TECHNICAL GUIDE

Best Practices for Deploying P4 in AWS

Introduction

Organizations across industries can greatly improve their product development lifecycle and accelerate time to market by leveraging the powerful capabilities of Amazon Web Services (AWS). By offering flexible computing power, scalable storage, and secure networking, AWS provides an ideal environment for hosting Perforce P4 deployments. This technical guide breaks down best practices for implementing and optimizing P4 development workflows on AWS, focusing on architectural components.

This document guides a standard deployment of Perforce P4 (formerly Helix Core) in the AWS cloud environment. We'll cover deployment topologies, storage configurations, and server management, focusing on using Amazon Elastic Block Store (EBS) and Amazon FSx for NetApp ONTAP to enable scalable, resilient workflows. These recommendations are based on the experience of senior consultants from Perforce Software and Amazon Web Services (AWS), as well as insights gathered from Perforce customers who operate on AWS.

Note:

This guide focuses on implementing Perforce P4 in AWS environments. Readers should have basic familiarity with Perforce terminology ("edge servers," "replicas," etc.), and AWS terminology ("EC2 instances," "Virtual Private Clouds," etc.). References are also made to the Perforce Server Deployment Package (SDP) —a collection of scripts and configuration standards used to automate and streamline the deployment and maintenance of P4 servers. We'll cover it in more detail later in this document.

Not covered in this guide: the rest of the tools and services available to you with P4, which includes integration with identity management systems (Microsoft Entra, Okta, Active Directory, LDAP), workflow management, defect tracking systems, or Continuous Integration and Deployment solutions, etc.

Jase Lindgren

Senior P4 User Advocate

Watch the Video Guide

Want to skip reading? Follow Senior P4 User Advocate Jase Lindgren in this step-by-step video tutorial on deploying and configuring your own P4 server on AWS.

Contents

AWS Deployment Topologies

Both hybrid and exclusive AWS topologies offer distinct advantages for versioning at scale with P4. By adopting a 'Version Everything' mindset, you unlock the full capabilities of P4, extending versioning beyond source code to digital assets and build results. When versioning build results and build artifacts, sharing modular code architectures allows for massive scale, which enables better module reuse and more fully automated continuous delivery processes. The movement of large binary files to support Continuous Integration and Deployment (CI/CD) workflows makes a hybrid infrastructure worth considering. However, teams should weigh this against the simplicity of an AWS-only approach.

In both scenarios, having the master server in AWS is the best practice. This delivers the best possible data durability and availability.

Another key goal in all topologies is to leverage the AWS world-class WAN infrastructure for the movement of data around the globe. Only connect through the public internet when necessary to link AWS data centers to your own offices or to end users. We recommend visiting instances.vantage.sh to review AWS instances.

Summary Of A Hybrid Topology

A hybrid deployment topology combines the strengths of both AWS and on-premises infrastructure, delivering a solution tailored for optimal availability, performance, and systems management. The hybrid topology is well-suited for organizations that have significant investment in existing on-premises infrastructure. It helps optimize minimization of AWS data egress charges by keeping most of the 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 AWS**. This works by deploying Perforce P4 Edge Servers in on-premises data centers globally, all connected to a master server in AWS. This creates a cloud-native master P4 server.

Edge servers deliver the best performance for this setup. However, edge servers require site-local backup and Perforce checkpoint operations, in addition to checkpoints occurring in AWS for the master. We also recommend that each edge server, like the master, deploys a site-local HA replica server machine for fast recovery if the edge server fails.

As an alternative to edge servers, basic forwarding replicas and Perforce Proxy servers should be considered. Proxy servers deliver big improvements for sync operations, which are the most common. Proxy servers are very easy to setup and maintain, and do not require site-local backups (as they have no unique local data). Forwarding replicas provide better performance for a wider range of user operations than a Proxy but are significantly more complex than a Proxy to setup and manage, yet still simpler than edge servers. Edge servers offer the best performance for an even wider range of user operations than a forwarding replica.

While edge servers offer the best performance for the widest range of user operations, we recommend a graduated approach to deployment based on your specific needs:

1. Start with Perforce proxies as your first option - they provide performance benefits with minimal management overhead and simpler failure modes.
2. Consider forwarding replicas if proxies don't meet your performance requirements.
3. Deploy edge servers only when necessary for specific high-performance scenarios, as they require more complex management, site-local backups, and have more sophisticated failure modes.

This graduated approach aligns with best practices we've observed with customers like Mi'Pu'Mi, who've successfully implemented proxy-first strategies to balance performance needs with operational simplicity.

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

Introduction Of Edge Servers

For customers who already use P4 Edge Servers, integration into a hybrid topology will be familiar. However, for those exploring cloud migration with a hybrid topology, P4 Edge Servers might be new.

One consideration when deploying new edge servers is their introduction into your topology. Unlike forwarding replicas or proxies, edge servers aren't transparent to users. Everyone must perform a one-time transition from their existing workspaces, to create new workspaces bound to the edge server, or migrate existing workspaces to the edge server. Since this requires coordination with end users, clear communication about team responsibilities is critical to project success.

Connecting Aws To On-Premises

There are two ways to connect your AWS infrastructure to your on-premises setup:

1. **VPN for Simplicity**

   The easiest way to connect your infrastructure on AWS to your on-premises topology is through a VPN that connects your Amazon VPC (Virtual Private Cloud) to your on-premises network or data center. It's connected by a Virtual Private Gateway on the Amazon side, and by a Customer Gateway on your side. The Customer Gateway is either a physical device or a software appliance. AWS 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.

2. **Direct Connect for High-Performance**

   If your operations demand higher bandwidth and lower latency than what a standard internet connection or VPN can offer, consider AWS Direct Connect, which uses dedicated network connections from your premises to AWS.

Storage Options

AWS provides a range of robust storage solutions tailored to meet the demands of P4 deployments. When deploying in the cloud, proper storage layout design is critical for performance and cost efficiency. For general use, Amazon EBS is a good choice, offering scalable, high-performance block storage for the various volume types critical to functionality.

For deployments exceeding 16TB depots, Amazon FSx for NetApp ONTAP is a recommended solution. It offers exceptional value by providing highly performant, highly available, cost-optimized, and multi-protocol shared storage specifically designed for P4. For more information, please reference the Server Deployment Storage Configuration section of this document.

The Aws-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 AWS-exclusive "all-in" topology.

An "all-in" topology has users connect to the AWS master server directly, with no on-premises infrastructure other than user equipment (desktops, laptops, and workstations).

An AWS-exclusive infrastructure simplifies administration. Even if on-premises infrastructure is an option, it's worthwhile to benchmark the cloud-only configuration. When you compare the performance of on-premises infrastructure with AWS-exclusive, the results may surprise you. Many organizations find that an AWS-exclusive setup delivers better performance, reliability, and scalability compared to traditional on-premises environments. An AWS-exclusive topology may also use edge servers, standby servers, forwarding replicas, and proxies, offering the same flexibility as a global on-premises topology.

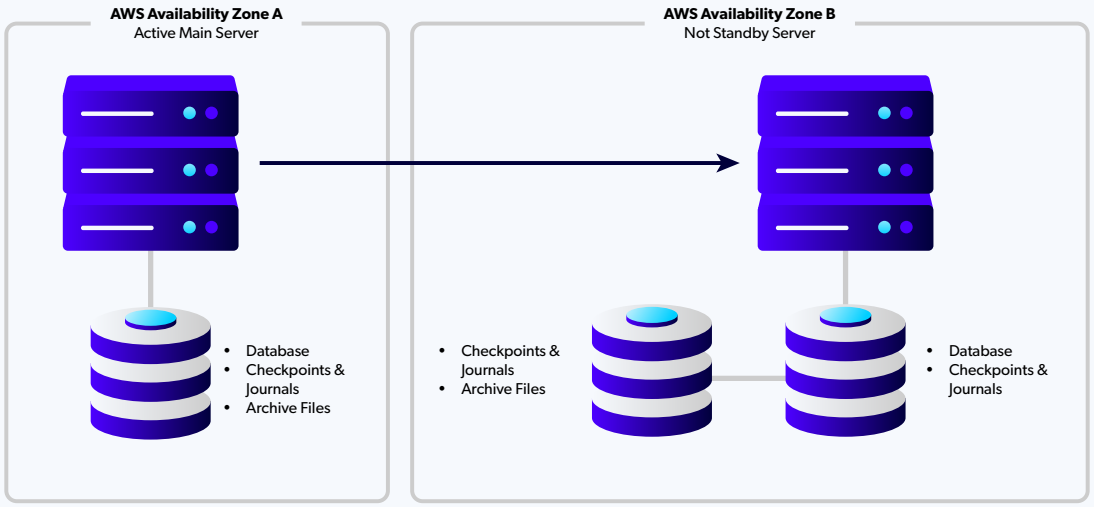

Figure 1: The AWS-exclusive topology

In addition to administration and performance benefits, an AWS-exclusive topology also brings benefits in areas like security, disaster recovery, integration with AWS 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 AWS-exclusive operation. These tools are easier to deploy with Perforce P4 on AWS, and you're paying only for what you use.

Build Edge Servers

Build servers are ideal for AWS configurations. You can configure a build edge server next to the master in an AWS cluster placement group, for 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 using EC2 instances or containers to bring additional build runner machines. Jenkins, for example, has a plugin that automates the process of launching instances and sending traffic to them, based on build server load. You can automatically terminate and remove the instances as the traffic goes back down.

Advanced storage solutions enhance the performance of your build pipelines. For example, FSx for ONTAP offers Snapshots and FlexClones for creating fast, isolated test environments. Snapshots provide point-in-time copies of your data for quick rollbacks and consistent state management. FlexClone creates thin, writeable clones almost instantly and without additional storage overhead, improving efficiency and agility in your development process.

By integrating these technologies into your build pipeline, you accelerate build verification, reduce costs, and streamline continuous deployment. This powers the fast creation of consistent test, staging, and production environments, minimizing downtime, enhancing efficiency, and lowering overall build expenses.

Security Benefits Of Aws-Exclusive

Beyond performance and easy administration, an AWS-exclusive infrastructure adds security through AWS IAM (Identity and Access Management), security groups, and NACLs (network access control lists). Combined with P4's built-in protection architecture, this creates a strong defense at all levels like the physical, network, system, application, and data layers. This includes:

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

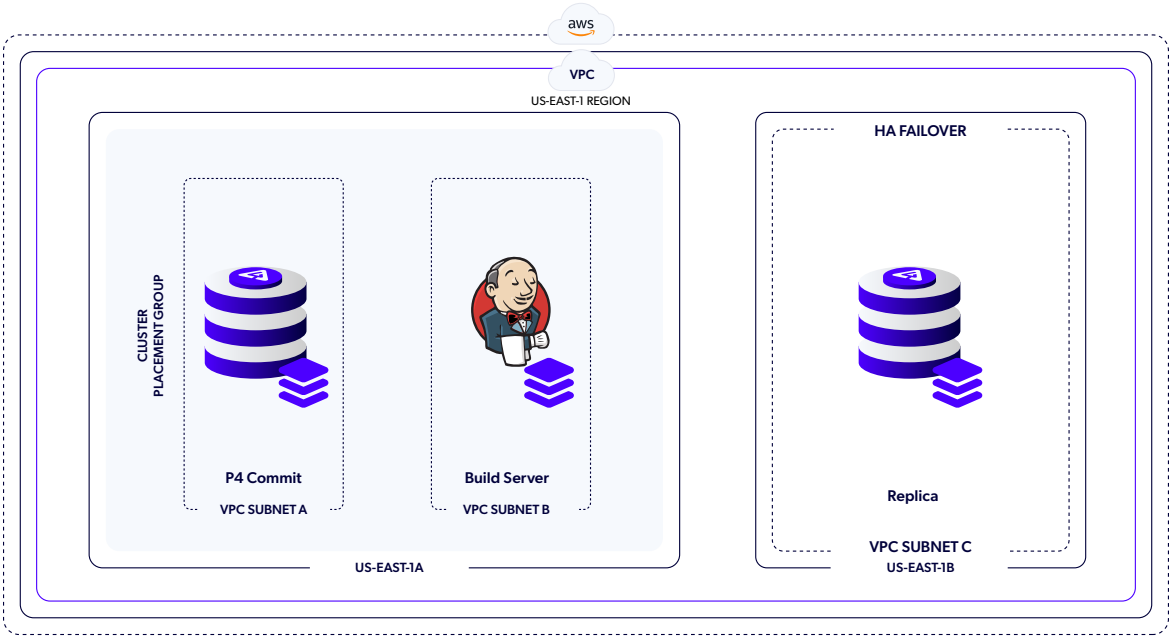

Figure 2 Build server in placement group

AWS services deliver these capabilities at a lower cost and with less administration effort than similar on-premises systems.

Monitoring The Topology

The AWS-exclusive topology includes built-in monitoring services. AWS CloudWatch delivers actionable insights about your global topology health. It provides logs, metrics, and events for all your AWS resources, applications, and services, and can monitor on-premises servers.

CloudWatch integrates with AWS SNS (Simple Notification Service), to send alerts (text, email) to your team when metrics fall outside the defined range. CloudWatch can also trigger remediation cycles, like rebooting, or initiating server failover processes when instances are unresponsive.

Note: The P4 Server software is also referred to as P4D in this document. P4D is short for P4 Daemon—the server process that runs in the background. It's also the name of the executable file (p4d) for the P4 Server.Fine-grained access controls.

For monitoring Perforce P4D Servers, we recommend P4Prometheus, which is designed to monitor P4D servers.

Hybrid Vs. Aws-Exclusive

Hybrid topology is essentially AWS-exclusive with added on-premises P4 deployments using intelligent local caching through our Perforce Federated Architecture. While hybrid significantly reduces data egress from AWS compared to exclusive topology, performance benchmarks for end users may not show major differences.

Results vary based on several factors, including the quality gap between on-premises hardware and AWS's consistent, continuously improving infrastructure.

Aws Backup Only

Some customers with existing on-premises infrastructure and limited AWS experience start with AWS as a simple disaster recovery solution by creating a disaster recovery (DR) P4 standby in AWS without changing any on-premises infrastructure. It's completely transparent to users and has no impact on them.

EBS snapshots that replicate across AWS regions enhance DR replica effectiveness.

P4 depots can use AWS S3 technology in archival strategies. A good strategy is to run active depots on FSx while creating archive depots in S3 buckets for assets not needing immediate access. You can extend the protection of archived assets with Amazon Data Lifecycle Manager (DLM) and AWS Backup cloud storage. Both strategies strengthen asset protection, see the AWS documentation for details on Amazon DLM and AWS Backup.

S3 lifecycle policies keep backups and log files "forever" at a low-cost tiered storage option. Rules define 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 too.

FSx for ONTAP deployment, there are additional protections provided to the assets. FSx for ONTAP delivers point-in-time, space-efficient snapshots that work instantly with file-level granularity, for greater control over backup and recovery processes. **Filesystem Retention** policies manage the lifecycle of FSx ONTAP snapshots, handling automated maintenance and deletion according to specified requirements.

FSx for ONTAP integrates with AWS Backup, simplifying backup management and centralizing data protection across the AWS environment. For long-term archiving, FSx for ONTAP can replicate snapshots to a different region than the production region, then tier these snapshots to a colder, longer-term, cheaper storage layer resulting in low-cost archive storage of assets.

Testing And Experimenting

Another great way to start with AWS is by deploying additional replicas in test environments. These could serve as new infrastructure tools and workflows, for external code testing — or in-game development for new releases of the game engine.

While this approach offers immediate benefits, it only scratches the surface of what is possible. The greatest advantages come from deeper cloud integration. P4 deployments in cloud environments are common and have proven successful, but their impact depends on implementation depth.

Contact Perforce to start your AWS transformation.

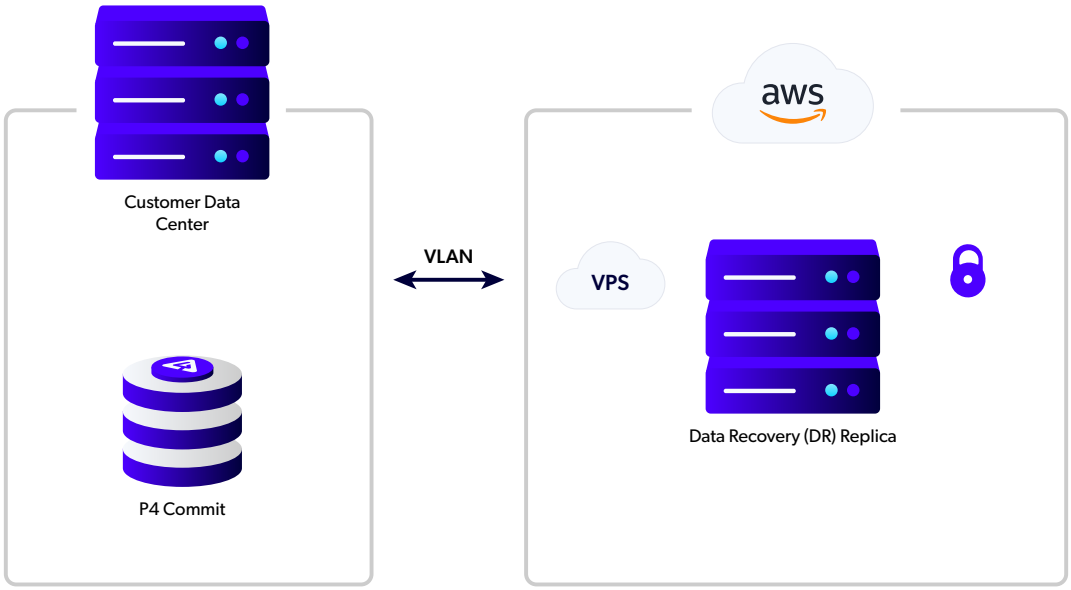

Figure 3 AWS Backup only

Considerations During a Cloud Migration

When planning a cloud migration, it's important to consider the impact of integration with existing systems. Based on our experience with numerous cloud migrations, these integrations typically require only minor adjustments. Typically, it involves adjusting network firewall rules to enable communication between systems, such as allowing the AWS master server to authenticate with an on-premises Active Directory (AD) server. Additionally, AWS offers a service called AWS Directory Service, which is built on Microsoft Active Directory. This service is compatible with the features you've already implemented on-premises and can continue to validate users even if the connection to the on-premises AD server is temporarily unavailable.

Our Professional Services team can help with your migration, contact them here.

Sample Hybrid Cloud Topology

A sample hybrid cloud topology will include a master server residing in AWS in a primary region, usually closest to the largest concentration of users. A P4 standby replica will be set up in a different availability zone in the same AWS region, providing high availability capability and real-time data protection.

Users will access P4 via edge servers deployed on-premises in a corporate data center, in cases where a local data center is available. Each edge server can have a site-local standby replica server to support potential failover of that edge server, enabling high availability at that site.

When development teams are globally distributed, additional forwarding replicas should be deployed within AWS in regions closest to the major concentrations of users. AWS's Global Accelerator can simplify network configuration. The on-premises edge servers will then connect to the closest regional forwarding replica, or the master if it's closer. This approach helps manage AWS infrastructure when moving data across transcontinental and transoceanic distances, improving performance and reducing costs compared to using public internet pipes.

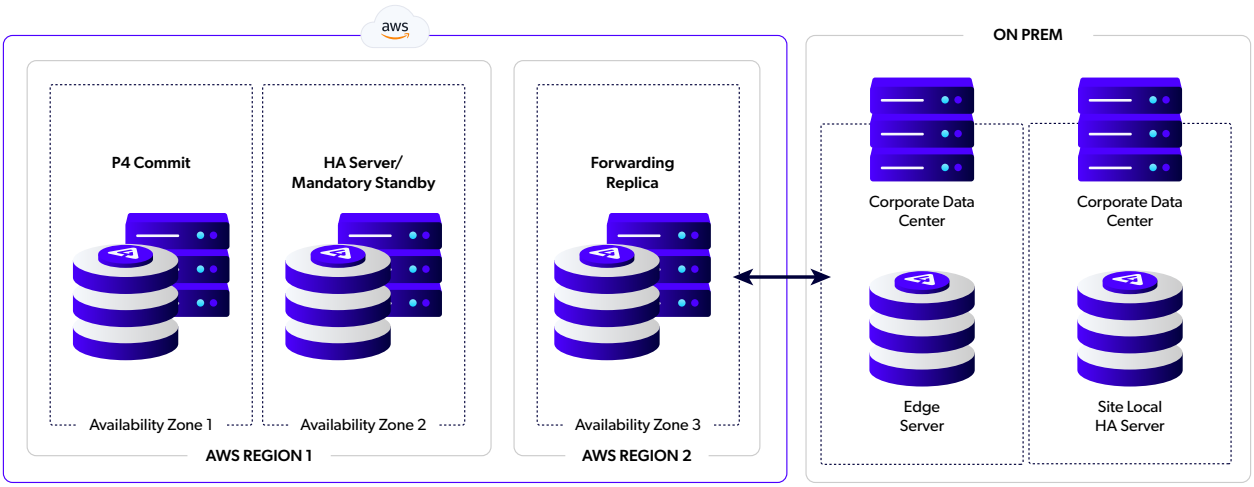

Figure 4 Example of hybrid topology (HA refers to a site-local high-availability server).

The figure above shows a sample topology that follows a hybrid cloud approach. It's designed to maximize the safety and availability of the master server while utilizing on-premises infrastructure and minimizing AWS data egress costs.

The following are sample hostnames and Perforce ServerID values. Nothing depends on the host naming convention described below; it is acceptable to substitute for a local host naming convention. It is generally a good idea that the AWS Name resource tag for the EC2 instance matches the hostname as set on the machine. While the host naming convention below is only a suggestion, the ServerID names are defined by the Server Spec Naming Standard in the SDP documentation.

Notes:

- Hostnames follow the pattern <P4ServerType>-<SiteTag|TeamTag>-NN where: - <P4ServerType> is p4 (for a commit server or standby of the commit), p4r (replica), p4p (proxy), p4b (broker), or p4e (edge server) - <SiteTag|TeamTag> identifies the location or team (e.g., 'awsuse2' for AWS us-east-2 region).
- NN is a numeric suffix (01, 02, etc.). These numbers initially are incremented where there are multiple machines of the same type, e.g. a commit, and two standbys may have –01, -02, and –03 suffixes initially. The numbers are retired when the EC2 instance and root volume are eventually replaced. For example, an older set of CentOS 7 machines with –01, -02, and –03 suffixes might be replaced by a trio of modern Ubuntu 24.04 machines with –04, -05, and –06 numbers. The suffixes get retired, not reused.
- Host names should avoid including words like "primary" and "backup" and should instead use number or letter suffixes.

`p4-awsuse2-01`
`commit.MyCo.fgs`
Commit server.
MyCo is the organization identifier, and fgs is the SDP instance name, a data set identifier.

`p4-awsuse2-02`
`p4d_ha_awsuse2`
High availability (HA) server, configured as a mandatory standby replica targeting the master.

`p4r-awssyd-01`
`p4d_fr_awssyd`
Forwarding replica in Sydney, Australia, targeting the master.

`p4e-syd-01`
`p4d_edge_syd`
Edge server on-premises in Sydney, Australia, targeting the forwarding replica in AWS.

`p4e-syd-02`
`p4d_fs_edge_syd`
Site-local HA server, configured as a standby server targeting the Sydney edge server.

This is because a failover can cause the roles of the machines to be reversed.

- The ServerID defines the current role of the P4 service on the given machine (for the given data set; this sample illustrates only a single data set).
- The ServerID values are standard, codified in the SDP mkrep.sh script. This standard should not be altered.
- End users will not need to know the server host names but will instead use a DNS host alias that is often the same as the server hostname, but with the numeric suffix and 'dash' characters removed, e.g. p4c-awsuse2 or p4e-syd (with the 'e' indicating an edge server).

Master/commit server. The .1 is the SDP instance name, a data set identifier. Forwarding replica in Sydney, Australia, targeting the master. Site-local HA server, configured as a standby server targeting the Sydney edge server be altered.

P4 commit and standby server machines (with the p4-commit prefix) do not indicate the specific role. This can change, and possibly even be reversed, in a failover situation. Host names have numeric suffixes in these examples, though some sites use letters instead (e.g. -a, -b).

P4 Ha Replication Details

The P4 commit servers will reside on a host named `p4-awsuse2-01`, 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 commit server will have a designated high availability (HA) server. For purposes of this guide, an HA replica is one that:

- Is running a supported version of P4 (P4D).
- Is in the same AWS region as its master server.
- Is in a different availability zone within the same AWS 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_awsuse2` for an HA server in the us-east-2 region in Ohio, USA.
- Has a `Services:` field of the server spec set to a value for `standby`.
- Has an `Options:` field of the server spec set to a value of `mandatory`.
- Has a db. replication setting or read-only.
- Has an `lbr.replication` setting as follows:
  - FSx for ONTAP, `lbr.replication` can have a value of shared indicating the servers are sharing the storage or that FSx for ONTAP technology is handling the replication rather than P4. Sharing eliminates the need for replication of archive files. Alternatively, the HA can have a separate FSx for ONTAP and defer replication (using FSx for ONTAP block-level replication technology) to provide faster replication of archive files.
  - For configurations using EBS, `lbr.replication` has a `read-only` value, indicating a full replica of metadata and archive files exists as a separate volume on the replica server.
- Has a `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.

P4 Disaster Recovery (Dr) Replication Details

In addition to an HA solution, a disaster recovery solution should also be in place. In this sample topology, a forwarding replica in the AWS Sydney data center serves as the DR solution. It also optimizes long-distance WAN traffic routing by leveraging AWS's global infrastructure.

TABLE 2 STORAGE CONFIGURATIONS

Mount	Name Tag	Base Size	Storage Notes
/p4depots	EBS: p4-commit-01-p4depots	EBS: 1TB	EBS: gp3, Encrypted
	FSx for ONTAP: p4-commit-01-p4depots	FSx-ONTAP: 16TB	FSx-ONTAP: Enable storage efficiencies
/p4logs	p4-commit-01-p4logs	24GB	gp3, Encrypted
/p4db	p4-commit-01-p4db	64GB	gp3, Encrypted
/p4depots	EBS: p4-commit-02-p4depots	EBS: 1TB	EBS: gp3, Encrypted
	FSx for ONTAP: p4-commit-01-p4depots	FSx-ONTAP: 16TB	FSx-ONTAP: Enable storage efficiencies
/p4logs	p4-commit-02-p4logs	24GB	gp3, Encrypted
/p4db	p4-commit-02-p4db	64GB	gp3, Encrypted

Caption/Legal - Lorem ipsum dolor sit amet, consectetur adipiscing elit aliqua. Proin gravida lorem vel ipsum dictum varius sed eget nisl. Morbi viverra iaculis luctus.

The EBS values are the defaults of the AWS CloudFormation template. For more information on the ESPs, https://www.perforce.com/products/helix-core/install-enhanced-studio-pack-aws

Server Deployment Package

The Perforce Server Deployment Package (SDP) is open-source software available from Perforce. It is used to manage a variety of P4 Platform topology components, including those deployed on EC2 instances in AWS as well as on-premises.

The SDP evolves with P4, providing an ongoing reference implementation of many best practices of various kinds, from routine maintenance (zero downtime checkpoints) to performance optimization and much more.

Storage Configuration

For purposes of this document, the key thing to understand about the SDP is its optimal storage layout. The SDP maintains three storage volumes for P4 (in addition to the OS root volume):

- `/p4depots` — 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.
- `/p4db` — Contains metadata databases only, frequently accessed with random I/O patterns. It must not be backed up directly, and it is constantly being written.
- `/p4logs` — Contains active journal, active server log, and various other logs as well as application temp storage (P4TMP). This does not need to be backed up. Before configuring the EC2 instance, create and configure storage volumes as noted in the following sections for EBS or FSx for ONTAP.

EBS Storage Configuration

Before configuring the EC2 instance, create and configure storage volumes as noted below. This is the set of storage volumes needed for both the master and HA replica EC2 instances. Here gp2 indicates AWS General Purpose SSD storage, and st1 indicates AWS Throughput Optimized HDD storage. The objectives:

- Spend the performance of gp2 where it will deliver the most value, e.g., meeting the low latency demands of the metadata volume.
- Use gp2 on the logs volume; the files are rotated frequently and shouldn't grow too large.
- Utilize cheaper st1 volumes where they will perform well, i.e. for long-stream read and write operations that commonly occur on the depot volume.

The base sizes are for a production development server. Actual sizes will vary and may be larger, especially for the /p4depots volume, which can easily be multiple terabytes (or more!) based on the scale of software to be developed with this server.

Amazon FSx for NetApp ONTAP Storage Configuration

Amazon FSx for NetApp ONTAP is a fully managed service that provides highly reliable, scalable, high-performing, and feature-rich file and block storage built on NetApp's popular ONTAP file system. FSx for ONTAP combines the familiar features, performance, capabilities, and API operations of NetApp file systems with the agility, scalability, and simplicity of a fully managed AWS service.

As a fully managed service, FSx for ONTAP makes it easier to launch and scale reliable, high-performing, and secure shared file storage in the cloud. With FSx for ONTAP, you don't have to worry about:

- Setting up and provisioning file servers and storage volumes
- Replicating data
- Installing and patching file server software
- Detecting and addressing hardware failures
- Managing failover and failback
- Manually performing backups

FSx for ONTAP also provides rich integration with other AWS services, such as AWS Identity and Access Management (IAM), Amazon WorkSpaces, AWS Key Management Service (AWS KMS), and AWS CloudTrail.

For more detailed information please visit: https://docs.aws.amazon.com/fsx/latest/ONTAPGuide/what-is-fsx-ontap.html

FSx for ONTAP can be used for all Perforce P4 storage volumes (/p4db, /p4logs, and /p4depots), but it is particularly recommended for the large /p4depots volumes. Because FSx for ONTAP is multi-protocol, both iSCSI and NFS can be used on the same shared storage.

- For P4 commit servers, multi-AZ scale-up file systems are recommended to achieve HA across AZ's in a single region for /p4depots. Replicating data
- For Edge and Build-Edge servers, both Single-AZ and Multi-AZ configurations can be considered depending on the level of performance and availability that is required.
- Use FSx for ONTAP Single-AZ and Multi-AZ for shared archives between commit and replica servers.
- If /p4db and /p4logs are considered for FSx for ONTAP, iSCSI with multiple connections is recommended. If not, EBS is the recommended AWS storage service type.
- For /p4depots on FSx for ONTAP, it is recommended to use iSCSI with multiple connections for the most optimal /p4depots performance, however, for increased ease of management with slightly less performance, NFS with nconnect is recommended and is generally the choice.
- For P4 commit server cost optimization, ensure FSx for ONTAP dedupe, compression, and compaction is enabled for the /p4depots volume.
- For Disaster Recovery and Archival destination deployments with Edge as the destination target, FSx for ONTAP SnapMirror replication can enable efficient cross-region replication. Snapshots must be enabled at the source. Depending on the RTO/RPO of the DR or Archival destination, data tiering, FSx for ONTAP's intelligent data movement from SSD to object, may also be enabled at the destination for further cost optimization of the DR/Archive data replica. It is not recommended to enable data tiering on the source/production P4 commit server at this time; however, it is not prevented.

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 Amazon EBS settings listed above.

EC2 Instance Configuration

The following information is useful for launching EC2 instances for P4 servers. The headings roughly match those seen when you select "Launch Instance" from the EC2 section of the AWS Console.

Choose Ami: Select A Rocky Linux Or Ubuntu Ami

Select the latest available Rocky Linux 9.x or Ubuntu 24.04 AMI to use. These distributions are recommended for optimal compatibility with Perforce SDP. Note that our Marketplace listing uses Rocky 8, which is also a suitable option if you're starting with our marketplace listing. We recommend visiting instances.vantage.sh to help you easily compare EC2 Instance features.

Choose Instance Type: Go With C6

Consider the following when selecting the Amazon EC2 instance type:

- Select an EC2 instance type in the compute-optimized family of instances, for example c7i , unless not available in a particular region. Memory-optimized instances may also perform well. However, on balance, compute-optimized is expected to deliver the best overall performance for P4 development. Faster processors help avoid bottlenecks with the large number of compression/ decompression operations typical when handling large digital assets.
- Select only an instance type with EBS-Optimized available.
- Select only an instance type with "Network Performance" indication of "Up to 10 Gigabit" or better.
- Select an instance with sufficient RAM. For example, for a customer developing software with total assets on the order of 8TB and 80 users, we might go with the **c7i.2xlarge** instance type, which has 16G RAM available.

Configure Instance Options

When configuring instance options, associate it with the VPC previously created for P4. Select the appropriate availability zone (subnet), being sure the master and HA servers are in the same region but with different subnets.

Ignore the following settings:

- Placement groups.
- Capacity reservation.

Check the following options:

- Protect against accidental termination.
- Enable CloudWatch for detailed monitoring.

The tenancy should be shared. We do not recommend using dedicated, as this is intended to apply to data workloads involving strict regulatory data isolation compliance requirements. The term dedicated in the context of AWS tenancy does not imply superior performance.

This is corroborated by detailed benchmarks done by an AWS customer. Note the following:

- There is a slight, but almost negligible, difference in CPU load. It's not much to begin with, and a preference of compute-optimized instance types makes it so even that small difference is completely absorbed.
- There is no impact on network performance or latency.
- There is no impact on memory access

Lastly, the elastic network interface should not be used with P4.

Configure Ec2 Root Storage Options

Only the instance root volume needed to hold the OS is created at instance creation time. The EBS storage volumes for the different P4-related /p4* volumes are created separately, as noted above.

After the first instance is created, subsequent instances can be more easily configured by selecting an existing instance from the AWS console to use as a template (e.g., the p4-commit-01 host), and selecting, "Launch more like this".

Note that the VPC and security groups should be created before launching the instance.

Add Tags

Add a name tag and have that name match the intended hostname of the machine, e.g., p4-commit-01.

Reserved Instances

When you launch EC2 instances, the standard option is On-Demand, which means you can start, stop, and terminate them at will. Once you have solidified your configuration, and you know you have specific servers that will be long-lived, you should replace those On-Demand Instances with Reserved 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 On-Demand price of the same configuration.

Availability Zones

An Availability Zone is an isolated location inside a region. Each region is made up of several Availability Zones. Each Availability Zone belongs to a single region. AZs (as AWS experts call them) are isolated from one another but are connected through low-latency links.

Ensure that the master and standby servers are in different EC2 availability zones, for optimal redundancy.

If leveraging FSx for ONTAP, Multi-AZ file system deployments will automatically provide cross-AZ redundancy for HA P4 data by providing a single shared /p4depots volume spread across two AZs.

Virtual Private Cloud (VPC) and Perforce License Files

Define a Virtual Private Cloud specifically with p4 in the VPC tag name, dedicated for usage by Perforce P4 components.

The *private* IP addresses of AWS 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 adjust to the following storage standards, as those are baked into the SDP. When moving to AWS and using EBS for depot storage, adopting the following standards is necessary to ensure all critical data is on an EBS volume, backed up, encrypted correctly, and on the cost-optimized storage solution.

The standards for depots backed by EBS volumes 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 /p4depots volume.
- Depot's specs should have only the default depot.
  Map: field value of: DepotName/…
- Both within the AWS environment and for the on-premises edge servers, server machines must be 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.

In AWS, this last point can be assumed as EBS storage can scale arbitrarily large. The goal here is to avoid an overly complex 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. If leveraging FSx for ONTAP, this is not a concern.

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

It is important to note that depots utilizing FSx for ONTAP can bypass the need to adopt these standards as this storage system manages the backup and encryption of all critical data.

SDP Replication Standards

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

Before 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 DNS name that will survive a failover, so should a replica. For example, the Sydney edge server would have a P4TARGET value like p4-commit:1666, but not p4-commit-01:1666 or p4-commit-02:1666. The failover procedure will redirect the p4-commit alias from the -01 to the -02 box. For greater clarity, DNS names can include an arbitrary tag for the business unit, team, or project using a P4 data set. This tag can replace the word commit in the DNS names and augment it in server hostnames. For example, the Friendly Greeting System project's commit server might use a DNS name of p4-fgs.orgname.corp or similar, and the commit server hostname might be p4-fgs-commit-01, and its standby p4-fgs-commit-02.

SDP Sample New 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-commit-01**:

```
source /p4/common/bin/p4_vars 1
mkrep.sh -i 1 -t edge -s syd -r p4e-awssyd-01
```

And its site-local HA replica command would be:

```
mkrep.sh -i 1 -t ha -s syd -r p4e-awssyd-02
```

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

Then, load that seed checkpoint on the edge as **perforce@p4e-awssyd-01**:

```
/p4/1/checkpoints/p4_1.edge_syd.seed.ckp.1234.gz
```

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

SDP Tweak For P4depots Snapshot

When deployed in AWS, the SDP "Instance Vars" file can be tweaked to add a value for the SNAPSHOT_SCRIPT. The goal of this optional adjustment is to make it so that the snapshot for the `/p4depots` volume occurs at the optimal point in time, immediately after the P4 checkpoint operation completes. This achieves the minimum lag between the time the digital asset recovery is created and the time it is whisked away to greater safety. For configurations using FSx for ONTAP, a custom script can be provided, as `/p4/common/site/bin/netapp_snapshot.sh`. This script will make the appropriate CLI call to request a snapshot. If this tweak is not used, scheduling based on expected checkpoint completion times is valid as well.

FSx for ONTAP snapshot creation happens automatically based on the snapshot policy configured on the filesystem. By default, each volume is linked to the filesystem's default snapshot policy, which includes:

- A maximum of six hourly snapshots taken five minutes past the hour
- A maximum of two daily snapshots taken Monday through Saturday at 10 minutes after midnight
- A maximum of two weekly snapshots taken every Sunday at 15 minutes after midnight

To ensure point-in-time consistency with P4 checkpoints, the checkpoint creation schedule can be synchronized with the filesystem's snapshot policy.. The SDP backup_functions.sh script can be adjusted to verify the snapshot with a minimal time difference from the checkpoint creation.

Custom snapshots can be manually created for FSx for ONTAP filesystems using the ONTAP CLI or other automation tools. For more information on FSx for ONTAP snapshots, refer to the FSx for ONTAP snapshot management documentation for details.

For configurations using EBS volumes for /p4depots, this is accomplished using AWS command line tools, doing an aws ec2 create-snapshot call. The call might look like:

```
perforce@p4-commit-01:/home/perforce aws ec2 create-snapshot --description "Backup of /p4depots on $(date)." --volume-id vol- aabb6c5e5a1c6cd8c
```

In this example, the `volume-id` specified would be the volume ID of the EBS volume mounted as the `/p4depots` volume on the given EC2 instance. Similarly, the root volume should also be snapshotted.

The /p4db volume should not be snapshotted. Snapshotting the /p4logs volume is not usually done, as the most important data is in the active P4JOURNAL file that 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 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, where the ccheck.sh "configurable check" script is described. The ccheck.sh script in the latest version of the SDP will always have current best practices. The following table lists sample values:

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

Setting Name	Recommended Value	Comments
net.autotune	1	Performance
db.monitor.shared	4096	Performance
net.backlog	2048	Performance
lbr.autocompress	1	Safety, Performance
lbr.bufsize	1	Safety, Performance
filesys.bufsize	1M	Performance
server.start. unlicensed	1	Licensing
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	P4 Code Review
dm.keys.hide	2	P4 Code Review
filetype.bypasslock	1	P4 Code Review

Compliance with security best practices for configurables on an existing server can easily be done by calling ccheck.sh -sec; call ccheck.sh -man for full documentation on how to check and even set security configurables, with specific guidance for configurables that may potentially be impactful to change. We also advise reviewing Securing and Hardening Your P4 Server in Today's Security Landscape

P4 Management System (P4MS)

The P4 Management System (P4MS) 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 P4 instance runs on a bastion host, p4ms.p4demo.com. Its SDP instance name is p4ms. It manages the SDP on all hosts where any P4 Platform topology components exist. This includes P4D commit servers, replicas, edge servers, proxies, and brokers, and P4 for plugins like Photoshop, Maya, 3ds Max, and more. Tight-ship management keeps the SDP scripts and configuration files current and in sync on all hosts, using P4 to deploy and verify files.

P4 Topology Definition

A single P4 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 P4 Topology configuration file.

Centralized Management

The P4 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 P4 Topology configuration file). This allows any instance's P4D (or any other component) to be stopped, started, or have its status checked from the P4MS server, with a command like these examples:

```
p4ms status 1:commit
p4ms stop 1:p4d_edge_syd
```

Failover Plan Definition

P4MS 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 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 like this:

```
p4ms failover local i:1 u
```

HA Failover Of AWS 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. Users and traffic must be redirected for HA failover plans.

HA failover of the AWS master can be executed with a command like:

```
p4ms failover ha i:1 u
```

Under the covers, the p4ms script executes the Perforce commands necessary to achieve promotion of the failover machine to the master machine, e.g. using the p4 failover command.

Failover Plan	From/To	Sample Usage Scenario
Commit Server Local	p4-awsuse1-01 (same host)	Use when host p4-awsuse1-01 is OK, but databases are corrupted, e.g., due to sudden power loss or human admin error. Failover recovers using offline databases on the same host.
Commit High Availability (HA)	p4-awsuse1-01 -> p4-awsuse1-02	Use when host p4-awsuse1-01 is unusable, or needs to be taken offline (e.g., to upgrade RAM in the EC2 instance). Failover recovers to a nearby standby replica.
Commit Disaster Recover (DR)	p4-awsuse1-01 -> p4-awsusw2-01	Failover to a Disaster Recover server far away in another AWS region. Failover recovers to a standby replica far from the commit server.
Sydney Edge Local	p4e-awssyd-01 (same host)	Use when host p4e-awssyd-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	p4e-awssyd-01 -> p4e-syd-02	Use when host p4-awssyd-01 is unusable, or needs to be taken offline (e.g., to upgrade RAM). Failover to a site-local standby replica.

P4MS 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 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

Use when host p4e-awssyd-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.

Following execution of the p4ms script, a corporate network DNS change must be made to change the p4-awsuse2 host alias, such that traffic routes to p4-awsuse2-02 instead of p4-awsuse2-01, to complete failover.

Because the HA server in AWS 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 like:

```
p4ms failover syd-edge-ha i:1 u
```

Following this command, a corporate network DNS change must be made. This changes the p4e-syd DNS alias used to refer to the Sydney edge server from `p4e-syd-01` to `p4e-syd-02`.

AWS Security Group Configuration

Network firewall rules should be considered when using AWS Security Groups. A security group should be created with the tag name of P4. It should be enabled with ports opened as identified below.

If you're running P4 Code Review (formerly Helix Swarm) on the same EC2 instance as P4, make sure to add an HTTPS rule that opens port 443 in the Perforce P4security group. However, the recommended approach is to host P4 Code Review on a separate EC2 instance with its security group. This security group should only open port 443 and connect to a P4 server using its private IP.

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 functionalities to P4 administrators but also introduce an extra topology component to maintain and upgrade.

If FSx for ONTAP is used for the `/p4depots` volume (optional), then add an NFS inbound rule, named FSx for ONTAP, with the source being the ID of the security group itself.

Note: Brokers offer valuable features, such as enhanced security and DLP, especially in larger environments. In smaller setups, they might not be necessary, and the overhead may not be justified.

P4MS Hub And Spokes

The P4MS server is the one P4 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 P4-related machines, in and out of AWS.

As such, this server should be configured, from an access control perspective, as a bastion host, with the highest available security. It should ideally be deployed in AWS as a separate EC2 instance from server instances containing development work, though this host can also be deployed on-premises.

The P4MS server (p4ms.p4demo.com) must allow traffic as follows:

- Inbound on port 7467 (always SSL encrypted) from all p4d servers managed by P4MS.
- Inbound on port 7468 (always SSL encrypted) from all p4d servers managed by P4MS.

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 P4MS server (a Linux bastion host).

Inbound Rules For Aws Servers

AWS servers require inbound access as follows:

- Into the VPC on port 1999 (optionally SSL encrypted) from edge server hosts and their standby machines.
- Into the VPC on port 1666 (optionally SSL encrypted) from the P4MS server.
- Into Port 22 (SSH) from the P4MS 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 incorporate trust and login logic into P4 interactions in code, as best practice.

- In some cases, P4 servers under heavy load may — due to SSL timeouts — drop connections, resulting in user errors for conditions that may otherwise have been successfully completed after a temporary pause. Similarly, potentially useful P4 ease of administration: Host and Port Access List for Perforce P4 Servers.

Perforce SSL Encryption

Perforce provides SSL encryption that can add an extra layer of defense, protecting data in transit across the network.

SSL can be used to enhance security. Benchmarks indicate that the performance costs for using SSL are minimal, often negligible.

SSL may not be required if good perimeter security is in place, including within AWS, on-premises, and between AWS and the on-premises infrastructure.

However, SSL does introduce a few complexities:

- As with any OpenSSL technology, certificates need to be generated and managed. P4 server products (p4broker, p4d, p4p) can generate their certificates, so this complexity is limited to dealing with occasional certificate expiration.
- Failover solutions are somewhat more complex and potentially less transparent with SSL, as the goal of failover — to transparently change the server that a user is connected to — is opposed to the goal of SSL, which is to ensure users are aware when a server they think they are communicating with changes. Solutions include getting all users to trust all possible failover target servers, or even offboarding SSL handling entirely from P4 using an HA proxy. Customers may also simply accept that failover may require users to trust the new server after failover. This approach may involve retooling for robotic users to incorporate trust logic into P4 interactions in code.
- When P4MS is deployed on a bastion host, the P4MS instance is always configured with SSL, as this server can contain sensitive information that could be used to access other “real data” instances.

Storage Encryption

If using EBS storage, EBS provides encryption of essential data at rest for the P4 servers.

If leveraging FSx for ONTAP, all FSx for ONTAP file systems are encrypted at rest with keys managed using AWS Key Management Service (AWS KMS). Data is automatically encrypted before being written to the file system and automatically decrypted as it is read. These processes are handled transparently by Amazon FSx, so you don't have to modify your applications. Amazon FSx uses an industry-standard AES-256 encryption algorithm to encrypt Amazon FSx data and metadata at rest.

FSx for ONTAP additionally supports the following methods for encrypting data in transit between FSx for ONTAP file systems and connected P4 servers:

- Automatic Nitro-based encryption over all supported protocols and clients running on supported Amazon EC2 Linux and Windows instance types.
- Kerberos-based encryption over NFS and SMB protocols.
- IPsec-based encryption over NFS, iSCSI, and SMB protocols

All of the supported methods for encrypting data in transit on FSx for ONTAP use industry standard AES-256 cryptographic algorithms that provide enterprise strength encryption.

Other Rules

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

What's Next?

Evolution

The AWS infrastructure and managed services offerings continue to evolve as AWS and its partners continue to innovate. This guide will continue to evolve to provide ongoing guidance for customers looking to deploy P4 on AWS for development.

The following are being considered:

- AWS CloudFormation will be utilized.
- SDP and P4MS improvements may simplify operations in AWS, eliminating any tweaks.
- AWS Route 53 will be explored.
- A reference implementation may be captured with CloudFormation, including an illustration of backup of EC2 snapshots and the move to Glacier.
- 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 AWS.
- 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, CloudFormation 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 development studio, in a remote location with respect to master server.

Parameterized CloudFormation templates could be employed to spin up edge servers with a subset of depots to enable limited collaboration for the new studio.

SDP Cloud Deploy

The SDP Cloud Deploy project is an early-stage prototype implementation demonstrating the use of simple Terraform template files to build a sample P4 environment, including building a replica from scratch while starting with an AWS account.

Terraform Deployment Options

While this guide focuses on manual deployment best practices, Perforce also offers Terraform templates to automate P4 infrastructure deployment on AWS. These templates provide production-grade infrastructure as code solutions for customers who prefer a more automated approach to cloud deployments. Our Terraform modules support flexible configuration options including:

- Complete P4 server infrastructure deployment
- High availability configurations
- Multi-region replication
- Security group and networking setup
- Integration with FSx for ONTAP and EBS storage
- For access to our latest Terraform deployment templates or to discuss how they can support your specific deployment needs, please contact our professional services team.

Additional Comments

Review additional details regarding decisions for this document.

1Ideally, consistent site tags should be used in P4 server spec names and host names, e.g., awsuse2 as a tag for the AWS us-east-2 region in Ohio, USA. For an "all in" AWS topology, the "aws" portion of the site tag is often dropped as redundant. The site tag would then simply be use2, a short form of us-east-2.

About Perforce

The best-run DevOps teams in the world choose Perforce. The Perforce suite is purpose-built to develop, build, and maintain high-stakes applications. Companies can finally manage complexity, achieve speed without compromise, and run their DevOps toolchains with full integrity. With a global footprint spanning more than 80 countries and including 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 Today ▶

Contact Support ▶