Azure HPC Technical Onboarding Guide

August 2020

Updated to V1.1 December 2021

Document Owners: **Kanchan Mehrotra, Azure HPC Customer Engineering Team**

TABLE OF CONTENTS

Introduction

Building and maintaining HPC architectures is a complex proposal that requires finding that sweet spot between cost optimization and time efficiency. Cloud platforms initially offered theoretical capacity that HPC applications needed to scale but lacked the network maturity to support either big data transfers or hybrid app models where their big data sets remained on premises. Fast forward to today and we see a completely different capability landscape in the cloud for HPC, one that provides InfiniBand support for ultra-fast networking, high performance file systems and flexible file caching for big data, and scaling services designed to extend developers' current workflows without major disruption.

The cloud value proposition for HPC workflows has matured leaps and bounds in recent times. Microsoft has been moving fast in making HPC on Azure a reality for organizations with on-premise HPC clusters and/or dedicated bare metal environments by bringing those same tools, platforms, and operational flexibility to Azure. Strategic partnerships with market-leading technology OEMs like Mellanox, Intel, AMD, and NVIDIA coupled with HPC industry leaders like Cycle Computing, NetApp, Avere, and Cray (HPE) have contributed to the most well-outfitted, interoperable, and versatile cloud platform for HPC workloads.

The recent pandemic has shown us how quickly business and market conditions can change without advance warning. This is giving HPC organizations yet another reason to consider a cloud platform moving forward. The time is ripe to not only talk about the value of HPC on Azure, but rather explain the specific decision-making processes and walk through exactly how it can work for a given organization. There is no better time to drill into those specifics than now. This document illustrates some of the context and processes associated with onboarding an HPC application workload to Azure to help bridge your prospect's HPC familiarity and comfortability to a cloud-based future.

Specialized Workload Types

Compute-Intensive

There are two main types of HPC workloads: loosely coupled and tightly coupled. There are also workloads that have combination of the two, but usually in the form of serial jobs (e.g., first loosely coupled, and then tightly coupled—or vice versa). Loosely coupled workloads are ideal for the cloud because they require little or no communication between VMs, so the VMs do not need to be interconnected or located physically

close to each other either in the same region or the same datacenter. On the other hand, tightly coupled workloads require constant communication between nodes, which needs to be provided by a high speed, fast performing network. The networks supporting these workloads are usually separate from the front-end network that provides normal communication to/from the nodes (e.g., to the Internet). Tightly coupled jobs also require specialized hardware finely tuned and optimized, while loosely coupled workloads benefit mostly from pure scale (i.e., as many cores, or computers, as possible). As the only public cloud platform that offers VM instances with InfiniBand-enabled hardware, Azure provides significant performance advantages for running tightly coupled HPC workloads. This improved performance narrows the performance gap and gets to near or better performance than current on-premises infrastructures.

Tightly Coupled

- Nodes require significant crosstalk ("chatter") with each other to operate properly
- Jobs require very fast low latency, high throughput interconnects
- Examples: automotive crash simulation, fluid dynamics, climate modeling, reservoir simulation, manufacturing modeling, computational chemistry

Loosely Coupled

- Nodes processing the job can operate independently of each other and don't have a lot of crosstalk
- Jobs usually entail a parameter sweep, a job splitting, or a search/comparison through a large data set
- Examples: risk analysis, image/video rendering, genetic algorithms, sequence matching, Monte Carlo simulations, integrated circuit design

Visualization Workloads

- Remote visualization workloads (VDI/DaaS)
- Graphically or visually intensive applications
- Examples: Virtual desktops, Desktop-as-a-service, remote visualization, remote workforce collaboration, graphics rendering, video editing, gaming, remote visualization

Intelligent Computing

- Workloads converging with opportunities to leverage AI services
- Highly complex model training & inferencing on the edge where real time data processing is required.
- Examples: AI/ deep learning has seen prominent use of GPU but applications like computational fluid dynamics, climate, weather and ocean, molecular dynamics, quantum chemistry, and physics.

The diagram below helps convey how many of the common HPC application workloads falls across the tightly-vs-loosely coupled spectrum, as well as their data access needs.

The delta between tightly vs. loosely coupled workloads is not necessarily indicative of the corresponding memory or data intensity levels. For example, workloads that utilize vast data sets of large object files, such as 3D images, tend to be more data-intensive than workloads focused more on the scale of calculations necessary, as in finance or chemistry. But both types of scenarios contain both tightly and loosely coupled workloads.
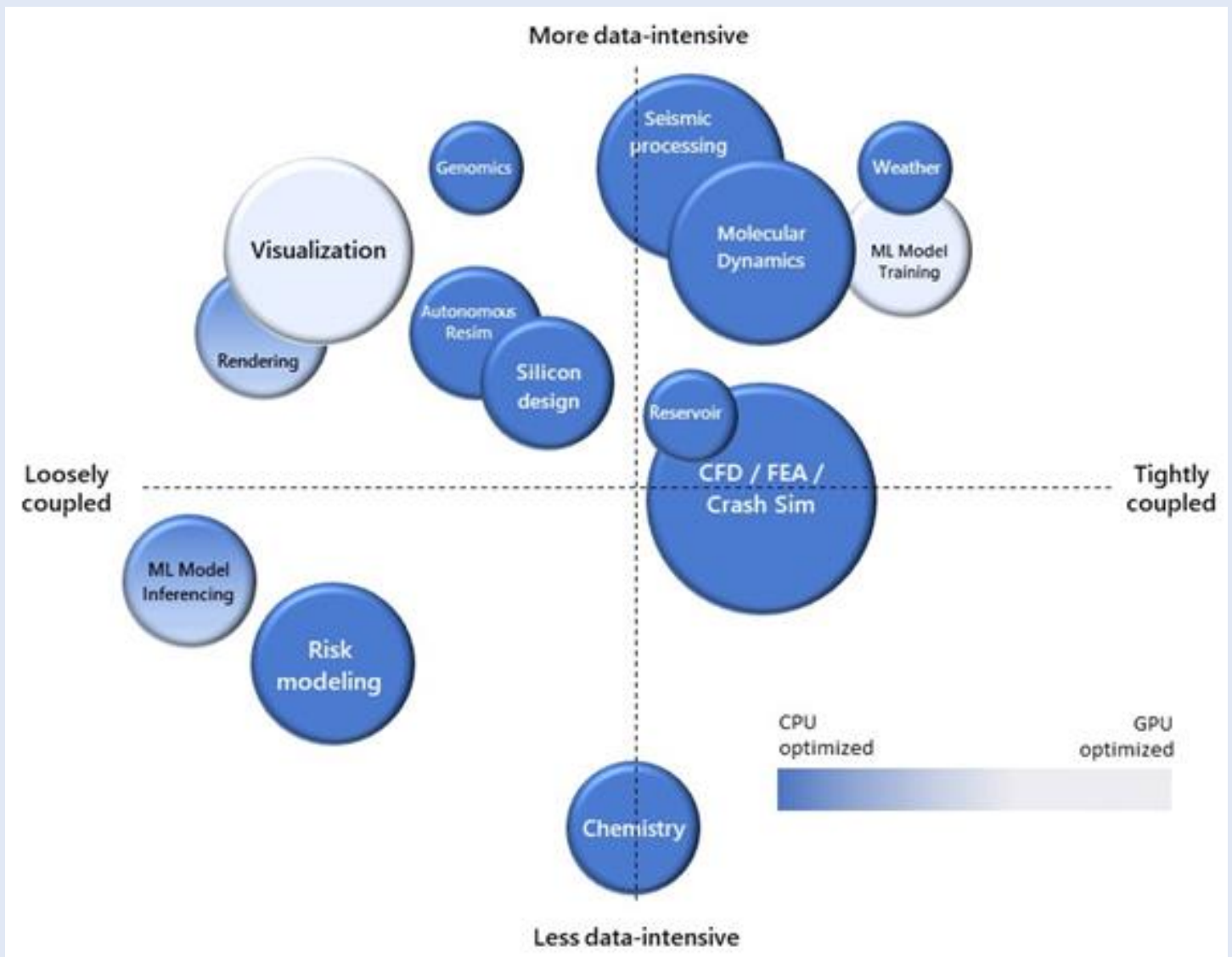
Typical HPC end-to-end environment

A typical HPC setup includes a front-end for submitting jobs, a job scheduler/workload manager (can also be named orchestrator), a compute cluster, AD authentication, and shared storage. While it is less complex to have all your resources in the cloud (compute, storage, and visualization), it is not uncommon for customers to have a hybrid model due to multiple business constraints.

Cloud Only

In a Cloud Only deployment you would normally have a front end for submitting your jobs to a workload manager / job scheduler (with autoscaling capabilities if desired). You can use PaaS storage offerings available or also setup your own IaaS cluster to back your workload. Authentication is managed through Active Directory.

The schema below illustrates a typical cloud only setup.

Cloud HPC scenario on Azure
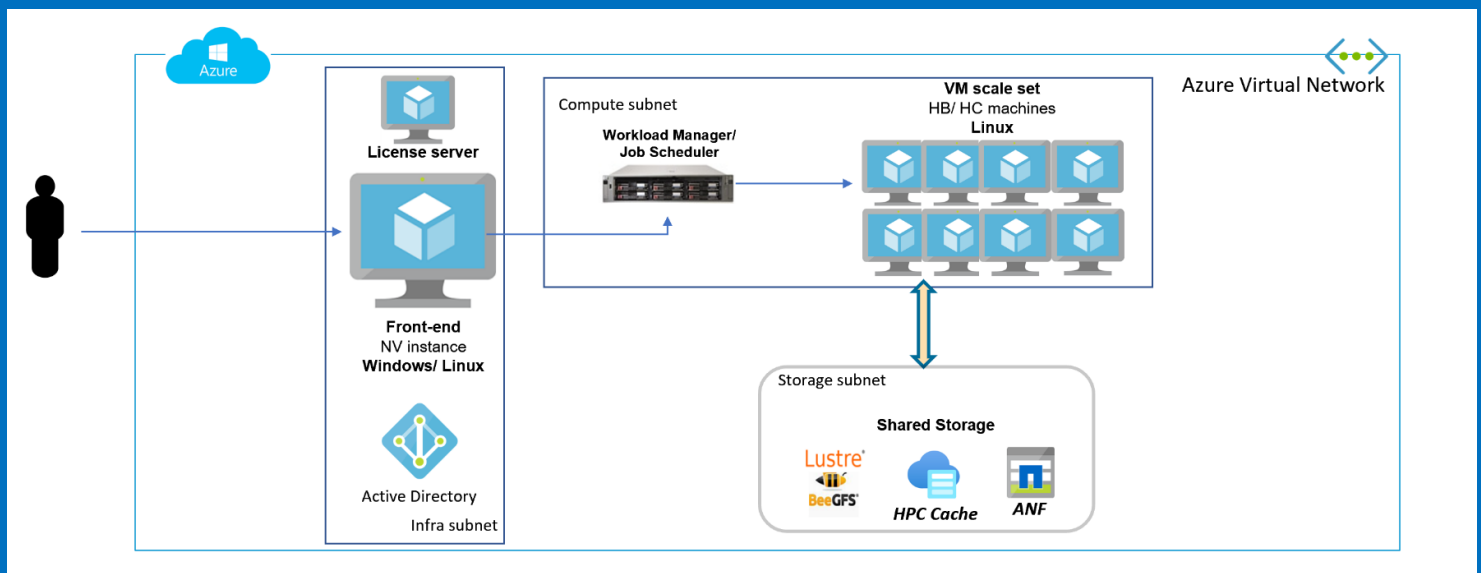
Hybrid

In a hybrid scenario you need to extend your on-premises capabilities, which comes with a couple of constraints mostly from the networking, security, and data access capabilities. Usually the first thing to do

is to establish a fast network connection between the Enterprise Network and Azure thru **Azure ExpressRoute** or thru an **Azure VPN Gateway** so that Azure resources are identified as an extension of your own resources. The diagram below illustrates how a cluster can be built in Azure with a dedicated job scheduler/ workload manager to provide auto-scaling capabilities if necessary. Authentication is done through Active Directory with a Read Only domain controller hosted in Azure to minimize traffic across the link with the enterprise. Finally, you need to minimize direct data access from Azure so you can optimize that network for latency, bandwidth, and cost.

As a result, you must either stage the whole dataset used by a project in Azure or implement a data mover as part of the job workflow.

Hybrid HPC scenario with Microsoft Azure
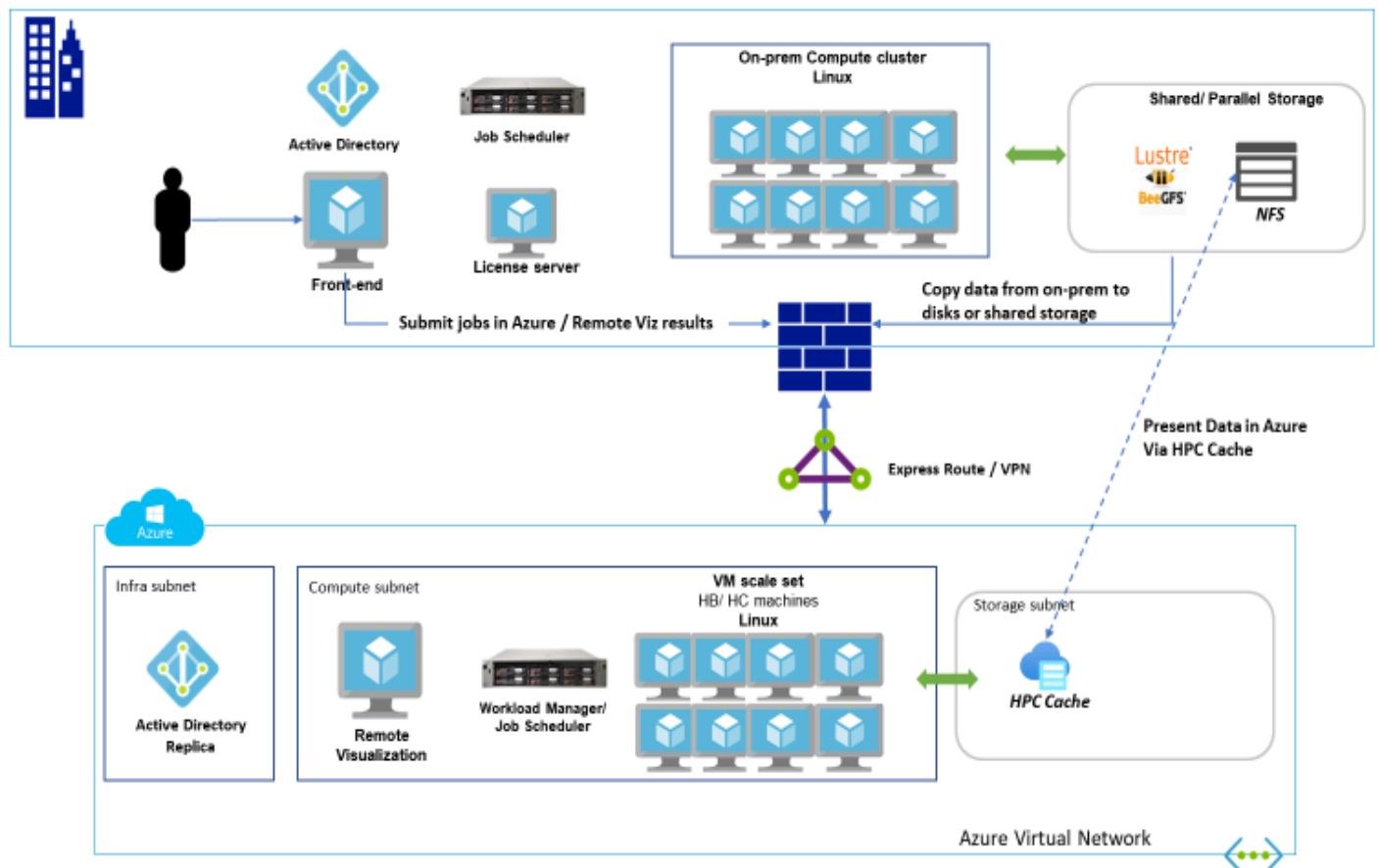
Selecting your HPC infrastructure on Microsoft Azure

Workload Manager/ Scheduler

Job queueing and scheduling are important logistical functions of HPC. Azure offers some specialized services that one can use to schedule compute-intensive work to run on a managed pool of virtual machines, and you can automatically scale compute resources to meet the needs of your jobs.
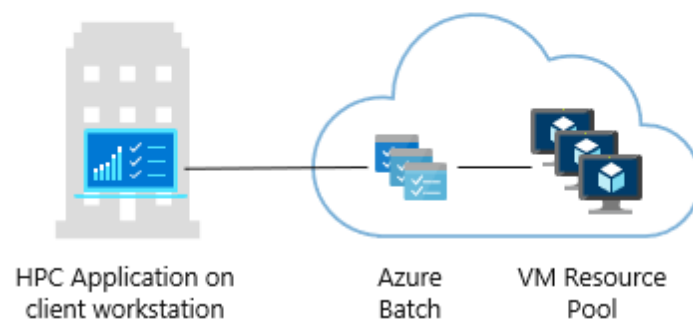
Azure Batch is a managed service for running large-scale HPC applications. Users specify their job parameters, the timing of the jobs, what regions to run them, and what specific VMs are required. The Azure Batch service then orchestrates everything based on those input parameters, including compute resource provisioning, task scheduling, automatic task recovery and/or retry on failure,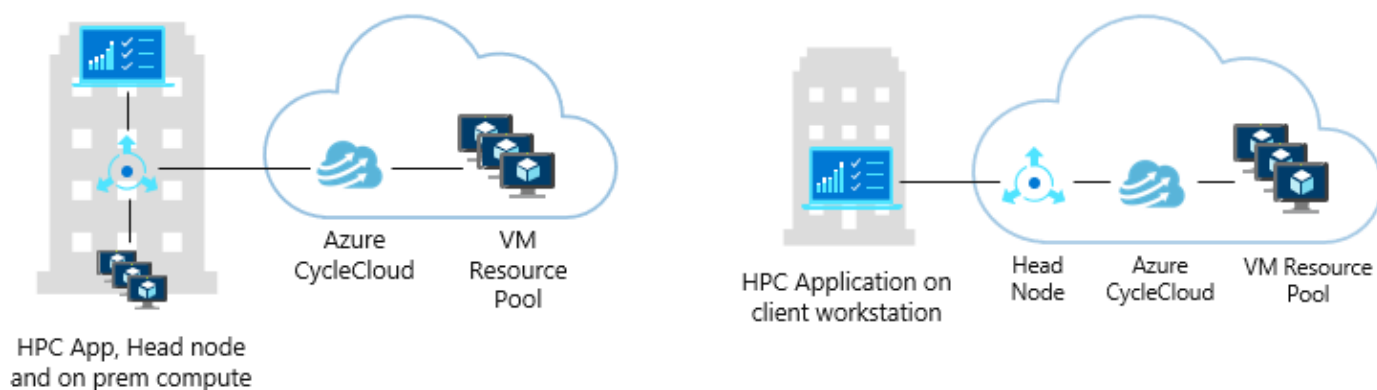 and auto-scaling if needed. Azure Batch service helps simplify how many other complexities that exist at cloud-scale can be addressed. Developers can also use Azure Batch as a platform service to build SaaS applications or client apps where large-scale execution is required, such as a large Monte Carlo risk simulation for a financial services company, or a fast media rendering application that offloads all rendering to the cloud in the background.

With Azure CycleCloud, users can dynamically provision HPC Azure clusters and orchestrate data and jobs for hybrid and cloud workflows. This cluster-based service is especially relevant in attracting traditional HPC engineers and developers that often the ultimate success in resourcing is a finely tuned, optimized compute cluster. Azure CycleCloud provides the simplest way to manage HPC workloads when using various Workload Managers like Grid Engine, HPC Pack, HTCondor, LSF, PBS Pro, Slurm, or Symphony on Azure. See the diagram at the top of the following page:

Azure CycleCloud

Traditional cluster scheduler orchestration

HPC App, Head node and on prem compute — Azure CycleCloud — VM Resource Pool

HPC Application on client workstation — Head Node — Azure CycleCloud — VM Resource Pool

Compute and Visualization

Azure offers a range of VM instances that are optimized for both areas of intense processing operation, whether it is more on the computational side utilizing CPU architectures or on the visualization side leveraging powerful GPUs.

Specialized HPC SKUs (such as the Azure H-Series and the Azure N-Series VMs), have been built specially for high-performance scenarios. However, Azure also offers other SKUs that may be suitable for certain workloads you are running on your HPC infrastructure, which could be run effectively on less expensive hardware. For example, some compute SKUs commonly used for loosely coupled Monte Carlo simulation workloads are the D-Series, the E-Series, and the F-Series VMs because they offer a good price-performance balance and offer enough processing capability to keep up.

The next page provides a quick 'at a glance' reference to the Azure VM family, but you can get all details about available Azure virtual machines, specs, pricing, etc. here.

InfiniBand

Azure H-Series
CPU-based VMs
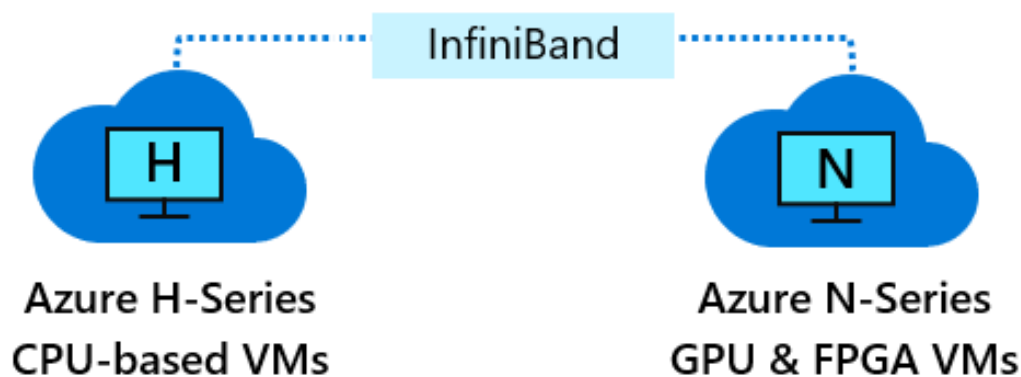
H—High memory

HB—Memory bandwidth

HC—Compute-bound

Azure N-Series
GPU & FPGA VMs

NV—Graphics applications

NC—GP-GPU compute

ND—Deep Learning

NP—Programmable FPGA

Azure A/B
Burstable VMs

A – Entry level, balanced

B – Economical bursting'

Azure D/E/F
General Purpose VMs

D - Standard workloads

E - Higher memory

F - Compute-bound

Azure L/M
High Memory VMs

M - Extreme memory

L - High SSD & IOPS

Azure Storage
Multiple options

>80,000 IOPs
Premium Storage

Low latency, high
throughput apps

Cray ClusterStor on Azure
Managed bare metal

Custom managed bare-metal

Large to extreme-scale HPC

Azure Network integration

Azure H-Series VM Specifications

	HBv3	HBv2	HB	HC	H
Workload Optimized	Memory Bandwidth	Memory Bandwidth	Memory Bandwidth	Dense Compute	Large-Memory HPC
CPU	AMD EPYC 3rd Gen w/ 3D V-Cache "Milan-X"	AMD EPYC 2nd Gen "Rome"	AMD EPYC 1st Gen "Naples"	Intel Xeon Platinum 1st Gen "Skylake"	Intel Xeon E5 v3 "Haswell"
Cores/VM	120	120	60	44	16
TeraFLOPS/VM (FP64)	4T	4 TF	.9 TF	2.6 TF	.7 TF
Memory Bandwidth	350 GB/sec	350 GB/s	263 GB/s	191 GB/s	80 GB/s
Memory	448 GB total	4 GB/core, 480 total	4 GB/core, 240 total	8 GB/core, 352 GB	14 GB/core, 224 GB
Local Disk	2 * 960 GB NVMe	960 GB NVMe	700 GB NVMe		2 TB SATA
InfiniBand	200 Gb HDR	200 Gb HDR	100 Gb EDR		56 Gb FDR
Network	50 GbE	50 GbE	50 GbE		16 GbE

[Learn more about Milan-X CPUs on Azure](#)

The H-Series is retiring in August 2022
The HB-Series is retiring in August 2024

RAM Per Core (Using Single VM)   MPI Parallel Processes (in thousands)
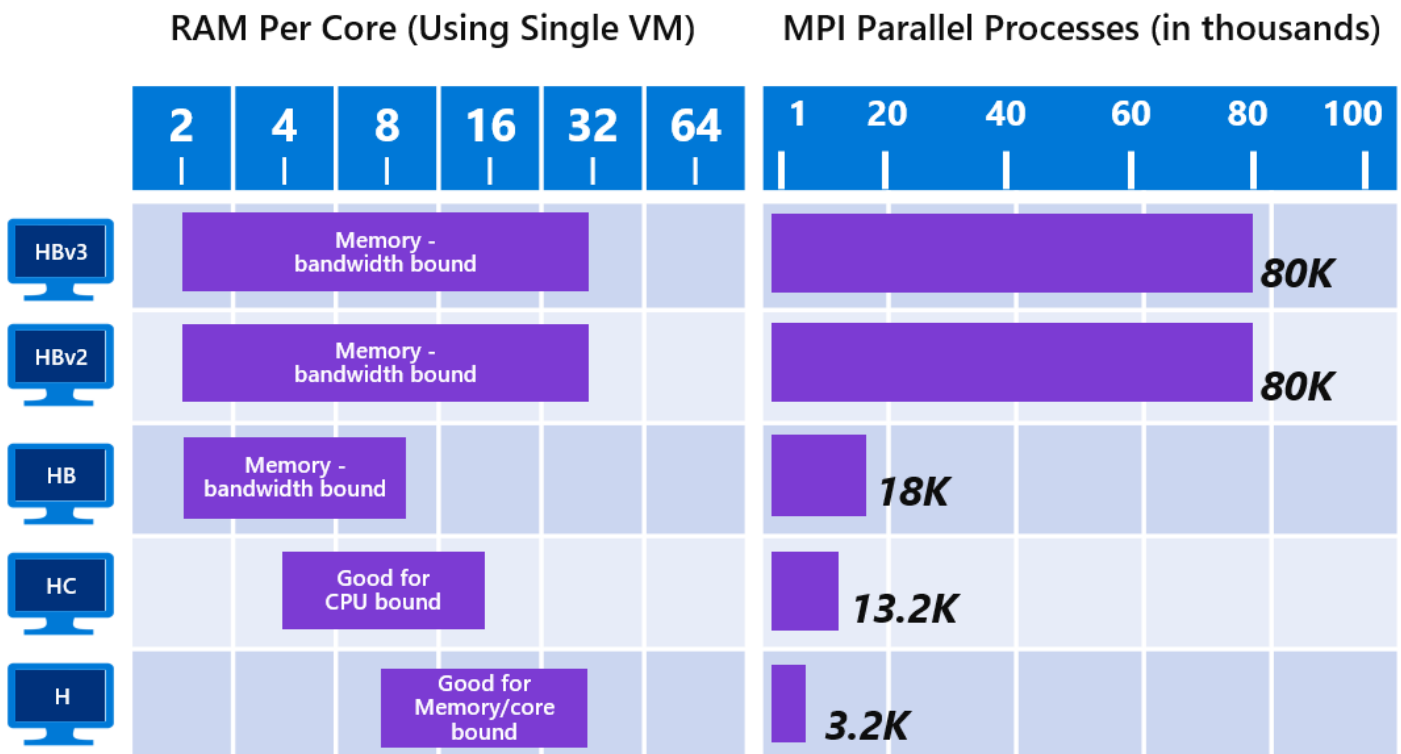
Always refer to our [HPC Architecture Center](#) for updated VM specifications and capabilities

Azure N-Series VM Specifications

GPUs for deep learning and real-time modeling:

	🖥️ ND	🖥️ NDv2	🖥️ ND A100 v4	🖥️ NDm A100 v4
CPU Cores	6,12,24	40	96	96
GPU	1x, 2x, or 4x P40 GPUs	8x V100 32 GB (NVLink) GPUs	8x A100 40 GB GPUs	8x A100 80 GB GPUs
Memory	12/224/448 GB	672 GB	900 GB	1900 GB
Local Disk	736/1474/2948 GiB SSD	2948 GiB SSD	6 TB SSD	6.4 TB SSD
Network	Azure Network + InfiniBand EDR	Azure Network + InfiniBand EDR + NVLink GPU Interconnect	Azure Network + InfiniBand EDR + NVLink GPU Interconnect	Azure Network + InfiniBand EDR + NVLink GPU Interconnect

Learn more about AI Supercomputing on Azure

The ND-Series is retiring in August 2022

GPUs for visualization, rendering, and remote desktops:

	🖥️ NV	🖥️ NVv3	🖥️ NVv4
Cores	6, 12, 24	12, 24, 48	4, 8, 16, 32
GPU	Tesla M60	Tesla M60	Radeon Instinct MI25
Memory	56,112,224 GB	112/224/448 GB	14/28/56/112 GB
Local Disk	340/680/1440 TB SSD	320/640/1280 GB SSD	88/176/352/704 GB
Network	Azure Network		

The NV-Series is retiring in August 2022

GPUs for computationally intensive applications:

	🖥️ NC	🖥️ NCv2	🖥️ NCv3	🖥️ NCasT4_v3
Cores	6, 12, 24	6, 12, 24	6, 12, 24	4, 8, 16, 64
GPU	Tesla K80	Tesla P100	Tesla V100	Tesla T4
Memory	56/112/224 GB	112/224/448 GB	112/224/448 GB	28/56/110/440 GB
Local Disk	340/680/1440 GB SSD	736/1474/2948 GB SSD	736/1474/2948 GB SSD	180/360/2880 GB SSD
Network	Azure Network + InfiniBand (largest size only)			

The NC-series and NCv2-series are retiring in August 2022

Always refer to our HPC Architecture Center for the most updated products & specifications

Storage

Using the right storage option is critical to ensuring an HPC application can execute its processing unfettered. There are a few factors that influence your data access approach:

Data location requirements. Your data may be largely located on-premises and compelling reasons (political, regulatory or technical) exist that require continued local storage.

Scale of cloud HPC deployment. Your Azure-based HPC cluster may grow to be fairly large, requiring an optimized approach to data access times.

Size of dataset. Workloads' data requirements vary – A job might have a small set of data run across a large number of machines, it may require the use of a large set of unstructured data processing, or anything in between.

Methods of access and security. The expected performance of a workload often relies of parallel file systems and their ability to facilitate large numbers of concurrent clients' reading and writing from the same file or files. These

POSIX support. Most HPC workloads rely on an underlying file structure known as POSIX. The ability to access bytes of data from files within directories, for example may be required. By contrast, object stores do not directly support POSIX file structures.

The concept of a **latency domain** must be considered as your HPC cluster scales, the size of your dataset grows, or the location of the source data becomes more distant. A latency domain defines an area of optimal latency (milliseconds or less) for pleasantly parallel HPC jobs. The goal is to ensure that the HPC VMs and the dataset are located in the same latency domain. This will ensure that the jobs will run as quickly as possible, which in turn creates more predictability around job run costs.

If the HPC workload is pleasantly parallelized, single node, single user w/no parallel I/O or ACL support, then you can stage the files in Azure Blob storage and from your VM's local disk. Azure VMs typically come with local SSDs, the capacity and performance of which will vary depending on the SKU type. For I/O intensive applications, the local SSDs could be used for scratch space. If more capacity is needed or persistent storage is required, Azure has a range of managed disks options as well. This is a cost-effective approach for small, portable data sets, but is not a viable storage option when shared file access is required, or the dataset is large.

A multi-node HPC cluster may also leverage local storage, either via the machines' local SSDs or Managed Disks. Assuming that the cluster and dataset are located within the same latency domain, and the dataset size is sufficiently small the copy-local approach may make sense. For example, if the data is located in Azure Blob and consists of gigabytes or small number of terabytes of data, you can leverage tools such as **azcopy** to transfer that data quickly to the VM disks. If however, the data is larger than a handful of terabytes or is located beyond the region where you are running your HPC cluster (for example, is located in your datacenter), then the latency of data transfer will come into play, especially as the size of the HPC cluster grows. Further, the data may be updated on a regular basis, and so ensuring that those updates are reflected on the disks will become increasingly difficult.

Shared File System

Shared file systems provide concurrent read/write access to data. Such file systems are client/server solutions based on access over a network leveraging well-defined protocol standards such as Network File System (NFS).

A **distributed file system** offers concurrent read/write access to data while ensuring the requests are serviced across multiple server entities. Distributed file systems distribute the requests and responses which allows the I/O and throughput to be scaled beyond what a single server shared file system could provide. Further, distributed file systems provide **high availability,** ensuring that jobs do not fail due to the underlying failure of a machine or software process.

A **parallel file system** offers the most performant storage architecture of all shared file systems. This is accomplished due to its ability to distribute large datasets across multiple servers. Parallel file systems support highly parallel I/O, allowing multiple clients to read and write concurrently from the same file. Parallel file systems consist of potentially large numbers of servers and ensure that requests are distributed across these servers. As a reference, most of the largest most parallel workloads must use parallel file systems, such as seismic analysis in the Oil & Gas exploration industry.
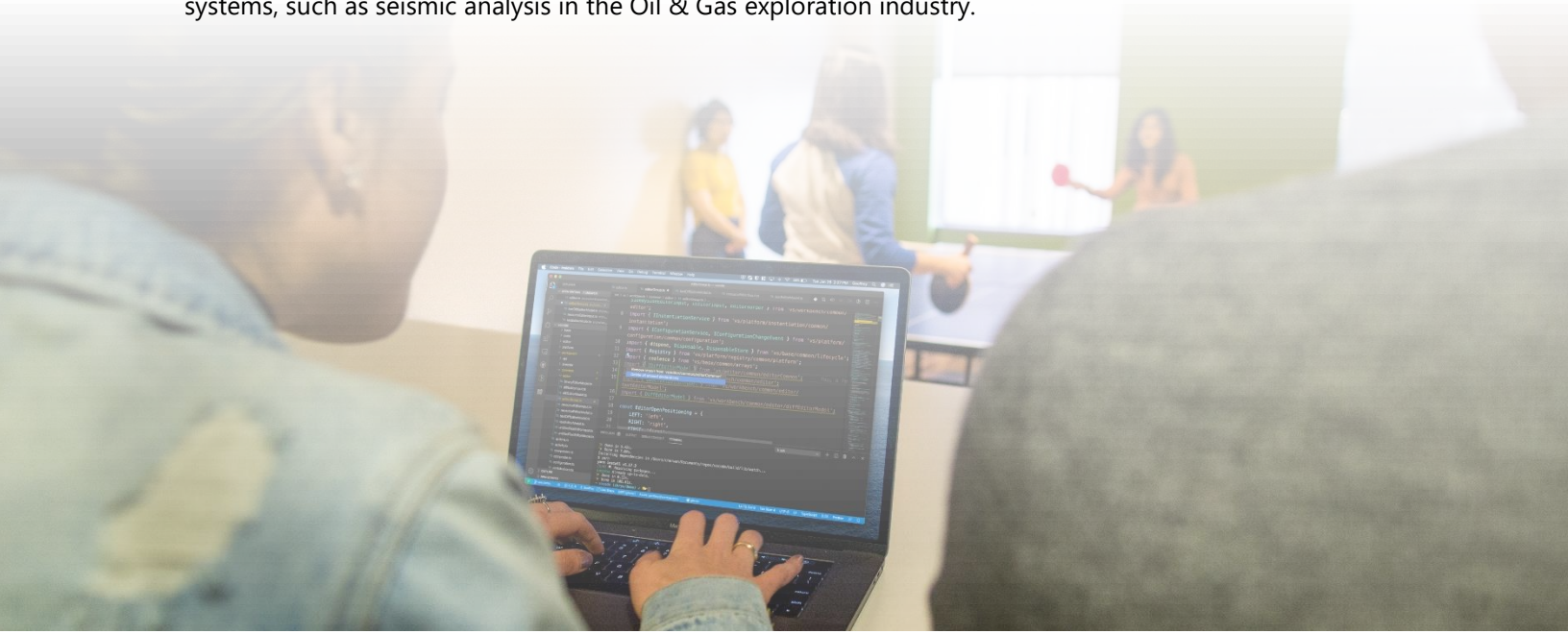
Shared file systems are needed when any of these conditions are met:

The **quantity of VMs** (or containers) operating against the dataset grows beyond a small number

The **size of the dataset** grows, and the ability to easily and quickly copy data to local disks becomes difficult to coordinate with job run times

The dataset is **accessed by other entities**, who may also update the data at any given time

The job writes out intermediate results which may be **consumed by any VM** in the HPC cluster

Data access requires ACL control, auditing, or other **security capabilities**

Considerations about the shared file system type to use include the following:

Scale. How many VMs are in the HPC Cluster? How much data is involved? How fast is the networking between the VMs (for example are they running InfiniBand)?

Duration of use. Is this a temporary file system that you will delete when your job is done running? Or is this system going to be a longer-running system that will continue serving data even when specific jobs aren't running?

Complexity. How much effort must you invest in creating and managing the filesystem?

Integration. Does the filesystem require interfaces with other systems? For example, does it require an interface with any sort of Directory Service?

Cost. How much is the total cost to run the file system, including your effort as well as the cost of the underlying systems?

Data location. Is the source data in Azure, in your datacenter, or perhaps in both locations?

Data characteristics. Is the source data a large number of very small files, a small number of very large files? How much writing of data will your HPC job do throughout its operation?

Performance

Several file system approaches exist in Azure.   The chart below provides a visual overview & summary of the points covered in this section of the paper:

	"Roll Your Own" NFS Server	Low Scale	Dev/Test Workloads	<1.5 GiBps; <19TB (Lsv2); 100TB (SSD); <500 cores
	Azure HPC Cache	Medium / Large Scale	Read-Heavy Workloads	Throughput 2GB/s to 16GB/s <192TB Cache; Backed by NAS or Blob Storage
	Azure NetApp Files	Medium Scale	Balanced or Write-Heavy Workloads	<4.5GiBps; <100TB/volume <4000 cores; 12TB max file size
lustre BeeGFS	"Roll Your Own" Parallel File System	Large Scale	Parallel Workloads	<50 GiBps Read/Write; <500TB; <50,000 cores
CRAY	Managed, bare metal HPC Storage	Maximum Scale	Highly Parallelized Workloads	<30 GiBps Read/Write; <500TB; <50,000 cores

Roll-Your-Own NFS (or SMB) Server

You can build your own SMB, or NFS file server based on either a Windows or Linux VM image. This approach is often used for dev/test environments or for highly ephemeral use cases (where the system may go away after completion). This approach requires the creation of the appropriate Virtual Machine and the configuration of the file server.   Rolling your own file system (NAS or Parallel File Systems) can provide a simple file environment, but comes with its own set of costs:

1. You must maintain the VM environment, including OS patches, for the life of the NAS of parallel FS

2. Availability of the file system is dependent on the underlying VM. NAS implementations offer an active-standby model but that requires complex configuration that must also be maintained.

3. Disk costs could grow significantly if the file system is kept for a long period of time. While you can copy data from the disks to something more long-term that requires managing a synchronization process, increasing TCO

4. You own responsibility for troubleshooting the file system itself, unless you purchase vendor-specific support.

16

Azure HPC Cache Hybrid File System

The HPC Cache is a high-performance **hybrid file system** that aggregates data from multiple sources and presents a global namespace to HPC clusters. You would leverage this technology when the required data is either (1) located in your on-premises data center and cannot be easily transferred (2) located in multiple systems or across multiple Azure regions or (3) located in both places (on-prem and in cloud). The purpose of this technology is to provide a low-latency and high-throughput file system that caches specific HPC cluster requests in memory and disk. The flexibility of the hybrid file system enables customers to keep data in place and only present requested data. Further, HPC Cache approaches data access on a per-byte basis rather than per-file, retrieving only the bytes requested by HPC cluster clients. This enables WAN architectures by limiting the amount of throughput required. Please note that if your workload is an Extract-Transform-Load or very write heavy application you would want to consider using other file systems.

Azure NetApp Files

NetApp's first party NAS-as-a-Service allows you to create a dynamically sized volume running on actual NetApp hardware. Because this is NetApp hardware, the overall latency of reading and writing data is quite low. This is especially useful for large numbers of small files but is certainly not limited to small file workloads. There are some resource limits to keep in mind when deploying Azure NetApp Files.

Roll-Your-Own Parallel File System

Just as with NFS, you can create a multi-node BeeGFS or Lustre file system. Performance of such systems is largely dependent on the type of Virtual Machines you select. You can leverage images found in the Azure Marketplace for **BeeGFS**, or a Lustre implementation by DDN called **Whamcloud**. Leveraging third party images from vendors such as BeeGFS (found here) or DDN allows you to purchase their support. Otherwise both BeeGFS and Lustre can be used via their GPL licenses without additional charges (beyond the machines and disks) and easy to roll-out with the scripts provided here with either ephemeral local disks (for scratch) or Premium / Ultra SSD for persistent storage.

Cray ClusterStor

One of the biggest challenges with the largest workloads is replicating the pure "bare-metal" performance of very large compute clusters working alongside very large Lustre environments (in terms of TB/s throughput, and perhaps in terms of Petabytes of storage). These workloads can now be run in Azure by leveraging the Cray ClusterStor solution. This is a pure bare-metal Lustre deployment placed in the relevant Azure data center.

17

Parallel file systems such as BeeGFS and Lustre provide the highest performance due to their architecture. However, that architecture comes with a high management price and so the use of these technologies are typically well-known by the existing application owners.

Building your HPC environment

Design Considerations

Designing an HPC solution in Azure opens up new scenarios and capabilities that need to be put in front of the application catalogue and workflow requirements. On-premises HPC facilities are usually designed in an attempt to provide a "one size fits all" approach to run many applications with a myriad of different requirements of memory, CPU, IOs, security and workflow. Constant tradeoffs need to be weighed and accommodated on premises, which may lead to struggling applications that aren't responsive or over-provisioned systems that are inducing more cost than necessary during peak usage or with super large shared storage systems. In contrast, the agility of building an HPC solution on Azure can be driven by a single application or by an application portfolio sharing the same profile requirements. Because most cloud-based parameters can be fully automated, these systems can be defined, created, and deleted "on-demand" to match these varying applications requirements. So while this will repurpose some operational IT cost to the automation process, it returns the value of high repeatability, reusability, testability and cost optimization, which - at the end of the day – is what an "HPC as a Service" model should theoretically provide.

Cloud only vs Hybrid

Now, when it comes to choose a Cloud only approach or Hybrid, this will be driven by the application workflow requirements and how much of it can be redesigned to embrace some of the cloud patterns. Questions which needs to be answered for help in the decision are:

- What is the data flow and size for the jobs?
- How jobs are submitted?
- Which authentication and authorization are required?
- Is there any license server to be used and how these licenses are shared between on-premises and Azure?
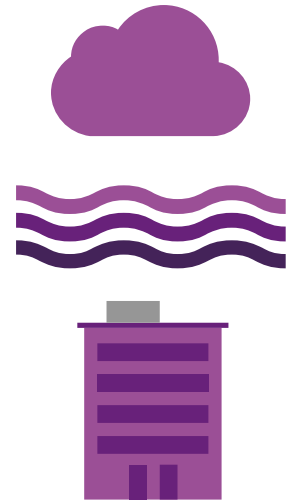
Scheduler (queues etc. in case of multiple workloads)

HPC systems relies on queueing and resource management system with complex policies to optimize the hardware resources and provide a fair sharing of these across all end users. It's usual to see submission tools, scripts, wrappers, or integration with in-house or ISVs applications to simplify, isolate and control how jobs can consume resources. Some customers are also extracting accounting information from the job schedulers database to report usage and sometimes charge back users.
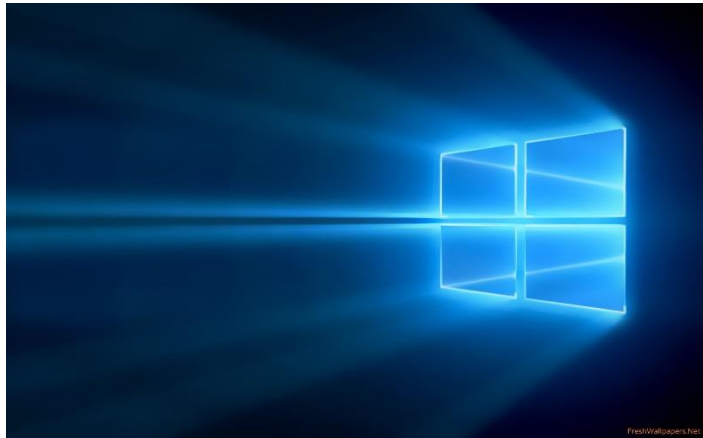
But when it comes to run these schedulers on moving infrastructures where nodes can appear and disappear and change their names, some of the legacy resource managers cannot keep up with this new world easily. When considering a lift & shift approach, this must be taken into account because while auto-scaling can help in optimizing costs, it is a process not well supported by existing job schedulers. This is where **Azure CycleCloud** can help in the job scheduler integration with Azure if you don't want to write your own integration. Moving forward it can be interesting to invest in using **Azure Batch** as a job scheduler and resource manager managed service and removed any overhead of having to manage a job scheduler infrastructure.

Platform Limits

Azure has its own limits which needs to be understood when designing HPC solutions. A first read to understand these is the public documentation all resources limits, but the main important one is certainly how many VMs can be part of a single MPI job. This limit is by default at 100 and can be extended to 300 upon special request. For storage, Azure NetApp Files limit to 100TB can be another limiting factor.

Remote desktop options

You can create powerful remote visualization workstations with Azure Virtual Machines. These VMs enable engineers needing high end workstation functionality, either at some or all of the time, to model and visualize processing job results and the ability to access those data sets. By making this move to the cloud, engineers and scientists are now able to make changes to their models and visualize results in or away from the office from their laptop. You can run most of the common remote desktop solutions on Azure today – HP RGS, TGX, Horizon, Citrix, Teradici to name a few. The document here provides guidance on how to setup some of the common remote desktop solutions effectively in Azure.

Ephemeral vs Persistent Storage

Azure Lv2 Virtual Machine that feature NVM Express (NVMe) disks that can be used in a shared filesystem. This VM provides a cost-effective way to provision a high-performance filesystem on Azure. The disks are internal to the physical host and don't have the same service-level agreement (SLA) as managed disk storage, but when coupled with a hardware security module (HSM) they are a fast on-demand, high-performance filesystem. You can find some examples here for deploying Lustre or BeeGFS file systems using Lv2.

Best Practices

Health checks

Many HPC applications are highly parallel and have tightly coupled communication, meaning that during a parallelized simulation job, all parallel processes must communicate with each other frequently. These types of applications usually perform best when the inter-communication between the parallel processes is done on high bandwidth/low latency networks. As the only public cloud directly supporting InfiniBand, Azure has the speed and capability to keep up with those inter-communication requirements. The tightly coupled nature of these applications means that if a single VM is not functioning optimally, then it may cause the job to have an impaired performance. The purpose of these checks/tests is to assist you in quickly identifying a non-optimal node, so it can be excluded from a parallel job. If your job needs an exact number of parallel processes, a slight overprovision is a good practice, just in case you find a few nodes that you need to exclude. The article here provides more information on how to run these health checks.

Application Checkpointing

Some HPC simulations take a long time to run and run over multiple nodes (up to days and weeks). When running these long jobs, failure rate and availability of the underlying infrastructure may become an issue. One solution for recovering from these failures is using checkpointing during the simulation to provide a starting point in case the simulation is terminated for any reason.

When running compute jobs in Azure, most issues from running on bare metal compute nodes still remain.

Hardware can and will fail, memory can have issues, and the InfiniBand network may encounter problems. On top of that, due to security and functional updates, the (virtualization) software stack may not remain static throughout the duration of the job.

A way to prevent the impact of the potential failures is to save intermediate results, in such a way that a calculation can be restarted from that result point. The creation of this

21
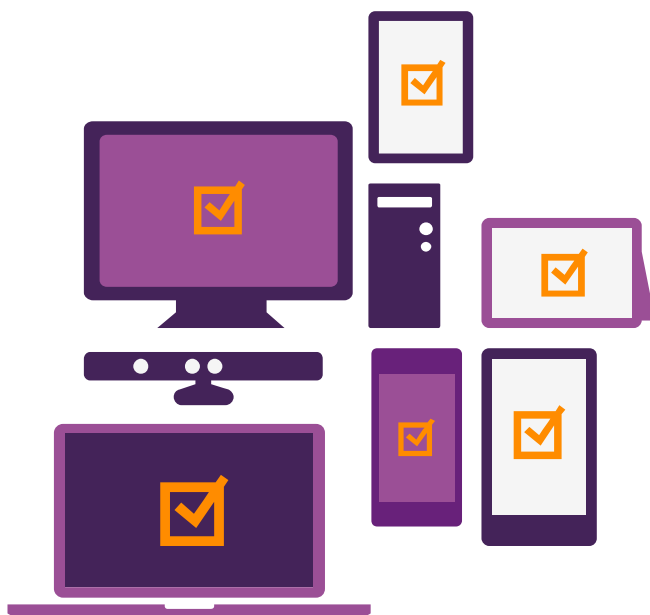
intermediate result is called creating a checkpoint. The article here covers checkpointing examples in more details for reference.

Cost optimizations

To make the most out of your cloud migration process, you must first prioritize cost management and up-front planning exercises. These are the most pivotal for a successful cloud migration effort for almost any organization. Fortunately, Azure Cost Management gives you the tools to plan for, analyze and reduce your spending to maximize your cloud investment. The document here offers an extensive list of ways you can optimize and plan your cloud spend. But, for the purposes of discussion, let's call out a few important ones here:

- Size your VMs properly to ensure balanced cost-for-performance

- Use Azure Hybrid Benefit to mitigate risks of extending on premises

- Azure Reservations for more predictable HPC patterns

- Use Azure Spot VMs where appropriate

- Use Autoscaling to allow you to automatically adjust the amount of compute resources used by your application, potentially saving you time and money. Both Azure Batch and CycleCloud support automatic scaling by dynamically adding or removing nodes based on your job queue. For more details on auto-scale in Azure, see here.

Monitoring

In a production environment, it's important to be able to track the way in which users use your system, trace resource utilization, and generally monitor the health and performance of your system. You can use this information as a diagnostic aid to detect and correct issues, and to help spot potential problems and prevent them from occurring. For an overview of the Azure components and services available to monitor Azure resources, see monitoring Azure applications and resources.

Further, if you are using Azure Batch, Batch Explorer is a free, rich-featured, stand-alone client tool to help create, debug, and monitor Azure Batch applications.

Also, Azure CycleCloud integrates with Azure services such as Azure Monitor and Azure Cost Management tools. It also supports monitoring of external services through its pluggable architecture. See more details here.

Start Building

Details below demonstrate how you can build the Cloud-only deployment shared above using some of the available tools:

Setting up HPC environment using CycleCloud

For a step by step walkthrough of deploying a CycleCloud environment and running a custom application you can refer to the CycleCloud QuickStart documentation.

Setting up HPC environment using Azure HPC OnDemand

The below diagram depicts the components needed in a typical on-premise e2e environment to support HPC end user workflow.
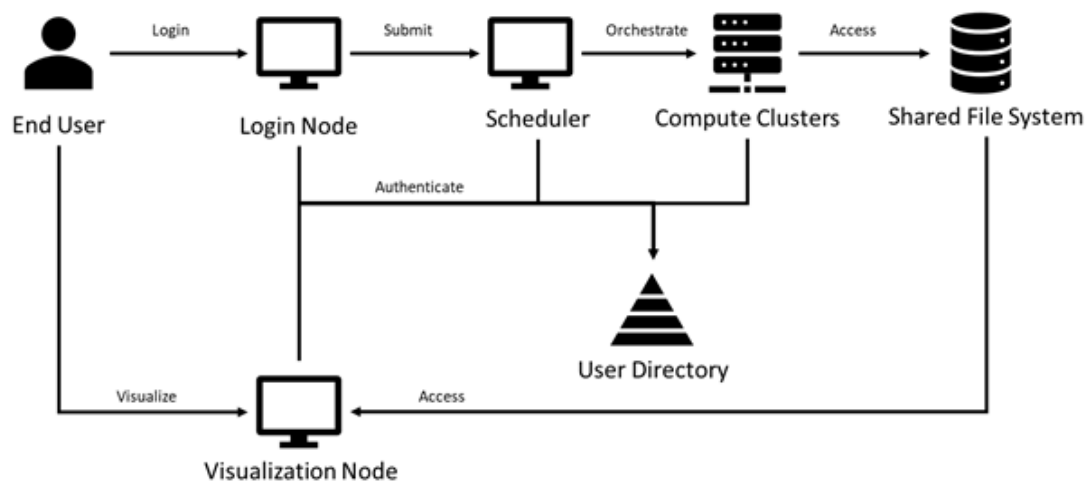
You can use **Azure HPC OnDemand Platform** (aka **az-hop**) available in the Azure/az-hop public GitHub repository to help HPC customers who need an e2e environment onboard faster. **az-hop** delivers a complete HPC cluster solution ready for users to run applications, which is easy to deploy and manage for HPC administrators. **az-hop** leverages the various Azure building blocks and can be used as-is, or easily customized and extended to meet any uncovered requirements.

Follow the blog here for more details on az-hop and how to set up your az-hop environment.

23

Application Specific Scripts and Guidance

There are several application setup and benchmark script examples available in the **azurehpc** GitHub repository here: https://github.com/Azure/azurehpc/tree/master/apps.

Additional Resources

Documentation and Blogs

- Azure HPC documentation
- Azure HPC building blocks
- Tech Community Blogs
- Azure Blog