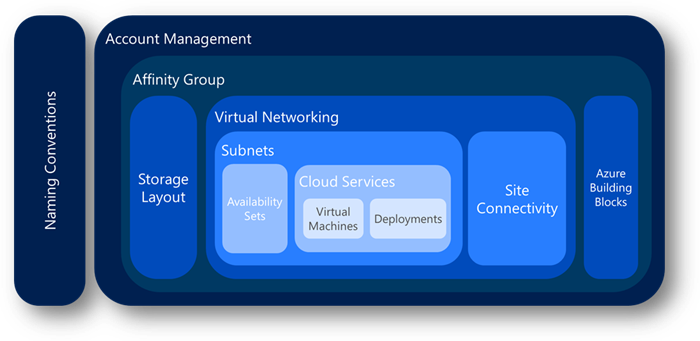
1 - Introduction

The Windows Azure Technical Documentation Library does an excellent job at describing each resource in the Azure infrastructure, but there exists little guidance on how to put the pieces together.

This document describes several areas where good planning goes a long way to ensure that the infrastructure built on top of Microsoft Azure can withstand the needs to make changes without affecting the manageability of the solutions.

2 - Implementation Guidance Overview

This guidance focuses on planning of a variety of resources that are involved in most of Azure implementations. The following diagram illustrates those resources:

  
*Figure 1 - Implementation Guidance Overview*

Although it is not a big secret that planning is key to a successful implementation of any solution, we have seen that many customers start using Azure without a complete view of the final product. This may be due to a variety of reasons, some of which are:

·         **Learning** – Customers start using Azure as they learn about it, and with very little effort, they achieve satisfactory use of the platform. Unfortunately, they rush to call their work “production-grade” which limits the flexibility to make changes to what has been implemented.

·         **Ease of Use** – Like many Microsoft products, Azure seems “easy enough” to use, and they, again, move to implement production workloads, without clear understanding of future growth.

·         **Proof of Concepts** – Azure represents an excellent platform to implement proof of concept exercises, since it requires very little investment to test a particular approach to implementation of solutions. Many times, customers incorrectly label these proof-of-concept exercises as baseline for a full implementation of the solution, and many decisions that they took lightly, with the intent of further research, are involuntarily pushed as permanent.

This guidance identifies many areas for which planning is key to the success of the Azure implementation.

In addition, it helps the implementation of solutions on the Azure platform by providing an order to the creation of the necessary resources. Although there is some flexibility, we suggest following a methodic order to minimize the need for rework.

v Naming Conventions

Ø Accounts

§ Subscriptions

Ø Affinity Groups

Ø Storage

Ø Virtual Networks

§ Subnets

§ Site Connectivity

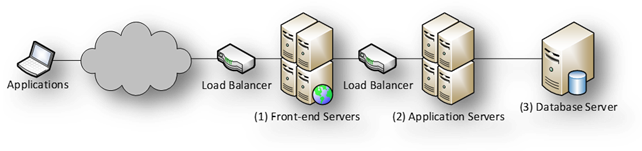
Ø Azure Building Blocks

Ø Cloud Services

§ Virtual Machines

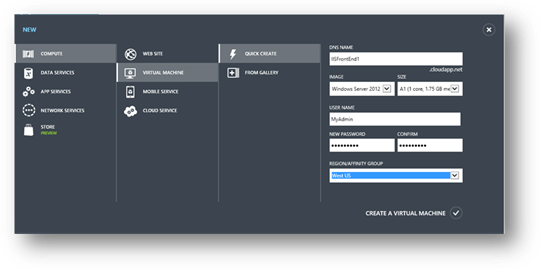
§ PaaS Deployments

For example, let us consider a fictitious application that the IT operations of Contoso is trying to implement on Azure. The application is a calculation engine for advanced financial operations. This application exposes a front end, implemented as a set of IIS web services, running on Microsoft Azure IaaS. These servers consume web services hosted on several application servers, which also expose a load balanced web service API. At the same time, these consume information from a database. The following diagram conceptually depicts the application’s infrastructure:

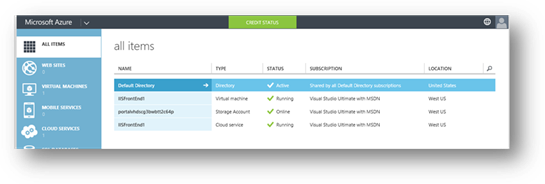
[](https://msdnshared.blob.core.windows.net/media/MSDNBlogsFS/prod.evol.blogs.msdn.com/CommunityServer.Blogs.Components.WeblogFiles/00/00/01/63/64/metablogapi/8877.clip_image0044_1794EE3E.png)  
*Figure 2 – Design of Contoso’s fictitious Financial Calculation Engine*

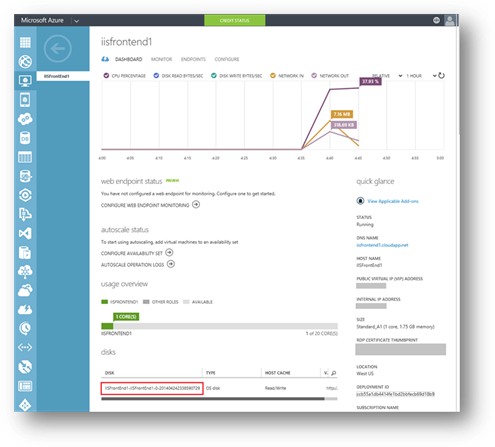
The implementation plan for Microsoft Azure is to create two cloud services under the same subscription: one cloud service to host the web servers, and another cloud service to host both the application servers and the database server.

If the default settings were to be used when creating one of the front servers, Microsoft Azure would use many default values. The following screen shows how one of the IIS Web servers where created for the front end:

[](https://msdnshared.blob.core.windows.net/media/MSDNBlogsFS/prod.evol.blogs.msdn.com/CommunityServer.Blogs.Components.WeblogFiles/00/00/01/63/64/metablogapi/5756.clip_image0064_0956754E.png)  
*Figure 3 - New Virtual Machine created using Quick Create*

When Microsoft Azure creates a virtual machine this way, the resulting resources are as follows:

[](https://msdnshared.blob.core.windows.net/media/MSDNBlogsFS/prod.evol.blogs.msdn.com/CommunityServer.Blogs.Components.WeblogFiles/00/00/01/63/64/metablogapi/7416.clip_image0084_3DF6D189.png)  
*Figure 4 - Resources associated to a Quick Created VM*

[](https://msdnshared.blob.core.windows.net/media/MSDNBlogsFS/prod.evol.blogs.msdn.com/CommunityServer.Blogs.Components.WeblogFiles/00/00/01/63/64/metablogapi/0871.clip_image0104_7DC0B80E.png)  
*Figure 5 - Detail of the virtual machine*

Several issues become evident in the above configuration:

·         The virtual machine name is the same as the cloud service name, which will make it confusing when managing the solution.

·         The storage account name is automatically generated using the string “portalvhds”, plus a set of random characters to make it unique.

·         The system disk name associated to the virtual machine is a combination of the cloud service name, the virtual machine name, the LUN number, and the date of its creation. When the number of disks increases, these names become difficult to identify, especially since the cloud service name coincides with one of the virtual machines name.

·         The vhd associated to the disk uses the same name as the disk with a “.vhd” extension. As with disks,

Establishing a good naming convention, as well as following a specific, systematic order to create the resources in Azure immensely reduces administrative burden and increases the chances of success for any implementation project.

3 - Guidance Areas

3.1 - Naming Conventions

A good naming convention should be in place before creating any artifact in an azure account. A naming convention ensures that all the resources will have a predictable name, which helps lower the administrative burden associated with management of those resources.

Each customer may choose to follow a specific set of naming conventions defined for the customer as a whole, for a particular Azure account, or for an Azure subscription.

It is easy for individuals to establish implicit rules when working with Azure resources. However, when a team needs to work on a project on Azure, that model does not scale well.

The important point is that the customers agree upon the set of naming conventions up front.

3.1.1 - General Considerations

There are some considerations regarding naming conventions that cut across the sets of rules that make up those conventions. The following sections describe these considerations.

*3.1.1.1 - Affixes*

When creating certain resources, Microsoft Azure will use some defaults to simplify management of the resources associated to these resources. For instance, when creating the first virtual machine for a new cloud service, Azure will suggest using the virtual machine’s name as the name for the cloud service.

Although this will not present problems, it may be beneficial to identify types of resources that need an affix to identify that type. In addition, clearly specify whether the affix will be at the beginning of the name (prefix) or at the end (suffix).

For instance, here are two possible names for a service hosting a calculation engine:

·         **Svc**CalculationEngine (prefix)

·         CalculationEngine**Svc** (suffix)

Affixes can refer to different aspects that describe the particular resources. The following table shows some examples typically used.

| **Aspect** | **Example** | **Notes** |
| --- | --- | --- |
| **Environment** | dev, stg, prod | Depending on the purpose and name of each environment. |
| **Location** | uw (US West), ue (US East) | Depending on the region of the datacenter or the region of the intended audience. |
| **Instance** | 01, 02 | For resources that may have more than one instance. For example, load balanced web servers in a cloud service. |
| **Product** | ce (for CalculationEngine) | Depending on the product for which the resource provides support. |
| **Role** | sql, ex, ora, sp, iis | Depending on the role of the associated VM. |

Make sure that the naming conventions clearly state which affixes to use for each type of resource, and in which position (prefix vs suffix).

*3.1.1.2 - Dates*

Many times, it is important to determine from the name of an resource, the date of creation. We recommend specifying dates in the**YYYYMMDD** format. This format ensures that not only the full date is recorded, but also that two resources whose names differ only on the date will be sorted alphabetically and chronologically at the same time.

3.1.2 - Naming Resources

Customers must define each type of resource in the naming convention, which should have rules that define how to assign names to each resource created. These rules should apply to all types of resources, for instance:

·         Accounts

·         Subscriptions

·         Affinity Groups

·         Storage Accounts

·         Virtual Networks

·         Subnets

·         Availability Sets

·         Cloud Services

·         Virtual Machines

·         Endpoints

·         Roles

·         Etc.

*3.1.2.1 – Descriptive Names*

Names should be as descriptive as possible, to ensure that the name can provide enough information to determine to which resource it refers.

3.1.3 - Computer Names

When administrators create a virtual machine from the gallery, Microsoft Azure will require them to provide a virtual machine name. Microsoft Azure will use the virtual machine name as the Azure virtual machine resource name. Azure will use the same name as the computer name for the operating system installed in the virtual machine. However, these names may not always be the same. In cases in which a virtual machine is created from a .vhd file that already contains an operating system, the virtual machine name in Microsoft Azure may differ from the virtual machine’s OS computer name. This situation may add a degree of difficulty to virtual machine management and we discourage it. Always ensure that the Azure virtual machine resource name is the same name as the computer name as assigned to the operating system of that virtual machine.

We recommend that the Azure Virtual Machine name be the same as the underlying OS computer name. Because of this, follow the NetBios naming rules as described in this article: http://support.microsoft.com/kb/188997.

3.1.4 – Storage Account Names

Storage accounts have special rules governing their names. In general, they are lower case names, and the assigned name, concatenated to the service (blob, table, or queue) and the default domain (core.windows.net) should render a valid, unique DNS name. For instance, if the storage account is called mystorageaccount, the following resulting URLs should be valid, unique DNS names[1]:

·         mystorageaccount.blob.core.windows.net

·         mystorageaccount.table.core.windows.net

·         mystorageaccount.queue.core.windows.net

In addition, storage accounts may take advantage of containers. These must adhere to the naming conventions as described in Naming and Referencing Containers, Blobs, and Metadata.

3.2 - Account Management

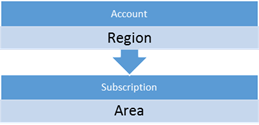
In order to work with Azure, customers need subscriptions. Resources, like cloud services or virtual machines, exist in the context of those subscriptions.

Enterprise customers typically have an Enterprise Enrollment, which is the top-most resource in the hierarchy, and is associated to one or more accounts.

For consumers and customers without an Enterprise Enrollment, the top-most resource is the Account.

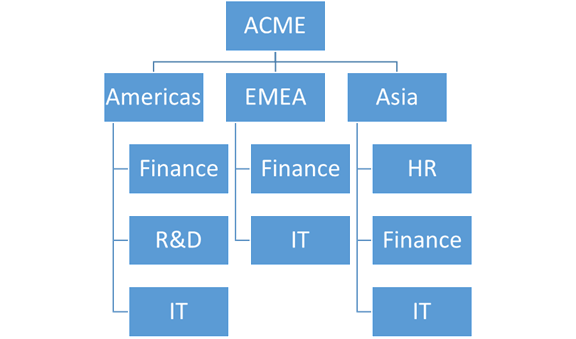
Subscriptions are associated to accounts, and there can be one or more subscriptions per account. Azure records billing information at the subscription level.

Due to the limit of two hierarchy levels on the Account/Subscription relationship, it is important to align the naming convention of accounts and subscriptions to the billing needs. For instance, if a global company uses Azure, they may choose to have one account per region, and have subscriptions managed at the region level.

[](https://msdnshared.blob.core.windows.net/media/MSDNBlogsFS/prod.evol.blogs.msdn.com/CommunityServer.Blogs.Components.WeblogFiles/00/00/01/63/64/metablogapi/8308.clip_image0124_3261144A.png)

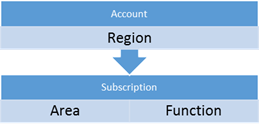
*Figure 6 - Simple Account/Subscription Hierarchy*

For instance, a company may use the following structure:

[](https://msdnshared.blob.core.windows.net/media/MSDNBlogsFS/prod.evol.blogs.msdn.com/CommunityServer.Blogs.Components.WeblogFiles/00/00/01/63/64/metablogapi/8321.clip_image0144_2B41D7D2.png)

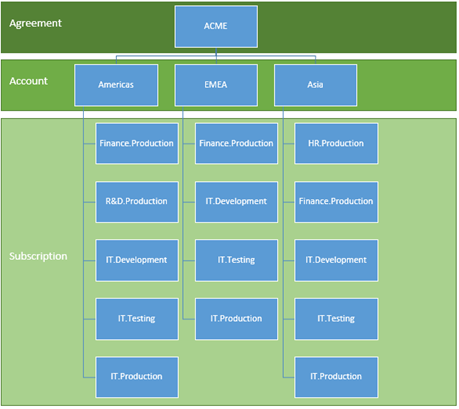
*Figure 7 - Example of a simple account/subscription hierarchy*

Following the same example, if a region decides to have more than one subscription associated to a particular group, then the naming convention should incorporate a way to encode the extra on either the account or the subscription name. This organization allows massaging billing data to generate the new levels of hierarchy during billing reports. This is shown below:

[](https://msdnshared.blob.core.windows.net/media/MSDNBlogsFS/prod.evol.blogs.msdn.com/CommunityServer.Blogs.Components.WeblogFiles/00/00/01/63/64/metablogapi/3124.clip_image0164_2DEA9383.png)

*Figure 8 - Complex account/subscription hierarchy, with subscription used to encode two levels*

This way, the organization could look as follows[2]:

[](https://msdnshared.blob.core.windows.net/media/MSDNBlogsFS/prod.evol.blogs.msdn.com/CommunityServer.Blogs.Components.WeblogFiles/00/00/01/63/64/metablogapi/5287.image_426FE301.png)

*Figure 9 - Example of a complex account/subscription hierarchy*

At the time of writing, Microsoft provides detailed billing via a downloadable file, typically for a single account, or for all accounts in an enterprise agreement. Customers can process this file, for example, using Excel. This process would ingest the data, partition the resources that encode more than one level of the hierarchy into separate columns, and use a pivot table or PowerPivot to provide dynamic reporting capabilities.

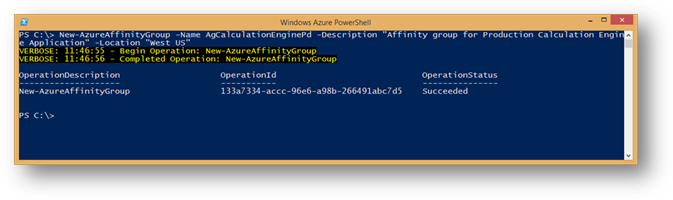
3.3 - Affinity Groups

Affinity groups are the way to group the services in your Microsoft Azure subscription that need to work together in order to achieve optimal performance.  It is very important that affinity groups be part of the solution design to guarantee the highest performance possible within the data center.

When customers create certain resources in an Azure subscription, they can specify to which affinity group the resource will belong. This ensures that all resources created within the same affinity group will be physically near in the data center, which increases the performance in their communications.

Customers should define the affinity groups ahead of time, following the naming convention.

For example, the following command creates the affinity group **AgCalculationEnginePd** for Contoso’s calculation engine application[3]:

[](https://msdnshared.blob.core.windows.net/media/MSDNBlogsFS/prod.evol.blogs.msdn.com/CommunityServer.Blogs.Components.WeblogFiles/00/00/01/63/64/metablogapi/6114.clip_image0214_6D484713.png)

To read more about affinity groups, please see: TechNet: Importance of Windows Azure Affinity Groups

3.4 – Storage Layout

Storage is an integral part of any Azure solution, since not only does it provide application level services, but it is also part of the infrastructure supporting virtual machines.

Things to bear in mind is that storage accounts are bound to scalability targets. At the time of writing, the highest scalability targets are for accounts created after June 7, 2012. These targets were published in the Windows Azure Storage Blog:

v Capacity – Up to 200 TBs

Ø Maximum size per blob – 1 TB

v Transactions – Up to 20,000 entities/messages/blobs per second

v Bandwidth for a Geo Redundant storage account

Ø Ingress - up to 5 gigabits per second

Ø Egress - up to 10 gigabits per second

v Bandwidth for a Locally Redundant storage account

Ø Ingress - up to 10 gigabits per second

Ø Egress - up to 15 gigabits per second

v Throughput for a single blob – up to 60 Mbytes/sec

When creating new virtual machines, Azure creates them with an operating system disk and a temporary disk. The operating system disk is blob-backed, whereas the temporary disk is backed by storage local to the node where the machine lives. This makes the temporary disk unfit for data that must persist during a system recycle, since the machine may silently be migrated from one node to another, losing any data in that disk.

Operating System disks and Data disks have a maximum size of 1 TB since the maximum size of a blob is 1 TB. Customers can implement disk striping in Windows to surpass this limit.

3.4.1 – Striped Disks

Besides providing the ability to create disks larger than 1 TB, in many instances, using stripped disks for data disks will enhance performance by allowing multiple blobs to back the storage for a single volume. This parallelizes the IO required to write and read data from a single disk.

Note that Azure imposes limits on the amount of data disks and bandwidth available, depending on the virtual machine size:

*Table 1 - Bandwidth and Disks for Virtual Machine sizes*

| **VM Size** | **CPU Cores** | **Memory** | **Bandwidth** | **# Data Disks** |
| --- | --- | --- | --- | --- |
| Extra Small | Shared | 768 MB | 5 (Mbps) | 1 |
| Small | 1 | 1.75 GB | 100 (Mbps) | 2 |
| Medium | 2 | 3.5 GB | 200 (Mbps) | 4 |
| Large | 4 | 7 GB | 400 (Mbps) | 8 |
| Extra Large | 8 | 14 GB | 800 (Mbps) | 16 |
| A6 | 4 | 28GB | 1,000 (Mbps) | 8 |
| A7 | 8 | 56GB | 2,000 (Mbps) | 16 |

3.4.2 - Multiple Storage Accounts

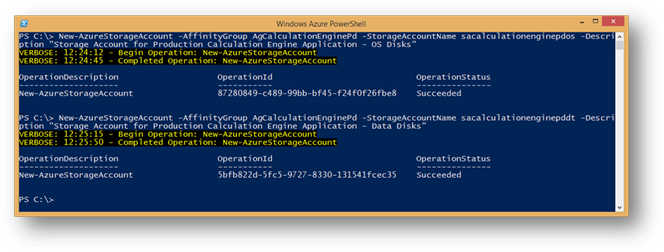
Using multiple storage accounts to back the disks associated to many virtual machines ensures that the aggregated IO of those disks is well below the scalability targets for each one of those storage accounts.

3.4.3 – Storage layout design

In order to implement these strategies to keep the disk subsystem of the virtual machines with good performance, the solution will take advantage of many storage accounts. These will host many vhd blobs. In some instances, more than one blob is associated to one single volume in a virtual machine.

This situation can add complexity to the management tasks. Designing a sound strategy for storage, including appropriate naming for the underlying disks and associated vhd blobs is key.

For example, for Contoso’s Calculation Engine application, two storage accounts are created, one for OS disks and another one for data disks. Note that this distinction is purely arbitrary to show the usage of two accounts. However, many other criteria for multiple storage accounts may be used.

[](https://msdnshared.blob.core.windows.net/media/MSDNBlogsFS/prod.evol.blogs.msdn.com/CommunityServer.Blogs.Components.WeblogFiles/00/00/01/63/64/metablogapi/6607.clip_image0234_36DA25C2.png)

3.5 - Virtual Networking

The next logical step is to create the virtual networks necessary to support the communications across the virtual machines in the solution.

Cloud services provide a communication boundary among computers within that cloud service. These computers can use IP Addresses or the Azure-provided DNS service to communicate with each other using the computer name.

Virtual networks also create a communication boundary, so that virtual machines within the same virtual network can access other computers within the same virtual network, regardless of to which cloud service they belong. Within the virtual network, this communication remains private, without the need for the communication to go through the public endpoints. This communication can occur via IP address, or by name, using a DNS service installed in the virtual network, or on premises, if the virtual machine is connected to the corporate network via a Site-to-Site connection. The Azure-provided DNS service will not aid in name resolution across cloud services, even if they belong to the same virtual network.

3.6 – Subnets

Subnets in Azure virtual networks do not impose communication boundaries, since a virtual network is able to bridge communications across subnets within the same virtual network.

However, subnets can offer a great way to organize resources that are related, either logically (e.g. one subnet for virtual machines associated to the same application, or physically (e.g. one subnet per cloud service.

Customers should design subnets with the same conceptual ideas that they use for on premise resources. Like all other named resources in Azure, subnets should adhere to the naming conventions.

In the Calculation Engine example, a virtual network is defined with the following schema:

<NetworkConfiguration xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"xmlns="http://schemas.microsoft.com/ServiceHosting/2011/07/NetworkConfiguration">

 <VirtualNetworkConfiguration>

   <Dns />

   <VirtualNetworkSites>

     <VirtualNetworkSite name="vnCalculationEnginePd" AffinityGroup="AgCalculationEnginePd">

        <AddressSpace>

          <AddressPrefix>10.0.0.0/8</AddressPrefix>

        </AddressSpace>

        <Subnets>

          <Subnet name="FrontEnd">

            <AddressPrefix>10.0.0.0/24</AddressPrefix>

          </Subnet>

          <Subnet name="BackEnd">

            <AddressPrefix>10.0.1.0/24</AddressPrefix>

          </Subnet>

          <Subnet name="DataServices">

            <AddressPrefix>10.0.2.0/24</AddressPrefix>

          </Subnet>

        </Subnets>

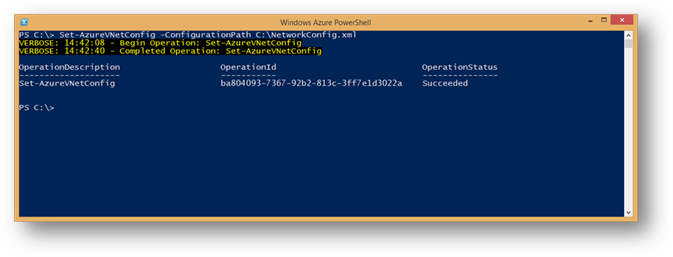
     </VirtualNetworkSite>

   </VirtualNetworkSites>

 </VirtualNetworkConfiguration>

</NetworkConfiguration>

The schema is applied by running the following command:

[](https://msdnshared.blob.core.windows.net/media/MSDNBlogsFS/prod.evol.blogs.msdn.com/CommunityServer.Blogs.Components.WeblogFiles/00/00/01/63/64/metablogapi/6281.clip_image0254_289BACD2.png)

3.7 – Site Connectivity

If virtual networks require connectivity to on premise resources, customers can establish a VPN connection that extends the virtual network’s reach to the corporate network via the VPN tunnel.

After creating a virtual network and defining its subnets, customers can establish Site-to-Site and/or Point-to-Site connectivity if required.

3.8 - Azure Building Blocks

Azure Building Blocks are application level services that Azure offers, typically to those applications taking advantage of PaaS features, although IaaS resources may leverage some, like Azure SQL, Traffic Manager, and others.

These services rely on an array of artifacts that are created and/or registered in Azure. These also need to be considered in the naming convention.

3.9 - Availability Sets

In Azure PaaS, cloud services contain one or more roles that execute application code. Roles can have one or more virtual machine instances that the fabric automatically provisions. At any given time, Azure may update the instances in these roles, but because they are part of the same role, Azure knows not to update all at the same time to prevent a service outage for the role.

In Azure IaaS, the concept of role is not significant, since each IaaS virtual machine represents a role with a single instance. In order to hint Azure not to bring down two or more associated machines at the same time (e.g. for OS updates of the node where they reside), the concept of Availability Sets was introduced. An availability set tells Azure not to bring down all the machines in the same availability set at the same time to prevent a service outage.

Availability sets must be part of the high-availability planning of the solution, and must exist before the creation of cloud services and virtual machines.

In the example of Contoso’s Calculation Engine application, both front-end servers should be part of one availability set, and should be the application servers.

As with all other resources in an Azure deployment, availability sets must follow the naming convention.

3.10 - Cloud Services

Cloud services are a fundamental building block in Azure, both for PaaS and IaaS services.

For PaaS, cloud services represent an association of roles whose instances can communicate among each other. Cloud services are associated to a public virtual IP and a load balancer, which takes incoming traffic and load balances it to the roles configured to receive that traffic.

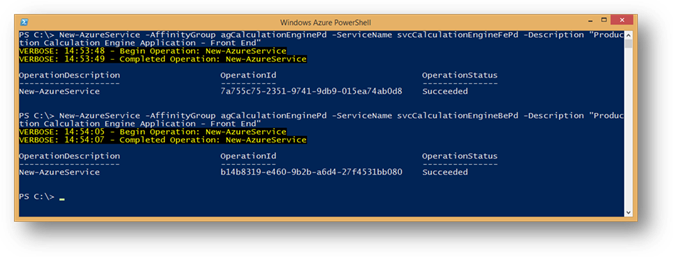
In the case of IaaS, cloud services offer similar functionality, although in most cases, the load balancer functionality is used to forward ports from the public endpoints to the many virtual machines within that cloud service.

Cloud service names are especially important in IaaS since they Azure uses them as part of the default naming convention for disks.

An important point to remember is that Microsoft Azure exposes the cloud service names, since they are associated to the public virtual IP, in the domain “**cloudapp.net”**. For better user experience of the application, a vanity name should be configured as needed to replace the fully qualified cloud service name.

In addition, the naming convention used for cloud services may need to tolerate exceptions since the cloud service names should be unique among all other Microsoft Azure cloud services, regardless of the Microsoft Azure tenant.

As discussed, the Contoso Calculation Engine application will use two cloud services: one for the front-end web servers and another one for the application and database servers:

[](https://msdnshared.blob.core.windows.net/media/MSDNBlogsFS/prod.evol.blogs.msdn.com/CommunityServer.Blogs.Components.WeblogFiles/00/00/01/63/64/metablogapi/2084.clip_image0274_7D5715CA.png)

3.11 - Virtual Machines

In Azure PaaS, Azure manages virtual machines and their associated disks. Customers have the ability to define and name roles, and Azure will create instances associated to those roles.

In the case of Azure IaaS, it is up to the customers to provide names for the cloud services, virtual machines, and associated disks.

To reduce administrative burden, the Azure Management Portal will use the computer name as a suggestion for the default name for the associated cloud service (in the case the customer chooses to create a new cloud service as part of the virtual machine creation wizard).

In addition, Azure names disks and their supporting vhd blobs using a combination of the cloud service name, the computer name, and the creation date.

In general, the number of disks will be much greater than the amount of virtual machines. Customers should be careful when manipulating virtual machines to prevent orphaning disks. Also, disks can be erased without deleting the supporting blob. If this is the case, the blob will remain in the storage account until manually erased.

These issues support the need to control the naming conventions for blobs supporting disks, disks, and virtual machines.

For in the example of Contoso, the virtual machines could be created using the following script:

$img = (Get-AzureVMImage |

    Where-Object OSImageName -Like "\*\_\_Windows-Server-2012-R2\*" |

    Sort-Object PublishedDate -Desc |

    Select-Object -First 1).OsImageName

$mediaLocationOS = "http://sacalculationenginepdos.blob.core.windows.net/vhds/"

$mediaLocationData = "http://sacalculationenginepddt.blob.core.windows.net/vhds/"

$pwd = "<ADMIN PASSWORD HERE>"

$fe1Vm = New-AzureVMConfig `

                -ImageName $img `

                -InstanceSize Small `

                -Name vmCePrdFe1 `

                -AvailabilitySetName asCalculationEnginePdFe `

                -DiskLabel vmCePrdFe1-OSDisk `

                -HostCaching ReadWrite `

                -MediaLocation ($mediaLocationOs + "vmCePrdFe1-OSDiskVer01.vhd") |

            Add-AzureProvisioningConfig `

                    -Windows `

                    -AdminUsername MyAdmin `

                    -Password $pwd |

            Add-AzureDataDisk `

                    -CreateNew `

                    -DiskLabel vmCePrdFe1-DataDisk00 `

                    -DiskSizeInGB 30 `

                    -LUN 0 `

                    -HostCaching ReadWrite `

                    -MediaLocation ($mediaLocationData + "vmCePrdFe1-DataDisk00Ver01.vhd") |

            Set-AzureSubnet `

                    -SubnetNames FrontEnd

$fe2Vm = New-AzureVMConfig `

                -ImageName $img `

                -InstanceSize Small `

                -Name vmCePrdFe2 `

                -AvailabilitySetName asCalculationEnginePdFe `

                -DiskLabel vmCePrdFe2-OSDisk `

                -HostCaching ReadWrite `

                -MediaLocation ($mediaLocationOs + "vmCePrdFe2-OSDiskVer01.vhd") |

            Add-AzureProvisioningConfig `

                    -Windows `

                    -AdminUsername MyAdmin `

                    -Password $pwd |

            Add-AzureDataDisk `

                    -CreateNew `

                    -DiskLabel vmCePrdFe2-DataDisk00 `

                    -DiskSizeInGB 30 `

                    -LUN 0 `

                    -HostCaching ReadWrite `

                    -MediaLocation ($mediaLocationData + "vmCePrdFe2-DataDisk00Ver01.vhd") |

            Set-AzureSubnet `

                    -SubnetNames FrontEnd

$be1Vm = New-AzureVMConfig `

                -ImageName $img `

                -InstanceSize Small `

                -Name vmCePrdBe1 `

                -AvailabilitySetName asCalculationEnginePdBe `

                -DiskLabel vmCePrdBe1-OSDisk `

                -HostCaching ReadWrite `

                -MediaLocation ($mediaLocationOs + "vmCePrdBe1-OSDiskVer01.vhd") |

            Add-AzureProvisioningConfig `

                    -Windows `

                    -AdminUsername MyAdmin `

                    -Password $pwd |

            Add-AzureDataDisk `

                    -CreateNew `

                    -DiskLabel vmCePrdBe1-DataDisk00 `

                    -DiskSizeInGB 30 `

                    -LUN 0 `

                    -HostCaching ReadWrite `

                    -MediaLocation ($mediaLocationData + "vmCePrdBe1-DataDisk00Ver01.vhd") |

            Set-AzureSubnet `

                    -SubnetNames BackEnd

$be2Vm = New-AzureVMConfig `

               -ImageName $img `

                -InstanceSize Small `

                -Name vmCePrdBe2 `

                -AvailabilitySetName asCalculationEnginePdBe `

                -DiskLabel vmCePrdBe2-OSDisk `

                -HostCaching ReadWrite `

                -MediaLocation ($mediaLocationOs + "vmCePrdBe2-OSDiskVer01.vhd") |

            Add-AzureProvisioningConfig `

                    -Windows `

                    -AdminUsername MyAdmin `

                    -Password $pwd |

            Add-AzureDataDisk `

                    -CreateNew `

                    -DiskLabel vmCePrdBe2-DataDisk00 `

                    -DiskSizeInGB 30 `

                    -LUN 0 `

                    -HostCaching ReadWrite `

                    -MediaLocation ($mediaLocationData + "vmCePrdBe2-DataDisk00Ver01.vhd") |

            Set-AzureSubnet `

                    -SubnetNames BackEnd

$sqlVm = New-AzureVMConfig `

                -ImageName $img `

                -InstanceSize Small `

                -Name vmCePrdSql1 `

                -DiskLabel vmCePrdSql1-OSDisk `

                -HostCaching ReadWrite `

                -MediaLocation ($mediaLocationOs + "vmCePrdSql1-OSDiskVer01.vhd") |

            Add-AzureProvisioningConfig `

                    -Windows `

                    -AdminUsername MyAdmin `

                    -Password $pwd |

            Add-AzureDataDisk `

                    -CreateNew `

                    -DiskLabel vmCePrdSql1-DataDisk00 `

                    -DiskSizeInGB 30 `

                    -LUN 0 `

                    -HostCaching ReadWrite `

                    -MediaLocation ($mediaLocationData + "vmCePrdSql1-DataDisk00Ver01.vhd") |

            Set-AzureSubnet `

                    -SubnetNames DataServices

New-AzureVM `

   -ServiceName svcCalculationEngineFePd `

   -VNetName vnCalculationEnginePd `

   -VMs $fe1Vm, $fe2Vm

New-AzureVM `

   -ServiceName svcCalculationEngineBePd `

   -VNetName vnCalculationEnginePd `

   -VMs $be1Vm, $be2Vm, $sqlVm

Customers should ensure that they can track vhd blobs and disks back to the virtual machine for which they were created. For this, naming conventions must have clear rules regarding how names help tie these resources together.

4 - Conclusion

Starting with a naming convention and staying consistent to it goes a long way in keeping the Azure as manageable as possible, while setting up the stage for a long-lasting and reliable infrastructure.

In addition, following the right order in the creation of Azure resources, reduces the amount of re-work and refactoring, while increasing the chances of success.

5 - Glossary[4]

**affinity group**. A named grouping that is in a single data center. It can include all the components associated with an application, such as storage, Microsoft Azure SQL Database instances, and roles.

**cloud**. A set of interconnected servers located in one or more data centers.

**hosted** **service**. See Cloud Service.

**Infrastructure as a Service (IaaS).** A collection of infrastructure services such as storage, computing resources, and network that you can rent from an external partner.

**Microsoft Azure**. Microsoft's platform for cloud-based computing. It is provided as a service over the Internet using either the PaaS or IaaS approaches. It includes a computing environment, the ability to run virtual machines, Microsoft Azure storage, and management services.

**Microsoft Azure Cloud Services**. Web and worker roles in the Microsoft Azure environment that enable you to adopt the PaaS approach.

**Microsoft Azure Management Portal**. A web-based administrative console for creating and managing your Microsoft Azure hosted services, including Cloud Services, SQL Database, storage, Virtual Machines, Virtual Networks, and Web Sites.

**Microsoft Azure SQL Database**. A relational database management system (RDBMS) in the cloud. Microsoft Azure SQL Database is independent of the storage that is a part of Microsoft Azure. It is based on SQL Server and can store structured, semi-structured, and unstructured data.

**Microsoft Azure storage**. Consists of blobs, tables, and queues. It is accessible with HTTP/HTTPS requests. It is distinct from Microsoft Azure SQL Database.

**Microsoft Azure Virtual Machine**. Virtual machines in the Microsoft Azure environment that enable you to adopt the IaaS approach.

**Microsoft Azure Virtual Network**. Microsoft Azure service that enables you to create secure site-to-site connectivity, as well as protected private virtual networks in the cloud.

**Microsoft Azure Web Sites**. A Microsoft Azure service that enables you to quickly and easily deploy web sites that use both client and server side scripting, and a database to the cloud.

**Platform as a Service (Paas).** A collection of platform services that you can rent from an external partner that enable you to deploy and run your application without the need to manage any infrastructure.

**web role**. An interactive application that runs in the cloud. A web role can be implemented with any technology that works with Internet Information Services (IIS) 7.

**worker role**. Performs batch processes and background tasks. Worker roles can make outbound calls and open endpoints for incoming calls. Worker roles typically use queues to communicate with Web roles.

6 - Appendix – Sample naming convention for Contoso’s Calculation Engine

The examples throughout this document follow this sample naming convention for Contoso’s Calculation Engine application:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Resource Type** | **Prefix(es)** | **Name** | **Suffix(es)** | **Example** |
| **Account** | - | Region | - | Americas |
| **Subscription** | - | <area>.<function> | - | Finance.Production |
| **Affinity Group** | Resource Type Ag=Affinity Group | CalculationEngine | Environment:  Dv = Dev  Qa = QA  Pd = Production | AgCalculationEnginePd |
| **Storage Account** | Resource Type  sa=Storage Account | calculationengine | Environment:  Dv = Dev  Qa = QA  Pd = Production    Disk Type:  os = Operating System Disks  dt = Data Disks | sacalculationenginepdos |
| **Virtual Network** | Resource Type  Vn=Virtual Network | CalculationEngine | Environment:  Dv = Dev  Qa = QA  Pd = Production | VnCalculationEnginePd |
| **Subnet** | - | <Descriptive Name> | - | FrontEnd |
| **Availability Sets** | Resource Type  As=Availability Set | CalculationEngine | Environment:  Dv = Dev  Qa = QA  Pd = Production    Role  Fe = Front-End  Be = Back End | AsCalculationEnginePdFe |
| **Cloud Service** | Resource Type  Svc = Cloud Service | CalculationEngine | Role  Fe = Front-End  Be = Back End    Environment:  Dv = Dev  Qa = QA  Pd = Production | svcCalculationEnginePdFe |
| **Virtual Machine** | Resource Type  Vm=Virtual Machine | Ce = CalculationEngine | Environment:  Dv = Dev  Qa = QA  Pd = Production    Role  Fe = Front-End  Be = Back End  SQL = SQL Server    Instance Number  Integer starting on 1 | vmCePrdBe2 |
| **EndPoint** | - | <Descriptive Name> | - | HTTP |
| **Virtual Disks (vhd)** | - | <VM Name>-<DiskType>Ver<Version>.vhd | - | vmCePrdSql1-OSDiskVer01.vhd |