15 July 2015 G00275100

Analyst(s): *Elias Khnaser*

Many IT organizations evaluate AWS and Azure for their cloud initiatives but are stuck between marketing information and a time-consuming exercise of finding the right data points. This document presents a side-by-side technical comparison of the core infrastructure differences.

Key Findings

There is a fundamental architectural and design difference between Amazon Web Services (AWS) and Azure when it comes to high availability (HA). Azure only offers HA within a single logical data center. In contrast, AWS implements HA across multiple logical data centers.

AWS Auto Scaling offers dynamic, scale-up and scale-down provisioning, termination, and instance recovery, whereas Azure Autoscale focuses on power on/off of preprovisioned virtual machines (VMs).

AWS's Direct Connect offers more partner options and more port speeds than Azure ExpressRoute Exchange Provider (EP). However, Azure ExpressRoute EP is less expensive than AWS Direct Connect.

AWS and Azure have strong storage offerings with high-durability expectations and strong-availability SLAs. However, AWS has an advantage with high-performance block storage.

Recommendations

Identify your application's availability, compute, networking and storage design criteria. This will help you level-set the differences between AWS and Azure.

Monitor your AWS and Azure monthly spend, and subscribe to discount programs like AWS Reserved Instances and Microsoft Enterprise Agreement as quickly as possible.

Choose AWS if your design criteria require local or global scalability, as well as HA across multiple logical data centers that are synchronously connected.

Choose Azure if your organization currently relies heavily on Microsoft technologies like Hyper-V/System Center and you are looking for a simple way to integrate with the cloud.

Organizations are moving to the cloud, but it's a crowded space, with many different providers. However, two cloud providers dominate market share, as well as Gartner clients' interest and evaluation: AWS and Microsoft Azure. It is important for clients to understand the difference in architecture between the providers and how that affects infrastructure, application design and deployment.

AWS and Azure architectures are large, complex and difficult to understand. The technologies and architectures used by cloud providers are often proprietary and therefore may not be divulged. However, for AWS and Azure, sufficient information exists to determine the key components of the architectures, how they work together and how that affects the customer. In particular, this document will focus on the key differentiators across four categories:

Availability: This section focuses on understanding the global infrastructure footprint and capabilities of AWS and Azure that will help cloud architects design and deploy applications that meet their availability and uptime requirements.

Network: This section focuses on AWS's and Azure's networking connectivity and feature set. Particular attention is paid to components such as load balancers, internetworking on the provider's platform and how the two vendors compare from a network architecture viewpoint.

Compute: This section will focus on differences between how AWS and Azure implement compute elasticity, provisioning, financials and container strategy.

Storage: In the storage section, we focus on the differences between AWS and Azure when it comes to local, block, object and network-shared storage. Furthermore, we analyze data durability, sovereignty and encryption.

Determining the suitability of a particular application requires an understanding of how the architectural differences in these categories will affect the design of the application itself, the difficulty of deployment and the overall ability of the application to serve its users. Table 1 highlights some of the major differences between AWS and Azure across each of these four categories. Additional discussion into the specifics of each of these categories is provided later in this document.

Table 1. AWS Versus Azure Key Difference Highlights

Category	AWS	Microsoft Azure
Availability	AWS offers regions and Availability Zones (AZs).	Azure offers regions and Availability Sets.
Network	The native load balancer service supports Layer 4 and Layer 7 capabilities.	The native load balancer service works at Layer 4. Azure Application Gateway offers Layer 7 capabilities.
Compute	Auto Scaling offers dynamic, scale-up and scale-down provisioning/termination, as well as instance recovery.	Autoscale focuses on power on/off preprovisioned VMs.
Storage	Elastic File System (EFS) for network-shared storage is based on Network File System (NFS) v4.0.	Azure Files for network-shared storage is based on Server Message Block (SMB) 2.1.

Source: Gartner (July 2015)

The cloud infrastructure as a service (IaaS) market is a very crowded space. However, two providers, AWS and Microsoft Azure, have clearly distinguished themselves by offering superior features and capabilities that are geared toward enterprises of all sizes. This assessment is corroborated by the findings of Gartner's "Magic Quadrant for Cloud Infrastructure as a Service, Worldwide" (http://www.gartner.com/document/code/265139?ref=grbody&refval=3093919) and in-depth assessments of major public IaaS providers, which all show that AWS and Azure are clear leaders in the public cloud IaaS market (links to these in-depth assessment documents are provided in the Gartner Recommended Reading section near the end of this document). Furthermore, Gartner's client inquiries on evaluating IaaS cloud service providers are largely about comparing the capabilities and features of AWS and Azure.

However, AWS and Azure did not just simply slide into this pole position because of their brand recognition or brand power. Instead, both providers invest heavily in building a global infrastructure and layered platform that are capable of delivering cloud characteristics that clients are expecting, including:

- Self-service capabilities, so that all of their services can be easily and instantly provisioned
- Elasticity, so that clients can grow and shrink their cloud environment based on their business needs
- Broad network access, so that application administrators can find a viable alternative on the cloud service provider's platform should enterprises decide to migrate existing, or develop new, applications
- Granular security, regulatory compliance certifications and robust operational capabilities, so that clients can satisfy their regulatory, privacy and security requirements

Everything described so far has led to AWS's and Azure's current position in the market. What promises to keep them in these leading roles is their continuous innovation and introduction of new features, while still reducing prices for end users. With regard to innovation, both AWS and Azure are:

- Embracing containers with support for Docker
- Introducing technology in the form of machine-learning services that can be leveraged for use with the Internet of Things (IoT)
- Supporting mobile applications by offering quick development and deployment services
- Supporting vertical-specific requirements like high-performance computing (HPC)
- Supporting government requirements for security and increased isolation with offerings such as AWS GovCloud and Azure Government Cloud that can only be accessed by government agencies and partners that are certified to work with government agencies

This document is intended to help cloud architects who need to understand the key architectural and service differences between AWS and Azure from an availability, networking, compute and storage standpoint.

Availability

When considering public cloud IaaS, it is important that organizations understand the scale of the cloud service provider's global infrastructure footprint and how it is designed. Having this knowledge enables cloud architects to design highly available workloads by leveraging the characteristics and capabilities of this global infrastructure. This section highlights the differences between AWS's and Azure's global infrastructure implementations and how their architecture affects the design and availability of applications. This section focuses on differences in the following areas:

- Global infrastructure
- High-availability modes
- Availability terminology comparison
- Designing availability in AWS and Azure

Key Take-Aways

AWS's multiple Availability Zones (AZs; logical data centers) within a region allow it to support all three HA modes (local mirroring, synchronous mirroring and asynchronous mirroring), while Azure's single logical data center per region means that it can only support the local mirroring and asynchronous mirroring modes.

AWS regions house multiple AZs that are within 60 miles (100 kilometers) of each other, thereby offering synchronous connectivity. Conversely, Azure regions house a single logical data center and are greater than 60 miles (100 kilometers) apart, thereby only able to offer asynchronous connectivity.

AWS offers 11 regions and 30 AZs (logical data centers), including two AZs for the GovCloud region, while Azure offers 19 regions (logical data centers), including two regions dedicated for government cloud.

AWS services are enabled regionwide, across all AZs. For example, Elastic Load Balancing, Virtual Private Clouds (VPCs) and Auto Scaling are all services that span the entire region, thereby offering higher availability and redundancy. Azure services are deployed within the same logical data center (Azure region), thereby offering less availability and redundancy.

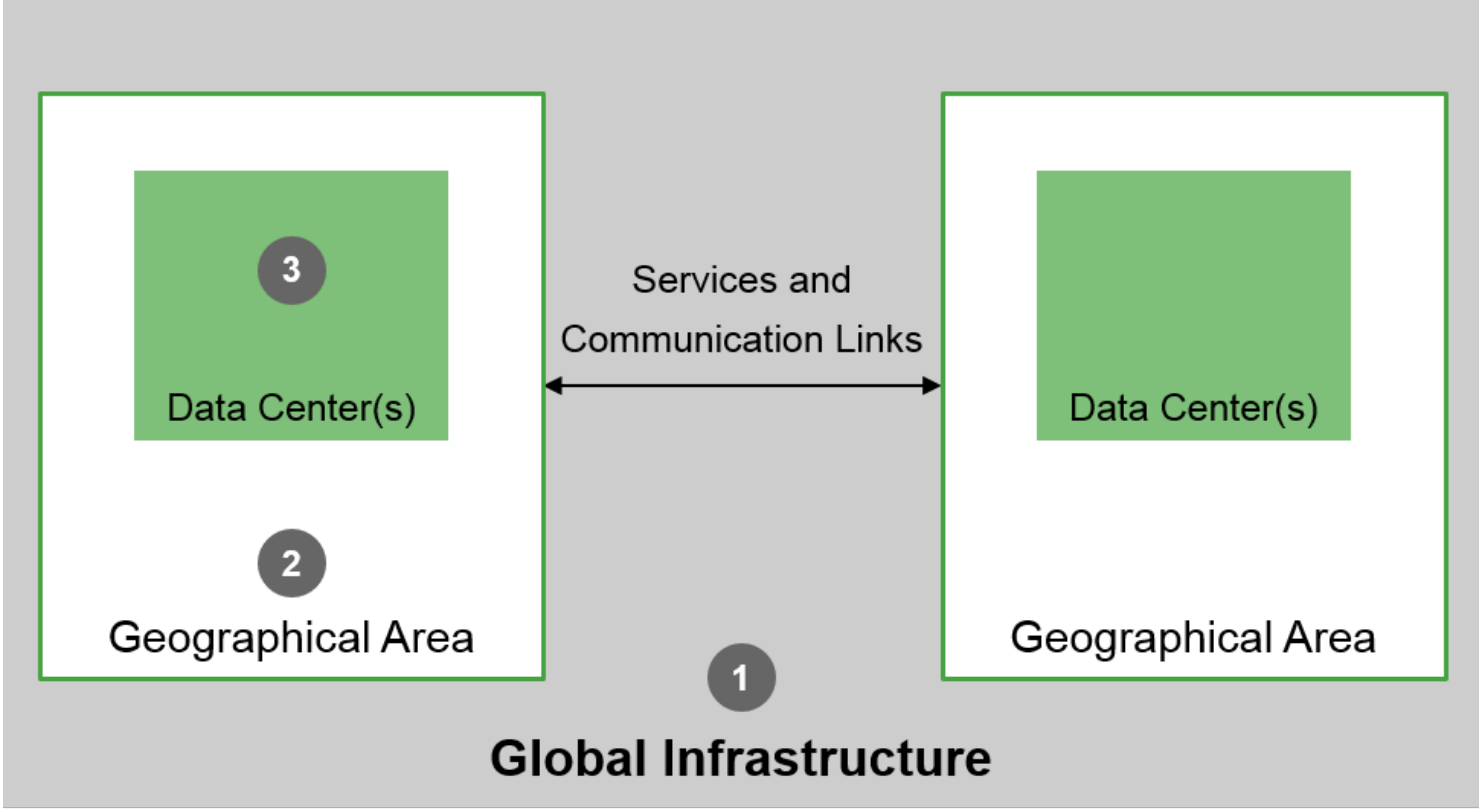
Azure is more highly available within a single logical data center architecture, leveraging technologies like Availability Sets, Fault Domains and Update Domains, whereas AWS is more highly available across multirole logical data centers (AZs).

AWS's SLA for compute requires that instances be deployed in two separate AZs within the same region. Azure's SLA requires that at least two same-role VMs be deployed in different Fault Domains and Update Domains within the same Availability Set.

Global Infrastructure

The infrastructures built by AWS and Microsoft Azure have much in common, but there are also some key differences. Figure 1 outlines the basic building blocks used by both vendors.

Figure 1. Logical Global Infrastructure



Source: Gartner (July 2015)

The key differences (as noted in Figure 1) are:

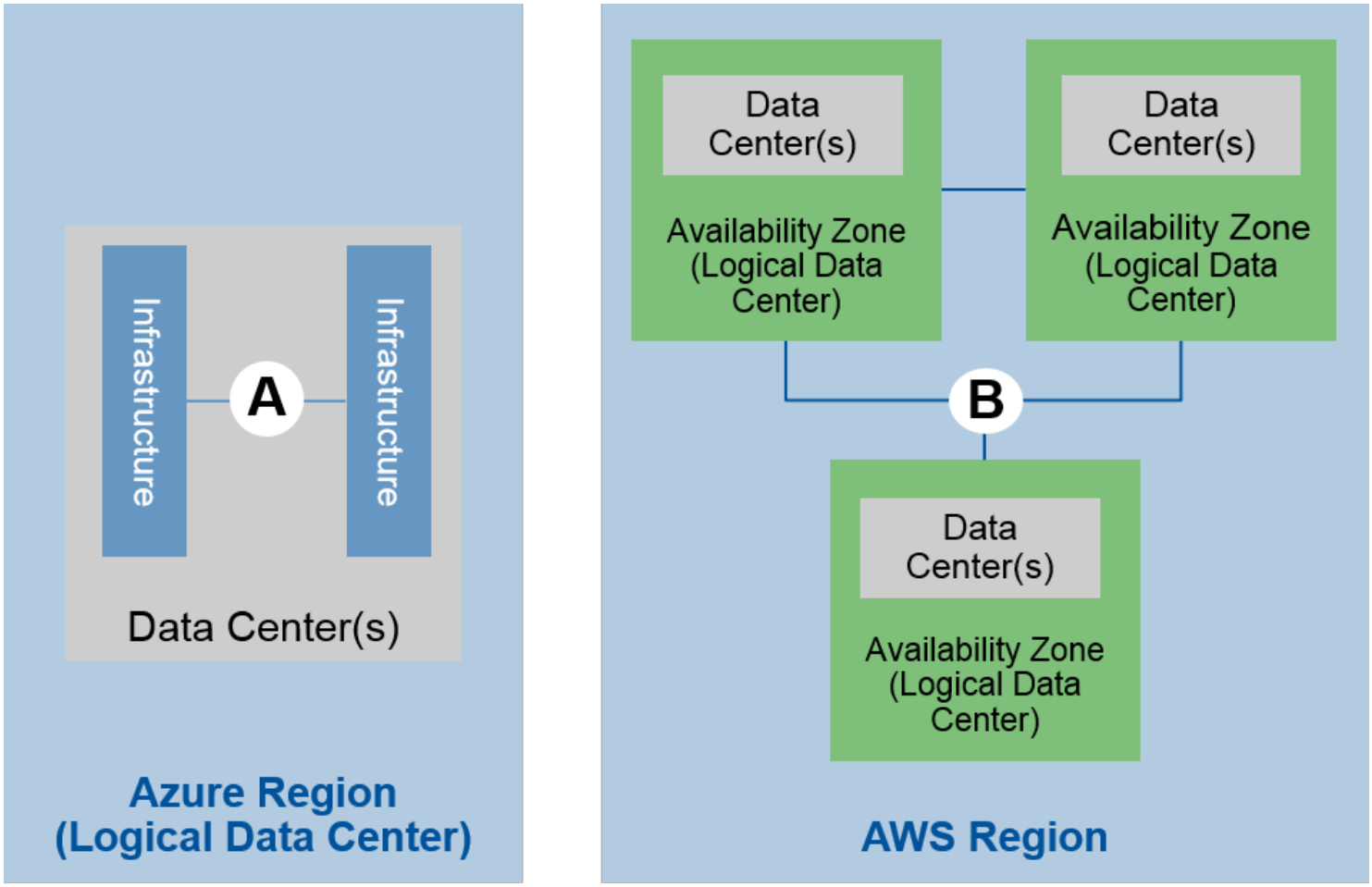
1. **Global infrastructure:** Both Azure and AWS offer infrastructure that spans the globe, with regions in multiple geographic locations. For an updated list of geographies, visit the following Web pages:

   AWS regions (http://aws.amazon.com/about-aws/globalinfrastructure/regional-product-services/)

   Microsoft Azure regions (http://azure.microsoft.com/en-us/regions/#overview)

2. **Regional implementation:** Within each geographic area, there will be one or more logical data centers supported by one or more physical data centers. Figure 2 illustrates the clear distinction and terminology differences.

Figure 2. Regional Implementation



Source: Gartner (July 2015)

For Azure, a region is covered by a single logical data center (A), which is in turn supported by one or more physical data centers. This architecture renders an Azure "region" synonymous to a logical data center. The lack of multiple logical data centers in a region limits HA options to those that are available within a single logical data center and those that are possible between widely geographically dispersed data centers (that is, a logical data center in another Azure region).

In contrast, AWS has multiple AZs within each AWS region (B), and each AZ has one or more physical data centers, thereby rendering an AZ synonymous to a logical data center. It is worth noting that, while each AZ has one or more physical data centers, a data center is confined to a single AZ and is never in two AZs.

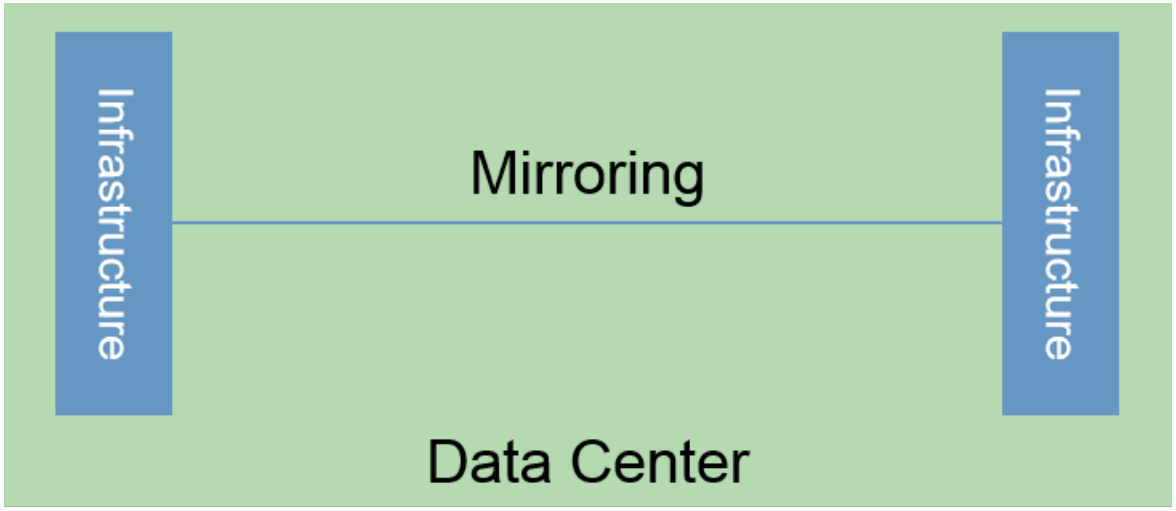
3. **Services and communication:** AWS and Azure interconnect their regions with private data links. These links provide a variety of built-in services, such as replication, as well as serve as a conduit for communications between applications that are hosted in the different regions.

High-Availability Modes

As organizations move more business-critical applications to the cloud, there is a need to deliver HA solutions to protect against faults in the provider's infrastructure from interrupting access to the application. Traditionally, applications requiring some degree of HA and disaster tolerance have been implemented using one or more of the following operating modes:

Local synchronous mirroring and failover: In this approach (see Figure 3), a secondary copy of the infrastructure running the application is created, with data mirrored between the primary and the secondary. This approach can handle a failure in the primary infrastructure in seconds by switching over to the secondary infrastructure. However, a catastrophic data-center-wide problem, such as power loss or fire, will cause application downtime.

Figure 3. Synchronization Within a Data Center



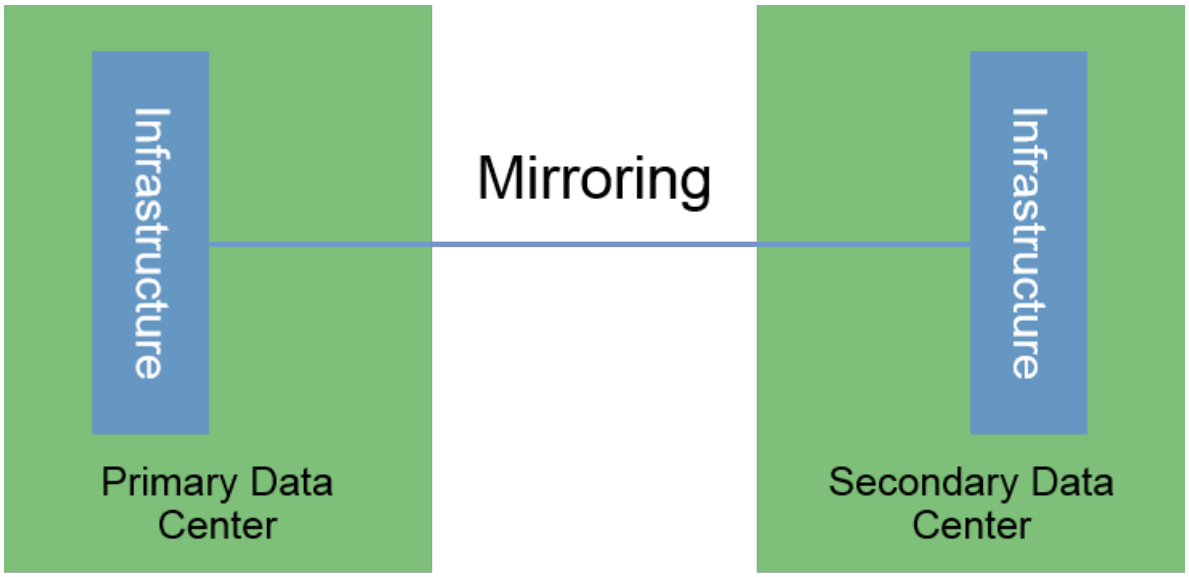
Source: Gartner (July 2015)

Supported by AWS? — Yes

Supported by Azure? — Yes

Remote synchronous mirroring and failover: In this approach (shown in Figure 4), the key difference is that a secondary infrastructure is in a separate data center, which can be up to 60 miles (100 kilometers) from the primary. The data is still mirrored in real time, so near-instant failover is possible. This approach offers similar levels of HA to the local option but with the advantage that the data center is removed as a single point of failure.

Figure 4. Synchronous Mirroring Between Data Centers



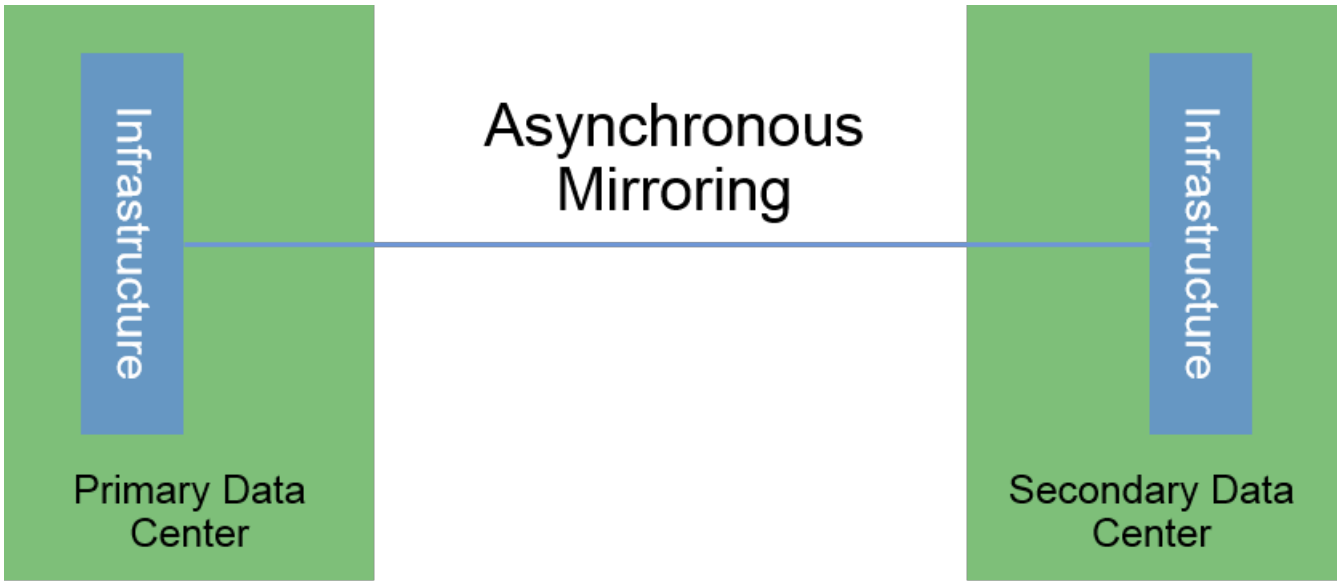
Source: Gartner (July 2015)

Supported by AWS? — Yes

Supported by Azure? — No

Remote asynchronous mirroring and failover: In this approach (shown in Figure 5), the key difference is that data is replicated asynchronously so that the secondary data center can be located anywhere in the world. Asynchronous mirroring means that the remote copy of the data lags behind the real application, preventing instant failover. (The amount of lag will vary based on distance, bandwidth and latency.) This approach cannot match the near-instant failover that is possible with the previous two methods, but it does provide additional protection from regional disasters such as earthquakes and can be an effective data protection solution.

Figure 5. Asynchronous Mirroring Between Data Centers



Source: Gartner (July 2015)

Supported by AWS? — Yes

Supported by Azure? — Yes

Availability Terminology Comparison

Comparing AWS and Azure infrastructures is made harder by the lack of standard terminology between the providers. This problem is compounded by the decision by AWS and Azure to use the same terms to mean different things. Table 2 provides a simple guide to each provider's availability terminology.

Table 2. Availability Terminology Comparison

Industry	AWS	Microsoft Azure
Geographic presence with resource-hosting capabilities	Region	Region
Geographic area with multiple logical data centers	Region	Microsoft doesn't offer multiple logical data centers within a geography
Logical data center	Availability Zone	Region
Physical data center	Data center	Data center

Source: Gartner (July 2015)

AWS and Azure both use the term "region." However, it is important to note that a region has two characteristics as follows:

Geographic presence with resource-hosting capabilities. This refers to a geography where AWS and Azure have cloud hosting capabilities (IaaS, PaaS). Both AWS and Azure have this characteristic.

Geographic area with multiple data centers. This refers to a geographic area with high availability, isolation and multiple logical data center capabilities. At this time, only AWS has this characteristic.

GEOGRAPHIC PRESENCE WITH RESOURCE-HOSTING CAPABILITIES

AWS and Azure both have global geographic presence across the board with regions in North and South America, Europe, Asia (including China), and Australia. Increasing the global geographic presence with hosting capabilities is important because it brings AWS and Azure clouds closer to clients, thereby overcoming data residency issues and performance issues.

As of this writing, AWS offers 11 regions and 30 AZs, including two for the AWS GovCloud (U.S.) region, while Azure offers 19 regions, including two regions dedicated for government cloud. Clients seeking to be within close proximity of AWS or Azure should examine the geographical location of those providers' hosting data centers to determine suitability.

Table 3. AWS and Azure Hosting Data Centers' Geographical Locations

	North America	South America	Europe	Asia	Australia
AWS	Oregon, North Virginia, Northern California and AWS GovCloud (U.S.)	São Paolo	Ireland and Frankfurt	Singapore, Tokyo and China	Sydney
Azure	Iowa, Virginia (East U.S.), Virginia (East U.S. 2), Illinois, Texas, California, U.S. Gov Iowa and U.S. Gov Virginia	São Paolo	Ireland and Netherlands	Hong Kong, Singapore, Tokyo, Osaka, China North and China South	New South Wales and Victoria

Source: Gartner (July 2015)

For an updated list of geographies, visit the following Web pages:

AWS Global Infrastructure (http://aws.amazon.com/about-aws/global-infrastructure/)

Azure regions (http://azure.microsoft.com/en-us/regions/#overview)

GEOGRAPHIC AREA WITH MULTIPLE LOGICAL DATA CENTERS

AWS has the concept of a region, which is a specific geographic area that contains two or more AZs and has the following characteristics:

- A region houses multiple AZs that are within 60 miles (100 kilometers) and less than 2ms of each other. The importance of this architecture is that it enables synchronous replication and distributed application deployments.
- AWS services are enabled regionwide. For example, Elastic Load Balancing, VPCs and Auto Scaling are all services that span the entire region.
- AWS currently has 10 regions worldwide, with several AZs in each region. AWS also has an eleventh region, which is dedicated to U.S. government agencies, known as AWS GovCloud (U.S.).

Azure, on the other hand, does not have the equivalent of an AWS region. However, Azure does use the term region to signify a logical data center, as described in the next section.

LOGICAL DATA CENTER

AWS and Azure use different terms to refer to a logical data center. AWS's AZs house one or more physical data centers within the same data center campus, or within close proximity. Physical data centers within an AZ are interconnected and have .25ms latency; this configuration creates a single logical data center where resources such as instances can be hosted.

On the other hand, Azure regions also consist of one or more physical data centers in the same data center campus, or within close proximity. Physical data centers within an Azure region are also interconnected, thereby creating a single logical data center against which clients can host resources like VMs.

The key differences are:

- An AWS region consists of two or more AZs (logical data centers), whereas all Azure regions consist of a single logical data center — this is why an AWS AZ is the equivalent of an Azure region. (AWS and Azure do not publish which AZs or regions consist of more than one physical data center.)
- AWS can offer synchronous replication and distributed applications spread across multiple AZs in the same geographic area; Azure cannot.

While both AWS AZs and Azure regions are engineered to be insulated from failures (such as compute management software, servers, power units and top-of-rack switches), AWS's AZ architecture and isolation offers more protection against events such as "fires." However, both an AWS region with a multiple-AZ architecture and an Azure region with single logical data center architecture will be similarly affected by large natural disasters in the same geographic area. This is a result of the close-proximity nature of the physical data centers in the AWS AZs and Azure regions.

Today, AZs give AWS a definitive availability advantage. If Azure evolves its infrastructure architecture in the future to include AZ-like functionality in the same region, it is unclear if the name Availability Zones will be used. For that matter, it is important to note that, today, AWS and Azure use the term region in a completely different way, and region architecture and functionality are also completely different between the two providers.

AZURE AVAILABILITY TERMINOLOGY

Microsoft Azure has three commonly used terms that broadly align with common industry terms (as shown in Table 4).

Table 4. Azure and Industry Availability Terminology

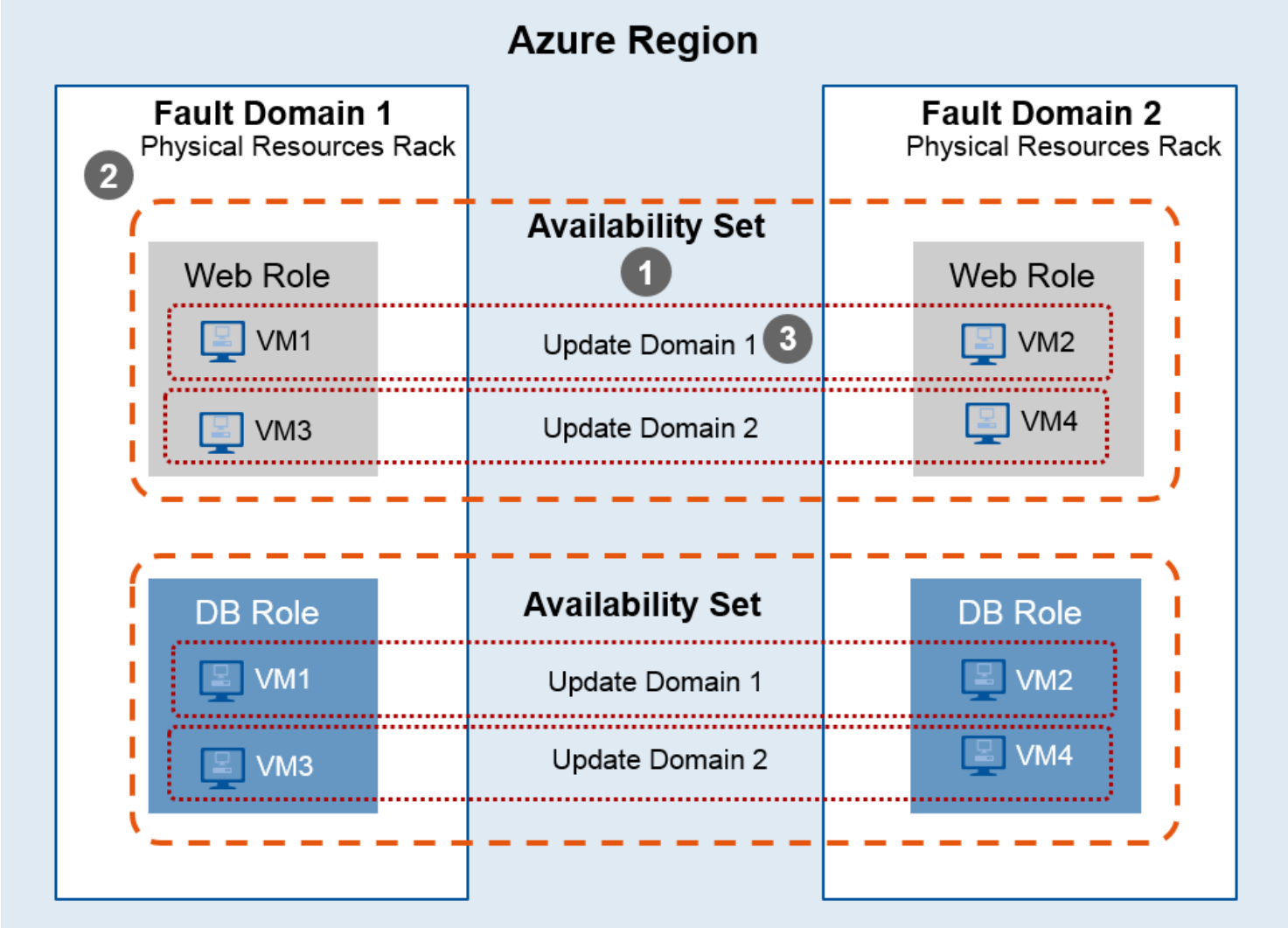
Industry Term	Microsoft Azure
Logical grouping of compute instances	Availability Set
Anti-affinity group (protect against software updates)	Update Domain

Anti-affinity group (protect against hardware failure)	Fault Domain

Source: Gartner (July 2015)

In Figure 6, an architectural diagram of Availability Sets, Fault Domains and Update Domains is provided to highlight how they work together to provide HA within an Azure region.

Figure 6. Azure Availability Sets, Fault Domains and Update Domains



Source: Adapted from Microsoft

The Azure availability components illustrated in Figure 6 are:

1. **Availability Sets:** Microsoft Azure has the concept of an Availability Set (AS), which is a collection of logical compute instances that perform the same function in a customer application (for example, one AS for Web servers, another for application servers and a third for database servers). An AS is used to group these instances so that they can be placed in different Fault Domains for hardware fault isolation, as well as in different Update Domains to maintain availability during updates by only updating a portion of the AS at a time.

2. **Fault Domains:** Azure uses Fault Domains as a way to separate same-role VMs that are deployed in an AS to avoid a hardware single point of failure. As shown in Figure 6, a Fault Domain is an isolated set of resources. It is designed to avoid specific failure points in the topology (for example, power, networking and servers); it often, but not always, aligns to a data center "rack."

When a service is deployed on Microsoft Azure, you are required to deploy at least two components (for example, VMs) of that service in different Fault Domains and different Update Domains in order to meet the requirements of Azure's SLA. Fault Domains provide an algorithm to ensure that services don't end up on the same physical hardware, which is why a Fault Domain is more widely known in the industry as an anti-affinity group.

AWS does not have the concept of an affinity or anti-affinity group per se. However, by deploying instances into different AZs, an organization is able to mimic anti-affinity. Conversely, deploying instances or resources within the same AZ accomplishes the equivalent of an affinity group. Though theoretically possible, AWS's approach is manual and potentially error prone. Affinity and anti-affinity concepts should be policy-based and designed to automatically follow rules for affinity and anti-affinity when provisioning applications.

3. **Update Domains:** Another form of anti-affinity group that Azure offers is called an Update Domain. Update Domains are created from a group of resources that are updated together. Deploying a VM across two different Update Domains protects against bad software updates or flaws in the update system in the same way that Fault Domains protect against hardware failure. Update Domains also prevent outages caused by reboots of the physical host or guest OS that are triggered by a software update. Therefore, an Update Domain is intended to prevent VMs that have the same role from suffering an outage from software updates.

The advantage of Fault Domains and Update Domains is that they enable customers to get very fine-grained availability guarantees from the platform. For example, only one-tenth of a customer's service is taken down at one time for platform updates if a customer selects 10 Update Domains. AWS does not offer Fault Domain- and Update Domain-like technology. In order to accomplish the same results on the AWS platform, clients would need to deploy instances across 10 AZs. For that matter, Azure is more highly available within a single logical data center architecture, whereas AWS is more highly available across multirole logical data centers (AZs).

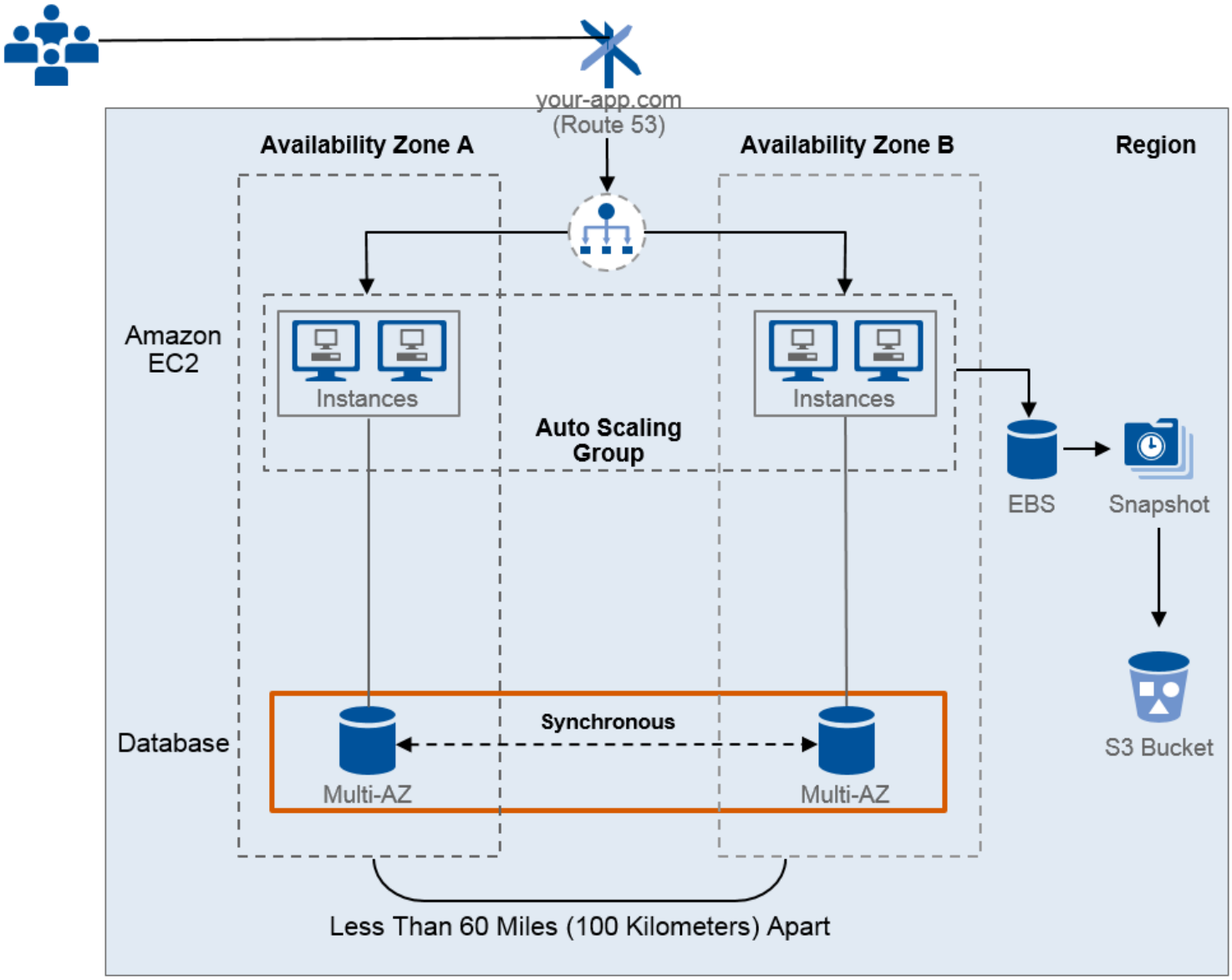
Designing Availability in AWS and Azure

As discussed above, AWS and Azure are implemented differently and, consequently, require cloud architects to determine the effects that the architectural design will have on application design and deployment. In particular, cloud architects should focus on the following key areas:

SLA adherence: In order to abide by the terms of AWS's SLA for compute, instances must be deployed into two separate AZs within the same region. Azure requires that at least two same-role VMs be deployed in different Fault Domains and different Update Domains within the same Availability Set in order to meet the compute SLA requirements.

Fault tolerance: AWS AZs in the same region are connected by low-latency links, which allow instances to be deployed across AZs without impacting performance. See Figure 7 for a sample AWS architecture. Organizations should consider a multiple-AZ deployment that is very similar to a deployment across two data centers in an active-active configuration. All the requirements for an active-active configuration in a traditional data center space exist, and they will need to be satisfied.

Figure 7. AWS Architecture



EBS = Elastic Block Store

S3 = Simple Storage Service

Source: Gartner (July 2015)

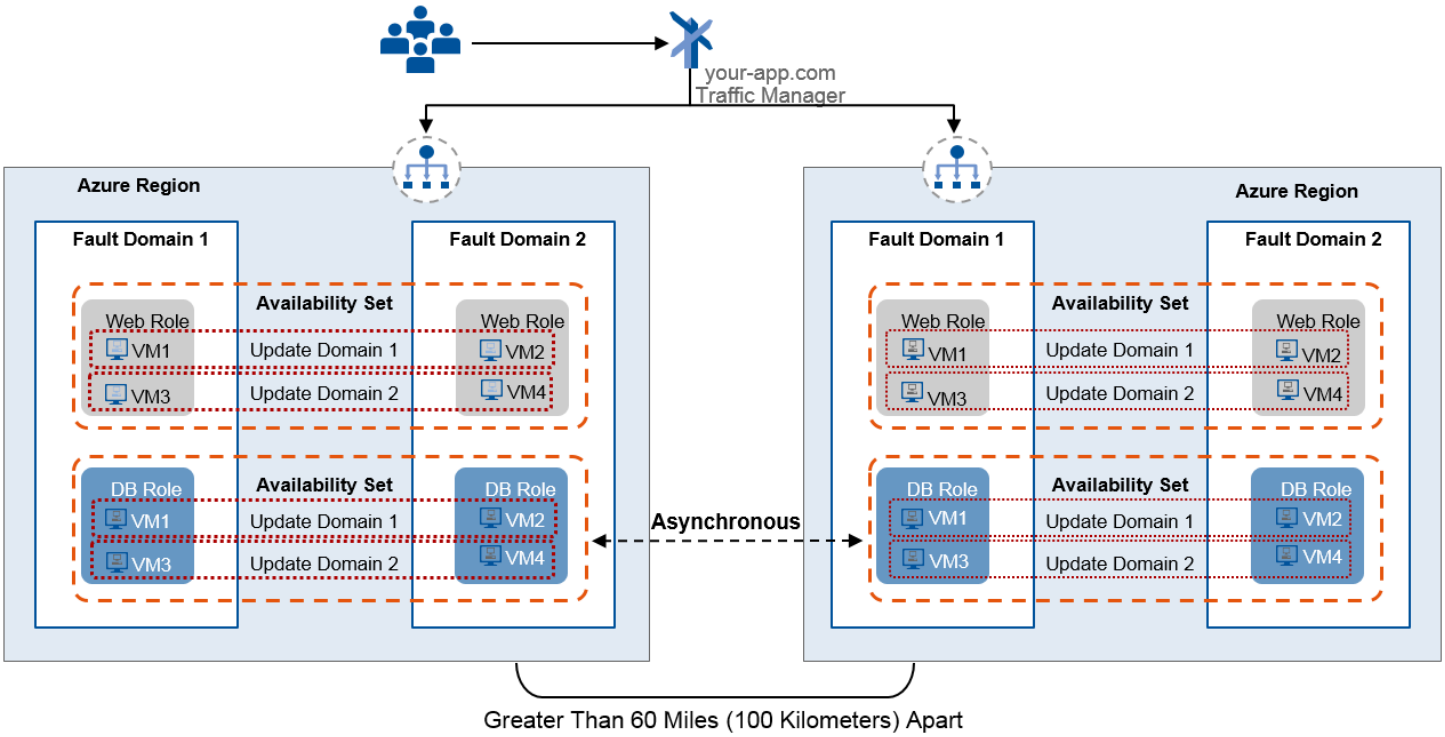
In contrast, Azure does not have regions within synchronous distance, so cloud architects must design application deployments that are highly available within a single logical data center, leveraging concepts like Availability Sets and Fault Domains. When two or more VMs are provisioned in the same Availability Set, Azure's anti-affinity policy ensures that they are automatically placed in different Fault Domains and Update Domains. By separating these VMs across Fault Domains and Update Domains, Azure's design protects against simple hardware failure. However, because the Fault Domains reside in the same logical data center, the application remains vulnerable to a data-center-wide outage. See Figure 6 for an illustration of Availability Sets, Fault Domains and Update Domains.

Difficulty of implementation: Today, most enterprises have a single production data center deployment. AWS's multi-AZ architecture resembles an active-active production data center, which few organizations were ever able to achieve because of the significant capital expenditure and long deployment cycle required. Therefore, deployments on the AWS infrastructure necessitate more careful planning to adapt applications to an active-active-like data center deployment. The advantage with AWS is that an organization can realistically expect to achieve this type of configuration cost-effectively and rapidly.

While Azure's approach is architecturally easier to understand, it is less available because Azure regions are a single logical data center architecture.

Although Azure by design is implemented to support local HA, cloud architects can design and deploy applications across two or more Azure regions (as illustrated in Figure 8), provided that they are aware of the technical limitations, such as the fact that Azure regions are asynchronously connected.

Figure 8. Microsoft Azure Architecture



Source: Gartner (July 2015)

Deciding which availability architecture to use will largely depend on business objectives and technical criteria. On the one hand, AWS makes it very easy to achieve the holy grail of availability by facilitating an active-active configuration, but it does require a learning curve and a lot more planning to achieve. However, once you understand how to leverage AWS's availability, it is quite simple to implement, and the result will be a highly available, highly resilient deployment.

The Azure design, on the other hand, is easier to understand, requires less planning and is more automated — at the expense of availability and resiliency due to the single logical data center limitation.

There is no simple "one way is always better" answer, and your organization may have already selected one provider over the other. Instead, architects should focus on:

What are the organization's current and future requirements?

What are the availability and architectural constraints of workloads being deployed to the cloud?

Answering these two questions will go a long way toward dictating the public cloud platform selection.

Network

Cloud service providers that build IaaS invest heavily in building a robust shared infrastructure for their tenants, and tenants expect to consume that infrastructure in a secure and isolated manner. It is through different implementations of virtual networking that cloud service providers are able to carve out a logical, secure chunk of their IaaS for clients to use. AWS's implementation of virtual networking is called a Virtual Private Cloud (VPC), whereas Azure uses the term Virtual Network. They both use security groups and network access control lists (ACLs) to control access privileges. Significant differences exist in how security is used between AWS and Azure. Because security is out of scope for this document, we recommend that clients who are interested in further investigating the security differences consult the Gartner research "Implementing Effective IaaS Cloud Security in Microsoft Azure" (http://www.gartner.com/document/code/272882?ref=grbody&refval=3093919&latest=true) and "Implementing Effective IaaS Cloud Security in Amazon Web Services." (http://www.gartner.com/document/code/260748?ref=grbody&refval=3093919&latest=true)

Both AWS and Azure offer excellent networking capabilities that allow organizations to build and deploy applications on a local or global level, leveraging networking components such as global server load balancing as well as direct network connectivity that extends on-premises connectivity into the cloud. While both platforms offer similar capabilities, there are technical features and pricing differences that organizations should be aware of. This section focuses on differences in the following areas:

Networking terminology comparison

Native load balancers

Cloud-based DNS

Virtual networking interconnectivity

Dedicated private network connections

Key Take-Aways

AWS's Elastic Load Balancing (ELB) is a more capable service than Azure Load Balancer (ALB) because it offers Layer 4 and 7 capabilities with Secure Sockets Layer (SSL) termination and processing, while ALB is strictly a Layer 4 load-balancing service.

Azure Application Gateway is an HTTP-based application load balancer that offers Layer 7 capabilities like SSL offloading and cookie-based session stickiness. However, unlike AWS ELB, it can only be managed via Azure APIs, and becomes a second component to integrate with ALB and Traffic Manager.

AWS ELB offers application-level cookie affinity but does not provide source Internet Protocol (IP) affinity. ALB offers source IP affinity but does not offer application-level cookie affinity. Azure Application Gateway offers cookie-based affinity, URL hash and weight but does not offer source IP affinity.

AWS's ELB is metrics-driven load balancing, while ALB is not and is limited to round robin load balancing.

AWS's ELB can be monitored using Amazon CloudWatch, whereas Azure's ALB and Application Gateway do not support client monitoring at the time of this writing.

AWS's Route 53 and Azure's Traffic Manager are both very robust cloud-based DNS load managers, except that Route 53 has a geolocation algorithm that is currently not supported with Traffic Manager.

AWS's VPC peering is easier and simpler to configure, requires no additional components to manage, and is free of charge. Azure's VNet to VNet requires configuring of a gateway, has a cost associated with it, and is generally more complicated to configure and maintain.

Azure VNet to VNet offers the ability to leverage the Microsoft network to pass traffic between regions. Similarly, AWS inter-region traffic is also passed over private fiber.

Azure ExpressRoute offers a network service provider (NSP) and an EP/broker service, while AWS Direct Connect only supports the broker model.

Azure ExpressRoute offers a 99.9% SLA, whereas AWS Direct Connect does not publish an SLA.

AWS's Direct Connect offers more partner options and more port speeds than Azure ExpressRoute EP. However, Azure ExpressRoute EP is less expensive than AWS Direct Connect.

Azure ExpressRoute offers redundant ports by default, while AWS charges extra for redundancy.

Networking Terminology Comparison

When comparing technologies, it is important to categorize which features each vendor provides, because more often than not, the wrong technologies are compared when they share similar names.

In Table 5, we detail the AWS and Azure feature terminologies and how they map to industry standard terms.

Table 5. Networking Terminology Comparison

Industry	AWS	Microsoft Azure
Virtual networking	VPC	Virtual Network
Load balancer	Elastic Load Balancing (ELB)	Azure Load Balancer (ALB)/Azure Application Gateway
Cloud-based DNS	Route 53	Traffic Manager
Internetwork connectivity	VPC peering	VNet to VNet
On-premises-to-provider connectivity	Direct Connect	ExpressRoute

Source: Gartner (July 2015)

Native Load Balancers

Both AWS and Azure offer native load-balancing services that are free to use and very useful in many workload deployment scenarios (see Table 6). In addition, both AWS and Azure support many third-party load balancers with more advanced features and capabilities. Native load balancers are intended to support local instances and VMs. We will discuss external traffic load balancing in the next section.

Table 6. AWS and Azure Load Balancers Comparison

	AWS Elastic Load Balancing	Azure Load Balancer	Azure Application Gateway
Layer level	Layers 4 and 7	Layer 4	Layer 7
SSL offloading	Yes	No	Yes
Source IP affinity	No	Yes — two- and three-tuple hash	No
Application layer session stickiness	Duration-based Application-controlled (cookie affinity)	No	Cookie affinity URL hash Weight (load)
Management	AWS Management Console; API; Command Line Interface (CLI)	Azure Management Portal; REST API; PowerShell	REST APIs; PowerShell
Monitoring	Amazon CloudWatch	N/A	N/A
Load-balancing algorithm	Round robin/least-outstanding requests	Five-tuple hash	N/A
Price	$0.025 to $0.32 per ELB/hour $0.008/GB of data processed by ELB (U.S. East pricing)	Free	$0.028 to $0.3584 per gateway/hour $0.008/GB to $0.0035/GB of data processing

(U.S. East pricing)

Source: Gartner (July 2015)

AWS Elastic Load Balancing

AWS's ELB is implemented at the region level, offering load balancing across all the AZs in the region. It offers Layer 4 and Layer 7 network capabilities, specifically SSL offloading, and session affinity (stickiness). Support for SSL offloading is important because without it individual SSL certificates have to be installed and processed on each instance behind the load balancer.

AWS ELB offers two load-balancing algorithms:

Round robin , which ensures that TCP traffic is distributed evenly across nodes based on the performance of each node

Least-outstanding requests , which ensures that HTTP/HTTPS traffic is distributed evenly across nodes based on the performance of each node

AWS offers two session affinity (cookie-based affinity) HTTP load-balancing algorithms:

Duration-based session stickiness: A cookie-based affinity policy with a time to live

Application-controlled session stickiness: A cookie-based affinity policy that does not expire until a new cookie is issued

AWS goes further by integrating ELB with Amazon CloudWatch (AWS's monitoring service), which gives it the advantage of probing for and recovering nonresponsive nodes as well as enabling metrics-based load balancing to distribute load more accurately between instances. For example, if a server node is experiencing heavy utilization (high CPU, memory or disk input/output [I/O]), the load balancer will route traffic to servers that are less busy. Additionally, if an instance exceeds the configured maximum resource utilization, it will be removed from rotation until its metrics reach acceptable levels.

Azure Load Balancer

In Azure, ALBs are also deployed at the Azure region level; however, because Azure's region architecture consists of a single logical data center, ALBs can only deploy VMs in one logical data center. ALB is a Layer 4 load balancer and does not offer any Layer 7 capabilities natively. ALB does integrate with Azure Application Gateway and many third-party vendors to offer more advanced load-balancing features, including Layer 7 capabilities like application-level session stickiness.

Azure load balances TCP- and UDP-based traffic using a five-tuple hash algorithm that distributes load evenly between nodes based on source IP, source port, destination IP, destination port and protocol type. This algorithm is the default ALB load-balancing configuration.

ALB offers two session affinity (source IP affinity) TCP and UDP load-balancing algorithms:

A three-tuple hash algorithm that distributes load evenly between nodes based on source IP, destination IP and protocol type

A two-tuple hash algorithm that distributes load evenly between nodes based on source IP and destination IP

At the time of this writing, cloud architects cannot monitor ALBs or Azure Application Gateway data. However, if Azure ALB or Application Gateway detects a nonresponsive server node, that server is automatically removed from rotation.

Azure Application Gateway

Azure offers another load-balancing service called Azure Application Gateway, which offers HTTP Layer 7 capabilities like SSL offloading and cookie-based session stickiness; however, it is not a free service (for pricing information, see the Application Gateway Web page (http://azure.microsoft.com/en-us/services/application-gateway/)), and management of the service is through Azure APIs only. While Azure Application Gateway integrates with ALB and Traffic Manager, the downside is that it is another component to configure, whereas AWS's ELB is a single component that offers this functionality.

Azure offers three HTTP-based application-level policies:

Cookie affinity — connects a client session to the same back-end VM

URL hash — offloads Web server SSL termination to the application gateway for processing

Weight (load) — enables multiple HTTP requests on the same TCP connection and is routed/load-balanced back to different back-end VMs

It is important to note that AWS ELB offers application-level cookie affinity but does not provide source IP affinity. ALB offers source IP affinity but does not offer application-level cookie affinity. Azure Application Gateway offers cookie-based affinity, URL hash and weight but does not offer source IP affinity.

Cloud-Based DNS

One of the main advantages of deploying an application in the cloud is the ability to make the application available to users in different geographies. This concept, however, is not new: Organizations have been able to do that by distributing the application to different data centers and implementing global server load balancing. However, it was always difficult, time-consuming and expensive. AWS and Azure make it quite simple.

AWS's Route 53 and Azure Traffic Manager are cloud-based DNS, policy-driven services that manage and distribute incoming traffic load. They both provide a high SLA for this service, with Route 53 offering 100% uptime and Traffic Manager offering 99.99% uptime.

Depending on the desired outcome, both services can distribute load based on a set of policies. Table 7 shows the names of the different policies that are available from both AWS and Azure.

Table 7. Cloud-Based DNS Traffic Policies

AWS	Microsoft Azure
Simple (round robin)	Round robin
Weighted (round robin)	Round robin
Latency	Performance
Failover	Failover
Geolocation	Not supported

Source: Gartner (July 2015)

The different DNS load-balancing methods are:

Simple and weighted round robin: These are very simple policies where the load is distributed across all instances or endpoints in a rotating manner. Weighted round robin rotates the load between instances but favors those instances with a higher-priority value. This is helpful in the event that some instances have larger resource configurations, allowing them to handle more load.

Latency/performance: Probably the most popular choice of policy is based on latency. This policy will direct the user to the endpoint with lowest response latency. The load balancer is continuously measuring the query response time and selecting the endpoint with the lowest latency to provide the best user experience.

Failover: In this scenario, the administrator designates primary and secondary application endpoints to control where users connect to in the event of an outage. For example, the administrator can direct all users to the U.S. application endpoint. If the U.S. endpoint fails, traffic is redirected to the European endpoint (and then the Australian endpoint if the European endpoint fails, and so on). This scenario can be used to protect against unexpected failures but can also be leveraged for maintenance purposes.

Geolocation: This is a policy that only AWS Route 53 offers at this time. It allows administrators to limit users of a certain geography to a specific application endpoint while using the same global DNS name. This allows organizations to advertise a single Web address but redirect users to a website that is optimized for their location.

Virtual Networking Interconnectivity

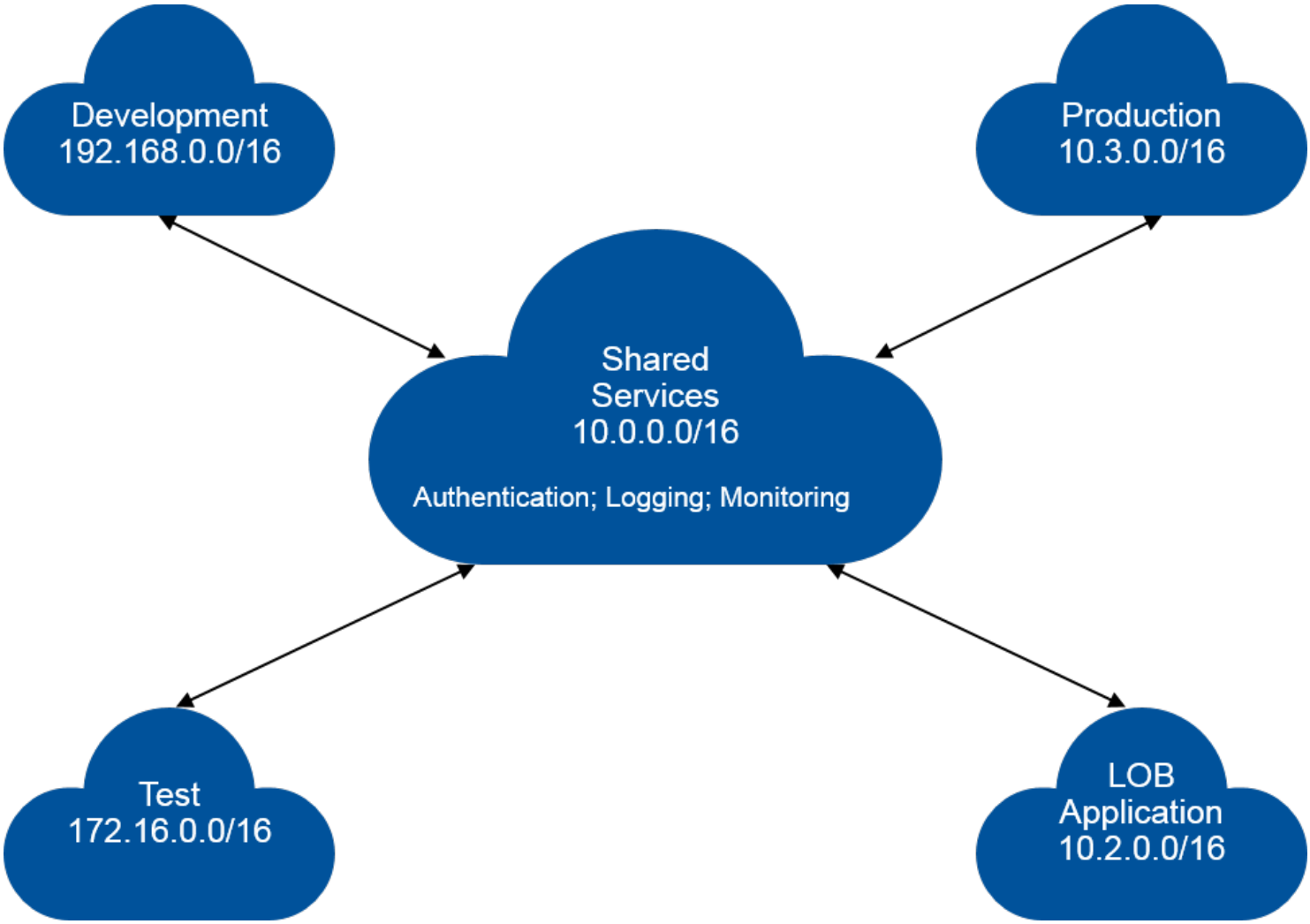
When designing your network on AWS or on Azure, you must consider how you plan to isolate and secure your resources. This exercise will consider network designs in which you:

Deploy all resources into a single VPC or Virtual Network

Deploy resources across multiple VPCs or multiple Virtual Networks (see Figure 9)

There is no best practice that we suggest for all situations; rather, the design you choose will depend on your organization's existing best practices and requirements.

Figure 9. Centralized Shared-Services Network Model



Source: Gartner (July 2015)

In many cases, however, organizations will find that they need to adopt several VPCs or Virtual Networks with different roles for each network. The example in Figure 9 illustrates how this might be used with separate VPCs/VNets for each major application or function as well as a shared VPC with commonly shared services such as:

Authentication

Logging

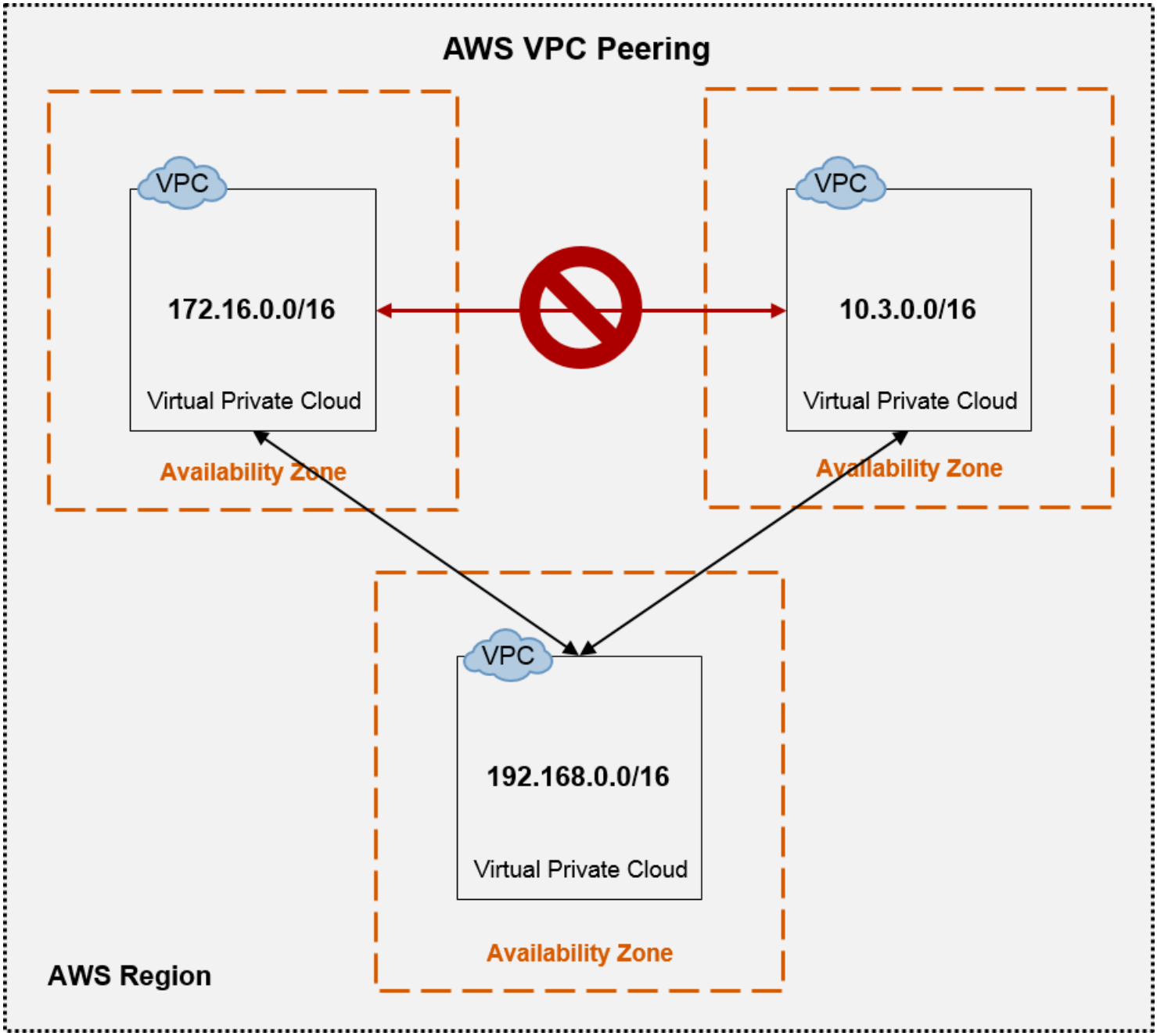
Monitoring

By leveraging this centralized shared-services model, cloud architects can isolate and control these services better while allowing other workloads to take advantage of them. A good example would be development efforts that can be done on a separate network but that are still able to connect and communicate with the shared-services network for all authentication services. The test network can also leverage the centralized shared-services model to test and validate applications against the authentication service that will be used in production. Line of business (LOB) applications that need to be isolated on separate networks can also find a shared-services model useful for many of the same reasons mentioned earlier.

AWS VPC Peering

AWS VPC Peering was designed to connect two VPCs in the same region using private IP addresses and without the need for additional infrastructure, such as a gateway or a VPN. It can be used with the same AWS account or across different accounts and different organizations (see Figure 10). A popular use case of this type of architecture is for organizations to connect and collaborate with other organizations in the cloud. It's a safe haven where neither organization has to expose its data center to the other.

Figure 10. AWS VPC Peering



Source: Gartner (July 2015)

The VPC peering design does not require an additional component such as a VPN gateway; it does not have any single points of failure and relies on the existing VPC infrastructure. Establishing a peering relationship between two VPCs is a relatively simple task that is accomplished from within the AWS Management Console. Both VPCs would have to accept the peering request before it is available, and once connected, VPC routing tables must be updated in order for traffic to flow.

If you plan to have a shared-resources VPC, it can be paired with several other VPCs that you have created for production, test and development or just for other applications.

There are considerations to AWS VPC peering, which are:

- There is no Classless Inter-Domain Routing (CIDR) block overlap, which means you would have to plan your subnet design appropriately if you plan to use VPC peering.
- No transitive-peering relationship means that a peer relationship doesn't allow any pass-through of traffic. For example, a peering relationship between VPC A and VPC B, and another between VPC A and VPC C, cannot be used to pass traffic between VPC B and VPC C using VPC A as a gateway.
- DNS resolution is limited to the local VPC. This is a big downside to using VPC peering. While traffic can route between instances in both VPCs, DNS does not resolve across both; as such, DNS requests will resolve out to the Internet instead.
- VPC peering is not supported across regions. However, connectivity between VPCs in different regions is possible through the use of Internet gateways (IGWs), which create an IPsec VPN tunnel.
- When connecting VPCs across regions using an IGW, AWS routes traffic across the public Internet.
- There is no cost associated with the use of IGWs. This is an advantage that AWS has over Azure because Azure charges per hour for the use of a VPN Gateway. However, Internet data transfer charges apply with both providers.
- Security is enforced using security groups and ACLs. However, it is important to note that security groups do not span regions, and as such, they would have to be configured in both regions.

Unlike a VPN tunnel, AWS VPC peering does not need any component to be maintained. This maintenance-free approach simplifies the implementation of the shared-services model.

Azure VNet to VNet

Microsoft Azure has a similar technology to VPC peering. However, it is architecturally implemented very differently and requires the deployment and configuration of a VPN Gateway between connecting sites. The need for a VPN Gateway introduces an additional component that cloud architects now need to monitor, maintain and scale. Furthermore, VPN Gateways carry certain costs:

- The cost of using the VPN Gateway is shown in Table 8. This cost will apply to all VNet to VNet deployments, whether in the same region or across regions.
- The cost of data transfer, which is free for VNet to VNet in the same region (logical data center), applies when traffic leaves the region. For up-to-date pricing information on this service, consult the VPN Gateway Pricing Web page (http://azure.microsoft.com/en-us/pricing/details/vpn-gateway/) .

Table 8. Azure VPN Gateway Hourly Pricing

VPN Gateway Type	Hourly Price
Static/Dynamic Routing VPN Gateway	$0.036/hour ~$27/month (744 hours, 31-day month)
Standard VPN Gateway	$0.19/hour ~$141/month (744 hours, 31-day month)
High-Performance Gateway	$0.49/hour ~$365/month (744 hours, 31-day month)

Source: Gartner (July 2015)

In addition, VNet to VNet configuration is more complex than creating a VPC peering relationship. VNet to VNet configuration relies on the Azure Management Portal for most of the steps, but it also requires PowerShell scripting to configure the VPN Gateways.

Organizations that are looking to implement VNet to VNet should consider the following:

- The configuration of a VNet to VNet closely resembles that of a site-to-site configuration.
- No overlapping CIDR blocks means you would have to plan your subnet design appropriately if you plan to use VNet to VNet.
- Only dynamic routing VPN Gateways are supported; you cannot configure static VPN Gateways.
- VNet to VNet will work intra-region, across regions and across subscriptions.
- VNet to VNet traffic never traverses the public Internet; rather, it is always carried across the Microsoft network backbone, even when connecting VNets in different geographies (for example, U.S. to Europe). This is a very powerful capability that delivers low-latency, high-performance throughput connections. This is an advantage that Azure has over AWS because AWS would route the traffic between gateways over the public Internet.
- VNet to VNet can be configured with a no-encryption option to lessen the overhead and improve the connection performance between VNets. This can be especially helpful for intra-VNet to VNet connections where encryption may not be required.
- Securing traffic between VMs in different VNets, or between different subnets, is implemented using Azure's Network Security Groups and ACLs.

Because Azure's VNet to VNet implementation requires the deployment of software components such as gateways, it is more complicated to implement the shared-services example.

Dedicated Private Network Connections

Dedicated private connectivity into the AWS and Azure clouds is important as organizations seek to improve application performance, stability and predictability. It is also important for extending on-premises environments into the cloud. AWS Direct Connect offers connectivity through partners (brokers). Azure ExpressRoute, on the other hand, offers connectivity either through an NSP or a partner (broker). There are two options for dedicated private connections:

- **Colocation hub (broker):** Companies offering carrier-neutral colocation hub services maintain direct, high-speed connections to major cloud providers. These direct connections allow customer equipment installed in the colocation facility to have rapid and low-latency access to and from applications running in the cloud. This service is also ideal for clients that want to connect to multiple cloud providers. Because the colocation facility already has established connections to the major cloud providers, enabling these connections for clients is easy. Amazon refers to this service as AWS Partner Network (APN), and Azure refers to it as ExpressRoute Exchange Provider (EP).
- **Network service providers:** NSPs offer direct connections from the client's data center to an Azure region. This service is offered through traditional telco providers such as AT&T, Verizon and others. It essentially establishes another node on the client's WAN. Availability and performance will depend on the location and the target Azure region. AWS does not offer an NSP service, while Azure does, referring to it as ExpressRoute NSP.

Although both AWS Direct Connect and Azure ExpressRoute offer similar direct connection options, there are important differences, which are highlighted in Table 9.

Table 9. AWS Direct Connect and Azure ExpressRoute Comparison Summary

	Direct Connect APN	ExpressRoute NSP	ExpressRoute EP

Availability	All regions	Available in all zones	Available in all zones
Included egress traffic	No included egress traffic offered	Unlimited egress traffic	3TB to 90TB
Egress traffic cost per GB	$0.02 to $0.11 (depending on location)	Unlimited egress traffic	$0.025 to $0.14 per GB (depending on location) beyond the included 3TB to 90TB
Partners	61	11	11
Redundancy by default	No	Yes	Yes
SLA	Not published	99.9%	99.9%

Source: Gartner (July 2015)

REDUNDANCY BY DEFAULT

An important design consideration to remember is that, by default, Azure ExpressRoute includes two ports on two routers for redundancy at no extra cost. AWS Direct Connect does not offer port redundancy by default, although AWS strongly recommends it and charges extra for it. The cost of a redundant port is the same as the cost of a single port multiplied by two.

COST COMPARISON

The cost difference between AWS Direct Connect APN and Azure ExpressRoute EP is quite significant when you factor in port redundancy. AWS Direct Connect APN is less expensive without redundancy, while Azure ExpressRoute EP is less expensive with port redundancy. In addition, AWS Direct Connect APN has more port speed options than Azure ExpressRoute EP. Organizations should determine which port speeds are needed and whether port redundancy is required because those will factor into the cost calculation.

Colocation Hub (Broker) Cost Comparison

If organizations choose to go with a broker to provide dedicated network connectivity, then AWS Direct Connect APN and Azure ExpressRoute EP offer the port speeds listed in Table 10.

Table 10. Direct Connect and ExpressRoute Broker Port Speed Comparison

Port Speed	50 Mbps	100 Mbps	200 Mbps	300 Mbps	400 Mbps	500 Mbps	1 Gbps	10 Gbps
Direct Connect APN/hour	$0.03	$0.06	$0.12	$0.18	$0.24	$0.30	N/A	N/A
Direct Connect APN/month	$22.32	$44.64	$89.28	$133.92	$178.56	$223.30	N/A	N/A
ExpressRoute EP/month	N/A	N/A	$145	N/A	N/A	$290	$436	$5,000

Source: Gartner (July 2015)

From a financial perspective, the calculations are as follows:

AWS monthly cost for 500 Mbps port speed = $223.20 ($0.30 per hour x 744 hours per month)

Azure monthly cost for 500 Mbps port speed = $290

Azure includes 3TB (3,000GB) of egress traffic for Zone 1

AWS cost of 3TB of egress traffic = $60 ($0.02/GB x 3,000GB)

AWS monthly cost, including 3TB of egress traffic = $283.20 ($223.20 + $60)

NSP Cost Comparison

Alternatively, Azure offers a second service for dedicated private connectivity known as ExpressRoute NSP. As shown in Table 11, the NSP service offers fewer port speed options but is significantly more expensive than both AWS Direct Connect APN and Azure ExpressRoute EP.

Table 11. ExpressRoute NSP Port Speed Cost Comparison

Port Speed	10 Mbps	50 Mbps	100 Mbps	500 Mbps	1 Gbps
ExpressRoute NSP/month	$436	$872	$1,300	$5,200	$8,700

Source: Gartner (July 2015)

The NSP option does have some advantages: It does not require clients to maintain equipment at a colocation facility, and it offers unlimited egress traffic. However, for clients to truly take advantage of the unlimited egress traffic that this option offers, it is important to identify how much egress traffic an organization expects to use and then run a cost calculation to see if it is financially viable. Below is a cost comparison between AWS Direct Connect APN and Azure ExpressRoute NSP. Furthermore, we compare the cost of Azure ExpressRoute EP and Azure ExpressRoute NSP

Azure ExpressRoute NSP Versus AWS Direct Connect APN

In this comparison, organizations would have to consume egress traffic upward of 42.3TB without AWS Direct Connect port redundancy and 41.2TB with port redundancy before the unlimited offer from Azure begins to make financial sense — any less, and AWS is cheaper. To arrive at the 42.3TB without port redundancy and 41.2TB with port redundancy delta between AWS and Azure, clients must consider the following formula:

AWS monthly cost for 1 Gbps port speed = $223.20 ($0.30 per hour x 744 hours per 31-day month); for a redundant port, the price becomes $223.20 x 2 = $446.40

Azure monthly cost for 1 Gbps port speed = $8,700 (includes a redundant port)

Break-even point between AWS (without port redundancy) and Azure = 42.3TB ($8,700 Azure monthly cost; $223.20 AWS monthly cost/$0.02 AWS data transfer per GB)

Break-even point between AWS (with port redundancy) and Azure = 41.2TB ($8,700 Azure monthly cost; $446.40 AWS monthly cost/$0.02 AWS data transfer per GB)

One thing to note here is that the AWS data transfer rate varies by location. Cloud architects are encouraged to always consult AWS's documentation (http://aws.amazon.com/directconnect/pricing/) for up-to-date pricing.

Azure ExpressRoute NSP Versus Azure ExpressRoute EP

When comparing the cost of 1 Gbps between Azure ExpressRoute NSP and Azure ExpressRoute EP, organizations would have to consume egress traffic upward of 33.05TB before the unlimited offer from Azure ExpressRoute NSP begins to make financial sense — any less, and Azure ExpressRoute EP is less expensive. To arrive at the 33.05TB, clients must consider the following formula:

Azure ExpressRoute EP monthly cost for 1 Gbps port speed = $436 (includes a redundant port)

Azure ExpressRoute NSP monthly cost for 1 Gbps port speed = $8,700 (includes a redundant port)

Azure ExpressRoute EP includes 3TB (3,000GB) of egress traffic for Zone 1

Break-even point between Azure ExpressRoute EP and NSP = 33.05TB ($8,700 Azure monthly cost; $436 monthly cost Azure ExpressRoute EP with 3TB of free egress traffic/$0.025 Azure ExpressRoute data transfer per GB)

Azure ExpressRoute data transfer rates vary by location. Cloud architects are encouraged to always consult Azure's documentation (http://azure.microsoft.com/en-us/pricing/details/expressroute/) for up-to-date pricing.

Another consideration to take into account when comparing NSP to the broker model is that clients still have to account for the cost of connecting from the enterprise data center to the colocation hub, which will increase the cost of the colocation hub when compared with NSP. These pricing exercises illustrate the level of analysis that an organization would need in order to calculate an accurate total cost of ownership and ROI for a cloud initiative.

Compute

All the components of IaaS are designed to work together for one purpose — and that is to provide a robust platform for hosting instances/VMs. This section examines the following issues:

Compute terminology comparison

Interoperability with on-premises hypervisors

Concurrent provisioning

Autoscaling of workloads

Container strategy

Optimizing compute cost

Key Take-Aways

Azure has a stronger story and interoperability capabilities with on-premises environments that are using Microsoft technologies. AWS offers tools to help manage and migrate on-premises and cloud assets.

AWS Auto Scaling can provision/terminate instances on demand based on configured policies, whereas Azure Autoscale requires that architects preprovision VMs to be turned on or off when needed.

AWS and Azure both support instance-level/VM-level integration with containers, except that AWS also offers a managed container service (EC2 Container Service [ECS]) with advanced features and functionality.

AWS Reserved Instances are self-service discount plans that clients can subscribe to immediately, whereas Microsoft Enterprise Agreements are complex and time-consuming.

AWS Elastic Compute Cloud (EC2) Instances are billed by the hour, while Azure VMs are billed by the minute.

AWS Auto Scaling can also replace unhealthy instances on behalf of customers if instances become unhealthy or unreachable.

Compute Terminology Comparison

The lack of standard terminology among cloud providers complicates the comparison of their features. In Table 12, a side-by-side comparison of terminology between AWS and Azure is provided to facilitate the comparison process.

Table 12. Compute Terminology Comparison

Industry	AWS	Azure
VM/server	EC2 Instance	Virtual machine
VM file format	Amazon Machine Image (AMI)	Virtual hard disk (VHD)
Scale up/down (elasticity)	Auto Scaling	Autoscale

Managed-container service	ECS	N/A

Source: Gartner (July 2015)

Interoperability With On-Premises Hypervisors

Organizations that are currently using Microsoft Hyper-V on-premises may see advantages to adopting Azure because of the compatibility between the products. In addition, Microsoft offers Azure Stack (http://www.microsoft.com/en-us/server-cloud/products/azure-in-your-datacenter/) for a 100%-compatible on-premises deployment of Azure as a private cloud solution. The commonality between Hyper-V, Azure and Azure Stack provides a number of potential benefits:

- Simple migration of VMs and their data between public and private infrastructures due to a standard VHD format
- New disaster recovery options between on-premises infrastructure and Azure
- Common management tools and provisioning portals between on-premises infrastructure and Azure

Conversely, AWS does not offer a hypervisor or private cloud platform for on-premises deployment. As a result, on-premises deployment is more loosely integrated with the public cloud. Amazon partially addresses the problem by offering a Microsoft System Center plug-in to facilitate the conversion of Hyper-V VMs into Amazon's format. The AWS plug-in for Microsoft System Center also allows customers to manage on-premises and cloud assets from a single pane of glass. AWS offers similar capabilities for VMware vCenter and Citrix XenServer.

However, considering that VMware vSphere still captures the lion's share of the server virtualization market, and considering that neither AWS nor Azure can run vSphere VMs natively, the playing field is leveled between both providers. It is only in Azure's favor if the on-premises deployment is based on Hyper-V.

Concurrent Provisioning

The ability to deploy multiple, concurrent VMs is an important criterion when comparing the capabilities of AWS and Azure because one of the popular value propositions of using an IaaS is speed of provisioning. AWS allows organizations to deploy multiple, simultaneous instances by populating a field in the AWS Management Console; alternatively, an API flag can be used to accomplish the task.

Azure offers a self-service templating service for automating the deployment of multiple concurrent stacks of infrastructure using the Azure Resource Manager. Customers can deploy a template and its associated collection of resources (called a stack) by using the Azure PowerShell, Azure CLI, REST API or Microsoft Azure Preview Portal.

Autoscaling of Workloads

The ability to automatically scale up or down the number of VMs to meet the load requirements of an application is a very intriguing value proposition, especially during peak times or during the rollout of a new application with unpredictable load expectations. Both AWS and Azure have their own implementations of elasticity. AWS has Auto Scaling, and Azure has Autoscale; however, these implementations are quite different.

AWS AUTO SCALING

AWS Auto Scaling is a dynamic scale-up and scale-down service that allows organizations to provision EC2 Instances on demand based on utilization metrics and thresholds, or based on a schedule. Auto Scaling integrates with Amazon CloudWatch and allows architects to configure triggers using any of the available metrics. For example, Auto Scaling can be configured to automatically provision and deploy additional instances when the memory utilization exceeds a certain threshold for a sustained period of time. By the same token, if the memory utilization drops below the configured threshold, Auto Scaling will power off or terminate instances.

AWS Auto Scaling can also replace instances on behalf of customers if the instances become unhealthy or unreachable. Cloud architects can configure an Amazon CloudWatch alarm to monitor EC2 Instances and automatically recover instances should they become impaired due to a problem that requires Amazon intervention to repair. It is important to note the following when considering instance replacement:

AWS Auto Scaling automatically deploys a replacement instance without any customer intervention.

Recovered instances retain the original instance ID, private IP, Elastic IP and instance metadata.

Original instances configured with a public IP address will receive a new public IP address when the instance is recovered. To receive the same public IP address, instances must be configured with an Elastic IP.

Instance recovery is only possible on EC2 Instances that use EBS storage.

These are significant because they highlight the availability and resilience capabilities of AWS. Considering that an AWS Auto Scaling group spans an entire region with multiple AZs, this technology protects against an entire AZ failure. If an AZ should fail, the instances deployed in that AZ are automatically replaced with other instances in different AZs.

Other examples of problems that could cause AWS Auto Scaling to flag an instance as unhealthy include:

Loss of network connectivity

Loss of system power

Software issues on the physical host

Hardware issues on the physical host

AZURE AUTOSCALE

Azure's implementation of elasticity is less flexible. Autoscale cannot automatically provision new VMs based on a resource trigger or schedule. Instead, architects must preprovision VMs so that Autoscale can power them on or off as needed based on a schedule, CPU metric or queue depth metric trigger only.

There are two key differences between AWS Auto Scaling and Azure Autoscale:

The number of VMs available to support the workload is preprovisioned in Azure, whereas AWS can keep provisioning on-demand instances to meet demand. The Azure approach forces the architect to make an educated guess as to what the peak workload requirements will be.

AWS Auto Scaling spans an entire AWS region. This regionwide implementation means that an AWS Auto Scaling group can span multiple AZs and deploy or remove instances as needed to support the application. Azure's Autoscale, on the other hand, is confined to a single region (logical data center) and is useful for an application that is limited to that data center.

It is important to note that the elasticity features we are discussing are limited to IaaS. Azure and AWS have more advanced capabilities in their platform as a service (PaaS) offerings that developers can take advantage of as they build new applications.

Containers Strategy

Containers are a promising technology that is quickly becoming a favorite among developers for new applications. Both AWS and Azure have support for containers, although AWS has a clear advantage in this space.

AWS and Azure both have extensive support for Docker extensions in their Linux and Windows instances and VMs. However, this is where the similarities end. Microsoft's strategy, at this point in time, is to leverage the ecosystem of partners to augment its container offering, and the company has not indicated an interest in building a platform or a managed container service.

AWS also leverages an ecosystem of partners to augment its container offering. However, AWS also provides the following additional container services:

EC2 Container Service (ECS) is a fully managed, API-enabled container platform that can integrate with other AWS services to facilitate the deployment, management, scalability, performance and security of a container environment.

Elastic Beanstalk for Docker allows developers to quickly provision an ideal production platform for their Docker applications. Elastic Beanstalk automatically provisions the required AWS services, such as EC2 Instances, load balancing, autoscaling and integration with ECS for cluster creation.

As the adoption of containers grows, so does the pain of managing the infrastructure that supports them. Container adoption faces similar challenges to those that VMs faced in the early days, such as scalability, cluster management, availability, configuration management, security, scheduling and container sprawl. AWS's ECS goes some way to mitigating these problems in the following areas:

Cluster management: Advanced cluster management in ECS abstracts infrastructure functions related to availability, scalability and security.

Scheduling: Scheduling capabilities that are available in ECS give clients the ability to use a variety of scheduling services, including the ability to write a custom scheduler.

Isolation: ECS enforces isolation by not allowing two customers to launch and run containers on the same EC2 Instance. ECS requires a VPC for enhanced isolation, networking connectivity and other VPC services.

Availability: ECS is a regionwide service, which means that it can take advantage of multi-AZ deployments.

Although AWS has an advantage in container support at the moment, Microsoft is clearly committed to delivering a solid container strategy. This is evident by Microsoft's support for containers and Docker on the Windows Server 2016 platform. Given that level of commitment, it would not be a surprise to see Azure introduce a container-managed service in the future.

Optimizing Compute Cost

Organizations that are looking to understand the financial implications of using public cloud IaaS, or are trying to compare the compute cost of deploying on IaaS versus deploying traditionally on-premises, should invest the time to investigate the provider's discount programs. While both providers offer standard hourly pricing for VMs that is very similar, leveraging the discount programs will yield higher savings (especially compared with on-premises) and will significantly impact cost comparisons. This section highlights the differences between:

AWS Reserved Instances

Microsoft Enterprise Agreements

AWS RESERVED INSTANCES

AWS Reserved Instances are designed to lower a client's EC2 Instance hourly rate in exchange for a commitment over a one- or three-year term and an upfront fee. Here are the program highlights:

Organizations can select a one-year or a three-year term. While longer terms offer a deeper discount, organizations should keep in mind that cloud pricing is always changing. For example, AWS has dropped prices over 40 times since 2008. Therefore, locking in a longer term could be inefficient as prices drop over the course of the commitment. One-year terms are strongly advised, while three-year terms should be carefully weighed.

Some Reserved Instance terms are coupled with an upfront fee to calculate the level of hourly discount that a client will receive. Paying no upfront fee will still give you a discounted hourly rate over on demand, but paying a partial or full upfront fee will offer clients a higher discounted hourly rate.

Once committed to a Reserved Instance term, you will be billed for every hour, whether your instance is powered on or off.

Reserved Instance types can be modified within the same family, provided that the capacity is available to be reserved and the size is maintained. For example, architects can take the resources of an M3.Large instance and divide it into two M3.Medium instances. Similarly, the resources of four M3.Small instances can be combined to create a single M3.Large instance.

Reserved Instances reserve physical capacity in an AWS AZ, so the resources that are required for the instance to power on and run are reserved. AWS does not oversubscribe capacity reservations.

Discounted rates vary between Windows and Linux instances.

Reserved Instances are purchased on an instance-by-instance basis.

Reserved Instances are transferable between AZs in the same region.

Organizations that want to get out of a Reserved Instance term before its maturity can sell their instances on the Reserved Instance Marketplace.

Because clients are effectively paying for all the hours in a Reserved Instance term, powering off instances will not save money. However, it is possible to transfer the balance of a Reserved Instance term from one instance to another. This is useful in cases where an instance needs to be terminated, rebuilt or simply repurposed to a role that does not require a Reserved Instance. In essence, the Reserved Instance is a fixed cost that is already incurred, but the instances that use the Reserved Instance can be changed.

The Reserved Instance pricing essentially renders the calculation of the instance hourly rate a meaningless exercise. Clients are instead advised to examine how much they are spending monthly for a Reserved Instance, instead of hourly. Furthermore, clients are advised to pay the full upfront fee when possible because that will yield the highest discount level over the term of the Reserved Instance.

Table 13 shows the monthly price difference between on-demand pricing and one- and three-year Reserved Instances for an M3.Large EC2 Instance:

Table 13. Reserved Instance Sample Pricing — M3.Large

Upfront Payment Type	Upfront Payment Amount	Discounted Hourly Rate	On-Demand Hourly Rate	Savings
M3.Large — One-Year Term				
No upfront	$0	$0.095	$0.133	29%
Partial	$421	$0.083	$0.133	38%
Full	$713	$0.081	$0.133	39%
M3.Large — Three-Year Term				
Partial	$672	$0.056	$0.133	58%
Full	$1,373	$0.52	$0.133	61%

Source: Adapted from AWS

MICROSOFT ENTERPRISE AGREEMENTS

Azure does not have a self-service discount program that is comparable to AWS Reserved Instances. Instead, Microsoft offers a discount for Azure through Enterprise Agreements. While Enterprise Agreements could provide a significant discount to organizations, the process for clients who do not already have an EA is lengthy and complicated. For clients who already have an EA agreement, adding Azure is relatively easy. Microsoft Enterprise Agreements do not require an upfront Azure commitment spend.

Azure previously had a discount program known as Azure commitment plans, which allowed clients to estimate the monthly spend with Azure and receive a discount based on the spend tier they chose to commit to. However, as of August 2014, Azure has discontinued commitment plans.

Storage

Storage is a foundational service in cloud IaaS whether a customer is using compute or networking. Therefore, clients must understand the different storage offerings that AWS and Azure provide in order to align the right service with the right workload. The following topics are examined in this section:

Storage terminology comparison

Object-based storage

Block storage

Local host storage

Network-shared storage

Data sovereignty

Data encryption

Key Take-Aways

AWS and Azure both have excellent storage offerings that are quite similar in features and capabilities.

AWS has an advantage concerning block-level, high-performance storage in that it allows architects to choose the input/output operations per second (IOPS) level that they wish to use for a particular workload. Azure's implementation has more predefined ranges.

Azure supports expandable block storage volumes to increase the size of a volume, while AWS does not. Instead, AWS offers a methodology on how to migrate to a larger volume size.

AWS offers instance store (temporary local host) storage, which can be used in production for certain workloads. Azure's implementation of temporary disks should not be used for any production workloads per Microsoft best practices.

AWS and Azure both offer managed network-shared storage, with Azure having an edge in protocol support, while AWS has an edge in scalability and performance.

AWS has an edge with data encryption services that offer several server-side and client-side encryption options, whereas Azure is limited to client-side encryption.

Both providers offer key management services, with the possibility to use a third-party key management service if a client chooses.

Storage Terminology Comparison

Once again, terminology used for similar features and capabilities varies between the vendors. Table 14 translates AWS and Azure terminology to its appropriate industry equivalent:

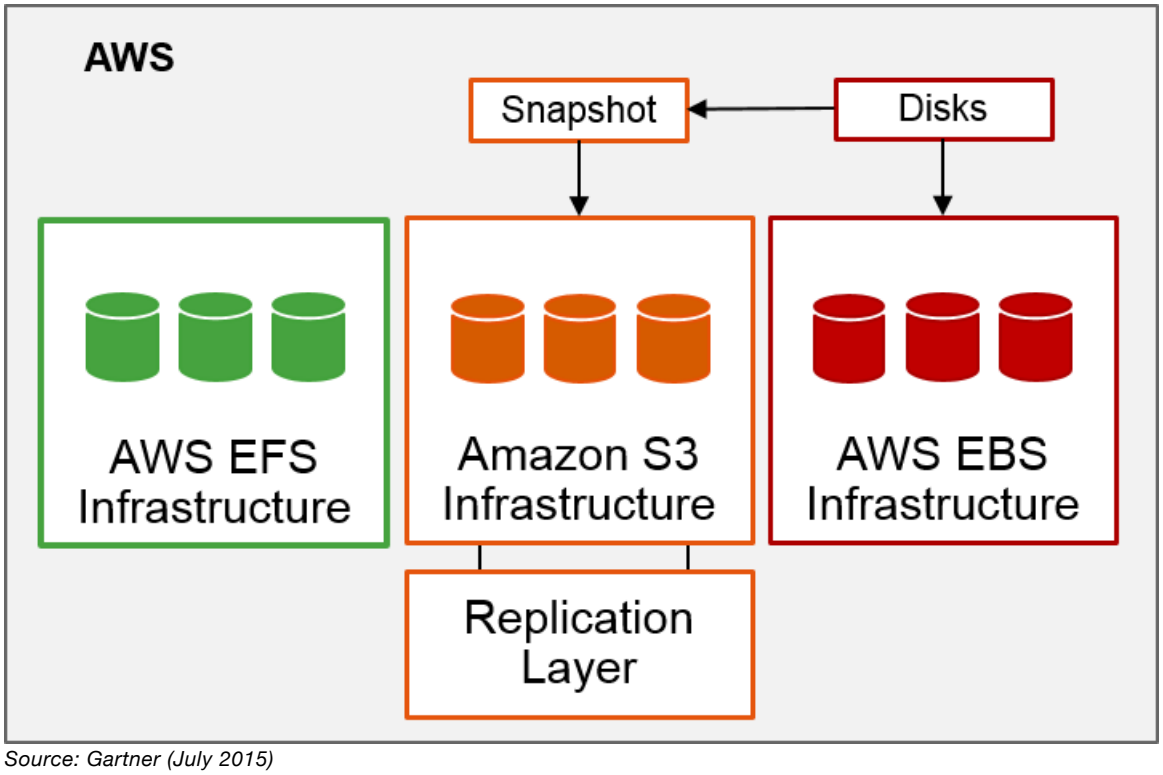
Table 14. Storage Terminology Comparison

Industry	AWS	Azure
Object-based storage	Simple Storage Service (S3)	Block Blob
Folder/directory	Amazon S3 Bucket	(Blob) container
Shared block storage	Elastic Block Store (EBS)	Page Blob/Disks
Guaranteed IOPS	Provisioned IOPS	Premium Storage
Local host storage	Instance store (ephemeral)	Azure temporary disks
Network shared storage	Elastic File System (EFS)	Azure Files

Source: Gartner (July 2015)

AWS and Azure have several different types of storage infrastructure that can be used to satisfy various technical requirements. Figure 11 illustrates the storage architectures of AWS.

Figure 11. AWS Storage Architecture



Source: Gartner (July 2015)

AWS has the following storage services:

Amazon S3 is highly durable, highly scalable, object-based-storage. This type of file system is suitable for unstructured file types like images, videos and documents. Amazon S3 is also suitable for use cases like content delivery, disaster recovery, backup and archiving, and big data analytics (see the Object-Based Storage section for more details).

Amazon S3 Reduced Redundancy Storage (RRS) is similar to Amazon S3 except that it is considered less durable than the standard S3 because objects are replicated to fewer disks and locations (see the Object-Based Storage section for more details).

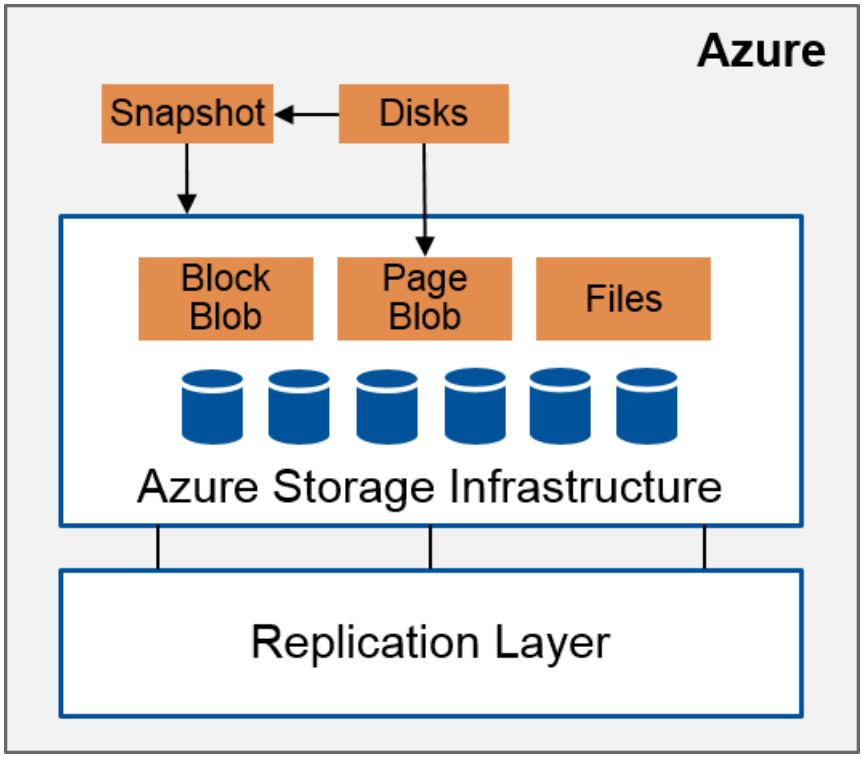
EBS is block-level storage that is used with EC2 Instances.

Instance store , also known as ephemeral storage, is nonpersistent storage that is available with EC2 Instances that are provisioned from an AMI that is configured with an instance store.

EFS is AWS's network-shared storage, which is based on the NFS v4.0 standard and provides a fully managed file system. With a fully managed file system service, clients don't have to worry about scaling, maintaining or monitoring the file system.

Figure 12 illustrates the storage architecture of Azure:

Figure 12. Azure Storage Architecture



Source: Gartner (July 2015)

As shown in Figure 12, Azure has the following storage types available:

Block Blob is highly durable, highly available, and scalable object-based storage. This type of blob is suitable for unstructured file types such as images, videos and documents. Block Blobs are also suitable for use cases like content delivery, disaster recovery, backup and archiving, and big data analytics (see the Object-Based Storage section for more details).

Page Blob is Azure block-level shared storage for VMs. This type of blob is suitable for random read/write type files such as VHD.

Files is Azure's network-shared storage, which is based on SMB 2.1 and is a fully managed file system. With a fully managed file system service, clients don't have to worry about scaling, maintaining or monitoring the file system.

Object-Based Storage

Any data stored in Amazon S3 is stored as an object. Groups of objects are stored in buckets. The Azure equivalents are called Block Blobs and containers. Buckets and containers serve the same organizational function of a folder or a directory. It is worth noting, however, that both S3 and Block Blobs do not support folders and directories beyond a container or a bucket. In order to organize files, architects use a prefix (which looks and feels like a directory or folder when creating it). While a prefix does not share the same characteristics as a folder or directory on the file system, it provides the organizational capabilities of a directory or folder.

As far as the maximum capacity that can be stored in buckets and containers, see Table 15.

Table 15. Platform Maximums

	AWS	Azure
S3 bucket size/Azure Blob container size	Unlimited	Azure's maximum container size is based on the standard storage account maximum of 500TB. Clients can have up to 100 storage accounts per subscription.
Maximum object size	5TB	200GB

Source: Gartner (July 2015)

This document focuses on differences around data replication because it is a topic that clients frequently inquire about. For a more complete and in-depth look, Gartner recommends that clients read "Public Cloud Object Storage Comparison: Amazon, Google and Microsoft." (http://www.gartner.com/document/2778717?ref=QuickSearch&sthkw=Cloud Object Storage Comparison%3A Amazon%2C Google and Microsoft&refval=150961932&qid=7aa31b2674b6b5897c5a444c3948ed8a)

AWS REPLICATION CAPABILITIES

AWS S3 is limited to a single region by default. Within a region, an object is synchronously replicated across facilities, and each facility stores multiple copies of the object to ensure data durability. AWS also calculates checksums on all network traffic to detect corruption of data packets when storing or retrieving data. Amazon S3 is designed to withstand data loss across one facility.

AWS's S3 RRS service operates on the same principles, but it synchronously replicates the object across fewer facilities and stores fewer copies per AZ. Amazon S3 RRS is designed to withstand data loss across one facility. This type of reduced architecture is designed to maintain the availability of the data but with the relaxed durability standards that make it appropriate for reproducible data.

Organizations that need to replicate and store data across AWS regions that are hundreds of miles apart for regulatory reasons, or for added data protection, can use Cross-Region Replication (CRR). Cloud architects can manually choose which AWS region to replicate the data to during the configuration of CRR.

MICROSOFT AZURE REPLICATIONS CAPABILITIES

Azure has many options for protecting data at both the local level and the regional level, offering the following:

Locally Redundant Storage (LRS): The LRS service stores an object within a single facility of a single region. Within this facility, three separate copies are synchronously written to separate Fault and Update Domains to protect against hardware failures and software updates.

Zone Redundant Storage (ZRS): The ZRS service stores the "equivalent" of three object copies across two or three facilities. These facilities may be in the same region, but they may also be in two separate regions.

Geographically Redundant Storage (GRS): The GRS service synchronously writes three copies of an object to a single facility in a region. In addition, GRS asynchronously copies the object to a facility in a separate, secondary region, where another three object copies are stored.

Read Access (RA)-GRS: The RA-GRS service is an option for the GRS service that stores objects using the same methodology as the GRS service; however, the copies in the secondary region are available for read requests. Microsoft may fail over customers to the secondary region in some circumstances.

DATA DURABILITY EXPECTATION AND AVAILABILITY SLAS

Amazon does not publish durability SLAs for its object-based storage service. However, Amazon does publish a durability expectation for both S3 and S3 RRS (as shown in Table 16). Microsoft does not publish a durability expectation, but given the replication methods discussed earlier, it is safe to assume that the durability expectation is very high.

Table 16. Data Durability and Availability

	Amazon S3	Amazon S3 RRS	Microsoft Azure
Durability expectation	99.999999999%	99.99%	N/A
Availability SLA	99.9%	99.9%	99.9%

Source: Gartner (July 2015)

VERSIONING AND SNAPSHOTS

While both AWS and Azure offer excellent data durability and availability capabilities, cloud architects may also enable object versioning in the Amazon S3 and S3 RRS services to mitigate against data loss from hardware failure or accidental deletion.

Azure does not support versioning, but it does support snapshots. Consequently, cloud architects working on the Azure platform may implement object snapshots for additional data protection. However, snapshots do not protect against loss of the original object due to events such as hardware failure or data corruption. Like versioning, snapshots can be used to restore objects to a previous state, but unlike versioning, they store only the changed blocks of an object rather than a whole object.

Clients should be aware that they would not be entitled to any service credits or recompense for data loss. This is important, because if clients wish to protect even more against data loss, they would have to consider alternative methods, like traditional backup. However, consider the cost of deploying and managing an additional backup environment and whether or not it is justified and cost-effective to protect against such a small percentage of data loss. It is further worth noting that, with traditional backup, data durability expectation is much lower since it is copied fewer times, and availability SLAs do not exist.

Block Storage

While object-based storage is great for unstructured data like images, videos or documents, it is not ideal for instances and VMs. To address the need for nonobject storage, AWS and Azure offer traditional block storage that is ideal for instances and VMs:

AWS EBS is a separate storage infrastructure that does not sit on top of S3; it is designed to provide storage volumes for AWS EC2 Instances.

Azure Page Blobs reside on top of the same shared-storage infrastructure that supports all the other Azure storage services like Block Blobs, Files, Tables and Queues. Page Blobs are used to provision block storage devices, such as disks, for VMs.

EBS CHARACTERISTICS

EBS volumes are confined to a specific AZ. However, architects can take a snapshot and restore it in a different AZ or region, thereby making it available to instances within that AZ. To protect and replicate EBS data volumes, cloud architects must create point-in-time snapshots of an EBS volume, which is automatically stored in Amazon S3. Amazon S3 then replicates data to multiple AZs within a region.

Organizations that need to replicate and store data across multiple AWS regions that are hundreds of miles apart for regulatory reasons, or for added data protection, can use CRR. Cloud architects can manually choose which AWS region to replicate the data to during the configuration of CRR.

EBS has three tiers of storage that can be provisioned with all instances:

General-purpose SSD is backed by all solid-state drives (SSDs) and is the default option when provisioning EC2 Instances.

General-purpose magnetic is backed by conventional disk drives using magnetic recording media.

AWS EBS Provisioned IOPS is backed by all SSDs (see the Performance Targets section below).

PAGE BLOB CHARACTERISTICS

Azure Page Blobs offer two storage tiers, which are:

Standard Page Blobs are backed by conventional disk drives using magnetic recording media.

Premium Storage is backed by all SSDs (see the Performance Targets section below).

Azure Page Blobs have the following characteristics:

Azure Page Blobs are limited to an Azure region (data center) and enjoy the same replication capabilities previously described for Block Blobs.

Azure Page Blobs are stored on conventional magnetic disks. Table 17 illustrates the differences between EBS and Page Blobs.

Azure does offer all-SSD local disks with its G- and D-series VMs, but they are only available as a preconfigured feature of the G- and D-series VMs (see the Performance Targets section for more details).

Table 17. EBS and Page Blobs Characteristics

	EBS General-Purpose (Magnetic) Disk	Azure Page Blob/Disks
Volume/disk size maximum	1TB can be striped for higher capacity	1TB can be striped for higher capacity
IOPS	~100 IOPS, with the ability to burst to hundreds of IOPS	Up to 500 IOPS
Expandable block storage volumes	No	Yes
Price (based on U.S.-East location)	$0.05 per GB per month of provisioned storage $0.05 per 1 million I/O requests	LRS: $0.045 to $0.05/GB GRS: $0.06 to $0.095/GB RA-GRS: $0.075 to $0.12/GB

Source: Gartner (July 2015)

EXPANDABLE BLOCK STORAGE VOLUMES

A common task that administrators are faced with is the expansion of a volume to increase disk space. AWS and Azure approach this in different ways.

Azure storage supports expandable volumes using a two-step process. First, the customer must expand the Page Blob. Second, the customer must expand the VHD itself. This process requires that the VM be powered off at the time of expansion.

Conversely, AWS does not support increasing the size of an EBS volume on the fly. However, AWS provides a methodology to expand the storage space of an Amazon EBS volume by migrating data to a larger volume and then extending the file system on the OS to recognize the newly available space. After verifying that the new volume is working properly, the old volume may be deleted. This methodology is implemented without powering off the instance.

PERFORMANCE TARGETS

Clients that are considering deploying applications with demanding storage performance requirements should be aware of the differences in high-performance storage offerings for AWS and Azure. Table 18 summarizes the key differences between AWS's and Azure's offering.

Table 18. AWS-Provisioned IOPS and Azure Premium Storage

Feature	AWS EBS Provisioned IOPS	EBS General-Purpose SSD	Azure Premium Storage
Minimum size	4GB	4GB	3.5GB
Maximum size	16TB	16TB	1TB (up to 32TB with striping)
Striping support	Yes	Yes	Yes
Maximum disks that can be striped	Multiple; exact number not published	Multiple; exact number not published	32 disks (up to 64 disks with the G-series VM)
Maximum IOPS single volume	20,000	10.000	5,000
Maximum IOPS/instance and VM	48,000	48,000	50,000 (up to 64,000 read IOPS when using server cache)
Maximum throughput/volume	320 MB/s	160 MB/s	200 MB/s
Maximum throughput/instance and VM	800 MB/s	800 MB/s	512 MB/s
Price (AWS based on U.S.-East; Azure based on U.S.-West location)	$0.125 GB/month of provisioned storage $0.065 per provisioned IOPS/month	$0.10 per GB per month of provisioned storage	$0.15 GB/month (P10) $0.14 GB/month (P20) $0.13 GB/month (P30)

Source: Gartner (July 2015)

For the most up-to-date pricing information, visit the provider's website:

AWS EBS Provisioned IOPS/EBS General-Purpose SSD (http://aws.amazon.com/ebs/pricing/)

Azure Premium Storage (https://azure.microsoft.com/en-us/documentation/articles/storage-premium-storage-preview-portal/)

AWS-Provisioned IOPS

AWS EBS provides Provisioned IOPS volumes backed by SSDs. Clients can specify during the volume creation the exact number of IOPS desired, up to the maximum allowed (see Table 18 for maximums).

A single EBS-Provisioned IOPS volume can range in size from 4GB to 16TB. During the creation of the volume, cloud architects can specify either the capacity required or the IOPS required. If IOPS required is chosen, architects should scale the capacity based on a 30 IOPS per GB formula. For example, if the desired IOPS are 2,000, then the volume size would be 66.66GB (2,000 IOPS/30).

Clients can provision up to 20,000 IOPS per volume and stripe across multiple volumes for a maximum of 48,000 IOPS per instance. Provisioned IOPS volumes can be attached to any instance in AWS; they are not limited to an instance family.

Azure Premium Storage

The Azure high-performance storage offering is called Premium Storage. Premium Storage disks provide a consistent low-latency and predictable I/O throughput service. Premium Storage disks offer 5,000 IOPS per 1TB, and clients can stripe up to 32 disks together for higher performance. Azure Premium Storage disks can scale up to 32TB, and up to 50,000 IOPS per VM (see Table 19). Azure Premium Storage is backed by SSD disks and enjoys the same data durability level that LRS provides.

Table 19. Azure Premium Storage Disk Types

Premium Storage Disk Type	P10	P20	P30
Disk size	128GB	512GB	1,024GB
IOPS per disk	500	2,300	5,000
Throughput per disk	100 MB/s	150 MB/s	200 MB/s
Price per month (based on U.S.-East)	$19.71	$73.22	$135.17

Source: Adapted from Microsoft

The disadvantages to Premium Storage are that:

- Architects cannot manually specify the exact number of IOPS required, or the capacity required. Instead, architects would have to choose from three preconfigured disk options (as shown in Table 19).
- Azure Premium Storage is restricted to the DS-series VMs only, whereas AWS EBS Provisioned IOPS can be used with any EC2 Instance.

Host Cache

Azure Premium Storage also includes a read-only host cache capability that can be enabled at the time of Premium Storage disk creation. When enabled, host cache leverages the local SSD-based temporary disk that is automatically created with every DS-series VM to cache and boost read IOPS operations.

It is important to note that the cache IOPS are not subject to the Premium Storage disk limits. Instead, cache boosts read IOPS based on the VM's configuration. A DS-series VM will typically average about 4,000 IOPS and 33 MB/s per core, or up to 64,000 read IOPS per VM.

Host cache is an advantage that Azure Premium Storage has over AWS EBS Provisioned IOPS, which does not have a similar capability.

Azure Premium Storage Considerations

At the time of the writing of this document, there are some additional restrictions to consider for Azure Premium Storage:

- Clients must create a Premium Storage account.
- Azure Premium Storage accounts are limited to 35TB. Architects should consider this limit and create additional Premium Storage accounts as needed.
- It is available through the Microsoft Azure Preview Portal, PowerShell or CLI.
- It is only supported with DS-series VMs.
- It is only available in the following regions: West U.S., East U.S. 2, West Europe, East China, Southeast Asia and West Japan.

Local Host Storage

Both AWS and Azure offer storage that is provisioned from the direct-attached storage on the server hosting the instance or VM. However, AWS's and Azure's implementations of local host storage is very different in capabilities and limitations.

AWS INSTANCE STORE

AWS refers to local host storage as an "instance store." It is block-level, temporary (ephemeral) storage directly provisioned from the server that hosts the instance. Instance stores are provisioned automatically during the deployment of an AMI, providing that the AMI configuration includes an instance store. The process does not require any manual configuration to connect the instance to the storage.

Instance stores are free and offer acceptable performance for many applications, just as direct-attached disk is useful in some applications in the data center. When used with the right workload, they can save money and be quite effective. Because instance stores are temporary storage, clients that want to use them should be aware of the following characteristics and limitations:

- Data will persist during reboot.
- Data will be available during the lifetime of the instance.

Instance stores will be deleted if the instance is terminated.

Once the instance store is deleted, it cannot be restored.

The instance store is local to its corresponding instance and cannot be reassigned.

Data cannot be recovered.

Instance stores are frequently used for Web server farms, remote desktop session hosts, Citrix XenApp servers or compute clusters where the loss of the local disk is not critical.

AZURE TEMPORARY DISKS

Azure's implementations of local host storage are called temporary disks. Contrary to the way AWS does instance stores, Azure temporary disks are automatically created with every VM and are used to store temporary application and OS files, such as the Windows pagefile. During the provisioning process of a Windows VM, a temporary disk is created with the drive letter D:\; during the provisioning of a Linux VM, it is /dev/sdb1. Disk sizes will vary based on the resource configuration of the VM, and they cannot be manually configured.

Where AWS's instance store has an advantage is the fact that it can be used in some use cases because its data persistence rules are manageable. Azure temporary disks, on the other hand, are significantly more temporary; consequently, using them in any capacity is harder. Here are some of the characteristics and limitations of Azure temporary disks:

Azure temporary disks are free.

Data will persist during reboot.

Data does not persist during live migrate.

Data cannot be recovered.

Microsoft very clearly states that the purpose of temporary disks is to save the system paging file, and they should not be used to store any data that the client cannot afford to lose.

Network-Shared Storage

Network file shared storage is important to clients that are looking to migrate on-premises applications that already rely on this type of storage, as well as for clients that are looking to deploy new cloud-native applications. Both AWS and Azure offer implementations of a managed network shared-storage infrastructure. Gartner believes that both offerings are incomplete due to a lack of enterprise-grade features like snapshot and copy. Table 20 provides a side-by-side comparison of the features and capabilities.

Table 20. AWS EFS and Azure Files Comparison

Industry	AWS	Azure
Protocol	NFS v4.0	SMB 2.1
Capacity	Grows/shrinks automatically	Up to 5TB per share; no limit on number of shares (account limit applies)
Create share method	AWS Management Console; CLI; software development kit (SDK)	Management Portal/PowerShell/REST API
Client OS	Linux	Windows/Linux
Snapshots and copy	N/A	N/A
Authentication/security	Identity and access management (IAM), Security Groups, ACLs and inbound/outbound filtering at instance and subnet	Shared Keys

Throughput	50 MB/s to 1 GB/s	Up to 60 MB/s per file share
IOPS performance	Baseline: ~1,500 IOPS per TB Burst: ~3,000 IOPS per TB	1,000 IOPS per share
Disk type	SSD	Mix of conventional and SSD storage
Internet-accessible	No	Yes, via REST APIs
Price	$0.30 per GB per month	LRS: $0.08/GB GRS: $0.10/GB

Source: Gartner (July 2015)

The key points are:

Protocol: AWS's implementation of a network-shared storage is called Elastic File System (EFS) and is based on the NFS v4.0 protocol. Microsoft's implementation is called Azure Files and is based on the SMB 2.1 protocol. In this instance, both services need to support NFS and SMB, and Microsoft must support a more recent, more robust version of SMB, namely SMB 3.0. A big miss for AWS is that it only supports Linux from a client OS perspective, although Windows users can install an NFS client for Windows and access the service. Azure Files supports both Windows and Linux clients natively.

Scalability: The AWS implementation is a more cloudlike implementation in that it provisions the share with 0KB and then grows and shrinks the share dynamically depending on the amount of data stored on it. EFS can grow to petabyte scale. Conversely, Azure Files needs to be preprovisioned with a specific fixed capacity that doesn't change over the lifetime of the share. Azure Files also has a capacity limit of 5TB per share. However, apart from storage account limits, there is no limit to the number of shares that can be provisioned. Both EFS and Azure Files are priced based on the storage space used and not on the provisioned space.

Provisioning: AWS can provision shares from the AWS Management Console, CLI or SDK, while Azure Files can provision shares from the Azure Management Portal, PowerShell and REST API.

Authentication: AWS also has an advantage in that it supports ACLs, security groups and IAM. Azure Files currently relies on storage Shared Keys. Azure uses a separate authentication method based on encrypted keys for allowing access to storage resources; this method is called Shared Keys. For more information about Azure storage authentication, visit the Authentication for the Azure Storage Services (https://msdn.microsoft.com/en-us/library/azure/dd179428.aspx) Web page.

Performance: From a performance perspective, AWS EFS is built on an all-SSD platform and therefore offers superior IOPS and throughput. EFS is also architected at the region level, spreading data across multiple AZs for HA. EFS is built on top of AWS's EBS, not on AWS's object-based storage, S3.

Azure Files is built on top of Block Blobs, Azure's object-based storage, so it inherits the mixed-storage implementation used for Block Blobs, along with its performance limitations. Azure Files offers data durability by leveraging LRS and GRS.

Price: There is a significant gap in pricing between AWS EFS and Azure Files. However, this price difference is because EFS is built on top of EBS and uses all-SSD storage. Meanwhile, Azure Files is built on top of Azure's object-based storage, Block Blobs, so pricing is similar to that storage platform.

Internet accessibility: Azure Files can be accessed over the Internet via REST API — AWS EFS cannot. If there are applications that require a file share to drop files over the Internet, Azure Files offers that feature. It is important to note the share is not mountable via SMB/Common Internet File System (CIFS) over the Internet.

Data Sovereignty

AWS implements data sovereignty by limiting the data to a single region, and with AWS's multiple AZs in every region architecture, the data is maintained within the 60 miles (100 kilometers) that separate AZs. For clients that wish to separate the data across distances greater than 200 miles (322 kilometers), AWS also offers CRR, which allows the customer to select which region to replicate the data to in order to maintain data sovereignty. With CRR, every object uploaded to an S3 bucket is automatically replicated to a destination bucket in

a different AWS region of the customer's choosing. CRR is available in the U.S. Standard, U.S. West (Oregon), U.S. West (Northern California), EU (Ireland), EU (Frankfurt), Asia/Pacific (Tokyo), Asia/Pacific (Singapore), Asia/Pacific (Sydney) and South America (São Paulo) regions.

Azure also limits data to a single region and has carefully paired regions together under the same geographic area in order to maintain data sovereignty during a GRS replication. The one exception to Azure's geographic areas is Brazil, which is currently paired with a U.S.-based region (South Central U.S.). Table 21 shows Azure region pairings.

Table 21. Azure Data Sovereignty Regional Pairing

Primary Region	Secondary Region
North Central U.S.	South Central U.S.
South Central U.S.	North Central U.S.
East U.S.	West U.S.
West U.S.	East U.S.
U.S. East 2	Central U.S.
Central U.S.	U.S. East 2
North Europe	West Europe
West Europe	North Europe
South East Asia	East Asia
East Asia	South East Asia
East China	North China
North China	East China
Japan East	Japan West
Japan West	Japan East
Brazil South	**South Central U.S.**
Australia East	Australia Southeast
Australia Southeast	Australia East

Source: Adapted from Microsoft

Data Encryption

Gartner clients are increasingly concerned about data privacy and security in the public cloud. AWS offers two options to choose from:

Server-side encryption (data at rest): Customers can encrypt data that is stored with their VMs in the cloud.

Client-side encryption (data in flight): Customers can encrypt their data from the moment it leaves their data center.

Amazon S3 server-side encryption (SSE) has three options that a client can choose from:

SSE with Amazon S3-managed keys (SSE-S3): All objects are encrypted with a unique key, but AWS manages the encryption keys on behalf of the customer.

SSE with AWS Key Management Service (KMS)-managed keys (SSE-KMS): This service is similar to SSE-S3 except that it adds another layer of security in offering an envelope key (in other words, a key that protects the encryption keys that are used to encrypt the data). It also offers an audit report that shows when the keys were last used and by whom.

SSE with customer-provided keys (SSE-C): This method offers clients full control over encryption and decryption, along with the keys and the tools used.

On the client-side encryption side, AWS offers two options:

AWS KMS-managed customer master key (CMK): With this option, clients leverage AWS's KMS to generate encryption keys. These keys are then presented to the AWS S3 encryption client, and it does the rest. The client is still responsible for encrypting and decrypting the data. AWS's KMS simply secures and encrypts the keys that clients use to encrypt and decrypt their data.

Using a client-side master key: The client-side master key and the unencrypted data are never sent to AWS; it is, therefore, completely on the client to secure the keys and to encrypt and decrypt the data.

Azure, on the other hand, offers only client-side encryption. In general, the advantage that client-side encryption has over server-side is that clients are in complete control of the encryption keys. Microsoft's implementation is called client-side encryption for Microsoft Azure Storage. It offers options that are similar to AWS:

It uses the envelope technique to encrypt and secure the keys that are used to encrypt the data.

It allows clients to use Azure Key Vault to generate the keys to be used in the envelope technique, and it allows clients to use third-party services if clients prefer not to use Azure Key Vault.

It supports the following Azure Storage services: Block and Page Blobs, Queues, and Tables.

AWS and Azure also offer encrypted key management solutions; AWS has KMS, and Azure has Key Vault. Both providers also allow clients to use third-party key management services if they choose to not leverage the AWS and Azure solutions.

Cloud architects will have to determine which provider to use, making an educated comparison between AWS and Azure critical. In some cases, the answer is to use both for different workloads and to take advantage of the areas where each provider offers the highest value.

In a public cloud IaaS, unlike traditional data center infrastructure, IT is unaware of the infrastructure design, implementation or monitoring in any great detail. Therefore, cloud architects should assume an unreliable infrastructure and should design for failure by taking advantage of the capabilities provided by the platform, such as load balancing and auto scaling.

Guidance for clients choosing AWS:

Design for HA across AZs when using AWS to deliver active-active application deployment that meets Amazon SLA requirements.

Configure AWS ELB to integrate with Amazon CloudWatch for advanced, metrics-based load-balancing capabilities.

Design AWS Auto Scaling to deploy instances when certain thresholds are exceeded, but also remember to configure Auto Scaling to terminate instances when metric usage falls below threshold highs.

Configure Direct Connect with redundant ports per AWS best practices, and remember that, by default, redundancy is not included.

Use an Amazon EC2 instance store with the right workload type, and only after you have properly studied the pros and cons.

Guidance for clients choosing Azure:

Design Autoscale with enough preprovisioned VMs to meet your workload's peak requirements. Continuously monitor and adjust preprovisioned VMs to satisfy workload requirements.

Understand ALB limitations if you plan to deploy workloads that require load-balancing capabilities. If Layer 7 functionality is required, you will need to use the Azure Application Gateway in addition to ALB.

Design for HA within a logical data center when using Azure, leveraging Availability Sets, Fault Domains and Update Domains to provide applications the highest levels of uptime and to adhere to Azure SLAs.

Leverage the Azure ability to expand block storage volumes to dynamically increase the size of a VM's volume.

Integrate on-premises Hyper-V and System Center deployments to take advantage of Azure services.

General guidance:

Clients that are designing a multicloud strategy should consider AWS and Azure as ideal candidates given their current market leadership and feature set capabilities.

Test your design and configuration. AWS and Azure offer a free tier that cloud architects can leverage before incurring cost.

Do not overprovision — it is very easy to reconfigure instances or VMs in AWS and Azure. It is also very easy to increase storage capacity. Because everything is very fluid, overprovisioning is unnecessary, which is a practice that is very difficult to do with on-premises data centers.

When using on-demand instances from both AWS and Azure, remember that AWS bills by the hour, and Azure bills by the minute.

Cost per hour should not be the primary driver when designing for IaaS in the public cloud. Focus on the speed, agility and elasticity of services and their impact on your business.

Evaluate as many of your workloads as you can for public cloud IaaS fitness. This exercise will identify the cloud service provider that can support the largest number of your applications and may indicate the need for a multiprovider strategy.

Some documents may not be available as part of your current Gartner subscription.

"Public Cloud Object Storage Comparison: Amazon, Google and Microsoft" (http://www.gartner.com/document/code/262420?ref=ggrec&refval=3093919&latest=true)

"Amazon Web Services: In-Depth Assessment" (http://www.gartner.com/document/code/263761?ref=ggrec&refval=3093919&latest=true)

"Microsoft Azure: In-Depth Assessment" (http://www.gartner.com/document/code/263763?ref=ggrec&refval=3093919&latest=true)

"A Manager's Introduction to Amazon Web Services, 2014" (http://www.gartner.com/document/code/268418?ref=ggrec&refval=3093919)

"Evaluation Criteria for Cloud Infrastructure as a Service" (http://www.gartner.com/document/code/277367?ref=ggrec&refval=3093919&latest=true)

"Implementing Effective IaaS Cloud Security in Microsoft Azure" (http://www.gartner.com/document/code/272882?ref=ggrec&refval=3093919&latest=true)

"Implementing Effective IaaS Cloud Security in Amazon Web Services" (http://www.gartner.com/document/code/260748?ref=ggrec&refval=3093919&latest=true)

Amazon EFS — Preview (http://aws.amazon.com/efs/)

Amazon EBS Volume Types (http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSVolumeTypes.html)

AWS Direct Connect (http://aws.amazon.com/directconnect/)

Amazon EBS Product Details (http://aws.amazon.com/ebs/details/)

Amazon EBS Pricing (http://aws.amazon.com/ebs/pricing/)

Elastic Load Balancing (http://aws.amazon.com/elasticloadbalancing/)

Amazon EC2 Reserved Instances (http://aws.amazon.com/ec2/purchasing-options/reserved-instances/)

Auto Scaling (http://aws.amazon.com/autoscaling/)

Global Infrastructure (http://aws.amazon.com/about-aws/global-infrastructure/)

Understanding the Temporary Drive on Windows Azure Virtual Machines (http://blogs.msdn.com/b/mast/archive/2013/12/07/understanding-the-temporary-drive-on-windows-azure-virtual-machines.aspx)

Exploring Windows Azure Drives, Disks, and Images (http://blogs.msdn.com/b/windowsazurestorage/archive/2012/06/28/exploring-windows-azure-drives-disks-and-images.aspx)

VPN Gateway Pricing (http://azure.microsoft.com/en-us/pricing/details/vpn-gateway/)

Application Gateway (http://azure.microsoft.com/en-us/services/application-gateway/)

Premium Storage: High-Performance Storage for Azure Virtual Machine Workloads (http://azure.microsoft.com/en-us/documentation/articles/storage-premium-storage-preview-portal/)

ExpressRoute Partners and Peering Locations (https://azure.microsoft.com/en-us/documentation/articles/expressroute-locations/)

Authentication for the Azure Storage Services (https://msdn.microsoft.com/en-us/library/azure/dd179428.aspx)

Client-Side Encryption for Microsoft Azure Storage — Preview (http://blogs.msdn.com/b/windowsazurestorage/archive/2015/04/28/client-side-encryption-for-microsoft-azure-storage-preview.aspx)

Azure Storage Pricing (http://azure.microsoft.com/en-us/pricing/details/storage/)

Azure Regions (http://azure.microsoft.com/en-us/regions/)
