

OVER THE WIRE BANDIT WARGAME

Level 10)

```
bandit10@bandit:~$ cat data.txt
VGhlIHBhc3N3b3JkIGlzIGR0UjE3M2ZaS2IwUjJzREZTR3NnMlJXbnBOVmozcVJyCg==
bandit10@bandit:~$ base64 -d data.txt
The password is dtR173fZKb0RRsDFSGsg2RWnpNVj3qRr
```

Here, the level told us that the password was encoded in base64 in data.txt so we used base64 to decode the file hence revealing the password

Level 11)

Recipe	Input
ROT13 <input checked="" type="checkbox"/> Rotate lower case chars <input checked="" type="checkbox"/> Rotate upper case chars <input type="checkbox"/> Rotate numbers Amount: 13	Gur cnffjbeq vf 7k16JARuVV5LxVuJfsSVdbbtaHG1w9D4
	Output The password is 7x16WNeHIi5YkIhwsfFIqoognUTyj9Q4

Here, the level said the password is stored in a file where all the lowercase (a-z) and uppercase (A-Z) letters have been rotated by 13 positions, so I used the cyberchef io to help me with the same and save time (snapshot), revealing the password, first I cat the data.txt to show the contents of the file and then put the same in “input” section of cyberchef io (search for rot13 for rotation 13)

Level 12)

```
bandit12@bandit:~$ ls
data.txt
bandit12@bandit:~$ mkdir /tmp/sadge
bandit12@bandit:~$ mv data file.gz
bandit12@bandit:~$ cd /tmp/sadge
bandit12@bandit:/tmp/sadge$ ls
data data.txt
bandit12@bandit:/tmp/sadge$ file data
data: gzip compressed data, was "data4.bin", last modified: Thu Apr 10 14:22:57 2025, max compression, from Unix, original size modulo 2^32 20480
bandit12@bandit:/tmp/sadge$ mv data file.gz
bandit12@bandit:/tmp/sadge$ gzip -d file.gz
bandit12@bandit:/tmp/sadge$ file file
file: POSIX tar archive (GNU)
bandit12@bandit:/tmp/sadge$ mv file file.tar
bandit12@bandit:/tmp/sadge$ tar xf file.tar
bandit12@bandit:/tmp/sadge$ file file
file: cannot open 'file' (No such file or directory)
bandit12@bandit:/tmp/sadge$ ls
data5.bin data.txt file.tar
bandit12@bandit:/tmp/sadge$ file data5.bin
data5.bin: POSIX tar archive (GNU)
bandit12@bandit:/tmp/sadge$ rm file.tar
bandit12@bandit:/tmp/sadge$ rm data.txt
bandit12@bandit:/tmp/sadge$ ls
data5.bin
bandit12@bandit:/tmp/sadge$ mv data5.bin data.tar
bandit12@bandit:/tmp/sadge$ xar xf data.tar
Command 'xar' not found, but there are 20 similar ones.
bandit12@bandit:/tmp/sadge$ tar xf data.tar
bandit12@bandit:/tmp/sadge$ ls
data6.bin data.txt
bandit12@bandit:/tmp/sadge$ file data6.bin
data6.bin: bzip2 compressed data, block size = 900k
bandit12@bandit:/tmp/sadge$ mv data data6bin.data.bz2
mv: target 'data.bz2': No such file or directory
bandit12@bandit:/tmp/sadge$ mv data data6bin.data.bz2
mv: target 'data.bz2': No such file or directory

bandit12@bandit:/tmp/sadge$ mv data6bin.data.bz2
mv: cannot stat 'data6bin': No such file or directory
bandit12@bandit:/tmp/sadge$ mv data6bin.data.bz2
mv: cannot stat 'data6bin': No such file or directory
bandit12@bandit:/tmp/sadge$ mv data6bin.data.bz2
bandit12@bandit:/tmp/sadge$ bzip2 -d data.bz2
bandit12@bandit:/tmp/sadge$ ls
data data.txt
bandit12@bandit:/tmp/sadge$ file data
data: POSIX tar archive (GNU)
bandit12@bandit:/tmp/sadge$ mv data data.tar
bandit12@bandit:/tmp/sadge$ tar xf data.tar
bandit12@bandit:/tmp/sadge$ ls
data8.bin data.txt
bandit12@bandit:/tmp/sadge$ file data8.bin
data8.bin: gzip compressed data, was "data9.bin", last modified: Thu Apr 10 14:22:57 2025, max compression, from Unix, original size modulo 2^32 49
bandit12@bandit:/tmp/sadge$ mv data8.bin data.gzip
bandit12@bandit:/tmp/sadge$ gzip -d data.gzip
gzip: data.gzip: unknown suffix -- ignored
bandit12@bandit:/tmp/sadge$ mv data8.bin data.gz
mv: cannot stat 'data8.bin': No such file or directory
bandit12@bandit:/tmp/sadge$ ls
data.gzip data.txt
bandit12@bandit:/tmp/sadge$ mv data.gzip data.gz
bandit12@bandit:/tmp/sadge$ gzip -d data.gz
bandit12@bandit:/tmp/sadge$ ls
data data.txt
bandit12@bandit:/tmp/sadge$ file data
data: ASCII text
bandit12@bandit:/tmp/sadge$ cat data
The password is C05wErcsch1l1nH0j220ks2vdTDwAn
bandit12@bandit:/tmp/sadge$ rm data.tar
bandit12@bandit:/tmp/sadge$ exit
logout
Connection to bandit.labs.overthewire.org closed.
```

Okay starting off with, this level being soooo different from the previous ones, increase in difficulty is crazy too but the basic logic here was to repeatedly decompress the repeatedly compressed file until we get the file with the password (different format than any compressed format) we checked this by using the file command

Level 13)

```
bandit14@localhost: Permission denied (publickey).
bandit13@bandit:~$ ssh -i sshkey.private bandit14@localhost -p 2220
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/bandit13/.ssh' (Permission denied).
Failed to add the host to the list of known hosts (/home/bandit13/.ssh/known_hosts).
```

```
bandit14@bandit:~$ cat /etc/bandit_pass/bandit14
MU4VWeTyJk8R0of1qqmcBPALh7lDCPvS
bandit14@bandit:~$
```

Here, first we use the ssh key to login to the next level and then as it was mentioned in the level we cat the content of the “/etc/bandit_pass/bandit14” file which gives us the password, why this happens?? Because it can only be read by the user bandit14 and we login as the user itself in as the first step

Level 14)

```
bandit14@bandit:~$ cat /etc/bandit_pass/bandit14
MU4VWeTyJk8R0of1qqmcBPALh7LDCPvS
bandit14@bandit:~$ nc localhost 30000
MU4VWeTyJk8R0of1qqmcBPALh7LDCPvS
Correct!
8xCjnmgoKbGLhHFAZlGE5Tmu4M2tKJQo
```

Staying as the same user now to find the password for the next level we need to access the port 30000 of the localhost (the machine we are working on) therefore we use nc “netcat” to get the password for level 15, it asks us to give the password we had just discovered and then provides the password for the next level

Level 15)

```
bandit15@bandit:~$ cat /etc/bandit_pass/bandit15
8xCjnmgoKbGLhHFAZlGE5Tmu4M2tKJQo
bandit15@bandit:~$ man ncat | grep ssl
--ssl          Connect or listen with SSL
--ssl-cert     Specify SSL certificate file (PEM) for listening
--ssl-key      Specify SSL private key (PEM) for listening
--ssl-verify   Verify trust and domain name of certificates
--ssl-trustfile PEM file containing trusted SSL certificates
--ssl-ciphers  Cipherlist containing SSL ciphers to use
--ssl-servername Request distinct server name (SNI)
--ssl-alpn     ALPN protocol list to use
--ssl (Use SSL)
--ssl-verify (Verify server certificates)
    In client mode, --ssl-verify is like --ssl except that it also requires verification of the server certificate. Ncat comes with a default set
    used if available. Use --ssl-trustfile to give a custom list. Use -v one or more times to get details about verification failures. Ncat does
--ssl-cert certfile.pem (Specify SSL certificate)
    mode). Use it in combination with --ssl-key.
--ssl-key keyfile.pem (Specify SSL private key)
    This option gives the location of the PEM-encoded private key file that goes with the certificate named with --ssl-cert.
--ssl-trustfile cert.pem (List trusted certificates)
--ssl-verify. The argument to this option is the name of a PEM file containing trusted certificates. Typically, the file will contain
--ssl-ciphers cipherlist (Specify SSL ciphersuites)
--ssl-servername name (Request distinct server name)
--ssl-alpn ALPN list (Specify ALPN protocol list)
http://www.openssl.org
bandit15@bandit:~$ ncat --ssl localhost 300001
Ncat: Invalid port number "300001". QUITTING.
bandit15@bandit:~$ ncat --ssl localhost 30001
8xCjnmgoKbGLhHFAZlGE5Tmu4M2tKJQo
Correct!
kSkvUpMQ7lBYyCM4GBPvCvT1BfWRy0Dx
^C
bandit15@bandit:~$ echo kSkvUpMQ7lBYyCM4GBPvCvT1BfWRy0Dx
kSkvUpMQ7lBYyCM4GBPvCvT1BfWRy0Dx
bandit15@bandit:~$ exit
logout
Connection to bandit.labs.overthewire.org closed.
```

Here, we were told the password is on the port 30001 and is saved using SSL/TLS encryption, so we used --ssl and connected to that port then we entered the password to get the password for the next level.

Level 16)

```
bandit16@bandit:~$ nmap localhost -p 31000-32000
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-06-02 10:28 UTC
Nmap scan report for localhost (127.0.0.1)
Host is up (0.00010s latency).
Not shown: 996 closed tcp ports (conn-refused)
PORT      STATE SERVICE
31046/tcp  open  unknown
31518/tcp  open  unknown
31691/tcp  open  unknown
31790/tcp  open  unknown
31960/tcp  open  unknown

Nmap done: 1 IP address (1 host up) scanned in 0.05 seconds
bandit16@bandit:~$ nc localhost 31046
^C
bandit16@bandit:~$ cat /etc/bandit_pass/bandit16
kSkvUpMQ7lBYyCM4GBPvCvT1BfWRy0Dx
bandit16@bandit:~$ nc localhost 31046
kSkvUpMQ7lBYyCM4GBPvCvT1BfWRy0Dx
kSkvUpMQ7lBYyCM4GBPvCvT1BfWRy0Dx
^C
bandit16@bandit:~$ nc localhost 31518
kSkvUpMQ7lBYyCM4GBPvCvT1BfWRy0Dx
bandit16@bandit:~$ nc localhost 31790
^C
bandit16@bandit:~$ ncat -ssl localhost 31046
kSkvUpMQ7lBYyCM4GBPvCvT1BfWRy0Dx
^C
bandit16@bandit:~$ ncat -ssl localhost 31790
kSkvUpMQ7lBYyCM4GBPvCvT1BfWRy0Dx
Ncat: Could not resolve source address "sl": Temporary failure in name resolution. QUITTING.
bandit16@bandit:~$ kSkvUpMQ7lBYyCM4GBPvCvT1BfWRy0Dx
kSkvUpMQ7lBYyCM4GBPvCvT1BfWRy0Dx: command not found
bandit16@bandit:~$ ncat --ssl localhost 31790
kSkvUpMQ7lBYyCM4GBPvCvT1BfWRy0Dx
Correct!
-----BEGIN RSA PRIVATE KEY-----
```

Here we were told that the password for the next level is on the active ports ranging from 31000 to 32000, so we used nmap in this range to get to know the active ports and used ssl again to gain access to that particular port, enter our password and then get the password for the next level.

Level 17)

```
bandit17@bandit:~$ ls
passwords.new  passwords.old
bandit17@bandit:~$ diff passwords.new passwords.old
42c42
< x2gLTTjFwMOhQ8oWNBmN362QKxfRqGLO
---
> C6XNBdY0kgt5ARXESMKWWOUwBeaIQZ0Y
bandit17@bandit:~$ exit
logout
Connection to bandit.labs.overthewire.org closed.
```

Here, this level we just had to use the diff command to differentiate between the old and new password file, which has the changed password.


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

Here, to verify if the password is correct, we try to login to the next level, if it is correct then at the end of the default OTW message it'll say "byebye!" because someone has modified the .bashrc to reject our access whenever we try to login with SSH

Level 18)

```
(kali@kali: ~)/OTWbandit
$ man ssh | grep terminal
-t      Disable pseudo-terminal allocation.
-T      Force pseudo-terminal allocation. This can be used to execute arbitrary screen-based programs on a remote machine, which can be very useful, e.g. when implementing menu
If an interactive session is requested, ssh by default will only request a pseudo-terminal (pty) for interactive sessions when the client has one. The flags -T and -t can be used
If a pseudo-terminal has been allocated, the user may use the escape characters noted below.
If no pseudo-terminal has been allocated, the session is transparent and can be used to reliably transfer binary data. On most systems, setting the escape character to "none" will
When a pseudo-terminal has been requested, ssh supports a number of functions through the use of an escape character.
SSH_ASKPASS If ssh needs a passphrase, it will read the passphrase from the current terminal if it was run from a terminal. If ssh does not have a terminal associated
```

```
(kali㉿kali)-[~/OTWbandit]
$ ssh -t bandit18@bandit.labs.overthewire.org -p 2220 /bin/sh

      _ _ _ _ _
     / /   / /
    /_/   /_/

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

bandit18@bandit.labs.overthewire.org's password:
$ echo sadge fujin
sadge fujin
$ cat readme
cGwPmaKXVwDUNGPAVJbWYuGHVn9z13j8
$ echo cGwPmaKXVwDUNGPAVJbWYuGHVn9z13j8
cGwPmaKXVwDUNGPAVJbWYuGHVn9z13j8
$ exit
Connection to bandit.labs.overthewire.org closed.
```

Here, like we said the `bashrc` was modified to not let us gain access, so we used a different command to access the shell, the “-t” forces a pseudo terminal allocation which is necessary for interactive shell sessions. And WKT the default shell of the server (`bashrc`) has been modified so we need to use a different shell to login other than the `bash` shell, so in the end we specify which shell to use “`/bin/sh`” this is done to circumvent the problem of the `bashrc` file, now we enter the password we had obtained from the previous level and enter it, it shows that we are in the `sh` shell and from here on we can `cat` the `readme` file to get the password for the next level.

this was def a tricky ques lol took me some time, trial and error methods

Level 19)

```
bandit19@bandit:~$ ls
bandit20-do
bandit19@bandit:~$ ./bandit20-do ID
env: 'ID': Permission denied
bandit19@bandit:~$ ./bandit20-do id
uid=11019(bandit19) gid=11019(bandit19) euid=11020(bandit20) groups=11019(bandit19)
bandit19@bandit:~$ ./bandit20-do cat /etc/bandit_pass/bandit20
0qXahG8ZjOVMN9Ghs7iOWsCfZyXOUbY0
bandit19@bandit:~$
```

Here, when we execute the executable it temporarily changes the user's effective uid (it is 20 in this case) makes bandit20 the owner of the file, meaning commands executed with bandit20-do will run the perms of bandit20, and now because bandit20 is the owner we use the cat command to reveal the password

