The Red Oak Microsoft Azure HPC Collaboration Centre in partnership with Intel

Considerations for the migration of a HPC environment to the Azure Cloud

Manveer Munde
Dairsie Latimer

21 June 2021

Red Oak Consulting
Expert advice, exceptional delivery

Contents

Introduction

This is the last in a series of three white papers in which we, along with our partners at Microsoft and Intel, wish to share some of our best principles and practices for migrating an HPC environment to the Microsoft Azure cloud service with an emphasis on higher education. This work has been funded by Microsoft in partnership with Intel.

The first white paper, 'From requirement to proof of concept, preparing for HPC and research computing in the Azure Cloud', considered prerequisites and readiness requirements to get started with the cloud migration journey. The second white paper, 'Building and running a proof of concept in the Azure Cloud', discussed how to create a simple proof of concept (PoC) HPC environment on Microsoft Azure to gauge the performance of HPC workflows in the Cloud.[1]

Assuming a successful PoC, the next stage should be a pilot HPC environment which more closely resembles a production environment, with greater integration with existing IT infrastructure as well as a wider user community. With additional practical experience of the pilot environment, better choices regarding the production implementation, especially where compromises between security and usability may be necessary, can be made. Before going on, it is important to revise the differences between PoC, pilot and production HPC environments:

- **PoC**: A PoC environment is deliberately simplified and intended to run representative workloads from which performance (both scalability and absolute) can be gauged and from which a select few users can get a general feeling for the cloud environment. No organisation-specific requirements such as security, governance or user experience are considered in much detail here.
- **Pilot**: A pilot environment builds on a PoC, going some of the way to fulfilling organisation-specific requirements while leaving others out. For example, the pilot HPC system may be designed to simulate the intended HPC storage system, job accounting structure or user interface environment. On the other hand, full integration with an organisation's Active Directory for authentication and firewall would typically be reserved for the implementation of the production HPC environment since a deeper level of commitment would be required. In general, it is up to the organisation to decide the main aspects to concentrate on during the pilot stage while keeping in mind that this is still very much an experimental phase.
- **Production**: A production environment is fully integrated with all aspects of an organisation's infrastructure and complies fully with all security, governance and other requirements.

The reason for proceeding in three distinct steps from PoC, to pilot, to production is the minimisation and mitigation of risk. After each step, a review can be undertaken to ensure that the correct decisions are being made in carrying out a cloud migration.

In this third white paper, we discuss some key considerations for planning, designing and implementing a production-ready HPC environment on the Cloud. As a result, only a subset of what is discussed here applies to a pilot environment depending on an organisation's chosen scope and requirements. It should be noted that here we only aim to provide an overview rather than an in-depth discussion of all the various options available on Microsoft Azure. References to more comprehensive documentation are given where appropriate. In addition, as has been the case in the previous white papers, it is assumed that Azure CycleCloud is the chosen HPC solution with the SLURM scheduler configured.[2]

Finally, it should be noted that the advice provided in this white paper is a result of accumulated experience from over 20 real-world cloud-based projects which we have carried out in the past. These projects have spanned both the higher education and commercial sectors.

[1] https://www.redoakconsulting.co.uk/microsoftazurehpcwhitepapers2021
[2] https://docs.microsoft.com/en-us/azure/cyclecloud/overview?view=cyclecloud-8

Planning a cloud migration

Prerequisites to system design

In our experience, the following prerequisites are necessary before proceeding with the preliminary design stage:

- A restating of the organisation's business case for migrating to the Cloud in light of the PoC.
- The assignment of a project manager for the entire course of the cloud migration project. The chosen person should, if possible, possess relevant technical knowledge.
- The involvement of important stakeholders such as the to-be system administrator, relevant security and IT personnel and senior users.

In general, a cloud migration project can be broken down in order into the high-level design (HLD), low-level design stage (LLD), implementation and the acceptance testing stages. The aim of a phased approach like this is to minimise risk by carefully reviewing each stage on its completion and comparing with the business case. During previous projects, it has been the case that particular stakeholders (eg users) are not aware of the stages and need for this phased approach and, as a result, internal stakeholder communication needs to be well structured and delivered.

The high-level design (HLD)

A high-level design (HLD) contains basic conceptual designs of logical components and various lifecycles of the desired cloud HPC environment. These designs often take the form of flow diagrams as shown in Figure 1, which gives an overview of the infrastructure component of an example cloud HPC environment based on a previous production-level migration project. In this system, users log in to a cloud based virtual machine (VM) workstation via a jump server from which data processing can take place, and from which users can access the HPC cluster login node. Before logging in users must pass a multi-factor authentication (MFA) step. Typical of an HLD, Figure 1 contains no details of the specific Microsoft Azure components to carry out the various processes.

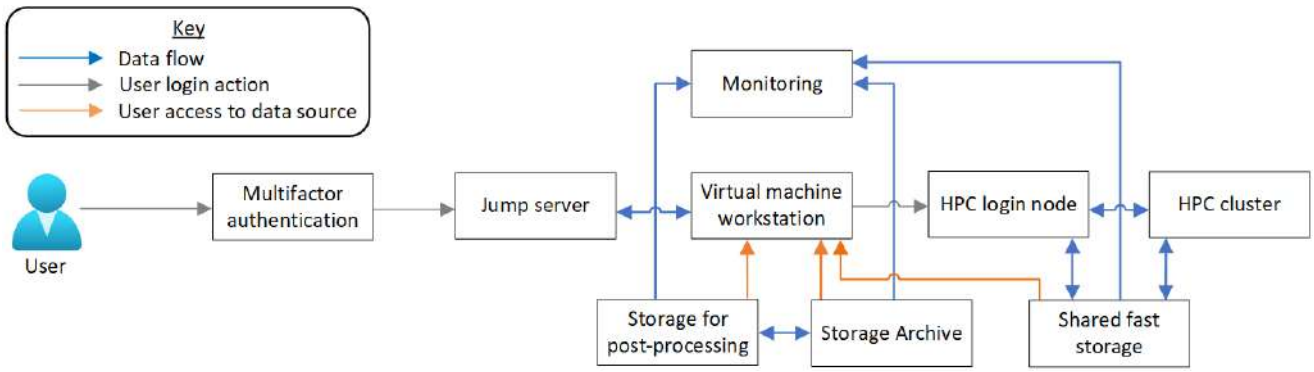

Figure 1: An example HLD schematic for the infrastructure components of a cloud HPC environment

In addition to Figure 1, an HLD will contain additional conceptual diagrams focusing more deeply on aspects such as business processes, security requirements and the data lifecycle. Assuming the content in the HLD is judged to fulfil the requirements of the business case, work can begin on the next planning stage.

The low-level design (LLD)

The low-level design (LLD) essentially takes the conceptual diagrams in the HLD and refines them with explicit details. For example, the design would now consider the exact cloud services responsible for particular processes. The successor to Figure 1 in an LLD may look like the diagram in Figure 2.

In this case, the MFA service is specified to be Azure Active Directory (AAD) and the jump server is specified to be provided by the Azure Bastion service: a user would log in to the Azure Portal using MFA via AAD and then use the Azure Bastion service to connect to a Linux workstation which makes use of a D4s v3 series VM. The HPC cluster itself requires additional components on Microsoft Azure not detailed on the HLD such as the CycleCloud orchestration server for HPC node configuration and the Azure Database for MariaDB to manage HPC job accounting (assuming the SLURM scheduler is used). Finally, detailed information is provided with respect to storage, monitoring services and networking infrastructure.

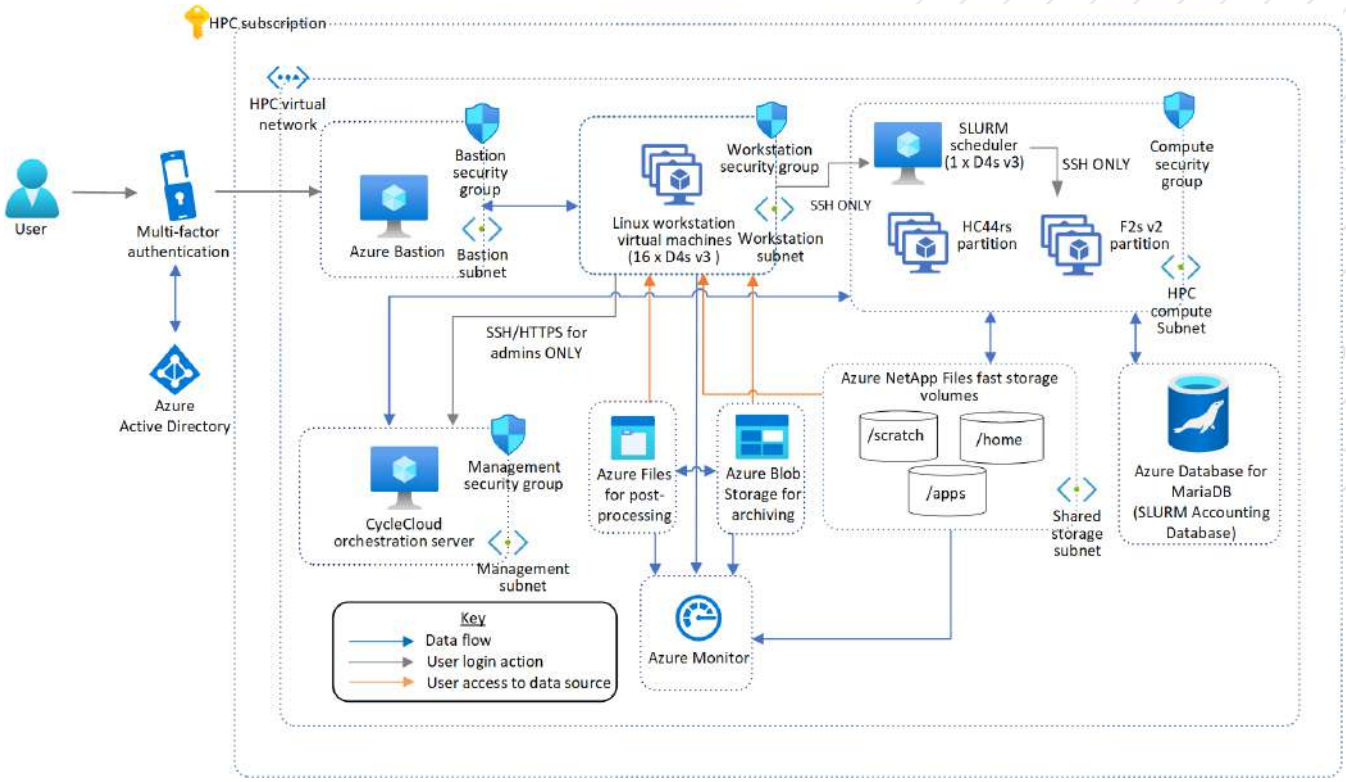

Figure 2: An example LLD schematic for the infrastructure components of a cloud HPC environment

The LLD aims to ensure that the concepts from the HLD are fully implementable using the resources available on the cloud service. It also ensures that all (even non-technical) stakeholders can make a final informed decision as to whether the proposed solution will truly benefit an organisation. If this is judged to be the case, the technical implementation phase can begin.

Implementation

During past projects, we have found that particular situations tend to repeat themselves during the implementation phase and often result in delay. Careful pre-planning as outlined in the following points can avoid this:

- Ensuring that the system implementer has the right security permissions to implement the design.
- Ensuring any relevant software licenses are cloud compatible.
- Access to and installation instructions for any third-party monitoring or security agents are provided to the implementer.
- If the system implementer cannot perform the whitelisting for a firewall, this should be carried out before deployment of the environment.
- Ensuring details of the HPC accounting system are passed to the implementer.

There may of course be more uncertainty in some aspects of system implementation than others which are unique to a project. In this case, it is important to identify these aspects and build in time contingencies to anticipate any delays. On previous projects, we have found that components of the system which require the interaction between

two or more parties to setup predictably leads to delays. This is most common for larger organisations. One common example may be the setup of an authentication system: in many cases this would require the system implementer to work with a security team. Since security is such an important element for any production environment we would strongly recommend a security engineer/architect be assigned for as close to full time as possible for the duration of the project.

If a large variety of software is requested initially, this can also often be a source of delay, especially if performance optimisation, various CPU architectures or different software versions are involved. Tools which can facilitate software building and installation such as Spack or EasyBuild can greatly ease the burden as compared to manual compilation. These tools also facilitate the management of multiple software versions across various architectures. An organisation may also want to focus initially only on key software which is well-optimised rather than all requested. For this, tools such as the Intel OneAPI base and HPC toolkits can greatly facilitate the building and performance optimisation of workflows. [3,4]

Acceptance testing

As far as possible, explicit testing and consolidation periods should be built into the implementation plan to test specific components of the HPC system as they are completed. It could be preferable to design a series of unit tests during the LLD phase. An important example is again software: as a minimum, application test suites should run successfully after installation. On completion of the HPC system implementation (which itself should have involved testing internally) we have previously used and recommend the following testing pattern:

1. Integration testing. This concerns testing by the system implementer. These tests should link directly to the requirements of the HLD and the LLD. The customer and implementer should both play a role in deciding these tests. For example, reports from particular benchmarks and health check tools such as the Intel Cluster Checker[5] could form part of the testing output.[6]

2. Remediation phase 1. In this first remediation phase, the system implementer addresses any problems encountered during the integration testing phase.

3. User acceptance testing. This concerns running a series of pre-planned tests on the system by the users. The time period for these tests should be decided well in advance since this requires the selected test users to design and agree on appropriate tests.

4. Stress testing. This concerns placing the system under the peak levels of stress it is likely to encounter in a production environment. This stage may be included as part of the user testing stage.

5. Remediation phase 2. In this second remediation phase, the system implementer addresses any problems encountered during user acceptance and stress testing.

6. Regression testing. This concerns confirming that the implementer has successfully addressed any previous issues which arose during testing by ensuring that the system can run the entire test suite used previously without any issues. This also establishes performance baselines against which the customer can carry out regression testing and benchmarking when modifications are made to the system (eg the addition of a new compute instance type).

Security considerations

When considering security there are two important concepts to take into account. The first is a 'zero-trust' policy: there should be no element of the system which relies purely on trust in the system users. A basic example of this is shown in Figure 2 where after MFA authorisation is used to gain access to a workstation, SSH access is still

[3] https://software.intel.com/content/www/us/en/develop/tools/oneapi/base-toolkit.html
[4] https://software.intel.com/content/www/us/en/develop/tools/oneapi/hpc-toolkit.html
[5] https://software.intel.com/content/www/us/en/develop/tools/oneapi/components/cluster-checker.html
[6] https://github.com/Azure/azhpc-diagnostics

required to log in to the HPC scheduler node and the compute nodes thereafter. In this particular case, for the most stringent scenario, a password should be associated with the SSH key. This ensures that if unauthorised access is gained to a workstation, access to the HPC cluster is still restricted. This should not be confused with single-sign options (SSO)[7] which streamline the need for constantly entering the same credentials to access various aspects of an environment. There should be good reason for re-authentication without causing inconvenience to users.

The previous example also leads nicely to the second important security concept: 'defence in depth'. This concerns implementing many layers of security in the system so that if one layer is compromised, additional security layers remain. Microsoft Azure possesses a growing variety of tools to tackle various security threats which are partially summarised in Figure 3.

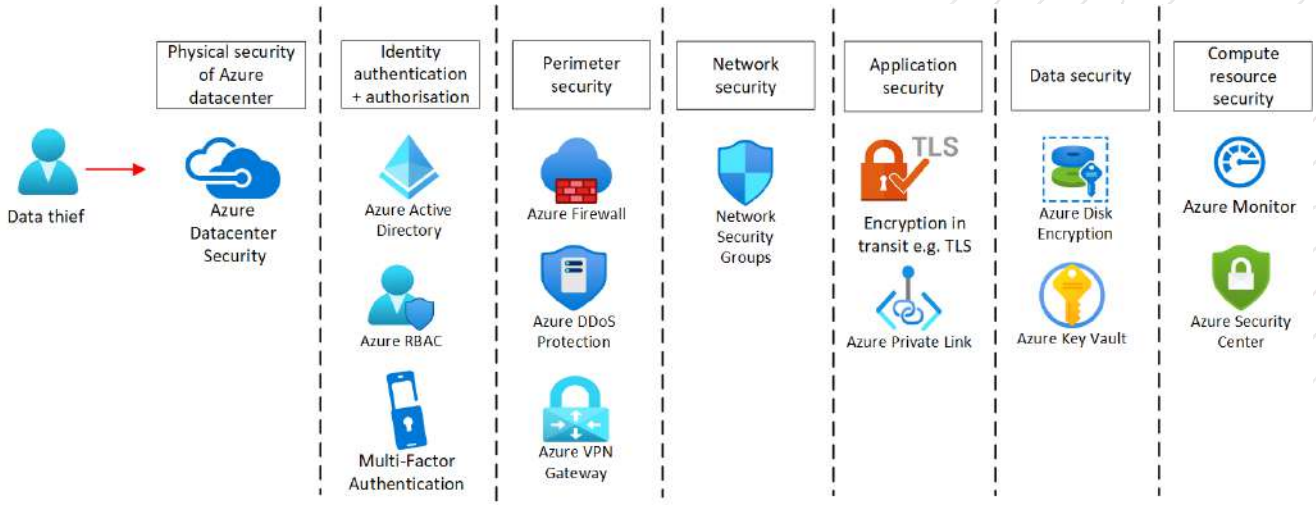

Figure 3: Schematic summarising various security tools on Microsoft Azure

The left-most column in Figure 3 concerns physical datacentre security for which a user of Microsoft Azure has relatively less control but can be assured that a world-leading and comprehensive arrangement is in place.[8]

The second column refers to both user authentication and authorisation. Authentication is concerned with confirming the identity of a user and for this Azure offers the purpose-built Azure Active Directory (AAD) which offers, amongst other tools, in-built multifactor authorisation which is now accepted best practice for authentication.[9] Alternative options such as syncing with an on-premise Active Directory as well as third party options are also available if required by an organisation.[10] Authorisation is concerned with the permissions a user is entitled to when logging in to an environment and users should not be offered any more than required. Azure role-based access control (RBAC) offers a variety of in-built roles specific to many use cases as well as the ability to create custom roles.[11]

Perimeter security refers in the current context to the virtual network layer of a Cloud environment (ie the 'HPC virtual network' in Figure 2). A variety of native and third-party tools are available for this purpose. For example, Azure Firewall can be used to allow access to specific websites by filtering via fully-qualified domain names (FQDNs). A virtual private network (VPN) could firstly offer an additional authentication later together with AAD. An organisation may prefer to use MFA when logging into a company VPN rather than when logging into Azure. Azure VPN Gateway allows an organisation's on-premise environment to connect to Azure via a VPN connection ensuring that all network traffic between the on-premises network and Azure is encrypted. Finally, ensuring protection against cyber attacks such as denial-of-service attacks via tools such as Azure DDoS Protection (DDoS=distributed denial-of-service) helps ensure that such attacks are stopped early at the perimeter layer before they can advance further into a network.

[7] https://docs.microsoft.com/en-us/azure/active-directory/manage-apps/what-is-single-sign-on
[8] https://docs.microsoft.com/en-us/azure/security/fundamentals/physical-security
[9] https://docs.microsoft.com/en-us/azure/active-directory/authentication/concept-authentication-methods
[10] https://docs.microsoft.com/en-us/azure/architecture/reference-architectures/identity/azure-ad
[11] https://docs.microsoft.com/en-us/azure/role-based-access-control/overview

Network security refers in the present context to security at the subnet boundaries within a virtual network. This primarily refers to network security groups (NSGs) in Azure. In many ways NSGs act like a firewall, allowing traffic into the subnet to be filtered based on IP ranges and specific Azure services. An important distinction between NSGs and firewalls is the inability to filter via FQDNs for NSGs.

Application security refers particularly to the encryption of data when in transit and is concerned with the prevention of man-in-the-middle attacks. This can be ensured for example by enforcing SSL communication between the scheduler node and the Azure Database for MariaDB in Figure 2 and enforcing the HTTPS protocol when accessing Azure Storage accounts or the CycleCloud web interface.[12] Azure Private Links ensure that data traffic moves via the Azure backbone network rather than via the public network, therefore minimising the likelihood of attacks.

Data security refers on the other hand to the encryption of data at rest. All Azure VMs make use of disks which come by default with encryption at rest. However, this encryption does not extend to the operating system (OS) level. For this, tools such as Bitlocker for Windows machines and DM-crypt for Linux machines can be used to add an extra layer of protection.[13] Azure Key Vault offers users the ability to securely store secrets, access keys and security certificates. These entities can then be accessed as needed by applications via API calls rather than being included in application code (the latter should be avoided at all costs).

The final column in Figure 3 refers to security monitoring of compute resources in a Cloud environment. Azure Monitor can be used to setup alerts on Azure, for example to warn when a storage volume is filling up unusually quickly or CPU usage is unusually high. Azure Security Centre is a general security management service able to provide security-specific alerts, threat-prevention recommendations and vulnerability assessments.[14] In previous projects customers have often preferred third party monitoring tools which are not available on Azure natively or cannot integrate with Azure as required. Azure has the flexibility to accommodate such tools and often this requires the installation of monitoring agents on individual virtual machines. In this case, we recommend minimising the usage of monitoring tools on the performance-sensitive resources such as HPC cluster nodes. However, such tools are worth considering for example in the Linux workstations in Figure 2.

The use of such third party monitoring tools requires the opening up of the cloud environment so that monitoring agents can report back to a central server. An organisation should consider thoroughly whether this is a security risk it is willing to take. On the other hand, having a homogenous monitoring infrastructure across an organisation rather than an exception for a cloud environment offers an administrative advantage. Another advantage is that by having several different monitoring agents at a time, weaknesses in one solution can be covered by the strengths of another, therefore providing security by diversity. This of course requires opening up the cloud environment further.

Governance considerations

A central concept of governance on the Cloud is the idea of a 'Landing Zone' (LZ). A landing zone provides a fundamental cloud environment configuration on which any additions to the environment can be based. Azure Management Groups are a purpose-built tool for setting up a LZ. A management group can be thought of as a container for a number of subscriptions.[15] Settings in the management group are inherited by its child subscriptions. Azure Blueprints, Azure Policy and Azure RBAC act as tools for configuring a management group as illustrated in Figure 4. In this case the management group has three child subscriptions: a production HPC subscription, a Dev/Test HPC subscription (for testing modifications to the HPC environment without unintentionally affecting users) and an enterprise subscription for business-related tasks.

[12] https://docs.microsoft.com/en-us/azure/mariadb/howto-configure-ssl
[13] https://docs.microsoft.com/en-us/azure/security/fundamentals/azure-disk-encryption-vms-vmss
[14] https://docs.microsoft.com/en-us/azure/security-center/security-center-introduction
[15] https://docs.microsoft.com/en-us/azure/governance/management-groups/overview

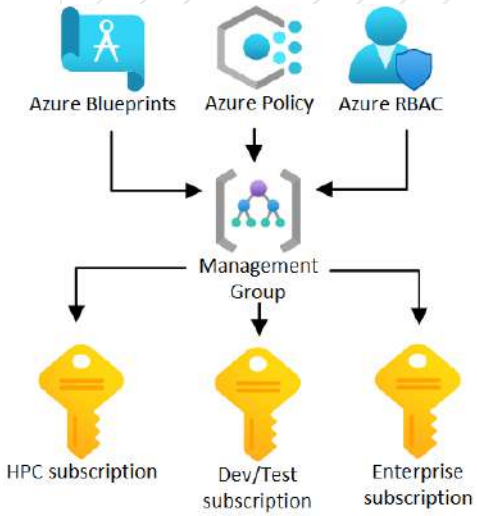

Figure 4: Schematic summarising useful tools for the configuration of an Azure Manangement Group

Azure Policy

Azure Policy allows the creation of a policy template for a cloud environment. One example policy could be that any cloud resources must be deployed in the 'East US' Azure region or that only specific virtual machine SKUs can be permitted on a subscription (correlating with the types chosen to run compute jobs). These particular policies come already built-in to Azure Policy and not only reduce the likelihood of human error but may also help control spending on the Cloud. Many policies can be grouped together to form a 'policy initiative'. A number of built-in policy initiatives are also available such as the ISO 27001 policy initiative.[16] Such initiatives could form the base on top of which customised policies are added such as for resource naming or tagging. [17,18,19]

Example use case

Below is an example two-policy initiative definition in JSON format. The first policy ("Allowed locations_1") ensures that any resources deployed can only be in the EastUS Azure region. The second policy ("Allowed virtual machine size SKUs_1") ensures that only Intel Xeon-processor based HC44rs and D4sv3 VM instances can be deployed. The former SKU type is designed especially for use as HPC compute nodes, whereas the latter has been used in the past by customers as a HPC login node and a virtual machine workstation.

```
[
    {
     "policyDefinitionReferenceId": "Allowed locations_1",
     "policyDefinitionId":
"/providers/Microsoft.Authorization/policyDefinitions/e56962a6-4747-49cd-b67b-
bf8b01975c4c",
     "groupNames": [],
     "parameters": {
      "listOfAllowedLocations": {
       "value": [
        "eastus"
       ]
      }
     }
    },
    {
```

[16] https://docs.microsoft.com/en-us/azure/governance/policy/samples/built-in-initiatives
[17] https://docs.microsoft.com/en-us/azure/cloud-adoption-framework/ready/azure-best-practices/resource-naming
[18] https://docs.microsoft.com/en-us/azure/cloud-adoption-framework/ready/azure-best-practices/resource-tagging
[19] https://docs.microsoft.com/en-us/azure/azure-resource-manager/management/tag-policies

```
    "policyDefinitionReferenceId": "Allowed virtual machine size SKUs_1",
    "policyDefinitionId":
"/providers/Microsoft.Authorization/policyDefinitions/cccc23c7-8427-4f53-ad12-
b6a63eb452b3"
    "groupNames": [],
    "parameters": {
     "listOfAllowedSKUs": {
      "value": [
       "standard_hc44rs",
       "standard_d4s_v3"
    ]
   }
  }
 }
]
```

A policy initiative can be created via the Azure Portal or via the Azure Command Line Interface (CLI) using a JSON definition (as above). In the latter case (which is speedier), the following command would be used:[20]

```
 az policy set-definition create --name example_initiative_name --definitions
"/path/to/example_policy_initiative.json"
```

Azure RBAC

Azure RBAC, as mentioned previously, allows for the precise control of user permissions on cloud resources. For example, the built-in 'Virtual Machine Administrator Login' and 'Virtual Machine User Login' roles allow users to log in to a virtual machine from the Azure Portal either as an administrator or a regular user.[21] These roles are highly applicable to the Linux workstations in Figure 2. As with policies, RBAC roles can also be customised. For example, the two previous login roles could be customised to give specific users the right to switch on/off and restart their corresponding virtual machines from the Azure Portal while giving the administrator rights to all the virtual machines.

Example use case

Below is an example custom RBAC role in JSON format which we developed and used in a previous customer engagement which combines the built-in 'Virtual Machine User Login' role with the ability to start, restart and deallocate virtual machines, With this role, in the case of the Linux workstations in Figure 2: An example LLD schematic for the infrastructure components of a cloud HPC environment, users of the system could independently manipulate an individually assigned virtual machine.

```
{
        "Name": "Virtual Machine User",
        "Description": "Can deallocate and start virtual machines",
        "AssignableScopes": [
            "/subscriptions/d57b176e-d3ad-46c0-8f9d-33ab0cd2af37"
        ],
        "Actions": [
                    "Microsoft.Compute/virtualMachines/start/action",
                    "Microsoft.Compute/virtualMachines/deallocate/action",
                    "Microsoft.Network/publicIPAddresses/read",
                    "Microsoft.Network/virtualNetworks/read",
                    "Microsoft.Network/loadBalancers/read",
                    "Microsoft.Network/networkInterfaces/read",
                    "Microsoft.Compute/virtualMachines/*/read",
                    "Microsoft.Compute/virtualMachines/restart/action"
         ],
         "NotActions": [],
         "DataActions": [
                    "Microsoft.Compute/virtualMachines/login/action"
         ],
         "NotDataActions": []
```

[20] https://docs.microsoft.com/en-us/cli/azure/
[21] https://docs.microsoft.com/en-us/azure/role-based-access-control/built-in-roles#compute

```
}
```

As was the case for the policy initiative, Azure CLI could be used to create the custom role from this file using the following command:

```
az role definition create --role-definition
/path/to/example_custom_role_definition.json
```

Table 1: Summary of some useful RBAC roles to consider in a HPC environment below summarises some useful RBAC roles to bear in mind for users of a system such as that in Figure 2: An example LLD schematic for the infrastructure components of a cloud HPC environment The 'Scope' column describes the resource to which the RBAC role should be applied. Further information about various options for storage on Azure in the context of HPC is given in the next section.

RBAC role	Description	Scope
Virtual Machine User	A custom role used in previous customer projects allowing users to log in to particular workstations and start, deallocate and restart virtual machines.	Virtual machine to be used as a workstation
Reader	Gives a user basic read access to an Azure resource.	Storage account. In order for users to make use of the last two roles in this table via the Azure Portal (or other GUI tools), they need Reader access to a storage account.
Storage Blob Data Contributor	Allows users to manipulate data in an Azure Blob Storage container.	Azure Blob Storage container
Storage File Data SMB Share Contributor	Allows users to manipulate data in an Azure File Share quota.	Azure Files quota

Table 1: Summary of some useful RBAC roles to consider in a HPC environment

Azure Blueprints

Finally, Azure Blueprints allow for the creation of a standard template for an Azure subscription.[22] In the most complete sense, an Azure Blueprint could consist of a series of well-defined resource groups with resources which are subject to a particular Azure Policy initiative and Azure RBAC roles. With reference to Figure 4, the same basic Azure Blueprint could be used for the HPC and Dev/Test subscriptions to create two separate but similar HPC environments on Azure which follow specific organisational standards. A standard deployment like this also reduces the likelihood of errors when compared to each environment being created manually. As with Azure Policy and RBAC roles, a blueprint can be represented in JSON format and deployed via the Azure CLI. A blueprint consists of a central JSON file which references 'artifact' JSON files which describe various resources. Microsoft has a large library of blueprints available on GitHub.[23] Of particular interest and a good starting point may be the ISO27001 blueprint which comes with built-in policies to satisfy ISO27001 requirements.[24]

Any Azure Blueprints, Policies and RBAC roles configured in an Azure Management Group are inherited by its child subscriptions. To add another layer of abstraction, management groups can also be subject to a parent management group and have child management groups.

[22] https://docs.microsoft.com/en-us/azure/governance/blueprints/overview
[23] https://github.com/Azure/azure-blueprints/tree/master/samples
[24] https://github.com/Azure/azure-blueprints/tree/master/samples/001-builtins/iso_27001

Storage considerations

Typically, as illustrated in Figure 5: Schematic comparing data life cycle of cloud storage vs on-premises storage in a HPC environment, an on-premises storage system is hierarchical and consists of parallel file system with an solid state drive storage (SSD) and hard disk drive storage (HDD) tiers and an archive storage tier (tape-based). Backups of a tape archive would need to be performed manually while other storage tiering can be automated with a hierarchical storage management (HSM) solution. High performance storage on the Cloud is typically an order of magnitude more expensive to obtain similar performance. However, far more variety in storage services is available on the Cloud offering a trade-off between price and performance. Backup is provided in all cases, with more comprehensive options available at increasing cost. It is important to make use of each storage service effectively to minimise cost. As alluded to in Figure 5: Schematic comparing data life cycle of cloud storage vs on-premises storage in a HPC environment and discussed in the following sub-sections, we would recommend three main levels of storage in a cloud HPC environment: high performance storage, fast file storage for data processing and object storage for archiving.

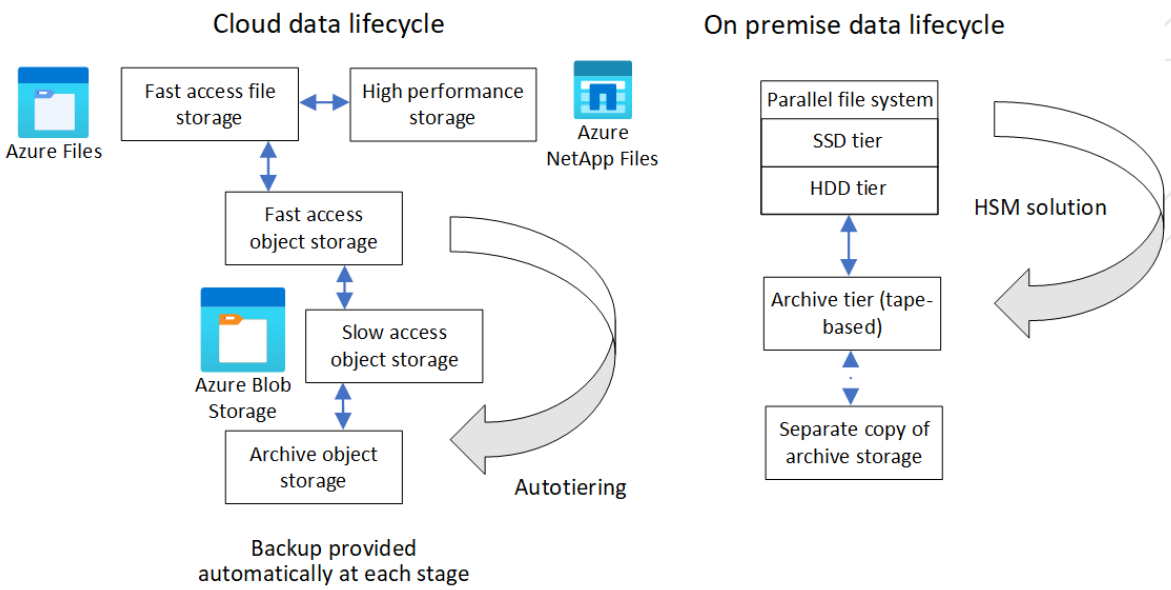

Figure 5: Schematic comparing data life cycle of cloud storage vs on-premises storage in a HPC environment

High-performance storage

At the time of writing, the most commonly recommended high-performance storage system for HPC workloads on Microsoft Azure is the Azure NetApp Files (ANF) service. This service comes in three access tiers of varying performance as shown in Table 2, with pricing correct as of May 2021 for the East US Azure region.

Tier	Performance/TiB	Storage cost/month
Standard	16 MiB/s	$0.14746/GiB
Premium	64 MiB/s	$0.29419/GiB
Ultra	128 MiB/s	$0.39274/GiB

Table 2: Summary of Azure NetApp Files tiers

The ANF service cost model is straightforward with charges such as ingress and egress included in the prices in Table 2. As can be seen, the cost to performance ratio decreases moving from the least performant Standard tier to the most performant Ultra tier. The exact tier chosen depends on the I/O requirement of the HPC workloads to be carried out and we recommend the Premium tier to start with. An ANF deployment can be broken down into

units called 'capacity pools' with a size determined by the user. The capacity pools themselves consist of volumes of varying size whose combined size cannot exceed that of the capacity pool. These volumes can be mounted separately as network file share (NFS) or server message block (SMB) volumes.

The total available throughput of an ANF capacity pool can be calculated by multiplying the performance of the chosen tier in Table 2 by the chosen capacity pool size. For example, if a capacity pool size of 20 TiB is chosen for the Premium tier, the total available throughput would be 1.28 GiB/s. This total available throughput is by default then divided proportionally across the various ANF volumes within the capacity pool.

In Figure 2, three ANF volumes have been created: /scratch for usage by HPC workloads; /home for user home directories and /apps for the storage of applications. This is the typical usage pattern we recommend for a cloud HPC environment. If /scratch is designated 15 TiB of the above 20 TiB capacity pool, then by default the maximum throughput of the scratch volume will be 75 % of 1.28 GiB/s, which works out to be 0.96 GiB/s. Typically, since the /scratch volume is responsible for the HPC workload, the bulk of the capacity pool is dedicated to it. The ANF service also provides an option to divide available throughput between each of the volumes in a capacity pool arbitrarily rather than according to volume size.[25]

The effective functioning of an ANF volume will be dependent on three general factors:

1. The number of users of the HPC environment and their typical home directory storage footprint.

2. The nature of workloads. This would include the number of concurrent jobs expected to run at any time and I/O patterns (IOPS vs bandwidth). Some workloads have a larger working set at runtime than others and just looking at completed job statistics doesn't tell the whole story.

3. The sizes of inputs and outputs of workloads: a proportion of throughput will be used to move data on, off and across volumes.

In the context of higher education, these factors will be unique for each department and groups within it and highly dependent on types of research activity and the number of users requiring a HPC service. There is therefore no universal rule for determining the correct size for capacity pools and volumes and associated throughput. Forensic evidence of throughput on-premises would be a useful starting point in this respect for initially determining volume size and throughput. However, due to the cost of keeping data on an ANF volume, the need for cost optimisation may require data movement from ANF to a cheaper storage service, which may result in a higher rate of data movement on and off volumes than on-premises. Furthermore, the total budget available will ultimately limit the maximum performance achieved and will partially depend on the cost per percentage improvement in performance.

In our experience, particularly in higher education establishments, the available throughput from a typical HPC file system is rarely fully utilised, and so although the price per performance in Table 2 may seem comparatively high, comparable performance is achievable in many cases if only the required throughput is provisioned. Cloud costs can further be reduced by moving the output data from an HPC workload to a lower cost storage service for pre- and post-processing. Presently this is even more important since the ANF service does not provide cost-effective data backup options beyond locally redundant snapshots (ie data is only backed up within a single datacentre).[26]

Finally, it is worth mentioning that ANF is not a conventional high performance parallel file system such as Lustre or BeeGFS although it is possible to deploy the latter on Azure. A more cost-effective option is BeeOND, which makes use of the ephemeral disks provided with compute nodes themselves at no extra cost.[27] However, data stored on the ephemeral disks must be removed before the compute node is deprovisioned.

[25] https://docs.microsoft.com/en-us/azure/azure-netapp-files/manual-qos-capacity-pool-introduction
[26] https://docs.microsoft.com/en-us/azure/azure-netapp-files/snapshots-introduction
[27] https://techcommunity.microsoft.com/t5/azure-global/tuning-beegfs-and-beeond-on-azure-for-specific-i-o-patterns/ba-p/1015446

Storage for data processing

The Azure Files service provides a less performant but more cost-effective approach to carrying out routine pre- and post-processing tasks for a cloud HPC environment. This is a fully managed cloud-native file system which can be accessed with common protocols such as NFS and SMB. In this case four access options are available as summarised in Table 3. Pricing is correct as of May 2021 for the East US Azure region, assumes a General Purpose Version 2 (GPv2) storage account is used with the locally-redundant storage option (LRS-data is backed up only within a single datacentre) and assumes file shares less than 5 TiB in size. Pricing in the present and next sections does not account for reserved storage options which come at a discounted price. Such options will be discussed in a later section.

Access option	Performance	Storage cost/month
Premium	Up to 60 MiB/s + 0.06 * (provisioned GiB) egress Up to 40 MiB/s + 0.04 * (provisioned GiB) ingress	$0.16/provisioned GiB
Standard Transaction-optimised	Up to 60 MiB/s	$0.06/GiB
Standard Hot	Up to 60 MiB/s	$0.0287/GiB
Standard Cool	Up to 60 MiB/s	$0.0228/GiB

Table 3: Summary of Azure Files access options

The Premium access option is the only one which makes use of solid state drives (SSDs) and provides the highest throughput. It also offers the lowest latency. However, storage must be provisioned in advance and unused storage must be paid for. Furthermore, the maximum throughput cannot be fixed as is the case with ANF and will not consistently reach its highest value. This makes Premium a less useful option for carrying out heavily I/O dependent HPC workloads on the Azure Files service.

The Standard access options make use of hard disk drives (HDD) with decreasing latency moving from the Transaction-optimised to Cool options. Since storage isn't pre-provisioned, the storage charges are based on used capacity. The Cool option is generally recommended for archival usage and we would recommend the Hot access option as a starting point for processing. If any of the Standard options are chosen, a user can switch between them to fine tune performance; however, this is not the case if the Premium option is chosen.

Unlike with the ANF service the prices in Table 3 are storage costs only and do not account for other operations such as transactions (eg read, write, list) and data retrieval. In general, moving from Premium to Standard Cool, cost per transaction and data retrieved per GiB increases. There is also a per GiB file metadata storage cost for the Hot and Cool options. Furthermore, Table 3 assumes the LRS storage option. More comprehensive redundancy options increase storage cost.[28] During cost modelling careful attention should be paid to these extra charges.[29]

A typical use of Azure Files on past customer projects has been to assign each user an individual Azure Files storage quota. Users would then move files between this quota and ANF before/after running workloads. The storage quota to be assigned to each user may initially be based on typical space used on-premises. A perfectly reasonable starting point used by previous customers is 0.5-1 TiB.

Archival storage

In the present context, the Azure Blob service is an excellent object storage option for archiving data which is expected to be rarely used. Unlike Azure Files, it lacks true file system support and does not offer the same operations as a POSIX file system which users will typically be most accustomed to. It is therefore not suitable for

[28] https://docs.microsoft.com/en-us/azure/storage/common/storage-redundancy
[29] https://azure.microsoft.com/en-gb/pricing/details/storage/files/

data pre- and post-processing activities. Access options follow a similar pattern to Azure Files as summarised in Table 4. Pricing is correct as of May 2021 for the East US Azure region and assumes block blob storage in a General purpose version 2 (GPv2) storage account is used with LRS.

The Premium and Standard access options are again distinguished by SSDs and HDDs and the storage costs do not account for other costs as discussed in the previous section (minus metadata costs since no file system layer is used). Unlike with Azure Files, both Premium and Standard access options charge for used capacity and no storage pre-provisioning is required. The blob storage costs in Table 4 are in all cases lower than those for Azure Files in Table 3. Some important distinctions can be made between Hot, Cool and Archive tiers here and the supposed equivalents in Table 3. Firstly, for blob storage, data stored in the Cool and Archive tiers must be stored for at least 30 and 180 days, respectively.[30] Secondly, movement between the Hot, Cool and Archive tiers in blob storage can be automated so that data which is rarely used is moved to lower cost storage tiers and frequently used data remains in fast-access storage tiers.[31] This can be done via the Azure Portal by accessing the 'Lifecycle management' tab when viewing a storage account as shown in Figure 6: Screenshot showing 'Lifecycle management' tab for an Azure Storage account and then following the prompts. This helps ensure cost optimisation.

Access option	Performance	Storage cost/month
Premium	Up to 6.25 GB/s egress Up to 1.25 GB/s ingress	$0.15/GB
Standard Hot	Up to 6.25 GB/s egress Up to 1.25 GB/s ingress	First 50 TB: $0.0208/GB Next 450 TB: $0.0200/GB Over 500 TB: $0.0192/GB
Standard Cool	Up to 6.25 GB/s egress Up to 1.25 GB/s ingress	$0.0152/GB
Standard Archive	Up to 6.25 GB/s egress Up to 1.25 GB/s ingress	$0.00099/GB

Table 4: Summary of Azure Blob Storage access options

Although not offering many advantages of a true file system, Azure Blob Storage can be accessed from a command shell via the Azure Command Line Interface (CLI) with blob-level authorisation implemented via Azure RBAC. [32] This way files can at least be moved from and to user-specific Blob storage with command shell commands assuming Azure CLI is installed on a user's shell environment.[33] This can be facilitated with the workstation setup in Figure 2. Here each workstation is connected to ANF, Azure Files and Azure Blob Storage. This allows users to take HPC workload outputs from ANF to Azure Files for processing and then move them to Azure Blob Storage for archiving via the Linux command shell. Alternatively, Azure Storage Explorer provides a graphical user interface (GUI) equivalent.[34]

Files in Azure Blob Storage could initially be set to the Hot tier and made to move the Cool and Archive tiers after certain periods of disuse. It should be noted that once a file enters the archive tier it would need to be manually 'rehydrated' back to the Cool tier before access which can take up to 15 hours and comes with an additional cost.[35] In one example customer engagement, users would send data to Azure Blob storage's Hot tier when they deemed it necessary, and the data was moved to Cold tier after 30 days. The customer decided to not make use of the Archive tier due to the need to get quick access.

[30] https://docs.microsoft.com/en-us/azure/storage/blobs/storage-blob-storage-tiers
[31] https://docs.microsoft.com/en-us/azure/storage/blobs/storage-lifecycle-management-concepts?tabs=azure-portal
[32] https://docs.microsoft.com/en-us/azure/storage/blobs/authorize-data-operations-cli
[33] https://docs.microsoft.com/en-us/cli/azure/install-azure-cli
[34] https://azure.microsoft.com/en-gb/features/storage-explorer/
[35] https://docs.microsoft.com/en-us/azure/storage/blobs/storage-blob-rehydration?tabs=azure-portal

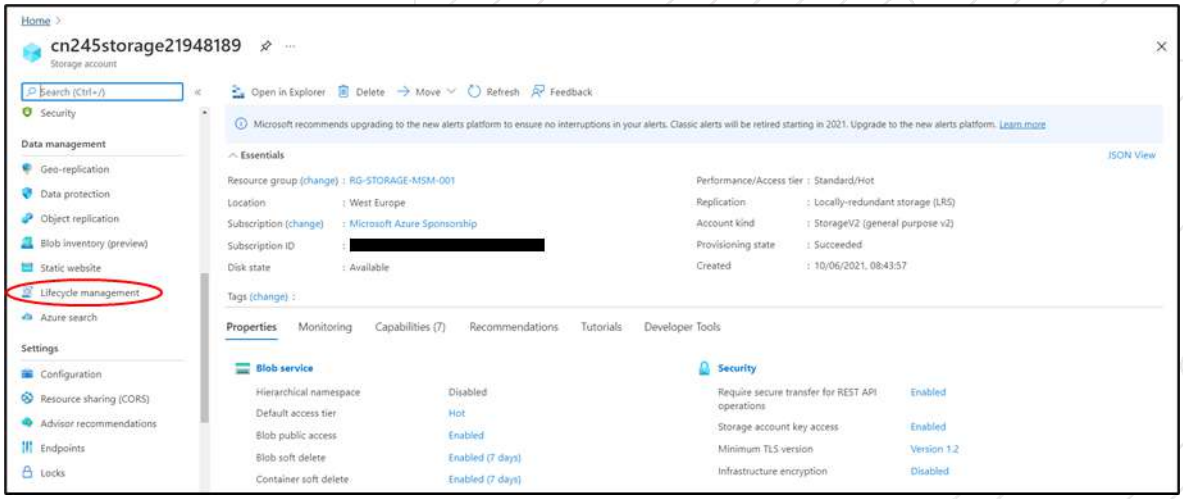

Figure 6: Screenshot showing 'Lifecycle management' tab for an Azure Storage account

Example use case

To end this section Table 5 below summarises the monthly costs of one potential storage setup on Microsoft Azure for a HPC environment. In this setup, three year reserved storage capacity is purchased where possible to minimise costs and zone-redundant storage (ZRS) is chosen to ensure that data backups are available in separate sites. The general workflow (which reflects a real workflow used by a previous customer) would consist of running HPC workloads on ANF (automatic throughput provisioning), transferring the workloads to Azure Files for processing, and then transferring to the Azure Blob Hot storage tier for 30 days before archiving in the Cold tier for quick access. It should be noted that other costs (eg storage metadata for Azure Files and egress costs are note considered here and would be context specific). Costs are correct of May 2021 and assume the East US region.

Storage Option	Storage SKU	Storage cost/month
ANF 3 TiB /apps volume	Premium	$903.75
ANF 5 TiB /home volume	Premium	$1,506.25
ANF 25 TiB /scratch volume	Premium	$7,531.26
Azure Files 50 TB	GPv2 Hot, ZRS	$1,216.53
Azure Blob Storage 100 TB	GPv2, Hot, ZRS	$1,758.00
Azure Blob Storage 100 TB	GPv2, Cold, ZRS	$1,285.00
Total		$14,200.29

Table 5: Cost calculation of an example HPC storage architecture

It is immediately apparent that the largest percentage of cost relates to ANF ie the high performance storage aspect of the architecture followed by Azure Files. This table highlights that storage requirements and cost management are of utmost importance.

User-experience

While designing a system it is often easy to focus too strongly on security needs and cost optimisation, resulting in a system which causes inconvenience to users and by implication lowers productivity. Here we consider three important points to account for when considering user experience which have tended to repeat themselves during previous customer engagements.

1. Cloud-based user workstations for processing data in the HPC environment. Another term for these workstations is 'virtual desktop infrastructure' (VDI). Such a setup gives users peace of mind in knowing that their data is backed up and stored securely. Furthermore, data need not be carried around in individual physical devices which can increase the likelihood of data loss and theft. Another advantage to such a setup is that egress charges can be avoided to a certain degree. The transfer of data out of an Azure datacentre is chargeable (transfer in is free), whereas by employing a VDI within the same region as an Azure Storage resource and using a private endpoint for access can ensure that data movement and resulting egress charges are minimised.[36] However, in some use-cases it may be more cost efficient to not make use of VDIs due to their inherent cost despite a reduction in data transfer costs and have users login straight to a HPC login node.

2. Over-intrusive security environments. Firstly, the overuse of monitoring agents on workstations and compute resources can reduce performance. As mentioned before, we recommend minimising this on compute resources and similar considerations should be given to VDIs. Secondly, in a higher education environment, users will tend to require access to certain web resources (eg code repositories and journal articles) which may not necessarily be permitted in other environments. If a firewall is used, care should be taken to allow for this to avoid inconvenience. Finally, as mentioned in the security section, constant re-authentication using the same credentials should be kept to a minimum by implementing SSO where possible.

3. Scaling limitations. Unlimited scaling is often cited as one of the biggest advantages of HPC in the Cloud. However, if the virtual network deployed on Azure lacks the necessary amount of IP addresses, then scaling will be limited by this leading to unnecessary delay for users. Therefore, in designing a system, carefully consider capacity requirements and adjust the size of the virtual network to account for this: by default, a single IP address will be required per compute resource when using Azure CycleCloud.

Automation

Virtualising a HPC environment comes with the advantage of infrastructure and configuration-as-code. This is advantageous not only in repeatedly deploying a system but also for routine system administration. For example, tasks such as updating VM images and on- and off-boarding users from the HPC environment can all be automated.

An example of infrastructure-as-code (IaC) has been alluded to in the 'Governance considerations' section in the form of Azure Blueprints. In this case required resources, policies and RBAC roles can be integrated into a blueprint. Azure also provides purpose built Powershell- and BASH-based (Azure CLI) command line interfaces for resource deployment as an alternative, with examples of syntax already shown in the 'Governance considerations' for the latter. With these methods initial time will need to be invested in the development of provisioning scripts but this will pay off in the long term and eventually allow automated deployments. In addition, if an organisation is working in a broader cloud context with many providers, cloud neutral deployment options may be of interest. To name but a few, popular infrastructure-as-code options may include Terraform, Pulumi and various software development kits available supporting many common programming languages.[37]

Automation can also form part of a disaster recovery strategy. Although comprehensive storage redundancy backup options are available in Azure as mentioned in the 'Storage Considerations' sections, Azure CycleCloud doesn't currently provide an explicit disaster recovery option. However, by having a tested automated deployment, a new HPC environment can be quickly brought up in another Azure region to help minimise the impact on productivity.

Once infrastructure is deployed, additional configuration will often be required both in the case of VDIs and compute VMs. In the case of the VDIs a useful native option is Azure custom script extension for virtual machines.[38] Cloud neutral options are also available including Ansible, Chef and Puppet. Tools such as Packer are useful for the automation of custom Azure VM image creation.[39]

[36] https://azure.microsoft.com/en-us/pricing/details/bandwidth/
[37] https://azure.github.io/azure-sdk/
[38] https://docs.microsoft.com/en-us/azure/virtual-machines/extensions/custom-script-linux
[39] https://docs.microsoft.com/en-us/azure/virtual-machines/linux/build-image-with-packer

In the case of compute VMs, Azure CycleCloud allows the further configuration of newly provisioned nodes via Cloud Init and Cluster Init scripts.[40,41] Cloud Init scripts run before any other CycleCloud configuration scripts run on the node, whereas Cluster Init scripts run after the default CycleCloud configuration scripts.

One easily overlooked example for Cluster Init is for the SLURM scheduler and its use of prologue and epilogue scripts. It is recommended on Azure that before starting a compute job the Azure Linux Agent is temporarily stopped via the '`systemctl stop waagent`' command and caches are cleared via the '`echo 3 > /proc/sys/vm/drop_caches`' command.[42,43] After the job, the Linux Agent should be restarted using '`systemctl start waagent`'. These prologue and epilogue scripts could be created via Cluster Init when the master node is configured.

In general, although configuring compute VMs has its uses, we recommend this is minimised such that any pre-configuration is implemented into the image used by Azure VMs. This can greatly reduce node provisioning time, therefore allowing for a more pleasant user experience and cost reduction (see following section).

With such a wide variety of tools available with various strengths and weaknesses, an organisation may prefer an approach where multiple tools are used for various aspects of the HPC environment. In this case Azure DevOps pipelines offer a way of combining all the mentioned tools together to form a continuous pipeline which uses scripts stored in a code repository. The end result is a centrally management platform from which any aspect of a HPC environment can be managed with just a few clicks, even allowing a complete one-click deployment if the design permits. Azure offers its own native repository platform and popular options such as GitHub and BitBucket are also available.[44]

Finally, the Azure Automation service has not yet been mentioned. This service provides a way of running routine tasks on infrastructure automatically without user intervention. Use cases may include installing updates on VMs regularly or ensuring idle virtual machines are switched off at specific times in order to save on cost.[45]

Example use case

To end this section, below is an example Azure CLI script which automates the creation of the database component of the HPC environment in Figure 2: An example LLD schematic for the infrastructure components of a cloud HPC environment After setting some environmental variables (note the example naming convention used), a resource group is created. This is followed by assigning sensitive values (the administrator username and password) which are obtained from an Azure Key Vault to two additional environmental variables: it is important for obvious reasons to never reveal sensitive information in plain text in such scripts. Next, the database server is deployed and here it is important to highlight the use of SSL enforcement in production environments as well as consider backup requirements. The 'geo-redundant-background' ensures that backup is available LRS. Finally, the MariaDB is opened up for access from Azure services: this is particularly important for the CycleCloud scheduler node to be able to access the database.

```
#!/bin/bash

# Define some variables to use later in the script

Location="eastus"
AccountingResourceGroup="RG-HPC-ACCOUNTING-001"
KeyVaultName="KV-HPC-001"
MariaDBServerName="MDB-HPC-ACCOUNTING-001"

# Create resource group

az group create \
    --name $AccountingResourceGroup \
    --location $Location
```

[40] https://docs.microsoft.com/en-us/azure/cyclecloud/how-to/cloud-init?view=cyclecloud-8
[41] https://docs.microsoft.com/en-us/azure/cyclecloud/how-to/projects?view=cyclecloud-8
[42] https://docs.microsoft.com/en-us/azure/virtual-machines/workloads/hpc/configure
[43] https://docs.microsoft.com/en-us/azure/virtual-machines/extensions/agent-linux
[44] https://docs.microsoft.com/en-us/azure/devops/pipelines/repos/?view=azure-devops
[45] https://azure.microsoft.com/en-gb/services/automation/

```
# Obtain secret variables from Azure Key Vault
# Assuming database admin username is a secret called 'DBAdminUsername'
# Assuming database admin password is a secred called 'DBAdminPassword'

DatabaseAdminUsername=$(az keyvault secret show --name DBAdminUsername --vault-name
KV-HPC-001 --query 'value' -o tsv)
DatabasePassword=$(az keyvault secret show --name DBAdminPassword --vault-name KV-HPC-
001 --query 'value' -o tsv)

# Create MariaDB server for SLURM accounting

az mariadb server create \
    --name "$MariaDBServerName" \
    --resource-group "$AccountingResourceGroup" \
    --location "$Location" \
    --admin-user "$DatabaseAdminUsername" \
    --admin-password "$DatabasePassword" \
    --auto-grow Enabled \
    --geo-redundant-backup enabled \
    --ssl-enforcement enabled \ # enforce SSL connection only
    --sku-name GP_Gen5_4 # SKU of database

# Allow trusted Microsoft services to access MariaDB.
# This ensures Azure resources (such as the HPC scheduler node) can access the
database.

az mariadb server firewall-rule create \
    --resource-group "$AccountingResourceGroup" \
    --server-name "$MariaDBServerName" \
    --name "allowAzureResources" \
    --start-ip-address 0.0.0.0 \
    --end-ip-address 0.0.0.0
```

Financial considerations

In creating an HPC environment on the Cloud, there are a number of financial considerations which are easily overlooked. The first has been mentioned previously and is the use of reserved VM instances and storage to save on cost. At the time of writing, cost savings of up to 72 % and 38 % can be achieved for VMs and storage compared to standard pay-as-you-go (PAYG) pricing, respectively. In the latter case this does not include the ANF file system. Reservations can be made with commitments of varying numbers of years, with the largest cost savings achieved for a longer reservation period. In the context of an Azure CycleCloud-based HPC environment, it would certainly be worth reserving VMs for use as CycleCloud orchestration servers and login nodes as well as any VDIs, all of which have been a standard customer choice in previous engagements. A minimum amount of storage space could also be reserved based on expectations from the current on-premise environment.

The first two rows of Table 6: Cost comparison of running three (Linux-based) D4s v3 VMs below show a cost comparison for PAYG vs 1 year reserved instances (as of June 2021) when using three Linux-based D4s v3 VMs continuously for a year. The intention here is that one VM is used as the CycleCloud orchestration server and the others as HPC login nodes. In this case the cost benefit of reserved VM instances is apparent. In some cases the choice may not be so clear cut and it is worth weighing the benefits of having a minimum number of reserved instances while using PAYG for others as is the case for the third row of Table 6: Cost comparison of running three (Linux-based) D4s v3 VMs. In this case, it is assumed that only one HPC login node is needed continuously for most of the year, with an additional node being added for three months where a large increase in users is expected. It is therefore now favourable to reserve only two of the VMs while using the other via PAYG for three months rather than reserving all three VMs for a year. The important point here is that reserved VMs need to be paid for regardless of whether they are used or not.

Payment model	Annual Cost	Percentage saving compared to PAYG
PAYG for all VMs used continuously for 1 year	$5045.76	-
1 year reserved for all VMs used continuously all year	$3009.06	40%
PAYG for one VM used only for 3 months, reserved for the other VMs used continuously all year	$2426.52	52%

Table 6: Cost comparison of running three (Linux-based) D4s v3 VMs

In a higher education environment, an HPC system may be free at the point of use or use a chargeback model (or indeed a combination of the two). For a chargeback model, the accounting structure typically consists of users assigned to various projects and research groups which are charged accordingly and given an initial accounting budget. However, on Azure CycleCloud using the example of the SLURM scheduler, although accounting can be setup and charged for the duration of a compute job, the cost relating to the time for provisioning and deprovisioning a VM cannot be accounted for. The end result is each compute job costs a little more than can be calculated from purely the duration of the job. This is something which needs to be considered when setting up an accounting system. We recommend setting aside 20% of the compute budget to account for this and then adjusting overtime by comparing spending as calculated via the scheduler accounting system against the actual spend on Azure. The end figure will depend heavily on the types of compute jobs run on the HPC cluster and the available compute node capacity. For example, jobs requiring a very large number of nodes will typically take a longer time to provision, resulting in higher costs, whereas fewer nodes can be provisioned faster.

Azure also offers Spot pricing which adds further complexity to accounting considerations.[46] With Spot pricing unused Azure compute capacity can be accessed at discounts of up to 90% compared to PAYG pricing.[47] The price is charged by the second at the price being charged at that moment in time for that instance type. Spot pricing can fluctuate depending on available capacity, types of VM and region. Nodes available via Spot can be taken away or 'evicted' if deemed necessary by Azure, making them more suitable for short jobs or software which provides a job restart option. Furthermore, even if the bid price is equivalent to the pay-as-you-go (PAYG) price, a user using the PAYG option will always be treated preferentially.

Therefore, if an organisation is considering heavy use of Spot pricing, it is inevitable that at least some compute jobs will be lost via eviction. If the jobs can be restarted they will be more expensive to run since nodes must be re-provisioned, and if a job cannot be restarted, it must be re-run from the beginning. Although the likelihood of this can be reduced by educating users on when to use Spot, it is recommended that a Spot budget is considered to account for the cost penalty in re-running evicted jobs. This will again depend on the types of compute jobs and software being run on the HPC cluster. Despite Spot eviction, relatively early access to the latest compute node technology on the Cloud for these reduced prices is still likely to be cost effective and benefit time to solution when compared to typically available on-premises resources.

In the higher education context, Spot pricing may be more applicable to 'blue sky' research as opposed to critical workloads being performed with the intention of publication. Table 7: Comparison between PAYG vs Spot pricing of running a 60 minute workload on ten HC44rs VMs compares the cost of using 10 Intel HC44rs VM instances for a 60 minute workload in the East US region via PAYG and Spot pricing as of June 2021. PAYG pricing comes at more than three times that of Spot: this may indeed be judged unnecessary for a 'blue sky' workload but necessary for a critical workload with an approaching deadline.

Payment model	Cost for 60 minute workload	Percentage saving compared to PAYG
PAYG	$31.68	-
Spot	$9.97	69%

[46] https://docs.microsoft.com/en-us/azure/cyclecloud/how-to/use-spot-instances?view=cyclecloud-8
[47] https://azure.microsoft.com/en-gb/pricing/spot/

Table 7: Comparison between PAYG vs Spot pricing of running a 60 minute workload on ten HC44rs VMs

Next steps

Once a production system has been implemented and tested, the next recommended step is a short transitional period in which the system is observed with a limited number of users. This allows time for any longer term issues which may for one reason or another not arise during acceptance testing to be identified and dealt with. After this the system is ready to 'go live'.

As is the case with on-premises HPC environments, technological improvements are constantly being made which warrant updating of the HPC environment in order to remain competitive. Since hardware does not need to be purchased on the Cloud, updating an HPC environment is not nearly as time consuming. For example, adding a new VM type as a compute node simply requires an alteration of the relevant Azure CycleCloud template (and any policy which limits Azure VM types which can be used). Some changes may require more time like the decision to use a new storage service offering. In such cases, with an effective deployment automation strategy, a Dev/Test HPC environment can be used for testing without affecting user experience. Due to the relatively fast rate of evolution on the Cloud, we recommend always having a Dev/Test environment to hand.

As is likely to have become apparent during the course of these white papers, although the Cloud offers numerous benefits with respect to HPC environments, there are a host of ways to design such an environment using any number of services, many of which are subtly different. As a result, the migration process can be just as challenging as for a new on-premises system procurement. If your organisation requires assistance in any stage in the process, please do not hesitate to reach out to Microsoft and its partners.

About Red Oak Consulting

Red Oak Consulting's core business is high performance computing (HPC) consultancy. With a reputation for expert knowledge combined with real-world experience, we've delivered strategy, given procurement advice and helped to install more than 30 on-premises HPC systems (including several in the Top 10 of the Top500). To date, we have also completed more than 20 cloud-based projects, bringing first-class technical skills to working with 'early adopter' customers in both the higher education and commercial sectors.

All three white papers can be found at https://www.redoakconsulting.co.uk/microsoftazurehpcwhitepapers2021.