AD Tiering model – how to deploy that

azureblog.pl/2020/04/13/deploying-tiering

Robert Przybylski

Hi

This article is a kind of appendix to my previous article with a presentation from WGUIS 119 about the AD Tiering Model.

Here I will focus on how to deploy tiering in a proper way.

Ready, Steady, Study

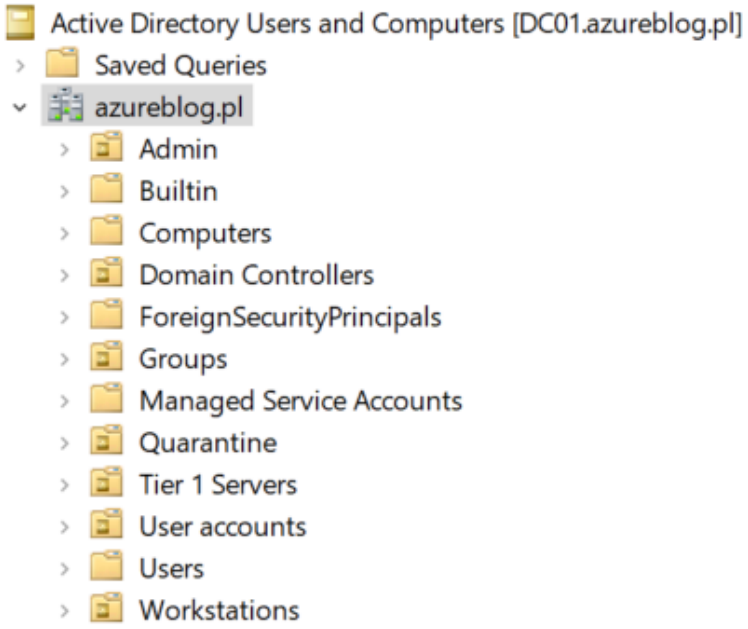
All the required scripts you can find under my GitHub repo:
https://github.com/przybylskirobert/ADSecurity/tree/master/Tiering

Before running code for OU creation, let's properly setup the directory and do other configuration changes.

```
$location = Get-Location
Set-Location C:\Tools
Import-Module ActiveDirectory
$dNC = (Get-ADRootDSE).defaultNamingContext
```

OU Creation

The first thing to create is a proper OU Structure like in the picture below



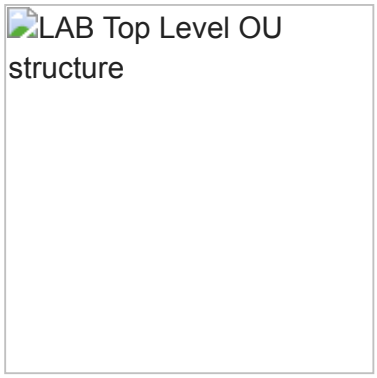
LAB Top Level OU structure

We can create this using the following code

```
$OUs = @(
$(New-Object PSObject -Property @{Name = "Admin"; ParentOU = "" }),
$(New-Object PSObject -Property @{Name = "Groups"; ParentOU = "" }),
$(New-Object PSObject -Property @{Name = "Tier 1 Servers"; ParentOU = "" }),
$(New-Object PSObject -Property @{Name = "Workstations"; ParentOU = "" }),
$(New-Object PSObject -Property @{Name = "User accounts"; ParentOU = "" }),
$(New-Object PSObject -Property @{Name = "Quarantine"; ParentOU = "" })
)
.\Create-OU.ps1 -OUs $OUs -Verbose
```

As an output, we should receive information about creating new OU or information that OU already exist

LAB Top-Level OU's Creation



As you may be noticed, we are using a custom script called ***Create-OU*** to create new organizational units. I'd like to stop here for a minute and describe the parameters of this script

Create-OU script is using ***OU*** variable, which is an array of PSObjects. Those objects contain 2 properties:

- ***Name*** – Name of the OU that we would like to create.
- ***ParentOU***– part of the path to the parent ou, without domain distinguished name.

Because we were creating top-level OU's, we didn't provide values for ParentOU property.

Now we can proceed with other OU's creation.
Our goal is to create all required OU's under **Admin** top-level OU

For that, we will use the same **Create-OU** script but with different values

```
$OUs = @(
$(New-Object PSObject -Property @{Name = "Tier0"; ParentOU = "ou=Admin" }),
$(New-Object PSObject -Property @{Name = "Tier1"; ParentOU = "ou=Admin" }),
$(New-Object PSObject -Property @{Name = "Tier2"; ParentOU = "ou=Admin" }),
$(New-Object PSObject -Property @{Name = "Accounts"; ParentOU =
"ou=Tier0,ou=Admin" }),
$(New-Object PSObject -Property @{Name = "Groups"; ParentOU =
"ou=Tier0,ou=Admin" }),
$(New-Object PSObject -Property @{Name = "Service Accounts"; ParentOU =
"ou=Tier0,ou=Admin" }),
$(New-Object PSObject -Property @{Name = "Devices"; ParentOU =
"ou=Tier0,ou=Admin" }),
$(New-Object PSObject -Property @{Name = "Tier0 Servers"; ParentOU =
"ou=Tier0,ou=Admin" }),
$(New-Object PSObject -Property @{Name = "Accounts"; ParentOU =
"ou=Tier1,ou=Admin" }),
$(New-Object PSObject -Property @{Name = "Groups"; ParentOU =
"ou=Tier1,ou=Admin" }),
$(New-Object PSObject -Property @{Name = "Service Accounts"; ParentOU =
"ou=Tier1,ou=Admin" }),
$(New-Object PSObject -Property @{Name = "Devices"; ParentOU =
"ou=Tier1,ou=Admin" }),
$(New-Object PSObject -Property @{Name = "Accounts"; ParentOU =
"ou=Tier2,ou=Admin" }),
$(New-Object PSObject -Property @{Name = "Groups"; ParentOU =
"ou=Tier2,ou=Admin" }),
$(New-Object PSObject -Property @{Name = "Service Accounts"; ParentOU =
"ou=Tier2,ou=Admin" }),
$(New-Object PSObject -Property @{Name = "Devices"; ParentOU =
"ou=Tier2,ou=Admin" })
)
.\Create-OU.ps1 -OUs $OUs -Verbose
```

Output below

```
PS C:\Tools> $OUs = @(
>>     $(New-Object PSObject -Property @{Name = "Tier0"; ParentOU = "ou=Admin" }),
>>     $(New-Object PSObject -Property @{Name = "Tier1"; ParentOU = "ou=Admin" }),
>>     $(New-Object PSObject -Property @{Name = "Tier2"; ParentOU = "ou=Admin" }),
>>     $(New-Object PSObject -Property @{Name = "Accounts"; ParentOU = "ou=Tier0,ou=Admin" }),
>>     $(New-Object PSObject -Property @{Name = "Groups"; ParentOU = "ou=Tier0,ou=Admin" }),
>>     $(New-Object PSObject -Property @{Name = "Service Accounts"; ParentOU = "ou=Tier0,ou=Admin" }),
>>     $(New-Object PSObject -Property @{Name = "Devices"; ParentOU = "ou=Tier0,ou=Admin" }),
>>     $(New-Object PSObject -Property @{Name = "Tier0 Servers"; ParentOU = "ou=Tier0,ou=Admin" }),
>>     $(New-Object PSObject -Property @{Name = "Accounts"; ParentOU = "ou=Tier1,ou=Admin" }),
>>     $(New-Object PSObject -Property @{Name = "Groups"; ParentOU = "ou=Tier1,ou=Admin" }),
>>     $(New-Object PSObject -Property @{Name = "Service Accounts"; ParentOU = "ou=Tier1,ou=Admin" }),
>>     $(New-Object PSObject -Property @{Name = "Devices"; ParentOU = "ou=Tier1,ou=Admin" }),
>>     $(New-Object PSObject -Property @{Name = "Accounts"; ParentOU = "ou=Tier2,ou=Admin" }),
>>     $(New-Object PSObject -Property @{Name = "Groups"; ParentOU = "ou=Tier2,ou=Admin" }),
>>     $(New-Object PSObject -Property @{Name = "Service Accounts"; ParentOU = "ou=Tier2,ou=Admin" }),
>>     $(New-Object PSObject -Property @{Name = "Devices"; ParentOU = "ou=Tier2,ou=Admin" })
>> )
PS C:\Tools> .\Create-OU.ps1 -OUs $OUs -Verbose
VERBOSE: OU 'Tier0' already exists under 'ou=Admin,DC=azureblog,DC=pl'
VERBOSE: OU 'Tier1' already exists under 'ou=Admin,DC=azureblog,DC=pl'
VERBOSE: OU 'Tier2' already exists under 'ou=Admin,DC=azureblog,DC=pl'
VERBOSE: OU 'Accounts' already exists under 'ou=Tier0,ou=Admin,DC=azureblog,DC=pl'
VERBOSE: OU 'Groups' already exists under 'ou=Tier0,ou=Admin,DC=azureblog,DC=pl'
VERBOSE: Creating new OU 'OU=Service Accounts,ou=Tier0,ou=Admin,DC=azureblog,DC=pl'
VERBOSE: OU 'Devices' already exists under 'ou=Tier0,ou=Admin,DC=azureblog,DC=pl'
VERBOSE: Creating new OU 'OU=Tier0 Servers,ou=Tier0,ou=Admin,DC=azureblog,DC=pl'
VERBOSE: OU 'Accounts' already exists under 'ou=Tier1,ou=Admin,DC=azureblog,DC=pl'
VERBOSE: OU 'Groups' already exists under 'ou=Tier1,ou=Admin,DC=azureblog,DC=pl'
VERBOSE: Creating new OU 'OU=Service Accounts,ou=Tier1,ou=Admin,DC=azureblog,DC=pl'
VERBOSE: OU 'Devices' already exists under 'ou=Tier1,ou=Admin,DC=azureblog,DC=pl'
VERBOSE: OU 'Accounts' already exists under 'ou=Tier2,ou=Admin,DC=azureblog,DC=pl'
VERBOSE: OU 'Groups' already exists under 'ou=Tier2,ou=Admin,DC=azureblog,DC=pl'
VERBOSE: Creating new OU 'OU=Service Accounts,ou=Tier2,ou=Admin,DC=azureblog,DC=pl'
VERBOSE: Creating new OU 'OU=Devices,ou=Tier2,ou=Admin,DC=azureblog,DC=pl'
```

LAB Sub-OU's creation under Admin OU

After Create-OU script usage we should have the following structure under Admin OU

LAB Admin OU Structure

Let's focus on sub-OU's under Admin OU
Inside Admin OU we have 3 main sub-OU's

- Tier0
- Tier1
- Tier2

These are the OU's representing the Tiering structure for admin resources like accounts, groups, service accounts, etc.

Inside each Tier OU, you can find 4 common OU's

- **Accounts** – for all user accounts in the tier
- **Devices** – for all computer objects in the tier
- **Groups** – for all groups in the tier
- **Service Accounts** – for all service accounts in the tier

For Tier0, we have one more OU called **Tier0 Servers**.
This OU should contain all servers marked as Tier 0, that are not Domain Controllers (e.g., CA servers, AD Connect Servers, AD FS, dedicated WSUS/SCCM).

Those servers might exist under separate sub-OU's inside Tier0 Servers OU

Now we will speed-up little bit and create Sub-OU's for top-level OU's like:

- **Groups**
- **Tier 1 Servers**
- **Workstations**

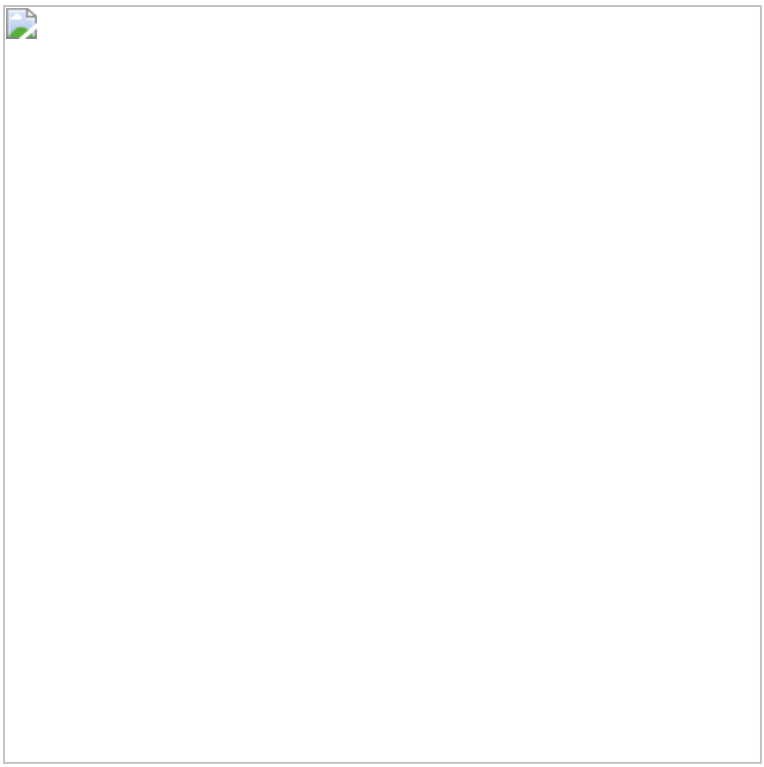
One more time we will use our well-known script **Create-OU**

```
$OUs = @(
$(New-Object PSObject -Property @{Name = "Security Groups"; ParentOU =
"ou=Groups" }),
$(New-Object PSObject -Property @{Name = "Distribution Groups"; ParentOU =
"ou=Groups" }),
$(New-Object PSObject -Property @{Name = "Contacts"; ParentOU = "ou=Groups"
})
)
.\Create-OU.ps1 -OUs $OUs -Verbose
$OUs = @(
$(New-Object PSObject -Property @{Name = "Application"; ParentOU = "ou=Tier 1
Servers" }),
$(New-Object PSObject -Property @{Name = "Collaboration"; ParentOU = "ou=Tier
1 Servers" }),
$(New-Object PSObject -Property @{Name = "Database"; ParentOU = "ou=Tier 1
Servers" }),
$(New-Object PSObject -Property @{Name = "Messaging"; ParentOU = "ou=Tier 1
Servers" }),
$(New-Object PSObject -Property @{Name = "Staging"; ParentOU = "ou=Tier 1
Servers" })
)
.\Create-OU.ps1 -OUs $OUs -Verbose
$OUs = @(
$(New-Object PSObject -Property @{Name = "Desktops"; ParentOU =
"ou=Workstations" }),
$(New-Object PSObject -Property @{Name = "Kiosks"; ParentOU =
"ou=Workstations" }),
$(New-Object PSObject -Property @{Name = "Laptops"; ParentOU =
"ou=Workstations" }),
$(New-Object PSObject -Property @{Name = "Staging"; ParentOU =
"ou=Workstations" })
)
.\Create-OU.ps1 -OUs $OUs -Verbose
$OUs = @(
$(New-Object PSObject -Property @{Name = "Enabled Users"; ParentOU = "ou=User
Accounts" }),
$(New-Object PSObject -Property @{Name = "Disabled Users"; ParentOU =
"ou=User Accounts" })
)
.\Create-OU.ps1 -OUs $OUs -Verbose
```

Output below

```
PS C:\Tools> $OUs = @(
>>     $(New-Object PSObject -Property @{Name = "Security Groups"; ParentOU = "ou=Groups" }),
>>     $(New-Object PSObject -Property @{Name = "Distribution Groups"; ParentOU = "ou=Groups" }),
>>     $(New-Object PSObject -Property @{Name = "Contacts"; ParentOU = "ou=Groups" })
>> )
PS C:\Tools> .\Create-OU.ps1 -OUs $OUs -Verbose
VERBOSE: Creating new OU 'OU=Security Groups,ou=Groups,DC=azureblog,DC=pl'
VERBOSE: Creating new OU 'OU=Distribution Groups,ou=Groups,DC=azureblog,DC=pl'
VERBOSE: Creating new OU 'OU=Contacts,ou=Groups,DC=azureblog,DC=pl'
PS C:\Tools> $OUs = @(
>>     $(New-Object PSObject -Property @{Name = "Application"; ParentOU = "ou=Tier 1 Servers" }),
>>     $(New-Object PSObject -Property @{Name = "Collaboration"; ParentOU = "ou=Tier 1 Servers" }),
>>     $(New-Object PSObject -Property @{Name = "Database"; ParentOU = "ou=Tier 1 Servers" }),
>>     $(New-Object PSObject -Property @{Name = "Messaging"; ParentOU = "ou=Tier 1 Servers" }),
>>     $(New-Object PSObject -Property @{Name = "Staging"; ParentOU = "ou=Tier 1 Servers" })
>> )
PS C:\Tools> .\Create-OU.ps1 -OUs $OUs -Verbose
VERBOSE: Creating new OU 'OU=Application,ou=Tier 1 Servers,DC=azureblog,DC=pl'
VERBOSE: Creating new OU 'OU=Collaboration,ou=Tier 1 Servers,DC=azureblog,DC=pl'
VERBOSE: Creating new OU 'OU=Database,ou=Tier 1 Servers,DC=azureblog,DC=pl'
VERBOSE: Creating new OU 'OU=Messaging,ou=Tier 1 Servers,DC=azureblog,DC=pl'
VERBOSE: Creating new OU 'OU=Staging,ou=Tier 1 Servers,DC=azureblog,DC=pl'
PS C:\Tools> $OUs = @(
>>     $(New-Object PSObject -Property @{Name = "Desktops"; ParentOU = "ou=Workstations" }),
>>     $(New-Object PSObject -Property @{Name = "Kiosks"; ParentOU = "ou=Workstations" }),
>>     $(New-Object PSObject -Property @{Name = "Laptops"; ParentOU = "ou=Workstations" }),
>>     $(New-Object PSObject -Property @{Name = "Staging"; ParentOU = "ou=Workstations" })
>> )
PS C:\Tools> .\Create-OU.ps1 -OUs $OUs -Verbose
VERBOSE: Creating new OU 'OU=Desktops,ou=Workstations,DC=azureblog,DC=pl'
VERBOSE: Creating new OU 'OU=Kiosks,ou=Workstations,DC=azureblog,DC=pl'
VERBOSE: Creating new OU 'OU=Laptops,ou=Workstations,DC=azureblog,DC=pl'
VERBOSE: Creating new OU 'OU=Staging,ou=Workstations,DC=azureblog,DC=pl'
PS C:\Tools> .\Create-OU.ps1 -OUs $OUs -Verbose
VERBOSE: OU 'Desktops' already exists under 'ou=Workstations,DC=azureblog,DC=pl'
VERBOSE: OU 'Kiosks' already exists under 'ou=Workstations,DC=azureblog,DC=pl'
VERBOSE: OU 'Laptops' already exists under 'ou=Workstations,DC=azureblog,DC=pl'
VERBOSE: OU 'Staging' already exists under 'ou=Workstations,DC=azureblog,DC=pl'
PS C:\Tools> ^C
PS C:\Tools> $OUs = @(
>>     $(New-Object PSObject -Property @{Name = "Enabled Users"; ParentOU = "ou=User Accounts" }),
>>     $(New-Object PSObject -Property @{Name = "Disabled Users"; ParentOU = "ou=User Accounts" })
>> )
PS C:\Tools> .\Create-OU.ps1 -OUs $OUs -Verbose
VERBOSE: Creating new OU 'OU=Enabled Users,ou=User Accounts,DC=azureblog,DC=pl'
VERBOSE: Creating new OU 'OU=Disabled Users,ou=User Accounts,DC=azureblog,DC=pl'
```

LAB Sub-OU's creation



LAB Top-level OU's structure

In this moment we have required OU structure created.

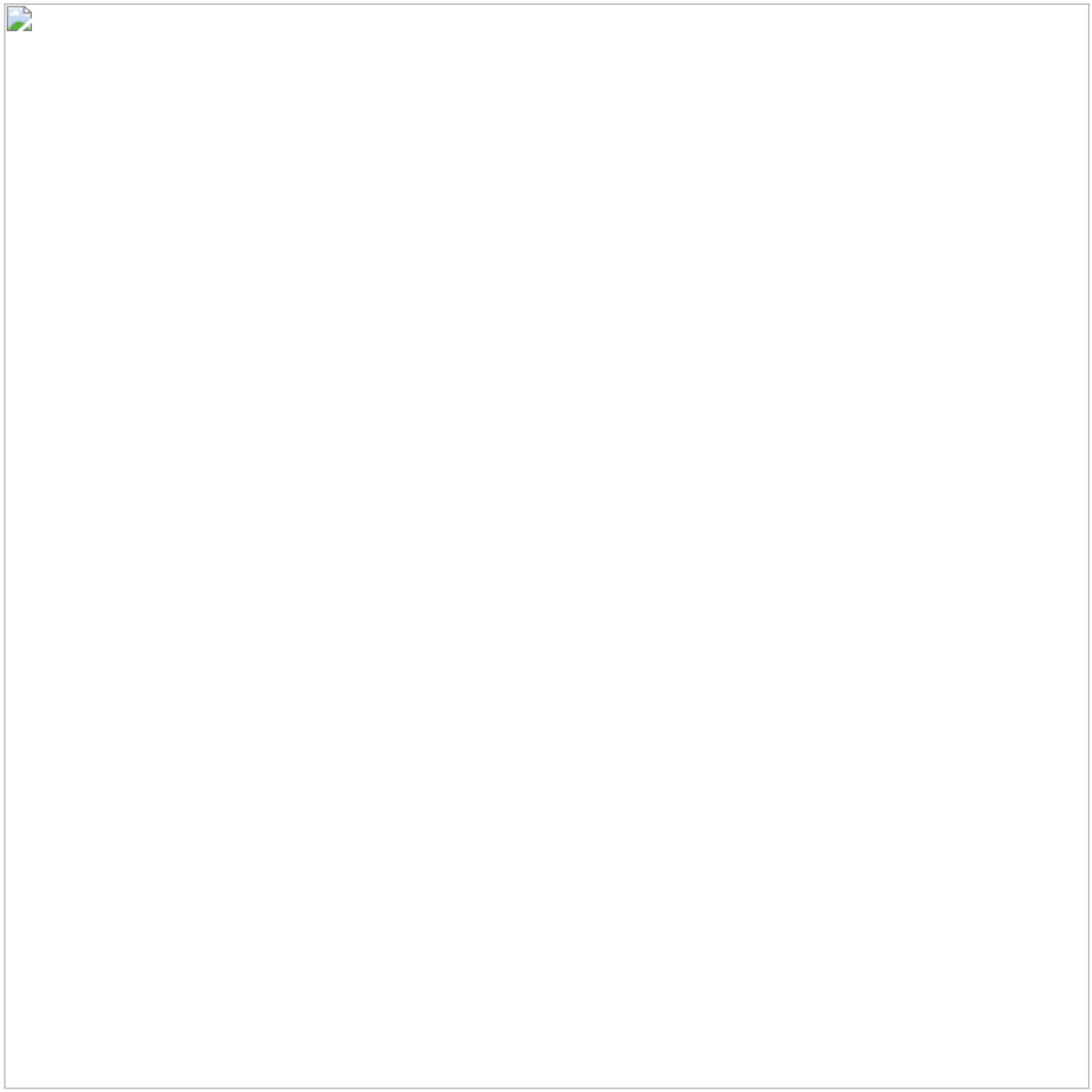
- **Groups**
  - **Contacts** – here all contacts objects should be placed
  - **Distribution Groups** – this is the place for all distribution groups
  - **Security Groups** – here all security groups should go
- **Tier 1 Servers**
  - **Application** – Here, all application servers should be placed.
  - **Collaboration** – Here, all collaboration servers like Sharepoint should be placed. It could contain sub-OU's.
  - **Database** – Here, all database servers like SQL should be placed. It could contain sub-OU's.
  - **Messaging** – this is an OU for Exchange / Lotus servers
  - **Staging** – this is a staging OU
- **User Accounts**
  - **Disabled Users**
  - **Enabled Users**
- **Workstations**
  - **Desktops**
  - **Kiosks**
  - **Laptops**
  - **Staging**

Groups Creation

The first part is done, let's focus on the second part, which is group creation.
We will handle it using another script called **Create-Group**

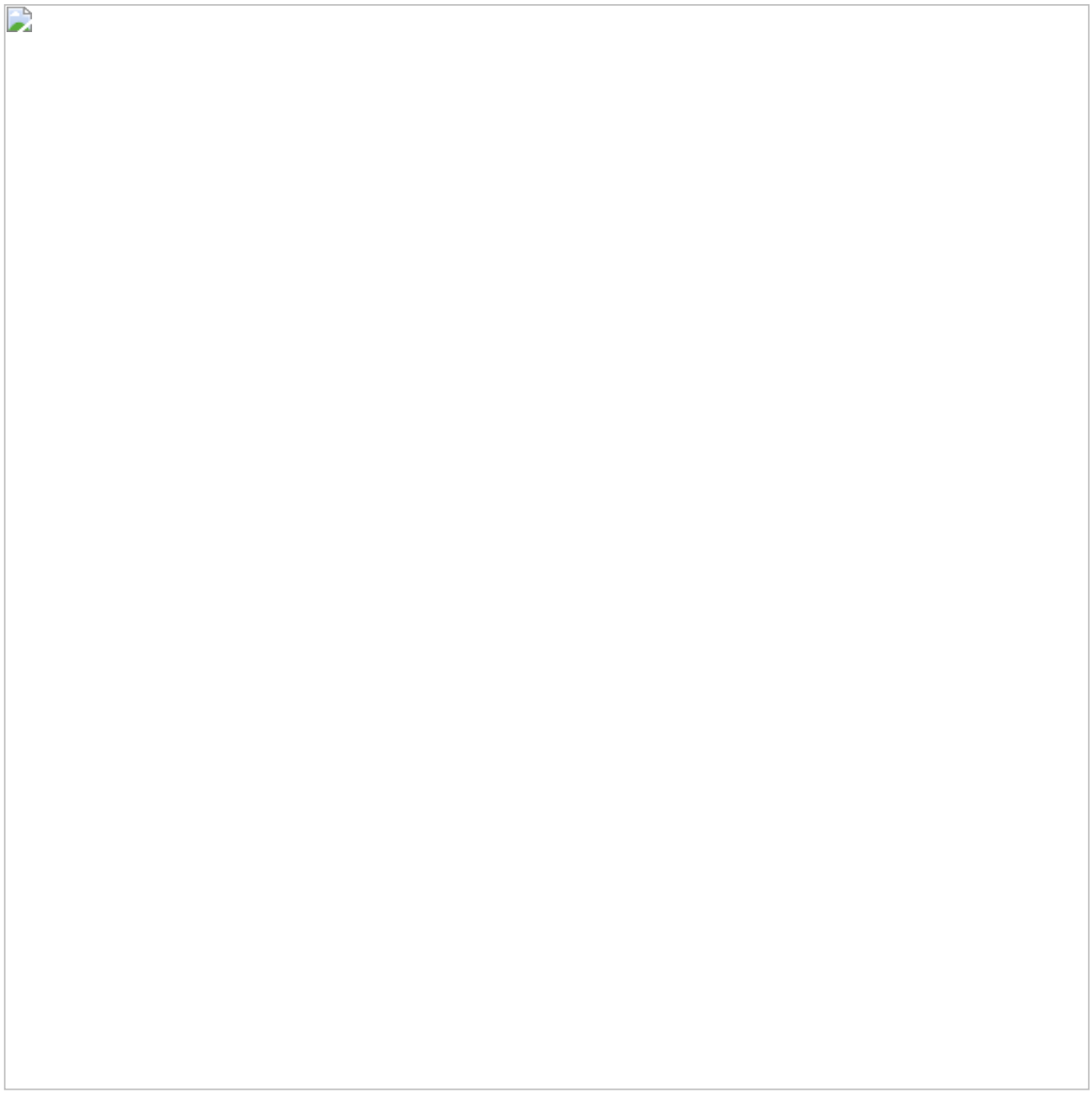
```
$csv = Read-Host -Prompt "Please provide full path to Groups csv file"
.\Create-Group.ps1 -CSVfile $csv -Verbose
```

Output from **Create-Group** script



LAB Tiering Groups Creation

During the code run, you will be asked to provide a path to the CSV file that contains groups.
Please provide the following path to AdminGroups CSV file: C:\Tools\AdminGroups.csv
For standard groups creation, you will have to run again the 2 lines above but with a different path to the file: C:\Tools\AdminGroups.csv



AdminGroups.csv body

Below I will describe every column in CSV file:

- **Name** – Name of the group that you want to create
- **samAccountName** – samAccountName for the group
- **GroupCategory** – This will be a security group
- **GroupScope** – this will be a global group
- **OU** – Distinguished name of the OU where groups should be created
- **Description** – Description of the group
- **Membership** – this value should contain the Group name that should contain the newly created group

Now let's check what this script did and describe it a little bit

- **Tier 0 Replication Maintenance** – members of this group will have permission to perform replication maintenance (e.g., for MIM purposes)

- **Tier 1 Admins** – members of this group will have permission to administer objects under Admin\Tier 1 OU
- **Tier 1 Server Maintenance** – members of this group will have permission to maintenance Tier 1 Servers. This group will be Tier 1 Server Admins, not application admins.
- **Tier 2 Admins**– members of this group will have permission to create and administer under Admin\Tier 1 OU
- **Workstation Maintenance** – this is a Tier 2 level group that will allow members of this group to maintenance all objects under Workstation OU
- **Service Desk Operators** – members of these groups will be able to perform service desk operations like password reset for the users etc.

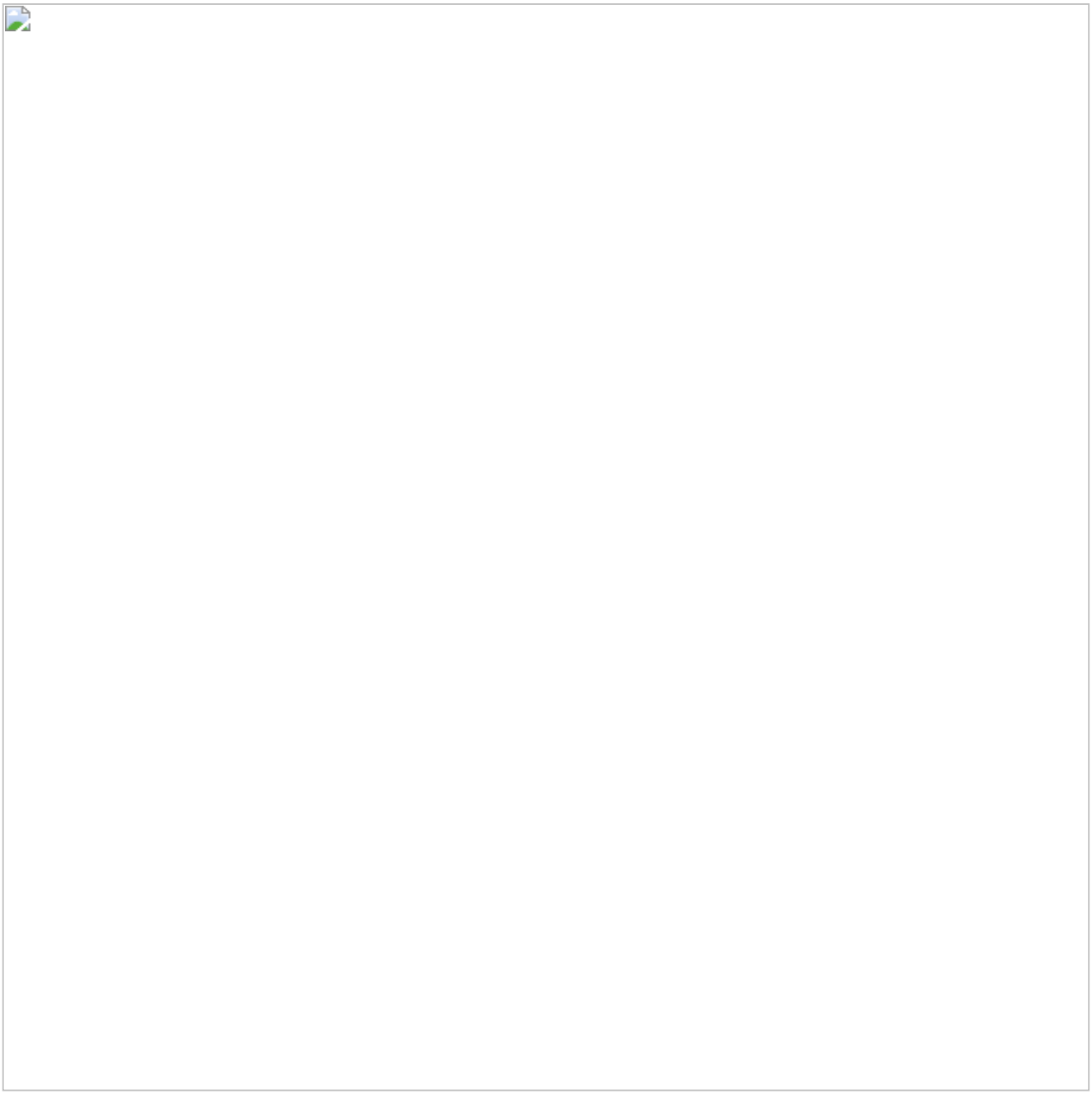
Permission Delegation

So we have OU's created, groups also, let's assign proper permissions
All scripts that we are going to use will have similar logic and variables used during the run. Every time we need to declare an array of PSObjects well known from the **Create-OU** script

Firstly we will run the **Set-OUUserPermissions** script to assign user permissions on OU for a specific group.

```
$List = @(
$(New-Object PSObject -Property @{Group = "Tier2ServiceDeskOperators";
OUPrefix = "OU=User Accounts" }),
$(New-Object PSObject -Property @{Group = "Tier1Admins"; OUPrefix =
"OU=Accounts,ou=Tier1,ou=Admin" }),
$(New-Object PSObject -Property @{Group = "Tier1Admins"; OUPrefix =
"OU=Service Accounts,ou=Tier1,ou=Admin" }),
$(New-Object PSObject -Property @{Group = "Tier2Admins"; OUPrefix =
"OU=Accounts,ou=Tier2,ou=Admin" }),
$(New-Object PSObject -Property @{Group = "Tier2Admins"; OUPrefix =
"OU=Service Accounts,ou=Tier2,ou=Admin" })
)
.\Set-OUUserPermissions.ps1 -list $list -Verbose
```

Output below

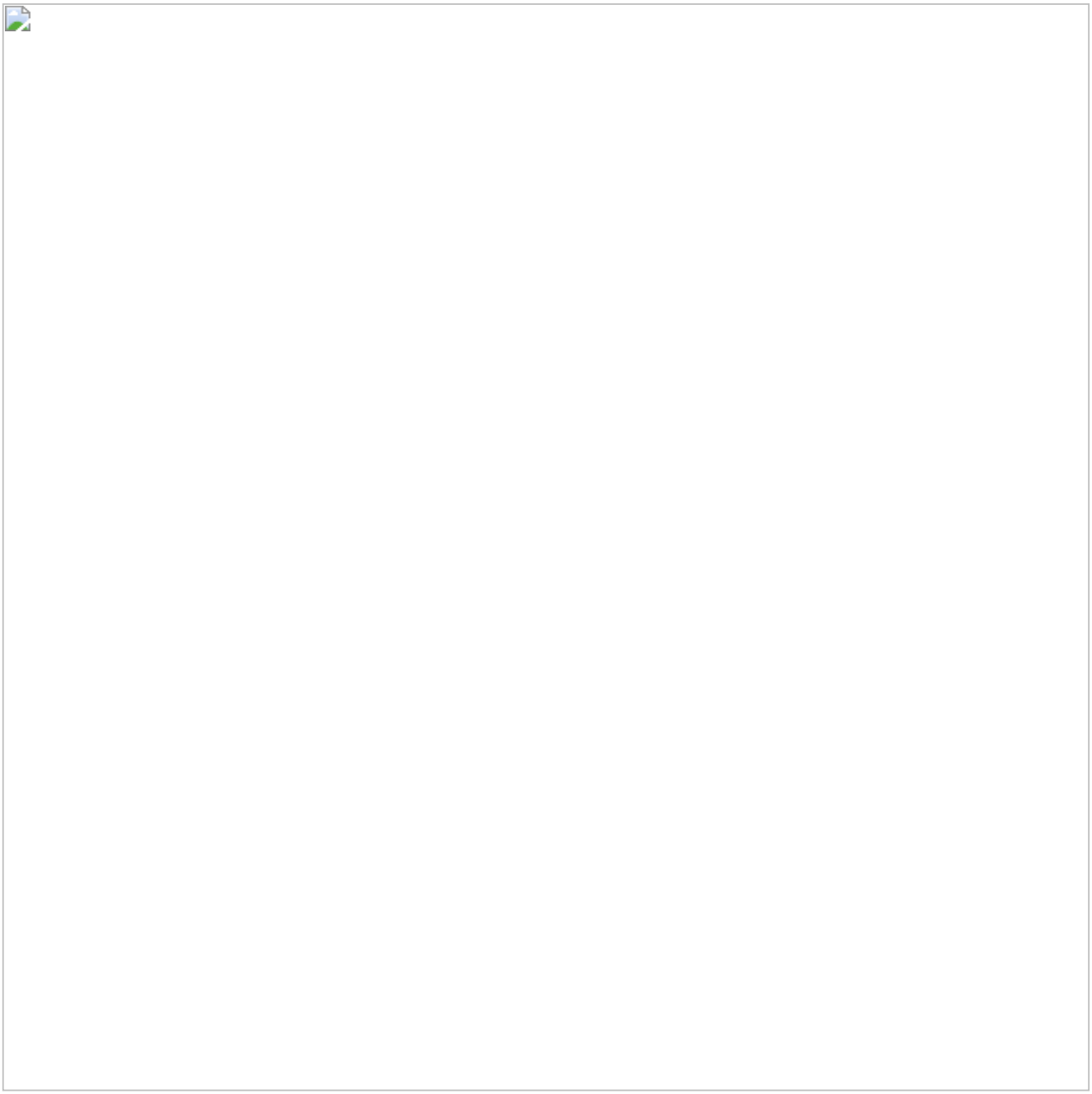


User Permissions Assignment

The next script is **Set-OUWorkstationPermissions,** and it will assign permissions to read computer object properties, including TPM related.

```
$List = @(
$(New-Object PSObject -Property @{Group = "Tier2ServiceDeskOperators";
OUPrefix = "OU=Workstations" }),
$(New-Object PSObject -Property @{Group = "Tier1Admins"; OUPrefix =
"OU=Devices,ou=Tier1,ou=Admin" }),
$(New-Object PSObject -Property @{Group = "Tier2Admins"; OUPrefix =
"OU=Devices,ou=Tier2,ou=Admin" })
)
.\Set-OUWorkstationPermissions.ps1 -list $list -Verbose
```

Output below

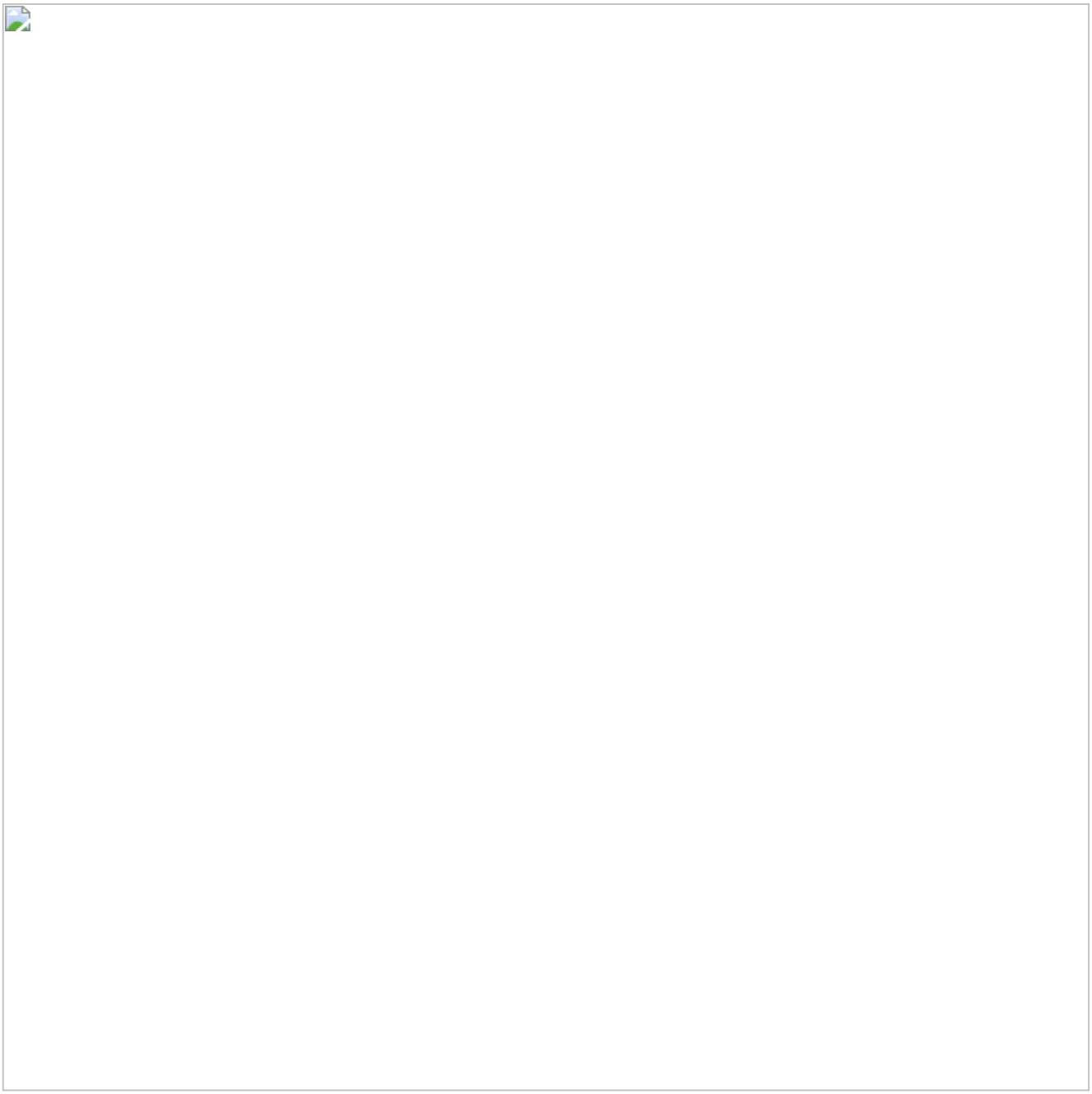


Workstation Permissions Assignment

Moving forward we will need to assign permissions to manage group objects using the **Set-OUGroupPremissions** script.

```
$List = @(
$(New-Object PSObject -Property @{Group = "Tier1Admins"; OUPrefix =
"OU=Groups,ou=Tier1,ou=Admin"}),
$(New-Object PSObject -Property @{Group = "Tier2Admins"; OUPrefix =
"OU=Groups,ou=Tier2,ou=Admin"})
)
.\Set-OUGroupPermissions.ps1 -list $list -Verbose
```

Output below

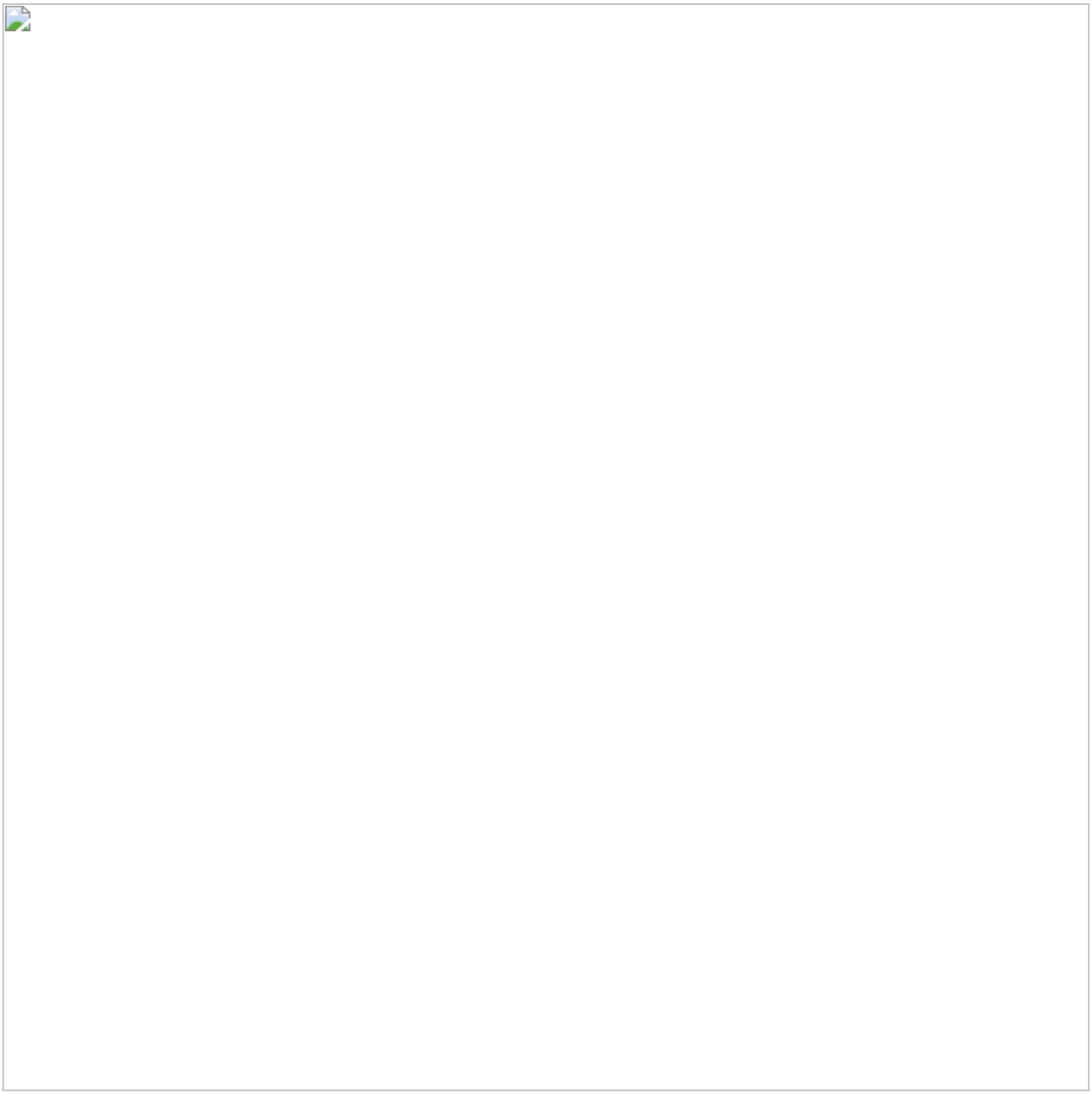


Group Permissions Assignment

Now we are going to assign servicedesk related permissions for computer objects using the **Set-OUComputerPremissions** script.

```
$List = @(
$(New-Object PSObject -Property @{Group = "WorkstationMaintenance"; OUPrefix =
"OU=Quarantine" }),
$(New-Object PSObject -Property @{Group = "WorkstationMaintenance"; OUPrefix =
"OU=Workstations" }),
$(New-Object PSObject -Property @{Group = "Tier1ServerMaintenance"; OUPrefix =
"OU=Tier 1 Servers" })
)
.\Set-OUComputerPermissions.ps1 -list $list -Verbose
```

Output below

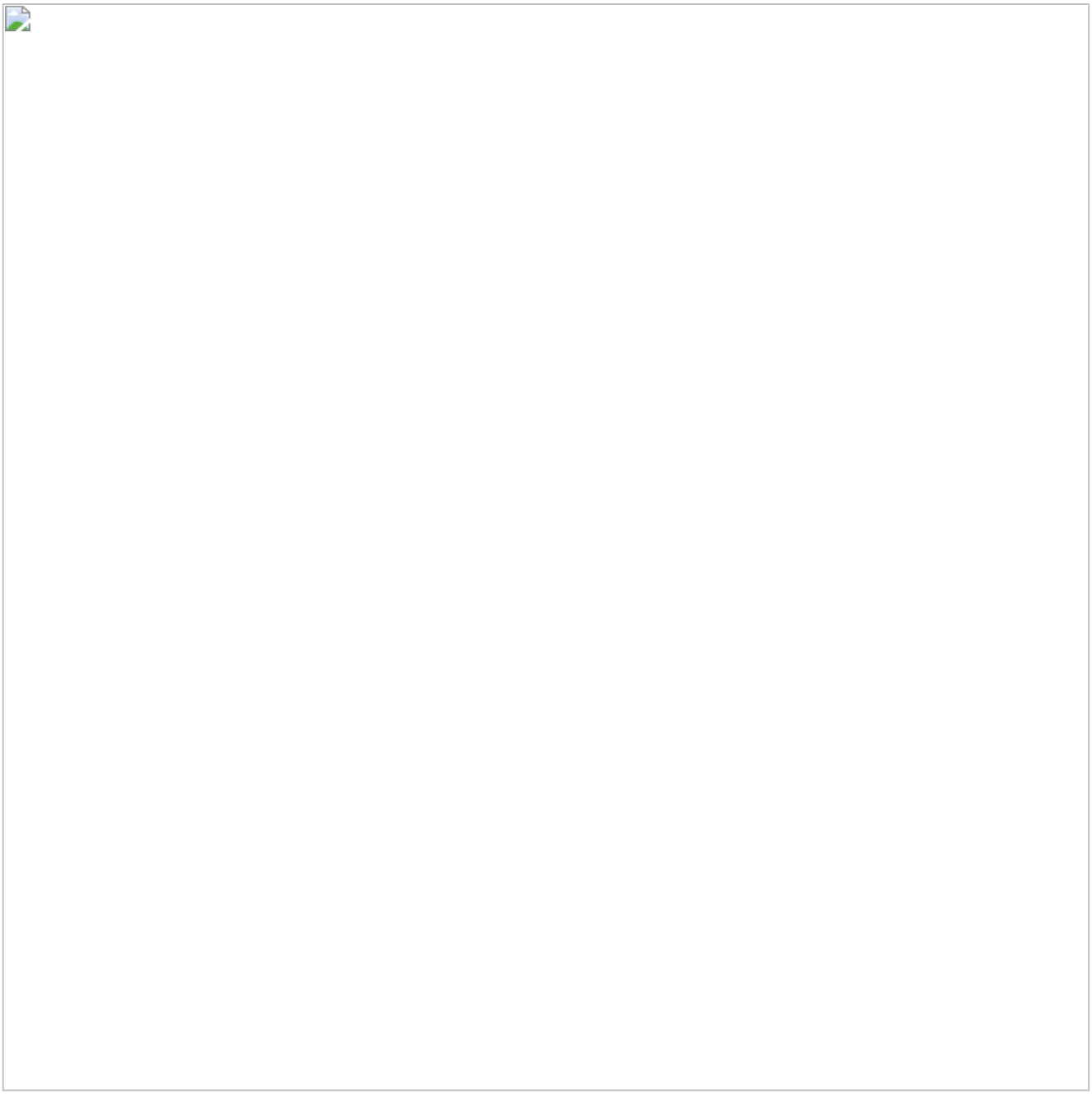


Computer Permissions Assignment

The next step will refer to configuring replication permissions using the **Set-OUReplicationPremissions** script.

```
$List = @(
$(New-Object PSObject -Property @{Group = "Tier0ReplicationMaintenance";
OUPrefix = "" })
)
.\Set-OUReplicationPermissions.ps1 -list $list -Verbose
```

Output below

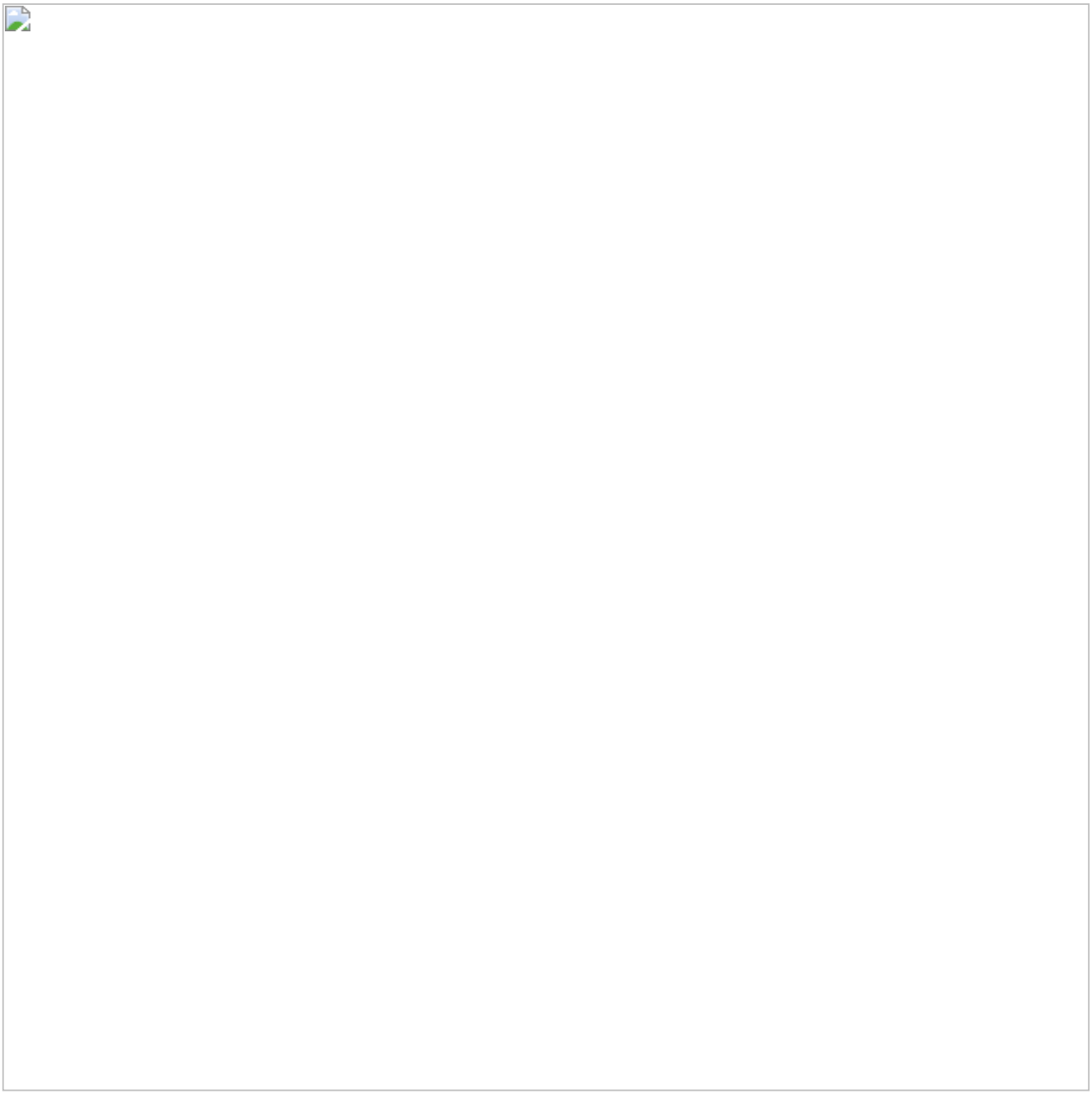


Replication MaintenancePermissions Assignment

The last script that we are going to run will setup proper GPO permissions, for that purpose we will use the **Set-OUGPOPremissions** script.

```
$List = @(
$(New-Object PSObject -Property @{Group = "Tier1ServerMaintenance"; OUPrefix =
"OU=Tier 1 Servers" })
)
.\Set-OUGPOPermissions.ps1 -list $list -Verbose
```

Output below



GPO Permissions Assignment

Going to the end

So we did it !!

We have completely created Tiering OU structure, including group creation and assigning them to the proper OU with proper permissions.

The next article will refer to Privileged Access Workstations deployment. This is a very close topic to Tiering because PAW's are deployed under the following OU's

- Admin\Tier0\Devices
- Admin\Tier1\Devices