The Guide v2.0

poc

Codename: Janice

These days it's a common scenario to see a company acquiring another, and having both organizations hosting their email service in the cloud using Office 365. Unfortunately Microsoft doesn't provide us with easy to use tools - and ways - to simply merge the two company's cloud tenants, which gives other companies room to offer their cloud migration services, using their own proprietary software, which is probably a convenient way for system administrators but surely not ideal for mail flow or for the users. They technically offer their proprietary software to connect to both organizations, create the corresponding new mailboxes in the target tenant, copy the data over and after a final synchronization they remove the source mailbox and finalize the target. Now the first of the two biggest problems with this approach is that we need to purchase extra licenses for the target tenant to accommodate the new mailboxes until the other tenant is demolished and those licenses can be transferred over: it's inconvenient, costs extra and it's hard to rely on Microsoft to make the transfer fast – which can be delayed by many things. The other thing is the fact that we cannot have the same domain name in two tenants at the same time, you can't use the source company's existing email addresses on the target tenant until all of the mailboxes are migrated over and the domain is removed from the source.

Luckily with a little effort there is a way to maintain perfect mail flow during the migration while users are able to use their original email addresses, the same mailbox profile in their Outlook, with no need to purchase extra licenses not even temporarily and with this method we only use the built-in functions of Microsoft EOL and Exchange 2013.

So enough talking, let's jump into it!

This guide simulates a scenario where a company called "Jd0e Inc" (in our case) was acquired by "Agzsolt Inc". They both use a hybrid Office365 environment, in this guide we focus on the email service. Our goal is to merge Jd0e into the Agzsolt tenant while we maintain mail- and workflow the whole time.

To make the situation a little more complex, the source Jd0e tenant is using **ADFS** for single sign-on functionality. Also the Jd0e users are still using the old 2010 version of Outlook so we stick with it here.

In our example **agzsolt.com** is the destination tenant and **jd0e.com** is the organization to be moved the mailboxes from.

The starting scenario

Destination:

agzsolt.com

Hybrid

O365: agzsolt.onmicrosoft.com tenant, agzsolt.com as the default domain

Onprem: **Exchange 2013 CU18** server: **51.143.185.87**

```
root@kali:~# nslookup -q=mx agzsolt.com
Server:         192.168.6.2
Address:        192.168.6.2#53

Non-authoritative answer:
agzsolt.com     mail exchanger = 10 mail.agzsolt.com.

Authoritative answers can be found from:
mail.agzsolt.com        internet address = 51.143.185.87
```

Source:

Jd0e.com

Hybrid and ADFS federated

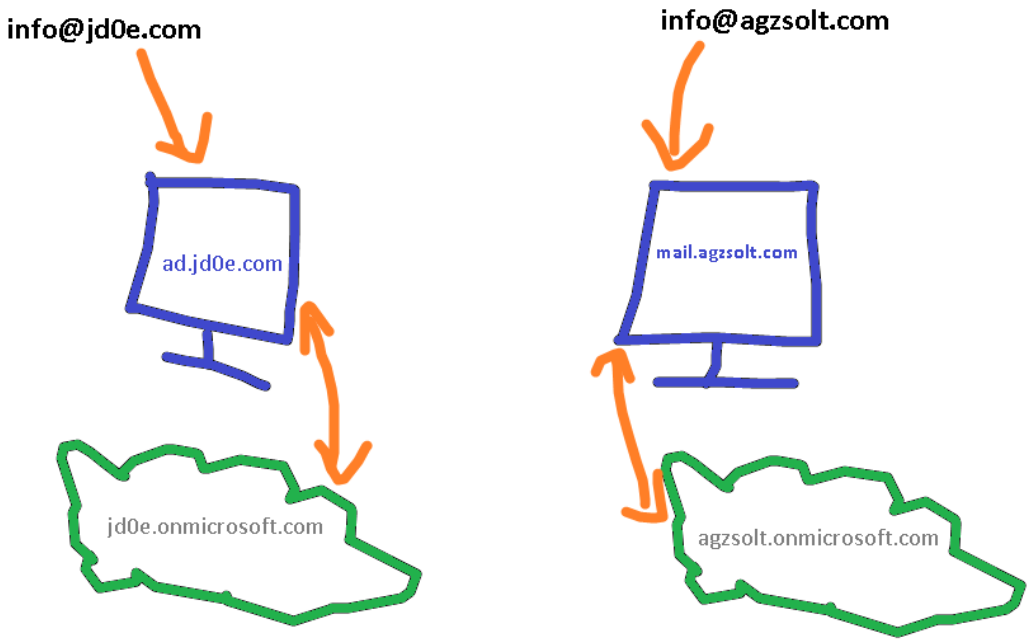
O365: jdoe.onmicrosoft.com tenant; jd0e.com as the default domain

Onprem: **Exchange 2013 CU18** server+DC: **51.143.157.86**

Adfs proxy: 51.143.188.208

```
root@kali:~# nslookup -q=mx jd0e.com
Server:         192.168.6.2
Address:        192.168.6.2#53

Non-authoritative answer:
jd0e.com        mail exchanger = 10 dc.jd0e.com.

Authoritative answers can be found from:
dc.jd0e.com     internet address = 51.143.157.86
```

As expected, the incoming emails are directed to the corresponding on-prem mail server, where they are forwarded into the cloud if the target mailbox is not found locally with the help of to the hybrid setup. To visualize the mail flow now:



Mailbox situation:

Destination – agzsolt.com:

mailboxes groups resources contacts shared migration

DISPLAY NAME	MAILBOX TYPE	EMAIL ADDRESS
Person 1	**Office 365**	**person1@agzsolt.com**
Person 10	Office 365	person10@agzsolt.com
Person 2	Office 365	person2@agzsolt.com
Person 3	Office 365	person3@agzsolt.com
Person 4	Office 365	person4@agzsolt.com
Person 5	Office 365	person5@agzsolt.com
Person 6	Office 365	person6@agzsolt.com
Person 7	Office 365	person7@agzsolt.com
Person 8	Office 365	person8@agzsolt.com
Person 9	Office 365	person9@agzsolt.com
za	User	za@agzsolt.com

Source – jd0e.com:

mailboxes groups resources contacts shared migration

DISPLAY NAME	MAILBOX TYPE	EMAIL ADDRESS
Chandler Bing	Office 365	Chandler@jd0e.com
Janice Hosenstein	Office 365	Janice@jd0e.com
Joey Tribbiani	Office 365	Joey@jd0e.com
Monica Geller	Office 365	Monica@jd0e.com
Phoebe Buffay	Office 365	Phoebe@jd0e.com
Rachel Green	Office 365	Rachel@jd0e.com
Ross Geller	Office 365	Ross@jd0e.com
za	User	za@jd0e.com

Ok, let's start!

Prepare the destination on-prem server

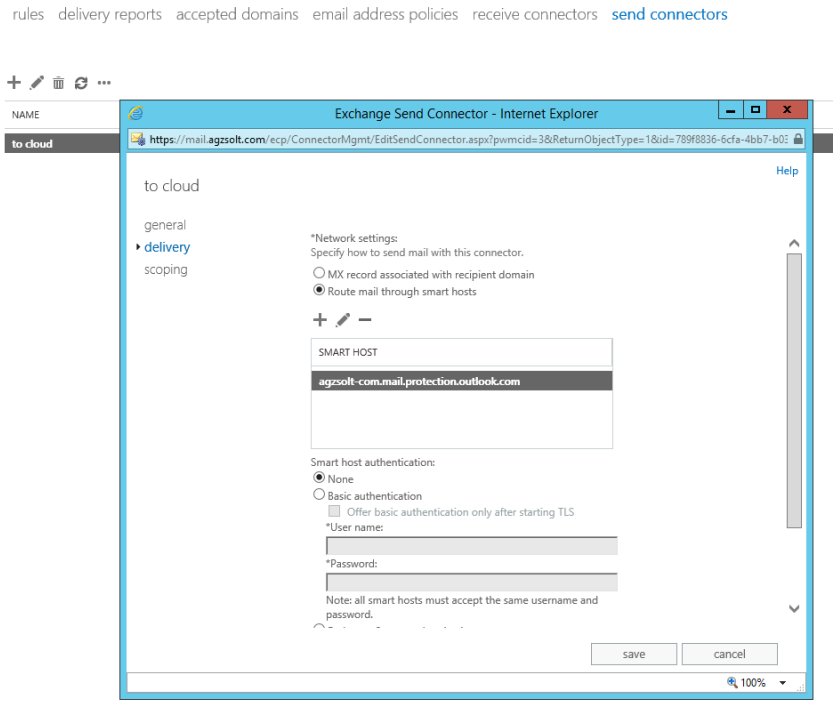
1. Put jd0e.com in the **accepted domain** list

recipients
permissions
compliance management
organization
protection
mail flow

rules delivery reports accepted domains email address policies re

NAME	ACCEPTED DOMAIN	DOMAIN TYPE
agzsolt.com (default domain)	**agzsolt.com**	**Internal relay**
agzsolt.mail.onmicrosoft.com	agzsolt.mail.onmicrosoft.com	Internal relay
jd0e.com	jd0e.com	Internal relay

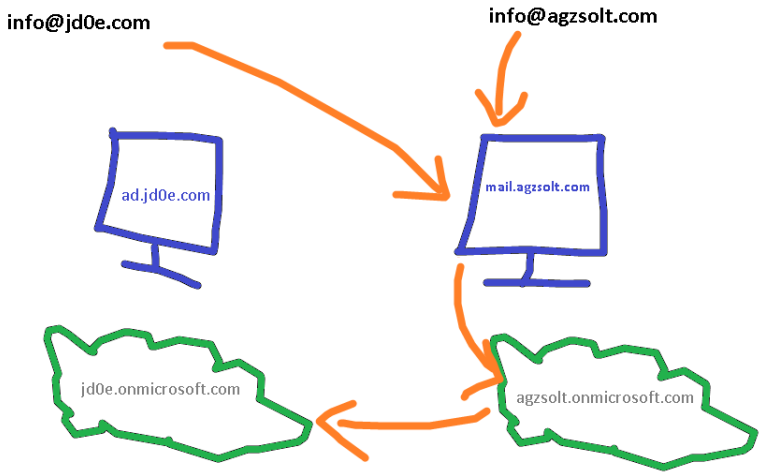
2. If not already done, create a **send connector to the cloud**



Change the jd0e.com MX record to point to the agzsolt.com on-prem server

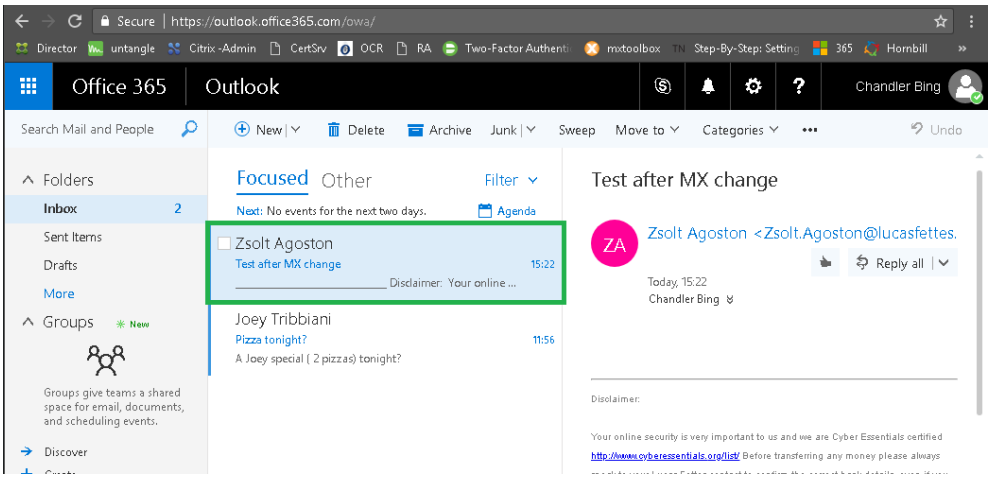
```
root@kali:~# nslookup -q=mx jd0e.com 8.8.8.8
Server:         8.8.8.8
Address:        8.8.8.8#53

Non-authoritative answer:
jd0e.com        mail exchanger = 10 mail.agzsolt.com.

Authoritative answers can be found from:

root@kali:~#
```

Because of this change the incoming mails to @jd0e.com will be directed to the mail.agzsolt.com server where the migrated mailboxes will sit or if no local mailbox is found by the server it forwards them to the cloud where the cloud servers will find the route to the correct tenant.



Now test the mail flow to verify the emails are still arriving to the @jd0e.com mailboxes: we send an email from a Gmail address to a jd0e mailbox. Note that the jd0e MX record has already been directed to the destination on-prem server (mail.agzsolt.com). Also the SPF record is updated accordingly to prevent the sent emails to be put in the recipient's junk folder.

```
Non-authoritative answer:
jd0e.com        text = "v=spf1 include:mail.agzsolt.com include:spf.protection.outlook.com -all"
```

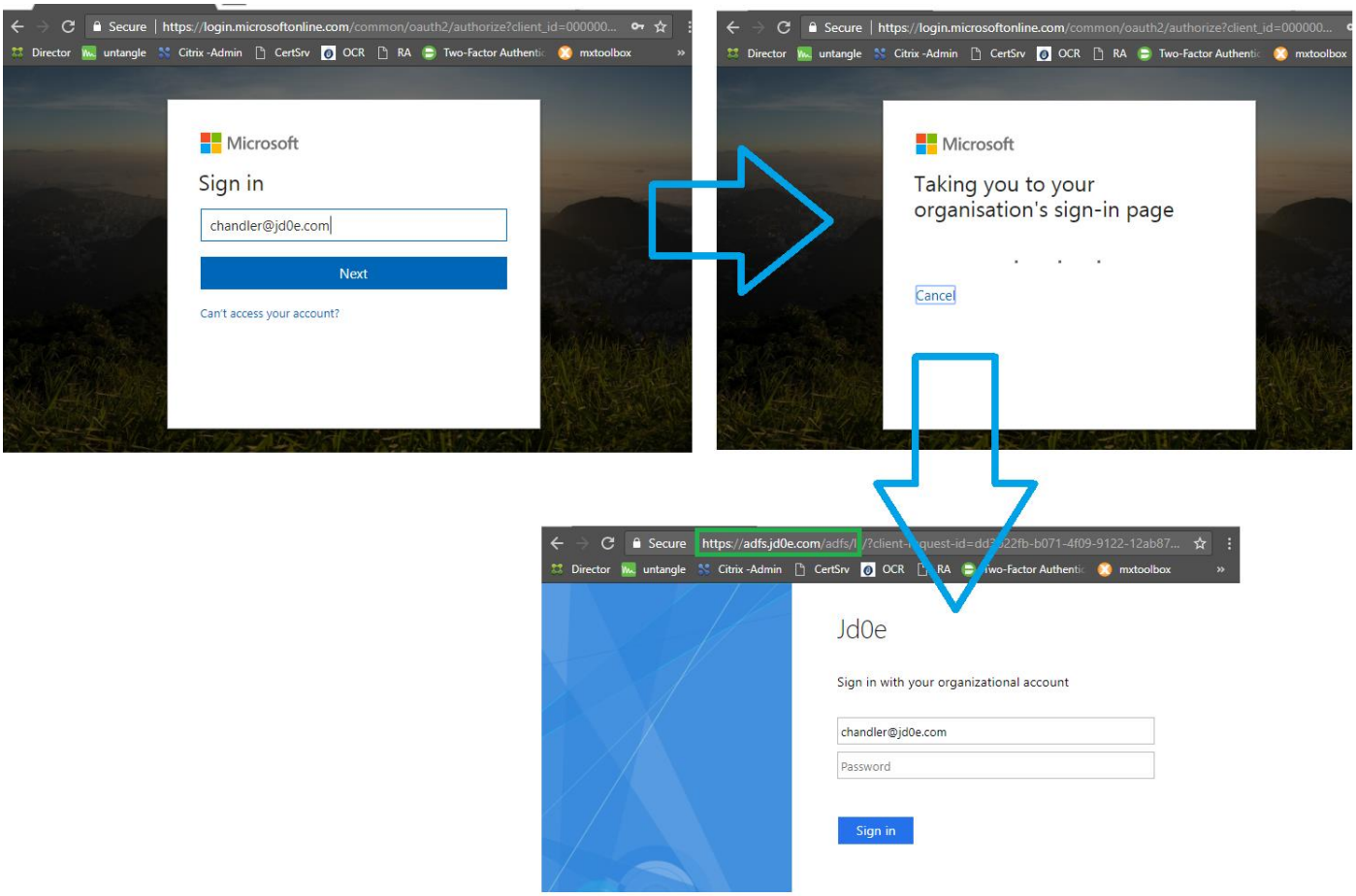


The email has arrived without an issue!

Disable Federation

First we need to disable ADFS before we can proceed and cut dirsync to make the mailboxes fully cloud-managed.

Let's see how the OWA portal behaves. As soon as we try to log in on https://out.look.office365.com it redirects us to the local ADFS proxy server for authentication



We can see on dc.jd0e.com server that the domain is federated:

```
[PS] C:\>Get-MsolDomain | fl name,authentication

Name           : jd0e.com
Authentication : Federated

Name           : jd0e.onmicrosoft.com
Authentication : Managed

Name           : jd0e.mail.onmicrosoft.com
Authentication : Managed
```

Now we **make the domain standalone**. First we connect to the ADFS server

```
Set-MsolADFSContext -Computer adfs.jd0e.com
```

And set the domain to standard:

```
Convert-MsolDomainToStandard -DomainName jd0e.com -SkipUserConversion:$true -PasswordFile c:\passwdfile.txt
```

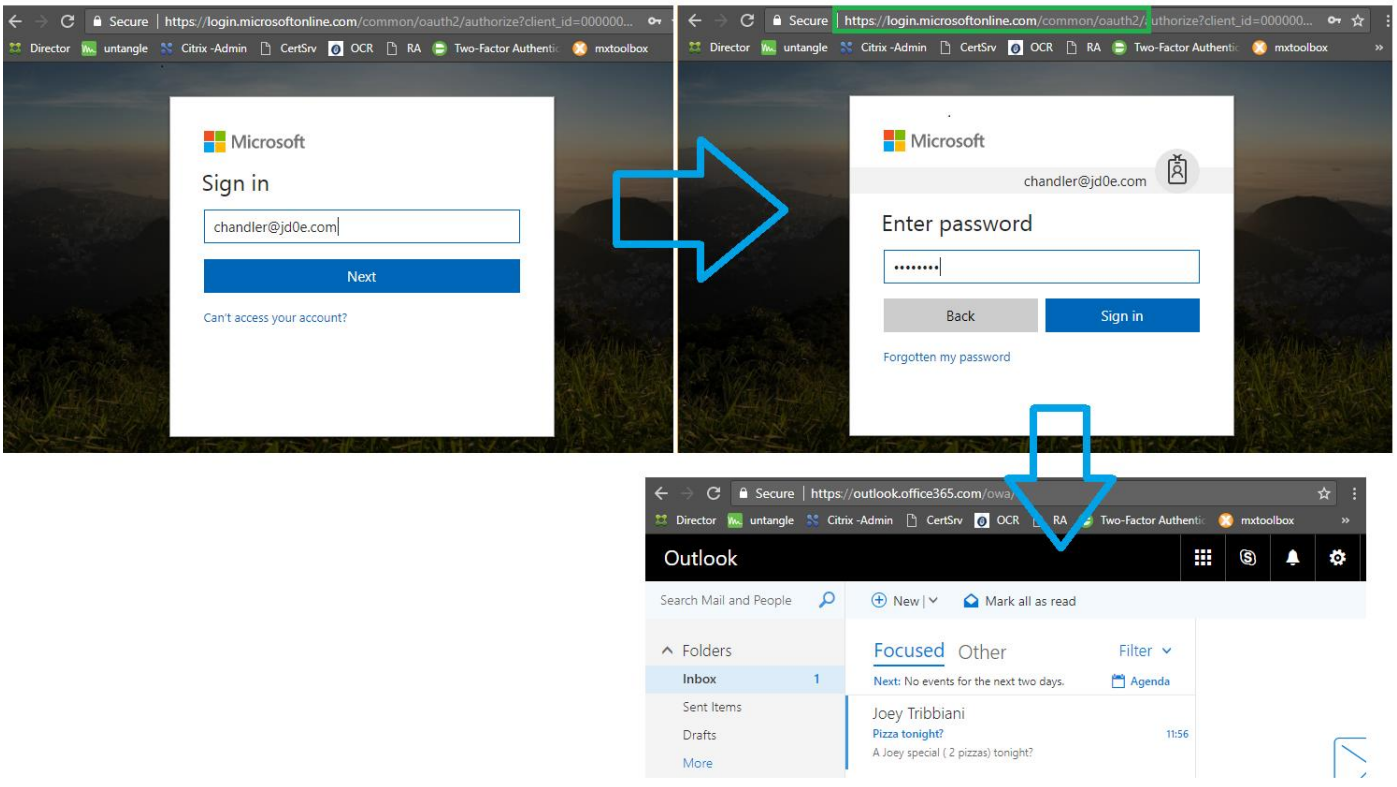
Setting the authentication method as well:

```
Set-MsolDomainAuthentication -Authentication managed -DomainName jd0e.com
```

At this point the domain becomes cloud-managed again, that we will confirm

```
[PS] C:\>Convert-MsolDomainToStandard -DomainName jd0e.com -SkipUserConversion:$true -PasswordFile c:\passwdfile.txt
Successfully updated 'jd0e.com' domain.
[PS] C:\>
[PS] C:\>Get-MsolDomain | fl name,authentication

Name           : jd0e.com
Authentication : Managed

Name           : jd0e.onmicrosoft.com
Authentication : Managed

Name           : jd0e.mail.onmicrosoft.com
Authentication : Managed
```

To see if the authentication works we check the login interface again:

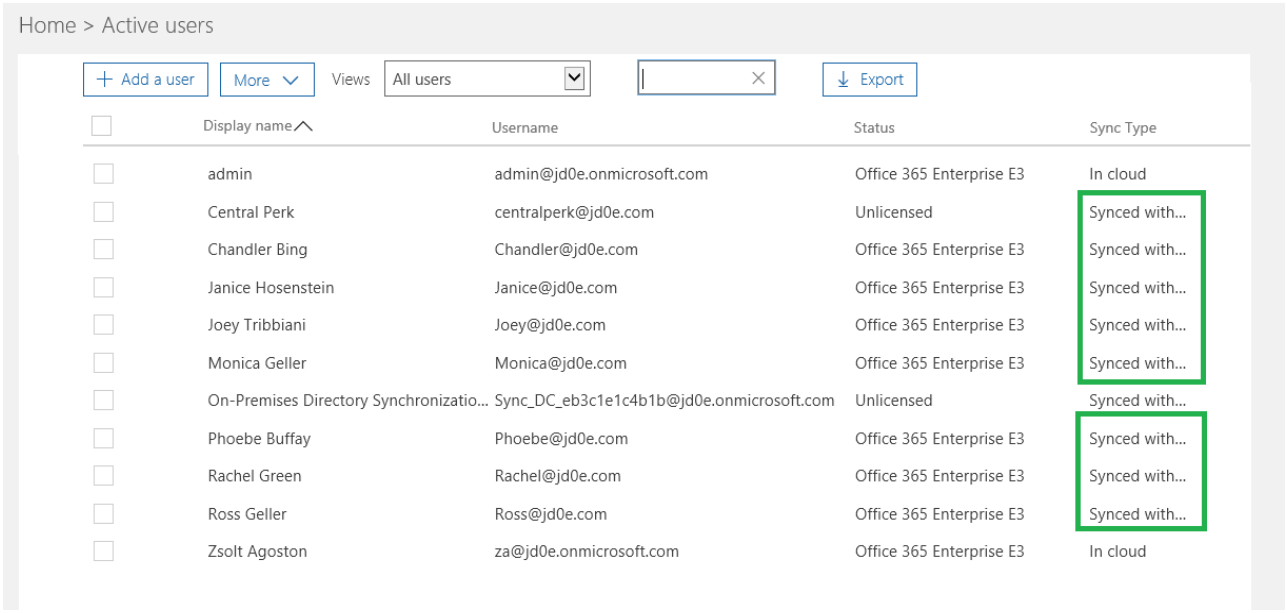


We are in, the authentication is happening in the cloud with the AD password that is synced with DirSync! Excellent!

Cut jd0e.com dirsync (can take up to 72 hours to complete), or…

In this step we convert accounts to purely cloud account while keeping their original password

Before:



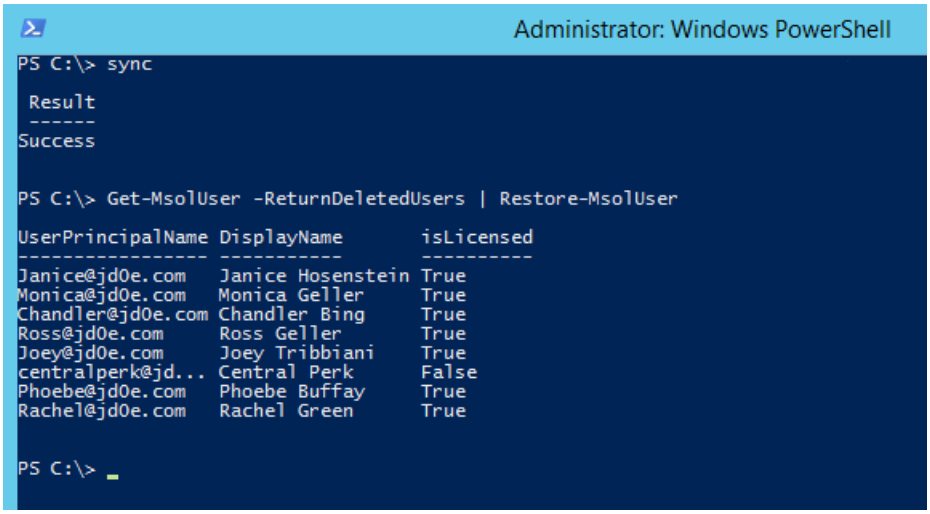

```
Set-MsolDirSyncEnabled -EnableDirSync:$false
```

Check if the process has run:

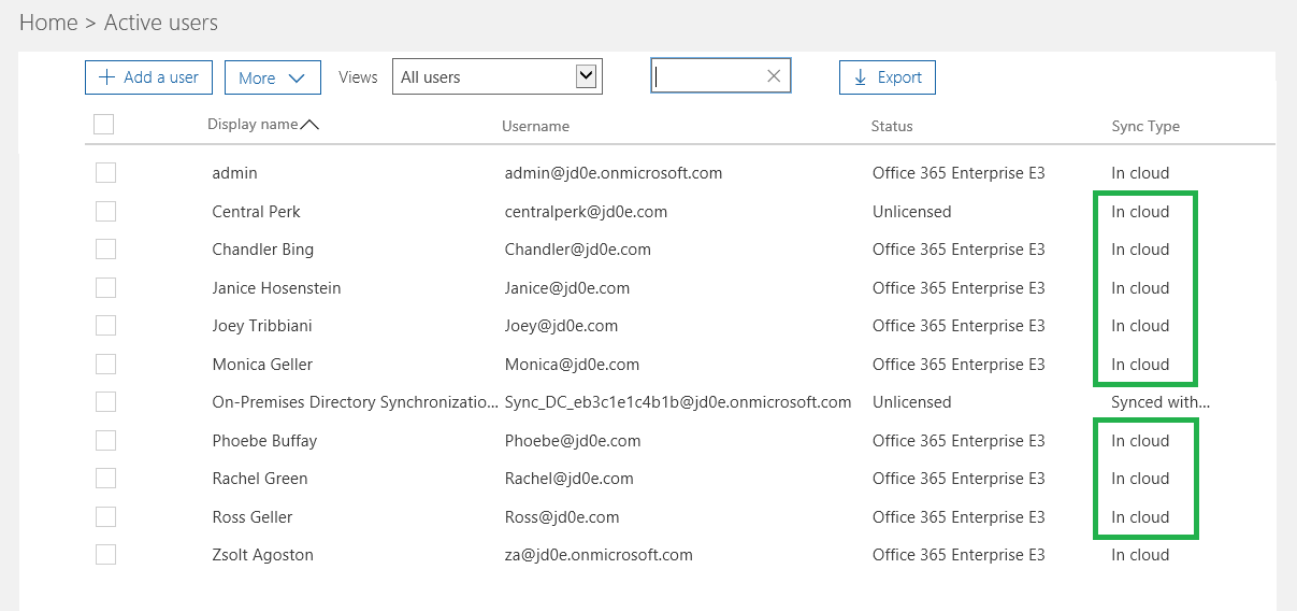
```
(Get-MSOLCompanyInformation).DirectorySynchronizationEnabled
```

Note, if the process takes very long there's another way: **simply move the user accounts to an OU that is not synced to the cloud, and wait for or force a sync cycle.** That will soft delete the cloud accounts, after which they can be restored using the following command (the cloud system will restore them as cloud accounts, preserving the original passwords, permission settings as well):

```
Get-MsolUser -ReturnDeletedUsers | Restore-MsolUser
```



After which:

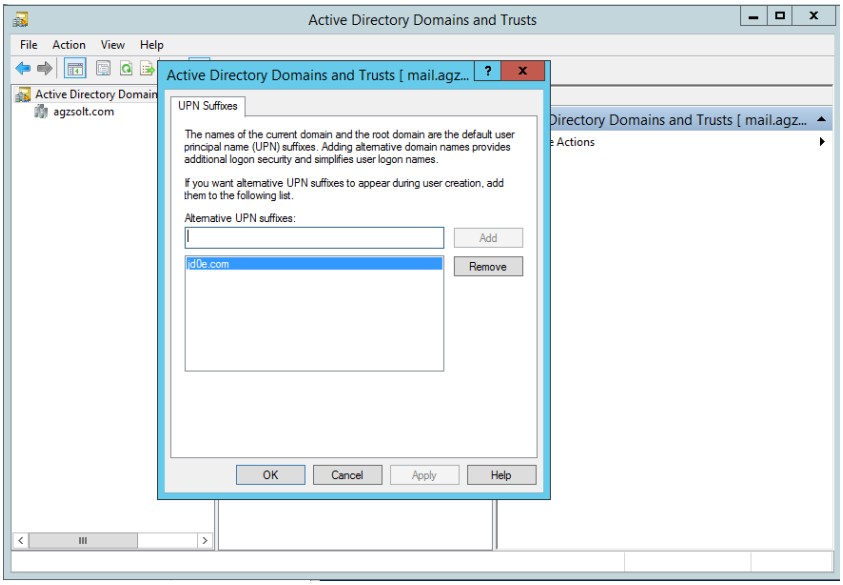


Now we check the permissions on the shared mailboxes to make sure they are not lost just like after a license unassign-reassign scenario. As seen below the permission structure is preserved post-cloudization ☺

```
PS C:\> Get-MailboxPermission -Identity centralperk@jd0e.com

Identity             User                 AccessRights                                        IsInherited Deny
--------             ----                 ------------                                        ----------- ----
Central Perk         NT AUTHORITY\SELF    {FullAccess, ReadPermission}                        False       False
Central Perk         NT AUTHORITY\SELF    {FullAccess, ExternalAccount, ReadPermission}       False       False
Central Perk         Ross@jd0e.com        {FullAccess}                                        False       False
Central Perk         Rachel@jd0e.com      {FullAccess}                                        False       False
Central Perk         Joey@jd0e.com        {FullAccess}                                        False       False
Central Perk         Janice@jd0e.com      {FullAccess}                                        False       False
Central Perk         Phoebe@jd0e.com      {FullAccess}                                        False       False
Central Perk         Monica@jd0e.com      {FullAccess}                                        False       False
Central Perk         Chandler@jd0e.com    {FullAccess}                                        False       False
Central Perk         GBRP265\Administr... {FullAccess}                                        True        True
Central Perk         GBRP265\Domain Ad... {FullAccess}                                        True        True
Central Perk         GBRP265\Enterpris... {FullAccess}                                        True        True
Central Perk         GBRP265\Organizat... {FullAccess}                                        True        True
Central Perk         NT AUTHORITY\SYSTEM  {FullAccess}                                        True        False
Central Perk         NT AUTHORITY\NETW... {ReadPermission}                                    True        False
Central Perk         PRDTSB01\JitUsers    {ReadPermission}                                    True        False
Central Perk         GBRP265\Administr... {FullAccess, DeleteItem, ReadPermission, ChangePermissio... True        False
Central Perk         GBRP265\Domain Ad... {FullAccess, DeleteItem, ReadPermission, ChangePermissio... True        False
Central Perk         GBRP265\Enterpris... {FullAccess, DeleteItem, ReadPermission, ChangePermissio... True        False
Central Perk         GBRP265\Organizat... {FullAccess, DeleteItem, ReadPermission, ChangePermissio... True        False
Central Perk         GBRP265\Public Fo... {ReadPermission}                                    True        False
Central Perk         GBRP265\Exchange ... {FullAccess, ReadPermission}                        True        False
Central Perk         GBRP265\Exchange ... {FullAccess, DeleteItem, ReadPermission, ChangePermissio... True        False
Central Perk         GBRP265\Managed A... {ReadPermission}                                    True        False


PS C:\> _
```

Create jd0e.com users in the agzsolt.com local AD in a non-synced OU

First, we add the jd0e.com domain using the **Active Directory Domains and Trusts** applet temporarily to make the transition simpler for the users. This way they will be able to log in with the help of the underlying kerberos ticketing system – meaning no password prompts (at least while the mailboxes are sitting on the on-prem server) ☺



We run the following script to create the users, which will be created from **users.csv**

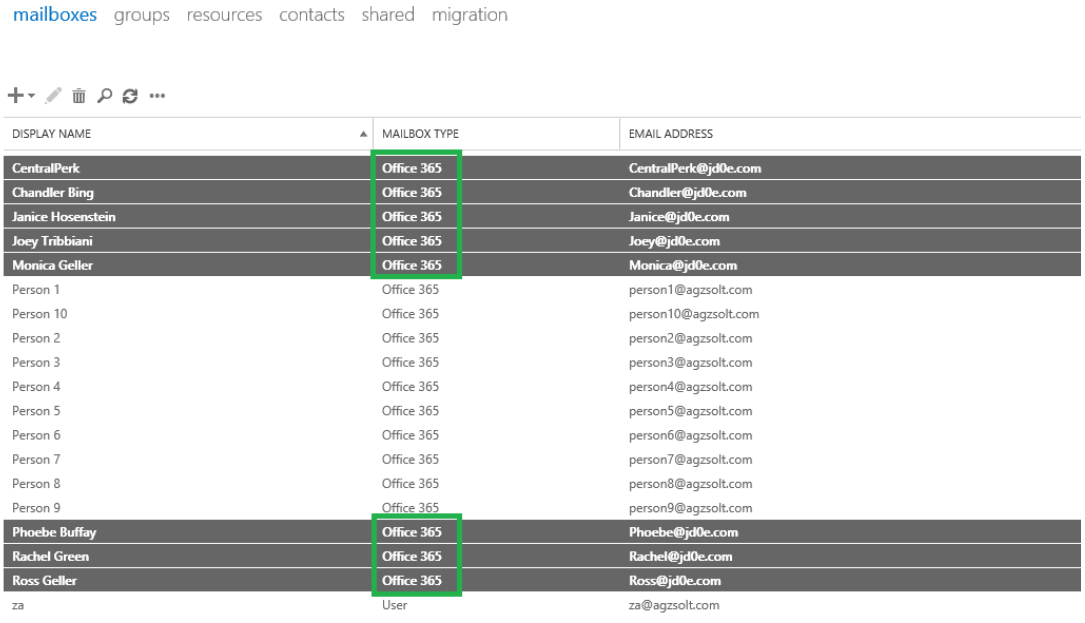
Users.csv

FirstName	LastName
Ross	Geller
Joey	Tribbiani
Monica	Geller
Rachel	Green
Chandler	Bing
Phoebe	Buffay
Janice	Hosenstein
CentralPerk	

Script:

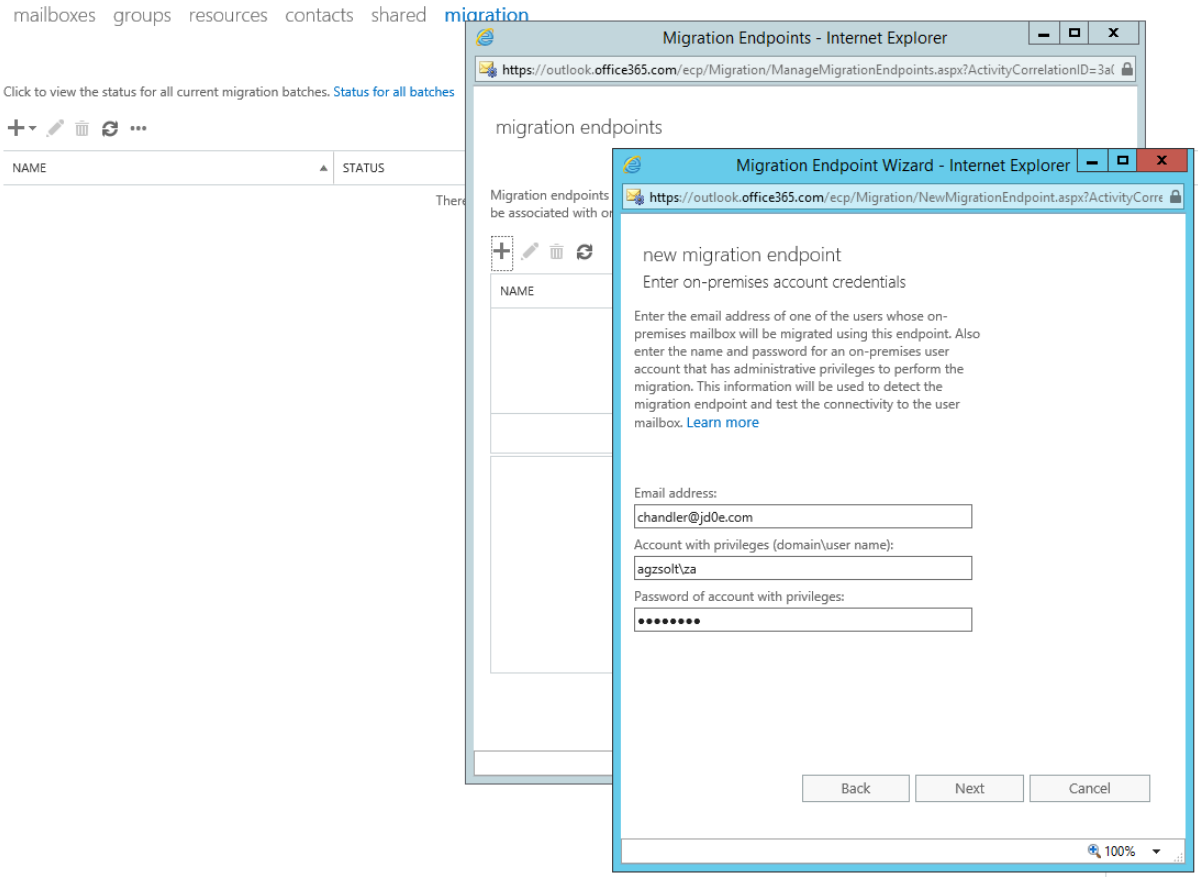
```
New-ADOrganizationalUnit -Name "jd0e" -Path "OU=My Business,DC=agzsolt,DC=com" –Verbose

import-csv users.csv | foreach {
$fn=$_.FirstName
$ln=$_.LastName
New-ADUser -Name "$fn $ln" -DisplayName "$fn $ln" -GivenName "$fn" -Surname "$ln" -UserPrincipalName $fn@jd0e.com -
Path "OU=jd0e,OU=My Business,DC=agzsolt,DC=com" -Enabled:$true -EmailAddress "$fn@jd0e.com" -
AccountPassword(ConvertTo-SecureString "Password12345!" -AsPlainText -Force)
Enable-RemoteMailbox -Identity $fn@jd0e.com -RemoteRoutingAddress $fn@jd0e.onmicrosoft.com
Set-RemoteMailbox -Identity $fn@jd0e.com -EmailAddressPolicyEnabled:$false
}
```

After the commands being run we check the results on the agzsolt.com server:

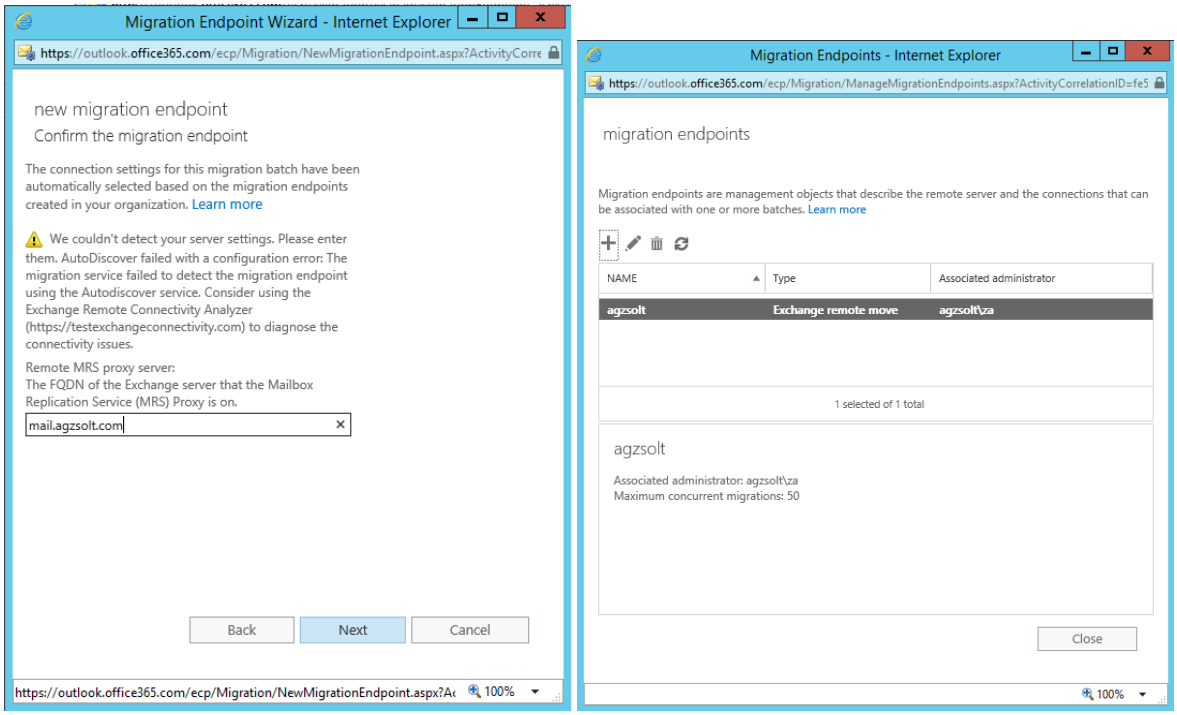


Configure the cross-forest hybrid environment

To make the servers able to move jd0e.com mailboxes to the agzsolt.com server we need to create a migration endpoint in the jd0e.com **cloud** server. It is done in **recipients/migration/migration endpoints**, as the new endpoint's type we use is **"exchange remote"**



It will fail because the server tries to determine the destination FQDN using autodiscover which points to the wrong location of course. We put the right server manually:



In our example we will call the connector "**agzsolt**"

Sync the MSOL attributes into the agzsolt.com local AD accounts

Here the most important thing is that the **ExchangeGUID** attribute of the accounts on the destination on-prem server must match the **ExchangeGUID, WindowsEmailAddress and PrimarySMTPAddress** attributes of the actual cloud mailboxes. We sync them doing the following:

1. On the cloud server we run the following command that will create a file called **mailboxes.csv**, with all the mailboxes and with the attributes we need:

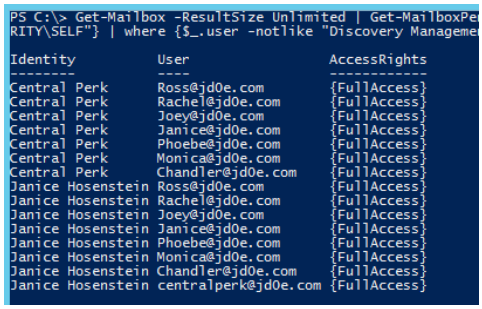
```
Get-Mailbox -ResultSize Unlimited | select userprincipalname,windowsemailaddress,alias,exchangeGUID |
Export-Csv mailboxes.csv
```

   Now we import the important attributes into the on-prem server for compliance:

```
import-csv mailboxes.csv | foreach {
$name=$_.userprincipalname
$winname=$_.windowsemailaddress
$alias=$_.alias
$guid=$_.exchangeGUID
Set-RemoteMailbox -Identity $name -ExchangeGuid $guid -WindowsEmailAddress $winname -EmailAddresses
@{add="$name"}
Set-RemoteMailbox -Identity $name -PrimarySmtpAddress $name
write-host "$name has given GUID: $guid"
}
```

2. To save the mailbox permission structure, we create a backup file that stores that information, called **perm.csv**

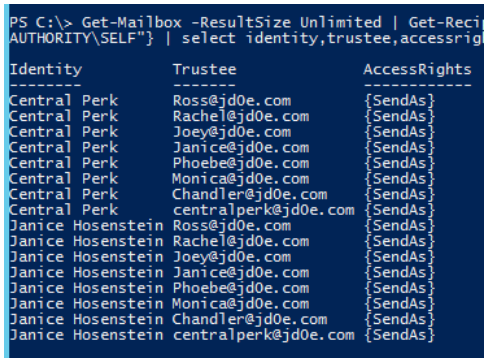
```
Get-Mailbox -ResultSize Unlimited | Get-MailboxPermission | where {$_.isinherited -like "FALSE"} | where
{$_.user -notlike "NT AUTHORITY\SELF"} | where {$_.user -notlike "Discovery Management"} | select
identity,user,accessrights | Export-Csv perm.csv
```



```
PS C:\> Get-Mailbox -ResultSize Unlimited | Get-MailboxPe
RITY\SELF"} | where {$_.user -notlike "Discovery Manageme

Identity          User                AccessRights
--------          ----                ------------
Central Perk      Ross@jd0e.com       {FullAccess}
Central Perk      Rachel@jd0e.com     {FullAccess}
Central Perk      Joey@jd0e.com       {FullAccess}
Central Perk      Janice@jd0e.com     {FullAccess}
Central Perk      Phoebe@jd0e.com     {FullAccess}
Central Perk      Monica@jd0e.com     {FullAccess}
Central Perk      Chandler@jd0e.com   {FullAccess}
Janice Hosenstein Ross@jd0e.com       {FullAccess}
Janice Hosenstein Rachel@jd0e.com     {FullAccess}
Janice Hosenstein Joey@jd0e.com       {FullAccess}
Janice Hosenstein Janice@jd0e.com     {FullAccess}
Janice Hosenstein Phoebe@jd0e.com     {FullAccess}
Janice Hosenstein Monica@jd0e.com     {FullAccess}
Janice Hosenstein Chandler@jd0e.com   {FullAccess}
Janice Hosenstein centralperk@jd0e.com {FullAccess}
```

   We do the same with send-as permissions (**sendasperm.csv**):

```
Get-Mailbox -ResultSize Unlimited | Get-RecipientPermission | where {$_.isinherited -like "FALSE"} | where
{$_.trustee -notlike "NT AUTHORITY\SELF"} | select identity,trustee,accessrights | Export-Csv sendasperm.csv
```



```
PS C:\> Get-Mailbox -ResultSize Unlimited | Get-Reci
AUTHORITY\SELF"} | select identity,trustee,accessrig

Identity          Trustee             AccessRights
--------          -------             ------------
Central Perk      Ross@jd0e.com       {SendAs}
Central Perk      Rachel@jd0e.com     {SendAs}
Central Perk      Joey@jd0e.com       {SendAs}
Central Perk      Janice@jd0e.com     {SendAs}
Central Perk      Phoebe@jd0e.com     {SendAs}
Central Perk      Monica@jd0e.com     {SendAs}
Central Perk      Chandler@jd0e.com   {SendAs}
Central Perk      centralperk@jd0e.com {SendAs}
Janice Hosenstein Ross@jd0e.com       {SendAs}
Janice Hosenstein Rachel@jd0e.com     {SendAs}
Janice Hosenstein Joey@jd0e.com       {SendAs}
Janice Hosenstein Janice@jd0e.com     {SendAs}
Janice Hosenstein Phoebe@jd0e.com     {SendAs}
Janice Hosenstein Monica@jd0e.com     {SendAs}
Janice Hosenstein Chandler@jd0e.com   {SendAs}
Janice Hosenstein centralperk@jd0e.com {SendAs}
```

3. Next, we save the distribution groups and members as well in a file called **distro.csv**

```
$distro = Get-DistributionGroup -ResultSize unlimited
$distro | select samaccountname,displayname,windowsemailaddress | Export-Csv distro-list.csv
$members = foreach ($m in $distro) { Get-DistributionGroupMember -Identity $m.Identity | Select
@{Name="Group";Expression={$m.name}},PrimarySMTPAddress}
$members | export-csv distro.csv
```

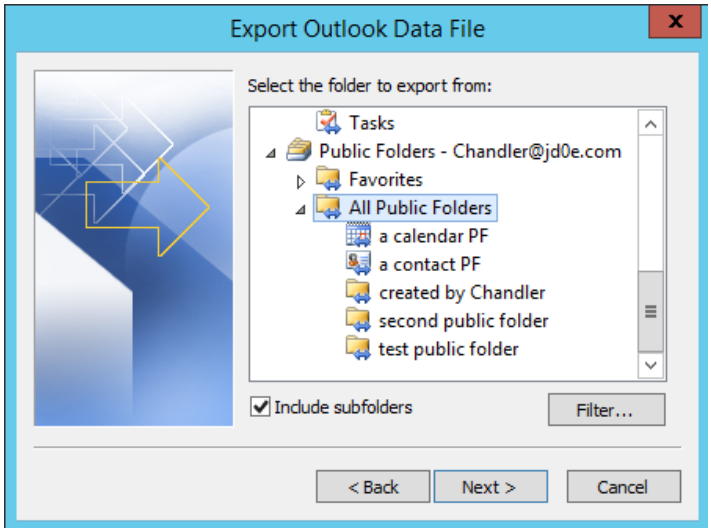
```
[PS] C:\>echo $members

Group  PrimarySmtpAddress
-----  -------------------
Cast   Ross@jd0e.com
Cast   Joey@jd0e.com
Cast   Monica@jd0e.com
Cast   Rachel@jd0e.com
Cast   Chandler@jd0e.com
Cast   Phoebe@jd0e.com
Cast   Janice@jd0e.com
Others Janice@jd0e.com
```

4. Let's export the contacts:

```
Get-Contact -ResultSize unlimited | select identity, name,displayname,firstname,lastname,windowsemailaddress | export-csv contacts.csv
```

```
DisplayName   WindowsEmailAddress
-----------   -------------------
Zsolt Agoston zsolt.agoston@lucasfettes.co.uk
Zsolt Gmail   zsolt@gmail.com
```

5. As the public folder migration is a little cumbersome between O365 and on-prem, even with modern public folder mailboxes, we simply **export all the public folders** from an Outlook client into a PST file for future ingestion



Migrate users to agzsolt.com on-prem

Now we create a migration batch that will synchronize and move the jd0e.com mailboxes to the on-prem endpoint. We create our first batch called "**First Ones**" from **first.csv**:

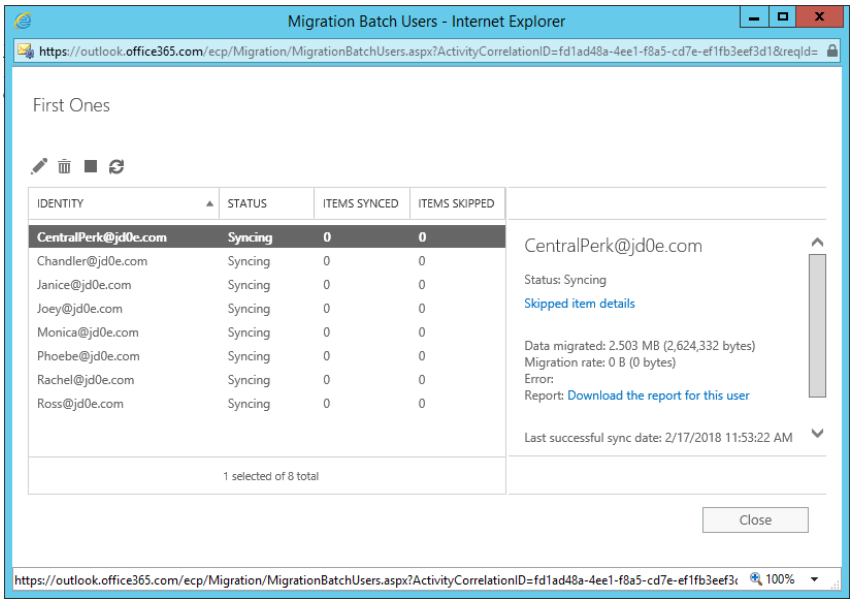
first.csv

```
EmailAddress
Ross@jd0e.com
Joey@jd0e.com
Monica@jd0e.com
Rachel@jd0e.com
Chandler@jd0e.com
Phoebe@jd0e.com
Janice@jd0e.com
CentralPerk@jd0e.com
```

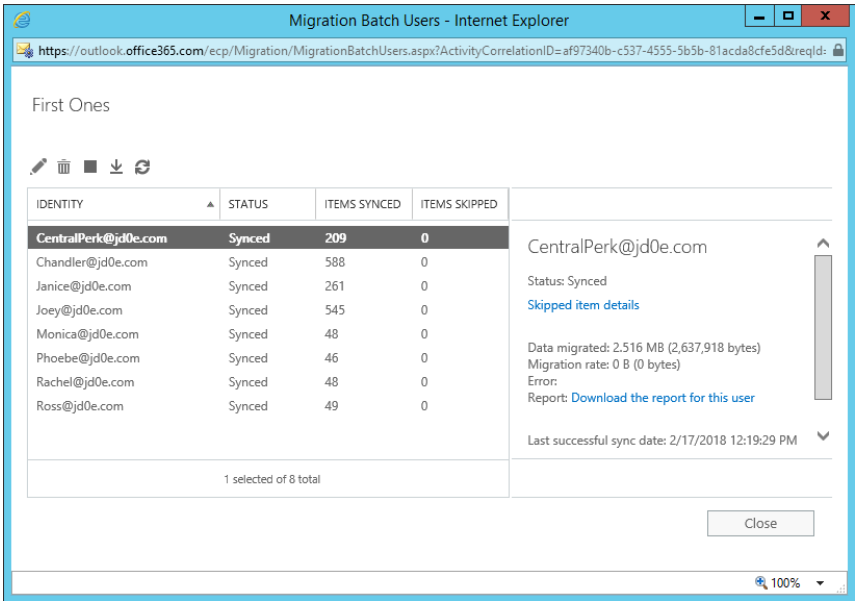
Script

```
New-MigrationBatch -Name "First Ones" -TargetEndpoint "agzsolt" -BadItemLimit unlimited -TargetDeliveryDomain agzsolt.com -CSVData
([System.IO.File]::ReadAllBytes("C:\first.csv")) -TargetDatabases "edebdf60-047b-4542-a36c-e923f9f14d9d" -Verbose
```

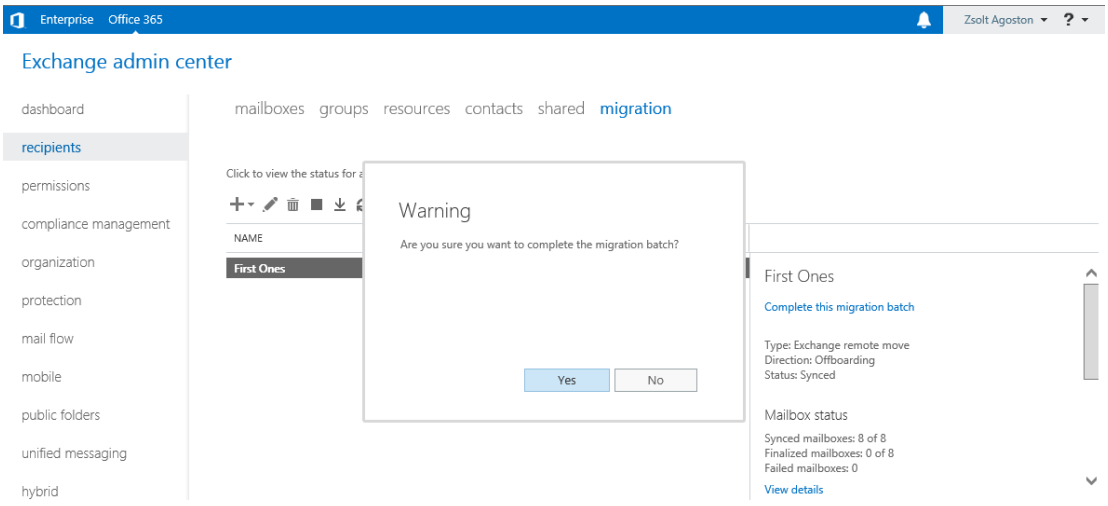
```
PS C:\> New-MigrationBatch -Name "First Ones" -TargetEndpoint "agzsolt" -BadItemLimit unlimited -TargetDeliveryDomain agzsolt.com -CSVData (
[System.IO.File]::ReadAllBytes("C:\first.csv")) -TargetDatabases "edebdf60-047b-4542-a36c-e923f9f14d9d" -Verbose
VERBOSE: Do you want to import the CSV file "First Ones" to migrate mailboxes for "jd0e.onmicrosoft.com"?

Identity   Status  Type              TotalCount
--------   ------  ----              ----------
First Ones Stopped ExchangeRemoteMove 8

PS C:\> _
```



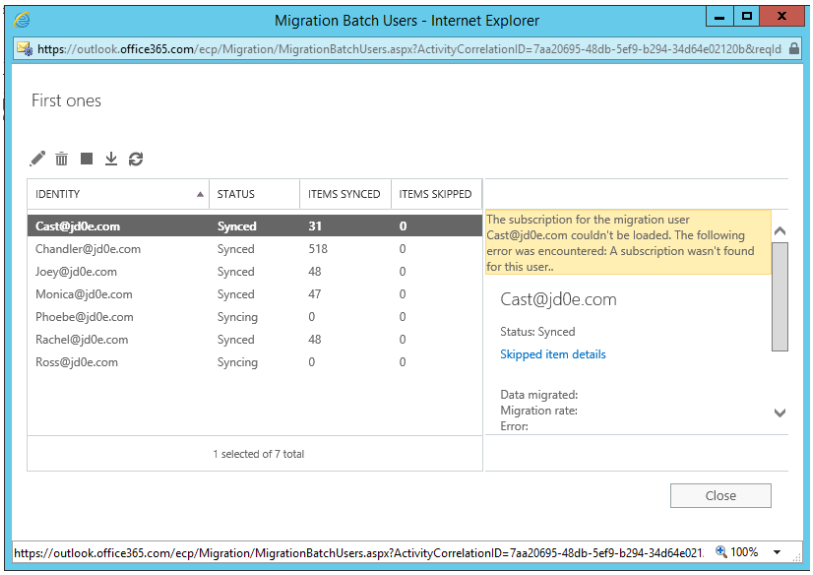
When it's done we are ready to complete the whole batch (or just specific mailboxes, it depends on our needs)



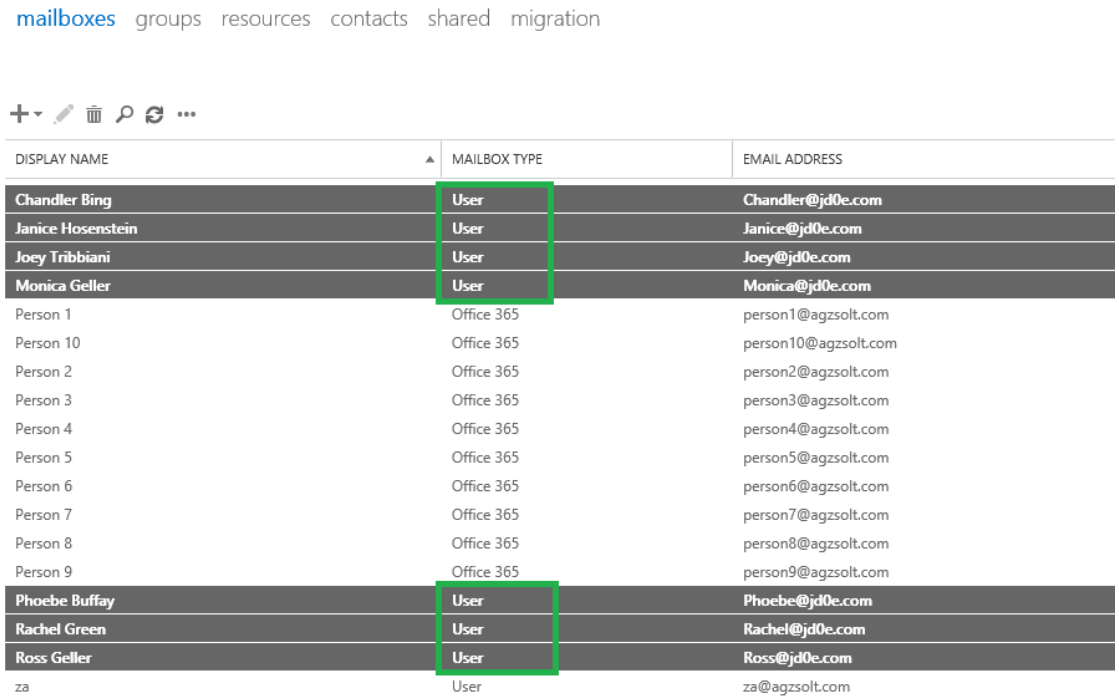
We **complete the migration batch** and make sure the users can access their mailboxes and they are functional



Just to mention: the shared mailboxes need a license to be assigned to them



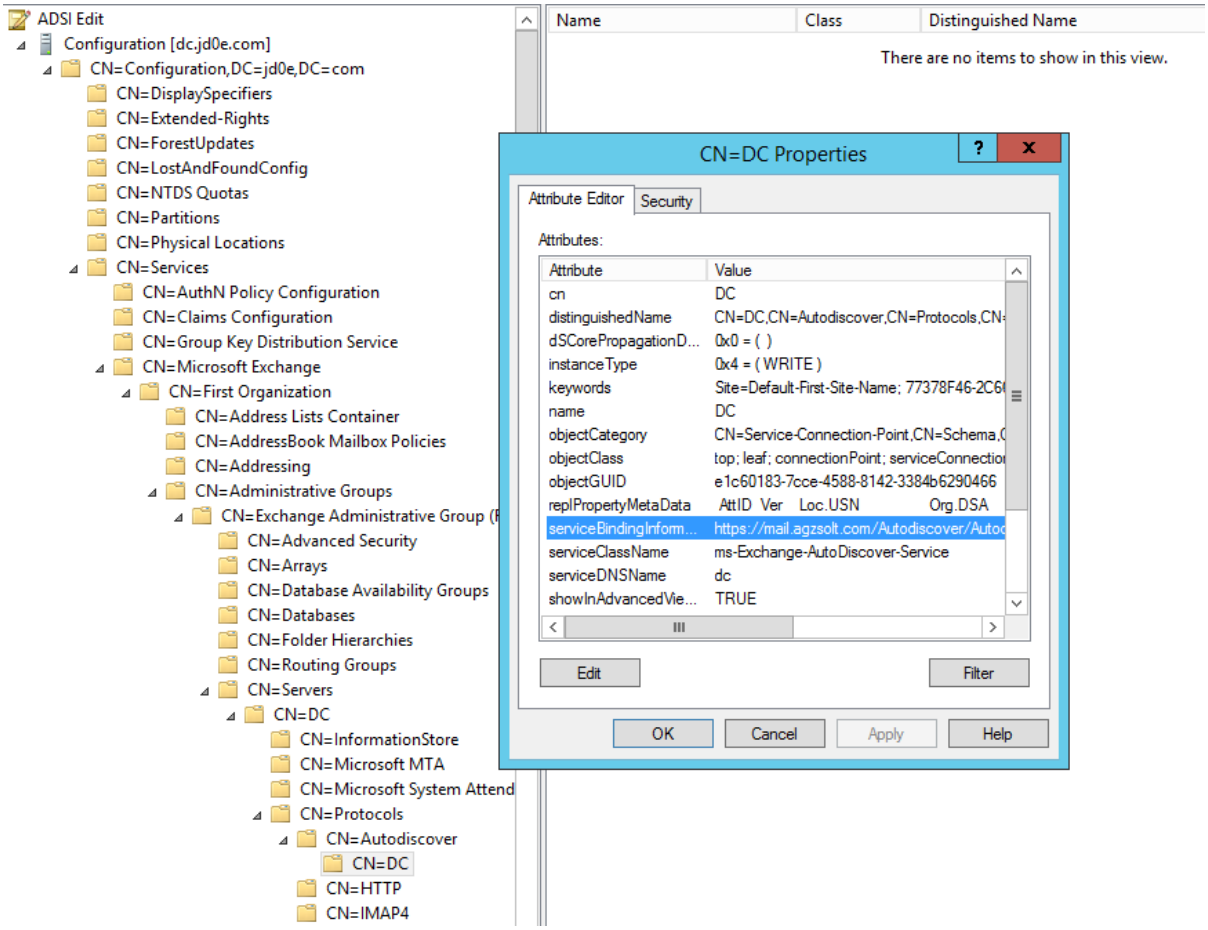
After the migration, we see that the on-prem server handles the mailboxes as local mailboxes:



At this point we change the local **SCP** record for **autodiscover** on the jd0e.com (source) **domain controller**, because this is the first place the server is looking for the autodiscover.xml data file which will not return the right values. We change that to point to the agzsolt.com (destination) autodiscover file:

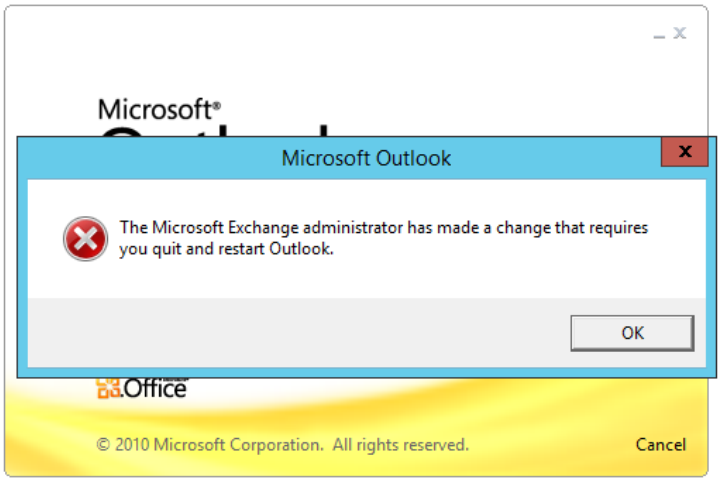
```
https://mail.agzsolt.com/Autodiscover/Autodiscover.xml
```

This step ensures un-broken Outlook functionality for all the users with domain-joined computers.



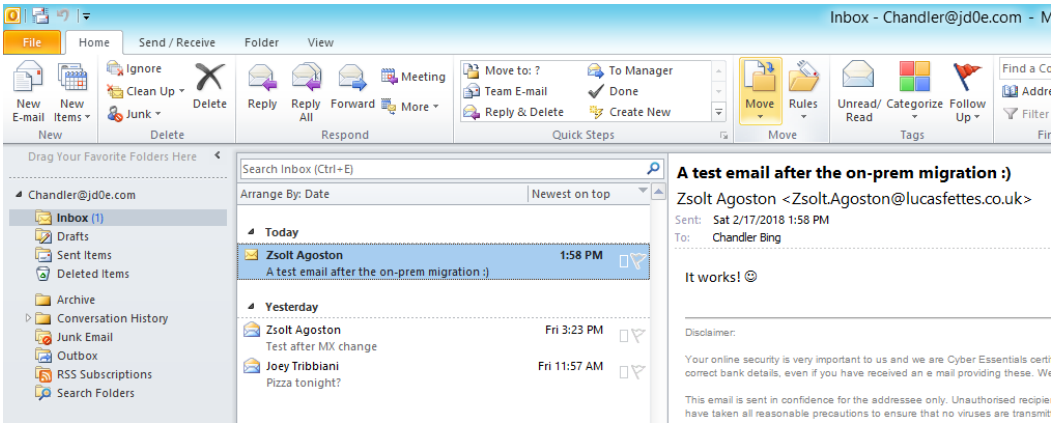
Note that the windows Outlook client is relying on the autodiscover record to locate public folders and it checks the SCP record first when it's opened. If we changed it before moving the public folder mailbox to the new server, it would give us an error message on the client side!

Once the users try to log in, Outlook notifies them of the changes in the background - the server name will be updated in the profile - and the users will be asked to close and open Outlook again, just like after a cloud migration:



Now another good news is that from this point if a user needed to re-create his/her Outlook profile on a domain-joined client (even on the source-side in the jd0e.com forest), they won't be prompted for their password, as long as the passwords match on the original jd0e.com and the new agzsolt.com forests - the kerberos ticketing will work flawlessly

After logging in, we test the mail flow:



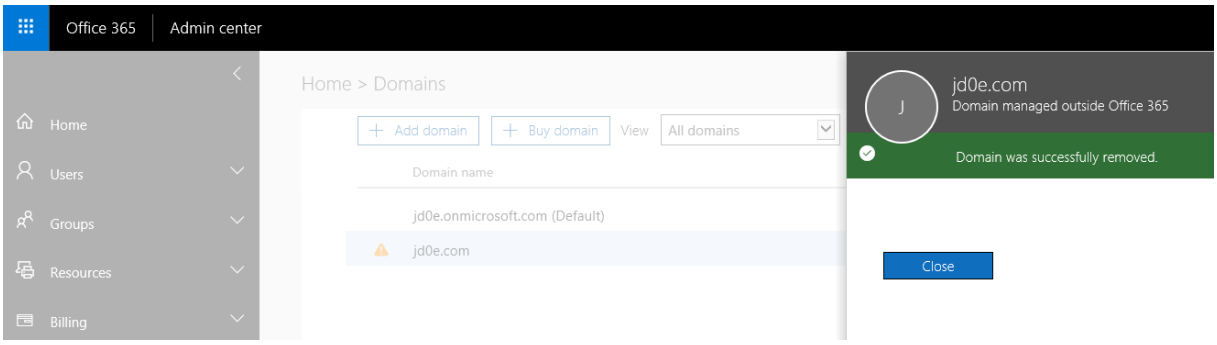
Excellent! The mail flow works in both directions!

Before we forget, we remove the old **jd0e.onmicrosoft.com STMP addresses** from the moved mailboxes.

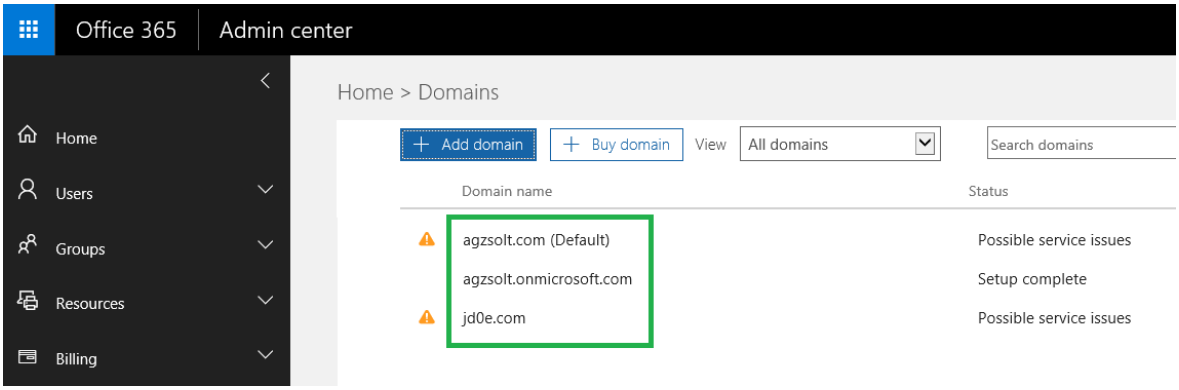
```
import-csv users.csv | foreach {
$name=$_.FirstName
Set-Mailbox -Identity $name -EmailAddresses @{remove="$name@jd0e.onmicrosoft.com" }
}
```

Strip down the old tenant

As a next step we remove of our **business domains** from the source tenant and add them to the target. In our example it is a single domain: **jd0e.com**

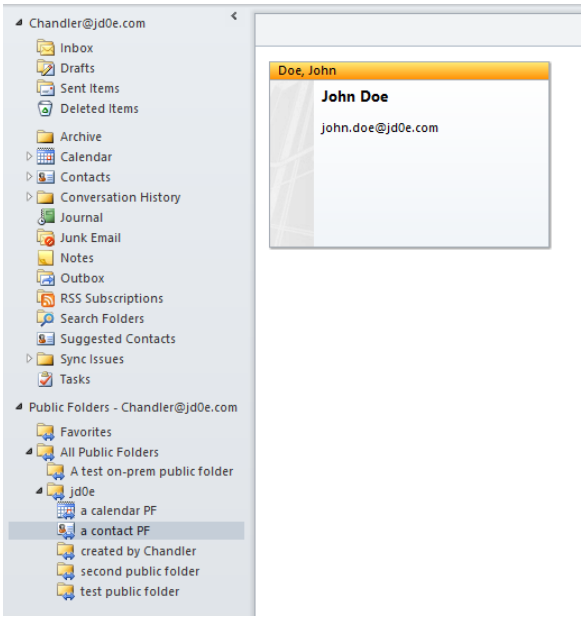


After removing jd0e.com domain from the jd0e.onmicrosoft.com tenant we **add that to agzsolt.onmicrosoft.com**



Perfect!

Now to sort the public folders we simply **create a jd0e subfolder in the local Public Folder database** and import the PF.pst file there



Perfect!

Sorting the post-migration tasks: permissions check, distribution lists and contacts creation

Amazing news that the **mailbox permissions are inherited**, they were mirrored during the migration so we don't need to worry about that

```
[PS] C:\>get-mailbox centralperk | Get-MailboxPermission | select user,accessrights

User                                    AccessRights
----                                    ------------
NT AUTHORITY\SELF                       {FullAccess, ReadPermission}
NT AUTHORITY\SELF                       {FullAccess, ExternalAccount, ReadPermission}
AGZSOL\Ross Geller                      {FullAccess}
AGZSOL\Joey Tribbiani                   {FullAccess}
AGZSOL\Monica Geller                    {FullAccess}
AGZSOL\Rachel Green                     {FullAccess}
AGZSOL\Chandler Bing                    {FullAccess}
AGZSOL\Phoebe Buffay                    {FullAccess}
AGZSOL\Janice Hosenstein                {FullAccess}
AGZSOL\za                               {FullAccess}
AGZSOL\Domain Admins                    {FullAccess}
AGZSOL\Enterprise Admins                {FullAccess}
AGZSOL\Organization Management          {FullAccess}
NT AUTHORITY\SYSTEM                     {FullAccess}
NT AUTHORITY\NETWORK SERVICE            {ReadPermission}
AGZSOLT\za                              {FullAccess, DeleteItem, ReadPermission, ChangePermission, ChangeOwner}
AGZSOLT\Domain Admins                   {FullAccess, DeleteItem, ReadPermission, ChangePermission, ChangeOwner}
AGZSOLT\Enterprise Admins               {FullAccess, DeleteItem, ReadPermission, ChangePermission, ChangeOwner}
AGZSOLT\Organization Management         {FullAccess, DeleteItem, ReadPermission, ChangePermission, ChangeOwner}
AGZSOLT\Public Folder Management        {ReadPermission}
AGZSOLT\Delegated Setup                 {ReadPermission}
AGZSOLT\Exchange Servers                {FullAccess, ReadPermission}
AGZSOLT\Exchange Trusted Subsystem      {FullAccess, DeleteItem, ReadPermission, ChangePermission, ChangeOwner}
AGZSOLT\Managed Availability Servers    {ReadPermission}
```

Send-as permissions are moved through as well:

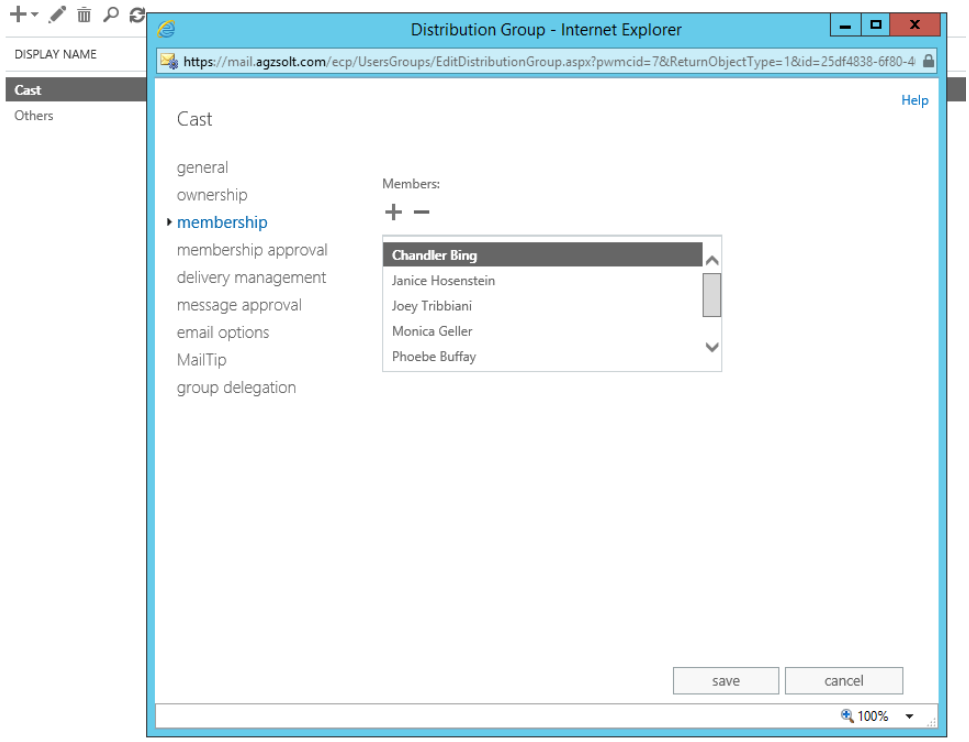
```
[PS] C:\>get-mailbox centralperk | get-ADPermission | where {$_.extendedrights -like "*send*"}

Identity              User                 Deny  Inherited
--------              ----                 ----  ---------
agzsolt.com/My Bu...  NT AUTHORITY\SELF    False False
agzsolt.com/My Bu...  AGZSOLT\Ross Geller  False False
agzsolt.com/My Bu...  AGZSOLT\Joey Trib... False False
agzsolt.com/My Bu...  AGZSOLT\Monica Ge... False False
agzsolt.com/My Bu...  AGZSOLT\Rachel Green False False
agzsolt.com/My Bu...  AGZSOLT\Chandler ... False False
agzsolt.com/My Bu...  AGZSOLT\Phoebe Bu... False False
agzsolt.com/My Bu...  AGZSOLT\Janice Ho... False False
agzsolt.com/My Bu...  AGZSOLT\CentralPerk  False False


[PS] C:\>_
```

Now we **create the distribution groups** with the right useraccounts

```
import-csv distro-list.csv | foreach {
$SAM=$_.SamAccountName
$name=$_.DisplayName
$win=$_.WindowsEmailAddress
New-DistributionGroup -Name $name -DisplayName $name -PrimarySmtpAddress $win -Type distribution -
IgnoreNamingPolicy:$true -ModerationEnabled:$false -OrganizationalUnit "OU=DGs,OU=jd0e,OU=My
Business,DC=agzsolt,DC=com" -Confirm:$false
Set-DistributionGroup -Identity $win -RequireSenderAuthenticationEnabled:$false
}
```

Populate with the members:

```
import-csv distro.csv | foreach {
$group=$_.Group
$member=$_.PrimarySmtpAddress
Add-DistributionGroupMember -Identity $group -Member $member -Confirm:$false
}
```



And to finish the process, we create the **contacts**:

```
import-csv contacts.csv | foreach {
$name=$_.Name
$disp=$_.DisplayName
$fn=$_.FirstName
$ln=$_.LastName
$email=$_.WindowsEmailAddress
New-MailContact -Name $name -DisplayName $disp -FirstName $fn -LastName $ln -ExternalEmailAddress $email -
OrganizationalUnit "OU=Contacts,OU=jd0e,OU=My Business,DC=agzsolt,DC=com" -Confirm:$false
}
```

From this point it's a normal migration to the cloud scenario.

MIGRATE BACK TO THE CLOUD

That's the easy and well documented part of our job, first we create the migration endpoint in the cloud server, just like we did the first time on the source tenant. This time we do the exact same steps, and we call this endpoint "**agzsolt**" as well.

An important thing is to move the OU that contains all the migrated accounts to an OU that is a **synced OU** so the users will appear in the tenant!

```
Move-ADObject -Identity "OU=jd0e,OU=non-syncing,OU=My Business,DC=agzsolt,DC=com" -TargetPath "OU=synced,OU=My Business,DC=agzsolt, DC=com"
```

We create a file called **UpToTheCloud.csv** and start the migration batch:

UpToTheCloud.csv

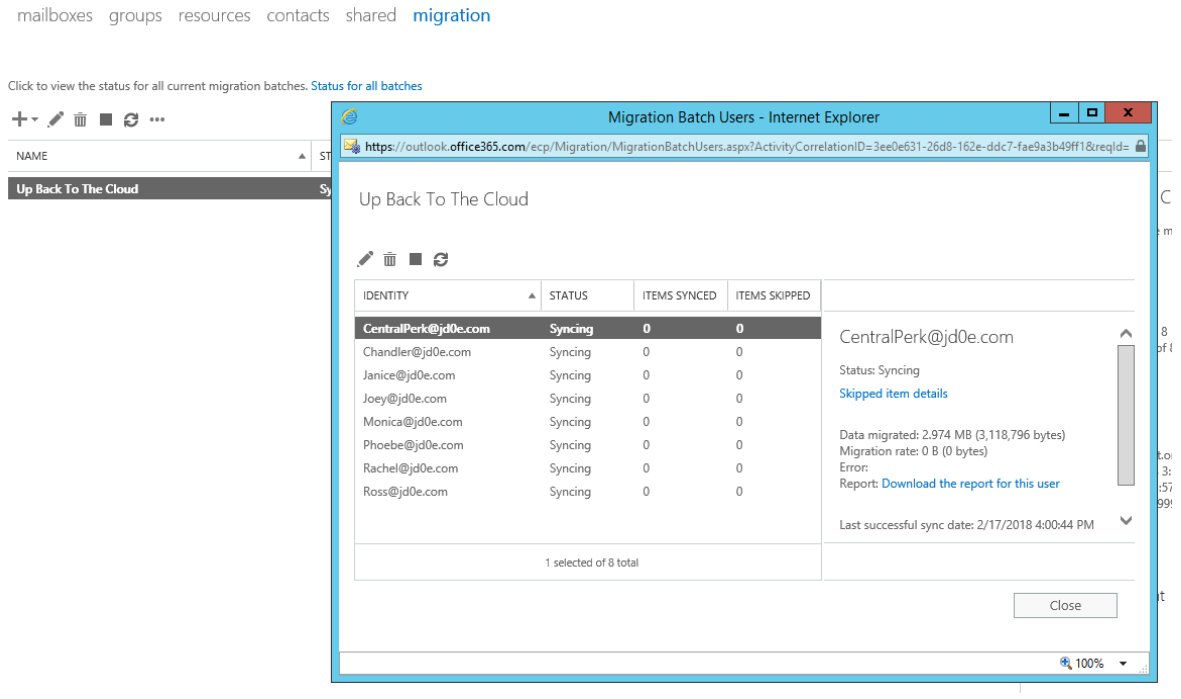
```
EmailAddress
Ross@jd0e.com
Joey@jd0e.com
Monica@jd0e.com
Rachel@jd0e.com
Chandler@jd0e.com
Phoebe@jd0e.com
Janice@jd0e.com
CentralPerk@jd0e.com
```

Script:

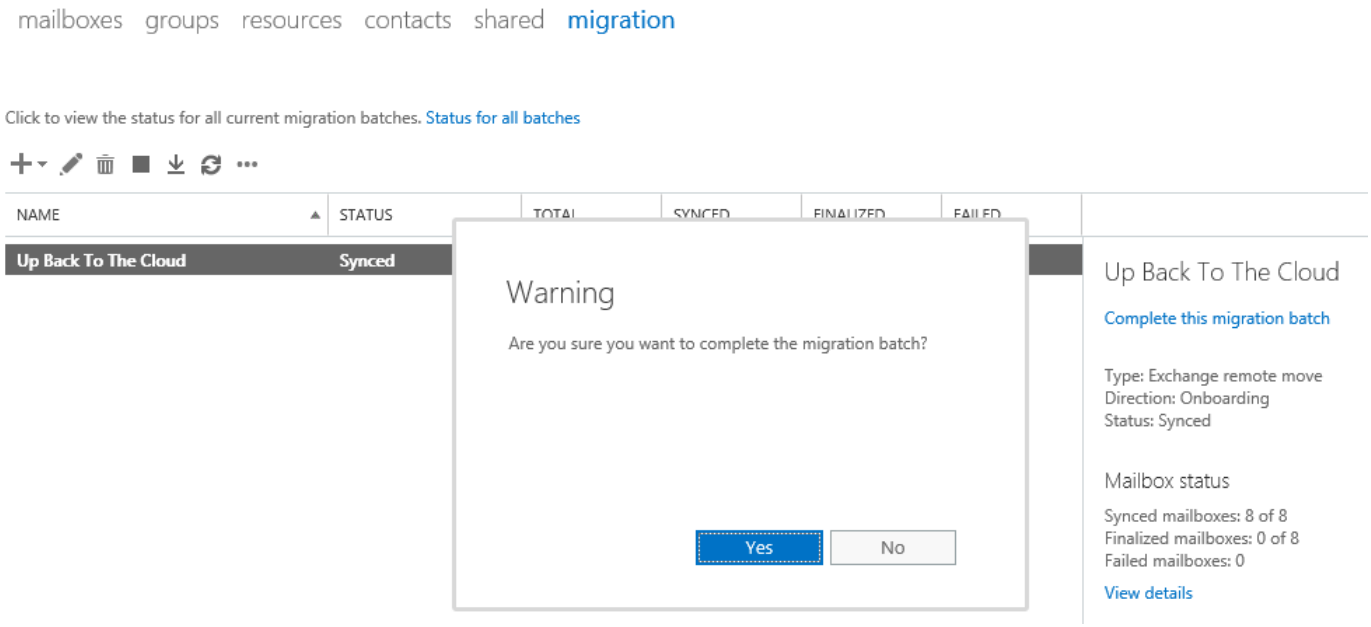
```
New-MigrationBatch -Name "Up Back To The Cloud" -SourceEndpoint "agzsolt" -BadItemLimit unlimited -TargetDeliveryDomain agzsolt.mail.onmicrosoft.com -CSVData ([System.IO.File]::ReadAllBytes("C:\UpToTheCloud.csv"))
```

```
PS C:\> New-MigrationBatch -Name "Up Back To The Cloud" -SourceEndpoint "agzsolt" -BadItemLimit unlimited -TargetDeliveryDomain agzsolt.mail
.onmicrosoft.com -CSVData ([System.IO.File]::ReadAllBytes("C:\UpToTheCloud.csv"))

Identity            Status  Type              TotalCount
--------            ------  ----              ----------
Up Back To The Cloud Stopped ExchangeRemoteMove 8


PS C:\>
```

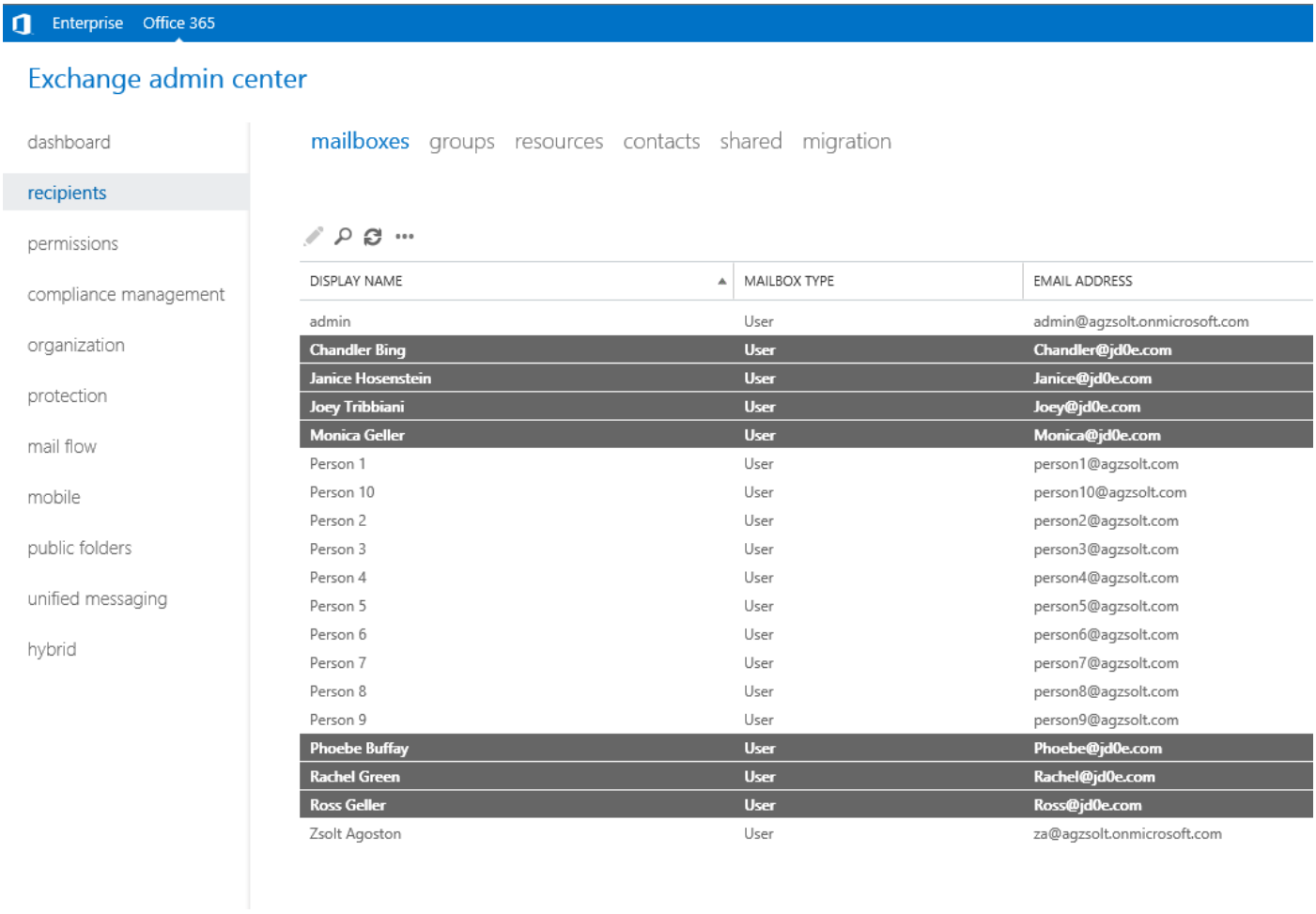
Start the migration



Now it's time to complete the migration batch

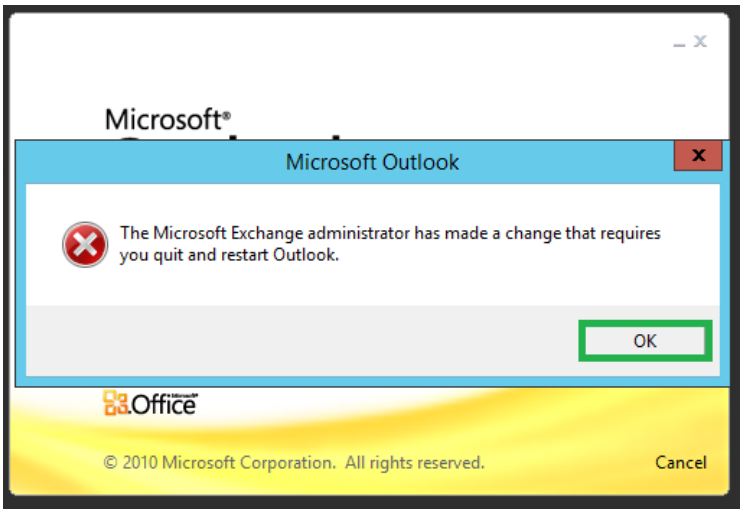


After the batch is done we can see the mailboxes finally appearing in the cloud



I personally like to **change the SCP domain for autodiscover** in the jd0e domain to https://autodiscover-s.outlook.com/Autodiscover/Autodiscover.xml which makes faster discovery at later profile creations, but this step can be omitted.

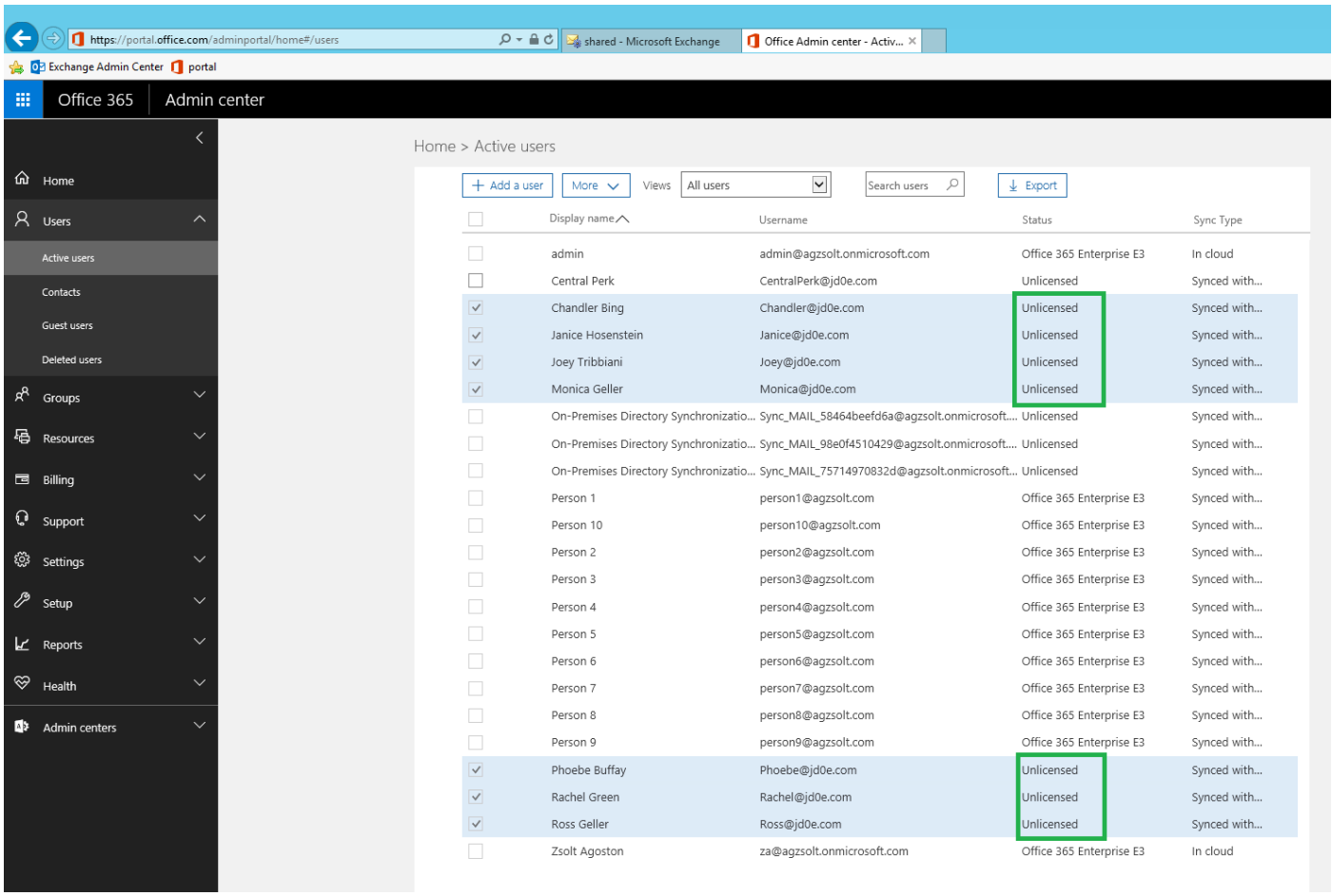
The clients will receive another notification of the background changes after which they need to restart Outlook again and we are done!



Tidying up

We still have a few things to sort:

1. We **assign the appropriate licenses** to the migrated accounts

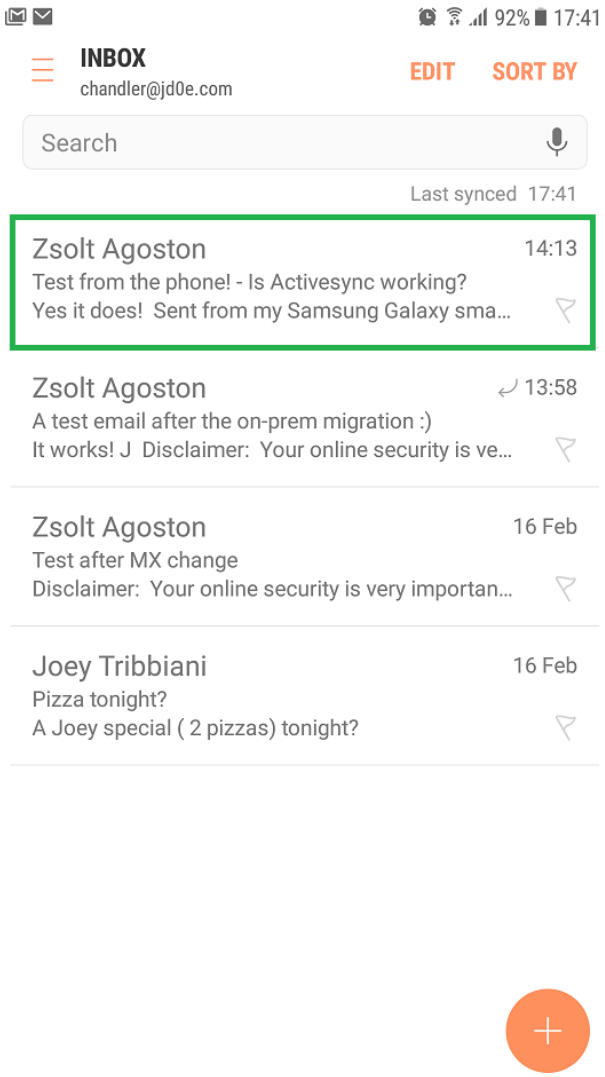



```
import-csv users.csv | foreach {
$fn=$_.FirstName
Set-MsolUser -UserPrincipalName "$fn@jd0e.com" -UsageLocation "GB"
Set-MsolUserLicense -UserPrincipalName "$fn@jd0e.com" -AddLicenses "agzsolt:ENTERPRISEPACK"
}
```

2. The **public folders need to be moved to the cloud**. Again, Microsoft's solution is a pretty cumbersome way, since the mailbox database is pretty small I use a simple client to export them in a PST file and import it back to a cloud managed PF mailbox.

After migrating back to the cloud, the **mobile phones will start working again**. In few cases users are prompted for their passwords by the device after which the connection goes back to normal.

Samsung S7 phone



We are done!

Zsolt Agoston (agzsolt@gmail.com)
17/02/2018