SDP Legacy Upgrade Guide (for Unix)

Perforce Professional Services

Version v2020.1, 2021-01-29

Table of Contents

Preface . . . 1
1. Overview . . . 2
1.1. Upgrade Order: SDP first, then Helix P4D . . . 2
1.2. Upgrading Helix P4D and Other Software . . . 2
1.3. SDP and P4D Version Compatibility . . . 2
1.4. SDP Upgrade Methods . . . 2
2. Planning . . . 4
2.1. Mount Point Names . . . 4
2.2. Operating System User . . . 5
3. Upgrade Procedure . . . 6
3.1. Preparation . . . 6
3.1.1. Acquire Downloads . . . 6
3.1.2. Deploy new SDP common files . . . 6
3.1.3. Generate new SDP config files . . . 6
3.1.4. Configure new SDP instance bin files and symlinks. . . . 7
3.1.5. Determine Your Metadata Symlink Type (Fixed or Variable) . . . 7
3.1.6. Upgrade *_init scripts . . . 8
3.1.7. Review systemd service files . . . 8
3.1.8. Account for customization and additions (if any) . . . 9
3.2. Execution . . . 9
3.2.1. Stop Services . . . 9
3.2.2. Move old SDP Aside . . . 10
3.2.3. Upgrade Physical Structure . . . 10
3.2.4. Replace Instance Symlink with Directory . . . 11
3.2.5. Convert Fixed to Variable Metadata Symlinks . . . 11
3.2.6. Replace Instance Symlink with Directory . . . 12
3.2.7. Put new SDP common files in place. . . . 13
3.2.8. Put new SDP instance bin files in place. . . . 13
3.2.9. Upgrade systemd service files . . . 13
3.3. Post Operation Steps . . . 14
3.3.1. Cleanup . . . 14
Appendix A: Custom HMS Managed Installations . . . 14

Preface

This document provides an overview of the process to upgrade the Perforce Helix Server Deployment Package (SDP) from any older version (dating back to 2007) to the SDP 2020.1 (aka "r20.1") release.

If your SDP version is 2020.1 or newer, refer to the SDP Guide (Unix) for instructions on how to upgrade from SDP 2020.1 to any later version. Starting from SDP 2020.1, the upgrade procedure for the SDP is aided by automated and incremental upgrade mechanism similar to p4d itself, capable of upgrade SDP from the current release to any future version so long as the current release is SDP 2020.1 or newer.

This document describes the process of upgrading to SDP r20.1.

Chapter 1. Overview

The SDP software package, just like the Helix Core software it manages, evolves over time and requires occasional upgrades to remain supported. Further, patches may be released over time.

This document discusses how to upgrade the SDP, and when in relationship to Helix Core itself.

1.1. Upgrade Order: SDP first, then Helix P4D

The SDP should be upgraded prior to the upgrade of Helix Core (P4D). If you are planning to P4D to or beyond P4D 2019.1 from a prior version of P4D, you *must* upgrade the SDP first. If you run multiple instances of P4D on a given machine (potentially each running different versions of P4D), upgrade the SDP first before upgrading any of the instances.

The SDP should also be upgraded before upgrading other Helix software on machines using the SDP, including p4d, p4p, p4broker, and p4 (the command line client on the server machine). Even if not strictly required, upgrading the SDP first is **strongly** recommended, as SDP keeps pace with changes in the p4d upgrade process, and can ensure a smooth upgrade.

1.2. Upgrading Helix P4D and Other Software

See the SDP Guide (Unix) for instructions on how to upgrade Helix binaries in the SDP structure after the SDP has been upgraded to 2020.1 or later.

1.3. SDP and P4D Version Compatibility

The SDP is often forward- and backward-compatible with P4D versions. However, for best results they should be kept in sync by upgrading SDP before P4D. This is partly becuase the SDP contains logic to upgrade P4D, which can change as P4D evolves.

The SDP is aware of the P4D version(s) it manages, and has backward-compatibility logic to support older versions of P4D. This is guaranteed for supported versions of P4D. Backward compatiblity of SDP with older versions of P4D may extend farther back, though without the "offically supported" guarantee.

1.4. SDP Upgrade Methods

There are several methods for upgrading to a new version of the SDP:

- **In-Place, Manual**: Manual upgrades of the SDP must be peformed to upgrade from versions older than r20.1 in-place on existing machines. This document provides details on how to do this.
- **In-Place, Automated**: Automated in-place upgrades can be done if your current SDP version is r20.1 or later. Refer to documentation in SDP Guide (Unix) for upgrading from SDP r20.1 onward.
- **Migration-Style**: A migration-style upgrade is one in which the existing server machines

(virtual or physical) are left in place, and brand new "green field" machines are installed fresh using the Helix Installer (which installs the latest SDP on a "green field" baseline machien with only the operating system installed). Then the Helix Core data is migrated from the existing hardware to the new hardware. This approach is especially appealing when upgrading other aspects of the infrastructure at the same time as the SDP, such as the hardware and/or operating system.

- **Custom with HMS**: The Helix Management System is used by some customers. See Appendix 3.A, *Custom HMS Managed Installations*

Chapter 2. Planning

Legacy SDP upgrades require some familiarization that should be done prior to scheduling an upgrade.

Key questions to answer during planning are:

- Will p4d need to be taken down?
- If so, how long?

Most upgrades to SDP to r20.1 will require downtime for the Helix Core server, even if p4d is not being upgraded. The only exception to requiring downtime is your current SDP is r2019.1 or later AND your SDP is not conifgured to use `systemd`, i.e. does not have `/etc/systemd/system/p4_*.service` files.

If the current SDP version is r2019.1 or later, the downtime can be brief. The scripts can be upgraded while the server is live, stopping p4d only to change the `systemd *.service` files with new ones.

If your SDP is older, mount point names and other changes may be needed, which will extend the downtime for p4d needed to upgrade the SDP. Read on to get a sense for what steps are requried.

2.1. Mount Point Names

You will need to be aware of your moint point names. While referred to as mount point names in this document, in any given installation any or all of the three SDP "mount points" may be simple directories on the root volume or some other volume. In some installations, creative liberties were taken to use fewer than three volumes, and in some cases the operating system root volume was used as one of the volumes. Investiage and be aware of how your installation was configured.

In the examples below, the modern SDP mount point names are used:

- `/hxdepots` - Volume for versioned files and rotated/numbered metadata journals.
- `/hxmetadata`- Volume for active and offline metadata
- `/hxlogs` - Volume for active journal and various logs.

Depending on the version of the SDP, the above values may be used, or earlier defaults such as, `/depotdata`, `/metadata`, and `logs`. However, often customer sites changed from the defaults to custom values, such as `/depots`, `/p4db`, `/p4journal`, and others.

In the sample steps in this document, adapt the steps to use your local values for mount point names to the new values.

If your site uses two volumes for metadata, `/hxmetadata1` and `/hxmetadata2`, continue using the same names below.

2.2. Operating System User

You will need to be aware of your operating system user that `p4d` runs as in your environment.

The sample steps below assume that Perforce runs as the `peforce` operating system user, which is typical. Adapt if your user is something else, such as `p4admin` or `p4super`.

In modern installations, the default home directory is `/home/perforce`, though in some installations the home directory is `/p4`. In either case, this does not need to be changed during the upgrade process.

Chapter 3. Upgrade Procedure

After Chapter 2, *Planning*, the SDP procedure can be planned in detail.

The procedure is broken into 3 phases, Preparation, Execution, and PostOp. Preparation steps can be done in a non-disruptive manner on a production server ahead of the Execution. Execution steps are generally performed in a scheduled maintenance window. PostOp steps are done some time after the upgrade is complete, perhaps days or weeks later.

3.1. Preparation

Preparation steps are:

1. Acquire Downloads.
2. Deploy new SDP common files.
3. Generate new SDP config files.
4. Configure new SDP instance bin files and symlinks.
5. Determine Metadata Symlink Type (Fixed or Variable)
6. Account for customization (if any)

3.1.1. Acquire Downloads

Download the latest SDP tarball release from this link: https://swarm.workshop.perforce.com/projects/perforce-software-sdp/download/downloads/sdp.Unix.tgz.

Copy the downloaded tarball to the machine and put it in `/hxdepots/sdp.Unix.tgz`. (If a file with the same name exists from a previous upgrade, move it aside first.)

3.1.2. Deploy new SDP common files

```
mkdir /hxdepots/new
cd /hxdepots/new
tar -xzf /hxdepots/sdp.Unix.tgz
cat sdp/Version
```

Verify that the contents of the `Version` file are as expected.

3.1.3. Generate new SDP config files

The following SDP config files are generated and should be reviewed, comparing new files generated with a `*.new` extension with the files in the existing installation (without the `.new` suffix).

- `/p4/common/bin/p4_vars.new` - The main SDP shell environment file.
- `/p4/common/config/p4_N.vars.new` - Instance-specific files, one per instance.

- /p4/common/config/p4_N.review.cfg.new (if used, often Swarm configured with honor_p4_reviews makes them better)

Broker config files, p4_N.broker.cfg, may also exist in /p4/common/config. These do not need to change due to an SDP upgrade, and can be ignored.

The following templates are available:

- /p4/sdp/Server/Unix/p4/common/config/p4_vars.template - Template from which to create /p4/common/bin/p4_vars.new.
- /p4/sdp/Server/Unix/p4/common/config/instance_vars.template - Template from which to create /p4/common/config/p4_N.vars.new.
- /p4/sdp/Server/Unix/p4/common/config/p4review.cfg.template - Optional template from which to create /p4/common/config/p4_N.review.cfg.new.

The SDP config files sometimes contain local custom modifications made by administrators. Often the need for customization goes away with new SDP versions. However, when generating new config files, be sure to review old files for any custom values.

3.1.4. Configure new SDP instance bin files and symlinks.

If the current SDP is 2018.1 or newer, skip this section.

Examine the p4d_N_init script in the 'instance bin' folder, /p4/N/bin.

Does the actual code look like this sample (with comments and the "shebang" #!/bin/bash line removed)?

```
export SDP_INSTANCE=N
/p4/common/bin/p4d_base $SDP_INSTANCE $@
```

If the p4d_N_init script already looks like this, then the 'instance bin' folder does not need to be touched during the upgrade process.

If, however, the *_init script has more code, then all the p4*_init scripts will need to be upgraded during the upgrade execution. Templates are available in /p4/common/etc/init.d. The templates contains a few values that will need to be replaced.

3.1.5. Determine Your Metadata Symlink Type (Fixed or Variable)

Login as the perforce operating system user, and run this command:

```
ls -l /p4/1/root /p4/1/offline_db
```

The root and offline_db will be symlinks.

Depending on how old the SDP is, the structure will either be *fixed* or *variable* metadata symlinks. Determine which you have.

Variable Metadata Symlink References

If one of the symlinks points to a directory ending in `db1`, and the other in `db2` (it doesn't matter which is pointing to which), you have **variable metadata symlinks**.

Fixed Metadata Symlink References

If the target of the `root` and `offline_db` symlinks points to directories ending in the same names, i.e. `root` and `offline_db`, then you have **fixed metadata symlinks.**

3.1.6. Upgrade *_init scripts

The format of SDP init scripts may have changed since your legacy version. Check them to see if they need to be modified.

For each instance, look in the `/p4/N/bin` folder, and review the scripts. Compare them to templates in `/p4/common/etc/init.d`. For example, compare `/p4/1/bin/p4d_1_init` with `/p4/common/etc/init.d/p4d_instance_init.template`.

If your current init scripts look *exactly* like the templates, except for subsitutions of any `REPL_*` strings from the template, then they do not need to be updated. Older SDP versions had more complex `*_init` scripts.

If they need to be replaced, plan to do so during your upgrade with steps like these samples:

```
cd /p4/N/bin
mkdir OLD_DELETE_ME_LATER
mv p4d_N_bin OLD_DELETE_ME_LATER/.
sed s:REPL_SDP_INSTANCE:N:g /p4/common/etc/init.d/p4d_instance_init.template >
p4d_N_init
chmod +x p4d_N_init
```

If there are `p4broker_N_init`, `p4d_N_init`, and/or `p4dtg_N_init` scripts, follow the same procedure for those, generating new init scripts from the templates.

These steps can only be executed after the `/p4/common` folder has been updated.

3.1.7. Review systemd service files

The format of Systemd service files (sometimes referred to as 'Unit' files) changed with the SDP 2020.1 release. As part of planning, it is helpful to identify if systemd is already in use, and which Perforce Helix services are managing with systemd.

You can get a list of such services with:

```
ls -ld /etc/systemd/system
ls -lrt /etc/systemd/system/p4*.service
```

If the `/etc/systemd/system` directory exists, then the Systemd init mechanism is available. On systems that use the Systemd init mechanism, we recommend using it. Once systemd is configured for any given service, the SDP requires using the systemd mechanism (i.e. the `systemctl` command) to start/stop Perfore Helix services (for safety and consistency of management). Depending on your SDP version and how it was installed, there may or may not already be `p4*.service` files.

In any case, in the Execution phase below, new systemd `p4*.service` files will be put in place, which may be new or replace existing files.

3.1.8. Account for customization and additions (if any)

If the SDP has been customized in your environment, custom upgrade procedures may be required. An understanding of what was customized and why will be useful in determining if custom ugprade procedures are required.

In typical deployments, the SDP is not customized, or only customized in some way that is no longer needed due to improvements in the "stock" SDP.

In many cases, customers have added custom trigger scripts into the SDP structure. In this case, the script files may be moved around in the SDP structure during the uprade process, but should not need to be changed.

If you need help determining if and how the SDP was customized in your environment, Perforce Consulting may be of assistance. Note that customizations are not supported.

3.2. Execution

This section outlines sample steps for executing an actual upgrade after Chapter 2, *Planning* and Section 3.1, “Preparation” have been completed. The following is typically performed in a scheduled maintenance window.

Execution steps are:

1. Stop Services
2. Move Old SDP aside.
3. Upgrade Physical Structure
4. Put new SDP common files in place.
5. Put new SDP config files in place.
6. Put new SDP instance bin files in place.

3.2.1. Stop Services

Stop the `p4d` service for all instances on this machine. Also stop all p4broker services running on

this machine (if any).

For this short maintenance, the broker cannot be left running (e.g. to broadcast a "Down For Maintenance (DFM)" message) because the structure change cannot be started until *all* processes launched from the SDP directory structure have stopped.

Sample commands:

```
p4d_1_init status
p4d_1_init stop
p4d_1_init status

p4broker_1_init status
p4broker_1_init stop
p4broker_1_init status
```

The extra `status` commands before and after the start/stop commands are for situational awareness. These are not strictly necessary.

3.2.2. Move old SDP Aside

First, move the old SDP common files aside, like so:

```
cd /hxdepots/p4
mv common OLD.common.$(date +'%Y-%m-%d')
```

Next, move the old SDP instance-specific files aside.

```
cd /p4/1
mv 1 OLD.1.$(date +'%Y-%m-%d')
```

3.2.3. Upgrade Physical Structure

In this step, the physical structure of the upgrade is done for pre-2019.1 SDP.

The structure of the SDP changed in the 2019.1 release, to increase performance and reduce complexit in post-failover operations. The following notes describe how to do an in-place conversion to the new structure.

First, become familar with the Pre-2019.1 and 2019.1+ structures.

SDP Pre-2019.1 Structure:

- `/p4` is a directory on the operating system root voume, `/`.
- `/p4/N` is a symlink to a directory is typically the mount point for a storage volume (`/hxdepots` by default).

- `/p4/N` contains symlinks for `/hxdepots`, `/hxmetadata`, and `hxlogs`, as well as tickets and trust files.

SDP 2019.1+ Structure:

- `/p4` is a directory on the operating system root volume, `/`, (same as Pre-2019.1 Structure).
- `/p4/N` is local directory on the operating system root volume,
- `/p4/N` contains symlinks for `/hxdepots`, `/hxmetadata`, and `hxlogs`, as well as tickets and trust files (same as the Pre-2019.1 structure)
- `/p4/N/bin` is local directory on the operating system root volume. The `bin` directory is the only actual directory in `/p4/N`; other items are files or symlinks to directories.

The `verify_sdp.sh` script (included in the SDP starting with SDP 2019.1) give errors if the 2019.1+ SDP structure is not in place.

Converting the SDP structure in-place to the new style requires downtime on the edge/replica of interest. While the downtime can be brief if only the SDP structure is changed, commonly the P4D is upgraded in the same maintenance window. If the P4D is pre-2019.1, a longer maintenance window will be required, depending on duration of checkpoints.

Following is the procedure to upgrade the structure in-place on a machine.

In the following sample procedure, the default SDP instance name of `1` is used, and default mount point names are used. Adapt this to your environment by applying this procedure to each instance on any given machine. If you have multiple instances, apply this procedure for each instance, one at a time.

3.2.4. Replace Instance Symlink with Directory

Move the instance symlink aside, and replace it with a regular directory. Then copy the `.p4*` files (e.g. `.p4tickets` and `.p4trust`) into the new directory. Sample commands:

```
cd /p4
mv 1 1.old_symlink
mkdir 1
cd 1
cp -p /p4/1.old_symlink/.p4t* .
```

3.2.5. Convert Fixed to Variable Metadata Symlinks

If you have Fixed Metadata Symlinks, first convert them to to Variable Metadata Symlinks. If you already have Varialbe Metadata Symlinks, proceed to Section 3.2.4, “Replace Instance Symlink with Directory”

In this step, move the underlying directories that will be pointed to by the `root` and `offline_db` sylmink names, and move them to their `db1` and `db2` names.


```
mv /hxmetadata/p4/1/root /hxmetadata/p4/1/db1
mv /hxmetadata/p4/1/offline_db /hxmetadata/p4/1/db2
```

3.2.6. Replace Instance Symlink with Directory

Next, recreate the same symlinks you see reported by the `ls` command:

```
ls -l /p4/1.old_symlink/*
cd /p4/1

ln -s /hxmetadata/p4/1/db1 root
ln -s /hxmetadata/p4/1/db2 offline_db
```

Do not just copy the sample commands above. Pay close attention to the `ls` output, and make sure the `root` points to whatever it was pointing to before, either a directory ending in `db1` or `db2` (unless you just converted from Fixed Metadata Symlinks in STEP 4). Also confirm that `offline_db` and `root` aren't both pointing to the same directory; one should be pointing to `db1` and the other to `db2`.

Then, create additional symlinks akin to whatever else is in `/p4/1.old_symlink`

That should look something like this:

```
cd /p4/1
ln -s /hxdepots/p4/1/depots
ln -s /hxdepots/p4/1/checkpoints
ln -s /hxdepots/p4/1/checkpoints.YourEdgeServerID
ln -s /hxlogs/p4/1/logs
ln -s /hxlogs/p4/1/tmp

ls -l
```

Next, create the `bin` directory, as a local directory and copy files to it:

```
mkdir bin
cd bin
cp /p4/1.old_symlink/bin/p4d_1_init .
cp /p4/1.old_symlink/bin/p4broker_1_init .
ln -s /p4/common/bin/p4broker_1_bin p4broker_1
ln -s /p4/common/bin/p4_bin p4_1
```

Last, take a look at `/p4/1.old_symlink/bin/p4d_1` - that `p4d_1` will be either a tiny script or a symlink (depending on whether your p4d is case sensitive or not). If you server is case sensitive, it will be a symlink. If your server is case-insensitive, it will be a tiny script.

If your server is case sensitive, create the symlink like this:

```
ln -s /p4/common/bin/p4d_1_bin p4d_1
```

OR, if your server is case-sensitive, that p4d_1 will be a tiny script, so just copy it:

```
cp /p4/1.old_symlink/bin/p4d_1 .
```

Then, start your server again, and run the `verify_sdp.sh` script and confirm that it's happy now.

3.2.7. Put new SDP common files in place.

```
rsync /p4/sdp/Server/Unix/p4/common/ /hxdepots/p4/common
```

3.2.8. Put new SDP instance bin files in place.

```
cd /p4/1/bin

sed s:REPL_SDP_INSTANCE:1:g /p4/common/etc/init.d/p4d_instance_init.template >
p4d_1_init
chmod +x p4d_1_init
sed s:REPL_SDP_INSTANCE:1:g /p4/common/etc/init.d/p4broker_instance_init.template >
p4broker_1_init
chmod +x p4broker_1_init
```

3.2.9. Upgrade systemd service files

The format of systemd unit files changed with the SDP 2020.1 release.

The service must be down when these until files are added, or existing ones.

The SDP r20.1 release includes templates for System unit files in `/p4/common/etc/systemd/system`. These should be deployed on each machine that uses SDP, and for each Helix service (e.g. `p4d`, `p4broker`, `p4p`) within each SDP instance.

For example, the following installs or replaces system Unit files for `p4d` and `p4broker` for SDP instance 1. These must be executed as `root`

First, stop the services if they are running.

```
systemctl stop p4d_1 p4broker_1
/p4/1/bin/p4d_1_init stop
/p4/1/bin/p4broker_1_init stop
```

Next, add/replace the *.service files:

```
cd /etc/systemd/system
```

```
sed -e s:__INSTANCE__:1:g -e s:__OSUSER__:perforce:g
/p4/common/etc/systemd/system/p4d_N.service.t > p4d_1.service
```

```
sed -e s:__INSTANCE__:1:g -e s:__OSUSER__:perforce:g
/p4/common/etc/systemd/system/p4broker_N.service.t > p4broker_1.service
```

```
systemctl daemon-reload
```

Enable and start the services.

```
systemctl enable p4d_1 p4broker_1
systemctl start p4d_1 p4broker_1
```

Confirm that they are happy:

```
systemctl status p4d_1 p4broker_1
```

3.3. Post Operation Steps

Cleanup steps can occur after the upgrade. In some cases cleanup is done immediately following the upgrade; in other cases it may be deferred by days or weeks.

3.3.1. Cleanup

Temporary directories with DELETE_ME created durnig the upgrade procedure can now be deleted.

Appendix A: Custom HMS Managed Installations

If the Helix Management System (HMS) is used to manage this installation, you should have custom site-specific documentation for upgrading the SDP that supercedes this documentation. If the file `/p4/common/bin/hms` exists at your site, you have an HMS-managed site. Conact Perforce Consulting for more information.

Note that HMS solutions are inherently custom and not officially supported, but can be fully automated for global Helix Core topologies.