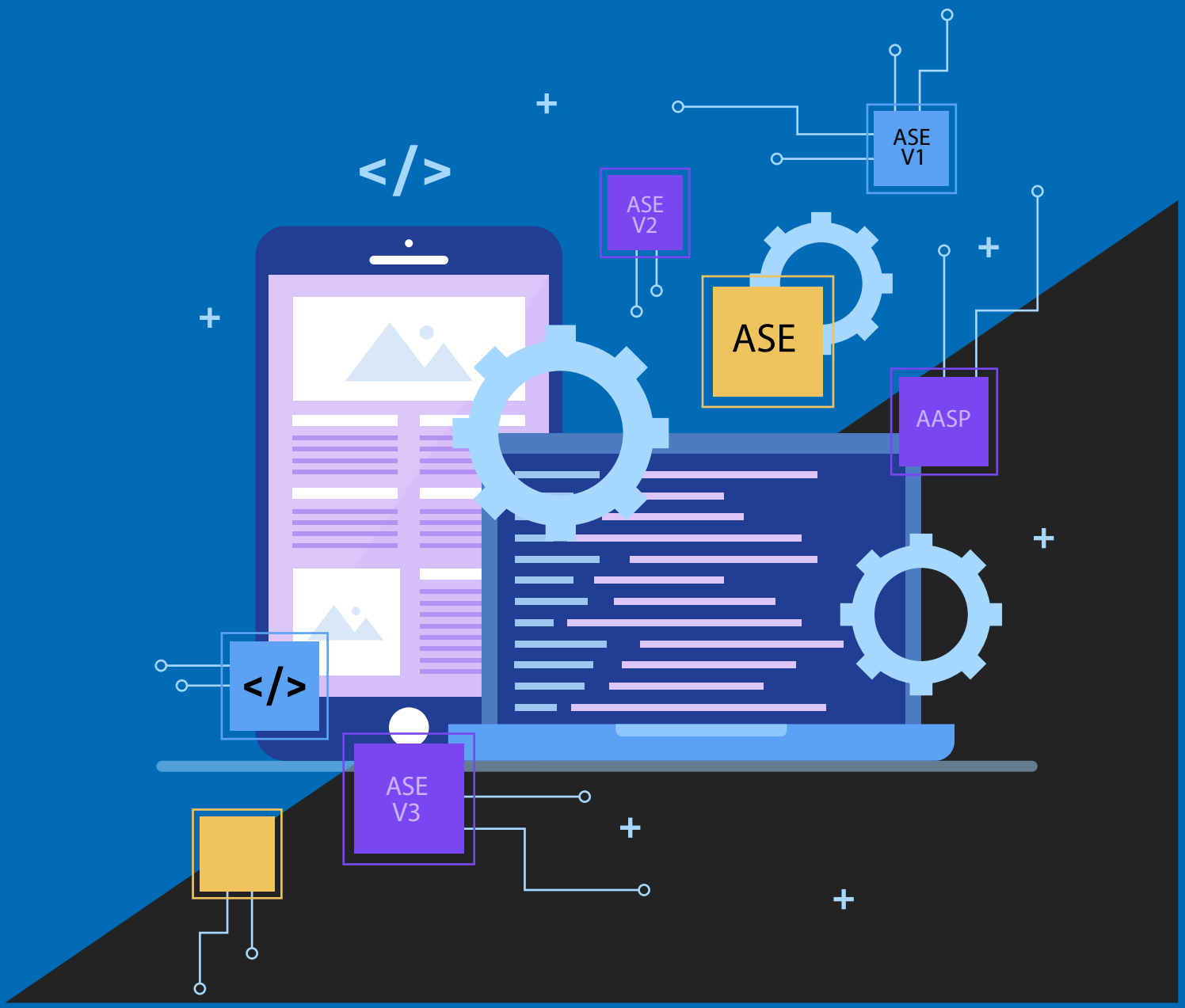
Digital & Analytics
by HCL Technologies

HCL

Azure App Service Environment

An Overview



Introduction

Azure App Service Environment (ASE) is an Azure PaaS service that provides a good alternative to customers who are already leveraging Azure App Service Plan, and also want additional security for surfacing or directly connecting to business-critical data in on-premise environments. This service is also an alternative for customers contemplating migration of sensitive line-of-business or intranet-based applications to Azure, which are expensive and risk-ridden programs. ASE allows customers to host business-critical applications on Azure service while adhering to stringent security and compliance requirements

What is Azure App Service Environment

Azure provides different options for hosting web applications, APIs, functions and container-based apps on the cloud. First option that is most widely used is Azure App Service Plan (AASP). Azure App Service Plan is multi-tenant environment, where the environment is shared by multiple clients and their apps. There are various pricing plans for Azure App Service ike Standard, Premium etc., which provide infrastructure based on the plan selected. In multi-tenant Azure App Service Plan, the end points of the hosted application are exposed to public internet. Though there are ways to secure applications hosted in Azure App Service Plan, they are still exposed to a certain degree, to vulnerabilities associated with public internet.

A more secured option for hosting applications is Azure App Service Environment, where there is greater focus on isolation of the hosting infrastructure from the external world and a more powerful infrastructure than Azure App Service Plan. ASE can be considered as Azure App Service Plan injected in client's virtual network in Azure. In ASE, network traffic can be controlled with network security group via direct access from internet. Applications hosted in ASE can be completely locked down from the external world. Client's virtual network in Azure can be connected with on-premise network with site-to-site VPN connectivity or Express Route, which allows the application hosted in ASE to be integrated with on-premise applications or services or databases. Also, the application hosted in ASE can be treated as an intranet application which can be accessed by on-premise users. ASE can host Windows web apps, Linux web apps, Docker container-based apps, and Azure Functions. A table on the differences between AASP and ASE is given below.

Comparison between App Service Plan and App Service Environment

App Service Plan (AASP)

- It is an environment (i.e. compute and storage resources) shared between applications or app services (Web apps, API apps, mobile apps or Azure functions) hosted by the customer
- Incoming traffic from internet is allowed.
- In App Service Plan, users cannot select virtual network or subnet where application will be hosted.
- App Service Plan is more suitable for public facing applications where there is no requirement for complete isolation of the hosting infrastructure.

App Service Environment (ASE)

- It is a secured environment where multiple app services or App Service Plans for a single customer are hosted. It wraps these services or plans in a VNet and provides load balancers for each ASE.
- Incoming traffic from internet can be totally closed by setting up ILB ASE.
- In ASE, applications are hosted in customer's virtual network and subnet
- ASE can be setup with different models like External ASE, ILB ASE and with WAF (Web Application Firewall) which enables to host both external facing and internal applications.

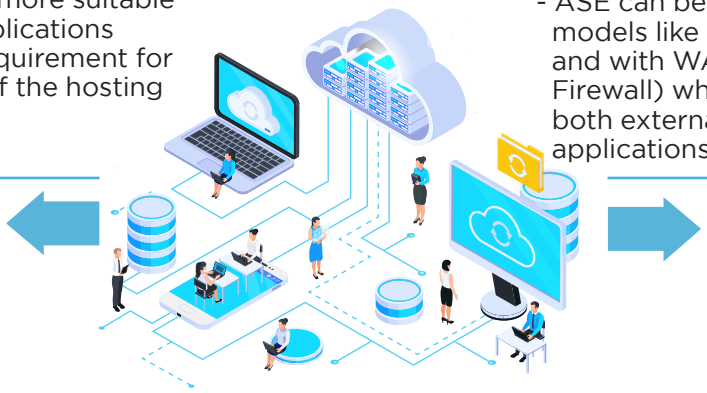
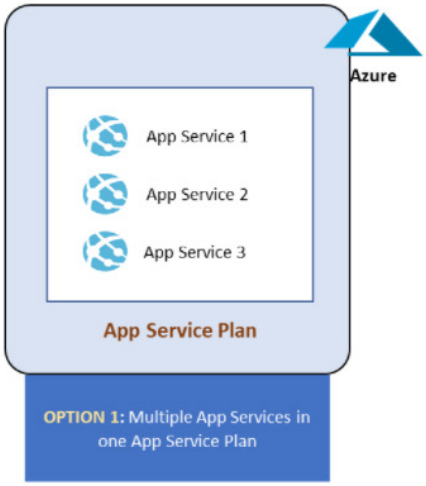
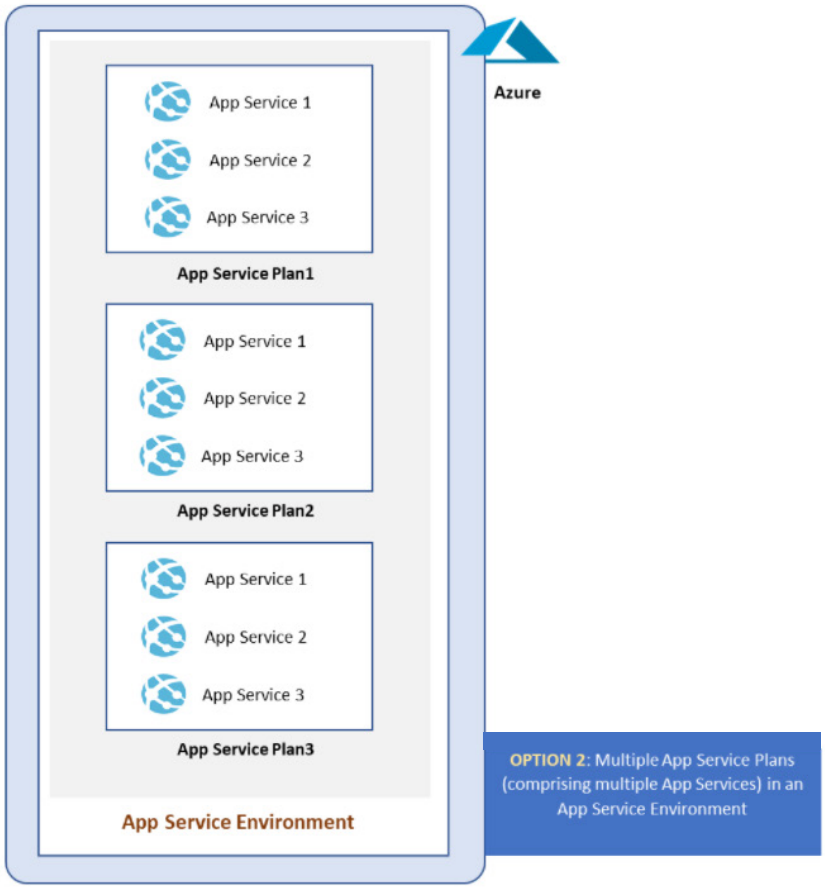


Figure 1 :

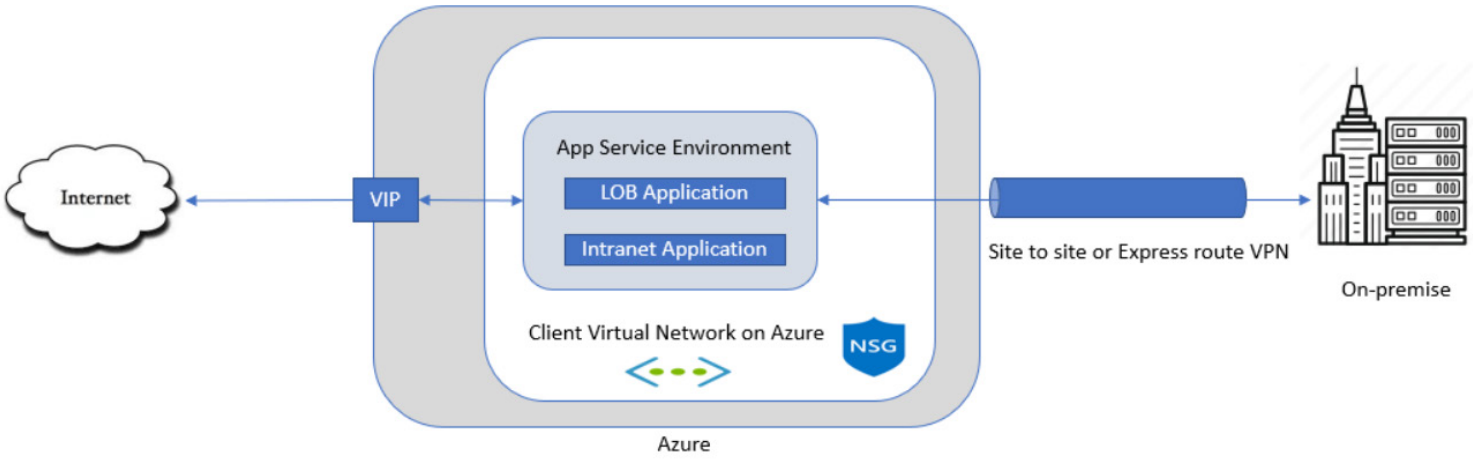
Shows a logical depiction of the main difference between an App Service Plan and an App Service Environment:





With the logical distinction established above, let us move on to the next level details of the App Service Environment. The rest of this whitepaper will explore different aspects of App Service Environment. Figure 2 below depicts how traffic from the internet (of the left) is routed through the Azure App Service Environment, hosted on a client virtual network on Azure to leverage services and data hosted on on-premise data centers.

Figure 2: Azure App Service Environment on client virtual network



Important features of ASE:

- An ASE is composed of front ends and workers. Front ends are responsible for HTTP/HTTPS termination and automatic load balancing of app requests within an ASE.

- Front ends are automatically added as the App Service Plans in the ASE are scaled out.

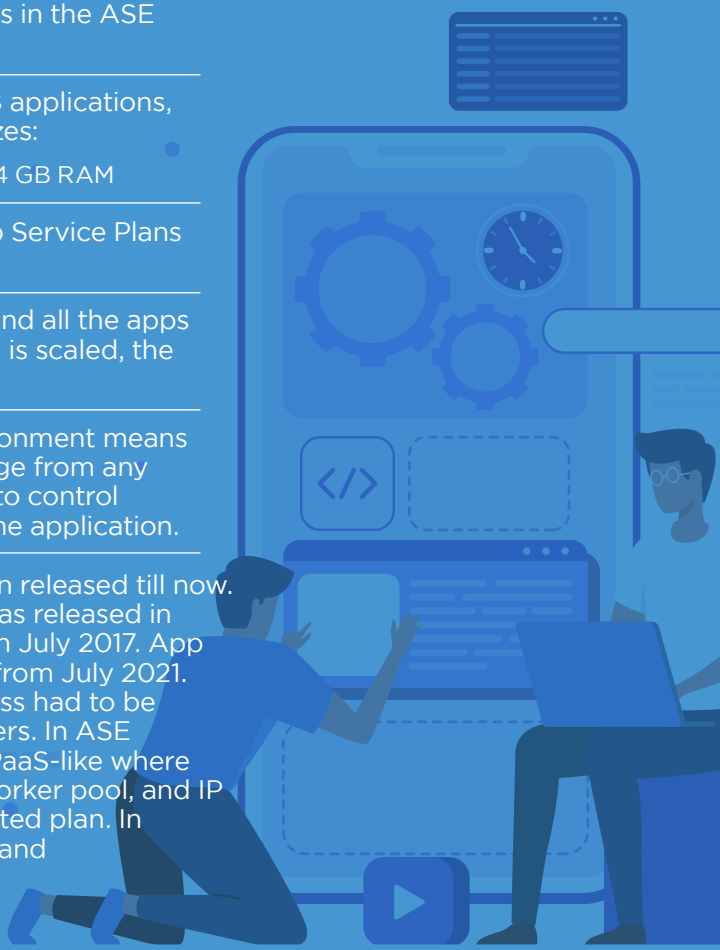
- Workers are roles that host customer applications (e.g., LOB applications, intranet applications). Workers are available in three fixed sizes:
  • One vCPU/3.5 GB RAM • Two vCPU/7 GB RAM • Four vCPU/14 GB RAM

- App Service Environments hold App Service plans, and App Service Plans hold apps.

- When an app is scaled, the App Service Plan is also scaled and all the apps in that same plan are also scaled. When an App Service Plan is scaled, the needed infrastructure is added automatically.

- Security: The fact that application is hosted in isolated environment means that application will not be impacted by any activity or outage from any applications of other customers. Rules can be setup in NSG to control network traffic which ensures no unwanted traffic reaches the application.

- Three versions of App Service Environment (ASE) have been released till now. The first version of the App Service Environment (ASE v1) was released in late 2015. App Service Environment (ASE) v2 was released in July 2017. App Service Environment (ASE) v3 has been generally available from July 2021. In ASE v1, resources like front end, worker pool and IP address had to be manually managed. That caused some confusion to customers. In ASE v2 this issue was addressed, and the service became more PaaS-like where a customer does not need to manually manage front end, worker pool, and IP address. The customer only needs to select the type of Isolated plan. In ASE v3 there has been further improvement in pricing front and underlying infrastructure.



A Case Study:

Many organizations are using ASE for greater security and isolation. One such case study is of Nobel Prize web site where Linux platform over ASE is used. Nobel Media used custom built Linux container to host their website in Azure. The website is built with WordPress content management system with PHP scripting language and MySQL backend. As the application is hosted inside their own virtual network in Azure, it provides better control over network access. As all the infrastructure is managed by Azure, it allows Nobel Media to focus on their business areas, like content creation. During the time of Nobel Prize announcement, the traffic in the site increases many folds. Azure ASE has provided the required scaling to handle large traffic spikes. Apart from this, by hosting their website in Azure, they are able to leverage other Azure services like AI to improve their content.

Different deployment models for ASE

There are 2 ASE deployment models:

External ASE model:

This allows application hosted in ASE to be exposed on an internet-facing IP.

Internal Load Balancer ASE model:

In this model, application hosted in ASE can be either be accessed from within the virtual network or network connected to the virtual network. The internal endpoint is an internal load balancer (ILB).

External ASE:

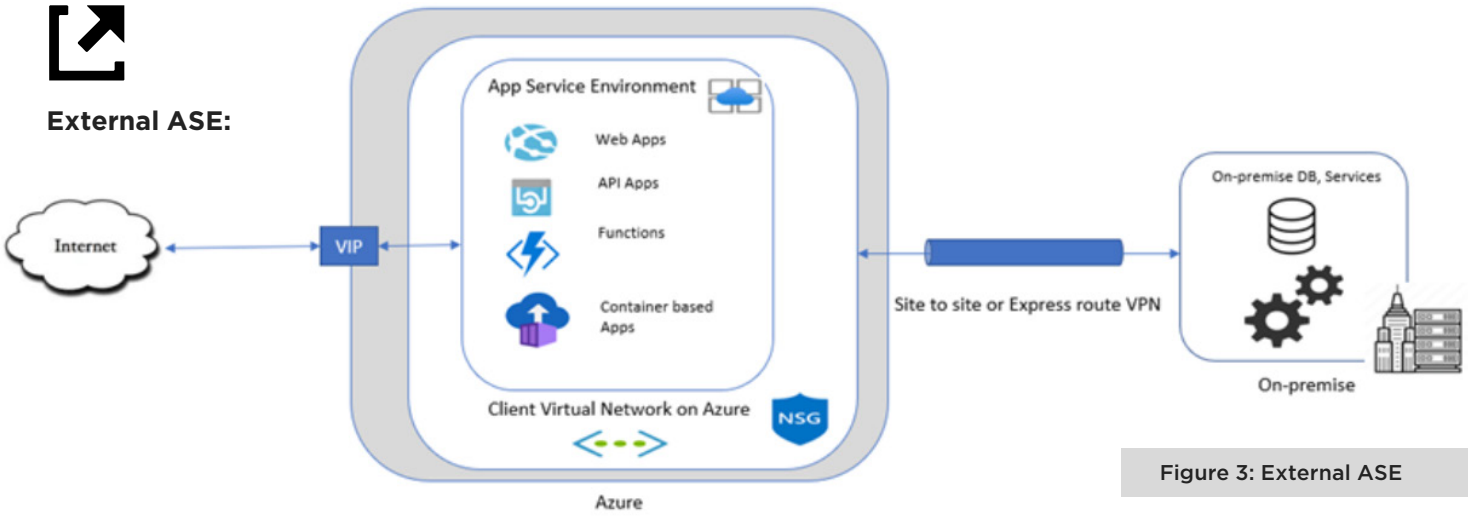


Figure 3: External ASE

In External ASE (refer to figure 3 above), hosted apps are exposed on an internet-accessible IP address. External ASE have virtual IP on an external public facing IP address. In External ASE, apps are registered with Azure DNS. There are no additional steps required for the apps to be publicly available. Applications hosted in ASE can be published similar to app hosted in multi-tenant app service though web deployment, FTP, CI, and from IDE like Visual Studio. For External ASE, dedicated IP address can be allocated to hosted app. In External ASE, IP-based TLS/SSL binding can be configured for hosted app in the same way as in the multi-tenant app service.

Internal Load Balancer ASE model:

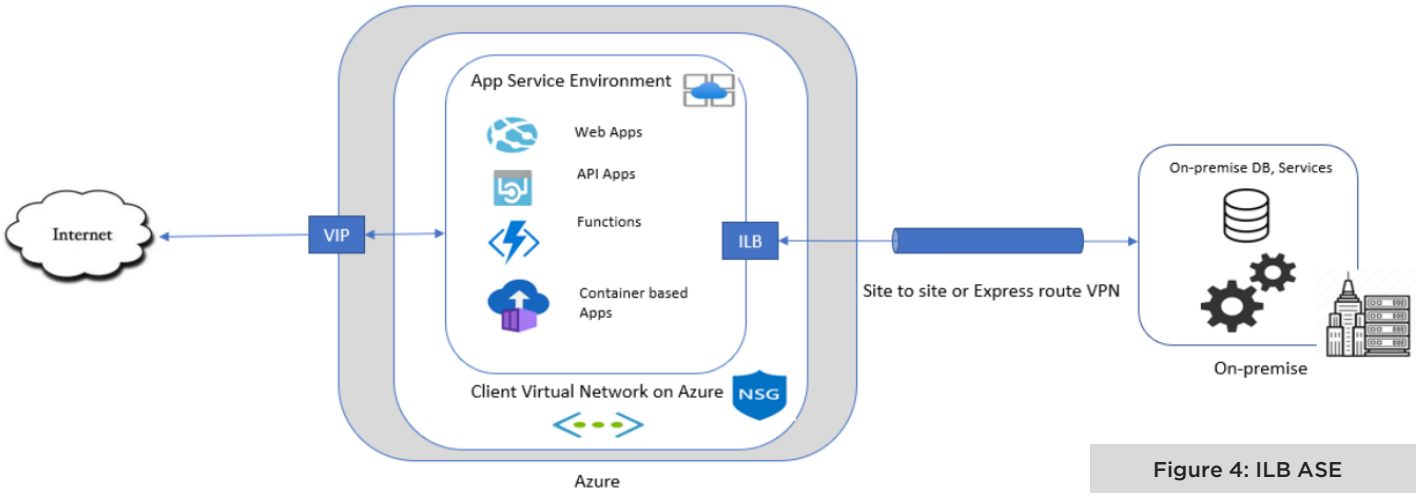


Figure 4: ILB ASE

This model (refer to figure 4 above) of deployment consists of Internal Load Balancer (ILB) which acts as endpoint for communication with the hosted apps. For ILB ASE, the address of the ILB address is the endpoint for HTTP/S, FTP/S, web deployment, and remote debugging. Applications hosted in ILB ASE can be protected with a WAF device. With an ILB ASE, DNS entries need to be maintained in clients own DNS server or with Azure DNS private zones. In ILB ASE, the SCM (Kudu) site isn't accessible from outside the VNet. With an ILB ASE, the publishing/deployment endpoints are only available through the ILB.Applications can be published to an ILB ASE from Azure DevOps by installing a self-hosted release agent in the virtual network that contains the ILB ASE. For SCM also, DNS endpoints need to be defined.

Configure an ILB ASE with a WAF device

Web Application Firewall (WAF) provide protection against vulnerabilities like DDos attack, SQL injections, cross site scripting (XSS) etc., to web applications. Refer to figure 5 below. A web application firewall (WAF) device can be configured with ILB ASE to expose selected apps to the internet and keep the rest only accessible from the VNet. This enables clients to build secure multi-tier applications where web-based, front-end layer is exposed to the internet and backend services remain secured in client virtual network. Web application firewall can be configured with Barracuda WAF, which is available in Azure Marketplace or with Azure Application Gateway WAF.

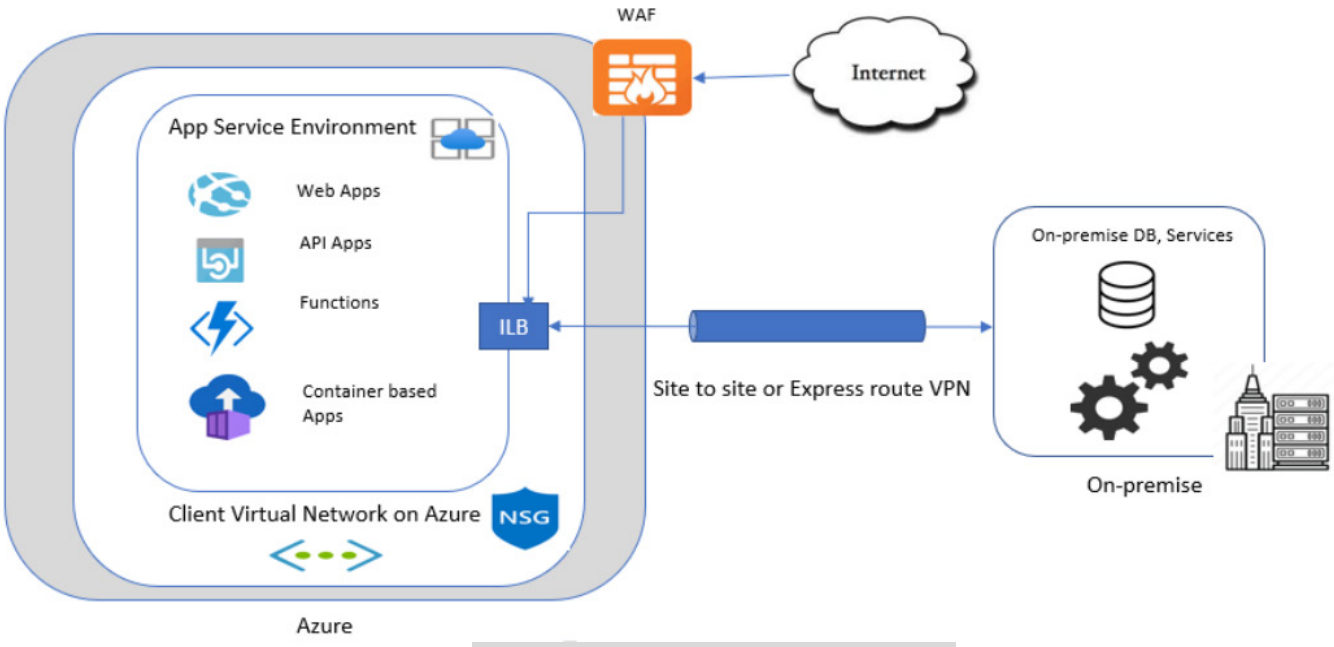


Figure 5: ILB ASE with a WAF Device

GEO-distributed scale

Application which are accessed in different geographic regions need to be setup in those geographies to reduce latency. The end users can access such applications with generic URL, and load balancer like Azure Traffic manager can point the user request to the application instance closer to end user.

Applications which have opted for ASE and have higher requests per second can scale by setting up a number of ASEs in different regions. Refer to figure 6 below. Azure Traffic Manager can be used as load balancer which can redirect the user to ASE closer to his location.

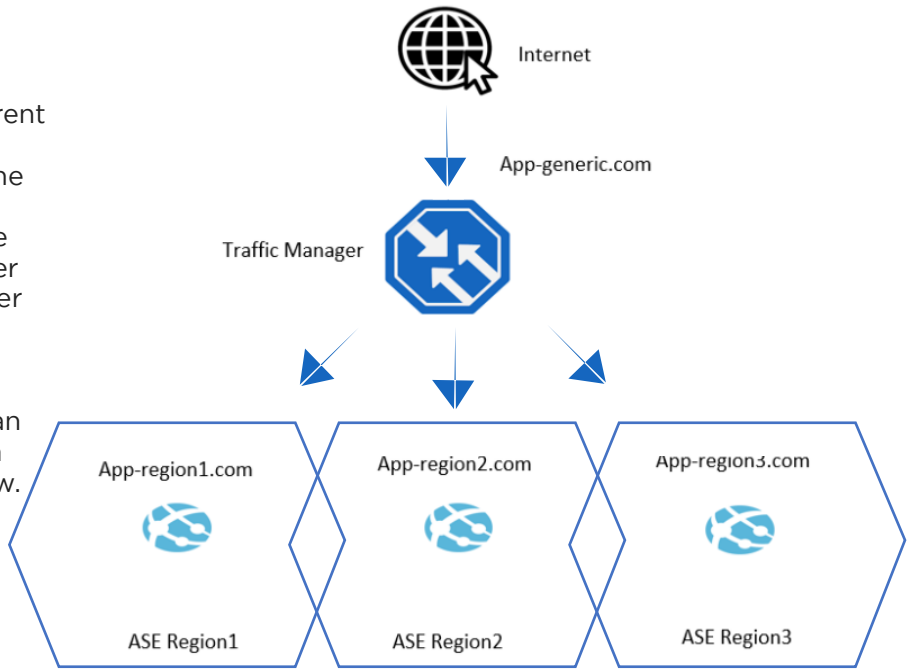


Figure 6: instances spread across geos load-balanced by azure traffic manager

Advantages of ASE

- ASE can host 100 App Service Plan instances.

- Among the benefits of ASE is a static IP address that can be used for both the inbound and outbound IP address for the apps. The IP address is dedicated for the client.

- The addition of ILB support meant that customers could now host intranet sites in the cloud. Clients could take an LOB application that they didn't want to be Internet-accessible and deploy it into ILB-enabled ASE. The ILB sits on one of the VNet IP addresses, so it's accessible only from within the VNet or from hosts that have access to the VNet over a VPN.

- The ILB-enabled ASE can be deployed with Web Application Firewall (WAF)-fronted applications.

- For WAF-fronted ASE applications, a customer could use a WAF virtual device to act as the internet endpoint for its ILB ASE-hosted apps, which adds an additional security layer for internet-accessible apps.

- In a two-tier application, the web-accessible app could be hosted in either the multi-tenant app service or from another ASE, and the back-end-secured API apps could then be hosted in the ILB ASE.

- Application can be scaled geographically by hosting instances in geo-specific ASEs and then load balancing them with traffic manager. End user will access the application with generic URL and traffic manager will redirect them to region-specific endpoint.

Setting up ASE

ASE can be setup in the following three ways:

1 **From Azure Portal,** when creating App Service plan, select Isolated price plan. This way, ASE and App Service plan within it is created in one go. Here details like runtime stack, OS, region, ASE name, Virtual IP type (internal or external) etc., need to be selected/filled up.

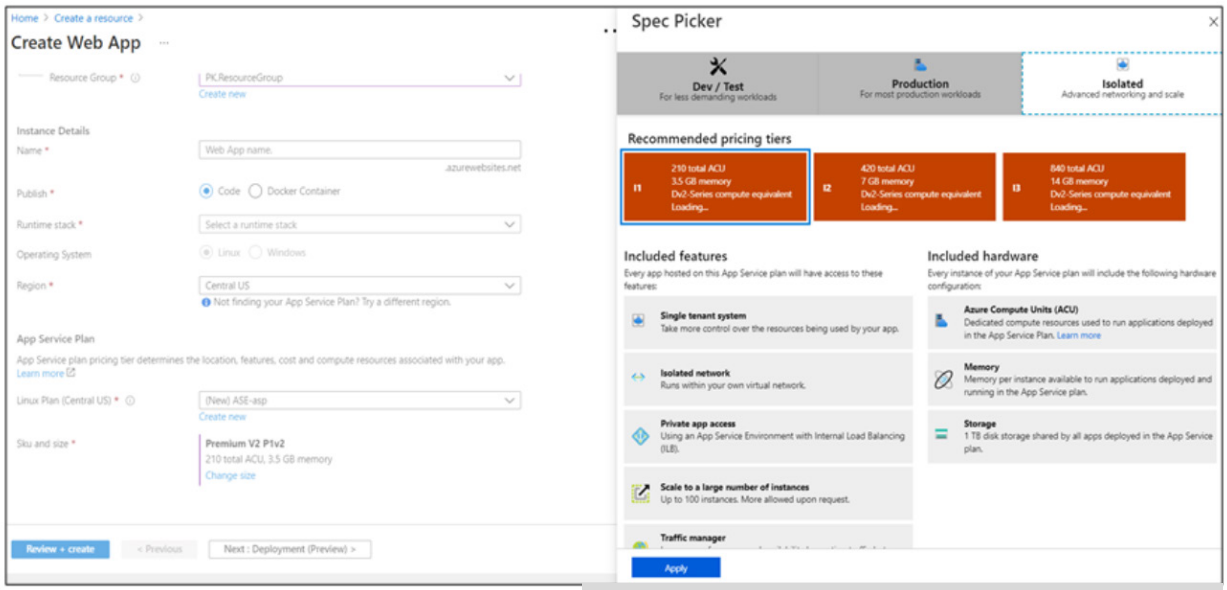


Figure 7: Select Isolated Plan for creating ASE

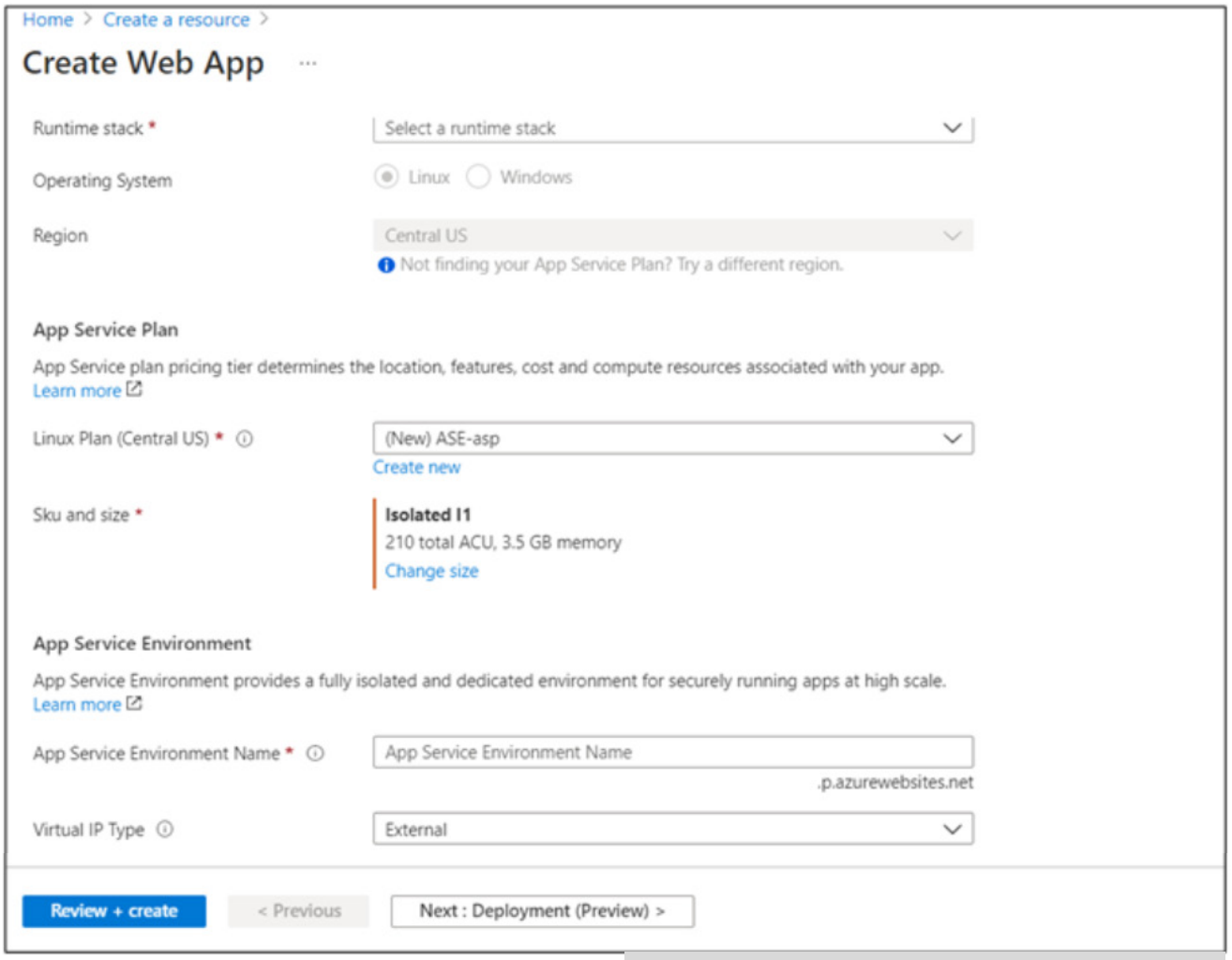


Figure 8: Select options like VIP type, OS etc.

2 **Standalone ASE:** Search for App Service Environment in Azure Market place and create ASE. This approach is for creating ILB ASE. While creating ASE, details like subscription name, resource group, OS type, name of ASE, Virtual IP (Internal for ILB ASE or External for Eexternal ASE), virtual network name and subnet etc., need to be selected/filled up. After ASE is created, apps can be created in it by using the normal process. ASE need to be selected as the location for the new apps to be hosted in ASE.

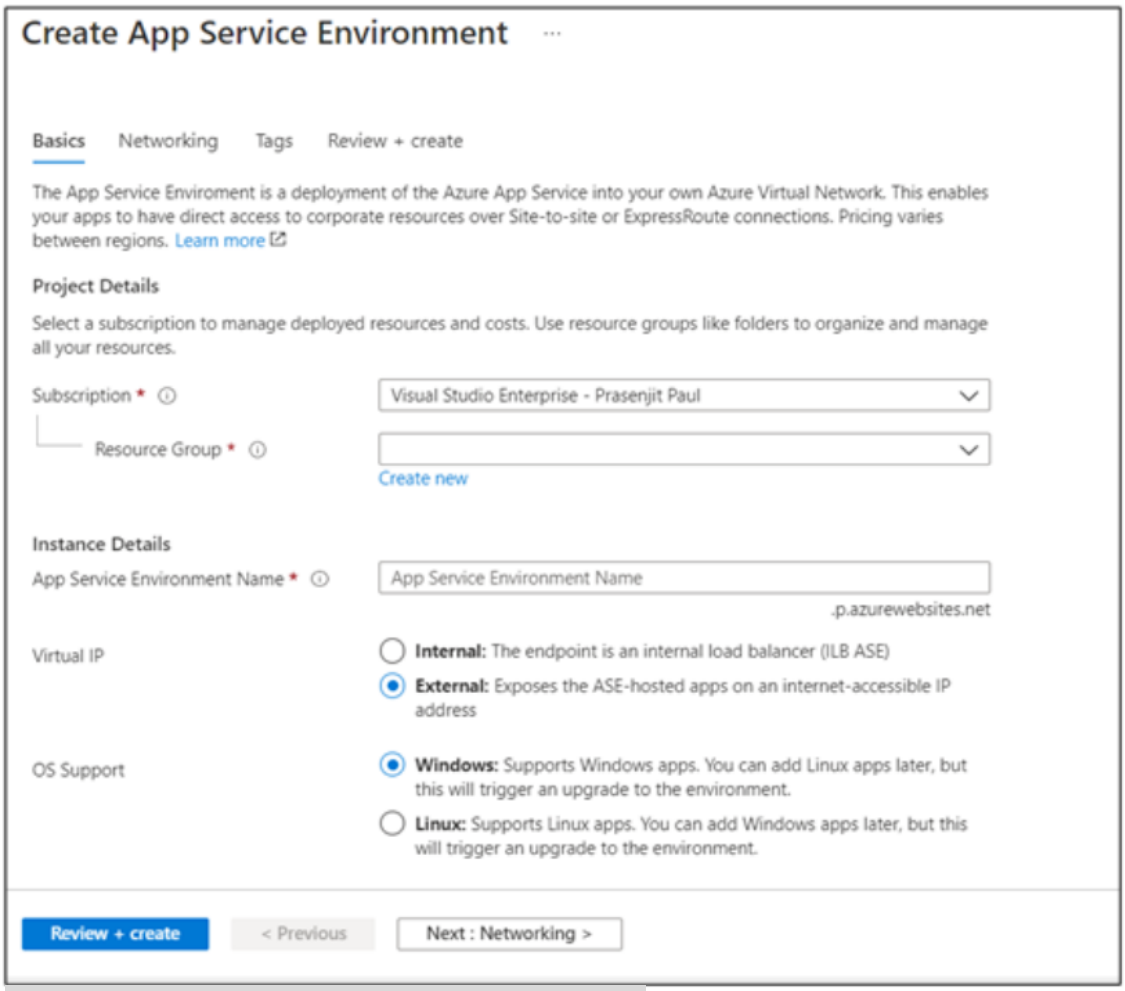


Figure 9: Create standalone ASE

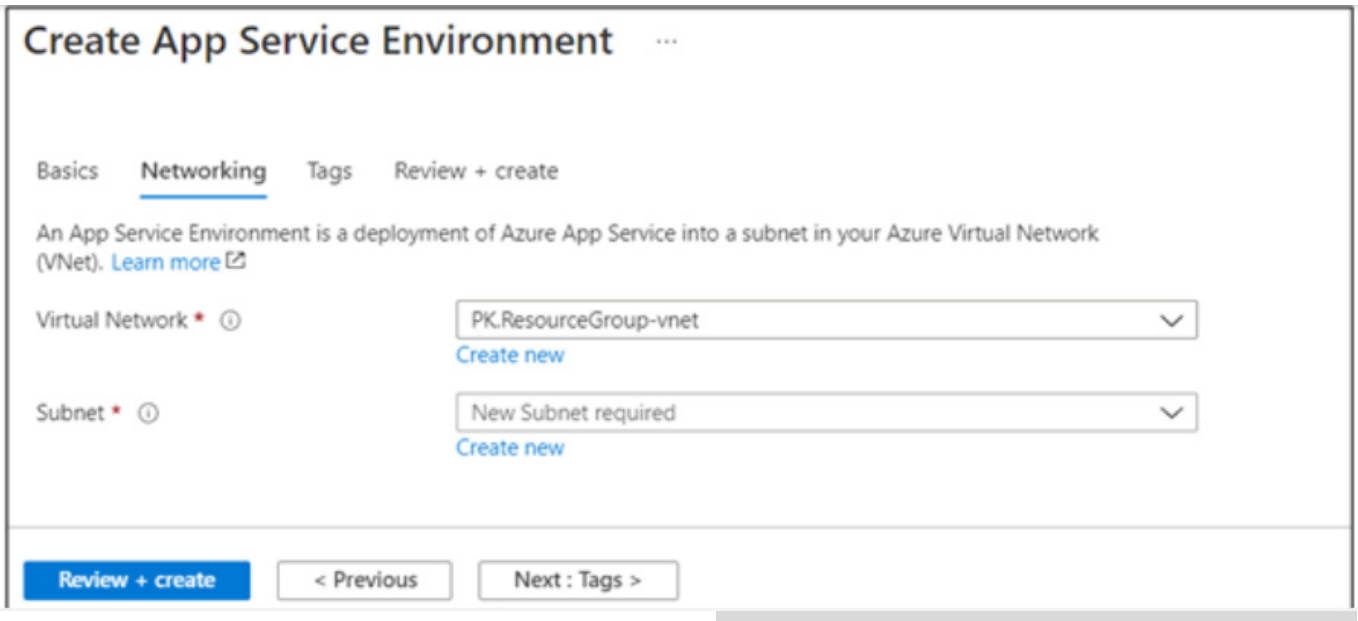


Figure 10: Select virtual network and subnet

3

By deploying ARM template:

Pre-requites for creating ASE through ARM template are:

- Resource Manager VNet.
- A subnet in that VNet. Recommended subnet size is /24 with 256 addresses to accommodate future growth and scaling needs.
- The resource ID from the VNet. This information is available from the Azure portal under your virtual network properties.
- The subscription where the ASE is to be created.
- The location where ASE needs to be deployed.

Steps for ASE creation:

- Create ASE using ARM template. ARM template and examples are available in the reference section at the end.

Parameters to be passed to ARM template while creating ASE:

- aseName
- location
- existingVirtualNetworkName
- existingVirtualNetworkResourceGroup
- existingSubnetName

- While creating ILB ASE, some additional parameters need to be passed:

- internalLoadBalancingMode: The value is set to 3 in most cases which means HTTP/HTTPS traffic and control/data channel ports listened to by FTP service will be bound to an ILB allocated virtual network internal address. If the value is set to 2 then control/data channel ports listened to by FTP service will be bound to an ILB allocated virtual network internal address. HTTP/HTTPS traffic remains on public virtual IP.

- dnsSuffix: This is default root domain assigned to ASE. For ILB ASE, this should be something which is relevant and resolvable to client internal environment.

- ipSslAddressCount: Default value is 0 as there are no explicit IP-SSL addresses for an ILB ASE.

- If External ASE is created, no more steps are required. For ILB ASE, few more steps are required.

- After ILB ASE is created, TLS/SSL certificate that matches ILB ASE domain need to be uploaded. The uploaded certificate is assigned to ILB ASE as default TLS/SSL certificate.

Different versions of ASE (v1, v2, v3)

ASE

In ASEv1, all of the resources need to be managed manually. That includes the front ends, workers, and IP addresses used for IP-based TLS/SSL bindings. Before scaling out App Service plan, the worker pool needs to be scaled out. In ASEv1, user need to pay for each vCPU allocated. That includes vCPUs used for front ends or workers that aren't hosting any workloads. In ASEv1, the default maximum-scale size of an ASE is 55 total hosts. That includes workers and front ends. One advantage to ASEv1 is that it can be deployed in a classic virtual network and a resource manager virtual network.

ASE

With ASEv2, there are no more worker pools to manage. When ASP in ASE is scaled, needed workers are added automatically. In ASE2, user need to select the Isolated pricing plan and accordingly resources are created. With ASEv2, the maximum default scale is now 100. The ASEv2 also now uses Dv2-based dedicated workers which have faster CPU's, twice the memory per core and SSDs. The new ASE dedicated workers' sizes are 1 core 3.5 GB, 2 core 7 GB, and 4 core 14 GB. The end result is that 1 core on ASEv2 performs better than 2 cores in ASEv1.

ASE V3 Latest Version

The flat rate stamp fee per ASE instance has been removed in ASE v3. ASE v3 is available through Isolated v2 pricing plan. As part of Isolated v2 plan, the PAYG rates are reduced and the per instance stamp fee has been removed, reducing the cost of deployment by up to 80%. In ASEv3, there is no longer any inbound or outbound management traffic in the customer VNet. ASE v3 has no internet hosted dependencies being called from the customer network. In ASEv3, the underlying technology is based on Virtual Machine Scale Sets (VMSS) instead of cloud services which provides number of improvements including better load balancers, zone redundancy and multiple other things.

Pricing plans

There is an App Service pricing plan called 'Isolated Plan' which is used for ASEs. All App Service plans that are hosted in the ASE are in the Isolated pricing SKU. Isolated rates for App Service plans can vary by region. In addition to the price of App Service plans, there's a flat rate stamp fee for the ASE itself which is $1.430/hour (~$1,043.811/month). The flat rate doesn't change with the size of ASE. There are 3 categories of Isolated plans which are listed below :

Plan	Cores	RAM	Storage	Pay as you go
I1	1	3.5 GB	1 TB	$0.40/hour
I2	2	7 GB	1 TB	$0.80/hour
I3	4	14 GB	1 TB	$1.60/hour

Isolated v2 Plan (ASE v3)

Plan	Cores	RAM	Storage	Pay as you go
I1v2	2	8 GB	1 TB	$0.562/hour
I2v2	4	16 GB	1 TB	$1.1.24/hour
I3v2	8	23 GB	1 TB	$2.248/hour

Conclusion

This whitepaper was an attempt to give the reader a holistic view of the Azure Service Environment since this Azure PaaS offering is relatively unknown. Since cloud services and their features change frequently. A few things stated here might get outdated eventually. The reader is requested to look up reliable sites from Microsoft for most recent updates. The references used for this whitepaper are provided below.

References:

https://docs.microsoft.com/en-us/azure/app-service/environment/intro

https://docs.microsoft.com/en-us/azure/app-service/environment/using-an-ase

https://docs.microsoft.com/en-us/azure/app-service/environment/using

https://docs.microsoft.com/en-us/azure/app-service/environment/network-info

https://docs.microsoft.com/en-us/archive/msdn-magazine/2017/april/azure-the-new-azure-app-service-environment

ARM Template and example for creating ASE:
https://azure.microsoft.com/en-us/resources/templates/web-app-asev2-create/

ARM Template and example for creating ILB ASE:
https://azure.microsoft.com/en-us/resources/templates/web-app-asev2-ilb-create/

AUTHOR

Prasenjit Paul

Technical Architect, Digital Business, HCL

Prasenjit is a technical Architect with 17+ years of experience in design and development of applications in Microsoft platform and Azure cloud. He has been working on solution design and managing teams engaged in application development in Microsoft platform and Azure. His main areas of expertise are Azure, Azure DevOps, ASP.NET MVC, Angular, SQL Server, and SharePoint.

HCL Technologies (HCL) empowers global enterprises with technology for the next decade today. HCL's Mode 1-2-3 strategy, through its deep-domain industry expertise, customer-centricity and entrepreneurial culture of ideapreneurship™ enables businesses to transform into next-gen enterprises.

HCL offers its services and products through three lines of business - IT and Business Services (ITBS), Engineering and R&D Services (ERS), and Products & Platforms (P&P). ITBS enables global enterprises to transform their businesses through offerings in areas of Applications, Infrastructure, Digital Process Operations, and next generation digital transformation solutions. ERS offers engineering services and solutions in all aspects of product development and platform engineering while under P&P. HCL provides modernized software products to global clients for their technology and industry specific requirements. Through its cutting-edge co-innovation labs, global delivery capabilities, and broad global network, HCL delivers holistic services in various industry verticals, categorized under Financial Services, Manufacturing, Technology & Services, Telecom & Media, Retail & CPG, Life Sciences, and Healthcare and Public Services.

As a leading global technology company, HCL takes pride in its diversity, social responsibility, sustainability, and education initiatives. As of 12 months ending on March 31, 2021, HCL has a consolidated revenue of US$ 10.17 billion and its 168,977 ideapreneurs operate out of 50 countries. For more information, visit www.hcltech.com

www.hcltech.com

WI-105242708270-EN00GL