249.5|:443... connected.
HTTP request sent, awaiting response ... 200 OK
Length: 3783 (3.7K) [application/x-gzip]
Saving to: 'access_log.txt.gz'

access_log.txt.gz   100%[═══════════════════⟹]   3.69K  --•-KB/s    in 0s

2022-03-11 08:18:02 (18.0 MB/s) - 'access_log.txt.gz' saved [3783/3783]


┌──(kali㉿kali)-[~]
└─$ gunzip access_log.txt.gz
gzip: access_log.txt already exists; do you wish to overwrite (y or n)? y

┌──(kali㉿kali)-[~]
└─$ mv access_log.txt access.log

┌──(kali㉿kali)-[~]
└─$ head access.log
201.21.152.44 - - [25/Apr/2013:14:05:35 -0700] "GET /favicon.ico HTTP/1.1" 40
4 89 "-" "Mozilla/5.0 (Windows NT 6.2; WOW64) AppleWebKit/537.31 (KHTML, like
 Gecko) Chrome/26.0.1410.64 Safari/537.31" "random-site.com"
70.194.129.34 - - [25/Apr/2013:14:10:48 -0700] "GET /include/jquery.jshowoff.
min.js HTTP/1.1" 200 2553 "http://www.random-site.com/" "Mozilla/5.0 (Linux;
U; Android 4.1.2; en-us; SCH-I535 Build/JZO54K) AppleWebKit/534.30 (KHTML, li
ke Gecko) Version/4.0 Mobile Safari/534.30" "www.random-site.com"
┌──(kali㉿kali)-[~]
└─$ wc -l access.log
1173 access.log
```

- 請注意，日誌檔是基於文字的，包含由空格分隔的不同欄位（IP位址、時間戳記、HTTP請求等）
- 這是一個非常適合用grep來處理的檔案，而且也適用於我們到目前為止介紹的所有工具
- 首先我們搜尋向伺服器發出的HTTP請求，以查找此日誌檔中記錄的所有IP位址
- 我們將cat命令的輸出透過管線傳到cut和sort命令中
- 這可提供有關需要消除的潛在攻擊者數量的線索

```
┌──(kali㉿kali)-[~]
└─$ cat access.log | cut -d " " -f 1 | sort -u
201.21.152.44
208.115.113.91
208.54.80.244
208.68.234.99
70.194.129.34
72.133.47.242
88.112.192.2
98.238.13.253
99.127.177.95
```

- 我們看到日誌檔中記錄的IP位址不到10個，儘管這仍然沒有告訴我們關於攻擊者的任何資訊
- 接下來，我們用uniq和sort來顯示沒有重覆的行，進一步完善我們的輸出
- 並根據每個IP位址存取伺服器的次數對資料進行排序
- -c 選項將在輸出行前面加上出現的次數

```
┌──(kali㉿kali)-[~]
└─$ cat access.log | cut -d " " -f 1 | sort | uniq -c | sort -urn
   1038 208.68.234.99
     59 208.115.113.91
     22 208.54.80.244
     21 99.127.177.95
      8 70.194.129.34
      1 201.21.152.44
```

- 有幾個IP位址出現次數比較多，但我們將首先關注存取頻率最高的位址
- 把208.68.234.99位址過濾出來，並顯示和統計該IP請求的資源：

```
┌──(kali㉿kali)-[~]
└─$ cat access.log | grep '208.68.234.99' | cut -d "\"" -f 2 | uniq -c
   1038 GET //admin HTTP/1.1
```

- 從這個輸出來看，208.68.234.99的IP位址似乎是只存取/admin 目錄，讓我們進一步檢查一下

```
┌──(kali㉿kali)-[~]
└─$ cat access.log | grep '208.68.234.99' | grep '/admin ' | sort -u
208.68.234.99 - - [22/Apr/2013:07:51:20 -0500] "GET //admin HTTP/1.1" 401 742
 "-" "Teh Forest Lobster"
208.68.234.99 - admin [22/Apr/2013:07:51:25 -0500] "GET //admin HTTP/1.1" 200
 575 "-" "Teh Forest Lobster"

┌──(kali㉿kali)-[~]
└─$ cat access.log | grep '208.68.234.99' | grep -v '/admin '
```

- 顯然208.68.234.99嘗試了針對該web伺服器的HTTP暴力破解。此外，在大約1000次嘗試後，似乎暴力破解成功了

Exercise 3-3

1. 在Kali的/etc/passwd 檔案中，找出所有shell為/bin/false的使用者的家目錄，需以一行Bash指令列印到螢幕上，輸出範例如下：

```
The user mysql home directory is /nonexistent
The user tss home directory is /var/lib/tpm
The user Debian-snmp home directory is /var/lib/snmp
The user speech-dispatcher home directory is /run/speech-dispatcher
The user lightdm home directory is /var/lib/lightdm
```

2. 將/etc/passwd檔複製到家目錄（/home/kali）
3. 再用cat在一行命令中列印kali/passwd，並將其中"Light Display Manager"字串用"LDM"取代

從命令列編輯檔案

- vi是一個強大的文本編輯器，效率很高，學習成本也比較大。使用命令 vi 來編輯文件

```
┌──(kali㉿kali)-[~]
└─$ vi intro_to_vi.txt
```

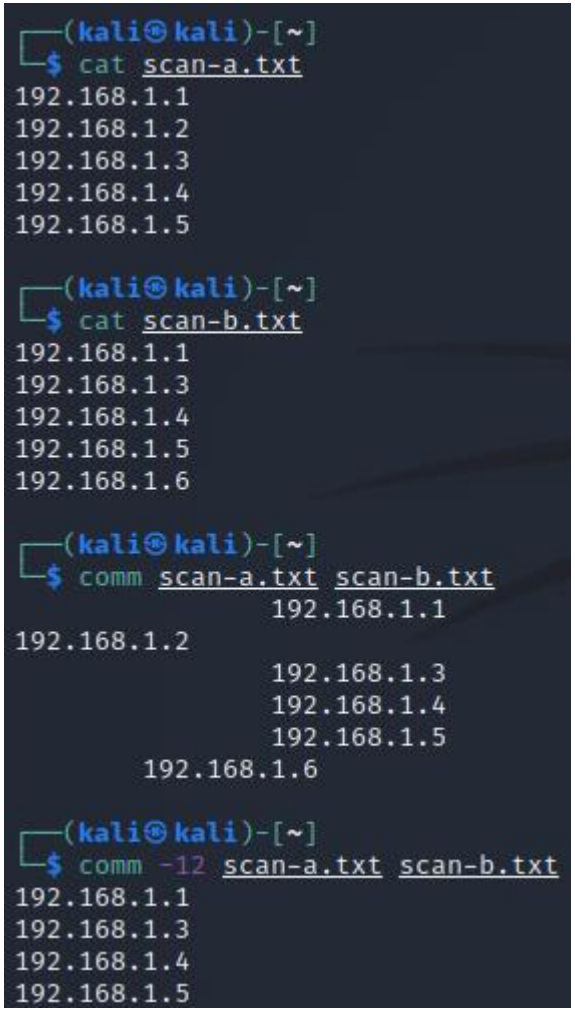
- 進入vi後，用 i 命令，啟用編輯模式
- 編輯完成後，用 Esc 鍵退出編輯模式，回到命令模式
- 在命令模式下，常用操作為：
  - ✓ dd: 刪除目前行
  - ✓ yy: 複製目前行
  - ✓ p: 貼上已複製的內容
  - ✓ x: 刪除目前游標處的字元
  - ✓ :w: 儲存文件
  - ✓ :q!: 強制退出編輯並放棄儲存
  - ✓ :wq: 儲存檔案並退出

檔案比對

- comm
- diff
- vimdiff

comm

- comm命令簡單對比兩個檔案，輸出每行的對比結果
- 對比結果分為3個欄位，分別代表近第一個檔中的行，兩個檔中同時存在的行，以及僅在第二個檔中的行
- 可透過 -n 選項，選擇顯示對比結果



diff

- diff命令和comm命令差不多，但是支援更多輸出選項
- 兩個常用的選項是 -c (context format)和-u (unified format)

```
┌──(kali㉿kali)-[~]
└─$ diff -c scan-a.txt scan-b.txt
*** scan-a.txt  2022-03-11 10:03:53.043255289 -0500
--- scan-b.txt  2022-03-11 10:04:28.877329271 -0500
***************
*** 1,5 ****
  192.168.1.1
- 192.168.1.2
  192.168.1.3
  192.168.1.4
  192.168.1.5
--- 1,5 ----
  192.168.1.1
  192.168.1.3
  192.168.1.4
  192.168.1.5
+ 192.168.1.6
```

```
┌──(kali㉿kali)-[~]
└─$ diff -u scan-a.txt scan-b.txt
--- scan-a.txt  2022-03-11 10:03:53.043255289 -0500
+++ scan-b.txt  2022-03-11 10:04:28.877329271 -0500
@@ -1,5 +1,5 @@
 192.168.1.1
-192.168.1.2
 192.168.1.3
 192.168.1.4
 192.168.1.5
+192.168.1.6
```

diff輸出所代表的意思

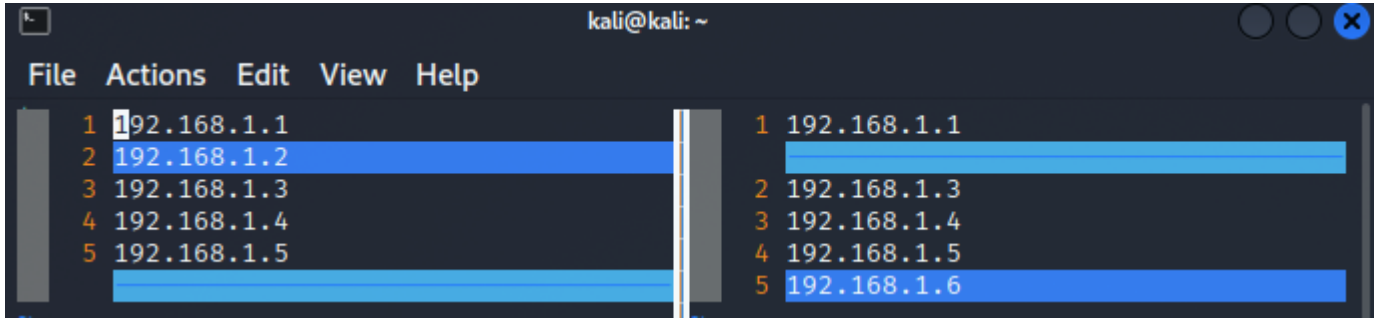
- 輸出的"-"表示該行出現在第一個檔中，但不在第二個檔中
- "+"表示該行出現在第二個檔中，但不在第一個檔中
- 這些格式之間最顯著的區別是，統一格式不顯示檔之間匹配的行，從而縮短了結果

vimdiff

- vimdiff透過vim打開多個檔案，不同部分用高亮顯示

```
┌──(kali㉿kali)-[~]
└─$ vimdiff scan-a.txt scan-b.txt
```

- 常用快捷鍵：
  - ✓ do: 從其他視窗獲取修改，應用到目前視窗
  - ✓ dp: 將目前視窗的改變應用到另一個視窗
  - ✓]c: 跳到下一處修改部分
  - ✓ [c: 跳到上一處
  - ✓ Ctrl+W: 切換到另一個視窗



Exercise 3-4

1. 從https://offensive-security.com/pwk-files/scans.tar.gz 下載掃瞄結果檔案
2. 解壓縮並比對掃描結果有何不同

管理程序

- Linux 核心透過程序管理多個任務
- 核心掌握每個程序的資訊
- 每個程序擁有一個PID（Process ID）
- Linux shell 引進jobs這個概念
- 比如，cat error.txt | wc -m是一個擁有兩個程序的管線命令，shell 將其視為一個job

bg：改為後台執行

- 之前的命令都是在前台（foreground）執行，這意味著在目前命令結束前，不能執行其他命令
- 我們可以將命令放在後台執行以便同時執行其他命令
- 一個簡單方法是在命令後面加上 & 符號，這表示在命令開始時將其送到後台

```
┌──(kali㉿kali)-[~]
└─$ ping -c 400 localhost > ping_results.txt &
[1] 65686
```

任務管理：jobs和fg命令

- 用 jobs 和 fg 命令可以快速檢查ICMP echo 的狀態
- jobs命令可以列舉出目前終端機下正在執行的任務，fg命令可以將後台任務改到前台

```
┌──(kali㉿kali)-[~]
└─$ ping -c 400 localhost > ping_results.txt
^Z
zsh: suspended  ping -c 400 localhost > ping_results.txt

┌──(kali㉿kali)-[~]
└─$ jobs
[1]  - running    ping -c 400 localhost > ping_results.txt
[2]  + suspended  ping -c 400 localhost > ping_results.txt
┌──(kali㉿kali)-[~]
└─$ fg %1
[1]  - running    ping -c 400 localhost > ping_results.txt
^C

┌──(kali㉿kali)-[~]
└─$ jobs
[2]  + suspended  ping -c 400 localhost > ping_results.txt
```

任務管理：ps 和kill

- ps (process status)命令比 jobs 命令更強大也更有用
- ps命令不止列舉出當前終端會話下的程序，也會列出系統程序等
- 在滲透測試中，存取主機後的一件事就是檢查系統中有那些軟體正在執行，這可以幫助我們評估當前權限、收集附加資訊，為後續的操作做準備

```
┌──(kali㉿kali)-[~]
└─$ ps -ef
UID          PID    PPID  C STIME TTY          TIME CMD
root           1       0  0 05:22 ?        00:00:12 /sbin/init splash
root           2       0  0 05:22 ?        00:00:00 [kthreadd]
root           3       2  0 05:22 ?        00:00:00 [rcu_gp]
root           4       2  0 05:22 ?        00:00:00 [rcu_par_gp]
root           6       2  0 05:22 ?        00:00:00 [kworker/0:0H-kblockd]
root           8       2  0 05:22 ?        00:00:00 [mm_percpu_wq]
root           9       2  0 05:22 ?        00:00:00 [rcu_tasks_rude_]
root          10       2  0 05:22 ?        00:00:00 [rcu_tasks_trace]
```

- 如果要結束一個程序，可以透過 kill 命令後跟PID 來實現

Exercise 3-5

1. 在後臺執行特定命令，找出過去7天內在Kali機器上更改的檔
2. 重新執行上一個命令並將其暫停(suspend)；一旦被暫停，它就會跑到後臺
3. 把之前後臺的job帶到前臺
4. 在Kali系統上啟動Firefox瀏覽器，用ps和grep找出Firefox的PID
5. 從命令列(透過PID)終止Firefox

檔案和命令監控

- 對檔案和命令進行即時監控在滲透測試中非常重要
- 兩個很有用的命令：
  - ✓ tail
  - ✓ watch

tail

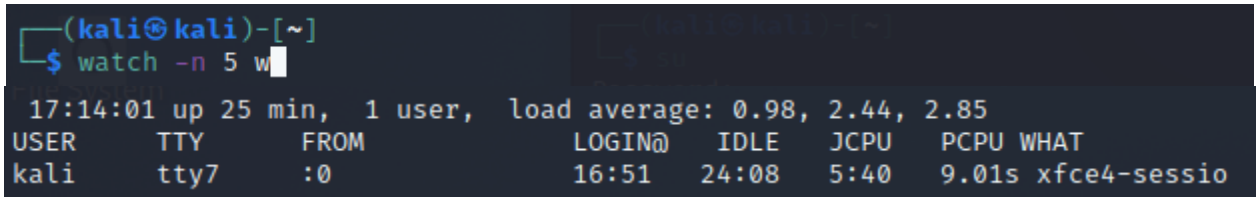
- ***tail***命令通常用來來監控日誌檔，以下為監控 Apache 日誌的例子：

```
┌──(kali㉿kali)-[~]
└─$ sudo tail -f /var/log/apache2/access.log                                130 ×
::1 - - [11/Mar/2022:17:08:26 -0500] "GET / HTTP/1.1" 200 3380 "-" "Mozilla/5
.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0"
::1 - - [11/Mar/2022:17:08:31 -0500] "GET /favicon.ico HTTP/1.1" 404 488 "-"
"Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0"
::1 - - [11/Mar/2022:17:08:31 -0500] "GET /icons/openlogo-75.png HTTP/1.1" 20
0 6040 "http://localhost/" "Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20
100101 Firefox/78.0"
```

- -f選項可以在目標檔案更新時持續更新輸出
- -nX選項可以輸出指定行數（n），預設輸出10 行

watch

- watch命令可以用指定的時間間隔執行一個（已定義的）命令，預設間隔時間是2s
- -n X 選項可自行定義間隔時間
- 例如列出登錄使用者（w命令），每5s 刷新：
- 用 Ctrl + C 結束命令



Exercise 3-6

1. 啟動apache2 web service並在本機存取，同時即時監控其access.log檔。
2. 結合watch和ps來監控Kali機器上CPU密集度最高的程序；啟動不同的應用程式，查看清單如何即時更改。

下載檔案

- wget
- curl
- axel

wget

- wget是常用的命令，可以透過HTTP/HTTPS 或 FTP 協議下載檔案
- 選項 -O 將文件以指定的名稱下載到指定位置：

```
┌──(kali㉿kali)-[~]
└─$ wget -O report_wget.pdf https://www.offensive-security.com/reports/penetr
ation-testing-sample-report-2013.pdf
--2022-03-11 17:52:15--  https://www.offensive-security.com/reports/penetrati
on-testing-sample-report-2013.pdf
Resolving www.offensive-security.com (www.offensive-security.com)... 192.124.
249.5
Connecting to www.offensive-security.com (www.offensive-security.com)|192.124
.249.5|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 27691955 (26M) [application/pdf]
Saving to: 'report_wget.pdf'

report_wget.pdf     100%[===================>]  26.41M  2.76MB/s    in 13s

2022-03-11 17:52:30 (1.97 MB/s) - 'report_wget.pdf' saved [27691955/27691955]
```

curl

- curl可以從伺服器取得資料或將資料傳到伺服器
- 支援多種協議，包括IMAP/S, POP3/S, SCP, SFTP, SMB/S, SMTP/S, TELNET, TFTP 等
- 在滲透測試中，使用curl可以下載或上傳檔案，或者建構更複雜的請求

```
┌──(kali㉿kali)-[~]
└─$ curl -o report.pdf https://www.offensive-security.com/reports/penetration
-testing-sample-report-2013.pdf
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Curre
nt
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--
  0 26.4M    0  7782    0     0   7044      0  1:05:31  0:00:01  1:05:30  711
  1 26.4M    1  463k    0     0   263k      0  0:01:42  0:00:01  0:01:41  264
  3 26.4M    3  975k    0     0   352k      0  0:01:16  0:00:02  0:01:14  354
```

axel

- axel是一個下載加速器，透過多個連線將檔案從FTP/HTTP 伺服器下載下來
- axel有很多特性，較常用的是 -n 選項，指定連線數（-a顯示進度條；-o指定儲存檔案名稱）：

```
┌──(kali㉿kali)-[~]
└─$ axel -a -n 20 -o report_axel.pdf https://www.offensive-security.com/reports/penetration-testing-sample-report-2013.pdf
Initializing download: https://www.offensive-security.com/reports/penetration-testing-sample-report-2013.pdf
File size: 26.4091 Megabyte(s) (27691955 bytes)
Opening output file report_axel.pdf
Starting download

Connection 1 finished
Connection 1 finished
Connection 19 finished
[ 52%] [.0....2.3.4.5 .6.7.8 .9.A.BJ.C.D1E .F.G.H .I ... ] [ 357.4KB/s] [00:35]
```

客製化Bash環境

- 客製化Bash 歷史記錄
- 別名（Alias）
- 持久化Bash 配置

客製化Bash 歷史記錄

- 常用的三個自定義 history 命令操作和相關傳回資料的環境變數：
  - ✓ HISTCONTROL
  - ✓ HISTIGNORE
  - ✓ HISTTIMEFORMAT

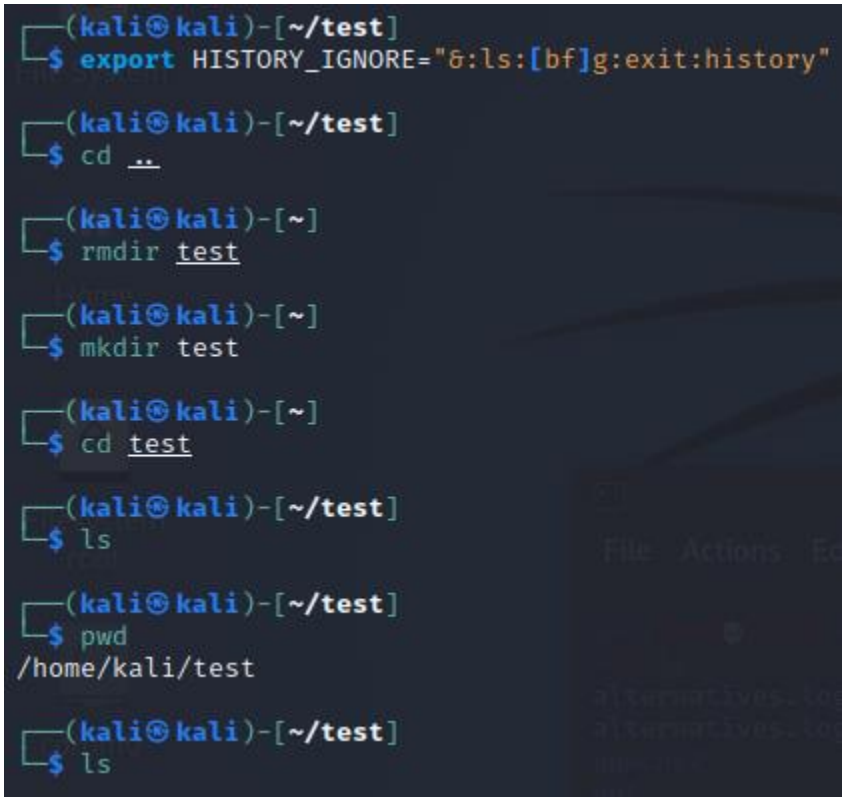
HISTCONTROL

- 定義是否移除重複的命令，或者開頭是空格的命令，預設是移除
- 調整該參數，使其只移除重複命令：

```
┌──(kali㉿kali)-[~]
└─$ export HISTCONTROL=ignoredups
```

HISTIGNORE

- 過濾 ls, exit, history, bg 等常用命令：



HISTTIMEFORMAT

- 控制時間戳記的顯示：

```
┌──(kali㉿kali)-[~]
└─$ export HISTTIMEFORMART='%F %T '

┌──(kali㉿kali)-[~]
└─$ history
    1
    2  gcc
    3  su
    4  sudo passwd su
    5  sudo passwd root
    6  su
```

別名（Alias）

- 別名是一串可以代替常用命令和參數的字串
- 使用別名可以讓常用命令更簡短

```
┌──(kali㉿kali)-[~]
└─$ alias lsa='ls -la'

┌──(kali㉿kali)-[~]
└─$ lsa
total 68644
drwxr-xr-x 20 kali kali     4096 Mar 11 18:16 .
drwxr-xr-x  3 root root     4096 Dec 20 01:31 ..
-rw-r--r--  1 kali kali       56 Mar 11 09:53 a.c
-rw-r--r--  1 kali kali   141136 Feb 11  2020 access.log
```

持久化Bash 配置

- Bash 的系統配置在/etc/bash.bashrc
- 每個使用者都可以編輯自己根目錄下的.bashrc檔（~/.bashrc）自行定義自己的配置
- .bashrc會在每次使用者登錄時執行，本質上是一個shell script，所以可以插入任何希望執行的命令

```
┌──(kali㉿kali)-[~]
└─$ cat ~/.bashrc
# ~/.bashrc: executed by bash(1) for non-login shells.
# see /usr/share/doc/bash/examples/startup-files (in the package bash-doc)
# for examples

# If not running interactively, don't do anything
case $- in
    *i*) ;;
      *) return;;
esac
```


END