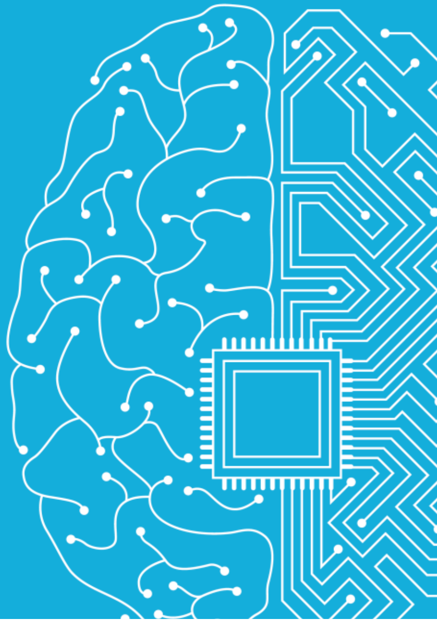
Azure Custom Script Extensions



architech

Cloud Computing Lunch & Learn Series

1. Cloud Computing for Non-Techies

2. Overview of Azure IaaS and deploying an HA, secure Linux cluster

3. **Overview of Custom Script Extensions and deploying Ansible to a Linux cluster**

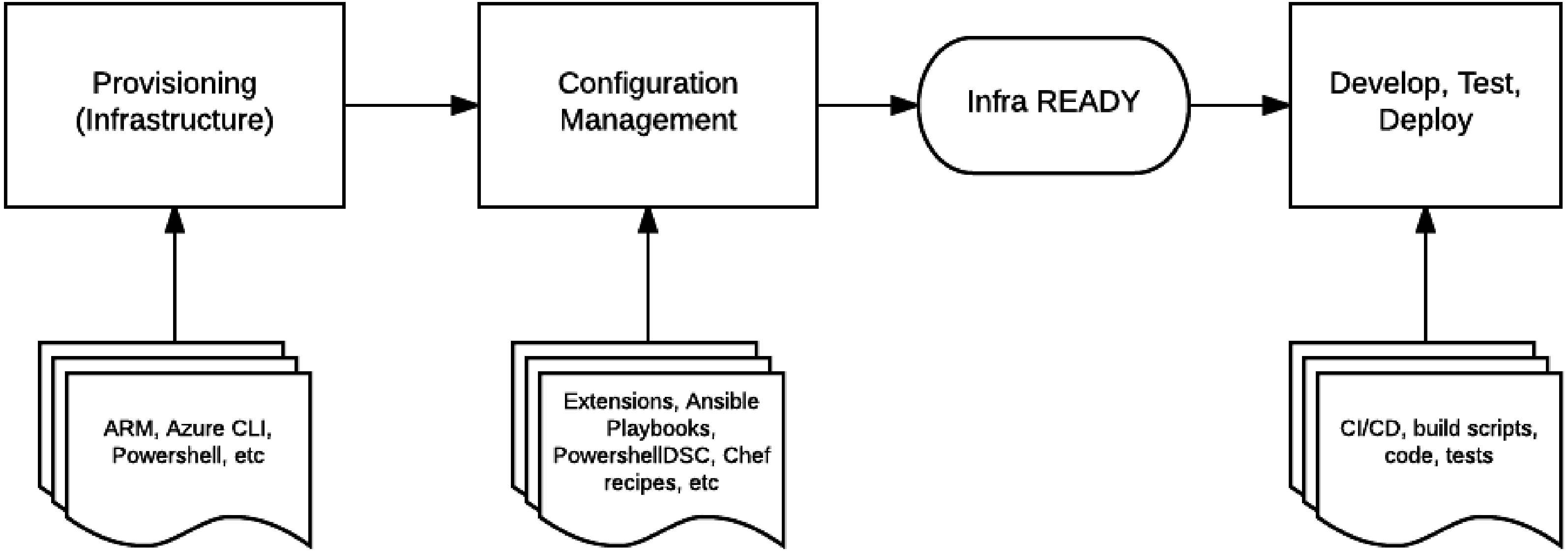
4. Deploying nginx, node.js, mongoDB using ansible to a Linux cluster

5. Deploying a containerized node.js and mongoDB application to a Linux cluster

Agenda

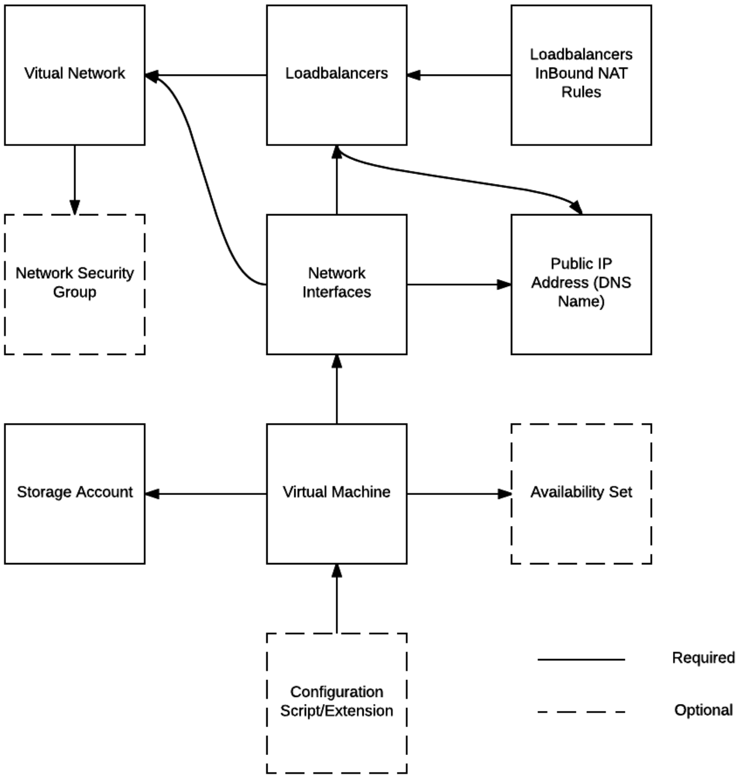
1. Infrastructure Provisioning & Configuration Management process
2. Azure Linux Custom Script Extensions
3. Defining script extensions in ARM
4. waagent service
5. Troubleshooting deployment
6. Demo – deploying extensions to install Ansible on the control and host machines

Infrastructure Provisioning & Configuration Management

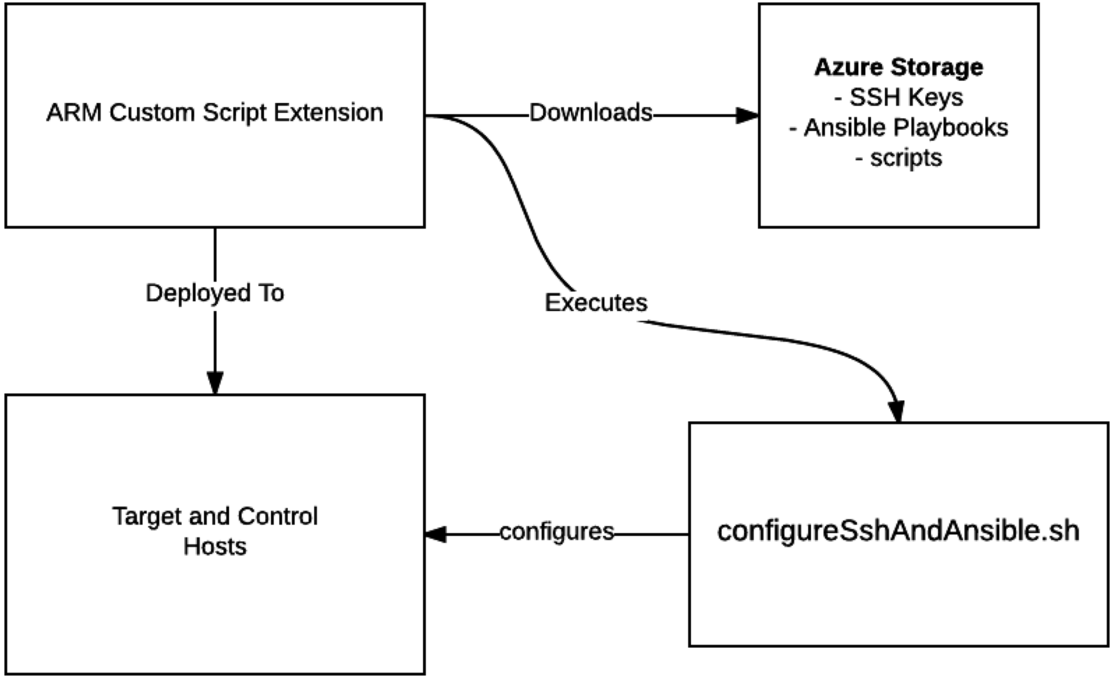

Provisioning (Infrastructure)
Configuration Management
Infra READY
Develop, Test, Deploy
ARM, Azure CLI, Powershell, etc
Extensions, Ansible Playbooks, PowershellDSC, Chef recipes, etc
CI/CD, build scripts, code, tests

Azure Custom Script Extensions

- The Linux cluster is provisioned. How do we install Ansible on the cluster?
- That is where custom script extensions come in…
- During deployment of the ARM template, you can execute custom scripts to perform initial configuration management
- Custom Script Extensions is just another ARM resource
- (There are actually multiple extension types")



Azure Custom Script Extensions



ARM Custom Script Extension

Azure Storage
- SSH Keys
- Ansible Playbooks
- scripts

Target and Control Hosts

configureSshAndAnsible.sh

Defining extensions in ARM

- Extension resources can be defined in following ways with ARM templates:
  1. Extension only ARM template
  2. Defined as a top-level resource in a template along with other resources
  3. Defined as a child-level resource (e.g. of the VM resource)
- Depending on the approach, you use different Azure CLI commands
- See https://goo.gl/vbSQuK for all the deployment scenarios and command used for each.

Extension Resource Definition

```
{
    "type": "extensions",
    "name": "[concat(variables('appVmExtensionName'), copyIndex(1))]",
    "apiVersion": "2015-06-15",
    "location": "[resourceGroup().location]",
    "dependsOn": [
        "[resourceId('Microsoft.Compute/virtualMachines', \
concat(variables('appVmName'), copyIndex(1)))]"
    ],
    "properties": {...}

}
```

Extension Resource Definition

```
"properties": {
    "publisher": "Microsoft.Azure.Extensions",
    "type": "CustomScript",
    "typeHandlerVersion": "2.0",
    "settings": {
        "fileUris": [ ... ],
        "commandToExecute": "[variables('targetHostCommand')]“
    },
    "protectedSettings": {
        "storageAccountKey": "[parameters('storageAccessKey')]",
        "storageAccountName": "[parameters('storageAccountName')]"
    }
}
```

waagent

- So what executes the script that is uploaded to the host machines?
- It is a service called ‘waagent’ – Azure Linux Agent.
- This is a very important service. It is what communicates with the Azure Fabric Controller and enables image provisioning, networking, diagnostics, vm extensions and more.
- This agent downloads the required files to the following directory then executes the command you specify:

```
/var/lib/waagent/custom-script/download/0
```

- To access this directory you need for first ‘sudo su’

Troubleshooting

- First add appropriate logging in the shell script so you have something to help you troubleshoot!
- Unfortunately, the logs are not available through the portal interface.
- Find out which extension failed to deploy by going to the portal and clicking on the VM then extensions. You will see if provisioning succeeded or failed. Then go to the logs.
- Logs are found in:

```
/var/log/azure/custom-script/handler.log

/var/lib/waagent/custom-script/download/0/std*.log
```

Deploying an ARM template with Extensions

https://bitbucket.org/architech/azure-linux-ansible

How can this solution be improved?

- Instead of storing the scripts in azure storage, it would be better to download the script from git. If git is not already installed on the base image, we have a chicken and an egg situation.
- The ssh private key is stored in Azure storage then deployed to the jumpbox, and the public keys are deployed to the target hosts and the jumpbox. You could just deploy the public key then use ssh-agent forwarding on your local machine. But what if, multiple people need to access the jumpbox and the target hosts?
- There are options and we will discuss in future sessions.

References

- Azure Linux Extensions - https://blogs.msdn.microsoft.com/linuxonazure/2017/02/12/extensions-custom-script-for-linux/
- Azure Linux VM Agent Overview - https://docs.microsoft.com/en-us/azure/virtual-machines/virtual-machines-linux-agent-user-guide
- Linux VM extension configuration samples - https://acom-feature-videos-twitter-card.azurewebsites.net/en-us/documentation/articles/virtual-machines-linux-extensions-configuration-samples/