## Contents

- Optimizing the PostgreSQL database  7-3
- Securing the Polarion activation application  7-4
- Change the default system administrator password  7-5
- Changing the Password for SVN User ‘polarion’  7-6
- Enter an error reporting email  7-6

### Configure OLE object support and attachment preview generator

- Overview  8-1
- Prerequisites  8-1
- Install OpenSSH  8-2
- Configure OpenSSH  8-3
- Set up SMB shares and configure the client  8-5
- Configure the client  8-5
- Set up the scripts  8-8
- Configure Polarion  8-10

### Licensing and activation

- Overview  9-1
- Using different license types  9-1
- Assigning named and concurrent users  9-1
- License usage log file  9-2

### Subversion optimization  10-1

### Multiple repository setup  11-1

### Accessing the Polarion Portal  12-1

### LDAP authorization  13-1

### Next steps after installation  14-1

### Removing Polarion  15-1

### Technical support  16-1

### Appendix

- Default users and groups  17-1
- Enabling email notifications  17-1
- Manual installation on Linux  17-1
- System requirements and recommendations  17-2
- Required third-party software components  17-2
- Import a certificate to the Java Keystore  17-7
1. About this guide

Overview

Welcome and thanks for using Polarion.

This guide covers installation information and procedures for creating a production installation for all Polarion products based on the Polarion Application Lifecycle Management Platform. The list of products covered by this guide currently includes:

- Polarion ALM™
- Polarion REQUIREMENTS™
- Polarion QA™

In general, the information is applicable to all the above products. Any product-specific differences will be explicitly noted. The information covers both new installations and, where applicable, updating of existing installations.

This guide applies to installation of the above Polarion products on supported Linux operating systems.

If you want to install a product on a Windows system, please see the Windows Installation guide.

Evaluation installations

If you are installing Polarion for evaluation purposes, we recommend the Polarion ALM™ Trial guide. It focuses on getting you up and running with an evaluation installation as quickly as possible, using the Evaluation install option of the Windows installer.

This is recommended for the initial stage of any evaluation.
Large-scale installations

If you need a large-scale server environment with multiple clustered servers and failover capabilities, multiple repositories and so forth, see Polarion ALM™ Enterprise Setup.

Topics covered there include:

• Requirements
• Installation use cases
• Configuring shared data
• Security options
• Using Resource Traceability in a cluster
# 2. System requirements and recommendations

## Server software

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Operating System</strong></td>
<td>SUSE Linux Enterprise Server 12 with service pack 4 or SUSE Linux Enterprise Server 15 with service pack 1, Red Hat Enterprise Linux 6.x or 7.x (7.2 or higher), CentOS 6.x or 7.x (7.2 or higher), Debian GNU/Linux 9.0, and Ubuntu Server 16.04 LTS or 18.04 LTS. Supported architectures are x86_64, or amd64. URW fonts must be installed in the operating system. For SUSE, use the ghostscript-fonts-std package from <a href="#">Ghost Script Fonts</a>.</td>
</tr>
<tr>
<td><strong>Java Runtime Environment</strong></td>
<td>OpenJDK 11 - See <a href="#">Install OpenJDK</a> Polarion only supports the 64 bit version of Java.</td>
</tr>
<tr>
<td><strong>Version Control System</strong></td>
<td>Subversion version 1.6.x, 1.7.x, 1.8.x, or 1.9.x: (Using the latest Subversion 1.9x is recommended.) If you are compiling Subversion yourself, compile using the --with-apxs or the --with-httpd option.</td>
</tr>
<tr>
<td><strong>Web Server</strong></td>
<td>Apache HTTPD server with mod_proxy_ajp and Subversion extension (WebDAV+SVN apache modules): <a href="http://httpd.apache.org/">httpd.apache.org</a> In general, the Polarion server should run with whatever Apache version is present on a Linux system provided it is at least the minimum required version (2.2), and mod_proxy_ajp and Subversion extension modules are also installed.</td>
</tr>
</tbody>
</table>
| **Database**              | **Minimum**: PostgreSQL version 8.4  
**Recommended**: PostgreSQL version 9.2 - 9.x (Versions 10+ are not yet supported.)  
**Note**: PostgreSQL version 9.2 or higher is required to create the new rel_module_workitem database table. The table contains information about Work Items both contained or referenced within a document. |
## Server hardware

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RAM</td>
<td>• Minimum: 4 GB (gigabytes) for production installation.</td>
</tr>
<tr>
<td></td>
<td>• Recommended: 8 GB or more.</td>
</tr>
<tr>
<td></td>
<td>Tip: Set Polarion's memory allocation to 50% of your server's total RAM.</td>
</tr>
<tr>
<td>Disk Storage Space</td>
<td>• Minimum: 10 GB.</td>
</tr>
<tr>
<td></td>
<td>• Recommended: 40 GB or more.</td>
</tr>
<tr>
<td></td>
<td>There is no hard and fast rule for disk storage space. The actual amount you</td>
</tr>
<tr>
<td></td>
<td>require depends on the number and size of projects managed with Polarion.</td>
</tr>
<tr>
<td></td>
<td>The more projects, and the larger they are, the more disk storage you require.</td>
</tr>
</tbody>
</table>

## Client software

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operating System</td>
<td>Any operating system that can run the supported web browsers with support for the Flash plugin (see below).</td>
</tr>
<tr>
<td>Web Browser</td>
<td>All you need to use Polarion is a web browser. The most current list of supported browsers and versions is provided in the Release Notes section of the README.html file, delivered in all download distributions of all Polarion products.</td>
</tr>
<tr>
<td>Adobe Flash</td>
<td>The Polarion web portal displays Live Plan chart data using Adobe Flash. To use this particular functionality, the client computer must have Adobe Flash Player installed. You can download it free at <a href="http://www.adobe.com/products/flashplayer/">http://www.adobe.com/products/flashplayer/</a>.</td>
</tr>
</tbody>
</table>
# Client hardware

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RAM</td>
<td>Minimum: 2 GB (4 GB recommended)</td>
</tr>
<tr>
<td>Display Resolution</td>
<td>Minimum: 1280 x 800 pixels</td>
</tr>
<tr>
<td>Server Connection</td>
<td>Not less than 1 Mbit/s</td>
</tr>
</tbody>
</table>
2. System requirements and recommendations
3. Additional recommendations

Use of anti-virus (AV) software

When running on server OS platforms that run instances of Polarion server, anti-virus (AV) software intercepts I/O requests to Polarion data structures in order to scan the data looking for virus signatures. This inserts latency (time delay) into all underlying file system read and write operations, which has the potential to directly impact Polarion Server performance, potentially in the magnitude of hundreds of percent.

Also, the AV software may detect false positives in Polarion data structures, which can result in data corruption when the AV software attempts to either re-write the data, or worse, quarantines data files. Any of the various methods used by AV software to deal with false positives could potentially result in data corruption.

Best practice guidance is to use caution when implementing AV software products on a server that hosts Polarion ALM, as it can impose performance and stability issues that may lead to poor response times or data corruption. Where feasible, not running AV software on well protected and/or network isolated server platforms assures that there will be no impact on Polarion server operation. Where AV software must be running, then at minimum it is strongly suggested to exclude the underlying file system supporting a Polarion server’s Subversion repository and PostgreSQL database from real-time checking and/or dynamic scanning.

Appropriate security hygiene restricting outside access to the underlying file system supporting a Polarion server’s Subversion repository is recommended practice that effectively moderates the need for AV data protections. Assuring that attachments to Polarion data structures (work Items, etc.) are only allowed from sources that are subject to AV data protections is an equally prudent security measure that effectively moderates need for AV data protections for Subversion data.

Warning:

Siemens disclaims liability for corruption of any Polarion data structure that is caused by running AV software on platforms supporting a Polarion server.
Libraries required for building the demo projects

The items described here are not critical for running and evaluating Polarion. However, the components described are needed to be able to fully utilize Polarion's capabilities.

The distribution contains several demo projects. Each of them needs its particular set of 3rd party libraries to be correctly built and have the project reports generated. Any missing libraries are automatically downloaded from the internet during project processing, so you may need a connection to the internet when you first try building the demo projects or run reports for them.

Install fonts for exporting Highchart charts

The urw-fonts package must be installed on CentOS and SuSE.

Enable email notifications

The Polarion server can send email notifications in response to various events in the system such as build completions and new Work Items. It can also notify users about external changes.

1. You can enable email notifications before starting the Polarion server by setting the host name in the announce.smtp.host property in the polarion.properties file located in $POLARION_HOME/etc/.

2. In the same polarion.properties file, set the announce.smtp.user and announce.smtp.password properties to a valid email account on the SMTP host specified in announce.smtp.host. If your SMTP server doesn't listen on default port 25, change the port setting in announce.smtp.port. You may wish to create a special account on your SMTP host for use with Polarion notifications.

3. Provide a valid email address for each user in their user account. (Administration → User Management → Users.) This can be automated through user self-creation of accounts, or integration with LDAP. See Administrator’s Guide: Managing Users and Permissions in Polarion’s Help.

When the configuration is correctly set up, the system sends notification emails about various events according to the notification targets configuration. For information on configuring email notifications, see Administrator’s Guide: Configuring Notifications in Polarion’s Help.

Note:

If a Work Item is modified outside of the Polarion portal, for example manually in the SVN, email notifications are sent as if the modification occurred in the portal.

See also, Appendix: Enabling Email Notifications
Enable support for Javadoc

The demo and your own projects can be configured to provide Javadoc reports.

Javadoc must also be enabled for the descriptors.xml file. Access it in the Repository browser: Repository/.polarion/reports/descriptors.xml. Refer to comments within the in the descriptors.xml for details on how to enable Javadoc.

Hide versions of installed components

It's a good security precaution to update Apache's configuration to hide the exact version numbers for installed components by default.

• Once the Polarion installation is complete, add the following settings to your Apache configuration:
  ServerSignature Off
  ServerTokens Prod

The Apache configuration file names will vary depending upon the distribution.
4. Linux installation and overview

Installation types

Automated installation

The entire installation process is driven automatically by an installation script. At most you have to answer a few questions, but you do not pre-install anything or do any other manual work in order to satisfy the prerequisites before you start the script. Automated installation is possible on most, but not all, of the supported Linux platforms.

Manual installation

You must pre-install the required infrastructure software before you attempt to install Polarion. You follow steps in the documentation as opposed to interactive instructions from a script. You may also need to write some shell scripts by hand or modify the manual_install.sh file before running it.

Semi-automated installation

The automated installation process script is not able to install all the prerequisites for a platform automatically, so it requests that you to install something manually before proceeding.

See supported platforms and installation types for the details on the distributions they apply to.

Automated installation

The automated installer scripts detect the operating system (OS). The OS version is not checked, however, so you must ensure that you have the minimal required version of your system as specified in Server Software Requirements.

If your OS is supported, the automated scripts will install the prerequisite third-party software for the specific supported Linux platform from existing package repositories on your system during the Polarion installation process. They will then install the Polarion platform itself. Ensure that you have set up the package repository defaults provided for your system.

If you have a different version of the OS from the supported distributions, it is probably still possible to perform an automatic installation. However, during the process you will need to manually install the required third-party software before you respond No to installation default dependencies.
Supported Linux versions for automated installation

If you have one of the supported Linux platforms, you will find installation on Linux easier, faster and smoother. Each supported platform comes with predefined configurations of the recommended version of third-party software. If the installer script finds that your operating system is not one of the supported variants, it will inform you and allow you to install Polarion manually.

The table below lists the Linux platforms that are currently fully supported. Installation support for other platforms may become available. Be sure to check the list of available installers on the web page where you download Polarion.

We highly recommend using stable repositories and installing tested versions of third-party software rather than installing newer, untested versions.

<table>
<thead>
<tr>
<th>Platform</th>
<th>Type</th>
<th>Default Repositories Necessary</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUSE</td>
<td>Automated</td>
<td>All default repositories including SDK.</td>
</tr>
<tr>
<td>RedHat</td>
<td>Semi-automated</td>
<td>Default repositories from installation. For more information</td>
</tr>
<tr>
<td></td>
<td></td>
<td>about mirrors and repositories see: <a href="http://www.centos.org/download/mirrors/">http://www.centos.org/download/mirrors/</a></td>
</tr>
<tr>
<td>Ubuntu</td>
<td>Automated</td>
<td>Main, Universe and Multiverse. For more information see: <a href="https://help.ubuntu.com/community/Repositories/Ubuntu">https://help.ubuntu.com/community/Repositories/Ubuntu</a></td>
</tr>
<tr>
<td>Debian</td>
<td>Automated</td>
<td>Main, updates.</td>
</tr>
<tr>
<td>All Others</td>
<td>Manual</td>
<td>Not applicable</td>
</tr>
</tbody>
</table>

Automated installation overview

The automated installation scripts perform the following installation steps:

1. Create a special system account for Polarion.
2. Set valid permissions for Polarion.
3. Adds Polarion as a system service at the default run levels of the Linux distribution it’s being installed on, and ensures that it starts after Apache.
4. Setup the default Apache configuration for Polarion and the Subversion used with it.
5. For a clean installation, create a new Subversion repository with initial or sample demo data.
6. Create symbolic links into default system places such as `/var/logs`, `/var/run`, `/srv/polarion`, and so on.
7. Start Polarion and third-party servers. Automated installation steps are detailed in *Automated Installation Procedure*. 
Manual installation

No automated installation script is used for a manual installation. Only the Polarion platform is installed without any of the required third-party software (SVN, Apache, and so on). Manual installation requires that you obtain all the required third-party software needed to support Polarion, install and configure them, and then install and start the Polarion server. An example of when a manual installation is necessary is if the computer does not yet have an internet connection.

For more information, see Appendix: Manual Installation on Linux.
5. Automated installation procedure

See the *Supported platforms and installation types* for the Linux versions that support the automated installation script procedure described below.

Warning:

If you already have one or more of the *required third-party* software packages installed, you can respond **NO** to the prompts asking if you want to install them, but be absolutely sure that you have it installed, otherwise you can break the automated process and it can be problematic to resume. If you are not sure that you have the required installation and configuration, it is better to say **YES** to the request to install the required third-party software.

1. Unpack the archive you downloaded from the Polarion web site into an empty directory.

2. If SELinux is bundled with or installed on your Linux OS, make sure it is not activated. Check the SELinux status with the `/usr/sbin/seatool -v` command.

3. Log in to the root account.

4. Navigate into the unpacked directory.

5. Run `chmod +x install.sh` after unzipping. (Only needs to be run once.)

6. Run the installation script: `./install.sh`.

7. Continue with the installation, answering questions when prompted by the script.

8. After the installation process the Polarion server will start automatically, which you can verify by opening `http://localhost/polarion` in a supported web browser.

Note:

Before using the system in a production environment, see *Securing the Polarion activation application*. 
5. Automated installation procedure
6. System startup and shutdown

Secure the activation application

Polarion includes an activation application that makes it possible to install or update a license while the Polarion server is running. Access to this application is not protected by any username or password. For production use, it is highly recommended to secure access to this application. See *Securing the Polarion activation application* for details.

Starting Polarion

The first time you start the Polarion server, no time estimates appear in the console because there is no data on which to base the estimate. Instead, not enough data for startup estimation is written to a log file at `/var/log/polarion.log`.

The first startup after an automated installation is automatic. This section provides the procedure for subsequent startups. This includes the first startup after a manual installation.

To start the Polarion server, execute whichever of the following is applicable:

- **Common command:** $POLARION_HOME$/bin/polarion.init start
  
  For a system that supports systemd: systemctl start polarion

- On CentOS, RHEL, Debian or Ubuntu: service polarion start

- SUSE/SLES 11: rcpolarion start

- **SUSE/SLES 12:** The Common command ($POLARION_HOME$/bin/polarion.init start )

On startup and on re-index operations, Polarion estimates and reports the amount of time the operation will take. You will see this estimate in the console and log file on subsequent startups, but for the first time, no data exists on which to base the estimate. The following startup phases are reported in the console and log file:

- Platform startup
- Context recognition
- Context initialization
- Revisions processing
- Build artifacts recognition
• BIR inspection
• Data indexing
• Polarion startup

**Shutting down Polarion**

To shut down Polarion, execute whichever of the following is applicable:

- **Common command**: `$POLARION_HOME$/bin/polarion.init stop`
  - For a system supporting `systemd`: `systemctl stop polarion`

- On CentOS, RHEL, Debian or Ubuntu: `service polarion stop`
- On SUSE: `rcpolarion stop`
- SUSE/SLES 11: `rcpolarion stop`
- SUSE/SLES 12: The Common command ($POLARION_HOME$/bin/polarion.init stop )

**Starting and stopping the PostgreSQL database**

The automated installation process creates a service for the PostgreSQL database named `postgresql-polarion`. If you manually start/restart your Polarion system, this service must be started after Apache and before the Polarion server. You can use one of the following commands to start the service:

- CentOS, RHEL, Debian or Ubuntu: `service postgresql-polarion start`
- SUSE/SLES 11: `rcpostgresql-polarion start`
- SUSE/SLES 12: `service postgresql-polarion start`

You can stop or restart the service by replacing `start` with `stop` or `restart` in the above commands.

If you install Polarion manually, the `postgresql-polarion` service is not present, so it is necessary to use utilities provided by PostgreSQL to start/stop the database. Starting/stopping of the PostgreSQL server is common for all Linux distributions and needs to be done on behalf of the Postgres user. You must initialize the database storage prior to running the server for the first time. (See *Configuration of third-party components*). All PostgreSQL utilities must be run on behalf of the postgres system user.

**Command Syntax:**

```
su postgres
```
pg_ctl -D <PATH_TO_PG_DATA> -l <PG_LOGFILE> start/stop/restart

In a typical installation this would translate to:

su postgres

pg_ctl -D /opt/polarion/data/postgres-data -l /opt/polarion/data/postgres-data/log.out -o "-p 5433" start

(The pg ctl command should be written to a single command line.)

You can define additional options, for example, on which port the database should run. For an overview of options, see http://www.postgresql.org/docs/9.4/static/app-pg-ctl.html.

Integration with Systemd

On Linux systems that support systemd, Polarion installs support for it and uses it for system startup. Integration with systemd ensures that other necessary software (Apache and PostgreSQL, for example) is started before Polarion.

With systemd, the start, stop, and restart commands can be used. (These are also delegated to polarion.init.) Other commands like reindex, demo and so on must be used with /opt/polarion/bin/polarion.init reindex. The start/stop/restart commands can also be used with polarion.init in systems supporting systemd.

Integration with Init System

The script /opt/polarion/bin/polarion.init is suitable for integration with the init system to start the Polarion server if systemd is not used. It supports start, stop and restart commands using the special system polarion account.

Caution:

polarion.init does not start the Apache and PostgreSQL servers. You should always make sure to start these by other means before starting Polarion server.
7. After installation

Adjust server memory allocation

The default installation uses **640 m (megabytes)** as the maximum memory allocation settings. This is fine for evaluation purposes but should be increased in production installations to avoid running out of memory.

**Tip:**
Set the Xmx/Xms values to 50% of the server’s total RAM.

1. Open the `[POLARION_HOME]/etc/config.sh` file in a text editor.
2. Adjust the following values to half of the server’s total RAM:
   
   - `Xms 650m`
   
   - `Xmx 650m`

   (Use MEGABYTES as the measure. For example, for 2 GB, specify `-Xmx 2000m`.)

3. Save the changed file.
4. Restart the Polarion server.

Configure the PostgreSQL database

The following should be done immediately following the installation and initial configuration:

- Configure the PostgreSQL database
- **Optimize the PostgreSQL database**
- **Secure the Polarion Activation Application**
- Change the default administrator password
- **Change the password for the 'polarion' SVN user**
- **Enter an error reporting email**

The PostgreSQL database must be properly configured before you can start the Polarion server. Automated installation scripts attempt to perform basic configuration so that Polarion can run. If the
script was unable to configure PostgreSQL you will need to configure it manually. See Appendix: Manual configuration of third-party components. Even if the script succeeds, some additional configuration of the PostgreSQL database is recommended for optimal performance.
Optimizing the PostgreSQL database

Beginning with version 2015 SR2, Polarion integrates the PostgreSQL database in all new installations. After a new Polarion installation containing this database, it is highly recommended that the administrator adjust some PostgreSQL settings to optimize performance. You should make the following changes in the `postgresql.conf` file. The database server needs to be restarted following this configuration.

**Default Linux Path: `/opt/polarion/data/postgres-data/`**

```plaintext
max_connections = 80 # should be < 10 * number of CPUs

shared_buffers = 2GB # should be 10% - 15% of total system RAM

work_mem = 10MB # should be 10MB - 100MB

maintenance_work_mem = 200MB

fsync = off

synchronous_commit = off

full_page_writes = off

wal_buffers = 256kB # should be more than size of common

# transaction

checkpoint_segments = 32

effective_cache_size = 4GB # should be approx 1/3 of total

# system RAM

max_locks_per_transaction = 100 # specific for Polarion

# Optimal planner performance setting

# For HDD, keep default setting. Otherwise, uncomment the
# applicable setting below:

# For SSD:
```
# random_page_cost = 1.5

# For SAN:

# random_page_cost = 2.0

Note:
On some Linux environments, the automatic configuration of PostgreSQL's shared_buffers parameter may calculate a value that exceeds the available shared memory configured by kernel parameter SHMMAX. In automated installations, the installer script assigns a default value of 24 MB for shared_buffers and posts the following message:

"NOTICE: Polarion attempted to change value of shared_buffers in postgresql.conf
to $v_shared_buffers_old but $v_shared_buffers_new was used instead because of low value in /proc/sys/kernel/shmmax."

If you see this message during the installation, you are advised to review your SHMMAX configuration and adjust shared_buffers for PostgreSQL manually.

Securing the Polarion activation application

Beginning with version 2015, Polarion includes an activation application that makes it possible to install or update a license during runtime of the Polarion server, without the need to copy the license file manually to the target machine. Access to this application is NOT initially protected by user name and password. For production use, it is highly recommended to secure access to this application directly in the Apache configuration.

Beginning with version 2015, there is a template Apache configuration file in the Polarion installation folder: /polarion/polarion/install/polarion.activation.conf.template

To ensure that a user name and password is requested when accessing the activation application (/polarion/activate/online and /polarion/activate/offline), copy this file to the Apache configuration folder, on Linux usually /etc/httpd/conf.d/.

After copying the file, rename it to remove the extension .template. Then open the file in any text editor and modify it according to the instruction comments provided.

The template configuration is prepared for both user file authentication (like Polarion uses for Subversion by default, with user passwords data in a file) and for authentication against an LDAP server.

In a multi-instance setup (a coordinator and one or more instances, that can be clustered), it is necessary to use this configuration only on the coordinator server (the activation application runs only on the coordinator). For additional information about this type of setup, see Polarion Enterprise Setup.
Change the default system administrator password

To help ensure the security of your Polarion system, you should change the default password of the System Administrator user account described below, and the password for the polarion user account of the integrated Subversion (SVN) repository.

The default System Administrator user account has access to all administrative functions of Polarion, including read-write access to the Subversion repository. After installing Polarion for actual production use, you should change the password on the default System Administrator account. Before doing so, consider creating another account with administrator permissions for yourself, and perhaps someone else.

To change the default administrator password:

1. Log on to the Polarion portal with the default System Administrator credentials. (username: admin, password: admin.)

2. Click My Polarion. The My Polarion page for the System Administrator account loads in the content area.

3. Click on the top right and click My Account.

4. In the My Account page click Edit.

5. Enter the new password in the New Password field, and again in the Reenter Password field.

6. If you want to continue using this account as the main system administrator account, you may wish to add your email address in the Email field, and add a description for the account in the Description field.

7. When finished editing the System Administrator profile, click Save. The password is now changed and you must use it next time you log on.

Warning:
Do not lose the new password.
If you lose the changed password, you will not be able to log on as the System Administrator user. If no other accounts exist with administrator permissions, it will not be possible to change the configuration, add projects, manage user accounts, etc.
Changing the Password for SVN User ‘polarion’

A Subversion repository user named polarion is created by default when you install Polarion. This user acts on behalf of the Polarion application and consequently has extensive permissions including read permission for all projects. Access to this user by unauthorized people would compromise the security of your Polarion system, so it is advisable to change this password before putting the system into production use.

The following steps assume you use passwd file authentication, which is the most common method.

1. Stop the Polarion server before changing this password.

2. Use the `htpasswd` utility to change the password for the user polarion. The utility is installed on your system together with the Apache server binaries.
   - **Utility Syntax:** `htpasswd path/passwdfilename username`
   - **Example:** `htpasswd /opt/polarion/data/svn/passwd polarion`

3. Next, change the value of the password property in the `polarion.properties` file to the password you set with the `htpasswd` utility. Typical location of this file is `/opt/polarion/etc/polarion.properties`.

   **Note:**
   **For LDAP Users**
   The typical setup for most Polarion users is passwd file authentication for the polarion user with failover to LDAP for company users. This is also the default Polarion setup.
   For such setups, you do not need to enter the polarion user to your LDAP users.

**Enter an error reporting email**

The email for the `error.report.email` property in the `polarion.properties` file is empty by default for new installations. Add the email you want error reports sent to.
8. Configure OLE object support and attachment preview generator

Overview

It is possible to import Microsoft Word documents that contain OLE objects. Polarion can display OLE Object thumbnails during Word document import, including EMF thumbnails for Visio diagrams but a third-party image converter must be installed and configured to work with Polarion first. While ImageMagick is recommended, you can also use other third-party image conversion tools.

OLE objects in documents must also contain their thumbnails in the .emf or .wmf file formats, and the image converter you use must support their conversion to JPEG. OLE Objects themselves are not imported, only their thumbnails.

Note:
When configuring OLE imports for DOORS and ReqIF, thumbnails will be generated on the fly.

Polarion can also display previews for common attachment file types like Word, Excel, Visio, PDF if an external image conversion application is installed and configured to work with Polarion.

Prerequisites

For the conversion server

• A computer or virtual machine running Microsoft Windows 10, Windows Server 16 or Windows Server 19 accessible as a network share to the Linux machine hosting Polarion.

• The applications that you want to view previews from. (Microsoft Word, Excel, Power Point and Acrobat Reader.)

• Teamcenter Visualization Convert & Print installed.

• An image converter: (ImageMagick is recommended but you also use an alternative.)

  Tip:
  Having trouble configuring an alternative image converter? Support can help.

• OpenSSH installed and configured.
Configuration workflow

1. Install an image converter. (ImageMagick is recommended.)
2. Install OpenSSH.
3. Configure OpenSSH.
4. Set up SMB shares.
5. Configure the client.
6. Set up scripts.
7. Configure Polarion.

Install OpenSSH

Select the OpenSSH installation method below that aligns with your server setup.

On Windows Server 2019 or Windows 10.1809

Install OpenSSH with Powershell

1. Open a PowerShell window as an administrator.
   (Type Powershell in the Windows menu.)
2. Type `Get-WindowsCapability -Online | ? Name -like 'OpenSSH*'` and press Enter.
   This should return the following:
   
   Name : OpenSSH.Client~~~~0.0.1.0State : NotPresent
   Name : OpenSSH.Server~~~~0.0.1.0State : NotPresent
3. Type `Add-WindowsCapability -Online -Name OpenSSH.Server~~~~0.0.1.0` and press Enter.
   This will install the OpenSSH Server.
4. Now you must configure OpenSSH.
Install OpenSSH on older versions of Windows

1. Download OpenSSH.

2. Unpack it into C:\Program Files\OpenSSH.

3. Run PowerShell as an administrator.

4. Run the following commands to install the ssh service.
   a. Set-Location "C:\Program Files\OpenSSH"
   b. .\install-sshd.ps1

5. Now you must configure OpenSSH.

Configure OpenSSH

1. Open a PowerShell window as an administrator.

2. Enter the following to start the SSHD service:
   Start-Service sshd

3. (Optional but recommended.) Set the SSHD so that it automatically starts:
   Set-Service -Name sshd -StartupType 'Automatic'

4. Confirm that the Firewall rule is configured. (It should be created automatically.)
   Get-NetFirewallRule -Name *ssh*
   The firewall rule named OpenSSH-Server-In-TCP should now be enabled.

5. Check permissions using the following commands:
   FixHostFilePermissions.ps1
   FixUserFilePermissions.ps1
   OpenSSH should now be configured so that it uses password verification.

6. Set PowerShell as the default for OpenSSH:
New-ItemProperty -Path "HKLM:\SOFTWARE\OpenSSH" -Name DefaultShell -Value "C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe" -PropertyType String -Force

You should now be able to connect using a Linux machine with the following:

    ssh your_user@hostname Get-Process

7. (Optional but recommended.) Configure key verification.
   
   a. Open a [PowerShell] window as an administrator.
   
   b. Enter the following command:

       Icacls authorized_keys /remove "NT SERVICE\sshd"

       *Due to issues with PowerShell, NT SERVICE\sshd should not have permission to access the authorized_keys file. If it does, then key-based authentication will not work. This command removes that permission.*

   c. Open the following file in a text editor. *(c:\ProgramData\ssh\sshd_config file)*
   
   d. Comment out (#) the following lines:

       # Match Group administrators
       # AuthorizedKeysFile __PROGRAMDATA__/ssh/administrators_authorized_keys

   e. Create an .ssh file and folder in [PowerShell] *(C:\Users\your_SSH_user\.ssh)* by running the following commands:

       New-Item -ItemType Directory -Path "C:\Users\your_SSH_user\.ssh"
       Set-Location "C:\Users\your_SSH_user\.ssh"
       New-Item -ItemType File authorized_keys

   f. Insert the user public key pair that was generated on the clients into the authorized_keys file.

       *(See setup ssh-key pair for details.)*

   g. Enter the following commands to check the permissions.

       Set-Location "C:\Program Files\OpenSSH"
8. Now you must set up the SMB shares.

Set up SMB shares and configure the client

Once you’ve installed and configured OpenSSH it’s time to set up your SMB shares.

1. Create the following directories:
   
   C:\visualization\magick-tmp
   
   C:\visualization\tcvis-tmp

2. Configure ACL for your ssh_user so that they’re allowed to modify permissions.

3. Enable the shares.

4. Now you must configure the client.

Configure the client

Select one of the following ways to configure the client so that OLE users can see OLE object previews.

1. ssh-key verification (Recommended)

2. password verification
   (Requires more maintenance.)

1. ssh-key verification configuration option (Recommended)

1. Set up the ssh-key pair by entering the following console commands.

   sudo -i
   su polarion
   ssh-keygen -t rsa

   (You should see ls -la /home/polarion/.ssh.)

   -rw------- 1 polarion www-data 1675 Jun 3 13:38 id_rsa
2. Check the permissions of the key pair files and make sure that Polarion is the owner.

3. Copy and paste the `id_rsa.pub` public key into the target user profile on your conversion server.

   (The user that will be used for the ssh connection. For example, `C:\Users\your_SSH_user\.ssh\authorized_keys`.)

4. Test your settings by entering the following commands:

   ```bash
   TARGET_SERVER='SSH_user_on_the_windows_side@windows_server_FQDN'
   ssh $TARGET_SERVER
   
   You should see the following:
   
   Windows PowerShell
   Copyright (C) Microsoft Corporation. All rights reserved.
   
   5. Enter the following:

   ```bash
   PS C:\Users\your_SSH_user>hostname
   
   You should see the host name of your conversion server.
   
   6. Exit the ssh connection.

   ```bash
   PS exit
   
   7. Prepare your folders on your Linux machine as a root user by entering the following commands.

   ```bash
   mkdir /mnt/magick-tmp
   mkdir /opt/ole-convert
   mkdir /mnt/tcvis-preview-gen
   mkdir /opt/tcvis-preview-gen
   
   8. Now you can **set up the scripts**.

   **2. Password verification configuration option**

   1. Install sshpass by entering the following console commands:
sudo -i

apt-get install sshpass

2. Prepare the target folder and files by entering the following commands:

a. mkdir /mnt/magick-tmp
   mkdir -p /opt/ole-convert/pw
   vim /opt/ole-convert/pw/.sshzz

b. Enter your ssh account password.

c. Enter the following commands:

   chown polarion:root /opt/ole-convert/pw/.sshzz
   chmod 600 /opt/ole-convert/pw/.sshzz
   mkdir -p /opt/tcvis-preview-gen/pw
   mkdir/mnt/tcvis-preview-gen
   vim /opt/tcvis-preview-gen/pw/.sshzz

d. Enter your ssh account password.

e. chown polarion:root /opt/tcvis-preview-gen/pw/.sshzz
   chmod 600 /opt/ole-convert/pw/.sshzz

3. Now it’s time to establish and test your setup.

a. **Important! You must switch to the polarion user.**
   
   su polarion

b. Enter the following commands as the polarion user.

   PWFILE=/opt/tcvis-preview-gen/pw/.sshzz

   TARGET_SERVER='SSH_user_on_the_windows_side@windows_server_FQDN'

   (And add your server as trusted.)
sshpass -f $PWFILE ssh $TARGET_SERVER "get-process | fl *"

You should see all of your SHH server's running processes.

4. Now you can set up the scripts.

Set up the scripts

1. Switch to an elevated user, then open the oleconvert.sh file.

   ```bash
   vim /opt/ole-convert/oleconvert.sh
   ```

2. Paste the following into the file and save.

   (Replacing the generic server information with your own.)

   ```bash
   #!/bin/sh
   infile=$1
   inname=`basename "$1"`
   outfile=$2
   outname=`basename "$2"`
   LINUX_PATH=/mnt/magick-tmp
   WIN_PATH=C:/visualization/magick-tmp
   #enable for password verification
   #PWFILE=/opt/ole-convert/pw/.sshzz
   TARGET_SERVER='SSH_user_on_the_windows_side@windows_server_FQDN'
   cp "$infile" "$LINUX_PATH/$inname"
   PWS_COMMAND="Set-Location "C:\Program Files\ImageMagick" ; . \magick.exe convert "$WIN_PATH/$inname" "$WIN_PATH/$outname"
   #enable for password verification
   #sshpass -f $PWFILE ssh $TARGET_SERVER $PWS_COMMAND
   #for ssh-key verification
   ssh $TARGET_SERVER $PWS_COMMAND
   rm "$LINUX_PATH/$inname"
   mv "$LINUX_PATH/$outname" "$outfile"
   ```

3. Enter the following command:

   ```bash
   chmod +x /opt/ole-convert/oleconvert.sh
   ```

4. Open the tcvis.sh file.

   ```bash
   vim /opt/tcvis-preview-gen/tcvis.sh.
   ```

5. Paste the following into the file and save. (Replacing the generic server information with your own.)
#!/bin/sh
infile=$1
inname=`basename "$1"`
outfile=$2
outname=`basename "$2"`
LINUX_PATH=/mnt/tcvis-preview-gen
WIN_PATH=C:/visualization/tcvis-tmp
#enable for password verification
#PWFILE=/opt/tcvis-preview-gen/pw/.sshzz
TARGET_SERVER='SSH_user_on_the_windows_side@windows_server_FQDN'
cp "$infile" "$LINUX_PATH/$inname"
echo running $PWS_COMMAND
#for password verification
#sshpass -f $PWFILE ssh $TARGET_SERVER $PWS_COMMAND
#enable for ssh-key verification
ssh $TARGET_SERVER $PWS_COMMAND
rm "$LINUX_PATH/$inname"
mv "$LINUX_PATH/$outname" "$outfile"

6. Enter the following command:

   chmod +x /opt/tcvis-preview-gen/tcvis.sh

7. Mount the SMB shares by entering the following:

   apt-get install cifs-utils

   vim /etc/fstab

8. Insert rows by entering the following:

   //conversion_server/tcvis-tmp /mnt/tcvis-preview-gen cifs
   vers=2.1,_netdev,uid=999,gid=33,rw,exec,auto,iocharset=utf8,sec=ntlm,credentials=/var/pw/.cifs 0 0

   //conversion_server/magick-tmp /mnt/magick-tmp cifs
   vers=2.1,_netdev,uid=999,gid=33,rw,exec,auto,iocharset=utf8,sec=ntlm,credentials=/var/pw/.cifs 0 0

   mkdir /var/pw

   vim /var/pw/.cifs
And input the share password in the following format:

```plaintext
username=msusername
password=mspassword
```

9. Secure it by entering the following:

```bash
chown root:root /var/pw/.cifs
chmod 600 /var/pw/.cifs
```

10. Mount the drives with:

```bash
mount -a
```

11. Check the mounted drives with:

```bash
df -h
```

12. All that's left is to configure Polarion.

**Configure Polarion**

1. Open the `polarion.properties` file in a text editor by entering the following command:

```bash
vim /opt/polarion/etc/polarion.properties
```

2. Paste the following into the open `polarion.properties` file.

```plaintext
com.siemens.polarion.previewgenerator.external.app=/opt/tcvis-preview-gen/tcvis.sh
com.siemens.polarion.previewgenerator.external.extensions=pdf,docx,doc,xlsx,xls,ppt,pptx
com.siemens.polarion.previewgenerator.external.param10=1600x?px
com.siemens.polarion.previewgenerator.external.param1=$in
com.siemens.polarion.previewgenerator.external.param2=$out
com.siemens.polarion.oleconverter.usefiles=true
com.siemens.polarion.oleconverter.app=/opt/ole-convert/oleconvert.sh
com.siemens.polarion.oleconverter.param1=$in
com.siemens.polarion.oleconverter.param2=$out
com.siemens.polarion.oleconverter.convertedImageFormat=png
```

3. Save the file then restart Polarion with:
systemctl restart polarion

You should now see a preview of inserted attachments rather than icons.
8. Configure OLE object support and attachment preview generator
9. Licensing and activation

Overview

In order to use Polarion you must obtain a license. A license with the necessary key and file is normally delivered by email to the address provided by the person who purchased the license. If you need help obtaining a license please contact sales@polarion.com.

Polarion installs with a 30-day evaluation license. After obtaining a license for production use, you must activate your Polarion installation. The login page provides options leading to online and offline activation instructions. You will need the information provided by Polarion to complete the activation.

Using different license types

Several different license types are available - Evaluation, Site, User-limited, and so on. If you begin using Polarion with one type of license key (Evaluation, for example), and want to continue using it with a different license type, simply remove the current license key file from the license folder and copy the new license key file there. If Polarion server is running, you will need to restart it for the new license to take effect. You may install multiple license keys for different license types and/or Polarion products on the same server.

Assigning named and concurrent users

If your license provides for named and/or concurrent users, you will need to add assignments for each type of user in the appropriate section of the users file. By default this file is located in the license folder of your Polarion installation. If you change the location for license key files, be sure to move the users file to the same folder that stores your license key file.

You can edit it in the License topic in Global Administration in the portal. The file contains comments with complete instructions on how to find the user IDs of your named/concurrent users, and make the relevant assignments. Be sure that write permission is set for the /opt/polarion/polarion/license folder.
License usage log file

Administrators and managers can monitor license usage by checking the license usage log file `log4j-licensing-TIMESTAMP.log`. This file is located in the `/opt/polarion/data/workspace/.metadata` directory by default.

(The `/opt/polarion/data` location is configurable during installation.)

When a concurrent user logs in/out, a license usage statistics report is written to the licensing log.

**Note:**

Concurrent licensing is not supported for all products.

The following example shows one user currently using an enterprise concurrent license type, the greatest number of users of this license during the current server session (peak), and the maximum number of users allowed by the license (limit).

```
2008-05-14 11:12:29,609 [TP-Processor2] INFO PolarionLicensing -
STATS:enterpriseConcurrentUsers,current:1,peak:2,limit:20
```
10. Subversion optimization

Polarion uses a subversion (SVN) repository as its main data storage. There are two topics in the Administrator's Guide component of Polarion's online Help that provide guidance for administrators about optimizing subversion for best performance. It is recommended that you review them before going into production with a new or updated installation.

- Administrator's Guide → Advanced Administration→Topics → Optimizing Subversion

- Administrator's Guide→ System Maintenance→ Topics → Maintaining Subversion (SVN)

Tip:
You should always set up and use the svn:// protocol for system user access.
11. Multiple repository setup

There are two Polarion features that enable you to work with multiple repositories, but they are fundamentally different. You need to understand the basics of each feature before deciding which approach to multiple repositories best meets your needs.

The **External Repository** feature gives you the ability to link Polarion artifacts stored in Polarion’s integrated repository with source code changes (revisions) stored on one or more external SVN or Git repositories. After installation, you can configure Polarion to use one or more external repositories in addition to the SVN repository bundled and installed with Polarion. See [Administrator’s Guide: Configuring Repositories](#) for more information.

The **Clustering** feature enables you to run Polarion on multiple servers, either physical, virtual, or a combination of both. The topography can be set up to host multiple Polarion servers running on separate machines each with its own Polarion repository, and/or multiple machines all accessing a single Polarion repository. (Polarion servers on any node can optionally be configured to access external repositories, as described above.)

Special installation and configuration procedures beyond the scope of this guide are required to set up a clustered multi-server environment. See [Polarion® ALM™ Enterprise Setup](#) for more information.
12. Accessing the Polarion Portal

Open a supported web browser and enter the following URL:

http://localhost/polarion/

On your first login after installation, you can login with the default system administrator credentials:

- User ID: admin
- Password: admin
13. LDAP authorization

In a new installation, users are authorized using the Subversion integrated policy access functions (directives `AuthzSVNAccessFile` and `AuthUserFile` in `polarionSVN.conf` file). If you have an LDAP infrastructure, you can make Polarion authorize users against the LDAP database.

Information on performing this configuration, together with some examples, is provided in the `polarionSVN.conf` configuration file. The file is located at: `[POLARION_HOME]\bundled\apache\conf\extra\polarionSVN.conf`

The file is located one of the following paths, depending on your Linux distribution:

- `/etc/apache2/conf.d`
- `/etc/httpd/conf.d`

After modifying the configuration file, the Apache server must be restarted to reflect the changes.

For more information about the Apache LDAP modules and their capabilities, visit these web pages:

- `https://httpd.apache.org/docs/2.4/mod/mod_authnz_ldap.html`

You can find information on configuring Polarion to work with LDAP in the Polarion Help topic `Administrator’s Guide → User Management → Integrating Polarion Server with LDAP/Active Directory`. 
13. LDAP authorization
14. Next steps after installation

Once you have installed Polarion and logged in to the Portal, consider taking a look at the demo projects (assuming you installed demo data). Click on the drop-down control in the Navigation panel on the left, select Open Project or Project Group, and open any project in the Demo Projects group. See the User Guide → Getting Started with Projects section in Polarion's Help for tips.

You may want to do some initial global customizations such as custom Work Item types, Workflows, Reports, SSL Support and more. You will find topics on these configurations in the Administrator's Guide section of Polarion's Help.

Once you have your Polarion system running, and any global customizations done, you are ready to begin setting up your own projects and user accounts. Look up the following topics in the Administrator's Guide: Creating and Managing Projects and Managing Users and Permissions section in Polarion's Help.

Tip:

Polarion supports Single Sign On (SSO) authentication using Security assertion markup language 2.0 (SAML), Kerberos tokens and Teamcenter security services. See Single Sign On (SSO) with Polarion on Polarion's Doc Center portal for details.
15. Removing Polarion

An automated uninstaller script `uninstall.sh` is provided for Linux platforms.

Remove the `/var/log/polarion` folder and `/var/run/polarion.pid`.

Polarion should then be uninstalled.

**Warning:**

Polarion's Subversion repository is stored in the `/opt/polarion` folder. Be sure this repository does not contain production data that must be preserved! If it does, be sure to make a backup before uninstalling Polarion.
16. Technical support

Polarion is problem-free for most people... at least that's been our experience. However, it's impossible to anticipate all the conditions and environments where Polarion may be used. If an arises, Polarion’s Technical Support team maintains the Customer Self-service Portal which includes an extensive knowledge base of common problems and solutions and troubleshooting information, as well as the possibility to submit, manage, and review your own specific support cases.

For information about the portal and technical support options, please visit: https://polarion.plm.automation.siemens.com/techsupport/resources
17. Appendix

Default users and groups

<table>
<thead>
<tr>
<th>OS</th>
<th>User</th>
<th>Group</th>
<th>Passwd Utility</th>
</tr>
</thead>
<tbody>
<tr>
<td>RedHat (CentOS)</td>
<td>apache</td>
<td>apache</td>
<td>htpasswd</td>
</tr>
<tr>
<td>Debian</td>
<td>www-data</td>
<td>www-data</td>
<td>htpasswd</td>
</tr>
<tr>
<td>Ubuntu Server</td>
<td>www-data</td>
<td>www-data</td>
<td>htpasswd</td>
</tr>
<tr>
<td>SUSE</td>
<td>wwwrun</td>
<td>www</td>
<td>htpasswd2</td>
</tr>
</tbody>
</table>

Enabling email notifications

If you did not configure email notification settings in the installation program, you can do this after installation by setting the host name in the `announcer.smtp.host` property in the `polarion.properties` file located in `[POLARION_HOME]\polarion\configuration`. There you should also set the `announcer.smtp.user` and `announcer.smtp.password` properties to a valid email account on the SMTP host specified in `announcer.smtp.host`. You may want to create a dedicated email address on your SMTP host for use by the Polarion notifications system.

When this configuration is correctly set up, the system will send notification emails about various events according to the notification targets configuration. For information on configuring email notifications, see the Administrator’s Guide: Configuring Notifications in Polarion’s Help.

Manual installation on Linux

It is highly recommended that you run Polarion on one of the supported Linux operating systems and use the automated installation scripts provided in the product distributions for Linux.

There may be cases when Polarion must be installed manually. For example, perhaps an automated installation script doesn’t recognize your system as one of the supported Linux systems due to customization or other factors. Or possibly, your server does not yet have an internet connection, so various packages downloaded during the automated installation are not accessible. Therefore, automated installation cannot proceed. (Scripts inform you if automated installation is not possible).
System requirements and recommendations

Before starting with a manual installation, go over the information listed below:

- **Server software requirements** (OS, OpenJDK 11, VCS, web server).
- **Server hardware requirements** (RAM, disk storage).
- **Client software requirements** (OS, web browsers, Adobe Flash).
- **Client hardware requirements** (RAM, display, connection).

Required third-party software components

In order to run Polarion, you need to have several other software components installed and configured:

- **Apache HTTP Server version 2.2** or later. (The latest 2.4.x is recommended.)
- Subversion server version 1.6.x, 1.7.x, 1.8.x or 1.9.x
- **OpenJDK 11** *(Recommended version: AdoptOpenJDK 11 LTS)*
- **PostgreSQL version 8.4 or higher.**
- Fonts package *urw-fonts* (on SUSE and CentOS)

**Apache server**

Apache HTTP Server is a required application for Polarion. We highly recommend installing it first.

You can download the appropriate distribution at the Apache website at:


Installation Help for Apache HTTP server is available from Apache at:

[http://httpd.apache.org/docs/2.4/install.html](http://httpd.apache.org/docs/2.4/install.html).

**htpassword Utility**

After installation you must ensure that the `htpasswd` utility is executable. The `apache/bin` folder must be present among other paths in PATH system variable, or the full path needs to be specified in the system configuration file `polarion.properties`. 
Note:
On SUSE the utility may be called `httpasswd2`
Subversion

The Subversion version control system (versions 1.6.x, 1.7.x, 1.8.x or 1.9.x) is also required for Polarion. You can download the appropriate distribution at http://subversion.apache.org/. Full user/administrator documentation is available on a third-party website at http://svnbook.red-bean.com/.

PostgreSQL

In new installations beginning with version 2015 SR2, Polarion server cannot start if PostgreSQL is not installed and configured. Please use your distribution's package manager to download these two packages: postgresql and postgresql-contrib. On RedHat you will need the postgresql-server package instead of the postgresql package.

You can follow the instructions posted online at http://www.postgresql.org/download/ or use one of the following command sets:

CentOS, RedHat:

- yum install postgresql
- yum install postgresql-contrib

Debian, Ubuntu:

- apt-get install postgresql
- apt-get install postgresql-contrib

SUSE: Available from openSuse Build Service (https://build.opensuse.org/) in the server: database:postgresql project:

- yast --install postgresql
- yast --install postgresql-contrib

After installing PostgreSQL, you must configure it and Polarion. This configuration is covered in the Configuration of third-party components.
Install OpenJDK 11

Tip:
Recommended OpenJDK 11 distribution: AdoptOpenJDK 11 (LTS).

Warning:
- Backup the Java Keystore before installing OpenJDK and reimport it once it and the Polarion installation are complete.
- When installing or upgrading to AdoptOpenJDK 11 (LTS) make sure the default file encoding matches the same encoding used by the previous version of Java.
- As of Polarion 19 Oracle Java SE Development Kit 8 is no longer supported.

1. Go to AdoptOpenJDK 11 (LTS).
   (AdoptOpenJDK 11 (LTS) is the recommended OpenJDK distribution because it's continuously tested with Polarion and offers long-term support.)

2. Select the HotSpot implementation for the Linux x64 platform and download the JDK archive with your preferred tool and extract the files to a preferred location.

3. Set the JDK_HOME and JAVA_HOME environment variables. (Refer to the manual for your Linux distribution.)

4. You can test the configuration by executing $JDK_HOME/bin/java -version and $JAVA_HOME/bin/java -version
   (This should display the information about AdoptOpenJDK 11)

Match default file encoding:

1. Search for the file.encoding property in the main log file (/opt/polarion/data/logs/main/).

2. If the default file encoding for the new Open JDK 11 differs, then define it explicitly as a Java Runtime property by adding the following property to the config.sh file: (Default location: /opt/polarion/etc/config.sh)
   -Dfile.encoding=file_encoding
   Replace file_encoding with the one you use.
Fonts Package

Phantom JS, used for previewing and exporting highcharts in PDF, is installed on all OS platforms. On SUSE and CentOS, URW fonts must be explicitly installed (see Server software requirements), otherwise exported diagrams rendered by Phantom JS will display rectangles instead of letters.
Import a certificate to the Java Keystore

You will need to import a certificate to the Java Keystore if:

• You are not using a SSL certificate that is signed by an authority trusted by Java.
  Use of a trusted certificate is preferred and recommended because using an untrusted certificate,
  such as a self-signed certificate, will cause web services communication to fail with the
  SSLHandshakeException error.

• Before making the switch from Oracle JDK8 to OpenJDK 11.

The information is important only if you are not using a SSL certificate that is signed by an authority
trusted by Java. Use of a trusted certificate is preferred and recommended because using an untrusted
certificate, such as a self-signed certificate, will cause web services communication to fail with the
SSLHandshakeException error. If you do opt to use an untrusted certificate, then you must import it
into the Java keystore. The general import procedure is described below, followed by examples for Linux
and Windows.

1. Copy the default keystore $JDK_HOME/lib/security/cacerts as $JDK_HOME/lib/security/jssecacerts.
   This will leave the original cacerts file available as a backup. JSSE will use the jssecacerts file,
   if present, instead of cacerts. Jssecacerts needs to start as a copy of cacerts, which it
   overrides rather than extends.

2. Import the certificate to the jssecacerts keystore using the following command, replacing
   variables as noted below:

   $JDK_HOME/bin/keytool -importcert -file $CERT -alias $ALIAS
   -keystore $JRE_HOME/lib/security/jssecacerts -storepass changeit

   a. Replace $JDK_HOME with your actual JDK home path.
   b. Replace $CERT with the path to your certificate the you previously installed to the system.
   c. Replace $ALIAS with the preferred alias to be used in the keystore.
   d. Note that changeit is the default password for Java's cacerts file. Check whether it has
      been changed on your system.

3. When prompted, check the certificate and confirm that it should be trusted. The prompt to verify
   and confirm the certificate can be suppressed by adding option -noprompt.
Windows example:

The following command should be written as a single line. It must be run as Administrator. If the Java paths on your system contain spaces, they must be contained in a pair of double straight quotes, as shown.

"C:\Program Files\Java\jdk-11.0.1\bin\keytool" -importcert -file C:\Polarion\bundled\apache\conf\certificate.crt -alias labs.polarion.com -keystore "C:\Program Files\Java\jdk-11.0.1\lib\security\jssecacerts" -storepass changeit

Linux Example (CentOS)

This example following command should be written as a single line:


Depending on your operating system and version, additional command parameters may be necessary.

(See https://www.cloudera.com to learn more.)

Keytool Commands

Here are some potentially useful keytool commands:

keytool -list -keystore %JAVA_HOME%\lib\security\jssecacerts -storepass changeit

keytool -delete -alias mykey -keystore %JAVA_HOME%\lib\security\jssecacerts -storepass changeit

keytool -importcert -help

keytool -help

Configure third-party components

This section covers various configuration requirements for third-party software that enable Polarion to run with it.

For further inspiration you may also inspect supported OS configurations located in
Apache HTTPD

1. Apache must be compiled with the WebDAV module (mod_dav.so).

2. Modules from Subversion must be properly installed (mod_authz_svn.so and mod_dav_svn.so)
The following lines must be added into an Apache configuration file into the LoadModule section.
This can be the default configuration file httpd.conf ([Apache2_Dir]/conf/httpd.conf), or any other conf file, including a new one you create for the purpose. Lines:
   LoadModule dav_svn_module modules/mod_dav_svn.so
   LoadModule authz_svn_module modules/mod_authz_svn.so

3. Add the following lines to your httpd.conf:
   <Location /repo>
   DAV svn
   SVNPath "/opt/polarion/data/svn/repo"
   # our access control policy
   AuthzSVNAccessFile "/opt/polarion/data/svn/access"
   # try anonymous access first, resort to real
   # authentication if necessary.
   Satisfy Any
   Require valid-user
   # How to authenticate a user. (NOTE: Polarion does not
   # currently support HTTP Digest access authentication.)
   AuthType Basic
   AuthName "Subversion repository on localhost"
   AuthUserFile "/opt/polarion/data/svn/passwd"
   SVNAutoversioning on
   </Location>
   # polarion specific configuration of apache for svn
   # and maven2 repo
   # Maven 2 shared repository
   <Directory "/opt/polarion/data/shared-maven-repo">
   Options Indexes
   Order allow,deny
   Allow from all
   </Directory>
   Alias /maven2 "/opt/polarion/data/shared-maven-repo"
Pay attention to the name of repository (repo) in the first tag (<Location/repo>). It's the name (an
alias) of your subversion repository.
Don't forget to specify the correct paths to the real repository folder in the SVNPath parameter, to
the access file in the AuthzSVNAccessFile parameter, and to the passwd file in the AuthUserFile
parameter.
Pay attention to the paths used in the httpd.conf example above. They must be the same as those
used by default by the libinstall/default.sh script. This script won't work if different paths are used.
4. Create a file named `workers.properties` in your Apache server’s configuration directory (where the `httpd.conf` file resides), with the following:
   ```properties
   worker.list=worker1
   worker.worker1.type=ajp13
   worker.worker1.host=127.0.0.1
   worker.worker1.port=8889
   worker.worker1.lbfactor=50
   worker.worker1.cachesize=50
   worker.worker1.cache_timeout=600
   worker.worker1.socket_keepalive=1
   worker.worker1.recycle_timeout=300
   ```

5. Create a file named `polarion_mounts.properties` in your Apache server’s configuration directory (where the `httpd.conf` file resides). Add the following lines:
   ```properties
   /polarion/*=worker1
   /polarion=worker1
   /svnwebclient/*=worker1
   /svnwebclient=worker1
   ```

Subversion

Either the subversion binary (svn) must be on PATH or the javahl libraries must be properly installed.

JDK

The configuration requires several files from the JDK.

- The environment variable `JDK_HOME` must be properly set.
  (Remember to update the `JDK_HOME` variable in `/opt/polarion/etc/config.sh`.)

Warning:

(Do not update to a new major JDK version until support for it by Polarion is officially announced.)

Java Virtual Machine memory limit

If you allocate too much memory for the Java Virtual Machine (JVM), the operating system will not initialize it. Diagnosing the issue can be difficult because the service simply does not start and no error log is written. On some Linux versions, a message may appear in the console indicating that the Java Virtual Machine could not be initialized, but without indicating the reason.

The amount of memory you can allocate to the JVM depends on what OS you have and how much total memory exists on the computer. The more total memory the more you can allocate to the JVM before the operating system imposes a limit the better.
Update Java

It's good to update OpenJDK regularly for security reasons, but check the README.html file to ensure that major Java version updates are officially supported by Polarion before updating to them.

Warning:
Oracle's Java SE Development Kit 8 is no longer supported.

Warning:
If you added extra GC related runtime parameters for Java, you will need to update so that they'll work with OpenJDK 11. (If you don't, the Java Virtual Machine may fail to start.)

Tip:
Not sure how to update your custom GC related runtime parameters?
The following links will help:
http://openjdk.java.net/jeps/158
http://openjdk.java.net/jeps/271
Still stuck?
Contact SIEMENS' GTAC support system.

Please take the usual steps specific to your distribution.

When updating Java please remember to update variable JDK_HOME in /opt/polarion/etc/config.sh.

Warning:
When installing or upgrading to OpenJDK 11 make sure the default file encoding matches the same encoding used by the previous version of Java.
PostgreSQL

The location of PostgreSQL utilities depends on the specific Linux distribution. The command examples below assume that the folder containing PostgreSQL binaries is added to your system PATH variable.

The OS user postgres is normally created as a result of the PostgreSQL installation. To be able to use the psql utility, you must access it as this user.

A) Create the PostgreSQL database storage:

1. Create a folder to store PostgreSQL data and make the postgres user the owner of this folder on Linux. Any location will be fine, as long as the postgres user owns the folder. You can use the following example where the folder for PostgreSQL data is: /opt/polarion/data/postgres-data

   mkdir /opt/polarion/data/postgres-data
   chown postgres:postgres /opt/polarion/data/postgres-data

2. As the postgres user, initialize the database using the initdb command. The syntax differs for PostgreSQL versions, as shown in the following examples, with our /opt/data/postgres-data folder. Use psql --version to check for the version.

   For PostgreSQL version 9.1 and earlier:
   
   su postgres
   initdb -D /opt/polarion/data/postgres-data -E utf8

   For PostgreSQL versions after 9.1:
   
   initdb -D /opt/polarion/data/postgres-data -E utf8
   --auth-host=md5

   In some cases, initdb is not available immediately after the installation, and you need to add to your path. You can use the locate command to find the location of psql. If it cannot be found, use the updatedb command and search for it again. More information about initdb can be found at: http://www.postgresql.org/docs/9.4/static/app-initdb.html.

3. Start the PostgreSQL server. Remember that all PostgreSQL utilities must be run on behalf of the postgres system user, and you must initialize the database storage as described in step 2 above prior to running the server for the first time.

   su postgres
   pg_ctl -D /opt/polarion/data/postgres-data -l /opt/polarion/data/postgres-data/log.out -o "-p 5433" start

   (The pg_ctl command should be written to a single command line.) You can define additional options, such as the port on which the database should run. Check http://www.postgresql.org/docs/9.4/static/app-pg-ctl.html for an overview of options.

B) Connect to the Root Database of PostgreSQL:

For configuration of Polarion databases, you need to connect to the root database of PostgreSQL postgres using your favorite SQL client, for example Squirrel), or just use PostgreSQL's built in the psql tool.
1. On Linux, the operations with PostgreSQL database must be run on behalf of the **postgres** user: `su postgres

2. Connect to the PostgreSQL postgres root database on behalf of the postgres user (or the database's superuser): `psql -p <port> postgres`

The syntax for psql is: `psql -p <port> -U <user> <database>`

Example: `psql -p 5433 -U polarion polarion_history`

**C) Create the Polarion User and the Databases:**

Polarion uses two databases, **polarion** and **polarion_history**, which will be owned by the polarion user, so the first step is to create the polarion user. To do this, execute the following three SQL statements using your SQL client:

1. Create the **polarion** user and replace the `<your-password>` with a password of your choice: (The password cannot contain any of the following characters: @, :, \, ' or leading/trailing spaces.)
   ```sql
   CREATE USER polarion WITH PASSWORD '<your-password>' CREATEROLE;
   ```

2. Create the Polarion database:
   ```sql
   CREATE DATABASE polarion OWNER polarion ENCODING 'UTF8';
   ```

3. Create the polarion_history database:
   ```sql
   CREATE DATABASE polarion_history OWNER polarion ENCODING 'UTF8';
   ```

4. Quit the psql utility using the `\q` command.

**D) Install dblink Extension and PLPGSQL Language:**

After you have created the two databases for Polarion, you need to install the DBLink extension on both of them. The DBLink extension is used to enable Lucene queries within SQL queries. Some functions used by Polarion use the PLPGSQL language, which needs to be installed as well.

1. Connect to the polarion database via psql client as postgres database user (or the superuser): `psql -p 5433 polarion`

2. Create the DBLink extension.
   a. If your PostgreSQL version is 9.1 or later, just execute the following SQL command: `CREATE EXTENSION dblink;
   b. If your PostgreSQL version is 9.0 or earlier, use the psql tool to set up the extension using the following command:
      - If not connected to psql: `psql -p <port> polarion -f dblink.sql`
      - If already connected: `\i dblink.sql;`
3. To verify that the dblink extension was successfully installed, execute this SQL query in your database client:

```sql
SELECT p.proname
FROM pg_catalog.pg_namespace n
JOIN pg_catalog.pg_proc p
ON p.pronamespace = n.oid
WHERE n.nspname = 'public';
```

It should return rows containing `db_link` as a prefix.

4. Install the PLPGSQL language by executing the following SQL command:

```sql
CREATE LANGUAGE plpgsql;
```

5. Quit the `psql` utility (`\q`) and repeat the above steps for the `polarion_history` database.

6. If you did not run the initialization of the database with `--auth-host=md5`, open `[PG_DATA]/pg_hba.conf` and change all the host entries with trust to md5, then restart PostgreSQL. (In our examples, `[PG_DATA]` was `/opt/polarion/data/postgres-data` (the location on which the database was initialized).

E) Optimize the PostgreSQL Database:

Optimization is the same as described for automated installation.

(See After installation: optimizing the PostgreSQL database.)

F) Configure Polarion to use PostgreSQL database:

Add the following property to the `polarion.properties` system configuration file:

```properties
com.polarion.platform.internalPG=polarion:<password>@<hostname>:<port>
```

For example:

```properties
com.polarion.platform.internalPG=polarion:polarion@localhost:5433
```

If you have the polarion and polarion_history databases running on different servers, you can use these two properties rather than the above property:

```properties
com.polarion.platform.sql.internalDB.head.url=jdbc:postgresql://<hostname>:<port>/polarion?user=<userName>&password=<password>
com.polarion.platform.sql.internalDB.historical.url=jdbc:postgresql://<hostname>:<port>/polarion_history?user=<userName>&password=<password>
```
//<hostname>:<port>/polarion_history?user=<userName>&password=<password>

Caution:
Each property should be written in a single line.

After you have set the system properties, reindex your Polarion installation so that the PostgreSQL database will be populated. Using any SQL client after the reindex, you should see the database filled in the `polarion` schema.

G) (Optional) Allow remote connections to the PostgreSQL server:

By default the PostgreSQL database is only available on `localhost` after a fresh install. If you need to have access from outside the host machine, please follow these steps to allow remote access:

1. Change the listener addresses in the `postgresql.conf` file. This file can be found in the PostgreSQL data folder, the one you have defined under Starting/Stopping PostgreSQL. In this file, change: 
   ```
   #listen_addresses='localhost' to listen_addresses='*'
   ```
2. In the same folder, you will find the `pg_hba.conf` file. Open it in a text editor and add the following line:
   ```
   host all all 0.0.0.0/0 md5
   ```

Warning:
If you allow for external connections to the database, an external user has blanket access to the entire database content and effectively to all Polarion data - all projects, the HEAD revision as well as history. There is no further access control. All Polarion user names will also be visible. (Polarion user passwords will not be visible as those are not stored in a database or SVN.) Access is read-only.

Administrators are advised to consider carefully before opening database access in a system in use on a production level.

H) Check Shared Buffers Configuration

Check that the value of the `shared_buffers` parameter in `postgresql.conf` does not exceed the available shared memory configured by the `SHMMAK` kernel parameter.

Install Polarion

After the required third-party software is installed and configured, you are ready to install Polarion.

The Linux distribution comes in a ZIP archive. The archive file name includes the version number and the word `linux`. For example: `PolarionALM_nnnn_linux.zip` (where `n` is a number from 0-9).
Installation steps

1. Be sure that all required third-party components are properly installed and configured.

2. Unpack the distribution ZIP archive to a temporary location.

3. Change current directory to **Polarion**.

4. If SELinux is bundled with/installed on your Linux OS, make sure it is not activated. Check the status of SELinux with the **/usr/sbin/sestatus -v** command.

5. Create a Unix system account named **polarion** for the Polarion server.

6. Review the path variables defined in **libinstall/default.sh**. Set up **v_web_user** and **v_web_group**.

7. Execute (under root):
   ```
   chmod +x manual_install.sh; ./manual_install.sh
   ```

   This will install Polarion into the standard **/opt/polarion** location. Data will be installed into the standard **/opt/polarion/data** location.

Setting permissions

After manual installation you need to set up permissions on unpacked folders.

The standard permissions are:

- Read only for everybody, owner and group root: root on the following folders and their subfolders:
  - **/opt/polarion/polarion**
  - **/opt/polarion/maven**

- Writeable on the **/opt/polarion/datafolder** and its subfolders.

For "**polarion**":**web user**, where **polarion** is a system account that you must create manually and **web user** is user used by Apache.

For more details, see the `setPermissions()` function in **[UnpackDir]/libinstall/helper_functions.sh**.

Notes on a manual installation

If Polarion is already installed, any user data (repository, builds, reports) will not be overwritten. However, it is recommended to back up any changed files in **/opt/polarion** (typically **polarion.properties** and any customized shell scripts), because they will be overwritten.

Possible changes include:
• Change the value of the `host.name` parameter from `localhost` to the host name specified in `httpd.conf`.

• Check that the repository alias in the `svn.url` parameter corresponds to the one in the `httpd.conf` file. (See Required Configuration.)

• Check that the correct paths to the `passwd` and `access` files are specified in the relevant `svn.passwd.file` and `svn.access.file` parameters.

• You may need to reconfigure the `htpasswd.path` parameter and set it to `/usr/sbin/htpasswd2` instead of the usual `/usr/local/apache2/bin/htpasswd`.

• Check Apache's htpasswd utility path in the `htpasswd.path` parameter.

• Check that the port specified in the `tomcat.ajp13-port` parameter corresponds to the port specified in the `worker.worker1.port` parameter of the `workers.properties` file.

• Specify the correct SMTP host in the `announcer.smtp.host` parameter.
Configure Polarion

1. Execute commands from installation temporary directory to create new Subversion repository:
   
   ```bash
   svnadmin create /opt/polarion/data/svn/repo -fs-type fsfs
   chown -R "webuser"::"webgroup" /opt/polarion/data/svn/repo
   ```
   
   The items `webuser` and `webgroup` are the user and group under them on the Apache server.

2. Import either the production data or demo data configuration. You cannot import both. You can only execute both scripts on a fresh repository.
   
   ```bash
   [POLARION]/bin/polarion.init init
   or
   [POLARION]/bin/polarion.init demo
   ```

   **Note:**
   
   Parameters could be different on your OS. For example **User** and **Group** directive values from `httpd.conf`: The **Group** value in the standard Apache distribution is `-1`, the numerical **group ID** `-1`, can't be used. Use `apache/apache` as **User** and **Group** instead.

   **Warning:**
   
   Once the ([POLARION]/bin/polarion.init init, or [POLARION]/bin/polarion.init demo) scripts have been executed, it is not possible to install demo data without deleting the subversion repository in `/srv/polarion/data/svn/repo`.

3. Before using the system for production, see *Securing the Polarion activation application* and perform the recommended configuration as described.

Next steps

After completing a manual installation, review the following topics.

- **System Startup and Shutdown**
- **After Installation**
- **Appendix: Default Users and Groups**
- **Appendix: Enabling Email Notifications**
18. Supported Microsoft Office versions

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>LiveDocs (2011)</td>
<td>Word Import &amp; Round-trip</td>
<td>.docx</td>
<td>DOCX</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>Excel Import &amp; Round-trip</td>
<td>.xlsx</td>
<td>XLSX</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>Word Export</td>
<td>.doc</td>
<td>XML</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
</tr>
</tbody>
</table>

- Supported ✓
- Unsupported ✗

- Beginning with Polarion 18, Microsoft Office 2007 is no longer supported in Polarion LiveDocs™.

Caution:
(The only exception is Polarion's internal templates. If updated in Microsoft Word, they should still be saved in the 2007 .docx format to ensure that content like shapes import/export as expected.)

- Beginning with Polarion version 2014-SR1, Microsoft Office 2003 is no longer supported in Polarion LiveDocs™.

- Prior to Polarion version 2011, Live Documents referred to Microsoft Office Word and Excel documents based on special document templates that could define and store Polarion Work Items. Beginning with version 2011, the technology was completely refactored, but backward compatibility was maintained. Beginning with Polarion version 2014-SR1, support for this legacy format was dropped completely and the feature was renamed as LiveDocs.

- Beginning with Polarion 19, Microsoft Office 2010 and 2013 are no longer supported in Polarion LiveDocs™.
18. Supported Microsoft Office versions
This software and related documentation are proprietary to Siemens Digital Industries Software.

© 2019 Polarion AG.
Polarion is a registered trademark of Polarion AG. Polarion ALM, Polarion REQUIREMENTS, Polarion QA and Polarion VARIANTS are trademarks or registered trademarks of Polarion AG.

Siemens and the Siemens logo are registered trademarks of Siemens AG. NX, Solid Edge, and Teamcenter are trademarks or registered trademarks of Siemens Digital Industries Software or their subsidiaries in the United States and in other countries. All other trademarks, registered trade marks, or service marks belong to their respective holders.

© 2019 Polarion AG