Deploy ASP.NET app to a virtual machine on Azure

This example shows how you can deploy a new ASP.NET MVC app from Visual Studio Team Services to a virtual machine in Azure **using Machine Groups**

## **Pre-requisites**

You will need a virtual machine with IIS features enabled and the website with the app pool should exist.

If you would like to create a fresh VM in Azure using a template, refer [Appendix](#_Appendix_2).

For this document, we are using “**Default Web Site**” and ‘**Default application pool’**. You may create your own websites and app pool as well.

## **Code**

Carry out the following steps to create a simple .NET MVC app to deploy.

Or skip directly to ‘[Create machine group](#_Create_machine_group)’ section, and use GitHub as an artifact feature for Release Management and link the [prebuilt artifact](https://github.com/RoopeshNair/AspNetWebApp) as a source.

1. Create a new **ASP.NET Web Application** project in Visual Studio, targeting .NET Framework 4.5.
2. Choose the **MVC** project type and leave the other settings at their defaults. Skip “Host in the cloud” option in the template, if present.
3. Save the solution and commit it to your chosen repository.

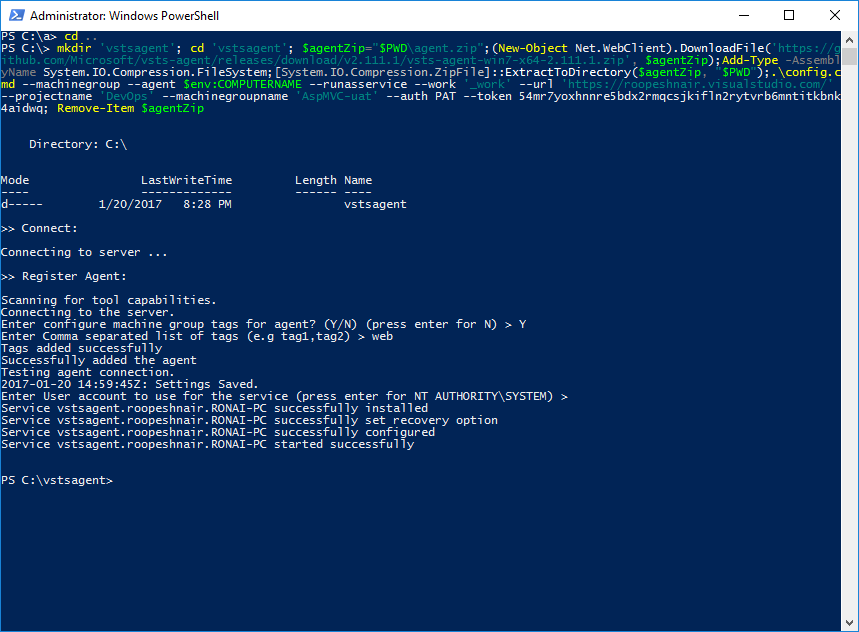
## **Build**

Carry out the following steps to build the sample web application in the **Build** hub.

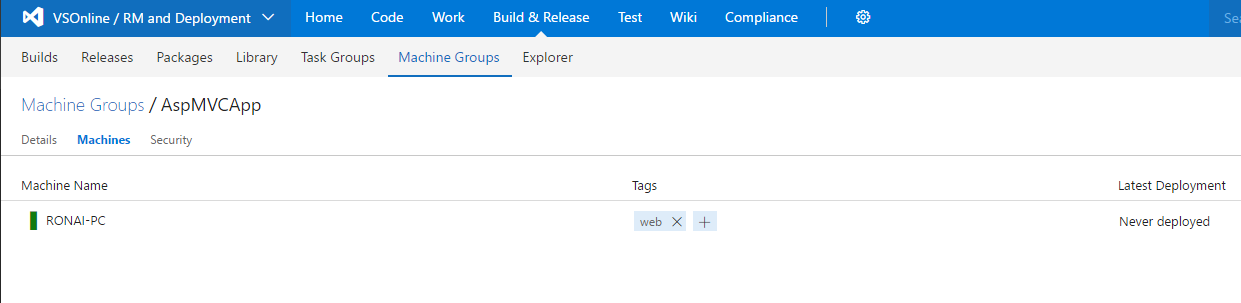
1. Open the **Builds** tab of the **Build & Release** hub and choose **+ New** to create a new build definition.
2. In the **Build** tab of the **Create new build definition** dialog, select **ASP.NET (PREVIEW)** template and choose **Next**.
3. Select the repository where you committed your app and choose **Create**.
4. Save the build definition.
5. Queue a new build and verify that the artifact containing the zip file and the website are correctly published.

## **Create machine group**

1. Open your Team Services in your web browser.
2. Open the **Machine Groups** tab of the Build & Release hub and choose “+ **Machine Group**” to create a new Machine Group
3. Important: Provide a name for the Machine Group under the “**Details**” tab and “**Save**” (save step is important – we have fixed this in the new UI)
4. From the **Register machines using command line** section, click “**Regenerate script with PAT**” button
5. Copy the script by selecting the cmdlet displayed in the machine group details console or click “Copy” icon. Have the script saved locally.
6. Login to the target virtual machine where you want to deploy the web app.
7. Open **Administrator Powershell command prompt**.
8. Paste and run the script to register the machine with this machine group.
9. When prompted to configure machine group tags for agent, enter **‘Y’** and enter ‘**web’** sans-quotes.
10. At the next prompt for user account, press enter to accept the defaults
11. You should see that the service vstsagent.<account>.<computername> started successfully. Refer to screenshot below for reference.



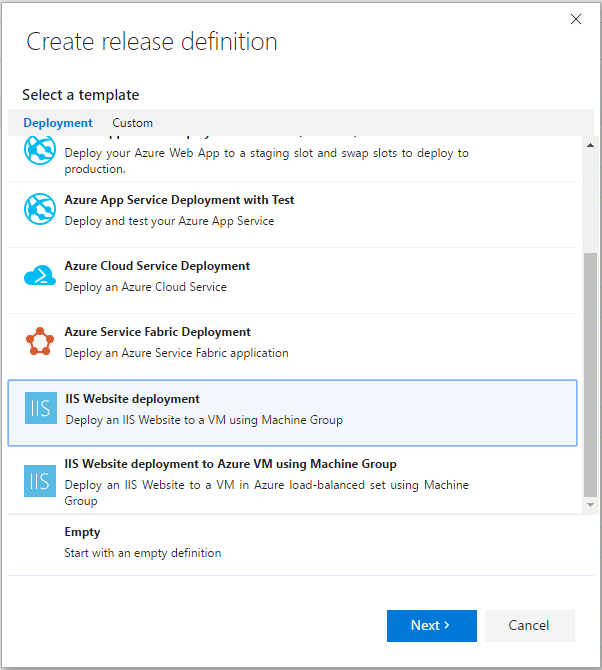
1. In the machine groups page click “**Machines**” tab and verify that the agent is up and running. In case you don’t see the tag info, **refresh** the page once.



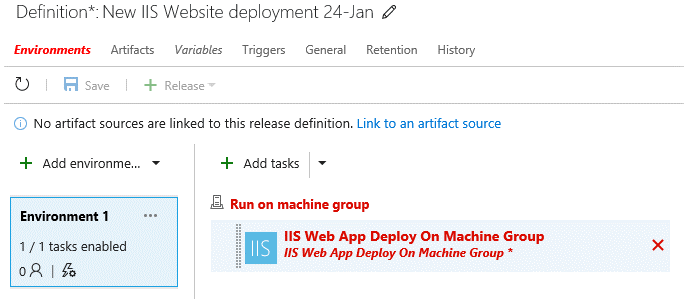
## **Deploy**

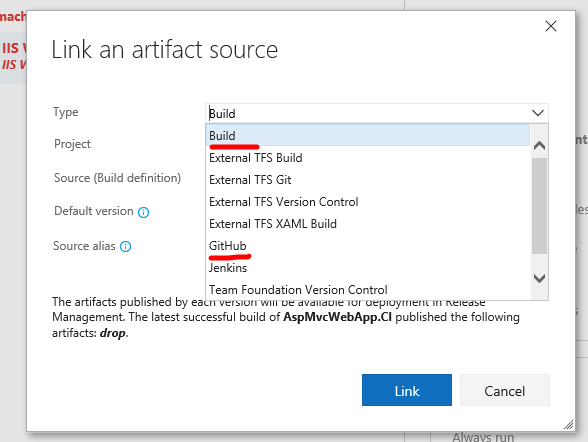
Carry out the following steps to deploy the app to the target server.

1. Open the **Releases** tab of the **Build & Release** hub and choose the plus icon to create a new release definition.
2. In the **Create release definition** dialog, select the **IIS Website deployment** template and choose **Next**.
3. In the next page, click ‘**Choose later’** for artifacts section and click “**Create**”. This creates a new release definition with one default environment. (ignore, queues option in the wizard, we don’t need, and we will remove it from the wizard)

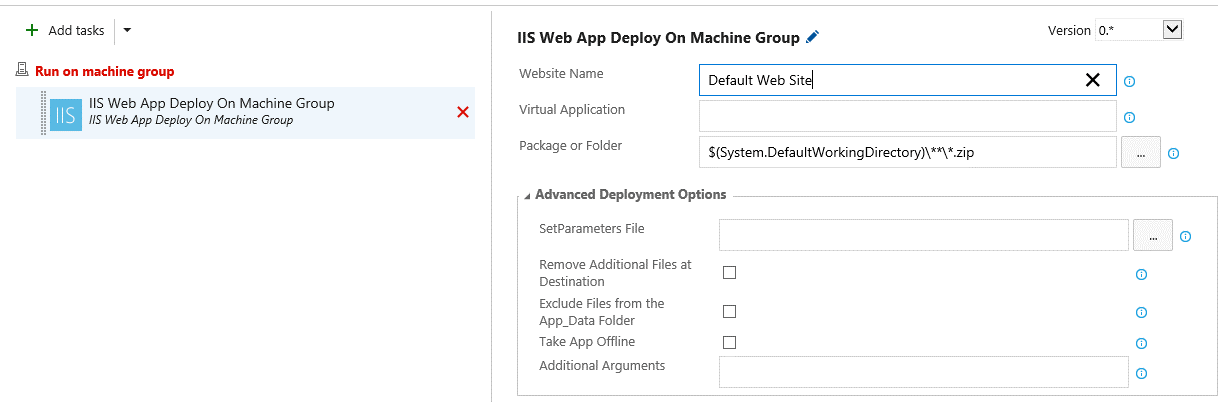


1. Click on ‘Link to an artifact source’, -
   * If you are coming from “Code, Build” steps, select the build definition and click link.
   * Or if you are coming directly from ‘Create machine group’ step and planning to use ‘GitHub’ as an artifact source, fork the [GitHub repo](https://github.com/RoopeshNair/AspNetWebApp) to your github account and select the forked repo as artifact source type: “Github” and click link.

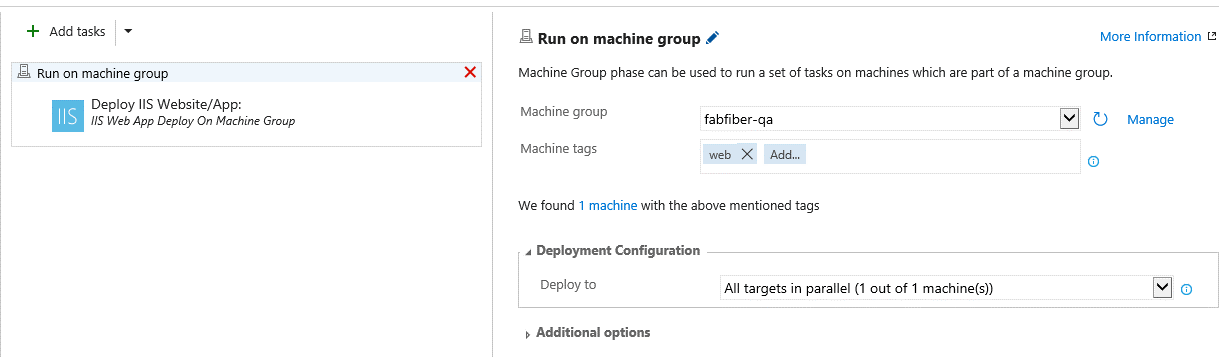




1. Configure the **Deploy IIS Website/App** task by entering the website name, for example enter “Default Web Site”



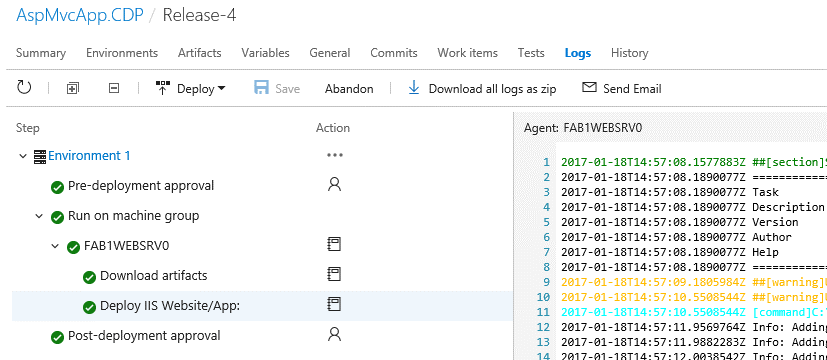
1. Choose the **Run on machine group** phase properties panel, select the machine group you created, where you installed and registered the agent earlier in this example.
2. Click on Add tag in “Machines Tags” input field. Select “web” tag from drop down.



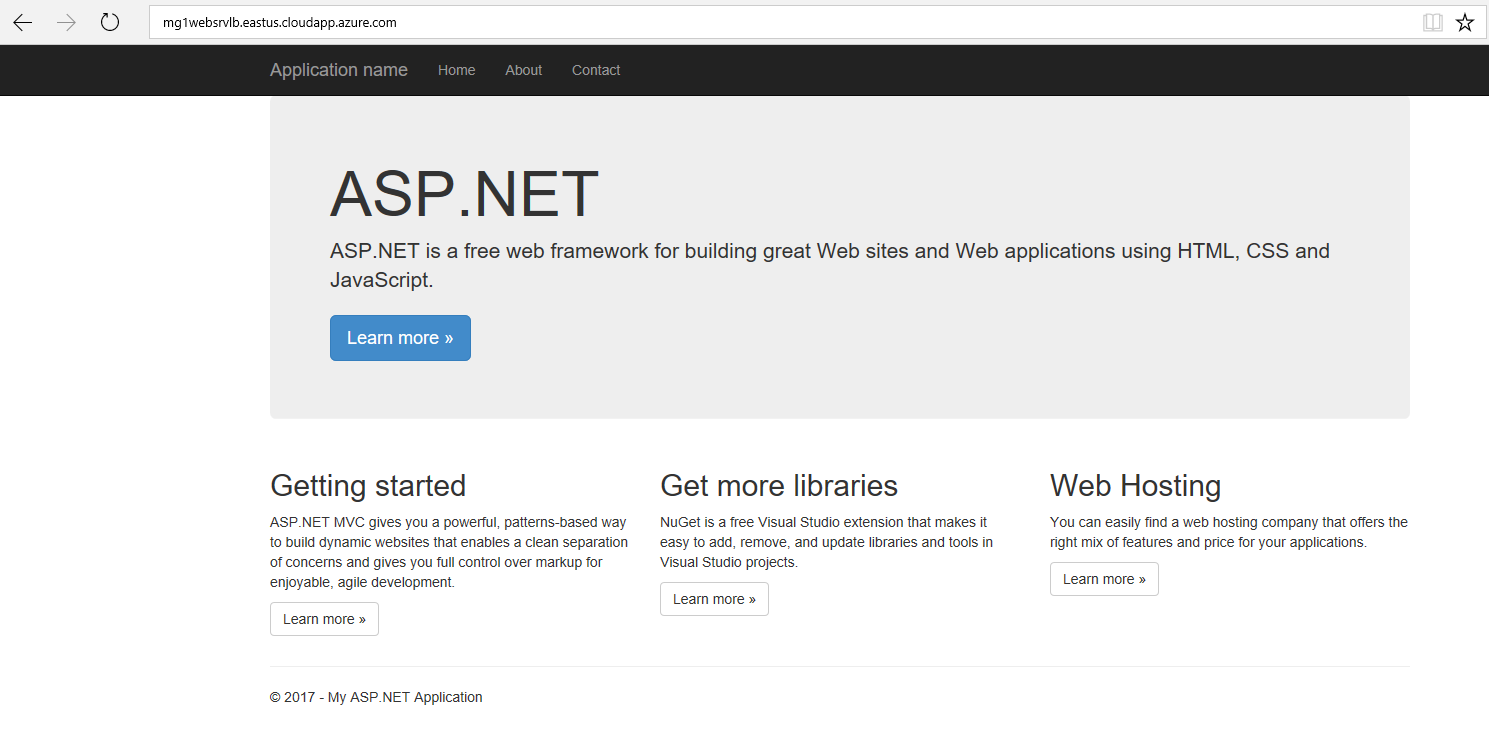
**Note:** Deployment configuration will determine the number of targets that can be deployed to, in parallel. The selection accounts for the percentage of targets that must remain available at any time during a deployment. It is also used to determine the success and failure conditions during deployment.

For example: **½ of the targets in parallel**: Deploys to half of targets at once. The status of the overall deployment will be displayed as ‘succeeded or partially succeeded’, if the deployment was successful to at least half of the targets else the overall status will be displayed as ‘Failed’

1. Enter a name for the release definition, and optionally change the name of the environment, then save the new release definition.
2. Create a new release, select the latest build, and deploy it to the single environment in the definition.
3. Navigate to the ‘Release’, you should see the deployment of IIS website succeed on the VM.



1. You can navigate to the newly deployed website using the publicip/dns name, for example: <http://mg1websrvlb.eastus.cloudapp.azure.com>**.** 
   * Note: if you used the template from Appendix to provision the VM, the url will be in the format of: http://<envPrefixName>websrvlb.eastus.cloudapp.azure.com



## **Appendix**

We have provided a sample script below to provision a new VM on Azure with necessary pre-requisites. The VM provisioning takes about ~30 minutes or more.

**Note:**

1. Set the execution policy to run the scripts, for example,

PS> **Set-ExecutionPolicy** **Unrestricted**

1. Script relies on Azure RM modules, you can get from [here](https://github.com/Azure/azure-powershell/releases/download/v3.4.0-January2017/azure-powershell.3.4.0.msi)
2. Extract the zip to your local folder.



1. **Mandatory step:** Edit the ‘iiswebtemplateparameters.json’ file with appropriate parameter values. Please change, envPrefixName (to less than 5 characters), username and password values
2. Launch Azure Powershell command
3. Navigate to the extracted folder
4. Run .\iisvmformg.ps1
   1. When prompted, enter your subscription name
   2. For rgname: <**Enter new Resource Group Name**>; If you provide an existing resource group, you might lose other resources in the resource group / script might fail to cleanup

