Microsoft

Azure Stack

Building an end-to-end validation environment

By Paul Appleby, Kath McBride, Joel Yoker, and Derek Gamlyn
Azure Customer Advisory Team (AzureCAT)

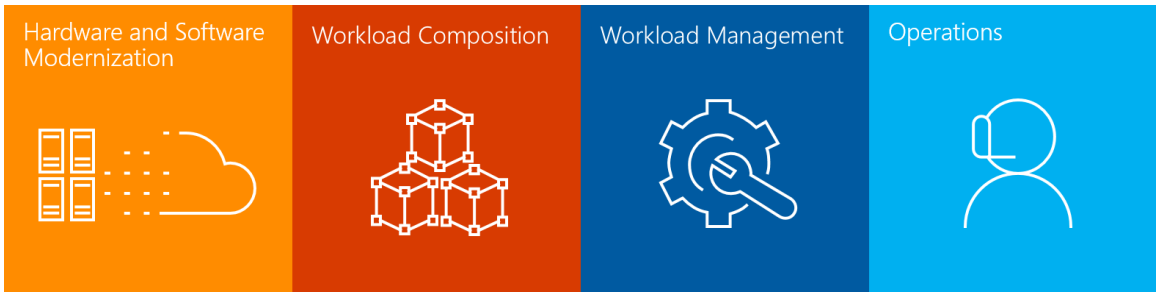
September 2017

Contents

Introduction

Azure Stack is an extension of Azure, bringing the agility and fast-paced innovation of cloud computing to on-premises environments. Organizations can now build modern applications across hybrid cloud environments, balancing the right amount of flexibility and control. Building a validation environment is critical to the success of any deployment of Azure Stack as it helps lay the foundation for cloud architects, operators, and developers within your organization as they plan their environment. Azure Stack differs from traditional on-premises virtualization solutions in four key areas:

- **Hardware and Software Modernization.** Organizations familiar with traditional storage and networking solutions need to learn how software-defined networking and storage models will affect IT service delivery.
- **Workload Composition.** You will need to review current workload architectural models and determine how faster, standardized design and deployment cycles will affect your application and data platforms.
- **Workload Management.** Azure Stack will enable you to adopt a technology lifecycle model that moves at the speed of public cloud releases.
- **Operations.** Azure Stack will require your organization to introduce new operational processes, service models, and oversight that can help your teams make the most of your deployment, immediately.



In this guide, we will provide you with the necessary information to help you plan for an end-to-end Azure Stack validation environment using the Azure Stack Development Kit. We will cover many of the core concepts required to build a functional Azure Stack environment, including subscriptions, services, quotas, plans, and offers. You will learn about the constructs and configurable options available in Azure Stack, and see how to tackle the key considerations that go into planning a successful implementation.

This content is based on our work with early adopter customers and is not intended to be prescriptive guidance. Our goal is to help you understand the foundational elements of Azure Stack to assist you in evaluation and planning. You will find information relevant to the following three scenarios:

- **Architecting for the enterprise.** Enterprises are likely to deploy self-service solutions to meet the needs of their organization and the internal divisions they currently support. While these must be properly secured, the focus tends to be on enabling business

outcomes, reduced time to market, solution cost, availability, and scoping the solution to provide scalability in a cost-effective manner.

- **Architecting for the service provider.** Service providers tend to design solutions that are directed at the specific market they are targeting. Services must be available, secure in a multi-tenant environment, billable, and deliver value to both the customers and the service providers themselves.
- **Architecting for the DevOps scenario.** This scenario applies to either enterprise or service provider solutions and deploys the base services for consumption by developers. Developers are interested in rapidly standing up development environments, integrating with the wide range of tools in use by the organization, and creating applications that can be efficiently deployed to the cloud and updated on a continuous basis. For more information on DevOps, read the article What is DevOps?.

Overview and getting started

To provide a truly hybrid environment for cloud developers, Azure Stack provides on-premises versions of several popular Azure Services. These services include the commonly referenced cloud computing models:

- Infrastructure as a Service (IaaS)
- Platform as a Service (PaaS)

These models can be combined and integrated to build complex, robust solutions for different audiences and use cases. The combination of on-premises IaaS and PaaS services is the primary difference that Azure Stack can bring to your organization since Azure Stack extends these cloud-native services to your datacenter.

PaaS services in Azure Stack are fundamentally different than traditional on-premises virtualization solutions or platforms that host traditional operating systems for developers. They provide an Azure-consistent experience and API that can be consumed by your developers regardless if they exist on-premises or in Microsoft-owned datacenters. These services include App Service (which includes Azure Web Apps, Mobile Apps, API Apps, and Functions), Key Vault, SQL, and MySQL, to name a few. All Azure Stack PaaS services require IaaS to be in place to offer this service to your customers.

Within Azure, services are deployed by Microsoft within an Azure region and made available to subscriptions with defined service limits. In Azure Stack, *you* own the regional planning, deployment, and configuration of these services and the allocation of these services to your subscribed users. Azure Stack subscriptions, services, quotas, plans, and offers are the building blocks used to provide Azure Services to your users. These constructs are connected and define the capabilities and quantities that a user can consume. Figure 1 is a logical diagram of this interaction between these capabilities and shows what is available in a single region.

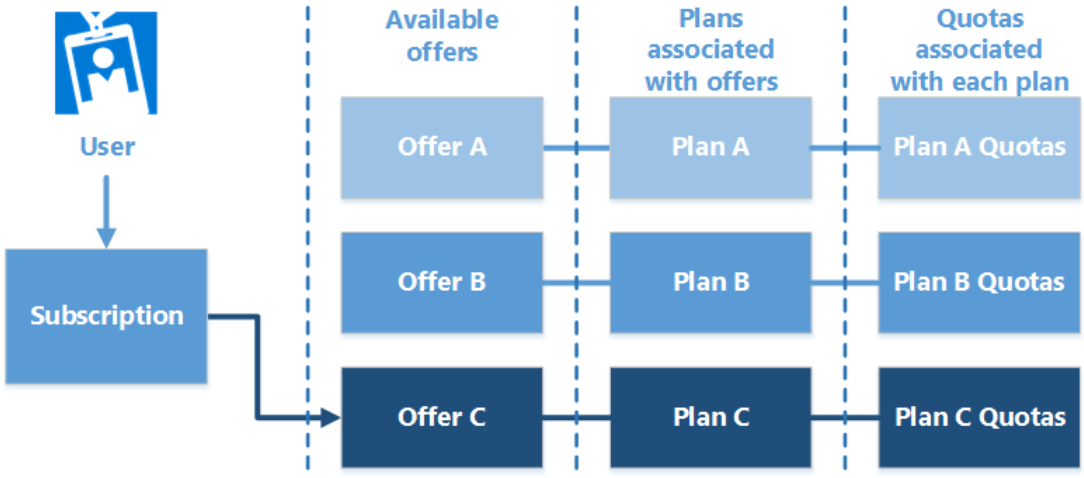


Figure 1: View of how components interact

A high-level overview of this relationship is as follows:

- Users sign up for a subscription.
- Users can browse the available offers and select one for their subscription.

- Each offer is associated with a base plan, and each plan has specific quotas assigned for the resources within the plan.

By selecting an offer, the services within the associated plan and the quotas assigned to those services are assigned to the user.

Deployments

There are two key categories of Azure Stack deployments, each with many common decisions, but some individual considerations:

Customer-owned models (enterprise):

- **Application owners hosting their application in a dedicated Azure Stack instance.** A cloud scenario where an application development team within a customer environment wants to take advantage of Azure Stack capabilities to host their application and make the most of the services managed by enterprise IT.
- **Application divisions (business units) hosting their services in a shared Azure Stack instance.** A cloud scenario where an application development division within a customer environment wants to take advantage of Azure Stack capabilities to host their suite of applications and development/test environment using services managed by enterprise IT.
- **Enterprise IT extending their datacenter infrastructure to Azure Stack.** A cloud scenario where a mature enterprise IT organization wants to extend their existing physical or virtual environment to Azure Stack to support a large number of growing IT requirements for their organization and its customers.

Service provider-managed models:

- **Shared instance.** A cloud scenario where the service provider offers a shared deployment of Azure Stack Azure services to users across multiple organizations. This model applies to a customer base that can coexist with other users in a multi-tenant Azure Stack environment.
- **Dedicated instance.** A cloud scenario where the service provider offers a dedicated deployment of Azure Stack to provide Azure services to users in a single organization. This model applies to a customer base that, due to regulatory compliance constraints or data sovereignty requirements, cannot leverage a shared deployment of Azure Stack.

For both service provider scenarios, customers may choose to consume Azure services using an existing connection to the provider's network or provide their own connection to these services. In most service provider scenarios, the Azure Stack subscription is created, owned, and managed by the service provider, but the customer consumes cloud services by interacting directly with the Azure Stack cloud footprint. The service provider may choose to provide managed services for the customer application workloads deployed on Azure Stack, but this is not required.

Scale units

In Azure Stack, a scale unit is a defined collection of compute (servers), storage, and networking that represents a unit of capacity expansion, an Azure fault domain, and a homogenous set of hardware. A scale unit is always associated with a single Azure Stack region, and in future updates, scale units can be combined within a given region for enhanced scalability.

Regions

Azure Stack uses regions to represent sets of scale units that share a common physical location and are logically managed by a single administrative entity. Multiple regions are optional; you may, for instance, decide to deploy a single Azure Stack instance to meet your business requirements. However, if you have multiple locations and datacenters, or you need to separate the services you offer for compliance reasons, you can elect to have multiple regions in your solution.

In Azure, regions are service-defined boundaries that help users make decisions about where they host workloads within the public cloud. Azure Stack brings new flexibility to both enterprises and service providers, by allowing you to define regions based on your organization's or customer's needs.

Regions allow you to architect your Azure Stack solution to physically manage the delivery of services and applications in a way that is visibly exposed to your users. Regions enable you to deliver services with the following factors in mind:

- **Network latency:** If response time for your applications is a concern, it is important that you deploy applications in close proximity to your users. If you have multiple locations, deploying applications closer to the user improves their experience. Azure Stack will provide methods for ensuring users are directed to the correct region. You must also consider synchronization of data if applications reside in multiple regions.
- **Availability:** Applications need to be highly available and implementing multiple regions in Azure Stack enables you to reduce the impact of an outage. If an outage occurs, users can be redirected to a secondary region, maintaining the availability of the application.
- **Workload segregation:** In some environments, workloads must be kept isolated for security reasons. In Azure Stack, you can deploy multiple regions to provide boundaries between such applications.
- **Scale:** In large environments, your Azure Stack requirements may exceed the capabilities of a single Azure Stack region. In this case, you can use multiple regions to meet customer scale requirements.

Like Azure, many of the capabilities in Azure Stack are region dependent. For instance, marketplace items, role-based access, resource providers, quotas, offers, plans, and resource groups are deployed on a per-region basis. You can configure users to access multiple regions and different services in each, however, when you do this consider latency and workload segregation requirements.

As you design your Azure Stack solution, it is important to understand the number of regions and their placement, as well as the user base that will access each. As new functionality is released, and regions become available, your early planning will allow you to quickly add the regions you require.

Clouds

An Azure Stack cloud is a single instance of Azure Resource Manager and defines a delineation of management in an Azure Stack deployment. A cloud may contain one or more regions consisting of one or more scale units that are managed under a single set of Resource Manager endpoints.

These can be retrieved by enumerating the endpoints and metadata associated with the Azure Stack instance of Azure Services using the PowerShell cmdlet Get-AzureRMEnvironment, as illustrated below in Figure 2.

```
PS C:\> Get-AzureRmEnvironment -Name AzureStackAdmin


Name                                     : AzureStackAdmin
EnableAdfsAuthentication                 : False
ActiveDirectoryServiceEndpointResourceId : https://adminmanagement.[redacted].onmicrosoft.com
AdTenant                                 :
GalleryUrl                               : https://adminportal.local.azurestack.external:30015/
ManagementPortalUrl                      :
ServiceManagementUrl                     :
PublishSettingsFileUrl                   :
ResourceManagerUrl                       : https://adminmanagement.local.azurestack.external
SqlDatabaseDnsSuffix                     :
StorageEndpointSuffix                    : local.azurestack.external
ActiveDirectoryAuthority                 : https://login.windows.net/
GraphUrl                                 : https://graph.windows.net/
GraphEndpointResourceId                  : https://graph.windows.net
TrafficManagerDnsSuffix                  :
AzureKeyVaultDnsSuffix                   : vault.local.azurestack.external
AzureDataLakeStoreFileSystemEndpointSuffix :
AzureDataLakeAnalyticsCatalogAndJobEndpointSuffix :
AzureKeyVaultServiceEndpointResourceId   : https://vault.local.azurestack.external
```

Figure 2: Using the PowerShell cmdlet Get-AzureRMEnvironment

The relationship of each of these elements (scale unit, region, and cloud), using a single region, is shown in the diagram in Figure 3.

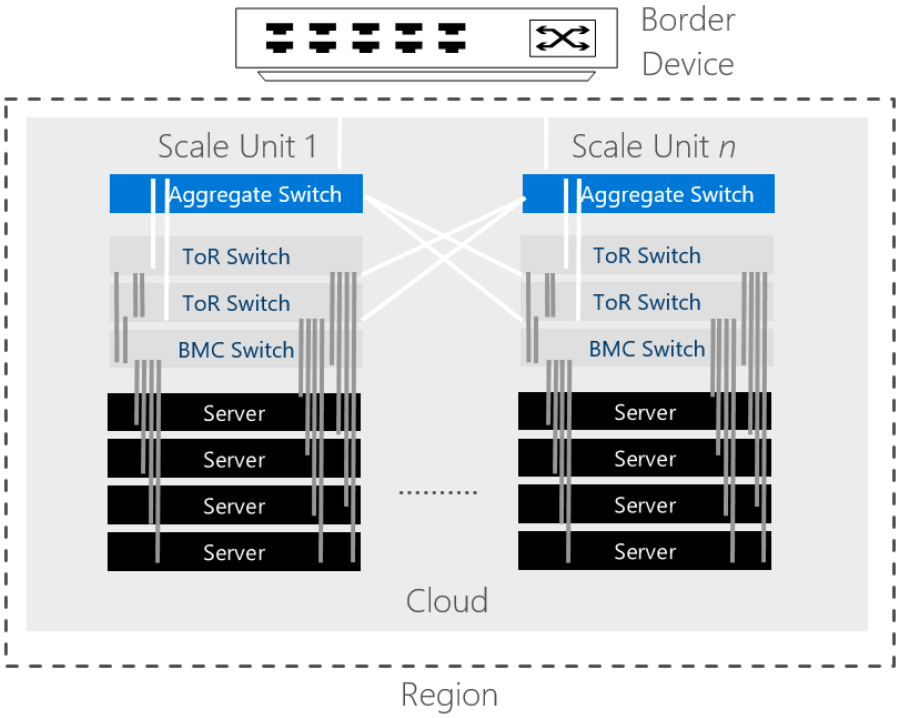


Figure 3. The relationship between regions, clouds, and scale units

Roles

Azure Stack has two general types of users:

- **Azure Stack Operator:** Configures and manages resource providers, offers, plans, services, quotas, and pricing.

- **Azure Stack User:** Acquires (or purchases) services that the Azure Stack operator offers. Users can provision, monitor, and manage services to which they are subscribed.

By default, the Azure Active Directory (Azure AD) account that you use to deploy Azure Stack, will become the primary management account. This account will also be the subscription owner for the "Default Provider Subscription." It is recommended that additional management users be added with rights to manage Azure Stack via the portal or PowerShell.

The administrator used in delegation scenarios is a user account with increased rights enabling the management and configuration of delegated offers for their users. Traditionally, delegated administrators should be part of your overall Azure Stack design. To add additional Azure Stack operators, they must be configured in the Default Provider Subscription as subscription admins. Three levels of delegated rights are available:

- **Owner:** Manages everything, *including* access to resources.
- **Contributor:** Manages everything, *except* access to resources.
- **Reader:** Views everything, cannot make changes.

NOTE: By default, the user that signs up for a subscription will become a subscription administrator.

Marketplace

The Azure Marketplace in Azure Stack is the location where your users will access Azure Stack resources. Items that are available to Azure Stack users in the Marketplace include, but are not limited to, virtual machines, networking components, storage, databases, and websites, depending on the resource providers installed. The four types of offerings you can bring to market on Azure Stack for your users are:

- Existing pre-built solutions
- Virtual Machine images
- Azure Marketplace syndication
- Custom pre-built solutions (consisting of one or more images)

Users are presented Azure services based on the subscribed offers that have been defined by their Azure Stack administrator. Figure 4 shows a user's view of the Marketplace in Azure Stack.

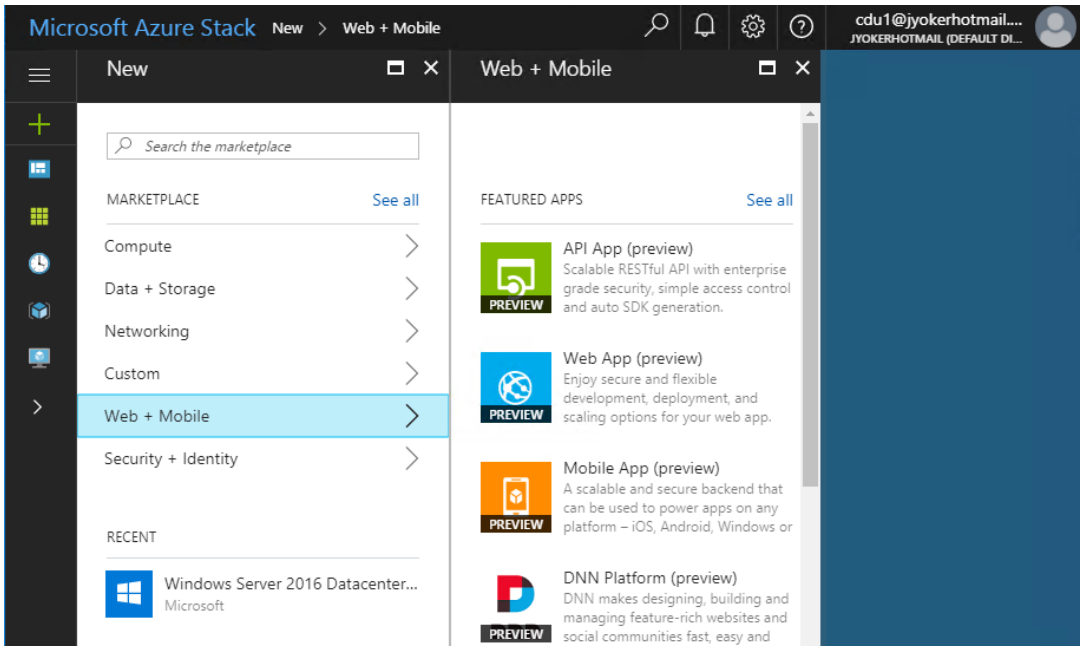


Figure 4: User's view of the Marketplace in Azure Stack

Administrators can also use Marketplace to provision Azure resources in the Default Provider Subscription to support your internal operations. For your users, however, the items that are available are either provided through installed resource providers, custom images, or syndicated items from the commercial Azure Marketplace. You can think of the Marketplace as the catalog of all the items that are available for selection by a user. When designing Marketplace, you first need to determine the requirements of your users. In an enterprise, you will likely need to contact the various business units to determine their use cases and requirements for Azure services. For service providers, you need to consider the needs of your specific customer base.

To provide Azure Marketplace items, they must be syndicated through Marketplace by the Azure Stack operator. Getting syndicated requires Azure Stack registration with a public Azure subscription. Registration of Azure Stack with Azure can be performed using an online or offline process. For Azure Stack operators, this is one of the first steps that should be performed after Azure Stack is initially deployed. Once added, Marketplace items can be made accessible to Azure Stack subscription users through Marketplace management. The types of items that can be syndicated from Azure include virtual machine images, virtual machine extensions, and vendor pre-built solutions. Figure 5 shows the Administrator's view of Marketplace management in Azure Stack.

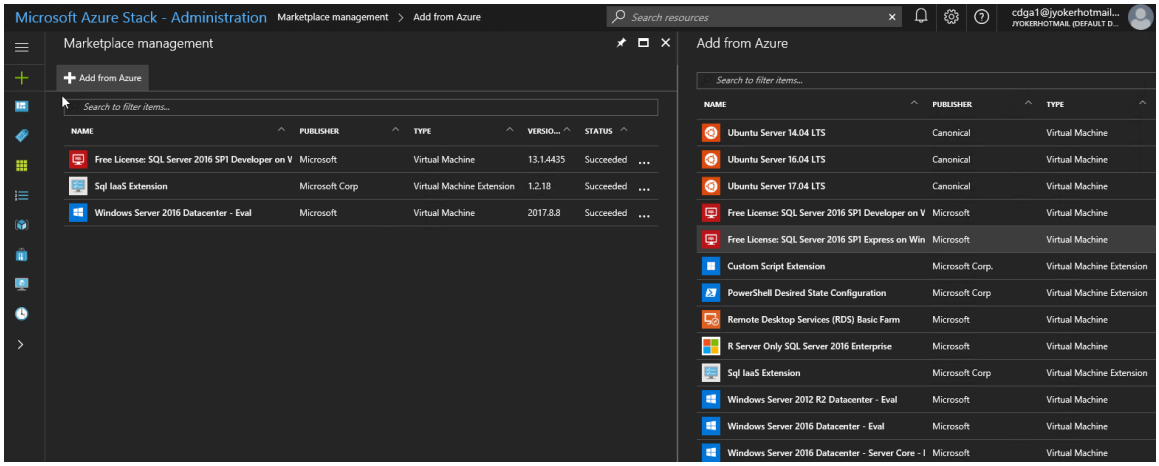


Figure 5: Administrator's view of Marketplace management in Azure Stack

In addition to syndicated items from Azure, you may have custom virtual machine images that contain configurations and settings that are specific to your organization's users. You can easily add things like Linux images, additional operating system virtual machine images, and custom images to the Marketplace without using the syndication feature. Read Deploy Linux virtual machines on Azure Stack for information on adding Linux images, as well as instructions to create your own Linux image.

An important point to consider is that all Marketplace items are visible to all subscribed users in Azure Stack. You can regulate this to a certain degree when you design your offers and plans, but it requires careful planning. For instance, if a specific set of users does not need access to deploy SQL databases, you can create an offer that does not include that service. You must also ensure that all offers are not public.

For deployment of Marketplace resources in Azure Stack, you may also be able to use the templates available for Azure. Resource Manager templates allow you to declaratively describe the resources that are deployed to an Azure resource group. The easiest way to do this is to use Azure Stack-specific Resource Manager templates. However, each template must be reviewed carefully to ensure the parameters, variables, resources, and schema match those that are available in the current version of Azure Stack. To assist with rationalizing the differences between Azure and Azure Stack Resource Manager templates, a template validation tool is available in the Azure Stack tools repository. The template validation tool provides offline analysis of a specified Resource Manager template and provides reporting on potential compatibility challenges between the template's use across Azure and Azure Stack environments.

For more information on the Marketplace, please see the following resources:

- The Azure Stack Marketplace overview
- Create and publish a Marketplace item
- Download marketplace items from Azure to Azure Stack
- Make a custom virtual machine image available in Azure Stack

Interacting with Azure Stack

One of the most important design goals for Azure Stack is providing consistency with Azure to

support an unparalleled hybrid experience for organizations that are looking to embrace and leverage Azure Services across premises. The consistency between Azure and Azure Stack is built at the API level, so you can use familiar tools to build solutions and manage Azure Stack including the Azure portal in Azure Stack, Azure PowerShell, the Azure cross-platform command line tools (CLI), Visual Studio, or the Azure API natively.

Using the portal

The portal is an excellent way to start using both Azure and Azure Stack and remains a good option for experimenting and performing one-off tasks. By contrast, command line tools excel as you move towards a more DevOps approach to managing your applications and services in an automated fashion.

The portal allows administrators to do the following:

- Manage access.
- Manage accounts.
- Manage subscriptions.
- View usage summary.
- Provision/de-provision Azure Stack resource providers.
- Create plans and offers.
- Manage co-administrators on subscriptions.
- View the health of Azure Stack.

The portal is the method most users will use to initially access the services and solutions you offer in Azure Stack. Once the user logs in, the portal is "scoped" based on their credential and the access rights you have defined for that user. For example, a general user will access the portal and sign up for a subscription to begin to consume Azure services from your instance of Azure Stack. Based on the plan and quota parameters you define, they will be given access to compute, storage, and network resources. These are the minimum services required to be able to deploy a virtual machine, storage account, or network as a user. Additional resource providers can be added, such as SQL Server resource provider, and access for your users provided via plans.

Users will be able to sign up for subscriptions through the offers created by the administrator and deploy compute, network, and storage as well as any other enabled services.

For information on using PowerShell and cross-platform command line tools, please see the following resources:

- Install PowerShell for Azure Stack
- Install and configure CLI for use with Azure Stack
- Understand the structure and syntax of Azure Resource Manager templates will help you in using Visual Studio.
- Resource Manager REST APIs will help you in connecting to Azure Stack via APIs.

Quotas, plans, and offers

The bulk of the design decisions you need to make once Azure Stack is installed are related to the configuration of the services offered to your users. In Azure, services are designed, deployed, and offered to various regions by Microsoft engineers and administrators. In Azure Stack, the cloud operator deploys these services and makes them available to subscribed users using quotas, plans, and offers. Each of these elements is connected and defines both the capabilities and quantities that users can consume. The related items required to offer services in Azure Stack include regions, subscriptions, services, offers, plans, and quotas.

- **Quotas** determine the amount of resources a user can consume.
- **Plans** allow administrators to group services and their quotas to be offered to users.
- **Offers** allow administrators to group plans to be offered to users.

Figure 6 shows a diagram outlining how these elements interact.

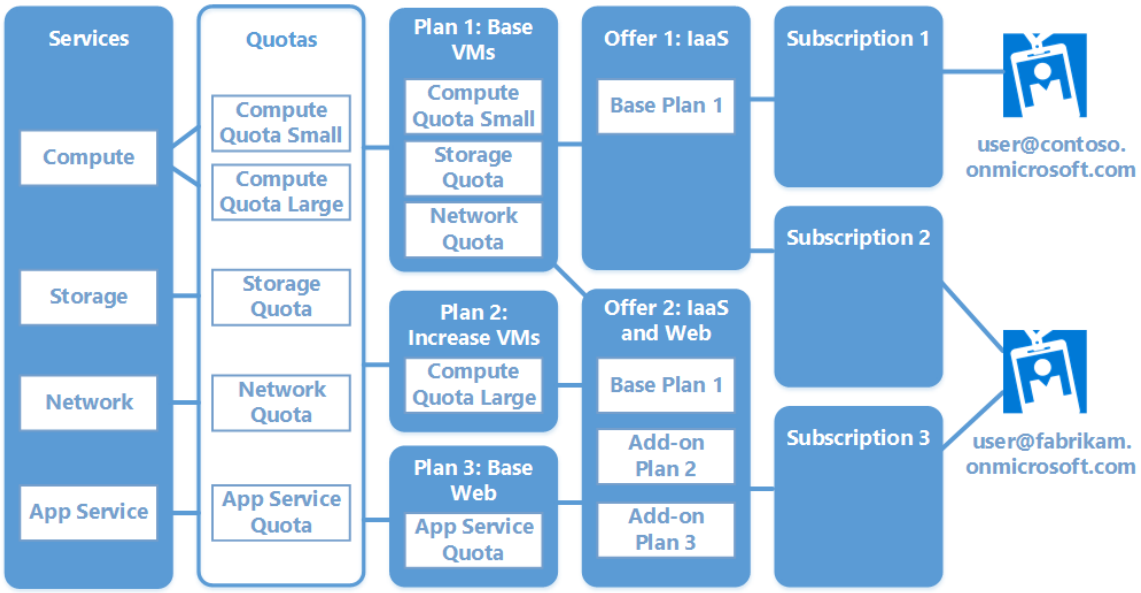


Figure 6: Quotas, plans, and offers in Azure Stack

There are some basic steps you must follow when designing your solutions in Azure Stack:

1. Deploy the Azure services you want to deliver to your users.
2. Create quotas to define the amount of resources you want users to be able to consume from these services.
3. Create plans that contain the desired services and applicable quotas.
4. Create offers that align to the plans.
5. Create add-on plans as needed.

Once you have completed this, your users can create subscriptions aligned to the offers you have created. Let's dive into each of these components, look at how each works, and identify some things you will want to keep in mind when offering services on Azure Stack.

Quotas

After you have identified the services you wish to offer to your users (see Services), you need to determine how much of each service your users may consume. To do this in Azure Stack, you must configure quotas for each resource provider. Note that you can set only the upper limits for quotas.

It is critical to keep in mind the amounts of each resource, such as storage and public IPs, that you have available in your environment, and how much quota is assigned. This awareness ensures you do not exceed capacity unexpectedly. A basic view of capacity within the infrastructure can be reviewed in the Administrative portal by selecting **Region Management** > **Local (Region)** > **Resource Providers**, and selecting **Capacity**.

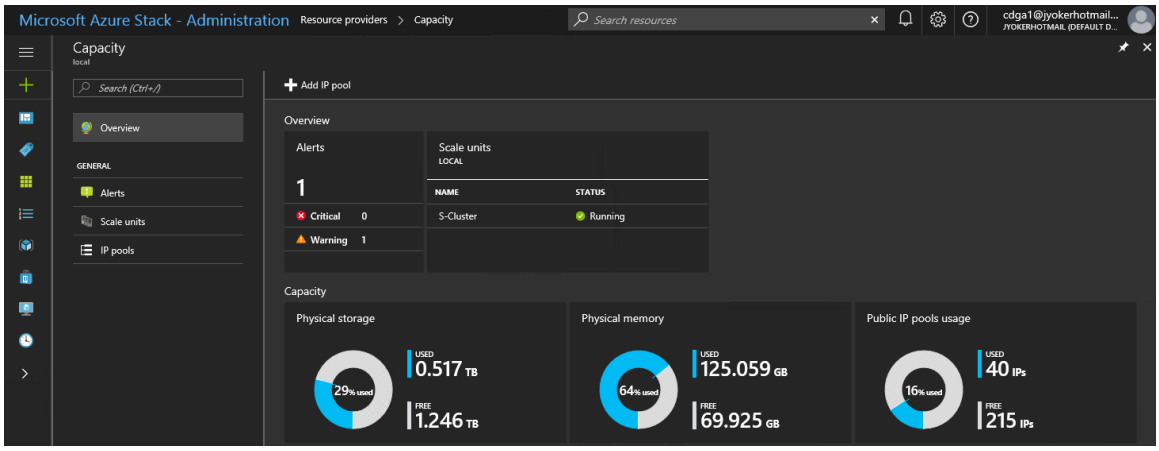


Figure 7: Capacity within the infrastructure as seen in the Administrative portal

Additionally, you can review the configured quotas by selecting each resource provider.

Resource Provider	Configurable Quotas	Default Quotas
Compute	Yes	Default
Key Vault	No	Unlimited
Network	Yes	Default
Storage	Yes	Default
MySQL	Yes	None
SQL	Yes	Default Basic Default Standard Default Premium Default Admin
App Service	Yes	Default

In most cases, you will have several different quotas that align to the needs of different business

groups or customers. Several resource providers offer a default set of quotas, which often have the maximum amount of resources available that can be configured for a resource provider.

> **NOTE:** Quotas may not necessarily align with the actual resources available in your Azure Stack environment.

It is highly recommended that you review the default quotas and add your own to meet your organization's needs best. For example, your organization may want to configure quotas based on the various business units that consume resources, for instance, to have specific quotas for developers. You may also choose to configure quotas based on a specific application.

When defining your quotas, keep in mind the following:

- **Infrastructure resources:** The amount of compute, storage, and network resources available in your Azure Stack environment.
- **Resource providers installed:** Each resource provider, except Key Vault, has quotas that must be configured. All resource providers consume at minimum compute, network, and storage resources.
- **Number of subscriptions:** Each subscription consumes resources, most notably compute, storage, and network. Depending upon the services offered to them, they could also consume other resources, such as SQL databases. It's important to keep track of your total allocation of resources and services across subscriptions.
- **User requirements:** Designing quotas specific to an application is a relatively easy process because the compute, storage, and network requirements of the application are known. To create quotas for groups of users, you should understand their resource usage patterns: whether they deploy a set of resources with *limited* growth or with a *predictable* growth rate, and if they tend to be volatile users, like developers—deploying and deleting on a regular basis.
- **Applications:** Each application instance requires a specific group of resource providers and a known amount of resources. This quota tends to be the easiest one to define.

Defining quotas typically requires discussions with the various user groups within an enterprise, including developers of the applications consuming these services to precisely determine their needs. In most cases, it is recommended that this is periodically reviewed as usage patterns change or new services are offered in your Azure Stack environment. For service providers, defining quotas is more aligned to the offerings you would like to make available to the customer base.

Although you can create quotas at the same time you define your plans, we recommend creating quotas separately. This way, you can be sure to deploy exactly the quotas you need.

Read the article Plan, offer, quota, and subscription overview for step-by-step instructions when you are ready to configure quotas.

Plans

You use plans to create the groupings of services offered to users, along with their applicable quotas. Most designs include more than one plan. Each plan is tied to a specific set of quotas

based on the services offered in that plan.

The first step to designing your plans is to understand the following:

- Services to be included in each plan.
- Differences between how these services are provided to users.
- Users who will access each plan.
- Whether users are assigned to specific plans and offers, or you advertise them instead.
- Quota limits required for each service in a plan.

You can configure users to consume one plan or multiple plans within a single subscription. This option may be beneficial when a user requires more or different services that are not offered within a single plan or these services are differentiated in some way such as quota, performance, and capability.

Two distinct types of plans can be configured on Azure Stack:

- **Base plan:** Contains the core services to be offered to customers.
    - One base plan can be added per offer.
- **Add-on plan:** Contains additional services that can be added to a base plan.
    - Used to extend compute/storage/network to a base plan.
    - Can add multiple add-ons to a base plan.
    - Cannot be used in an offer by itself. To do this, it would need to be created as a base plan.

For a service provider, add-on plans could be upgrade offerings to a base plan. Although combining everything in a single plan may be optimal in some cases, you may want to have a base plan, and offer additional services using add-on plans. For instance, a service provider could decide to offer IaaS services as part of its base plans, with all PaaS services treated as add-on plans with additional charges or to upsell additional services. Similarly, a small organization might decide that one base plan is sufficient, but also offer SQL as an add-on to select users, or for select applications that require it.

Plans can also be used to control consumption of resources in your environment. For example, if you want your users to be mindful of their resource usage, you could have a relatively small base plan (depending on the services required) and as users reach capacity, they would be alerted that they have already consumed the allocation of resources based on their assigned plan. From there, the users may select an available add-on plan for additional resources.

Naming and organizing plans

Like other Azure services, establishing a naming standard within your organization is recommended when configuring plans. Among other benefits, this allows you to easily identify its purpose and track the use of defined plans. Plans have both a display name and a resource name, where the resource name may not contain spaces and must consist of letters, numbers, and a small set of nonalphanumeric characters. Figure 8 shows an example of the creation and naming of a new plan.

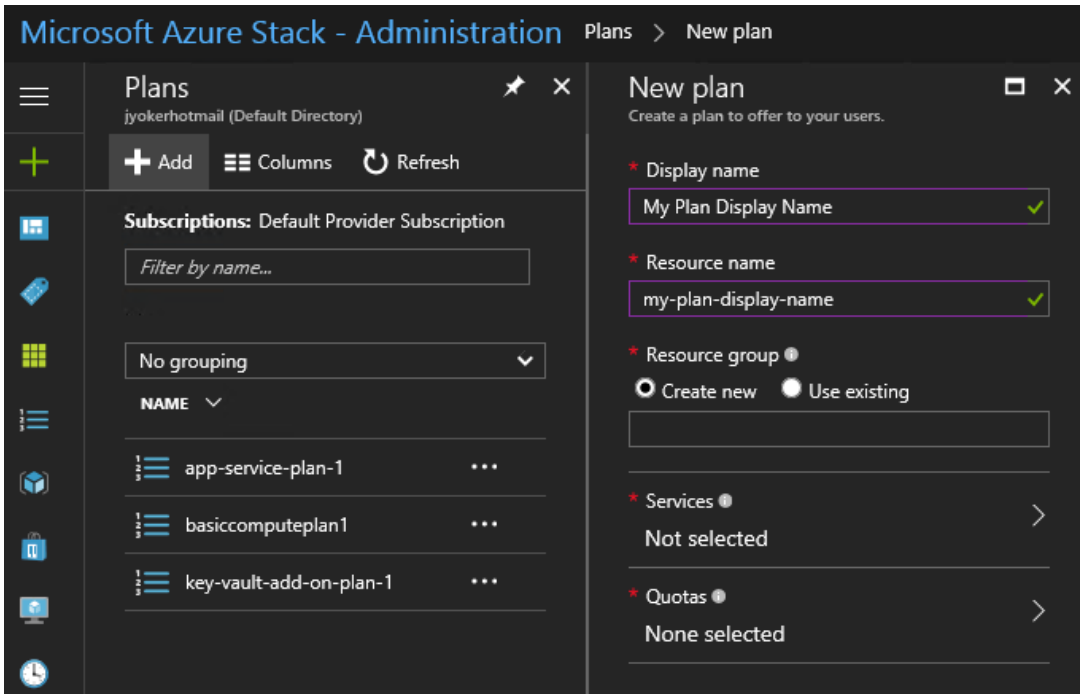


Figure 8: Creation and naming of a new plan as seen in Administration Portal

While naming conventions are subjective, it is recommended to include the type of plan, business unit, sizing, customer name, or the application for which the plan is designed. Below are a few high-level examples of plans. You can modify them to fit your specific needs. These examples show base plans only; add-on plans could easily be included.

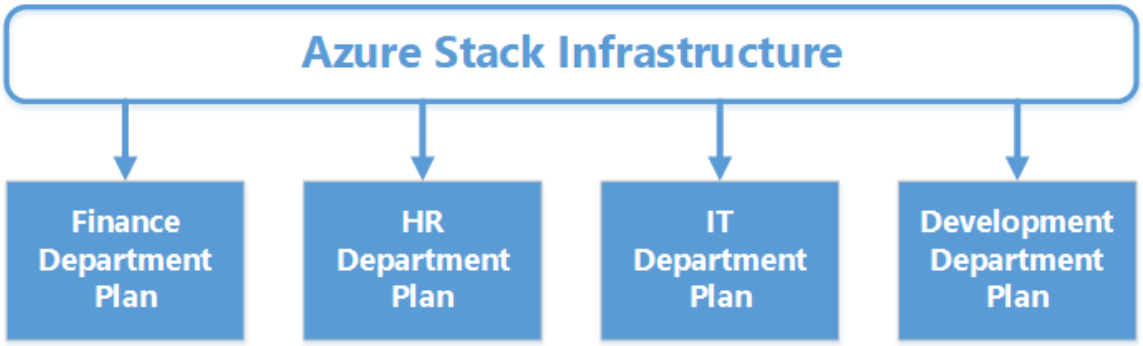


Figure 9: Enterprise business unit-based plans

An enterprise may organize its plans based on departments, as shown in Figure 9.

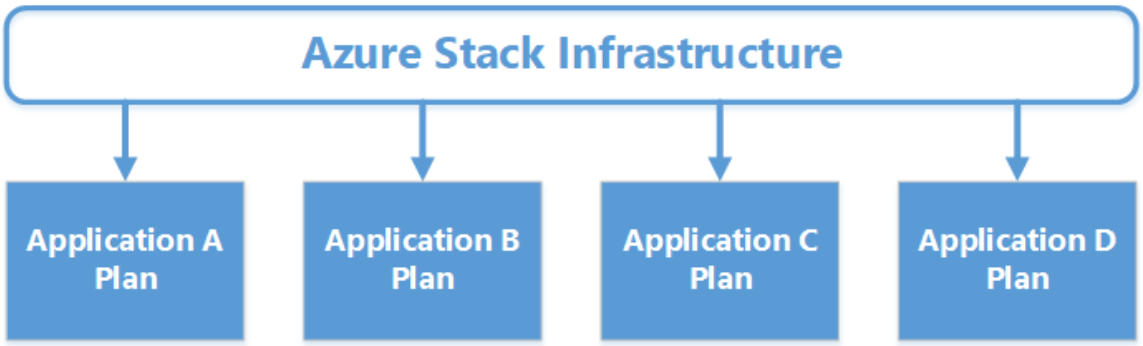


Figure 10: Enterprise application-based plans

The example in Figure 10 is based on applications deployed within Azure Stack that are available across business units within an organization.

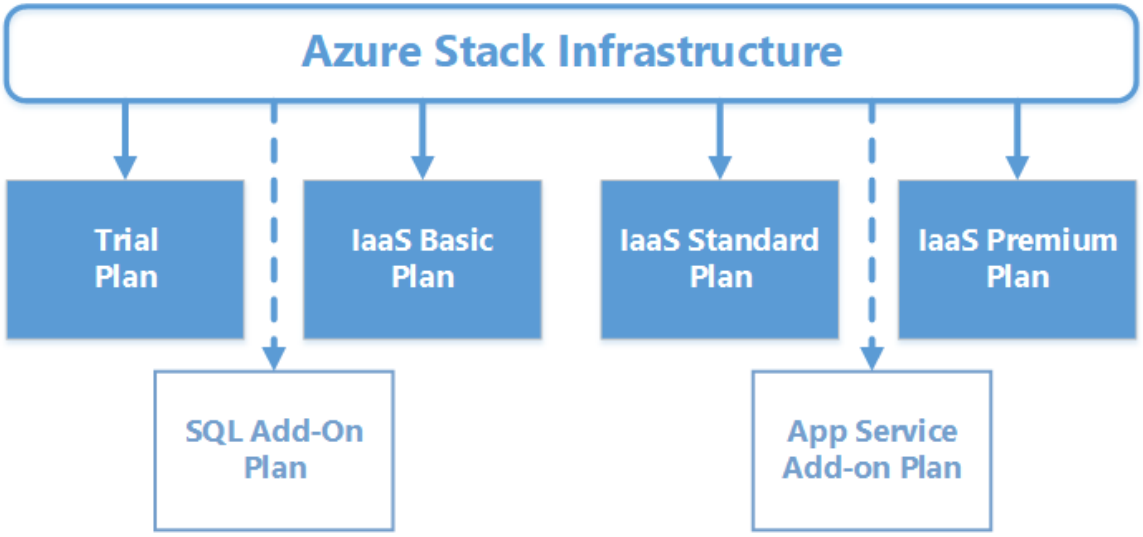


Figure 11: Service provider-based plans

Figure 11 shows a service provider example that represents customers accessing a service provider infrastructure to obtain IaaS/PaaS services.

As you can see from the above examples, plans can get quite complex. Each plan you create has some administration required, and as you add more plan options, the planning calculations for resource consumption become more difficult.

Read the article Create a plan in Azure Stack for more information and step-by-step instructions.

Offers

Your users access the resources they require by selecting an offer when creating a subscription. When an Azure Stack user uses the portal to sign up for a new subscription, they can only select a single offer. Once the subscription is created, they can select additional add-on plans. As an Azure Stack operator, you create the add-on plans and make them available, so your users can get the additional resources they require for their solutions.

At a minimum, an offer must contain at least one base plan. You can add multiple base plans and add-on plans to an offer, providing the users who consume this offer with exactly the resources

they require. All users subscribed to a specific offer have access to the plans and associated quotas that are contained in the offer.

When creating a subscription, you must align it to an offer. Thus, the offers and plans that you want to align to your various subscriptions must be in place first.

Read the article Create an offer in Azure Stack for more information and step-by-step instructions.

Private and public offers

Offers can have one of three states: public, private, or decommissioned. An important decision is whether your offers are private or public. By default, all new offers are set to a private state. The choice to leave an offer private and assign user subscriptions to it is up to the administrator creating this offer. One reason to maintain an offer as private is to ensure that administrative oversight is required to subscribe to specific offerings (see Advertised and assigned offers). User subscriptions must be assigned to private offers by the cloud administrators, increasing the workload to manage this on an ongoing basis. To help mitigate this, you can use delegation (see Delegation), in combination with private offers, to assign the rights necessary to make delegated offers available to users.

When you make an offer public, all users can see and select it when they sign up for a subscription. It is the simplest method of providing offers that will be visible to all of your users when creating subscriptions.

Decommissioned offers are an important part of planning the lifecycle of an offer or set of offers in your Azure Stack environment. Decommissioned offers are those offers that are available to existing subscribers, but new subscriptions are unable to select. As you plan the long-term use of your offers, you may wish to transition users from an originally designed offer to new offers in your environment. Discontinuing additional use of the original offers while maintaining service for existing users of those offers is key towards maintaining continuity of service for your users.

Advertised and assigned offers

Offers can either be advertised or assigned. When you make an offer advertised, it is visible to all users accessing the portal and follows the self-service model for obtaining services. Therefore any user with rights to access the portal can add an advertised plan to a subscription. Advertised offers must be set to public state as outlined above.

To create a subscription and configure an advertised offer, follow the instructions in Subscribe to an offer.

Assigning offers allows you to configure multiple offers that are not visible to your users and perform directed assignment to support your desired usage pattern. For users to access the offer, you must assign their user ID to the offer. Once assigned, a user will see the new subscription and have access to the resources and quotas aligned to the offer when they access the portal.

Cases where assigning offers will benefit you:

- As a service provider, you want to assign company specific offers to customers.
- Within an enterprise, you want to ensure only users of a specific business unit have access to an offer.
- You want to manage tightly the resources that users can consume.

As mentioned earlier, there is an overhead to this approach as every user must be assigned to the applicable offers by an Azure Stack operator. You should carefully think before you define your offers as assigned and depending on your ability to manage this on an ongoing basis, use this approach in a limited fashion.

Delegation

Delegation provides the ability to put other non-administrative users in charge of creating offers and signing up users based on offers that the Azure Stack operator defines. As a service provider, you may want to offload the creation of offers and signing up users to authorized resellers that are purchasing your services. In an enterprise, you can delegate the same ability to add users and create offers to the various divisions, or subsidiaries. This offloads to local resources the day-to-day operations of the administrator tasked with operating Azure Stack.

Configuring delegation is relatively easy, with significant benefits for the operator. There are three roles used when setting up delegation:

- **Azure Stack Operators:** An administrative user who manages the Azure Stack infrastructure, creates an offer template, and delegates the offer to others.
- **Delegated Provider:** A standard user who creates offers based on the delegated offer and assigns users to subscriptions.
- **Azure Stack Users:** A standard user who signs up for offers, or can be assigned the offers.

Figure 12 shows a diagram of how the delegation process works:

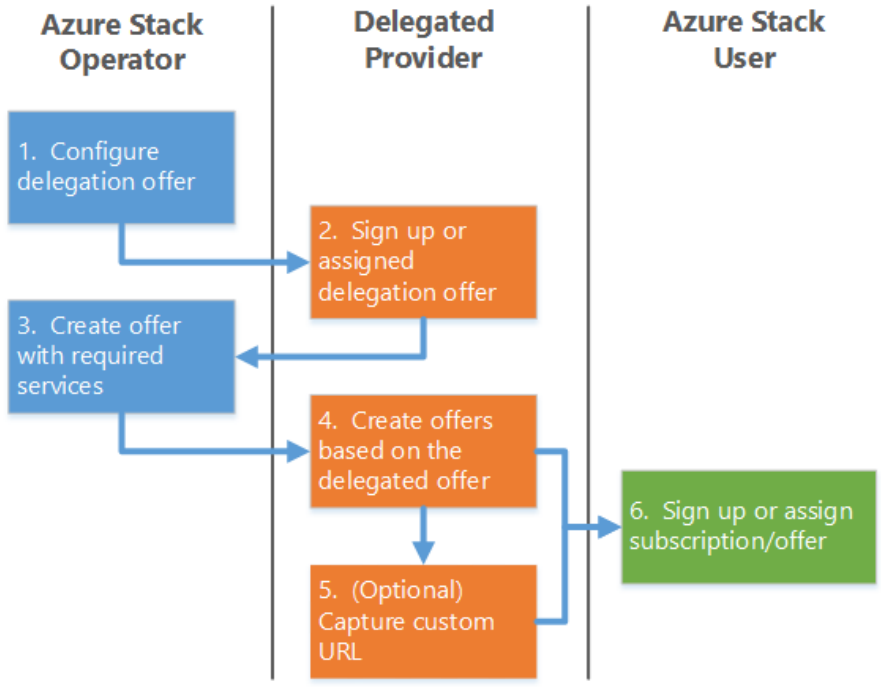


Figure 12: Delegation process

Read the article Delegating offers in Azure Stack for step-by-step instructions on configuring delegation.

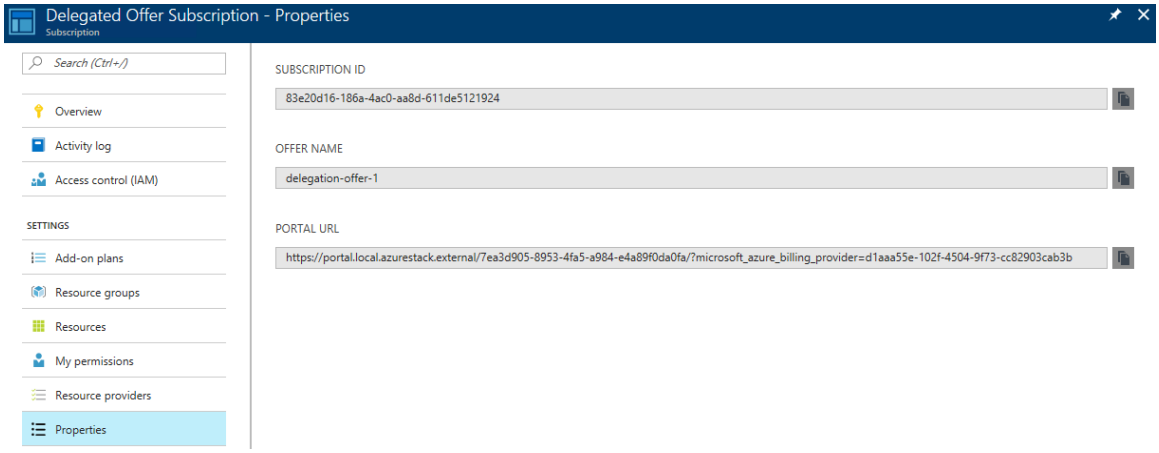


Figure 13: Properties and custom URL created when configuring delegation

Important points

- Role-Based Access Control (RBAC) can be configured to secure access to these (and other) resources.
- Any changes to offers apply to all users consuming the offer.
- Offers that are public are visible to all users.
- Quotas can be configured that exceed the capacity of your infrastructure, causing the provider to run out of resources.
- Additional capacity can be applied to a quota. However, users already consuming the current plan and quota will not see the updates for up to an hour.
- When a user adds an add-on to an existing subscription, the additional resources could take up to an hour to appear.
- To add resources immediately, you may need to configure a new add-on for users to consume.
- When a user runs out of capacity, they receive errors as they try to create new resources within their subscription.
- To change plans, users require a new subscription.
- When a plan or offer is decommissioned, users already consuming the plan and quotas continue to have access to those resources.
- When a plan or offer is removed, users already consuming the plan and quotas continue to have access to those resources.
- If you change a quota amount to reduce it, users already consuming more than the new quota amount continue to receive the quota they already are consuming but are not able to add more resources.
- Changes to a quota, plan, or offer are global and impact all users.

Subscriptions

Azure Stack subscriptions grant users access to the Azure Stack services being offered, as well as to the Azure Stack portal itself. Subscriptions are the first thing a user sees when beginning to use Azure. Like Azure, Azure Stack subscriptions provide a logical boundary of scale, administration, and billing for your users who are consuming services from Azure Stack.

- **Scale:** Subscriptions are a logical limit of scale by which resources can be allocated. These limits include quotas of various resource types offered by an administrator. Scalability is the key element for understanding how the subscription strategy will account for growth as your user's consumption of Azure services increases.
- **Administration:** Resource Manager provides a granular RBAC model for assigning administrative privileges at the resource level, although administration of Azure Stack resources is ultimately defined at the subscription level.
- **Billing:** Consumption of Azure services in Azure Stack is measured at the subscription level, forming a logical billing unit. Additionally, Resource Manager provides the ability to assign resource tags (see Tags) to provide additional information for granular chargeback/show back scenarios; this can be done based on either a resource group or resource.

Configuring Azure Active Directory users

You will need to configure your users in Azure Active Directory (Azure AD) to allow them to access resources within Azure Stack. Azure Stack users are usually configured as Azure AD users, while Azure Stack operators completing the infrastructure installation will need to be provisioned with global admin organizational rights. Users need to be members of your Azure AD domain and will use the account and password assigned to them when logging into Azure Stack. Figure 14 shows where to configure users in Azure AD.

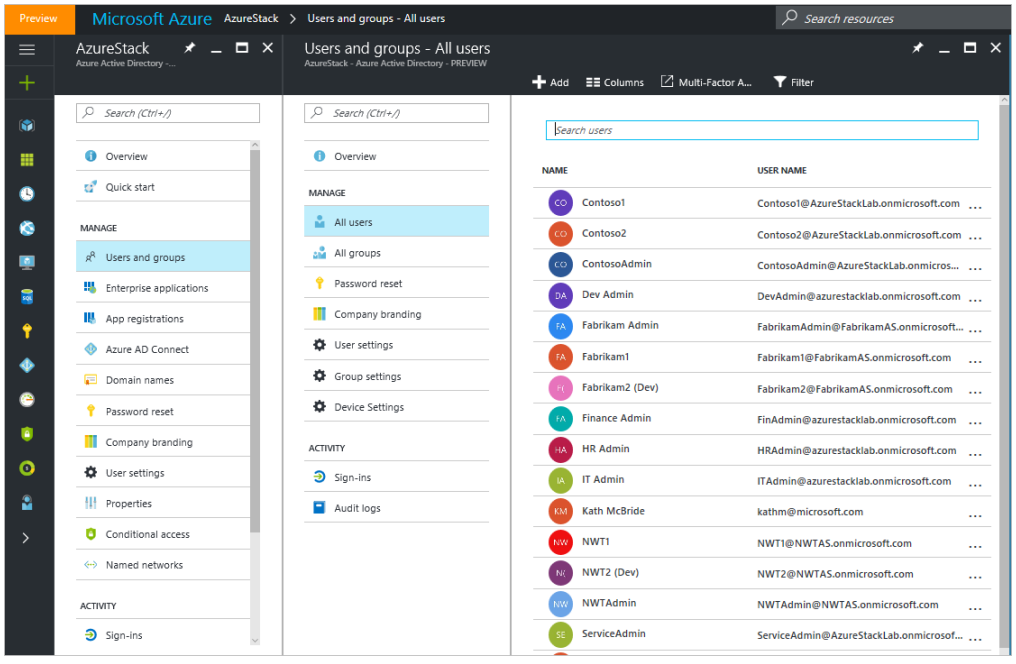


Figure 14: Configuring users in Azure AD

Read the article Add a new Azure Stack account in Azure Active Directory for additional instructions.

Defining subscriptions

There are a few aspects to consider when defining a subscription:

- The subscription governs access to and use of the services via offers and plans to which the user will subscribe.
- The Azure Stack operator manages the services offered to users in the subscription.
- An Azure AD account is required, as this is how usage and billing will be managed.

We can think about how to use subscriptions in two key ways:

- **Self-service:** The user selects Get a Subscription and browses the available offers their Azure Stack operator has created, and creates their own subscription.
- **Assigned:** The administrator (or delegated administrator, see Delegation) creates subscriptions with offers, plans, and quotas aligned to them, and then assigns them to users.

Many of your early decisions in architecting and planning an Azure Stack environment, and related subscriptions, can have an impact on future decisions and designs as the cloud environment grows. Get participation and input from several groups within your organization, including IT leadership and those responsible for networking, security, and identity within your organization.

To help you in defining your subscriptions, consider whether you require:

- Different configurations of offers and plans for various user groups or organizations.
- Segregation of users for security or compliance reasons.
- Large-scale consumption or administrative isolation for a specific application.
- Separate billing and usage monitoring for user groups or organizations.
- Co-administrators from business units, departments, or other organizations.

Considering multiple subscriptions

A single subscription will work just fine for most of your user organizations, but others may require multiple subscriptions for more granular control.

Using multiple subscriptions can create complexity. If isolation is required, then you may opt for subscription administration. Some considerations for multiple subscriptions include:

- A subscription on its own does not carry any costs.
- A subscription has its own administrators, and multiple subscriptions may require additional admins.
- You need to overcome limits per subscription, such as number of virtual networks or CPU cores.

Multiple subscriptions may introduce additional complexity when you consider that the on-premises networking and security infrastructures are typically shared resources within an organization. For most large organizations, core IT services such as patch management, monitoring, and auditing are frequently provided by dedicated staff who are trained in a set of centralized solutions. As you design your subscriptions, be aware that you may need to extend or duplicate services, including monitoring, patching, and antivirus to span these services across subscriptions.

Creating and deleting subscriptions

As an Azure Stack operator, you can create or delete subscriptions via the portal or through the Azure Stack PowerShell module. This applies to both administrator subscriptions and user subscriptions. But be careful, because when you delete a subscription you will remove all resources that are part of that subscription, and recovery from backups will be your only option.

> **NOTE:** A user who creates a subscription can also delete their own subscriptions.

Choosing a subscription model

It is critical to develop your subscription, fabric, and administrative models together to have a cohesive approach towards subscription design. Understanding each component's considerations, constraints, and how each impacts the others is critical to building an Azure Stack solution that can scale and be flexible enough to support the needs of your business.

Self-service subscription model

If your organization allows for users to consume IT services on-demand or is choosing to monetize your Azure Stack deployment, you may want to leverage self-service for user subscriptions. The same care around configuring offers and plans is needed; however, each user will be allowed to get a subscription of the size and type they require directly from the Azure Stack portal. Once the subscription is created, assigned users can start consuming the services that are offered. Figure 15 provides an example of the self-service subscription model.

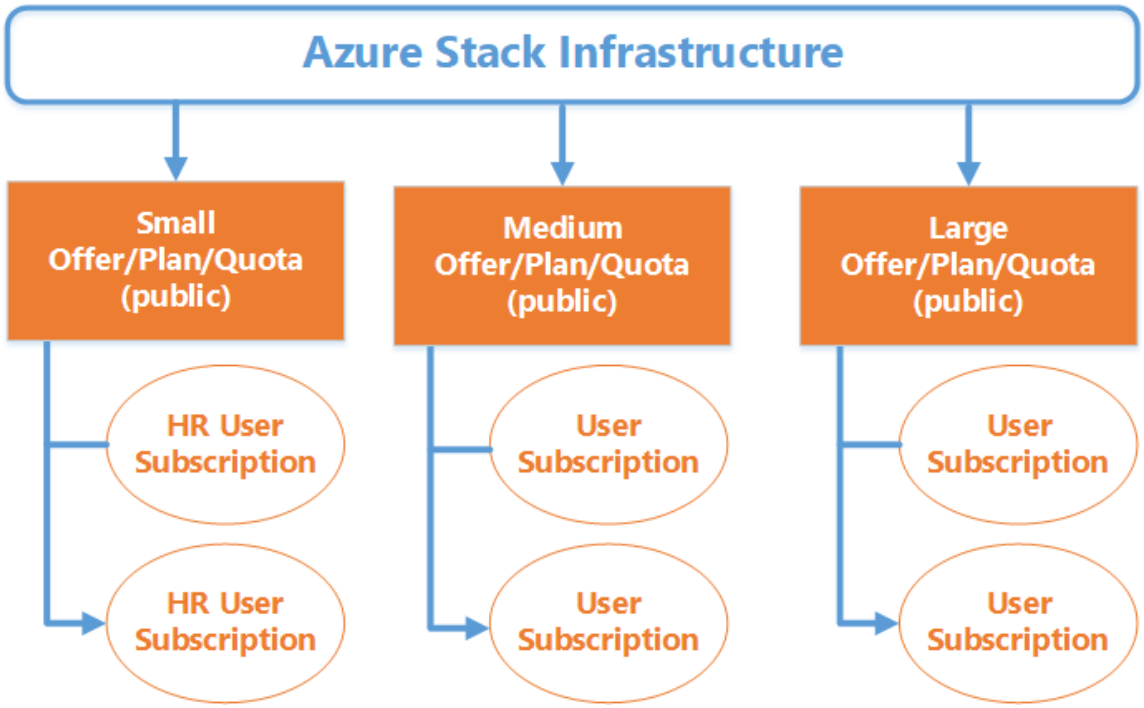


Figure 15: Example of a self-service subscription model

A high-level process for the self-service subscription model is as follows:

1. The Azure Stack cloud operator (or delegated provider administrator) creates offers, plans, and assigns quotas.
2. The Azure Stack cloud operator (or delegated provider administrator) makes offers public.
3. Users can authenticate and access the Azure Stack user portal and select Get a Subscription.
4. Users can browse the available offers and select one for their subscription.
5. Users can now deploy resources within the boundaries imposed by the quotas for that subscription.

Assigned subscription model

In this model, users are specifically assigned to subscriptions via assignment at the plan level. The Azure Stack operator does this in the administrative portal under User Subscriptions. Offers are not made public, and the operator (or delegated administrator, see Delegation) can choose to utilize custom URLs as well to ensure the boundaries imposed by these hidden subscriptions are not compromised. To onboard user subscriptions, the Azure Stack operator will send users a link to their specific URL, and after authentication, they will have access to the subscription and the offers, plans, and quotas that are configured for their subscription. You can see an example of the assigned subscription model in Figure 16.

While requiring slightly more management from IT, the assigned subscription model best fits service providers or enterprises where knowledge of other users is either a security issue or otherwise unwanted. Your organization may also choose this model to facilitate compliance, restrictions on co-administrators viewing other subscriptions, or special requirements for billing.

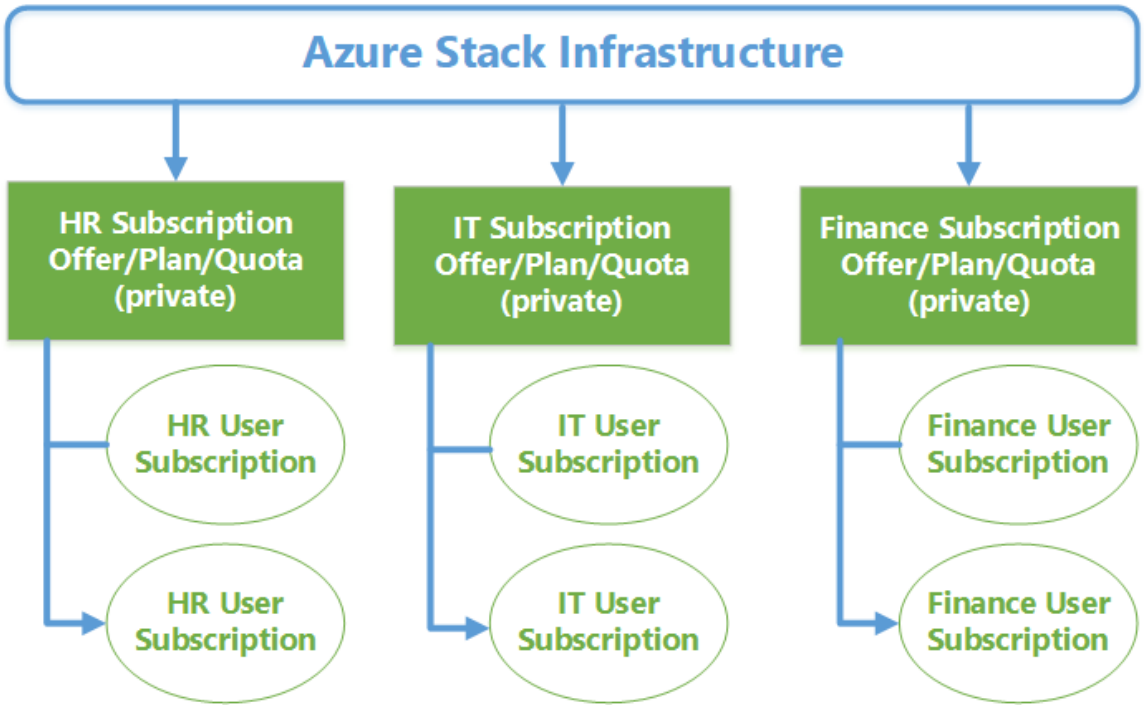


Figure 16: Example of an assigned subscription model

A high-level process for the self-service subscription model is as follows:

1. The Azure Stack cloud operator (or delegated provider administrator) creates offers, plans, and assigns quotas.
2. The Azure Stack cloud operator (or delegated provider administrator) creates department level subscriptions and aligns to appropriate offers/plans/quotas.
3. The Azure Stack cloud operator (or delegated provider administrator) assigns users to the subscription to which they require access.
4. Users access the Azure Stack user portal and assigned subscriptions are automatically available to them.
5. Users deploy resources within the boundaries imposed by the quotas for each assigned subscription.

Another scenario for this subscription model is for organizations requiring an additional level approval using an IT Service Management (ITSM) solution or process. In this scenario, some customers may require approvals before consuming Azure services within the organization. Those customers can easily achieve this by automation, with their existing ITSM portal listing offers through the APIs, possibly even syncing them in their Configuration Management Database (CMDB), and automation assigning the subscriptions after the required approval workflow.

Hybrid subscription model

Depending on circumstances, you may opt to use a hybrid subscription model that includes a combination of self-service and assigned subscriptions, as shown in Figure 17. For instance, your organization might have both experienced and novice users, each with different requirements. The experienced users need the ability easily acquire a subscription. But novices would have a trial offer that contains minimal quotas for resources, enabling users to get started quickly with Azure

services within your organization. If you want them to consume a specific subscription, you can either use publicly visible offers or create offers that are private, assigning them to specific subscriptions.

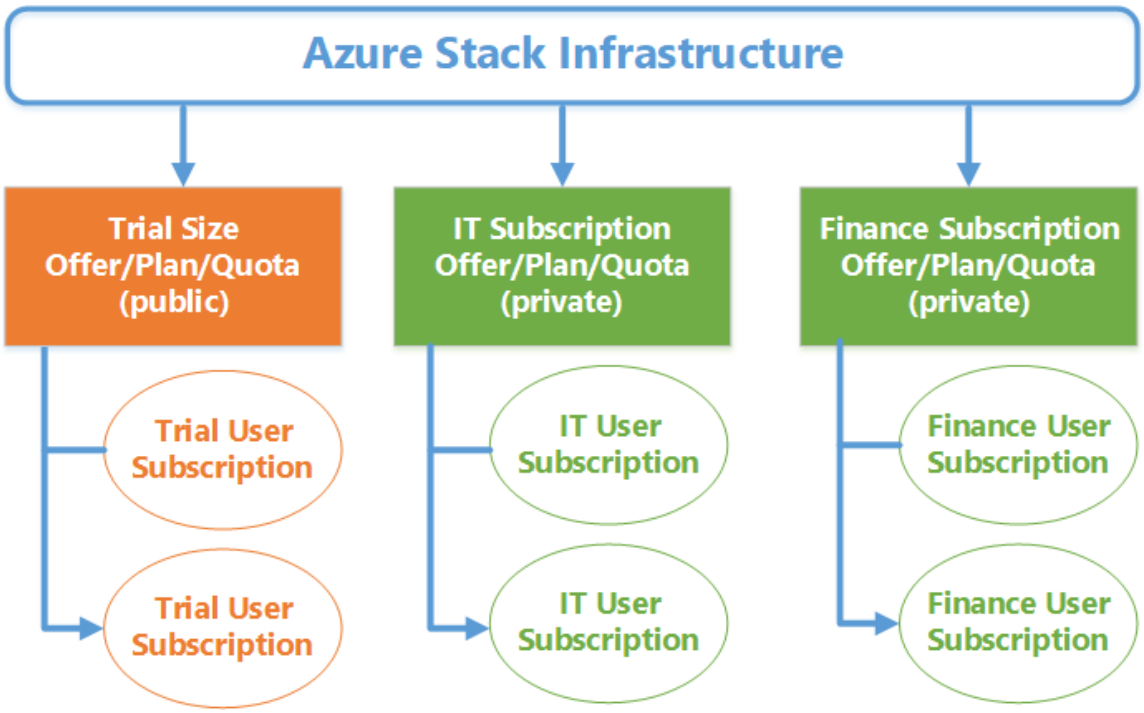


Figure 17: Example of a hybrid subscription model

A high-level process for the self-service subscription model is as follows:

1. The Azure Stack cloud operator (or delegated provider administrator) creates offers, plans and assigns quotas.
2. The Azure Stack cloud operator (or delegated provider administrator) creates a trial offer and configures it as public.
3. The Azure Stack cloud operator (or delegated provider administrator) creates department user subscriptions and aligns to appropriate offers/plans/quotas.
4. The Azure Stack cloud operator (or delegated provider administrator) assigns users to the required subscriptions.
5. Self-service users can authenticate and access the Azure Stack user portal and select Get a Subscription for a public offer, such as a trial offer.
6. Assigned users can authenticate and access the Azure Stack user portal and access their assigned subscriptions.
7. All users can now deploy resources within the boundaries imposed by the quotas for their subscription.

Naming subscriptions

When naming your subscriptions or any other objects within Azure Stack, you will want to consider using a consistent set of naming conventions to ease management and facilitate billing.

Considerations when naming subscriptions are provided below:

- **Company:** Service providers must ensure that the company name is included.
- **Department:** Enterprises may want to use department names as an identifier for the groups within the organization.
- **Product line:** Some enterprises need to include the name of a product or application.
- **Environment:** Some enterprises have multiple environments for the lifecycle management of applications, such as PROD, QA, or DEV, and want to identify them.
- **Characters:** Be aware that certain characters are not allowed, or will cause issues in naming conventions, like an ampersand.

Naming standards extend beyond subscriptions. There are many objects in Azure Stack that require proper naming so they can be easily identified, for example, storage accounts, virtual machines, availability sets, and virtual networks. When choosing your naming standard, keep in mind that you must work within several constraints:

- Storage names must be unique within an Azure Stack environment.
- Some resource names must be alphanumeric.
- Some resources are case sensitive.
- Some objects' names are constrained by length. You can determine this by reviewing the information provided in the portal, as shown in Figure 18.

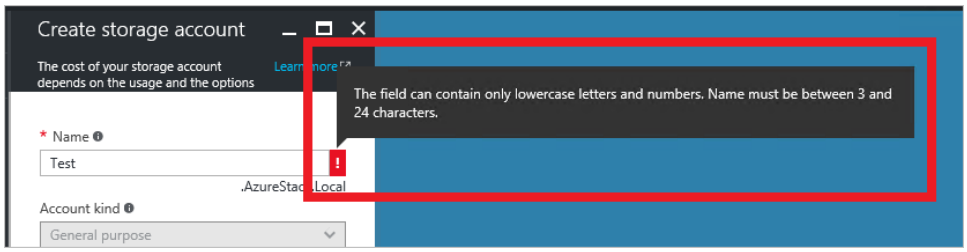


Figure 18. Constraints error in Azure Stack portal

You can read more about the best practices for Azure naming conventions here: https://docs.microsoft.com/en-us/azure/architecture/best-practices/naming-conventions

Adding additional subscription administrators

You can add additional administrators to a subscription using role-based access, which will allow others to assist you in managing offers and plans. Select the subscription you want to add a co-administrator to, and then click on the **Access** icon as shown in Figure 19.

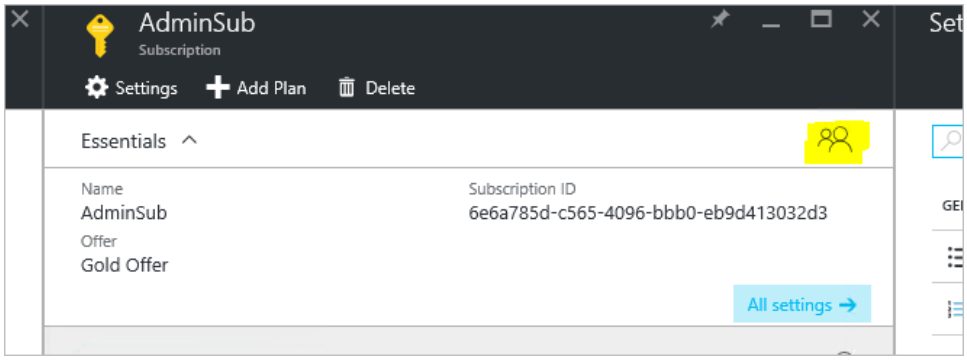


Figure 19: In the blade for the resource, click the **Access** icon

Clicking **Access** opens the Users blade where you can then add additional Azure AD users as Owners, Contributors, or Readers.

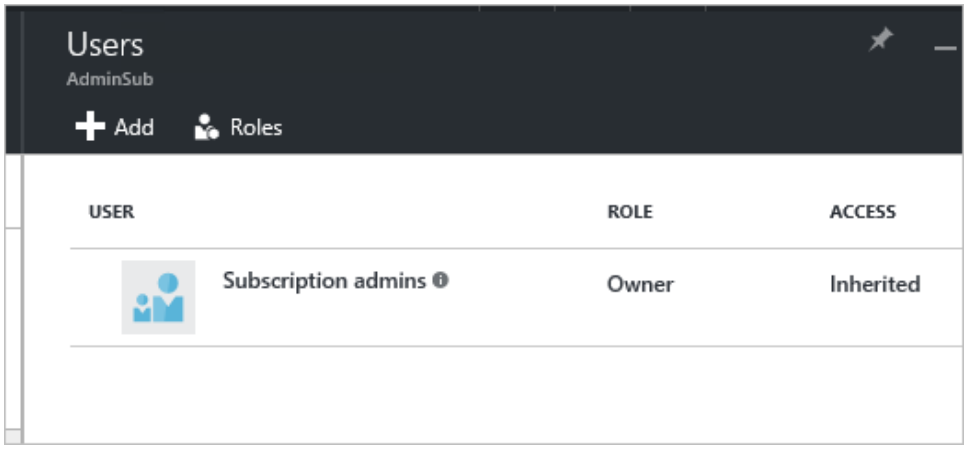


Figure 20: In the **Users** blade, add users and manage roles

Read the article Manage Role-Based Access Control for information on adding additional administrative users.

Azure Resource Manager limits

Like other Azure services, Resource Manager limits also apply to Azure Stack at the subscription level and are generally the same values as those found in Azure. The exception is any recent changes in Azure that have not yet propagated to Azure Stack.

These limits are applied per-region for each region accessible by your subscription. Service management limits are limits that are applied per-subscription, and the tables below provide examples of these limits.

Subscription level throttling

Resource	Default Limits
Resource Quota Limit	800
Resource Group Name Length Limit	90
Resource Group Quota Limit	800

Resource Move Limit	800
Deployment Name Length Limit	64
Deployment Quota Limit	800
Subscription Tag Quota Limit	900
Subscription Tag Name Quota Limit	100
Tenant Resources Query Subscription Count Limit	40
Tenant Resources Query Subscription Batch Size	10
Tag Key Limit	512
Tag Value Limit	256
Max Tags per Resource	15
Realtime Tag Count Threshold	600

Tenant level request throttling

Resource	Default Limits
Max Tenant Read Requests	15000
Max Tenant Write Requests	1200
Tenant Throttling Window Time	01:00:00
Tenant Throttling Bucket Size	00:05:00

Health check request throttling

Resource	Default Limits
Unauthenticated Throttling Max Requests	15000
Unauthenticated Throttling Window Time	01:00:00
Unauthenticated Throttling Bucket Size	00:05:00

Per subscription throttling

Resource	Default Limits
Max Subscription Bad Requests	15000
Max Subscription Write Requests	1200
Subscription Throttle Window Time	01:00:00
Subscription Throttle Bucket Size	00:05:00

Timers

Updates to subscriptions are made visible in the portal based on timers. If you create a new subscription, the change is immediate upon completion. However, to see the changes, you must refresh the portal.

General portal updates to subscriptions are hourly, so they will take place within 60 minutes after a given change.

Services

The key value of Azure Stack is the ability to run Azure services in your datacenter. This ability allows you to transform your on-premises datacenter resources and services into cloud services and cloud-native applications within your organization.

Understanding solutions and services

Azure is comprised of a series of discrete infrastructure and platform services deployed across global datacenters. Think of these as the discrete capabilities that, by themselves, provide a type of service such as storage, websites, or virtual machines. You can combine one or more of these services to support various *solutions*, which are tangible outcomes used to enable a range of aspects of a business. Using an infrastructure example, services such as Virtual Machines, Virtual Network, and Blob/Table Storage would enable traditional server-based application services like SQL Server or SharePoint Server. Alternatively, these services allow customers to run cloud-first applications as solutions using on-premises deployments of platform services Azure App Service (Web Apps) and Docker-integrated containers. That is a key difference between what most customers can achieve today with traditional server virtualization since Azure platform services allow your developers to describe, deploy, and control their applications and solutions using standard Azure APIs and using familiar Azure services that expand beyond traditional virtual machine-based solutions.

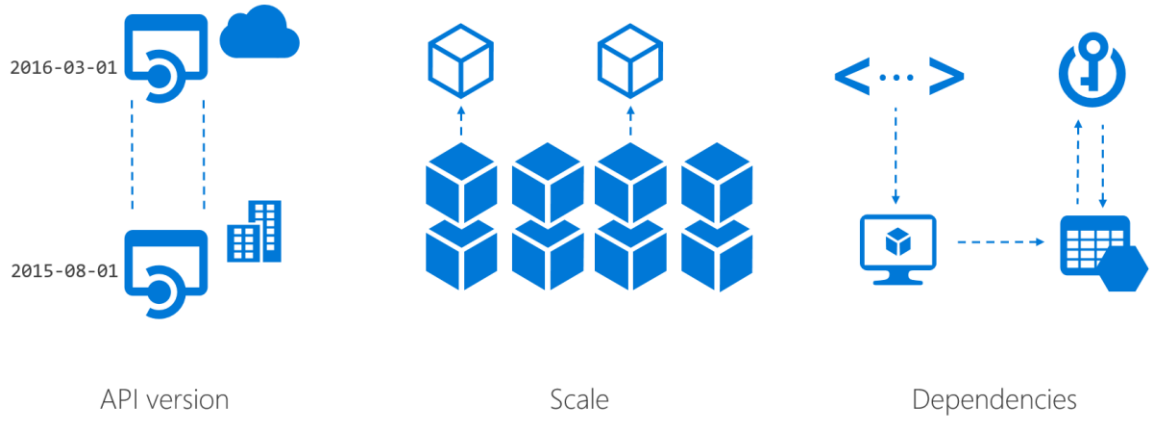


Figure 21: Example Azure services differences on Azure Stack differences

Azure Stack provides an Azure-consistent experience for developers and consumers of these services in your environments that go beyond the user interface. Having a consistent experience means that the Resource Manager API can be used to compose services to build your solutions

and host your organization's applications on Azure stack in a consistent way, using a consistent set of tools by your developers. However, Azure services on Azure Stack can sometimes contain some differences due to API version, scale, and dependencies, as illustrated in Figure 21.

- **API version:** The version of the API on Azure Stack may be different than the current version in Azure, specifically when using traditional infrastructure services as outlined above.
- **Scale:** In many cases, the scale of an Azure service is hardware dependent. In Azure datacenters, the smallest scale unit spans hundreds of servers while the smallest scale unit in Azure Stack for on-premises infrastructures is much smaller (under ten servers). This scale difference is critical to the planning process for any solution you wish to provide to your users.
- **Dependencies:** To enable familiar solutions and capabilities within Azure Stack, each service may have a dependency on another *resource provider*, a specified *agent version,* or dependent *hardware capabilities* to support an expected set of functionalities.

On an Azure Stack instance you can list the installed resource providers along with their API versions using the following PowerShell cmdlet:

```
Get-AzureRmResourceProvider | `
Select ProviderNamespace -Expand ResourceTypes | `
Select * -Expand ApiVersions | `
Select ProviderNamespace, ResourceTypeName, @{Name="ApiVersion";
Expression={$_}}
```

Read more about the considerations for Azure services on Azure Stack here:

- Key considerations: Using services or building apps for Azure Stack
- Considerations for Virtual Machines in Azure Stack
- Azure Stack Storage: Differences and considerations
- Considerations for Azure Stack networking

Envisioning your solution in Azure Stack

When building your organization's solutions on Azure Stack, it's important not to follow the same models you may have used with your existing virtualization platform. In these cases, most applications leverage only IaaS virtual machines for their solutions. In addition to traditional infrastructure services, Azure Stack brings the power of Azure platform-based services to your organization's datacenter. You should review both existing and new solutions with this in mind, and consider the possibilities when running these systems on Azure services. For example, if you have a traditional tiered web application running within a virtualized infrastructure using Windows Server web server (IIS) as the front end and a database running MySQL on the back end, you could review these solutions and determine if the features being used by the web server front end are compatible with the version of Azure App Services that exists in Azure Stack. The same could be said for the data in MySQL, which you could evaluate to see if the versions are compatible and will support porting this solution from a single virtual machine to a shared instance of MySQL Server. In doing this analysis, you may find that many of your existing applications or solutions, which currently require isolated virtual machines, could be transitioned to using native Azure cloud-based services. Not only would your organization find potential savings in the operating system and associated third-party management licenses, but you would also likely see operational

benefits, given the differences between management, configuration, and maintenance of individual virtual machines and Azure Stack.

NOTE: To offer PaaS services in your Azure Stack environment, you will need to add the following resources providers:

- **MySQL resource provider:** Create MySQL servers and databases through Resource Manager deployment templates and provide MySQL databases as a service. MySQL databases, which are common on websites, support many website platforms. Read the article Use MySQL databases on Microsoft Azure Stack for more information.
- **SQL resource provider:** An adapter to expose SQL databases as a service of Azure Stack. After you install the resource provider and connect it to a SQL Server instance, you and your users can create databases for cloud-native apps, SQL-based websites, and SQL-based workloads SQL without having to provision a virtual machine that hosts SQL Server each time. Read the article Use SQL databases on Microsoft Azure Stack for more information.
- **App Service resource provider:** Infrastructure virtual machines deployed to support hosting Web, Mobile, and API Apps. Read the article App Service on Azure Stack overview for more information.

Important points

The following core resource providers are visible in the portal in the Resource Providers blade. You will break core functionality in Azure Stack if you remove any of these resource providers.

Resource Provider	NameSpace
Compute	Microsoft.Compute.Admin
Health	Microsoft.InfrastructureInsights.Admin
Network	Microsoft.Network.Admin
Storage	Microsoft.Storage.Admin

Additional resource providers that are installed by default are visible under the **Resource Explorer** > **Resource Providers** blade. These are the base resource providers that enable the functionality you see in the portal after installation. Removing of any of these also breaks core functionality in Azure Stack.

Resource Provider	NameSpace
Authorization	Microsoft.Authorization
BackupAdmin	Microsoft.Backup.Admin
Commerce	Microsoft.Commerce
CommerceProviders	Microsoft.Commerce.Providers
Fabric	Microsoft.Fabric

FabricAdmin	Microsoft.Fabric.Admin
Gallery	Microsoft.Gallery
GalleryAdmin	Microsoft.Gallery.Admin
GalleryProviders	Microsoft.Gallery.Providers
InfrastructureInsightsProviders	Microsoft.Infrastructure.Insights.Providers
Insights	Microsoft.Insights
InsightsProviders	Microsoft.Insights.Providers
KeyVault	Microsoft.KeyVault
KeyVaultAdmin	Microsoft.KeyVault.Admin
Resources	Microsoft.Resources
ResourcesAdmin	Microsoft.Resources.Admin
Subscriptions	Microsoft.Subscriptions
SubscriptionsAdmin	Microsoft.Subscriptions.Admin
SubscriptionsProviders	Microsoft.Subscriptions.Providers
UpdateAdmin	Microsoft.Update.Admin

Tags, polices, and locks

Resource Manager provides the ability to apply additional management constructs to your resources to assist with billing, consistency, and prevent administrative error. Like Azure, Resource Manager resource tags, policy, and locks can be used in Azure Stack to mitigate these types of issues when planning and managing a large Azure Stack environment. As an administrator of an instance of Azure in your datacenter, you'll need to develop a strategy to implement these and other constructs in your subscriptions to provide a well-managed Azure Stack experience for your users.

Tags

Resource Manager provides the ability to use resource tags to logically categorize and assign meaningful metadata to each resource in your subscription. Resource tags can be assigned and viewed in the portal, are accessible programmatically, and are visible in your Azure bill. Since resource tags are metadata on a given resource, you can assign them to resources outside of a single deployment, allowing for ease of identification when reviewing your bill.

When creating tags, you must use a name/value pair. Some common examples of situations where you may want to use tags include:

- Grouping components of an application.
- Grouping of resources related to applications, cost centers, or user units for billing purposes.

- Organization of components for management purposes.

Careful planning is required when applying tags, as each resource or resource group can have a maximum of 15 tags. The tag name is limited to 512 characters, and the tag value is limited to 256 characters. Having consistency in your naming convention for resource tags is important for being able to reconcile these in your Azure billing statement.

You can apply tags through the portal, PowerShell or within Resource Manager JSON templates at deployment time. Applying tags ensures that any new resources that should be part of a logical grouping are automatically assigned the proper resource tags. Figure 22 shows the Tags blade in the Azure Stack portal.

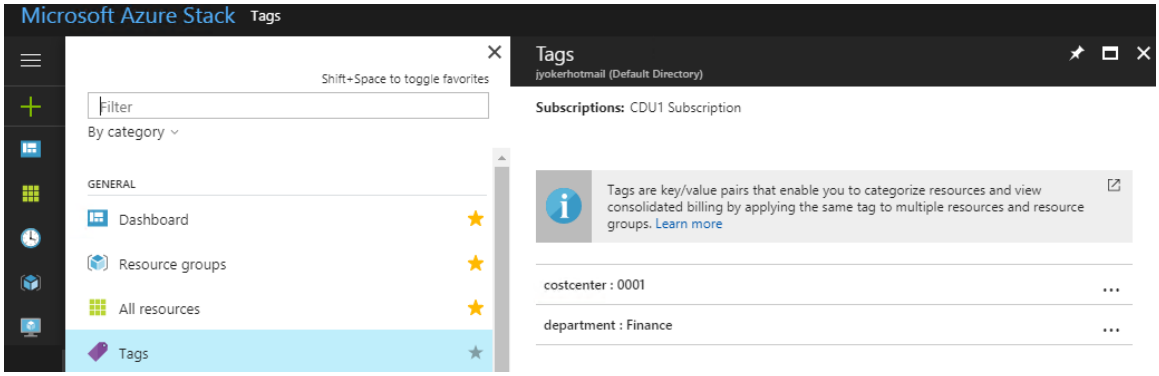


Figure 22: Tags in the Azure Stack portal

You can also add tags using PowerShell scripts, which is especially valuable if you have a lot of existing resources that require updating with a new tag. The command below will allow you to obtain a list of all tags in a subscription and the number of resources to which each tag is applied:

```
Get-AzureRmTag
```

To obtain a list of all tags applied to a resource, you can use this command:

```
(Get-AzureRmResource -ResourceName <Name of Resource> -ResourceGroupName
<Name of RG>).Tags | %{ $_.Name + ": " + $_.Value }
```

To obtain a list of all resources with a specific tag applied, you can use this command:

```
(Find-AzureRmResource -TagName <Name of Tag> -TagValue <Value of
Tag>).Name
```

For example, `(Find-AzureRmResource -TagName Dept -TagValue Development).Name`

Policies

Resource Manager uses policies to prevent users in your organization from breaking conventions that are needed to manage resources. Policy definitions can be used to describe the actions or resources that are specifically denied. You apply policy definitions at a specific scope, such as the subscription, resource group, or resource. Policy rules allow you to set explicit allow (default) or deny permissions to operations based on your defined criteria.

It is important to note that all policies are inherited by all child resources. So, if you apply a policy to a resource group, it affects all resources in that resource group.

You may want to use polices for:

- Application of required tags on resources for chargeback and billing purposes.
- Managing environments of resources.
- Applying naming conventions.

Policy definitions are created within the JSON templates you use to deploy a resource. You configure one or more conditions or logical operators, and these define the actions that are to be taken and the effect when a condition is satisfied.

A policy requires the following three components:

- **Parameters:** Values that are specified when the policy is assigned.
- **Condition/logical operators:** A set of conditions that you can manipulate through a set of logical operators.
- **Effect:** What happens when the condition is satisfied—either deny or audit.

The example below allows you to configure a deny policy based on the parameters of the naming prefix and suffix not being met.

```
{
  "if" : {
    "not" : {
      "field" : "name",
      "like" : "namePrefix*nameSuffix"
    }
  },
  "then" : {
    "effect" : "deny"
  }
}
```

Read the article Resource policy overview for more information on the configuration and use of policies. Also, AzureStack-Tools on GitHub provides a policy example for Azure Stack.

Locks

You can use locks to restrict operations on resources. By using locks, you can prevent deletion of, or tampering with, high-value assets. As a best practice, you should apply locks on resources that tend to be static, and those that have a high impact if they are deleted.

You can apply locks at the subscription, resource group, or resource levels through the portal, or by using any of the common tools for managing Azure Stack, such as PowerShell, CLI, or REST APIs. When applied, they allow two types of enforcement:

- **CanNotDelete:** Authorized users can only read and modify resources.
- **ReadOnly:** Authorized users can only read from the resource but may not modify or delete it.

Those applied to a parent will flow down to all child resources through inheritance, with the most restrictive lock being applied if there is more than one lock on a resource. To create or delete a lock you must have owner access to that resource.

Important points

- A maximum of 15 tags can be applied to any one resource or resource group.
- Policies work together with RBAC to deny or audit access to resources.
- Policies are inherited by all child resources.
- Resource locks should be used on static resources to prevent against accidental modification or deletion.

Pulling it all together

Now that we have reviewed the core capabilities and features of Azure Stack that are required to deliver services to your users, we'll look at design decisions and then steps to building an end-to-end validation environment for your installation.

Customer scenario

In our example, we will use an imaginary enterprise customer called Contoso. Like many enterprises, Contoso contains several business units which independently operate when consuming services from IT while still adhering to corporate IT standards. Figure 23 is a diagram of the high-level organizational model for Contoso. The company has four departments: Human Resources (HR), Finance, Information Technology (IT), and Development.

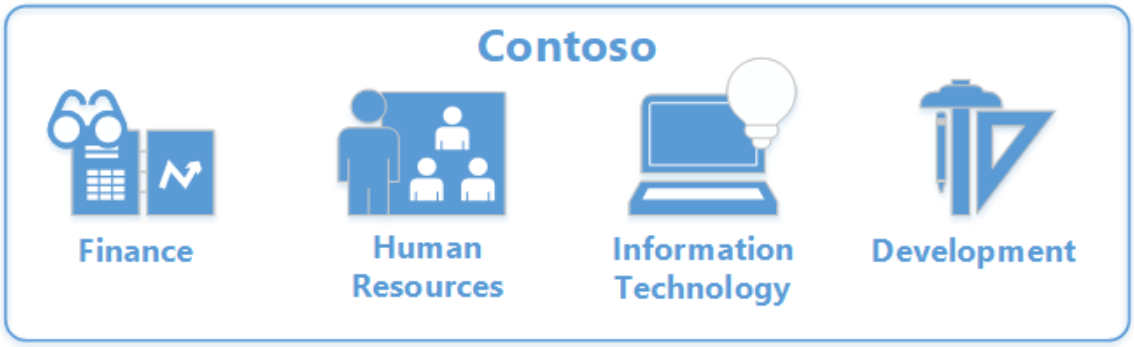


Figure 23. Contoso organizational design for this example

Contoso wants to deploy their first Azure Stack scale unit as a lab environment in a single datacenter to provide each business unit access to on-premises Azure services for their applications. Contoso envisions two primary users of Azure Stack, their IT department and their departments that align to the other business units within the organization. Contoso's departments typically require different services or quantity of resources on the Azure Stack based on their application's needs. At Contoso, HR and Finance both have specific applications that they require, and IT and Development use standard services as they build and deploy applications into the cloud. In addition to Azure marketplace images, Contoso's departments are required to use internally certified operating system images for the development of their applications. Operationally, Contoso has established naming conventions for resources they provision in their datacenter, so they wish to extend similar requirements to resources deployed in Azure Stack. Aside from the benefits consistent naming provides towards achieving standardization, Contoso wants to ensure that resources created in the environment are traceable by department,

operating system, and application. To help drive adoption of this new service, a trial offer should be made available to each department to support an early learning experience.

As we walk thru the design decisions in the following sections, we will refer to this business scenario and explore how different capabilities can be used to address Contoso's needs. Keep in mind that while these decisions are specific to the creation of a lab, the same considerations are required when designing your production deployments.

Design decision 1: Deployments

In our example, we will focus on the customer-owned deployment model, based on Contoso's primary business units. It is important that we understand the department's applications and services because they will impact our design decisions. As we move through the process of designing the solution, we also need to consider the plans and offers that will be required by each of these departments, as well as determine how regions will be utilized when this functionality becomes available. Given that Contoso has a single datacenter we'll focus on a single region deployment of Azure Stack.

An outline of the single-region deployment applications required by each group is listed below.

Domain	Business Units	Applications
local.azurestack.external	Human Resources	Multiple websites using a MySQL backend.
	Finance	A single commercial off-the-shelf finance application using a backend SQL Database.
	Information Technology	Core IT services, no specific applications.
	Development	Responsible for maintenance and upgrades to the HR websites and the Finance application. Investigating the inclusion of Key Vault capabilities to new applications.

To provide redundancy for administration, we will also add a second administrator to the Default Provider subscription.

Region	Azure Stack Operators	Co-administrators
Local	Admin1@"mylab".onmicrosoft.com	ITAdmin1

Design decision 2: Subscriptions and naming conventions

Contoso has determined the following high-level goals when designing their subscriptions:

- Design an enterprise strategy based on business units.
- Local administrators will manage users and services for their business units.
- Use a naming strategy that meets Contoso's tracking requirements.

From a subscription perspective, Contoso will use a hybrid subscription model to support the distinct needs of their departments. HR and Finance will use a self-service subscription model publicly visible offers. In contrast, the IT and Development departments will have delegation enabled, so the subscriptions will be created for each business unit, and a delegated administrator will manage each teams' user subscriptions.

Contoso has defined naming standards they require their users to follow and in the table below we have defined the naming for each department's subscriptions. Based on those goals, we will use the following naming convention values which will be used in each subscription:

- Azure Stack Cloud: CAS ("Contoso Azure Stack")
- Business Units:
    - Human Resources: HR
    - Finance: Fin
    - Information Technology: IT
    - Development: Dev

The following naming conventions will be used for each Azure Stack object type:

Object Type	Naming Convention	Example
Subscription	[cloud]-[business unit]-Subscription-[optional: sequential number]	CAS-IT-Subscription-01
Quota	[cloud]-[resource provider or service name]-[optional: service tier]	CAS-AppSvc-Premium
Plan	[cloud]-[business unit]-Plan-[optional: sequential number]	CAS-IT-Plan
Add-on Plan	[cloud]-[resource provider or service name]-AO	CAS-KeyVault-AO
Offer	[cloud]-[business unit]-Offer-[optional: sequential number]	CAS-IT-Offer
Tag	[authorized tag name]: [authorized value]	Dept : CAS-IT
Lock	[object/scope]-[type]-Lock	

Based on these goals and standards, the resulting design decisions are reflected in the following worksheet:

Business Units	Subscription Name	Subscription Administrator
Human Resources	CAS-HR-HRAdmin	HR Admin
	CAS-HR-HRuser1	

Finance	CAS-Fin-FinAdmin CAS-FIN-Finuser1	Fin Admin
Information Technology	CAS-IT-Subscription	IT Admin
Development	CAS-Dev-Subscription	Dev Admin

Design decision 3: Services

To support a wide range of Azure services to each of the business units, additional resource providers will need to be deployed. We will include the following resource providers in our solution to enable provisioning of both IaaS and PaaS services to user subscriptions.

Resource Provider	Default
Compute	Yes
Health	Yes
Network	Yes
Storage	Yes
Key Vault	Yes
MySQL	No
SQL	No
App Service	No

Design decision 4: Quotas, plans, and offers

For Contoso's business units to access services, we need to create offers and plans, and align quotas to each plan. When we design our offers, we must be aware of the services required by each business unit's applications.

First, let's review the needs of each of Contoso's business units. HR has some websites that use MySQL as a backend database tier. To meet their application needs, they require the AppService and MySQL resource providers, as well as compute, storage, and networking resource providers. These websites do not use a lot of resources, so we will set quotas at 50% of the defaults; this will support both current and future needs and can be increased as required. Finance, on the other hand, relies heavily on an off-the-shelf cloud-ready application that uses SQL server as its database. Like HR, their usage is not projected to be overly heavy, so we will set quotas at 50% of the defaults. To support the needs of both HR and Finance, along with the default IaaS services, we will be required to deploy the optional SQL/MySQL resource providers.

Next, we can look at the needs of Contoso's IT and Development division. Contoso's IT department manages all the operational tasks required for their datacenter environment, as well as consuming resources for deployment and quality testing of the company's applications. From a quota perspective, the IT department would be permitted to consume services using the default

limits of each resource provider while development is restricted to 75% of the default values. IT is also responsible for ensuring that any custom images the company requires are made available in the marketplace. The Development department develops and maintains all the applications that Contoso requires. They work closely with IT on the release of these applications across the company. Both departments require all the IaaS services, as well as the additional resource providers accessed by HR and Finance. Development is looking to enable Key Vault capabilities for business applications in the future and will require this resource provider for applications which are under development.

Base offers and plans

The following goals have been identified when designing Contoso's offers, plans, and quotas:

- Use an easy to follow naming standard for all offers, plans, and quotas.
- Create separate offers, plans, and quotas for each business unit.
- Delegate administration for non-IT users to team administrators.
- Configure quotas so they can be adjusted on a per-team basis.

Based on these goals, the resulting design decisions are reflected in the following worksheet:

Offers	Advertised /Assigned	Assigned Users	Plans	Base Services	Quota Name	Quota Limits
CAS-HR-Offer	Advertised	None	CAS-HR-Plan	Compute Network Storage *App Service* *My SQL*	CAS-Compute CAS-Network CAS-Storage *CAS-AppSvc-Premium* *CAS-MySQL-Premium*	50% of defaults
CAS-Finance-Offer	Advertised	None	CAS-FIN-Plan	Compute Network Storage *SQL*	CAS-Compute CAS-Network CAS-Storage *CAS-SQL-Premium*	50% of defaults
CAS-IT-Offer	Assigned	IT1 IT2 IT3	CAS-IT-Plan	Compute Network Storage	CAS-IT-Compute CAS-IT-Network CAS-IT-Storage	100% of Defaults
CAS-Dev-Offer	Assigned	Dev1 Dev2 Dev3	CAS-Dev-Plan	Compute Network Storage *App Service* *My SQL* *SQL* *Key Vault*	CAS-Dev-Compute CAS-Dev-Network CAS-Dev-Storage *CAS-AppSvc* *CAS-MySQL* *CAS-SQL* *CAS-KeyVault*	75% of Defaults
CAS-Trial-Offer	Advertised	None	CAS-Trial-Plan	Compute Network	CAS-Trial-Compute	2 virtual machines

		Storage	CAS-Trial-Network CAS-Trial-Storage	10GB Storage 4 x IP

Add-on plans

Add On Plans	Quotas	Quota Limits
SQL Plan	CAS-SQL-AO	CAS-SQL-Premium
App Service Plan	CAS-AppSrv-AO	CAS-AppSvc-Premium
MySQL Plan	CAS-MySQL-AO	CAS-MySQL-Premium
Key Vault Plan	CAS-KeyVault-AO	Unlimited

To provide their business units with the operating system images Contoso requires, we will download any available images using Azure Marketplace syndication and upload any custom images to Marketplace in Azure Stack.

Design decision 5: Delegation

Contoso has determined that for their Development subscriptions, they will assign an administrator to manage users and resources. They will also use a custom URL for their delegated site.

HR and Finance, however, will rely upon the IT team to provision offers to them and manage their users. Offers for both business units will be set to *public*, so they are visible to all the business units, and each will select the appropriate offer.

Goals:

- Design a naming standard that is easy to follow.
- Business unit administrators will manage users and delegated offers.

Business Units	Delegated Administrator	Custom URL
Development	Dev Admin	ContosoDev

Design decision 6: Resource tags and locks

To support standardization of billing and the ability to trace resources back to their owners, Contoso will implement resource tags. Contoso will apply resource tags on each resource to identify business unit to enable chargeback or billing for each group and to identify any virtual machine operating system being deployed.

Resource tags associated with operating systems will be performed during provisioning. However, tagging of resources within the business units will be based upon corporate policy with the expectation users will tag their resources. Should resources without tags be found, the IT

department can tag them using PowerShell.

Because there will be multiple administrators of the Azure Stack solution across the IT organization, we will use locks for all IT and Development offers to prevent accidental deletion.

Tags:

Tag Name	Tag Value	Tag Scope
Dept	CAS-Finance	All Finance resources
	CAS-HR	All HR resources
	CAS-IT	All IT resources
	CAS-Dev	All Development resources
Operating System	WS2016	All Windows Server 2016 virtual machine resources
	WS2012R2	All Windows Server 2012 R2 virtual machine resources
Application	Application name	Resource groups for each application

Locks:

Lock Name	Locked Resource	Lock Value
IT-Delete-Lock	CAS-IT-Offer	Delete
Dev-Delete-Lock	CAS-Dev-Offer	Delete

Steps to building the solution

In the preceding sections, we created several tables of information based on design decisions that represent the high-level design for this lab. We will use that information to build a solution that delivers an enterprise Azure Stack environment following these steps.

1. Deploy the Azure Stack Development Kit.
2. Install PowerShell for Azure Stack.
3. Register Azure Stack against your Azure subscription.
4. Configure additional Azure Stack operators.

5. Download Marketplace items.
6. Deploy and configure the App Service resource provider.
7. Deploy the SQL and MySQL resource providers.
8. Create services quotas, plans and offers.
9. Import the required Azure Stack modules.
10. Import utilities and templates.
11. Create required delegated offers.
12. Configure quotas.
13. Add Marketplace items.
14. Configure resource tags.
15. Configure resource locks.

Completed design

Now that we have reviewed our design decisions based on our organization's requirements, we can create the initial Azure Stack logical design. The diagram in Figure 24 shows an example of a finished solution that can be implemented as a validation environment with the Azure Stack Development Kit and in the production scale-unit deployment of Azure Stack.

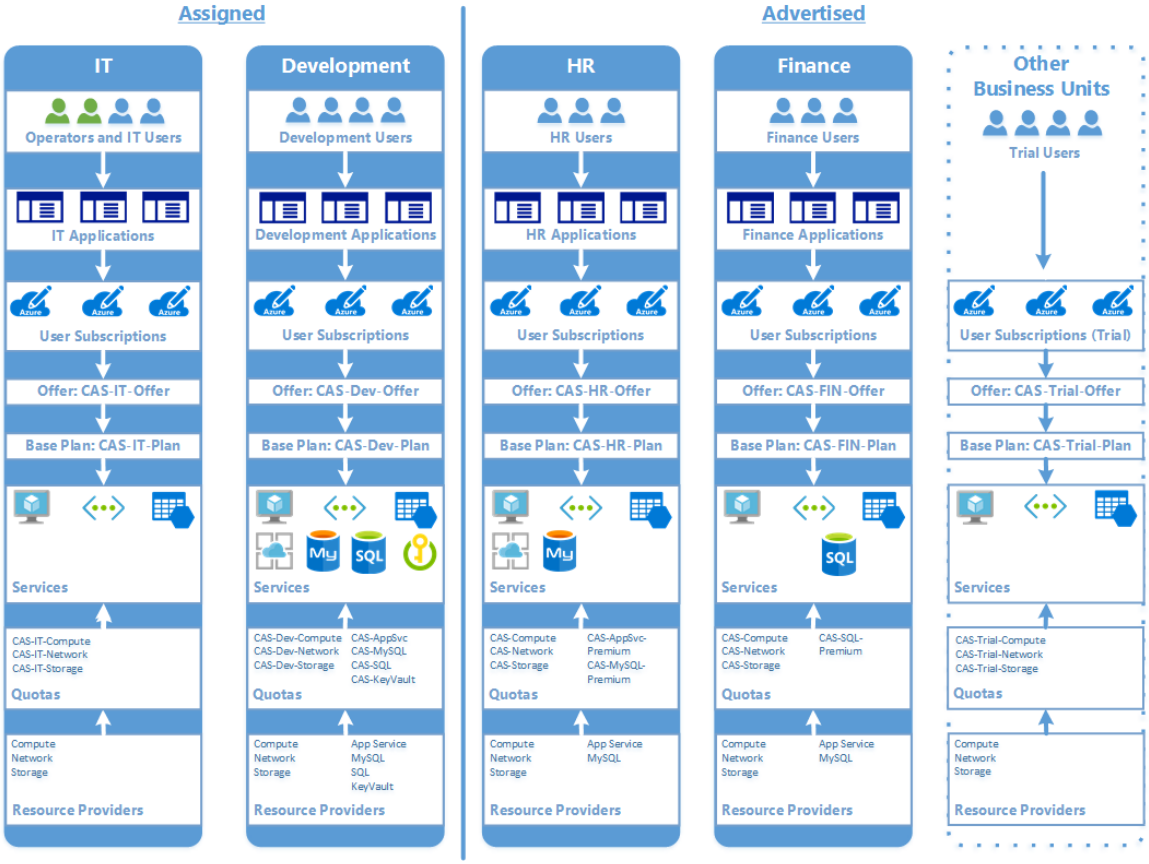


Figure 24: Envisioned Contoso Azure Stack solution

Conclusion

Azure Stack extends the power of Azure to your organization's datacenter, enabling development of hybrid solutions using Azure Services, and offering a range of new possibilities for your users' applications and services. Like most IT solutions, you must carefully plan when implementing an Azure Stack deployment, so your users can get started developing applications quickly and easily.

In this article, we have discussed some of the key decisions you encounter around the topics of Azure subscriptions, quotas, plans, offers, and administrative role design. Next, we use these decisions to model an environment with the Azure Stack Development Kit, enabling users and developers to gain hands-on experience prior to the deployment of your first Azure Stack scale unit.

We hope this has provided you with the necessary information to help your organization get the most out of Azure Stack.

Learn more

For more information, see the following resources:

- Channel 9: The Azure Stack Channel - https://aka.ms/ASWP_Ch9Blog
- Develop for Azure Stack - https://aka.ms/ASWP_Dev
- Create an offer in Azure Stack - https://aka.ms/ASWP_Offer
- Create a plan in Azure Stack - https://aka.ms/ASWP_Plan
- Create an offer in Azure Stack - https://aka.ms/ASWP_Offer
- Deploy Linux virtual machines on Azure Stack - https://aka.ms/ASWP_Linux
- Usage and billing in Azure Stack - https://aka.ms/ASWP_Billing
- Introduction to Key Vault in Azure Stack - https://aka.ms/ASWP_KVIntro
- Install PowerShell for Azure Stack - https://aka.ms/ASWP_PS
- Azure Stack Tools GitHub - https://aka.ms/ASWP_Tools
- Deploy templates in Azure Stack using Visual Studio - https://aka.ms/ASWP_VS
- Deploy templates in Azure Stack using the command line - https://aka.ms/ASWP_DeployCLI
- Deploy templates in Azure Stack using PowerShell - https://aka.ms/ASWP_DeployPS