Helix Management System (HMS) Deployment Planning Guide

Perforce Professional Services

Version v2020.2, 2020-09-17

Table of Contents

Introduction

This document discuss various factors to be considered when planning a Perforce Helix Management System (HMS) deployment.

HMS Server Host Name

The HMS server is typically named something like `p4hms.corp.company.com` with `p4hms` as the short name. This short name is assumed in samples in this document.

Topology and Port Targets

This section describes a typical HMS deployment topology. Note that various aspects of this topology are recommended, but optional. When using HMS for "tight ship" management, HMS does not rely on using the topology recommendations below. When using HMS to simplify execution of failover, there are dependencies on following topology guidelines as suggested below, including the use of brokers as described. (Such dependencies could be removed with minor customization or new features).

Recommended Stack Topology

A typical HMS deployment uses Helix Brokers to simplify control of access to production systems, and to keep users (humans and robots) out of production systems during maintenance activity. This prevents any possible loss of user work in event of an abort of maintenance activities, and prevents other undesirable complexity during maintenance activity.

Brokers are popular as they enable a wide array of policy enforcement options. Such usage is made simpler with HMS, but is outside the scope of this document.

A typical HMS deployment includes `p4broker` processes deployed in "stack topology," such that brokers are deemed part of the technology stack alongside `p4d` on every machine that runs a `p4d` process (be it a master/commit server, a replica, or an edge server). This broker port is typically open to the network, as it is the customer-facing port. Non-customer-facing `p4d` instances, such as read-only `standby` replicas, will have a broker configured, though it typically is not started until a failover occurs.

Down For Maintentance (DFM) Mode

In maintenance windows, various brokers in a global topology are used to broadcast a "Down For Maintenance" (DFM) message to users. The message indicate to users that the system is not available for general usage, and rejects all requests. The message to users can be configurable. It can (for example) direct users to an internal wiki page with a summary of the maintenance being performed, expected return to service, etc. And perhaps reminders of the notifications users may have forgotten to read about the coming service outage.

HMS automates enabling and disabling DFM mode for a global topology, and relies on brokers to do so.

Customer-Facing and Private Ports

Where the `p4broker` runs on a port familiar to and accessible by users (often 1666), `p4d` runs on a private, "for admins only" port (often 1999). Port numbers other than 1666 and 1999 are commonly used, and always used when Helix multiple instances are run on a machine.

The key concept is to have a distinction between customer-facing (`p4broker`) and private (`p4d`) ports.

All processes, whether coming from the network or even those running on the same `p4d` server

machine, should hit the customer-facing broker port, with a few exceptions:

- Other `p4d` servers in the same topology, for purposes of replication set their P4TARGET to bypass the broker and go direct `p4d`-to-`p4d`.
- The `p4hms` server, for monitoring purposes.
- Trigger scripts should target the `p4d` port directly, bypassing the broker. Note that triggers only run master servers and edge servers, but are deployed to other replicas (along with any dependencies) to support failover. In maintenance mode, triggers won't fire as there is no activity to cause them to fire.
- Sanity checking during maintenance windows. In some cases, admins and selected users participating in sanity checking, such as after a system upgrade, may need to the p4d server for testing purposes, even when the brokers are broadcasting DFM messages for users.

The target port of integrated systems target the customer-facing broker port. Some examples:

- Continuous Integration (Build/Test/Deploy) Systems
- Helix Swarm
- Proxy servers
- P4DTG, the Perforce Defect Tracking Gateway, which is typically deployed on the same machine as the master Helix Core server machine.

Firewall rules can enforce that `p4d` ports are open on p4d servers from other p4d servers associated with the same instance, and from the `p4hms` server, while the customer-facing ports are generally open to the internal network (and in some cases truly "public", if accessible outside the local network).

Proxy Servers

Proxy servers typically run on separate server machines from `p4d` servers, and their P4TARGET setting points them to the `p4broker` port on the "nearest" (or otherwise most appropriate) `p4d` server machine (which may in fact not be physically nearby, given a primary role of proxies is to improve performance for remote users. The numeric value of the port is typically the same as that used for the customer-facing broker, though it can be something else.

TCP Ports for Network and Host Firewall Rules

The following documents network firewall ports to be opened for a Perforce Helix Management System (HMS) server deployment. This can also guide host-local firewall rules if host local firewall rules are deployed in addition to (or instead of) network firewalls. Network firewall rules are generally easier to manage as they are centralized, and are the preferred technical approach. In some cases, Perforce administrators may choose to rely on host firewall rules that are more likely to be under their direct control.

Note: Using firewall rules (network and/or host-local) is recommended due their security benefits. When used, HMS requires certain ports to be opened.

Table 1. TCP Ports for HMS

Port	Service	From/To	Notes
22	SSH	From `p4hms` to all managed Helix servers.	
22	SSH	To `p4hms` from where HMS admins want to SSH from.	
7467	P4Broker	To `p4hms` from all managed Helix servers.	Recommended but optional.
7468	P4D	To `p4hms` from all managed Helix servers.	
7467	P4Broker	From where HMS admins want to manage from.	Recommended but optional.
7468	P4D	From where HMS admins want to manage from.	
7427	HAS	To `p4hms` from where HMS admins want to auth with HAS from.	If using the Helix Authentication Service (HAS).
P4Broker_Port	P4Broker	From `p4hms` to each managed `p4broker`, using managed broker P4PORT, often 1666. This customer-facing port is usually open to the internal network.	

Port	Service	From/To	Notes
P4D_Port	P4D	From `p4hms` to each managed `p4d`, using managed `p4d` P4PORT, often 1999.	
P4D_Port	P4D	From all `p4d` servers related to the same Helix Core instanct to all other `p4d` related to the same data set, using its `p4d` P4PORT, often 1999.	
P4Proxy_Port	P4P	From `p4hms` to each managed `p4p`, using managed `p4p` P4PORT, often 1666.	
443	Swarm/HTTPS	To `p4hms` from where HMS admins want to manage from.	Assumes HMS will have a dedicated Swarm on same box (recommended).
9090	Prometheus	To `p4hms` from where HMS admins want to manage from.	
3000	Grafana	To `p4hms` from where HMS admins want to manage from.	
9100	Node Exporter	From `p4hms` to all managed Helix server machines.	

Single Host with Basic Backup

We strongly recommend HA (High Availability) and Disaster Recovery (DR) solutions be deployed for managed Helix Core instances. And indeed, HMS plays a role in comprehending such topologies and supporting execution of failover processing for the Helix Core instances.

However, for the HMS server machine itself and the tiny Helix Core `p4d_hms` instance that lives on it, typically only a traditional machine backup is done. For example, a virtual machine with daily snapshots of the root volume is sufficient. Typically replication features of the tiny `p4d_hms` Helix Core instance are not used because:

- While HMS supports failover of other instances, the tiny `p4d_hms` instance doesn't play a role in the actual execution of failover. HMS is designed such that the critical role it plays in supporting failover is accomplished on the "Happy Days" between failovers, ensuring that all scripts and configuration files are managed "tight ship" style, such that there is no version skew across managed server machines for scripts and configuration file that could impact the success of failover. So while the HMS server plays a critical overall role in failover, availability of the `p4d_hms` instance does not play a role in actual execution of failover, and thus does not impact the capability to execute a failover for other instances.
- An HMS server typically has only a small number of human users, those few Perforce Helix administrators entrusted with the "keys to the kingdom" for all managed Helix Core servers that typically contain an organizations "real data". From the perspective of an HMS user, other Helix Core instances are "customer-facing" (i.e. considering their internal business units using Helix Core as customers of the team responsible for Perforce Helix administration).
- An HMS server server does not grow very large, as it only manages scripts and a small number of binaries (e.g. p4d executables) that are updated infrequently. Thus, it can be typically recovered very quickly with traditional backup and recovery solutions, in a time frame that could meet typical RPO and RTO needs for a back-end server with few users.

That said, some sites do apply Helix Core replication to provide additional failover practice opportunity for a real but non-customer-facing Helix Core server.