

Azure App Services Architecture

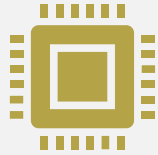
conf.dr. Cristian Kevorchian
University of Bucharest
Computer Science Department



Azure App Service Architecture



Azure App Service is considered an excellent Platform as a Service , offering an application platform for developers to build Web, mobile and API applications



Its offerings range from simple marketing and digital presence applications to scalable e-commerce solutions and hyper-scale, customizable applications



App Service is fully managed, which means no administrative tasks are required to manage underlining compute infrastructures on which your applications run

Global and Geo- Distributed Architecture

User makes a request to create a new site

ARM makes sure user has access to the resource to allow the given operation and forwards the requests to App Service Geo-Master

Geo-Master finds the best suitable scale unit for the user's request and forwards the request

The scale unit creates the new application

Geo-Master reports success on the create request

App Service Scale Unit



An App Service scale unit is a collection of servers that host and run your applications

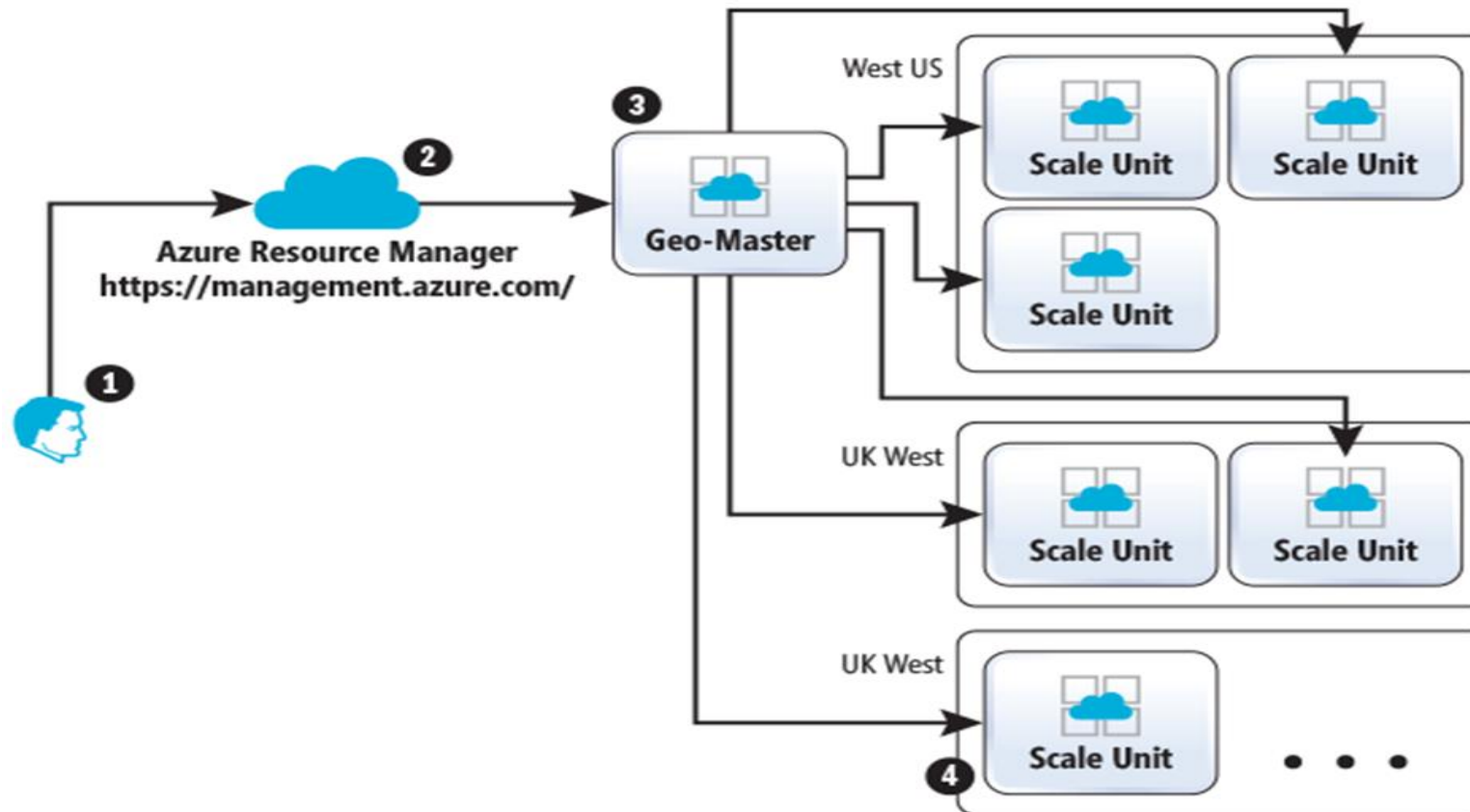


The clustering of servers enables economy of scale and reuse of shared infrastructure



The fundamental building block of App Service scale unit is an Azure Cloud Service deployment

Scale Unit



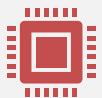
Scale Unit Main Building Blocks



The main functionality of **scale unit** is to host and run customer applications



Applications run on Windows servers and are referred to as Web Workers or Workers for short



However, a unit of scale includes several additional support servers required to achieve the functionality provided by App Service

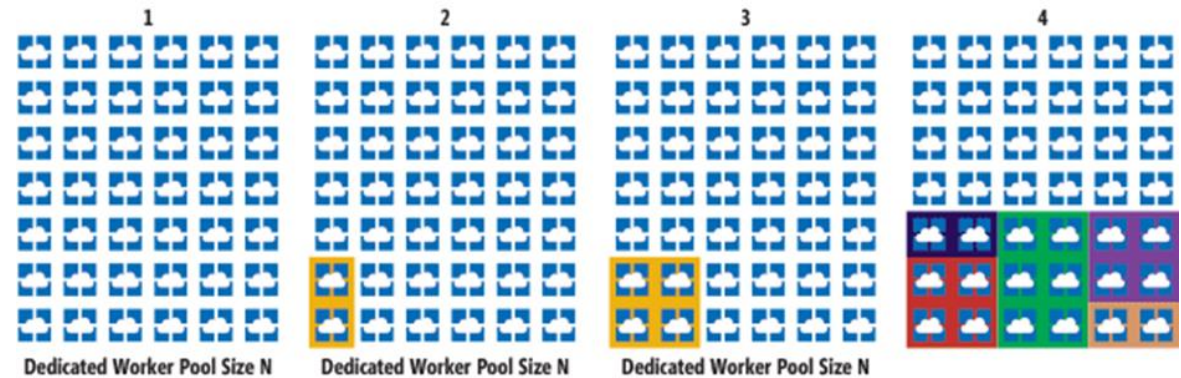
Front End

The front end is a layer seven-load balancer, acting as a proxy, distributing incoming HTTP requests between different applications and their respective Workers

Currently, the App Service load-balancing algorithm is a simple round robin between a set of servers allocated for a given application


Web Workers

The figure shows multiple App Service Plans, identified as multi-colored rectangles, each representing an App Service Plan that can belong to multiple customers



File Servers

Any app needs storage to hold content such as HTML, .js files, images, or code files, and any other content required for the application to work



A file server mounts Azure Storage blobs and exposes them as network drives to the Worker

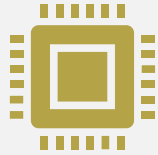


A Worker, in return, maps such network drives as local, allowing any application running on any given Worker to use the "local" drive, just like you would expect if the application were running on a server using its local disk

API Controllers



API controllers can be viewed as an extension to App Service Geo-Master




While the Geo-Master is aware of all App Service applications across all scale units, it's the API controller that actually performs the management operation that affects your applications




Geo-Master delegates API fulfilment to a given scale unit via the API controllers

Publishers

Because app content is stored in Azure Storage blobs and mapped by file servers, App Service has a Publisher role to expose FTP functionality



The Publisher role lets customers use FTP to access their application content and logs



One of the common deployment methods is Web Deploy or any of the supported Continuous Deployment options such as Visual Studio Release Manager or GitHub

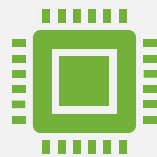
SQL Azure



Each App Service scale unit uses Azure SQL Database to persist application metadata



Each application that's assigned to a given unit of scale has a representation in a SQL Database



The SQL Database is also used to hold runtime information about applications

Data Role

As examples: a Web Worker needs site configuration information when launching an app; front ends need to know which servers are assigned to run a specific application in order to correctly forward HTTP requests to the appropriate servers; and controllers read and update data from the database based on API calls made by customers



The data role can be described as a cache layer between SQL Database and all other roles in a given unit of scale



It abstracts the data layer from the rest of the roles, improving scale and performance, as well as simplifying software development and maintenance

Less-Than-Obvious Best Practices

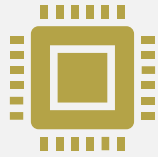
Now that you know how Azure App Service is built, we'll review several tips and tricks from the App Service team

These are hands-on lessons learned by App Service engineering teams from numerous customer engagements

Controlling Density



40 low-volume applications remain in a single App Service Plan running on one compute resource

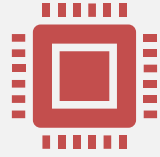


Five mid-to-low volume applications use a second App Service Plan running on one compute resource



The remaining five applications are found to have high-volume usage

Per-App Scaling



40 low-volume applications set to run on a maximum of a single server each



Five mid- to low-volume applications set to run on a maximum of two servers each



Five remaining high-volume applications set to run on a maximum of 10 servers

Application Slots



Create additional App Service Plan for the non-production slots



Move a non-production slot to a different App Service Plan and, thus, a separate pool of compute resources



Carry out resource-intensive tasks while running in the separate App Service Plan



When the non-production slot is ready to be swapped into production, move it back to the same App Service Plan running the production slot

Deploying to Production with No Downtime

You have a successful application running on an App Service Plan and you have a great team to make updates to your application on a daily basis

The swap operation doesn't restart your application and in return the Controller notifies the front-end load balancer to redirect traffic to the latest slots

Some applications need to warm up before they can safely handle production load—for example, if your application needs to load data into cache, or for a

You can use Continuous Deployment tools such as Visual Studio Release Manager to set up a pipeline for your code to get deployed into pre-production slots, run test for verification and warm all required paths in your app prior to swapping it into production

Scale Unit Network Configuration



The App Service scale unit is deployed on Cloud Service



The scale unit has a single Virtual IP address exposed to the world



In the case of IP-based SSL, a given application is allocated a dedicated IP address for only inbound traffic, which is associated with the Cloud Service deployment

Public VIP



Line #1 runs an nslookup querying resolution for `awseomwebapp.azurewebsites.net`



Line #5 shows the domain name of the scale unit running `awseomwebapp` app



Line #6 shows the VIP of the scale unit




Line #7 shows all domain aliases mapped to the same IP address

Outbound VIPs


If we looking for a dedicated set of inbound and outbound IPs, you can explore this by using a fully isolated and dedicated App Service Environment.

IP and SNI SSL

When using IP-SSL, App Service allocates to your application a dedicated IP address for only inbound HTTP traffic



Unlike the rest of Azure dedicated IP addresses, the IP address with App Service via IP-SSL is allocated as long as you opt to use it



App Service also supports SNI SSL, which doesn't require a dedicated IP address and is supported by modern browsers

Network Port Capacity for Outbound Network Calls

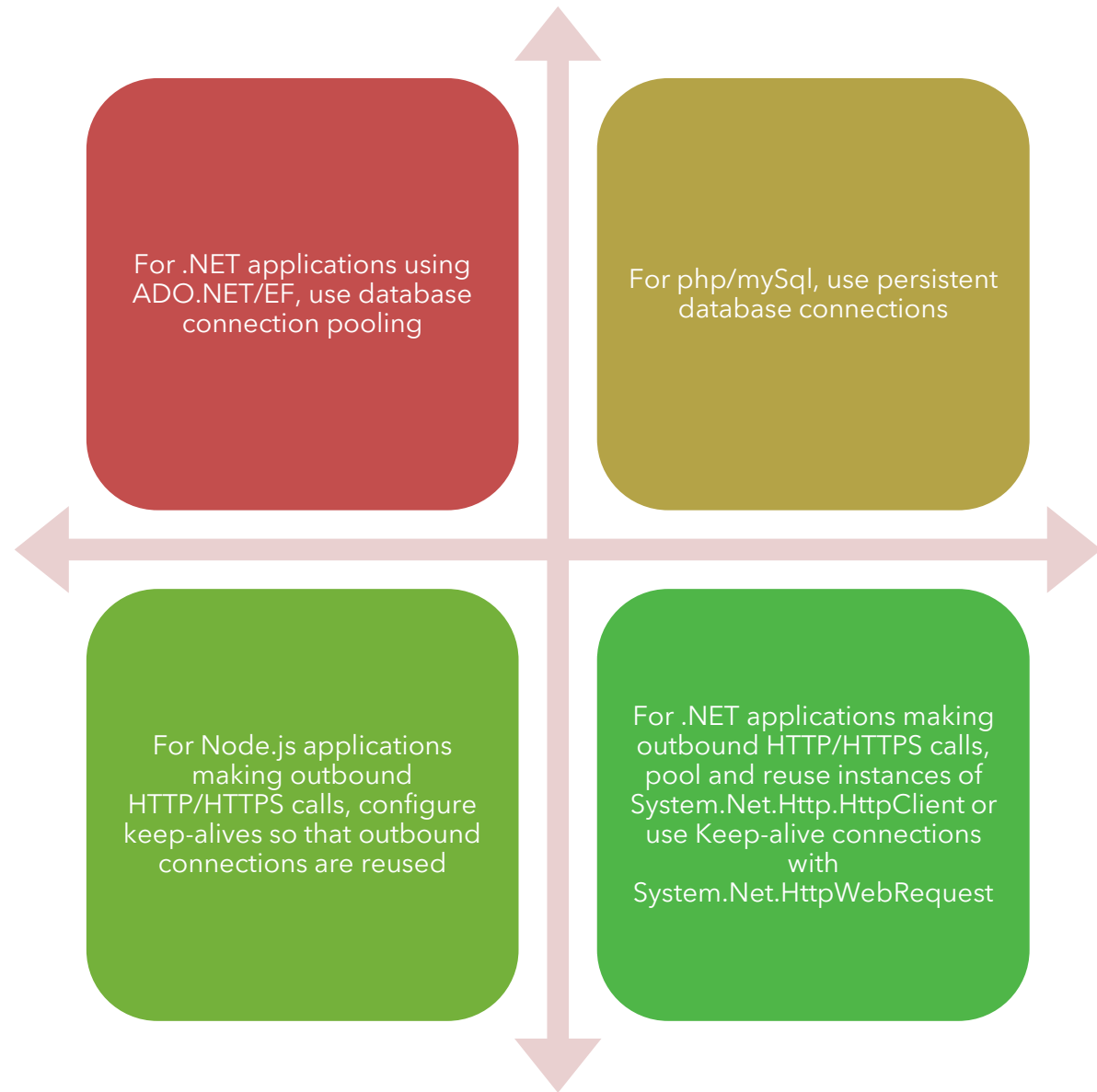
1,920
connections
per
B1/S1/P1
instance

3,968
connections
per
B2/S2/P2
instance

8,064
connections
per
B3/S3/P3
instance

64K max
upper limit
per App
Service
Environment

Network Port Capacity for Outbound Network Calls



Wrapping Up



Azure App Service provides a rich PaaS offering for Web, mobile and API applications



While the service has a lot of moving internal parts, these are abstracted away so that developers can concentrate on writing great apps while the service handles the complexities of running those apps at global scale



Having a good understanding of how applications map to Web Workers in an App Service Plan is important as you increase the number of your apps running on the service, while still maintaining an optimal scaling configuration