TECHNICAL GUIDE

Best Practices for Deploying Perforce Helix Core on Microsoft Azure

Introduction

A diverse range of organizations can significantly enhance their product development lifecycle and time to market by taking advantage of the near infinite compute, storage, and network resources available on Microsoft. This technical guide provides recommendations for implementing and optimizing development workflows on Azure by describing architectural components. This guide is based on the knowledge and experience of senior consultants at both Perforce Software and Microsoft Azure, as well as Perforce customers operating on Azure infrastructure. Please send feedback to consulting@perforce.com.

Contents

This document is intended to provide guidance for a standard deployment of Perforce Helix Core servers in the Azure cloud environment. A sample topology is illustrated, which is suitable for the demanding needs of large-scale software development. The concepts and details of mechanics are generally applicable to a wide range of Perforce workloads on Azure.

This guide presumes some familiarity with Perforce terminology ("edge servers," "replicas", etc.), and Azure terminology ("VM instances", "Virtual Networks," etc.). References are also made to the Perforce Server Deployment Package (SDP), which is introduced in this document.

Azure Deployment Topologies

In this document, both hybrid (cloud and on-premises) and exclsive Azure topologies are explore. For many clients, including game development and virtual production studios, software development entails versioning large numbers of large binary files, in addition to source code. The large binary files and continuous movement of them to support Continuous Integration and deployment processes make a hybrid infrastructure worthy of consideration; however, that must be considered against simplicity of an exclusive Azure topology.

In both scenarios, having the master server in Azure is the best practice. This allows for the best possible data durability and availability.

Another key goal in all topologies is to leverage the Azure world-class WAN infrastructure for movement of data around the globe, including Azure Front Door, and use the public internet only when it is necessary to connect from Azure data centers to your own offices or to end users.

SUMMARY OF A HYBRID TOPOLOGY

A hybrid deployment topology takes advantage of both Azure and on-premises infrastructure to deliver a solution meeting availability, performance, and systems management needs.

The hybrid topology illustrated in this document is most appropriate for organizations that have significant investment in existing on-premises infrastructure to leverage. It helps optimize for minimization of Azure data egress charges by keeping the vast majority of routine read-only traffic, more than 98% of total traffic, within the corporate LAN environments.

The key concept of a hybrid topology is that **syncs are local**, and **submits go to Azure**. This is achieved by deploying Helix Core edge servers in on-premises data centers around the globe, all connected to a master server in Azure. In reality, this is a "cloud native" master Helix Core server.

Edge servers are best suited for the job, as they provide the best performance. However, edge servers require site-local backup and Helix Core checkpoint operations, in addition to checkpoints that will occur in Azure for the master. It is also recommended that, like the master, each edge server be deployed with a site-local HA replica server machine to enable fast recovery in event of loss the edge server.

As an alternative to edge servers, using basic forwarding replicas or even more basic Helix Core proxies can simplify the topology. Replicas and proxies can reduce or eliminate the need for site-local backups that are required for edge servers. However, edge servers provide better performance for a wider range of user operations. Therefore, they are typically the preferred approach.

There are no Perforce software license cost implications for using edge servers, forwarding replicas, or proxies as the on-premises element of a hybrid topology.

Another way to deploy hybrid topology, especially for organizations that want to implement work from home (WFH) scenario or reduce investment in on-premises infrastructure is to keep only the master server on-premises and deploy Helix Core edge servers in Azure data centers as illustrated in the diagram below:

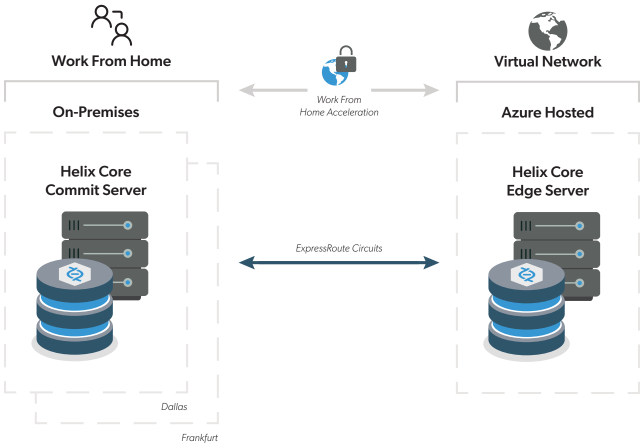

Figure 1: Hybrid topology with master on-premises & edge server in Azure

CONNECTING AZURE TO ON-PREMISES

The easiest way to connect your infrastructure on Azure to your on-premises topology is by using a VPN. This connects your Azure VNet (Virtual Network) to your on-premises network or data center. It is connected by a Virtual Private Network (VPN) Gateway on the Azure side, and by a Customer Gateway on your side. The Customer Gateway is either a physical device or software appliance. Azure has tested devices from vendors like Checkpoint, Cisco, Dell, Fortinet, Juniper, and Palo Alto. The Customer Gateway can also run on some Windows Server machines.

If the bandwidth and potential latency of an internet connection doesn't meet your throughput needs, you can use another Azure service called ExpressRoute, which uses dedicated network connections from your premises to Azure.

THE AZURE EXCLUSIVE TOPOLOGY

Environments that are considering upgrading infrastructure to meet growing software development needs, or do not want to maintain on-premises servers, should consider an Azure exclusive "all-in" topology.

Essentially, this has users connect to the Azure master server directly, with no on-premises infrastructure other than user equipment (desktops, laptops, and workstations). It is also possible to run Azure workstations, with GPU support for advanced graphics use cases.

An Azure exclusive infrastructure optimizes for ease of administration. It is important to note that even in cases where on-premises infrastructure is an option, it is worthwhile to do benchmarking of the cloud-only configuration. When you compare the performance of on-premises infrastructure with Azure exclusive, the results may surprise you (e.g., Azure may perform better than on-premises).

An Azure exclusive topology may also use edge servers, standby servers, forwarding replicas, and proxies, just as a global on-premises topology would.

In addition to administration and performance benefits, an Azure exclusive topology brings additional benefits in areas such as security, disaster recovery, integration with Azure platform services (such as DNS and directory services), and in supporting other workloads such as CI/CD. A simpler caching strategy can yield savings on data egress charges, while having the simplicity of an Azure exclusive operation. These tools are easier to deploy with Azure/Perforce and you're paying only for what you use.

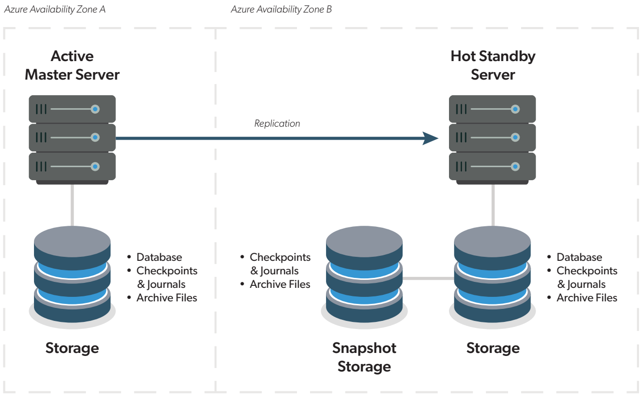

Figure 2: The Azure exclusive topology

BUILD SERVERS

Build servers are a great example of additional workloads that are ideal for Azure configurations. You can configure a build server next to the master in an Azure resource group using proximity placement group (PPG) if needed as explained here, which provides low network latency, and/or high network throughput between them. This facilitates high velocity CI/CD applications.

If you have cyclical or unpredictable demand for CI/CD resources, you can automatically scale either using Azure VMs or containers to bring up additional build runner machines. Jenkins, for example, has a plugin that lets you automate the process of launching instances and sending traffic to them, based on build server load. You can automatically terminate the instances as the traffic goes back down.

SECURITY BENEFITS OF AZURE EXCLUSIVE

Beyond performance and ease of administration, an Azure exclusive infrastructure offers security benefits, through the use of Azure Active Directory (AAD) and Azure Role Based Access Control (RBAC). Combined with Helix Core's built-in protection architecture, you can build out a strong defense in depth strategy, that covers physical, network, system, application, and data layers. This includes:

- Fine-grained access controls.
- Multi-factor authentication.
- HSM-based key storage.
- Server-side encryption.

As with many Azure services, implementing these capabilities is generally less expensive, and requires less effort on the part of systems administrators, than similar on-premises systems delivering such functionality.

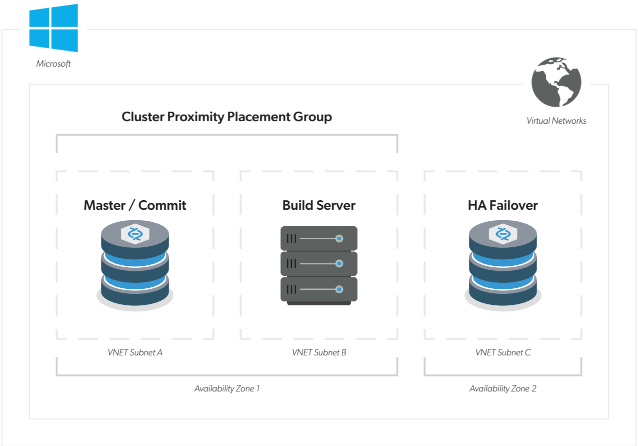

Figure 3: Build server in Proximity Placement Group

MONITORING THE TOPOLOGY

Azure exclusive topology brings you built-in services that can be leveraged for monitoring. Azure Monitor gives you directly actionable insights into the global health of your topology. Logs, metrics, and alerts give you a view of resources, applications, and services that you are using on Azure. It can also be used to monitor on-premises servers. Monitor also integrates with another Azure service called Azure Alerts, which can be used to send alerts (text, email) to your team based on metrics out of defined range. Azure Monitor can even be used to fire off remediation actions using alerts, such as rebooting, or even initiating server failover processes in the event it is discovered that a particular instance is unresponsive. For more details on Azure Monitors, refer to the documentation here.

In addition, there are detailed Helix Core monitoring options based on Prometheus and Grafana, which can be integrated into the overall monitoring solution together with Azure Monitor.

HYBRID VS. AZURE EXCLUSIVE

In many ways, hybrid should be looked at as Azure exclusive with the addition of intelligent local caching on-premises delivered by Perforce Federated Architecture. As a general statement, it is certain that the hybrid topology will greatly reduce data egress from Azure as compared to an exclusive topology. However, benchmarks testing performance benefits for end users might not show significant difference in performance for end users when leveraging on-premises hardware, perhaps counter-intuitively.

Results will vary based on many factors, including greatly varying quality of on-premises hardware as compared with consistent and continuously evolving Azure infrastructure.

AZURE FOR BACKUP ONLY

Some customers with existing on-premises infrastructure and limited Azure experience can start with Azure in a very non-intrusive manner. They first create a disaster recovery (DR) Helix Core replica in Azure without changing any on-premises infrastructure. This can be useful in gaining valuable experience for your IT staff in Azure before making a larger move. It is completely transparent to users and has no impact on them.

The effectiveness of a DR replica is enhanced by using Azure Disk Backup with the built-in multi-region durability of storage and backup options (e.g. locally redundant storage). You can extend this with cross-region replication (geo-redundant storage). Both of these strategies help to further ensure your assets are protected.

With Azure Disk Backup lifecycle policies, you can affordably keep your backups and log files "forever", at a low cost with tiered storage options driven by rules with specific SLAs and costs based on the frequency and speed of retrieval you need. If you don't want to keep them forever, you can control when they will be deleted through a policy, too.

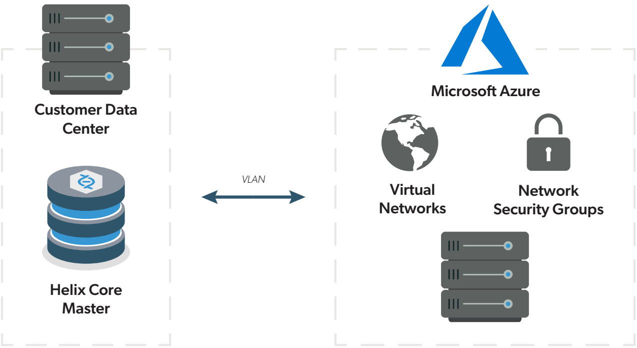

Figure 4: Azure for Backup Only

TESTING AND EXPERIMENTING

Another great way to get started with Azure is the deployment of additional replicas as test environments for new infrastructure tools and workflows, external code testing – or in game development for new releases of the game engine. While this approach has benefit, it also ultimately defers some of the biggest benefits that are only fully realized further into the cloud. Helix Core cloud deployments have been successful to the extent the approach allows. With the help of Perforce Consulting, you can go more fully and boldly into Azure.

INTRODUCTION OF EDGE SERVERS

For customers already familiar with Helix Core edge servers, the following is nothing new. Customers considering a cloud migration involving a hybrid topology might find their first exposure to Helix Core edge servers.

One consideration for deploying new edge servers in general is the initial introduction of an edge server into the topology. It is not as transparent to users as a forwarding replica or proxy can be. This is because users must do a one-time transition from their existing workspaces to either transfer them to the edge server, or to create new workspaces that are bound to the edge server. This may involve dealing with opened files in the to-be-abandoned workspaces, and shelving, submitting, or reverting opened files. Shelved files can be reopened in the new workspaces on the edge server. As this transition involves coordination with end users, communication of user responsibilities is key to project success.

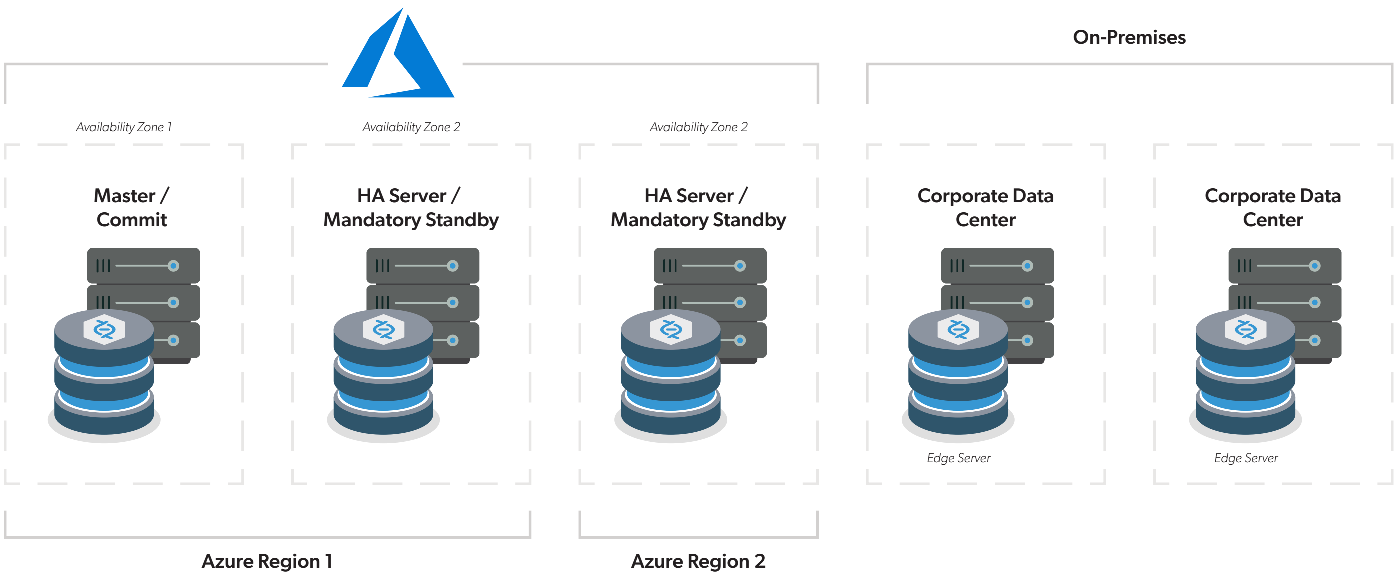

Figure 5: Example hybrid topology

THE REST OF THE ECOSYSTEM

This guide does not discuss the rest of the Helix Core ecosystem, which can include integrations with identity management systems (such as Active Directory or LDAP), workflow management, defect tracking systems, Continuous Integration and deployment solutions, and others.

The impact on each such system should be accounted for when doing a cloud migration. Having done many cloud migrations, the impact on such integrations is usually minimal. It typically involves opening narrow network firewall rules to, for example, allow the Azure master server to authenticate from an on-premises AD server. Azure Active Directory Domain Services (Azure AD DS) provides a managed domain service with a subset of fully compatible traditional AD DS features such as domain join, group policy, LDAP, and Kerberos/NTLM authentication.

It integrates with Azure AD and, when synchronized with an on-premises AD DS environment, allows you to extend your on-prem identities to run in Azure as part of a lift-and-shift strategy.

For more details on comparing on-premises AD DS, Azure AD, and Azure Active Directory Domain Services, refer to the documentation here.

Sample Hybrid Cloud Topology

A sample hybrid cloud topology will include a master server residing in Azure in a primary region, typically nearest the largest concentration of users. A Helix Core standby replica will be configured in a separate availability zone in the same Azure region, enabling high availability capability and real-time data protection.

Users will access Helix Core via edge servers deployed on-premises in a corporate data center, in cases where a local data center is available. Each edge server will have a site-local standby replica server to support potential failover of that edge server, enabling high availability capability at that site.

When development teams are globally distributed, additional forwarding replicas should be deployed within Azure in the region nearest major concentrations of users. The aforementioned Azure Front Door can make this simpler to configure from a networking standpoint. The on-premises edge servers will then connect to the closest regional forwarding replica, or the master if it is closer. This approach is intended to manage Azure infrastructure when moving data across transcontinental and transoceanic distances, improving performance and reducing costs as compared to using public internet pipes.

See Figure 5

For purposes of illustration, a sample topology will be described. This sample topology uses a hybrid cloud approach, achieving the highest possible data safety and availability for the master server, while leveraging on-premises infrastructure and minimizing Azure data egress charges.

The following are host names and Perforce ServerID values:

Virtual Machine Hostname	ServerID	Description
p4-azure-usea-01	master.1	Master/commit server. The .1 is the SDP instance name, a data set identifier.
p4-azure-usea-02	p4d-ha-azure-usea	High availability (HA) server, configured as a mandatory standby replica targeting the master.
p4-azure-auea-01	p4d-fr-azure-auea	Forwarding replica in Australia East, targeting the master.
p4-syd-01	p4d-edge-syd	Edge server on-premises in Sydney, Australia, targeting the forwarding replica in Azure.
p4-syd-02	p4d-fs-edge-syd	Site-local HA server, configured as a standby server targeting the Sydney edge server.

Notes:

- The host naming convention indicates that the host is dedicated to the Helix Core infrastructure (hence the p4- prefix), but does not indicate the specific role. This can change, and possibly even be reversed, in a failover situation. Host names have numeric suffixes.
- The Helix Core ServerID by defines the current role of the Helix Core service on the given machine (for the given data set; this sample illustrates only a single data set).
- The ServerID values are a standard, codified in the SDP `mkrep.sh` script. This standard should not be altered.
- The host naming convention is illustrative of a sample best practice, but can be adapted to use an organization's host naming convention. However, it is generally deemed best that the host name not be tied to a specific role, to avoid confusion in failover scenarios.
- End users will not need to know the server host names, but will instead use a host alias that is the same as the server hostname, but with the numeric suffix and 'dash' characters removed, e.g. `p4azure-usea` or `p4syd`.

HELIX CORE HA REPLICATION DETAILS

The Helix Core master/commit servers will reside on a host named `p4-azure-usea-01`, with the `-01` suffix being an arbitrary integer. This may change from time-to-time as the hardware is updated to a new operating system, a new instance type, or otherwise replaced.

The master/commit server will have a designated high availability (HA) server. For purposes of this guide, an HA replica is one that:

- Is located in the same Azure region as its master server.
- Is in a different availability zone within the same Azure region as the master server it targets.
- Is configured with a **server spec** named according to the server spec naming convention. For example, `p4d_ha_azure-usea` for an HA server in the eastus region, USA.
- Has a `Services:` field of the server spec set to a value of `standby`.
- Has an `Options:` field of the server spec set to a value of `mandatory`.

- Is running Helix Core (P4D) 2018.2 or later, as this version includes key features that our replication strategy relies on for using `standby` and `forwarding-standby` types of replica, and supports the `p4 failover` command.
- Has a `db.replication` setting of `readonly`.
- Has an `lbr.replication` setting of `readonly`, indicating a full replica of metadata and archive files.
- Has an `rpl.journalcopy.location` setting of `1`, optimizing journal storage.
- Is not filtered in any way, with no use of '`-T`' flag in the replica's configured "pull" startup commands, and no use of various `*DataFilter` values in the server spec.

HELIX CORE DR REPLICATION DETAILS

In addition to an HA solution, a disaster recovery solution should be provided. In this sample topology, the forwarding replica in the Azure Australia East data center provides DR capability and routes long-distance WAN traffic on the optimal route using Azure Global network.

Server Deployment Package

The Perforce Server Deployment Package (SDP) is open-source software available from Perforce. It is used to manage a variety of Helix Core topology components, including those deployed on VMs in Azure as well as on-premises.

The SDP evolves with Helix Core, providing an ongoing reference implementation of many best practices of various kinds, from routine maintenance (zero downtime checkpoints) to performance optimization and much more.

For purposes of this document, the key things to understand about the SDP are its optimal storage layout. The SDP maintains three storage volumes for Helix Core (in addition to the OS root volume):

- `/hxdepots` — Contains contents of archive files as well as rotated journals and checkpoints, with potentially massive amounts of data. This is typically accessed with long-stream reads and writes. It must be backed up and should also be encrypted.
- `/hxmetadata` — Contains metadata databases only, frequently accessed with random I/O patterns. It must not be backed up directly, and it is constantly being written.
- `/hxlogs` — Contains active journal, active server log, and various other logs. This is normally backed up (although it is not a requirement if you have HA/DR replicas).

Azure Storage Configuration

Before configuring the VM instance, create and configure Azure managed disk storage volumes. This is the set of storage volumes needed for both the master and HA replica VM instances.

Here Standard SSD (SSSD) indicates Azure General Purpose SSD storage, and Standard HDD (SHDD) indicates Azure Throughput Optimized HDD storage. The objectives:

- Spend for the performance of SSSD where it will deliver the most value, e.g., meeting the low-latency demands of the metadata volume.
- Use SSSD on the logs volume, the files in which are rotated frequently and thus should not grow extremely large.
- Utilize cheaper SHDD volumes where they will perform well, i.e. for long-stream read and write operations that commonly occur on the depots volume.

The base sizes are for a production development server. Actual sizes will vary and may be larger, especially for the `/hxdepots` volume which can easily be multiple terabytes (or more!) based on the scale of software to be developed with this server.

It's important to consider using stripped disks in Azure in order to get the most IOPS. Please refer to Azure documentation on options for this here.

Storage Configuration for On-Premises Edge Servers

For configuring storage for the on-premises edge server machines (an edge server and its site-local HA server), use the best equivalents of the Azure managed disk settings listed above.

Azure VM Instance Configuration

The following information is useful for creating VM instances for Helix Core servers. The headings roughly match those seen when you select "Create a virtual machine" from the Azure Portal.

After the first instance is created, subsequent instances can be configured by selecting an existing instance from the Azure Portal to use Export Template with the required parameters for the new instance.azure-usea-01 host). Note that the VNet and security groups should be created before launching the instance.

Managed Disks Name Tag	Mount	Base Size	Managed Disks Storage Notes
p4-azure-usea-01-hx-depots	/hxdepots	1TB	SHDD, SSE with CMK
p4-azure-usea-01-hx-logs	/hxlogs	128G	SSSD, SSE with PMK
p4-azure-usea-01- hx-metadata	/hxmeta-data	64G	SSSD, SSE with PMK
p4-azure-usea-02-hx-depots	/hxdepots	1TB	SHDD, SSE with PMK
p4-azure-usea-02-hx-logs	/hxlogs	128G	SSSD, SSE with PMK
p4-azure-usea-02dhx-metadata	/hxmeta-data	64G	SSSD, SSE with PMK

SSE = Storage Service Encryption CMK = Customer Managed Keys
PMK = Platform Managed Keys

While it is simple to use the default Storage Service Encryption (SSE) for encrypting disks at rest, Azure also supports Encryption-at-host feature for the end-to-end encryption of disks, which, if required, can be configured using the details here.

CHOOSE AZURE IMAGE

Various Linux distributions have been tuned for optimum performance in Azure: see this list. For example, the OpenLogic supported CentOS images (OpenLogic is now part of Perforce) are easy to maintain with standard RHEL/CentOS administration practices.

CHOOSE VM SIZE: GO WITH FSV2 FAMILY

Consider the following when selecting the Azure VM size:

- The compute optimized instances are generally expected to deliver the best overall performance for Helix Core development. Faster processors help avoid bottlenecks with the large number of compression/decompression operations typical when handling large digital assets.
- Select only VM Sizes with **Premium Storage** available.
- We recommend "Accelerated Networking" capable VM Sizes.
- Select an instance with sufficient RAM. For example, for a customer developing software with total assets on the order of 2TB, we went with **Standard_F32s_v**2 instance type, which has 32G of RAM available.

CONFIGURE INSTANCE OPTIONS

When configuring instance options, associate it with the Helix Core VNet. Select the appropriate availability zone (subnet), being sure the master and HA servers are in the same region but different subnets. Ignore the following settings:

- Proximity Placement Groups (PPG).

Check the following options:

- Enable Azure Monitor's detailed monitoring.
- The tenancy should be shared. We do not recommend using Dedicated Hosts as this is intended to apply to data workloads involving strict regulatory data isolation compliance requirements. The term dedicated in the context of Azure tenancy does not imply superior performance.

CONFIGURE STORAGE OPTIONS

Only the instance root volume needed to hold the OS is created at instance creation time. The Azure managed disk volumes for the different Helix Core-related /hx* volumes are created separately, as noted above.

Reserved Instances

When you launch Azure instances, the standard option is Pay-as-you-go, which means you pay by the minute each VM is running. Once you have solidified your configuration, and you know you have specific servers that will be long-lived, you should replace those PAYG Instances with Azure Reserved VM Instances. With Reserved Instances, you pay a certain amount upfront and commit to a term contract. This can result in a steep discount compared to the pay-as-you-go price of the same configuration.

Availability Zones (AZ)

An Availability Zone is a unique physical location within an Azure Region. Each Availability Zone consists of one or more datacenters with independent power, cooling and networking. AZs are isolated from one another, but are connected through low-latency links. Ensure that the master and standby server are in different availability zones for optimal redundancy.

Virtual Network (Vnet) and Perforce License Files

Define a Virtual Network dedicated for usage by Perforce components. The private IP addresses of Azure instances should be the ones given to Perforce sales (sales@perforce.com) when requesting Perforce license files.

Server Deployment Package (SDP)

SERVER STORAGE AND DEPOT SPEC STANDARDS

Perforce customers already using the SDP will not need to make adjustments to follow storage standards, as those are baked into the SDP. When moving to Azure, adopting the following standards is necessary to ensure all critical data is on managed disk volumes, backed up, encrypted correctly, and on the cost-optimized storage solution.

The standards are:

- The `server.depot.root` configurable should be set per the SDP standard, e.g. with a value like: `/p4/1/`depots, which is on the `/hxdepots` volume.
- Depot specs should have only the default (relative) depot `Map:` field value of: `DepotName/...`
- Both within the Azure environment and for any on-premise edge servers, server machines are normally configured to have exactly one logical storage volume for depots, referenced by `server.depot.root`. This volume must be sufficiently large to contain all archive files, optimally with the capability to grow beyond the starting capacity.

Azure managed disks can go to 32TB, and it is possible to have multiple disks if required. The goal here is to avoid (where possible, e.g. 32TB or less) an idiosyncratic installation for the edge servers on-premises with symlinks on a per-depot basis, as some customers have done with on-premises solutions due to limited physical hardware. Such idiosyncrasies may seem harmless at first, but introduce complexity and tend to cause problems in failure/recovery situations. In some cases, depots can be stored on the wrong volume where they are not backed up, and deprive P4ROOT of high-value disk space.

It must always be true that the Helix Core `server.depot.root` configurable be set to `/p4/N/depots` (per the SDP standard) and honored such every depot can be always accessed via the path `/p4/N/depots/DepotName`, where *N* is the SDP instance name.

SDP REPLICATION STANDARDS

The edge server and replicas are set up using the SDP `mkrep.sh` script. This codifies creation of replicas with appropriate best-practice configurables based on the replica type.

Prior to using `mkrep.sh`, the geographic site tags must be configured in the file `/p4/common/config/SiteTags.cfg`.

Ensure that the P4TARGET value of each replica is set to a symbolic alias that will survive a failover. Just as end users should reference their local primary edge server or the master server using a host alias that will survive a failover, so should a replica. So, for example, the Sydney edge server would have a P4TARGET value something like `p4azure-usea:1666`, but not `p4azure-usea-01:1666` or `p4azure-usea-02:1666`. The failover procedure will redirect the `p4azure-usea` alias from the `-01` to the `-02` box.

SDP SAMPLE EDGE SETUP

The following sample commands are illustrative. Actual commands and site-specific operational procedures would be used for an actual deployment.

The edge server in the sample topology with the SDP installed, which has an SDP instance name of 1 (the default first instance), would be configured with this command, run as **perforce@p4-azure-usea-01**:

```
cd /p4/common/bin
./mkrep.sh -i 1 -t edge -s syd -r p4-azuresyd-01
```

And its site-local HA replica command would be:

```
./mkrep.sh -i 1 -t fa -s syd -r p4-azuresyd-02
```

Both of these commands would be run from the master server before doing further configuration.

Then, the edge server seed checkpoint is created using a command like this sample:

```
./edge_dump.sh 1 p4d_edge_syd
```

That creates an edge seed checkpoint in /p4/1/ checkpoints, which must be transferred to the edge server. The SDP must also be configured for the instance. Then, recover that edge seed by running the following as **perforce@p4-syd-01**:

```
./recover_edge.sh 1 /p4/1/checkpoints/p4_1.edge_
syd.seed.ckp.1234.gz
```

Replace `1234` with the checkpoint number of the edge seed checkpoint generated above.

Next, transfer the generated edge seed checkpoint to both the edge server and its site-local replica, and replay the checkpoint on each edge server.

SDP TWEAK FOR AZURE SNAPSHOTS

When deployed in Azure, the SDP `backupfunctions.sh` should be tweaked slightly. The goal of the adjustment is to make it so that the snapshot for the `/hxdepots` volume occurs at the optimal point in time, immediately after the Helix Core checkpoint operation completes. This achieves the minimum lag between the time the digital asset for recovery is created and the time it is whisked away to greater safety. This is accomplished using Azure command line tools, doing an `az snapshot create` call. The call might look like:

```
perforce@p4-azure-usea-01:/home/perforce az
snapshot create -g myResourceGroup --source
“$diskID” --name hxdepots-backup
```

In this example, the `$diskID` specified can be obtained using **az vm show** (see snapshot a disk). In similar fashion, the root volume should also be snapshotted. The `/hxmetadata` volume should not be snapshotted. Snapshotting the `/hxlogs` volume is not usually done, as the most important data is in the active P4JOURNAL file is replicated. Other data, such as server logs, is more transitory.

TUNING FOR PERFORMANCE, SECURITY, SAFETY, ETC.

The SDP contains a script that promotes various best practices for tuning performance, security, data safety, etc. It is intended to be run on brand new SDP instances, but it also provides guidance on settings that can be applied to any existing data set.

The following settings should be checked using the `p4 configure show SettingName` command. Adjust if necessary to these values using `p4 configure set SettingName SettingValue` based on the following table.

For the most current information, see Best Practices Settings Guidance in the SDP.

Setting Name	Recommended Value	Comments
run.users.authorize	1	Security
filesys.P4ROOT.min	5G	Safety
filesys.depot.min	5G	Safety
filesys.P4JOURNAL.min	5G	Safety
server	4	Logging
monitor	1 (or 2)	Monitoring
db.reorg.disable	1	Performance
net.tcpsize	0	Performance
net.autotune	1	Performance
db.monitor.shared	4096	Performance
net.backlog	2048	Performance
lbr.autocompress	1	Safety, Performance
lbr.buffsize	1M	Performance
filesys.bufsize	1M	Performance
server.start.unlicensed	1	Licensing

Setting Name	Recommended Value	Comments
rejectList	P4EXP,version=2014.2, Operating System	Security/Cyber Defense, Safety
server.global.client.views	1	Edge Functionality
server.locks.global	1	Edge Functionality
auth.id	p4_SDPInstanceName	Functionality
rpl.forward.login	1	Functionality
dm.shelve.promote	1	Swarm
dm.keys.hide	2	Swarm
filetype.bypasslock	1	Swarm

Helix Management System (HMS)

The Helix Management System is included with the latest version of the SDP. Among other things, it provides key features useful for managing a sophisticated global hybrid topology.

TIGHT SHIP MANAGEMENT OF SDP AND EXTENSIONS

A tiny dedicated Helix Core instance runs on a bastion host, `p4hms.p4demo.com`. Its SDP instance name is `hms`. It manages the SDP on all hosts where any Helix Core topology components exist. This includes Helix Core (P4D) master servers, replicas, edge servers, proxies, and brokers, and Helix Core Plugin for Graphical Tools (P4GT). Tight-ship management keeps the SDP scripts and configuration files current and in sync on all hosts, using Helix Core to deploy and verify files.

HELIX TOPOLOGY DEFINITION

A single Helix Topology configuration file defines all SDP instances and all hosts, and is aware of which topology components operate on which hosts in "normal" as "post-failover" modes. See this Sample Helix Topology file.

CENTRALIZED MANAGEMENT

The Helix Topology file provides a syntax for naming any component related to any SDP instance, by combining the SDP instance name with the component name (defined in the Helix Topology configuration file). This allows any instance's P4D (or any other component) to be stopped, started, or have its status checked from the HMS server, with a command like these examples:

```
hms status 1:master
hms stop 1:p4d_edge_syd
```

FAILOVER PLAN DEFINITION

HMS emphasizes and requires that preparation and planning for failover be defined in advance. Should failover ever be required, it can be executed swiftly according to a defined plan. The plan should account for recovery from various of failure scenarios that might occur, starting with the most obvious ones like a failure of the master or edge server machines.

Based on the situation that leads one to contemplate that a failover is or might be necessary, the following plans are available as options:

SIMPLIFIED FAILOVER EXECUTION

Failover To Local Offline DBS

Failover plans with "Local" in the name involve replacing the live (and presumable corrupt) databases on a given server machine with the spare set of databases maintained by the SDP on the same host in the `offline_ db` folder. No change is needed to redirect users or network traffic for Local failover plans.

Local failover plans can be executed with a command similar to this:

```
hms failover local i:1 u
```

HA Failover of Azure Master

Failover plans with HA in the name involve promotion of a standby, with live and real-time replicated databases and archive files, to become the new master. User and traffic must be redirected for HA failover plans.

HA failover of the Azure master can be executed with a command similar to:

```
hms failover ha i:1 u
```

Under the covers, the `hms` script executes the Perforce commands necessary to achieve promotion of the failover machine to the master machine, e.g. using the `p4 failover` command.

Following execution of the `hms` script, a corporate network DNS change must be made to change the `p4azure-usea` host alias, such that traffic routes to `p4azure-usea-02` instead of `p4azure-usea-01`, to complete failover. Because the HA server in Azure is configured as a mandatory standby type of replica, global downstream edge servers and replicas will pick up where they left off with replication post-failover.

HA Failover of Sydney Edge

HA failover of the Sydney edge can be executed with a command similar to:

```
hms failover syd-edge-ha i:1 u
```

Following this command, a corporate network DNS change must be made. This changes the `p4syd` DNS alias used to refer to the Sydney edge server from `p4- syd01` to `p4d-syd-02`.

Failover Plan	From/To	Sample Usage Scenario
Master Local	p4azure-usea-01 (same host)	Use when host p4azure-usea-01 is OK, but databases are corrupted, e.g., due to sudden power loss or human admin error. Failover recovers databases on the same host.
Master HA	p4azure-usea-01 to p4azure-usea-02	Use when host p4azure-usea-01 is unusable, or needs to be taken offline (e.g., to upgrade RAM). Failover recovers to a standby replica.
Sydney Edge Local	p4-syd-01 (same host)	Use when host p4-syd-01 is OK, but databases are corrupted, e.g., due to sudden power loss or human admin error. Failover recovers databases on the same host.
Sydney Edge HA	p4-syd-01 to p4-syd-02	Use when host p4-syd-01 is unusable, or needs to be taken offline (e.g., to upgrade RAM). Failover to a site-local standby replica.

Azure Network Security

Network Security Groups (NSGs) should be considered – these are basic, stateful packet filtering firewalls. It should be enabled with ports opened as identified below.

If Helix Swarm is used for code review and repository browsing, then an HTTPS rule opening port 443 should be included.

In the examples that follow, p4d processes run on port 1999, and p4broker processes run on port 1666. Using p4broker processes is optional, and if they are not used, then p4d processes run on 1666 instead of brokers. Brokers provide a variety of functionality to Helix Core administrators, but also introduce an extra topology component to maintain and upgrade.

HMS HUB AND SPOKES

The HMS server is the one Helix Core server that rules all others, in that it manages things such as backup scripts, custom trigger scripts, and even crontabs for all instances on all Helix Core-related machines, in and out of Azure. As such, this server should be configured, from an access control perspective, as a Bastion host or Azure Bastion service, with the highest available security. It should ideally be deployed in Azure as a separate VM instance from other server instances containing development work, though this host can also be deployed on-premises.

The HMS server (`p4hms.p4demo.com`) must allow traffic as follows:

- inbound on port 7467 (always SSL encrypted) from all p4d servers managed by HMS.
- inbound on port 7468 (always SSL encrypted) from all p4d servers managed by HMS.

INBOUND RULES FOR ON-PREMISES EDGE SERVERS

On-premises edge servers require inbound access as follows:

- Port 1666 from the internal corporate network.
- Port 1999 from the internal corporate network.
- Port 22 (SSH) from the HMS server (a Linux bastion host).

INBOUND RULES FOR AZURE SERVERS

Azure servers require inbound access as follows:

- Into the VNet on port 1999 (optionally SSL encrypted) from VA1 edge servers.

- Into the VNet on port 1666 (optionally SSL encrypted) from the HMS server.
- Into Port 22 (SSH) from the HMS server and Linux bastion host(s).

OUTBOUND RULES FOR ALL SERVERS

Outbound rules can be restricted or unrestricted, per corporate policy. If they are to be restricted, restrictions should follow these guidelines that balance security and ease of administration: Host and Port Access List for Helix Core Servers.

PERFORCE SSL ENCRYPTION

Perforce provides SSL encryption that adds an extra layer of defense and security, protecting data in transit across the network. (Azure storage provides encryption of all data at rest on the Helix Core servers.)

Benchmarks indicate that the performance costs for using SSL are minimal, often negligible.

SSL may not be required if good perimeter security in place, including within Azure, on-premises, and between Azure and the on-premises infrastructure. However, SSL does introduce a few complexities:

- As with any OpenSSL technology, certificates need to be generated and managed. Helix Core server products can generate their own certificates, so this complexity is limited to dealing with occasional certificate expiration.
- Failover solutions are somewhat more complex and less transparent with SSL, as the goal of failover — to transparently change the server that a user is connected to — is diametrically opposed to the goal of SSL, which is to ensure users are aware when a server they think they are communicating with changes. Theoretical options exist, such as getting all users to trust all possible failover target servers. Customers using SSL thus far have been willing to accept that failover may require users to redo the trust of the new server after failover. This approach may involve retooling for robotic users to incorporate trust and login logic into Helix Core interactions in code, as a best practice.
- In some cases, Helix Core servers under heavy load may — due to SSL timeouts — drop connections, resulting in user errors for conditions that may otherwise have successfully completed after a temporary pause. Similarly, potentially useful Helix Core server log messages are sometimes replaced with unhelpful SSL timeout errors. These issues have thus far not been significant enough to deter customers from using SSL.
- When HMS is deployed on a bastion host, the HMS instance is always configured with SSL, as this server can contain sensitive information that could be used to access other "real data" instances.

3.11.6 OTHER RULES

Depending on what other systems are integrated with Perforce, additional network firewall rules may be needed. For example, if Helix Core LDAP integration is used, a network firewall rule may be needed to enable access (e.g., on port 389 or 636) from the Azure master server into an on-premises corporate Active Directory server.

What's Next?

EVOLUTION

The Azure infrastructure and managed services offerings continue to evolve as Azure and its partners continue to innovate. This guide will continue to evolve to provide ongoing guidance for customers looking to deploy Helix Core on Azure for development.

Likewise, Perforce and Microsoft are partnering on ways to make it even easier for customers to deploy Perforce products on Azure. With the release of Perforce Enhanced Studio Pack, any user now has access to a turnkey bundle of Perforce products, infrastructure as code, and configuration management that can be deployed on Azure in just a few clicks. Any team, regardless of size or technical expertise, can quickly get started with the infrastructure and tools needed to support large files, numerous iterations, and remote team members. This click-to-start Azure deployment can be used by small creative studios looking to accelerate and win in an ever-competitive market, or large entertainment companies that need to quickly spin up new projects.

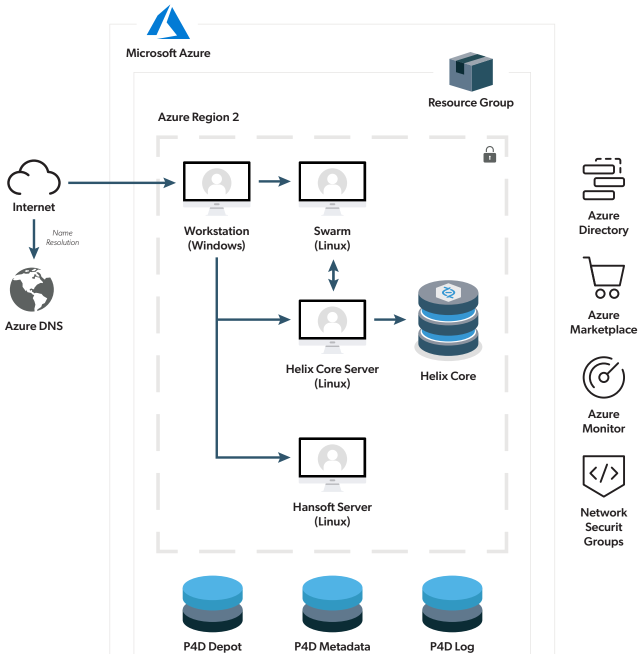

Figure 6: Perforce’s Enhanced Studio Pack on Azure

The following are also being considered:

- Azure Resource Manager (ARM) Templates will be utilized.
- SDP and HMS improvements may simplify operation in Azure, eliminating any tweaks.
- Azure DNS management and Azure Active Directory Domain Services will be explored.
- A reference implementation may be captured with ARM Templates, including illustration of backup of snapshots.
- The Perforce Battle School training class, which currently simulates a sophisticated global topology on-premises using cloud-hosted virtual machines, may itself be migrated to Azure.
- HA and DR solutions may continue to evolve.
- The benefits and costs of container technology such as Kubernetes will be explored with respect to applicability to Perforce server deployment.
- In addition to following general infrastructure as code best practices, using ARM Templates (or alternatives such as Terraform) may be applied in specific development use cases – for example spinning up a new game or virtual production studio. There is a common need to quickly spin up a new studio, in a remote location with respect to master server. Parameterized ARM templates could be employed to spin up edge servers with a subset of depots

to enable limited collaboration for the new studio.

Additional Comments

Review additional details regarding decisions for this document.

1. Ideally, consistent site tags should be used in Helix Core server spec names and host names, e.g., azure-usea as a tag for the Azure eastus region in USA.
2. Note that Azure Files (essentially NFS storage in Azure, but better) was considered for /hxdepots volumes. Azure Files with NFS share is well suited for this purpose, but with some limitations. The chief limitation of NFS in preview is that there are a few Azure storage features that are not supported yet, including Snapshots and Azure Backup support. For the full list of limitations, unsupported features, and prerequisites, refer to the documentation here.

Though there are limitations in preview, NFS has compelling advantages, and it will be revisited for use with the /hxdepots volume in the future.

As of this writing, however, Azure managed disk volumes are more proven, have ideal functionality (e.g., snapshot capability) and offer the best possible performance (with lower latency than NFS).

About Perforce Software

Perforce powers innovation at unrivaled scale. Perforce solutions future-proof competitive advantage by driving quality, security, compliance, collaboration, and speed – across the technology lifecycle. We bring deep domain and vertical expertise to every customer, so nothing stands in the way of success. Privately held and funded by Clearlake Capital and Francisco Partners, our global footprint spans more than 80 countries and includes over 75% of the Fortune 100. Perforce is trusted by the world’s leading brands to deliver solutions to even the toughest challenges. Accelerate technology delivery, with no shortcuts. Get the Power of Perforce.