**1. Import all dns record using powershell**

# Define the DNS zone details

$ZoneName = "lord.itzz2k.local"

$ImportPath = "C:\dns\_records.csv"

# Read the CSV file

$Records = Import-Csv -Path $ImportPath

# Iterate through each record in the CSV file

foreach ($Record in $Records) {

$RecordType = $Record.Type

$RecordName = $Record.Name

$RecordData = $Record.Data

# Create the DNS record string

$RecordString = "$RecordType,$RecordName,$RecordData"

# Use dnscmd command to add the DNS record

dnscmd /RecordAdd $ZoneName $RecordString

}

# Display a success message

Write-Host "DNS records imported successfully from $ImportPath"

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

**2. Export SamAccountName based on OU using powershell**

# Specify the list of OUs to fetch SamAccountName from

$OUs = "CN=Users,DC=lord,DC=itzz2k,DC=local"

# Create an array to store user objects

$UserObjects = @()

# Iterate through each OU

foreach ($OU in $OUs) {

# Retrieve user accounts in the OU and fetch SamAccountName attribute

$Users = Get-ADUser -Filter \* -SearchBase $OU -Properties SamAccountName

# Add user objects to the array

$UserObjects += $Users | Select-Object SamAccountName, DistinguishedName

}

# Export the user objects to a CSV file

$ExportPath = "C:\export\_samaccount.csv"

$UserObjects | Export-Csv -Path $ExportPath -NoTypeInformation

# Display a success message

Write-Host "User SamAccountNames exported to $ExportPath"

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

**3. GPO migration**

Link: https://www.urtech.ca/2012/08/solved-how-to-move-group-policy-objects-gpos-between-domains/

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

**4. User installation disabled via policy**

Link: https://support.solarwinds.com/SuccessCenter/s/article/User-installations-are-disabled-via-policy-on-the-machine?language=en\_US

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

**5. Configure NTP server**

Link: https://computingforgeeks.com/how-to-configure-ntp-server-in-windows-server/

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

**6. Export proxy addresses using powershell based on OU**

Get-ADUser -Filter \* -SearchBase ‘CN=Users,DC=lord,DC=itzz2k,DC=local’ -Properties proxyaddresses,mail | Select-Object Samaccountname,mail, @{L = "ProxyAddresses"; E = { ($\_.ProxyAddresses -like 'smtp:\*') -join ";"}} | Export-Csv -Path C:\AdUsersProxyAddresses.csv -NoTypeInformation

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

**7. Import proxy addresses using powershell**

# Specify the path to the CSV file containing proxy addresses

$ImportPath = "C:\AdUsersProxyAddresses.csv"

# Import the user proxy addresses from the CSV file

$ProxyAddresses = Import-Csv -Path $ImportPath

# Iterate through each entry in the CSV file

foreach ($Entry in $ProxyAddresses) {

$UserSamAccountName = $Entry.SamAccountName

$UserProxyAddresses = $Entry.ProxyAddresses -split ";"

# Retrieve the user account and update proxy addresses

$User = Get-ADUser -Identity $UserSamAccountName

$User.ProxyAddresses = $UserProxyAddresses

Set-ADUser -Instance $User

}

# Display a success message

Write-Host "Proxy addresses imported successfully from $ImportPath"

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

**8. LastLoginDate (Inactive system) using powershell**

$Days = 180

$Time = (Get-Date).Adddays(-($Days))

Get-ADComputer -Filter {LastLogonTimeStamp -lt $Time} -Properties \* | Select Name, LastLogonDate | Export-CSV -Path C:\inactive\_System.csv

**9. Disable password change at next logon using powershell**

# Import the Active Directory module

Import-Module ActiveDirectory

# Get all user accounts

$Users = Get-ADUser -Filter \* -Properties PasswordExpired

# Iterate through each user account and disable the "Password must change at next logon" flag

foreach ($User in $Users) {

if ($User.PasswordExpired) {

$User | Set-ADUser -ChangePasswordAtLogon $false

Write-Host "Disabled password change at next logon for user $($User.SamAccountName)"

}

}

# Display a success message

Write-Host "Password change at next logon disabled for all users."

**10. Export email address using powershell**

# Import the Active Directory module

Import-Module ActiveDirectory

# Get all user objects with their email addresses

$usersWithEmails = Get-ADUser -Filter \* -Properties EmailAddress | Select-Object Name, SamAccountName, EmailAddress

# Specify the path for the output CSV file

$csvFilePath = "C:\Path\To\Output\File.csv"

# Export the data to a CSV file

$usersWithEmails | Export-Csv -Path $csvFilePath -NoTypeInformation

Write-Host "Email addresses have been exported to $csvFilePath"

**11. Import email address using powershell**

# Import the Active Directory module

Import-Module ActiveDirectory

# Specify the path to the CSV file

$csvFilePath = "C:\Path\To\Input\File.csv"

# Read the CSV file

$usersWithEmails = Import-Csv -Path $csvFilePath

# Loop through each row in the CSV and update the email address for each user

foreach ($user in $usersWithEmails) {

$samAccountName = $user.SamAccountName

$emailAddress = $user.EmailAddress

# Check if the user exists in Active Directory

$adUser = Get-ADUser -Filter { SamAccountName -eq $samAccountName }

if ($adUser) {

# Update the email address for the user

Set-ADUser -Identity $samAccountName -EmailAddress $emailAddress

Write-Host "Email address updated for user: $samAccountName"

} else {

Write-Host "User not found: $samAccountName"

}

}

**12. Fetch immutable id**

# Import the Active Directory module

Import-Module ActiveDirectory

# Path to the CSV file containing user data (username and UPN)

$csvFilePath = "C:\path\to\users.csv"

# Read the CSV file and fetch the immutable ID for each user

$users = Import-Csv -Path $csvFilePath

# Array to store the user data with Immutable ID

$usersWithImmutableID = @()

# Function to get the immutable ID of a user from Active Directory

function Get-ImmutableID {

param(

[string]$UserPrincipalName

)

# Search for the user in Active Directory by UPN

$user = Get-ADUser -Filter { UserPrincipalName -eq $UserPrincipalName } -Properties objectGUID

# If the user is found, convert the objectGUID to Immutable ID format

if ($user) {

$immutableID = [System.Guid]::New($user.objectGUID).ToString('N')

return $immutableID

} else {

Write-Warning "User not found in Active Directory: $UserPrincipalName"

return $null

}

}

# Fetch the immutable ID for each user in the CSV file

foreach ($user in $users) {

$upn = $user.UserPrincipalName

$immutableID = Get-ImmutableID -UserPrincipalName $upn

if ($immutableID) {

# Add the user details with Immutable ID to the array

$user | Add-Member -NotePropertyName "ImmutableID" -NotePropertyValue $immutableID

$usersWithImmutableID += $user

}

}

# Export the result to a new CSV file

$exportPath = "C:\path\to\users\_with\_immutable\_id.csv"

$usersWithImmutableID | Export-Csv -Path $exportPath -NoTypeInformation

Write-Host "Results exported to: $exportPath"

**13. Hard synchronization**

# Install the MSOnline PowerShell module (if not already installed)

# Install-Module MSOnline

# Connect to your Azure AD tenant

Connect-MsolService

# Path to the CSV file containing user data (username, UPN, and immutable ID)

$csvFilePath = "C:\path\to\users.csv"

# Read the CSV file and synchronize users with their immutable IDs

$users = Import-Csv -Path $csvFilePath

foreach ($user in $users) {

$upn = $user.UserPrincipalName

$immutableID = $user.ImmutableID

# Create or update the user object with immutable ID

try {

Set-MsolUser -UserPrincipalName $upn -ImmutableId $immutableID

Write-Host "Successfully synced user: $upn"

} catch {

Write-Host "Failed to sync user: $upn. Error: $($\_.Exception.Message)"

}

}

# Verify the synchronization and check for any errors

Get-MsolUser -All | Select UserPrincipalName,ImmutableId

**14. Hard sync for single user**

Go to Local AD server, and open PowerShell as admin & run below command.

$ADUser = "bhaveshmmehta"

$guid =(Get-ADUser $ADUser).Objectguid

$immutableID=[system.convert]::ToBase64String($guid.tobytearray())

$immutableID

As you run those commands you will get Immutable id, example : ZR2/tEXekUGy+gsfarWimw==

Save it and in PowerShell run, Connect-MsolService

Log in with Global Admin credentials, and run below commands,

Set-MsolUser -UserPrincipalName [bhaveshmmehta@astralgroup.onmicrosoft.com](mailto:bhaveshmmehta@astralgroup.onmicrosoft.com) -ImmutableId "ZR2/tEXekUGy+gsfarWimw=="

Change immutable id which we have saved earlier and user id accordingly([bhaveshmmehta2815@astralgroup.onmicrosoft.com](mailto:bhaveshmmehta2815@astralgroup.onmicrosoft.com))

Run delta sync.

**15. SQL Express install using powershell**

function Install-SQLServerExpress2019 {

Write-Host "Downloading SQL Server Express 2019..."

$Path = $env:TEMP

$Installer = "SQL2019-SSEI-Expr.exe"

$URL = "https://go.microsoft.com/fwlink/?linkid=866658"

Invoke-WebRequest $URL -OutFile $Path\$Installer

Write-Host "Installing SQL Server Express..."

Start-Process -FilePath $Path\$Installer -Args "/ACTION=INSTALL /IACCEPTSQLSERVERLICENSETERMS /QUIET" -Verb RunAs -Wait

Remove-Item $Path\$Installer

}

Install-SQLServerExpress2019

**16. Export OUs to CSV using PowerShell**

Get-ADOrganizationalUnit -Filter 'Name -like "\*"' -Properties \* | Select -Property Name, DistinguishedName | Export-Csv -Path C:\export\_OUs.csv -NoTypeInformation -Encoding UTF8

**17. Import Ous using PowerShell**

# Replace "Path\to\OUs.csv" with the actual path to your CSV file

$csvPath = "Path\to\OUs.csv"

# Read the CSV file and store the OU names and paths in an array

$ouData = Import-Csv $csvPath

# Connect to Active Directory if you haven't already done so

Import-Module ActiveDirectory

# Loop through the array of OU data and create each OU

foreach ($ouInfo in $ouData) {

$ouName = $ouInfo.OUName

$ouPath = $ouInfo.OUPath

New-ADOrganizationalUnit -Name $ouName -Path $ouPath

Write-Host "OU '$ouName' created at $ouPath"

}

**18. Password Never Expires**

# Import the Active Directory module if not already loaded

Import-Module ActiveDirectory

# Define the domain

$domain = "merittrac.net" # Replace with the actual domain name

# Get all user accounts in the domain

$users = Get-AdUser -Filter {(ObjectClass -eq "user")} -SearchBase "DC=$domain,DC=com"

# Loop through each user and set the "Password never expires" option

foreach ($user in $users) {

if (-not $user.PasswordNeverExpires) {

Set-AdUser -Identity $user -PasswordNeverExpires $true

Write-Host "Password for user '$($user.SamAccountName)' in domain '$domain' will never expire."

} else {

Write-Host "Password for user '$($user.SamAccountName)' in domain '$domain' is already set to never expire."

}

}

Write-Host "Password never expires option has been set for all user accounts in domain '$domain'."

**19. Export Users from OU**

# Import the Active Directory module

Import-Module ActiveDirectory

# Specify the OU path you want to export users from

$ouPath = "OU=Technology,OU=UNext Users,DC=unext,DC=net"

# Specify the output file path and name

$outputFile = "C:\TargetExportedUsers.csv"

# Get users from the specified OU

$users = Get-ADUser -Filter \* -SearchBase $ouPath -Properties \*

# Export user information to a CSV file

$users | Select-Object SamAccountName, GivenName, Surname, DisplayName, UserPrincipalName, EmailAddress, Enabled | Export-Csv -Path $outputFile -NoTypeInformation

# Display a message indicating the export is complete

Write-Host "Users exported to $outputFile"

**20. User cannot change Password**

# Import the Active Directory module if not already loaded

Import-Module ActiveDirectory

# Define the domain

$domain = "merittrac.net" # Replace with the actual domain name

# Get all user accounts in the domain

$users = Get-AdUser -Filter {(ObjectClass -eq "user")} -SearchBase "DC=$domain,DC=com" -Properties "UserCannotChangePassword"

# Loop through each user and set the "User cannot change password" option

foreach ($user in $users) {

if (-not $user.UserCannotChangePassword) {

Set-AdUser -Identity $user -CannotChangePassword $true

Write-Host "User '$($user.SamAccountName)' in domain '$domain' cannot change their password."

} else {

Write-Host "User '$($user.SamAccountName)' in domain '$domain' already cannot change their password."

}

}

Write-Host "The 'User cannot change password' option has been set for all user accounts in domain '$domain'."

**21. Delete Users from Active Directory Admin Center**

Get-ADObject -Filter 'isDeleted -eq $true -and Name -like "\*DEL:\*"' -IncludeDeletedObjects | Remove-ADObject -Confirm:$false

**22. Command to disable SID Filtering**

NETDOM TRUST india.com /Domain:bitscape.ishan-kansara.in /Quarantine:No

**23. Command to display SID**

Get-LocalUser -Name 'tenders@merittrac.com' | Select-Object sid

**24. Get SID History**

Get-ADUser -Identity urack -Properties SidHistory | Select-Object -ExpandProperty SIDHistory

**25. Command to check status of SID Filtering**

netdom trust ishan.india.com /domain:bharat.com /quarantine

**26. Change UPN Suffix**

# Define the old and new UPN suffixes

#$oldSuffix = "@merittrac.net"

#$newSuffix = "@merittrac.com"

# Retrieve a list of users whose UPN suffix matches the old suffix in a specific OU

$users = Get-ADUser -Filter {UserPrincipalName -like "\*@merittrac.net"} -SearchBase "DC=merittrac,DC=net"

# Iterate through the list of users and update their UPN suffix

foreach ($user in $users) {

$newUPN = $user.UserPrincipalName.Replace("@merittrac.net", "@merittrac.com")

# Update the UPN suffix for the user

Set-ADUser -Identity $user -UserPrincipalName $newUPN

Write-Host "Changed UPN for $($user.SamAccountName) from $($user.UserPrincipalName) to $newUPN"

}

**27. Get all DNS Record from the Zone**

Get-DnsServerResourceRecord -ZoneName "contoso.com" -RRType "NS" -Node

**28. Exporting NameServer Records**

Import-Module DnsServer

$dnsZone = "shriram.in"

$nsRecords = Get-DnsServerResourceRecord -ZoneName $dnsZone -RRType "NS"

$nsRecords | Select-Object HostName, RecordType, TimeToLive, @{Name='RecordData'; Expression={$\_.RecordData.NameServer}} | Export-Csv -Path "c:\NSRecords.csv" -NoTypeInformation

**29. Add DNS Name Server to Zone through PowerShell**

# Import the DNS Server module if it's not already imported

Import-Module DnsServer

# Specify the path to the CSV file

$csvFilePath = "C:\records33.csv"

# Import the CSV data

$nsRecords = Import-Csv -Path $csvFilePath

# Loop through each record in the CSV and add NS records

foreach ($record in $nsRecords) {

$domain = $record.Domain

$nameServer = $record.NameServer

# Add the NS record to the DNS server

Add-DnsServerResourceRecord -ZoneName $domain -NS -Name "@" -NameServer $nameServer

# Output a message indicating the addition of each NS record

Write-Host "Added NS record for $domain with name server $nameServer"

}

**30. Remove Name Server Records**

Get-DnsServerZone | ForEach-Object { Get-DnsServerResourceRecord -ZoneName $\_.ZoneName -RRType Ns | Where-Object {$\_.RecordData.NameServer -like '\*.ishan.shriram.in.'} | Remove-DnsServerResourceRecord -ZoneName $\_.ZoneName -Confirm:$false }

**30. Decomissioning Server Commands:**

ntdsutil

metadata cleanup

connections

connect to server SD-VM-MIG.shriram.in

q

select operation target

list domains

select domain 8

list sites

select site 1

list servers in site

select server 0

q

remove selected server

repadmin /syncall