**Ansible Documentation**

**Installing and Configuring IIS with Ansible**

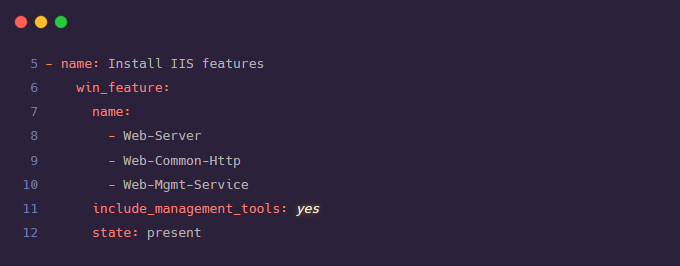
In this step we will use an Ansible playbook to configure our windows virtual machines. After configuration, we will be able to publish .NET CORE web applications to azure VMs directly using Visual Studio. The playbook prepares the VMs to receive the web application using WebDeploy.

There are two important parts of the ansible architecture required for configuration management.

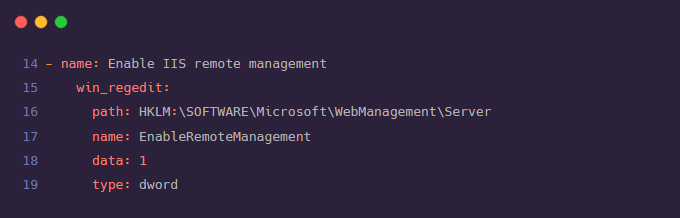
1. Ansible playbook: This is the main script with the set of instructions that need to be implemented on multiple hosts.
2. Inventory: This is the part that maintains the structure of the network environment. We create different nodes and pass these nodes to the playbook, so that it knows which action needs to be implemented on which resource.

**The ansible playbook (iis.yml) for our deployment will do the following steps:**

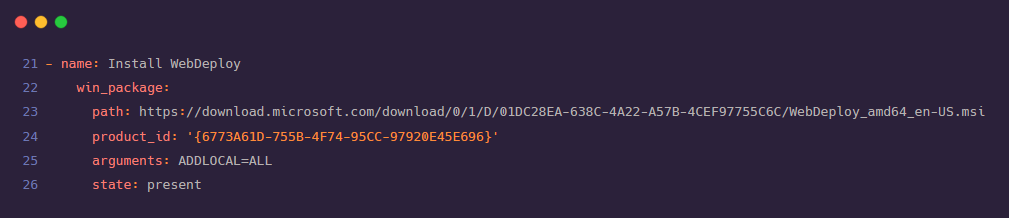
1. Install the basic set of IIS windows features and Install the Web Management Windows feature.
   1. win\_feature: installs or uninstalls features or roles on windows server.
   2. include\_management\_tools: adds corresponding management tools to specified features



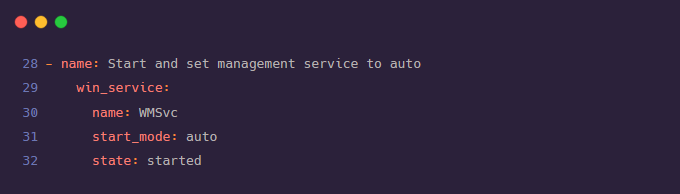
1. Set the registry value to enable remote management of the Web Management service.
   1. win\_regedit: add, modify or remove registry keys and values.



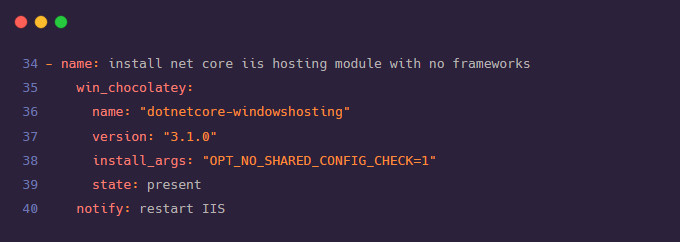
1. Downloads and installs the Web Deploy software package.
   1. win\_package: installs or uninstalls software packages for Windows.



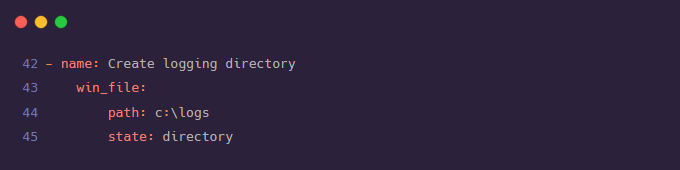
1. Starts and sets the Web management service to automatic.
   1. win\_service: manage and query Windows services.



1. install .net core IIS hosting module with no frameworks.
   1. win\_chocolatey: manage packages using chocolatey. If chocolatey is missing the module will install it.

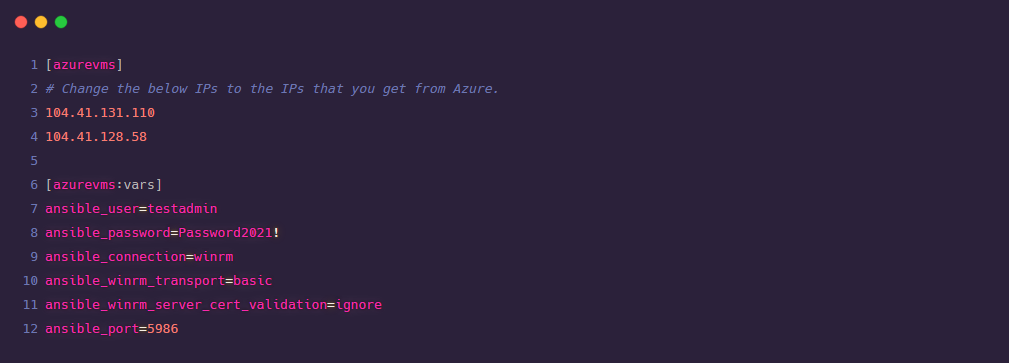


1. Create logging directory.
   1. win\_file: create or remove files or directories.



**The inventory (hosts) for our ansible includes the following:**

1. [azurevms]: This is a node that contains the list of windows virtual machines on azure that we want to configure.
2. [azurevms:vars]: These are the variables that are needed to make the WinRM connection to work.
   1. ansible\_user=testadmin (username)
   2. ansible\_password=Password1234! (password)
   3. ansible\_connection=winrm (this connection is set to SSH by default, so to make the connection to Windows VM to work, this needs to be set as winrm)
   4. ansible\_winrm\_transport=basic
   5. ansible\_winrm\_server\_cert\_validation=ignore (set to ignore as the certificates are generated locally to virtual machines are self-signed)
   6. ansible\_port=5986 (ansible port)



**Steps to invoke ansible playbook:**

1. Open the Azure cloud shell.
2. Make a new directory for Ansible files and change to this directory using following commands:

*mkdir ansibleConfigs*

*cd ansibleConfigs*

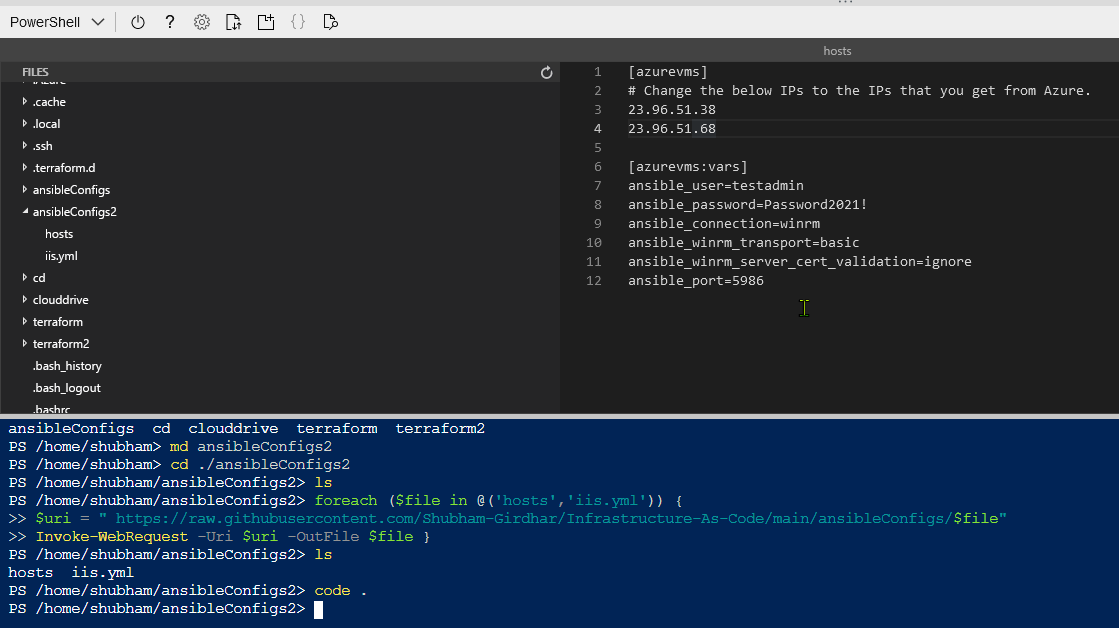
1. Get the hosts (inventory) and iis.yml (playbook) files from the git repo using the following command:

*foreach ($file in @('hosts','iis.yml')) {*

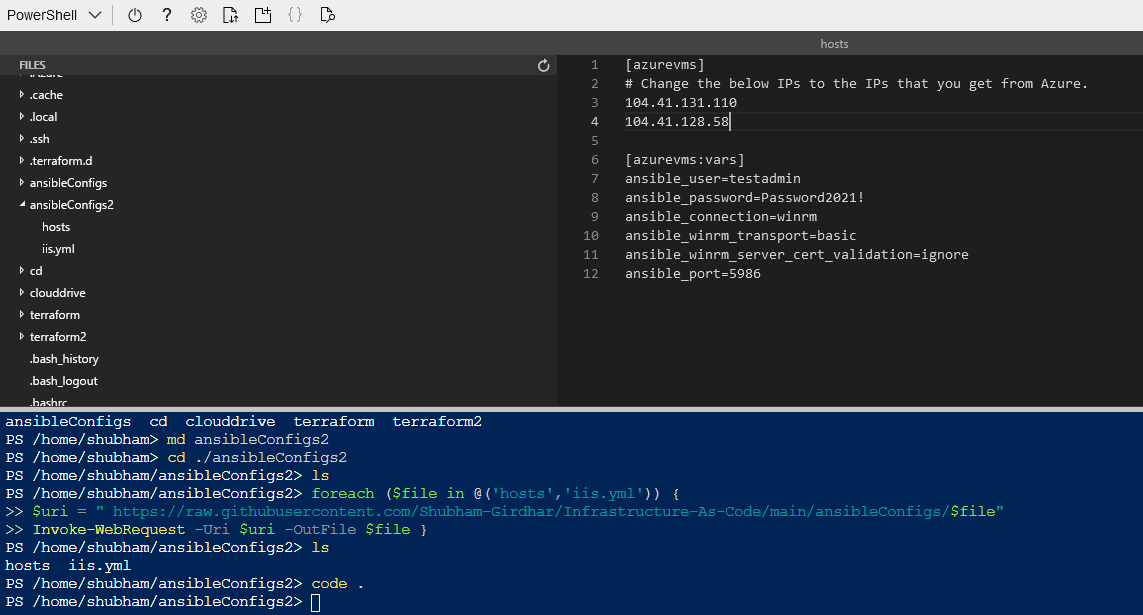
*$uri = " https://raw.githubusercontent.com/Shubham-Girdhar/Infrastructure-As-Code/main/ansibleConfigs/$file"*

*Invoke-WebRequest -Uri $uri -OutFile $file }*

1. Now, we update the hosts inventory and add the ip address of the VMs created using terraform. This is done so that the inventory points the playbook to the correct VMs.
2. To Edit the inventory, type *code .* This will open VS Code editor. Now update the IP addresses under the [azurevms] node and add the IPs of the two VMs we got as output of the *terraform apply* step.



1. Use Ctrl + s to save and Ctrl + q to exit the editor.



1. Finally, it’s time to execute the ansible playbook. This is done using the ansible-playbook command which takes the following two parameters:
   1. name of the playbook (iis.yml)
   2. - i name of host (- i hosts)
2. So, the final command looks like:

*ansible playbook iis.yml - i hosts*

1. This should configure the VMs by installing IIS features, enabling IIS remote management, installing WebDeploy, starting and setting management service to auto, installing .net core iis hosting module and creating a logging directory.

