KPMG

cutting through complexity

Saviynt Windows Server & Powershell Prerequisites

<Client> KPMG Build

July 2022
Version: 1.0

Table of Contents

1 Document Control

1.1 Document Review & Feedback

An updated version of this document has been created and will be reviewed by the team members and stakeholders listed below. The feedback obtained from their review will be incorporated.

Manjunath Madiraju	07/20/2022	Architect	07/20/2022
Architect	Date	<Title>	Date
Security Architect	Date	<Title>	Date
<Title>	Date	<Title>	Date

1.2 Document Acceptance

Representative Approvers

By signing this document, you confirm that you have read, reviewed, and approved the contents of this deliverable.

AGREED TO AND ACCEPTED BY:	**AGREED TO AND ACCEPTED BY:**
Client Project Management	Vendor Project Management
By: _____	By: _____
Name: _____	Name: _____
Title: _____	Title: _____
Date: _____	Date: _____

1.3 Modification History

Change Date	Author	Version Modified	Description of Changes
July 20, 2022	Manjunath Madiraju	1.0	Initial Creation

2 Document Purpose

Enabling access to technology resources in a secure and efficient manner is at the core of a strong cyber security program. An organization must provide its workforce (employees, contractors and business partners) with the required access to securely enable business operations and collaboration. The purpose of this document is to provide Windows Server Prerequisites for <Client>, and execute powershell scripts.

3 Introduction

- The **Win-PS Connector** helps you to manage provisioning and deprovisioning of accounts as well as granting and revoking access to accounts on the Windows Server and the utilities installed on the server such as Active Directory (AD) Domain Services.
- **EIC** communicates with the Windows server or its utilities (example, AD Domain Services) through an application developed by Saviynt named, **SaviyntApp**. The SaviyntApp invokes a PowerShell (PS) session on the Windows Server and executes PS scripts.

Supported Features:

Feature	Supported Operation	Supported Features
Provisioning and De-provisioning	Manage Accounts	Supports the following provisioning and de-provisioning operations of O365 Mailbox and Licenses.
	Change Password	Supports provisioning and de-provisioning password change from EIC.

Supported Operating System or Platform Versions

OS/Platform	Version
.Net Framework	4 and later
Windows Server	2012 R2 and later
Windows Internet Information Services (IIS) Server	• Windows Server 2012 R2 or later • IIS Version 7 or later installed with the **URL Rewrite** module **Note:** If the URL Rewrite module is not installed, a 404 error is reported when you try to add outbound rules to manage links.
SaviyntApp or Saviynt PowerShell Agent Package	

Note: You must have the login credentials with administrator rights for hosting the Windows server and the ability to execute desired PowerShell scripts or commands.

Supported Hardware Specifications:

Specifications	Value

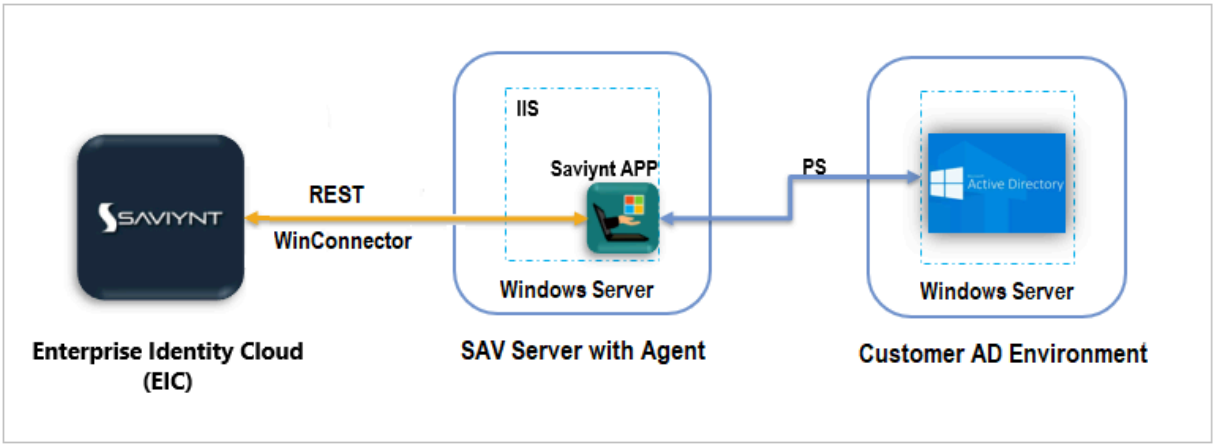
CPU	4 core 64-bit processor
RAM	16 GB
Storage	100 GB

4 Connector Architecture

By default, the connector is configured to run in provisioning mode.

The following architecture diagram illustrates the Win-PS Connector architecture and communication with Windows Server via SaviyntApp (which is a Saviynt application installed on the Windows server). The connector is used for provisioning and de-provisioning of data from EIC to the Windows Server using REST API.

The SaviyntApp invokes a PowerShell session on the customer's Windows Server and connects to the utility services such as SharePoint, Office 365, or AD Domain Services. This PowerShell session is used to administer such utilities in the Windows Server.



5 Win-PS Prerequisites

5.1 Preparing the Target System (Windows Server) for Integration

1. **Installing Windows Internet Information Services Server**

   You can install IIS in two ways: GUI or Windows PowerShell command-line interface (CLI). If you do not have the GUI installed, use the PowerShell method. This method is relatively faster than the GUI method. For usage instructions, visit the Microsoft Documentation web site and search for the instructions for your operating system.

   This section provides a detailed procedure for installing IIS web server version 10.0 on Windows Server 2016. For installing IIS on other supported Windows Server versions, see the corresponding instructions in the Microsoft Documentation.

Note: You must have the login credentials with administrator rights for hosting Windows Server and the ability to execute desired PowerShell scripts/commands.

2. **Installing IIS from Windows Server User Interface**

To install IIS Server from the Windows Server User Interface, perform the following steps:

3. Click the **Start** icon.

4. Open **Server Manager**. Alternatively, type Server Manager from the search bar near the **Start** menu to open the Server Manager.

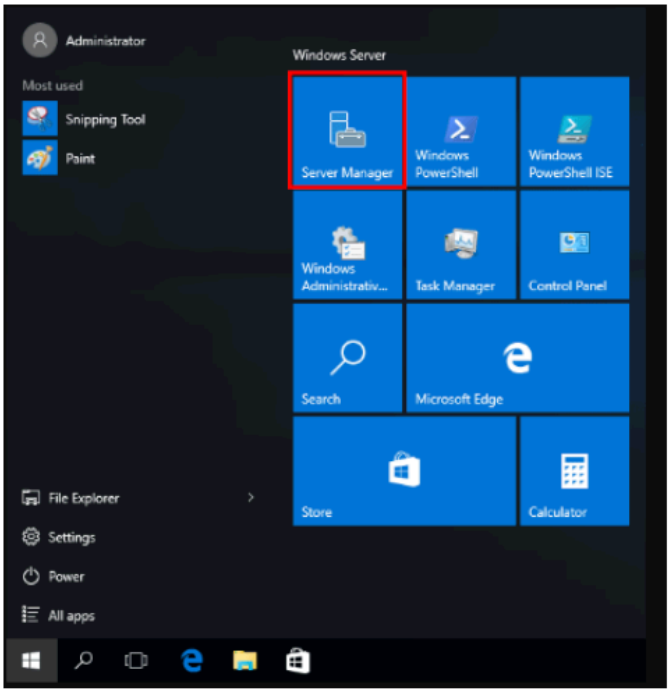


Figure: Selecting Server Manager option

5. In the **Server Manager Dashboard** page, click **Add roles and features**.

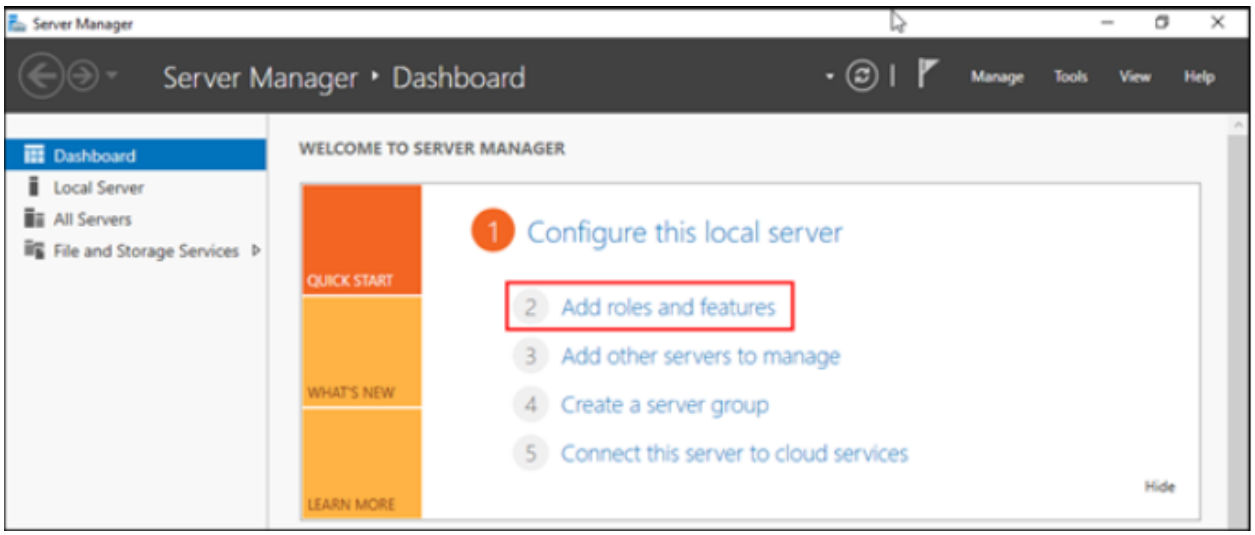


Figure: The Add Roles and Features Wizard

6. In the **Before You Begin** window, click **Next**.

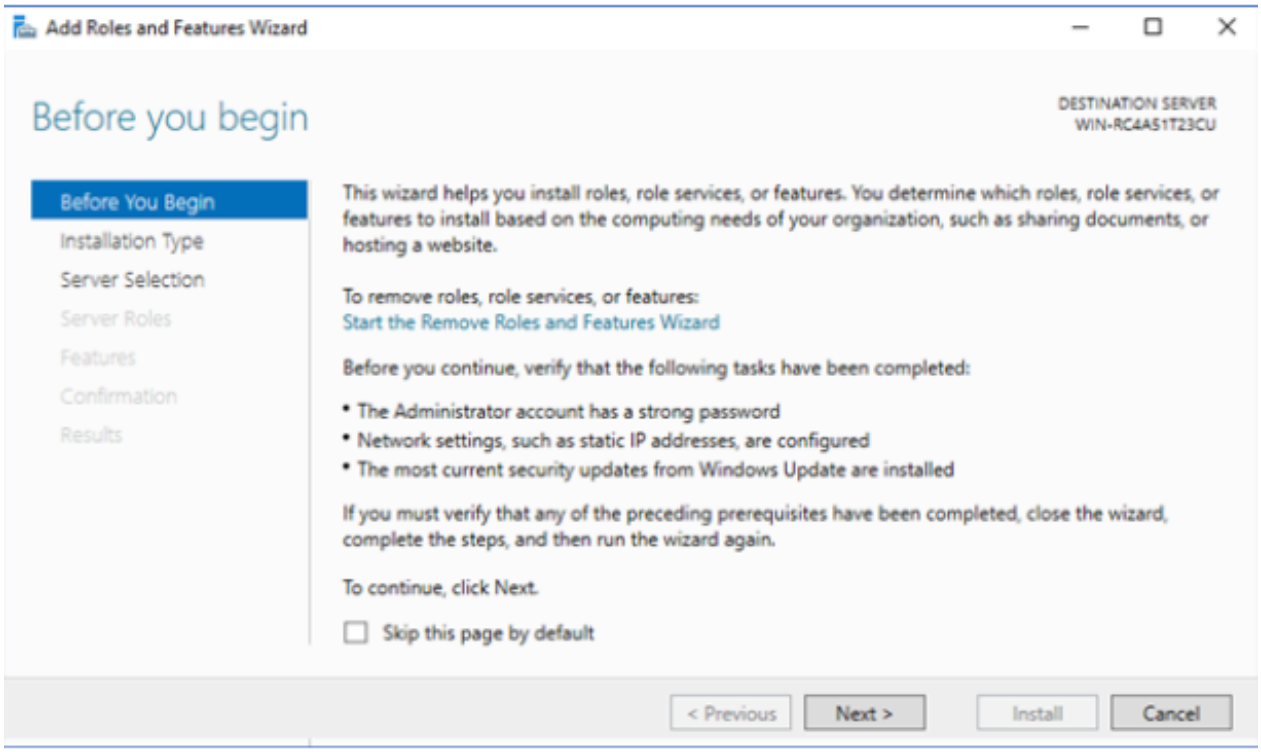


Figure: Before You Begin Window in Add Roles and Features Wizard

7. In the **Select installation type** window, select **Role-based or feature-based installation**, and click **Next**.

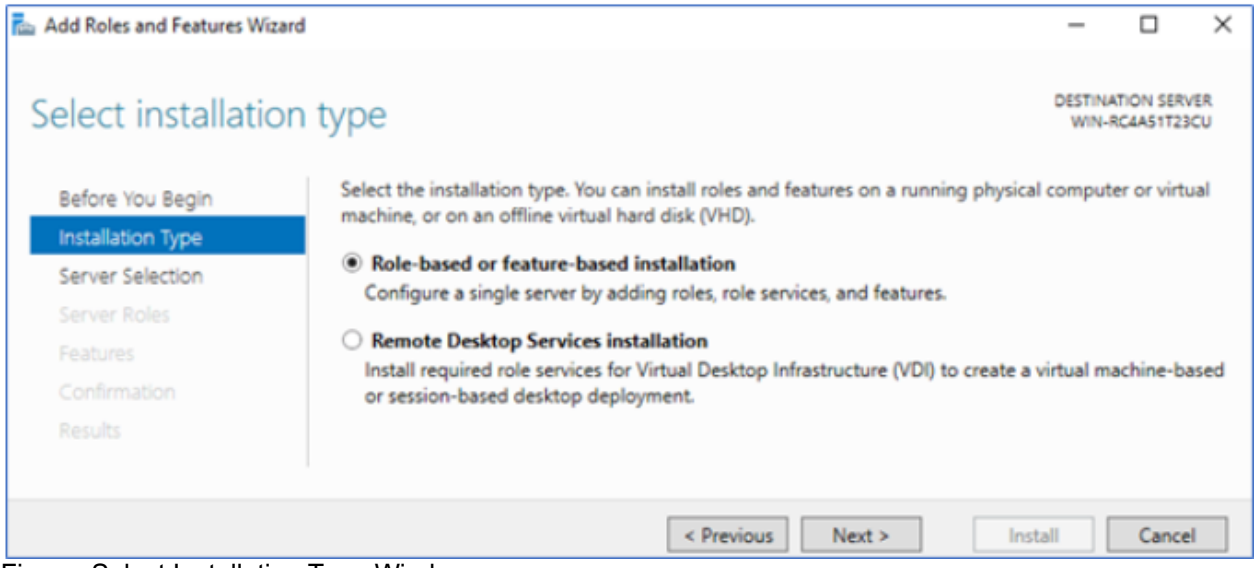


Figure: Select Installation Type Window

8. Select **Select a server from the server pool** with the current Windows Server selected, and click **Next**. Alternatively, you can select another server that you are managing from here.

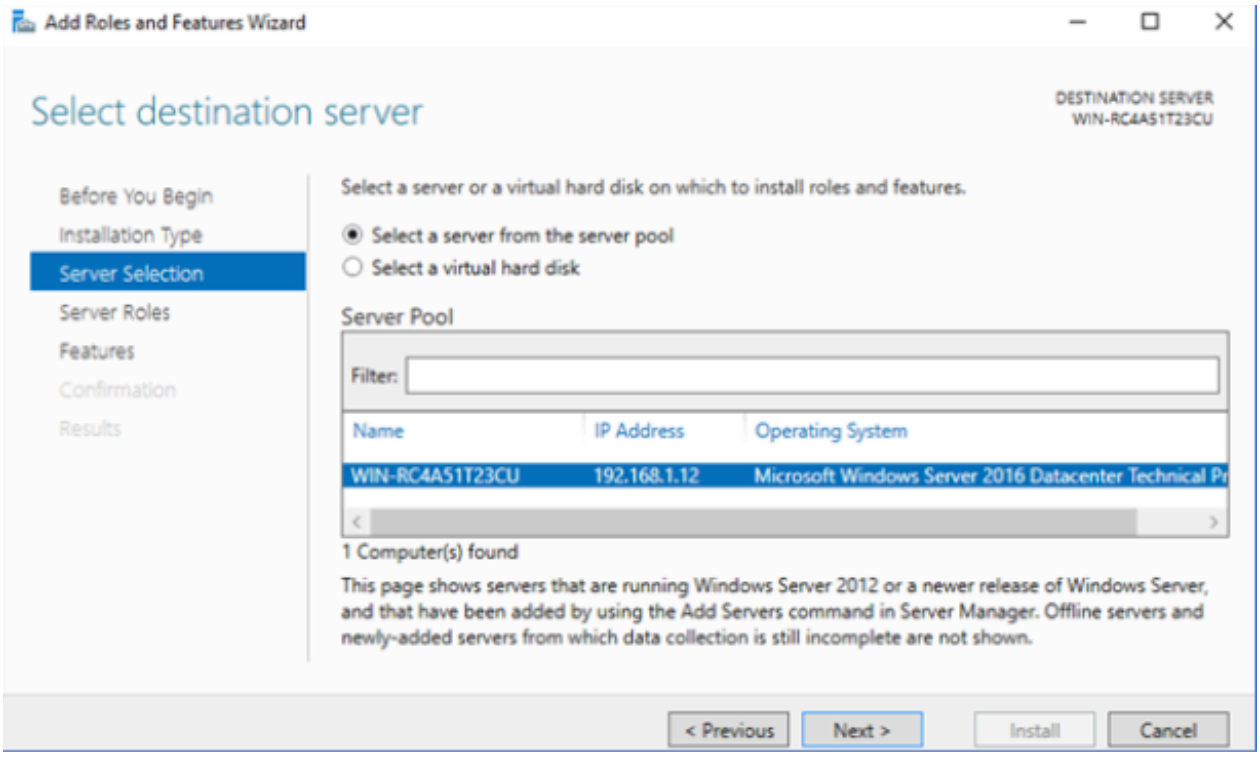


Figure: Selecting the Destination Server

9. In the **Select server roles** window, select **Web Server (IIS)**.

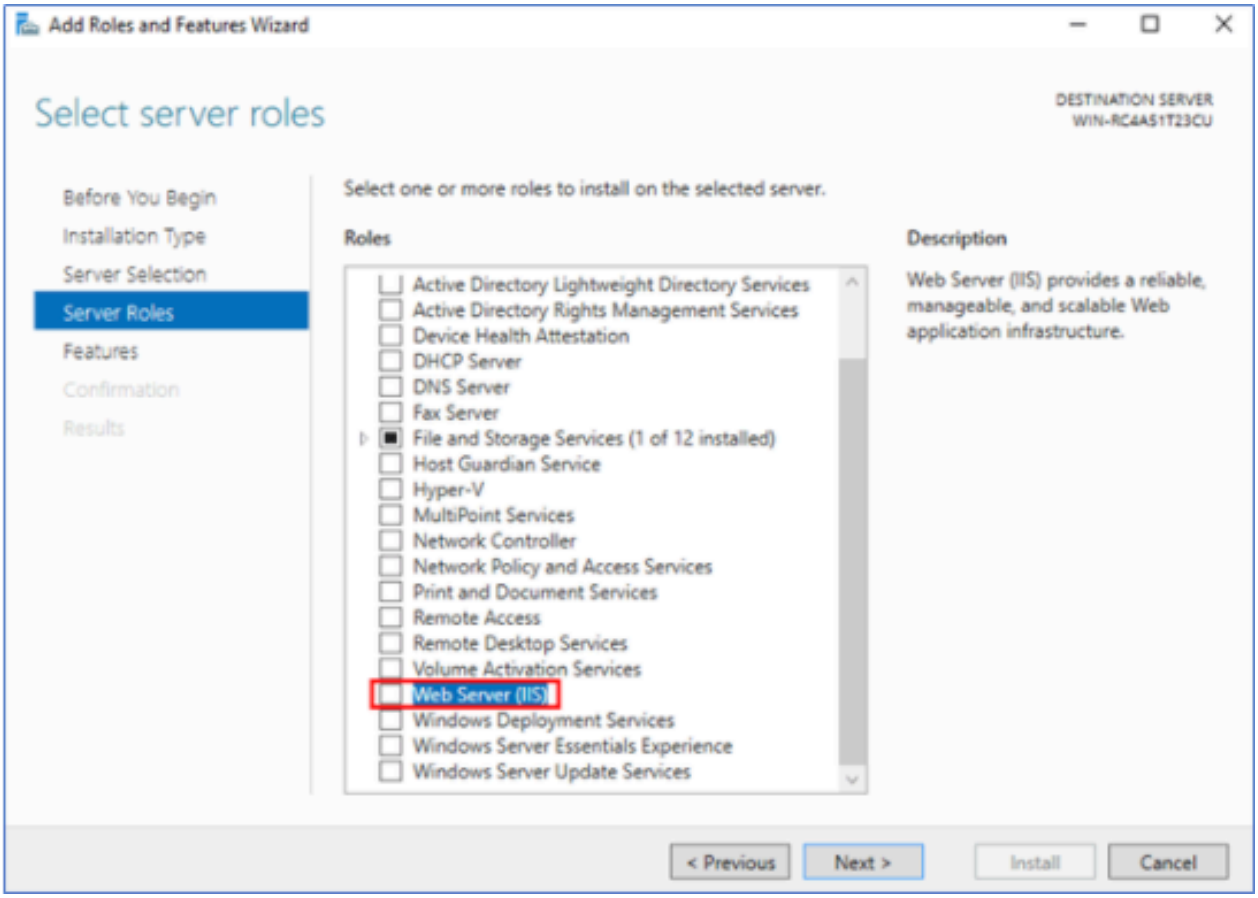


Figure: Selecting Web Server

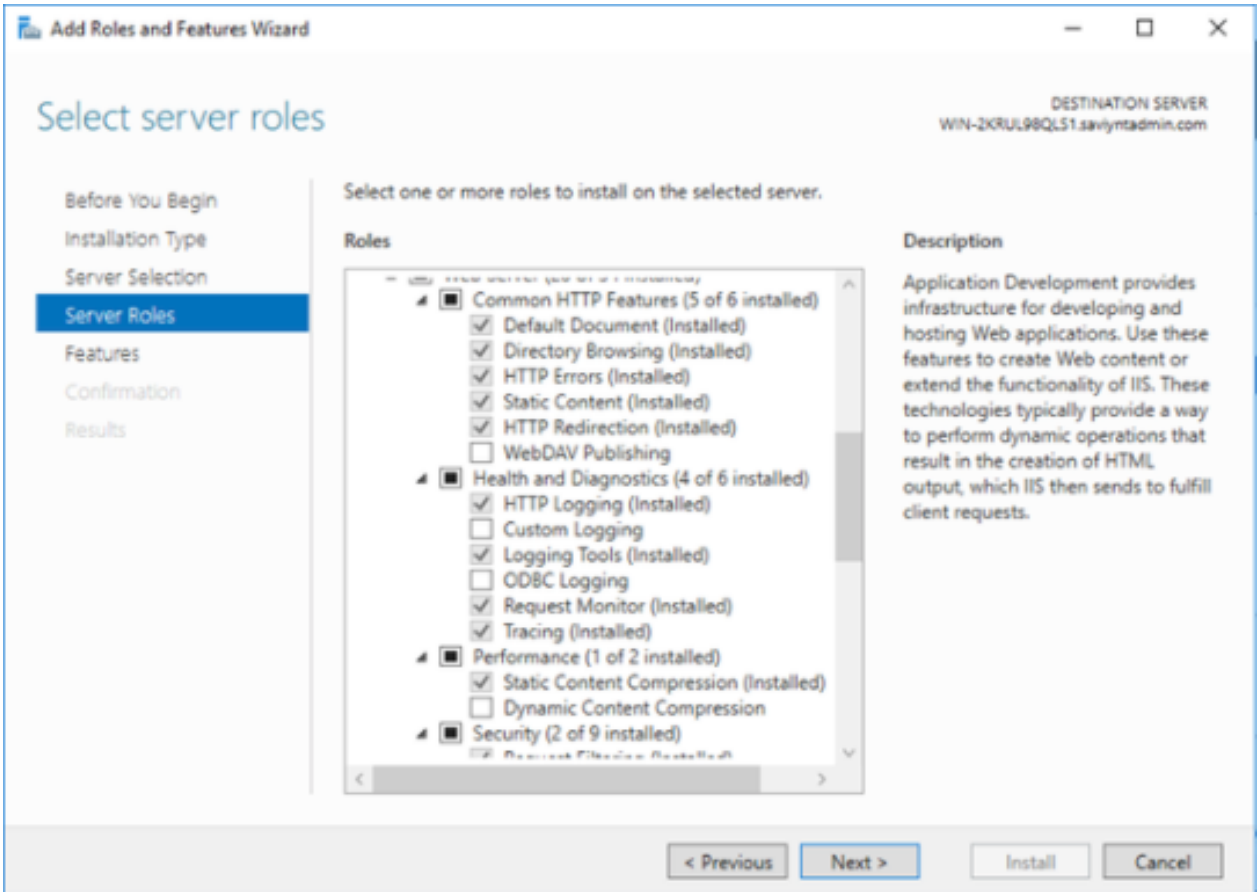


Figure: Selecting Server Roles

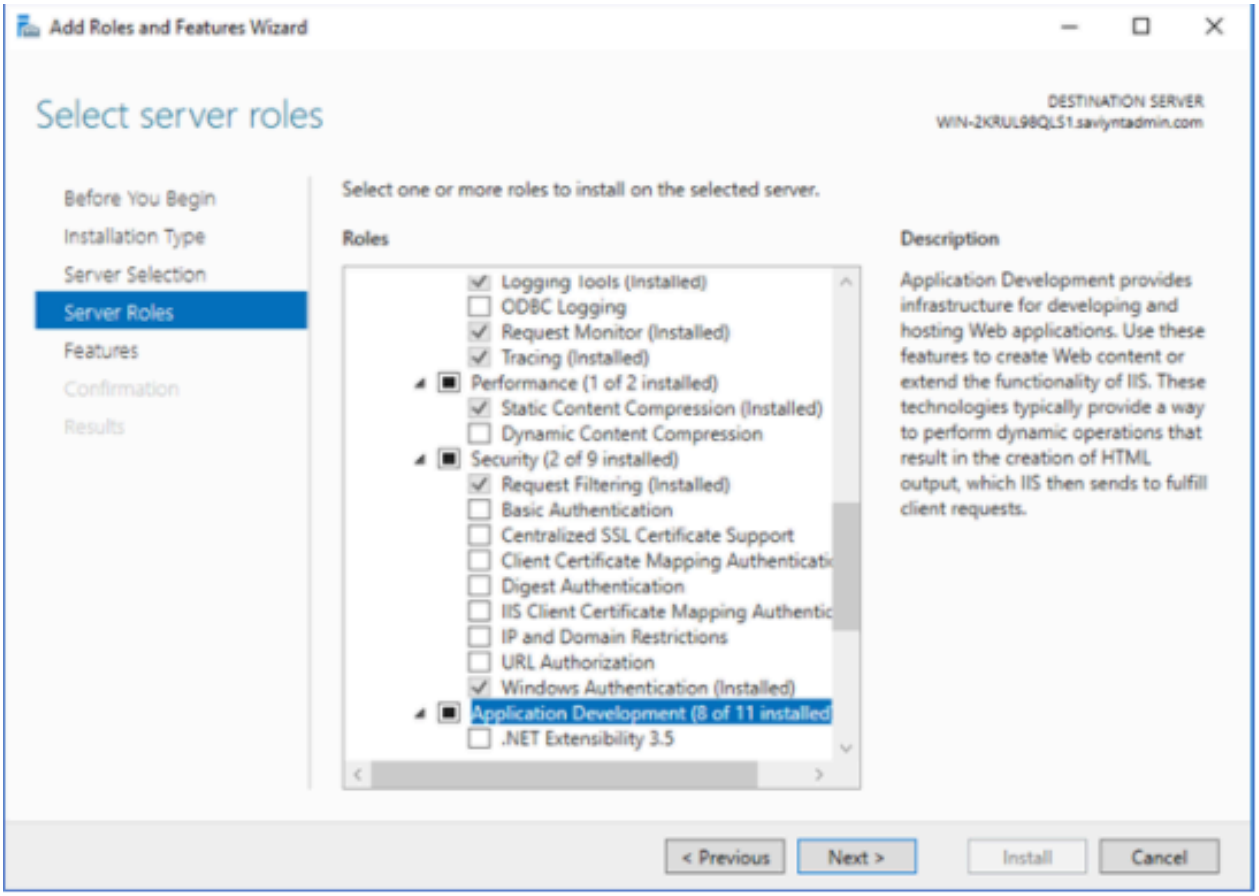


Figure: Select Server Roles Continued

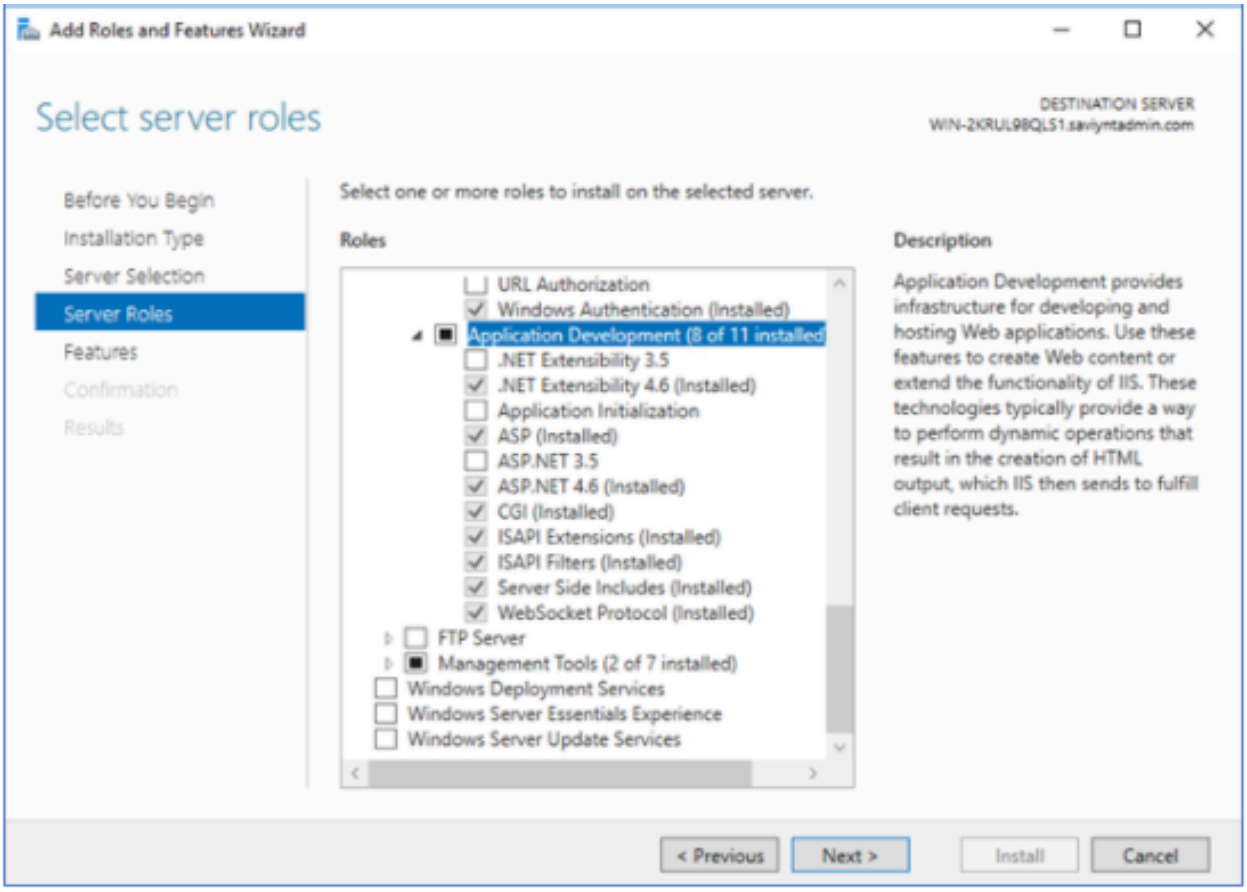


Figure: Select Server Roles Continued

10. Click **Next**.

11. In the **Add features that are required for Web server (IIS)** window, click **Add Features** to install additional features.

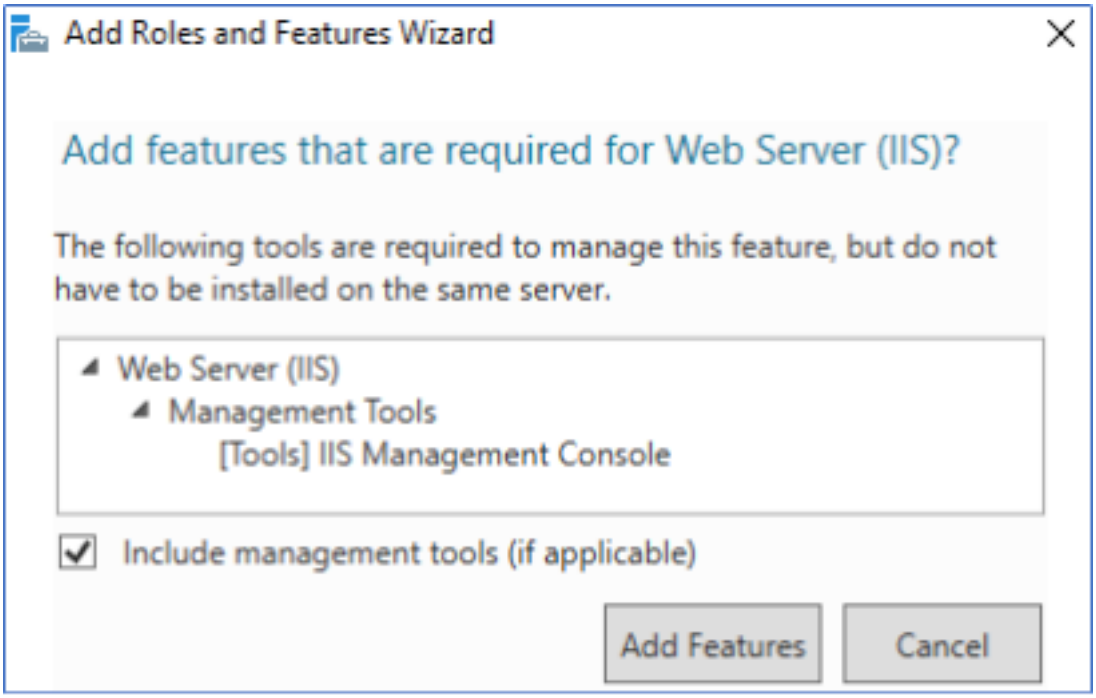


Figure: Add Features Option

12. In the **Select features** window, click **Next**.

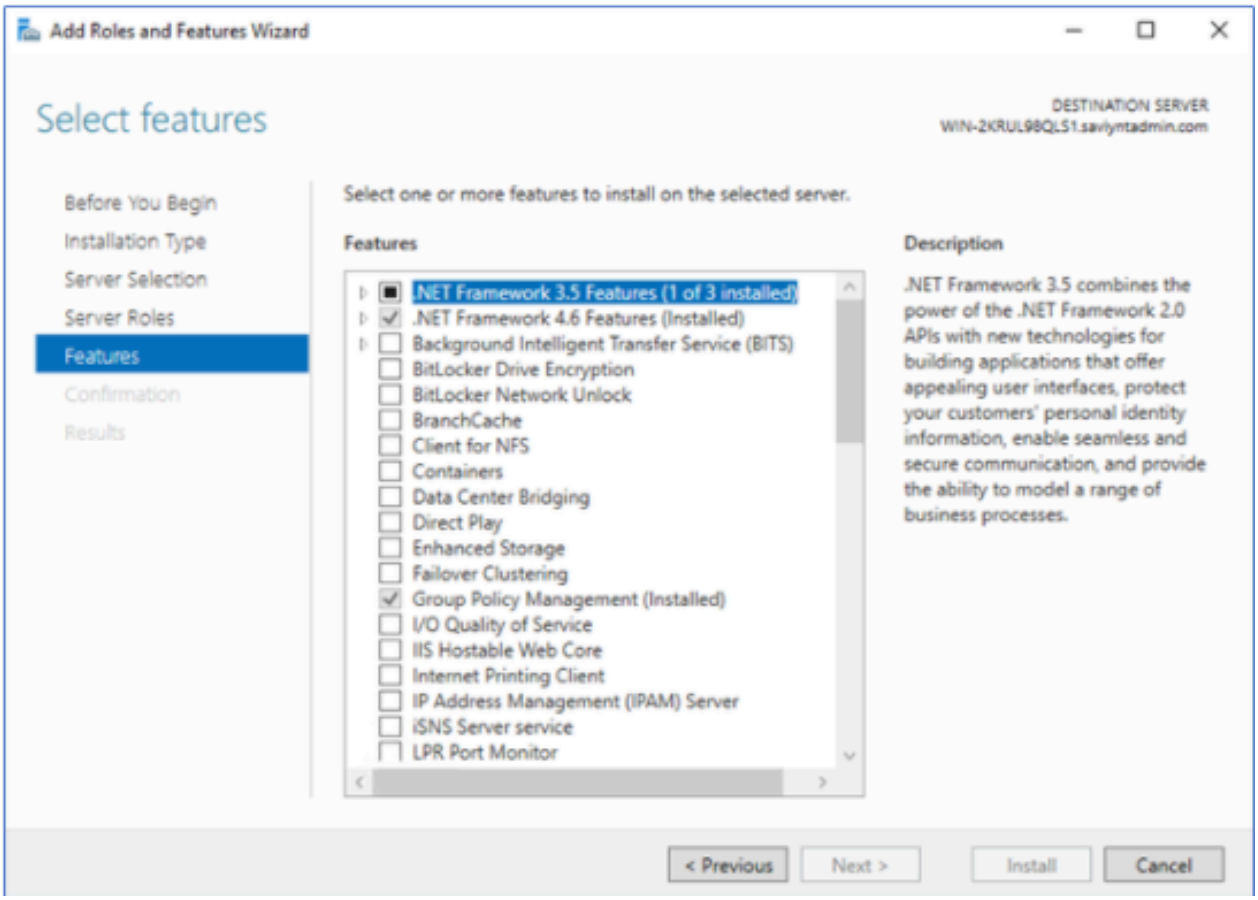


Figure: Select Features Window

13. In the **Web Server Role (IIS)** window, read the information provided about Web Server (IIS) role click **Next**.

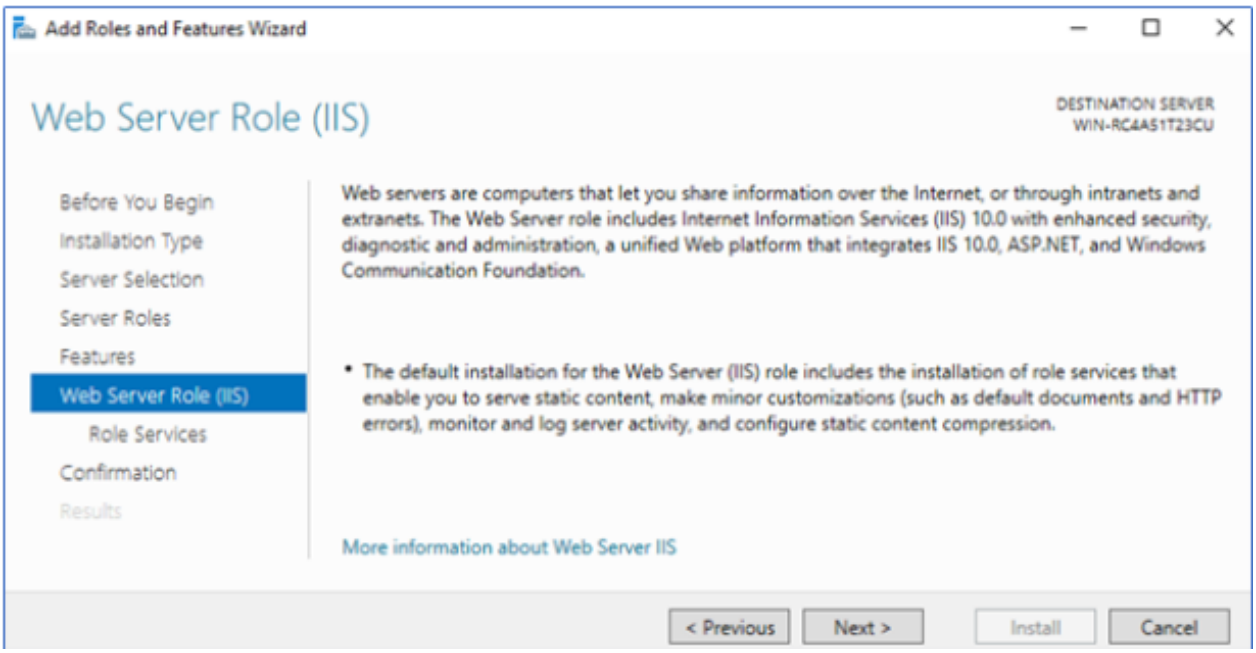


Figure: Web Server Role (IIS)

14. In the **Select role services** page, install additional services for IIS.

15. In the **Confirm installation selections** window, review the items to be installed and click **Install**. This installs the IIS web server.

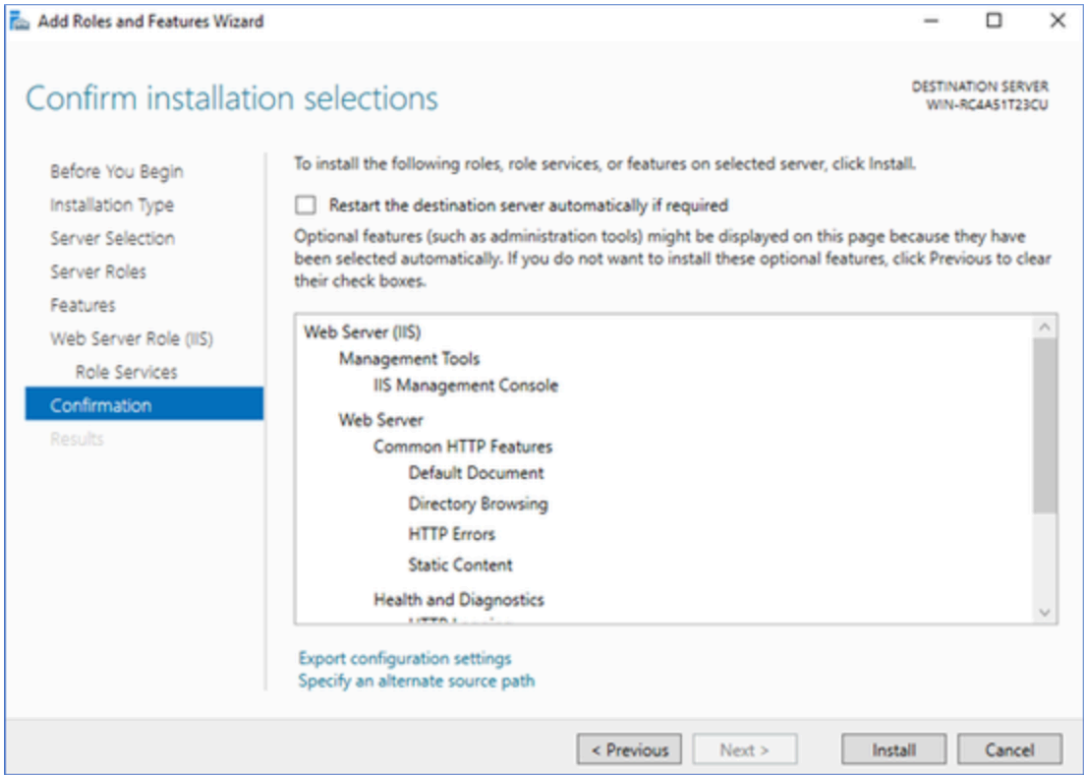


Figure: Confirming Installation Selections

Note: A reboot is not required in a standard IIS server installation; however, if you remove the role, a reboot is required.

16. Click **Close** after the installation is complete.

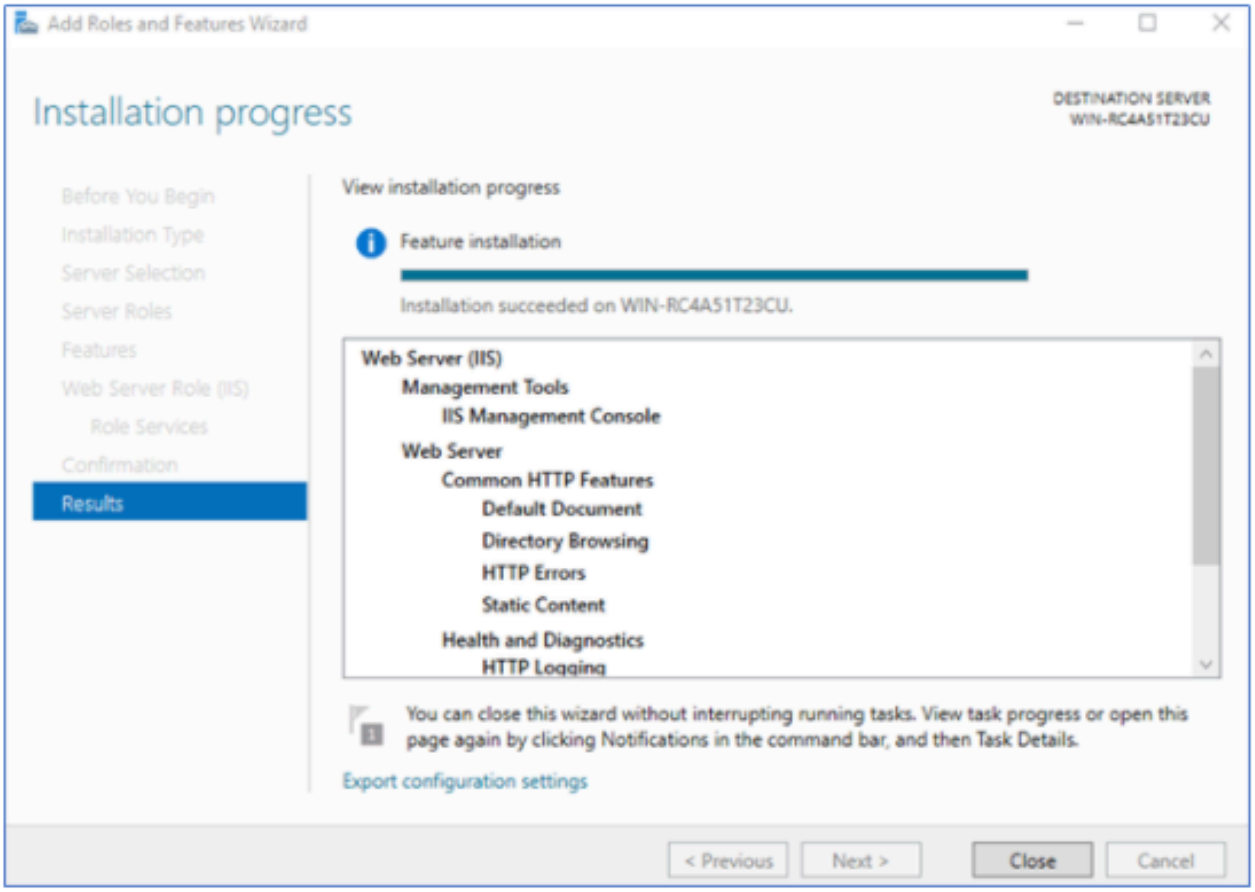


Figure: Installation Progress and Completion

Note: You can always add information in the **Select role services** window or the previous steps of the **Add Roles and Features** Wizard, if required.

IIS is installed and runs on port 80 (by default) with the firewall rule **World Wide Web Services (HTTP Traffic-In)** automatically enabled in the Windows Firewall.

After this installation is completed, install the **SaviyntApp**. For more information, see the *Downloading and Deploying the Saviynt App* section**.**

17. **Downloading and Deploying the Saviynt App**

This section explains how to download and deploy the Saviynt App in Windows IIS Server.

18. **Downloading the Saviynt App**

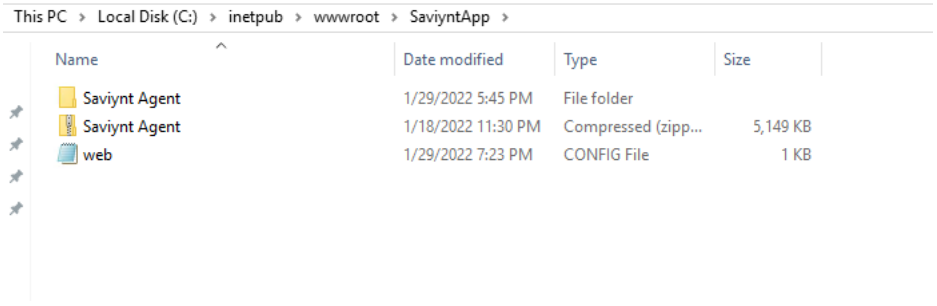
Download the Saviynt App (or Saviynt PowerShell Agent) to your system.

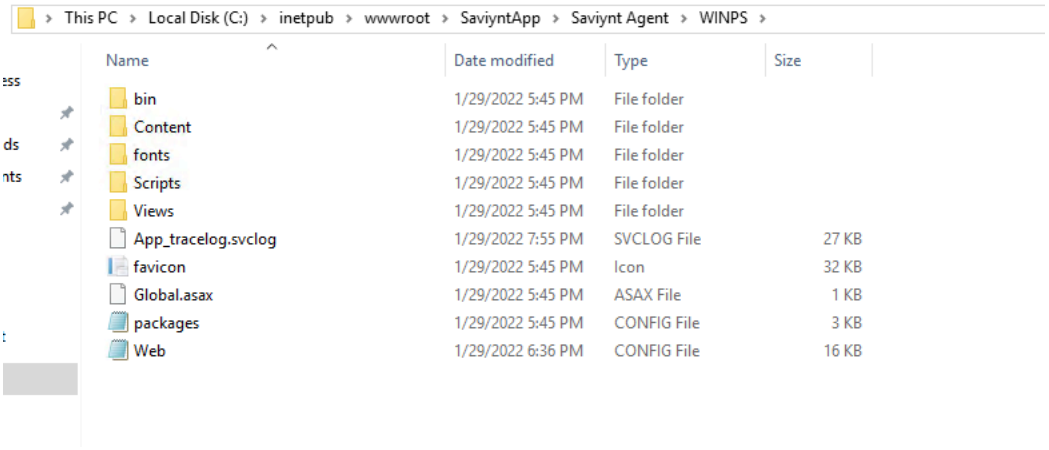
19. **Deploying Saviynt App**

This section explains the deployment of the Saviynt App within the Windows IIS Server. This process is essential for the communication between EIC to open a PowerShell session on the target system (Windows Server) that executes PowerShell scripts.

To deploy Saviynt App in the Windows IIS Server, perform the following steps:

1. Copy the Saviynt Agent.zip to C:\inetpub\wwwroot\SaviyntApp and Unzip to C:\inetpub\wwwroot\SaviyntApp\Saviynt Agent





2. To add an application pool with .Net Framework 4 version.

    1. In the IIS Manager, create an application pool and provide a name.
    2. Select the framework version as **.NET Framework v4.0.30319.**
    3. Select mode as **Integrated**.
    4. Click **OK**.

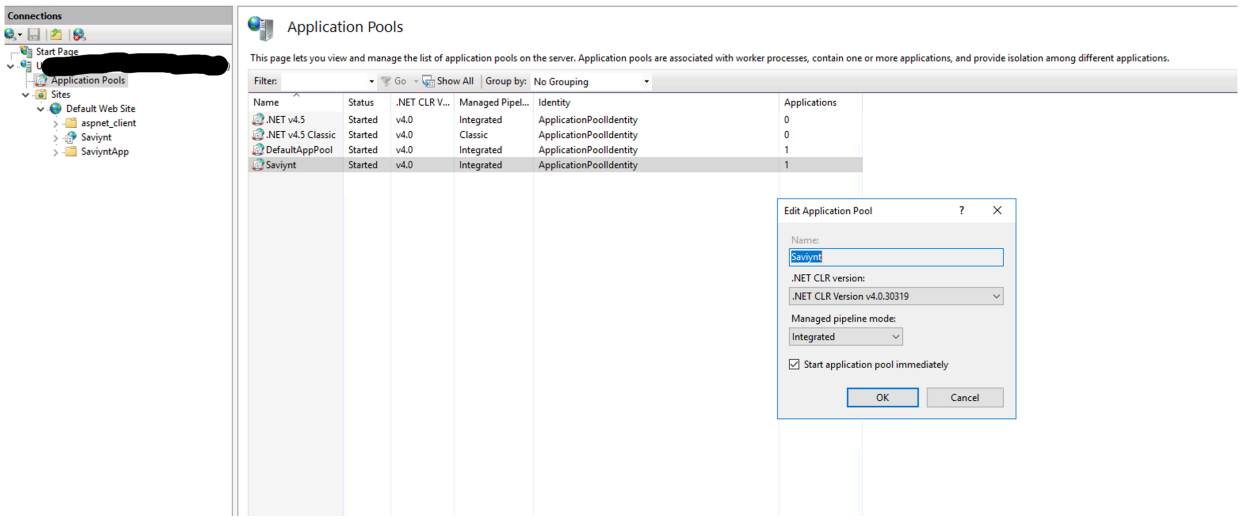


Figure: Adding Application Pool

3. Create an application under **Default Web Site**.

1. Provide a name under **Alias**.

2. Select the application pool you created in Step 1.

3. Select the physical path as local folder path where you plan to deploy the code, for example, C:\inetpub\wwwroot\SaviyntApp\Saviynt Agent\WINPS
You can create new folders in the **Browse For Folder** window.

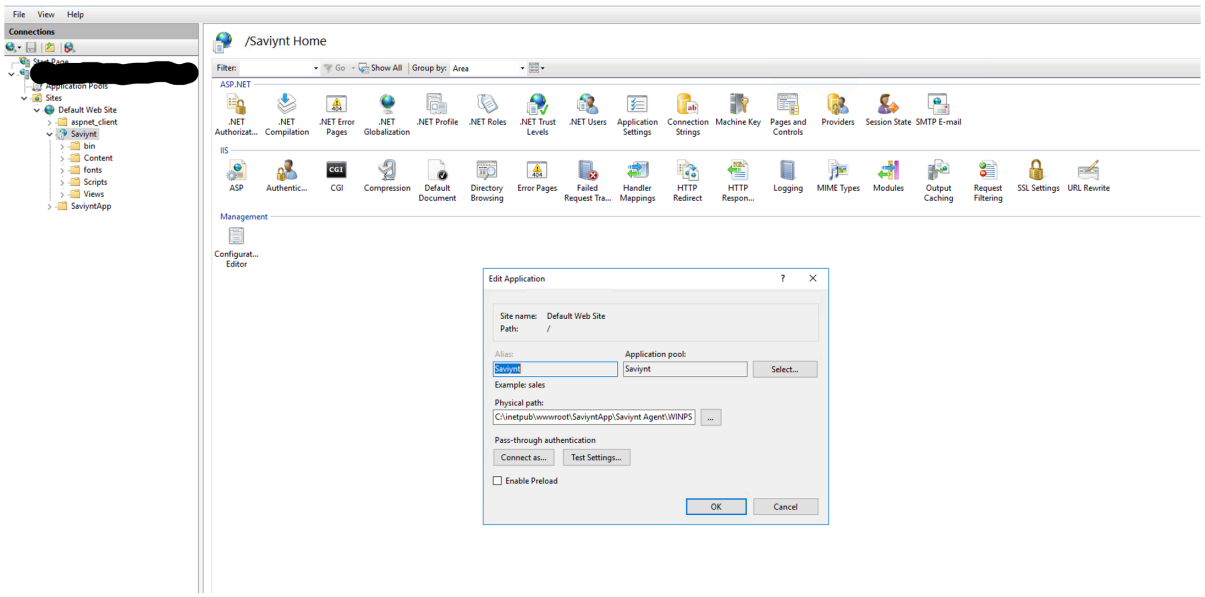


Figure: Creating an Application within the Default Web Site

4. Copy the Saviynt App package in the physical path mentioned in Step 2.

5. Install the SSL certificate for this website.
   To install the SSL certificate, see [Installation of SSL/TLS Certificate In Microsoft IIS 7].

6. Go to **Sites**, right-click **Default Web Site** and select **Edit Bindings**.

7. Select the SSL site with port 443 and click **Edit**.

8. In the **Edit Site Binding** page, view the SSL certificate and click **OK**.

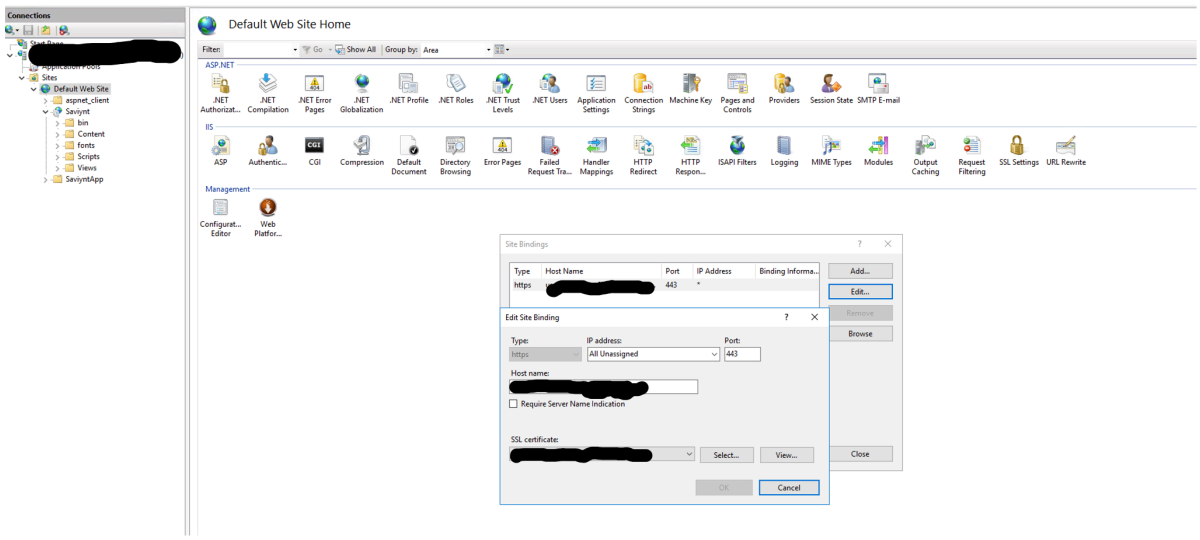


Figure: Edit Site Binding page

9. In the **Site Bindings** page, select the http binding and click **Remove** to delete the http binding for the application.

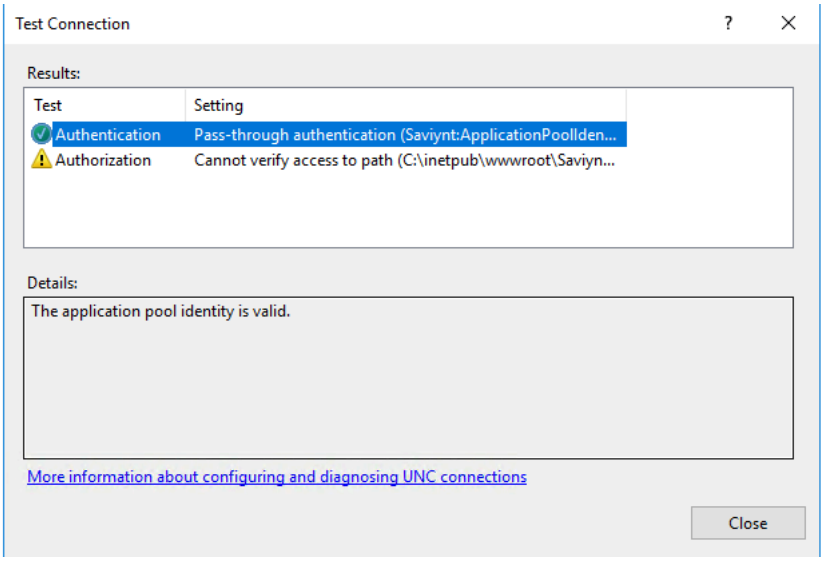
10. Go to **Sites** > **Default Web Site** and select **Basic Settings** under **Actions**.

11. In the **Edit Site** window, click **Connect as**.

12. In the **Connect As** window, select **Application user (pass-through authentication)** and click **OK**.

13. In the **Edit Site** window, click **Test Settings**.

14. In the **Test Connection** page, view the results and click **Close**.



15. Click **Restart** under **Actions** to restart the IIS Server.

20. **Configuring Outbound Rules**

You configure the outbound rules to remove server and ASP.NET information from response headers.

To configure outbound rules, perform the following steps:

1. Navigate to IIS Manager and select a computer name.

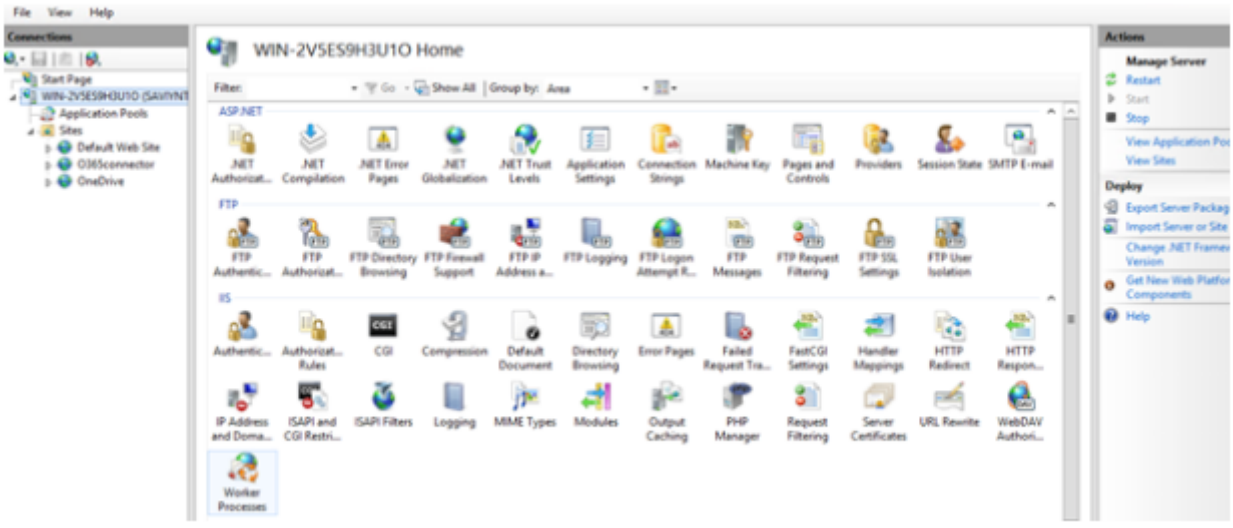


Figure: IIS Manager Home

2. Click on **URL Rewrite** under IIS to add outbound rules.

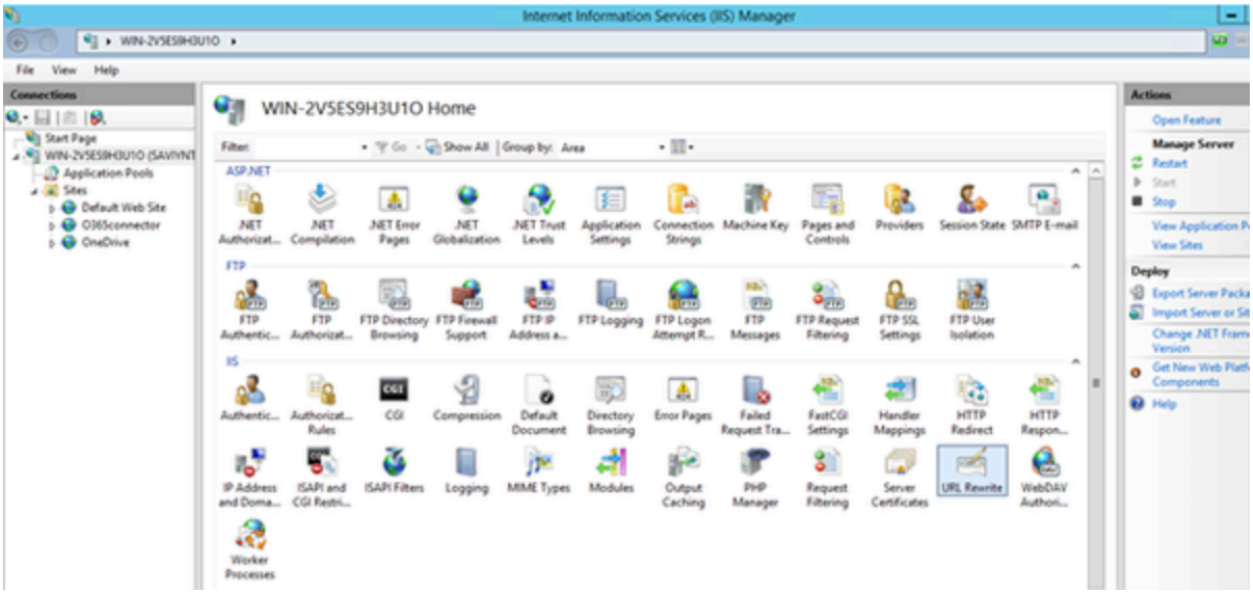


Figure: URL Rewrite option

3. Click **Add Rules** under **Actions**.

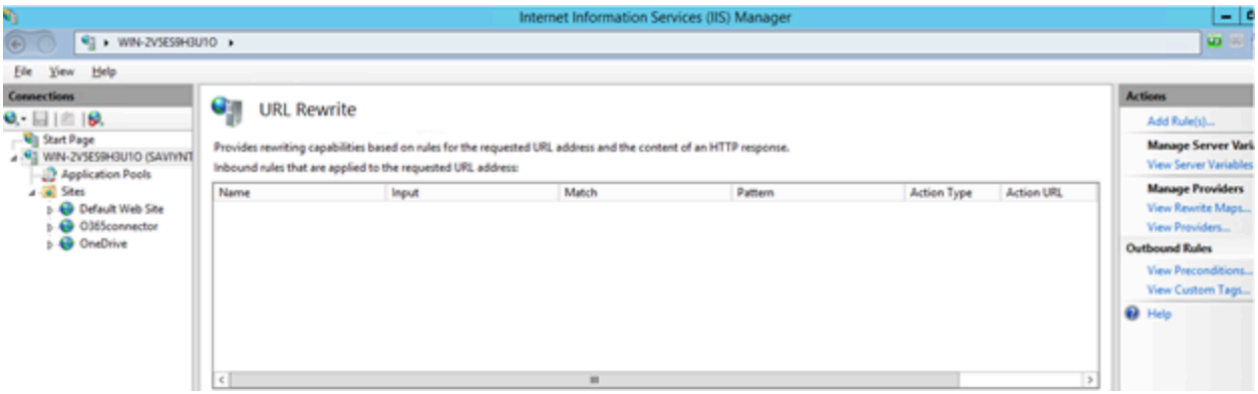


Figure: Add rules page

4. Select **Blank Rule** under **Outbound Rules** and click **Ok**.

5. Specify a name for the rule and specify the details as shown below:

    1. Select precondition as **None**.

    2. Select matching scope as **Server Variable**.

    3. Select variable name as **RESPONSE_SERVER**.

    4. Select a pattern as ***.

    5. Select action type as **Rewrite**.

6. Select value as **Null**.

7. Enable the Replace existing server variable checkbox.

Figure: Edit outbound rules

6. Click **Apply** under **Actions**.

7. Repeat steps 3 to 5 for the following variable names:

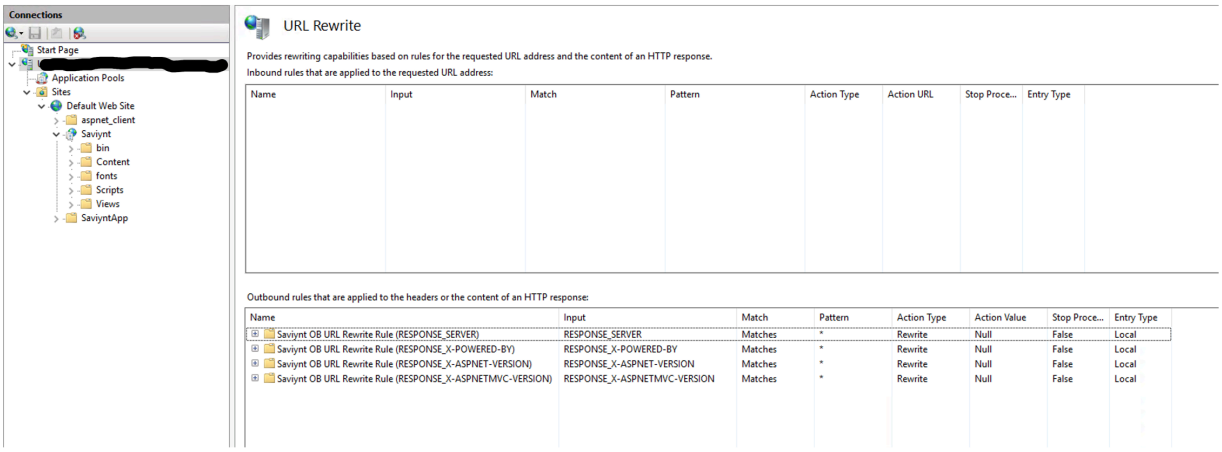
    1. RESPONSE_X-POWERED-BY

    2. RESPONSE_X-ASPNET-VERSION

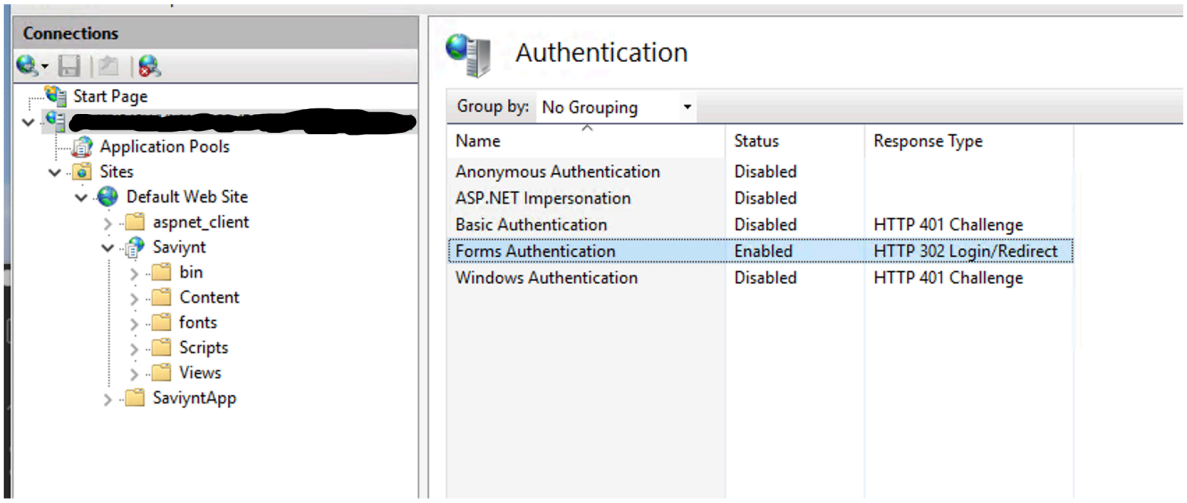
    3. RESPONSE_X-ASPNETMVC-VERSION

8. Click **Restart** under **Actions** to restart the IIS Server.

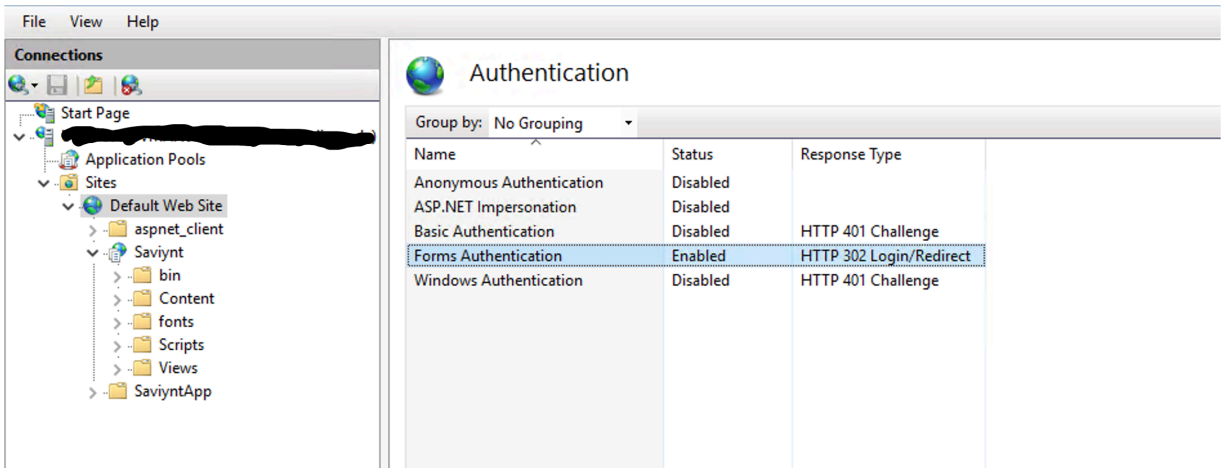
Note: In the web.config file add the following configuration under System.webServer to hide the physical path in the error response for the local users also.
update <httpErrors errorMode="Custom" />



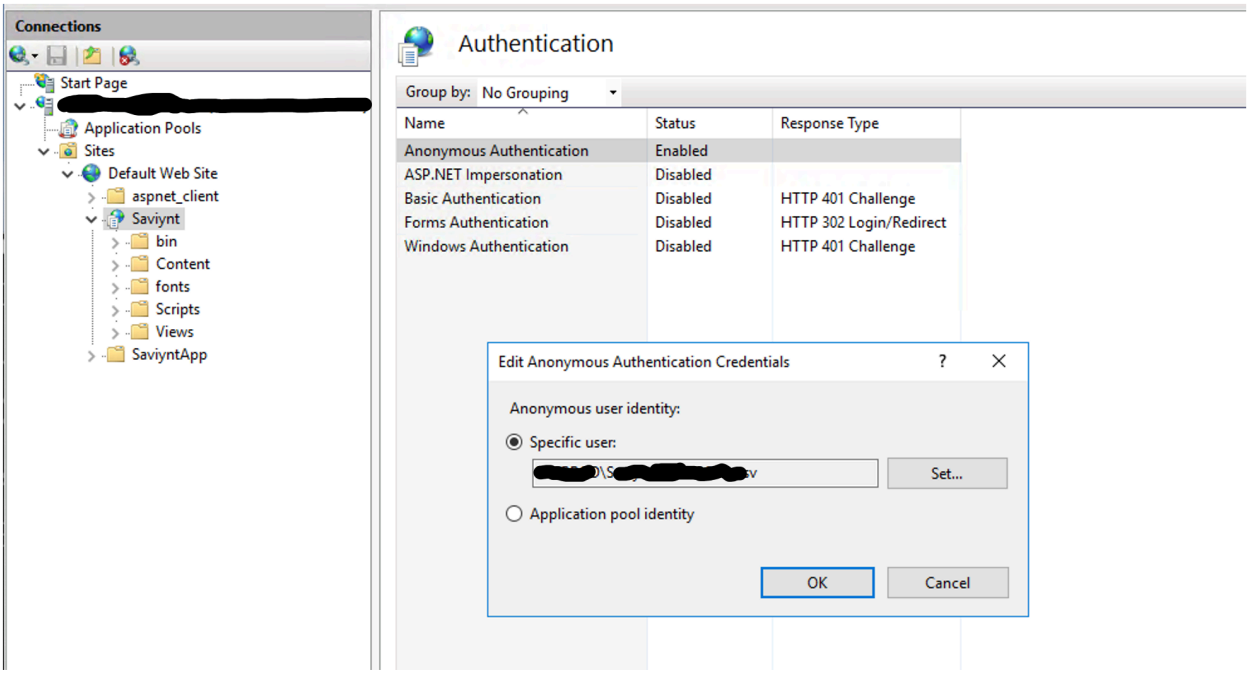
Server → Authentication → Forms Authentication enabled.



Default Web Site → Authentication → Forms Authentication enabled.



Saviynt App → Authentication → Anonymous Authentication enabled.



Updated web.config file in C:\inetpub\wwwroot\SaviyntApp\Saviynt Agent\WINPS with below values:

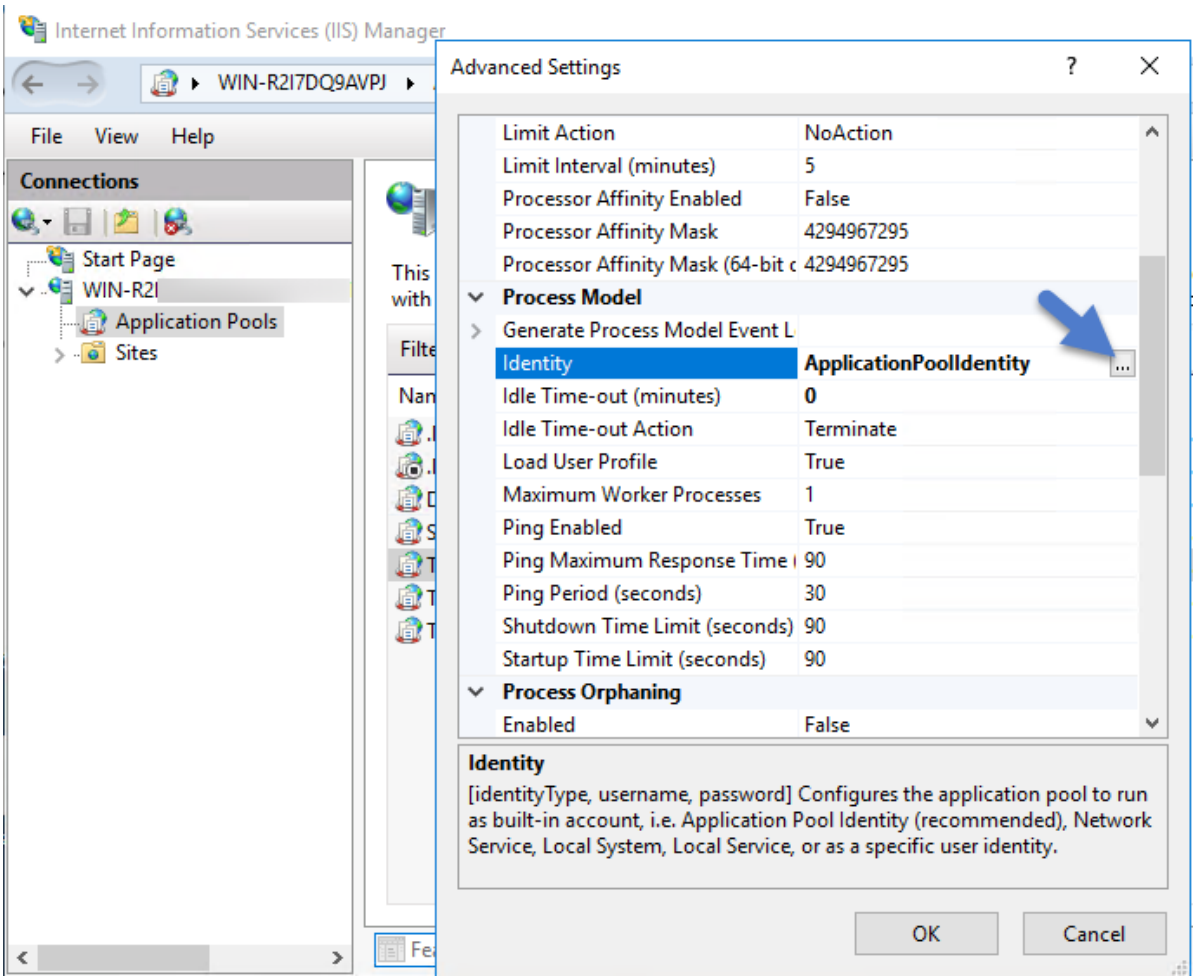
```
<appSettings>
    <add key="webpages:Version" value="3.0.0.0" />
    <add key="webpages:Enabled" value="false" />
    <add key="ClientValidationEnabled" value="true" />
    <add key="UnobtrusiveJavaScriptEnabled" value="true" />
   <add key="PSSleepTime" value="6000" />
   <add key="PSTimeoutHours" value="0" />
     <add key="PSTimeoutMinutes" value="20" />
     <add key="PSTimeoutSeconds" value="6000" />
  </appSettings>
```

5.2 Assigning Identity of Application Pool(s) in IIS

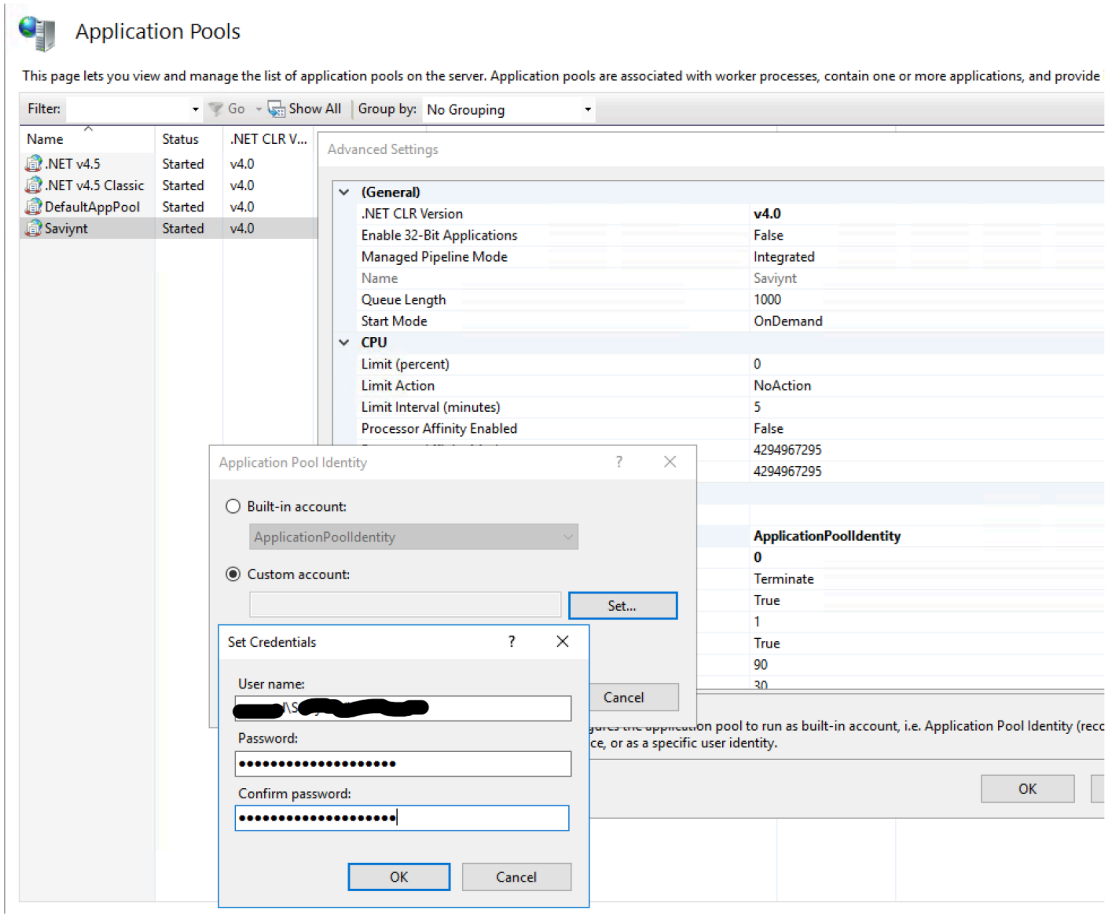
Note: This is required so that application pool uses service account for actions like starting powershell.

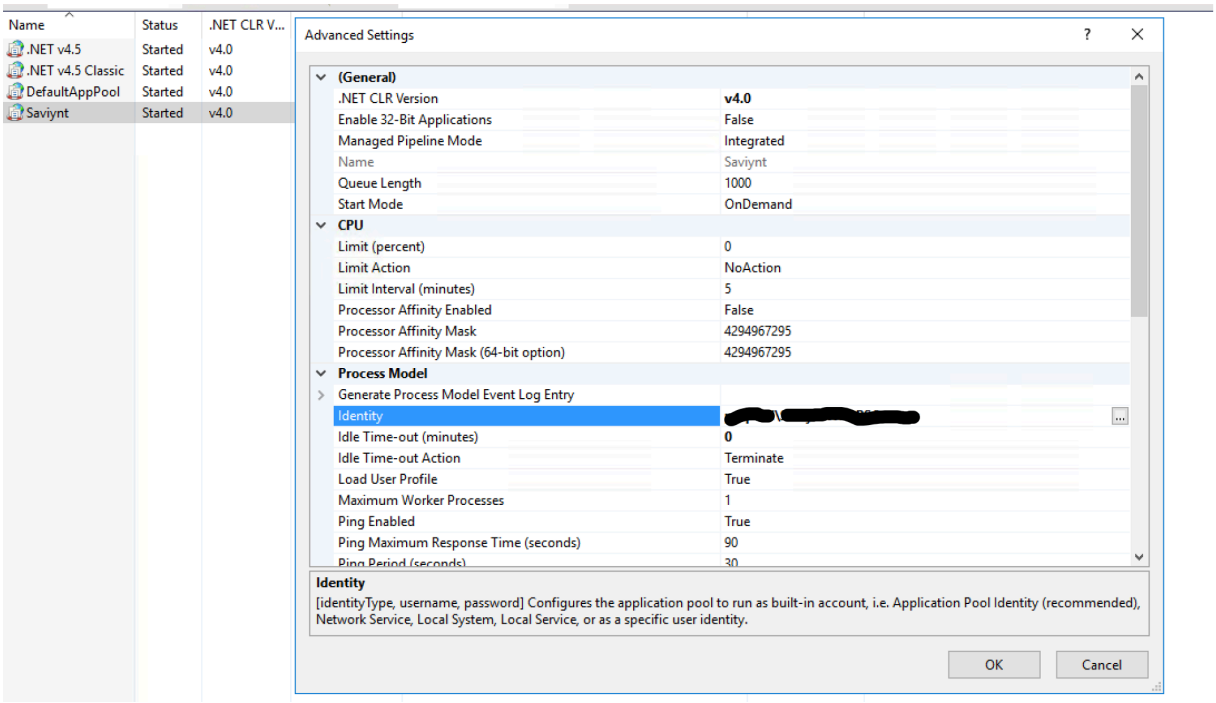
You must have IIS installed on your web server before completing these steps.

1. Open IIS on your web server (**Search > inetmgr**)
2. Locate the application pool(s) that your Thycotic product is using, right-click **Advanced Settings...** then the **Identity** box in the "Process Model" section, click the three dots on the right of the box.



3. Select the **Custom Account** radio button, click **Set**, enter your service account's name and password, and click **OK**.
    - **Note**: For Privilege Manager you will need to perform this step for multiple application pools.


Application Pools
This page lets you view and manage the list of application pools on the server. Application pools are associated with worker processes, contain one or more applications, and provide
Filter:
Go
Show All
Group by: No Grouping
Name
Status
.NET CLR V...
.NET v4.5
Started
v4.0
.NET v4.5 Classic
Started
v4.0
DefaultAppPool
Started
v4.0
Saviynt
Started
v4.0
Advanced Settings
(General)
.NET CLR Version
v4.0
Enable 32-Bit Applications
False
Managed Pipeline Mode
Integrated
Name
Saviynt
Queue Length
1000
Start Mode
OnDemand
CPU
Limit (percent)
0
Limit Action
NoAction
Limit Interval (minutes)
5
Processor Affinity Enabled
False
4294967295
4294967295
ApplicationPoolIdentity
0
Terminate
True
1
True
90
30
Application Pool Identity
Built-in account:
ApplicationPoolIdentity
Custom account:
Set...
Cancel
Set Credentials
User name:
Password:
Confirm password:
OK
Cancel
pool to run as built-in account, i.e. Application Pool Identity (recc
ce, or as a specific user identity.
OK


Name
Status
.NET CLR V...
.NET v4.5
Started
v4.0
.NET v4.5 Classic
Started
v4.0
DefaultAppPool
Started
v4.0
Saviynt
Started
v4.0
Advanced Settings
(General)
.NET CLR Version
v4.0
Enable 32-Bit Applications
False
Managed Pipeline Mode
Integrated
Name
Saviynt
Queue Length
1000
Start Mode
OnDemand
CPU
Limit (percent)
0
Limit Action
NoAction
Limit Interval (minutes)
5
Processor Affinity Enabled
False
Processor Affinity Mask
4294967295
Processor Affinity Mask (64-bit option)
4294967295
Process Model
Generate Process Model Event Log Entry
Identity
Idle Time-out (minutes)
0
Idle Time-out Action
Terminate
Load User Profile
True
Maximum Worker Processes
1
Ping Enabled
True
Ping Maximum Response Time (seconds)
90
Identity
[identityType, username, password] Configures the application pool to run as built-in account, i.e. Application Pool Identity (recommended), Network Service, Local System, Local Service, or as a specific user identity.
OK
Cancel

5.3 Test HTTPS Connection

1. Open Windows PowerShell in the Run dialog as administrator on the IIS Server to test locally

```
add-type @"
using System.Net;
using System.Security.Cryptography.X509Certificates;
public class TrustAllCertsPolicy : ICertificatePolicy {
public bool CheckValidationResult(
ServicePoint srvPoint, X509Certificate certificate,
WebRequest request, int certificateProblem) {
return true;
}
}
"@
[System.Net.ServicePointManager]::CertificatePolicy = New-Object TrustAllCertsPolicy

$Body = @{SCRIPT = "Test-NetConnection <hostname.preprod.net>"}

Invoke-RestMethod https://<hostname.preprod.net>/Saviynt/PS/ExecutePSScript -Method Post -
Body $Body
```

5.4 Connection Parameters

The connector uses the following connection parameters to connect to the target application:

Parameter	Description
URL	Specify the path to invoke SaviyntApp deployed in the Windows IIS Server Syntax: http://<hostname>/<ApplicationName>/PS/ExecutePSScript Example: http://WIN-2KRUL98QLS1/FIMDemo/PS/ExecutePSScript https://hostname.preprod.net:443/Saviynt/PS/ExecutePSScript
USERNAME	Specify the username for the Windows IIS Server authentication. Eg: PREPROD\TestAccount-PSConn.sv

PASSWORD	Specify the password to connect to the Windows IIS Server.

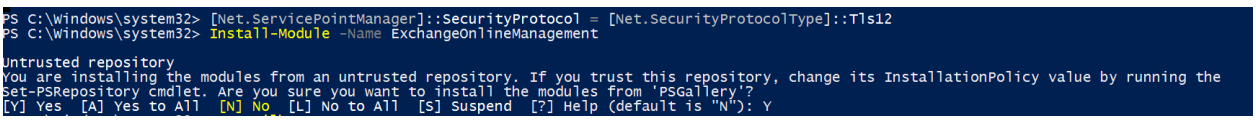
TESTCONNECTIONJSON	{
	"TESTCOMMAND": "SCRIPT=Test-NetConnection <hostname>",
	"RESPONSE": {
	"OBJECTTOREAD": "get(0).PingSucceeded",
	"SUCCESSVALUE": "TRUE"
	}
	}

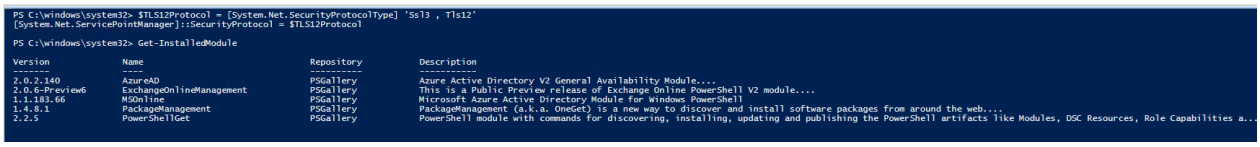
5.5 Install Exchange Office 365, AzureAD, MSOnline Modules

The connector uses the following connection

```
  $TLS12Protocol = [System.Net.SecurityProtocolType] 'Ssl3 , Tls12'
[System.Net.ServicePointManager]::SecurityProtocol = $TLS12Protocol
Register-PSRepository -Default
Install-Module PowershellGet -Force
Get-PSRepository
Get-InstalledModule
Install-Module AzureAd
Install-Module MSOnline
Install-Module -Name ExchangeOnlineManagement
Get-InstalledModule

Find-Module ExchangeOnlineManagement -AllVersions -AllowPrerelease
# Install latest EXO v2 for Compliance
Install-Module -Name ExchangeOnlineManagement -RequiredVersion 2.0.6-Preview6 -
AllowPrerelease
```





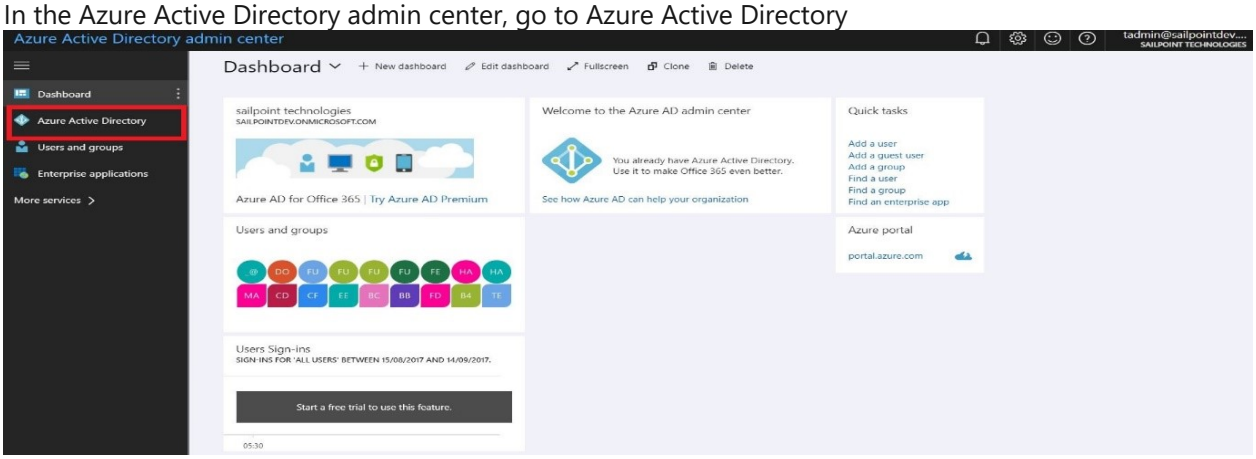
6.0 Azure AD Permissions for executing powershell from windows management server

6.1 App-only authentication for unattended scripts

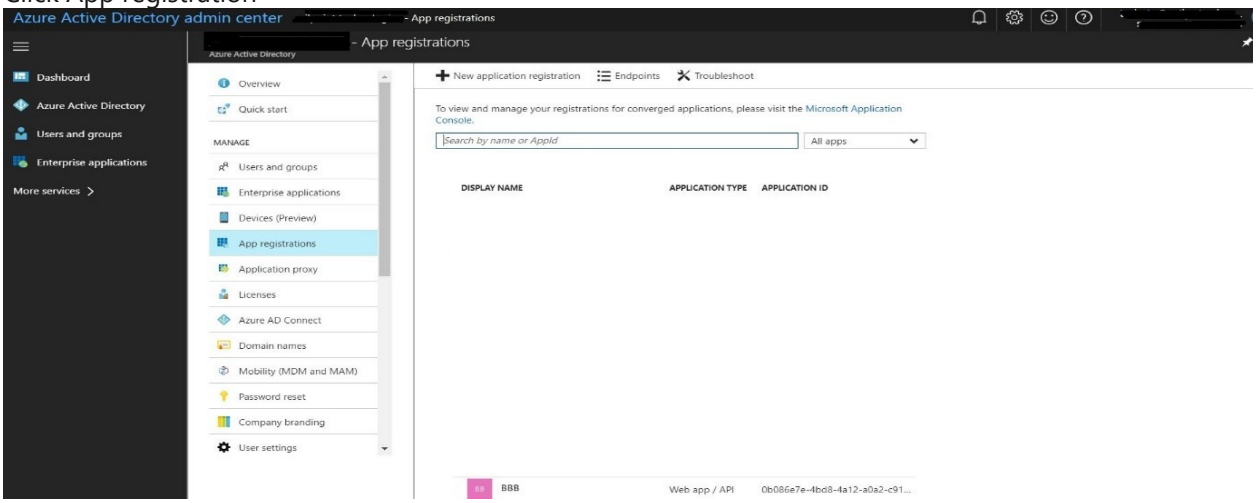
Step-1: Creating a New Azure AD Application via the Azure UI

If you prefer to use the Azure UI to create the application, you can follow these steps to create a new Azure AD application:

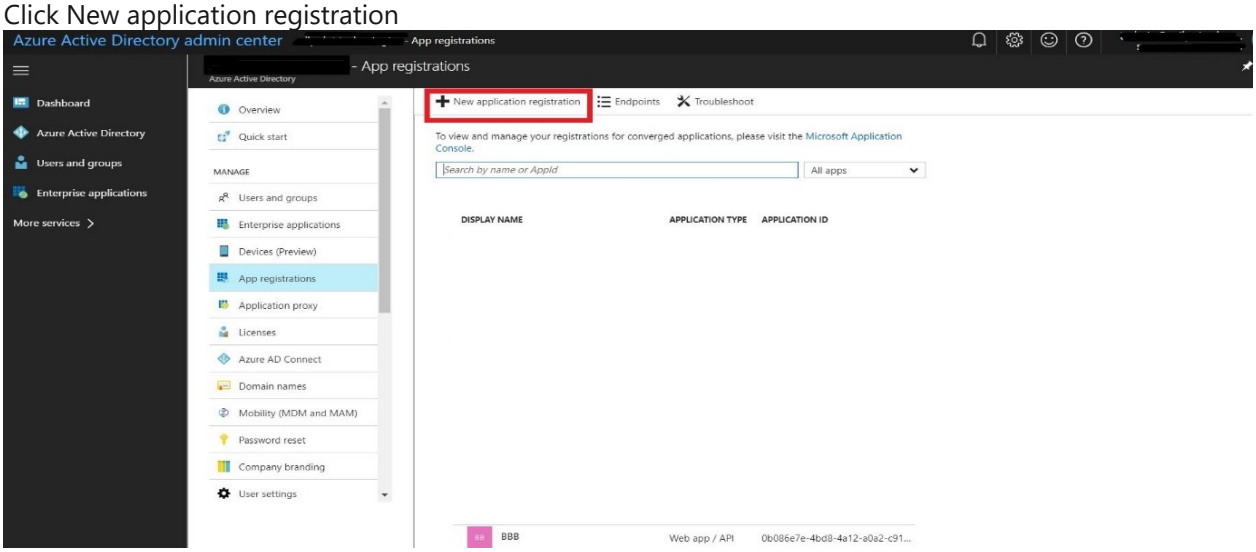
1. In the Azure Active Directory admin center, go to Azure Active Directory



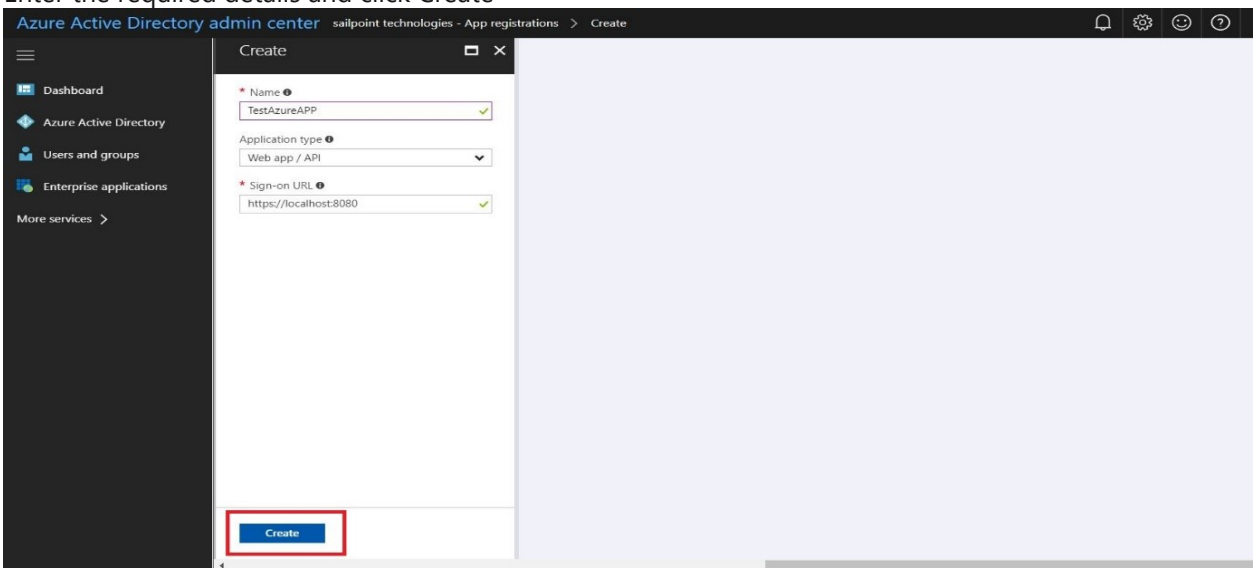
2. Click App registration



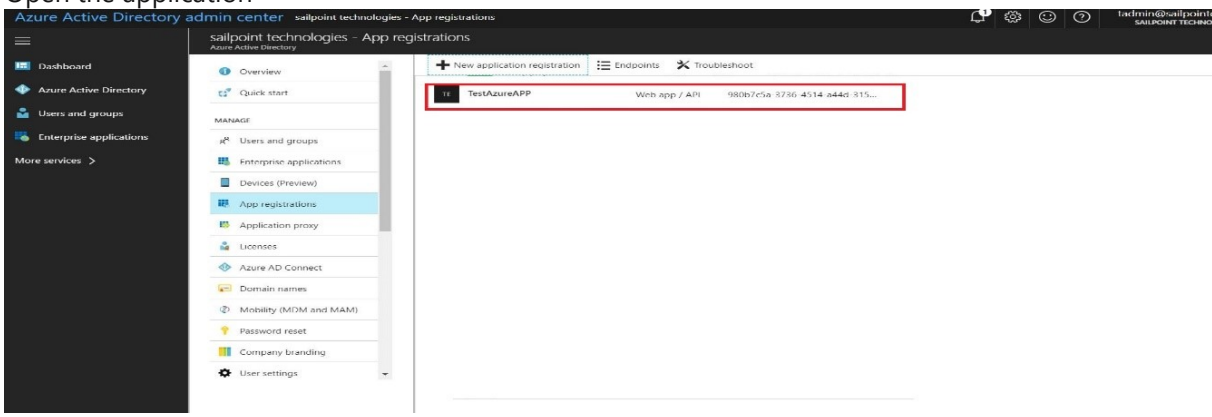
3. Click New application registration



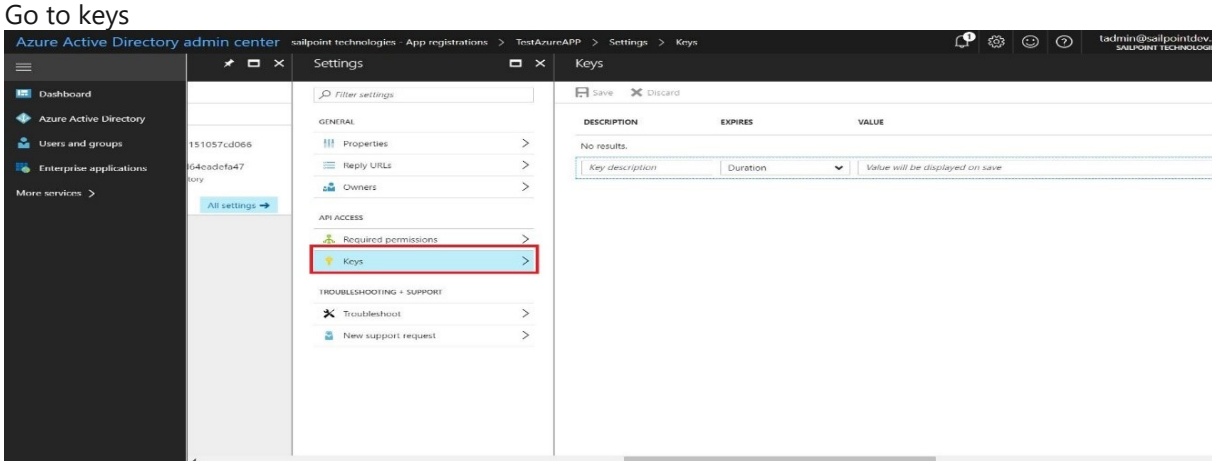
4. Enter the required details and click Create



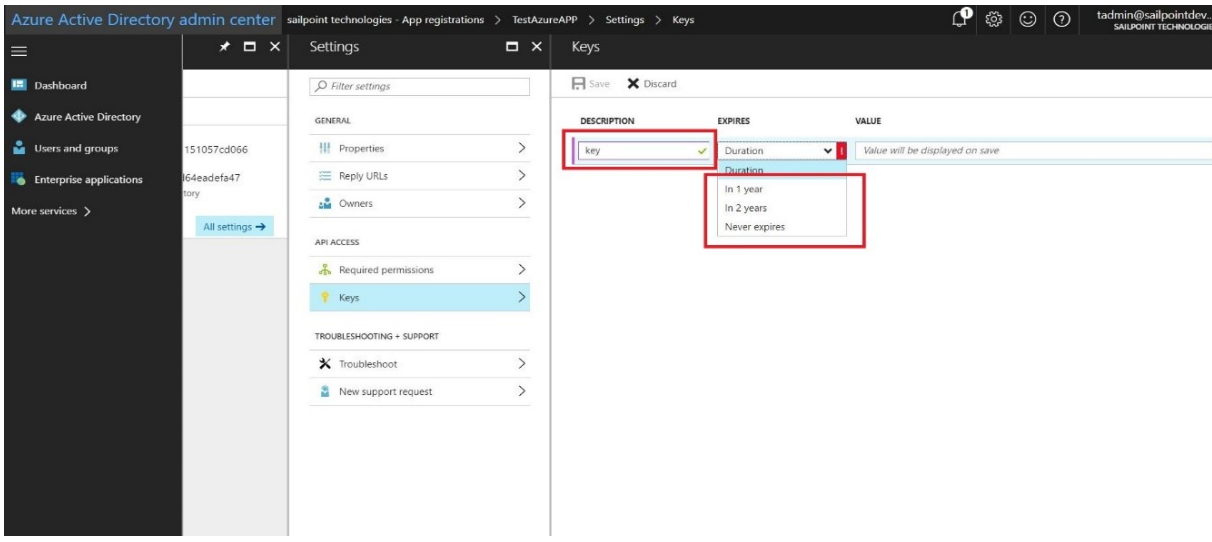
5. Retrieve the Application ID (Client ID) and Key (Secret Key)
    1. Open the application



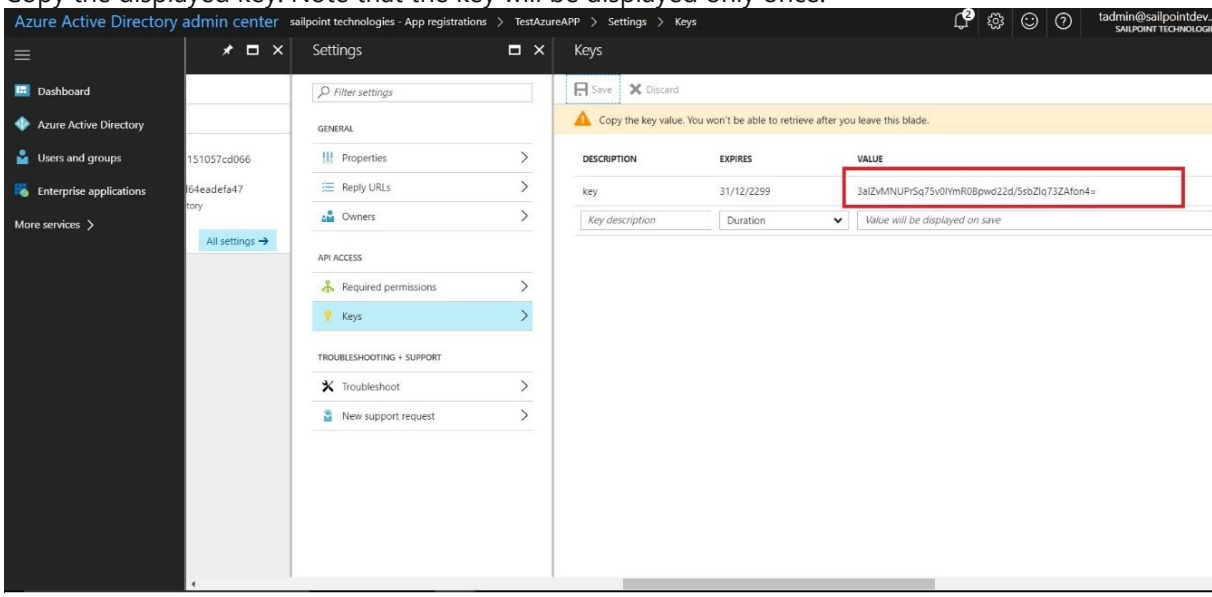
2. Go to keys



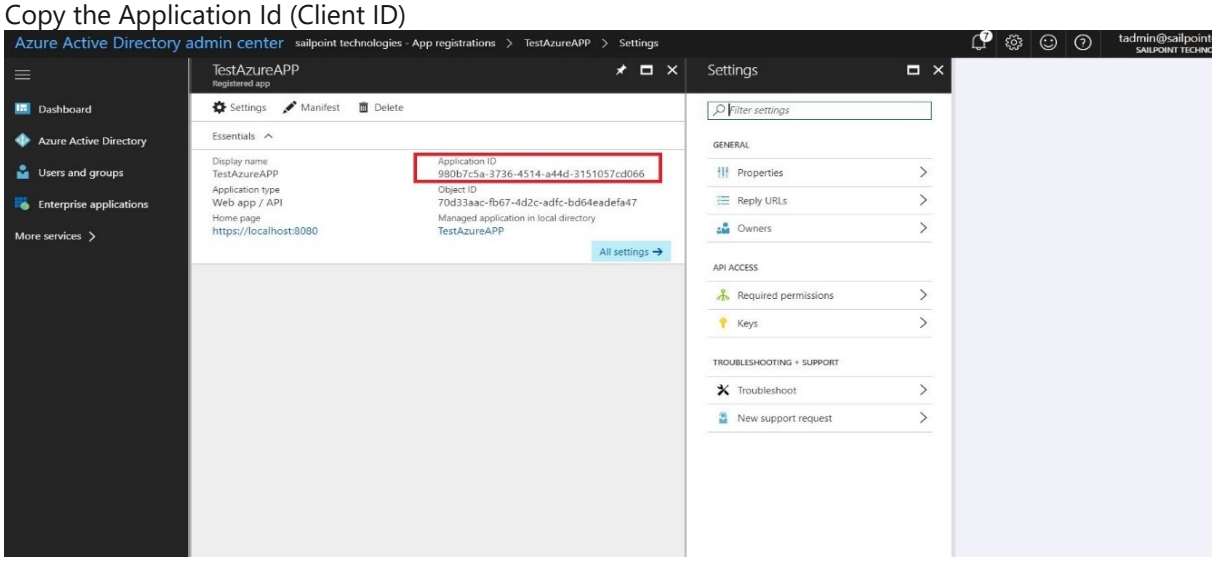
3. Provide a key description and validity duration and click Save



4. Copy the displayed key. Note that the key will be displayed only once.



5. Copy the Application Id (Client ID)

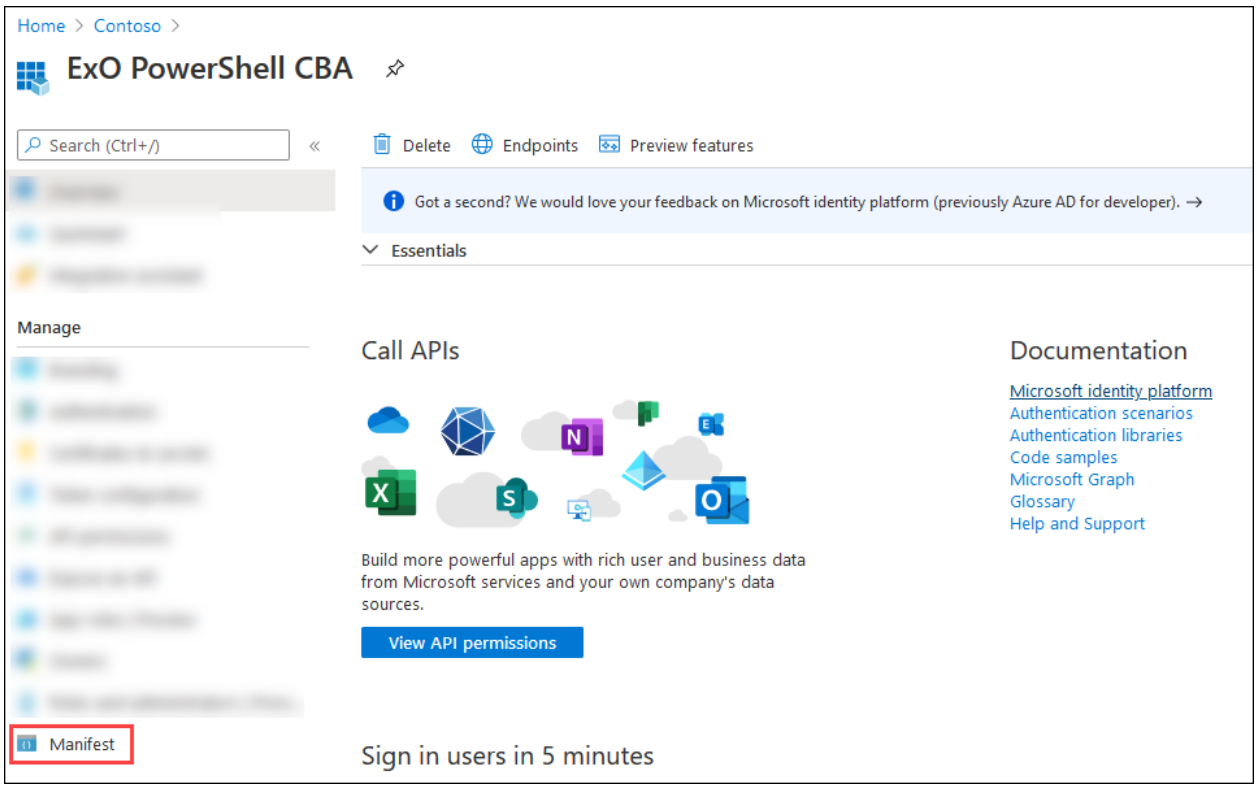


Step 2: Assign API permissions to the application

Note

The procedures in this section replace any default permissions that were automatically configured for the new app. The app doesn't need the default permissions that were replaced.

1. On the app page under **Management**, select **Manifest**.



2. On the **Manifest** page that opens, find the `requiredResourceAccess` entry (on or about line 44).

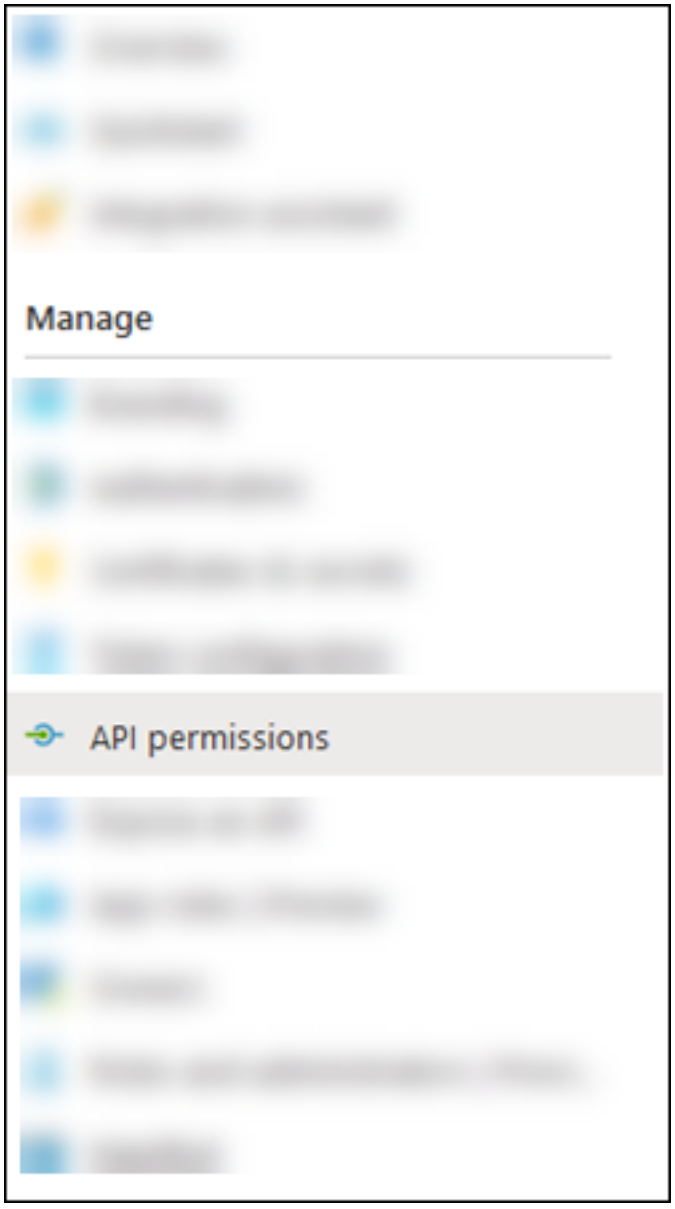
Modify the `resourceAppId`, `resourceAccess`, `id`, and `type` values as shown in the following code snippet:

JSONCopy

```
"requiredResourceAccess": [
   {
      "resourceAppId": "00000002-0000-0ff1-ce00-000000000000",
      "resourceAccess": [
         {
            "id": "dc50a0fb-09a3-484d-be87-e023b12c6440",
            "type": "Role"
         }
      ]
   }
],
```

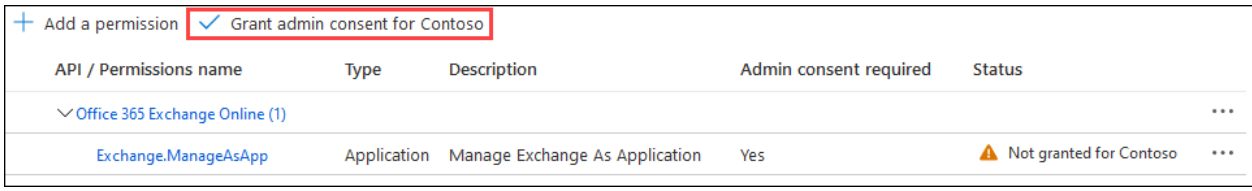
When you're finished, click **Save**.

3. Still on the **Manifest** page, under **Management**, select **API permissions**.



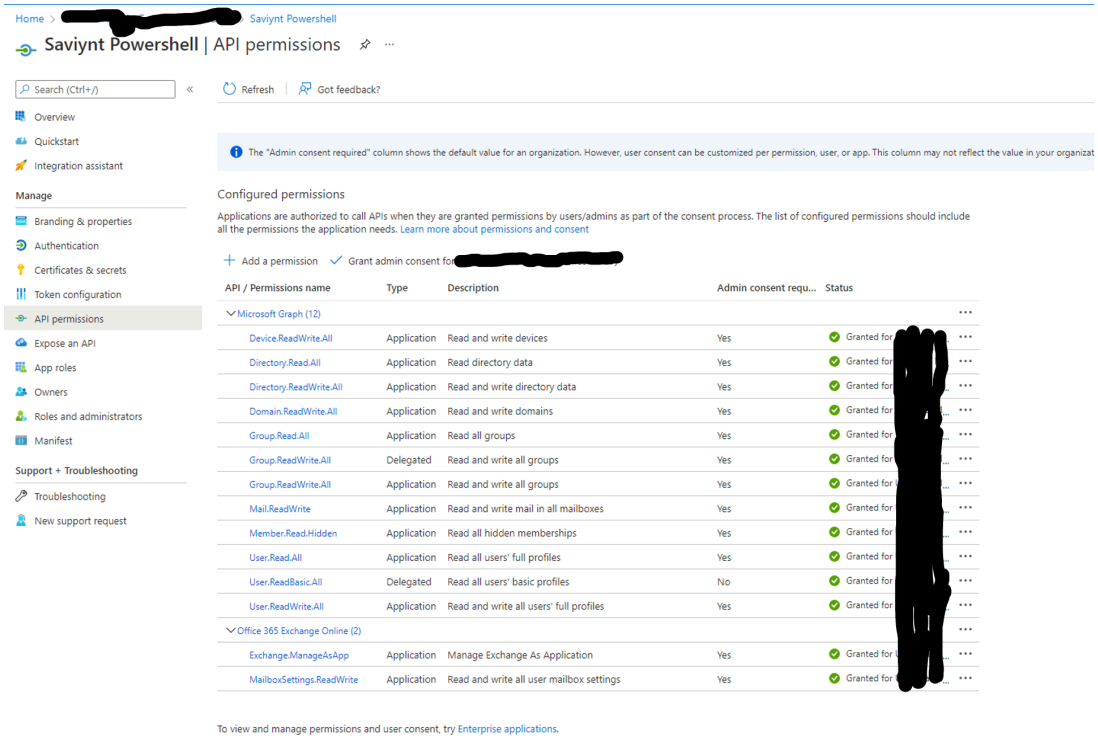
On the **API permissions** page that opens, do the following steps:

- **API / Permissions name**: Verify the value **Exchange.ManageAsApp** is shown.
- **Status**: The current incorrect value is **Not granted for <Organization>**, and this value needs to be changed.



Select **Grant admin consent for <Organization>**, read the confirmation dialog that opens, and then click **Yes**.

The **Status** value should now be **Granted for <Organization>**.



4. Close the current **API permissions** page (not the browser tab) to return to the **App registrations** page. You'll use it in an upcoming step.

Step-3 Generate Certificate and Assign to the Application created in Azure for Saviynt

$cert = New-SelfSignedCertificate -Subject "CN={saviyntAADTestTenant}" -CertStoreLocation "Cert:\CurrentUser\My" -KeyExportPolicy Exportable -KeySpec Signature -KeyLength 2048 -KeyAlgorithm RSA -HashAlgorithm SHA256 ## Replace {certificateName}

Export-Certificate -Cert $cert -FilePath "C:\Saviynt\SaviyntDev\{saviyntAADTestTenant}.cer" ## Specify your preferred location and replace {certificateName}

$mypwd = ConvertTo-SecureString -String "cert-password-tosafeit" -Force -AsPlainText ## Replace {myPassword}

Export-PfxCertificate -Cert $cert -FilePath "C:\Saviynt\SaviyntDev\{saviyntAADPK}.pfx" -Password $mypwd ## Specify your preferred location and replace {privateKeyName}

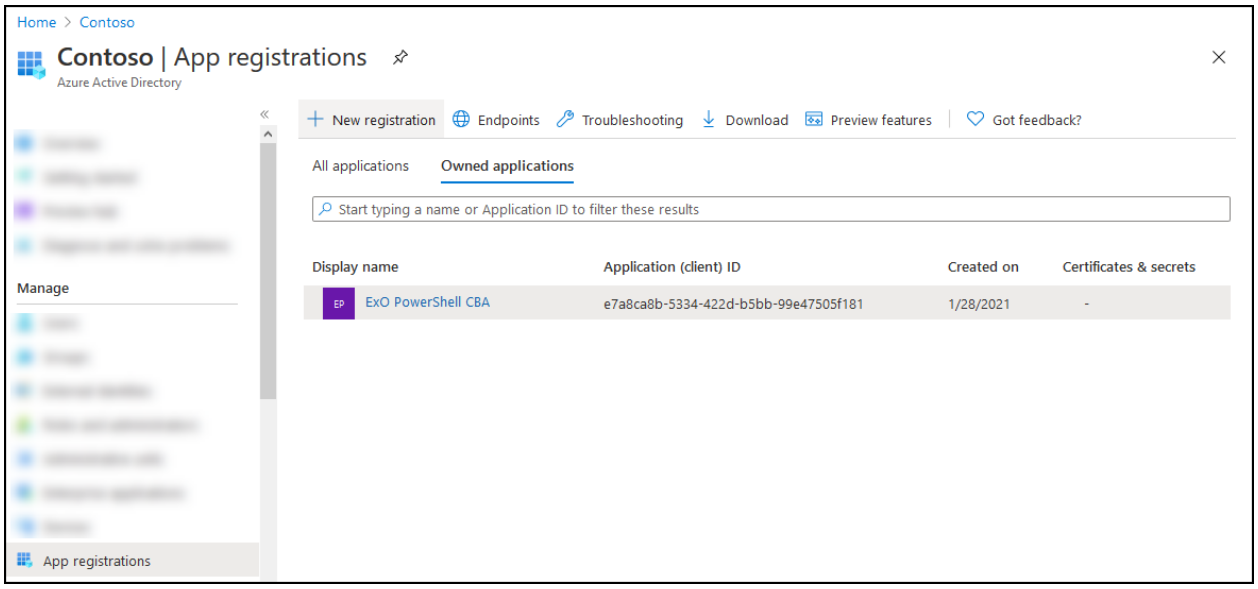
1.1 Step 4: Attach the certificate to the Azure AD application

After you register the certificate with your application, you can use the private key (`.pfx` file) or the thumbprint for authentication.

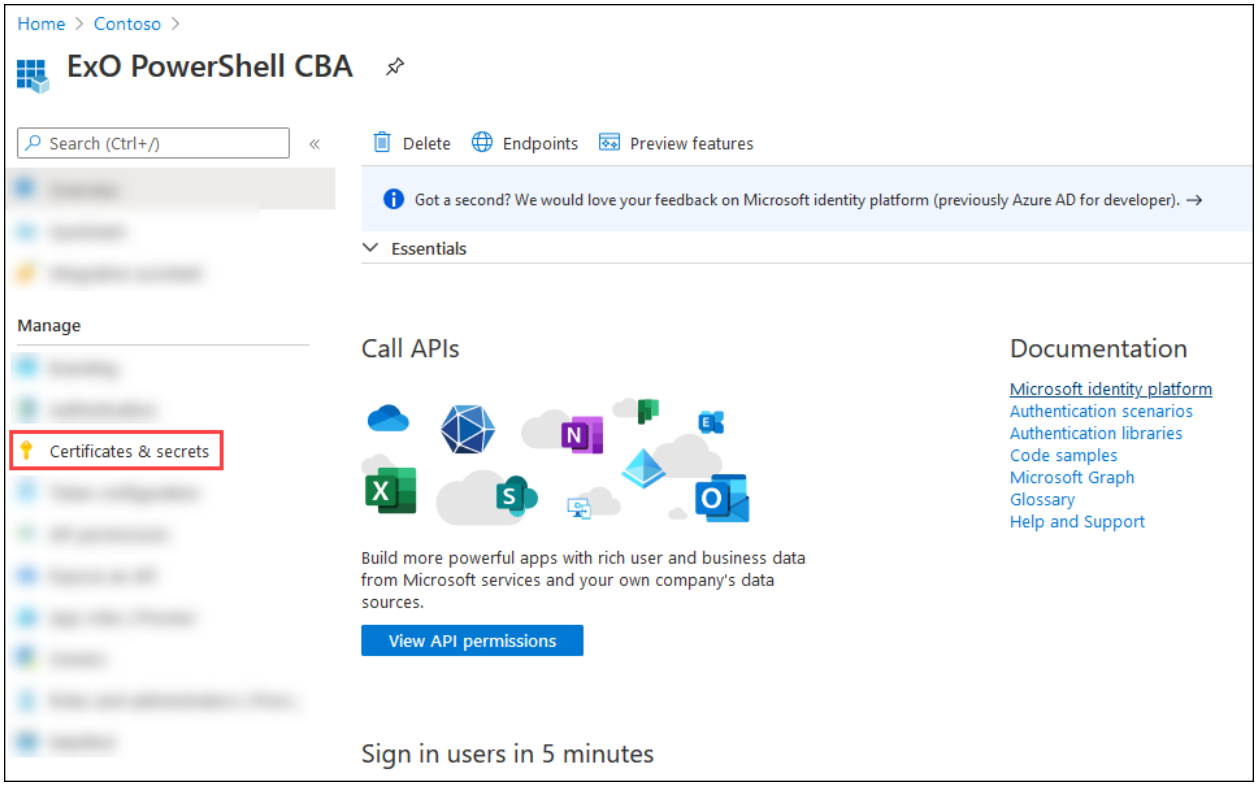
1. On the **Apps registration** page from the end of Step 2, select your application.

   If you need to get back to **Apps registration** page, do the following steps:

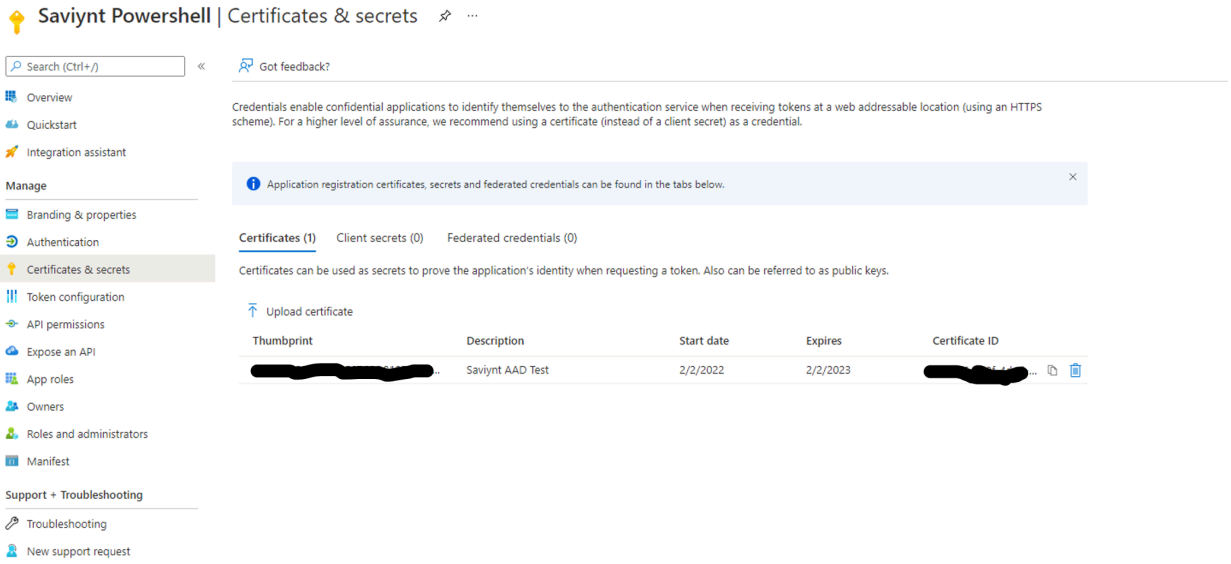
   1. Open the Azure AD portal at https://portal.azure.com/.
   2. Under **Manage Azure Active Directory**, click **View**.
   3. Under **Manage**, select **App registrations**.



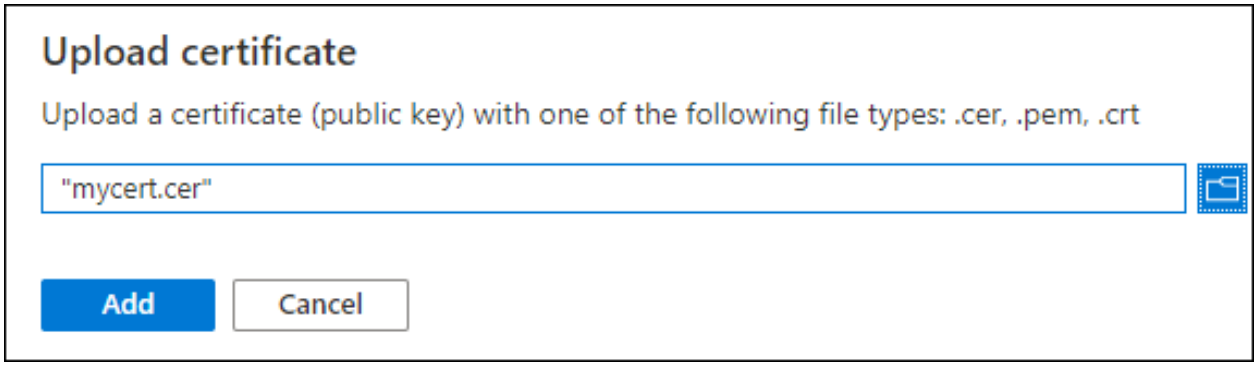
2. On the application page that opens, under **Manage**, select **Certificates & secrets**.



3. On the **Certificates & secrets** page that opens, click **Upload certificate**.

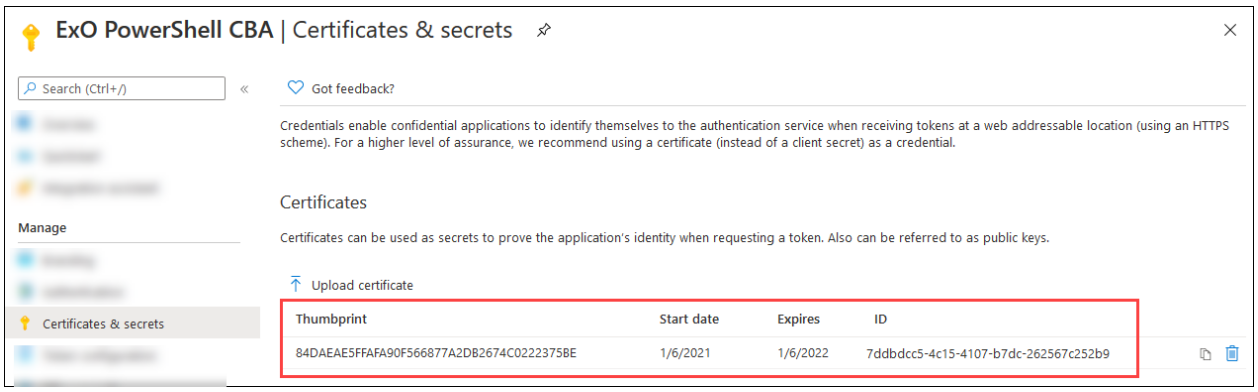


In the dialog that opens, browse to the self-signed certificate (`.cer` file) that you created in Step 3.



When you're finished, click **Add**.

The certificate is now shown in the **Certificates** section.



4. Close the current **Certificates & secrets** page, and then the **App registrations** page to return to the main https://portal.azure.com/ page. You'll use it in the next step.

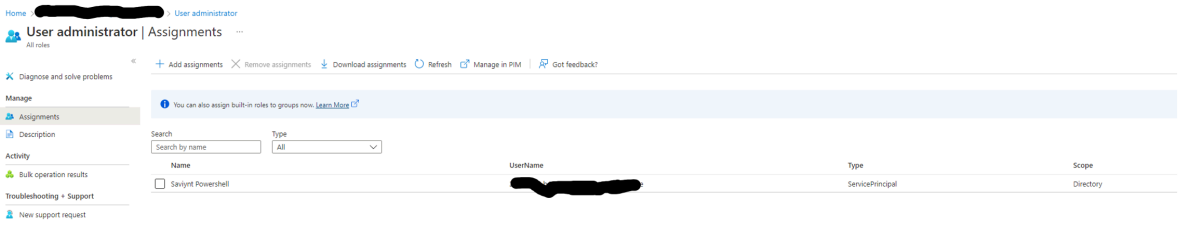
1.2 Step 5: Assign Azure AD roles to the application

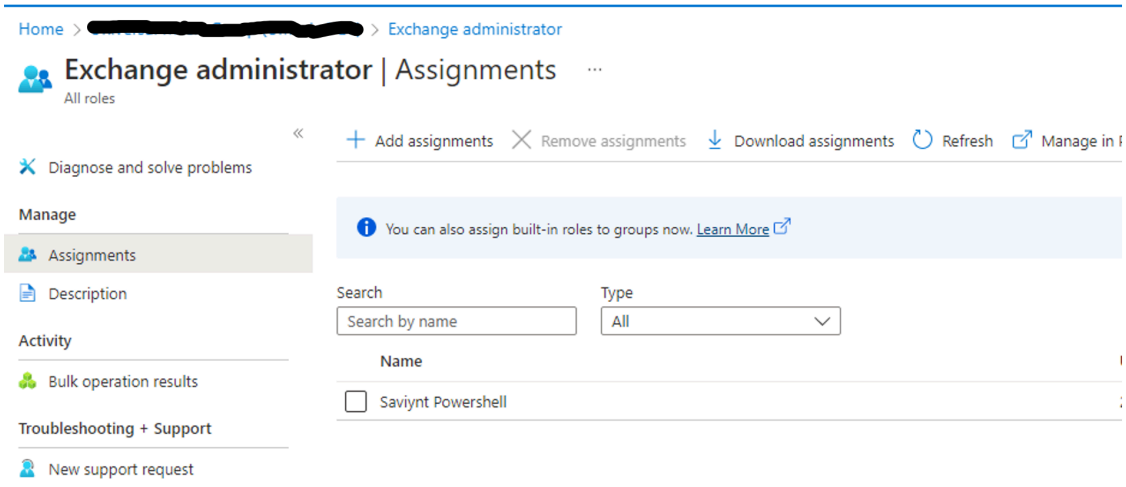
Azure AD has more than 50 admin roles available. The supported roles are described in the following table:

Role	Exchange Online PowerShell	User Management
User Administrator		✓
Exchange Administrator*	✓	

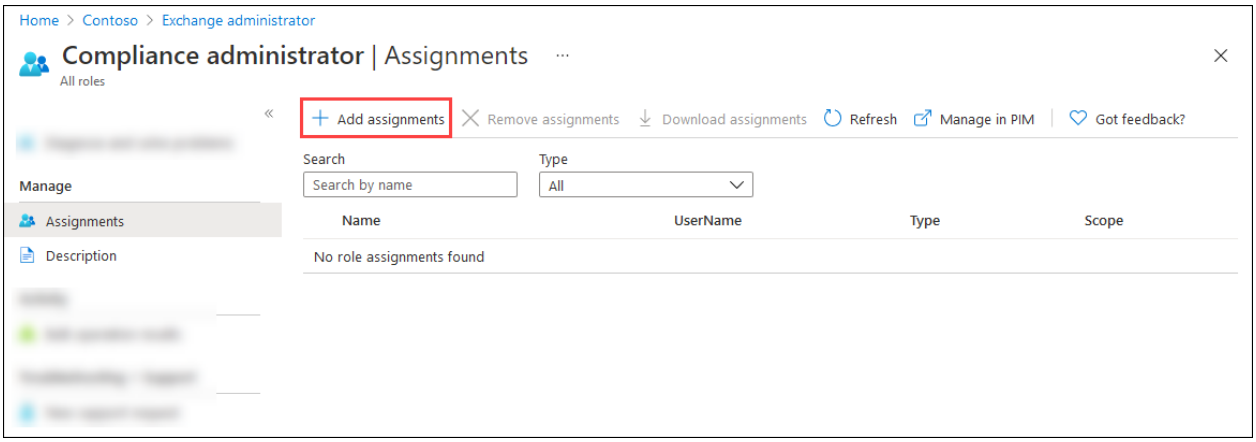
* The Exchange Administrator roles provide the required permissions for any task in Exchange Online PowerShell.

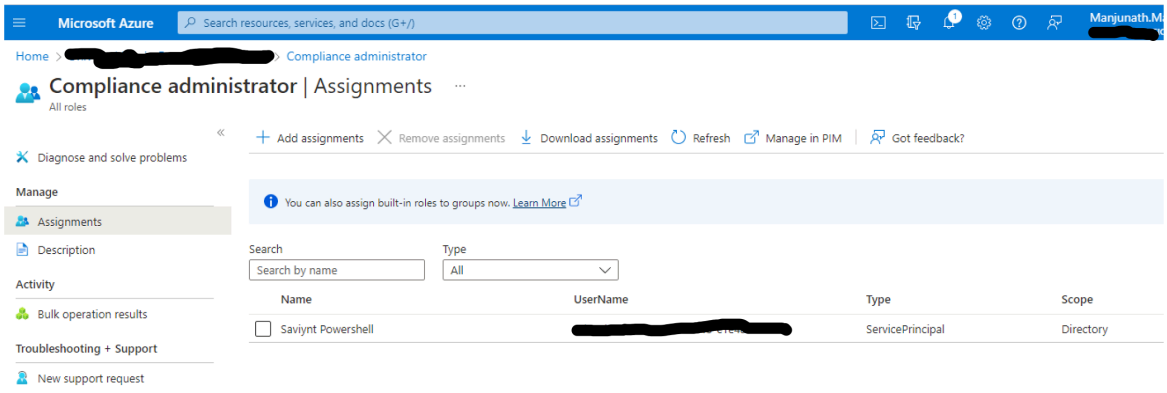
* The User Administrator roles provide the required permissions for any task in Azure AD PowerShell.





Security & Compliance Center PowerShell:

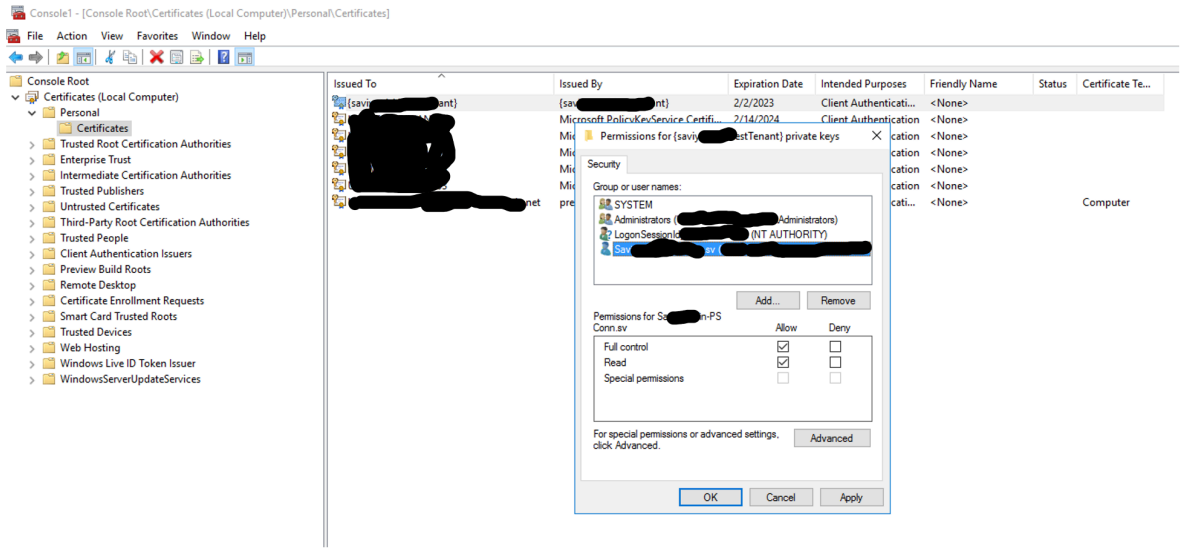




Step-6: Allow ServiceAccount to read the certificate

Note: If this step is not performed, then we see error: Connect-AzureAD : One or more errors occurred. "KeySet does not exist".

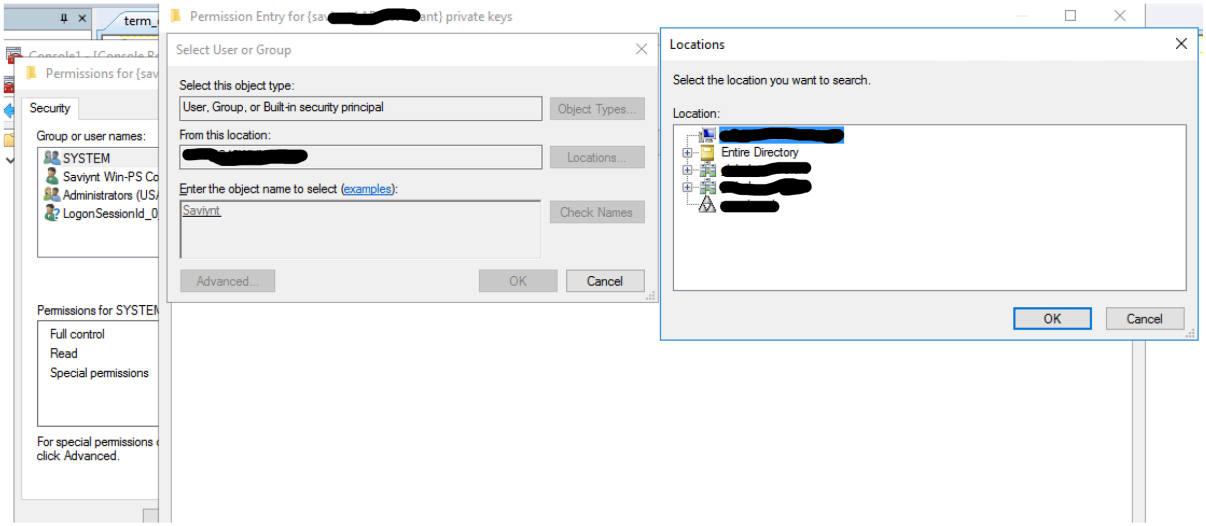
1. Open certificate snap-in in MMC: WIN + R -> type mmc -> File -> Add/Remove Snap-in -> Add Certificates (Computer Account).
2. Find your certificate -> Right click and choose All Tasks/Manage Private Keys
3. Grant Permission for SERVICE Account user

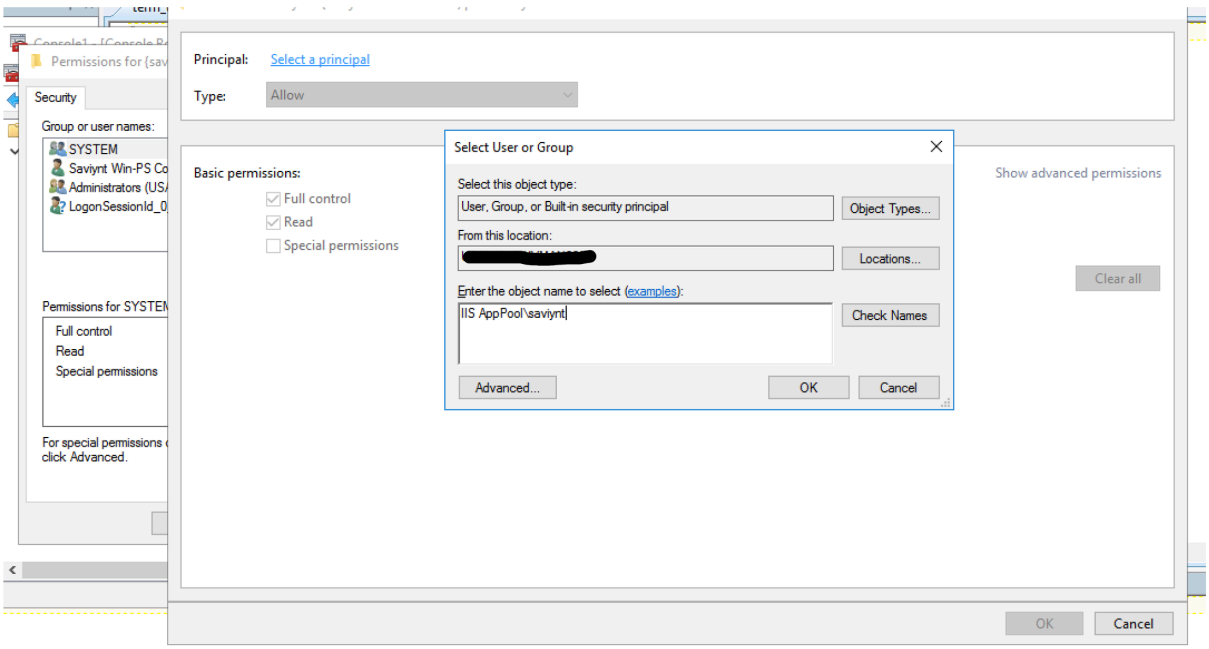


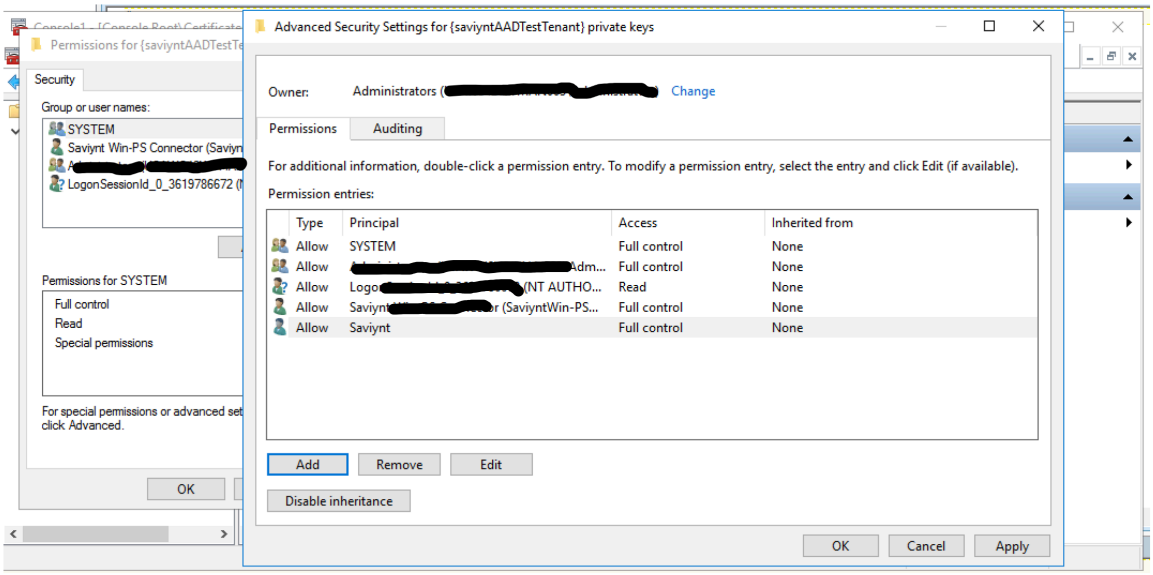
We should also provide permissions for Saviynt application pool to access the certificate.

1. Create / Purchase certificate. Make sure it has a private key.
2. Import the certificate into the "Local Computer" account. Best to use Certificates MMC. **Make sure to check "Allow private key to be exported"**
3. Based upon which, IIS 7.5 Application Pool's identity use one of the following.

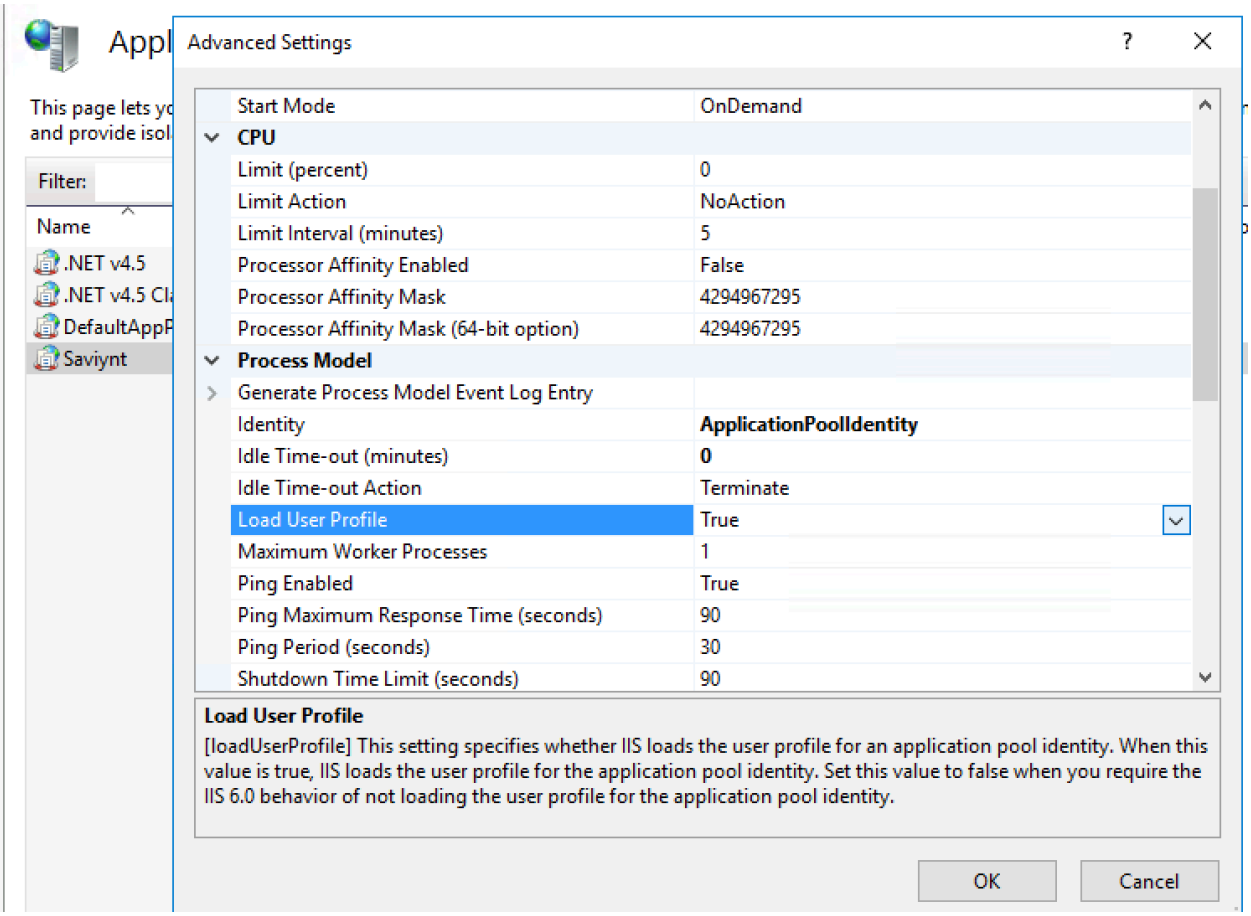
- IIS 7.5 Website is running under ApplicationPoolIdentity.
- Using Certificates MMC,
    1. Select “From this Location” as that local server. By default it would be preprod or global domain.
    2. added "IIS AppPool\Saviynt" to Full Trust on certificate in "Local Computer\Personal". Saviynt is the "*AppPoolName*" of your application pool.







Also, in IIS Application pool, set Load user Profile to True.

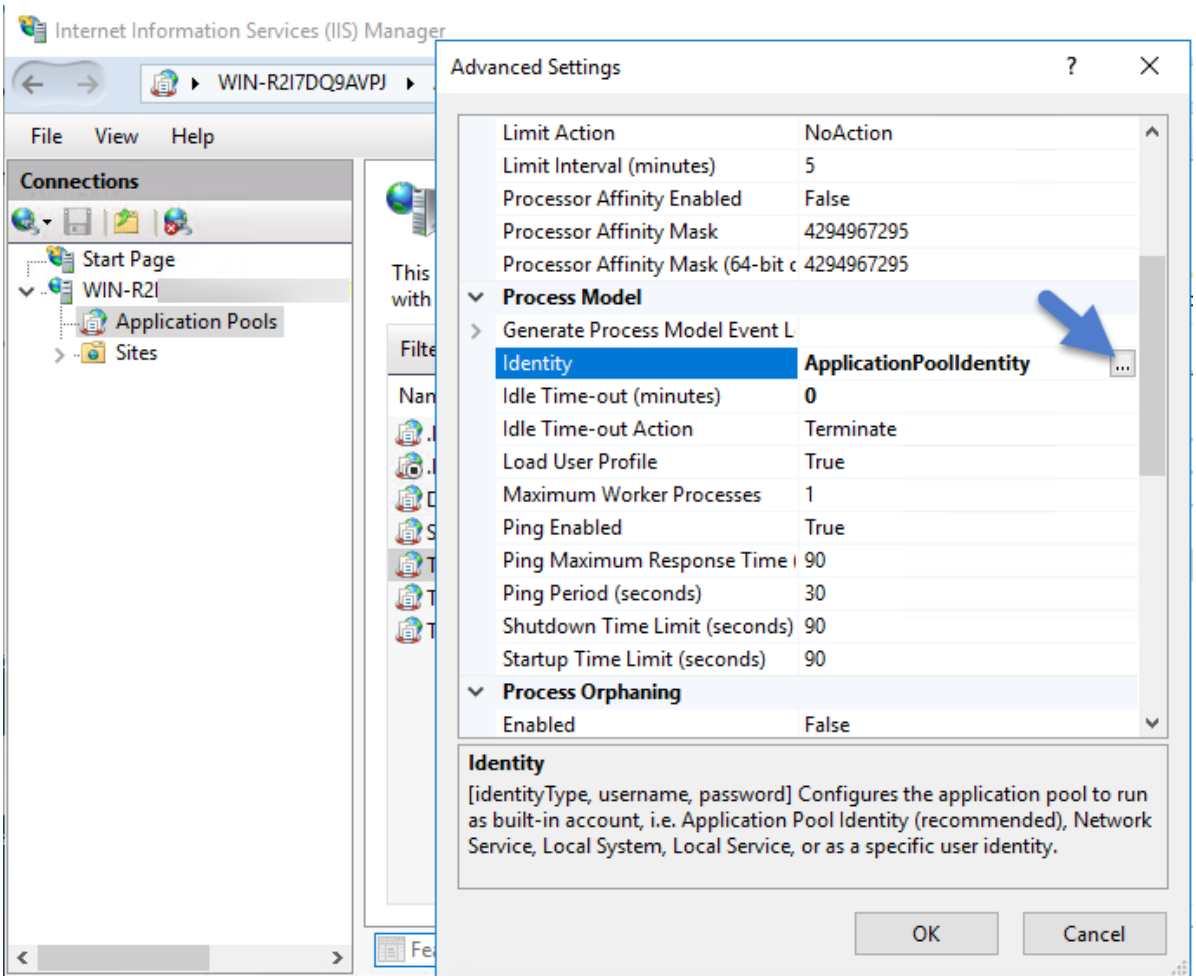


2 Assigning Identity of Application Pool(s) in IIS

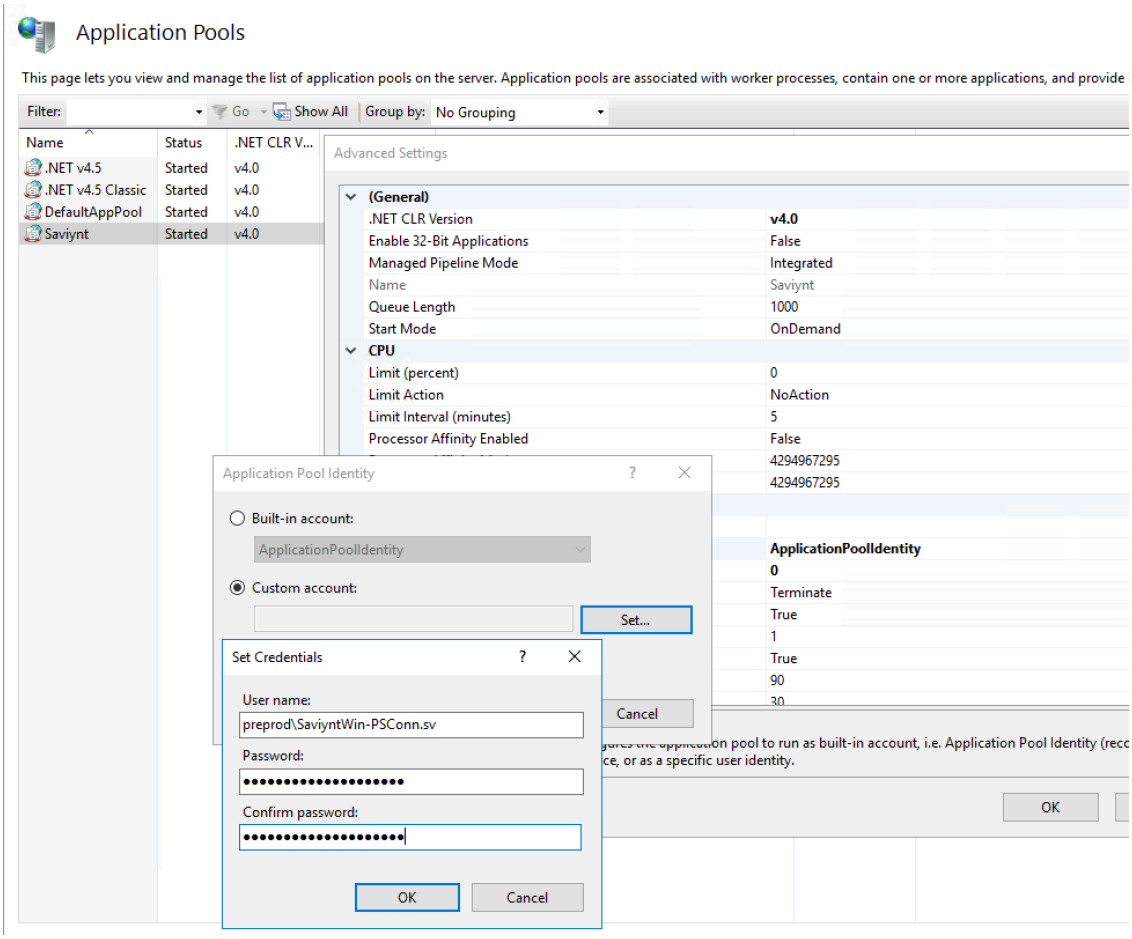
Note: This is required so that application pool uses service account for actions like starting powershell.

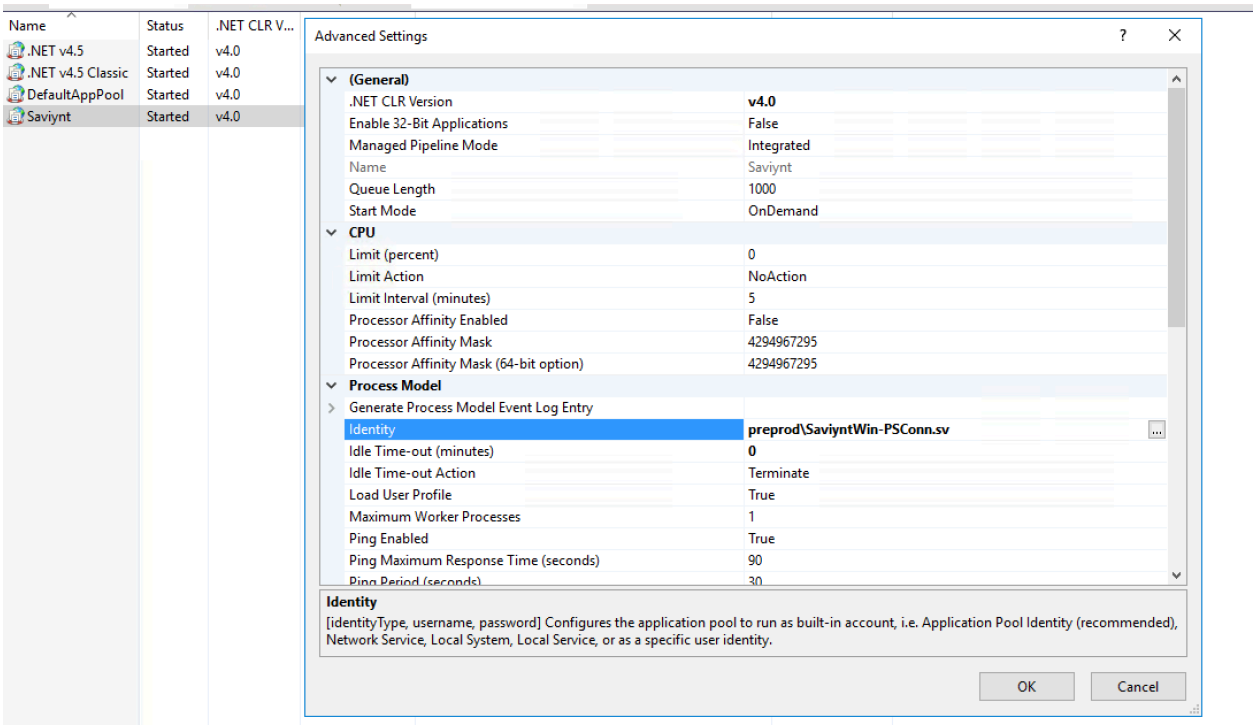
You must have IIS installed on your web server before completing these steps.

4. Open IIS on your web server (**Search > inetmgr**)
5. Locate the application pool(s) that your Thycotic product is using, right-click **Advanced Settings...** then the **Identity** box in the "Process Model" section, click the three dots on the right of the box.



6. Select the **Custom Account** radio button, click **Set**, enter your service account's name and password, and click **OK**.
    - **Note**: For Privilege Manager you will need to perform this step for multiple application pools.


Application Pools
This page lets you view and manage the list of application pools on the server. Application pools are associated with worker processes, contain one or more applications, and provide
Filter:
Go
Show All
Group by: No Grouping
Name
Status
.NET CLR V...
.NET v4.5 Started v4.0
.NET v4.5 Classic Started v4.0
DefaultAppPool Started v4.0
Saviynt Started v4.0
Advanced Settings
(General)
.NET CLR Version v4.0
Enable 32-Bit Applications False
Managed Pipeline Mode Integrated
Name Saviynt
Queue Length 1000
Start Mode OnDemand
CPU
Limit (percent) 0
Limit Action NoAction
Limit Interval (minutes) 5
Processor Affinity Enabled False
4294967295
4294967295
ApplicationPoolIdentity
0
Terminate
True
1
True
90
30
Application Pool Identity
Built-in account:
ApplicationPoolIdentity
Custom account:
Set...
Cancel
Set Credentials
User name:
preprod\SaviyntWin-PSConn.sv
Password:
Confirm password:
OK
Cancel
pool to run as built-in account, i.e. Application Pool Identity (recc
ce, or as a specific user identity.
OK


Name
Status
.NET CLR V...
.NET v4.5 Started v4.0
.NET v4.5 Classic Started v4.0
DefaultAppPool Started v4.0
Saviynt Started v4.0
Advanced Settings
(General)
.NET CLR Version v4.0
Enable 32-Bit Applications False
Managed Pipeline Mode Integrated
Name Saviynt
Queue Length 1000
Start Mode OnDemand
CPU
Limit (percent) 0
Limit Action NoAction
Limit Interval (minutes) 5
Processor Affinity Enabled False
Processor Affinity Mask 4294967295
Processor Affinity Mask (64-bit option) 4294967295
Process Model
Generate Process Model Event Log Entry
Identity preprod\SaviyntWin-PSConn.sv
Idle Time-out (minutes) 0
Idle Time-out Action Terminate
Load User Profile True
Maximum Worker Processes 1
Ping Enabled True
Ping Maximum Response Time (seconds) 90
Ping Period (seconds) 30
Identity
[identityType, username, password] Configures the application pool to run as built-in account, i.e. Application Pool Identity (recommended), Network Service, Local System, Local Service, or as a specific user identity.
OK
Cancel

Contact us

Manjunath Madiraju
Manager, Technical Architect

Email mmadiraju@kpmg.com

Ken Dunbar
Engagement Director

Email kbdunbar@kpmg.com