EUROPEAN COLLABORATION SUMMIT 2018

MAINZ, GERMANY / MAY 28-30 2018 - WWW.COLLABSUMMIT.EU

Tony Redmond

USING POWERSHELL TO MANAGE OFFICE 365 GROUPS AND MICROSOFT TEAMS

Sensitivity: Confidential

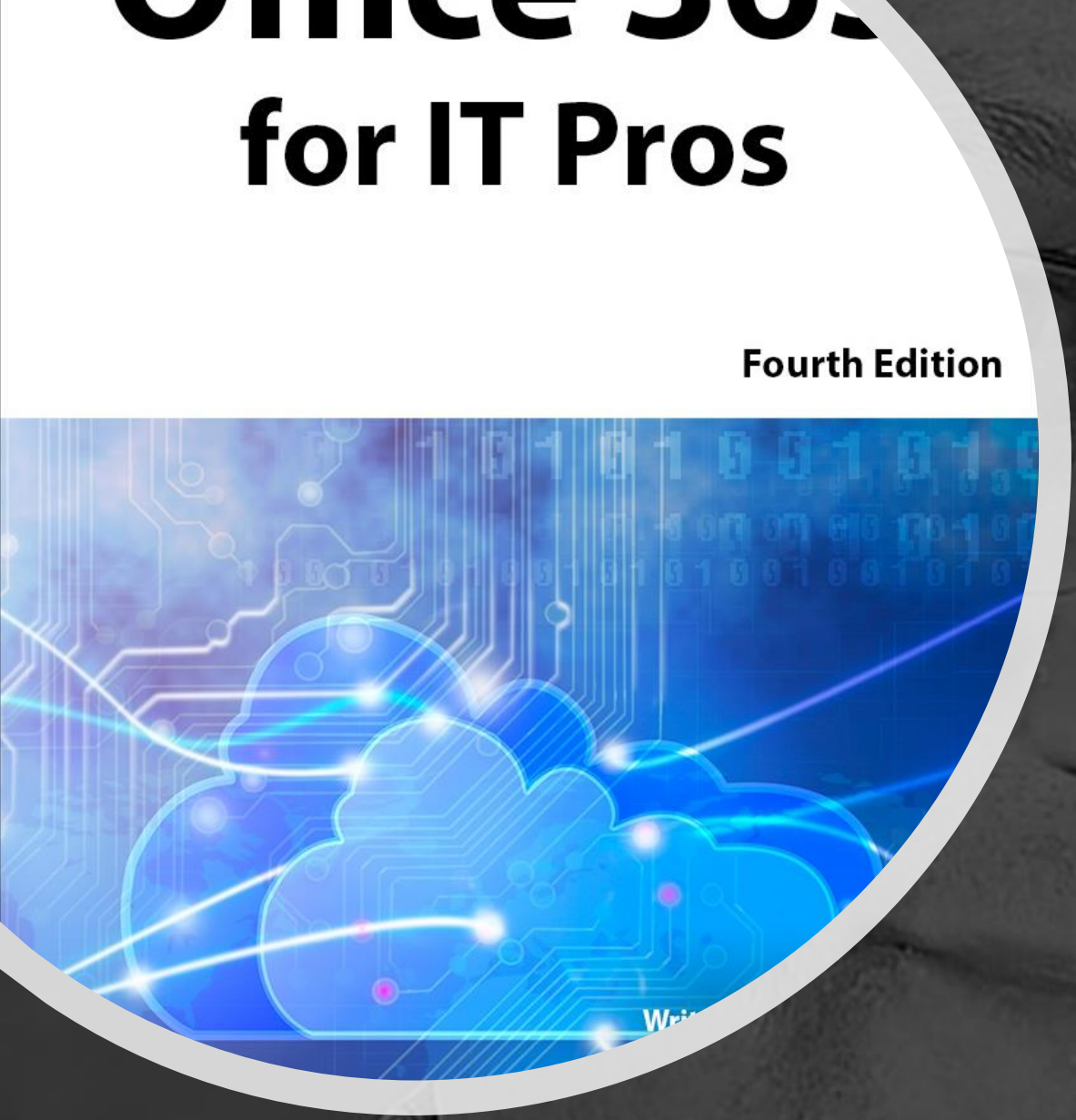
DIAMOND, PLATINUM AND GOLD SPONSORS

Sensitivity: Confidential

Using PowerShell to manage Office 365 Groups and Microsoft Teams

Tony Redmond

@12Knocksinna



Tony Redmond

- Lead author for “Office 365 for IT Pros” eBook
  - https://practical365.com/ebooks/office-365-for-it-pros/
- MVP since 2004
- Columnist for Petri.com
  - https://www.petri.com/category/office/office-365

Context

- Not an Expert class
- **Practical** rather than beautiful
- Focused on getting work done
- You can always improve PowerShell code later

Sensitivity: Confidential

Office 365 is great, so why do I need PowerShell?

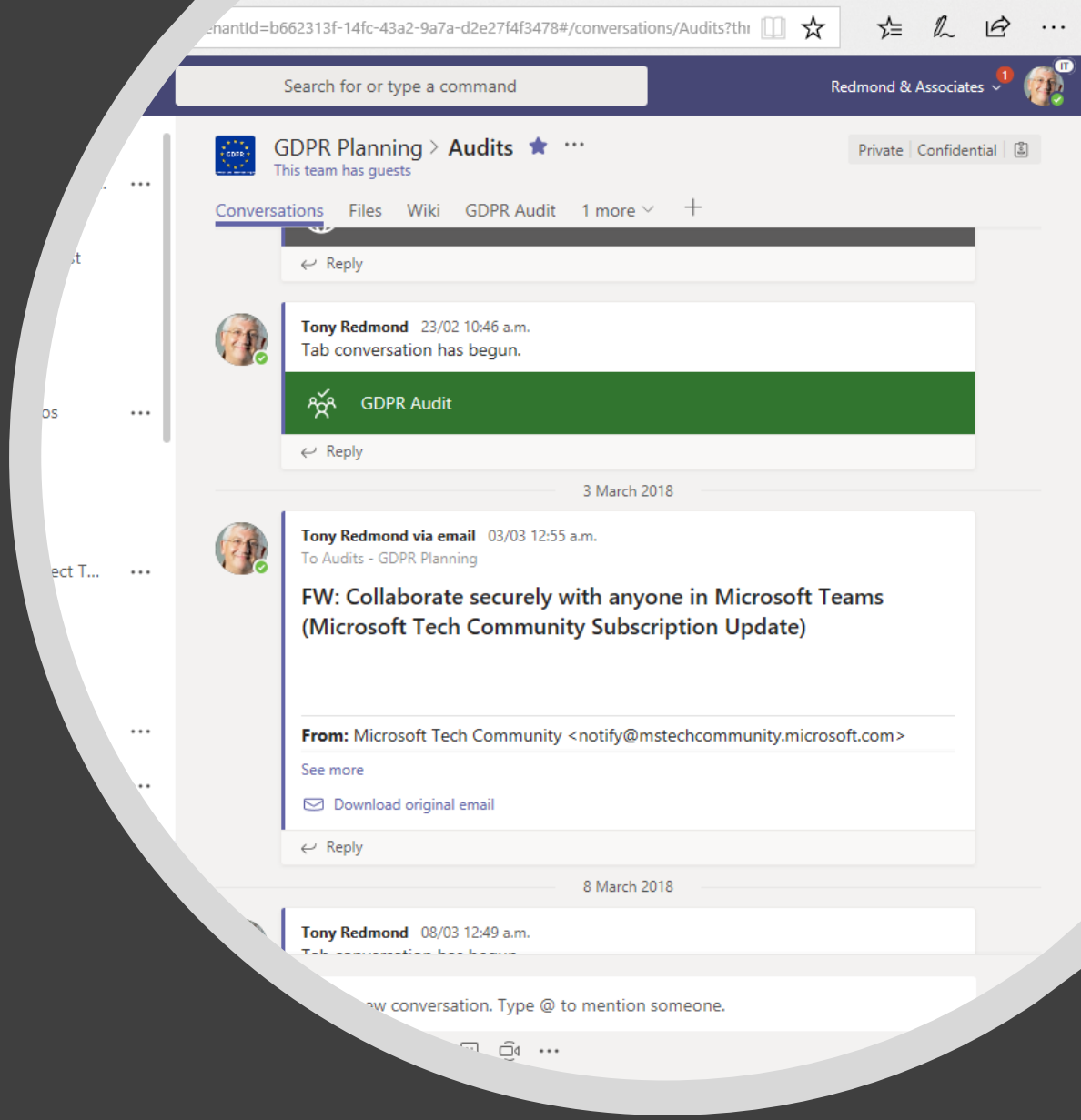
- Admin Tools are imperfect
- Microsoft cannot anticipate every possible administrative task
- PowerShell helps you to understand Office 365
- PowerShell isn't difficult, and it can be fun
- But sometimes, PowerShell is the wrong answer…

Tools of the Trade

- Exchange Online (*-UnifiedGroup and *-UnifiedGroupLinks, and Search-UnifiedAuditLog)
- Teams PowerShell module
- SharePoint Online PowerShell module
- Security and Compliance Center
- Azure Active Directory Preview (V2.0.0.137)

A Complex Environment

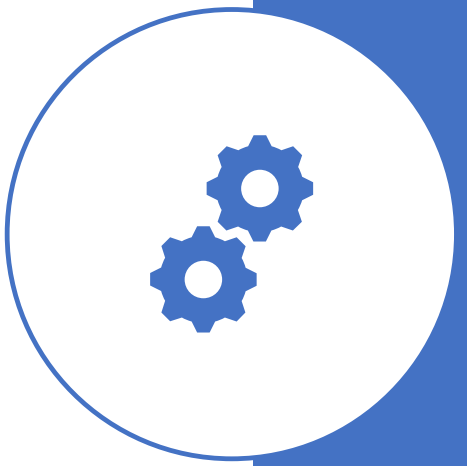
- Teams depend on Office 365 Groups and share common membership
- Azure Active Directory is the master
- UnifiedGroup cmdlets read EXODS; SharePoint cmdlets read SPODS
- Whenever possible, use Teams cmdlets to update groups used with Teams and Groups cmdlets otherwise
- We can read settings, but not content (conversations)



Sensitivity: Confidential

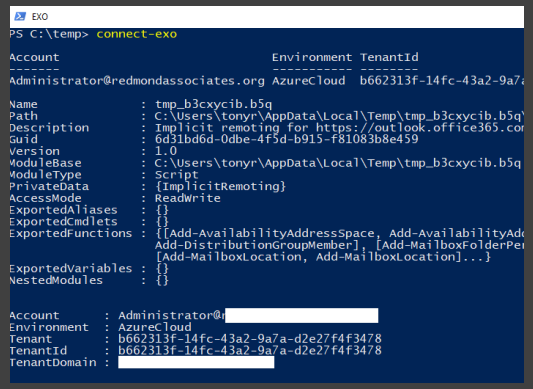
Avoid PowerShell Throttling

- Use server-side filters whenever possible
- Select properties to reduce data passing between client and server
- Use Invoke-Command to process data remotely



```
$Groups = (Invoke-Command -Session $Session -ScriptBlock {Get-
UnifiedGroup -ResultSize Unlimited | Select-Object DisplayName, Alias,
GroupMemberCount, WhenCreated})
```

Sensitivity: Confidential



Connecting to PowerShell

(otherwise your cmdlets won't work…)

- Connect to Exchange Online to use *–UnifiedGroups cmdlets and other Exchange cmdlets (like Search-UnifiedAuditLog)
- Connect to Teams to use Teams cmdlets
- Connect to Azure Active Directory to use Azure AD cmdlets
- Connect to SharePoint Online to interact with sites
- Confused? Use Michel de Rooij's PowerShell profile for Office 365

https://eightwone.com/2015/08/31/connecting-to-office-365exchange/

Some Basic Cmdlets

Function	Cmdlets
Create Groups and Teams	New-UnifiedGroup/New-Team
Retrieve settings	Get-UnifiedGroup Get-TeamMemberSettings Get-TeamMessagingSettings Get-TeamGuestSettings Get-TeamFunSettings Get-TeamChannel
Update settings	Set-UnifiedGroup Set-Team*
Access membership	Get-UnifiedGroupLinks [-LinkType Members, Owners] Add-UnifiedGroupLinks Get-TeamUser Add-TeamUser

Sensitivity: Confidential

Cmdlet Identities

- Groups cmdlets behave like Exchange mailbox cmdlets

```
Get-UnifiedGroup –Identity GroupAdmins@Office365ItPros.com
```

- Some Exchange mailbox cmdlets work against group mailboxes
- Teams and Azure AD cmdlets use GUIDs

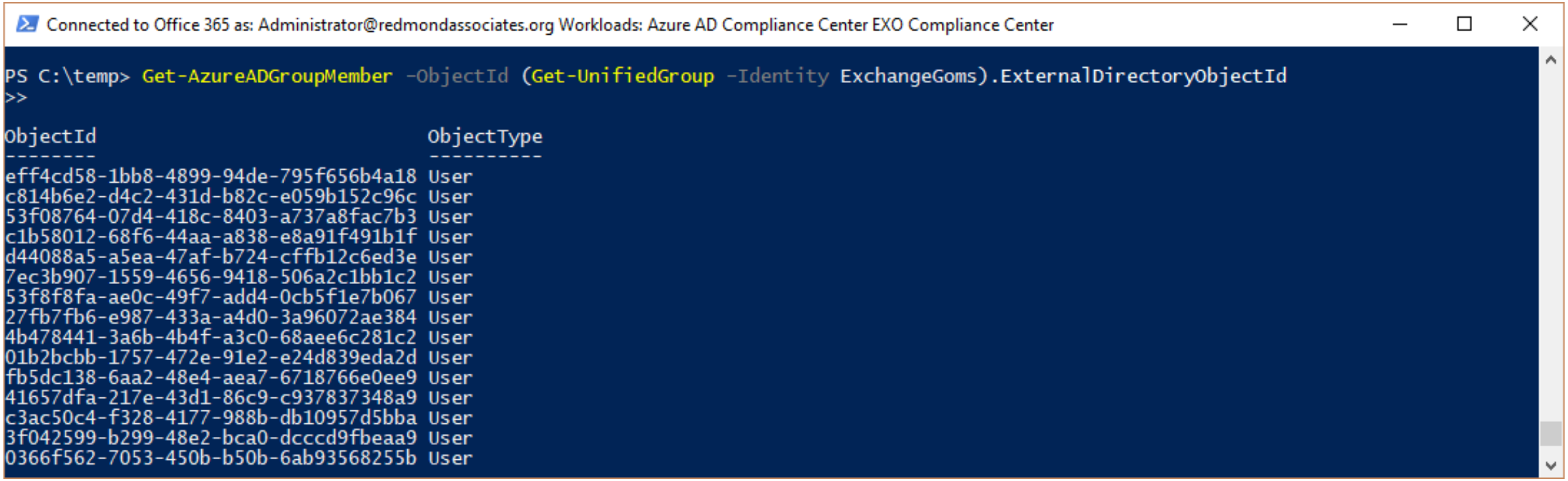
```
Get-TeamUser -GroupId 72ee570e-3dd8-41d2-bc84-7c9eb8024dd4
```

- If you don’t know a GUID, ask Groups

```
Get-TeamUser –GroupId (Get-UnifiedGroup –Identity ExchangeGoms).ExternalDirectoryObjectId
```

```
Get-AzureADGroupMember –ObjectId (Get-UnifiedGroup –Identity ExchangeGoms).ExternalDirectoryObjectId
```

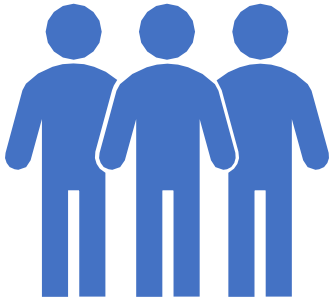
Sensitivity: Confidential



Demo

Hacking Around with Groups and Teams in PowerShell

Azure AD Policy for Groups



- Controls whether users can create groups and teams
- Controls whether guest users are supported on a tenant-wide or group-specific basis
- Defines classifications
- Defines naming policy for new groups and teams
- Must be manipulated with PowerShell
- Policy settings in Azure AD settings object

Sensitivity: Confidential

Creating a new Groups policy

```
[PS] C:\> Connect-AzureAD
```

```
[PS] C:\> $Policy = Get-
AzureADDirectorySettingTemplate | ?
{$_.DisplayName -eq "Group.Unified"}
```

```
[PS] C:\> $Settings =
$Policy.CreateDirectorySetting()
```

```
[PS] C:\> New-
AzureADDirectorySetting -
DirectorySetting $Settings
```

Examining Groups Policy Settings

```
[PS] C:\> Get-AzureADDirectorySetting | ForEach Values

Name                            Value
----                            -----
CustomBlockedWordsList
ClassificationDescriptions      General Usage:Anyone can access,External
Access:Available outside the company,Internal Only:Must not be shared
with external people,Confidential:Can only be disclosed with management
permission
DefaultClassification           General Usage
PrefixSuffixNamingRequirement
AllowGuestsToBeGroupOwner       False
AllowGuestsToAccessGroups       True
GuestUsageGuidelinesUrl
GroupCreationAllowedGroupId     A3c13e4d-7083-4448-9224-287f10f23e10
AllowToAddGuests                True
UsageGuidelinesUrl
http://office365itpros.com/GroupGuidelines.html
ClassificationList              General Usage,External Access,Internal
Only,Confidential
EnableGroupCreation             False
```

Sensitivity: Confidential

Updating Group Policy Setting for Group Creation

GUID for group specifying people who can create groups

```
[PS] C:\> $ObjectId = (Get-AzureADGroup
–SearchString
GroupCreationControl).ObjectId
```

```
[PS] C:\> $Settings = Get-
AzureADDirectorySetting | ?
{$_.DisplayName -eq "Group.Unified"}
```

Retrieve current policy settings

```
[PS] C:\>
$Settings[“EnableGroupCreation”] =
"False"
```

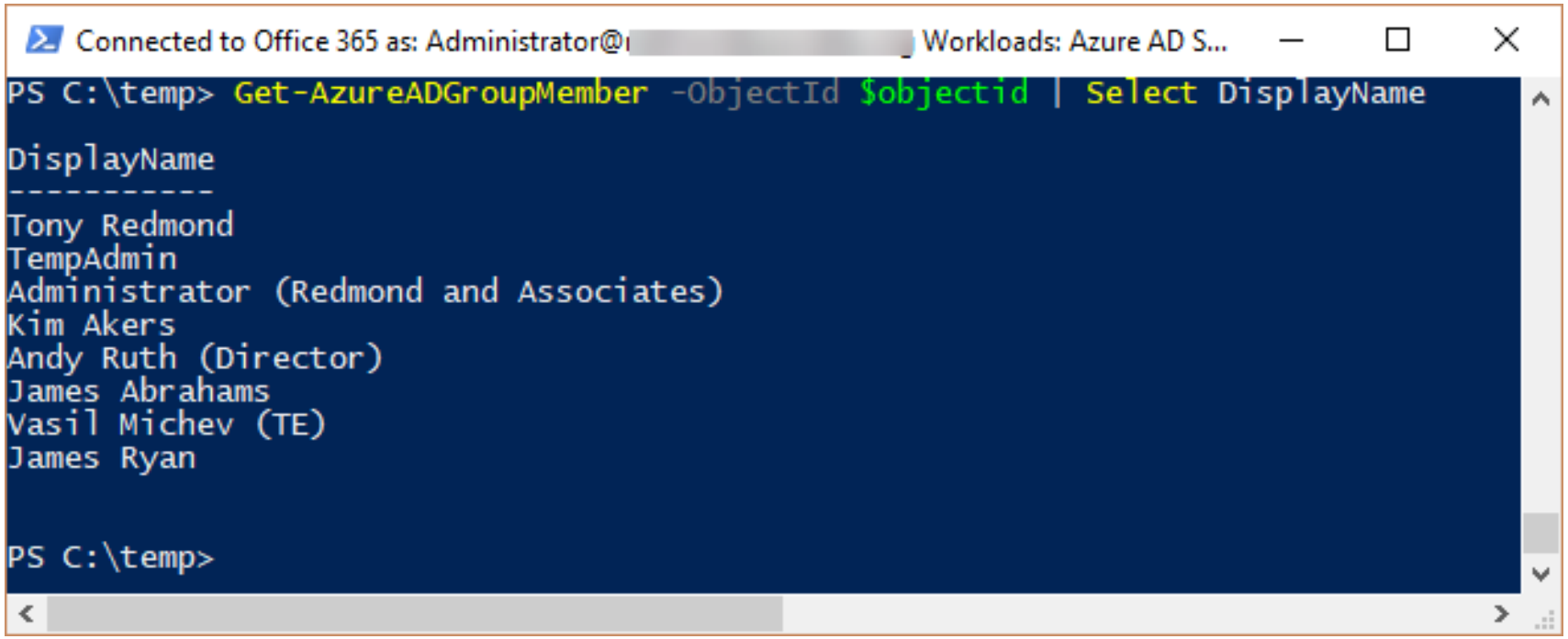
Block general group creation

```
[PS] C:\>
$Settings[“GroupCreationAllowedGroupId”]
= $ObjectId
```

Except for the group pointed to in $ObjectId

```
[PS] C:\> Set-AzureADDirectorySetting
-Id $Settings.Id -DirectorySetting
$Settings
```

And update the policy…



Who Can Create Groups and Teams?

```
Get-AzureADGroupMember –ObjectId $ObjectId | Format-Table
DisplayName, UserPrincipalName
```

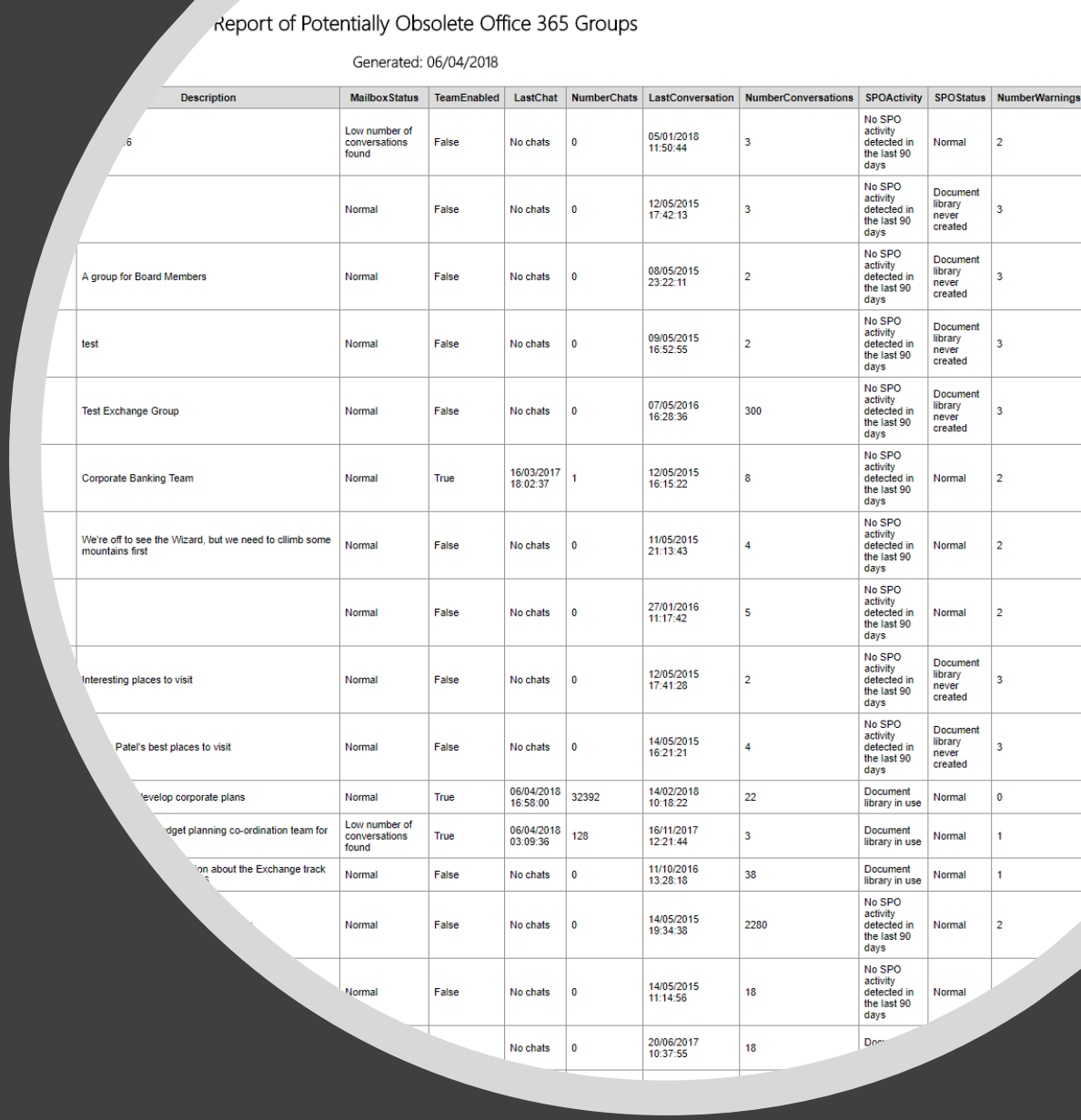
Some PowerShell Projects with Groups and Teams

Just to get your creative juices going…

Sensitivity: Confidential

Project: Find Inactive Groups and Teams

- Some groups and teams are created and then go into a state of disuse after a short period
- Goal is to find these groups and teams and then remove or archive them
- No functionality available in standard product
- To detect latest activity:
  - Look for SharePoint activity in Office 365 Audit Log
  - Look for conversations in group inbox
  - Look for Teams compliance records
- Generate nice report



Report of Potentially Obsolete Office 365 Groups

Generated: 06/04/2018

Description	MailboxStatus	TeamEnabled	LastChat	NumberChats	LastConversation	NumberConversations	SPOActivity	SPOStatus	NumberWarnings
6	Low number of conversations found	False	No chats	0	05/01/2018 11:50:44	3	No SPO activity detected in the last 90 days	Normal	2
	Normal	False	No chats	0	12/05/2015 17:42:13	3	No SPO activity detected in the last 90 days	Document library never created	3
A group for Board Members	Normal	False	No chats	0	08/05/2015 23:22:11	2	No SPO activity detected in the last 90 days	Document library never created	3
test	Normal	False	No chats	0	09/05/2015 16:52:55	2	No SPO activity detected in the last 90 days	Document library never created	3
Test Exchange Group	Normal	False	No chats	0	07/05/2016 16:28:36	300	No SPO activity detected in the last 90 days	Document library never created	3
Corporate Banking Team	Normal	True	16/03/2017 18:02:37	1	12/05/2015 16:15:22	8	No SPO activity detected in the last 90 days	Normal	2
We're off to see the Wizard, but we need to cllimb some mountains first	Normal	False	No chats	0	11/05/2015 21:13:43	4	No SPO activity detected in the last 90 days	Normal	2
	Normal	False	No chats	0	27/01/2016 11:17:42	5	No SPO activity detected in the last 90 days	Normal	2
Interesting places to visit	Normal	False	No chats	0	12/05/2015 17:41:28	2	No SPO activity detected in the last 90 days	Document library never created	3
Patel's best places to visit	Normal	False	No chats	0	14/05/2015 16:21:21	4	No SPO activity detected in the last 90 days	Document library never created	3
evelop corporate plans	Normal	True	06/04/2018 16:58:00	32392	14/02/2018 10:18:22	22	Document library in use	Normal	0
dget planning co-ordination team for	Low number of conversations found	True	06/04/2018 03:09:36	128	16/11/2017 12:21:44	3	Document library in use	Normal	1
on about the Exchange track	Normal	False	No chats	0	11/10/2016 13:28:18	38	Document library in use	Normal	1
	Normal	False	No chats	0	14/05/2015 19:34:38	2280	No SPO activity detected in the last 90 days	Normal	2
	Normal	False	No chats	0	14/05/2015 11:14:56	18	No SPO activity detected in the last 90 days	Normal	
			No chats	0	20/06/2017 10:37:55	18	Do...		

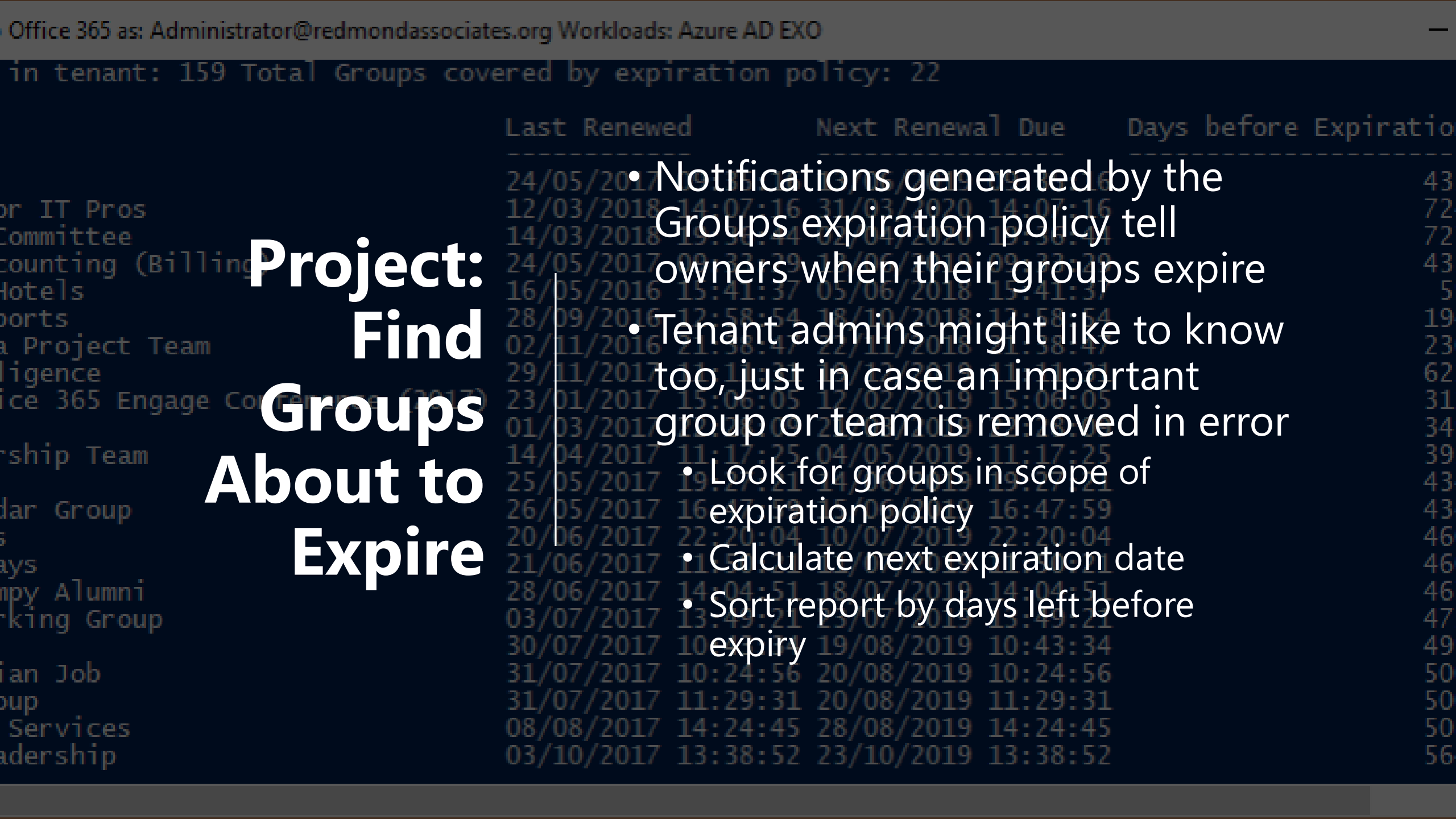
Sensitivity: Confidential

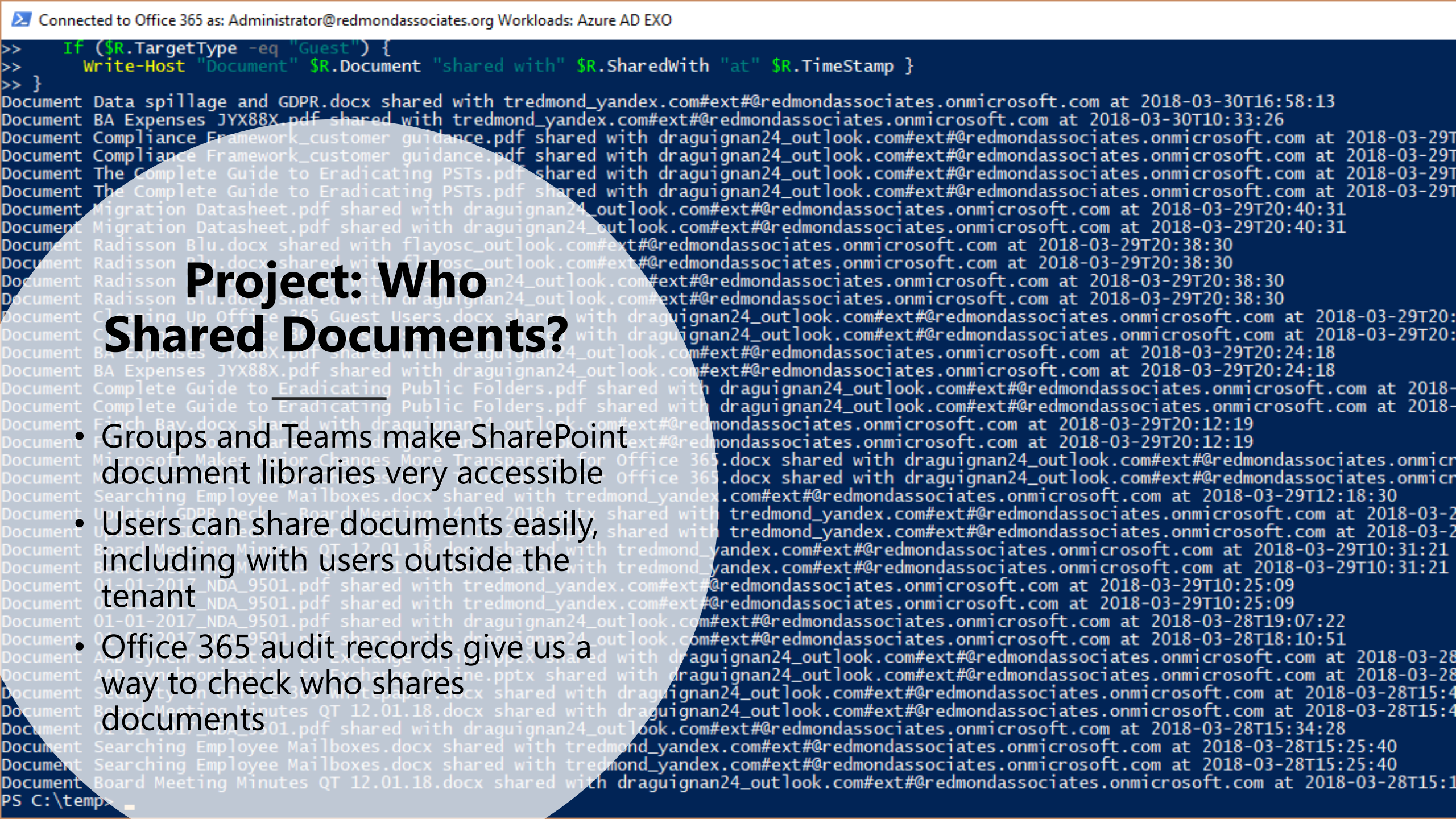
```
    Write-Host "Archiving" $AGroup.DisplayName -ForegroundColor Yellow
  } Else {
    Write-Host $CheckGroup "group not found - terminating"
    Return }

# Get lists of current owners and members
$CurrentOwners = (Get-UnifiedGroupLinks -Identity $AGroup.Alias -LinkType Owners | Select Name)
$CurrentMembers = (Get-UnifiedGroupLinks -Identity $AGroup.Alias -LinkType Members | Select Name)
# Add a new owner - this is the address of the account that will continue to access the group
$AdminAccount = "Administrator"
Add-UnifiedGroupLinks -Identity $AGroup.Alias -LinkType Members -Links $AdminAccount
Add-UnifiedGroupLinks -Identity $AGroup.Alias -LinkType Owners -Links $AdminAccount
# Remove the other members and owners
ForEach ($O in $CurrentOwners) {
        Remove-UnifiedGroupLinks -Identity $AGroup.Alias -LinkType Owners -Links $O.Name -Confirm:$False}
ForEach ($M in $CurrentMembers) {
        Remove-UnifiedGroupLinks -Identity $AGroup.Alias -LinkType Members -Links $M.Name -Confirm:$False}
# Create SMTP Address for the archived group
$OldSmtpAddress = $AGroup.PrimarySmtpAddress -Split "@"
$NewSmtpAddress = $OldSmtpAddress[0] +  "_archived" + "@" + $OldSmtpAddress[1]
$AddressRemove = "smtp:"+$AGroup.PrimarySmtpAddress
$ArchiveInfo = "Archived on " + (Get-Date)  + " by " + $o365cred.username
# Update Group properties
Set-UnifiedGroup -Identity $AGroup.Alias -AccessType Private -RequireSenderAuthenticationEnabled $True -Hidden
$True -CustomAttribute1 "Archived" -CustomAttribute2 $ArchiveInfo -PrimarySmtpAddress $NewSmtpAddress
Set-UnifiedGroup -Identity $AGroup.Alias -EmailAddresses @{remove=$AddressRemove}

Write-Host $AGroup.DisplayName "is now archived and" $AdminAccount "is the new group owner"
```

Project: Archive Obsolete Groups and Teams

- If we find obsolete groups or teams, we might want to keep them in an inactive state, just in case they are needed again
- Remove current membership
- Add new owner
- Replace SMTP address
- Set custom attribute to make it easy to find archived groups

Sensitivity: Confidential


Office 365 as: Administrator@redmondassociates.org Workloads: Azure AD EXO
in tenant: 159 Total Groups covered by expiration policy: 22
Last Renewed
Next Renewal Due
Days before Expiratio
Project: Find Groups About to Expire
• Notifications generated by the Groups expiration policy tell owners when their groups expire
• Tenant admins might like to know too, just in case an important group or team is removed in error
• Look for groups in scope of expiration policy
• Calculate next expiration date
• Sort report by days left before expiry


Connected to Office 365 as: Administrator@redmondassociates.org Workloads: Azure AD EXO
Project: Who Shared Documents?
• Groups and Teams make SharePoint document libraries very accessible
• Users can share documents easily, including with users outside the tenant
• Office 365 audit records give us a way to check who shares documents

Unified Audit Log

- Ingests data from multiple Office 365 sources and normalizes audit records
- Retained for 90 days (180 days with Advanced Security Management, or more with ISV products)
- Rich source for investigations about what happens inside Office 365

```
$Records = (Search-UnifiedAuditLog -StartDate 28-Mar-2018 -EndD
"SharingInvitationCreated", "SharingInvitationAccepted")
If ($Records.Count -eq 0) {
   Write-Host "No SharePoint sharing records found." }
 Else {
   Write-Host "Processing" $Records.Count " SharePoint sharing
   $Report = @()
   ForEach ($Rec in $Records) {
      $AuditData = ConvertFrom-Json $Rec.Auditdata
      Switch ($AuditData.TargetUserOrGroupType)
      {
         "Member" { $SharedType = "Tenant User" }
         "SharePointGroup" { $SharedType = "SPO Sharing Link" }
         "Guest"  { $SharedType = "Guest User" }
      }
      $ReportLine = [PSCustomObject][Ordered]@{
           TimeStamp    = $AuditData.CreationTime
           User         = $Rec.UserIds
           Action       = $AuditData.Operation
           Workload     = $AuditData.Workload
           L            = $AuditData.ObjectId
              ment      = $AuditData.SourceFileName
                dWith   = $AuditData.TargetUserOrGroupName
                 pe     = $SharedType
                        = $AuditData.ItemType
                        = $AuditData.EventData
                        = AuditData.TargetUserOrGroupType
```

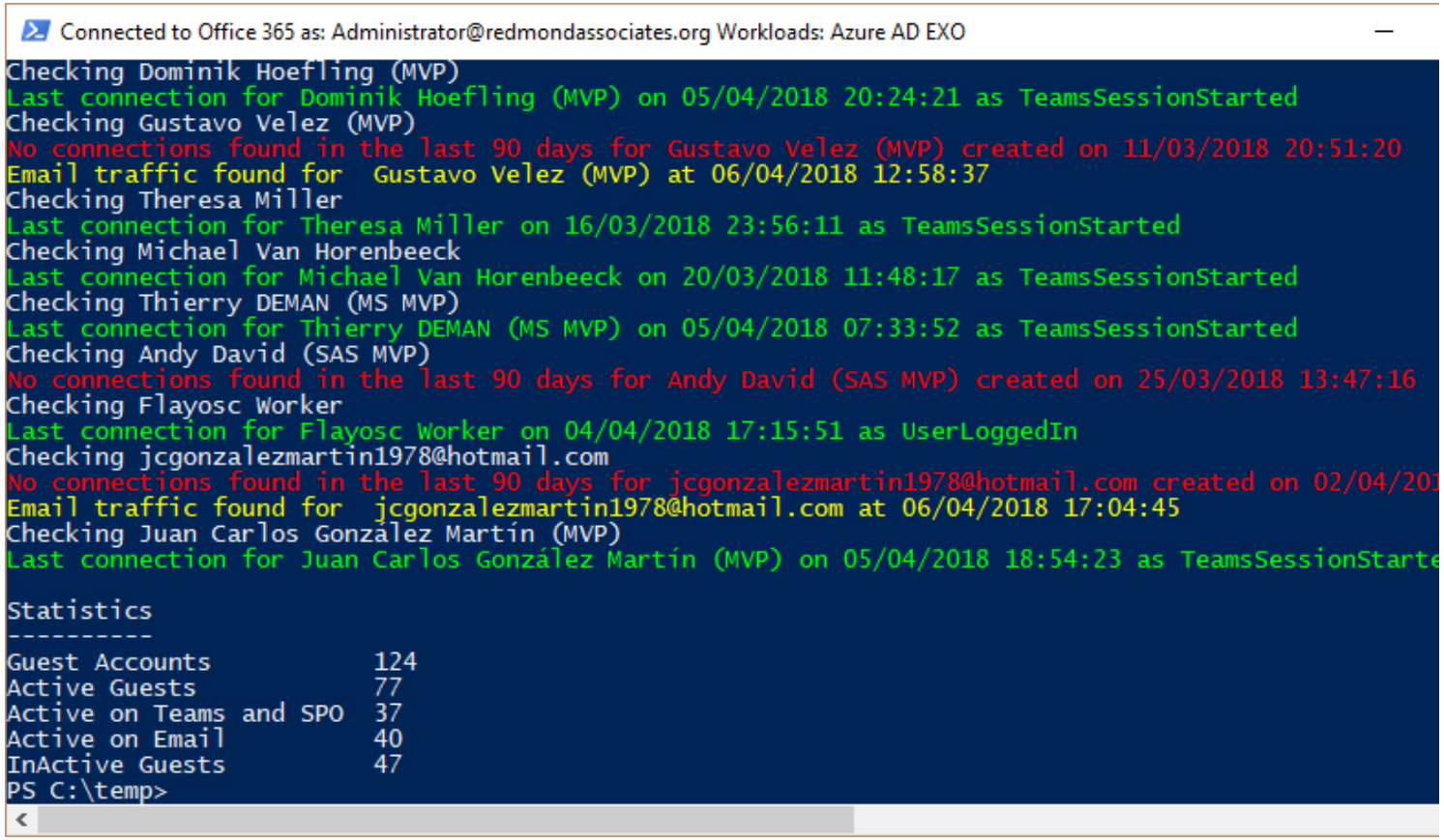
Sensitivity: Confidential

```
$Groups = (Get-UnifiedGroup -Filter {GroupExternalMemberCount -gt 0} | Select Alias, DisplayName, SharePointSiteURL, Gr
If ($Groups.Count -gt 0) {
   Write-Host "Processing" $Groups.Count "groups with guest members"
   $Report = @()
   $NumExt = 0
   $LargestGroup = $Null
   $LargestGroupNum = 0
   ForEach ($G in $Groups) {
      Write-Host "Processing" $G.DisplayName
      $Users = Get-UnifiedGroupLinks -Identity $G.Alias -LinkType Members
      ForEach ($U in $Users) {
         If ($U.Name -Match "#EXT#" -and $U.Name -NotLike "*teams.ms*") {
            $NumExt++
            $CheckName = $U.Name + "@redmondassociates.onmicrosoft.com"
            $User = (Get-AzureADUser -ObjectId $CheckName).DisplayName
            $ReportLine = [PSCustomObject][Ordered]@{
               Email          = $U.Name
               User           = $User
               Group          = $G.DisplayName
               Site           = $G.SharePointSiteURL }
            $Report += $ReportLine }
               }
    If ($G.GroupExternalMemberCount -gt $LargestGroupNum) {
       $LargestGroupNum = $G.GroupExternalMemberCount
       $LargestGroup = $G.DisplayName}
    }
Write-Host $NumExt "guest user memberships found in" $Groups.Count "groups"
Write-Host "Largest external group is" $LargestGroup "with" $LargestGroupNum "guests"
}
```

Project: Report Guest Users

- It's easy for Group Owners to add guests to Teams and Groups, but do you know who the guests are and where they come from?
- It's a bad idea to put very confidential information in a group that has guests, unless you intend to share that way

Sensitivity: Confidential



Project: Last Time Guest Users Connected

- Teams, Office 365 Groups, SharePoint Online, and Planner use Azure B2B Collaboration for guest user access
- How do you know whether guests actually connect and use resources in your tenant?

Sensitivity: Confidential

Things to Remember

- Lots of PowerShell examples and advice available on the internet
- Never download and run code without checking it first
- You don’t have to be an expert to screw things up…

Thanks!