Abstract

If you are an enterprise who builds an application that processes credit card data, you need to conform to PCI DSS (Payment Card Industry Data Security Standard). Adherence to the standard means that you need to meet control objectives for your network, protect cardholder data, implement strong access controls, manage operations and more. To help customers to quickly standup infrastructure that conform to PCI DSS, we are releasing an Azure QuickStart sample. The template describes a stack that deploys a multi-tiered azure PaaS web application stack. It makes use of many nested templates, and can be customized as desired.

AZURE PAAS - MANAGEMENT & PCI COMPLIANT SOLUTION DEPLOYMENT GUIDE

Contents

[1 High level summary 2](#_Toc474353460)

[2 Pre-Requisites & Permissions required to deploy 2](#_Toc474353461)

[Local machine setup 2](#_Toc474353462)

[Application stack should be configured by 2](#_Toc474353463)

[3 Pre-deployment Steps 2](#_Toc474353464)

[3.1 Manual creation of Azure Automation Account 3](#_Toc474353465)

[3.2 Run Pre Deployment PowerShell Script 3](#_Toc474353466)

[3.3 Configure AD App: 4](#_Toc474353467)

[4 Deployment steps 8](#_Toc474353468)

[5 Post Deployment Steps 8](#_Toc474353469)

[5.1 Update 1&1 DNS setting with Application Gateway IP 8](#_Toc474353470)

[5.2 Run Post Deployment PowerShell Script 11](#_Toc474353471)

[5.3 Schedule Runbooks 13](#_Toc474353472)

[5.4 Setting up the web application (by a global admin) 13](#_Toc474353473)

[5.5 Install OMS Dashboards Views. 14](#_Toc474353474)

[5.6 Check and verify OMS solutions are collecting data 17](#_Toc474353475)

# High level summary

# Pre-Requisites & Permissions required to deploy

|  |
| --- |
| ☛ **Local machine setup**  * You should run the pre and post deployment scripts with Admin or remote signed credentials (runas Administrator will work just fine) * Install Azure PowerShell (if you don’t know how to install, refer this link <https://docs.microsoft.com/en-us/powershell/azureps-cmdlets-docs/> ) * Install Azure AD Powershell, refer this link <https://technet.microsoft.com/en-us/library/dn975125.aspx> |
| ☛ **Application stack should be configured by**  * AD Global Admin (if you don’t know what that is refer this link <https://docs.microsoft.com/en-us/azure/active-directory/active-directory-assign-admin-roles#global-administrator> ) * Azure Subscription Role (either of the following roles)   + Service Administrator – refer <https://docs.microsoft.com/en-us/azure/billing-add-change-azure-subscription-administrator>   + Co-administrator - <https://docs.microsoft.com/en-us/azure/billing-add-change-azure-subscription-administrator>   + Subscription Owner - <https://docs.microsoft.com/en-us/azure/billing-add-change-azure-subscription-administrator> |

# Pre-deployment Steps

## Manual creation of Azure Automation Account

Create an Automation account with **RunAs Service principal**. Unfortunately, ARM templates don't allow for creating AD service principals yet, so this step is currently a manual one.

* Refer the blog https://azure.microsoft.com/en-us/documentation/articles/automationsecconfigure-azure-runas-account/ for the steps.
* Creation of ServicePrincipal has a propensity to fail randomly. A basic verification whether it was successfully created is **mandatory**

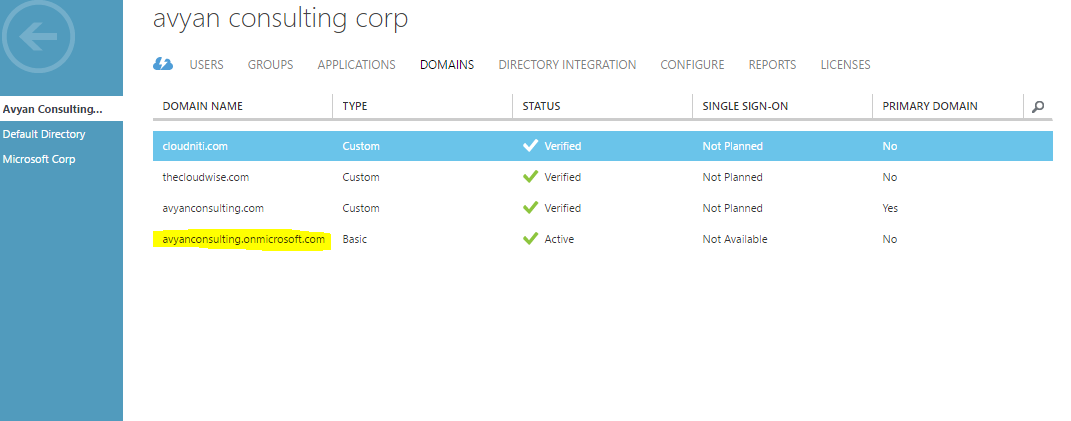
Note the name of the automation account. You will be using that as a parameter to the ARM template

## Run Pre Deployment PowerShell Script

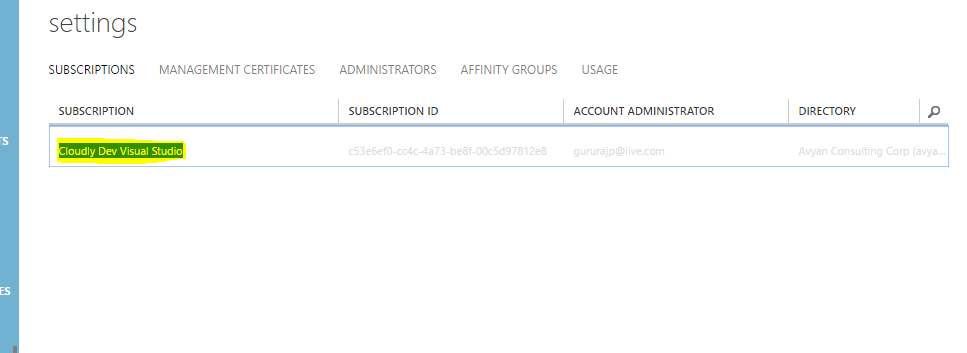
* + - Open PowerShell file(PreDeploy-ARMInputValues.ps1) and run as administrator in PowerShell
    - Change below mandatory values before run script

|  |  |  |
| --- | --- | --- |
| Parameter Name | How to get Value? | Comments |
| $azureADDomainName | * Login to <https://manage.windowsazure.com> * Open Domains under Active directory | Screenshots attached below |
| $subscriptionName | * Login to <https://manage.windowsazure.com> * Open Settings in left pane | Screenshots attached below |
| $suffix | * Provide unique website name | This is used to create a unique website name in your organization.  This could be your company name or business unit name  Some examples:   * “MSFT” * “ITCOE” * “ITAppDev” |

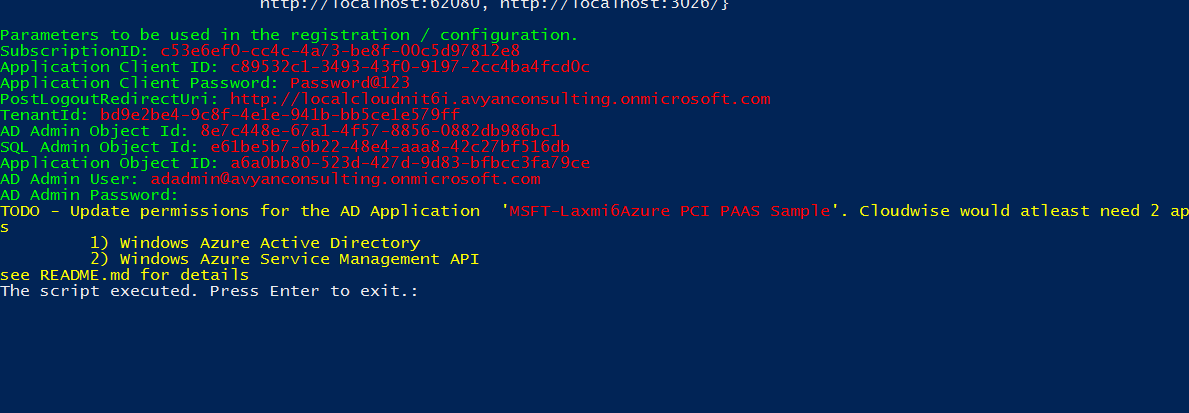
(Screenshot for to Get Domain Name)



(Screenshot for to Get Subscription Name)



* + - After Deployment Please copy below values, we have to pass these values in ARM Deployment



## Configure AD App:

* 1. In Azure Portal search for Azure Active directory. Open the “App Registrations” tab
  2. Open the AD Application that you just created. It should start with the name ($suffix + “Azure PCI PAAS Sample” )

$suffix is whatever you used during pre-deployment script.

* 1. Configure the following permissions in the “Required Permissions” tab
  2. List of Permissions

|  |  |
| --- | --- |
| **Category** | **Permission** |

|  |  |
| --- | --- |
| **Windows**  **Service**  **Management**  **API** |  |
| **Microsoft.Azure. ActiveDirectory** |  |
| **Azure Key Vault** |  |
| **Microsoft Graph API** | **Application Permissions** |

|  |  |
| --- | --- |
|  | Delegated Permissions |

|  |  |
| --- | --- |
|  |  |

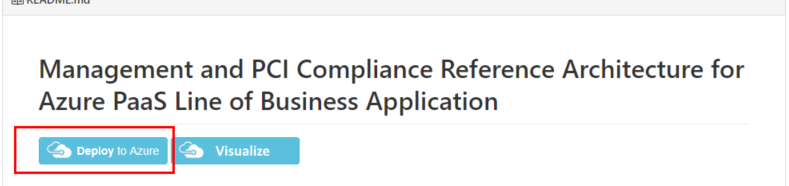


# Deployment steps

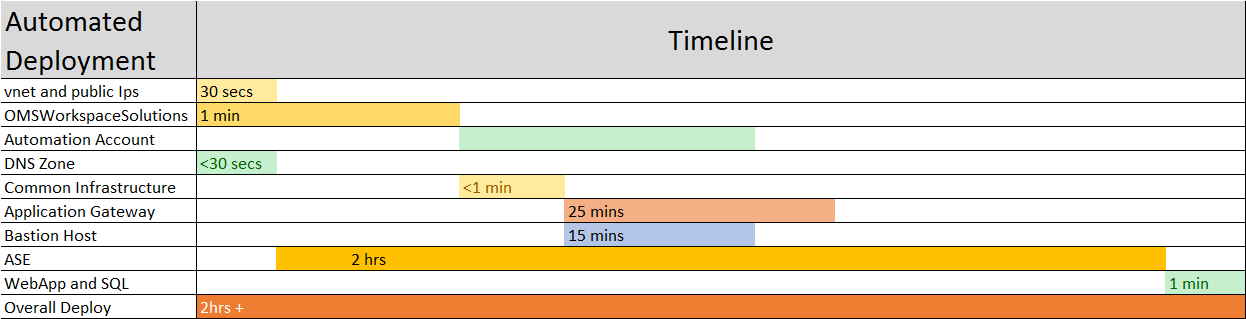
## Click on the Deploy to Azure on Github

Temporary location:

<https://github.com/AvyanConsultingCorp/pci-paas-webapp-ase-sqldb-appgateway-keyvault-oms>



## Deployment Timeline

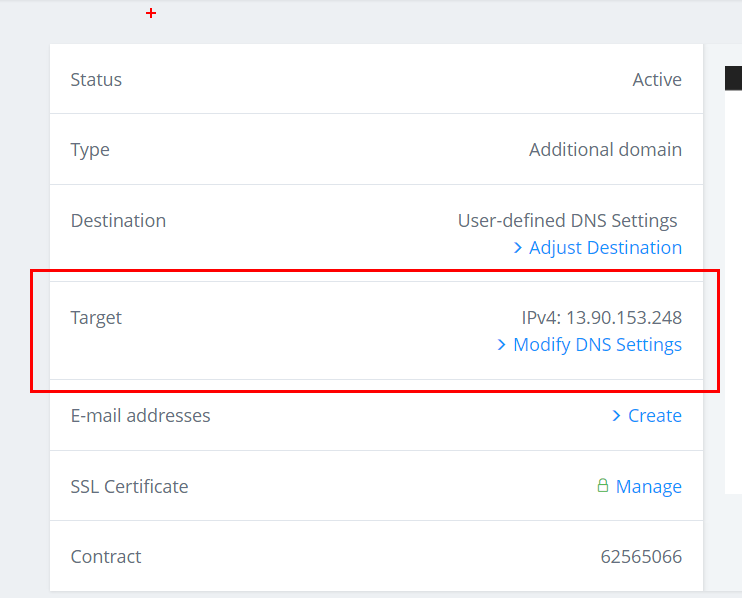


# Post Deployment Steps

## Update 1&1 DNS setting with Application Gateway IP

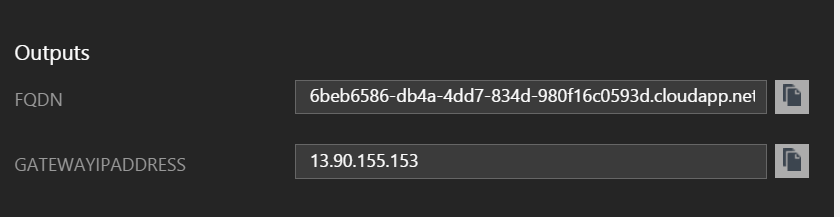
We are using 1&1 for managing DNS. It might be different for you.

Modify the DNS settings under the Target settings

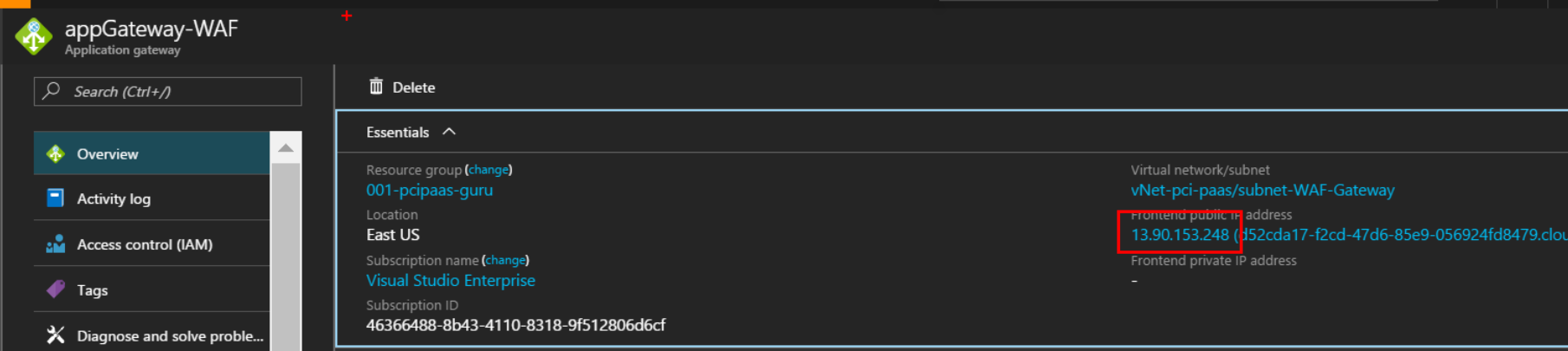


Note the public IP address of App gateway

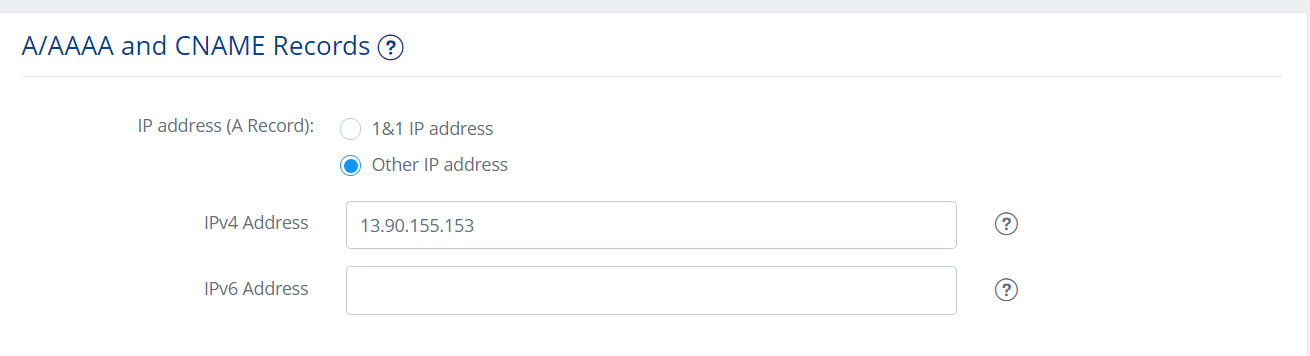
You can find the gateway IP address in the output of the deployment



Or go to the Application gateway object (appGateway-WAF) and checkout the IPaddress

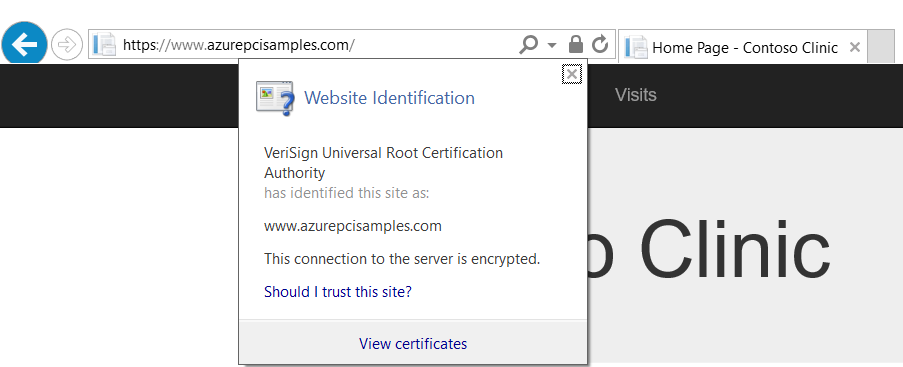


Update the A record IP address to be the App Gateway address



Verification

Site is working with <https://www.azurepcisamples.com>



## Run Post Deployment PowerShell Script

Post Deployment Powershell script used for following configuration

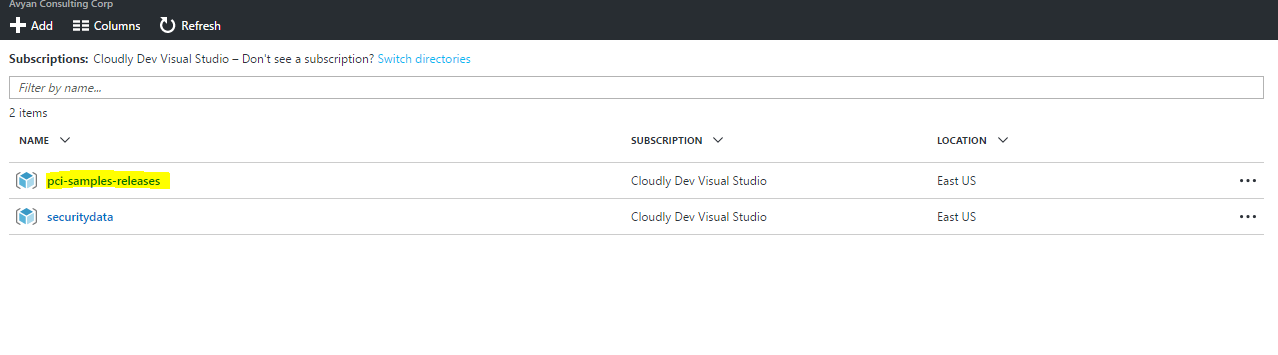
* Set Firewall rules for ASE Outbound and Client IP Address(To run scripts)
* Restore bacpac file into ContosoClinicDB DB
* Configure Dynamic Data Masking in Patients table
* Encrypt Columns using Key vault
* Set AD Authentication Admin
* Enable OMS Diagnostics

**Process to run Post Deployment Script**

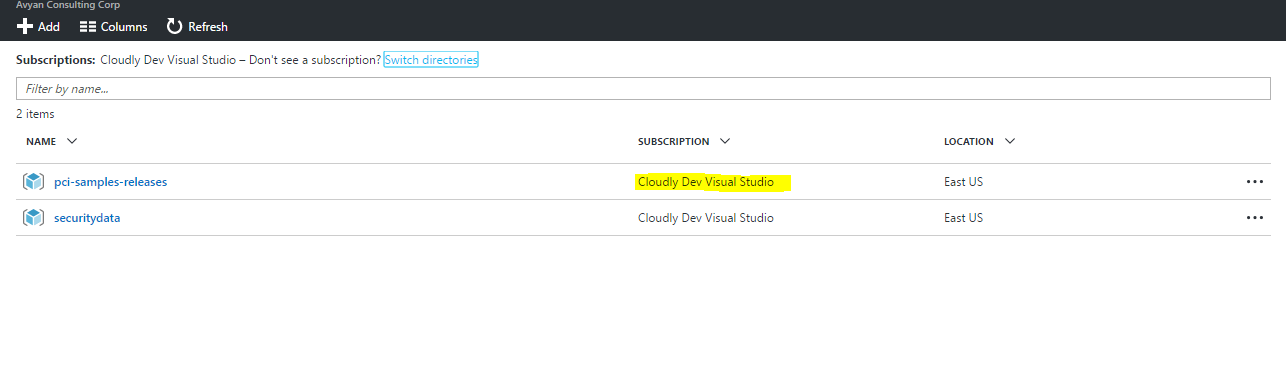
* + - Open Post Deployment PowerShell script (Path: ~/ pre-post-deployment/ PostDeployment.ps1) and run as administrator in PowerShell
    - Change below mandatory values before run script. It will ask Subscription credentials and SQL Credentials to execute complete script

|  |  |  |
| --- | --- | --- |
| Parameter Name | How to get Value? | Comments |
| $ResourceGroupName | * Login to <https://portal.azure.com> * Open Resource groups and copy resource group created using ARM deployment | Screenshots attached below. Provide Resource Group Name Created through ARM template |
| $ServerName | * Login to <https://portal.azure.com> * Copy Sql Server name under resource group created using ARM deployment | Provide Sql Server name (not required full name) Created through ARM template |
| $sqlPassword | * Provide SQL Password passed while creating ARM template | This value passed as input parameter in ARM Deployment |
| $ClientIPAddress | * Open windows cmd prompt and type ipconfig | Your system ip address |
| $ASEOutboundAddress | * Login to <https://portal.azure.com> * Open ASE under resource group created using ARM deployment * Open Properties and copy Outbound address |  |
| $ADAdministrator | * Provide SQL Password passed while creating ARM template | This value passed as input parameter in ARM Deployment |
| $subscriptionName | * Login to <https://portal.azure.com> * Open Resource groups and copy subscription name of resource group created using ARM deployment | Screenshots attached below |
| $KeyVaultName | * Login to <https://portal.azure.com> * Copy Key Vault name under resource group created using ARM deployment | Provide Key Vault Name Created through ARM template |

(Screenshot for to Get Resource Name)



(Screenshot for to Get Subscription Name)

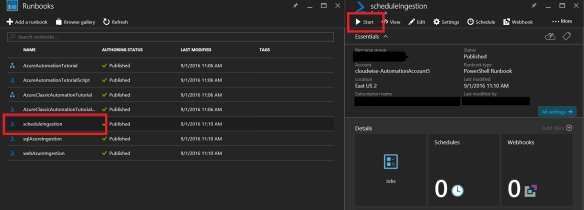


|  |
| --- |
| $ResourceGroupName = "01-pci-paas-automation" # Provide Resource Group Name Created through ARM template  $ServerName = "sqlserver-taer4k4qg5zfa" # Provide only Server name not full name  $userId = "testuser" # Provide user id of sql server  $sqlPassword = "PartnerSolutions123" # Provide password of sql server  $cmkName = "CMK1" # Provide Any Name  $cekName = "CEK1" # Provide Any Name  $keyName = "CMK1" # Provide Any Name  $ClientIPAddress = "168.62.48.129" # Provide Client IP address  $ASEOutboundAddress = "13.90.43.202" # Provide ASE Outbound address, we will get it ASE properties in Azure portal  $ADAdministrator = "globaladmin@sunilklive.onmicrosoft.com" # Pass AD Administrator, same we used for ARM Deployment  $subscriptionName = 'Visual Studio Enterprise' # Pass Subscription Name we used to create ARM Deployment  $ArtifactssubscriptionName = 'Cloudly Dev Visual Studio' # Pass Artifacts Subscription Name  $KeyVaultName= 'kv-pcisamples-taer4k4q' # Pass Key Vault Created through ARM template |

* + - It will ask Subscription credentials and SQL Credentials to execute complete script

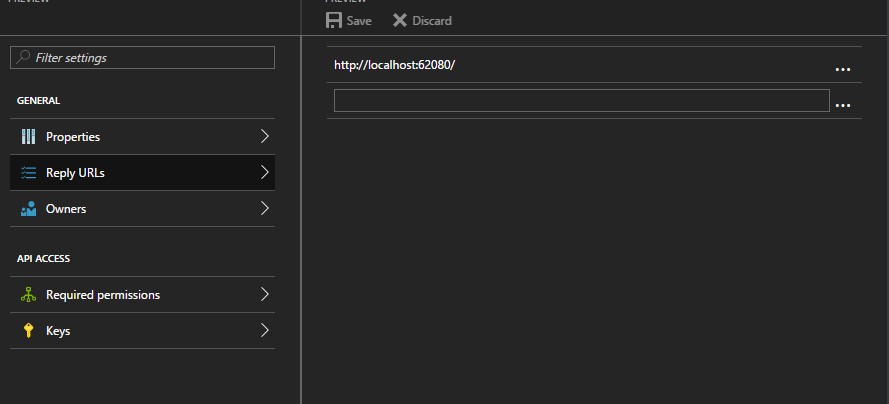
## Schedule Runbooks

* + - Click open the scheduleIngestion runbook and click start to run the runbook. This step will kickstart the data ingestion to the OMS workspace specified.



|  |
| --- |
| Setting up the web application (by a global admin) |

1. **Ensure**: Reply url of the webapplication is configured in the AD application. It should be something like http://webapp.ase.azurepcisamples.com . or <http://www.azurepcisamples.com>



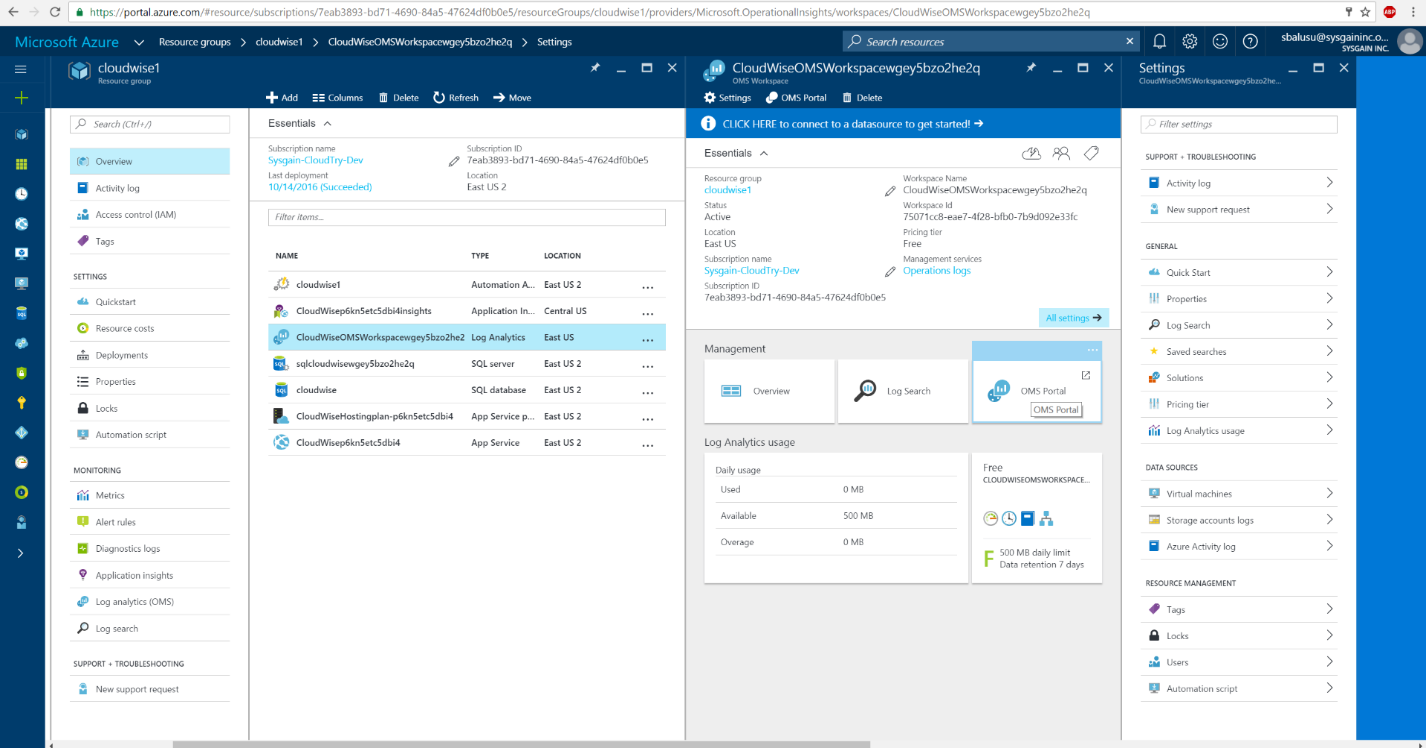
1. Open Portal and then you get an initial screen where you need to put subscription id,Client APplicaion ID and Client secret and then press submit.
2. After submit you will be redirect to AD login page where you need to input your active directory url and then press GO.
3. After this you need to input your LiveID credentials and after successfully login you will be redirect to Rule page where you can see all rules and submit rules.

## Install OMS Dashboards Views.

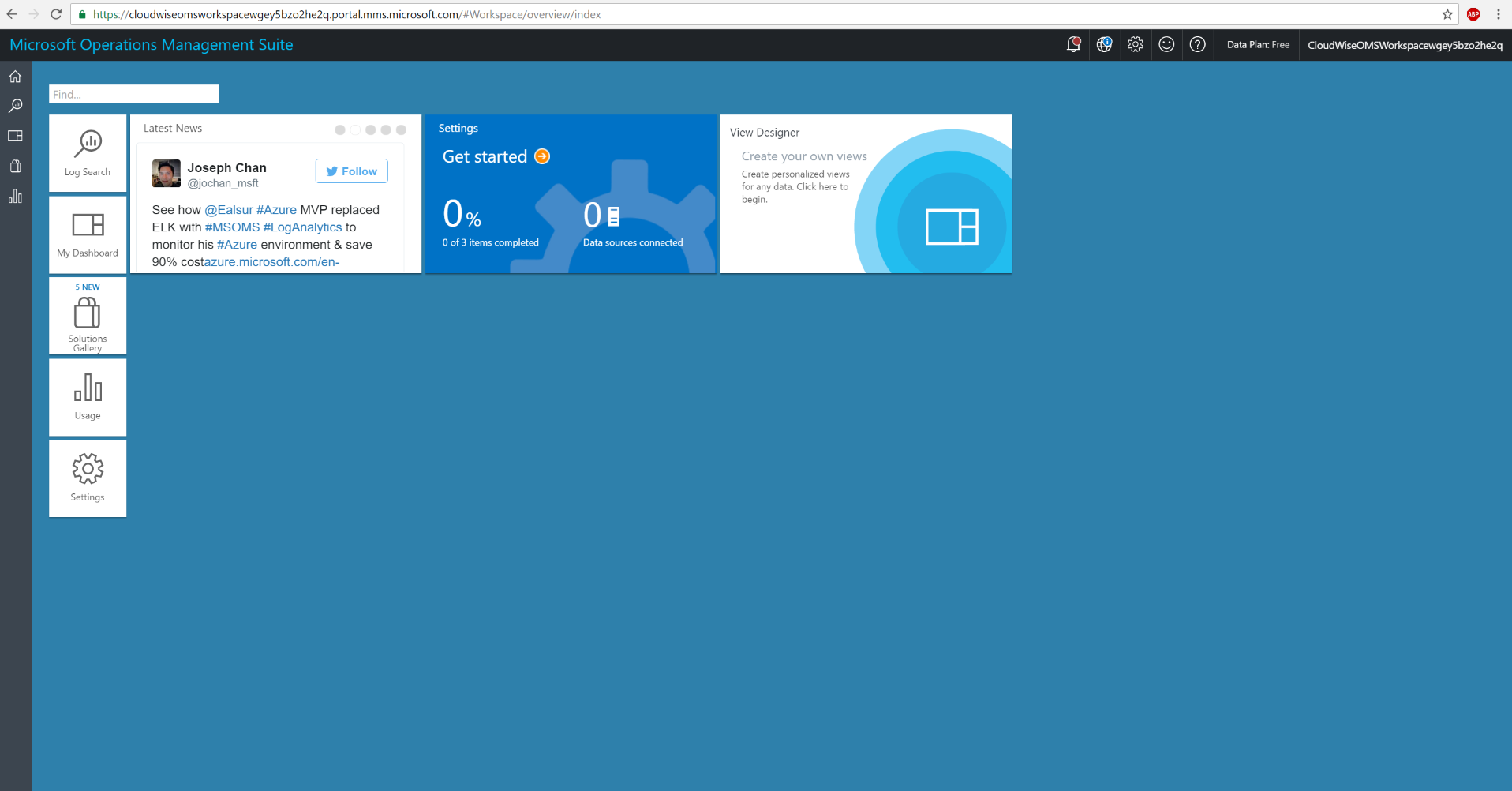
This is currently a manual process as ARM json deploys do not yet support creation of OMS views.

(By a ServiceAdmin/Contributor role)

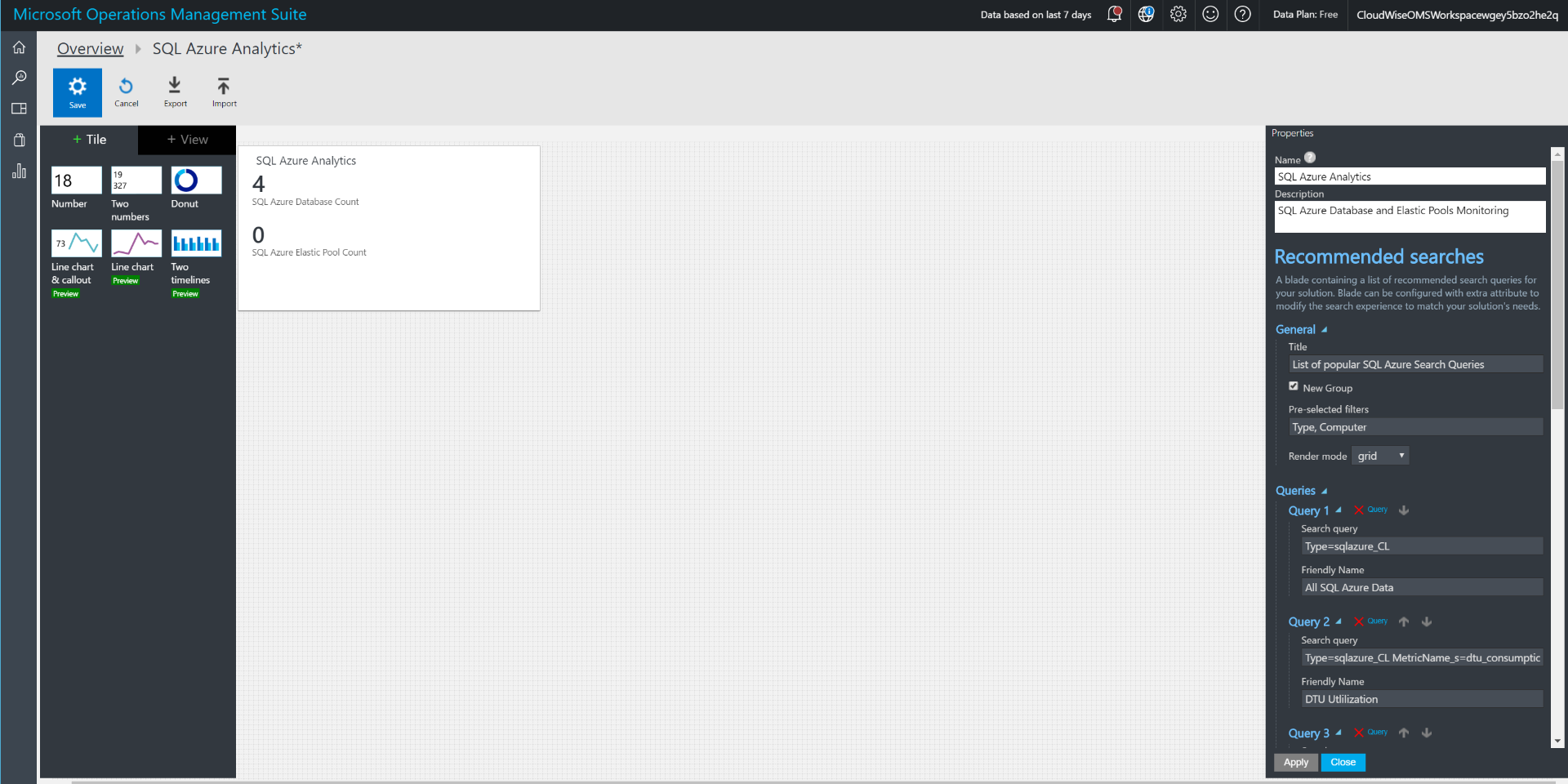
1. Open the resource group and click on the OMS Portal link. This will open the OMS portal in a different window

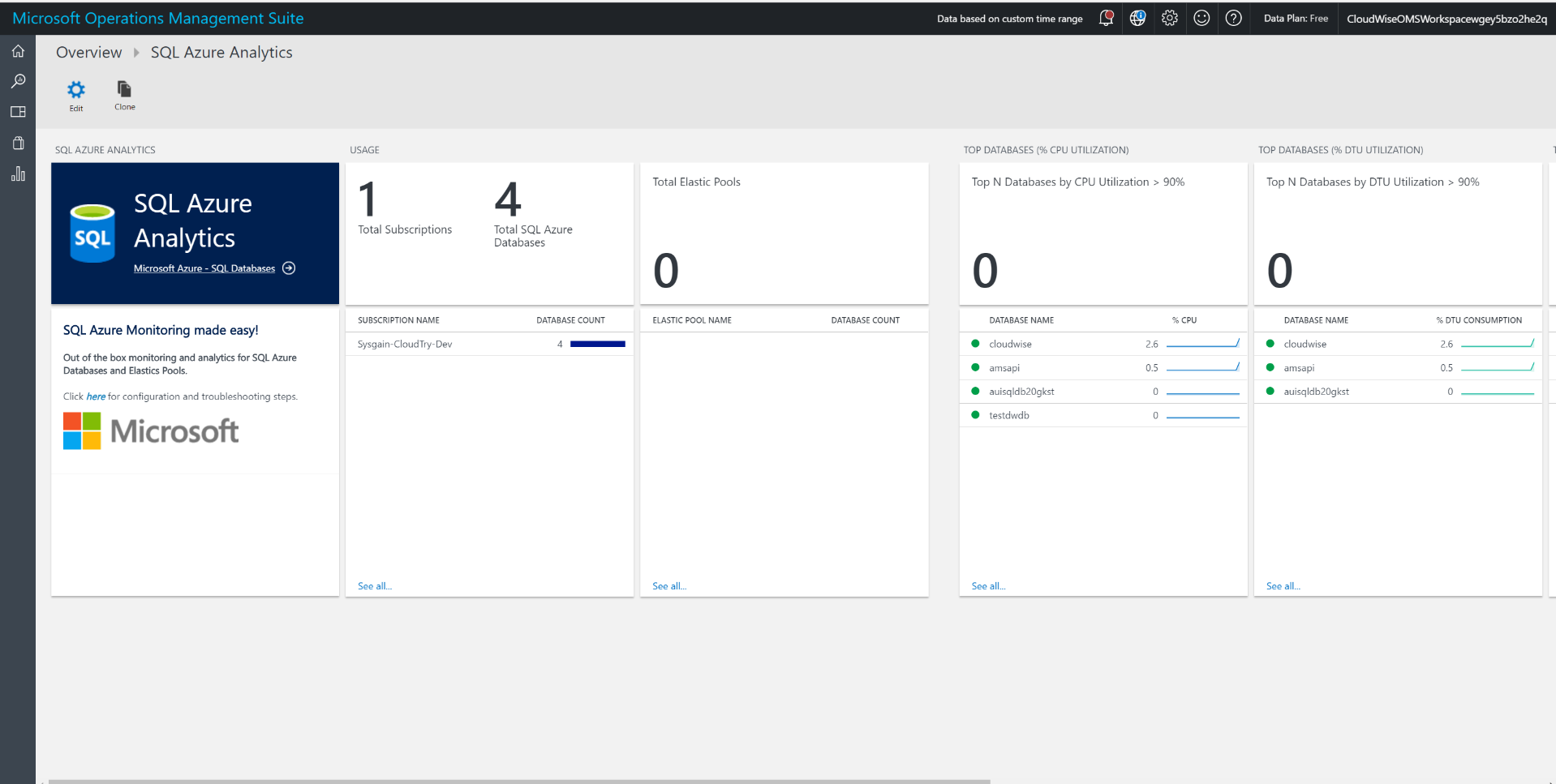


1. Click on the View Designer

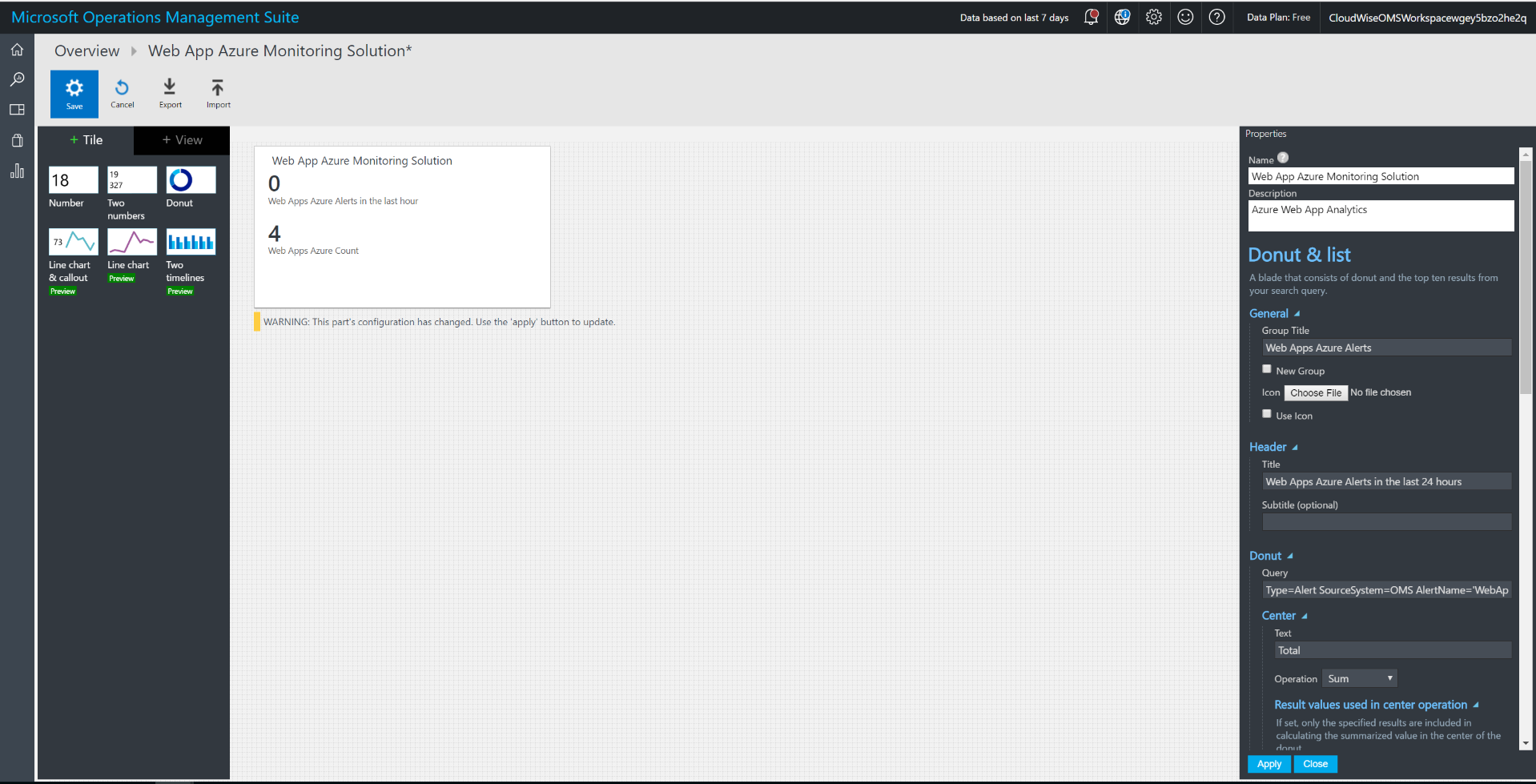


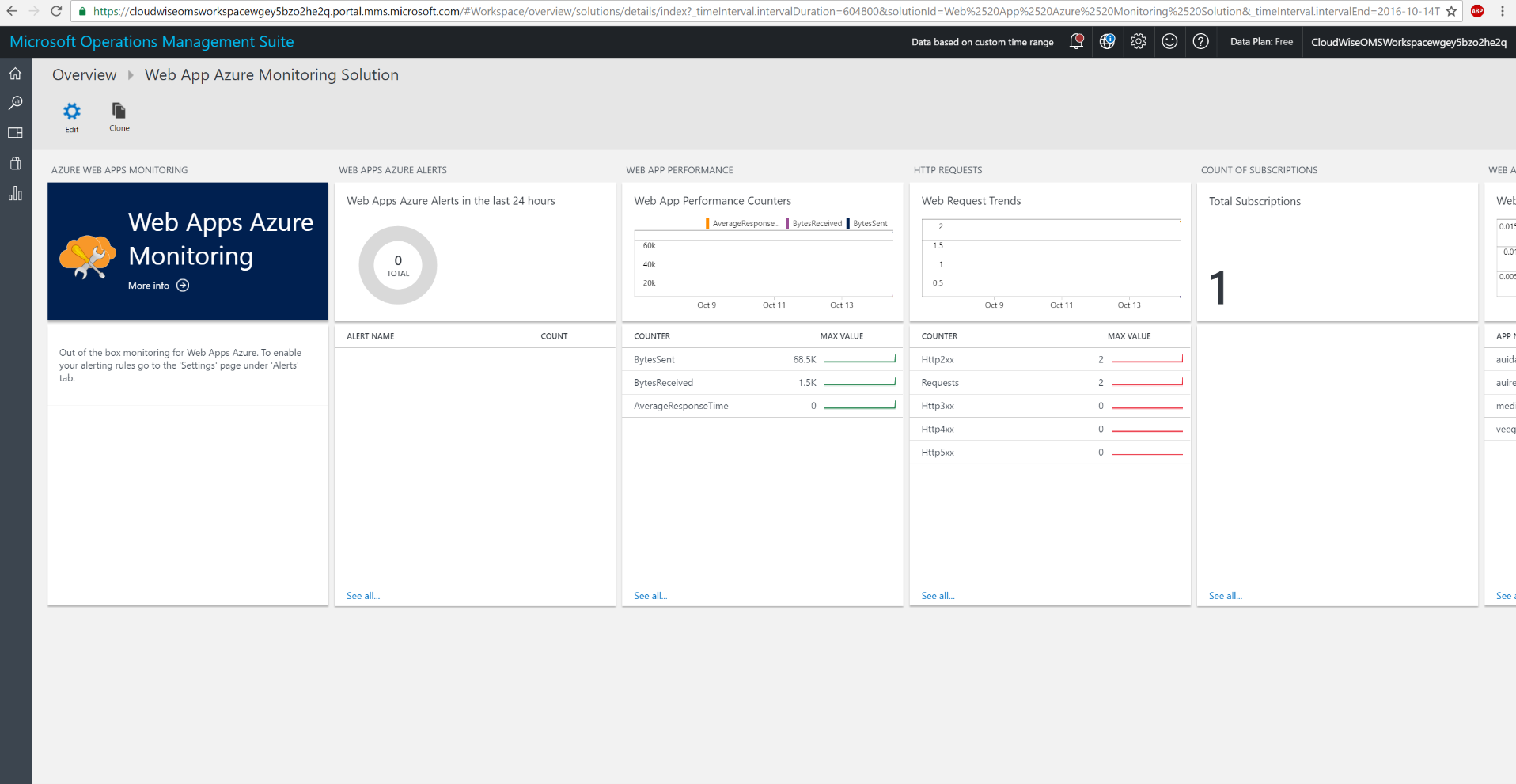
1. Import SQL DB view by clicking on the Import button and browsing to the file (OMSAzureDashboards\OMSSQLDBAzureMonitoringSolution.omsview)





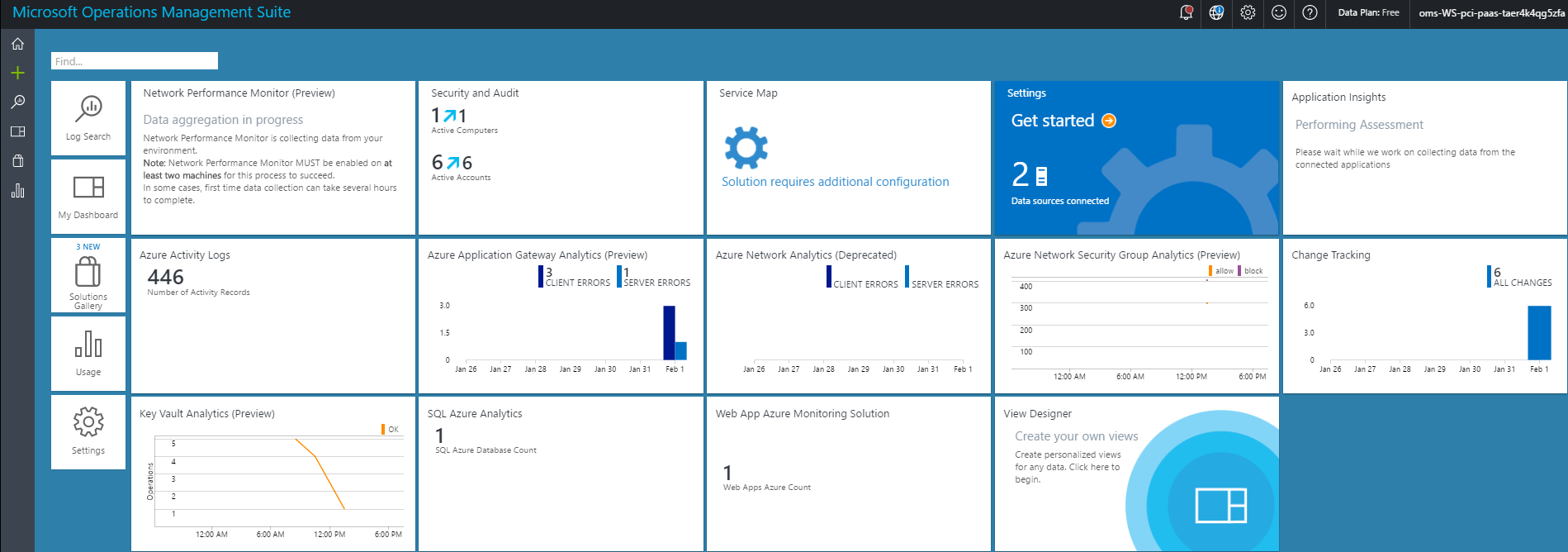
1. Repeat the same step for the Web App Monitoring datboard. Import view OMSAzureDashboards\ OMSWebAppAzureMonitoringSolution.omsview





## Check and verify OMS solutions are collecting data

When you login to the OMS workspace installed in the resourcegroup, you should navigate to the portal and you’ll see something like this. Data is being collected and Solution is being monitored and operationally ready for you to test out.



## Enable Azure Security Center

Follow instructions from here to enable data collections from Azure Security Center <https://docs.microsoft.com/en-us/azure/security-center/security-center-get-started>

To Do

* **Step 1**: Enable Data collection at the Subscription level.
* **Step 2**: Ensure all rules/policies are enabled (they are enabled by default)
* **Step 3**: Run recommendations
* **Step 4**: View the recommendation results. You will see that the solutions passes most rules
  + WAF enabled for public IP address
  + VM disk and data disk encrypted
  + All Azure Storages accounts are encrypted
  + SQL TDE, Auditing enabled
  + And many more…

☛ **Note:**

1. Currently OMS Monitoring agent is automatically installed along with the BastionHost deployment. We have (on purpose) not installed the security center VM agent, as the ASC team is moving towards using OMS agent for their purposes. Once they completely move to OMS agent, this solution will automatically work well with ASC.