**Access**

Difficulty: Easy

OS: Windows

**Nmap**

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

|  |
| --- |

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.

|  |
| --- |

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.

| *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.

| *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.

| 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.

| *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.

| Password*:* **access4u@security** |
| --- |

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

| *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.

| *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.

| *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.

|  |
| --- |

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”

|  |
| --- |

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.

| *Readpst Access\ Control.pst* |
| --- |

Reading this file, we find some interesting information

| **4Cc3ssC0ntr0ller** |
| --- |

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

| *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.

|  |
| --- |