

1. GIỚI THIỆU

CTF (Capture the Flag) - là loại hình thi đấu dành riêng cho cộng đồng bảo mật máy tính. Các đội chơi tham gia sẽ tìm kiếm “cờ” (flag) được giấu trong hệ thống, trong các ứng dụng hoặc các file đã được bảo vệ. Mỗi cờ là một chuỗi ký tự đặc biệt và các đội chơi phải tìm cách truy cập và chiếm cờ đó.

Để chơi CTF, người tham gia cần có các kiến thức về mã hóa, an ninh mạng, phân tích mã độc, kỹ năng lập trình. CTF chia thành các mảng chính:

- **Web exploit** (Khai thác lỗ hổng web): Các thử thách tập trung vào lỗ hổng bảo mật của ứng dụng web.
- **Reverse engineering** (Dịch ngược): Các thử thách tập trung vào phân tích và giải mã các phần mềm, tệp tin.
- **Crypto** (Mật mã): Các thử thách tập trung vào các thuật toán mã hóa và giải mã.
- **Pwnable** (Khai thác lỗ hổng): Các thử thách tập trung vào khai thác các lỗ hổng bảo mật trong hệ thống hoặc phần mềm.
- **Forensics** (Truy vết): Các thử thách tập trung vào việc thu thập và phân tích các dữ liệu số.

Bandit của OverTheWire là một tập hợp các bài thi CTF dành cho người hoàn toàn mới. Người tham gia sẽ học cách sử dụng dòng lệnh, tìm kiếm thông tin trong các tệp tin và thư mục, kiểm soát quyền truy cập của người dùng và các khái niệm khác liên quan đến bảo mật

2. WRITEUP

2.1. Tổng quan

Bandit có tổng cộng 35 level (gồm cả level 0). Server Bandit có thể kết nối thông qua Secure Shell (SSH). Thông tin đăng nhập được cấp từ level 0, và mỗi level sẽ có một flag là chuỗi password của level tiếp theo. Có nhiều cách để truy cập vào tệp password, ta chỉ cần làm đúng một cách là đủ.

Link tham gia: <https://overthewire.org/wargames/bandit/>

2.2. Bandit0

Mục tiêu của level0 là đăng nhập vào game thông qua SSH. Tên host server cần kết nối là **bandit.labs.overthewire.org**, port là **2220**. Username và password đồng thời là **bandit0**. Sau khi kết nối ta sẽ thấy tệp *readme*, trong đó có chứa flag (password cho level1).

Để kết nối SSH, ta sử dụng câu lệnh `ssh` với các tham số:

```
ssh [tên_host] -p [số_hiệu_cổng] -l [username]
```

Trong đó:

[tên_host] của chúng ta là **bandit.labs.overthewire.org**.

-p là port (cổng), ta kết nối đến cổng **2220**.

-l là login (đăng nhập), ta điền username **bandit0**.

Câu lệnh kết nối sẽ là:

```
ssh bandit.labs.overthewire.org -p 2220 -l bandit0
```

Ngoài ra, ta cũng có thể sử dụng lệnh cũng có chức năng tương tự:

```
ssh bandit0@bandit.labs.overthewire.org -p 2220
```

Sau khi yêu cầu kết nối, ta phải nhập password để truy cập

Sau khi truy cập xong, ta kiểm tra xem thư mục hiện tại có gì bằng lệnh

```
ls
```

`ls` viết tắt của *list* - nghĩa là liệt kê các tệp tin và thư mục có trong thư mục hiện tại

Ta nhận thấy có tệp *readme*. Để mở và xem nội dung tệp *readme*, ta có thể sử dụng một số câu lệnh như:

```
vi readme
```

```
vim readme
```

```
nano readme
```

```
cat readme
```

```
...
```

Dù bằng cách nào thì ta cũng đọc được flag trong đó là
NH2SXQwcBdpmTEzi3bvBHMM9H66vVXjL

Ta kết nối đến level 1 thông qua câu lệnh ssh:

```
ssh bandit.labs.overthewire.org -p 2220 -l bandit1
```

Mật khẩu chính là flag ta nhận được ở level 0

NH2SXQwcBdpmTEzi3bvBHMM9H66vVXjL

2.3. Bandit1

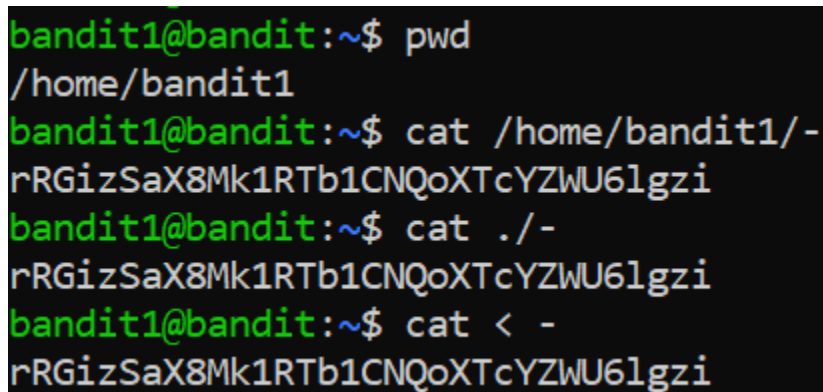
Sau khi kết nối, ta thực hiện lệnh ls để xem trên server có gì, nhận thấy có 1 file tên -

Lưu ý rằng dấu - thường được sử dụng làm chỉ thị cho một option nào đó khi chạy câu lệnh, nên nếu ta viết `cat -`, `vi -`... Máy tính sẽ không hiểu và cho rằng ta đang muốn sử dụng lệnh cat với một option nào đó nhưng chưa viết hết. Anh em có thể đơn giản google cách mở file - trong linux: “*How to read a - filename in linux*”.

Do đó, ta có thể sử dụng dấu < trước tên file để tường minh rằng ta chỉ muốn đọc nội dung 1 file có tên bắt đầu bằng kí tự -

```
cat < -
```

Hoặc ta có thể chỉ rõ đường dẫn đến file này thông qua đường dẫn tuyệt đối (lấy bằng câu lệnh `pwd`), hoặc đường dẫn tương đối (`./`):



```
bandit1@bandit:~$ pwd
/home/bandit1
bandit1@bandit:~$ cat /home/bandit1/-
rRGizSaX8Mk1RTb1CNQoXTcYZWU6lgzi
bandit1@bandit:~$ cat ./-
rRGizSaX8Mk1RTb1CNQoXTcYZWU6lgzi
bandit1@bandit:~$ cat < -
rRGizSaX8Mk1RTb1CNQoXTcYZWU6lgzi
```

Ta được giá trị flag: rRGizSaX8Mk1RTb1CNQoXTcYZWU6lgzi

Tiếp tục truy cập Bandit2 với flag trên

2.4. Bandit2

Ở level này, nhiệm vụ của chúng ta là tìm flag được chứa trong file “spaces in this filename”, và đúng như tên file miêu tả, file này chứa các ký tự khoảng trắng. Nếu ta chỉ đọc bằng dòng lệnh cat như thông thường:

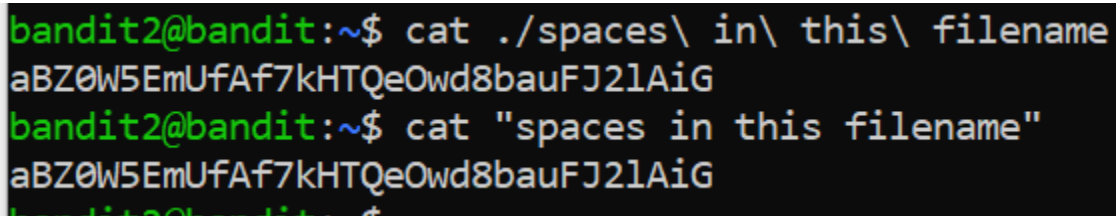
```
cat spaces in this filename
```

Máy tính sẽ hiểu ta sử dụng câu lệnh cat với các đối số lần lượt là spaces, in, this, filename; chứ không phải một tệp có tên là “spaces in the filename”. Để mở tệp có chứa ký tự khoảng trắng, ta đơn giản chỉ cần bao đóng tên file trong dấu nháy đôi là được.

```
cat "spaces in this filename"
```

Hoặc ta cũng có thể thêm các dấu \ trước ký tự khoảng trắng, nhằm tường minh cho máy hiểu ta muốn sử dụng ký tự khoảng trắng trong tên file:

```
cat ./spaces\ in\ this\ filename
```



```
bandit2@bandit:~$ cat ./spaces\ in\ this\ filename
aBZ0W5EmUfAf7kHTQeOwd8bauFJ2lAiG
bandit2@bandit:~$ cat "spaces in this filename"
aBZ0W5EmUfAf7kHTQeOwd8bauFJ2lAiG
```

Flag nhận được: aBZ0W5EmUfAf7kHTQeOwd8bauFJ2lAiG

2.5. Bandit3

```
ssh bandit3@bandit.labs.overthewire.org -p 2220
```

Flag của bài này sẽ được giấu trong thư mục **inhere** dưới dạng file ẩn. Để hiển thị file ẩn ta có thể sử dụng option -a (--all) với lệnh ls

```
ls -a
```

hoặc sử dụng câu lệnh find

Lưu ý rằng các tệp ẩn sẽ có tiền tố . và các thư mục sẽ có màu xanh.

```
bandit3@bandit:~$ ls -a
.  ..  .bash_logout  .bashrc  inhere  .profile
bandit3@bandit:~$ cd inhere/
bandit3@bandit:~/inhere$ ls -a
.  ..  .hidden
bandit3@bandit:~/inhere$ find
.
./ .hidden
```

Đọc nội dung tệp `.hidden`, ta thu được flag:
 2EW7BBsr6aMMoJ2HjW067dm8EgX26xNe

2EW7BBsr6aMMoJ2HjW067dm8EgX26xNe

2.6. Bandit4

```
ssh bandit4@bandit.labs.overthewire.org -p 2220
```

Yêu cầu ở level này là xác định loại file (cụ thể là: The password for the next level is stored in the only human-readable file in the **inhere** directory, tức là trong thư mục **inhere**, sẽ có 1 tệp ở định dạng con người đọc được và trong đó có flag). Ta sẽ sử dụng câu lệnh `file` để xác định loại tệp

```
file ./inhere/*
```

```
bandit4@bandit:~$ file ./inhere/*
./inhere/-file00: data
./inhere/-file01: data
./inhere/-file02: data
./inhere/-file03: data
./inhere/-file04: data
./inhere/-file05: Non-ISO extended-ASCII text, with NEL line terminators
./inhere/-file06: Non-ISO extended-ASCII text, with no line terminators, with escape sequences
./inhere/-file07: ASCII text
./inhere/-file08: data
./inhere/-file09: data
bandit4@bandit:~$ cat ./inhere/-file07
lrIWWI6bB37kxfiCQZqUdOIYfr6eEqR
```

Xác định được file ASCII khả đọc được và thu được flag tương ứng là
 lrIWWI6bB37kxfiCQZqUdOIYfr6eEqR

2.7. Bandit5

```
ssh bandit5@bandit.labs.overthewire.org -p 2220
```

Để lọc các file có các thuộc tính

- Human-readable
- Kích thước 1033 bytes
- Không thực thi được

Ta có thể sử dụng câu lệnh find:

```
find . -type f -readable ! -executable -size 1033c
```

Trong đó:

find .: Tìm kiếm các file trong thư mục hiện tại và các thư mục con của nó.

-type f: Lọc ra các file (không lọc thư mục).

-readable: Chỉ lấy các file mà người dùng hiện tại có quyền đọc.

! -executable: Không lấy các file có quyền thực thi.

-size 1033c: Lọc các file có kích thước đúng 1033 bytes.

```
bandit5@bandit:~$ find . -type f -readable ! -executable -size 1033c
./inhere/maybeh ere07/.file2
bandit5@bandit:~$ cat ./inhere/maybeh ere07/.file2
P4L4vucdmLnm8I7Vl7jG1ApGSfjYKqJU
```

Thu được flag: P4L4vucdmLnm8I7Vl7jG1ApGSfjYKqJU

2.8. Bandit6

```
ssh bandit6@bandit.labs.overthewire.org -p 2220
```

The password for the next level is stored somewhere on the server and has all of the following properties:

- owned by user bandit7
- owned by group bandit6
- 33 bytes in size

Tức là flag sẽ được lưu trữ đâu đó trên server, có những thuộc tính:

- do user bandit7 sở hữu
- do group bandit6 sở hữu
- có kích thước 33 bytes

Sử dụng lệnh `ls`, ta không thấy thư mục/tệp tin nào, tức là ta sẽ phải tìm ở gốc `/`:

```
find / -type f -size 33c -user bandit7 -group bandit6 2>/dev/null
```

Trong đó:

`find /`: tìm kiếm từ thư mục gốc `/`

`-type f`: chỉ tìm kiếm các tập tin (không phải thư mục)

`-size 33c`: các tập tin có kích thước bằng 33 bytes

`-user bandit7`: tập tin được sở hữu bởi user `"bandit7"`

`-group bandit6`: tập tin được sở hữu bởi group `"bandit6"`

`2>/dev/null`: chuyển hết các lỗi ra khỏi kết quả tìm kiếm để thuận tiện cho việc đọc kết quả. Cụ thể, `2>/dev/null` là một câu lệnh trong Linux được sử dụng để chuyển hướng thông báo lỗi (error output) từ màn hình điều khiển (console) sang "null device" (`/dev/null`), nghĩa là hủy bỏ thông báo lỗi đó. Trong Linux, `2` được sử dụng để chỉ định đầu ra lỗi (error output), và `/dev/null` là một thiết bị ảo (virtual device) được sử dụng để loại bỏ các dữ liệu được ghi vào đó.

```
bandit6@bandit:~$ find / -type f -size 33c -user bandit7 -group bandit6 2>/dev/null
/var/lib/dpkg/info/bandit7.password
bandit6@bandit:~$ cat /var/lib/dpkg/info/bandit7.password
z7WtoNQU2XfjmMtWA8u5rN4vzqu4v99S
```

Thu được flag: `z7WtoNQU2XfjmMtWA8u5rN4vzqu4v99S`

2.9. Bandit7

```
ssh bandit7@bandit.labs.overthewire.org -p 2220
```

Mật khẩu được lưu trữ trong file `data.txt`, cạnh từ khóa `millionth`.

Sử dụng `grep` để lọc:

```
cat data.txt | grep millionth
```

`Grep` (global regular expression print) sử dụng để tìm kiếm và in ra các dòng có chứa chuỗi ký tự được chỉ định trong các tập tin hoặc đầu ra của một lệnh khác. Câu lệnh trên là lọc nội dung tệp `data.txt`, tìm dòng có chứa chữ `millionth`

```
bandit7@bandit:~$ cat data.txt | grep millionth
millionth      TESKZC0XvTetK0S9xNwm25STk5iWrBvP
```

Thu được flag: TESKZC0XvTetK0S9xNwm25STk5iWrBvP

2.10. Bandit8

```
ssh bandit8@bandit.labs.overthewire.org -p 2220
```

Level này yêu cầu chúng ta đọc thông tin trong file **data.txt**, lọc dòng thông tin xuất hiện 1 lần duy nhất.

Để lấy thông tin trong file **data.txt**, ta dùng lệnh `cat data.txt` hoặc `strings data.txt` (`strings` là câu lệnh lấy các chuỗi con người đọc được trong file), sau đó sắp xếp theo bảng chữ cái bằng lệnh `sort` (để các dòng giống nhau sẽ được đặt cạnh nhau), và lọc các dòng trùng bằng câu lệnh `uniq -u`. Câu lệnh này sẽ hiển thị dòng nào xuất hiện một lần duy nhất.

```
cat data.txt | sort | uniq -u
strings data.txt | sort | uniq -u
```

```
bandit8@bandit:~$ strings data.txt | sort | uniq -u
EN632PlfYiZbn3PhVK3XOGS1NIInNE00t
bandit8@bandit:~$ cat data.txt | sort | uniq -u
EN632PlfYiZbn3PhVK3XOGS1NIInNE00t
```

Thu được flag: EN632PlfYiZbn3PhVK3XOGS1NIInNE00t

2.11. Bandit9

```
ssh bandit9@bandit.labs.overthewire.org -p 2220
```

Level yêu cầu tìm mật khẩu lưu trong file **data.txt**, là các chuỗi con người đọc được (human-readable), và có tiền tố là một vài dấu “=”

Ta lọc các chuỗi đọc được bằng câu lệnh `strings`, rồi tìm tiền tố là dấu “=” thông qua `grep`.

```
strings data.txt | grep '^='
```

Trong đó, `^` đại diện cho vị trí đầu tiên của một dòng.


```
bandit9@bandit:~$ strings data.txt | grep ==
f===== theM
===== password
===== is
===== G7w8LIi6J3kTb8A7j9LgrywtEUlyyp6s
bandit9@bandit:~$ strings data.txt | grep '^=='
===== password
===== is
===== G7w8LIi6J3kTb8A7j9LgrywtEUlyyp6s
bandit9@bandit:~$ strings data.txt | grep '^='
=XeOh
=Vb`
=I6a
===== password
===== is
===== G7w8LIi6J3kTb8A7j9LgrywtEUlyyp6s
```

Flag thu được: G7w8LIi6J3kTb8A7j9LgrywtEUlyyp6s

2.12. Bandit10

```
ssh bandit10@bandit.labs.overthewire.org -p 2220
```

Mật khẩu được lưu trong file data.txt, là dữ liệu được mã hóa theo chuẩn base64.

Ta chỉ cần giải mã bằng câu lệnh `base64 -d [tên_file]` (trong đó, `-d` tức là decode - giải mã)

```
base64 -d data.txt
```

```
bandit10@bandit:~$ cat data.txt
VGhlIHBhc3N3b3JkIGlzIDZ6UGV6aUxkUjJSS05kTl1GTmI2b1ZDS3pwaGxYSEJNCg==
bandit10@bandit:~$ base64 -d data.txt
The password is 6zPezILdR2RKNdNYFNb6nVCKzphlXHBM
```

Thu được flag: 6zPezILdR2RKNdNYFNb6nVCKzphlXHBM

2.13. Bandit11

```
ssh bandit11@bandit.labs.overthewire.org -p 2220
```

Flag lưu trữ trong file `data.txt`, với các kí tự chữ cái được mã hóa caesar với khóa dịch là 13 vị trí. (Rot13 \leftrightarrow rotate 13 position). Ta có thể sử dụng bất kỳ trình giải mã Caesar online nào, hoặc có thể sử dụng câu lệnh trên Terminal tr. Câu lệnh `tr (translate)` nhằm thay thế văn bản đã mã hóa với cụm văn bản đã dịch.

```
echo ciphertext | tr '[n-za-mN-ZA-M]' '[a-zA-Z]'
```

Trong đó, `ciphertext` là chuỗi đã được mã hóa cần giải mã, và `'[n-za-mN-ZA-M]' '[a-zA-Z]'` là hai bảng chữ cái tương ứng với ký tự đã mã hóa và ký tự gốc. Lệnh `tr` sẽ tìm ký tự tương ứng trong bảng chữ cái đã mã hóa và thay thế bằng ký tự tương ứng trong bảng chữ cái gốc để giải mã.

```
bandit11@bandit:~$ cat data.txt | tr '[n-za-mN-ZA-M]' '[a-zA-Z]'
```

The password is JVNBBFSmZwKKOP0XbFXOoW8chDz5yVRv

Thay thế `ciphertext` bằng nội dung trong `data.txt` bằng câu lệnh `cat data.txt`, thu được flag: `JVNBBFSmZwKKOP0XbFXOoW8chDz5yVRv`

2.14. Bandit12

```
ssh bandit12@bandit.labs.overthewire.org -p 2220
```

Ở level này, flag được giấu trong một tệp tin hexdump đã được nén nhiều lần. Để thực hiện tìm flag, hãy tạo một thư mục ở đường dẫn `/tmp`. Copy dữ liệu qua thư mục này, thực hiện lần lượt các bước: xác định loại file, đổi tên về đúng định dạng file, giải nén/đọc file tương ứng.

Để tạo thư mục ở `/tmp`, ta sử dụng lệnh `mkdir`:

```
mkdir /tmp/kit
```

```
bandit12@bandit:~$ mkdir /tmp/kit
```

Ta copy file cần phân tích qua thư mục và chuyển qua đường dẫn `/tmp/kit`

```
bandit12@bandit:~$ cp data.txt /tmp/kit
bandit12@bandit:~$ cd /tmp/kit
bandit12@bandit:/tmp/kit$
```

Thực hiện lệnh `cat data.txt`, ta nhận được nội dung tệp có dạng:

```

000001e0: c1d8 c71a 9b5d 5435 afa0 5eca 34ca a83c .....]T5..^.4.<
000001f0: 309e 6b5d 532f a0af 20e0 bc3f bb03 a680 0.k]S/.. ..?....
00000200: 6616 4b13 9d09 bf8b 3a93 6f16 b48a e6cf f.K.....:o.....
00000210: ccb9 084c 8a35 12a7 447d 8224 4491 e534 ...L.5..D}.$D..4
00000220: 0c71 2f36 fda1 8b54 0808 a144 9894 966f .q/6...T...D...o
00000230: be74 2140 952c 0294 a1d6 841e 1658 756f .t!@.,.....Xuo
00000240: 0d7f c5dc 914e 1424 14d9 a5a0 4043 a8c0 .....N.$....@C..
00000250: f434 0200 00 .4...

```

Đây là cấu trúc của file hexdump, có dạng:

OFFSET	HEXA BYTES	ASCII BYTES
00000000	48 65 6c 6c 6f 20 77 6f 72 6c 64	Hello world

Trong đó

- **OFFSET:** Địa chỉ bắt đầu của dòng trong file hexdump, được biểu diễn dưới dạng hexa.
- **HEXA BYTES:** Chuỗi các ký tự hexa biểu diễn các byte của file binary.
- **ASCII BYTES:** Chuỗi các ký tự ASCII tương ứng với các byte của file binary. Nếu byte có giá trị nhỏ hơn 32 hoặc lớn hơn 126 (trừ các ký tự đặc biệt như tab, carriage return, line feed...), thì sẽ được thay thế bằng ký tự ..

Để giải mã file hexdump, ta sử dụng câu lệnh `xxd` với cú pháp:

```
xxd [tùy chọn] [file]
```

Trong đó, [tùy chọn] là các tùy chọn để cấu hình đầu ra của câu lệnh, và [file] là tên file mà ta muốn xử lý. Một số tùy chọn thường được sử dụng trong câu lệnh `xxd` bao gồm:

- c: Số byte được hiển thị trên mỗi dòng. (character)
- g: Số byte được nhóm lại với nhau. (group)
- l: Số byte được đọc từ file. (length)
- r: Chuyển đổi từ hexdump sang file nhị phân (reverse)

Để sử dụng câu lệnh `xxd` để chuyển định dạng file hexdump về dạng khác, ta viết:

```
xxd -r data.txt > flag
```

Ta xác định kiểu file flag bằng câu lệnh `file flag`

```
file flag
```

```
bandit12@bandit:/tmp/kit$ xxd -r data.txt > flag
bandit12@bandit:/tmp/kit$ ls
data.txt  flag
bandit12@bandit:/tmp/kit$ file flag
flag: gzip compressed data, was "data2.bin", last modified: Tue Feb 14 2023, max compression, from Unix, original size modulo 2^32 564
```

Vậy đây là file đã được nén bằng gzip, ta đổi tên về đúng định dạng của gzip và giải nén:

```
mv flag flag.gz
```

```
gunzip flag.gz
```

```
bandit12@bandit:/tmp/kit$ mv flag flag.gz
bandit12@bandit:/tmp/kit$ gunzip flag.gz
bandit12@bandit:/tmp/kit$ ls
data.txt  flag
bandit12@bandit:/tmp/kit$ file flag
flag: bzip2 compressed data, block size = 900k
```

Tiếp tục xác định loại file thông qua file flag, ta nhận thấy đây là file được nén bằng bzip2, ta đổi tên về đúng định dạng bzip2 và giải nén:

```
mv flag flag.bz2
```

```
bzip2 -d flag.bz2
```

```
bandit12@bandit:/tmp/kit$ mv flag flag.bz2
bandit12@bandit:/tmp/kit$ bzip2 -d flag.bz2
bandit12@bandit:/tmp/kit$ ls
data.txt  flag
bandit12@bandit:/tmp/kit$ file flag
flag: gzip compressed data, was "data4.bin", last modified: Tue Feb 14 2023, max compression, from Unix, original size modulo 2^32 20480
```

Tiếp tục xác định loại file gzip, thực hiện tương tự

```
bandit12@bandit:/tmp/kit$ mv flag flag.gz
bandit12@bandit:/tmp/kit$ gunzip flag.gz
bandit12@bandit:/tmp/kit$ ls
data.txt  flag
bandit12@bandit:/tmp/kit$ file flag
flag: POSIX tar archive (GNU)
```

File tar sẽ được giải nén bằng câu lệnh:

```
tar -xvf flag
```

```
bandit12@bandit:/tmp/kit$ tar -xvf flag
data5.bin
bandit12@bandit:/tmp/kit$ file data5.bin
data5.bin: POSIX tar archive (GNU)
```

Lặp lại các bước: **xác định file** > **đổi tên** > **giải nén**, ta lần lượt thu được:
data5.bin (tar) > data6.bin (bzip2) > data6 (tar) > data8.bin (gzip) >
data8 (text)

```

bandit12@bandit:/tmp/kit$ file data5.bin
data5.bin: POSIX tar archive (GNU)
bandit12@bandit:/tmp/kit$ tar -xvf data5.bin
data6.bin
bandit12@bandit:/tmp/kit$ file data6.bin
data6.bin: bzip2 compressed data, block size = 900k
bandit12@bandit:/tmp/kit$ mv data6.bin data6.bz2
bandit12@bandit:/tmp/kit$ bzip2 -d data6.bz2
bandit12@bandit:/tmp/kit$ ls
data5.bin  data6  data.txt  flag
bandit12@bandit:/tmp/kit$ file data6
data6: POSIX tar archive (GNU)
bandit12@bandit:/tmp/kit$ tar -xvf data6
data8.bin
bandit12@bandit:/tmp/kit$ file data8.bin
data8.bin: gzip compressed data, was "data9.bin", last
original size modulo 2^32 49
bandit12@bandit:/tmp/kit$ mv data8.bin data8.gz
bandit12@bandit:/tmp/kit$ gunzip data8.gz
bandit12@bandit:/tmp/kit$ ls
data5.bin  data6  data8  data.txt  flag
bandit12@bandit:/tmp/kit$ file data8
data8: ASCII text

```

Đến đây, ta đọc nội dung tệp data8:

```
cat data8
```

```

bandit12@bandit:/tmp/kit$ cat data8
The password is wbWd1BxEir4CaE8LaPhauu0o6pwRmrDw

```

Và thu được flag wbWd1BxEir4CaE8LaPhauu0o6pwRmrDw

2.15. Bandit13

```
ssh bandit13@bandit.labs.overthewire.org -p 2220
```

Yêu cầu: flag được lưu trữ trong /etc/bandit_pass/bandit14 và chỉ có thể được đọc bởi người dùng bandit14. Ta có sẵn khóa bí mật SSH đã được lưu trữ ở file sshkey.private:

```
bandit13@bandit:~$ ls  
sshkey.private
```

```
bandit13@bandit:~$ cat sshkey.private  
-----BEGIN RSA PRIVATE KEY-----  
MIIEpAIBAAKCAQEAXkkOE83W2cOT7IWhFc9aPaaQmQDgzuxCv+ppZHa++buSkN+  
gg0tcr7Fw8NLGa5+Uzec2rEg0WmeevB13AIoYp0MZyETq46t+jk9puNwZwIt9XgB  
ZufGtZEwWbFWw/vVLNwOXBe4UWStGRWzgPpEeSv5Tb1VjLZIBdGphTIK22Amz6Zb  
ThMsiMnyJafEwJ/T8PQO3mys91vUHEuoOMAZoUID4kN0MEZ3+XahyK0HJVq68KsV  
ObefXG1vvA3GAJ29kxJaqrRfgYnqZryWN7w3CHjNU4c/2Jkp+n8L0SnxaNA+wYA7  
jiPyTF0is8uzM1YQ4l1Lzh/8/MpvhCQF8r22dwIDAQABAoIBAQC6dWBjhYE0zjeA  
J3j/RWmap9M5zfJ/wb2bfidNpwbB8rsJ4sZIDZQ7XuIh4LfygoAQSS+bBw3RXvzE  
pvJt3SmU8hIDuLsCjL1VnBY5pY7Bju8g8aR/3FyjyNAqx/TLfz1LYfOu7i9Jet67  
xAh0tONG/u8FB5I3LAI2Vp6OviwvdWeC4n0xCthldpuPKNLA8rmMMVRTKQ+7T2VS  
nXmwYckKUcUgzoVSpINZaS0zUDypdpy2+tRH3MQa5kqN1YKjvF8RC47woOYCKtsD  
o3FFpGNFec9Taa3Msy+DfQQhHKZFKIL3bJDONTmrVvtYK40/yeU4aZ/HA2DQzwhe  
o11AfiEhAoGBAOnVjosBkm7sb1K+n4IEwPxs8sOmhPnTDUy5WGrpSCrXOmsVIBUf  
1aL3ZGLx3xCTwtCnFucB9DvN2H7kunc/b6hTKLVLcXuyLD8njTrbRhlghC90nKs
```

Thử truy cập vào thư mục /etc/bandit_pass:

```
bandit13@bandit:/etc/bandit_pass$ ls  
bandit0  bandit13  bandit18  bandit22  bandit27  bandit31  bandit6  
bandit1  bandit14  bandit19  bandit23  bandit28  bandit32  bandit7  
bandit10  bandit15  bandit2  bandit24  bandit29  bandit33  bandit8  
bandit11  bandit16  bandit20  bandit25  bandit3  bandit4  bandit9  
bandit12  bandit17  bandit21  bandit26  bandit30  bandit5  
bandit13@bandit:/etc/bandit_pass$ cat bandit14  
cat: bandit14: Permission denied
```

Permission denied, tức là ta không có quyền mở tệp do hiện giờ ta đang đăng nhập thông qua user bandit13, user này không có quyền mở tệp mà phải là user bandit14. Ta cần đăng nhập vào đúng server này với user bandit14, sử dụng key đã có trong tệp sshkey.private.

Nhớ lại cú pháp câu lệnh ssh:

```
ssh [username]@[remote_host] -p [port_number]
```


Trong đó,

- [username] là user đăng nhập, ta điền bandit14
- [remote_host] chính là máy hiện tại, ta điền localhost.
- [port_number] là số hiệu cổng kết nối, ta điền 2220.

Ngoài ra, ta thêm tham số `-i sshkey.private` (`-i` là identity), tức là sử dụng khóa trong tệp `sshkey.private`. Thực hiện câu lệnh này ở đường dẫn `~` để câu lệnh trở vào đúng tệp `sshkey.private`.

```
cd ~
```

```
ssh bandit14@localhost -p 2220 -i sshkey.private
```

```
bandit13@bandit:/etc/bandit_pass$ cd ~
bandit13@bandit:~$ ssh bandit14@localhost -p 2220 -i sshkey.private
The authenticity of host '[localhost]:2220 ([127.0.0.1]:2220)' can't be established.
ED25519 key fingerprint is SHA256:C2ihUBV7ihnV1wUXRb4RrEcLfXC5CX1hmAAM/urerLY.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes_
```

Đăng nhập thành công, ta lấy thông tin flag ở đường dẫn `/etc/bandit_pass/bandit14`

```
cat /etc/bandit_pass/bandit14
```

```
Enjoy your stay!

bandit14@bandit:~$ cat /etc/bandit_pass/bandit14
fGrHPx402xGC7U7rXKDaxiWFTOiF0ENq
bandit14@bandit:~$
```

Thu được: `fGrHPx402xGC7U7rXKDaxiWFTOiF0ENq`

2.16. Bandit14

```
ssh bandit14@bandit.labs.overthewire.org -p 2220
```

Yêu cầu: Flag có thể thu được bằng cách nhập mật khẩu của level hiện tại lên port 30000 ở localhost.

Tức là ta cần truy cập vào cổng 30000, ở server localhost, sau đó nhập flag của level trước để lấy flag của level tiếp theo. Ta sẽ sử dụng câu lệnh `nc` (netcat).

Câu lệnh này cho phép thiết lập, tạo các kết nối TCP hoặc UDP, cũng như dò quét cổng.

Cú pháp của netcat là:

```
nc [OPTIONS] HOST PORT
```

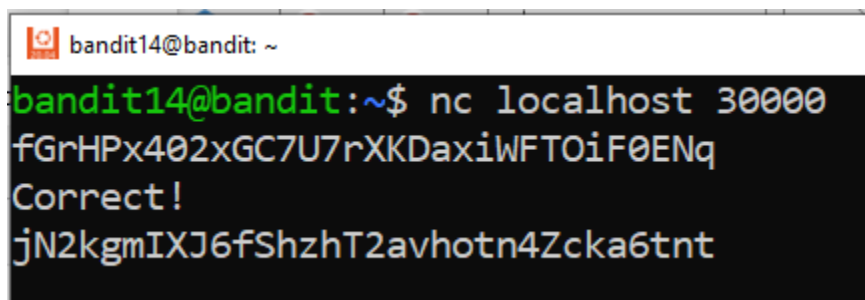
Trong đó:

- nc hoặc netcat là lệnh để khởi động netcat
- OPTIONS là tùy chọn thêm để cấu hình netcat, ví dụ như -v (verbose) để hiển thị thông tin chi tiết hơn về kết nối hoặc -l (listen) để netcat lắng nghe kết nối đến.
- HOST là địa chỉ IP hoặc tên miền của máy tính muốn kết nối đến.
- PORT là số cổng mà netcat sẽ kết nối đến.

Với yêu cầu cụ thể của level này, ta chỉ cần viết

```
nc localhost 30000
```

rồi nhập flag từ level trước (fGrHPx402xGC7U7rXKDaxiWFTOiF0ENq) là đủ:



```
bandit14@bandit: ~  
bandit14@bandit:~$ nc localhost 30000  
fGrHPx402xGC7U7rXKDaxiWFTOiF0ENq  
Correct!  
jN2kgmIXJ6fShzhT2avhotn4Zcka6tnt
```

Thu được: jN2kgmIXJ6fShzhT2avhotn4Zcka6tnt

2.17. Bandit15

```
ssh bandit15@bandit.labs.overthewire.org -p 2220
```

Yêu cầu: Flag có thể thu được bằng cách nhập mật khẩu của level hiện tại lên port 30001 ở localhost, sử dụng mã hóa SSL (Secure Socket Layer).

Lưu ý: SSL (Secure Sockets Layer) là một giao thức bảo mật dùng để bảo vệ sự truyền tải dữ liệu giữa máy khách và máy chủ trên internet. Nó được sử dụng để tạo ra một kết nối mạnh mẽ, mã hóa giữa trình duyệt của người dùng và máy chủ web để bảo vệ thông tin nhạy cảm như mật khẩu, số thẻ tín dụng và thông tin cá

nhân khác. Khi SSL được sử dụng, thông tin sẽ được mã hóa và chỉ có thể được giải mã bởi máy chủ được chỉ định, do đó tránh được các hacker truy cập và đánh cắp thông tin. SSL đã được thay thế bởi TLS (Transport Layer Security), nhưng thuật ngữ "SSL" vẫn được sử dụng rộng rãi cho các phiên bản mới hơn của giao thức bảo mật này.

netcat không hỗ trợ SSL. Ta sẽ phải sử dụng câu lệnh khác. Ở đây ta sẽ dùng OpenSSL. OpenSSL là một thư viện mã nguồn mở cung cấp các công cụ và thủ tục để mã hóa, giải mã và tạo chứng chỉ SSL / TLS. OpenSSL cũng cung cấp các công cụ để kiểm tra và xác minh chứng chỉ. Cú pháp như sau:

```
openssl s_client -connect <host>:<port>
```

Trong đó:

- `s_client` là tham số cho OpenSSL để tạo một kết nối SSL client
- `connect` là tham số để chỉ định ta muốn kết nối đến server với địa chỉ IP hoặc tên miền và port cụ thể.
- `<host>` và `<port>` được thay bằng thông tin tương ứng của server ta muốn kết nối đến.

Với level này, ta cần thiết lập `s_client`, kết nối đến tên miền `localhost` và cổng `30001`

```
openssl s_client -connect localhost:30001
```

```
bandit15@bandit:~$ openssl s_client -connect localhost:30001
```

Nhập flag của level trước

```
---
read R BLOCK
jN2kgmIXJ6fShzhT2avhotn4Zcka6tnt
Correct!
JQttfApK4SeyHwDlI9SXGR50qc10Ail1
```

thu được flag `JQttfApK4SeyHwDlI9SXGR50qc10Ail1`

2.18. Bandit16

```
ssh bandit16@bandit.labs.overthewire.org -p 2220
```

Yêu cầu: Thông tin đăng nhập của level có thể nhận được bằng cách nhập mật khẩu của level trước tới một cổng trên localhost, trong khoảng từ 31000 tới 32000. Trước tiên hãy tìm xem cổng nào đang mở (listening). Sau đó xác định cổng nào giao tiếp SSL và cổng nào không. Sẽ chỉ có một cổng có thông tin cần thiết, các cổng khác đơn giản trả về những gì ta gửi.

Để quét một dải các cổng trên một host, ta sử dụng nmap. Nmap (Network Manager) là công cụ quét mạng, cho phép ta biết thiết bị hay máy tính nào đang hoạt động trên mạng, cũng như các cổng mạng nào đang mở/đóng, xác định dịch vụ nào đang hoạt động trên đó và nhiều thông tin khác nữa.

Cú pháp nmap có dạng:

```
nmap [Scan Type(s)] [Options] {target specification}
```

Trong đó:

- [Scan Type(s)]: Loại quét của nmap, bao gồm các loại:
 - -sS (TCP SYN scan)
 - -sT (TCP connect scan)
 - -sU (UDP scan)
 - -sA (TCP ACK scan)
 - -sN (TCP NULL scan)
 - -sF (TCP FIN scan)
 - -sX (TCP Xmas scan)
- [Options]: Tùy chọn cho lệnh nmap, ví dụ như:
 - -p để quét một hay nhiều port cụ thể
 - -O để xác định hệ điều hành của target
 - -v (verbose) để hiển thị chi tiết hơn
 - -A (run all check) kiểm tra cả phiên bản dịch vụ, OS, default script...
- {target specification}: Địa chỉ IP hoặc tên miền của target được quét.

Trong level của chúng ta, ta cần quét các cổng ở localhost, có port từ 31000 đến 32000.

```
nmap -v -p31000-32000 localhost
```

```
bandit16@bandit:~$ nmap -v -p31000-32000 127.0.0.1
Starting Nmap 7.80 ( https://nmap.org ) at 2023-04-06 08:51 UTC
Initiating Ping Scan at 08:51
Scanning 127.0.0.1 [2 ports]
Completed Ping Scan at 08:51, 0.00s elapsed (1 total hosts)
Initiating Connect Scan at 08:51
Scanning localhost (127.0.0.1) [1001 ports]
Discovered open port 31046/tcp on 127.0.0.1
Discovered open port 31960/tcp on 127.0.0.1
Discovered open port 31691/tcp on 127.0.0.1
Discovered open port 31790/tcp on 127.0.0.1
Discovered open port 31518/tcp on 127.0.0.1
Completed Connect Scan at 08:51, 0.02s elapsed (1001 total ports)
Nmap scan report for localhost (127.0.0.1)
Host is up (0.00010s latency).
Not shown: 996 closed ports
PORT      STATE SERVICE
31046/tcp  open  unknown
31518/tcp  open  unknown
```

Nhận xét: ta có 5 cổng đang mở (listening), là 31046, 31960, 31691, 31790, 31518, đều sử dụng giao thức TCP. Trong này sẽ có một cổng sử dụng SSL. Ta thực hiện scan cụ thể thông tin từng cổng, hoặc toàn bộ các cổng trên:

```
nmap -A -p 31046,31960,31691,31790,31518 localhost
```

```
bandit16@bandit:~$ nmap -A -p 31518 localhost
Starting Nmap 7.80 ( https://nmap.org ) at 2023-04-06 09:03 UTC
Nmap scan report for localhost (127.0.0.1)
Host is up (0.00015s latency).

PORT      STATE SERVICE VERSION
31518/tcp  open  ssl/echo
| ssl-cert: Subject: commonName=localhost
| Subject Alternative Name: DNS:localhost
| Not valid before: 2023-04-05T16:41:34
|_Not valid after: 2023-04-05T16:42:34
```

Ví dụ như trên, ta thấy cổng 31518 đang sử dụng dịch vụ ssl/echo, tức là mã hóa bằng ssl, dịch vụ echo là phản hồi lại thông tin ta gửi. Ta có thể thử kết nối đến cổng này và gửi một vài từ khóa

```
openssl s_client -connect 127.0.0.1:31518
```

```
---
read R BLOCK
JQttfApK4SeyHwDlI9SXGR50qc1OAi11
JQttfApK4SeyHwDlI9SXGR50qc1OAi11
a
Ta
```

Tiếp tục phân tích các cổng được quét, ta phát hiện cổng 31790 có phản hồi một dịch vụ không xác định:

```
nmap done: 1 IP address (1 host up) scanned in 30.00 seconds
bandit16@bandit:~$ nmap -A -p 31790 localhost
Starting Nmap 7.80 ( https://nmap.org ) at 2023-04-06 09:35 UTC
Nmap scan report for localhost (127.0.0.1)
Host is up (0.000083s latency).

PORT      STATE SERVICE      VERSION
31790/tcp  open  ssl/unknown
| fingerprint-strings:
|_  FourOhFourRequest, GenericLines, GetRequest, HTTPOptions, Help, Kerberos, LDAPSearchReq, LPDString, RTSPRequest, SIPOptions, SSLSessionReq, TLSSessionReq, TerminalServerCookie:
|_  Wrong! Please enter the correct current password
|_  ssl-cert: Subject: commonName=localhost
|_  Subject Alternative Name: DNS:localhost
|_  Not valid before: 2023-04-05T16:41:34
|_  Not valid after: 2023-04-05T16:42:34
1 service unrecognized despite returning data. If you know the service/version, please submit the following fingerprint at https://nmap.org/cgi-bin/submit.cgi?new-service :
```

Ta thực hiện gửi thông tin đến cổng này:

```
openssl s_client -connect 127.0.0.1:31790
```

Thực hiện submit flag của bài trước, ta thu được một khóa sshkey để đăng nhập vào bài sau.

```

Extended Master Secret: no
Max Early Data: 0
---
read R BLOCK
JQtTfApK4SeyHwDlI9SXGR50qc10Ail1
Correct!
-----BEGIN RSA PRIVATE KEY-----
MIIEogIBAAKCAQEAvM0kuifmMg6HL2YPI0jon6iWfbp7c3jx34YkYWqUH57SudyJ
imZzeyGC0gtZPGujUSxiJSWI/oTqexh+cAMTSMl0Jf7+BrJ0bArnxd9Y7YT2bRPQ
Ja6Lzb558YW3FZl870RiO+rW4LDCdNd2lUvLE/GL2GwyuKN0K5iCd5TbtJzEkQTu
DSt2mcNn4rhAL+JFr56o4T6z8WWAW18BR6yGrMq7Q/kALHYW30ekePQAzL0VUYbW
JGTi65CxbCnzc/w4+mqQyvmzpwTMAzJTzAzQxNbK2MBGySxDLrjg0LWN6sK7wNX
x0YVztz/zbIkPjfkU1jHS+9EbVNj+D1XF0JuaQIDAQABaoIBABagpxpM1aoLWfvD
KHcj10nqcoBc4oE11aFYQwik7xfW+24pRNUDE6SFth0ar69jp5R1LwD1NhPx3iB1
J9nOM80J0VToum43UOS8YxF8WwhXriYGnc1sskbwpXOUDc9uX4+UESzH22P29ovd
d8WErY0gPxun8pbJLmxkAtWNhpMvfe0050vk9TL5wqbu9AlbssgTcCXkMQnPw9nC

```

Tạo một thư mục để chứa khóa trên ở đường dẫn /tmp (do ta không có quyền tạo file ở pwd)

```

mkdir /tmp/keybandit17
cd /tmp/keybandit17
nano sshkey.private

```

Ta copy đoạn key trên vào file sshkey.private, Ctrl + O để lưu và Ctrl + X để thoát.

```

GNU nano 6.2 sshkey.private *
-----BEGIN RSA PRIVATE KEY-----
MIIEogIBAAKCAQEAvM0kuifmMg6HL2YPI0jon6iWfbp7c3jx34YkYWqUH57SudyJimZzeyGC0gt
-----END RSA PRIVATE KEY-----

```

Tiếp đó, ta cần thay đổi chế độ truy cập cho file sshkey.private này. Ta sử dụng câu lệnh chmod (change mode):

```

chmod 600 sshkey.private

```

Tham số 600 tức là thiết lập mode 6 cho chủ sở hữu owner (chữ số đầu tiên), mode 0 cho group (chữ số thứ hai) và mode 0 cho người dùng khác (chữ số thứ ba).

Trong đó, mỗi mode sẽ là một giá trị nhị phân 3 chữ số, tức là từ 000 (0) đến 111 (7). Với mỗi bit chỉ thị quyền đọc, ghi, thực thi tương ứng.

Mode 6 tương ứng với giá trị nhị phân **110**, tức là người sở hữu có thể đọc (**1**), ghi (**1**) nhưng không có quyền thực thi (**0**).

Ta thực hiện ssh đến máy localhost với người dùng bandit7, sử dụng key vừa tạo. Lưu ý hãy thực hiện tại đường dẫn `/tmp/keybandit17/`:

```
ssh bandit7@localhost -p 2220 -i sshkey.private
```

```
bandit16@bandit:/tmp/keybandit17$ chmod 600 sshkey.private
bandit16@bandit:/tmp/keybandit17$ ssh bandit7@localhost -p2220 -i sshkey.private
The authenticity of host '[localhost]:2220 ([127.0.0.1]:2220)' can't be established.
ED25519 key fingerprint is SHA256:C2ihUBV7ihnV1wUXRb4RrEcLFXC5CX1hmAAM/urerLY.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
```

```
Enjoy your stay!
```

```
bandit17@bandit:~$
```

Thực hiện tương tự bandit, ta tìm được flag trong file `/etc/bandit_pass/bandit17`

```
bandit17@bandit:~$ cat /etc/bandit_pass/bandit17
VwOSWtCA7lRKkTfbr2IDh6awj9RNZM5e
```

VwOSWtCA7lRKkTfbr2IDh6awj9RNZM5e

2.19. Bandit17

```
ssh bandit17@bandit.labs.overthewire.org -p 2220
```

Yêu cầu: Có 2 tệp trong đường dẫn gốc: `passwords.old` và `passwords.new`. Flag cho level này ở file `passwords.new` và đó là dòng duy nhất khác biệt giữa `passwords.old` và `passwords.new`

Ta sử dụng câu lệnh `diff` để tìm sự khác biệt giữa nội dung 2 tệp:

```
diff passwords.new passwords.old
```

```
bandit17@bandit:~$ diff passwords.new passwords.old
42c42
< hga5tuuCLF6fFzUpnagiMN8ssu9LFrdg
---
> f9wS9ZUDvZoo3PooHgYuuWdawDFvGld2
```

Dòng trên tương ứng với dòng thay đổi ở tệp `passwords.new`, là flag ta cần `hga5tuuCLF6fFzUpnagiMN8ssu9LFrdg`

2.20. Bandit18

```
ssh bandit18@bandit.labs.overthewire.org -p 2220
```

Yêu cầu: Flag được lưu trong tệp readme ở đường dẫn gốc. Đen cái là ai đó đã sửa tệp .bashrc để ta tự động bị đăng xuất khi đăng nhập thông qua ssh.

Hình dưới đây hiển thị thông báo đăng xuất ngay sau khi đăng nhập thành công:

```

For support, questions or comments, contact us on discord or IRC.

Enjoy your stay!

Byebye !
Connection to bandit.labs.overthewire.org closed.

```

Đây là do khi ssh tới server, server sẽ load tệp `./bashrc` để thực thi các dòng lệnh khởi tạo. Ta có thể bypass bằng cách không load tệp trên thông qua một trong các câu lệnh sau:

```
ssh bandit18@bandit.labs.overthewire.org -p 2220 "bash
--noprofile"
```

```
ssh bandit18@bandit.labs.overthewire.org -p 2220 -T
```

```
anhnt0093@Internet-Doi2:~$ ssh bandit18@bandit.labs.overthewire.org -p 2220 -T
```



```
This is an OverTheWire game server.  
More information on http://www.overthewire.org/wargames  
  
bandit18@bandit.labs.overthewire.org's password:  
ls  
readme  
cat readme  
awhqfNnAbc1naukrpqDYcF95h7HoMTrC
```


Hoặc ta cũng có thể ssh và gửi trực tiếp câu lệnh đọc file readme

```
ssh bandit18@bandit.labs.overthewire.org -p 2220 "cat
readme"
```

```
anhnt0093@Internet-Doi2:~$ ssh bandit18@bandit.labs.overthewire.org -p 2220 "cat readme"
```



This is an OverTheWire game server.
More information on <http://www.overthewire.org/wargames>

```
bandit18@bandit.labs.overthewire.org's password:  
awhqfNnAbc1naukrpqDYcF95h7HoMTrC
```

Thu được flag awghfNnAbc1naukrpqDYcF95h7HoMTrC

2.21. Bandit19

```
ssh bandit19@bandit.labs.overthewire.org -p 2220
```

Yêu cầu: Để truy cập level tiếp theo, ta chạy tệp setuid binary ở thư mục gốc. Thực thi không đối số để biết cách sử dụng. Flag có thể tìm thấy ở thư mục /etc/bandit pass.

setuid binary là tệp thực thi được thiết lập với quyền sở hữu của người dùng khác. setuid là “set user id”, tệp tin setuid sẽ được thực thi với quyền của chủ sở hữu tệp tin thay vì quyền của người dùng hiện tại. Tệp này cho phép thực thi các chương trình có quyền đặc biệt mà không cần đăng nhập tài khoản của chủ sở hữu chương trình đó.

```
bandit19@bandit:~$ ls
bandit20-do
bandit19@bandit:~$ ./bandit20-do
Run a command as another user.
Example: ./bandit20-do id
bandit19@bandit:~$
```

Run a command as another user: tức là câu lệnh này sẽ có thể thực thi lệnh dưới vai trò của user khác (cụ thể ở đây là bandit20). Tức là ta có thể đọc được tệp chứa mật khẩu của bandit20 ở đường dẫn /etc/bandit_pass thông qua file trên:

```
./bandit20-do cat /etc/bandit_pass/bandit20
```

Thu được: VxCazJaVyki6W36BkBU0mJTCM8rR95XT

2.22. Bandit20

```
ssh bandit20@bandit.labs.overthewire.org -p 2220
```

Yêu cầu: Có một tệp `setuid` binary ở thư mục gốc thực hiện việc sau: thiết lập kết nối tới localhost với số hiệu [port] được nhận vào thông qua tham số truyền vào. Sau đó tệp này sẽ đọc một dòng văn bản từ kết nối, so sánh với flag ở level trước. Nếu đúng thì sẽ nhận được flag ở level hiện tại.

```
bandit20@bandit:~$ ./suconnect
Usage: ./suconnect <portnumber>
This program will connect to the given port on localhost using TCP. If it receives t
he correct password from the other side, the next password is transmitted back.
bandit20@bandit:~$
```

Trước hết ta phải thiết lập một [port] listener bằng netcat, sau đó sử dụng suconnect để truy cập đến cổng đó, rồi ta truyền flag ở level trước để nhận flag level hiện tại.

Để thiết lập một [port] listener, ta thực hiện câu lệnh

```
nc -nlvp [port]
```

`nc -nlvp [port]` là lệnh để lắng nghe kết nối đến một cổng (port) cụ thể trên máy tính sử dụng netcat (nc). Chi tiết các tham số:

- n: chạy ở chế độ không dùng DNS.
- l: lắng nghe kết nối đến.
- v: in ra thông tin chi tiết khi có kết nối.
- p: xác định số cổng (port) sử dụng cho kết nối.

Chọn một port không phổ biến, chẳng hạn 9876

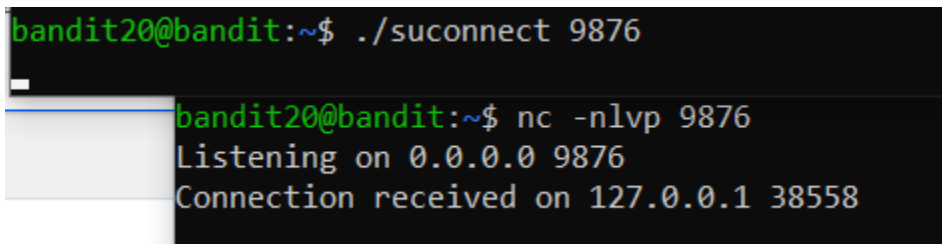
```
nc -nlvp 9876
```

```
bandit20@bandit:~$ nc -nlvp 9876
Listening on 0.0.0.0 9876
```

Tuy nhiên, ta để ý rằng nếu ta thiết lập một port listening ở terminal nào, terminal đó sẽ phụ trách quan sát giao tiếp ở port đó, do đó ta không thực hiện được các câu lệnh khác. Vì thế ta sẽ cần thiết lập thêm một kết nối đến host ở terminal khác. Kết nối thứ hai này sẽ thực hiện nhiệm vụ sử dụng suconnect để liên lạc tới port 9876 trên localhost.

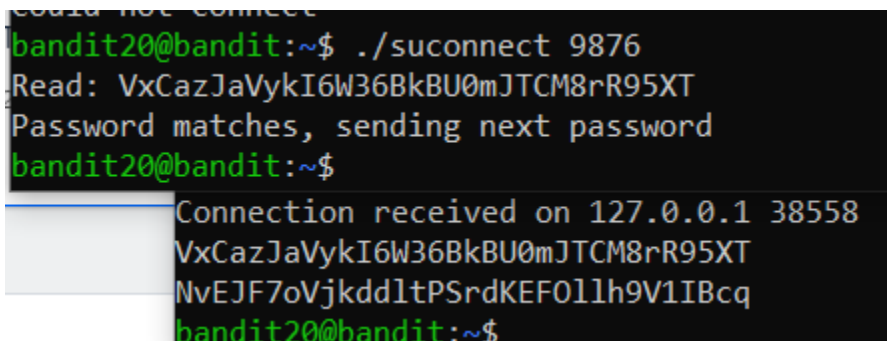
Ở terminal thứ hai, ta gõ:

```
./suconnect 9876
```



```
bandit20@bandit:~$ ./suconnect 9876
bandit20@bandit:~$ nc -nlvp 9876
Listening on 0.0.0.0 9876
Connection received on 127.0.0.1 38558
```

Tiếp đó, ở Terminal 1, ta điền flag cũ. suconnect sẽ so sánh giá trị flag nhận được xem có đúng không, nếu đúng sẽ trả về flag của level này.



```
bandit20@bandit:~$ ./suconnect 9876
Read: VxCazJaVyKI6W36BkBU0mJTCM8rR95XT
Password matches, sending next password
bandit20@bandit:~$
Connection received on 127.0.0.1 38558
VxCazJaVyKI6W36BkBU0mJTCM8rR95XT
NvEJF7oVjkddlTPSrdKEF01lh9V1IBcq
bandit20@bandit:~$
```

Thu được flag mới: NvEJF7oVjkddlTPSrdKEF01lh9V1IBcq

2.23. Bandit 21

```
ssh bandit21@bandit.labs.overthewire.org -p 2220
```

Yêu cầu: một chương trình đang được chạy tự động với theo chu kỳ thời gian đều đặn bởi cron (Chronograph), một công cụ lập lịch trình dựa trên thời gian. Hãy kiểm tra file cấu hình ở đường dẫn /etc/cron.d/ và xem câu lệnh nào được thực thi trong đó.

Xem trong thư mục /etc/cron.d có gì:

```
ls /etc/cron.d
```

Vì level này là bandit21->bandit22, nên ta mở cronjob_bandit22

```
cat /etc/cron.d/cronjob_bandit22
```

Thấy chạy tệp /usr/bin/cronjob_bandit22.sh, ta mở tiếp tệp này

```
cat /usr/bin/cronjob_bandit22.sh
```

Trong này có cấp quyền chmod cho tệp /tmp/t706lds9S0RqQh9aMcz6ShpAoZKF7fgv, ta lại đọc tiếp nội dung tệp:

```
cat /tmp/t706lds9S0RqQh9aMcz6ShpAoZKF7fgv
```

```
bandit21@bandit:~$ ls /etc/cron.d
cronjob_bandit15_root  cronjob_bandit22  cronjob_bandit24      e2scrub_all  sysstat
cronjob_bandit17_root  cronjob_bandit23  cronjob_bandit25_root  otw-tmp-dir
bandit21@bandit:~$ cat /etc/cron.d/cronjob_bandit22
@reboot bandit22 /usr/bin/cronjob_bandit22.sh &> /dev/null
* * * * * bandit22 /usr/bin/cronjob_bandit22.sh &> /dev/null
bandit21@bandit:~$ cat /usr/bin/cronjob_bandit22.sh
#!/bin/bash
chmod 644 /tmp/t706lds9S0RqQh9aMcz6ShpAoZKF7fgv
cat /etc/bandit_pass/bandit22 > /tmp/t706lds9S0RqQh9aMcz6ShpAoZKF7fgv
bandit21@bandit:~$ cat /tmp/t706lds9S0RqQh9aMcz6ShpAoZKF7fgv
WdDozAdTM2z9DiFEQ2mGlwnGmfj4EZff
bandit21@bandit:~$
```

Thu được flag WdDozAdTM2z9DiFEQ2mGlwnGmfj4EZff

2.24. Bandit22

```
ssh bandit22@bandit.labs.overthewire.org -p 2220
```

Yêu cầu: tương tự như bài trên

Ta mở danh sách tệp trong /etc/cron.d

```
ls /etc/cron.d
```

Đọc tệp cronjob_bandit23

```
cat /etc/cron.d/cronjob_bandit23
```

```
bandit22@bandit:~$ cat /etc/cron.d/cronjob_bandit23
@reboot bandit23 /usr/bin/cronjob_bandit23.sh &> /dev/null
* * * * * bandit23 /usr/bin/cronjob_bandit23.sh &> /dev/null
```

Đọc tệp /usr/bin/cronjob_bandit23.sh

```
cat /usr/bin/cronjob_bandit23.sh
```

```
bandit22@bandit:~$ cat /usr/bin/cronjob_bandit23.sh
#!/bin/bash

myname=$(whoami)
mytarget=$(echo I am user $myname | md5sum | cut -d ' ' -f 1)

echo "Copying passwordfile /etc/bandit_pass/$myname to /tmp/$mytarget"

cat /etc/bandit_pass/$myname > /tmp/$mytarget
```

Đoạn bash dưới là đoạn code bash điển hình. `$(whoami)` chính là tên user. Nếu ta in thử ở terminal hiện tại thì sẽ in ra bandit22:

```
bandit22@bandit:~$ echo $(whoami)
bandit22
```

Tuy nhiên lưu ý rằng đoạn lệnh trên sẽ được user bandit23 thực thi, nên giá trị thực tế sẽ là bandit23, như vậy sau dòng lệnh nay ta sẽ có biến `$myname` chứa giá trị bandit23

Dòng tiếp theo:

```
mytarget = $(echo I am user $myname | md5sum | cut -d ' ' -f 1)
```

nghĩa là:

- `echo`: in ra dòng lệnh
- `I am user $myname` tương ứng với chuỗi “I am user bandit23”
- `| md5sum` tức là tính giá trị băm md5sum của chuỗi trước: “I am user bandit23”
- `| cut -d ' ' -f 1` tức là cắt chuỗi trước (chuỗi đã băm) theo kí tự khoảng trắng ' ', sau đó lấy phần tử đầu tiên (`-f 1`), tức là lấy chuỗi băm.

Ta thử:

```
echo I am user bandit23
echo I am user bandit23 | md5sum
echo I am user bandit23 | md5sum | cut -d ' ' -f 1
```

```
bandit22@bandit:~$ echo I am user bandit23
I am user bandit23
bandit22@bandit:~$ echo I am user bandit23 | md5sum
8ca319486bfbbc3663ea0fbe81326349 -
bandit22@bandit:~$ echo I am user bandit23 | md5sum | cut -d ' ' -f 1
8ca319486bfbbc3663ea0fbe81326349
```

Chuỗi băm thu được là 8ca319486bfbbc3663ea0fbe81326349, được lưu vào biến \$mytarget.

Dòng lệnh tiếp theo:

```
echo "Copying passwordfile /etc/bandit_pass/$myname to
/tmp/$mytarget"
```

Ta biết \$myname có giá trị bandit23, \$mytarget có giá trị 8ca319486bfbbc3663ea0fbe81326349, đoạn lệnh trên thực tế có ý nghĩa là in ra màn hình đoạn sau:

```
Copying passwordfile /etc/bandit_pass/bandit23 to
/tmp/8ca319486bfbbc3663ea0fbe81326349
```

Tiếp đó:

```
cat /etc/bandit_pass/$myname > /tmp/$mytarget
```

Tức là đọc nội dung của tập tin có đường dẫn /etc/bandit_pass/\$myname và ghi vào tập tin có tên là \$mytarget trong thư mục /tmp/

Sau khi thêm các giá trị biến vào, câu lệnh trở thành:

```
cat /etc/bandit_pass/bandit23 >
/tmp/8ca319486bfbbc3663ea0fbe81326349
```

Câu lệnh trên có nghĩa đã copy nội dung của tệp /etc/bandit_pass/bandit23 (nơi chứa flag) sang tệp /tmp/8ca319486bfbbc3663ea0fbe81326349.

Ta mở tệp /tmp/8ca319486bfbbc3663ea0fbe81326349 để lấy flag:

```
cat /tmp/8ca319486bfbbc3663ea0fbe81326349
```

```
bandit22@bandit:~$ cat /tmp/8ca319486fbfcc3663ea0fbe81326349
QYw0Y2aiA672PsMmh9puTQuhoz8SyR2G
```

Thu được: QYw0Y2aiA672PsMmh9puTQuhoz8SyR2G

2.25. Bandit23

```
ssh bandit23@bandit.labs.overthewire.org -p 2220
```

Yêu cầu: như level trước, nhưng ta sẽ phải tự viết bash script. Và cũng lưu ý rằng server sẽ tự động xóa đoạn mã lệnh sau khi viết, nên tốt nhất hãy lưu đoạn mã lệnh vào đâu đó.

Thực hiện tương tự như level trước để kiểm tra yêu cầu trong cron.d:

```
bandit23@bandit:~$ cat /etc/cron.d/cronjob_bandit24
@reboot bandit24 /usr/bin/cronjob_bandit24.sh &> /dev/null
* * * * * bandit24 /usr/bin/cronjob_bandit24.sh &> /dev/null
bandit23@bandit:~$ cat /usr/bin/cronjob_bandit24.sh
#!/bin/bash

myname=$(whoami)

cd /var/spool/$myname/foo
echo "Executing and deleting all scripts in /var/spool/$myname/foo:"
for i in * .*;
do
    if [ "$i" != "." -a "$i" != ".." ];
    then
        echo "Handling $i"
        owner="$(stat --format "%U" ./$i)"
        if [ "${owner}" = "bandit23" ]; then
            timeout -s 9 60 ./$i
        fi
        rm -f ./$i
    fi
done
```

Đoạn script trên thực hiện:

```
myname = $(whoami) (=bandit24)
```

```
cd /var/spool/$myname/foo (= cd/var/spool/bandit24/foo)
```

Sau đó, với vòng lặp trên có nghĩa là:

```
for i in * .*: duyệt tất cả các tệp có trong thư mục
```

```
if [ "$i" != "." -a "$i" != ".." ]; nếu tên của tệp hoặc thư  
mục đang được xử lý không phải là "." hoặc ".." thì in ra thông báo "Handling  
[tên của tệp/thư mục]"
```

```
owner="$(stat --format "%U" ./$i)": lưu tên chủ sở hữu của  
tệp/thư mục vào biến owner.
```

```
if [ "${owner}" = "bandit23" ]; then timeout -s 9 60  
./$i: Nếu chủ sở hữu là bandit23 (tức là user đang sử dụng) thì sẽ thực thi tệp  
./$i, với thời gian giới hạn 60s, và tùy chọn -s 9 để dừng tệp thực thi nếu tệp  
không hoàn thành trong thời gian giới hạn đó.
```

Cuối cùng sẽ xóa tệp/thư mục bằng lệnh `rm -f ./$i`.

Kết luận: Đoạn lệnh trên sẽ thực thi các tệp trong thư mục `/var/spool/bandit24/foo` do user `bandit23` sở hữu, sau đó xóa các tệp trên.

Mục tiêu của ta hiện nay sẽ là tạo một file để lưu trữ mật khẩu ở `/tmp`, sau đó viết một đoạn script ở thư mục `/var/spool/bandit24/foo` với mục đích in ra thông tin flag nằm ở `/etc/bandit_pass/bandit24` sang file lưu trữ mật khẩu trên.

Trước hết, ta tạo file `flag_bandit24` và cấp mọi quyền để người dùng nào cũng in được ra file này. ta tạo ở thư mục gốc `/tmp`, cụ thể ở đây ta tạo `flag_bandit24` trong đường dẫn `/tmp/get_flag_bandit24`:

```
mkdir /tmp/get_flag_bandit24
```

```
chmod 777 -R /tmp/get_flag_bandit24
```

```
touch /tmp/get_flag_bandit24/flag_bandit24
```

```
chmod 777 /tmp/get_flag_bandit24/flag_bandit24
```



```
bandit23@bandit:~$ mkdir /tmp/get_flag_bandit24
bandit23@bandit:~$ chmod 777 -R /tmp/get_flag_bandit24
bandit23@bandit:~$ touch /tmp/get_flag_bandit24/flag_bandit24
bandit23@bandit:~$ chmod 777 /tmp/get_flag_bandit24/flag_bandit24
bandit23@bandit:~$ ls -la /tmp/get_flag_bandit24
total 144
drwxrwxrwx  2 bandit23 bandit23  4096 Apr  7 06:47 .
drwxrwx-wt 529 root      root    139264 Apr  7 06:47 ..
-rwxrwxrwx  1 bandit23 bandit23     0 Apr  7 06:47 flag_bandit24
```

Tiếp đến, ta sẽ tạo một file bash để lấy dữ liệu từ /etc/bandit_pass/bandit24 sang file flag_bandit24 của chúng ta. Tạo file bash này ở trong thư mục /tmp/get_flag_bandit24 để có nơi lưu trữ.

```
#!/bin/bash
cat /etc/bandit_pass/bandit24 >
/tmp/get_flag_bandit24/flag_bandit24
```

```
bandit23@bandit: ~
GNU nano 6.2 /tmp/get_flag_bandit24/bandit_24.sh
#!/bin/bash
cat /etc/bandit_pass/bandit24 > /tmp/get_flag_bandit24/flag_bandit24
```

Sau đó, ta cấp quyền thực thi file này cho mọi user và copy file bash này vào thư mục /var/spool/bandit24/foo, chờ khoảng 1 phút để hệ thống tự động thực thi file bash.

```
chmod 777 /tmp/get_flag_bandit24/bandit_24.sh
cp /tmp/get_flag_bandit24/bandit_24.sh
/var/spool/bandit24/foo/bandit_24.sh
```

```
bandit23@bandit:~$ chmod 777 /tmp/get_flag_bandit24/bandit_24.sh
bandit23@bandit:~$ cp /tmp/get_flag_bandit24/bandit_24.sh /var/spool/bandit24/foo/bandit_24.sh
```

Thu được flag: VAfGXJ1PBSsPSnvsjI8p759leLZ9GGar

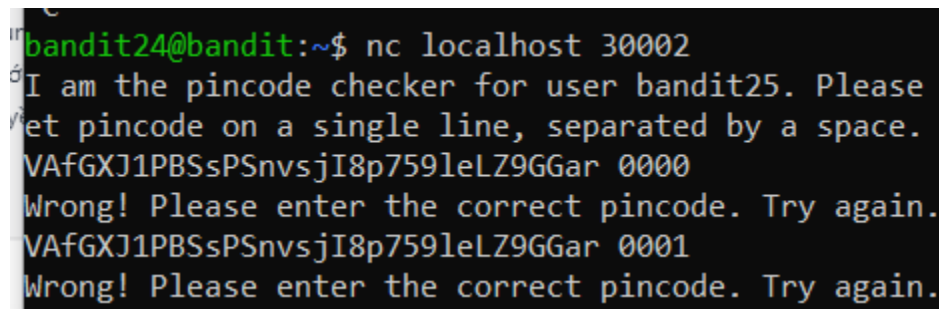
```
bandit23@bandit:~$ cat /tmp/get_flag_bandit24/flag_bandit24
VAfGXJ1PBSsPSnvsjI8p759leLZ9GGar
```

2.26. Bandit24

```
ssh bandit24@bandit.labs.overthewire.org -p 2220
```

Yêu cầu: Có một chương trình Daemon đang chạy trên cổng 30002 và sẽ cung cấp cho chúng ta flag của level bandit25, nếu ta gửi flag bandit24 và một mã pincode có 4 chữ số. Sẽ có tổng cộng 10.000 khả năng của pincode (từ 0000 đến 9999). Ta không có cách nào khác ngoài thử nghiệm tất cả trường hợp cho đến khi ra đáp án. Đây gọi là vét cạn (brute force).

Thử nghiệm:



```
bandit24@bandit:~$ nc localhost 30002
I am the pincode checker for user bandit25. Please
let pincode on a single line, separated by a space.
VAfGXJ1PBSsPSnvsjI8p759leLZ9GGar 0000
Wrong! Please enter the correct pincode. Try again.
VAfGXJ1PBSsPSnvsjI8p759leLZ9GGar 0001
Wrong! Please enter the correct pincode. Try again.
```

Như vậy, mục tiêu của ta là gửi một loạt các request đến cổng 30002 có dạng:

`[flag_bandit24] [pin]`

Trong đó,

- `[flag_bandit24]` là flag của level trước:
`VAfGXJ1PBSsPSnvsjI8p759leLZ9GGar`
- `[pin]` là các giá trị từ 0000 đến 9999

Ta sẽ lưu toàn bộ các request này vào một tệp đặt tên là `request.txt`, và gửi nội dung tệp đến cổng 30002, rồi ta lấy tất cả các phản hồi lưu vào tệp đặt tên là `response.txt`

Viết đoạn bash `request.sh` để lưu toàn bộ request vào file `request.txt`:

```
mktemp -d
cd /tmp/tmp.JX36Fhgixz
nano request.sh
```

```
bandit24@bandit: /tmp/tmp.JX36Fhgixz
GNU nano 6.2 request.sh
#!/bin/bash

for i in {0000..9999}
do
    echo VAfGXJ1PBSsPSnvsjI8p759leLZ9GGar $i >> request.txt
done
```

Cấp quyền và thực thi đoạn lệnh

```
chmod 777 request.sh
./request.sh
```

Mở thử file request.txt và thấy ta đã lưu đoạn lệnh request thành công:

```
cat request.txt
```

```
VAfGXJ1PBSsPSnvsjI8p759leLZ9GGar 9983
VAfGXJ1PBSsPSnvsjI8p759leLZ9GGar 9984
VAfGXJ1PBSsPSnvsjI8p759leLZ9GGar 9985
VAfGXJ1PBSsPSnvsjI8p759leLZ9GGar 9986
VAfGXJ1PBSsPSnvsjI8p759leLZ9GGar 9987
VAfGXJ1PBSsPSnvsjI8p759leLZ9GGar 9988
VAfGXJ1PBSsPSnvsjI8p759leLZ9GGar 9989
VAfGXJ1PBSsPSnvsjI8p759leLZ9GGar 9990
VAfGXJ1PBSsPSnvsjI8p759leLZ9GGar 9991
VAfGXJ1PBSsPSnvsjI8p759leLZ9GGar 9992
VAfGXJ1PBSsPSnvsjI8p759leLZ9GGar 9993
VAfGXJ1PBSsPSnvsjI8p759leLZ9GGar 9994
VAfGXJ1PBSsPSnvsjI8p759leLZ9GGar 9995
```

Ta tiếp tục gửi tất cả đoạn request này đến cổng 30001 thông qua lệnh netcat, đồng thời lưu mật khẩu vào file response.txt:

```
cat request.txt | nc localhost 30002 > response.txt
```

In ra dòng có mật khẩu đúng nhờ sort và in ra dòng duy nhất uniq -u

```
bandit24@bandit:/tmp/tmp.JX36Fhgixz$ ./request.sh
bandit24@bandit:/tmp/tmp.JX36Fhgixz$ cat request.txt | nc localhost 30002 > response.txt
bandit24@bandit:/tmp/tmp.JX36Fhgixz$ cat response.txt | sort | uniq -u

Correct!
Exiting.
I am the pincode checker for user bandit25. Please enter the password for user bandit24 and
gle line, separated by a space.
The password of user bandit25 is p7TaowMYrmu230l8hiZh9UvD009hpx8d
bandit24@bandit:/tmp/tmp.JX36Fhgixz$
```

Thu được flag: p7TaowMYrmu230l8hiZh9UvD009hpx8d

2.27. Bandit25

```
ssh bandit25@bandit.labs.overthewire.org -p 2220
```

Yêu cầu: Truy cập vào bandit26 không khó, tuy nhiên Shell cho bandit26 không nằm ở /bin/bash mà ở vị trí khác. Hãy tìm và bẻ khóa.

Đúng như vậy, truy cập vào bandit26 không khó, private key để ssh nằm ở ngay đường dẫn ngoài:

Enjoy your stay!

```
bandit25@bandit:~$ ls
```

```
bandit26.sshkey
```

```
bandit25@bandit:~$ cat bandit26.sshkey
```

```
-----BEGIN RSA PRIVATE KEY-----
```

```
MIIEpQIBAAKCAQEApis2AuoooEqeYWamtwX2k5z9uU1Af12F8VyXQqbv/LTrIwdW  
pTfaeRHXzr0Y0a50e3GB/+W2+PReif+bPZlZTY1XFwpk+DiHk1kmL0moEW8HJuT9  
/5XbnpjSzn0eEAFax20copjrzVqdBJQerkj0puv3UXY07AskgyD5XepwGAlJOG  
xZsMq1oZqQ0W29aBtfykuGie2bxroRjuAPrYM4o3MMmtlNE5fC4G9Ihq0eq73MDi  
1ze6d2jIGce873qxn308BA2qhRPJNEbnPev5gI+5tU+UxebW8KLbk0EhoXB953Ix  
3lg0IrT9Y6skRjsMSFmC6WN/07ovu8QzGqxdywIDAQABAoIBAAaXoETtVT9GtpHW  
qLaKHgYtLE01tFOhInWyolyZgL4inuRRva3CivVEWK6TcnDyI1NL4MfcerehwGi  
il4fQFvLR7E6UFcopvhJiSJHicvPQ9FfNFR3dYcNOQ/IFvE73bEqMwSISPwie16w  
e1DjF3C7jHaS1s9PJfWFN982aub1L/yLbJP+ou3ifdljS7QzjWZA8NRiMwmBGPIh  
Yq8weR3jIVQ13ndEYx07Cr/wXXebZw1P6CPZb67rBy0jg+366mxQbDZIwZYEaUME  
zY5izFc1r/kKj4s7NTRkC76Yx+rTNP5+BX+JT+rgz5aoQq8ghMw43NYwxjXym/MX  
c8X8g0ECgYEA1crBUAR1gSkM+5mGjjoFLJKrFP+IhUHFh25qGI4Dcxxh1f3M531e  
wF1rkp5SJnHRFm9IW3gM1JoF0PQxI5aXHRGHphwPeKnsQ/xQBRWCeYpqTme9amJV  
tD3aDHkpIhYxkNxqol5gDCAt6tdFSxqPaNfdfsfaAOXiKGrQESUjIBcCgYEAxvmI  
2ROJsBXaiM4Iyg9hUpjZIn8TW2U1H76pojFG6/KBd1NcnW3fu0ZUU790wAu7QbbU  
i7pieeqCqSYcZsmkhN0vbdx54A6NNCR2btc+si6pD0e1jdsGdXISDRHFb9QxjZCj  
6xzWMNvb5n1yUb9w9nfN1PZZATfUsOV+FY8CbG0CgYEAifkTLwfhaqZyLk2huTSWm  
pzB0ltWfDpj22MNqVzR3h3d+sHLeJVjPzIe9396rF8KGdNsWsG1WpnJMZKDjgZsz  
JQBmMc6UMYRARVP1dIKANN4eY0FSHFebHcqXLho0mXOUTXe37DwfZza5V90ify3  
JquBd8uUptW1Ue41H4t/ErsCgYEArc5FYtF1QXI1fcDz3oUGz16itUZpgz1b71nd  
1cbTm8EupCwWR5I1j+IEQU+JTUQyI1nwWcnKwZI+5kBbKNJUu/mLsRyY/UXYxEZh  
ihNk1m94373kV1US/0D17UDc0ha7iz9Yn/C3dT/R1woTw5mP3Ux0CizEsnNK0Se
```

Sử dụng key này để thử truy cập bandit26:

```
bandit25@bandit:~$ ssh bandit26@localhost -p 2220 -i bandit26.sshkey
```

```
The authenticity of host '[localhost]:2220 ([127.0.0.1]:2220)' can't be established.
```

```
ED25519 key fingerprint is SHA256:C2ihUBV7ihNv1wUXRb4RrEcLfXC5CX1hmAAM/urerLY.
```

```
This key is not known by any other names
```

```
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
```

Tuy nhiên ngay sau truy cập vào bandit26, ta bị đá ra ngoài:


```
cat /etc/passwd | grep bandit26
```

```
drifter8:x:19008:19008:drifter level 8:/home/drifter8:/bin/bash
drifter9:x:19009:19009:drifter level 9:/home/drifter9:/bin/bash
formulaone0:x:20000:20000:formulaone level 0:/home/formulaone0:/bin/bash
formulaone1:x:20001:20001:formulaone level 1:/home/formulaone1:/bin/bash
formulaone2:x:20002:20002:formulaone level 2:/home/formulaone2:/bin/bash
formulaone3:x:20003:20003:formulaone level 3:/home/formulaone3:/bin/bash
formulaone5:x:20005:20005:formulaone level 5:/home/formulaone5:/bin/bash
formulaone6:x:20006:20006:formulaone level 6:/home/formulaone6:/bin/bash
krypton1:x:8001:8001:krypton level 1:/home/krypton1:/bin/bash
krypton2:x:8002:8002:krypton level 2:/home/krypton2:/bin/bash
krypton3:x:8003:8003:krypton level 3:/home/krypton3:/bin/bash
krypton4:x:8004:8004:krypton level 4:/home/krypton4:/bin/bash
krypton5:x:8005:8005:krypton level 5:/home/krypton5:/bin/bash
krypton6:x:8006:8006:krypton level 6:/home/krypton6:/bin/bash
krypton7:x:8007:8007:krypton level 7:/home/krypton7:/bin/bash
bandit25@bandit:~$ cat /etc/passwd | grep bandit26
bandit26:x:11026:11026:bandit level 26:/home/bandit26:/usr/bin/showtext
bandit25@bandit:~$
bandit25@bandit:~$
bandit25@bandit:~$
bandit25@bandit:~$
bandit25@bandit:~$
bandit25@bandit:~$
bandit25@bandit:~$
bandit25@bandit:~$
```

Ở các kết nối khác đều
trở vào /bin/bash, riêng bandit26
trở vào /usr/bin/showtext

chương trình đc chạy khi kết nối
thành công

id user
password
id group
username
comment
đường dẫn đến thư mục
gốc của ng dùng

Ta thu được các giá trị:

```
bandit26:x:11026:11026:bandit level  
26:/home/bandit26:/usr/bin/showtext
```

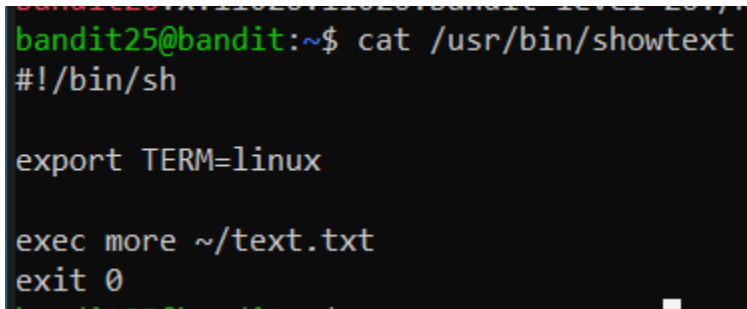
Trong file `/etc/passwd`, mỗi dòng tương ứng với một tài khoản người dùng. Các trường dữ liệu được phân tách bởi dấu `:`. Dưới đây là giải thích các trường dữ liệu cho tài khoản `bandit26`:

- `bandit26`: Tên đăng nhập của tài khoản người dùng.
- `x`: Mật khẩu được mã hóa. Trong trường hợp này, `x` chỉ đơn giản là đại diện cho mật khẩu được lưu trữ trong tệp `/etc/shadow`.
- `11026`: UID (User ID) của tài khoản người dùng. Đây là một số nguyên duy nhất được gán cho mỗi tài khoản người dùng.
- `11026`: GID (Group ID) của tài khoản người dùng. Đây là một số nguyên duy nhất được gán cho mỗi nhóm người dùng.
- `bandit level 26`: Tên hiển thị hoặc chú thích về tài khoản người dùng.
- `/home/bandit26`: Đường dẫn đến thư mục gốc của tài khoản người dùng. Khi kết nối thành công thì thư mục hiển thị sẽ là thư mục này

- `/usr/bin/showtext`: Chương trình được chạy khi kết nối thành công

Câu lệnh trên tức là nếu ta SSH sang bandit26, máy sẽ chạy đoạn `/usr/bin/showtext`. Ta mở đoạn này lên xem

```
cat /usr/bin/showtext
```



```
bandit25@bandit:~$ cat /usr/bin/showtext
#!/bin/sh

export TERM=linux

exec more ~/text.txt

exit 0
```

Trong đó:

`export TERM=linux` là gán giá trị linux cho biến TERM

`exec more ~/text.txt`

- Chạy lệnh `more` với tham số là tệp `~/text.txt`
- `more` là lệnh để hiển thị thông tin tệp (tương tự `cat`), nhưng hiển thị dạng trang tùy thuộc vào kích thước của terminal. Tức là nếu nội dung tệp quá dài so với kích thước terminal thì `more` sẽ hiển thị từng phần một.
- `exit 0` là lệnh thoát, đây chính là lí do tại sao khi ta kết nối đến bandit26, sau khi hiển thị đoạn text giới thiệu thì ta bị ngắt kết nối ngay lập tức.

Tức là, để có thể làm được gì, ta sẽ phải cố gắng xử lí trong khoảng từ khi kết nối được đến trước khi câu lệnh `exit(0)` được kích hoạt. Trong số 2 câu lệnh ở trước là `export` và `more`, thì `export` về cơ bản không thể ngắt được (đây chỉ là phép gán giá trị cho biến).

Với lệnh `more`, ta có thể đọc hướng dẫn sử dụng lệnh bằng

```
man more
```



```

MORE(1)                                User Commands                                MORE(1)

NAME
    more - file perusal filter for crt viewing

SYNOPSIS
    more [options] file ...

DESCRIPTION
    more is a filter for paging through text one screenful at a time. This
    version is especially primitive. Users should realize that less(1)
    provides more(1) emulation plus extensive enhancements.

OPTIONS
    Options are also taken from the environment variable MORE (make sure to
    precede them with a dash (-)) but command-line options will override
    those.

    -d --silent

```

Kéo xuống dưới, có một đoạn về COMMANDS:

```

!command or :!command
    Execute command in a subshell.

v
    Start up an editor at current line. The editor is taken from the
    environment variable VISUAL if defined, or EDITOR if VISUAL is not
    defined, or defaults to vi(1) if neither VISUAL nor EDITOR is
    defined.

```

Đoạn trên miêu tả rằng ta có thể thực thi các câu lệnh trong một subshell! Tức là một shell phụ ngay trong khi đang thực hiện lệnh more bằng cú pháp `![tên_câu_lệnh]`. Chẳng hạn `!ls` thì sẽ thực hiện lệnh `ls`.

Ngoài ra ta có thể mở vi editor bằng câu lệnh `v`. Lưu ý là ta có thể chạy các câu lệnh script trong vi editor.

Vấn đề đặt ra là khi ssh, ta không được “dừng” để thực hiện câu lệnh mà ngay lập tức bị đá ra khỏi server. Vậy phải làm gì?

Nhớ lại rằng câu lệnh more sẽ hiển thị phân trang theo KÍCH THƯỚC của terminal so với số lượng text. Số lượng text không thay đổi được thì ta sẽ thay đổi kích thước của Terminal. Hãy thu nhỏ terminal lại và thử ssh lại một lần nữa.

```
bandit25@bandit: ~  
bandit25@bandit:~$ ssh bandit26@localh  
ost -p 2220 -i bandit26.sshkey
```

Eureka, chúng ta đã dừng lại để thực hiện các câu lệnh của man được:

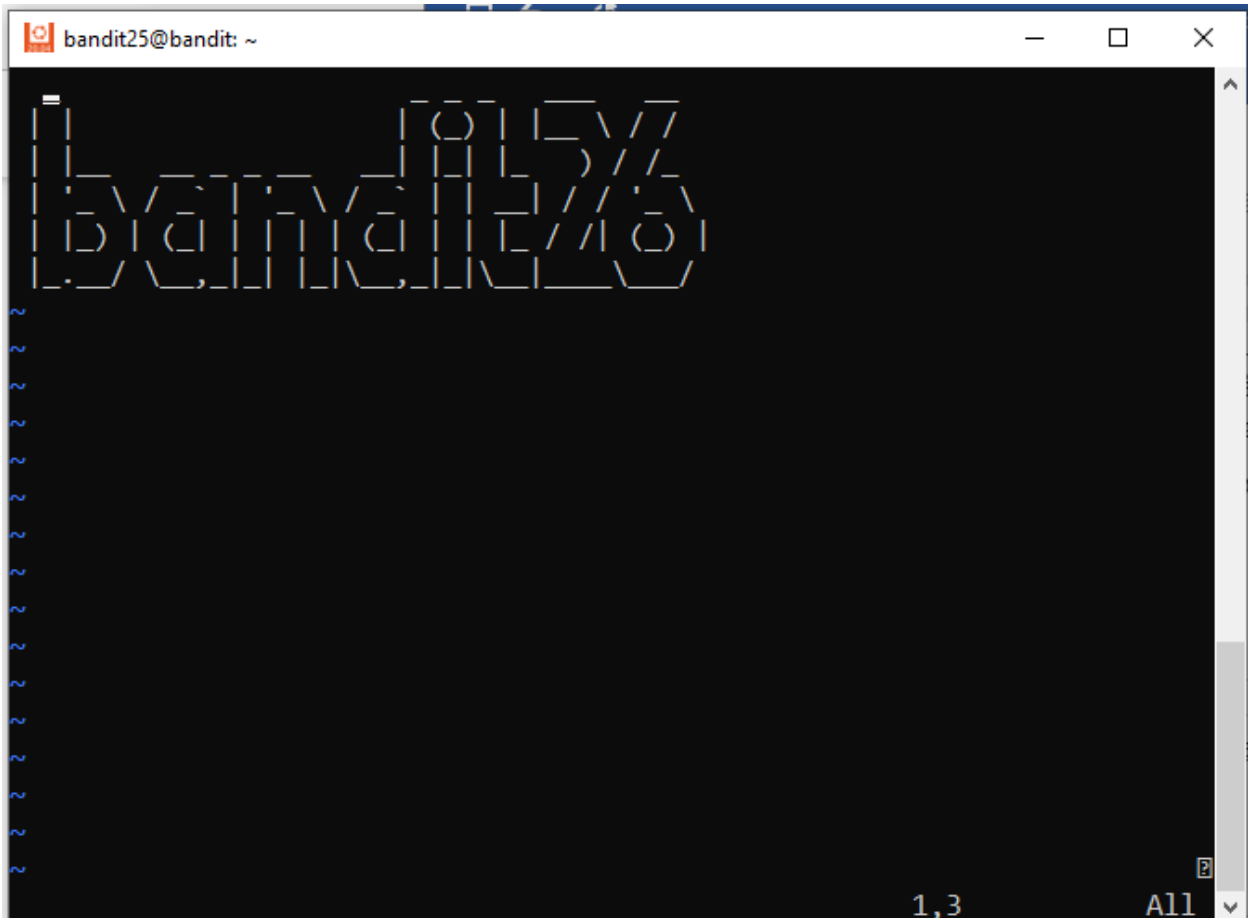
```
bandit25@bandit: ~  
--More-- (31%)
```

Tuy nhiên dù ta thử bất kì lệnh nào thì lập tức sẽ in lại các thông tin trong file `text ~/text.txt`

```
!man more  
--More-- (33%)
```

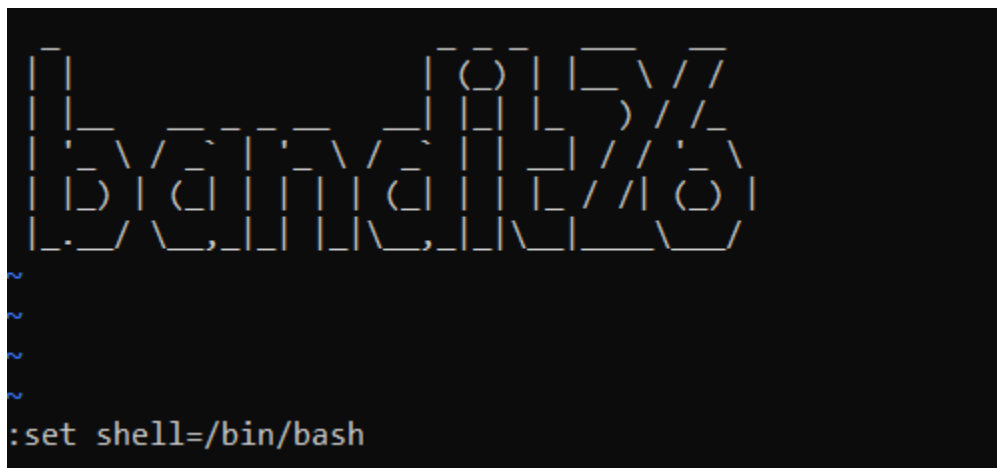
Điều này xảy ra là vì khi ta viết 1 câu lệnh trong more, more sẽ sinh một subshell để thực hiện câu lệnh này, và subshell này lại gọi lên file shell mặc định khi truy cập tới bandit26, tức là file shell nằm ở `/usr/bin/showtext`. File shell này lại in mọi thứ trong file `~/text.txt` ra.

Ta sẽ mở trình soạn thảo Vi trong quá trình kết nối để dễ dàng thực hiện các câu lệnh script bằng cách gõ v



Ta sẽ phải thay đổi giá trị shell mặc định bằng câu lệnh `set shell=[vị_trí_shell]` để kích hoạt shell ở [vị_trí_shell] khi ta mở một shell mới. Ta viết:

```
:set shell=/bin/bash
```



Xong ta thực hiện lệnh `:sh` để chạy đoạn lệnh trên

```
bandit25@bandit: ~  
:i  
bandit25@bandit: ~  
:sh  
[No write since last change]  
bandit26@bandit:~$
```

Truy cập thành công vào bandit26. Truy cập vào thư mục chứa flag ở /etc/bandit_pass/bandit26

```
bandit25@bandit: ~  
bandit26@bandit:~$ ls  
bandit27-do text.txt  
bandit26@bandit:~$ cat /etc/bandit_pass/bandit26  
c7GvcKlw9mC7aUQaPx7nwFstuAIBw1o1  
bandit26@bandit:~$
```

Thu được flag: c7GvcKlw9mC7aUQaPx7nwFstuAIBw1o1

2.28. Bandit26

Yêu cầu: Lấy flag dưới quyền bandit27

Tương tự như level trước, ta có file bandit27 trong đường dẫn gốc của user bandit26, chương trình này cho phép ta thực hiện câu lệnh dưới quyền bandit27

Truy cập vào thư mục chứa flag của bandit27 bằng tệp trên:

./bandit27-do cat /etc/bandit_pass/bandit27

```
bandit26@bandit:~$ ./bandit27-do cat /etc/bandit_pass/bandit27  
YnQpBuifNMas1hcUFk70ZmqkhUU2EuaS
```

Thu được flag YnQpBuifNMas1hcUFk70ZmqkhUU2EuaS

2.29. Bandit27

```
ssh bandit27@bandit.labs.overthewire.org -p 2220
```

Yêu cầu: kéo repo từ đường link `ssh://bandit27-git@localhost/home/bandit27-git/repo`

Trước hết, hãy tạo một thư mục trong /tmp để có quyền thao tác với git:

```
mktmp -d
```

```
cd /tmp/... (tên đường dẫn vừa tạo)
```

Cú pháp kéo repo về là:

```
git clone
```

```
ssh://username@hostname:port/path/to/repository
```

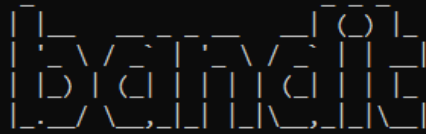
Trong đó:

- username: tên người dùng trên máy chủ Git
- hostname: địa chỉ hoặc tên miền của máy chủ Git
- port: cổng kết nối SSH của máy chủ Git
- path/to/repository: đường dẫn đến repository cần clone.

Ta nhập cú pháp:

```
ssh://bandit27-git@localhost:2220/home/bandit27-git/repo
```

```
bandit27@bandit:/tmp/tmp.W5o7Baor0e$ git clone ssh://bandit27-git@localhost:2220/home/bandit27-git/repo
Cloning into 'repo'...
The authenticity of host '[localhost]:2220 ([127.0.0.1]:2220)' can't be established.
ED25519 key fingerprint is SHA256:C2ihUBV7ihnV1wUXRb4RrEcLfXC5CXlhmAAM/urerLY.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Could not create directory '/home/bandit27/.ssh' (Permission denied).
Failed to add the host to the list of known hosts (/home/bandit27/.ssh/known_hosts).
```



```

      This is an OverTheWire game server.
  More information on http://www.overthewire.org/wargames

bandit27-git@localhost's password:
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (3/3), done.
```

Mở file README trong repo đã clone về, ta nhận được flag:

```
bandit27@bandit:/tmp/tmp.W5o7Baor0e$ ls
repo
bandit27@bandit:/tmp/tmp.W5o7Baor0e$ cd repo
bandit27@bandit:/tmp/tmp.W5o7Baor0e/repo$ ls
README
bandit27@bandit:/tmp/tmp.W5o7Baor0e/repo$ cat README
The password to the next level is: AVanL161y9rsbcJIsFHuw35rjaOM19nR
bandit27@bandit:/tmp/tmp.W5o7Baor0e/repo$
```

Flag: AVanL161y9rsbcJIsFHuw35rjaOM19nR

2.30. Bandit28

```
ssh bandit28@bandit.labs.overthewire.org -p 2220
```

Tương tự bài trước, yêu cầu là clone repo từ git và tìm flag trong đó:

```
bandit28@bandit:~$ mkdir -p /tmp/tmp.XoYxVAPq0s
bandit28@bandit:~$ cd /tmp/tmp.XoYxVAPq0s
bandit28@bandit:/tmp/tmp.XoYxVAPq0s$ git clone ssh://bandit28-git@localhost:2220/home/bandit28-git/repo
Cloning into 'repo'...
The authenticity of host '[localhost]:2220 ([127.0.0.1]:2220)' can't be established.
ED25519 key fingerprint is SHA256:C2ihUBV7ihnV1wUXRb4RrEcLFXC5CX1hmAAM/urerLY.
```

```
bandit28@bandit:/tmp/tmp.XoYxVAPq0s$ ls
repo
bandit28@bandit:/tmp/tmp.XoYxVAPq0s$ cd repo
bandit28@bandit:/tmp/tmp.XoYxVAPq0s/repo$ ls
README.md
bandit28@bandit:/tmp/tmp.XoYxVAPq0s/repo$ cat README.md
# Bandit Notes
Some notes for level29 of bandit.

## credentials

- username: bandit29
- password: xxxxxxxxxxxx
```

Mật khẩu đã được masked, ta kiểm tra lịch sử commit của git thông qua git log:

```
git log
```

```
bandit28@bandit:/tmp/tmp.XoYxVAPq0s/repo$ git log
commit 104db85a904e9691ff22aafe1a96124c88f75afa (HEAD -> master
, origin/master, origin/HEAD)
Author: Morla Porla <morla@overthewire.org>
Date: Tue Feb 21 22:03:10 2023 +0000

    fix info leak

commit 6c3c5e485cc531e5d52c321587ce1103833ab7c3
Author: Morla Porla <morla@overthewire.org>
Date: Tue Feb 21 22:03:10 2023 +0000

    add missing data

commit cd3b97ef95879ec34df0d6c82f2a96d552f0e56c
Author: Ben Dover <noone@overthewire.org>
Date: Tue Feb 21 22:03:10 2023 +0000

    initial commit of README.md
```

Dựa theo message (fix info leak - sửa thông tin bị lộ lọt), rất có thể git 6c3c5e485cc531e5d52c321587ce1103833ab7c3 có thông tin nhạy cảm, ta mở git trên:

Ta chuyển qua nhánh git trên để kiểm tra:

```
git checkout 6c3c5e485cc531e5d52c321587ce1103833ab7c3
```

```
bandit28@bandit:/tmp/tmp.XoYxVAPq0s/repo$ git checkout 6c3c5e485cc531e5d52c321587ce1103833ab7c3
Note: switching to '6c3c5e485cc531e5d52c321587ce1103833ab7c3'.
```

```
cat README.md
```

```
bandit28@bandit:/tmp/tmp.XoYxVAPq0s/repo$ cat README.md
# Bandit Notes
Some notes for level29 of bandit.

## credentials

- username: bandit29
- password: tQKvmcwNYcFS6vmPHIUSI3ShmsrQZK8S
```

Thu được flag: tQKvmcwNYcFS6vmPHIUSI3ShmsrQZK8S

2.31. Bandit29

`ssh bandit29@bandit.labs.overthewire.org -p 2220`

Yêu cầu: tương tự level trên với git repo của bandit 29

```
bandit29@bandit:~$ mkdir -p /tmp/tmp.pjjdsX1m0z
bandit29@bandit:~$ cd /tmp/tmp.pjjdsX1m0z
bandit29@bandit:/tmp/tmp.pjjdsX1m0z$ git clone ssh://bandit29-git@localhost:2220/home/bandit29-git/repo
```

Khi mở README.md, ta nhận thấy dòng password ghi là:

no passwords in production!

```
bandit29@bandit:/tmp/tmp.pjjdsX1m0z/repo$ cat README.md
# Bandit Notes
Some notes for bandit30 of bandit.

## credentials

- username: bandit30
- password: <no passwords in production!>
```

Câu này tức là sẽ không có mật khẩu trong môi trường production (Production environment). Môi trường Prod environment là nhánh môi trường sản xuất, nơi mà phần mềm hoặc ứng dụng được triển khai và sử dụng thực tế bởi người dùng cuối.

Như vậy chắc hẳn còn nhánh khác, ta kiểm tra các branch:

`git branch -a`

```
bandit29@bandit:/tmp/tmp.pjjdsX1m0z/repo$ git branch -a
* master
remotes/origin/HEAD -> origin/master
remotes/origin/dev
remotes/origin/master
remotes/origin/spl0its-dev
```

Nhận thấy môi trường dev, ta switch sang môi trường dev thông qua:

`git checkout dev`

```

remotes/origin/sploits dev
bandit29@bandit:/tmp/tmp.pjjdsX1m0z/repo$ git checkout dev
Branch 'dev' set up to track remote branch 'dev' from 'origin'.
Switched to a new branch 'dev'
bandit29@bandit:/tmp/tmp.pjjdsX1m0z/repo$ ls
code  README.md
bandit29@bandit:/tmp/tmp.pjjdsX1m0z/repo$ cat README.md
# Bandit Notes
Some notes for bandit30 of bandit.

## credentials

- username: bandit30
- password: xbhV3HpNGlTIdnjUrdAlPzc2L6y9EOnS

```

Thành công lấy được flag: xbhV3HpNGlTIdnjUrdAlPzc2L6y9EOnS

2.32. Bandit30

```
ssh bandit30@bandit.labs.overthewire.org -p 2220
```

Yêu cầu: như level trước

Thực hiện tương tự như bài trước:

```

bandit30@bandit:~$ mktemp -d
/tmp/tmp.mojpJxdhwF
bandit30@bandit:~$ cd /tmp/tmp.mojpJxdhwF
bandit30@bandit:/tmp/tmp.mojpJxdhwF$ git clone ssh://bandit30-git@localhost:2220/home/bandit30-git/repo
Cloning into 'repo'...

```

```

bandit30@bandit:/tmp/tmp.mojpJxdhwF$ cd repo
bandit30@bandit:/tmp/tmp.mojpJxdhwF/repo$ ls
README.md
bandit30@bandit:/tmp/tmp.mojpJxdhwF/repo$ cat README.md
just an empty file... muahaha

```

Thử lần lượt như các level trước, ta không thấy có thêm thông tin:

```

bandit30@bandit:/tmp/tmp.mojpJxdhwF/repo$ git log
commit cf552c166d78421e64ddf52f850e680075d216e1 (HEAD -> master, origin/master, origin/HEAD)
Author: Ben Dover <noone@overthewire.org>
Date: Tue Feb 21 22:03:13 2023 +0000

    initial commit of README.md
bandit30@bandit:/tmp/tmp.mojpJxdhwF/repo$ git branch -a
* master
  remotes/origin/HEAD -> origin/master
  remotes/origin/master

```

Git có cơ chế đánh dấu một commit trong lịch sử repository là sử dụng tag. Ta có thể kiểm tra tag của git này để tìm thêm thông tin:

```
git tag
```

```
bandit30@bandit:/tmp/tmp.mojpJxdhwF/repo$ git tag
secret
```

Nghe rất “hấp dẫn” đúng không? Cái gì mà cứ hidden, secret, password là có mùi flag phảng phất đâu đây. Ta xem tag secret này có gì:

```
git show secret
```

```
bandit30@bandit:/tmp/tmp.mojpJxdhwF/repo$ git show secret
OoffzGDlzhAlerFJ2cAiz1D41JW1Mhmt
```

Thành công lấy được flag: OoffzGDlzhAlerFJ2cAiz1D41JW1Mhmt

2.33. Bandit31

```
ssh bandit31@bandit.labs.overthewire.org -p 2220
```

Yêu cầu: như các level trước

Thực hiện tương tự để clone git:

```
bandit31@bandit:~$ mkdir -p /tmp/tmp.aT1FqVITgz
bandit31@bandit:~$ cd /tmp/tmp.aT1FqVITgz
bandit31@bandit:/tmp/tmp.aT1FqVITgz$ git clone ssh://bandit31-git@localhost:2220/home/bandit31-git/repo
Cloning into 'repo'...
```

Yêu cầu rất cụ thể:

```
bandit31@bandit:/tmp/tmp.aT1FqVITgz/repo$ cat README.md
This time your task is to push a file to the remote repository.

Details:
  File name: key.txt
  Content: 'May I come in?'
  Branch: master
```

Tức là: tạo một tệp tên là key.txt, có nội dung là ‘May I come in?’, và commit lên branch master. Ta thực hiện lần lượt câu lệnh:

Tạo tệp với nội dung ‘May I come in?’ với tên file là key.txt

```
echo "May I come in?" > key.txt
```

Thêm file `key.txt` và commit:

```
git add key.txt -f
```

Lệnh `git add key.txt` để thêm file vào **staging area** (phân vùng chuẩn bị sẵn sàng trước khi commit và push lên repository). Do thông thường Git chỉ đánh dấu các thay đổi mới hoặc sửa đổi so với lần commit trước, nên ta thêm tham số `-f` để *force*, bỏ qua quá trình kiểm tra trên.

```
git commit -m "Add key.txt"
```

Lệnh trên tạo một commit mới, đưa những file hiện có trong phân vùng **staging area** đi commit. `-m "Add key.txt"` chỉ là message tùy chọn.

```
bandit31@bandit:/tmp/tmp.aT1FqVITgz/repo$ git add key.txt -f
bandit31@bandit:/tmp/tmp.aT1FqVITgz/repo$ git commit -m "Add key.txt"
[master b1d9b4a] Add key.txt
1 file changed, 1 insertion(+)
create mode 100644 key.txt
```

Sau đó ta sử dụng câu lệnh

```
git push origin master
```

`git push origin master` là lệnh để đẩy các thay đổi từ **local repository** lên **remote repository** trên nhánh master.

- `git push` là lệnh để đẩy các thay đổi từ local repository lên remote repository.
- `origin` là tên của remote repository.
- `master` là tên của nhánh trên remote repository muốn đẩy các thay đổi lên.

```
bandit31@bandit:/tmp/tmp.aT1FqVITgz/repo$ git push origin master
The authenticity of host '[localhost]:2220 ([127.0.0.1]:2220)' can't be e
stablished.
ED25519 key fingerprint is SHA256:C2ihUBV7ihnV1wUXRb4RrEcLfXC5CX1hmAAM/ur
erLY.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Could not create directory '/home/bandit31/.ssh' (Permission denied).
Failed to add the host to the list of known hosts (/home/bandit31/.ssh/kn
own_hosts).
```

Ta thu được flag

```
bandit31-git@localhost's password:
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 2 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 324 bytes | 324.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
remote: ### Attempting to validate files... ###
remote:
remote: .o0o.o0o.o0o.o0o.o0o.o0o.o0o.o0o.o0o.o0o.
remote:
remote: Well done! Here is the password for the next level:
remote: rmCBvG56y58BXzv98yZGd07ATVL5dW8y
remote:
remote: .o0o.o0o.o0o.o0o.o0o.o0o.o0o.o0o.o0o.o0o.
remote:
To ssh://localhost:2220/home/bandit31-git/repo
! [remote rejected] master -> master (pre-receive hook declined)
```

Flag: rmCBvG56y58BXzv98yZGd07ATVL5dW8y

2.34. Bandit32

```
ssh bandit32@bandit.labs.overthewire.org -p 2220
```

Xong phần git!!!!

Ok không có một hướng dẫn cụ thể nào cho level này, những gì ta biết là:

- Đoạn bash khởi tạo của bandit32 là 1 đoạn bash có chức năng viết hoa tất cả kí tự chữ cái ta nhập.

```

WELCOME TO THE UPPERCASE SHELL
>> ls
sh: 1: LS: not found
>> pwd
sh: 1: PWD: not found
>> shell
sh: 1: SHELL: not found
>> $shell
WELCOME TO THE UPPERCASE SHELL
>> $HOME
sh: 1: /home/bandit32: Permission denied
>> ECHO $HOME
sh: 1: ECHO: not found
>>

```

- Mật khẩu khả năng cao sẽ được lưu trữ trong `bandit_pass`

Ta có thể kiểm tra lại file bash được kích hoạt khi ssh đến bandit32, thông qua việc truy cập từ bandit31 và kiểm tra file `/etc/passwd`

```

bandit31@bandit:~$ cat /etc/passwd | grep bandit32
bandit32:x:11032:11032:bandit level 32:/home/bandit32:/home/bandit32/uppershell

```

Đúng như dự đoán, file bash được kích hoạt không phải `/bin/bash` mà là `/home/bandit32/uppershell`

Rất tiếc ta không có quyền truy cập vào file bash trên:

```

bandit31@bandit:~$ cat /home/bandit32/uppershell
cat: /home/bandit32/uppershell: Permission denied

```

Ta sẽ lại ssh vào bandit32 để tìm cách bẻ khóa

*Trước khi thực hiện, chúng ta cần một phần kiến thức bổ trợ về **positional parameter** (vị trí các tham số).*

Trong Linux, **positional parameter** là các tham số được truyền vào khi gọi một script hoặc một lệnh trong Terminal. Các tham số này được đánh số từ `$0` đến `$n`, với `$0` là tên của script hoặc lệnh đang được gọi, `$1` là tham số đầu tiên, `$2` là tham số thứ hai và tiếp tục cho đến `$n` là tham số cuối cùng. Chúng ta có thể sử dụng các positional parameter này để truyền các giá trị hoặc đối số vào trong script hoặc lệnh mà không cần nhập trực tiếp từ bàn phím.

Ví dụ với một script tên là anhnt.sh có nội dung như sau:

```
#!/bin/bash
```

```
echo "First position parameter: $0"
```

```
echo "Second position parameter: $1"
```

```
echo "Third position parameter: $2"
```

Thì khi ta chạy câu lệnh:

```
./anhnt.sh Viet Nam vo dich
```

Kết quả thu được sẽ là:

```
First position parameter: ./anhnt.sh
```

```
Second position parameter: Viet
```

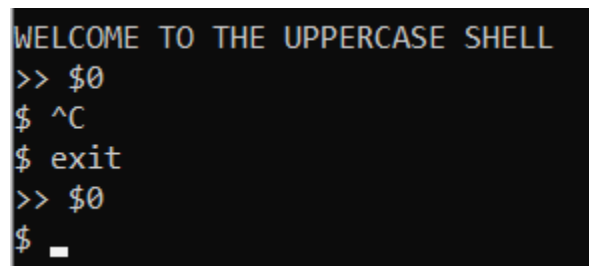
```
Third position parameter: Nam
```

Tức là \$0 sẽ có giá trị là tên đoạn script/câu lệnh, \$1 \$2... lần lượt là giá trị các tham số truyền vào.

Điểm đặc biệt là nếu ta không chạy câu lệnh nào, thì \$0 mặc định sẽ là file shell mặc định cấu hình trong hệ thống (bash shell). Tức là nếu ta gõ \$0, thì sẽ chạy file shell mặc định đó.

Như vậy ta sẽ mở được một subshell mới ở vị trí shell mặc định bằng cách gõ:

```
$0
```



```
WELCOME TO THE UPPERCASE SHELL
>> $0
$ ^C
$ exit
>> $0
$ _
```

Ta thử mở thư mục chứa mật khẩu:

```
$ cat /etc/bandit_pass/bandit32
cat: /etc/bandit_pass/bandit32: Permission denied
```

Tại sao lại bị từ chối? Ta kiểm tra lại user của chúng ta có đúng là bandit32 hay không:

```
$ whoami
bandit33
```

Như vậy, thư mục ta cần truy cập tới sẽ là mật khẩu của bandit33, thay vì bandit32:

```
cat /etc/bandit_pass/bandit33
```

```
$ cat /etc/bandit_pass/bandit33
odHo63fHiFqcWWJG9rLiLDtPm45KzUKy
```

Thu được flag: odHo63fHiFqcWWJG9rLiLDtPm45KzUKy

2.35. Bandit33

```
ssh bandit33@bandit.labs.overthewire.org -p 2220
```

```
bandit33@bandit:~$ ls
README.txt
bandit33@bandit:~$ cat README.txt
Congratulations on solving the last level of this game!

At this moment, there are no more levels to play in this game. However, we are working on new levels and will most likely expand this game with more levels. Keep an eye out for an announcement on our usual communication channels. In the meantime, you could play some of our other wargames.

If you have an idea for an awesome new level, please let us know!
bandit33@bandit:~$
```

Chúc mừng anh em đã “phá đảo”. Hãy tiếp tục thử thách mình ở các wargame khác nhé. :D