# How to determine which Windows Azure Cloud Service Role instance handles the request

## Introduction

Windows Azure cloud service can host many instances in both production deployment and staging deployment.

Sometimes developers want to know which Role Instance handles requests from the client side for several conditions:

1. They want to know which deployment slot handles the request.
2. They use traffic manager to connect 7 cloud services together, and want to know which cloud service will handle the requests from south Asia.
3. They want to test load balance behavior.

So they may need to let the role instance connect to Azure management service and get the necessary messages.

This sample will demonstrate how to determine which windows Azure Cloud Service Role instance handles the request.

## Building the Sample

This sample requires Windows Azure management class libraries. Please run the following command in the [Package Manager Console](http://docs.nuget.org/docs/start-here/using-the-package-manager-console)

PM> Install-Package Microsoft.WindowsAzure.Management.Libraries -Pre

Before running the sample, you need to get some information.

Download the **publishsettings file** from:

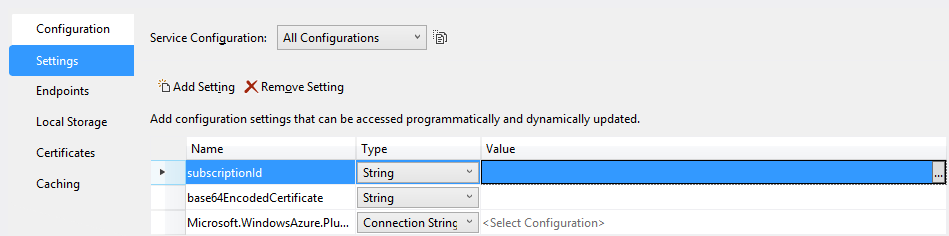
<https://manage.windowsazure.com/publishsettings/index?client=vs&schemaversion=2.0&whr=azure.com>

And get **SubscriptionID** and **ManagementCertificate Base64 string**.

Open the **Cloud project->Roles->DeterminRoleLocation**



Double click DetermineRoleLocation and then click the **Settings** tab.



Fill the **SubscriptionID** and **ManagementCertificate Base64 string** to **subscriptionId** and **base64EncodedCertificate**.

Then go to the **Default.aspx.cs/Default.aspx.vb** in the web project, change ‘cloudServiceNames’ in the following code to your cloud service names.

|  |
| --- |
| -Code block start-  --C# code snippet start--  //You can store your cloudServiceName in Azure Table storage, and get the value dynamically.  public static string[] cloudServiceNames = new string[3] { "testcloud", "testcloud1", "testcloud2" };  --C# code snippet end--  --VB code snippet start--  'You can store your cloudServiceName in Azure Table storage, and get the value dynamically.  Public Shared cloudServiceNames As String() = New String(2) {"testcloud", "testcloud1", "testcloud2"}  --VB code snippet end--  -Code block end- |

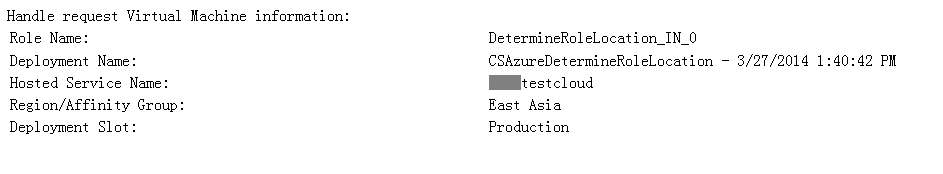
The code snippet above shows that in one subscription there are three cloud services which are connected by the traffic manager.

## Running the Sample

After the steps above, now you can deploy the cloud service to Windows Azure.

After completion of the deployment, input your website address:

You can get the message like below:



## Using the code

The following code snippet shows how to achieve the goal.

|  |
| --- |
| -Code block start-  --C# code snippet start--  protected void Page\_Load(object sender, EventArgs e)  {  var hostedServiceDetails = getCloudServiceDetailsByDeploymentId();  if (hostedServiceDetails!=null)  {  lbRoleName.Text = RoleEnvironment.CurrentRoleInstance.Id;  var deployment = hostedServiceDetails.Deployments.Where(dep => dep.PrivateId == RoleEnvironment.DeploymentId).FirstOrDefault();  lbDeploymentName.Text = deployment.Label;  lbHostServiceName.Text = hostedServiceDetails.ServiceName;  lbRegionOrGroup.Text =  hostedServiceDetails.Properties.AffinityGroup == null ? hostedServiceDetails.Properties.Location : hostedServiceDetails.Properties.AffinityGroup;  lbslot.Text = deployment.DeploymentSlot.ToString();  }  else  {  Response.Write("can't find this VM in current subscription");  }  }  static HostedServiceGetDetailedResponse getCloudServiceDetailsByDeploymentId()  {  var managementClient = new ComputeManagementClient(getCredentials());  var currentDeployment = new HostedServiceGetDetailedResponse.Deployment();  foreach (var serviceName in cloudServiceNames)  {  var hostedServiceDetails = managementClient.HostedServices.GetDetailed(serviceName);  foreach (var deployment in hostedServiceDetails.Deployments)  {  if (deployment.PrivateId == RoleEnvironment.DeploymentId)  {  return hostedServiceDetails;  }  }  }  return null;  }  static SubscriptionCloudCredentials getCredentials()  {  return new CertificateCloudCredentials(subscriptionId, new X509Certificate2(Convert.FromBase64String(base64EncodedCertificate)));  }  --C# code snippet end--  --VB code snippet start--  Protected Sub Page\_Load(sender As Object, e As EventArgs)  Dim hostedServiceDetails = getCloudServiceDetailsByDeploymentId()  If hostedServiceDetails IsNot Nothing Then  lbRoleName.Text = RoleEnvironment.CurrentRoleInstance.Id  Dim deployment = hostedServiceDetails.Deployments.Where(Function(dep) dep.PrivateId = RoleEnvironment.DeploymentId).FirstOrDefault()  lbDeploymentName.Text = deployment.Label  lbHostServiceName.Text = hostedServiceDetails.ServiceName  lbRegionOrGroup.Text = If(hostedServiceDetails.Properties.AffinityGroup Is Nothing, hostedServiceDetails.Properties.Location, hostedServiceDetails.Properties.AffinityGroup)  lbslot.Text = deployment.DeploymentSlot.ToString()  Else  Response.Write("can't find this VM in current subscription")  End If  End Sub  Private Shared Function getCloudServiceDetailsByDeploymentId() As HostedServiceGetDetailedResponse  Dim managementClient = New ComputeManagementClient(getCredentials())  Dim currentDeployment = New Deployment()  For Each serviceName In cloudServiceNames  Dim hostedServiceDetails = managementClient.HostedServices.GetDetailed(serviceName)  For Each deployment In hostedServiceDetails.Deployments  If deployment.PrivateId = RoleEnvironment.DeploymentId Then  Return hostedServiceDetails  End If  Next  Next  Return Nothing  End Function  Private Shared Function getCredentials() As SubscriptionCloudCredentials  Return New CertificateCloudCredentials(subscriptionId, New X509Certificate2(Convert.FromBase64String(base64EncodedCertificate)))  End Function  --VB code snippet end--  -Code block end- |

## More Information

Please refer to the project for more details.