HelixTeamHub

Helix TeamHub Administrator Guide

2020.1 Enterprise

April 2020

PERFORCE

www.perforce.com

Contents

How to Use this Guide

The Helix TeamHub Administrator Guide is intended for system administrators and engineers responsible for maintaining a Helix TeamHubOn-Premises installation. It includes information on planning deployment, installing, bootstrapping, and maintaining Helix TeamHub software.

This section provides information on typographical conventions, feedback options, and additional documentation.

Syntax conventions

Helix documentation uses the following syntax conventions to describe command line syntax.

Notation	Meaning
`literal`	Must be used in the command exactly as shown.
italics	A parameter for which you must supply specific information. For example, for a *serverid* parameter, supply the ID of the server.
[`-f`]	The enclosed elements are optional. Omit the brackets when you compose the command.
`...`	Previous argument can be repeated. ■ `p4 [g-opts] streamlog [ -l -L -t -m max ]` ***`stream1`*** `...` means `1` or more stream arguments separated by a space ■ See also the use on `...` in Command alias syntax in the *Helix Core P4 Command Reference* **Tip** `...` has a different meaning for directories. See Wildcards in the *Helix Core P4 Command Reference*.
element1 | *element2*	Either *element1* or *element2* is required.

Feedback

How can we improve this manual? Email us at manual@perforce.com.

Other documentation

See https://www.perforce.com/support/self-service-resources/documentation.

What's new in this guide for this release

Following is a summary of new information with links to the most prominent topics. For a complete list, see the *Helix TeamHub Release Notes*.

- Added support for installing Helix TeamHub on the Ubuntu 18.04, RHEL 8, and CentOS 8 Operating systems. For a list of supported Operating Systems, see "Operating System" on page 25.

Getting Started

This section provides a quick introduction to Helix TeamHub On-Premises installation in a single server environment (see "Combo Setup" on page 34) with one of the supported platforms.

Download the Helix TeamHub Combo package specific to the host operating system from Perforce's package repositories. After uploading the package to the server, install the dependencies and the package itself as root.

Alternatively, use the Perforce's package repositories directly to install the package instead of using the **`rpm -ivh`** or **`dpkg -i`** commands below.

RHEL and CentOS

```
yum install postfix bzip2 sudo cronie mailx libgomp
rpm -ivh hth-X.X.X-stable.el6.x86_64.rpm
```

Ubuntu

```
apt-get update
apt-get install postfix bzip2 sudo cron mailutils libgomp1
dpkg -i hth_X.X.X_amd64.deb
```

Next, follow the steps provided by the installation package to configure the instance.

For a more detailed and production ready setup, please visit "Types of Deployment" on page 17.

1 | Limitations with Helix authentication

TeamHub is integrated with Helix server. If your TeamHub instance is configured to use Helix authentication, you can manage repositories and kick off reviews using the TeamHub user interface.

Helix authentication implies that part of the configuration and management happens in Helix server. As a result, some elements in the TeamHub UI have been removed or disabled and others have been added, as detailed in the following table.

Entity	Change	Helix server Documentation (where applicable)
Collaborators	Removed the **Collaborators** view and ability to view or add collaborators in **Team** view	N/A
Bots	Disabled ability to change a bot's short name	
Repositories	Added ability to: ■ Create Git repositories stored in Helix server. For details, see the *Helix TeamHub User Guide*. ■ View Helix server connection details (port and path) by selecting the **Clone > Helix P4** option Removed ability to: ■ Fork native Git repositories ■ Configure the garbage collection feature for Git repositories managed by Helix server Git repositories managed by Helix server do not require garbage collection because they do not store data on the disk where the Helix TeamHub instance resides.	
Company settings	■ **Authentication** tab: Disabled ability to configure SAML authentication **Note** With Helix authentication, Helix TeamHub supports only one company per instance. Company creation is disabled.	

Overview

This section provides the following information:

System Overview

Helix TeamHub On-Premises is the private installation of Helix TeamHub running on the infrastructure of the organization, behind the firewall. The overview section of this guide provides information about the operating environment of Helix TeamHub, the architecture, release process, and other important background information.

Operating Environment

Helix TeamHub is composed of a number of popular open source components (see "System Architecture" on the next page). Helix TeamHub is provided as a comprehensive solution for hosting and managing enterprise software. Helix TeamHub is provided as a native package, including .rpm and .deb, and can be installed on one of the supported **64-bit** architecture Linux platforms.

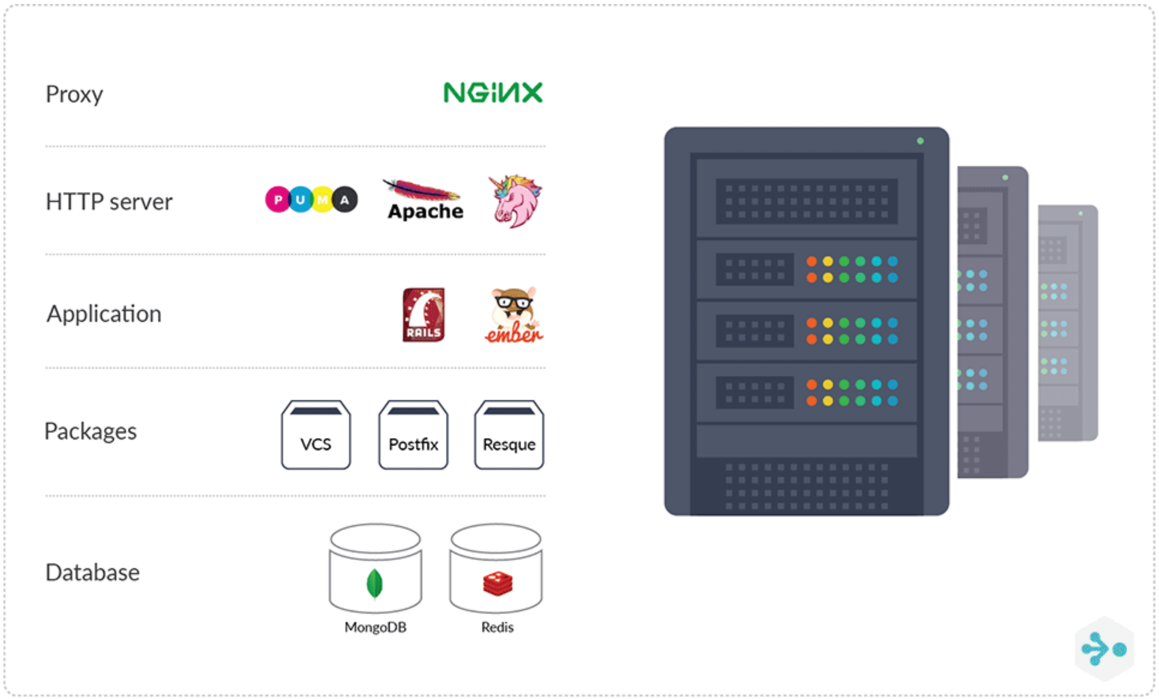
Administrative Account

Most of the Helix TeamHub services are running as local *hth* user account, except services that require binding to privileged ports. Therefore full **sudo** rights are required.

Additionally, Helix TeamHub provides a special Admin account, which is used to administer Helix TeamHub from the UI. Administrator privileges allows a user to maintain the Helix TeamHub installation.

System Architecture

Helix TeamHub software is composed of popular and powerful open source technologies, which are used in todays most advanced cloud solutions. The architecture is layered. Each component has its own role in serving user requests.



The entire stack is provided as a stand-alone native operating system package, also known as a "Combo" on page 18 setup. Alternatively, Helix TeamHub stack can be split into a number of servers, known as a "Cluster" on page 19 setup, and further expanded to a "High Availability" on page 21 (HA) cluster. Regardless of the deployment type, the logical layers are as follows:

Proxy

Nginx proxy sits at front and handles all user requests. Based on the operation type (static page, version control system, or Web application) requests are forwarded to the appropriate component.

HTTP server

Helix TeamHub has three HTTP servers: Apache, Unicorn and Puma. Apache is responsible for handling version control operations, Unicorn serves Helix TeamHub Web applications, and Puma handles websocket connections.

Application

Helix TeamHub consists of two major components. On the front-end is Ember powered application responsible for providing UI to Helix TeamHub APIs. Back-end is powered by Ruby on Rails, which exposes RESTful APIs and Helix TeamHub Admin UI. It's important to note, Helix TeamHub adheres to an API-first strategy. Any functionality seen in the Helix TeamHub UI can be achieved programatically with Helix TeamHub APIs.

Packages

Helix TeamHub relies on various open source packages that are bundled with Helix TeamHub On-Premises. Packages are precompiled for Helix TeamHub supported operating systems, and are completely isolated from the rest of the system. The below is the list of some of the packages bundled with Helix TeamHub:

- Git, Subversion, Mercurial - Helix TeamHub supported Version Control Systems.
- Resque - Handles background jobs.
- GraphicsMagick - Processes images uploaded to Helix TeamHub.

Database

At the heart of the stack are the two NoSQL databases. MongoDB is used for storing application data. Redis keeps intermediate data, background jobs and events for example.

File system hierarchy

Helix TeamHub closely follows Linux File System Hierarchy Standard for its data and bundled packages. Here is the layout of the file system hierarchy Helix TeamHub uses:

- `/opt/hth` The root folder for application and its dependencies.
- `/opt/hth/bin` The admin provided utilities and tools (see Helix TeamHub control).
- `/var/opt/hth/shared` The data directory with all user data, such as repositories and uploads.
- `/var/opt/hth/db` The database directory.
- `/var/opt/hth/backups` The directory where backups are stored.
- `/var/log/hth` Application and system logs.

For a detailed list of system wide configuration files that Helix TeamHub touches, see System overrides section.

Releases

Helix TeamHub does not have a predefined release cycle. Instead, new versions are continuously delivered. See release notes for details about latest releases and for any release specific upgrade instructions.

Obtaining Helix TeamHub

Use the Perforce's package repositories to download or install TeamHub and future releases.

Package names

All TeamHub packages follow the below naming convention:

```
hth-[ROLE]-<MAJOR-YEAR>.<MAJOR-NUMBER>.<MINOR>.<PATCH>...
```

Where:

- `ROLE` in optional role this package is designated for, for example *db* or *web*, or simply nothing in case of Combo type of deployment.
- `MAJOR-YEAR` is the year the version was released.
- `MAJOR-NUMBER` is the major version release number.
- `MINOR` is the minor version release number.
- `PATCH` is the patch version release number.

Getting help

Helix TeamHub Support:	Go to our Support page
Contact the Team:	support@perforce.com

In many cases, the Support Team will ask for a *report* of the Helix TeamHub environment, which is obtained by running the following command:

```
sudo hth-ctl report
```

This will generate a compressed archive at `/tmp/hth_report.tar.gz` with TeamHub logs and system information, which can be attached to the support email.

To request a feature, please post a new idea at our Support Portal.

Preparation

This section provides the following information:

Types of Deployment

Helix TeamHub can be deployed in a number of ways, including: Combo, Cluster, or High Availability Cluster. This section explains the major differences between the types of deployment, depending on the installation environment.

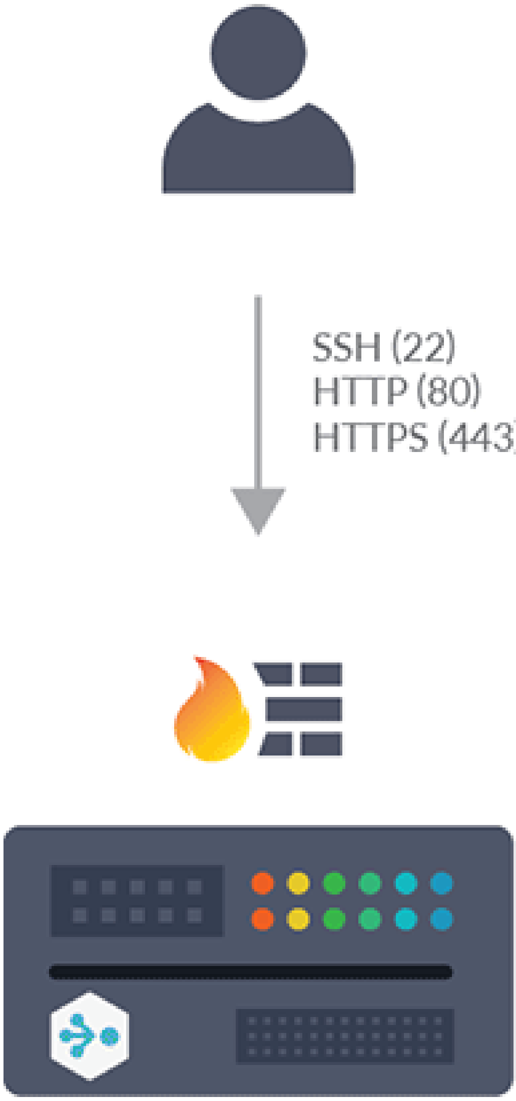
Quick Comparison

	Combo	Cluster	High Availability
Time to setup	Minutes	Hours	Days
Complexity	Low	Moderate	High
Servers	1	2	5+
Availability	Low	Moderate	High

Combo

Combo deployment is the easiest to deploy. It's quick to setup, and doesn't require provisioning multiple servers, or worrying about inter-server firewalls and advanced configuration.

The entire Helix TeamHub technology stack is bundled in a single operating system package ready to be deployed virtually anywhere- whether it's a bare metal server, cloud computing instance, or a local virtual machine.

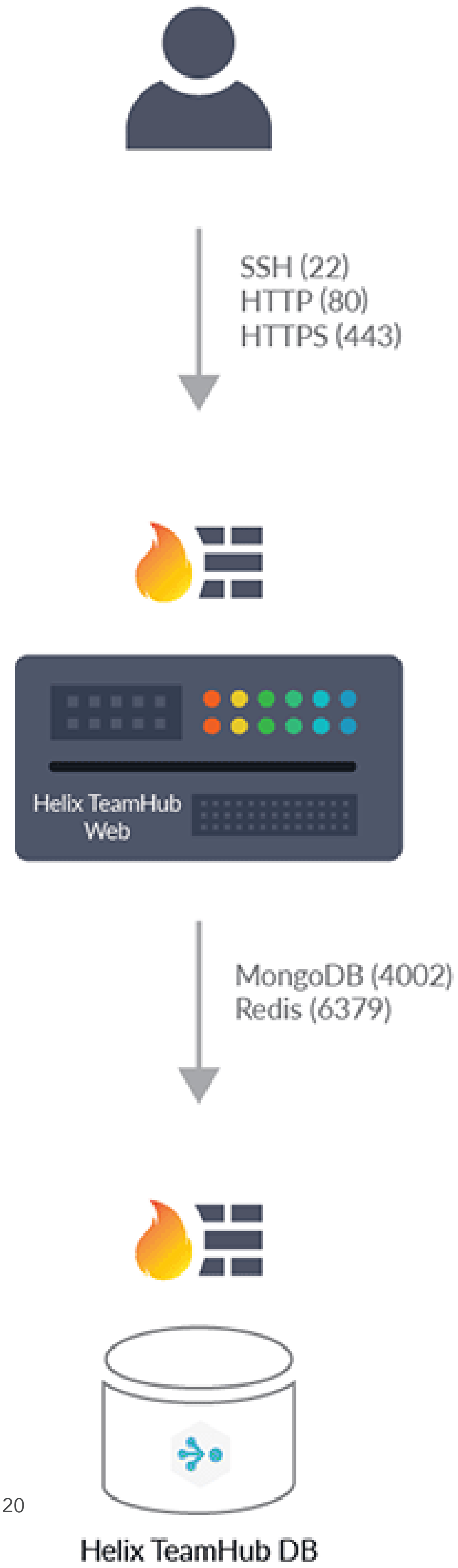


The biggest drawbacks of this setup are the required downtime during hardware failures, and the migration to a clustered setup if necessary.

The Bottom Line: Choose Combo deployment for the fastest setup, without a large up-front investment in hardware or IT resources.

Cluster

Cluster deployment splits Helix TeamHub into two roles: Helix TeamHub Web (application) and Helix TeamHub DB (database). Therefore, Cluster deployment needs at least two servers to operate. Since Helix TeamHub Web connects to Helix TeamHub DB, this type of deployment usually requires tuning the network firewall to allow inter-server communication (see firewall requirements).


SSH (22)
HTTP (80)
HTTPS (443)
Helix TeamHub Web
MongoDB (4002)
Redis (6379)
Helix TeamHub DB

Helix TeamHub is then delivered in two standalone packages: `hth-web` and `hth-db`. As described in "Releases" on page 16, the major versions need to be kept in sync.

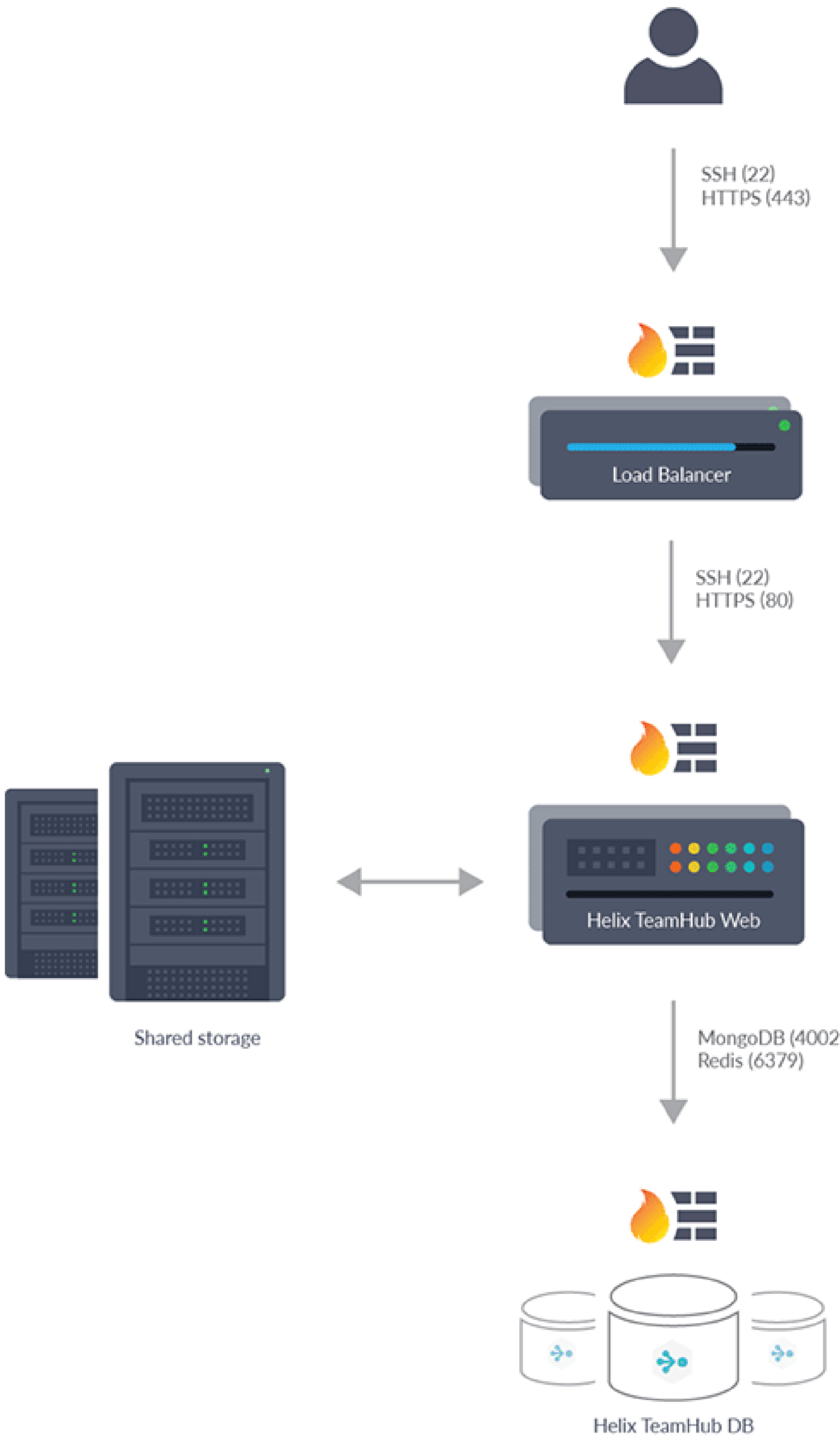
Cluster setup shares similar drawbacks to Combo. However, it has an advantage of being future-proof for highly-available and redundant cluster (see below). Moreover, the physical separation of the server roles increases security and eases maintenance.

The bottom line: Choose the Cluster deployment for a highly available and redundant Helix TeamHub environment that is scalable, without a large up-front investment.

High Availability

High Available deployment is the most comprehensive and advanced. It requires a bigger up-front investment in hardware and IT resources. However, the benefits of high available deployment include: on demand scalability, zero-downtime maintenance, and maximum availability of Helix TeamHub.

This type of deployment resembles Cluster setup in terms of the packages and server roles, however it requires additional components and redundancy. For instance, SSL is offloaded by the hardware or software Load Balancer (see "How to Setup HAProxy" on page 132), which distributes load to a number of Helix TeamHub Web servers (we recommend at least 3), where each in turn uses a number of Helix TeamHub Database servers. Additionally, the shared storage is utilized to have the same data across the cluster nodes.


SSH (22)
HTTPS (443)
Load Balancer
SSH (22)
HTTPS (80)
Helix TeamHub Web
Shared storage
MongoDB (4002)
Redis (6379)
Helix TeamHub DB

The bottom line: Choose the High Available type of deployment for maximum availability of Helix TeamHub service and full control over its capacity.

Hardware Requirements

The hardware requirements vary depending on multiple factors including: the deployment type, seat count, repository size, and usage patterns. This page provides guidelines that help estimate the required hardware specifications.

Helix TeamHub is written to be efficient and lightweight. However, due to the nature of the product, most of the activities trigger calls to the underlying version control system. Keeping this in mind makes choosing the right hardware easier because Helix TeamHubusage is estimated along with the underlying components.

Minimal Requirements

Helix TeamHub technology stack consists of a number of components. In order for them to stay coherent, the following minimum hardware specifications are required for any server having Helix TeamHub installation:

Resource	Requirement
CPU	2.6GHz
Memory	4GB
Root disk	40GB
Data Storage	High-performance SAN or locally attached storage

Determining hardware capacity requirements

Helix TeamHub is battle-tested to serve thousands of users. The variation in usage patterns makes it hard to give accurate numbers, but the following matrix can be used as reference based on past experience.

Seats	Memory	CPU Cores	Root Disk Space
0-100	8GB	2	40GB
100-500	16GB	4	40GB
500-1000	32GB	8	40GB
1000-2000	64GB+	16	40GB

Since the system deals with a lot of IO operations to the repositories on disk, it is recommended to have an ultra-fast (SSD/SAN) data storage with speedy access.

Scaling Horizontally

Built-in support for horizontal scalability makes Helix TeamHub ready to handle a large number of requests with increased performance. It's recommended to use a hardware or software (see "How to Setup HAProxy" on page 132) load balancer for distributing the load across Helix TeamHub cluster nodes. Combining round-robin algorithms with decent hardware specs delivers the best performance.

Helix TeamHub DB

Helix TeamHub uses MongoDB extensively. It is recommended giving MongoDB a decent amount of RAM to have the working set reside in memory for fast access.

HAproxy

If using HAProxy for load balancing, Helix TeamHub recommends 1-2 CPU cores and 2GB of RAM. Having a reliable and fast network between the load balancer and the Helix TeamHub Web servers is extremely important.

Understanding Helix TeamHub Resource Usage

Git cloning is the most demanding task for CPU/RAM. For a large Git repo (1.5 GB, 500K commits), git-pack-objects utilizes a single core CPU from 45% to around 90% and around 10% of RAM. It then uses lower resources during git-receive-pack operation to about 20% CPU and 40% RAM. For the same repository, the initial Git push/import triggers git-index-pack, which utilizes CPU from 45% to around 90% and about 10% of RAM.

Further assistance

For questions or concerns about performance issues and/or capacity management, don't hesitate to contact the Helix TeamHub Support Team.

Firewall Requirements

Inbound Connections

To allow users access to Helix TeamHub via Web browsers and version control clients, make sure Helix TeamHub server is reachable by the following connections:

- `TCP 22` - Version control access over SSH.
- `TCP 80` - Web application and version control access over HTTP.
- `TCP 443` - Web application and version control access over HTTPS (only if SSL will be used).

If Helix TeamHub LDAP interface is enabled, the following connections are required:

- `TCP 389` - Unencrypted LDAP connection.
- `TCP 636` - Encrypted LDAPS connection (only if SSL will be used).

Inter-server connections

For Cluster or High Availability type of deployment, the following connections are required:

- `TCP 4002` from Helix TeamHub Web to Helix TeamHub DB server - Access to MongoDB database.
- `TCP 6379` from Helix TeamHub Web to Helix TeamHub DB server - Access to Redis database.

For convenience, it's recommended the user check arrows and corresponding ports for deployment type, and then reference here for a more detailed explanation of the port usage.

Outbound connections

Helix TeamHub may use a number of external services for mailing, authentication, etc. If external services are required, make sure the following connections are open for Helix TeamHub servers to access:

- `TCP [usually 25]` to use SMTP gateway for mailing.
- `TCP [usually 389]` to use corporate LDAP server for authentication.

Operating System

As mentioned in System overview, Helix TeamHub is provided as a native operating system package, such as .rpm and .deb for one of our supported **64-bit** architecture Linux platforms:

- Red Hat Enterprise Linux 6, 7, 8
- CentOS 6, 7, 8
- Ubuntu 16.04, 18.04

Since some of the above distributions have several minor versions, only the **2 latest minor** releases are supported. For example, as of writing this document, the latest two releases of RedHat Enterprise Linux 6 family are *6.5* and *6.6*. To benefit from the OS level security updates, performance optimizations, and compatibility with Helix TeamHub, closely follow the release cycle of the operating system in use and upgrade in a timely manner.

In addition to the operating system, the following preparations are required:

Ports

Helix TeamHub application binds to a number of ports that must be free before proceeding with installations. In some cases, the Linux distribution may already have an installed package that uses the ports Helix TeamHub needs. Therefore, make sure all the ports listed in Inbound and Inter-server connections are available. The following command can be executed to check if anything is listening on port *80*:

```
netstat -tulpn | grep :80
```

Local Firewall

In addition to the company wide firewall, the local firewall may also be installed by default, for example iptables. Make sure it's either disabled or configured to accept the ports listed in Inbound and Inter-server connections.

UID and GID

When Helix TeamHub is installed, the new **`hth`** user account and system group are created with predefined UID and GID of **`21212`**, so make sure they are not reserved.

Locale

Make sure **`en_US.UTF-8`** locale is installed and no errors are reported when running **`export LC_ALL=en_US.UTF-8`**.

Linux Security Modules

LSMs (Linux Security Modules) such as SELinux may also prevent Helix TeamHub from running. To disable LSMs:

RHEL and CentOS

Edit **`/etc/selinux/config`** and ensure that **`SELINUX`** is either in **`disabled`** or **`permissive`** mode. To avoid restarting the server for changes to come into effect, run the command below to immediately disable SELinux:

```
setenforce 0
```

Ubuntu

Does not need any changes.

OpenSSH and repository SSH access

Helix TeamHub supports accessing repositories over SSH protocol. OpenSSH version 6.9 or later is required with support for **`AuthorizedKeysCommand`** with arguments. Repository SSH access can be enabled after installing Helix TeamHub by either using the system or bundled OpenSSH. It's recommended to use system OpenSSH, but the bundled OpenSSH can be used if upgrading OpenSSH is not otherwise possible.

Use system OpenSSH

Append following configuration to the end of the sshd configuration file (**`/etc/ssh/sshd_config`**) and reload sshd:

```
Match User hth
    AuthorizedKeysCommand /usr/bin/hth-ssh-auth %t %k
    AuthorizedKeysCommandUser hth
```

Use bundled OpenSSH

When using the bundled OpenSSH, it is important that automatic updates are configured to skip OpenSSH package. OpenSSH updates can be disabled as follows:

RHEL and CentOS

Open **`/etc/yum.conf`** and add the following line under **`[main]`** section:

```
exclude=openssh*
```

Ubuntu

```
sudo apt-mark hold openssh-server
```

When using systemd, change the service configuration file (usually in **`/etc/systemd/system/sshd.service`**) to use simple type under **`[Service]`** section:

```
Type=simple
```

Reload systemd configuration after changing service configuration:

```
systemctl daemon-reload
```

In order to use the bundled OpenSSH, merge the following configuration to **`/var/opt/hth/shared/hth.json`**, run **`sudo hth-ctl reconfigure`** and reload sshd: **Note: this will symlink the existing sshd to the bundled sshd.**

```
{
    "opensshp": {
        "enable": true
    }
}
```

SSH Optimization

For Helix TeamHub setups that are relatively large, we found that setting the following parameters for SSHD helps with security and efficiency of the system overall:

```
MaxStartups 100
ClientAliveInterval 60
ClientAliveCountMax 3
```

You can add those parameters manually to **`/etc/ssh/sshd_config`** on either the Helix TeamHub combo node or the hth-web node.

Memory Optimization

Helix TeamHub requires Redis and MongoDB to be installed. To avoid latencies and memory usage issues in these services, we recommend disabling Transparent Huge Pages (THP) in the kernel for Combo installation and Cluster DB nodes.

Network Optimization

Depends on the load on your installation, it is recommended to increase the limit of the backlog for connections (**`somaxconn`**) to higher value. We suggest setting it to **`512`** at minimum.

Dependency requirements

You must install the Helix TeamHub dependencies before you install TeamHub.

RHEL and CentOS

> **Tip**
> If you are creating a DB node, the **`libgomp`** dependency is not required in the following command. However, installing it does no harm.

Run the following command as **`root`** to install the Helix TeamHub dependencies:

```
yum install postfix bzip2 sudo cronie mailx libgomp
```

Configure **`postfix`** as **`Internet Site`** during setup.

Ubuntu

> **Tip**
> If you are creating a DB node, the `libgomp1` dependency is not required in the following command. However, installing it does no harm.

Run the following command as `root` to install the Helix TeamHub dependencies:

```
apt-get update
apt-get install postfix bzip2 sudo cron mailutils libgomp1
```

Configure `postfix` as `Internet Site` during setup.

Configure the Perforce repository

You must add the Perforce packaging key to your keyring and configure the Perforce repository before installing Helix TeamHub from the Perforce repository.

RHEL and CentOS

1. Add the Perforce packaging key to your RPM keyring:

   ```
   sudo rpm --import https://package.perforce.com/perforce.pubkey
   ```

2. Add the Perforce repository to your YUM configuration by creating a file called `/etc/yum.repos.d/perforce.repo` with the following content, where *`<version>`* is either 6 for RHEL/CentOS 6, 7 for RHEL/CentOS 7, or 8 for RHEL/CentOS 8.

   ```
   [perforce]
   name=Perforce
   baseurl=http://package.perforce.com/yum/rhel/<version>/x86_64
   enabled=1
   gpgcheck=1
   ```

   For example, to install TeamHub on RHEL/CentOS 7, add:

   ```
   [perforce]
   name=Perforce
   baseurl=http://package.perforce.com/yum/rhel/7/x86_64
   enabled=1
   gpgcheck=1
   ```

Ubuntu

1. Add the Perforce packaging key to your APT keyring:

   ```
   wget -qO - https://package.perforce.com/perforce.pubkey | sudo apt-key add -
   ```

2. Add the Perforce repository to your APT configuration by creating a file called **/etc/apt/sources.list.d/perforce.list** with the following line:

   ```
   deb http://package.perforce.com/apt/ubuntu <distro> release
   ```

   where **<*distro*>** is either xenial or bionic.

   For example, to install TeamHub on Ubuntu xenial, add:

   ```
   deb http://package.perforce.com/apt/ubuntu xenial release
   ```

Helix authentication prerequisites

Configuring Helix TeamHub with Helix authentication requires specific setup in Helix server. In particular, you need:

- An installation of the following products, each on its own, dedicated machine:
  - Helix TeamHub 2018.1 or later

    We recommend at least 3-4GB of memory and proper provisioning.
  - Helix server 2017.2 or later

    We recommend a server security level of +1. For more information, see Server security levels in the *Helix Core Server Administrator Guide*.
  - Helix4Git

  > **Warning**
  > If Helix TeamHub resides on the same machine as Helix4Git, port conflicts occur. For best results, we recommend setting up all 3 components on separate machines.

- A Helix server license with the correct number of seats for your users, your **gconn-user** (Git Connector), and your Bots. For instructions on how to calculate the number of licensed seats you need, see "TeamHub License" on page 50.
- **super** level access to Helix server with an optional unlimited timeout ticket if ticket-based authentication will be used (recommended).

  For more information, see *Helix Core Server Administrator Guide*, sections Setting protections with p4 protect and Ticket-based authentication.
- Host name, protocol, port, and user information for the Git Connector

For more information, see the *Helix4Git Administrator Guide*.

- `Admin` permission for the `gconn-user` on any manually created graph depots

  For information on granting permissions, see the *Helix4Git Administrator Guide*, section Grant permissions.

- Users with appropriate access to Helix server. All user and group administration occurs in Helix server.

  > **Note**
  > Make sure that the names of Helix server users that need access to Helix TeamHub do not exceed 100 characters. TeamHub only supports user names up to 100 characters.

- An entry in the protections table for all users that need access to Helix TeamHub

  For more information, see *Helix Core Server Administrator Guide*, Setting protections with p4 protect.

  The admin user needs `superuser` access to the protections table to view permissions.

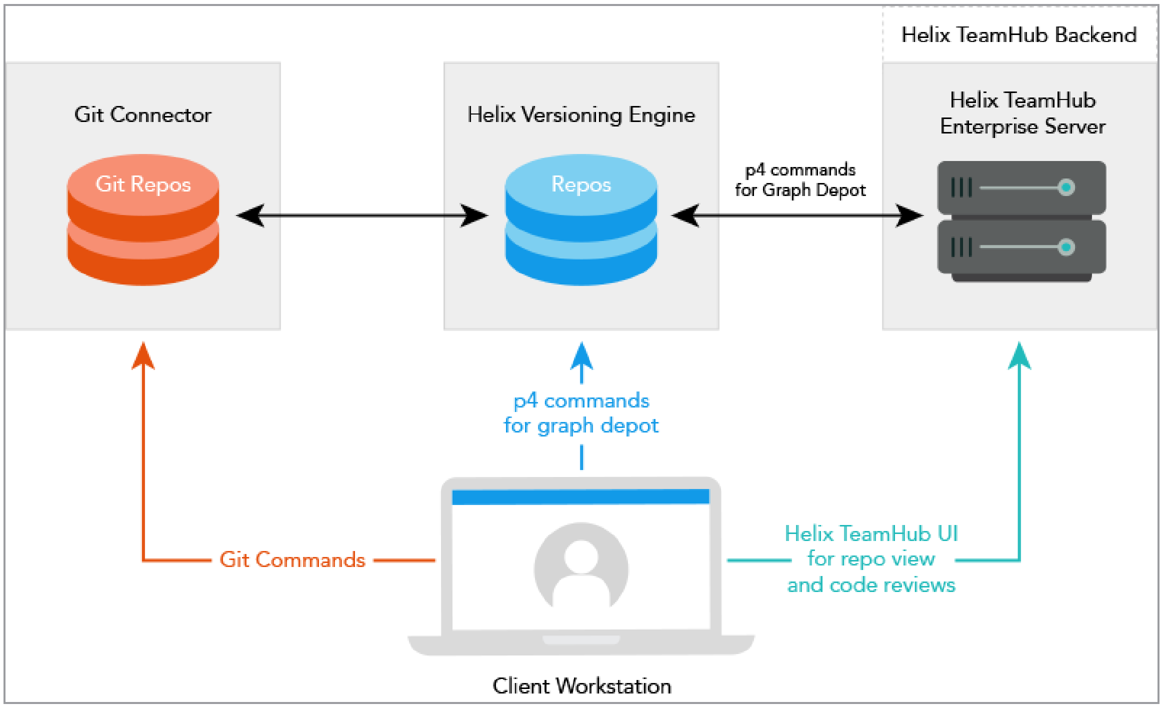
- An access level of either `admin` or `create-repo` for users that need the ability to add repos in specific depots

  For more information, see *Helix Core P4 Command Reference*, `p4 grant-permission` command.

If you intend to install the TeamHub trigger, make sure the machine hosting Helix server has Perl 5.08+ and Perl Core on CentOS installed.

> **Warning**
> Once you have configured Helix authentication and the TeamHub instance is in use, it is not possible to revert back to a different authentication method.

To get a better understanding of the underlying architecture, the following figure provides a high-level overview.



Installation and configuration flow

The following table outlines the flow of setting up Helix authentication in Helix TeamHub and specifies the product it pertains to.

Step	Description	Product	Instructions
1.	Install Helix TeamHub.	Helix TeamHub	"Installation" on page 34 section in this manual

Step	Description	Product	Instructions
2.	If you start from a new Helix server, add users and groups in Helix server. For existing Helix server installations, verify that the protections table includes the required entries for the users that need access to Helix TeamHub.	Helix server	"Add users and groups in Helix server" on page 65 in this manual *Helix Core Server Administrator Guide*: ■ Managing users ■ Granting access to groups of users ■ Setting protections with p4 protect ■ Ticket-based authentication *Helix Core P4 Command Reference*: ■ `p4 grant-permission` command ■ `p4 group` command ■ `p4 protect` command
3.	Configure TeamHub to use Helix server authentication.	Helix TeamHub	"Set up Helix server authentication" on page 59 section in this manual
4.	Install triggers in Helix server to enable the activity stream in Helix TeamHub.	Helix server	"Set up Helix trigger scripts for TeamHub" on page 62 in this manual
5.	Create a new project and repository.	Helix TeamHub	*Helix TeamHub User Guide*: ■ Getting started ■ Version control with Git
6.	Configure group access to the newly created project.	Helix TeamHub	"Add users and groups in Helix server" on page 65 in this manual
7.	Clone the new repository to your local workspace.	Helix TeamHub	*Helix TeamHub User Guide*: ■ Cloning a repository

Installation

This section provides the following information:

Combo Setup

It doesn't take long to get Helix TeamHub up and running with Combo deployment.

You can either download Helix TeamHub packages and install them manually, or install Helix TeamHub using the Perforce package repositories (recommended).

Step 1: Before you begin

Confirm that you have met all of the prerequisites listed below before installing and configuring Helix TeamHub:

- The hardware you are installing Helix TeamHub on must meet the "Hardware Requirements" on page 23.
- Your firewall must be configured to meet the Helix TeamHub "Firewall Requirements" on page 24.
- The machines you are installing Helix TeamHub on must have a supported "Operating System" on page 25.
- The machines you are installing Helix TeamHub on must have the TeamHub dependencies installed, see "Dependency requirements" on page 28.
- **Installing from a repository only:** the machines you are installing Helix TeamHub on must be configured for the Perforce repository, see "Configure the Perforce repository" on page 29.
- **Helix authentication only:** your system must meet the "Helix authentication prerequisites" on page 30.

Step 2: Installing Packages

Install using repositories

Install the package itself as root (recommended). If you have downloaded the TeamHub package, see "Manually install from a downloaded Helix TeamHub package" below.

RHEL and CentOS

1. Configure the Perforce repository if you have not already done so, see "Configure the Perforce repository" on page 29.
2. Run the following command to install the TeamHub package:

```
sudo yum install hth
```

Ubuntu

1. Configure the Perforce repository if you have not already done so, see "Configure the Perforce repository" on page 29.
2. Run the following commands to install the TeamHub package:

```
sudo apt-get update
sudo apt-get install hth
```

Manually install from a downloaded Helix TeamHub package

If you have downloaded the TeamHub package, perform the following steps to install it.

RHEL and CentOS

1. Run the following command to install the package where ***`<version>`*** is the TeamHubversion and ***`<OS-Version>`*** is the operating system version:

   ```
   rpm -ivh hth-<version>-stable.el<OS-Version>.x86_64.rpm
   ```

   For example, to install TeamHub 2020.1, run:

   - On RHEL/CentOS 7:

     ```
     rpm -ivh hth-2020.1-stable.el7.x86_64.rpm
     ```

   - On RHEL/CentOS 8:

     ```
     rpm -ivh hth-2020.1-stable.el8.x86_64.rpm
     ```

Ubuntu

1. Run the following command to install the package:

   ```
   dpkg -i hth_<version>-stable~<distro>_amd64.deb
   ```

   where:

   - ***`<version>`*** is the TeamHub release you are installing, such as 2019.1.
   - ***`<distro>`*** is either xenial or bionic.

   For example, to install TeamHub 2019.2 on xenial, run:

   - On Ubuntu xenial:

     ```
     dpkg -i hth_2019.2-stable~xenial_amd64.deb
     ```

Step 3: Configuring Helix TeamHub

Every installation, upgrade, or configuration change in Helix TeamHub will require the Helix TeamHub Control utility to reconfigure Helix TeamHub for changes to take effect. To enable all required services after installation, run the following command:

```
sudo hth-ctl reconfigure
```

Step 4: Bootstrapping

At this stage Helix TeamHub is installed and running, and can be accessed using **`hth`** as Company ID and **`admin`** as user ID and password (change them in company and user settings):

```
http://IP_ADDRESS_OF_YOUR_SERVER/login
```

However, the Helix TeamHub instance needs to have a valid license before accessing the dashboard. The license can be managed in Helix TeamHub Admin. See License for more information.

Helix TeamHub Admin can be used to further configure the instance (at minimum, configuring hostname and mail settings are recommended), see Bootstrapping Helix TeamHub for more information.

Enable MongoDB Authentication (Optional)

By default MongoDB listens only on localhost on Combo installations and does not use authentication. For added security, authentication can also be configured on Combo installations.

1. Run the following command to create Helix TeamHub MongoDB admin and user credentials:

```
sudo su - hth
create_mongodb_users.sh
```

2. Update Helix TeamHub Configuration file at **`/var/opt/hth/shared/hth.json`** with Helix TeamHub MongoDB user credentials:

```
"backend": {
  ...
  "db_username": "Enter username of Helix TeamHub MongoDB user",
  "db_password": "Enter password of Helix TeamHub MongoDB user"
  ...
}
...
"mongodb": {
  ...
  "username": "Enter username of Helix TeamHub MongoDB user",
  "password": "Enter password of Helix TeamHub MongoDB user"
  ...
}
```

3. Finally, apply the changes by reconfiguring Helix TeamHub:

```
sudo hth-ctl reconfigure
```

Cluster Setup

Use the following instructions to install and configure Helix TeamHub for Cluster deployment.

Download the Helix TeamHub Cluster (**`hth-db`** and **`hth-web`**) packages specific to the host operating system from Perforce's package repositories.

Alternatively, use the Perforce's package repositories directly to install the packages instead of using the `rpm -ivh` or `dpkg -i` commands below.

Step 1: Before you begin

Confirm that you have met all of the prerequisites listed below before installing and configuring Helix TeamHub:

- The hardware you are installing Helix TeamHub on must meet the "Hardware Requirements" on page 23.
- Your firewall must be configured to meet the Helix TeamHub "Firewall Requirements" on page 24.
- The machines you are installing Helix TeamHub on must have a supported "Operating System" on page 25.
- The machines you are installing Helix TeamHub on must have the TeamHub dependencies installed, see "Dependency requirements" on page 28.
- **Installing from a repository only:** the machines you are installing Helix TeamHub on must be configured for the Perforce repository, see "Configure the Perforce repository" on page 29.
- **Helix authentication only:** your system must meet the "Helix authentication prerequisites" on page 30.

Step 2: Installing Helix TeamHub DB

Install using repositories

Install the package itself as root (recommended). If you have downloaded the TeamHub package, see "Manually install from a downloaded TeamHub package" on the next page.

RHEL and CentOS

1. Configure the Perforce repository if you have not already done so, see "Configure the Perforce repository" on page 29.
2. Run the following command to install the TeamHub package:

   ```
   sudo yum install hth-db
   ```

Ubuntu

1. Configure the Perforce repository if you have not already done so, see "Configure the Perforce repository" on page 29.
2. Run the following commands to install the TeamHub package:

```
sudo apt-get update
sudo apt-get install hth-db
```

Manually install from a downloaded TeamHub package

Upload the **`hth-db`** package to the server designated for Database role, install the package itself as root:

RHEL and CentOS

```
rpm -ivh hth-db-X.X.X-stable.el7.x86_64.rpm
```

Ubuntu

```
dpkg -i hth-db_X.X.X_amd64.deb
```

Step 3: Configuring Helix TeamHub DB

Every installation, upgrade, or configuration adjustment in Helix TeamHub will require the Helix TeamHub Control utility to reconfigure Helix TeamHub so the changes can take effect. To enable all required services after installation, run the following command:

```
sudo hth-ctl reconfigure
```

Step 4: Enabling Cluster Mode

In order for Helix TeamHub Web to be able to communicate to Helix TeamHub DB, configure MongoDB and Redis to accept authentication and remote connections.

Run the following command to create Helix TeamHub MongoDB admin and user credentials:

```
sudo su - hth
create_mongodb_users.sh
```

> **Important**
> Remember the MongoDB credentials, they're also required to set up Helix TeamHub Web servers.

Next, reconfigure Helix TeamHub DB for Redis authentication. Open the Helix TeamHub Configuration file at **`/var/opt/hth/shared/hth.json`** add the following lines to the JSON file and update credentials:

```
{
  ... (snipped)
  "app" : {
    "is_cluster": true
  },
  ... (snipped)
  "redis": {
    "enable": true,
    "password": "Choose Redis password"
  },
  "mongodb": {
    "enable": true,
    "username": "Enter username of Helix TeamHub MongoDB user",
    "password": "Enter password of Helix TeamHub MongoDB user"
  },
  ... (snipped)
}
```

Finally, apply the changes by reconfiguring Helix TeamHub DB:

```
sudo hth-ctl reconfigure
```

Step 5: Installing Helix TeamHub Web

> **Note**
> The Helix TeamHub Web machine must also meet the prerequisites, see "Step 1: Before you begin" on page 38.

Install using repositories

Install the package itself as root (recommended). If you have downloaded the TeamHub package, see "Manually install from a downloaded TeamHub package" on the next page.

RHEL and CentOS

1. Configure the Perforce repository if you have not already done so, see "Configure the Perforce repository" on page 29.
2. Run the following command to install the TeamHub package:

   ```
   sudo yum install hth-web
   ```

Ubuntu

1. Configure the Perforce repository if you have not already done so, see "Configure the Perforce repository" on page 29.
2. Run the following commands to install the TeamHub package:

```
sudo apt-get update
sudo apt-get install hth-web
```

Manually install from a downloaded TeamHub package

Upload the `hth-web` package to the server designated for Web application role, install the package itself as root.

RHEL and CentOS

```
rpm -ivh hth-web-X.X.X-stable.el7.x86_64.rpm
```

Ubuntu

```
dpkg -i hth-web_X.X.X_amd64.deb
```

Step 6: Connecting Helix TeamHub Web to Helix TeamHub DB

Note: Don't reconfigure Helix TeamHub Web yet. First, 'Turn' the server into a cluster node, and add Helix TeamHub DB credentials for it to connect to Helix TeamHub databases. Open the Helix TeamHub Configuration file at `/var/opt/hth/shared/hth.json` and add the following lines to the JSON file and update credentials and hosts:

```
{
  ... (snipped)
  "app" : {
    "is_cluster": true
  },
  ... (snipped)
  "backend": {
        "db_host": "Enter IP address or hostname of Helix TeamHub DB
server",
        "db_port": "4002",
        "db_username": "Enter username of Helix TeamHub MongoDB user",
        "db_password": "Enter password of Helix TeamHub MongoDB user",
        "redis_host": "Enter IP address or hostname of Helix TeamHub DB
server",
        "redis_password": "Enter Redis password chosen on Helix TeamHub DB
server",
        "enable": true
  },
  ... (snipped)
}
```

Finally, apply the changes by reconfiguring Helix TeamHub Web:

```
sudo hth-ctl reconfigure
```

Step 7: Bootstrapping

At this stage Helix TeamHub is installed and running in Cluster mode, and can be accessed using `hth` as Company ID and `admin` as user ID and password (change them in company and user settings):

```
http://IP_ADDRESS_OF_YOUR_WEB_SERVER/login
```

However, the Helix TeamHub instance needs to have a valid license before accessing the dashboard. The license can be managed in Helix TeamHub Admin, see License for more information.

Helix TeamHub Admin can be used to further configure the instance (at minimum, configuring hostname and mail settings are recommended), see Bootstrap for more information.

High Availability Setup

High Availability can be applied to a Helix TeamHub cluster installation. The benefits of high availability include: on demand scalability, zero-downtime maintenance, and maximum availability of Helix TeamHub.

Step 1: Before you begin

Before applying High Availability to your Helix TeamHub cluster setup, make sure that you have completed the following steps:

- Complete your Helix TeamHub "Cluster Setup" on page 37
- Make sure that Helix TeamHub has been bootstrapped, see "Bootstrapping Helix TeamHub" on page 45

Step 2: Scaling up with Load Balancer

As was mentioned in HA Deployment, an SSL load balancer is required, which will decrypt SSL connections and also balance requests across the Helix TeamHub Web servers.

The Helix TeamHub package does not include load balancer, therefore it needs to be installed separately. The following guide is recommended to setup the load balancer first if none exists.

Step 3: Mounting Shared Storage

With a load balancer, the user requests will be randomly distributed across the cluster nodes, so the data will become immediately out of sync. To fix this issue, attach the same shared storage to **each Helix TeamHub Web** server. If existing storage with a clustered file system doesn't exist, contact the Support team for further help.

After shared storage is available, stop Helix TeamHub, then mount the storage to **`/var/opt/hth/shared`**. Next, bring Helix TeamHub back online:

```
sudo hth-ctl stop
sudo mv /var/opt/hth/shared /var/opt/hth/shared.bak
# Mount storage to /var/opt/hth/shared and sync back the data
sudo rsync -av /var/opt/hth/shared.bak/ /var/opt/hth/shared/
rm -rf /var/opt/hth/shared.bak
sudo hth-ctl start
```

Step 4: Synchronizing SSH Host Keys

Since the SSH host keys will differ between the cluster nodes, they need to be synchronized. Helix TeamHub configuration process can use the **`ssh`** directory on the shared storage and copy the SSH host keys to the usual **`/etc/ssh`**. This will enable every new Helix TeamHub Web server added to the cluster to have the same SSH host keys. So on the first Helix TeamHub Web server, create the directory and copy SSH host keys:

```
mkdir -p /var/opt/hth/shared/ssh
cp /etc/ssh/ssh_host_* /var/opt/hth/shared/ssh/
chown root.root /var/opt/hth/shared/ssh/*
chmod 600 /var/opt/hth/shared/ssh/*
```

Step 5: Adding More Helix TeamHub Web Servers

After you have performed the steps above, additional Helix TeamHub Web servers can be added to the cluster. Because the Helix TeamHub Configuration file is stored on a shared partition **`/var/opt/hth/shared`**, simply install the Helix TeamHub Web package and reconfigure it, and Helix TeamHub will automatically pick up the needed configurations.

> **Note**
> The Helix TeamHub Web machines must also meet the prerequisites, see "Step 1: Before you begin" on page 38 in "Cluster Setup" on page 37.

Install using repositories

Install the package itself as root (recommended). If you have downloaded the TeamHub package, see "Manually install from a downloaded TeamHub package" below.

RHEL and CentOS

1. Configure the Perforce repository if you have not already done so, see "Configure the Perforce repository" on page 29.
2. Run the following command to install the TeamHub package:

```
sudo yum install hth-web
```

Ubuntu

1. Configure the Perforce repository if you have not already done so, see "Configure the Perforce repository" on page 29.
2. Run the following commands to install the TeamHub package:

```
sudo apt-get update
sudo apt-get install hth-web
```

Manually install from a downloaded TeamHub package

Upload the **`hth-web`** package to the server designated for Web application role, install the package as root.

RHEL and CentOS

```
rpm -ivh hth-web-X.X.X-stable.el7.x86_64.rpm
```

Ubuntu

```
dpkg -i hth-web_X.X.X_amd64.deb
```

Step 6: Changing Hostname

At this stage everything should be up and running, and requests should be distributed across all Helix TeamHub Web servers. However, Helix TeamHub is still bootstrapped with the hostname of the first Helix TeamHub Web server installed. To fix this, go to Helix TeamHub Admin and change the hostname to the load balancer.

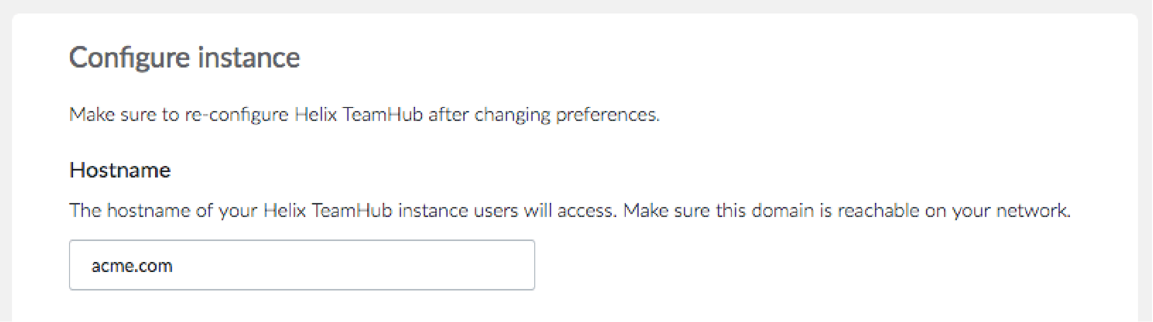
Bootstrapping Helix TeamHub

The Helix TeamHub Admin is a simple UI application for configuring and managing Helix TeamHub configuration. Use the following steps to configure hostname and mail settings to your instance. To start, go to the **`/admin`** URL of the Helix TeamHub installation.

Step 1: Configure Hostname

In order to use Helix TeamHub, a valid host name is required. The Helix TeamHub hostname must be a fully qualified domain name (FQDN) and resolvable by product users. Helix TeamHub will use this configured hostname in repository URLs, email links, and API requests. Even though Helix TeamHub can be configured to use IP address or another hostname assigned to the server, a delegated alias (CNAME) or name (A) to simplify future server relocations is recommended.

The "Preferences" page provides a number of other configurations, such as: Authentication method, enabling Helix TeamHub LDAP interface, and enforcing SSL. However, the first step is to bootstrap Helix TeamHub to make sure everything works as expected. Enter the hostname and click **Save preferences**.



Configure instance

Make sure to re-configure Helix TeamHub after changing preferences.

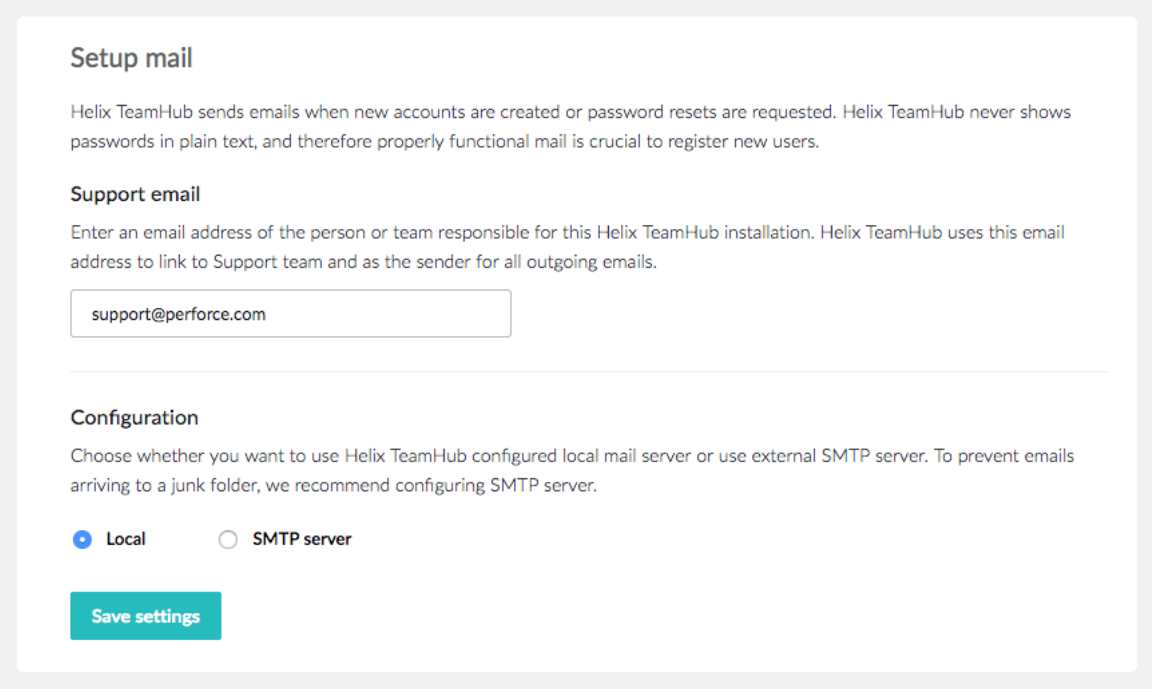
Hostname

The hostname of your Helix TeamHub instance users will access. Make sure this domain is reachable on your network.

acme.com

Step 2: Setup Mail

Enter the Support email, which is used in all outgoing Helix TeamHub emails, as well as the links to the Support team throughout application. To configure external SMTP server for handling Helix TeamHub emails, see Mailing Configuration section.



Setup mail

Helix TeamHub sends emails when new accounts are created or password resets are requested. Helix TeamHub never shows passwords in plain text, and therefore properly functional mail is crucial to register new users.

Support email

Enter an email address of the person or team responsible for this Helix TeamHub installation. Helix TeamHub uses this email address to link to Support team and as the sender for all outgoing emails.

support@perforce.com

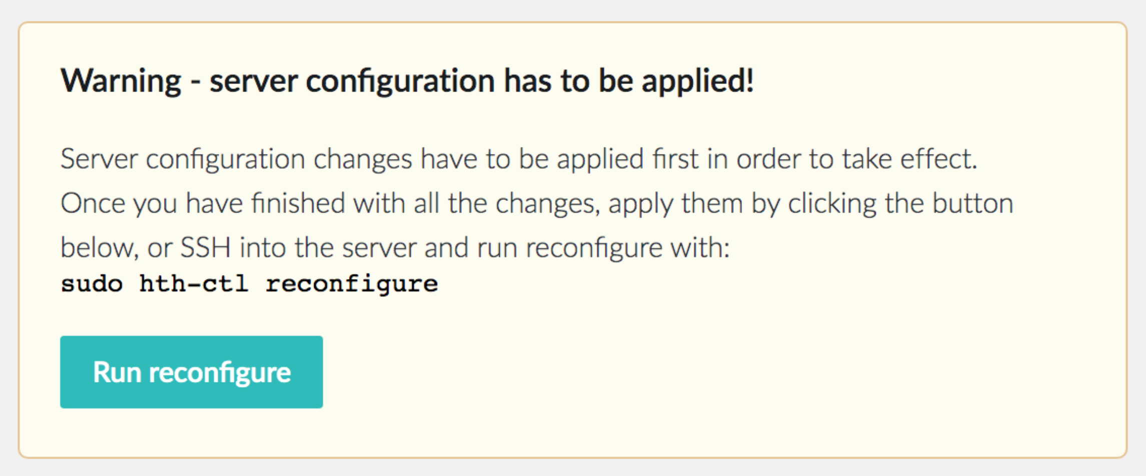
Configuration

Choose whether you want to use Helix TeamHub configured local mail server or use external SMTP server. To prevent emails arriving to a junk folder, we recommend configuring SMTP server.

● Local ○ SMTP server

Step 3: Apply Configuration Changes

While completing the previous steps, the following warning appears:



Warning - server configuration has to be applied!

Server configuration changes have to be applied first in order to take effect. Once you have finished with all the changes, apply them by clicking the button below, or SSH into the server and run reconfigure with:

```
sudo hth-ctl reconfigure
```

This warning occurs every time the configuration is changed in some way that affects underlying Helix TeamHub services. Click **Run reconfigure** to address this issue.

> **Note**
> For Cluster Deployment, apply the changes on every Helix TeamHub Web server.

Congratulations, the Helix TeamHub instance is successfully bootstrapped and ready to use. It's a good time to test everything and further extend Helix TeamHub for production use:

- Choosing Authentication
- Enforcing SSL
- Enabling backups

Upgrades

Helix TeamHub is shipped as a native operating system package, like .rpm or .deb, which greatly simplifies upgrades. Upgrading Helix TeamHub is usually a zero-downtime operation and can be performed on a live system. Before upgrading, visit the release notes for more information on any release specific upgrade instructions.

Download the Helix TeamHub packages specific to the host operating system from Perforce's package repositories. After uploading the packages to the server, follow the commands below.

Alternatively, use the Perforce's package repositories directly to upgrade the packages instead of using the **`rpm -Uvh`** or **`dpkg -i`** commands below.

Combo

RHEL and CentOS upgrades

```
rpm -Uvh hth-X.X.X-stable.el6.x86_64.rpm
sudo hth-ctl reconfigure
```

Ubuntu

```
dpkg -i hth_X.X.X_amd64.deb
sudo hth-ctl reconfigure
```

Cluster or HA

When updating Cluster or HA deployment type, the package install order (Web or DB) does not matter, **as long as** they are the same major version. Refer to the release notes for more information on the package install order.

Helix TeamHub DB

Upload the **`hth-db`** package to the server designated for Database role and update the package:

RHEL and CentOS

```
rpm -Uvh hth-db-X.X.X-stable.el6.x86_64.rpm
sudo hth-ctl reconfigure
```

Ubuntu

```
dpkg -i hth-db_X.X.X_amd64.deb
sudo hth-ctl reconfigure
```

Helix TeamHub Web

Upload the **`hth-web`** package to the server designated for Web application role and install the package:

RHEL and CentOS

```
rpm -Uvh hth-web-X.X.X-stable.el6.x86_64.rpm
sudo hth-ctl reconfigure
```

Ubuntu

```
dpkg -i hth-web_X.X.X_amd64.deb
sudo hth-ctl reconfigure
```

Administration

This section provides the following information:

TeamHub License

Every Helix TeamHub instance requires a license, which can be obtained by contacting Helix TeamHub Sales. The type of licenses and number of seats will depend on your installation, see "License plan types and number of seats required" on the facing page.

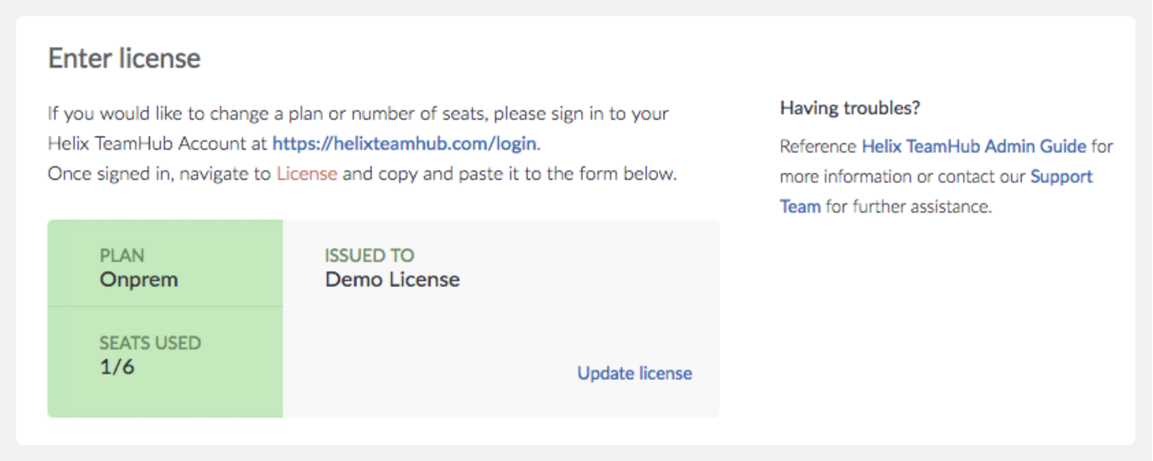
Tip

- If required, you can reduce your active storage by deleting repositories, see Maintenance settings in the *Helix TeamHub User Guide*.
- When you delete content from your repositories:
  - **Helix Git** repositories are versioned and track change history, repository data is not deleted from the Helix server.
  - **Git, Mercurial, and Subversion** repositories are versioned and track change history. Files committed to the remote repository stay in the history even if you remove them. Those files continue to consume storage.
  - **Ivy, Maven, and WebDAV** repositories are unversioned and do not track history. Removing files frees up storage.
  - **Docker** repositories are unversioned and do not track history. Removing tags frees up storage.

Adding or updating the Helix TeamHub license file

To change the license, navigate to **`/admin`** URL of the Helix TeamHub installation, log in with the administrator account, and click **Dashboard**.

Click on **Update license**, paste the license contents, and hit **Apply**.



Enter license

If you would like to change a plan or number of seats, please sign in to your Helix TeamHub Account at https://helixteamhub.com/login.
Once signed in, navigate to License and copy and paste it to the form below.

PLAN
Onprem

SEATS USED
1/6

ISSUED TO
Demo License

Having troubles?

Reference Helix TeamHub Admin Guide for more information or contact our Support Team for further assistance.

License plan types and number of seats required

This section will help you to decide what licenses you need and how many seats you need on each license. To buy or renew licenses, contact Helix TeamHub Sales.

Helix TeamHub Cloud license plan

Various seat and storage options are available for Helix TeamHub Cloud, see Pricing for Helix TeamHub on Perforce.com.

> **Note**
> You cannot use Helix authentication for Git repositories with Helix TeamHub Cloud plans.

Helix TeamHub on-premise license plan

You must purchase enough seats on the license plan to cover all of your active Helix TeamHub Accounts. A seat is any individual account that logs in to Helix TeamHub, this includes user and collaborator accounts. For instructions on how to add or update your Helix TeamHub on-premise license plan, see "Adding or updating the Helix TeamHub license file" above.

> **Note**
> - Bots do not count against your Helix TeamHub license plan seats but each bot requires a

> background license. Request background licenses for your bots using the Background user request form on www.perforce.com.
>
> - If you are using Helix authentication, all Git repositories are stored in the Helix server and you will also need Helix server licenses. To calculate the number of Helix server licenses you need, see "Helix server license plan" below below.

Helix server license plan

If you are using Helix TeamHub Enterprise in conjunction with "Helix authentication" on page 59 you also need a Helix server license. The number of seats required on Helix server license is:

Total number of seats on the **Helix server license** = Total number of **Active Helix TeamHub Accounts (users and collaborators)** + Total number of **Helix TeamHub bots**

For instructions on how to add or update your Helix server license plan, see Adding or updating the license file in the *Helix Core Server Administrator Guide*.

> **Note**
> The Git Connector (`gconn`) user does not count against your Helix server license plan seats but it does require a background license. Request a background license for the `Gconn` user using the Background user request form on www.perforce.com.

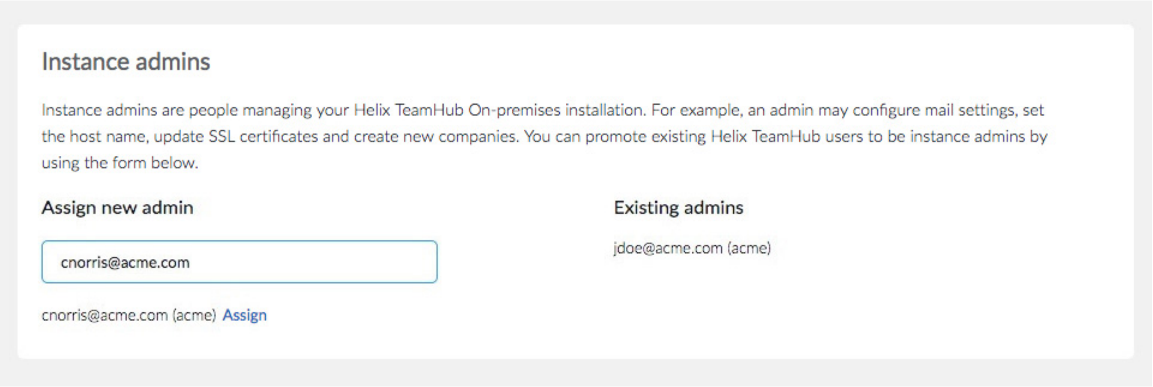
TeamHub Administrators

Helix TeamHub instance administrators are individuals in the organization responsible for maintaining and administering the TeamHub instance. Instance administrators are different from company administrators. Internally, they are regular TeamHub users, but with elevated privileges and access to the Admin UI, also referred to as the admin portal. Only instance admins can log in to the admin portal. Company administrators are users with admin permission to access and modify company settings, but they cannot access the admin portal.

If Helix authentication is configured, the login process first tries to use Helix authentication. If Helix authentication fails, it falls back to the local, built-in password. This mechanism prevents lockouts on misconfigured instances.

By default, assuming Helix TeamHub has been bootstrapped, at least one administrator is required. To add more administrators, navigate to the `/admin` URL of the Helix TeamHub installation, login with the administrator account, and click **Admins**.

Assign new administrators by entering the email of each user. New administrators must already be an existing account in one of the Helix TeamHub companies:



Instance admins

Instance admins are people managing your Helix TeamHub On-premises installation. For example, an admin may configure mail settings, set the host name, update SSL certificates and create new companies. You can promote existing Helix TeamHub users to be instance admins by using the form below.

Assign new admin

cnorris@acme.com

cnorris@acme.com (acme)

Existing admins

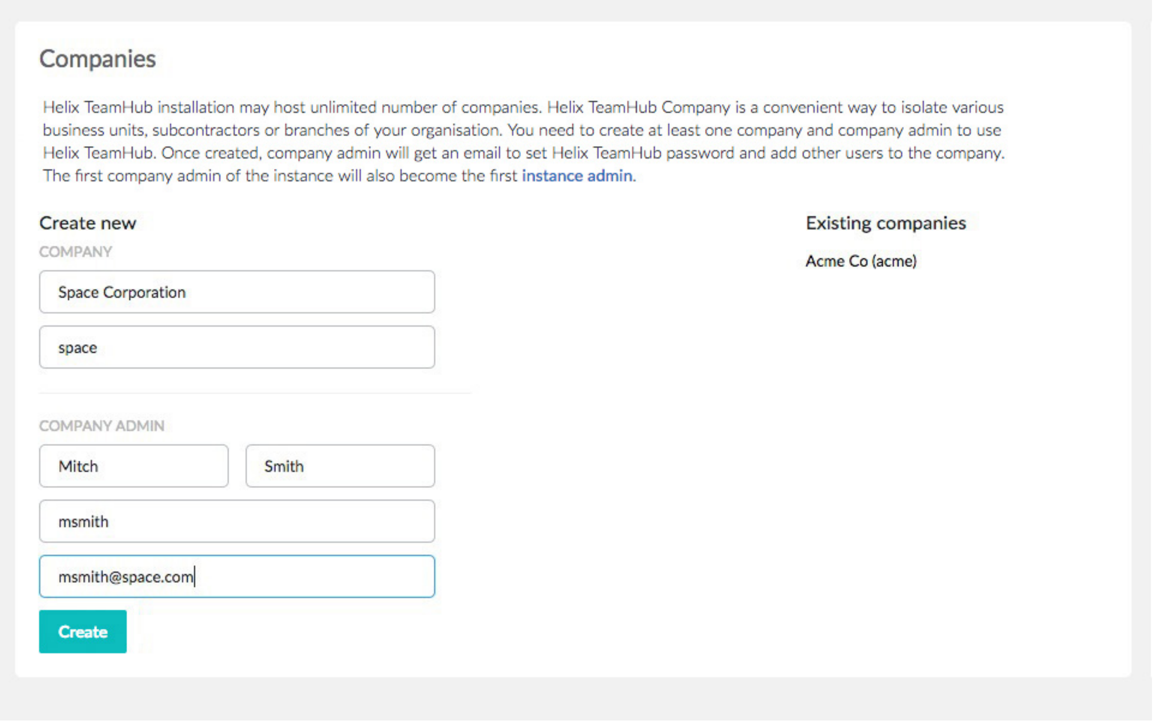
jdoe@acme.com (acme)

TeamHub Companies

Helix TeamHub Companies are logical containers (a.k.a. namespace) that isolate various business units, subcontractors, or branches within an organization.

Assuming Helix TeamHub has been bootstrapped, by default there should be at least one company. To add more companies, navigate to **`/admin`** of the Helix TeamHub installation, login with the administrator account, and click **Companies**.

To create a new company, add the company title, a short name (unique identifier), and enter the details of the first Company Admin.



Companies

Helix TeamHub installation may host unlimited number of companies. Helix TeamHub Company is a convenient way to isolate various business units, subcontractors or branches of your organisation. You need to create at least one company and company admin to use Helix TeamHub. Once created, company admin will get an email to set Helix TeamHub password and add other users to the company. The first company admin of the instance will also become the first instance admin.

Create new

COMPANY

Space Corporation

space

COMPANY ADMIN

Mitch

Smith

msmith

msmith@space.com

Existing companies

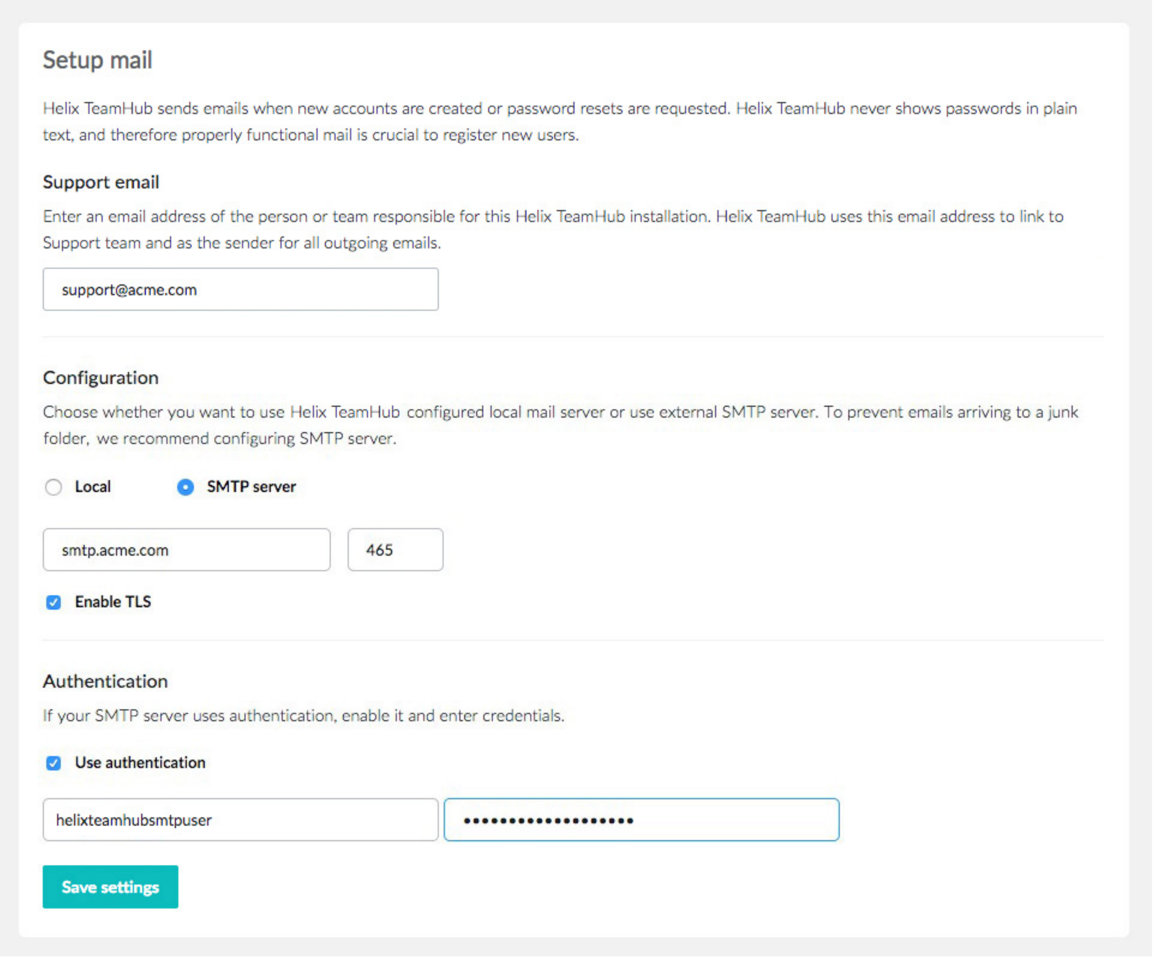
Acme Co (acme)

Mailing Configuration

Helix TeamHub uses email to create new user credentials and send application notifications. Therefore, email is required. Helix TeamHub comes with prepackaged Postfix, which is an open-source Mail Transfer Agent (MTA).

Although the default local mailing may be sufficient, a corporate mail gateway is required in most cases. It's possible to configure a remote SMTP server responsible for delivering Helix TeamHub emails. Authentication and TLS are also options.

To configure external SMTP server, log in to the `/admin` URL of the Helix TeamHub installation, navigate to **Mail settings**, and choose **SMTP server**. The following configuration options are available:



Setup mail

Helix TeamHub sends emails when new accounts are created or password resets are requested. Helix TeamHub never shows passwords in plain text, and therefore properly functional mail is crucial to register new users.

Support email

Enter an email address of the person or team responsible for this Helix TeamHub installation. Helix TeamHub uses this email address to link to Support team and as the sender for all outgoing emails.

support@acme.com

Configuration

Choose whether you want to use Helix TeamHub configured local mail server or use external SMTP server. To prevent emails arriving to a junk folder, we recommend configuring SMTP server.

○ Local ● SMTP server

smtp.acme.com 465

☑ Enable TLS

Authentication

If your SMTP server uses authentication, enable it and enter credentials.

☑ Use authentication

helixteamhubsmtpuser ••••••••••••••••••••

Email templates

The layout and content of automated Helix TeamHub emails are based on email templates. To view and edit the email templates, go to the `/admin` URL of the Helix TeamHub installation and navigate to the **Email templates** page. The following email templates can be customized:

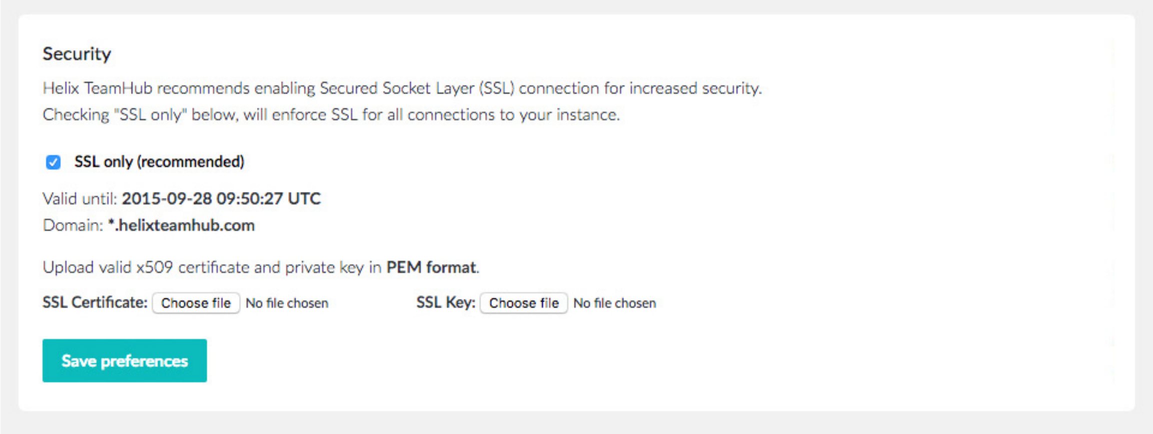
- **Email signature:** the signature attached to all Helix TeamHub emails
- **Welcome email:** the email is sent to new users when only external authentication is in use
- **Registration email:** the email is sent to new users when "Built-in" or "Built-in + LDAP" authentication is in use
- **Recovery email:** the email is sent to users when "Built-in" or "Built-in + LDAP" authentication is in use
- **Recovery email for synced accounts:** the email is sent to users when only external authentication is in use
- **Password expiration email:** the email is sent to users when their password is about to expire

SSL Configuration

When Helix TeamHub is installed, it does not enforce an SSL connection by default to access the application and repositories. While this may be acceptable for services running behind an organization's firewall, enforcing SSL is highly recommended if the Helix TeamHub instance is exposed to a public network.

To enable SSL:

1. Login to the `/admin` URL of the Helix TeamHub installation and navigate to **Preferences**.
2. Under **Security**, select the **SSL only (recommended)** check box.



Security

Helix TeamHub recommends enabling Secured Socket Layer (SSL) connection for increased security.
Checking "SSL only" below, will enforce SSL for all connections to your instance.

☑ SSL only (recommended)

Valid until: **2015-09-28 09:50:27 UTC**
Domain: ***.helixteamhub.com**

Upload valid x509 certificate and private key in **PEM format**.

SSL Certificate: Choose file No file chosen   SSL Key: Choose file No file chosen

3. Upload a valid x509 certificate and private key (RSA) in PEM format, with base64-encoded content between header and footer lines.

> **Note**
> Instructions for generating the certificate and private key depend on the provider. For security reasons, we recommend that you only use a self-signed certificate for testing.

To generate a self-signed certificate and key, you can use OpenSSL:

```
openssl req -newkey rsa:2048 -new -x509 -days 730 -nodes -out hth.crt
-keyout hth.key
```

4. Click **Save preferences**.

   TeamHub displays the certificate expiration date and the assigned domain.

Troubleshooting tips

Include all certificates to the PEM file

A single PEM file can contain a number of certificates and a key, for example:

- Public certificate
- Intermidiate Certificate
- Root certificate
- Private key

You should include all the certificates to the PEM file, but not the private key. Otherwise, Git clients may receive the following error messages when doing operations against repositories:

```
https://helixteamhub.cloud/hth/projects/platform/repositories/git/
insufficient-ssl-cert/': SSL certificate problem: unable to get
local issuer certificate
```

or

```
error: SSL certificate problem, verify that the CA cert is OK.
Details: error:14090086:SSL routines:SSL3_GET_SERVER_
CERTIFICATE:certificate verify failed while accessing.
```

Authentication

Helix TeamHub supports two authentication types: SSH key authentication and password-based authentication.

SSH Key Authentication

SSH key authentication can be used when accessing repositories. This authentication type will always use a SSH key pair to authenticate an account. Helix TeamHub accounts may have multiple SSH keys, but a single SSH key is unique within a TeamHub instance. The same key cannot be shared along accounts even if they are from different companies.

> **Note**
> If TeamHub is set up with Helix authentication, adding an SSH key through the TeamHub UI automatically updates the **`pubkey`** table in the Helix server schema.

See also "OpenSSH and repository SSH access" on page 27.

Password-Based Authentication

Password-based authentication can be used when accessing TeamHub data from repositories, APIs, or the user interface. TeamHub can be configured to use one of the three (**`Built-in`**, **`LDAP`**, **`Built-in + LDAP`**, or **`Helix`**) supported password authentication methods. The effects of the first 3 methods for Helix TeamHub accounts are listed below. For the effects of **`Helix`**, see "Helix authentication" on page 59.

User and Collaborator Accounts

Built-In	LDAP	Built-In + LDAP	Use Case
	✓	✓	New accounts can sign up by logging in using LDAP password and email or accountID.
	✓	✓	New accounts can be added to Helix TeamHub from LDAP by email or accountID.
✓		✓	New accounts outside of LDAP can be added to Helix TeamHub by email.
✓		✓	New accounts will receive a registration email to set the initial password.
	✓		New accounts will receive a welcome email.
	✓		Only accounts found from LDAP can be added to Helix TeamHub.
✓		✓	Accounts can login with local password and email or accountID.
	✓	✓	Accounts can login with LDAP password and email or accountID.
✓		✓	Accounts can use password recovery unless password is synchronized.

Collaborator Accounts without LDAP Support

When LDAP authentication is also enabled for collaborator accounts, they will behave the same way as normal users regarding authentication (see listing above). When LDAP authentication is disabled for collaborators, the following listing is applicable instead.

Built In	LDAP	Built-in + LDAP	Use Case
✓	✓	✓	New collaborators can be added to Helix TeamHub by email.
✓	✓	✓	New collaborators will receive a registration email to set the initial password.
✓	✓	✓	Collaborators can login with local password and email or accountID.
✓	✓	✓	Collaborators can use password recovery unless password is synchronized.

Bot Accounts

Bot accounts will always use local password regardless of the authentication method.

Built In	LDAP	Built-in + LDAP	Use Case
✓	✓	✓	Can access repositories using local password and accountID.

Instance Admin Accounts

Users with admin privileges can always use local password to login to Helix TeamHub Admin.

Built In	LDAP	Built-in + LDAP	Use Case
✓	✓	✓	Can login to Helix TeamHub Admin using local password and email or accountID.
	✓	✓	Can login to Helix TeamHub Admin using LDAP password and email or accountID.
✓	✓	✓	Can use password recovery.

Password expiration

You can configure passwords for built-in authentication to expire a certain number of days after the last password change. You turn on this feature by defining `password_expire_days` via configuration flags. Helix TeamHub sends out an email notification and displays a notification in the UI when the password is close to expiration. To configure how far in advance TeamHub notifies users of the password expiration, set the `password_expire_notify` flag.

When you enable the feature for the first time, the last changed timestamp is set for accounts and the expiration period starts. Changing the password resets the period for the account. If you do not change the password before the expiration period ends, you can use the *forgot password* feature to request a link to the account's email to reset the password. Password expiration only affects users and collaborators; passwords do not expire for bots.

Company admins can disable password expiration for an account in the **Account Details** view. This is recommended for service accounts that are used with integrations and whose passwords are managed separately.

Preventing password reuse

You can prevent password reuse for built-in authentication by defining `password_expire_count` via configuration flags. This setting controls how many old passwords are prevented from being used again. This setting only affects users and collaborators.

Helix authentication

This section provides instructions on how to:

- "Set up Helix server authentication" below
- "Set up Helix trigger scripts for TeamHub" on page 62
- "Add users and groups in Helix server" on page 65

For more information, see:

- "Frequently asked questions (FAQ) about Helix authentication" on page 68
- "Troubleshooting Helix authentication" on page 69

Set up Helix server authentication

The following procedure walks you through setting up your Helix TeamHub instance with Helix server authentication.

> **Important**
> If **Helix** is the selected method of authentication and Helix server uses LDAP authentication, there is

no need to use LDAP authentication in TeamHub. Authentication requests from TeamHub are parsed to Helix server, which then connects to the LDAP/AD server to perform the authentication for the user. Note that LDAP authentication on the Helix server side must be established using the *LDAP specification* method (for details, see Authenticating against Active Directory and LDAP servers section in the *Helix Core Server Administrator Guide*).

Note
Some options are dependent on others and only display if required. For example, the option to add fingerprints only displays if TeamHub detects a **P4PORT** value that starts with **`SSL`**.

If you need information on configuring the Git Connector, see the *Helix4Git Administrator Guide*, section Installation and configuration. Before you proceed, make sure you read the "Helix authentication prerequisites" on page 30.

An unlicensed Helix server is limited to 10 repos. For more information, see Helix server Licenses, which include a license for Helix server and a separate license for Helix4Git.

Warning
Once you have configured Helix authentication and the TeamHub instance is in use, it is not possible to revert back to a different authentication method.

To set up Helix server authentication:

1. In a web browser, go to the Helix TeamHub admin portal: ***`<TeamHub-instance-URL>`*`/admin/login`**
2. Enter your user name and password and click **Log in**.

   The default values are **`admin`**/**`admin`**.
3. When prompted, enter your license information and click **Apply**.
4. In the **Preferences** view:
   a. Enter the hostname of your Helix TeamHub instance.
   b. Under **Authentication**, select **Helix**.
   c. Specify Helix server details:

      The Helix server must have at least one graph depot created in it, and the user below must have **`admin`** access to it.

      - **P4PORT value**: The host (name or IP address) and port for the Helix server, in the following format: **`host:port`**
      - **Fingerprint:** If you connect to Helix server using an SSL connection, add a fingerprint of the key received for SSL connections.

If the fingerprint changes (or expires), you can add more fingerprints and delete outdated fingerprints. This is the equivalent of running the `p4 trust` command in the P4 command line. For details, see the *Helix Core P4 Command Reference*.

> **Warning**
> Deleting a fingerprint configured for the port removes the trust established with Helix server. As a result, everything in TeamHub stops working against the respective SSL port.

d. Specify details for the Helix integration user:

- **Username**: The name of an existing Helix server user with super level privileges. This user must have unlimited ticket timeout.

> **Note**
> You cannot change a username (or email address) in TeamHub. Instead, you need to make this change on the Helix server side and wait for the sync.

- **Ticket/Password**: The password/ticket for the existing Helix server`super` user
- **Charset**: The character encoding set for Helix server, such as `utf8` or `none`

  To find the Helix server charset, run: `p4 -ztag info`

  When connecting to a non-unicode server, the charset is `none` by default. If the charset is not shown, select `none` from the list. For more information on Helix server charsets, run: `p4 help charset`

e. Click **Test Helix Core** connection and wait for the message: `Successfully connected to Helix.`

f. Specify Git Connector details:

- **Hostname**: The host name of the server where the Git Connector is installed
- **SSH User**: The OS user of the Git Connector (default: `git`)
- **Helix User:** The Helix server user of the Git Connector (default: `gconn-user`)

For more information, see Configure the Git Connector in the *Helix4Git Administrator Guide*.

g. Select **SSH** or **HTTPS** as the method used to connect to the **Git Connector**.

We strongly suggest enabling both.

h. Click **Save preferences**.

A warning appears because the changes have not been applied to the TeamHub server yet. Perform the following step to finalize the configuration.

5. To apply server configuration changes to the TeamHub server, connect to the server via SSH and run the following command:

```
sudo hth-ctl reconfigure
```

6. After successfully running the reconfigure command, reload the TeamHub admin portal.

> **Warning**
> Failure to do so may result in normal web server interruption messages, such as the **HTTP Error 503. The service is unavailable.** error, because the services come back online.

Set up Helix trigger scripts for TeamHub

Helix server provides triggers to customize the operation of the server, or to invoke additional processing for specific kinds of versioning operations. Helix TeamHub provides a trigger script written in Perl that notifies TeamHub about activity within the Helix server. When configured, any pushes you do to your graph depots are visible as events in the Helix TeamHub activity stream in the **Activity** view.

For TeamHub to display push events in the UI, you need to configure the TeamHub trigger in the Helix server. The trigger supports the following events:

- **`branch create`**
- **`branch delete`**
- **`tag create`**
- **`tag delete`**
- **`push`**

The trigger script is part of the installation package you downloaded earlier (see "Combo Setup" on page 34).

For more information about triggers, see the section Using triggers to customize behavior in the *Helix Core Server Administrator Guide*.

Configuring the trigger requires that you already have an *admin* bot account. For more information on setting up an *admin* bot account, see the *Helix TeamHub User Guide*, section Bots & programmatic repository access.

> **Note**
> Creating bots is not part of the TeamHub admin portal. You need to log out of **`<`*`TeamHub-instance-URL`*`>/admin/login`** and log on to **`<`*`TeamHub-instance-URL`*`>/login`** as *company admin*.

To set up the TeamHub trigger, you have the following options:

- "Filesystem installation" on the facing page
- "Depot installation" on page 64 (recommended)

Important

Make sure that the following required trigger dependencies have been installed on the machine hosting Helix server:

- Perl 5.08+
- Perl Core on CentOS
- On Ubuntu Xenial (16.04) only, packages **`libnet-ssleay-perl`** and **`libio-socket-ssl-perl`**

Filesystem installation

1. Copy the TeamHub trigger file from the TeamHub instance to the server hosting Helix server. The trigger file lives on the TeamHub instance at the following location: **`/opt/hth/external/helix/triggers/hth-trigger.pl`**
2. If your Helix server deployment uses the commit-edge architecture, you must also copy the script to all edge servers. In this case, make sure the script location has the same path on all servers.
3. Copy the configuration file (**`/opt/hth/external/helix/triggers/hth-trigger.conf`**) to the same directory in Helix server as the trigger file. If you copied the trigger script to the commit server and all edge servers in the previous step, also copy the configuration to the same directory on all servers.
4. Follow steps for trigger configuration.
5. To make sure that the script has execute permissions, run:

   ```
   chmod +x ./hth-trigger.pl
   ```

6. In the filesystem location of your trigger script, run the following command to invoke it:

   ```
   ./hth-trigger.pl -o
   ```

   Note

   To see additional usage information, run the trigger script without any arguments.

7. As a Helix server user with **`super`** privileges, edit the Helix server trigger table by running the `p4 triggers` command (P4 Command Reference) and adding the output lines from the previous command (including the initial tab character).

   Update the trigger script and configuration file paths in each line to reflect the actual paths on your Helix server.

   Provided that you have copied the trigger script and configuration file to common paths on all servers, the trigger line will resemble the following:

   ```
   hth.push-ref-complete graph-push-reference-complete //...
   "%quote%/path/to/hth-trigger.pl%quote% -t graph-push-reference-
   ```

```
complete -d %depotName% -n %repo% -N %repoName% -p %pusher% -r
%quote%%reference%%quote% -O %oldValue% -v %newValue%"
```

Depot installation

1. Copy the TeamHub trigger file from the TeamHub instance to the server hosting Helix server. The trigger file lives on the TeamHub instance at the following location: **`/opt/hth/external/helix/triggers/hth-trigger.pl`**
2. (Recommended) Submit the trigger script (**`hth-trigger.pl`**) to Helix server and run it from the following location in the depot: **`//.hth/triggers/hth-trigger.pl`**

   > **Note**
   > Helix TeamHub does not create this depot. You either need to create it yourself or specify another depot that the TeamHub **`admin`** user can write to.
   >
   > To create a **`//.hth`** depot, run the following as a user with admin-level privileges:
   >
   > ```
   > $ p4 depot .hth
   > ```
   >
   > Ensure that the TeamHub admin user can write to **`//.hth`** depot.
   >
   > For more information on creating and working with depots, see Working with depots in *Helix Core Server Administrator Guide*.

3. Copy the configuration file (**`/opt/hth/external/helix/triggers/hth-trigger.conf`**) to the same directory in Helix server as the trigger file.
4. Submit the configuration file to the depot. The recommended depot location is: **`//.hth/triggers/hth-trigger.conf`**
5. Follow steps for trigger configuration.
6. Invoke the trigger script by running the following command:

   ```
   p4 print -q //.hth/triggers/hth-trigger.pl | perl - -o
   ```

   > **Note**
   > To see additional usage information, run the trigger script without any arguments.

7. Provided you have copied the script and the configuration file to common paths on all servers, the trigger line will resemble the following:

   ```
   hth.push-ref-complete graph-push-reference-complete //...
   "%//.hth/triggers/hth-trigger.pl% -c %//.hth/triggers/hth-
   trigger.conf% -t graph-push-reference-complete -d %depotName% -n
   %repo% -N %repoName% -p %pusher% -r %quote%%reference%%quote% -O
   %oldValue% -v %newValue%"
   ```

The trigger line is also available in the **hth-triggers** file in the same directory from which you copied the trigger and configuration scripts. However, the line might not resemble your actual setup. Therefore, we suggest that you use the `-o` flag mentioned in step 6.

> **Important**
> If your output does not use the depot location, you should update this line to reflect the depot location of the trigger file and the trigger configuration file in your installation.

Trigger configuration

The trigger by itself cannot run without connection to Helix TeamHub. Edit the configuration file by specifying the values for the following variables:

- **HTH_HOST** = Hostname of your Helix TeamHub instance accessible from Helix server (in the form of **https://hostname** or **http://hostname**)
- **HTH_COMPANY_KEY** = Company key from your company admin bot settings in Helix TeamHub.
- **HTH_ACCOUNT_KEY** = Account key for an admin bot account.
- **VERIFY_SSL** = Verification of the SSL certificate for Helix TeamHub if running in SSL mode. Enabled by default.

Add users and groups in Helix server

With Helix authentication, you add users and groups through the Helix TeamHub user interface. TeamHub then provides the information to Helix server, where it is stored.

To create users and groups, see the following topics in the *Helix TeamHub User Guide*:

- Users
- Groups

To perform these tasks on the Helix server side, see the following sections in the *Helix Core Server Administrator Guide*:

- Managing users for information on user types, adding new licensed users, and renaming users
- Authorizing access for information on the protections table and setting permission levels for users
- Granting access to groups of users for information on creating and editing groups

> **Warning**
> In Helix server, do not alter or edit groups with a name following any of these conventions in any way:
>
> ***HTH-//<depot-path>/<repo-path>-<HTH_ACCESS_LEVEL>***
>
> ***HTH-<depotname>-<hth_access_level>***

```
HTH-<company-admin>
```

Mapping of TeamHub roles to graph depot permissions

Whereas in Helix server, you grant permissions to users or groups, in TeamHub, you assign roles. The following table indicates how each TeamHub role is mapped to specific graph depot permissions in Helix server.

TeamHub Role	Graph Depot Permission
Admin	**admin**
Manager	**write-all**
Master	**force-push** **delete-repo** **create-repo**
Developer	**delete-ref** **create-ref** **write-all**
Guest	**read**

For more information on Helix server permissions, see the `p4 grant-permission` command in the *Helix Core P4 Command Reference*.

For more information on Helix TeamHub roles, see the Roles chapter in the *Helix TeamHub User Guide*.

Including and excluding of Helix server users and groups

You may want to include users or groups, or one or more users in a group, that exist in Helix server in the Helix TeamHub UI, or exclude them from being displayed in the Helix TeamHub UI. This is possible by configuring the respective keys, either in the TeamHub Admin portal, under **Preferences > Helix server > Account and group synchronization**, or in the **`/var/opt/hth/shared/hth.json`** configuration file. For details on these keys, see "Section: pilsner" on page 126 in the Helix TeamHub configuration section.

From the resulting list of users, TeamHub first synchronizes existing bots and then synchronizes all other users.

When synchronizing users, TeamHub proceeds in the following order. TeamHub:

1. Includes direct users
2. Adds users from groups

3. Excludes users from groups
4. Excludes direct users

When synchronizing groups, TeamHub first includes groups and then extracts any groups that are marked as excluded in the configuration.

To include or exclude Helix server users or groups:

1. In the **Preferences** view, under **Account and group synchronization**, enter the name of users or groups as needed.

   For example:

   To include the users called `user1` and `user2`, enter the following in the **Include users** field: `^(user1|user2)$`

   To include users from a group called perforce-group, enter the following in the **Include users from groups** field: `^perforce-group$`

   For details and more examples, see "Section: pilsner" on page 126.

2. Click **Save preferences**.

   A warning appears because the changes have not been applied to the TeamHub server yet. Perform the following step to finalize the configuration.

3. To apply server configuration changes to the TeamHub server, connect to the server via SSH and run the following command:

   ```
   sudo hth-ctl reconfigure
   ```

4. After successfully running the reconfigure command, reload the TeamHub admin portal.

   > **Warning**
   > Failure to do so may result in normal web server interruption messages, such as the **HTTP Error 503. The service is unavailable.** error, because the services come back online.

JSON configuration examples

This section includes examples of how to exclude groups and users by editing the `pilsner` key in the `/var/opt/hth/shared/hth.json` file. Including or excluding users, users from groups, and groups works exactly the same way, so you can apply the following examples to all cases.

To exclude all groups starting with `external-` or `test-` or ending with `test`:

```
^(external-|test-).*, test$
```

To exclude `user1` and `user2`, and any user starting or ending with `test`:

```
^(user1|user2)$, test$, ^test
```

Following is a code snippet from the `hth.json` file with these values included under the `"pilsner"` key:

```
"helix_users_include_regex": "",
"helix_users_exclude_regex": "^(user1|user2)$, test$, ^test",
"helix_users_from_groups_include_regex": "",
"helix_users_from_groups_exclude_regex": "",
"helix_groups_include_regex": "",
"helix_groups_exclude_regex": "^(external-|test-).*, test$"
```

Frequently asked questions (FAQ) about Helix authentication

This section provides answers to commonly asked questions related to Helix authentication.

Question	Answer
If I update a user or group on the Helix server, how long does it take for TeamHub to pick up the change?	By default, TeamHub polls Helix server every 5 minutes for updates. You can configure this interval via an environment variable.
Where can I find a list of all Git related configurables for TeamHub?	A list of environment variables is located in the following location: **`/opt/hth/.profile_backend`**
Can a TeamHub user who has different roles in different projects use the same email address?	TeamHub allows one user to have different roles in different projects, but a user can only be linked to a single email address. Vice versa, a single email address can only be linked to one user.
Why can I log into TeamHub as instance admin with two different passwords?	This may happen as the result of an internal TeamHub failsafe to prevent you from locking yourself out of misconfigured instances. You can always log in to a TeamHub instance with the credentials for the default **`admin`** user. If you have Helix authentication enabled and a user called **`admin`** also exists in Helix server, you can also use the Helix server password to log in to the TeamHub admin portal. See also "TeamHub Administrators" on page 52.

Question	Answer
If I remove a user with admin role in a TeamHub instance from Helix server, this user is still able to log back into TeamHub with all previous admin privileges even though the user no longer exists in Helix server. How do you revoke permissions from a user with an **`admin`** role in a TeamHub instance?	This is related to the previous question. In this scenario, the user called **`admin`** still exists in the built-in admin portal and gets authenticated because the **`admin`** user has a special authentication flow, separate from the normal authentication flow. To remove access to the TeamHub admin portal: 1. Log in to the admin portal: **`<`*`TeamHub-instance-URL`*`>/admin/login`** 2. In the **Admins** view, remove the user from the list of existing administrators. See also "TeamHub Administrators" on page 52.

Troubleshooting Helix authentication

When trying to resolving a problem with Helix TeamHub authentication, start with running the following command as an **`admin`** user on the Helix TeamHub server:

```
hth-ctl tail
```

This command will give you an overview of what is going on in all Helix TeamHub log files. Following is a list of log file locations and descriptions.

Folder	Description
`unicorn_backend`	Unicorn logs for TeamHub backend errors
`puma_pilsner`	TeamHub to Helix server adapter logs
`mongodb`	Mongo database logs for backend
`redis`	Redis storage logs used by backend
`logrotate`	logrotate logs for all log files
`docker_registry`	Logs for docker repositories
`nginx`	All HTTP requests
`resque`	Logs for background jobs
`resque-scheduler`	Logs for scheduled background jobs
`puma`	Logs for websockets
`streamer`	Logs for streaming files from repositories
`maven`	Logs for Maven and Ivy repositories

In addition, the following table may assist you in troubleshooting common issues experienced with Helix server authentication.

	Issue	Root Cause/Resolution
For administrators	Sync with Helix server seems to fail.	Make sure the Helix server user is a valid user in Helix TeamHub. TeamHub only supports user names up to 100 characters while Helix server supports longer names. This gives an error during the sync operation that only appears in the log files. The TeamHub UI does not indicate a problem.

Issue	Root Cause/Resolution
Authentication fails and the logs indicate Redis problems.	This may happen if Redis is configured to save snapshots but cannot persist on disk. You can either turn off snapshot saving or verify that Redis can save to the specified path.

Issue	Root Cause/Resolution
Git repos stored in Helix server are unresponsive.	Helix server may be down.

	Issue	Root Cause/Resolution
	I have added a user in Helix server, but it is not available in the TeamHub UI.	TeamHub probably has not synced the data from Helix server. By default, sync happens every 5 minutes.

Issue	Root Cause/Resolution
The Git repository type I am looking for is not available even though I enabled Helix authentication for the TeamHub instance.	Make sure you reloaded the TeamHub client page after enabling Helix authentication in the TeamHub admin portal. TeamHub fetches instances settings during the initial page load. This means that if you change instance settings in another tab or window while a TeamHub client is already open, it won't retrieve the updated instance settings until you refresh the page.
I cannot log in to the new company I created.	With Helix authentication, TeamHub only supports one company per instance. The option to create additional companies is unavailable. It is recommended that you start from scratch with a new TeamHub instance when using Helix authentication.
The TeamHub trigger script (`hth-trigger.pl`) cannot find the required Perl libraries and fails with a 599/Internal Exception error written to the system log.	**Root cause:** The TeamHub trigger script is written in Perl. In lieu of hardcoding the path to the Perl installation, the script includes the following shebang line to use the Perl installation found in the system: `#!/usr/bin/env perl` However, if the system includes more than one Perl installation, the script might access one that does not include the packages needed on Ubunty Xenial 16.04 (`libnet-ssleay-perl` and `libio-socket-ssl-perl`) to use HTTPS URL for the TeamHub API. As a result, when the SDP (Server Deployment Package) puts `/p4/common/bin/perl` in the `perforce` UNIX user's path before `/usr/bin/perl`, the SDP's Perl code does not know where to find the required Perl libraries. **Workaround:** Change the shebang line in the trigger file to the following: `#!/usr/bin/perl`

	Issue	Root Cause/Resolution
For users	I cannot push anything to a Git repo that is stored in Helix server.	Make sure you have added an SSH public key through the Helix TeamHub UI. For more information, see the Configuring SSH keys section in the *Helix TeamHub User Guide*. If you have added an SSH public key, TeamHub probably has not synced the data. By default, sync happens every 5 minutes. If you are using a self-signed certificate, this may happen because an SSL connection is enforced. Verify that your git configuration has the appropriate setting for `http.sslVerify`.

LDAP Authentication

LDAP is one of the most commonly used application protocols for accessing and maintaining corporate user directories. Helix TeamHub can be configured to use corporate LDAP for authentication. Once LDAP authentication is enabled, all successful login attempts either create a new TeamHub user or update an existing one along with the LDAP group information. The configuration process is explained below.

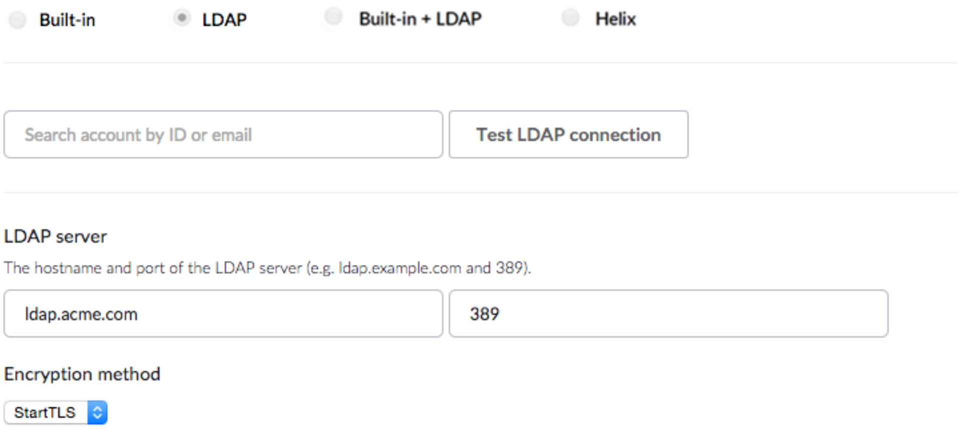
Important
If **Helix** is the selected method of authentication and Helix server uses LDAP authentication, there is no need to use LDAP authentication in TeamHub. Authentication requests from TeamHub are parsed to Helix server, which then connects to the LDAP/AD server to perform the authentication for the user. Note that LDAP authentication on the Helix server side must be established using the *LDAP specification* method (for details, see Authenticating against Active Directory and LDAP servers section in the *Helix Core Server Administrator Guide*).

Configuring LDAP Authentication

To enable LDAP authentication, browse to Helix TeamHub Admin at `http(s)://[hostname]/admin`. From the Admin pane, click on the Preferences link in the navigation bar.

Choose the LDAP authentication option and specify the hostname and port of the LDAP server. The hostname might use a URL like `ldap.acme.com` or an IP like `10.0.0.30`. The port of the LDAP server might vary depending on the connection type. For secure communication between Helix TeamHub and the LDAP server, choose either StartTLS or LDAPS encryption method.

Select `Both` as the authentication option to create local Helix TeamHub users while still using LDAP authentication. Note that once `Both` is enabled, all Helix TeamHub users will be able to set their local passwords and Helix TeamHub will only attempt to bind to LDAP when built in authentication is unsuccessful.



○ Built-in ● LDAP ○ Built-in + LDAP ○ Helix

Search account by ID or email

LDAP server

The hostname and port of the LDAP server (e.g. ldap.example.com and 389).

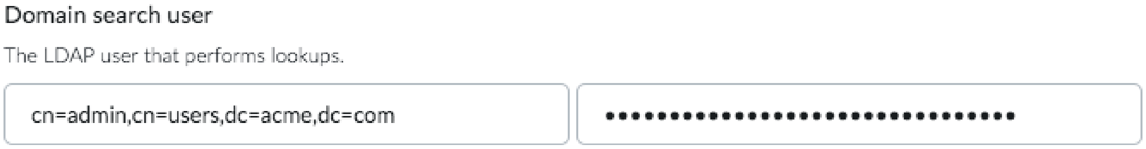
ldap.acme.com | 389

Encryption method

StartTLS

The Domain search user performs lookups to authenticate other accounts when users sign in. The Domain search user is typically a service account used specifically for third-party integrations. For the Domain search user, only read-access to LDAP is needed.

Use the fully qualified user name, which would look something like this: `cn=admin,cn=Users,dc=acme,dc=com`.



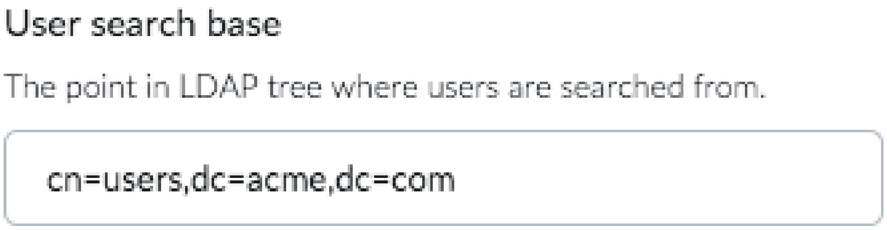
Domain search user

The LDAP user that performs lookups.

cn=admin,cn=users,dc=acme,dc=com | ••••••••••••••••••••••••••••••••

The User search base function specifies the fully qualified name of the starting point in the LDAP tree to search for users. If no search filters are specified, then the User search base will retrieve the entire data set.

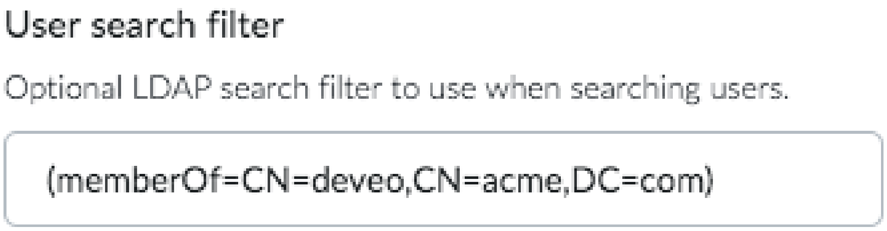
A user search filter can be used to specify conditions that must be met for a record to be included when searching for users. This setting is optional.



User search base

The point in LDAP tree where users are searched from.

cn=users,dc=acme,dc=com



User search filter

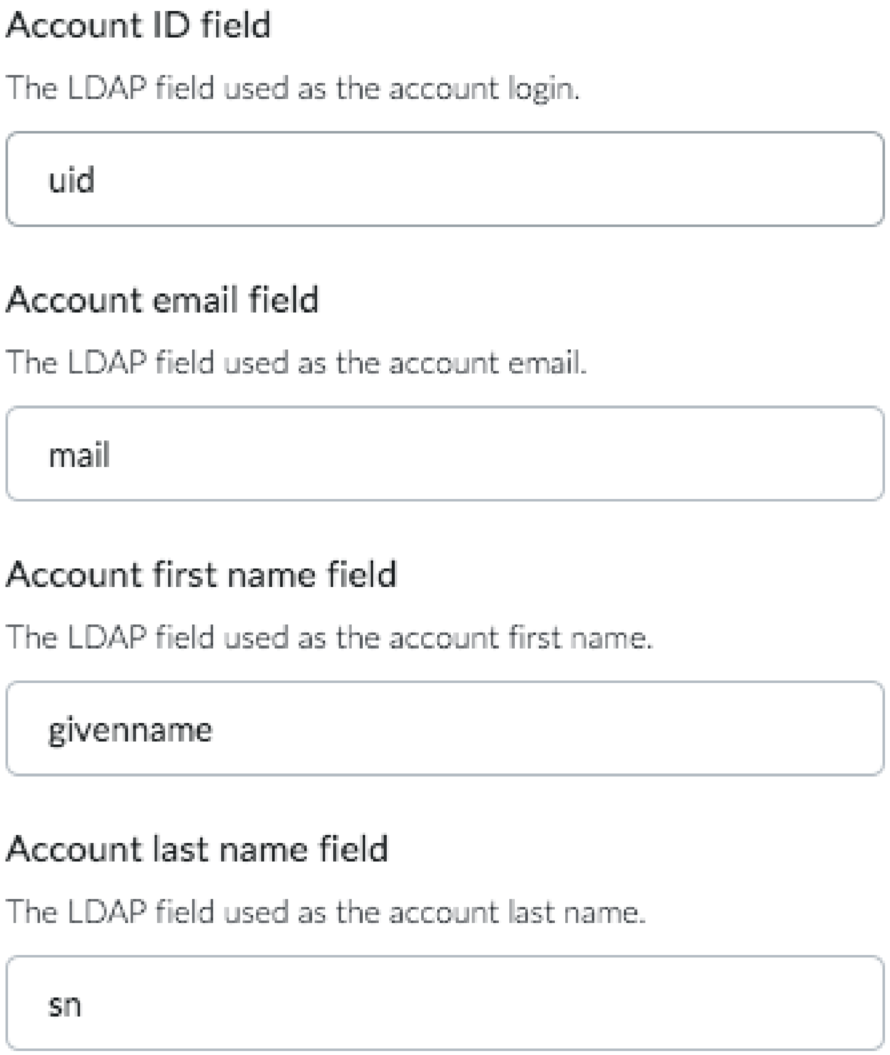
Optional LDAP search filter to use when searching users.

(memberOf=CN=deveo,CN=acme,DC=com)

The Account ID field is the name of the LDAP attribute used as the account login. For most Active Directory installations this will be `sAMAccountName`. For other LDAP solutions like OpenLDAP, the value of this field is usually `uid`.

The Account email field is the name of the LDAP attribute used as the account email. Usually the value of this field is `mail`.

> **Note**
> The Account first name and last name fields are optional.



Account ID field

The LDAP field used as the account login.

uid

Account email field

The LDAP field used as the account email.

mail

Account first name field

The LDAP field used as the account first name.

givenname

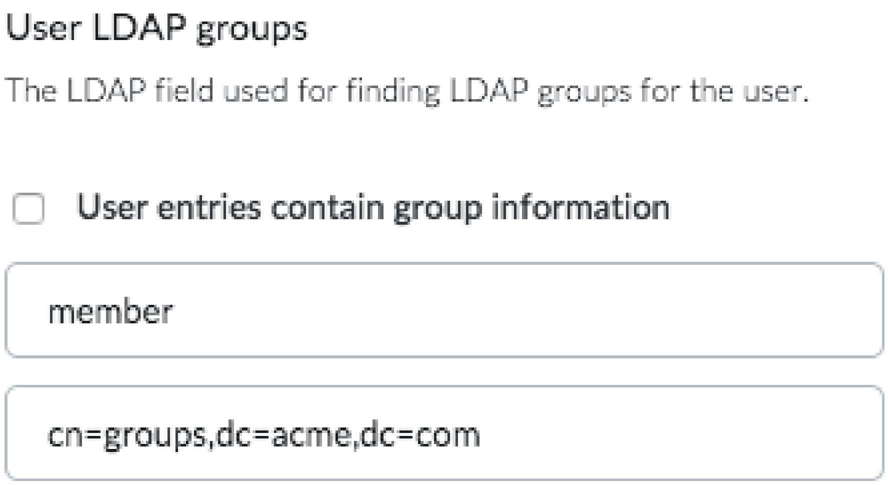
Account last name field

The LDAP field used as the account last name.

sn

The User LDAP groups field is the name of the LDAP attribute used for finding LDAP groups for a user. Check `User entries contain group information` if the directory allows finding LDAP group information directly from user entries. The name of the LDAP field is commonly `memberOf`.

Otherwise set the value of the field to either `member`, `uniqueMember` or `memberUid` depending on the LDAP schema, and set the base path where to search for groups. Nested group support can be enabled for Active Directory by using `member:1.2.840.113556.1.4.1941:` as a value for the field.



User LDAP groups

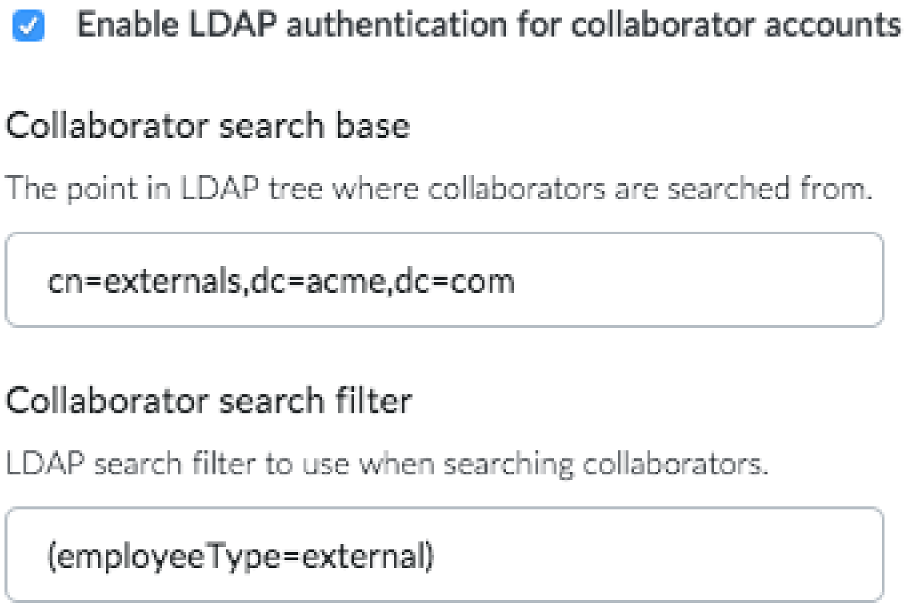
The LDAP field used for finding LDAP groups for the user.

☐ User entries contain group information

member

cn=groups,dc=acme,dc=com

Collaborators use built-in authentication by default. LDAP authentication can also be enabled for collaborators by using a different search base or search filter from normal users. Use the `Test LDAP connection` feature to search for a user and a collaborator account, and make sure it returns only either a user or a collaborator.



☑ Enable LDAP authentication for collaborator accounts

Collaborator search base

The point in LDAP tree where collaborators are searched from.

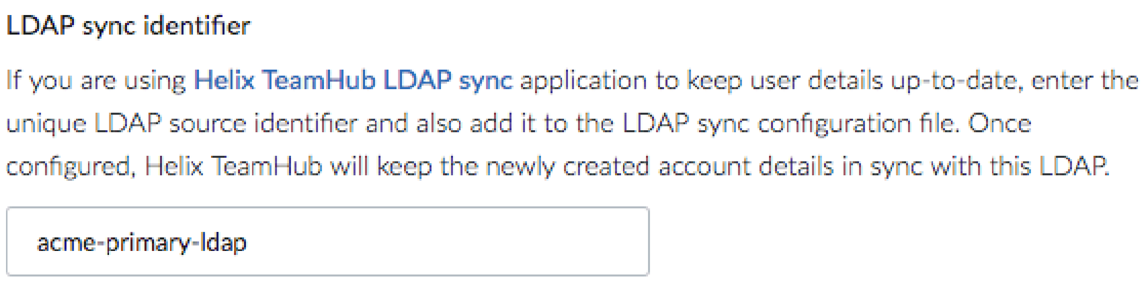
cn=externals,dc=acme,dc=com

Collaborator search filter

LDAP search filter to use when searching collaborators.

(employeeType=external)

If using Helix TeamHub LDAP Sync application to keep user details up-to-date, enter the unique LDAP source identifier and also add it to the LDAP sync configuration file. Once configured, Helix TeamHub will keep the newly created account details in sync with this LDAP.



LDAP sync identifier

If you are using Helix TeamHub LDAP sync application to keep user details up-to-date, enter the unique LDAP source identifier and also add it to the LDAP sync configuration file. Once configured, Helix TeamHub will keep the newly created account details in sync with this LDAP.

acme-primary-ldap

Finally, use the `Test LDAP connection` feature to test the validity of the configuration. Enter an email or ID of an account and verify that correct results are returned.

Further Integration

To go even further with integrating the Helix TeamHub installation to the corporate LDAP, consider keeping user accounts in sync with Helix TeamHub LDAP Sync application.

Caveats

When changing the authentication method back from LDAP to built-in, all users that have been created via LDAP will need to perform a password reset to login again. After the authentication method changes, users will not be able to edit any user attributes that were originally synced from LDAP (e.g. username, email and password).

Helix TeamHub SAML 2.0 Authentication

Helix TeamHub SAML 2.0 authentication allows Single Sign-On for users and collaborators by creating or updating their account through an external Identity Provider.

> **Note**
> Accessing repositories over the HTTPS protocol requires setting up a Helix TeamHub password.

> **Note**
> With Helix authentication, certain restrictions apply to TeamHub functionality. For details, see "Limitations with Helix authentication" on page 12.

Configure SAML Authentication

It is recommended to use SAML over SSL. Configure SSL to your instance first.

Login to Helix TeamHub Client with a Company Admin account (On-premises users: login to your instance), click the company name on the top navigation and select Overview. Click on Company Settings link on top of the page, select Authentication tab, and enable SAML authentication:

Name	Description	Required/Optional
`IdP SSO URL`	Authentication endpoint of the Identity Provider.	`Required`
`IdP certificate`	Certificate of the Identity Provider.	`Required`
`Signed authentication`	When enabled, authentication requests are signed with provided private key.	`Optional`
`Signed metadata`	When enabled, metadata is signed with provided private key.	`Optional`
`Certificate`	Certificate of the Service Provider.	`Optional`
`Private key`	Private key of the Service Provider.	`Optional`

Configuring certificate and private key are optional, but required when signing is enabled. Metadata and authentication requests are signed with SHA1 algorithm (`http://www.w3.org/2000/09/xmldsig#rsa-sha1`) when enabled.

Certificates (X.509) and private key (RSA) must be given in PEM format, with base64 encoded content between header and footer lines. A self-signed certificate and private key can be created with openssl:

```
openssl req -newkey rsa:2048 -new -x509 -days 3652 -nodes -out saml.crt -
keyout saml.key
```

Attributes

Identity Provider is expected to return following attributes in authentication response:

Name	Description	Required/Optional
`email`	Unique email of the account.	`Required`
`short_name`	Unique username of the account. Generated from email if not provided.	`Optional`
`first_name`	Given name of the account.	`Optional`
`last_name`	Surname of the account.	`Optional`
`type`	Defines type of the account to create.	`Optional`

By default a user account is created on the first login. If the provided attributes include a `type` attribute and its value equals the configured collaborator type value, a collaborator account is created instead. Attribute name mapping can be optionally configured in Attribute mapping section.

Metadata

Helix TeamHub supports SAML 2.0 Web Browser SSO Profile with Service Provider initiated HTTP Redirect binding for authentication requests and HTTP POST binding for responses. Service Provider metadata is available in `http(s)://[hostname]/account/saml/[company]/metadata` after enabling SAML authentication in the company.

- Service Provider Entity ID: `http(s)://[hostname]/account/saml/[company]/metadata`
- Authentication initialization: `http(s)://[hostname]/account/saml/[company]/init`
- Assertion Consumer Service: `http(s)://[hostname]/account/saml/[company]/consume`
- Name ID format: `urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified`

Troubleshooting

- Make sure correct Identity Provider certificate is configured in the settings.
- Update metadata on Identity Provider after changing settings.
- Make sure server time is synced between Helix TeamHub and Identity Provider.
- Inspect Identity Provider and Helix TeamHub logs.

Helix TeamHub LDAP Interface

Helix TeamHub LDAP interface allows integrating external tools to Helix TeamHub users, bots, and groups, and use of Helix TeamHub as an authentication provider.

> **Note**
> Helix TeamHub LDAP interface is available to On-premises customers with a license of at least 7 seats.

Upgrading from Helix TeamHub 3.10.0 and older versions

Helix TeamHub 3.11.0 added support for bot accounts and introduced additional accounts branch. Configurations should be updated to use the new `ou=users,ou=accounts` branch instead of the old `ou=users` branch (see Directory Tree below). Binding and searching users by using the old branch is still supported until the next major release.

Enable Helix TeamHub LDAP Interface

The following steps describe how to enable the LDAP interface in TeamHub. For advanced configuration parameters, see "Section: backend" on page 116.

You can also perform a full synchronization of external tools by enabling hashed passwords. However, hashed passwords are only visible to company administrators. Existing Helix TeamHub users must change their password to make it available.

> **Note**
> You cannot enable hashed passwords with pass-through LDAP authentication.

To enable the LDAP interface:

1. Browse to your Helix TeamHub instance at `http(s)://[`*`hostname`*`]/admin`.
2. In the navigation pane on the left, click **Preferences**.
3. In the **Configure instance** view, under **Helix TeamHub LDAP interface**, select the **Enable Helix TeamHub LDAP interface** check box.
4. To perform a full synchronization of external tools, select the **Enable SHA hashed user passwords** check box.
5. Click **Save preferences**.

Configure Helix TeamHub LDAP with External Tools

To access Helix TeamHub LDAP, use either the unencrypted `ldap://example.com:389` , or encrypted `ldaps://example.com:636` (LDAPS) URLs. Configure SSL in order to use LDAPS.

The following table includes commonly used settings to use with external tools. OpenLDAP may be used as a base configuration if the tool provides pre-configured settings. Helix TeamHub LDAP interface supports read-only access.

Setting	Description	Examples
Base DN	The root node of the LDAP to search from. Use company ID in place of `example`.	`o=example`
Additional account DN	Prepended to the base DN to search users and bots. The complete DN will be `ou=accounts,o=example`.	`ou=accounts`
Additional user DN	Prepended to the base DN to search users. The complete DN will be `ou=users,ou=accounts,o=example`.	`ou=users,ou=accounts`
Additional bot DN	Prepended to the base DN to search bots. The complete DN will be `ou=bots,ou=accounts,o=example`.	`ou=bots,ou=accounts`
Additional group DN	Prepended to the base DN to search groups. The complete DN will be `ou=groups,o=example`.	`ou=groups`
Username and password	Provide a full user DN of a company admin account and a password. These credentials are used when binding to LDAP. Anonymous binding is not supported.	`uid=norris,ou=users,ou=accounts,o=example`

Setting	Description	Examples
Account search filter	Use `inetOrgPerson` to search for accounts by objectClass. Use `employeeType` attribute to search for accounts by type (`bot` or `user`). Use `uid` attribute to search for a specific account.	`(objectClass=inetOrgPerson)(employeeType=user)(uid=norris)`
Group search filter	Use `groupOfNames` to search for groups by objectClass. Use `cn` attribute to search for a specific group.	`(objectClass=groupOfNames)(cn=developers)`
Member search filter	Use `member` attribute to find groups for a user. The value of the attribute contains the full user DN.	`(member=uid=norris,ou=users,ou=accounts,o=example)`
MemberOf search filter	Use `memberOf` attribute to find users for a group. The value of the attribute contains the full group DN.	`(memberOf=cn=developers,ou=groups,o=example)`
Account password attribute	Account password will be visible to company admins in `userPassword` attribute when SHA hashed password setting is enabled.	`{SHA}5en6G6MezRroT3XKqkdPOmY/BfQ=`
Unique identifier	A universally unique identifier is available in `entryUUID` attribute.	`cdfd2ece-c1db-4c76-ae45-2d75968afddd`

Helix TeamHub LDAP Structure and Example Entries

Directory Tree

```
  o=example
    ├── ou=accounts
    │   ├── ou=users
    │   │   └── uid=norris
    │   └── ou=bots
    │        └── uid=bot
    ├── ou=groups
    │   └── cn=developers
    └── ou=projects
        └── ou=sample
```

Example User Entry

```
# norris, users, accounts, example
dn: uid=norris,ou=users,ou=account,o=example
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
uid: norris
cn: Chuck Norris
displayName: Chuck Norris
givenName: Chuck
sn: Norris
mail: norris@example.com
telephoneNumber: +123456
description: Not needed
title: Champion
entryUUID: cdfd2ece-c1db-4c76-ae45-2d75968afddd
memberOf: cn=developers,ou=groups,o=example
memberOf: cn=managers,ou=groups,o=example
employeeType: user
```

Example Bot Entry

```
# bot, bots, accounts, example
dn: uid=bot,ou=bots,ou=accounts,o=example
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
uid: bot
cn: bot
displayName: bot
givenName: bot
sn: bot
entryUUID: 21f02b44-3832-4523-b7b4-c3602932535e
employeeType: bot
mail: bot@helixteamhub.invalid
```

Example Group Entry

```
# developers, groups, example
dn: cn=developers,ou=groups,o=example
objectClass: top
objectClass: groupOfNames
cn: developers
description: All our developers
entryUUID: 3c9ad9eb-6234-4cf2-b147-f2d945d77b67
member: uid=norris,ou=users,ou=accounts,o=example
member: uid=bruce,ou=users,ou=accounts,o=example
```

Example Project Entry

```
# sample, projects, example
dn: ou=sample,ou=projects,o=example
objectClass: top
objectClass: organizationalUnit
ou: sample
description: Sample project
entryUUID: a671a3bb-edb9-45f7-aa71-f3be44a075c2
```

Controlling Helix TeamHub

As described in the Architecture section, Helix TeamHub is composed of popular open-source technologies. Helix TeamHub runs and monitors a number of services, which are responsible for executing user requests, scheduling, and running background operations.

Helix TeamHub comes with the `hth-ctl` tool, which helps manage Helix TeamHub services and configuration, and aids in performing administrative tasks. This is the same tool used during the Helix TeamHub installation and upgrades.

Control Commands

Retrieve the list of all available commands by running `sudo hth-ctl`. The most commonly used commands are listed below:

- `status` - Show the status of all the services.
- `report` - Generates system report for Helix TeamHub troubleshooting.
- `service-list` - List all the services (enabled services appear with a *).
- `show-config` - Show the configuration that would be generated by reconfigure.
- `tail` - Watch the service logs of all enabled services.
- `start` - Start services if they are down, and restart them if they stop.
- `restart` - Stop the services if they are running, then start them again **(causes slight downtime)**.

Warning! Some control commands are destructive! Be extremely careful when executing unfamiliar switches.

Services

Helix TeamHub runs and monitors a number of services using the runit service supervision tool. The list below outlines all available services on the Helix TeamHub servers. In Cluster or HA deployment, the services will be spread across the Web and DB roles.

- `anycable` - Daemon for WebSocket connections together with moonshine.
- `apache` - Apache Web server for handling version control operations.
- `docker_registry` - Daemon for Docker repositories.
- `ldap` - Daemon for LDAP protocol of the Helix TeamHub LDAP Interface.
- `ldaps` - Daemon for LDAPS protocol of the Helix TeamHub LDAP Interface.
- `logrotate` - Logrotate daemon for rotating application logs.
- `maven` - Daemon for Maven repositories.
- `mongodb` - MongoDB database stores entire application data.

- **`moonshine`** - Daemon for WebSocket connections together with anycable.
- **`nginx`** - Nginx proxies all users requests to other services.
- **`puma_pilsner`** - Daemon for Perforce integration.
- **`redis`** - Redis database stores intermediate data, such as background jobs and events.
- **`resque`** - Processes background jobs, such as hooks, notifications, events, backups.
- **`resque_scheduler`** - Schedules resque background jobs.
- **`slave_syncer`** - Daemon for performing replication tasks on slave site.
- **`streamer`** - Daemon for streaming files.
- **`unicorn_backend`** - Unicorn Web server serving Helix TeamHub APIs, including authorization.

Logging

As mentioned in Controlling Services, Helix TeamHub uses runit to run and monitor its services. Two types of logs are created:

- Logs written by svlogd for the services that write to stdout, for example: **`resque`** service.
- Logs written by the service directly, for example **`apache`** service.

The log files are located at **`/var/log/hth`**, and its recommended to keep them on a separate partition. The rule of thumb is, if the **`/var/log/hth/<service>/current`** (maintained by svlogd) file is empty, look for **`*.log`** files (maintained by the service directly) within the same directory.

To see all logs in real-time, use the "tail" switch provided by the Helix TeamHub control utility:

```
sudo hth-ctl tail
```

Log Rotation

The svlogd logs are rotated by the daemon itself. However, the logs written by the services directly are rotated with logrotate. Regardless of the type, by default all logs are rotated and compressed **daily** and kept for **30 days**. See and customize log rotation settings through logging configuration flags.

Audit Logs

In addition to logs created by the services, Helix TeamHub stores a number of audit logs. These logs are in JSON format located under **`/var/log/hth/audit`**. By default, the audit logs are rotated and compressed **daily** and kept for **90 days**, however they can be fine-tuned through logging configuration flags.

Backups and Restoration

Backups are available as part of the Helix TeamHub packages, and can be activated via configuration flags. The nature of backups changes from one TeamHub component to another:

Component	Method	Schedule	Descrption
Assets	Archival	Daily at **00:00**	Attachments, Avatars, Logs, Configuration files
MongoDB	Archival	Daily at **00:00**	Mongo dumps
Repositories	Incremental sync	Daily at **02:00**	Repository type specific tools: git, svnsync, hg
Docker Registry	Archival	Daily at **04:00**	Docker images

Preparation

TeamHub stores backups at `/var/opt/hth/backups`. For production deployments, the recommendation is to mount a reliable external storage that has enough space to accommodate at least 1.5 times the TeamHub data set (`/var/opt/hth/shared`).

Enabling backups

To have backups running, activate them through the configuration flags as shown below. With replication enabled, backups can only be configured on the master site.

Combo

1. Merge the following configuration to `/var/opt/hth/shared/hth.json` and **make sure to add the backup settings under the existing keys if they already exist in the configuration**.

```
{
  "backend": {
    "backups": true
  },
  "mongodb": {
    "backups": true
  },
  "repos": {
    "backups": true
  },
  "docker_registry": {
        "backups": true
  }
}
```

2. Apply the changes by reconfiguring TeamHub:

   ```
   sudo hth-ctl reconfigure
   ```

Cluster and HA

In Cluster and HA deployment, the TeamHub services are distributed across the server roles. To enable backups, activate the backup flags in `/var/opt/hth/shared/hth.json` on the appropriate servers. **Make sure to add the backup settings under the existing keys if they already exist in the configuration**.

Helix TeamHub DB

```
{
  "mongodb": {
    "backups": true
  }
}
```

By default, mailing is not configured for the DB node. To get notifications if failures occur during DB backups, you need to add the following:

1. Configure the `postfix` section of `hth.json` to be identical with the one on your web node.
2. Add the email that will receive the notifications to the `app` section of `hth.json`:

   ```
   {
     "app": {
       "email": "support@acme.com"


     }
   }
   ```

Helix TeamHub Web

```
{
  "backend": {
    "backups": true
  },
  "repos": {
    "backups": true
  },
  "docker_registry": {
       "backups": true
  }
}
```

Apply the changes by reconfiguring TeamHub on each server:

```
sudo hth-ctl reconfigure
```

Configuring how many backups to keep before oldest backup gets removed

You can configure archival mode backups to prune old backups. The configuration below keeps the 30 latest backup archives and deletes the oldest when a new backup occurs.

```
{
  "backups": {
    "keep": 30
  }
}
```

A separate setting exists for the Docker Registry:

```
{
  "docker_registry": {
    "backups_keep": 5
  }
}
```

Unlike with other backups, the Docker Registry **`backups_keep`** value should be low because images tend to take up a lot of space, even if they are compressed.

Restoring Backups

Because TeamHub backups are modular, when it comes to restoring the system from a backup, it is important to consider the following:

- Follow the correct order of restoring the backup components:
  1. Assets
  2. Database
  3. Repositories
- Because backups for TeamHub components are taken daily, it is important to restore all components from the same day. Doing otherwise will lead to data inconsistency.

See Backups and Restoration for more information on how to restore backups with replication enabled.

Stopping Services

Before starting the restoration process, it is a good idea to stop all the Helix TeamHub services:

```
sudo hth-ctl stop
```

Restoring Assets

1. SSH into one of the TeamHub Web servers (or the Combo) and switch to the **`hth`** user.
2. Copy the **`backend_backup.tar`** from backup storage, which is located at **`/var/opt/hth/backups/backend_backup/<date>/`**.
3. Extract the archives and restore:

   ```
   sudo su - hth
   tar xvf backend_backup.tar
   cd backend_backup/archives; ls *.tar.gz | xargs -i tar xvf {} -C /
   ```

Restoring MongoDB Database

1. SSH into the TeamHub DB server (or the Combo) and switch to the **`hth`** user.
2. Copy **`mongodb_backup.tar`** from the backup storage **`/var/opt/hth/backups/mongodb_backup/<date>/`**.
3. Start MongoDB, extract the archives, and restore:

   ```
   sudo su - hth
   sudo hth-ctl start mongodb
   ```

```
tar xvf mongodb_backup.tar; cd mongodb_backup/databases/; tar xvf
MongoDB.tar.bz2
```

For Combo deployment without MongoDB authentication, use the following command to restore the database:

```
mongorestore --port 4002 --drop MongoDB/
```

For Cluster, HA, Replication or Combo deployments with MongoDB authentication, use admin credentials to restore the database:

```
mongorestore --port 4002 -u <admin-username> -p <admin-password> --
drop MongoDB/
```

Restoring Repositories

Repositories are stored under the **`/var/opt/hth/shared/companies/<company>/projects/<repo_type>/repositories`** directory. To restore backups from the backups store:

1. SSH into one of the TeamHub Web servers (or the Combo) and switch to the **`hth`** user.
2. Run the following script:

```
#!/bin/bash

companies=`ls /var/opt/hth/backups/repos_
backup/var/opt/hth/shared/companies/`
# Loop through companies
for company in $companies; do
  # Loop through projects
  projects=`ls /var/opt/hth/backups/repos_
backup/var/opt/hth/shared/companies/$company/projects/`
  for project in $projects; do
    echo "Restoring repositories for project $project in company
$company"
    repos_
dest="/var/opt/hth/shared/companies/$company/projects/$project/reposi
tories/"
    if [ ! -d $repos_dest ]; then
      # Company/Project may have been renamed
      echo "Creating $repos_dest"
      mkdir -p $repos_dest
    fi
    rsync -av --delete /var/opt/hth/backups/repos_
backup/var/opt/hth/shared/companies/$company/projects/$project/reposi
tories/ $repos_dest
  done
done
```

3. To restore the repositories hooks, execute the following commands to regenerate them:

```
cd /opt/hth/application/backend/current
rake hth:restore:all
```

Restoring Docker Registry

1. SSH into one of the Helix TeamHub Web servers (or the Combo) and switch to the **`hth`** user.
2. Copy **`docker_registry_backup.tar`** from the backup storage **`/var/opt/hth/backups/docker_registry_backup/`*`<date>`*`/`**.
3. Extract the archives and restore:

```
sudo su - hth
rm -rf /var/opt/hth/shared/storage/docker_registry/docker/
cd /var/opt/hth/backups/docker_registry_backup/<date>
tar xvf docker_registry_backup.tar
cd docker_registry_backup/archives; ls *.tar.gz | xargs -i
tar xvf {} -C /
```

Starting Helix TeamHub

Start back all TeamHub services:

```
sudo hth-ctl start
```

Reconfiguring Helix TeamHub

If there have been changes to the **`hth.json`** restored from backups, apply those changes by running the **`reconfigure`** command:

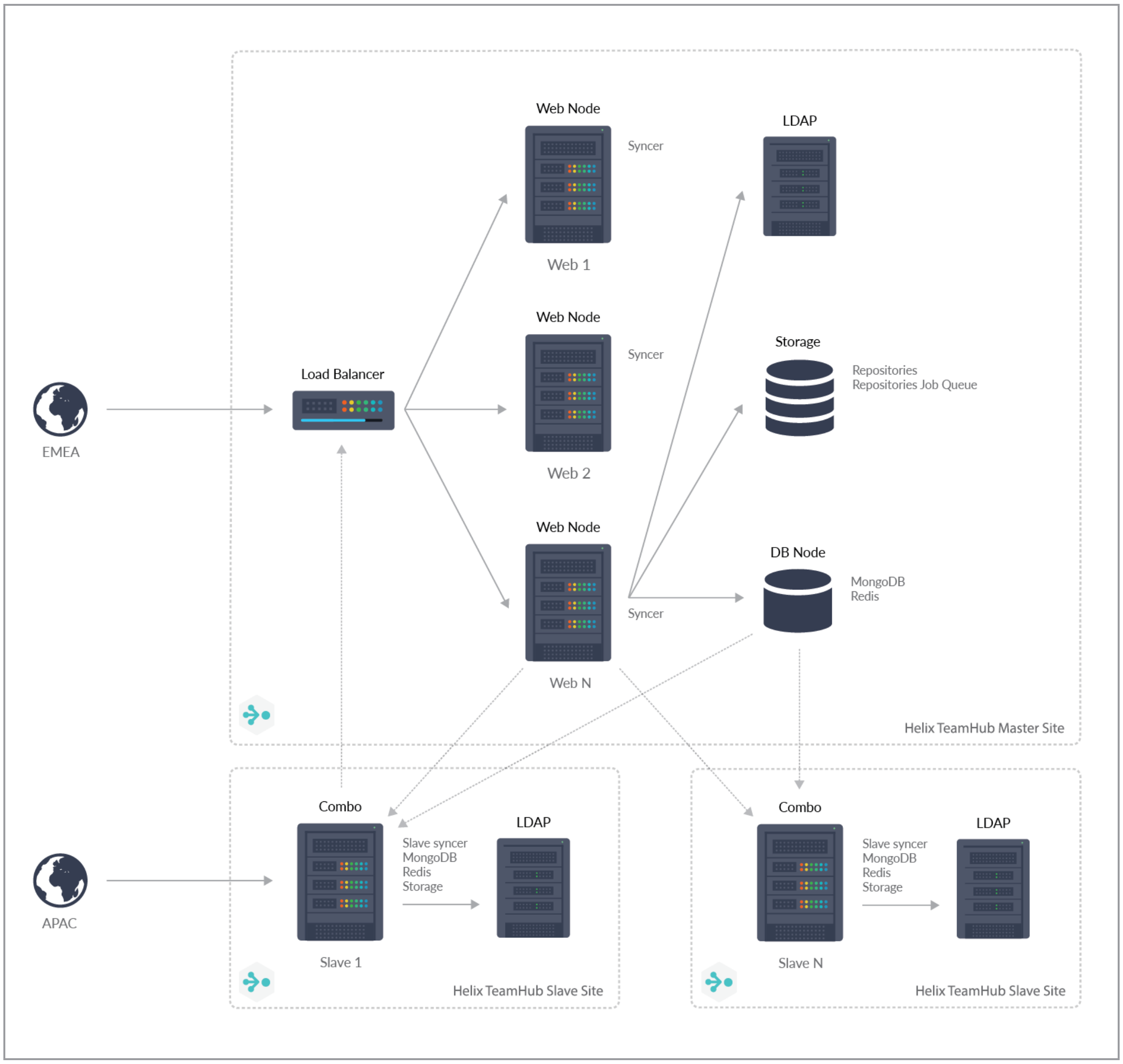
```
sudo hth-ctl reconfigure
```

Setup and Management of Subversion Master-Slave Replication

Note

- Helix TeamHub only supports master-slave replication for Subversion repositories.
- Replication is available as a separate paid feature. Please contact Helix TeamHub Sales to upgrade your plan and to enable this feature.

This section guides you through the steps of setting up and maintaining master-slave sites for Subversion replication. Slave sites can be used to provide faster read access to repositories with HTTP access protocol (write operations will be proxied to master). SSH repository access is not supported on slave sites.



This section contains the following information:

- Prerequisites
- Configure Master Site
- Configure Slave Site
- Replicate a Subversion Repository
- Remove Replication of a Repository
- Update Site Details
- Remove Slave Site

- Backups and Restoration
- Troubleshooting

Prerequisites

- Helix TeamHub license with replication feature.
- Subversion replicator script.
- Master site can be any of the Helix TeamHub Setup types but slave sites can only be Combo setups.
- All the machines in the replication setup must have proper hostname set for the system.
- Make sure sites are able to connect to each member and required ports are open.
- Converting an existing Helix TeamHub instance requires downtime.

Configure Master Site

1. Install and setup **master** site normally using Helix TeamHub Admin and add replication license. Make sure correct hostname is set also for the system on all the nodes.
2. Create a company admin bot for replication and take a note of the credentials.
3. Create admin and application accounts for MongoDB by executing `create_mongodb_users.sh` as hth user on the **master** site. Use existing credentials if you have already created them.
4. Generate a keyfile as hth user to be used with MongoDB replica set authentication. Place this file on DB node on cluster setups.

```
cd /var/opt/hth/shared
openssl rand -base64 741 > mongodb-keyfile
chmod 600 mongodb-keyfile
```

5. Cluster setups only: Edit hth.json on DB node on **master** site. **Make sure to merge values for the existing keys**.

```
{
  // Setup site as master.
  "app": {
    "is_master": true
  }
}
```

6. Combo setups only: Edit hth.json on **master** site. **Make sure to merge values for the existing**

keys.

```
{
  // Add application user credentials for MongoDB.
  "mongodb": {
    "username": "<input db username here>",
    "password": "<input db password here>"
  }
}
```

7. Edit hth.json on Web node (or Combo) on **master** site. **Make sure to merge values for the existing keys**.

```
{
  // Setup site as master.
  "app": {
    "is_master": true,
    "hostname": "master.helixteamhub.dev"
  }

  // Enable and add company admin bot credentials for replication.
  "replication": {
    "enable": true,
    "username": "<input sync username here>",
    "password": "<input sync password here>"
  }

  // Add application user credentials for MongoDB.
  "backend": {
    "db_username": "<input db username here>",
    "db_password": "<input db password here>"
  }
}
```

8. Reconfigure only DB node (or Combo) on **master** site.

```
sudo hth-ctl reconfigure
```

9. Connect to mongo console on **master** site (DB node or Combo) to initialize MongoDB replica set.

Connect to MongoDB.

```
mongo --port 4002
```

Authenticate with admin account.

```
use admin
db.auth("<input admin username here>", "<input admin password here>")
```

Initialize replica set (replace correct hostname for master).

```
var config = {
  _id: "hth",
  members: [
    {
     _id: 0,
     host: "master.helixteamhub.dev:4002",
     priority : 1
    }
  ]
}
rs.initiate(config)
```

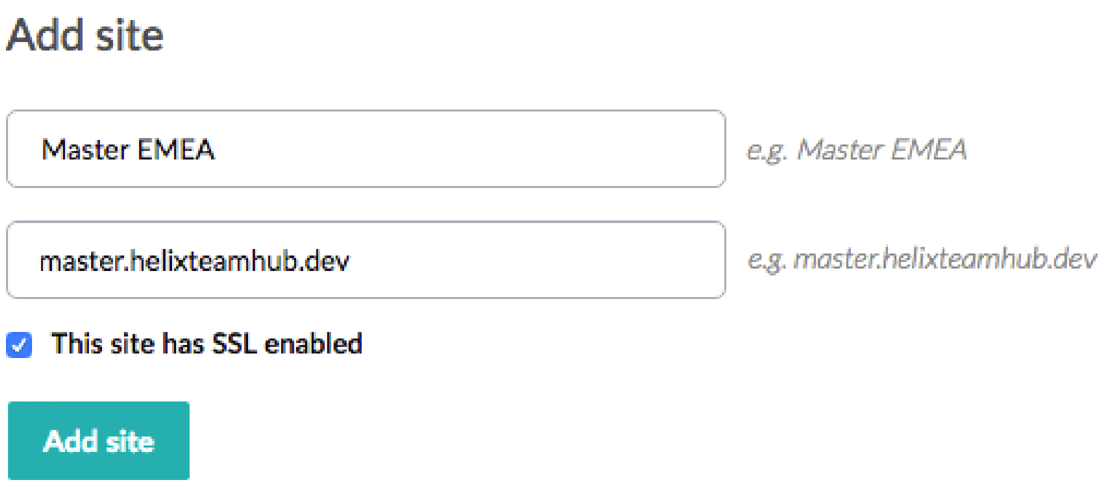
Check status of replica set and wait until primary is available. You should see a member with **`"stateStr": "PRIMARY"`**.

```
rs.status()
```

10. Cluster setups only: Reconfigure rest of the Web nodes on **master** site.

    ```
    sudo hth-ctl reconfigure
    ```

11. Install Subversion replicator on the **master** site in **`/opt/hth/application/backend/current/bin/svnreplicator.sh`**.
12. Login to Helix TeamHub Admin on **master** site and create the master site in the Sites section.



Add site

Master EMEA

e.g. Master EMEA

master.helixteamhub.dev

e.g. master.helixteamhub.dev

☑ This site has SSL enabled

Configure Slave Site

1. Install Helix TeamHub **slave** site as Combo and mark the instance as slave before configuring Helix TeamHub in step 2.

   ```
   touch /var/opt/hth/hth_slave
   sudo hth-ctl reconfigure
   ```

   Bootstrap the instance using Helix TeamHub Admin and add replication license. Make sure correct hostname is set also for the system.

2. Copy mongodb-keyfile from **master** site and place it in **`/var/opt/hth/shared/mongodb-keyfile`**. Make sure it is owned by hth account and has correct permissions.

   ```
   cd /var/opt/hth/shared/
   scp root@master.helixteamhub.dev:/var/opt/hth/shared/mongodb-keyfile
   .
   chmod 600 mongodb-keyfile
   chown hth.hth mongodb-keyfile
   ```

3. Edit hth.json on the **slave** site. **Make sure to merge values for the existing keys**.

   ```
   {
     // Setup site as slave.
     "app": {
       "is_slave": true,
       "hostname": "slave1.helixteamhub.dev"
   ```

```
  }

  // Add application user credentials for MongoDB and enable slave
syncer.
  "backend": {
    "db_username": "<input db username here>",
    "db_password": "<input db password here>",
    "slave_syncer_enabled": true
  }

  // Disable unnecessary services.
  "resque": {
    "enable": false
  },
  "resque_scheduler": {
    "enable": false
  },
  "sangria": {
    "enable": false
  },
  "streamer": {
    "enable": false
  },
  "puma": {
    "enable": false
  },
  "maven": {
    "enable": false
  }
}
```

4. Reconfigure **slave** site.

```
sudo hth-ctl reconfigure
```

5. Connect to mongo console on **master** site (DB node or Combo) and add new **slave** to the replica

set.

Connect to MongoDB.

```
mongo --port 4002
```

Authenticate with admin account.

```
use admin
db.auth("<input admin username here>", "<input admin password here>")
```

Copy current configuration to a variable and add new slave member with 0 votes and 0 priority and unique _id.

```
var config = rs.conf()

config.members.push({
  "_id" : 1,
  "host" : "slave1.helixteamhub.dev:4002",
  "votes" : 0,
  "priority" : 0
})
```

Verify modified configuration.

```
config
```

Reconfigure replica set with new config.

```
rs.reconfig(config)
```

Wait for the slave to sync up into secondary state. You should see a member with `"stateStr": "SECONDARY"`.

```
rs.status()
```

6. Login to Helix TeamHub Admin on **master** and create a new slave site in the Sites section.

Replicate a Subversion Repository

See Replicate a Large Subversion Repository first before enabling replication.

Log in to Helix TeamHub Client on **master** site as a company or project admin and select Replication tab from repository settings. Select slave sites where the repository should be relicated to. The time until the slave repository is available for use depends on the size of the repository.

Slave APAC (slave1.helixteamhub.dev)

Replicate a Large Subversion Repository

Initial replication can take a long time for large repositories. Consider transferring large repositories to slaves by other means before enabling replication. One option is to rsync a hotcopy in order to avoid issues with ongoing commits:

1. Create a hotcopy of the existing repository on master site as hth user.

   ```
   svnadmin hotcopy /var/opt/hth/shared/companies/$company/projects/$project/repositories/subversion/$repo /tmp/$repo
   ```

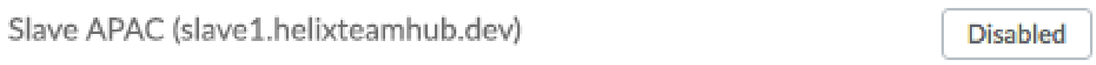
2. Transfer the repository copy to slaves without hooks (make sure file and directory permissions remain for hth user).

   ```
   rsync -a --exclude hooks/* /tmp/$repo/ root@slave1.helixteamhub.dev:/var/opt/hth/shared/companies/$company/projects/$project/repositories/subversion/$repo
   ```

3. Follow the steps above in Replicate a Subversion Repository.

Remove Replication of a Repository

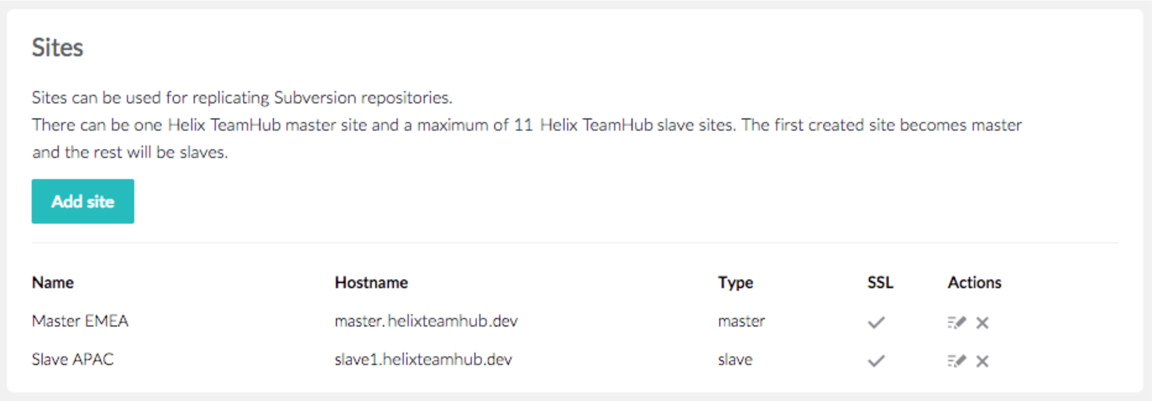
Login to Helix TeamHub Client on **master** site as a company or project admin and select Replication tab from repository settings. Mark removed sites as disabled.



Slave APAC (slave1.helixteamhub.dev) Disabled

Update Site Details

Login to Helix TeamHub Admin on **master** and update site details in the Sites section. Update **`hth.json`** and reconfigure the site when changing hostname or SSL (restart also slave_syncer **`sudo hth-ctl restart slave_syncer`** for a slave site). Update MongoDB replica set configuration after changing hostname for a site.



Sites

Sites can be used for replicating Subversion repositories.
There can be one Helix TeamHub master site and a maximum of 11 Helix TeamHub slave sites. The first created site becomes master and the rest will be slaves.

Name	Hostname	Type	SSL	Actions
Master EMEA	master.helixteamhub.dev	master	✓	✎ ✕
Slave APAC	slave1.helixteamhub.dev	slave	✓	✎ ✕

Remove Slave Site

1. Login to Helix TeamHub Admin on **master** site and delete the targeted slave site in the Sites section.
2. Connect to mongo console on **master** site (DB node or Combo) and remove **slave** site from replica set:

   Connect to MongoDB.

   ```
   mongo --port 4002
   ```

   Authenticate with admin account.

   ```
   use admin
   db.auth("<input admin username here>", "<input admin password here>")
   ```

   Copy current configuration to a variable and remove the targeted slave from members.

   ```
   var config = rs.conf()

   // Check the zero-based index of the slave.
   config.members

   // Remove the slave (index is the first argument).
   config.members.splice(1, 1)
   ```

   Verify modified configuration.

   ```
   config
   ```

   Reconfigure replica set with new configuration.

   ```
   rs.reconfig(config)
   ```

Check status of replica set.

```
rs.status()
```

3. Shutdown and destroy the **slave** site instance.

Backups and Restoration

Backups can be configured normally on **master** site, but restoration follows different process.

1. Remove slave sites by following the steps in Remove Slave Site.
2. Follow steps in Restoring Backups on **master** site.
3. Create slave sites by following the steps in Configure Slave Site.
4. Export metadata about replicated repositories on **master** site as hth user.

   ```
   cd /opt/hth/application/backend/current
   rake hth:replication:export_metadata
   ```

5. Optional. Consider transferring repositories to slaves by other means before proceeding to the next step. See Replicate a Large Subversion Repository for more information. Metadata of each slave is saved in **`/var/opt/hth/shared/slave_metadata/`** as json files, which can be used to obtain a list of repositories.

   ```
   cat /var/opt/hth/shared/slave_metadata/slave1.helixteamhub.dev.json
   {
     "slave_host": "slave1.helixteamhub.dev",
     "repositories": [
       {
         "company": "hth",
         "project": "platform",
         "repository": "docs",
         "path":
   "/var/opt/hth/shared/companies/hth/projects/platform/repositories/sub
   version/docs"
       }
     ]
   }
   ```

6. Run slave setup task on **master** site as hth user for each slave. Replace filename with correct metadata file.

```
cd /opt/hth/application/backend/current
rake hth:replication:setup_slave[/var/opt/hth/shared/slave_
metadata/$slave_host_name.json]
```

Troubleshooting

- Use mongo console to check replica set status.
- Check slave_syncer logs on slaves `/var/log/hth/slave_syncer/`.
- Check replication logs on master `/var/log/hth/replication/`.

Docker Registry

Helix TeamHub supports Docker image repositories. You can store and pull images via Docker engine version 1.6.0 or later.

Requirements

For the use of Docker repositories, SSL must be enabled. The certificate must be signed by a trusted Certificate Authority.

Storage Driver

Docker Registry uses the **filesystem** as the default storage driver, but it can be configured to utilize other drivers.

Driver	Description
filesystem	Local filesystem
s3	Amazon Simple Storage Service
azure	Microsoft Azure Blob Storage
swift	Openstack Swift
oss	Aliyun OSS
gcs	Google Cloud Storage

Configuration

To push or pull images, clients must be able to access storage backends (other than **filesystem**) directly.

Note
TeamHub can only back up Docker images when the storage driver is **filesystem**.

To change the storage driver:

1. Edit `hth.json` and provide appropriate configuration:

   Default configuration:

   ```
   {
     "docker_registry": {
       "storage_driver": "filesystem",
       "storage_settings": {
         "rootdirectory": "/var/opt/hth/shared/storage/docker_registry/"
       }
     }
   }
   ```

   ```
   {
     "docker_registry": {
       "storage_driver": "s3",
       "storage_settings": {
         "accesskey": "s3-access-key",
         "secretkey": "s3-secret-key",
         "bucket": "s3-bucket",
         "region": "s3-region"
       }
     }
   }
   ```

   ```
   {
     "docker_registry": {
       "storage_driver": "azure",
       "storage_settings": {
         "accountname": "azure-storage-account-name",
         "accountkey": "azure-storage-account-key",
         "container": "azure-storage-container"
       }
   ```

```
  }
}
```

```
{
  "docker_registry": {
    "storage_driver": "swift",
    "storage_settings": {
      "authurl": "auth-token-url",
      "username": "openstack-username",
      "password": "openstack-password",
      "container": "swift-container",
      "region": "openstack-container-region"
    }
  }
}
```

```
{
  "docker_registry": {
    "storage_driver": "oss",
    "storage_settings": {
      "accesskeyid": "aliyun-oss-access-key-id",
      "accesskeysecret": "aliyun-oss-access-key-secret",
      "bucket": "aliyun-oss-bucket",
      "region": "aliyun-oss-region"
    }
  }
}
```

```
{
  "docker_registry": {
    "storage_driver": "gcs",
    "storage_settings": {
      "bucket": "gcs-storage-bucket"
    }
  }
}
```

For more details, see the Docker Registry docs.

2. Reconfigure the environment:

```
sudo hth-ctl reconfigure
```

Garbage Collection

Delete action provided by the TeamHub backend does not remove docker repositories permanently. Because Docker data is still persisted under the hood, you need to perform an additional step.

Combo

```
sudo hth-ctl docker-registry-garbage-collect
```

Cluster and HA

For cluster setups, the garbage collection process must respect other instances.

1. Stop the docker registry service for all instances except the one from which the command is run:

```
sudo hth-ctl stop docker_registry
```

2. Run the command on the chosen instance:

```
sudo hth-ctl docker-registry-garbage-collect
```

3. Start the docker registry services for other instances:

```
sudo hth-ctl start docker_registry
```

Code Search

> **Note**
> Helix TeamHub supports code searching for Mercurial, Git, and Helix Git repositories.

Helix TeamHub supports configuring Elasticsearch for searching code in repositories. Elasticsearch is not included as part of Helix TeamHub packages. It can be installed in multiple ways depending on the needs: inside a Helix TeamHub instance (only recommended for small instances), as a separate machine, or as a cluster of separate machines. See the official documentation for installing Elasticsearch and for configuring production deployment.

The following Elasticsearch versions are supported:

- TeamHub 2018.2 and earlier: Elasticsearch 5.x
- TeamHub 2019.1 and later: Elasticsearch 6.x

For information on upgrading from an earlier TeamHub version, see the release notes.

Prerequisites

The first step is to estimate the index size based on the storage size of the repositories that support code search. Using this figure with future growth in mind, you can decide the type of Elasticsearch installation and number of shards to create. See capacity planning for more details.

> **Note**
> Changing the number of shards requires recreating the index and reindexing all the repositories.

Configuration

Once you have a running instance of Elasticsearch, you can configure Helix TeamHub to use it.

Combo

Append the following minimum configuration to **`/var/opt/hth/shared/hth.json`** and **make sure to add the settings under the existing `backend` key**.

```
{
  "backend": {
    "search_engine": "elasticsearch",
    "es_hosts": [
        {
            "host": "your.elastic.host",
            "port": 9200
        }
    ]
  }
}
```

See **`backend`** section in configuration flags for additional settings:

```
{
  "backend": {
    "search_engine": "elasticsearch",
    "es_index_prefix": "optional",
    "es_number_of_shards": 5,
    "es_number_of_replicas": 1,
    "es_hosts": [
        {
            "host": "your.elastic.host",
            "port": 9200,
            "user": "username",
            "password": "password",
            "scheme": "https"
        }
    ]
  }
}
```

Apply the changes by reconfiguring Helix TeamHub:

```
sudo hth-ctl reconfigure
```

Create the Elasticsearch index as a **`hth`** user. Make sure to load a new session for **`hth`** user after running reconfigure:

```
sudo su - hth
cd /opt/hth/application/backend/current/
bundle exec rake hth:search:create_index
```

Cluster and HA

Follow the same steps as in **Combo** section above in one of the web nodes. After creating the index, reconfigure rest of the web nodes before enabling search in Helix TeamHub Client.

```
sudo hth-ctl reconfigure
```

Enabling search

After configuration, you can enable the search inside a company, as follows:

1. Log in to the Helix TeamHub client as a company admin.
2. To access the company settings, do one of the following:
   - In the **My Dashboard** view, click the gear icon ⚙ next to the company name.
   - In any view, click the user name in the site header and select **Company settings**.
3. In the **Company settings** form, on the **Features** tab, enable **Code Search**.
4. Save your settings.

5. Wait for the indexing to complete. The initial indexing can take a long time, depending on the size of the repositories. You can see the indexing status in the company overview.

For information on other features that company admins can configure, see Feature settings in the *Helix TeamHub User Guide*.

Index management

After changing index configurations later on, such as number shards, the index can be recreated as a **`hth`** user. Make sure to load a new session for **`hth`** user after running reconfigure:

```
sudo su - hth
cd /opt/hth/application/backend/current/
```

Create, delete, or recreate the index:

```
bundle exec rake hth:search:create_index

bundle exec rake hth:search:delete_index

bundle exec rake hth:search:recreate_index
```

Refresh all repositories for a company (by short_name) since last indexing, or perform full reindexing (can take a long time):

```
bundle exec rake hth:search:reindex_company[company]

bundle exec rake hth:search:reindex_company[company,full]
```

Refresh all repositories for all companies since last indexing, or perform full reindexing (can take a long time):

```
bundle exec rake hth:search:reindex_instance

bundle exec rake hth:search:reindex_instance[full]
```

Refresh a repository (by short_names) since last indexing, or perform full reindexing:

```
bundle exec rake hth:search:reindex_repository[company,project,repository]

bundle exec rake hth:search:reindex_repository
[company,project,repository,full]
```

Advanced Configuration

This section provides the following information:

Helix TeamHub Configuration

The `/var/opt/hth/shared/hth.json` configuration file and some of the flags it provides should be familiar by now. This is where Helix TeamHub configuration settings are kept, including settings manipulated through Helix TeamHub Admin UI from the Helix TeamHub Bootstrap phase. This configuration file is also part of Helix TeamHub Backups, if enabled.

Every time Helix TeamHub is reconfigured, the configuration file is read and the configuration is applied to all of Helix TeamHub services. Some of the configuration flags dictate what mode of deployment Helix TeamHub is running, where others simply override default Helix TeamHub settings.

Format

The configuration file is formatted as JSON. It is important to keep the correct format of the file, otherwise the Helix TeamHub reconfiguration will not work. JSONLint is a trusted open-source JSON linter option to help verify the syntax of the JSON contents.

> **Note**
> Remember that duplicate keys override the previously defined keys.

To see the current configuration applied to the server, run the following command:

```
sudo hth-ctl show-config
```

Overriding Defaults

Helix TeamHub comes with many sensible defaults for both service and application behavior. If the default configuration needs to be adjusted, use the dictionary provided below as a reference.

Each configuration flag has a section, for example `app`, `nginx`, etc. These sections separate settings into logical categories. To use the dictionary, simply merge the section to the existing Helix TeamHub configuration file at `/var/opt/hth/shared/hth.json` with a required key and value.

Whenever changing any of the configurations, pay attention to the type of the key and where available refer to the linked documentation. Also, remember that configuration is not applied until the `sudo hth-ctl reconfigure` command has been run.

Section: apache

This section groups Helix TeamHub Apache-related settings.

Key	Type	Default	Description
`timeout`	Integer	`120`	Defines the length of time Apache httpd will wait for I/O

Section: app

This section groups instance and general application related settings.

Key	Type	Default	Description
`backups_email`	String	`email`	Allows overriding receiver of backup related emails.
`default_company`	String		Default company short name to use with login
`email`	String	`support@FQDN`	Email of the sender of all outgoing emails and links to Support team
`hostname`	String	`FQDN`	Helix TeamHub application hostname
`http_proxy`	String		Defines HTTP proxy to use with external services like hooks. Provide absolute url including possible credentials: `http://user:password@proxy.com:8008`.
`is_cluster`	Boolean	`false`	Defines whether Helix TeamHub runs in Cluster or HA mode
`is_master`	Boolean	`false`	Defines whether Helix TeamHub runs in master mode with Subversion replication

Key	Type	Default	Description
`is_slave`	Boolean	`false`	Defines whether Helix TeamHub runs in slave mode with Subversion replication
`is_ssl`	Boolean	`false`	Defines whether Helix TeamHub enforces SSL
`notifications_email`	String	`email`	Allows overriding sender of notification related emails.
`registrations_email`	String	`email`	Allows overriding sender of registration related emails.
`ssh_port`	Integer	`22`	Defines SSH port for Git and Mercurial clone urls when the instance is using non-standard SSH port.

Section: audit

This section groups Helix TeamHub audit logging related settings.

Key	Type	Default	Description
`logrotate_frequency`	String	`daily`	Frequency of logrotate rotation
`logrotate_rotate`	Integer	`90`	Number of logrotate files to keep
`logrotate_size`	Integer		Size of logrotate rotation. Does not rotate by size by default

Section: backend

This section groups Helix TeamHub backend (APIs, Helix TeamHub Admin) application-related settings.

Key	Type	Default	Description
`auth_method`	String	`builtin`	Defines Helix TeamHub Authentication type. Allowed: `builtin`, `ldap`, or `both`

Key	Type	Default	Description
`backup_s3`	Boolean	`false`	Defines whether asset backups need to be taken offline to Amazon S3. Requires backups section configuration.
`backups`	Boolean	`false`	Defines whether Helix TeamHub asset backups are enabled
`command_timeout`	Integer	`60`	Timeout for command execution, in seconds.
`company_disk_usage_calculator_queue_size`	Integer	`1`	Number of workers performing company disk usage calculation jobs.
`db_host`	String	`localhost`	MongoDB hostname for Cluster or HA setup
`db_password`	String		MongoDB password
`db_pool_size`	Integer	`10`	MongoDB connection pool size
`db_port`	Integer	`4002`	MongoDB port
`db_username`	String		MongoDB username
`diff_file_max_bytes`	Integer	`51200`	Maximum number of bytes for a file in a diff
`diff_max_bytes`	Integer	`1024000`	Maximum number of bytes for a diff output
`diff_max_files`	Integer	`150`	Maximum number of files in a diff
`diff_max_lines`	Integer	`50000`	Maximum number of lines in a diff
`diff_process_max_bytes`	Integer	`2048000`	Maximum number of bytes to process for a diff
`diff_timeout`	Integer	`5`	Timeout in seconds for generating a diff
`es_hosts`	Array	`nil`	Array of Elasticsearch host hashes, with supported keys: `host`, `port`, `scheme`, `user`, `password`.
`es_index_prefix`	String	`nil`	Defines the optional index name prefix for Elasticsearch indices.

Key	Type	Default	Description
`es_number_of_replicas`	Integer	`1`	Defines the number of replicas for Elasticsearch indices.
`es_number_of_shards`	Integer	`5`	Defines the number of shards to use with Elasticsearch indices.
`es_ssl_verify`	Boolean	`true`	Defines whether to validate Elasticsearch host certificate.
`events_queue_size`	Integer	`2`	Number of workers performing event jobs.
`failed_login_interval`	Integer	`2`	Minimum time between failed login attempts
`failed_login_limit`	Integer	`6`	Limit of failed login attempts in specified time frame
`failed_login_period`	Integer	`60`	Time frame for `failed_login_limit`
`hooks_queue_size`	Integer	`2`	Number of workers performing repository event jobs.
`index_queue_size`		`1`	Number of workers performing code search indexing jobs
`ldap_collaborators_base`	String		LDAP search base for collaborators
`ldap_collaborators_enabled`	Boolean	`false`	LDAP authentication for collaborators
`ldap_collaborators_filter`	String		LDAP search filter used when finding collaborators
`ldap_domain_base`	String		LDAP search base for users
`ldap_email`	String		LDAP account email field mapped to Helix TeamHub email
`ldap_encryption`	String	`plain`	LDAP encryption. Allowed: `plain`, `start_tls`, `simple_tls`

Key	Type	Default	Description
`ldap_filter`	String		LDAP search filter used when finding users
`ldap_first_name`	String		LDAP account first name field
`ldap_groups`	String		LDAP field defining users groups
`ldap_groups_base`	String		LDAP groups search base
`ldap_groups_from_user`	Boolean	`false`	LDAP users contain group information
`ldap_host`	String		LDAP hostname
`ldap_interface_max_connections`	Integer	`10000`	Maximum connections for a child process until it is replaced with a new fork in the LDAP interface
`ldap_interface_max_idle`	Integer	`10`	Maximum idle time, in seconds, for a child process after stopping serving requests until it is replaced with a new fork in the LDAP interface
`ldap_interface_max_servers`	Integer	`32`	Maximum number of forked child processes in the LDAP interface
`ldap_interface_min_servers`	Integer	`4`	Minimum number of forked child processes in the LDAP interface
`ldap_interface_timelimit`	Integer	`30`	Maximum server-side time limit, in seconds, for a request in the LDAP interface
`ldap_last_name`	String		LDAP account last name field
`ldap_password`	String		LDAP search password
`ldap_port`	String		LDAP port
`ldap_short_name`	String		LDAP account ID field mapped to Helix TeamHub login
`ldap_source`	String		LDAP sync identifier

Key	Type	Default	Description
`ldap_ssl_verify`	Boolean	`false`	Defines whether to validate external LDAP host certificate
`ldap_user`	String		LDAP search username
`license_expire_notify`	String	`30,14,7,3`	Defines the intervals (number of days) before license expiration to notify instance admins through email.
`merge_queue_size`	Integer	`2`	Number of workers performing merge jobs.
`merge_timeout`	Integer	`120`	Timeout for code review merge in seconds.
`password_expire_count`	Integer	`0`	Defines the number of old passwords that cannot be used again. Value of `0` allows reusing old passwords.
`password_expire_days`	Integer	`0`	Defines the maximum number of days a password can be used before it expires. Value of `0` means that passwords never expire.
`password_expire_notify`	Integer	`7`	Defines the number of days before password expiration to notify accounts.
`password_validation_entropy`	Integer	`-1`	Defines the desired password entropy level related to possible `email`, `short_name`, `first_name`, `last_name` field values using Levenshtein algorithm. Value `0` means extact matching (checks if password is the same like a value of one of aforementioned fields). Value greater than `0` defines a threshold of similarity password must fulfil. Value less than `0` turns off this validation (default).

Key	Type	Default	Description
password_validation_format	String	/*./	Defines the password format requirements for account password validation. For example, to ensure that password contains at least 1 uppercase letter, 1 lowercase letter and 1 digit, you can use the following pattern: /(?=.*[A-Z])(?=.*[a-z])(?=.*[0-9]).*/
password_validation_range	String	8..100	Defines the minimum and maximum length for account password validation.
pilsner_timeout	Integer	55	Pilsner request timeout in seconds.
redis_host	String	localhost	Redis hostname for Cluster or HA setup
redis_password	String		Redis password
redis_port	Integer	6379	Redis port
repository_gc_queue_size	Integer	1	Number of workers performing garbage collection jobs.
search_engine	String	nil	Defines the search engine to use with Code Search. Supported values: nil and elasticsearch.
slave_syncer_enabled	Boolean	false	Defines whether slave_syncer is enabled for slave site with Subversion replication.

Section: backups

This section groups Helix TeamHub Backups related settings.

Key	Type	Default	Description
keep	Integer	30	How many backups to keep before oldest backup gets removed (Archival method only)
s3_access_key	String		Amazon S3 access key for offline backups

Key	Type	Default	Description
`s3_bucket`	String		Amazon S3 bucket name for offline backups
`s3_key_id`	String		Amazon S3 key ID for offline backups
`s3_region`	String		Amazon S3 region for offline backups

Section: docker_registry

This section groups Helix TeamHub"Docker Registry" on page 107 related settings.

Key	Type	Default	Description
`backups`	Boolean	`false`	Whether Docker backups are enabled
`backups_keep`	Integer	`5`	How many backups to keep before oldest backup gets removed
`log_level`	String	`warn`	Log level for Docker service
`storage_driver`	String	`filesystem`	Docker storage driver to use, see "Docker Registry" on page 107
`storage_settings`	Object		Docker storage driver options, see Docker Registry storage driver on the Docker website

Section: gconn

This section groups the Git Connector (Gconn) settings.

Key	Type	Default	Description
`helix_user`	string	nil	Helix server user of Git Connector
`host`	string	nil	Hostname of the server Git Connector is installed on
`https_enabled`	Boolean	`false`	Defines whether https is enabled or disabled for Git Connector
`https_port`	Integer	nil	Https protocol port.
`ssh_enabled`	Boolean	`false`	Defines whether SSH is enabled or disabled for Git Connector

Key	Type	Default	Description
`ssh_port`	integer	nil	SSH protocol port
`user`	string	nil	OS user of the Git Connector

Section: helix

This section groups Helix server connection settings.

Key	Type	Default	Description
`charset`	string	`utf8`	Character set encoding on the Helix server. For example, `utf8`or `none`.
`p4port`	string	nil	The hostname or IP address and port for the Helix server, in the form of: `host:port`
`password`	string	nil	Password or ticket for the Helix server super user
`sync_interval`	string	`*/5 * * * *`	Interval to sync with the Helix server. Set in Cron format, default is every 5 minutes
`user`	string	nil	An existing Helix server user with super level privileges. This user must have unlimited ticket timeout

Section: logging

This section groups Helix TeamHub logging related settings.

Key	Type	Default	Description
`logrotate_frequency`	String	`daily`	Frequency of logrotate rotation
`logrotate_rotate`	Integer	`30`	Number of logrotate files to keep
`logrotate_size`	Integer		Size of logrotate rotation. Does not rotate by size by default.
`svlogd_num`	Integer	`30`	Number of SV log files to keep
`svlogd_size`	Integer	`209715200`	The maximum size when SV rotation should happen (200MB)
`svlogd_timeout`	Integer	`86400`	Number of seconds when SV rotation should happen (24 hours)

Section: mongodb

This section groups Helix TeamHub MongoDB database related settings, which are usually required in Cluster or HA deployment for tools such as Helix TeamHub Backups accessing MongoDB database.

Key	Type	Default	Description
`backup_s3`	Boolean	`false`	Defines whether MongoDB backups need to be taken offline to Amazon S3. Requires backups section configuration
`backups`	Boolean	`false`	Defines whether Helix TeamHub MongoDB backups are enabled
`keyfile`	String	`/var/opt/hth/shared/mongodb-keyfile`	Defines location for MongoDB keyfile with Subversion replication
`password`	String		MongoDB password
`port`	Integer	`4002`	MongoDB port
`replset`	String	`hth`	Defines replica set name for MongoDB with Subversion replication
`username`	String		MongoDB username

Section: nginx

This section groups Helix TeamHub Nginx related settings.

Key	Type	Default	Description
`enable_sslv3`	Boolean	`false`	Whether SSLv3 should be enabled, see Poodle vulnerability

Key	Type	Default	Description
`keepalive_timeout`	Integer	`65`	Number of seconds for keep-alive connection
`max_body_size`	String	`4G`	Max size of client request body
`proxy_read_timeout`	Integer	`120`	Number of seconds for reading a response from backend services
`proxy_send_timeout`	Integer	`120`	Number of seconds for sending a request to backend services
`server_names`	String	`_`	Server names Nginx will listen on
`ssl_ciphers`	String	See default nginx ciphers below [1]	Specifies enabled ciphers in the format understood by the OpenSSL library
`ssl_protocols`	String	`TLSv1 TLSv1.1 TLSv1.2 TLSv1.3`	SSL protocols to enable.
`worker_connections`	Integer	`1024`	Number of Nginx simultaneous worker connections
`worker_processes`	Integer	`2`	Number of Nginx worker processes to start

[1] Default nginx ciphers:

```
ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-
AES256-GCM-SHA384:ECDHE-ECDSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-
SHA256:DHE-DSS-AES128-GCM-SHA256:kEDH+AESGCM:ECDHE-RSA-AES128-
SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-RSA-AES128-SHA:ECDHE-ECDSA-AES128-
SHA:ECDHE-RSA-AES256-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-RSA-AES256-
SHA:ECDHE-ECDSA-AES256-SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES128-SHA:DHE-
DSS-AES128-SHA256:DHE-RSA-AES256-SHA256:DHE-DSS-AES256-SHA:DHE-RSA-AES256-
SHA:AES128-GCM-SHA256:AES256-GCM-SHA384:AES128-SHA256:AES256-
SHA256:AES128-SHA:AES256-SHA:AES:CAMELLIA:DES-CBC3-
SHA:!aNULL:!eNULL:!EXPORT:!DES:!RC4:!MD5:!PSK:!aECDH:!EDH-DSS-DES-CBC3-
SHA:!EDH-RSA-DES-CBC3-SHA:!KRB5-DES-CBC3-SHA
```

Section: opensshp

This section groups OpenSSH related settings.

Key	Type	Default	Description
`enable`	Boolean	`false`	Whether the bundled OpenSSH is used or not, see OpenSSH.

Section: pilsner

This section groups settings related to "Helix authentication" on page 59.

Key	Type	Default	Description
`helix_groups_exclude_regex`	String		Names of groups to be excluded from mirroring between Helix server and Helix TeamHub, specified as a comma-delimited list of Ruby regular expressions, for example: `^swarm-group$` The specified groups do not appear in the TeamHub UI. **Important** By default, TeamHub ignores all groups starting with **`HTH-`** and all legacy Perforce product groups in Helix server, such as Swarm groups.

Key	Type	Default	Description
`helix_groups_include_regex`	String		Names of groups to be included when mirroring between Helix server and Helix TeamHub, specified as a comma delimited list of Ruby regular expressions, for example: `^swarm-group$` The specified groups appear in the TeamHub UI.
`helix_timeout`	Integer	`50`	Helix server request timeout in seconds.
`helix_users_exclude_regex`	String		Names of users to be excluded from mirroring between Helix server and Helix TeamHub, specified as a comma delimited list of Ruby regular expressions, for example: `^(user1|user2)$` The specified users do not appear in the TeamHub UI.
`helix_users_from_groups_exclude_regex`	String		Names of groups to exclude users from when mirroring between Helix server and Helix TeamHub, specified as a comma delimited list of Ruby regular expressions, for example: `^perforce-group$` The users from the specified groups do not appear in the TeamHub UI.
`helix_users_from_groups_include_regex`	String		Names of groups to include users from when mirroring between Helix server and Helix TeamHub, specified as a comma delimited list of Ruby regular expressions, for example: `^perforce-group$` The users from the specified groups appear in the TeamHub UI.

Key	Type	Default	Description
`helix_users_include_regex`	String		Names of users to be included while mirroring between Helix server and Helix TeamHub, specified as a comma delimited list of Ruby regular expressions, for example: `^(user1|user2)$` The specified users appear in the TeamHub UI.
`host`	string	`localhost`	Pilsner service hostname.
`port`	integer	`9292`	Pilsner service port.

Section: postfix

This section groups Helix TeamHub local Postfix MTA mailing settings.

Key	Type	Default	Description
`masquerade_domain`	String	Domain of the email key	Masquerade domain
`message_size_limit`	Integer	`20000000`	Max size of the message in bytes
`password`	String		Password for SASL authentication
`relay_host`	String		Relay hostname
`relay_port`	Integer	`25`	Relay port
`sasl_auth_enable`	Boolean	`false`	Whether SASL authentication is enabled
`tls_auth_enable`	Boolean	`false`	Whether TLS is used
`tls_ca_crt_bundle`	String		TLS CA certificates file
`user_name`	String		Username for SASL authentication

Section: puma_pilsner

This section groups Helix TeamHub Puma Pilsner server related settings.

Key	Type	Default	Description
`max_memory`	Integer	`1000`	Maximum total memory (MB) for Puma Pilsner when multiple workers are used
`max_threads`	Integer	`4`	Maximum size of worker's thread pool
`min_threads`	Integer	`0`	Minimum size of worker's thread pool
`worker_processes`	Integer	`2`	Number of Puma Pilsner worker processes to start

Section: redis

This section groups Redis related settings.

Key	Type	Default	Description
`password`	string		Redis server password
`port`	Integer	`6379`	Redis server port

Section: replication

This section groups Subversion replication related settings.

Key	Type	Default	Description
`password`	String		Password of a company admin account to be used with replication on master site
`queue_size`	Integer	`2`	Number of workers performing replication jobs on master site
`timeout`	Integer	`60`	Number of seconds to wait after replication failure on master site
`username`	String		Username of a company admin account to be used with replication on master site

Section: repos

This section groups Helix TeamHub repositories related settings.

Key	Type	Default	Description
`backup_s3`	Boolean	`false`	Defines whether repository backups need to be taken offline to Amazon S3. Requires backups section configuration.
`backups`	Boolean	`false`	Defines whether Helix TeamHub repository backups are enabled

Section: unicorn_backend

This section groups Helix TeamHub backend (APIs, Helix TeamHub Admin) Unicorn server related settings.

Key	Type	Default	Description
`backlog_socket`	Integer	`64`	Unicorn socket backlog size
`worker_processes`	Integer	`4`	Number of Unicorn worker processes to start
`worker_timeout`	Integer	`60`	Number of seconds Unicorn worker times out

System Overrides

Besides the directories listed in File system hierarchy, Helix TeamHub touches a number of system wide configuration files:

General

- `/etc/group` - Creates `hth` group with default GID 21212
- `/etc/passwd` - Creates `hth` user with default UID 21212

Init Services

- `/etc/inittab` - Injects a line to start `/opt/hth/embedded/bin/runsvdir-start` on boot
- `/etc/init/hth-runsvdir.conf` - Helix TeamHub Runit configuration

SSH Operations

- `/etc/ssh/ssh_host_*` - Copies SSH host keys from `/var/opt/hth/shared/ssh` to support Helix TeamHub HA setup keys synchronization
- `/usr/sbin/sshd` - Symlink to Helix TeamHub OpenSSH binary at `/opt/hth/embedded/sbin/sshd` when bundled OpenSSH is used.
- `/etc/ssh/sshd_config` - Injects `AuthorizedKeysCommand` to use Helix TeamHub key based authorization when bundled OpenSSH is used.

For more information, see "OpenSSH and repository SSH access" on page 27.

Sudo Rights

- `/etc/sudoers` - Injects a line to include `hth` file from `/etc/sudoers.d`
- `/etc/sudoers.d/hth` - Creates Helix TeamHub sudo entries

Common Binaries

- `/usr/bin/git` - Symlink to Helix TeamHub Git binary at `/opt/hth/embedded/bin/git`
- `/usr/bin/hg` - Symlink to Helix TeamHub Hg binary at `/opt/hth/embedded/bin/hg`
- `/usr/bin/svn` - Symlink to Helix TeamHub Subversion binary at `/opt/hth/embedded/bin/svn`
- `/usr/bin/hth-ctl` - Symlink to Helix TeamHub Control tool at `/opt/hth/bin/hth-ctl`

Mailing Configuration

- `/etc/postfix/generic`
- `/etc/postfix/main.cf`
- `/etc/postfix/sasl_passwd`
- `/etc/aliases`

Resources

This section provides the following information:

How to Setup HAProxy

HAProxy is a reliable, high performance TCP/HTTP Load Balancer, and it works nicely with Helix TeamHub HA setup.

Preparation

Make sure `/etc/ssh` SSH host keys are synchronized (see Synchronizing SSH host keys section) across all cluster nodes, otherwise a random "SSH RSA host key has been changed" error will occur.

Follow these steps to install and configure HAProxy according to the host operating system:

RHEL and CentOS

Install HAProxy:

```
cd /tmp
yum install wget openssl-devel pcre-devel make gcc wget
wget http://www.haproxy.org/download/1.5/src/haproxy-1.5.3.tar.gz
tar -zxvf haproxy-1.5.3.tar.gz && cd haproxy-1.5.3
make TARGET=linux2628 CPU=x86_64 USE_OPENSSL=1 USE_ZLIB=1 USE_PCRE=1
make install
```

Create the *init* script:

```
ln -sf /usr/local/sbin/haproxy /usr/sbin/haproxy
cp /tmp/haproxy-1.5.3/examples/haproxy.init /etc/init.d/haproxy
chmod  755 /etc/init.d/haproxy
```

Add default configuration and user:

```
mkdir /etc/haproxy
cp /tmp/haproxy-1.5.3/examples/examples.cfg /etc/haproxy/haproxy.cfg
mkdir /var/lib/haproxy
touch /var/lib/haproxy/stats
useradd haproxy
```

Start the service and enable on boot:

```
service haproxy check
service haproxy start
chkconfig haproxy on
```

Sample configuration

Below is the example configuration to use with Helix TeamHub with two Web application servers. Replace the *VALUES* with the required data.

It is recommended to use at minimum 2048-bit Diffie-Hellman group. You may generate DH parameter file using OpenSSL (**`openssl dhparam -out dhparams.pem 2048`**) and append it to your certificate file.

```
global
    log 127.0.0.1 local0 notice
    maxconn 2000
    user haproxy
    group haproxy
    ssl-default-bind-ciphers ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-
AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-GCM-
SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-DSS-AES128-GCM-
SHA256:kEDH+AESGCM:ECDHE-RSA-AES128-SHA256:ECDHE-ECDSA-AES128-
SHA256:ECDHE-RSA-AES128-SHA:ECDHE-ECDSA-AES128-SHA:ECDHE-RSA-AES256-
SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-RSA-AES256-SHA:ECDHE-ECDSA-AES256-
SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES128-SHA:DHE-DSS-AES128-SHA256:DHE-
RSA-AES256-SHA256:DHE-DSS-AES256-SHA:DHE-RSA-AES256-SHA:AES128-GCM-
SHA256:AES256-GCM-SHA384:AES128-SHA256:AES256-SHA256:AES128-SHA:AES256-
SHA:AES:CAMELLIA:DES-CBC3-
SHA:!aNULL:!eNULL:!EXPORT:!DES:!RC4:!MD5:!PSK:!aECDH:!EDH-DSS-DES-CBC3-
SHA:!EDH-RSA-DES-CBC3-SHA:!KRB5-DES-CBC3-SHA
defaults
    log     global
    option  dontlognull
    retries 3
    timeout connect  5000
    timeout client  10000
    timeout server  10000
# SSH connections to Helix TeamHub
frontend hth-sshd
    bind *:22
    mode tcp
    default_backend hth-scm
# HTTP -> HTTPS redirection
frontend hth-http
    bind *:80
    mode http
    redirect scheme https code 301 if !{ ssl_fc }
# HTTPS connections to Helix TeamHub
frontend hth-https
    bind *:443 ssl crt __PATH_TO_CERTIFICATE_PEM_FILE__ no-sslv3
    mode http
    option http-server-close
    option forwardfor
    reqadd X-Forwarded-Proto:\ https
    default_backend hth-web
backend hth-scm
    mode tcp
    option tcplog
    balance roundrobin
    server scm1 __IP_ADDRESS_OF_FIRST_NODE__:22 check
    server scm2 __IP_ADDRESS_OF_SECOND_NODE__:22 check
backend hth-web
    mode http
    option httplog
    stats enable
    stats uri /haproxy?stats
    stats realm Strictly\ Private
    stats auth __WEBADMIN_USERNAME__:__WEBADMIN_PASSWORD__
    balance roundrobin
    cookie HTHSTICKY insert indirect nocache
    server web1 __IP_ADDRESS_OF_FIRST_NODE__:80 check cookie web1
    server web2 __IP_ADDRESS_OF_SECOND_NODE__:80 check cookie web2
```

Helix TeamHub LDAP Sync Application

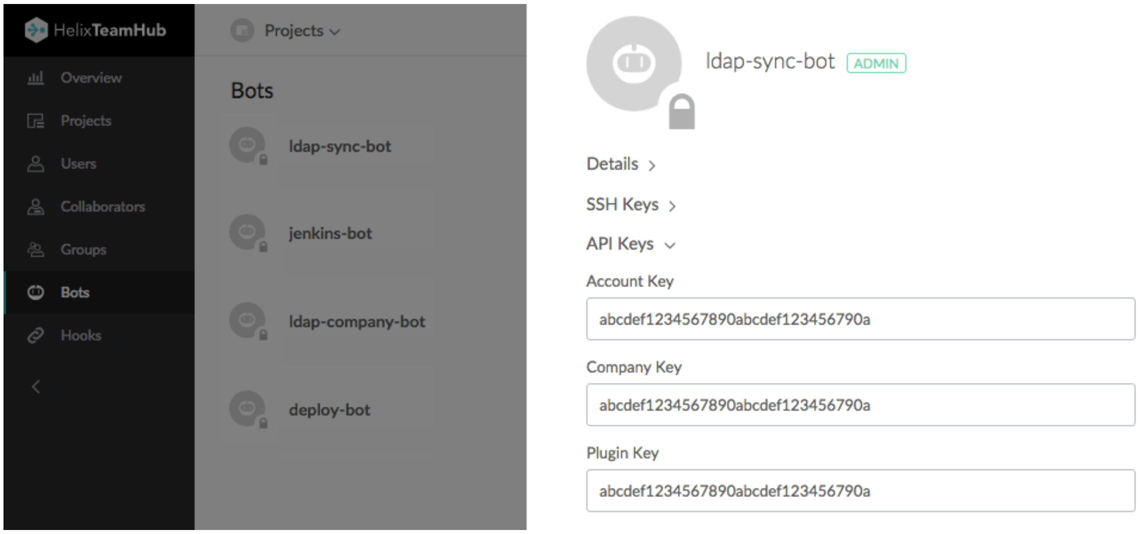
Integrating the corporate LDAP directory to Helix TeamHub is straightforward. Follow the steps below to setup synchronization. On-premises installations of Helix TeamHub can also be configured to use LDAP Authentication. In such case the LDAP Sync application can still be used to keep existing Helix TeamHub accounts up to date.

Requirements

- Java runtime (1.6+)
- Read access to the source
- The source must provide at least the following user attributes: (**`first_name`**, **`last_name`**, **`username`**, **`email`**)

Create Company Admin Bot

Login to Helix TeamHub App with a Company Admin account (On-premises users: login to your instance). Click the company name on the top navigation and select Bots from the left menu. Create a new Company Admin bot and take a note of the API keys, they'll be used later when configuring synchronization.



Account Key

abcdef1234567890abcdef123456790a

Company Key

abcdef1234567890abcdef123456790a

Plugin Key

abcdef1234567890abcdef123456790a

Setup and Configure LDAP Sync

Download Helix TeamHub LDAP Sync application from the LDAP Sync app download page and extract it. The application is configured using YAML file. Copy one of the example config files and use it as a base for configuration. Use the `example_ad.yml` if the source is Active Directory or `example_ldap.yml` if the source is some other OpenLDAP compliant server.

Configure General and Helix TeamHub API Settings

Uncomment and set a unique name for the `source`. The `source` is used to identify which users in Helix TeamHub are synchronized from this LDAP source. Make sure to use the same value for the `source` that was used in Helix TeamHub Admin authentication preferences if using LDAP Authentication. For synchronization strategy, choose either to sync all the users from LDAP to Helix TeamHub, or only keep existing Helix TeamHub users in sync. In both cases deactivated or deleted LDAP users will also be deleted from Helix TeamHub. The `keep_in_sync` strategy is recommended for LDAP Authentication.

Please note: Using `sync_all` strategy will sync all the users found from LDAP to Helix TeamHub and new users will receive a registration/welcome email.

Set value for `company_key` and `account_key` previously noted. If using Helix TeamHub On-premises, set the `api_url` to point to the Helix TeamHub instance API. You may enable certificate verification with Helix TeamHub API requests by setting a path to the root certificate of the certificate used in Helix TeamHub for the `api_server_certificate` attribute. The path can be either relative to where hth-ldapsync.jar is executed or an absolute path.

Configure your LDAP settings

Set the LDAP `host`, `port`, `auth_username` and `auth_password` for a user with read access for the source, and choose the encryption method for LDAP connection. Then set the LDAP `base` tree where to sync the users from. Users can be ignored from synchronization by adding the `short_name` to the `ignored_users` list.

Starting from Helix TeamHub version 2018.1 and LDAP Sync version 2.1.0, it is now possible to enable nested group support for Active Directory by enabling the `request_user_groups` setting and by using `member:1.2.840.113556.1.4.1941:` as a value for the `ldap_groups` attribute.

Finally configure the attribute mappings between the LDAP schema and Helix TeamHub.

Configure Groups (optional)

Starting from Helix TeamHub version 2.7.1 and LDAP Sync version 1.3.0, it is now possible to synchronize groups from the base directory to Helix TeamHub. Define which groups are created by using `group_base` and `group_query_filters`.

Attribute mappings between the directory and Helix TeamHub can be configured using `ldap_group_attributes`. The default mapping will work for most users, but revise the value of the `source` attribute. The `source` defines whether the LDAP groups are linked to Helix TeamHub groups by common name `cn` or distinguished name (`dn`).

Group synchronization can be enabled by using the `group-sync` switch:

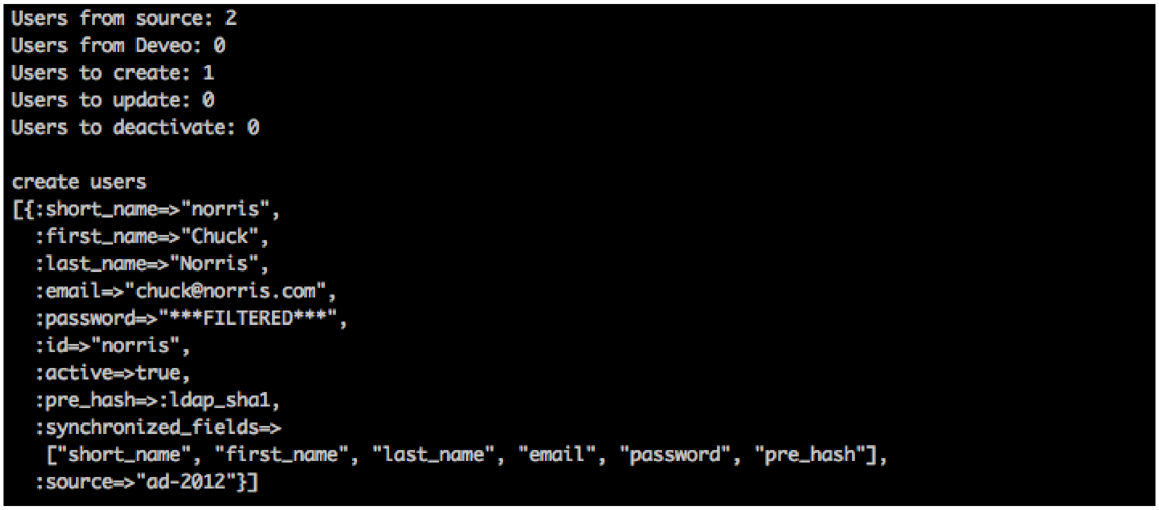
```
java -jar hth-ldapsync.jar --config myconfig.yml --group-sync
```

Test Configurations

Try out the configurations by running the LDAP Sync application and giving it the configuration file as a parameter. By default no modifications are made, shown instead are details of what the synchronization would do. For example:

```
java -jar hth-ldapsync.jar --config myconfig.yml
```

If there are already users in Helix TeamHub that need to be synchronized from the source, use the `--force-sync` switch with the first run. It will map the existing Helix TeamHub users to the source and update them instead of creating new users.



```
Users from source: 2
Users from Deveo: 0
Users to create: 1
Users to update: 0
Users to deactivate: 0

create users
[{:short_name=>"norris",
  :first_name=>"Chuck",
  :last_name=>"Norris",
  :email=>"chuck@norris.com",
  :password=>"***FILTERED***",
  :id=>"norris",
  :active=>true,
  :pre_hash=>:ldap_sha1,
  :synchronized_fields=>
   ["short_name", "first_name", "last_name", "email", "password", "pre_hash"],
  :source=>"ad-2012"}]
```

Run the LDAP Sync

Once the configuration is ready, run the actual synchronization by adding the `--apply` switch. This will synchronize users to Helix TeamHub.

```
java -jar hth-ldapsync.jar --config config.yml --apply
```

The LDAP Sync application can also be setup to run at intervals like once per hour, by using a scheduler.