

# Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps

Authors: Yordan Bechev  
Premier Field Engineer at Microsoft  
[yordan.bechev@microsoft.com](mailto:yordan.bechev@microsoft.com)

Yorick Kuijs  
Premier Field Engineer at Microsoft  
[yorick.kuijs@microsoft.com](mailto:yorick.kuijs@microsoft.com)

Date: October 1<sup>st</sup> 2021  
Version: v1.2



## Disclaimer

This document is provided "as-is." Information and views expressed in this document, including URL and other Internet web site references, may change without notice. You bear the risk of using it.

This document does not provide you with any legal rights to any intellectual property in any Microsoft product. You may copy and use this document for your internal, reference purposes.

© 2021 Microsoft Corporation. All rights reserved.

## Changelog

Version	Date	Author	Changes
1.0	November 1st 2020	Yordan Bechev Yorick Kuijs	First release
1.0.1	November 3rd 2020	Yorick Kuijs	Updated incorrect links
1.1	December 2nd 2020	Yorick Kuijs	Incorporated feedback from Zaki Semar Shahul Added Azure Conditional Access for the used service account
1.2	October 1st 2021	Yorick Kuijs	Corrected issues Added Certificate authentication scenario

## Table of Contents

1	Introduction.....	5
2	Prerequisites.....	6
3	Preparation .....	7
3.1	Create a DSC account in Microsoft 365 .....	7
3.2	Create a new project in Azure DevOps.....	7
3.3	Create an Agent Pool in Azure DevOps.....	7
3.4	Create Personal Access Token.....	10
3.5	Configure Azure DevOps Agent on the virtual machine.....	13
3.6	Configure Azure Key Vault.....	17
3.6.1	Create Service Principle Name .....	17
3.6.2	Create Azure KeyVault.....	18
3.6.3	Add secrets to your Vault .....	22
3.6.4	Adding Service Connection to the Azure DevOps project.....	23
3.7	Configure the Local Configuration Manager.....	27
4	Configuring Azure DevOps.....	30
4.1	Populate scripts.....	30
4.2	Configure Azure DevOps project .....	35
4.2.1	Create Build pipeline.....	35
4.2.2	Create Release pipeline .....	37
4.2.3	Validate that changes to the config are deployed successfully .....	46
5	Security Enhancements .....	50
5.1	Using Azure Conditional Access to secure service account.....	50
5.2	Using Certificates instead of Username/Password for authentication .....	54
5.2.1	Creating the authentication certificate .....	55
5.2.2	Adding certificate to Azure KeyVault .....	56
5.2.3	Adding the certificate password to Azure KeyVault .....	57
5.2.4	Create an App Registration in Azure Active Directory .....	58
5.2.5	Updating the DSC configuration with the certificate thumbprint .....	65
5.2.6	Creating the Build and Release pipelines .....	66
6	Script details.....	67
7	Learning materials.....	68
7.1	Desired State Configuration.....	68
7.2	Microsoft365Dsc.....	69

# Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps

7.3	Git.....	69
8	Acronyms.....	70

## 1 Introduction

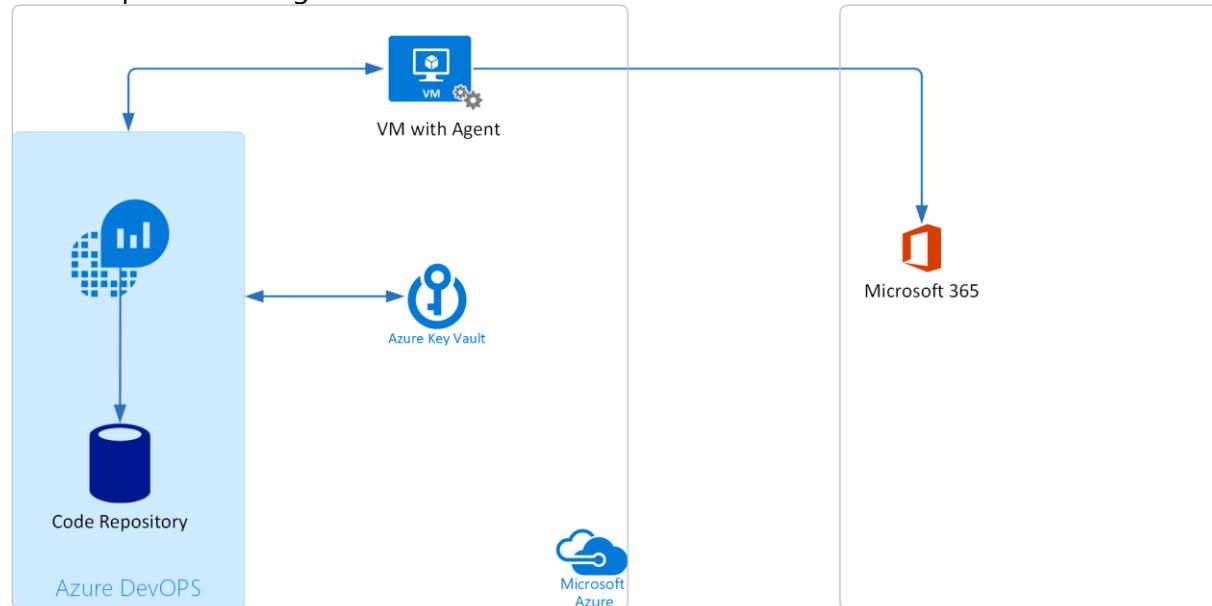
Microsoft 365 is the very popular productivity cloud solution of Microsoft. Each customer has its own tenant in which their data is stored. Using the Administration Portal (<https://admin.microsoft.com>) each customer can configure and manage their own tenant.

Many companies are adopting DevOps practices and are applying these practices against Microsoft 365 as well. Infrastructure as Code and Continuous Deployment/Continuous Integration are important concepts in DevOps.

Microsoft365Dsc is a PowerShell Desired State Configuration (DSC) module, which can configure and manage Microsoft 365 in a true DevOps style: Configuration as Code.

In this document we are going to describe the process and steps required to implement Configuration as Code using Microsoft365Dsc, Azure DevOps and Azure KeyVault. Changes to Microsoft 365 are done on a Git repository in Azure DevOps and then fully automatically deployed to a Microsoft 365 tenant.

The setup we are using is:



Chapter 5 "Security Enhancements" describe two alternatives that implement different scenarios to enhanced security.

## 2 Prerequisites

To deploy DSC configurations, we need a machine that will do the actual deployment to Microsoft 365. This can be a physical or virtual machine. In this guide we assume the use of a virtual machine. The requirements for this virtual machine are:

- Windows Server 2016 or above
  - .Net Framework 4.7 or higher
    - <https://dotnet.microsoft.com/download/dotnet-framework>
  - PowerShell v5.1
    - Installed by default on all current versions of Windows Server
  - Up to date PowerShellGet:  
Install-PackageProvider Nuget –Force  
Install-Module –Name PowerShellGet –Force
- Note:** If you run into issues downloading these updates, check out the following article: <https://devblogs.microsoft.com/powershell/powershell-gallery-tls-support/>
- A local account with administrative privileges, to deploy configurations from Azure DevOps

We are using Azure DevOps to store, compile and deploy the configurations. This means we need:

- An Azure DevOps tenant and permissions to configure this tenant
- A project in Azure DevOps

We also need a Microsoft 365 tenant, which is going to be managed using Microsoft365Dsc. In this tenant we need:

- An account with Global Administrator privileges, used to access the Admin Portal
- A service account with Global Administrative privileges, used to deploy setting using DSC
  - This account cannot be configured to use Multi-Factor Authentication
  - The actual required permissions depend on the used resources

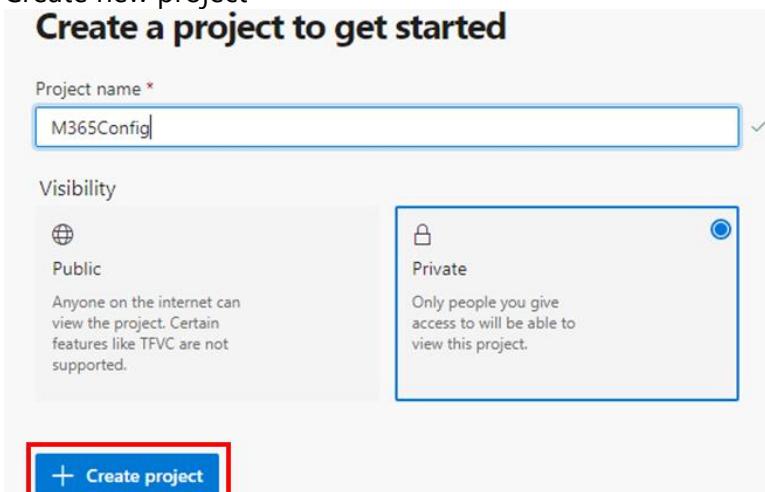
### 3 Preparation

#### 3.1 Create a DSC account in Microsoft 365

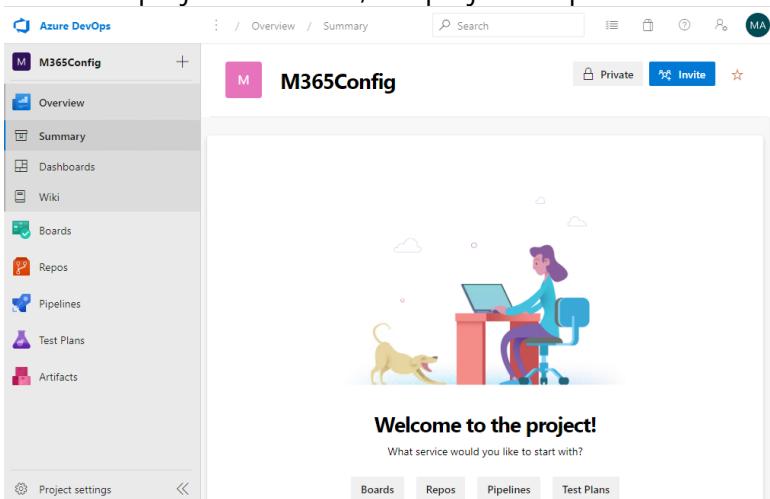
- Open an Internet browser
- Browse to the Microsoft 365 Admin Portal
- Create a new account
  - For example: DscConfigAdmin
  - Don't assign any license
  - Grant the user Global Admin permissions
    - More limited permissions possible depending on the resources in your configuration

#### 3.2 Create a new project in Azure DevOps

- Log into the Azure DevOps portal
- Create new project



- When the project is created, the project is opened automatically



#### 3.3 Create an Agent Pool in Azure DevOps

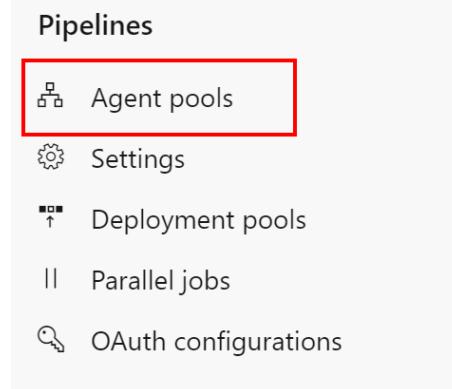
- Browse to the main Azure DevOps page
- Create a new Agent Pool

## Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps

- In Azure DevOps, click "Organization Settings" in the lower left corner



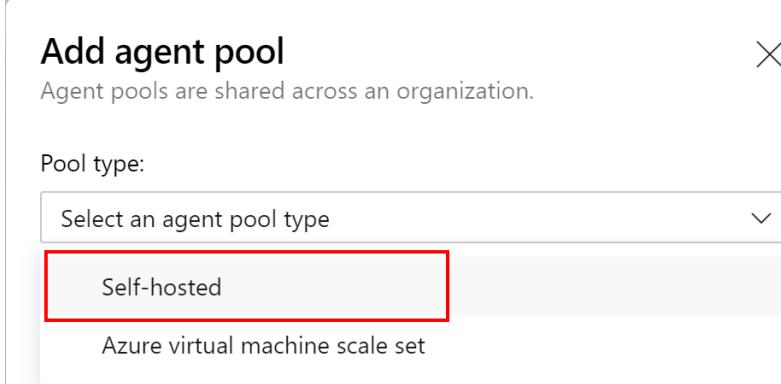
- Scroll down and under "Pipelines", click "Agent Pools"



- Create a new Agent Pool by clicking the "Add pool" button in the upper right corner



- Select "Self-hosted" as "Pool type"



- Enter a Name (for example: Microsoft365Dsc) and Description for the new pool and click "Create"

## Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps

Add agent pool X

Agent pools are shared across an organization.

Pool type:

Self-hosted

A pool of agents that you set up and manage on your own to run jobs. [Learn more](#).

Name:

Microsoft365Dsc

Description (optional):

Agent pool used for deploying DSC configurations to Microsoft 365

Markdown supported.

Pipeline permissions:

Grant access permission to all pipelines

Auto-provision this agent pool in all projects

Create

- Click the newly created pool to open the pool
- Click the "New agent" button to open the required information to add a new agent

Microsoft365DSC

Update all agents New agent

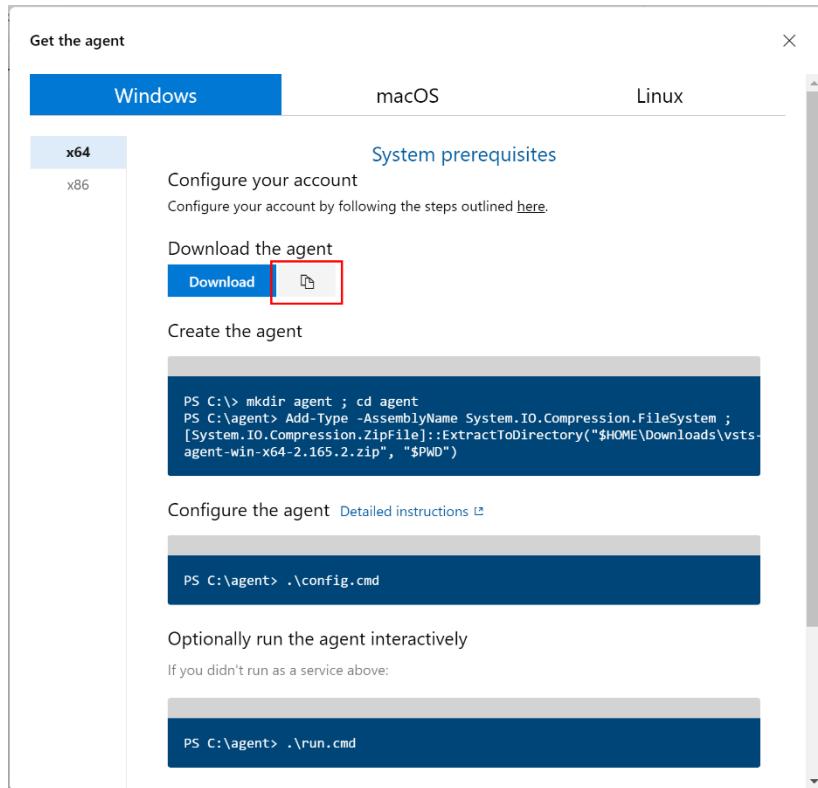
Jobs Agents Details Security Settings Maintenance History

No jobs have run on this agent pool

Run a pipeline on this agent pool to see more details

- Copy the download link and download the agent on the virtual machine

## Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps

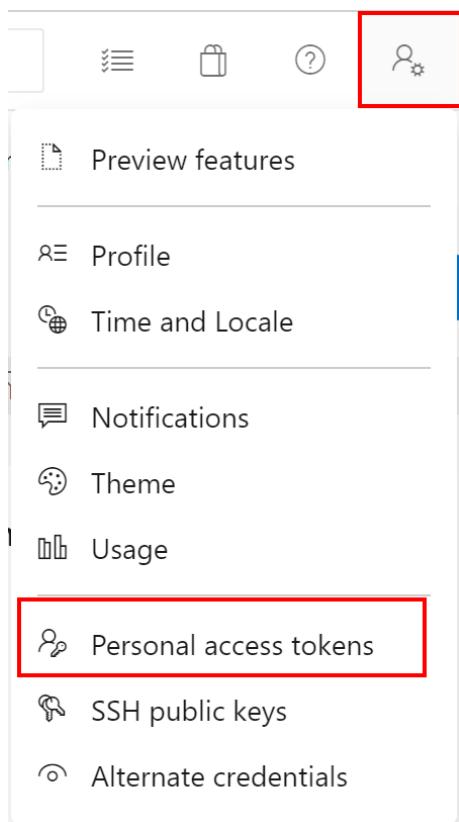


- Extract the downloaded zip file to the C:\Agent folder

### 3.4 Create Personal Access Token

- Open Azure DevOps
- Click the user icon in the upper right corner and select the "Personal access tokens" menu item

## Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps



- Click "New Token" to create a new token

### Personal Access Tokens

These can be used instead of a password for applications like Git or can be passed in the authorization header to access REST APIs

<a href="#">+ New Token</a>	Status	Organization	Expires on ↓

## Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps

- Enter a Name and select next year (not possible to select more than a year) as Expiration

Create a new personal access token

Name: DevOpsAgent

Organization: ykuujs

Expiration (UTC): Custom defined (4/16/2021)

Scopes:

Full access (radio button selected):

- Read, update, and delete releases, release pipelines, and stages
  - Read
  - Read, write, & execute
  - Read, write, execute, & manage
- Test Management
  - Read
  - Read & write
- Packaging
  - Read
  - Read & write
  - Read, write, & manage

Show all scopes (27 more)

Create Cancel

- Click "Show all scopes" and click "Create" to create the token

Create a new personal access token

Name: DevOpsAgent

Organization: ykuujs

Expiration (UTC): Custom defined (4/16/2021)

Scopes:

Custom defined (radio button selected):

Agent Pools

- Read
- Read & manage

Analytics

- Read

Auditing

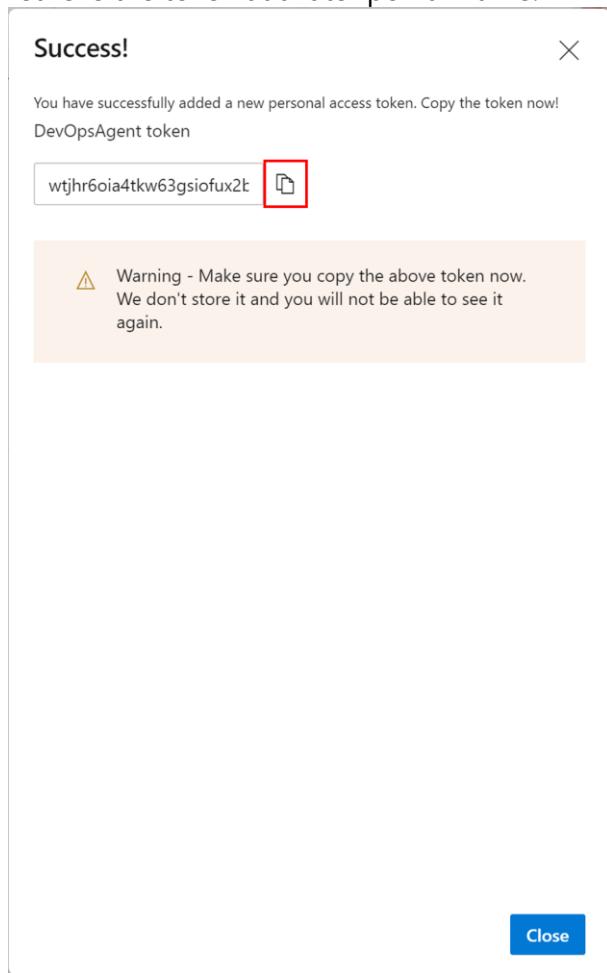
- Read Audit Log

Show less scopes

Create Cancel

## Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps

- **IMPORTANT:** Copy and store the generated token in a secure place. You cannot retrieve the token at a later point in time.



- Click "Close" to close the wizard. Your token is now created.

**Personal Access Tokens**  
These can be used instead of a password for applications like Git or can be passed in the authorization header to access REST APIs

New Token	Status	Organization	Expires on
DevOpsAgent Agent Pools (Read & manage)	Active	ykujis	4/16/2021

### 3.5 Configure Azure DevOps Agent on the virtual machine

- Connect to your virtual machine
- Create a service account, either local or domain, for the Azure DevOps agent.
  - NOTE: The account needs local Administrator permissions to be able to push configurations to the Local Configuration Manager.
- Open an elevated Command Prompt
- Browse to the C:\Agent folder
- Run config.cmd

A screenshot of an "Administrator: Command Prompt" window. The command "c:\Agent>config.cmd" is being typed into the prompt.

- Enter the Server URL as [https://dev.azure.com/<org\\_name>](https://dev.azure.com/<org_name>) and press [Enter]

```
Administrator: Command Prompt - config.cmd
c:\Agent>config.cmd

\Microsoft\PowerShell
agent v2.166.3          (commit 87e2e0a)

>> Connect:
Enter server URL > https://dev.azure.com/M365Automation
```

- Press [Enter] to use the Personal Access Token for authentication

```
Administrator: Command Prompt - config.cmd
c:\Agent>config.cmd

\Microsoft\PowerShell
agent v2.166.3          (commit 87e2e0a)

>> Connect:
Enter server URL > https://dev.azure.com/M365Automation
Enter authentication type (press enter for PAT) >
```

- Paste the Personal Access Token and press [Enter]

```
Administrator: Command Prompt - config.cmd
c:\Agent>config.cmd

\Microsoft\PowerShell
agent v2.166.3          (commit 87e2e0a)

>> Connect:
Enter server URL > https://dev.azure.com/M365Automation
Enter authentication type (press enter for PAT) >
Enter personal access token > *****
```

## Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps

- Enter "Microsoft365Dsc" (use the name specified earlier) as the Agent Pool and press [Enter]

```
Administrator: Command Prompt - config.cmd
c:\Agent>config.cmd

\Windows\system32\cmd.exe
agent v2.166.3           (commit 87e2e0a)

>> Connect:
Enter server URL > https://dev.azure.com/M365Automation
Enter authentication type (press enter for PAT) >
Enter personal access token > *****
Connecting to server ...

>> Register Agent:
Enter agent pool (press enter for default) > Microsoft365DSC
```

- Enter a custom Agent name or press [Enter] to use the server name (max 15 characters)

```
Administrator: Command Prompt - config.cmd
c:\Agent>config.cmd

\Windows\system32\cmd.exe
agent v2.166.3           (commit 87e2e0a)

>> Connect:
Enter server URL > https://dev.azure.com/M365Automation
Enter authentication type (press enter for PAT) >
Enter personal access token > *****
Connecting to server ...

>> Register Agent:
Enter agent pool (press enter for default) > Microsoft365DSC
Enter agent name (press enter for M365AUTOMATIONV) >
```

- The Agent checks some prerequisites. Press [Enter] to use the default work folder

```
Administrator: Command Prompt - config.cmd
c:\Agent>config.cmd

\Windows\system32\cmd.exe
agent v2.166.3           (commit 87e2e0a)

>> Connect:
Enter server URL > https://dev.azure.com/M365Automation
Enter authentication type (press enter for PAT) >
Enter personal access token > *****
Connecting to server ...

>> Register Agent:
Enter agent pool (press enter for default) > Microsoft365DSC
Enter agent name (press enter for M365AUTOMATIONV) >
Scanning for tool capabilities.
Connecting to the server.
Successfully added the agent
Testing agent connection.
Enter work folder (press enter for _work) >
```

## Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps

- If prompted: Press Enter to acknowledge "N" for "Perform an unzip for each step"
- Type "Y" to run the agent as a service

```
Administrator: Command Prompt - config.cmd
c:\Agent>config.cmd

agent v2.166.3          (commit 87e2e0a)

>> Connect:
Enter server URL > https://dev.azure.com/M365Automation
Enter authentication type (press enter for PAT) >
Enter personal access token > *****
Connecting to server ...

>> Register Agent:
Enter agent pool (press enter for default) > Microsoft365DSC
Enter agent name (press enter for M365AUTOMATIONV) >
Scanning for tool capabilities.
Connecting to the server.
Successfully added the agent
Testing agent connection.
Enter work folder (press enter for _work) >
2020-04-23 08:39:27Z: Settings Saved.
Enter run agent as service? (Y/N) (press enter for N) > Y
```

- Enter the created service account credentials and press [Enter]

```
Administrator: Command Prompt - config.cmd
c:\Agent>config.cmd

>> Connect:
Enter server URL > https://dev.azure.com/M365Automation
Enter authentication type (press enter for PAT) >
Enter personal access token > *****
Connecting to server ...

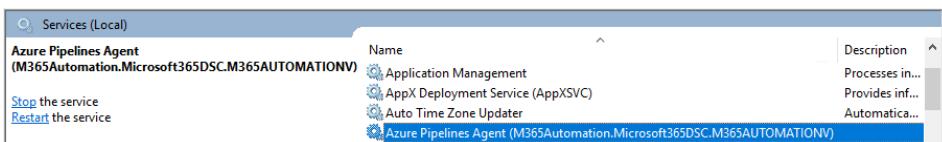
>> Register Agent:
Enter agent pool (press enter for default) > Microsoft365DSC
Enter agent name (press enter for M365AUTOMATIONV) >
Scanning for tool capabilities.
Connecting to the server.
Successfully added the agent
Testing agent connection.
Enter work folder (press enter for _work) >
2020-04-23 08:39:27Z: Settings Saved.
Enter run agent as service? (Y/N) (press enter for N) > Y
Enter User account to use for the service (press enter for NT AUTHORITY\NETWORK SERVICE) > M365AutomationV\M365ConfigAge
ntSvc
Enter Password for the account M365AutomationV\M365ConfigAgentSvc > *****
```

- The agent is being configured and started automatically.

```
Administrator: Command Prompt
Enter personal access token > *****
Connecting to server ...

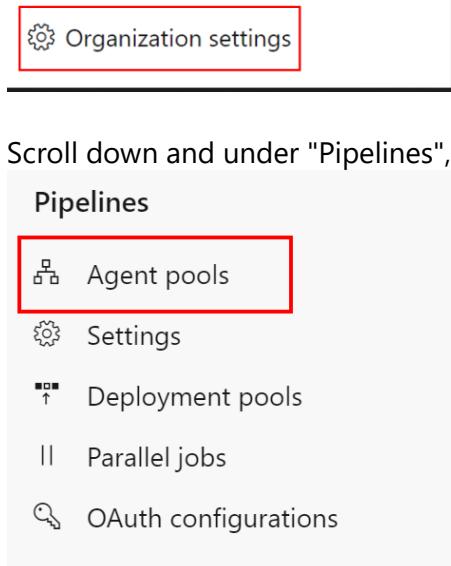
>> Register Agent:
Enter agent pool (press enter for default) > Microsoft365DSC
Enter agent name (press enter for M365AUTOMATIONV) >
Scanning for tool capabilities.
Connecting to the server.
Successfully added the agent
Testing agent connection.
Enter work folder (press enter for _work) >
2020-04-23 08:39:27Z: Settings Saved.
Enter run agent as service? (Y/N) (press enter for N) > Y
Enter User account to use for the service (press enter for NT AUTHORITY\NETWORK SERVICE) > M365AutomationV\M365ConfigAge
ntSvc
Enter Password for the account M365AutomationV\M365ConfigAgentSvc > *****
Error reported in diagnostic logs. Please examine the log for more details.
  C:\Agent\diag\Agent_20200423-083409-utc.log
Granting file permissions to 'M365AutomationV\M365ConfigAgentSvc'.
Service vstsagent.M365Automation.Microsoft365DSC.M365AUTOMATIONV successfully installed
Service vstsagent.M365Automation.Microsoft365DSC.M365AUTOMATIONV successfully set recovery option
Service vstsagent.M365Automation.Microsoft365DSC.M365AUTOMATIONV successfully set to delayed auto start
Service vstsagent.M365Automation.Microsoft365DSC.M365AUTOMATIONV successfully configured
Service vstsagent.M365Automation.Microsoft365DSC.M365AUTOMATIONV started successfully

c:\Agent>
```

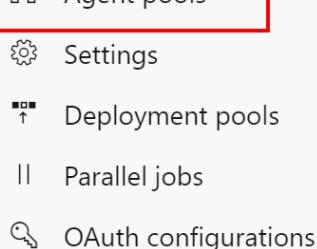


## Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps

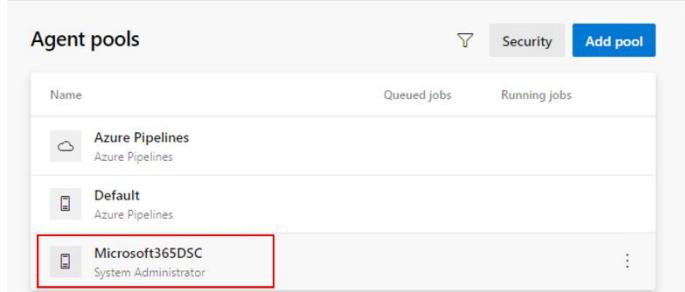
- Verify agent is successfully registered in Azure DevOps
  - Open the Azure DevOps portal
  - Click "Organization Settings" in the lower left corner



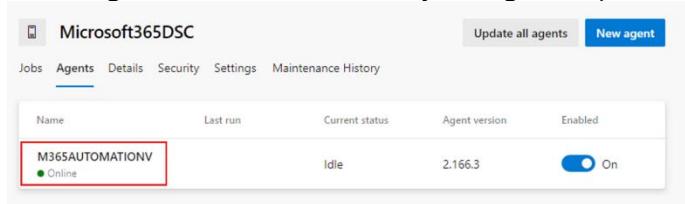
- Scroll down and under "Pipelines", click "Agent Pools"



- Click your custom Agent Pool



- Click "Agents" and validate that your agent is present and Online



## 3.6 Configure Azure Key Vault

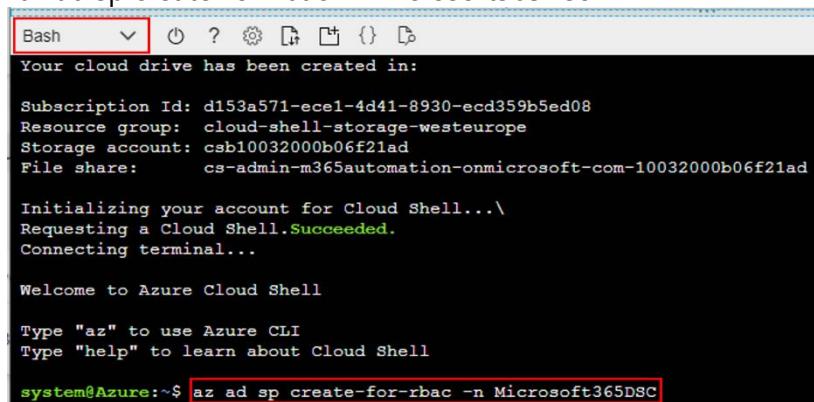
### 3.6.1 Create Service Principle Name

- Log into the Azure Portal
- Open Cloud Shell



## Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps

- Make sure you select the Bash shell and enter:  
"az ad sp create-for-rbac -n Microsoft365Dsc"



```
Bash
Your cloud drive has been created in:
Subscription Id: d153a571-ece1-4d41-8930-ecd359b5ed08
Resource group: cloud-shell-storage-westeurope
Storage account: csb10032000b06f21ad
File share: cs-admin-m365automation-onmicrosoft-com-10032000b06f21ad

Initializing your account for Cloud Shell...\nRequesting a Cloud Shell.Succeeded.\nConnecting terminal...

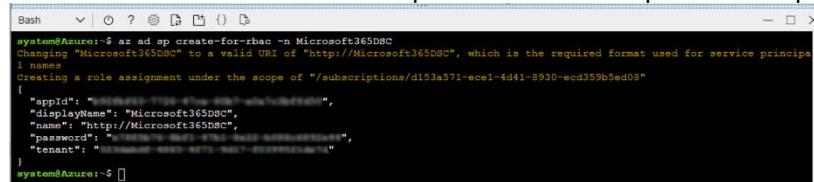
Welcome to Azure Cloud Shell

Type "az" to use Azure CLI
Type "help" to learn about Cloud Shell

system@Azure:~$ az ad sp create-for-rbac -n Microsoft365DSC
```

**Note:** You can change "Microsoft365Dsc" for a custom string.

- Azure creates the Service Principal Name and outputs the required information.



```
Bash
system@Azure:~$ az ad sp create-for-rbac -n Microsoft365DSC
Changing "Microsoft365DSC" to a valid URI of "http://Microsoft365DSC", which is the required format used for service principals
Creating a role assignment under the scope of "/subscriptions/d153a571-ece1-4d41-8930-ecd359b5ed08"
{
  "appId": "e1e6a49d-110d-4f0c-9077-aef77a000000",
  "displayName": "Microsoft365DSC",
  "name": "http://Microsoft365DSC",
  "password": "XXXXXXXXXXXXXXXXXXXXXX",
  "tenant": "72f988bf-86f1-41af-91ab-2d7cd011db4b"
}
system@Azure:~$
```

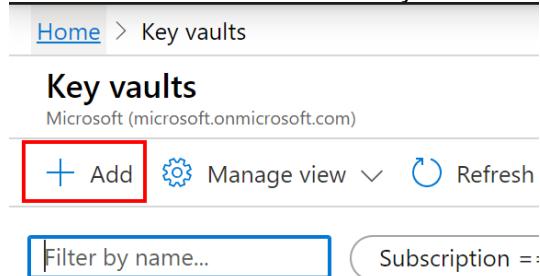
- Copy this information and store it in a secure place

### 3.6.2 Create Azure KeyVault

- Log into the Azure Portal
- Enter "Keyvault" in the top search bar and select "Key vaults"



- Click "Add" to create a new Key Vault



Home > Key vaults

## Key vaults

Microsoft (microsoft.onmicrosoft.com)

+ Add Manage view Refresh

Filter by name... Subscription =:

Showing 1 to 1 of 1 records.

## Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps

- Enter the desired Resource group, Name and Region and then click "Review + create"

Home > Key vaults > Create key vault

### Create key vault

Project details  
Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription \* Free Trial  
Resource group \* (New) KeyVaultDsc (New) KeyVaultDsc Create new

Instance details  
Key vault name \* M365DscAzureKeyVault M365DscAzureKeyVault  
Region \* (Europe) West Europe (Europe) West Europe

Pricing tier \* Standard  
Soft delete Enable Disable  
Retention period (days) \* 90  
Purge protection Enable Disable

Review + create < Previous Next : Access policy >

- Review the settings and click "Create" to create the Key Vault

Home > Key vaults > Create key vault

### Create key vault

Validation passed

Basics

Subscription	Free Trial
Resource group	KeyVaultDsc
Key vault name	M365DscAzureKeyVault
Region	West Europe
Pricing tier	Standard
Enable soft delete	Enabled
Enable purge protection	Enabled
Retention period (days)	90 days

Access policy

Azure Virtual Machines for deployment	Disabled
Azure Resource Manager for template deployment	Disabled
Azure Disk Encryption for volume encryption	Disabled
Azure Disk Encryption for volume encryption	Disabled

Permission model

Access control list

Create < Previous Next > Download a template for automation

- The KeyVault will be created

## Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps

- Go to the created KeyVault by clicking "Go to resource"

The screenshot shows the Microsoft Azure Key Vault Overview page for 'M365DscAzureKeyVault'. A prominent message box at the top right says 'Your deployment is complete' with a green checkmark icon. Below it, deployment details are listed: Deployment name: M365DscAzureKeyVault, Start time: 4/23/2020, 2:15:08 AM, Subscription: Free Trial, Correlation ID: 0ef7c008-0087-47f1-a062-61e312b321ac, Resource group: KeyVaultDsc. A table shows one resource: M365DscAzureKeyVault, Type: Microsoft.KeyVault/vaults, Status: OK, with a link to 'Operation details'. At the bottom, a blue button labeled 'Go to resource' is highlighted with a red box.

- Click "Access policies" and click "Add Access Policy"

The screenshot shows the Microsoft Azure Key Vault Access policies page for 'M365DscAzureKeyVault'. On the left, a sidebar lists options like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Events (preview), Keys, Secrets, Certificates, and Access policies, with 'Access policies' highlighted. The main area shows 'Enable Access to:' with three checkboxes: 'Azure Virtual Machines for deployment', 'Azure Resource Manager for template deployment', and 'Azure Disk Encryption for volume encryption'. Below is a red box around the blue '+ Add Access Policy' button. A table titled 'Current Access Policies' shows one entry: 'USER' System Administrator with email admin@M365Automati... and 9 selected key permissions. A red box highlights the 'Access policies' tab in the sidebar.

- Select the "Select principal" option, enter the Service Principal Name you created earlier in the search box on the right, select your principal and click "Select".

The screenshot shows the 'Add access policy' dialog. In the 'Select principal' dropdown, a search bar contains 'microsoft365dc' and a list below shows 'Microsoft365DC'. A red box highlights the search bar and the list item. To the right, a 'Selected member:' section shows 'Microsoft365DC' with a 'Remove' link. At the bottom right of the dialog, a blue 'Select' button is highlighted with a red box.

## Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps

- Open the "Secret permissions" and select the "Get" and "List" permissions

The screenshot shows the 'Add access policy' interface. In the 'Secret permissions' section, two checkboxes are selected: 'Get' and 'List'. These checkboxes are highlighted with a red box. Other options like 'Set', 'Delete', 'Recover', 'Backup', and 'Restore' are also listed but not selected.

- Validate that everything is configured correctly and click "Add"

The screenshot shows the 'Add access policy' interface with the following fields filled:

- Configure from template (optional): Empty dropdown.
- Key permissions: Empty dropdown.
- Secret permissions: 2 selected (dropdown).
- Certificate permissions: 0 selected (dropdown).
- Select principal: Microsoft365DSC (selected).
- Authorized application: None selected (dropdown).

The 'Add' button at the bottom is highlighted with a red box.

## Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps

- Validate that the access policy has been added and click "Save" to store the new policies.

The screenshot shows the 'Access policies' page for a Key Vault named 'M365DscAzureKeyVault'. At the top, there are 'Save', 'Discard', and 'Refresh' buttons. A message says 'Please click the 'Save' button to commit your changes.' Below this, under 'Enable Access to:', there are three checkboxes for 'Azure Virtual Machines for deployment', 'Azure Resource Manager for template deployment', and 'Azure Disk Encryption for volume encryption'. There is also a link to '+ Add Access Policy...'. The main table lists 'Current Access Policies' under 'APPLICATION' and 'USER'. The 'APPLICATION' section shows 'Microsoft365DSC' with '0 selected', '2 selected', and '0 selected' dropdowns. The 'USER' section shows 'System Administrator' with '9 selected', '7 selected', and '15 selected' dropdowns. A red box highlights the 'Save' button and another red box highlights the 'APPLICATION' section.

- Next you should see the message that the KeyVault was updated successfully

The screenshot shows the 'Access policies' page for the same Key Vault. A red box highlights a green circular icon with a checkmark and the text 'Updating the key vault 'M365DscAzureKey...' 234 AM. The key vault 'M365DscAzureKeyVault' has been successfully updated.' at the bottom of the page.

### 3.6.3 Add secrets to your Vault

- Click "Secrets" in the left menu

The screenshot shows the 'Overview' page for the 'M365DscAzureKeyVault'. The left sidebar has a 'Secrets' option highlighted with a red box. Other options like 'Keys', 'Certificates', and 'Access policies' are also visible.

## Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps

- Click "Generate/Import" to create a new secret

The screenshot shows the 'M365DscAzureKeyVault | Secrets' page in the Azure portal. On the left, there's a navigation menu with options like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Events (preview), Settings, Keys, Secrets (which is selected and highlighted in grey), and Certificates. At the top right, there are buttons for '+ Generate/Import', 'Refresh', and 'Restore Backup'. A search bar is at the top left. The main area shows a table with columns Name, Type, and Status. A message says 'There are no secrets available.'

- Enter the correct information and click "Create"
  - Select "Manual" under "Upload options"
  - Use "DscConfigAdmin" as the Name
  - Enter the password of the DSC account in your Microsoft 365 tenant

The screenshot shows the 'Create a secret' form. It has fields for 'Upload options' (set to 'Manual'), 'Name' (set to 'DscConfigAdmin'), and 'Value' (containing a masked password). There are optional fields for 'Content type (optional)', 'Set activation date?', 'Set expiration date?', and 'Enabled?' (set to 'Yes'). At the bottom is a large blue 'Create' button.

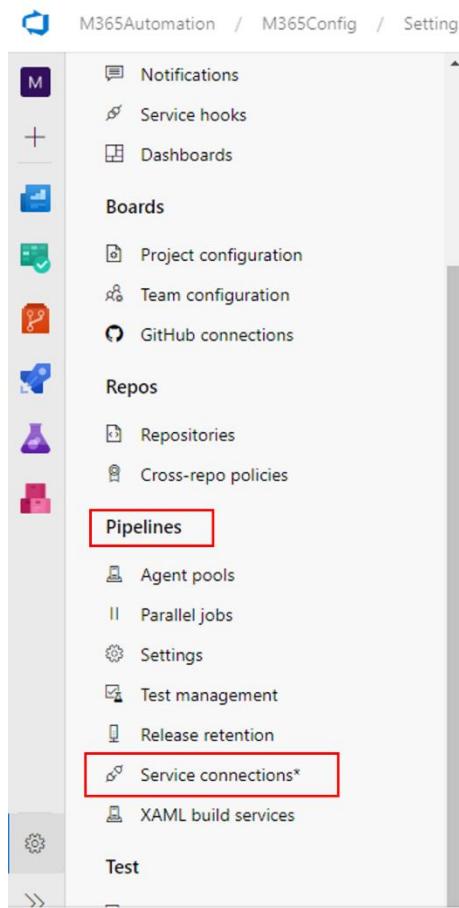
### 3.6.4 Adding Service Connection to the Azure DevOps project

- Open the Azure DevOps Portal
- Browse to your project

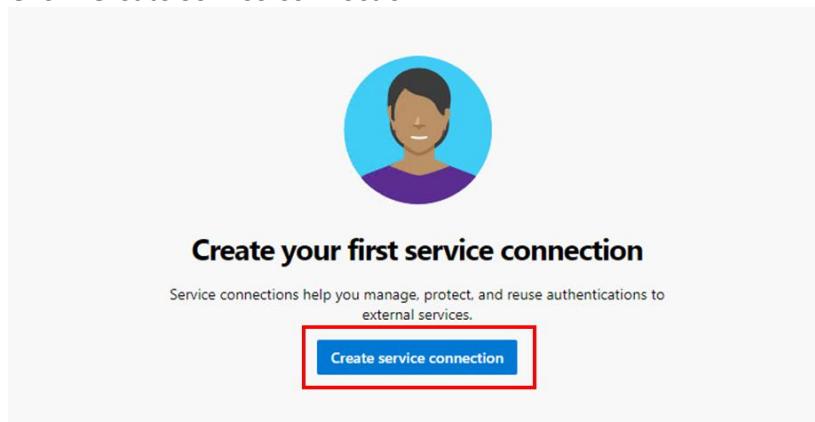


## Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps

- Click "Project Settings" in the lower left corner
- Scroll to the "Pipelines" section and select "Service connections\*\*"

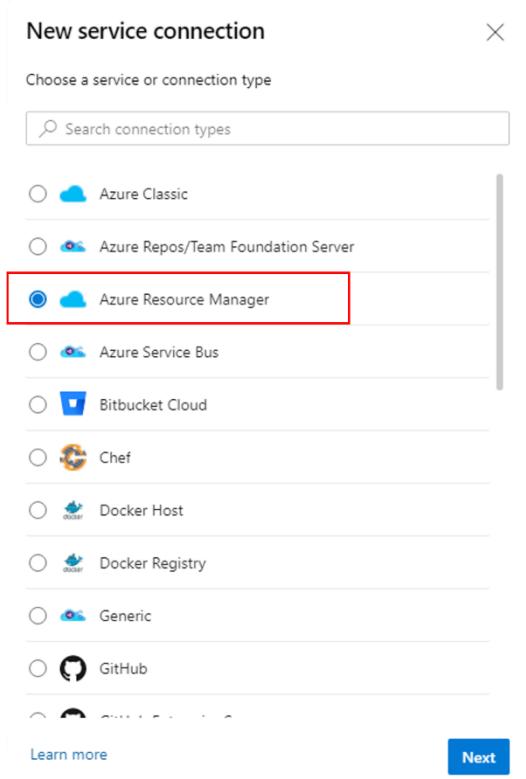


- Click "Create service connection"

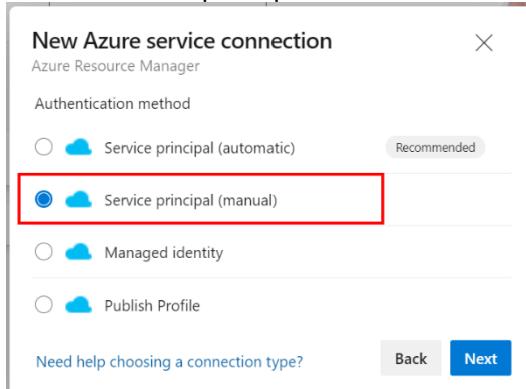


## Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps

- Select "Azure Resource Manager" and click "Next"



- Select "Service principal (manual)" and click "Next"



NOTE: We are using the already created Service Principal Name

- Enter the information you stored when creating the Service Principal Name
  - Enter the GUID of the subscription in which the KeyVault was created as the "Subscription Id"
  - Enter the name of the subscription in which the KeyVault was created as the "Subscription Name"
  - Enter the "appId" as the "Service principal client ID"
  - Enter the "password" as the "Service principal key"
  - Enter the "tenant" as the "Tenant ID" (potentially already populated)
  - Enter a "Service connection name"

## Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps

New Azure service connection

Azure Resource Manager using service principal (manual)

Environment: Azure Cloud

Scope Level: Subscription

Subscription Id: c37373a3-2499-4b46-9504-4e5486e1ff19

Subscription Name: Free Trial

Authentication: Service Principal Id: b92fbf63-7726-47ca-80b7-a0a7c3bffd50

Service principal key: (redacted)

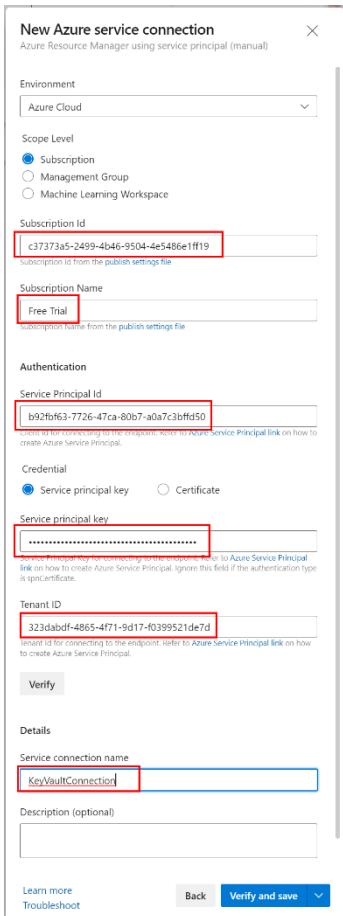
Tenant ID: 323dabdf-4865-4f71-9d17-f0399521de7d

Verify

Details: Service connection name: KeyVaultConnection

Description (optional):

Learn more Troubleshoot Back Verify and save



- Click "Verify" to validate the entered information. The status "Verified" (in green) should appear behind the Verify button.
- Select "Grant access permission to all pipelines" and click "Verify and save"

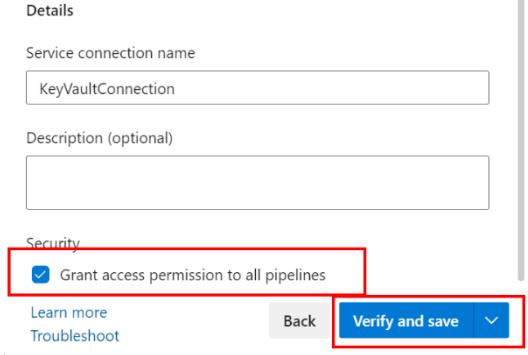
Details

Service connection name: KeyVaultConnection

Description (optional):

Security:  Grant access permission to all pipelines

Learn more Troubleshoot Back Verify and save

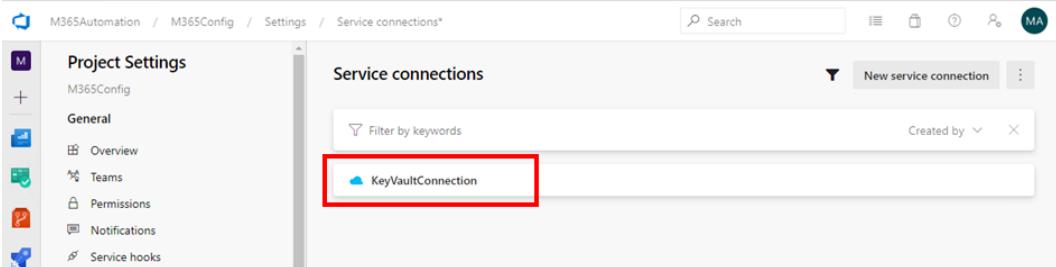


- The "Service connection" is now created and displaying

M365Automation / M365Config / Settings / Service connections\*

Project Settings: General, Overview, Teams, Permissions, Notifications, Service hooks

Service connections: New service connection, Filter by keywords, KeyVaultConnection



### 3.7 Configure the Local Configuration Manager

- Create a Desired State Configuration signing certificate
  - Log onto your virtual machine
  - Open an elevated PowerShell session and run the following command

```
$cert = New-SelfSignedCertificate -Type DocumentEncryptionCertLegacyCsp -DnsName 'DSCNode Document Encryption' -HashAlgorithm SHA256 -NotAfter (Get-Date).AddYears(10)
```

**NOTE:** This will create a self-signed signing certificate for the Local Configuration Manager to use. You can also use a certificate created via a Certificate Authority.

- Run the following command and store the value:

```
$cert.Thumbprint
```

- Export public certificate (required for MOF compilation)
  - Export the public certificate to a CER file by running the following command:

```
Export-Certificate -Cert $cert -FilePath C:\DSCCertificate.cer
```

- Configure the certificate Thumbprint (and the mode eventually and/or other settings) in Local Configuration Manager
  - Log onto your virtual machine
  - Open an elevated PowerShell ISE
  - Browse to a folder M365Dsc (create if it not yet exists)
  - Paste the following code:

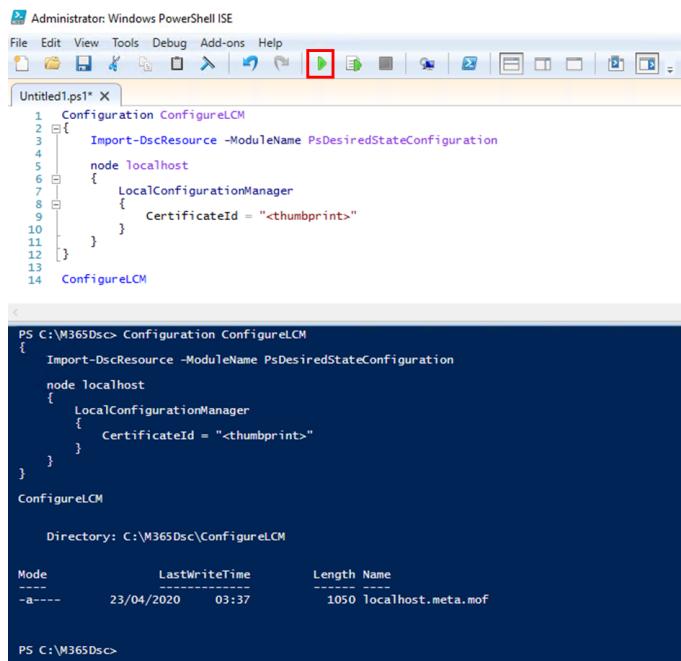
```
Configuration ConfigureLCM
{
    Import-DscResource -ModuleName PsDesiredStateConfiguration

    node localhost
    {
        LocalConfigurationManager
        {
            CertificateId = "<thumbprint>"
        }
    }
}

ConfigureLcm
```

- Update the "<thumbprint>" with your own certificate thumbprint and run the code (F5)

# Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps



```
Configuration ConfigureLCM
{
    Import-DscResource -ModuleName PsDesiredStateConfiguration
    node localhost
    {
        LocalConfigurationManager
        {
            CertificateId = "<thumbprint>"
        }
    }
}
ConfigureLCM
```

```
PS C:\M365Dsc> Configuration ConfigureLCM
{
    Import-DscResource -ModuleName PsDesiredStateConfiguration
    node localhost
    {
        LocalConfigurationManager
        {
            CertificateId = "<thumbprint>"
        }
    }
}
ConfigureLCM

Directory: C:\M365Dsc\ConfigureLCM

Mode                LastWriteTime     Length Name
----                -----          -----   Name
-a----       23/04/2020     03:37           1050 Localhost.meta.mof

PS C:\M365Dsc>
```

- Run the following command to deploy the Local Configuration Manager config:

```
Set-DscLocalConfigurationManager -Path  
C:\M365Dsc\ConfigureLcm -Verbose
```

The output should look like this:

```
PS C:\M365Dsc> Set-DscLocalConfigurationManager -path C:\M365Dsc\ConfigureLCM -Verbose
VERBOSE: Performing the operation "Start-DscConfiguration" on target "MSFT_DSCLocalConfigurationManager".
VERBOSE: Perform operation 'Invoke_CimMethod' with following parameters, "'methodName' = SendMetaConfigurationApply, 'className' = MSFT_DSCLocalConfigurationManager, 'namespaceName' = root/Microsoft/Windows/DesiredStateConfiguration'.
VERBOSE: An LCM method call arrived from computer M365AUTOMATIONV with user sid S-1-5-21-773870280-229285111-404544293-500.
VERBOSE: [M365AUTOMATIONV]: LCM: [ Start Set      ]
VERBOSE: [M365AUTOMATIONV]: LCM: [ Start Resource ] [MSFT_DSCLocalConfiguration]
VERBOSE: [M365AUTOMATIONV]: LCM: [ Start Set      ] [MSFT_DSCLocalConfiguration]
VERBOSE: [M365AUTOMATIONV]: LCM: [ End  Set      ] [MSFT_DSCLocalConfiguration] in 0.0210 seconds.
VERBOSE: [M365AUTOMATIONV]: LCM: [ End  Resource ] [MSFT_DSCLocalConfiguration]
VERBOSE: [M365AUTOMATIONV]: LCM: [ End  Set      ]
VERBOSE: [M365AUTOMATIONV]: LCM: [ End  Set      ] in 0.0610 seconds.
VERBOSE: Operation 'Invoke_CimMethod' complete.
VERBOSE: Set-DscLocalConfigurationManager finished in 0.118 seconds.
```

- To validate a successful configuration of the thumbprint, run Get-DscLocalConfigurationManager

```
PS C:\M365Dsc> Get-DscLocalConfigurationManager

ActionAfterReboot          : ContinueConfiguration
AgentId                    : E338629E-854E-11EA-8B88-00155DFAFE97
AllowModuleOverWrite        :
CertificateID              : 21C1BD64D7294C742A9E2D3834A0DC4742D5C6C9
ConfigurationDownloadManagers : {}
ConfigurationID             :
ConfigurationMode           : ApplyAndMonitor
ConfigurationModeFrequencyMins : 15
Credential                 :
DebugMode                  :
DownloadManagerCustomData  :
DownloadManagerName         :
LCMCompatibleVersions       : {1.0, 2.0}
LCMState                   : Idle
LCMStateDetail              :
LCMVersion                 : 2.0
StatusRetentionTimeInDays   : 10
SignatureValidationPolicy   : NONE
SignatureValidations        : {}
MaximumDownloadSizeMB       : 500
PartialConfigurations       :
RebootNodeIfNeeded          : False
RefreshFrequencyMins        : 30
RefreshMode                 : PUSH
ReportManagers              : {}
ResourceModuleManagers       : {}
PSCoputerName               :
```

## Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps

- Optional: Secure your certificate
  - Export the certificate to PFX format
  - Delete the certificate from the certificate store
  - Reimport the certificate from the PFX file and do not select the option to make the private key exportable
  - Import the PFX file into Azure KeyVault for secure backup

## 4 Configuring Azure DevOps

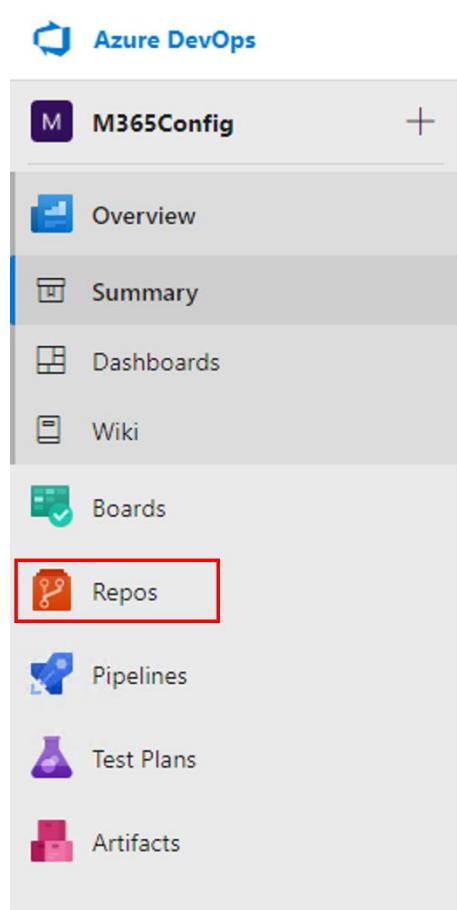
### 4.1 Populate scripts

- Download and install Visual Studio Code from <https://code.visualstudio.com>
- Download and install Git from <https://git-scm.com>
  - Download the most recent version of Git by clicking the "Download" button



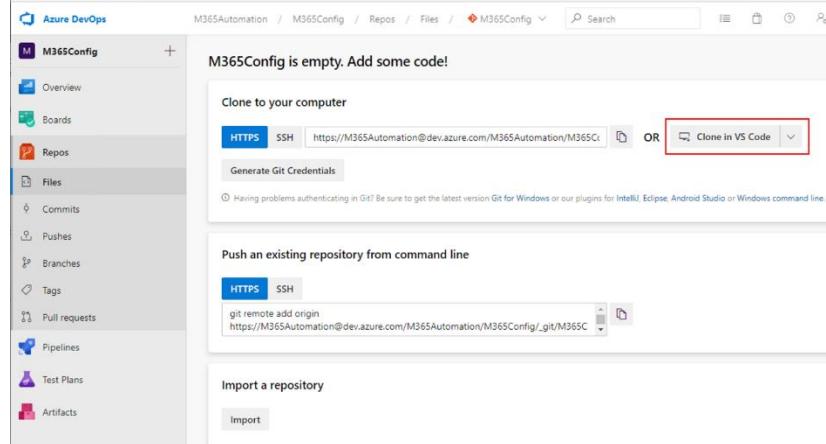
- Run the downloaded installer and use the default settings
- Download the DSC scripts from <https://microsoft365dsc.com/Pages/Resources/Whitepapers/M365Automation.zip>
  - This package contains several scripts, check chapter 6 "Script details" for more details.
- Upload the scripts to the DevOps repository
  - Open Azure DevOps Portal and browse to your project
  - Click the "Repos" icon in the left menu

## Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps

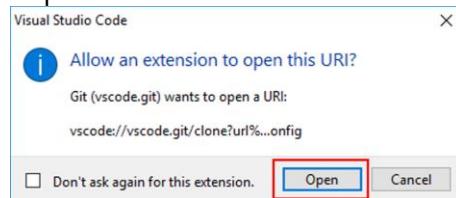


## Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps

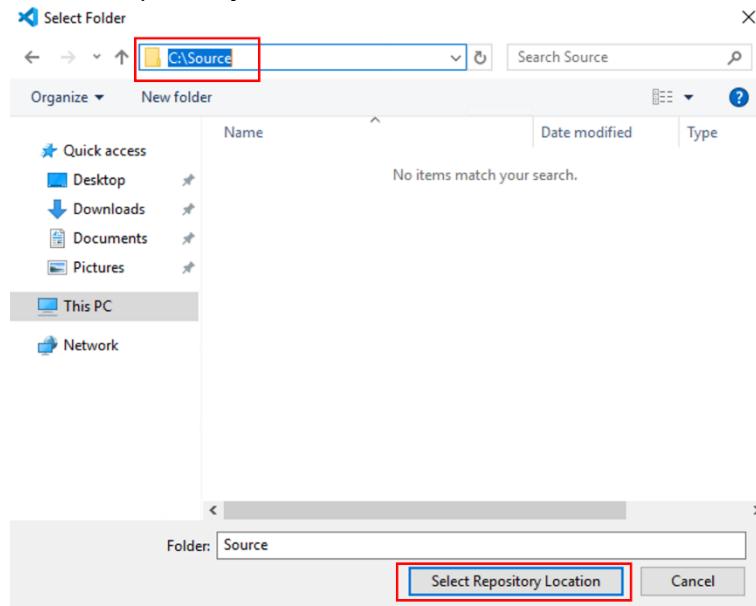
- Click on "Clone in VS Code" (acknowledge any browser notifications for opening any files)



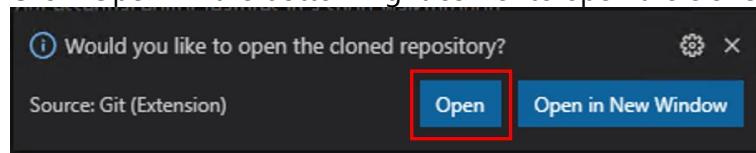
- Acknowledge that Visual Studio Code can open the external URL by clicking "Open"



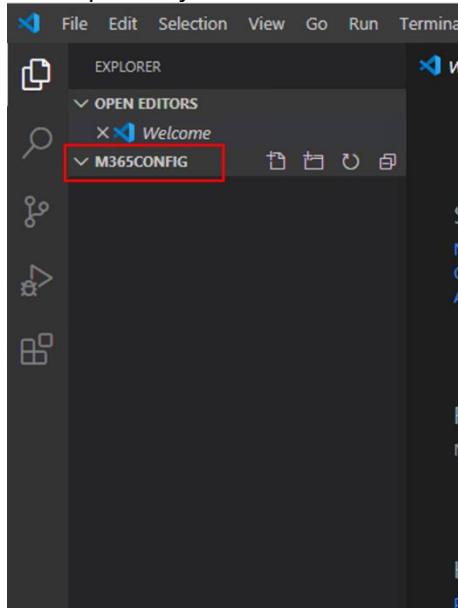
- Select "C:\Source" as the source folder (create if does not exist) and select "Select Repository Location"



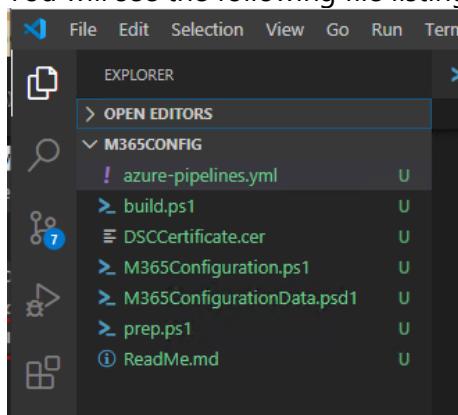
- Login with your Microsoft 365 admin account
- Click "Open" in the bottom right corner to open the cloned folder



- The repository is now available (but still empty) in Visual Studio Code



- Open Explorer and browse to your C:\Source\<project> folder
- Copy the script files from the script download package to this folder
- Copy the DSCCertificate.cer file which you created in paragraph 3.7 to the folder
- You will see the following file listing:



- Open the file "M365ConfigurationData.psd1" and update the domain names used in your environment.

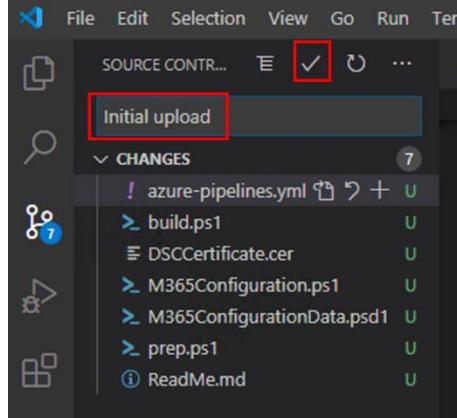
```
NonNodeData = @{
    HRSiteCol = @{
        Url      = 'https://M365Automation.sharepoint.com/sites/hr'
        Owner   = 'admin@M365Automation.onmicrosoft.com'
    }
}
```

- Open the file "build.ps1" and update account name to the account used in your environment.

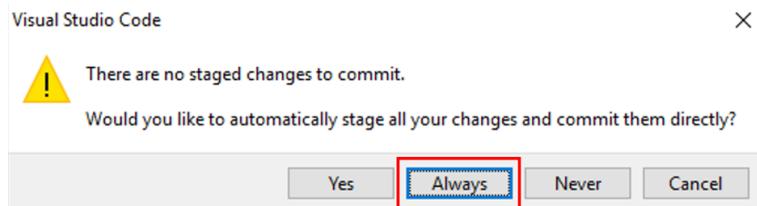
## Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps

```
$password = ConvertTo-SecureString -String $env:SAPW -AsPlainText -Force  
$username = "admin@M365Automation.onmicrosoft.com"  
$cred = New-Object System.Management.Automation.PSCredential($username, $password)
```

- Click on the Git Source Control icon in the left menu, type a commit message (e.g. "Initial upload") and click the checkmark icon



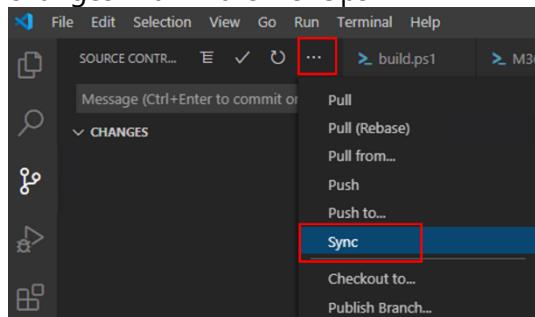
- Click "Always" if you get the message that there are no staged changes to commit.



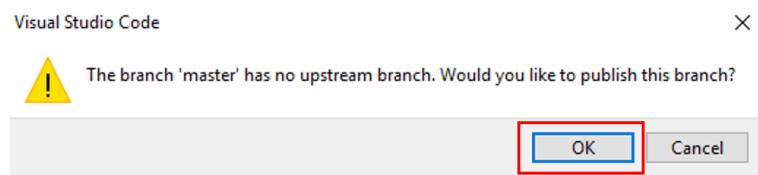
- If you get an error about an unknown e-mail address, run the following commands with your own information and retry the commit:

```
PS C:\Source\VM365Config> git config --global user.email "DscConfigAdmin@M365Automation.onmicrosoft.com"  
PS C:\Source\VM365Config> git config --global user.name "Dsc Config Admin"
```

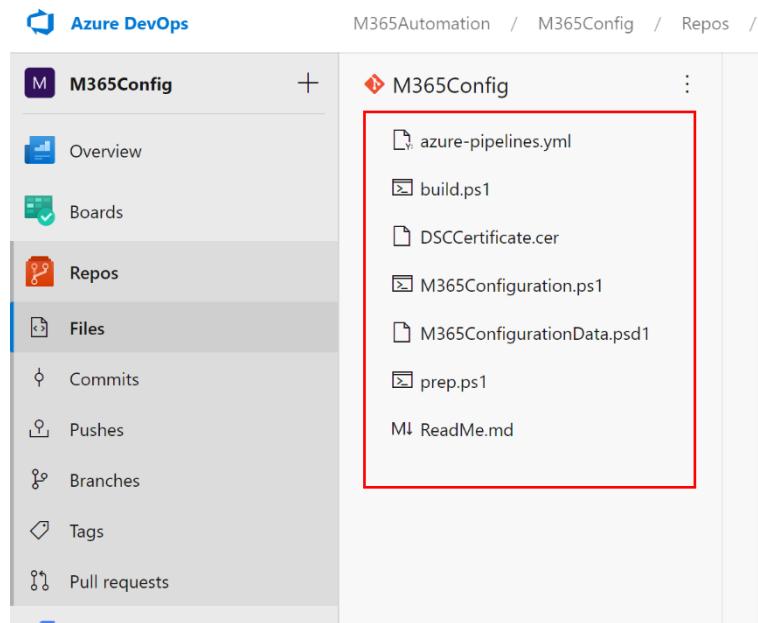
- Click the three dots icon and select "Push, Pull > Sync" to sync your local changes with Azure DevOps



- You might get the below message when running the Sync. Click "OK" to publish the branch to DevOps



- Validate a successful sync by opening the Azure DevOps Portal, browsing to Repos and validating that all files have been uploaded

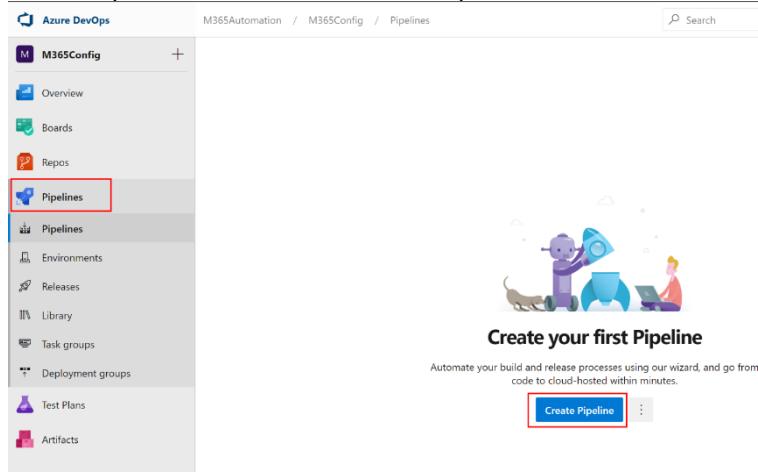


The screenshot shows the Azure DevOps Portal interface. On the left, there's a sidebar with 'M365Config' selected. Under 'Repos', 'Files' is highlighted. The main area shows a list of files: 'azure-pipelines.yml', 'build.ps1', 'DSCCertificate.cer', 'M365Configuration.ps1', 'M365ConfigurationData.psd1', 'prep.ps1', and 'ReadMe.md'. The 'ReadMe.md' file is enclosed in a red box.

## 4.2 Configure Azure DevOps project

### 4.2.1 Create Build pipeline

- Go to the Azure DevOps Portal
- Click "Pipeline" and click "Create Pipeline"



The screenshot shows the Azure DevOps Portal with 'M365Config' selected. In the sidebar, 'Pipelines' is highlighted with a red box. The main area features a 'Create your first Pipeline' wizard with a cartoon character and the text 'Automate your build and release processes using our wizard, and go from code to cloud-hosted within minutes.' A 'Create Pipeline' button is also visible.

## Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps

- Select "Azure Repos Git"

New pipeline  
**Where is your code?**

- Azure Repos Git YAML  
Free private Git repositories, pull requests, and code search
- Bitbucket Cloud YAML  
Hosted by Atlassian
- GitHub YAML  
Home to the world's largest community of developers
- GitHub Enterprise Server YAML  
The self-hosted version of GitHub Enterprise
- Other Git  
Any generic Git repository
- Subversion  
Centralized version control by Apache

Use the classic editor to create a pipeline without YAML.

- Click the name of your Project

New pipeline  
**Select a repository**

Filter by keywords

- M365Config

- Select the "Existing Azure Pipelines YAML file"

New pipeline  
**Configure your pipeline**

- Starter pipeline  
Start with a minimal pipeline that you can customize to build and deploy your code.
- Existing Azure Pipelines YAML file  
Select an Azure Pipelines YAML file in any branch of the repository.

Show more

- Select the file "azure-pipelines.yml" and click "Continue"

Select an existing YAML file X

Select an Azure Pipelines YAML file in any branch of the repository.

Branch master

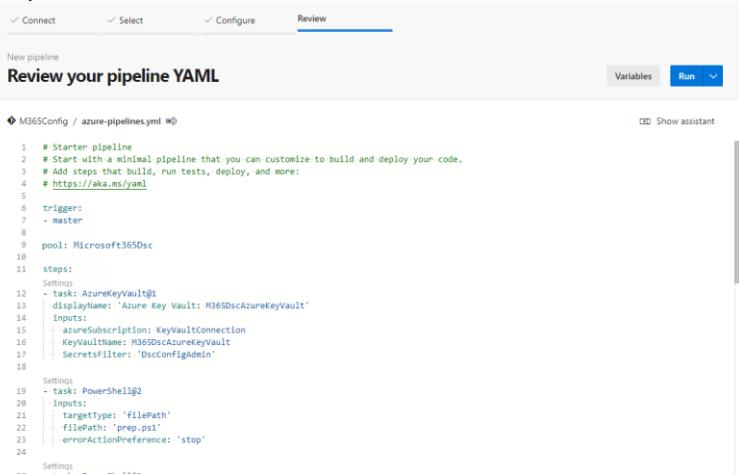
Path /azure-pipelines.yml

- /azure-pipelines.yml /azure-pipelines.yml
- /azure-pipelines\_cert.yml

Cancel Continue

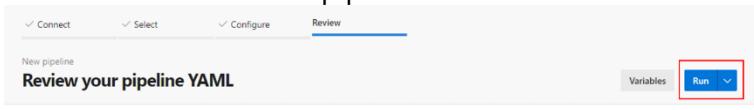
## Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps

- The pipeline then shows you the azure-pipelines.yml file you uploaded in a previous step

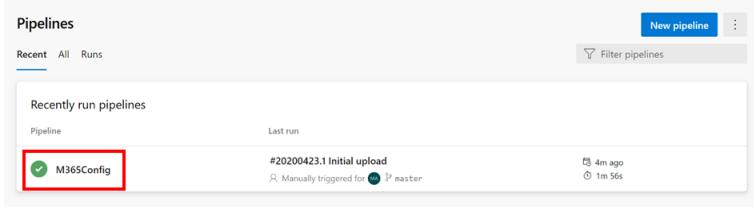


```
M365Config / azure-pipelines.yml #0
1 # Start a pipeline
2 # Start with a minimal pipeline that you can customize to build and deploy your code.
3 # Add steps that build, run tests, deploy, and more:
4 # https://aka.ms/yaml
5
6 trigger:
7 - master
8
9 pool: Microsoft365Dsc
10
11 steps:
12   - task: AzureKeyVault@1
13     displayName: 'Azure Key Vault: M365DscAzureKeyVault'
14     inputs:
15       azureSubscription: KeyVaultConnection
16       KeyVaultName: M365DscAzureKeyVault
17       SecretFilter: 'DscConfigAdmin'
18
19   - task: PowerShell@2
20     inputs:
21       targetType: 'filePath'
22       filePath: 'prep.ps1'
23       errorActionPreference: 'stop'
24
25   - task: AzureKeyVault@1
26     displayName: 'Azure Key Vault: M365DscAzureKeyVault'
27     inputs:
28       azureSubscription: KeyVaultConnection
29       KeyVaultName: M365DscAzureKeyVault
30       SecretFilter: 'DscConfigAdmin'
```

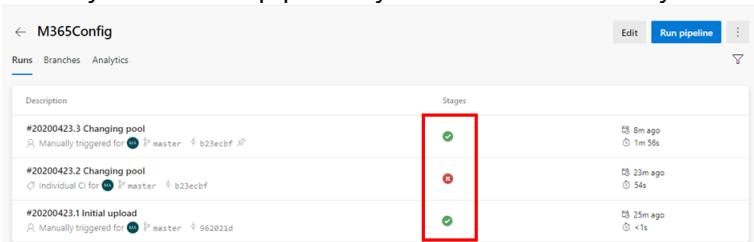
- Select "Run" to start the pipeline



- Check if the pipeline completes successfully



- When you click the pipeline, you can see the history of all runs

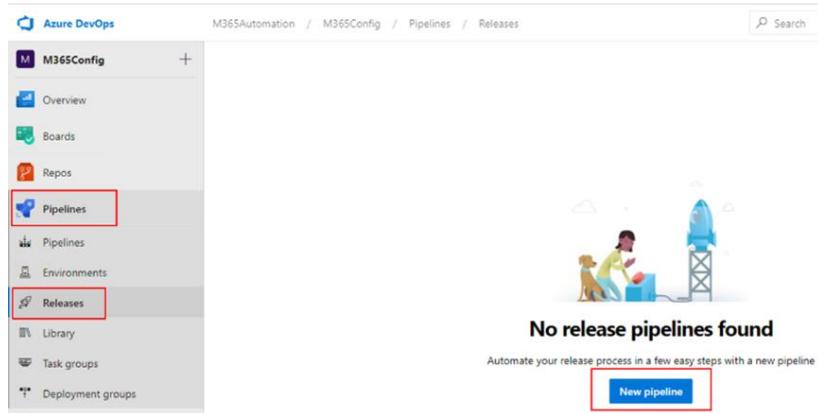


- When you click a run, you can see the logging and other details.

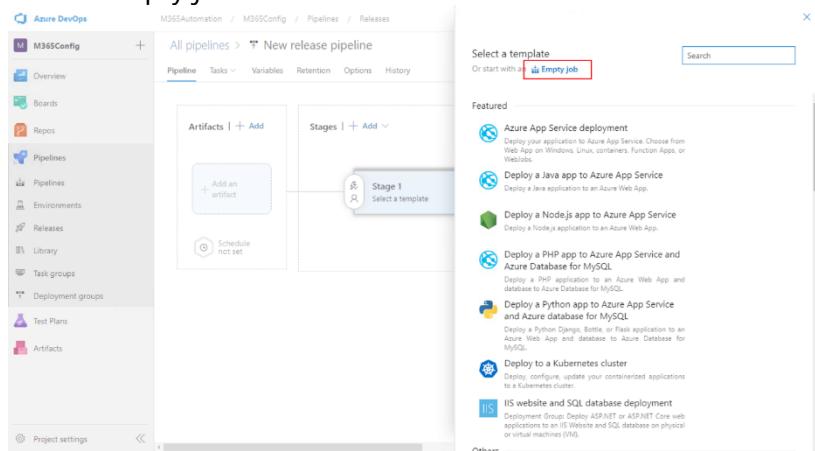
### 4.2.2 Create Release pipeline

- Go to the Azure DevOps Portal
- Click "Pipelines", click "Releases" and then click "New pipeline"

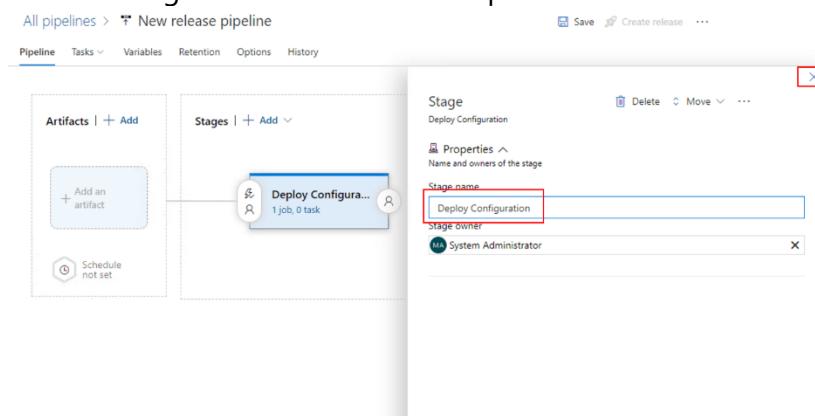
## Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps



- Select "Empty job"



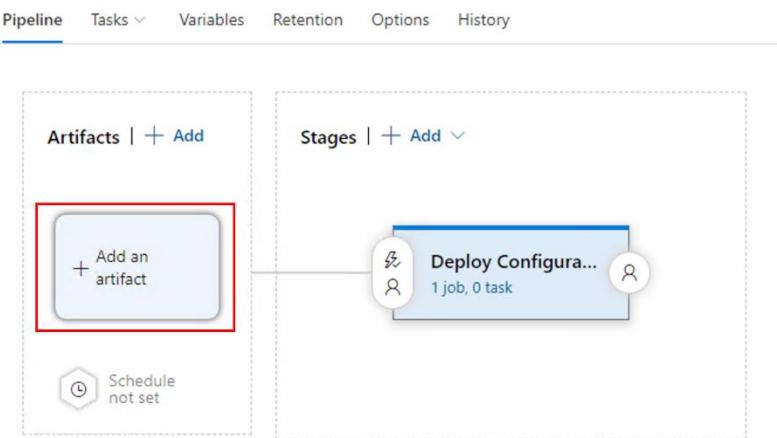
- Give the Stage a name and close the pane



## Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps

- Click "Add an artifact"

All pipelines > New release pipeline



- Under "Source" select the build pipeline

The screenshot shows the 'Add an artifact' configuration dialog. Under 'Source type', the 'Build' option is selected and highlighted with a blue border. Below it, there are other options: 'Azure Repos ...', 'GitHub', and 'TFVC'. A link '5 more artifact types' is visible. Under 'Project \*', the value 'M365Config' is selected. Under 'Source (build pipeline) \*', the value 'M365Config' is listed in a dropdown menu, which is also highlighted with a red box. An 'Add' button is at the bottom.

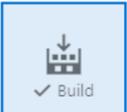
- After selecting the Source, more options will appear. Leave them default and click "Add".

**NOTE:** Notice the "Source alias" value. We need this value in a next step.

## Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps

Add an artifact

Source type

Build  Azure Repos ... GitHub TFVC

5 more artifact types ▾

Project \* ⓘ M365Config

Source (build pipeline) \* ⓘ M365Config

Default version \* ⓘ Latest

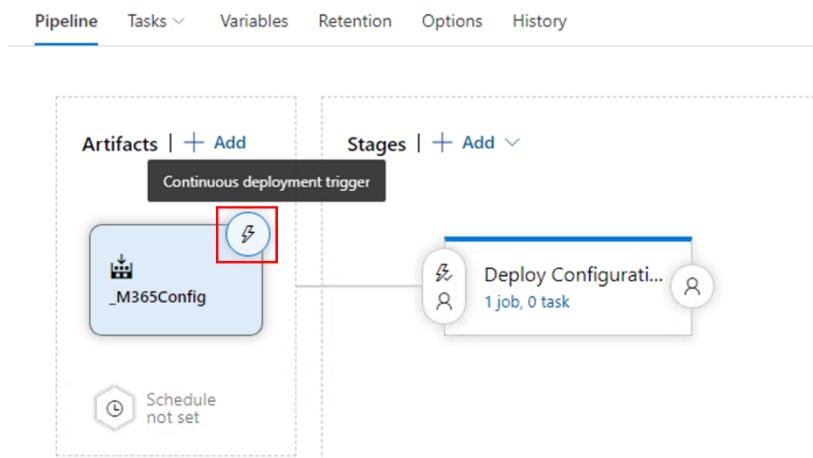
Source alias \* ⓘ \_M365Config

ⓘ The artifacts published by each version will be available for deployment in release pipelines. The latest successful build of M365Config published the following artifacts: M365ConfigMOFFile.

**Add**

- Configure the Release pipeline triggers by clicking the Lightning icon next to Artifacts

All pipelines >  New release pipeline



## Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps

- Enable the "Continuous deployment trigger", under "Build branch filters" click "Add" and select "The build pipeline's default branch"

Continuous deployment trigger

Build: \_M365Config

Enabled

Creates a release every time a new build is available.

Build branch filters ⓘ

No filters added.

+ Add | ↴

Branch filter

The build pipeline's default branch

Disabled

ⓘ Enabling this will create a release every time a selected artifact is available as part of a pull request workflow

- Make sure the branch has been added successfully and close the pane

Continuous deployment trigger

Build: \_M365Config

Enabled

Creates a release every time a new build is available.

Build branch filters ⓘ

Type	Build branch	Build tags
Include	↳ The build pipeline's default bra...	<input type="button" value="Delete"/>

+ Add | ↴

Pull request trigger

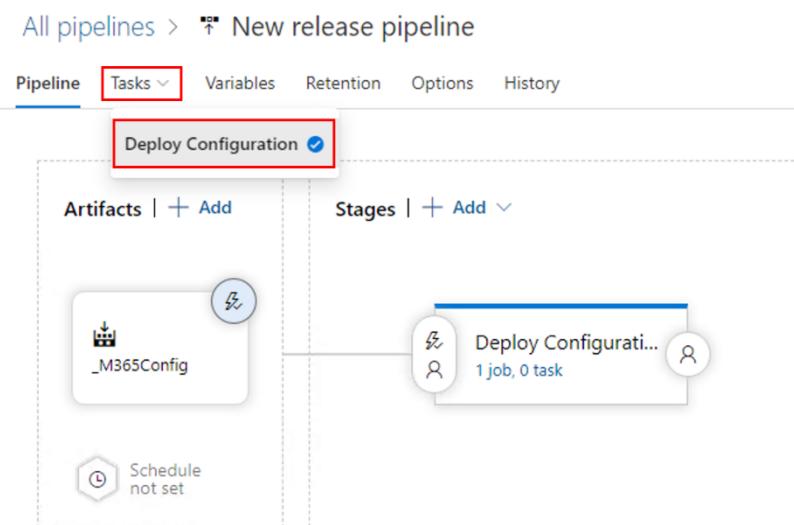
Build: \_M365Config

Disabled

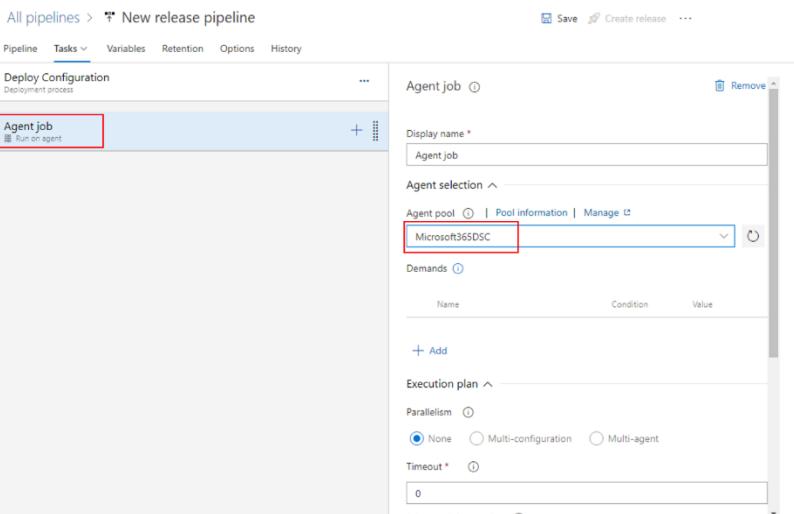
ⓘ Enabling this will create a release every time a selected artifact is available as part of a pull request workflow

## Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps

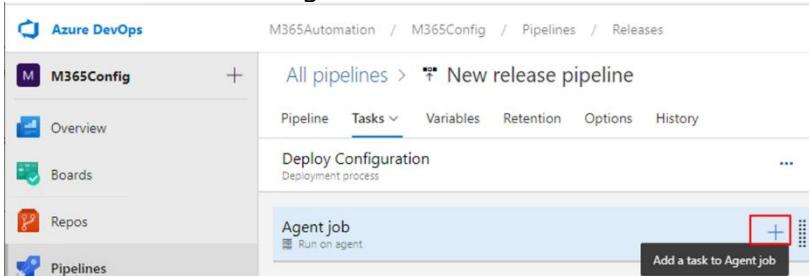
- Select "Tasks > <Stage name>"



- Select the task "Agent job" in the left part of the pane and change the "Agent pool" to "Microsoft365Dsc". Leave the rest default.

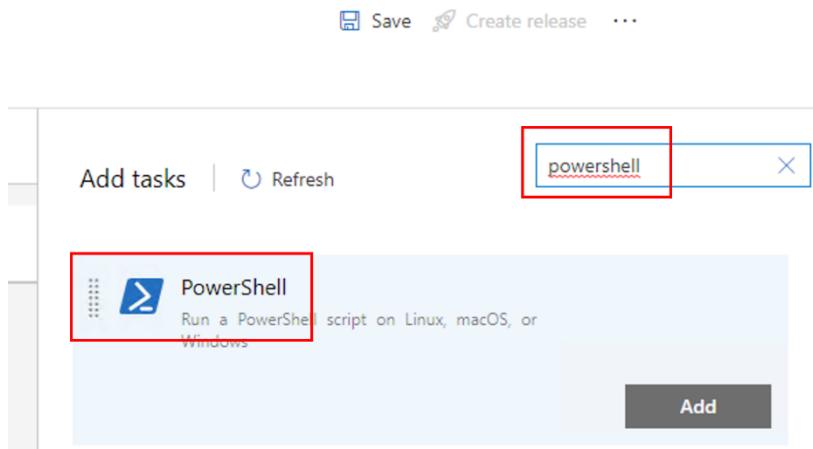


- Click the "+" behind "Agent Job"

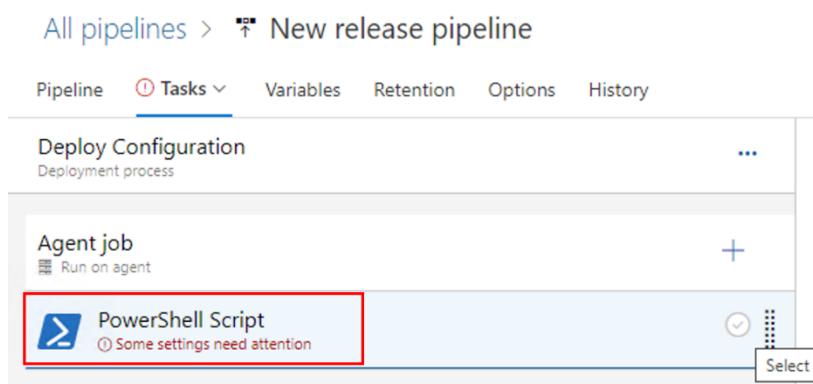


## Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps

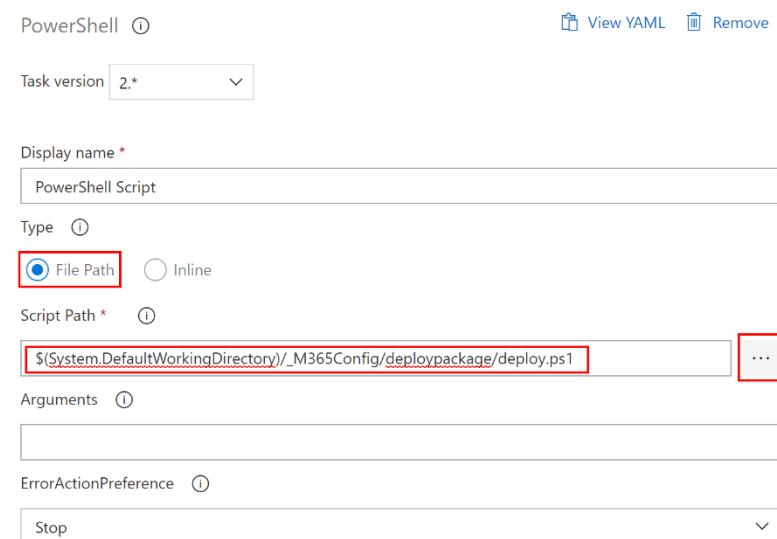
- Search for "PowerShell" and select "PowerShell".  
**Note:** Do not select the "PowerShell on Target Machines" task



- Select the "PowerShell" task

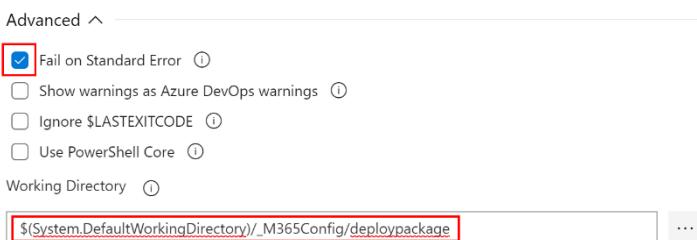


- Select "File Path" as "Type" and browse to the deploy.ps1 file by clicking the "..." button

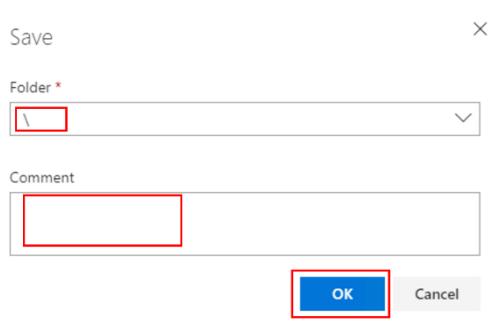


## Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps

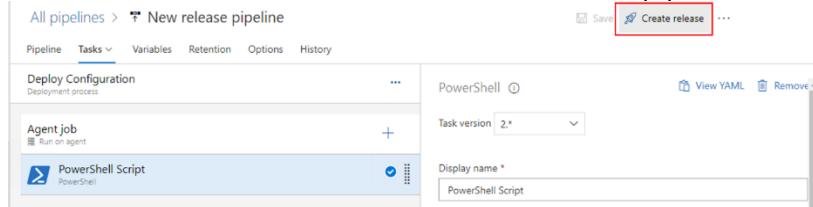
- Open the "Advanced" section and select "Fail on Standard Error"



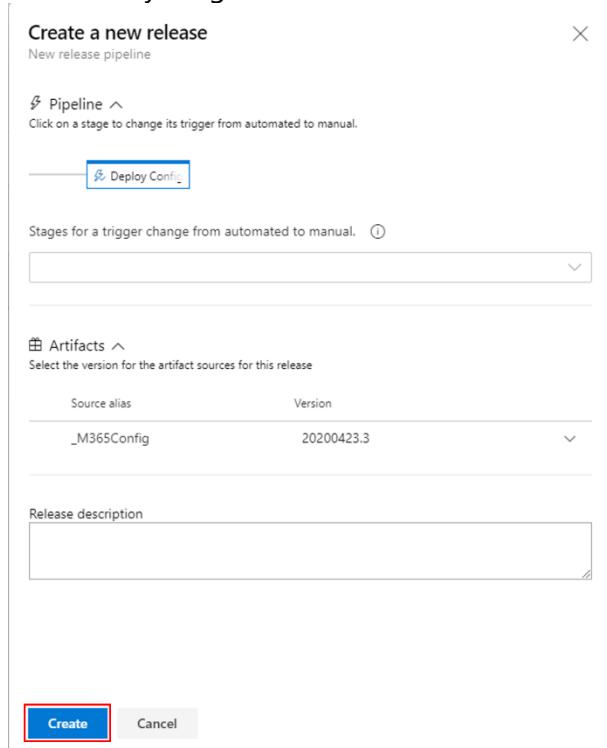
- Click "Save". Use "\\" as the folder and add a comment if you prefer. Click "OK"



- Click "Create release" to test the created Release pipeline.



- Leave everything default and click "Create"



## Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps

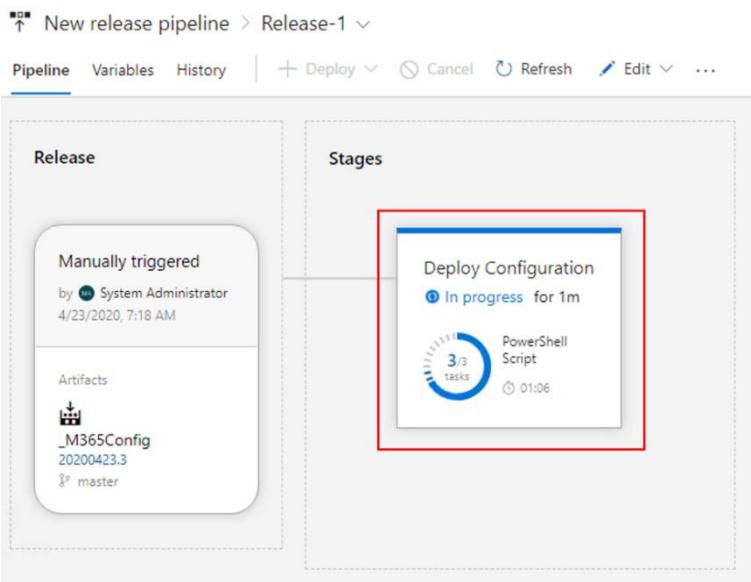
- Click "Release-<nr>" in the top bar to open the release and review its progress

All pipelines >  New release pipeline

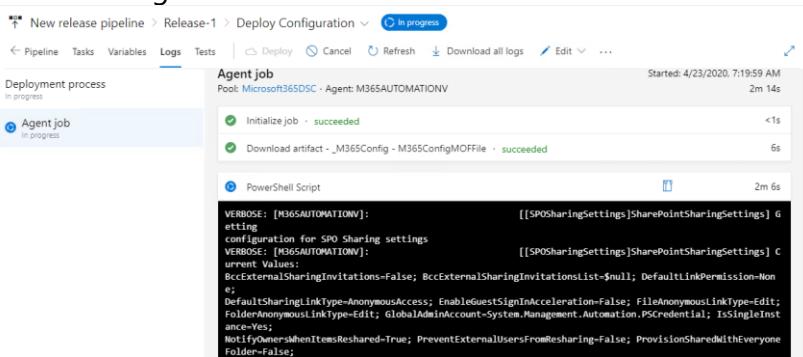


- Review the progress

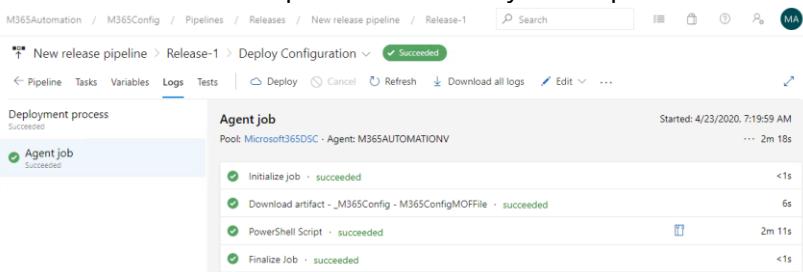
M365Automation / M365Config / Pipelines / Releases / New release pipeline / Release-1



- Click the stage for more details



- When the release completes successfully, all steps should have green check marks.



## Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps

### 4.2.3 Validate that changes to the config are deployed successfully

- Make sure the following setting is configured:  
SharePoint Admin Center > Policies > File and folder links > Only people in your organization

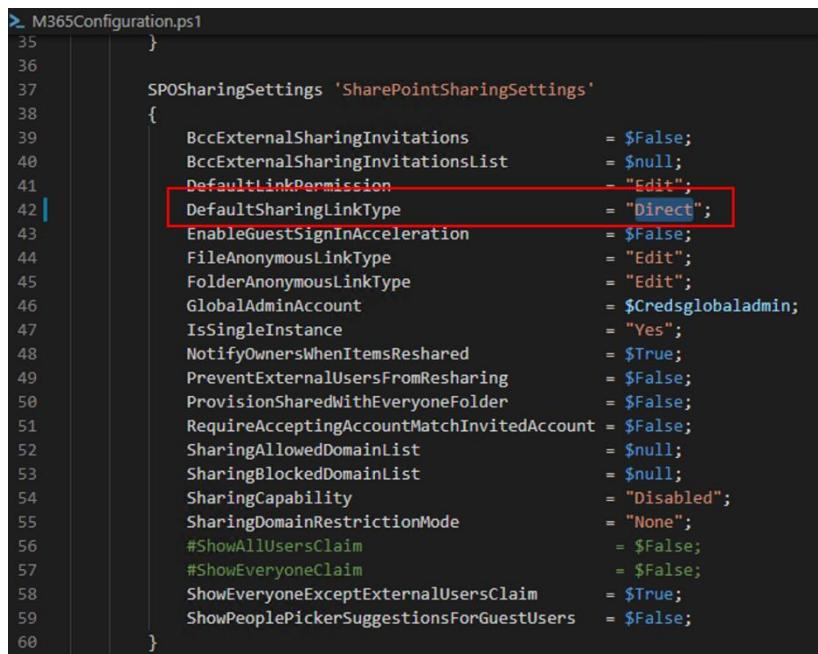
The screenshot shows the SharePoint Admin Center interface under the 'Policies' section. In the 'File and folder links' configuration, the 'DefaultSharingLinkType' is explicitly set to 'Only people in your organization', which is highlighted with a red box. Other options like 'View' and 'Edit' are also listed.

- The above setting is configured by the "DefaultSharingLinkType" DSC setting:

```
M365Configuration.ps1
35     }
36
37     SPOSharingSettings 'SharePointSharingSettings'
38     {
39         BccExternalSharingInvitations      = $False;
40         BccExternalSharingInvitationsList = $null;
41         DefaultLinkPermission            = "Edit";
42         DefaultSharingLinkType           = "Internal"; -----
43         EnableGuestSignInAcceleration    = $False;
44         FileAnonymousLinkType          = "Edit";
45         FolderAnonymousLinkType        = "Edit";
46         GlobalAdminAccount             = $Credsglobaladmin;
47         IsSingleInstance                = "Yes";
48         NotifyOwnersWhenItemsReshared   = $True;
49         PreventExternalUsersFromResharing = $False;
50         ProvisionSharedWithEveryoneFolder = $False;
51         RequireAcceptingAccountMatchInvitedAccount = $False;
52         SharingAllowedDomainList       = $null;
53         SharingBlockedDomainList      = $null;
54         SharingCapability              = "Disabled";
55         SharingDomainRestrictionMode   = "None";
56         #ShowAllUsersClaim             = $False;
57         #ShowEveryoneClaim             = $False;
58         ShowEveryoneExceptExternalUsersClaim = $True;
59         ShowPeoplePickerSuggestionsForGuestUsers = $False;
```

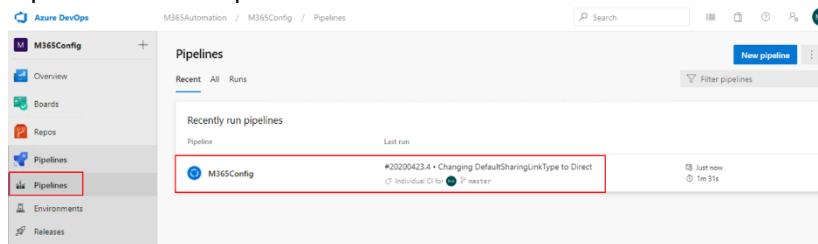
- Change this setting from "Internal" to "Direct"

## Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps

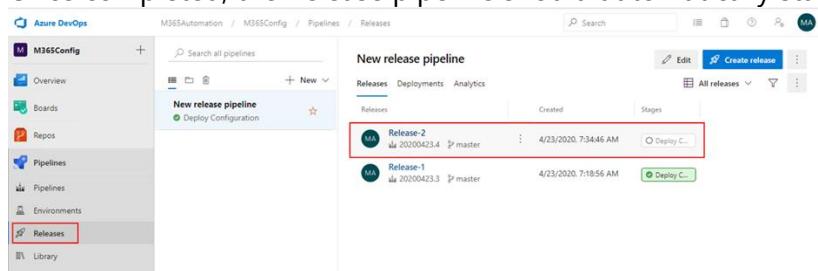


```
M365Configuration.ps1
35     }
36
37     SPOSharingSettings 'SharePointSharingSettings'
38     {
39         BccExternalSharingInvitations           = $False;
40         BccExternalSharingInvitationsList      = $null;
41         DefaultLinkPermission                 = "Edit";
42         DefaultSharingLinkType                = "Direct"; DefaultSharingLinkType
43         EnableGuestSignInAcceleration          = $False;
44         FileAnonymousLinkType                 = "Edit";
45         FolderAnonymousLinkType               = "Edit";
46         GlobalAdminAccount                   = $Credsglobaladmin;
47         IsSingleInstance                     = "Yes";
48         NotifyOwnersWhenItemsReshared        = $True;
49         PreventExternalUsersFromResharing    = $False;
50         ProvisionSharedWithEveryoneFolder   = $False;
51         RequireAcceptingAccountMatchInvitedAccount = $False;
52         SharingAllowedDomainList            = $null;
53         SharingBlockedDomainList           = $null;
54         SharingCapability                  = "Disabled";
55         SharingDomainRestrictionMode       = "None";
56         #ShowAllUsersClaim                 = $False;
57         #ShowEveryoneClaim                 = $False;
58         ShowEveryoneExceptExternalUsersClaim = $True;
59         ShowPeoplePickerSuggestionsForGuestUsers = $False;
60     }
```

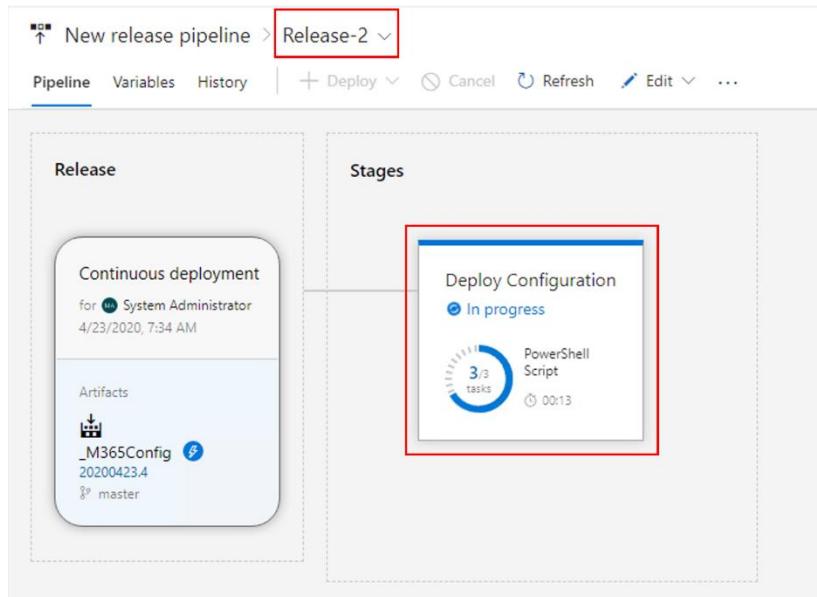
- Save the file, go to the Git Source Control section, enter a commit description, commit the change and sync the repository with Azure DevOps
- Open the Build Pipeline, which should have started



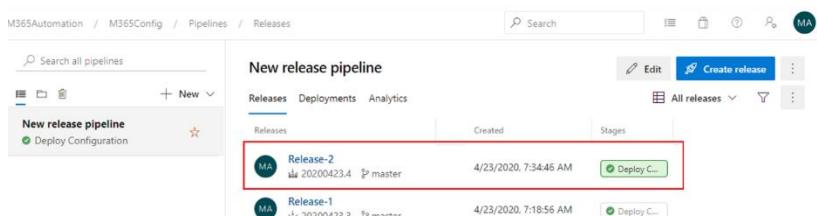
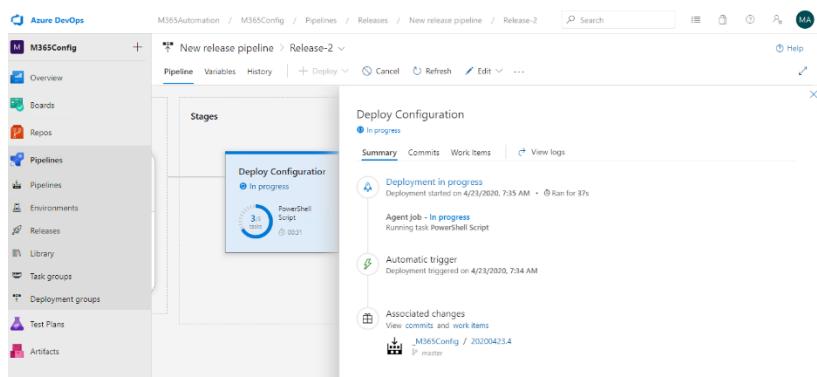
- Once completed, the Release pipeline should automatically start



## Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps



- When the Release pipeline completes, the setting should have been changed in the SharePoint Admin Portal



## Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps

The screenshot shows the SharePoint admin center interface. On the left, there's a navigation menu with options like Home, Sites, Policies (which is selected), Settings, Migration, More features, OneDrive admin center, Customize navigation, and Show all. The main content area is titled "File and folder links". It asks to choose the type of link selected by default when users share files and folders. There are three options: "Specific people (only the people the user specifies)" (selected and highlighted with a red box), "Only people in your organization", and "Anyone with the link". Below that, it asks to choose the permission selected by default for sharing links, with "Edit" selected (radio button is checked).

## 5 Security Enhancements

### 5.1 Using Azure Conditional Access to secure service account

Azure Conditional Access<sup>1</sup> can be used to prevent the created service account login into Microsoft 365, except when coming from a specified location / IP address. This section describes the steps to implement this restriction.

Requirements:

- All VMs have a fixed IP address configured
- List of the IP address of all the VMs
- Name of DSC service account created in paragraph 3.1, e.g., "DscConfigAdmin"

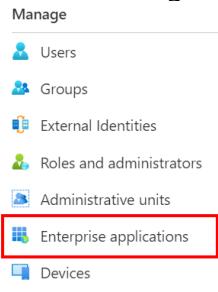
Steps

- Open the Azure Portal (<https://portal.azure.com>)
- Go to Azure Active Directory

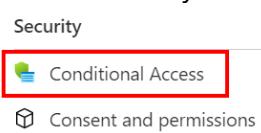


Azure Active  
Directory

- Under "Manage", click "Enterprise applications"

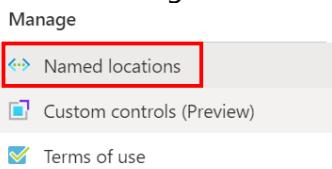


- Under "Security", click "Conditional Access"



- First, we are going to create a Named Location

- Under "Manage", click "Named locations"



<sup>1</sup> Azure AD Premium P1 license required

## Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps

- Click "New location"

[+ New location](#)

- Enter the required information:

- Name: "Azure Self Hosted VMs" (or any other name you want to use)
  - Define the location using: "IP Ranges"
  - IP ranges: The public IP address of the VM in the "123.123.123.123/32" format

Home > Contoso > Enterprise applications > Conditional Access >

### New named location

Upload Download

Name \*  
Azure Self Hosted VMs

Define the location using:  
 IP ranges  
 Countries/Regions

Mark as trusted location

IP ranges  
123.123.123.123/32

- Click "Create" to create the Named location
- Next, select "Policies" and click "New policy"

Home > Contoso > Enterprise applications >

### Conditional Access | Policies

Azure Active Directory

Policies

+ New policy

What If

Refresh

Got feedback?

Insights and reporting

Diagnose and solve problems

Policy Name

- Create a new policy, using the following settings:
  - Name: "Conditional Access for DSC Service Account" (or the name you would like to use)
  - Users and groups > Include
    - Select "Selected users and groups"
    - Check "Users and groups"
    - Search and select the DSC Service Account

# Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps

Home > Conditional Access >

## New

Conditional access policy

Control user access based on conditional access policy to bring signals together, to make decisions, and enforce organizational policies. [Learn more](#)

Name \*

Conditional Access for DSC Service Acco... ✓

Assignments

Users and groups ⓘ >  
Specific users included

Cloud apps or actions ⓘ >  
No cloud apps or actions selected

Conditions ⓘ >  
0 conditions selected

Access controls

Grant ⓘ >  
0 controls selected

Session ⓘ >  
0 controls selected

Control user access based on users and groups assignment for all users, specific groups of users, directory roles, or external guest users [Learn more](#)

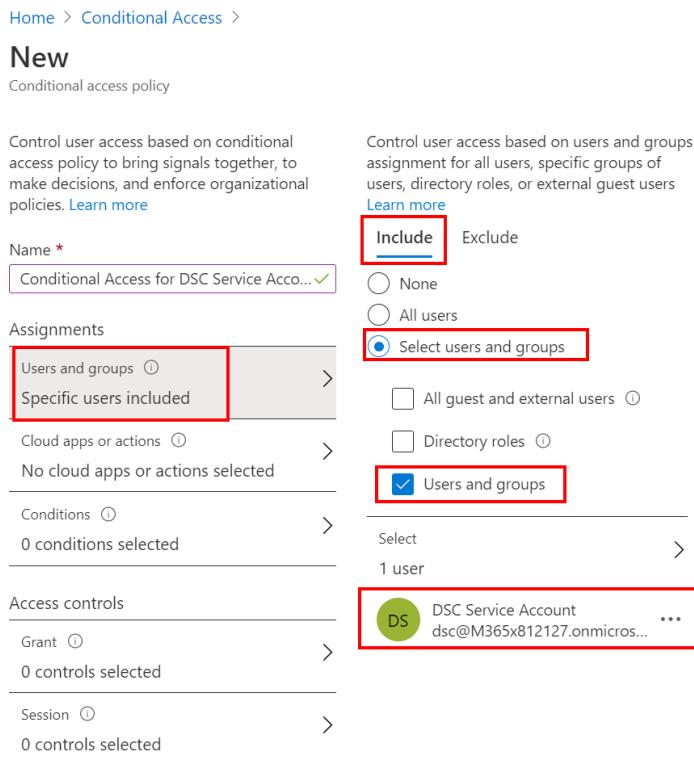
**Include** **Exclude**

None  
 All users  
 Select users and groups

All guest and external users ⓘ  
 Directory roles ⓘ  
 Users and groups

Select  
1 user

DS DSC Service Account  
dsc@M365x812127.onmicrosoft.com ...



- Cloud apps or actions: Select "All cloud apps"

Home > Conditional Access >

## New

Conditional access policy

Control user access based on conditional access policy to bring signals together, to make decisions, and enforce organizational policies. [Learn more](#)

Name \*

Conditional Access for DSC Service Acco... ✓

Assignments

Users and groups ⓘ >  
Specific users included

Cloud apps or actions ⓘ >  
All cloud apps

Conditions ⓘ >  
0 conditions selected

Access controls

Grant ⓘ >

Control user access based on all or specific cloud apps or actions. [Learn more](#)

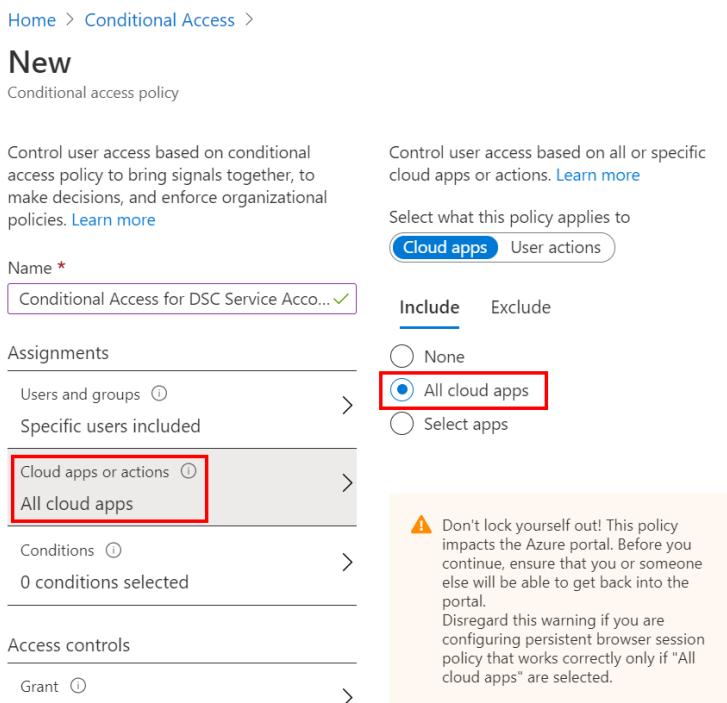
Select what this policy applies to

**Cloud apps** **User actions**

**Include** **Exclude**

None  
 All cloud apps  
 Select apps

**⚠️** Don't lock yourself out! This policy impacts the Azure portal. Before you continue, ensure that you or someone else will be able to get back into the portal.  
Disregard this warning if you are configuring persistent browser session policy that works correctly only if "All cloud apps" are selected.



- Conditions > Locations
  - Include: "Any location"
  - Exclude: Select "Selected locations" and select the newly created Named location "Azure Self Hosted VMs"

# Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps

The screenshot shows the 'Conditional Access' configuration page. In the 'Access controls' section, the 'Grant' tab is selected. Under 'Grant', the 'Block access' option is chosen. A red box highlights the 'Block access' button. Below it, under 'For multiple controls', the 'Require all the selected controls' radio button is selected.

- Access controls > Grant
  - Select "Block access"

The screenshot shows the 'Conditional Access' configuration page. In the 'Access controls' section, the 'Grant' tab is selected. Under 'Grant', the 'Block access' option is chosen. A red box highlights the 'Block access' button. Below it, under 'For multiple controls', the 'Require all the selected controls' radio button is selected.

## Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps

- Under "Enable policy", select "On" to activate the policy and click "Create"

Home > Conditional Access >

### New

Conditional access policy

Control user access based on conditional access policy to bring signals together, to make decisions, and enforce organizational policies. [Learn more](#)

Name \*

Conditional Access for DSC Service Acco... ✓

Assignments

Users and groups >  
Specific users included

Cloud apps or actions >  
All cloud apps

Conditions >  
1 condition selected

Access controls

Grant >  
Block access

Session >  
0 controls selected

Enable policy

Report-only **On** Off

**Create**

- The DSC service account can now only be used to authenticate from the Azure DevOps Self Hosted VMs

## 5.2 Using Certificates instead of Username/Password for authentication

The Microsoft 365 PowerShell modules, used by Microsoft365DSC, are supporting several authentication methods:

- Username / Password
- Certificate stored in the computers certificate store / Thumbprint of certificate
- Certificate stored in PFX file / Password of the file
- Application secret
  - o Only supported by a few modules and therefore not implemented in Microsoft365DSC (yet).

The following table shows the supported authentication method for each Microsoft365DSC workload:

## Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps

Workload	PowerShell Module	Credential	Certificate Thumbprint	Certificate Path	Application Secret*
AzureAD	AzureADPreview (Connect-AzureAD)	✓	✓	✗	✗
Exchange Online	ExchangeOnlineManagement (Connect-ExchangeOnline)	✓	✓	✓	✗
Intune	Microsoft.Graph.Intune (Connect-MSGraph)	✓	✗	✗	✗
Office 365	AzureADPreview (Connect-AzureAD)	✓	✓	✗	✗
OneDrive	SharePointPnPowershellOnline (Connect-PnPOnline)	✓	✓	✓	✗
PowerApps	Microsoft.PowerApps.Administration.PowerShell	✓	✓	✗	✗
Planner	Microsoft.Graph.Authentication (Connect-Graph)	✗	✓	✗	✗
Security & Compliance Center	ExchangeOnlineManagement (Connect-IPPSession)	✓	✗	✗	✗
Skype for Business Online	MicrosoftTeams (New-CSOnlineSession)	✓	✗	✗	✗
SharePoint Online	SharePointPnPowershellOnline (Connect-PnPOnline)	✓	✓	✓	✗
Teams	MicrosoftTeams (New-CSOnlineSession)	✓	✓	✗	✗

\* Application secret is possible with the OneDrive, PowerApps and SharePoint cmdlets, but has not been implemented for Microsoft365DSC yet.

This section describes the required steps to use certificate with thumbprint authentication instead of a username / password.

To automate the deployment of the certificate to the Azure DevOps agent, we are storing the certificate and the password in Azure KeyVault. In the Release pipeline we are checking if the certificate is present and import it if not.

To implement this scenario, follow the regular process as described in chapter 3 "Preparation" and 4 "Configuring Azure DevOps", but replace paragraphs 4.2.1 and 4.2.2 with the below steps.

### 5.2.1 Creating the authentication certificate

- Log onto the virtual machine
- Open an elevated PowerShell window
- Create and export a new certificate by running the following PowerShell commands:
  - **NOTE:** Update the [PASSWORD] parameter to your own password

```
$cert = New-SelfSignedCertificate -Subject "CN=Microsoft365DSC"
-CertStoreLocation "Cert:\LocalMachine\My" -KeyExportPolicy
Exportable -KeySpec Signature

$password = ConvertTo-SecureString -String "[PASSWORD]" -
AsPlainText -Force

Export-PfxCertificate -Cert $cert -FilePath
C:\M365ClientCert.pfx -Password $password

Export-Certificate -Cert $cert -FilePath C:\M365ClientCert.cer
```



For more information on creating a certificate for application authentication, see:  
<https://docs.microsoft.com/en-us/azure/active-directory/develop/howto-create-self-signed-certificate>

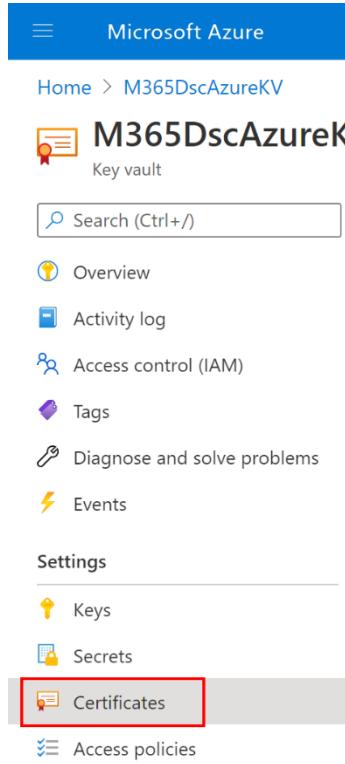
- Run the following command, copy the outputted Thumbprint and store it for later use

```
$cert.Thumbprint
```

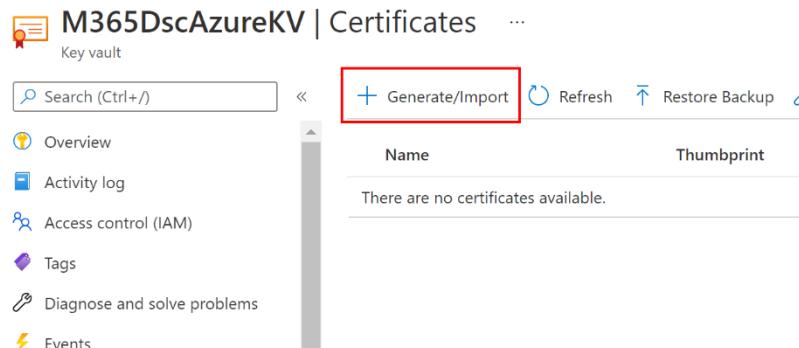
## 5.2.2 Adding certificate to Azure KeyVault

Instead of paragraph 3.6.3 "Add secrets to your Vault", take the following steps to add the certificate to the KeyVault:

- Click "Certificates" in the left menu



- Click "Generate/Import" to create a new secret



- Enter the correct information and click "Create"
  - Select "Import" under "Method of Certificate Creation"

- Use "M365ClientCert" as the "Certificate Name"
- At "Upload Certificate File" click the file browse icon at the right and select the "C:\M365ClientCert.pfx" file
- Enter the password of the certificate you specified while creating the certificate

Create a certificate ...

Method of Certificate Creation

Import

Certificate Name \* ⓘ  
M365ClientCert

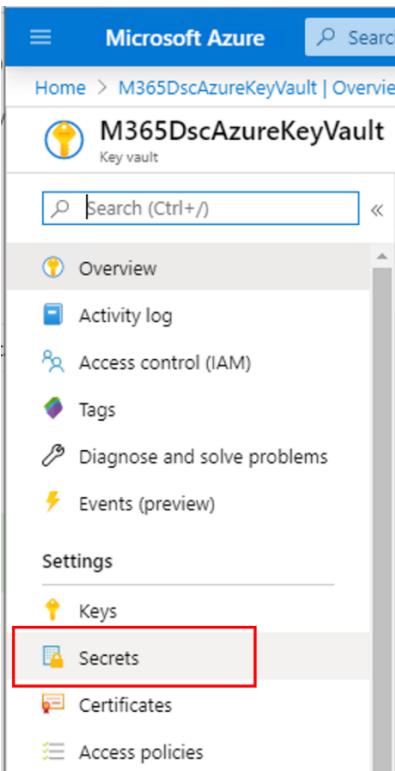
Upload Certificate File \*  
M365ClientCert.pfx

Password  
.....

### 5.2.3 Adding the certificate password to Azure KeyVault

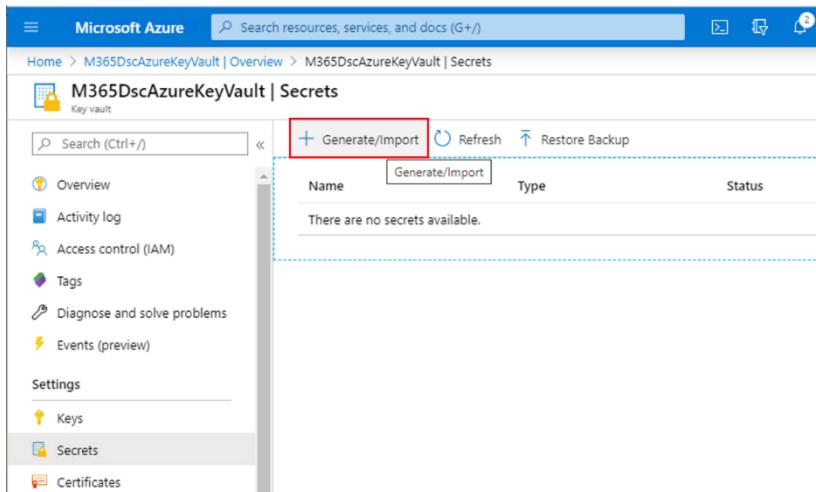
The Release pipeline needs the password of the certificate to import the certificate into the local certificate store. We are storing this password in KeyVault as well.

- Click "Secrets" in the left menu



- Click "Generate/Import" to create a new secret

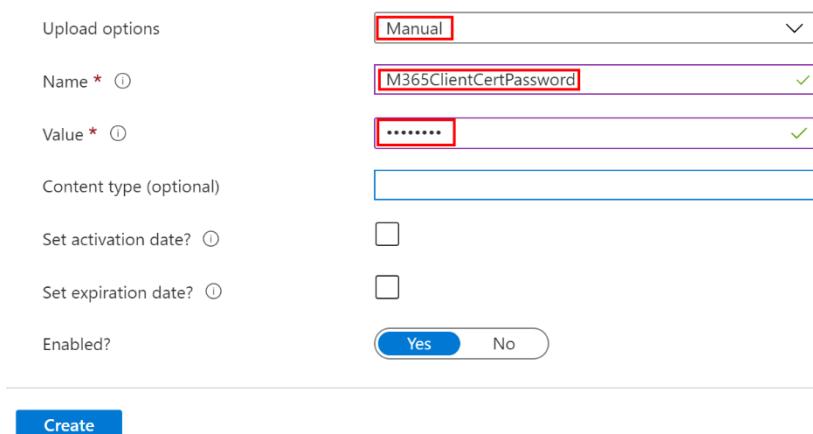
## Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps



The screenshot shows the Microsoft Azure Key Vault Secrets page for the 'M365DscAzureKeyVault' key vault. The left sidebar shows navigation options like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Events (preview), and Settings (Keys, Secrets, Certificates). The main area displays a table with columns Name, Type, and Status. A message says 'There are no secrets available.' At the top right, there are buttons for Generate/Import, Refresh, and Restore Backup. A red box highlights the 'Generate/Import' button.

- Enter the correct information and click "Create"
  - Select "Manual" under "Upload options"
  - Use "M365ClientCertPassword" as the Name
  - Enter the password of the certificate you specified while creating the certificate

Create a secret ...



The screenshot shows the 'Create a secret' form. It includes fields for Upload options (set to Manual), Name (M365ClientCertPassword), Value (redacted), Content type (optional), Set activation date? (unchecked), Set expiration date? (unchecked), and Enabled? (Yes selected). A 'Create' button is at the bottom.

### 5.2.4 Create an App Registration in Azure Active Directory

- Open the Azure Portal (<https://portal.azure.com>)
- Go to Azure Active Directory



- Under "Manage", click "App registrations"

## Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps

The screenshot shows the Microsoft Azure Active Directory Overview page. On the left, there's a sidebar with links like 'Overview', 'Getting started', 'Preview features', 'Diagnose and solve problems', 'Manage' (with sub-links for 'Users', 'Groups', 'External Identities', 'Roles and administrators', 'Administrative units', 'Enterprise applications', 'Devices', and 'App registrations'), and 'Identity Governance'. The 'App registrations' link is highlighted with a red box.

- Click "New registration" to create a new registration

The screenshot shows the Microsoft Azure Active Directory App registrations page. At the top, it says 'Microsoft | App registrations' and 'Azure Active Directory'. Below that, there are buttons for 'Overview', 'New registration' (which is highlighted with a red box), and 'Endpoints'. There's also a back arrow and a three-dot menu icon.

- Enter the following information and click "Register":
  - Name: Microsoft365DSC
  - Supported account types: Accounts in this organizational directory only
  - Redirect URI: Select Web and leave the URL empty

# Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps

Dashboard > Microsoft >  
Register an application ...

\* Name  
The user-facing display name for this application (this can be changed later).

Supported account types  
Who can use this application or access this API?  
 Accounts in this organizational directory only (Microsoft only - Single tenant)  
 Accounts in any organizational directory (Any Azure AD directory - Multitenant)  
 Accounts in any organizational directory (Any Azure AD directory - Multitenant) and personal Microsoft accounts (e.g. Skype, Xbox)  
 Personal Microsoft accounts only  
[Help me choose...](#)

Redirect URI (optional)  
We'll return the authentication response to this URI after successfully authenticating the user. Providing this now is optional and it can be changed later, but a value is required for most authentication scenarios.

By proceeding, you agree to the Microsoft Platform Policies

**Register**

- Copy the "Application (client) ID" and store it for later use

Delete Endpoints Preview features

Got a second? We would love your feedback on Microsoft identity platform (previously Azure AD for developer). →

Essentials

Display name : Microsoft365DSC	Supported account types : My organization only
Application (client) ID :	Redirect URIs : Add a Redirect URI
Directory (tenant) ID :	Application ID URI : Add an Application ID URI
Object ID :	Managed application in I... : M365Dsc

## Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps

- Click the option "Certificates & secrets"

Dashboard > Microsoft >

 Microsoft365DSC ✘

Search (Ctrl+/) <<

Overview Quickstart Integration assistant

Manage

Branding Authentication **Certificates & secrets**

Token configuration API permissions Expose an API App roles Owners Roles and administrators | Preview Manifest

Support + Troubleshooting

Troubleshooting New support request

- Click "Upload certificate" to add a certificate

Search (Ctrl+)/ Got feedback?

Overview Quickstart Integration assistant

Credentials enable confidential applications to identify themselves to the authentication service when receiving tokens at a web addressable scheme). For a higher level of assurance, we recommend using a certificate (instead of a client secret) as a credential.

Manage

Branding Authentication **Certificates & secrets**

Certificates

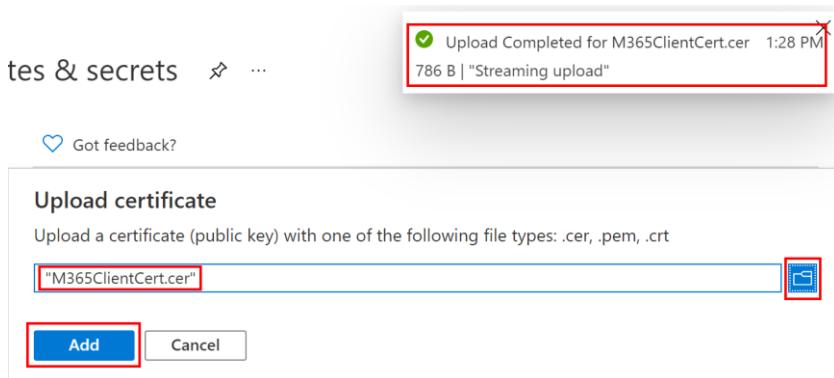
Certificates can be used as secrets to prove the application's identity when requesting a token. Also can be referred to as public keys.

Upload certificate

Thumbprint	Start date	Expires	ID
No certificates have been added for this application.			

## Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps

- In the popup, click the browse icon, select the "C:\M365ClientCert.cer" file and click "Add"



**NOTE:** After selecting the file, an upload message appears.

- The certificate has been added:

Certificates

Certificates can be used as secrets to prove the application's identity when requesting a token. Also can be referred to as public keys.

Thumbprint	Start date	Expires	ID
REDACTED	2/25/2021	2/25/2022	REDACTED ...

- Click the option "API permissions"

Manage

Branding

Authentication

Certificates & secrets

Token configuration

API permissions

Expose an API

App roles

- Click "Add a permission"

The "Admin consent required" column shows the default value for an organization. However, user consent can be customized per permission, user, or app. This column may not reflect the value in your organization, or in organizations where this app will be used. [Learn more](#)

Configured permissions

Applications are authorized to call APIs when they are granted permissions by users/admins as part of the consent process. The list of configured permissions should include all the permissions the application needs. [Learn more about permissions and consent](#)

+ Add a permission ✓ Grant admin consent for Microsoft

API / Permissions na...	Type	Description	Admin consent req...	Status
-------------------------	------	-------------	----------------------	--------

No permissions added

To view and manage permissions and user consent, try [Enterprise applications](#).

**NOTE:** Each resource/workload requires different permissions. This example only requires the currently listed permissions.

## Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps

- Select “SharePoint”  
Request API permissions

Select an API

[Microsoft APIs](#) APIs my organization uses My APIs

Commonly used Microsoft APIs

<b>Microsoft Graph</b>  Take advantage of the tremendous amount of data in Office 365, Enterprise Mobility + Security, and Windows 10. Access Azure AD, Excel, Intune, Outlook/Exchange, OneDrive, OneNote, SharePoint, Planner, and more through a single endpoint.	<b>Azure Rights Management Services</b>  Allow validated users to read and write protected content	<b>Azure Service Management</b>  Programmatic access to much of the functionality available through the Azure portal	<b>Data Export Service for Microsoft Dynamics 365</b>  Export data from Microsoft Dynamics CRM organization to an external destination
<b>Dynamics 365 Business Central</b>  Programmatic access to data and functionality in Dynamics 365 Business Central	<b>Dynamics CRM</b>  Access the capabilities of CRM business software and ERP systems	<b>Flow Service</b>  Embed flow templates and manage flows	
<b>Intune</b>  Programmatic access to Intune data	<b>Office 365 Management APIs</b>  Retrieve information about user, admin, system, and policy actions and events from Office 365 and Azure AD activity logs	<b>OneNote</b>  Create and manage notes, lists, pictures, files, and more in OneNote notebooks	
<b>Power BI Service</b>  Programmatic access to Dashboard resources such as Datasets, Tables, and Rows in Power BI	<b>SharePoint</b>  Interact remotely with SharePoint data	<b>Skype for Business</b>  Integrate real-time presence, secure messaging, calling, and conference capabilities	

- Click “Application permissions”, select “Sites.FullControl.All” and click “Add permissions”

# Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps

## Request API permissions

The screenshot shows the 'Request API permissions' interface for SharePoint. The 'Application permissions' section is highlighted with a red box. Under 'Select permissions', the 'Sites.FullControl.All' checkbox is checked and highlighted with a red box. At the bottom, the 'Add permissions' button is highlighted with a red box.

- The permission has been added, but permission has not yet been granted. Click "Grant admin consent for <org name>" to grant these permissions

### Configured permissions

Applications are authorized to call APIs when they are granted permissions by users/admins as part of the consent process. The list of configured permissions should include all the permissions the application needs. [Learn more about permissions and consent](#)

+ Add a permission	✓ Grant admin consent for Microsoft			
API / Permissions name	Type	Description	Admin consent req...	Status
<b>▼ SharePoint (1)</b>				
Sites.FullControl.All	Application	Have full control of all site collections	Yes	⚠ Not granted for Microsoft

- Click "Yes" to confirm granting the permissions

Do you want to grant consent for the requested permissions for all accounts in Microsoft? This will update any existing admin consent records this application already has to match what is listed below.

- You should receive the message that the permissions have been granted and see that the status is "Granted for <org name>"

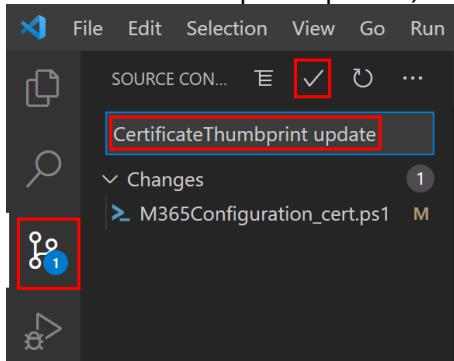
## Managing Microsoft 365 in true DevOps style with Microsoft365Dsc and Azure DevOps

The screenshot shows the Microsoft 365 Admin Center interface. At the top, there is a message box with a blue info icon and the text "Successfully granted admin consent for the requested permissions." Below this, a note says, "The 'Admin consent required' column shows the default value for an organization. However, user consent can be customized per permission, user, or app. This column may not reflect the value in your organization, or in organizations where this app will be used." A link "Learn more" is provided. Under the heading "Configured permissions", there is a table with the following data:

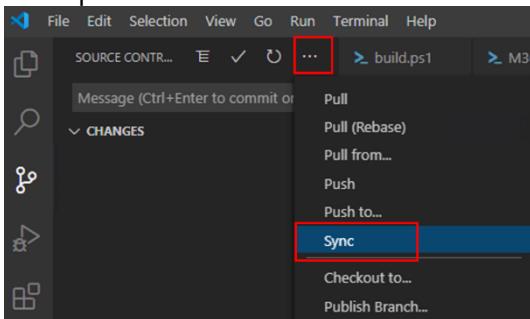
API / Permissions name	Type	Description	Admin consent req...	Status
Sites.FullControlAll	Application	Have full control of all site collections	Yes	<span>Granted for Microsoft</span>

### 5.2.5 Updating the DSC configuration with the certificate thumbprint

- Open Visual Studio Code
- Open the file "M365Configuration\_cert.ps1"
- For each resource, update the following parameters:
  - ApplicationId: Application ID stored in paragraph 5.2.4 "Create an App Registration in Azure Active Directory".
  - TenantId: Change <tenant> with your tenant name.
  - CertificateThumbprint: Certificate thumbprint stored in the last step of paragraph 5.2.1 "Creating the authentication certificate".
- Click on the Git Source Control icon in the left menu, type a commit message (e.g. "CertificateThumbprint update") and click the checkmark icon



- Click the three dots icon and select "Sync" to sync your local changes with Azure DevOps



## 5.2.6 Creating the Build and Release pipelines

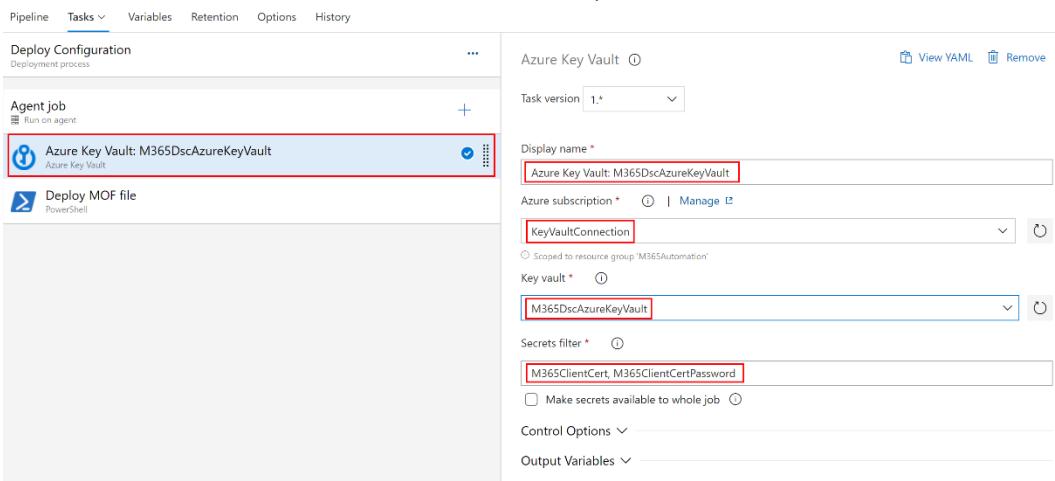
To create the Build and Release pipelines using certificate, you can now follow the paragraphs 4.2.1 "Create Build pipeline" and 4.2.2 "Create Release pipeline", with two exceptions

- 1.) Change the following files:

- azure-pipelines.yml -> azure-pipelines\_cert.yml
- build.ps1 -> build\_cert.ps1
- deploy.ps1 -> deploy\_cert.ps1
- M365Configuration.ps1 -> M365Configuration\_cert.ps1

- 2.) Add a "Azure Key Vault" task to the Release pipeline (before the PowerShell task) with the following details:

- Display name : Azure KeyVault: M365DscAzureKeyVault
- Azure subscription : KeyVaultConnection
- Key vault : M365DscAzureKeyVault
- Secrets filter : M365ClientCert, M365ClientCertPassword



## 6 Script details

This whitepaper used some pre-created scripts. You can use these scripts as-is or tailor them to your own situation. This section describes what each script is for.

You can download the script package at:

<https://microsoft365dsc.com/Pages/Resources/Whitepapers/M365Automation.zip>

The package contains these files:

File name	Description
<b>.gitignore</b>	File used by Git, which specifies all files and folders Git has to ignore.
<b>azure-pipelines.yml</b>	The configuration file for the Azure DevOps Build Pipeline that is using the username/password. This file defines which steps are required to build the DSC MOF file.
<b>azure-pipelines_cert.yml</b>	The configuration file for the Azure DevOps Build Pipeline that is using Certificates for authentication (see paragraph 5.2). This file defines which steps are required to build the DSC MOF file.
<b>build.ps1</b>	The script that is responsible for retrieving the service account password from Azure KeyVault and building the DSC MOF file.
<b>build_cert.ps1</b>	The script that is responsible for building the DSC MOF file and using a certificate for authentication.
<b>deploy.ps1</b>	The script that is responsible for deploying the DSC MOF file to the LCM of the virtual machine.
<b>deploy_cert.ps1</b>	The script that is responsible for importing the authentication certificate and deploying the DSC MOF file with certificate authentication to the LCM of the virtual machine.
<b>DscResources.psd1</b>	Data file that specifies the version of Microsoft365Dsc to be used.
<b>M365Configuration.ps1</b>	The Microsoft 365 configuration file which defines the target state using username/password.
<b>M365Configuration_cert.ps1</b>	The Microsoft 365 configuration file which defines the target state using certificate authentication.
<b>M365ConfigurationData.psd1</b>	The parameter file used to make sure the configuration can be used across multiple environments.
<b>ReadMe.md</b>	A project description file in Markdown format. This will be displayed when opening the repository in Azure DevOps.

## 7 Learning materials

### 7.1 Desired State Configuration

- Channel9: "Getting Started with PowerShell Desired State Configuration"
  - <https://channel9.msdn.com/Series/Getting-Started-with-PowerShell-Desired-State-Configuration-DSC>
- Channel9: "Advanced PowerShell Desired State Configuration"
  - <https://channel9.msdn.com/Series/Advanced-PowerShell-Desired-State-Configuration-DSC-and-Custom-Resources>
- Desired State Configuration Overview for Engineers
  - <https://docs.microsoft.com/en-us/powershell/scripting/dsc/overview/dscforengineers>
- Creating configurations
  - Configurations: <https://docs.microsoft.com/en-us/powershell/scripting/dsc/configurations/configurations>
  - DependsOn: <https://docs.microsoft.com/en-us/powershell/scripting/dsc/configurations/resource-depends-on>
  - DSC Resources: <https://docs.microsoft.com/en-us/powershell/scripting/dsc/resources/resources>
- Using configuration data in DSC
  - <https://docs.microsoft.com/en-us/powershell/scripting/dsc/configurations/configdata>
  - <https://docs.microsoft.com/en-us/powershell/scripting/dsc/configurations/separatingenvdata>
- Secure the MOF file
  - <https://docs.microsoft.com/en-us/powershell/scripting/dsc/pull-server/securemof>
  - <https://docs.microsoft.com/en-us/powershell/scripting/dsc/configurations/configdatacredentials>
- Local Configuration Manager
  - Configuring: <https://docs.microsoft.com/en-us/powershell/scripting/dsc/managing-nodes/metaconfig>
  - Push/Pull model: <https://docs.microsoft.com/en-us/powershell/scripting/dsc/pull-server/enactingconfigurations>
- Apply, Get, and Test Configurations on a Node
  - <https://docs.microsoft.com/en-us/powershell/scripting/dsc/managing-nodes/apply-get-test>
- Debugging DSC
  - <https://docs.microsoft.com/en-us/powershell/scripting/dsc/troubleshooting/debugresource>

## 7.2 Microsoft365Dsc

- Microsoft365dsc.com
  - <https://microsoft365dsc.com/>
- Microsoft365Dsc promotion video
  - <https://aka.ms/m365dscpromo>
- GitHub repository
  - <https://github.com/microsoft/Microsoft365DSC>
- What is Configuration-as-Code?
  - <http://nikcharlebois.com/what-is-configuration-as-code>
- Microsoft365Dsc YouTube channel
  - <https://www.youtube.com/channel/UCveScabVT6pxzqYgGRu17iw>
- PluralSight: "SharePoint Conference '19: IT Pros, Get Relevant! Upskilling for Today's Cloud" (subscription required)
  - <https://www.pluralsight.com/courses/sharepoint-conference-2019-session-19>

## 7.3 Git

- Git manual
  - <https://git-scm.com/book/en/v2>
- PluralSight: "How Git Works" (subscription required)
  - <https://app.pluralsight.com/library/courses/how-git-works/table-of-contents>
- PluralSight: "Mastering Git" (subscription required)
  - <https://app.pluralsight.com/library/courses/mastering-git/table-of-contents>

## 8 Acronyms

Acronym	Meaning
DSC	Desired State Configuration
LCM	Local Configuration Manager
VM	Virtual Machine